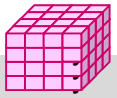
# 6. Überblick Data Mining/ML-Verfahren

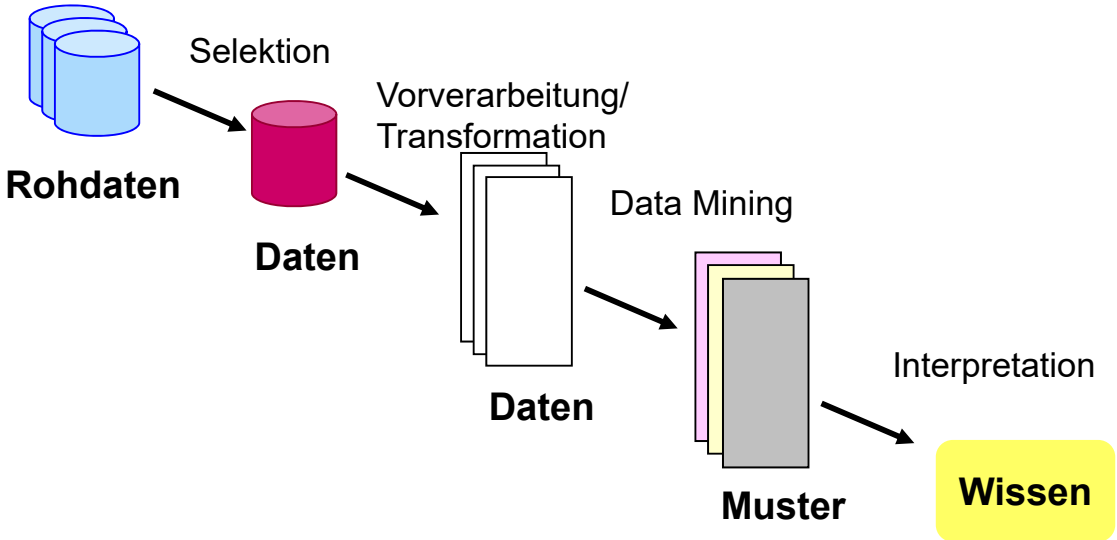
- Einführung Data Mining / maschinelles Lernen
  - KDD-Prozess
  - Anwendungsbeispiele
- Assoziationsregeln / Warenkorbanalyse
  - Support und Konfidenz
  - A Priori-Algorithmus
  - Frequent Pattern (FP)-Trees
- Clusteranalyse
  - k-Means-Algorithmus
  - Canopy Clustering
- Klassifikation
  - Klassifikationsprozess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

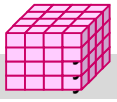




# Knowledge Discovery in Databases (KDD)

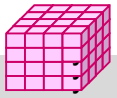
- (semi-)automatische Extraktion von Wissen aus Datenbanken, das
  - gültig (im statistischen Sinn)
  - bisher unbekannt
  - und potentiell nützlich ist
- Kombination von Verfahren zu Datenbanken, Statistik und KI (maschinelles Lernen)



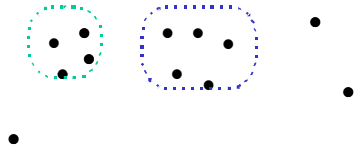
# Data Mining / Machine Learning

- Data Mining: Anwendung effizienter Algorithmen zur Erkennung von Mustern in großen Datenmengen
- Machine Learning: lernbasierte (KI-) Ansätze für Vorhersagen / Wissensgenerierung
- Data Mining/ML für Big Data / Datenbanken bzw. Data Warehouses
  - Skalierbarkeit auf große Datenmengen (nicht nur im Hauptspeicher)
  - parallele Realisierungen
- umfassende Datenaufbereitung und Vorverarbeitung erforderlich
  - Diskretisierung numerischer Attribute (Aufteilung von Wertebereichen in Intervalle, z.B. Altersgruppen)
  - Erzeugen abgeleiteter Attribute (z.B. Aggregationen für bestimmte Dimensionen, Umsatzänderungen)
  - Extraktion von analyserelevanten Merkmalen / Features aus unstrukturierten Daten

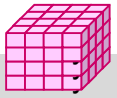




# Analysetechniken

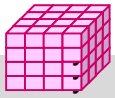


- Clusteranalyse
  - Objekte werden aufgrund von Ähnlichkeiten in Klassen eingeteilt (Segmentierung)
- Assoziationsregeln
  - Warenkorbanalyse (z.B. Kunde kauft A und B => Kunde kauft C)
  - Sonderformen zur Berücksichtigung von Dimensionshierarchien (z.B. Produktgruppen), quantitativen Attributen, zeitlichen Beziehungen (sequence mining)
- Klassifikation
  - Zuordnung von Objekten zu Gruppen/Klassen mit gemeinsamen Eigenschaften bzw. Vorhersage von Attributwerten
  - Verwendung von Stichproben (Trainingsdaten)
  - Ansätze: Entscheidungsbaum-Verfahren, neuronale Netze, statistische Auswertungen (z.B. Maximum Likelihood-Schätzung / Bayes-Schätzer)
- weitere Ansätze:
  - genetische Algorithmen (multivariate Optimierungsprobleme, z.B. Identifikation der besten Bankkunden)
  - Regressionsanalyse zur Vorhersage numerischer Attribute . . .



# Data Mining: Anwendungsbeispiele

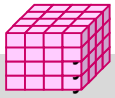
- Kundensegmentierung für Marketing
  - Gruppierung von Kunden mit ähnlichem Kaufverhalten / ähnlichen Interessen
  - Nutzung für gruppenspezifische Empfehlungen, Product Bundling, ...
- Warenkorbanalyse: Produkt-Platzierung im Supermarkt, Preisoptimierung, ...
- Bestimmung der Kreditwürdigkeit von Kunden
  - 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, ...





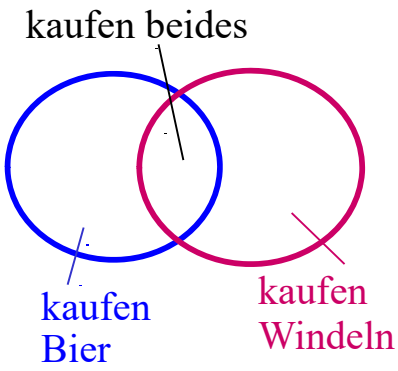
# Evaluation/Interpretation

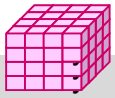
- Ablauf
  - Präsentation der gefundenen Muster, z.B. über Visualisierungen
  - Bewertung der Muster durch den Benutzer
  - falls schlechte Bewertung: erneute Analyse mit anderen Parametern, anderem Verfahren oder anderen Daten
  - falls gute Bewertung: Integration des gefundenen Wissens in die Wissensbasis / 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



# 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



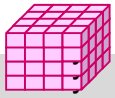





# Assoziationsregeln (2)

- 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 $X \rightarrow Y$, wobei $X \subseteq R$, $Y \subseteq R$, $X \cap Y = \{\}$

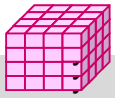
$$supp\,(r,c) = \frac{\left|\left\{t \in r \middle| X \cup Y \in t\right\}\right|}{|r|} \geq s_{\min} \qquad conf\,(r,c) = \frac{\left|\left\{t \in r \middle| X \cup Y \in t\right\}\right|}{\left|\left\{t \in r \middle| X \in r\right\}\right|} \geq conf_{\min}$$



# Beispiel: Warenkorbanalyse

| EinkaufsID | Aftershave | Bier | Chips |
|---|---|---|---|
| 1 | 0 | 1 | 1 |
| 2 | 1 | 1 | 0 |
| 3 | 0 | 1 | 1 |
| 4 | 1 | 0 | 1 |
| 5 | 1 | 1 | 1 |

{Aftershave} → {Bier} s = 2/5, conf = 2/3
{Bier} → {Chips}
{Aftershave} → {Chips}
{Chips} → {Aftershave}
{Aftershave} → {Bier,Chips}





# Weitere Beispiele

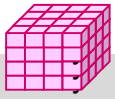
- Assoziationsregeln:
  - A → C
  - C → A

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

*minsup = 50%,*
*minconf = 50%*

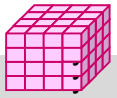
- Warenkörbe:
  - Drucker, Papier, PC, Toner
  - PC, Scanner
  - Drucker, Papier, Toner
  - Drucker, PC
  - Drucker, Papier, PC, Scanner, Toner

Assoziationsregel: PC → Drucker



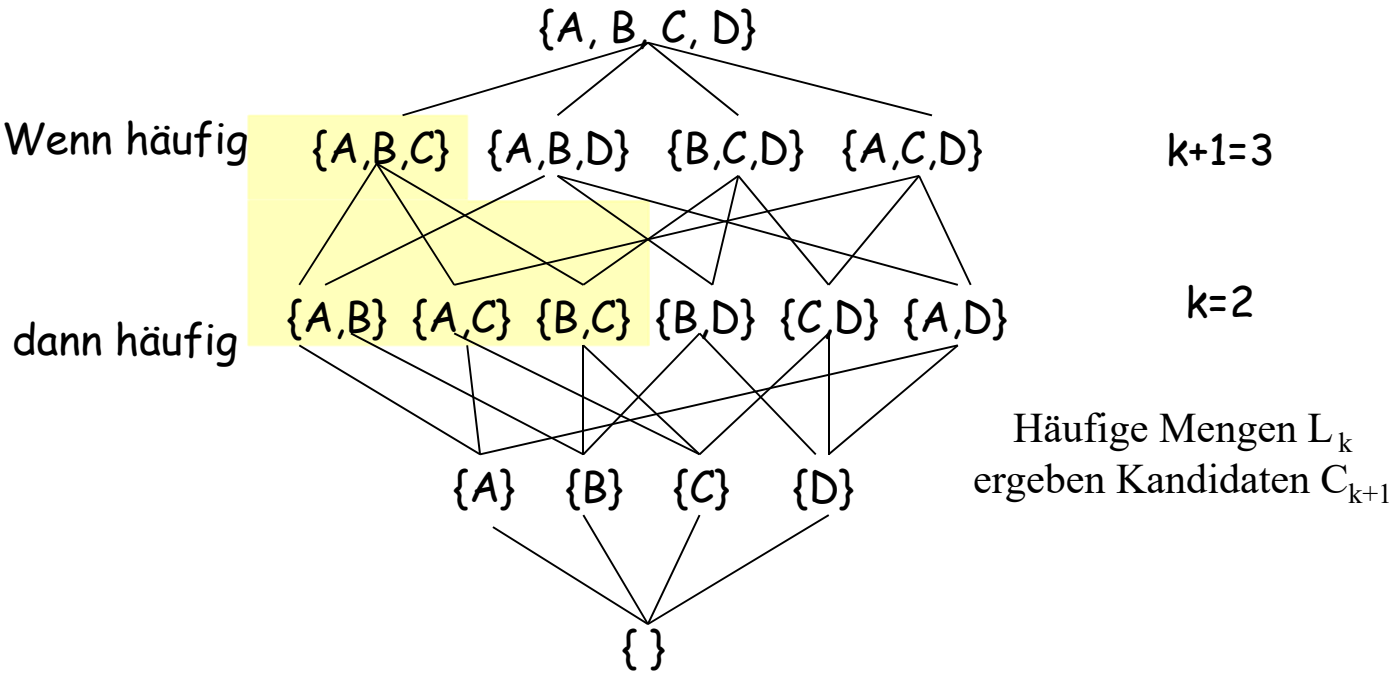
# Frequent Itemsets

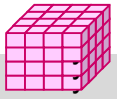
- Frequent Item-Set (häufige Mengen):
  - Menge von Items/Objekten, deren Support Schranke $S_{min}$ übersteigt
- Bestimmung der Frequent-Itemsets wesentlicher Schritt zur Bestimmung von Assoziationsregeln
- effiziente Realisierung über A-Priori-Algorithmus
- Nutzung der sog. A-Priori-Eigenschaft:
  - 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



# 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

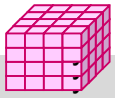
while $L_k \neq \{\}$

  $C_{k+1}$ := 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

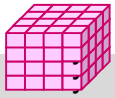
  if (alle k-elementigen Teilmengen von $l$ in $L_k$ )

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

Return $C_{k+1}$

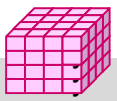
Beispiel:

| *k=3* | *k=4* |
|---|---|
| ABC | |
| ABD | |
| ACD | |
| BCD | |
| CDE | |



# 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

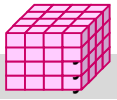
- Itemset $l = \{i_1, ..., i_{k-1}, i_k\}$ erlaubt viele Regeln der Form X → Y, mit $X \cup Y = I$, $X \cap Y = \{\}$
- Konfidenz: conf (X → Y)= supp ($l$) / supp (X)
- Generierung der Assoziationsregeln aus Itemset I
  - wenn die Konklusion (rechte Seite) länger wird, kann die Konfidenz sinken.
  - die Ordnung der Attribute kann ausgenutzt werden

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

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

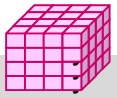
    $c_k = \{i_1\} \rightarrow \{..., i_{k-1}, i_k\}$ $conf_k$

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

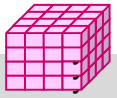
- Input: Transaktionsdatenbank D, Schwellwert min-support / $s_{min}$
- Scan von D, Erstellen der Menge F häufiger Objekte und ihrer Häufigkeiten, Ordnen von F in absteigender Häufigkeit.
- Wurzel des FP Trees ist leere Menge
- pro Transaktion in D: ordne Objekte gemäß F (absteigende Häufigkeit); füge 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 | Drucker, Papier, PC, Toner |
| 20 | PC, Scanner |
| 30 | Drucker, Papier, Toner |
| 40 | Drucker, PC |
| 50 | Drucker, Papier, PC,Scanner, Toner |

| *Objekt* | *count* |
|---|---|
| *Drucker* | *4* |
| *PC* | *4* |
| *Papier* | *3* |
| *Toner* | *3* |

Objekt-Tabelle F

*min_support = 0.5*



# FP-Tree: Generierung (2)

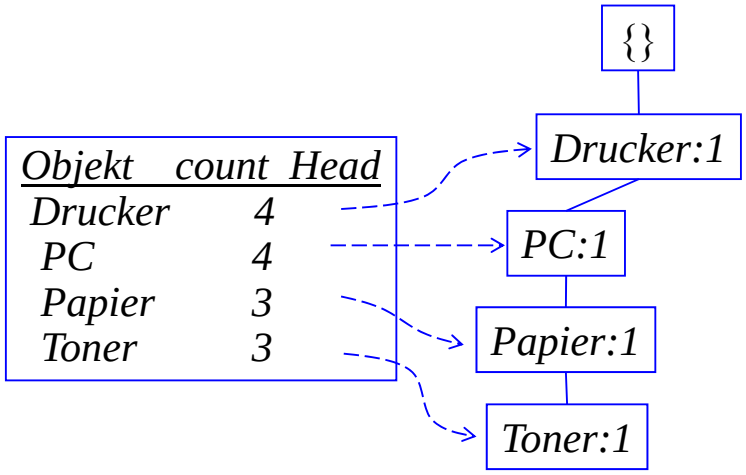
| TID | Objekte |
|---|---|
| 10 | Drucker, Papier, PC, Toner |
| 20 | PC, Scanner |
| 30 | Drucker, Papier, Toner |
| 40 | Drucker, PC |
| 50 | Drucker, Papier, PC,Scanner, Toner |

*min_support = 0.5*

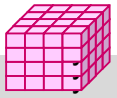
*gemäß F-Ordnung*
Drucker, PC, Papier, Toner

Objekt-Tabelle F

| Objekt | count | Head |
|---|---|---|
| Drucker | 4 | |
| PC | 4 | |
| Papier | 3 | |
| Toner | 3 | |



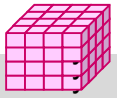

pro Transaktion gibt es einen Pfad, wobei gemeinsame Präfixe nur einmal repräsentiert werden (Komprimierungseffekt)





# FP-Tree: Erkennung häufiger Mengen

- Erkennung von Frequent Itemsets durch rekursive Suche auf Teilbäumen (Divide and Conquer)
- Erzeuge Musterbasis (*conditional pattern base*) 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 dieses Suffix. Die Präfixpfade eines Suffixes bilden seine Musterbasis
- Erzeuge reduzierten FP-Baum *(conditional FP tree)* für jede Musterbasis
  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, verbleiben im reduzierten FP-Tree
- rekursives Ableiten von Frequent Itemsets aus reduzierten FP-Trees
  - bei einzigem Pfad im Baum: Aufzählen aller Knoten-Kombinationen



# Algorithmus *FP-Growth (tree, I)*

Mining frequent patterns durch sukzessives Wachstum von Pattern-Fragmenten

initialer Aufruf mit *FP-Growth (FP-Baum, null)*

**Input:** Frequent Pattern Tree tree, Frequent Itemset *I*

**Output:** vollständige Menge häufiger Itemsets

**if** *tree hat nur einen Pfad P* **then**

**foreach** *Kombination K von Knoten in P* **do**

**return** $K \cup I$ mit support = minimaler Support der Items in K **end**

**else**

**foreach** *Item i in Tabelle F (in umgekehrter Häufigkeitsreihenfolge)* **do**

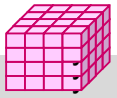
$K = i \cup I$ mit support = Support von i;

Erstelle Musterbasis von K und reduzierten FP-Baum $FP_K$ ;

**if** $FP_K$ *nicht leer* **then** FP-Growth ($FP_K$ ,K)

**end**

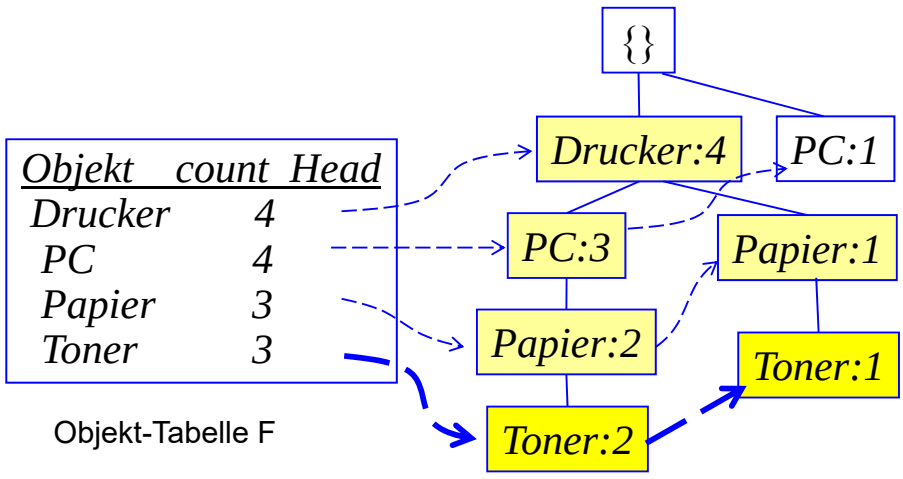
**end**





# Schritt 1: Musterbasis-Bestimmung

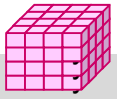
- gehe Objekt-Tabelle von unten (selten) nach oben durch. Die Verweise führen zu den Pfaden, in denen das Objekt vorkommt.
- das Objekt wird als Suffix betrachtet und alle Präfixe davon als Bedingungen für dieses Suffix. Die transformierten Präfixpfade eines Suffixes bilden seine Musterbasis




| Objekt | count | Head |
|---|---|---|
| Drucker | 4 | |
| PC | 4 | |
| Papier | 3 | |
| Toner | 3 | |

Objekt-Tabelle F

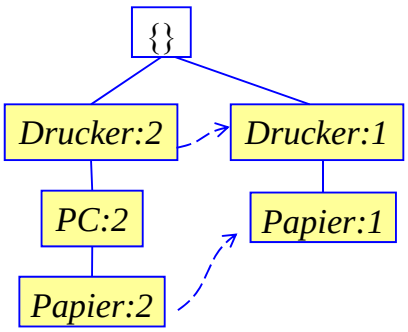
| Objekt | Musterbasis |
|---|---|
| *Toner* | *Drucker,PC,Papier:2, Drucker, Papier:1* |
| *Papier* | |
| *PC* | |
| *Drucker* | |



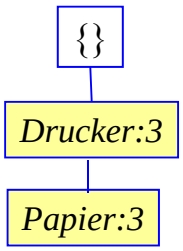
# Schritt 2: Erzeuge reduzierte FP-Trees

- propagiere Counts pro Objekt der Musterbasis von unten nach oben.
- nur Knoten/Präfix-Pfade, die min_support erfüllen, verebleiben im reduzierten FP-Tree

*Musterbasis für Toner*



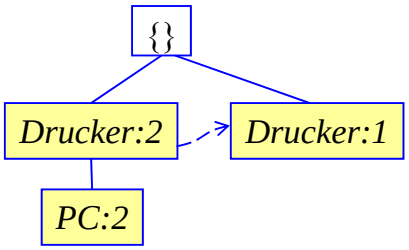

*reduzierter FP-Tree für Toner*



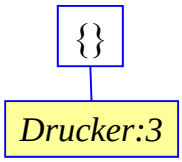

*Frequent Itemsets:*

*Toner*
*Papier, Toner*
*Drucker, Toner*
*Drucker, Papier, Toner*

*Musterbasis für Papier*

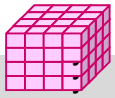



*reduzierter FP-Tree für Papier*




*Frequent Itemsets:*

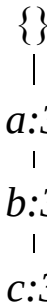
*Papier*
*Drucker, Papier*





# Reduzierte FP-Trees mit einem Pfad

- bei nur einem Pfad T kann rekursive Auswertung entfallen
- Bestimmung der Frequent Itemsets durch Aufzählung aller Kombinationen von Knotern / Teil-Pfaden

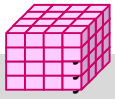



→

Frequent Itemsets bezüglich *d*

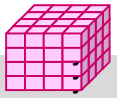
*d*,
*ad*, *bd*, *cd*,
*abd*, *acd*, *bcd*,
*abcd*

*d*-reduzierter FP-Tree



# Ergebnisse im Beispiel

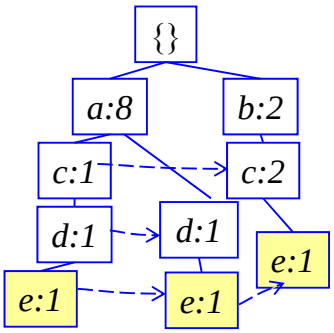
| Objekt | Musterbasis | reduzierter FP-Tree | Frequent Itemsets |
|---|---|---|---|
| Toner | {(Drucker,PC,Papier:2, Drucker, Papier:1} | { (Drucker:3, Papier:3)} \| Toner | Toner<br>Papier, Toner<br>Drucker, Toner<br>Drucker, Papier, Toner |
| Papier | {(Drucker,PC:2, Drucker:1} | { (Drucker:3)} \| Papier | Papier<br>Drucker, Papier |
| PC | {(Drucker:3) } | { (Drucker:3)} \| PC | PC<br>Drucker, PC |
| Drucker | {} | { } | Drucker |



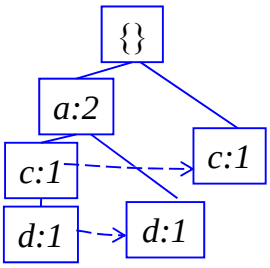


# Alternativbeispiel zur rekursiven Auswertung

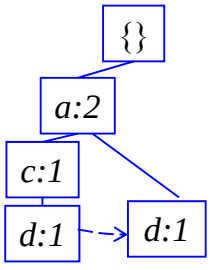
reduzierter FP-Baum bezüglich Objekt e, min-support=2



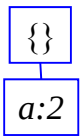

***e**-reduzierter FP-Tree*




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



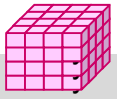

Pfade mit Suffix *de*




***de**-reduzierter FP-Tree*

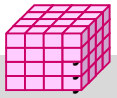
-> Frequent Itemsets: ***de***, ***ade***

Quelle: Han/Kamber



# Assoziationsregeln: weitere Aspekte

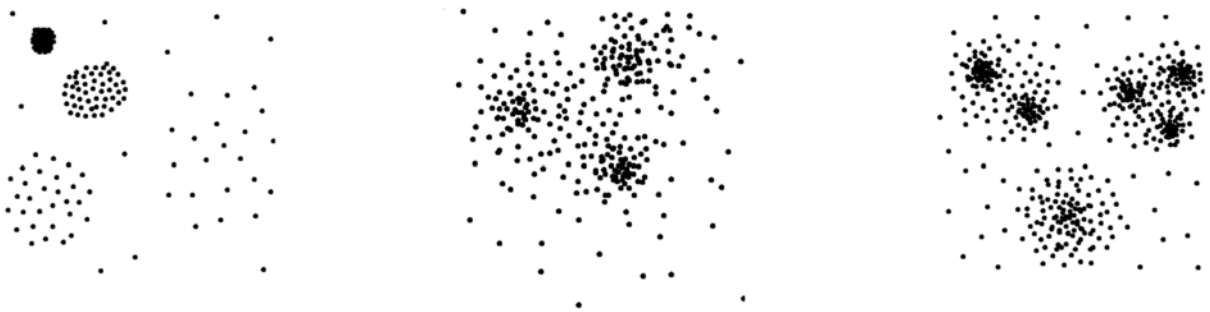
- Nutzbarkeit u.a. für Cross-Selling, Produkt-Platzierung ...
  - Amazon: Kunden die dieses Buch gekauft haben, kauften auch ...
- Sonderfall: Sequenzanalyse (Erkennung sequentieller Muster)
  - Berücksichtigung der Zeit-Dimension
  - Bsp. 1: In 50% der Fälle, wenn Produkt A gekauft wurde, wird bei einem 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)

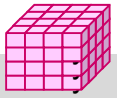




# Clusteranalyse

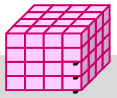
- Ziele
  - automatische Identifikation einer endlichen Menge von Kategorien, Klassen oder Gruppen (Cluster) in den Daten
  - Objekte im gleichen Cluster sollen möglichst ähnlich sein
  - Objekte aus verschiedenen Clustern sollen möglichst unähnlich zueinander sein
- Ähnlichkeitsbestimmung
  - meist: 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: hierarchisch, partitionierend, dichtebasiert, ...





# Clusteranalyse

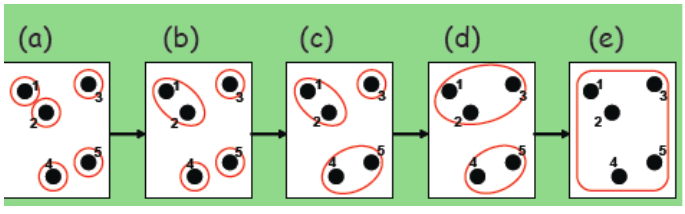
- oft nicht 2, sondern 10 oder 10.000 Dimensionen
  - fast alle Objektpaare sind ähnlich weit voneinander entfernt
  - Methoden zur Dimensionsreduzierung erforderlich
- Beispiel: ähnliche Musik-CDs
  - einige wenige Kategorien vs. käuferspezifische Präferenzen
  - jeder CD-Käufer bildet potenziell eigene Dimension
  - ähnliche CDs haben ähnliche Käufer und umgekehrt
- ähnliche Text-Dokumente / News-Meldungen
  - ähnliche Mengen von Keywords

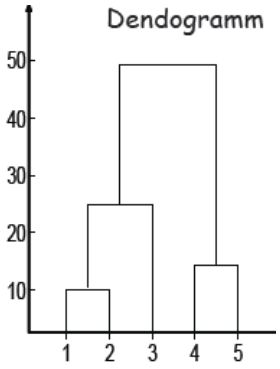




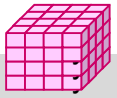
# Hierarchisches Clustering

- **Top-Down** (aufteilend)
  - Beginne mit einem Cluster
  - rekursives Splitten in Teil-Cluster
- **Agglomerativ** (Bottom-Up-Ansatz)
  - zunächst ist jeder Punkt eigenes Cluster
  - wiederholte Vereinigung der "ähnlichsten" Cluster
  - Kernproblem: Bestimmung der Merge-Kandidaten
  - naiv: kubische Komplexität



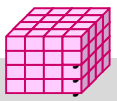




- Repräsentation durch Dendogramme
- Bestimmung von Cluster-Repräsentanten
  - z.B. Cendroid (bei Euklidischer Distanz)
  - „Clustroid": ex. Punkt, der am nächsten zu allen anderen im Cluster liegt



# K-Means Algorithmus

- Ausgangssituation
  - Objekte besitzen Distanzfunktion (meist Euklidische Distanz)
  - 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:

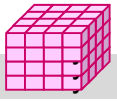
(3) Zentren:

(2) Cluster:

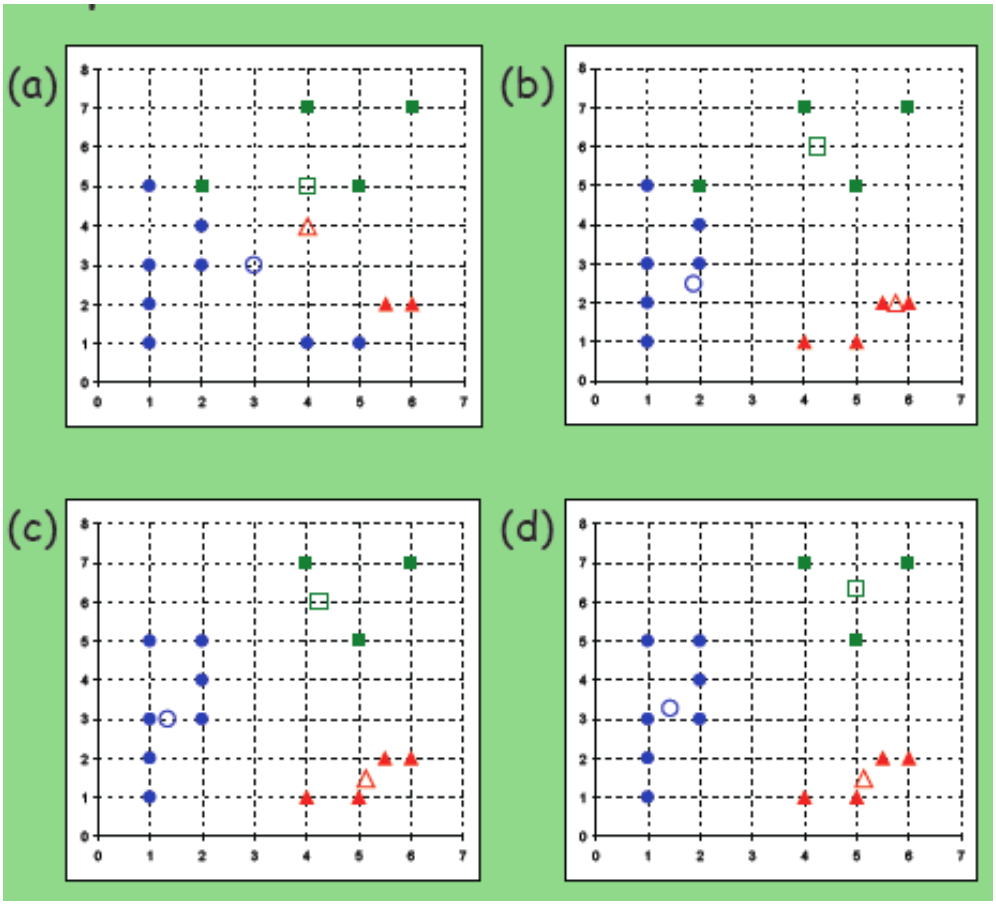
(3) Zentren:

(2) Cluster:

Abbruch, da sich das Clustering nicht mehr geändert hat.



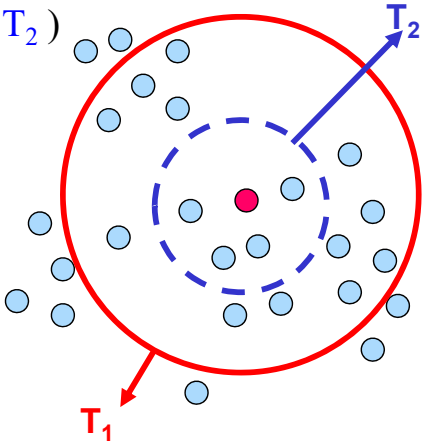
# K-Means: 2-D-Beispiel (k=3)






# Canopy Clustering*

- 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 Canopy-Zentren wird mit allen Objekten initialisiert
  - Schritt 2 (Canopy-Zentrum): Zentrum 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

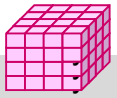
- Clustering der Zahlen 1, 3, 6, 11, 13

  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

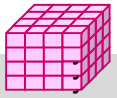
  (5) Abbruch, da Kandidatenliste leer





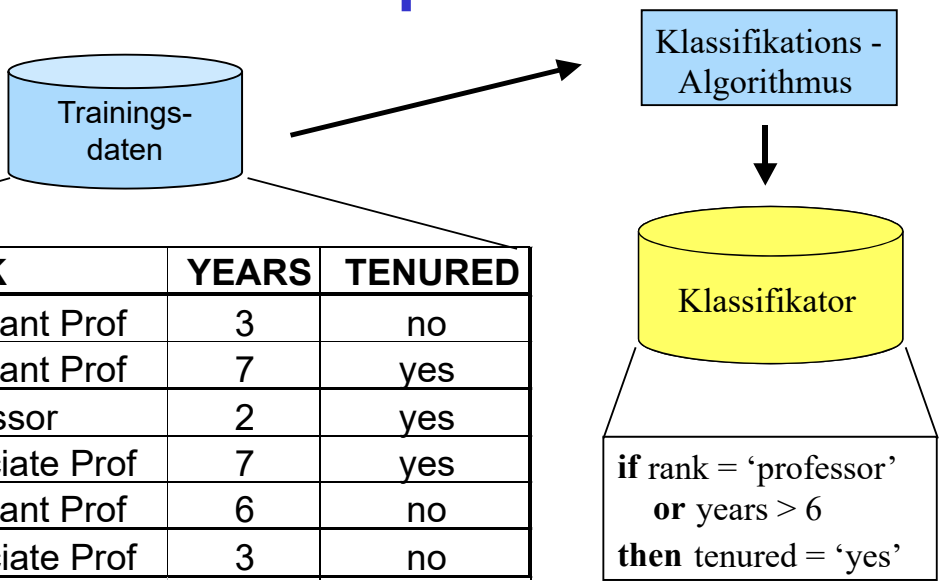
# Klassifikation

- 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



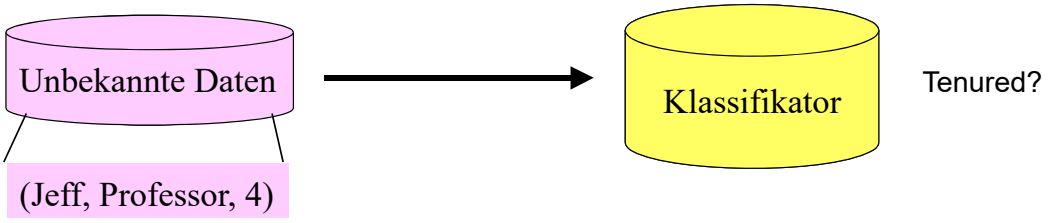
# Klassifikationsprozess

**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 |

**2. Anwendung des Modells zur Vorhersage (Prediction)**

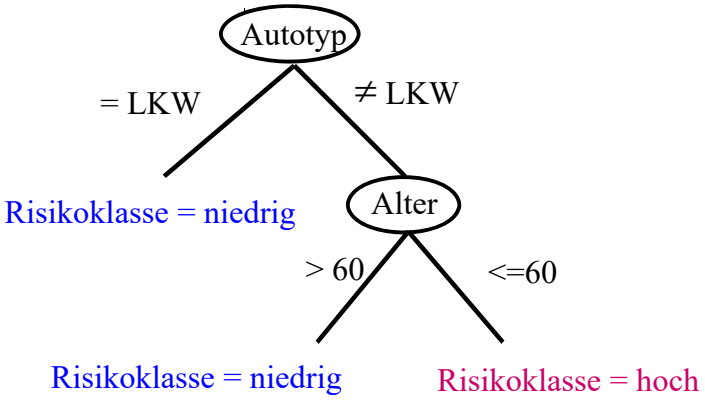





# Entscheidungsbäume

- 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 |

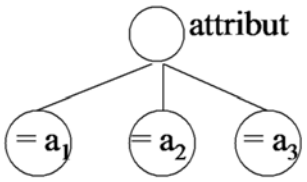


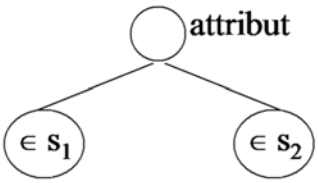

Regeldarstellung:

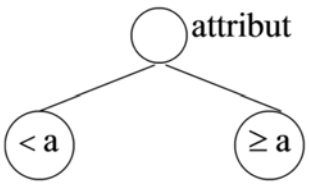
- Entscheidungsbaum ist Baum mit folgenden Eigenschaften:
  - ein innerer Knoten repräsentiert ein Attribut
  - eine Kante repräsentiert einen Test auf dem Attribut des Vaterknotens
  - ein Blatt repräsentiert eine der Klassen
- Anwendung zur Vorhersage:
  - Top-Down-Durchlauf des Entscheidungsbaums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

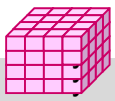
# Konstruktion eines Entscheidungsbaums

- 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)



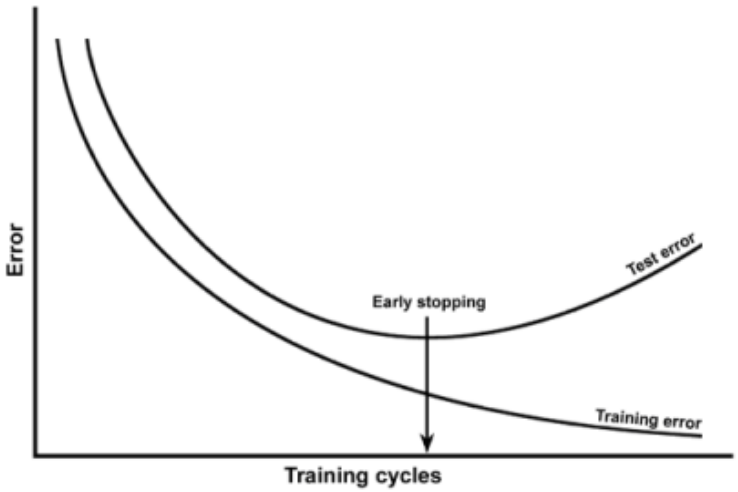



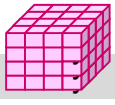





# Bewertung von Klassifikatoren

- Klassifikator ist für die Trainingsdaten optimiert
  - liefert auf der Grundgesamtheit der Daten evtl. schlechtere Ergebnisse (-> Overfitting-Problem)
- Bewertung mit von Trainingsmengen unabhängigen Testmengen
- Klassifikationsgenauigkeit:
  - Anteil der korrekten Klassenzuordnungen in Testmenge
- Klassifikationsfehler:
  - Anteil der falschen Klassenzuordnungen



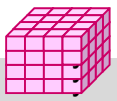
# Bewertung (2)

- Klassifikationsgenauigkeit (*Accuracy*):
  - $n$ Sätze/Objekte, $n_r$ korrekt zugeordnet
  - Accuracy $a = n_r / n$
  - alle Kategorien gehen gleichberechtigt ein
  - bei 2 Kategorien (positive/negative) gilt
    *Accuracy = true-positives+true-negatives / n*
  - Accuracy problematisch, falls eine der Kategorien stark dominiert (zB non-Matches)

| | **real:** class C | **real:** not class C |
|---|---|---|
| **predicted:** class C | True Positive | False Positive |
| **predicted:** not class C | False Negative | True Negative |

- Maße *Recall* / *Precision* / *F-Measure* fokussieren auf eine der Kategorien C (z.B. Risiko, Krankheit, Match) in binären Entscheidungsproblemen
  - seien $n_c$ der $n$ Objekte von dieser Kategorie
  - Klassifikationsmodell ordne $m_c$ Objekte dieser Klasse zu, davon $m_{cr}$ (TP) richtig
  - Recall $r = m_{cr} / n_{c,}$ Precision $p = m_{cr} / m_{c,}$ F-Measure $f = 2\ r\ \ p\ /(r+p)$

- Beispiel binäres Entscheidungsproblem
  - $n=100$, $n_c = 50$, $m_c = 60$, $m_{cr} = 40$



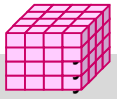


# Bewertung (3)

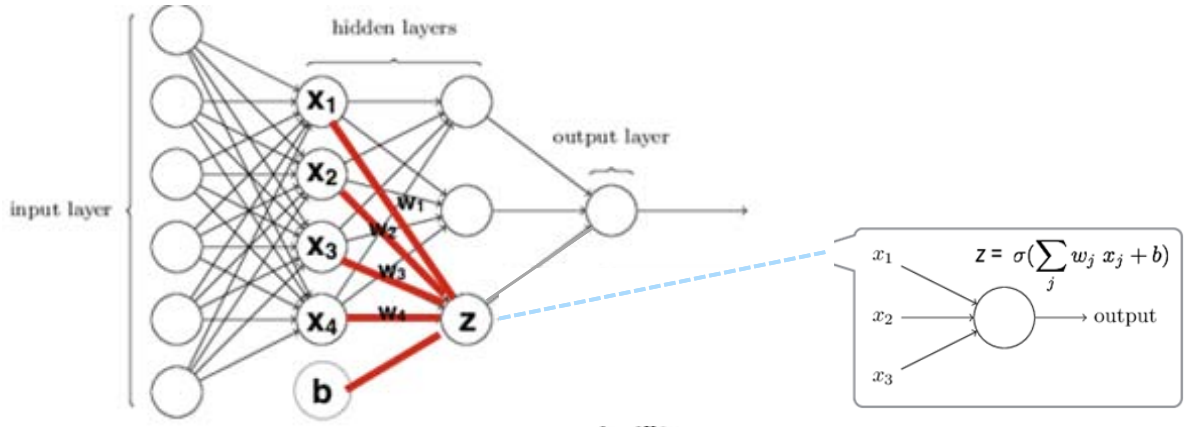
- Konfusionsmatrix

| | A | B | C | *Summe* |
|---|---|---|---|---|
| **a** | **37** | 12 | 4 | *53* |
| **b** | 5 | **23** | 11 | *39* |
| **c** | 12 | 8 | **42** | *62* |
| *Summe* | *54* | *43* | *57* | *154* |

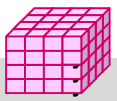
- Accuracy: (37+23+42)/154=102/154 = 66,2%
- Kategoriebezogene Bewertungen
  - Recall für Kat. C:
  - Precision für Kat. C:



# Neuronale Netze



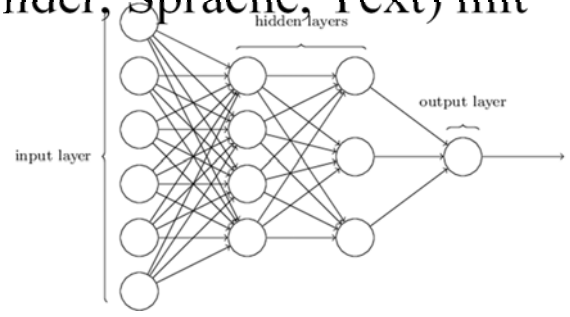

- Neuronales Netz (NN) besteht aus mehreren Schichten
  - Eingabe-/Ausgabeschicht
  - mind. einer verdeckten (hidden) Schicht
- jede Schicht besteht aus Neuronen, die mit anderen Neuronen verbunden sind
  - Verbindungen / Kanten verwenden Zahlen, z.B. Gewichte ($\boldsymbol{w}i \in \mathbb{R}$)
- Deep Learning: mehrere hidden layers

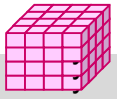




# Deep Learning

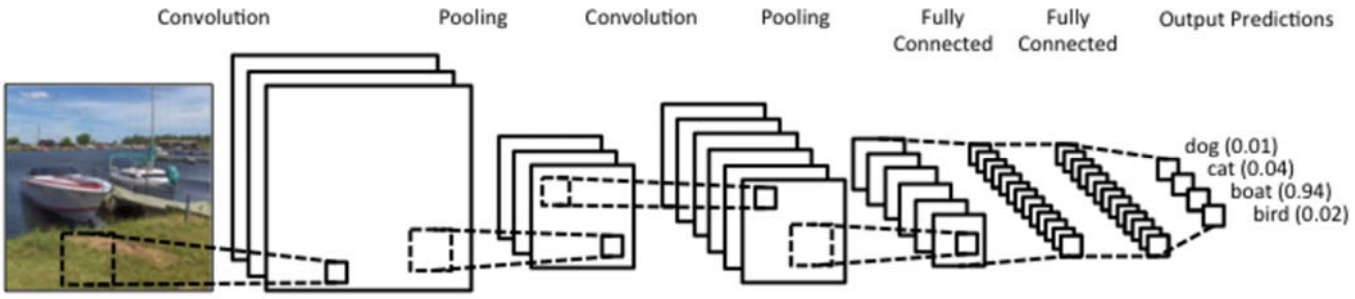
- Nutzung tiefer neuronaler Netze zum Lernen einer Datenrepräsentation auf großen Mengen an Trainingsdaten (Feature Engineering)
- Nutzen des gelernten Wissens für Klassifikation, Vorhersagen etc. vor allem für unstrukturierte Daten (Bilder, Sprache, Text) mit verborgener Struktur
- zahlreiche Anwendungsfälle
  - Erkennung von Bildern
  - Erkennung von Handschriften
  - Spracherkennung
  - Verarbeitung von Texten …
- verschiedene Varianten von Netzen
  - Convolutional deep neural networks (CNN)
  - Recurrent neural networks (RNN), u.a. LSTM (Long short-term memory)
  - Autoencoder networks (Erzeugung verbesserter Repräsentationen)



# Bildklassifizierung

- Nutzung z.B. von Convolutional Neural Networks
  - lokale Filter fassen Pixelaktivität zusammen (convolutional layer)
  - nur ausgewählte Informationen daraus werden weitergereicht und somit überflüssige Information verworfen (pool layer)
  - lokale Filter fassen Pixelaktivität zusammen (convolutional layer)
  - dieser Vorgang kann wiederholt Anwendung finden

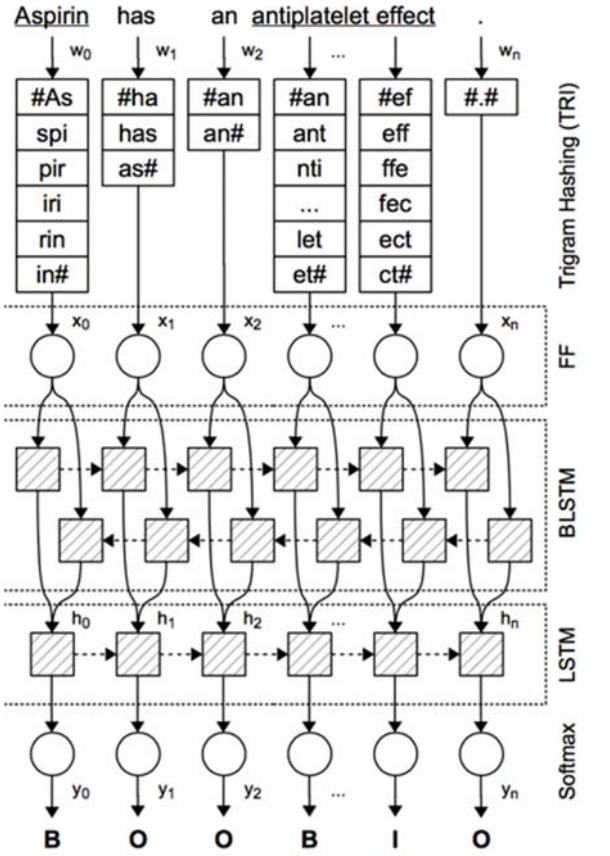



www.wildml.com/2015/11/understanding-convolutional-neural-networks-for-nlp



# Text/Sprachverarbeitung

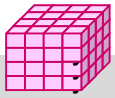
- Lernen der Nachbarschaft von Wörtern (word embeddings) in Texten, um deren semantische Ähnlichkeit zu ermitteln
- Vektorisierung von Texten auf Basis neu gelernter (spezifischer) Embeddings oder mit vortrainierten Embeddings
  - Wortebene: word2vec, glove
  - Zeichenebene: fasttext
- trainierte Datenrepräsentationen nutzen für weitere ML-Aufgaben, zB
  - Named Entity Recognition
  - Entity Linking
  - Machine Translation
  - Spracherkennung
- häufiger Einsatz von *Recurrent Neural Networks (RNN)*
  - oft bidirektional mit Forward + Backward NN




**RNN Pipeline für Named Entity Recognition**
https://arxiv.org/abs/1608.06757

# Zusammenfassung (1)

- Data Mining/ML-Verfahrensklassen: Clusteranalyse, Klassifikation, Assoziationsregeln
- zahlreiche Nutzungsmöglichkeiten: Kundensegmentierung, Vorhersage des Kundenverhaltens, Warenkorbanalyse etc.
- Assoziationsregeln, u.a. zur Warenkorbanalyse
  - Berechnung der Frequent Itemsets mit minimalem Support
  - Ableitung der Regeln mit ausreichender Konfidenz
  - Nutzung der Anti-Monotonie: Itemset ist nur häufig, wenn es alle Teilmengen davon sind
  - A-Priori-Algorithmus: Bottom-Up-Auswertung mit Kandidatengenerierung für häufige Itemsets
  - FP-Tree: schnellere Alternative mit kompakter Repräsentation von Transaktionen und rekursivem Pattern Mining auf Teilbäumen (Divide-and-Conquer)





# Zusammenfassung (2)

- 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
- Bewertung von Klassifikationsergebnissen
  - Accuracy
  - für 2 Klassenprobleme: Precision/Recall/F-Measure
- Neuronale Netze / Deep Learning
  - vielseitige Nutzungsmöglichkeiten, u.a. für Datenvorbereitung und Klassifikationsprobleme, auch für unstrukturierte Daten (Bilder, Texte)
  - hoher Ressourcenbedarf