Universität Leipzig
Institut für Informatik

Data Warehousing und Data Mining

Problemseminar: Sommersemester 1998

Thema 8: *Data Mining. Überblick*

Lew Bessonow, E-Mail: bessonow@aix550.informatik.uni-leipzig.de
Betreuer: Holger Märtens

Inhaltsverzeichnis

1. Einleitung

In der letzten Zeit sind unsere Möglichkeiten in der Generierung und im Sammeln der Information drastisch gestiegen. Im Endeffekt wird oft die ganze Information als Datenbanken gespeichert und durch die verschiedenen Arten von Transaktionen erneuert.
Damit entsteht ein neues Problem:

- wie kann man dann solche Transaktionen gestalten, um die nützliche Information aus den riesigen Datenbanken automatisch herauszubekommen?

Als Folge daraus spielt immer mehr der Begriff *Data Mining* die ausschlaggebende Rolle. Es gibt noch eine Reihe von Begriffen, die eine ähnliche oder die gleiche Bedeutung tragen. Dazu gehören *Knowledge Mining from Databases*, *Knowledge Extraction*, *Data Archaeology*, *Data Dredging*, *Data Analysis* usw.

Es gibt noch einen Ausdruck, den man häufig verwendet, besonders wenn es um das Vorhersageproblem geht, und der sehr nah zum Begriff *Data Mining* liegt. Folglich wird im allgemeinen das Verfahren für das Finden und Interpretieren der Muster von Daten in Datenbanken als das *Knowledge Discovery in Databases* (*KDD*) bezeichnet. Unter einem *KDD*-System versteht man also das System, das in der Lage ist, die Untersuchungsergebnisse zu interpretieren, zu analysieren und für weitere Algorithmen, *KDD*-Systeme bzw. den Endbenutzer bereitzustellen.

Das *Data Mining* ist ein großer Teil des *KDD*, das einen kompletten Satz von Algorithmen, Methoden und Werkzeugen repräsentiert, die solche *KDD*-Muster liefern können. Anders gesagt, *Data Mining* ist eine neue Generation der Werkzeuge und Verfahren, die eine automatisierte und intelligente DB-Analyse ermöglichen. Man kann aber nicht sich ans *Data-Mining*-System wenden, um die Resultate des Systemablaufs zu interpretieren.

Die letzte Bemerkung ist nicht immer durchsichtig und oft einfach formal. Deswegen wird vielleicht bei mehreren Autoren der Unterschied nicht mehr beachtet ([CHY96], [DF95], [HS94]).

Der *Data-Mining*-Prozeß bezieht in sich die ganze Reihe von verschiedenen Verfahren mit ein. Wie diese Verfahren klassifiziert werden und wodurch sie sich voneinander unterscheiden, hängt davon ab, welche Anforderungen ans System gestellt werden, wie das System gestaltet wird und welche Arten von Resultaten als Ergebnis des Systemablaufs erwartet werden.

Einige Verfahren, die unter bestimmten Umständen als auch *Data-Mining*-Verfahren bezeichnet werden, sind uns unter eigenen Namen bereits bekannt (*Machine Learning, Simulated Annealing*, *Neurale Netze* etc.). Es wird aber nur dann von *Data-Mining*-Algorithmen gesprochen, wenn eine oder mehrere Datenbanken als die Forschungsobjekte zugrunde liegen. Beim Unterscheiden der Zugehörigkeit von verschiedenen Verfahren sind aber noch die anderen entsprechenden Randbedingungen und zusätzlichen Forderungen zu beachten.

In diesem Bericht möchte ich aber nicht die einzelnen *Data-Mining*-Verfahren auf ihre Effizienz und Vorgehensweise untersuchen (das ist kein Ziel des Überblicksvortrages), sondern einen Blick auf das gesamte *Data-Mining*-Problem werfen. Die Algorithmen, die im Rahmen dieses Überblicks weiter betrachtet werden, haben keinen Anspruch auf die

Performanz. Sie sollen lediglich als veranschaulichende Beispiele von verschiedenen *Data-Mining*-Ansätzen dienen. Auf die einzelnen Ansätze tiefer einzugehen, ist Aufgabe späterer Vorträge.

Wenn man über ein neues System zu sprechen anfängt, will man in erster Linie immer feststellen, was dieses System weiter leisten soll. Deswegen möchte ich auch in meinem Bericht die Anforderungen ans *Data-Mining*-System als solches zunächst beschreiben (Abschnitt 2).

Im Abschnitt 3 dieses Berichts werden die Klassifizierungsmöglichkeiten eines *Data-Mining*-Systems vorgestellt. Dann wird eine Variante der Klassifikation ausgewählt und im Abschnitt 4, in dem der Schwerpunkt meiner Ausarbeitung liegt, näher angesehen.

Im Abschnitt 5 werden einige Beispiele für *Data-Mining*-Anwendungsgebiete genannt und ihre einzelnen Ziele, die beim Einsatz von *Data-Mining*-Systemen zur Lösung gebracht werden können, aufgezählt.

Im Abschnitt 6 werden einige bereits entwickelte *Data-Mining*-Systeme kurz erwähnt und als Zusammenfassung ein paar Worte über die Zukunft von solchen Systemen gesprochen.

2. Anforderungen ans Data-Mining-System

Um *Data-Mining-V*erfahren effizient durchführen zu können, ist es notwendig festzustellen, welche Art der Charakteristika von dem Entdeckungssystem erwartet wird und welche Forderungen ein solches System erfüllen soll.

Die modernen Datenbanken bestehen in der Regel aus verschiedenen Typen von Daten. Außerdem können auch einzelne Typen von Daten mehrere komplexere Datentypen bilden. Es wird erwartet, daß *Data-Mining*-Algorithmen in der Lage sein sollen, eine komplexere Verarbeitung und Bewertung von Daten unabhängig von Datentypen korrekt durchzuführen.

Es wird auch erwartet, daß *Data-Mining*-Methoden genug effizient und skalierbar sind. Man muß in der Lage sein, die Laufzeit eines solchen Programms einschätzen zu können und mit Hilfe von Parametern zu beschränken. Die Methoden, die einen exponentiellen Grad der Komplexität haben, sind nicht von praktischer Bedeutung.

Die verteilten und heterogenen Datenbanken sollen auch als Quellen für die *Data-Mining*-Algorithmen dienen. Dies bedeutet, daß das *Data-Mining*-System die entsprechende Parallelarbeit und den Distributionsgrad für seine Methoden gewährleisten soll.

Die entsprechende Oberfläche muß so entwickelt werden, daß der Endbenutzer immer eine vollständige Information über die Ergebnisse bekommen kann und Einfluß aufs System nehmen kann.

Die *Data-Mining*-Algorithmen dürfen nicht den Datenschutz gefährden. Es darf nicht vorkommen, daß die Schlußfolgerungen, die die *Data-Mining*-Algorithmen liefern, aufgrund der unzulässigen Informationen erhalten wurden. Diese Anforderung kann im Widerspruch zur Forderung nach Korrektheit der DB-Analyse stehen. Das möchte ich auf einem kleinen Beispiel erklären.

Nehmen wir an, daß jemand die Rechte auf den Zugang nur zu einer Hälfte des Datenbestands besitzt. Er läßt ein *Data-Mining*-Programm laufen, um beispielsweise eine günstige Partitionierung (Abschnitt 4.4) des Datenbestandes zu bekommen. Wenn der Informationsgehalt gerade in anderer Hälfte des Datenbestands liegt, bekommt man höchstwahrscheinlich die Ergebnisse, die keiner richtigen Partitionirung entsprechen.

Solche Ausnahmefälle sollen sehr elegant behandelt werden. So könnte das Programm in unserem Beispiel eine Warnung über die nicht genügenden Personszugriffsrechte zurückgeben.

3. Klassifizierungsmöglichkeiten

Es gibt viele verschiedene Varianten, auf welche Art und Weise die *Data-Mining*-Algorithmen klassifiziert werden können. Dabei werden verschiedene Sichten auf die gleichen Methoden hergestellt, um deren einzelne Eigenschaften abhängig davon, was den Endbenutzer am meisten interessiert, genauer unter die Lupe zu nehmen.

Das Erste, wodurch sich *Data-Mining*-Algorithmen unterscheiden können, ist der Typ der Datenbank, mit dem gearbeitet wird. Im allgemeinen unterscheidet man zwischen relationalen, objekt-orientierten, deduktiven, temporalen, multimedialen, heterogenen usw. Datenbanken, auf die die jeweils entsprechenden *Data-Mining*-Algorithmen gerichtet werden.

Das Zweite, wofür man sich interessieren könnte, sind die Annäherungsstrategien, die Typen von Algorithmen (mathematische, heuristische, genetische etc.) oder die allgemeinen Konzeptarten wie Mustererkennung oder Generalisierung usw., die für die *Data-Mining*-Methoden zugrunde liegen.

Das Dritte, was der Endbenutzer immer klären soll, sind die Datentypen bzw. Regeln, die als die Endergebnisse des Ablaufs der *Data-Mining*-Algorithmen herauskommen sollen. Solche Ergebnisvariationen stellen eine spezifische Art der Klassifikation dar, die gerade für *Data-Mining*-Algorithmen besonders geeignet ist.

Diese dritte Klassifizierungsmöglichkeit scheint dem Endbenutzer besonders klar und deutlich zu sein, weil sie genau seinen Interessen entspricht [CHY96]. Außerdem ist diese Klassifizierungsmethode am besten in den vorhandenen Literaturquellen vorgestellt. Sie wird daher auch in meinem Bericht im Folgenden verwendet.

4. Klassifikation nach den Endergebnissen

Als Ergebnisse (Ziele) von *Data-Mining*-Verfahren können die folgenden Varianten aufgefaßt werden:
- *Gewinnung von Assoziationsregeln (Abschnitt 4.1);*
- *Datengeneralisierung und Datenaggregation (Abschnitt4.2);*
- *Voraussage (Abschnitt 4.3);*
- *Datenpartitionierung (Clustering) (Abschnitt 4.4);*
- *Mustersuche (Pattern-based Similarity Search) (Abschnitt 4.5);*
- *Zugriffsanalyse in verteilten DBS (Abschnitt 4.6).*

Diese Aufteilung wurde in [CHY96] angeboten und umfaßt den größten Teil aller Ergebnisse, die als Resultat der Arbeit von *Data-Mining*-Algorithmen herauskommen können. Es ist aber oft nicht möglich, die einzelnen *Data-Mining*-Algorithmen bzw. Verfahren eindeutig zuzuordnen, weil diese Verfahren an verschiedenen Stellen eingesetzt werden können oder einfach zu allgemein sind.

Ich betrachte beispielsweise im Abschnitt 4.3 das Vorhersageproblem mehr in bezug auf das *Maschinelle Lernen*, weil dieses Verfahren eine gewisse Verallgemeinerung für die Lösung des Vorhersageproblems darstellt. Das schließt aber nicht aus, daß die anderen Verfahren bei der Lösung behilflich sein können. Sie dienen eher als Basis für die weiteren Entwicklungen und liefern keine direkten Antworten wie Datenklassifikation oder Datenregression zurück, welche dann eindeutig interpretiert werden können.

In den folgenden Abschnitten gehe ich die aufgezählten Klassifizierungsmöglichkeiten der Reihe nach durch und versuche klarzumachen, wo der Schwerpunkt bei jeder von ihnen liegt und wie sie sich voneinander unterscheiden lassen.

4.1. Gewinnung von Assoziationsregeln

4.1.1. Hauptmerkmale

Die Assoziationsregeln haben die Aufgabe, Sätze von strengen Regeln, die die Gestalt haben:

$$\text{" } A \wedge B \wedge \ldots \wedge K \;\Rightarrow\; L \wedge M \wedge \ldots \wedge X \text{ ",}$$

wobei A ... X Sätze von relevanten Attributwerten in der Datenbank sind, zu gewinnen.

Um die gewonnenen Assoziationsregeln zu bewerten, führt man verschiedene Kriterien ein. Diese Kriterien werden meistens mit entsprechenden Gewichtsfaktoren versorgt und dann zusammen betrachtet. Auf dem folgenden Beispiel werden zwei solche Kriterien wie *Confidence* und *Support* illustriert. Die *Confidence* bezeichnet die Implikationsstärke der Regel, das *Support* illustriert die Häufigkeit der Regel in der Datenbank.

Es ist meistens erwünscht, nur die Regeln in Betracht zu ziehen, die ausreichend großen *Support* besitzen. Deswegen wird *Support* als Eingabeparameter für die Algorithmen, die Assoziationsregeln erforschen, vordefiniert. Um möglichst allgemeine Assoziationsregeln zu erreichen, versucht man auch, die Regeln zu gewinnen, die die größte Anzahl der Elemente enthalten.

4.1.2. Beispiel für den Apriori-Algorithmus für eine Datenbank aus Verkaufstransaktionen

Das folgende Beispiel wird zum Zweck, das Verfahren der Entdeckung von Assoziationsregeln anschaulicher zu machen, aus [CHY96] komplett übernommen.

Sei $\boldsymbol{Q}$ = { a, b, c, d, e } eine Menge aller möglichen Waren. Gegeben sei eine Datenbank $\boldsymbol{D}$ aus Verkaufstransaktionen $\boldsymbol{T}$ ($\boldsymbol{T\_ID}$ = {100, 200, 300, 400}), wobei jede Transaktion $\boldsymbol{T}$ ein Warenset ist.

D

T_ID	***Waren***
100	a c d
200	b c e
300	a b c e
400	b e

Abbildung 1. *Datenbank **D** als Beispielsziel für den Einsatz des Apriory-Algorithmus*

Seien ***X,Y*** zwei beliebige Warenmengen. Wir bezeichnen Regeln von der Art ***X => Y*** als Assoziationsregeln für die Datenbank ***D*** (Abbildung 1) genau dann, wenn ***X*** ein Bestandteil einer Transaktion ist und ***(X ∩ Y) = Ø*** ist. Zum Beispiel läßt sich aus Transaktion 300 die Regel { a, c } => { b, e } gewinnen.

Support ?

Waren	***Support***
a	0.5
b	0.75
c	0.75
d	0.25
e	0.75

Waren	***Support***
a	0.5
b	0.75
c	0.75
e	0.75

Scan ***D***

Waren
{ a, b }
{ a, c }
{ a, e }
{ b, c }
{ b, e }
{ c, e }

Waren	***Support***
{ a, b }	0.25
{ a, c }	0.5
{ a, e }	0.25
{ b, c }	0.5
{ b, e }	0.75
{ c, e }	0.5

Support ?

Waren	***Support***
{ a, b }	0.5
{ b, c }	0.5
{ b, e }	0.75
{ c, e }	0.5

Scan ***D***

Waren
{ b, c, e }

Waren	***Support***
{ b, c, e }	0.5

Support ?

Waren	***Support***
{ b, c, e }	0.5

Abbildung 2. *Vorgehensweise beim Apriori-Algorithmus*

Die Datenbank ***D*** enthält die Regel ***X => Y*** mit *Confidence* ***C*** $\in$ [0 , 1] genau dann, wenn ein Anteil von ***C*** aller Transaktionen in ***D***, die ***X*** enthalten, auch ***Y*** enthält. Die Datenbank ***D*** enthält die Regel ***X => Y*** mit *Support* ***S*** $\in$ [0 , 1] genau dann, wenn ein Anteil von ***S*** aller Transaktionen in ***D*** ***(X*** $\cup$ ***Y)*** enthält.

Bei jedem Schritt bildet der *Apriori*-Algorithmus ein Set von Kandidaten, wo die Trefferzahl für jeden Kandidaten in der Datenbank gezählt und mit dem vorgegebenen *Support*-Wert verglichen wird. Am Anfang werden alle einzelnen Waren als Kandidaten angesehen. In unserem Beispiel wird angenommen, daß ein *Support*-Wert ***S ≥ 0.5*** angestrebt wird (zwei Regeln aus fünf müssen die Annahme unterstützen).

Beim nächsten Schritt bekommt man ein Set von zweielementigen Kandidaten (Abbildung 2), indem man alle möglichen Paare von Waren aus der erhaltenen Tabelle zusammenbaut (die Reihenfolge spielt dabei keine Rolle). Dann wird der *Support*-Wert für jeden Kandidaten im neuen Set ausgerechnet und mit dem vorgegebenen Wert verglichen. Anschließend werden dreielementige Kandidaten gebildet usw. Das Verfahren terminiert, wenn keine weiteren Möglichkeiten bestehen, ein neues größeres Set von Kandidaten aufzubauen.

Nachdem man das größte Set von Kandidaten (das Set { b, c, e } in unsrem Beispiel) entdeckt hat, werden gewünschte Assoziationsregeln als Kombination der Elemente des Sets zusammengefaßt. Die entsprechenden *Confidence*-Werte für jede Regel werden berechnet und mit dem Eingabewert verglichen.

e ^ c => b, ***C = 1***
b => c ^ e, ***C = 0.66***
c => b ^ e, ***C = 0.66***
usw.

4.1.3. Probleme bei der Entdeckung von Assoziationsregeln

Solche Werte wie *Support* und *Confidence* sind nicht die einzigen Kriterien, die beim Auswählen von Assoziationsregeln helfen können. Ein weiteres Beispiel in [CHY96] zeigt, wie man aufgrund falsch ausgewählter Werten zu den falschen Endschlüssen kommen kann.

Es wird angenommen, daß in einer Schule 5,000 Schüler sind. Man befragte sie nach ihrem Verhalten morgens und hat gemerkt, daß 3,000 Schüler Basketball spielen, 3,750 Schüler Getreideflocke essen und 2,000 beides machen (spielen Basketball und essen Getreideflocken). Wenn ein *Data-Mining*-Programm mit minimalem *Support = 0,4* (2,000 Schüler) und minimaler *Confidence = 0,6* eingesetzt wird, kommtdie folgende Assoziationsregel heraus:

"spielt Basketball => ißt Getreideflocke".

Diese erreicht den geforderten *Support* und hat einen *Confidence*-Wert von *2000/3000 = 0.66*. Jedoch ist die erhaltene Assoziationsregel irreführend, weil die Schüler, die Getreideflocken essen, schon 75 % aller Schüler in der Schule umfassen. Tatsächlich sind Basketball und Flockenverzehr negativ korreliert.

Aus dem Beispiel wird klar, daß auch *Support* und *Confidence* als Kriterien nutzlos sein können, wenn ihre Werte nicht richtig eingeschätzt wurden oder wenn sie zur Entdeckung von Regeln dienen, die offensichtlich vor dem Einsatz des *Data-Mining*-Programms absolut klar und uninteressant waren. Man muß sich immer überlegen, welche Kriterien und mit welchen Gewichtsfaktoren genommen werden sollen, um die Korrektheit des Resultats zu gewährleisten.

Es gibt noch ein paar Punkte, auf die man bei der Entwicklung von Assoziationsregeln seine Aufmerksamkeit richten soll. *Data-Mining*-Algorithmen werden in der Regel auf großen Datenbanken verwendet. Deswegen wurden schon andere Algorithmen entwickelt, die unter bestimmten Umständen mit einer geringen Anzahl von *Scan-Operationen* auskommen können.

Für die *Update-Operationen* müssen entsprechende Algorithmen verwendet werden, die anhand der neuen Daten die schon vorhandenen Assoziationsregeln anpassen können, ohne diese Regeln neu zu entwickeln.

4.2. Datengeneralisierung und Datenaggregation

Die Daten und Objekte in einer Datenbank enthalten häufig sehr detaillierte Information auf niedrigen konzeptuellen Ebenen. Manchmal wäre es sehr nützlich, die relevanten Datencharakteristika in einer Datenbank zu entdecken. Da benutzt man die *Data-Mining*-Verfahren, die eine Generalisierung und Aggregation von Daten ermöglichen. Dabei werden die materialisierten Sichten (das Thema 4 unseres Seminars) auf verschiedenen abstrakten Ebenen definiert, die eine gröbere Darstellung der Information anbieten. Die *Data-Mining*-Verfahren liefern die Definitionen der Sichten, auf denen weiter die *Data-Warehousing*-Verarbeitungstechniken eingesetzt werden können.

Bei der Generalisierung unterscheidet man zwischen zwei Hauptmethoden. Die erste heißt *Data Cube Approach* und hat den Zweck, alle am häufigsten an die Datenbank gestellten Anfragen im voraus auszurechnen und sich als materialisierte Sichten zu merken. Ein solcher Würfel muß mit zusätzlichen Mechanismen versorgt werden, die automatisch die Änderungen in der Datenbank den entsprechenden Sichten zuordnen. Im Rahmen unseres Seminars haben wir auch schon den Vortrag über die Anfrageverarbeitung (das Thema 6) gehört.

Die zweite dynamische Hauptmethode nennt man *Attribute-Oriented Induction Approach.* Die Sichten werden in dem Fall nicht im voraus gebildet, sondern werden durch immer wieder an die Datenbank gestellte Anfragen gewonnen. Die ersten materialisierten Sichten werden beim Ankommen der ersten Anfrage aus der Daten, die bei der Anfragestellung betroffen worden sind, aufgebaut. Diese Sichten werden dann im Laufe der weiteren Anfrageverarbeitung erweitert bzw. neue Sichten werden definiert. In dem Fall besteht die Aufgabe eines *Data-Mining*-Systems gerade darin, daß versucht wird, die richtige Entscheidung zu treffen, wie die bereits gespeicherten Sichten direkt ausgewertet werden können, ohne sie neu zu berechnen, oder wie neue materialisierte Sichten aufgebaut werden sollen.

Die Probleme, die bei der *Attribute-Oriented Induction Approach*-Methode auftreten, sind oft nicht leicht zu lösen. Wenn die Anzahl der Sichten in der Datenbank ziemlich schnell wächst, wird die Testsuche nach Sichtdefinition nicht mehr effizient. Deswegen beschränkt man sich häufig heutzutage auf die *Data Cube Approach*-Methode, wo alle Sichten vorgegeben werden.

In beiden Fällen kann man die gewünschten Daten erreichen, wenn man die üblichen bekannten Vorgehensweisen *"roll-up"* oder *"drill-down"* unternimmt. Die generalisierten Beziehungen können als Abbildungen, Karten, Diagramme veranschaulicht werden.

4.3. Voraussage

4.3.1. Datenklassifikation und Datenregression

Es wird zwischen zwei möglichen Arten der *Data-Mining*-Voraussage unterschieden. Beide Verfahren bestehen darin, daß versucht wird, aufgrund der vorhandenen Attributwerte das weitere Verhalten der einzelnen Objekte in der Datenbank zu erraten.

Die Datenklassifikation liefert uns nur Antworten der Art : "Ja" oder "Nein", "richtig" oder "falsch". Das Problem kann auch für mehrere Varianten generalisiert werden, aber es kann nicht vorkommen, daß die Anfrage mehrdeutig beantwortet wird.

Nehmen wir an, daß ein Autohändler sich die Frage stellt, an welche potentiellen Kunden sein neuer Autokatalog direkt verschickt werden soll. Das ist ein Beispiel für die typische Anfrage bei der Datenklassifikation. Man geht die Reihe nach alle Kunden durch und fragt sich ständig, ob einem konkreten Kunden der Katalog zugesendet wird, abhängig von seinen Eigenschaften (z.B. Alter, Beruf, früheres Konsumverhalten). Hierbei wird angenommen, daß Kunden mit ähnlichen Eigenschaften für die gleichen Angebote in Frage kommen.

Die Korrektheit der Antwort wird leicht berechnet, indem man eine Fehlerfunktion

$$\boxed{accuracy = 1 - errors / cases}$$

einsetzt.

Die Datenregression (auch unter dem Namen *Funktionsapproximation* bekannt) rechnet hingegen eine bestimmte Zahl als Antwort aus. Dieses Verfahren ist viel komplizierter als die Datenklassifikation, aber es gibt eine Reihe von Problemen wie Gewinn- oder Verlustvoraussage, wo es einfach unentbehrlich ist.

Nehmen wir nochmals unser Beispiel mit dem Autohändler. Diesmal will er nur ein Auto verkaufen und hat schon einige potentielle Kunden in Aussicht. Er versucht, aus den Eigenschaften der Kunden den erzielbaren Preis zu berechnen. Der Autohändler befaßt sich schon mit einem einfachen Datenregressionproblem.

In beiden Fällen wird in der Regel nicht direkt mit der ganzen großen Datenbank gearbeitet, sondern es wird zunächst mit Hilfe von den schon erforschten und geprüften Mustern ein Modell aufgebaut und aufgrund dieses Modells die zukünftige Lösung angestrebt. Dieses Modell heißt *Training Set* und stellt eine Beschreibung von Daten dar, die auf die relevanten

Umwelteigenschaften reduziert ist. In unseren Beispielen könnte als *Training Set* für den Autohändler eine Liste von potentiellen Kunden mit ihren sämtlichen finanziellen und persönlichen Eigenschaften betrachtet werden, denn er trifft alle Entscheidungen aufgrund dieser Liste.

Wenn ein solches Schema vorhanden ist, taucht sofort die ganze Reihe von Begriffen wie *Machine Learning, Simulated Annealing, Neuronale Netze* als Mechanismen, die beim Aufbau eines solches Modells behilflich sein könnten, immer wieder auf.

4.3.2. Maschinelles Lernen

Man fängt schon an, über *Machine Learning* als *Data-Mining*-Verfahren zu sprechen, wenn es gelingt, mit Hilfe von *Machine Learning* einen Trend beim Gestalten von Daten innerhalb einer Datenbank zu entdecken.

Man unterscheidet zwischen überwachtem und unüberwachtem *Machine Learning*. Bei dem überwachten *Machine Learning* (*Supervised Learning* oder *Learning from Examples*) werden die Musterklassen von dem Benutzer des Systems explizit definiert, die weiterhin als Beispiele für das System dienen sollen. Dabei wird das System gezwungen, die allgemeinen Eigenschaften für jede Klasse herauszufinden.

Bei dem unüberwachten *Machine Learning* (*Unsupervised Learning* oder *Learning from Observation and Discovery*) erzeugt das System aufgrund der allgemeinen Eigenschaften des Forschungsobjekts seine eigenen Klassen.

Kehren wir noch mal zum Autohändler-Beispiel zurück. Dort hatten wir die Liste von potentiellen Kunden mit ihren sämtlichen finanziellen und persönlichen Eigenschaften schon vorhanden. In der Liste standen schon alle Kunden, die für ein *Machine-Learning*-System, dessen Aufgabe nach wie vor die Kundenauswahl bleibt, als Objekte auftreten können. Ein solches System gehört offensichtlich zu überwachtem *Machine Learning*. Um zu unüberwachtem *Machine Learning* überzugehen, ist es ausreichend, wenn man sich vorstellt, daß der Autohändler keinen festen Kundenstamm besitzt und versucht das Verarbeitungssystem zu zwingen, eine solche Liste potentieller Interessenten aufgrund einer elektronischen Sammlung von allgemeinen Personendaten herzustellen.

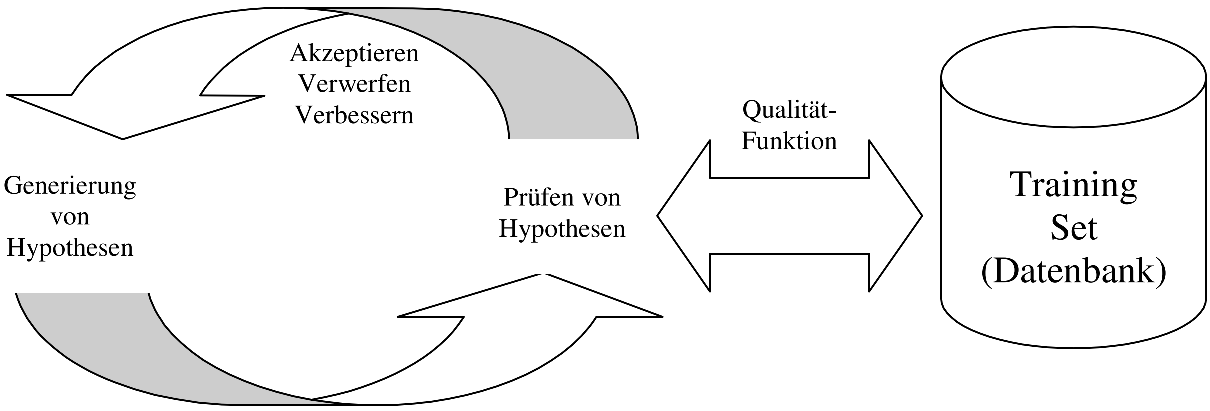

Abbildung 3. *Interaktives Suchverfahren in einem Machine-Learning-System*

Die Suche nach der Antwort in einem beliebigen *Machine-Learning*-System ist immer ein interaktives Verfahren. Das System entwirf verschiedene Hypothesen, vergleicht sie mit dem *Training Set,* bewertet sie mit einer Qualitätsfunktion und nimmt entsprechende Änderungen an den Hypothesen vor (Abbildung 3).

Unter dem *Training Set* versteht man wie bei anderen Verfahren eine Beschreibung von Daten, die auf die relevanten Umwelteigenschaften reduziert ist. Ein *Machine-Learning*-System wird dann als *Data-Mining*-System bezeichnet, wenn sein *Training Set* mit einer echten Datenbank übereinstimmt.

Man ist leider nicht immer in der Lage, *Machine-Learning*-Systeme für die Lösung eines *Data-Mining*-Problems direkt einzusetzen. In der Regel sind die reinen *Machine-Learning*-Systeme nicht besonders groß.

Die *Data-Mining*-Systeme haben dagegen eine Vielfalt von verschiedenen Attributen und die Größe ihrer Datenbank soll praktisch unbeschränkt bleiben. Außerdem sind bei Ihnen alle zusätzlichen Integritätsbedingungen zu beachten.

4.4. Datenpartitionierung (*Clustering*)

4.4.1. Clustering-Problem

Die *Clustering*-Analyse hilft bei der Unterteilung von großen Datenbanken in kleinere, welche danach leichter skaliert und erforscht werden können. In Hinsicht auf das *Data-Mining*-Problem werden durch *Clustering*-Algorithmen die Regionen entdeckt, wo gemäß eines oder mehrerer gegebenen Kriterien die Anzahl an bedeutenden Werten am dichtesten liegt (Abbildung 4).

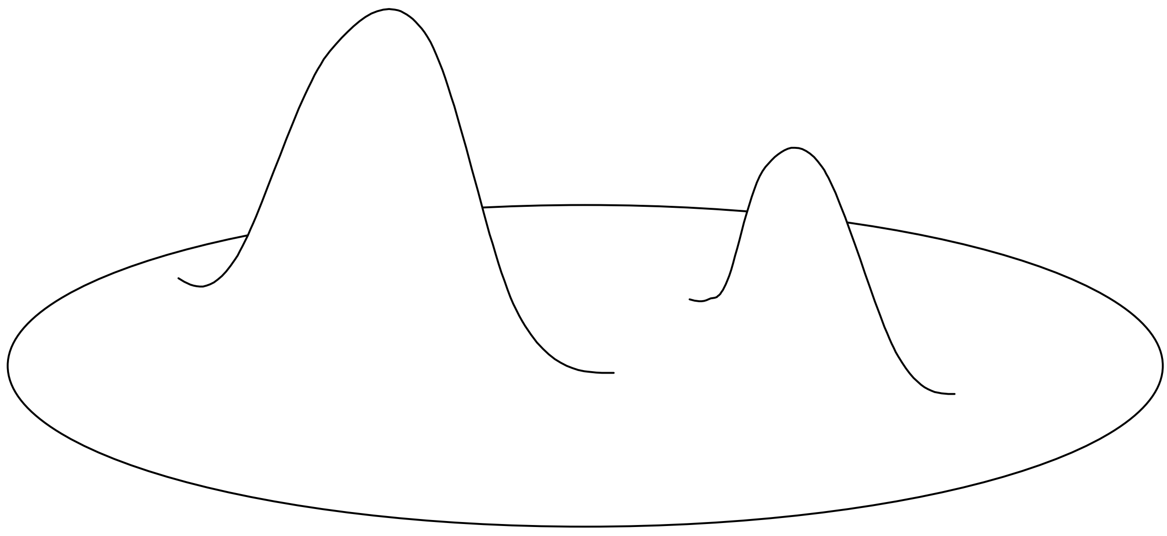

Abbildung 4. *Regionen von bedeutenden Werten als Erkennungsziel für Clustering-Algorithmen*

Es ist auch möglich, eine günstige Partitionierung der Datenbank (des Modells der Datenbank) durch *Clustering*-Algorithmen zu finden, die danach als Basis für die weitere Lösung des Vorhersageproblems dienen kann. Deswegen verwendet man auch in den einzelnen Literaturstellen, die sich dem *Data-Mining*-Problem, besonders in bezug auf Maschinelles Lernen, widmen, den Begriff *Unsupervised Classifikation*[1] anstatt *Clustering*.

4.4.2. Simulated Annealing

Das *Simulated Annealing* ist ein Zufallsverfahren, das eine günstige Unterteilung einer Objektmenge ermöglicht. Die Gruppe von Objektmengen, die als Ergebnisse einer solchen Unterteilung erhalten wurden, werden als *Partitionen* oder *Cluster* bezeichnet.

Bei diesem Verfahren wird zunächst eine willkürliche Partitionierung bestimmt und anhand einer Kostenfunktion ausgewertet. An dieser ersten Lösung werden nun kleine Änderungen vorgenommen. Für die entstandene neue Partitionierung wird die Kostenfunktion neu berechnet. Ist das neue *Clustering* besser als das bisherige, wird es akzeptiert und die alte Lösung verworfen. Mit einer gewissen Wahrscheinlichkeit wird aber eine neue Lösung auch dann akzeptiert, wenn sie eine Verschlechterung darstellt. Die Wahrscheinlichkeit für das Akzeptieren von Verschlechterungen nimmt jedoch im Laufe des Verfahrens ab (die *Temperatur* wird langsam abgekühlt). Dieser Vorgang wird so lange wiederholt, bis ein Endkriterium erfüllt ist oder eine Lösung (*ein gutes Clustering*) erreicht wurde.

Das *Simulated Annealing* kann beispielsweise beim Entwurf eines neuen Mehrrechnersystems eingesetzt werden. Es wird eine günstige Partitionierung von Rechnern angestrebt, bei der die durchschnittliche Ausführungszeit eines verteilten Algorithmus am niedrigsten liegt (Kostenfunktion). Die Eigenschaften von einzelnen Rechner wie ihre CPU-Kapazität oder die Kommunikationszeit mit den Nachbarn werden zunächst willkürlich festgelegt und dann durch jede neue Aufteilung und Umtausch von Rechnern im Sinne der Kostenfunktion immer wieder verbessert.

Wenn anstatt einer Objektmenge eine Datenbank genommen wird, kann das *Simulated Annealing* als *Data-Mining*-Verfahren aufgefaßt werden. Dann sind die Sätze von Attributen oder die Gruppen von Assoziationsregeln[2] unter den *Clustern* zu meinen. In dem ersten Fall kann z.B. das Zusammenhalten aller wichtigen voneinander abhängigen Attribute angestrebt werden. Im zweiten wird dagegen ausgewählt, welche Gruppe von Assoziationsregeln besonders für die Datenbank geeignet ist.

Der *Simulated-Annealing*-Algorithmus kann auch unter Umständen als ein echtes Stück eines *Machine-Learning*-Systems betrachtet werden. Daraus folgt, daß alle Probleme, die beim Einsatz eines *Machine-Learning*-Systems anstatt eines *Data-Mining*-System auftreten, auch für ihn gelten. Es kann schwierig und sogar unmöglich werden, bei häufigen und umfangreichen *Update*-Operationen das *Simulated Annealing* anzuwenden.

---

[1] Als Klassen werden da die Kollektionen von Objekten definiert, dessen innere Änlichkeit und äußere Differenz am größten sind.
[2] Auch die anderen *Data-Mining*-Ansätze können mit Simulated Annealing kombiniert werden.

4.5. Mustersuche (*Pattern-based Similarity Search*)

In den zahlreichen temporalen Datenbanken befindet sich eine große Portion von gespeicherten Daten in Rechnern. Als Beispiele für solche Datenbanktypen können medizinische oder wirtschaftliche Datenbanken betrachtet werden. Die Suche nach Ähnlichkeiten in temporalen Datenbanken ist sehr wichtig in vielen *Data-Mining*-Operationen, um das Risiko, die Kausalität, die Trends, die mit einem bestimmten Muster assoziiert werden, voraussagen zu können. Zu den typischen Anfragen, die an die temporalen Datenbanken gestellt werden können, gehören beispielsweise das Identifizieren von Firmen mit ähnlichem Verhalten des Wachstums oder von Waren mit ähnlichen Verkaufsmustern. In solchen Anfragen wird nicht die exakte Übereinstimmung der Objekten gefordert, sondern ihr vergleichbares Verhalten.

Es wird zwischen zwei möglichen Varianten der Anfragestellung bei solchen Suchverfahren unterschieden. Die erste hat die Aufgabe, innerhalb einer Menge die Zielobjekte herauszufinden, die laut einem von dem Endbenutzer vorgegebenen Kriterium ähnlich dem Musterobjekt sind. Die Anfragen des zweiten Typs suchen nach Paaren von Objekten, die einander nah im Sinne des vordefinierten Kriteriums liegen.

In beiden Fällen können die Übereinstimmungen der Objekte entweder aufgrund des *Whole Matching* (die Objekte werden als Ganzes verglichen) oder des *Subsequence Matching* (das Musterobjekt ist kleiner als die Objekte in der Datenbank, und es wird nur mit einem Teil des Objekts in der Datenbank verglichen) bewertet werden.

Es gibt verschiedene Möglichkeiten, auf welche Art und Weise Vergleichsmethoden aufgeteilt werden können. Die erste ist z.B. die Auswahl des Ähnlichkeitsmaßes. Die zweite Möglichkeit besteht im Unterscheiden zwischen Domänentypen (zeitabhängige oder zeitunabhängige transformierte Domäne), auf denen die Vergleichsoperationen durchgeführt werden. Man kann auch zwischen den Algorithmen unterscheiden, die das Umrechnen des Maßstabs, das Verwenden von Übersetzungsmechanismen und Unterketten variabler Längen bei Vergleichen zulassen.

In der letzten Zeit wurden ausschlaggebende Fortschritte bei der Entwicklung der Algorithmen, die Vergleiche von Objekten in temporalen Datenbanken durchführen sollen, erreicht. Aus Effizienzgründen sind diejenigen *Data-Mining*-Algorithmen zu bevorzugen, die mit einer geringen Anzahl von Vergleichen auskommen, wobei die Korrektheit der Antwort gewährleistet bleiben muß.

4.6. Zugriffsanalyse in verteilten Systemen

In verteilten Systemen werden einzelne Objekte zusammengebunden, um interaktive Zugriffe zu ermöglichen. Als Beispiel für ein solches System kann das *World Wide Web* genannt werden. Das Verständnis der Struktur der Zugriffspfade von Endbenutzern in einem solchen System kann nicht nur bei dem Aufbau dieses System behilflich sein, sondern auch bei seiner Ausbeutung (im *WWW*-Beispiel könnte man sich beispielsweise fragen, wo die wichtigsten Werbeseiten plaziert werden sollten, damit die maximale Anzahl der Endbenutzer sie besuchen kann). Der Prozeß des Speicherns solcher Pfade, um sie danach auf die entstandenen Interessen der Endbenutzer zu testen, ist unter dem Namen *Mining Path Traversal Patterns* (*MPTP*) bekannt.

Solche *MPTP*-Systemanalyse ist leider noch nicht sehr entwickelt und beschränkt sich meistens auf die aufgrund der Häufigkeitsrate der zu besuchenden Seiten gezogenen Schlußfolgerungen.

Ein Algorithmus wurde vorgeschlagen, der eine Version der Lösung implementiert. Im ersten Schritt wird die Reihe aller Referenzen auf die aus dem Anfangspunkt erreichten Seiten in einer Datenbank gespeichert.

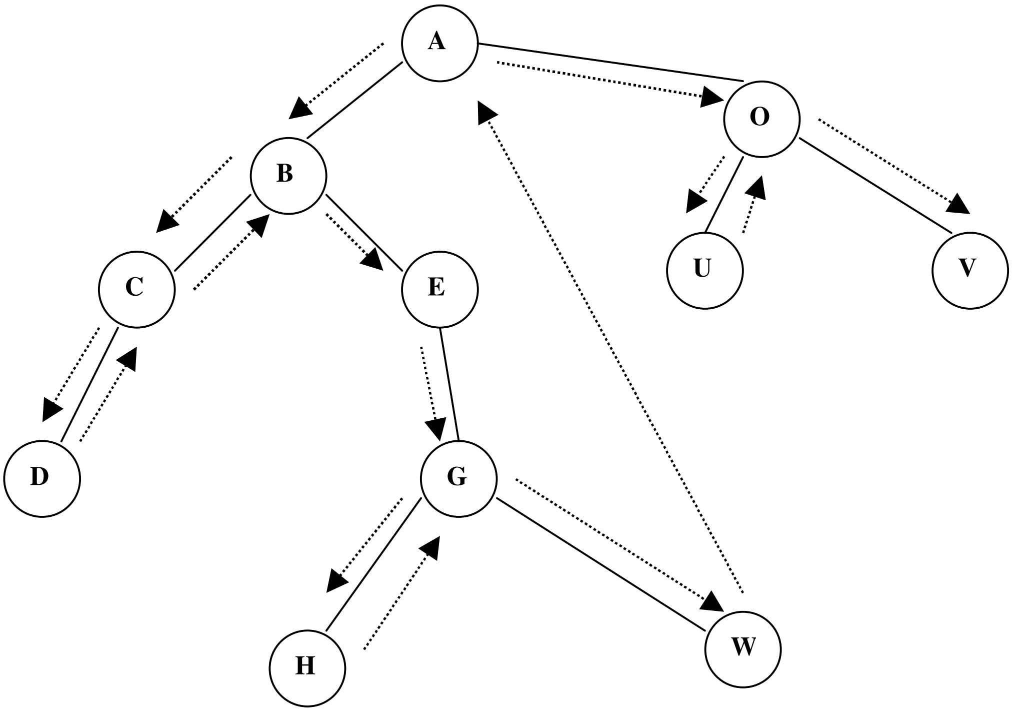

Abbildung 5. *Illustrierendes Beispiel für eine Zugriffsfolge (Traversal Patterns)*

Beim zweiten Schritt werden die Folgen herausgezogen, die die Referenzen von dem Ausgangspunkt aus bis auf die Referenzen der letzten Seiten enthalten. Als drittes werden alle möglichen Unterfolgen nach ihrer Häufigkeit beim Benutzen sortiert und die, die am meisten vorkamen, als Ergebnis genommen.

Nehmen wir an, daß der in der Abbildung 5 gezeigte Baum {A,B,C,D,C,B,E,G,H,G, W,A,O,U,O,V} einen solchen Referenzenpfad darstellt. Im zweiten Schritt entstehen im Beispiel die Folgen: {A,B,C,D}, {A,B,E,G,H}, {A,B,E,G,W}, {A,O,U}, {A,O,V}. Nehmen wir an, daß {AB, BE, AD, CG, GH, BG} die Unterfolgen der Länge 2 und {ABE, CGH} die Unterfolgen der Länge 3 sind, die wegen ihrer Häufigkeit herauskristallisiert wurden. Da die Folgen {AB, BE, CG, GH} Unterpfade der Folgen {ABE, CGH} sind, werden sie weggelassen. Die verbleibenden Folgen {ABE, CGH, AD, BG} werden als die am häufigsten benutzten Muster für die inneren Ziele weiter betrachtet.

5. Anwendungsgebiete für *Data-Mining*-Verfahren

Das *Data Mining* findet praktische Anwendung in vielen Branchen von Industrie und Handel. Auf einige von ihnen, in der Chemie und Pharmakologie, in der Finanzsphäre und im Einzelhandel, werde ich tiefer eingehen.

5.1. Chemie- und Pharmakologie-Industrie

In der Chemie- und Pharmakologie-Industrie nutzt man *Data Mining* als Teil der Lösung von Problemen der Informationsverwaltung. Die Hersteller bekommen die Information über die Produkte entweder als Ergebnisse der Verkaufsvolumen der einzelnen Betriebe oder als Publikationen und Notationen, die herausgegeben werden. So fanden z.B. die Hersteller wie *Sandoz* und *Pfizer* es sehr nützlich, eine zusätzliche Datenbank, in der Information bezüglich der biologischen Aspekten von Medikamenten gesammelt wird, aufzubauen.

Entsprechende *Data-Mining*-Algorithmen wurden entwickelt, die diese spezifische Information im Zusammenhang mit anderen Informationsquellen wie z.B. kommerziellen Datenbanken oder Datenbanken mit den Ergebnissen von Universitätsforschungen bewerteten und verarbeiteten. Dabei verfolgte man die ganze Reihe von Zielen wie das Erarbeiten neuer effektiver Mischungen und Präparate oder das Erhalten des maximalen Profits.

Die Datenoperationen bei Data Mining in der Chemie- und Pharmakologie werden auf die folgenden unterteilt:
- Klassifikation von Molekülen nach Ähnlichkeit;
- Aussuchen aller Moleküle, die einem bestimmten Muster entsprechen;
- Aussuchen aller Moleküle, die andere Zielmoleküle binden können;
- Voraussage des Resultats eines Tests aufgrund der gespeicherten Daten;
- u. v. a.

5.2 Einzelhandel

In diesem Gebiet sind *Data-Mining*-Systeme besonders fortgeschritten, und nicht alle Einzelhändler können sich leisten, ein eigenes *Data-Mining*-System zu entwickeln. Sie nutzen aber gern Produkte der Forschungszentren und Universitäten oder anderer großer Unternehmen.

Um zu verstehen, welche Rolle das *Data Mining* im Einzelhandel bei den großen Unternehmen spielt, muß man sich zunächst vorstellen, wie groß der Warenumsatz bei ihnen ist. Solche Firmen wie *Wal-Mart* oder *K-Mart* in den USA oder *Marks & Spencer* in England verkaufen bis zu zweihunderttausend Warenartikel in mehreren hundert Verkaufsstellen, wo das Kundenverhalten und Einkommen sehr unterschiedlich sind.

Es gibt viele verschiedene Faktoren, die für den Warenverkauf sehr sensibel sind. Zu ihnen gehören sowohl die Größen der Bereitstellungsflächen in den Lagern, ihre Lage und Anfahrtszugänge, Jahreszeit als auch die Stärke der Werbekampagne oder der Einfluß der Konkurrenten. Außerdem ist häufig eine innere Korrelation zwischen den Produkten vorhanden, die solche Abhängigkeiten noch kompliziert (so z.B. zwischen Kleidern und

Accessoires). Im Endeffekt muß die Kurve von Parametern zu dem maximalen Profit bei der Verkaufsstrategie optimiert werden.

Die bei Data Mining in dem Einzelhandel verfolgten Ziele sind:
- Verkaufsvorhersagen;
- Optimierung von Marketingstrategien.

5.3 Automatische Verarbeitung von Fernerkundungsdaten

Täglich wird eine große Anzahl von Fernerkundungsdaten aus entfernten Quellen wie Satelliten, Flugzeugen, Radaren usw. erhaltenen Daten empfangen und verarbeitet.

Militär und Wissenschaft haben großes Interesse an solchen Daten. So können z.B. das Verhalten feindlicher Objekte im Militär oder Anomalien der Atmosphäre und der Erdoberfläche erforscht und kontrolliert werden.

Solche Daten sind in der Regel sehr bequem durch *Data-Mining*-Verfahren zu erforschen. Die Menge von Daten ist reichlich genug, und die Mehrheit der Phänomene ist stabil und identifizierbar.

5.4. Finanzielle Anwendung

Die Benutzung von Data-Mining-Algorithmen in der finanziellen Welt kann grob in zwei Hauptaspekte unterteilt werden:
- Die Untersuchung der Tätigkeit der Banken als finanzielle Vermittlungsstellen (Die Ziele sind ähnlich wie die von den Einzelhändlern);
- Finanzielle Voraussage.

Es ist bis jetzt noch nicht klar, ob die Serien der finanziellen Operationen überhaupt vorhersagbar sind. Man unternahm schon eine Reihe von Versuchen in diesem Gebiet. Das Wissen aus den Bereichen Statistik und neuronale Netzwerke wurde eingesetzt. Es existiert schon eine ganze Menge von Publikationen[3], die das Problem aus verschiedenen Sichten ansehen.

6. Zusammenfassung

In der Arbeit wurde versucht, einen kurzen Überblick über das *Data-Mining*-Problem mit dem Schwerpunkt auf die Verfahren, die in der Lage sind, eine mächtige Datenbankanalyse durchzuführen, zu schaffen. Die wichtigste Klassifizierung wurde vorgestellt und die kennzeichnenden Beispiele vorgeführt. Ein bewertender Vergleich der verschiedenen Ansätze ist kaum möglich, da sie für unterschiedliche Einsatzgebiete und Benutzeranforderungen entwickelt wurden. Statt dessen möchte ich noch einige der in der letzten Zeit entwickelten Werkzeuge und ihre Einsatzgebiete kurz aufzählen.

[3] Dr. Paul Refenes, der Gründer des "*NeuroForecasting*" Klubs in London, bietet in seinem *Buch "Neural Network Applicatons in Financial Asset Management*" einen grobe Überblick von Problemen des *Data Mining* in bezug auf die finanziellen Ansätze an.

1. *Quest,* ein *Data-Mining*-System, entwickelt von IBM, befaßt sich mit der Entdeckung verschiedener Assoziationsregeln, Mustersuche und Klassifizierungsproblemen;
2. *KEFIR,*ein *KDD*-System für die automatische Analyse von Gesundheitszustandsdaten, entwickelt im GTE Labor von *Piatetsky-Shapiro*;
3. *SCICAT,* ein *KDD*-System für die automatische Analyse der Himmelobjekte aus ihren Konturen, entwickelt im *Jet Propulsion Laboratory*;
4. *DBMiner,* ein relationales *Data-Mining*-System, entwickelt an der Simon Fraser University, befaßt sich mit der Entdeckung von Assoziationsregeln und Klassifizierungsregeln auf verschieden abstrakten Ebenen;
5. *KnowledgeMiner,* ein *Data-Mining*-System von *Shen*, integriert *Data Mining* mit deduktiven Datenbanken und benutzt Meta-Regeln, um diesen Prozeß zu unterstützen;

Das *Data Mining* ist ein relativ junges Gebiet, welches sich sehr schnell verbreitet. Viele *Data-Warehousing*-Systeme befinden sich auf der Suche nach *Data-Mining*-Werkzeugen für die Erweiterung ihrer analytischen Fähigkeiten. Der Integrationstrend beim Gestalten der beiden Systeme wird in der kommenden Zukunft immer deutlicher. Mehrere moderne erfolgreiche Untersuchungen in der Statistik leisten ständig Beiträge zur Entwicklung neuer *Data-Mining*-Werkzeuge. Die anderen Richtungen, in denen *Data-Mining*-Werkzeuge immer populärer werden, sind das logische Programmieren, die Visualisierung von Daten, die weitergehende Internetanalyse etc. Die zukünftige Entwicklung von *Data Mining* scheint also sehr versprechend zu sein.

Literaturverzeichnis

[WI98] S.M Weiss, N. Indurkhya : *Predictive Data Mining*, Morgan Kaufmann, 1998

[CHY96] M.-S. Chen, J. Han, P.S. Yu : *Data Mining: An Overview from Database Perspective,* IEEE TKDE 8 (6), 1996

[DF95] K.M. Decker, S. Focardi : *Technology Overview: A Report on Data Mining,*TechReport TR-95-02, CSCS-ETH, 1995

[HS94] M. Holsheimer, A.P.J.M. Siebes : *Data Mining: the search for knowledge in databases*, TechReport Cs-R9406, CWI Amsterdam, 1994

[Fa94] U.M. Fayyad et al. : *Advances in Knowledge and Data Mining,* AAAI/MIT Press, 1996