Seminar

Business Intelligence

OLAP & Data Warehousing

Ausarbeitung zum Thema

Data Preprocessing II

Sabine Queckbörner
s_queckb@informatik.uni-kl.de
SS 2003

Data Preprocessing II
Datenreduktion und Kompression

Inhalt:

1 Einleitung

Möchte man eine sehr große Menge von Daten zum Beispiel aus einem Data-Warehouse analysieren, wird eine komplexe Auswertung und Datenmustererkennung aufgrund der Datenmenge sehr lange dauern oder ganz unmöglich sein. Durch Anwendung verschiedener Reduktions- und Kompressionstechniken auf die gesammelten Daten können Datenrepräsentationen mit wesentlich geringerem Volumen erzeugt werden, die jedoch die gleichen Informationen enthalten.

Um eine Datenreduktion zu erreichen, kann man mit Hilfe von „Data-Cubes“ (Datenwürfeln) die gesammelten Daten gruppieren und zusammenfassen. Eine einfache Mengenreduktion, bei der irrelevante, weniger relevante oder redundante Attribute entdeckt und entfernt werden, kann ebenfalls zu diesem Zweck durchgeführt werden. Man kann eine numerosity Reduction verwenden, um Daten abzuschätzen und durch alternative, kleinere Datenrepräsentationen zu ersetzen. Ebenso wäre die Diskretisierung von Daten zu nennen, welche die Werte der „Rohdaten“ durch Intervalle ersetzt.
Zur Datenkomprimierung, welche die Größe der Daten reduziert, können Transformationsverfahren angewandt werden. Diese stellen die gegebenen Daten ein einer anderen Art und Weise dar, so dass sie im Anschluss effektiver kodiert werden können.

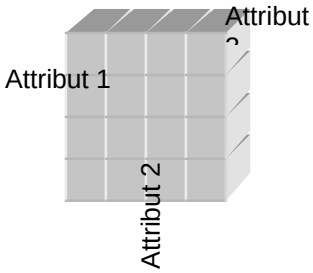
Eine Datenmustererkennung auf einem solchen reduzierten Datensatz ist wesentlich effizienter und liefern die selben, beziehungsweise fast die selben Analyseergebnisse.

2 Reduktion

2.1 Aggregation von Daten

2.1.1 Data-Cubes

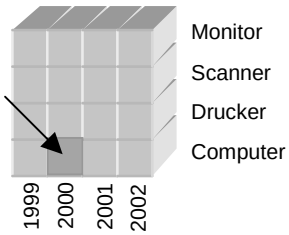
Unter einem *Data-Cube* versteht man mehrdimensional angeordnete Informationen. Jede Dimension kann dabei ein Attribut dieser Information darstellen, so dass durch Zusammenfügen von Attributen, beziehungsweise deren Werten, ein Würfel entsteht, wie er nebenstehend angeführt ist.



Ein Beispiel für einen Data Cube

Für jedes Attribut können Konzepthierarchien bestehen, die bei der Datenanalyse verschiedene Abstraktionsebenen ermöglichen. Zum Beispiel kann eine Hierarchie für ein Attribut „Geschäftszweig“ die Einträge aufgrund ihrer Adressen nach Regionen gruppieren.

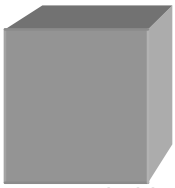
Die kleinste Einheit einer solchen Datenanordnung bildet der Basiswürfel, ein *Base-Cuboid*, der die detailliertesten, beziehungsweise speziellsten Informationen enthält. Basiswürfel bilden so die unterste Abstraktionsebene. Als Beispiel für einen Base-Cuboid könnte hier die Verkaufssumme der Firma XY im Jahre 2000 für Computer genannt werden.



Base-Cuboid: Verkaufssumme im Jahr 2000 für Computer

Durch Zusammensetzen mehrerer Basiswürfel entstehen neue Teilwürfel, die deren Informationen in einem Gesamtwert zusammenfassen. Zur Aggregation bieten sich beispielsweise die Summe, die Anzahl, der Mittelwert, die Varianz, das Minimum oder Maximum über bestimmte Attribute an [vgl. 5].

Einen Würfel der höchsten Abstraktionsebene bezeichnet man als *Apex-Cuboid*. Für das eben herangezogene Beispiel würde der Apex-Cuboid das Gesamtergebnis aller Verkäufe aller Jahre für alle Geschäftszweige und alle Artikel zurückgeben.



Apex-Cuboid

[vgl. 1, 5]

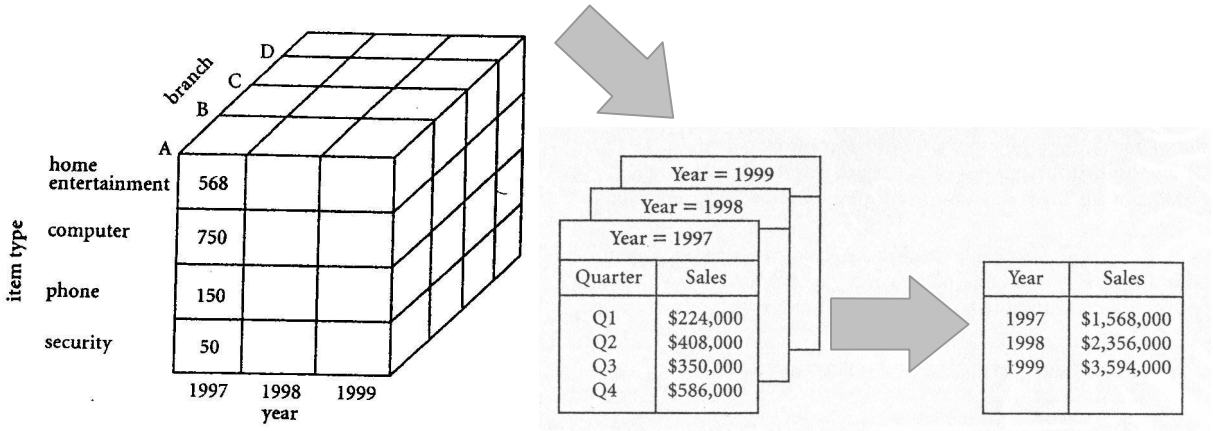
2.1.2 Reduktion durch Aggregation

Eine Reduktion mittels Gruppierung gesammelter Daten in Data-Cubes wird durch verschiedene Abstraktionsebenen erreicht, in denen Teile des Würfels immer weiter zusammengefasst werden. Jede höhere Abstraktionsebene reduziert so weiter die Größe der resultierenden Daten. Die höchste Abstraktionsebene und damit die größte Reduktion, wird mit der Aggregation aller Teilwürfel zu einem Gesamtwürfel erreicht.

Data Cubes unterstützen unmittelbare Zugriffe auf vorbereitete (precomputed), zusammengefasste Daten, wodurch sowohl On-line-Analyseprozesse als auch Data Mining (Datenmustererkennung) begünstigt und dadurch beschleunigt werden.
Deshalb sollten sie, wenn möglich, benutzt werden, um Anfragen auf zusammengesetzte Informationen schneller zu beantworten. Dazu sollte der kleinstmögliche auf diese Anfrage bezogene Cuboid verwendet werden [aus 1].

2.1.3 Beispiel

Um die Verkaufszahlen einer Firma zu analysieren, wurden Daten gesammelt, die aus den Verkaufsdaten pro Quartal der Jahre 1997 bis 1999 bestehen. Da für die Analyse allerdings die jährlichen Verkäufe interessanter sind als die eines Quartals, werden die Daten entsprechend vorbereitet:



Beispiel, wie ein Data-Cube in Tabellen zerlegbar ist, die auf eine Tabelle reduziert werden kann [aus 1, S.117, 118].

Die Daten eines Würfels können auch als mehrere Tabellen dargestellt werden. Diese kann man zur Vorbereitung der Anfrage nach den jährlichen Verkaufsdaten zu einer Tabelle zusammenfassen. Der daraus resultierende Datensatz hat ein

wesentlich kleineres Volumen (zweidimensionale Tabelle anstatt dreidimensionaler Würfel) als die ursprünglich gesammelten Daten, ohne bedeutende Informationen für diese Auswertung verloren zu haben.

2.2 Mengenreduktion

Bei einer Mengenreduktion werden redundante, irrelevante oder weniger wichtige Attribute der zu bearbeitenden Datensätze ignoriert oder gelöscht.

Da zu analysierende Datensätze sehr viele Attribute enthalten, von denen mehrere irrelevant oder redundant sind, kann durch Weglassen beziehungsweise Wegstreichen dieser Attribute die Größe eines einzelnen Datensatzes reduziert werden.

Im Allgemeinen wäre es zwar möglich, die für eine bestimmte Anfrage nützlichen Attribute herauszusuchen und den Rest zu streichen, allerdings ist dies schwierig und zeitaufwändig. Ein Weglassen von relevanten Attributen, beziehungsweise ein Mitnehmen oder Speichern von irrelevanten oder redundanten Attributen kann die Qualität der gesammelten Daten negativ beeinträchtigen und sie unter Umständen durch Ausgabe überflüssiger Attribute aufblähen. Eine Mengenreduktion verringert die Datensatzgröße, indem solche Attribute entfernt werden. Zum Herausfiltern von bestimmten Attributen gibt es mehrere Verfahren, von denen im Folgenden einige aufgeführt werden sollen [vgl. 1].

2.2.1 Auswahlverfahren für Attribute eines Datensatzes

Solche Auswahlverfahren haben zum Ziel, eine minimale Menge von Attributen zu finden, die für eine Analyse wirklich relevant sind, ohne dass bei einer Auswertung irgendwelche Informationen fehlen. Für eine anschließende Datenmustererkennung hat ein solches Verfahren einen zusätzlichen Nutzen: Die Zahl der Attribute, die in den gefundenen Mustern erscheinen wird verringert, wodurch diese besser zu erkennen und leichter zu verstehen sind.

Da es für eine Menge von n Attributen 2^n mögliche Teilmengen gibt, kann eine „blinde“ Suche (wie zum Beispiel mit Backtracking) nach einer optimalen Teilmenge sehr zeitaufwändig und daher teuer sein. Deshalb werden hierfür normalerweise heuristische Verfahren verwendet.

Heuristische Algorithmen versuchen eine (möglichst optimale) Lösung in einem (exponentiell) großen Lösungsraum mit Hilfe einer durch ‚Heuristiken' gesteuerten Suche zu finden. Dabei sind Heuristiken „problem-spezifische Informationen, die es ermöglichen, evtl. schneller zu einer (optimalen oder wenigstens passablen) Lösung zu gelangen als durch ‚blindes' Suchen“.

Die benötigten Auswahlmethoden sind typischerweise Greedy-Heuristiken, die nur auf Grund lokal verfügbarer Informationen den nächsten Lösungs-Erweiterungsschritt vornehmen, nämlich denjenigen, der für den Moment am Besten scheint. Ihre Strategie ist es, die „lokal“ beste Wahl zu treffen, in der Hoffnung, dass diese zur „global“ optimalen (Gesamt-)Lösung führt [vgl. 1,2].

2.2.1.1 Heuristische Basismethoden zur Attributauswahl

Stepwise forward selection (schrittweise Vorwärtsauswahl):

Ausgehend von einer leeren (Attributs-)Menge ermittelt dieses Verfahren das Beste der „Originalattribute“ und fügt es zu dieser Menge hinzu. Bei jedem weiteren Schritt wird das beste verbleibende Attribut zur Lösungsmenge hinzugefügt.

Stepwise backward elemination (schrittweise Rückwärtseliminierung):

Dieses Verfahren startet mit der Menge der Originalattribute als Lösungsmenge. Bei jedem Schritt wird das schlechteste verbleibende Attribut aus ihr entfernt.

Combination of forward selection and backward elemination
(Kombination aus Vorwärtsauswahl und Rückwärtseliminierung):

Die Methoden der schrittweisen Vorwärtsauswahl und die der schrittweisen Rückwärtseliminierung können kombiniert werden, so dass bei jedem Schritt des Verfahrens das beste Attribut ausgewählt und einer Attributmenge hinzugefügt und das schlechteste Attribut aus der Menge der verbleibenden Attribute entfernt wird [aus 1].

Die „besten“ und „schlechtesten“ Attribute werden typischerweise mit Hilfe statistischer Signifikanztests bestimmt, mit der Annahme, dass die Attribute voneinander unabhängig sind.
Eine andere Methode die Attribute zu bewerten ist zum Beispiel die Bildung von Entscheidungsbäumen zur Klassifizierung. Dabei wird ein Baum aufgebaut, dessen interne Knoten einen Test eines Attributes darstellen, jede Kante ein Ergebnis eines solchen Tests und jedes Blatt einen Klassifizierungsvorschlag. Zur Auswahl einer Attributmenge wird ein solcher Baum aus den gegebenen Daten aufgebaut. Alle Attribute, die nicht im Baum auftauchen, können verworfen werden, die vorkommenden Attribute bilden die gesuchte Untermenge.

Die Abbruchkriterien für die oben genannten Verfahren können variieren. Man kann zum Beispiel einen Grenzwert für die Auswahl einsetzen, bei dessen Erreichen das Auswahlverfahren stoppt. [aus 1]

2.2.1.2 Qualität der Auswahlverfahren

Die Qualität (oder den Informationsverlust) eines solchen Auswahlverfahrens kann man mit Hilfe der Stochastik folgendermaßen beurteilen:
Eine Wahrscheinlichkeitsverteilung der Klassifizierung der relevanten Attribute sollte nach dem Herausfiltern von irrelevanten Attribute nahe der ursprünglichen Verteilung sein. Das heißt, dass die Attribute nach dem Herausfiltern genauso klassifiziert werden wie vor dem Filtern [vgl. 1].

2.3 Numerosity Reduction

Mit der *Numerosity Reduction* (numerosity = zahlreiches Auftreten) werden die ursprünglichen Daten durch alternative, kleinere Datenrepräsentationen ersetzt. Bei dieser Technik der Datenreduktion können parametrisierte und nichtparametrisierte Modelle verwendet werden. Im Folgenden wird die numerische Reduktion in Form von Regression und Log-linearen Modellen, Histogrammen, Clustering und Sampling weiter erläutert.

2.3.1 Regression und log-lineare Modelle

Regressionen und log-lineare Modelle können verwendet werden, um die gegebenen Daten zu approximieren. Eine Reduktion der Originaldaten wird bei der Regression dadurch erreicht, das die Daten durch Koeffizienten einer linearen Funktion ersetzt werden.

Bei der linearen Regression werden die Daten so modelliert, dass die Näherungen einer Geraden gleichen. Anstelle der aktuellen Daten müssen bei einer solchen parametrisierten Datenreduktion nur die Parameter in Form einer linearen Gleichung, zum Beispiel: $Y = \alpha + \beta X$ gespeichert werden, wobei X und Y Zufallsvariablen sind.

Die multiple Regression ist eine Erweiterung der linearen Regression. Hier kann die Variable Y als lineare Funktion eines multidimensionalen Vektors modelliert werden: $Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2$

Log-lineare Modelle approximieren diskrete, mehrdimensionale Wahrscheinlichkeits-verteilungen. Diese Methode der numerischen Datenreduktion kann zum Beispiel bei der mehrdimensionalen Datendarstellung dazu verwendet werden, um die Wahrscheinlichkeit einer jeden Zelle eines Basiswürfels für eine Menge von diskreten Attributen abzuschätzen, um dann ein Gitter aus diesen Datenwürfeln zu bilden[aus 1].

2.3.2 Histogramme

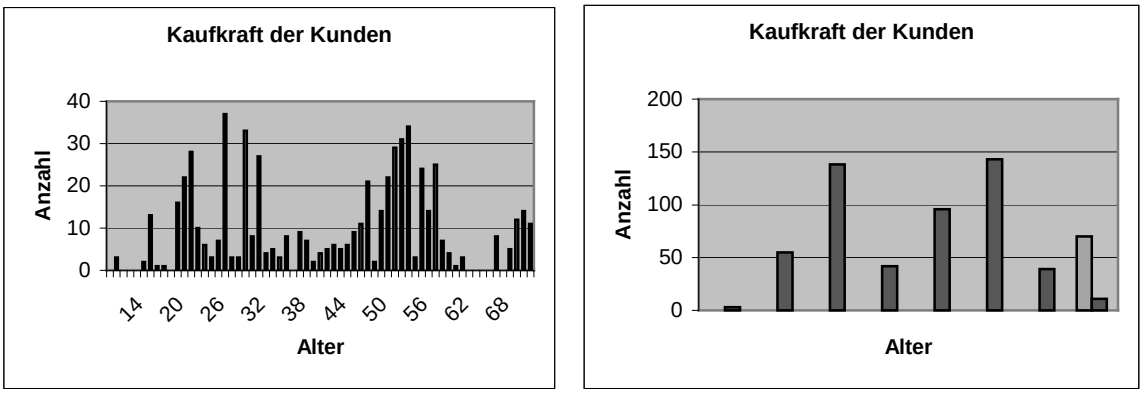
Histogramme nutzen das „Binning“ (Zusammenfassen benachbarter Zeichen, Pixel, Zeilen...) zum Approximieren von Daten und sind eine populäre Form der Datenreduktion.
Ein Histogramm teilt die Daten in disjunkte Teilmengen (buckets) auf, die auf einer horizontalen Achse aufgetragen werden; die Höhe und Fläche der Teilmengen (buckets) entlang der Y-Achse geben die Anzahl der Vorkommen der Elemente in den einzelnen Teilmengen an. Es gibt dabei folgende Verteilungsmöglichkeiten:

- Equiwidth: In einem solchen Histogramm ist der Wertebereich einer jeden Teilmenge gleich
  (Im Beispiel unten: 1-9 Jahre, 10-19Jahre, ... ,70-79Jahre)
- Equidepth: (auch Equiheight) In einem solchen Histogramm sind die Teilmengen so gewählt, dass die Anzahl der Vorkommen pro Teilmenge konstant ist, also jedes *bucket* die selbe Anzahl von zusammenhängenden Datenmustern enthält)
- V-Optimal: Das Histogramm mit der kleinsten Streuung: In einem solchen Histogramm ist die Mächtigkeit der Menge der Teilmengen (buckets) gleich der Mächtigkeit der Elemente der Mengen (Anzahl der Werte in den buckets).
- MaxDiff: Bei einem MaxDiff-Histogramm wird die Differenz zwischen jedem benachbarten Werte-Paar betrachtet. Eine Mengengrenze wird zwischen jedem Paar mit einer (β – 1)-größten Differenz gezogen, wobei β vom Benutzer festgelegt wird.

V-Optimal und MaxDiff sind hiervon die praktikabelsten und akkuratesten Methoden. Gespeichert wird hierbei die Mengen-ID und nicht das Datum. Reduziert werden die Daten zum Beispiel, wenn die X-Achse des Histogramms eine gröbere Einteilung bekommt. Dadurch werden die Werte, die bisher einzeln standen zu einem Wert zusammengefasst [aus 1].

Beispiel

Ein Histogramm, dass zum Beispiel die Kaufkraft von Kunden eines Unternehmens in Abhängigkeit ihres Alters darstellt, könnte folgendermaßen aussehen:

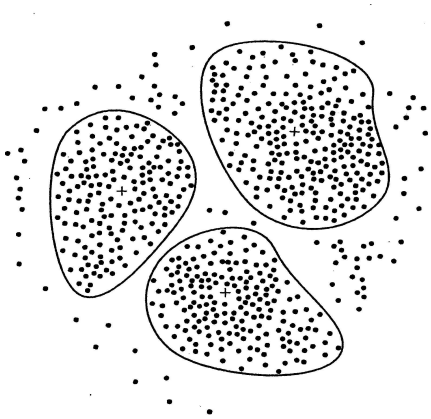


Ein Histogramm, das die Käufe von Kunden eines bestimmten Alters darstellt.

Reduziert wurde hier dadurch, dass die X-Achse in buckets eingeteilt wurde, die jeweils 10 Jahre enthalten, wodurch sichtbar weniger Werte gespeichert werden müssen.

2.3.3 Clustering

Clustering-Methoden setzen Datentupel als Objekte voraus. Zur Reduktion der Daten partitionieren sie diese Objekte in Gruppen oder Cluster, so dass Objekte innerhalb eines Clusters einander ähnlich sind und sich von Objekten aus anderen Clustern unterscheiden. Ähnlichkeit wird hier, basierend auf einer Distanzfunktion, als Nähe zu einem anderen Objekt definiert.
Die Qualität eines Clusters wird durch dessen Durchmesser (der maximalen Distanz zwischen beliebigen zwei beliebigen Objekten eines Clusters) repräsentiert. Ein alternatives Qualitätsmaß ist die sog. *Centroid-Distanz*, die als durchschnittliche Distanz eines jeden Objektes zum Cluster-Centroid, einer Art Mittelpunkt des Clusters, definiert ist.
Ein Beispiel einer Einteilung von Daten in Cluster:



Beispiel:
Adressen von Kunden
einer Stadt, [aus 1, S. 128]

Um die ursprünglich gegebenen Daten zu reduzieren, ersetzt man nun diese durch eine Repräsentation der Cluster. Die Effektivität dieser Technik hängt von der Beschaffenheit der Daten ab, bei schlecht in Cluster einzuteilenden Daten ist sie eher ineffektiv.
In Datenbanksystemen werden zur schnellen Datenbereitstellung hauptsächlich (multidimensionale) Indexbäume verwendet, die ebenfalls zur Reduktion von hierarchischen Daten benutzt werden können, indem die Clusterstruktur in einem solchen mehrdimensionalen Baum gespeichert wird [vgl. 1].

2.3.4 Sampling (Stichproben)

Stichproben können als Datenreduktionsmethode genutzt werden, da es erlaubt ist, große Datenmengen mittels kleinerer Zufallsstichproben der Daten darzustellen.

Mögliche Stichprobenarten eines Datensatzes D mit N Tupeln sind:

- *Einfache zufällige Stichprobe ohne Ersetzung der Größe n:*
  (simple random sample without replacement, SRSWOR), wird durch n Tupel aus D, n < N erzeugt, wobei jedes Tupel gleichwahrscheinlich ist und jedes gezogene Tupel aus D entfernt wird.

- *Einfache zufällige Stichprobe mit Ersetzung der Größe n:*
  (simple random sample with replacement, SRSWR), genau wie SRSWOR, nur dass jedes aus D entnommene Tupel in der Stichprobe verzeichnet wird, dann aber wieder an D zurückgegeben wird, so dass es erneut gezogen werden kann.

- *Clusterstichprobe* (cluster sample)*:*
  Wenn die Tupel in D in M disjunkte Cluster aufgeteilt sind, kann eine einfache zufällige Stichprobe von m Clustern mit m < M gezogen werden.

- *Schichtenweise Stichproben* (stratified sample):
  Ist der Datensatz D in disjunkte Teile (strata) geteilt, wird eine schichtenweise Stichprobe durch ein SRS (Simple Random Sample, ein zufälliges Ziehen eines Elements) auf jedem solchen „Stratum“ durchgeführt, was garantiert, dass alle Schichten in der Stichprobenmenge enthalten sind.

Ein Vorteil des Samplings ist, dass die Kosten zum Erhalten einer Datenrepräsentation proportional zur Größe n der Stichprobe ist. Andere Datenreduktionsmethoden erfordern meistens einen kompletten Durchlauf der Daten [aus 1].

2.4 Diskretisierung von Datenwerten

Diskretisierungstechniken verringern die Anzahl der Werte für ein durchgängiges Attribut (continuos attribute), in dem sie dessen Wertebereiche in einzelne Intervalle aufteilen. Die Intervallbezeichnungen (labels) können dann zum Ersetzen der eigentlichen Werte verwendet werden.
Dieses Verfahren der Datenreduktion ist besonders dann von Vorteil, wenn Entscheidungsbaum-basierte Methoden des „Klassifizierungs-Mining“ auf die vorbereiteten Daten angewandt werden sollen. Viele der Diskretisierungsmethoden können rekursiv angewendet werden, um eine hierarchische oder unterschiedlich aufgelöste Aufteilung der Attributwerte zu erzeugen, die als Konzepthierarchie bekannt ist. Diese Konzepthierarchien sind für eine Mustererkennung auf unterschiedlichen Abstraktionsebenen nützlich und können dafür benutzt werden, Daten einer tieferen Ebene durch Daten höherer Konzeptebenen zu ersetzen (zum Beispiel Geburtsjahrgänge durch Altersklassen wie jung, mittel, alt; mehrere Postleitzahlen durch Städtenamen, mehrere Städte durch Regionen). Auch wenn Details verloren gehen, können die so generalisierten Daten aussagekräftiger und einfacher als die Originaldaten zu interpretieren oder weiter zu verarbeiten sein
[vgl. 1].

2.4.1 Numerische Daten

Zur Diskretisierung und Konzepthierarchiebildung für numerische Daten können verschieden Methoden verwendet werden. Eine davon ist das sogenannte Binning, weitere wären die Histogramm- und die Cluster-Analyse. Außerdem sind die Entropie-basierte Diskretisierung und die Segmentierung durch natürliche Partitionierung zu nennen.

2.4.1.1 Binning

Diese Methode reduziert und glättet die Daten durch Beobachtung der Umgebung: Die gegebenen Datenwerte werden sortiert und in sogenannte „Bins“ (Eimer) gruppiert und durch Mittelwerte oder Grenzwerte ersetzt. Um Konzepthierarchien zu erzeugen können diese Schritte rekursiv angewandt werden [vgl. 1].

2.4.1.2 Histogramm-Analyse

Histogramme können zur Datendiskretisierung genutzt werden, indem mit Hilfe von Partitionierungsregeln die Wertebereiche festgelegt werden. Die Histogramm-Analyse kann auf jede Partition angewandt werden, um automatisch eine Multiebenen-Konzepthierarchie zu erzeugen.

2.4.1.3 Cluster-Analyse

Ein Cluster-Analyse-Algorithmus kann angewandt werden, um Daten in Gruppen einzuteilen, von denen jede einen Knoten einer (gleichen) Konzepthierarchie darstellt. Jedes dieser Cluster kann wieder in Untergruppen geteilt werden, die dann eine tiefere Hierarchieebene bilden.

2.4.1.4 Entropie-basierte Diskretisierung

Ein Informationsbasiertes Maß, das Entropie genannt wird, kann genutzt werden, um die Werte eines numerischen Attributes A rekursiv zu partitionieren. Dadurch kann man ebenfalls eine hierarchische Diskretisierung erhalten. Eine solche Diskretisierung bildet eine numerische Konzepthierarchie für dieses Attribut.

2.4.1.5 Segmentierung durch natürliche Partitionierung

Obwohl Binning, Histogramm-Analysen, Clustering und Entropie-basierte Diskretisierungen nützlich sind, um numerische Hierarchien zu bilden, möchten viele Benutzer die numerischen Daten lieber in relativ gleichen Intervallen, die einfach zu lesen sind und intuitiv oder „natürlich“ eingeteilt sind, sehen. Zum Beispiel Preisintervalle mit glatten Grenzen anstatt Kommawerten.

Die sogenannte *3-4-5-Regel* kann genutzt werden, um numerische Daten in relativ uniforme, "natürliche" Intervalle zu segmentieren. Generell teilt diese Regel einen gegebenen Wertebereich in 3, 4 oder 5 relativ gleichweite Intervalle, rekursiv und Ebene für Ebene, basierend auf dem Wertebereich der größten Zahl. Im Folgenden eine Erläuterung der Regel:

- Falls ein Intervall drei, sechs, sieben oder neun unterschiedliche Werte an der positivsten Stelle aufweist, so wird der Bereich in drei Intervalle (drei gleich weite für drei, sechs und neun, und 2-3-2 für sieben) partitioniert.
- Falls ein Intervall zwei, vier oder acht unterschiedliche Werte an der positivsten Stelle hat, so wird der Bereich in vier gleichweite Intervalle aufgeteilt.
- Falls ein Intervall einen, fünf oder zehn unterschiedliche Werte an der positivsten Stelle aufweist, so wird der Bereich in fünf gleichweite Intervalle aufgeteilt.

Diese Regel kann rekursiv auf jedes Intervall angewendet werden, wodurch eine Konzepthierarchie für das gegebene numerische Attribut erzeugt wird [aus 1].

2.4.1.5.1 Beispiel:

Um diese Regel zu erläutern, sei zuerst einmal erklärt, was mit der positivsten Stelle einer Zahl gemeint ist: Hat man zum Beispiel eine Dezimalzahl 461, so wird die Ziffer 1 mit 1 multipliziert, die Ziffer 6 mit 10 und die Ziffer 4 mit 100. Daraus ergeben sich die Werte 1, 60 und 400. Die 4 ist in diesem Fall die sogenannte positivste Stelle der Zahl. Damit ist immer die am weitesten links stehende Ziffer die positivste Stelle. Um unterschiedlich lange Zahlen miteinander vergleichen zu können, zum Beispiel die 461 mit der 43, werden die „kürzeren“ Zahlen vorne mit Nullen aufgefüllt, so dass zum Beispiel die 43 als 043 dargestellt wird.

Zur Erläuterung der 2-3-4 – Regel seien nun die Verkäufe von Waschmaschinen einer Firma innerhalb eines Jahres in 8 verschiedenen Filialen genannt: Die zu vergleichenden Werte sind 9, 13, 20, 37, 42, 43, 59, und 63. Vergleicht man die Ziffern der sogenannten positivsten Stelle, so bemerkt man sieben unterschiedliche Werte. Nach der oben genannten Regel ist das bisherige Intervall nun wie folgt in drei Unterintervalle einzuteilen: (0..19), (20..49), (50..69). Um sich nun einen Bereich genauer anschauen zu können, zum Beispiel den zweiten (von 20 bis 49), da dort vier der zu untersuchenden Werte enthalten sind, wendet man wieder die Regel an: Man stellt bei den genannten Werten drei verschiedene Werte fest, nämlich 2, 3 und 4. Daraus ergibt sich, dass das Intervall in drei Unterintervalle einzuteilen ist: (20..29), (30..39), (40..49). Um weiter in die einzelnen Bereiche schauen zu können, kann man die 2-3-4 – Regel immer wieder rekursiv anwenden.

2.4.2 Kategorische Daten

Kategorische Daten sind einzelne Daten wie zum Beispiel Orte, Job-Kategorien oder Gegenstandstyp (item-types). Kategorische Attribute haben eine endliche Anzahl verschiedener Werte, die nicht geordnet sind. Auch hier gibt es verschiedene Methoden, Konzepthierarchien zu generieren [aus 1]:

- *Spezifizierung einer partiellen Ordnung der Attribute explizit auf Schemaebene durch Nutzer oder Experten:* Konzepthierarchien für kategorische Attribute oder Dimensionen beinhalten typischerweise wiederum eine Gruppe von Attributen. Ein Nutzer oder Experte kann einfach eine Konzepthierarchie definieren, indem er eine partielle oder totale Ordnung der Attribute auf Schemaebene spezifiziert.

- *Spezifizierung einer Menge von Attributen, aber nicht deren partieller Ordnung:* Der Benutzer kann eine Menge von Attributen spezifizieren, die eine Konzepthierarchie bilden. Das System versucht dann, automatisch eine Attributordnung zu erzeugen, um eine aussagekräftige Konzepthierarchie zu konstruieren.
  Wenn Konzepte höherer Ebene viele untergeordnete aus einer tieferen Ebene beinhalten, dann besteht ein Attribut aus einer höheren Konzeptebene aus einer kleineren Anzahl von unterschiedlichen Werten als ein Attribut einer tieferen Ebene, so dass man eine Konzepthierarchie basierend auf der Anzahl unterschiedlicher Werte pro Attribut erzeugen lassen kann.

  Das Attribut mit den meisten unterschiedlichen Werten wird in der Konzepthierarchie nach unten geordnet, je weniger Unterschiede es gibt, desto höher wird das Attribut eingeordnet. Am Ende können noch Eingriffe durch Benutzer oder Experten erfolgen, um letzte Korrekturen durchzuführen.

- *Spezifizierung nur einer kleinen Menge von Attributen:*
  Um mit solchen nur zum Teil spezifizierten Hierarchien umzugehen, ist es wichtig, Datensemantik mit in das Schema einzubinden, so dass Attribute mit ähnlicher Semantik aneinandergesetzt werden können. So kann die Spezifikation eines Attributes das Erzeugen einer Gruppe von semantisch nah beieinanderstehenden Attributen in eine Konzepthierarchie auslösen.

3 Kompression

Bei der Datenkompression werden Datenkodierungen oder -umwandlungen (Transformationen) angewendet, um eine verringerte oder komprimierte Darstellung der ursprünglichen Daten zu erreichen.

Wenn die ursprünglichen Daten aus den komprimierten Daten ohne irgendeinen Informationsverlust wieder aufgebaut werden können, wird die verwendete Kompressionstechnik verlustfrei genannt. Kann man statt dessen nur einen Näherungswert der ursprünglichen Daten aufbauen, so ist die Kompressionstechnik verlustbehaftet.

3.1 Kompression multidimensionaler Daten

Die Kompression von multidimensionalen Daten kann mit den gleichen Verfahren erfolgen wie die von Bild- und Videodaten, da auch diese einen mehrdimensionalen Charakter aufweisen.
Die momentan üblichen Komprimierungsverfahren für Bilddaten arbeiten nach einem dreistufigen Prinzip, der sogenannten *Transformationskodierung*. Mathematisch gesehen versteht man unter einer Transformation einen Basiswechsel. Das heißt, man sieht den gleichen Sachverhalt aus einem anderen Blickwinkel – also auf eine andere Basis aufgebaut. Die Informationen gehen hierbei nicht verloren, sie werden nur anders dargestellt. Eine Transformation auf eine Datenmenge angewandt reduziert allerdings noch nicht die Größe der Daten, sie stellt diese lediglich so dar, dass ein Kodierungsalgorithmus sie platzsparender speichern kann als die Originaldaten.

3.1.1 Das Prinzip der Transformationskodierung:

Die erste Stufe dieses Verfahrens besteht aus der eigentlichen *Transformation* des Bildes, die noch keine Verringerung der Datenmenge zur Folge hat, sondern lediglich eine bessere Ausgangsbasis für die nachfolgenden Schritte schafft. Bekannte und bewährte Transformationsverfahren sind beispielsweise die Fourier - Transformation (FT) beziehungsweise die Diskrete Fourier - Transformation (DFT), sowie die ähnliche Diskrete Wavelet - Transformation (DWT) und die Diskrete Cosinus - Transformation (DCT). Letztere wird beim bislang wohl bekanntesten und am häufigsten verwendeten Kompressionsverfahren JPEG eingesetzt.

In der nächsten Stufe, der *Quantisierung*, wird der Wertebereich der Bildpunkte eingeschränkt, indem die Werte auf festgelegte Quantisierungsstufen gerundet werden. Dieser Schritt erzeugt unvermeidlich Verluste, die nicht mehr rückgängig gemacht werden können. Bei einer verlustfreien Bildkompression muss daher auf die Quantisierung verzichtet werden. Allerdings ist dann die Kompressionsrate auch wesentlich niedriger.

Der entscheidende und letzte Schritt dieses Verfahrens ist die *Binärcodierung* der quantisierten Bildpunkte. Bekannte Verfahren sind hierfür beispielsweise die Lauflängenkodierung (Run Length Encoding (RLE), welche die Zahl aufeinanderfolgender Nullen (beziehungsweise aufeinanderfolgender Einsen) als Symbol kodiert, oder der Huffman - Code.

Die Rekonstruktion der quantisierten Bilddatenbits erfolgt im entsprechend umgekehrten Durchlaufen des Schemas (Decodierung, Wertrekonstruktion, Rücktransformation) [vgl. 8, 11].

3.1.2 Diskrete Wavelet-Transformation

Die Diskrete Wavelet-Transformation (DWT) ist eine lineare Signalverarbeitungstechnik, die einen Datenvektor in einen Vektor (vorerst) gleicher Länge transformiert, der jedoch durch Bearbeitung gekürzt werden kann: Eine komprimierte Näherung der ursprünglichen Daten kann durch das Speichern eines kleinen Teils des stärksten Wellenkoeffizienten erreicht werden. Aus dieser Darstellung können die ursprünglichen Daten wieder hergestellt werden, wobei der Informationsverlust hierbei von dem Umfang der Kürzungen abhängt.

Mathematischer beschrieben ist die Diskrete Wavelet-Transformation eine Frequenz-Transformation, die als Basisfunktion zur Komprimierung Wavelets verwendet und zeit- und frequenzdiskret durchgeführt wird. Am häufigsten wird sie bei der Bild- und Videokompression verwendet. [siehe auch 7, 11, 12]

Sie stellt in gewissem Sinne eine Weiterentwicklung der Diskreten Fourier-Transformation (DCT) dar, mit der sie einiges gemeinsam hat. Beide Verfahren werden in einer Art Vorstufe eingesetzt, um ein Bild in eine Form zu bringen, die eine effiziente Entropie-Kodierung ermöglicht.

Bei der Wavelet-Transformation wird ein Signal durch Faltung und Differenzbildung mit einem sogenannten Wavelet in verschiedene Ebenen zerlegt.
Wavelets („kleine Wellen“) sind mathematische Funktionen, die den Sinus- und Cosinusfunktionen der Fourier-Transformation ähnlich sind und dazu benutzt werden, eine gegebene Funktion auf enthaltene Frequenzbestandteile hin zu untersuchen. Dazu können sie gestreckt und gestaucht oder an eine bestimmte Stelle des Signals verschoben werden. Zum Beispiel zerlegt man mit Hilfe der Wavelet-Transformation ein Bild in solche mathematische Funktionen („Wavelets“) [vgl. 12].

Verschiedene Wavelet-Funktionen:

Haar-Wavelet, Daubechis4-Wavelet und Daubechis20-Wavelet [aus 11]

Im Folgenden wird die Wavelet – Transformation an der einfachsten Wavelet-Funktion, dem Haar-Wavelet erläutert.
Eine Analyse (eine Faltung) mit den Haar-Funktionen bedeutet im einfachsten Fall, dass zuerst der Mittelwert und die Differenz zwischen zwei benachbarten Pixelwerten berechnet wird. Nachdem man dann die Ergebnisse unterabgetastet und als Tiefpass- und Hochpassanteile gespeichert hat, wird der Tiefpassanteil mit den Haar-Funktionen weiter analysiert. Am Ende besteht das Wavelet-transformierte Bild aus einer gewissen Anzahl immer kleiner werdender Hochpassanteile und einem einzigen Tiefpassanteil [vgl. 9, 10].

3.1.2.1 Beispiel:

Das Ursprungsbild soll in zwei Transformationsstufen zerlegt werden. In der ersten Stufe wird es in drei Hochpassanteile (Dx1, Dy1, Dxy1), sowie in einen Tiefpassanteil (H1) zerlegt.

In einer weiteren Transformationsstufe wurde nur noch der erste Tiefpassanteil (H1) wiederum in drei Hochpassanteile (Dx2, Dy2, Dxy2) sowie einen neuen Tiefpassanteil (H2) zerlegt. Die Hochpassanteile der ersten Stufe bleiben dabei erhalten.
Auf diese Weise können nun weitere Transformationsstufen durchlaufen werden. In den Hochpassanteilen der ersten Stufe werden die feinen Bildstrukturen erfasst, in denen der folgenden Transformationsstufen werden zunehmend gröbere Bildstrukturen erfasst. [aus 10, 8]

Originalbild und zwei Iterationen [aus 10]

3.1.2.2 Qualität:

Untersuchungen im Zusammenhang mit der Kompression von Fingerabdrücken für das FBI haben gezeigt, dass – zumindest für diese speziellen Bilder – Wavelet-basierte Methoden anderen Kompressionsmethoden überlegen sind.
Die Vorteile äußern sich vor allem in einer wesentlich höheren möglichen Kompressionsrate beziehungsweise einer deutlich besseren Bildqualität bei gleicher Kompressionsrate. Außerdem treten keine störenden Blockartefakte im rekonstruierten Bild auf, wie sie beispielsweise bei JPEG durch die blockweise angewandte DCT - Transformation entstehen.

Ein direkter Vergleich zwischen einer JPEG - und einer Wavelet-Transformation macht den Qualitätsunterschied (bei einer Kompressionsrate von ca. 178:1) deutlich sichtbar:

JPEG

Wavelet [Abb. aus 8]

3.1.3 Principal Component Analyse – Hauptkomponentenanalyse

Die Hauptkomponentenanalyse (Principal Component Analysis, PCA) ist eine weitere, oft verwendete Kompressionsmethode, die auf dem Prinzip der Transformation beruht: Die komprimierte Darstellung der ursprünglichen Daten bildet eine orthonormale Basis der ursprünglichen Unterbilder.

Bestehen die zu komprimierenden Daten aus N k-dimensionalen Tupeln, so sucht PCA nach c k-dimensionalen Tupeln ($c \leq k$), die am besten geeignet sind, die gegebenen Daten zu repräsentieren. Die Hauptkomponentenanalyse kombiniert das Wesentliche der Attribute durch Erzeugung einer alternativen kleineren Variablenmenge. Dies geschieht wie folgt:

- Zuerst werden die gegebenen Daten normalisiert, so dass alle Daten den selben Wertebereich haben.
- Anschließend werden c orthonormale Vektoren berechnet, die eine Basis für die Eingabedaten bilden und als Hauptbestandteile deklariert werden. Die Eingabedaten stellen eine Linearkombination der Hauptbestandteile dar.
- Die Hauptbestandteile werden in absteigender Signifikanz oder Stärke sortiert und in Form einer Menge von Achsen dargestellt, welche die wichtigen Informationen über die Abweichungen bereitstellen.
- Danach kann die Größe der Daten durch Eliminieren von schwächeren Komponenten verkleinert werden. Das Nutzen stärkerer Hauptkomponenten ermöglicht die Rekonstruktion einer guten Annäherung der Originaldaten.

PCA ist vom Rechenaufwand günstig und kann auf geordnete und ungeordnete Attribute angewandt werden. Vergleicht man die beiden hier aufgeführten Transformationsverfahren, so eignet sich die Hauptkomponentenanalyse besser für spärliche Daten (sparse data), während sich die Wavelet Transformation eher für multidimensionale Daten eignet [siehe auch 1, 13].

3.2 Zeichenkettenkompression

Um Zeichenketten verlustfrei zu komprimieren, existieren drei grundsätzliche Ansätze. Zum einen wären die Dictionary-basierten Algorithmen zu nennen, die auch beim Packen von GIF-Dateien oder bei Modemverbindungen genutzt werden. Eine Kompression der Daten findet bei diesem Verfahren dadurch statt, dass Wiederholungen von Zeichen oder Zeichenblöcken nicht ein weiteres Mal abgespeichert wird, sondern lediglich eine Referenz auf das erste Vorkommen. Neben dieser Methode gibt es weiterhin die statistischen Kodierer. Diese Methode nutzt das unterschiedlich häufige Vorkommen einzelner Zeichen oder Zeichenblöcke, um diese in entsprechend kurzer Weise darzustellen. Eine weitere Methode Zeichenketten zu komprimieren, ist die sogenannte *Borrows-Wheeler-Transformation*.

3.2.1 Dictionary-basierte Algorithmen

Diese Algorithmen zur Zeichenkettenkompression bauen, während sie die Eingabe einlesen, ein *Dictionary* auf, in dem sie bereits aufgetretene Zeichenketten speichern. Diese Art Lexikon besteht im einfachsten Fall aus den letzten n KByte der gelesenen Daten. Soll nun ein neuer Text oder Textabschnitt komprimiert werden, wird immer zuerst geprüft, ob das jeweilige Textfragment bereits im „Dictionary“ vorhanden ist. Ist dies der Fall, muss nur noch eine Referenz auf den entsprechenden Eintrag kodiert werden, der meist wesentlich kürzer als die ursprüngliche Zeichenkette ist, wodurch die Kompression letztendlich zu Stande kommt. Ein Vorteil dieser Methode der Zeichenkettenkompression besteht darin, dass das „Dictionary“ während des Dekomprimierens aufgebaut werden kann, ohne dass zusätzliche Informationen zu den komprimierten Daten gespeichert und übertragen werden müssen [vgl. 6].

3.2.2 Statistische Kodierer

Statistische Kodierer (statistical Encoder) verwenden Wahrscheinlichkeits-verteilungen, um häufig vorkommende Zeichen oder Zeichenketten kürzer darstellen zu können als solche, die weniger oft vorkommen. Bei einem längeren Text kommt gerade dadurch eine Kompression zu Stande, dass die sehr klein gespeicherten Zeichenketten sehr oft und die auf mehr Platz gespeicherten Zeichen relativ selten

vorkommen. Hierbei gilt, dass die Kompressionsrate besser wird, je länger der Text ist.

Ein statistischer Kodierer besteht aus zwei Teilen, einem Modell und einem Entropie-Kodierer. Das Modell ist das jeweils zu Grunde liegende Wahrscheinlichkeitsmodell, das die eingelesenen Zeichen bewertet und gegebenenfalls das als nächstes auftretende Zeichen voraussagt. Ergibt sich die Wahrscheinlichkeit für das Auftreten eines Zeichens aus dem Quotienten der Vorkommen des jeweiligen Zeichens und der Summe aller Zeichen, so spricht man von einem *Order-0-Modell*, da hierbei kein Zeichen „vorhergesagt“ wird. Je nachdem, wie viele Zeichen vorausgesagt werden, spricht man von einem *Order-n-Modell*, wobei n die Länge der vorausgesagten Zeichenkette darstellt. Höhere Order-n-Modelle zeichnen sich zusätzlich durch eine starke Ungleichverteilung der Zeichen aus.
Zur Verdeutlichung ein Beispiel für die Einschätzung eines Zeichens in einem Order-1-Modell, bei dem *ein* Zeichen vorhergesagt wird: In einem deutschen Text tritt ein „u“ mit einer Wahrscheinlichkeit von 0,03 auf, was sich allerdings auf 0,99 erhöht, wenn das Zeichen davor ein „q“ ist.

Der zweite Teil eines statistischen Kodierers ist der Entropie-Kodierer. Er ist ein Algorithmus, der aufbauend auf dem gegebenen Modell die Zeichen kodiert. Die bekanntesten Verfahren sind der Huffman Code und die arithmetische Kodierung.

Ein Nachteil der statistischen Kodierer ist die lange Aufbauphase eines Modells: Je größer der betrachtete Kontext ist, desto länger dauert diese an. Die Modelle, die eine gute Kompressionsrate versprechen, erreichen diese erst, wenn bereits eine große Menge von Daten verarbeitet wurde. Außerdem muss dieses Modell mit den komprimierten Daten gespeichert werden [vgl. 6].

3.2.3 Borrows-Wheeler-Transformation

Die Borrows-Wheeler-Transformation (BWT) wurde 1994 ursprünglich unter dem Namen „A Block-Sorting Lossless Data Compression“ bekannt. Mit ihrer Hilfe kann man fast beliebig lange Kontexte für die Vorhersage eines folgenden Symbols verwenden, ohne dafür eine Art Lexikon anlegen oder ein Modell aufbauen zu müssen. Eine Kompression, die dieses Verfahren benutzt, läuft in zwei Schritten ab:

Mit Hilfe der Borrows-Wheeler-Transformation werden die zu komprimierenden Daten umsortiert und anschließend kodiert.

Im Unterschied zu den zuerst beschriebenen Verfahren verarbeitet dieser Algorithmus die Eingabedaten nicht zeichenweise sondern blockweise. Jeder Datenblock wird folgendermaßen bearbeitet:

Zuerst wird aus dem Block eine quadratische Matrix erstellt, wobei die erste Zeile die originale Zeichenkette enthält. Jede weitere Zeile stellt die gleiche Kette um ein Zeichen nach links verschoben dar. Dabei „herausfallende“ Zeichen werden am Ende eingefügt. Da die Matrix genauso viele Zeilen wie Spalten hat, kommt jede mögliche Rotation der ursprünglichen Zeichenkette genau einmal vor.

Danach sortiert der Borrows-Wheeler-Algorithmus die Zeilen der Matrix alphabetisch. Die Erste Spalte wird mit „F“ (first) benannt, die letzte mit „L“ (last). Die Besonderheit der letzten Spalte ist, dass ein Zeichen aus ihr dasjenige ist, das der in der gleichen Zeile beginnenden Zeichenkette vorangeht. Die Zeichenketten in den einzelnen Zeilen betrachtet man als Kontext für das Zeichen in Spalte L. Dadurch, dass die Kontexte alphabetisch sortiert sind, stehen ähnliche nahe beieinander. Da ähnlichen Kontexten aber auch meist der gleiche Buchstabe vorangeht, gibt es die Tendenz, dass auch in Spalte L gleiche Buchstaben nahe beieinander stehen.

Die Ausgabe des Borrows-Wheeler-Algorithmus’ besteht nun aus der L-Spalte sowie der Position der Ausgangszeichenkette in der sortierten Matrix. Aus diesen Angaben wird nun ein Transformationsvektor berechnet.

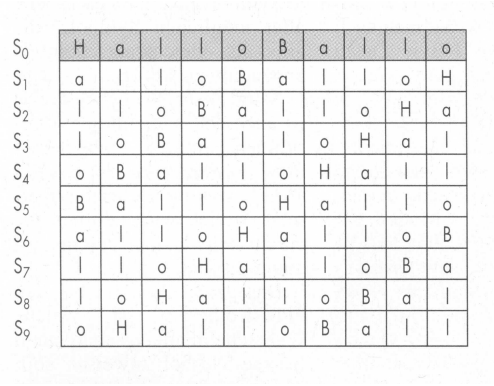
Dekomprimiert wird, in dem zuerst die Spalte F aus der Spalte L durch ein Sortieren der Einträge rekonstruiert wird. Aus diesen Angaben wird nun ein Transformationsvektor T berechnet, ein eindimensionales Array, das genau so groß ist wie die Ausgangszeichenkette und das für jede Zeile der Matrix einen Eintrag aufweist. Für eine gegebene Zeile i der Matrix, welche die Zeichenkette S_j enthält, steht dabei im Feld T[i] des Vektors der Index derjenigen Zeile, in der sich die Zeichenkette S_{j+1} befindet. Dieser Vektor wird aus den Spalten F und L berechnet, in dem das erste Zeichen der Zeichenkette einer Zeile in der Spalte L an der jeweiligen Stelle des Transformationsvektors gespeichert wird.

Um die Originaldaten zurückzugewinnen, werden mit Hilfe des Transformationsvektors, der Spalte F und der Position der Zeile, die S_0 enthielt, die ursprünglichen Daten rekonstruiert.

3.2.3.1 Beispiel:

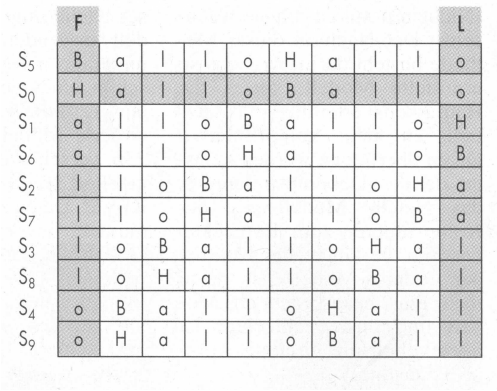
Um dieses Verfahren etwas mehr zu verdeutlichen, wird es im Folgenden an einer Zeichenkette „HalloBallo“ beschrieben [Beispiel aus 6]:

Zuerst wird aus dem Block eine quadratische Matrix (hier 10x10 Zeichen) erstellt, wobei die erste Zeile die originale Zeichenkette enthält, und die anderen Zeilen die Kette jeweils nach links verschoben.



S_0	H	a	l	l	o	B	a	l	l	o
S_1	a	l	l	o	B	a	l	l	o	H
S_2	l	l	o	B	a	l	l	o	H	a
S_3	l	o	B	a	l	l	o	H	a	l
S_4	o	B	a	l	l	o	H	a	l	l
S_5	B	a	l	l	o	H	a	l	l	o
S_6	a	l	l	o	H	a	l	l	o	B
S_7	l	l	o	H	a	l	l	o	B	a
S_8	l	o	H	a	l	l	o	B	a	l
S_9	o	H	a	l	l	o	B	a	l	l

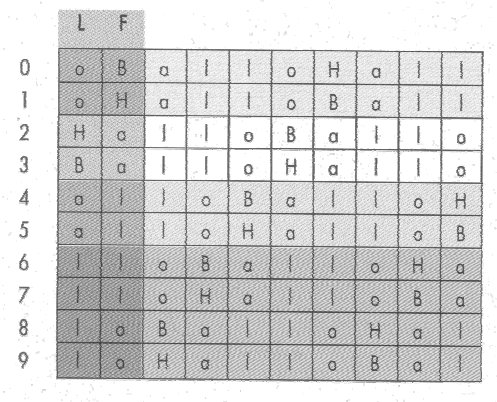
Als nächstes sortiert der Borrows-Wheeler-Algorithmus die Zeilen der Matrix alphabetisch. Die Erste Spalte wird mit „F“ benannt, die letzte mit „L“.



	F									L
S_5	B	a	l	l	o	H	a	l	l	o
S_0	H	a	l	l	o	B	a	l	l	o
S_1	a	l	l	o	B	a	l	l	o	H
S_6	a	l	l	o	H	a	l	l	o	B
S_2	l	l	o	B	a	l	l	o	H	a
S_7	l	l	o	H	a	l	l	o	B	a
S_3	l	o	B	a	l	l	o	H	a	l
S_8	l	o	H	a	l	l	o	B	a	l
S_4	o	B	a	l	l	o	H	a	l	l
S_9	o	H	a	l	l	o	B	a	l	l

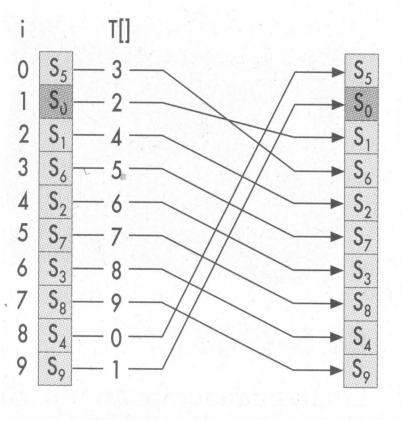
Ausgegeben und kodiert wird nun die letzte Spalte und die Position der Ausgangs-zeichenkette der sortierten Matrix, hier „1“. Dieses Ergebnis wird nun als nächstes kodiert (siehe dazu Kapitel 3.2.3.2).

Zur Dekomprimierung werden die übertragenen Daten nun wieder dekodiert. Danach wird der Transformationsvektor berechnet, indem das erste Zeichen der Zeichenkette einer Zeile in der Spalte L an der jeweiligen Stelle des Transformationsvektors gespeichert wird. In Zeile 0 steht hier als erstes Zeichen (Spalte F) ein „B“. Die Position des „B“ in der Spalte L ist die 3. Also steht im Transformationsvektor an der Stelle T[0] eine 3.



	L	F								
0	o	B	a	l	l	o	H	a	l	l
1	o	H	a	l	l	o	B	a	l	l
2	H	a	l	l	o	B	a	l	l	o
3	B	a	l	l	o	H	a	l	l	o
4	a	l	l	o	B	a	l	l	o	H
5	a	l	l	o	H	a	l	l	o	B
6	l	l	o	B	a	l	l	o	H	a
7	l	l	o	H	a	l	l	o	B	a
8	l	o	B	a	l	l	o	H	a	l
9	l	o	H	a	l	l	o	B	a	l

Um die Originaldaten zurückzugewinnen, werden mit Hilfe des Transformationsvektors, der Spalte F und der Position der Zeile, die S_0 enthielt, die ursprünglichen Daten rekonstruiert. Hier wurde die Spalte L (ooHBaallll) mit der Position 1 übertragen. Daraus konstruiert man die Spalte F (BHaalllloo) und daraus den Transformationsvektor T = [3,2,4,5,6,7,8,9,0,1].



Position 1 bedeutet, dass hier angefangen wird, den Text zu rekonstruieren. An dieser Stelle steht in F das „H“ und T[1] = 2. An zweiter Stelle in F steht das „a“, T[2] = 4. In der vierten Zeile ist der Wert der Spalte F ein „l“, der Vektor verweist auf Zeile 6. Diese beinhaltet ebenfalls ein „l“ und T[6] = 8. An der achten Stelle steht in F ein „o“, als Nachfolger wird die Zeile 0 angegeben. Dies wird so lange weiter durchgeführt, bis in dem Transformationsvektor wieder auf den Anfang verwiesen wird. In diesem Beispiel verweist der Vektor in T[9] auf die 1.

3.2.3.2 Kodierung

Da die Daten nach der Anwendung des Borrows-Wheeler-Algorithmus’ dazu neigen, dass gleiche Symbole in der Ausgabe (Spalte L) in Gruppen zusammenstehen, die Ausgabe also aus langen Folgen gleicher Buchstaben besteht, die ab und zu durch andere Zeichen unterbrochen werden, bietet es sich an, die so vorliegenden Daten mit Lauflängenkodierung (Run Length Encoding (RLE)) zu kodieren. Hierbei werden statt einer Folge von gleichen Symbolen nur ein spezieller RLE-Code, das Symbol selbst und die Länge der Folge gespeichert. Mit Hilfe der Borrows-Wheeler-Transformation und der Lauflängenkodierung erzielt man so bereits erstaunliche Kompressionsraten [vergleiche 6, S. 200].

4 Zusammenfassung

Um die aufgeführten Verfahren zur Datenreduktion zusammenzufassen, möchte ich sie in zwei grobe Kategorien unterteilen:

Eine Kategorie manipuliert die gegebene Daten, in dem sie diese zusammenfasst (Aggregation), Teile der Daten löscht oder ignoriert (Mengenreduktion), ursprüngliche Daten durch alternative, kleinere Datenrepräsentationen ersetzt (numerische Reduktion) oder die Anzahl der Werte ihrer Attribute verändert (Diskretisierung). Die andere Kategorie stellt die ursprüngliche Daten einfach in einer anderen Art und Weise dar, speichert sie anders ab oder gibt sie in veränderter Form weiter.

Beide Arten der Datenbehandlung reduzieren die Größe der gegebenen Daten so, dass sie einfacher, schneller oder überhaupt zu analysieren sind. Dabei werden die Verfahren der ersten Kategorie dann, wenn anschließend eine Analyse oder Auswertung der Daten erfolgen soll. Dies geht dann schneller oder ist überhaupt erst möglich, da sie nun leichter zu handhaben sind. Die Verfahren der zweiten Kategorie werden meistens verwendet, wenn die Daten verschickt oder platzsparender abgespeichert werden sollen.

Man sollte vielleicht noch beachten, dass der Informationsgehalt der reduzierten Daten für eine bestimmte Analyse oder Weiterverarbeitung der Daten durch die Reduktion nicht wesentlich geändert hat.

Literaturangaben

[1] Han, J.; Kamber, M.;
Data Mining: Concepts and Techniques;
Morgan Kaufmann, San Francisco, California, 2001

[2] Schöning, Uwe;
Algorithmen kurz gefasst, Spektrum Akademischer Verlag, Heidelberg 1997

[3] Harry Mucksch, Wolfgang Behme (Hrsg.);
Das Data-Warehouse-Konzept; Gabler Verlag, Wiesbaden 1996

[4] Jan Holthuis;
Der Aufbau von Data Warehouse-Systemen,
Konzepte – Datenmodellierung – Vorgehen; Deutscher Universitätsverlag, Wiesbaden 1998

[5] Markus Helfert;
Planung und Messung der Datenqualität in Data-Warehouse-Systemen; Dissertation Nr. 2648 Universität St. Gallen; Logos Verlag, Bamberg 2002

[6] Michael Tamm,
„Packen wie noch nie – Datenkompression mit dem BWT-Algorithmus“;
c’t – magazin für computer technik Heft 16 / 2000; Hannover 2000

[7] video-4-all, Glossar
http://www.video-4-all.info/glossar/wavelets.html

[8] C. Rauch,
Bildkompression mit der Wavelet-Transformation, Hauptseminar Multimediale Informationsverarbeitung, TU München 1999

[9] M.Zeller,
Signalverarbeitung mit Wavelets c’t – magazin für computer technik
Heft 11 / 1994; Hannover 1994

[10] J. Ansorg,
WAVELETS – Transformationskodierung,
JPEG 2000 - ISO / IEC-Standard 15444-1, Jena 2002
http://www.fh-jena.de/contrib/fb/et/personal/ansorg/ftp/wavelet/wavelet.htm

[11] M. Clemens;
Wavelet Tutorial; Kaiserslautern, 1999;
http://nt.eit.uni-kl.de/wavelet/index.html

[12] C. Esser,
Wavelet Transformation von Standbildern, Studienarbeit,
Universität Mannheim 2001;
http://www-mm.informatik.uni-mannheim.de/veranstaltungen
/animation/multimedia/wavelet/documentation/WaveletDemo.pdf

[13] R. Brause
Neuronale Netze; Teubner-Verlag Stuttgart, 1995