Übersicht der Vorlesung

1. Einführung
2. Bildverarbeitung
3. Morphologische Operationen
4. Bildsegmentierung
5. Merkmale von Objekten
6. Klassifikation
7. Dreidimensionale Bildinterpretation
8. Bewegungsanalyse aus Bildfolgen
9. PCA (Hauptkomponentenanalyse)
10. ICA (Independent Component Analysis – Unabhängigkeitsanalyse)

6 Klassifikation

6 Klassifikation

6.1 Prinzipielle Vorgehensweise bei der Klassifikation

6.2 Numerische Klassifikation

6.3 Statistische Klassifikation

6.4 Diskriminanzfunktion

6.5 Support Vektor Maschinen

6.6 Syntaktische Klassifikation

6.7 Kontextabhängige Klassifikation

6.8 Boosting

Klassifikation

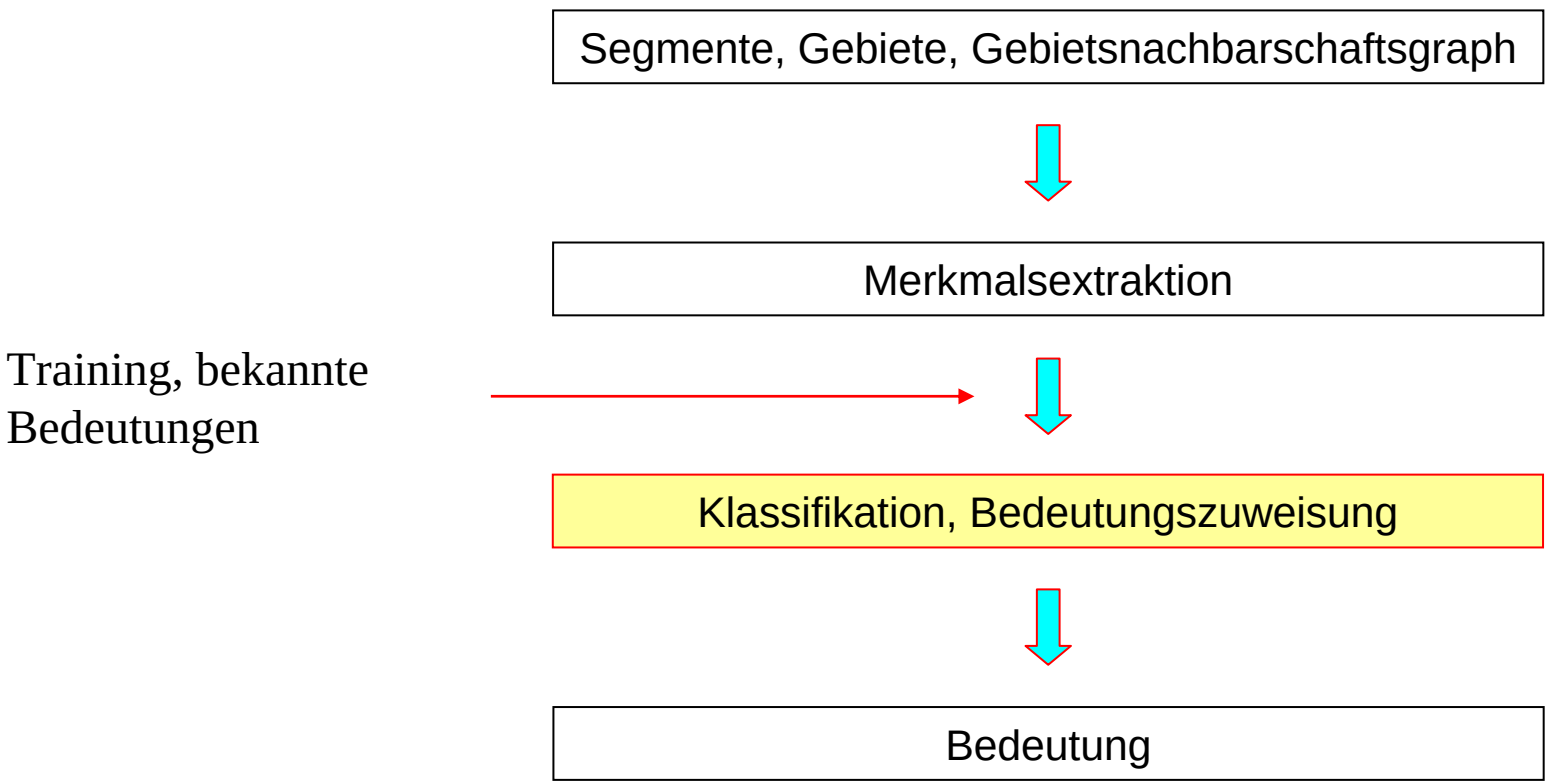
- Ziel der Klassifikation (Mustererkennung) ist es, den durch die Segmentierung gefundenen Gebieten (Objekten), Bedeutungen zuzuweisen.
- Bedeutungen sind z.B.:
  - Quadrat, Kreis
  - Tisch, Stuhl
- Bedeutungen müssen bekannt sein (vorgeben oder lernen)
- es gibt zahlreiche Klassifikationsverfahren
- hier nur Überblick
- nicht vollständig (z.B. keine Neuronalen Netze)
- auf Lernverfahren wird ebenfalls nicht eingegangen

Klassifikation

- kontextunabhängig
  - Klassifikation einzelner Segmente (Objekte)
- kontextabhängig
  - Klassifikation einer Gruppe von Segmenten
  - zwischen den Segmenten bestehen Relationen (z.B. benachbart)
  - Gebietsnachbarschaftsgraph

6.1 Prinzipielle Vorgehensweise bei der Klassifikation

Prinzipielle Vorgehensweise bei der Klassifikation



6.2 Numerische Klassifikation

Numerische Klassifikation

Merkmalsvektor eines Objektes: $\vec{x} = (x_1, x_2, \ldots, x_n) \in R^n$

bekannte Klassen (Bedeutungen): $K_1, K_2, \ldots, K_N$

Jede Klasse besteht aus mehreren Mustern.

Merkmalsvektor des j – ten Musters in der Klasse K_i: $\vec{x}_{ij} = (x_1^{ij}, x_2^{ij}, \ldots, x_n^{ij})$

$i = 1, \ldots, N \quad j = 1, \ldots, N_i$

Klassifikation: $c : \{\vec{x}\} \rightarrow \{K_i : i = 1, \ldots, N\}$

Beispiel

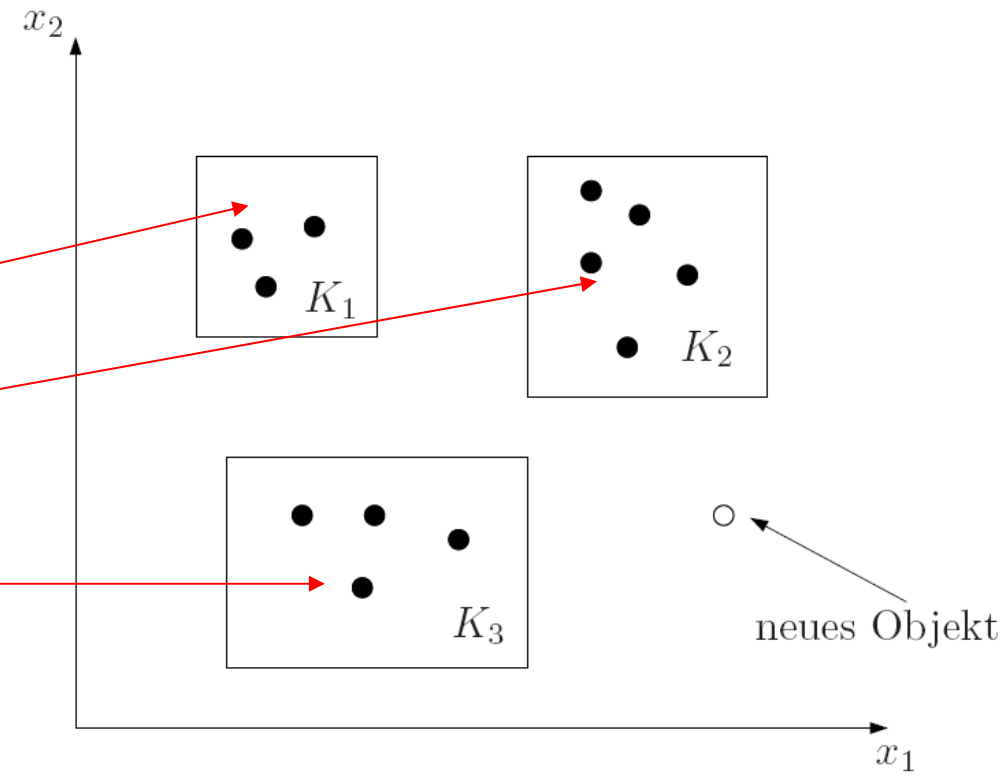
$n = 2$

$N = 3$

$N_1 = 3$

$N_2 = 5$

$N_3 = 4$



Arten der numerischen Klassifikation

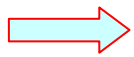
- lineare Klassifikation
- Abstandsklassifikatoren

Entscheidungsfunktion

Entscheidungsfunktionen: $e_i : R^n \to R, \quad i = 1, \ldots, N$

$$e_i(\vec{x}) = w_0^i + w_1^i x_1 + w_2^i x_2 + \cdots + w_n^i x_n, \quad w_j^i \in R, \quad j = 0, \ldots, n$$

Für jede Klasse benötigen wir eine Entscheidungsfunktion.

$e_i(\vec{x}) = 0$  Hyperebene im R^n

Lineare KLassifikation

$$c(\vec{x}) = K_i \quad \Longleftrightarrow$$

$$e_i(\vec{x}) > 0$$

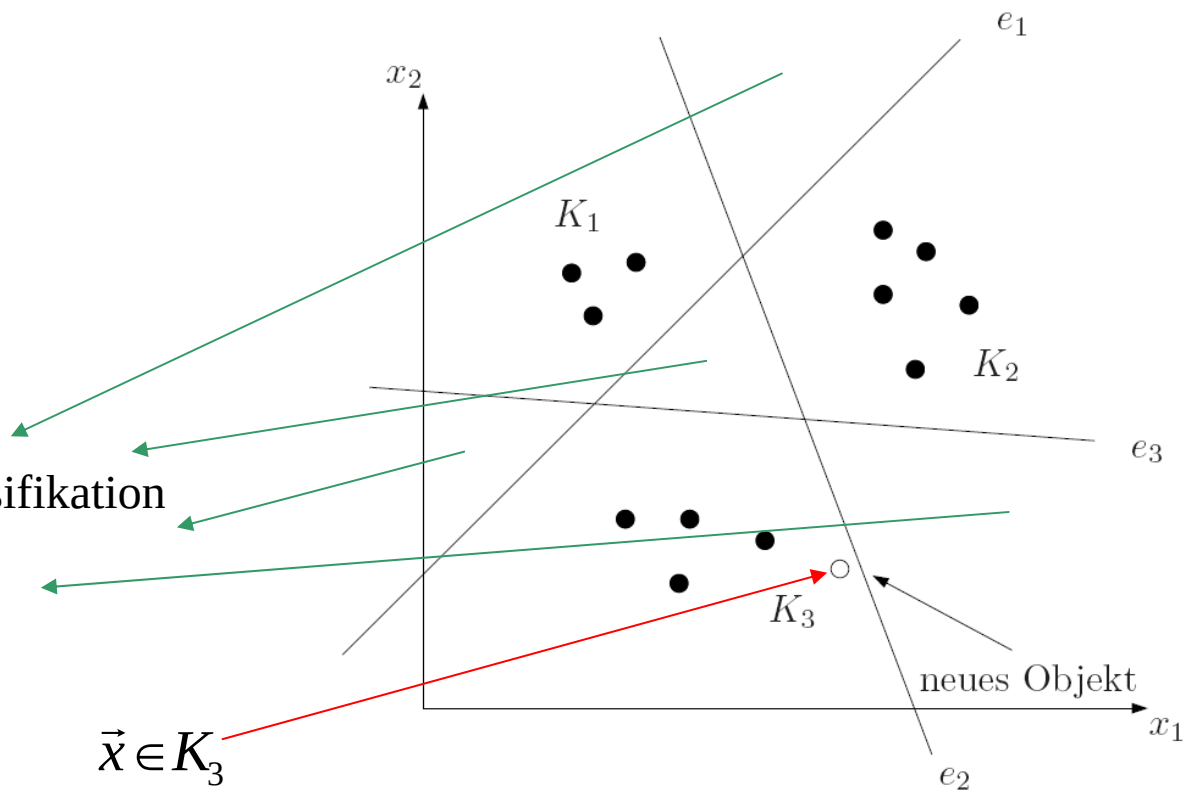
$$e_j(\vec{x}) < 0, \text{ für alle } j = 1, \ldots, N, \; j \neq i$$

Beispiel

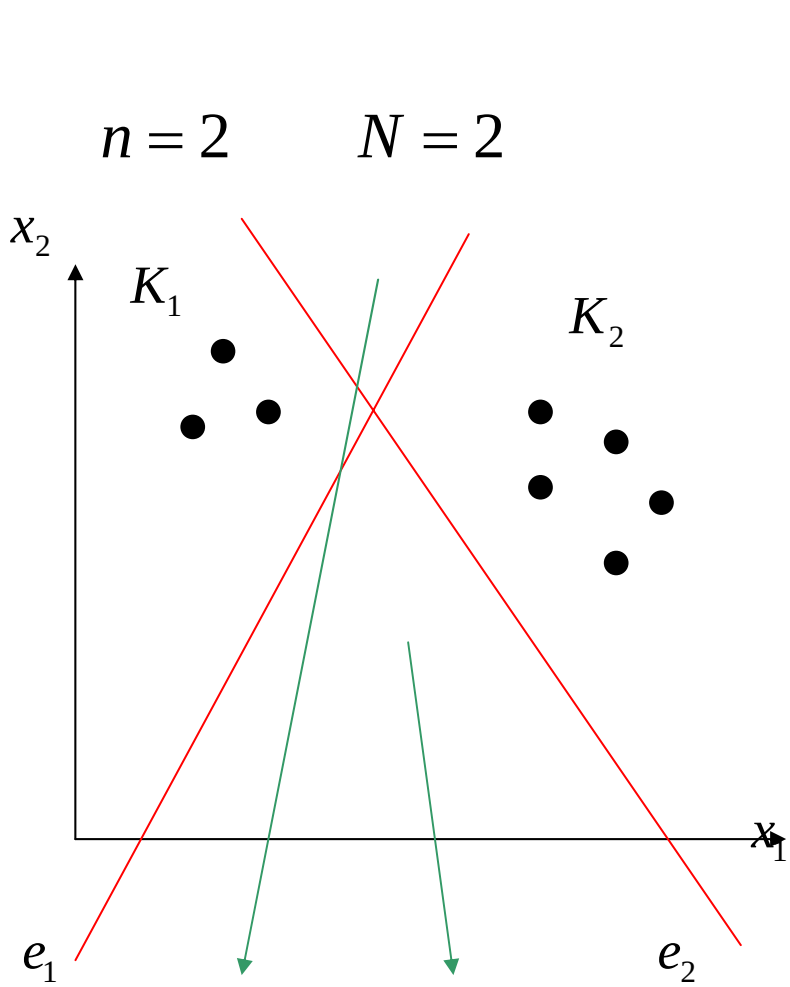
$n = 2$

$N = 3$

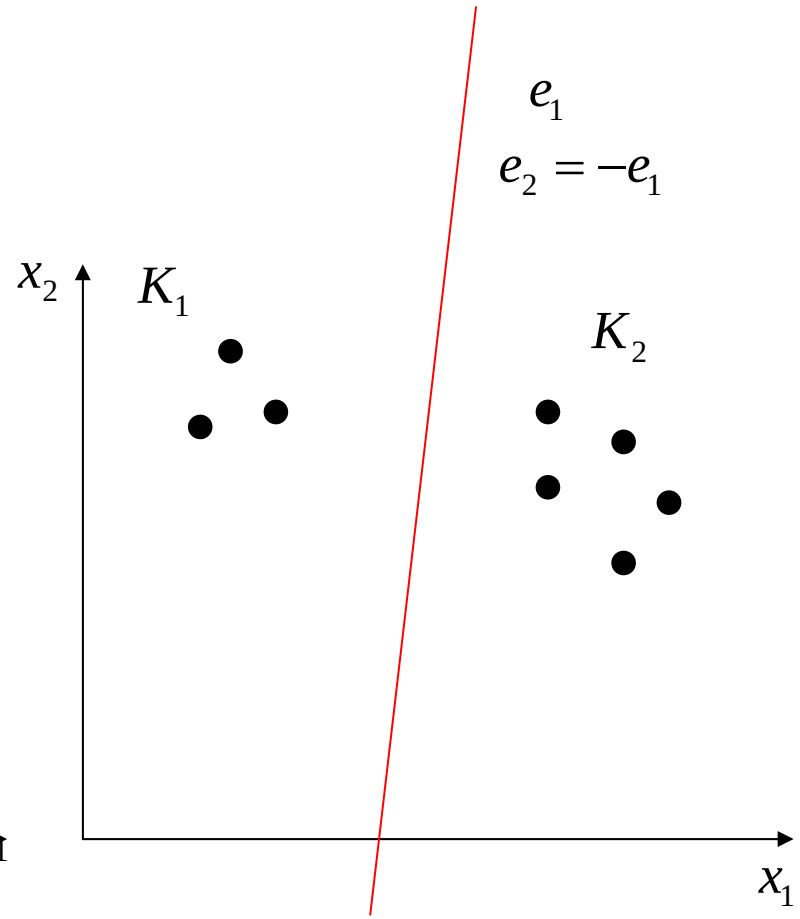
hier ist keine Klassifikation möglich



Beispiel

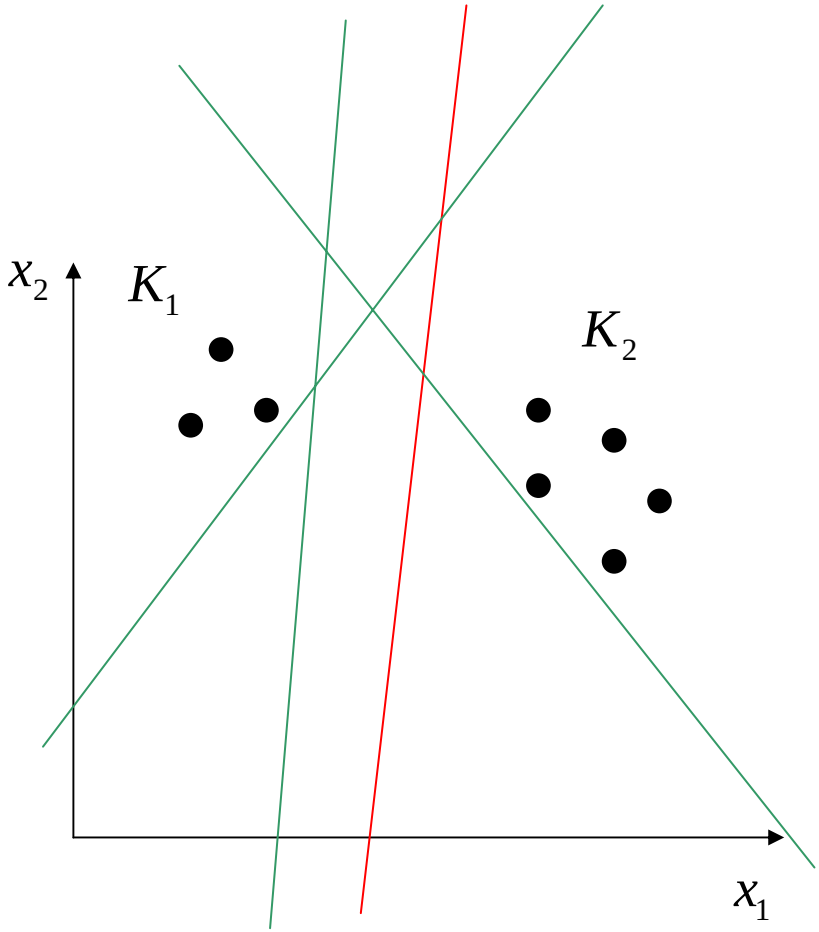


hier ist keine Klassifikation möglich



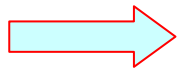
eine trennende Gerade
Klassifikation immer möglich

Beispiel



Problem:

Welches ist die beste trennende Gerade?



Support Vector Machines (6.5)

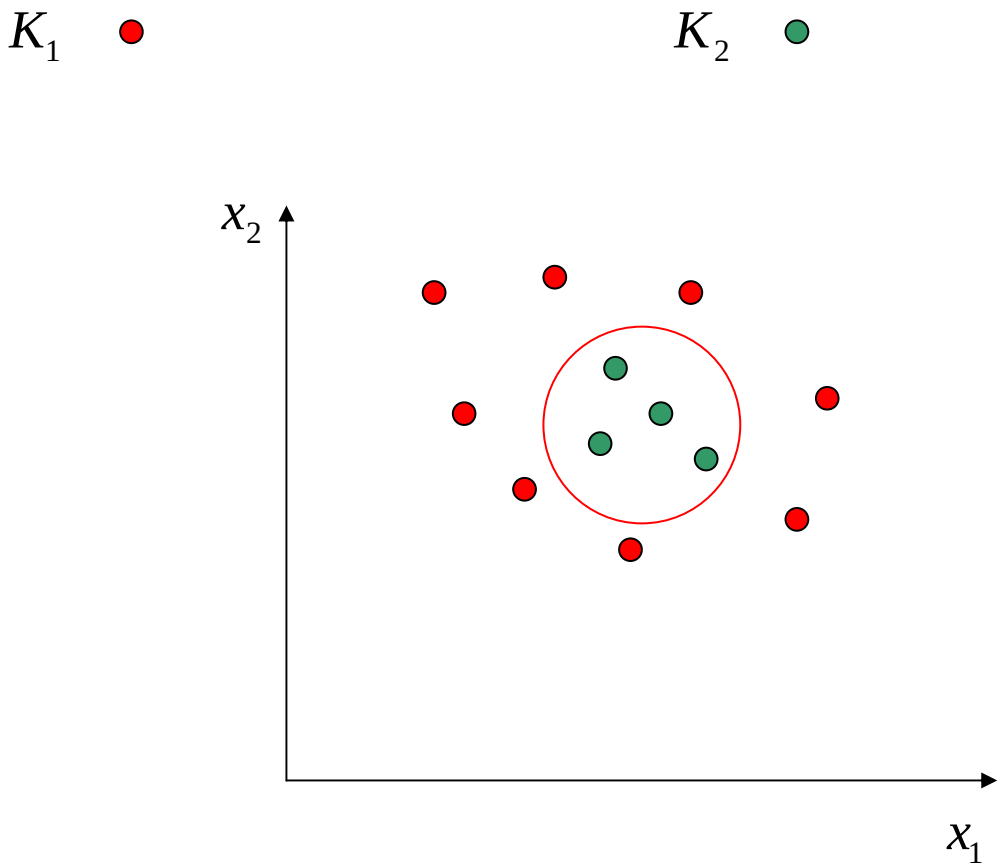
Bemerkungen

$$\vec{x}_{pq} \in K_p \quad \Rightarrow \quad e_i(\vec{x}_{pq}) \begin{cases} > 0 & \text{falls } p = i \\ < 0 & \text{falls } p \neq i \end{cases}$$

$$q = 1, \ldots, N_p$$

- Die Festlegung der Entscheidungsfunktionen geschieht in einer Lernphase und ist nicht immer möglich.
- Anstelle der Hyperebenen können auch kompliziertere Flächen betrachtet werden.

Beispiel – nichtlineare Trennung



Abstandsklassifikatoren

- Hier wird der geometrische Abstand des Objektes zu den vorhandenen Klassen berechnet und das Objekt wird der Klasse mit dem geringsten Abstand zugeordnet.
- Arten:
  - Minimum Distance Klassifikator
  - Nearest Neighbour Klassifikator

Minimum – Distance – Klassifikator

Merkmalsvektor des j – ten Musters in der Klasse K_i:

$$\vec{x}_{ij} = (x_1^{ij}, x_2^{ij}, \ldots, x_n^{ij})$$

$$i = 1, \ldots, N \qquad j = 1, \ldots, N_i$$

$$c(\vec{x}) = K_m \leftrightarrow m = \underset{i=1,\ldots,N}{\operatorname{argmin}} \left\{ \sum_{n=1}^{N} (x_n - f_n^i)^2 \right\}$$

$$\vec{f}_i = \frac{1}{N_i} \sum_{j=1}^{N_i} \vec{x}_{ij} = (f_1^i, \cdots, f_n^i)$$

Mittelwertsvektor der Klasse K_i

Nearest – Neighbour – Klassifikator

$$c(\vec{x}) = K_m \quad \Leftrightarrow$$

$$\exists m' \in \{1, \ldots, N_m\} \text{ mit:}$$

$$\sum_{k=1}^{n} (x_k - x_k^{mm'})^2 = \min_{i=1,\ldots,N \;\; j=1,\ldots,N_i} \left\{ \sum_{k=1}^{n} (x_k - x_k^{ij})^2 \right\}$$

Hier berechnet man den Abstand vom Testmuster zu jedem einzelnen Muster aller Klassen und bestimmt das Muster mit dem geringsten Abstand.
Das Testmuster wird der zugehörigen Klasse zugeordnet.

6.3 Statistische Klassifikation

Wahrscheinlichkeiten

$p(\vec{x} \mid i)$ bedingte Wahrscheinlichkeit, dass ein zur Klasse K_i gehörendes Objekt den Merkmalsvektor $\vec{x}$ liefert

$p(i \mid \vec{x})$ bedingte Wahrscheinlichkeit, dass der Merkmalsvektor $\vec{x}$ zur Klasse K_i gehört

p_i Wahrscheinlichkeit des Auftretens eines Objektes der Klasse K_i

c_{pq} Kosten für den Fall, dass ein Objekt der Klasse K_p fälschlicherweise der Klasse K_q zugeordnet wird

Bayes – Klassifikator

$$c(\vec{x}) = K_m \leftrightarrow m = \underset{k=1,\ldots,N}{\operatorname{argmin}} \left\{ d_k(\vec{x}) \right\}$$

$$d_k(\vec{x}) = \sum_{l=1}^{N} c_{kl} \cdot p_l \cdot p(\vec{x} \mid l)$$

Spezialfall

$$c_{pq} = \begin{cases} 0 \text{ falls } p = q \\ 1 \text{ falls } p \neq q \end{cases}$$

$$\sum_{l=1}^{N} c_{kl} \cdot p_l \cdot p(\vec{x} \mid l) = \sum_{l=1, l \neq k}^{N} p_l \cdot p(\vec{x} \mid l) = \sum_{l=1}^{N} p_l \cdot p(\vec{x} \mid l) - p_k \cdot p(\vec{x} \mid k)$$

Konstante

zu maximieren

dies führt zum Maximum – Likelihood – Klassifikator

Maximum – Likelihood – Klassifikator

$$c(\vec{x}) = K_m \leftrightarrow m = \underset{k=1,\ldots,N}{\operatorname{argmax}} \left\{ p(\vec{x} \mid k) \cdot p_k \right\}$$

Bayes →

$$p(k \mid \vec{x}) = \frac{p_k \cdot p(\vec{x} \mid k)}{p(\vec{x})} = \frac{p_k \cdot p(\vec{x} \mid k)}{\sum_{i=1}^{N} p_i \cdot p(\vec{x} \mid i)}$$

$$c(\vec{x}) = K_m \leftrightarrow m = \underset{k=1,\ldots,N}{\operatorname{argmax}} \left\{ p(k \mid \vec{x}) \right\}$$

Problem: $p(\vec{x} \mid k)$

6.4 Diskriminanzfunktion

Diskriminanzfunktion

$\vec{x} = (x_1, x_2, \ldots, x_n) \in R^n$ $\quad K_1, K_2, \ldots, K_N$

$g_i(\vec{x}) \quad i = 1, \ldots, N$

Klassifikation:

$$c(\vec{x}) = K_m \quad \Leftrightarrow \quad m = \underset{k}{\operatorname{argmax}}\, g_k(\vec{x})$$

Beispiele: $g_i(\vec{x}) = p(i \mid \vec{x})$

$g_i(\vec{x}) = p(\vec{x} \mid i) \cdot p_i$

$g_i(\vec{x} \mid \vec{w}_i, w_{i0}) = \vec{w}_i^{\,T} \cdot \vec{x} + w_{i0} = \sum_{j=1}^{n} w_{ij} \cdot x_j + w_{i0}$ lineare Diskriminanz

2 Klassen

$$g(\vec{x}) = g_1(\vec{x}) - g_2(\vec{x})$$

$$c(\vec{x}) = \begin{cases} K_1 & \text{falls} \quad g(\vec{x}) > 0 \\ K_2 & \text{sonst} \end{cases}$$

linear: $g_1(\vec{x}) = \vec{w}_1^{\,T} \cdot \vec{x} + w_{10} \qquad g_2(\vec{x}) = \vec{w}_2^{\,T} \cdot \vec{x} + w_{20}$

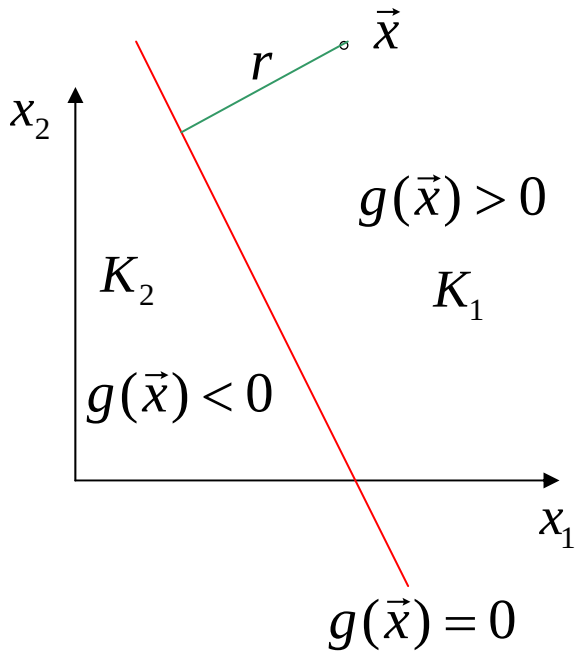
$$g(\vec{x}) = \vec{w}_1^{\,T} \cdot \vec{x} + w_{10} - (\vec{w}_2^{\,T} \cdot \vec{x} + w_{20}) = (\vec{w}_1 - \vec{w}_2)^T \cdot \vec{x} + w_{10} - w_{20}$$

$$g(\vec{x}) = \vec{w}^T \cdot \vec{x} + w_0$$

2 Klassen

$$g(\vec{x}) = \vec{w}^T \cdot \vec{x} + w_0$$

$$c(\vec{x}) = \begin{cases} K_1 & \text{falls} \quad g(\vec{x}) > 0 \\ K_2 & \text{sonst} \end{cases}$$



$$g(\vec{x}^1) = g(\vec{x}^2) = 0 \Rightarrow \vec{w}^T \cdot (\vec{x}^1 - \vec{x}^2) = 0$$

$$r = \frac{|g(\vec{x})|}{\|w\|}$$

$$r_0 = \frac{|g(\vec{0})|}{\|w\|} = \frac{|w_0|}{\|w\|}$$

6.5 Support Vektor Maschinen

Problem

2 Klassen:

bekannte Muster: $X = \left\{ \vec{x}^t, r^t \right\}$ $\vec{x}^t = (x_1^t, x_2^t, \ldots, x_n^t)$

$t = 1,2,\ldots,N$

$$\vec{x}^t \in K_1 \Rightarrow r^t = +1$$

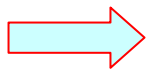
$$\vec{x}^t \in K_2 \Rightarrow r^t = -1$$

suchen Hyperebene (Diskriminante):

$$g(\vec{x}) = \vec{w}^T \cdot \vec{x} + w_0$$

$$\vec{w}^T \cdot \vec{x}^t + w_0 > 0 \quad \text{für} \quad r^t = 1$$

$$\vec{w}^T \cdot \vec{x}^t + w_0 < 0 \quad \text{für} \quad r^t = -1$$



$$r^t (\vec{w}^T \cdot \vec{x}^t + w_0) > 0 \quad \forall r^t$$

Optimale Trennebene

$$g(\vec{x}) = \vec{w}^T \cdot \vec{x} + w_0 \qquad r^t(\vec{w}^T \cdot \vec{x}^t + w_0) > 0 \quad \forall r^t$$

Abstand von x^t zur Hyperebene:

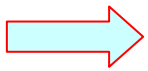
$$\frac{\left|\vec{w}^T \cdot \vec{x}^t + w_0\right|}{\left\|\vec{w}\right\|} = \frac{r^t(\vec{w}^T \cdot \vec{x}^t + w_0)}{\left\|\vec{w}\right\|}$$

Optimale Hyperebene – kleinsten Abstand maximieren:

$$\frac{r^t(\vec{w}^T \cdot \vec{x}^t + w_0)}{\left\|\vec{w}\right\|} \geq \rho \rightarrow \max$$

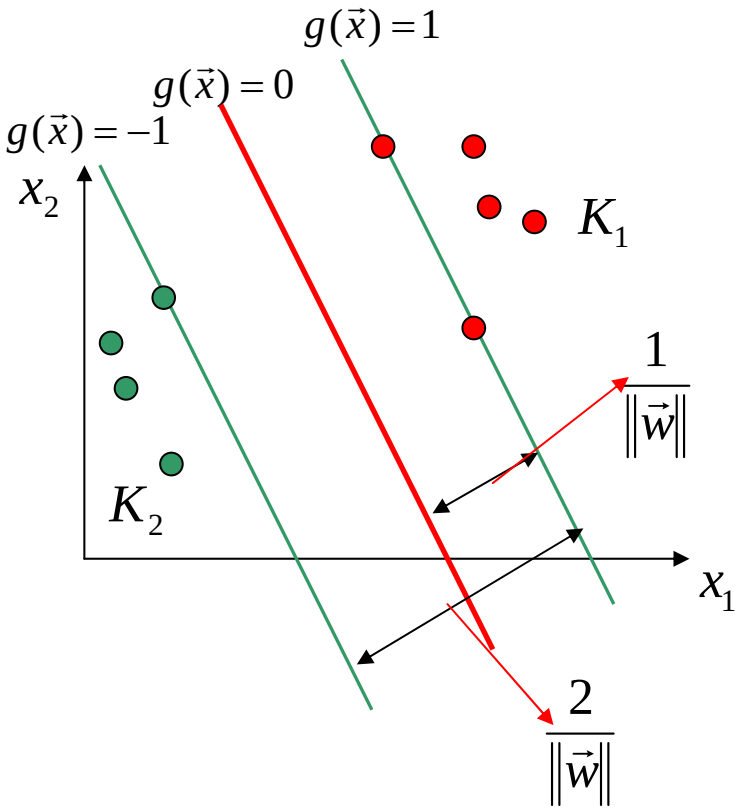
Annahme:

$$\rho \cdot \left\|\vec{w}\right\| = 1$$



$$r^t(\vec{w}^T \cdot \vec{x}^t + w_0) \geq 1$$

Optimale Trennebene



$$\frac{r^t(\vec{w}^T \cdot \vec{x}^t + w_0)}{\|\vec{w}\|} \geq \rho \rightarrow \max$$

$$\rho \cdot \|\vec{w}\| = 1$$

$$\|\vec{w}\| \rightarrow \min$$

$$r^t(\vec{w}^T \cdot \vec{x}^t + w_0) \geq 1$$

Optimale Trennebene

$$\frac{r^t(\vec{w}^T \cdot \vec{x}^t + w_0)}{\|\vec{w}\|} \geq \rho \rightarrow \max$$

$$\rho \cdot \|\vec{w}\| = 1$$

$$r^t(\vec{w}^T \cdot \vec{x}^t + w_0) \geq 1$$

$$\|\vec{w}\| \rightarrow \min$$

Optimierungsproblem:

$$\min \frac{1}{2}\|\vec{w}\|^2$$

Nebenbedingungen:

$$r^t(\vec{w}^T \cdot \vec{x}^t + w_0) \geq 1$$

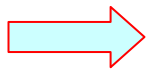
Lösung

Bestimme: α^t $\sum_t \alpha^t \cdot r^t = 0$ $\alpha^t \geq 0$

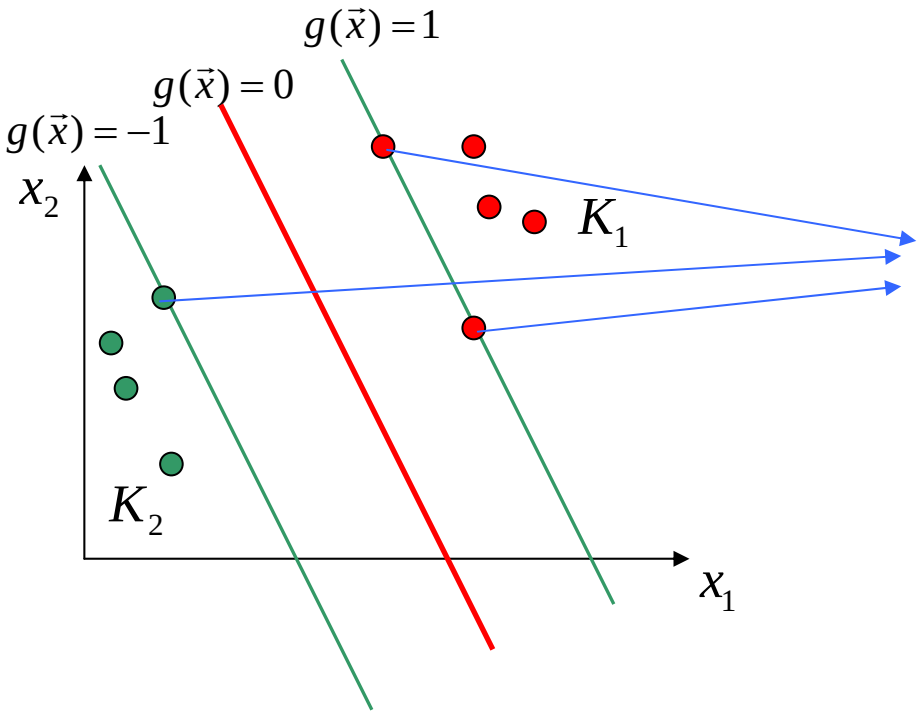
$$-\frac{1}{2}\sum_t \sum_s \alpha^t \alpha^s r^t r^s (\vec{x}^t)^T \cdot \vec{x}^s + \sum_t \alpha^t \rightarrow \max$$

$$\vec{w} = \sum_t \alpha^t r^t \vec{x}^t$$

$t, s = 1,2,\ldots,N$

$\alpha^t > 0 \leftrightarrow r^t(\vec{w}^T \cdot \vec{x}^t + w_0) = 1$  $w_0 = r^t - \vec{w}^T \cdot \vec{x}^t$

Support Vektoren



$$r^t(\vec{w}^T \cdot \vec{x}^t + w_0) = 1$$

Klassifikation

$$g(\vec{x}) = \vec{w}^T \cdot \vec{x} + w_0$$

$$c(\vec{x}) = \begin{cases} K_1 & \text{falls} \quad g(\vec{x}) > 0 \\ K_2 & \text{sonst} \end{cases}$$

$$\vec{w} = \sum_t \alpha^t r^t \vec{x}^t$$

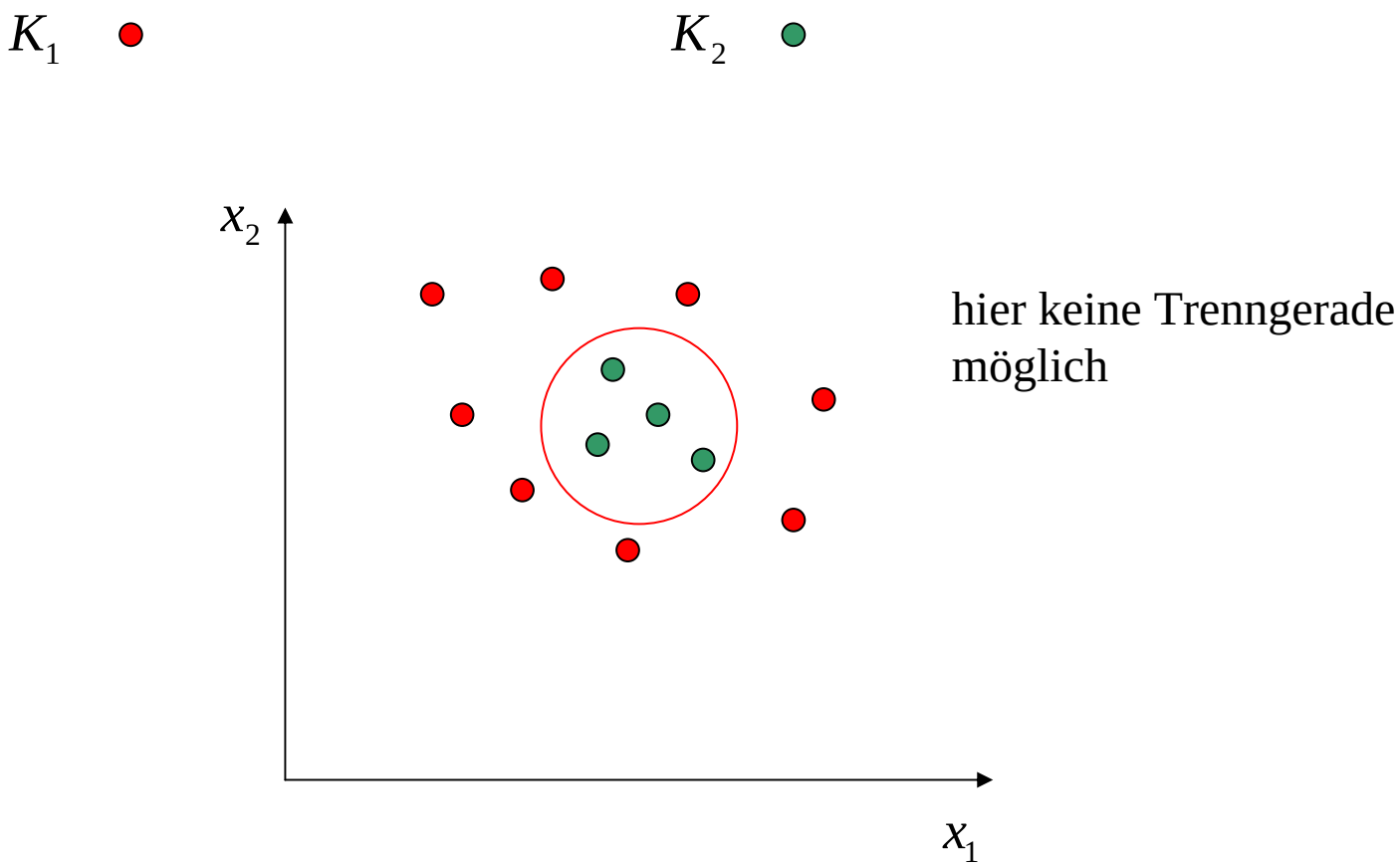
$$w_0 = r^t - \vec{w}^T \cdot \vec{x}^t$$

$$t = 1,2,\ldots,N$$

$$g(\vec{x}) = \sum_t \alpha^t r^t (\vec{x}^t)^T \cdot \vec{x} + w_0$$

Skalarprodukt

Nichtlineare Trennung



hier keine Trenngerade möglich

Kernfunktionen

- Problem in einem neuen Raum abbilden
- durch nichtlineare Transformation
- lineares Modell in diesem neuen Raum benutzen
- Das lineare Modell im neuen Raum entspricht einem nichtlinearen Modell im Originalraum.
- neuer Raum hat eine höhere Dimension

Kernfunktionen

$$\vec{x} \in R^n \quad \Rightarrow \quad \vec{z} = \phi(\vec{x}) \in R^k \qquad z_j = \phi_j(\vec{x})$$

$$k >> n \qquad z_1 = \phi_1(\vec{x}) = 1$$

Diskriminanzfunktion:

$$g(\vec{x}) = \vec{w}^T \cdot \phi(\vec{x}) = \sum_{j=1}^{k} w_j \cdot \phi_j(\vec{x})$$

Lösung:

$$\vec{w} = \sum_t \alpha^t r^t \vec{z}^t = \sum_t \alpha^t r^t \phi(\vec{x}^t)$$

$$g(\vec{x}) = \vec{w}^T \cdot \phi(\vec{x}) = \sum_t \alpha^t r^t \phi(\vec{x}^t)^T \cdot \phi(\vec{x}) = \sum_t \alpha^t r^t K(\vec{x}^t, \vec{x})$$

Skalarprodukt

Kernfunktion

Beispiel

$$g(\vec{x}) = \vec{w}^T \cdot \phi(\vec{x}) = \sum_t \alpha^t r^t \phi(\vec{x}^t)^T \cdot \phi(\vec{x}) = \sum_t \alpha^t r^t K(\vec{x}^t, \vec{x})$$

$$n = 2 \qquad k = 6$$

$$\phi(\vec{x}) = \left(1, \sqrt{2}x_1, \sqrt{2}x_2, \sqrt{2}x_1 x_2, x_1^2, x_2^2\right)^T$$

$$K(\vec{x}, \vec{y}) = (\vec{x}^T \cdot y + 1)^2 \qquad \vec{x} = (x_1, x_2) \qquad \vec{y} = (y_1, y_2)$$

$$K(\vec{x}, \vec{y}) = (\vec{x}^T \cdot y + 1)^2 = (x_1 y_1 + x_2 y_2 + 1)^2$$

$$K(\vec{x}, \vec{y}) = 1 + 2x_1 y_1 + 2x_2 y_2 + 2x_1 y_1 x_2 y_2 + x_1^2 y_1^2 + x_2^2 y_2^2$$

$$K(\vec{x}, \vec{y}) = \phi(\vec{x})^T \cdot \phi(\vec{y})$$

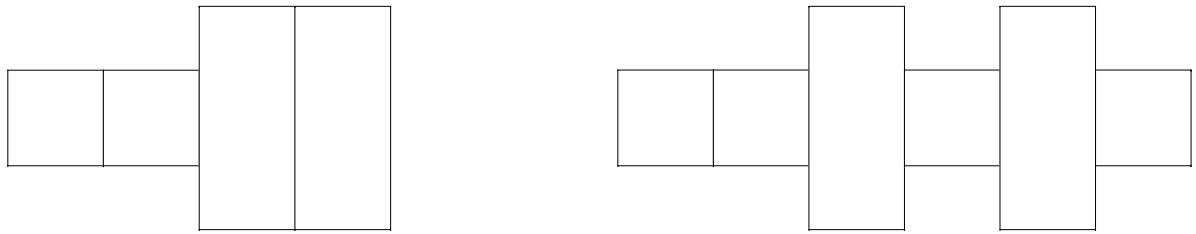
6.6 Syntaktische Klassifikation

Syntaktische Klassifikation

- Objekte werden durch eine Aneinanderreihung von Objektteilen beschrieben.
- Art der Objektteile, als auch die Art der Aneinanderreihung kann für die Bedeutung des Objektes relevant sein.

Objektteile:

Objekte:



Grammatik

$$G = (V_N, V_T, R, S)$$

V	endliche Menge von Symbolen (Alphabet)		
V_N	nichtterminale Symbole	$V_N \subseteq V$	$V_N \cup V_T = V$
V_T	terminale Symbole	$V_T \subseteq V$	$V_N \cap V_T = \varnothing$
S	Startsymbol	$S \in V_N$	
V^*	Menge aller Symbolketten über V		
L	Sprache	$L \subseteq V^*$	
ε	leere Symbolkette		$V^+ = V^* \setminus \{\varepsilon\}$

Grammatik

$$G = (V_N, V_T, R, S)$$

R endliche Menge von Regeln

$$R \subseteq (V^+ \setminus V_T^*) \times V^*$$

$$r \in R \qquad r = (\varphi, \psi) \qquad r : \varphi \rightarrow \psi \qquad \varphi \neq \varepsilon$$

$$\exists r = (S, \psi) \in R$$

L(G) sei die Menge aller Symbolketten, die aus *S* durch Anwendung der Regeln aus *R* abgeleitet werden können.

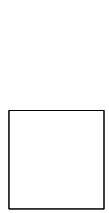
Beispiel

$$G_1 = \left(\{S,T\},\{a,b\},R,S\right)$$

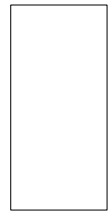
Regeln:

$$S \rightarrow aTa$$
$$T \rightarrow aT$$
$$T \rightarrow Ta$$
$$T \rightarrow bT$$
$$T \rightarrow Tb$$
$$T \rightarrow a$$
$$T \rightarrow b$$

a und b entsprechen den beiden Objektarten „Art a“ und „Art b“



Art a



Art b

Klassifikation – 1

Klassen: $K_1, K_2, \ldots, K_N$

Grammatiken: $G_1, G_2, \ldots, G_N$

Klassifikation: $v \in V^*$

$$c(v) = K_m \quad \Longleftrightarrow \quad v \in L(G_m)$$

Klassifikation – 2

Klassen: $K_1, K_2, \ldots, K_N$

bekannte Bedeutungen: $v_1, v_2, \ldots, v_N$ $\quad v_i \in V^*$

Klassifikation: $v \in V^*$

$$c(v) = K_m = v_m \quad \Longleftrightarrow \quad m = \underset{k=1,\ldots,N}{\operatorname{argmin}} d_L(v, v_k)$$

Levenshtein – Abstand

Levenshtein – Abstand

$x, y \in V^*$

$d_L(x, y)$

Minimale Anzahl von Operationen, die notwendig ist, um *x* in *y* zu überführen.

Operationen:

$\alpha, \beta \in V^*$

$a, b \in V$

$\alpha a \beta \rightarrow \alpha b \beta$ Ersetzen

$\alpha a \beta \rightarrow \alpha \beta$ Löschen

$\alpha \beta \rightarrow \alpha a \beta$ Einfügen

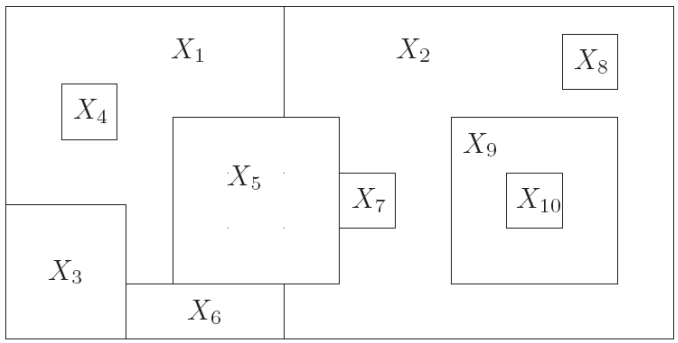
Anmerkung

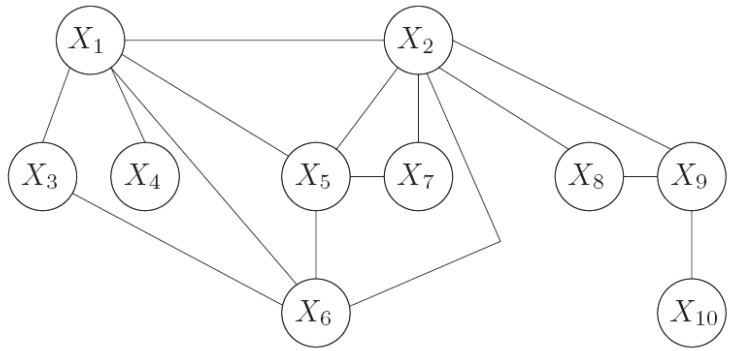
- Numerische und syntaktische Klassifikation lassen sich kombinieren mit **attributierten Grammatiken**.
- Jedem Objekt $v \in V$ wird eine Menge von Attributen zugeordnet.
- Jede Regel $r \in R$ wird erweitert durch eine Beschreibung, wie bei Anwendung der Regel die Attribute der an der Regel beteiligten Objekte verändert werden.

6.7 Kontextabhängige Klassifikation

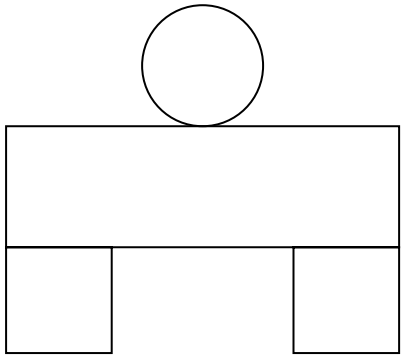
Kontextabhängige Klassifikation

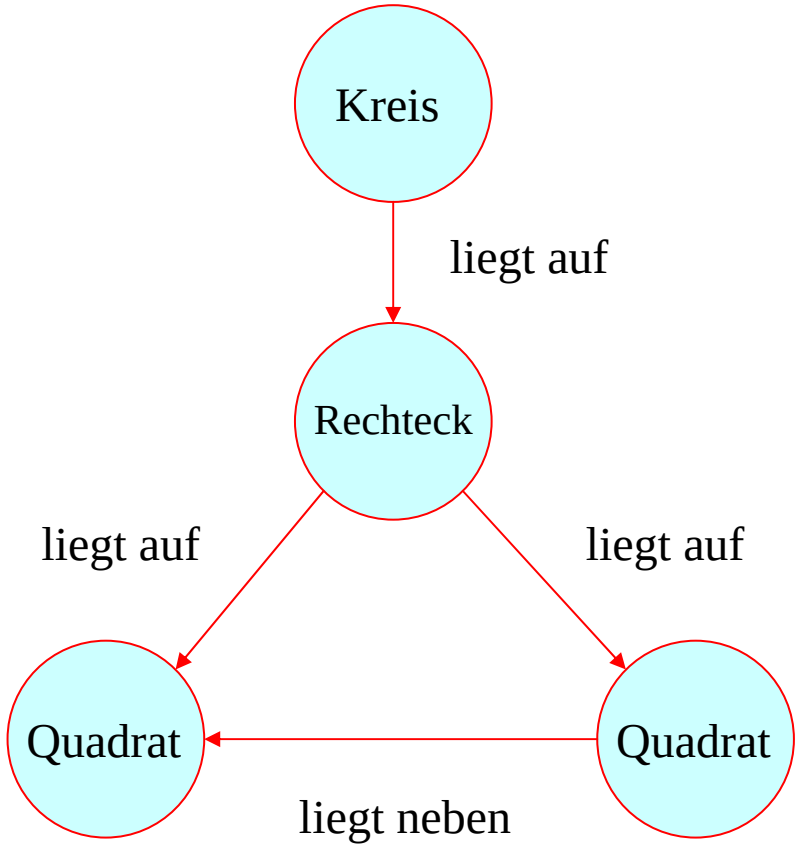
- Ein Objekt wird nicht für sich allein klassifiziert, sondern im Kontext mit einem oder mehreren anderen Objekten.
- Klassifikation von Strukturen (Graphen)
- Knoten des Graphen sind die Objekte
- Kanten beschreiben Relationen zwischen den Objekten, z.B. die Nachbarschaftsrelation (Gebietsnachbarschaftsgraph)





Beispiel




Kreis
liegt auf
Rechteck
liegt auf
liegt auf
Quadrat
Quadrat
liegt neben

2 Verfahren

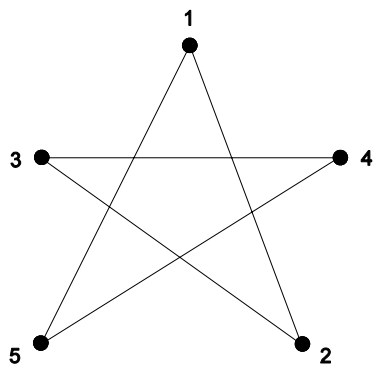
- Relaxation
  - Die Vielfalt der möglichen Graphen ist groß.
  - Durch ein iteratives Verfahren wird versucht, die Objekte (Knoten) des Graphen so zu klassifizieren, dass die vorgegebenen Relationen erfüllt werden.
  - Zu Beginn werden den Objekten mehrere Bedeutungen zugewiesen, die auch mit Wahrscheinlichkeiten versehen werden können.
  - diskrete Relaxation
  - kontinuierliche Relaxation
- Vergleich von Graphen
  - Die Graphen der bekannten Objekte sind gegeben und deren Anzahl klein.
  - Ein zu klassifizierender Graph wird dann einem gegebenen Graphen (Bedeutung) zugeordnet.
  - Ähnlichkeit von Graphen: minimale Anzahl von Modifikationen (Einfügen, Löschen, von Knoten und Kanten)
  - Graphmatching

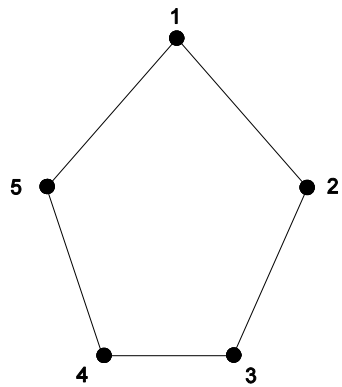
6.7.1 Graphmatching

Isomorphie

Gegeben seien 2 Graphen $G_1 = (V_1,K_1)$ und $G_2 = (V_2,K_2)$. G_1 und G_2 heißen **isomorph**, wenn eine eineindeutige Abbildung $f : V_1 \rightarrow V_2$ existiert mit:

$$\forall v,w \in V_1 : (v,w) \in K_1 \Leftrightarrow (f(v),f(w)) \in K_2$$





2 isomorphe Graphen

Graphmatching

- Folgende Aufgaben sind von Interesse:
  - Sind G_1 und G_2 isomorph ? G_1 ist zu klassifizieren und G_2 ist eine bekannte, gegebene Bedeutung
  - Finde einem (oder alle) Teilgraphen von G_2, die isomorph zu G_1 sind. Hier ist der zu klassifizierende Graph G_1 nicht vollständig gegeben, z.B. teilweise verdeckt.
  - Finde isomorphe Teilgraphen von G_1 und G_2.
- Graphmatching ist aber recht schwierig zu behandeln.

6.7.2 Diskrete Relaxation

Diskrete Relaxation

Segmentierung: $Z_S = \{X_1, \ldots, X_m\}$

Gebietsnachbarschaftsgraph: $G_N = [Z_S, R_N]$

zu klassifizierende Objekte: X_i

Klassen (Bedeutungen): $B = \{K_1, \ldots, K_N\}$

$B_i \subseteq B$ Menge der möglichen Bedeutungen für das Objekt X_i $\quad B = \bigcup_{i=1}^{m} B_i$

$$B_{ij} = \{(b_1, b_2) : b_1 \in B_i, b_2 \in B_j\} \subseteq B_i \times B_j$$

Bedeutungen sind kompatibel auf Grund der Relation $(X_i, X_j) \in R_N$

Ziel: iteratives Auffinden einer Zuordnung zwischen Objekten und Bedeutungen unter Beachtung der Kanten R_N

Diskrete Relaxation

Anfangszuordnung: $B^{(0)} = (B_1^{(0)}, B_2^{(0)}, \ldots, B_m^{(0)})$ $B_i^{(0)} \subseteq B_i$

nach dem *r*-ten Iterationsschritt:

$B^{(r)} = (B_1^{(r)}, B_2^{(r)}, \ldots, B_m^{(r)})$ $B_i^{(r)} \subseteq B_i$

Iteration: $B_i^{(r)} \rightarrow B_i^{(r+1)} \quad i = 1, \ldots, m$

Iteration

$$B_i^{(r)} \subseteq B_i, \quad i = 1, \ldots, m$$

$$\exists X_j \text{ mit: } (X_i, X_j) \in R_N \text{ und}$$

$$\exists b_1 \in B_i^{(r)} \text{ mit: } \forall b_2 \in B_j^{(r)} \text{ gilt } (b_1, b_2) \notin B_{ij}$$

$$B_i^{(r+1)} = B_i^{(r)} \setminus \{b_1\}$$

sonst: $B_i^{(r+1)} = B_i^{(r)}$

Iterationsende: $\exists l \text{ mit: } B_i^{(l+1)} = B_i^{(l)}, \ \forall i = 1, \ldots, m$

6.7.3 Kontinuierliche Relaxation

Kontinuierliche Relaxation

- Jedem Objekt (Bildpunkt, Knoten eines Gebietsnachbarschaftsgraphen) wird eine Wahrscheinlichkeit für die Zugehörigkeit zu jeder der in Frage kommenden Klassen (Bedeutungen) zugeordnet – Iterationsbeginn
- Iterationsschritt – Wahrscheinlichkeiten der Nachbarobjekte werden daraufhin überprüft, ob ihre Klassifizierung mit dem betrachteten Objekt X_i kompatibel ist oder nicht
  - Kompatibel → die Wahrscheinlichkeit der aktuellen Klassenzuweisung von X_i wird erhöht
  - Inkompatibel → die Wahrscheinlichkeit der aktuellen Klassenzuweisung von X_i wird verringert

Kontinuierliche Relaxation

zu klassifizierende Objekte: $X_i \qquad i = 1, \ldots, m$

Klassen (Bedeutungen): $K_1, \ldots, K_N$

Zuordnung benachbarter Objekte:

$$X_i \rightarrow K_j \text{ und } X_h \rightarrow K_l \quad (X_i, X_h) \in R_N$$

$$c(i, j, h, l) \in [-1, +1]$$ positiv - kompatibel

$$c(i, j, h, l) = 2\frac{p(X_i \rightarrow K_j \mid X_h \rightarrow K_l)}{p(X_i \rightarrow K_j)} - 1 = 2\frac{p(X_i \rightarrow K_j, X_h \rightarrow K_l)}{p(X_i \rightarrow K_j)\, p(X_h \rightarrow K_l)} - 1$$

Iteration

Initialisierung: $X_i \rightarrow K_j$ ⇒ $p_{ij}^{(0)}$ mit $0 \le p_{ij}^{(0)} \le 1$

$$\sum_{j=1}^{N} p_{ij}^{(0)} = 1 \quad \forall i = 1, \ldots, m$$

Wahrscheinlichkeiten

Iteration: $p_{ij}^{(r)}$ mit $0 \le p_{ij}^{(r)} \le 1$ ⇒ $p_{ij}^{(r+1)}$ mit $0 \le p_{ij}^{(r+1)} \le 1$

$$p_{ij}^{(r+1)} = \frac{p_{ij}^{(r)}(1+q_{ij}^{(r)})}{\sum_{l=1}^{N} p_{il}^{(r)}(1+q_{il}^{(r)})}$$

$$q_{ij}^{(r)} = \prod_{h=1,(X_h,X_i)\in R_N}^{m} \sum_{l=1}^{N} c(i,j,h,l)\, p_{hl}^{(r)}$$

Iteration

$p_{hl}^{(r)}$ groß $c(i, j, h, l) >> 0$ ⇒ $p_{ij}^{(r)}$ vergrößern

$p_{hl}^{(r)}$ groß $c(i, j, h, l) << 0$ ⇒ $p_{ij}^{(r)}$ vermindern

$p_{hl}^{(r)}$ klein

$c(i, j, h, l) \approx 0$ ⇒ $p_{ij}^{(r)}$ geringfügig ändern

6.8 Boosting

Verstärken

Boosting

- Algorithmus der Klassifizierung, der mehrere schlechte Klassifikatoren zu einem einzigen guten Klassifikator verschmilzt
- 2 Klassen
- vorgegeben ist eine Menge von Objekten und eine Menge schwacher Klassifikatoren
- gesucht ist ein Klassifikator, der die Objekte möglichst fehlerfrei in zwei Klassen einteilt

Schwache Klassifikatoren

- sehr einfach aufgebaut
- berücksichtigen oft nur ein einziges Merkmal der Objekte
- liefern deswegen schlechte Ergebnisse
- können aber sehr schnell ausgewertet werden

Schwache Klassifikatoren – Beispiel (decision stumps)

$$f : \{\vec{x}\} \rightarrow \{+1, -1\} \qquad \vec{x} = (x_1, x_2, \ldots, x_n)$$

$$f(\vec{x}) = \begin{cases} +1 & \text{falls } x_j \geq l \\ -1 & \text{falls } x_j < l \end{cases}$$

Bezeichnungen

2 Klassen:

bekannte Muster: $X = \left\{ \vec{x}^t, r^t \right\}$ $\vec{x}^t = (x_1^t, x_2^t, \ldots, x_n^t)$

$t = 1, \ldots, N$

$$\vec{x}^t \in K_1 \Rightarrow r^t = +1$$

$$\vec{x}^t \in K_2 \Rightarrow r^t = -1$$

Schwache Klassifikatoren:

$$f_1, \ldots, f_m : \{\vec{x}\} \rightarrow \{+1, -1\}$$

Gesucht: $F : \{\vec{x}\} \rightarrow \{+1, -1\}$

$$F(\vec{x}) = \operatorname{sgn}\left(\sum_{i=1}^{m} w_i f_i(\vec{x}) \right)$$

soll möglichst wenig Fehler machen

Lösung

$$F(\vec{x}) = \operatorname{sgn}\left(\sum_{i=1}^{m} w_i f_i(\vec{x})\right)$$

$$L = \frac{1}{N}\sum_{t=1}^{N} e^{-r^t \cdot F(\vec{x}^t)} \rightarrow \min$$

wird umso kleiner, je weniger Objekte durch *F* falsch klassifiziert werden

Die Optimierung wird schrittweise über *m* ausgeführt.

Sie kann vorher abgebrochen werden.

Schrittweise Optimierung

konstruieren: $F_s : \{\vec{x}\} \rightarrow \{+1,-1\}$ $\quad s = 1,\ldots,m$

f_s wird hinzugenommen

Hilfsvariablen: $t_{s,1},\ldots,t_{s,N}$ $\quad t_{1,1} = t_{1,2} = \cdots = t_{1,N} = \frac{1}{N}$

bewerten die bekannten Muster (je höher der Wert, desto stärker geht das Muster in den aktuellen Durchgang ein)

$$t_{s,i} = \frac{t_{s-1,i} \cdot e^{r^i w_{s-1} f_{s-1}(\vec{x}^i)}}{\eta} \qquad s = 2,\ldots,m \qquad \sum_i t_{s,i} = 1$$

Schrittweise Optimierung

$$e_s = \sum_{i: f_s(\vec{x}^i) \neq r^i} t_{s,i}$$

$$w_s = \frac{1}{2} \log\left(\frac{1-\mathrm{e}_s}{\mathrm{e}_s}\right)$$

$$s = 1, \ldots, m$$

$e_s < \frac{1}{2} \Rightarrow w_s > 0$

besser als Raten

$e_s = \frac{1}{2} \Rightarrow w_s = 0$

f_s klassifiziert genauso gut, als würde er bei jedem Muster eine Münze werfen

$e_s > \frac{1}{2} \Rightarrow w_s < 0$

klassifiziert falsch herum

$$F_s(\vec{x}) = \mathrm{sgn}\left(\sum_{i=1}^{s} w_i f_i(\vec{x})\right)$$