

Skript zur Vorlesung  
**Knowledge Discovery in Databases**  
im Wintersemester 2010/2011

# Kapitel 3: Klassifikation

Vorlesung+Übungen:  
PD Dr. Peer Kröger, Dr. Arthur Zimek

Skript © 2010 Johannes Aßfalg, Christian Böhm, Karsten Borgwardt, Martin Ester, Eshref Januzaj, Karin Kailing, Peer Kröger, Jörg Sander, Matthias Schubert, Arthur Zimek

[http://www.dbs.ifi.lmu.de/cms/Knowledge\\_Discovery\\_in\\_Databases\\_I\\_\(KDD\\_I\)](http://www.dbs.ifi.lmu.de/cms/Knowledge_Discovery_in_Databases_I_(KDD_I))

## 3. Klassifikation

### *Inhalt dieses Kapitels*

- 3.1 Grundbegriffe der Klassifikation
- 3.2 Bewertung von Klassifikatoren
- 3.3 Bayes-Klassifikatoren
- 3.4 Nächste-Nachbarn-Klassifikatoren
- 3.5 Entscheidungsbaum-Klassifikatoren
- 3.6 Neuronale Netze
- 3.7 Support Vector Machines und Kernel Learning

## 3.1 Grundbegriffe der Klassifikation

### *Das Klassifikationsproblem*

**Gegeben:** Eine Menge  $O$  von Objekten des Formats  $(o_1, \dots, o_d)$  mit Attributen  $A_i$ ,  $1 \leq i \leq d$ , und Klassenzugehörigkeit  $c_i$ ,  $c_i \in C = \{c_1, \dots, c_k\}$

**Gesucht:** die Klassenzugehörigkeit für Objekte aus  $DB \setminus O$  ein *Klassifikator*  $K : DB \rightarrow C$

### **Abgrenzung zum Clustering**

Klassifikation: Klassen a priori bekannt

Clustering: Klassen werden erst gesucht

### **Verwandtes Problem: Vorhersage (Prediction)**

gesucht ist der Wert für ein numerisches Attribut  
Methode z.B. Regression (siehe Kapitel 4).

# Einfacher Klassifikator

## Beispiel

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

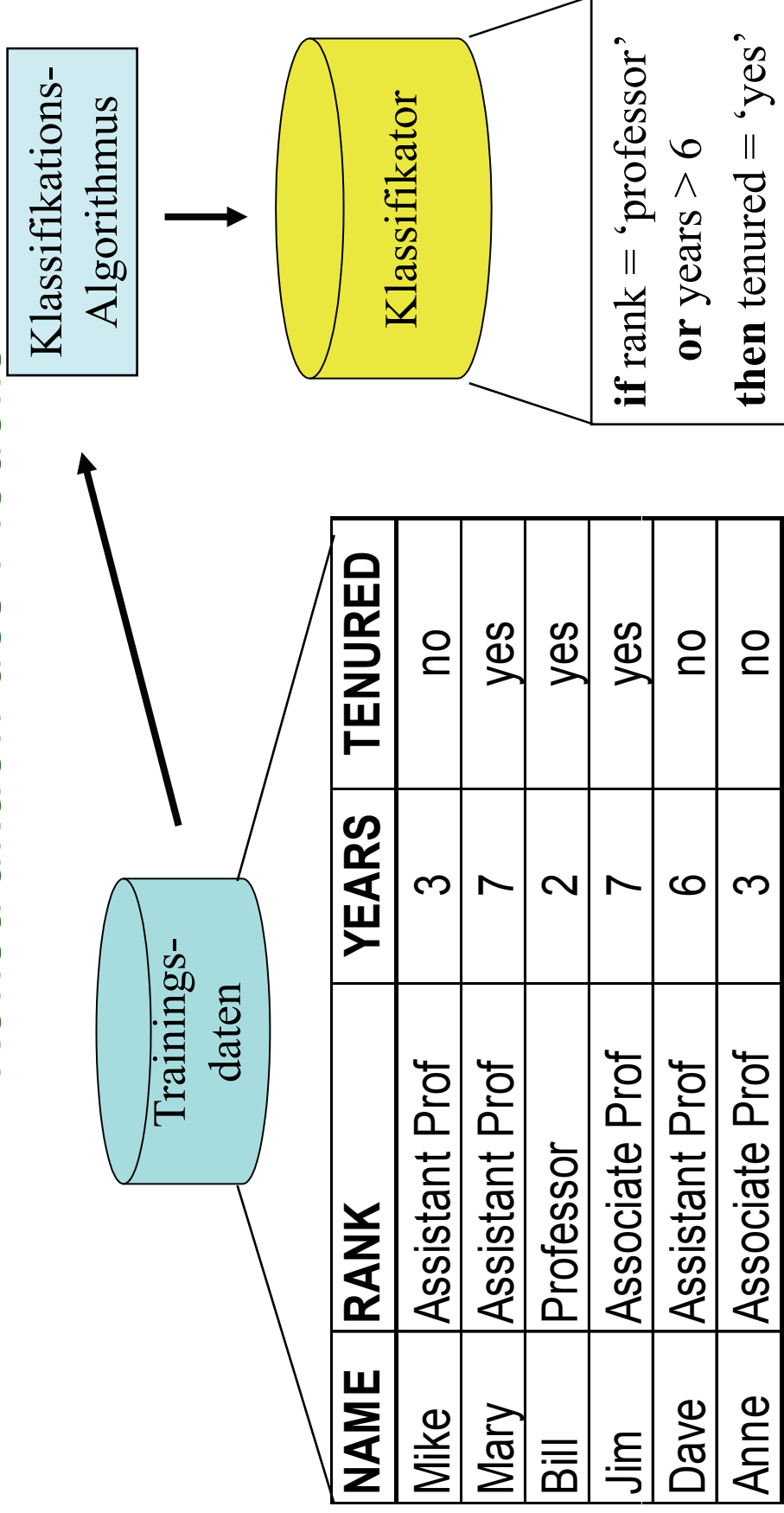
## Einfacher Klassifikator

```

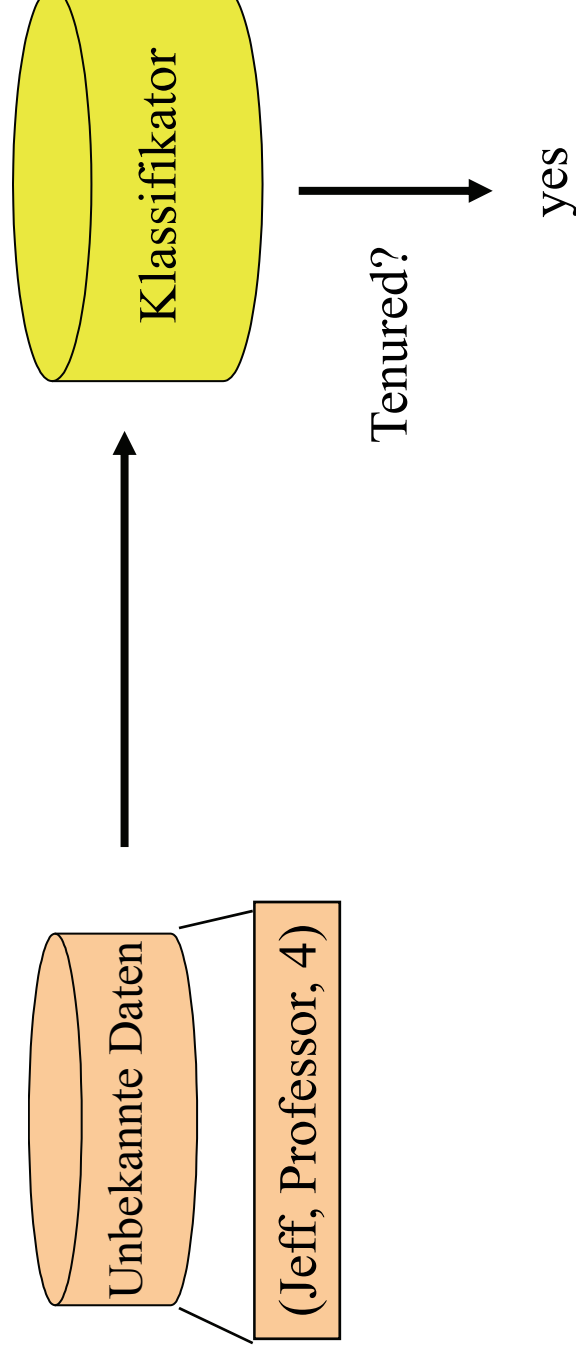
if Alter > 50 then Risikoklasse = Niedrig;
if Alter ≤ 50 and Autotyp=LKW then Risikoklasse=Niedrig;
if Alter ≤ 50 and Autotyp ≠ LKW
    then Risikoklasse = Hoch.

```

## Konstruktion des Modells

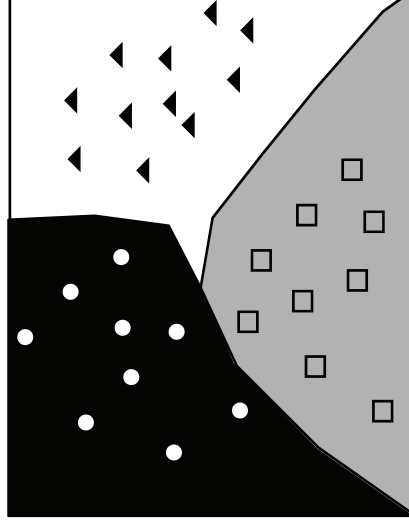
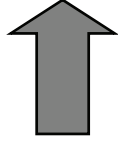
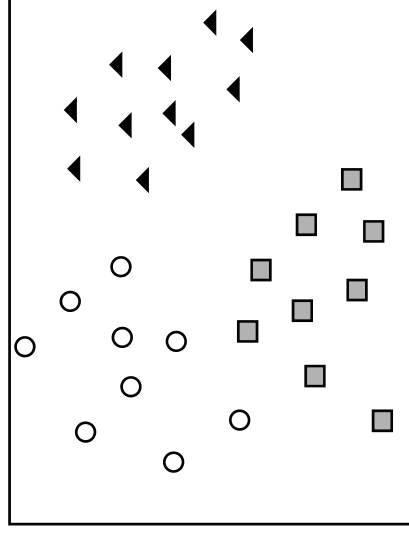


## Anwendung des Modells



manchmal:  keine Klassifikation unbekannter Daten sondern „nur“  
besseres Verständnis der Daten

Trainingsmenge mit 3 Klassen      3 Klassenbereiche (weiß, grau, schwarz)

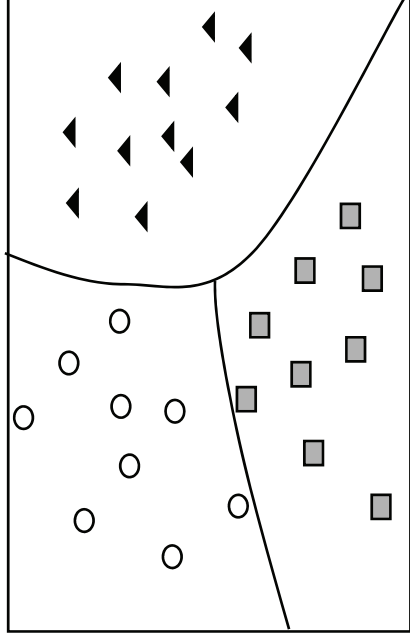


Klassifikatoren legen beim Training im Allgemeinen Klassengrenzen fest.

**Aber:** Es gibt viele Methoden, Klassengrenzen aus Trainingsdaten abzuleiten.

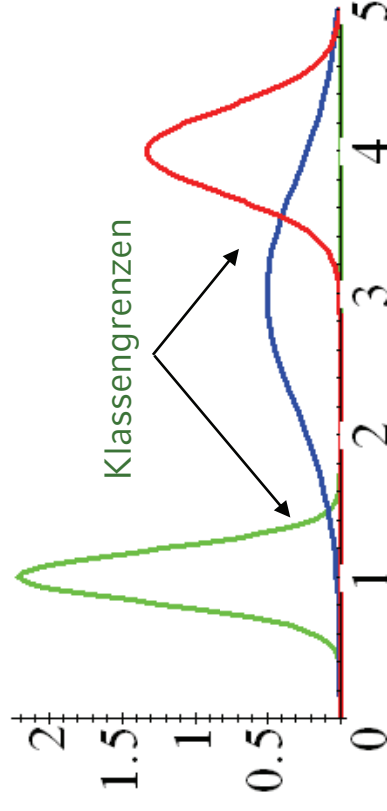
- ⇒ Unterschiedliche Klassifikatoren  
 ( statische KI., Entscheidungsbäume, Support Vektor Maschinen,  
 kNN-Klassifikatoren, neuronale Netze, ...)

# Motivation der Klassifikationsmethoden(1)

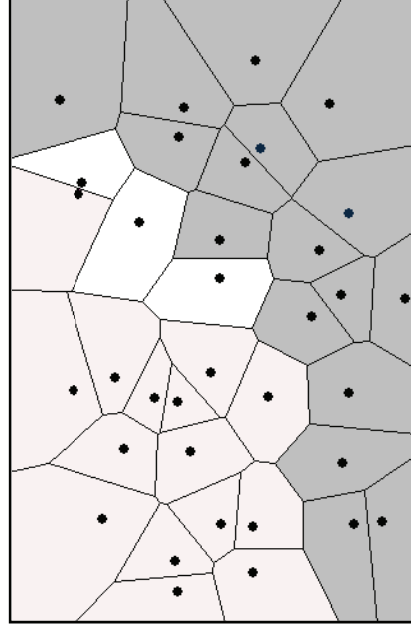


Bayes Klassifikatoren

1-dimensionale Projektion



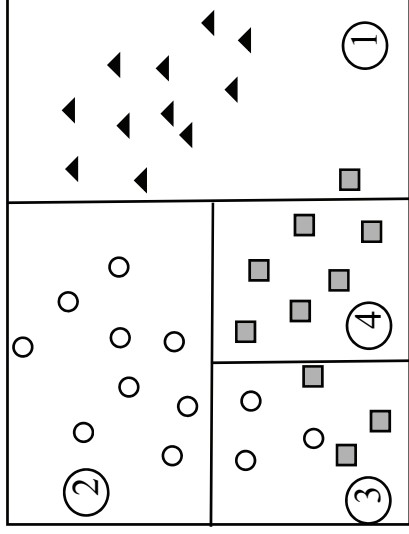
Unterscheidung durch Dichtefunktionen.



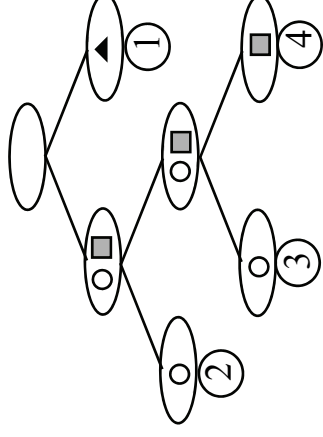
NN-Klassifikator

Unterscheidung durch Voronoi-Zellen  
(1 nächster Nachbar Klassifikator)

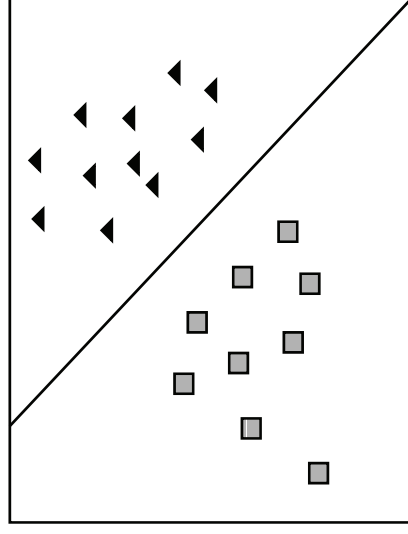




Entscheidungsbäume



Festlegen der Grenzen durch rekursive Unterteilung in Einzeldimension.



Support Vektor Maschinen

Grenzen über lineare Separation

- Es gibt einen (natürlichen, technischen, sozial-dynamischen, ...) (Entscheidungs-)Prozess (im statistischen Sinne), der die beobachtbaren Daten  $O$  als Teilmenge der möglichen Daten  $D$  erzeugt bzw. für ein  $x \in D$  eine Klassenentscheidung für eine Klasse  $c_j \in C$  trifft.
- Die beobachteten Daten sind Beispiele für die Wirkung des Prozesses.
- Es gibt eine ideale (unbekannte) Funktion, die einen Beispiel-Datensatz auf die zugehörige Klasse abbildet:  
 $f: D \rightarrow C$
- Aufgabe des Lernalgorithmus ist es, eine möglichst gute Approximation  $h$  an  $f$  zu finden, eine Hypothese.

- Beispiel:

$$f: \text{Sky} \times \text{AirTemp} \times \text{Humidity} \times \text{Wind} \times \text{Water} \times \text{Forecast} \rightarrow \{\text{Yes}, \text{No}\}$$

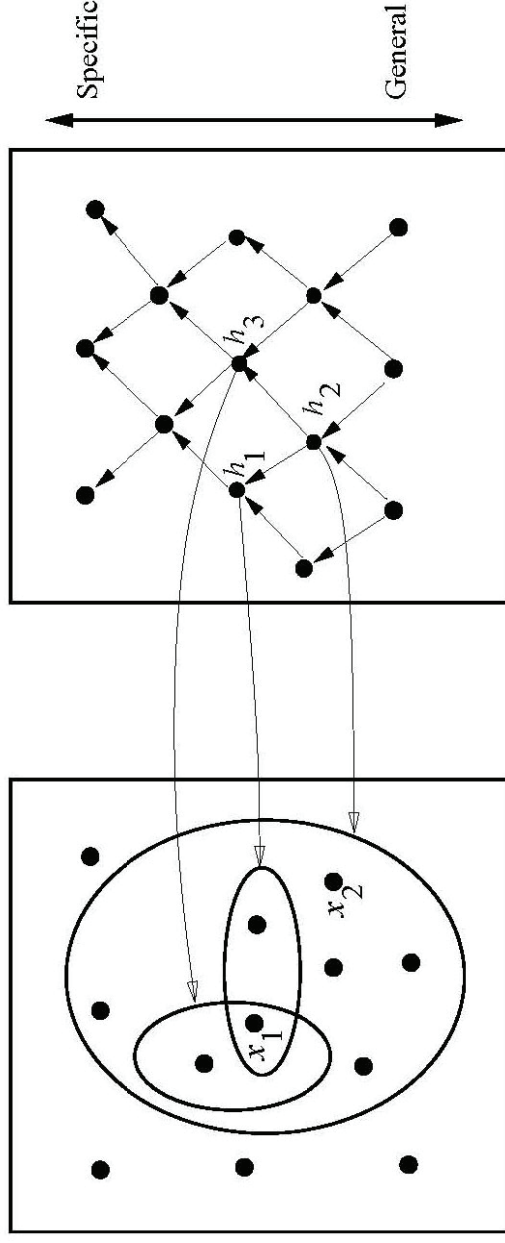
Sky	AirTemp	Humidity	Wind	Water	Forecast	EnjoySport
Sunny	Warm	Normal	Strong	Warm	Same	Yes
Sunny	Warm	High	Strong	Warm	Same	Yes
Rainy	Cold	High	Strong	Warm	Change	No
Sunny	Warm	High	Strong	Cool	Change	Yes

- Gibt es ein generelles Konzept, wann Sport getrieben wird?
- Lernen braucht Annahmen (Bias), z.B., das Konzept ist eine Konjunktion ausgewählter Attributwerte.

Sky	AirTemp	Humidity	Wind	Water	Forecast	EnjoySport
Sunny	Warm	Normal	Strong	Warm	Same	Yes
Sunny	Warm	High	Strong	Warm	Same	Yes
Rainy	Cold	High	Strong	Warm	Change	No
Sunny	Warm	High	Strong	Cool	Change	Yes

- Mögliche Hypothesen für Yes unter dieser Annahme:
  - <Sunny,?,?,?,?,>
  - <?,Warm,?,?,?,?,>
  - <Sunny,Warm,?,?,?,?,>

- Durch die Annahmen eines Klassifikators wird der Raum möglicher Hypothesen (lernbarer Konzepte) definiert.



$x_1 = \langle \text{Sunny, Warm, High, Strong, Cool, Same} \rangle$

$x_2 = \langle \text{Sunny, Warm, High, Light, Warm, Same} \rangle$

$h_1 = \langle \text{Sunny, ?, ?, Strong, ?, ?} \rangle$

$h_2 = \langle \text{Sunny, ?, ?, ?, ?, ?} \rangle$

$h_3 = \langle \text{Sunny, ?, ?, ?, Cool, ?} \rangle$

- Sind komplexere Annahmen sinnvoll (erlaubt auch Disjunktionen, Negationen)?

Konjunktive Hypothesen für Yes:

<Sunny,?,?,?,?,?,>

<Sunny,?,?,Strong,?,?,>

<?,Warm,?,?,?,?,>

<?,Warm,?,Strong,?,?,>

<Sunny,Warm,?,?,?,?,>

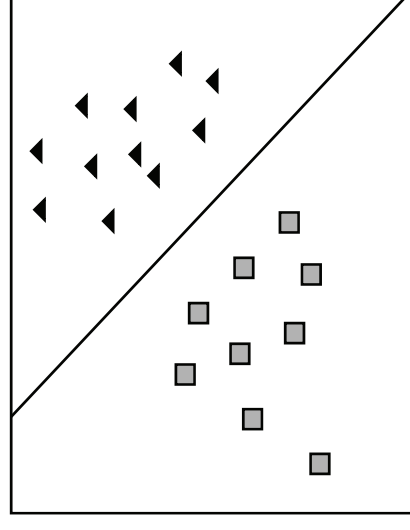
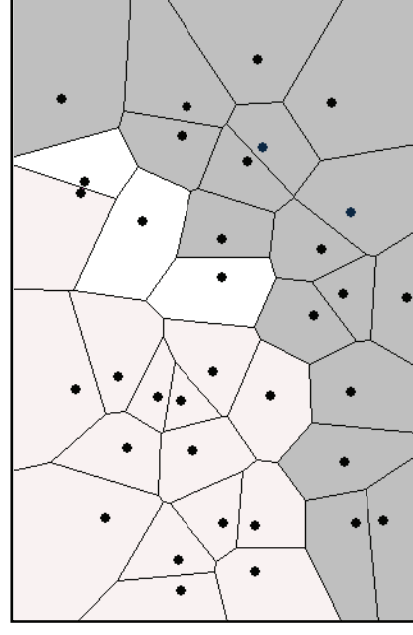
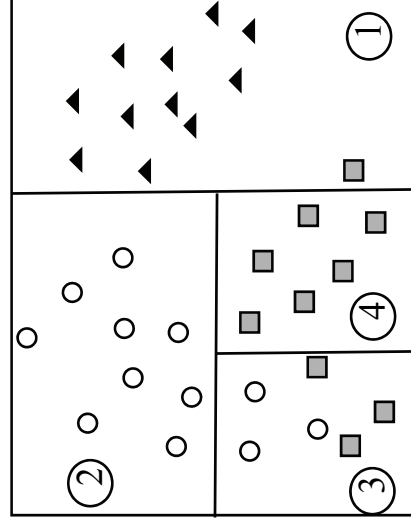
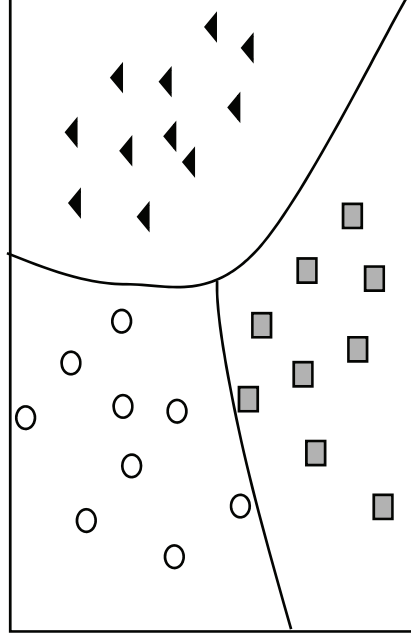
<Sunny,Warm,?,Strong,?,?,>

Neue Daten:

| Sky    | AirTemp | Humidity | Wind   | Water | Forecast | EnjoySport |
|--------|---------|----------|--------|-------|----------|------------|
| Sunny  | Warm    | Normal   | Strong | Cool  | Change   | Yes        |
| Cloudy | Warm    | Normal   | Strong | Cool  | Change   | Yes        |
| Rainy  | Warm    | Normal   | Strong | Cool  | Change   | No         |

- Disjunktives Konzept: *if Sky=Sunny or Sky = Cloudy, then Yes.*
- Ermöglicht aber Auflistung aller Lernbeispiele als Hypothese.
- Allgemein: Ohne einschränkende Annahmen kann Auswendiglernen nicht mehr ausgeschlossen werden.

Verschiedene Klassifikationsalgorithmen basieren auf verschiedenen Annahmen (unterschiedlicher Bias).



## 3.2 Bewertung von Klassifikatoren

### *Grundbegriffe*

Sei  $K$  ein Klassifikator und sei  $TR \subseteq O$  die Trainingsmenge.  $O \subseteq DB$  ist die Menge der Objekte, bei denen die Klassenzugehörigkeit bereits bekannt ist.

#### **Problem der Bewertung:**

- gewünscht ist gute Performanz auf ganz  $DB$ .
  - Klassifikator ist für  $TR$  optimiert.
  - Test auf  $TR$  erzeugt in der Regel viel bessere Ergebnisse, als auf  $DB \setminus TR$ .
- Daher kein realistisches Bild der Performanz auf  $DB$ .

⇒ *Overfitting*



## *Train-and-Test*

Bewertung ohne *Overfitting* durch Aufteilen von  $O$  in :

- *Trainingsmenge TR*
  - zum Lernen des Klassifikators (Konstruktion des Modells)
- *Testmenge TE*
  - zum unabhängigen Bewerten des Klassifikators
- *Ziel: Abschätzen der Erfolg- bzw. Fehlerrate des Klassifikators*
  - Daher: Test-Daten müssen unabhängig von Trainingsdaten sein
  - Trainings- und Testdaten sollen das Problem angemessen widerspiegeln  
(z.B. proportionale Anteile der verschiedenen Klassen)

## *m*-fache Überkreuz-Validierung

- sinnvolle manuelle Aufteilung in Trainings- und Testmenge nicht trivial
- Train-and-Test nicht anwendbar, wenn nur wenige Objekte mit bekannter Klassenzugehörigkeit vorhanden sind.
- Stattdessen: *m*-fache Überkreuz-Validierung (*m*-fold Cross-

## *Validation*)

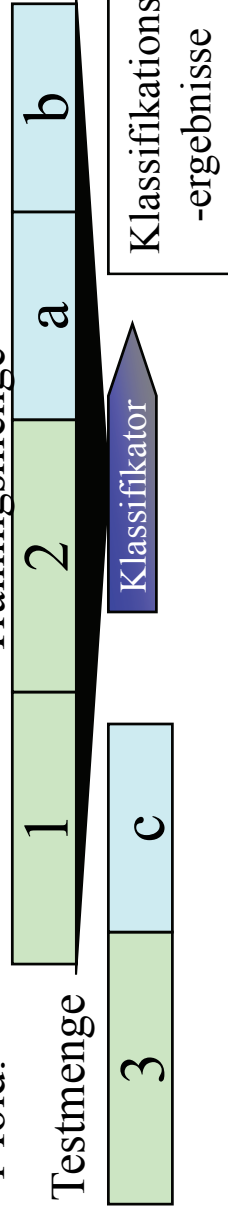
- teile die Menge  $O$  zufällig in  $m$  gleich große Teilmengen
- verwende jeweils  $m-1$  Teilmengen zum Training und die verbleibende Teilmenge zur Bewertung
- kombiniere die erhaltenen  $m$  Klassifikationsfehler
- Wiederhole das Verfahren mehrmals

## Ablauf 3-fache Überkreuzvalidierung (3-fold Cross Validation)

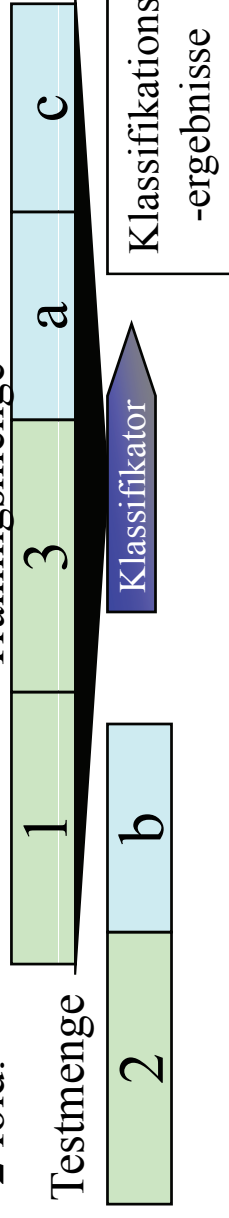
Sei  $n = 3$  : Menge aller Daten mit Klasseninformation die zu Verfügung stehen



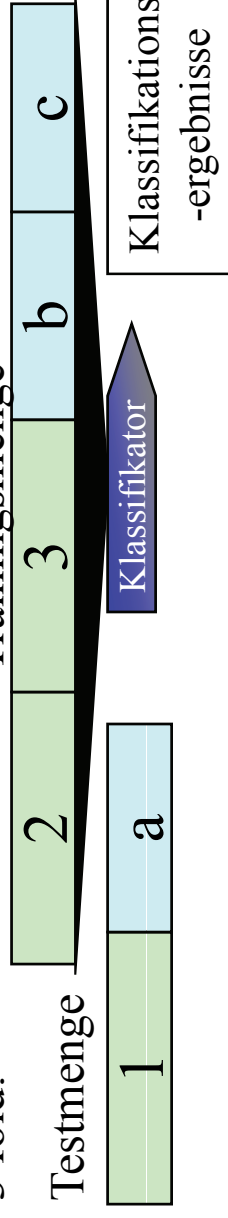
1 fold:



2 fold:



3 fold:



gesamtes  
Klassifikations-  
ergebnis

## *zusätzliche Anforderung: Stratifikation*

- Proportionalität der Klassen erhalten
- zumindest: jede Klasse sollte in Trainingsmenge enthalten sein
- sinnvoll: Verteilung der Klassen sollte in Trainings- und Testmenge die Verteilung auf dem gegebenen Problem widerspiegeln
- Standard: 10-fache, stratifizierte Kreuzvalidierung, 10-mal wiederholt (Erfahrungswerte)

## Alternative: Bootstrap

bilden einer Trainingsmenge aus einer gegebenen Datenmenge durch Ziehen mit Zurücklegen.

- jedes Sample hat die gleiche Größe wie die ursprüngliche Trainingsmenge
- ein Sample enthält durchschnittlich 63% der Ausgangsbeispiele (einige mehrfach, etwa 37% gar nicht):
  - ein einzelnes Beispiel in einem Datensatz mit  $n$  Beispielen hat bei jedem Ziehen die Chance  $1/n$  gezogen zu werden, wird also mit Wahrscheinlichkeit  $1-1/n$  **nicht** gezogen
  - nach  $n$ -mal Ziehen ist ein bestimmtes Element mit Wahrscheinlichkeit

$$\left(1 - \frac{1}{n}\right)^n \text{ nicht gezogen worden}$$

- für große  $n$  ist  $\left(1 - \frac{1}{n}\right)^n \approx e^{-1} \approx 0.368$

- daher auch der Name „0.632 bootstrap“ für diese Sampling-Methode
- Im Allgemeinen eher optimistische Fehlerschätzung

*Alternative: Leave-one-out (auch: Jackknife)*

- Trainingsmenge wird gebildet durch Weglassen eines einzigen Elementes, dieses wird als Test-Objekt verwendet
- Verfahren wird wiederholt für alle Objekte der gelabelten Daten, Fehler-Abschätzung gemittelt
- Vorteil: kein Zufallselement
- Nachteil: garantiert nicht stratifiziert
- Im Allgemeinen eher pessimistische Fehlerschätzung

## Ergebnis des Tests : Konfusionsmatrix (confusion matrix)

tatsächliche Klasse ..

klassifiziert als ...

|          | Klasse1 | Klasse 2 | Klasse 3 | Klasse 4 | other |
|----------|---------|----------|----------|----------|-------|
| Klasse 1 | 35      | 1        | 1        | 1        | 4     |
| Klasse 2 | 0       | 31       | 1        | 1        | 5     |
| Klasse 3 | 3       | 1        | 50       | 1        | 2     |
| Klasse 4 | 1       | 0        | 1        | 10       | 2     |
| other    | 3       | 1        | 9        | 15       | 13    |

korrekt klassifizierte Objekte

Aus der Konfusionsmatrix lassen sich u.a. folgende Kennzahlen berechnen :  
Accuracy, Classification Error, Precision und Recall.

## Gütemaße für Klassifikatoren

Sei  $K$  ein Klassifikator,  $TR \subseteq O$  die Trainingsmenge,  $TE \subseteq O$  die Testmenge. Bezeichne  $C(o)$  die tatsächliche Klasse eines Objekts  $o$ ,  $K(o)$  die von  $K$  vorhergesagte.

- *Klassifikationsgenauigkeit* (classification accuracy) von  $K$  auf  $TE$ :

$$G_{TE}(K) = \frac{|\{o \in TE \mid K(o) = C(o)\}|}{|TE|}$$

- *Tatsächlicher Klassifikationsfehler* (true classification error)

$$F_{TE}(K) = \frac{|\{o \in TE \mid K(o) \neq C(o)\}|}{|TE|}$$

- *Beobachteter Klassifikationsfehler* (apparent classification error)

$$F_{TR}(K) = \frac{|\{o \in TR \mid K(o) \neq C(o)\}|}{|TR|}$$



## Recall:

Anteil der Testobjekte einer Klasse  $i$ , die richtig erkannt wurden.

Sei  $C_i = \{o \in TE \mid C(o) = i\}$ , dann ist

$$\text{Recall}_{TE}(K, i) = \frac{|\{o \in C_i \mid K(o) = C(o)\}|}{|C_i|}$$

## Precision:

Anteil der zu einer Klasse  $i$  zugeordneten Testobjekte, die richtig erkannt wurden.

Sei  $K_i = \{o \in TE \mid K(o) = i\}$ , dann ist

$$\text{Precision}_{TE}(K, i) = \frac{|\{o \in K_i \mid K(o) = C(o)\}|}{|K_i|}$$

Zugeordnete Klasse  $K(o)$

|   |   |   |  |  |  |
|---|---|---|--|--|--|
|   | 1 | 2 |  |  |  |
| 1 |   |   |  |  |  |
| 2 |   |   |  |  |  |
|   |   |   |  |  |  |

Tatsächl. Klasse  $C(o)$

$K_i$  (red box around row 2)

$C_i$  (red box around column 2)

## *weitere Gütekriterien für Klassifikatoren*

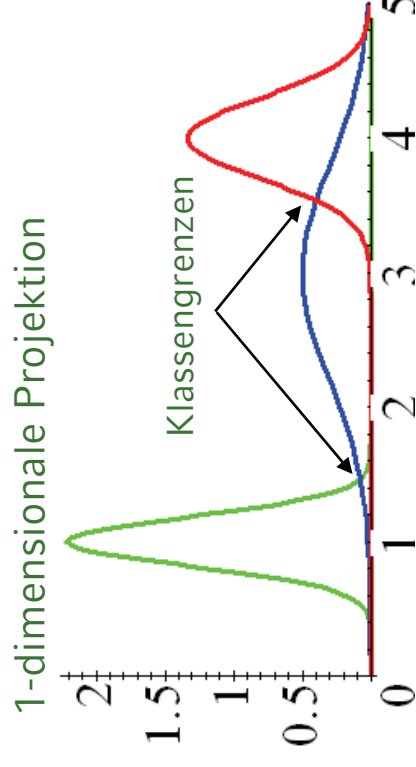
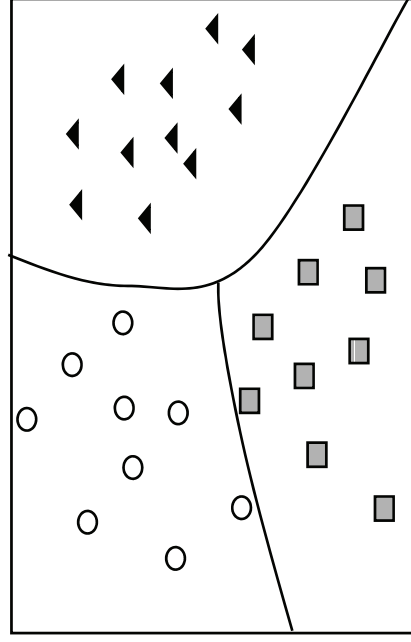
- **Kompaktheit** des Modells
  - z.B. Größe eines Entscheidungsbaums
- **Interpretierbarkeit** des Modells
  - Wieviel Einsichten vermittelt das Modell dem Benutzer?
- **Effizienz**
  - der Konstruktion des Modells
  - der Anwendung des Modells
- **Skalierbarkeit**
  - für große Datenmengen
  - für sekundärspeicherresidente Daten
- **Robustheit** gegenüber Rauschen und fehlenden Werten

## 3.3 Bayes-Klassifikatoren

### Was sind Bayes-Klassifikatoren?

#### Statistische Klassifikatoren

- Klassen werden durch statistische Prozesse beschrieben
- Beruht auf dem Satz von Bayes
- Bestimme Wahrscheinlichkeiten mit denen jeder Prozess das Objekt erklärt  
(Class-Membership-Probability)
- Vorhersage der wahrscheinlichsten Klasse  
(Maximum Likelihood Classification)



## Grundlagen statistischer Klassifikatoren

1. A-priori und A-posteriori Wahrscheinlichkeiten
2. Regel von Bayes
3. „Maximum Likelihood“ Klassifikation

## Klassifikatoren und Statistische Prozesse

1. Naive Bayes
2. Bayes Netzwerke
3. LDA
4. multivariate Gauss-Prozesse

## Grundlagen

Regeln und Fakten zur Klassifikation werden mit Hilfe des Satzes von Bayes als bedingte Wahrscheinlichkeiten formuliert

A-Priori-Wahrscheinlichkeiten modellieren Faktenwissen über die

Häufigkeit einer Klasse und das Auftreten von Merkmalen, z.B.

- 20% der Objekte sind Äpfel
  - 30% sind Orangen
  - 50% der Objekte sind rund
  - 40% haben Farbe orange
- } A-Priori Wahrsch. f. Klassenzugehörigk.
- } A-Priori Merkmalshäufigkeit

Bedingte Wahrscheinlichkeiten („A-Posteriori“) modellieren

Zusammenhänge zwischen Klassen und Merkmalen:

- 100% der Orangen sind rund:  $P(\text{rund} \mid \text{Orange}) = 100\%$
- 100% der Äpfel sind rund:  $P(\text{rund} \mid \text{Apfel}) = 100\%$
- 90% der Orangen sind orange:  $P(\text{orange} \mid \text{Orange}) = 90\%$

Bei einem gegebenen Merkmals-Vektor  $M$  lässt sich die Wahrscheinlichkeit der Klassenzugehörigkeit zu Klasse  $C_i$  mit dem Satz von Bayes ermitteln:

$$P(C_i | M) = \frac{P(M | C_i) \cdot P(C_i)}{P(M)} = \frac{P(M | C_i) \cdot P(C_i)}{\sum_{c_j \in C} P(C_j) \cdot P(M | C_j)}$$

Im Beispiel: Wahrscheinlichkeit, dass ein oranges Objekt eine Orange ist:

$$P(\text{Orange} | \text{orange}) = \frac{P(\text{orange} | \text{Orange}) \cdot P(\text{Orange})}{P(\text{orange})} = \frac{0.9 \cdot 0.3}{0.4} = 0.675$$

Die entsprechenden Wahrscheinlichkeiten werden aus den Trainingsdaten geschätzt.

Der Bayes-Klassifikator schätzt die Wahrscheinlichkeit der Klassenzugehörigkeit eines Merkmalsvektors

Zur eindeutigen Zuordnung eines Klassen-Labels geht man meist nach dem Prinzip „Maximum Likelihood“ vor:

$$C = \operatorname{argmax}_{C_i} P(C_i | M) = \operatorname{argmax}_{C_i} \frac{P(M | C_i) \cdot P(C_i)}{P(M)} = \operatorname{argmax}_{C_i} P(M | C_i) \cdot P(C_i)$$

Da  $P(M)$  bei allen  $C_i$  gleich ist, ist nur das Produkt zu optimieren. Beispiel:

$$\left. \begin{array}{l} - P(\text{Apfel} | M) = 32\% \\ - P(\text{Orange} | M) = 32\% \\ - P(\text{Kiwi} | M) = 36\% \end{array} \right\} \Rightarrow C = \text{Kiwi}$$

## „A-priori“ Wahrscheinlichkeiten

Meistens: relative Häufigkeit in den Trainingsdaten.

$$\text{Bsp: 7 Orangen, 2 Äpfel, 1 Stein} \Rightarrow P(\text{Orange}) = \frac{7}{7+2+1} = 70\%$$

## „A-Posteriori“ Wahrscheinlichkeiten

- Statistischer Prozess modelliert Zusammenhänge zwischen Merkmalen und einer Klasse
- Unterschiede verwendeter Prozesse:
  - Abhängigkeit der Merkmale ( Korrelation oder Unabhängigkeit)
  - Verwendete Verteilungsfunktionen der Merkmalswerte (diskret, Normalverteilung, Multinomialverteilung...)
  - Beschaffenheit der Objekte (Vektor, Sequenz...)



## Diskrete Merkmale

Auszählen relativer Häufigkeiten

Bsp:

$$P(\text{Form} = \text{rund} \mid A) = \frac{3}{4} = 75\%$$

$$P(\text{Farbe} = \text{grün} \mid A) = \frac{2}{4} = \frac{1}{2} = 50\%$$

$$P(\text{Form} = \text{oval} \mid A) = \frac{0}{4} = 0\%$$

| ID | Form  | Farbe  | Klasse |
|----|-------|--------|--------|
| 1  | rund  | orange | A      |
| 2  | rund  | grün   | A      |
| 3  | rund  | gelb   | A      |
| 4  | eckig | grün   | A      |
| 5  | oval  | weiß   | B      |

Problem: (Form = oval)  $\Rightarrow$  Klasse  $\neq$  A

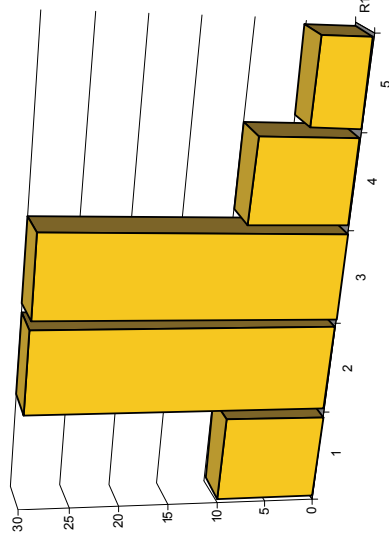
Man verwendet häufig „Smoothing“, d.h.  $P(x \mid \text{Klasse}) > \varepsilon$ .  
mit  $0 < \varepsilon \ll 1$ .

D.h.  $P(\text{Form} = \text{oval} \mid A) = \max\left(\frac{0}{4}, \varepsilon\right) = \varepsilon$

## Kontinuierliche metrische Attribute

### diskrete Approximation

- P ( 9.0 < Durchmesser ≤ 9.5 | Orange) = 10%
- P ( 9.5 < Durchmesser ≤ 10.0 | Orange) = 30%
- P (10.0 < Durchmesser ≤ 10.5 | Orange) = 30%
- P (10.5 < Durchmesser ≤ 11.0 | Orange) = 10%
- P (11.0 < Durchmesser ≤ 11.5 | Orange) = 5%



### Wahrscheinlichkeits-Dichtefunktionen

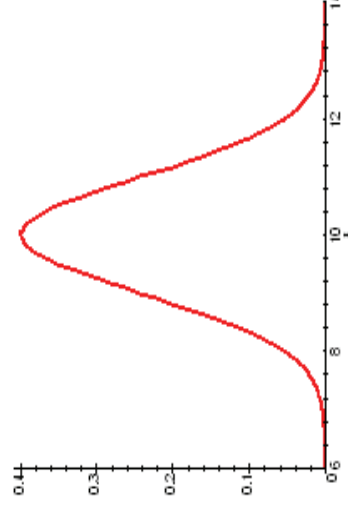
z.B. Orangen haben einen Durchmesser von  $10 \pm 1$  cm:

$$p(\text{Durchmesser} \mid \text{Orange}) = N(10, 1)$$

meist Berechnung nach Normalverteilung:

$$P(x \mid C) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

wobei  $\mu = \frac{\sum_{x \in TR} x}{|TR|}$  und  $\sigma = \sqrt{\frac{\sum_{x \in TR} (x-\mu)^2}{|TR|}}$



# Motivation

Bei hochdimensionalen Merkmalsvektoren schwierige Schätzung der bedingten Wahrscheinlichkeiten  $P(M | C)$  und damit  $P(C | M)$ :

- $M$  besteht aus vielen einzelnen Komponenten, die UND-verknüpft sind:

$$P(C | M_1 \wedge M_2 \wedge \dots) = \frac{P(M_1 \wedge M_2 \wedge \dots | C) \cdot P(C)}{P(M_1 \wedge M_2 \wedge \dots)}$$

- Bei  $d$  verschiedenen Merkmalen und jeweils  $r$  verschiedenen Werten ergeben sich  $r^d$  verschiedene Merkmalskombinationen

## Probleme:

- Die Wahrscheinlichkeiten lassen sich nicht mehr abspeichern
- Man bräuhete  $\gg r^d$  Trainingsdatensätze, um die Wahrscheinlichkeit der einzelnen Merkmalskombinationen überhaupt ermitteln zu können

Lösung dieses Problems beim naiven Bayes-Klassifikator:

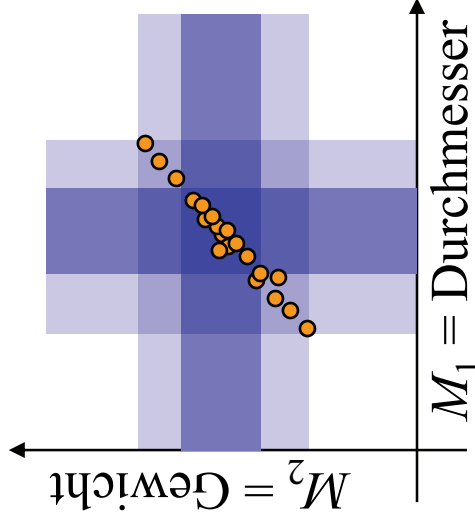
## Annahme der bedingten Unabhängigkeit

d.h. bei jeder einzelnen Klasse werden die Merkmale so behandelt als wären sie voneinander statistisch unabhängig:

$$P(M_1 \wedge M_2 | C) = P(M_1 | C) \cdot P(M_2 | C)$$

Was bedeutet dies?

Klasse=Orange:



- Annahme kann falsch sein
- Dies führt *nicht* unbedingt dazu, dass die Klassifikation versagt
- Aber schlechte Leistung, wenn...
  - alle Merkmale bei mehreren Klassen etwa gleich verteilt sind
  - Unterschiede nur in „Relationen“ der Merkmale zueinander

Damit ist die Wahrscheinlichkeit der Zugehörigkeit zur Klasse  $C_i$ :

$$\begin{aligned} P(C_i | M_1 \wedge M_2 \wedge \dots) &= \frac{P(C_i) \cdot P(M_1 \wedge M_2 \wedge \dots | C_i)}{P(M_1 \wedge M_2 \wedge \dots)} \\ &= \frac{P(C_i) \cdot \prod_j P(M_j | C_i)}{\sum_k P(C_k) \prod_j P(M_j | C_k)} \end{aligned}$$

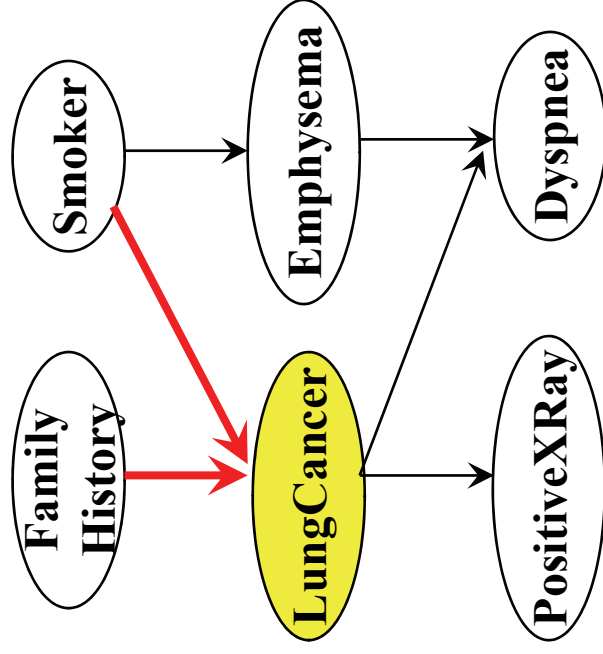
Auch hier ist der Nenner für alle Klassen gleich, so dass nur der Zähler zu maximieren ist:

$$C = \operatorname{argmax}_{C_i} \{P(C_i) \cdot \prod_j P(M_j | C_i)\}$$

## Grundbegriffe

- Graph mit Knoten = *Zufallsvariable* und Kante = *bedingte Abhängigkeit*
- Jede Zufallsvariable ist bei gegebenen Werten für die Vorgänger-Variablen bedingt unabhängig von allen Zufallsvariablen, die keine Nachfolger sind.
- Für jeden Knoten (Zufallsvariable):  
Tabelle der bedingten Wahrscheinlichkeiten
- Trainieren eines Bayes-Netzwerkes
  - bei gegebener Netzwerk-Struktur und allen bekannten Zufallsvariablen
  - bei gegebener Netzwerk-Struktur und teilweise unbekanntem Zufallsvariablen
  - bei apriori unbekannter Netzwerk-Struktur

## Beispiel



|     |       |              |              |                     |
|-----|-------|--------------|--------------|---------------------|
| LC  | FH, S | FH, $\neg$ S | $\neg$ FH, S | $\neg$ FH, $\neg$ S |
| ~LC | 0.8   | 0.5          | 0.7          | 0.1                 |
|     | 0.2   | 0.5          | 0.3          | 0.9                 |

bedingte Wahrscheinlichkeiten  
für LungCancer

bei gegebenen Werten für FamilyHistory und Smoker liefert der Wert für Emphysema keine zusätzliche Information über LungCancer.

- Modelliere alle Klassen als multivariate Normalverteilungen
- Berücksichtigt Korrelationen der Attribute
- Varianzen und Korrelationen für alle Klassen gleich

Basis multivariate Normalverteilung (Gauss-Verteilung)

$$P(x | C) = \frac{1}{\sqrt{(2\pi)^d |\Sigma|}} e^{-\frac{1}{2} \cdot (x-\mu)^T \cdot (\Sigma)^{-1} \cdot (x-\mu)}$$

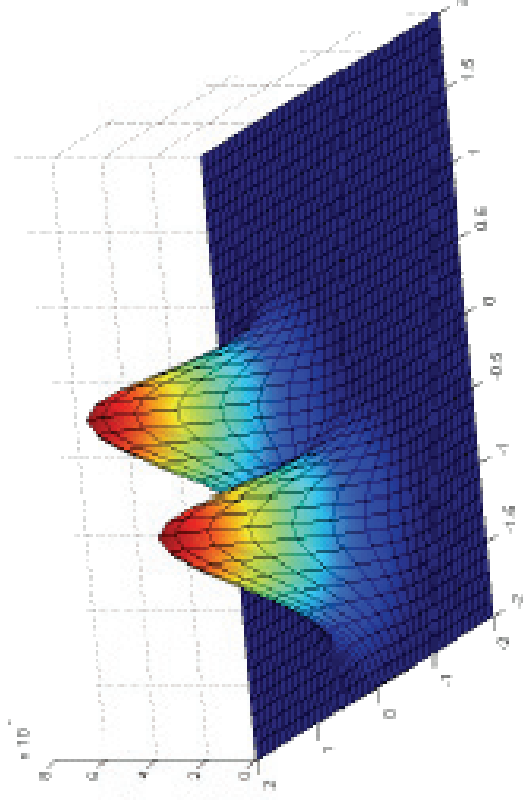
Erwartungsvektor:  $\mu = \frac{\sum_{x \in TR} x}{|TR|}$

Kovarianzmatrix :

$$\Sigma(i, j) = \frac{\sum_{x \in TR} (x_i - \mu_i) \cdot (x_j - \mu_j)}{|TR|}$$

**Eigenschaften:**

- Korrelation zwischen i und j
- Varianz in der Diagonalen





## Training:

- Bestimme  $\mu_C$  und  $\Sigma_C$  für alle Klassen  $C$ .

- Mittlere globale Kovarianzmatrix  $\Sigma$ .  
(Gewichteter Durchschnitt der Kovarianzmatrizen aller Klassen)
- $$\Sigma = \frac{\sum_{C_i \in C} \Sigma_{C_i}}{|C|}$$

## Klassifikation:

$$\arg \max_{C_i \in C} P(x | C_i) = \arg \max_{C_i \in C} \left( \frac{1}{\sqrt{(2\pi)^d |\Sigma|}} e^{-\frac{1}{2}(x-\mu_{C_i})^T \cdot (\Sigma)^{-1} \cdot (x-\mu_{C_i})} \cdot P(C_i) \right)$$

$$= \arg \max_{C_i \in C} \left( -\frac{1}{2} \cdot (x - \mu_{C_i})^T \cdot (\Sigma)^{-1} \cdot (x - \mu_{C_i}) + \log(P(C_i)) \right)$$

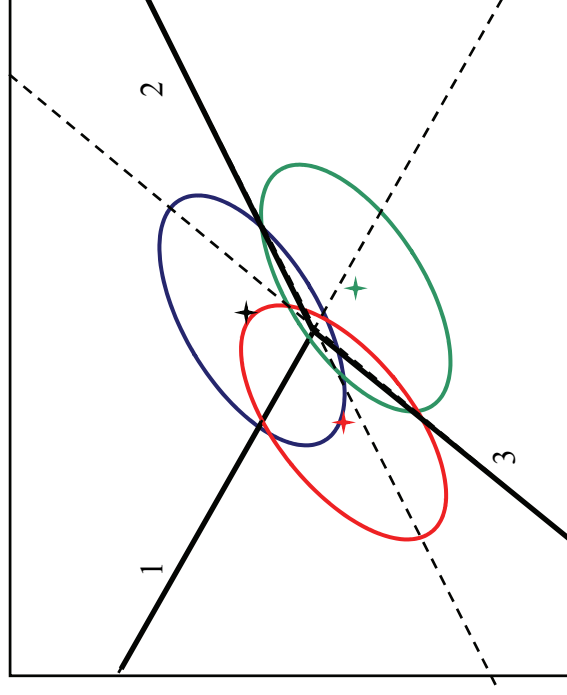
$$= \arg \max_{C_i \in C} \left( \underline{x^T (\Sigma)^{-1} \mu_{C_i} - \frac{1}{2} \mu_{C_i}^T (\Sigma)^{-1} \mu_{C_i} + \log(P(C_i))} \right) = \sigma_{C_i}(x)$$

↙ Lineare Diskriminanzfunktion

- Beobachtung: Da nur Erwartungswerte unterschiedlich  
⇒ Lineare Separation
- Man muss nicht die Wahrscheinlichkeit berechnen.
- Es reicht die Auswertung der folgenden Diskriminanzfunktion:

$$\sigma_{C_i}(x) = x^T \Sigma^{-1} \mu_{C_i} - \frac{1}{2} \mu_{C_i}^T \Sigma^{-1} \mu_{C_i} + \log P(C_i)$$

Klasse mit maximalem  $\sigma_C(x)$  wird vorhergesagt.

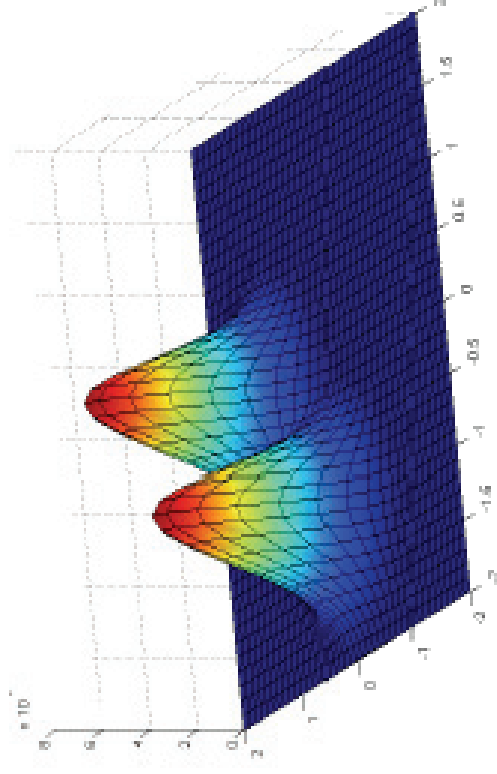


# Multivariate Gauss-Prozesse

- Modelliere jede Klasse als multivariate Normalverteilung (Vektoren im  $R^d$ )
- Berücksichtigt Korrelationen der Attribute
- Hier: Varianzen und Korrelationen für alle Klassen **individuell**
- Berechnung der Wahrscheinlichkeiten zur Klassifikation (Maximum Likelihood)

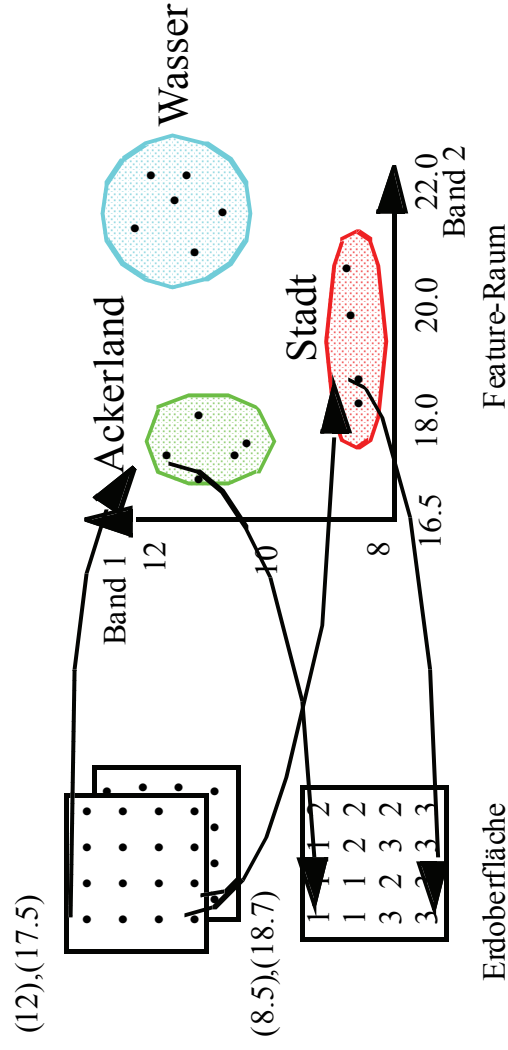
## Probleme:

Braucht sehr viel Trainingsobjekte für jede Klasse, um signifikante Korrelationswerte zu bestimmen.



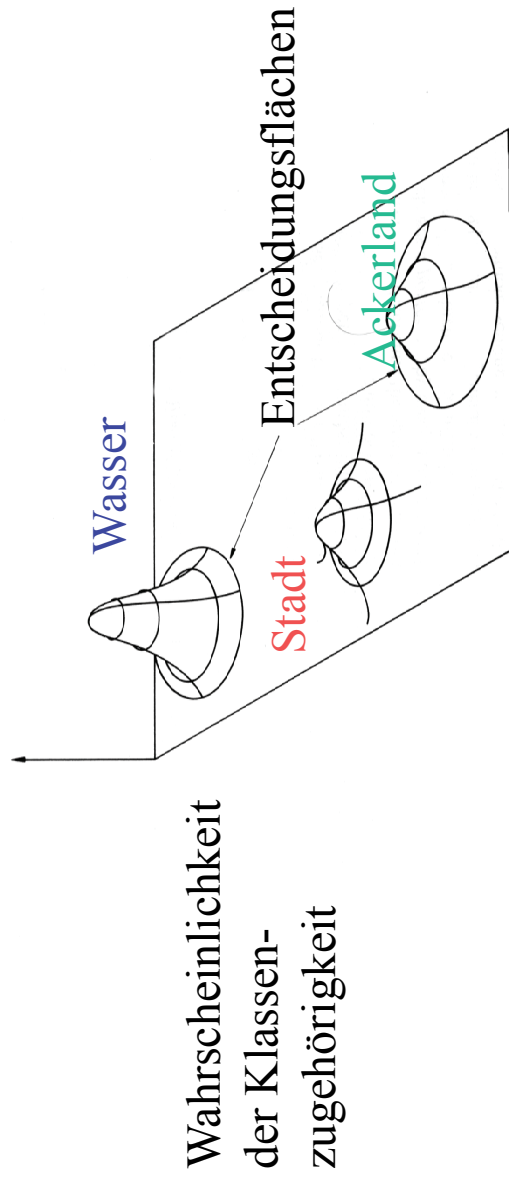
## Motivation

- *automatische* Interpretation von  $d$  Rasterbildern eines bestimmten Gebiets für jedes Pixel ein  $d$ -dimensionaler Grauwertvektor  $(o_1, \dots, o_d)$
- verschiedene Oberflächenbeschaffenheiten der Erde besitzen jeweils ein charakteristisches Reflexions- und Emissionsverhalten



## Grundlagen

Anwendung des Bayes-Klassifikators mit Gauss Prozess  
 Schätzung der  $P(o|c)$  ohne Annahme der bedingten Unabhängigkeit  
 Annahme einer  $d$ -dimensionalen Normalverteilung für die  
 Grauwertvektoren einer Klasse



## Methode

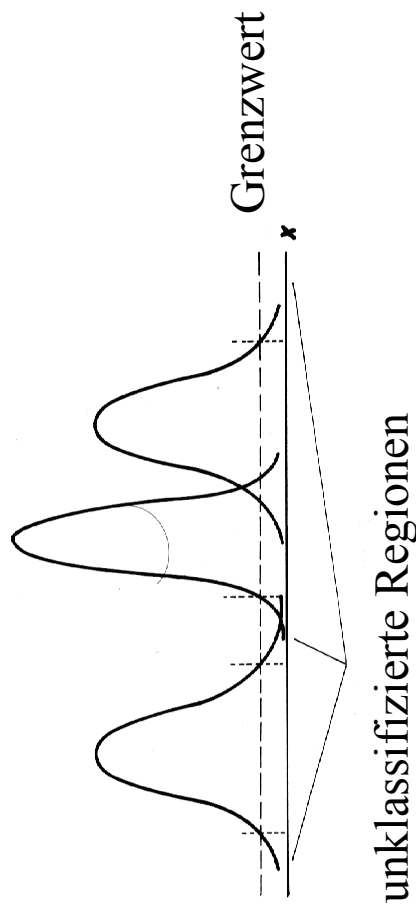
Zu schätzen aus den Trainingsdaten:

$\mu_i$ :  $d$ -dimensionaler Mittelwertvektor aller Feature-Vektoren der Klasse  $c_i$

$\Sigma_i$ :  $d \times d$  Kovarianzmatrix der Klasse  $c_i$

Probleme der Entscheidungsregel:

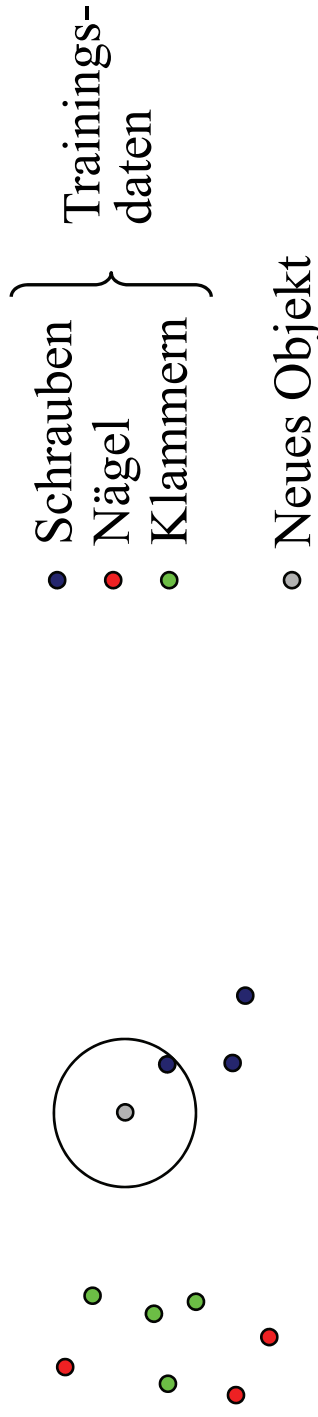
- Wahrscheinlichkeit für die gewählte Klasse sehr klein
- Wahrscheinlichkeit für mehrere Klassen ähnlich



## *Diskussion*

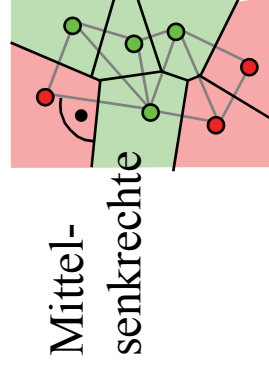
- + hohe Klassifikationsgenauigkeit in vielen Anwendungen
- + Inkrementalität: Klassifikator kann einfach an neue Trainingsobjekte adaptiert werden
- + Einbezug von Anwendungswissen
- Anwendbarkeit (Bayes-Netzwerke): die erforderlichen bedingten Wahrscheinlichkeiten sind oft unbekannt
- Ineffizienz bei sehr vielen Attributen (insbesondere Bayes-Netzwerke)

## 3.4 Nächste-Nachbar- Klassifikatoren

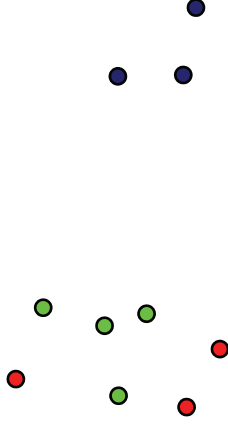


Instanzbasiertes Lernen (*instance based learning*)

- Einfacher Nächste-Nachbar-Klassifikator:  
Zuordnung zu der Klasse des nächsten Nachbarpunkts
- Im Beispiel: Nächster Nachbar ist eine Schraube
- Regionen der Klassenzuordnung können als *Voronoi-Diagramme* dargestellt werden:



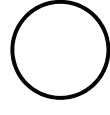
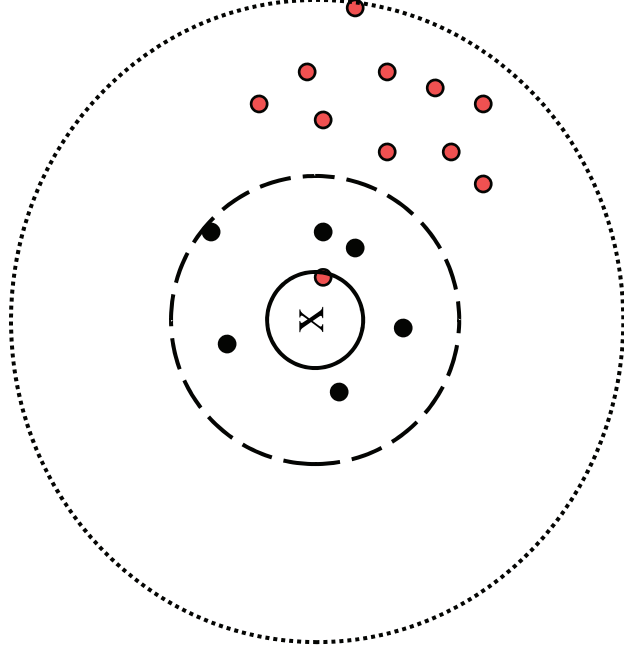




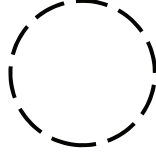
- Problem: Punkt rechts oben wahrscheinlich nur Ausreißer
- Besser: Betrachte mehr als nur einen Nachbarn  
⇒  $k$ -Nächste-Nachbarn-Klassifikator
- *Entscheidungsmenge*  
die Menge der zur Klassifikation betrachteten  $k$ -nächsten Nachbarn
- *Entscheidungsregel*  
Wie bestimmt man aus den Klassen der Entscheidungsmenge die Klasse des zu klassifizierenden Objekts?
  - Interpretiere Häufigkeit einer Klasse in der Entscheidungsmenge als Wahrscheinlichkeit der Klassenzugehörigkeit
  - Maximum-Likelihood-Prinzip: Mehrheitsentscheidung
  - Ggf. Gewichtung

# Wahl des Parameters $k$

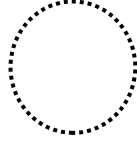
- „zu kleines“  $k$ : hohe Sensitivität gegenüber Ausreißern
- „zu großes“  $k$ : viele Objekte aus anderen Clustern (Klassen) in der Entscheidungsmenge.
- mittleres  $k$ : höchste Klassifikationsgüte, oft  $1 \ll k < 10$



Entscheidungsmenge für  $k = 1$



Entscheidungsmenge für  $k = 7$



Entscheidungsmenge für  $k = 17$

x: zu klassifizieren

# Entscheidungsregel

- Standardregel

⇒ wähle die Mehrheitsklasse der Entscheidungsmenge

- Gewichtete Entscheidungsregel

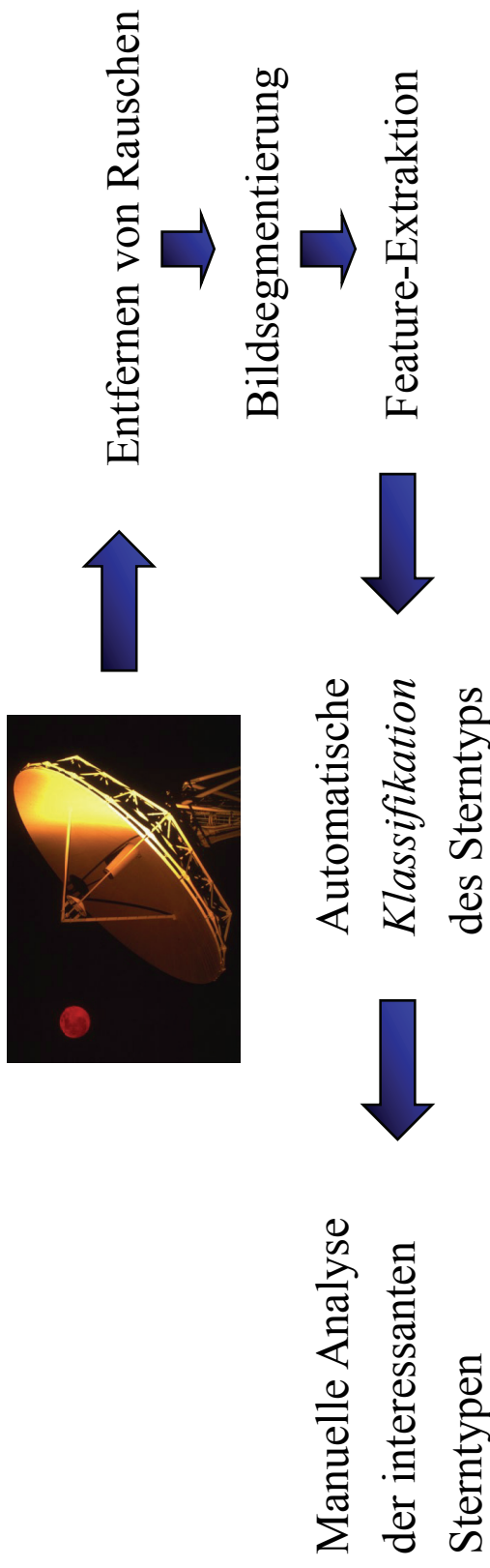
gewichte die Klassen der Entscheidungsmenge

- nach Distanz, meist invers quadriert:  $weight(dist) = 1/dist^2$
- nach Verteilung der Klassen (oft sehr ungleich!)

Problem: Klasse mit zu wenig Instanzen ( $< k/2$ ) in der Trainingsmenge bekommt keine Chance, ausgewählt zu werden, selbst bei optimaler Distanzfunktion

- Klasse A: 95 %, Klasse B 5 %
- Entscheidungsmenge = {A, A, A, A, B, B, B}
- Standardregel ⇒ A, gewichtete Regel ⇒ B

## Analyse astronomischer Daten



Klassifikation des Sterntyps mit Nächste-Nachbarn Klassifikator basierend auf dem Hipparcos-Katalog

## *Hipparcos-Katalog* [ESA 1998]

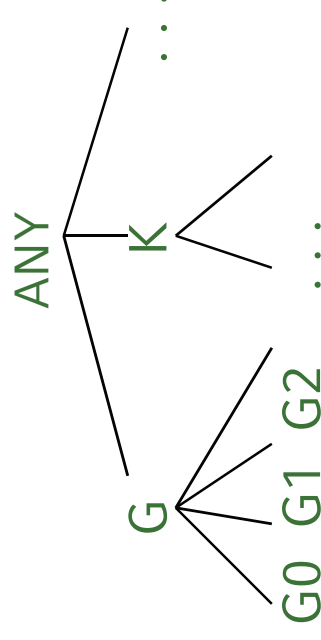
- enthält ca. 118 000 Sterne
- mit 78 Attributen (Helligkeit, Entfernung, Farbe,...)
- Klassenattribut: Spektraltyp (Attribut H76)

z.B.

H76: G0

H76: G7.2

H76: KIII/IV



- Werte des Spektraltyps sind vage  
**➔** Hierarchie von Klassen benutze die erste Ebene der Klassenhierarchie

## Verteilung der Klassen

| Klasse          | #Instanzen | Anteil Instanzen |
|-----------------|------------|------------------|
| K               | 32 036     | 27.0             |
| F               | 25 607     | 21.7             |
| G               | 22 701     | 19.3             |
| A               | 18 704     | 15.8             |
| B               | 10 421     | 8.8              |
| M               | 4 862      | 4.1              |
| häufige Klassen |            |                  |
| O               | 265        | 0.22             |
| C               | 165        | 0.14             |
| R               | 89         | 0.07             |
| W               | 75         | 0.06             |
| N               | 63         | 0.05             |
| S               | 25         | 0.02             |
| D               | 27         | 0.02             |
| seltene Klassen |            |                  |

## *Experimentelle Untersuchung* [Poschenrieder 1998]

- Distanzfunktion
  - mit 6 Attributen (Farbe, Helligkeit und Entfernung)
  - mit 5 Attributen (ohne Entfernung)  
⇒ beste Klassifikationsgenauigkeit mit 6 Attributen
- Anzahl  $k$  der Nachbarn
  - ⇒ beste Klassifikationsgenauigkeit für  $k = 15$
- Entscheidungsregel
  - Gewichtung nach Distanz
  - Gewichtung nach Klassenverteilung  
⇒ beste Klassifikationsgenauigkeit bei Gewichtung nach Distanz  
aber nicht nach Klassenverteilung

# Klassifikation von Sternen

| Klasse       | Falsch klassifiziert | Korrekt klassifiziert | Klassifikationsgenauigkeit |
|--------------|----------------------|-----------------------|----------------------------|
| K            | 408                  | 2338                  | 85.1%                      |
| F            | 350                  | 2110                  | 85.8%                      |
| G            | 784                  | 1405                  | 64.2%                      |
| A            | 312                  | 975                   | 75.8%                      |
| B            | 308                  | 241                   | 43.9%                      |
| M            | 88                   | 349                   | 79.9%                      |
| C            | 4                    | 5                     | 55.6%                      |
| R            | 5                    | 0                     | 0%                         |
| W            | 4                    | 0                     | 0%                         |
| O            | 9                    | 0                     | 0%                         |
| N            | 4                    | 1                     | 20%                        |
| D            | 3                    | 0                     | 0%                         |
| S            | 1                    | 0                     | 0%                         |
| <b>Total</b> | <b>2461</b>          | <b>7529</b>           | <b>75.3%</b>               |



hohe Klassifikationsgenauigkeit für die häufigen Klassen, schlechte Genauigkeit für die seltenen Klassen

die meisten seltenen Klassen besitzen weniger als  $k / 2 = 8$  Instanzen!



## *Diskussion*

- + Anwendbarkeit erfordert als Eingabe nur die Trainingsdaten
- + hohe Klassifikationsgenauigkeit in vielen Anwendungen
- + inkrementell Klassifikator kann sehr einfach an neue Trainingsobjekte adaptiert werden
- + auch zur Vorhersage einsetzbar
- Ineffizienz bei der Auswertung des "Modells" erfordert k-nächste-Nachbarn Anfrage an die Datenbank
- liefert kein explizites Wissen über die Klassen