Modellierung und Analyse individuellen Konsumentenverhaltens mit probabilistischen Holonen

Dissertation

zur Erlangung des Grades
des Doktors der Ingenieurwissenschaften
der Naturwissenschaftlich-Technischen Fakultäten
der Universität des Saarlandes
von

Arndt Stephan Georg Schwaiger

Saarbrücken im Jahr 2006

Tag des Kolloquiums:

8. September 2006

Dekan der Fakultät 6 – Naturwissenschaftlich-Technische Fakultät I:

Prof. Dr. Thorsten Herfet

Prüfungsvorsitzender:

Prof. Dr. Reinhard Wilhelm

Berichterstatter:

Prof. Dr. Jörg Siekmann

Prof. Dr. Joachim Hertel

Prof. Dr. Anthony Jameson

Akademischer Beisitzer:

Dr. Christoph Jung

Kurze Zusammenfassung

Der Schwerpunkt dieser Arbeit liegt in der Entwicklung eines agentenbasierten, probabilistischen Konsumentenverhaltensmodells zur Repräsentation und Analyse individuellen Kaufverhaltens. Das Modell dient zur Entscheidungsunterstützung im Handel und speziell im Customer Relationship Management (CRM). Als Modellgrundlage wird eine Klasse probabilistischer Agenten eingeführt, die sich zu Holonen zusammenschließen können und deren Wissensbasen erweiterte Bayes'sche Netze (Verhaltensnetze) sind. Mit Hilfe probabilistischer Holone werden Kundenagenten entwickelt, die einzelne reale Kunden modellieren. Dazu werden kundenindividuelle Verhaltensmuster unter Berücksichtigung von Domänenwissen aus historischen Kundendaten extrahiert und als nichtlineare Abhängigkeiten zwischen Einflussfaktoren und artikelbezogenen Kundenreaktionen in Verhaltensnetzen repräsentiert. Ein Kundenagent ist dabei ein Holon aus mehreren so genannten Feature-Agenten, die jeweils einzelne Kundeneigenschaften repräsentieren, entsprechende Feature-Verhaltensnetze verwalten und durch Interaktion das Gesamtverhalten des Kunden bestimmen. Die Simulation des Verhaltens besteht aus der Ermittlung von Kundenreaktionen auf vorgegebene Einkaufsszenarien mit Hilfe quantifizierbarer probabilistischer Schlussfolgerungen. Kundenagenten können sich durch Holonisierung zu Kundengruppenagenten zusammenschließen, die unterschiedliche Aggregationen des Kaufverhaltens der Gruppenmitglieder repräsentieren. Zur Bestimmung gleichartiger Kunden werden auf Basis der Verhaltensnetze mehrere Ähnlichkeitsanalyseverfahren sowie verhaltensbezogene Ähnlichkeitsmaße zum Vergleich des dynamischen Kaufverhaltens entwickelt. Bestehende Klassifikations- und Clusteringverfahren werden anschließend so erweitert, dass sie neben klassischen Attributvektoren verhaltensnetzbasierte Repräsentationen als Vergleichsgrundlage verwenden können. Darüber hinaus werden Verfahren zur Zuordnung anonymer Kassenbons zu vorgegebenen Kundengruppen entwickelt, um Ergebnisse von Kundensimulationen auf die Gesamtheit der anonymen Kunden eines Unternehmens übertragen zu können. Nutzen und Qualität der entwickelten Modelle, Verfahren und Maße werden mit Hilfe einer umfangreichen Software-Implementierung anhand mehrerer Anwendungsbeispiele aus der Praxis demonstriert und in einigen Fallstudien evaluiert – basierend auf realen Daten eines deutschen Einzelhandelsunternehmens.

Short abstract

The focus of this work is the development of an agent-based, probabilistic model for representing and analysing individual consumer behaviour. The model provides a basis for decision making in marketing and especially in customer relationship management (CRM). As foundation of the model, a class of *probabilistic agents* is introduced. These agents can be merged to *holonic agents* (*holons*) and have probabilistic knowledge bases adapted from *Bayesian networks* (*behaviour networks*). An individual customer is modelled as a *customer agent* which is a probabilistic holon consisting of several *feature agents*. A feature agent represents a particular property (feature) of the customer's behaviour and encapsulates appropriate feature-related behaviour networks. The total behaviour of a customer agent is determined by interaction of its feature agents. Individual behaviour patterns of a customer are extracted from real data – in consideration of given domain knowledge – and are represented within behaviour networks as non-linear dependencies between influencing factors and the customer's product-related reactions. Behaviour simulation is realised by evaluation of expected reactions of customers on given shopping scenarios based on quantifiable, probabilistic reasoning. Customer agents are able to join to *customer group agents* which represent different behaviour aggregations of their members. Based on behaviour networks, several behaviour-related methods of analysis as well as distance measures are developed to identify homogeneous customers on the basis of their dynamic shopping behaviour. Subsequently, existing vector-based methods of classification and clustering are extended by these behaviour-related methods and measures. In addition, methods are developed to assign anonymous receipts to given customer groups in order to extent customer-related simulation results to anonymous customers of a company. Benefits and quality of the developed models, methods and measures, which are implemented within a complex software system, are shown by practical examples and evaluated in several case studies – based on real data from a German retailer.

Danksagung

Als erstes möchte ich Herrn Prof. Dr. Jörg Siekmann für die Betreuung meiner Arbeit danken. Meine Mitarbeit in dem von ihm geführten Forschungsbereich *Deduktion und Multiagentensysteme* am *Deutschen Forschungszentrum für Künstliche Intelligenz* (*DFKI*) hat es mir ermöglicht, das interessante Thema „agentenbasierte Modellierung und Analyse individuellen Konsumentenverhaltens“ im Rahmen einer Doktorarbeit in einem interdisziplinären Umfeld zu bearbeiten.

Ich danke Herrn Prof. Dr. Joachim Hertel, der mir die Einbeziehung praktischer Fragestellungen aus dem Handel in meine Forschungsarbeiten und deren Evaluation anhand realer Daten ermöglicht hat. Ich danke ihm darüber hinaus dafür, dass er sich zur Begutachtung meiner Arbeit bereit erklärt hat.

Mein Dank gilt ebenso Herrn Prof. Dr. Anthony Jameson für seine spontane Bereitschaft die vorliegende Dissertation als zweiter Gutachter zu beurteilen sowie Herrn Prof. Dr. Reinhard Wilhelm dafür, dass er den Vorsitz des Prüfungsausschusses übernommen hat.

Ich bedanke mich bei allen Mitarbeitern, Studenten und Praktikanten des Forschungsbereiches *Deduktion und Multiagentensysteme* am *Deutschen Forschungszentrum für Künstliche Intelligenz* für die angenehme und produktive Arbeitsatmosphäre. Mein Dank gilt dabei vor allem Herrn Dr. Klaus Fischer und Herrn Dr. Matthias Klusch für die Möglichkeit, in einer erfolgreichen und international anerkannten Forschungsgruppe arbeiten zu dürfen. Ich bedanke mich darüber hinaus bei meinen mehrjährigen Zimmerkollegen und Freunden Herrn Björn Stahmer und Herrn Sven Jacobi für die angenehme Zusammenarbeit.

Mein Dank gilt allen, die am Verbundprojekt *SimMarket* mitgearbeitet haben, vor allem den Mitarbeitern, Studenten und Praktikanten des *DFKI*, der *Dacos Software GmbH*, des *Instituts für Handel und internationales Marketing* (*H.I.MA.*) unter der wissenschaftlichen Leitung von Herrn Prof. Dr. Joachim Zentes sowie der drei Handelsunternehmen *Globus*, *tegut…* und *dm-drogerie markt*.

Gewidmet ist diese Arbeit meinen Eltern, Großeltern und Freunden sowie besonders meiner Lebensgefährtin Barbara Räsch für ihre Unterstützung und Freundschaft.

Arndt Stephan Georg Schwaiger im September 2006

Inhaltverzeichnis

1 Einleitung

1.1 Motivation

Der Einzelhandel in Deutschland befindet sich seit einigen Jahren in einer strukturellen und konjunkturellen Krise [ZBSK04]. Der Preisdruck der Discounter nimmt zu und durch die zunehmende Globalisierung drängen immer mehr internationale Anbieter in bisher von lokalen Unternehmen dominierte Regionen vor. Funktion, Qualität und Preis vieler Produkte und Dienstleistungen sind aufgrund des gesättigten Marktes in den meisten westlichen Industrienationen nahezu substituierbar [ABKW02]. Gleichzeitig haben sich sowohl das Anspruchsniveau als auch die Erwartungshaltung der Konsumenten deutlich erhöht [ZBSK04]. Um wettbewerbsfähig zu bleiben, sind Unternehmen gezwungen, effektivere *Marketing-Strategien* zu entwickeln. Eine gute Strategie bedeutet vor allem eine optimierte Auswahl von Produkten und Dienstleistungen sowie konkurrenzfähige Preise und wirksame Werbemaßnahmen, deren Mix optimal auf ausgewählte Zielgruppen ausgerichtet ist.

Allerdings gestaltet sich die Entwicklung geeigneter Strategien aufgrund der enormen Komplexität der Handlungsmöglichkeiten als schwierig: Ein SB-Warenhaus führt durchschnittlich ca. 28.500 verschiedene Artikel [SW04], die aus einer Menge von bis zu einer halben Million möglicher Produkte bzw. Produktvariationen ausgewählt werden können. Darüber hinaus haben Änderungen des Sortiments und der Preise, Platzierungen und Bewerbungen einzelner Produkte oft auch deutliche Effekte auf die Abverkäufe einer Vielzahl anderer Artikel. Es gilt, Sortimente und Maßnahmen der Konkurrenz sowie eine Vielzahl weiterer Einflussfaktoren zu beachten, wie beispielsweise die Ausprägung der Kundentypologie, regionale Besonderheiten, Trends, saisonale Einflüsse, besondere Events oder meteorologische Einflüsse.

Während einerseits allein die enorme Datenmenge und die Anzahl der Auswahl- und Kombinationsmöglichkeiten die Fähigkeiten menschlicher Informationsverarbeitung übersteigen, haben andererseits entsprechende rechnergestützte Verfahren zur Entwicklung geeigneter Strategien und Maßnahmen den Nachteil, dass sie das Verhalten der Kunden meist zu „global“ betrachten. Die Verfahren modellieren das Konsumverhalten als Gesamtheit, ohne dessen Entstehung - d. h. das komplexe Zusammenspiel einer Vielzahl von kundenindividuellen Einzelentscheidungen - genauer verstehen zu können. Daher können sie gerade bei kundengruppen- bzw. kundenindividuellen Fragestellungen, die heute zur Entwicklung optimaler Strategien immer entscheidender werden, nur ungenügende Unterstützung bieten, beispielsweise, wie sich konkrete Maßnahmen auf einzelne Kundengruppen oder individuelle Kunden auswirken.

Die Modellierung des individuellen Konsumentenverhaltens, d. h. des Verhaltens von Menschen beim Kauf und Konsum von wirtschaftlichen Gütern [KRW03], ist aus diesen Gründen ein viel versprechender Ansatz zur Entwicklung geeigneter Strategien. Je genauer ein Unternehmen seine Kunden und deren Verhaltensweisen kennt, desto optimaler können Sortiments-, Preis-, Werbe- und Platzierungsmaßnahmen gestaltet werden, die maßgeblich den Erfolg des Unternehmens bestimmen. Das heißt, die Kenntnis des Kundenverhaltens sollte die Grundlage jeder Marketing-Strategie sein [SW04], was bisher jedoch mangels geeigneter Daten sowie entsprechender Modelle und Verfahren nicht möglich war.

Klassische Verhaltensmodelle lassen sich in Stimulus-Response- (SR) und Stimulus-Organism-Response-Modelle (SOR) unterteilen [KRW03]. Während SR-Modelle die Reaktionen der Kunden auf Einflussfaktoren modellieren, wobei der Kunde als „Blackbox“ angesehen wird, versuchen SOR-Modelle zusätzlich psychologische Prozesse abzubilden. SOR-Modelle haben den Nachteil, dass sich die Modellierung der kundeninternen Prozesse nur mit Hilfe theoretischer Modelle durchführen lässt. Dadurch können in der Praxis nur in Ausnahmefällen quantifizierbare Aussagen ermittelt werden. SR-Modelle stützen sich zwar ausschließlich auf beobachtbare Reaktionen des Kunden auf messbare Einflüsse, allerdings eignen sich die bisherigen Modelle nur unzureichend zur quantifizierbaren Simulation des Kaufverhaltens realer Kunden bezüglich konkreter „Was-wäre-wenn-Szenarien“.

Ein geeignetes Kundenmodell sollte individuelles Kundenverhalten aus historischen Daten extrahieren, abbilden und anschließend in unterschiedlichen Situationen quantifizierbar simulieren können. Es sollte dabei eine Vielzahl von Einflussfaktoren und deren gegenseitigen Beeinflussungen modellieren sowie die eventuell nichtlinearen Abhängigkeiten zwischen Einflüssen und Reaktionen quantitativ repräsentieren können. Natürlich ist es auf diese Weise in absehbarer Zeit nicht möglich, ein „perfektes“ Modell des Kaufverhaltens realer Personen zu erstellen. Aber durch den Einsatz heutiger Warenwirtschaftssysteme [HJ99] und elektronischer Verfahren des Geldverkehrs, durch die Einführung von Kundenkarten sowie durch die Benutzung von *Data Warehouses* und der Anwendung von *Data Mining* werden reichhaltige Informationen über das Kaufverhalten einzelner Kunden und Kundengruppen gesammelt. Aus diesen Informationen können mit Hilfe geeigneter maschineller Lernverfahren immer detailliertere kundenindividuelle Verhaltensweisen extrahiert werden, die in einem Kundenmodell vereint werden können.

Ein Kundenmodell eignet sich als Basis für das *Customer Relationship Management* (*CRM*), das in vielen Unternehmen eine immer wichtigere Rolle bei der Entwicklung von (nicht nur rein kundenbezogenen) Marketing-Strategien spielt [ABKW02]. Das CRM stellt die für das Unternehmen *wertvollen* Kunden in den Mittelpunkt und steuert entsprechende kundenorientierte Prozesse abteilungsübergreifend.

Grundlage ist der Aufbau und Unterhalt einer Kundendatenbank, damit allen

Unternehmensbereichen fundierte, aktuelle, vollständige und entscheidungsrelevante Informationen über Kunden zur Verfügung stehen [SW04]. Die Datenbank kann nun um Kundenmodelle erweitert werden, die Grundlage für kundenindividuelle und kundengruppenspezifische Analysen und Prognosen sind. Diese dienen als Basis kundenorientierter Strategien [ZJM99], die optimal auf ausgewählte Zielkundengruppen angepasst werden können, da sich im Vorfeld deren Reaktionen auf die geplanten Maßnahmen evaluieren lassen. Ebenso lassen sich im Vorhinein die unterschiedlichen Reaktionen verschiedener Kundengruppen auf geplante Aktionen abschätzen und die Entscheidungsfindung bei kundenindividueller Ansprache oder kundenbezogenen Aktionen unterstützen, wie beispielsweise beim *Direct Marketing* bzw. *One To One Marketing* [ZJ00].

Ein detailliertes verhaltensbasiertes Kundenmodell dient darüber hinaus als Grundlage für Ähnlichkeitsanalysen zwischen einzelnen Kunden und Kundengruppen, wodurch sich die Qualität der Kundenklassifikation deutlich verbessern lässt, da Kunden nun anhand ihres Kaufverhaltens verglichen werden können. Bisherige Verfahren und Kundentypologien stellen soziodemographische Merkmale und einfache statistische Ausprägungen in den Vordergrund [ABKW02], die für viele verhaltensbezogene Fragestellungen zu „oberflächlich“ sind.

1.2 Ziel der Arbeit

Das Ziel meiner Arbeit ist die Entwicklung und Implementation eines Modells individuellen Konsumentenverhaltens, das zur Unterstützung der Entscheidungsfindung und zur Optimierung von Strategien im Marketing und speziell im Customer Relationship Management dient. Dabei soll sowohl die Modellierung des Kaufverhaltens einzelner Personen als auch die des Gesamtverhaltens von Kundengruppen ermöglicht werden. Die Abbildung des Kaufverhaltens soll ebenfalls skalierbar sein, von der Abbildung einzelner Verhaltensaspekte, wie beispielsweise der Preissensibilität, bis hin zur Modellierung von Kaufentscheidungen bezüglich komplexer Einkaufsszenarien. Das Kaufverhalten soll aus realen Daten extrahiert und so exakt wie möglich abgebildet werden. Ziel ist es daher, möglichst viele messbare Einflussfaktoren sowie deren eventuell nichtlineare Abhängigkeiten zu modellieren. Die Modelle und darauf aufbauende Verfahren sollen das Kaufverhalten von Kunden und Kundengruppen bezüglich vorgegebener Szenarien quantifizierbar simulieren und analysieren, um Kundenreaktionen auf geplante Maßnahmen prognostizieren und Kunden aufgrund ihres Verhaltens klassifizieren zu können.

1.3 Gliederung

Kapitel 2 führt anfangs in Modelle, Verfahren und Konzepte der Künstlichen Intelligenz ein, und beschreibt anschließend die Architektur der darauf aufbauenden probabilistischen Holone, an deren Entwicklung ich maßgeblich beteiligt war und die Grundlage meines Kundenmodells sind.

In Kapitel 3 erläutere ich, nach einem kurzen Überblick über bekannte Erkenntnisse und Modelle des Konsumentenverhaltens, die Modellierung einzelner Kunden durch spezielle probabilistische Holone (Kundenagenten), deren Kaufverhalten aus historischen Informationen extrahiert und in probabilistischen Verhaltensnetzen gespeichert werden.

Kapitel 4 beschreibt die Vorgehensweise zur szenariobasierten Simulation des kundenindividuellen Kaufverhaltens mit Hilfe von Inferenz-Algorithmen für probabilistische Verhaltensnetze.

Kapitel 5 beschäftigt sich mit der Bildung von Kundengruppen, deren Modellierung auf unterschiedlichen Arten der Holonisierung einzelner Kundenagenten basiert. Kern ist die Entwicklung verhaltensbezogener Ähnlichkeitsmaße für probabilistische Verhaltensnetze, die als Grundlage für Ähnlichkeitsanalysen, Klassifikationen und Clusterings dienen.

In Kapitel 6 stelle ich eine konkrete Implementierung des probabilistischen Kundenmodells im Rahmen des Verbundprojektes SimMarket am Deutschen Forschungszentrum für Künstliche Intelligenz (DFKI) vor. Ich präsentiere konkrete Anwendungsbeispiele der Kundenmodellierung und zeige anhand von Evaluationen die Qualität der Kundensimulation und der entwickelten verhaltensbezogenen Ähnlichkeitsmaße. Anschließend gebe ich einen Überblick über die wichtigsten verwandten Arbeiten und Forschungsprojekte.

Kapitel 7 fasst die Forschungsbeiträge meiner Arbeit zusammen und gibt einen Überblick über offene Fragen, sinnvolle Erweiterungen und Einsatzmöglichkeiten des vorgestellten Kundenmodells.

Anhang A gibt eine kurze Einführung in Marketing und Customer Relationship Management, während in Anhang B zwei bekannte Kundengruppentypologien präsentiert werden.

Anhang C zeigt abschließend einen Überblick über die Komplexität der in meiner Arbeit vorgestellten Verfahren und Maße.

2 Probabilistische Holone

Dieses Kapitel beschreibt Konzepte und Verfahren der Künstlichen Intelligenz, die Grundlage meines Kundenmodells sind. Ziel ist es, das Verhalten realer Kunden mit Hilfe so genannter *probabilistischer Holone* zu modellieren, speziellen Software-Agenten, an deren Entwicklung ich maßgeblich beteiligt bin.

Kapitel 2.1 führt in das Konzept der Software-Agenten ein. Dabei handelt es sich um Software-Objekte mit speziellen Eigenschaften, die in verschiedene Komplexitätsklassen unterteilt werden (2.1.1). Je nach Klasse und Aufgabenstellungen besitzen Agenten unterschiedliche Komponenten und Funktionalitäten, die durch verschiedene Architekturen umgesetzt werden können (2.1.2). Einzelne Agenten können Mitglieder von Agentengesellschaften sein, den *Multiagentensystemen* (*MAS*) (2.1.3).

In Kapitel 2.2 erläutere ich das Konzept der *holonischen Multiagentensysteme* (*HMAS*), wobei *Holone* aus dem Zusammenschluss einer Menge von Agenten entstehen.

Kapitel 2.3 gibt einen Überblick über den Bereich der *Bayes'schen Netze*, die in Form von *Verhaltensnetzen* Grundlage der Wissensrepräsentation und Schlussfolgerungsprozesse der probabilistischen Agenten und Holone sind, die ich anschließend in Kapitel 2.4 einführe.

2.1 Software-Agenten und Multiagentensysteme

Die Erforschung von Agenten und Multiagentensystemen ist Teil der Künstlichen Intelligenz (KI), einem jungen Forschungsgebiet, das neben der allgemeinen Erforschung von Intelligenz die Entwicklung intelligenter Systeme und die Modellierung menschlichen Schlussfolgerns und Verhaltens zum Ziel hat. Letzteres ist stark mit dem *Turing-Test* [TA50] verbunden, bei welchem menschliche Probanten einen Dialog mit einem Computerprogramm führen und nicht erkennen sollen, ob sie sich mit einer Person oder einem Programm unterhalten. Die Entwicklung intelligenter Systeme basiert auf *rationalem Verhalten*, wobei die Erforschung von Agenten und Multiagentensystemen im Vordergrund steht, die in den siebziger Jahren des 20. Jahrhunderts begann (siehe [CH98] und [FJ01]).

Für den Begriff des „Agenten" bzw. „Software-Agenten" existieren verschiedene Definitionen, allerdings ist keine exakt und allgemeingültig [FG96]. Umgangssprachlich bezeichnet „Agent" *jemanden, der im fremden Auftrag selbstständig handelt*, während ein Software-Agent in der Regel ein *autonomes Programm* ist [BHD98], dessen Eigenschaften unterschiedlich definiert werden. Jennings und Wooldridge definieren einen Agenten als spezielles Computersystem, das innerhalb seiner Umgebung autonom agieren kann, um seine

Ziele zu erreichen [JW98]. Diese Definition kommt der von Russell und Norvig sehr nahe, die mit „Agenten“ alles bezeichnen, was seine Umgebung durch *Sensoren* wahrnehmen und durch so genannte *Effektoren* verändern kann [RN03].

Das Konzept der Software-Agenten eröffnet eine neue Sichtweise auf die Architektur und Funktionsweise von Computerprogrammen, die einer Erweiterung der *objektorientierten* hin zur *agentenorientierten* Software-Entwicklung entspricht (siehe z. B. [SY93]). Agenten sind in diesem Zusammenhang intelligente und autonome Software-Objekte, Software-Komponenten und Computerprogramme. Oft bezeichnet man bei der Mensch-Computer-Interaktion die menschlichen Akteure ebenfalls als Agenten [BHD98]. Agentenbasierte Programme können nach Eingang eines Auftrages bzw. einer Anfrage selbstständig handeln, um die entsprechenden Aufgaben zu erfüllen, d. h. sie können eigenständig auf Signale der Umwelt reagieren.

Neben den Eigenschaften Selbstständigkeit und Autonomie, zeichnen sich Software-Agenten durch weitere spezielle Eigenschaften aus, die je nach Typ bzw. Klasse des Agenten unterschiedlich stark ausgeprägt sein können. Die wichtigsten Eigenschaften sind:

- Mobilität
- Reaktivität
- Deliberativität
- Kooperation
- Kommunikation
- Soziales Verhalten
- Rationalität
- Intelligenz
- Lernfähigkeit

Eine Eigenschaft, die stark mit der Autonomie von Agenten verbunden ist, ist die *Mobilität*. Sie ermöglicht es Agenten, sich in der realen Welt (z. B. Roboter) oder virtuell zwischen verschiedenen Rechnern zu bewegen, beispielsweise aus Performanzgründen oder im Rahmen einer Informationssuche [MF98].

Unter *Reaktivität* versteht man die Fähigkeit, ausschließlich auf Basis simpler Verhaltensregeln auf wahrgenommene Umwelteinflüsse zu reagieren, d. h. der Agent führt, aufgrund einer bestimmten Sensorinformation, unmittelbar eine vorgegebene Aktion aus. Im Gegensatz dazu verfügen *deliberative* Agenten über eine Repräsentation ihrer Umwelt und ihres Zustandes und können symbolische Schlussfolgerungen ziehen [NS76].

Die Fähigkeit zur *Kooperation* und zum *sozialen Verhalten* ist eine entscheidende Grundlage zur Zusammenarbeit mehrerer Agenten (siehe Kapitel 2.1.3) und damit auch zur Interaktion zwischen Mensch und Agenten. Eine wichtige Voraussetzung zur Kooperation ist die *Kommunikation* zwischen Agenten bzw. zwischen Agenten und menschlichen Benutzern. Dabei regeln spezielle Kommunikations- bzw. Kooperations-Protokolle (wie beispielsweise FIPA oder KQML) die Möglichkeiten des Informationsaustauschs bzw. der Kooperation.

Die wichtigen Eigenschaften *Rationalität* und *Intelligenz* ermöglichen es einem Agenten, *vernünftige* Entscheidungen zu treffen. So sollte ein Agent nur sinnvolle Ziele verfolgen, die

in der aktuellen Situation und unter den gegebenen Umständen erreichbar sind und deren Nutzen größer als der Aufwand ist. Rationalität setzt Planungs- bzw. Entscheidungsprozesse sowie die Möglichkeit der Bewertung von Situationen voraus. Intelligente Agenten können in vielen Fällen ihr Verhalten verbessern, d. h. aus Erfahrungen neue Schlüsse ziehen und aus Fehlern lernen (*Lernfähigkeit*).

2.1.1 Klassen und Typen von Agenten

Je nach Existenz, Ausprägung bzw. Kombination der oben beschriebenen Eigenschaften werden Agenten in verschiedene Klassen eingeteilt. Corsten und Gössinger differenzieren zwischen *primitiven*, *technischen*, *kognitiven* und *sozialen* Agenten [CG97]. Russell und Norvig hingegen unterscheiden anhand des Grades der Rationalität folgende vier Agentenklassen [RN03]:

1. Einfache Reflex-Agenten,
2. Modellbasierte Agenten,
3. Zielgerichtete Agenten und
4. Nutzenorientierte Agenten.

Einfache *Reflex-Agenten* reagieren auf aktuelle Umweltinformation ausschließlich anhand von fest vorgegebenen *Verhaltensregeln*. Die Verhaltensregeln folgen einer *Wenn-Dann-Struktur*: *Wenn* bestimmte Kombination von Umweltzuständen wahrgenommen *Dann* führe bestimmte Aktionen aus.

Als Erweiterung zu einfachen Reflex-Agenten besitzen *modellbasierte Agenten* eine interne Repräsentation der Umwelt und haben Kenntnis über die Auswirkungen eigener Aktionen. Die Auswahl der folgenden Aktion basiert auf dem aktuellen Zustand des internen Umweltmodells und der aktuellen Sensorinformation.

Zielgerichtete Agenten erweitern modellbasierte Agenten um konkrete Zielsetzungen, die sie mit Hilfe von *Planungsverfahren* zu erreichen versuchen.

Nutzenorientierte Agenten bewerten anhand einer *Nutzenfunktion* (*utility function*) verschiedene Zielsituationen, wobei die Nutzenfunktion interne sowie externe Zustände bzw. Sequenzen von Zuständen auf reelle Zahlen abbildet. Auf diese Weise kann ein Agent mit Hilfe der Nutzenfunktion verschiedene Pläne und Ziele evaluieren, um die Folge von Aktionen wählen zu können, die erwartungsgemäß den größten Nutzen bringt.

Neben den beschriebenen Klassen können Agenten auch aufgrund ihres Zwecks bzw. ihrer Tätigkeit typisiert werden. So suchen *Informationsagenten* selbstständig nach vorgegebenen Informationen (z. B. Internet, Web-Services oder Datenbanken) und tauschen sich mit anderen Informationsagenten aus. *Assistenz-Agenten* versuchen, Anwender bei ihrer Arbeit zu unterstützen, indem sie selbstständig erkennen, was diese planen und daraufhin geeignete Vorschläge anbieten oder im Hintergrund entsprechende Vorbereitungen durchführen. Bei

Online-Auktionen werden immer häufiger so genannte *Bieter-Agenten* verwendet, die nach vorgegebenen Regeln im Auftrag von menschlichen Benutzern bei Auktionen mitbieten.

2.1.2 Architekturen von Agenten

Im Allgemeinen besitzen Agenten folgende drei Architekturkomponenten:

1. *Sensorkomponente*,
2. *Aktionskomponente* (*Effektoren*) sowie die
3. *Entscheidungskomponente* (eventuell inklusive einer Wissensbasis).

Die *Sensorkomponente* setzt sich aus einer Menge von Sensoren zusammen, mit deren Hilfe ein Agent seine Umwelt wahrnehmen kann. Agenten können somit die Fähigkeit besitzen, akustische oder visuelle Informationen aufzunehmen oder in einem Rechnernetzwerk Informationen von einem Server zu empfangen.

Die *Aktionskomponente* dient dem Ausführen von Aktionen mittels so genannter *Effektoren*, die aus Hard- oder Software bestehen können. *Hardware-Effektoren* sind Hilfsmittel, mit denen ein Agent seine Umwelt manipulieren bzw. seinen eigenen Zustand verändern kann, wie z. B. ein Roboter-Arm bzw. ein motorgesteuerter Antrieb. *Software-Effektoren* führen bestimmte Methoden aus einer vorgegebenen funktionalen Bibliothek durch, die den Zustand des Agenten bzw. den Zustand des Umweltmodells verändern.

Die *Entscheidungskomponente* kann je nach Komplexitätsgrad der Agentenklasse unterschiedlich umfangreich ausfallen: von einfachen Wenn-Dann-Regeln der Reflex-Agenten bis hin zu komplexen Modellen zur Repräsentation der Umwelt und der eigenen Zustände sowie Planungsmodulen der ziel- bzw. nutzenorientierten Agenten. Dabei kann sowohl das Wissen eines Agenten über seine Umwelt als auch über seine internen Zustände und Verhaltensmuster in einer *Wissensbasis* (*Knowledge Base* bzw. *KB*) gespeichert werden.

2.1.2.1 BDI-Agenten

BDI-Agenten versuchen mentale Zustände durch technische Äquivalente von *Glauben* (*Beliefs*), *Wünschen* (*Desires*) und *Absichten* (*Intentions*) zu modellieren (siehe [BME87]). Der BDI-Ansatz leistete einen wichtigen Beitrag zur Entwicklung von Agentenarchitekturen (siehe [RG92] und [RG95]) und kann um Pläne (*Plans*) und Ziele (*Goals*) erweitert werden [BIP87], um weitere mentale Zustände repräsentieren zu können [JW98].

Der Glauben (Belief) des Agenten besteht aus Annahmen über mögliche aktuelle Zustände der Umwelt sowie Annahmen über die Effekte von möglichen Aktionen. Ein Agent kann mehrere Beliefs bezüglich eines Zeitpunktes haben, d. h. Beliefs sind die Menge aller vom Agenten für möglich gehaltenen Zustände der Umwelt.

Wünsche (Desires) eines Agenten sind die von ihm favorisierten zukünftigen Zustände, wobei einzelne Wünsche sich durchaus widersprechen oder nicht erreichbar bzw. erfüllbar

sein können.

Ziele (Goals) beschreiben eine konsistente Teilmenge der Wünsche, die ein Agent im gegenwärtigen Zeitpunkt reichen könnte. Die Ziele müssen entweder eine Teilmenge der Beliefs sein (so genannter *Strong Realism*) oder ihr Gegenteil darf zumindest nicht unvermeidbar sein (*Weak Realism*).

Absichten (Intentions) sind aufgrund ihrer Konsistenz und der aktuellen Ressourcen-Planung ausgewählte Ziele, die der Agent als nächstes erreichen möchte bzw. gerade zu erreichen versucht.

Pläne (Plans) stellen eine Menge zu erreichender Absichten zu einem übergeordneten Plan zusammen und verwalten sowohl Aktionspläne zur Zielerreichung als auch den aktuellen Bearbeitungszustand.

Traditionelle BDI-Agenten verwenden zur Darstellung der möglichen Zustände der Welt so genannte *Zeitbäume* (*Time Trees*), die durch eine Variation der „*Computation Tree Logic*“ beschrieben werden können, während Beliefs, Desires, Goals, Intentions und Plans meist in einer speziellen logischen Beschreibungssprache dargestellt werden.

2.1.2.2 InteRRaP

Die am Deutschen Forschungszentrum für Künstliche Intelligenz (DFKI GmbH) entwickelte *InteRRaP-Agentenarchitektur* (*Integration of Reactivity and Rational Planning*) [FK93] [MJP96] ist eine erweiterte BDI-Architektur. Ziel ist die Integration reaktiver, deliberativer und kooperativer bzw. sozialer Eigenschaften in einer möglichst allgemeinen Agentenarchitektur. Die grundsätzliche Idee des InteRRaP-Modells ist die Aufteilung der Architektur in folgende drei Schichten (siehe Abbildung 1) [MP94]:

1. *Reaktivebene* (*Behavior-based Layer*),
2. *Lokale Planungsebene* (*Local Planning Layer*) und
3. *Kooperationsebene* (*Cooperative Planning Layer*).

Jede Schicht besteht aus zwei Modulen: einer, auf die Schicht angepassten Wissensbasis sowie einer Kontrollkomponente, die unter anderem Methoden und Objekte zur Situationserkennung, Zielauswahl sowie zur Planung und Plandurchführung beinhaltet. Die Wissensbasen bestehen aus Modellen der Umwelt und des internen Zustandes des Agenten sowie aus Bibliotheken von Regeln, Verhaltensmustern, Plänen und Taktiken.

Die *Reaktivebene* ist zuständig für das reaktive Verhalten eines Agenten und enthält eine Vielzahl reaktiver Verhaltensmuster, die aus Regeln oder speziellen Prozeduren (*Behaviours*) bestehen. Sie ist über eine Schnittstelle, bestehend aus Sensoren, Effektoren und verschiedenen Kommunikationskanälen, bidirektional mit der Umwelt verbunden.

Die über der Reaktivebene liegende *lokale Planungsebene* ermöglicht das zielgesteuerte Verhalten des Agenten. Sie umfasst, neben einer Planungskomponente und einer

Wissensbasis zur Repräsentation des Zustandes des Agenten, alle notwendigen Methoden bzw. Objekte zur lokalen Planungssteuerung, Zielauswahl bzw. Plandurchführung.

Oberste Ebene ist die *kooperative Planungsebene*, die es dem Agenten ermöglicht, das Verhalten anderer Agenten bzw. die Koordination und Interaktion mit anderen Agenten in den eigenen Planungs- und Plandurchführungsprozess zu integrieren, um die Erstellung globaler bzw. gemeinsamer Pläne zu ermöglichen.

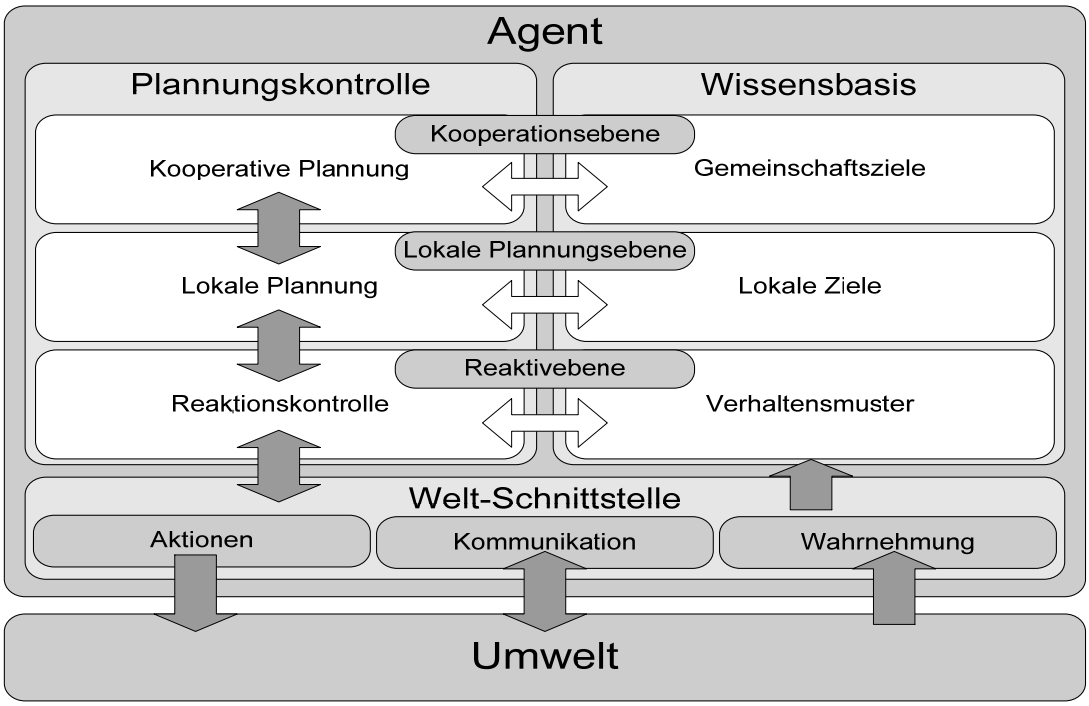


Abbildung 1. Die InteRRaP-Agenten-Architektur

2.1.3 Multiagentensysteme

Multiagentensysteme stellen eine Spezialform des Verteilten Problemlösens dar [CG97]. Sie werden von vielen unterschiedlichen Forschungsrichtungen beeinflusst, wie beispielsweise der Künstlichen Intelligenz, der Informatik, der Entscheidungstheorie, der Psychologie und der Sozionik. Grundsätzlich handelt es sich bei Verteilten Problemlöseverfahren um zentralisierte Top-Down-Lösungsansätze, wobei globale Probleme in lokale Teilprobleme zerlegt werden, die anschließend auf einzelne Problemlöse-Einheiten verteilt werden. Im Falle eines Multiagentensystems bedeutet das, dass ein Problem in eine Menge von Teilproblemen unterteilt wird, die von speziellen Agenten gelöst werden können. Im Gegensatz zum klassischen Verteilten Problemlösen ist die Verteilung der Teilprobleme bei Multiagentensystemen auf einzelne Agenten nicht von vorne herein fest definiert, d. h. die Teilprobleme können während der Laufzeit flexibel umverteilt werden. Die Leistung des

gesamten MAS kann dabei aufgrund von *Emergenzeffekten* über der Summe der Einzelleistungen der Agenten liegen [WG99][WM02].

Multiagentensysteme kommen heute in vielen Bereichen zum Einsatz, wie beispielsweise in der Logistik (z. B. das TeleTruck System [BFV98], das auf dem MARS System [SD98] (**M**odeling **A**utonomous Coope**R**ating **S**hipping Companies) aufbaut) oder im Rahmen so genannter Social Simulations, die das gesellschaftliche Verhalten einer Menge von Individuen untersuchen.

Multiagentensysteme besitzen in der Regel einen *Verzeichnisdienst* (*Repository Service*) [FSS03], der entweder durch einen speziellen Agenten, dem *Verzeichnisagenten* (*Repository Agent* (*RA*)), repräsentiert wird oder ein Bestandteil des Metasystems ist. Der Repository Service hat die Bereitstellung verschiedener Agentenverzeichnisse (z. B. *White Pages* oder *Yellow Pages*) zur Aufgabe. Damit sind Agenten in der Lage, Beschreibungen, Adressen bzw. Expertisen anderer Agenten zu ermitteln, wobei jedem Agenten eine eindeutige Adresse sowie ein Agententyp zugeordnet sind. Der Verzeichnisagent besitzt Kenntnis über alle im konkreten Multiagentensystem zugelassenen Agententypen in Form einer Menge $AP := \{A^1,..., A^n\}, n \in N$ von Agentenprototypen.

Ein Multiagentensystem umfasst somit zur Laufzeit eine Menge von Agenten $AI := \{RA, A_1^1,..., A_{k1}^1,..., A_1^n,..., A_{kn}^n\}$ mit $k_1,\ldots, k_n \in N$, wobei A_i^t eine Instanz des Agentenprototypen A^t ist (siehe Abbildung 2) und während der Laufzeit neue Agenten hinzugefügt und aktive Agenten beendet werden können.

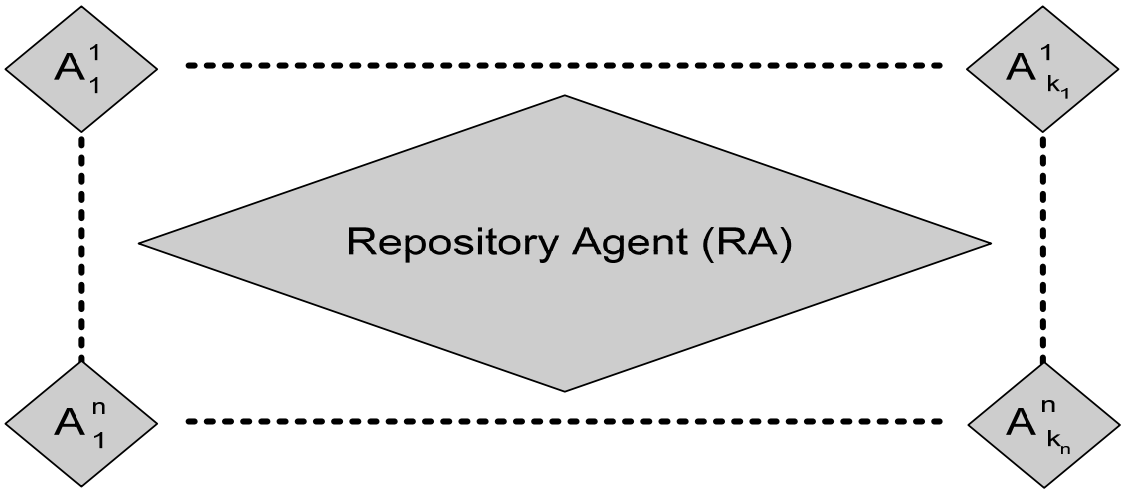


Abbildung 2: Spezifikation eines Multiagentensystems

2.2 Holonische Multiagentensysteme (HMAS)

Die wachsende Komplexität skalierbarer Systeme fordert effiziente Strukturen und Organisationsformen innerhalb von Multiagentensystemen [JNR99], da die Performanz unter den wachsenden Kosten für Kommunikation und Interaktion zwischen den einzelnen Agenten leidet. Eine effiziente Struktur kann durch Realisierung dynamisch organisierter Agentengesellschaften erreicht werden, in denen sich autonome Agenten temporär zu einer übergeordneten Einheit formieren können, um eine gemeinsame Aufgabe effizient zu lösen. Dabei kann eine Vielzahl von unterschiedlichen Agentengesellschafts- und Organisationsformen verwendet werden, die den menschlichen Organisations-, Gesellschafts- oder Unternehmensformen nachempfunden sind (siehe [SM04]).

2.2.1 Definition holonischer MAS

Holonische Multiagentensysteme unterstützen die dynamische Organisation autonomer Agenten, da hier einzelne Agenten beschließen können, temporär einen Teil ihrer Autonomie aufzugeben und sich zu einem übergeordneten Agenten (*Holon*) zusammenzuschließen, um gemeinsam auf effiziente Weise Aufgaben lösen zu können. Der resultierende Holon verhält sich anschließend nach außen wie ein einzelner Agent. Auf diese Weise ist ein Holon eine kohärente fraktale Struktur [KA67], die aus einer Menge von *Unterholonen* gebildet wird (*Holonisierung*), wobei ein Agent ohne Unteragenten als *atomarer* Holon bezeichnet wird. Ein Agent kann dabei zur gleichen Zeit Mitglied bei verschiedenen Holonen sein. Ein Holon wird nach außen durch einen individuellen Agenten repräsentiert, den *Kopf des Holonen* (*Head of Holon*), der entweder ein ausgewähltes Mitglied des Holons oder ein eigens für diesen Zweck erzeugter Agent ist. Alle anderen Mitglieder bilden den *Rumpf* des Holonen (*Body of Holon*).

2.2.2 Holonenstrukturen

Generell gibt es verschiedene Formen der Holonisierung, die man anhand des Autonomiegrades der einzelnen Unteragenten unterscheiden kann. Im Folgenden beschreibe ich die wichtigsten Formen der Holonisierung und erläutere deren jeweiligen Vor- und Nachteile.

2.2.2.1 Gruppe autonomer Agenten

Die Art der Holonisierung, bei der die einzelnen Mitglieder die größte Autonomie behalten, ist die *Gruppe autonomer Agenten*, die einem traditionellen Multiagentensystem entspricht und hauptsächlich als Designhilfe zu strukturiertem agentenorientierten Programmieren dient (siehe Abbildung 3). Zur Limitierung der Kommunikationskosten können einzelne Agenten ausschließlich mit Agenten desselben Holons interagieren, um ein gemeinsames Ziel zu

erreichen. Ein Vorteil besteht darin, dass einzelne Agenten autonom bleiben und mehreren Holonen gleichzeitig angehören können. Im Besonderen bleibt die Fähigkeit zum verteilten Problemlösen innerhalb des Multiagentensystems erhalten. Der größte Nachteil besteht in der Praxis häufig in den hohen Kommunikations- bzw. Interaktionskosten.

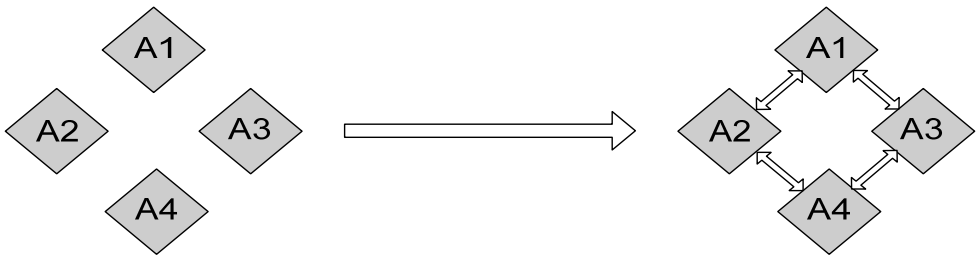


Abbildung 3: Gruppe autonomer Agenten

2.2.2.2 Vollständige Verschmelzung

Die *vollständige Verschmelzung* ist eine Form der Holonisierung, bei der alle teilnehmenden Agenten ihre Autonomie völlig aufgeben und zu einem einzigen Agenten verschmelzen (siehe Abbildung 4). Je nach Anwendungsfall können bei der Verschmelzung verschiedene Arten der Wissensaggregation geeignet sein. Mögliche Aggregationsformen sind beispielsweise die Vereinigung, Durchschnittsbildung oder Summation. Der Vorteil besteht aus der totalen Eliminierung der Kommunikations- und Interaktionskosten, wobei die Fähigkeit zum verteilten Problemlösen verloren geht. Nachdem das gemeinsame Ziel erreicht wurde, wird der Holon terminiert und alle Unteragenten wieder hergestellt.

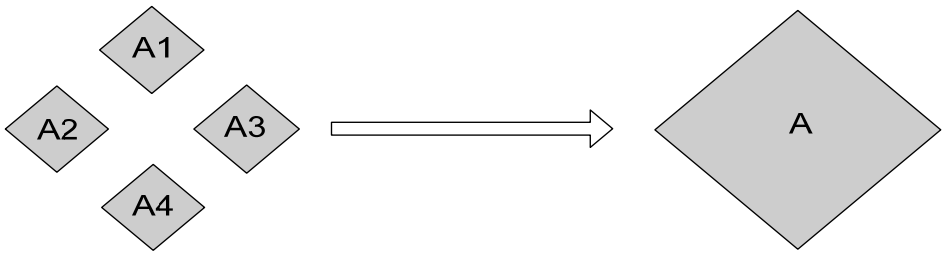


Abbildung 4: Vollständige Verschmelzung

2.2.2.3 Kopfgesteuerte Gesellschaft

Die dritte Form der Holonisierung, die *kopfgesteuerte Gesellschaft*, liegt hinsichtlich des Autonomiegrades der teilnehmenden Agenten zwischen den beiden Extremen *Gruppe autonomer Agenten* und *vollständiger Verschmelzung*. Alle Mitglieder geben nur einen Teil ihrer Selbstständigkeit auf, wobei zur Steigerung der Effizienz ein Agent zum Kopf des Holons gewählt wird, der den Holon nach außen repräsentiert und intern verwaltet (siehe

Abbildung 5). Die Kompetenzen des Kopfes sind skalierbar von rein administrativen Aufgaben bis hin zur Autorität, an die Rumpfagenten Aufgaben zu verteilen sowie für den gesamten Holon zu verhandeln und zu planen, basierend auf den Plänen und Zielen aller Unteragenten, wodurch sich Kommunikations- und Interaktionskosten reduzieren. Dem Kopfagenten kann zusätzlich die Fähigkeit gegeben werden, einzelne Agenten neu einzuführen oder vom Holon auszuschließen [FSS03], wobei diese wie bei der Gruppe autonomer Agenten mehreren Holonen gleichzeitig angehören können.

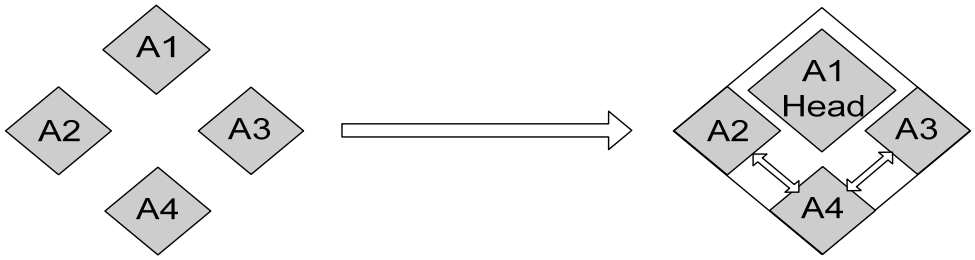


Abbildung 5: Kopfgesteuerte Agentengesellschaft

2.2.3 Holonisierung durch partielles Klonen und Verschmelzen

Alle drei beschriebenen Holonenformen besitzen einen Nachteil. Sowohl bei der kopfgesteuerten Gesellschaft als auch bei der Gruppe autonomer Agenten werden Kommunikation und Interaktion zwischen den Agenten nicht völlig überflüssig. Im Falle der vollständigen Verschmelzung entfallen diese zwar, allerdings ist der Prozess der vollständigen Verschmelzung und Wiederherstellung in der Praxis oft sehr aufwendig und das System verliert darüber hinaus meist die Fähigkeit zum Verteilten Problemlösen.

Ein Ansatz, der die Vorteile der Holonisierung bietet und gleichzeitig die aufgeführten Nachteile limitiert, ist die im Rahmen meiner Arbeit entstandene Holonenbildung basierend auf *partiellem Klonen und Verschmelzen von Agenten* (siehe Abbildung 6).

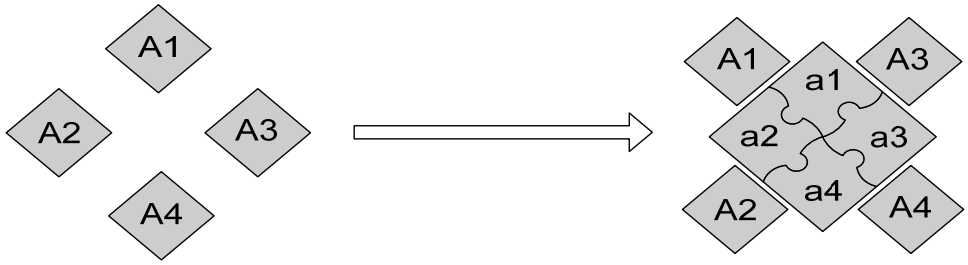


Abbildung 6: Partielles Klonen und Verschmelzen

Um gemeinsam ein bestimmtes Problem zu lösen, erstellen alle teilnehmenden Agenten eine Kopie des Wissens und der Fähigkeiten, die für die Lösung des gemeinsamen Problems relevant sind (*Partielles Klonen*). Die erzeugten Klone werden anschließend zu einem Holon

verschmolzen (*Partielles Verschmelzen*). Der Vorteil besteht darin, dass alle Agenten vollständig erhalten bleiben und für andere Aufgaben verwendet werden können, während der erzeugte Holon nur die für die Aufgabe relevanten Informationen und Fähigkeiten enthält. Somit bleibt dem System die Fähigkeit zum Verteilten Problemlösen erhalten. Zudem ist das partielle Verschmelzen effizienter als die vollständige Verschmelzung.

2.3 Bayes'sche Netze

Bayes'sche Netze sind die Grundlage der so genannten *Verhaltensnetze*, die ich zur Modellierung des individuellen Kaufverhaltens verwende.

2.3.1 Einführung

Um das Kauferhalten von Konsumenten modellieren zu können, müssen die entsprechenden Modelle in der Lage sein, Unsicherheit zu repräsentieren, da in dieser Domäne unvollständige Beobachtungen die Regel sind, über deren Vervollständigungen bzw. Auswirkungen Hypothesen erstellt werden müssen. Im Allgemeinen gibt es drei verschiedene Arten von Unsicherheit [KN04]:

1. *Unkenntnis* (*Ignorance*),
2. *Zufall* bzw. *Nichtdeterminismus* (*Randomness* bzw. *Indeterminism*) und
3. *Verschwommenheit* bzw. *Unklarheit* (*Vagueness*).

Unkenntnis bedeutet, dass keinerlei Information über bestimmte Sachverhalte vorhanden ist. Zufall umfasst generell Ereignisse, die nicht vorhersehbar sind bzw. für die lediglich die Wahrscheinlichkeiten des Eintretens bekannt sind. Unter Verschwommenheit versteht man in der Regel qualitative Attribute mit unscharfen Definitionen.

Der vielversprechenste Ansatz zum Schlussfolgern unter Unsicherheit ist das *probabilistische Schließen* (*probabilistic reasoning*), das auch als so genanntes *Bayes'sches Schließen* (*Bayesian reasoning*) bezeichnet wird [KN04]. Die Basis des Bayes'schen Schließens bildet die *Wahrscheinlichkeitsrechnung* (siehe beispielsweise [BB96] für eine Einführung), wobei das *Bayes'sche Theorem* im Mittelpunkt steht:

$$P(h \mid e) = \frac{P(e \mid h) \cdot P(h)}{P(e)}$$

Das Bayes'sche Theorem sagt aus, dass die Wahrscheinlichkeit des Zutreffens einer Hypothese h in Abhängigkeit einer beobachteten Evidenz e gleich dem Produkt aus der Wahrscheinlichkeit $P(e|h)$ und der Grundwahrscheinlichkeit der Hypothese $P(h)$ ist, das

durch die Division durch $P(e)$ normalisiert wird, d. h. dass sich die Wahrscheinlichkeiten aller Hypothesen zum Wert 1 summieren lassen [KN04]. Mit Hilfe dieses einfachen Theorems lässt sich die *Konditionalisierung* ausdrücken, d. h. die Abhängigkeit des *Glaubens* bzw. *Belief*s $Bel(h)$ an die Gültigkeit einer Hypothese h von der Beobachtung einer Evidenz e, wobei $Bel(h) = P(h|e)$ gilt. Anders gesagt ändert sich der Glaube an eine Hypothese $Bel(h)$ aufgrund des Auftretens von Beobachtungen $\boldsymbol{e}$, wobei man $Bel(h)$ auch als *Folgewahrscheinlichkeit* von h (*Posterior Probability*) bezeichnet.

Die einfachste Form des Bayes'schen Schließens ist das *Naive-Bayes-Modell* [DH73], bei dem sich eine Menge untereinander unabhängiger Einflussfaktoren auf abhängige Zufallsvariablen auswirken. Die unabhängigen Auswirkungen werden durch *Belief-Update-Regeln* repräsentiert, wobei eine Regel vereinfacht aus der Angabe einer bedingten Wahrscheinlichkeitsverteilung besteht. Konkrete Werte der Einflussfaktoren werden im Rahmen von Schlussfolgerungen durch Nennen entsprechender Evidenzen ausgedrückt, deren Auswirkungen auf die abhängigen Variablen anschließend durch einfache Inferenz-Algorithmen berechnet werden.

Das Naive-Bayes-Modell wird erfolgreich zur Klassifikation eingesetzt (siehe z. B. [LIT92]) und kann zum *Tree-Augment-Naive-Bayes-Modell* (*TAN*) [FGG97] erweitert werden, bei dem Korrelationen zwischen Einflussfaktoren durch Baumstrukturen eingeschränkt modelliert werden können. Zur Modellierung nahezu beliebiger Korrelationen bzw. Abhängigkeiten und Unabhängigkeiten zwischen Zufallsvariablen können *Bayes'sche Netze* (*BN*) eingesetzt werden (siehe beispielsweise [PJ88], [KN04], [CDLS99] oder [HD95] für ausführliche Einführungen in probabilistische Netze). Sie sind daher eine ideale Grundlage zur Modellierung des Konsumentenverhaltens, da sie die Reaktionen der Kunden unter Berücksichtigung einer Vielzahl von eventuell nichtlinearen Abhängigkeiten modellieren können, wobei sich die konkreten Beziehungen zwischen Einflussfaktoren und Reaktionen in Form von bedingten Abhängigkeitsverteilungen quantifizierbar aus realen historischen Daten lernen lassen.

2.3.2 Grundlagen

Bayes'sche Netze repräsentieren probabilistische Zusammenhänge in Form bedingter Abhängigkeiten bzw. Unabhängigkeiten zwischen Zufallsvariablen. Die graphische Repräsentation eines Bayes'schen Netzes besteht dabei aus einem Graph $G=(X, E)$ mit folgenden Komponenten und Eigenschaften:

1. Die Menge der Knoten X von G repräsentieren Zufallsvariablen X_i mit $i=1,...,n$ der zu modellierenden Domäne, wobei Knoten *diskrete* oder *stetige* Werte annehmen können. Im weiteren Verlauf dieser Arbeit werde ich die Begriffe Zufallsvariable und Knoten synonym verwenden und bei diskreten Zufallsvariablen eine *endliche*

Anzahl verschiedener Werte annehmen. Die Schreibweise der Werte des Knotens X_i entspricht x_j mit $j=1, ..., z$, wobei z die Anzahl der verschiedenen zulässigen Werte der Zufallsvariablen X_i definiert. Stetige Knoten werden entweder diskretisiert (siehe Kapitel 3.3.2) oder mittels einer Verteilung angegeben, wobei in den meisten Fällen die Gauß-Verteilung verwendet wird.

2. Die Menge der Kanten E der Form $X_a \rightarrow X_b$ von G sind gerichtet und modellieren jeweils Abhängigkeiten zwischen Startknoten X_a und Endknoten X_b, wobei X_a als *Elternknoten* bzw. *Parent* von X_b definiert ist, d. h. $X_a \in Parents(X_b)$, wobei $Parents(X_i)$ als die Menge aller Elternknoten eines Knotens X_i definiert ist.
3. Jeder Knoten X_i in G besitzt eine Tabelle, die so genannte C*onditional Probability Table* (*CPT*), die die bedingte Wahrscheinlichkeitsverteilung $P(X_i|Parents(X_i))$ umfasst, die die quantitativen Effekte der Elternkonten $Parents(X_i)$ auf X_i modellieren. Die Einträge der CPT bestehen dabei aus je einer Zeile für jede mögliche Zustandskombination k aus $parents(X_i)$ der Form $P(x_1|k)$, …, $P(x_z|k)$, wobei $parents(X_i)$ als die Menge aller möglichen Zustandskombinationen der Elternkonten $Parents(X_i)$ definiert ist. Besitzt X_i keine Eltern, so besteht die CPT nur aus einer einzigen Zeile mit unbedingten *Apriori-Wahrscheinlichkeiten* $P(x_j)$ für $j=1, ..., z$. Die Summe der Einzelwerte einer Zeile ergibt in allen Fällen den Wert 1, da sie eine Wahrscheinlichkeitsverteilung angeben.
4. Der Graph G ist gerichtet und azyklisch, d. h. ein so genannter *directed acyclic graph* (*DAG*).

Ein Bayes'sches Netz mit den genannten Eigenschaften repräsentiert die *gemeinsame Wahrscheinlichkeitsverteilung* über alle Zufallsvariablen X_i der Form $P(X_1=x_1 \wedge ... \wedge X_n=x_n)$ bzw. Kurzform $P(x_1,...,x_n)$, die man mittels der *Kettenregel* folgendermaßen faktorisieren kann:

$$P(x_1,...,x_n) = P(x_1) \cdot P(x_2 \mid x_1) ..., \cdot P(x_n \mid x_1,...,x_{n-1}) = \prod_{i=1}^{n} P(x_i \mid x_1,...,x_{i-1}).$$

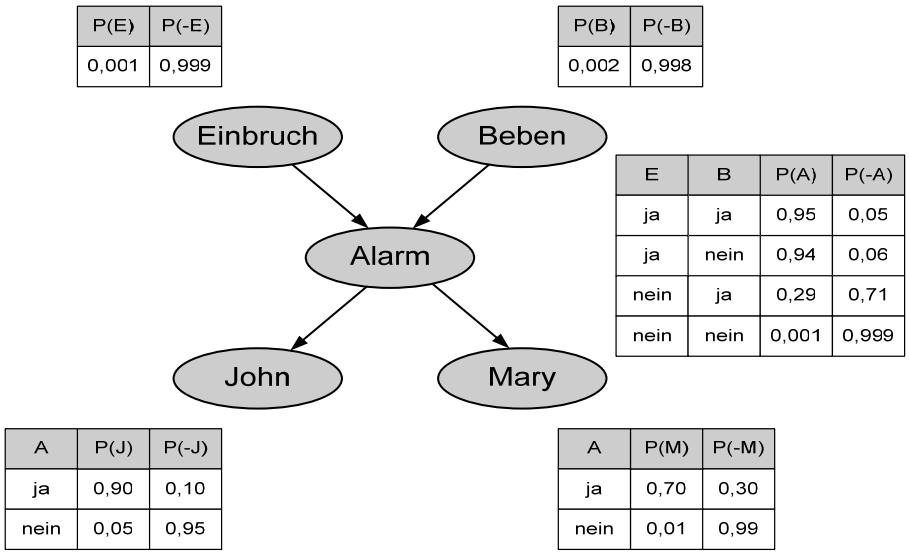
Aufgrund der Tatsache, dass der Wert eines Knotens nur durch die Wertekombination der Elternknoten bedingt wird, lässt sich die Gleichung reduzieren auf (unter der Vorraussetzung $parents(Xi) \subseteq \{x_1,...,x_{i-1}\}$):

$$P(x_1,...,x_n) = \prod_{i=1}^{n} P(x_i \mid parents\,(X_i)).$$

Bayes'sche Netze sind durch diese Reduktion oft exponentiell weniger komplex als die entsprechende gemeinsame Wahrscheinlichkeitsverteilung [RN03], d. h. die Summe der Größen der einzelnen CPT ist in der Praxis oft exponentiell kleiner als die einer vollständigen

Tabelle, deren Einträge sämtliche Zustandskombinationen aller Knoten beinhaltet.

Ein Beispiel für ein Bayes'schen Netz ist folgendes [RN03]: Ein Hausbesitzer in Los Angeles hat ein neues Alarmgerät installiert, das bei einem Einbruch sehr zuverlässig ein Alarmsignal auslöst. Allerdings kann der Alarm auch fälschlicherweise durch leichte Erdbeben ausgelöst werden. Es gibt zwei Nachbarn, John und Mary, die gebeten wurden, per Telefon Bescheid zu geben, wenn sie den Alarm des Hauses hören. John ruft sehr zuverlässig an, wenn er den Alarm hört, aber da er etwas zerstreut ist, verwechselt er ab und zu das Klingeln der Haustür bzw. des Telefons mit einem Alarm und ruft in diesen Fällen fälschlicherweise an. Mary hört sehr oft laute Musik und überhört daher in manchen Fällen den Alarm. Das Szenario kann durch das in Abbildung 7 dargestellte einfache Bayes'sche Netz mit den diskreten Zufallsvariablen *Einbruch*, *Beben*, *Alarm*, *John* und *Mary* modelliert werden, die jeweils nur zwei Zustände besitzen.



P(E)	P(-E)
0,001	0,999

P(B)	P(-B)
0,002	0,998

E	B	P(A)	P(-A)
ja	ja	0,95	0,05
ja	nein	0,94	0,06
nein	ja	0,29	0,71
nein	nein	0,001	0,999

A	P(J)	P(-J)
ja	0,90	0,10
nein	0,05	0,95

A	P(M)	P(-M)
ja	0,70	0,30
nein	0,01	0,99

Abbildung 7: Beispiel eines einfachen Bayes'schen Netzes

Die Wahrscheinlichkeitstabellen der Variablen *Einbruch* und *Beben* bestehen jeweils aus den unbedingten Apriori-Wahrscheinlichkeiten des Auftretens von Einbrüchen und Erdbeben, während die Tabellen der übrigen Variablen vollwertige CPT sind, die die Wahrscheinlichkeitsverteilungen über den Zuständen der Variablen in Abhängigkeit von allen möglichen Zustandskombinationen der Elternknoten repräsentieren. Durch Verwendung der oben genannten Formel kann man nun mit Hilfe des Netzes beispielsweise die Wahrscheinlichkeit dafür berechnen, dass nach einem ausgelösten Alarm sowohl John als auch Mary anrufen, obwohl weder ein Einbruch noch ein Erdbeben stattgefunden hat:

$$P(j \wedge m \wedge a \wedge -e \wedge -b) = P(j \mid a) \cdot P(m \mid a) \cdot P(a \mid -e \wedge -b) \cdot P(-e) \cdot P(-b)$$

$$= 0{,}90 \cdot 0{,}70 \cdot 0{,}001 \cdot 0{,}999 \cdot 0{,}998 = 0{,}00062$$

Diese Berechnung ist generell für alle Zustandskombinationen der Zufallsvariablen möglich, falls das Netz die gesamte Wahrscheinlichkeitsverteilung der Zufallsvariablen repräsentiert. Allgemein lassen sich unter diesen Bedingungen die Folgewahrscheinlichkeiten und deren Verteilungen einer Menge so genannter *Antwortknoten* (*Query Nodes*) mittels Inferenz-Algorithmen (*Probability Propagation* bzw. *Belief Updating*) berechnen, wenn die Zustände einer Menge von *Evidenzknoten* (*Evidence Nodes*) vorgegeben werden, die die Annahmen der Fragestellung repräsentieren (siehe Kapitel 4.3). Beispielsweise wäre die Wahrscheinlichkeit, dass ein Einbruch (Anfragevariable) verübt wurde und sowohl John als auch Mary (Evidenzvariablen) anrufen, wie folgt:

$$P(e \mid j \wedge m) = 0{,}284\,.$$

Zur Modellierung und Simulation mit Hilfe Bayes'scher Netze sind neben den beschrieben Apriori-Wahrscheinlichkeiten und den bedingten Wahrscheinlichkeitstabellen (CPT) die so genannten *Randwahrscheinlichkeiten* (*marginal distributions*) sowie die darauf aufbauenden *Erwartungswerte* der einzelnen Zufallsvariablen von großer Bedeutung.

Die Randwahrscheinlichkeitsverteilung $P(X_i)$ einer diskreten Zufallsvariable X_i mit den möglichen Zuständen $x_1,\ldots,x_z$ besteht aus der Menge der Wahrscheinlichkeiten $p_1, \ldots, p_z$ mit $p_i = P(X_i = x_i)$, wobei p_i auch als *Zustandwahrscheinlichkeit* des Zustands x_i bezeichnet wird. Im Falle eines Wurzelknotens besteht die Randwahrscheinlichkeitsverteilung also direkt aus der Apriori-Wahrscheinlichkeitsverteilung. Im Falle einer Zufallsvariable X_i, die bedingt abhängig von einer Menge von Elternknoten $Parents(X_i)$ mit den möglichen Zustandskombinationen $parents(X_i)$ ist, berechnen sich die einzelnen p_i wie folgt:

$$p_i = \sum_{parents(X_i)} P(X_i = x_i \mid parents(X_i)) \cdot P(parents(X_i))$$

Aufbauend auf der Randwahrscheinlichkeitsverteilung lässt sich der Erwartungswert $E(X_i)$ von X_i ermitteln, wobei der Erwartungswert einer diskreten Zufallsvariable X_i mit z möglichen Zuständen im Allgemeinen als die Summe der reellen Zustandswerte $x_1, \ldots, x_z$ multipliziert mit der entsprechenden Zustandswahrscheinlichkeit p_i definiert ist:

$$E(X_i) = \sum_{i=1}^{z} x_i \cdot p_i$$

Die reellen Zustandswerte x_i sind dabei die (realen) Mittelwerte der diskretisierten Intervalle, die die entsprechenden Zustände repräsentieren (siehe Kapitel 3.3.2).

2.3.3 Struktur

Diskrete BN bzw. *stetige BN* sind Netze, die nur diskrete bzw. stetige Knoten besitzen, während man Netze mit diskreten und stetigen Knoten als *hybride BN* bezeichnet. Existiert eine Kante der Form $X_a \rightarrow X_b$, so nennt man X_a einen *Elternknoten* (*Parent Node*) von X_b und X_b einen *Kindknoten* (*Child Node*) von X_a. Gibt es eine Folge bzw. Kette von gerichteten Kanten der Form $X_1 \rightarrow X_2 \rightarrow ... \rightarrow X_n$, so nennt man X_1 einen *Vorfahren* (*Ancestor*) von X_n und X_n einen *Nachfahren* (*Descendant*) von X_1. Einen Knoten ohne Eltern nennt man *Wurzelknoten* (*Root Node*), einen Knoten ohne Kinder einen *Blattknoten* (*Leaf Node*). Alle anderen Knoten innerhalb des Netzes werden *Zwischenknoten* (*Intermediate Nodes*) genannt. Im Allgemeinen stellen Wurzelknoten Einflussfaktoren innerhalb der zu modellierenden Domäne dar, während Blattknoten Effekte und Auswirkungen repräsentieren. Ein für spätere Beschreibungen von Bayes'schen Netzen nützliches Konzept ist die so genannte *Markov-Decke* (*Markov Blanket*) eines Knotens, die aus der Menge der Elternknoten, seinen Kindern sowie wiederum deren Eltern besteht.

2.3.3.1 Markov-Eigenschaft

Eine wichtige Vorraussetzung der akkuraten Modellierung von Domänen mit Hilfe Bayes'scher Netze ist die *Markov-Eigenschaft*, die darin besteht, dass es keine direkten Abhängigkeiten in der zu modellierenden Domäne geben darf, die nicht explizit durch direkte Kanten im Bayes'schen Netz modelliert werden [KN04]. Bayes'sche Netze, die die Markov-Eigenschaft haben, nennt man auch *Unabhängigkeitskarten* (*Independence-Maps* bzw. *I-Maps*), da jede fehlende Kante im Netz die reale Unabhängigkeit der beiden entsprechenden Zufallsvariablen repräsentiert.

Während es wichtig ist, alle realen Unabhängigkeiten durch entsprechende fehlende Kanten zu modellieren, ist es nicht unbedingt nötig, dass alle direkten Kanten eines Bayes'schen Netzes auch direkte Abhängigkeiten in der zu modellierenden Domäne repräsentieren, d. h. es ist durchaus möglich, mit Hilfe eines vollständig verbundenen Bayes'schen Netzes die gemeinsame Wahrscheinlichkeitsverteilung der Zufallsvariablen zu modellieren. Um die exponentiell steigende Komplexität allerdings so gering wie möglich zu halten, werden minimale Bayes'sche Netze angestrebt. Solche minimalen Netze sind I-Maps, bei denen keine Kante gelöscht werden kann, ohne dass die Markov-Eigenschaft verloren

geht, d. h. sie keine I-Map mehr darstellen.

Im Falle, dass zu jeder Kante in einem Bayes'schen Netz auch eine reale direkte Abhängigkeit zwischen den beiden Zufallsvariablen existiert, spricht man von einer *Abhängigkeitskarte* (*Dependence-Map* bzw. *D-Map*). Ein Bayes'sches Netz, das sowohl eine I-Map als auch eine D-Map darstellt wird auch *perfekte Karte* (*Perfect-Map* bzw. *P-Map*) genannt.

2.3.3.2 Bedingte Unabhängigkeit

Die Markov-Eigenschaft Bayes'scher Netze, also die explizite Repräsentation von realen Unabhängigkeiten, spielt eine entscheidende Rolle bei der Modellierung der Netzstruktur sowie insbesondere bei Schlussfolgerungsprozessen. Im Mittelpunkt stehen dabei die beiden folgenden äquivalenten Unabhängigkeitskriterien:

1. Wenn alle Zustände der Eltern *Parents*(X_i) einer Zufallsvariable X_i bekannt sind, so ist X_i bedingt unabhängig von seinen Nicht-Nachfolgern innerhalb der Struktur des Bayes'schen Netzes [WF03].
2. Wenn alle Zustände der Markov-Decke einer Zufallsvariable bekannt sind, so ist X_i bedingt unabhängig von allen anderen Zufallsvariablen innerhalb der Struktur des Bayes'schen Netzes [RN03].

Diese beiden Kriterien lassen sich erweitern zur so genannten *D-Separation***,** die eine wichtige Grundlage für viele Inferenz-Algorithmen bildet. Mit ihrer Hilfe ist es möglich zu entscheiden, ob eine Knotenmenge X von einer Knotenmenge Y bedingt unabhängig ist, wenn für eine Knotenmenge E alle Zustände bekannt sind. Weiterführende Informationen über die D-Separation sind [RN95], [PJ88] sowie [KN04].

Obwohl Bayes'sche Netze als I-Maps auf der Modellierung von bedingten realen Unabhängigkeiten basieren, wird zur ihrer Konstruktion häufig eine *kausale Interpretation ihrer Kanten* zugrunde gelegt, die aus der Annahme besteht, dass Elternknoten direkte kausale Einflüsse auf ihre Kindknoten besitzen [WF03]. Interessanterweise führt diese Vorgehensweise in den meisten praktischen Anwendungen zu einer Netzstruktur, die die bedingten Unabhängigkeiten im Sinne der beiden Unabhängigkeitskriterien sowie der D-Separation widerspiegeln [HD98].

2.3.4 Bayes'sche Netzklassen

2.3.4.1 Dynamische Bayes'sche Netze (DBN)

Dynamische Bayes'sche Netze (DBN) erweitern Bayes'sche Netze um *temporale Abhängigkeiten* zwischen Zufallsvariablen. Die Idee ist, für jeden Zeitpunkt t_i eines diskreten Zeitraumes $T = \{t_1,\ldots,t_m\}$ jeweils für jede Zufallsvariable X_i des Netzes eine Instanz X_i^t zu

erzeugen, wobei die Knotenmenge $\{X_1^t,...,X_n^t\}$ als *Zeitscheibe* TS_t (*Time-Slice*) zum Zeitpunkt t bezeichnet wird und Kanten zwischen Knoten derselben Zeitscheibe *Scheibenkanten* genannt werden. Die Struktur der Netzinstanzen wird im Zeitverlauf üblicherweise als statisch angenommen, obwohl DBN auch zeitliche Veränderungen der Netzstrukturen unterstützen. Die Menge der Knoten X^t zum Zeitpunkt t kann in die Menge der *beobachtbaren* Knoten E^t (*Sensorknoten*) und die Menge der *unbeobachtbaren* Knoten S^t (*Status- bzw. Zustandsknoten*) unterteilt werden. Die Wahrscheinlichkeitsverteilung $P(S^t|S^{t-1})$ nennt man *Transition-Modell*, wohingegen die Verteilung $P(E^t|S^t)$ als *Sensor-Modell* definiert ist. DBN können temporale Abhängigkeiten zwischen einzelnen Zufallsvariablen durch *Temporalkanten* zwischen Zustandsknoten aus verschiedenen Zeitscheiben repräsentieren, wobei nur Kanten der Form $S_r^t \rightarrow S_r^{t+1}$ bzw. $S_r^t \rightarrow S_s^{t+1}$ zugelassen sind. Das heißt, temporale Einflüsse sind nur gerichtet zwischen zwei direkt aufeinander folgenden Zeitscheiben erlaubt, was einem *Markov-Prozess erster Ordnung* [RN03] entspricht. Ein Markov-Prozess ist ein Prozess, der die *Markov-Annahme* erfüllt, d. h. der aktuelle Zustand des Prozesses ist nur von einer *endlichen* Anzahl vorangegangener Zustände abhängig. Analog ist bei einem Markov-Prozess n-ter Ordnung der aktuelle Zustand ausschließlich von den n letzten Vorgängerzuständen abhängig. Unter der Annahme, dass sich die Struktur der Netzinstanzen im Zeitverlauf nicht ändert, können DBN in sehr kompakter Weise repräsentiert werden, indem jeweils nur wenige aufeinander folgende Zeitscheiben (so genanntes *Zeitfenster*) betrachtet werden [KU95] und das Zeitfenster im Laufe von Inferenz-Algorithmen entsprechend verschoben wird. Die Berechnung der Wahrscheinlichkeitswerte der Knoten zukünftiger Zeitscheiben wird in diesem Zusammenhang auch als *probabilistische Projektion* [KN04] bezeichnet, wobei eine Zeitscheibe als zukünftig gilt, wenn sie jünger ist als der späteste Zeitpunkt einer vorgegebenen Evidenz. Generell gibt es vier verschiedene Inferenz-Arten bezüglich temporärer probabilistischer Modelle [RN03]:

1. *Filterung (Filtering* bzw. *Monitoring)*,
2. *Vorhersage (Prediction)*,
3. *Glättung (Smoothing* bzw. *Hindsight*) sowie
4. *Ermittlung der wahrscheinlichsten Erklärung (Most Likely Explanation)*.

Unter Filterung versteht man die Berechnung der Wahrscheinlichkeitsverteilung $P(X^t|e_{1:t})$ der Zeitscheibe TS_t zum aktuellen Zeitpunkt t in Kenntnis aller bisher gültigen Evidenzen $e_1,\ldots,e_t$. Vorhersage bezeichnet die Ermittlung der Wahrscheinlichkeitsverteilung $P(X^{t+k}|e_{1:t})$ eines zukünftigen Zustandes $t+k$, wohingegen Glättung die Berechnung der Wahrscheinlichkeitsverteilung $P(X^k|e_{1:t})$ eines vergangenen Zustands zum Zeitpunkt k mit $0 \leq k < t$ darstellt. Die Methode der Ermittlung der wahrscheinlichsten Erklärung dient zur Beantwortung der Frage, welche konkreten Zustände $x_{1:k}$ der Statusknoten über k Zeitscheiben am ehesten zur Entstehung einer vorgegebenen Sequenz von Beobachtungen

bzw. Evidenzen $e_{1:k}$ an den Sensorknoten geführt haben könnte. DBN bilden unter anderem aufgrund der Möglichkeiten dieser vier Arten der Inferenz sowie ihrer relativ zu anderen Modellen kompakten Darstellung eine der effektivsten Methoden zur Modellierung komplexer temporal probabilistischer Domänen.

2.3.4.2 Objektorientierte Bayes'sche Netze (OOBN)

Die Komplexität des Konstruktionsprozesses probabilistischer Netze hängt von der Komplexität der zugrunde liegenden Domäne ab [WF03]. Zur Reduktion des Aufwandes eignen sich dabei die von Koller und Pfeffer entwickelten objektorientierten Bayes'schen Netze (OOBN) [KP97], die auf der Idee basieren, Konzepte des objektorientierten Software-Engineering in den Prozess des Knowledge-Engineering einfließen zu lassen. OOBN bestehen aus einer objektorientierten Netzbeschreibungssprache, bei der Bayes'sche Netze als Objektklassen mit Eigenschaften definiert werden, die wiederum aus Netzen bzw. Teilnetzen bestehen können. Mit OOBN können auf diese Weise in komplexen Szenarien *situationsspezifische* Netzinstanzen vorgegebener Netzklassen erlernt und sogar zur Laufzeit, abhängig von den aktuellen Gegebenheiten und Aufgabenstellungen, konstruiert werden [ML98], indem mit Hilfe einer Bibliothek von *Teilnetzen* bzw. *Netzfragmenten* [LM97] situationsspezifische Netze flexibel zusammengefügt werden, die zu sehr kompakten Netzstrukturen und somit zu verkürzten Berechnungszeiten von Inferenz-Algorithmen führen. Darüber hinaus können einige zu wiederholende Berechnungen während eines Inferenz-Prozesses durch die objektorientierte Konstruktion vermieden werden [KP97] [PKMT99] und das in den objektorientierten Strukturbeschreibungen enthaltende Wissen über die zu modellierende Domäne kann dazu genutzt werden, das Lernen der Struktur und der Wahrscheinlichkeitswerte gegenüber dem unstrukturierten Vorgehen deutlich zu beschleunigen [BLN01].

2.3.4.3 Probabilistische relationale Modelle (PRM)

Probabilistische relationale Modelle (PRM) [GFKP01] [KP98] sind eine Erweiterung der objektorientierten Bayes'schen Netze, bei der Datentabellen einer relationalen Datenbank mit ihren Attributen und eventuellen Relationen zu anderen Tabellen auf entsprechende Netzklassen abgebildet werden. Der Vorteil der PRM liegt darin, dass sie sowohl Unsicherheiten bezüglich der Relationen zwischen verschiedenen Klassen als auch zwischen einzelnen Instanzen einer Klasse modellieren können. Die Verwendung von PRM ist auf diese Weise eine effektive Methode, komplexe Domänen mit einer Vielzahl objektorientierter Bayes'scher Netze sowie deren Relationen zu repräsentieren, die durch spezielle Lernverfahren größtenteils automatisch aus relationalen Datenbanken generiert werden können.

2.3.4.4 Entscheidungsnetzwerke und Einflussdiagramme

Bayes'sche Netze modellieren zwar probabilistische Zusammenhänge zwischen einzelnen Zufallsvariablen, aber weder der Prozess der Entscheidungsfindung noch der des Planens werden dabei direkt unterstützt. Dies wird durch die Erweiterung Bayes'scher Netze zu Entscheidungsnetzwerken (Decision Networks) bzw. Einflussdiagrammen (Influence Diagrams) ermöglicht, die Entscheidungsmöglichkeiten bei durch Evidenzen vorgegebenen Situationen nach ihrem Nutzen bewerten bzw. den Gesamtnutzen einer Entscheidungskette optimieren können. Ein Entscheidungsnetzwerk bzw. Einflussdiagramm besteht aus einem azyklischen gerichteten Graph über der Vereinigung der folgenden drei Knotenmengen:

1. *Zufallsknoten* (*Chance Nodes*) $X = \{X_1,\ldots,X_r\}$,
2. *Entscheidungsknoten* (*Decision Nodes*) $D = \{D_1,\ldots,D_s\}$ sowie
3. *Bewertungsknoten* (*Utility Nodes*) $U = \{U_1,\ldots,U_t\}$.

Die Menge der Zufallsknoten X entspricht dabei den Knoten bzw. Variablen eines Bayes'schen Netzes, die CPT besitzen und zusammen mit der Menge ihrer Kanten das dem Entscheidungsnetzwerk zugrunde liegende Bayes'sche Netz bilden. Die CPT einer Zufallsvariable X_i gibt die Wahrscheinlichkeitsverteilung über die Menge aller für X_i zulässigen Werte in Abhängigkeit sämtlicher Wertkombinationen aller Elternknoten *Parents*(X_i) an, wobei ein Zufallsknoten sowohl andere Zufallsknoten als auch Entscheidungsknoten als Elternknoten besitzen kann.

Entscheidungsknoten D_i repräsentieren jeweils eine konkrete Entscheidung zu einem bestimmten Zeitpunkt, wobei die verschiedenen Werte eines Entscheidungsknotens die Menge aller möglichen Entscheidungsalternativen bzw. Handlungsalternativen definiert. Besitzt ein Entscheidungsnetzwerk mehrere Entscheidungsknoten, so werden sie anhand eines gerichteten Kantenpfades, bestehend aus *Rangkanten* (*Precedence Links*), in eine zeitliche Reihenfolge gebracht. Entscheidungsknoten besitzen keine CPT, können aber neben anderen Entscheidungsknoten in bestimmten Fällen auch Zufallsknoten als Elternknoten besitzen, um den Sachverhalt auszudrücken, dass die Werte bzw. Evidenzen dieser Zufallsknoten zum Zeitpunkt der entsprechenden Entscheidung bekannt sein werden bzw. müssen. Die gerichteten Kanten zwischen Zufallsknoten und Entscheidungsknoten werden als *Informationskanten* (*Information Links*) bezeichnet. Besitzt ein Entscheidungsknoten ein oder mehrere Zufallsknoten als Eltern, so wird der Entscheidungsknoten um eine *Entscheidungstabelle* (*Decision Table*) erweitert, die angibt, welche Entscheidung in Abhängigkeit der konkreten Werte der Zufallsknoten zu treffen ist.

Bewertungsknoten U_i repräsentieren die Bewertung bzw. den Nutzen der durch spezielle Inferenz-Algorithmen für Entscheidungsnetzwerke bzw. Einflussdiagramme [JFV01] erhaltenen Wahrscheinlichkeitsverteilungen der Elternknoten *Parents*(U_i) von U_i. Die

Bewertungsknoten dürfen Zufallsknoten und Entscheidungsknoten als Eltern besitzen, aber ihrerseits nicht Elternteil eines beliebigen anderen Knotens sein. Die Bewertung erfolgt anhand der Werte einer *Bewertungs-* bzw. *Nutzentabelle* (*Utility Table*) oder wird durch eine *Bewertungs-* bzw. *Nutzenfunktion* f_U (*Utility Function*) ermittelt. Bewertungstabellen beinhalten den Nutzen aller Zustandskombinationen der Elternknoten, wohingegen eine Nutzenfunktion $f_U(parents(U_i))$ den Nutzen auf Basis der Zustandkombinationen $parents(U_i)$ der Elternknoten berechnet. Existieren in einem Entscheidungsnetzwerk mehrere Bewertungsknoten, so ist die Gesamtbewertung die Summe der Bewertungen der einzelnen Bewertungsknoten.

Entscheidungsnetzwerke bzw. Einflussdiagramme können ebenso wie Bayes'sche Netze zu *dynamischen Entscheidungsnetzwerken* (*Dynamic Decision Networks*) bzw. *dynamischen Einflussdiagrammen* (*Dynamic Influence Diagrams*) [KN04] erweitert werden, um zeitliche Abhängigkeiten modellieren zu können, wobei Zufallsknoten und Entscheidungsknoten in allen Zeitscheiben vorkommen können und Bewertungsknoten nur in der letzten Zeitscheibe erlaubt sind. Dynamische Entscheidungsnetzwerke werden bezüglich generischer Planungsprozesse sehr schnell zu komplex und sind somit in der Praxis in vielen Fällen nicht praktikabel.

2.3.5 Anwendungsmöglichkeiten und alternative Verfahren

Bayes'sche Netze können zur probabilistischen Modellierung des Konsumentenverhaltens verwendet und basierend auf Schlussfolgerungsprozessen sowohl zur Beantwortung prognostischer als auch diagnostischer Fragestellungen eingesetzt werden. Darüber hinaus lassen sie sich zur Mustererkennung und Klassifikation nutzen. Sie besitzen eine semantische Repräsentation und können nichtlineare Abhängigkeiten und zeitliche Abläufe modellieren. Aus diesen Gründen habe ich sie als homogene Grundlage der Verhaltensmodellierung alternativen Verfahren und Modellen vorgezogen, da diese sich meist nur zur Modellierung bzw. Beantwortung von Teilaspekten eignen oder eine geringere Mächtigkeit besitzen. Dennoch können sie in manchen Fällen alternativ oder ergänzend verwendet werden. Die wichtigsten alternativen Verfahren bezogen auf einzelne Fragestellungen sind unter anderem:

- Hidden-Markov-Modelle (HMM),
- Kalman-Filter (KF),
- Dempster-Shafer-Modelle (DSM),
- Künstliche neuronale Netze,
- Lineare und nichtlineare Regression sowie
- Fuzzy Logik.

Im Folgenden führe ich kurz in die einzelnen Verfahren ein und gehe auf Vor- und Nachteile gegenüber Bayes'scher Netze ein.

2.3.5.1 Hidden-Markov-Modelle (HMM)

Ein Hidden-Markov-Modell (HMM) ist ein temporal probabilistisches Modell, das auf dem *Markov-Prozess* basiert, wobei die zugelassenen Prozess-Zustände durch die verschiedenen Werte einer *einzigen diskreten Zufallsvariable* definiert sind. Sollen mehrere Zustandsvariablen verwendet werden, müssen diese zu einer so genannten *Megavariable* kombiniert werden, die alle Wertkombinationen der beteiligten Zustandsvariablen umfasst [RN03]. Die Komplexität nimmt auf diese Weise exponentiell mit der Anzahl der Zustandsvariabeln und deren Zuständen zu. HMM zeichnen sich vor allem durch vergleichsweise elegante Inferenz-Algorithmen (z. B. für Filterung, Vorhersage oder Glättung) aus, die hauptsächlich auf einfachen Matrixberechnungen basieren, da sich sowohl das Transition-Modell als auch das Sensor-Modell als Matrizen darstellen lassen (vgl. dynamische Bayes'sche Netze).

Es ist offensichtlich, dass jedes HMM ein spezielles DBN ist, das nur aus einer einzigen Zustandsvariable und einer Sensorvariable besteht, wohingegen jedes DBN mit ausschließlich diskreten Zufallsvariablen in ein HMM umgewandelt werden kann, indem alle Zustandsknoten zu einer einzigen Megavariable zusammengefasst werden [RN03], d. h. HMM und diskrete DBN können als äquivalente Modelle angesehen werden. DBN besitzen allerdings eine wesentlich geringere Komplexität, da die Transition-Matrix des HMM bei der Kombination aller Zustandsvariablen zu einer Megavariablen exponentiell wächst, wohingegen das DBN eine deutlich kompaktere Darstellung des Transition-Modells bietet. Russell und Norvig vergleichen die Beziehung zwischen HMM und DBN mit der Beziehung zwischen „normalen" Bayes'schen Netzen und der vollständigen Tabelle der Gesamtwahrscheinlichkeitsverteilung über alle Zufallsvariablen. Zur Modellierung komplexer Domänen sollten daher in der Praxis DBN verwendet werden, da HMM in diesen Fällen meist zu komplex und somit unbrauchbar werden, wohingegen einfache Domänen auf elegante und einfache Weise mit Hilfe von HMM als Spezialfall eines DBN modelliert werden können.

2.3.5.2 Kalman-Filter (KF)

Kalman-Filter (KF) sind spezielle, temporale, probabilistische Modelle und stellen entsprechende Inferenz-Algorithmen zur Filterung in Domänen mit *stetigen* Zustandsänderungen und *verrauschten* Beobachtungswerten zur Verfügung. Die Menge der Zufallsvariablen besteht dabei entweder ausschließlich aus stetigen Variablen (*stetiges Netzwerk*) oder aus einer Mischung aus stetigen und diskreten Knoten (*hybrides Netzwerk*). Das eventuelle Rauschen in den Beobachtungswerten wird durch *lineare Gauß-Verteilungen* modelliert. *Erweiterte Kalman-Filter* (*EKF*) bzw. so genannte *Wechselnde Kalman-Filter* (*WKF*) versuchen KF um die Modellierung von Nichtlinearität zu erweitern, wobei EKF nur

in wenigen Domänen mit ganz speziellen Eigenschaften einsetzbar sind und WKF durch den parallelen Einsatz mehrerer KF sehr schnell unpraktikabel werden. Jedes KF kann durch ein stetiges DBN mit linearen Gauß-Verteilungen modelliert werden, wohingegen aufgrund der Nichtlinearität von DBN nicht jedes DBN durch ein KF repräsentiert werden kann [RN03]. Zusätzlich können stetige DBN im Gegensatz zu KF neben Gauß-Verteilungen beliebige Wahrscheinlichkeitsverteilungen modellieren.

2.3.5.3 Dempster-Shafer-Modelle (DSM)

Dempster-Shafer-Modelle (DSM) basieren auf der Dempster-Shafer-Theorie, die sich im Besonderen mit der Modellierung von *Unkenntnis* und *Belief-Funktionen* der Form *Bel*(*X*) beschäftigt, wobei die zu modellierende Domäne durch eine Menge von Hypothesen repräsentiert wird. DSM eignen sich somit zur Modellierung von Domänen über die nur unvollständige bzw. beschränkte Informationen vorliegen. Die Belief-Funktionen geben dabei die Stärke der Überzeugung an, mit der eine Hypothese *X* für wahr bzw. zutreffend gehalten wird. DSM berechnen somit im Gegensatz zu BN nicht die Wahrscheinlichkeitswerte für Ereignisse unter bestimmten Voraussetzungen (Evidenzen), sondern die Wahrscheinlichkeit des Zutreffens einer Menge von über die Domäne aufgestellten Hypothesen. Die Deduktions-Algorithmen basieren bei DSM auf der *Dempster'schen Kombinationsregel* [PJ88], wobei DSM in der Lage sind, partielle Evidenzen, d. h. unsichere Angaben über Evidenzen, zu verarbeiten. DSM werden in der Praxis, aufgrund der umfangreichen Beschreibungen von Entscheidungsregeln, häufig sehr komplex und eigenen sich für Analysen (z. B. Diagnosen) weniger gut als BN, können dafür allerdings auch bei unvollständigem Domänwissen angewendet werden, wobei die Anzahl bzw. die Aussage der unterschiedlichen Schlussfolgerungen in diesem Fall ebenfalls beschränkt bzw. unvollständig sind [PJ88].

2.3.5.4 Neuronale Netze

Künstliche neuronale Netze repräsentieren komplexe nichtlineare Funktionen, deren Parameter anhand einer möglicherweise verrauschten Trainingsmenge gelernt werden können. Die Menge der Knoten *N* eines neuronalen Netzes besteht aus *Neuronen*, die vereinfachte Modelle biologischer Neuronen sind und durch Kanten verbunden werden. Jeder Kante der Form $N_r \rightarrow N_s$ wird ein Gewicht $W_{r,s}$ zugeordnet, welches das Vorzeichen und die Stärke des Einflusses von N_r auf N_s modelliert. Jedes Neuron N_s besitzt eine so genannte Aktivierung a_s, deren Wert basierend auf der Summe der aktuellen Aktivierungen sämtlicher Elternknoten *Parents*(N_s) mit Hilfe einer Funktion *g* berechnet wird, wobei alle Aktivierungen mit den entsprechenden Kantengewichten multipliziert werden:

$$a_s = g\left(\sum_{r=0}^{n} W_{r,s} \cdot a_r\right),$$

wobei $W_{0,s}$ als so genanntes *Bias-Gewicht* bezeichnet wird und a_0 den konstanten Wert -1 annimmt. Ein Neuron N_s gilt dabei als *aktiv* bzw. *aktiviert*, wenn gilt:

$$\sum_{r=1}^{n} W_{r,s} \cdot a_r > W_{0,s}$$

Als Funktion *g* werden bei neuronalen Netzen entweder die *Treshold-Funktion* bzw. S*ign-Funktion* oder die *Sigmoid-Funktion* $1/(1+e^{-x})$ eingesetzt, um zu gewährleisten, dass die durch das neuronale Netz repräsentierte Funktion nichtlinear ist sowie der aktive Zustand eines Neurons nahe am Wert 1 und der inaktive nahe am Wert 0 liegt [RN03].

Generell kann man neuronale Netze in *Feed-Forward-Netze* (*FFN*) und *Recurrent-Netze* (*RN*) unterteilen. Bei FFN besteht das zugrunde liegende Netz aus einem azyklischen Graphen, während RN Zyklen im Graphen erlauben. FFN stellen somit Funktionen dar, die ausschließlich auf Basis der Eingabewerte basieren, wohingegen RN ein dynamisches System bilden, bei dem Ausgaben wiederum als Eingaben dienen können, wodurch RN eine Art Kurzzeitspeicher modellieren, da der Zustand eines RN von vorangegangen Eingaben abhängig ist [RN03]. RN zeigen aufgrund dieser Tatsache in der Praxis oft unerwünschte Eigenschaften, die in den meisten Fällen nur schwer nachvollziehbar sind. Die bis jetzt wahrscheinlich am weitesten entwickelten RN sind die so genannten *Hopfield-Netzwerke* (siehe [HJJ82] für weiterführende Details).

Die Struktur von FFN basiert auf Schichten bzw. Layern von Neuronen, wobei einzelne Neuronen Aktivierungen ausschließlich von Neuronen der direkt vorhergehenden Schicht erhalten können. Es existieren bei FFN drei verschiedene Arten von Schichten:

1. *Eingabeschicht* (*Input Layer*),
2. *Versteckte Schicht* (*Hidden Layer*) und
3. *Ausgabeschicht* (*Output Layer*).

Die Eingabeschicht besteht aus Neuronen zur Aufnahme von Eingabewerten, während die Neuronen der Ausgabeschicht die auf den Eingabewerten resultierenden Ausgabewerte zur Verfügung stellen. Versteckte Schichten dienen zur Erhöhung der Komplexität der durch das FFN darstellbaren Funktionen. FFN, die nur aus einer Eingabe- und einer Ausgabeschicht bestehen, nennt man auch *Einschicht-Netzwerke* (*Single Layer Feed-Forward Network*) oder auch *Perceptron*, wohingegen man FFN mit einer oder mehreren versteckten Schichten als *Mehrschichten-Netzwerke* (*Multilayer Feed-Forward Network*) bezeichnet. Während Einschicht-Netze lediglich linear separierbare Funktionen abbilden, können Mehrschichten-Netze beliebige Funktionen repräsentieren.

Das Trainieren bzw. Lernen der entsprechenden Parameter (Kantengewichte) neuronaler Netze erfolgt durch mehrfachen Durchspielens einer Trainings- bzw. Testmenge. Nach jedem Testfall wird die Fehlerrate ermittelt und anschließend durch entsprechende Veränderung der Kantengewichte verringert. Bei Perceptrons ist dies einfach möglich, wohingegen die Fehlerrate der versteckten Neuronen bei Mehrschichten-Netzen durch *Back-Propagation* ermittelt werden muss (siehe [RN03] für entsprechende Algorithmen). Im Rahmen des Trainings von neuronalen Netzen kommt es oft zum so genannten *Overfitting*, wenn die gewählte Netzstruktur zu komplex für die abzubildende Funktion ist [HTF01], d. h. wenn zu viele Neuronen, Schichten oder Kanten verwendet werden und sich die Gewichtung der Kanten quasi für jeden Testfall eine eigene Wertkombination merken kann. In diesem Fall kann das neuronale Netz nur zur exakten Rekonstruktion der Vergangenheit verwendet werden und eignet sich somit nicht zur Generalisierung bzw. Prognose unbekannter Eingabewerte. Um Overfitting zu verhindern gibt es mehrere praktikable Ansätze (siehe z. B. [RN03] oder [HTF01]).

Neuronale Netze werden meist zur Klassifikation oder Regression verwendet, wobei sie sich vor allem bei der Klassifikation und bei dynamischen Domänen als sehr leistungsfähig erwiesen haben (in [WB03] wird beispielsweise der erfolgreiche Einsatz neuronaler Netze im Marketing beschrieben). Zusätzlich können Perceptrons, die als Aktivierungsfunktion die Sigmoid-Funktion verwenden, probabilistisch interpretiert werden und eignen sich zur kompakten Repräsentation der kanonischen Wahrscheinlichkeitsverteilungen bzw. CPT innerhalb Bayes'scher Netze [RN03].

Als Nachteil erweist sich bei neuronalen Netzen das so genannte *Black-Box-Verhalten*, da die Interpretierbarkeit der Ergebnisse in der Praxis sehr eingeschränkt ist, d. h. es schwer nachzuvollziehen ist, wie Eingabe- und Ausgabewerte in Relation stehen. Allerdings gibt es mehrere Ansätze, eine interpretierbare Regelmenge aus dem in neuronalen Netzen codierten Wissen zu extrahieren (siehe z. B. [CZ00]). Ein weiterer großer Nachteil besteht in der Tatsache, dass bei der Modellierung von neuronalen Netzen Domänen- bzw. Apriori-Wissen nicht explizit in die Struktur einfließen kann. Ein Vorteil der BN gegenüber neuronaler Netze besteht darüber hinaus in der Fähigkeit Wahrscheinlichkeitsverteilungen modellieren zu können, bei denen beliebige Evidenzen bzw. Teilevidenzen als Grundlage des Schlussfolgerungsprozesses dienen, da die Eingabe-Neuronen und deren Anzahl bei neuronalen Netzen fest vorgegeben sind.

2.3.5.5 Lineare und nichtlineare Regression

Bei der klassischen Regressionsanalyse betrachtet man im Allgemeinen eine metrische Variable y, deren Wert von einer bzw. mehreren unabhängigen metrischen Variablen $x_1,\ldots,x_n$ bestimmt wird (einfache bzw. multiple Regression). Die Variable y wird in der Literatur

entweder als *abhängige Variable*, *Response-Variable*, *Kriterium* oder *Zielvariable* bezeichnet, während man die Variablen x_i einfach *unabhängige Variablen* oder *Prediktor-Variablen* nennt. Je nach Komplexität und Art der Abhängigkeitsfunktion unterscheidet man zwischen *linearer* und komplexer *nichtlinearer* Regression. Bei der einfachsten Form der linearen multiplen Regression nimmt man an, dass der Wert der Response-Variablen y von einer linearen Kombination der Prediktor-Variablen x_i der folgenden Form abhängt:

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + ... + \beta_n x_n + \varepsilon ,$$

wobei ε die Störgröße bzw. den Modellierungsfehler repräsentiert. Allgemein haben lineare Regressionsmodelle die Form:

$$y = \beta_0 + \beta_1 z_1 + \beta_2 z_2 + ... + \beta_n z_n + \varepsilon ,$$

wobei z_i beliebige Funktionen auf den ursprünglichen Prediktor-Variablen x_i sein können [MPV01]. Die Linearität bezieht sich dabei auf die unbekannten Parameter β_i. Auf diese Weise gehören unter anderem auch *polynomiale* Funktionen *n*-ter Ordnung zu den linearen Regressionsmodellen, wobei man die Ordnung in der Praxis so gering wie möglich halten sollte, um eine Explosion der Komplexität sowie das Overfitting des Modells zu verhindern.

In vielen Fällen reichen lineare Funktionen allerdings nicht zur Annäherung der beobachteten Daten aus [MPV01], beispielsweise wenn es sich um Differentialgleichungen handelt, oder die Funktionen nicht linear bezüglich der unbekannten Variablen β_i sind:

$$y = \frac{\beta_1 x}{x + \beta_2} + \varepsilon \quad \text{oder} \quad y = \beta_1 e^{\beta_2 x} + \varepsilon .$$

In diesem Fall eignen sich entsprechende nichtlineare Regressionsmodelle, wobei es in der Praxis oft sehr schwierig ist, für beliebige Daten ein passendes nichtlineares Regressionsmodell auszuwählen bzw. dessen Parametrisierung zu berechnen. In vielen Fällen versucht man daher, nichtlineare Modelle durch lineare Modelle zu approximieren.

Sowohl bei linearen als auch bei nichtlinearen Regressionsmodellen wird versucht durch geeignete Verfahren, die Funktionen bzw. die genauen Ausprägungen und Gewichtungen der Prediktor-Variablen aus vorhandenen Datenpunkten zu approximieren. Eines der bekanntesten Verfahren ist hierbei die *Methode der kleinsten Quadrate* (*Least Squares Method*).

Ein großer Nachteil der linearen Regressionsmodelle ist die Tatsache, dass man eine Unabhängigkeit zwischen den Prediktor-Variablen annimmt, die in der Praxis in den meisten Fällen nicht gegeben ist. Zusätzlich können Regressionsmodelle nur in Ausnahmefällen effizient für nichtlineare Probleme verwendet werden, wobei meist ein Experte notwendig ist, der je nach konkreter Fragestellung bzw. vorliegenden Daten ein geeignetes Regressionsmodell auswählen muss. Einen ausführlichen Überblick sowie Details über die

unterschiedlichen Regressionsverfahren bietet beispielsweise [MPV01].

2.3.5.6 Fuzzy Logik

Die Fuzzy Logik erlaubt Schlussfolgerungen auf der Basis logischer Ausdrücke, die die Wahrheitswerte im Intervall [0;1] der Mitgliedschaft von Objekten in so genannten *Fuzzy Sets* [ZLA65] angeben. Fuzzy Sets sind dabei *vage* oder *qualitative* Attribute bzw. Mengendefinitionen mit *unscharfen* Abgrenzungen. Um eine probabilistische Interpretation der Fuzzy Sets zu ermöglichen, können diese zu *Random Sets* erweitert werden, mit deren Hilfe man die Wahrscheinlichkeit modellieren kann, dass ein zufällig gewähltes Objekt Mitglied im Random Set ist. Fuzzy Logik kann um die Methodik des Fuzzy Control ergänzt werden, indem Funktionen zwischen reellen Eingabe- bzw. Ausgabewerten durch Fuzzy Regeln repräsentiert werden. Einer der größten Nachteile der Fuzzy Logik ist, dass sie im Gegensatz zu DSM bzw. BN keine mathematische Fundierung besitzt, so dass es durch Änderungen im Rahmen von Regeloptimierungen in der Praxis oft zu unvorhersehbaren bzw. nicht erwünschten Effekten kommen kann [JA96]. Darüber hinaus können mit Hilfe der Fuzzy Logik keine Korrelationen bzw. Antikorrelationen zwischen einzelnen Aussagen modelliert werden [RN03]. Ein Ansatz zur Modellierung von Unsicherheit in Fuzzy Systemen ist die *Möglichkeitstheorie* (*Possibility Theory*) [ZLA78], die viele Konzepte und Eigenschaften der Wahrscheinlichkeitstheorie übernommen hat [DP94].

2.4 Probabilistische Agenten und Holone

Im Folgenden werden *probabilistische Agenten* und die darauf aufbauenden *probabilistischen Multiagentensysteme* sowie *probabilistische Holone* eingeführt, an deren Entwicklung ich maßgeblich beteiligt bin und die die Grundlage der von mir vorgeschlagenen Kundenmodellierung bilden.

Probabilistische Agenten verfügen über spezielle probabilistische Wissensbasen, die im Rahmen meiner Arbeit aus verschiedenen Klassen bzw. Erweiterungen von Bayes'schen Netzen bestehen, die sich gegenüber anderen möglicher probabilistischer Modelle durch die aufgezeigten Vorteile auszeichnen (siehe Kapitel 2.3.5). Erste Ansätze zur Entwicklung von Agenten mit probabilistischen Wissensbasen - speziell Bayes'schen Netzen - finden sich unter anderem in [PJ88].

Ein probabilistisches Multiagentensystem besteht aus einer Menge von probabilistischen Agenten, die neben herkömmlichen Kommunikationsmethoden auch probabilistische Informationen in Form von probabilistischen Modellen bzw. Teilmodellen austauschen können (siehe beispielsweise [VKV02]).

Probabilistische Holone entstehen durch den Zusammenschluss probabilistischer Agenten,

der sich durch Anpassungen der in Kapitel 2.2 beschriebenen unterschiedlichen Holonisierungsarten realisieren lässt.

Probabilistische Holone und MAS eignen sich unter anderem zum agentenbasierten *Verteilten Data Mining* (*Distributed Data Mining* bzw. *DDM*) und zur agentenbasierten Simulation unter Unsicherheit.

2.4.1 Probabilistische Agenten und Multiagentensysteme

Einen probabilistischen Agenten definiere ich durch A_{prob}:= (*TM*, *FK*, *PSK*), wobei *TM* eine Menge $\{T_1,\ldots,T_n\}$ von Tasks und Methoden bzw. Aktionen ist, die der Agent durchführen kann, *FK* eine Menge $\{F_1,\ldots,F_p\}$ von Fakten bzw. Informationen darstellt, die das Faktenwissen (*Fact Knowledge*) des Agenten bildet, und *PSK* das *probabilistische Strukturwissen* (*Probabilistic Structural Knowledge*) des Agenten repäsentiert, das unter anderem aus dessen Faktenwissen generiert werden kann (siehe Abbildung 8).

Das *PSK* besteht aus einer Menge $\{PN_1,\ldots,PN_q\}$ von probabilistischen Netzen, wobei ein probabilistisches Netz durch *PN* := (*DMT*, *N*, *DK*) definiert ist. Dabei ist *N* ein strukturelles Netz, das anhand einer Trainingsmenge in Form einer so genannten *Data-Mining-Tabelle DMT* (*Data Mining Table*) und des Domänenwissens *DK* (*Domain Knowledge*) des Agenten generiert wird. Das Domänenwissen umfasst in diesem Zusammenhang Definitionen bezüglich der Semantik und des Kontextes der dem Netz zugrunde liegenden Knoten sowie domänenspezifische Netztopologien (siehe Kapitel 3.3.7). Als Netze können beispielsweise verschiedenartige Bayes'sche Netze oder verwandte Modelle wie Hidden-Markov-Modelle oder Kalman-Filter verwendet werden. In meiner Arbeit verwende ich *Verhaltensnetze* (siehe Kapitel 3.2.2), deren Grundlage die verschiedenen Klassen und Erweiterungen Bayes'scher Netze bilden und das Verhalten des Agenten oder der Umwelt in verschiedenen Situationen widerspiegeln. Das probabilistische Strukturwissen der Agenten wird aus dem Faktenwissen mit entsprechenden Lernverfahren der zugrunde liegenden Netze gelernt. Bei einer Aktualisierung des Faktenwissens werden die Netze entweder durch Adaption oder durch Neulernen angepasst. Schlussfolgerungs- bzw. Entscheidungsprozesse werden bei probabilistischen Agenten auf der Basis der Inferenz- bzw. Propagierungsalgorithmen der jeweiligen Netzmodelle durchgeführt.

Die Kommunikation zwischen probabilistischen Agenten basiert unter anderem auf dem Austausch von probabilistischen Informationen, wie zum Beispiel der Angabe der Wahrscheinlichkeit des Eintretens eines bestimmten Ereignisses bis hin zum Austausch kompletter probabilistischer Netze bzw. unvollständiger Teilnetze, die auf verschiedene Arten zu neuen Netzen verschmolzen werden können. Durch den Informationsaustausch kann es zur Entdeckung neuen probabilistischen Wissens kommen (*emergentes Wissen*).

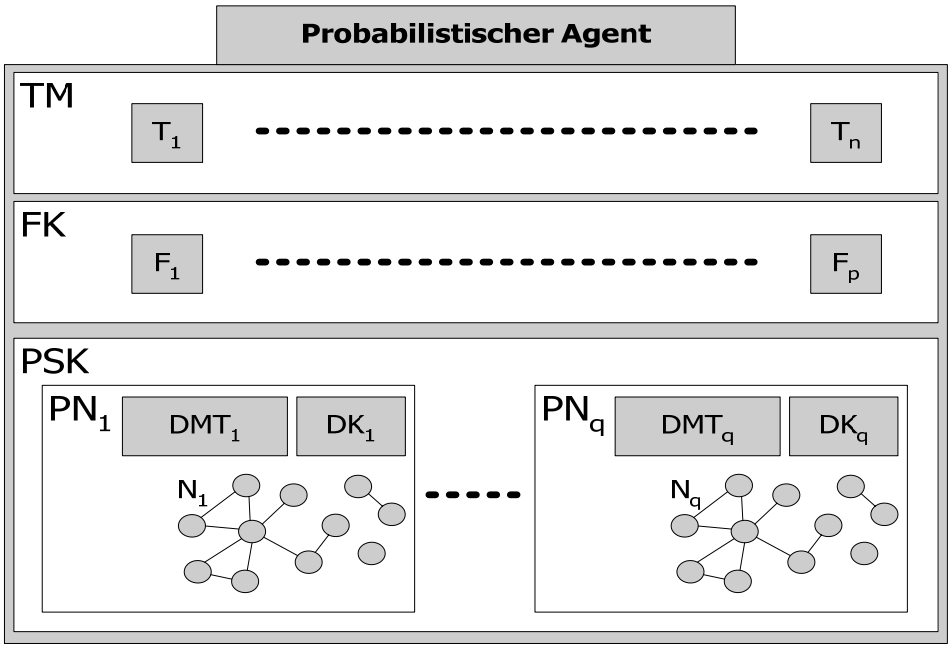


Abbildung 8: Probabilistische Agenten-Architektur

2.4.2 Probabilistische Holone

Im Folgenden zeige ich anhand zweier Beispiele kooperierender probabilistischer Agenten innerhalb eines Multiagentensystems, wie ich die verschiedenen in Kapitel 2.2 eingeführten Arten der Holonisierung bei probabilistischen Agenten realisiere und welche Vorteile sie gegenüber herkömmlichen MAS bringen können.

Beide Beispiele beziehen sich auf ein Multiagentensystem mit n probabilistischen Agenten, deren Aufgabe die Durchführung einer Sequenz von s Simulationsschritten ist. Das Ziel ist es, in jedem der s Simulationsschritte die Eintrittswahrscheinlichkeit $P(X)$ zu berechnen, dass Agent 1 eine bestimmte Tätigkeit X ausführt, wobei X durch eine Zufallsvariable eines probabilistischen Verhaltensnetzes in der Wissensbasis von Agent 1 repräsentiert wird. Ich nehme für meine Beispiele an, dass die Zustände bzw. Werte der Einflussfaktoren A, B und C ausschließlich von Agenten ermittelbar sind, die die entsprechenden Variablen kennen. Alle weiteren Variablen sind nicht direkt beobachtbar, d. h. die konkreten Werte dieser Variablen sind keinem der Agenten bekannt. Während der s Simulationsschritte können sich die Werte bzw. Beobachtungen bezüglich der Einflussfaktoren A, B und C ändern.

2.4.2.1 Beispiel 1: Einfacher Austausch von Wahrscheinlichkeitsverteilungen

Abbildung 9 zeigt die probabilistischen Agenten Agent 1 und Agent 2 innerhalb des oben beschriebenen Multiagentensystems. Die Wissensbasen bzw. die PSK der Agenten bestehen

jeweils aus einem vereinfachten probabilistischen Netz. Die von den Agenten beobachtbaren Variabeln *A*, *B* und *C* sind durch punktierte Linien gekennzeichnet.

Um die Aufgabe zu erfüllen, versucht Agent 1 in jedem Simulationsschritt die Wahrscheinlichkeitsverteilung der Zufallsvariable *X* zu berechnen. Dabei wird er feststellen, dass er zwar die Variable *X* kennt, diese aber von einer unbeobachtbaren Variablen *G* anhängig ist, die wiederum von der beobachtbaren Variable *C* und einer weiteren unbeobachtbaren Variablen *D* abhängig ist. Aus diesem Grund kann Agent 1 die Berechnung nicht alleine durchführen und ist daher auf die Hilfe anderer Agenten angewiesen. Agent 1 wird daher beginnen, innerhalb des Multiagentensystems nach geeigneten Kooperationspartnern zu suchen (in meinem Beispiel kennt nur Agent 2 den Knoten *D,* während der Knoten *G* keinem anderen Agenten bekannt ist). Je nach Art der Kooperation bzw. der Holonisierung sieht die Vorgehensweise dabei unterschiedlich aus.

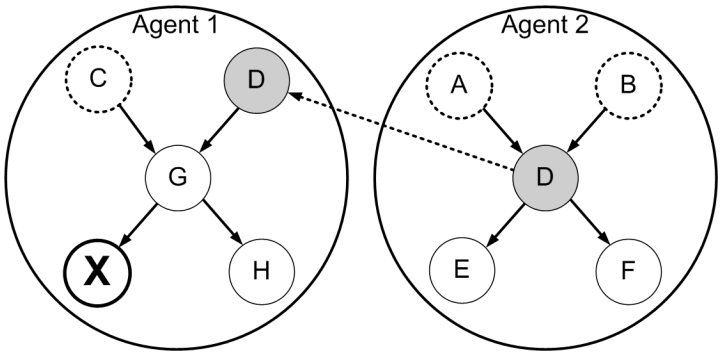


Abbildung 9: Beispiel zweier kooperierender probabilistischer Agenten

Im Falle eines herkömmlichen MAS muss Agent 1 im kritischsten Fall alle anderen *n*-1 Agenten des MAS fragen, ob sie die Wahrscheinlichkeitsverteilungen für die Knoten *G* oder *D* angeben können, die auf beobachtbaren Fakten basieren. Wenn Agent 1 Agent 2 befragt, wird dieser feststellen, dass er Knoten *D* kennt, der von den beobachtbaren Knoten *A* und *B* abhängig ist. Agent 2 wird daraufhin die korrekte Wahrscheinlichkeitsverteilung für *C* in Abhängigkeit der aktuellen Werte von *A* und *B* berechnen und an Agent 1 weiterleiten, der nun in der Lage ist, den gewünschten Erwartungswert für *X* zu berechnen.

Da die Aufgabe aber eine Sequenz von *s* Berechnungen des Wertes *X* umfasst, muss Agent 1 im schlechtesten Fall jedes Mal alle *n*-1 Agenten nach dem Werten von *G* und *D* fragen, was sich allerdings umgehen ließe, wenn sich Agent 1 merkt, dass Agent 2 über die gewünschten Informationen verfügt.

Um die Kommunikation zur Lösung der Aufgabe einzuschränken, kann temporär ein Holon, basierend auf einer Gruppe autonomer Agenten mit $h \leq n$ Agenten, gebildet werden, dessen Mitglieder mindestens einen der für Agent 1 zur Berechnung von $P(X)$ relevanten Knoten kennen müssen (in diesem Fall *X*, *G* oder *D*). Da Agent 1 und Agent 2 jeweils die

Anforderungen erfüllen, sind beide Mitglied des Holons. In diesem Fall muss Agent 1 im kritischsten Fall statt n-1 nur h-1 Agenten nach G und D fragen.

Mit Hilfe einer kopfgesteuerten Gesellschaft können der Prozess bzw. die Kommunikationskosten noch weiter vereinfacht bzw. gesenkt werden, falls der Kopf des Holons das Wissen und die Autorität besitzt, einen Plan zu entwerfen, der bestimmt, dass Agent 1 nur Agent 2 bezüglich des Wertes von Knoten D befragen muss.

Im Falle der vollständigen Verschmelzung der Mitglieder des Holons müssen alle Wissensbasen kombiniert werden, wobei die Verschmelzung in manchen Fällen sehr kostspielig bzw. sogar unpraktikabel sein kann. In der Praxis ist der Prozess der Verschmelzung in vielen Fällen nicht sehr aufwändig und kann zu neuem Wissen führen, wenn während der Verschmelzung neue Abhängigkeiten zwischen Knoten von verschiedenen Agenten entdeckt werden. Abbildung 10 zeigt auf der linken Seite das Resultat der vollständigen Verschmelzung der probabilistischen Netze der beiden Agenten. Man sieht, dass eine neue Abhängigkeit zwischen dem nicht beobachtbaren Knoten F und der zu berechnenden Variable X durch Kreuzkorrelationsanalyse erkannt wurde. Dieses neue Wissen resultiert quasi aus dem Zusammenspiel der beiden Agenten und wird daher *emergentes Wissen* genannt. Da der Holon neben dem Wissen auch die Fähigkeiten der ursprünglichen Agenten erbt, kann der entstandene Holon die Werte bzw. Zustände der beobachtbaren Variablen A, B und C ermitteln, um die entsprechende abhängige Wahrscheinlichkeit $P(X)$ zu berechnen.

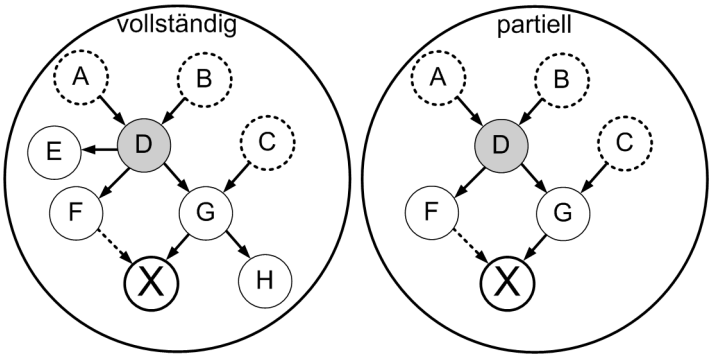


Abbildung 10: Vollständige und partielle Verschmelzung

Beim s-fachen Berechnen der Wahrscheinlichkeitsverteilung von X in Abhängigkeit der sich verändernden Werte von A, B und C kann das vollständig kombinierte Netz bei jedem Durchgang verwendet werden, so dass keine weitere Interaktion bzw. Kommunikation zwischen dem Holon und anderen Agenten nötig ist. Ferner kann die Ermittlung von X etwas differenzierter durchgeführt werden, da X nun zusätzlich in Abhängigkeit von F berechnet wird.

Im Falle der Holonisierung, basierend auf partiellem Klonen und Verschmelzen, werden nur die Teile der Wissensbasen bzw. PSK verschmolzen, die zur Lösung der Aufgabe relevant sind. Dazu stellen alle beteiligten Agenten eine Kopie bzw. einen Klon ihrer PSK her, deren relevanten Teile zu einem Holon verschmolzen werden. Abbildung 10 zeigt auf der rechten Seite das Ergebnis der partiellen Verschmelzung der probabilistischen Netze der Agenten 1 und 2. Wiederum wurde emergentes Wissen in Form der Abhängigkeit der Variable *X* von der unbeobachtbaren Variable *F* gefunden. Im Gegensatz zur vollständigen Verschmelzung können hier allerdings die Knoten *E* und *H* vernachlässigt werden, da sie keinen Einfluss auf die Variable *X* haben. Generell resultieren aus dem partiellen Verschmelzen in der Praxis deutlich kompaktere Holonennetze, da nur die relevanten Teile kombiniert werden. Durch das Erstellen von partiellen Kopien bzw. Klonen stehen die ursprünglichen Agenten parallel autonom für andere Aufgaben zur Verfügung.

Abbildung 11 zeigt zusammenfassend einen Vergleich der Bearbeitungsdauer der verschiedenen Holonentypen bezüglich der gestellten Aufgabe, wobei ich bei dem normalen MAS, der Gruppe autonomer Agenten und dem kopfgesteuerten Holon davon ausgehe, dass sich Agent 1 nach dem ersten Durchgang merkt, dass er nur Agent 2 nach der Wahrscheinlichkeitsverteilung von *D* fragen muss.

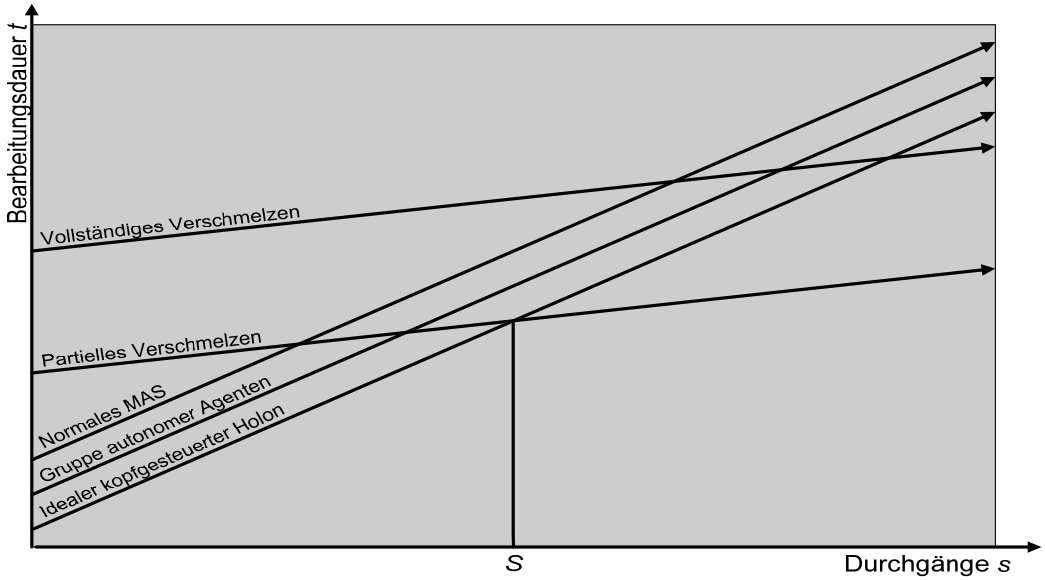


Abbildung 11: Vergleich der Bearbeitungsdauer der verschiedenen Holonentypen (1)

Wie gezeigt, benötigen die beiden Verschmelzungsholonen anfangs zwar mehr Berechnungszeit, da neben dem Finden relevanter Agenten deren Wissensbasen verschmolzen werden müssen, danach jedoch können die kombinierten Netze für jeden der *s* Durchgänge zur Berechnung von $P(X)$ verwendet werden, wohingegen bei den anderen

Holonenarten neben der Berechnung stets zwischen den Agenten kommuniziert werden muss. Daher ergibt es sich bei einem hinreichend großen *S*, dass partielles Verschmelzen am effizientesten ist, wobei man durch das Verschmelzen zusätzlich emergentes Wissen mit in die Berechnung einbeziehen kann, so dass das Resultat exakter ist und durch das Klonen die ursprünglichen Agenten parallel für andere Aufgaben verwendet werden können. Der Vorteil des partiellen Verschmelzens wird umso größer, je mehr Agenten zur Lösung einer Aufgabe nötig sind, da dadurch der Kommunikationsaufwand stark zunimmt.

2.4.2.2 Beispiel 2: Komplexer Austausch von probabilistischen Teilnetzen

Hier wird der Vorteil des partiellen Klonens und Verschmelzens noch deutlicher. Abbildung 12 zeigt die probabilistischen Agenten 1, 2 und 3, die zur Lösung der Aufgabe – also zur sequenziellen Berechnung des Erwartungswertes der Variable *X* – kooperieren müssen. Agent 2 wurde gegenüber Beispiel 1 so verändert, dass er nun keine Kenntnis mehr über Variable *B* besitzt, während Agent 3 zwar *B* aber nicht *A* kennt. Ich gehe davon aus, dass innerhalb des gesamten MAS neben Agent 1 nur Agent 2 und 3 Wissen über die Variablen *A*, *B*, *C* und *D* besitzen.

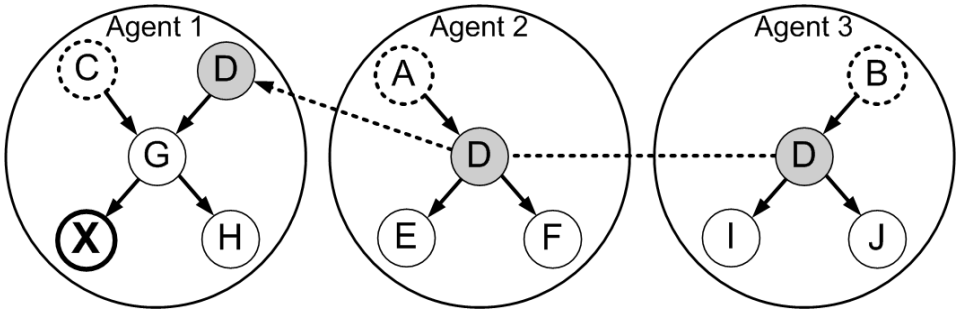


Abbildung 12: Beispiel dreier kooperierender probabilistischer Agenten

Wie in Beispiel 1 wird Agent 1 feststellen, dass er zwar die von *G* abhängige zu berechnende Variable *X* kennt, aber keine Information über *G* und *D* besitzt, die neben dem beobachtbaren Wert von *C* zur Berechnung von *P*(*X*) nötig sind. Agent 1 ist daher wiederum auf die Hilfe anderer Agenten angewiesen.

Im Falle eines normalen MAS muss Agent 1 im schlechtesten Fall wieder alle *n*-1 Agenten nach *G* und *D* fragen und wird dabei auf die Agenten 2 und 3 stoßen, die beide gewisse Teilkenntnisse über *D* besitzen: Agent 2 kennt die Relation zwischen *D* und *A*, während Agent 3 die zwischen *D* und *B* kennt. Um nun den exakten Wert von *D* in Abhängigkeit von *A* und *B* berechnen zu können, müssen komplexere Informationen zwischen den drei Agenten ausgetauscht werden. In diesem Beispiel besteht eine Möglichkeit darin, dass Agent 2 und Agent 3 jeweils das für die Berechnung relevante Teilnetz $A \rightarrow D$ bzw. $B \rightarrow D$ an Agent 1 senden. Agent 1 ist dann in der Lage, die beiden Teilnetze mittels Verschmelzung zu einem

einzigen Netz zu kombinieren (siehe Abbildung 13 links), das den Erwartungswert von *D* in Abhängigkeit von *A* und *B* berechnen kann, auf dessen Grundlage sich der zu berechnende Erwartungswert von *X* ermittelt lässt. Im schlechtesten Fall muss dieser Prozess also für jeden der *s* Berechnungsdurchgänge erneut durchgeführt werden, wobei die Kommunikation eingeschränkt werden kann, indem Agent 1 sich merkt, dass er nur mit Agent 2 und 3 interagieren muss.

Mit Hilfe einer autonomen Gruppe von Agenten mit $h \leq n$ Mitgliedern, deren Mitglieder mindestens einen der für Agent 1 zur Berechnung von $P(X)$ relevanten kennen müssen (hier *X*, *G* oder *D*), kann die anfängliche Suche von Agent 1 wieder von n-1 auf h-1 Anfragen reduziert werden.

Bei der kopfgesteuerten Gesellschaft können auch hier wieder die Kommunikationskosten reduziert werden, wenn der Kopf die Autorität sowie das Wissen besitzt, zu bestimmen, dass Agent 1 nur mit Agent 2 und Agent 3 kommunizieren muss.

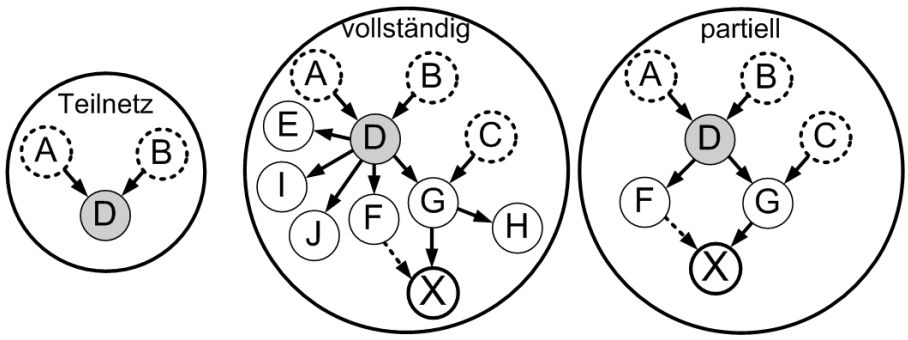


Abbildung 13: Nötiges Teilnetz, vollständige und partielle Verschmelzung

Bei der Holonisierung mittels vollständiger Verschmelzung ändert sich gegenüber dem ersten Beispiel, dass nun drei statt zwei probabilistische Netze zu einem einzigen Netz verschmolzen werden müssen, das gegenüber Beispiel 1 um die Knoten *I* und *J* erweitert wurde (siehe Abbildung 13). Statt eines Austauschs von einfachen Wahrscheinlichkeitswerten handelt es sich hier um den Austausch von probabilistischen Teilnetzen. Wiederum wurde beim Verschmelzungsprozess emergentes Wissen entdeckt (Kante von Knoten *F* nach *X*).

Beim partiellen Klonen und Verschmelzen hingegen erhält man nach dem Verschmelzen der PSK der drei Agenten exakt dasselbe Netz wie in Beispiel 1 (siehe Abbildung 13 rechts), da die relevanten Teile gleich geblieben sind. In Abbildung 14 ist der Vergleich der Bearbeitungsdauer der verschiedenen Formen der Holonisierung für Beispiel 2 dargestellt.

Die Anfangsbearbeitungsdauer des normalen MAS, der Gruppe autonomer Agenten sowie des idealen kopfgesteuerten Holonen, hat sich gegenüber Beispiel 1 jeweils erhöht, da nun nicht mehr einfache Wahrscheinlichkeitsverteilungen zwischen zwei, sondern komplexe

Teilnetze zwischen drei Agenten ausgetauscht werden müssen. Aus demselben Grund hat sich auch die Steigungsrate der Graphen der drei Holonisierungsarten etwas erhöht (ich gehe auch hier davon aus, dass sich bei allen drei Verfahren Agent 1 merkt, dass er nur mit Agent 2 und 3 kommunizieren muss). Die Anfangsbearbeitungsdauer der vollständigen bzw. partiellen Verschmelzung hat sich ebenfalls leicht erhöht, da nun drei statt zwei Netze verschmolzen werden. Danach verhalten sich beide Holonenformen nahezu äquivalent zum ersten Beispiel, wobei die Berechnung bei der vollständigen Verschmelzung minimal länger dauert, da nun „nebenbei" auch die Werte der Knoten *I* und *J* berechnet werden. Wie in Abbildung 14 ersichtlich, verringert sich gegenüber Beispiel 1 die Anzahl *S* an Durchgängen deutlich, bei der das partielle Klonen und Verschmelzen effizienter ist als alle anderen Holonenformen, wobei zusätzlich emergentes Wissen mit in die Berechnung einfließt und das Ergebnis somit exakter wird.

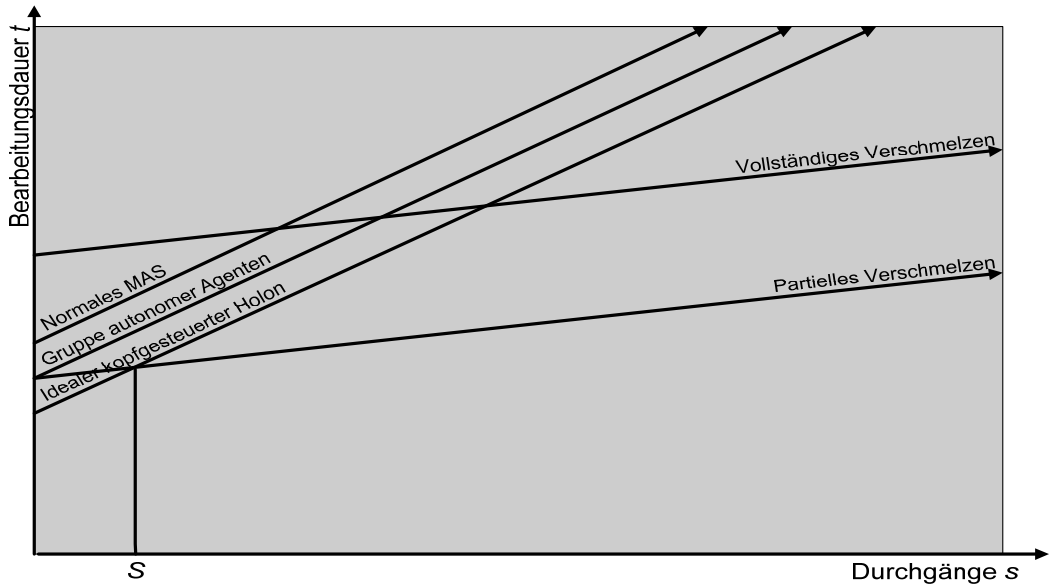


Abbildung 14: Vergleich der Bearbeitungsdauer der verschiedenen Holonentypen (2)

2.4.2.3 Anwendungsgebiete der verschiedenen probabilistischen Holonenformen

Im Allgemeinen haben alle vorgestellten Holonenformen Vorteile in gewissen Situationen, wobei der Hauptunterschied in dem Trade-Off zwischen den Kommunikationskosten, der Gewinnung emergenten Wissens, der Wahrung der Konsistenz und den Verschmelzungskosten besteht. Daher schlage ich die Entwicklung umfassender MAS-Softwaresysteme vor, die möglichst alle der hier vorgestellten Holonformen unterstützen, um für den jeweiligen Anwendungsfall die effizienteste Form wählen zu können. Ideal wäre auch die Entwicklung geeigneter Kriterien und Maße, um die Entscheidung dynamisch während

der Laufzeit treffen bzw. vereinfachen zu können, d. h. dass die Agenten selbstständig dynamisch abschätzen können, welche Holonenform in der aktuellen Situation überhaupt durchführbar bzw. am besten ist.

In Bereich der agentenbasierten Simulation mit probabilistischen Agenten, deren Netze das Verhalten des Agenten bezüglich bestimmter Einflussfaktoren repräsentieren, wobei die Netze anhand realer bzw. realistischer Daten gelernt wurden, erweist sich der Ansatz des partiellen Klonens und Verschmelzens als äußerst effizient. Die Konsistenz zwischen den Agentenklonen und den ursprünglichen Agenten kann in diesem Fall sehr leicht gewahrt bleiben, da die bereits gelernten Netze nur zur Simulation bzw. Berechnung verwendet werden und somit nicht verändert werden. Durch das Verschmelzen kann emergentes Wissen entdeckt und in die Simulation mit einfließen, wobei die Performanz bei mehrmaligen Simulationen deutlich über der der anderen Holonenformen liegt. Der Performanzunterschied ist umso größer, je mehr Agenten zur Lösung kooperieren müssen und je komplexer das auszutauschende Wissen ist. Um den Kommunikationsaufwand vor der partiellen Verschmelzung bzw. des Austauschs von probabilistischen Teilnetzen zu minimieren, können entweder Ansätze zur Verringerung des Informationsaustauschs bei verteilten probabilistischen Netzen (siehe beispielsweise [SLC03]) als auch ein Algorithmus zur Entdeckung maximaler probabilistischer Subnetze verwendet werden (siehe [SS04] bzw. [SS05]).

In anderen Fällen kann es in der Praxis vorkommen, dass der Verschmelzungsprozess oder die Wahrung der Konsistenz zu teuer bzw. gar nicht erst möglich sind. Hier sind die kopfgesteuerte Gesellschaft bzw. die Gruppe autonomer Holonen sehr gute Alternativen, da gegenüber den normalen MAS Kommunikationskosten gespart werden können.

2.4.2.4 Probabilistische Durchschnittsbildung, Additions- und Relationsverschmelzung

Je nach Problemstellung können verschiedene Arten der vollständigen bzw. partiellen Verschmelzung notwendig sein. Drei der wichtigsten Formen für meine Arbeit sind die so genannte *Durchschnittsbildung,* die *Additions*- sowie die *Relationsverschmelzung*.

Abbildung 15 zeigt entsprechende Beispiele, die sich auf die Verschmelzung zweier gleichartiger Agenten beziehen, die jeweils dieselbe Struktur basierend auf den Knoten *A* bis *E* besitzen.

Bei der Durchschnittsbildung werden die probabilistischen Netze der Agenten „gemittelt“, indem für bestimmt Knoten durchschnittliche Werte verwendet werden, um das durchschnittliche Verhalten bzw. Wissen der Agenten zu repräsentieren. Durchschnittsnetze setze ich beispielsweise bei der Analyse von Kundengruppen ein, da dort ein Durchschnittsnetz das durchschnittliche Verhalten der Kundengruppe bzw. deren Median modelliert.

Bei der Additionsverschmelzung werden beide probabilistischen Netze der Agenten quasi addiert, indem die Werte bzw. Wahrscheinlichkeitsverteilungen einiger Knoten „summiert“ werden. Das Ergebnis ist ein Netz, das das aggregierte Verhalten bzw. Wissen einer Gruppe von Agenten repräsentiert, das im Rahmen meiner Arbeit unter anderem für agentenbasierte Simulationen des aggregierten Verhaltens von Kundengruppen verwendet wird.

In anderen Fällen ist es interessant, Abhängigkeiten zwischen einzelnen gleichartigen Agenten zu entdecken und in agentenbasierten Simulationen oder verteiltem Data Mining einfließen zu lassen. In diesen Fällen eignet sich die Relationsverschmelzung, bei der probabilistische Netze so verschmolzen werden, dass manche Knoten wie bei der Durchschnittsbildung vereinigt werden, wohingegen zwischen anderen Knoten Abhängigkeiten durch Kreuzkorrelationsanalysen festgestellt werden und in das resultierende Netz übernommen werden. Abbildung 15 zeigt auf der rechten Seite, dass bei der Verschmelzung der probabilistischen Netze von Agent 1 und 2 die Knoten *A*, *B* und *C* vereinigt wurden, während zwischen den jeweiligen Instanzen der Knoten *D* und *E* Abhängigkeiten gefunden wurden.

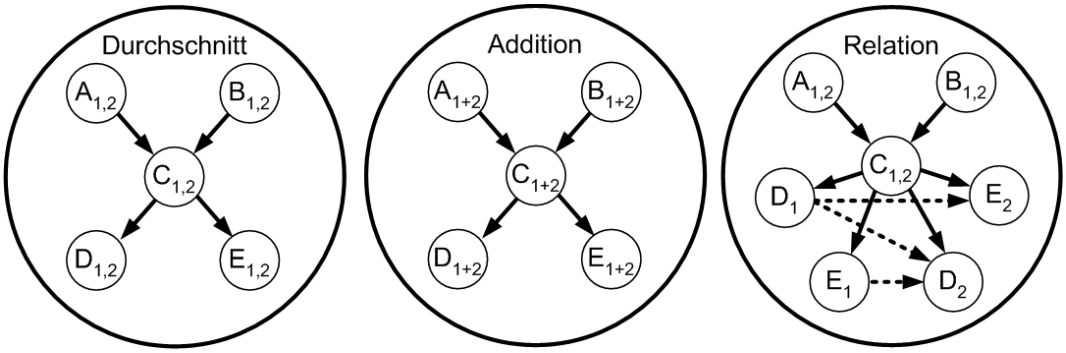


Abbildung 15: Durchschnittsbildung, Additions- und Relationsverschmelzung

Es sind darüber hinaus Kombinationen der verschiedenen Verschmelzungsarten möglich, wobei für jeden Knoten bzw. Knotentyp angegeben werden kann, auf welche Art er gegebenenfalls verschmolzen werden soll.

In Kapitel 3.3.5 zeige ich konkrete Vorgehensweisen zur Verschmelzung von probabilistischen Verhaltensnetzen, wobei ich dabei auf jede der hier aufgeführten Verschmelzungsarten im Detail eingehe, während ich in Kapitel 3.3.7.1 erläutere, wie Domänenwissen genutzt werden kann, um den Verschmelzungsprozess der Netze effizienter zu gestalten.

3 Probabilistische Modellierung individuellen Konsumentenverhaltens

Im Folgenden präsentiere ich ein Modell des individuellen Konsumentenverhaltens basierend auf probabilistischen Holonen. In Kapitel 3.1 führe ich in das Thema Konsumentenverhalten ein und beschreibe bekannte Modelle aus dem Marketing und der Psychologie. Anschließend erläutere ich in Kapitel 3.2 die im Rahmen meiner Arbeit entstandene Abbildung einzelner realer Kunden auf probabilistische Kundenagenten. Kapitel 3.3 zeigt, wie das individuelle Kaufverhalten aus historischen Kundendaten gelernt und anschließend in Form probabilistischer Verhaltensnetze für simulationsbasierte Analysen und Prognosen zur Verfügung steht, die ich in Kapitel erläutere.

3.1 Konsumentenverhalten

Der Begriff *Konsumentenverhalten* umfasst im engeren Sinne das Verhalten von Menschen beim Kauf und Konsum von wirtschaftlichen Gütern [KRW03]. Im weiteren Sinne wird Konsumentenverhalten als das Verhalten der *Letztverbraucher* von materiellen und immateriellen Gütern definiert, so dass auf diese Weise auch Patienten, Wähler, Spender usw. als Konsumenten angesehen werden [SW04]. Im Folgenden beziehe ich mich auf das Konsumentenverhalten im engeren Sinne und werde die Begriffe *Käufer* und *Konsument* bzw. *Kaufverhalten* und *Konsumentenverhalten* aus Vereinfachungsgründen synonym verwenden, obwohl Käufer und Verbraucher nicht unbedingt identisch sein müssen, z. B. im Falle von Geschenken oder Mitbringseln. Kaufentscheidungen bestehen im Allgemeinen aus einer *Vorkauf-*, *Kauf-* und *Nachkaufphase* [MH92] und lassen sich anhand verschiedener Kriterien in unterschiedliche Arten differenzieren, z. B. nach dem *Grad der kognitiven Kontrolle* in *extensive Kaufentscheidungen* (z. B. Hauskauf), *limitierte Kaufentscheidungen*, *habituelle Käufe* sowie *Impulskäufe* [KRW03]. Das Kaufverhalten einzelner Personen wird durch eine Vielzahl von *Umwelteinflüssen* (*externen Determinanten*) und *psychischen Faktoren* (*internen Determinanten*) beeinflusst.

3.1.1 Umwelteinflüsse und externe Determinanten

Die externen Determinanten des Käuferverhaltens untergliedern sich in *situative* bzw. *physische* und *soziale* Determinanten. Zu den situativen und physischen Faktoren des Kaufverhaltens gehören unter anderem:

- natürliche Umwelteinflüsse wie Landschaft, Klima und Wetter,
- die vom Menschen geschaffene Umwelt wie Städte, Gebäude und Infrastruktur,
- Lage, Layout und Atmosphäre des Geschäftes,
- Zeitfaktor (Tageszeit, Jahreszeit/Saison, Dringlichkeit) und
- Zweck des Kaufs (z. B. Eigenbedarf oder Geschenk).

Die sozialen Determinanten entspringen dem sozialen Umfeld des Konsumenten. Sie lassen sich nach folgenden Kriterien gliedern [SW04]:

- *Kultur*, d. h. das Wertesystem bzw. der Wertewandel innerhalb einer Gesellschaft.
- *Subkultur*, d. h. durch gemeinsame Erfahrungen oder Merkmale geprägte Gruppen innerhalb einer Kultur (z. B. religiöse Minderheiten oder nationale Volksgruppen).
- *Soziale Schicht*, d. h. eine relativ homogene und stabile Untergruppe einer Gesellschaft mit ähnlichen Werten, Interessen, Verhaltensweisen und Lebensstil.
- *Bezugsgruppen*, d. h. Primärgruppen wie Familie, Freunde, Nachbarn und Kollegen sowie Sekundärgruppen (Berufsverbände, Parteien, Vereine etc.) und Leitbild- bzw. Antileitbildgruppen wie Popstars, Schauspieler oder Sportler.

3.1.2 Psychische Prozesse und interne Determinanten

Die internen bzw. psychischen Vorgänge können in *aktivierende Prozesse* und *kognitive Prozesse* gegliedert werden. Aktivierende Prozesse beschreiben sämtliche psychischen Vorgänge, die mit inneren Erregungen und Spannungen verbunden sind und das Verhalten antreiben, wohingegen bei kognitiven Vorgängen Informationen aufgenommen, verarbeitet und gespeichert werden [KRW03].

Aktivierende Prozesse gelten als notwendige Voraussetzung für sämtliche menschlichen Leistungen und somit auch für Kaufentscheidungen. Sie umfassen im Wesentlichen folgende aufeinander aufbauenden hypothetischen Konstrukte:

- Emotion,
- Motivation,
- Einstellung und
- Zufriedenheit.

Unter einer *Emotion* versteht man einen *zentralnervösen Erregungszustand*, der mehr oder minder bewusst erlebt und kognitiv interpretiert wird, wobei er als angenehm oder unangenehm empfunden wird. Emotionen können anhand verschiedener Kriterien in Klassen eingeteilt werden (siehe beispielsweise [ICE94]).

Motivation bezeichnet kognitiv *handlungs- bzw. zielorientierte* Emotionen, Triebe und Motive, wie z. B. Sparsamkeit, Prestige, Einhaltung von Normen, Erotik, Konsistenz oder Risikoneigung. Ziel einer Motivation ist die Vermeidung unangenehmer und die Steigerung

angenehmer Emotionen und Gefühle, wobei es dabei zu drei verschiedenen Motiv-Konflikten kommen kann:

- *Appetenz-Appetenz-Konflikt*, d. h. es kollidieren zwei oder mehr Motive mit unterschiedlicher Zielorientierung aufeinander.
- *Aversions-Aversions-Konflikt*, d. h. man muss sich zwischen zwei oder mehr ungeliebten oder unangenehmen Optionen entscheiden.
- *Appetenz-Aversions-Konflikt*, hier verhindert das Erreichen der Befriedigung eines Motivs die Möglichkeit ein oder mehrere andere Motive zu befriedigen.

Eine *Einstellung* wird gekennzeichnet durch die *kognitive Beurteilung bzw. positive oder negative Bewertung* von Dingen und Handlungen, also inwieweit bestimmte Objekte oder Handlungen zur Befriedigung von Motiven geeignet sind.

Die Definition von *Zufriedenheit* erweitert die der Einstellung um *konkrete Objekterfahrung*, d. h. bei Einstellungen können Objektbeurteilungen spekulativ sein, wohingegen Zufriedenheit eine konkrete Erfahrung mit dem Objekt voraussetzt. Zufriedenheit ist dabei das Ergebnis eines komplexen Informationsverarbeitungsprozesses, bei dem die Bewertung eines Objekts oder einer Handlung aus einem *Soll/Ist-Vergleich* resultiert.

Aktivierende Prozesse können nach [KRW03] durch folgende Arten von *Reizen* ausgelöst werden:

- *Physikalische* Reize, z. B. Farbe, Größe, Kontrast, Klarheit einer Anzeige.
- *Kognitive* Reize, z. B. Überraschungen, Widersprüche, gedankliche Konflikte.
- *Emotionale* Reize, d. h. biologische Schlüsselreize, die weitgehend automatisch vorprogrammierte Reaktionen auslösen, die gedanklich nicht kontrolliert werden können und visueller (Sehen), akustischer (Hören) oder olfaktorischer (Riechen) Natur sein können.

Kognitive Prozesse des Konsumenten umfassen sämtliche Vorgänge im psychischen Bereich, mit denen ein Individuum sich selbst und seine Umwelt erkennt und das eigene Verhalten kontrolliert und willentlich steuert [SW04]. Zu den kognitiven Prozessen gehören:

- Informationsaufnahme (Wahrnehmung)
- Informationsverarbeitung (Denken und Entscheiden)
- Informationsspeicherung (Lernen und Gedächtnis)

Prozesse der *Informationsaufnahme* gliedern sich in *interne* und *externe* Informationsaufnahme. Bei der *internen Informationssuche* greift der Konsument ausschließlich auf seine eigenen Erfahrungen aus früheren Kaufentscheidungen zurück. Im Falle der externen Informationsaufnahme unterscheidet man zwischen der *absichtslosen*

Übernahme von Informationen (z. B. durch unabsichtliche Aufnahme von Werbebotschaften) und der *aktiven Informationssuche*, bei der der Kunde bewusst und absichtlich nach Informationen über bestimmte Produkte sucht (z. B. durch das Studieren von Testberichten, Einkaufsberatern, Produktkatalogen oder Werbebroschüren).

Im Rahmen der *Informationsverarbeitung* analysiert der Kunde die gewonnen Informationen mit Hilfe von *Heuristiken* und gelangt so zu einem *Qualitätsurteil* [SW04]. Die verwendeten Heuristiken lassen sich in *kompensatorische* und *nicht-kompensatorische Entscheidungsregeln* unterteilen. Dabei kann die kundenindividuelle Beurteilung der Qualität eines Produktes *KQ*, im Falle der kompensatorischen Heuristik, und *NKQ* im Falle der nicht-kompensatorischen Entscheidungsregel, durch folgende Formeln ausgedrückt werden (in Anlehnung an [KRW03]):

$$KQ = \sum_{k=1}^{n} E_k \quad \text{und} \quad NKQ = \sum_{k=1}^{n} E_k \cdot W_k \,,$$

wobei E_k die kundenindividuelle Beurteilung der Produkteigenschaft k und W_k die kundenindividuelle Wichtigkeit dieses Attributes ausdrückt. Bei den kompensatorischen Entscheidungsregeln können negative Objekteigenschaften durch positive Eigenschaften ausgeglichen werden, wohingegen attributspezifische Schwächen bei nicht-kompensatorischen Heuristiken bereits zur Ablehnung des Objektes führen können.

Informationsspeicherung setzt sich zusammen aus *Lernen* und *Gedächtnis*, wobei das Gedächtnis als Grundlage des Lernens zu sehen ist, indem Informationen abgespeichert und jederzeit wieder verfügbar gemacht werden können. Es kann in *sensorischen Speicher*, *Kurzzeitgedächtnis* und *Langzeitgedächtnis* unterteilt werden [KRW03]. Unter Lernen versteht man die relativ dauerhafte Verhaltensänderung aufgrund gesammelter Erfahrungen oder Beobachtungen, wobei drei Arten des Lernens unterschieden werden [SW04]:

- *Lernen durch Konditionierung* (Stimulus-Response-Verknüpfungen)
- *Lernen durch Nachahmung* (anderer Personen)
- *Kognitives Lernen* (Erkennen von Mittel-Zweck-Beziehungen)

3.1.3 Modelle des Konsumentenverhaltens

Modelle des Konsumentenverhaltens haben die Aufgabe, das reale Verhalten von Käufern und Konsumenten vereinfacht aber hinreichend genau abzubilden, indem sie Verhaltensvariablen miteinander in Verbindung setzen, um das Zustandekommen des Gesamtverhaltens theoretisch zu erklären [KRW03]. Verhaltensmodelle können nach den verschiedensten Kriterien gegliedert werden. Grundsätzlich unterscheidet man zwischen den *naturwissenschaftlich* geprägten *Black-Box-Modellen* bzw. *Stimulus-Response-Modellen (SR-Modelle)* und den *sozialpsychologisch* beeinflussten *Struktur-* bzw. *Stimulus-Organism-Response-Modellen (SOR-Modelle)* [SW04]. Darüber hinaus ist eine Einteilung der

Verhaltensmodelle anhand der Abstraktion in *Partial-* und *Totalmodelle* möglich, wobei komplexe Totalmodelle versuchen, das gesamte Kauf- oder Entscheidungsverhalten abzubilden und Partialmodelle sich nur auf Ausschnitte des Verhaltens beziehen [KRW03].

3.1.3.1 Black-Box-Modelle / Stimulus-Response-Modelle

Black-Box- bzw. Stimulus-Response-Modelle basieren auf dem *Behaviorismus* [NAF74], da sie voraussetzen, dass sich objektive Erkenntnisse über die Bestimmungsfaktoren des Verhaltens ausschließlich über beobachtbare Reiz-Reaktions-Prozesse gewinnen lassen und sie sämtliche Prozesse in der Psyche des Kunden bewusst ignorieren [SW04]. SR-Modelle bilden daher ausschließlich beobachtbare Stimuli und Reize (beispielsweise Produktpreise, Werbung oder Produktplatzierung) auf messbare Kundenreaktionen ab (z. B. Kauf eines Produktes, oder der Wechsel des Anbieters). Bekannte Stimulus-Response-Modelle sind unter anderem *regressionsanalytische Modelle* und *stochastische Prozessmodelle*.

Bei den *regressionsanalytischen Modellen* werden die *unabhängigen* Variablen der Reize und Stimuli mit den *abhängigen* Variablen der Reaktionen in Beziehung gesetzt. Die Bandbreite der Abhängigkeitsmodellierung zwischen den Variablen reicht dabei von einfachen *linear-additiven* über *multiplikative* bis hin zu mathematisch sehr aufwändigen Verfahren. Regressionsanalytische Modelle bestechen durch methodische Pragmatik, universelle Einsetzbarkeit, praxisnahe Variablen und die vergleichsweise einfache Datenbeschaffung und Operationalisierung [SW04].

Bei *stochastischen Prozessmodellen* werden Beziehungen zwischen Reizen und Reaktionen mittels Zufalls- bzw. Wahrscheinlichkeitsmechanismen repräsentiert, d. h. die Verfahren fokussieren auf die Wahrscheinlichkeit mit der ein Kunde auf gewisse Reize reagiert. Bisherige Anwendungen beziehen sich z. B. auf die Wahl von Marken, Einkaufsstätten oder Einkaufszeitpunkten, sowie Wiederkaufrate, Markentreue bzw. Markenwechsel [SW04].

3.1.3.2 Struktur-Modelle / Stimulus-Organism-Response-Modelle

Im Gegensatz zu Black-Box- bzw. Stimulus-Response-Modellen versuchen die Struktur- bzw. Stimulus-Organism-Response-Modelle (SOR-Modelle), die Psyche des Konsumenten zu erfassen und kognitiv kontrollierte Kaufentscheidungen darzustellen. Zu den bekanntesten SOR-Modellen im Bereich des Marketings gehören die Modelle von *Nicosia* [NFM66], von *Engel*, *Blackwell* und *Miniard* [EBM00], von *Howard* und *Sheth* [HS69], und von *Bettmann* [BJR79].

Das *Modell von Nicosia* ist ein dreistufiges Verfahren basierend auf Flussdiagrammen, das potentielle Reaktionen von Konsumenten auf neue Marken, Produkte oder Werbemaßnahmen repräsentiert [WG01]. Der Prozess beginnt mit der Annahme, dass ein Konsument auf ein

neues Angebot aufmerksam wird und dabei eventuell, aufgrund bestimmter Produkteigenschaften, Interesse entwickelt. Im positiven Fall schließt sich eine Suche nach Alternativen sowie deren Bewertung an, die schließlich in einer Kaufentscheidung mündet. In der Nachkaufphase evaluiert der Konsument seinen Kauf und gibt an, ob er mit seiner Kaufentscheidung zufrieden war.

Das *Strukturmodell von Engel, Blackwell und Miniard* versucht das Zusammenwirken der zu Kaufentscheidungen führenden psychischen Vorgänge zu beschreiben, ist aber in der Praxis aufgrund seiner Komplexität nur eingeschränkt anwendbar [SW04]. Es unterscheidet, in Anlehnung an das klassische Phasenmodell, mehrere aufeinander folgende Prozessphasen einer Kaufentscheidung [KRW03]:

- Problemerkenntnis
- Informationssuche
- Informationsverarbeitung
- Alternativenbewertung
- Auswahl einer Alternative
- Entscheidung: Kauf
- Entscheidungsfolgen

Das *Strukturmodell von Howard und Sheth* versucht, das Zustandekommen des Kaufverhaltens anhand unterschiedlicher Konstellationen der in das Modell aufgenommenen Variablen zu erklären und dabei Nachteile des Phasenmodells zu vermeiden [KRW03]. Es werden drei Klassen von Modellvariablen unterschieden: *Inputvariablen* (beobachtbare und messbare Reizeinflüsse), *Outputvariablen* (beobachtbare und messbare Reaktionen) sowie Variablen für die Modellierung von *hypothetischen Vorgängen* in der Psyche des Konsumenten. Eine detaillierte Beschreibung sowie Kritik bezüglich des für die Praxis zu komplexen und nicht anwendbaren Modells geben beispielsweise [MJ78] und [WP81].

Das besonders kognitiv ausgeprägte *Strukturmodell von Bettmann* verwendet *Entscheidungsheuristiken*, die die Aufnahme und Verarbeitung von Informationen lenken [KRW03], wobei die Heuristiken mittels so genannter *Entscheidungsnetze* modelliert werden, die den *Entscheidungsbäumen* der *Entscheidungstheorie* entsprechen. In Bettmanns Modell beziehen sie sich auf die Wahrnehmung und Verarbeitung von Produkteigenschaften sowie situativen Reizen, wobei die Kaufentscheidung aufgrund der Ablehnung bzw. Zustimmung von Alternativen gefällt wird.

Die beschriebenen Strukturmodelle versuchen jeweils, mit einem einzigen Modell das komplexe Kaufverhalten von Konsumenten zu beschreiben, indem sie verschiedene verhaltenswissenschaftliche Theorien und Konzeptionen miteinander kombinieren und Bezüge zu empirischen Ergebnissen der Konsumentenforschung herstellen [KRW03]. Zur Erklärung bestimmter Verhaltensweisen sind diese umfassenden Strukturmodelle allerdings nur äußerst eingeschränkt nützlich, da sie nicht das gesamte Kaufverhalten von Konsumenten so umfassend beschreiben können, dass daraus empirisch validierbare Vorhersagen entstehen [KRW03]. Die Modelle sind in der Praxis aufgrund ihrer Komplexität nur eingeschränkt

einsetzbar, und es fehlen Erkenntnisse und Variablen bezüglich der Modellierung von Impulskäufen [KRW03]. Die Konsumentenforschung konzentriert sich daher auf Modelle, die nur bestimmte Teilverhalten der Konsumenten abbilden und dadurch eine durchgehende Überprüfung von Hypothesen ermöglichen, wobei eine Beschränkung des Verhaltensausschnittes nur bei Kenntnis der notwendigen Faktoren und Variablen zu aussagekräftigen und anwendbaren Ergebnissen führen kann [KRW03].

3.2 Kundenmodellierung mit probabilistischen Holonen

Die vorgestellten Modelle des Konsumentenverhaltens haben einige Nachteile. Stimulus-Organism-Response-Modelle setzen theoretische Annahmen über interne psychologische Prozesse voraus, deren genauen Ausprägungen bei konkreten Kunden nicht messbar sind. Stimulus-Response-Modelle hingegen modellieren zwar messbare Reaktionen auf beobachtbare Einflüsse, verwenden allerdings meist Verfahren, die nur eine geringe Anzahl von Einflussfaktoren quantitativ berücksichtigen und Abhängigkeiten nur linear modellieren können. Darüber hinaus eignet sich die Mehrzahl der in den Modellen verwendeten Analyseverfahren nicht für Prognosen und umgekehrt [ABKW02].

Es fehlen somit geeignete softwaregestützte Konsumentenmodelle, mit deren Hilfe sich komplexe (eventuell nichtlineare) individuelle Verhaltensmuster automatisiert aus realen Daten extrahieren lassen, die sowohl für quantitative Analysen als auch für simulationsbasierte Prognosen von „Was-wäre-wenn-Szenarien“ verwendet werden können.

Um ein Kundenmodell zu verwirklichen, das die genannten Ansprüche erfüllt, habe ich im Rahmen dieser Arbeit ein Stimulus-Response-Modell zur Modellierung individueller Kunden entwickelt, das sowohl regressionsanalytische als auch stochastische Prozessmodelle unterstützt und auf beobachtbaren und quantitativ messbaren Fakten basiert. Ein einzelner Kunde und dessen Kaufverhalten werden dabei durch einen *Kundenagenten* (*Customer Agent*) repräsentiert. Ein Kundenagent ist ein probabilistischer Agent (siehe Abbildung 16), dessen Faktenwissen und probabilistisches Strukturwissen aus Informationen über den Kunden gewonnen werden, die in einer Datenbank oder einem Data Warehouse zur Verfügung stehen.

Die wichtigste Funktionalität (TM) der Kundenagenten – neben der Verwaltung und Erstellung des kundenindividuellen Wissens – ist die Durchführung virtueller Einkäufe in vorgegebenen Szenarien.

Das Faktenwissen (FK) eines Kundenagenten besteht aus anonymisierten Kundenkarteninformationen - wie beispielsweise soziodemographischen und geographischen Angaben über Alter, Geschlecht, Familienstatus und Wohnort - und der kundenbezogenen Einkaufshistorie in Form von Kassenbons und darauf aufbauenden statistischen Kennzahlen,

wie beispielsweise kundenbezogener Gesamtumsatz, Umsatz-Wochentagsverteilungen und Kundenwertkennzahlen.

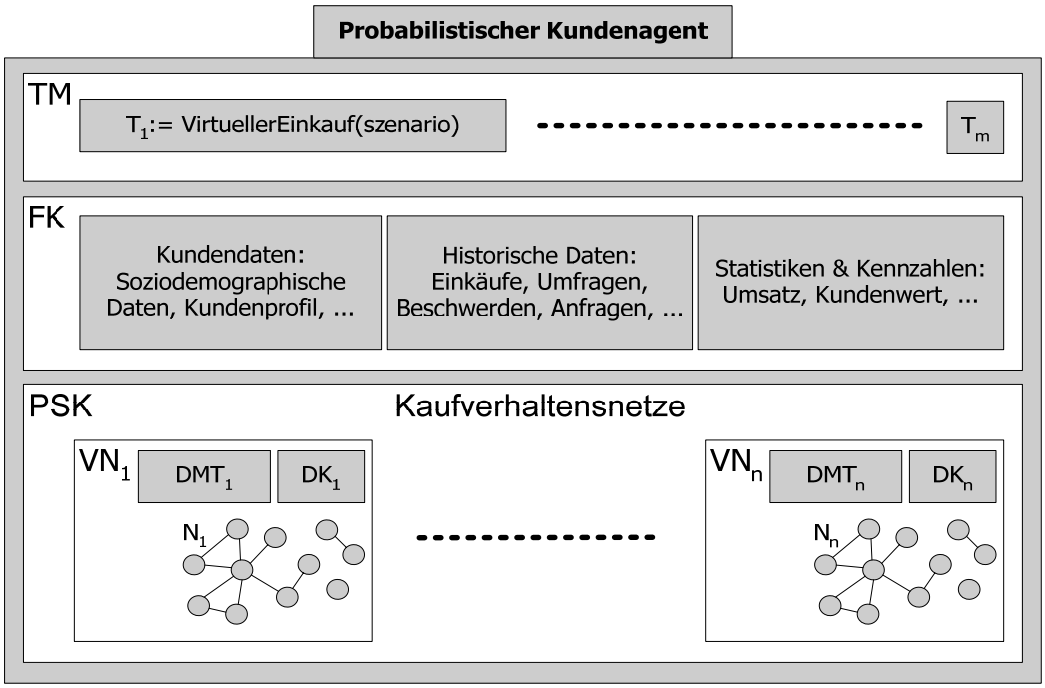


Abbildung 16: Generelle Architektur der probabilistischen Kundenagenten

Das probabilistische Strukturwissen (PSK) wird mit probabilistischen Verhaltensnetze repräsentiert, die das Kaufverhalten des Kunden modellieren. Die Netze werden anhand des Faktenwissens und weiterer realer Daten über Artikeleigenschaften, Konkurrenzmaßnahmen und Umwelteinflüsse individuell gelernt.

Das Kaufverhalten eines Kunden wird durch das Zusammenspiel vieler Einflussfaktoren wie Charaktereigenschaften, Neigungen, Bedürfnissen und Umwelteinflüssen bestimmt. Jede dieser Eigenschaften - wie beispielsweise Sparsamkeit, Qualitätsbewusstsein, Markentreue oder Preissensibilität - trägt ihren Teil zum persönlichen Verhalten bei, wobei die Eigenschaften sich gegenseitig beeinflussen. So ist es beispielsweise für die Modellierung und Simulation des Kaufverhaltens interessant, ob das Qualitätsbewusstsein oder die Preissensibilität eines Kunden bei einem bestimmten Angebot überwiegt, d. h. ob der Kunde sich durch einen sehr niedrigen Preis zum Kauf eines minderwertigen Produktes „überreden“ lässt oder nicht. Bildlich gesprochen streiten sich die Eigenschaften eines Kunden, wer von ihnen den größten Einfluss auf dessen Kaufentscheidung hat. Aus diesem Grund liegt es nahe, die Eigenschaften eines Kunden selbst als probabilistische Agenten zu betrachten, als so genannte *Feature-Agenten*. Aufbauend auf dieser Sichtweise ist ein Kundenagent ein Holon aus einer Menge von Feature-Agenten (so genanntes *Feature-MAS*), wobei jeder

Feature-Agent jeweils eine einzelne Charaktereigenschaft, Gewohnheit, Neigung oder individuelle Reaktion auf Umwelteinflüsse des Kunden repräsentiert (siehe Abbildung 17). Dies kommt der Sichtweise Marvin Minskys nahe, die er in seinem einflussreichen Buch [MM86] als „society of mind“ bezeichnet.

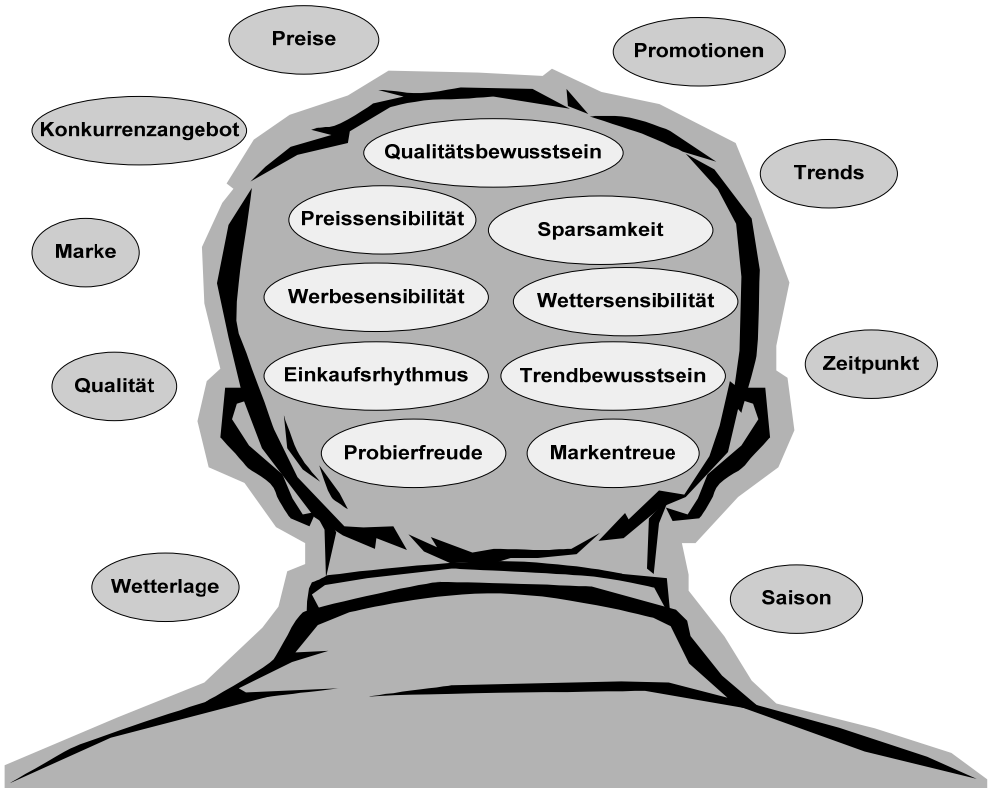


Abbildung 17: Society of Mind

Abbildung 18 zeigt die holonische Kundenagentenstruktur, bei der ein Kundenagent aus einer Menge von Feature-Agenten besteht, die jeweils komplette probabilistische Agenten mit eigenen Funktionalitäten, eigenem Faktenwissen und entsprechenden Verhaltensnetzen sind. Der Kopf des Holons wird durch den eigentlichen Kundenagenten repräsentiert. Einzelne Feature-Agenten können dabei jeweils als Partialmodell oder *Feature-Dimension* des Kunden angesehen werden, d. h. man kann sie als eine gefilterte Sicht auf Funktionalitäten, Wissen und Verhalten des Kundenagenten betrachten.

Das Verhalten bzw. die Reaktionen des Kundenagenten bezogen auf ein vorgegebenes Szenario ergeben sich aus dem Zusammenspiel der einzelnen Feature-Agenten. Die zugrunde liegende Holonenform kann dabei unterschiedlich sein: von der kommunikationsgesteuerten Interaktion zwischen den einzelnen Feature-Agenten bis hin zu deren partiellen bzw. vollständigen Verschmelzung.

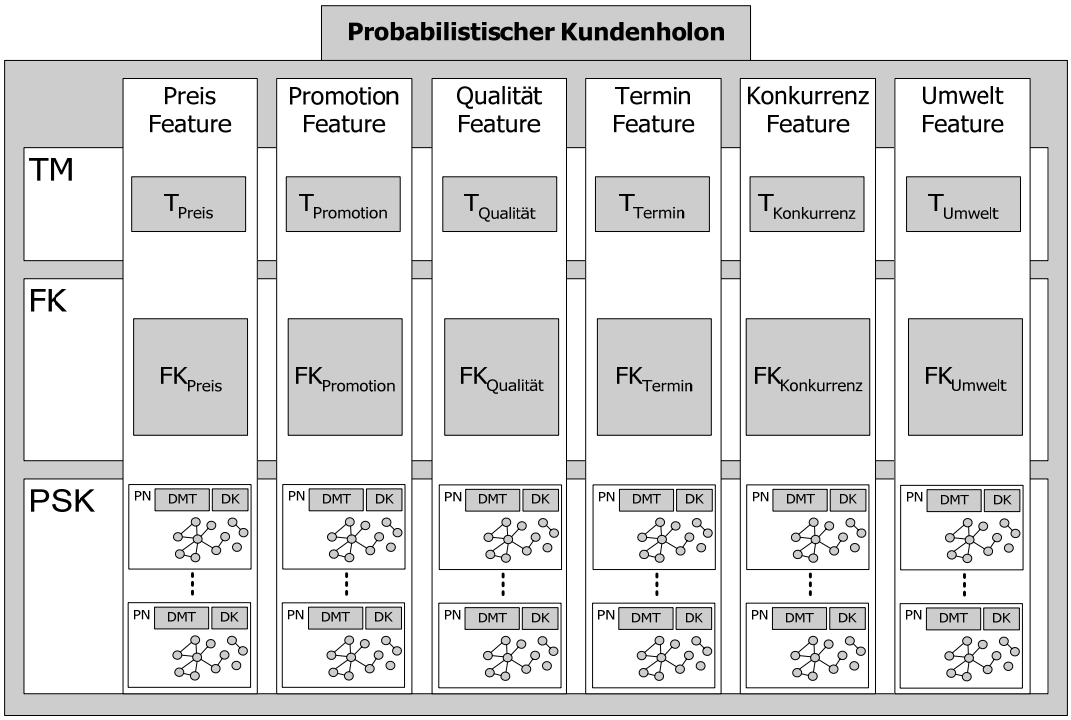


Abbildung 18: Kundenagent als Holon aus Feature-Agenten

3.2.1 Kundenattribute

Das individuelle Kaufverhalten eines Kunden kann anhand persönlicher Ausprägungen bezüglich einer Menge von Kundenattributen beschrieben werden, die im Faktenwissen und der PSK der entsprechenden Feature-Agenten des Kundenagenten gekapselt sind. Ich unterscheide bei den verwendeten Attributen zwischen:

- statischen Attributen,
- einfachen dynamischen Attributen und
- komplexen verhaltensbezogenen Attributen.

3.2.1.1 Statische Attribute

Statische Attribute sind Elemente des Faktenwissens, deren Gültigkeit konstant ist oder deren Veränderungen in der zugrunde liegenden Datenbank nicht gepflegt werden. Dabei handelt es sich meist um soziodemographische und geographische Merkmale. Beispiele für einfache statische Attribute sind Geburtsjahr, Geschlecht, Wohnort, Einkommen, Familienstand, Anzahl der Kinder, etc. Während beispielsweise das Geburtsjahr unveränderlich ist, können sich beispielsweise Wohnort, Einkommen oder Anzahl der Kinder eines Kunden im Laufe der Zeit ändern, was aber in der Praxis in den meisten Fällen nicht informationstechnisch erfasst wird.

3.2.1.2 Einfache dynamische Attribute

Einfache dynamische Attribute sind in der Regel Kennzahlen bzw. statistische Werte, die aus dem Faktenwissen generiert werden und sich im Laufe der Zeit ändern. Es handelt sich meist um einfache verhaltensbezogene Merkmale. Beispiele, die für verschiedene Zeiträume berechnet werden können (z. B. pro Tag, Woche, Monat, Quartal, Jahr, oder insgesamt), sind:

- Anzahl der getätigten Einkäufe,
- Anzahl unterschiedlicher erworbener Artikel,
- Anzahl unterschiedlicher Warengruppen, aus denen Artikel erworben wurden,
- Umsatz bzw. Ertrag,
- Anzahl bzw. Anteil zum Zeitpunkt des Kaufs beworbener Artikel,
- Anzahl bzw. Anteil von Markenartikeln,
- Anzahl bzw. Anteil von preiswerten/mittelpreisigen/teuren Artikeln[1],
- Wochentagsverteilung der getätigten Einkäufe oder
- Anzahl der Reklamationen.

Je nach Aufgabenstellung können dabei Absolutwerte oder Prozentangaben von größerer Bedeutung sein. Möglich sind ebenso Kombinationen der Attribute, wie beispielsweise der Anteil zum Zeitpunkt des Kaufes beworbener Markenartikel, die an einem Samstag gekauft wurden. Einfache dynamische Attribute werden in der Regel mit Hilfe geeigneter Datenbankanfragen ermittelt.

3.2.1.3 Komplexe verhaltensbezogene Attribute

Unter komplexen verhaltensbezogenen Attributen verstehe ich dynamische Attribute, die nur durch aufwändige Berechnungen bzw. durch Simulationen mit Hilfe entsprechender Verhaltensnetze innerhalb des PSK des Kundenagenten bzw. der Feature-Agenten ermittelt werden können. Beispiele für komplexe dynamische Attribute sind Preis- und Werbesensibilität eines Kunden.

Komplexe Attribute modelliere ich mit Hilfe vollständiger bzw. partieller Verhaltensnetze, die Veränderungen einer oder mehrerer Kennzahlen in Abhängigkeit von verschiedenen Einflussfaktoren repräsentieren. Zur Ermittlung der entsprechenden Kennzahlen werden mehrere Simulationsläufe mit jeweils geänderten Werten der Einflussfaktoren durchgeführt,

[1] Artikel können nach verschiedenen Verfahren in Preiskategorien eingeteilt werden. Wichtig ist dabei die Betrachtung des Preises in Relation zu den Preisen ähnlicher Artikel. Eine geeignte Basis ist dabei der *Grundpreis*, also der Preis pro Mengeneinheit bzw. Gewicht, wie beispielsweise Preis pro ml, kg, Stück etc.

wobei das Ergebnis durch Mittelwert- bzw. Summenbildung der jeweiligen Teilergebnisse gebildet wird.

Beispielsweise kann eine Kennzahl bezüglich der Preissensibilität eines Kunden berechnet werden, indem bei einem Verhaltensnetz, das die Abhängigkeit des kundenindividuellen Absatzes vom Verkaufspreis eines Produktes modelliert, der Preis jeweils um einen Prozent- oder Absolutwert (beispielsweise 0,1 Euro) erhöht und anschließend die prozentuale bzw. absolute durchschnittliche Veränderung der Absatzmenge durch Simulation ermittelt wird. So erhält man die durchschnittliche Veränderung der vom Kunden gekauften Artikelmenge in Abhängigkeit von Preisänderungen. Analog können Kennzahlen wie beispielsweise Werbe- oder Qualitätssensibilität berechnet werden. Dabei sind nahezu beliebige Kombinationen von Einflüssen möglich, um entsprechende Kennzahlen zu bilden (siehe Kapitel 4).

3.2.2 Verhaltensnetze

Die in dieser Arbeit verwendeten Verhaltensnetze basieren auf Bayes'schen Netzen (siehe Kapitel 2.3). Sie modellieren das jeweilige individuelle Kaufverhalten bzw. Teilverhalten (Feature-Verhalten) einzelner Kunden und werden zur kundenindividuellen Analyse, Diagnose, Prognose und Simulation verwendet.

Das Lernen der Netze wird mit historischen, kundenbezogenen Daten anonymisierter Kundenkarten, Kreditkarten, Kassenbons und Kundenbefragungen durchgeführt, die in den Data Marts und Data Warehouses des Einzelhandels zur Verfügung stehen. Ein Data Warehouse ist eine spezielle unternehmensweite, einheitliche und konsistente Meta-Datenbank, die parallel zu den operativen Datenbanken eine integrierte Datensicht für die vielfältigsten analytischen Anwendungen bietet [GKS02]. Im Gegensatz dazu stellt ein Data Mart lokalen Nutzern eine ausgewählte Datenbasis für spezielle Anwendungen zur Verfügung (z. B. *Business-Intelligence-Lösungen*). Auf Basis von Data Warehouses und Data Marts werden im Rahmen des *On-Line Analytical Processing* (*OLAP*) sämtliche betriebswirtschaftlich relevanten Daten (z. B. Absatz, Kosten, Deckungsbeiträge etc.) in *multidimensionalen Datenwürfeln* (so genannten *Cubes*) abgebildet, deren Dimensionen betriebswirtschaftlichen Gliederungskriterien entsprechen [HW04] (z. B. Produktgruppen, Kundengruppen, Verkaufsgebiete etc.).

Kundenbezogene Daten transformiere ich in so genannte *Data-Mining-Tabellen*, die als Lerndaten der Verhaltensnetze dienen, wobei mit Hilfe maschineller Lernverfahren des Data Minings Zusammenhänge und Verhaltensmuster aus den Data-Mining-Tabellen extrahiert werden. Der Prozess des Data Minings beinhaltet die Aufbereitung und Bereinigung der Unternehmensdaten. Es werden beispielsweise Inkonsistenzen innerhalb der Daten beseitigt sowie fehlende oder fehlerhafte Einträge entdeckt, die gegebenenfalls korrigiert oder gelöscht werden.

Zur Modellierung der Grundstruktur und deren Evaluierung fließen Erkenntnisse aus dem Marketing und der Psychologie in Form von Domänenwissen ein. Um Umwelteinflüsse in den Netzen zu modellieren, werden weitere Informationen benötigt, wie etwa tagesgenaue Wetterdaten, Informationen über Wirtschaftslage, Feiertage oder andere Events wie beispielsweise Rabatt- oder Jubiläumsaktionen. Abbildung 19 zeigt zusammenfassend alle Daten und Informationen, die ich zur Erstellung der kundenbezogenen, probabilistischen Verhaltensnetze verwende.

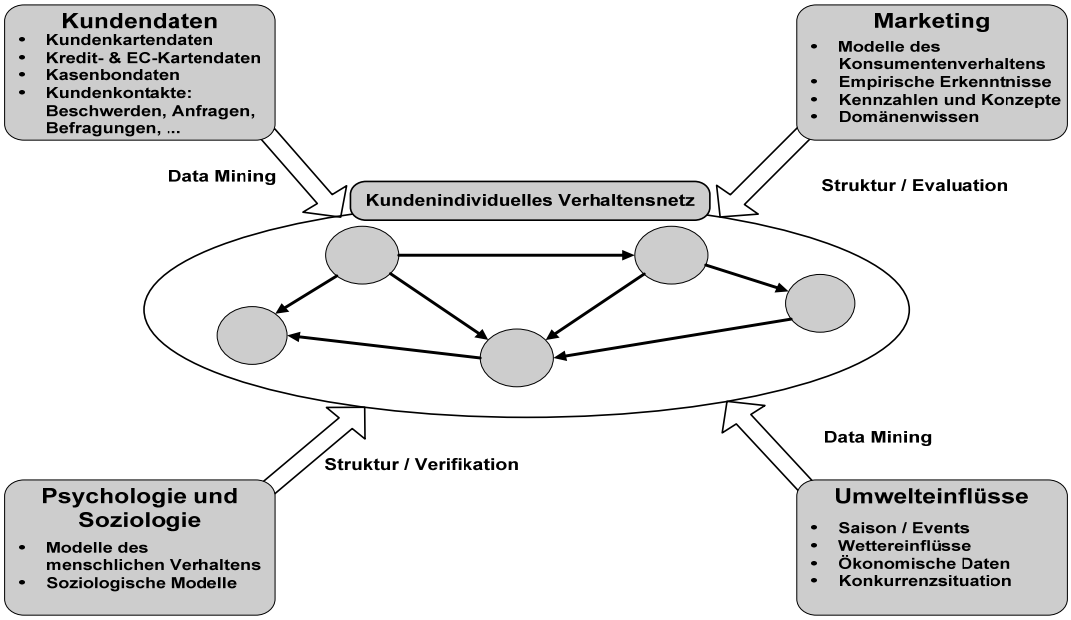


Abbildung 19: Generierung der probabilistischen Verhaltensnetze

3.2.3 Feature-Agenten im Detail

Ich unterscheide Feature-Agenten bezüglich des *semantischen Typs* der entsprechenden Kundenattribute und Verhaltensaspekte. Im Rahmen dieser Arbeit habe ich folgende semantischen Typen von Stimuli identifiziert, die jeweils als einzelne Feature-Agenten zur Verfügung stehen:

- Preis-Feature-Agent,
- Promotion-Feature-Agent,
- Qualität-Feature-Agent,
- Termin-Feature-Agent,
- Konkurrenz-Feature-Agent sowie
- Umwelt-Feature-Agent.

Jeder dieser Feature-Agenten verwaltet Kundenattribute und Verhaltensweisen, die sich auf den entsprechenden semantischen Typ beziehen und ist somit ein partielles Stimulus-Response-Modell des individuellen Kaufverhaltens.

Damit die verschiedenen Netze der Feature-Agenten durch Holonisierung verschmolzen werden können, beziehe ich die entsprechenden Einflussvariablen (Stimuli) jeweils auf dieselben Response-Variablen mit den semantischen Typen „Absatz“, „Umsatz“ und „Ertrag“, wobei der kundenindividuelle „Absatz“ im Vordergrund steht. Um beispielsweise die Preissensibilität eines Kunden bezüglich eines bestimmten Artikels zu modellieren, wird ein Verhaltensnetz erlernt, dessen Struktur die Beziehung zwischen Artikelpreis und der vom Kunden gekauften Artikelmenge bzw. dem kundenbezogenen Umsatz oder Gewinn repräsentiert.

Die Verhaltensnetze der Feature-Agenten können sich sowohl auf das entsprechende Verhalten des Kunden bezüglich eines einzelnen Artikels als auch auf eine Menge von Artikeln einer Warengruppe bzw. einer beliebigen Artikelmenge beziehen.

Um Attributausprägungen und Kennzahlen ermitteln und entsprechende Verhaltensnetze lernen zu können, haben Feature-Agenten Zugriff auf eine Datenbank mit kundenbezogenen historischen Daten und Fakten über historische Zustände der Stimulivariablen (Informationen über Artikelpreise, Werbemaßnahmen, Umweltzustände, etc.).

In der Regel ist die Struktur der Netze sowie Anzahl und Semantik der Variablenzustände von entsprechenden Informationen bzw. Datenspalten in der Datenquelle bzw. der Data-Mining-Tabelle abhängig. Im Folgenden präsentiere ich daher die allgemeine Semantik und Grundstruktur der einzelnen Feature-Agenten, die in der Praxis jedoch je nach Datengrundlage und Fragestellung unterschiedliche Ausprägungen besitzen können.

3.2.3.1 Der Preis-Feature-Agent

Der Preis-Feature-Agent kapselt alle Fakten, Attribute und Verhaltensnetze, die sich direkt oder indirekt auf das Preisverhalten und die Finanzen des Kunden beziehen. Fakten bzw. statische Attribute sind beispielsweise Informationen über das Einkommen des Kunden. Ein Beispiel für ein einfaches dynamisches Attribut ist die Verteilung der vom Kunden in einem bestimmten Zeitraum gekauften Produkte in unterschiedliche Preiskategorien.

Der Preis-Feature-Agent enthält Verhaltensnetze, die das dynamische Preisverhalten des Kunden bezogen auf einzelne Artikel bzw. Warengruppen repräsentieren. Die Grundstruktur dieser Netze modelliert in der Regel die vom Kunden gekaufte Artikelmenge (bzw. den generierten Umsatz und Gewinn) in Abhängigkeit von Verkaufspreis, Preiskategorie und eventuell gewährten Rabatten. Abbildung 20 zeigt ein entsprechendes Preis-Feature-Verhaltensnetz mit den zugehörigen semantischen Knotentypen.

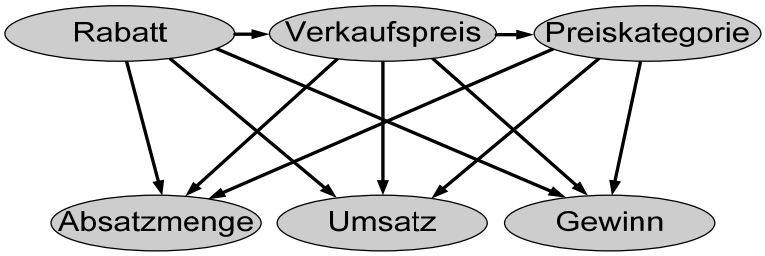


Abbildung 20: Generelle Preis-Feature-Agent-Verhaltensnetz-Struktur

Je nach Anwendungsfall kann sich die in Abbildung 20 gezeigte Struktur ändern, d. h. bestimmte Knoten können wegfallen (z. B. Rabatte, Umsatz oder Gewinn), oder es ergibt sich eine völlig andere Kantenstruktur (beispielsweise kann es Kanten von Absatzmenge zu Umsatz oder Gewinn geben).

Auf Grundlage der Preis-Verhaltensnetze können weitere dynamische Attribute berechnet bzw. simuliert werden, wie beispielsweise die kundenindividuelle Preis- oder Rabattsensibilität bezogen auf eine vorgegebene Artikelmenge.

3.2.3.2 Der Promotion-Feature-Agent

Der Promotion-Feature-Agent verwaltet sämtliche Informationen über das kundenindividuelle Verhalten bezüglich verschiedenartiger Promotionsarten und Werbemaßnahmen, die produktbezogen oder allgemeiner Natur sein können. Darüber hinaus können Abhängigkeiten von Platzierungen und Verpackungsdesigns modelliert werden. Beispiele für einfache promotionsbezogene Attribute sind die Anteile der vom Kunden gekauften beworbenen und nicht beworbenen Artikel oder deren Verteilung in A-, B- und C-Platzierungen.

Analog zum Preis-Feature-Agent wird das promotionsbezogene Verhalten modelliert, indem die kundenindividuelle Absatzmenge (bzw. Umsatz und Gewinn) in Abhängigkeit von promotionsbezogenen Variablen, wie beispielsweise Promotionsart und Platzierung, repräsentiert wird (siehe Abbildung 21).

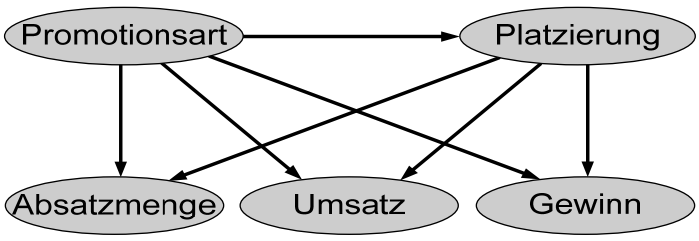


Abbildung 21: Generelle Promotion-Feature-Agent-Verhaltensnetz-Struktur

Mit Hilfe promotionsbezogener Verhaltensnetze lassen sich komplexe dynamische Attribute durch Simulation erzeugen, wie etwa die Werbesensibilität eines Kunden oder ein

Maß, das angibt, wie sehr sich ein Kunde von der Platzierung oder der Verpackung eines Artikels im Regal beeinflussen lässt. Die konkrete Struktur und Semantik promotionsbezogener Netze hängt dabei wiederum von der Datengrundlage ab.

3.2.3.3 Der Qualität-Feature-Agent

Der Qualität-Feature-Agent modelliert kundenindividuelle Präferenzen bezogen auf Marke und Qualität von Produkten oder ähnlichen Artikeleigenschaften, wie beispielsweise ökologische und gesundheitliche Angaben. Entsprechende einfache dynamische Attribute können die Anzahl bzw. der Anteil von Markenartikeln oder Eigenmarkenartikeln unter den vom Kunden erworbenen Produkten sein, oder die Verteilung der gekauften Artikel in qualitativ minder- bzw. hochwertige Produkte. Voraussetzung ist wiederum die Verfügbarkeit entsprechender eindeutiger Qualitätsmerkmale sowie entsprechender Informationen und Daten.

Ein Verhaltensnetz des Qualität-Feature-Agenten kann beispielsweise die Abhängigkeit der vom Kunden erworbenen Absatzmenge von der Marke, Eigenmarke und Produktqualität modellieren (siehe Abbildung 22). Mit Hilfe solcher Verhaltensnetze können weitere dynamische Attribute durch Simulation ermittelt werden, wie zum Beispiel die absolute bzw. prozentuale Veränderung der vom Kunden erworbenen Absatzmenge bezüglich der Änderung der Qualität von Produkten oder Produktgruppen (Qualitätssensibilität).

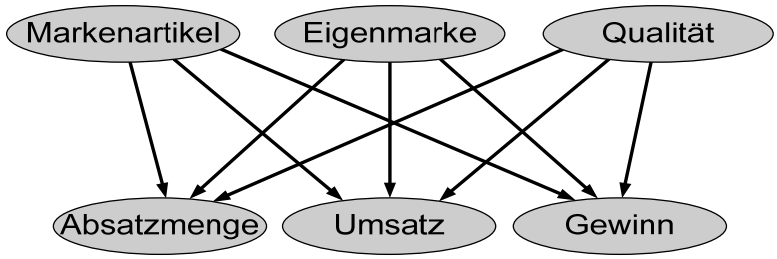


Abbildung 22: Generelle Qualität-Feature-Agent-Verhaltensnetz-Struktur

3.2.3.4 Der Termin-Feature-Agent

Der Termin-Feature-Agent verwaltet Fakten, Information, Daten und Modelle, die sich auf das Verhalten des Kunden bezüglich zeitlicher, zeitpunktbezogener oder eventabhängiger Aspekte beziehen. Events sind in diesem Zusammenhang besondere Zeiträume wie Schulferien, Sommerschlussverkauf oder Weihnachtszeit, nationale und regionale Feiertage sowie spezielle Ereignisse wie Karneval, Valentinstag oder Halloween. Ebenso können Events auch spezielle örtliche Sonderaktionen sein (z. B. Stadt- oder Dorffeste mit verkaufsoffenen Sonntagen) oder von den Einzelhändlern veranstaltete Aktionen wie Familientage, Weinproben oder Jubiläen.

Einfache dynamische zeitliche Attribute sind beispielsweise die Wochentags- oder Monatsverteilung des kundenindividuellen Umsatzes, die Anzahl der Einkäufe in einem gewissen Zeitraum oder die durchschnittliche Dauer zwischen zwei Einkäufen etc.

Der Termin-Feature-Agent verwaltet darüber hinaus Verhaltensnetze, die die Abhängigkeit der kundenindividuellen Absatzmenge (sowie Umsatz bzw. Gewinn) von zeitlichen und saisonalen Faktoren modelliert (siehe Abbildung 23).

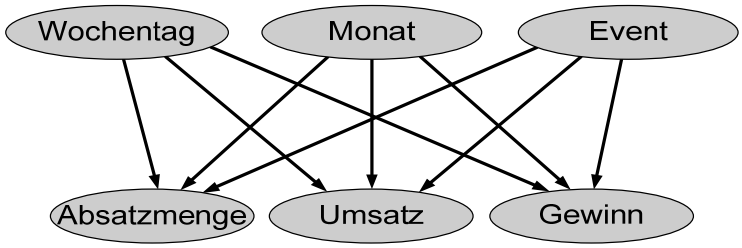


Abbildung 23: Generelle Termin-Feature-Agent-Verhaltensnetz-Struktur

3.2.3.5 Der Konkurrenz-Feature-Agent

Der Konkurrenz-Feature-Agent verwaltet Fakten, Informationen und Verhaltensmodelle, die das Kaufverhalten des Kunden eines Unternehmens in Abhängigkeit der Situation bzw. der Maßnahmen der Unternehmenskonkurrenz modellieren, wie beispielsweise deren Sortimente, Verkaufspreise, Promotionen und Events. Voraussetzung dafür ist allerdings die detaillierte Kenntnis der entsprechenden historischen Konkurrenzdaten. Als einfache dynamische Attribute dienen z. B. die Anzahl bzw. der durchschnittliche Umsatz eines Kunden bei der Konkurrenz oder die Anzahl bzw. die prozentuale Verteilung der beim Unternehmen gekauften Artikel, deren Verkaufspreise bei der Konkurrenz niedriger bzw. höher waren.

Die Verhaltensnetze des Konkurrenz-Feature-Agenten modellieren die Abhängigkeit der kundenbezogenen Reaktionen (unternehmensbezogener kundenindividueller Absatz, Umsatz und Ertrag) von den Verkaufspreisen, Promotionen und Events der Konkurrenz (siehe Abbildung 24).

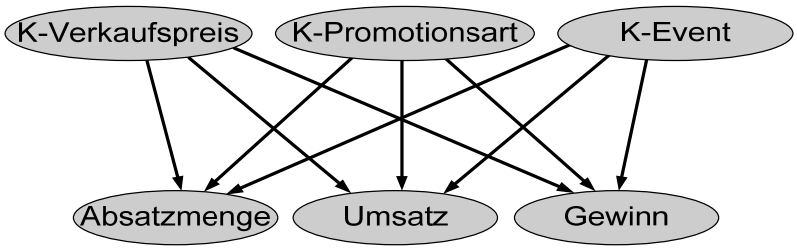


Abbildung 24: Generelle Konkurrenz-Feature-Agent-Verhaltensnetz-Struktur

3.2.3.6 Der Umwelt-Feature-Agent

Der Umwelt-Feature-Agent kapselt Fakten, Informationen und Verhaltensmodelle des Kunden bezüglich der Umwelt, die sich beispielsweise auf Wettereinflüsse oder die allgemeine Wirtschaftslage beziehen. Einfache Attribute sind dabei zum Beispiel die Anzahl bzw. Verteilung der Einkäufe bei schönem oder schlechtem Wetter, wobei meteorologische Einflussfaktoren weiter in Unterfaktoren zerlegt werden können, beispielsweise in Temperatur, Luftfeuchte, Sonnenscheindauer, Niederschlagsart und Schneehöhe. Entsprechende Informationen können in Form von Datenbanktabellen bzw. Data-Mining-Tabellen vom Deutschen Wetterdienst zur Verfügung gestellt werden. Die allgemeine Wirtschaftslage kann anhand verschiedener Kennzahlen, zum Beispiel dem Stand des Deutschen Aktien Index (DAX), Inflationsrate oder dem durchschnittlichen Pro-Kopf-Einkommen, repräsentiert werden.

Die Verhaltensnetze des Umwelt-Feature setzen die kundenbezogene Absatzmenge (bzw. Umsatz und Gewinn) in Relation zu Wetterlagen und Wirtschaftssituationen (siehe Abbildung 25). Mit Hilfe der Verhaltensnetze können wieder komplexere dynamische Attribute durch Simulation ermittelt werden, wie beispielsweise die prozentuale Steigerung bzw. Verringerung des kundenbezogenen Absatzes von Speiseeis oder Mineralwasser in Abhängigkeit von der Lufttemperatur.

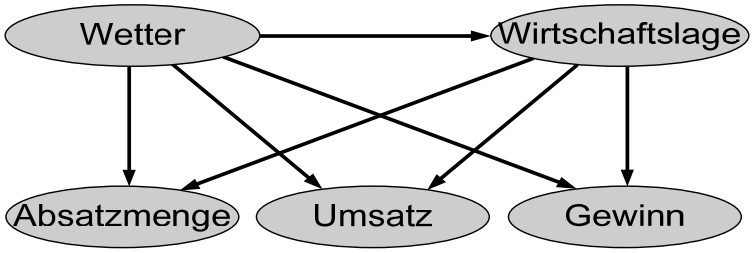


Abbildung 25: Generelle Umwelt-Feature-Agent-Verhaltensnetz-Struktur

3.2.4 Kundenindividuelle Artikel-Feature-Verhaltensmatrix

Generell können Kunden gegenüber verschiedenen Artikeln bzw. Warengruppen ein völlig unterschiedliches Kaufverhalten besitzen. Aus diesem Grund müssen Kundenagenten das individuelle Kaufverhalten eines Kunden bezüglich jedes beliebigen Artikels eines Marktes repräsentieren können. Deshalb modelliere ich das kundenindividuelle Verhalten gegenüber einzelnen Artikeln als Zusammenspiel bzw. Interaktion artikelbezogener Instanzen der verschiedenen kundenindividuellen Feature-Agenten. So kann beispielsweise der Preis-Feature-Agent ein konkretes Produkt als günstig bewerten und somit zum Kauf animieren, während der Qualität-Feature-Agent, aufgrund der schlechten Produktqualität oder der Tatsache, dass es sich nicht um einen Markenartikel handelt, den Kauf ablehnt. Um nun zu

entscheiden, welcher Feature-Agent im konkreten Fall mehr Einfluss besitzt, müssen die Feature-Agenten miteinander „verhandeln“, um die artikelbezogene Reaktion des Kunden zu ermitteln.

Die Interaktion bzw. die Verhandlung der Feature-Agenten realisiere ich durch probabilistische Holonisierung (siehe Kapitel 2.4.2), wobei die partielle bzw. vollständige Verschmelzung aus Gründen der Performance bevorzugt wird. Dazu werden die Feature-Verhaltensnetze der einzelnen Feature-Agenten artikelbezogen gelernt und anschließend temporär zu einem holonischen Kaufverhaltensnetz verschmolzen. Welche Features, semantische Knotentypen und deren konkreten Konfigurationen dabei zum Einsatz kommen, hängt stark von der jeweiligen Fragestellung ab und muss in der Praxis manuell vorgegeben werden. Das resultierende Netz modelliert anschließend die als signifikant bewerteten Zusammenhänge und Abhängigkeiten zwischen den Knoten der jeweiligen relevanten Feature-Verhaltensnetze (siehe Abbildung 26).

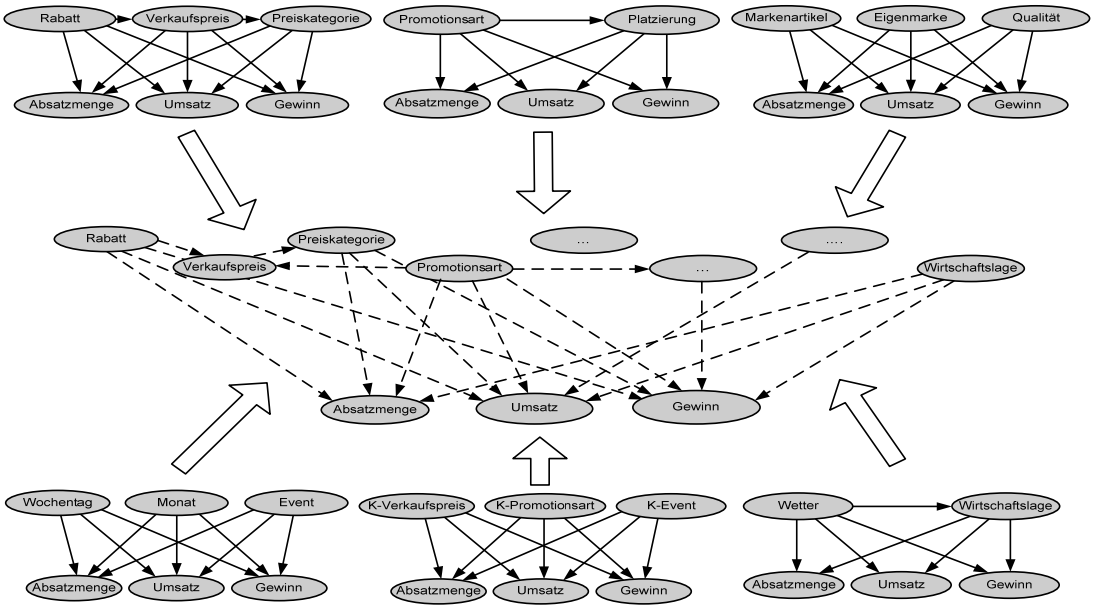


Abbildung 26: Holonisches Kaufverhaltensnetz durch partielle Verschmelzung

Die Relevanz einzelner Knoten und deren Abhängigkeiten wird dabei durch entsprechende Lernverfahren ermittelt, wobei je nach Anwendungsfall und Fragestellung entsprechendes Domänenwissen genutzt wird, um Performanz und Güte zu steigern (siehe 3.3.7).

Das Erstellen holonischer Verhaltensnetze aus einzelnen relevanten Feature-Netzen ist für jeden einzelnen Artikel eines Marktes möglich. Ein Kunde wird daher durch eine so genannte kundenindividuelle Artikel-Feature-Verhaltensmatrix modelliert (siehe Abbildung 27), die sein Kaufverhalten bzw. featurebezogenes Teilverhalten bezüglich jedes einzelnen Artikels

eines Einzelhändlers ausdrückt. So ergeben sich, je nach Datengrundlage und Kundenhistorie, folgende drei Fälle der Verhaltensmodellierung bezüglich

1. Artikeln, die der Kunde regelmäßig oder häufig gekauft hat,
2. Artikeln, die der Kunde selten gekauft hat, und
3. Artikeln, die der Kunde nie gekauft hat.

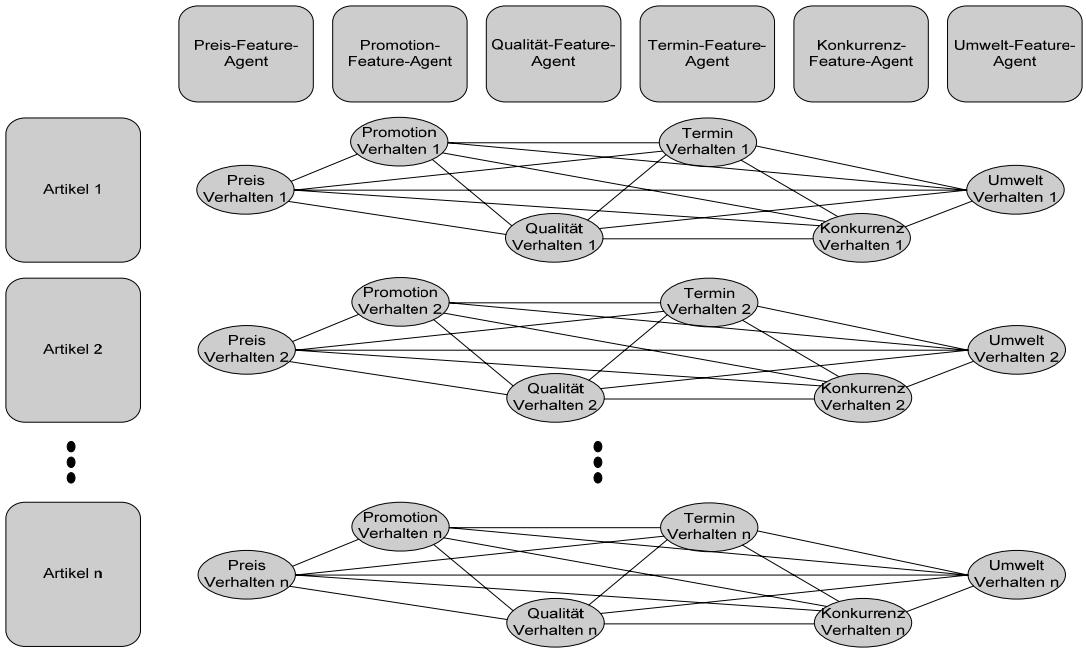


Abbildung 27: Kundenindividuelle Artikel-Feature-Verhaltensmatrix

3.2.4.1 Verhaltensnetze für regelmäßig bzw. häufig gekaufte Artikel

Im Fall, dass ein Kunde einen bestimmten Artikel regelmäßig oder häufig kauft, liegen detaillierte Informationen vor, die aus historischen Kundenkartendaten und der entsprechenden Einkaufshistorie ermittelt werden können. Aus diesen Daten lassen sich nach Bereinigung und Aufbereitung Data-Mining-Tabellen erstellen, die als Grundlage für das Erstellen entsprechender Feature-Verhaltensnetze bzw. dem holonischen Kaufverhaltensnetz dienen (siehe Kapitel 3.3).

3.2.4.2 Verhaltensnetze für selten gekaufte Artikel

Bei Artikeln, die ein Kunde nur einmal oder sehr selten gekauft hat, ist es schwierig, eine Aussage zu treffen. Allerdings kann man sehr leicht feststellen, ob es sich bei den entsprechenden Artikeln um Produkte handelt, die im Allgemeinen nur sehr selten gekauft werden, wie zum Beispiel Fernseher, Waschmaschinen oder Möbel. Falls es sich um neu eingelistete Artikel handelt, die der Kunden aus diesem Grunde nicht häufig kaufen konnte,

kann man durch artikelbezogene Ähnlichkeitsanalysen ähnliche Artikel (meist aus der gleichen Warengruppe) finden, die der Kunde schon öfters erworben hat und dann deren kundenindividuellen Verhaltensnetze übernehmen bzw. das durchschnittliche Verhalten des Kunden innerhalb derselben Produktkategorie verwenden (siehe Kapitel 3.2.5.2). Alternativ können bei geringer Datengrundlage auch Netze von ähnlichen Kunden (siehe Kapitel 5.3 für Details über kundenbezogene vektor- und verhaltensnetzbasierte Ähnlichkeitsanalysen) oder das durchschnittliche Verhalten von geeigneten Kundengruppen verwendet werden (siehe Kapitel 5.2).

3.2.4.3 Verhaltensnetze für nie gekaufte Artikel

Über Artikel, die ein Kunde nie gekauft hat, kann man zunächst keine Aussagen machen, da in diesem Fall keine historischen Daten zur Verfügung stehen. Falls der Kunde den Artikel nie erwerben konnte, weil es sich um eine Neueinlistung handelt, kann jedoch analog zu Verhaltensnetzen für selten gekaufte Artikel als Approximation ein Verhaltensnetz eines ähnlichen Artikels oder ein Durchschnittsverhaltensnetz einer Menge ähnlicher Artikel verwendet werden. Auf diese Weise ist es möglich, das Verhalten bzw. das potentielle Interesse des Kunden an dem entsprechenden Produkt zu schätzen. Falls ein Kunde einen langzeitig eingelisteten Artikel nie erworben hat, besitzt der Kunde entweder aus unbekannten Gründen kein Interesse an ihm oder er wurde nie auf ihn aufmerksam.

3.2.5 Kundenindividuelle holonische Artikelgruppenverhaltensnetze

Im Folgenden beschreibe ich die Modellierung des kundenindividuellen Kaufverhaltens bezüglich einer Menge von Artikeln, wobei ich je nach Anwendungsfall und Fragestellung folgende drei unterschiedliche Modellierungsformen verwende:

- kundenindividuelles Artikelgruppen-Relationsverhaltensnetz,
- kundenindividuelles Artikelgruppen-Durchschnittsverhaltensnetz, und
- kundenindividuelles Artikelgruppen-Additionsverhaltensnetz.

3.2.5.1 Kundenindividuelles Artikelgruppen-Relationsverhaltensnetz

Die kundenindividuellen Artikelverhaltensnetze lassen sich bereits für simulationsbasierte Analysen, Diagnosen und Prognosen unter Berücksichtigung einer Vielzahl von Einflussfaktoren nutzen (siehe Kapitel 4). Allerdings werden dabei keinerlei Abhängigkeiten zwischen Artikeln untereinander betrachtet, d. h. es wird vernachlässigt, dass beispielsweise Verkaufspreis, Promotion, Platzierung oder die Qualität eines Artikels immer auch direkten bzw. indirekten Einfluss auf den Absatz ähnlicher, konkurrierender bzw. ergänzender Produkte haben können. Beispielsweise kann die Promotion eines Produktes *P* der Ketchup-Marke *M* den Kunden nicht nur zum Kauf von *P*, sondern auch dem anderer Produkte von *M*

animieren, da er durch die Werbung möglicherweise auf *M* aufmerksam wird. Das könnte andererseits auch dazu führen, dass der Kunde keine Ketchup-Produkte anderer Marken erwirbt, vor allem wenn er bei Produkt *P* neben der Promotion auch durch einen verringerten Promotions-Verkaufspreis „angelockt" wird. Das bedeutet, dass die Promotion bzw. Preissenkung bei Produkt *P* zu erhöhtem Absatz bei *P* und anderen Produkten der Marke *M*, aber gleichzeitig auch zur Verringerung des Absatzes konkurrierender Produkte führen kann. Zusätzlich könnte durch die Fokussierung auf Ketchup beispielsweise der Absatz von tiefgekühlten Pommes Frites leicht erhöht werden, da Ketchup und Pommes Frites oft zusammen gekauft werden (so genannter *Cross-Selling-Effekt*).

Um mögliche kausale Zusammenhänge bzw. Abhängigkeiten zwischen einer Menge von Artikeln kundenindividuell zu modellieren, müssen entsprechende Abhängigkeiten zwischen den jeweiligen Artikelverhaltensnetzen erkannt und repräsentiert werden. Dies realisiere ich durch Verschmelzung der kundenindividuellen Artikelverhaltensnetze mit Hilfe der Relationsverschmelzung, die emergentes Wissen (hier Artikelabhängigkeiten) entdecken und modellieren kann (siehe Kapitel 2.4.2.4). Abbildung 28 zeigt ein einfaches Beispiel einer solchen Relationsverschmelzung zweier Artikel *A* und *B*.

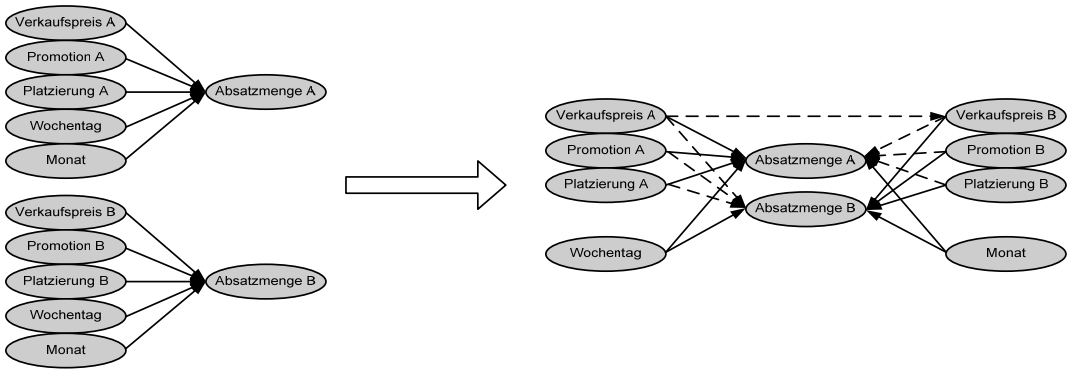


Abbildung 28: Kundenindividuelles Artikelgruppen-Relationsverhaltensnetz

Auf der linken Seite sind die beiden individuellen Artikelverhaltensnetze des Kunden bezüglich der Artikel *A* und *B*, wobei aus Gründen der Vereinfachung nicht alle Feature-Agenten bzw. Knoten der Feature-Verhaltensnetze berücksichtigt werden. Auf der rechten Seite erscheint das Ergebnis der Relationsverschmelzung der beiden Einzelnetze. Die Knoten für Wochentag und Monat wurden vereinigt, da sie in den beiden Ursprungsnetzen exakt die gleichen Werte beinhalten. Die gestrichelten Kanten geben mögliche Abhängigkeiten an, die durch Korrelationsanalysen bzw. durch die Anwendung des Verschmelzungsalgorithmus entdeckt werden können und als emergentes Wissen gespeichert werden.

Da die Komplexität der Artikelgruppennetze mit der Anzahl der Artikel exponentiell zunehmen kann, werden nur solche Kanten hinzugefügt, bei denen der Mehrwert (also der

Zugewinn an Modellierungsgüte) größer ist als die Erhöhung der Komplexität (kann mit Hilfe von Metriken wie beispielsweise dem MDL-Gewicht gewährleistet werden; siehe Kapitel 3.3). Durch den Einsatz von Domänenwissen in Form von erzwungenen, vorgeschlagenen und verbotenen Kantenkombinationen kann die Komplexität des Verschmelzens in der Praxis enorm reduziert werden (siehe Kapitel 3.3.7). Darüber hinaus bietet es sich an, nur solche Artikelverhaltensnetze einer vorgegebenen Artikelmenge zu einem Artikelgruppenverhaltensnetz zu verschmelzen, bei denen eine deutliche Abhängigkeit bzw. Relation durch im Vorfeld durchgeführte Korrelationsanalysen ermittelt werden konnte.

3.2.5.2 Kundenindividuelles Artikelgruppen-Durchschnittsverhaltensnetz

Kundenindividuelle Artikelgruppen-Durchschnittsverhaltensnetze dienen zur Modellierung des durchschnittlichen Verhaltens eines Kunden bezüglich einer vorgegebenen Artikelmenge, beispielsweise einer thematischen Warengruppe. Ich realisiere die Bildung entsprechender Netze mit Hilfe der in Kapitel 2.4.2.4 vorgestellten Durchschnittsverschmelzung für probabilistische Verhaltensnetze, indem die relevanten Artikelverhaltensnetze quasi „gemittelt“ werden (siehe Abbildung 29). Dabei ist es in der Praxis oft sinnvoll, beispielsweise den Knoten „Verkaufspreis“ durch die entsprechende „Preiskategorie“ zu ersetzen, wenn die Artikel der vorgegebenen Menge sehr verschiedenartig sind.

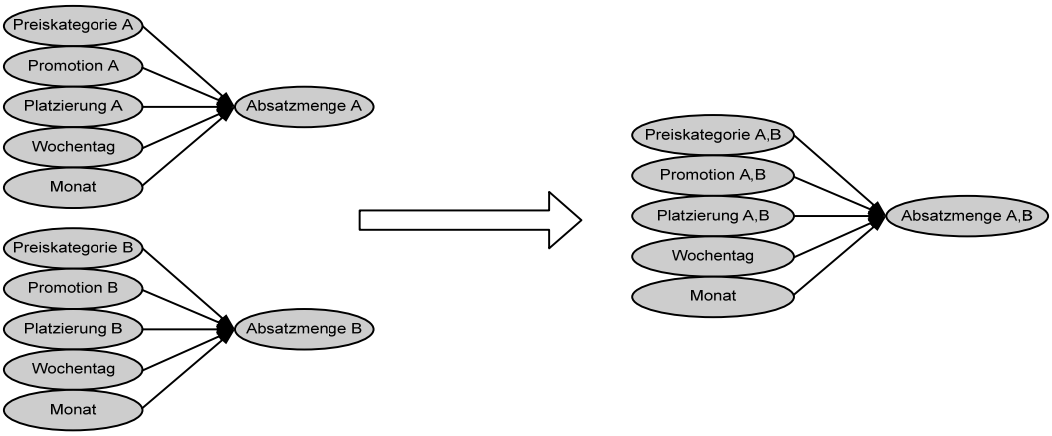


Abbildung 29: Kundenindividuelles Artikelgruppen-Durchschnittsverhaltensnetz

Artikelmengen-Durchschnittsverhaltensnetze können beispielsweise eingesetzt werden, wenn für einen speziellen Artikel nur wenige kundenbezogene Daten vorliegen und es dadurch nicht möglich ist, ein detailliertes Einzelartikelverhaltensnetz zu erzeugen oder wenn ein neuer Artikel ins Sortiment aufgenommen werden soll und daher noch keine historischen Daten verfügbar sind. In diesen Fällen kann das Durchschnittsverhaltensnetz als Schätzung verwendet werden. Die zugrunde liegende Artikelmenge wird dabei ausschließlich aus Produkten gebildet, die dem zu untersuchenden Produkt ähnlich sind, was im Vorfeld durch

Ähnlichkeitsanalysen bzw. durch Klassifikation ermöglicht werden kann.

Darüber hinaus können Durchschnittsnetze auch zur Erzeugung statischer oder dynamischer Attribute bezüglich beliebiger Artikelgruppen bzw. Warengruppen und Warenkörben verwendet werden. Einen guten Überblick über das individuelle Kaufverhalten einzelner Kunden bietet dabei beispielsweise das *globale Durchschnittskaufverhaltensnetz* eines Kunden, das ein spezielles Durchschnittsverhaltensnetz ist, wobei die Artikelmenge aus allen Artikeln besteht, die der Kunde jemals erworben hat.

Analog können Durchschnittsnetze für thematische Warengruppen wie Food, Nonfood, Haushaltswaren oder elektrische Geräte erzeugt werden, die einen guten Überblick über das jeweilige warengruppenspezifische Verhalten bieten und als Basis für kundenbezogene Ähnlichkeitsanalysen, Klassifikationen und Clusterings dienen (siehe Kapitel 5.2).

3.2.5.3 Kundenindividuelles Artikelgruppen-Additionsverhaltensnetz

Neben den Durchschnittsverhaltensnetzen benötige ich in einigen Fällen Artikelgruppen-Additionsverhaltensnetze, die das Verhalten bezüglich aller Einzelartikel einer vorgegebenen Artikelgruppe quasi „summieren“. Konkret werden dabei ausschließlich die Werte der Knoten *Absatzmenge*, *Umsatz* und *Gewinn* durch entsprechende Additionsverschmelzung summiert (siehe Kapitel 2.4.2.4). Abbildung 30 zeigt ein vereinfachtes Beispiel der Additionsverschmelzung der Artikel *A* und *B*.

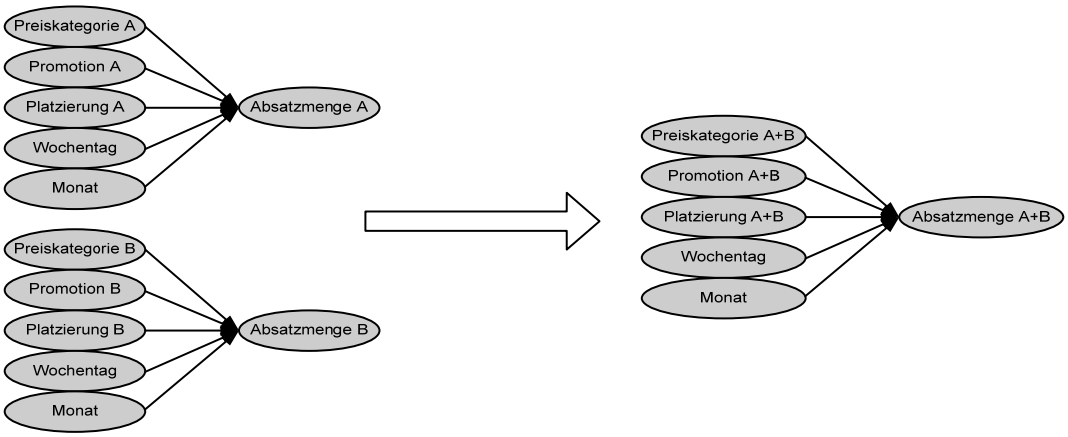


Abbildung 30: Kundenindividuelles Artikelgruppen-Additionsverhaltensnetz

Kundenindividuelle Artikelgruppen-Additionsverhaltensnetze können verwendet werden, um aggregierte Simulationen bezüglich der vorgebenden Artikelgruppe durchführen zu können, ohne dabei auf die einzelnen Artikel genauer eingehen zu müssen. Darüber hinaus sind sie neben den Durchschnittsnetzen eine gute Grundlage zur kundenbezogenen Ähnlichkeitsanalyse, Kundenklassifikation und zum Kundenclustering.

Analog zum globalen Durchschnittskaufverhaltensnetz kann je nach Fragestellung ein

globales Additionskaufverhaltensnetz erlernt werden, indem die entsprechenden DM-Tabellen aller jemals vom Kunden erworbenen Artikel „summiert“ werden, um Erkenntnisse über das Gesamtverhalten eines Kunden zu gewinnen (beispielsweise in welcher Weise der Gesamtumsatz eines Kunden von Werbemaßnahmen und Preiskategorien abhängt).

3.2.6 Modellierung von Zeitpunkten, Verhaltensänderungen und Trends

Die bisher beschriebenen Verhaltensnetze modellieren das kundenindividuelle Kaufverhalten in Abhängigkeit von einer Vielzahl von Einflussfaktoren. Dabei spielt die Berücksichtigung von zeitpunktbezogenen und sequenziellen Einflüssen oft eine wichtige Rolle, wie beispielsweise die Modellierung von *Zeitpunkten vor*, *während* und *nach* gewissen Ereignissen oder Zuständen, wobei man die zeitlichen Abstände in vielen Fällen quantifizieren und in geeigneter Form diskretisieren möchte. Beispielsweise, wenn das Verhalten eines Kunden vor oder nach Feiertagen, oder nach Ablauf oder vor Beginn einer angekündigten Promotion modelliert werden soll.

Zur Modellierung dieser zeitpunktbezogenen Abhängigkeiten kann die Verwendung von dynamischen Bayes’schen Netzen (siehe Kapitel 2.3.4.1) oder Hidden-Markov-Modellen (siehe Kapitel 2.3.5.1) in Betracht gezogen werden. Allerdings erreichen dynamische Bayes’sche Netze und vor allem Hidden-Markov-Modelle in der Praxis sehr schnell eine nicht mehr handhabbare Komplexität. Zur Einschränkung der Komplexität verwende ich daher für die Praxis eine *Drei-Zeitscheiben-Modellierung*, bei der *Vergangenheit*, *Gegenwart* und *Zukunft* jeweils durch eine Scheibe modelliert werden. Die zeitlichen Abstände werden dabei durch entsprechende Aggregationen der Lerndaten vorgegeben.

Alternativ bietet sich eine Vorgehensweise an, die man als *Delta-Modellierung* bezeichnen kann. Dabei werden Zufallsvariablen entweder um *Deltazustände* erweitert, die zeitliche Abstände modellieren oder man erweitert das Netz um ein oder mehrere *Deltaknoten*, deren Zustände Zeitpunktbeschreibungen repräsentieren und als Elternknoten für Variablen verwendet werden, deren Zustände zeitpunktabhängig modelliert werden sollen. Beispielsweise können bei einem Promotionsknoten die Zustände „Ja“ und „Nein“ durch Deltazustände „Keine Promotion“, „Vor Promotion“, „Während Promotion“ und „Nach Promotion“ ersetzt werden. Einen Deltaknoten mit den Zuständen „Kurz vor…“, „“Kurz nach…“ und „Weder noch“ eignet sich als Elternknoten für einen Feiertagsknoten, dessen Zustände eine Reihe von unterschiedlichen Feiertagen repräsentieren.

Die Verwendung von Deltazuständen hat bei der zeitpunktbezogenen Modellierung mehrerer Zustände den Nachteil, dass in diesem Fall die Anzahl der benötigten Deltazustände exponentiell zunehmen kann, wie beispielsweise bei der Modellierung von mehreren Promotionsarten und entsprechenden Zeitpunkten. Deltaknoten sind in diesen Fällen in der Praxis besser geeignet, obwohl sie ebenfalls die Komplexität des Verhaltensnetzes stark

erhöhen können, da die entsprechenden CPT exponentiell mit der Anzahl der Elternknoten bzw. deren Zuständen wachsen.

Das Verhalten eines Kunden kann sich im Laufe der Zeit ändern, beispielsweise aufgrund eines veränderten Einkommens oder der Änderung des Familienstandes. Das bedeutet, dass sich das heutige Verhalten eines Kunden deutlich von seinem vergangenen Verhalten unterscheiden kann. Wird ein Verhaltensmodell aus historischen Daten erlernt, so werden dabei alle Lerndaten gleich stark berücksichtigt, wodurch das Modell letztendlich das durchschnittliche Verhalten des Kunden über die Zeit repräsentiert, das sich folglich vom jetzigen Verhalten unterscheiden kann. Möchte man allerdings explizit das heutige Verhalten modellieren, müssen beim Lernen der entsprechenden Netze jüngere Daten stärker gewichtet werden als ältere. Dies kann beispielsweise durch so genannte *gewichtete Adaption* der probabilistischen Verhaltensnetze erreicht werden, bei der existierende Netze an neue Lerndaten angepasst bzw. jüngere Daten beim Erlernen von Verhaltensnetzen durch geeignete Faktoren bzw. Gewichtungsfunktionen stärker gewichtet werden. Kapitel 3.3.6 gibt einen Überblick über unterschiedliche Adaptionsmodelle und -verfahren.

Weist das Verhalten eines Kunden einen zeitlichen Trend auf, dann sollte dieser zur Modellierung des zukünftigen kundenindividuellen Verhaltens berücksichtigt werden. Ein Trend stellt im Allgemeinen eine erkennbare kontinuierliche Verhaltensveränderung bzw. eine statistisch messbare Tendenz dar, von der ausgegangen wird, dass sie in der Zukunft anhält. Meiner Meinung nach spielen Trends bei einzelnen Kunden keine sehr große Rolle, wenn man das Verhalten der Kunden mit Hilfe von Adaptionsverfahren – also stärkerer Gewichtung jüngerer Lernfälle – aus historischen Daten erlernt und damit das Verhalten des Kunden der *nahen Zukunft* prognostizieren möchte. Falls bei einem Kunden in der Praxis starke Trends vorliegen, können diese mit Hilfe geeigneter Verfahren ermittelt – beispielsweise durch Trendprognose oder Exponentielle Glättung (siehe Anhang A.7) – und durch Extrapolation als so genannte *Liftfaktoren* in die Ergebnisse der Modellierung zukünftigen Verhaltens eingerechnet werden.

3.3 Lernen individueller Kaufverhaltensnetze mit realen Daten

Die Generierung der probabilistischen Verhaltensnetze läuft in mehreren Prozessschritten ab (*Knowledge Engineering*, siehe Abbildung 31). Im ersten Schritt des Knowledge Engineerings werden aus *bereinigten historischen Rohdaten*, die in einer Datenbank bzw. einem Data Warehouse zur Verfügung stehen, so genannte *Data-Mining-Tabellen* (*DM-Tabelle*) erstellt, deren Zeilen einzelnen Trainingsfällen entsprechen und jede Spalte in der Regel einer Zufallsvariablen zugeordnet ist. In Kapitel 3.3.1 erläutere ich den Aufbau und die Generierung der DM-Tabellen und gebe einen kurzen Überblick über die Bereinigung von

Rohdaten. Im nächsten Schritt werden einzelne Tabellenspalten *diskretisiert*, falls die Anzahl ihrer distinkten Werte eine vorgegebene Maximalanzahl für Knotenzustände überschreitet oder die Werte vorgegebenen Partitionierungen angepasst werden sollen (siehe Kapitel 3.3.2). Je nach Anwendungsfall bieten sich dabei unterschiedliche *Diskretisierungsalgorithmen* an, um die Anzahl der zulässigen Knotenzustände zu limitieren, wobei alternativ für einzelne semantische Knotentypen vordefinierte Diskretisierungen verwendet werden können.

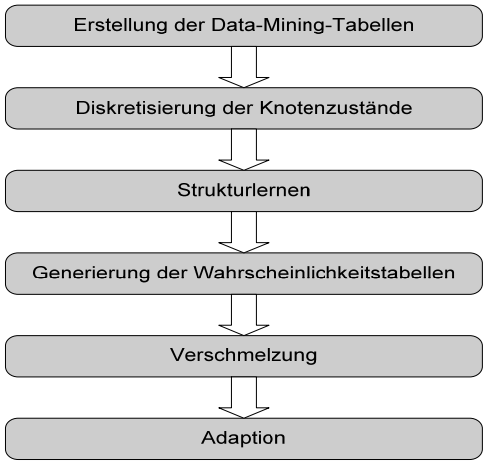


Abbildung 31: Erstellung probabilistischer Verhaltensnetze

Anschließend werden sowohl die Struktur des Netzes als auch Wahrscheinlichkeitstabellen der Knoten, basierend auf den diskretisierten Trainingsfällen $D = \{d_1,\ldots,d_m\}$ der Data-Mining-Tabelle, erlernt. In der Regel besteht der Lernprozess aus zwei Schritten:

- Lernen der Netzstruktur (d. h. Kanten und deren Richtung) und
- Lernen der Wahrscheinlichkeitstabellen (CPT und Apriori-Tabellen),

wobei Algorithmen existieren, die beide Teilprozesse kombinieren. Kapitel 3.3.3 erläutert das Lernen der Wahrscheinlichkeitstabellen basierend auf den Lerndaten in den DM-Tabellen unter der Vorraussetzung, dass die Netzstruktur bereits erlernt wurde. Zum Erlernen der Netzstruktur gibt es dabei die folgenden beiden Verfahrensansätze:

- *testbasierte* Lernverfahren und
- *metrikbasierte* Lernverfahren.

Testbasierte Lernverfahren versuchen mit Hilfe statistischer Tests die Ab- bzw. Unabhängigkeiten zwischen Zufallsvariablen unter Berücksichtigung der D-Separation (siehe Kapitel 2.3.3.2) zu entdecken, während man bei metrikbasierten Lernverfahren nach Netzstrukturen sucht, die eine bestimmte Bewertungsfunktion (Metrik) optimieren. Ein Vergleich der beiden unterschiedlichen Ansätze ist beispielsweise in [CGKBL02] ersichtlich, wobei gezeigt werden kann, dass testbasierte Verfahren den metrikbasierten Verfahren in der

Praxis meist qualitativ unterlegen sind, vor allem wenn die Trainingsmenge sehr klein oder verrauscht ist (siehe beispielsweise [HGC95]).

In Kapitel 3.3.4 gebe ich zunächst einen Überblick über bekannte metrikbasierte Lernalgorithmen und erläutere anschließend das Verfahren von Lam und Bacchus, das ich in Kapitel 3.3.7.1 zur Erstellung der kundenbezogenen Verhaltensnetze unter anderem um die Verwendung von Domänenwissen erweitere.

Verhaltensnetze können auf unterschiedliche Arten verschmolzen werden, wobei ich hauptsächlich die Durchschnitts-, Additions- und Relationsverschmelzung zur Modellierung von einzelnen Kunden und Kundengruppen verwende, deren konkrete Umsetzung Kapitel 3.3.5 zeigt. Darüber hinaus können Verhaltensnetze durch *Adaption* an neue Trainingsfälle angepasst werden, wobei sich sowohl die Wahrscheinlichkeitsverteilungen als auch die Netzstruktur anpassen lassen (siehe Kapitel 3.3.6).

3.3.1 Erstellung der Data-Mining-Tabellen

Data-Mining-Tabellen basieren sowohl auf kundenbezogenen Daten, wie Informationen von anonymisierten Kundenkarten, Kreditkarten, Kassenbons und Kundenbefragungen als auch auf Informationen über Einflussfaktoren, beispielsweise artikelbezogene Preis- und Promotionshistorien des Unternehmens oder historische Wetterdaten.

Abbildung 32 zeigt ein Beispiel einer kundenindividuellen Data-Mining-Tabelle bezogen auf einen einzelnen Artikel. Jede Spalte ist einer Zufallsvariablen zugeordnet, während die Zeilen jeweils einen Lern- bzw. Trainingsfall darstellen. Die Werte einer Zeile repräsentieren die konkreten Variablenzustände des jeweiligen Lernfalles. Die Lernfälle können mit Hilfe der speziellen Datumsspalte historisch sortiert werden.

Data-Mining-Tabellen sollten generell möglichst lückenlose historische Daten enthalten. Beispielsweise kann eine Data-Mining-Tabelle, die das Kaufverhalten eines Kunden tagesgenau für ein bestimmtes Jahr bezüglich eines bestimmten Artikels repräsentiert, aus 365 Zeilen bestehen (für jeden Tag des Jahres genau eine Zeile), wobei jede Zeile die kundenbezogene Absatzmenge zusammen mit den zu diesem Tag gültigen Artikelverkaufspreisen, Platzierungen, Promotionsarten, Konkurrenzpreisen etc. speichert. Bei einer Mehrzahl von Fragestellungen spielen dabei so genannte *negative Testfälle* für die Modellierung von Verhaltensnetzen eine wichtige Rolle: Im Beispiel wurde für jeden der 365 Tage die kundenbezogene Absatzmenge gespeichert, obwohl der Kunde in den meisten Fällen an den entsprechenden Tagen überhaupt nicht einkaufen war. Durch die Einbeziehung von negativen Fällen kann in den Verhaltensnetzen beispielsweise modelliert werden, welche Umstände dazu geführt haben könnten, dass der Kunde nicht einkaufen war. Die aus positiven und negativen Lernfällen resultierenden Verhaltensnetze repräsentieren nach der Initialisierung das durchschnittliche Verhalten des Kunden bezogen auf die zugrunde

liegende Zeitbasis.

Datum	Monat	Tag	Wochentag	Verkaufspreis	Promotion	Absatzmenge	Umsatz	...
01.01.2003	1	1	Mittwoch	1,99	ja	5	9,95	...
02.01.2003	1	2	Donnerstag	1,99	ja	0	0	...
03.01.2003	1	3	Freitag	1,99	ja	0	0	...
04.01.2003	1	4	Samstag	1,99	ja	0	0	...
05.01.2003	1	5	Sonntag	-	-	-	-	-
06.01.2003	1	6	Montag	2,19	nein	0	0	...
07.01.2003	1	7	Dienstag	2,19	nein	2	4,38	...
.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.
31.12.2003	12	31	Mittwoch	2,49	nein	0	0	...

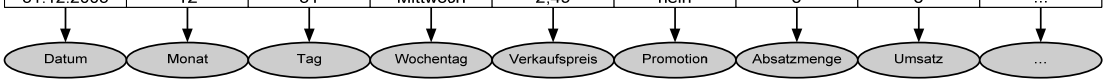


Abbildung 32: Beispiel einer kundenindividuellen Data-Mining-Tabelle

Data-Mining-Tabellen können je nach Anwendungsfall unterschiedliche Ausprägungen besitzen. Beispielsweise können negative Fälle in bestimmten Situationen entfallen, die Zeitbasis von Tag auf Woche, Monat, Quartal oder Jahr geändert, oder einzelne Knoten bzw. Spalten entfernt oder hinzugefügt werden.

Die Tabellen können auch aus aggregierten Werten bestehen, z. B. wenn sich das Netz nicht auf das kundenindividuelle Verhalten bezüglich eines Einzelartikels bezieht, sondern auf das aggregierte Verhalten (Durchschnitts- bzw. Additionsverhaltensnetze) bezüglich einer Warengruppe oder im Extremfall auf alle jemals vom Kunden erworbenen Artikel. Im Fall von Relationsverhaltensnetzen bezüglich einer Artikelmenge besteht die Data-Mining-Tabelle aus einer entsprechend erweiterten Anzahl von Spalten, da es in diesem Fall für jeden Artikel beispielsweise eine eigene Spalte „Verkaufspreis“ oder „Absatzmenge“ gibt.

Typischerweise liegen Kundendaten und Informationen über mögliche Einflussfaktoren in der Praxis nur sehr selten in einer Form vor, die direkt in Data-Mining-Tabellen übernommen werden kann. Deshalb sind in den meisten Fällen mehrere Zwischenschritte des *Data Preprocessing* nötig, um aus Rohdaten geeignete Data-Mining-Tabellen generieren zu können. Abbildung 33 zeigt einen Überblick über den Ablauf des Data-Preprocessing im Rahmen der Erstellung von Data-Mining-Tabellen aus realen Rohdaten.

Zur Erstellung von Data-Mining-Tabellen für kundenindividuelle Verhaltensnetze werden entsprechende historische, kundenbezogene und artikelbezogene Daten sowie historische Informationen über relevante interne und externe Einflussfaktoren benötigt. Die kunden- und artikelbezogenen Daten stehen in der Praxis in vielen Fällen durch die mittlerweile übliche Speicherung in Form von Kunden-, Transaktions- und Marktdaten in Supermärkten zur Verfügung, während andere externe Einflussfaktoren von Dienstleistungsunternehmen

angeboten werden (beispielsweise historische Wetterdaten vom Deutschen Wetterdienst). Diese Daten können fehlerhaft, unvollständig, verrauscht oder inkonsistent sein. Deshalb wird im Rahmen der Datenbereinigung versucht, fehlerhafte Daten zu korrigieren bzw. zu beseitigen, fehlende Daten zu ergänzen, verrauschte Daten zu glätten, Ausnahmen zu identifizieren und Inkonsistenzen zu beheben.

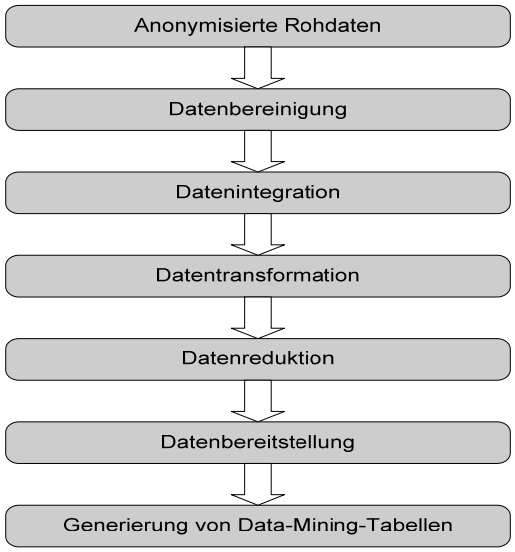


Abbildung 33: Prozess-Schritte der Generierung von Data-Mining-Tabellen

Bei der Datenintegration geht es um die Zusammenführung von Daten verschiedener Quellen. Dabei wird versucht, die möglicherweise auftretenden Datenkonflikte und Inkonsistenzen zu beseitigen und die Daten in eine einheitliche Struktur zu bringen.

Da die Rohdaten in den seltensten Fällen direkt zum Data Mining bzw. zur effizienten Erstellung von Data-Mining-Tabellen verwendet werden können, müssen sie nach ihrer Zusammenführung durch die Datentransformation in ein geeignetes Datenbankschema bzw. Format überführt werden (Datenkonvertierung).

Nach der Transformation können die Daten zur Steigerung der Zugriffsgeschwindigkeit bzw. zur Reduzierung des Speicherbedarfs in so genannten *Cubes* aggregiert werden.

Abschließend werden die Daten in Datenbanken bzw. Data Warehouses für die Generierung von Data-Mining-Tabellen bereitgestellt, wobei der Datenzugriff durch geeignete Optimierungen der Anfragen sowie der Verwendung von materialisierten Sichten bzw. Indizes optimiert wird.

Eine detaillierte Beschreibung der Prozesskette Gewinnung, Bereinigung, Integration, Konvertierung, Speicherung und optimierten Bereitstellung von Kundendaten im Rahmen des DFKI-Verbund-Projektes SimMarket XT bietet [SP04].

3.3.2 Diskretisierung

Jede Tabellenspalte S_i einer DM-Tabelle enthält die konkreten Werte der Testfälle einer Zufallsvariable X_i. Die Menge $W_i = \{w_1,\ldots,w_d\}$ definiert dabei die distinkten (d. h. unterschiedlichen) Werte von S_i.

Da die zulässigen Zustände einer diskreten Zufallsvariable X_i auf eine endliche Zustandsmenge $\{x_1, \ldots, x_z\}$ beschränkt sind (siehe Kapitel 2.3.2), deren Anzahl maßgeblich die Komplexität der Wahrscheinlichkeitstabellen beeinflusst, ist es essentiell, die möglicherweise große Anzahl unterschiedlicher w_i auf eine möglichst geringe Anzahl von Knotenzuständen x_j abzubilden. Dies kann durch entsprechende Diskretisierungsverfahren realisiert werden [HK01]. Die Werte einer Tabellenspalte S_i werden diskretisiert, indem ihre distinkten Werte $\{w_1, \ldots, w_d\}$ in geeignete Wertintervalle $DI = \{DI_1, \ldots , DI_k\}$ eingeteilt werden und jedes Vorkommen eines Wertes w_i in S_i anschließend durch Angabe des Intervalls DI_j ersetzt wird, in das w_i fällt, also für das gilt: $w_i \in [DI_j.\text{Start}, DI_j.\text{End}[$. Die zulässigen Zustände $\{x_1, \ldots, x_n\}$ von X_i sind anschließend die einzelnen Intervalle von DI.

Um den Erwartungswert einer Zufallsvariable basierend auf diskretisierten Wertintervallen berechnen zu können (siehe Kapitel 2.3.2), werden die *realen* Mittelwerte der einzelnen Intervalle benötigt. Der reale Mittelwert eines Intervalls $DI_i.$Average ist dabei der Durchschnittswert aller Werte in S_i, die in das Intervall DI_i fallen.

Diskretisierungen können auf verschiedene Arten durchgeführt werden. Die einfachste Art besteht in der Vorgabe geeigneter Intervalle für bestimmte semantische Variablentypen (so genannte *Benutzerintervalle* bzw. *User Intervals*). Beispielsweise kann eine Diskretisierung der Zufallsvariable „Tag“ in vier Intervalle durchgeführt werden, indem die Tage eines Monats in Monats-Viertel eingeteilt werden (Die Tage 1 bis 8 fallen dabei beispielsweise in das erste Monats-Viertels, während Tag 24-30 im letzten Monats-Viertel liegen). Analog kann die Variable „Monat“ auch in vier Jahresquartale eingeteilt werden.

Die Definition von Benutzerintervallen ist allerdings nicht in allen Fällen die beste Lösung. Beispielsweise ist es nicht trivial, eine geeignete vordefinierte Intervallaufteilung der Knoten „Absatzmenge“ und „Umsatz“ zu finden, da die konkreten Werte von Kunde zu Kunde stark variieren und selbst bei einzelnen Kunden, bezogen auf unterschiedliche Artikel bzw. Artikelmengen, sehr unterschiedlich sein können. In diesen Fällen müssen zur Laufzeit geeignete individuelle Diskretisierungen erstellt werden.

Bei der Diskretisierung von Werten, die zur Erzeugung von probabilistischen Verhaltensnetzen dienen, gilt es im Allgemeinen, die maximale Anzahl von zulässigen Wertintervallen durch einen Maximalwert Z zu begrenzen (in der Praxis verwende ich häufig $Z \leq 20$), um die Komplexität der Verhaltensnetze nicht unpraktikabel werden zu lassen, da die Anzahl der Intervalle maßgeblich die Größe der Apriori- bzw. bedingten Wahrscheinlichkeitstabellen bestimmt.

Generell gibt es zwei unterschiedliche Vorgehensweisen zur automatisierten Diskretisierung: Ein Verfahren startet entweder mit einem einzigen Intervall, das alle Werte umfasst und zerteilt dieses solange bis eine geeignete Aufteilung gefunden wurde und die Anzahl der Intervalle kleiner oder gleich Z ist (*Top-Down-Ansatz*) oder es beginnt damit, dass die distinkten Werte zu Beginn je ein so genanntes Punktintervall darstellen und benachbarte Intervalle so lange zu neuen Intervallen zusammengefasst werden, bis nach vorgebenden Kriterien wiederum eine geeignete Aufteilung mit maximal Z Intervallen gefunden wurde (*Bottom-Up-Ansatz*).

Die von mir implementierte Diskretisierungsmethode verfolgt den Bottom-Up-Ansatz, indem anfangs für jeden distinkten Wert w_i der Tabellenspalte ein Punktintervall erzeugt wird und die resultierenden Intervalle aufsteigend sortiert werden. Anschließend werden so lange benachbarte Intervallpaare zusammengefasst bis die Anzahl der Intervalle gleich Z ist, wobei ich die Intervalle vereine, die gemeinsam die geringste Anzahl von zutreffenden Werten in der Tabellenspalte besitzen. Auf diese Weise versuche ich eine Gleichverteilung der Intervallgewichte zu approximieren, die für verschiedene Inferenz-Algorithmen geeignet ist. Die algorithmische Vorgehensweise meines Verfahrens hat eine Komplexität von $O(|W|^2)$ und kann in dem folgenden Pseudocode beschrieben werden, der an C# angelehnt ist und den ich im Weiteren zur Darstellung meiner Algorithmen verwenden möchte:

```
Discretisation(W, Z)
   DI = NewList()
   Foreach (Wi in W) do
      DIi = NewInterval(Wi, Wi)
      DI.Add(DIi)
   DI.SortByStartValue(ASC)
   While (DI.Members.Count > Z)
      DIa = DI.Members[1]
      DIb = DI.Members[2]
      For(i = 2; i < DI.Members.Count ; i++) do
         If (DI.Members[i].Values.Count + DI.Members[i+1].Values.Count <
                                          DIa.Values.Count + DIb.Values.Count) then
            DIa = DI.Members[i]
            DIb = DI.Members[i+1]
      DIa.End = DIb.End
      DIa.Values = DIa.Values UNION DIb.Values
      DI.Delete(DIb)
Return DI
```

3.3.3 Generierung der Wahrscheinlichkeitstabellen

Auf Basis einer vorgegebenen Netzstruktur sowie einer entsprechenden diskretisierten Data-Mining-Tabelle können die Apriori-Wahrscheinlichkeitstabellen (APT) für Knoten ohne Eltern sowie die bedingten Wahrscheinlichkeitstabellen (CPT) für Knoten mit Eltern berechnet werden. Dabei gibt es zwei verschiedene Fälle:

1. alle Datensätze (Zeilen) der Data-Mining-Tabelle sind vollständig oder
2. einige Datensätze (Zeilen) der Data-Mining-Tabelle sind unvollständig.

Im Fall vollständiger Datensätze (*Complete Data*) kann die Erstellung der Tabellen direkt durchgeführt werden. Bei unvollständigen Zeilen (*Incomplete Data*) müssen die fehlenden Einträge eines Datensatzes erst durch „geschickte Schätzung“ ergänzt werden. Unvollständige Daten kommen in der Praxis vor, wenn bestimmte Informationen nur sporadisch erhoben werden (beispielsweise Daten über Verkaufspreise und Promotionen eines konkurrierenden Marktes), keine Informationen erhältlich sind, oder Datenpflegefehler vorliegen.

Wie in Kapitel 2.3.2 beschrieben, besteht eine Apriori-Wahrscheinlichkeitstabelle einer Zufallsvariable X_i aus einer Zeile, die eine entsprechende Wahrscheinlichkeitsverteilung repräsentiert. Dabei wird für jeden möglichen Variablenzustand x_i die Wahrscheinlichkeit $P(x_i)$ angeben (siehe Abbildung 34 oben). Um diese Wahrscheinlichkeitsverteilung anhand einer Data-Mining-Tabelle mit Datensätzen $D = \{d_1, \ldots, d_m\}$ zu berechnen, wird zunächst für jeden der z möglichen Zustände x_i aus $\{x_1, \ldots, x_z\}$ die Anzahl der Vorkommen in der Data-Mining-Tabelle ermittelt. Anschließend werden diese Werte *normalisiert*, indem sie durch die Gesamtanzahl m der Datensätze dividiert werden. Die Wahrscheinlichkeit $P(x_i)$ des Auftretens des Zustandes x_i berechnet sich somit wie folgt:

$$P(x_i) = \frac{|x_i|}{|D|} = \frac{|x_i|}{m},$$

wobei die Summe aller $P(x_i)$ den Wert 1 ergibt. Die algorithmische Vorgehensweise besitzt die Komplexität $O(|X_i| \cdot |D|) = O(z \cdot m)$ und lautet in Pseudocode:

```
LearnAprioriProbabilityTable(D, Xi)
   Xi.APT = NewTable[Xi.NodeStates.Count]
   Foreach(xi in Xi.NodeStates) do
      Xi.APT["Xi = xi"].Experience = D.Rows.CountMembersWhere("Xi = xi")
      Xi.APT["Xi = xi"].Probability = Xi.APT["Xi = xi"].Experience / D.Rows.Count
Return Xi.APT
```

Apriori

$X_i=x_1$	…	…	…	…	$X_i=x_n$
$P(x_1)$	…	…	…	…	$P(x_n)$

CPT

$X_i=x_1$	…	$X_i=x_n$
$P(x_1\|y_{11}\dots y_{p1})$	…	$P(x_n\|y_{11}\dots y_{p1})$
…	…	…
…	…	…
$P(x_1\|y_{1r1}\dots y_{prp})$	…	$P(x_n\|y_{1r1}\dots y_{prp})$

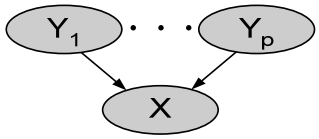


Abbildung 34: Aufbau von Apriori- und bedingten Wahrscheinlichkeitstabellen (CPT)

Zur Generierung einer bedingten Wahrscheinlichkeitstabelle (CPT) einer Zufallsvariable X_i mit den Elternknoten $Y = \{Y_1, \dots, Y_p\} = Parents(X_i)$, die die bedingte Wahrscheinlichkeitsverteilung $P(X_i|Parents(X_i))$ repräsentiert, wird für jeden möglichen Zustand x_i des Knotens X_i wiederum die Anzahl des Auftretens in der Data-Mining-Tabelle ermittelt, diesmal allerdings in Abhängigkeit aller Zustandskombinationen der Elternknoten. Das bedeutet, dass für alle möglichen Zustandskombinationen $y_i = parents(X_i)$ der Elternknoten Y zunächst die Anzahl der Vorkommen von x_i in der Data-Mining-Tabelle ermittelt wird, um anschließend die bedingten Wahrscheinlichkeiten $P(x_i|y_i)$ zu berechnen, indem die Vorkommen $|x_i|$ jeweils durch die Anzahl der Vorkommen $|y_i|$ der entsprechenden Elternzustandskombination dividiert werden (siehe Abbildung 34 unten). Die bedingte Wahrscheinlichkeit $P(x_i|y_{1a}\dots y_{pz})$ des Zustandes x_i unter der Bedingung einer konkreten Elternzustandskombination $y_{1a}\dots y_{pz}$ berechnet sich somit wie folgt:

$$P(x_i \mid y_{1a}...y_{pz}) = \frac{|\, x_i \wedge y_{1a}...y_{pz} \,|}{|\, y_{1a}...y_{pz} \,|}.$$

Diese Vorgehensweise stellt eine vereinfachte Form der Methode von Spiegelhalter und Lauritzen dar [SL90], die die Wahrscheinlichkeitsverteilungen in den CPT vor Ermittlung der Vorkommen der einzelnen x_i mit einer *Dirichlet-Verteilung* initialisieren. Die von mir verwendete Variante besitzt die Komplexität $O(Z^{|Y|}\cdot|X_i|\cdot|D|)$, wenn jeder Elternknoten Y_i höchstens Z unterschiedliche Zustände besitzt und lautet in Pseudocode:

```
LearnConditionalProbabilityTable(D, Y, Xi)
  Xi.CPT = NewTable[Xi.NodeStates.Count * Y.PossibleStateCombinations.Count]
  Foreach(yj in Y.PossibleStateCombinations) do
    Foreach(xi in Xi.NodeStates) do
      Xi.CPT["Xi = xi ∧ Y = yj"].Experience = D.Rows.CountMembersWhere("Xi = xi ∧ Y = yj")
```

X_i.CPT["$X_i = x_i \wedge Y = y_j$"].Probability = X_i.CPT["$X_i = x_i \wedge Y = y_j$"].Experience /
D.Rows.CountMembersWhere("$Y = y_j$")

Return X_i.CPT

Im Falle unvollständiger Datensätze müssen diese erst ergänzt werden bevor obige Methoden angewendet werden können. Ein ideales Verfahren zur Ergänzung von fehlenden Werten basiert auf der Berechnung der vollständigen bedingten Wahrscheinlichkeitsverteilung über alle Zufallsvariablen, um anschließend fehlende Werte geeignet ergänzen zu können. Dabei steigt die Komplexität allerdings exponentiell mit der Anzahl fehlender Werte. Zwei praktikable Approximationsverfahren sind das *Gibbs-Sampling* und die *Expectation Maximization* (*EM*) mit den Ausprägungen *Maximum Likelihood* (*ML*) und *Maximum Aposteriori Probability* (*MAP*) [DLR77]. Das Gibbs-Sampling ist eine stochastische Sampling-Technik, während EM ein iterativer, deterministischer Algorithmus ist. Beide Verfahren machen die stark vereinfachende Annahme, dass die fehlenden Werte unabhängig von den beobachteten sind, was in der Praxis allerdings in den seltensten Fällen tatsächlich zutrifft. Dennoch liefern beide Ansätze gute Schätzungen der fehlenden Werte in annehmbarer Laufzeit.

Eine extremere Form der Unvollständigkeit von Daten kann mit Hilfe so genannter *Hidden Variables* oder *Latent Variables* betrachtet werden. Dabei handelt es sich um die Analyse, ob in einem probabilistischen Netz ein oder mehrere Einflussfaktoren bzw. Zufallsvariablen gänzlich fehlen. Zur Entdeckung latenter Variablen sind in der Statistikforschung mehrere Verfahren bekannt, unter anderem die Faktorenanalyse. Für detaillierte Beschreibungen entsprechender Methoden siehe beispielsweise [LJC98].

3.3.4 Metrikbasierte Lernalgorithmen

Die Grundidee metrikbasierter Lernalgorithmen für probabilistische Netze ist die Suche nach einer geeigneten Netzinstanz, deren kausale Struktur eine vorgegebene Bewertungsfunktion (Metrik) optimiert. Die Metrik bewertet dabei eine Netzinstanz in der Regel sowohl bezüglich ihrer Qualität als auch bezüglich ihrer Komplexität.

Das größte Problem metrikbasierter Lernverfahren ist die enorme Größe des Suchraumes. Robinson gibt für die Anzahl möglicher Strukturen $S(n)$ eines Netzes mit n Knoten folgende rekursive Formel an [RWR77]:

$$S(n) = \sum_{i=1}^{n} (-1)^{i+1} \frac{n!}{(n-i)!\,i!} \cdot 2^{i(n-i)} \cdot S(n-i) \text{ mit } S(0)=1.$$

Die Suche nach einer optimalen Netzstruktur ist im Allgemeinen ein *NP-hartes* Problem

(siehe [CGH94] und [BRR95]). Metrikbasierte Lernverfahren nutzen daher in der Regel *Greedy*-Suchstrategien oder bekannte Heuristiken, wie *Hill Climbing* und *Simulated Annealing*, wodurch sich in der Praxis akzeptable Laufzeiten ergeben (siehe z. B. [HCMRK00] und [NBWSSS01]). Beispielsweise kann mit einer Netzinstanz ohne Kanten bzw. einer zufälligen oder vorgegebenen Netzstruktur begonnen werden und das Netz solange durch Hinzufügen, Herumdrehen oder Löschen von Kanten verändert werden, bis ein lokales Maximum der Bewertungsfunktion gefunden wurde. Zur Optimierung kann dieses Vorgehen um mehrere Durchgänge mit unterschiedlichen Ausgangssituationen ergänzt werden, um anschließend das lokale Maximum mit der höchsten Bewertung zu wählen. Der Suchraum kann dabei durch Domänenwissen stark eingeschränkt werden (siehe Kapitel 3.3.7). Vor der Suche können geeignete Korrelationsanalysen durchgeführt werden, um Paare von Zufallsvariablen zu entdecken, deren bedingte Abhängigkeiten sehr wahrscheinlich sind. Anschließend evaluiert man die Kanten während der Suche in absteigender Reihenfolge der Korrelationswerte der entsprechenden Knotenpaare.

Zur Beurteilung der Modellierungsgüte können unterschiedliche Maße verwendet werden. Ein Qualitätsmaß misst dabei die Güte, mit der eine Netzinstanz die Trainingsmenge $D = \{d_1,\ldots, d_m\}$ im probabilistischen Sinne erklärt, d. h. wie exakt die Wahrscheinlichkeitsverteilung innerhalb der Trainingsmenge durch die von der Netzinstanz repräsentierte Wahrscheinlichkeitsverteilung wiedergegeben wird. Die zwei bekanntesten Metriken dieser Art sind die *Belief-Scoring-Funktion* von Cooper und Herskovits [CH92] und die *MDL-Funktion* von Lam und Bacchus [LB94], die äquivalent zum *Bayesian Information Criterion* (*BIC*) von Schwarz [SG78] ist.

Metrikbasierte Lernverfahren wie das MDL-basierte Verfahren nach Lam und Bacchus setzen in der Regel die Vollständigkeit der Trainingsdaten voraus, d. h. dass bei allen Trainingsfällen d_i für alle Zufallsvariablen konkrete Werte vorliegen müssen. Ein Verfahren, das Strukturlernen und die Vervollständigung unvollständiger Trainingsfälle vereint, ist der *SEM-Algorithmus* (*Structural Expectation Maximization Algorithm*) von Friedman [FN97][FN98], der je nach Variante die MDL-Funktion oder Varianten der Metrik von Cooper und Herskovits verwendet.

3.3.4.1 Die MDL/BIC-Metrik

Die Idee der *MDL*-Metrik (*Minimum Description Language*) ist das geschickte Abwägen zwischen der Komplexität und der Qualität beim Erlernen probabilistischer Netze. Ziel ist die Optimierung der Modellqualität bei gleichzeitiger Minimierung der Modellbeschreibung in Form der MDL [RJ78]. Der *MDL-Wert* (*MDL-Score*) eines Bayes'schen Netzes B mit n Knoten, das basierend auf einer Trainingsmenge $D = \{d_1,\ldots, d_m\}$ mit m Trainingsfällen erlernt wurde, wird mit folgender Formel berechnet:

$$MDL_D(B) = L_D(B) - K_D(B) = \sum_{i=1}^{m} \log(P_B(d_i)) - \frac{\log(m)}{2} \#(B),$$

wobei #(*B*) die Komplexität des Bayes'schen Netzes angibt (beispielsweise die Anzahl der Parameter oder die Gesamtgröße aller APT und CPT) und *L* die so genannte *Log-Likelihood* ist, während *K* die Komplexität des Netzes „bestraft". Je höher die Log-Likelihood ist, umso exakter modelliert *B* die Wahrscheinlichkeitsverteilung, die in der Trainingsmenge *D* vorliegt. Der Strafterm ist nötig, da ohne ihn ein vollständig verbundenes Netz den höchsten Wert erhalten würde, das in der Regel zu komplex ist. Der Term $L_D(B)$ kann durch eine bekannte Dekomposition als Summe der Log-Likelihoods aller *n* Knoten $X_1,\ldots, X_n$ angegeben werden:

$$L_D(B) = \sum_{i=1}^{n} L_D(X_i) = \sum_{i=1}^{n} \sum_{x_i, parents(X_i)} P_D(x_i \mid parents(X_i)) \cdot \log \frac{P_B(x_i \mid parents(X_i))}{P_B(x_i) \cdot P(parents(X_i))},$$

wobei über alle möglichen Werte der X_i und Eltern $Parents(X_i)$ summiert wird und $L_D(X_i)$ auch als *Mutual Weight* von X_i bezeichnet wird (siehe [LB94]). P_D ist der Anteil der Trainingsfälle, bei denen X_i den Wert x_i annimmt, während die Elternknoten eine feste Wertekombination besitzen. Dieser Anteil ist die Zahl der entsprechenden Trainingsfälle geteilt durch die Gesamtanzahl der Trainingsfälle, bei denen die Elternknoten die entsprechende Wertekombination besitzen. Diese Vorgehensweise ist nur bei vollständigen und diskretisierten Trainingsdaten möglich. P_B entspricht den korrespondierenden Einträgen der zum Knoten X_i gehörenden CPT bzw. kann mit Hilfe der CPT durch einfaches Zählen und Dividieren ermittelt werden.

Der Strafterm $K_D(B)$ kann analog zur Log-Likelihood durch Dekomposition als Summe der Komplexitätsterme aller *n* Knoten angegeben werden:

$$K_D(B) = \sum_{i=1}^{n} K_D(X_i) = \sum_{i=1}^{n} \frac{\log(m)}{2} \#(X_i, Parents(X_i)),$$

wobei #(X_i, $Parents(X_i)$) die Größe bzw. Dimension der zu X_i gehörenden CPT repräsentiert. Der MDL-Score eines Bayes'schen Netzes *B* lässt sich somit als Summe der MDL-Werte seiner Knoten $X_1,\ldots,X_n$ angeben:

$$MDL_D(B) = \sum_{i=1}^{n} L_D(X_i) - \sum_{i=1}^{n} K_D(X_i) = \sum_{i=1}^{n} MDL_D(X_i).$$

Die Dekomposition des MDL-Wertes auf die Summe der MDL-Werte der einzelnen Knoten erlaubt die effiziente Berechnung des gesamten MDL-Wertes bei Veränderungen der Netzstruktur, da in diesen Fällen lediglich die MDL-Werte der durch die Veränderungen betroffenen Knoten neu berechnet werden müssen (Delta-Berechnung).

3.3.4.2 MDL-basiertes Strukturlernen nach Lam und Bacchus

Zur Erstellung der kundenbezogenen Verhaltensnetze nutzte ich das auf der MDL-Metrik basierende Lernverfahren von Lam und Bacchus [LB94]. In Kapitel 3.3.7 erweitere ich das Verfahren um Domänenwissen, wodurch sich der Suchraum in der Praxis in vielen Fällen sehr stark einschränken lässt.

Zum Lernen der Struktur eines Bayes'schen Netzes mit n Zufallsvariablen $X_1,\ldots,X_n$ teilen Lam und Bacchus den Suchraum in die Menge der Suchräume $\{S_0,\ldots,S_k\}$ auf, wobei S_i aus Netzinstanzen mit genau i Kanten besteht und k durch $n(n\text{-}1)/2$ beschränkt ist, da ein azyklisches Netz mit n Knoten maximal diese Anzahl an Kanten besitzen kann. Für jeden dieser Suchräume S_i wird nun nach einem Netz mit maximalem MDL-Wert gesucht, wobei nur eine gewisse Anzahl von Such-Durchgängen erlaubt ist, um die Laufzeit ihres Verfahrens einzuschränken. Anschließend wird das Netz mit dem höchsten MDL-Wert aller Suchräume favorisiert. Wenn man die Anzahl der Durchgänge pro Suchraum auf $O(n^2)$ beschränkt - was nach Lam und Bacchus in der Praxis zu befriedigenden Ergebnissen führt - besitzt der gesamte Lernprozess eine Komplexität von $O(mn^4)$, wobei m die Anzahl der Trainingsfälle repräsentiert.

Bei der Suche nach einer geeigneten Struktur werden im Vorfeld für alle Knotenpaare X_i, X_j die so genannte *Mutual Information* nach Chow und Liu [CL68] mit folgender Formel berechnet:

$$\text{Mutual Information}(X_i, X_j) = \sum_{X_i, X_j} P(X_i, X_j) \log \frac{P(X_i, X_j)}{P(X_i)P(X_j)},$$

wobei über alle möglichen Zustände von X_i und X_j aufsummiert wird. Die Mutual Information liefert einen Anhaltspunkt, inwieweit eine Abhängigkeit bzw. Relation zwischen zwei Zufallsvariablen besteht (je höher der Wert, desto eher existiert eine Interdependenz). Die bewerteten Knotenpaare werden anschließend nach der Höhe des Wertes sortiert. Auf diese Weise können bei der Suche nach einer geeigneten Netzstruktur zuerst Kanten zwischen Knoten ausprobiert werden, die mit hoher Wahrscheinlichkeit zu einer signifikanten

Erhöhung des MDL-Wertes beitragen. Beim Hinzufügen einer Kante $X_i \rightarrow X_j$ wird die konkrete Netzinstanz dem nächst höheren Suchraum zugeordnet, wobei überprüft wird, ob eventuell schon existierende Kanten der Form $X_j \rightarrow Y$ oder $Y \rightarrow X_j$ so umgedreht werden können, dass sich eine Erhöhung des MDL-Scores ergibt. Im positiven Fall wird rekursiv bei allen beteiligten Knoten getestet, ob eine Umkehr einer oder mehrerer Kanten zu einer weiteren Verbesserung des MDL-Wertes führen.

Die Veränderung des MDL-Wertes einer Netzinstanz beim Hinzufügen und eventuellen Umdrehen von Kanten kann effizient berechnet werden, da nur die Knoten auf die Veränderung des MDL-Wertes des Netzes einen Einfluss haben, deren Elternmenge sich verändert hat. D. h. um zu ermitteln, ob sich eine neue Kante $X_i \rightarrow X_j$ positiv oder negativ auf den MDL-Score eines Netzes auswirkt, reicht es aus, den alten MDL-Wert $MDL_D(X_j)$ mit dem neuen Wert zu vergleichen. Die Zusammensetzung des MDL-Scores aus Likelihood und Strafterm gewährleistet dabei, dass nur die Kanten hinzugefügt werden, die auch die Qualität des Netzes signifikant stärker erhöhen als dessen Komplexität.

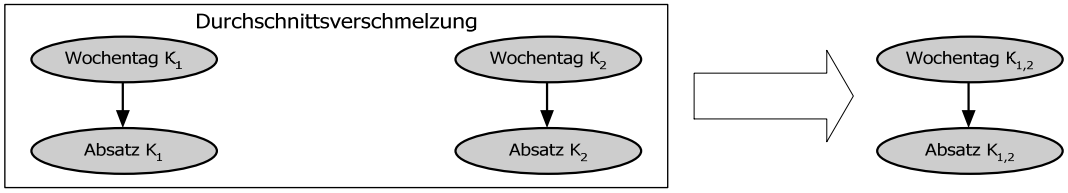
3.3.5 Verschmelzen von probabilistischen Verhaltensnetzen

Probabilistische Verhaltensnetze können neben der Erzeugung bzw. dem Lernen aus Data-Mining-Tabellen auch durch Verschmelzung mehrerer einzelner existierender Verhaltensnetze bzw. Teilnetze erzeugt werden. Ich verwende im Rahmen meiner Arbeit zur Bildung probabilistischer Holone drei unterschiedliche Verschmelzungsarten sowie deren Kombination: Durchschnitts-, Additions- und Relationsverschmelzung (siehe Kapitel 2.4.2.4). Alle drei Arten realisiere ich durch die aggregierte Verschmelzung semantisch identischer Knoten und anschließender optionaler Entdeckung neuer Relationen.

3.3.5.1 Realisierung der Durchschnittsverschmelzung

Die Durchschnittsbildung lässt sich je nach Art und Eigenschaften der zu verschmelzenden Netze bzw. Netzknoten auf zwei verschiedene Arten durchführen, wobei die zu verschmelzenden Knoten denselben semantischen Typ besitzen müssen. Ich erläutere die beiden unterschiedlichen Vorgehensweisen anhand eines einfachen Beispiels, in dem das Kaufverhalten zweier Kunden in Abhängigkeit des Wochentages zu einem durchschnittlichen Kundengruppenverhaltensnetz verschmolzen werden sollen (siehe Abbildung 35).

Der denkbar einfachste Fall liegt vor, wenn zwei Knoten desselben semantischen Typs miteinander verschmolzen werden sollen, deren Werte in der Data-Mining-Tabelle identisch sind. In meinem Beispiel trifft dies auf die Knoten „Wochentag K_1“ und „Wochentag K_2“ zu.



Wochentag K_1	Absatz K_1
1	5
2	7
3	10
4	2

Wochentag K_2	Absatz K_2
1	3
2	4
3	9
4	2

Abbildung 35: Beispiel einer Durchschnittsverschmelzung

Um die beiden Knoten mittels Durchschnittsverschmelzung zu vereinen, genügt die Erstellung einer Kopie bzw. das Unbenennen einer der beiden Knoten, um den neuen Knoten „Wochentag $K_{1,2}$“ zu erzeugen (siehe Abbildung 36). Im Rahmen meiner Arbeit verschmelze ich sämtliche zeitbezogenen Knoten ausschließlich auf diese Art der Durchschnittsbildung.

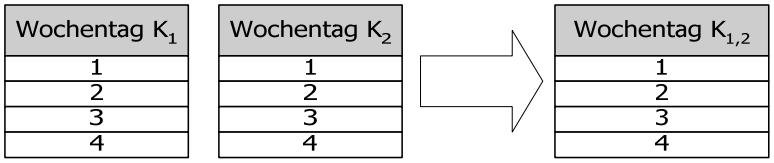
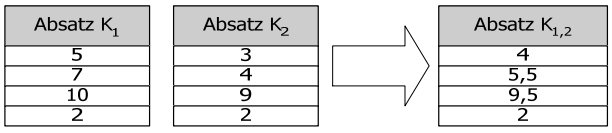


Abbildung 36: Durchschnittsverschmelzung bei identischen Data-Mining-Tabellenwerten

Falls die Data-Mining-Tabellenwerte der zu verschmelzenden Knoten nicht identisch sind, werden die Data-Mining-Tabellenwerte der beiden Knoten in einer neu erzeugten Tabellenspalte gemittelt, die anschließend bei Bedarf individuell diskretisiert werden kann. Abbildung 37 zeigt die Durchschnittsbildung der Knoten „Absatz K_1“ und „Absatz K_2“, wobei ein Durchschnittsknoten „Absatz $K_{1,2}$“ erzeugt wird, dessen Werte anschließend in fünf vorgegebene Intervalle diskretisiert werden.



Knoten	Werte	1-2	3-4	5-6	7-8	9-10	Erwartungswert
Absatz $K_{1,2}$	Anzahl	1	1	1	0	1	5,25
	Prozent	0,25	0,25	0,25	0	0,25	
	Mittel	2	4	5,5	-	9,5	

Abbildung 37: Mittelwertbildung der Data-Mining-Tabellenwerte

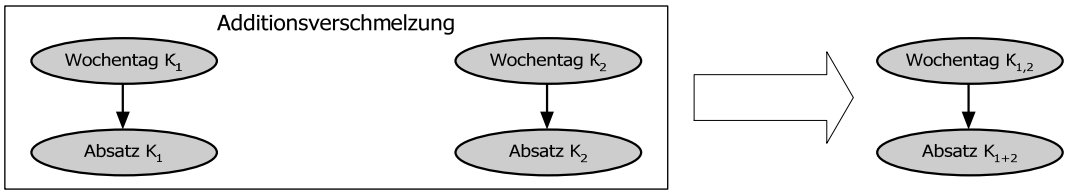
Anhand der gemittelten Werte lässt sich beispielsweise eine Apriori-Wahrscheinlichkeitsverteilung über die vorgegebenen Diskretisierungsintervalle erstellen. Darauf aufbauend kann der Erwartungswert des Knotens „Absatz $K_{1,2}$“ berechnet werden, der

in meinem Beispiel bei 5,25 liegt. Dieser sagt aus, dass die Kundengruppe bestehend aus den Kunden K_1 und K_2 im Schnitt 5,25 Artikeleinheiten pro Tag erwirbt.

Durchschnittsknoten, die durch diese Vorgehensweise erzeugt wurden, können anschließend wie „normale“ Knoten bei weiteren Strukturlernprozessen bzw. Verschmelzungen verwendet werden, da sie entsprechende Data-Mining-Tabellenspalten besitzen. In der Regel muss eine Diskretisierung der Werte des neu erzeugten Durchschnittsknotens durchgeführt werden, da sich das Wertespektrum durch die Durchschnittsbildung durchaus stark ändern kann. Beispielsweise besitzt der durchschnittliche Absatzknoten $K_{1,2}$ keinen Wert im Intervall „7-8“, während die Data-Mining-Tabelle von Kunde K_1 den Wert 7 enthielt.

3.3.5.2 Realisierung der Additionsverschmelzung

Die Additionsverschmelzung von vollständigen Verhaltensnetzen bzw. Teilnetzen lässt sich durch Summation der Werte der zu verschmelzenden Knoten realisieren. Zur Veranschaulichung verwende ich ein vereinfachtes Verhaltensnetz, das die Abhängigkeit der kundengruppenbezogenen Absatzmenge von konkreten Wochentagen modelliert, wobei die Kundengruppe wiederum nur aus den beiden Kunden K_1 und K_2 besteht (Abbildung 38).



Wochentag K_1	Absatz K_1
1	5
2	7
3	10
4	2

Wochentag K_2	Absatz K_2
1	3
2	4
3	9
4	2

Abbildung 38: Beispiel einer Additionsverschmelzung

Die Additionsverschmelzung der beiden kundenbezogenen Absatzknoten führt zu einem neuen Knoten „Absatz K_{1+2}“, dessen Data-Mining-Tabellenwerte durch Addition der entsprechenden Werte der Knoten „Absatz K_1“ und „Absatz K_2“ erzeugt werden (siehe Abbildung 39). Bei der Additionsverschmelzung ist es in den meisten Fällen notwendig, den erzeugten Additionsknoten anschließend neu zu diskretisieren, da sich der Wertebereich der konkreten Werte durch die Addition stark verändern kann. Im Beispiel wurde der Wertebereich in fünf vorgegebene Wertintervalle diskretisiert, die als Grundlage der beispielhaften Berechnung der Apriori-Wahrscheinlichkeitsverteilung und des darauf aufbauenden Erwartungswertes dienen.

Absatz K_1	Absatz K_2	Absatz K_{1+2}
5	3	8
7	4	11
10	9	19
2	2	4

Knoten	Werte	1-4	5-8	9-12	13-16	17-20	Erwartungswert
Absatz K_{1+2}	Anzahl	1	1	1	0	1	10,5
	Prozent	0,25	0,25	0,25	0	0,25	
	Mittel	4	8	11	-	19	

Abbildung 39: Additionsverschmelzung durch Addition der Data-Mining-Tabellenwerte

3.3.5.3 Realisierung der Relationsverschmelzung

Die Relationsverschmelzung zweier probabilistischer Netze bzw. Teilnetze N_1 und N_2 besteht im ersten Schritt aus der Ermittlung des maximalen Teilnetzes, das in beiden zu verschmelzenden Netzen identisch vorkommt. Alle Mitgliedsknoten des maximalen identischen Teilnetzes werden mit Hilfe der Durchschnittsverschmelzung für identische Knoten vereinigt. Alle übrigen Knoten und Kanten von N_1 und N_2 werden anschließend an das erzeugte Durchschnittsnetz angehangen, wobei zwischen den Knoten von N_1 und N_2 mit Hilfe von Korrelationsanalysen nach Abhängigkeiten gesucht wird (dem so genannten *emergenten Wissen*). Falls eine starke Abhängigkeit zwischen zwei Knoten A und B entdeckt wird, fügt man analog zur Vorgehensweise beim Lernen der Netzstruktur (siehe Kapitel 3.3.4.2) nacheinander die Kanten $A \rightarrow B$ und $B \rightarrow A$ in das Netz ein, die beide durch den MDL-Wert bewertet werden. Die Kante mit dem höheren positiven Beitrag zum MDL-Wert des gesamten Netzes wird übernommen. Falls keine der beiden Kanten den MDL-Wert positiv beeinflusst, werden beide Kanten verworfen. Neben der Verschmelzung der Netze werden auch die entsprechenden Data-Mining-Tabellen vereinigt, so dass das erzeugte Relationsnetz anschließend wie ein „normales“ Verhaltensnetz für weitere Verschmelzungen bzw. Strukturlernprozesse verwendet werden kann. Für eine detaillierte Beschreibung der Relationsverschmelzung siehe [SS04] bzw. [SB06].

3.3.6 Adaption

Unter der Adaption probabilistischer Verhaltensnetze versteht man die Anpassung eines bestehenden Netzes an neue Lerndaten. Die Adaption kann dabei aus der reinen Angleichung der bedingten Wahrscheinlichkeitstabellen bestehen oder auch aus der Modifikation der Struktur.

3.3.6.1 Adaption der Wahrscheinlichkeitstabellen

Die Adaption der Wahrscheinlichkeitsverteilungen in den Apriori- und bedingten Wahrscheinlichkeitstabellen eines probabilistischen Netzes durch neue Lernfälle kann auf dieselbe Weise realisiert werden wie das eigentliche Lernen selbst (siehe Kapitel 3.3.3): Die

neuen Fälle werden den vorhandenen Wahrscheinlichkeitsverteilungen durch einfaches Zählen und Normieren hinzugefügt. Ein Standardverfahren ist dabei das *AHUGIN-Verfahren* von Spiegelhalter und Lauritzen [SL90], das sowohl vollständige als auch unvollständige Lernfälle verarbeiten kann. Im Falle von unvollständigen Daten werden die fehlenden Werte durch Setzen der gegebenen Werte als Evidenzen im existierenden Netz und anschließender Inferenz approximiert (*Fractional Updating*).

Eine Methode, um ältere Lernfälle beim Lernen bzw. bei der Adaption weniger stark zu gewichten, ist der Einsatz von so genannten *Fading-Faktoren*. Konkret werden beim Zählen einzelne Lernfälle mit einem Faktor f mit $0 \leq f \leq 1$ multipliziert, wobei f durch eine Funktion in Abhängigkeit des Alters des Lernfalles bestimmt wird. In der Praxis nimmt das Gewicht des Lernfalles häufig mit dem Alter exponentiell ab. Durch Verwendung von Fading-Faktoren wird das aktuelle Verhalten stärker gewichtet, um somit Verhaltensänderungen und Trends zu berücksichtigen.

3.3.6.2 Adaption der Netzstruktur

Durch das Hinzufügen von neuen Lernfällen zu einem existierenden probabilistischen Netz kann es zu neuen kausalen Abhängigkeiten bzw. Unabhängigkeiten kommen (beispielsweise wenn ein Kunde sein Verhalten ändert), die eine Modifikation der Netzstruktur erfordern, d. h. bestimmte Kanten müssten hinzugenommen, gelöscht oder invertiert werden. Laut Jensen gibt es allerdings keine praktikable Methode zur inkrementellen Adaption der Struktur [JFV01]. Die Gründe dafür liegen darin, dass im Gegensatz zur Adaption der Wahrscheinlichkeitstabellen die Anpassung der Struktur in diskreten Schritten erfolgen muss, die jeweils eine ausreichende Menge von neuen Trainingsfällen voraussetzen. Jensen schlägt daher vor, die Struktur des Netzes periodisch neu zu lernen anstatt sie anzupassen. Andere Ansätze verwalten eine Menge alternativer Netzstrukturen (siehe beispielsweise [FG97]) oder sammeln solange neue Daten bis aufgrund der Datenbasis lokale Adaptionsentscheidungen getroffen werden (siehe beispielsweise [LB94b]). Einen detaillierten Vergleich verschiedener struktureller Adaptionsverfahren bietet [RS99].

3.3.7 Einsatz von Domänenwissen beim Erstellen von Verhaltensnetzen

Um Domänenwissen im Lernprozess für probabilistische Netze nutzen zu können, ergänze ich Knoten, Kanten, Data-Mining-Tabellen und Teilnetze, neben den schon eingeführten semantischen Knotentypen um weitere Attribute, mit deren Hilfe Domänenwissen repräsentiert werden kann. Die wichtigsten konkreten Attribute und Objekte, die ich zur Modellierung von Domänenwissen vorschlage bzw. entworfen habe, sind:

- objektorientierte Konzepte (Definition von Netzgrundstrukturen),
- strukturbeschreibende Knotentypen,

- strukturbeschreibende Kanten und Kantentypen,
- benutzerdefinierte Diskretisierungen bzw. Diskretisierungstypen sowie
- benutzerdefinierte Wahrscheinlichkeitstabellen und Adaptionsparameter.

Abbildung 40 zeigt die entsprechenden Erweiterungen zur Berücksichtigung von Domänenwissen anhand eines einfachen Beispiels, in dem drei Feature-Verhaltensnetze (Preis, Promotion und Termin) zu einem kundenindividuellen Einzelartikelverhaltensnetz verschmolzen werden, das den kundenindividuellen Absatz in Abhängigkeit der Einflussfaktoren modelliert.

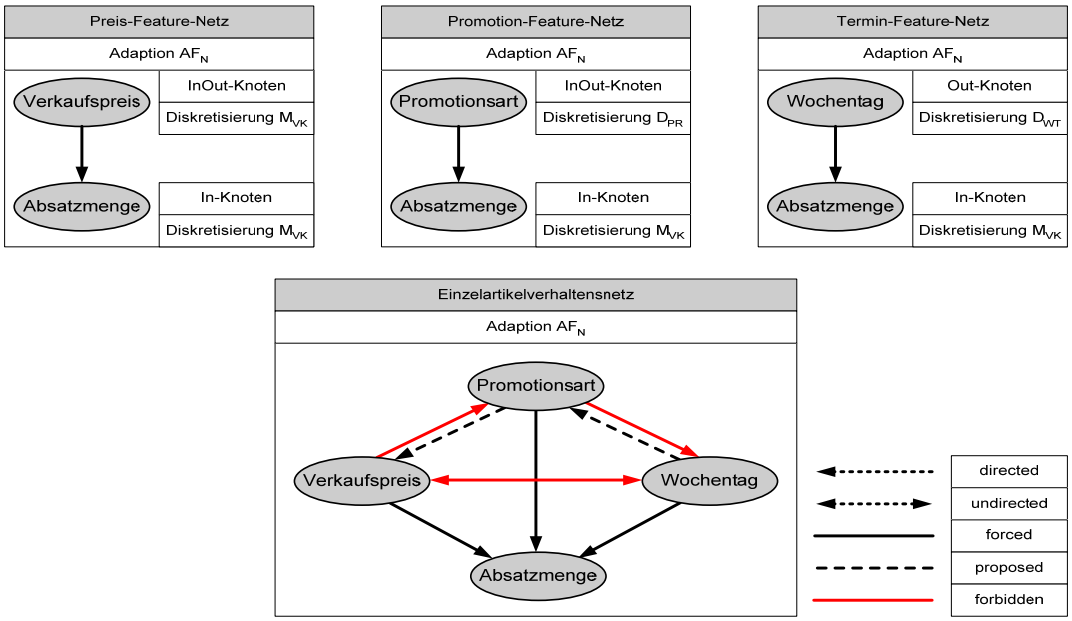


Abbildung 40: Domänenwissen bei der Erstellung und Verschmelzung von Verhaltensnetzen

Da bei der Modellierung individuellen Kundenverhaltens die potenziellen Einflussfaktoren in den meisten Fällen bekannt sind, ist es sinnvoll, die Feature-Netzstrukturen, wie ich sie in Kapitel 3.2.3 beschrieben habe, als Prototypen bzw. Initialnetze zu verwenden, wobei man dabei Konzepte der objektorientierten Bayes'schen Netze (OOBN) (siehe Kapitel 2.3.4.2) bzw. der probabilistischen relationale Modelle (PRM) (siehe Kapitel 2.3.4.3) übernehmen kann. Das bedeutet, dass für jedes Feature-Verhaltensnetz eine Netzklasse entworfen wird, die eine durchschnittliche Netzstruktur und darüber hinaus relevantes Domänenwissen repräsentiert. Beim Lernen einer neuen Instanz dient diese Netzklasse als Grundstruktur, die in den meisten Fällen nur noch geringfügig angepasst werden muss. Im Beispiel enthalten die Initialnetze der einzelnen Features Preis, Promotion und Termin zur Vereinfachung nur jeweils einen einzigen featurebezogenen Einflussfaktor und die abhängige Response-Variable „Absatzmenge".

Im Folgenden wird erläutert, wie ich Domänenwissen in den Teilschritten des Lernens und Erstellens probabilistischer Verhaltensnetze modelliere und zur Verbesserung der Laufzeit und Qualität nutze:

- beim Strukturlernen und Verschmelzen von Netzen bzw. Teilnetzen,
- bei der Diskretisierung,
- beim Erstellen der Apriori- und bedingten Wahrscheinlichkeitstabellen und
- bei der Adaption.

3.3.7.1 Domänenwissen beim Strukturlernen und Verschmelzen von Netzen bzw. Teilnetzen

Da die Ermittlung der optimalen Struktur eines probabilistischen Verhaltensnetzes ein NP-hartes Problem ist, muss versucht werden, den Suchraum in der Praxis möglichst stark einzuschränken. Ich verwende dazu unter anderem an OOBN bzw. PRM angelehnte Konzepte, indem ich beim Erlernen der individuellen Struktur einer konkreten Netzinstanz eine vordefinierte Netzklassen als Initialstruktur verwende, die meist nur geringfügig an die konkreten Lerndaten angepasst werden muss. Beispielsweise kann beim Lernen eines Promotion-Feature-Verhaltensnetzes (siehe Kapitel 3.2.3.2) eine Kante zwischen Platzierung und Absatzmenge vorgegeben sein, die dann aber bei einem bestimmten Kunden entfernt werden muss, da sich bei ihm durch Analyse der kundenbezogenen historischen Transaktionsdaten keine Verbindung zwischen Platzierung und Kaufentscheidung entdecken lässt.

Zur Modellierung von strukturbezogenem Domänenwissen in den Netzklassen habe ich zur Vorgabe der Struktur drei *strukturbeschreibende Kantentypen* eingeführt, die entweder *gerichtet* (*directed*) oder *ungerichtet* (*undirected*) sein können:

- *erzwungene Kanten* (*Forced Edges*),
- *vorgeschlagene Kanten* (*Proposed Edges*) und
- *verbotene Kanten* (*Forbidden Edges*).

Erzwungene Kanten erzwingen die Existenz entsprechender Kanten und müssen in jeder individuellen Instanz übernommen werden. Im Beispiel (Abbildung 40 s. o.) wurden bei allen Feature-Netzen erzwungene Kanten zwischen den Einflussfaktorknoten und dem Knoten „Absatzmenge“ vorgegeben. Somit wird davon ausgegangen, dass diese Faktoren immer einen starken Einfluss auf den Absatz besitzen (was in der Praxis allerdings nicht immer so sein muss) oder dass die entsprechenden Einflüsse auf jeden Fall modelliert werden sollen, unabhängig davon, wie stark diese tatsächlich sind.

Vorgeschlagene Kanten dienen als Strukturrichtlinien. Sie geben an, dass diese Kanten in der Regel wichtige Abhängigkeiten modellieren, die aber nicht bei jeder Instanz mit ausreichender Signifikanz vorliegen müssen. Beispielsweise symbolisiert die gerichtete

vorgeschlagene Kante zwischen „Promotionsart“ und „Verkaufspreis“, dass sehr wahrscheinlich eine Abhängigkeit des Verkaufspreises von der Art der Promotion existiert, da Promotionen häufig mit gleichzeitigen Preissenkungen durchgeführt werden, was allerdings nicht für jeden Artikel gelten muss.

Verbotene Kanten geben an, dass zwischen zwei Knoten auf keinen Fall eine Abhängigkeit modelliert werden soll. Im Beispiel wurde eine ungerichtete verbotene Kante zwischen „Verkaufspreis“ und „Wochentag“ vorgegeben, da zwischen den beiden Einflüssen entweder keine Abhängigkeit vermutet wird oder diese zur Einschränkung der Komplexität ignoriert werden soll.

Basierend auf den strukturbeschreibenden Vorgaben in einer vordefinierten Netzklasse *BC* habe ich folgenden Struktur-Lern-Algorithmus entwickelt:

1. Erstellung eines neuen kantenlosen Netzes mit allen im Initialnetz *BC* befindlichen semantischen Knoten.
2. Einfügen aller in der Netzklasse definierten gerichteten erzwungenen Kanten.
3. Ausprobieren beider Richtungen jeder ungerichteten erzwungenen Kante und anschließendem Einfügen der Kante in der Richtung, in der sie den besseren MDL-Wert erzeugt.
4. Durchführung einer Korrelationsanalyse aller in den vorgeschlagenen Kanten vorkommenden Knotenpaare mit Hilfe des Mutual-Information-Gewichtes (siehe Kapitel 3.3.4.2) und anschließender Sortierung der Knotenpaare nach Höhe der Korrelation.
5. Ausprobieren aller vorgeschlagenen Kanten in absteigender Reihenfolge.
    a. Im Falle einer gerichteten vorgeschlagenen Kante: falls die Kante den MDL-Wert des Netzes erhöht, wird sie eingefügt, andernfalls verworfen.
    b. Im Falle einer ungerichteten vorgeschlagenen Kante: Ausprobieren beider Richtungen. Falls die Kante mit der höheren MDL-Bewertung den MDL-Wert des Netzes verbessert, wird sie eingefügt, ansonsten werden beide Kantenrichtungen verworfen.

Nach Schritt 5 erhält man auf diese Weise eine konkrete Netzinstanz, bei der das in der vorgegebenen Grundklasse definierte strukturbezogene Domänenwissen individuell auf den konkreten Fall angewendet wurde.

Anschließend ist es möglich nach weiteren relevanten Kanten zu suchen, wobei wiederum Domänenwissen genutzt wird, um den verbleibenden Suchraum weiter stark einzuschränken. Generell gilt es, gerichtete Kanten zwischen Knotenpaaren auszuprobieren, bei denen eine entsprechende Kante nach obigem Schritt 5 nicht existiert bzw. noch nicht ausprobiert wurde. Der Suchraum kann dabei stark eingeschränkt werden, indem man die Kanten, die in der

Netzklasse als verbotene Kanten definiert sind, nicht berücksichtigt. Darüber hinaus können auch folgende *strukturbeschreibenden Knotentypen* zur Reduktion der Anzahl der zu untersuchenden Kanten berücksichtigt werden:

- *In-Knoten*,
- *Out-Knoten* und
- *InOut-Knoten*.

Ein In-Knoten darf nur Eltern besitzen aber keine Kinder, während ein Out-Knoten nur Kinder, aber keine Eltern haben darf. Ein *InOut-Knoten* kann sowohl Eltern als auch Kinder besitzen. Diese Knotentypen können neben den semantischen Knotentypen genutzt werden, um den Suchraum weiter stark einzuschränken, unter der Berücksichtigung, dass nur Knoten bestimmter Typen oder Typenkombinationen durch Kanten miteinander verbunden werden dürfen.

Nach Ermittlung der Menge der möglichen Kanten werden diese wie vorgeschlagene Kanten behandelt, d. h. sie werden nach einer Korrelationsanalyse in absteigender Reihenfolge nacheinander ausprobiert und im Falle einer Erhöhung des Netz-MDL-Wertes übernommen. Alternativ kann auch eine an Lam und Bacchus angelehnte Vorgehensweise verwendet werden (siehe Kapitel 3.3.4.2), bei der das nach Schritt 5 erzeugte Netz als Initialnetz dient und in jedem der $O(n^2)$ Suchräume das beschriebene Domänenwissen auf analoge Weise genutzt wird, um die Anzahl der möglichen Kantenkombinationen deutlich einzuschränken. Der gesamte Lernalgorithmus mit Berücksichtigung von Domänwissen (in Form einer vorgegebenen Netzklasse *BC*) lautet in Pseudocode wie folgt:

```
LearnBehaviourNetworkWithDomainKnowledge(BC, option)
   BNet = NewBehaviourNetwork(BC)
   Foreach(Ei in BNet.DirectedForcedEdges) do
      BNet.Edges.Add(Ei)
   Foreach(Ei in BNet.UndirectedForcedEdges) do
      BNet2 = BNet.Clone()
      If (MDLScore(BNet.Edges.Add(Ei.NodeA, Ei.NodeB)) <
                        MDLScore(BNet2.Edges.Add(Ei.NodeB, Ei.NodeA))) then
         BNet = BNet2
   Foreach(Ei in BNet.ProposedEdges) do
      CalculateMutualInformation(Ei.NodeA, Ei.NodeB)
   BNet.ProposedEdges.SortyByMutualInformation(DESC)
   Foreach(Ei in BNet.ProposedEdges) do
      Switch(Ei.Direction)
         Case „directed“:
```

```
            BNet2 = BNet.Clone()
               If (MDLScore(BNet2.Edges.Add(Ei)) > MDLScore(BNet)) then
                  BNet = BNet2
               Break
            Case "undirected":
               BNet1 = BNet.Clone()
               BNet2 = BNet.Clone()
               BNetMax = BNet1
               If (MDLScore(BNet2.Edges.Add(Ei.NodeA, Ei.NodeB)) >
                                 MDLScore(BNet1.Edges.Add(Ei.NodeB, Ei.NodeA))) then
                  BNetMax = BNet2
               If (MDLScore(BNetMax) > MDLScore(BNet)) then
                  BNet = BNetMax
               Break
      If (option = "LamBacchus") then
         BNet = LamBacchus(BNet)
      Else
         BNet.PossibleEdges = NewList()
         Foreach(Outi in BNet.OutNodes) do // OutNodes := OutNodes UNION InOutNodes
            Foreach(Inj in BNet.InNodes) do // InNodes := InNodes UNION InOutNodes
               If (Outi = Inj) then skip
               If (BNet.Edges.Exists(Outi, Inj) OR BNet.Edges.Exists(Inj, Outi)) then skip
               If (BNet.ForbiddenEdges.Exists(Outi, Inj)) then skip
               BNet.PossibleEdges.Add(Outi, Inj)
         Foreach(Ei in BNet.PossibleEdges) do
            CalculateMutualInformation(Ei.NodeA, Ei.NodeB)
         BNet.PossibleEdges.SortyByMutualInformation(DESC)
         Foreach(Ei in BNet.PossibleEdges) do
            BNet2 = BNet.Clone()
            If (MDLScore(BNet2.Edges.Add(Ei)) > MDLScore(BNet)) then
               BNet = BNet2
Return BNet
```

Der Algorithmus besitzt in der Variante ohne Lam und Bacchus eine Komplexität von $O(n^2 \cdot |D| \cdot Z^n)$, wenn jeder Knoten X_i höchstens Z unterschiedliche Zustände besitzt, da man zwischen höchstens $O(n^2)$ Knotenpaaren entsprechende Kanten mit Hilfe des MDL-Scores ausprobieren muss, wobei die Berechnung der neuen CPT und das anschließende Ermitteln des MDL-Gewichtes jeweils die Komplexität $O(|D| \cdot Z^n)$ aufweist. Mit anschließendem um

Domänenwissen modifizierten Verfahren nach Lam und Bacchus erhöht sich die Laufzeit auf $O(n^4 \cdot |D| \cdot Z^n)$, da dort in $O(n^2)$ Suchräumen jeweils $O(n^2)$ Kantenkombinationen getestet werden (siehe Kapitel 3.3.4.2).

Beim Prozess des Verschmelzens von probabilistischen Netzen bzw. Teilnetzen kann das Domänenwissen in ähnlicher Weise wie beim Lernen der Struktur zur Einschränkung des Suchraums und somit zur Optimierung der Laufzeit verwendet werden.

Bei der Durchschnittsverschmelzung können Knoten mit derselben temporalen Semantik, wie beispielsweise *Wochentag*, *Tag*, *Woche*, *Monat* oder *Quartal*, in der Regel durch Durchschnittsverschmelzung für identische Knoten vereinigt werden, da die Gleichheit dieser Werte eine Vorraussetzung der Verschmelzung darstellt. Generell können für alle Verschmelzungsarten Regeln bzw. Knotenattribute erstellt werden, die angeben, auf welche Art ein Knoten bzw. semantische Knotentypen verschmolzen werden können oder sollen.

Bei der Suche nach emergentem Wissen nach einer Verschmelzung zweier Netze N_1 und N_2 – also der Findung von möglichen Abhängigkeiten zwischen Knoten aus N_1 und N_2 – kann das Domänenwissen ebenfalls genutzt werden, um den Suchraum stark einzuschränken. Beispielsweise kann festgelegt sein, zwischen welchen semantischen Knotentypen überhaupt Kanten existieren dürfen oder es werden analog zum Strukturlernen eine Menge von erzwungenen, vorgeschlagenen und verbotenen Kanten vorgegeben, wobei die strukturbeschreibenden Knotentypen erneut zur Beschränkung der Möglichkeiten dienen können.

3.3.7.2 Domänenwissen bei der Diskretisierung

Neben den semantischen und strukturbeschreibenden Knotenattributen benutze ich *Knotendiskretisierungstypen*, die angeben, mit welchem Diskretisierungsverfahren die Werte einer Zufallsvariablen diskretisiert werden sollen. Neben verschiedenen automatischen Diskretisierungsmethoden verwende ich darüber hinaus *benutzerdefinierte Diskretisierungen*, die durch konkrete *Integer*- oder *Double-Werte* und *Zeichenketten* (*Strings*) definiert werden können, um die Diskretisierung zu beschleunigen bzw. zu vereinheitlichen. Dabei können bestimmten semantischen Knotentypen spezielle benutzerdefinierte Diskretisierungen oder bestimmte automatische Diskretisierungsverfahren mit entsprechenden Parametrisierungen zugewiesen werden. In Abbildung 40 wurden den Knoten „Absatzmenge“ und „Verkaufspreis“ unterschiedliche automatische Diskretisierungsmethoden zugeordnet, während die Knoten „Promotionsart“ und „Wochentag“ konkrete benutzerdefinierte Diskretisierungen besitzen.

3.3.7.3 Domänenwissen beim Erstellen der Wahrscheinlichkeitstabellen

Bei der Erstellung der Wahrscheinlichkeitstabellen kann es in der Praxis vorkommen, dass

für bestimmte Knoten keine genauen Informationen über Apriori-Wahrscheinlichkeitsverteilungen oder bedingten Wahrscheinlichkeitsverteilungen in den Vergangenheitsdaten vorliegen oder diese aufgrund unzureichender Informationen nicht exakt ermittelt werden können. In diesen Fällen ist es möglich, allgemeingültige oder durch Experten vorgegebene Wahrscheinlichkeitsverteilungen zu verwenden. Dabei sind entweder aus empirischen Daten ermittelte Statistiken oder von Domänenexperten manuell erstellte Verteilungen anwendbar. Analog können für einzelne Kunden, bei denen die Datengrundlage für einige Variablen sehr dünn ist, in manchen Fällen Verteilungen verwendet werden, die sich auf ähnliche Artikel beziehen. Alternativ können kundenbezogene Durchschnittsverteilungen oder Verteilungen von ähnlichen Kunden als Approximation dienen (siehe Kapitel 3.2.4.2 bzw. 3.2.4.3).

3.3.7.4 Domänenwissen bei der Adaption

Bei der Adaption probabilistischer Netze an neue Daten sowie bei der Verwendung von Fading-Faktoren zur stärkeren Gewichtung von jüngeren Lernfällen kann Domänenwissen auf unterschiedliche Arten angewendet werden. Zum einen kann die Durchführung von strukturellen Anpassungen durch geeignete Heuristiken angestoßen werden, d. h. die Menge der neuen Lernfälle kann mit entsprechenden Verfahren analysiert werden, um zu bestimmen, ob eine Adaption der Netzstruktur durchgeführt werden soll. Zum anderen können neue Daten unterschiedlich stark gewichtet werden, je nachdem wie relevant oder typisch sie für die zugrunde liegende Domäne sind. Dazu werden durch einen Domänenexperten Regeln bzw. Gewichtungen für Lernfälle definiert (so genannte *Expert Elicitation*). In Abbildung 40 wurde für alle Verhaltensnetze die gleiche Gewichtungsfunktion für die stärkere Berücksichtigung von jüngeren Lernfällen vorgegeben.

Analog können Fading-Faktoren durch Domänwissen optimiert werden, d. h. die Auswahl der entsprechenden Fading-Funktion kann an die aktuelle Problemstellung angepasst werden, je nachdem wie „schnelllebig“ die Verhaltensmuster bzw. Lernfälle sind.

4 Simulation mit probabilistischen Kundenagenten

In diesem Kapitel erläutere ich die Vorgehensweise zur Simulation des individuellen Kaufverhaltens einzelner Kunden mit Hilfe der vorgestellten probabilistischen Verhaltensnetze. Kapitel 4.1 erklärt zunächst die Idee der szenariobasierten Kundensimulation, auf deren Grundlage das individuelle Kaufverhalten bezüglich definierbarer Einkaufsszenarien simuliert werden kann. In Kapitel 4.2 beschreibe ich den eigentlichen Simulationsprozess, der auf probabilistischem Schlussfolgern basiert. Kapitel 4.3 gibt abschließend einen detaillierten Überblick über unterschiedliche Inferenz-Algorithmen für probabilistische Verhaltensnetze, die ich in Abhängigkeit der Szenarioparameter zur Verhaltenssimulation verwende.

4.1 Szenariobasierte Simulation individuellen Kaufverhaltens

Das individuelle Kaufverhalten einzelner Kunden simuliere ich, indem die entsprechenden Kundenagenten mit Einkaufssituationen konfrontiert werden, die durch *Szenarien* definiert sind. Abbildung 41 zeigt den Prozess der szenariobasierten kundenindividuellen Verhaltenssimulation, bei der ein Szenario vorgegeben wird, das eine mehr oder weniger komplexe Einkaufssituation beschreibt, auf die der Kunde mit Hilfe seiner individuellen Verhaltensnetze reagiert. Dabei werden die Verhaltensnetze genau auf die im Szenario beschriebenen Sachverhalte zugeschnitten, d. h. die Netze werden ausschließlich basierend auf Daten erlernt, die sich auf die im Szenario vorgegebenen Artikel und deren Eigenschaften sowie Einflussfaktoren und Konfigurationen beziehen. Die Reaktionen des Kunden werden mit Hilfe von Schlussfolgerungen - basierend auf Inferenz-Algorithmen der probabilistischen Verhaltensnetze - ermittelt und anschließend durch geeignete Kennzahlen wie Umsatz, Absatz oder Gewinn repräsentiert und als Resultate der Verhaltenssimulation im Szenario gespeichert.

Ein Kunde kann beispielsweise mit einer Menge einzelner Artikel einer bestimmten Warengruppe mit unterschiedlichen Eigenschaften bezüglich Marke, Qualität, Bewerbung, Preisen und Regalplatzierungen konfrontiert werden. Basierend auf den entsprechenden kundenindividuellen Verhaltensnetzen mit den relevanten Feature-Teilnetzen bezüglich der vorgegebenen Artikel wird nun simuliert, wie viele Einheiten der Kunde von jedem der präsentierten Produkte mit den beschriebenen Eigenschaften wahrscheinlich erwerben wird, wobei weitere im Szenario definierte Einflussfaktoren, wie beispielsweise zeitliche und saisonale Einflüssen oder Wettbewerberpreise, berücksichtigt werden können.

Ein Szenario kann sich auf einzelne Kundeneigenschaften beziehen, wie allgemeine Preis- oder Werbesensibilität oder auf komplexe Einkaufssituationen, bei denen sich über einen längeren Zeitraum bei einer großen Menge von Artikeln bestimmte Parameter wie Preise oder Bewerbungen mehrmals ändern können.

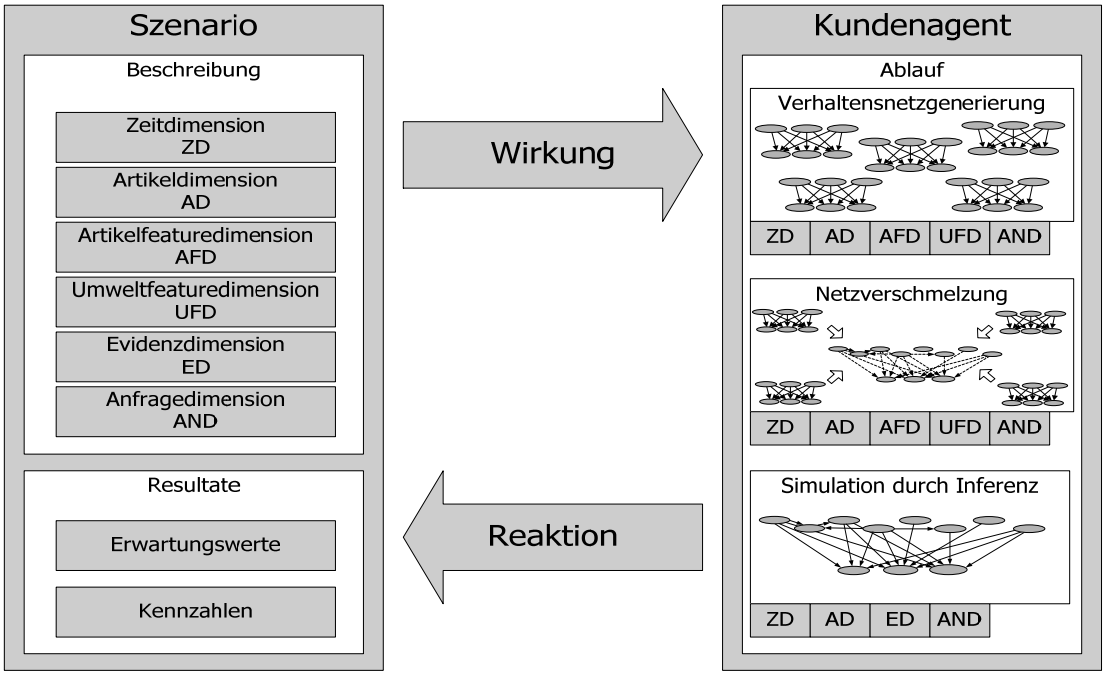


Abbildung 41: Szenariobasierte Simulation individuellen Kaufverhaltens

Konkret gibt die Szenariobeschreibung die Parameter zur Generierung der entsprechenden probabilistischen Verhaltensnetze und der durchzuführenden Schlussfolgerungen vor. Die Beschreibung ist dabei in so genannte *Simulationsdimensionen* unterteilt:

- *Zeitdimension*,
- *Artikeldimension*,
- *Artikelfeaturedimension*,
- *Umweltfeaturedimension*,
- *Evidenzdimension und*
- *Anfragedimension.*

Die Simulationsdimensionen umfassen alle nötigen Angaben über das Szenario wie Kaufzeitpunkt bzw. -zeitraum (Zeitdimension), relevante Artikel (Artikeldimension) und Artikeleigenschaften (Artikelfeaturedimension), Umwelteinflüsse (Umweltfeaturedimension) sowie die konkreten zu simulierenden Ausprägungen von Evidenzvariablen (Evidenzdimension) und die auszulesenden Effektvariablen (Anfragedimension).

Evidenzvariablen sind in der Regel Artikeleigenschaften und Umwelteinflüsse, während die Effektvariablen Kundenreaktionen repräsentieren. Im Falle einer *diagnostischen* Simulation (siehe Kapitel 4.2) beinhaltet die Evidenzdimension allerdings die gewünschten Kundenreaktionen und die Anfragesimulation die zu untersuchenden Artikeleigenschaften und Umwelteinflüsse.

Die **Zeitdimension** definiert den zu simulierenden *Zeitpunkt* bzw. *Zeitraum ST* und die der Simulation zugrunde liegende *Zeitbasis BT*. Die Zeitbasis definiert dabei den kleinsten simulierbaren Zeitraum $|BT|$, dessen Minimum durch die Datengrundlage der Einzelhändler beschränkt ist. Da Einzelhändler allgemein in „Tagen", „Wochen", „Monaten" und „Quartalen" denken, eignen sich diese Zeiträume in der Praxis als Zeitbasen, wobei ich „Tag" und „Woche" favorisiere, da bei der Verwendung größerer Zeiträume in den meisten Fällen viele Informationen durch Datenaggregationen verloren gehen. Der zu simulierende Zeitraum muss mindestens die Dauer der Zeitbasis besitzen (entspricht quasi einer Zeitpunktsimulation mit $|ST| = |BT|$), oder einem Vielfachen entsprechen (Zeitraumsimulation mit $|ST| = a \cdot |BT|$, $a \in \mathrm{N}_+$). Beispielsweise kann mit der Zeitbasis „Tag" ein bestimmter Tag oder auch mehrere konkrete Wochen oder Monate eines angegebenen Jahres simuliert werden. Im Falle einer Zeitraumsimulation werden a Simulationsdurchläufe durchgeführt, wobei jeweils der aktuelle Zeitraum mit der Dauer der Zeitbasis simuliert wird. Falls kein konkreter Simulationszeitraum angeben wird, bezieht sich das Verhalten auf einen durchschnittlichen Zeitraum der Länge der Zeitbasis $|BT|$, also beispielsweise auf einen durchschnittlichen Einkaufstag. Die Zeitdimension definiert somit auch die Existenz bzw. Konfiguration der entsprechenden Zeitknoten im Verhaltensnetz und die Zeitbasis der historischen Lerndaten der zugrunde liegenden Data-Mining-Tabellen (siehe Kapitel 3.3.1).

Um den Artikelbezug der Kaufsituation zu definieren, können in der **Artikeldimension** einzelne Artikel, Artikelmengen oder die Gesamtheit aller Produkte ausgewählt werden, wobei die Gesamtheit aller Produkte für Fälle verwendet wird, in denen das *allgemeine* Kaufverhalten eines Kunden bezüglich bestimmter Features wie beispielsweise die individuelle Preissensibilität untersucht werden soll. Bei der Angabe konkreter Artikelmengen kann vorgegeben werden, ob das durchschnittliche, aggregierte oder abhängige Verhalten modelliert und simuliert werden soll. Im Falle des durchschnittlichen Verhaltens werden die Verhaltensnetze bezüglich der einzelnen Artikel mit Hilfe der Durchschnittsverschmelzung generiert, während die Einzelnetze beim aggregierten Verhalten durch Additionsverschmelzung vereinigt werden. Sollen die Abhängigkeiten zwischen den Artikeln berücksichtigt werden, wird nach der Erzeugung der artikelbezogenen Verhaltensnetze eine Relationsverschmelzung durchgeführt, um eventuelle Abhängigkeiten und emergentes Wissen zu entdecken, das anschließend modelliert und bei der Simulation berücksichtigt werden kann (für Details zu den verschiedenen Verschmelzungsarten siehe

Kapitel 3.3.5). In jedem Fall hat die Konfiguration der Artikeldimension Auswirkungen auf Form und Inhalt der zum Erzeugen der entsprechenden Verhaltensnetze notwendigen Data-Mining-Tabellen, da deren Trainingssätze nur Daten bezüglich der definierten Artikelmenge und gewählten Verschmelzungsformen enthalten dürfen.

Die **Artikelfeaturedimension** beschreibt die Konfiguration der Feature-Teilnetze, die sich auf Artikeleigenschaften beziehen, anhand derer man auf Teilverhaltensmuster des Kunden schließen kann (siehe Kapitel 3.2.3 für detaillierte Informationen über die einzelnen Features und Featurenetze). Die Artikelfeaturedimension definiert, welche Attribute bzw. Features (beispielsweise Verkaufspreis, Bewerbung oder Marke) als Teilnetze berücksichtigt werden sollen und gibt deren konkrete Struktur vor, d. h. welche Zufallsvariablen pro Feature mit welcher Konfiguration verwendet werden. So kann die Konfiguration der Artikelfeaturedimension die Berücksichtigung der artikelbezogenen Bewerbung (Promotion-Feature) mit der Zufallsvariable „Promotionsart“ jedoch ohne den Knoten „Platzierung“ vorgeben (siehe Kapitel 3.2.3.2). Alle zu berücksichtigen Feature-Teilnetze werden vor der Simulation unter Berücksichtigung der Angaben in der Zeit- und Artikeldimension erstellt und anschließend mit dem Gesamtnetz verschmolzen.

Die **Umweltfeaturedimension** dient analog zur Artikelfeaturedimension zur Konfiguration der zu berücksichtigenden Umwelteinflüsse wie Saison, Wetter, wirtschaftliche Situation oder besondere Events.

Die **Evidenzdimension** gibt die zu simulierenden punktuellen bzw. sequenziellen Evidenzen vor, die aus konkreten zeitbezogenen Werten und Zustandsbeschreibungen der Zufallsvariablen bestehen. Beispielsweise lassen sich für die Artikel der Artikeldimension konkrete Zustände für die in der Artikelfeaturedimension spezifizierten Eigenschaften angeben, z. B. konkrete Verkaufspreise oder Werbemaßnahmen innerhalb eingeschränkter Zeiträume. Analog können für die in der Umweltdimension spezifizierten Einflüsse konkrete Zustände vorgegeben werden, wie beispielsweise bestimmte Wettersituationen oder konkrete Events wie Rabattaktionen oder Valentinstag.

In der **Anfragedimension** sind schließlich die Zufallsvariablen und Kennzahlen definiert, die als Bewertung der Reaktionen oder zur Prognose bzw. Diagnose verwendet werden. Beispielsweise eignen sich als Reaktionskennzahlen die Erwartungswerte und Wahrscheinlichkeitsverteilungen der Knoten „Absatz“ bzw. „Umsatz“, um zu ermitteln, wie sich konkrete Evidenzen bezüglich der Artikeleigenschaften und Umwelteinflüsse auf die vom Kunden gekaufte Artikelmenge bzw. seinen Umsatz auswirken.

4.2 Simulation durch Schlussfolgern mit Verhaltensnetzen

Die Simulation der Reaktionen eines Kunden auf ein vorgegebenes Szenario führe ich mit Hilfe von Inferenz-Algorithmen für probabilistische Verhaltensnetze durch. Die Hauptaufgabe der Inferenz-Algorithmen besteht in der Berechnung der so genannten *Nachfolgewahrscheinlichkeitsverteilung* (*posterior probability distribution*) für eine Menge von *Anfragevariablen* (*query nodes*) bei Vorgabe konkreter *Evidenzen* für eine Menge von *Evidenzknoten* (*evidence nodes*). Dieser Inferenzprozess wird in der Literatur auch als *Schlussfolgerung*, *Reasoning*, *Belief Update*, *Belief Updating* oder *Probabilistic Inferenz* bezeichnet. Es existieren verschiedenartige Inferenz-Algorithmen für probabilistische Netze, die sich in exakte und approximative Verfahren unterteilen lassen (siehe Kapitel 4.3).

Beim szenariobasierten Schlussfolgern mit probabilistischen Verhaltensnetzen wird die Menge der Anfragevariablen in der Anfragedimension vorgegeben, während die Menge der Evidenzknoten und der konkreten Evidenzwerte aus der Konfiguration der Evidenzdimension ersichtlich sind. Je nach Wahl der vorgegebenen Anfrage- und Evidenzvariablen ergeben sich nach [KN04] unterschiedliche Arten des Schlussfolgerns (siehe Abbildung 42).

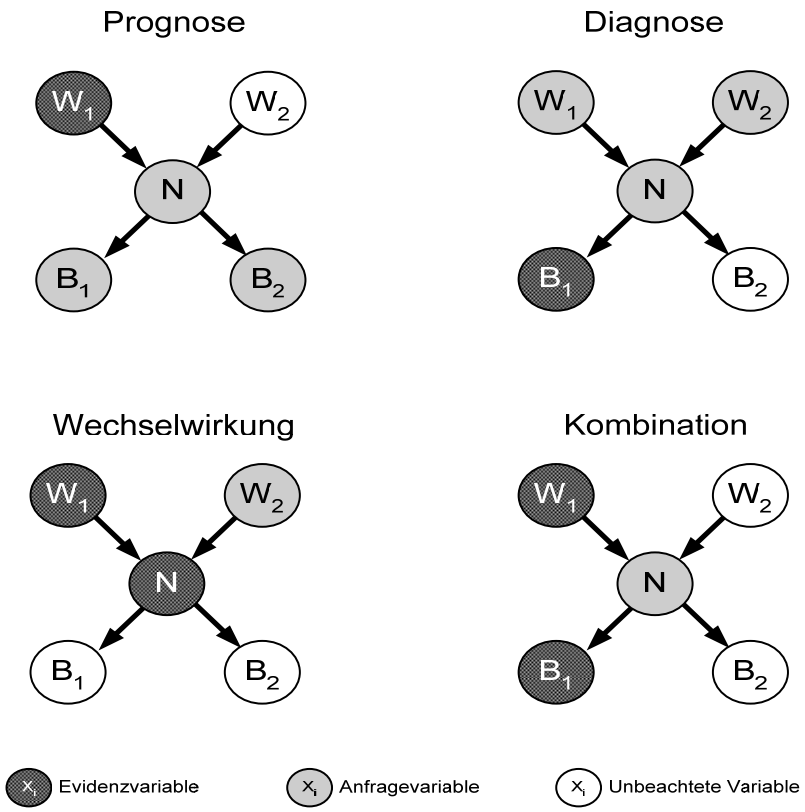


Abbildung 42: Schlussfolgerungsarten

Im Rahmen von **Vorhersagen** und **Prognosen** (*Predictive Reasoning*) werden in der Regel Evidenzen bei einer Menge von Einflussfaktorknoten bzw. Wurzelknoten gesetzt, um deren Auswirkungen auf die Wahrscheinlichkeitsverteilungen einer Menge von Effektknoten bzw. Blattknoten zu ermitteln. Diese Art des Schlussfolgerns nutze ich zur Verhaltenssimulation, indem ich bei den Knoten der Evidenzdimension entsprechende Zustände als Evidenzen vorgebe und die Reaktionen des Kunden aus den Effektknoten der Anfragedimension durch Inferenz berechne. Beispielsweise wird die Auswirkung eines konkreten Verkaufspreises und einer bestimmten Werbemaßnahme auf die kundenindividuelle Absatzmenge eines Artikels prognostiziert, indem bei den Einflussknoten „Verkaufspreis“ und „Bewerbung“ entsprechende Evidenzen gesetzt werden, die mit Hilfe eines Inferenz-Algorithmus über das Netz propagiert werden und anschließend der Erwartungswert des Knotens „Absatzmenge“ ermittelt wird.

Zum **diagnostischen** Schlussfolgern (*Diagnostic Reasoning*) werden Evidenzen bei einer Menge von Effektknoten bzw. Blattknoten gesetzt, um die Wahrscheinlichkeitsverteilungen bezüglich der Werte der möglichen Auslöserknoten bzw. Wurzelknoten zu ermitteln. Das Berechnen der Werte geschieht dabei quasi „entgegen“ der Kantenrichtung. Diese Art des Schlussfolgerns nutze ich zur Beantwortung diagnostischer Fragestellungen, indem ich die vom Kunden gezeigten oder gewünschten Reaktionen als Evidenzen in den Effektknoten setze und die durch Inferenz resultierenden Wahrscheinlichkeitsverteilungen der Auslöserknoten als Ergebnisse bzw. zur Ermittlung entsprechender Kennzahlen verwende. Beispielsweise kann ermittelt werden, welche konkreten Zustände der Einflussfaktoren „Verkaufspreis“ und „Bewerbung“ vorliegen müssen, damit ein Kunde einen bestimmten Absatz tätigt. Hierfür setzt man bei der Effektvariable „Absatzmenge“ den gewünschten Absatz per Evidenz und betrachtet nach durchgeführter Propagierung die Erwartungswerte bzw. die Wahrscheinlichkeitsverteilungen der Variablen „Verkaufspreis“ und „Bewerbung“.

Im Rahmen des Schlussfolgerns über **Wechselwirkungen von Einflussfaktoren** (*Intercausal Reasoning*) bezüglich gemeinsamer Effekte steht vor allem die Spezialform *Explaining Away* im Vordergrund. Dabei handelt es sich um eine spezielle „V-Struktur“, bei der zwei von einander unabhängige Einflussfaktorknoten jeweils eine Kante zu einem gemeinsamen Effektknoten besitzen. Beim Setzen von Evidenzen am Effektknoten verändern sich im Allgemeinen die Wahrscheinlichkeitsverteilungen der beiden Einflussfaktoren in der Art, dass das Auftreten der Einflüsse wahrscheinlicher wird. Wird nun zusätzlich eine Evidenz bei einem der Einflussfaktorknoten gesetzt, so verändert dies die Wahrscheinlichkeitsverteilung des anderen Einflussfaktorknotens, woraufhin das Auftreten dieses Faktors unwahrscheinlicher wird (daher der Name „Explaining Away“), obwohl die beiden Einflussfaktoren zu Beginn bedingt unabhängig waren.

Diese Art des Schlussfolgerns wird für spezielle diagnostische Fragestellungen verwendet.

Beispielsweise kann die Beeinflussung der Preissensibilität durch verschiedene Bewerbungsarten ermittelt werden, indem beim Effektknoten „Absatzmenge“ eine konkrete Evidenz gesetzt wird, anschließend nacheinander alle möglichen Promotionsarten durch entsprechende Evidenzen am Einflussfaktorknoten „Promotionsart“ vorgegeben werden und jeweils die Veränderung der Wahrscheinlichkeitsverteilung des Knotens „Verkaufspreis“ ermittelt wird.

Es können auch **beliebige Kombinationen** (*Combined Reasoning*) der vorgestellten Arten des Schlussfolgerns durchgeführt werden, indem beliebige Mengen aus Effekt- und/oder Auslöserknoten in der Evidenz- bzw. Anfragedimension definiert werden. Das bedeutet, dass die Variablen eines Netzes nicht immer eindeutig als reine Effekt- oder Einflussknoten bezeichnet werden können, da jeder Knoten je nach Fragestellung als Effekt- oder Einflussknoten angesehen werden kann.

Die algorithmische Vorgehensweise zur Simulation der Reaktionen eines Kunden K_i auf ein vorgegebenes Szenario S mit Hilfe der vorgestellten unterschiedlichen Schlussfolgerungsarten besitzt in allen Fällen die Komplexität $O(|ST|\cdot n\cdot Z^n)$ und lautet in Pseudocode wie folgt:

```
SimulateScenario(Ki, S) // implies that Ki.BNet = Ki.LearnBehaviourNet(S)
    Foreach(BTj in S.ST) do
        Foreach(Ek in S.EvidenceDimension[BTj]) do
            Ki.BNet.Nodes[Ek].SetEvidence(Ek.Value)
        Ki.BNet.Propagate()
        Foreach(Qk in S.QueryDimension) do
            S.Result[BTj].Add(Ki.BNet.Nodes[Qk].ExpectationValue)
Return S
```

Die in der Evidenzdimension eines Simulationsszenarios vorgegebenen Evidenzen bestimmen in der Regel die Klasse der Inferenz-Algorithmen, die für entsprechende Schlussfolgerungsprozesse verwendet werden können. Ich unterscheide dabei zwischen folgenden Szenariotypen und deren Kombinationen:

- Szenarien mit bekannten Evidenzen bzw. Evidenzkombinationen,
- Szenarien mit approximierbaren Evidenzen bzw. Evidenzkombinationen und
- Szenarien mit unbekannten Evidenzen bzw. Evidenzkombinationen.

Im Folgenden beschreibe ich die drei unterschiedlichen Szenariotypen genauer und erläutere, mit welchen Evidenzarten und Inferenz-Algorithmen ich im jeweiligen Fall szenariobasierte Simulationen durchführe.

4.2.1 Simulation mit bekannten Evidenzen und Evidenzkombinationen

Falls die Werte der Evidenzvariablen ausschließlich aus Zuständen bestehen, die in der Lernmenge des probabilistischen Verhaltensnetzes vorkommen, wird die Simulation des Szenarios mit Hilfe **harter Evidenzen** (*Hard Evidences* oder auch *Specific Evidences*) durchgeführt. Beim Setzen einer harten Evidenz wird einer Zufallsvariablen X_i ein konkreter Zustand x_j zugewiesen, der in der Menge der zulässigen Knotenzustände von X_i enthalten sein muss. Das bedeutet konkret, dass die Wahrscheinlichkeit $P(X_i=x_j)$ auf 100 % gesetzt wird. Anschließend wird die Auswirkung der gesetzten Evidenz durch einen Inferenz-Algorithmus für harte Evidenzen auf das gesamte Netz und dabei vor allem auf die Effekt- bzw. Anfrageknoten berechnet. So lassen sich beispielsweise die Auswirkungen von konkreten in der Vergangenheit vorgekommenen Verkaufspreisen auf die kundenindividuelle Absatzmenge berechnen. Hierfür werden beim Knoten „Verkaufspreis“ harte Evidenzen auf die gewünschten Einkaufspreise gesetzt und nach der Propagierung der Erwartungswert der Variablen „Absatzmenge“ ermittelt.

4.2.2 Simulation mit approximierbaren Evidenzen und Evidenzkombinationen

Um Szenarien mit Evidenzen zu simulieren, die in der Lernmenge des Netzes nicht enthalten sind, können in vielen Fällen **weiche Evidenzen** (*Soft Evidences*) [VKV02] verwendet werden. Weiche Evidenzen geben eine Wahrscheinlichkeitsverteilung mit den Einzelwahrscheinlichkeiten p_1, …, p_z über die Menge der Zustände x_1, ..., x_z einer Evidenzvariablen X_i vor. Um nun einen Wert w_i als Evidenz zu verwenden, der nicht in den historischen Werten der Variablen X_i vorgekommen ist, wird eine weiche Evidenz durch eine entsprechende Wahrscheinlichkeitsverteilung über die bekannten Knotenzustände x_i berechnet, deren Erwartungswert dem Wert w_i entspricht. Das heißt, es wird eine Wahrscheinlichkeitsverteilung gesucht, für die gilt:

$$E(X_i) = \sum_{i=1}^{z} x_i \cdot p_i = w_i$$

Dies ist nur für Werte w_i möglich, mit $x_{min} \leq w_i \leq x_{max}$, wobei x_{min} das Minimum und x_{max} das Maximum der historischen Werte bzw. der Kontenzustände der Variablen X_i repräsentieren und mindestens zwei unterschiedliche Werte vorliegen.

Mit Hilfe weicher Evidenzen simuliere ich beispielsweise die Auswirkungen von neuen Artikelpreisen, die zwischen zwei historischen Verkaufspreisen liegen, auf den entsprechenden kundenindividuellen Absatz.

Weiche Evidenzen werden mit speziellen Inferenz-Algorithmen propagiert, wobei die Inferenz je nach Evidenzkombination sowie Art und Struktur des zugrunde liegenden Netzes

nur mit gewissen Einschränkungen durchgeführt werden kann (siehe Kapitel 4.3).

Eine andere Möglichkeit, um unbekannte Evidenzwerte zu approximieren oder um Unsicherheit bezüglich einer Wahrscheinlichkeitsverteilung zu repräsentieren, bieten so genannte **virtuelle Evidenzen** (*Virtual Evidences* oder *Likelihood Evidences*). Eine virtuelle Evidenz besteht analog zu einer weichen Evidenz aus der Angabe einer Wahrscheinlichkeitsverteilung über die Zustände einer Evidenzvariablen. Die Verteilung gibt im Falle virtueller Evidenzen eine Art Gewichtung der eigentlichen Zustandswahrscheinlichkeitsverteilung an, um eventuelle Unsicherheiten über den konkreten Zustand der Zufallsvariable widerzuspiegeln.

Bei Inferenz-Algorithmen für virtuelle Evidenzen kann es vorkommen, dass die durch eine virtuelle Evidenz vorgegebene Verteilung eines Evidenzknotens während der Berechnung verändert werden muss und stattdessen eine Verteilung gewählt wird, die der angegebenen möglichst ähnlich ist. Mit Hilfe virtueller Evidenzen kann beispielsweise die Unsicherheit bezüglich zukünftiger Preisstrategien und Promotionsmaßnahmen der Konkurrenz eines Unternehmens ausgedrückt werden.

4.2.3 Simulation mit unbekannten Evidenzen und Evidenzkombinationen

Unter gewissen Umständen können mit Hilfe der probabilistischen Verhaltensnetze unbekannte Evidenzwerte und Evidenzkombinationen simuliert werden, wobei ich die zwei folgenden unterschiedlichen Situationen unterscheide:

- Unbekannte Kombination von harten Evidenzen und
- Setzen von Werten, die niedriger bzw. höher sind als die bisher bekannten Zustandswerte.

Das Setzen einer bisher unbekannten Kombination von harten Evidenzen kann in einigen Fällen durch eine Verallgemeinerung des Netzes möglich gemacht werden. Unter Verallgemeinerung verstehe ich das Gegenteil von Overfitting, indem beim Lernen des Netzes durch bestimmte Maßnahmen (z. B. einem Strafterm im MDL-Gewicht oder Initialisierung aller Zustände mit einer Wahrscheinlichkeitsverteilung) verhindert wird, dass das Netz die Lernfälle zu exakt modelliert. Auf diese Weise können seltene oder nie vorgekommene Zustände einer Variablen geringe Wahrscheinlichkeit erhalten und zum Setzen harter Evidenzen verwendet werden.

Alternativ kann das Ergebnis einer unbekannten Evidenzkombination angenähert werden, indem mehrere Inferenzdurchgänge durchgeführt werden, bei denen jeweils andere Teilevidenzen ausgelassen werden, um eine Propagierung zu ermöglichen. Anschließend ermittelt man die Erwartungswerte der Effektknoten durch geeignete Mittelung der einzelnen Inferenzergebnisse.

Soll ein Wert als Evidenz gesetzt werden, der niedriger oder höher als die bisher bekannten Zustandswerte ist, kann er unter gewissen Umständen mit Hilfe eines neu erzeugten Zustands und einer entsprechenden weichen Evidenz repräsentiert werden, wobei das Wahrscheinlichkeitspotenzial des neuen Zustands mit Hilfe des *Least-Squares-Verfahrens* geschätzt wird, das mindestens zwei bekannte Zustände erfordert. Für eine genauere Beschreibung dieses Verfahrens siehe [SB06].

Unter gewissen Umständen können weitere Möglichkeiten zur Simulation unbekannter Situationen bestehen. Falls sich die Simulation auf das individuelle Verhalten eines Kunden *K* bezüglich eines konkreten Artikels *A* bezieht, wobei in diesem Fall keine oder zu wenige Lernfälle vorliegen, um eine gewünschte Situation durch Setzen von Evidenzen und anschließender Inferenz durchzuführen, können Lernfälle bezüglich anderer Artikel, die *A* sehr ähnlich sind, zum Erstellen eines entsprechenden Netzes verwendet werden. Auf analoge Weise kann mit dieser Vorgehensweise auch das Verhalten eines Kunden auf neu eingelistete Artikel geschätzt werden. Falls *K* zu einer Kundengruppe gehört, kann in manchen Fällen das durchschnittliche Verhalten der Kundengruppe zur Schätzung des individuellen Verhaltens von *K* verwendet werden, da auf diese Weise mehr Lernfälle vorliegen.

4.3 Inferenz-Algorithmen für probabilistische Verhaltensnetze

Um im Rahmen einer kundenindividuellen Verhaltenssimulation die Auswirkungen vorgegebener Evidenzen auf Anfragevariablen zu berechnen, können je nach Typ der Evidenzen verschiedene Inferenz-Algorithmen angewendet werden. Grundsätzlich lassen sich Inferenz-Algorithmen in exakte und approximative Verfahren unterscheiden, wobei die Berechnung der Nachfolgewahrscheinlichkeiten in allen Fällen ein NP-hartes Problem darstellt. Die Performance der verschiedenen Algorithmen hängt daher direkt von der konkreten Komplexität des Netzes ab, d. h. von der Anzahl der Zufallsvariablen und deren möglichen Zustände, sowie vor allem von der Komplexität der Kantenverbindungen und der Anzahl der ungerichteten Schleifen im Graph [KN04].

Im Folgenden gebe ich einen Überblick über bekannte exakte und approximative Inferenz-Algorithmen für verschiedene Klassen Bayes'scher Netze. In den meisten Fällen unterstützen die Algorithmen in ihrer ursprünglichen Version ausschließlich harte Evidenzen. Daher beschreibe ich anschließend, auf welche Weise die vorgestellten Verfahren um virtuelle und weiche Evidenzen erweitert werden können.

4.3.1 Exakte Inferenz-Algorithmen

Zur exakten Inferenz können je nach Klasse und Struktur des probabilistischen Netzes unterschiedliche Verfahren angewendet werden. Die Netze werden dabei in folgende drei

Komplexitätsklassen unterteilt:

- Ketten von Zufallsvariablen,
- Mehrfachbäume (Polytrees) und
- mehrfach verbundene Netze.

Im Falle einer **einfachen Kette von Zufallsvariablen** der Form $X_1 \rightarrow X_2 \rightarrow \ldots \rightarrow X_n$ kann die Propagierung einer harten Evidenz durch einfaches Ablesen der entsprechenden Werte in den bedingten Wahrscheinlichkeitstabellen bzw. durch mehrfache Anwendung des Bayes'schen Theorems (siehe Kapitel 2.3.1) in Verbindung mit der so genannten *Kettenregel* durchgeführt werden.

Falls die Struktur des probabilistischen Netzes einem **Polytree** entspricht, d. h. einem azyklischen Graph, bei dem zwei beliebige Knoten durch maximal einen Pfad miteinander verbunden sind, können so genannte *Message-Passing-Algorithmen* [KP83] angewendet werden, die die Propagierung von harten Evidenzen in linearer Zeit ermöglichen. Message-Passing-Algorithmen berechnen die Auswirkung einer Menge von Evidenzen an Evidenzknoten E auf eine Anfragevariable X, wobei die Menge der Evidenzknoten in folgende Untergruppen eingeteilt wird:

- in die Menge E^+ der Evidenzknoten, die mit X über einen Elternknoten von X verbunden sind (so genannter *Predictive Support*) und
- in die Menge E^- der Evidenzknoten, die mit X über einen Kinderknoten von X verbunden sind (so genannter *Diagnostic Support*).

Während eines Message-Passing-Algorithmus wird erst der Einfluss von E^+ rekursiv auf die Eltern von X und anschließend auf X selbst mit Hilfe des Bayes'schen Theorems berechnet, um anschließend in einem zweiten Schritt die Auswirkungen von E^- auf die Kinder von X und anschließend auf X selbst wiederum rekursiv zu ermitteln, wobei zum Austausch der Einflüsse zwischen den beteiligten Knoten entsprechende Nachrichten (Messages) ausgetauscht werden. Dabei enden die jeweiligen Rekursionen entweder in den Evidenzknoten selbst oder in Blatt- oder Wurzelknoten. Dabei wird jede Zufallsvariable höchstens einmal besucht. Das Verfahren ist somit linear bezüglich der Anzahl der Knoten bzw. der Größe der entsprechenden bedingten Wahrscheinlichkeitstabellen, die durch Einführung einer maximalen Tabellengröße während des Lernens konstant gehalten werden kann [RN03]. Detaillierte Beschreibungen der Message-Passing-Algorithmen bieten unter anderem [NRE03], [PJ88] oder [RN95].

Bei **mehrfach verbundenen Netzen** funktionieren klassische Message-Passing-Algorithmen nicht, da es zwischen zwei beliebigen Knoten mehr als eine ungerichtete Verbindung geben kann. Auf diese Weise würden Evidenznachrichten mehrmals verschickt. Eine Lösung des Problems bieten die so genannten *Clustering-Inferenz-Algorithmen*, bei

denen ein mehrfach verbundenes Netz in einen Polytree mit derselben Wahrscheinlichkeitsverteilung umgewandelt wird. Dazu werden Knoten, zwischen denen ein ungerichteter Zyklus existiert, zu einem einzigen *Metaknoten* (einer so genannten *Clique*) zusammengefasst. Anschließend können auf dem modifizierten Netz Inferenz-Algorithmen für Polytrees angewandt werden. Einer der bekanntesten Clustering-Inferenz-Algorithmen ist der so genannte *Junction-Tree-Algorithmus* [JFV96][JFV01], bei dem ein mehrfach verbundenes Netz durch einen Umformungsprozess mit folgenden Prozessschritten in einen Polytree (*Junction Tree*) umgewandelt wird:

1. Moralisierung
2. Triangulierung
3. Finden von maximalen Cliquen
4. Erstellung des Junction Trees und der so genannten Separatoren
5. Initialisierung

Der dem probabilistischen Netz zugrunde liegende gerichtete azyklische Graph (DAG) mit den Zufallsvariablen $X_1,\ldots,X_n$ kann in einen **moralisierten Graphen** überführt werden, indem man alle vorhandenen gerichteten Kanten in ungerichtete umwandelt und jeweils alle Eltern $Parents(X_i)$ eines jeden Knotens X_i durch ungerichtete Kanten verbindet.

Ein **triangulierter Graph** kann aus einem moralisierten Graphen erzeugt werden, indem jeder Zyklus von mehr als drei Knoten eine Kante erhält, die zwei vorher nicht direkt miteinander verbundene Knoten des Zyklus miteinander verbindet. Die **Cliquen** sind anschließend die *maximal verbundenen Teilgraphen* des triangulierten Graphen. Maximal verbundene Teilgraphen setzen sich aus Knoten zusammen, die alle paarweise miteinander verbunden sein müssen und kein weiterer Knoten hinzugefügt werden kann, ohne dass diese Regel gebrochen wird. Ziel ist es hierbei, eine gute Triangulierung zu finden, die die Größe der zu den Cliquen gehörenden bedingten Wahrscheinlichkeitstabellen möglichst gering hält. Die Berechnung einer optimalen Triangulierung ist im Allgemeinen ein NP-hartes Problem [JJ94]. Aus diesem Grund müssen heuristische Verfahren zur Triangulierung und Findung idealer Cliquen angewendet werden, wie zum Beispiel der so genannte *Eliminierungs-Algorithmus* [JFV96].

Nach Moralisierung, Triangulierung und Bestimmung geeigneter Cliquen wird der **Junction Tree** erstellt, dessen Knoten aus den gefundenen Cliquen bestehen und dessen Kanten durch so genannte **Separatoren** repräsentiert werden. Ein Separator besteht aus der Schnittmenge der Knoten der beiden verbundenen Cliquen. Dabei gilt es, die so genannte *Running-Intersection-Eigenschaft* zu bewahren, d. h. dass ein Knoten, der in zwei Cliquen C_1 und C_2 vorkommt, auch in jeder Clique und somit in den Separatoren auf dem Pfad zwischen C_1 und C_2 vorkommen muss, was nur in einem triangulierten Graph möglich ist.

Nach Erstellung des Junction Trees folgt die dreistufige **Initialisierung** der

entsprechenden Wahrscheinlichkeitstabellen der Cliquen und Separatoren. In der ersten Phase werden alle Cliquen- und Separatortabellen mit 1 initialisiert, um anschließend im zweiten Schritt für jeden Knoten X_i dessen bedingte Wahrscheinlichkeitstabelle in eine Clique hinein zu multiplizieren, die sowohl den Knoten X_i selbst als auch seine Elternknoten *Parents*(X_i) beinhaltet. Anschließend werden im letzten Schritt die durch die Einmultiplizierung der bedingten Wahrscheinlichkeiten entstandenen *Evidenzpotenziale* durch einen zweistufigen Propagierungsprozess (bestehend aus einer *Collect*- und *Distributephase*) im Junction Tree verteilt. Auf diese Weise werden sämtliche Wahrscheinlichkeitstabellen der Cliquen und Separatoren initialisiert.

Nach der Durchführung des mehrstufigen Umformungsprozesses, der insgesamt eine NP-harte Komplexität besitzt [RN03], können harte Evidenzen anschließend mit Hilfe eines an die Junction-Tree-Struktur angepassten Message-Passing-Algorithmus für Polytrees in linearer Zeit propagiert werden. Im Rahmen des SimMarket Projektes wird bei der Implementierung des Junction-Tree-Algorithmus auf statische Separatoren verzichtet. Stattdessen werden die entsprechenden Messages dynamisch berechnet und direkt zwischen den Cliquen ausgetauscht (siehe [KA05] und [SB06]).

Weitere exakte Inferenzverfahren zur Ermittlung der Nachfolgewahrscheinlichkeit einer einzelnen Anfragevariable in probabilistischen Netzen mit Ketten-, Baum und Polytree-Struktur sind die so genannte *Variable Elimination* und deren Variante *Bucket Elimination* für mehrfach verbundene Netze, die in [RN03] detailliert beschrieben werden. Für mehrfach verbundene Netze bietet sich alternativ das *Cutset Conditioning* an, bei dem ein Netz in mehrere einfache Polytrees transformiert wird (siehe [PJ86]).

4.3.2 Approximative Inferenz-Algorithmen

Exakte Inferenz-Algorithmen eignen sich auf Grund ihrer enormen Komplexität [CGF90] in der Praxis je nach Kantenstruktur nur für kleine bis mittelgroße probabilistische Netze. Daher werden bei großen Netzen mit hoher Kantendichte in vielen Fällen approximative Verfahren angewendet, die zwar ebenfalls eine NP-harte Komplexität besitzen [DL93], aber in der Praxis oft geringere Laufzeiten aufweisen.

Einer der bekanntesten Ansätze für approximative Inferenz-Algorithmen ist die *stochastische Simulation*, bei der eine große Anzahl von Testfällen bezüglich einer vorgegebenen Evidenzkombination generiert werden, mit deren Hilfe eine Annäherung für die Wahrscheinlichkeitsverteilung einer oder mehrerer Anfragevariablen möglich ist. Nach dem Gesetz der Großen Zahl gilt hierbei: je mehr Fälle generiert werden, desto exakter wird die Approximation. Im Folgenden beschreibe ich drei bekannte Verfahren der stochastischen Simulation: Rejection Sampling (Logic Sampling), Likelihood Weighting und Markov Chain Monte Carlo (MCMC).

Das **Rejection Sampling** stammt ursprünglich aus der Statistik und wurde 1988 erstmals von Max Henrion als **Logic Sampling** an Bayes'sche Netze angepasst [HM88]. Die Idee besteht in der zufälligen Generierung einer großen Zahl von Samples. Dazu werden die Variablen topologisch sortiert. Der konkrete Samplewert einer Variablen wird dann, jeweils in Abhängigkeit der zufälligen Belegung seiner Elternknoten, zufällig aus den Werten seiner CPT ermittelt. Anschließend werden alle Samples, die nicht der definierten Evidenzkombination entsprechen, aussortiert (Rejection). Mit Hilfe der übrigen Samples kann nun die Wahrscheinlichkeitsverteilung der Anfragevariable(n) approximiert werden, indem man für jeden ihrer Zustände die Anzahl der positiven Samples durch die Gesamtanzahl der Samples dividiert und anschließend eine Normalisierung der Werte durchführt.

Der Nachteil dieses Verfahrens liegt in der oft enormen Anzahl aussortierter und somit unnötig erzeugter Samples, vor allem wenn die definierte Evidenzkombination eine sehr geringe Wahrscheinlichkeit besitzt. Konkret wächst die Anzahl der zurückgewiesenen Samples exponentiell mit der Anzahl der Evidenzvariablen.

Ein Ansatz, der die Erzeugung unrelevanter Samples ausschließt und somit deutlich effizienter ist, ist das **Likelihood Weighting** [FC89][SP89]. Die Vorgehensweise besteht darin, die Zustände der Evidenzvariablen mit den Evidenzwerten zu fixieren, um nur noch Zufallswerte für die übrigen Nichtevidenz- und Anfragevariablen zu erzeugen. So ist garantiert, dass alle resultierenden Samples zur vorgegebenen Evidenzkombination passen und zur Berechnung der Nachfolgewahrscheinlichkeitsverteilung der Anfragevariablen verwendet werden können. Allerdings müssen die erzeugten Samples zum Ausgleich mit ihrer Auftrittswahrscheinlichkeit gewichtet werden, die aus dem Produkt der bedingten Wahrscheinlichkeiten der konkreten Zustände der Evidenzvariablen bezüglich der Elternknotenzustandskombinationen besteht, d. h. die entsprechenden Werte können einfach aus den bedingten Wahrscheinlichkeitstabellen der Evidenzvariablen ermittelt werden. Obwohl das Likelihood Weighting deutlich effizienter als das Logic Sampling ist, wächst die Laufzeit dennoch sehr stark mit der Anzahl der Evidenzvariablen. Ein weiteres Problem entsteht, wenn die Evidenzvariablen in der Netztopologie sehr weit unten stehen, da die Generierung der Samples und deren Gewichtung in diesem Fall immer aufwändiger und gleichzeitig ungenauer werden.

Während Logic Sampling und Likelihood Weighting die Samples völlig unabhängig voneinander erzeugen, wird im Rahmen des **Markov Chain Monte Carlo** (MCMC) Verfahrens ein neues Sample durch zufällige Veränderung des vorherigen Samples erzeugt. Der Algorithmus startet mit einem zufällig erzeugten Sample, wobei nur die Nichtevidenzvariablen zufällig belegt werden, während die Evidenzvariablen auf die konkreten Evidenzwerte festgelegt sind. Anschließend werden neue Samples durch zufällige

Veränderung der Zustände der Nichtevidenzvariablen generiert, wobei der neue Wert eines Nichtevidenzknotens unter Berücksichtigung seiner bedingten Abhängigkeiten innerhalb seiner Markov-Decke (siehe Kapitel 2.3.3) generiert wird. Das Verfahren läuft also zufällig durch den möglichen Zustandsraum, wobei in jedem Durchgang der Wert einer Nichtevidenzvariablen verändert wird. Die Nachfolgewahrscheinlichkeitsverteilung der Anfragevariablen ergibt sich anschließend aus der Normalisierung der Vorkommen der entsprechenden positiven Samples, d. h. die Wahrscheinlichkeit eines konkreten Zustands einer Anfragevariable ergibt sich aus der Anzahl der Samples, in denen die Variablen den entsprechenden Wert besitzen, dividiert durch die Gesamtanzahl der Samples. Der MCMC geht auf den *Metropolis Algorithmus* [MRRTT53] zurück und wurde erstmals von Pearl auf Bayes'sche Netze angewendet [PJ87]. Zwei bekannte Varianten des MCMC sind der so genannte *Gibbs Sampler* (siehe [RN03] für eine detaillierte Beschreibung) sowie der *Metropolis-Hastings Sampler*.

Weitere approximative Inferenz-Algorithmen sind *State Space Abstraction* [WL94], *Localized Partial Evaluation* [DD95], *Weak Arc Removal* [KU94], *Mutual Information Guided Approximate Evaluation* [JN00] und *Loopy Propagation* [PJ88].

4.3.3 Unterstützung von virtuellen und weichen Evidenzen

Die in den Kapiteln 4.3.1 und 4.3.2 vorgestellten exakten und approximativen Inferenz-Algorithmen unterstützen in ihren ursprünglichen Versionen meist nur die Propagierung ausschließlich harter Evidenzen. Eine Möglichkeit, bei einem Knoten X_i virtuelle Evidenzen setzen zu können, ist die Einführung eines so genannten *virtuellen Elternknotens* [KN04], der die Unsicherheit bezüglich der Wahrscheinlichkeitsverteilung von X_i repräsentiert. Diese Vorgehensweise lässt sich sowohl bei exakten Algorithmen als auch bei approximativen Verfahren anwenden. Beim approximativen Likelihood Weighting lässt sich die virtuelle Wahrscheinlichkeitsverteilung sogar direkt berücksichtigen [KN04].

Einzelne weiche Evidenzen können bei mehrfach verbundenen Netzen durch eine Erweiterung des Junction-Tree-Algorithmus propagiert werden (siehe [VKV02] für eine detaillierte Beschreibung). Sollen mehrere weiche Evidenzen gleichzeitig möglich sein, so können entweder die *IPFP* (*Iterative Proportional Fitting Procedure*) oder deren effizientere Variante in Form des *Big-Clique-Algorithmus* [KVV04] angewendet werden.

Die im Rahmen des Projektes SimMarket implementierten exakten und approximativen Algorithmen für harte und weiche Evidenzen werden in [SB06] und [KA05] im Detail beschrieben.

5 Modellierung, Simulation und Bestimmung von Kundengruppen

In den Kapiteln 3 und 4 wurde erläutert, wie das Kaufverhalten einzelner Kunden aus historischen Daten extrahiert und mit Hilfe probabilistischer Verhaltensnetze gespeichert, analysiert und simuliert werden kann. Das vorgestellte Kundenmodell unterstützt in erster Linie die Entscheidungsfindung im Rahmen kundenindividueller Marketingstrategien, wie sie beispielsweise im „*One to One Marketing*“ [ZJ00] entworfen werden.

Im klassischen Einzelhandel werden Sortimente und Maßnahmen in der Regel nicht auf einzelne Kunden abgestimmt. Vielmehr werden Entscheidungen entweder auf die Gesamtheit der Kunden oder auf spezielle Zielgruppen zugeschnitten, beispielsweise auf die Gruppe der umsatz- bzw. ertragsstärksten Kunden. Aufgrund eines zunehmenden Wettbewerbs wird es für Einzelhändler allerdings immer entscheidender, ihre Angebote und Dienstleistungen durch stärkere Individualisierung aus der Masse der homogener werdenden Angebotsflut hervorzuheben [ZJ00]. Da sich der Ansatz des „One to One Marketing“ bei vielen Unternehmen (vor allem bei SB-Warenhäusern) aus Gründen der Praktikabilität und anfänglichen Kosten nicht vollständig umsetzen lässt, werden im Einzelhandel Lösungen angestrebt, die nicht auf Individuen, sondern auf Gruppen *gleichartiger* Kunden fokussieren. Im Marketing existiert eine Vielzahl unterschiedlicher Kundengruppentypen und Kundentypologien, die sich vor allem bezüglich der zugrunde liegenden Vergleichsmerkmale unterscheiden. Die meisten dieser Typologien und Typbeschreibungen besitzen allerdings den gemeinsamen Nachteil, dass sie sich beispielsweise nicht zur Entscheidungsfindung bezüglich wirksamer artikelbezogener Preis- und Promotionsmaßnahmen eignen, da ihr Fokus zu sehr auf soziodemographischen und einfachen, statistischen Merkmalen liegt und nicht auf der Betrachtung des dynamischen Kaufverhaltens (siehe Anhang B).

In diesem Kapitel erläutere ich daher die verhaltensbezogene Modellierung von Kundengruppen, die sich zur Analyse und Simulation des aggregierten Kaufverhaltens bezüglich konkreter Produkte und Maßnahmen eignet. Kundengruppen werden dazu mit Hilfe probabilistischer Kundengruppenagenten modelliert, die durch Holonisierung gleichartiger Kundenagenten entstehen (Kapitel 5.1).

In Kapitel 5.2 beschäftige ich mich mit der wichtigen Frage, mit welchen Verfahren Kunden gefunden werden können, die sich bezüglich ihres dynamischen Kaufverhaltens ähnlich sind.

In den folgenden Kapiteln 5.3 bis 5.5 stelle ich dazu die im Rahmen meiner Arbeit entwickelten Verfahren zur Ähnlichkeitsanalyse des kundenindividuellen, dynamischen

Kaufverhaltens vor. Kapitel 5.3 präsentiert entsprechende Verfahren, die auf verhaltensbeschreibenden Attributvektoren basieren. In Kapitel 5.4 erläutere ich zwei Vergleichsverfahren für Verhaltenssimulationsergebnisse. In Kapitel 5.5 stelle ich fünf Methoden zum direkten Vergleich von probabilistischen Verhaltensnetzen vor.

Sämtliche entwickelten Verfahren eignen sich als Grundlage der verhaltensbezogenen Kundengruppenfindung. In den Kapitel 5.6 und 5.7 erweitere ich bekannte Klassifikations- und Clusteringverfahren um verhaltensbezogene Ähnlichkeitsmaße.

In Kapitel 5.8 erläutere ich abschließend, wie auf Basis von Kunden- und Kassenbondaten eindeutige, verhaltensbezogene Kundensegmentierungen in hinreichender Qualität ermittelt werden können.

5.1 Modellierung und Simulation von Kundengruppen

Im Einzelhandel beziehen sich viele Strategien und Maßnahmen auf *Zielgruppen* mit ausgesuchten Eigenschaften. Zielgruppen können je nach Fragestellung und Datengrundlage anhand verschiedenster Kriterien definiert werden. In der Regel beziehen sich entsprechende Kriterien auf soziodemographische, geographische oder einfache verhaltensbezogene Kundenmerkmale oder deren Kombination. Je nach Auswahl dieser Merkmale können einzelne Kunden gleichzeitig mehreren Kundengruppen zugeordnet werden.

Kundengruppen modelliere ich durch *holonische Kundengruppenagenten*, die entweder aus einer Menge einzelner Kundenagenten bestehen oder anhand kundenbezogener Kriterien generiert werden, ohne die Mitglieder vorher explizit als einzelne Kundenagenten modellieren zu müssen. Im letzteren Fall kann ein Kundengruppenagent analog zu individuellen Kundenagenten modelliert und zur Simulation verwendet werden, da die aggregierten Kundengruppendaten wie Informationen über eine einzelne Person behandelt werden können. Im Normalfall bilde ich Kundengruppenagenten aus einer Menge einzelner Kundenagenten, die durch Holonisierung zusammengefasst werden. Als Holonenform verwende ich dabei hauptsächlich die kopfgesteuerte Agentengesellschaft sowie die Verschmelzung durch partielles oder vollständiges Klonen. Kundenagenten können auf diese Weise zur gleichen Zeit Mitglieder verschiedener Kundengruppenagenten sein.

Im Falle einer kopfgesteuerten Kundenagentengesellschaft erzeuge ich einen neuen Kundengruppenagenten, der die Agentengesellschaft als Kopf des Holons nach außen repräsentiert (siehe Abbildung 43). Ihm sind alle einzelnen Kundenagenten der Kundengruppe bekannt, die mehr oder weniger autonom bleiben. Das Gruppenmodell bzw. das Gruppenverhalten wird je nach Aufgabenstellung entweder durch Durchschnittsbildung oder durch „Addition“ der Einzelmodelle bzw. individuellen Verhaltensweisen gebildet (siehe Kapitel 2.4.2.4).

Je nach Problemstellung oder Anwendung kann es sehr aufwändig sein, alle Kundenagenten einer Kundengruppe einzeln zu befragen, um die individuellen Ergebnisse anschließend zu aggregieren. Aus diesem Grund ist es in der Praxis oft effizienter, aufgabenbezogene, aggregierte Verhaltensmodelle temporär im Kundengruppenagenten zu speichern.

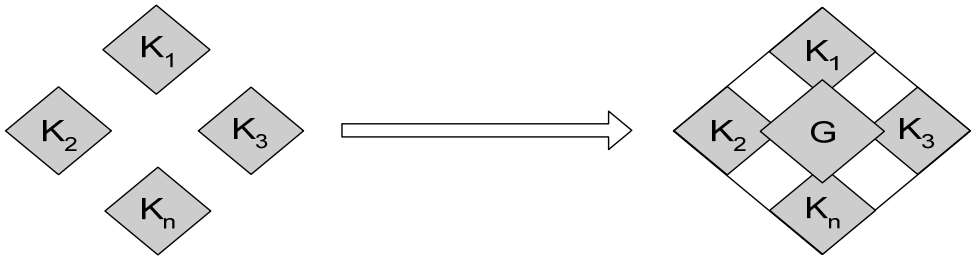


Abbildung 43: Kundengruppenbildung durch Holonisierung einzelner Kundenagenten

Die aggregierten Modelle können, neben der oben beschriebenen direkten Methode, durch partielles bzw. vollständiges Klonen der einzelnen Kundenagenten und anschließender Additions- oder Durchschnittsverschmelzung generiert werden.

Im Falle der Additionsverschmelzung steht das aggregierte Verhalten der Gruppenmitglieder im Vordergrund, beispielsweise um die absoluten Auswirkungen einer bestimmten Werbemaßnahme auf den kundengruppenbezogenen Absatz zu simulieren, während der Kundengruppenagent bei der Durchschnittsverschmelzung das durchschnittliche Verhalten seiner Mitglieder repräsentiert.

Eine Relationsverschmelzung von einzelnen Kundenagenten zur probabilistischen Modellierung von gesellschaftlichen Abhängigkeiten, wie beispielsweise Mund-zu-Mund-Propaganda, scheitert in der Praxis an der Komplexität: je nach Komplexität der individuellen Verhaltensnetze können maximal zehn bis einhundert verschiedene Kundenagenten durch Relationsverschmelzung vereint werden. Da sich die Modellierung der Kunden in meiner Arbeit lediglich auf das messbare Verhalten von Kunden beschränkt und nicht auf die Modellierung von gesellschaftlichen Einflüssen, verweise ich auf entsprechende Literatur bezüglich *agentenbasierter Gesellschaftssimulationen* (siehe beispielsweise [DP02] für eine Einführung in *agent-based social simulation*).

Die Simulation des Kaufverhaltens von Kundengruppen verläuft analog zu den in Kapitel 4 vorgestellten szenariobasierten Simulationsverfahren für individuelle, probabilistische Kundenagenten. Je nach Holonenform kann die Simulation entweder mit Hilfe des aggregierten kundengruppenbezogenen Verhaltensnetzes oder durch Einzelsimulation aller Gruppenmitglieder und anschließender Aggregation durchgeführt werden.

5.2 Bildung von Kundengruppen und Kundentypologien

Kunden können hinsichtlich unterschiedlicher Merkmale in Gruppen eingeteilt werden. Eine konkrete Einteilung entspricht dabei einer so genannten *Kundentypologie*. Eine Kundentypologie ist ein Schema, das eine Menge von Kundengruppen mit Hilfe von Merkmalen beschreibt, von denen nicht bekannt sein muss, ob sie zur *vollständigen* Klassifikation oder Segmentierung der Kundengesamtheit geeignet sind. Typologien können nach [WWHTM01] anhand der zugrunde liegenden Merkmalsarten gegliedert werden:

- Geographische Typologien
- Soziodemographische Typologien
- Verhaltensbezogene Typologien
- Psychographische und lifestylebezogene Typologien
- Mischtypologien

Geographische Typologien fokussieren geographische Kriterien, beispielsweise Wohnort oder Postleitzahlenbereich, während bei soziodemographischen Typologien Merkmale wie Alter, Geschlecht oder Einkommen der Kunden im Mittelpunkt stehen. Verhaltensbezogene Typologien betrachten in der Regel einfache, statistische Kennzahlen über das kundenindividuelle Konsumverhalten. Psychographische und lifestylebezogene Typologien unterscheiden Kunden aufgrund ihrer Lebensweisen. Darüber hinaus gibt es eine Vielzahl von Mischtypologien, die eine Kombination unterschiedlicher Merkmalsarten berücksichtigen (in Anhang B stelle ich zwei bekannte europäische Kundentypologien vor).

Die Grundlage der Kundengruppenbildung ist die Entdeckung gleichartiger Kunden, bezogen auf die in der zugrunde liegenden Typologie definierten Merkmale. Voraussetzung ist die Ermittlung der Ähnlichkeit zweier Kunden mit Hilfe geeigneter Vergleichs- bzw. Abstandsmaße.

Üblicherweise werden dazu im Marketing *vektorisierte Kundenbeschreibungen* (so genannte *Attributvektoren*) verwendet, deren Einträge die konkreten kundenindividuellen Ausprägungen einer Menge ausgewählter Merkmale beinhalten und die mit Hilfe entsprechender vektorbasierter Distanzmaße verglichen werden können. In der Praxis werden dabei zur Bildung der Attributvektoren meist demographische und einfache verhaltensbezogene Merkmale verwendet. Da auf diese Weise das kundenindividuelle Kaufverhalten für spezifische verhaltensbezogene Fragestellungen nicht exakt genug repräsentiert wird, habe ich Verfahren und Maße entwickelt, mit deren Hilfe sich kundenbezogene Verhaltensnetze zur Ähnlichkeitsanalyse verwenden lassen, die das dynamische Kaufverhalten wesentlich exakter modellieren als klassische Attributvektoren.

Die im Rahmen dieser Arbeit entwickelten Vergleichsverfahren und Ähnlichkeitsmaße basieren entweder auf dem Vergleich verhaltensbezogener Simulationsergebnisse (Kapitel 5.4) oder dem direkten Vergleich probabilistischer Verhaltensnetze (Kapitel 5.5). Mit diesen Verfahren ist es möglich, Kunden anhand sehr dynamischer Verhaltensaspekte zu vergleichen, wie beispielsweise aufgrund ihrer Preis- oder Werbesensibilität bezogen auf vorgegebene Warengruppen oder anhand ihrer Reaktionen auf komplexe Einkaufsszenarien.

Aufbauend auf vektor- bzw. verhaltensnetzbasierten Ähnlichkeitsanalysen kann eine Vielzahl verschiedener Verfahren zur Entdeckung von Gruppen gleichartiger Kunden durchgeführt werden, wie beispielsweise verschiedene Varianten der *Kundenklassifikation* oder des *Kundenclusterings*.

Bei der Kundenklassifikation ist bereits eine Kundentypologie bzw. eine Menge von Kundengruppen bekannt, die jeweils durch einen so genannten *Prototypen* mit gruppentypischen Merkmalsausprägungen repräsentiert werden. Um neue bzw. unklassifizierte Kunden in die schon bekannten Gruppen einzuordnen, werden diese mit sämtlichen Prototypen verglichen und anschließend der Kundengruppe zugeordnet, zu der die größte Ähnlichkeit besteht. Falls eine ausreichende Menge bereits klassifizierter Kunden zur Verfügung steht, können *Klassifikationsmodelle* (*Classifier*) gelernt werden, die effizient zur Klassifikation unbekannter Kunden verwendet werden können. Klassifikationsmodelle basieren häufig auf neuronalen Netzen, Naive-Bayes-Modellen oder vollständigen Bayes'schen Netzen (siehe Kapitel 5.6.1).

Beim Kundenclustering liegt eine unklassifizierte Kundenmenge vor und es sind keine Kundengruppen bekannt. Die Aufgabe besteht darin, eine geeignete Kundentypologie bzw. geeignete Kundengruppen (*Cluster*) und deren markante Merkmalsausprägungen zu ermitteln, in die die unklassifizierte Kundenmenge so unterteilt werden kann, dass sich Kunden innerhalb eines Clusters möglichst ähnlich und sich Cluster untereinander möglichst unähnlich sind. Eine Vielzahl von Verfahren basiert dabei auf einem iterativen Prozess, wobei einzelne Cluster durch so genannte *Means* oder *Medoids* repräsentiert werden. Dies sind Prototypen, deren Merkmalsausprägungen den Durchschnitt der Clustermitglieder repräsentieren. Sie können entweder berechnete, virtuelle oder konkrete Kundeninstanzen sein. Während der Iteration werden die unklassifizierten Kunden mit sämtlichen Prototypen verglichen und anschließend in den Cluster eingeordnet, zu dessen Prototypen sie am ähnlichsten sind. Daraufhin werden die Means bzw. Medoids der Cluster neu berechnet, um anschließend wiederum alle Kunden mit den Prototypen zu vergleichen. Die Iteration endet, wenn sich keine nennenswerten Mitgliedschaftsveränderungen ergeben oder der Wert eines vorher definierten Gütemaßes einen bestimmten Schwellenwert erreicht hat.

5.3 Kundenähnlichkeitsanalyse mit Attributvektoren

Im Folgenden erläutere ich die Repräsentation einzelner Kunden durch Attributvektoren, die als Grundlage für effiziente kundenbezogene Ähnlichkeitsanalysen dienen.

Ein Attributvektor $AV = (a_1, \ldots, a_v)$ der Länge v eines Kunden beschreibt die konkreten Ausprägungen a_i des Kunden bezüglich der Attribute bzw. Merkmale A_i. Die einzelnen Attribute beschreiben dabei in der Regel soziodemographische, geographische, psychographische, lifestylebezogene oder verhaltensbezogene Merkmale des Kunden.

Aus mathematischer Sicht können die Attribute, je nach Art der zugrunde liegenden Werte, in metrische und nicht-metrische Variablen unterteilt werden (siehe Anhang A.6). Metrische Attribute können in *Maß-* und *Verhältniszahlen* unterteilt werden. Beispiele sind arithmetische Werte wie Mittelwert, Median, Varianz, Summe sowie Prozent- und Indexzahlen. Metrische Attribute können darüber hinaus *normiert* sein. Das bedeutet, dass sie auf einen bestimmten Wertebereich beschränkt sind, beispielsweise auf reelle Werte zwischen null und eins bzw. zwischen null und hundert Prozent.

Ich gliedere kundenbezogene Attribute darüber hinaus in die in Kapitel 3.2.1 vorgestellten statischen, einfach dynamischen und komplexen, verhaltensbezogenen Merkmalsarten, die ich im Folgenden unter dem Begriff *Kundenprofil* zusammenfasse. Bei den statischen Merkmalen handelt es sich hauptsächlich um soziodemographische und geographische Informationen, die im Laufe der Zeit konstant bleiben, oder deren Veränderungen in der zugrunde liegenden Datenbank nicht gepflegt werden. Einfache dynamische Attribute sind in der Regel Kennzahlen bzw. statistische Werte, die aus dem kundenbezogenen Faktenwissen generiert werden und sich über die Zeit ändern können. Es handelt sich dabei in den meisten Fällen um einfache verhaltensbezogene Merkmale. Komplexe verhaltensbezogene Merkmale können nur mit aufwändigen Verfahren oder mit Hilfe der probabilistischen Verhaltensnetze ermittelt werden.

Je nach Fragestellung können für jeden Kunden entsprechende Attributvektoren erzeugt werden, wobei die konkreten Merkmalsausprägungen eines Kunden entweder im Faktenwissen oder in der zugrunde liegenden Datenbank zur Verfügung stehen oder durch geeignete Verfahren bzw. mit Hilfe der kundenindividuellen Verhaltensnetzen ermittelt werden.

Kundengruppen können auf analoge Weise durch Attributvektoren repräsentiert werden, deren entsprechende Merkmalsausprägungen durch geeignete Summen- oder Durchschnittsbildung der kundenindividuellen Attributvektoren erzeugt werden (siehe auch vektorbasierte Repräsentation von Agenten und Holonen in [GA04]).

Kunden bzw. Kundengruppen können verglichen werden, indem die Ähnlichkeit ihrer Attributvektoren mit Hilfe vektorbasierter Distanz- und Ähnlichkeitsmaße berechnet wird.

5.3.1 Vektorvergleich mit Distanz- und Ähnlichkeitsmaßen

Die Ähnlichkeit zweier Attributvektoren der Form $AV = (a_1, \ldots, a_v)$ kann mit Hilfe vektorbasierter Distanz- und Ähnlichkeitsmaße ermittelt werden.

Distanzmaße bestimmen den Abstand bzw. den Unterschied zwischen den beiden Vektoren, während Ähnlichkeitsmaße deren Übereinstimmung berechnen. Sowohl Distanz- als auch Ähnlichkeitsmaße können in *normierte* und *nicht-normierte Maße* untergliedert werden.

Normierte Maße liefern ein normiertes Ergebnis, d. h. einen Wert in einer festgelegten Skala, wie beispielsweise einen reellen Wert im Wertebereich von null bis eins oder null bis hundert Prozent. Normierte Ergebnisse eignen sich für absolute Ähnlichkeitsangaben (wie zum Beispiel: zwei Vektoren sind sich zu 80 Prozent ähnlich), während sich nicht-normierte Ergebnisse nur relativ mit anderen Ergebnissen vergleichen lassen.

Im Falle normierter Maße besteht zwischen Distanz- und Ähnlichkeitsmaßen folgender Zusammenhang:

$$\textit{Ähnlichkeit}\,(X, Y) = 1 - \textit{Distanz}\,(X, Y)$$

Bei der Berechnung der Ähnlichkeit zweier Attributvektoren X und Y werden in der Regel deren *Richtungen* und *Längen* verglichen. Dabei repräsentiert die Richtung *relative* und die Länge *absolute* Merkmalsausprägungen des entsprechenden Kunden. Zwei Kunden sind sich demnach bezüglich der relativen Ausprägungen umso ähnlicher, je ähnlicher die Richtungen der beiden Attributvektorrichtungen sind, d. h. je kleiner der Winkel zwischen den beiden Vektoren ist. Analog unterscheiden sich zwei Kunden in ihren absoluten Merkmalsausprägungen umso weniger, je kleiner die Differenz zwischen den Längen ihrer Attributvektoren ist.

Je nach Fragestellung interessieren bei Kundenvergleichen relative oder absolute Ausprägungen unterschiedlich stark. Daher ist es sinnvoll, je nach Anwendung Maße zu verwenden, die Richtung und Länge der Attributvektoren unterschiedlich stark gewichten. Darüber hinaus ist es in vielen praktischen Anwendungen entscheidend, einzelne Attribute individuell gewichten zu können, beispielsweise, wenn bestimmte Merkmale stärker oder schwächer berücksichtigt werden sollen. Dies lässt sich durch Vorgabe eines Gewichtungsvektors $G = (g_1, \ldots, g_n)$ realisieren, der vor oder während der Ähnlichkeitsanalyse mit den Kundenvektoren multipliziert wird.

Folgende Maße ziehe ich zum Vergleich kundenindividueller Attributvektoren in Betracht:

- den Euklidischen Abstand,
- das Kosinus-Maß,
- die Clark-Distanz,
- die Canberra-Distanz,
- die Bray-Curtis-Distanz sowie
- heterogene Vergleichmaße.

Der **Euklidische Abstand** gehört wie die *Manhattan-Distanz* zur Familie der *Minkowski-Metriken*, ist ein klassisches Distanzmaß und entspricht im mathematischen Sinne der kürzesten Strecke zwischen zwei Punkten in einem *n*-dimensionalen Raum. Er ist formal als Abstand $d(X, Y)$ zwischen zwei *n*-dimensionalen Vektoren $X = (x_1,\ldots,x_n)$ und $Y = (y_1,\ldots,y_n)$ folgendermaßen definiert:

$$Euklid\ d(X,Y) = |\ X - Y\ | = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2}$$

Der Euklidische Abstand eignet sich für den Vergleich von *normierten* Vektoren, d. h. alle Vektordimensionen bzw. Attribute müssen der gleichen Normierung entsprechen, da sich andernfalls Attribute mit höheren zulässigen Zahlenwerten stärker auf den Abstand auswirken und dadurch eine meist ungewollte Gewichtung entsteht. Der Euklidische Abstand eignet sich daher nur in diesen Fällen zur Ähnlichkeitsanalyse von Kundenattributvektoren, wie beispielsweise bei der ausschließlichen Verwendung von Prozentwerten. Bei gemischten Vektoren, also Vektoren mit nicht-normierten oder sowohl metrischen und nicht-metrischen Attributen, kann eine Fehlgewichtung nicht ausgeschlossen werden. Darüber hinaus ist der ursprüngliche Euklidische Abstand kein normiertes Maß und somit nur für relative Vergleiche geeignet. Das bedeutet, dass bei einem Vergleich eines Objektes mit mehreren Testkandidaten der Kandidat ermittelt werden kann, der den geringsten Euklidischen Abstand zum Objekt besitzt. Dabei kann allerdings nicht *absolut* angegeben werden, wie ähnlich oder unähnlich sich Objekt und Testkandidat sind.

Der **Kosinus-Abstand** bzw. die Kosinus-Distanzfunktion ermittelt den Kosinus des Winkels α zwischen zwei Vektoren in einem reellen Vektorraum. Er ist formal als Abstand $d(X, Y)$ zwischen zwei *n*-dimensionalen Vektoren $X = (x_1,\ldots,x_n)$ und $Y = (y_1,\ldots,y_n)$ definiert:

$$Kos\ d(X,Y) = \cos\alpha\ (X,Y) = \frac{X \cdot Y}{|\ X\ | \cdot |\ Y\ |} = \frac{\sum_{i=1}^{n} x_i \cdot y_i}{\sqrt{\sum_{i=1}^{n} {x_i}^2 \cdot \sum_{i=1}^{n} {y_i}^2}}$$

Im Falle von ausschließlich positiven Attributwerten liefert der Kosinus-Abstand Werte im Intervall [0; 1], wobei sich *X* und *Y* umso ähnlicher sind, je größer der Wert ist. In diesem Fall ist der Kosinus-Abstand ein normiertes Distanzmaß und ist somit als absolutes Ähnlichkeitsmaß verwendbar, d. h. es ist möglich auszudrücken, dass sich zwei Objekte (Vektoren) bei einem ermittelten Abstand von 0,9 nach dem Kosinusmaß 90% ähnlich sind.

Allerdings spielen beim Kosinusmaß nur die relativen Ausprägungen der Attribute eine Rolle, nicht ihre absoluten Werte, d. h. die Länge des Vektors wird nicht berücksichtigt. Aus diesem Grund eignet sich das Kosinusmaß nur zum Vergleich relativer Ausprägungen. Beispielsweise werden zwei Kunden als sehr ähnlich angesehen, wenn beide in ihrer Einkaufshistorie genau doppelt so viele Marken- wie No-Name-Produkte gekauft haben, obwohl die konkrete Anzahl unterschiedlicher Produkte und deren Menge bei beiden Kunden stark variieren können.

Da sowohl der Euklidische Abstand als auch das Kosinusmaß die Länge der Vektoren bzw. die Unterschiede der Attributwertebereiche kaum oder gar nicht berücksichtigen und es somit beispielsweise im Falle von Mischvektoren zu Fehlgewichtungen kommen kann, gilt es nach Abstandsmaßen zu suchen, die die Länge der Vektoren in den Vordergrund stellen. Zwei geeignete längenorientierte Maße sind die Distanzen **Clark** und **Canberra** von Lance und Williams [LW66], die für zwei n-dimensionale Vektoren $X = (x_1,\ldots,x_n)$ und $Y = (y_1,\ldots,y_n)$ folgendermaßen definiert sind [NSCCRP03]:

$$Clark\ d(X,Y) = \sum_{i=1}^{n} \frac{|\ x_i - y_i\ |^2}{|\ x_i + y_i\ |^2}$$

und

$$Canberra\ d(X,Y) = \sum_{i=1}^{n} \frac{|\ x_i - y_i\ |}{|\ x_i + y_i\ |},$$

wobei auch die folgende Variante der Clark-Distanz existiert [WV03]:

$$Clark\ d(X,Y) = \sqrt{\sum_{i=1}^{n} \frac{|\ x_i - y_i\ |^2}{|\ x_i + y_i\ |^2}}\ .$$

Beide Maße summieren die so genannten *Bruchdifferenzen* zwischen den einzelnen Vektorattributen auf, wobei der Wert einer einzelnen Bruchdifferenz im Intervall [0; 1] liegt und die absoluten Werte - also indirekt die Vektorlängen - berücksichtigt werden. Allerdings sind sie in dieser Form nicht als normierte Vergleichsmaße einsetzbar, da die Summe der Bruchdifferenzen eine beliebige Zahl $S \geq 0$ sein kann. Eine Normierung der Distanzmaße sowie eine eventuelle Gewichtung der einzelnen Attribute lassen sich analog zur Normierung des Euklidischen Abstandes auf einfache Art folgenderweise durchführen [ZZ05]:

$$Norm.\,Clark\;d(X,Y) = \frac{\sum_{i=1}^{n} g_i \frac{|x_i - y_i|^2}{|x_i + y_i|^2}}{\sum_{i=1}^{n} g_i}, \text{ bzw.}$$

$$Norm.\,Clark\;d(X,Y) = \sqrt{\frac{\sum_{i=1}^{n} g_i \frac{|x_i - y_i|^2}{|x_i + y_i|^2}}{\sum_{i=1}^{n} g_i}}$$

und

$$Norm.\,Canberra\;d(X,Y) = \frac{\sum_{i=1}^{n} g_i \frac{|x_i - y_i|}{|x_i + y_i|}}{\sum_{i=1}^{n} g_i},$$

wobei g_i das Gewicht des i-ten Attributs repräsentiert. Auf diese Weise liegen die Distanzwerte im Intervall [0; 1], so dass beide Distanzmaße als normierte und bei Bedarf gewichtete Ähnlichkeitsmaße verwendet werden können, mit

$$Ähnlichkeit(X,Y) = 1 - Norm.\,Clark\;d(X,Y), bzw. 1 - Norm.\,Canberra\;d(X,Y).$$

Ein weiteres normiertes Abstandsmaß, das häufig in der Ökologie und Geologie verwendet wird und sich als Ähnlichkeitsmaß verwenden lässt, ist die **Bray-Curtis-Distanz**, die auch als *Sorensen-Distanz* bekannt ist. Sie ist folgendermaßen definiert:

$$BrayCurtis\;d(X,Y) = \frac{\sum_{i=1}^{n} |x_i - y_i|}{\sum_{i=1}^{n} x_i + y_i},$$

wobei Gewichtungen der einzelnen Attribute durch Hinzunahme von Gewichtungsfaktoren g_i vorgenommen werden können:

$$Gewichtete\ BrayCurtis\ d(X,Y) = \frac{\sum_{i=1}^{n} g_i \mid x_i - y_i \mid}{\sum_{i=1}^{n} g_i \cdot (x_i + y_i)},$$

wobei die Ähnlichkeit von zwei Vektoren analog zur Clark- bzw. Canberra-Distanz folgendermaßen ermittelt wird:

$$Ähnlichkeit(X,Y) = 1 - BrayCurtis\ d(X,Y), bzw. 1 - Gewichtete\ BrayCurtis\ d(X,Y).$$

Die vorgestellten Maße sind in dieser Form ungeeignet für nicht-metrische Attribute. Um Mischvektoren, also Vektoren deren Attribute aus einer Kombination metrischer, nicht-metrischer, normierter und nicht-normierter Merkmale bestehen, vergleichen zu können, sind *heterogene Ähnlichkeitsmaße* wie beispielsweise *HVDM* (*Heterogeneous Value Difference Metric*) [WM97] [SCRP98] oder das gewichtssensitive *L'Eixample-Ähnlichkeitsmaß* [NSCCal03] besser geeignet. Ein Nachteil vieler heterogener Verfahren liegt im Fehlen einer Trainingsmenge, die zur Bestimmung der Ähnlichkeit von nicht-metrischen Attributen notwendig ist. Eine Möglichkeit besteht dann in der Definition von Ähnlichkeitsfunktionen für die nicht-metrischen Variablen mit Hilfe von Ähnlichkeitsmatrizen. Alternativ können heterogene Abstands- und Ähnlichkeitsmaße verwendet werden, die bei Ungleichheit zweier nicht-metrischer Variablen einen hohen Beitrag zur Distanz bzw. einen niedrigen Beitrag zur Ähnlichkeit annehmen.

Die unterschiedlichen semantischen Merkmalsarten der Kundenattribute können mathematischen Typen zugeordnet werden. Bei soziodemographischen oder geographischen Kennzahlen (wie Postleitzahl, Alters- oder Einkommensgruppe) handelt es sich hauptsächlich um nicht-metrische Merkmale. Einfache verhaltensbezogene Attribute (beispielsweise der getätigte Gesamtumsatz oder die Anzahl der Einkäufe in einem Jahr), besitzen oft einen metrischen Wertebereich, wobei entsprechende verhaltensbezogene Attributvektoren häufig aus einer Mischung von absoluten Kennzahlen und Verhältniszahlen (wie beispielsweise Prozentangaben) bestehen. Aus diesem Grund normiere ich in solchen Fällen vor einer Ähnlichkeitsanalyse alle Merkmale auf den gleichen Wertebereich oder verwende geeignete Gewichtungsfaktoren. Umfangreiche Attributvektoren bestehen in der Praxis oft aus einer Mischung von metrischen und nicht-metrischen Merkmalen, entweder weil Kunden bezogen auf Mischtypologien klassifiziert oder miteinander verglichen werden sollen oder auch metrische soziodemographische Merkmale (wie z. B. das Einkommen) oder nicht-metrische verhaltensbezogene Merkmale (wie beispielsweise die ABC-Klassifizierung) verwendet

werden. In diesen Fällen können nur heterogene Ähnlichkeitsmaße angewendet werden.

5.4 Vergleich der Ergebnisse von Verhaltenssimulationen

Der klassische Vergleich von Kunden mit Hilfe soziodemographischer bzw. geographischer Attribute und einfachen verhaltensbezogenen Merkmalen führt zu relativ „oberflächlichen" Analysen, da auf diese Weise nicht das dynamische Kaufverhalten der Kunden verglichen wird. Für viele Fragestellungen bieten sich allerdings Vergleiche bezogen auf komplexe, verhaltensbasierte Merkmale wie beispielsweise Preis-, Werbe- oder Mailing-Sensitivität an, um zum Beispiel geeignete Zielgruppen für Werbeaktionen zu finden oder die von einer geplanten Maßnahme betroffenen Kunden zu identifizieren. Das Problem besteht hier in den meisten Fällen in der Beschaffung der entsprechenden Merkmale bzw. der konkreten Merkmalsausprägungen. Eine Möglichkeit zur Beschaffung dieser Art von Attributen und Merkmalen sind die in Kapitel 3 und 4 beschriebenen probabilistischen Verhaltensnetze und Verhaltenssimulationen.

5.4.1 Vergleich komplexer verhaltensbezogener Attribute

Um komplexe, verhaltensbezogene Kundenattribute zu vergleichen, berechne ich mit Hilfe probabilistischer Simulationen kundenindividuelle Ausprägungen, die anschließend in einer vektorisierten Form miteinander verglichen werden. Das bedeutet, dass die Beschreibungen der zu vergleichenden Kunden durch einen Attributvektor dargestellt werden, dessen Werte nicht einfach aus der Datenbank ermittelt werden, sondern erst mit Hilfe entsprechender Verhaltensnetze berechnet werden müssen.

Die konkrete Vorgehensweise besteht darin, dass ich für zwei Kunden jeweils die individuellen Auswirkungen einer Sequenz von Zustandskombinationen $TE = \{TE_1,\ldots, TE_t\}$ einer Menge von Einflussvariablen $E = \{E_1,\ldots, E_n\}$ auf eine Effektvariable R simuliere (siehe Abbildung 44). Die jeweils durch Inferenz ermittelten Auswirkungen auf R werden als Erwartungswerte in Vektoren gespeichert, die anschließend mit den oben beschriebenen vektorbasierten Vergleichsmaßen verglichen werden, um die Ähnlichkeit der beiden Kunden bezüglich des durch R repräsentierten Attributes zu ermitteln. Die Menge der Testkombinationen - und somit die Anzahl der durchzuführenden Simulationen - besteht dabei in der Regel aus allen möglichen Kombinationen vorgegebener Zustände der Einflussfaktoren, deren Anzahl sich wie folgt berechnet:

$$|TE| = t = |TE(E_1)| \cdot \ldots \cdot |TE(E_n)|,$$

wobei $TE(E_i)$ die Menge der für die Einflussvariable E_i vorgegebenen Testzustände repräsentiert. In der Praxis verwende ich als $TE(E_i)$ oft die Gesamtmenge oder eine Teilmenge aller in der Vergangenheit vorgekommenen Zustandswerte der Variable E_i (beispielsweise alle historischen Verkaufspreise eines Produktes für die Einflussvariable „Verkaufspreis“ oder sämtliche Bewerbungsarten für die Variable „Promotionsart“). Durch die Verwendung von Inferenz-Algorithmen für weiche Evidenzen ist es darüber hinaus möglich Zustände zu definieren, die in der Vergangenheit nicht vorgekommen sind (beispielsweise neue Verkaufspreise). Auf diese Weise lassen sich als $TE(E_i)$ auch relative Veränderungen von Werten angeben, um beispielsweise eine Reihe von relativen Preiserhöhungen zu simulieren.

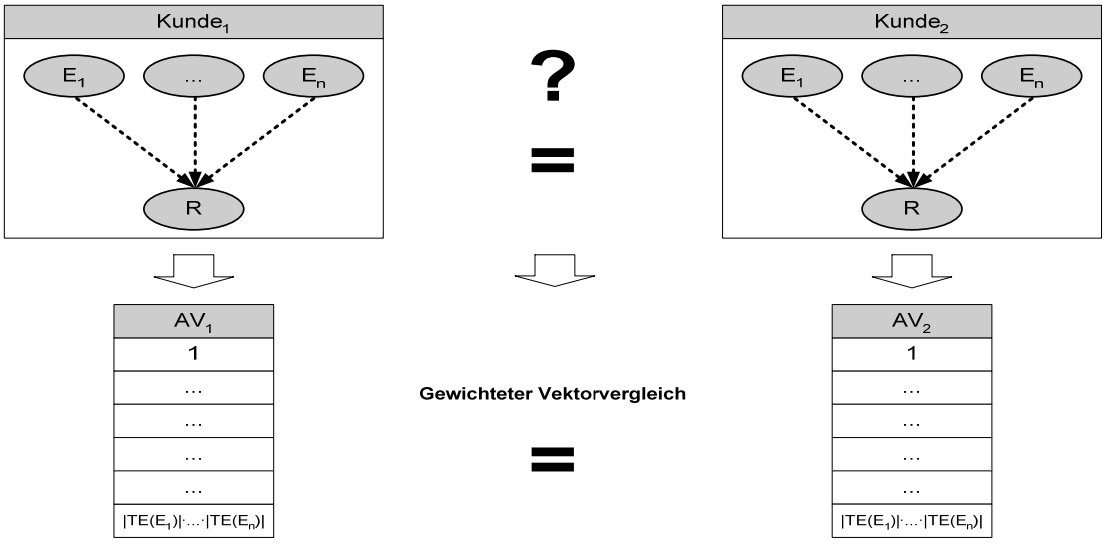


Abbildung 44: Vergleich komplexer verhaltensbezogener Attribute

Insgesamt ergibt sich zur Ermittlung der Ähnlichkeit zweier Kunden bezüglich komplexer verhaltensbezogener Attribute in Pseudocode folgender Algorithmus, der die Komplexität $O(|TE|\cdot n\cdot Z^n)$ besitzt:

```
CompareComplexBehaviour(K1, K2, TE, E, R)
   Foreach(Ki) do
      AVi = NewVector()
         Foreach(TEj) do
            AVi.Add(CalculateExpectationValue(Ki.BNet.Propagate(SetEvidences(TEj, E), R)))
Return CompareVectors(AV1, AV2)
```

Ein Beispiel für ein komplexes verhaltensbezogenes Attribut ist die Preissensibilität eines Kunden. Die Preissensibilität kann je nach Fragestellung als *absolute* oder *relative* Preis-

Absatz-Kurve dargestellt werden. Die absolute Kurve zeigt den absoluten Absatz in Abhängigkeit konkreter Preise, während die relative Kurve die prozentuale Änderung des Absatzes in Abhängigkeit von prozentualen Veränderungen des Preises darstellt, wobei üblicherweise entweder der aktuelle oder der durchschnittliche Verkaufspreis als Vergleichsmaßstab verwendet wird. Um zwei Kunden aufgrund ihrer Preissensibilität zu vergleichen, müssen die beiden kundenindividuellen Preisabsatzkurven erzeugt werden, die anschließend durch geeignete Verfahren verglichen werden. Je nachdem, ob sich die Preissensibilität auf einen einzelnen Artikel, eine Menge bestimmter Produkte oder im Extremfall auf das gesamte Sortiment eines Marktes beziehen soll und je nachdem wie viele historische Preisinformationen vorliegen, ist es mehr oder weniger aufwändig, entsprechende Kurven zu erzeugen. Falls die Preis-Absatzkurven unter Berücksichtigung weiterer Einflussfaktoren wie Bewerbung, Konkurrenzpreise oder zeitlicher Gesichtspunkte modelliert und verglichen werden sollen, ist es mit herkömmlichen Methoden entweder sehr aufwändig oder erst gar nicht möglich, eine entsprechende multidimensionale Preis-Absatzkurve mit ausreichender Detaillierung zu erzeugen. Durch die Verwendung der Verhaltensnetze kann der kundenindividuelle Absatz jedoch in Abhängigkeit der entsprechenden Einflussfaktoren modelliert und eine Annäherung an die gewünschte mehrdimensionale Absatzkurve durch Setzen geeigneter Evidenzsequenzen ermittelt werden.

5.4.2 Vergleich von simulierten Reaktionen

Der Vergleich von Kunden basierend auf simulierten Reaktionen ist eine Erweiterung des Vergleiches bezüglich komplexer, verhaltensbezogener Attribute. Ziel ist es, zwei Kunden aufgrund ihrer Reaktionen auf eine Menge von Simulationsszenarien $S = \{S_1,\ldots, S_s\}$ miteinander zu vergleichen (siehe Abbildung 45).

Ein Szenario S_i besteht aus einer Sequenz von Evidenzkombinationen $TE = \{TE_1,\ldots, TE_t\}$ bezüglich einer Menge von Einflussvariablen $E = \{E_1,\ldots, E_n\}$ (siehe Kapitel 4.1). Die Reaktionen der Kunden auf die vorgegebenen Evidenzkombinationen werden jeweils anhand der Erwartungswerte der Effektvariablen $R = \{R_1, \ldots, R_m\}$ gemessen und im Laufe der s Simulationsdurchläufe in m Attributvektoren der Länge s gespeichert. Anschließend wird die Ähnlichkeit der beiden Kunden durch einen gewichteten Vektorvergleich der resultierenden Vektoren ermittelt (beispielsweise als Durchschnitt der Ähnlichkeiten der Einzelvektoren $AV_{1,i}$ und $AV_{2,i}$). Der entsprechende Algorithmus zum Vergleich zweier Kunden bezüglich ihrer simulierten Reaktionen auf vorgegebene Szenarien lautet in Pseudocode wie folgt:

```
CompareSimulatedReactions(K1, K2, S, R)
   Foreach(Ki) do
      Foreach(Rj) do
         AVi, j = NewVector()
```

Foreach(S_k) do
Foreach(R_m) do
$AV_{i,m}$.Add(CalculateExpectationValue(K_i.Simulate(S_k, R_m)))
Return Average(CompareVectors($AV_{1,i}$, $AV_{2,i}$))

Die Komplexität dieses Algorithmus beträgt $O(|S|\cdot|R|\cdot|ST|\cdot n\cdot Z^n)$, wobei ST die maximale Menge der zu simulierenden Zeitpunkte innerhalb eines Szenarios ist und jeder der n Knoten im zugrunde liegenden Verhaltensnetz höchstens Z unterschiedliche Zustände besitzt.

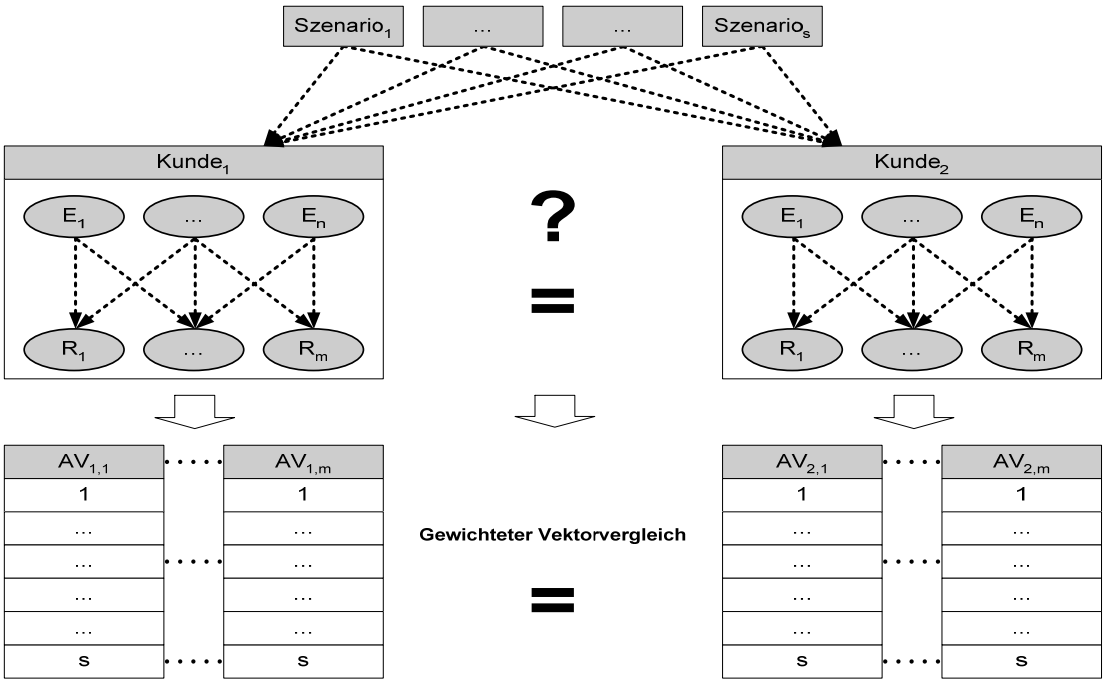


Abbildung 45: Vergleich simulierter Reaktionen

Beispielsweise können hundert verschiedene Simulationsszenarien bezogen auf unterschiedliche Produkte erstellt werden, wobei die Szenarien jeweils über einen Zeitraum von mehreren Wochen definiert sind, in denen verschiedene Preis- und Bewerbungsmaßnahmen sowohl innerhalb des Marktes als auch bei der Konkurrenz vorgegeben werden. Anschließend werden jeweils die Reaktionen zweier Kunden K_1 und K_2 auf jedes der hundert Szenarien simuliert, indem die absoluten Werte der kundenindividuellen Effektvariablen „Absatz“ und „Umsatz“ in Abhängigkeit der vorgegebenen Evidenzkombinationen ermittelt werden, die jeweils in Form eines Vektors der Länge s = 100 speichert werden. Die Ähnlichkeit der Kunden ergibt sich anschließend aus dem Durchschnitt der vektorbezogenen Ähnlichkeiten bezüglich der beiden Effektvariablen.

5.5 Direkter Vergleich kundenindividueller Verhaltensnetze

Um dynamische Verhaltensweisen zu vergleichen, werden Merkmale benötigt, die das kundenindividuelle Verhalten möglichst präzise unter Berücksichtigung einer Vielzahl von Einflussfaktoren widerspiegeln. Eine Repräsentationsmöglichkeit ist die Verwendung von Attributvektoren, deren konkreten Ausprägungen mit Hilfe der kundenindividuellen Verhaltensnetze durch Simulationen ermittelt werden (siehe Kapitel 5.4).

Eine andere Möglichkeit besteht im direkten Vergleich der Verhaltensnetze, die das Kundenverhalten in Abhängigkeit aller möglichen Einflussfaktorkombinationen nichtlinear repräsentieren und aus diesem Grund eine deutlich detailliertere Beschreibung des dynamischen Kaufverhaltens bieten als klassische Attributvektoren. Selbst wenn ein Vektor aus der gesamten Wahrscheinlichkeitsverteilung über alle Zustandskombinationen aller Zufallsvariablen gebildet würde - was in der Praxis schon an der exponentiellen Komplexität scheitert - würde der Vergleich solcher Vektoren neben der relativ geringen Anzahl unbekannter signifikanter Abhängigkeiten eine deutlich größere Anzahl von unrelevanten Kombinationen berücksichtigen, die das Ergebnis enorm verfälschen. Und um den Vektor auf die signifikanten Werte zu reduzieren, müssten diese erst durch geeignete Verfahren identifiziert werden, die dann dem Lernen von probabilistischen Netzen entsprächen.

Interessanterweise gibt es in der Literatur eine Vielzahl von Vergleichs- bzw. Abstandsmaßen für Vektoren, aber kaum geeignete Vergleichs- oder Distanzmaße für probabilistische Netze. Zwar existieren viele Ähnlichkeitsmaße für Graphen und Netze [CKS98] und einige Graph Matching Algorithmen [BH00], die aber nur die Struktur eines Netzes berücksichtigen und Wahrscheinlichkeitsverteilungen nicht beachten. Lediglich Wang und Vassileva haben ein Vergleichsverfahren für naive Bayes'sche Netze entwickelt [WV03], das auf Clarks Distanzmaß basiert.

Um zwei probabilistische Verhaltensnetze zu vergleichen, gilt es in erster Linie, eine Ähnlichkeitsanalyse der zugrunde liegenden Gesamtwahrscheinlichkeitsverteilungen bzw. der Abhängigkeiten zwischen den einzelnen Variablen durchzuführen, da zwei probabilistische Netze durchaus dieselbe Wahrscheinlichkeitsverteilung repräsentieren können, obwohl sich ihre Graphstrukturen stark unterscheiden.

Im Folgenden präsentiere ich die im Rahmen dieser Arbeit entstandenen Verfahren und Ähnlichkeitsmaße zum Vergleich probabilistischer Verhaltensnetze, die voraussetzen, dass die zu vergleichenden Netze N_1 und N_2 die selben semantischen Knoten $X_1,\ldots, X_n$ besitzen, da ein Vergleich sonst keinen Sinn machen würde. Die Verfahren sind im Einzelnen:

- Vergleich der Knotenkorrelationen zwischen den Netzen
- Vergleich der Knotenrelationen innerhalb der Netze
- Vergleich der bedingten Wahrscheinlichkeitsverteilungen

- Vergleich der Randwahrscheinlichkeitsverteilungen
- Vergleich von Evidenzstichproben

5.5.1 Vergleich der Knotenkorrelationen zwischen den Netzen

Da die beiden zu vergleichenden probabilistischen Verhaltensnetze N_1 und N_2 nach Voraussetzung dieselbe Knotenmenge $X_1,\ldots, X_n$ besitzen und die zugrunde liegenden Data-Mining-Tabellen mit den Spalten $S_1,\ldots, S_n$ bekannt sind, kann ein naiver Vergleich der einzelnen Knotenkorrelationen zwischen den Netzen durchgeführt werden. Dazu wird für jeden Knoten X_i eine Korrelationsanalyse auf den beiden entsprechenden (eventuell diskretisierten) DM-Tabellenspalten $S_{1,i}$ und $S_{2,i}$ durchgeführt, wobei die Korrelationsergebnisse anschließend gemittelt werden. Dabei kann für jeden Knoten ein individuelles Gewicht g_i definiert werden, um interessante Knoten stärker zu gewichten. Allerdings ist diese Vorgehensweise keine sehr geeignete Methode zum Vergleich probabilistischer Verhaltensnetze, weil auf diese Weise die Relationen und bedingten Wahrscheinlichkeiten zwischen den Knoten eines Netzes unberücksichtigt bleiben und unterschiedliche Zeitpunktbezüge in den beiden DM-Tabellen die Aussagekraft des Ergebnisses zudem stark beeinträchtigen können. Das naive Verfahren besitzt die Komplexität $O(n\cdot|D|)$ und lautet in Pseudocode:

CompareExternalCorrelations(N_1, N_2)
Return Average(Correlation(N_1.DataMiningTable.Column[X_i], N_2.DataMiningTable.Column[X_i]))

5.5.2 Vergleich der Knotenrelationen innerhalb der Netze

Probabilistische Verhaltensnetze zeichnen sich neben ihrer Netzstruktur vor allem durch die bedingten Abhängigkeiten zwischen den Zufallsvariablen aus. Um eine Ähnlichkeitsanalyse zwischen zwei Verhaltensnetzen N_1 und N_2 mit den Knoten $X_1, \ldots, X_n$ durchzuführen, liegt es nahe, die Abhängigkeiten zwischen den Knoten innerhalb des Netzes N_1 mit den Abhängigkeiten innerhalb des Netzes N_2 zu vergleichen, um festzustellen, ob in beiden Netzen ähnliche Abhängigkeiten zwischen Knotenpaaren bestehen.

Konkret berechne ich Korrelationen zwischen Knotenpaaren beider Netze, um diese in Vektoren zu speichern, die anschließend mit geeigneten vektorbezogenen Distanz- und Ähnlichkeitsmaßen verglichen werden. Diese Vorgehensweise setzt für die Durchführung der einzelnen Korrelationsanalysen die Existenz der Data-Mining-Tabellen voraus, die im Falle der von mir verwendeten Verhaltensnetze zutrifft. Zur Durchführung der Korrelationsanalysen setze ich neben den Standardkorrelationsverfahren auch vektorbasierte Distanz- und Ähnlichkeitsmaße ein, da die entsprechenden Tabellenspalten auch als Vektoren interpretiert werden können. Vor allem aber bietet sich die in Kapitel 3.3.4.2 vorgestellte *Mutual Information* an, da sie auf bedingten Wahrscheinlichkeiten basiert und beim Erlernen

der Netze eine große Rolle spielt.

Bei der Auswahl der zu vergleichenden Knotenpaare ergeben sich folgende Alternativen, wobei einzelne Knotenpaare jeweils durch individuelle Gewichtung mehr oder weniger stark berücksichtigt werden können:

- Vergleich der Korrelationen zwischen allen Knotenpaaren (Variante 1)
- Vergleich der Korrelationen zwischen Knotenpaaren, die mindestens in einem der Netze eine Kante besitzen (Variante 2)
- Vergleich der Korrelationen zwischen durch Domänenwissen vorgegebenen Knotenpaaren *EL* (Variante 3)

Alle drei Varianten besitzen die Komplexität $O(n^2 \cdot Z^2)$, da für $O(n^2)$ Knotenpaare die Mutual Information mit der jeweiligen Laufzeit $O(Z^2)$ berechnet werden muss. Der entsprechende Pseudocode der Verfahren lautet:

```
CompareInternalCorrelations(N1, N2) // Variante 1
   Foreach(Ni) do
      Vi = NewVector()
      Foreach(Xj, Xk) do
         Vi.Add(MutualInformation(Ni.Node[Xj], Ni.Node[Xk]))
Return CompareVectors(V1, V2)

CompareInternalCorrelations(N1, N2) // Variante 2
   Foreach(Ni) do
      Vi = NewVector()
      Foreach(Xj, Xk) do
         If (Edge.Exists(N1.Edge[Xj, Xk]) OR Edge.Exists(N2.Edge[Xj, Xk])) then
            Vi.Add(MutualInformation(Ni.Node[Xj], Ni.Node[Xk]))
Return CompareVectors(V1, V2)

CompareInternalCorrelations(N1, N2, EL) // Variante 3
   Foreach(Ni) do
      Vi = NewVector()
      Foreach(Xj, Xk in EL) do
         Vi.Add(MutualInformation(Ni.Node[Xj], Ni.Node[Xk]))
Return CompareVectors(V1, V2)
```

5.5.3 Vergleich der bedingten Wahrscheinlichkeitsverteilungen

Eine sinnvolle Vorgehensweise zum Vergleich probabilistischer Verhaltensnetze besteht im direkten Vergleich der von den Netzen repräsentierten bedingten Gesamtwahrscheinlichkeitsverteilung zwischen den diskretisierten Zufallsvariablen. Das bedeutet, dass zum Vergleich zweier Verhaltensnetze N_1 und N_2 der gesamte Ereignisraum über allen Zuständen der Zufallsvariablen betrachtet werden müsste. Dieser wächst allerdings bei probabilistischen Netzen exponentiell mit Anzahl der Knoten und deren zulässigen Zuständen. Aus diesem Grund scheint ein Vergleich über alle möglichen Ereignisse nicht praktikabel.

Es bietet sich allerdings an, sämtliche Wahrscheinlichkeitsverteilungen der bedingten Wahrscheinlichkeitstabellen (CPT) sowie der Apriori-Tabellen (APT) zu vergleichen. Die Gesamtähnlichkeit der beiden Netze ergibt sich anschließend aus dem Durchschnitt der Ähnlichkeiten zwischen den einzelnen Wahrscheinlichkeitstabellen. Diese Vorgehensweise setzt voraus, dass die Tabellen bei beiden Netzen N_1 und N_2 eine identische Struktur aufweisen, d. h. dass bei den CPT dieselben Elternknoten in derselben Reihenfolge vorliegen und alle Knoteninstanzen dieselben Diskretisierungen besitzen. Diese Bedingung stellen auch Wang und Vassileva bezüglich ihres Vergleichsverfahrens für einfache naive Bayes'sche Netze [WV03].

Da diese Voraussetzung in der Praxis in den meisten Fällen nicht erfüllt wird, führe ich vor einer Ähnlichkeitsanalyse eine Anpassung der Wahrscheinlichkeitstabellen der beiden Netze durch. Eine vollständige Anpassung ist dabei nur mit Hilfe der Trainingsfälle der Data-Mining-Tabellen möglich, die allerdings im Falle der von mir verwendeten Verhaltensnetze bekannt sind.

Zur Strukturanpassung der beiden Netze existieren zwei Möglichkeiten. Die erste besteht darin, ein Netz an die Struktur des anderen anzupassen. Dabei werden für jeden Knoten sowohl die Diskretisierung als auch die Menge der Elternknoten übernommen, um anschließend mit Hilfe der Trainingsfälle entsprechende CPT bzw. Apriori-Wahrscheinlichkeitstabellen zu lernen. Bei der Anpassung der Diskretisierung eines Knotens X_1 an die eines Knotens X_2 wird vorausgesetzt, dass die Diskretisierung von X_2 auch auf den Wertebereich von X_1 passt. Bei den von mir verwendeten Verhaltensnetzen ist diese Voraussetzung immer erfüllt, da die Intervalle stets alle Fließkommawerte von $-\infty$ bis $+\infty$ lückenlos abdecken. Die Anpassung der Struktur der CPT und APT setzt die vorherige Anpassung der Knotendiskretisierungen aller beteiligten Knoten voraus. Die algorithmische Vorgehensweise zur Netzanpassung besitzt eine Komplexität von $O(n^2+n\cdot Z^n)$ und lautet in Pseudocode:

```
AdaptTo(N1, N2) // Adapts N2 to N1
   Foreach (Xi in N1.Nodes) do
      AdaptDiscretisationTo(Xi, N2.Node[Xi])
   Foreach (Xi in N1.Nodes) do
      AdaptParentsTo(Xi, N2.Node[Xi])
      N2.Node[Xi].LearnProbabilityTable // APT or CPT
Return N2
```

Die zweite Möglichkeit ist eine Vereinigung N beider Netzstrukturen N_1 und N_2 zu bilden und anschließend beide Netze an diese anzupassen. Die Struktur von N entsteht, indem für jeden Knoten X_i die Menge seiner Eltern durch die Vereinigung $N_1.Parents(X_i) \cap N_2.Parents(X_i)$ der Eltern von X_i in N_1 und N_2 gebildet wird (mit der Komplexität von $O(n^2)$):

```
CreateNetStructureUnion(N1, N2)
   N = NewBehaviourNetwork()
   N.Nodes = N1.Nodes
   Foreach (Xi in N.Nodes) do
      Xi.Parents = N1.Node[Xi].Parents UNION N2.Node[Xi].Parents
Return N
```

Die Diskretisierung eines Knotens X_i in N wird entweder durch „Mischung“ der Diskretisierungen der Knoten $N_1.X_i$ und $N_2.X_2$ gebildet (Variante 1) oder durch Neuberechnung einer geeigneten gemeinsamen Diskretisierung für $N_1.X_i$ und $N_2.X_2$ ermittelt (Variante 2).

Bei der Mischung der Diskretisierungen der Knoten $N_1.X_i$ und $N_2.X_2$ werden diese quasi „übereinander“ gelegt, um eine neue Intervallaufteilung zu erhalten. Wenn $D_1 = \{[x_{1a}, x_{1b}[, \ldots, [x_{ra}, x_{rb}[\}$ die Diskretisierung von $N_1.X_i$ und $D_2 = \{[y_{1a}, y_{1b}[, \ldots, [y_{sa}, y_{sb}[\}$ diejenige von $N_2.X_i$ repräsentiert, dann kann eine Mischung der Diskretisierungen durch Vereinigung und Sortierung der Intervallgrenzen in aufsteigender Reihenfolge zur distinkten Menge $\{m_1, \ldots, m_t\}$, wobei $m_i<m_{i+1}$, und anschließender Intervallbildung der Art $\{]-\infty, m_1[, [m_1, m_2[, [m_2, m_3[, \ldots, [m_{t-1}, m_t[, [m_t, +\infty[\}$ durchgeführt werden. Die Komplexität dieser Variante beträgt dann insgesamt $O(n^2+n \cdot Z^n)$.

Zur Berechnung einer gemeinsamen Diskretisierung werden die distinkten Werte der DM-Tabellenspalten von $N_1.X_i$ und $N_2.X_i$ vereint und anschließend durch erneute Diskretisierung in geeignete Intervalle aufgeteilt. Die Komplexität dieser Variante beträgt bei Verwendung des in Kapitel 3.3.2 vorgestellten Diskretisierungsverfahrens $O(n \cdot |V|^2+n^2+n \cdot Z^n)$. Der Pseudocode der beiden Varianten zur Strukturanpassung lautet:

```
AdaptNetStructures(N1, N2) // Variante 1
   N = CreateNetStructureUnion(N1, N2)
   Foreach (Xi in N.Nodes) do
      Xi.Discretisation = Merge(N1.Node[Xi].Discretisation,  N2.Node[Xi].Discretisation)
   Foreach (Ni)
      Ni = AdaptTo(N, Ni)
Return N1, N2

AdaptNetStructures(N1, N2) // Variante 2
   N = CreateNetStructureUnion(N1, N2)
   N.DataMiningTable = N1.DataMiningTable UNION N2.DataMiningTable
   N.Nodes.Discretisation
   Foreach(Ni)
      Ni = AdaptTo(N, Ni)
Return N1, N2
```

Um die Ähnlichkeit der beiden Netze N_1 und N_2 zu berechnen, werden entweder beide Netze an die vereinigte Struktur adaptiert und anschließend die Wahrscheinlichkeitsverteilungen in den angepassten CPT und APT verglichen oder man berechnet sowohl die Ähnlichkeit der Wahrscheinlichkeitsverteilungen bei einer Anpassung von N_2 an N_1 als auch die bei einer Anpassung von N_1 an N_2, und bildet anschließend den Durchschnitt der beiden Ergebnisse. Der entsprechende Algorithmus lautet in Pseudocode:

```
CompareNetProbabilities(N1, N2, Option)
   Case of (Option)
      Case „Union“:
         AdaptNetStructures(N1, N2)
         Return ComparePTs(N1, N2)
      Case “Cross-Over”:
         Return Average(ComparePTs(AdaptTo(N1, N2), N1), ComparePTs(AdaptTo(N2, N1), N2))
 Return null
```

Abbildung 46 zeigt ein Beispiel mit zwei probabilistischen Verhaltensnetzen N_1 und N_2. Beide Netze bestehen aus den Zufallsvariablen X, Y und Z, entsprechenden Wahrscheinlichkeitstabellen, sowie den zugrunde liegenden Data-Mining-Tabellen mit den Trainingsfällen.

Zum Vergleich der beiden Netze müssen sowohl die Diskretisierungen der Knoten X und Y als auch die Struktur der bedingten Wahrscheinlichkeitstabellen des Knotens Z angepasst werden.

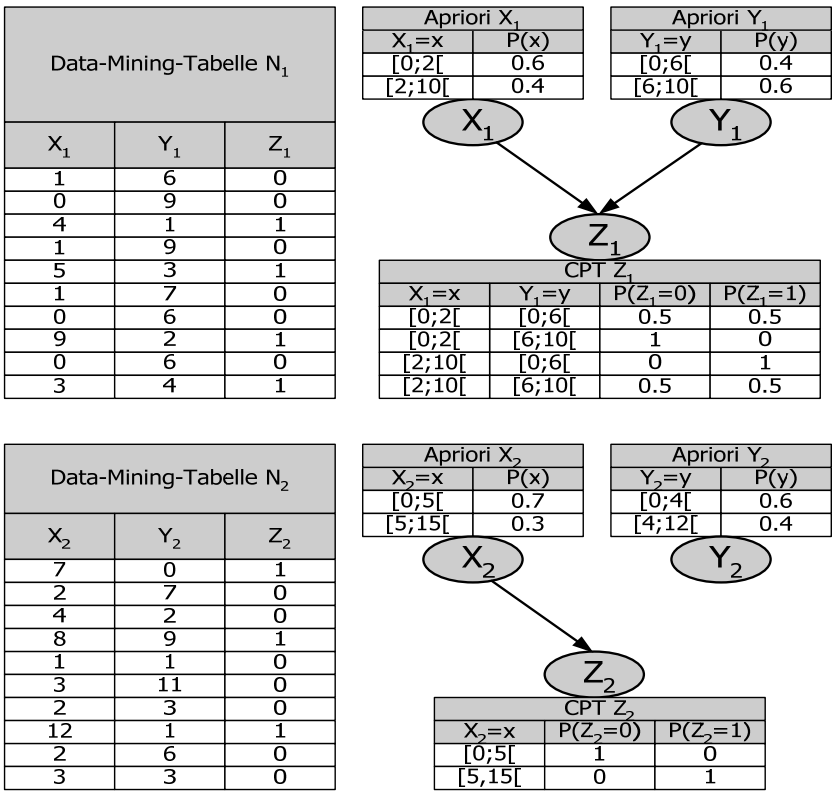


Abbildung 46: Beispielnetze N_1 und N_2 mit Data-Mining-Tabellen

Die Diskretisierungen der Knoten X und Y decken unterschiedliche Wertebereiche zwischen 0 und maximal 15 ab. Eine Anpassung ist mit Hilfe der Werte in den Data-Mining-Tabellen möglich. Abbildung 47 zeigt die Intervallmischung der Knoten X_1 und X_2 bzw. Y_1 und Y_2.

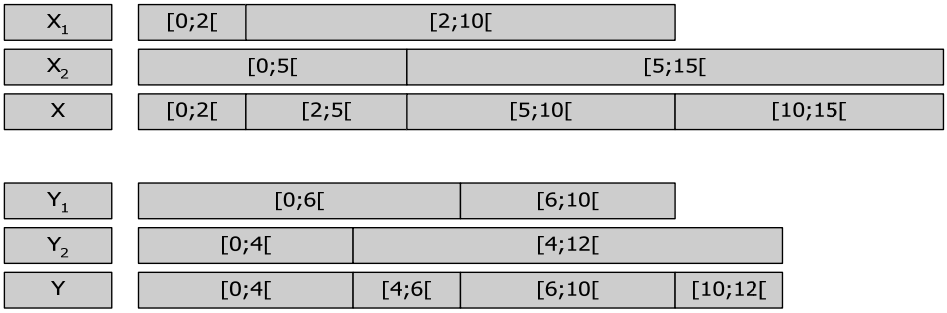


Abbildung 47: Intervallmischung für die Knoten X_1 und X_2 bzw. Y_1 und Y_2

Die Anpassung der CPT-Struktur des Knotens Z kann durch Vereinigung der Elternknoten erreicht werden (Vereinigung von $\{X, Y\}$ und $\{X\}$ zu $\{X, Y\}$). Alternativ wird entweder Z_1 an Z_2 angeglichen (durch Wegfall von Y) oder Z_2 an Z_1 (durch Hinzunahme von Y).

Nach Anpassung der Diskretisierungen und Wahrscheinlichkeitstabellen können die bedingten Wahrscheinlichkeitsverteilungen zweier Netze N_1 und N_2 mit Hilfe unterschiedlicher Verfahren miteinander verglichen werden.

Die Wahrscheinlichkeitsverteilungen in den Tabellen können entweder als eine komplexe bedingte Wahrscheinlichkeitsverteilung über den entsprechenden Ereignisraum (Variante 1) oder als ein einzelner Vektor aufgefasst werden (Variante 2).

In der ersten Variante können die den angepassten Wahrscheinlichkeitstabellen zugrunde liegenden bedingten Wahrscheinlichkeitsverteilungen mit geeigneten Vergleichsmaßen verglichen werden, während man die Vektoren in der zweiten Variante mit klassischen, vektorbasierten Korrelations- und Ähnlichkeitsmaßen vergleichen kann. Beispielsweise benutzen Wang und Vassileva Clarks Distanzmaß zum Vergleich der naiven Wahrscheinlichkeitsverteilungen [WV03].

Ein bekanntes Maß zum Vergleich zweier Wahrscheinlichkeitsverteilungen über denselben Ereignisraum ist die so genannte *Kullback-Leibler-Entropie* (oder auch *KL-Divergenz*) [KL51]. Die KL-Divergenz zwischen zwei Wahrscheinlichkeitsverteilungen P und Q über die Zufallsvariablen X_1, …, X_n und dem Ereignisraum $E = \{e_1, \ldots, e_t\}$, wobei e_i die Wahrscheinlichkeit p_i unter P und q_i unter Q besitzt, ist folgendermaßen definiert:

$$KL(P,Q) = \sum_{i=1}^{t} p_i \log \frac{p_i}{q_i}$$

Da die Kullback-Leibler-Divergenz nicht symmetrisch ist, d. h. es gilt in der Regel $KL(P, Q) \neq KL(Q, P)$, berechne ich beide Richtungen, um anschließend den Mittelwert zu bilden. Das modifizierte Kullback-Leibler-Gewicht (*KLWeight*) bezüglich zweier Wahrscheinlichkeitsverteilungen definiere ich somit wie folgt:

$$\mathsf{KLWeight}(P,Q) = \mathsf{KLWeight}(Q,P) = \frac{KL(P,Q) + KL(Q,P)}{2}.$$

Die gesamte Kullback-Leibler-Divergenz zwischen zwei probabilistischen Verhaltensnetzen mit n Knoten ist dann die Summe der Mittelwerte der KL-Gewichte sämtlicher Knotenwahrscheinlichkeitsverteilungen:

$$\mathsf{KL\text{-}Divergenz}(\mathsf{N}_1,\mathsf{N}_2) = \sum_{i=1}^{n} \mathsf{KLWeight}(P_i,Q_i) = \frac{\sum_{i=1}^{n} KL(P_i,Q_i) + \sum_{i=1}^{n} KL(Q_i,P_i)}{2},$$

wobei P_i die Wahrscheinlichkeitsverteilung der Tabelle des i-ten Knotens von N_1 und Q_i die Wahrscheinlichkeitsverteilung der Tabelle des i-ten Knotens von N_2 repräsentiert. Alternativ können andere Vergleichsmaße für Wahrscheinlichkeitsverteilungen angewendet werden, wie beispielsweise die auf Kullback-Leibler aufbauende symmetrische *Resistor-Average-Distanz* (*RAD*) [JS01] oder die *Chernoff-Distanz* (*CD*) [CH52], die allerdings nicht die Dreieckungleichung erfüllt [KT67]. Mit der symmetrischen RAD-Distanz ist das algorithmische Vorgehen zur Ermittlung der Ähnlichkeit zwischen N_1 und N_2 nach eventuellen Strukturanpassungen beispielsweise folgende:

```
ComparePTs (N1, N2) // Variante 1
   Similarity = 0;
   Foreach (Xi in N1.Nodes) do
      Similarity += RAD-Distance(N1.Node[Xi].CPT, N2.Node[Xi].CPT)
 Return Similarity // optional "Return Similarity / N1.Nodes.Count" to obtain the average
```

Ein Nachteil der beschriebenen Maße ist, dass sowohl die Kullback-Leibler-Divergenz als auch die RAD- und CD-Distanzen keine normalisierten Abstandsmaße sind, d. h. sie nur zu relativen Vergleichen verwendet werden können. Zur Anwendung normalisierter Vergleichsmaße können die Wahrscheinlichkeitstabellen alternativ als große Vektoren (Variante 2a) bzw. als Menge von Vektoren angesehen werden (Variante 2b), die mit vektorbasierten Distanz- bzw. Ähnlichkeitsmaßen verglichen werden, um die Ähnlichkeit der Wahrscheinlichkeitsverteilungen zu berechnen:

```
ComparePTs (N1, N2) // Variante 2a
   Foreach (Ni) do
      Vi = NewVector()
      Foreach (Xj in Ni.Nodes) do
         Foreach(Columnk in Ni.Node[Xj].CPT) do // If Xj has no parents then CPT = Apriori
            Vi.Add(Columnk)
Return CompareVectors(V1, V2)

ComparePTs (N1, N2) // Variante 2b
   Foreach (Ni) do
      Foreach (Xj in Ni.Nodes) do
         Vi,j = NewVector()
         Foreach(Columnk in Ni.Node[Xj].CPT) do // If Xj has no parents then CPT = Apriori
            Vi,j.Add(Columnk)
Return Average(CompareVectors(V1,i, V2,i))
```

Zusammenfassend ist der Vergleich der bedingten Wahrscheinlichkeitsverteilungen, nach einer Anpassung sämtlicher Diskretisierungen und Wahrscheinlichkeitstabellen auf Basis der konkreten Trainingsfälle in den Data-Mining-Tabellen, eine sinnvolle Methode, um eine Ähnlichkeitsanalyse zwischen zwei probabilistischen Netzen durchzuführen. Ein Nachteil besteht in der aufwändigen Anpassung der Netzstrukturen. Die Komplexität der drei vorgestellten Varianten des ComparePTs-Verfahrens beträgt jeweils $O(n \cdot Z^n)$.

5.5.4 Vergleich der Randwahrscheinlichkeitsverteilungen

Zur Bestimmung der Ähnlichkeit zweier probabilistischer Netze N_1 und N_2 kann alternativ zum Vergleich der bedingten Wahrscheinlichkeiten ein paarweiser Vergleich der *Randverteilungen* (*marginal distributions*) der einzelnen Netzknoten X_i durchgeführt werden. Auf diese Weise sinkt zwar die Komplexität, aber damit auch die Aussagequalität, da während des Vergleichs bedingte Abhängigkeiten zwischen den einzelnen Zufallsvariablen unberücksichtigt bleiben. Dennoch liefert das Analyseergebnis in der Praxis hinreichende Anhaltspunkte über die mögliche Existenz einer hohen Ähnlichkeit zwischen zwei probabilistischen Netzen.

Die Berechnungen der einzelnen Randwahrscheinlichkeiten p_i einer Zufallsvariable X_i mit z möglichen Zuständen bzw. Zustandsintervallen $x_1,\ldots,x_z$ und den Eltern $Parents(X_i)$ mit den möglichen Zustandskombinationen $parents(X_i)$ lassen sich (wie in Kapitel 2.3.2 beschrieben) mit folgender Formel durchführen:

$$P(X_i = x_i) = p_i = \sum_{parents(X_i)} P(X_i = x_i \mid parents(X_i)) \cdot P(parents(X_i)) \text{ mit } \sum_{i=1}^{z} p_i = 1.$$

In meinem Beispiel (Kapitel 5.5.3) berechnet sich die Randwahrscheinlichkeit $P(Z_1=0)$ folgendermaßen:

$$P(Z_1 = 0) = \sum_{x,y} P(Z_1 = 0 \mid X_1 = x \wedge Y_1 = y) \cdot P(X_1 = x \wedge Y_1 = y) =$$

$$P(Z_1 = 0 \mid X_1 = [0;2] \wedge Y_1 = [0;6]) \cdot P(X_1 = [0;2] \wedge Y_1 = [0;6]) +$$

$$P(Z_1 = 0 \mid X_1 = [0;2] \wedge Y_1 = [6;10]) \cdot P(X_1 = [0;2] \wedge Y_1 = [6;10]) +$$

$$P(Z_1 = 0 \mid X_1 = [2;10] \wedge Y_1 = [0;6]) \cdot P(X_1 = [2;10] \wedge Y_1 = [0;6]) +$$

$$P(Z_1 = 0 \mid X_1 = [2;10] \wedge Y_1 = [6;10]) \cdot P(X_1 = [2;10] \wedge Y_1 = [6;10])$$

$$= 0{,}12 + 0{,}36 + 0 + 0{,}12 = 0{,}6$$

Entsprechend ergibt sich für $P(Z_1=1) = 0{,}4$ und somit $P(Z_1=0) + P(Z_1=1) = 1$. Da X_1 und Y_1

keine Eltern besitzen sind ihre Randwahrscheinlichkeiten die entsprechenden Werte in den Apriori-Wahrscheinlichkeitstabellen.

Voraussetzung für den Vergleich der Randwahrscheinlichkeitsverteilungen ist, dass die jeweiligen Instanzen der Zufallsvariablen X_i in beiden Netzen dieselbe Diskretisierung besitzen. Die Vorgehensweise zur Anpassung der Diskretisierungen habe ich in Kapitel 5.5.3 beschrieben. Allerdings müssen für den Vergleich der Randwahrscheinlichkeiten bei einer Anpassung der zu vergleichenden Netze nicht nur die Diskretisierungen, sondern darauf aufbauend auch die entsprechenden Randwahrscheinlichkeitsverteilungen neu berechnet werden. Mit Hilfe der konkreten Trainingsfälle in den DM-Tabellen können entsprechende Berechnungen durchgeführt werden. Stehen keine DM-Tabellen zur Verfügung, können die Randwahrscheinlichkeiten an Mischdiskretisierungen durch das folgende Verfahren angepasst werden, das eine Gleichverteilung der Werte innerhalb aller Zustandsintervalle annimmt.

Ich erläutere die Vorgehensweise am Beispiel eines Knotens X_1 mit der ursprünglichen Diskretisierung $D_1 = \{[x_{1a}, x_{1b}[, \ldots, [x_{ra}, x_{rb}[\}$ und den entsprechenden Randwahrscheinlichkeiten $p_1, \ldots, p_r$, der zusammen mit einem Knoten X_2 an die gemeinsame Mischdiskretisierung $\{[m_{1a}, m_{1b}[, \ldots, [m_{sa}, m_{sb}[\}$ angepasst werden soll, wobei x_{min} den minimalen und x_{max} den maximalen Wert in den Trainingsfällen von X_1 und X_2 repräsentieren. Die Aufgabe ist dabei die Berechnung der neuen Randwahrscheinlichkeiten q_i mit $1 \leq i \leq s$ des Knotens X_i. Je nach Aufteilung der neuen Intervalle müssen dabei folgende Fallunterscheidungen betrachtet werden:

- Ein neues Intervall $[m_{ia}, m_{ib}[$ liegt komplett innerhalb eines alten Intervalls $[x_{ja}, x_{jb}[$ und $x_{ja} \neq -\infty$ und $x_{jb} \neq +\infty$. In diesem Fall berechnet sich q_i wie folgt:
$$q_i = p_j \frac{m_{ib} - m_{ia}}{x_{jb} - x_{ja}}$$
- Ein neues Intervall $[m_{ia}, m_{ib}[$ liegt komplett innerhalb eines alten Intervalls $[x_{ja}, x_{jb}[$ und $x_{ja} = -\infty$ und $m_{ia} \geq x_{min}$:
$$q_i = p_1 \frac{m_{ib} - m_{ia}}{x_{jb} - x_{min}}$$
- Ein neues Intervall $[m_{ia}, m_{ib}[$ liegt komplett innerhalb eines alten Intervalls $[x_{ja}, x_{jb}[$ und $x_{ja} = -\infty$ und $m_{ia} < x_{min}$:
$$q_i = 0$$
- Ein neues Intervall $[m_{ia}, m_{ib}[$ liegt komplett innerhalb eines alten Intervalls $[x_{ja}, x_{jb}[$ und $x_{jb} = +\infty$ und $m_{ib} \leq x_{max}$:
$$q_i = p_j \frac{m_{ib} - m_{ia}}{x_{jb} - x_{max}}$$

- Ein neues Intervall $[m_{ia}, m_{ib}[$ liegt komplett innerhalb eines alten Intervalls $[x_{ja}, x_{jb}[$ und $x_{jb} = +\infty$ und $m_{ib} > x_{max}$:

$$q_i = 0$$

Ein neues Intervall wird immer komplett in einem alten Intervall liegen, da die neue Intervallaufteilung eine Verfeinerung der alten Diskretisierung ist. Analog können beliebige Verteilungen innerhalb der Werteintervalle angenommen werden, wobei es denkbar ist, unterschiedlichen Knoten unterschiedliche Verteilungen zuzuordnen, um eventuell vorhandenes Domänenwissen über die „typischen" Zustandsverteilungen der Zufallsvariablen berücksichtigen zu können. Falls genauere Informationen über die Werteverteilung in den alten Intervallen vorliegen – beispielsweise in Form einer Liste und Gewichtung der tatsächlich in die Intervalle fallenden Trainingswerte – können die Berechnungen der neuen Randwahrscheinlichkeiten entsprechend exakter durchgeführt werden.

Abbildung 48 zeigt die Anpassung der Randwahrscheinlichkeiten der Knoten X_1 und X_2 meines Beispiels aus Abbildung 46, unter der Annahme, dass die Werte in allen Intervallen gleich verteilt sind.

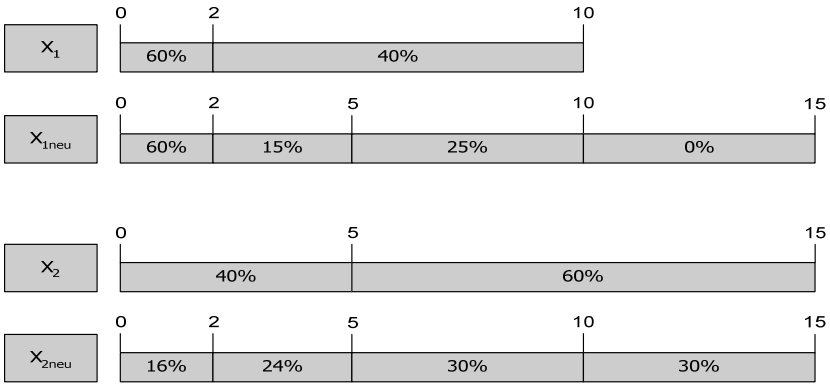


Abbildung 48: Neuberechnung der Randwahrscheinlichkeiten

Insgesamt ergeben sich, je nach Existenz der Data-Mining-Tabellen, folgende algorithmische Vorgehensweisen zum Vergleich der Randwahrscheinlichkeiten zweier probabilistischer Netze N_1 und N_2, deren Komplexität jeweils nur $O(n \cdot Z)$ beträgt:

```
CompareMarginalProbabilities(N1, N2) // Variante 1, using data mining tables
                                     // Implies an adaption of the net structures
  Foreach (Ni) do
    Foreach (Xj in Ni.Nodes) do
      Vi,j = NewVector()
      Foreach(Intervalk in Ni.Node[Xj].Discretisation.Intervals) do
        Vi,j.Add(Intervalk.Marginalprobability)
Return Average(CompareVectors(V1,i, V2,i))
```

```
CompareMarginalProbabilities(N1, N2) // Variante 2, without data mining tables
   Foreach (Ni) do
      Foreach (Xj in Ni.Nodes) do
         Vi,j = NewVector()
         Y = Copy(Xj)
         Y.Discretisation = Merge(N1.Node[Xj].Discretisation,  N2.Node[Xj].Discretisation)
         Foreach(Intervalk in Y.Discretisation.Intervals) do
            Vi,j.Add(CalculateMarginalProbability(Intervalk, Ni.Node[Xj].Discretisation))
Return Average(CompareVectors(V1,i, V2,i))
```

Alternativ können die einzelnen Randwahrscheinlichkeitsverteilungen der Netze N_1 und N_2 jeweils zu einem einzigen Vektor V_1 und V_2 zusammengefasst werden, statt sie durch einen eigenen Vektor zu repräsentieren. Dies ist vor allem dann sinnvoll, wenn einzelne Diskretisierungen nur eine geringe Anzahl von unterschiedlichen Intervallen aufweisen:

```
CompareMarginalProbabilities(N1, N2) // Variante 1a, using data mining tables
                                     // Implies an adaption of the net structures
   Foreach (Ni) do
      Vi = NewVector()
      Foreach (Xj in Ni.Nodes) do
         Foreach(Intervalk in Ni.Node[Xj].Discretisation.Intervals) do
            Vi.Add(Intervalk.Marginalprobability)
Return CompareVectors(V1, V2)

CompareMarginalProbabilities(N1, N2) // Variante 2a, without data mining tables
   Foreach (Ni) do
      Vi = NewVector()
      Foreach (Xj in Ni.Nodes) do
         Y = Copy(Xj)
         Y.Discretisation = Merge(N1.Node[Xj].Discretisation,  N2.Node[Xj].Discretisation)
         Foreach(Intervalk in Y.Discretisation.Intervals) do
            Vi.Add(CalculateMarginalProbability(Intervalk, Ni.Node[Xj].Discretisation))
Return CompareVectors(V1, V2)
```

5.5.5 Vergleich von Evidenzstichproben

Ein praktikables Verfahren zum Vergleich der bedingten Wahrscheinlichkeitsverteilungen zweier probabilistischer Netze N_1 und N_2, ohne Strukturanpassungen der bedingten Wahrscheinlichkeitstabellen (CPT) durchführen zu müssen, ist der Vergleich der Ergebnisse

einer Sequenz von Evidenzstichproben. Meine zugrunde liegende Idee ist, bei beiden Netzen eine große Anzahl von Evidenzkombinationen zu setzen und anschließend die jeweils durch Inferenz berechneten Veränderungen zu vergleichen. Die Veränderungen können dabei anhand der Randwahrscheinlichkeitsverteilungen (Variante 1) oder den darauf basierenden Erwartungswerten der Zufallsvariablen (Variante 2) gemessen werden (siehe Kapitel 2.3.2). Der Vergleich der Randwahrscheinlichkeiten ist zwar genauer, erfordert aber eine Anpassung der Diskretisierungen beider Netze (siehe Kapitel 5.5.3).

Konkret werden an den Knoteninstanzen $X_1, \ldots, X_n$ der Netze N_1 und N_2 jeweils eine Reihe von Evidenzkombinationen $TE_1,\ldots,TE_m$ gesetzt und nach jedem der m Durchgänge entweder die durch Inferenz resultierenden Randwahrscheinlichkeitsverteilungen oder die entsprechenden Erwartungswerte aller Knoten X_i beider Netze verglichen. Die Evidenzkombinationen können dabei sowohl relative als auch absolute Wertvorgaben sowie harte und weiche Evidenzen beinhalten.

Zum Vergleich der Randwahrscheinlichkeitsverteilungen verwende ich die in Kapitel 5.5.4 beschriebene Methode, während die Erwartungswerte der Knoten $X_1, \ldots, X_n$ zum Vergleich als Vektoren der Länge n gespeichert werden, die je nach Fragestellung mit relativen oder absoluten Vektorvergleichsmaßen verglichen werden. Die Gesamtähnlichkeit der beiden Netze ist anschließend der Durchschnittswert der Ähnlichkeiten aller m Durchgänge. Beide Varianten besitzen eine Komplexität von $O(n\cdot|TE|\cdot Z^n)$ und lauten in Pseudocode:

```
CompareEvidenceSampleResults(N1, N2, TE) // Variante 1
   V = NewVector()
   Foreach(TEi) do
      V.Add(CompareMarginalProbabilities(N1.Propagate(SetEvidences(TEi)),
                                         N2.Propagate(SetEvidences(TEi))))
Return Average(V)

CompareEvidenceSampleResults(N1, N2, TE) // Variante 2
   V = NewVector()
   Foreach(TEi) do
      Foreach (Nj) do
         Vj = NewVector()
         Nj.Propagate(SetEvidences(TEi))
         Foreach (Xk in Nj.Nodes) do
            Vj.Add(Xk.ExpectationValue)
      V.Add(CompareVectors(V1, V2))
Return Average(V)
```

Der Vorteil des Vergleichs von Evidenzstichproben liegt darin, dass sich Vergleiche der bedingten Abhängigkeiten zwischen den Zufallsvariablen der Netze durchführen lassen, ohne dass die Wahrscheinlichkeitstabellen angepasst werden müssen. Falls man sich auf den Vergleich der Erwartungswerte beschränkt, kann zusätzlich auf die Anpassung der Diskretisierungen verzichtet werden. Für den Fall, dass die CPT aufgrund fehlender Informationen über die Trainingsmenge nicht exakt angepasst werden können, ist diese Methode die einzig sinnvolle Möglichkeit, um die nichtlinearen Abhängigkeiten zwischen netzinternen Zufallsvariablen zu vergleichen. Ein Nachteil ist die enorme Anzahl möglicher Testevidenzkombinationen, die exponentiell mit der Anzahl der Zufallsvariablen und deren konkreten Zuständen wächst. In der Praxis kann die Komplexität deutlich beschränkt werden, indem beispielsweise eine feste Anzahl von zufällig gewählten Evidenzkombinationen als Vergleichsgrundlage vorgegeben wird.

5.6 Kundenklassifikation mit probabilistischen Netzen

Das Ziel der Kundenklassifikation ist die Zuordnung einzelner Kunden einer Kundenmenge $K = \{K_1, \ldots, K_k\}$ zu *vorgegebenen* Kundengruppen (Klassen) $C_1, \ldots, C_m$. Ein Kunde K_i wird dabei in der Regel der Kundengruppe mit der höchsten Zugehörigkeitswahrscheinlichkeit zugeordnet. Alternativ kann er Mitglied sämtlicher Kundenklassen sein, zu welchen seine Zugehörigkeitswahrscheinlichkeit höher als ein vorgegebener Schwellenwert ist (beispielsweise ab einer Wahrscheinlichkeit von mehr als 80 Prozent). Im alternativen Fall ist es somit möglich, dass ein Kunde entweder keiner, genau einer oder mehreren Kundengruppen C_j zugeordnet wird.

In der Literatur sind viele unterschiedliche Klassifikationsverfahren bekannt, die sich unter anderem in *Verfahren mit Klassifikationsmodellen* und *instanzbasierte Verfahren* gliedern lassen.

In Kapitel 5.6.1 beschreibe ich bekannte probabilistische Klassifikationsmodelle, mit deren Hilfe die Gruppenzugehörigkeiten von einzelnen Kunden berechnet werden können, die durch Attributvektoren beschrieben sind. Ich erweitere die Modelle um die Verwendung der in Kapitel 5.4 vorgestellten verhaltensbezogenen bzw. simulationsbasierten Attributvektoren.

In Kapitel 5.6.2 erläutere ich verschiedene instanzbasierte Klassifikationsverfahren, die die Gruppenzugehörigkeiten einzelner Kunden durch den Vergleich mit gruppenbeschreibenden Prototypen bzw. Instanzen ermitteln. Als Kundenbeschreibung und Vergleichsgrundlage verwende ich dabei sowohl klassische und verhaltensbezogene Attributvektoren als auch die in dieser Arbeit entworfenen probabilistischen Verhaltensnetze.

5.6.1 Probabilistische Kundenklassifikationsmodelle

Verfahren mit Klassifikationsmodellen bestehen aus zwei Prozessschritten. In der ersten Phase wird ein Modell (Klassifikationsmodell) aus einer vorgegebenen Trainingsmenge bereits klassifizierter Kunden erlernt, mit dessen Hilfe anschließend in der zweiten Phase neue Kunden klassifiziert werden. Diese Vorgehensweise ist auch bekannt als *supervised learning*.

Neben *Künstlichen Neuronalen Netzen* (*KNN*) (siehe Kapitel 2.3.5.4) und *Support-Vektor-Maschinen* (*SVM*) [HTF01] werden probabilistische Netze, in Form von Naive-Bayes-Modellen und vollständigen Bayes'schen Netzen, als Klassifikationsmodelle verwendet. KNN und SVM besitzen dabei gegenüber probabilistischen Netzen Nachteile. Im Falle klassischer SVM ist die Anzahl der vorgegebenen Klassen auf zwei beschränkt. Um Kunden bezüglich mehrerer Kundengruppen zu klassifizieren, müssen hierarchische SVM verwendet werden. KNN eignen sich in der Praxis sehr gut zur Klassifikation mit mehreren vorgegebenen Klassen, leiden allerdings unter dem so genannten „Blackboxverhalten". Das bedeutet, dass die Ermittlung der Klassenzugehörigkeit für Menschen meist nicht nachvollziehbar ist, da die Neuronen der Hidden-Layer keine Semantik besitzen.

Bei der *Naive-Bayes-Klassifikation* wird mit Hilfe einer Trainingsmenge ein einfaches Klassifikationsmodell basierend auf bedingten Wahrscheinlichkeiten erlernt, das anschließend zur Klassifikation von unklassifizierten Kunden bezüglich vorgegebener Klassen $\{C_1,\ldots,C_m\}$ dient. Ein Kunde wird durch einen v-dimensionalen Attributvektor $AV = (a_1, \ldots, a_v)$ beschrieben, dessen Werte a_i die konkreten kundenindividuellen Ausprägungen bezüglich der Attribute A_i angeben. Dabei wird die vereinfachende Annahme gemacht, dass die Ausprägungen der verschiedenen Attribute *unabhängig* voneinander sind, was in der Praxis meist nicht gültig ist.

Zur Klassifikation eines Kunden X_i mit Attributvektor AV_i wird für jede Klasse C_j die Zugehörigkeitswahrscheinlichkeit $P(C_j|AV_i)$ berechnet, wobei der Kunde anschließend der Klasse mit der höchsten Zugehörigkeitswahrscheinlichkeit zugeordnet wird, d. h. X_i gehört zur Klasse C_j genau dann, wenn $P(C_j|AV_i) > P(C_r|X)$ für $1 \leq r \leq m$, $j \neq r$, wobei man $P(C_j|AV_i)$ als *Maximum Posterior Hypothesis* bezeichnet.

Zur Berechnung der Zugehörigkeitswahrscheinlichkeiten wird das Bayes'sche Theorem verwendet:

$$P(C_j \mid AV_i) = \frac{P(AV_i \mid C_j) \cdot P(C_j)}{P(AV_i)},$$

Da $P(AV_i)$ für alle Klassen C_j konstant ist, muss nur der Term $P(AV_i|C_j) \cdot P(C_j)$ maximiert werden, wobei $P(C_j)$ durch

$$P(C_j) = \frac{s_j}{s}$$

approximiert wird (s_j steht dabei für die Anzahl der Trainingsfälle, die zur Klasse C_j gehören, während s die Anzahl der Trainingsfälle angibt). Aufgrund der Annahme, dass die Attributausprägungen $a_1, \ldots, a_v$ unabhängig voneinander sind, kann $P(AV_i|C_j)$ wie folgt ermittelt werden:

$$P(AV_i \mid C_j) = \prod_{l=1}^{n} P(a_l \mid C_j), \text{ mit } P(a_l \mid C_j) = \frac{s_{jl}}{s_j},$$

wobei s_{jl} die Anzahl der Trainingsfälle ist, die zu C_j gehören und deren Ausprägung bezüglich des Attributs A_l den Wert a_l besitzt, und s_j die Anzahl der Trainingsfälle ist, die zur Klasse C_j gehören. Falls A_l einen stetigen Wertebereich besitzt, muss dieser vor dem Erlernen des Klassifikationsmodells entsprechend diskretisiert werden (siehe Kapitel 3.3.2).

Alternativ kann ein einfaches Bayes'sches Netz mit je einem diskreten Knoten für jedes der v Attribute und einem einzelnen Klassenknoten (siehe Abbildung 49) als Klassifikationsmodell verwendet werden, wobei jeder Attributknoten ausschließlich den Klassenknoten als Elternknoten besitzt, da die Attribute als voneinander unabhängig angenommen werden. Die Werte der Apriori- bzw. bedingten Wahrscheinlichkeitstabellen werden anhand der Attributausprägungen und Klassenzugehörigkeiten der Trainingsfälle erlernt.

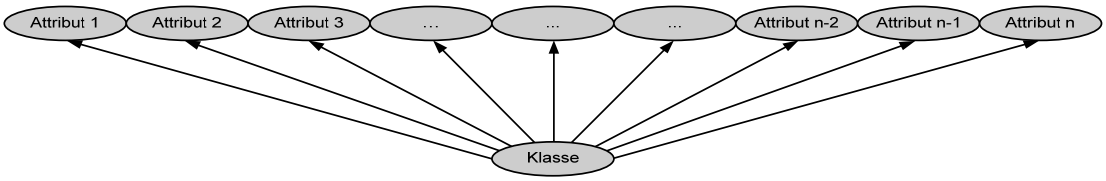


Abbildung 49: Naive-Bayes-Klassifizierung mit einem einfachen Bayes'schen Netz

Zur Klassifikation werden die Ausprägungen des zu klassifizierenden Kunden als Evidenzen in den Attributknoten eingegeben, woraufhin die Klassenzugehörigkeit durch Inferenz berechnet wird und sich als Wahrscheinlichkeitsverteilung im Klassenknoten ablesen lässt (der Kunde gehört dann zu der Klasse, deren entsprechende Randwahrscheinlichkeit am größten ist). Alternativ kann für jede Klasse C_j ein eigener binärer Klassenknoten verwendet werden, der die Zugehörigkeit des Kunden repräsentiert. Auf diese Weise kann die Zugehörigkeitswahrscheinlichkeit des Kunden zu allen Klassen berechnet werden, wobei er einer Klasse zugeordnet wird, wenn die entsprechende Randwahrscheinlichkeit einen vorgegebenen Schwellenwert übersteigt (z. B. 80 Prozent).

Die Klassifikation mit Hilfe des Naive-Bayes-Verfahrens setzt die Unabhängigkeit der Attribute eines Kunden voraus, was in der Praxis in vielen Fällen nicht gewährleistet ist. Um entsprechende Abhängigkeiten zuzulassen, kann das vereinfachte Naive-Bayes-Modell zu baumartigen Naive-Bayes-Modellen (Tree Augmented Naive-Bayes) und darüber hinaus zu vollständigen Bayes'schen Netzen erweitert werden [FGG97], die auch Abhängigkeiten zwischen Attributen repräsentieren können. Einen ausführlichen Vergleich der unterschiedlichen Modelle bietet beispielsweise [CG99].

Abbildung 50 zeigt ein Klassifikationsmodell, das auf einem vollständigen Bayes'schen Netz basiert.

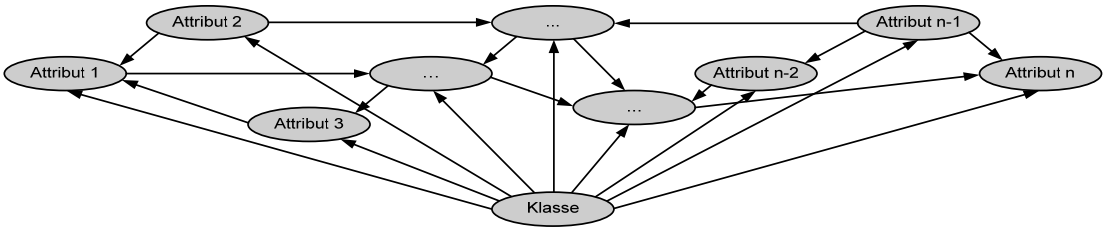


Abbildung 50: Klassifizierung mit Hilfe eines vollständigen Bayes'schen Netzes

Das Netz wird mit Hilfe der Trainingsfälle erlernt, für die sowohl die Attributausprägungen als auch die konkreten Klassen bekannt sind. Die Klassenzugehörigkeit eines unklassifizierten Kunden kann anschließend durch Inferenz ermittelt werden, indem die Attributausprägungen des Kunden als Evidenzen in das Netz eingegeben und nach deren Propagierung die Wahrscheinlichkeitsverteilung des Klassenknotens betrachtet werden. Der Kunde wird dann der Klasse zugeordnet, deren Randwahrscheinlichkeit am größten ist. Alternativ ist es wiederum möglich, für jede Klasse einen eigenen binären Klassenknoten zu verwenden, um die Zugehörigkeitswahrscheinlichkeit des Kunden zu allen Klassen zu berechnen.

Die Attributvektoren können sowohl beim Naive-Bayes-Verfahren als auch bei der Verwendung vollständiger Bayes'scher Netze nicht nur einfache soziodemographische und statistische Daten (siehe Kapitel 5.3), sondern auch dynamische, verhaltensbezogene Merkmale beinhalten, deren kundenindividuellen Ausprägungen in einem vorgelagerten Schritt mit Hilfe entsprechender Verhaltensnetze ermittelt werden (siehe Kapitel 5.4). Als Merkmale der Attributvektoren können darüber hinaus auch konkrete Ausprägungen der kundenindividuellen Verhaltensnetze verwendet werden, die im Rahmen der von mir vorgestellten direkten Vergleichsverfahren berechnet werden (siehe Kapitel 5.5).

Die algorithmische Vorgehensweise zur Klassifikation einer Menge von Kunden $K = \{K_1, \ldots, K_k\}$ bezüglich vorgegebener Klassen $C = \{C_1, \ldots, C_m\}$ mit Hilfe eines Klassifikationsmodells *CM*, basierend auf einer Trainingsmenge $T = \{T_1, \ldots, T_t\}$ aus bereits

klassifizierten Kunden, die durch Attributvektoren der Form $AV = (a_1, \ldots, a_v)$ beschrieben sind, lautet in Pseudocode:

```
ClassifyCustomersWithClassifier(C, T, K) // Variante 1
   Foreach (Ti in T) do
      Ti.AV = CreateAttributVector(v) // using data from DB or calculate complex properties
   CM = NewClassifier(n) // Naïve-Bayes or complete Bayesian network with n attribute nodes
                         // and a single class node
   CM.LearnFrom(T)
   Foreach (Ki in K) do
      Ki.AV = CreateAttributVector(v) // using data from DB or calculate complex properties
      Ki.Class = CM.Classify(Ki)
Return K
```

Diese Vorgehensweise besitzt, im Falle naiver Bayes'scher Netze und bereits vorliegenden Attributvektoren, eine Komplexität von $O((|T|+|K|)\cdot n\cdot Z^2)$, die sich bei der Verwendung vollständiger Bayes'scher Netze auf $((|T|+|K|)\cdot n\cdot Z^n)$ erhöht (n ist die von v abhängige Anzahl der Netzknoten). Ein Kunde K_i wird dabei der Klasse C_j mit der höchsten Randwahrscheinlichkeit zugeordnet. Alternativ kann ein Kunde zu mehreren Klassen gehören, wenn die entsprechende Ähnlichkeit einen gewissen Schwellenwert (threshold) übersteigt, wodurch sich die Komplexitäten in den beiden Varianten jeweils um den Faktor $|C|$ auf $O((|T|+|K|)\cdot n\cdot|C|\cdot Z^2)$ bzw. $O((|T|+|K|)\cdot n\cdot|C|\cdot Z^n)$ erhöhen:

```
ClassifyCustomersWithClassifier(C, T, K, treshold) // Variante 2
   Foreach (Ti in T) do
      Ti.AV = CreateAttributVector(v) // using data from DB or calculate complex properties
   CM = NewDetailedClassifier(n, m) // Naïve-Bayes or complete Bayesian network
                                    // with n attribute nodes and m class nodes
   CM.LearnFrom(T)
   Foreach (Ki in K) do
      Ki.AV = CreateAttributVector(v) // using data from DB or calculate complex properties
      Foreach (Cj in C) do
         If (CM.Classify(Ki, Cj) >= treshold) then Ki.MemberOfClass(Cj)
Return K
```

5.6.2 Instanzbasierte Kundenklassifikation

Bei instanzbasierten Kundenklassifikationsverfahren werden im Vorfeld keine Klassifikationsmodelle gelernt. Stattdessen werden die Kunden $K = \{K_1, \ldots, K_k\}$ zur

Laufzeit mit bereits klassifizierten Kunden (Instanzen) $I = \{I_1, \ldots, I_j\}$ verglichen und anschließend der Kundengruppe C_j aus der Menge der vorgegebenen Klassen $C = \{C_1, \ldots, C_m\}$ zugeordnet, die die meisten der ähnlichsten Instanzen besitzt. Eines der bekanntesten instanzbasierten Klassifikationsverfahren ist das *k-nächste-Nachbarn-Verfahren* (*k-Nearest Neighbours*), bei dem ein Kunde der Klasse zugeordnet wird, zu der die Mehrheit seiner k nächsten Instanzen gehört. Die Beschreibung der Kunden und Instanzen basiert dabei üblicherweise auf Attributvektoren der Form $AV = (a_1, \ldots, a_v)$ mit soziodemographischen und/oder einfachen statistischen Kennzahlen, die beispielsweise mit den in Kapitel 5.3 vorgestellten vektorbasierten Ähnlichkeitsmaßen verglichen werden können.

Um Kunden aufgrund ihres dynamischen Kaufverhaltens mit Hilfe instanzbasierter Klassifikationsverfahren zu klassifizieren, können die Attributvektoren auch aus komplexeren Merkmalen sowie aus Ergebnissen von szenariobasierten Verhaltenssimulationen bestehen, die ich in Kapitel 5.4 eingeführt habe. Das auf diesen erweiterten Attributvektoren aufbauende k-nächste-Nachbarn-Verfahren lautet in Pseudocode:

```
ClassifyCustomersWithInstances(C, I, K, k) // Variante 1
   Foreach (Ii in I) do
      Ii.AV = CreateAttributVector(v) // using data from DB or calculate complex properties
   Foreach (Ki in K) do
      Ki.AV = CreateAttributVector(v) // using data from DB or calculate complex properties
      Neighboursi = NewList()
      Foreach (Ij in I) do
         Neighboursi.Add(CompareVectors(Ki.AV, Ij.AV), Ij)
      Neigboursi.SortbySimilarity(DESC)
      Ki.Class = MajorityOfClassesFromInstances(FirstMembersFrom(Neighboursi, k))
Return K
```

Das Verfahren besitzt bei bereits vorliegenden Attributvektoren eine Komplexität von $O(|K|\cdot|I|\cdot|AV|)$. Ich erweitere das Verfahren um probabilistische Verhaltensnetze als Vergleichsgrundlage, indem ich diese anstelle der Attributvektoren zur Beschreibung der einzelnen Kunden und Instanzen verwende. Um Kunden mit Instanzen zu vergleichen, nutze ich die in Kapitel 5.5 eingeführten direkten Vergleichsmaße (was im Worstcase zu einer Komplexität von $O(|K|\cdot|I|\cdot n\cdot Z^n)$ führt, wobei n die Anzahl der Netzknoten ist):

```
ClassifyCustomersWithInstances(C, I, K, k) // Variante 2
   Foreach (Ii in I) do
      Ii.BNet = CreateBehaviourNetwork() // using data from DB
   Foreach (Ki in K) do
```

```
    Ki.BNet = CreateBehaviourNetwork() // using data from DB
    Neighboursi = NewList()
    Foreach (lj in l) do
       Neighboursi.Add(CompareNets(Ki.BNet, lj.BNet), lj)
    Neigboursi.SortbySimilarity(DESC)
    Ki.Class = MajorityOfClassesFromInstances(FirstMembersFrom(Neighboursi, k))
Return K
```

Instanzen können nicht nur bereits klassifizierte Kunden, sondern auch kundengruppenbeschreibende Prototypen sein, die jeweils eine vorgegebene Klasse C_j repräsentieren. Ein Prototyp ist dabei entweder eine durchschnittliche Kundenbeschreibung der Mitglieder von C_j oder ein konkreter „durchschnittlicher" Kunde.

Durchschnittliche metrische Vektoren können einfach durch Mittelwertbildung der kundenindividuellen Attributvektoren gebildet werden, während bei Verwendung von nicht-metrischen Merkmalen der Attributvektor eines konkreten Kunden als Prototyp verwendet wird. Durchschnittliche Verhaltensnetze einer Kundengruppe werden mit Hilfe einer Durchschnittsverschmelzung der kundenindividuellen Verhaltensnetze (siehe Kapitel 3.3.5.1) aller Mitglieder erstellt.

Prototypen können auch *virtuelle* Beschreibungen sein, die die gewünschten Eigenschaften der Kundengruppe C_j vorgeben, ohne dass bereits konkrete Kunden als Mitglieder von C_j bekannt sind. Dabei können beispielsweise Prototypenbeschreibungen verwendet werden, die gängige Kundentypologien mit Hilfe von beobachtbaren Attributen beschreiben. Die Suchkriterien bzw. Attributausprägungen der einzelnen Merkmale a_i der virtuellen Prototypen können dabei sowohl durch konkrete Werte als auch durch *Filter* angegeben werden (vergleiche [GA04]), wie beispielsweise:

- $a_i \rightarrow min$ (minimaler Wert aller untersuchter Kunden bezüglich Attribut A_i)
- $a_i \rightarrow max$ (maximaler Wert aller untersuchter Kunden bezüglich Attribut A_i)
- $a_i \rightarrow avg$ (durchschnittlicher Wert aller untersuchter Kunden bezüglich Attribut A_i)
- $a_i \geq v$ bzw. $a_i > v$ (a_i muss größer oder gleich bzw. größer als ein vorgegebener Wert sein)
- $a_i \leq v$ bzw. $a_i < v$ (a_i muss kleiner oder gleich bzw. kleiner als ein vorgegebener Wert sein)
- $a \leq a_i \leq b$ (a_i muss innerhalb des Intervalls $[a;b]$ liegen)
- $a_i > b \wedge a_i < a$ (a_i muss außerhalb des Intervalls $[a;b]$ liegen)

Alle Attributvektoren, die die entsprechenden Filterungen erfüllen, gehören anschließend zur vorgegebenen Kundenklasse. Alternativ kann im Falle metrischer Variablen auch der Abstand der einzelnen Vektoren zu der Gesamtheit der Suchkriterien bestimmt werden,

indem man die Filterungen mit Hilfe geeigneter Funktionen der Form f_{Filter}: $a_i \rightarrow [0;1]$ auf den Wertebereich [0;1] normiert und als Abstand den durchschnittlichen Wert der Filterfunktionswerte verwendet. Dabei ist es möglich, einzelne Attribute des Ergebnisvektors vor der Durchschnittsbildung mit geeigneten Gewichtungen g_i zu multiplizieren, um bestimmte Merkmale bei der Suche stärker zu gewichten (siehe [GA04]).

Prototypen werden eingesetzt, wenn entweder keine ausreichende Trainingsmenge bereits klassifizierter Kunden zur Verfügung steht oder man eine Kundengruppe aus Performancegründen nur durch einen durchschnittlichen Kunden repräsentieren möchte.

Die Klassifikation einer Menge von Kunden $K = \{K_1, \ldots, K_k\}$ mit vorgegebenen Kundengruppen $C = \{C_1, \ldots, C_m\}$, die jeweils durch einen Prototypen P_j repräsentiert werden, ist ein Spezialfall der instanzbasierten Klassifikation. Ein Kunde K_i wird dabei der Kundengruppe C_j zugeordnet, zu dessen Prototypen P_j er am ähnlichsten ist. Prototypen und Kunden können dabei durch Attributvektoren (Variante 1) oder Verhaltensnetze (Variante 2) beschrieben werden. Variante 1 besitzt bei bereits vorliegenden Attributvektoren eine Komplexität von $O(|K|\cdot|P|\cdot|AV|)$, während die Komplexität der zweiten Variante $O(|K|\cdot|P|\cdot n\cdot Z^n)$ ist. Der Pseudocode beider Versionen lautet:

```
ClassifyCustomersWithPrototypes(C, P, K) // Variante 1, prototypes are attribute vectors
   Foreach (Ki in K) do
      Ki.AV = CreateAttributVector(v) // using data from DB or calculate complex properties
      Neighbours_i = NewList()
      Foreach (Pj in P) do
         Neighbours_i.Add(CompareVectors(Ki.AV, Pj.AV), Pj)
      Neigbours_i.SortbySimilarity(DESC)
      Ki.Class = FirstMemberFrom(Neighbours_i).Class
Return K

ClassifyCustomersWithPrototypes(C, P, K) // Variante 2, prototypes are behaviour nets
   Foreach (Ki in K) do
      Ki.BNet = CreateBehaviourNetwork() // use data from DB
      Neighbours_i = NewList()
      Foreach (Pj in P) do
         Neighbours_i.Add(CompareNets(Ki.BNet, Pj.BNet), Pj)
      Neigbours_i.SortbySimilarity(DESC)
      Ki.Class = FirstMemberFrom(Neighbours_i).Class
Return K
```

Alternativ können die einzelnen Kunden K_i auch allen Klassen C_j zugeordnet werden, zu

deren Prototypen P_j die Ähnlichkeit größer als ein vorgegebener Schwellenwert (treshold) ist, wobei sich die Komplexität in beiden Fällen nicht erhöht:

```
ClassifyCustomersWithPrototypes(C, P, K, treshold) // Variante 1a
   Foreach (Ki in K) do
      Ki.AV = CreateAttributVector(v) // using data from DB or calculate complex properties
      Foreach (Pj in P) do
         If (CompareVectors(Ki.AV, Pj.AV) >= threshold) then Ki.MemberOfClass(Pj.Class)
Return K

ClassifyCustomersWithPrototypes(C, P, K, treshold) // Variante 2a
   Foreach (Ki in K) do
      Ki.BNet = CreateBehaviourNetwork() // using data from DB
      Foreach (Pj in P) do
         If (CompareNets(Ki.BNet, Pj.BNet) >= threshold) then Ki.MemberOfClass(Pj.Class)
Return
```

5.7 Kundenclustering mit probabilistischen Netzen

Das Ziel eines Kundenclusterings ist das Einteilen einer Kundenmenge $K = \{K_1, \ldots, K_n\}$ in k Klassen (Cluster) $C = \{C_1, \ldots, C_k\}$, wobei Kunden eines Clusters möglichst ähnlich und Kunden aus verschiedenen Clustern möglichst unterschiedlich sein sollen [HK01]. Der Unterschied zur Klassifikation besteht darin, dass beim Clustering keine Klassen oder Trainingsmengen vorgegeben werden. Das bedeutet, die Aufgabe eines Clusteringverfahrens besteht sowohl aus der Ermittlung geeigneter Klassen und deren konkreten Anzahl k als auch aus der Bestimmung der Clusterzugehörigkeiten der einzelnen Kunden K_i. Wesentlich ist dabei die Entdeckung unterschiedlicher Merkmale bzw. Verhaltensweisen zwischen den Kundenclustern. Clustering ist ein Beispiel für *unsupervised learning*, während Klassifikationsverfahren mit Trainingsmengen dem *supervised learning* entsprechen.

In der Literatur existiert eine enorme Anzahl unterschiedlicher Clusteringverfahren, die anhand verschiedener Kriterien untergliedert werden können. [HK01] verwendet beispielsweise folgende Kategorisierung:

- Partitionierungsmethoden,
- Hierarchische Methoden,
- Dichtebasierte Methoden,
- Gridbasierte Methoden und
- Modellbasierte Methoden.

Bei Partitionierungsmethoden werden die zu klassifizierenden n Kunden anfänglich meist zufällig in k Partitionen aufgeteilt, mit $k \leq n$, wobei jede Partition mindestens einen Kunden beinhalten muss und ein Kunde nur zu genau einer Partition gehören darf (bei der Klasse der *Fuzzy-Partitionierungsverfahren* (wie beispielsweise *EM-Clustering*) können Kunden gleichzeitig mehreren Partitionen angehören). Anschließend werden die Kunden in einer Schleife solange zwischen den Partitionen verschoben (*iterative relocation technique*) bis die Güte der Partitionierung einen vorgegebenen Schwellenwert erreicht hat. Die Güte wird dabei umso besser, je ähnlicher sich Kunden innerhalb der einzelnen Partitionen sind und je stärker sich die Partitionen unterscheiden. Es existiert eine Vielzahl unterschiedlicher Kriterien zur Bestimmung der Güte einer Partitionierung. Da zur Erzeugung einer optimalen Partitionierung alle möglichen Partitionierungskombinationen erstellt und bewertet werden müssten, bedienen sich die meisten Partitionierungsmethoden einer der beiden Heuristiken *k-Means* und *k-Medoid*, bei denen jede Partition durch einen Mittelwert der Mitgliedereigenschaften repräsentiert wird. Daher eignen sie sich zur Erzeugung sphärischer Partitionen, d. h. die Mitglieder einer Partition liegen „kreis- bzw. kugelförmig“ um den Mittelwert.

Hierarchische Clusteringmethoden führen eine hierarchische Dekomposition einer Kundenmenge durch, wobei zwischen *bottom-up* bzw. *agglomerativen* und *top-down* bzw. *teilenden* Dekompositionen unterschieden wird. Bei der bottom-up Dekomposition stellt anfangs jeder Kunde eine separate Gruppe dar, um anschließend ähnliche Gruppen stufenweise so lange zu vereinen bis die höchste Hierarchiestufe erreicht oder eine Abbruchbedingung erfüllt wurde. Bei der top-down Dekomposition sind alle Kunden zu Beginn in einer einzigen Gruppe, die schrittweise in Untergruppen zerteilt wird, bis diese nur noch aus einzelnen Kunden bestehen oder entsprechende Terminierungskriterien erfüllt werden. Ein Nachteil der hierarchischen Verfahren besteht darin, dass sie eine eventuell fehlerhafte Entscheidung innerhalb eines Dekompositionsschrittes nicht wieder rückgängig machen können [HK01]. Aus diesem Grund existieren verschiedene Ansätze, um die Wahrscheinlichkeit einer Fehlentscheidung zu minimieren. Eine Möglichkeit besteht darin, in jedem Iterationsschritt zu analysieren, inwieweit Verbindungen zwischen einzelnen Kunden bzw. Gruppen existieren und diese in den Entscheidungsprozess einzubeziehen (Beispiele für solche Verbindungskriterien sind *Nearest Neighbour* bzw. *Furthest Neighbour*). Eine andere Möglichkeit ist die Integration der bottom-up Dekomposition und der Iterative-Relocation-Technik der Partitionierungsverfahren, bei der das Ergebnis nach einer Dekomposition durch nachträgliches Verschieben von Kunden verbessert wird. Eine der bekanntesten hierarchischen Methoden ist das so genannte BIRCH-Verfahren [ZRL96].

Dichtebasierte Methoden erlauben die Erzeugung von nichtsphärischen Clustern, d. h. dass die Mitglieder eines Clusters nicht unbedingt „kreis- oder kugelförmig“ um den Mittelwert

des Clusters liegen müssen. Dichtebasierte Verfahren erweitern bestehende Cluster iterativ so lange bis die Anzahl der Kunden ("Kundendichte") in der Nachbarschaft einen gewissen Grenzwert erreicht hat. Das bedeutet, dass für jeden Datenpunkt in einem Cluster innerhalb eines bestimmten Radius eine Mindestanzahl von Kunden vorliegen muss. Auf diese Weise ist es möglich, neben sphärischen auch beliebige andere Formen von Clustern zu erzeugen.

Gridbasierte Methoden modellieren den Beschreibungsraum durch eine endliche Anzahl von Zellen eines mehrdimensionalen Würfels, wodurch die Laufzeit der entsprechenden Algorithmen nicht von der Anzahl der Kunden, sondern nur von der Anzahl der Zellen im Beschreibungsraum abhängig ist. Ein Beispiel für ein gridbasiertes Verfahren ist die CLIQUE-Methode [AGGR98], die Gemeinsamkeiten in Unterräumen entdecken kann.

Bei modellbasierten Methoden wird für jedes Cluster ein individuelles Modell angenommen und im Anschluss nach Kunden gesucht, die am besten zu diesem Modell passen. Modellbasierte Methoden umfassen in vielen Fällen Dichtefunktionen, um automatisch eine geeignete Anzahl von Clustern zu ermitteln oder „Ausreißer" zu erkennen. Modellbasierte Methoden verwenden meist statistische Verfahren oder neuronale Netze.

Bei allen Clusteringverfahren, die zu einer oder mehreren der oben genannten Kategorien gehören und vektorbasierte Kundenbeschreibungen verwenden, können neben soziodemographischen und statistischen Kennzahlen wiederum komplexere, verhaltensbezogene Merkmale und simulierte Kundenreaktionen in den Attributvektoren verwendet werden, die in einem vorgelagerten Schritt mit Hilfe entsprechender Verhaltensnetze berechnet werden (siehe Kapitel 5.4).

Ebenso können bei einigen Verfahren, die während des Clusterings vektorbasierte Ähnlichkeitsmaße zum Vergleich der Attributvektoren verwenden, Verhaltensnetze als Beschreibungsgrundlage einzelner Kunden und Kundengruppen dienen, die mit Hilfe der in Kapitel 5.5 eingeführten Ähnlichkeitsmaße verglichen werden.

Im Folgenden werden die Partitionierungsverfahren *k-Means* und *k-Medoid* beschrieben, die ich zunächst um komplexe, verhaltensbezogene Attributvektoren und anschließend um Verhaltensnetze als Kundenbeschreibungs- und Vergleichsgrundlage erweitere, damit sie zur verhaltensbezogenen Entdeckung gleichartiger Kunden verwendet werden können.

5.7.1 *k*-Means und *k*-Medoid mit erweiterten Attributvektoren

Eine der bekanntesten Partitionierungsmethoden ist *k*-Means [HJA75], bei der alle Cluster durch ihren Mittelwert (Mean) repräsentiert werden. Die generelle Vorgehensweise dabei ist:

1. Auswahl von *k* Kunden als anfängliche Cluster-Means
2. Wiederhole bis definierte Abbruchbedingung erfüllt ist:
    a. Ordne jeden Kunden dem Cluster zu, dessen Mean er am ähnlichsten ist
    b. Neuberechnung aller Cluster-Means

Die Abbruchbedingung ist erfüllt, wenn entweder keine Verschiebungen mehr stattfinden oder eine geeignete Gütefunktion konvergiert. Meist wird dabei das folgende *Squared-Error-Kriterium* verwendet:

$$E = \sum_{i=1}^{k} \sum_{c \in C_i} | c - m_i |^2$$

E ist dabei die Summe der quadrierten Fehler, während c die Kunden und m_i den Mittelwert der jeweiligen Cluster repräsentieren, wobei sowohl c als auch m_i multidimensional sind. Ziel ist es, den Wert von E zu minimieren, da auf diese Weise die resultierenden Cluster gleichzeitig kompakt und separierend werden [HK01]. *k*-Means eignet sich zum Partitionieren großer Datenmengen, da die Laufzeit bei bereits vorliegenden Attributvektoren $O(|K| \cdot k \cdot i \cdot |AV|)$ beträgt, wobei i die Anzahl der Iterationen und $|AV|$ die Länge der Attributvektoren ist. Die im Rahmen meiner Arbeit entstandene Erweiterung von *k*-Means um komplexere Attributvektoren als Beschreibungs- und Vergleichsgrundlage lautet in Pseudocode:

```
KMeansClusteringWithAttributVectors(K, k, threshold)
   Foreach (Ki in K) do
      Ki.AV = CreateAttributVector(v) // using data from DB or calculate complex properties
   C = NewClusterList(k)
   Foreach(Ci in C) do
      M = SelectRandomFrom(K)
      Ci.Members.Add(M)
      Ci.Mean = M.AV
   Changed = NewBoolean(true)
   While ((CalculateSquaredError(C, K) >= threshold) AND (Changed = true)) do
      Changed = false
      Foreach (Ki in K) do
         Clusteri = NewList()
         Foreach (Cj in C) do
            Clusteri..Add(CompareVectors(Ki.AV, Cj.Mean), Cj)
         Clusteri.SortBySimilarity(DESC)
         Ki.Class = FirstMemberFrom(Clusteri)
         If (Not(IsMember(Ki, Ki.Class.Members))) then
            Changed = true
            Ki.Class.Members.Add(Ki)
            DeleteMember(Ki, Ki.OldClass)
```

```
    Foreach (Cj) do
        Cj.CalculateNewMean()
Return C, K
```

k-Means ist allerdings nur anwendbar, wenn für jedes Cluster tatsächlich immer ein Mean berechnet werden kann, was bei nicht-metrischen bzw. kategorischen Variablen innerhalb der Attributvektoren nicht ohne weiteres möglich ist. Um diese Einschränkung zu beheben, ersetzt ein unter *k-Modes* bekannter Ansatz die Means der Cluster durch so genannte *Modes*, die mit kategorischen Variablen umgehen können, und passt die Modes mit Hilfe von frequenzbasierten Methoden nach jeder Iteration an. Eine Kombination von *k*-Means und *k*-Modes ist *k-Prototypes*, mit der Objekte geclustert werden können, die sowohl durch numerische als auch durch kategorische Variablen beschrieben werden.

Alle drei Ansätze haben den gemeinsamen Nachteil, dass die Ermittlung der Means bzw. Modes durch „Ausreißer" in der Kundenmenge stark beeinflusst wird und nur mehr oder weniger sphärische Cluster gefunden werden können.

Um diesen Nachteil zu umgehen, wird ein Cluster beim *k*-Medoid-Verfahren nicht durch den Mean, sondern durch den am zentralsten liegenden Kunden repräsentiert (so genannter *Medoid*). Auf diese Weise ist *k*-Medoid nicht so anfällig gegenüber „Ausreißern" wie *k*-Means und lässt sich auch bei nicht-metrischen Vektoren anwenden, bei denen der Mittelwert nicht ohne weiteres berechnet werden kann. Aufgrund der Ermittlung der Medoids in jeder Iteration erhöht sich die Komplexität auf $O(|K|^2 \cdot k \cdot i \cdot |AV|)$. Die im Rahmen dieser Arbeit entwickelte und um verhaltensbezogene Attributvektoren erweiterte Variante des *k*-Medoid-Verfahrens lautet in Pseudocode:

```
KMedoidClusteringWithAttributVectors(K, k, threshold)
  Foreach (Ki in K) do
    Ki.AV = CreateAttributVector(v) // using data from DB or calculate complex properties
  C = NewClusterList(k)
  Foreach(Ci in C) do
    M = SelectRandomFrom(K)
    Ci.Members.Add(M)
    Ci.Medoid = M
  Changed = NewBoolean(true)
  While ((CalculateSquaredError(C, K) >= threshold) AND (Changed = true)) do
    Changed = false
    Foreach (Ki in K) do
      Clusteri = NewList()
      Foreach (Cj in C) do
        Clusteri..Add(CompareVectors(Ki.AV, Cj.Medoid.AV), Cj)
```

```
        Cluster_i.SortBySimilarity(DESC)
        K_i.Class = FirstMemberFrom(Cluster_i)
        If (Not(IsMember(K_i, K_i.Class.Members))) then
            Changed = true
            K_i.Class.Members.Add(K_i)
            DeleteMember(K_i, K_i.OldClass)
    Foreach (C_j) do
        C_j.SelectNewMedoid()
Return C, K
```

Sowohl *k*-Means als auch *k*-Medoid setzen ein konkretes *k* voraus. Das bedeutet, dass die Anzahl der Cluster bzw. Klassen im Vorhinein angegeben werden muss. In der Praxis ist es allerdings nicht immer ohne weiteres möglich, ein geeignetes *k* anzugeben, da die Ermittlung eines möglichst optimalen *k* ein wesentlicher Teil der Aufgabenstellung ist. Eine Strategie, um gute Resultate zu erzielen ist, vor der Partitionierung entweder eine agglomerative hierarchische Clustering-Methode (wie beispielsweise beim BIRCH-Verfahren) oder eine *Kreuzvalidierung* (*Cross-Validation)* anzuwenden, um sowohl ein geeignetes *k* als auch anfängliche Clustermittelwerte zu berechnen.

Bei der Kreuzvalidierung wird die Menge der Kunden in eine *Trainingsmenge* und in eine *Testmenge* aufgeteilt. Auf Grundlage der Testmenge wird ein Clustering-Modell erzeugt, das anschließend mit Hilfe der Testmenge evaluiert wird. Eine Variante der Kreuzvalidierung ist die *v-fache Kreuzvalidierung* (*v-fold cross validation*), bei der die Menge der Kunden in *v* Untergruppen unterteilt wird, von denen jede einmal als alleinige Testmenge verwendet wird, während die übrigen Untergruppen als Trainingsmenge dienen, d. h. jede Untergruppe ist *v*-1 mal Teil der Trainingsmenge. Zur Evaluation werden im Fall von *k*-Means oder *k*-Medoid meist die durchschnittlichen Distanzen der Testkunden zu den Clustermittelpunkten herangezogen. Bei der *v*-fachen Kreuzvalidierung werden die Ergebnisse der *v* Durchgänge zur Evaluierung entsprechend gemittelt. Zur Ermittlung eines geeigneten *k* wird die Kreuzvalidierung bzw. die *v*-fache Kreuzvalidierung für verschiedene *k* durchgeführt und anschließend das *k* mit dem besten Evaluationsergebnis gewählt. Da im Customer Relationship Management in der Praxis selten mehr als 20 verschiedene Kundengruppen unterschieden werden, lässt sich die Ermittlung eines geeigneten Wertes in der Praxis in vielen Fällen auf $k \leq 20$ beschränken.

5.7.2 Erweiterung von *k*-Means und *k*-Medoid um Verhaltensnetze

Sowohl *k*-Means als auch *k*-Medoid sind so allgemein gehalten, dass die zugrunde liegenden Kundenbeschreibungen und entsprechenden Ähnlichkeitsmaße ausgetauscht werden können. So ist es möglich, anstelle der üblichen Attributvektoren und vektorbasierten

Ähnlichkeitsmaße, probabilistische Verhaltensnetze und die in dieser Arbeit eingeführten direkten Vergleichsverfahren zu verwenden, wodurch Kundenclusterings auf komplexen, verhaltensbezogenen Merkmalen ermöglicht werden.

Bei *k*-Mediod können die Attributvektoren einfach durch entsprechende Verhaltensnetze ersetzt werden, da die Clustermittelpunkte durch konkrete Kunden repräsentiert werden. Die verhaltensnetzbasierte Variante von *k*-Mediod mit einer Komplexität von $O(|K|^2 \cdot k \cdot i \cdot n \cdot Z^n)$ lautet in Pseudocode:

```
KMedoidClusteringWithBehaviourNetworks(K, k, threshold)
   Foreach (Ki in K) do
      Ki.BNet = CreateBehaviourNetwork() // using data from DB
   C = NewClusterList(k)
   Foreach(Ci in C) do
      M = SelectRandomFrom(K)
      Ci.Members.Add(M)
      Ci.Medoid = M
   Changed = NewBoolean(true)
   While ((CalculateSquaredError(C, K) >= threshold) AND (Changed = true)) do
      Changed = false
      Foreach (Ki in K) do
         Clusteri = NewList()
         Foreach (Cj in C) do
            Clusteri..Add(CompareNets(Ki.BNet, Cj.Medoid.BNet), Cj)
         Clusteri.SortBySimilarity(DESC)
         Ki.Class = FirstMemberFrom(Clusteri)
         If (Not(IsMember(Ki, Ki.Class.Members))) then
            Changed = true
            Ki.Class.Members.Add(Ki)
            DeleteMember(Ki, Ki.OldClass)
      Foreach (Cj) do
         Cj.SelectNewMedoid()
Return C, K
```

Bei *k*-Means besteht jedes Cluster-Mean, im Falle der Verwendung probabilistischer Verhaltensnetze als Beschreibungsgrundlage, aus dem Durchschnitt aller kundenindividuellen Verhaltensnetze der Clustermitglieder. Dieser kann durch die in Kapitel 2.4 vorgestellte holonische Durchschnittsverschmelzung (siehe Kapitel 3.3.5.1) berechnet werden. Die auf diese Weise erweiterte Version von *k*-Means besitzt eine Komplexität von $O(|K| \cdot k \cdot i \cdot n \cdot Z^n)$ und lautet in Pseudocode:

```
KMeansClusteringWithBehaviourNetworks(K, k, threshold)
   Foreach (Ki in K) do
      Ki.BNet = CreateBehaviourNetwork() // using data from DB
   C = NewClusterList(k)
   Foreach(Ci in C) do
      M = SelectRandomFrom(K)
      Ci.Members.Add(M)
      Ci.Mean = M.BNet
   Changed = NewBoolean(true)
   While ((CalculateSquaredError(C, K) >= threshold) AND (Changed = true)) do
      Changed = false
      Foreach (Ki in K) do
         Clusteri = NewList()
         Foreach (Cj in C) do
            Clusteri.Add(CompareNets(Ki.BNet, Cj.Mean), Cj)
         Clusteri.SortBySimilarity(DESC)
         Ki.Class = FirstMemberFrom(Clusteri)
         If (Not(IsMember(Ki, Ki.Class.Members))) then
            Changed = true
            Ki.Class.Members.Add(Ki)
            DeleteMember(Ki, Ki.OldClass)
      Foreach (Cj) do
         Cj.GenerateAverageBNet() // using AverageMerging
Return C, K
```

5.8 Kundensegmentierung

Unter Kundensegmentierung verstehe ich die Aufteilung der Menge der Kunden $K = \{K_1, \ldots, K_n\}$ eines Marktes in eine Partitionierung von Kundengruppen $C = \{C_1, \ldots, C_m\}$, mit $K = C_1 \cup \ldots \cup C_m$ und $C_i \cap C_j = \varnothing$, bezogen auf einen statischen Zeitraum T. Kundensegmentierungen dienen beispielsweise zur Beantwortung der Frage, aus welchen Kundengruppen sich die Menge der Kunden an einem bestimmten Tag oder einer bestimmten Woche zusammensetzt und wie deren Verteilung ist. Mit Hilfe dieser Erkenntnis lassen sich Maßnahmen des Marketings besser auf den jeweiligen Markt anpassen, da die Menge der positiv bzw. negativ betroffenen Kunden im Vorfeld abgeschätzt werden kann.

Kundensegmentierungen können auf unterschiedliche Weise mit Hilfe von Klassifikations- und Clusteringverfahren ermittelt werden. Im Folgenden werden zwei generelle

Verfahrensweisen zur Kundensegmentierung beschrieben, die sich in der zur Verfügung stehenden Datengrundlage unterscheiden. Im ersten Fall nehme ich an, dass Kundendaten in Form von soziodemographischen Informationen und kundenbezogenen Einkaufshistorien vorliegen (Unterkapitel 5.8.1). Im zweiten Fall erläutere ich einen Ansatz zur Durchführung von Kundensegmentierungen auf Basis von anonymen Kassenbonanalysen (Unterkapitel 5.8.2). Im letzteren Fall können Erkenntnisse und Prognosen, die mit Hilfe von Kundengruppensimulationen ermittelt wurden, auf den gesamten Markt hochgerechnet werden, da die Verteilung der simulierten Kundengruppen durch Kassenbonsegmentierungen abgeschätzt werden kann.

5.8.1 Kundensegmentierung mit Kundendaten

Falls für eine ausreichende Anzahl von einzelnen Kunden individuelle soziodemographische Informationen und Einkaufshistorien vorliegen, kann eine Kundensegmentierung bezogen auf einen Zeitraum auf drei verschiedene Arten durchgeführt werden:

- Normierung der Kundenanteile bei Kenntnis eindeutiger Gruppenzugehörigkeiten,
- Klassifikation mit eindeutiger Zuordnung sowie
- Clustering mit eindeutiger Zuordnung.

Falls für jeden Kunden K_i aus K, der in einem bestimmten Zeitraum T eingekauft hat, Informationen über eine eindeutige Gruppenzugehörigkeit zu einer der vorgegebenen Kundengruppen C_i aus C vorliegen, dann kann eine Kundensegmentierung berechnet werden, indem die Anteile aller definierten Kundengruppen durch jeweilige Ermittlung der Mitgliederanzahl und anschließendes Normieren ermittelt werden. Die Gruppenzugehörigkeit der Kunden ist dabei entweder durch bereits durchgeführte Klassifikationen oder Clusterings bekannt, oder kann durch einfache Datenbankabfragen ermittelt werden (z. B. bezüglich umsatz- oder ertragsbezogenen ABC-Einteilungen, oder bezüglich soziodemographischen Merkmalen wie Einkommen, Familienstand, Wohnort und Geschlecht).

Liegt beispielsweise für jeden Kunden eine eindeutige ABC-Zuordnung vor, kann die Verteilung der drei Klassen A, B und C bezüglich des Zeitraumes T berechnet werden, indem für jede Klasse der Anteil der Kunden ermittelt wird, die Mitglied der entsprechenden Klasse sind und im vorgegebenen Zeitraum T einkaufen waren.

Die Komplexität dieser Vorgehensweise hängt hauptsächlich von der Anzahl der Kunden ab und beträgt somit $O(|K|)$. Der entsprechende Pseudocode lautet:

```
CustomerSegmentationByClassQuota(C, K, T)
   CT = 0
   Foreach (Ci in C) do
      Ci.Quota = 0;
```

```
   Foreach (Ki in K) do
      If (WasShoppingInPeriod(Ki, T) = true) then
         KC = Ki.Class // implies that class is known
         KC.Quota += 1
         CT += 1
   Foreach (Ci in C) do
      Ci.Quota = Ci.Quota / CT
Return C
```

Falls keine Zugehörigkeitsinformationen für einzelne Kunden vorliegen, kann eine Menge von Klassen $C = \{C_1, \ldots, C_m\}$ definiert werden, um anschließend eine Kundenklassifikation (siehe Kapitel 5.6) durchzuführen, wobei ein Kunde genau der Klasse zuordnet wird, zu der er am ehesten passt. Auf diese Weise erhält man eine Partitionierung der Kunden in die vorher definierten Klassen. Die Klassifikation kann dabei entweder mit Klassifikationsmodellen oder mit Hilfe von Ähnlichkeitsanalysen durchgeführt werden, wobei im letzteren Fall eine Klasse C_i aus C durch einen Prototypen CP_i aus der Menge der Prototypen $CP = \{CP_1, \ldots, CP_m\}$ repräsentiert wird. Ein Prototyp CP_i ist dabei entweder ein durchschnittlicher Kunde der Kundengruppe C_i (entweder Mean oder Medoid) oder eine vorgegebene Attributbeschreibung.

Der entsprechende Algorithmus besitzt eine Komplexität von $O(|K|)\cdot CLAMC$, wobei $CLAMC$ die Komplexität der Verwendung des zugrunde liegenden Klassifikationsmodells bzw. der Klassifikationsmethode repräsentiert, und lautet in Pseudocode:

```
CustomerSegmentationByClassification(CP, K, T) // Variante 1
   CT = 0
   Foreach (CPi in CP) do
      CPi.Quota = 0;
   Foreach (Ki in K) do
      If (WasShoppingInPeriod(Ki, T) = true) then
         Ki.Class = ClassifyCustomer(Ki, CP) // using a classifier model or similarity analysis
         KC = Ki.Class
         KC.Quota += 1
         CT += 1
   Foreach (CPi in CP) do
      CPi.Quota = CPi.Quota / CT
Return CP
```

Der größte Nachteil dieser Vorgehensweise besteht darin, dass Kunden, die eigentlich in

keine angegebene Kundenklasse passen, dennoch in diejenige eingeordnet werden, zu der sie am ähnlichsten sind. Das bedeutet, dass die Güte der Segmentierung sehr stark von den vorgegebenen Kundenklassen abhängig ist. Dies kann teilweise entschärft werden, indem ein Klassifikationsverfahren gewählt wird, das für jeden Kunden K_i die Ähnlichkeiten zu sämtlichen Klassenprototypen CP_j ermittelt. Ist anschließend die größte Ähnlichkeit zu einer Klasse kleiner als ein vorgegebener Schwellenwert, so wird der Kunde der speziell für diesen Fall erzeugten Kundengruppe „Sonstige Kunden“ zugeordnet. Die Komplexität dieser Variante beträgt $O(|K|)\cdot SIMMC$, wobei *SIMMC* die Komplexität der verwendeten Vergleichsmethode ist. Der entsprechende Pseudocode lautet:

```
CustomerSegmentationByClassification(CP, K, T, threshold) // Variante 2
   CT = 0
   OC = NewCustomerGroup("Other Customers")
   CP.Add(OC)
   Foreach (CPi in CP) do
      CPi.Quota = 0;
   Foreach (Ki in K) do
      If (WasShoppingInPeriod(Ki, T) = true) then
         Foreach (CPj in CP) do
            If(Similarity(Ki, CPj) >= threshold) then
               Ki.Classes.Add(CPj, Similarity(Ki, CPj))
         If(Ki.Classes.Count = 0) then
            Ki.Class = OC
            OC.Quota +=1
         Else
            Ki.Class = FirstMemberFrom(Ki.Classes.SortBySimilarity(DESC))
            Ki.Class.Quota += 1
         CT += 1
   Foreach (CPi in CP) do
      CPi.Quota = CPi.Quota / CT
Return CP
```

Eine gängige Art zur Ermittlung einer Kundensegmentierung ist die Durchführung eines Kundenclusterings (siehe Kapitel 5.7) auf der Menge der Kunden K_i, die im fraglichen Zeitraum T eingekauft haben, wobei die Kunden genau einem Cluster zugeordnet werden, um eine Partitionierung zu erreichen. Aus diesem Grund eignen sich in diesem Fall besonders die klassischen Partitionierungsverfahren k-Means und k-Medoid (siehe Kapitel 5.7.1). Die Vorgehensweise besitzt dabei die Komplexität *CLUMC* des verwendeten Clusteringverfahrens und lautet in Pseudocode:

```
CustomerSegmentationByClustering(K, T, k) // Variante 1
   CT = NewList()
   Foreach (Ki in K) do
      If (WasShoppingInPeriod(Ki, T) = true) then
         CT.Add(Ki)
   C = CustomerPartitionClustering(CT, k) // for example k-Means or k-Mediod
   Foreach (Ci in C) do
      Ci.Quota = 0;
   Foreach (Ki in CT) do
         KC = Ki.Class
         KC.Quota += 1
   Foreach (Ci in C) do
      Ci.Quota = Ci.Quota / CT.Members.Count
Return C
```

Bei Clusteringverfahren, die einzelne Kunden gleichzeitig mehreren Clustern mit bestimmten Zugehörigkeitswahrscheinlichkeiten zuordnen (wie z. B. beim EM-Clustering oder anderen Fuzzy-Partitionierungsverfahren), muss nach deren Durchführung eine eindeutige Partitionierung (Segmentierung) durchgeführt werden, indem jeder Kunde genau dem Cluster zugeordnet wird, zu dem er am wahrscheinlichsten gehört:

```
CustomerSegmentationByClustering(K, T, k) // Variante 2
   CT = NewList()
   Foreach (Ki in K) do
      If (WasShoppingInPeriod(Ki, T) = true) then
         CT.Add(Ki)
   C = CustomerPartitionClustering(CT, k) // for example a fuzzy partition clustering algorithm
   Foreach (Ci in C) do
      Ci.Quota = 0;
   Foreach (Ki in CT) do
         KC = FirstMemberFrom(Ki.Classes.SortbySimilarity(DESC))
         KC.Quota += 1
   Foreach (Ci in C) do
      Ci.Quota = Ci.Quota / CT.Members.Count
Return C
```

Analog zur Segmentierung von Kundenklassifikationsergebnissen hat diese Vorgehensweise den Nachteil, dass ein Kunde nur genau einem Cluster zugeordnet werden

darf, obwohl er eventuell mehreren Clustern sehr ähnlich ist, was allerdings in der Praxis im Falle von Partitionierungsalgorithmen daraufhin deutet, dass ein zu kleines k für die Anzahl der Partitionen gewählt wurde.

Clusterergebnisse müssen in der Regel nachträglich untersucht werden, um die markanten Attribute und deren Ausprägungen zu ermitteln, die die jeweiligen Cluster kennzeichnen. Dies kann entweder manuell durchgeführt oder durch entsprechende Verfahren unterstützt werden (beispielsweise durch den *Newman-Keuls-Test* oder den *Scheffé-Test*).

Alternativ schlage ich für die Praxis eine Methode vor, bei der die Cluster C_i mit einer vorgegebenen Menge $CP = \{CP_1, \ldots, CP_n\}$ vorgegebener Kundenprototypen bzw. bestehender Kundenklassen verglichen werden. Dabei werden Ähnlichkeitsmaße verwendet, die sowohl absolute als auch relative Ähnlichkeiten berücksichtigen, da es sich sowohl bei den Prototypen als auch bei den Clusterrepräsentanten (z. B. Means oder Medoids) um Durchschnittsmodelle handelt. Ein Cluster C_i wird jedem Prototypen CP_j zugeordnet, zu dem die Ähnlichkeit größer als ein vorgegebener Schwellenwert ist. Auf diese Weise wird für jedes Cluster ein Ähnlichkeitsprofil bezüglich der vorgegebenen Prototypen erstellt, mit dessen Hilfe die Unterschiede zwischen den Clustern schnell erkannt und verbalisiert werden können, indem jedem Cluster die Vereinigung der markanten Eigenschaften der ihm zugeordneten Prototypen zugesprochen wird.

Die algorithmische Vorgehensweise hat eine Komplexität von $O(|C|\cdot|CP|)\cdot SIMMC$, wobei *SIMMC* wiederum die Komplexität der verwendeten Vergleichsmethode ist, und lautet in Pseudocode:

```
ClassifyCustomerClusteringResults(C, CP, threshold)
   Foreach (Ci in C) do
      Foreach (CPj in CP) do
         If (Compare(Ci, CPj) >= threshold) then // using CompareVectors() or CompareNets()
            Ci.SimilarPrototypes.Add(CPj)
            Ci.Properties = Ci.Properties UNION CPj.Properties
Return C
```

Da ich bei den hier beschriebenen Segmentierungsverfahren davon ausgehe, dass für jeden Kunden K_i individuelle Einkaufshistorien vorliegen, können sowohl bei den Klassifikations- als auch bei den Clusteringverfahren probabilistische Verhaltensnetze und darauf aufbauende Verhaltenssimulationen als Beschreibungs- und Vergleichsgrundlage verwendet werden. Verhaltensbezogene Segmentierungen eignen sich für eine Vielzahl von Fragestellungen im CRM, da die auf diese Weise entstandenen Kundensegmente detaillierte Differenzierungen bzw. Ähnlichkeiten bezüglich komplexer, verhaltensbezogener Merkmale zwischen Kunden aufdecken bzw. berücksichtigen können.

Obwohl ein Kunde im Rahmen einer Kundensegmentierung nur genau einem Segment zugeordnet werden darf, selbst wenn er zu anderen Segmenten ebenfalls sehr hohe Ähnlichkeiten aufweist oder er zu mehreren Segmenten gleich ähnlich ist, liefern Kundensegmentierungen in der Praxis wichtige entscheidungsunterstützende Erkenntnisse. Diese können um weitere interessante Informationen ergänzt werden, beispielsweise, indem für jeden Kunden K_i die Ähnlichkeit zu den einzelnen kundengruppenbeschreibenden Prototypen CP_j berechnet werden. Dadurch lassen sich die Häufigkeiten der einzelnen CP_j ermitteln, wobei jeweils alle Kunden K_i berücksichtigt werden, die CP_j ähnlicher als ein vorgegebener Schwellenwert sind. Auf diese Weise kann ermittelt werden, wie viel Prozent der Kunden den einzelnen Prototypen ähnlich sind, d. h. vor allem, wie viel Prozent der Kunden die Eigenschaftskombinationen von CP_j besitzen. Die Komplexität dieser Vorgehensweise beträgt $O(|K|\cdot|CP|)\cdot SIMMC$ und lautet:

```
CalculatePrototypesQuota(K, CP, threshold)
   Foreach (CPi in CP) do
      CPi.Quota = 0
   Foreach (Ki in K) do
      Foreach (CPj in CP) do
         If (Compare(Ki, CPj) >= threshold) then // using CompareVectors() or CompareNets()
            CPj.Quota += 1
   Foreach (CPi in CP) do
      CPi.Quota = CPi.Quota / K.Members.Count
Return CP
```

Die Berechnung der Prototypenhäufigkeiten kann bei einer Segmentierung mit Hilfe eines Klassifikationsverfahrens, bei dem ein Kunde K_i mehreren Klassen zugeordnet werden darf, quasi nebenbei berechnet werden. Analog kann die Häufigkeit von CP_j bei der Klassifikation von Clusteringergebnissen nebenbei approximiert werden, indem jeweils die Anzahl der Kunden K_i ermittelt wird, die in Cluster fallen, deren Mean bzw. Medoid CP_j ähnlicher als ein vorgegebener Schwellenwert ist.

5.8.2 Kundensegmentierung mit anonymen Kassenbons

Ein großer Anteil der Kassenbons $B = \{B_1, \ldots, B_n\}$ eines Marktes besitzt keinen Kundenbezug und kann aus diesem Grund nicht zum Lernen von kundenindividuellen bzw. kundengruppenbezogenen Verhaltensnetzen verwendet werden. Das heißt vor allem, dass Erkenntnisse und Prognosen, die mit Hilfe von kundenindividuellen oder kundengruppenbezogenen Verhaltensnetzen gewonnen bzw. erstellt wurden, nur auf den durch Kundenkarten bekannten Teil der Kunden eines Marktes bezogen sind und darauf

aufbauend in der Regel keine exakten quantitativen Aussagen über das Verhalten des gesamten Marktes getroffen werden können (es sei denn, es handelt sich bei den verwendeten Kundendaten um eine repräsentative Menge). Ziel ist es daher, die Verteilung bekannter Kundengruppen $C = \{C_1, \ldots, C_m\}$ in der Menge der anonymen Kassenbons B zu ermitteln, um Ergebnisse von kundengruppenbezogenen Verhaltenssimulationen auf den gesamten Markt hochrechnen zu können.

Ein möglicher Ansatz ist die eindeutige Zuordnung einzelner anonymer Kassenbons zu vorgegebenen Kundengruppen. Dies kann durch eine Kassenbonsegmentierung mit kundengruppenbeschreibenden Kassenbonprototypen $BP = \{BP_1, \ldots, BP_m\}$ realisiert werden, d. h. entweder durch Klassifikation der einzelnen Kassenbons oder durch Klassifikation des Ergebnisses eines Kassenbonclusterings, wobei BP_i den Kassenbonprototyp der Kundengruppe C_i repräsentiert. Die Komplexität *SEGMC* hängt dabei hauptsächlich von der zugrunde liegenden Kassenbonsegmentierungsmethode ab:

```
CustomerSegmentationByReceiptSegmentation(B, BP,C)
   BC = ReceiptSegmentation(B, BP) // similar to customer segmentation with attribute vectors
   Foreach (Ci in C) do
      Ci.Quota = BPi.Members.Count / B.Members.Count
Return C
```

Als Kassenbonprototypen BP_i einer Klasse verwende ich dabei den durchschnittlichen Kassenbon der entsprechenden Kundengruppe. Es werden sowohl die einzelnen Kassenbons B_i als auch die kundengruppenbeschreibenden Kassenbonprototypen BP_i durch Attributvektoren beschrieben, die mit Hilfe vektorbasierter Ähnlichkeitsmaße verglichen werden.

Im Falle einer Klassifikation einzelner Kassenbons verwende ich zur Ähnlichkeitsanalyse hauptsächlich Attribute und Ähnlichkeitsmaße, die einen Schwerpunkt auf relative Werte legen (wie z. B. das Kosinus-Maß), da konkrete Kassenbons mit durchschnittlichen Prototypen verglichen werden. Bei der Klassifikation eines Kassenbonclusterings setze ich hingegen Attribute und Maße ein, die sowohl absolute als auch relative Ähnlichkeiten berücksichtigen, da es sich sowohl bei den Clusterrepräsentationen als auch bei den Prototypen um Durchschnittsmodelle handelt.

Ein Nachteil der Kundensegmentierung basierend auf einer Kassenbonsegmentierung ist, dass einzelne Kassenbons nur genau einer Kundengruppe zugesprochen werden, obwohl sie in der Regel mehreren zugeordnet werden können. Aus diesem Grund bietet sich als zusätzliche Information die Zuordnung der Kassenbons B_i bzw. der Kassenboncluster BC_j zu allen Kundengruppen an, zu deren kundengruppenbeschreibenden Kassenbonprototypen sie ähnlicher sind als ein vorgegebener Schwellenwert.

Alternativ können die Kassenbonprototypen BP_i aber auch reine Kassenbontypen sein, die keine vollständigen Kundengruppen repräsentieren, sondern einzelne Kundeneigenschaften beschreiben. Tabelle 1 zeigt beispielsweise die im Rahmen meiner Arbeit und in Zusammenarbeit mit dem H.I.MA entwickelten Kassenbonprototypen. Sie stellen typische Kassenbonausprägungen in den Vordergrund und entsprechen damit nicht unbedingt vollwertigen Kundentypen, d. h. die Einkaufshistorie eines Kunden weist in der Regel unterschiedliche Kassenbontypen auf.

Tabelle 1: Die Charakterisierung verschiedener Kassenbontypen[2]

Kassenbontyp	**Charakterisierung**
Markenkäufer	• Geringe Anzahl an Artikeleinheiten • Geringe Sortimentsvielfalt • Hoher Prozentsatz an Markenartikeln • Artikel eher aus dem höheren Preissegment • Geringer Prozentsatz an Aktionsartikeln
Vielkäufer	• Sehr hohe Anzahl von Artikeleinheiten • Sehr große Sortimentsvielfalt • Artikel überwiegend aus dem niedrigen Preissegment • Geringer bis mittelmäßiger Prozentsatz an Aktionsartikeln
Wocheneinkäufer	• Hohe Anzahl von Artikeleinheiten • Große Sortimentsvielfalt • Artikel aus allen Preiskategorien, wobei maßgeblich aus dem niedrigen Preissegment • Geringer bis mittelmäßiger Prozentsatz an Aktionsartikeln
Billigkäufer	• Geringe Anzahl von Artikeleinheiten • Sehr geringe Sortimentsvielfalt • Keine Markenartikel • Artikel hauptsächlich aus dem niedrigen Preissegment • Geringer bis mittelmäßiger Prozentsatz an Aktionsartikeln
Smart-Shopper	• Geringe Anzahl an Artikeleinheiten • Sehr geringe Sortimentsvielfalt • Keine Markenartikel • Artikel hauptsächlich aus dem mittleren Preissegment • Sehr geringer Prozentsatz an Aktionsartikeln

[2] Quelle: Institut für Handel & internationales Marketing (H.I.MA.)

5.8.3 Feature-Matching

Um einzelne anonyme Kassenbons vorgegebenen Kundengruppen zuzuordnen, kann ein *Feature-Matching* zwischen Kassenbons und Kundengruppen durchgeführt werden (siehe Abbildung 51). Eine Kundengruppe C_j wird dabei nicht durch einen einfachen durchschnittlichen Kassenbon repräsentiert, sondern durch einen Profilvektor PV_j, der für jede Warengruppe WG_i einer Menge vorgegebener Warengruppen $WG = \{WG_1, \ldots, WG_n\}$ beschreibt, welche Eigenschaften (Features) die von den jeweiligen Kunden gekauften Artikel im Durchschnitt besitzen. Der Vorteil dabei ist, dass mit Hilfe dieser kundengruppenbeschreibenden Eigenschaftsmerkmale, die aus den Artikelstammdaten ermittelt werden, sowohl Kassenbons als auch Kundengruppen aufgrund von Attributen charakterisiert werden, mit denen sich auch klassische Kundengruppentypologien beschreiben lassen.

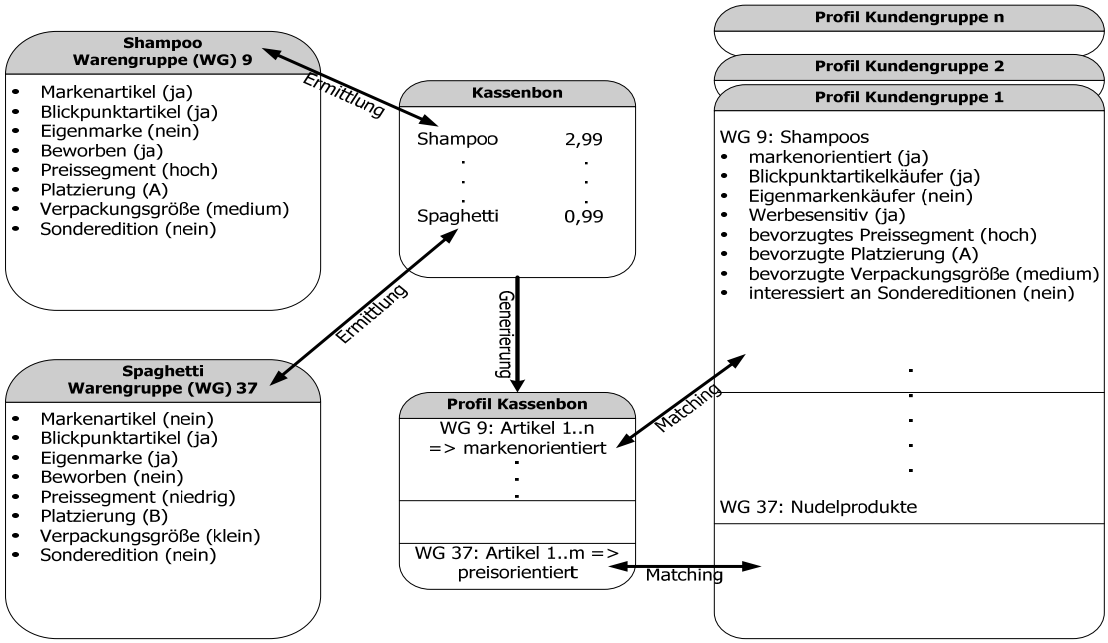


Abbildung 51: Feature-Matching zwischen Kassenbons und Kundengruppen

Um nun einen einzelnen Kassenbon B_i einer oder mehreren Kundengruppen C_j zuzuordnen, wird für B_i ein entsprechender Profilvektor PVB_i erstellt, der anschließend mit den Profilvektoren PV_j aller vorgegebener Kundengruppen verglichen wird. Anschließend wird B_i entweder der Kundengruppe zugeordnet, zu deren Profilvektor PV_j PVB_i am ähnlichsten ist oder alternativ allen Kundengruppen, zu deren Profilvektoren PV_j PVB_i ähnlicher als ein definierter Schwellenwert ist.

6 SimMarket: Applikation, Anwendungsbeispiele und Evaluation

In diesem Kapitel präsentiere ich eine konkrete Implementierung des vorgestellten Kundenmodells, die im Rahmen des Verbundprojektes *SimMarket* am *Deutschen Forschungszentrum für Künstliche Intelligenz* (*DFKI GmbH*) entwickelt wurde. Dabei waren neben dem DFKI das *Institut für Handel & internationales Marketing* an der Universität des Saarlandes (*H.I.MA.*), das Saarbrücker Softwareunternehmen *Dacos Software GmbH* sowie die deutschen Handelsunternehmen *dm-drogerie markt*, *Globus* und *tegut…* als Projektpartner beteiligt. Der Schwerpunkt des Projektes liegt in der agentenbasierten Modellierung und Simulation realer Supermarktfilialen und deren Kunden. Kapitel 6.1 gibt einen Überblick über das SimMarket System, mit dessen Hilfe ich Anwendungsbeispiele der Modellierung von Kunden und Kundengruppen (Kapitel 6.2) sowie Beispiele und Evaluationen der Kundensimulation (Kapitel 6.3) und Kundengruppenbestimmung (Kapitel 6.4) präsentiere. In Kapitel 6.5 zeige ich Anwendungsbeispiele der in Kapitel 5.8 vorgestellten Verfahren zur Kunden- bzw. Kassenbonsegmentierung. Kapitel 6.6 gibt einen Überblick über verwandte Arbeiten und Forschungsprojekte.

6.1 Das SimMarket System

Das im Rahmen des Forschungsprojektes SimMarket [SS04b] [RSS04] des Deutschen Forschungszentrums für Künstliche Intelligenz entwickelte SimMarket System ist eine konkrete Umsetzung des in den vorherigen Kapiteln beschriebenen kundenbezogenen probabilistischen Multiagentensystems [SS03]. Ziel des Projektes ist es, alle relevanten Entitäten, Beziehungen und Einflussfaktoren des Einzelhandels – bezogen auf eine einzelne Filiale – möglichst exakt zu modellieren, um anschließend die Auswirkungen nahezu beliebiger „Was-wäre-wenn-Szenarien“ simulieren zu können, wobei der Schwerpunkt auf den Auswirkungen von Preis- und Promotionsmaßnahmen liegt.

Die allgemeine Vorgehensweise zur Entscheidungsfindung bezüglich der Preis- und Promotionsplanung besteht darin, dass der Sortimentsverantwortliche, ausgehend von der *heutigen Situation*, bestimmte Annahmen über die *zukünftigen externen Einflüsse* macht und daraufhin eine Menge von *Maßnahmen* und *Aktionen* auswählt, von denen er überzeugt ist, dass sie die zu erreichenden *Unternehmensziele* erfüllen werden. Im SimMarket System wird dieser Prozess mit Hilfe agentenbasierter Simulation unterstützt. Dabei stellt die Komplexität der zugrunde liegenden Domäne extreme Anforderungen an ein solches System, das in der

Lage sein muss, sehr viele individuelle Objekte und deren Abhängigkeiten modellieren und anschließend simulieren zu können. Deshalb werden alle relevanten Entitäten wie Kunden, Kundengruppen, Artikel, Warengruppen, Konkurrenten, Hersteller und Umwelteinflüsse als probabilistische Agenten innerhalb eines durch Holonisierung skalierbaren Multiagentensystems repräsentiert. Die entsprechenden Verhaltensnetze und Abhängigkeiten zwischen den einzelnen Agenten werden aus realen Daten jeweils einer Filiale der Verbundprojektpartner *dm-drogerie markt*, *Globus* und *tegut…* gelernt.

6.1.1 Agentenbasierte Architektur des SimMarket Systems

Die Architektur des SimMarket Systems basiert auf einem holonischen Multiagentensystem, in dem alle relevanten Entitäten einer Supermarktfiliale als probabilistische Agenten repräsentiert werden (siehe Abbildung 52). Dabei wird die konkrete Filiale durch einen Filial- bzw. Supermarktagenten modelliert, der Kunden und Kundengruppen sowie Artikel und Warengruppen umfasst, die jeweils durch entsprechende Agenten repräsentiert werden. Die auf den Markt wirkenden externen Einflussfaktoren wie Konkurrenz, Lieferanten und sonstige Umweltfaktoren werden ebenfalls durch probabilistische Agenten modelliert.

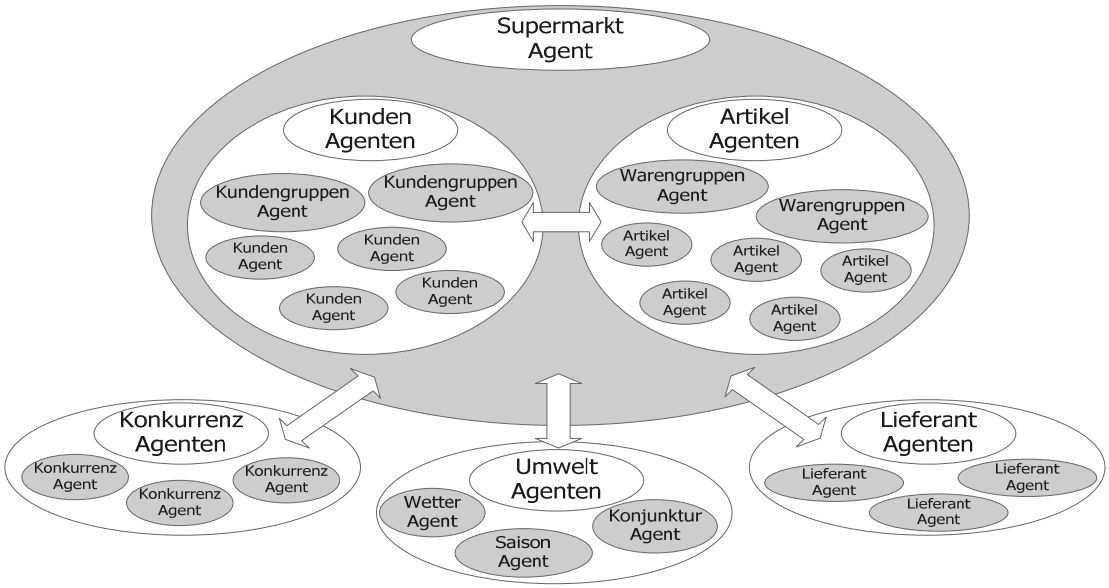


Abbildung 52: Agentenbasierte Architektur des SimMarket Systems

Der **Supermarktagent** verwaltet die Daten über die modellierte Filiale, wie beispielsweise Informationen über im Markt durchgeführte Maßnahmen und Aktionen wie Preis-, Sortiments- und Platzierungsänderungen, Promotionen, Sonderaktionen sowie filialbezogene Kennzahlen. Er besitzt damit sämtliche historischen Informationen über vergangene Szenarien im Sinne der in Kapitel 4 vorgestellten Simulationsszenarien.

Kundenagenten sind eine konkrete Umsetzung der von mir entwickelten probabilistischen Kundenagenten, die sich durch verschiedenartige Holonisierungsformen zu Kundengruppen zusammenfinden können, die als Kundengruppenagenten modelliert werden. Ein Kunde kann dabei zur gleichen Zeit Mitglied beliebig vieler Kundengruppen sein kann. Kundenagenten repräsentieren jeweils einen individuellen Kunden mit dessen persönlichem Verhalten, das aus entsprechenden Kundenkartendaten, Umfragen, Daten des elektronischen Zahlungsverkehrs und des Kontakts zum Kundenservice gewonnen und in probabilistischen Verhaltensnetzen gespeichert wird.

Neben den Kunden werden im SimMarket System zusätzlich sämtliche Artikel des Supermarktes durch einzelne **Artikelagenten** repräsentiert, die das globale Abverkaufsverhalten der entsprechenden Artikel modellieren (für eine Beschreibung der Architektur und Funktionalität der Artikelagenten des SimMarket Systems siehe [SS04b] und [SB06]). Artikelagenten können analog zu Kundenagenten durch Holonisierung zu Warengruppen zusammengefasst werden, die in der Regel die Warengruppenhierarchie der Filiale nachbilden.

In der agentenbasierten Architektur des SimMarket Systems sind neben Kunden- und Artikelagenten auch probabilistische Agenten zur Repräsentation von **Lieferanten bzw. Herstellern** vorgesehen, die deren historischen und aktuellen Produktpaletten, Verkaufspreise und Promotionsmaßnahmen verwalten. Aus diesen Daten können Verhaltensmuster der Lieferanten erlernt werden, beispielsweise bezüglich Lieferzuverlässigkeiten, Zahlungsmoral oder typischer Werbe- und Preisstrategien.

Analog zu den Lieferantenagenten sieht die Architektur **Konkurrenzagenten** vor, die relevante konkurrierende Supermarktfilialen repräsentieren. Konkurrenzagenten verwalten das bekannte Wissen über das Verhalten der Konkurrenz, wie Artikelpreise, Werbeaktionen oder anderer Marketingstrategien.

Die auf die Filiale wirkenden externen Einflussfaktoren, wie beispielsweise die allgemeine Wirtschaftslage, meteorologische Einflüsse, Jahreszeit oder spezielle Termine (Weihnachten, Valentinstag, usw.) werden durch **Umweltagenten** verwaltet. Umweltagenten können als *probabilistische Informationsagenten* realisiert werden, die relevante Informationen, wie beispielsweise aktuelle Börsenkurse oder momentane Wetterlage, selbstständig im Internet bzw. bei entsprechenden Webservices recherchieren und verwalten können. Aufgrund ihrer Datenbasis können sie qualitative Prognosen über das erwartete Verhalten oder die erwarteten Ausprägungen der externen Einflussfaktoren erstellen oder diese von externen Diensten - wie der Wettervorhersage vom deutschen Wetterdienst – anfordern und in das Gesamtsystem einfließen lassen.

6.1.2 Simulationsablauf

Abbildung 53 zeigt den Ablauf der szenariobasierten Simulation innerhalb des SimMarket Systems zur Entscheidungsunterstützung im Rahmen der Planung von filialbezogenen Marketing-Maßnahmen. Ausgehend von einer Initialsituation wird ein mögliches zukünftiges Szenario durch Angabe konkreter interner Maßnahmen und externer Einflüsse definiert. Anschließend werden Kunden- und Artikelagenten mit der im Szenario beschriebenen Situation konfrontiert, auf die sie anhand des in ihren Verhaltensnetzen modellierten Verhaltens reagieren. Die Simulation der Reaktionen der einzelnen probabilistischen Agenten auf das Szenario entspricht dem in Kapitel 4 beschriebenen szenariobasierten Simulationsprozess. Die Verhaltenssimulation basiert dabei auf entsprechenden Schlussfolgerungen, die mit Hilfe von Inferenz-Algorithmen für probabilistische Verhaltensnetze durchgeführt werden.

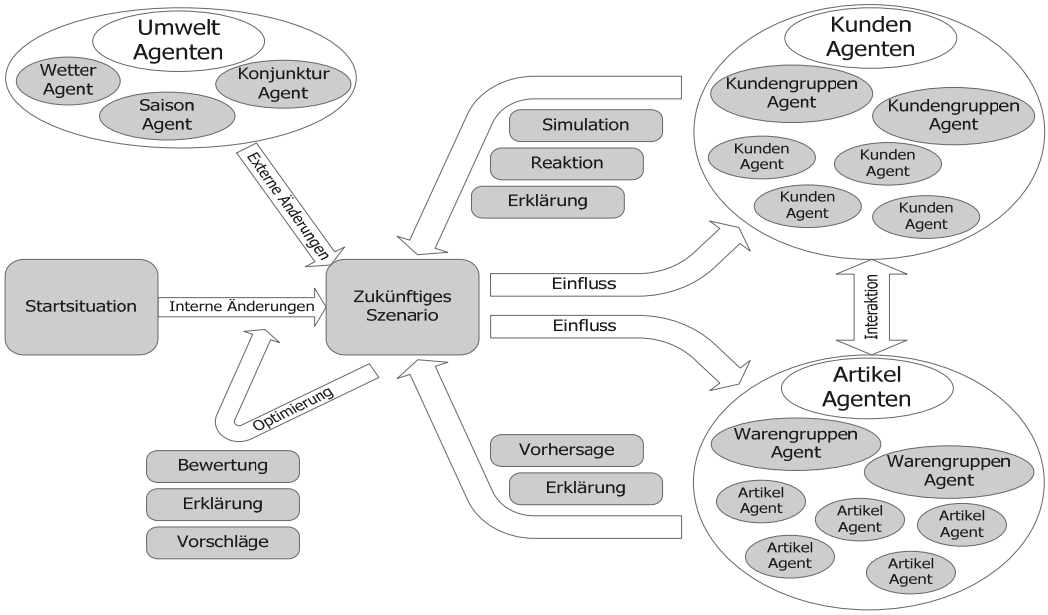


Abbildung 53: Entscheidungsunterstützender Simulationsablauf des SimMarket Systems

Ein Simulationsszenario wird im SimMarket System als *Szenario-Objekt* gespeichert. Ein Szenario-Objekt ist eine konkrete Implementierung des in Kapitel 4.1 beschriebenen Simulationsszenarios. Attribute und Parameter entsprechen dabei einer vereinfachten Umsetzung der vorgestellten Szenariodimensionen. Die im Szenario vorgegebenen internen Maßnahmen werden als *Aktions-Objekte* gespeichert, die neben Angaben zum Artikelbezug und Aktionszeitraum einen semantischen Typ besitzen (beispielsweise den Typ „Preissenkung“) und die konkreten Aktionsevidenzen beinhalten. Analog werden Annahmen, Prognosen und Informationen über die zukünftigen externen Einflüsse vorgemerkt.

Während der Simulation werden die Reaktionen der Kunden auf die durch Maßnahmen definierten Sortimentsveränderungen unter Berücksichtigung der vorgegebenen Umwelteinflüsse berechnet, indem jeder Kundenagent angibt, wie viele Einheiten er von jedem in der Artikeldimension angegebenen Artikel, im angegebenen Zeitraum und unter den beschriebenen Umständen, kaufen wird. Analog ermitteln alle Artikelagenten ihre zu erwartenden Absatzzahlen.

Nachdem alle Kundenagenten ihre virtuellen Einkäufe getätigt haben, werden die Reaktionen summiert und zur Berechnung der vorgegebenen Kennzahlen, wie Umsatz oder Ertrag, herangezogen.

Der auf diesem Simulationsprozess aufbauende Entscheidungsprozess zur Findung geeigneter Maßnahmenkombinationen besteht aus mehreren Schritten. Ausgehend von einer vorgegebenen Startsituation, die in der Regel die aktuelle Situation beschreibt, werden ein oder mehrere zukünftige Szenarien erstellt. Anschließend werden die Auswirkungen der einzelnen Szenarien jeweils durch agentenbasierte Simulation prognostiziert, die Ergebnisse evaluiert und miteinander verglichen. Danach können einzelne Szenarien entweder verworfen, übernommen oder durch Maßnahmenänderungen modifiziert und erneut simuliert werden, bis ein Szenario gefunden wird, dessen Maßnahmen die gewünschten Ziele am besten erfüllt.

6.1.3 Software-Architektur und Implementierung

Die vorgestellte agentenbasierte Architektur und der szenariobasierte Simulationsablauf des SimMarket Systems stellen sehr hohe Ansprüche an die Entwicklung eines entsprechenden Softwaresystems, das darüber hinaus bei den Verbundpartnern aus dem Einzelhandel auf Akzeptanz stoßen und entsprechenden Nutzen vorweisen muss:

- Die zugrunde liegende Agentenplattform muss performant und im Idealfall auf mehrere Rechner verteilbar sein, um einzelne Agenten je nach Last und Rechenleistung auf ein Netzwerk verteilen zu können, da die Anzahl der Kunden und Artikel bei SB-Warenhäusern mehrere 100.000 Agenten erfordern kann.
- Die einzelnen Agenten müssen einen performanten Zugriff auf Datenbanken und Data Warehouses besitzen, um auf die realen Daten der Verbundpartner zugreifen zu können, die in der Praxis mehrere hundert Gigabytes in Anspruch nehmen.
- Die Agenten müssen über effiziente Kommunikationskanäle Informationen miteinander austauschen und sich zur Laufzeit mit Hilfe verschiedener Holonisierungsarten zu Gruppen zusammenschließen können.
- Die Agenten müssen in der Regel eine Vielzahl unterschiedlicher probabilistischer Verhaltensnetze verwalten, die anhand historischer Daten aus den Datenbanken gelernt werden.

- Es müssen performante Bibliotheken zum Lernen und Verschmelzen probabilistischer Verhaltensnetze vorliegen, die darüber hinaus die beschriebenen unterschiedlichen Arten des Schlussfolgerns und die Modellierung von Domänenwissen unterstützen.
- Das System sollte idealerweise eine Client-Server-Architektur besitzen sowie benutzerfreundliche graphische Oberflächen, die durch einen hohen Wiedererkennungswert und Bedienkomfort bei den Verbundpartnern aus dem Handel auf hohe Akzeptanz stoßen.

Zu Beginn der Implementierungsphase im Rahmen des Verbundprojektes SimMarket wurde nach einer ausführlichen Analyse möglicher Softwareumgebungen und Agentenplattformen deutlich, dass keine der existierenden Multiagentenplattformen die definierten Anforderungen zufrieden stellend erfüllen konnte. Die Mehrzahl der Plattformen sind entweder für spezielle Anwendungen entwickelt worden – wie beispielsweise *Avalanche* zur agentenbasierten Koordination von Wertschöpfungsketten, *Aglets* von IBM als Plattform für mobile Agenten oder die freie *SWARM* Software, die auf soziale Simulationen bzw. Artificial-Life-Anwendungen spezialisiert ist – oder sie sind zu ineffizient, um eine große Anzahl komplexer Agenten verwalten zu können, wie beispielsweise *JACK* oder *JiVE*, die beide in JAVA implementiert sind und sich nur für „kleine“ Multiagentensysteme eignen. Sogar *Cougaar*, eine Plattform, die speziell für große Multiagentensysteme entwickelt wurde, verwaltet nur bis zu 1000 Agenten. Ein weiterer Nachteil der meisten Plattformen ist, dass sie nicht ohne weiteres auf die Ansprüche von SimMarket angepasst werden können, wie beispielsweise die Integration von probabilistischen Agenten, Verhaltensnetzen und verschiedenen Arten der Holonisierung sowie der Möglichkeit, Agenten effizient auf verschiedene Rechner verteilen zu können und sie performant auf Datenbanken und Data Warehouses zugreifen zu lassen.

Aus diesem Grund wurde im Rahmen des Forschungsprojektes ein eigenes holonisches Multiagenten-Framework für probabilistische Agenten implementiert. Das Framework basiert auf dem .NET-Framework, das sowohl objekt-, komponenten- und service-orientiertes Programmieren als auch Remote- und Webservices komfortabel unterstützt, sowie eine performante und komfortable Bibliothek für graphische Benutzeroberflächen besitzt, mit der sich Applikationen im typischen Windows- und Office-Stil in kurzer Zeit entwickeln lassen, der von der Mehrzahl der Verbundpartner bevorzugt wird. Darüber hinaus kann mit Hilfe von umfangreichen Bibliotheken (beispielsweise ADO.NET) performant auf Datenbanken und Data Warehouses zugegriffen werden. Die konkrete Implementierung wurde in der Programmiersprache C# - einer JAVA-ähnlichen Erweiterung von C++ - unter Verwendung der Entwicklungsumgebung Visual Studio .NET durchgeführt. Als Datenbank wurde der SQL-Server gewählt.

Neben der Eigenentwicklung einer Plattform für Multiagentensysteme war es ebenso notwendig, eine eigene Bibliothek für probabilistische Verhaltensnetze zu implementieren, da die existierenden Bibliotheken, nach einer Evaluation von 38 Programmen für Bayes'sche Netze, entweder eine unzureichende Funktionalität oder Performanz aufwiesen (wie beispielsweise *WineMine* und *Netica*) oder nicht individuell erweitert werden konnten, da es sich um kommerzielle Applikationen handelt (wie beispielsweise *HUGIN*), deren API sich als Blackbox darstellen, die nur bedingt modifiziert werden können (siehe [SS04b] für eine ausführliche Analyse der betrachteten Bibliotheken und Applikationen).

Die im Rahmen des Projektes SimMarket entwickelte Bibliothek für probabilistische Verhaltensnetze umfasst eine Vielzahl von Strukturen und Verfahren zum Lernen und Verschmelzen von Netzen sowie unterschiedliche Inferenz-Algorithmen für weiche und harte Evidenzen. Darüber hinaus unterstützt die Bibliothek die Modellierung von Domänenwissen und entsprechende Heuristiken zur Optimierung der Lern- und Verschmelzungs-Algorithmen sowie die Repräsentation zeitlicher Abhängigkeiten durch Deltaknoten und Deltazustände.

Aufbauend auf dem Agentenframework und der Bibliothek für probabilistische Netze wurden im Rahmen des Projektes SimMarket zwei prototypische Softwaresysteme implementiert.

Die erste Umsetzung entspricht einer *Drei-Schicht-Architektur*, die Daten, Funktionalität und Benutzeroberflächen trennt. Sämtliche Funktionseinheiten sind in *Managern* gekapselt, die jeweils als *Remote-Service* implementiert sind und Zugriff auf Informationen in Datenbanken und Data Warehouses besitzen (siehe Abbildung 54).

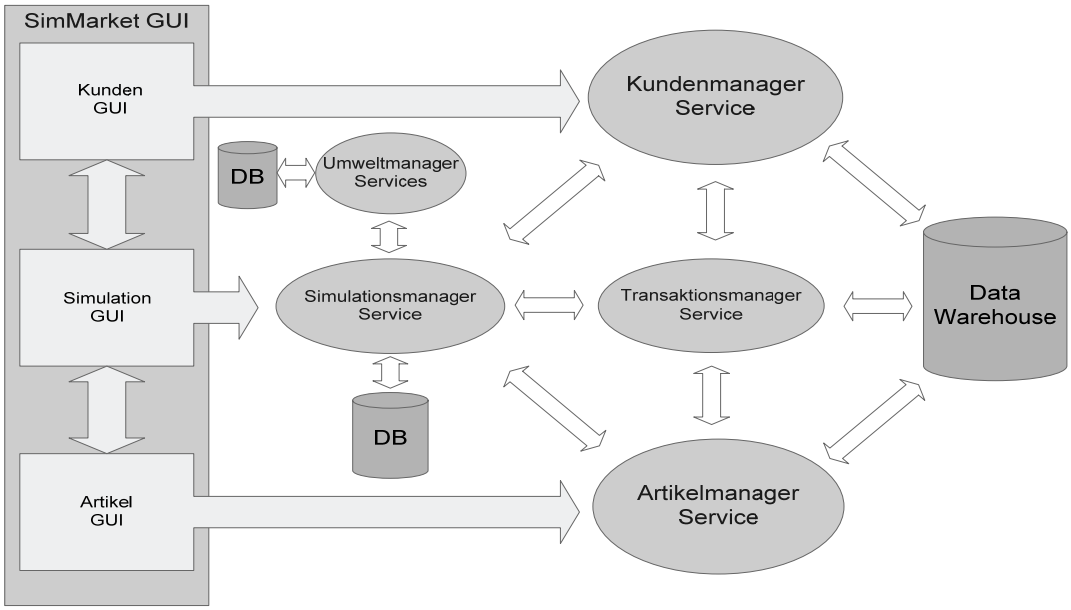


Abbildung 54: Service-orientierte SimMarket Software-Architektur

Die service-orientierte Architektur umfasst folgende Komponenten:

- den Kundenmanager, der alle Kunden- und Kundengruppenagenten verwaltet,
- den Artikelmanager, der alle Artikel- und Warengruppenagenten verwaltet,
- den Transaktionsmanager, der Kassenbons und die ihnen zugrunde liegenden Transaktionen verwaltet,
- den Umweltmanager, der Informationen über externe Einflussfaktoren verwaltet,
- den Simulationsmanager, der die Simulations-, Prognose- und Evaluationslogik enthält sowie Simulationsszenarien und deren Ergebnisse verwaltet,
- ein Data Warehouse, das sämtliche Transaktions-, Artikel- und Kundendaten verwaltet sowie
- graphische Benutzeroberflächen, die sowohl einzeln als auch unter einer Gesamtoberfläche vereint verwendet werden können.

In der zweiten Umsetzung des SimMarket Systems wurde das Agenten-Framework weiter ausgebaut, so dass sämtliche Entitäten und Services einheitlich als Agenten implementiert werden können. Das Framework ist eine vollwertige allgemeine Plattform für Multiagentensysteme, die auch für andere Anwendungen und Systeme effizient genutzt werden kann. Als Zielsetzung für das Framework wurden alle in Abbildung 55 gezeigten Funktionalitäten definiert. Die gesamte Plattform ist auf nahezu beliebig viele Rechner verteilbar und kann unterschiedliche Agentenklassen verwalten. Sämtliche Objekte und Entitäten – von einzelnen Artikeln und Kunden über deren Gruppierungen bis hin zu Systemanwendern – werden durch komponentenbasierte Agenten innerhalb des Multiagentensystems repräsentiert. Die Kommunikation zwischen einzelnen Agenten wird durch entsprechende Remote-Zugriffe realisiert. Zur Bildung von Agentengruppen wurden Cluster- und Klassifikationsverfahren sowie verschiedene Arten der Holonisierung implementiert. Die Datengrundlage der Agenten ist in Datenbanken (siehe [SP04]) und Data Warehouses gespeichert, auf die die einzelnen Agenten sowohl durch direkten Zugriff als auch mit Hilfe verteilbarer Data Mining Verfahren zugreifen können. Informationsagenten können auf das Internet und entsprechende Webservices zugreifen, wie zum Beispiel Wetterdienst oder Börsenticker. Durch die Modellierung der Systemanwender als Agenten wird die Anpassung des Programms und der jeweiligen graphischen Benutzeroberflächen sowie die Verwaltung von Zugriffsrechten, Einwahlmöglichkeiten und Sicherheitsvorkehrungen erleichtert. Für eine genauere Beschreibung der Umsetzung des generellen Agentenframeworks – insbesondere der Verteilung und Migration von einzelnen Agenten – siehe [SSt05].

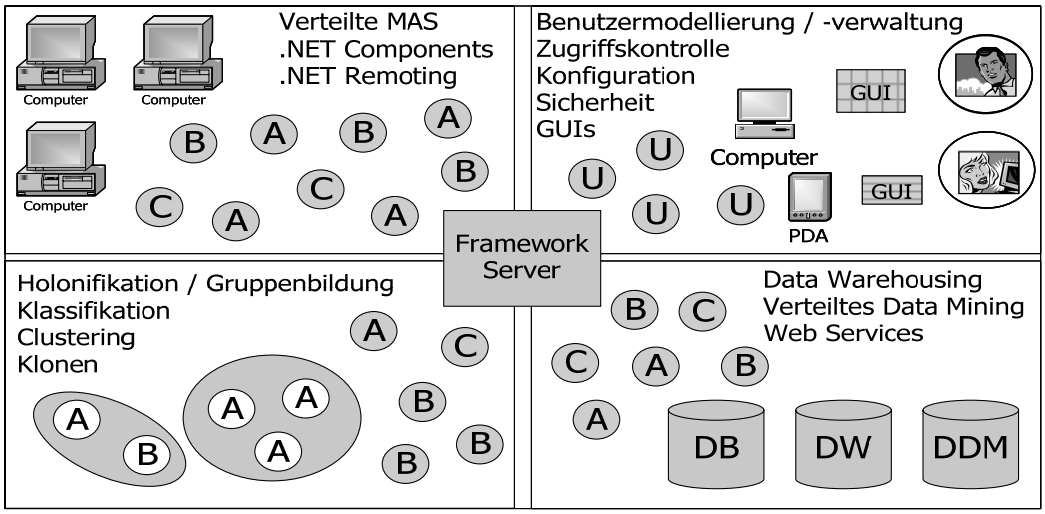


Abbildung 55: Das erweiterte Multiagenten-Framework

Insgesamt umfassen die beiden Implementierungen des SimMarket Systems, inklusive der entwickelten Multiagentenplattform und der umfangreichen Bibliothek für probabilistische Netze, nahezu 500.000 Zeilen C#-Code. Teile des SimMarket Systems dienen als Grundlage bzw. als Forschungsprototyp der Simulation Engine der Dacos Software GmbH in Saarbrücken.

6.1.4 Applikation

Das SimMarket System umfasst mehrere kundenbezogene Funktionalitäten, die in unterschiedlichen Komponenten zusammengefasst wurden:

- Kundenmanager,
- Kundenklassifikation,
- Kundenclustering,
- Simulationsmanager,
- Evaluationsmanager,
- Kassenbonklassifikation und
- Kassenbonclustering.

Der **Kundenmanager** bietet einen detaillierten Überblick über Kundengruppen und einzelne Kunden, und dient in erster Linie zur Bereitstellung von Informationen. Beispielsweise können anonymisierte soziodemographische Daten, einfache statistische Kennzahlen, Kundenwertmaße (z. B. customer lifetime value), Transaktionshistorien einzelner Kunden und Kundengruppen eingesehen werden, die die Grundlage des Erlernens der kundenindividuellen Verhaltensnetze bilden. Darüber hinaus können für einzelne Kunden und Kundengruppen bei Bedarf ausgewählte artikelbezogene Verhaltensnetze (siehe Kapitel 3.2.4, holonische Artikelgruppenverhaltensnetze (siehe Kapitel 3.2.5) und globale Verhaltensnetze gelernt und angezeigt werden. Die Netze stehen anschließend für detaillierte Analysen und manuelle Simulationen zur Verfügung.

Die Komponente **Kundenklassifikation** unterstützt eine Vielzahl unterschiedlicher Klassifikationsverfahren zur Einordnung einzelner Kunden in vorgegebene Kundenklassen. Kundenklassen können dabei sowohl durch einzelne Kunden und Kundengruppen als auch durch virtuelle Prototypen repräsentiert werden, die besondere Merkmalsausprägungen beschreiben. Als Vergleichsgrundlage der Ähnlichkeitsanalysen können sowohl einfache und erweiterte Attributvektoren als auch probabilistische Verhaltensnetze verwendet werden. Klassifikationsergebnisse können gespeichert werden, um neue Kundengruppen und Kundengruppenzugehörigkeiten zu erstellen. Darüber hinaus können mit Hilfe der Kundenklassifikation Kundensegmentierungen durchgeführt werden.

Die Komponente **Kundenclustering** umfasst mehrere Clusteringverfahren, die sowohl einfache und erweiterte Attributvektoren als auch probabilistische Verhaltensnetze als Vergleichsgrundlage verwenden. Eine vorgegebene Kundenmenge kann mit ihrer Hilfe in geeignete Cluster partitioniert werden, die sich anschließend als neue Kundengruppen abspeichern lassen. Darüber hinaus besteht die Möglichkeit entdeckte Cluster mit vorgegebenen Prototypen zu klassifizieren, um markante Clustereigenschaften zu ermitteln. Ferner können mit Hilfe der Clusteringverfahren sowohl klassische als auch erweiterte, verhaltensbezogene Kundensegmentierungen durchgeführt werden.

Der **Simulationsmanager** dient zur Konfiguration und Durchführung szenariobasierter Verhaltenssimulationen. Dazu können Szenarien für eine vorgegebene Artikelmenge durch Festlegung unterschiedlicher artikelbezogener Maßnahmen (wie Preisänderungen oder Promotionsaktionen) innerhalb eines ausgewählten Zeitraumes definiert werden. Anschließend können die Auswirkungen der Szenarien auf vorgegebene Kundengruppen simuliert werden. Auf diese Weise lassen sich Analysen vergangener Entscheidungen und Prognosen über die Auswirkungen zukünftiger Maßnahmen durchführen, auf deren Basis die Planung von Strategien und konkreten Aktionen optimiert werden kann.

Der **Evaluationsmanager** dient zur Evaluation der Simulationsergebnisse, um die Güte der verhaltensnetzbasierten Simulationen und Prognosen bewerten und optimieren zu können. Mit Hilfe des Evaluationsmanagers können historische Szenarien sowohl manuell als auch automatisiert nachgestellt werden, deren Auswirkungen anschließend durch Simulation prognostiziert werden, wobei die probabilistischen Netze aus Daten gelernt werden, die zeitlich vor den jeweiligen Evaluationsszenarien liegen. Auf diese Weise können die Simulationsergebnisse mit den realen Auswirkungen verglichen werden, um die jeweilige Simulationsgüte mit unterschiedlichen Güte- und Fehlermaßen zu quantifizieren.

Die Komponenten **Kassenbonklassifikation** und **Kassenbonclustering** stellen Funktionalitäten zur Verfügung, um Aussagen über Kundengruppen und deren Verteilung im Markt, basierend auf anonymen Kassenbondaten, treffen zu können (siehe Kapitel 5.8.2). Bei der Kassenbonklassifikation gibt der Anwender eine Menge virtueller Kassenbonprototypen

vor, die anschließend zur vektorbasierten Klassifikation einer vorgegeben Kassenbonmenge dienen. Da den verwendeten Kassenbonprototypen kundenbezogene Merkmale und Verhaltensmuster zugeordnet werden können, kann die Verteilung entsprechender Kundengruppen mit Hilfe der Kassenbonklassifikation geschätzt werden.

Analog können durch Clustering entdeckte Kassenboncluster mit vorgegebenen, kundengruppenbeschreibenden Kassenbonprototypen klassifiziert werden, um die Verteilung der entsprechenden Kundengruppen im Markt abschätzen zu können.

6.2 Anwendungsbeispiele der Kundenmodellierung

Im Folgenden präsentiere ich anhand einiger Anwendungsbeispiele die konkrete Umsetzung der in den vorangegangenen Kapiteln erläuterten Modellierung individuellen Kundenverhaltens im SimMarket System. Ich verwende dazu den Kundenmanager, der detaillierte Informationen über Kunden- und Kundengruppenagenten bietet, die aus den historischen Daten einer Filiale des Verbundpartners *Globus* erzeugt wurden. Über eine graphische Benutzerschnittstelle können, neben Kundengruppenhierarchien und einem Überblick über die Menge aller Kundenagenten, detaillierte Informationen sowohl über das Faktenwissen als auch über das probabilistische Strukturwissen der einzelnen Agenten angezeigt werden, die als Grundlage für unterschiedliche Fragestellungen des Customer Relationship Management dienen. Zunächst beschreibe ich in Unterkapitel 6.2.1 die Umsetzung und Visualisierung der Kundenagenten, die das Verhalten einzelner realer Kunden modellieren. Anschließend präsentiere ich in Unterkapitel 6.2.2 auf analoge Weise die Modellierung von Kundengruppen, die jeweils eine Menge einzelner Kunden repräsentieren.

6.2.1 Modellierung individueller Kunden

Einzelne Kunden werden im SimMarket System mit Hilfe probabilistischer Kundenagenten modelliert. Das Faktenwissen der Agenten beinhaltet unter anderem soziodemographische Informationen und einfache, statistische Kennzahlen, die ich in Form eines *Kundenprofils* zusammenfasse. Abbildung 56 zeigt das anonymisierte Kundenprofil eines konkreten Kunden. Das Profil enthält neben einer anonymisierten Kundennummer (Kunden-ID) soziodemographische Angaben über Alter, Anrede und Postleitzahlenbereich. Da Globus keine Kundenkarte einsetzt, liegen die konkreten Ausprägungen dieser Merkmale allerdings nur in seltenen Fällen vor. Interessanter sind daher die statistischen Verteilungen und Kennzahlen, die einen Überblick über das allgemeine Kaufverhalten des Kunden geben.

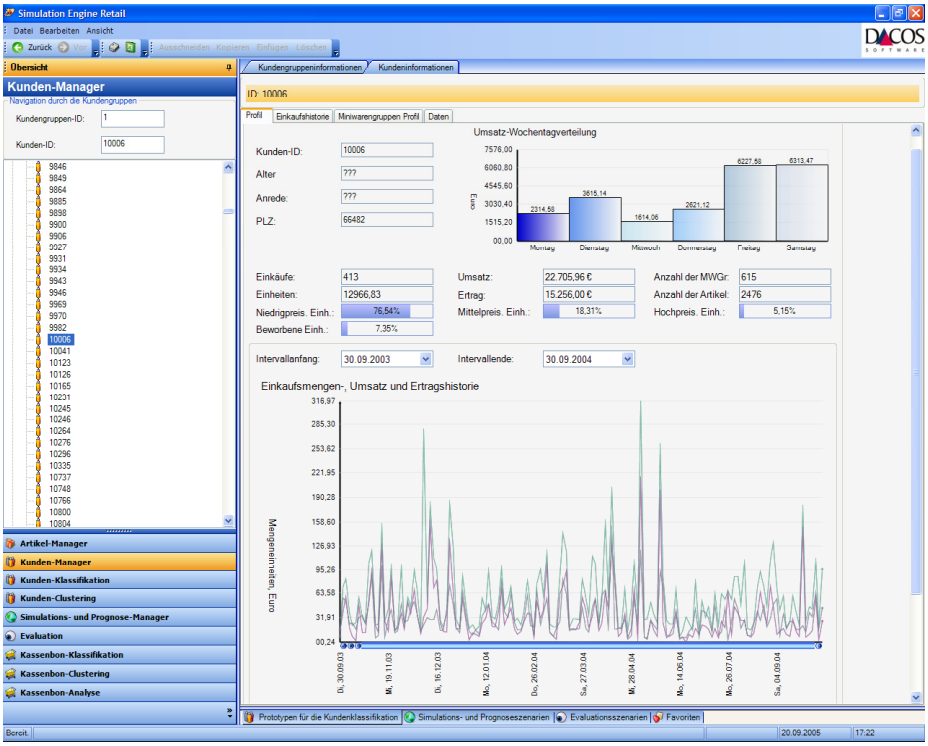


Abbildung 56: Einzelnes Kundenprofil im SimMarket Kundenmanager[1]

Im Vordergrund stehen dabei vor allem Gesamtanzahl der durchgeführten Einkäufe, Gesamtabsatzmenge, Gesamtumsatz und -ertrag sowie deren Historien im zeitlichen Verlauf. Eine weitere Information ist die kundenbezogene Umsatz-Wochentagsverteilung, die einen Anhaltspunkt gibt, welche Wochentage vom Kunden als Einkaufstage präferiert werden. Ebenso liefern Kennzahlen, wie die Anzahl unterschiedlicher Warengruppen, von denen der Kunde mindestens ein Produkt erworben hat sowie die Anzahl unterschiedlicher Artikel einen guten Überblick über die kundenbezogene *Sortimentsbreite*. Die Sortimentsbreite gibt einen Hinweis darauf, welche Art von Einkäufen der Kunde in der zugrunde liegenden Filiale tätigt. Beispielsweise, ob er Großeinkäufe oder Ergänzungskäufe in der Filiale durchführt, also ob er nahezu seinen kompletten Bedarf innerhalb des Filialangebotes deckt oder ob er nur wenige Artikel gezielt erwirbt. Bei Artikeln unterscheide ich zwischen niedrig-, mittel- und hochpreisigen Produkten, so dass die persönliche Preiskategorie-Verteilung einen Anhaltspunkt über die Preissensibilität des Kunden bietet. Analog gebe ich als „grobe“ Kennzahl der Werbesensibilität den Anteil der zum Kaufzeitpunkt beworbenen Artikel an.

[1] Der dargestellte Ertrag entspricht aus Datenschutzgründen nicht der Realität, da die Ertragsrechnung auf fiktiven Einkaufspreisen basiert

Sämtliche kundenbezogenen Kennzahlen des Faktenwissens bzw. des Kundenprofils werden aus der Einkaufshistorie des Kunden ermittelt, die in Form der kundenbezogenen Kassenbonhistorie ebenfalls als Faktenwissen gespeichert ist. Abbildung 57 zeigt einen konkreten einzelnen Kassenbon aus der Einkaufshistorie des Kunden.

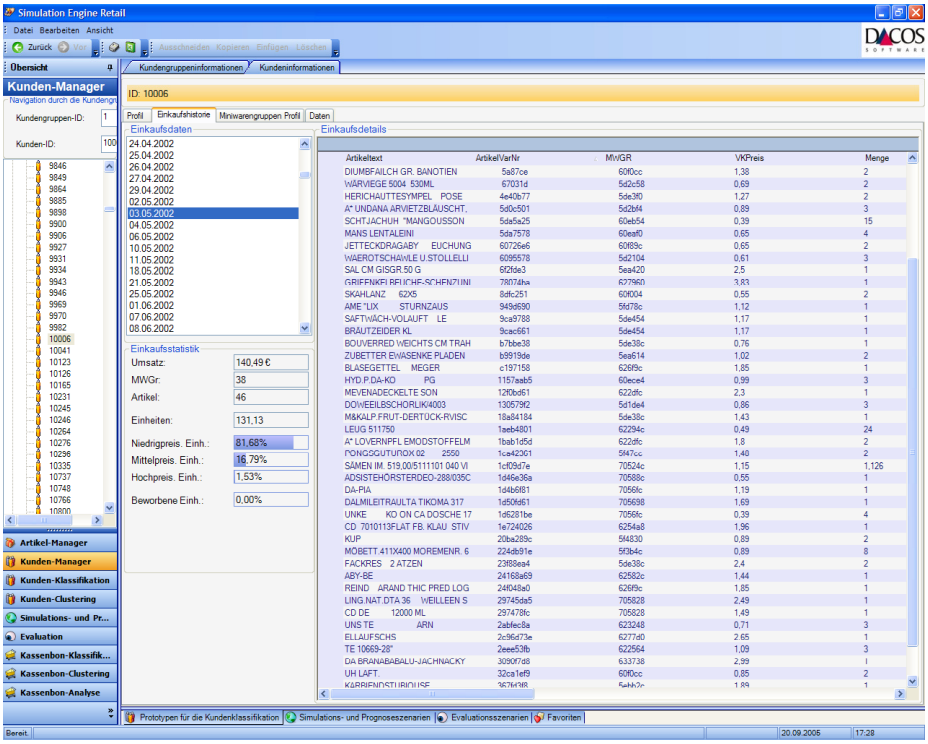


Abbildung 57: Einkaufshistorie in Form einzelner datierter Kassenbons[1]

Zur Übersicht wird eine Auswahl von Kennzahlen bezüglich des ausgewählten Bons berechnet, die inhaltlich den oben beschriebenen Kennzahlen des Kundenprofils entsprechen: Umsatz, Anzahl von Artikeleinheiten, unterschiedliche Warengruppen und unterschiedliche Produkte sowie die Verteilung der Preiskategorien und der Anteil beworbener Artikel. Darüber hinaus werden die einzelnen Bonpositionen (Transaktionen) angezeigt, wobei für jeden Artikel dessen Bezeichnung, Artikelnummer, Warengruppe, Preis, Menge und Bewerbungsmaßnahmen sowie weitere Artikelmerkmale angegeben sind.

Auf Basis der einzelnen Kassenbons werden, neben den eben beschriebenen einfachen

[1] Die Artikelbezeichnungen sowie Artikel- und Warengruppennummern wurden aus Gründen des Datenschutzes im Rahmen dieser Veröffentlichung anonymisiert, liegen dem Forschungsteam allerdings in unverschlüsselter Form vor

statistischen Kennzahlen, auch Verhaltensmodelle erlernt, die als Verhaltensnetze im probabilistischen Strukturwissen des Kundenagenten gespeichert werden. Da sich das Kaufverhalten eines Kunden bezüglich unterschiedlicher Warengruppen bzw. Artikel stark unterscheiden kann, basiert die Modellierung des Kaufverhaltens auf der in Kapitel 3.2.4 erläuterten Einzelartikel-Feature-Verhaltens-Matrix. Das bedeutet, dass für jeden Artikel, den ein Kunde in ausreichender Menge und Häufigkeit gekauft hat, bei Bedarf ein kundenindividuelles Artikelverhaltensnetz aus den historischen Kundendaten erlernt wird, das aus Feature-Netzen (siehe Kapitel 3.2.3) zusammengesetzt ist, die die jeweiligen Feature-Ausprägungen des Kunden repräsentieren.

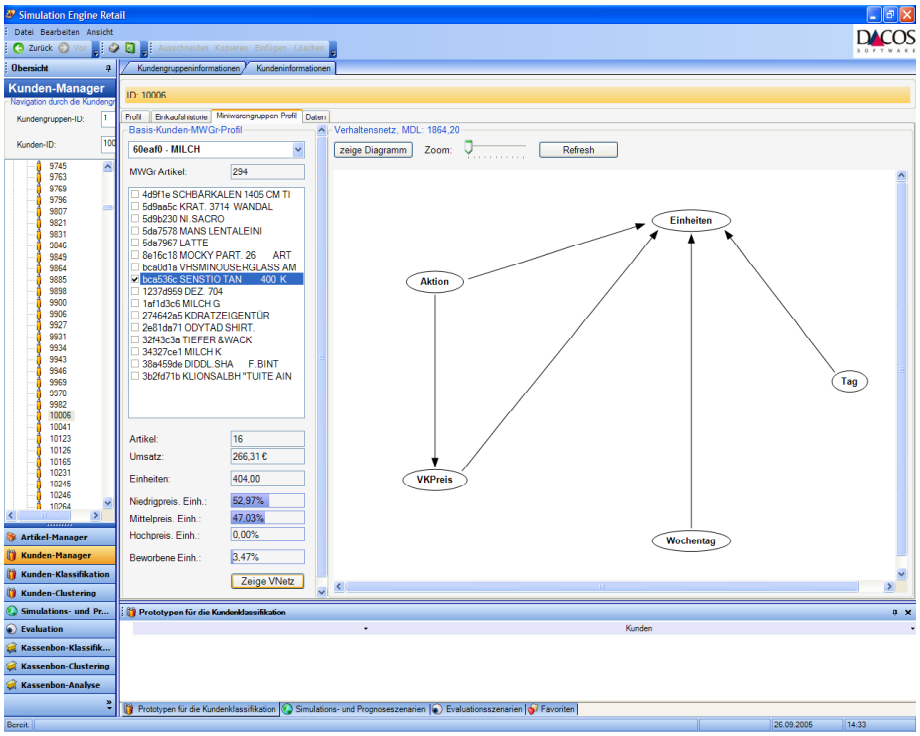


Abbildung 58: Einfaches kundenindividuelles Artikelverhaltensnetz

Abbildung 58 zeigt die graphische Repräsentation eines solchen kundenindividuellen Artikelverhaltensnetzes, das durch die Verschmelzung einfacher Preis-, Promotion- und Termin-Feature-Netze entstanden ist. Das Netz repräsentiert das Verhalten des ausgewählten Kunden bezogen auf ein bestimmtes Produkt der Warengruppe „Milch“ unter der Berücksichtigung der kundenindividuellen Ausprägungen der Features „Preis“, „Promotion“ und „Termin“, die anhand der historischen Kundendaten erlernt wurden.

Konkret modelliert das Netz wie viele Einheiten des ausgewählten Milchproduktes der Kunde in Abhängigkeit von Artikelpreis, Werbemaßnahmen und zeitlichen Einflüssen, wie dem Wochentag und Tag des Monats, erwartungsgemäß kaufen wird. Die Erwartungswerte

bzw. Randwahrscheinlichkeitsverteilungen der Zufallsvariablen sind in der so genannten *Diagrammansicht* in Abbildung 59 zu sehen.

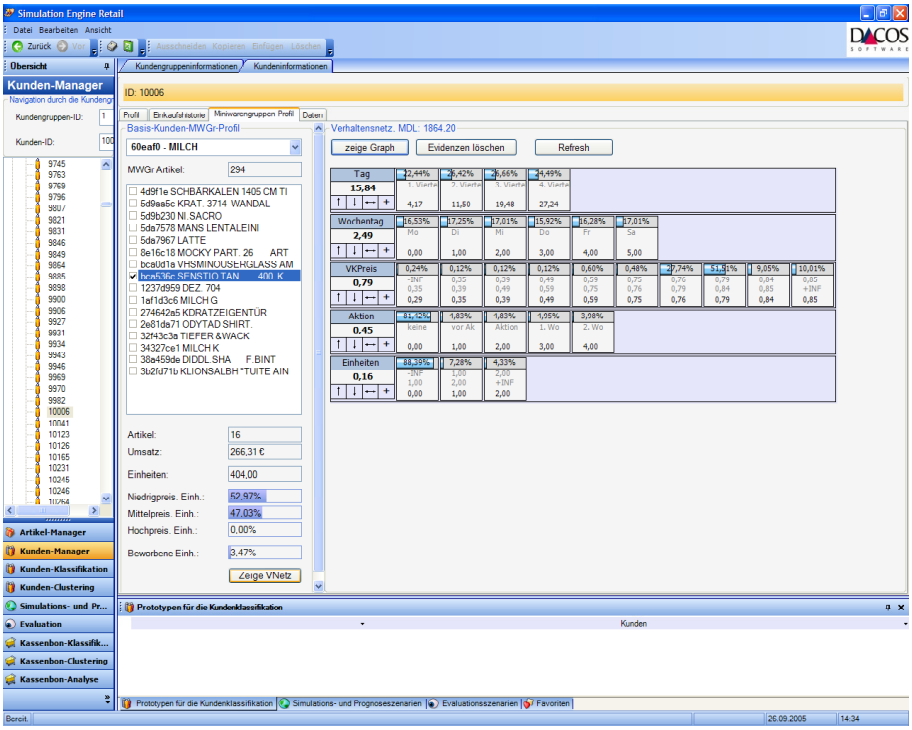


Abbildung 59: Diagrammansicht des kundenindividuellen Artikelverhaltensnetzes

In dieser Ansicht entspricht jede Zeile einer Zufallsvariable des Netzes (ersichtlich durch die entsprechende Namensgebung auf den Labels der linken Seite der jeweiligen Zeile). Rechts neben den Labels befinden sich die möglichen diskreten Zustände der jeweiligen Zufallsvariablen und die zugehörigen Rand- bzw. Apriori-Wahrscheinlichkeitsverteilungen. Jeder Zustand besitzt entweder eine *Intervallangabe* (beispielsweise besitzt der Knoten „Einheiten“ einen Zustand für Mengen innerhalb des Intervalls [1;2[) oder einen *Zustandsnamen* (beispielsweise hat die Zufallsvariable „Wochentag“ einen Zustand „Dienstag“) sowie einen *Mittelwert*, der sich aus den Werten berechnet, die tatsächlich in das entsprechende Intervall fallen.

Darüber hinaus besitzt jeder Zustand eine Wahrscheinlichkeit, die zusammen mit den Wahrscheinlichkeiten der anderen Zustände die Wahrscheinlichkeitsverteilung des Knotens bildet. Der auf der aktuellen Wahrscheinlichkeitsverteilung basierende Erwartungswert der Zufallsvariable wird unterhalb des entsprechenden Labels angezeigt, wobei diese Angabe nicht bei allen Knoten einen direkten Sinn ergibt (beispielsweise bei den Zufallsvariablen „Tag“ oder „Wochentag“). Der Erwartungswert der Zufallsvariablen „Einheiten“ hingegen spielt eine entscheidende Rolle bei der Modellierung des artikelbezogenen Kaufverhaltens.

Sie gibt die Anzahl der Artikeleinheiten an, die der Kunde erwartungsgemäß, unter durch Evidenzen vorgegebene Umstände, erwerben wird. Beispielsweise kauft der Kunde pro Tag im Durchschnitt 0,16 Einheiten des Milchproduktes für einen durchschnittlichen Preis von 0,79 Euro (Erwartungswert der Zufallsvariablen „VKPreis“).

Mit Hilfe der Diagrammansicht lassen sich einfache prognostische und diagnostische Anfragen an das Netz stellen, um Erkenntnisse über das individuelle Kaufverhalten des Kunden bezüglich des selektierten Einzelartikels zu erhalten. Anfragen können durch Setzen von Evidenzen bzw. Evidenzkombinationen an entsprechenden Zufallsvariablen vorgenommen werden. Dabei werden sowohl harte als auch weiche Evidenzen unterstützt (vgl. Kapitel 4.2). Abbildung 60 zeigt das Setzen einer harten Evidenz auf den Zustand „0,29 Euro“ der Zufallsvariablen „VKPreis“, wodurch sich durch Inferenz der Erwartungswert von 0,91 der Zufallsvariablen „Einheiten“ ergibt. Das bedeutet, dass der Kunde bei einem Preis von 0,29 Euro durchschnittlich 0,91 Produkteinheiten kauft.

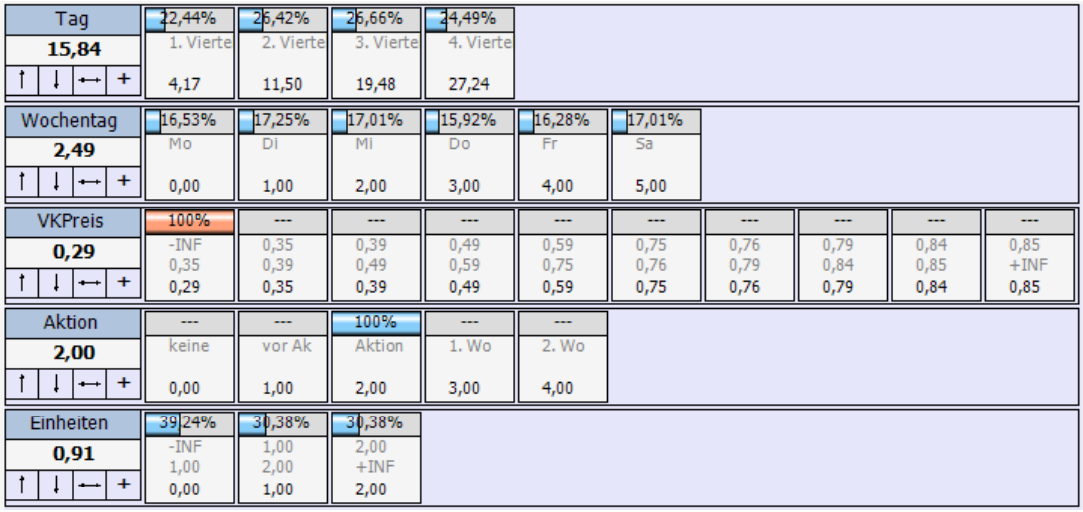


Abbildung 60: Setzen einer harten Evidenz auf die Zufallsvariable „VKPreis“

Während harte Evidenzen auf die Werte der Zustände einer Variable beschränkt sind, können mit Hilfe weicher Evidenzen auch Anfragen bezüglich von Werten gestellt werden, die nicht unbedingt in der Lernmenge des Netzes vorgekommen sein müssen. Beispielsweise können weiche Evidenzen genutzt werden, um Verkaufspreise von Artikeln zu setzen, zu denen ein Kunde keine Einkäufe getätigt hat. Abbildung 61 zeigt eine weiche Evidenz mit einem Erwartungswert von 0,77 für die Zufallsvariable „VKPreis“, die aus einer Wahrscheinlichkeitsverteilung bezogen auf die Zustände 0,76 und 0,79 besteht. Nach Inferenz ergibt sich ein Erwartungswert von 0,07 bei der Variable „Einheiten“, d. h. bei einem Verkaufpreis von 0,77 Euro kauft der Kunde durchschnittlich 0,07 Einheiten pro Tag.

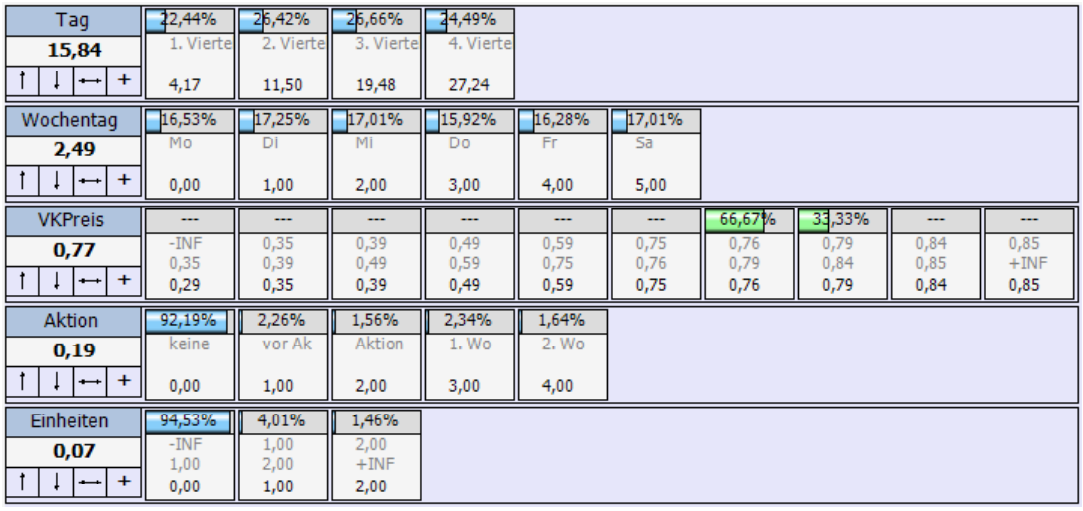


Abbildung 61: Setzen einer weichen Evidenz auf die Zufallsvariable „VKPreis“

Anfragen können auch aus mehreren Evidenzen bestehen. Abbildung 62 zeigt die Kombination von drei harten Evidenzen auf die Knoten „Wochentag“, „VKPreis“ und „Aktion“: Während einer Bewerbung (Aktion) kauft der Kunde das ausgewählte Milchprodukt samstags bei einem Verkaufspreis von 0,29 Euro durchschnittlich einmal.

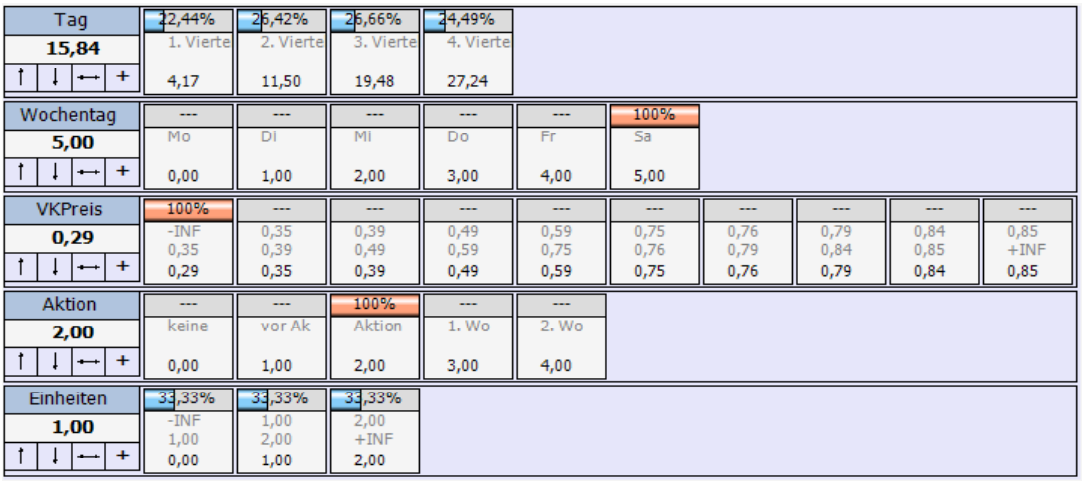


Abbildung 62: Setzen einer Kombination von harten Evidenzen

Neben prognostischen Anfragen lassen sich mit Hilfe des kundenindividuellen Artikelverhaltensnetzes auch diagnostische Fragestellungen beantworten. Beispielsweise klärt die diagnostische Anfrage in Abbildung 63 die Umstände, unter denen der Kunde keine einzige Einheit des Produktes kauft (Evidenz auf den Zustand „0,00“ der Zufallsvariable „Einheiten“). Die wichtigsten Gründe scheinen dabei eine fehlende Bewerbung (86,96 Prozent beim Zustand „keine“ der Zufallsvariable „Aktion“) und ein Verkaufspreis von 0,76

bzw. 0,79 Euro und höher zu sein (30,84 bzw. 50,73 Prozent bei Zustand 0,76 bzw. 0,79 des Knotens „VKPreis“). Ferner ist die Wahrscheinlichkeit eines Nichtkaufes im dritten und vierten „Quartal“ eines Monats wahrscheinlicher als im ersten oder zweiten.

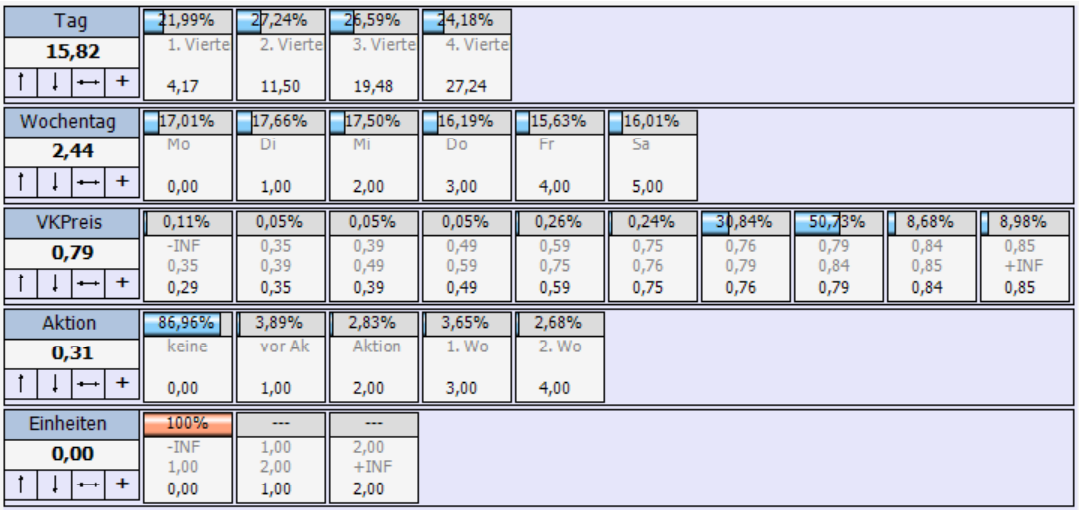


Abbildung 63: Diagnostische Anfrage durch Setzen einer Kombination von harten Evidenzen

Zusammenfassend lässt sich feststellen, dass kundenindividuelle Artikelverhaltensnetze geeignet sind, das Kaufverhalten eines Kunden bezüglich eines einzelnen Artikels in Abhängigkeit verschiedener Einflussfaktoren zu modellieren. Allerdings wird dabei der Einfluss konkurrierender, beeinflussender oder abhängiger Artikel nicht ausreichend berücksichtigt. Um diese Faktoren zu berücksichtigen, müssen mehrere kundenindividuelle Artikelverhaltensnetze zu so genannten kundenindividuellen Artikelgruppenverhaltensnetzen zusammengefügt werden, in denen eventuelle Abhängigkeiten zwischen den Artikeln quantitativ repräsentiert werden können (siehe Kapitel 3.2.5).

Abbildung 64 zeigt ein kundenindividuelles Artikelgruppenverhaltensnetz bezüglich zweier konkurrierender Milchprodukte, das aus der Verschmelzung der beiden kundenindividuellen Artikelnetze entstanden ist. Dabei stehen die Knoten mit der Namenserweiterung „_2“ für die auf das zweite Produkt bezogenen Einflüsse („VKPreis“ und „Aktion“) und Effekte („Einheiten“). Die Abhängigkeit zwischen den beiden Artikeln besteht darin, dass sowohl der Verkaufspreis als auch die Bewerbungsmaßnahme des zweiten Artikels Auswirkungen auf die verkauften Einheiten des ersten Produktes hat, was durch entsprechende Kanten zwischen den Knoten ersichtlich ist. Umgekehrt scheint nur die Bewerbung des ersten Artikels einen signifikanten Einfluss auf den kundenindividuellen Abverkauf des zweiten Artikels zu haben, da zwischen den Knoten „VKPreis“ und „Einheiten_2“ keine Kante existiert.

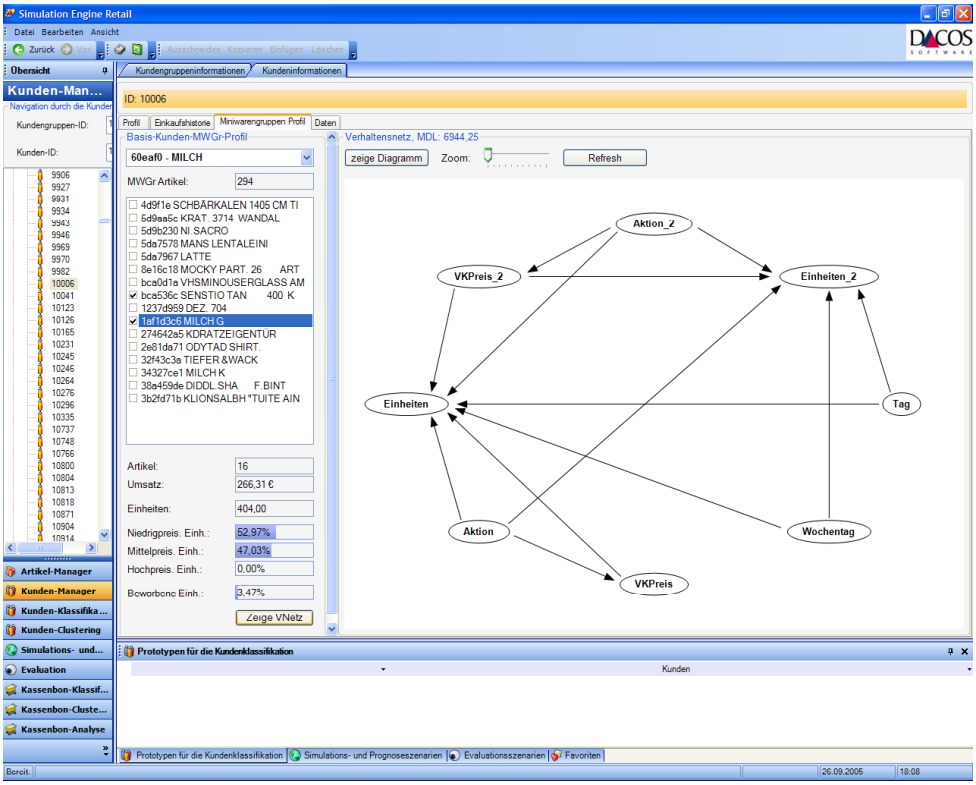


Abbildung 64: Kundenindividuelles Artikelgruppenverhaltensnetz

Mit Hilfe der Diagrammansicht lassen sich nun quantifizierbare, kundenindividuelle Abhängigkeitsanalysen - bezogen auf die beiden selektierten Milchprodukte - durch Setzen entsprechender Evidenzkombinationen bzw. einer Sequenz von Evidenzkombinationen durchführen.

Abbildung 65 zeigt einen Ausschnitt der Diagrammansicht, auf der im ersten Schritt für beide Artikel jeweils eine harte Evidenz auf den Zustand „keine“ der Zufallsvariablen „Aktion“ bzw. „Aktion_2“ gesetzt wurde. Unter diesen Umständen kauft der Kunde im Durchschnitt 0,35 Einheiten des ersten Artikels, während er von Artikel 2 durchschnittlich 0,02 Einheiten erwirbt. In Abbildung 66 wird durch eine entsprechende harte Evidenz bei der Zufallsvariable „Aktion_2“ eine Situation simuliert, in der Artikel 2 nun beworben wird, während Artikel 1 unbeworben bleibt. Dadurch erhöht sich nicht nur der kundenindividuelle Absatz von Artikel 2 von durchschnittlich 0,02 auf 0,43 Einheiten, sondern ebenso der durchschnittliche Absatz von Artikel 1 von 0,35 auf 0,66 Einheiten. Die Bewerbung von Artikel 2 übt also einen „Pull-Effekt“ auf Artikel 1 aus, d. h. sie „zieht“ den Absatz von Artikel 1 indirekt mit nach oben. Falls die Bewerbung von Artikel 2 den Absatz von Artikel 1 sinken ließe, läge ein so genannter „Kannibalisierungseffekt“ vor, der in der Praxis häufig vorkommen kann (in Kapitel 6.2.2 zeige ich im Rahmen der Modellierung des

Kaufverhaltens von Kundengruppen ein entsprechendes Beispiel).

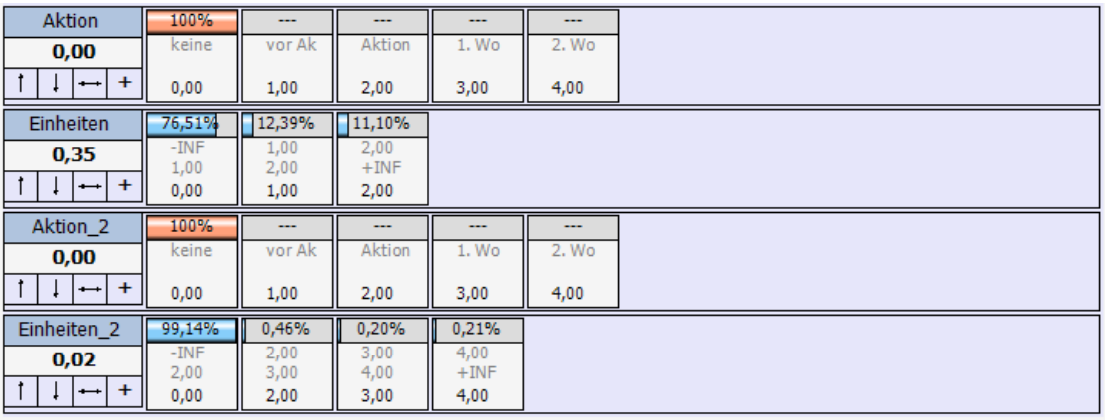


Abbildung 65: Beide Artikel des Artikelgruppenverhaltensnetzes sind unbeworben

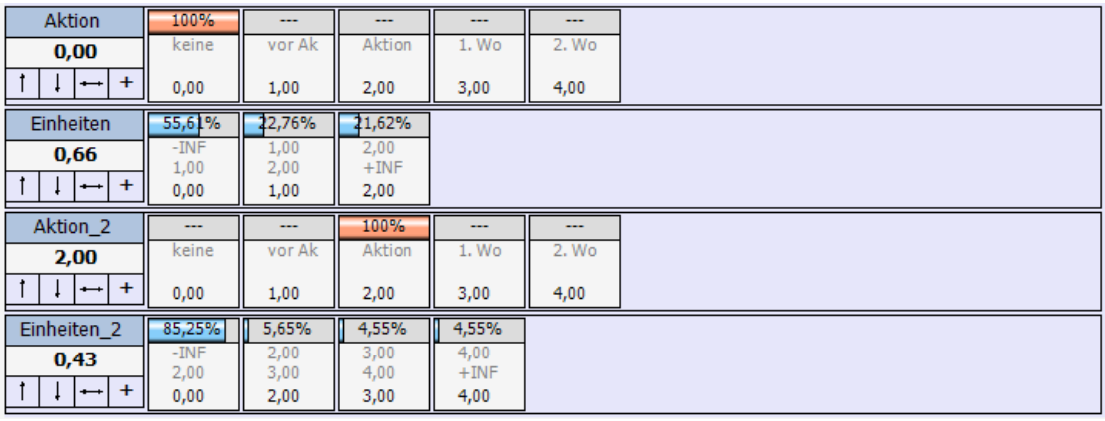


Abbildung 66: Die Bewerbung von Artikel 2 steigert auch den Absatz von Artikel 1

6.2.2 Modellierung von Kundengruppen

Kundengruppen werden im SimMarket System als Kundengruppenagenten repräsentiert, die durch unterschiedliche Arten der Holonisierung aus einzelnen Kundenagenten gebildet werden (siehe Kapitel 5.1). Bei einer Kundengruppe in Form einer kopfgesteuerten Agentengesellschaft werden die meisten kundengruppenbezogenen Kennzahlen bei Bedarf durch Aggregation der entsprechenden Merkmalsausprägungen der einzelnen Kunden ermittelt. Analog besteht die Modellierung des Kaufverhaltens in diesem Fall aus der Menge der einzelnen kundenindividuellen Verhaltensnetze. Für einige Anwendungsfälle eignet sich diese Modellierung in der Praxis allerdings nicht, da es zu aufwändig bzw. nicht notwendig ist, alle Kunden einzeln zu betrachten. Darüber hinaus kann bei manchen Fragestellungen die

Datengrundlage für einzelne Kunden zu „dünn“ sein, beispielsweise bei so genannten „Langsamdrehern“, also Produkten, die sich nicht häufig verkaufen, wie beispielsweise Rasierapparate. Aus diesem Grund werden Kundengruppenagenten zusätzlich durch Durchschnitts- oder Additionsverschmelzung einzelner Kundenagenten realisiert. Das bedeutet, dass eine Kundengruppe nicht primär durch die Menge der einzelnen Kundenagenten, sondern durch einen eigenständigen Kundengruppenagenten repräsentiert wird, der aus aggregierten Werten der Einzelkunden erzeugt wird.

Im Falle der Durchschnittsverschmelzung bestehen die Kennzahlen des Kundengruppenagenten aus dem Durchschnitt der Kennzahlenwerte der Einzelkunden. Analog werden die kundengruppenbezogenen Verhaltensnetze mit Hilfe einer Durchschnittsverschmelzung der kundenindividuellen Verhaltensnetze erzeugt. Im Falle der Additionsverschmelzung werden die Kennzahlenwerte sämtlicher Kunden „summiert“ und die kundengruppenbezogenen Verhaltensnetze mit Hilfe einer Additionsverschmelzung der kundenindividuellen Verhaltensnetze erzeugt.

Im Kundenmanager werden Kundengruppen analog zu einzelnen Kunden visualisiert. Abbildung 67 zeigt das Profil einer ausgewählten Kundengruppe „Topkunden“, deren Modellierung aus der Additionsverschmelzung ihrer 2.900 Mitglieder besteht.

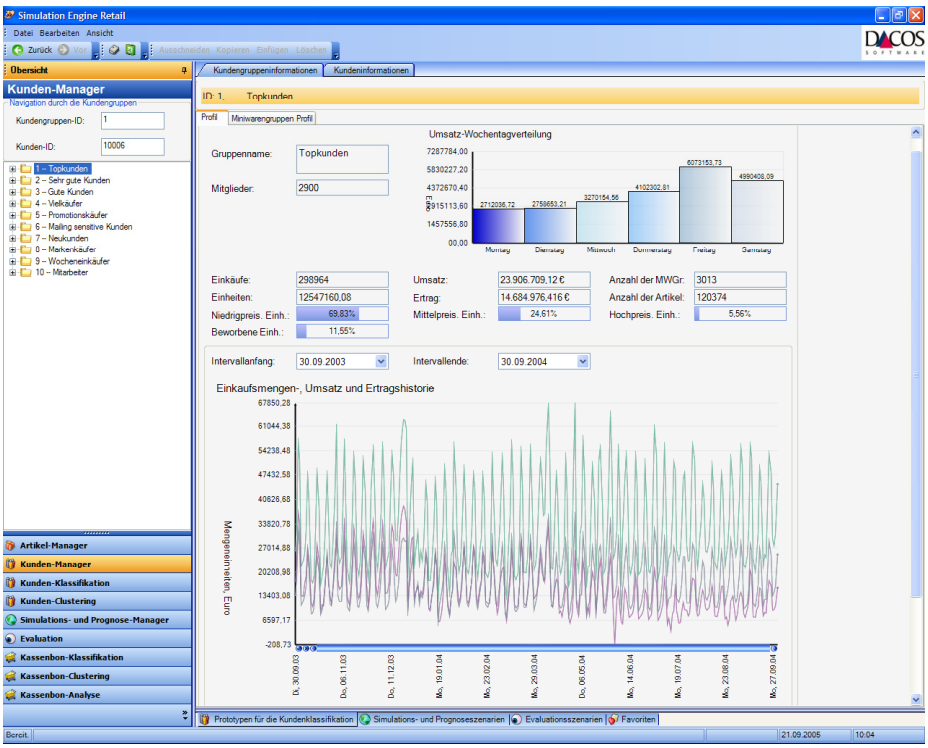


Abbildung 67: Profil einer ausgewählten Kundengruppe im SimMarket Kundenmanager

Im Mittelpunkt stehen wiederum einfache statistische Kennzahlen wie Anzahl der Einkäufe, Gesamtumsatz und -ertrag sowie deren Historien im Zeitverlauf, die als Basis für Kundengruppenwertkennzahlen dienen. Darüber hinaus vermitteln sowohl die Umsatz-Wochentagsverteilung als auch die Verteilung der gekauften Produkte auf die verschiedenen Preiskategorien sowie der Anteil der beworbenen Artikel und die Anzahl unterschiedlicher Warengruppen bzw. Artikel einen Überblick über die aggregierte Verhaltensweise der Kundengruppe.

Die Visualisierung der kundengruppenbezogenen Verhaltensnetze ist identisch zur graphischen Repräsentation der kundenindividuellen Verhaltensnetze. Beispielsweise zeigt Abbildung 68 ein kundengruppenbezogenes Artikelgruppenverhaltensnetz bezüglich zweier konkurrierender Milchprodukte. Die Bewerbung (Aktion) einer der beiden Artikel hat jeweils eine Auswirkung auf den kundengruppenbezogenen Absatz (Einheiten) des anderen Produktes. Allerdings scheint nur der Verkaufspreis des zweiten Artikels einen Einfluss auf den Abverkauf des ersten zu haben, während dies umgekehrt nicht der Fall zu sein scheint. Um eine quantifizierbare Abhängigkeitsanalyse durchzuführen, kann analog zu dem in Kapitel 6.2.1 gezeigten Beispiel eine Sequenz von Anfragen in Form entsprechender Evidenzkombinationen an das Verhaltensnetz gestellt werden.

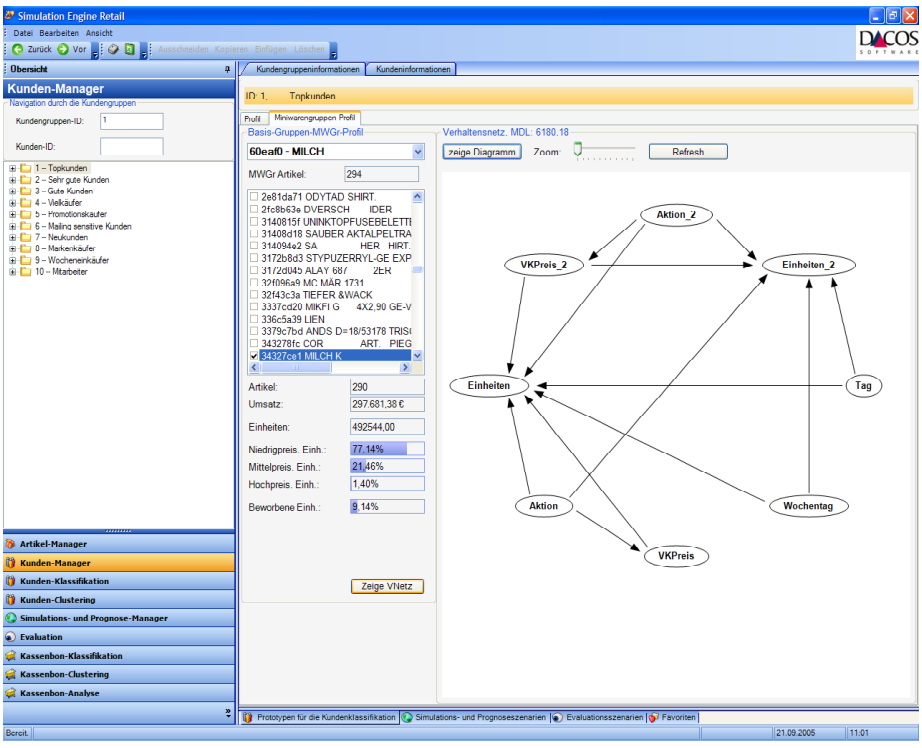


Abbildung 68: Kundengruppenbezogenes Artikelgruppenverhaltensnetz

Abbildung 69 zeigt den relevanten Ausschnitt der Diagrammansicht des kundengruppenbezogenen Artikelgruppenverhaltensnetzes. Zur Abhängigkeitssimulation der beiden Artikel werden zunächst bei beiden Artikeln deren Maximalpreise durch das Setzen entsprechender harter Evidenzen vorgegeben. Bei einem Verkaufspreis von 0,55 Euro ergibt sich beim ersten Artikel ein durchschnittlicher kundengruppenbezogener Abverkauf von 43,69 Einheiten pro Tag, während die Kundengruppe bei einem Verkaufspreis von 0,86 Euro im Durchschnitt nur 9,89 Einheiten des zweiten Artikels erwirbt.

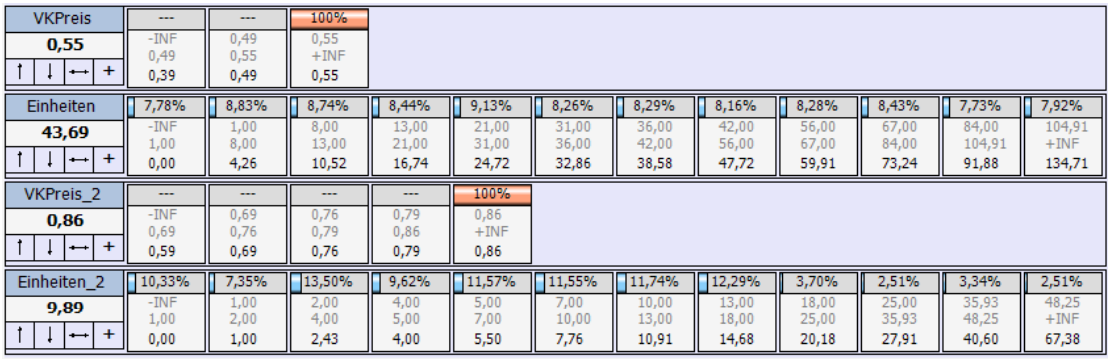


Abbildung 69: Beide Artikel des holonischen Verhaltensnetzes nehmen den höchsten Preis an

Anschließend wird der Verkaufspreis des zweiten Artikels von 0,86 auf 0,59 Euro gesenkt, während der Verkaufspreis des ersten Artikels konstant bleibt (siehe Abbildung 70). Die deutliche Preissenkung des zweiten Artikels bewirkt dessen Absatzsteigerung von 9,89 auf 23,16 Einheiten. Allerdings sinkt gleichzeitig der kundengruppenbezogene Absatz des ersten Artikels drastisch von 43,69 auf 13,16 Einheiten, d. h. es liegt ein deutlicher „Kannibalisierungseffekt“ vor.

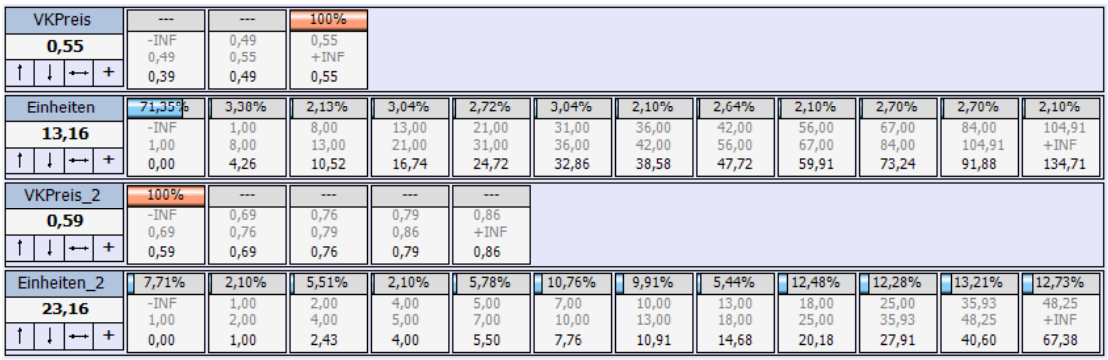


Abbildung 70: Eine Preissenkung bei Artikel 2 senkt den Absatz von Artikel 1

6.3 Beispiele und Evaluation der Kundensimulation

Im Folgenden präsentiere ich die konkrete Umsetzung der szenariobasierten Simulation des kundengruppenbezogenen Konsumentenverhaltens innerhalb des SimMarket Systems. In Unterkapitel 6.3.1 beschreibe ich zunächst die Vorgehensweise zur Definition und Erstellung von Simulationsszenarien mit Hilfe eines Anwendungsbeispiels und führe anschließend eine szenariobasierte Simulation des Kaufverhaltens zweier unterschiedlicher Kundengruppen durch. Unterkapitel 6.3.2 erläutert die allgemeine Vorgehensweise zur Evaluation von Simulationsergebnissen anhand eines Evaluationsbeispiels und präsentiert abschließend die Ergebnisse einiger kundengruppenbezogener Evaluierungen.

6.3.1 Szenariobasierte Kundengruppensimulation

Die szenariobasierte Simulation des Konsumentenverhaltens von Kundengruppen dient unter anderem zur Unterstützung der Entwicklung geeigneter kundengruppenbezogener Marketing-Strategien und entsprechenden Maßnahmen. Im Folgenden verdeutliche ich die Vorgehensweise anhand eines Anwendungsbeispiels, bei dem ein Anwender drei unterschiedliche Alternativen zur Gestaltung der Preise und Werbemaßnahmen bezüglich eines konkreten Kaffeeproduktes entwickelt hat und nun wissen möchte, wie sich die unterschiedlichen Szenarien auf die beiden Kundengruppen „Topkunden“ und „Gute Kunden“ auswirken würden, um entscheiden zu können, welche Alternative vorzuziehen ist.

Im ersten Schritt wählt der Benutzer das entsprechende Produkt im Artikelmanager des SimMarket Systems aus und definiert die drei alternativen Szenarien mit Hilfe einer Eingabemaske (siehe Abbildung 71). Alle drei Szenarien beziehen sich auf die Planung des Verkaufspreises und der Bewerbung eines Kaffeeproduktes über den Zeitraum von zwei Wochen (02.08.2004 bis 14.08.2004), die aus der Sicht des Anwenders in der Zukunft liegen (nehmen wir an, der Benutzer führt die Planung am Nachmittag des 01.08.2004 durch). Die drei Szenarien sind - wie in Abbildung 71 ersichtlich - folgendermaßen definiert:

- Szenario 1:
  - 02.08.2004 – 07.08.2004: Verkaufspreis 3,79 € ohne Bewerbung
  - 09.08.2004 – 14.08.2004: Preissenkung auf 3,69 € ohne Bewerbung
- Szenario 2:
  - 02.08.2004 – 07.08.2004: Verkaufspreis 3,79 € ohne Bewerbung
  - 09.08.2004 – 14.08.2004: Preissenkung auf 3,33 € ohne Bewerbung
- Szenario 3:
  - 02.08.2004 – 07.08.2004: Verkaufspreis 3,79 € ohne Bewerbung
  - 09.08.2004 – 14.08.2004: Preissenkung auf 3,33 € mit Bewerbung

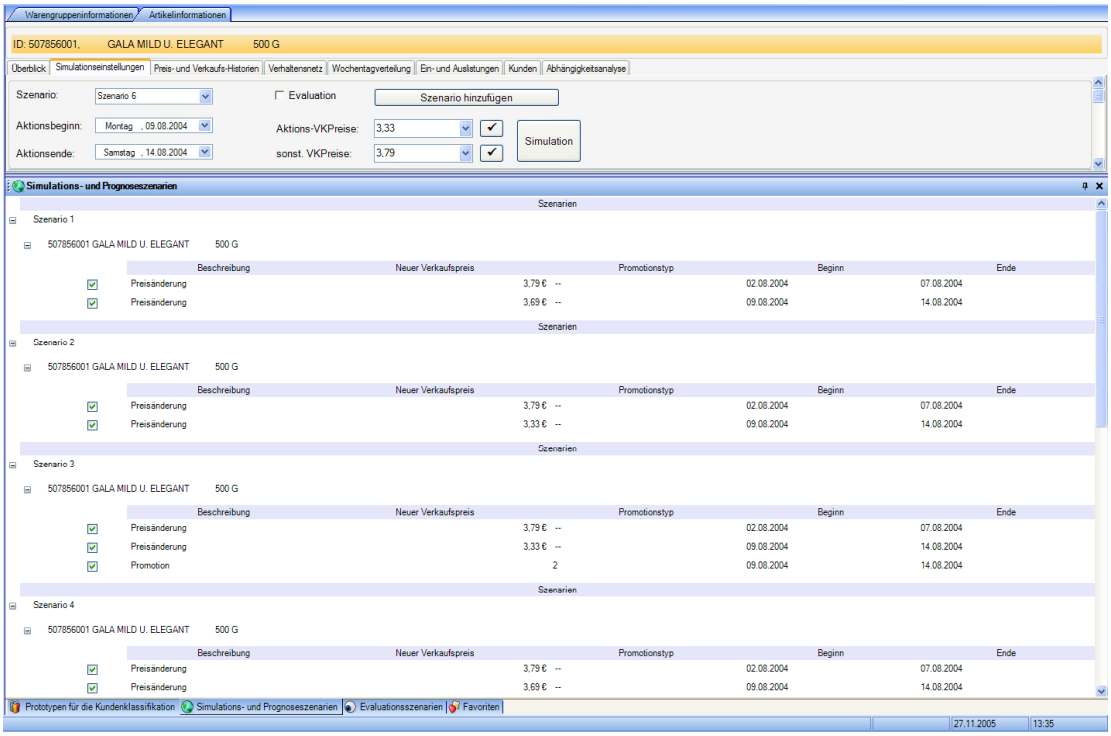


Abbildung 71: Definition der alternativen Simulationsszenarien

Um die Ergebnisse der Simulation des Konsumentenverhaltens der beiden unterschiedlichen Kundengruppen bequem vergleichen zu können, wird für jede der beiden Kundengruppen eine eigene Instanz der drei Szenarien erstellt (siehe Abbildung 72 auf der linken Seite), d. h. das Verhalten der „Topkunden“ wird bezüglich der Szenarien 1, 2 und 3 simuliert, während den „Guten Kunden“ die Szenarien 4, 5 und 6 zugeordnet werden.

Neben den Kundengruppen/Szenario-Zuordnungen können mit Hilfe der Eingabemaske des Simulations- und Prognosemanagers auch Lernzeitraum, Simulationsintervall und die der Simulation zugrunde liegende Zeitbasis ausgewählt werden (siehe rechte Seite der Abbildung 72). Im Beispiel wurde der Zeitraum vom 02.01.2002 bis zum 01.08.2004 als Lernbasis definiert, während der Simulationszeitraum auf den Zeitraum der Maßnahmen beschränkt wurde. Durch die Vorgabe der Zeitbasis von einem Tag wurde eine tagesgenaue Simulation definiert.

Nachdem die Simulationseinstellungen vorgenommen wurden, kann die eigentliche Simulation der definierten Szenarien gestartet werden. Während der Simulation reagieren die beiden Kundengruppenagenten auf die ihnen zugeordneten Szenarien. Das bedeutet, dass für jeden Tag des zu simulierenden Zeitraumes ermittelt wird, wie viele Einheiten die beiden Kundengruppen unter den vorgegebenen Umständen erwartungsgemäß erwerben würden. Dies geschieht durch Setzen der entsprechenden Evidenzkombinationen und anschließender probabilistischer Inferenz. Der tagesgenaue Absatz der jeweiligen Kundengruppe ist dabei

der Erwartungswert der Variable „Einheiten“, auf dessen Basis anschließend der kundengruppenbezogene Umsatz sowie der Ertrag berechnet werden.

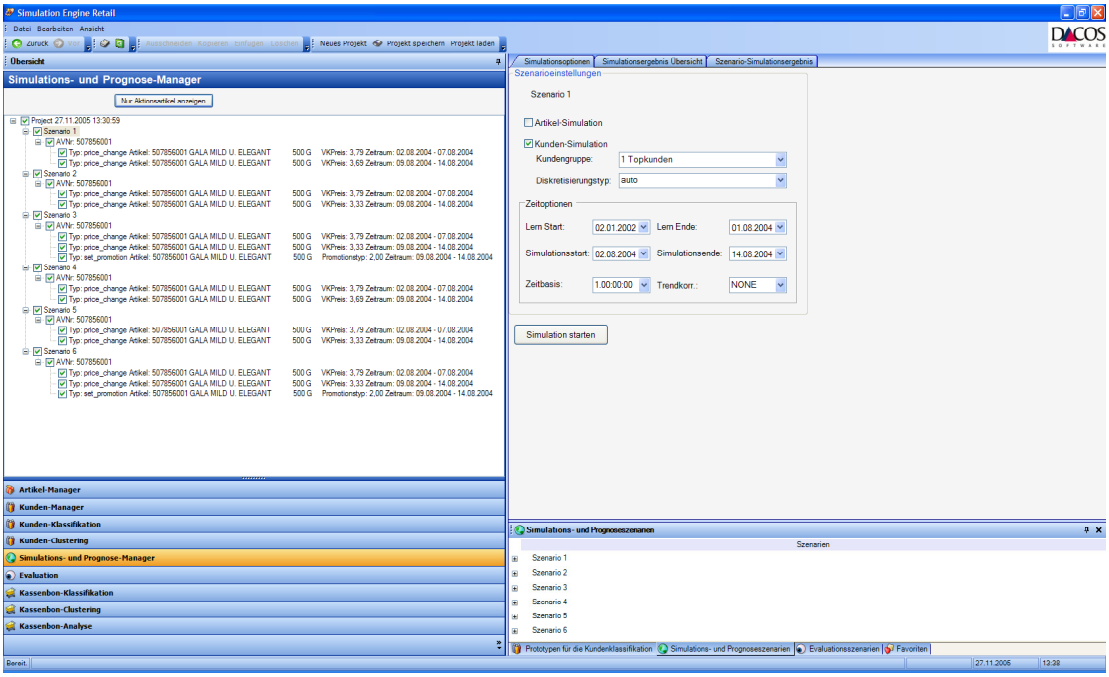


Abbildung 72: Simulationseinstellungen

Abbildung 73 zeigt für die im Beispiel definierten Szenarien die aggregierten Simulationsergebnisse als kundengruppenbezogene Absatzmenge, Umsatz und Ertrag.

Szenario	Verkaufsmenge (sim)	Umsatz (sim)	Ertrag (sim)
Szenario 1	30,44	112,99	15,59
Szenario 2	63,84	215,71	11,41
Szenario 3	75,67	254,20	12,06
Szenario 4	22,56	83,77	11,56
Szenario 5	51,43	173,58	9,02
Szenario 6	61,29	206,63	10,49

Abbildung 73: Simulationsergebnisse

Die aggregierten Simulationsergebnisse der beiden Kundengruppen sind tendenziell sehr ähnlich. Die Verkaufsmenge und der Umsatz in Szenario 2 bzw. 5 sind aufgrund der stärkeren Preissenkung etwas mehr als doppelt so hoch wie in Szenario 1 bzw. 4. Eine zusätzliche Bewerbung in Szenarien 3 und 6 erhöht Absatz und Umsatz um ca. weitere 19 Prozent. Die Ergebnisse der Ertragsrechnung mit einem fiktiven Einkaufpreis von 3,20 Euro zeigen jedoch, dass sich weder die stärkere Preissenkung noch die zusätzliche Bewerbung lohnen, da der Ertrag beider Kundengruppen im Falle der schwächeren Preissenkung höher ausfällt, zumal im Rahmen einer Bewerbung zusätzliche Kosten anfallen. Insgesamt bedeutet das Simulationsergebnis, dass der Sortimentsverantwortliche sich in diesem Falle für Szenario 1 bzw. 4 entscheiden sollte, d. h. den Verkaufspreis des Produktes nur leicht von 3,79 auf 3,69 Euro senken sollte. Die weitere Vorgehensweise könnte darin bestehen, alternative Szenarien mit veränderten Preissenkungen bzw. Bewerbungen zu erstellen und darüber hinaus Kannibalisierungs- sowie Push- und Pulleffekte innerhalb der Warengruppe zu berücksichtigen, um Absatzmengen-, Umsatz- und Ertragsveränderungen der gesamten Warengruppe bzw. der direkt sowie indirekt betroffenen Artikel zu betrachten.

Obwohl die Verhaltensweisen der beiden Kundengruppen in der aggregierten Ansicht der Simulationsergebnisse tendenziell sehr ähnlich erscheinen, können bei der Betrachtung der wochenbezogenen Resultate kleinere Unterschiede festgestellt werden (siehe Diagramm 1 und Diagramm 2). Während die Absatzmenge der ersten Woche in Szenario 3 bei den Topkunden gegenüber Szenarien 1 und 2 um knapp 30 Prozent niedriger ausfällt, liegt der Absatz der Kundengruppen „Gute Kunden“ in Szenario 6 um ca. acht Prozent höher als in den Szenarien 4 und 5.

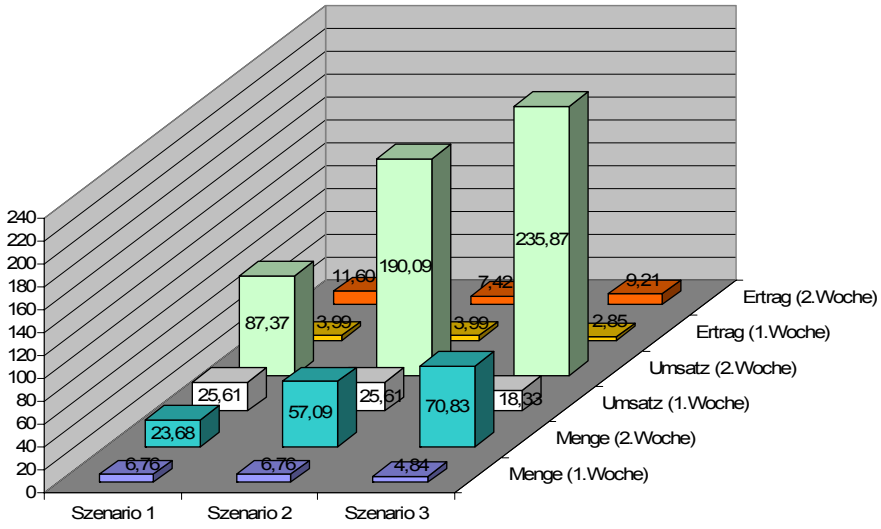


Diagramm 1: Wochenbezogene Simulationsergebnisse der Kundengruppe „Topkunden“

Der Grund dafür könnte darin liegen, dass die Topkunden im Falle von Szenario 3 durch die Ankündigung der Preisänderung in der zweiten Woche – in Form der bei Globus üblichen Faltblätter - im Vorhinein aufmerksam gemacht wurden, und deshalb im Durchschnitt mit dem Kauf bis zur Preissenkung warten. Da Topkunden aufgrund ihrer Gruppierungskriterien viel häufiger einkaufen als die Mitglieder der „Guten Kunden“, sind sie eventuell eher in der Lage, den Erwerb bis zur nächsten Woche herauszuzögern. Im Falle der „Guten Kunden“ könnte es sein, dass sie durch die Faltblätter im Durchschnitt auf Kaffee fokussiert werden, dieses Produkt aber auch schon vor der eigentlichen Preissenkung kaufen, weil ihr nächster Einkauf eventuell nicht für die Woche der Preisänderung geplant ist, und sie nicht extra wegen des Kaffees einen Ergänzungskauf machen möchten oder können.

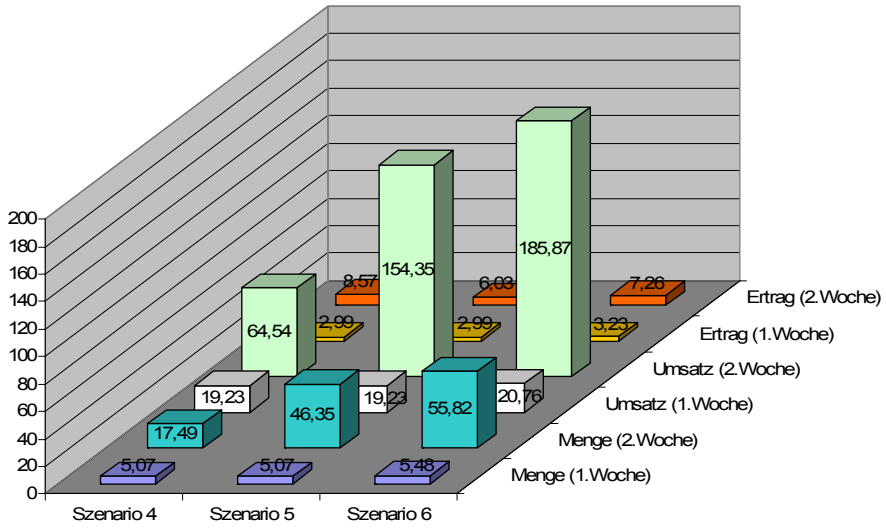


Diagramm 2: Wochenbezogene Simulationsergebnisse der Kundengruppe „Gute Kunden“

Mit Hilfe der wochenbasierten Simulationsergebnisse bezüglich der drei vorgegebenen Szenarien lassen sich weitere Unterschiede zwischen den beiden Kundengruppen feststellen. Beispielsweise reagieren beide Kundengruppen etwas unterschiedlich auf die in den Szenarien vorgegebenen Preisänderungen (siehe Diagramm 3). Durch die Preissenkung in Szenario 1 bzw 4 von 3,79 Euro in der ersten Woche auf 3,69 Euro in der zweiten Woche ergibt sich bei den Topkunden eine Absatzsteigerung um ca. 250 Prozent, während die Steigerung bei den „Guten Kunden“ mit ca. 245 Prozent etwas geringer ausfällt. Umgekehrt bewirkt die Preissenkung von 3,79 Euro auf 3,33 Euro in den Szenarien 2 und 5 bei den Topkunden eine deutliche Absatzsteigerung um ca. 745 Prozent, während die „Guten Kunden“ mit einer Steigerung um ca. 813 Prozent sogar noch etwas stärker auf die Preissenkung reagieren.

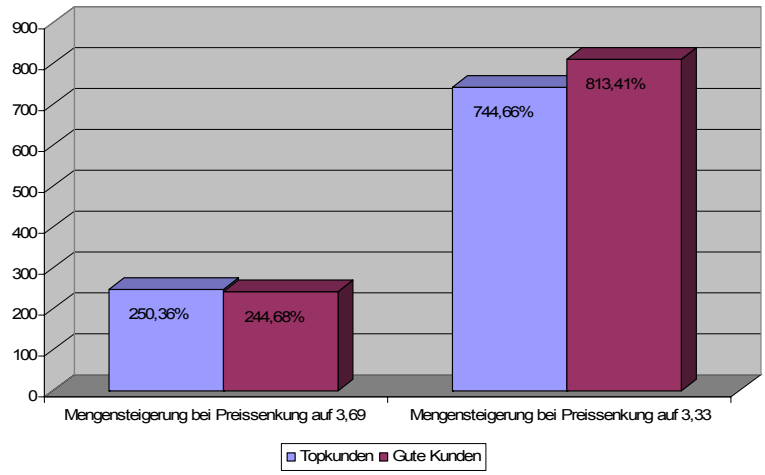


Diagramm 3: Unterschiedliche Reaktionen auf Preisänderungen

Auf analoge Weise kann der Unterschied zwischen den beiden Kundengruppen bezüglich ihrer Reaktionen auf die Bewerbung in den Szenarien 3 und 6 analysiert werden (siehe Diagramm 4). Während die Topkunden zwar etwas schwächer auf die deutliche Preissenkung reagiert haben, bewirkt bei ihnen eine zusätzliche Bewerbung eine Steigerung der Nachfrage um ca. 24 weitere Prozent, wohingegen die Bewerbung den Absatz der „Guten Kunden“ nur um ca. 20 Prozent erhöht.

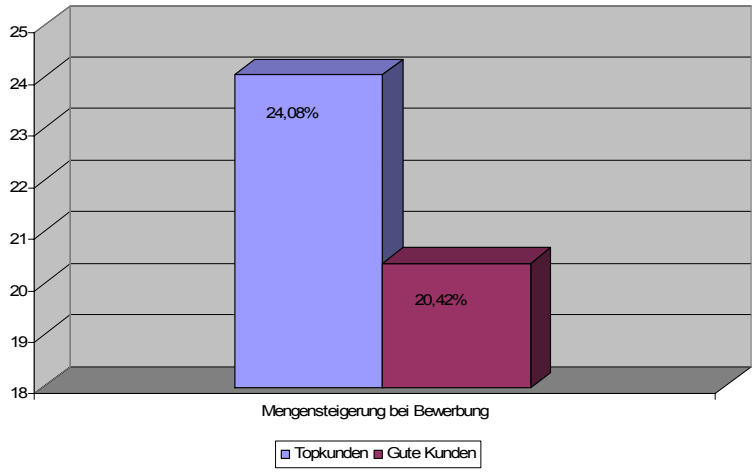


Diagramm 4: Unterschiedliche Reaktionen auf Bewerbung

6.3.2 Evaluationsbeispiel und Ergebnisse

Im Folgenden beschreibe ich zunächst anhand eines Beispiels die Vorgehensweise zur Evaluation der Prognosegüte der szenariobasierten Simulation des kundengruppenbezogenen

Kaufverhaltens. Anschließend gebe ich einen Überblick über die Evaluationskomponente des SimMarket Systems, mit dessen Hilfe automatisch eine große Anzahl geeigneter Evaluationsszenarien gefunden und evaluiert werden kann. Zum Abschluss präsentiere ich die Ergebnisse einiger durchgeführter automatischer Evaluationen. Ich beschränke mich bei sämtlichen nachfolgenden Evaluationen auf Szenarien mit harten Evidenzen und berücksichtige keine Abhängigkeiten zwischen einzelnen Artikeln sowie eventuell vorliegende saisonalunabhängige Trends.

Evaluationsszenarien bestehen aus der exakten Nachbildung vergangener Situationen, deren Auswirkungen durch Simulation prognostiziert und anschließend mit den realen Auswirkungen verglichen werden. Dabei werden die Verhaltensnetze ausschließlich aus historischen Daten gelernt, die zeitlich vor dem Evaluationszeitraum liegen. Auf diese Weise können beispielsweise kundengruppenbezogene Absatzmengen vergangener Zeiträume simuliert und anschließend mit den realen Abverkäufen verglichen werden.

Abbildung 74 zeigt die Preis- und Promotionshistorien eines Fruchtgummiproduktes im Artikelmanager des SimMarket Systems. Um geeignete Evaluationsszenarien zu erstellen, können Ausschnitte der Artikelhistorie ausgewählt werden, die anschließend zu einem Evaluationsszenario zusammengesetzt werden.

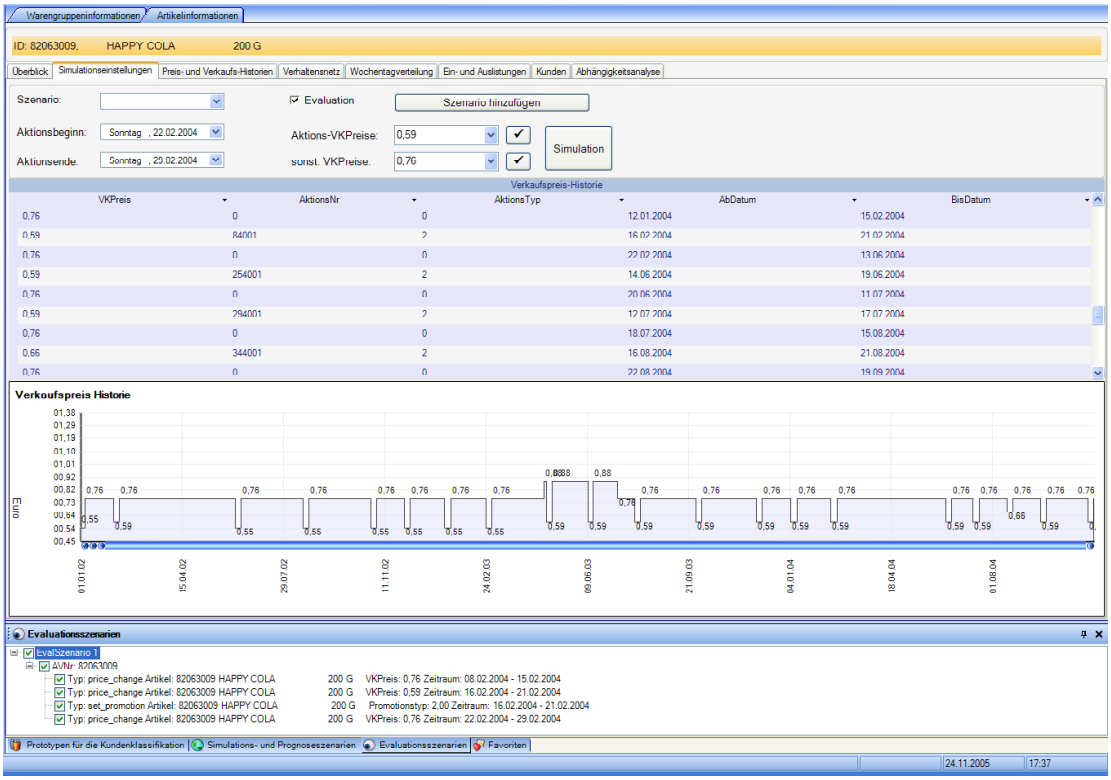


Abbildung 74: Auswahl historischer Szenarien

Hierzu wird die gleiche Maske verwendet, die auch zur Erstellung zukünftiger Szenarien benutzt wird (siehe Abbildung 71 in Kapitel 6.3.1), da es sich bei Evaluationsszenarien um spezielle Szenario-Objekte handelt, die auf historische Situationen beschränkt sind. In meinem Beispiel wurde als Evaluationszeitraum ein Zeitabschnitt von drei Wochen (08.02.2004 bis 29.02.2004) ausgewählt (siehe unterer Bildausschnitt). In der ersten Woche galt ein Verkaufspreis von 0,76 Euro, in der zweiten Woche wurde eine Preissenkung auf 0,59 Euro mit gleichzeitiger Faltblattwerbung durchgeführt und in der dritten Woche wurde der Verkaufspreis wieder auf 0,76 Euro angehoben.

Nach der Zusammenstellung des Evaluationsszenarios kann der Evaluationsprozess in der Evaluationskomponente konfiguriert und anschließend gestartet werden (siehe Abbildung 75). Die Komponente baut auf dem Simulationsmodul auf und wurde um den Vergleich der Simulationsergebnisse mit den entsprechenden historischen Daten und um die semi-automatische Generierung geeigneter Evaluationsszenarien erweitert.

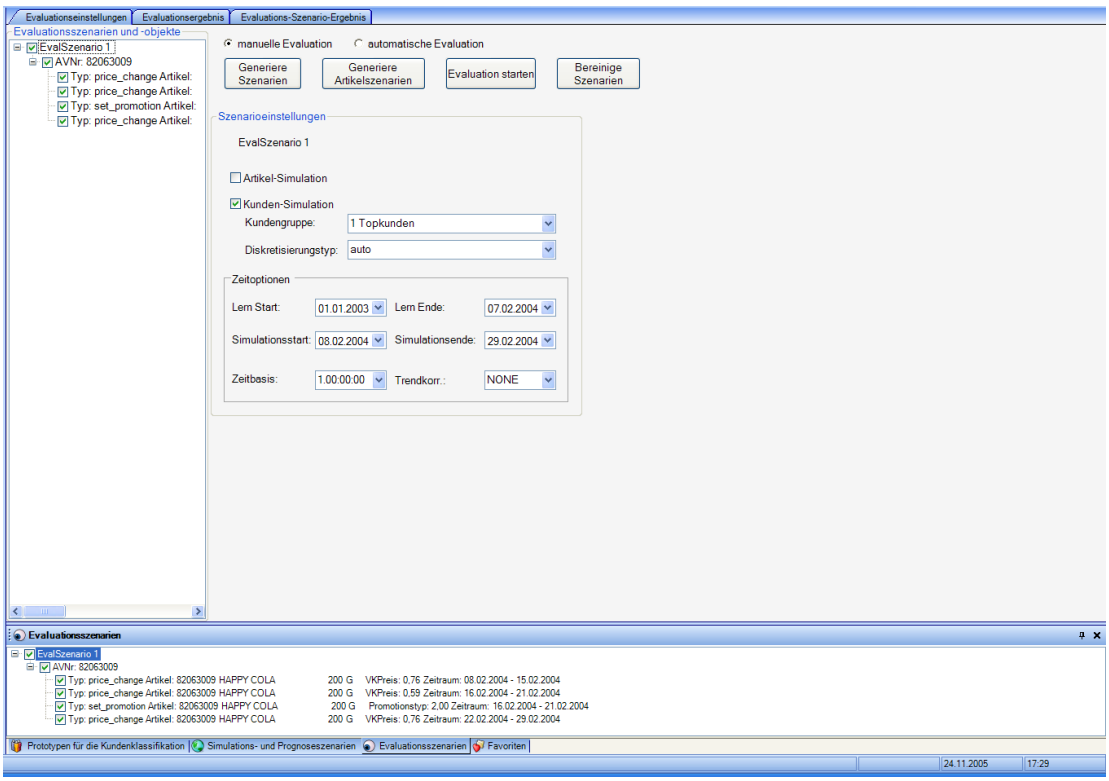


Abbildung 75: Evaluationseinstellungen

Analog zur Konfiguration der Simulationseinstellungen lassen sich mit der Eingabemaske entsprechende Evaluationseinstellungen durchführen. Beispielsweise ist auf Abbildung 75 ersichtlich, dass die Kundengruppe „Topkunden“ ausgewählt wurde, der Lernzeitraum zeitlich vor dem zu evaluierenden Simulationszeitraum liegt und die Simulation sowie deren

Evaluation tageweise durchgeführt werden soll.

Nach Start der Evaluation wird das kundengruppenbezogene Artikelverhaltensnetz bezüglich des vorgegebenen Fruchtgummiproduktes aus den entsprechenden historischen Realdaten gelernt und zur Simulation des aggregierten Kaufverhaltens der Kundengruppe unter den im Evaluationsszenario definierten Bedingungen verwendet.

Das Ergebnis der Simulation besteht aus einem prognostizierten Erwartungswert der Artikelmenge, die die Kundengruppe in der vorgegeben Situation wahrscheinlich kaufen wird. Zur Ermittlung der Prognosegüte wird dieser Wert anschließend sowohl absolut als auch relativ mit der realen Absatzmenge verglichen, die die Kundengruppe tatsächlich im angegebenen Zeitraum gekauft hat.

Abbildung 76 zeigt die konkreten Ergebnisse meines Evaluationsbeispiels. Die Kunden der Kundengruppe „Topkunden“ haben im vorgegebenen Zeitraum insgesamt 154 Einheiten des Softgummiproduktes gekauft, während mit Hilfe der szenariobasierten Simulation ihres aggregierten Kaufverhaltens 147,25 Einheiten prognostiziert wurden. Dies entspricht einer Abweichung von 6,75 Einheiten bzw. einer Prognosegüte von 95,62 Prozent.

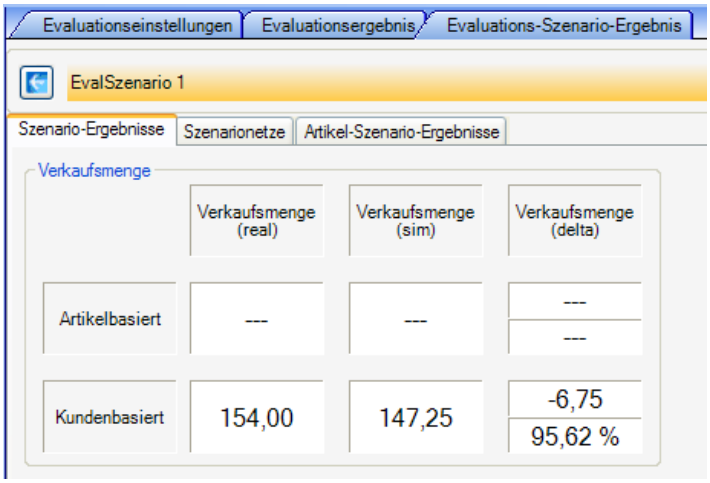


Abbildung 76: Evaluationsergebnis

Um allgemeine Aussagen über die Prognosegüte des vorgestellten szenariobasierten Verfahrens zur Simulation des Kaufverhaltens von Kundengruppen treffen zu können, sollte die in meinem Evaluationsbeispiel beschriebene Vorgehensweise auf einer möglichst großen Anzahl historischer Szenarien durchgeführt werden. Aus diesem Grund wurde in das SimMarket System eine Komponente zur semi-automatischen Evaluation integriert, mit der es möglich ist, automatisiert nach geeigneten historischen Szenarien zu suchen, die vorgegebenen Kriterien entsprechen.

Abbildung 77 zeigt die graphische Benutzeroberfläche mit den unterschiedlichen Eingabemasken zur Definition der Suchkriterien. Es lassen sich neben den

simulationsbezogenen Einstellungen - wie Zeitbasis, Lernzeitraum, Trendkorrektur, Diskretisierungstyp und zu simulierende Kundengruppe - Kriterien bezüglich der zugrunde liegenden Artikelmenge und der darauf aufbauenden Szenarien festlegen.

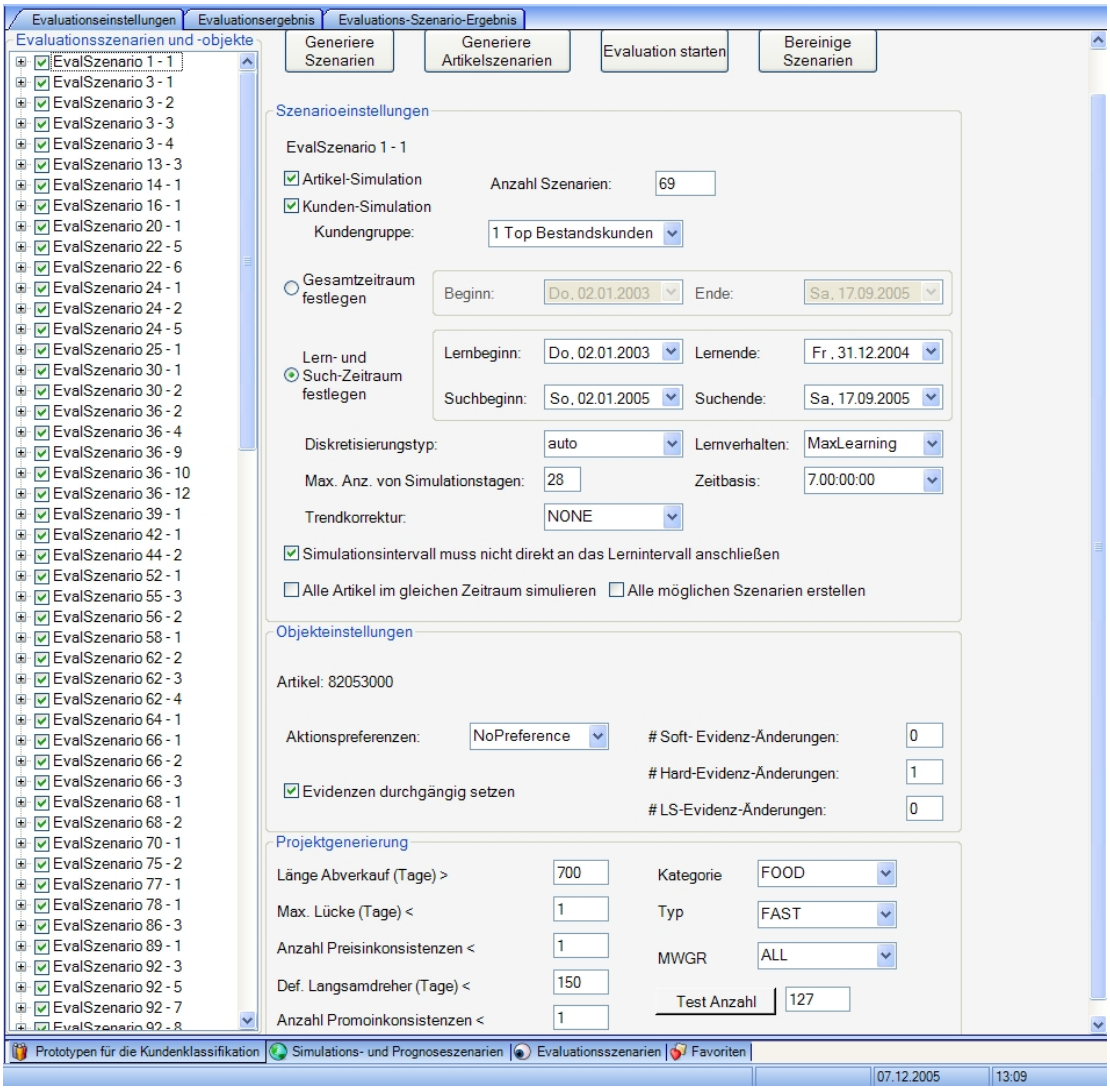


Abbildung 77: Semi-automatische Suche nach Evaluationsszenarien

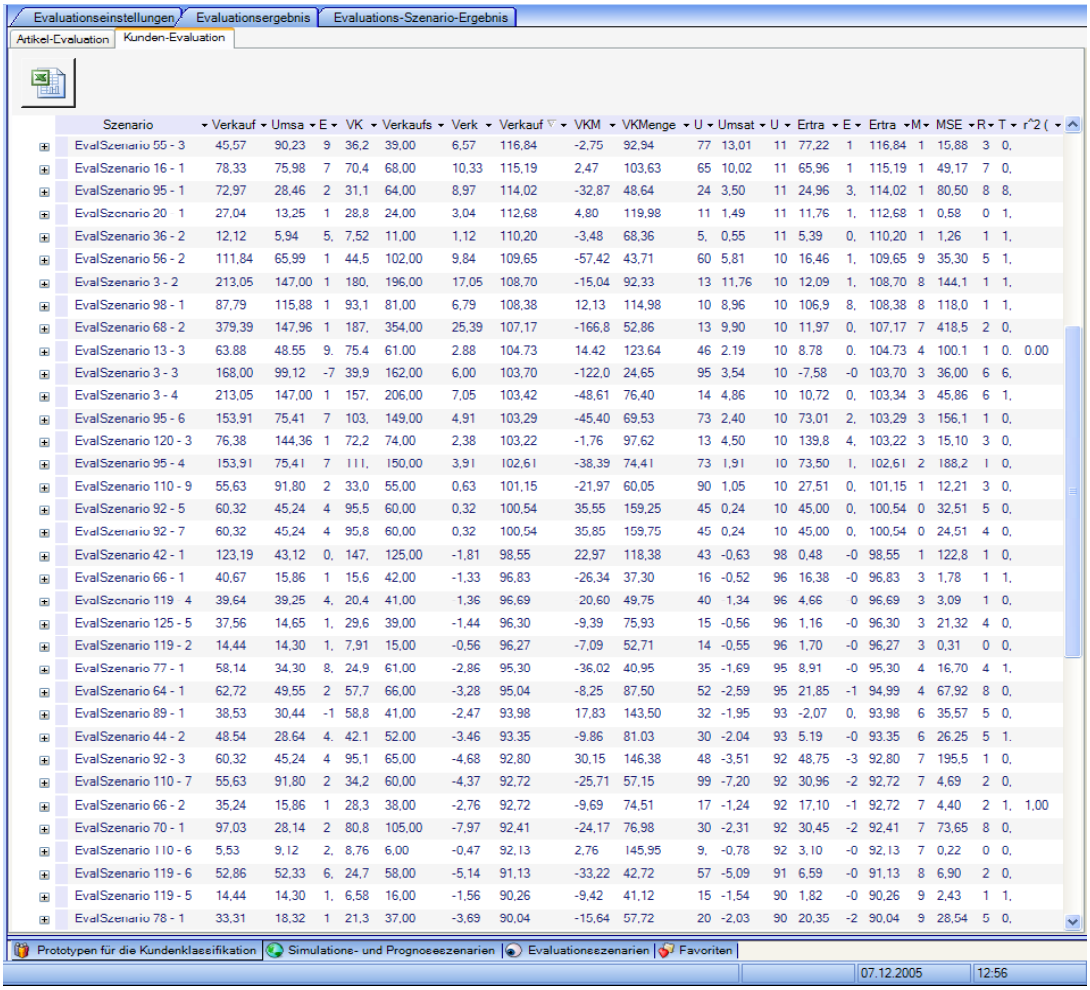
Die Einstellungen bezüglich der Artikeleigenschaften können im unteren Bildschirmbereich vorgenommen werden. Die Evaluation kann auf diese Weise zum Beispiel auf Produkte in bestimmten Unterwarengruppen im „Non-Food“- oder „Food“-Bereich eingeschränkt werden und es kann darüber hinaus festgelegt werden, ob „Schnell“- oder „Langsam“-Dreher betrachtet werden sollen (also Artikel mit sehr hohen oder sehr niedrigen Abverkäufen). Dabei können die in Frage kommenden Artikel bezüglich ihrer Einlistungs- bzw. Abverkaufszeitdauer und ihrer Abverkaufshäufigkeit eingeschränkt werden.

Da die Datengrundlage in vielen Fällen selbst nach intensiven Datenbereinigungen Inkonsistenzen aufweisen kann, die zu teilweise erheblichen Prognosefehlern führen können, kann bei der Suche nach geeigneten Evaluationsszenarien die maximal erlaubte Anzahl von Inkonsistenzen bezüglich Preisen und Bewerbungen für einzelne Artikel vorgegeben werden. Nachdem durch die Vorgabe geeigneter Filtereinstellungen eine Menge von Artikeln als Grundlage für die Evaluationsszenarien ermittelt wurde, können anschließend im oberen und mittleren Bildschirmbereich szenariobezogene Kriterien definiert werden, mit deren Hilfe sich eine automatische Suche nach historischen artikelbezogenen Szenarien durchführen lässt. Beispielsweise können der Suchzeitraum und die maximale Szenariodauer festgelegt werden. Darüber hinaus lässt sich einstellen, welche Arten von Veränderungen simuliert werden sollen, d. h. beispielsweise ob entweder reine Preisänderungen, reine Werbemaßnahmen oder deren Kombination evaluiert werden sollen. Dabei kann die Anzahl der maximal erlaubten Änderungen innerhalb eines Szenarios definiert werden. Ebenso kann festgelegt werden, ob historische Szenarien ausschließlich weiche oder harte Evidenzen beinhalten dürfen oder ob deren Kombination erlaubt ist.

Abbildung 77 zeigt ein Evaluationsbeispiel, in dem insgesamt 69 zufällige Evaluationsszenarien (siehe rechte Bildschirmseite) gefunden wurden. Dabei wurde die zugrunde liegende Artikelmenge auf absatzstarke Produkte aus dem Foodbereich beschränkt, die eine geringe Anzahl von Inkonsistenzen vorweisen. Der Suchzeitraum wurde auf einen Zeitraum von ca. 9 Monaten (Januar bis September 2005) begrenzt und die Szenarien dürfen sowohl reine Preis- und Promotionsmaßnahmen als auch deren Kombination beinhalten.

Nach Angabe der zu simulierenden Kundengruppe (hier die Kundengruppe „Top Bestandskunden“) kann die Evaluation der gefundenen Szenarien durchgeführt werden, wobei jedes Szenario einzeln simuliert und anschließend mit den entsprechenden Realdaten verglichen wird. Abbildung 78 zeigt die Detailansicht der einzelnen Simulations- und Evaluationsergebnisse. Für jedes Szenario werden neben den prognostizierten Werten für Absatz, Umsatz und Ertrag und den dazugehörigen historischen Realdaten eine Vielzahl unterschiedlicher Güte- und Fehlermaße angezeigt, unter anderem auch die absolute und relative Abweichung der Simulationsergebnisse von den Realdaten. Die Ergebnisse lassen sich speichern und als Exceltabellen exportieren.

Diagramm 5 zeigt die Verteilung der Prognosegüten der 69 evaluierten Szenarien: 49 der 69 Szenarien (ca. 71 Prozent) zeigen ein mindestens gutes Ergebnis (Prognosefehler ist kleiner als 20 Prozent), wobei davon 30 Szenarien sogar ein sehr gutes Ergebnis aufweisen (Fehler ist kleiner als 10 Prozent), was ca. 61 Prozent der guten und 43 Prozent der gesamten Szenarien entspricht. 21 Prozent der Szenarien haben eine mittlere Güte mit einem Prognosefehler von maximal 50 Prozent und nur 7 Prozent zeigen Abweichungen von über 50 Prozent.



Szenario	Verkauf	Umsa	E	VK	Verkaufs	Verk	Verkauf	VKM	VKMenge	U	Umsat	U	Ertra	E	Ertra	M	MSE	R	T	r^2 (
EvalSzenario 55 - 3	45,57	90,23	9	36,2	39,00	6,57	116,84	-2,75	92,94	77	13,01	11	77,22	1	116,84	1	15,88	3	0,	
EvalSzenario 16 - 1	78,33	75,98	7	70,4	68,00	10,33	115,19	2,47	103,63	65	10,02	11	65,96	1	115,19	1	49,17	7	0,	
EvalSzenario 95 - 1	72,97	28,46	2	31,1	64,00	8,97	114,02	-32,87	48,64	24	3,50	11	24,96	3,	114,02	1	80,50	8	8,	
EvalSzenario 20 1	27,04	13,25	1	28,8	24,00	3,04	112,68	4,80	119,98	11	1,49	11	11,76	1,	112,68	1	0,58	0	1,	
EvalSzenario 36 - 2	12,12	5,94	5,	7,52	11,00	1,12	110,20	-3,48	68,36	5,	0,55	11	5,39	0,	110,20	1	1,26	1	1,	
EvalSzenario 56 - 2	111,84	65,99	1	44,5	102,00	9,84	109,65	-57,42	43,71	60	5,81	10	16,46	1,	109,65	9	35,30	5	1,	
EvalSzenario 3 - 2	213,05	147,00	1	180,	196,00	17,05	108,70	-15,04	92,33	13	11,76	10	12,09	1,	108,70	8	144,1	1	1,	
EvalSzenario 98 - 1	87,79	115,88	1	93,1	81,00	6,79	108,38	12,13	114,98	10	8,96	10	106,9	8,	108,38	8	118,0	1	1,	
EvalSzenario 68 - 2	379,39	147,96	1	187,	354,00	25,39	107,17	-166,8	52,86	13	9,90	10	11,97	0,	107,17	7	418,5	2	0,	
EvalSzenario 13 - 3	63.88	48.55	9.	75.4	61.00	2.88	104.73	14.42	123.64	46	2.19	10	8.78	0.	104.73	4	100.1	1	0.	0.00
EvalSzenario 3 - 3	168,00	99,12	-7	39,9	162,00	6,00	103,70	-122,0	24,65	95	3,54	10	-7,58	-0	103,70	3	36,00	6	6,	
EvalSzenario 3 - 4	213,05	147,00	1	157,	206,00	7,05	103,42	-48,61	76,40	14	4,86	10	10,72	0,	103,34	3	45,86	6	1,	
EvalSzenario 95 - 6	153,91	75,41	7	103,	149,00	4,91	103,29	-45,40	69,53	73	2,40	10	73,01	2,	103,29	3	156,1	1	0,	
EvalSzenario 120 - 3	76,38	144,36	1	72,2	74,00	2,38	103,22	-1,76	97,62	13	4,50	10	139,8	4,	103,22	3	15,10	3	0,	
EvalSzenario 95 - 4	153,91	75,41	7	111,	150,00	3,91	102,61	-38,39	74,41	73	1,91	10	73,50	1,	102,61	2	188,2	1	0,	
EvalSzenario 110 - 9	55,63	91,80	2	33,0	55,00	0,63	101,15	-21,97	60,05	90	1,05	10	27,51	0,	101,15	1	12,21	3	0,	
EvalSzenario 92 - 5	60,32	45,24	4	95,5	60,00	0,32	100,54	35,55	159,25	45	0,24	10	45,00	0,	100,54	0	32,51	5	0,	
EvalSzenario 92 - 7	60,32	45,24	4	95,8	60,00	0,32	100,54	35,85	159,75	45	0,24	10	45,00	0,	100,54	0	24,51	4	0,	
EvalSzenario 42 - 1	123,19	43,12	0,	147,	125,00	-1,81	98,55	22,97	118,38	43	-0,63	98	0,48	-0	98,55	1	122,8	1	0,	
EvalSzenario 66 - 1	40,67	15,86	1	15,6	42,00	-1,33	96,83	-26,34	37,30	16	-0,52	96	16,38	-0	96,83	3	1,78	1	1,	
EvalSzenario 119 4	39,64	39,25	4,	20,4	41,00	1,36	96,69	20,60	49,75	40	1,34	96	4,66	0	96,69	3	3,09	1	0,	
EvalSzenario 125 - 5	37,56	14,65	1,	29,6	39,00	-1,44	96,30	-9,39	75,93	15	-0,56	96	1,16	-0	96,30	3	21,32	4	0,	
EvalSzenario 119 - 2	14,44	14,30	1,	7,91	15,00	-0,56	96,27	-7,09	52,71	14	-0,55	96	1,70	-0	96,27	3	0,31	0	0,	
EvalSzenario 77 - 1	58,14	34,30	8,	24,9	61,00	-2,86	95,30	-36,02	40,95	35	-1,69	95	8,91	-0	95,30	4	16,70	4	1,	
EvalSzenario 64 - 1	62,72	49,55	2	57,7	66,00	-3,28	95,04	-8,25	87,50	52	-2,59	95	21,85	-1	94,99	4	67,92	8	0,	
EvalSzenario 89 - 1	38,53	30,44	-1	58,8	41,00	-2,47	93,98	17,83	143,50	32	-1,95	93	-2,07	0,	93,98	6	35,57	5	0,	
EvalSzenario 44 - 2	48.54	28.64	4.	42.1	52.00	-3.46	93.35	-9.86	81.03	30	-2.04	93	5.19	-0	93.35	6	26.25	5	1.	
EvalSzenario 92 - 3	60,32	45,24	4	95,1	65,00	-4,68	92,80	30,15	146,38	48	-3,51	92	48,75	-3	92,80	7	195,5	1	0,	
EvalSzenario 110 - 7	55,63	91,80	2	34,2	60,00	-4,37	92,72	-25,71	57,15	99	-7,20	92	30,96	-2	92,72	7	4,69	2	0,	
EvalSzenario 66 - 2	35,24	15,86	1	28,3	38,00	-2,76	92,72	-9,69	74,51	17	-1,24	92	17,10	-1	92,72	7	4,40	2	1,	1,00
EvalSzenario 70 - 1	97,03	28,14	2	80,8	105,00	-7,97	92,41	-24,17	76,98	30	-2,31	92	30,45	-2	92,41	7	73,65	8	0,	
EvalSzenario 110 - 6	5,53	9,12	2,	8,76	6,00	-0,47	92,13	2,76	145,95	9,	-0,78	92	3,10	-0	92,13	7	0,22	0	0,	
EvalSzenario 119 - 6	52,86	52,33	6,	24,7	58,00	-5,14	91,13	-33,22	42,72	57	-5,09	91	6,59	-0	91,13	8	6,90	2	0,	
EvalSzenario 119 - 5	14,44	14,30	1,	6,58	16,00	-1,56	90,26	-9,42	41,12	15	-1,54	90	1,82	-0	90,26	9	2,43	1	1,	
EvalSzenario 78 - 1	33,31	18,32	1	21,3	37,00	-3,69	90,04	-15,64	57,72	20	-2,03	90	20,35	-2	90,04	9	28,54	5	0,	

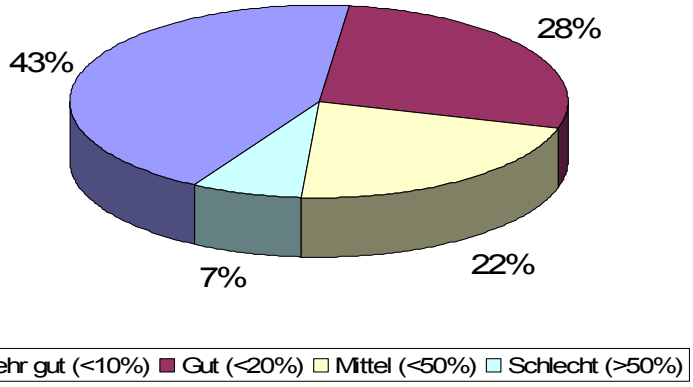


Diagramm 5: Verteilung der Prognosegüten

6.4 Beispiele und Evaluation der Kundengruppenbestimmung

Im Folgenden zeige ich anhand einiger Anwendungsbeispiele die Möglichkeiten zur Bestimmung von Kundengruppen innerhalb des SimMarket Systems. Die Basis bilden dabei die in Kapitel 5 vorgestellten Vergleichsverfahren für probabilistische Verhaltensnetze, mit deren Hilfe das Kaufverhalten einzelner Kunden verglichen werden kann.

In Unterkapitel 6.4.1 stelle ich an zwei praktischen Beispielen die Vorteile der verhaltensnetzbasierten gegenüber der klassischen, vektorbasierten Ähnlichkeitsanalyse dar. Anschließend präsentiere ich die Ergebnisse zweier Evaluationen, die auf Basis der Daten von 200 realen Kunden durchgeführt wurden.

Basierend auf der paarweisen Ähnlichkeitsanalyse des Kundenverhaltens können mit Hilfe des SimMarket Systems Kundenklassifikationen durchgeführt werden. Dabei werden vorgegebene Kundenklassen durch Prototypen und deren Verhaltensnetze repräsentiert. In Unterkapitel 6.4.2 zeige ich einen entsprechenden Anwendungsfall, bei dem Kunden bezüglich ihrer individuellen Preis- und Werbesensibilität klassifiziert werden, um die Auswirkungen von Preis- und Promotionsmaßnahmen im Vorfeld besser einschätzen zu können, da sich auf diese Weise die Menge der durch die Maßnahmen positiv bzw. negativ betroffenen Kunden abschätzen lässt.

6.4.1 Ähnlichkeitsanalysen mit Vektoren und Verhaltensnetzen

Klassische Kundenähnlichkeitsanalysen basieren auf dem Vergleich soziodemographischer und einfacher, verhaltensbezogener Attributbeschreibungen, die in Form von Vektoren mit Hilfe vektorbasierter Ähnlichkeits- bzw. Abstandsmaße verglichen werden (siehe Kapitel 5.3.1). Die resultierenden Ergebnisse können allerdings bei einer Vielzahl von verhaltensbezogenen Fragestellungen nur unzureichende Unterstützung bieten, da Kunden anhand oberflächlicher Merkmale verglichen werden, mit deren Hilfe nicht auf konkretes, dynamisches Kaufverhalten geschlossen werden kann.

Die im Rahmen meiner Arbeit vorgestellten simulationsbasierten Verfahren (Kapitel 5.4) und direkten Ähnlichkeitsmaße (Kapitel 5.5) für probabilistische Verhaltensnetze stellen den Vergleich dynamischen Kaufverhaltens in den Vordergrund und können bei entsprechenden Fragestellungen detaillierte Informationen über Verhaltensunterschiede von einzelnen Kunden und Kundengruppen liefern. Im Folgenden zeige ich an zwei praktischen Beispielen den Unterschied zwischen den Ergebnissen vektorbasierter und verhaltensnetzbasierter Ähnlichkeitsanalysen.

Abbildung 79 zeigt die dreigeteilte Ansicht zur Analyse der Ähnlichkeit zweier Kunden. Auf der linken Seite befinden sich im oberen Bereich Bildschirmmasken zur Auswahl der beiden Kunden und im unteren Bereich mehrere Kennzahlen vektorbasierter und verhaltensnetzbasierter Ähnlichkeitsmaße. Der restliche Bildschirmbereich dient zur

Visualisierung der verhaltensbezogenen Informationen über die beiden Kunden. Im oberen Bereich werden der globale Attributvektor in Tabellenform sowie das entsprechende globale Verhaltensnetz in Diagrammansicht des ersten Kunden angezeigt, während im unteren Bereich die entsprechenden Informationen des zweiten Kunden dargestellt sind. Ziel ist es, den Unterschied zwischen den Ergebnissen vektorbasierter und verhaltensnetzbasierter Ähnlichkeitsanalysen zu verdeutlichen.

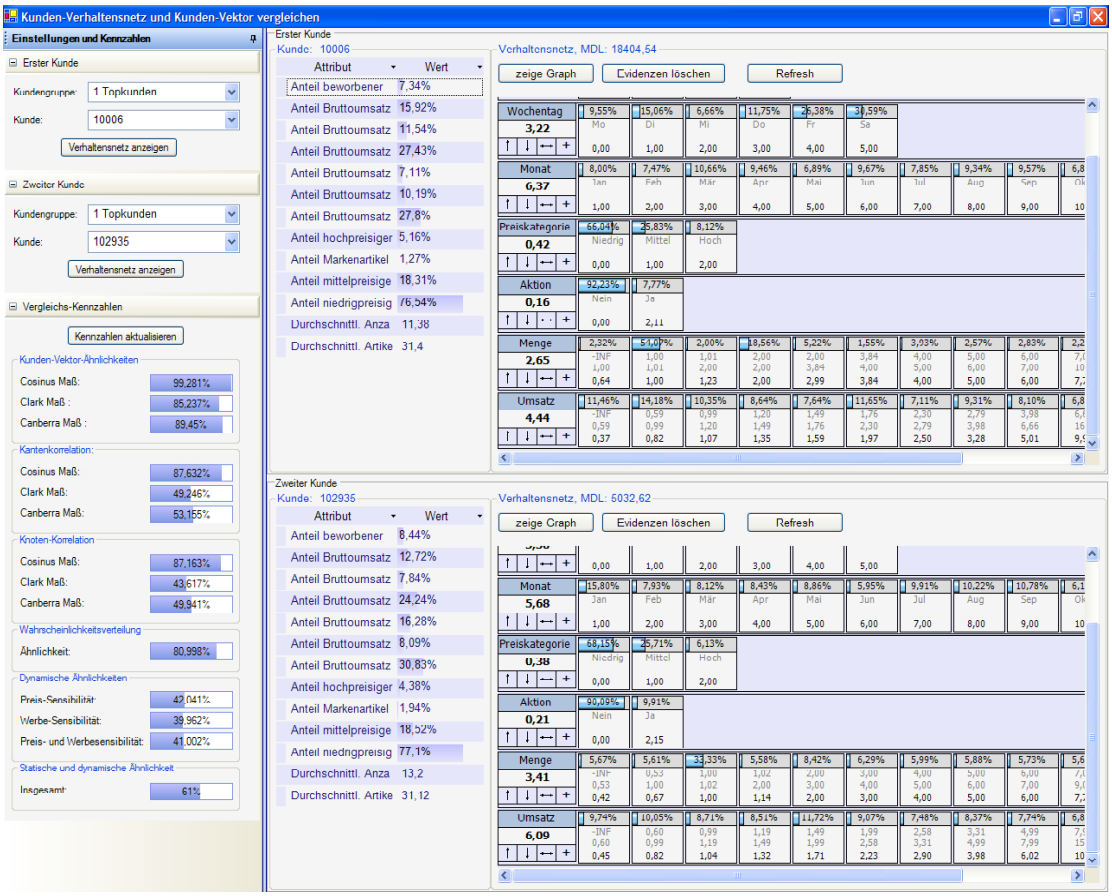


Abbildung 79: Ähnlichkeitsanalyse: Vektoren ähnlich – Verhaltensnetze unterschiedlich

Im ersten Beispiel wurden zwei Kunden der Kundengruppe „Topkunden“ ausgewählt. Um die inhaltlichen Informationen der vektorbasierten und verhaltensnetzbasierten Modelle so vergleichbar wie möglich zu gestalten, wurden sämtliche im Verhaltensnetz modellierten Zufallsvariablen und semantischen Inhalte durch geeignete Attribute im Vektor repräsentiert, wie beispielsweise Umsatz-Wochentagsverteilungen oder Verteilungen der gekauften Artikel in Preiskategorien bzw. in beworbene und nicht beworbene Artikel.

Bei der Betrachtung der beiden Attributvektoren fällt auf, dass sich sowohl die absoluten Zahlenwerte als auch deren Relationen untereinander sehr ähnlich sind. Entsprechend deuten

die Ergebnisse der klassischen vektorbasierten Ähnlichkeitsmaße auf eine sehr hohe Ähnlichkeit zwischen den beiden Kunden hin (ersichtlich durch die obersten drei Kennzahlen auf der linken Seite): Das Kosinus-Maß gibt eine Ähnlichkeit von 99,281 Prozent an, das Clark-Maß beträgt 85,237 Prozent und mit dem von mir favorisierten Canberra-Maß wurden 89,45 Prozent ermittelt. Bei einem in der Praxis üblichen Schwellenwert von 80 Prozent würden die beiden Kunden aufgrund dieser Analyseergebnisse derselben Kundengruppe zugeordnet, was in der Realität auch geschehen ist, da sich beide Kunden in der Kundengruppe „Topkunden“ befinden.

Die Ergebnisse der verhaltensnetzbasierten Ähnlichkeitsanalysen deuten allerdings auf ein sehr unterschiedliches dynamisches Kaufverhalten der beiden Kunden hin. Beispielsweise besitzen die Kanten- bzw. Knotenrelationsmaße Werte zwischen 43,617 und 53,155 Prozent, wenn das in diesem Falle ungeeignete relative Kosinus-Maß vernachlässigt wird. Lediglich das Randwahrscheinlichkeitsverteilungsmaß beträgt 80,998 Prozent, da hier ähnlich des vektorbasierten Vergleichs keine Abhängigkeiten berücksichtigt werden. Nach den Ergebnissen der simulationsbasierten Ähnlichkeitsanalysen bezüglich der Preissensibilität und der Werbesensibilität unterscheiden sich die beiden Kunden ebenfalls erheblich (42,041 bzw. 39,962 Prozent). Dazu wurden die Reaktionen der beiden Kunden bezüglich einer Vielzahl von Szenarien mit unterschiedlichen absoluten und relativen Preisänderungen sowie veränderten Preis/Werbemaßnahmen-Kombinationen mit Hilfe der kundenbezogenen Verhaltensnetze simuliert und anschließend miteinander verglichen.

Dieser Unterschied im dynamischen Kaufverhalten wird nicht durch die entsprechenden Merkmale der Attributvektoren sichtbar: dort sind sich sowohl der Anteil der Preiskategorien als auch der Anteil der beworbenen Artikel bei beiden Kunden - genauso wie alle übrigen Werte - sehr ähnlich. Das bedeutet zusammengefasst, dass klassische Verfahren beide Kunden derselben Kundengruppe zuordnen, obwohl sich die Kunden im Kaufverhalten erheblich unterscheiden. Dies kann zu enormen Fehlern bei der Bestimmung von Zielgruppen bzw. betroffenen Kundensegmenten im Rahmen verhaltensbezogener Marketingstrategien bzw. konkreter Maßnahmen führen. Die in dieser Arbeit entwickelten Verfahren hingegen zeigen die Unterschiede im dynamischen Kaufverhalten auf und eignen sich somit besser für entsprechende Zielgruppenbestimmungen.

Abbildung 80 zeigt den umgekehrten Fall, in dem zwei Kunden aufgrund der klassischen vektorbasierten Ähnlichkeitsanalyse als unterschiedlich betrachtet werden, während die simulationsbasierten Vergleiche das dynamische Kaufverhalten der beiden Kunden als sehr ähnlich deklarieren. Konkret betragen die Werte der vektorbasierten Ähnlichkeitsmaße Clark und Canberra 59,611 und 65,816 Prozent, während die mit Hilfe der Verhaltensnetze durchgeführten simulationsbasierten Vergleichsverfahren für die Preis- bzw. Werbesensibilität Ähnlichkeiten von 85,342 bzw. 96,477 Prozent ermitteln. Dies ist

wiederum nicht anhand der Merkmale in den beiden Attributvektoren ersichtlich, da die entsprechenden Werte relativ unterschiedlich sind.

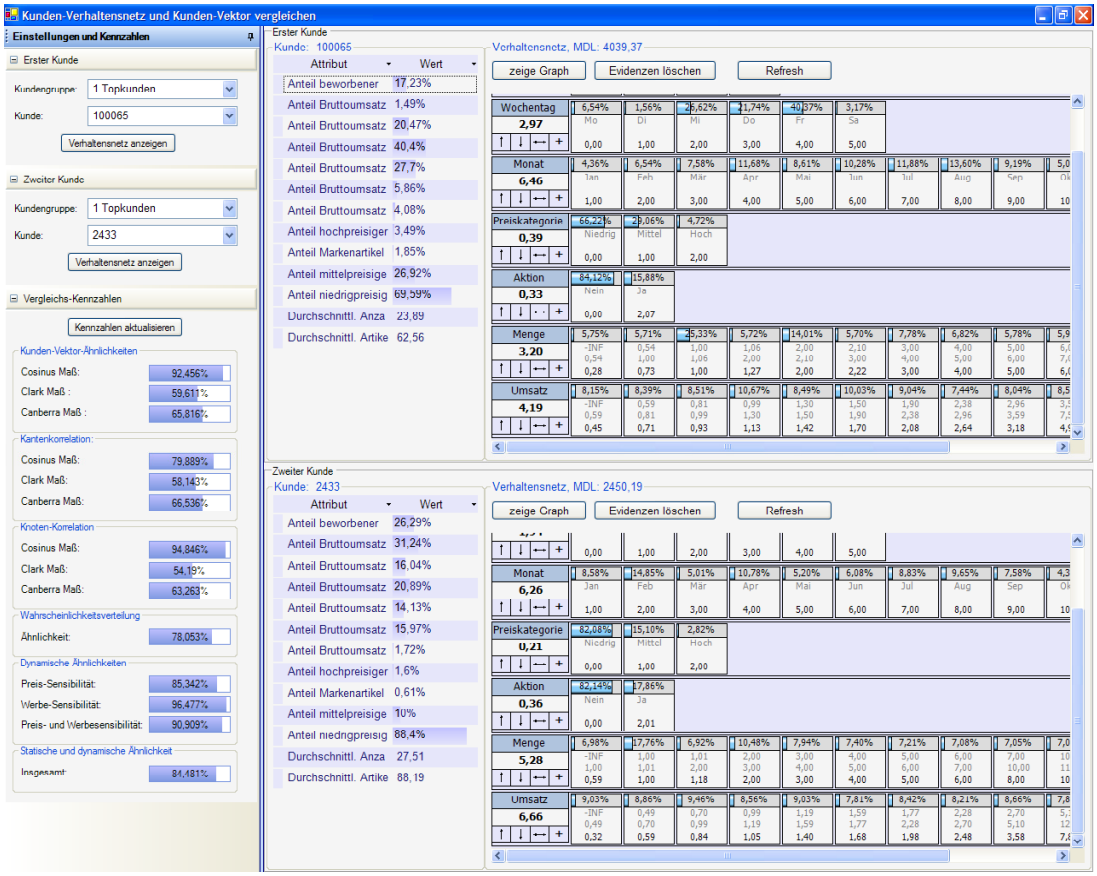


Abbildung 80: Ähnlichkeitsanalyse: Vektoren unterschiedlich – Verhalten ähnlich

Im Folgenden präsentiere ich zwei Evaluationsergebnisse zur Verdeutlichung der Unterschiede zwischen klassischen vektorbasierten und den in dieser Arbeit entwickelten verhaltensnetzbasierten Ähnlichkeitsanalyseverfahren. In beiden Fällen wurde ein konkreter Kunde der Kundengruppe „Topkunden200“ mit den 199 übrigen Gruppenmitgliedern verglichen. Auf der Seite der vektorbasierten Maße wurden das Kosinus-Maß (VECO), das Clark-Maß (VECL) und das Canberra-Maß (VECA) verwendet. Als verhaltensnetzbasierte Ähnlichkeitsmaße habe ich ein Knotenpaarkorrelationsmaß (BNMINVCA), ein Kantenkorrelationsmaß (BNMIEVCA), ein Randwahrscheinlichkeitsmaß (BNVECAMP) und ein simulationsbasiertes Maß bezüglich der Preis- und Werbesensibilität (BNDPA) eingesetzt. Sowohl BNMINVCA als auch BNMIEVCA verwenden als Metrik die Mutual-Information und vergleichen die resultierenden Vektoren mit dem Canberra-Maß, das auch beim Vergleich der Randwahrscheinlichkeiten (BNVECAMP) zum Einsatz kommt.

Diagramm 6 zeigt ein Teilergebnis der ersten Evaluation. Auf der Y-Achse ist eine Ähnlichkeitsskala von 0 bis 100 Prozent vorgegeben, während die 199 Kunden, mit denen der ausgewählte Kunde verglichen wurde, auf der X-Achse abgebildet sind. Die farblich unterschiedlichen Graphen geben die mit Hilfe der jeweiligen Vergleichsmaße ermittelten Ähnlichkeitswerte der einzelnen Kunden an, wobei die farbliche Zuordnung in der Legende des Diagramms ersichtlich ist. Die Reihenfolge der 199 Kunden wurde durch die absteigende Sortierung der Werte des vektorbasierten VECA-Maßes festgelegt, die als Referenzlinie dient.

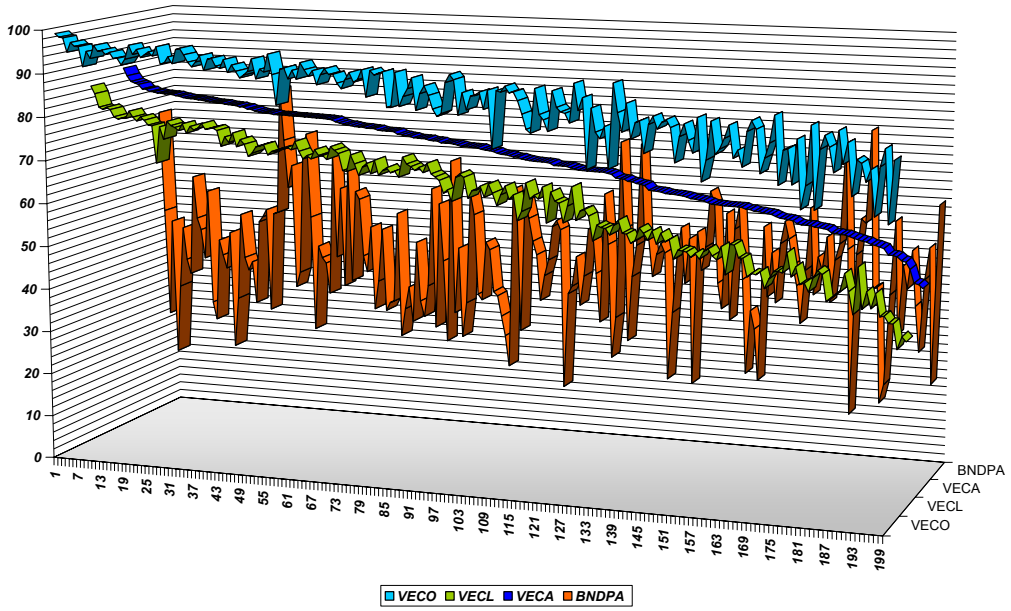


Diagramm 6: Klassische Vektormaße gegen ein simulationsbasiertes Maß (Beispiel 1)

Die entscheidende Aussage des Diagramms ist, dass die Linie des BNDPA-Maßes große Ausschläge an Stellen zeigt, wo die drei vektorbasierten Maße relativ konstante Werte besitzen. Das bedeutet, dass das Verhalten bezüglich der Preis- und Werbesensibilität als sehr unterschiedlich bewertet wird, während klassische Maße kaum einen Unterschied feststellen können. Wichtig sind dabei auch die in vielen Fällen großen Differenzen zwischen den Werten des BNDPA und des von mir bei den vektorbasierten Maßen favorisierten VECA. Darin spiegeln sich die in den oben beschriebenen Beispielen aufgetretenen Situationen wieder, dass das VECA-Maß und das auf Basis verhaltensnetzbasierter Simulationen ermittelte BNDPA-Maß häufig zu sehr unterschiedlichen Ergebnissen führen, wobei in diesem Fall die Werte des BNDPA vor allem deutlich geringer ausfallen als die des VECA (dies entspricht der Situation des oben beschriebenen ersten Beispiels in Abbildung 79). Auffällig ist darüber hinaus, dass die Kunden selbst mit den klassischen vektorbasierten

Maßen nicht vollständig über 75 bzw. 80 Prozent liegen (selbst bei dem extrem unkritischen und in diesem Rahmen ungeeigneten Kosinus-Maß). Das liegt daran, dass die Kundengruppe „Topkunden200“ ursprünglich aufgrund absatz- und einkaufhäufigkeitsorientierter Merkmale gebildet wurde, wobei Wochentagsverteilungen sowie Anteile der beworbenen Artikel bzw. der unterschiedlichen Preiskategorien keine Rolle gespielt haben.

Diagramm 7 zeigt den Vergleich zwischen den direkten Verhaltensnetzvergleichsmaßen mit dem klassischen Canberra-Maß, dessen Graph wiederum als Referenz dient. Bei den verhaltensnetzbasierten Vergleichsmaßen BNMINVCA und BNMIEVCA liegen gegenüber dem klassischen Vektormaß wiederum große Ausschläge vor, während das Randwahrscheinlichkeitsmaß geringere Ausschläge aufweist und dem Graph des VECA-Maßes am ähnlichsten ist, was aufgrund der gemeinsamen Vernachlässigung der Abhängigkeiten zwischen den Zufallsvariablen auch logisch erscheint.

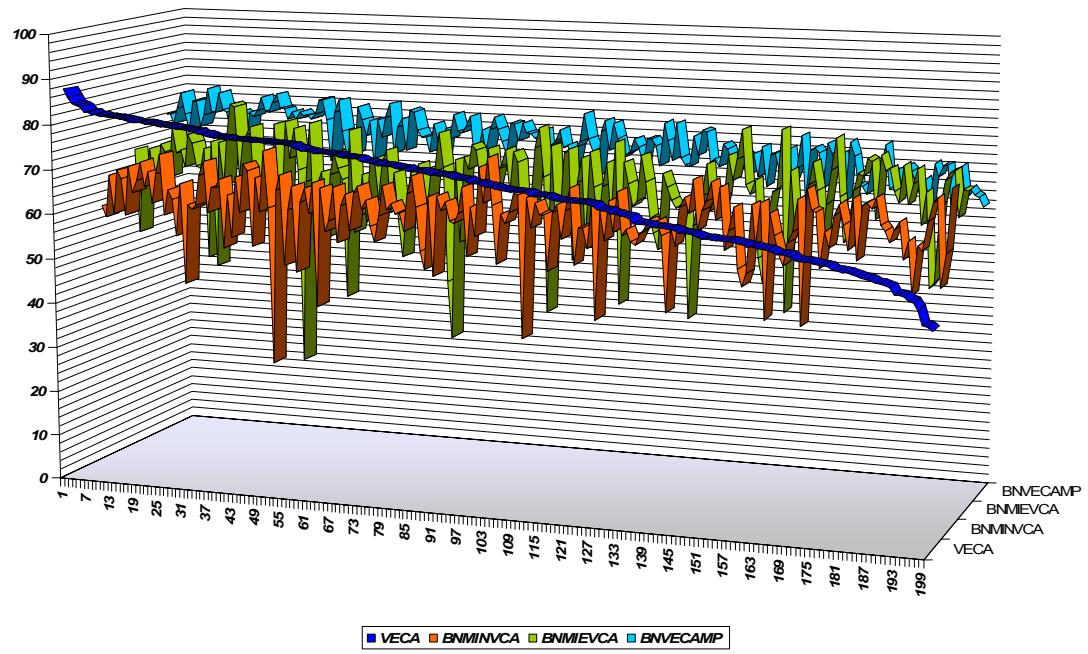


Diagramm 7: Klassisches Vektormaß gegen verhaltensnetzbasierte Maße (Beispiel 1)

Diagramm 8 zeigt das Ergebnis der zweiten Evaluation, bei der ich lediglich den Kunden ausgetauscht habe, mit dem alle anderen 199 Gruppenmitglieder verglichen werden. Die Ähnlichkeitswerte des VECA- und des BNDPA-Maßes weisen wiederum große Unterschiede auf. Gegenüber der Situation im ersten Evaluationsdiagramm befinden sich die Werte des BNDPA-Maßes etwas öfter oberhalb der Werte des VECA- und vor allem des VECL-Maßes, was der Situation in Abbildung 80 entspricht.

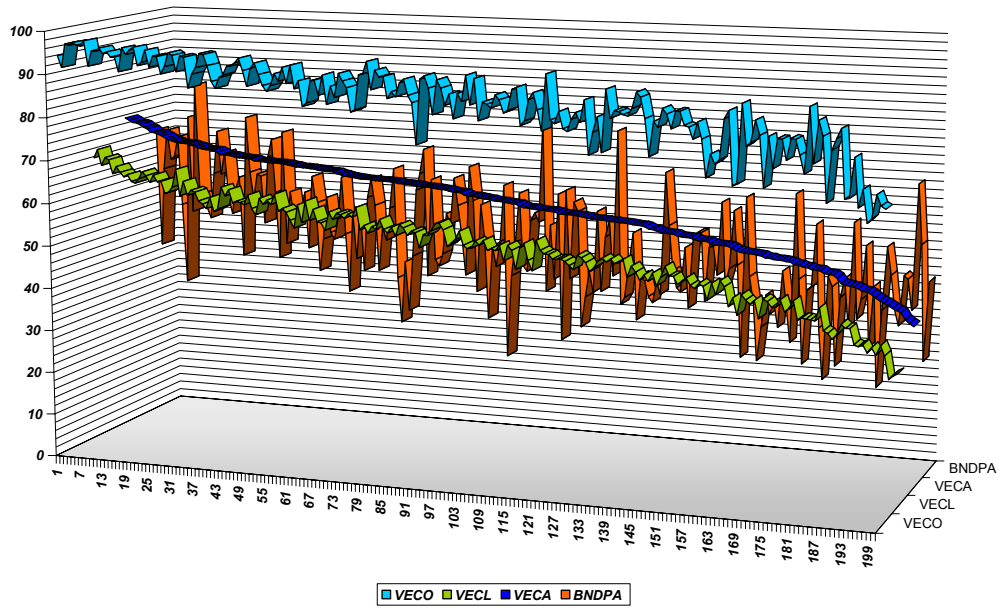


Diagramm 8: Klassische Vektormaße gegen ein simulationsbasiertes Maß (Beispiel 2)

Diagramm 9 zeigt die Gegenüberstellung der direkten Verhaltensnetzvergleichsmaße mit dem klassischen Canberra-Maß. Wiederum ist das BNVECAMP-Maß dem VECA-Maß am ähnlichsten, während die beiden Relationsvergleichsmaße deutliche Ausschläge aufweisen.

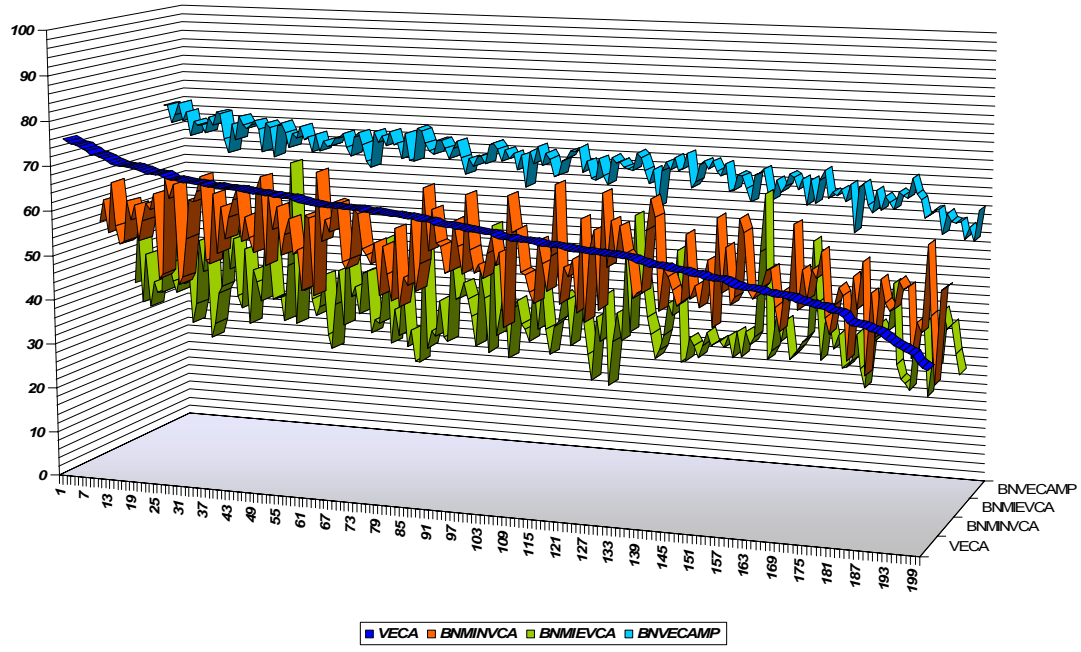


Diagramm 9: Klassisches Vektormaß gegen verhaltensnetzbasierte Maße (Beispiel 2)

Insgesamt bestätigen die beiden Evaluationsergebnisse, dass mit Hilfe der in dieser Arbeit entwickelten verhaltensnetzbasierten Vergleichsmaße detailliertere und differenziertere verhaltensbezogene Ähnlichkeitsanalysen durchgeführt werden können als mit klassischen vektorbasierten Maßen.

In den Beispielen und Evaluationen habe ich Verhaltensnetze und entsprechende Attributvektoren verwendet, die sich auf das „allgemeine“ Kaufverhalten der Kunden beziehen. Um das Verhalten je nach Fragestellung auf bestimmte Kundenmerkmale, einzelne Artikel oder Artikelgruppen zu beschränken, können statt der globalen Verhaltensnetze kundenindividuelle Feature-Teilnetze, Artikelverhaltensnetze oder holonische Artikelgruppenverhaltensnetze auf analoge Weise verwendet werden. Durch die Verwendung der Artikelgruppenverhaltensnetze ist es darüber hinaus möglich, das dynamische Cross-Selling-Verhalten einzelner Kunden zu vergleichen.

6.4.2 Instanzbasierte Kundenklassifikation mit Prototypen

Mit Hilfe der Kundenklassifikationskomponente des SimMarket Systems können sowohl vektorbasierte als auch verhaltensnetzbasierte Kundenklassifikationen durchgeführt werden. Die implementierten Klassifikationsverfahren sind instanzbasiert, d. h. dass jede vorgegebene Kundenklasse durch einen entsprechenden Prototypen repräsentiert wird. Prototypen sind entweder reale Kunden, Kundengruppendurchschnittsmodelle (Medoid oder Mean) oder virtuelle Kundenbeschreibungen. Zur Klassifikation werden einzelne Kunden mit sämtlichen Prototypen verglichen und anschließend jeder Kundenklasse zugeordnet, zu deren Prototyp ihre Ähnlichkeit größer als ein vorgegebener Schwellenwert ist. Auf diese Weise kann ein Kunde Mitglied mehrerer Klassen sein kann.

Abbildung 81 zeigt die Visualisierung der Komponente „Kundenklassifikation“ des SimMarket Systems. Auf der linken Seite lassen sich neben der zu klassifizierenden Kundenmenge verschiedene vektorbasierte und verhaltensnetzbasierte Ähnlichkeitsmaße auswählen. Zur Konfiguration der Attributvektoren können einzelne Merkmale aus einer Liste selektiert werden.

Im Beispiel sollen alle Kunden der Kundengruppe „Topkunden200“ gefunden werden, die allgemein sehr preis- und werbesensibel sind. Aus diesem Grund werden alle Mitglieder der entsprechenden Kundengruppe mit einem realen Kunden verglichen, der als Prototyp preis- und promotionssensibler Kunden dient. Als Ähnlichkeitsmaße werden die drei vektorbasierten Vergleichsmaße Kosinus, Clark und Canberra (siehe Kapitel 5.3.1) sowie ein verhaltensnetzbasiertes Maß (BNDPA) verwendet, das auf den Vergleich der Preis- und Werbesensibilität fokussiert (siehe Kapitel 5.4). Die Attribute der Kundenattributvektoren und die Zufallsvariablen der Verhaltensnetze wurden dabei auf die Fragestellung angepasst, indem nur preis- und promotionsrelevante Merkmale berücksichtigt werden.

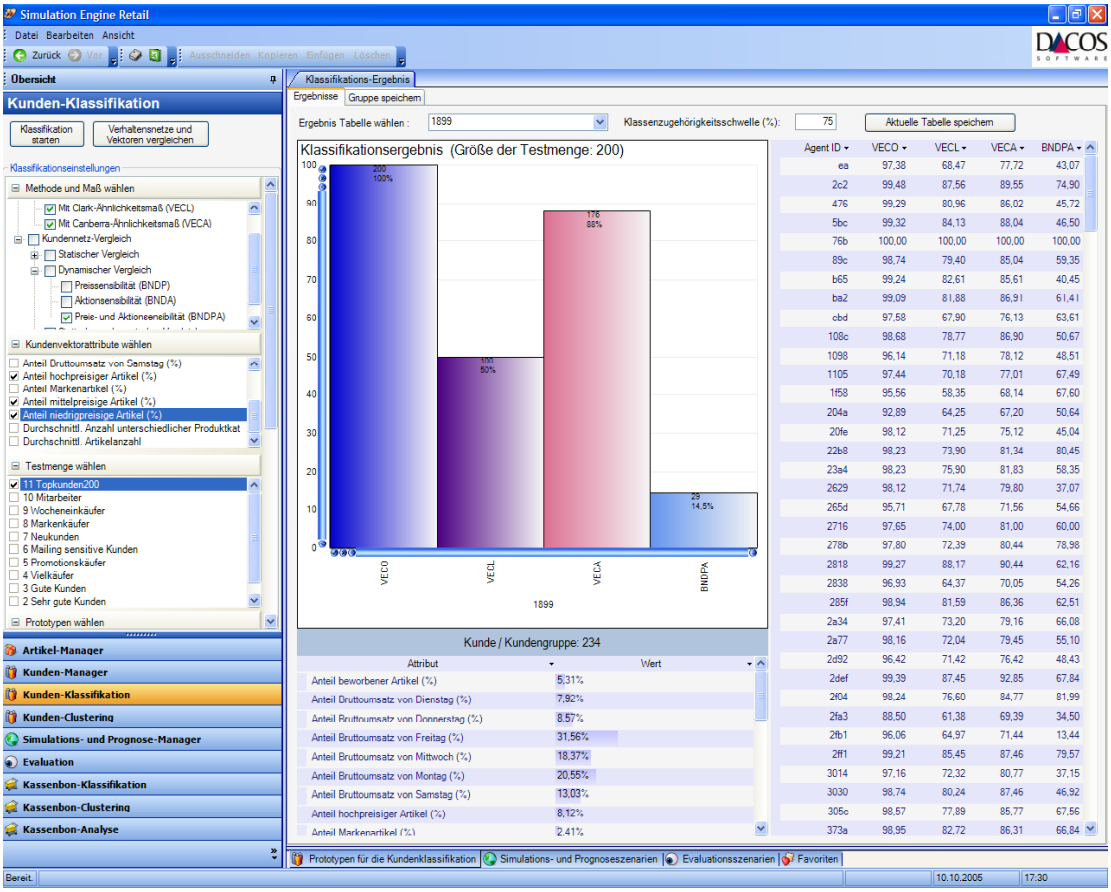


Abbildung 81: Kundenklassifikation mit Prototypen

Das Ergebnis der Klassifikation wird durch das Balkendiagramm in der Mitte des Bildschirmes angezeigt. Bei einem Schwellenwert von 75 Prozent sind nach dem Kosinus-Maß (VECO) 100 Prozent, nach Clark (VECL) 50 Prozent und nach Canberra (VEAC) 88 Prozent der ausgewählten Kunden sehr preis- und werbesensibel, während es nach dem verhaltensnetzbasierten Maß nur 14,5 Prozent sind. Diagramm 10 zeigt die entsprechenden Häufigkeiten bei veränderten Schwellenwerten (70 bzw. 80 Prozent). In allen Fällen ermittelt das verhaltensnetzbasierte Maß eine deutlich geringere Anzahl preis- und werbesensibler Kunden, wodurch die Vermutung nahe liegt, dass die Kunden durch das verhaltensnetzbasierte Verfahren detaillierter differenziert werden können. Diese Annahme verstärkt sich bei Betrachtung von Diagramm 11, in dem das detaillierte Ergebnis der Klassifikation ersichtlich ist. Die Kunden wurden dabei nach absteigender BNDPA-Ähnlichkeit sortiert. Sämtliche Werte der drei vektorbasierten Maße schwanken im Bereich von ca. 65 bis 100 Prozent, während nach BNDPA nur ein geringer Teil der Kunden eine Ähnlichkeit von mehr als 70 Prozent aufweist bzw. ein sehr großer Teil nur sehr geringe Ähnlichkeitswerte (unter 50 Prozent) besitzt.

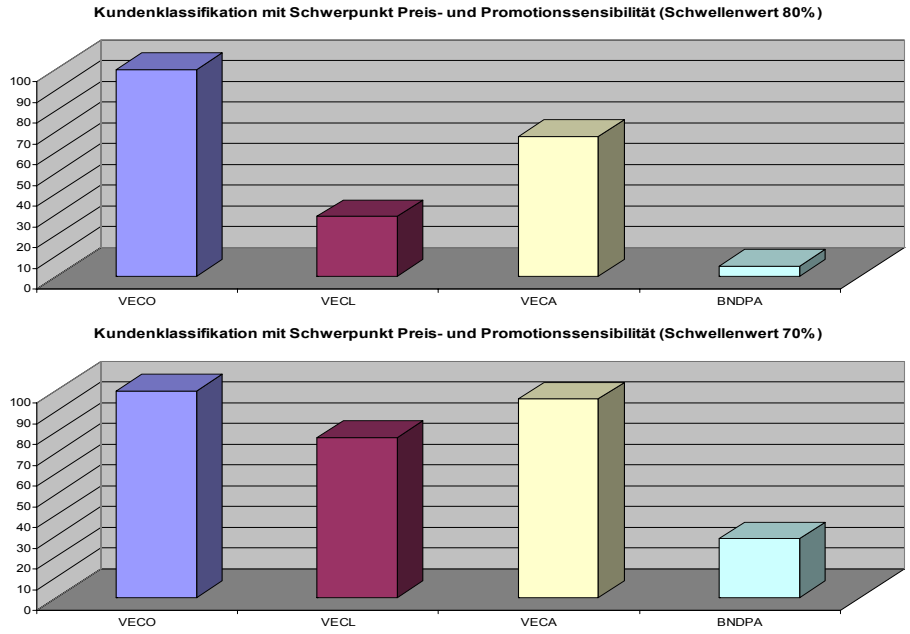


Diagramm 10: Klassifikationsergebnis: Schwerpunkt Preis- und Promotionssensibilität

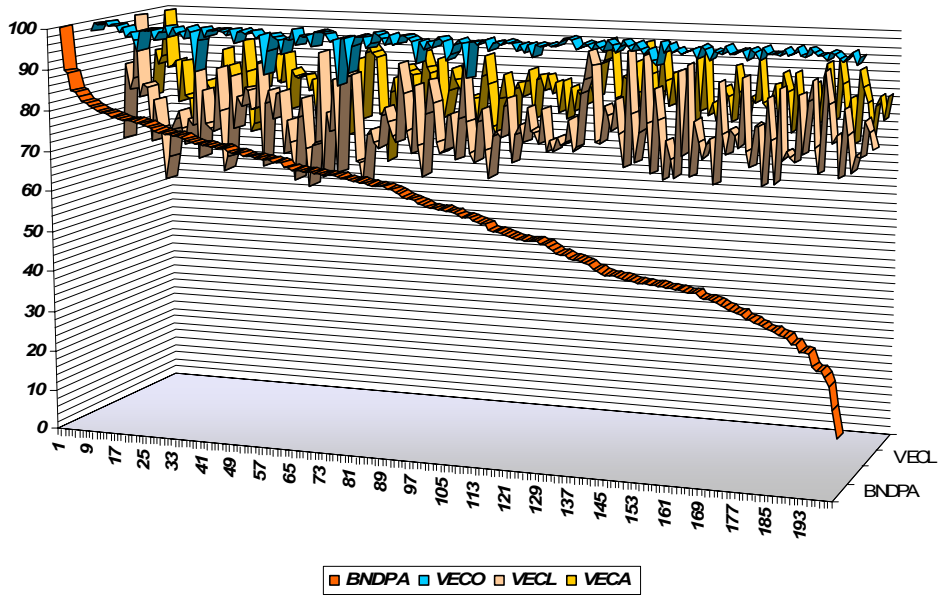


Diagramm 11: Detailliertes Klassifikationsergebnis

Durch die verhaltensnetzbasierte Klassifikation können, aufgrund der genaueren Differenzierung von Kunden bezüglich dynamischer, verhaltensbasierter Merkmale und simulierter Reaktionen, eine Vielzahl verhaltensbezogener Fragestellungen des Marketings

unterstützt werden. Beispielsweise lässt sich die Menge der von geplanten Maßnahmen - wie z. B. Preisänderungen, Promotionen oder Sortimentsveränderungen - negativ bzw. positiv betroffenen Kunden im Vorfeld ermitteln. Ebenso können Zielgruppen für bestimmte Maßnahmen, wie beispielsweise Mailing-Aktionen mit kundenindividuellen Angeboten, exakter ermittelt werden. Darüber hinaus können Erkenntnisse bezüglich des Erfolgs bzw. Misserfolgs filialorientierter Maßnahmen auf andere Standorte übertragen werden, indem der Anteil der erwartungsgemäß positiv bzw. negativ betroffenen Kunden durch entsprechende Kundenklassifikationen ermittelt wird.

6.5 Kundensegmentierung mit anonymen Kassenbons

Um Kundensegmentierungen auf der Gesamtheit der Kassenbons einer Filiale zu ermitteln, können anonyme Kassenbons ohne Kundenbezug mit Hilfe kundengruppenbeschreibender Kassenbonprototypen segmentiert werden (siehe Kapitel 5.8.2). Kassenbonsegmentierungen können dabei entweder durch eindeutige Kassenbonklassifikation oder durch Klassifikation der Ergebnisse von Kassenbonclusterings berechnet werden.

Im Folgenden präsentiere ich anhand praktischer Anwendungsbeispiele die konkreten Umsetzungen beider Verfahren im SimMarket System. Beide Verfahren können sowohl zur Ermittlung von Kassenbon- und Kundensegmentierungen als auch zur Analyse der Häufigkeiten von Kassenbon- und Kundentypen verwendet werden.

6.5.1 Kassenbonklassifikation mit kundengruppenbezogenen Prototypen

Im Rahmen einer Kassenbonklassifikation werden im SimMarket System einzelne Kassenbons durch Attributvektoren beschrieben, die anschließend mit Hilfe unterschiedlicher vektorbasierter Ähnlichkeitsmaße (siehe Kapitel 5.3.1) mit einer Menge vorgegebener Kassenbonprototypen verglichen werden. Die Prototypen beschreiben dabei bekannte Kassenbontypen oder entsprechen typischen, durchschnittlichen Kassenbons bestimmter Kundengruppen (siehe Kapitel 5.8.2).

Abbildung 82 zeigt die graphische Benutzeroberfläche der Komponente zur instanzbasierten Kassenbonklassifikation. Auf der linken Seite wird die Menge der zu klassifizierenden Kassenbons durch Auswahl eines Zeitraumes und weiterer Filterungen vorgegeben. Beispielsweise können Kassenbons eines bestimmten Wochentags oder einer speziellen Kundengruppe ausgewählt werden. Im Beispiel habe ich alle Kassenbons eines konkreten Datums (13.01.2003) einer Filiale des Verbundpartners Globus ausgewählt. Darüber hinaus können einzelne Attribute, die als Grundlage der Vektorrepräsentation der Kassenbons dienen, aus einer Liste selektiert werden, um die Kassenbonbeschreibungen inhaltlich auf die jeweiligen Fragestellungen anpassen zu können.

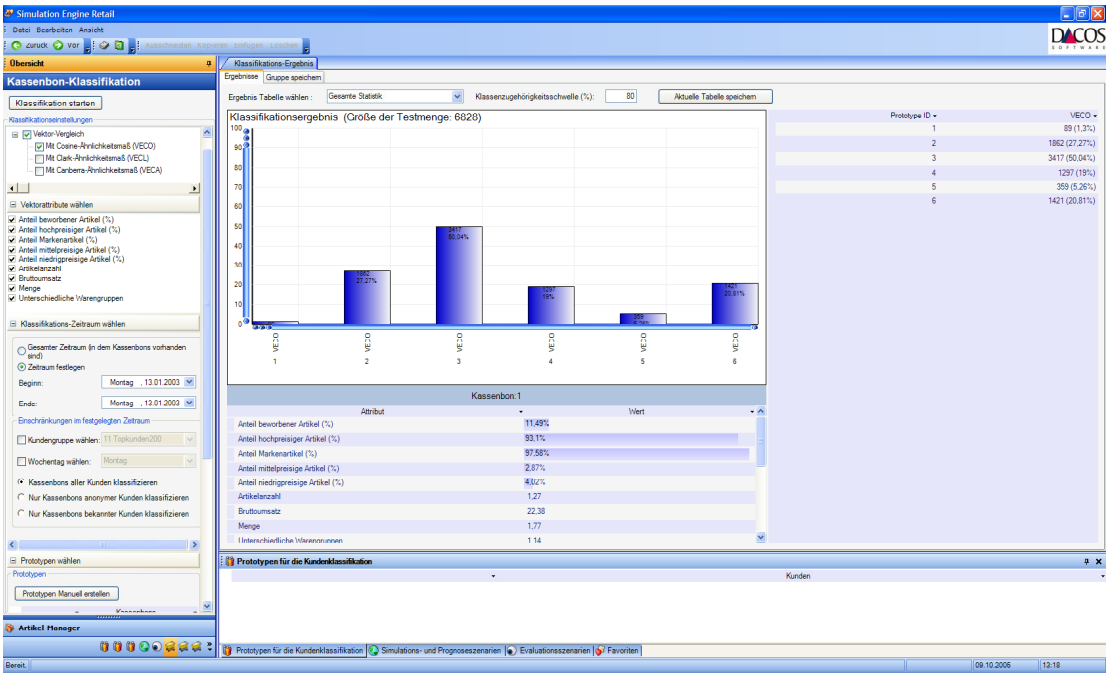


Abbildung 82: Kassenbonklassifikation mit Prototypen

Neben der zu klassifizierenden Kassenbonmenge können aus einer Liste mehrere Kassenbonprototypen sowie unterschiedliche vektorbasierte Ähnlichkeitsmaße vorgegeben werden, die als Grundlage der Klassifikation dienen. Im Beispiel wurden sechs Kassenbontypen sowie das Kosinus-Maß als Vergleichsgrundlage ausgewählt.

Während der Klassifikation werden alle Kassenbons der ausgewählten Menge mit sämtlichen selektierten Kassenbonprototypen mit Hilfe aller vorgegebener Ähnlichkeitsmaße verglichen. Das Ergebnis der Klassifikation ist als Balkendiagramm in der Mitte des Bildschirms ersichtlich. Für jeden der sechs Kassenbonprototypen ist die Anzahl bzw. der Anteil der Kassenbons, die ihnen ähnlicher als ein vorgegebener Schwellenwert sind (hier 80 Prozent), durch die Höhe des Balkens angegeben. Beispielsweise sind ca. 50 Prozent der ausgewählten Kassenbons mehr als 80 Prozent ähnlich zu Kassenbonprototyp 3, der einen durchschnittlichen Billigkäufer-Kassenbon repräsentiert, während ca. 27 Prozent der Bons Prototyp 2 (Vielkäufer) sehr ähnlich sind. Ein Kassenbon kann dabei gleichzeitig mehreren Prototypen zugeordnet werden. Das Ergebnis der Klassifikation vermittelt einen Überblick über die Häufigkeiten bestimmter Kassenbontypen bzw. Kassenbonausprägungen. Mit Hilfe dieser Information lässt sich abschätzen, wie viele Kunden potentiell von Auswirkungen bestimmter Marketing-Maßnahmen, wie beispielsweise der Auslistung typischer Smartshopper-Produkte, betroffen wären.

Kundengruppenbeschreibende Kassenbonprototypen bestehen in der Regel aus *durchschnittlichen* Merkmalsausprägungen der Kassenbons einer Kundengruppe, obwohl

sich deren einzelne Kassenbons bezüglich absoluter Werte deutlich unterscheiden können. Aus diesem Grund verwende ich zur prototypenbasierten Klassifikation einzelner Kassenbons vor allem relative Vergleichmaße, die sich in diesem Fall besser eignen als Maße, die absolute Werte in den Vordergrund stellen. Zur Verdeutlichung zeigt Diagramm 12 im oberen Bereich die Häufigkeiten der Kassenbons, die laut Kosinus-Maß mehr als 80 Prozent ähnlich zu den vorgegebenen Prototypen sind, während unten die entsprechenden Werte des Canberra-Maßes - bei einem Schwellenwert von nur 55 Prozent - angegeben sind. Offensichtlich findet das absolute Canberra-Maß in diesem Fall kaum Ähnlichkeiten zwischen einzelnen Kassenbons und den vorgegebenen relativen Kassenbonprototypen.

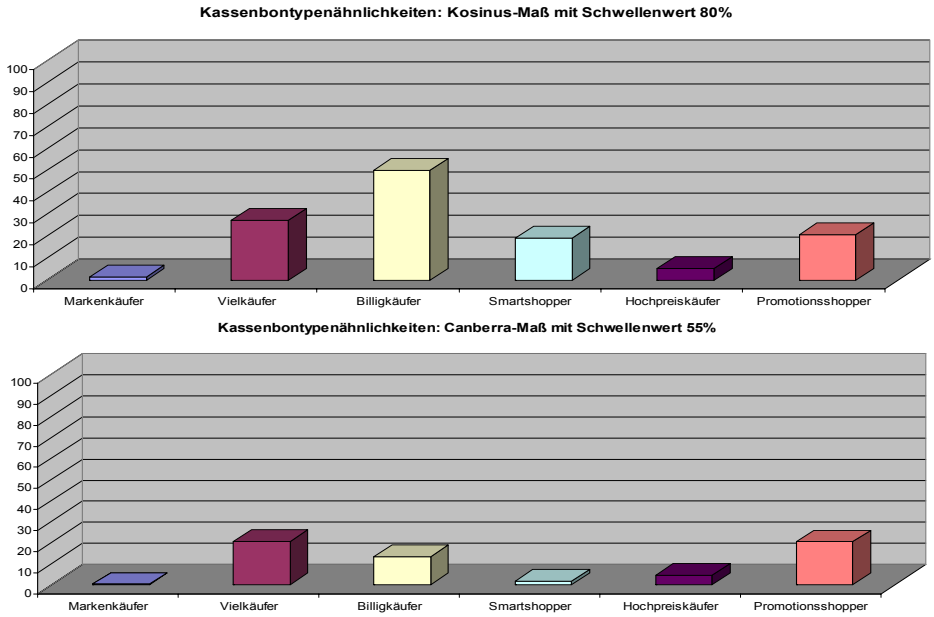


Diagramm 12: Kassenbontypenähnlichkeiten: Kosinus – Canberra

Basierend auf dem Ergebnis der Kassenbonklassifikation kann zur Ermittlung einer Kassenbonsegmentierung eine Partitionierung der Kassenbons durchgeführt werden. Dazu wird jeder Kassenbon derjenigen Klasse zugeordnet, zu der er am ähnlichsten ist. Die Ähnlichkeit muss dabei über einem vorgegebenen Schwellenwert liegen. Falls ein Kassenbon keiner Klasse ähnlich genug ist, wird er der Klasse „Sonstige“ zugeordnet. Diagramm 13 zeigt die Segmentierung des obigen Klassifikationsergebnisses mit einem Schwellenwert von 80 Prozent. Hier zeigt sich, dass 35,11 Prozent der Kassenbons vor allem den *Billigkäufern* zugeordnet werden, während 10,44 Prozent am ehesten von *Promotionsshoppern* stammen sowie 10 Prozent keinem der vorgegebenen Prototypen mehr als 80 Prozent ähneln und somit der Gruppe *Sonstige* zugeordnet wurden.

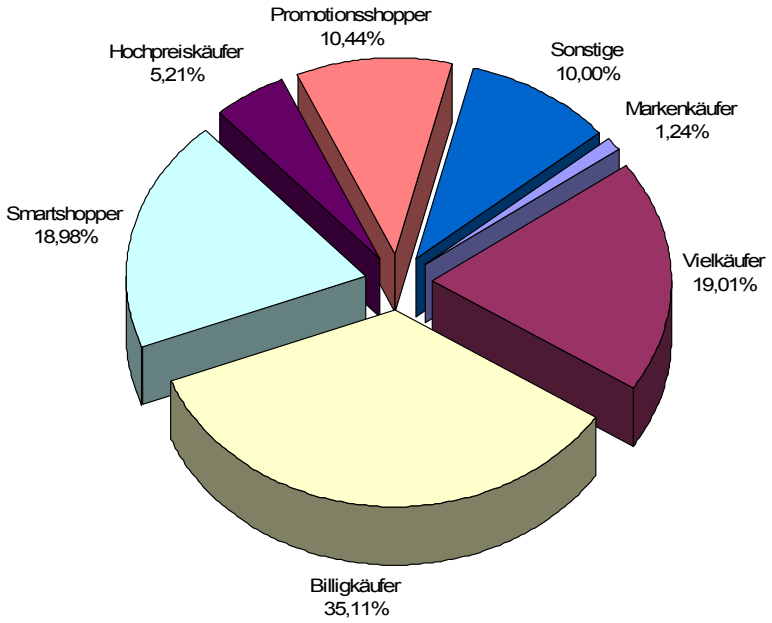


Diagramm 13: Kassenbon-Segmentierung mit kundenbezogenen Klassen

6.5.2 Klassifikation von Kassenbonclustern

Kassenbonsegmentierungen können neben der Klassifikation einzelner Bons alternativ mit Hilfe von Kassenbonclusterings erstellt werden. In Kapitel 5.7 habe ich einen Überblick über Clusteringverfahren gegeben und bin anschließend ausführlicher auf die bekannten Partitionierungsmethoden k-Means und k-Medoid eingegangen, die auch im SimMarket System zur Verfügung stehen.

Abbildung 83 zeigt die graphische Benutzeroberfläche der Komponente „Kassenbonclustering“. Auf der linken Seite der Bildschirmmaske können Einstellungen vorgenommen werden. Neben der Auswahl des konkreten Clusteringverfahrens kann der Inhalt der kassenbonbeschreibenden Attributvektoren konfiguriert werden, indem einzelne Merkmale aus einer Liste selektiert werden. Die Menge der zu clusternden Kassenbons wird durch Angabe eines Zeitraumes vorgegeben, wobei diese Auswahl durch weitere Filter eingeschränkt werden kann. Beispielsweise können die Kassenbons auf eine bestimmte Kundengruppe und/oder einen bestimmten Wochentag beschränkt werden. Ebenso lassen sich auch entweder nur anonyme Kassenbons (ohne Kundenbezug) oder ausschließlich Kassenbons registrierter Kunden auswählen.

Im Beispiel wurden das Verfahren k-Means und sämtliche Kassenbons eines Zeitraumes von zwei Wochen vorgegeben, wobei die Kassenbons auf den Wochentag „Mittwoch“ beschränkt wurden.

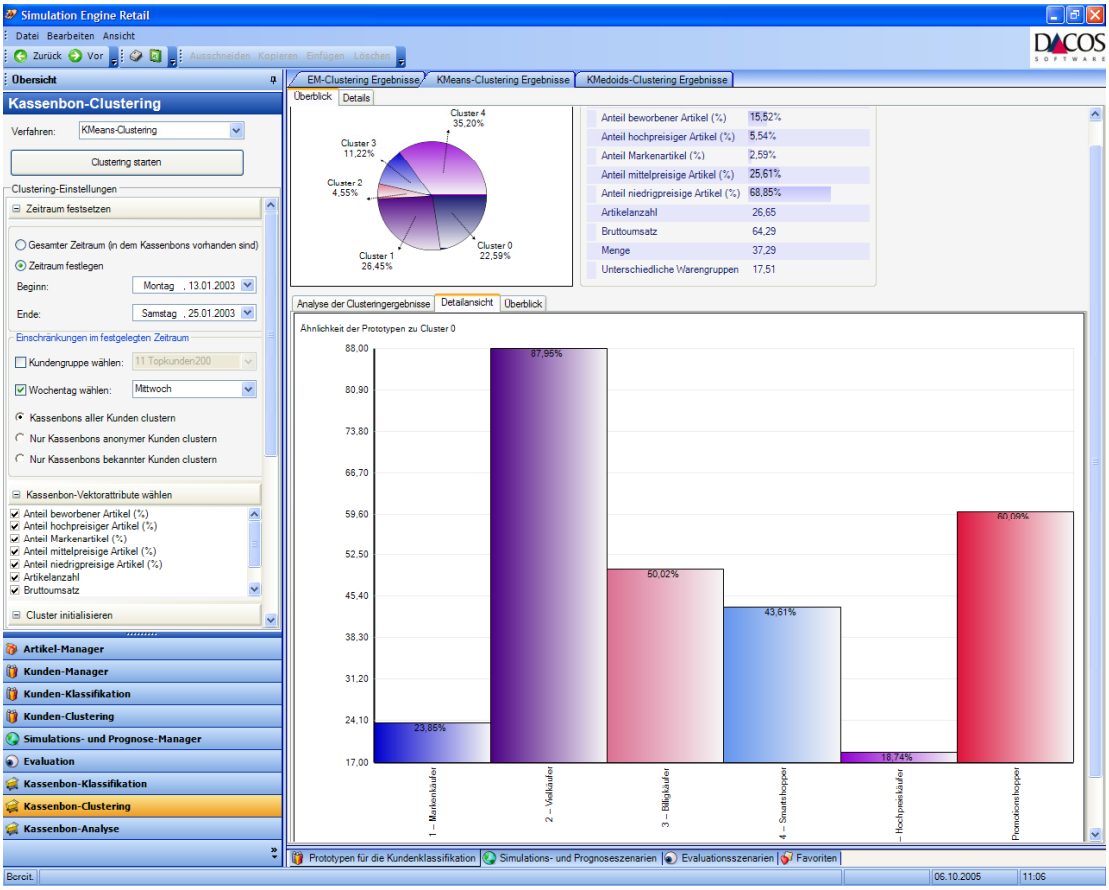


Abbildung 83: Klassifikation von Kassenbonclustern

Das Ergebnis des Clustering ist eine Kassenbonsegmentierung und wird als Kuchendiagramm im oberen Teil der Bildschirmmaske dargestellt.

In meinem Beispiel wurden die vorgegebenen Kassenbons in fünf Cluster (Cluster 0 bis Cluster 4) unterteilt, deren jeweilige Gewichtung an der entsprechenden Beschriftung im Kuchendiagramm ersichtlich ist. Beispielsweise gehören 35,2 Prozent der Kassenbons zu Cluster 4. Nach Auswahl eines Clusters durch Anklicken im Kuchendiagramm wird auf der rechten Seite der dazugehörige Attributvektor angezeigt. Die Werte repräsentieren die durchschnittlichen Attributsausprägungen (also den Mean) der Kassenbons des ausgewählten Clusters.

Durch manuellen Vergleich der unterschiedlichen Clusterattributvektoren können die markanten Merkmale der jeweiligen Cluster herausgefunden werden. Beispielsweise besitzen Kassenbons des selektierten Clusters 0 einen auffällig hohen Anteil niedrigpreisiger Artikel. Alternativ führe ich eine instanzbasierte Klassifikation der durchschnittlichen Clusterattributvektoren mit Hilfe von Kassenbonprototypen durch, die entweder typische Kassenbontypen oder kundengruppenbezogene Attributvektoren repräsentieren. Auf diese

Weise können die Kassenboncluster bekannten Käufertypen bzw. Kundengruppen zugeordnet werden, wobei gleichzeitig die markanten Unterschiede zwischen ihnen ersichtlich werden. Abbildung 83 zeigt das Ergebnis einer Klassifikation des ausgewählten Clusters 0 mit sechs kundengruppenbeschreibenden Kassenbontypen in einem Balkendiagramm. Die Höhe der Balken gibt die Ähnlichkeit des Kassenbonclusters mit dem jeweiligen Kassenbontyp an. Im Beispiel liegen folgende Ähnlichkeiten vor: Markenkäufer (23,85 %), Vielkäufer (87,95 %), Billigkäufer (50,02 %), Smartshopper (43,61 %), Hochpreiskäufer (18,74 %) und Promotionsshopper (60,09 %). Das bedeutet, dass es sich bei den Kassenbons des Clusters 0, im Falle eines Schwellenwertes von 70-80 Prozentpunkten, hauptsächlich um Vielkäuferbons handelt. Wird der Schwellenwert etwas gesenkt, können die Kassenbons als „Vielkäuferbons mit leichter Promotionssensibilität“ bezeichnet werden.

Im Folgenden erläutere ich anhand eines zweiten praktischen Anwendungsbeispiels, welche Informationen sich mit Hilfe von Kassenbonsegmentierungen gewinnen lassen. Diagramm 14 zeigt die resultierende Kassenbonsegmentierung eines Kassenbonclusterings, bei dem als Kassenbonmenge ausschließlich anonyme Kassenbons des Zeitraumes 13.01.2003 bis 25.01.2003 ausgewählt wurden. Das Ergebnis besteht aus fünf Clustern, deren Gewichtung jeweils unterhalb ihres Labels ersichtlich ist. Um die markanten Eigenschaften der Cluster zu ermitteln, wurde wiederum eine instanzbasierte Klassifikation ihrer durchschnittlichen Attributvektoren mit sechs bekannten kundengruppenbeschreibenden Kassenbonprototypen durchgeführt.

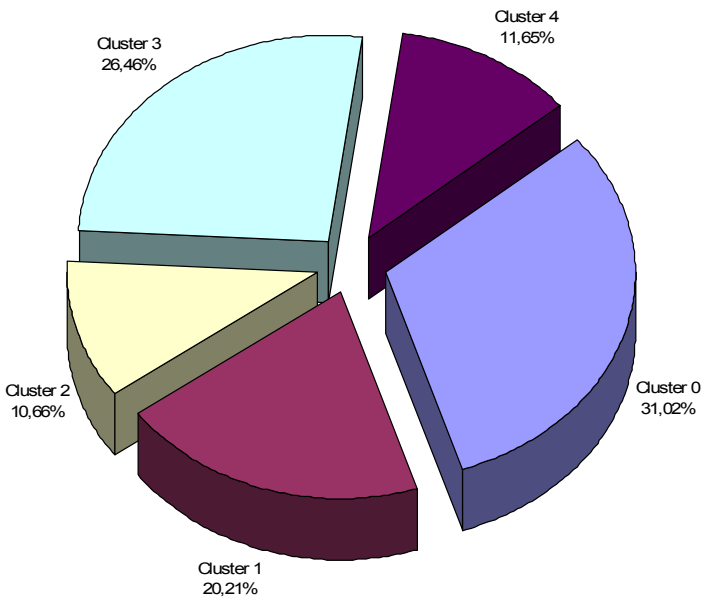


Diagramm 14: Ergebnis eines Kassenbonclusterings

In Diagramm 15 sind die Ähnlichkeiten der jeweiligen Cluster zu den vorgegebenen Kassenbonprototypen dargestellt. Mit geeignetem Schwellenwert ergeben sich folgende für Anwender verständliche Beschreibungen der markanten Clustermerkmale:

- Cluster 0: Billigkäufer mit Ansatz zu Promotionsshopper
- Cluster 1: Smartshopper
- Cluster 2: Vielkäufer
- Cluster 3: Promotionsshopper mit Hang zum Vielkäufer
- Cluster 4: am ehesten Marken- und Hochpreiskäufer

Je nach Fragestellung sind neben der Kassenbonsegmentierung auch die Häufigkeiten der verschiedenen Kassenbontypen interessant. Dazu können je Prototyp die Gewichte aller Cluster addiert werden, deren durchschnittliche Beschreibung (Mean) dem Prototyp ähnlicher als ein vorgegebener Schwellenwert ist. Diese Vorgehensweise hat den Vorteil, dass sowohl die Means der Cluster als auch die Kassenbonprototypen durchschnittliche Attributausprägungen repräsentieren. Aus diesem Grund können Ähnlichkeitsmaße verwendet werden, die absolute und relative Ähnlichkeiten berücksichtigen. Zur alternativen Klassifikation einzelner Kassenbons können hingegen meist nur relative Maße eingesetzt werden, was bei Prototypen mit sowohl absoluten als auch relativen Merkmalen zu Ungenauigkeiten führen kann.

Informationen über die Häufigkeiten von Kassenbontypen können zur Beantwortung unterschiedlicher Fragestellungen verwendet werden. Beispielsweise ist es interessant, inwieweit sich die Häufigkeiten bestimmter Kassenbontypen bezüglich unterschiedlicher Wochentage unterscheiden. Diagramm 16 und Diagramm 17 zeigen die gewichteten Ergebnisse einer Klassifikation von Clustern bezüglich unterschiedlicher Wochentage. Dabei fällt auf, dass die Kassenbons der Wochentage Montag und Samstag einen relativ hohen Anteil an Vielkäufern besitzen, der höchste Anteil an Billigkäufern am Dienstag zu finden ist, Smartshopper Montag und Dienstag bevorzugen, und reine Hochpreiskäuferkassenbons an keinem Wochentag in signifikanter Anzahl vorliegen.

Eine andere interessante Frage ist, inwieweit sich die Häufigkeiten der Kassenbontypen bzw. Kundengruppen bezüglich anonymer und kundenbezogener Kassenbons unterscheiden. Diagramm 18 zeigt eine Gegenüberstellung der Häufigkeiten von sechs Kassenbontypen bezüglich anonymer und kundenbezogener Kassenbons eines Zeitraumes von zwei Wochen (entspricht ca. 120.000 Kassenbons). Der Hauptunterschied liegt darin, dass der Anteil der Promotionsshopper bei anonymen Kassenbons deutlich höher ist als bei kundenbezogenen Bons, während bei registrierten Kunden ein deutlich höherer Anteil von Vielkäufern existiert. Interessanterweise sind darüber hinaus die Ähnlichkeiten der anonymen Kassenbons zu Billigkäufern, Smartshoppern sowie Markenkäufern höher als bei den kundenbezogenen Bons.

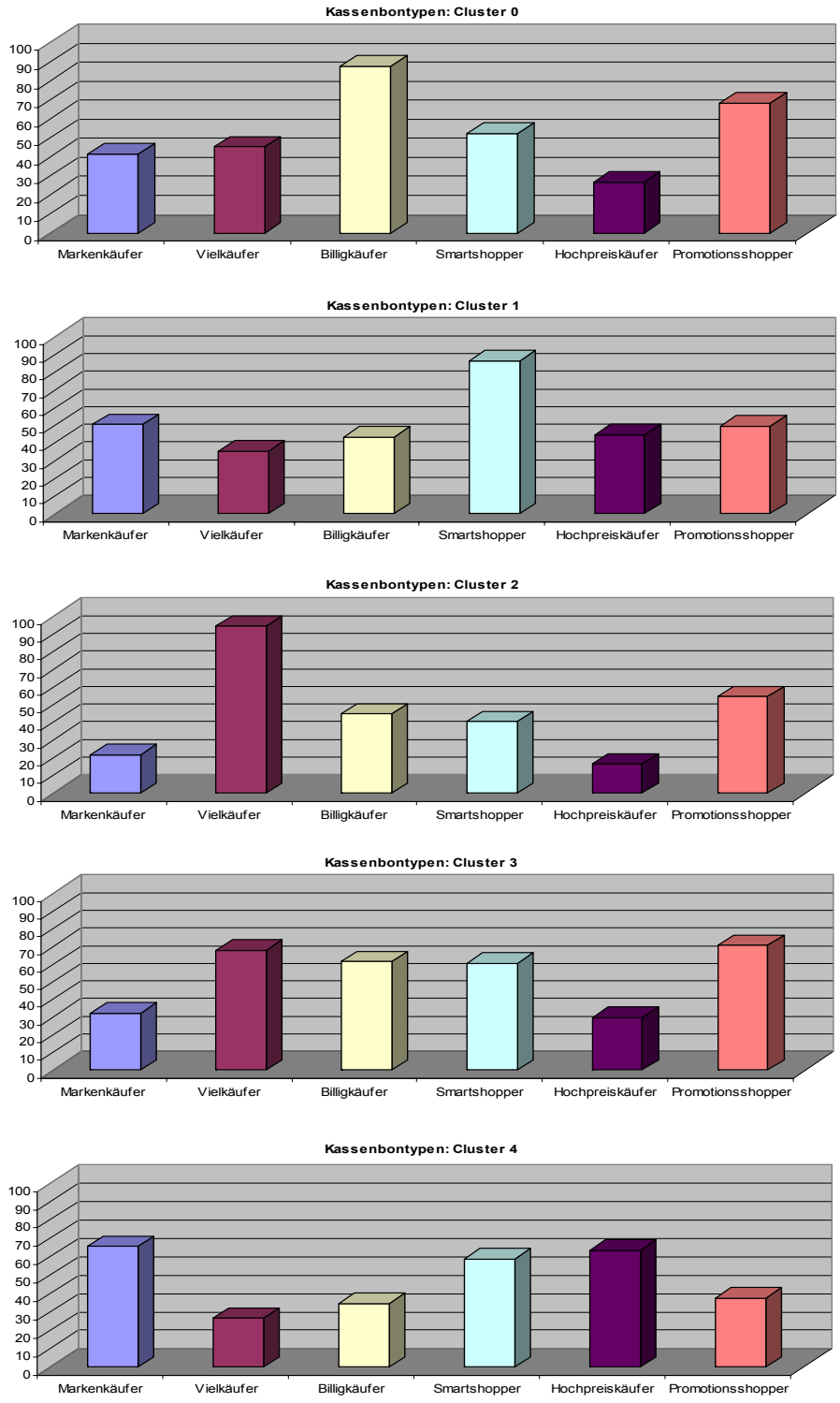


Diagramm 15: Ähnlichkeiten der Cluster mit vorgegebenen Kassenbontypen

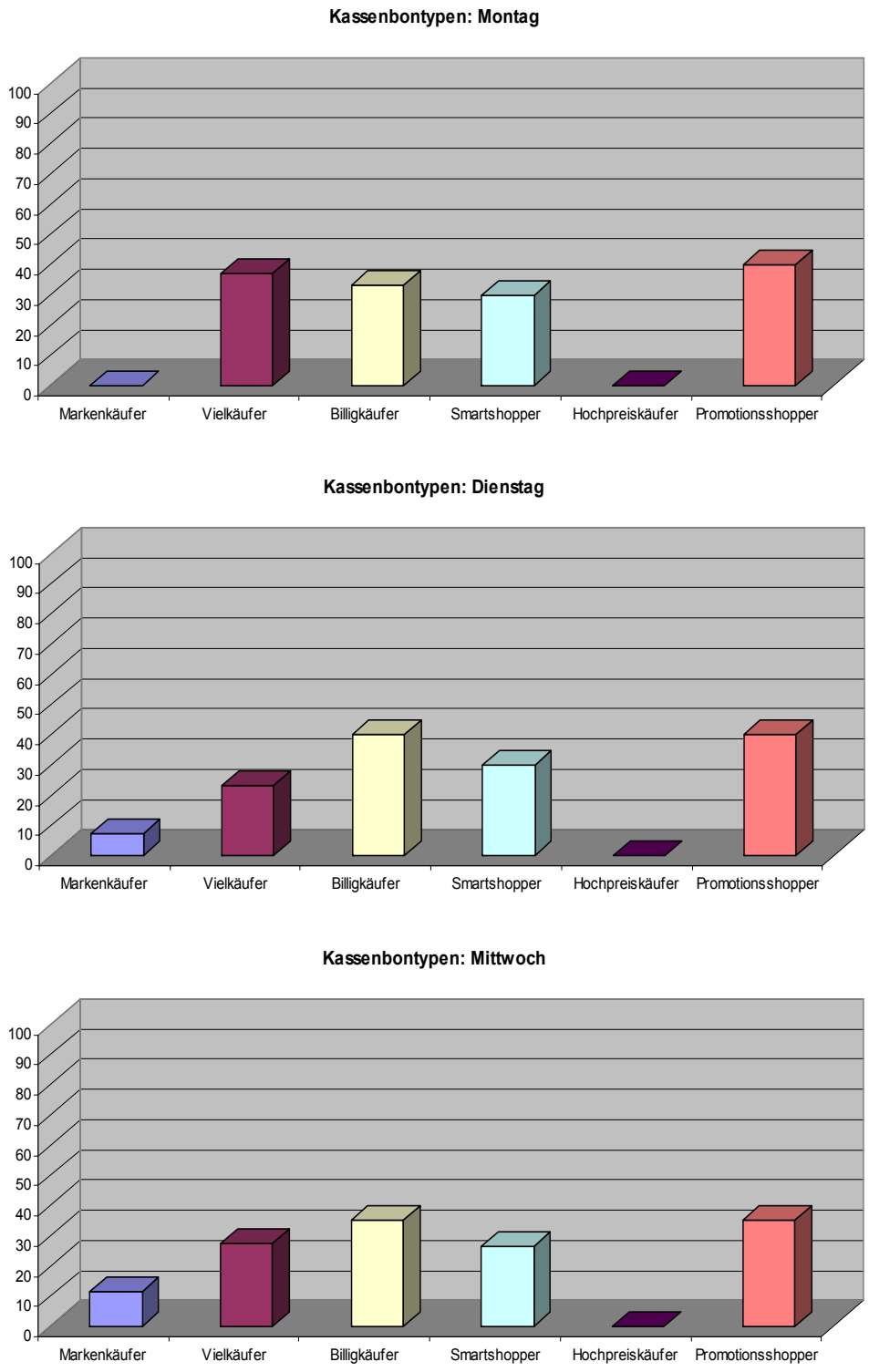


Diagramm 16: Kassenbontypen: Montag bis Mittwoch

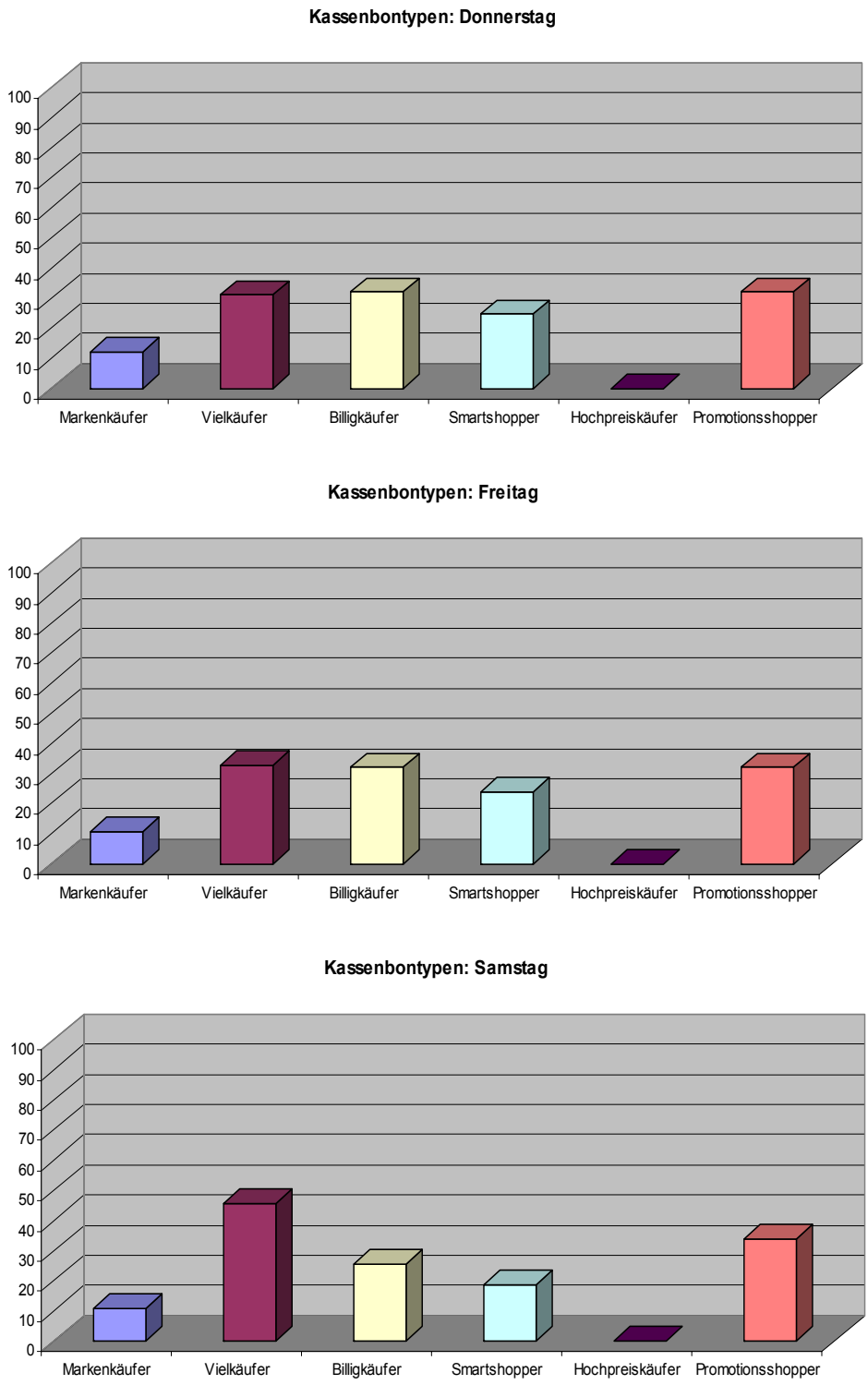


Diagramm 17: Kassenbontypen: Donnerstag bis Samstag

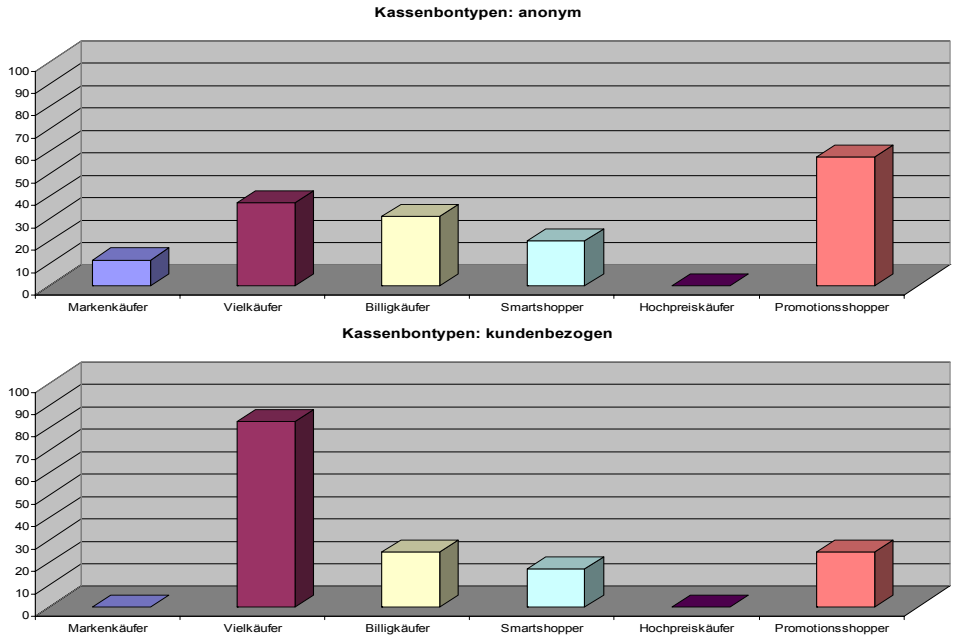


Diagramm 18: Vergleich: anonyme Kassenbons – kundenbezogene Kassenbons

6.6 Verwandte Arbeiten

Im Folgenden gebe ich einen Überblick über verwandte Arbeiten aus den Forschungsbereichen Konsumentenverhalten, Anwender- bzw. Benutzerverhalten und soziale Simulation. Die Arbeiten im Bereich Konsumentenverhalten und soziale Simulation basieren auf der Modellierung virtueller Personen und haben diesbezüglich einige theoretische Annahmen. Dies erschwert die Anpassung der Arbeiten an reale Szenarien, wodurch ein direkter Vergleich mit dem SimMarket System kaum möglich erscheint. Dennoch ergeben sich einige interessante Anknüpfungspunkte.

6.6.1 CONSUMAT

Das *Consumat*-Modell wurde von Wander Jager im Rahmen seiner Dissertation „Modelling Consumer Behaviour“ [JW00] entwickelt und ist eine multiagentenbasierte Umsetzung eines psychologischen Konsumentenmodells. Im Modell werden kognitive Prozesse einzelner Konsumenten von Motivationsfaktoren wie Bedürfnissen, Zufriedenheit, Unsicherheit und verfügbaren Konsummöglichkeiten bestimmt. Falls ein Konsument unzufrieden ist, wird er nach durchführbaren Konsummöglichkeiten suchen, um eine zufrieden stellende Situation zu erreichen. Wenn er unzufrieden aber zugleich unsicher ist, wird er soziale Vergleiche anstellen und verschiedene Möglichkeiten zur Nachahmung anderer Konsumenten

betrachten. Wenn ein Konsument zufrieden und sicher ist, wird er sein bisheriges Verhalten wiederholen. Häufige Wiederholungen bilden die kognitive Basis gewohnten Verhaltens. Letztlich wird jemand, der zufrieden aber unsicher ist, einfach das Verhalten anderer imitieren, die dieselben Möglichkeiten haben. Dieses konzeptuelle Modell wurde in ein Multiagentenmodell transformiert, in dem jeder Konsument durch einen *Consumat-Agenten* repräsentiert wird. Das Modell abstrahiert allerdings sehr stark von realen Kaufakten, wodurch eine Abbildung des Modells auf einen realen Markt erschwert wird, da keine Mechanismen existieren, die aus realen Kunden- und Artikeldaten automatisiert ein entsprechendes berechenbares Modell generieren.

6.6.2 CUBES

Der *CUstomer BEhaviour Simulator* (*CUBES*) von Said, Drogoul und Bouron [SBD01] [SBD02] ist ein Multiagentensystem, das aus bis zu mehreren tausend virtuellen Kundenagenten besteht und zur Simulation des Kundenverhaltens bezüglich verschiedener Wettbewerbssituationen dient. Dabei wird das Kaufverhalten einzelner Agenten durch zwei soziale Eigenschaften (*Imitation* und *Anpassung*) und drei persönliche Verhaltensaspekte (*Misstrauen*, *Opportunismus* und *Innovativität*) modelliert. Dabei steht der Einfluss von Freunden, Familie und Gesellschaft auf individuelle Kaufentscheidungen bezüglich unterschiedlicher Marken im Vordergrund.

CUBES basiert auf der *Swarm Simulation Engine* und umfasst neben der Simulation Engine mehrere Anwendungen zur Steuerung und Auswertung der Verhaltenssimulationen. Ziel ist es, das Gesamtverhalten des Marktes als Zusammenspiel individueller Kunden zu modellieren, die ihr Verhalten hauptsächlich durch Nachahmung und Anpassung verändern. Neben gesellschaftlichen Einflüssen können Werbemaßnahmen der Marken das Verhalten der Kunden beeinflussen, wobei Verkaufspreise nicht berücksichtigt werden. Simulationen umfassen meist zwischen fünf- und siebentausend virtuelle Kundenagenten mit zufälligen Verhaltensausprägungen, die sich innerhalb eines Marktes zwischen mehreren unterschiedlichen Marken entscheiden können. Im Laufe der Zeit wirken sich soziale Einflüsse sowie getätigte Werbemaßnahmen auf die Verteilung des Absatzes der verschiedenen Marken aus. Obwohl die Simulationsergebnisse in vielen Fällen realen Verhaltensmustern entsprechen, eignet sich das Verfahren nicht zur quantitativen Simulation konkreter Szenarien mit realen Kunden und Marken.

6.6.3 SIMSEG

Das *SIMSEG* Projekt [BM01] fokussiert die Analyse von Marktsegmentierungen und Positionierungsstrategien basierend auf der Simulation der Auswirkungen von Image- bzw. Wahrnehmungsänderungen einzelner Marken auf das Kundenverhalten. Dazu werden

Marken und Produkte durch eigenschaftsbeschreibende Wahrnehmungsklassen repräsentiert. Biermarken können beispielsweise durch drei Wahrnehmungsdimensionen mit vier Wahrnehmungsindikatoren beschrieben werden:

- *Intensität des Geschmacks* (stark, schmackhaft, würzig, schwer),
- *Leichtigkeit* (niedriger Alkoholgehalt, wenig Kalorien, erfrischend, leicht),
- *Lifestyle* (cool, jung, ‚in', dynamisch).

Analog werden einzelne Kunden durch Wahrnehmungsprofile modelliert, die ihre Konsumpräferenzen repräsentieren und als Grundlage für Kundensegmentierungen dienen. Zur Analyse vorgegebener Positionierungsstrategien werden einzelne Marken denjenigen Kunden zugeordnet, deren Konsumpräferenzen mit den Angaben im Wahrnehmungsprofil der Marke übereinstimmen. SIMSEG basiert dabei auf theoretischen Annahmen und virtuellen Kundenbeschreibungen, so dass die Abbildung des Modells auf konkrete Szenarien schwierig ist.

6.6.4 InfoSumers, Sphere of Influence und Virtual Consumer

Das *InfoSumers* Simulationssystem von Brannon, Thommesen und Duffield ist ein Multiagentensystem zur Simulation der Diffusion von Innovationen in den Bereichen *Fashion* und *Bekleidung* [BUAMD94]. Die Artificial-Life-Applikation baut auf der *Swarm Software* auf, modelliert das Zusammenspiel externer Einflüsse (wie beispielsweise Werbemaßnahmen) und kundenindividueller Persönlichkeitsprofile (z. B. Trendsetter oder Mitläufer) und simuliert darauf aufbauend kundengruppenbezogene Marktdurchdringungen konkurrierender Marken.

Sphere of Influence basiert ebenfalls auf der Swarm Software und dient zur Modellierung und Simulation der Beziehungen zwischen Herstellern/Lieferanten (Marken) und Endkunden [BUAMD94]. Einzelne Kunden werden dazu als Agenten repräsentiert, deren Beeinflussung von verschiedenen Marken jeweils durch geeignete Nutzenfunktionen modelliert werden. Während eines Simulationsschrittes wird die Zufriedenheit der einzelnen Kunden sowie deren Verteilung auf die verschiedenen Marken ermittelt. Die Simulation unterstützt dabei die Bildung von Allianzen zwischen Herstellern.

Der *Virtual Consumer* [BUAP98] simuliert Kaufentscheidungen einzelner virtueller Kunden bezüglich unterschiedlicher Bekleidungsartikel. Dabei stehen Verbindungen zwischen kundenindividuellen Präferenzen und Produktprofilen im Vordergrund. Mit Hilfe des Virtual Consumer können Entscheider in der Modebranche sowohl kurzfristige als auch langfristige Szenarien mit unterschiedlichen virtuellen Kundengruppen simulieren, um bei der Entwicklung ihrer Unternehmensstrategien unterstützt zu werden. Im Laufe des mehrstufigen Simulationsprozesses werden für einzelne Kunden, auf Basis ihrer persönlichen Profile, Produktpräferenzen erzeugt, die anschließend Verbindungen zu passenden Produkten

erhalten. Das Resultat der Simulation besteht aus der Angabe von Kaufwahrscheinlichkeiten der einzelnen Kunden unter verschiedenen Bedingungen. Das Simulationssystem setzt ebenfalls auf der Swarm Plattform auf und lässt unterschiedliche Modelle zur Erstellung von Verhaltensregeln der einzelnen Kundenagenten zu.

6.6.5 Virtual Market Place Simulator

Der Virtual Market Place Simulator von F. Neri [NF04] modelliert einen virtuellen Markt mit individuellen Kunden und verschiedenen gleichartigen Produkten, die sich hauptsächlich bezüglich ihrer Preise, Bewerbungen, Ausprägungen, Image und anfänglicher Marktdurchdringung unterscheiden. Die Kunden sind als Agenten modelliert, die vier unterschiedlichen Kundengruppen mit unterschiedlichen Präferenzen angehören können: *Bargain Hunters*, *Image Sensitives*, *Description Sensitives* sowie *Image and Description Sensitives*. Die Kunden besitzen darüber hinaus ein eingeschränktes Wissen über das Sortiment, d. h. sie kennen nur eine durch eine Konstante limitierte Anzahl von Artikeln über eine ebenfalls limitierte Zeitspanne.

Während der rundenbasierten Simulation wählen die Kunden in jeder Runde aus den ihnen zu diesem Zeitpunkt bekannten Artikeln denjenigen aus, der ihre Präferenzen am besten erfüllt. Zusätzlich können die Kunden nach jeder Runde mit einer gewissen Anzahl von befreundeten Kunden Informationen über Artikel austauschen. F. Neri zeigt durch Experimente mit 16 Produkten und 400 Kundenagenten (jeweils 100 pro Kundengruppe) mit zufälligem anfänglichem Wissen über Produkte, dass alle Kunden im Falle von anfänglich vollständiger Kenntnis aller Produkte und unterlassener Kommunikation mit Freunden erwartungsgemäß die für sie idealen Produkte kaufen. Im Falle unvollständiger Kenntnis und ebenfalls unterlassener Kommunikation werden auch Produkte gekauft, die den Präferenzen der Kunden nicht optimal entsprechen. Bei unvollständigem Wissen und Austausch von Informationen mit jeweils 20 Freunden, nähert sich das Ergebnis wieder dem Ideal an. Damit zeigt F. Neri, dass der Austausch von Informationen, z. B. auch über das Internet zu optimalen Kaufentscheidungen seitens der Kunden führen kann.

6.6.6 ASPEN

ASPEN ist ein Softwaresystem zur agentenbasierten Simulation der Wirtschaft der Vereinigten Staaten von Amerika, in dem eine große Anzahl ökonomischer Entitäten wie einzelne Haushalte, Firmen, Konsumgüterhersteller und -lieferanten, Banken, Regierung, Immobilienmakler etc. durch individuelle Agenten repräsentiert werden [BPQA96]. Die Simulation läuft in Simulationsschritten ab (elf Schritte pro simuliertem Tag), wobei die Interaktion der Agenten durch den Austausch von Nachrichten realisiert wird. Unter anderem können die Individuen der Haushalte einen Beruf ausüben und mit ihrem Lohn Häuser und

Waren kaufen. Banken geben Kredite, Firmen geben Stellengesuche auf und führen Entlassungen durch, Konsumgüterhersteller produzieren Waren, die wiederum von den Lieferanten an Haushalte verkauft werden, die Regierung setzt Steuersätze fest und nimmt Steuern ein, etc. Das Softwaresystem ist auf einen Hochleistungsparallelrechner angepasst, auf dem sowohl einzelne Agenten als auch Berechnungsprozesse effizient verteilt werden können. Auf diese Weise können Simulationen mit mehreren Tausend Haushaltsagenten durchgeführt werden.

Die Parallele zwischen ASPEN und dem SimMarket System besteht darin, dass sich die individuellen Haushaltsagenten beim Erwerb von Waren und Produkten zwischen den Angeboten konkurrierender Lieferanten entscheiden können. Die Kaufentscheidung wird bei ASPEN durch Wahrscheinlichkeitsfunktionen modelliert, deren Funktionswerte hauptsächlich durch Produktpreise beeinflusst werden.

6.6.7 Benutzermodellierung

Modelle des Konsumentenverhaltens weisen einige Parallelen zur Anwender- bzw. Benutzermodellierung auf. Diese Modelle sind Grundlage benutzeradaptiver Systeme, wie beispielsweise Dialog- oder Lernsysteme, die in der Lage sein müssen, das Verhalten eines Systemanwenders anhand weniger Merkmale zu erkennen, um sich seinen individuellen Bedürfnissen anpassen zu können. Lernprogramme sollten beispielsweise auf den Fortschritt eines Studenten reagieren, indem es sich an dessen individuellen Wissensstand, Lerngeschwindigkeit und Begabung anpasst (siehe [JA96] für einen Überblick). Benutzeradaptive Systeme werden oft auch als Assistenzprogramme eingesetzt. Bohnenberger beschreibt beispielsweise einen benutzeradaptiven Einkaufsassistenten, der einen Kunden in Abhängigkeit seiner bisherigen kundenindividuellen Einkaufshistorie und seiner angegebenen Produktinteressen durch ein Einkaufscenter führt (siehe [BT04]).

In der Regel startet ein benutzeradaptives System mit einem durchschnittlichen Benutzermodell und passt dieses aufgrund des gezeigten Verhaltens des aktuellen Anwenders durch Adaption auf dessen Bedürfnisse an (siehe beispielsweise [JW01]). Zur Benutzermodellierung eignen sich besonders Hidden-Markov-Modelle (HMM) und dynamische Bayes'sche Netze (DBN), die neben bedingten Abhängigkeiten zwischen Zufallsvariablen auch zeitliche Abhängigkeiten berücksichtigen können. Frank Wittig gibt einen allgemeinen Überblick über die Verwendung und Adaption Bayes'scher Netze zur Anwendermodellierung innerhalb benutzeradaptiver Systeme [WF03]. Ein konkretes Fallbeispiel der Anwendermodellierung mit Hilfe dynamischer Bayes'scher Netze bietet Schäfer in [SR98].

7 Zusammenfassung und Ausblick

7.1 Zusammenfassung und wissenschaftlicher Beitrag

Im Zentrum meiner Arbeit steht die Entwicklung eines agentenbasierten, probabilistischen Konsumentenverhaltensmodells zur Repräsentation, Simulation und Analyse des individuellen Kaufverhaltens. Das vorgestellte Modell unterstützt die Planung von Marketing-Strategien und entsprechender Maßnahmenbündel im Handel und dient zur Beantwortung kundenindividueller Fragestellungen im Bereich des Customer Relationship Managements. Dabei stehen die Entwicklung wettbewerbsfähiger Sortiments-, Preis- und Promotionsstrategien und die Identifikation geeigneter Zielgruppen für kundenindividuelle oder kundengruppenbezogene Maßnahmen im Vordergrund.

Betrachtet man bestehende Verfahren, so zeigt sich, dass ihre Sicht auf das Konsumentenverhalten zu „oberflächlich“ ist. Sie können das globale Kundenverhalten nicht als Zusammenspiel kundenindividueller Einzelentscheidungen verstehen, da sie entweder nicht auf Konsumentenverhaltensmodellen basieren oder Modelle verwenden, die entscheidende Nachteile besitzen.

Stimulus-Organism-Response-Modelle setzten meist sehr theoretische Annahmen über kognitive Prozesse voraus und basieren in der Regel auf schwer messbaren Einflüssen und Reaktionen. Reine Stimulus-Response-Modelle verwenden zur Repräsentation des Kundenverhaltens oft zu theoretische Verhaltensmuster, meist in Form einfacher Wenn-Dann-Regeln, die nicht aus historischen Daten gelernt werden. Ein weiterer Nachteil ist, dass die dabei verwendeten Verfahren nicht in der Lage sind, eine Vielzahl von Einflussfaktoren und deren Abhängigkeiten zu modellieren, insbesondere nicht unter Unsicherheit oder im Falle nichtlinearer Abhängigkeiten.

Aus diesen Gründen lässt sich eine Entscheidungsunterstützung zur Lösung kundenindividueller und verhaltensbezogener Fragestellungen, wie sie gerade in der aktuellen Wettbewerbssituation immer entscheidender wird, durch bestehende Verfahren und Modelle nur begrenzt realisieren.

Im Rahmen meiner Arbeit habe ich ein probabilistisches Stimulus-Response-Modell des Konsumentenverhaltens entwickelt, mit dessen Hilfe das individuelle Kaufverhalten einzelner Kunden aus historischen Kundendaten gelernt werden kann, um zu verhindern, dass das Modell auf zu theoretischen Annahmen basiert. Somit beruht mein Modell auf realen Daten und kann zur quantifizierbaren Analyse und Simulation individuellen Kaufverhaltens verwendet werden.

Als Modellgrundlage habe ich mich für ein holonisches Multiagentensystem entschieden,

um sowohl das Konsumentenverhalten einzelner Kunden (Kundenagenten) als auch das aggregierte Verhalten von Kundengruppen (Kundengruppenagenten) modellieren zu können.

Zur Extraktion und Repräsentation des dynamischen Kundenverhaltens habe ich Bayes'sche Netze bevorzugt, die im Rahmen der Arbeit zu probabilistischen Verhaltensnetzen erweitert wurden. Dies führte zur Entwicklung der probabilistischen Agenten, deren Wissensbasis unter anderem aus Verhaltensnetzen bestehen.

Verhaltensnetze eignen sich zur Repräsentation nichtlinearer Abhängigkeiten und Unsicherheiten und verfügen über eine nachvollziehbare, semantische Visualisierung. Darüber hinaus besitzen sie mächtige trainingsbasierte Lernverfahren und können durch verschiedene Arten von Inferenz-Algorithmen zum Schlussfolgern unter Unsicherheit verwendet werden. Mit ihrer Hilfe können die Auswirkungen zahlreicher Einflüsse auf das Kundenverhalten simuliert werden.

Da die auf die Kunden wirkenden Einflussfaktoren unterschiedlichen semantischen Typen (Features) zugeordnet werden können, habe ich das Stimulus-Response-Modell in entsprechende featurebezogene Teilverhaltensmodelle unterteilt. Konkret wurde ein Kundenagent in mehrere Feature-Agenten gegliedert, die featurebezogene Verhaltensnetze verwalten und sich gegenseitig beeinflussen können. Das Gesamtverhalten des Kunden ergibt sich dabei aus dem Zusammenspiel der einzelnen Feature-Agenten, das durch die dafür entwickelte Relationsverschmelzung realisiert wird. Auf diese Weise können die Komplexität der Modellierung und die Laufzeit der Simulation eingeschränkt werden, da nur die für die jeweilige Fragestellung relevanten Feature-Agenten berücksichtigt werden müssen.

Das Konsumentenverhalten eines Kunden kann sich bezüglich verschiedener Artikel deutlich unterscheiden. Aus diesem Grund habe ich die kundenindividuelle Einzelartikel-Feature-Matrix entwickelt, die das Verhalten des Kunden bezüglich sämtlicher relevanten Artikel modelliert. Das Verhalten eines Kunden bezogen auf einen einzelnen Artikel wird dabei wiederum durch Verschmelzung der relevanten Feature-Agenten realisiert.

Um kundenindividuelle Push-, Pull- und Kannibalisierungseffekte zwischen einzelnen Artikeln zu modellieren, habe ich die Artikel-Feature-Matrix zur Artikelgruppen-Feature-Matrix erweitert. Kundenindividuelle Artikelverhaltensnetze werden dabei durch Relationsverschmelzung zu Artikelgruppenverhaltensnetzen vereinigt, in denen kundenindividuelle Abhängigkeiten zwischen einzelnen Artikeln erkannt und modelliert werden.

Da die Komplexität der Verhaltensmodelle trotz der Einteilung in kundenindividuelle Feature- und Artikelverhaltensnetze exponentiell mit Anzahl der Artikel, Features und Feature-Ausprägungen wächst, habe ich Konzepte ausgearbeitet, um die Komplexität und Laufzeit des mehrstufigen Lernprozesses durch Berücksichtigung von Domänenwissen weiter einzuschränken. Dies führte unter anderem zur Einführung unterschiedlicher semantischer

Knoten- und Kantentypen und entsprechender Erweiterungen bestehender Lernalgorithmen, mit deren Hilfe sich der Suchraum in der Praxis beim Erlernen und Verschmelzen von Verhaltensnetzen deutlich reduzieren lässt.

Zur Modellierung zeitlicher Verhaltensabfolgen bzw. Zeitpunktabhängigkeiten des kundenindividuellen Einkaufverhaltens habe ich das Konzept der temporalen Deltaknoten und Deltazustände entwickelt, deren Verwendung in der Praxis zur einer geringeren Komplexität führt als der Einsatz von Hidden-Markov-Modellen und dynamischen Bayes'schen Netzen.

Die Simulation des Konsumentenverhaltens basiert auf der Idee, Kundenagenten virtuelle Teileinkäufe in vorgegebenen Einkaufsszenarien durchführen zu lassen. Dazu werden Szenarien durch zeitliche Abfolgen von Evidenzkombinationen beschrieben, deren Auswirkungen mittels Inferenz-Algorithmen berechnet werden können. Als geeignete Effektvariablen erweisen sich in diesen Fällen vor allem der kundenindividuelle Absatz und Umsatz. Die aus der Inferenz resultierenden Wahrscheinlichkeitsverteilungen der Effektvariablen rechne ich dabei in quantifizierbare Erwartungswerte um. Da Effektvariablen auch als Evidenzknoten verwendet werden können, ist das Verhaltensmodell einheitlich sowohl für Prognosen als auch für Diagnosen und Analysen einsetzbar.

Die Modellierung von Kundengruppen habe ich durch Holonisierung einzelner Kundenagenten zu holonischen Kundengruppenagenten realisiert. Dazu habe ich bestehende Formen der Holonisierung – von der freien Agentengesellschaft bis zur vollständigen Verschmelzung – auf probabilistische Agenten übertragen. Darüber hinaus habe ich eine alternative Holonisierungsart entwickelt, die auf dem vollständigen bzw. partiellen Klonen und Verschmelzen probabilistischer Agenten und ihren Verhaltensnetzen basiert und in der Praxis zur Steigerung der Performance und Modellgüte eingesetzt werden kann.

Da Kundengruppen je nach Fragestellung entweder das individuelle, das aggregierte oder das durchschnittliche Verhalten aller Gruppenmitglieder repräsentieren müssen, habe ich die Additions- und Durchschnittverschmelzung probabilistischer Agenten entwickelt, die auf entsprechenden Verschmelzungsarten der Verhaltensnetze basieren.

Zur Bildung von Gruppen ähnlicher Kunden ist es notwendig, einzelne Kunden miteinander vergleichen zu können. Bestehende Verfahren repräsentieren Kunden dabei mit Hilfe von Attributvektoren, die in der Regel soziodemographische und einfache, statistische Merkmale beinhalten. Sie eignen sich daher nicht zur Bildung von Kundengruppen aufgrund ähnlicher dynamischer Verhaltensweisen, die im Mittelpunkt meiner Arbeit stehen.

Um Kundengruppen bezüglich verhaltensbezogener Merkmale bilden zu können, habe ich die Idee entwickelt, Kunden aufgrund ihres Verhaltens zu vergleichen, das in den entsprechenden Kundenagenten vorliegt. Dabei habe ich zwei unterschiedliche Ansätze betrachtet.

Der erste Ansatz besteht darin, Kunden aufgrund ihrer simulierten Reaktionen auf vorgegebene Szenarien zu vergleichen. Dazu habe ich zwei Verfahren ausgearbeitet, die die individuellen Simulationsergebnisse zweier Kunden mit Hilfe klassischer, vektorbasierter Ähnlichkeitsmaße vergleichen.

Im zweiten Ansatz führe ich einen direkten Vergleich ihrer Verhaltensnetze durch, die innerhalb der Artikel-Feature-Matrix sowie der Artikelgruppen-Feature-Matrix zugleich sehr detailliert als auch standardisiert vorliegen. Dazu habe ich fünf verschiedene Abstands- bzw. Ähnlichkeitsmaße für probabilistische Verhaltensnetze konzipiert.

Ich habe mehrere bestehende vektorbasierte Klassifikations- und Clusteringverfahren so erweitert, dass neben den klassischen Vektorrepräsentationen auch probabilistische Verhaltensnetze als Vergleichsgrundlage verwendet werden können. Auf diese Weise können verhaltensbezogene Klassifikationen, Clusterings und Segmentierungen der registrierten Kunden eines Marktes durchgeführt werden, um beispielsweise geeignete Zielgruppen für kunden- bzw. kundengruppenindividuelle Marketing-Maßnahmen identifizieren zu können.

Da in der Praxis je nach Händler ca. 40 bis 85 Prozent der Kassenbons keinen historischen Kundenbezug besitzen, der zum Erlernen kundenindividueller Verhaltensnetze nötig ist, habe ich Verfahren entwickelt, die auf der Klassifikation anonymer Kassenbons mit Hilfe kundengruppenbezogener Prototypen und Feature-Matchings basieren, um Erkenntnisse, die auf Basis kundenbezogener Kassenbons gewonnen wurden, auf anonyme Kassenbons übertragen zu können.

Im Rahmen des Verbundprojektes SimMarket habe ich maßgeblich an der Entwicklung des SimMarket Systems mitgewirkt. Das System beinhaltet konkrete Implementierungen der von mir ausgearbeiteten Verfahren und Modelle und basiert auf selbstentwickelten Software-Bibliotheken für holonische Agentensysteme und probabilistische Verhaltensnetze. Das SimMarket System dient als Grundlage der Simulation Engine der Dacos Software GmbH.

Mit Hilfe des SimMarket Systems konnte ich die Vorteile der in dieser Arbeit vorgestellten Modelle und Verfahren zur Modellierung, Simulation und Analyse kundenindividuellen Kaufverhaltens in der Praxis nachweisen. Dazu habe ich mehrere praktische Anwendungsbeispiele beschrieben und umfangreiche Evaluationen der szenariobasierten Kundensimulation sowie der unterschiedlichen verhaltensbezogenen Vergleichsverfahren durchgeführt.

7.2 Offene Fragen, Einschränkungen und Ausblick

Die in dieser Arbeit vorgestellten Modelle und Verfahren zur Repräsentation, Simulation und Analyse individuellen Kaufverhaltens eignen sich bereits zur Lösung vieler praktischer Problemstellungen im Handel und speziell im Customer Relationship Management. Um

darüber hinaus weitere offene Forschungsfragen beantworten zu können – beispielsweise, wie exakt *vollständige Einkäufe* einzelner Kunden und deren *genauen Zeitpunkte* vorhergesagt werden können - müssen Verfeinerungen und Erweiterungen bezüglich der Extraktion des Konsumentenverhaltens aus realen Daten, der Verhaltensmodellierung, der szenariobasierten Simulation und der verhaltensbezogenen Vergleichsverfahren vorgenommen werden.

Bei der Extraktion des individuellen Kaufverhaltens aus realen Daten besteht in der Praxis die größte Beschränkung in der meist schwachen oder fehlerhaften Datengrundlage. Zur Verbesserung der Datenqualität können nur die Händler selbst durch entsprechende Optimierungen der Datenerfassung und Speicherung beitragen, während sie die Anzahl der Trainingsfälle in der Regel nicht beeinflussen können. Aus diesem Grund müssen Verfahren angewendet bzw. entwickelt werden, um die geringe Datengrundlage in geeigneter Weise zu vergrößern. Eine Möglichkeit besteht beispielsweise im Einsatz von EM-Lernalgorithmen, die Trainingsfälle berücksichtigen können, in denen Teilinformationen fehlen. Eine weitere Möglichkeit zur Erweiterung der Lernmenge ist die Verwendung relativer Zustandswerte. Dadurch können beispielsweise alle Lernfälle, bei denen Preissenkungen um einen gewissen Prozentsatz durchgeführt wurden, in einem gewissen Rahmen als ähnlich betrachtet werden, obwohl sich die absoluten Werte dabei deutlich unterscheiden können. Mit Hilfe relativer Knotenwerte wird auch die Bestimmung von Analogien zwischen einzelnen Kunden, Kundengruppen und Artikeln vereinfacht. So können Artikel gefunden werden, bei deren relativen Preisänderungen sich ein Kunde sehr ähnlich verhalten hat. Diese Information kann wiederum genutzt werden, um eine größere Lernmenge zu erhalten, indem die Lernfälle dieser Artikel vereinigt werden.

Die Modellierung des kundenindividuellen Verhaltens kann ebenfalls verfeinert werden. Ein Ansatz wäre, zeitliche Faktoren und Handlungsabfolgen stärker zu berücksichtigen. Dies könnte durch Anwendung und Erweiterung der in Kapitel 3.2.6 beschriebenen Deltaknoten und -zustände bzw. dynamischer Bayes'scher Netze realisiert werden. Es sollte nach versteckten Variablen (Hidden Variables) gesucht werden, um erkennen zu können, ob sich bestimmte Verhaltensmuster nicht durch die verwendeten Einflussfaktoren erklären lassen. Gegebenenfalls sollte die Anzahl der betrachteten Einflussfaktoren vergrößert werden, um die Chance zu erhöhen, alle relevanten Einflüsse zu betrachten. Verhaltensnetze sollten je nach Fragestellung angepasste Strukturen, Diskretisierungen und semantische Knotentypen besitzen. Es sollten weitere semantische Inhalte hinzugefügt werden, beispielsweise zur Kennzeichnung von Preiserhöhungen bzw. -senkungen. Ebenso sollte die Diskretisierung der Variablenzustände verbessert werden, da sie maßgeblich die Komplexität der Modellierung sowie deren Qualität bestimmt. Problematisch sind dabei vor allem die Werte der Effektknoten „Menge" und „Umsatz". Diese sollten idealerweise in Abhängigkeit der Zustandskombinationen ihrer Elternknoten diskretisiert werden, wofür bisher geeignete

Verfahren fehlen. Alternativ könnten Diskretisierungen vollständig vermieden werden, indem kontinuierliche Zufallsvariablen verwendet werden.

Um die Güte simulationsbasierter Prognosen bezüglich des kundenindividuellen Konsumentenverhaltens zu verbessern müssen Trends berücksichtigt werden. Die in dieser Arbeit beschriebenen Modelle repräsentieren das durchschnittliche, historische Verhalten der Konsumenten. Diese können zwar durch stärkere Gewichtung jüngerer Lernfälle (Adaption) an das aktuelle Verhalten angepasst werden und somit für kurzfristige Prognosen verwendet werden, allerdings eignen sie sich nicht zur Repräsentation von Trends. Deshalb müssen Trends im Falle starker Verhaltensänderungen bzw. bei mittel- bis langfristigen Prognosen entweder direkt bei der Simulation erkannt und berücksichtigt werden oder durch einen vor- oder nachgelagerten Prozess ermittelt und als Trendkorrekturfaktor in das Ergebnis eingerechnet werden, wofür bisher entsprechende Verfahren fehlen. Darüber hinaus sollten kundenindividuelle Verbundabhängigkeiten - auch zeitlich versetzte - von Artikeln stärker berücksichtigt werden, um vollständige Einkäufe und Einkaufsabfolgen exakter prognostizieren zu können. Interessant wäre es auch, das auf realen Daten basierende System um theoretische Konzepte aus dem Bereich der Gesellschaftssimulation zu erweitern, um soziale Einflüsse und Verhaltensweisen, wie beispielsweise Mund-zu-Mund-Propaganda und Nachahmung, modellieren und simulieren zu können.

Die vorgestellten Vergleichsverfahren für probabilistische Verhaltensnetze können verfeinert werden, indem die zugrunde liegenden Netzstrukturen, semantischen Inhalte und Merkmalsgewichtungen exakter an die jeweilige Fragestellung angepasst werden. Dies könnte beispielsweise durch Verwendung geeigneter fragebezogener, objektorientierter Netzklassen realisiert werden. Auf der Basis verhaltensbezogener Ähnlichkeitsmaße sollten verbesserte Methoden zur Erkennung von Analogien und zur Übertragung von Erkenntnissen von Kunde zu Kunde, Kundengruppe zu Kunde oder Filiale zu Filiale entwickelt werden. Darüber hinaus bietet es sich an, verhaltensbezogene Kundengruppentypologien zu entwickeln, die ausschließlich auf Merkmalen und Ausprägungen von Verhaltensnetzen oder verhaltensnetzbasierten Simulationsergebnissen basieren. Diese könnten in der Praxis alternativ oder ergänzend zu bestehenden Typologien verwendet werden, die sich meist auf soziodemographische oder einfach verhaltensbezogene Merkmale beschränken. Umgekehrt könnte versucht werden, bestehende Kundengruppentypologien auf Feature-Merkmale abzubilden, um entsprechende Kundensegmentierungen mit Hilfe des vorgestellten Feature-Matchings auf Basis von Kassenbonanalysen durchführen zu können.

Das in meiner Arbeit vorgestellte Konsumentenverhaltensmodell kann auf andere Domänen wie Konsumgüterindustrie, Banken, Versicherungen sowie Forschungsgebiete wie Anwender- und Gesellschaftssimulation übertragen werden, da die kunden- bzw. verhaltensbezogenen Fragestellungen in diesen Bereichen sehr ähnlich sind.

Anhang A Marketing - eine kurze Einführung

A.1 Begriff

Der Begriff *Marketing* wurde zum ersten Mal im Jahr 1910 in den USA als Schlagwort für die systematische Vermarktung von Produkten verwendet und war zu diesem Zeitpunkt nichts anderes als ein Synonym für die im deutschsprachigen Raum als *Absatzwirtschaft* bekannte unternehmerische Aufgabe bzw. wissenschaftliche Disziplin [SW04].

In Deutschland kam der Marketingbegriff erst in den 60er Jahren des 20. Jahrhunderts auf und entwickelte sich seitdem in sechs Phasen zum heutigen Begriffsverständnis. Marketing bezeichnet heute die *marktorientierte Unternehmensführung*, bei der die Signale des Marktes systematisch erfasst und berücksichtigt werden [SW04]. Das Unternehmen ist dabei innerhalb der Branche folgenden fünf Wettbewerbskräften ausgesetzt [PME99]:

1. Wettbewerber,
2. potentielle neue Konkurrenten,
3. Anbieter von Substitutionsprodukten,
4. Lieferanten und
5. Abnehmer.

Das Marketing hat innerhalb von Unternehmen eine Doppelfunktion (so genannter *Januskopf des Marketings*): Auf der einen Seite ist es das *Leitkonzept* eines Unternehmens, auf der anderen Seite eine Unternehmensfunktion, nämlich die der konkreten Ausgestaltung der *Absatzfunktion* [SW04].

Als Leitkonzept richtet das Marketing Unternehmen besonders konsequent an den Anforderungen der Kunden und Wettbewerber aus. Dabei steht das Marketing oft mit anderen Unternehmensbereichen in Konflikt, da deren lokalen Ziele den Zielen des Marketings widersprechen können [SD83].

In seiner Rolle als konkrete Ausgestaltung der Absatzfunktion steht das Marketing gleichberechtigt neben den anderen Unternehmensfunktionen und umfasst die marktorientierte Ausübung des Produkt-, Sortiments-, Preis-, Kunden- und Kommunikationsmanagements. Bei der Anwendung dieser Marketing-Instrumente dreht es sich im Wesentlichen um die so genannten *vier P's*: Preis, Produkt, Platzierung und Promotion [MJ60].

A.2 Konzeption, Organisation und Zielsetzung

Abbildung 84 zeigt den idealen Aufbau einer Marketingkonzeption innerhalb eines Unternehmens. Als Grundlage dient die *Marketing-Organisation*, die für sämtliche aufbau- und ablauforganisatorischen Regelungen verantwortlich ist, die zur optimalen Erfüllung der Marketingaufgaben erforderlich sind [SW04]. Die Marketing-Organisation umfasst sowohl die *Aufbauorganisation*, die sich mit der Aufteilung des Unternehmens in Unternehmenseinheiten und der Verteilung von Aufgaben und Kompetenzen beschäftigt, als auch die *Ablauforganisation*, die die inhaltliche, räumliche und zeitliche Abfolge von Arbeitsprozessen regelt. Allgemein kann zwischen *eindimensionalen* und *mehrdimensionalen Organisationssystemen* unterschieden werden. Eindimensionale Systeme richten die Organisation nach einem einzigen Strukturkriterium aus. Dies kann nach *funktionalen* (z. B. Vertrieb, Produktion oder Beschaffung) oder *objektorientierten* Gesichtspunkten (Produkte, Kunden, Regionen oder Projekte) geschehen. Mehrdimensionale Organisationen sind eine Kombination von zwei (*Matrixorganisation*) oder mehr (*Tensororganisation*) eindimensionalen Organisationsformen. Eine spezielle kundenorientierte und für die vorliegende Arbeit interessante Ablauforganisationsform ist das *Customer Relationship Management* (*CRM*) (siehe Anhang A.5).

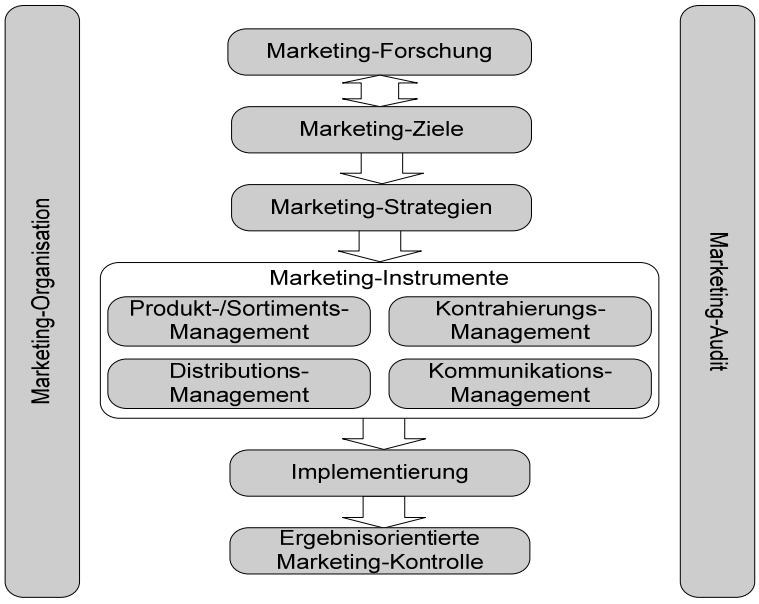


Abbildung 84: Bausteine der Marketing-Konzeption[5]

[5] Quelle: [SW04]

Im Rahmen der *Marketing-Forschung* werden Unternehmens- und Umweltsituation, basierend auf systematisch gesammelten, internen und externen Daten, mit Hilfe von *Chancen/Risiken-* und *Stärken/Schwächen-Analysen* bewertet. Diese Informationen dienen als Grundlage der Entwicklung von *Marketing-Zielen*, die anzustrebende zukünftige Sollzustände beschreiben und als Ausgangspunkt der Marketing-Kontrolle dienen. Aus den Zielen werden konkrete *Marketing-Strategien* und der entsprechende *Marketing-Mix* abgeleitet. Marketing-Strategien legen den langfristig ausgerichteten *Verhaltensplan* eines Unternehmens fest. Eine ausführliche Einführung in die Entwicklung und Verwendung von Strategien und Plänen im Marketing gibt beispielsweise [MM02].

Eine Strategie gibt vor, mit welchen *Marketing-Instrumenten* (siehe Anhang A.4) – d. h. mit welchen konkreten Maßnahmen – vorgegebene Unternehmensziele erreicht werden sollen. Grundsätzlich können Marketing-Strategien in *Instrumental-* und *Basisstrategien* unterschieden werden.

Instrumentalstrategien konzentrieren sich auf bestimmte Marketing-Instrumente, während Basisstrategien verschiedene Instrumente und Maßnahmen kombinieren.

Basisstrategien lassen sich in *kundenorientierte Strategien*, *konkurrenzorientierte Strategien* und *unternehmensübergreifende Strategien* unterteilen. Die wichtigsten Verfahren zur Strategiebestimmung sind:

- GAP-Analyse,
- Produktlebenszyklus-Analyse,
- Portfolio-Analyse,
- ABC-Analyse und
- Break-Even-Analyse.

Die Aufgabe der *Marketing-Kontrolle* umfasst alle Aktivitäten, die den Marketingmanagementprozess sowie dessen Ergebnisse überprüfen. Die *ergebnisorientierte Marketing-Kontrolle* führt ex-post Soll/Ist-Vergleiche durch, um zu prüfen, inwieweit die anvisierten Marketing-Ziele durch die ausgewählten Strategien und Maßnahmen erreicht wurden. Das *Marketing-Audit* hingegen kontrolliert *prozessbegleitend*, inwieweit Planungs- und Implementierungsprozesse angepasst werden müssen. Eine Kombination aus ergebnisorientierter und prozessbegleitender Kontrolle ist das aus einem Kennzahlensystem bestehende *Balanced Scoreboard* [SW04].

A.3 Marketing-Forschung

Die moderne Marketing-Forschung ist eine Erweiterung der *Marktforschung*, die sich auf die Gewinnung von Informationen über die Märkte eines Unternehmens konzentriert und sich

dabei ausschließlich extern erhobener Daten über die so genannte *Mikro-Umwelt* des Unternehmens bedient. Die Mikro-Umwelt besteht aus Objekten, die in einer wechselseitigen Beziehung zum Unternehmen stehen. Die wichtigsten Objekte sind gewerbliche und private Abnehmer, Konkurrenten, Lieferanten, Groß- und Einzelhandel sowie Absatzhelfer wie z. B. Speditionen, Banken oder Unternehmensberater.

Die Marketing-Forschung betrachtet darüber hinaus die *Makro-Umwelt* eines Unternehmens, die aus Umwelteinflüssen besteht, die vom Unternehmen nur geringfügig beeinflusst werden können, wie z. B. wirtschaftliche Rahmenbedingungen, Bevölkerungsstruktur, gesellschaftliche Werte, technische und rechtliche Entwicklungen oder das infrastrukturelle Umfeld. Daneben fällt der Marketing-Forschung die wichtige Aufgabe zu, unternehmensinterne Daten über sämtliche Unternehmensfunktionen und -bereiche zu erheben und zu analysieren.

Die Hauptaufgabe der Marketing-Forschung allerdings ist die Erstellung von *Chancen/Risiken-Analysen* bezüglich der Makro- und Mikro-Umwelt, sowie der Durchführung von *Stärken/Schwächen-Analysen*, bei denen die Ressourcen des Unternehmens in Relation zur Konkurrenz bewertet werden. Die Analysen werden mit Hilfe verschiedenartiger Verfahren (siehe Anhang A.6) durchgeführt und basieren auf externen und internen Daten, die durch Befragung, Beobachtung, Experimente, Haushaltspanels oder Scannings gewonnen wurden. Die Ergebnisse der Chancen/Risiken- und Stärken/Schwächen-Analysen können im Rahmen einer *SWOT-Analyse* [WG01] bzw. mit Hilfe einer *Key-Issue-Matrix* zusammengeführt werden und dienen als Grundlage der Entwicklung von Marketing-Zielen und -Strategien.

A.4 Marketing-Instrumente

Marketing-Instrumente sind das operationalisierte, marktorientierte und strategiekonforme Maßnahmenbündel (so genannter *Marketing-Mix*) zur Erreichung vorgegebener Unternehmensziele. [SW04] untergliedert Marketing-Instrumente in:

- Produkt-, Programm- und Sortimentsmanagement,
- Kontrahierungsmanagement,
- Distributionsmanagement und
- Kommunikationsmanagement.

Produkt-, Programm- und Sortimentsmanagement sind für das marktgerechte Leistungsangebot eines Unternehmens verantwortlich. Das Leistungsangebot von Herstellern und Dienstleistungsunternehmen wird als *Produkt-* bzw. *Angebotsprogramm* bezeichnet, während man bei Handelsunternehmen vom *Sortiment* spricht. Entscheidend ist der richtige

Mix der Produkte und Dienstleistungen. Im Handel bedeutet dies beispielsweise, dass *Umfang*, *Breite* (Anzahl geführter Produkte), *Tiefe* (Anzahl verschiedener Varianten pro Produkt) sowie *Mächtigkeit* (Anzahl der Einheiten pro Variante) des Sortiments marktgerecht gestaltet werden müssen.

Das **Kontrahierungsmanagement** beschäftigt sich mit sämtlichen Marketing-Maßnahmen, die der Gestaltung und Durchsetzung von Preisen für die vom Unternehmen angebotenen Produkte und Dienstleistungen dienen. Die wichtigsten Instrumente des Kontrahierungsmanagements sind das *Preismanagement* und das *Konditionenmanagement*.

Das Preismanagement umfasst alle Maßnahmen und Entscheidungen, die die Entgelte (Preise) für die vom Unternehmen angebotenen Leistungen und Produkte beeinflussen und am Markt durchsetzen. Die Hauptaufgaben sind dabei Preispositionierung, Preisstrategie, Bestimmung der optimalen Preisabstände sowie Preisdurchsetzung. Bei der Festlegung der Preise müssen vor allem die folgenden drei wichtigen Einflussfaktoren berücksichtigt werden (so genanntes *Magisches Dreieck der Preisfindung*): Kosten, Konkurrenten und Konsumenten.

Das Konditionenmanagement bestimmt sämtliche Vereinbarungen, die neben Preisen über das Leistungsangebot festgehalten werden. Die wichtigsten Instrumente sind das *Rabattmanagement* und das *Kreditmanagement*, so wie Regelungen bezüglich der Liefer- und Zahlungsbedingungen.

Das **Distributionsmanagement** umfasst sämtliche Maßnahmen, die den Transfer der Produkte und/oder Dienstleistungen an nachgelagerte Wirtschaftsstufen betreffen [SW04] und beinhaltet darüber hinaus die *physische* und *akquisitorische Distribution*.

Die physische Distribution ist dafür verantwortlich, Leistungen und Produkte vom Ort ihrer Entstehung in den Verfügungsbereich der Käufer zu bringen, während die akquisitorische Distribution den Kontaktaufbau zu Kunden sowie deren Bindung an das Unternehmen zur Aufgabe hat. Die wichtigsten Inhalte des Distributionsmanagement liegen in der Standortwahl, der Wahl der Absatzwege, im Kundenmanagement und der Distributionslogistik.

Das **Kommunikationsmanagement** steuert sämtliche Entscheidungen, die die bewusste Gestaltung von Informationen betreffen, die an Umwelt und Mitarbeiter eines Unternehmens gerichtet sind [SW04].

Die Hauptinstrumente des Kommunikationsmanagement lassen sich in *klassische* und *innovative Instrumente* untergliedern. Zu den klassischen Konzepten gehören vor allem *Werbung*, *Sales Promotions* und *Public Relations*, während *Sponsoring*, *Product Placement*, *Event-Marketing* sowie *Direkt-* und *Multimedia-Kommunikation* zu den innovativen Methoden zählen.

A.5 Customer Relationship Management (CRM)

Das Customer Relationship Management (CRM) beschreibt einen kundenorientierten Ansatz der Marketing-Ablauforganisation. Es integriert und optimiert sämtliche kundenbezogenen Prozesse abteilungsübergreifend in den Bereichen Beschaffung, Produktion, Logistik, Marketing sowie Forschung & Entwicklung mit dem Ziel, nachhaltig profitable Geschäftsbeziehungen mit ausgewählten Kunden aufzubauen und zu pflegen [ABKW02]. Die Hauptaufgabe des CRM besteht in der Optimierung des *Kundenwertes (Customer Lifetime Value)* aller profitablen Kunden, die über die Dauer der Beziehung einen Zahlungsstrom erbringen, der den entsprechenden Kostenstrom des Unternehmens überschreitet [GKS02]. Eine Möglichkeit der Berechnung des Kundenwertes auf der Basis der Kapitalformel ist beispielsweise [KR99]:

$$\text{Kundenwert} = A + \sum_{t=1}^{n} (E_t - K_t) \cdot d^{-t} ,$$

wobei A die Anfangsinvestition, E_t die kundenbezogenen Einnahmen und K_t die entsprechenden Kosten zum Zeitpunkt t, n die geschätzte Dauer der Geschäftsbeziehung, d die Diskontierungsrate $(1+i)$ und i den Zinssatz der Investition definieren.

Die Basis des CRM ist eine unternehmensweite und bereichsübergreifende *Kundendatenbank*, in der Mitarbeitern fundierte, aktuelle, vollständige und entscheidungsrelevante Informationen über Kunden zur Verfügung stehen [SW04]. Darauf aufbauend lässt sich das CRM in drei große Aufgabengebiete unterteilen:

- Kommunikatives CRM,
- operatives CRM und
- analytisches CRM.

Die Hauptaufgabe des **kommunikativen CRM** ist die Optimierung, Steuerung und Synchronisation sämtlicher Kommunikationskänale zum Kunden. Im Mittelpunkt stehen dabei Aufbau und Unterhalt eines *Call Centers* bzw. dessen multimediale Weiterentwicklung zum *Customer Interaction Center*, mit dem Ziel des koordinierten persönlichen Kundenkontakts per Email, Mailing, Telefon, WAP oder TV/Radio. Im Rahmen des so genannten *Multi-Kanal-Marketing*, dem parallelen Gebrauch verschiedener Kommunikationskanäle zu und von einem bestimmten Kunden, ist es erforderlich, die Kommunikation mittels eines *Single-Point-of-Entry* zu synchronisieren, um Missverständnisse zu vermeiden und im Sinne eines *One-Face-To-The-Customer* zu agieren [SW04].

Das **operative CRM** umfasst sämtliche Applikationen zur Unterstützung des direkten

Kontakts zwischen dem Unternehmen und seinen Kunden (so genanntes *Front Office*). Die Hauptaufgabe besteht aus der Automatisierung des Dialoges zwischen einzelnen Kunden und der Synchronisation der dazu erforderlichen Prozesse in den Bereichen Marketing (*Marketing-Automation*), Vertrieb (*Sales-Automation*) und Service (*Service-Automation*) [ABKW02]. Innerhalb dieser drei Bereiche umfasst das operative CRM jeweils administrative, analytische und kontaktunterstützende Aufgaben. Durch Anbindung an *Back-Office-Lösungen*, z. B. *Enterprise Resource Planning (ERP)*, *Supply Chain Management (SCM)* oder *Computer Integrated Manufacturing (CIM)*, unterstützt das operative CRM die Möglichkeit, Kunden über Verfügbarkeit von Waren oder Liefertermine zu informieren [SW04].

Marketing-Automation ist ein Sammelbegriff für den Einsatz marketingunterstützender Systeme in den Bereichen *Database Marketing* (individuelle Kommunikation basierend auf Informationen aus Kundendatenbanken), *Kampagnenmanagement* (koordinierte Ausführung kundenorientierter Marketing- und Vertriebsaktionen), *Kundensegmentierung*, *Kundenbewertung*, Analyse von *Kundencharakteristika* und *Kundenverhalten*, sowie der Erstellung und Verwendung von *Marketing-Enzyklopädien* (multimediale Wissensarchive) und *Produktkonfiguratoren* für Vertriebsmitarbeiter und Kunden.

Die Aufgaben der Sales-Automation sind die Bereitstellung eines Vertriebsinformationssystems für den Innen- und Außendienst mit Funktionen wie *Angebotserstellung/-überwachung*, *Erfolgsrechnung*, *Lost-Order-Analyse* (Analyse von Angeboten, die nicht zu einem Auftrag geführt haben), *Sales-Cycle-Analyse* (Verwaltung von Wiederbeschaffungszeitpunkten zur rechtzeitigen Ansprache von Kunden), *Opportunity Management* (mehrstufige Erfassung von Kundenkontakten), *Interactive Selling Systems* (vertriebsunterstützende, elektronische Produktkataloge und Produktkonfiguratoren), *Eskalationsmanagement* (Beschwerden und Störfälle werden nach einer kritischen Zeitschranke an die nächst höhere Instanz weitergeleitet) und *Workflow-Management* (Koordination kundenorientierter Geschäftsprozesse).

Der Service-Dienst ist für Kunden zuständig, die aus eigener Initiative mit einem Unternehmen in Kontakt treten. Die Service-Automation unterstützt den Service-Dienst mit Lösungen für *Fernwartung*, *Beschwerdemanagement*, Erfassung, Dokumentation und Diagnose von Störfällen *(Helpdesk)* und Beschreibungen von Lösungswegen *(Case Based Reasoning Systeme)*.

Die Aufgabe des **analytischen CRM** ist die systematische Aufzeichnung und Auswertung sämtlicher Kundenkontakte und -reaktionen sowie die darauf aufbauende kontinuierliche Optimierung kundenbezogener Geschäftsprozesse [GKS02]. Im Mittelpunkt stehen die Kundendatenbank, in Form eines *Data Warehouses* oder lokaler *Data Marts* sowie sämtliche auf den Kundendaten basierenden Analyse-Verfahren des *Data Mining* und des *On-Line*

Analytical Processing (*OLAP*). Die Ergebnisse des Data Mining bilden die Grundlage für die ganzheitliche Steuerung des Kundenkontakts über den gesamten Kundenlebenszyklus hinweg [ABKW02], wobei das Data Mining in den einzelnen Phasen der Kundenbeziehung unterschiedliche Beiträge zur Optimierung des Kundenwertes liefern kann [HW04]. Zur Festigung und Intensivierung von Kundenbeziehungen können, z. B. anhand von Kauf- und Zahlungshistorie, neben *Segmentierung* und *Klassifikation* von Kunden auch *Warenkorbanalysen*, *Kundenbewertungen* und *Cross/Up-Selling-Analysen* sowie *Verhaltensprognosen* durchgeführt werden. Darüber hinaus setzt das *Rückgewinnungsmanagement* unter anderem so genannte *Churn-Analysen* zur Identifizierung gefährdeter Kundengruppen ein, um Kündigungen zu verhindern und verlorene Kunden wieder zu reaktivieren.

Obwohl das Customer Relationship Management als eine der wichtigsten Marketing-Innovationen der letzten Jahre gilt, war dessen Einsatz im Handel bisher umstritten [SD02]. Durch den verstärkten Einsatz von Data Warehouses und kundenbezogenen Datenbanken sowie den Einsatz von IT-Systemen für OLAP und Data Mining [HW04] wird es jedoch auch für den Handel immer einfacher, an kundenbezogene Informationen zu gelangen. Zudem eröffnen Internetshops und Versandhandel völlig neue Möglichkeiten des Direkt- bzw. One To One Marketings [ZJ00], um in direkten Kontakt mit einzelnen Kunden treten und kundenbezogene Informationen erhalten zu können.

Aus diesen Gründen werden die im analytischen CRM gewonnenen kundenbezogenen Erkenntnisse verstärkt im operativen Marketing berücksichtigt, wobei sie heute darüber hinaus in Prozesse integriert werden, die ursprünglich nicht das Verhalten einzelner Kunden bzw. Kundengruppen berücksichtigt haben. Beispielsweise kann das *Category Management* (*CM*), bei dem Händler und Hersteller im Sortimentsmanagement, bei der Promotionsplanung und bei Produkteinführungen kooperieren, um den Güter- und Informationsfluss zwischen Industrie, Handel und Verbraucher zu optimieren, zum so genannten *kundengetriebenen CM* erweitert werden, in dem Kundenverhalten und Produktwünsche besser erfasst und zur Optimierung in die Planung miteinbezogen werden [AH02].

A.6 Analyseverfahren im Marketing

Analyseverfahren dienen im Rahmen der Marketing-Forschung zur Verdichtung gewonnener Einzeldaten (*Datenkomprimierung*), Beschreibung von Sachverhalten (*Deskription*) und zur Aufdeckung von Ursache-Wirkung-Zusammenhängen (*Erklärung und Prognose*) [SW04]. Als Datengrundlage dienen dabei Daten aus operativen Unternehmensdatenbanken, Data Marts und Data Warehouses sowie anderen interner (z. B. Scanning) und externer Datenquellen (z. B. Haushaltspanels).

Im Allgemeinen lassen sich Analyseverfahren anhand der Anzahl zu untersuchender Variablen in *univariate*, *bivariate* und *multivariate* Verfahren sowie nach Intension in *strukturprüfende* (*Dependenzanalysen*) und *strukturentdeckende* (*Interdependezanalysen*) Verfahren gliedern. Variablen können dabei je nach Verfahren unterschiedlichen Skalen entsprechen. Mögliche Skalen sind *Nominalskala* (Klassifizierungen qualitativer Eigenschaftsausprägungen, z. B. männlich - weiblich), *Ordinalskala* (Existenz einer Rangordnung, z. B. Produkt *A* ist qualitativ höherwertig als Produkt *B*), *Intervallskala* (Abstände zwischen Skalenwerten sind konstant, z. B. Celsius-Skala) und *Ratioskala* (Existenz eines natürlichen Nullpunktes, z. B. Länge, Gewicht, Einkommen etc.).

Intervall- und Ratioskalen werden als *metrische Skalen*, Nominal- sowie Ordinalskalen als *nicht-metrische* Skalen bezeichnet.

Reine *univariate Verfahren* umfassen *Maß*- und *Verhältniszahlen*, wie z. B. arithmetischer Mittelwert, Median, Varianz sowie Prozent- und Indexzahlen.

Bivariate Verfahren dienen zur Entdeckung und Überprüfung des Zusammenhangs zwischen zwei Variablen. Bekannte bivariate Verfahren sind die *Kreuztabellierung* sowie die *Korrelationsanalyse*.

Multivariate Verfahren analysieren Abhängigkeiten zwischen mehreren Variablen. Die wichtigsten multivariaten Verfahren sind nach [SW04]:

- Regressionsanalyse,
- Varianzanalyse,
- Diskriminanzanalyse,
- Kontingenzanalyse,
- logistische Regression,
- Conjoint Measurement,
- Kausalanalyse,
- Faktorenanalyse,
- Clusteranalyse und
- multidimensionale Skalierung.

Die **Regressionsanalyse** ist ein statistisches Verfahren zur Entdeckung und Modellierung linearer Beziehungen zwischen einer abhängigen und einer bzw. mehreren unabhängigen Variablen (*einfache* bzw. *multiple* Regression) [MPV01]. Ein Beispiel wäre die Abhängigkeit des Abverkaufs eines Produktes von Preis, Qualität und Werbebudget. Erweiterungen der Regressionsanalyse erlauben die Modellierung nicht-linearer Abhängigkeiten (siehe Kapitel 2.3.5.5).

Die **Varianzanalyse** ist ein statistisches Verfahren, das die Varianz einer metrischen Variablen in Abhängigkeit einer oder mehrerer nomineller Variablen (Faktoren) analysiert. Das Verfahren untersucht, ob (und gegebenenfalls wie) sich der Erwartungswert der abhängigen Variablen in verschiedenen Variationen der unabhängigen Variablen unterscheidet, z. B. wie sich unterschiedliche Bewerbungen und Platzierungen auf den Abverkauf eines Produktes auswirken.

Die lineare **Diskriminanzanalyse** (*LDA*) ist ein multivariates Verfahren zur Analyse von Gruppen bzw. Klassenunterschieden, wobei einzelne Elemente durch mehrere Variablen

beschrieben werden. Die Gruppenzugehörigkeit der Elemente wird dabei durch eine abhängige normalskalierte Variable angegeben, während die Gruppeneigenschaften durch mehrere unabhängige metrische Variablen repräsentiert werden. Die Diskriminanzanalyse eignet sich zur Überprüfung der Güte bzw. Optimalität vorgegebener Gruppierungen, zur Ermittlung markanter Unterscheidungsmerkmale für Gruppierungen sowie zur Klassifikation von Objekten. Eine ausführliche Einführung in die lineare Diskriminanzanalyse bietet beispielsweise [HTF01].

Die **Kontingenzanalyse** untersucht Abhängigkeiten bzw. Unabhängigkeiten zwischen mehreren ausschließlich nominalskalierten Variablen anhand von Kontingenztafeln, wobei die Kontingenz den Grad der Wahrscheinlichkeit des gemeinsamen Auftretens von zwei oder mehr qualitativen Merkmalen bezeichnet. Beispielsweise kann mit Hilfe der Kontingenzanalyse der Zusammenhang zwischen der Berufsgruppe einzelner Kunden und dem Kauf einer bestimmten Marke überprüft werden [SW04].

Die **logistische Regression** wird vor allem zur Erklärung von Gruppenunterschieden und zur Einteilung von Elementen in vorgegebene Gruppen verwendet, wobei hier die Bestimmung der Wahrscheinlichkeit der Gruppenzugehörigkeit einzelner Elemente in Abhängigkeit eines oder mehrerer Einflussfaktoren im Mittelpunkt steht [SW04]. Einen Überblick über die logistische Regression sowie einen Vergleich mit der Diskriminanzanalyse bietet [HTF01].

Das **Conjoint Measurement** ist eine Kombination aus Erhebungs- und Analyseverfahren, mit dessen Hilfe sich ordinalskalierte Werturteile bzw. Präferenzen messen und analysieren lassen. Eine typische Anwendung ist die Optimierung der Eigenschaften neuer Produkte, indem Testpersonen aufgefordert werden, kategoriale bzw. ordinale Präferenzurteile über fiktive Produkte abzugeben, deren Eigenschaften systematisch variiert wurden. Die Urteile werden anschließend so miteinander verknüpft, dass der Nutzensbeitrag jeder einzelnen Produkteigenschaft berechnet werden kann [GHH01].

Die **Kausalanalyse** dient der Ermittlung der Korrelation zwischen nicht unmittelbar beobachtbaren (*latenten*) Variablen, beispielsweise psychologischen Konstrukten wie Emotion oder Motivation. Die Kausalanalyse umfasst die Modellebenen *Messmodell* und *Strukturmodell*. Das Messmodell beschreibt die Beziehungen zwischen den latenten Variablen und geeigneten messbaren Indikatorvariablen, während das Strukturmodell die Abhängigkeiten zwischen den latenten Variablen modelliert, die es zu überprüfen gilt [SW04].

Die **Faktorenanalyse** ist ein Verfahren der multivariaten Datenanalyse zur Datenreduktion bzw. Verdichtung von Variablen. Ziel ist es, die Anzahl von Objektmerkmalen auf wenige zentrale Eigenschaften zurückzuführen, unter Abwägung zwischen Komplexitätsreduktion und Informationsverlust. Die Faktorenanalyse wird häufig im Rahmen von

Positionierungsanalysen (*faktorielle Positionierung*) verwendet: falls sich Eigenschaftsbeurteilungen von Objekten durch Reduktion auf zwei oder drei Dimensionen verdichten lassen, können die Objekte im Raum dieser Dimensionen grafisch dargestellt werden [SW04].

Die **Clusteranalyse** gehört wie die Faktorenanalyse zu den Verfahren der Datenreduktion und dient der Typologisierung von Objekten, die bestimmte Messwerte aufweisen bzw. durch verschiedene Eigenschaften beschrieben werden. Ziel ist die Bildung von Objektgruppen, wobei Objekte innerhalb einer Gruppe möglichst ähnlich und die Gruppen untereinander möglichst unähnlich sind. Die Clusteranalyse spielt vor allem zur Bildung von Marktsegmenten auf der Basis nachfragerelevanter Kundenmerkmale eine große Rolle [SW04].

Die **multidimensionale Skalierung** (MDS) ist ein Ähnlichkeitsanalyseverfahren zur Positionierung von Objekten im mehrdimensionalen Raum, wobei Ähnlichkeitsbeziehungen bzw. Distanzen zwischen Objekten räumlich dargestellt werden. Die Dimensionen der Abbildungsräume repräsentieren dabei unkorrelierte Merkmalsvariablen.

A.7 Prognoseverfahren im Marketing

Prognoseverfahren liefern Aussagen über zukünftige Ereignisse basierend auf historischen Beobachtungen, Erfahrungen und theoretischen Erkenntnissen. Grundsätzlich können Prognoseverfahren in *Entwicklungsprognosen* und *Wirkungsprognosen* untergliedert werden [SW04]. Entwicklungsprognoseverfahren „verlängern“ Zeitreihen in die Zukunft, ohne dass das Unternehmen die zu prognostizierende Variable beeinflussen könnte oder wollte (z. B. die Entwicklung der Wirtschaftslage). Wirkungsprognosen dienen zur Vorhersage der Auswirkungen getroffener bzw. geplanter Entscheidungen und Maßnahmen (z. B. wie sich eine Preis- oder Platzierungsänderung bezüglich eines Produktes auf dessen Abverkauf auswirken wird).

Parallel zur Unterteilung in Entwicklungs- und Wirkungsprognose können Prognoseverfahren in *quantitative* und *qualitative* Verfahren gegliedert werden. Während quantitative Verfahren auf mathematischen Methoden basieren und numerische Ergebnisse berechnen (z. B. *Trendextrapolation* und *exponentielle Glättung*), liefern qualitative Verfahren Zukunftseinschätzungen auf der Basis von Erfahrungen bzw. Intuitionen (z. B. *Szenario Technik* und *Delphi-Methode*) [SW04].

Die **Trendextrapolation** ist ein Verfahren zur Erstellung von Prognosen hinsichtlich zukünftiger Entwicklungen. Basierend auf der Annahme, dass durch Analyse langfristiger Zeitreihen ermittelte Regelmäßigkeiten (Trends) auch in der Zukunft gültig sein werden, wird die zukünftige Entwicklung einer Zielvariablen in Abhängigkeit einer oder mehrerer

unabhängiger Variabeln geschätzt [MH92].

Die **exponentielle Glättung** ist ein Verfahren zur Bestimmung vor allem kurzfristiger Trendprognosen basierend auf der Analyse langfristiger, historischer Zeitreihen. Im Gegensatz zur Trendextrapolation sorgt dabei ein *Gewichtungsfaktor* (*Glättungsfaktor*) dafür, dass die Zeitreihenwerte mit wachsender zeitlicher Entfernung vom Prognosezeitraum an Bedeutung verlieren. Das bedeutet, dass jüngere Zeitreihenwerte stärker als ältere gewichtet werden, unter der Annahme einer evolutionären Entwicklung des Marktgeschehens [SW04].

Das mehrstufige Verfahren der **Szenariotechnik** versucht mögliche und in sich stimmige Zukunftsbilder (Szenarien) zu entwickeln [RU91], indem auf Basis der heutigen Situation die positiven bzw. negativen Veränderungen einzelner Entwicklungsfaktoren in möglichen und nachvollziehbaren Zukunftsszenarien zusammengeführt werden. Das Verfahren fokussiert dabei vor allem auf die Modellierung und Analyse der positiven und negativen Extremszenarien sowie des Trendszenarios (so genannter *Szenario-Trichter*) [WP97].

Die **Delphi-Methode** ist ein qualitatives Prognoseverfahren, das auf systematischen, mehrstufigen Expertenbefragungen basiert. Dabei werden Experten gebeten, anonyme Einschätzungen und Antworten bezüglich vorgegebener Fragestellungen abzugeben, die in weiteren Befragungsrunden als Grundlage zur Diskussion, Verfeinerung bzw. Revision der Prognose dienen. Der anonyme Meinungsbildungsprozess wird dabei solange wiederholt bis ein Konsens unter den Experten erreicht wird und ein aussagekräftiges Gruppenurteil vorliegt [NDH02].

Anhang B Bekannte Kundengruppentypologien

Im Folgenden beschreibe ich zur Veranschaulichung zwei in Deutschland und Europa weit verbreitete Kundentypologien. Neben ihnen existiert eine Vielzahl weiterer internationaler, nationaler und regionaler Typologien, wie zum Beispiel die Studie *ACE* (*Anticipating Change in Europe*) des RISC (Research Institute on Social Change) oder die *VALS* (*Values and Life Styles*) des Stanford Research Institute, die beispielsweise in [HR94] ausführlich beschrieben werden.

B.1 Sinus-Milieus

Die Sinus-Milieus von Sinus Sociovision wurden zu Beginn der 80er Jahre basierend auf der so genannten *Lebensweltforschung* entwickelt und unterliegen seit dieser Zeit ständigen Veränderungen. Ziel der Entwicklung der Sinus-Milieus ist eine Abkehr von reinen demographischen Unterscheidungskriterien wie Einkommen, Beruf oder Bildung hin zu einer Typologie, die hauptsächlich Kriterien bezüglich Präferenzen, Einstellungen und Verhaltensweisen zur Unterscheidung verwendet. Durch diese Herangehensweise erhofft man sich Gruppen *gleich gesinnter* Kunden, da erkannt wurde, dass sich bezüglich demographischer Merkmale ähnliche Kunden in sehr vielen Fällen sehr stark im Konsum- und Kaufverhalten unterscheiden können.

Die Sinus-Milieus sind als leistungsfähiges Instrument zur Entwicklung von Marketing-Strategien auf höherer Ebene gedacht, wobei sie zur Unterstützung konkreter Entscheidungen bezüglich einzelner Maßnahmen nicht ausreichen.

Die Sinus-Milieus werden mittlerweile jährlich in zehn europäischen Ländern sowie in Russland, Kanada und den USA erhoben und können in Deutschland in folgende Segmente unterteilt werden (siehe Abbildung 85):

- **Gesellschaftliche Leitmilieus**: Etablierte, Postmaterielle und Moderne Performer
- **Traditionelle Milieus**: Konservative, Traditionsverwurzelte und DDR-Nostalgische
- **Mainstream-Milieus**: Bürgerliche Mitte und Konsum-Materialisten
- **Hedonistische Milieus**: Experimentalisten und Hedonisten

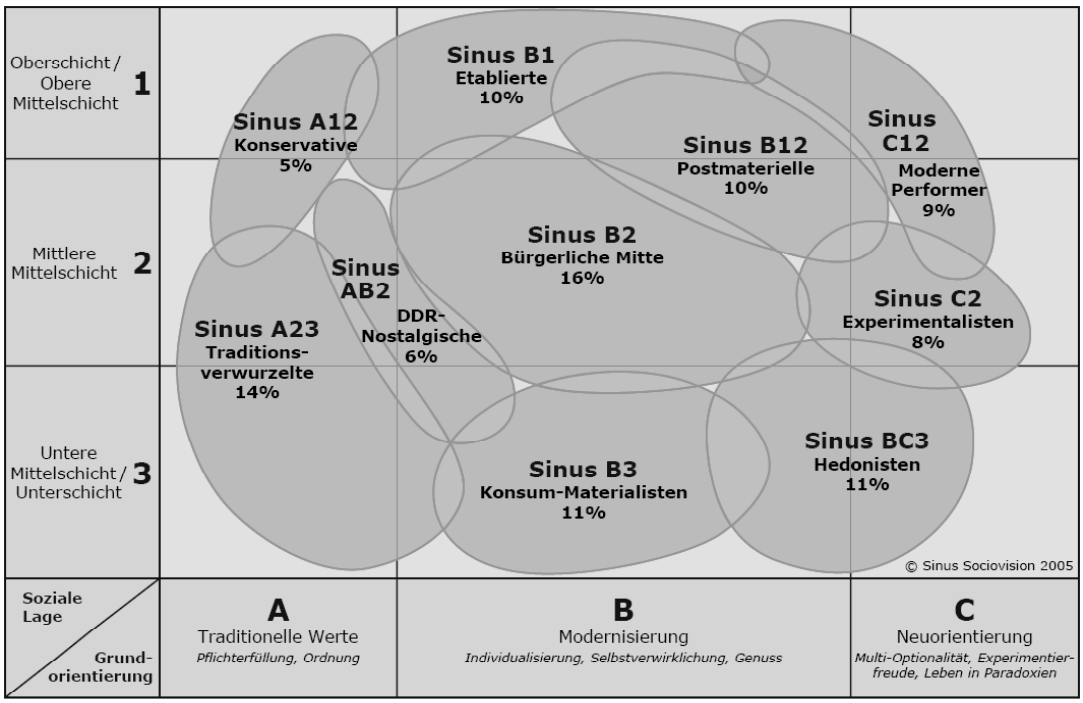


Abbildung 85: Sinus-Milieus 2005 in Deutschland[6]

Sinus A12: Konservative

Das alte deutsche Bildungsbürgertum: konservative Kulturkritik, humanistisch geprägte Pflichtauffassung und gepflegte Umgangsformen.

Sinus B1: Etablierte

Das selbstbewusste Establishment: Erfolgsethik, Machbarkeitsdenken und ausgeprägte Exklusivitätsansprüche.

Sinus B12: Postmaterielle

Das aufgeklärte „Nach 68er-Milieu“: Liberale Grundhaltung, postmaterielle Werte und intellektuelle Interessen.

Sinus C12: Moderne Performer

Die junge, unkonventionelle Leistungselite: intensives berufliches und privates Leben, Multi-Optionalität, Flexibilität und Multimedia-Begeisterung.

[6] Quelle: Sinus Sociovision (www.sinus-sociovison.de)

Sinus A23: Traditionsverwurzelte

Die Sicherheit und Ordnung liebende Kriegsgeneration: verwurzelt in der kleinbürgerlichen Welt bzw. in der traditionellen Arbeiterkultur.

Sinus AB2: DDR-Nostalgische

Die resignierten Wende-Verlierer: Festhalten an preußischen Tugenden und altsozialistischen Vorstellungen von Gerechtigkeit und Solidarität.

Sinus B2: Bürgerliche Mitte

Der statusorientierte moderne Mainstream: Streben nach beruflicher und sozialer Etablierung, nach gesicherten und harmonischen Verhältnissen.

Sinus C2: Experimentalisten

Die individualistische neue Bohème: ungehinderte Spontaneität, Leben in Widersprüchen, Selbstverständnis als Lifestyle-Avantgarde.

Sinus B3: Konsum-Materialisten

Die stark materialistisch geprägte Unterschicht: Anschluss halten an die Konsumstandards der breiten Mitte als Kompensationsversuch sozialer Benachteiligungen.

Sinus BC3: Hedonisten

Die spaßorientierte moderne Unterschicht / untere Mittelschicht: Verweigerung von Konventionen und Verhaltenserwartungen der Leistungsgesellschaft.

B.2 Euro-Styles

Die Euro-Styles wurden Mitte der 70er Jahre des 20. Jahrhunderts als Gemeinschaftsproduktion von Europanel, einem Zusammenschluss von 15 europäischen Forschungsinstituten und dem Centre de Communication Avancée (CCA), entwickelt[7]. Ziel ist die Erstellung einer europaweiten „Landkarte“, die die Lage der einzelnen Styles paneuropäisch, national und regional wiedergibt, um Unterschiede bezüglich der Einstellungen und Verhaltensweisen gegenüber Produkten, Marken, Medien und/oder Verkaufskanälen sichtbar zu machen. Zur Erstellung werden seit den 80er Jahren im 3-Jahres-Rhythmus 24.000 Interviews in 15 Ländern durchgeführt. Durch statistische Analysen

[7] Quelle: medialine.focus.de

wurden vier grundsätzliche Verhaltensmodelle entdeckt (siehe Abbildung 86):

- Uprooted
- Rulemakers
- Surfers
- Traditionalists

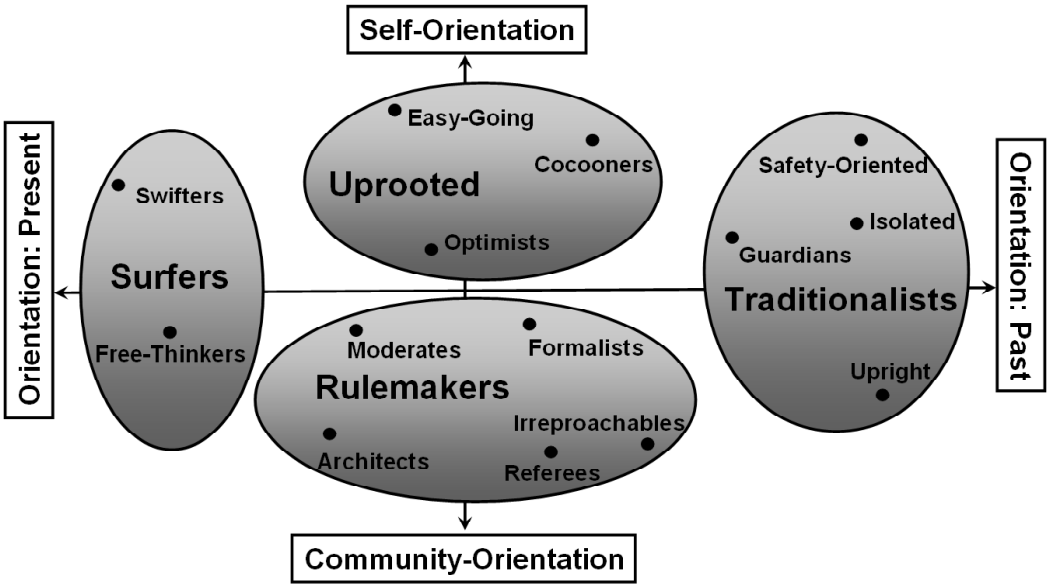


Abbildung 86: Euro-Styles von 1998[8]

Uprooted

Die „Entwurzelten“ zeichnen sich durch Impulsivität, reflexhafte, willkürliche und unkontrollierte Reaktionen auf äußere Reize, Frustration und Revolte aus. Sie lassen sich unterscheiden in:

Cocooners

Junge Haushalte aus einfachem Milieu mit abwehrender Grundhaltung gegenüber sozialen und kulturellen Einflüssen, die ihre Werte mehr und mehr an der Familie und an Kleingruppen ausrichten.

Easy-Going

Junge Opportunisten mit durchschnittlichem, teils hohem Einkommen, modern, materialistisch, die sich immer stärker dafür entschieden haben, ihres eigenen Glückes Schmied zu sein, um ihren „Platz an der Sonne“ zu finden.

[8] Quelle: GfK Lebensmittelforschung

Optimists

Hedonisten jungen und mittleren Alters, mit durchschnittlichem Einkommen, zukunftsoptimistisch, fast schon verträumt, die auf Zuhören, Dialog und gegenseitigen Austausch setzen.

Surfers

Surfers besitzen einen innovativen Geist, der sich allerdings immer mehr der träumerischen virtuellen Wirklichkeit der idealisierten Bilder statt der Realität widmet. Surfers lassen sich unterteilen in:

Swifters

Junge Innovatoren aus gut situiertem Milieu, immer auf der Suche nach neuen Lösungen, sehr weltoffen, bereit, jede Gelegenheit zu nutzen, um bei allen Events mit dabei zu sein.

Free-Thinkers

Wohlhabende, fortschrittliche Intellektuelle jungen und mittleren Alters, in Großstädten lebend, die sich im Rahmen der Selbstverwirklichung verstärkt in eine Welt der Ideen und Kreativität einbringen.

Rulemakers

Rulemakers sind auf der Suche nach moralischer und intellektueller Strenge, Disziplin und Ordnung, um wahrgenomme Krisen zu meistern:

Moderates

Personen überwiegend mittleren Alters, modern, die sich immer stärker auf ein harmonisches Leben in einer moralischen Gesellschaft ausrichten wollen.

Formalists

Vorsorgende Traditionalisten aus der Mittelschicht, die immer stärker Orientierungspunkte und eine klare Vision von der Gesellschaft fordern.

Architects

Wohlhabende Entscheider mittleren Alters, in Städten lebend, kultiviert, verantwortlich handelnd, die zwischen Tradition und Fortschritt navigieren und immer überzeugter sind, dass die soziale Vermittlung ein sehr nützliches konstruktives Mittel darstellt.

Referees

Junge Senioren von Rang und Namen, wohlhabend, in Städten lebend, Befürworter einer ethischen Gesellschaft, die Tradition und Moderne vereint.

Irreproachables

Senioren, die sich in jeder Hinsicht normkonform verhalten, klassisch, zunehmend offen gegenüber Dialog und Verständigung , solange man sich in einem strengen und moralischen Rahmen bewegt.

Traditionalists

Traditionalists sind sicherheitsbedürftig und Werte erhaltend zur Verteidigung von allem Greifbaren, Konkreten, physischen und sozialen Nahen und Unmittelbaren. Sie können unterschieden werden in:

Upright

Provinzielle, konformistische Senioren mit bescheidenem Einkommen, die immer stärker auf ihren Werten und Traditionen bestehen und die sog. Gute alte Zeit wieder aufleben lassen wollen.

Guardians

Junge offensive Senioren aus ländlichen Gegenden mit mittlerem Einkommen, auf der Suche nach Ordnung und Moral, die zunehmend eine Abwehrhaltung zeigen.

Isolated

Zurückgezogen lebende Senioren, Landbewohner, mit bescheidenem Einkommen, die sich mehr und mehr auf die ihnen vertraute Welt konzentrieren und sich auf ihre Wurzeln zurückbesinnen.

Safety-Oriented

Zurückhaltende Senioren mit bescheidenen Mitteln, die in ihrer abwehrenden Einstellung immer radikaler werden.

Anhang C Komplexität der vorgestellten Verfahren im Überblick

Im Folgenden gebe ich einen Überblick über die Komplexitäten der in dieser Arbeit vorgestellten Verfahren und Methoden. Die dabei verwendeten Zeichen und Variablen werden in der letzten Tabelle erläutert.

Lernverfahren	**Komplexität**
Discretisation(W, Z)	$O(\|W\|^2)$
LearnAprioriProbabilityTable(D, X_i)	$O(\|X_i\|\cdot\|D\|)$
LearnConditionalProbabilityTable(D, Y, X_i)	$O(Z^{\|Y\|}\cdot\|X_i\|\cdot\|D\|)$
LearnBehaviourNetworkWithDomainKnowledge(*BC, own*)	$O(n^2\cdot\|D\|\cdot Z^n)$
LearnBehaviourNetworkWithDomainKnowledge(*BC, LamBacchus*)	$O(n^4\cdot\|D\|\cdot Z^n)$

Szenariobasierte Simulationsverfahren	**Komplexität**
SimulateScenario(K_i, S)	$O(\|ST\|\cdot n\cdot Z^n)$

Verhaltensnetzbasierte Vergleichsverfahren	**Komplexität**
CompareComplexBehaviour(K_1, K_2, T, E, R)	$O(\|TE\|\cdot n\cdot Z^n)$
CompareSimulatedReactions(K_1, K_2, S, R)	$O(\|S\|\cdot\|R\|\cdot\|ST\|\cdot n\cdot Z^n)$

Direkte Vergleichsmethoden für probabilistische Netze	**Komplexität**
CompareExternalCorrelations(N_1, N_2)	$O(n\cdot\|D\|)$
CompareInternalCorrelations(N_1, N_2) // Var. 1-3	$O(n^2\cdot Z^2)$
ComparePTs (N_1, N_2) // Var. 1, 2a und 2b	$O(n\cdot Z^n)$
CompareMarginalProbabilities(N_1, N_2) // Var. 1, 1a, 2 und 2a	$O(n\cdot Z)$
CompareEvidenceSampleResults(N_1, N_2, TE) // Var. 1 und 2	$O(\|TE\|\cdot n\cdot Z^n)$

Strukturadaptionsmethoden	**Komplexität**
AdaptTo(N_1, N_2)	$O(n^2+n\cdot Z^n)$
CreateNetStructureUnion(N_1, N_2)	$O(n^2)$
AdaptNetStructures(N_1, N_2) // Var. 1	$O(n^2+n\cdot Z^n)$
AdaptNetStructures(N_1, N_2) // Var. 2	$O(n\cdot\|V\|^2+n^2+n\cdot Z^n)$

Verhaltensnetzbasierte Klassifikationsverfahren	**Komplexität**
ClassifyCustomersWithClassifier(*C*, *T, K*) // Var. 1 mit Naïve-Bayes	$O((T+K)\cdot n\cdot Z^2)$
ClassifyCustomersWithClassifier(*C*, *T, K*) // Var. 1 mit BN	$O((T+K)\cdot n\cdot Z^n)$
ClassifyCustomersWithClassifier(*C*, *T, K*) // Var. 2 mit Naïve-Bayes	$O((T+K)\cdot n\cdot\|C\|\cdot Z^2)$
ClassifyCustomersWithClassifier(*C*, *T, K*) // Var. 2 mit BN	$O((T+K)\cdot n\cdot\|C\|\cdot Z^n)$
ClassifyCustomersWithInstances(*C*, *I, K, k*) // Var. 1	$O(\|K\|\cdot\|I\|\cdot\|AV\|)$
ClassifyCustomersWithInstances(*C*, *I, K, k*) // Var. 2	$O(\|K\|\cdot\|I\|\cdot n\cdot Z^n)$
ClassifyCustomersWithPrototypes(*C*, *P, K*) // Var. 1 und 1a (n. V.)	$O(\|K\|\cdot\|P\|\cdot\|AV\|)$
ClassifyCustomersWithPrototypes(*C*, *P, K*) // Var. 2 und 2a (Netze)	$O(\|K\|\cdot\|P\|\cdot n\cdot Z^n)$

Verhaltensnetzbasierte Clusteringverfahren	**Komplexität**
KMeansClusteringWithAttributVectors(*K, k, threshold*)	$O(\|K\|\cdot k\cdot i\cdot\|AV\|)$
KMeansClusteringWithBehaviourNetworks(*K, k, threshold*)	$O(\|K\|\cdot k\cdot i\cdot n\cdot Z^n)$
KMedoidClusteringWithAttributVectors(*K, k, threshold*)	$O(\|K\|^2\cdot k\cdot i\cdot\|AV\|)$
KMedoidClusteringWithBehaviourNetworks(*K, k, threshold*)	$O(\|K\|^2\cdot k\cdot i\cdot n\cdot Z^n)$

Segmentierungsverfahren	**Komplexität**
CustomerSegmentationByClassQuota(*C*, *K, T*)	$O(\|K\|)$
CustomerSegmentationByClassification(*CP*, *K, T*)	$O(\|K\|)\cdot CLAMC$
CustomerSegmentationByClassification(*CP*, *K, T, threshold*)	$O(\|K\|)\cdot SIMMC$
CustomerSegmentationByClustering(*K, T, k*) // Var. 1 und 2	*CLUMC*
ClassifyCustomerClusteringResults(*C*, *CP, threshold*)	$O(\|C\|\cdot\|CP\|)\cdot SIMMC$
CalculatePrototypesQuota(*K*, *CP, threshold*)	$O(\|K\|\cdot\|CP\|)\cdot SIMMC$
CustomerSegmentationByReceiptSegmentation(*B*, *BP,C*)	*SEGMC*

Zeichen	Bedeutung
W	$W = \{w_1, \ldots, w_n\}$ ist die Menge der distinkten Werte einer DM-Tabellenspalte
X_i	$X_i = \{x_1, \ldots, x_z\}$ ist die Menge aller zulässigen Zustände einer Zufallsvariable X_i
D	$D = \{d_1, \ldots, d_m\}$ ist die Menge der Trainingsfälle (Zeilen) in einer DM-Tabelle
Y	$Y = \{Y_1, \ldots, Y_p\}$ ist die Menge der Elternknoten $Parents(X_i)$ eines Knotens X_i
Z	Z ist die maximale Anzahl von Zuständen, die ein Knoten X_i im Netz besitzen darf
n	n ist die Anzahl der Mitglieder der Knotenmenge $X = \{X_1, \ldots, X_n\}$ eines Netzes N_i
ST	ST ist der maximale in einem Szenario S definierte diskrete Simulationszeitraum
TE	$TE = \{TE_1,\ldots, TE_t\}$ ist die Menge konkreter Zustandskombinationen einer Menge von Einflussvariablen $E = \{E_1,\ldots, E_n\}$
T	$T = \{T_1, \ldots, T_t\}$ ist eine Menge bereits klassifizierter Kunden T_i
K	$K = \{K_1, \ldots, K_k\}$ ist eine Menge unklassifizierter Kunden K_i
C	$C = \{C_1, \ldots, C_m\}$ ist eine Menge von Kundenklassen bzw. Kundenclustern C_i
I	$I = \{I_1, \ldots, I_j\}$ ist eine Menge bereits klassifizierter Kunden I_i (Instanzen)
AV	$AV = (a_1, \ldots, a_v)$ ist ein kundenbeschreibender Attributvektor der Länge $\|AV\| = v$
P	$P = \{P_1, \ldots, P_p\}$ ist eine Menge kundengruppenbeschreibender Prototypen P_i
k	k bestimmt die Anzahl der Cluster bei Partitionierungsmethoden
i	i ist die Anzahl der Iterationen bei Partitionierungsmethoden
CP	$CP = \{CP_1, \ldots, CP_n\}$ ist eine Menge vorgegebener Kundenklassenprototypen CP_i
$CLAMC$	Komplexität des zugrunde liegenden Klassifikationsverfahrens
$SIMMC$	Komplexität des zugrunde liegenden Vergleichsverfahrens
$CLUMC$	Komplexität des zugrunde liegenden Clusteringverfahrens
$SEGMC$	Komplexität des zugrunde liegenden Segmentierungsverfahrens

Literaturverzeichnis

[AL04] L. Agosta. *Data Warehousing Lessons Learned: Data-Mining is Dead – Long Live Predictive Analytics*. DMReview, 2004

[AGGR98] R. Agrawal, J. Gehrke, D. Gunopulos und P. Raghavan. *Automatic Subspace Clustering of High Dimensional Data for Data Mining Applications*. Proceedings of the International Conference on Management of Data, 1998.

[AH02] D. Ahlert und J. Hesse. *Relationship Management im Beziehungsnetz zwischen Hersteller, Händler und Verbraucher*. In Ahlert, Becker, Knackstedt und Wunderlich: *Customer Relationship Management im Handel*. Springer, 2002.

[ABKW02] D. Ahlert, J. Becker, R. Knackstedt und M. Wunderlich. *Customer Relationship Management im Handel. Strategien, Konzepte, Erfahrungen*. Springer Verlag, 2002.

[BB96] G. Bamberg und F. Baur. *Statistik. 9. Auflage*. Oldenbourg Verlag, 1996.

[BLN01] O. Bangsø, H. Langseth und T. D. Nielsen. *Structural Learning in Object-oriented Domains*. Proceedings of the Fourteenth International FLAIRS Conference, 2001.

[BPQA96] N. Basu, R. J. Pryor, T. Quint und T. Arnold. *ASPEN: A Microsimulation Model of the Economy*. Sandia Report SAND96-2459, 1996.

[BJR79] J. R. Bettmann. *An Information Processing Theory of Consumer Choice*. Addison-Wesley, 1979.

[BT04] T. Bohnenberger. *Decision-Theoretic Planning for User-Adaptive Systems: Dealing With Multiple Golas and Resource Limitations*. Dissertation, Universität der Saarlandes, 2004

[BRR95] R. R. Bouckaert. *Bayesian Belief Networks: From Construction to Inference.* Dissertation, Universität Utrecht, 1995.

[BUAMD94] E. L. Brannon, P. V. Ulrich, L. J. Anderson, T. Marshall und A. Donaldson.

Artificial Life Simulation of the Textile/Apparel Marketplace: An Innovative Approach to Strategizing about Evolving Markets. Auburn University, 1994.

[BUAP98] E. L. Brannon, P. V. Ulrich, L. J. Anderson und A. B. Presley. *Agent-Based Simulation of the Consumer's Apparel Purchase Decision*. Auburn University, 1998.

[BME87] M. E. Bratman. *Intentions, Plans, and Practical Reason*. Havard University Press, 1987.

[BIP87] M. E. Bratman, D. J. Israel und M. E. Pollack. *Toward an Architecture for Resource-Bounded Agents*. Technical Report CSLI-87-104, SRI and Stanford University, 1987.

[BM01] C. Buchta und J. Mazanec. *SIMSEG/ACM – A Simulation Environment for Artificial Consumer Markets*. Working Paper Nr. 79, März 2001.

[BH00] H. Bunke. *Recent Developments in Graph Matching*. International Conference on Pattern Recognition, 2000.

[BHD98] H.-D. Burkhard. *Einführung in die Agenten-Technologie.* Informationstechnik und Technische Informatik Nr. 4. Oldenbourg Verlag, 1998.

[BFV98] H.-J. Bürckert, K. Fischer und G. Vierke. *Transportation Scheduling with Holonic MAS – The TeleTruck Approach*. In PAAM, 1998.

[CH98] A. K. Caglayan und C. G. Harrison. *Intelligente Software-Agenten: Grundlagen, Technik und praktische Anwendung im Unternehmen*. Carl hanser Verlag, 1998.

[CKS98] G. Chartrand, G. Kubicki und M. Schultz. *Graph similarity and distance in graphs*. Aequationes Mathematicae, 1998.

[CGKBL02] J. Cheng, R. Greiner, J. Kelly, D. Bell und W. Liu. *Learning Bayesian networks from data: An information-theory based approach*. Artificial Intelligence, Vol. 173, 2002.

[CG99] J. Cheng und R. Greiner. *Comparing Bayesian Classifiers*. Proceedings of the Fifteenth Conference on Uncertainty in Artificial Intelligence, 1999.

[CH52] H. Chernoff. *A measure of asymptotic efficiency for tests of a hypothesis based on the sum of observations*. Annals of Mathematical Statistics Vol. 23, 1952.

[CGH94] D. M. Chickering, D. Geiger und D. Heckerman. *Learning Bayesian Networks is NP-hard*. Technical Report MSR-TR-94-17, Microsoft Research, 1994.

[CL68] C. K. Chow und C. N. Liu. *Approximating discrete probability distributions with dependence trees*. IEEE Transactions on Information Theory, Vol. 14, 1968.

[CZ00] I. Cloete und J. M. Zurada. *Knowledge-Based Neurocomputing*. MIT Press, 2000.

[CGF90] G. F. Cooper. *The computational complexity of probabilistic inference using Bayesian belief networks*. Artificial Intelligence Vol. 42, 1990.

[CH92] G. F. Cooper und E. Herskovits. *A Bayesian method for the induction of probabilistic networks from data*. Machine Learning Nr. 9, 1992.

[CDLS99] R. G. Cowell, A. P. Dawid, S. L. Lauritzen und D. J. Spiegelhalter. *Probabilistic Networks and Expert Systems*. Springer Verlag, 1999.

[CG97] H. Corsten und R. Gössinger. *Entwurf eines konzeptionellen Rahmens für ein Multiagentensystem zur integrativen Unterstützung der Produktionsplanung und –steuerung*. Technical Report Nr. 13, Universität Kaiserslautern, 1997.

[DL93] P. Dagum und M. Luby. *Approximating probabilistic inference in belief networks is NP-hard*. Artificial Intelligence, 1993.

[DP02] P. Davidsson. *Agent-based Social Simulation: A Computer Science View*. Journal of Artificial Societies and Social Simulation, Vol. 5, 2002.

[DLR77] A. Dempster, N. Laird und D. Rubin. *Maximum Likelihood from Incomplete Data via the EM Algorithm*. Journal of the Royals Statistical Society, B 39, 1977.

[DD95] D. Draper. *Localized Partial Evaluation of Belief Networks*. Dissertation, Department of Computer Science, University of Washington, 1995.

[DP94] D. Dubois und H. Prade. *A Survey of Belief Revision and Updating Rules in Various Uncertainty Models*. International Journal of Intelligent Systems No. 9(1), 1994.

[DH73] R. Duda und P. Hart. P*attern Classification and Scene Analysis*. John Whiley & Sons, 1973.

[EBM00] J. Engel, R. D. Blackwell und P. Miniard. *Consumer Behavior. 9. Auflage*. New York, 2000.

[FJ01] J. Ferber. *Multiagentensysteme: Eine Einführung in die Verteilte Künstliche Intelligenz*. Addison-Wesley, 2001.

[FK93] K. Fischer und N. Kuhn. *A DAI Approach to Modeling the Transportation Domain*. Technical Report RR-93-25, DFKI GmbH, 1993.

[FSS03] K. Fischer, M. Schillo und J. Siekmann. *Holonic Multiagent Systems: The Foundation fort he Organization of Multiagent Systems*. Proceedings of the First International Conference on Applications of Holonic and Multiagent Systems (HoloMAS), 2003.

[FG96] S. Franklin und A. Graesser. I*s it an Agent, or just a Program: A Taxonomy for Autonomous Agents*. Proceedings of the Third International Workshop on Agent Theories, Architectures, and Languages (ATAL), 1996.

[FN97] N. Friedman. *Learning belief networks in the presence of missing values and hidden variables*. Proceedings of the Thirteenth International Conference on Machine Learning, 1997.

[FN98] N. Friedman. *The Bayesian Structural EM Algorithm*. Proceedings of the

Fourteenth Conference on Uncertainty in Artificial Intelligence (UAI), 1998.

[FG97] N. Friedman und M. Goldszmidt. *Sequential Update of Bayesian Network Structure*. Proceedings of the Thirteenth Conference on Uncertainty in Artificial Intelligence (UAI), 1997.

[FGG97] N. Friedman, D. Geiger und M. Goldszmidt. *Bayesian Network Classifiers*. Machine Learning Vol. 29, 1997.

[FC89] R. Fung und K. C. Chang. *Weighting and integrating evidence for stochastic simulation*. Proceedings of the Fifth Conference on Uncertainty in Artificial Intelligence, 1989.

[GKS02] T. Gawlik, J. Kellner und D. Seifert. *Effiziente Kundenbindung mit CRM*. Galileo Business, 2002.

[GA04] A. Gerber. *Flexible Kooperation zwischen Autonomen Agenten in Dynamischen Umgebungen*. Dissertation, Universität des Saarlandes, 2004.

[GFKP01] L. Getoor, N. Friedman, D. Koller und A. Pfeffer. *Learning Probabilistic Relational Models*. In S. Dzeroski & N Lavrac: Relational Data Mining. Springer Verlag, 2001.

[GHH01] A. Gustafsson, A. Hermann und F. Huber. *Conjoint Measurement. 2. Auflage*. Heidelberg, 2001.

[HK01] J. Han und M. Kamber. *Data Mining. Concepts and Techniques*. Academic Press, 2001.

[HJA75] J. A. Hartigan. *Clustering Algorithms*. Wiley Verlag, 1975.

[HTF01] T. Hastie, R. Tibshirani und J. Friedman. *The Elements of Statistical Learning. Data Mining, Inference, and Prediction*. Springer Verlag, 2001.

[HD95] D. Heckerman. *A Tutorial on Learning with Bayesian Networks*. Technical Report MSR-TR-95-06, Microsoft Research, 1995.

[HD98] D. Heckerman. *A Tutorial on Learning with Bayesian Networks*. In M. I. Jordan. *Learning in Graphical Models*. MIT Press, 1998.

[HCMRK00] D. Heckerman, D. M. Chickering, C. Meek, R. Rounthwaite und C. Kadie. *Dependency networks for collaborative filtering and data visualization*. Proceedings of the Sixteenth Conference on Uncertainty in Artificial Intelligence (UAI), 2000.

[HGC95] D. Heckerman, D. Geiger und D. M. Chickering. *Learning Bayesian networks: The combination of knowledge and statistical data*. Machine Learning, Vol. 20, 1995.

[HM88] M. Henrion. Propagation of uncertaintyin Bayesian networks by probabilistic logic sampling. Uncertainty in Artificial Intelligence Vol.2, 1988.

[HJ99] J. Hertel. *Warenwirtschaftssysteme. Grundlagen und Konzepte*. *3. Auflage*. Physica-Verlag, 1999.

[HW04] H. Hippner und K. D. Wilde. *IT-Systeme im CRM. Aufbau und Potenziale*. Gabler, 2004.

[HJJ82] J. J. Hopfield. *Neurons with graded response have collective computational properties like those of two-state neurons*. Proceedings of the National Academy of Sciences of the United States of America No. 79, 1982.

[HS69] J. A. Howard und J. N. Sheth. *The Theory of Buyer Behavior*. New York, 1969.

[HR94] R. Hünerberg. *Internationales Marketing (Moderne Industrie)*. Landsberg, 1994.

[ICE94] C. E. Izard. *Die Emotionen des Menschen*. Weinheim, 1994.

[JW00] W. Jager. *Modelling Consumer Behaviour*. Rijksuniversiteit Groningen, Dissertation, Juni 2000.

[JA96] A. Jameson. *Numerical Uncertainty Management in User and Student*

Modeling: An Overview of Systems and Issues. User Modeling and User-Adapted Interaction No. 5, 1996.

[JW01] A. Jameson und F. Wittig. *Leveraging Data About Users in General in the Learning of Individual User Models*. Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence, 2001.

[JW98] N. R. Jennings und M. J. Wooldridge. *Agent Technology: Foundations, Applications and Markets*. Springer Verlag, 1998.

[JNR99] N. R. Jennings. *Agent-based Computing: Promise and Perils*. Proceedings of the 16th International Joint Conference on Artificial Intelligence (IJCAI), 1999.

[JFV96] F. V. Jensen. *An Introduction to Bayesian Networks*. Springer Verlag, 1996.

[JFV01] F. V. Jensen. *Bayesian Networks and Decision Graphs*. Springer Verlag, 2001.

[JJ94] F. V. Jensen und F. Jensen. *Optimal Junction Trees*. Proceedings of the Tenth Conference on Uncertainty and Artificial Intelligence, 1994.

[JN00] N. Jitnah. *Using Mutual Information for Approximate Evaluation of Bayesian Networks*. Dissertation, School of Computer Science and Software Engineering, Monash University, 2000.

[JS01] D. Johnson und S. Sinanovic. *Symmetrizing the Kullback-Leibler Distance*. IEEE Transactions on Information Theory, 2001.

[KT67] T. Kailath. *The divergence and Bhattacharyya distance measures in signal selection*. IEEE Transactions on Communication Technology Vol. 15, 1967.

[KVV04] Y. G. Kim, M. Valtorta und J. Vomlel. *A Prototypical System for Soft Evidential Update*. Applied Intelligence, 2004.

[KP83] J. Kim und J. Pearl. A computational model for causal and diagnostic reasoning in inference systems. Proceedinsg of the Eighth International Joint

Conference on Artificial Intelligence (IJCAI'83), 1983.

[KU94] U. Kjaerulff. *Reduction of computation complexity in Bayesian networks through removal of weak dependencies*. Proceedings of the Tenth Conference on Uncertainty in Artificial Intelligence, 1994.

[KU95] U. Kjaerulff. *dHugin: A Computational System for Dynamic Time-sliced Bayesian Networks*. International Journal of Forecasting, Special Issue on Probabilistic Forecasting No. 11, 1995.

[KA67] A. Koestler. *The Ghost in the Machine*. Hutchinson & Co, London, 1967.

[KR99] R. Köhler. *Kundenorientiertes Rechnungswesen als Vorraussetzung des Kundenbindungsmanagements. Handbuch Kundenbindungsmanagement: Grundlagen, Konzepte, Erfahrungen. 2. Auflage*. Wiesbaden, 1999.

[KP97] D. Koller und A. Pfeffer. *Object-oriented Bayesian Networks*. Proceedings of the Thirteenth Conference on Uncertainty in Artificial Intelligence, 1997.

[KP98] D. Koller und A. Pfeffer. *Probabilistic Frame-based Systems*. Proceedings of the Fifteenth National Conference on Artificial Intelligence (AAAI), 1998.

[KN04] K. B. Korb und A. E. Nicholson. *Bayesian Artificial Intelligence*. Chapman & Hall / CRC, 2004.

[KRW03] W. Kroeber-Riel und P. Weinberg. *Konsumentenverhalten. 8. Auflage*. Vahlen Verlag, 2003.

[KA05] A. Kuklin. *Implementation of Inference Algorithms in Belief Networks*. Master thesis, University of Applied Sciences (HTW), Saarbrücken, 2005.

[KL51] S. Kullback und R. A. Leibler. *On information and sufficiency*. Annals of Mathematical Statistics Vol. 22, 1951.

[LB94] W. Lam und F. Bacchus. *Learning Bayesian Belief Networks. An approach based on the MDL Principle*. Computational Intelligence, Vol. 10:4, 1994.

[LB94b] W. Lam und F. Bacchus. *Using new data to refine a Bayesian network*. Proceedings of the Tenth Conference on Uncertainty in Artificial Intelligence, 1994.

[LIT92] P. Langley, W. Iba und K. Thompson. *An Analysis of Bayesian Classifiers*. Proceedings of the Tenth National Conference on Artifical Intelligence, 1992.

[LW66] G. N. Lance und W. T. Williams. *Computer programs for hierarchical polythetic classification ("similarity analysis")*. Computer Journal Nr. 9, 1966.

[LM97] K. B. Laskey und S. M. Mahoney. *Network Fragments: Representing Knowledge for Constructing Probabilistic Models*. Proceedings of the Thirteenth Conference on Uncertainty in Artificial Intelligence, 1997.

[LJC98] J. C. Loehlin. *Latent Variable Models: An Introduction to Factor, Path, and Structural Analysis*. *Third Edition*. Lawrence Erlbaum, 1998.

[ML98] S. M. Mahoney und K. B. Laskey. *Constructing Situation Specific Belief Networks*. Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence, 1998.

[MF98] F. Mattern. *Mobile Agenten*. Informationstechnik und Technische Informatik Nr. 4. Oldenbourg Verlag, 1998.

[MJ78] J. Mazanec. *Strukturmodelle des Konsumverhaltens*. Wirtschaftsverlag, 1978.

[MJ60] J. McCarthy. *Basic Marketing: A Managerial Approach*. Homewood, 1960.

[MM02] M. McDonald. *Marketing Plans. How to prepare them, how to use them. 5th Edition*. Butterworth/Heinemann, 2002.

[MH92] H. Meffert. *Marketingforschung und Käuferverhalten. 2. Auflage.* Wiesbaden, 1992.

[MRRTT53] N. Metropolis, A. Rosenbluth, M. Rosenbluth, A. Teller und E. Teller. *Equations of state calculations by fast computing machines*. Journal of

Chemical Physics Vol. 21, 1953.

[MM86] M. Minsky. *The Society of Mind*. Simon and Schuster (Touchstone), 1986.

[MPV01] D. C. Montgomery, E. A. Peck und G. G. Vining. *Introduction to Linear Regression Analysis. Third Edition*. John Whiley & Sons, 2001.

[MJP96] J. P. Müller. *The Design of Intelligent Agents. A Layered Approach*. Springer Verlag, 1996.

[MP94] J. P. Müller und M. Pischel. *Modelling Interacting Agents in Dynamic Environments*. Proceedings of the Eleventh European Conference on Artificial Intelligence (ECAI-94), 1994.

[NRE03] R. E. Neapolitan. *Learning Bayesian Networks*. Prentice Hall, 2003.

[NAF74] A. F. Neel. *Handbuch der psychologischen Theorien. 2.Auflage*. München, 1974.

[NF04] F. Neri. *Agent Based Simulation of Information Diffusion in a Virtual Market Place*. Proceedings of the IEEE/WIC/ACM International Conference on Intelligent Agent Technologies (IAT), 2004.

[NS76] A. Newell und H. A. Simon. *Computer Science as Empirical Enquiry: Symbols and Search*. Communications of the ACM No. 19, 1976.

[NBWSSS01] A. Nicholson, T. Boneh, T. Wilkin, K. Stacey, L. Sonenberg und V. Steinle. *A case study in knowledge discovery and elicitation in an intelligent tutoring application*. Proceedings of the Seventeenth Conference on Uncertainty in Artificial Intelligence (UAI), 2001.

[NFM66] F. M. Nicosia. *Consumer Decision Processes*. Prentice Hall, 1966.

[NDH02] R. Nieschlag, E. Dichtl und H. Hörschgen. *Marketing. 19. Auflage*. Berlin, 2002.

[NSCCal03] H. Núnez, M. Sànchez-Marrè, U. Cortés, J. Comas, M. Martinéz, I.

Rodríguez-Roda und M. Poch. *A comparative study on the use of similarity measures in case-based reasoning to improve the classification of environmental system situations*. Environmental Modelling & Software, Elsevier, 2003.

[NSCCRP03] H. Núnez, M. Sànchez-Marrè, U. Cortés, Q. Comas, I. Rodríguez-Roda und M. Poch. *Analysing Similarity Assessment in Feature-Vector Case Representations*. Research Report, LSI Department, Technical University of Catalonia, 2003.

[PJ86] J. Pearl. *Fusion, propagation, and structuring in belief networks*. Artificial Intelligence Vol. 29, 1986.

[PJ87] J. Pearl. *Evidential reasoning using stochastic simulation of causal models*. Artificial Intelligence Vol. 32, 1987.

[PJ88] J. Pearl. *Probabilistic Reasoning in Intelligent Systems. Networks of Plausible Inference*. Morgan Kaufmann Publishers, 1988.

[PKMT99] A. Pfeffer, D. Koller, B. Milch und K. T. Takusagawa: *SPOOK: A System for Probabilistic Object-oriented Knowledge Representation*. Proceedings of the Fifteenth Conference on Uncertainty in Artificial Intelligence, 1999.

[PM98] M. Plach. *Prozesse der Urteilsrevision. Kognitive Modellierung der Verarbeitung unsicheren Wissens*. Deutscher Universitäts-Verlag, 1998.

[PME99] M. E. Porter. *Wettbewerbsstrategie: Methoden zur Analyse von Branchen und Konkurrenten. 10. Auflage*. Campus Verlag, 1999.

[RG92] A. S. Rao und M. P. Georgeff. *An Abstract Architecture for Rational Agents*. Proceedings of the Third International Conference on Principles of Knowledge Representation (KR92), 1992.

[RG95] A. S. Rao und M. P. Georgeff. *BDI-Agents: From Theory to Practice*. Proceedings of the First International Conference on Multiagent Systems, 1995.

[RU91] U. von Reibnitz. *Szenario-Technik. Instrumente für die unternehmerische und persönliche Erfolgsplanung*. Wiesbaden, 1991.

[RJ78] J. Rissanen. *Modeling by shortest data description*. Automatica, Vol. 14, 1978.

[RWR77] R. W. Robinson. *Counting unlabeled acyclic digraphs*. In Combinatorial Mathematics. Springer Verlag, 1977.

[RS99] J. Roure und R. Sangüesa. *Incremental methods for Bayesian network learning*. Technical Report Nr. LSI-99-42-R), Software Department at the Technical University of Catalonia, 1999.

[RSS04] C. Ruß, A. Schwaiger und B. Stahmer. SimMarket – *Grundkonzeption und Anwendungserfahrungen*. In J. Zentes, H. Biesiada und H. Schramm-Klein, „*Performance Leadership im Handel. Zukunft im Handel Band XIX*“, Deutscher Fachverlag, 2004.

[RN95] S. Russell und P. Norvig., *Artificial Intelligence. A Modern Approach*. Prentice Hall, 1995.

[RN03] S. Russell und P. Norvig. *Artificial Intelligence. A Modern Approach. Second Edition*. Prentice Hall, 2003.

[SBD02] L. B. Said, T. Bouron und A. Drogoul. *Agent-based Interaction Analysis of Consumer Behavior*. Proceedings of the First International Joint Conference on Autonomous Agents and Multiagent Systems, 2002.

[SBD01] L. B. Said, T. Bouron und A. Drogoul. *Multi-Agent Based Simulation of Consumer Behaviour: Towards a New Marketing Approach*. Proceedings of the International Congress on Modelling and Simulation (MODSIM), 2001.

[SCRP98] M. Sànchez-Marré, M. Cortés, U. Rodríguez-Roda und M. Poch. *L'Eixample Distance: a new similarity measure for case retrieval*. Proceedings of the First Catalan Conference on Artificial Intelligence (CCIA98), 1998.

[SR98] R. Schäfer. *Benutzermodellierung mit dynamischen Bayes'schen Netzen als*

Grundlage adaptiver Dialogsysteme. Dissertation, Universität des Saarlandes, 1998.

[SD98] D. Schier. *Ein Multiagentenansatz zum Lösen von Fleet-Scheduling-Problemen*. Diplomarbeit, Universität des Saarlandes, 1998.

[SM04] M. Schillo. *Multiagent Robustness: Autonomy vs. Organistion*. Dissertation, Universität des Saarlandes, 2004.

[SSt05] S. Schlicker. *SimAgent – eine verteilte und mobile Multiagentenplattform für den Einsatz in einem Supermarkt-Simulationssystem*. Diplomarbeit, Universität des Saarlandes, 2005

[SP04] P. Schmidt. *Design und Implementierung einer generischen Datenbankarchitektur und geeigneter Schnittstellen zur Bereitstellung der Datengrundlage für adaptive probabilistische Agenten im Rahmen des Projektes SimMarket XT*. Diplomarbeit, Universität des Saarlandes, 2004.

[SD83] D. Schneider. *Marketing als Wirtschaftswissenschaft oder Geburt einer Marketingwissenschaft aus dem Geiste des Unternehmensversagens*. Zeitschrift für betriebswirtschaftliche Forschung, 35. Jg., Nr. 3, 1983.

[SD02] D. Schneider. Multi-Kanal-Management: *Der Kunde im Netzwerk der Handelsunternehmung*. In Ahlert, Becker, Knackstedt und Wunderlich: *Customer Relationship Management im Handel*. Springer, 2002.

[SW04] W. Schneider. *Marketing und Käuferverhalten*. Oldenbourg Verlag, 2004.

[SS03] A. Schwaiger und B. Stahmer. *SimMarket: Multiagent-based Customer Simulation and Decision Support for Category Management*. Proceedings of the First German Conference on Multiagent System Technologies (MATES), 2003.

[SS04b] A. Schwaiger und B. Stahmer. *SimMarket – Agentenbasierte Simulation menschlichen Kaufverhaltens*. Research Report RR-04-03, Deutsches Forschungszentrum für Künstliche Intelligenz (DFKI GmbH), 2004.

[SS05] A. Schwaiger und B. Stahmer. *Probabilistic Holons for Efficient Agent-Based Data Mining and Simulation*. Proceedings of the Second International Conference on Industrial Applications of Holonic and Multi-Agent Systems (HoloMAS 2005). Springer Verlag, 2005.

[SG78] G. Schwarz. *Estimating the dimension of a model*. Annals of Statistics Nr. 6, 1978.

[SP89] R. Shachter und M Poet. *Simulation approaches to general probabilistic inference on belief networks*. Proceedings of the Fifth Workshop on Uncertainty in Artificial Intelligence, 1989.

[SLC03] J. Shen, V. Lesser und N. Carver. *Minimizing Communication Cost in a Distributed Bayesian Network using a Decentralized MDP*. Proceedings of the Second International Joint Conference on Autonomous Agents and MultiAgent Systems (AAMAS 2003), 2003.

[SY93] Y. Shoham. *Agent-oriented Programming*. Artificial Intelligence No. 60, 1993.

[SL90] D. J. Spiegelhalter und S. L. Lauritzen. *Sequential Updating of Conditional Probabilities on Directed Graphical Structures*. Networks No. 20, 1990.

[SB06] B. Stahmer. *SimMarket: Simulation des Abverkaufsverhaltens von Artikeln des Einzelhandels mit probabilistischen Agenten*. Dissertation, Universität der Saarlandes, 2006

[SS04] B. Stahmer und A. Schwaiger. *Holonic Probabilistic Agent Merging Algorithm*. Proceedings of the IEEE/WIC/ACM International Conference on Intelligent Agent Technologies (IAT), 2004.

[TA50] A. Turing. *Computing Machinery and Intelligence*. Mind No. 59, 1950.

[VKV02] M. Valtorta, Y.-G. Kim und J. Vomlel. *Soft Evidential Update for Probabilistic Multiagent Systems*. International Journal of Approximate Reasoning, Vol. 29, 2002.

[WV03] Y. Wang und J. Vassileva. *Bayesian Network Trust Model in Peer-to-Peer Networks*. Proceedings of the Second International Joint Conference on Autonomous Agents and MultiAgent Systems (AAMAS 2003), 2003.

[WP81] P. Weinberg. *Das Entscheidungsverhalten der Konsumenten*. Paderborn, 1981.

[WP97] P. Weinbrenner. *Szenariotechnik*. Bielefeld, 1997.

[WG99] G. Weiss. *Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence*. MIT Press, 1999.

[WL94] M. P. Wellman und C. L. Liu. *State-space abstraction for anytime evaluation of probabilistic networks*. Proceedings of the Tenth Conference on Uncertainty in Artificial Intelligence, 1994.

[WB03] K.-P. Wiedmann und F. Buckler. *Neuronale Netze im Marketing-Management. 2. Auflage*. Gabler, 2003.

[WWHTM01] K.-P. Wiedmann, G. Walsh, T. Henning-Thurau und V.-W. Mitchell. *Consumers's Decision Making Style as a Basis for Market Segmentation*. In: Enhancing Knowlegde Development in Marketing: Proceedings of the 2001 AMA Summer Educators' Conference, Vol. 12, American Marketing Association, 2001.

[WG01] R. M. S. Wilson und C. Gilligan. *Strategic Marketing Management. Second Edition*. Butterworth/Heinemann, 2001.

[WM97] D. R. Wilson und T. R. Martinez. *Improved Heterogeneous Distance Functions*. Journal of Artificial Intelligence Research Vol. 6, 1997.

[WF03] F. Wittig. *Maschinelles Lernen Bayes'scher Netze für benutzeradaptive Systeme*. Akademische Verlagsgesellschaft Aka, 2003.

[WM02] M. Wooldridge. *An Introduction to Multiagent Systems*. John Whiley & Sons, 2002.

[ZLA65] L. A. Zadeh. *Fuzzy Sets*. Information and Control No. 8, 1965.

[ZLA78] L. A. Zadeh. *Fuzzy Sets as a Basis for a Theory of Possibility*. Fuzzy Sets and Systems No.1, 1978.

[ZJM99] J. Zentes, M. Janz und D. Morschett. *New Dimensions in Retail Marketing*. Institut für Handel & Internationales Marketing, 1999.

[ZJ00] J. Zentes. *One To One Marketing.* TrendForum Verlag, 2000.

[ZBSK04] J. Zentes, H. Biesiada und H. Schramm-Klein. *Performance Leadership im Handel*. *Zukunft im Handel Band XIX*. Deutscher Fachverlag, 2004.

[ZRL96] T. Zhang, R. Ramakrishnan und M. Livny. *BIRCH: An Efficient Data Clustering Method for Very Large Databases*. Proceedings of the International Conference on Management of Data, 1996.

[ZZ05] Z. Zheng. *Multi-Agents classification and clustering based on different representations in a supermarket simulation system*. Diplomarbeit, Universität des Saarlandes, 2005.