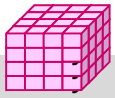
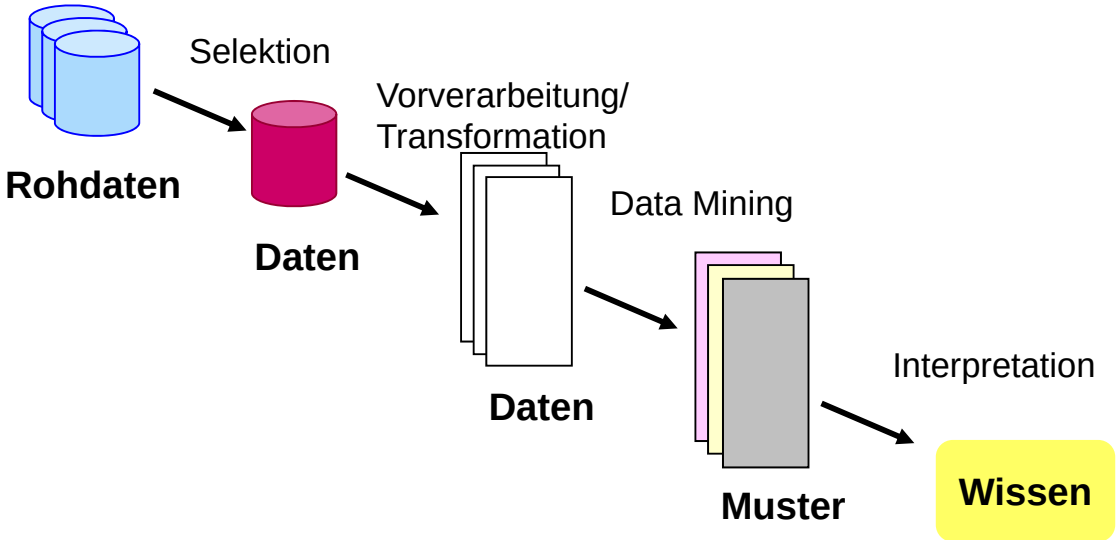
6. Überblick zu Data Mining-Verfahren

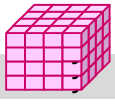
- Einführung
  - Data Mining-Prozeß
  - Anwendungsbeispiele
- Clusteranalyse
  - k-Means-Algorithmus
  - Canopy Clustering
- Klassifikation
  - Klassifikationsprozess
  - Konstruktion eines Entscheidungsbaums
- Assoziationsregeln / Warenkorbanalyse
  - Support und Konfidenz
  - A Priori-Algorithmus
  - Frequent Pattern (FP)-Trees



Knowledge Discovery in Databases (KDD)

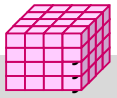
- (semi-)automatische Extraktion von Wissen aus Datenbanken, das
  - gültig (im statistischen Sinn)
  - bisher unbekannt
  - und potentiell nützlich ist
- Kombination von Verfahren zu Datenbanken, Statistik und KI (maschinelles Lernen)





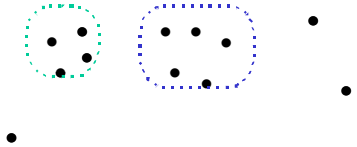
Data Mining

- Data Mining: Anwendung effizienter Algorithmen zur Erkennung von Mustern in großen Datenmengen
- bisher meist Mining auf speziell aufgebauten Dateien
- notwendig: Data Mining auf Datenbanken bzw. Data Warehouses
  - Portabilität
  - Skalierbarkeit auf große Datenmengen
  - Nutzung der DBS-Performance-Techniken
  - Integration mehrerer Datenquellen
  - Vermeidung von Redundanz und Inkonsistenzen
- Datenaufbereitung für Data Mining
  - Datenintegration und Datenbereinigung (data cleaning)
  - Diskretisierung numerischer Attribute (Aufteilung von Wertebereichen in Intervalle, z.B. Altersgruppen)
  - Erzeugen abgeleiteter Attribute (z.B. Aggregationen für bestimmte Dimensionen, Umsatzänderungen)
  - Einschränkung der auszuwertenden Attribute

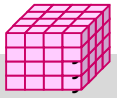


Data Mining: Techniken

- Clusteranalyse
  - Objekte werden aufgrund von Ähnlichkeiten in Klassen eingeteilt (Segmentierung)

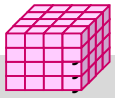


- Assoziationsregeln
  - Warenkorbanalyse (z.B. Kunde kauft A und B => Kunde kauft C)
  - Sonderformen zur Berücksichtigung von Dimensionshierarchien (z.B. Produktgruppen), quantitativen Attributen, zeitlichen Beziehungen (sequence mining)
- Klassifikation
  - Zuordnung von Objekten zu Gruppen/Klassen mit gemeinsamen Eigenschaften bzw. Vorhersage von Attributwerten
  - explizite Erstellung von Klassifikationsregeln (z.B. “guter Kunde” wenn Alter > 25 und ...)
  - Verwendung von Stichproben (Trainingsdaten)
  - Ansätze: Entscheidungsbaum-Verfahren, statistische Auswertungen (z.B. Maximum Likelihood-Schätzung / Bayes-Schätzer), neuronale Netze
- Weitere Ansätze:
  - Genetische Algorithmen (multivariate Optimierungsprobleme, z.B. Identifikation der besten Bankkunden)
  - Regressionsanalyse zur Vorhersage numerischer Attribute . . .



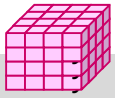
Data Mining: Anwendungsbeispiele

- Kundensegmentierung für Marketing
  - Gruppierung von Kunden mit ähnlichem Kaufverhalten / ähnlichen Interessen
  - Nutzung für gruppenspezifische Empfehlungen, Product Bundling, ...
- Warenkorbanalyse: Produkt-Platzierung im Supermarkt, Preisoptimierung, ...
- Bestimmung der Kreditwürdigkeit von Kunden
  - elektronische Vergabe von Kreditkarten
  - schnelle Entscheidung über Versicherungsanträge, ...
  - Technik: Entscheidungsbaum-Klassifikator
- Entdeckung wechselbereiter Kunden
- Entdeckung von Kreditkarten-Missbrauch
- Unterstützung im Data Cleaning
- Web Usage Mining
- Text Mining: inhaltliche Gruppierung von Dokumenten, E-Mails, ...



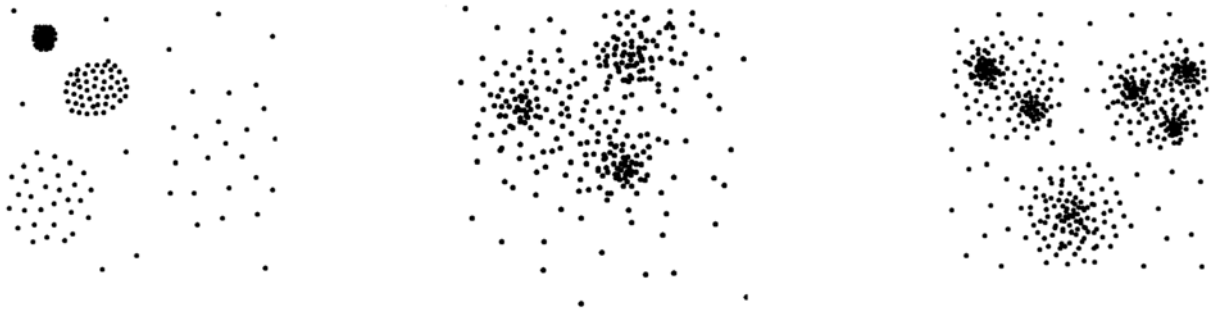
Evaluation/Interpretation

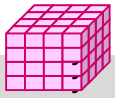
- Ablauf
  - Präsentation der gefundenen Muster, z.B. über Visualisierungen
  - Bewertung der Muster durch den Benutzer
  - falls schlechte Bewertung: erneutes Data Mining mit anderen Parametern, anderem Verfahren oder anderen Daten
  - falls gute Bewertung: Integration des gefundenen Wissens in die Wissensbasis / Metadaten und Nutzung für zukünftige KDD-Prozesse
- Bewertung der gefundenen Muster: Interessantheit, Vorhersagekraft
  - sind Muster schon bekannt oder überraschend?
  - wie gut lassen sich mit „Trainingsdaten“ (Stichproben) gefundene Muster auf zukünftige Daten verallgemeinern?
  - Vorhersagekraft wächst mit Größe und Repräsentativität der Stichprobe



Clusteranalyse

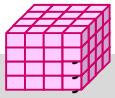
- Ziele
  - automatische Identifikation einer endlichen Menge von Kategorien, Klassen oder Gruppen (Cluster) in den Daten
  - Objekte im gleichen Cluster sollen möglichst ähnlich sein
  - Objekte aus verschiedenen Clustern sollen möglichst unähnlich zueinander sein
- Ähnlichkeitsbestimmung
  - meist: Distanzfunktion dist(o1,o2) für Paare von Objekten o1 und o2
    - z.B. Euklidische Distanz für numerische Attribute: $dist(x,y) = \sqrt{\sum_{i=1}^{d} (x_i - y_i)^2}$
  - spezielle Funktionen für kategorische Attribute oder Textdokumente
- Clustering-Ansätze: partitionierend, hierarchisch, dichtebasiert, ...





K-Means Algorithmus

- Ausgangssituation
  - Objekte besitzen Distanzfunktion
  - für jedes Cluster kann ein Clusterzentrum bestimmt werden („Mittelwert“)
  - Anzahl k der Cluster wird vorgegeben
- Basis-Algorithmus
  - Schritt 1 (Initialisierung): k Clusterzentren werden (zufällig) gewählt
  - Schritt 2 (Zuordnung): Jedes Objekt wird dem nächstgelegenen Clusterzentrum zugeordnet
  - Schritt 3 (Clusterzentren): Für jedes Cluster wird Clusterzentrum neu berechnet
  - Schritt 4 (Wiederholung): Abbruch, wenn sich Zuordnung nicht mehr ändert, sonst zu Schritt 2
- Probleme
  - Konvergenz zu lokalem Minimum, d.h. Clustering muss nicht optimal sein
    - Work-around: Algorithmus mehrfach starten
  - relativ hoher Aufwand für Abstandsberechnungen, Neuberechnung der Clusterzentren



K-Means Algorithmus: Beispiel

- Clustering der Zahlen 1, 3, 6, 14, 17, 24, 26, 31 in drei Cluster

(1) Zentren: 10, 21, 29 (zufällig gewählt)

(2) Cluster:

(3) Zentren (arithmetisches Mittel):

(2) Cluster:

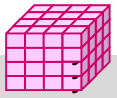
(3) Zentren:

(2) Cluster:

(3) Zentren:

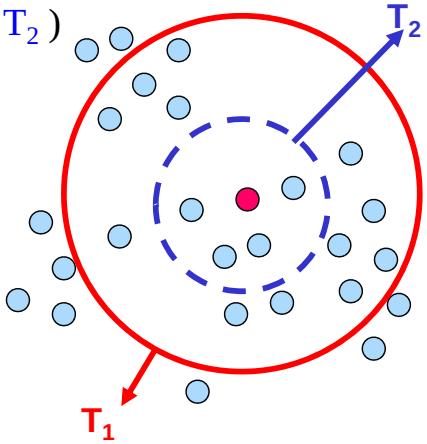
(2) Cluster:

Abbruch, da sich das Clustering nicht mehr geändert hat.

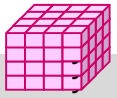


Canopy Clustering Algorithmus*

- Bildung von überlappenden Clustern (Canopies)
  - oft Nutzung als erster Schritt in mehrstufigem Analyseverfahren
  - skalierbar auf sehr große Datenmengen
  - auf Stringdaten anwendbar (Nutzung indexbasierter Ähnlichkeitsfunktionen zur Distanzberechnung, z.B. TF/IDF)
- Gegeben: Distanzfunktion, zwei Schwellwerte T_1 und T_2 ($T_1 > T_2$)
- Algorithmus
  - Schritt 1 (Initialisierung): Kandidatenliste für Wahl der Canopyzentren wird mit allen Objekten initialisiert
  - Schritt 2 (Canopyzentrum): Canopyzentrum Z wird (zufällig) aus der Kandidatenliste gewählt
  - Schritt 3 (Zuordnung): Alle Objekte, deren Abstand zu Z geringer ist als Schwellwert T_1, werden (Canopy) Z zugeordnet
  - Schritt 4 (Kandidatenliste): Alle Objekte, die innerhalb des Schwellwertes T_2 zu Z liegen, werden aus Kandidatenliste gelöscht
  - Schritt 5 (Ende/Wiederholung): Abbruch, wenn Kandidatenliste leer ist, sonst zu Schritt 2



* *McCallum, A. et al.:**Efficient clustering of high-dimensional data sets with application to reference matching***
Proc. ACM KDD, 2000



Canopy Clustering Algorithmus: Beispiel

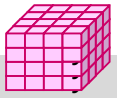
- Clustering der Zahlen 1, 3, 6, 11, 13

  Abstandsfunktion absolute Differenz, T_1= 8, T_2= 4
  (1) Kandidatenliste = {1, 3, 6, 11, 13}
  (2) Canopyzentrum
  (3) Canopy bilden
  (4) Entferne aus Kandidatenliste

  (5) Kandidatenliste =
  (2) Canopyzentrum:
  (3) Canopy bilden
  (4) Entferne aus Kandidatenliste

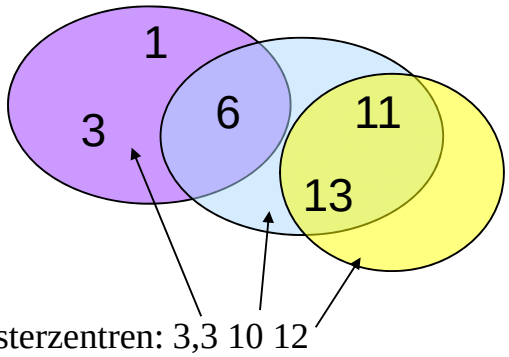
  (5) Kandidatenliste =
  (2) Canopyzentrum:
  (3) Canopy bilden
  (4) Entferne aus Kandidatenliste

  (5) Abbruch, da Kandidatenliste leer

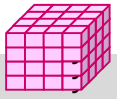


Kombination mit K-Means

- 1. Schritt: Canopy Clustering
- 2. Schritt: K-Means
  - für jedes Canopy wird mindestens ein Clusterzentrum bestimmt
  - statt für jedes Objekt die Distanz zu jedem Clusterzentrum zu bestimmen, werden nur die Distanzen zu den Clusterzentren bestimmt, die sich mit dem Objekt in denselben Canopies befinden.
- Beispiel:

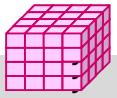


  - drei Clusterzentren: 3,3 10 12
  - für 1 und 3 keine Abstandsbestimmung notwendig, da sich nur ein Clusterzentrum im selben Canopy befindet
  - 6, 11 und 13 liegen jeweils in zwei Canopies, deshalb sind zwei Abstandsbestimmungen zu zwei Clusterzentren notwendig.



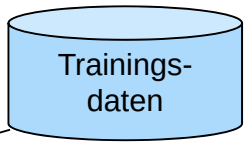
Klassifikation

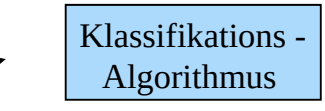
- Klassifikationsproblem
  - Gegeben sei Stichprobe (Trainingsmenge) O von Objekten des Formats $(a_1, \ldots, a_d)$ mit *Attributen* A_i, $1 \le i \le d$, und Klassenzugehörigkeit c_i, $c_i \in C = \{c_1, \ldots, c_k\}$
  - Gesucht: die Klassenzugehörigkeit für Objekte aus D \ O
    d.h. ein *Klassifikator* K : D → C
- weiteres Ziel:
  - Generierung (Lernen) des expliziten Klassifikationswissens (Klassifikationsmodell, z.B. Klassifikationsregeln oder Entscheidungsbaum)
- Abgrenzung zum Clustering
  - Klassifikation: Klassen vorab bekannt
  - Clustering: Klassen werden erst gesucht
- Klassifikationsansätze
  - Entscheidungsbaum-Klassifikatoren
  - Bayes-Klassifikatoren (Auswertung der bedingten Wahrscheinlichkeiten von Attributwerten)
  - Neuronale Netze



Klassifikationsprozess

1. Konstruktion des Klassifikationsmodells





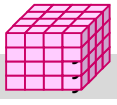
NAME	RANK	YEARS	TENURED
Mike	Assistant Prof	3	no
Mary	Assistant Prof	7	yes
Bill	Professor	2	yes
Jim	Associate Prof	7	yes
Dave	Assistant Prof	6	no
Anne	Associate Prof	3	no

if rank = 'professor'
or years > 6
then tenured = 'yes'

2. Anwendung des Modells zur Vorhersage (Prediction)

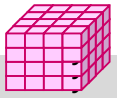
(Jeff, Professor, 4)

Tenured?



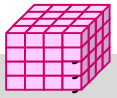
Bewertung von Klassifikatoren

- Klassifikator ist für die Trainingsdaten optimiert
  - liefert auf der Grundgesamtheit der Daten evtl. schlechtere Ergebnisse (-> Overfitting-Problem)
- Bewertung mit von Trainingsmengen unabhängigen Testmengen
- Gütemasse für Klassifikatoren
  - Klassifikationsgenauigkeit
  - Kompaktheit des Modells, z.B. Größe eines Entscheidungsbaums
  - Interpretierbarkeit des Modells (wie viel Einsichten vermittelt das Modell dem Benutzer?)
  - Effizienz der Konstruktion des Modells sowie der Anwendung des Modells
  - Skalierbarkeit für große Datenmengen
  - Robustheit gegenüber Rauschen und fehlenden Werten
- Klassifikationsgenauigkeit: Anteil der korrekten Klassenzuordnungen in Testmenge
- Klassifikationsfehler: Anteil der falschen Klassenzuordnungen



Bewertung (2)

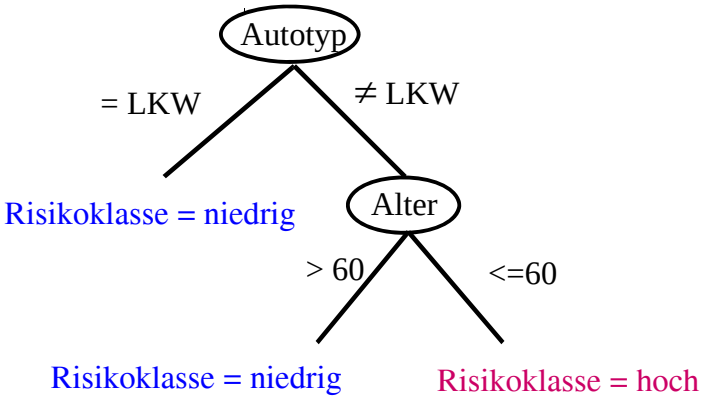
- Klassifikationsgenauigkeit (*Accuracy*):
  - n Sätze/Objekte, n_r korrekt zugeordnet
  - Accuracy $a = n_r / n$
  - alle Kategorien gehen gleichberechtigt ein
- Maße *Recall* / *Precision* / *F-Measure* fokussieren auf eine der Kategorien C (z.B. Risiko, Match etc.) in binären Entscheidungsproblemen
  - seien n_c der n Objekte von dieser Katerorie
  - Klassifikationsmodell ordne m_c Objekte dieser Klasse zu, davon m_{cr} richtig
  - Recall $r = m_{cr} / n_{c,}$ Precision $p = m_{cr} / m_{c,}$ F-Measure $f = 2\ r\ \ p\ /(r+p)$
- Beispiel binäres Entscheidungsproblem
  - $n=100$, $n_c = 50$, $m_c = 60$, $m_{cr} = 40$



Entscheidungsbäume

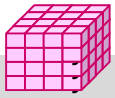
- explizite, leicht verständliche Repräsentation des Klassifikationswissens

ID	Alter	Autotyp	Risiko
1	23	Familie	hoch
2	17	Sport	hoch
3	43	Sport	hoch
4	68	Familie	niedrig
5	32	LKW	niedrig



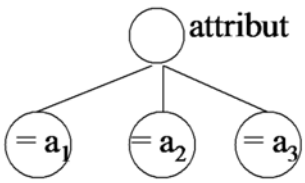
Regeldarstellung:

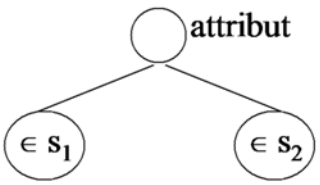
- Entscheidungsbaum ist Baum mit folgenden Eigenschaften:
  - ein innerer Knoten repräsentiert ein Attribut
  - eine Kante repräsentiert einen Test auf dem Attribut des Vaterknotens
  - ein Blatt repräsentiert eine der Klassen
- Anwendung zur Vorhersage:
  - Top-Down-Durchlauf des Entscheidungsbaums von der Wurzel zu einem der Blätter
  - eindeutige Zuordnung des Objekts zur Klasse des erreichten Blatts

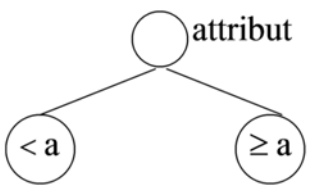


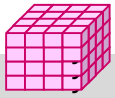
Konstruktion eines Entscheidungsbaums

- Basis-Algorithmus (divide and conquer)
  - Anfangs gehören alle Trainingsdatensätze zur Wurzel
  - Auswahl des nächsten Attributs (Split-Strategie): Maximierung des Informationsgewinns (meßbar über Entropie o.ä.)
  - Partitionierung der Trainingsdatensätze mit Split-Attribut
  - Verfahren wird rekursiv für die Partitionen fortgesetzt
- Abbruchbedingungen
  - keine weiteren Split-Attribute
  - alle Trainingsdatensätze eines Knotens gehören zur selben Klasse
- Typen von Splits
  - *Kategorische Attribute*: Split-Bedingungen der Form „attribut = a“ oder „attribut ∈ set“ (viele mögliche Teilmengen)
  - *Numerische Attribute*: Split-Bedingungen der Form „attribut < a“ (viele mögliche Split-Punkte)



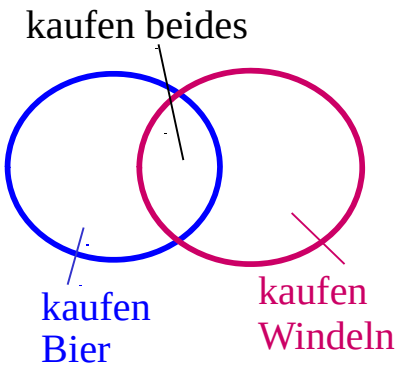


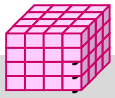




Assoziationsregeln

- Warenkorbanalyse auf Transaktions-Datenbank
  - Transaktion umfasst alle gemeinsam getätigten Einkäufe, innerhalb eines Dokuments vorkommenden Worte, innerhalb einer Web-Sitzung referenzierten Seiten, ...
- Regeln der Form "Rumpf → Kopf [support, confidence]"
- Beispiele
  - kauft("Windeln") → kauft("Bier") [0.5%, 60%]
  - 80% aller Kunden, die Reifen und Autozubehör kaufen, bringen ihr Auto auch zum Service
- Relevante Größen
  - Support einer Regel X → Y: Anteil der Transaktionen, in denen alle Objekte X und Y vorkommen
  - Konfidenz einer Regel X → Y: Anteil der Transaktionen mit Rumpf-Objekten X, für die Regel erfüllt ist (d.h. für die auch Objekte Y vorliegen)
  - Interessantheit: hoher Wahrscheinlichkeitsunterschied für Y gegenüber zufälliger Verteilung

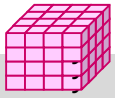




Assoziationsregeln (2)

- Aufgabe: Bestimmung aller Assoziationsregeln, deren Support und Konfidenz über bestimmten Grenzwerten liegen
- Gegeben:
  - R eine Menge von Objekten (z.B. Produkte, Webseiten)
  - t eine Transaktion, $t \subseteq R$
  - r eine Menge von Transaktionen
  - $S_{min} \in [0,1]$ die minimale Unterstützung,
  - $Conf_{min} \in [0,1]$ die minimale Konfidenz
- Aufgabe: Finde alle Regeln c der Form X → Y, wobei $X \subseteq R$, $Y \subseteq R$, $X \cap Y = \{\}$

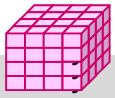
$$supp(r,c) = \frac{\left|\{t \in r \mid X \cup Y \in t\}\right|}{|r|} \geq s_{min} \qquad conf(r,c) = \frac{\left|\{t \in r \mid X \cup Y \in t\}\right|}{\left|\{t \in r \mid X \in r\}\right|} \geq conf_{min}$$



Beispiel: Warenkorbanalyse

Aftershave	Bier	Chips	EinkaufsID
0	1	1	1
1	1	0	2
0	1	1	3
1	0	0	4

{Aftershave} →{Bier} s = ¼, conf = ½
{Aftershave} → {Chips}
{Bier} → {Chips}
{Chips} →{Aftershave}
{Aftershave} →{Bier,Chips}



Weitere Beispiele

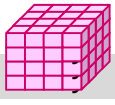
- Assoziationsregeln:
  - A → B
  - B → A

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

minsup = 50%,
minconf = 50%

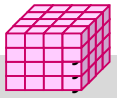
- Warenkörbe:
  - Drucker, Papier, PC, Toner
  - PC, Scanner
  - Drucker, Papier, Toner
  - Drucker, PC
  - Drucker, Papier, PC, Scanner, Toner

Assoziationsregel: PC → Drucker

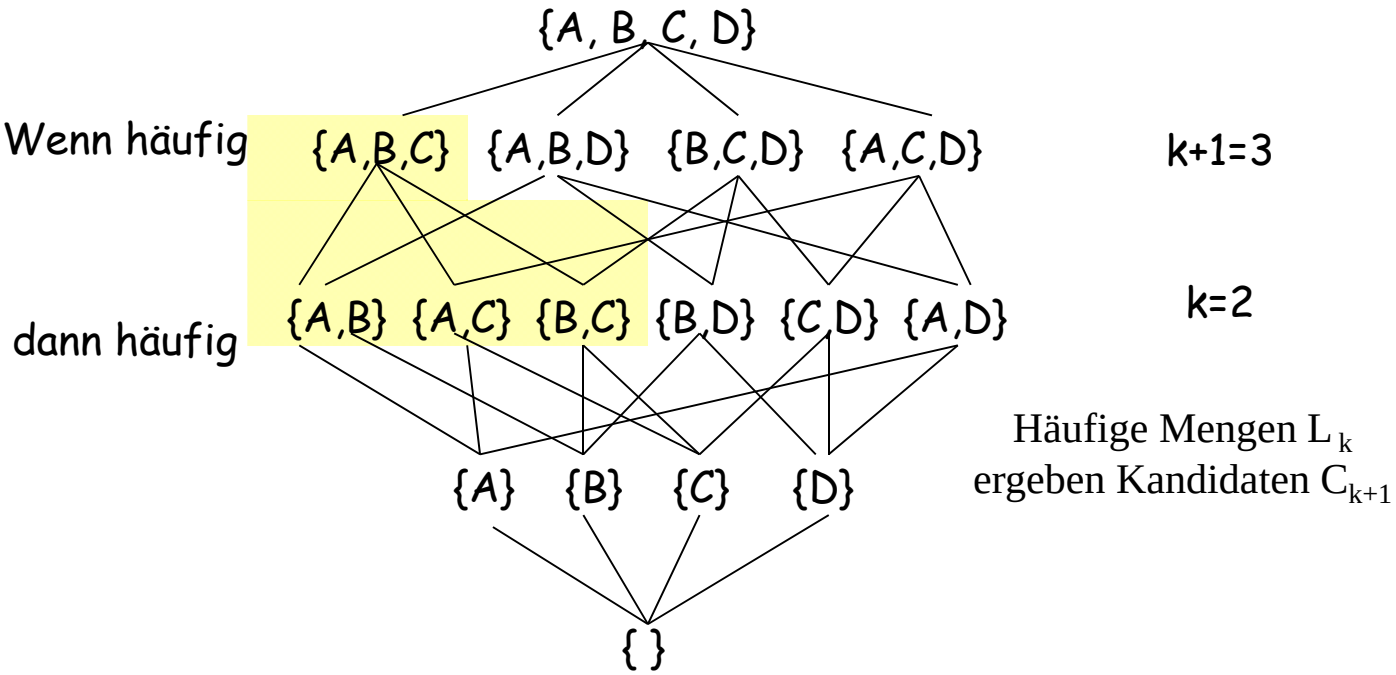


Frequent Itemsets

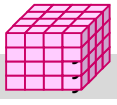
- Frequent Item-Set (häufige Mengen):
  - Menge von Items/Objekten, deren Support Schranke S_{min} übersteigt
- Bestimmung der Frequent-Itemsets wesentlicher Schritt zur Bestimmung von Assoziationsregeln
- effiziente Realisierung über A-Priori-Algorithmus
- Nutzung der sog. A-Priori-Eigenschaft:
  - wenn eine Menge häufig ist, so auch all ihre Teilmengen (Anti-Monotonie)
  - wenn eine Menge selten ist, so auch all ihre Obermengen (Monotonie)
- Support jeder Teilmenge und damit jedes einzelnen Items muss auch über Schranke S_{min} liegen
- Effiziente, iterative Realisierung beginnend mit 1-elementigen Itemsets
  - schrittweise Auswertung von k-Itemsets (k Elemente, k >=1),
  - Ausklammern von Kombinationen mit Teilmengen, die Support S_{min} nicht erreichen („Pruning“)
  - wird „A Priori“ getestet, bevor Support bestimmt wird



Verbandstruktur von Itemsets



Häufige Mengen L_k
ergeben Kandidaten C_{k+1}



A-Priori-Algorithmus

Apriori(R, r, s_{min}, $conf_{min}$)

L:= Häufige-Mengen(R, r, s_{min})

c:= Regeln (L, $conf_{min}$)

Return c.

Häufige-Mengen(R, r, s_{min})

k=1

L_1:= {i, supp(i)> s_{min}} // häufige Items

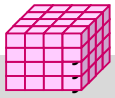
while $L_k \neq \{\ \}$

  C_{k+1} := Erzeuge-Kandidaten(L_k)

  L_{k+1} := Prune(C_{k+1}, r) // eliminiere Itemsets, die s_{min} nicht erreichen

  k:= k+1

Return $\bigcup_{j=2}^{k} L_j$



A-Priori-Algorithmus (2)

Erzeuge-Kandidaten(L_k)

C_{k+1} := {}

Forall l_1, l_2 in L_k , sodass $l_1 = \{i_1, ..., i_{k-1}, i_k\}$

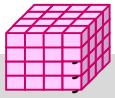
$l_2 = \{i_1, ..., i_{k-1}, i'_k\}$ $i_k < i'_k$

  l := $\{i_1, ..., i_{k-1}, i_k, i'_k\}$ // sortierte Items pro Itemset

  if alle k-elementigen Teilmengen von l in L_k sind

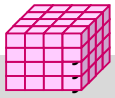
  then $C_{k+1} := C_{k+1} \cup \{l\}$

Return C_{k+1}



A-Priori-Algorithmus: Beispiel

- 5 PC-Warenkörbe (Forderung: minimaler Support: 60%)
  - Drucker, Papier, PC, Toner
  - PC, Scanner
  - Drucker, Papier, Toner
  - Drucker, PC
  - Drucker, Papier, PC, Scanner, Toner
- Algorithmus-Ausführung
  - k=1:
  - k=2:
  - k=3:
  - k=4:



Regelgenerierung

- Generierung der Assoziationsregeln aus den häufigen Mengen
  - wenn die Konklusion (rechte Seite) länger wird, kann die Konfidenz sinken.
  - die Ordnung der Attribute wird ausgenutzt:

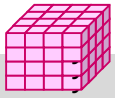
$l_1 = \{i_1, ..., i_{k-1}, i_k\}$ $c_1 = \{i_1, ..., i_{k-1}\} \rightarrow \{i_k\}$ $conf_1$

$l_1 = \{i_1, ..., i_{k-1}, i_k\}$ $c_2 = \{i_1, ...\} \rightarrow \{i_{k-1}, i_k\}$ $conf_2$ ...

$l_1 = \{i_1, ..., i_{k-1}, i_k\}$ $c_k = \{i_1\} \rightarrow \{..., i_{k-1}, i_k\}$ $conf_k$

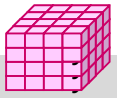
Es gilt dann: $conf_1 \geq conf_2 \geq ... \geq conf_k$

  - $conf_i = supp(l_1) / supp(i_1, ..., i_{k-i})$
  - Elimination aller Kombinationen, deren Konfidenz Minimalwert unterschreitet
- Beispiel: Regeln für ({Drucker, Papier, Toner}
  - conf ({Drucker, Papier} →{Toner}) =
  - conf ({Drucker} →{Papier, Toner}) =



Frequent Pattern-Baum (FP-Tree)

- Probleme des A-Priori-Algorithmus
  - exponentiell wachsende Anzahl zu prüfender Kandidaten
  - Bsp.: 10^4 häufige Objekte;
    $> 10^7$ 2-Itemsets; ca. 10^{30} Kandidaten für 100-Itemsets
- FP-Tree: Berechnung von Assoziationsregeln ohne Kandidatengenerierung
  - komprimierte Repräsentation aller Transaktionen durch Frequent-Pattern Tree
  - effiziente Erkennung von Frequent Itemsets (Pattern Mining) mit Divide-and-Conquer-Suche auf Teilbäumen
  - oft eine Größenordnung schneller als A-Priori



FP-Tree: Generierung

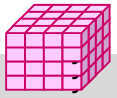
- Input: Transaktionsdatenbank D, Schwellwert min-support / s_{min}
- Scan von D, Erstellen der Menge F häufiger Objekte und ihrer Häufigkeiten, Ordnen von F in absteigender Häufigkeit.
- Wurzel des FP Trees ist leere Menge
- pro Transaktion in D: ordne Objekte gemäß F (absteigende Häufigkeit); füge Pfad in FP-Baum ein (Knotenformat: Objekt-Id, Zähler für Verwendungshäufigkeit in Transaktionen mit gleichem Präfix)
- Knoten mit gleicher Objekt-ID werden untereinander verkettet (ausgehend von Objekttabelle F)

TID	*Objekte*
10	{*a, c, d, f, g, i, m, p*}
20	{*a, b, c, f, l, m, o*}
30	{*b, f, h, j, o*}
40	{*b, c, k, p, s*}
50	{*a, c, e, f, l, m, n, p*}

Objekt	*count*
f	*4*
c	*4*
a	*3*
b	*3*
m	*3*
p	*3*

Objekt-Tabelle F

min_support = 0.5



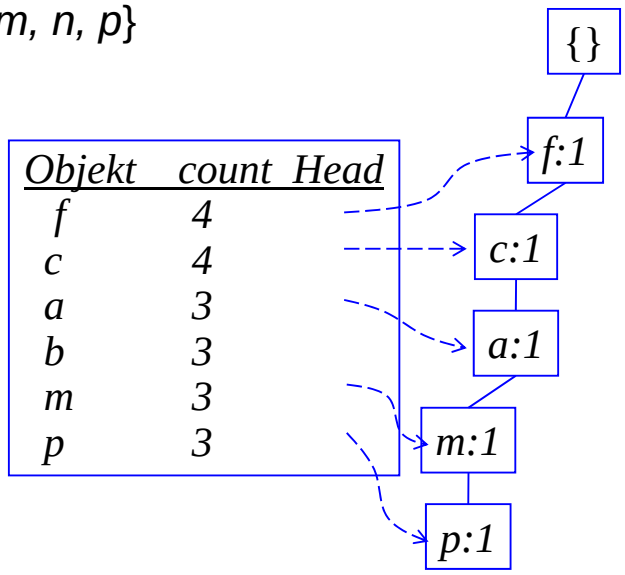
FP-Tree: Generierung (2)

TID	Objekte	gemäß F-Ordnung
10	{a, c, d, g, f, i, m, p}	{f, c, a, m, p}
20	{a, b, c, f, l, m, o}	
30	{b, f, h, j, o}	
40	{b, c, k, p, s}	
50	{a, c, e, f, l, m, n, p}	

min_support = 0.5

Objekt-Tabelle F

Objekt	count	Head
f	4	
c	4	
a	3	
b	3	
m	3	
p	3	



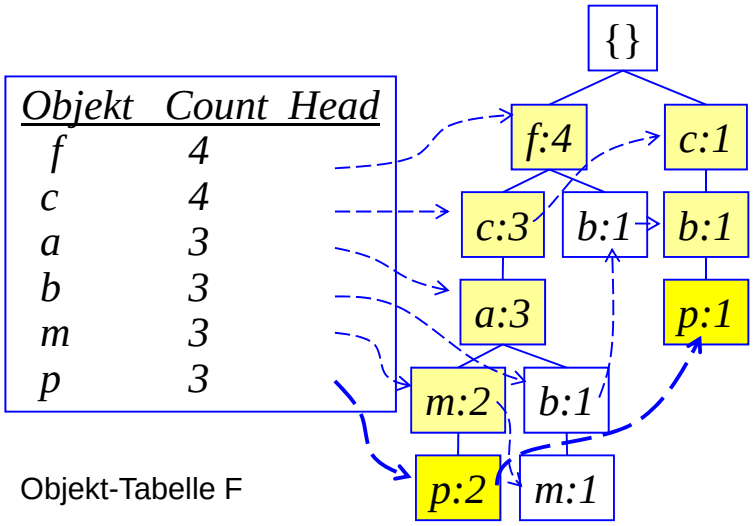
Pro Transaktion gibt es einen Pfad, wobei gemeinsame Präfixe nur einmal repräsentiert werden (Komprimierungseffekt)

FP-Tree: Erkennung häufiger Mengen

- Erkennung von Frequent Itemsets durch rekursive Suche auf Teilbäumen (Divide and Conquer)
- Erzeuge *Conditional Pattern Base* (“bedingte Basis“) für jeden Knoten im FP-Tree
  - gehe Objekt-Tabelle von unten (selten) nach oben durch. Die Verweise führen zu den Pfaden, in denen das Objekt vorkommt.
  - das Objekt wird als Suffix betrachtet und alle Präfixe davon als Bedingungen für dies Suffix. Die Präfixpfade eines Suffixes bilden seine „bedingte Basis“
- Erzeuge *Conditional FP-Tree* (bedingten FP-Baum) für jede *C. Pattern Base*
  - Häufigkeiten der Präfixe (für das betrachtete Suffix) werden von unten nach oben propagiert. Gleiche Präfixpfade zu einem Suffix (vom Anfang bis zu einer bestimmten Stelle) werden zusammengelegt und die ursprünglichen Häufigkeiten addiert.
  - nur Präfixpfade, die *min_support* erfüllen, bilden den „bedingten FP-Tree“
- Rekursives Ableiten von Frequent Itemsets aus bedingten FP-Trees
  - bei einzigem Pfad im Baum: Aufzählen aller Pattern

Schritt 1: Vom FP-Tree zur Cond. Pattern Base

- gehe Objekt-Tabelle von unten (selten) nach oben durch. Die Verweise führen zu den Pfaden, in denen das Objekt vorkommt.
- das Objekt wird als Suffix betrachtet und alle Präfixe davon als Bedingungen für dies Suffix. Die transformierten Präfixpfade eines Suffixes bilden seine „bedingte Basis“



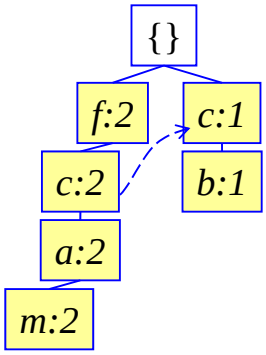
Objekt-Tabelle F

Objekt	Bedingte Pattern Basis
p	
m	
b	
a	
c	
f	

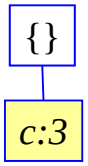
Schritt 2: Erzeuge Cond. FP-Trees

- Akkumuliere Counts pro Objekt der Cond. Pattern Base von unten nach oben.
- nur Knoten/Präfix-Pfade, die min_support erfüllen, bilden den „bedingten FP-Tree“

***p*-conditional pattern base**



***p*-conditional FP-Tree**

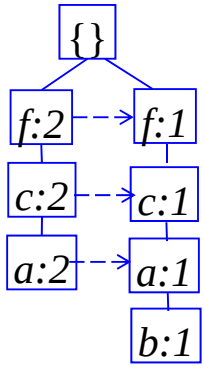


Frequent Itemsets:

p
cp

***m*-conditional FP-Tree**

***m*-conditional pattern base**



Schritt 3: Rekursive Abarbeitung der Cond. FP-Trees

{}
|
f:3
|
c:3
|
a:3

***m*-conditional FP-tree**

Cond. pattern base of “am”: (fc:3)

{}
|
f:3
|
c:3

***am*-conditional FP-tree**

Cond. pattern base of “cm”: (f:3)

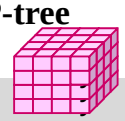
{}
|
f:3

***cm*-conditional FP-tree**

Cond. pattern base of “cam”: (f:3)

{}
|
f:3

***cam*-conditional FP-tree**



Schritt 3: Cond. FP-Trees mit einem Pfad

- Bei nur einem Pfad T kann rekursive Auswertung entfallen
- Bestimmung der Frequnet Itemsets durch Aufzählung aller Kombinationen von Teil-Pfaden

{}
|
f:3
|
c:3
|
a:3

m-conditional FP-Tree

➔

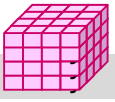
Frequent Itemsets bezüglich *m*

m,

fm, cm, am,

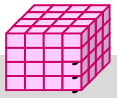
fcm, fam, cam,

fcam



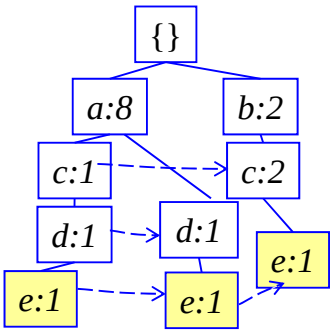
Ergebnisse im Beispiel

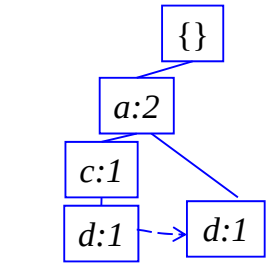
Objekt	Conditional Pattern Base	Conditional FP-Tree	Frequent Itemsets
p	{(fcam:2), (cb:1) }	{ (c:3)} | p	p, cp
m	{(fca:2), (fcab:1) }	{ (f:3, c:3, a:3)} | m	m, fm, cm, am, fcm, fam, cam, fcam
b	{(fca:1), (f:1), (c:1) }	{}	b
a	{(fc:3)}	{ (f:3, c:3)} | a	a, fa, ca, fca
c	{(f:3)}	{ (f:3)} | c	c, fc
f	{}	{}	f



Alternativbeispiel zur rekursiven Auswertung

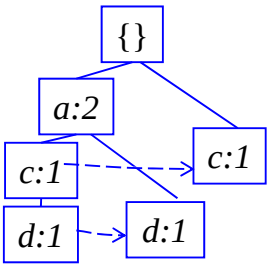
Teilbaum bezüglich Objekt e, min-support=2



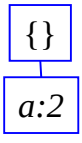


Pfade mit Suffix *de*

***e**-conditional FP-Tree*

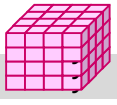


Nutzung dieses Baums zur Bestimmung der Frequent Itemsets für Suffixe *de*, *ce* und *ae*



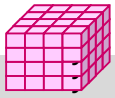
***de**-conditional FP-Tree*

-> Frequent Itemsets: ***de***, ***ade***



Assoziationsregeln: weitere Aspekte

- Nutzbarkeit u.a. für Cross-Selling, Produkt-Platzierung ...
  - Amazon: Kunden die dieses Buch gekauft haben, kauften auch ...
- Sonderfall: Sequenzanalyse (Erkennung sequentieller Muster)
  - Berücksichtigung der Zeit-Dimension
  - Bsp. 1: In 50% der Fälle, wenn Produkt A gekauft wurde, wird bei einem späteren Besuch Produkt B gekauft
  - Bsp. 2: In 40% aller Fälle, in denen ein Nutzer über einen Werbebanner auf die Web-Site gelangt und die Site vorher schon einmal besucht hat, kauft er einen Artikel; dies kommt in insgesamt 10% aller Sessions vor
- Probleme
  - sehr viele Produkte / Web-Seiten / Werbebanner / Besucher etc. erfordern Bildung gröberer Bezugseinheiten
  - es können sinnlose Korrelationen ermittelt werden
    - fehlender kausaler Zusammenhang
    - z.B. Schmutzeffekte aufgrund transitiver Abhängigkeiten (Bsp.: Haarlänge korreliert negativ mit Körpergröße)



DBS-Unterstützung für Data Mining

- SQL-Standardisierung: SQL/MM
  - Standardisierte Schnittstelle (UDTs, UDFs) zur Definition von Data Mining-Modellen, Bereitstellung für Trainingsdaten und Data Mining-Anfragen
- Ähnliche Unterstützung in DB2 und MS SQL Server
- Beispiel: Klassifikation bei MS SQL-Server
  - Repräsentation eines Mining-Modells als geschachtelte Tabellen

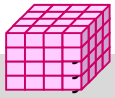
```
CREATE MINING MODEL AgePrediction
        ( CustomerID    LONG KEY,
          Gender TEXT   DISCRETE ATTRIBUTE,
          HairColor     TEXT    DISCRETE ATTRIBUTE,
          Age           DOUBLE CONTINUOUS ATTRIBUTE PREDICT )
USING [Microsoft Decision Tree]
```

  - Bereitstellung von Trainingsdaten (bekannte „Fälle“)

```
INSERT INTO AgePrediction (CustID, Gender, HairColor, Age)
                    ‘SELECT CustID, Gender, HairColor, Age  FROM Customers ’ )
```

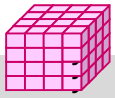
  - Anwendung des Modells zur Analyse über Prediction Joins

```
SELECT Customers.ID, MyDMM.Age, PredictProbability(MyDMM.Age)
FROM AgePrediction MyDMM PREDICTION JOIN Customers ON
  MyDMM.Gender = Customers.Gender AND ...
```



Zusammenfassung (1)

- Data Mining Verfahrensklassen: Clusteranalyse, Klassifikation, Assoziationsregeln
- zahlreiche Nutzungsmöglichkeiten: Kundensegmentierung, Vorhersage des Kundenverhaltens, Warenkorbanalyse etc.
  - keine „out of the box“-Lösungen
  - Interpretation der Ergebnisse nicht immer einfach
- Clusteranalyse: Segmentierung über Distanzmaß
  - K-Means: vorgegebene Anzahl von Clustern
  - Canopy Clustering: schnelle Berechnung überlappender Cluster; nützlich als Vorbereitung für K-Means
- Klassifikation z.B. über Entscheidungsbäume
  - erfordert Trainingsdaten mit bekannter Klassenzuordnung
  - Bestimmung der Split-Attribute eines Entscheidungsbaums gemäß Informationsgewinn



Zusammenfassung (2)

- Assoziationsregeln, u.a. zur Warenkorbanalyse
  - Berechnung der Frequent Itemsets mit minimalem Support
  - Ableitung der Regeln mit ausreichender Konfidenz
- A-Priori-Algorithmus
  - Anti-Monotonie: Wenn Menge häufig ist, so auch all ihre Teilmengen
  - Bottom-Up-Auswertung im Verband häufiger Mengen: Hinzufügen häufiger Teilmengen zur Kandidatengenerierung
  - Pruning durch Fallenlassen von Mengen mit seltenen Teilmengen
- FP-Tree zur schnelle Berechnung häufiger Mengen
  - kompakte Repräsentation aller Transaktionen
  - rekursives Pattern Mining auf Teilbäumen (Divide-and-Conquer)