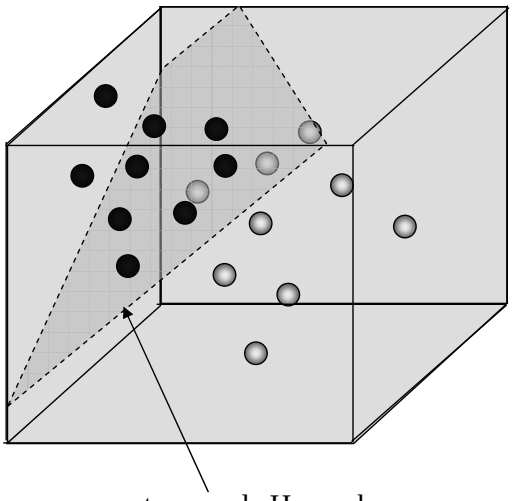
3.5 Support Vector Machines

Motivation: Lineare Separation



- Vektoren in $\mathfrak{R}^d$ repräsentieren Objekte.
- Objekte gehören zu genau einer von je 2 Klassen.
- Weil sich hierdurch die Mathematik vereinfacht, werden als Klassen-Label $y = -1$ und $y = +1$ verwendet

Klassifikation durch lineare Separation:

- Suche Hyperebene, die beide Vektor-Mengen „maximal stabil“ voneinander trennt.
- Ordne unbekannte Elemente der Seite der Ebene zu, auf der sie sich befinden.

Support Vector Machines

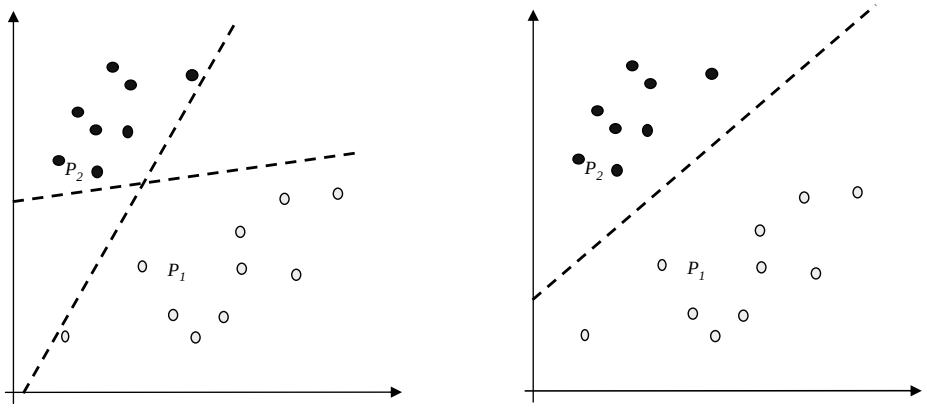
Probleme bei linearer Separation :

- Was ist die „maximal stabile“ Hyperebene und wie berechnet man sie effizient?
- Klassen nicht immer linear trennbar.
- Berechnung von Hyperebenen nach Auswahl sehr aufwendig.
- Einschränkung auf 2 Klassen.
- ...

⇨ *Lösungen dieser Probleme mit Support Vector Machines (SVMs) [Vapnik 1979 u. 1995].*

Maximum Margin Hyperplane

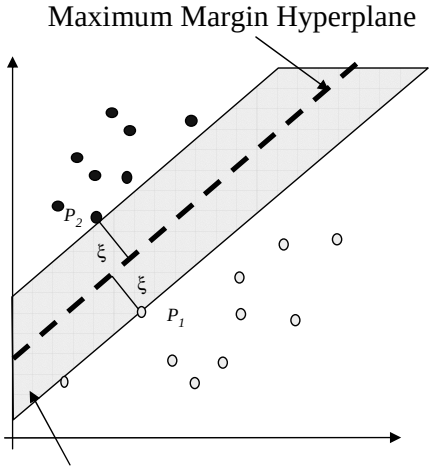
Problem: Hyperebene, die P_1 und P_2 trennt, ist nicht eindeutig.
$\Rightarrow$ *Welche Hyperebene ist für die Separation die Beste ?*



Kriterien:
- Stabilität beim Einfügen
- Abstand zu den Objekten beider Klassen

Prinzip von Support Vector Machines

Lineare Separation mit der Maximum Margin Hyperplane



- Abstand zu Punkten aus beiden Mengen ist maximal, d.h. mind. ξ.
- Wahrscheinlichkeit, dass beim Einfügen die trennende Hyperebene verschoben werden muss, ist minimal.
- generalisiert am besten.

$\Rightarrow$ Maximum Margin Hyperplane (MMH) ist „maximal stabil“

MMH ist nur von Punkten P_i abhängig, die Abstand ξ zur Ebene aufweisen.

$\Rightarrow$ *P_i heißt Support Vector*

Maximum Margin Hyperplane

Zusammenfassung der Schreibweisen der benötigten algebraischen Konstrukte für Featurespace *FS*:

- Skalarprodukt zweier Vektoren: $\langle \vec{x}, \vec{y} \rangle, \vec{x}, \vec{y} \in FS$

  z.B. $\langle \vec{x}, \vec{y} \rangle = \sum_{i=1}^{d} (x_i \cdot y_i)$ kanonisches Skalarprodukt

- Beschreibung einer Hyperebene: $H(\vec{w}, b) = \left\{ \vec{x} \in FS \,\middle|\, 0 = \langle \vec{w}, \vec{x} \rangle + b \right\}$

- Abstand eines Vectors zur Ebene: $dist(\vec{x}, H(\vec{w}, b)) = \left| \frac{1}{\sqrt{\langle \vec{w}, \vec{w} \rangle}} \langle \vec{w}, \vec{x} \rangle + b \right|$

Maximum Margin Hyperplane

Berechnung der Maximum Margin Hyperplane

1. Bedingung: kein Klassifikationsfehler (Klasse1: y_i=1,Klasse 2:y_i=-1)

$$\left.\begin{array}{l} (y_i = -1) \Rightarrow [\langle \vec{w}, \vec{x_i} \rangle + b] < 0 \\ (y_i = 1) \Rightarrow [\langle \vec{w}, \vec{x_i} \rangle + b] > 0 \end{array}\right\} \Leftrightarrow y_i \left[\langle \vec{w}, \vec{x_i} \rangle + b \right] > 0$$

2. Bedingung: Maximaler Rand (Margin)

maximiere: $\xi = \min_{x_i \in TR} \left| \frac{1}{\sqrt{\langle \vec{w}, \vec{w} \rangle}} \left(\langle \vec{w}, \vec{x_i} \rangle + b \right) \right|$ (Abstand von x_i zur Ebene $H(\vec{w},b)$)

zusammen:

maximiere: ξ , so dass $\left[y_i \left(\frac{1}{\sqrt{\langle \vec{w}, \vec{w} \rangle}} \left[\langle \vec{w}, \vec{x_i} \rangle + b \right] \right) \geq \xi \right]$ für $\forall i \in [1..n]$

Maximum Margin Hyperplane

maximiere ξ in $\left[y_i \left(\frac{1}{\sqrt{\langle \vec{w}, \vec{w} \rangle}} \left[\langle \vec{w}, \vec{x_i} \rangle + b \right] \right) \geq \xi \right]$, für $\forall i \in [1..n]$

Setze $\frac{1}{\sqrt{\langle \vec{w}, \vec{w} \rangle}} = \xi$: max. $\frac{1}{\sqrt{\langle \vec{w}, \vec{w} \rangle}}$, mit $\left(y_i \cdot \left(\xi \cdot \left(\langle \vec{w}, \vec{x_i} \rangle + b \right) \right) \geq \xi \right)$ $\forall i \in [1..n]$

$\Rightarrow$ max. $\frac{1}{\sqrt{\langle \vec{w}, \vec{w} \rangle}}$, mit $\left(y_i \left(\langle \vec{w}, \vec{x_i} \rangle + b \right) \geq 1 \right)$ $\forall i \in [1..n]$

Statt $\frac{1}{\sqrt{\langle \vec{w}, \vec{w} \rangle}}$ invertiere, quadriere und minimiere das Ergebnis:

Primäres Optimierungsproblem: minimiere $J(\vec{w}, b) = \langle \vec{w}, \vec{w} \rangle$

unter Nebenbedingung für $\forall i \in [1..n]$ sei $\left(y_i \left(\langle \vec{w}, \vec{x_i} \rangle + b \right) \geq 1 \right)$

Duales Optimierungsproblem

Zur Berechnung wird das primäre Optimierungsproblem in ein duales OP überführt (Umformulierung in Form mit Langrange Multipliers).

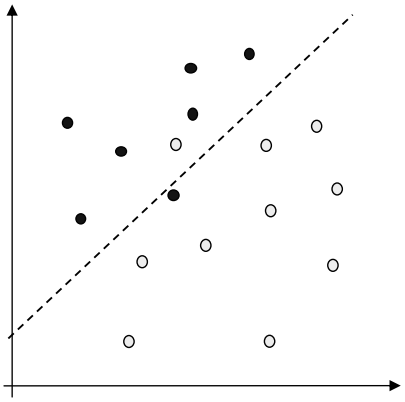
Duales OP: maximiere $L(\vec{\alpha}) = \left(\sum_{i=1}^{n} \alpha_i \right) - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \cdot \alpha_j \cdot y_i \cdot y_j \cdot \langle \vec{x}_i \cdot \vec{x}_j \rangle$

mit Bedingung $\sum_{i=1}^{n} \alpha_i \cdot y_i = 0$ und $0 \leq \alpha_i$

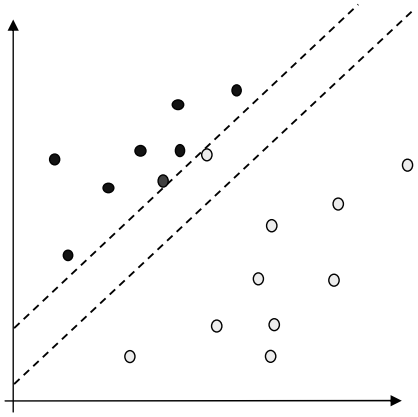
$\Rightarrow$ Lösung des Problems mit Algorithmen aus der Optimierungstheorie
$\Rightarrow$ bis jetzt nur linear separierbarer Fall: Soft Margin Optimierung
$\Rightarrow$ Einführung von Kerneln zur Steigerung der Kapazität

Soft Margin Optimierung

Behandlung nicht linear trennbarer Daten:



Daten nicht separierbar

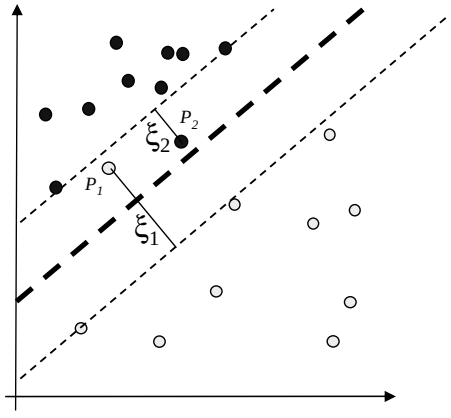


vollständige Separation ist nicht optimal

⇒ Trade-Off zwischen Trainingsfehler und Breite der Grenze

Soft Margin Optimierung

Betrachte beim Optimieren zusätzlich noch die Anzahl der Trainingsfehler.



- *ξ_i ist der Abstand von P_i zum Rand (wird auch Slack-Variable genannt)*
- *C reguliert den Einfluss eines einzelnen Trainingsvektors*

minimiere $J(\vec{w},b) = \frac{1}{2}\vec{w}\cdot\vec{w} + C\cdot\sum_{i=1}^{n}\xi_i$

mit Nebenbedingung für $\forall i \in [1..n]$ sei $\left[y_i\left(\vec{w}\cdot\vec{x_i}+b\right)\geq 1-\xi_i\right]$ und $\xi_i \geq 0$

⇒Primäres Optimierungsproblem unter weichen Grenzen (Soft Margin)

Soft Margin Optimierung

Das duale OP mit Langrange Multiplikatoren verändert sich wie folgt:

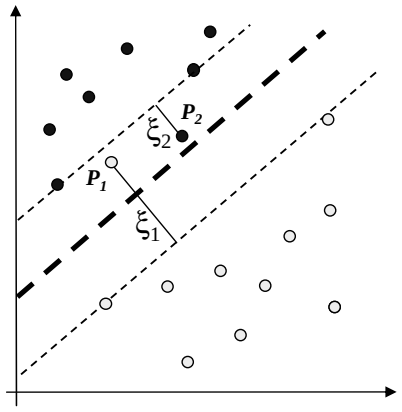
Duales OP: maximiere $L(\vec{\alpha}) = \left(\sum_{i=1}^{n} \alpha_i \right) - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \cdot \alpha_j \cdot y_i \cdot y_j \cdot \left\langle \vec{x}_i \cdot \vec{x}_j \right\rangle$

mit Bedingung $\sum_{i=1}^{n} \alpha_i \cdot y_i = 0$ und $0 \le \alpha_i \le C$

- $0 < \alpha_i < C \Leftrightarrow$ Support Vector mit $\xi_i = 0$
- $\alpha_i = C \Leftrightarrow$ Support Vector mit $\xi_i > 0$
- $\alpha_i = 0$ *else*

Entscheidungsregel:

$$h(\vec{x}) = sign\left(\sum_{x_i \in SV} \alpha_i \cdot y_i \left\langle \vec{x}_i, \vec{x} \right\rangle + b \right)$$

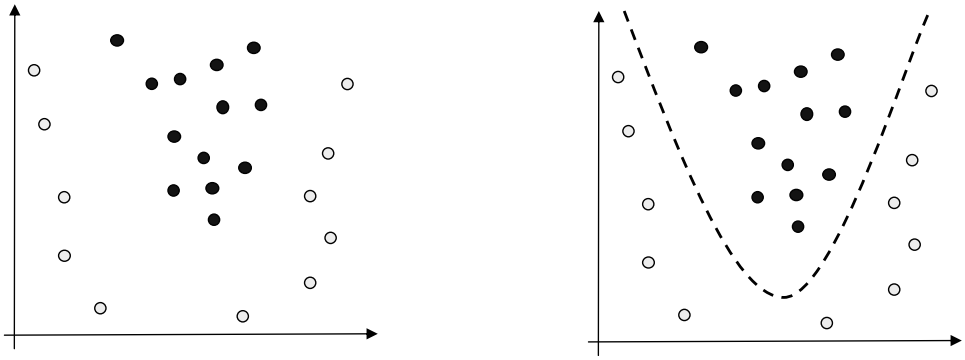


Kernel Machines

Lernen bei nicht linear trennbaren Datenmengen

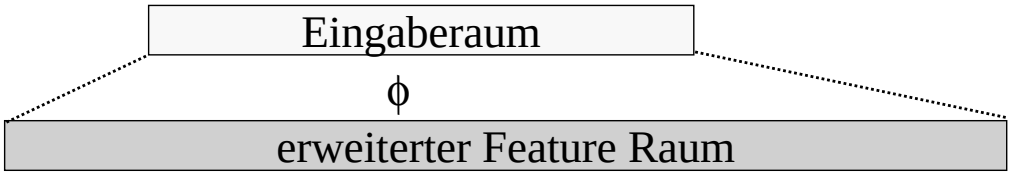
Problem : Bei realen Problemen ist häufig keine lineare Separation mit hoher Klassifikationsgenauigkeit mehr möglich.

Idee: Transformiere Daten in einen nicht linearen Raum und versuche sie im neuem Raum linear zu separieren. (Erweiterung der Hypothesenraumes)



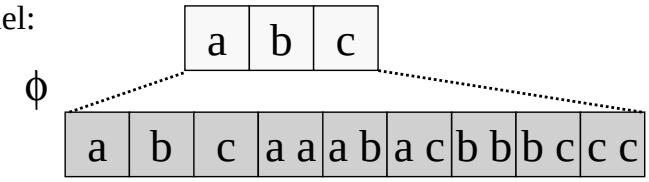
Beispiel: quadratische Transformation

Erweiterung des Hypothesenraumes



⇒Versuche jetzt in erweitertem Feature Raum linear zu separieren

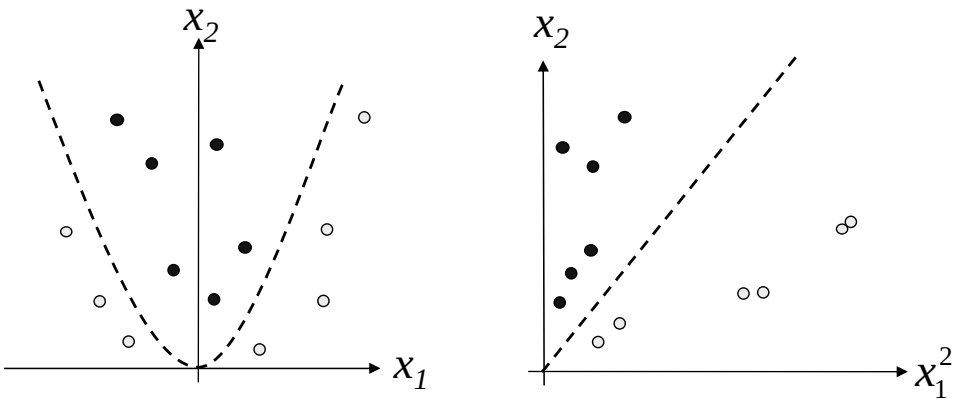
Beispiel:



hier: Eine Hyperebene im erweiterten Feature Raum ist ein Polynom 2. Grades im Eingaberaum.

Eingaberaum: $\vec{x} = (x_1, x_2)$ (2 Attribute)

im erweiterten Raum: $\phi(\vec{x}) = (x_1^2, x_2^2, \sqrt{2} \cdot x_1, \sqrt{2} \cdot x_2, \sqrt{2} \cdot x_1 \cdot x_2, 1)$
(6 Attribute)



Kernel Machines

Einführung eines Kernels ⇔ Featuretransformation

$$\phi\left(\vec{x}\right): FS_{alt} \longrightarrow FS_{neu}$$

Duales OP: maximiere

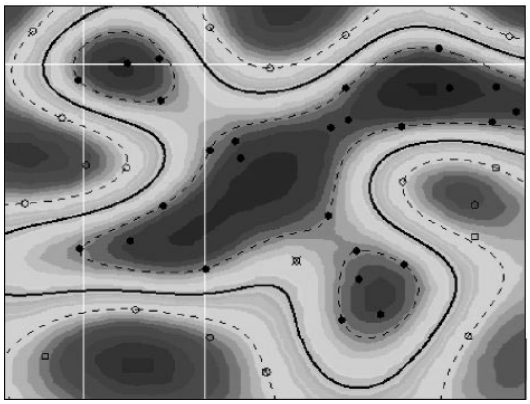
$$L(\vec{\alpha}) = \left(\sum_{i=1}^{n} \alpha_i\right) - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} \alpha_i \cdot \alpha_j \cdot y_i \cdot y_j \cdot \left\langle \phi(\vec{x}_i), \phi(\vec{x}_j) \right\rangle$$

mit Bedingung $\sum_{i=1}^{n} \alpha_i \cdot y_i = 0$ und $0 \le \alpha_i \le C$

Zusätzliche Featuretransformation wirkt sich nur auf das Skalarprodukt der Trainingsvektoren aus. ⇒ Kernel *K* ist eine Funktion mit:

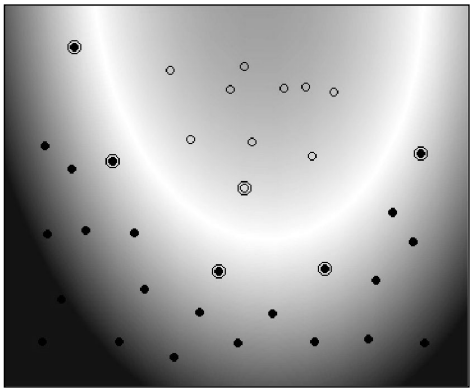
$$K_\phi\left(\vec{x}_i, \vec{x}_j\right) = \left\langle \phi(\vec{x}_i), \phi(\vec{x}_j) \right\rangle$$

Kernel Machines



Radial Basis Kernel:

$$K(\vec{x}, \vec{y}) = \exp\left(-\gamma \cdot \left|\vec{x} - \vec{y}\right|^2\right)$$



Polynomieller Kernel (Grad 2)

$$K(\vec{x}, \vec{y}) = \left(\left\langle \vec{x}, \vec{y} \right\rangle + 1\right)^d$$

Diskussion

+ erzeugt Klassifikatoren mit hoher Genauigkeit
+ verhältnismäßig schwache Tendenz zu Overfitting
  (Begründung durch Generalisierungtheorie)
+ effiziente Klassifikation neuer Objekte
+ kompakte Modelle

- unter Umständen lange Trainingszeiten
- aufwendige Implementierung
- gefundene Modelle schwer zu deuten

4 Assoziationsregeln

Inhalt dieses Kapitels

Motivation

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

Transaktionsdatenbank

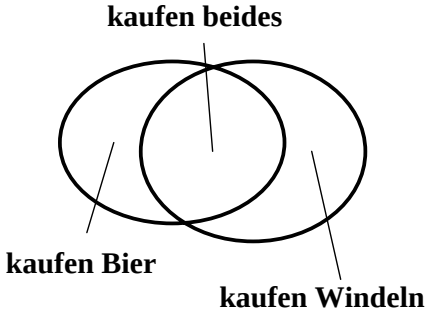
Warenkorbanalyse

- 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

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

Grundbegriffe [Agrawal & Srikant 1994]

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

Grundbegriffe

- *Support der Menge X in D*: Anteil der Transaktionen in *D*, die *X* enthalten
- *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 ≥ *minsup* und eine Konfidenz ≥ *minconf* besitzen

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%)

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

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)

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;
```

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

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

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

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

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-1} gehört

Beispiel

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

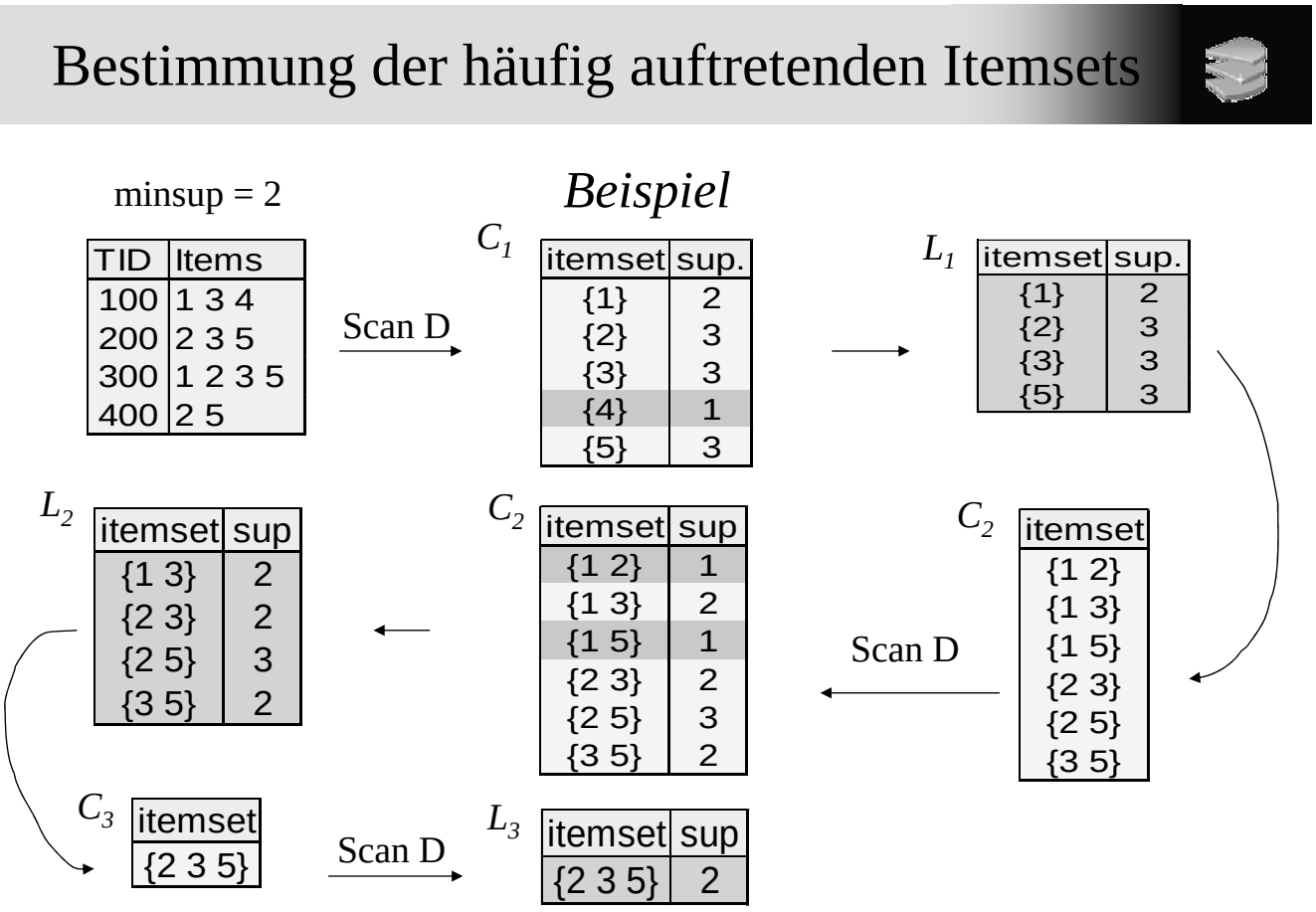
nach dem Join-Schritt: Kandidaten = {(1 2 3 4), (1 3 4 5)}

im Pruning-Schritt:

lösche (1 3 4 5)

⇒ C_4 = {(1 2 3 4)}

Bestimmung der häufig auftretenden Itemsets



Bestimmung der häufig auftretenden Itemsets

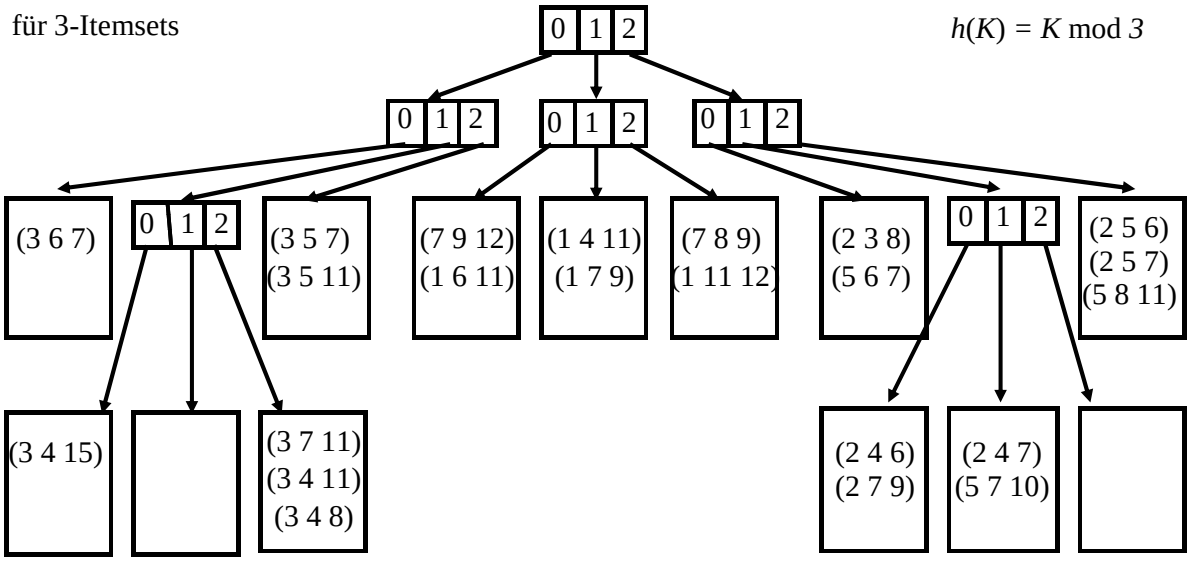
Effiziente Unterstützung der Subset-Funktion

- `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

Bestimmung der häufig auftretenden Itemsets

Beispiel



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

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)

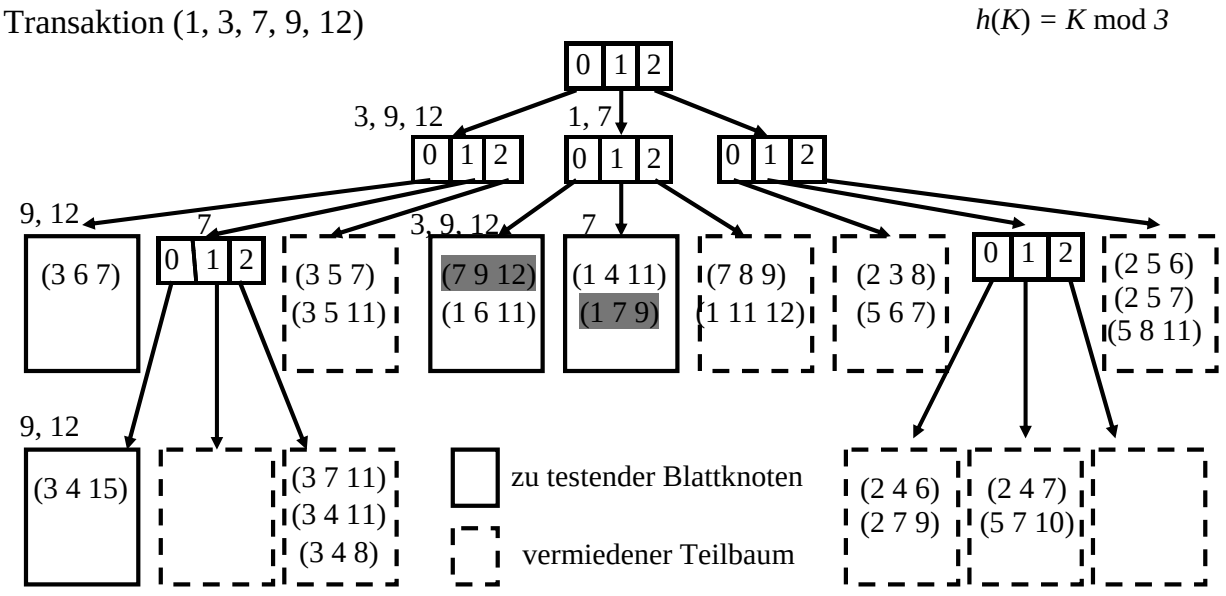
  bestimme die Hashwerte und suche weiter für jedes Item t_k mit $k > i$

- bei einem Blattknoten

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

Bestimmung der häufig auftretenden Itemsets

Beispiel



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

Bestimmung der häufig auftretenden Itemsets

Methoden der Effizienzverbesserung

Partitionierung der Datenbank [Savasere, Omiecinski & Navathe 1995]

- ein Itemset ist nur dann häufig, wenn es 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

Bestimmung der Assoziationsregeln

Methode

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

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

- Speicherung der Frequent Itemsets mit ihrem Support in einer Hashtabelle

⇒ keine Datenbankzugriffe

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*

Interessantheit von Assoziationsregeln

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*.