VII.3

Assoziationsregeln

VII.3 Assoziationsregelverfahren

VII.3.1 Einführung [Bollinger 96]

- Algorithmen zum Entdecken von Assoziationsregeln sind typische Vertreter von Data Mining Verfahren.
- Assoziationsregeln beschreiben Korrelationen zwischen gemeinsam auftretenden Dingen.
- **Beispiel: (Warenkorb-Analyse)**
  Artikel, die Kunden eines Supermarktes zusammen einkaufen
  „In 45% der Fälle, in denen Lachs gekauft wird, wird auch Weißwein gekauft. Diese beiden Produkte kommen in 2% aller Transaktionen vor.“
- typische Anwendungsbereiche:
  - Einzel- und Versandhandel
  - Tourismus

- Für Assoziationsregeln sind folgende Parameter relevant:
  - **Konfidenz** der Regel,
    d.h. Stärke der Korrelation („*in 45% der Fälle*“)
  - **Support** der Regel,
    d.h. Häufigkeit des gemeinsamen Auftretens („*in 2% aller Transaktionen*“)
- Algorithmen sind so konstruiert, daß sie **alle** Assoziationsregeln mit vorgegebener Mindestkonfidenz und Mindestsupport entdecken.
- Benutzer muß **keine** Annahmen darüber machen, welche Dinge korrelieren könnten. (Dies ist bei Tausenden von Artikeln auch nicht möglich.)
- Datenbestände sind typischerweise sehr groß

  => Algorithmen müssen **effizient** sein.

VII.3.2 Das Assoziationsproblem

- gegeben:
  - Menge J von **Items** (uninterpretierte, diskrete Entities)
  - Liste D von Transaktionen, wobei eine **Transaktion** eine Menge $t \subseteq J$ von Items ist.
- **Assoziationsregel** $X \rightarrow Y$ besteht aus:
  - **Regelrumpf** X (Menge von Items)
  - **Regelkopf** Y (Menge von Items)
- Transaktion t **erfüllt** Assoziationsregel $X \rightarrow Y$ falls $(X \cup Y) \subseteq t$

- **Support** einer Assoziationsregel $X \rightarrow Y$ ist der Anteil der Transaktionen aus D, die Regel erfüllen:

$$\text{support}(X \rightarrow Y) = \frac{|\{t \in D \mid (X \cup Y) \subseteq t\}|}{|D|}$$

- **Konfidenz** einer Assoziationsregel $X \rightarrow Y$ ist das Verhältnis der Transaktionen aus D, die Regel erfüllen, zur Gesamtheit aller Regeln, die Regelrumpf erfüllen:

$$\text{confidence}(X \rightarrow Y) = \frac{|\{t \in D \mid (X \cup Y) \subseteq t\}|}{|\{t \in D \mid X \subseteq t\}|} = \frac{\text{support}(X \rightarrow Y)}{\text{support}(X)}$$

- Definition (**Assoziationsproblem**):
  - gegeben: Menge D von Transaktionen
    Wert für minimalen Support s_{min}
    Wert für minimale Konfidenz c_{min}
  - gesucht: Finde alle Assoziationsregeln $X \rightarrow Y$, so daß
    $\text{support}(X \rightarrow Y) \geq s_{min}$
    $\text{confidence}(X \rightarrow Y) \geq c_{min}$

Beispiel: Transaktionen und Assoziationsregeln in einer Supermarkt-Anwendung

Einkaufs-Transaktionen	gekaufte Artikel (Items)
t_1	Saft, Cola, Bier
t_2	Saft, Cola, Wein
t_3	Saft, Wasser
t_4	Cola, Bier, Saft
t_5	Saft, Cola, Bier, Wein
t_6	Wasser

⇓

Artikel (Item)	Transaktionen, in denen der Artikel vorkommt	Support des Items
Saft	t_1, t_2, t_3, t_4, t_5	5/6 = 83.3%
Cola	t_1, t_2, t_4, t_5	4/6 = 66.6%
Bier	t_1, t_4, t_5	3/6 = 50 %
Wein	t_2, t_5	2/6 = 33.3%
Wasser	t_3, t_6	2/6 = 33.3%

Die Regel Saft → Cola gilt z.B. mit einem Support von 66.6% und einer Konfidenz von 80%

VII.3.3 Apriori-Algorithmus zum Finden von Assoziationsregeln

[Agrawal et al. 94]

Idee:

Schritt 1: Bestimme alle Itemmengen mit vorgegebenem Mindestsupport, die sog. **häufigen Itemmengen**

Schritt 2: Bilde alle **Assoziationsregeln** mit vorgegebener Mindestkonfidenz aus den häufigen Itemmengen

- zu Schritt 1:
  - berechne sukzessive die häufigen Itemmengen mit i=1,2,3,... Elementen.
  - fasse die häufigen Itemmengen mit i Elementen zur Menge I_i zusammen: $I_i := \{X \mid X \text{ ist häufige Itemmenge}, |X| = i\}$

- **Lemma:** (i) Für $X' \subseteq X$ gilt: $support(X') \geq support(X)$

  (ii) Wenn X häufige Itemmenge ist, dann auch jedes $X' \subseteq X$

  (iii) Für $X \in I_{n+1}$ gilt: alle n-elementigen Teilmengen von X sind häufig.

  (iv) Sei X eine n+1-elementige Menge von Items. Wenn es eine n-elementige Teilmenge X' von X gibt mit $X' \notin I_n$, dann gilt auch $X \notin I_{n+1}$.

- **Lemma (Wdh.):**

  (iv) Sei X eine n+1-elementige Menge von Items. Wenn es eine n-elementige Teilmenge X' von X gibt mit $X' \notin I_n$, dann gilt auch $X \notin I_{n+1}$.

- Berechnung von I_{n+1} aus I_n:
  - n-elementige **häufige** Mengen werden um jeweils ein Element erweitert
  - prüfe, ob (n+1)-elementige Menge häufig ist:

    wenn ja, nehme (n+1)-elementige Menge in I_{n+1} auf.

Apriori-Algorithmus zur Berechnung häufiger Itemmengen:

1. Initialisierung:
   s_{min} = Wert für minimalen Support;
   n := 1;
   I := ∅;
   $H_n := \{\{i\} \mid i \text{ ist ein Item}\}$;
2. gehe über die Datenbasis D und bestimme für alle $H \in H_n$ den Support;
3. $I_n := \{H \in H_n \mid support(H) \geq s_{min}\}$;
   $I := I \cup I_n$;
4. Falls $I_n = \emptyset$, gebe I als Ergebnis aus; Ende.
5. $H_{n+1} := \{\{i_1,i_2,...,i_{n+1}\} \mid \forall j: 1 \leq j \leq n+1:(\{i_1,i_2,...,i_{n+1}\} - \{i_j\}) \in I_n\}$;
   n := n+1; /* Details s. Tafel */
6. gehe nach 2.

Fortführung des Beispiels (mit s_{min}=50%)

Einkaufs-Transaktionen	gekaufte Artikel (Items)
t_1	Saft, Cola, Bier
t_2	Saft, Cola, Wein
t_3	Saft, Wasser
t_4	Cola, Bier, Saft
t_5	Saft, Cola, Bier, Wein
t_6	Wasser

Apriori, Schritt 1

n	H_n	I_n
1	{{Saft},{Cola},{Bier},{Wein},{Wasser}}	{{Saft},{Cola},{Bier}}
2	{{Saft, Cola},{Saft, Bier},{Cola, Bier}}	{{Saft, Cola},{Cola, Bier}, {Saft, Bier}}
3	{{Saft, Cola, Bier}}	{{Saft, Cola, Bier}}
4	{}	{}

Fortführung des Beispiels (mit s_{min}=50%)

n	H_n	I_n
1	{{Saft},{Cola},{Bier},{Wein},{Wasser}}	{{Saft},{Cola},{Bier}}
2	{{Saft, Cola},{Saft, Bier},{Cola, Bier}}	{{Saft, Cola},{Cola, Bier}, {Saft, Bier}}
3	{{Saft, Cola, Bier}}	{{Saft, Cola, Bier}}
4	{}	{}

Zu Schritt 2 (Erzeugung der Regeln):

- gegeben sei c_{min} = 75%
- Aus {Saft, Cola} $\in I_2$ erhalten wir z.B. die Regeln

$$\text{confidence}(\text{Saft} \rightarrow \text{Cola}) = \frac{\text{support}(\{\text{Saft},\text{Cola}\})}{\text{support}(\{\text{Saft}\})} = \frac{2/3}{5/6} = 80\%$$

$$\text{confidence}(\text{Cola} \rightarrow \text{Saft}) = \frac{\text{support}(\{\text{Saft},\text{Cola}\})}{\text{support}(\{\text{Cola}\})} = \frac{2/3}{2/3} = 100\%$$

Apriori-Algorithmus (Schritt 2)

Schritt 1: Bestimme alle Itemmengen mit vorgegebenem Mindestsupport, die sog. **häufigen Itemmengen**

Schritt 2: Bilde alle **Assoziationsregeln** mit vorgegebener Mindestkonfidenz aus den häufigen Itemmengen unter Verwendung des Satzes:

Satz: Sei X häufige Itemmenge, dann gilt für X‘ $\subseteq$ X:

$$\text{confidence}((X - X') \rightarrow X') = \frac{\text{support}(X)}{\text{support}(X - X')}$$

D.h. die Konfidenz läßt sich aus dem Support der häufigen Itemmengen berechnen.

Fortführung des Beispiels: abgeleitete Assoziationsregeln zu den häufigen Itemmengen (mit minsupp=50%)

Regeln mit Support ≥ 50%	erfüllende Transaktionen	Support	Konfidenz
Saft → Cola	t_1, t_2, t_4, t_5	66%	80%
Cola → Saft	t_1, t_2, t_4, t_5	66%	100%
Cola → Bier	t_1, t_4, t_5	50%	75%
Bier → Cola	t_1, t_4, t_5	50%	100%
Saft → Bier	t_1, t_4, t_5	50%	60%
Bier → Saft	t_1, t_4, t_5	50%	100%
Saft, Bier → Cola	t_1, t_4, t_5	50%	100%
Cola, Saft → Bier	t_1, t_4, t_5	50%	75%
Bier, Cola → Saft	t_1, t_4, t_5	50%	100%
Cola → Saft, Bier	t_1, t_4, t_5	50%	75%
Bier → Cola, Saft	t_1, t_4, t_5	50%	100%
Saft → Cola, Bier	t_1, t_4, t_5	50%	60%

Bemerkungen I/II:

- Wenn die größte häufige Itemmenge n Elemente enthält, benötigt der Algorithmus n Durchläufe
- 1 Megabyte Warenkorb-Daten wird im Sekundenbereich analysiert
- Algorithmus entdeckt Tausende von Regeln:
  - **Visualisierung, Browsing** notwendig
  - **Formale Begriffsanalyse** kann zur Reduktion verwendet werden
  - Regeln mit sehr hohen Support-/Konfidenzwerten schon bekannt
  - Regeln im ‘‘**Mittelfeld**‘‘ interessant
  - verwende zusätzlich statistische **Maße** für Bewertung von Regeln

Bemerkungen II/II:

- Regeln berücksichtigen viele **externe Einflußfaktoren** nicht, da diese in den Transaktionen nicht repräsentiert sind:
  - Käufergruppen (Alter, sozialer Status, ...)
  - Tageszeit, Wochentag
  - Werbekampagnen
- Das Verfahren kann auf interessante Teilmengen der Transaktionenmenge D eingeschränkt werden:
  - benutzerdefiniert (z.B. mit SQL-Statements oder OLAP)
  - oder durch Vorschaltung eines Clusterverfahrens

Weitere Auswertung:

Wenn die Regel $X \rightarrow Y$ vom Benutzer als interessant befunden ist, kann sie weiter ausgewertet werden:

- Für Regel $X \rightarrow Y$ sei D_x Menge der Transaktionen die X enthalten.
- Sei D_X^Y die Menge der Transaktionen, die auch Y enthalten
- Sei $D_X^{\neg Y}$ die Menge der Transaktionen, die Y nicht enthalten
- Was sind diskriminierende Faktoren?
- Bestimmung mit überwachten Lernverfahren (z.B. Entscheidungsbäume, ILP)!

VII.3.4 Mining Generalized Association Rules (Srikant/Agrawal 1995)

- Rules can be generalized with background knowledge, by using **taxonomies** on the items, e.g.

  Bier *is-a* alkoholisches Getränk

  Chips *is-a* Salzgebäck

  leading to rules like

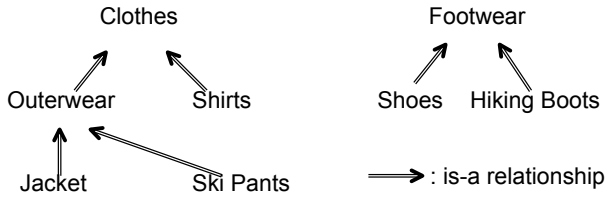
  Salzgebäck $\rightarrow$ Bier

  Chips $\rightarrow$ alhoholisches Getränk
- The knowledge can be derived from **ontologies** and **domain models.**
- The basic algorithm does not consider taxonomies: only items are considered which are the **leaves** of such a taxonomy, i.e. items are real products being purchased.

VII.3.4.1 Problem Statement

- **Def.:** Let is-a be a hierarchy on the set J* of generalized items, s.t. $J \subseteq J^*$ contains all leaves. A *generalized association rule* is an association rule X=>Y s.t. there are no $x \in X$ and $y \in Y$ with x is-a y.
- **Problem:** Determine, for given minsupp and minconf, all generalized association rules having at least minimum support minsupp and minimum confidence minconf.

Example: Taxonomy for Clothes and Footwear (Srikant/Agrawal 1995)



Idea: - derive rule 'Outerwear => Hiking Boots'
from 'Jacket => Hiking Boots' and
'Ski Pants => Hiking Boots'

Example: (Srikant/Agrawal 1995)

- given database D:

Database $\mathcal{D}$

Transaction	Items Bought
100	Shirt
200	Jacket, Hiking Boots
300	Ski Pants, Hiking Boots
400	Shoes
500	Shoes
600	Jacket

- assume: minimum support = 30%, minimum confidence = 60%

Frequent Itemsets

Itemset	Support
{ Jacket }	2
{ Outerwear }	3
{ Clothes }	4
{ Shoes }	2
{ Hiking Boots }	2
{ Footwear }	4
{ Outerwear, Hiking Boots }	2
{ Clothes, Hiking Boots }	2
{ Outerwear, Footwear }	2
{ Clothes, Footwear }	2

Rules

Rule	Support	Conf.
Outerwear ⇒ Hiking Boots	33%	66.6%
Outerwear ⇒ Footwear	33%	66.6%
Hiking Boots ⇒ Outerwear	33%	100%
Hiking Boots ⇒ Clothes	33%	100%

Observations:

(1) - rules 'Ski Pants => Hiking Boots',
'Jackets => Hiking Boots'
do **not** have minimum support

- generalized rule 'Outerwear => Hiking Boots'
may have minimum support.

- if X=>Y has minimum support, so do its **ancestors**
X=>Y*, X*=>Y, X*=>Y*

where X* denotes an itemset which is derived from X by replacing one or more of its items with more general items and where |X*|=|X| holds.

(2) - rule 'Outerwear => Hiking Boots' has minimum confidence

- generalized rule may not have minimum confidence:
'Clothes => Hiking Boots' has only confidence of 50%

- if X => Y has minimum confidence, so does X=>Y*.

Rules X*=>Y and X*=>Y* may not have minimum confidence.

(3) - support for an item x is not equal to the sum of the supports of its sub-concepts: several sub-concepts may be present in a single transaction

VII.3.4.2 Basic Algorithm

- **Def.:** Transaction t **supports** an itemset X, if for each item x $\in$X t contains x or some sub-concept(s) of x.
- Check becomes simpler if we first add to t all (direct/indirect) super-concepts of all its items; we call the resulting transaction t‘ an **extended** transaction.
- **Lemma:** t supports **X** iff[1] t‘ is a superset of X.

[1]) iff = if and only if ($\Leftrightarrow$)

- algorithm consists of 3 steps:

(1) find all frequent item sets

```
L1 := {frequent 1-itemsets};
k := 2; // k represents the pass number
while ( Lk-1 ≠ ∅ ) do
begin
   Ck := New candidates of size k generated from Lk-1.
   forall transactions t ∈ D do
   begin
      Add all ancestors of each item in t to t, removing
         any duplicates.
      Increment the count of all candidates in Ck that
         are contained in t.
   end
   Lk := All candidates in Ck with minimum support.
   k := k + 1;
end
Answer := ⋃k Lk;
```

(2) generate all rules with minimum support and minimum confidence

(3) evtl. prune all uninteresting rules (see next slide)

VI.3.4.3 Interesting Rules [Srikant/Agrawal 1995]

- structure of taxonomy can be used to **prune** redundant rules

Example:

- Assume ‘Outerwear $\rightarrow$ Hiking Boots‘ with support 8%, confidence 70%.
- Assume that 25% of ‘Outerwear‘-transactions are ‘Ski Pants‘-transactions.
- We therefore **expect** the rule

  Ski Pants $\rightarrow$ Hiking Boots

  to hold with support 2% and confidence 70%.
- If such a rule is generated, the more special rule is **redundant** when compared to the more general rule.

Definition: A rule $X_3 \rightarrow Y_3$ is a **close ancestor** of a rule $X_1 \rightarrow Y_1$, if there is no ancestor $X_2 \rightarrow Y_2$ of $X_1 \rightarrow Y_1$, which has $X_3 \rightarrow Y_3$ as ancestor.

Definition: **Interesting Rule**

Let $r \geq 1$ be a user-specified interest factor.

A rule is r-interesting with respect to one of its ancestors if its support is at least r times as high as the support expected based on the ancestor, or if its confidence is at least r times as high as the confidence expected based on the ancestor.

A rule is **interesting** if it has no ancestors or if it is r-interesting with respect to all its close ancestors.

Example: (Srikant/Agrawal 1995)

Rule #	Rule	Support
1	"Clothes ⇒ Footwear"	10
2	"Outerwear ⇒ Footwear"	8
3	"Jackets ⇒ Footwear"	4

Item	Support
Clothes	5
Outerwear	2
Jackets	1

Let r=2.

- Rule 1 is interesting, since it has no ancestors.
- Rule 2 is interesting, since its support is at least r times the support expected based on its only ancestor (rule 1) (and the same holds for its confidence):

(i) $\frac{\text{support(Outerwear)}}{\text{support(Clothes)}} = \frac{2}{5}$

(ii) expected_support(Outerwear → Footwear) =

$2/5 \cdot \text{support(Clothes} \rightarrow \text{Footwear)} = 2/5 \cdot 10 = 4$

(iii) support(Outerwear → Footwear) = $8 \geq 2 \cdot 4 = r \cdot 4$

- Rule 3 is **not interesting**, since its support is less than r times the support expected based on rule 2:

(i) $\frac{\text{support(Jackets)}}{\text{support(Outerwear)}} = \frac{1}{2}$

(ii) expected_support(Jackets → Footwear) =

½ · support(Outerwear → Footwear) = ½ · 8 = 4

(iii) support (Jackets → Footwear) = 4 < 2·4 = r · 4

Remark:

In practise, the measure of interest is able to prune between 40% and 60% of all generated rules.

Remarks:

- basic algorithm can be made more efficient by applying several optimization steps,
  - e.g. pre-compute all super-concepts of all available items
- taxonomy provides means for
  - generating **more useful** rules
  - **pruning uninteresting** rules

Examples of real data:

- **supermarket** data:
  - 548.000 items
  - taxonomy has 4 levels, 118 roots
  - ≈ 1.5 million transactions with an average of 9.6 items per transaction
  - optimized algorithm needs ≈ 90 minutes to generate all rules with support of 1%
- **department store** data:
  - 228.000 items
  - taxonomy has 7 levels, 89 roots
  - ≈ 570.000 transactions with an average of 4.4 items per transaction
  - optimized algorithm needs ≈ 5 minutes to generate all rules with support of 1%