4. Assoziationsregeln

Inhalt dieses Kapitels

4.1 Einleitung

Transaktionsdatenbanken, Warenkorbanalyse

4.2 Einfache Assoziationsregeln

Grundbegriffe, Aufgabenstellung, Apriori-Algorithmus, Hashbäume, Interessantheit von Assoziationsregeln, Einbezug von Constraints

4.3 Hierarchische Assoziationsregeln

Motivation, Grundbegriffe, Algorithmen, Interessantheit

4.4 Quantitative Assoziationsregeln

Motivation, Grundidee, Partitionierung numerischer Attribute, Anpassung des Apriori-Algorithmus, Interessantheit

4.1 Einleitung

Motivation

{Butter, Brot, Milch, Zucker}
{Butter, Mehl, Milch, Zucker}
{Butter, Eier, Milch, Salz}
{Eier}
{Butter, Mehl, Milch, Salz, Zucker}

Transaktionsdatenbank

Warenkorbanalyse

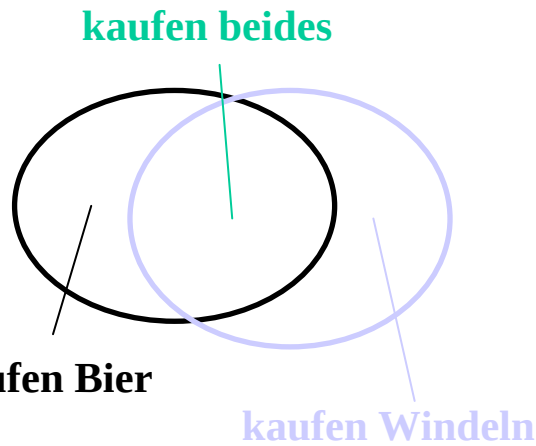
- Welche Artikel werden häufig miteinander gekauft?
- Anwendungen
  - Verbesserung des Laden-Layouts
  - Cross Marketing
  - gezielte Attached Mailings/Add-on Sales

4.1 Einleitung

Assoziationsregeln

Regeln der Form

"Rumpf $\rightarrow$ Kopf [support, confidence]"



Beispiele

kauft(X, "Windeln") $\rightarrow$ kauft(X, "Bier") [0.5%, 60%]

major(X, "CS") $\wedge$ takes(X, "DB") $\rightarrow$ grade(X, "A") [1%, 75%]

98% aller Kunden, die Reifen und Autozubehör kaufen,
bringen ihr Auto auch zum Service

4.2 Einfache Assoziationsregeln

Grundbegriffe [Agrawal & Srikant 1994]

- *Items* $I = \{i_1, ..., i_m\}$ eine Menge von Literalen
- *Itemset X*: Menge von Items $X \subseteq I$
- *Datenbank D*: Multimenge von *Transaktionen T* mit $T \subseteq I$
- *T enthält X*: $X \subseteq T$

- Eine Datenbank ist also (in der Sprache der Begriffsanalyse) ein formaler Kontext, bei dem
  - die Gegenstände die Transaktionen sind,
  - die Merkmale die Itemsets,
  - und die binäre Relation die ‚enthält'-Beziehung ist.
- Bemerke: es können (wie in der Begriffsanalyse auch) mehrere Transaktionen auftreten, die exakt die gleichen Items enthalten.

4.2 Einfache Assoziationsregeln

Fortsetzung Grundbegriffe

•Items in Transaktionen oder Itemsets sind lexikographisch sortiert:

Itemset $X = (x_1, x_2, ..., x_k)$, wobei $x_1 \leq x_{2 \leq} ... \leq x_k$

• *Länge des Itemsets*: Anzahl der Elemente in einem Itemset

• *k*-Itemset: ein Itemset der Länge *k*

4.2 Einfache Assoziationsregeln

Grundbegriffe

- *Support der Menge X in D* := Anteil der Transaktionen in *D*, die *X* enthalten. (d.h. wie im vorigen Kapitel definiert.)
- *Assoziationsregel*: Implikation der Form $X \Rightarrow Y$,

  wobei gilt: $X \subseteq I$, $Y \subseteq I$ und $X \cap Y = \varnothing$
- *Support s einer Assoziationsregel* $X \Rightarrow Y$ *in D*:

  Support von $X \cup Y$ in *D*
- *Konfidenz c einer Assoziationsregel* $X \Rightarrow Y$ *in D*:

  Anteil der Transaktionen, die die Menge *Y* enthalten, in der Teilmenge aller Transaktionen aus *D*, welche die Menge *X* enthalten
- *Aufgabenstellung*: bestimme alle Assoziationsregeln, die in *D* einen Support $\geq$ *minsup* und eine Konfidenz $\geq$ *minconf* besitzen

4.2 Einfache Assoziationsregeln

Beispiel

TransaktionsID	Items
2000	A,B,C
1000	A,C
4000	A,D
5000	B,E,F

minsup = 50%,
minconf = 50%

Support

(A): 75%, (B), (C): 50%, (D), (E), (F): 25%,

(A, C): 50%, (A, B), (A, D), (B, C), (B, E), (B, F), (E, F): 25%

Assoziationsregeln

$A \Rightarrow C$ (Support = 50%, Konfidenz = 66.6%)

$C \Rightarrow A$ (Support = 50%, Konfidenz = 100%)

4.2 Einfache Assoziationsregeln

Methode

1. Bestimmung der häufig auftretenden Itemsets in der Datenbank

   häufig auftretende Itemsets (*Frequent Itemsets*): Support ≥ *minsup*

   „naiver“ Algorithmus:

   zähle die Häufigkeit aller k-elementigen Teilmengen von I

   ineffizient, da $\binom{m}{k}$ solcher Teilmengen

2. Generierung der Assoziationsregeln aus den Frequent Itemsets

   Itemset X häufig

   A Teilmenge von X

   $A \Rightarrow (X - A)$ hat minimalen Support

4.2 Bestimmung der häufig auftretenden Itemsets

Grundlagen

Monotonie-Eigenschaft

Jede Teilmenge eines häufig auftretenden Itemsets ist selbst auch häufig.

Vorgehen

- zuerst die einelementigen Frequent Itemsets bestimmen, dann die zweielementigen und so weiter
- Finden von k+1-elementigen Frequent Itemsets:

  nur solche k+1-elementigen Itemsets betrachten, für die alle k-elementigen Teilmengen häufig auftreten
- Bestimmung des Supports durch Zählen auf der Datenbank (ein Scan)

4.2 Bestimmung der häufig auftretenden Itemsets

C_k: die zu zählenden Kandidaten-Itemsets der Länge k
L_k: Menge aller häufig vorkommenden Itemsets der Länge k

```
Apriori(I, D, minsup)
L_1 := {frequent 1-Itemsets aus I};
k := 2;
while L_{k-1} ≠ ∅ do
  C_k := AprioriKandidatenGenerierung(L_{k-1});
  for each Transaktion T ∈ D do
     CT := Subset(C_k, T); // alle Kandidaten aus C_k, die in
       der Transaktion T enthalten sind;
     for each Kandidat c ∈ CT do c.count++;
  L_k := {c ∈ C_k | (c.count / |D|) ≥ minsup};
  k++;
return ∪_k L_k;
```

4.2 Bestimmung der häufig auftretenden Itemsets

Kandidatengenerierung

Anforderungen an Kandidaten-Itemsets C_k

- Obermenge von L_k
- wesentlich kleiner als die Menge *aller* k-elementigen Teilmengen von I

Schritt 1: Join

k-1-elementige Frequent Itemsets p und q

p und q werden miteinander verbunden, wenn sie in den ersten k–1 Items übereinstimmen

$p \in L_{k\text{-}1}$ (1, 2, 3)

(1, 2, 3, 4) $\in C_k$

$q \in L_{k\text{-}1}$ (1, 2, 4)

4.2 Bestimmung der häufig auftretenden Itemsets

Kandidatengenerierung

Schritt 2: Pruning

entferne alle Kandidaten-Itemsets, die eine k–1-elementige Teilmenge enthalten, die nicht zu $L_{k\text{-}1}$ gehört

Beispiel

L_3 = {(1 2 3), (1 2 4), (1 3 4), (1 3 5), (2 3 4)}

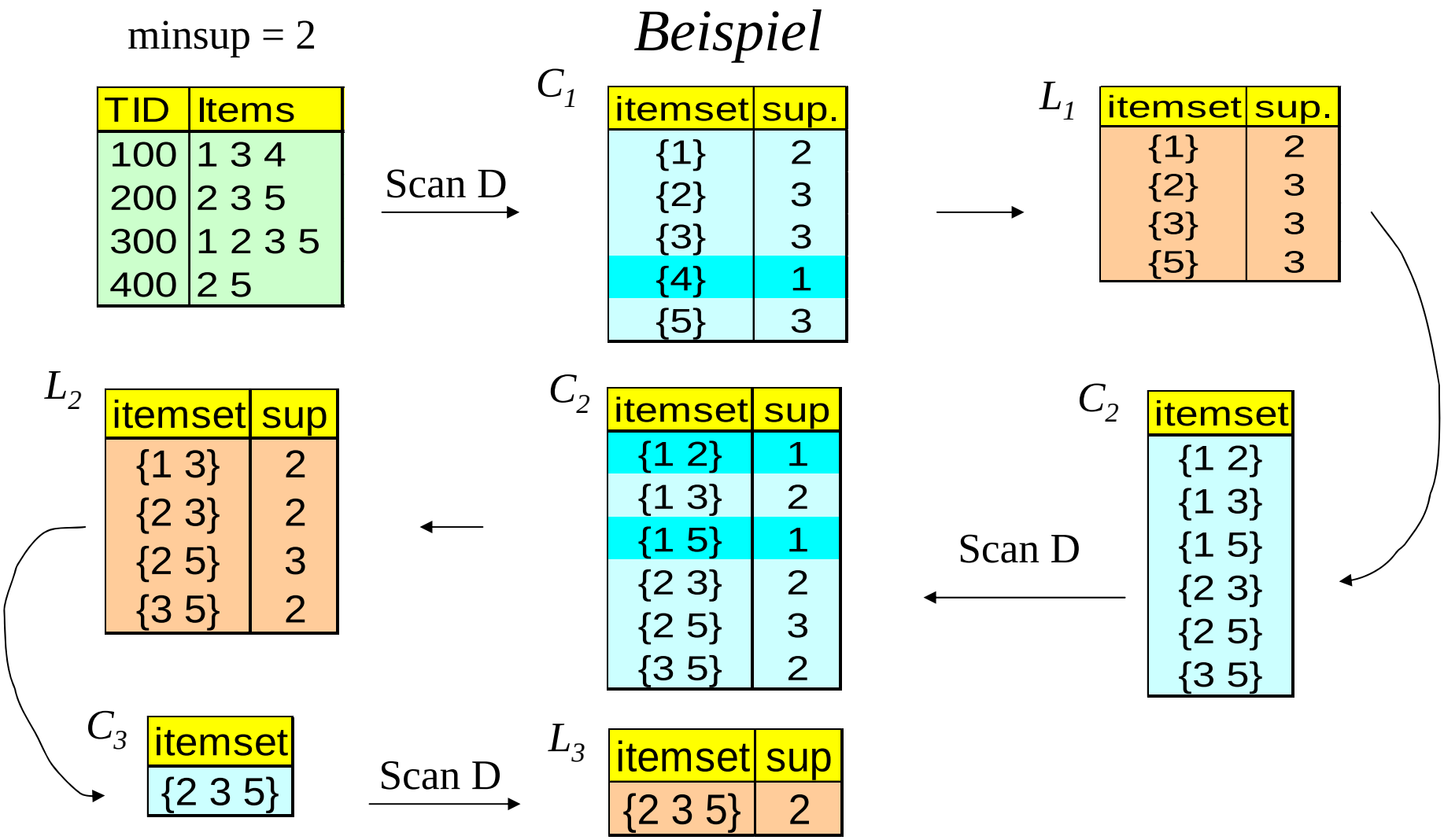
nach dem Join-Schritt: Kandidaten = {(1 2 3 4), (1 3 4 5)}

im Pruning-Schritt:

lösche (1 3 4 5)

⇒ C_4 = {(1 2 3 4)}

4.2 Bestimmung der häufig auftretenden Itemsets



4.2 Bestimmung der häufig auftretenden Itemsets

Effiziente Unterstützung der Subset-Funktion

- `Subset(`C_k`,`T`)`

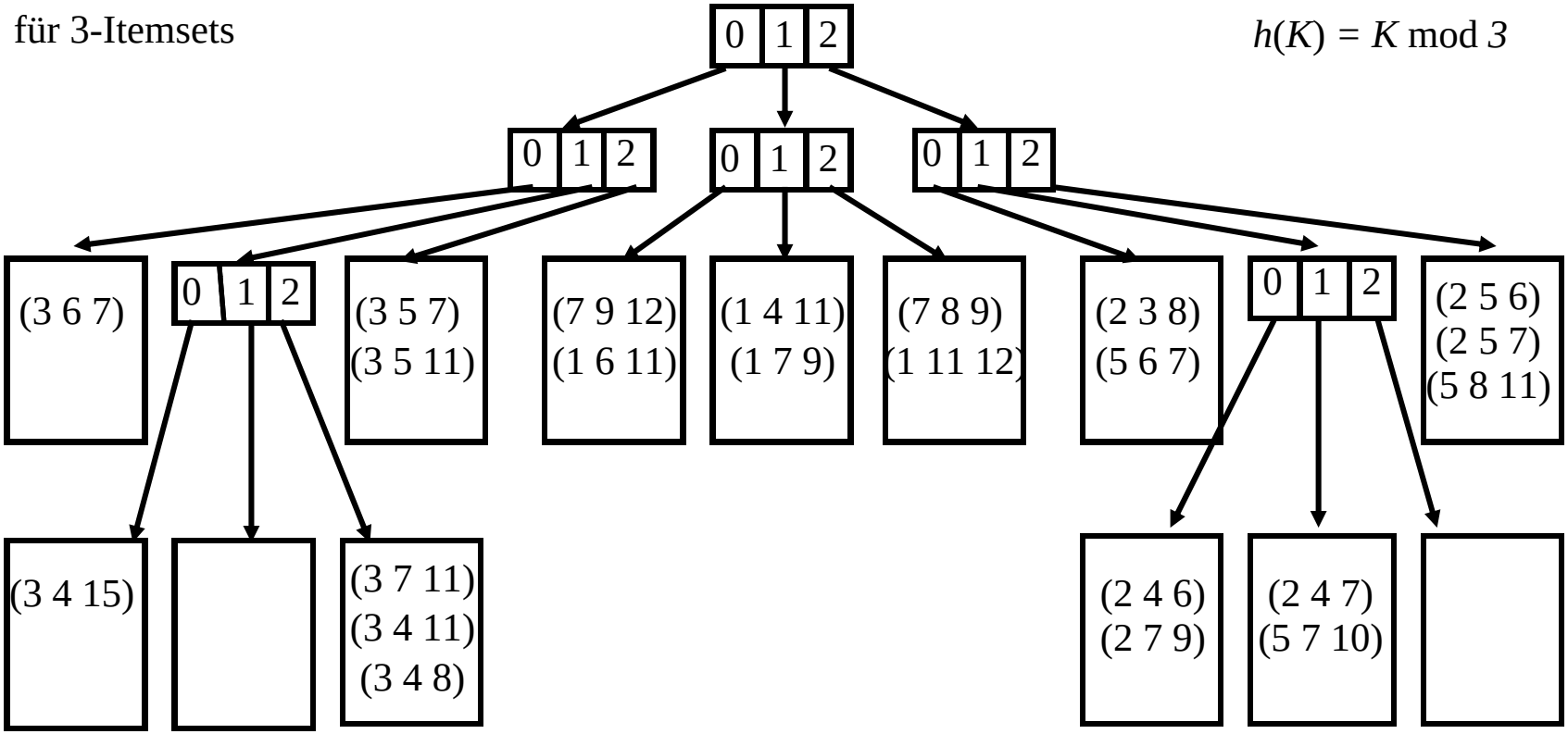
  ⇒ alle Kandidaten aus C_k, die in der Transaktion T enthalten sind

- Probleme
  - sehr viele Kandidaten-Itemsets
  - eine Transaktion kann viele Kandidaten enthalten
- Hashbaum zur Speicherung von C_k
  - *Blattknoten* enthält Liste von Itemsets (mit Häufigkeiten)
  - *innerer Knoten* enthält Hashtabelle

    jedes Bucket auf Level d verweist auf Sohnknoten des Levels d+1
  - *Wurzel* befindet sich auf Level 1

4.2 Bestimmung der häufig auftretenden Itemsets

Beispiel



4.2 Bestimmung der häufig auftretenden Itemsets

Hashbaum

Suchen eines Itemsets

- starte bei der Wurzel
- auf Level *d*: wende die Hashfunktion *h* auf das *d*-te Item des Itemsets an

Einfügen eines Itemsets

- suche das entsprechende Blatt und füge Itemset ein
- beim Overflow:
  - Umwandlung des Blattknotens in inneren Knoten
  - Verteilung seiner Einträge gemäß der Hashfunktion auf die neuen Blattknoten

4.2 Bestimmung der häufig auftretenden Itemsets

Hashbaum

Suchen aller Kandidaten, die in $T = (t_1\ t_2 \ldots t_m)$ enthalten sind

- bei der Wurzel

  bestimme die Hashwerte für jedes Item in T

  suche weiter in den resultierenden Sohnknoten

- bei einem inneren Knoten auf Level d

  (den man durch Hashing nach t_i erreicht hat)

  bestimme die Hashwerte und suche weiter für jedes Item t_k mit $k > i$

- bei einem Blattknoten

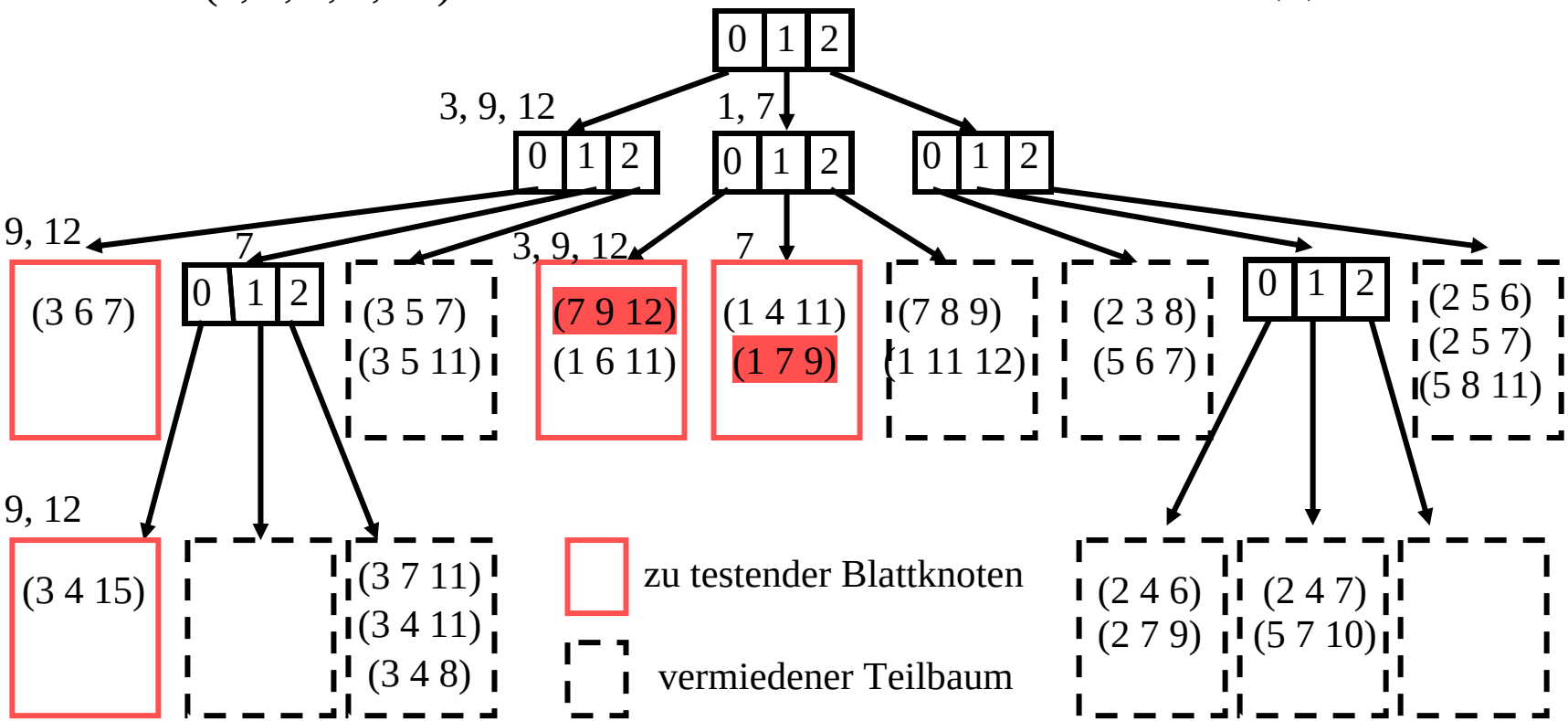
  teste für die enthaltenen Itemsets, ob sie in der Transaktion T vorkommen

4.2 Bestimmung der häufig auftretenden Itemsets

Beispiel

Transaktion (1, 3, 7, 9, 12)

$h(K) = K \bmod 3$



4.2 Bestimmung der häufig auftretenden Itemsets

Methoden der Effizienzverbesserung

Zählen des Supports mit Hashtabelle [Park, Chen & Yu 1995]

- Hashtabelle statt Hashbaum zum Bestimmen des Supports
- *k*-Itemset, dessen Bucket einen Zähler < minsup hat, kann nicht häufig auftreten

  ⇒ effizienterer Zugriff auf Kandidaten, aber ungenaue Zählung

Reduktion der Transaktionen [Agrawal & Srikant 1994]

- eine Transaktion, die keinen Frequent *k*-Itemset enthält, wird nicht mehr benötigt
- entferne solche Transaktionen für weitere Phasen aus der Datenbank

  ⇒ effizienterer Datenbank-Scan, aber vorher neues Schreiben der Datenbank

4.2 Bestimmung der häufig auftretenden Itemsets

Methoden der Effizienzverbesserung

Partitionierung der Datenbank [Savasere, Omiecinski & Navathe 1995]

- ein Itemset ist nur dann häufig, wenn er in mindestens einer Partition häufig ist
- bilde hauptspeicherresidente Partitionen der Datenbank

⇒ viel effizienter auf Partitionen, aber aufwendige Kombination der Teilergebnisse

Sampling [Toivonen 1996]

- Anwendung des gesamten Algorithmus auf ein Sample
- Zählen der gefundenen häufigen Itemsets auf der gesamten Datenbank
- Festellen evtl. weiterer Kandidaten und Zählen auf der gesamten Datenbank

4.2 Bestimmung der Assoziationsregeln

Methode

- häufig vorkommender Itemset X
- für jede Teilmenge A von X die Regel $A \Rightarrow X \setminus A$ bilden
- Regeln streichen, die nicht die minimale Konfidenz haben
- Berechnung der Konfidenz einer Regel $A \Rightarrow X \setminus A$

$$konfidenz(A \Rightarrow (X - A)) = \frac{support(X)}{support(A)}$$

- Speicherung der Frequent Itemsets mit ihrem Support in einer Hashtabelle

⇒ keine Datenbankzugriffe

4.2 Interessantheit von Assoziationsregeln

Motivation

Aufgabenstellung

- Daten über das Verhalten von Schülern in einer Schule mit 5000 Schülern

Beispiel

- Itemsets mit Support:

  60% der Schüler spielen Fußball, 75% der Schüler essen Schokoriegel

  40% der Schüler spielen Fußball *und* essen Schokoriegel

- Assoziationsregeln:

  „Spielt Fußball“ ⇒ „Ißt Schokoriegel“, Konfidenz = 67%

  TRUE ⇒ „Ißt Schokoriegel“, Konfidenz = 75%

Fußball spielen und Schokoriegel essen sind *negativ korreliert*

4.2 Interessantheit von Assoziationsregeln

Aufgabenstellung

- Herausfiltern von irreführenden Assoziationsregeln
- Bedingung für eine Regel $A \Rightarrow B$

$$\frac{P(A \wedge B)}{P(A)} > P(B) - d$$

für eine geeignete Konstante $d > 0$

- Maß für die „Interessantheit“ einer Regel

$$\frac{P(A \wedge B)}{P(A)} - P(B)$$

- Je größer der Wert für eine Regel ist, desto interessanter ist der durch die Regel ausgedrückte Zusammenhang zwischen *A* und *B*.

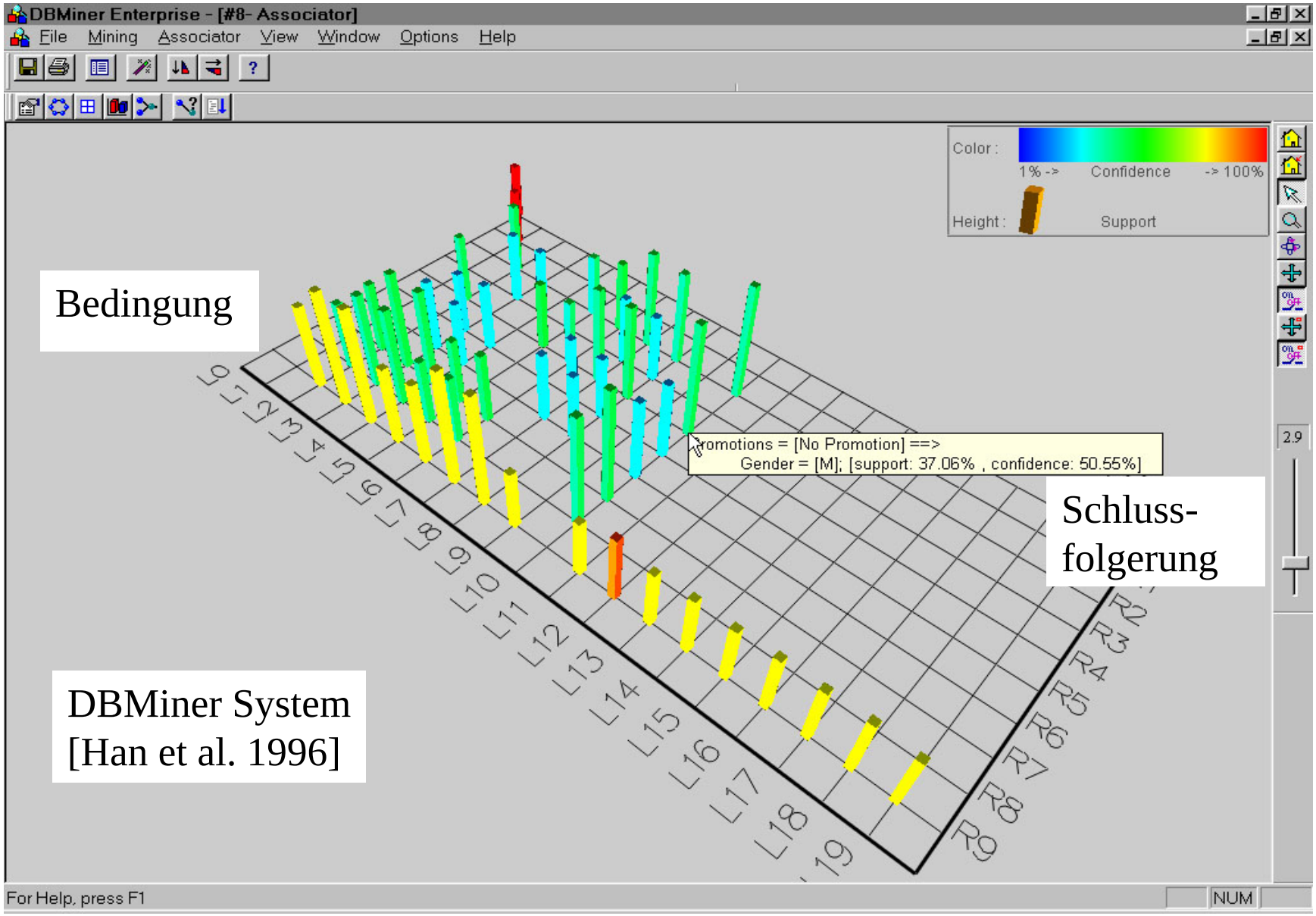
4.2 Präsentation von Assoziationsregeln

	Body	Implies	Head	Supp (%)	Conf (%)	F	G	H	I
1	cost(x) = '0.00~1000.00'	==>	revenue(x) = '0.00~500.00'	28.45	40.4				
2	cost(x) = '0.00~1000.00'	==>	revenue(x) = '500.00~1000.00'	20.46	29.05				
3	cost(x) = '0.00~1000.00'	==>	order_qty(x) = '0.00~100.00'	59.17	84.04				
4	cost(x) = '0.00~1000.00'	==>	revenue(x) = '1000.00~1500.00'	10.45	14.84				
5	cost(x) = '0.00~1000.00'	==>	region(x) = 'United States'	22.56	32.04				
6	cost(x) = '1000.00~2000.00'	==>	order_qty(x) = '0.00~100.00'	12.91	69.34				
7	order_qty(x) = '0.00~100.00'	==>	revenue(x) = '0.00~500.00'	28.45	34.54				
8	order_qty(x) = '0.00~100.00'	==>	cost(x) = '1000.00~2000.00'	12.91	15.67				
9	order_qty(x) = '0.00~100.00'	==>	region(x) = 'United States'	25.9	31.45				
10	order_qty(x) = '0.00~100.00'	==>	cost(x) = '0.00~1000.00'	59.17	71.86				
11	order_qty(x) = '0.00~100.00'	==>	product_line(x) = 'Tents'	13.52	16.42				
12	order_qty(x) = '0.00~100.00'	==>	revenue(x) = '500.00~1000.00'	19.67	23.88				
13	product_line(x) = 'Tents'	==>	order_qty(x) = '0.00~100.00'	13.52	98.72				
14	region(x) = 'United States'	==>	order_qty(x) = '0.00~100.00'	25.9	81.94				
15	region(x) = 'United States'	==>	cost(x) = '0.00~1000.00'	22.56	71.39				
16	revenue(x) = '0.00~500.00'	==>	cost(x) = '0.00~1000.00'	28.45	100				
17	revenue(x) = '0.00~500.00'	==>	order_qty(x) = '0.00~100.00'	28.45	100				
18	revenue(x) = '1000.00~1500.00'	==>	cost(x) = '0.00~1000.00'	10.45	96.75				
19	revenue(x) = '500.00~1000.00'	==>	cost(x) = '0.00~1000.00'	20.46	100				
20	revenue(x) = '500.00~1000.00'	==>	order_qty(x) = '0.00~100.00'	19.67	96.14				
21									
22									
23	cost(x) = '0.00~1000.00'	==>	revenue(x) = '0.00~500.00' AND order_qty(x) = '0.00~100.00'	28.45	40.4				
24	cost(x) = '0.00~1000.00'	==>	revenue(x) = '0.00~500.00' AND order_qty(x) = '0.00~100.00'	28.45	40.4				
25	cost(x) = '0.00~1000.00'	==>	revenue(x) = '500.00~1000.00' AND order_qty(x) = '0.00~100.00'	19.67	27.93				
26	cost(x) = '0.00~1000.00'	==>	revenue(x) = '500.00~1000.00' AND order_qty(x) = '0.00~100.00'	19.67	27.93				
27	cost(x) = '0.00~1000.00' AND order_qty(x) = '0.00~100.00'	==>	revenue(x) = '500.00~1000.00'	19.67	33.23				

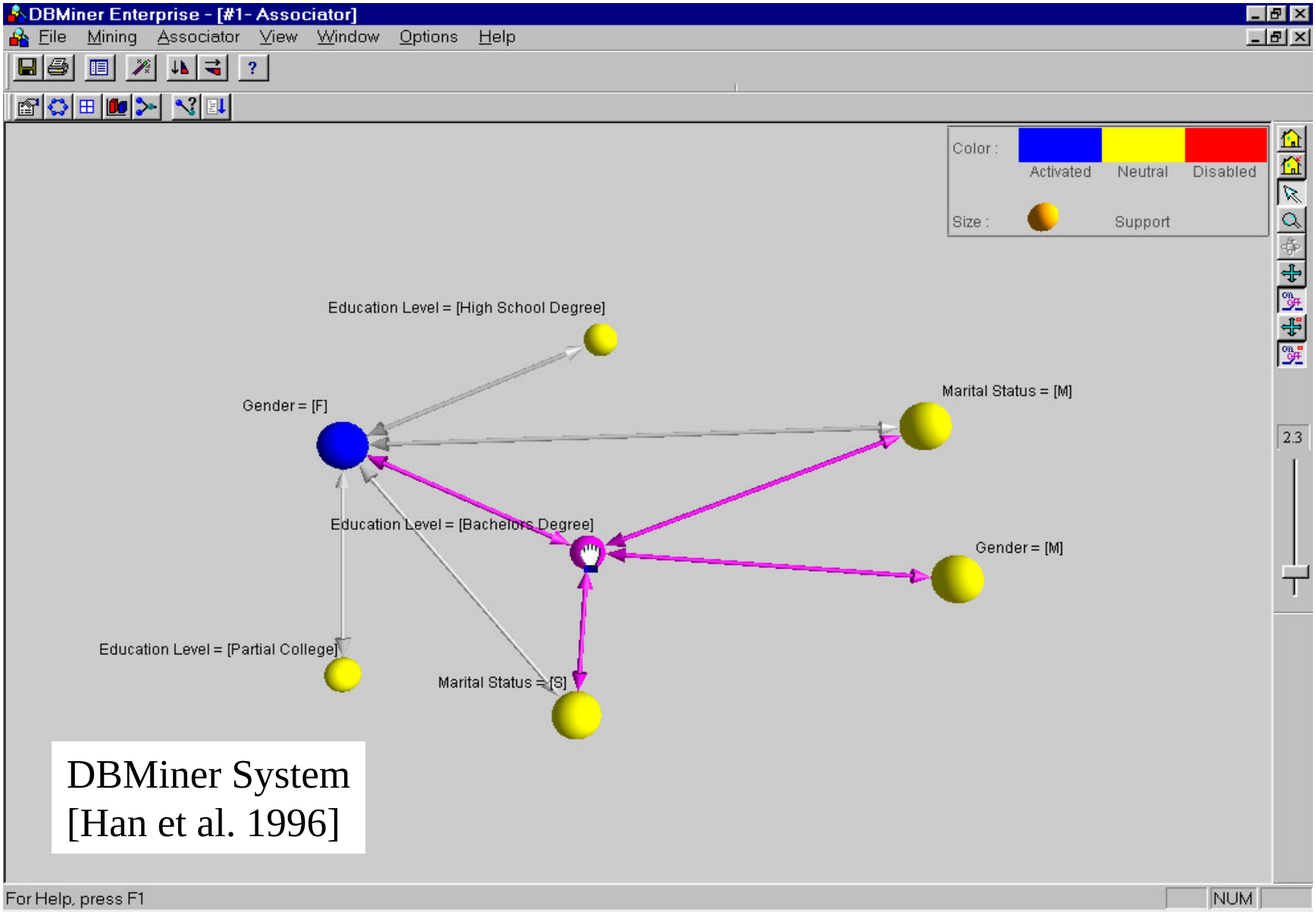
Sheet1

DBMiner System
[Han et al. 1996]

4.2 Präsentation von Assoziationsregeln



4.2 Präsentation von Assoziationsregeln



DBMiner System
[Han et al. 1996]

4.2 Constraints für Assoziationsregeln

Motivation

- zu viele Frequent Item Sets

  Effizienzproblem

- zu viele Assoziationsregeln

  Evaluationsproblem

- manchmal Constraints apriori bekannt

  „nur Assoziationsregeln mit Produkt A aber ohne Produkt B“

  „nur Assoziationsregeln mit Gesamtpreis > 100 der enthaltenen Produkte“

  ⇒ Constraints an die Frequent Itemsets

4.2 Constraints für Assoziationsregeln

Typen von Constraints

[Ng, Lakshmanan, Han & Pang 1998]

Domain Constraint

- $S \theta v$, $\theta \in \{ =, \neq, <, \leq, >, \geq \}$, z.B. *S.Preis < 100*
- $v \theta S$, $\theta \in \{\in, \notin\}$, z.B. *Snacks* $\notin$ *S.Typ*
- $V \theta S$ oder $S \theta V$, $\theta \in \{ \subseteq, \subset, \not\subset, =, \neq \}$, z.B. {*Snacks, Weine*} $\subseteq$ *S.Typ*

Aggregations Constraint

$agg(S)\ \theta\ v$ mit

- $agg \in \{min, max, sum, count, avg\}$
- $\theta \in \{ =, \neq, <, \leq, >, \geq \}$

z.B. $count(S_1.Typ) = 1$,
$avg(S_2.Preis) > 100$

4.2 Constraints für Assoziationsregeln

Anwendung der Constraints

Bei der Bestimmung der Assoziationsregeln

- löst das Evaluationsproblem
- nicht aber das Effizienzproblem

Bei der Bestimmung der häufig auftretenden Itemsets

- löst evtl. auch das Effizienzproblem
- Frage bei der Kandidatengenerierung:

welche Itemsets kann man mit Hilfe der Constraints ausschließen?

4.2 Constraints für Assoziationsregeln

Anti-Monotonie

Definition

Wenn eine Menge *S* ein anti-monotones Constraint *C* verletzt,
dann verletzt auch jede Obermenge von *S* dieses Constraint .

Beispiele

- *sum(S.Preis)* $\leq$ *v* ist anti-monoton
- *sum(S.Preis)* $\geq$ *v* ist *nicht* anti-monoton
- *sum(S.Preis)* = *v* ist *teilweise* anti-monoton

Anwendung

➡ baue anti-monotone Constraints in die Kandidatengenerierung ein

4.2 Constraints für Assoziationsregeln

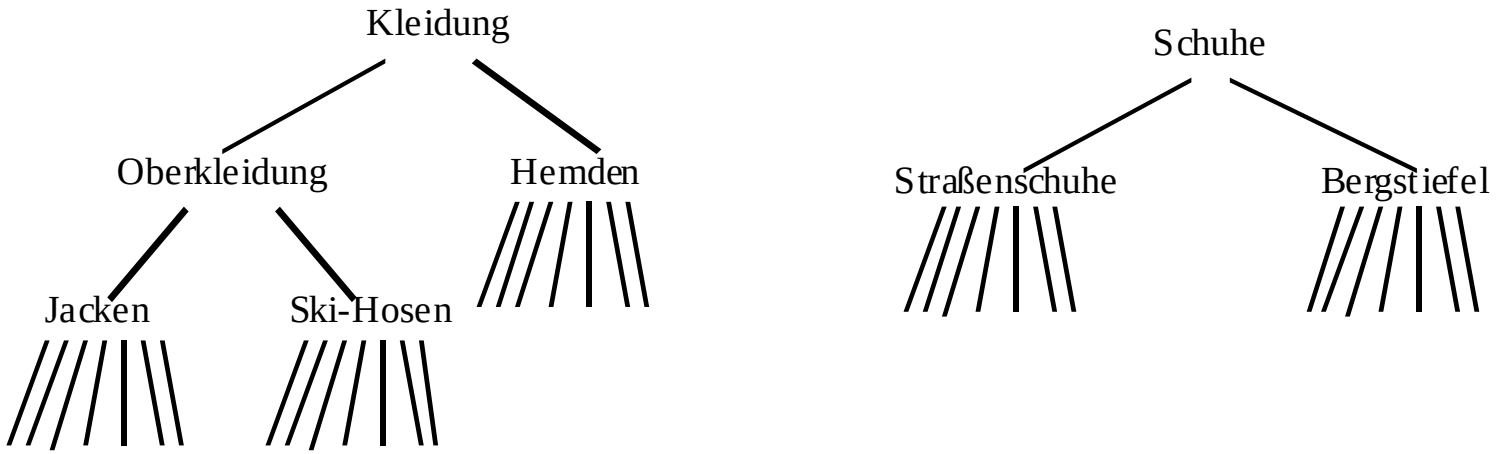
Typen von Constraints

	anti-monoton?
$S \theta v, \theta \in \{=, \leq, \geq\}$	ja
$v \in S$	nein
$S \supseteq V$	nein
$S \subseteq V$	ja
$S = V$	teilweise
$min(S) \leq v$	nein
$min(S) \geq v$	ja
$min(S) = v$	teilweise
$max(S) \leq v$	ja
$max(S) \geq v$	nein
$max(S) = v$	teilweise
$count(S) \leq v$	ja
$count(S) \geq v$	nein
$count(S) = v$	teilweise
$sum(S) \leq v$	ja
$sum(S) \geq v$	nein
$sum(S) = v$	teilweise
$avg(S) \theta v, \theta \in \{=, \leq, \geq\}$	nein
(frequent constraint)	(ja)

4.3 Hierarchische Assoziationsregeln

Motivation

- in vielen Anwendungen: Item-Taxonomien (*is-a* Hierarchien)



- suche auch Assoziationsregeln zwischen abstrakteren Items

  z.B. zwischen Warengruppen

  ⇒ wesentlich höherer Support

4.3 Hierarchische Assoziationsregeln

Motivation

Beispiel

Ski-Hosen ⇒ Bergstiefel
Jacken ⇒ Bergstiefel
} Support < minsup

Oberkleidung ⇒ Bergstiefel Support > minsup

Eigenschaften

- Support von „Oberkleidung ⇒ Bergstiefel“ nicht unbedingt gleich

  Support von „Jacken ⇒ Bergstiefel“ + Support von „Ski-Hosen ⇒ Bergstiefel“
- wenn „Oberkleidung ⇒ Bergstiefel“ minimalen Support besitzt,

  dann auch „Kleidung ⇒ Bergstiefel“

4.3 Hierarchische Assoziationsregeln

Grundbegriffe [Srikant & Agrawal 1995]

- $I = \{i_1, ..., i_m\}$ eine Menge von Literalen, genannt „*Items*“
- *H* ein gerichteter azyklischer Graph über der Menge von Literalen *I*
- Kante in *H* von *i* nach *j* :

  i ist eine Verallgemeinerung von *j*,

  i heißt *Vater* oder *direkter Vorgänger* von *j*,

  j heißt *Sohn* oder *direkter Nachfolger* von *i*.
- $\bar{x}$ heißt *Vorfahre* von *x* (*x Nachfahre* von $\bar{x}$) bezüglich *H*:

  es gibt einen Pfad von $\bar{x}$ nach *x* in *H*
- Menge von Items $\bar{Z}$ heißt *Vorfahre* einer Menge von Items *Z*:

  mindestens ein Item in $\bar{Z}$ Vorfahre eines Items in *Z*

4.3 Hierarchische Assoziationsregeln

Grundbegriffe

- *D* eine Menge von Transaktionen *T*, wobei $T \subseteq I$
- typischerweise:

  die Transaktionen *T* enthalten nur Items aus den Blättern des Graphen *H*
- Transaktion *T unterstützt ein Item* $i \in I$:

  i in *T* enthalten ist oder *i* ein Vorfahre eines Items, das in *T* enthalten ist
- *T unterstützt eine Menge* $X \subseteq I$ von Items:

  T unterstützt jedes Item in *X*
- *Support einer Menge* $X \subseteq I$ von Items *in D* :

  Prozentsatz der Transaktionen in *D*, die *X* unterstützen

4.3 Hierarchische Assoziationsregeln

Grundbegriffe

- *hierarchische Assoziationsregel*:

  $X \Rightarrow Y$ mit $X \subseteq I$, $Y \subseteq I$, $X \cap Y = \emptyset$

  kein Item in Y ist Vorfahre eines Items in X bezüglich H

- *Support s einer hierarchischen Assoziationsregel $X \Rightarrow Y$ in D* :

  Support der Menge $X \cup Y$ in D

- *Konfidenz c einer hierarchischen Assoziationsregel $X \Rightarrow Y$ in D*:

  Prozentsatz der Transaktionen, die auch die Menge Y unterstützen in der Teilmenge aller Transaktionen, welche die Menge X unterstützen

4.3 Hierarchische Assoziationsregeln

Beispiel

TransaktionsID	Items
1	Hemd
2	Jacke, Bergstiefel
3	Ski-Hose, Bergstiefel
4	Straßenschuhe
5	Straßenschuhe
6	Jacke

Support von {Kleidung}: 4 von 6 = 67%

Support von {Kleidung, Bergstiefel}: 2 von 6 = 33%

„Schuhe ⇒ Kleidung“: Support 33%, Konfidenz 50%

„Bergstiefel ⇒ Kleidung“: Support 33%, Konfidenz 100%

4.3 Bestimmung der häufig auftretenden Itemsets

Grundidee

- Erweiterung der Transaktionen der Datenbank um alle Vorfahren von enthaltenen Items
- Methode
  - jedes Item in einer Transaktion *T* wird zusammen mit all seinen Vorfahren bezüglich *H* in eine neue Transaktion *T'* eingefügt
  - es werden keine Duplikate eingefügt
- Bleibt zu tun:

  Finden von Frequent Itemsets für einfache Assoziationsregeln (Apriori-Algorithmus)

⇒ *Basisalgorithmus* für hierarchische Assoziationsregeln

4.3 Bestimmung der häufig auftretenden Itemsets

Optimierungen des Basisalgorithmus

Vorberechnung von Vorfahren

- zusätzliche Datenstruktur H

  Item $\rightarrow$ Liste aller seiner Vorfahren

- effizienterer Zugriff auf alle Vorfahren eines Items

Filtern der hinzuzufügenden Vorfahren

- nur diejenigen Vorfahren zu einer Transaktion hinzufügen, die in einem Element der Kandidatenmenge C_k des aktuellen Durchlaufs auftreten
- Beispiel: C_k = {{Kleidung, Schuhe}}

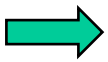
  „JackeXY“ durch „Kleidung“ ersetzen

4.3 Bestimmung der häufig auftretenden Itemsets

Optimierungen des Basisalgorithmus

Ausschließen redundanter Itemsets

- Sei X ein k-Itemset, i ein Item und $\bar{i}$ ein Vorfahre von i.
- $X = \{i, \bar{i}, \ldots\}$
- Support von $X - \{\bar{i}\}$ = Support von X
- X kann bei der Kandidatengenerierung ausgeschlossen werden.
- Man braucht kein k-Itemset zu zählen, das sowohl ein Item i als auch einen Vorfahren $\bar{i}$ von i enthält.



Algorithmus *Cumulate*

4.3 Bestimmung der häufig auftretenden Itemsets

Stratifikation

- Alternative zum Basis-Algorithmus (Apriori-Algorithmus)
- Stratifikation = Schichtenbildung der Mengen von Itemsets
- Grundlage

Itemset $\overline{X}$ hat keinen minimalen Support und $\overline{X}$ ist Vorfahre von X:

X hat keinen minimalen Support.

- Methode
  - nicht mehr alle Itemsets einer bestimmten Länge k auf einmal zählen
  - sondern erst die allgemeineren Itemsets zählen

    und die spezielleren Itemsets nur zählen, wenn nötig

4.3 Bestimmung der häufig auftretenden Itemsets

Stratifikation

Beispiel

C_k = {{Kleidung Schuhe}, {Oberkleidung Schuhe}, {Jacken Schuhe} }

zuerst den Support für {Kleidung Schuhe} bestimmen

nur dann den Support für {Oberkleidung Schuhe} bestimmen,
wenn {Kleidung Schuhe} minimalen Support hat

Begriffe

- *Tiefe* eines Itemsets:

  Für Itemsets X aus einer Kandidatenmenge C_k ohne direkten Vorfahren in C_k:
  Tiefe(X) = 0.
  Für alle anderen Itemsets X in C_k:
  Tiefe(X) = max{*Tiefe*($\overline{X}$) | $\overline{X} \in C_k$ ist direkter Vorfahre von X} + 1.

- (C_k^n): Menge der Itemsets der Tiefe n aus C_k, $0 \leq n \leq$ maximale Tiefe t

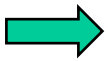
4.3 Bestimmung der häufig auftretenden Itemsets

Stratifikation

Algorithmus *Stratify*

- Zählen der Itemsets aus C_k^0
- Löschung aller Nachfahren von Elementen aus (C_k^0), die keinen minimalen Support haben
- Zählen der übriggebliebenen Elemente aus (C_k^1)
- und so weiter . . .

Tradeoff zwischen Anzahl der Itemsets, für die Support auf einmal gezählt wird und der Anzahl von Durchläufen durch die Datenbank



$|C_k^n|$ klein, dann Kandidaten der Tiefen (n, n+1, ..., t) auf einmal zählen

4.3 Bestimmung der häufig auftretenden Itemsets

Stratifikation

Problem von *Stratify*

falls sehr viele Itemsets mit kleiner Tiefe den minimalen Support haben:

Ausschluß nur weniger Itemsets größerer Tiefe

Verbesserungen von *Stratify*

- Schätzen des Supports aller Itemsets in C_k mit einer Stichprobe
- C_k': alle Itemsets, von denen man aufgrund der Stichprobe erwartet, daß sie oder zumindest alle ihre Vorfahren in C_k minimalen Support haben
- Bestimmung des tatsächlichen Supports der Itemsets in C_k' in einem Datenbankdurchlauf
- Entfernen aller Nachfahren von Elementen in C_k', die keinen minimalen Support haben, aus der Menge C_k'', $C_k'' = C_k - C_k'$
- Bestimmen des Supports der übriggebliebenen Itemsets in C_k'' in einem zweiten Datenbankdurchlauf

4.3 Bestimmung der häufig auftretenden Itemsets

Experimentelle Untersuchung

Testdaten

- Supermarktdaten

  548000 Items, Item-Hierarchie mit 4 Ebenen, 1,5 Mio. Transaktionen

- Kaufhausdaten

  228000 Items, Item-Hierarchie mit 7 Ebenen, 570000 Transaktionen

Ergebnisse

- Optimierungen von *Cumulate* und Stratifikation können kombiniert werden
- die Optimierungen von *Cumulate* bringen eine starke Effizienzverbesserung
- die Stratifikation bringt nur noch einen kleinen zusätzlichen Vorteil

4.3 Interessantheit hierarchischer Assoziationsregeln

Grundbegriffe

- $\overline{X} \Rightarrow \overline{Y}$ ist *Vorfahre* von $X \Rightarrow Y$:

  das Itemset $\overline{X}$ ist Vorfahre des Itemsets X ist und/oder das Itemset $\overline{Y}$ ist ein Vorfahre der Menge Y

- $\overline{X} \Rightarrow \overline{Y}$ *direkter Vorfahre* von $X \Rightarrow Y$ in einer Menge von Regeln:

  $\overline{X} \Rightarrow \overline{Y}$ ist Vorfahre von $X \Rightarrow Y$, und es existiert keine Regel $X' \Rightarrow Y'$, so daß $X' \Rightarrow Y'$ Vorfahre von $X \Rightarrow Y$ und $\overline{X} \Rightarrow \overline{Y}$ ein Vorfahre von $X' \Rightarrow Y'$ ist

- hierarchische Assoziationsregel $X \Rightarrow Y$ heißt *R-interessant*:

  hat keine direkten Vorfahren oder

  tatsächlicher Support > dem *R*-fachen des erwarteten Supports

  tatsächliche Konfidenz > dem *R*-fachen der erwarteten Konfidenz

4.3 Interessantheit hierarchischer Assoziationsregeln

Beispiel

Item	Support
Kleidung	20
Oberkleidung	10
Jacken	4

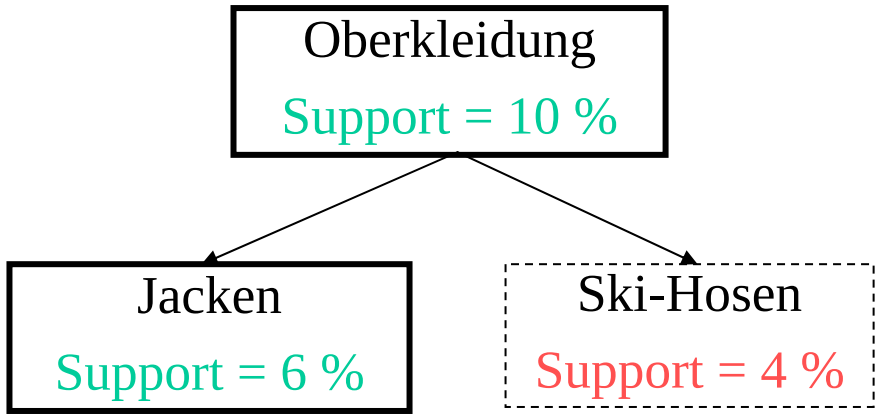
$R = 2$

Regel-Nr	Regel	Support	R-interessant?
1	Kleidung ⇒ Schuhe	10	ja, kein Vorfahre
2	Oberkleidung ⇒ Schuhe	9	ja, Support ≈ *R* * erwarteter Support (in Bezug auf Regel 1)
3	Jacken ⇒ Schuhe	4	nein, Support < *R* * erwarteter Support (in Bezug auf Regel 2)

4.3 Hierarchische Assoziationsregeln

Bestimmung von minsup

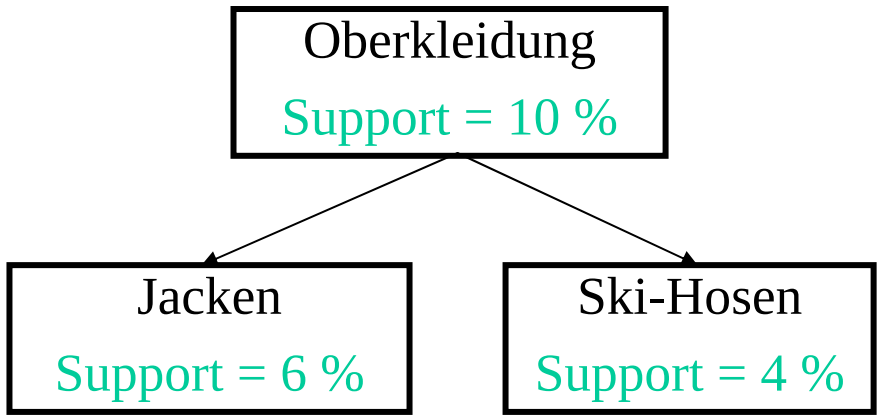
Fester Support



minsup = 5 %

minsup = 5 %

Variabler Support



minsup = 5 %

minsup = 3 %

4.3 Hierarchische Assoziationsregeln

Diskussion

Fester Support

• gleicher Wert für *minsup* auf allen Ebenen der Item-Taxonomie

\+ Effizienz: Ausschluß von Nachfahren nicht-häufiger Itemsets

\- beschränkte Effektivität

minsup zu hoch $\Rightarrow$ keine Low-Level-Assoziationen

minsup zu niedrig $\Rightarrow$ zu viele High-Level-Assoziationen

Variabler Support

• unterschiedlicher Wert für *minsup* je nach Ebene der Item-Taxonomie

\+ gute Effektivität

Finden von Assoziationsregeln mit der Ebene angepaßtem Support

\- Ineffizienz: kein Ausschluß von Nachfahren nicht-häufiger Itemsets

4.4 Quantitative Assoziationsregeln

Motivation

- Bisher: nur Assoziationsregeln für *boolesche* Attribute
- Jetzt: auch *numerische* Attribute

ID	Alter	Fam.stand	# Autos
1	23	ledig	0
2	38	verheiratet	2

ursprüngliche Datenbank

ID	Alter:20..29	Alter:30..39	Fam.stand:ledig	Fam.stand:verheiratet	. . .
1	1	0	1	0	. . .
2	0	1	0	1	. . .

boolesche Datenbank

4.4 Quantitative Assoziationsregeln

Lösungsansätze

Statische Diskretisierung

- Diskretisierung aller Attribute *vor* dem Bestimmen von Assoziationsregeln

  z.B. mit Hilfe einer Konzepthierarchie pro Attribut
- Ersetzung numerischer Attributwerte durch Bereiche / Intervalle

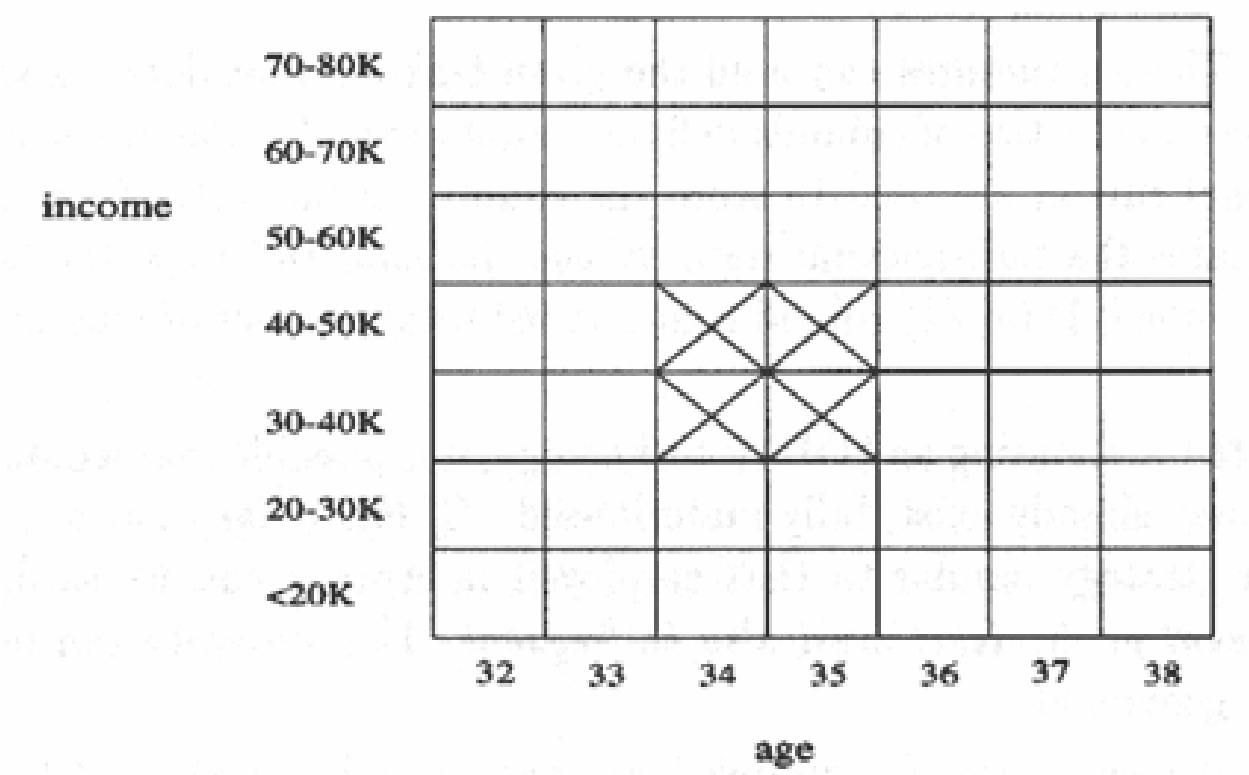
Dynamische Diskretisierung

- Diskretisierung der Attribute *beim* Bestimmen von Assoziationsregeln

  Ziel z.B. Maximierung der Konfidenz
- Zusammenfassen “benachbarter” Assoziationsregeln zu einer verallgemeinerten Regel

4.4 Quantitative Assoziationsregeln

Beispiel



$$age(X,\text{"34-35"}) \wedge income(X,\text{"30K - 50K"}) \Rightarrow buys(X,\text{"high resolution TV"})$$

4.4 Quantitative Assoziationsregeln

Grundbegriffe [Srikant & Agrawal 1996a]

- $I = \{i_1, ..., i_m\}$ eine Menge von Literalen, genannt „*Attribute*“
- $I_V = I \times \mathrm{IN}^+$ eine Menge von *Attribut-Wert-Paaren*
- *D* eine *Menge von Datensätzen R*, $R \subseteq I_V$

  jedes Attribut darf höchstens einmal in einem Datensatz vorkommen
- $I_R = \{<x, u, o> \in I \times \mathrm{IN}^+ \times \mathrm{IN}^+ \mid u \leq o\}$

  $<x, u, o>$: ein Attribut *x* mit einem zugehörigen Intervall von Werten $[u..o]$
- *Attribute*(*X*) für $X \subseteq I_R$: die Menge $\{x \mid <x, u, o> \in I_R\}$

4.4 Quantitative Assoziationsregeln

Grundbegriffe

- *quantitative Items*: die Elemente aus I_R

  quantitatives Itemset: Menge $X \subseteq I_R$

- Datensatz *R unterstützt* eine Menge $X \subseteq I_R$:

  zu jedem $\langle x, u, o \rangle \in X$ gibt es ein Paar $\langle x, v \rangle \in R$ mit $u \leq v \leq o$

- *Support der Menge X in D* für ein quantitatives Itemset *X*:

  Prozentsatz der Datensätze in *D*, die *X* unterstützen

- *quantitative Assoziationsregel*:

  $X \Rightarrow Y$ mit $X \subseteq I_R$, $Y \subseteq I_R$ und Attribute(X) $\cap$ Attribute(Y) = $\emptyset$

4.4 Quantitative Assoziationsregeln

Grundbegriffe

- *Support s einer quantitativen Assoziationsregel* $X \Rightarrow Y$ *in* D:

  Support der Menge $X \cup Y$ in D

- *Konfidenz c einer quantitativen Assoziationsregel* $X \Rightarrow Y$ *in* D:

  Prozentsatz der Datensätze, die die Menge Y unterstützen in der Teilmenge aller Datensätze, welche auch die Menge X unterstützen

- Itemset $\overline{X}$ heißt *Verallgemeinerung* eines Itemsets X (X *Spezialisierung* von $\overline{X}$):

  1. X und $\overline{X}$ enthalten die gleichen Attribute
  2. die Intervalle in den Elementen von X sind vollständig in den entsprechenden Intervallen von $\overline{X}$ enthalten

⇨ Entsprechung zu „Vorfahre“ und „Nachfahre“ im Fall von Itemtaxonomien

4.4 Quantitative Assoziationsregeln

Methode

- Diskretisierung numerischer Attribute

  Wahl geeigneter Intervalle

  Erhaltung der ursprünglichen Ordnung der Intervalle

- Transformation kategorischer Attribute auf aufeinanderfolgende ganze Zahlen
- Transformation der Datensätze in *D*

  gemäß der Transformation der Attribute
- Bestimmung des Supports für jedes einzelne Attribut-Wert-Paar in *D*

4.4 Quantitative Assoziationsregeln

Methode

- Zusammenfassung „benachbarter Attributwerte“ zu Intervallen

  solange der Support der entstehenden Intervalle kleiner ist als *maxsup*

  ⇒ häufig vorkommende 1-Itemsets

- Finden aller häufig auftretenden quantitativen Itemsets

  Variante des Apriori-Algorithmus

- Bestimmen quantitativer Assoziationsregeln

  aus häufig auftretenden Itemsets

- Entfernen aller uninteressanten Regeln

  Entfernen aller Regeln, deren Interessantheit kleiner ist als *min-interst*

  ähnliches Interessantheitsmaß wie bei hierarchischen Assoziationsregeln

4.4 Quantitative Assoziationsregeln

Beispiel

ID	Alter	Fam.stand	# Autos
100	23	ledig	1
200	25	verheiratet	1
300	29	ledig	0
400	34	verheiratet	2
500	38	verheiratet	2

Intervall	Integer
20..24	1
25..29	2
30..34	3
35..39	4

Wert	Integer
verheiratet	1
ledig	2

Vorverarbeitung

ID	Alter	Fam.stand	# Autos
100	1	2	1
200	2	1	1
300	2	2	0
400	3	1	2
500	4	1	2

4.4 Quantitative Assoziationsregeln

Beispiel

ID	Alter	Fam.stand	# Autos
100	23	ledig	1
200	25	verheiratet	1
300	29	ledig	0
400	34	verheiratet	2
500	38	verheiratet	2

Itemset	Support
(<Alter: 20..29>)	3
(<Alter: 30..39>)	2
(<Fam.stand: verheiratet>)	3
(<Fam.stand: ledig>)	2
(<#Autos: 0..1>)	3
(<Alter: 30..39> <Fam.stand: verheiratet>)	2
. . .	. . .

Data Mining

Itemset	Support	Konfidenz
<Alter: 30..39> und <Fam.stand: verheiratet> --> <#Autos: 2>	40%	100%
<Alter: 20..29> --> <#Autos: 0..1>	60%	67%

4.4 Quantitative Assoziationsregeln

Partitionierung numerischer Attribute

Probleme

- Minimaler Support

  zu viele Intervalle $\rightarrow$ zu kleiner Support für jedes einzelne Intervall

- Minimale Konfidenz

  zu wenig Intervalle $\rightarrow$ zu kleine Konfidenz der Regeln

Lösung

- Zerlegung des Wertebereichs in viele Intervalle
- Zusätzliche Berücksichtigung aller Bereiche, die durch Verschmelzen benachbarter Intervalle entstehen

durchschnittlich $O(n^2)$ viele Bereiche, die einen bestimmten Wert enthalten