Assoziationsregeln & Sequenzielle Muster

Übersicht

- Grundlagen für Assoziationsregeln
- Apriori Algorithmus
- Verschiedene Datenformate
- Finden von Assoziationsregeln mit mehren unteren Schranken für Unterstützung
- Finden von Assoziationsregeln für Klassifikation
- Finden von sequenziellen Mustern
- Zusammenfassung

Finden von Assoziationsregeln

- Vorgeschlagen von Agrawal et al, 1993.
- Es ist ein wichtiges Data Mining Modell, dass ausgiebig von Datenbank- und Data Mining Forschern bearbeitet wurde.
- Annahme: Alle Daten (Attribute) sind kategorisch
- Kein guter Algorithmus für numerische Daten bekannt
- Wurde anfänglich für Warenkorb-Analyse genutzt, um Waren zu finden die oft von Kunden gemeinsam gekauft werden

Bread $\rightarrow$ Milk [sup = 5%, conf = 100%]

Das Modell: Daten

- $I = \{i_1, i_2, \ldots, i_m\}$: eine Menge von *Artikeln*.
- Transaktion t :
  - t eine Menge von Artikeln und $t \subseteq I$.
- Transaktionsdatenmenge T: Menge von Transaktionen $T = \{t_1, t_2, \ldots, t_n\}$.

Transaktionsdaten:Supermarkt Daten

- Warenkorb Transaktionen:

  t1: {bread, cheese, milk}

  t2: {apple, eggs, salt, yogurt}

  … …

  tn: {biscuit, eggs, milk}

- Begriffe:
  - *Artikel*: ein Artikel in einem Warenkorb
  - *I*: Menge der in einem Markt verkauften Artikel
  - *Transaktion*: zusammengekaufte Artikel in einem Warebkorb; kann eine TID haben (Transaktions-ID)
  - *Transaktionsdaten*: Menge der Transaktionen

Transaktionsdaten:z.B. eine Menge von Dokumenten

- Datenmenge über Textdokumente
  Jedes Dokument wird als eine Menge von Schlüsselworten aufgefaßt

doc1:	Student, Teach, School
doc2:	Student, School
doc3:	Teach, School, City, Game
doc4:	Baseball, Basketball
doc5:	Basketball, Player, Spectator
doc6:	Baseball, Coach, Game, Team
doc7:	Basketball, Team, City, Game

Das Modell: Regeln

- Eine Transaktion *t* enthält *X*, eine Menge von Artikeln (Artikelmenge) aus *I*, wenn $X \subseteq t$.
- Eine Assoziationsregel ist eine Implikation der Form:

  $X \rightarrow Y$, mit $X, Y \subset I$ und $X \cap Y = \varnothing$

- Eine Artikelmenge ist eine beliebige Menge von Artikeln.
  - z.B., X = {milk, bread, cereal} ist eine Artikelmenge.
- Eine *k*-Artikelmenge ist eine Artikelmenge mit *k* Artikeln.
  - z.B., {milk, bread, cereal} ist eine 3-Artikelmenge

Gütemaße für Regeln

- Unterstützung: eine Regel hat die Unterstützung *sup* in *T* (der Transaktionsdatenmenge) wenn sup% der Transaktionen $X \cup Y$ enthalten.
  - $sup = \Pr(X \cup Y)$.
- Konfidenz: eine Regel hält in *T* mit Konfidenz *conf*, wenn *conf*% der Transaktionen, die *X* enthalten auch *Y* enthalten.
  - $conf = \Pr(Y \mid X)$
- Eine Assoziationsregel ist ein Muster, das besagt, wenn *X* auftritt, tritt auch *Y* mit einer gewissen Wahrscheinlichkeit auf.

Unterstützung und Konfidenz

- Anzahl der Unterstützungen: die Anzahl der Unterstützungen einer Artikelmenge *X*, (geschrieben *X.count)*, in einer Datenmenge *T* ist die Anzahl der Transaktionen in *T*, die *X* enthalten. Angenommen *T* hat *n* Transaktionen.
- Dann,

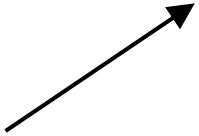
$$support = \frac{(X \cup Y).count}{n}$$

$$confidence = \frac{(X \cup Y).count}{X.count}$$

Ziel und wichtige Eigenschaften

- **Ziel:** Finde alle Regeln, die die vom Anwender vorgegebenen unteren Schranken für Unterstützung (minsup) und Konfidenz (minconf)
- **Wichtige Eigenschaften**
  - Vollständigkeit: finde alle Regeln.
  - Keine Zielartikel auf der rechten Seite
  - Analysiere die Daten ohne sie komplett in den Hauptspeicher zu laden, Daten bleiben auf der Festplatte

Beispiel



t1:	Beef, Chicken, Milk
t2:	Beef, Cheese
t3:	Cheese, Boots
t4:	Beef, Chicken, Cheese
t5:	Beef, Chicken, Clothes, Cheese, Milk
t6:	Chicken, Clothes, Milk
t7:	Chicken, Milk, Clothes

- Transaktionen
- Annahme:
  minsup = 30%
  minconf = 80%
- Ein Beispiel für eine häufige Artikelmenge:

{Chicken, Clothes, Milk} [sup = 3/7]

- Assoziationsregeln aus dieser Artikelmenge:

Clothes $\rightarrow$ Milk, Chicken [sup = 3/7, conf = 3/3]

… …

Clothes, Chicken $\rightarrow$ Milk, [sup = 3/7, conf = 3/3]

Repräsentation von Transaktionsdaten

- Ein vereinfachter Darstellung von Warenkörben
- Wichtige Informationen sind nicht berücksichtigt, z.B,
  - die Menge jedes gekauften Artikels
  - der gezahlte Preis
- Es wird nur repräsentiert, was gekauft wurde, nicht gekaufte Artikel sind nur impliziet dargestellt

Viele Algorithmen

- Es gibt sehr viele Algorithmen für das Problem!!
- Sie unterscheiden sich in der genutzten Suchstrategie und Datenstrukturen.
- Die gefundenen Regeln sind bei gleichen Parametern immer dieselben.
  - Für gegebene Transaktionsdaten *T* und untere Schranken für Unterstützung und Konfidenz ist die Menge der Assoziationsregeln eindeutig bestimmt.
- Jeder Algorithmus sollte die gleiche Regelmenge finden, sie unterscheiden sich aber in Effizienz und Speicherverbrauch
- Wir behandeln nur einen: den Apriori Algorithmus

Übersicht

- Grundlagen für Assoziationsregeln
- Apriori Algorithmus
- Verschiedene Datenformate
- Finden von Assoziationsregeln mit mehren unteren Schranken für Unterstützung
- Finden von Assoziationsregeln für Klassifikation
- Finden von sequenziellen Mustern
- Zusammenfassung

Der Apriori Algorithmus

- **Einer der besten bekannten Algorithmen**
- **Zwei Schritte**:
  - Finde alle Artikelmenge, die die untere Schranke für Unterstützung erfüllen (*häufige Artikelmengen*).
  - Nutze die häufigen Artikelmengen um die Regeln zu erzeugen.

- z.B., eine häufige Artikelmenge

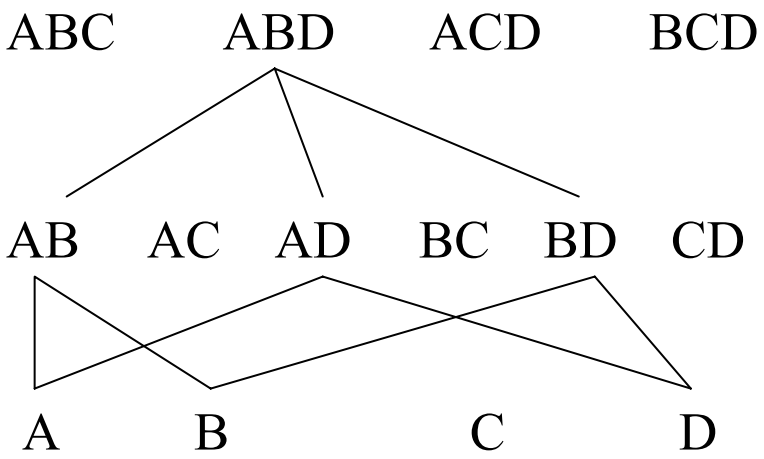
  {Chicken, Clothes, Milk} [sup = 3/7]

  und eine daraus erzeugte Regel

  Clothes $\rightarrow$ Milk, Chicken [sup = 3/7, conf = 3/3]

Schritt 1: Finde alle häufigen Artikelmengen

- Eine häufige Artikelmenge ist eine Artikelmenge, deren Unterstützung $\geq$ minsup ist.
- Hauptidee: die Apriori Eigenschaft (Monotonie): jede Teilmenge einer häufigen Artikelmenge muß auch eine häufige Artikelmenge sein



Der Algorithmus

- Iterativer Algorithmus. (auch Breitensuche): Finde alle häufigen 1-Artikelmengen; dann alle häufigen 2-Artikelmengen, und so weiter.
  - In jeder Iteration *k,* betrachte nur solche Artikelmengen, die häufige *k*-1 Artikelmengen enthalten.

- Finde alle häufigen Artikelmengen der Größe 1: F_1
- Ab $k = 2$
  - C_k = Kandidaten der Größe *k*: die Artikelmengen der Größe *k*, die häufig sein könnten bei gegebenem F_{k-1}
  - F_k = die Artikelmengen, die tatsächlich häufig sind, $F_k \subseteq C_k$ (die Transaktionen in *T* müssen einmal durchlaufen werden).

Beispiel – Finden von häufigen Artikelmengen

Daten T
minsup=0.5

TID	Items
T100	1, 3, 4
T200	2, 3, 5
T300	1, 2, 3, 5
T400	2, 5

Artikelmenge:Anzahl

1. scan T ➔ C_1: {1}:2, {2}:3, {3}:3, {4}:1, {5}:3

   ➔ F_1: {1}:2, {2}:3, {3}:3, {5}:3

   ➔ C_2: {1,2}, {1,3}, {1,5}, {2,3}, {2,5}, {3,5}

2. scan T ➔ C_2: {1,2}:1, {1,3}:2, {1,5}:1, {2,3}:2, {2,5}:3, {3,5}:2

   ➔ F_2: **{1,3}**:2, **{2,3}**:2, **{2,5}**:3, **{3,5}**:2

   ➔ C_3: {2, 3,5}

3. scan T ➔ C_3: **{2, 3, 5}**:2

   ➔ F_3: **{2, 3, 5}**

Details: ordnen der Artikel

- Die Artikel in *I* sind geordnet in lexikografischer Ordnung (die eine totale Ordnung ist).
- Die Ordnung wird im gesamten Algorithmus bei jeder Artikelmenge implizit genutzt.
- $\{w[1], w[2], \ldots, w[k]\}$ repräsentiert eine k-Artikelmenge w bestehend aus den Artikeln $w[1], w[2], \ldots, w[k]$, mit $w[1] < w[2] < \ldots < w[k]$ entsprechend der totalen Ordnung.

Details: der Algorithmus

Algorithm Apriori(*T*)

$C_1 \leftarrow$ init-pass(T);
$F_1 \leftarrow \{f \mid f \in C_1, f.\text{count}/n \geq minsup\}$; // n: no. of transactions in T
for (k = 2; $F_{k-1} \neq \emptyset$; k++) **do**
  $C_k \leftarrow$ candidate-gen(F_{k-1});
  for each transaction $t \in T$ **do**
    for each candidate $c \in C_k$ **do**
      if c is contained in t **then**
        $c.count$++;
    end
  end
  $F_k \leftarrow \{c \in C_k \mid c.count/n \geq minsup\}$
end
return $F \leftarrow \bigcup_k F_k$;

Apriori Kandidatenerzeugung

- Die Funktion candidate-gen nimmt F_{k-1} und gibt eine Obermenge (genannt Kandidaten) der Menge aller häufigen k-Artikelmengen zurück. Sie hat zwei Schritte
  - *Join* Schritt: Erzeuge alle möglichen Kandidatenartikelmengen C_k der Größe k
  - *Prune* Schritt: Entferne die Kandidaten aus C_k, die nicht häufig sein können, weil sie eine nicht-häufige Teilmenge enthalten.

Candidate-gen Funktion

Function candidate-gen(F_{k-1})

$C_k \leftarrow \emptyset$;

forall $f_1, f_2 \in F_{k-1}$

with $f_1 = \{i_1, \ldots, i_{k-2}, i_{k-1}\}$

and $f_2 = \{i_1, \ldots, i_{k-2}, i'_{k-1}\}$

and $i_{k-1} < i'_{k-1}$ **do**

$c \leftarrow \{i_1, \ldots, i_{k-1}, i'_{k-1}\}$; // join f_1 and f_2

$C_k \leftarrow C_k \cup \{c\}$;

for each (k-1)-subset s of c **do**

if ($s \notin F_{k-1}$) **then**

delete c from C_k; // prune

end

end

return C_k;

Ein Beispiel

- F_3 = {{1, 2, 3}, {1, 2, 4}, {1, 3, 4}, {1, 3, 5}, {2, 3, 4}}

- Nach dem Join
  - C_4 = {{1, 2, 3, 4}, {1, 3, 4, 5}}
- Nach dem Pruning:
  - C_4 = {{1, 2, 3, 4}}

    weil {1, 4, 5} nicht in F_3 ist, wird {1, 3, 4, 5} entfernt

Schritt 2: Erzeugen der Regeln aus häufigen Artikelmengen

- Häufige Artikelmengen ≠ Assoziationsregeln
- Ein weiterer Schritt wird gebraucht um Assoziationsregeln zu erzeugen
- Für jede häufige Artikelmenge *X*,

  Für jede echte nichtleere Teilmenge *A* von *X*
  - Sei *B* = X - *A*
  - A $\rightarrow$ B ist eine Assoziationsregel, wenn
    - Confidenz(A $\rightarrow$ B) $\geq$ minconf,

      support(A $\rightarrow$ B) = support(A$\cup$B) = support(X)

      confidence(A $\rightarrow$ B) = support(A $\cup$ B) / support(A)

Erzeugen der Regeln: Beispiel

- Angenommen {2,3,4} ist häufig mit sup=50%
  - Echte Teilmengen: {2,3}, {2,4}, {3,4}, {2}, {3}, {4}, mit sup=50%, 50%, 75%, 75%, 75%, 75%
  - Diese erzeugen die Assoziationsregeln:
    - $2,3 \rightarrow 4$, Konfidenz=100%
    - $2,4 \rightarrow 3$, Konfidenz=100%
    - $3,4 \rightarrow 2$, Konfidenz=67%
    - $2 \rightarrow 3,4$, Konfidenz=67%
    - $3 \rightarrow 2,4$, Konfidenz=67%
    - $4 \rightarrow 2,3$, Konfidenz=67%
    - Alle Regeln haben eine Unterstützung von sup = 50%

Erzeugen der Regeln: Zusammenfassung

- Um die Konfidenz von $A \rightarrow B$ zu bestimmen, brauchen wir die Unterstützung von $(A \cup B)$ und von A
- Alle benötigte Informationen zur Berechnung der Konfidenz sind bereits während der Erzeugung der häufigen Artikelmengen bestimmt worden. Die Transaktionsdaten T werden nicht mehr gebraucht.
- Dieser Schritt braucht meist viel weniger Zeit als die Suche nach den häufigen Artikelmengen

Apriori Algorithmus

Scheint sehr teuer zu sein

- Ebenen-weise Breitensuche
- K = die Größe der größten Transaktion
- Der Algorithmus macht höchstens K Durchläufe über die Daten
- In der Praxis ist K beschränkt (etwa 10).
- Für solche Daten ist der Algorithmus sehr schnell.
- Kann auf große Datenmengen skalieren

Mehr zum Finden von Assoziationsregeln

- Der Suchraum aller Assoziationsregeln ist exponentiell, $O(2^m)$, wobei m die Anzahl der Artikel in *I* ist.
- Der Algorithmus nutzt die kleinen Transaktionsgrößen und die hohen unteren Schranken für Unterstützung und Konfidenz.
- Jedoch werden trotzdem meist eine riesige Anzahl von Regeln erzeugt, tausende bis Millionen.

Übersicht

- Grundlagen für Assoziationsregeln
- Apriori Algorithmus
- Verschiedene Datenformate
- Finden von Assoziationsregeln mit mehren unteren Schranken für Unterstützung
- Finden von Assoziationsregeln für Oberklassen
- Finden von sequenziellen Mustern
- Zusammenfassung

Verschiedene Datenformate

- Die Daten können in Transaktionsform oder als Tabelle gespeichert sein.

Transaktionsform:

a, b

a, c, d, e

a, d, f

Tabelle :

Attr1	Attr2	Attr3
a,	b,	d
b,	c,	e

- Tabellen müssen in Transaktionsform konvertiert werden

Von einer Tabelle zu einer Transaktionsmenge

Tabelle:

Attr1	Attr2	Attr3
a,	b,	d
b,	c,	e

$\Rightarrow$Transaktionsform:

(Attr1, a), (Attr2, b), (Attr3, d)

(Attr1, b), (Attr2, c), (Attr3, e)

candidate-gen kann leicht verbessert werden.

Warum?