

4.1 Einleitung

4.2 Clustering

4.3 Klassifikation

Klassifikationsproblem

Gegeben:

- eine Menge $O \subseteq D$ von Objekten $o = (o_1, \dots, o_d) \in O$ mit *Attributen* A_i , $1 \leq i \leq d$
- eine Menge von Klassen $C = \{c_1, \dots, c_k\}$
- Klassenzuordnung $T : O \rightarrow C$

Gesucht:

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

Abgrenzung zum Clustering

- Klassifikation: Klassen a priori bekannt
- Clustering: Klassen werden erst gesucht

Verwandtes Problem: *Regression*

- gesucht ist der Wert für ein numerisches Attribut

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

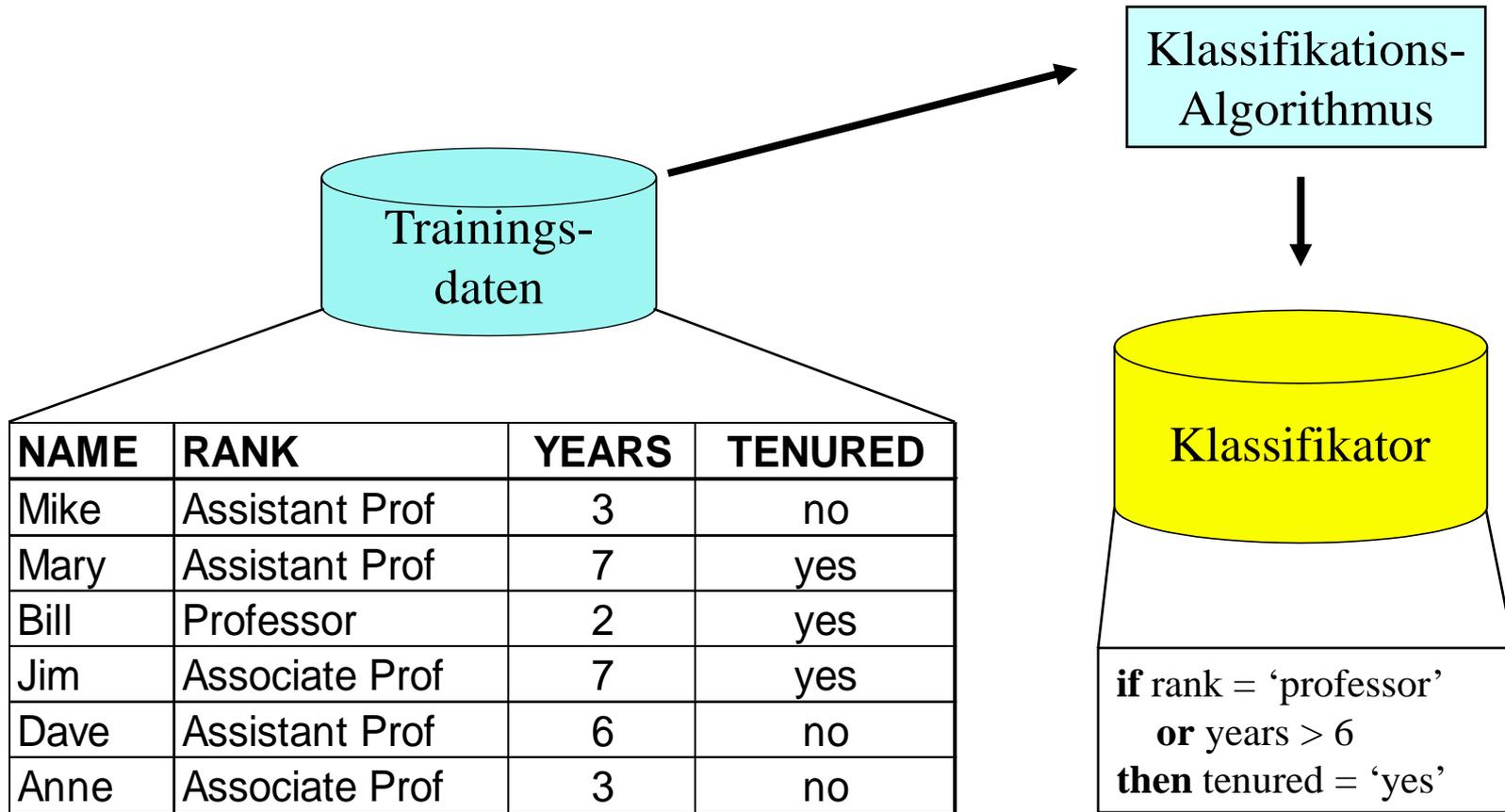
Einfacher Klassifikator

```

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

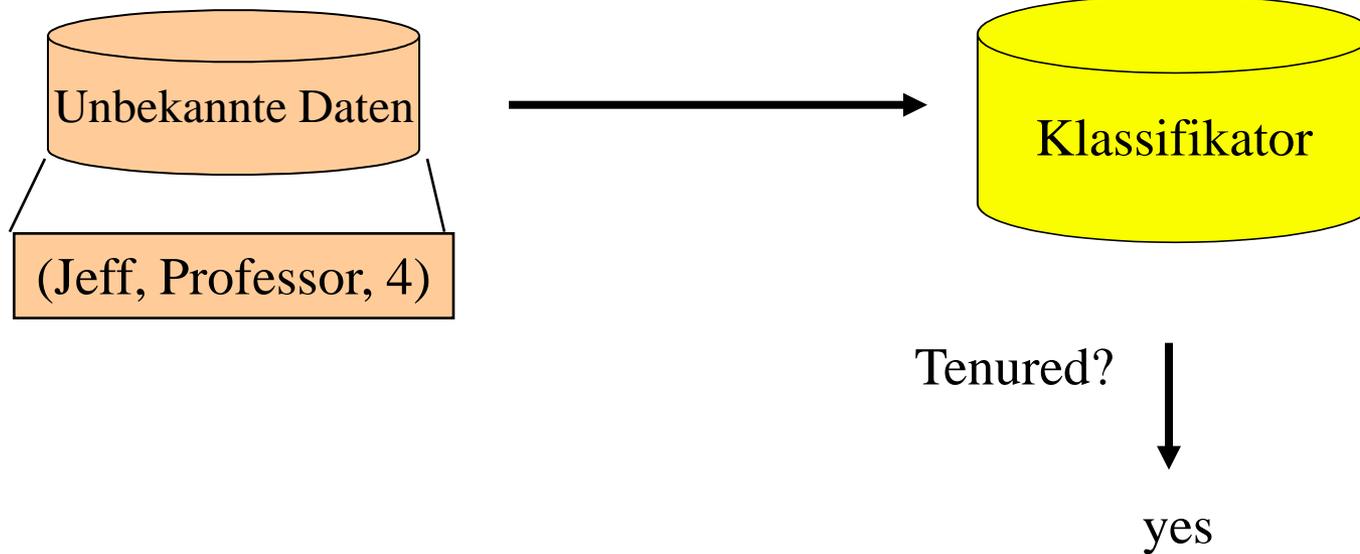
Klassifikations-Prozess

Konstruktion des Modells



Klassifikations-Prozess

Anwendung des Modells



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

Bewertung von Klassifikatoren

Grundbegriffe

- Sei K ein Klassifikator und sei $TR \subseteq O$ die Trainingsmenge. $O \subseteq D$ ist die Menge der Objekte, bei denen die Klassenzugehörigkeit bereits bekannt ist.
- **Problem der Bewertung:**
 - gewünscht ist gute Performanz auf ganz D .
 - Klassifikator ist für TR optimiert.
 - Test auf TR erzeugt in der Regel viel bessere Ergebnisse, als auf $D \setminus TR$.

Daher kein realistisches Bild der Performanz auf D .

⇒ *Overfitting*

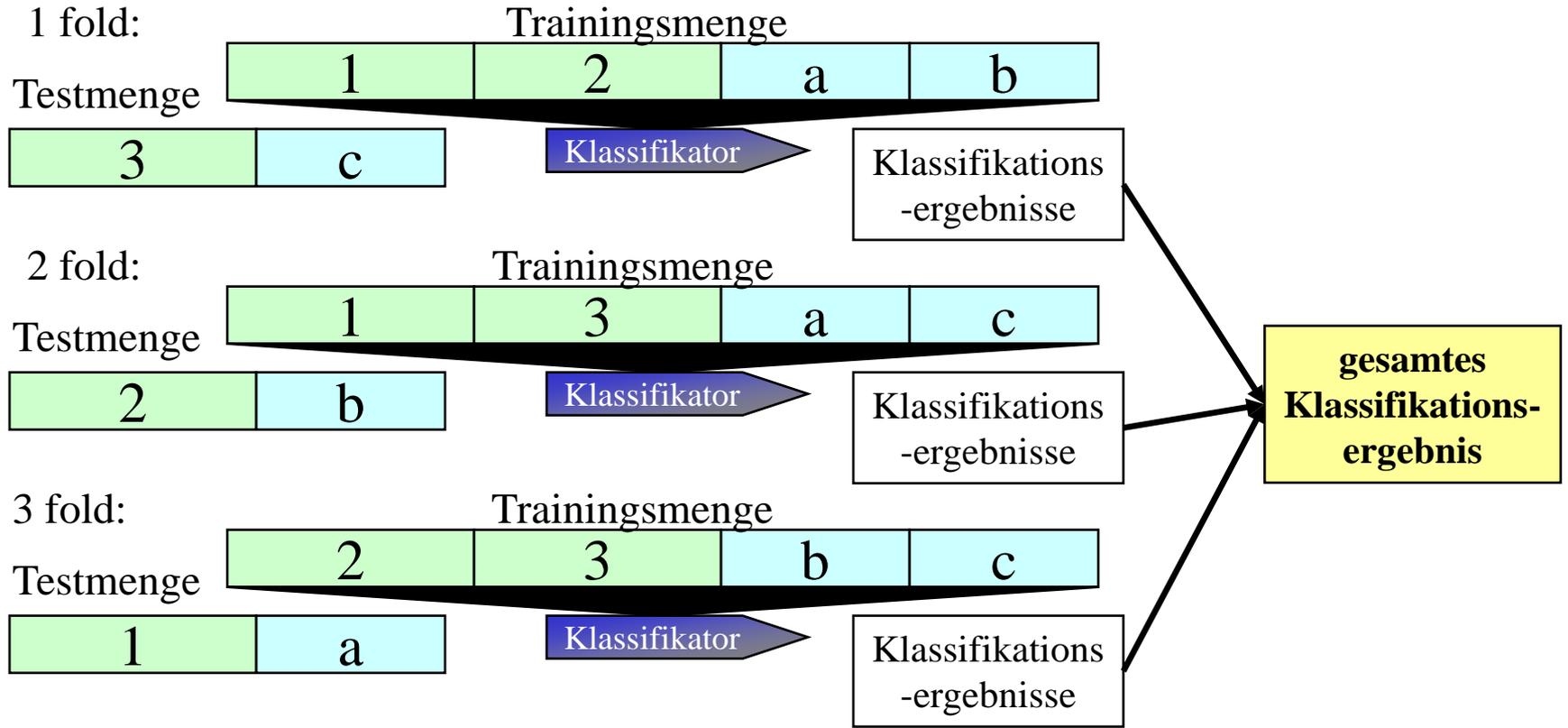
Bewertung von Klassifikatoren

- Abschätzung der Vorhersagequalität auf unbekanntem Daten: k-fache Kreuzvalidierung (k-fold cross-validation)
 - Teile Trainingsmenge $TR \subseteq O$ in k Partitionen TR_1, \dots, TR_k ein.
 - für $i = 1 \dots k$:
 - trainiere einen Klassifikator K_i auf $TR \setminus TR_i$
 - teste K_i auf TR_i
 - Middle die k beobachteten Fehlerraten

Bewertung von Klassifikatoren

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

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



Bewertung von Klassifikatoren

Ergebnis des Tests : Konfusionsmatrix (confusion matrix)

		klassifiziert als ...				
		Klasse 1	Klasse 2	Klasse 3	Klasse 4	other
tatsächliche Klasse ...	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 diverse Kennzahlen berechnen, z.B. Accuracy, Classification Error, Precision und Recall.

Bewertung von Klassifikatoren

• Gütemaße für Klassifikatoren

- Sei K ein Klassifikator, $TR \subseteq O$ die Trainingsmenge, $TE \subseteq O$ die Testmenge. Bezeichne $T(o)$ die tatsächliche Klasse eines Objekts o .
- *Klassifikationsgenauigkeit* (classification accuracy) von K auf TE :

$$G_{TE}(K) = \frac{|\{o \in TE : K(o) = T(o)\}|}{|TE|}$$

- *Tatsächlicher Klassifikationsfehler* (true classification error)

$$F_{TE}(K) = \frac{|\{o \in TE : K(o) \neq T(o)\}|}{|TE|}$$

- *Beobachteter Klassifikationsfehler* (apparent classification error)

$$F_{TR}(K) = \frac{|\{o \in TR : K(o) \neq T(o)\}|}{|TR|}$$

Bewertung von Klassifikatoren

Recall:

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

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

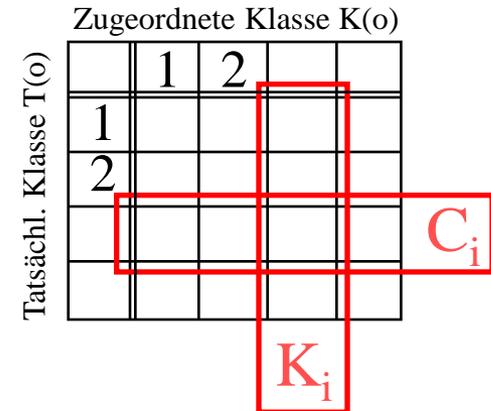
$$\text{Recall}_{TE}(K, i) = \frac{|\{o \in C_i : K(o) = T(o)\}|}{|C_i|}$$

Precision:

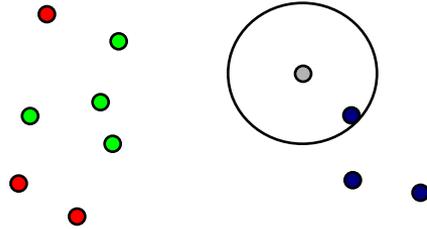
Anteil der zu einer Klasse i zugeordneten Testobjekte, die richtig erkannt wurden.

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

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



Nächste-Nachbarn-Klassifikatoren

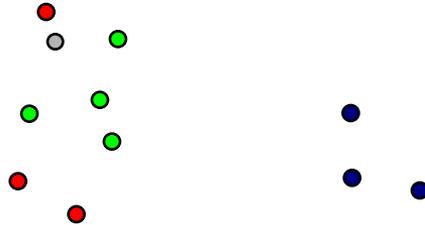


- Schrauben
 - Nägel
 - Klammern
- } Trainingsdaten

○ Neues Objekt => Schraube

- Instanzbasiertes Lernen (*instance based learning*)
- Einfachster Nächste-Nachbar-Klassifikator:
Zuordnung zu der Klasse des nächsten Nachbarpunkts
- Im Beispiel: Nächster Nachbar ist eine Schraube

Nächste-Nachbarn-Klassifikatoren

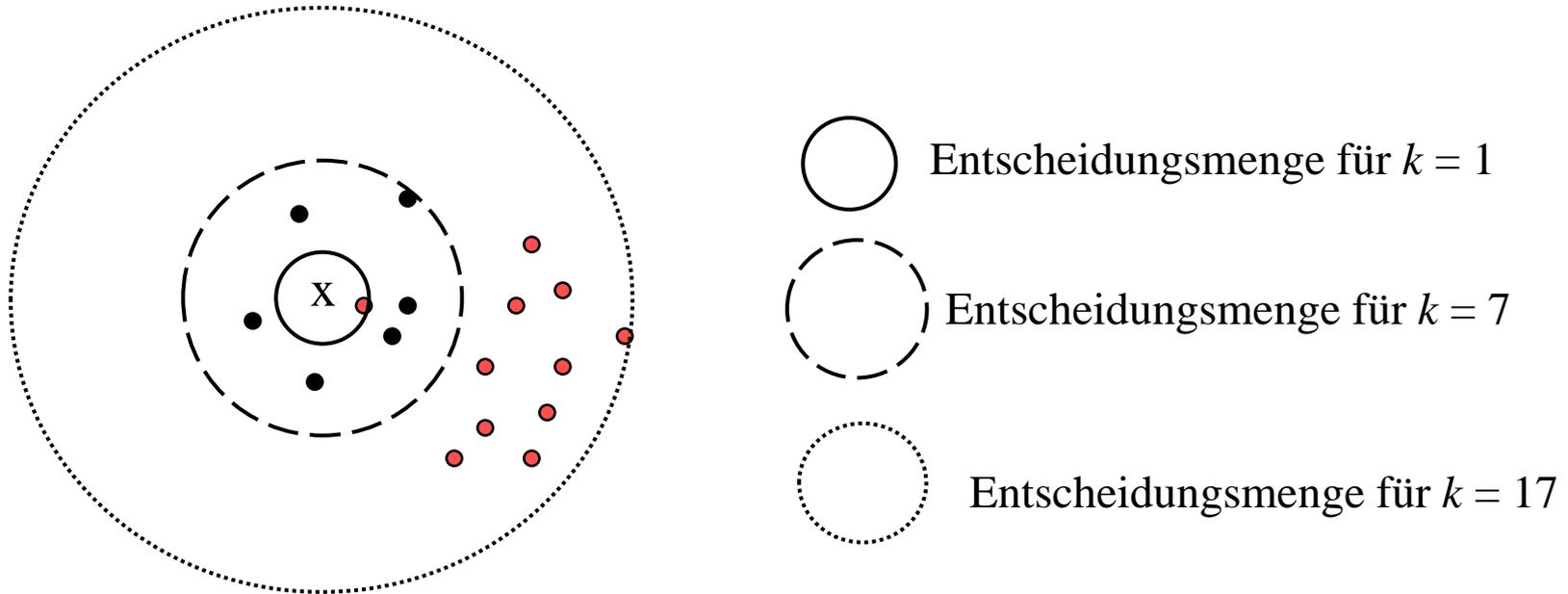


- Problem: Punkt links oben wahrscheinlich nur Ausreißer
=> neues Objekt vermutlich grün statt rot
- Besser: Betrachte mehr als nur einen Nachbarn
→ k -Nächste-Nachbarn-Klassifikator
- *Entscheidungsmenge*
die Menge der zur Klassifikation betrachteten k -nächsten Nachbarn
- *Entscheidungsregel*
wie bestimmt 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

Nächste-Nachbarn-Klassifikatoren

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 $1 \ll k < 10$



x: zu klassifizieren

Nächste-Nachbarn-Klassifikatoren

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 \Rightarrow A, gewichtete Regel \Rightarrow B

Ausblick

Data Mining und andere Wissenschaften

- Data Mining lebt von der Anwendung und muss für viele Anwendungsszenarien und Probleme zugeschnitten werden.
- Data Mining kann im Anwendungsgebiet (z.B. einer anderen Wissenschaft – Geographie, BWL, Kunst, Sprachwissenschaft, Physik, Biologie,...) zu neuen Erkenntnissen führen.
- Umgekehrt bietet ein konkretes Anwendungsszenario oft interessante Herausforderungen für die Forschung im Bereich Data Mining.