

Skript zur Vorlesung
Knowledge Discovery in Databases
im Wintersemester 2010/2011

Kapitel 7: Assoziationsregeln

Vorlesung+Übungen:
PD Dr. Peer Kröger, Dr. Arthur Zimek

Skript © 2010 Johannes Aßfalg, Christian Böhm, Karsten Borgwardt, Martin Ester, Eshref Januzaj, Karin Kailing, Peer Kröger, Jörg Sander, Matthias Schubert, Arthur Zimek

[http://www.dbs.ifi.lmu.de/cms/Knowledge_Discovery_in_Databases_I_\(KDD_I\)](http://www.dbs.ifi.lmu.de/cms/Knowledge_Discovery_in_Databases_I_(KDD_I))

Inhalt dieses Kapitels

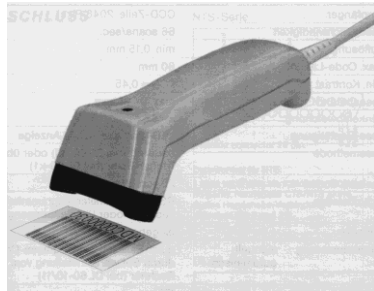
7.1 Einleitung

7.2 Grundlagen

7.3 Itemset Mining

7.4 Association Rule Mining

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

Assoziationsregeln

Regeln der Form

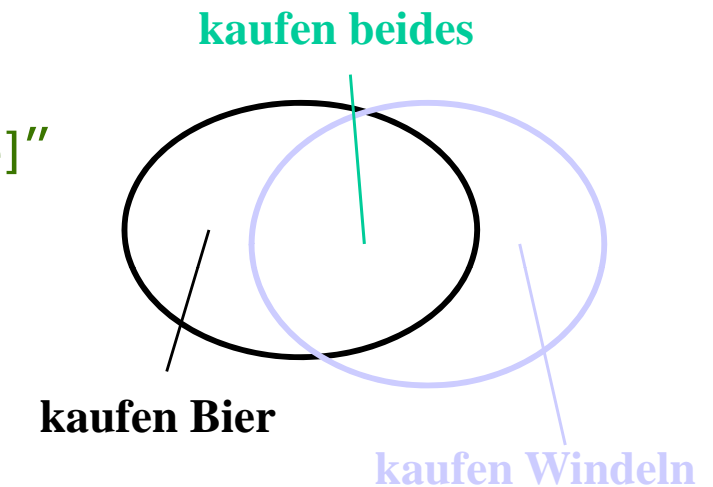
“Rumpf \rightarrow Kopf [support, confidence]”

Beispiele

$\text{kauft}(X, \text{“Windeln”}) \rightarrow \text{kauft}(X, \text{“Bier”})$ [0.5%, 60%]

$\text{hauptfach}(X, \text{“CS”}) \wedge \text{kurs}(X, \text{“DB”}) \rightarrow \text{abschluß}(X, \text{“A”})$ [1%, 75%]

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



- *Items* $I = \{i_1, \dots, i_m\}$ eine Menge von Literalen
z.B. Waren/Artikel bei einem Einkauf
- *Itemset* X : Menge von Items $X \subseteq I$
z.B. ein kompletter Einkauf
- *Datenbank DB*: Menge von *Transaktionen* T mit $T = (tid, X_T)$
z.B. Menge aller Einkäufe (=Transaktionen) in einem bestimmten Zeitraum
- Transaktion T enthält Itemset X : $X \subseteq T$
- Items in Transaktionen oder Itemsets sind lexikographisch sortiert:
Itemset $X = (x_1, x_2, \dots, x_k)$, wobei $x_1 \leq x_2 \leq \dots \leq x_k$
- *Länge des Itemsets*: Anzahl der Elemente in einem Itemset
- *k-Itemset*: ein Itemset der Länge k
{Butter, Brot, Milch, Zucker} ist ein 4-Itemset
{Mehl, Wurst} ist ein 2-Itemset

- *Cover* eines Itemset X : Menge der Transaktionen T , die X enthalten:

$$cover(X) = \{tid \mid (tid, X_T) \in DB, X \subseteq X_T\}$$

- *Support* des Itemset X in DB : Anteil der Transaktionen in DB , die X enthalten: $support(X) = |cover(X)|$

Bemerkung: $support(\emptyset) = |DB|$

- *Häufigkeit* eines Itemsets X in DB :

Wahrscheinlichkeit, daß X in einer Transaktion $T \in DB$ auftritt:

$$frequency(X) = P(X) = support(X) / |DB|$$

- *Häufig auftretendes (frequent) Itemset* X in DB :

$$support(X) \geq s \quad (0 \leq s \leq |DB|)$$

s ist ein absoluter support-Grenzwert

Alternativ: $frequency(X) \geq s_{rel}$ wobei $s = \lceil s_{rel} \cdot |DB| \rceil$

Problem 1 (Itemset Mining)

Gegeben:

- eine Menge von Items I
- eine Transaktionsdatenbank DB über I
- Ein absoluter support-Grenzwert s

Finde alle frequent Itemsets in DB , d.h. $\{X \subseteq I \mid \text{support}(X) \geq s\}$

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

Support der 1-Itemsets:

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

Support der 2-Itemsets:

(A, C): 50%,

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

- *Assoziationsregel*: Implikation der Form $X \Rightarrow Y$,
wobei gilt: $X \subseteq I$, $Y \subseteq I$ und $X \cap Y = \emptyset$,
 X heißt *Rumpf*
 Y heißt *Kopf*
- *Support einer Assoziationsregel* $A \equiv X \Rightarrow Y$ in DB : Support von $X \cup Y$ in DB
 $support(A) = support(X \cup Y)$
- *Häufigkeit einer Assoziationsregel* A in DB :
 $frequency(A) = support(A) / |DB|$
- *Konfidenz einer Assoziationsregel* $A \equiv X \Rightarrow Y$ in DB :
Anteil der Transaktionen, die die Menge Y enthalten, in der Teilmenge
aller Transaktionen aus DB , welche die Menge X enthalten

$$confidence(A) = \frac{support(X \cup Y)}{support(X)}$$

Problem 2 (Association Rule Mining)

Gegeben:

- eine Menge von Items I
- eine Transaktionsdatenbank DB über I
- Ein absoluter support-Grenzwert s und confidenz-Grenzwert c

Finde alle Assoziationsregeln $A \equiv X \Rightarrow Y$ in DB , die mind. einen Support von s und mind. eine Konfidenz von c haben, d.h.

$$\{A \equiv X \Rightarrow Y \mid \text{support}(A) \geq s, \text{confidence}(A) \geq c\}$$

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

Assoziationsregeln:

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

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

Problem 1 ist Teilproblem von Problem 2:

- Itemset X häufig bzgl. s
- Y Teilmenge von X
- $Y \Rightarrow (X - Y)$ hat minimalen Support bzgl. s

2-stufiges Verfahren um Assoziationsregeln zu bestimmen:

1. Bestimmung der frequent Itemsets:

„naiver“ Algorithmus: zähle die Häufigkeit aller k -elementigen Teilmengen von I ineffizient, da $\binom{|I|}{k}$ solcher Teilmengen

Gesamt-Kosten: $O(2^{|I|})$

=> Apriori-Algorithmus und Varianten, Tiefensuch-Algorithmen

2. Generierung der Assoziationsregeln mit minimaler Konfidenz bzgl. c :

generiere $Y \Rightarrow (X - Y)$ aus frequent Itemset X

Running Example

<i>tid</i>	X_T
1	{Bier, Chips, Wein}
2	{Bier, Chips}
3	{Pizza, Wein}
4	{Chips, Pizza}

Transaktionsdatenbank

$I = \{\text{Bier, Chips, Pizza, Wein}\}$

Itemset	Cover	Sup.	Freq.
{}	{1,2,3,4}	4	100 %
{Bier}	{1,2}	2	50 %
{Chips}	{1,2,4}	3	75 %
{Pizza}	{3,4}	2	50 %
{Wein}	{1,3}	2	50 %
{Bier, Chips}	{1,2}	2	50 %
{Bier, Wein}	{1}	1	25 %
{Chips, Pizza}	{4}	1	25 %
{Chips, Wein}	{1}	1	25 %
{Pizza, Wein}	{3}	1	25 %
{Bier, Chips, Wein}	{1}	1	25 %

Regel	Sup.	Freq.	Conf.
{Bier} \Rightarrow {Chips}	2	50 %	100 %
{Bier} \Rightarrow {Wein}	1	25 %	50 %
{Chips} \Rightarrow {Bier}	2	50 %	66 %
{Pizza} \Rightarrow {Chips}	1	25 %	50 %
{Pizza} \Rightarrow {Wein}	1	25 %	50 %
{Wein} \Rightarrow {Bier}	1	25 %	50 %
{Wein} \Rightarrow {Chips}	1	25 %	50 %
{Wein} \Rightarrow {Pizza}	1	25 %	50 %
{Bier, Chips} \Rightarrow {Wein}	1	25 %	50 %
{Bier, Wein} \Rightarrow {Chips}	1	25 %	100 %
{Chips, Wein} \Rightarrow {Bier}	1	25 %	100 %
{Bier} \Rightarrow {Chips, Wein}	1	25 %	50 %
{Wein} \Rightarrow {Bier, Chips}	1	25 %	50 %

- „naiver“ Algorithmus: zähle die Häufigkeit aller k -Itemsets von I

teste insgesamt $\sum_{k=1}^m \binom{m}{k} = 2^m - 1$ Itemsets, d.h. $O(2^m)$ mit $m = |I|$

- *Kandidaten Itemset X:*

Algorithmus evaluiert, ob X frequent ist
Kandidatenmenge sollte so klein wie möglich sein

- *Monotonie Eigenschaft* von frequent Itemsets

Wenn X frequent ist, sind alle Teilmengen $Y \subseteq X$ auch frequent.

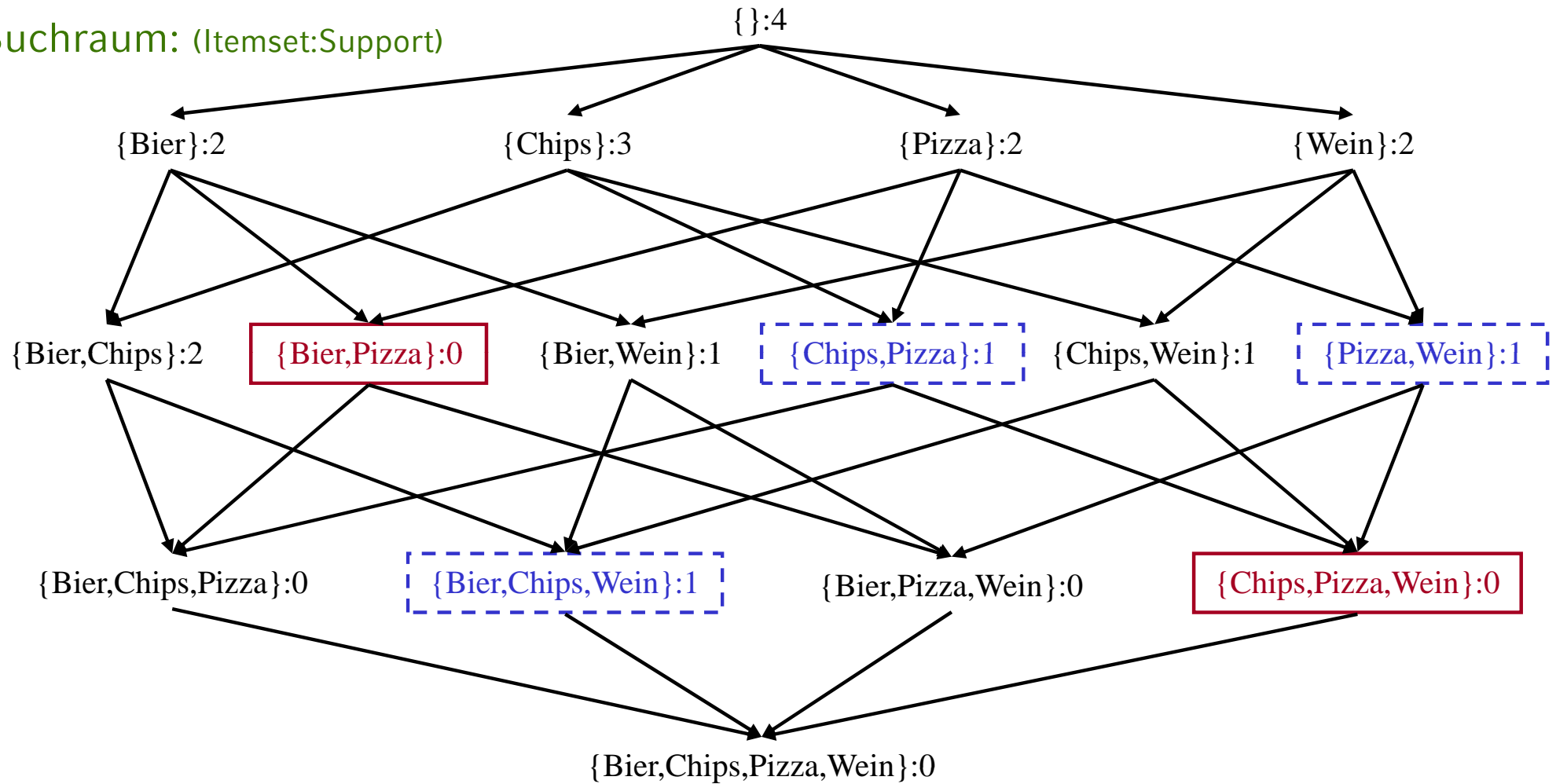
Umkehrung: Wenn X nicht frequent ist, können alle Itemsets, die X als Teilmenge enthalten, auch nicht mehr frequent sein!

- *Rand (Border) Itemset X:*

alle Teilmengen $Y \subset X$ sind frequent, alle Obermengen $Z \supset X$ sind nicht frequent

- *positiver Rand:* X ist selbst frequent
- *negativer Rand:* X ist selbst nicht frequent

Suchraum: (Itemset:Support)



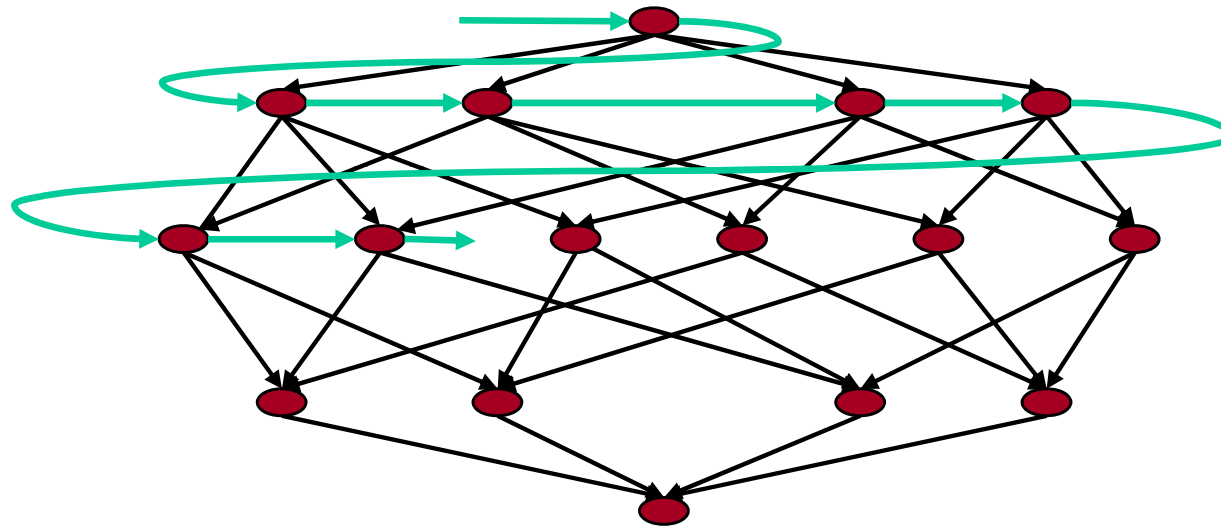
Positive Rand-Itemsets

Minimaler Support $s = 1$

Negative Rand-Itemsets

Apriori Algorithmus [Agrawal & Srikant 1994]

- zuerst die einelementigen Frequent Itemsets bestimmen, dann die zweielementigen und so weiter (Breitensuche)



- 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)

C_k : die zu zählenden Kandidaten-Itemsets der Länge k

L_k : Menge aller häufig vorkommenden Itemsets der Länge k

Apriori($I, DB, minsup$)

$L_1 := \{\text{frequent 1-Itemsets aus } I\};$

$k := 2;$

while $L_{k-1} \neq \emptyset$ **do**

$C_k := \text{AprioriKandidatenGenerierung}(L_{k-1});$

for each Transaktion $T \in DB$ **do**

$CT := \text{Subset}(C_k, T);$ // alle Kandidaten aus C_k , die
 // der Transaktion T enthalten sind;

for each Kandidat $c \in CT$ **do** $c.count++;$

$L_k := \{c \in C_k \mid c.count \geq minsup\};$

$k++;$

return $\bigcup_k L_k;$

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 $k-2$ Items übereinstimmen

$p \in L_{k-1}$

(Bier, Chips, Pizza)



(Bier, Chips, Pizza, Wein) $\in C_k$

$q \in L_{k-1}$

(Bier, Chips, Wein)



Kandidatengenerierung

Schritt 2: Pruning

entferne alle Kandidaten- k -Itemsets, die eine $k-1$ -elementige Teilmenge enthalten, die nicht zu L_{k-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)

 $C_4 = \{(1\ 2\ 3\ 4)\}$

Beispiel

minsup = 2

TID	Items
100	1 3 4
200	2 3 5
300	1 2 3 5
400	2 5

Scan D →

C_1

itemset	sup.
{1}	2
{2}	3
{3}	3
{4}	1
{5}	3

→

L_1

itemset	sup.
{1}	2
{2}	3
{3}	3
{5}	3

L_2

itemset	sup
{1 3}	2
{2 3}	2
{2 5}	3
{3 5}	2

←

C_2

itemset	sup
{1 2}	1
{1 3}	2
{1 5}	1
{2 3}	2
{2 5}	3
{3 5}	2

Scan D ←

C_2

itemset
{1 2}
{1 3}
{1 5}
{2 3}
{2 5}
{3 5}

↘

C_3

itemset
{2 3 5}

Scan D →

L_3

itemset	sup
{2 3 5}	2

↘

Eigenschaften:

- Benötigt für alle Itemsets der Länge k einen Datenbank-Scan
⇒ $O(l \cdot |D|)$
- Menge der generierten Kandidaten, die nicht frequent sind entspricht dem negativen Rand

$$\Rightarrow O\left(\binom{m}{\lfloor m/2 \rfloor}\right) \quad (\text{Sperners Theorem})$$

- Wenn nicht alle Kandidaten Itemsets in den Hauptspeicher passen, werden Kandidaten blockweise auf min. Support überprüft

Verbesserungen (u.a.)

- Hashbaum zur Unterstützung der Subset-Funktion:
 - Subset-Funktion muss für jede Transaktion in DB und jede Kandidatenmenge C_k alle Kandidaten in C_k finden, die T enthalten
 - Organisiere Kandidaten aus C_k in einem Hash-Baum
- Tiefensuche statt Breitensuche:
 - Die Cover der Kandidaten-Itemsets in einer Iteration kann sehr groß sein
⇒ beim Zählen des supports reicht evtl.
der Hauptspeicher nicht mehr aus
 - Die Größe der Cover kann evtl. deutlich reduziert werden, indem die Kandidaten-Itemsets in einer Tiefensuch-Strategie erzeugt werden

Methode

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

$$\text{confidence}(Y \Rightarrow (X - Y)) = \frac{\text{support}(X)}{\text{support}(Y)}$$


- Speicherung der Frequent Itemsets mit ihrem Support in einer Hashtabelle

➔ keine Datenbankzugriffe

- *Monotonie der Konfidenz* bei Assoziationsregeln:
seien $X, Y, Z \subseteq I$ Itemsets mit $X \cap Y = \emptyset$
Es gilt: $confidence(X \setminus Z \Rightarrow Y \cup Z) \leq confidence(X \Rightarrow Y)$
- Bottom-up Bestimmung der Assoziationsregeln
ähnlich Apriori-Algorithmus möglich
- Beachte: für jedes Itemset X mit $support(X) > 0$ gilt
 - $confidence(X \Rightarrow \emptyset) = 100\%$
 - $confidence(\emptyset \Rightarrow X) = frequency(X)$d.h. wenn $frequency(X) \geq c$
dann haben alle Regeln $Y \Rightarrow (X - Y)$ minimale Konfidenz
d.h. $confidence(Y \Rightarrow (X - Y)) \geq c$

Interessantheit von Assoziationsregeln

Beispiel

- Daten über das Verhalten von Schülern in einer Schule mit 5000 Schülern
 - Itemsets mit Support:
 - 60% der Schüler spielen Fußball, 75% der Schüler essen Schokoriegel
 - 40% der Schüler spielen Fußball *und* essen Schokoriegel
 - Assoziationsregeln:
 - „Spielt Fußball“ \Rightarrow „Isst Schokoriegel“, Konfidenz = 67%
 - TRUE \Rightarrow „Isst Schokoriegel“, Konfidenz = 75%
-  Fußball spielen und Schokoriegel essen sind *negativ korreliert*

Aufgabenstellung

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

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

für eine geeignete Konstante $d > 0$

- Maß für die „Interessantheit“ einer Regel

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

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

- Frequent Itemset Mining findet häufig auftretende Teilmengen in Transaktionsdatenbanken
- Assoziationsregeln unterteilen diese Teilmengen in Regeln (Kopf und Rumpf)
- Hauptaufwand entsteht beim Finden der frequent Itemsets
- Itemset Mining ist der bekannteste Vertreter des allgemeineren Data Mining Tasks, Frequent Pattern Mining
- Es existieren noch weitere Vertreter für kompliziertere Objektdarstellungen: frequent Substrings, frequent Subgraph...