Klassifikation

Franz Pernkopf

Institute of Communications and Wave Propagation
University of Technology Graz
Inffeldgasse 16c , 8010 Graz, Austria
Tel: +43 316 873 4436
E-Mail: pernkopf@inw.tugraz.at

Allgemeine Grundlagen zur Klassifikation

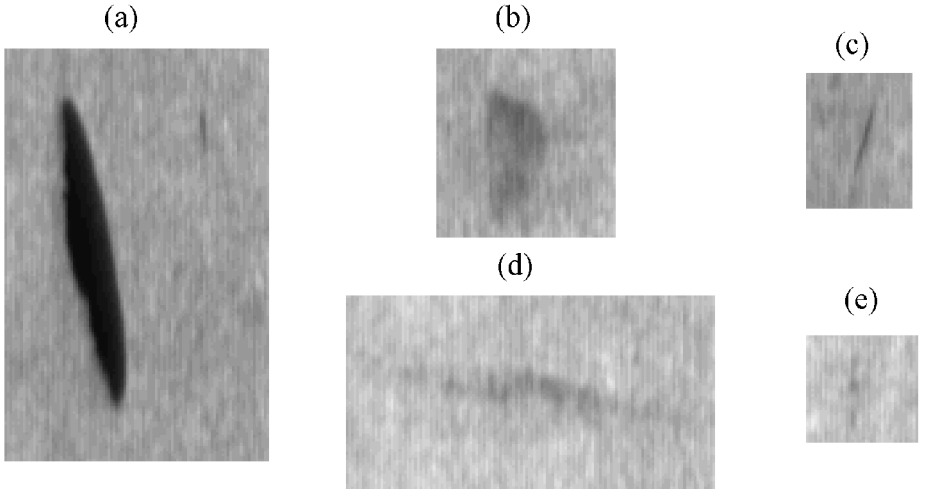
- Ziel:
  - AUTOMATISCHE Mustererkennung
  - Entwicklung von Maschinen, die neue Objekte anhand ihrer Merkmale zu bekannten Klassen zuordnen können
- Attribute können qualitiativ oder quantitativ, kontinuierlich, diskret oder Bool´sche Variable sein
- Merkmalsvektor:

$$\mathbf{x} = \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix}$$

- Statistisch Mustererkennung: Merkmale bestimmen, auftragen im Merkmalsraum, statistische Methoden zur Bestimmung von Trennflächen
  - Beispiel: Klassifikation von Fehlern auf Lagerbolzen (5 Klassen)

Beispiel

- Beispiel: 5 Klassen von Fehlern
  - Class 1: Material flaw (a)
  - Class 2: Grinding flaw (b)
  - Class 3: Scratch (c)
  - Class 4: Chafe mark (d)
  - Class 5: Spot (e)



- Gesamtsystem:
  - Sensoren: Camera
  - Vorverarbeitung (Filter)
  - Segmentierung
  - Merkmalsextraktor
  - Entscheidungsmodul (Klassifikator)
  - Ausgabemodul

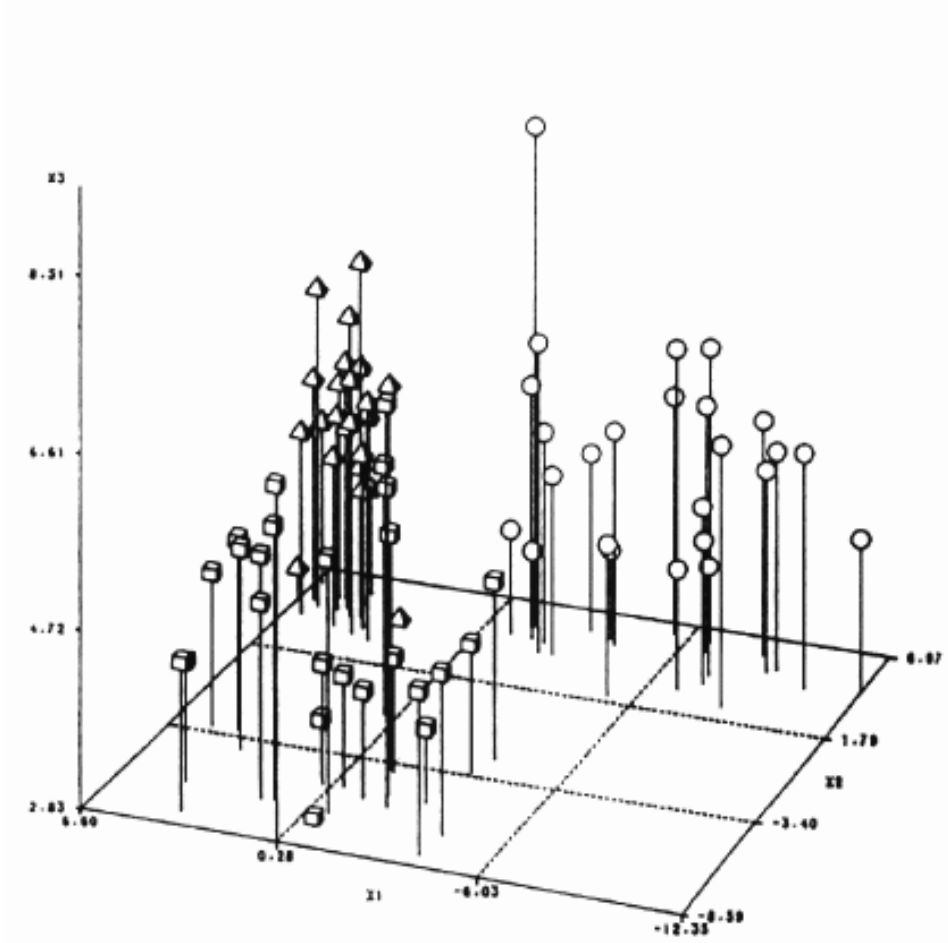
- Extrahierte Merkmale

No.	Feature name
1	Area of all segments
2	Boundary length of all segments
3	Compactness of all segments
4	Center of gravity in x
5	Center of gravity in y
6	Polar measure minimum
7	Polar measure maximum
8	Polar measure mean
9	Ratio of width to height of all segments
10	Height of all segments
11	Width of all segments
12	Roundness of all segments
13	Number of discontinuation in the horizontal projection
14	Number of discontinuation in the vertical projection
15	Number of free columns within the object
16	Number of free rows within the object
17	Mean gray value of all segments
18	Standard deviation of the gray values within the segments
19	Mean gray value of the background
20	Standard deviation of the gray values of the background
21	Ratio of the mean between the segments and the background
22	Ratio of the standard deviation between the segments and the background
23	Maximum gray value of the image
24	Minimum gray value of the image
25	Moment of 4^{th} order of the image
26	Skewness of the image
27	Standard deviation of the image
28	Mean of the image
29	Number of objects
30	Area of the largest segment
31	Boundary length of the largest segment
32	Direction of the largest segment
33	Length of the bounding rectangle of the largest segment
34	Width of the bounding rectangle of the largest segment
35	Rectangularity of the largest segment
36	Compactness of the largest segment
37	Axis ratio of the bounding rectangle
38	2. harmonic of the fourier descriptor
39	3. harmonic of the fourier descriptor
40	4. harmonic of the fourier descriptor
41	5. harmonic of the fourier descriptor
42	Maximum of the gradient in x direction
43	Maximum of the gradient in y direction
44	a_0 of the first order fit of the projection of the largest segment
45	a_1 of the first order fit of the projection of the largest segment
46	Angle of the regression line of the projection of the largest segment
47	Absolute error of the regression line to the projection
48	b_0 of the second order fit of the projection of the largest segment
49	b_1 of the second order fit of the projection of the largest segment
50	b_2 of the second order fit of the projection of the largest segment
51	Absolute error of the second order fit to the projection
52	Ratio of the Areas

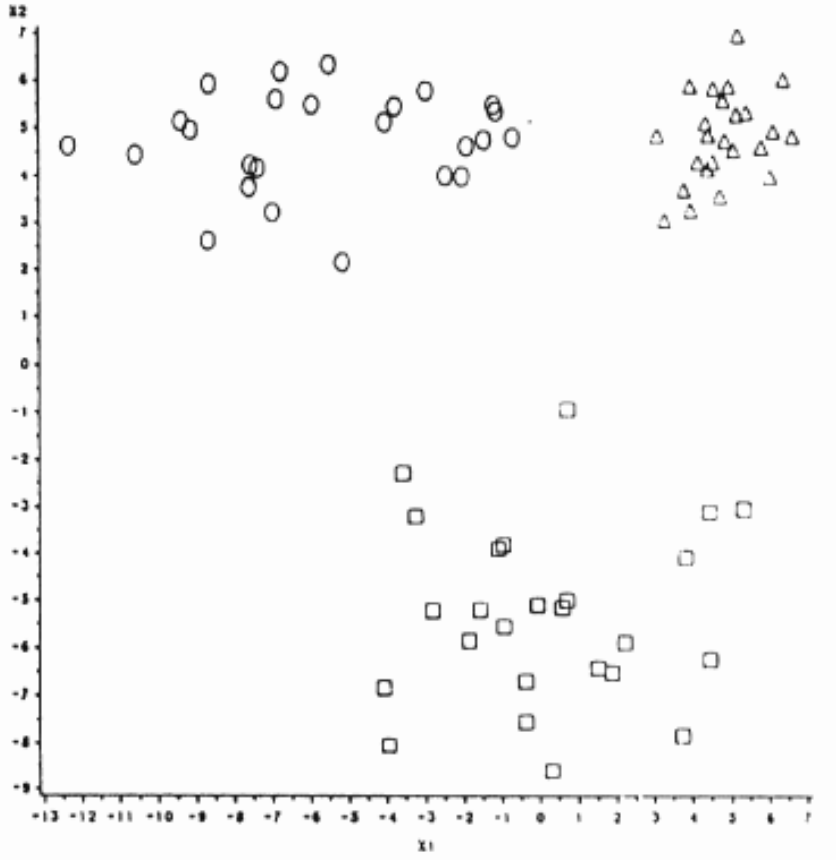
Merkmalsraum

- Objekte gleicher Klasse sollen im Merkmalsraum Cluster bilden.
- Probleme in der Übergangszone: RÜCKWEISUNGSKLASSE
- Oder: weitere Messungen einführen (z. B. Farbe)
- Jedes Objekt entspricht einem Punkt im Merkmalsraum
- > 3 Meßgrößen: Visualisierung nicht mehr möglich
- Dimension im Merkmalsraum:
  - Raum, der von den einzelnen Meßwerten aufgespannt wird - hochdimensional
    (z. B. Lagerbolzen - 52 Dimensionen im Musterraum)
  - Durch Kombination und Auswahl der Meßwerte: Reduktion der Dimension MERKMALSSELEKTION
- Meßwerte und Features werden so gewählt, dass zwischen den Klassen unterschieden werden kann.
- Distanz zu jedem Objekt derselben Klasse < Distanz zum nächsten Nachbarn jeder anderen Klasse
- Cluster sind leider selten kompakt

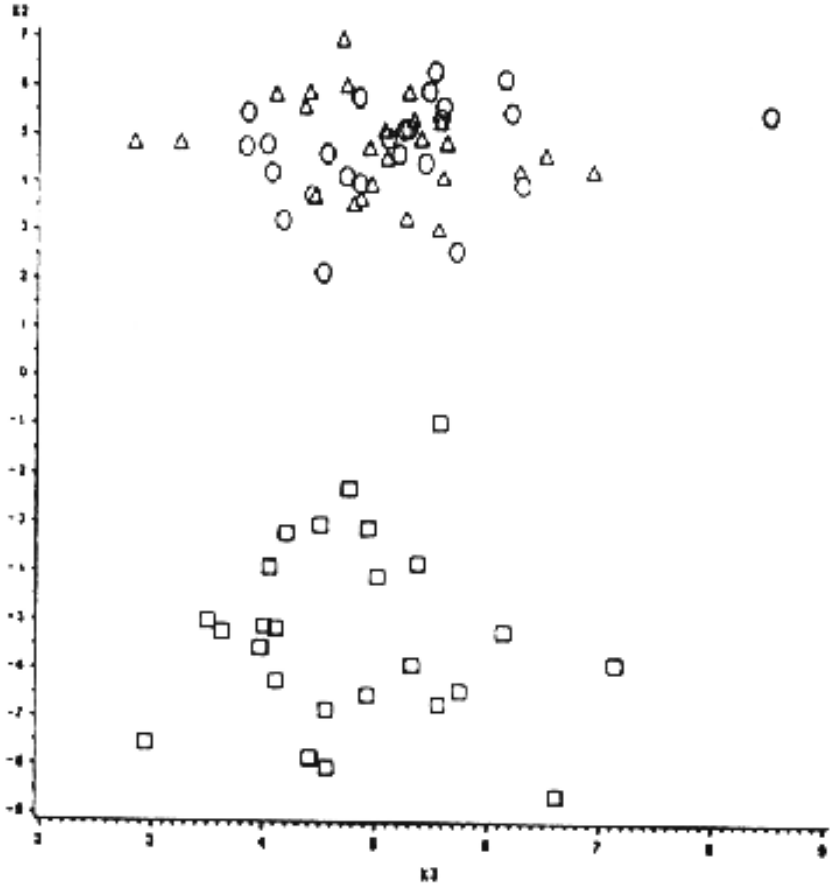
Merkmalsraum



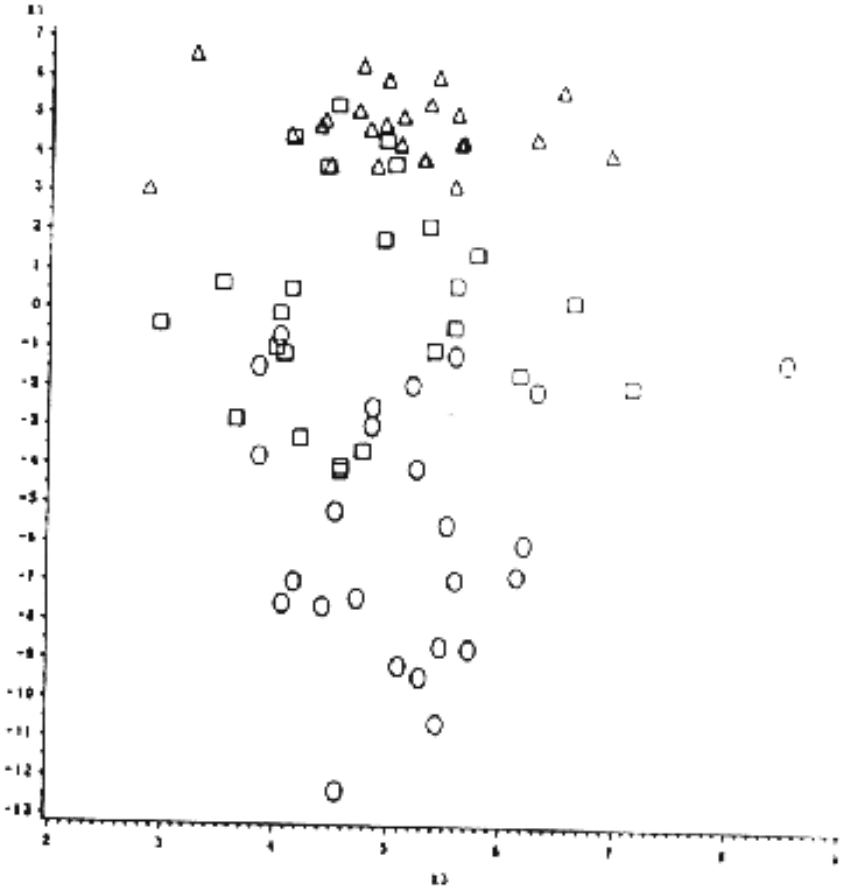
(a) Three-Dimensional Scattergram



(b) x_2 vs. x_1



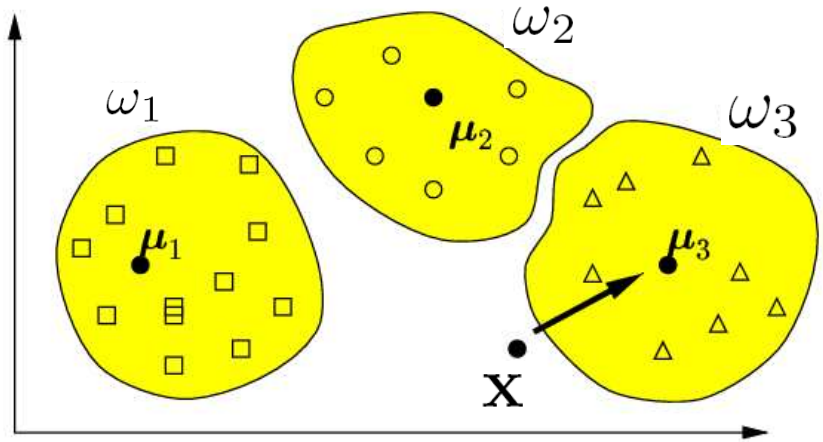
(c) x_2 vs. x_1



(d) x_1 vs. x_3

Überwachte Klassifikation

- Klassen sind bekannt
- Klassifikator wird durch LERNEN auf diese Klassen eingestellt
- Trainingsdaten sind Datenvektoren mit bekannter Klasseninformation
- sollen Struktur und Eigenschaften der gesamten Daten möglichst gut beschreiben
- Vorgangsweise:
  - Trainingsdatensatz erstellen
  - Trainingsdatensatz analysieren
  - Klassifikator auswählen
  - Klassifikator trainieren
  - Genauigkeitsabschätzung
  - Klassifikator ist einsatzbereit



- Notation:
  - Merkmalvektoren $X = \{\vec{x}_k \in \Re^o \mid k = 1,2,...n\}$
  - Klassen $\Omega = \{\omega_i \mid i = 1,2,...c\}$
  - Anzahl der Klassen $c \in \{j \in \aleph \mid 2 \le j\}$
  - Klassifikationsprozeß $\Theta : X \to \Omega$
  - Entscheidungsfunktionen $\Phi = \{\varphi_i \mid i = 1,2,...c\}$
  - wobei $\varphi_i : X \to \Re$ bezüglich ω_i entscheidet
  - Entscheidungsfunktion:
    - maximale Aktivierung

$$S_{\max}(\Phi, \vec{x}) = \omega_i \quad \text{falls} \quad \varphi_i(\vec{x}) = \max_{j=1,2,...,c} \varphi_j(\vec{x})$$

    - minimale Aktivierung

$$S_{\min}(\Phi, \vec{x}) = \omega_i \quad \text{falls} \quad \varphi_i(\vec{x}) = \min_{j=1,2,...,c} \varphi_j(\vec{x})$$

Bayes Klassifikator

- Minimierung der Fehlklassifikationen
- Entscheidungsfunktion $\varphi_{Bayes_i}(\vec{x}) = P(\omega_i \mid \vec{x})$
- Klassifikator $\Theta_{Bayes}(\vec{x}) = S_{\max}(\Phi_{Bayes}, \vec{x})$

- liefert immer optimale Entscheidung
- ABER: $P(\omega_i \mid \vec{x})$ nicht bekannt
- Satz von Bayes:

$$P(\omega_i \mid \vec{x}) = \frac{p(\vec{x} \mid \omega_i)P(\omega_i)}{\sum_{k=1}^{c} p(\vec{x} \mid \omega_k)P(\omega_k)}$$

- damit ergibt sich die Entscheidungsfunktion

$$\varphi_{Bayes_i}(\vec{x}) = p(\vec{x} \mid \omega_i)P(\omega_i)$$
$$\ln \varphi_{Bayes_i}(\vec{x}) = \ln p(\vec{x} \mid \omega_i) + \ln P(\omega_i)$$

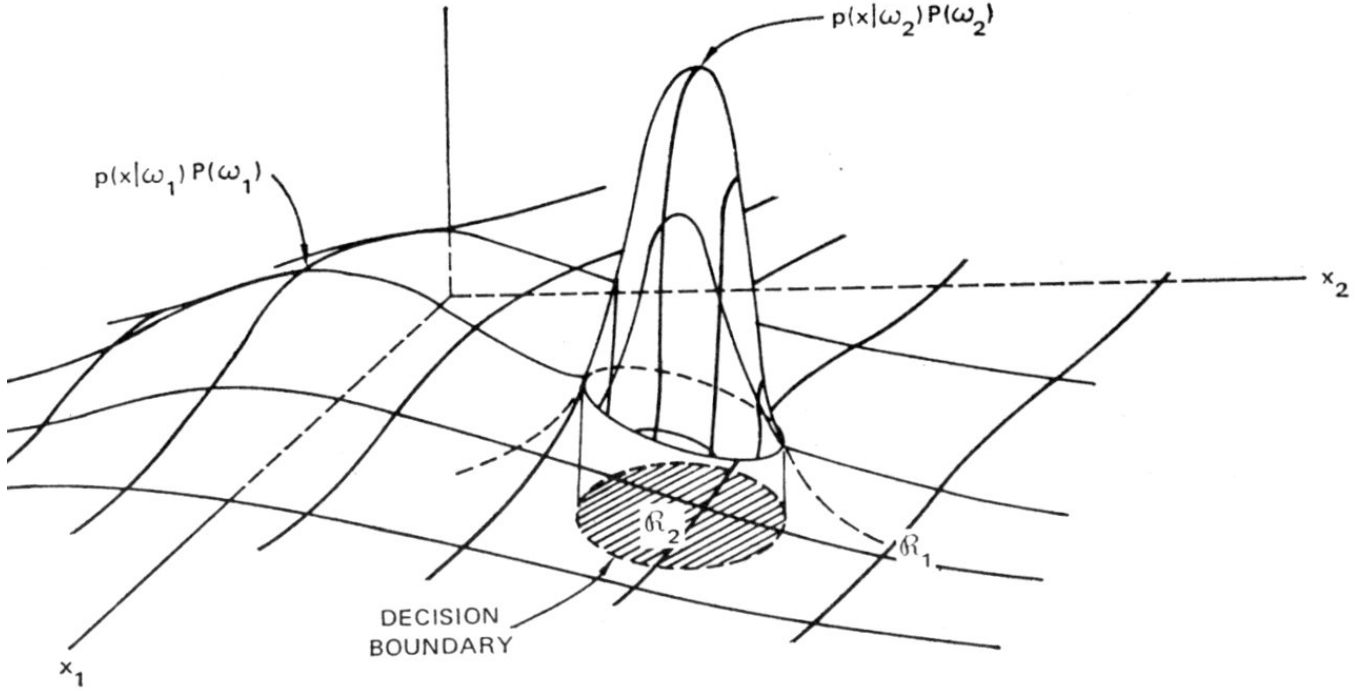
- Bayes Klassifikator wird anwendbar, wenn $p(\vec{x} \mid \omega_i)$ durch Modell geschätzt wird

Patterns $\mathbf{x}$ are assigned to a class ω_j as

$$\mathbf{x} \rightarrow \omega_j \quad \text{if} \quad P(\omega_j|\mathbf{x}) = \max_{i=1,\ldots,m} P(\omega_i|\mathbf{x}),$$

where m is the number of classes. $P(\omega_i|\mathbf{x})$ is the a posteriori probability of the i^{th} class computed by the probability density function $p(\mathbf{x}|\omega_i)$ as

$$P(\omega_i|\mathbf{x}) = \frac{p(\mathbf{x}|\omega_i)\,P(\omega_i)}{\sum_{j=1}^{m} p(\mathbf{x}|\omega_j)\,P(\omega_j)}.$$



Maximum Likelihood

- Annahme: Wahrscheinlichkeitsdichte jeder Klasse ist eine multivariate Normalverteilung

$$p\left(\mathbf{x}|\omega_i\right) = \frac{1}{(2\pi)^{d/2}\sqrt{|\Sigma_i|}} \exp\left[-\frac{1}{2}\left(\mathbf{x}-\mu_i\right)^T \Sigma_i^{-1}\left(\mathbf{x}-\mu_i\right)\right]$$

- Abschätzung der bedingten Wahrscheinlichkeit für jede Klasse $p\left(\mathbf{x}|\omega_i\right)$
- Abschätztung der a-priori Klassenwahrscheinlichkeit $P\left(\omega_i\right)$.

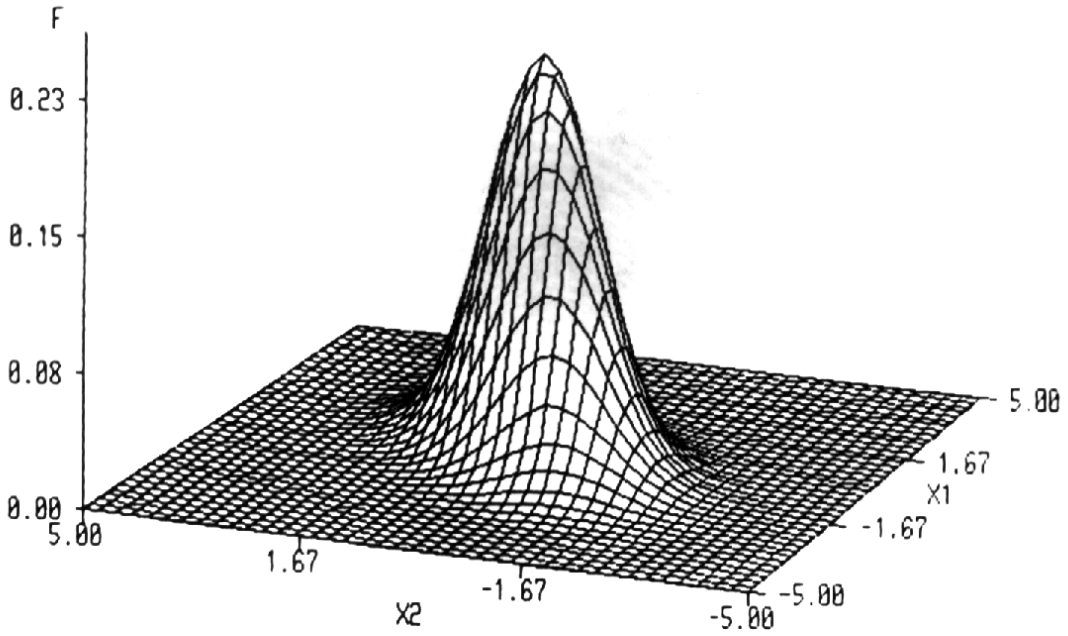


FIG. 7.3. The bivariate normal distribution.

- Entscheidungsfunktion (log-likelihood):

$$\ln \varphi_{Bayes_i}(\vec{x}) = \ln p(\vec{x} \mid \omega_i) + \ln P(\omega_i)$$

$$\varphi_{ML_i} = -\frac{1}{2}\ln|\Sigma_i| - \frac{1}{2}(\vec{x} - \vec{\mu}_i)^T \Sigma_i^{-1}(\vec{x} - \vec{\mu}_i) + \ln P(\omega_i)$$

$$\Theta_{ML}(\vec{x}) = S_{\max}(\Phi_{ML}, \vec{x})$$

Schätzung der Verteilungsparameter (MLE)

Gegeben:

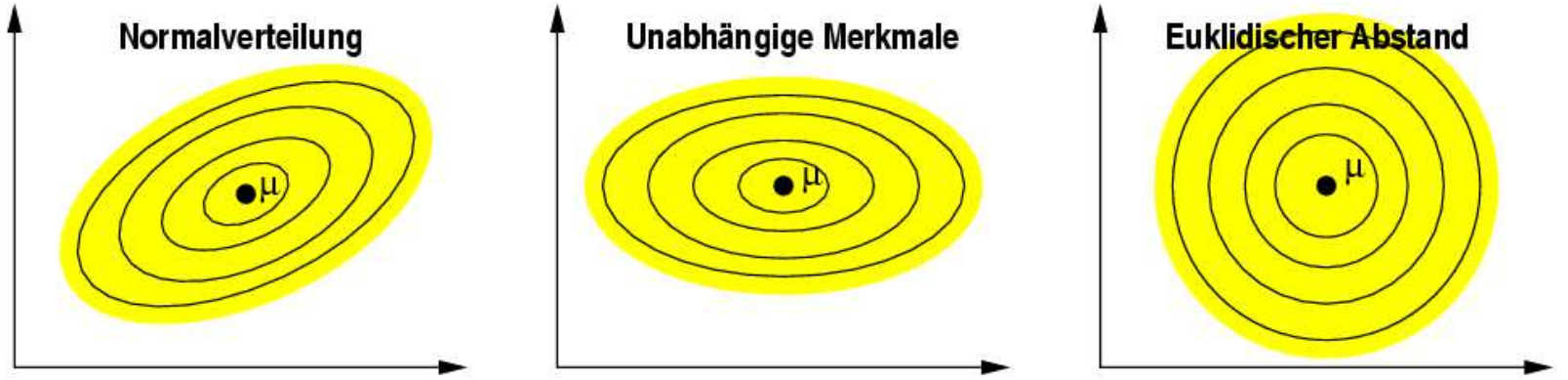
- Die parametrische Struktur der Verteilungsdichten $p(\mathbf{x}|\omega_i)$
- eine etikettierte Lernstichprobe $\{\langle \mathbf{x}_1, \omega_1 \rangle, \langle \mathbf{x}_2, \omega_2 \rangle, \ldots, \langle \mathbf{x}_m, \omega_m \rangle\}$

Gesucht:

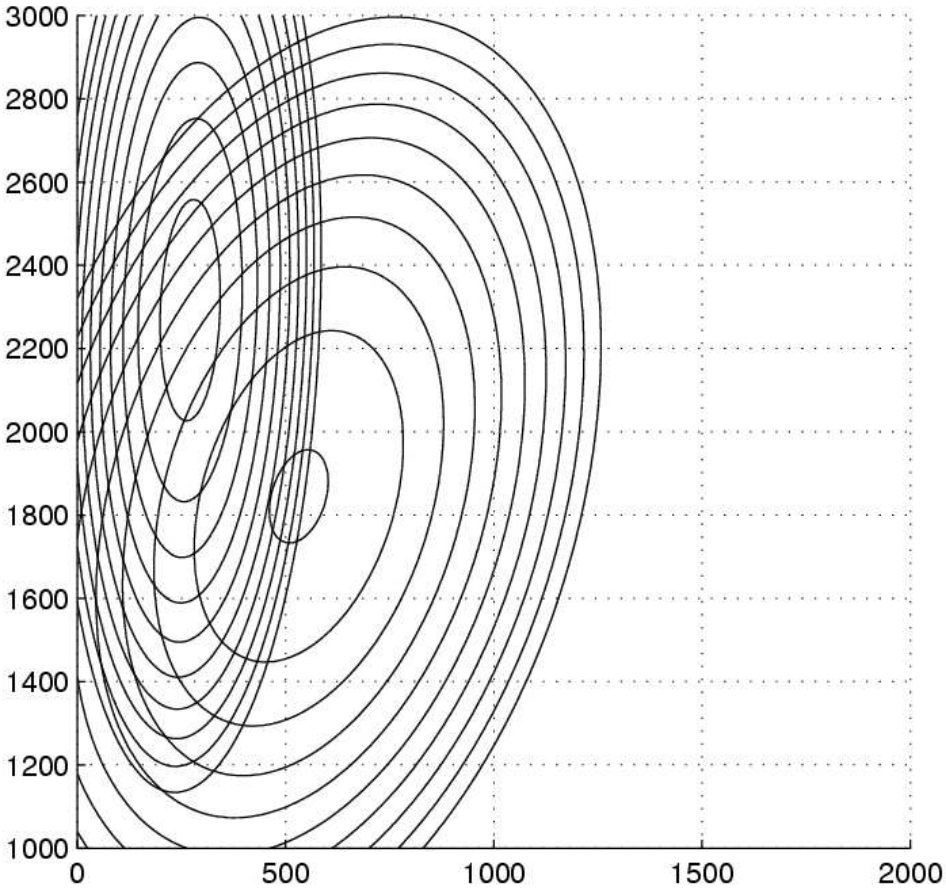
- Parameter $\vec{\mu}_i$ und Σ_i der Verteilungsdichten $p(\mathbf{x}|\omega_i)$ aus den Lerndaten:

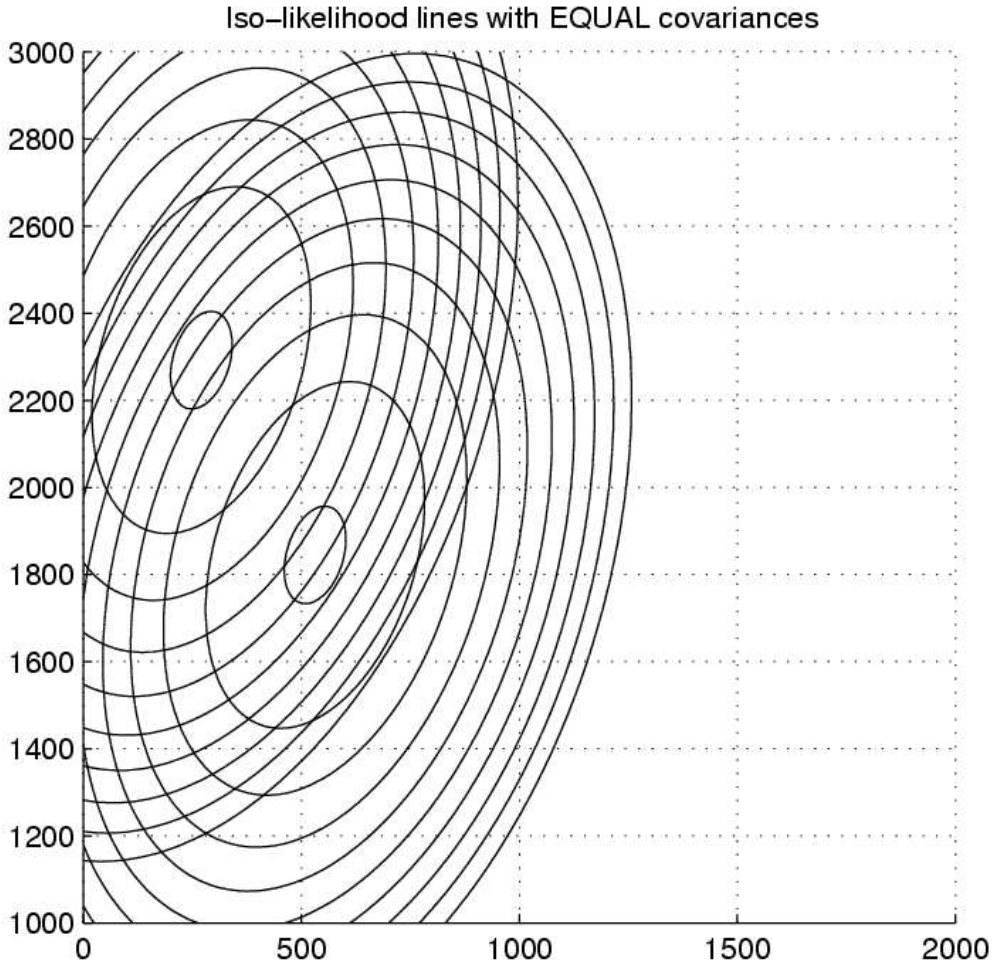
$$\vec{\mu}_i^* = \frac{1}{n_i} \sum_{j=1}^{n_i} \vec{x}_j \qquad \Sigma_i^* = \frac{1}{n_i - 1} \sum_{j=1}^{n_i} (\vec{x}_j - \vec{\mu}_i^*)(\vec{x}_j - \vec{\mu}_i^*)^T$$

Form der Kovarianz-Matrix:



Iso-likelihood lines for the Gaussian pdf





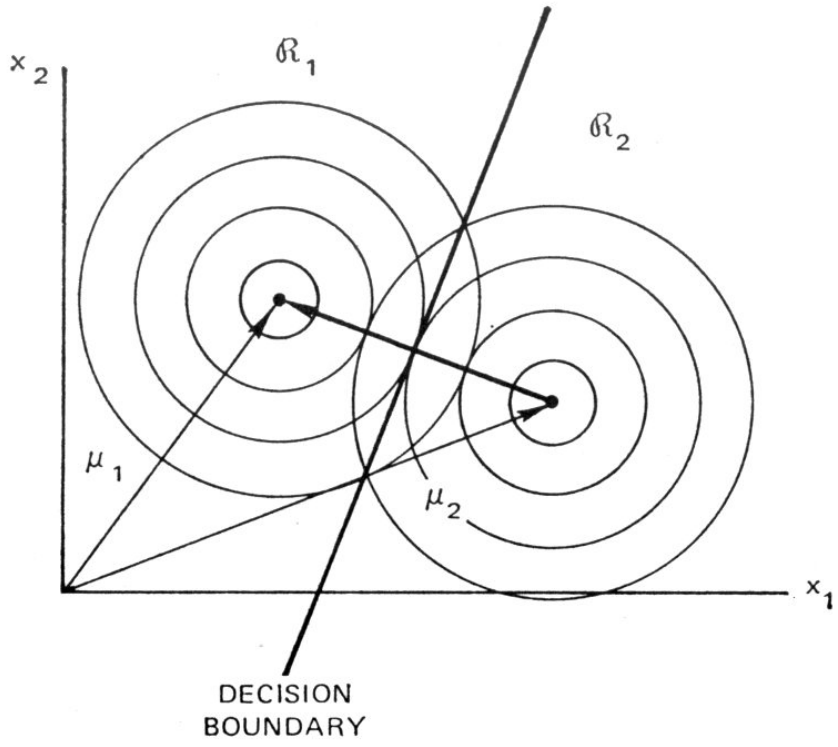
- Discriminant function:

$$\varphi_{ML_i} = -\frac{1}{2}(\mathbf{x}-\mu_i)^T \Sigma_i^{-1}(\mathbf{x}-\mu_i) - \frac{d}{2}\ln 2\pi - \frac{1}{2}\ln|\Sigma_i| + \ln P(\omega_i)$$

- Case 1 (Minimum distance classifier)

  Assumption: $\Sigma_i = \sigma^2 \mathbf{I}$

  Kovarianzmatrix aller Klassen sind gleich. Die Komponenten sind voneinander unabhängig und haben gleiche Varianz



  Discriminant function:

$$\varphi_{ML_i} = -\frac{\| \mathbf{x}-\mu_i \|^2}{2\sigma^2} + \ln P(\omega_i)$$

  weitere Vereinfachung: Verzicht auf a priori Wahrscheinlichkeit $P(\omega_i)$

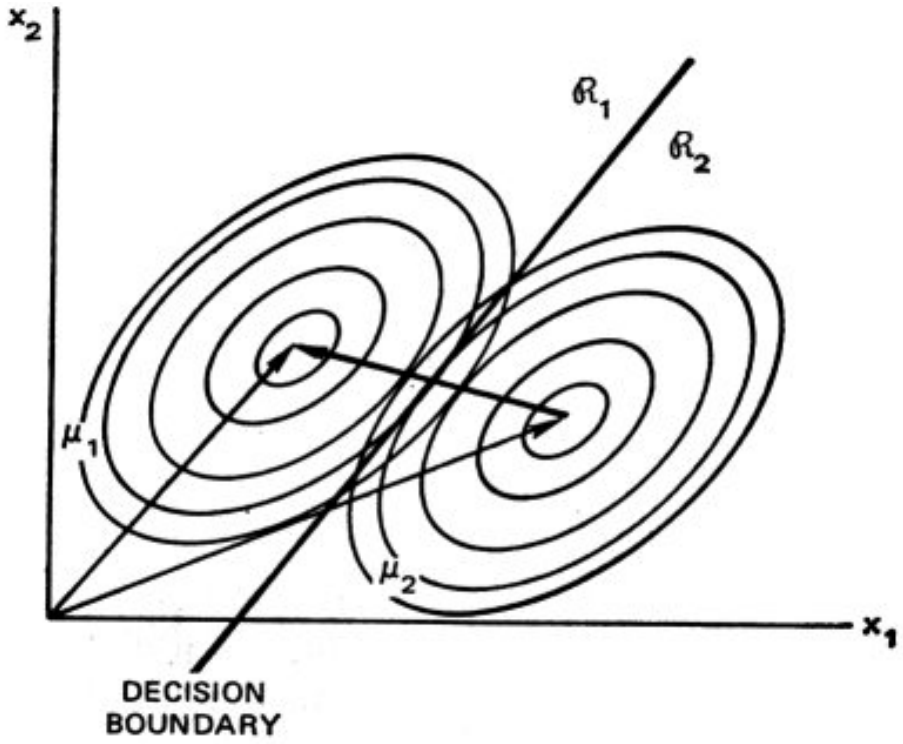
- Case 2 (Mahalanobis distance classifier)

Assumption: $\Sigma_i = \Sigma$

Kovarianzmatrix aller Klassen ist gleich

Discriminant function:

$$\varphi_{ML_i} = -\frac{1}{2}(\mathbf{x} - \mu_i)^T \Sigma^{-1} (\mathbf{x} - \mu_i) + \ln P(\omega_i)$$



- Case 3 (Covariance matrices are arbitrary)

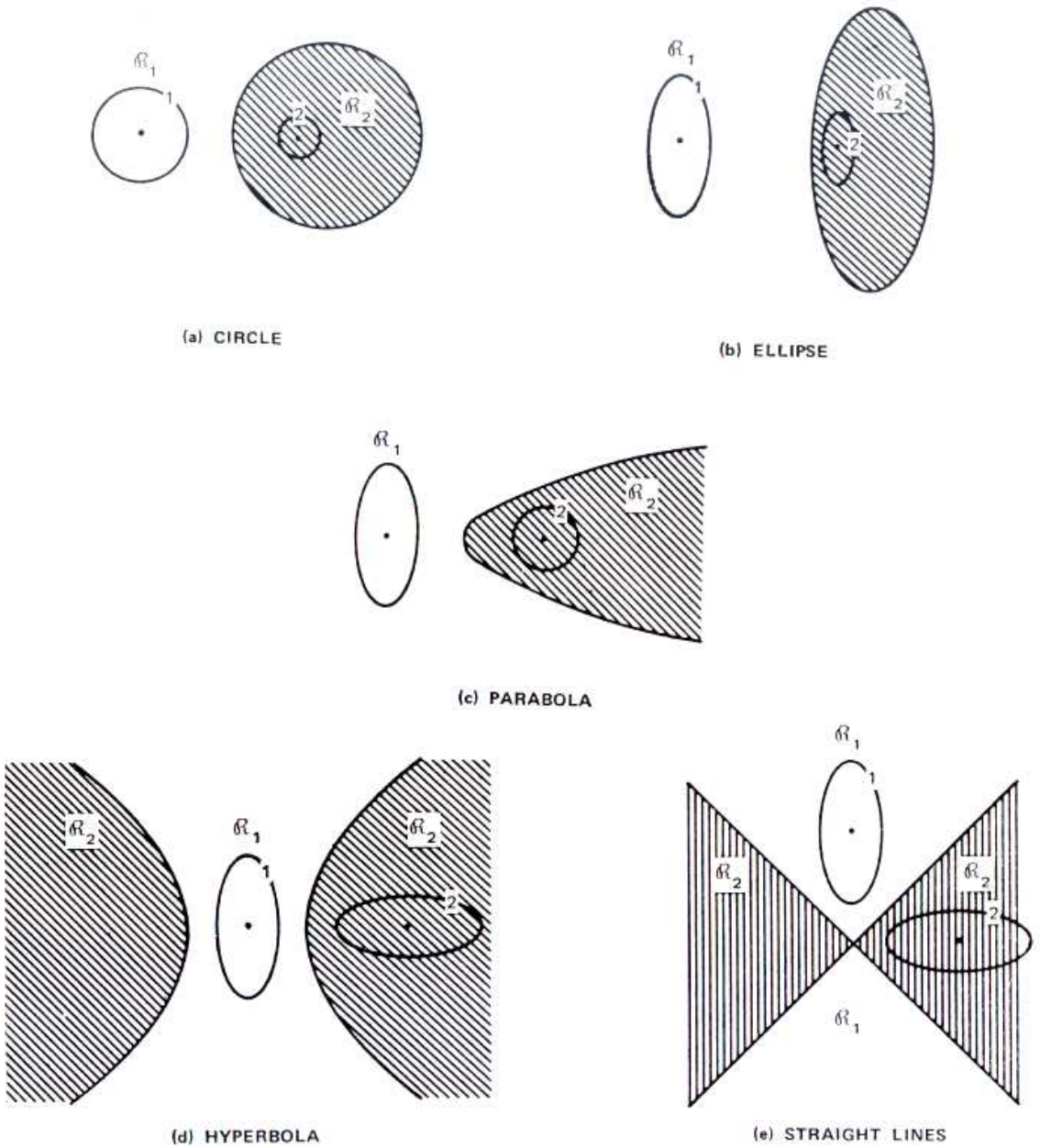


FIGURE 2.10. Forms for decision boundaries for the general bivariate normal case.

- Probleme beim Bayes Ansatz:
  - Entscheidungsregeln nicht immer einfach
  - Komplizierte Berechnung für nicht normalverteilte Parametrisierung
  - Aufgrund der Minimierung der Fehlklassifikationen: Klassen mit geringer a priori Wahrscheinlichkeit haben wenig Einfluß aufs Design
  - A priori Wahrscheinlichkeit ist schwer abzuschätzen

- Nearest Neighbor Klassifikator
  - Ein Prototyp pro Klasse speichern, Distanz zu allen Prototypen bestimmen:
    Minimum Distanz Klassifikator
  - Viele Prototypen pro Klasse speichern - Chancen für richtige Klasse steigen 1-NN
  - Verbesserung entlang der Klassengrenzen: *k* nächste Nachbarn bestimmen und Mehrheit auswählen *k*-NN Klassifikator

k-NN Klassifikator

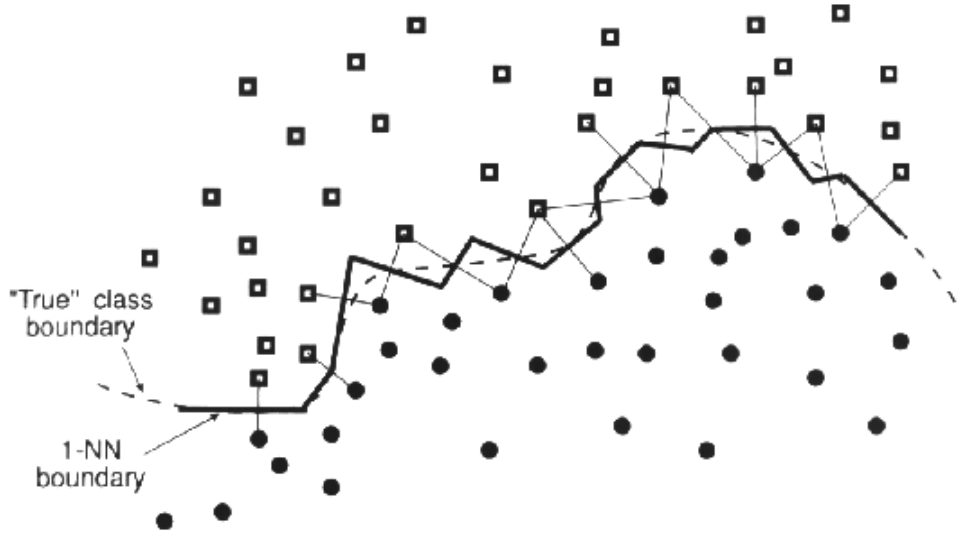


FIG. 8.1. 1 – *NN* classification.

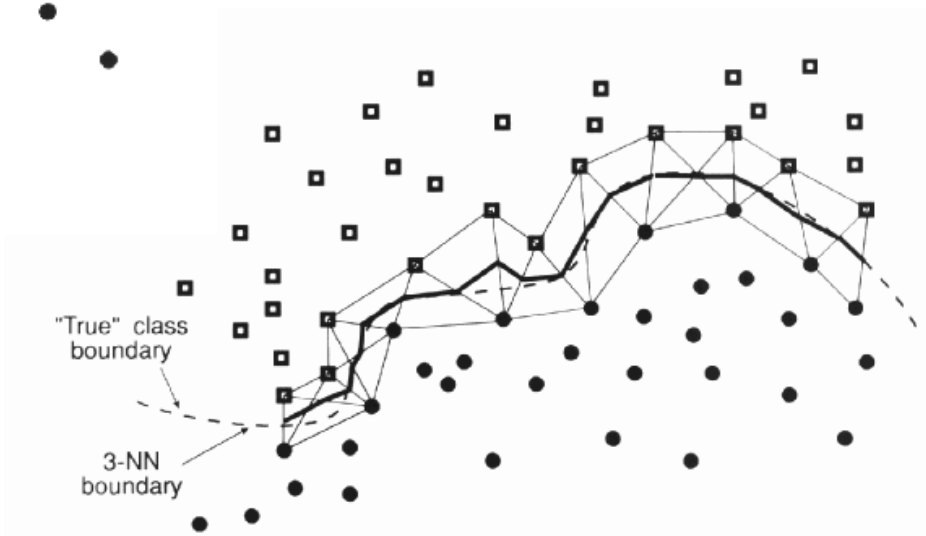


FIG. 8.2. 3 – *NN* classification.

Trainingsphase schnell - auswählen und speichern der Prototypen
Klassifikation sehr rechenaufwendig - Vergleich mit allen Prototypen

- 2-stufiger Schwellwert: **(k,l) Methode**: Mindestens l von k Nachbarn müssen dieselbe Entscheidung liefern, ansonsten wird das Objekt zurückgewiesen.
- ideal: Unendliche # von Prototypen
- aber: benötigt zuviel Speicherplatz und Rechenzeit
- Abhilfe: innere Prototypen sind redundant
- nur Prototypen entlang der Grenzen speichern

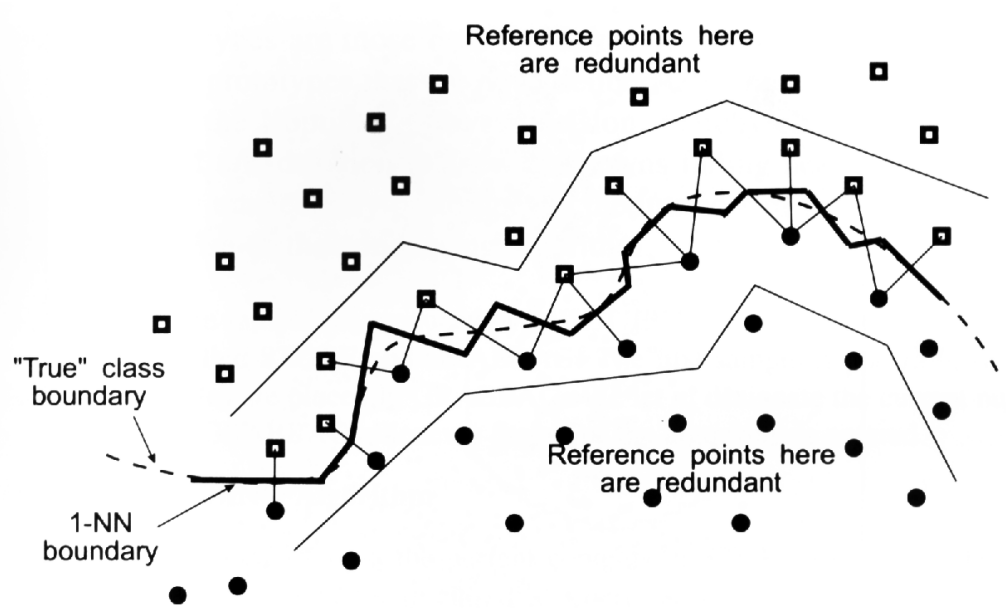
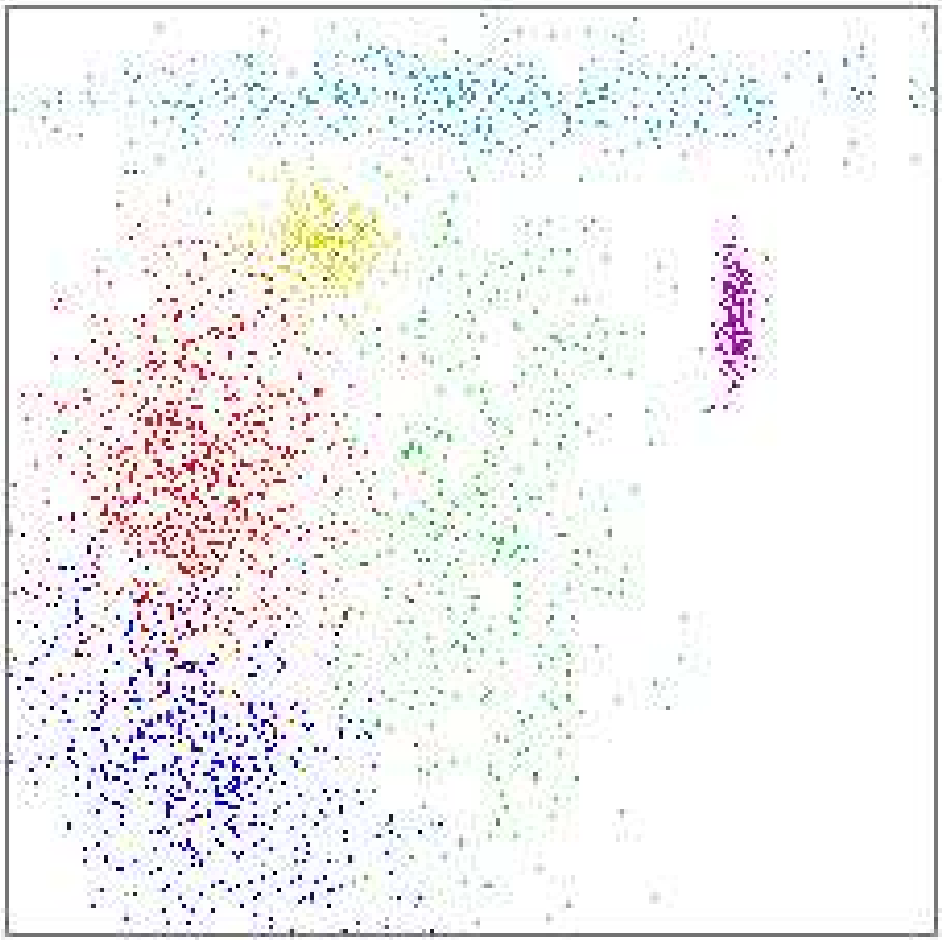
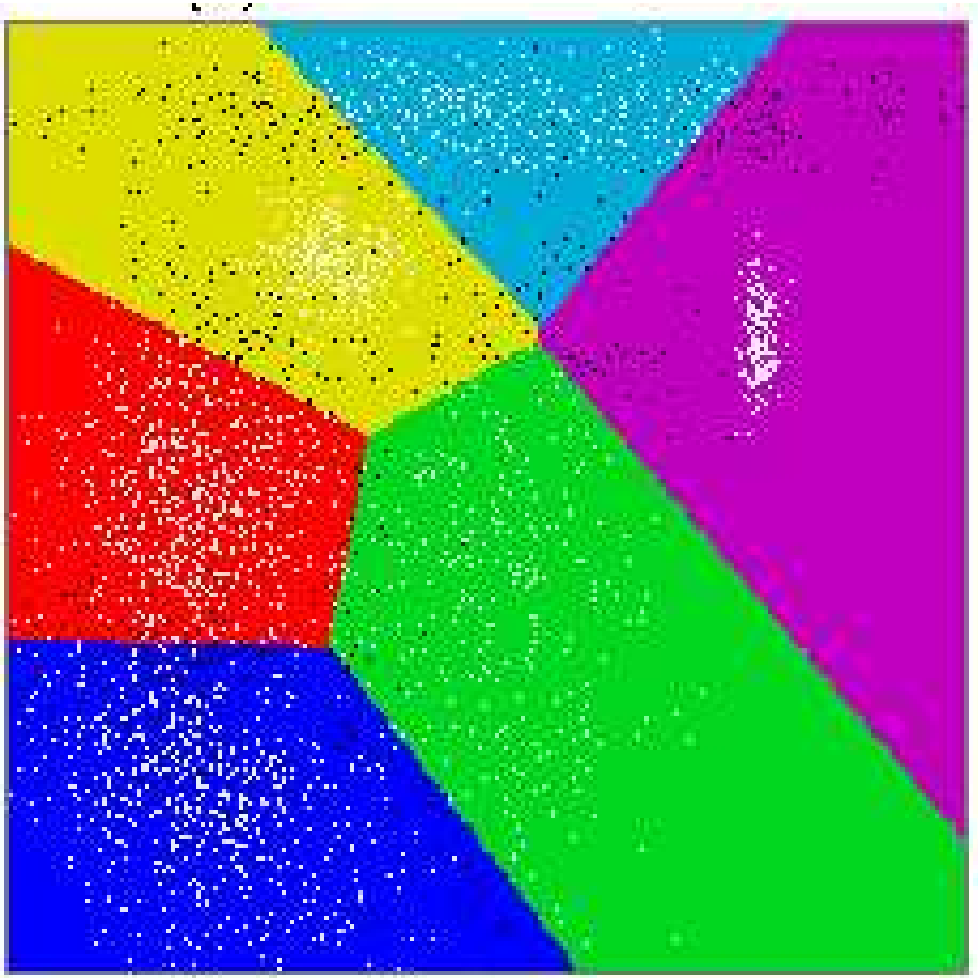


FIG. 8.4. Redundant points in nearest-neighbor method.

Vergleich der Klassifikatoren

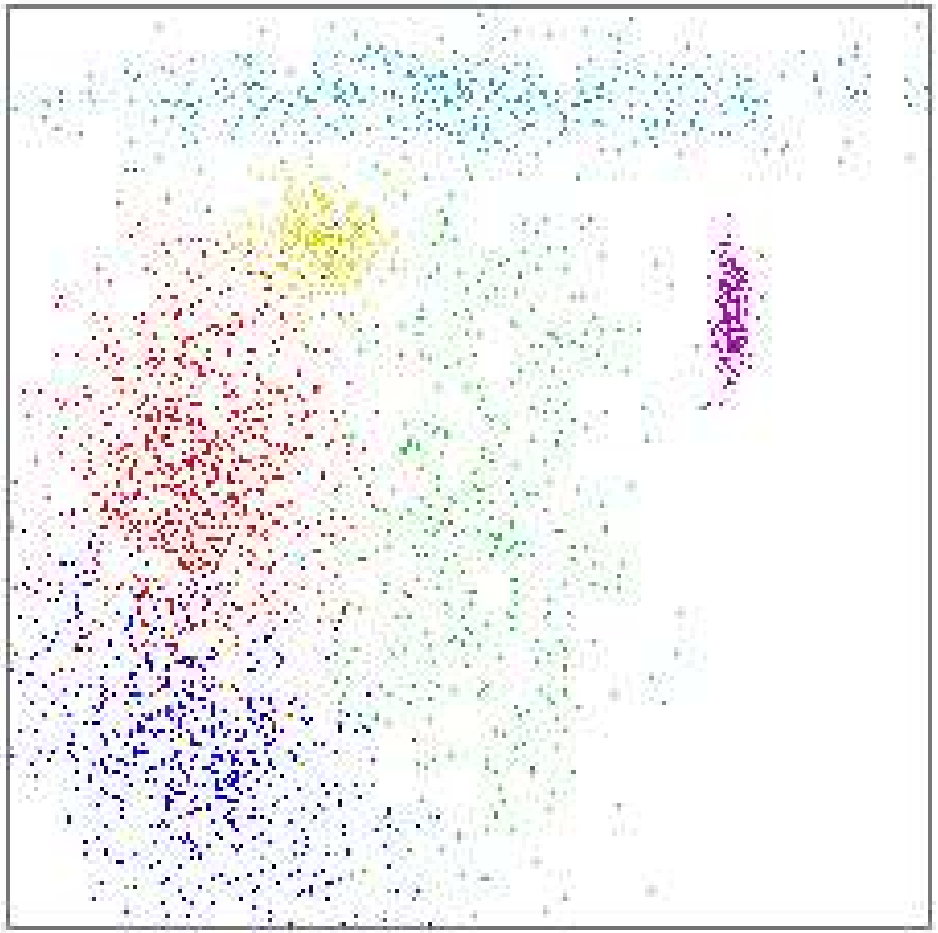


(a) Lerndaten

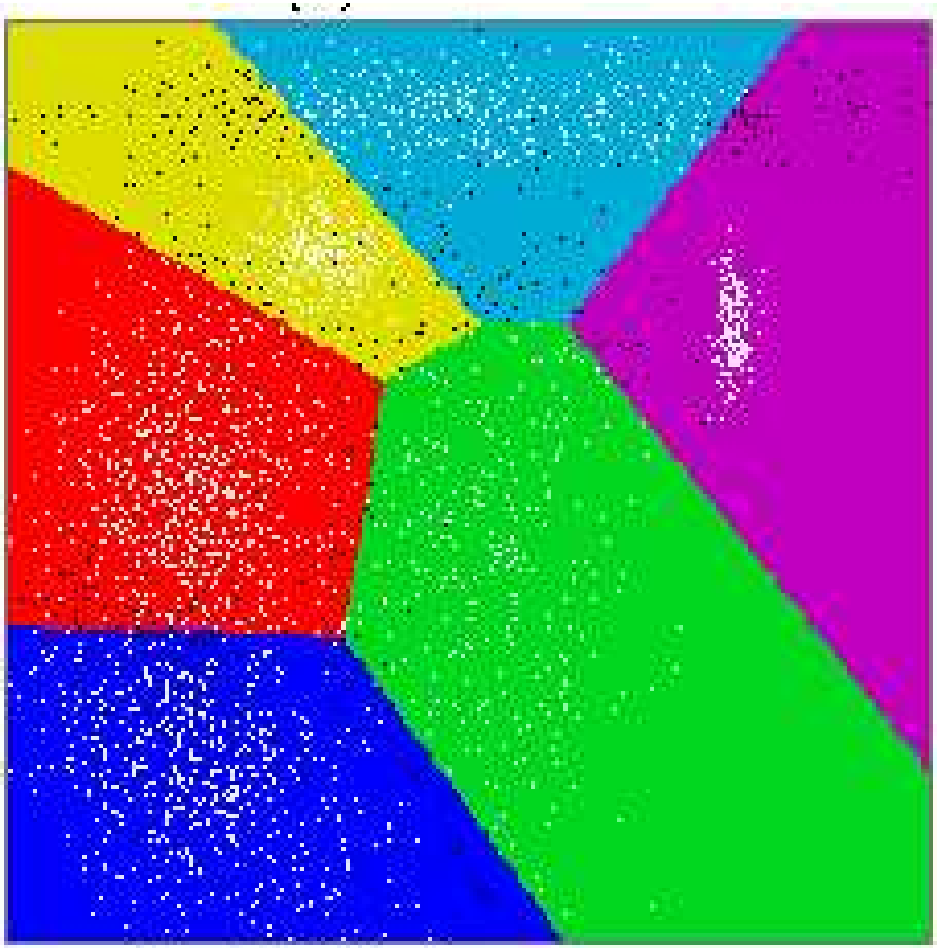


(f) MDs Klassifikator

Vergleich der Klassifikatoren

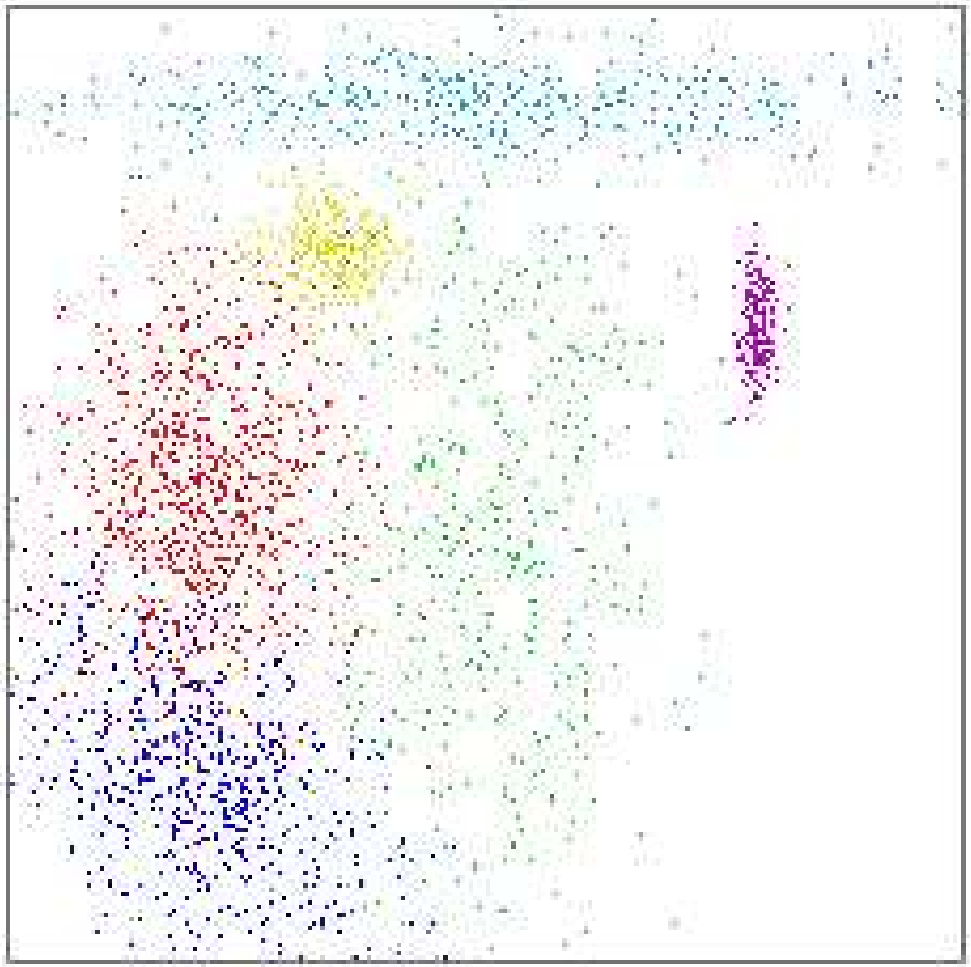


(a) Lerndaten

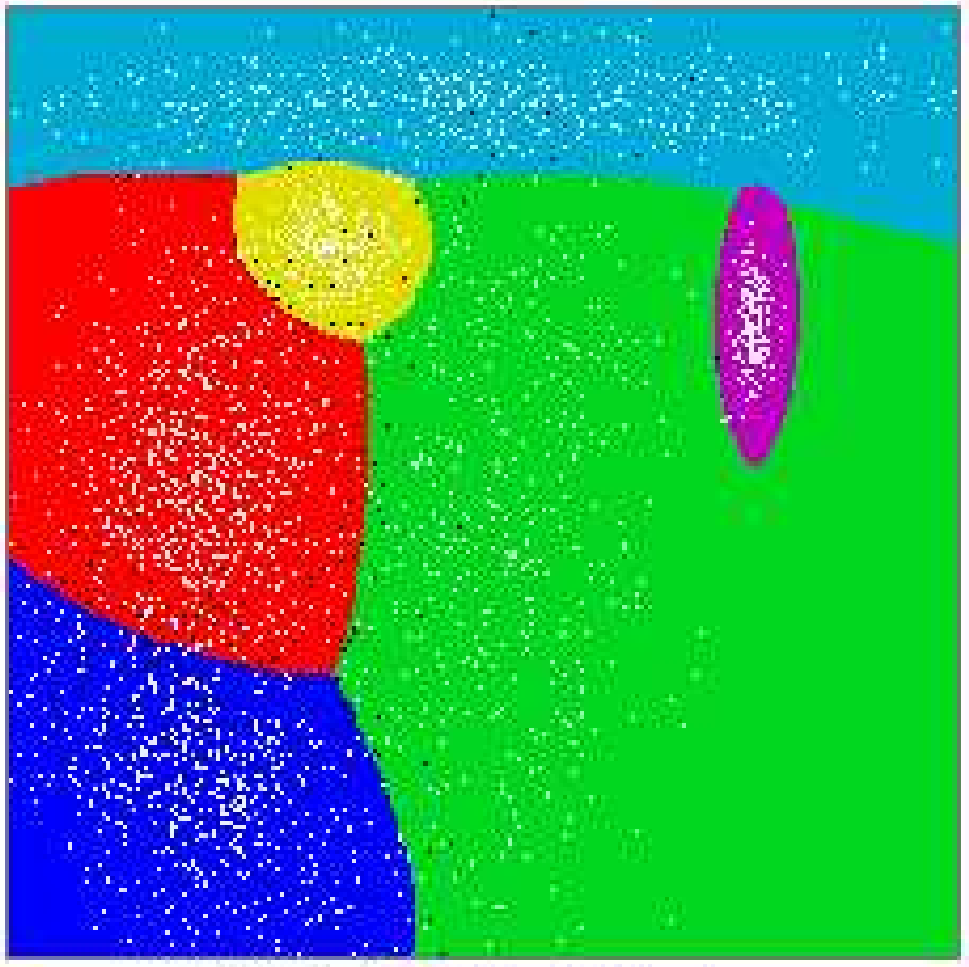


(c) MA Klassifikator

Vergleich der Klassifikatoren

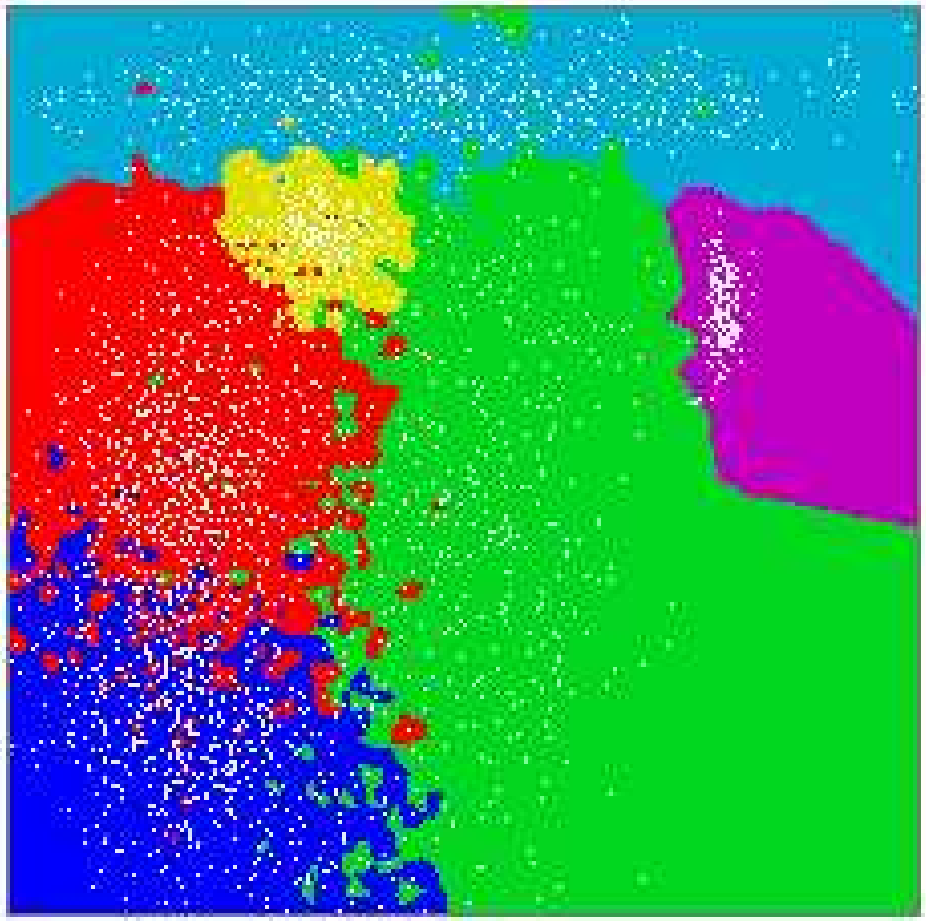


(a) Lerndaten

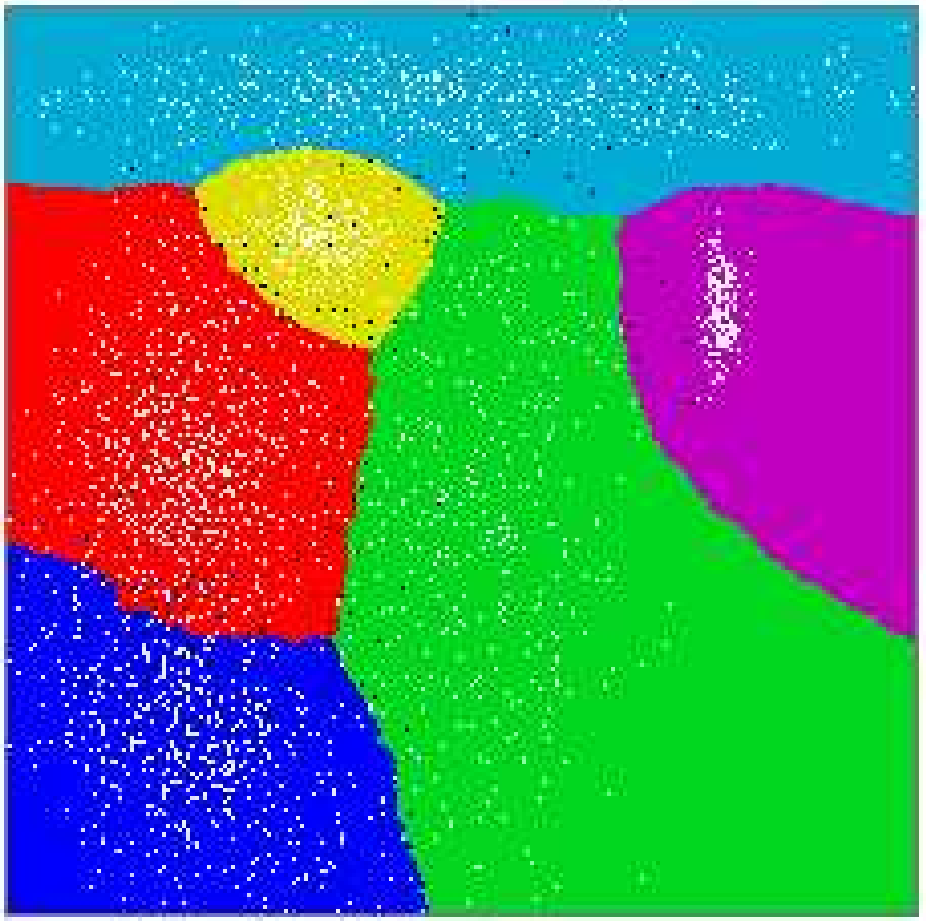


(b) ML Klassifikator

Vergleich der Klassifikatoren



(a) kNN Klassifikator mit k=1



(b) kNN Klassifikator mit k=70

Genauigkeitsabschätzung

- Warum?
  - Qualitätsbeurteilung
  - Klassifikatorauswahl
- Methoden:
  - aus den Lerndaten
  - Holdout/Cross-Validation
  - Leave one Out
  - *n*-fold Cross-Validation

Genauigkeitsabschätzung aus den Lerndaten

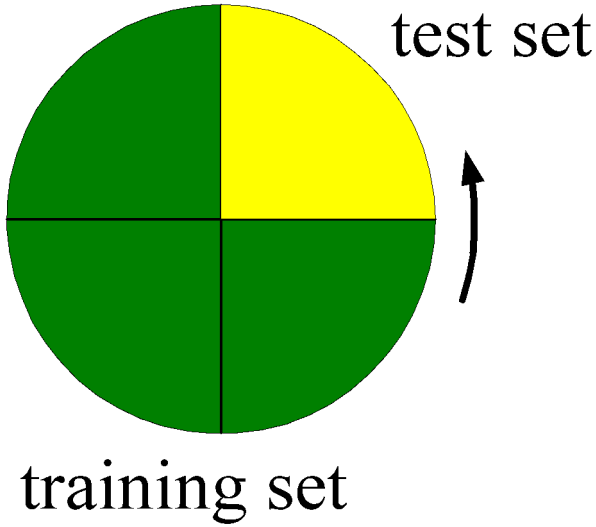
Fehlklassifikationstabelle:
(optimistische Schätzung)

Correct class	Predicted Class					No. of Objects
	1	2	3	4	5	
1	94 17.41%	1 0.19%	2 0.37%	0 0.00%	11 2.04%	108 20.00%
2	0 0.00%	95 17.59%	3 0.56%	10 1.85%	0 0.00%	108 20.00%
3	3 0.56%	1 0.19%	94 17.41%	3 0.56%	10 1.30%	108 20.00%
4	0 0.00%	10 1.85%	5 0.93%	88 16.30%	5 0.93%	108 20.00%
5	8 1.48%	1 0.19%	7 1.30%	2 0.37%	90 16.67%	108 20.00%

- Holdout/Cross-Validation
  - Aufteilung der Referenzdaten in Lerndatenset und Validationsdatenset (zufällige Auswahl!)
  - Fehlerschätzung am Validationsdatenset
  - Nachteil: nicht alle Referenzdaten werden zum Training verwendet
  - liefert pessimistische Schätzung

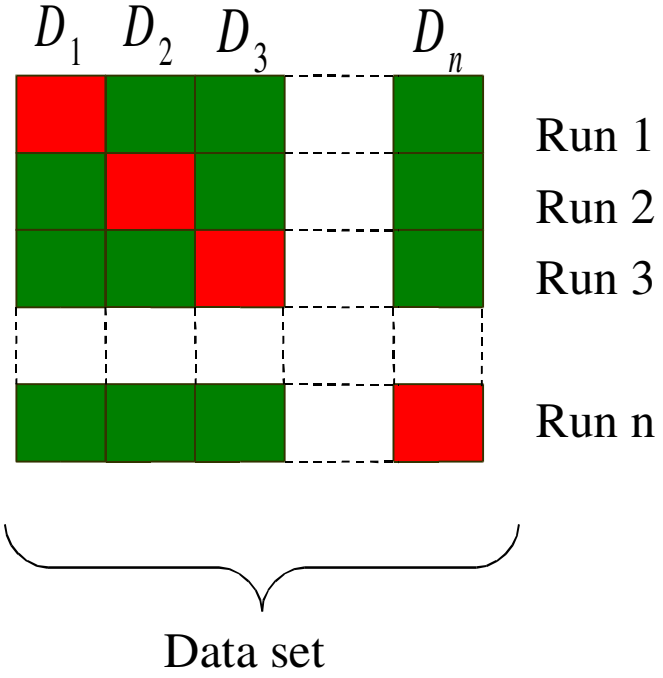
- Leave-one-out
  - p Referenzdaten aufspalten in $p-1$ Lerndaten und 1 Validationsdatum
  - Klassifikatordesign für alle p Partitionen (p mal Lernen!)
  - Fehlerschätzung durch Mittelwert über alle Validationsdaten
  - wird vor allem bei kleinen Referenzdatensätzen verwendet.

- *n*-fold Cross-Validation
  - Aufteilung der Referenzdaten in *n* Teile, *n*–1 Teile als Lerndaten und 1 Teil als Validationsdaten
  - Klassifikatordesign für alle *n* Partitionen (*n*-mal Lernen!)
  - Qualitätsbeurteilung durch Mittelung über alle Validationsergebnisse
  - Tatsächliches Design meist unter Verwendung aller Referenzdaten



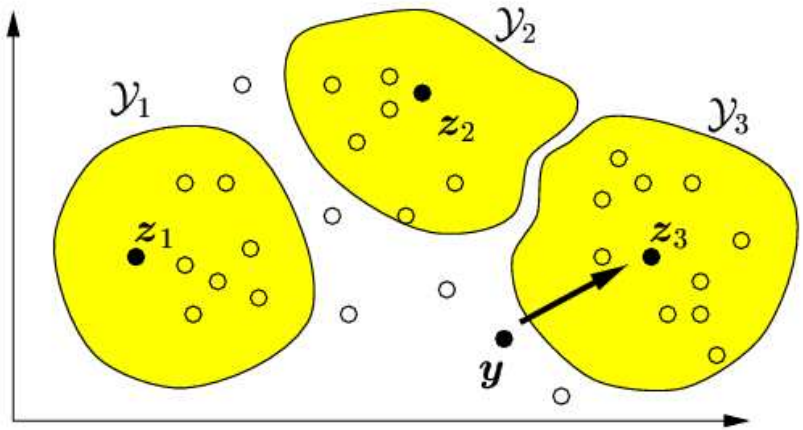
Genauigkeitsabschätzung: *n*-fold Cross-Validation

- Evaluating the performance of a classifier: *n*-fold cross validation
- Partition the data set in *n* segments
- Do *n* times
  - Train the classifier with the green segments
  - Test accuracy on the red segments
- Compute statistics on the *n* runs
  - Mean squared error
  - Variance

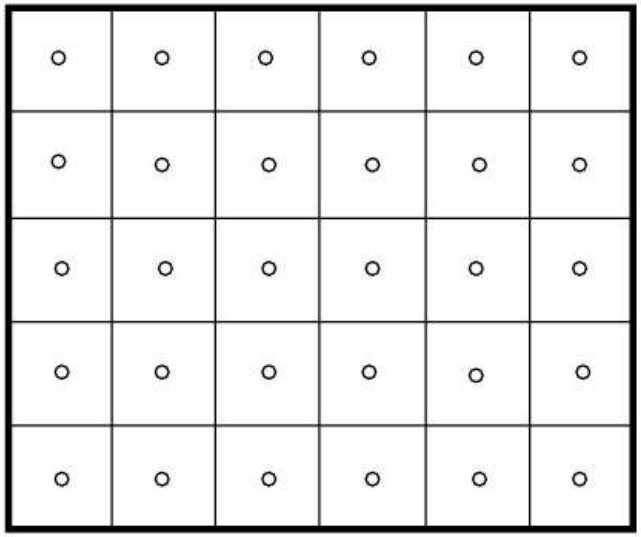


Unüberwachtes Lernen (Klassifikation)

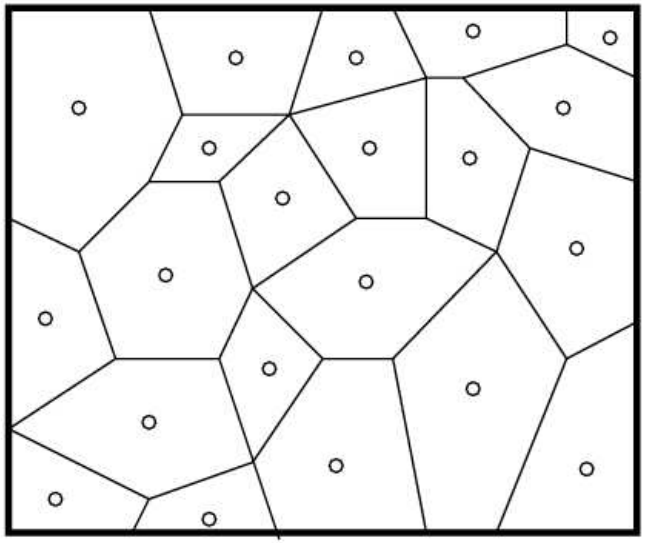
- Unüberwachtes Lernen ist der Versuch, Strukturen in einer Menge von Beobachtungen ohne gegeben Klassenzuordnung (Unlabeled Samples) zu finden.
  - Merkmalvektoren $\boldsymbol{x} \in \mathbb{R}^D$
  - Repräsentantenvektoren $\boldsymbol{z}_1, \ldots, \boldsymbol{z}_K$
  - Nichtetikettierte Lernstichprobe



- Anwendungsgebiete:
  - Datenreduktion durch Vektorquantisierung
  - Klassenmodellierung mit Mischverteilungen (Mixtures of Gaussians)
    - Wieviele Klassen kommen in den Daten vor?
    - Wo liegen die Cluster?

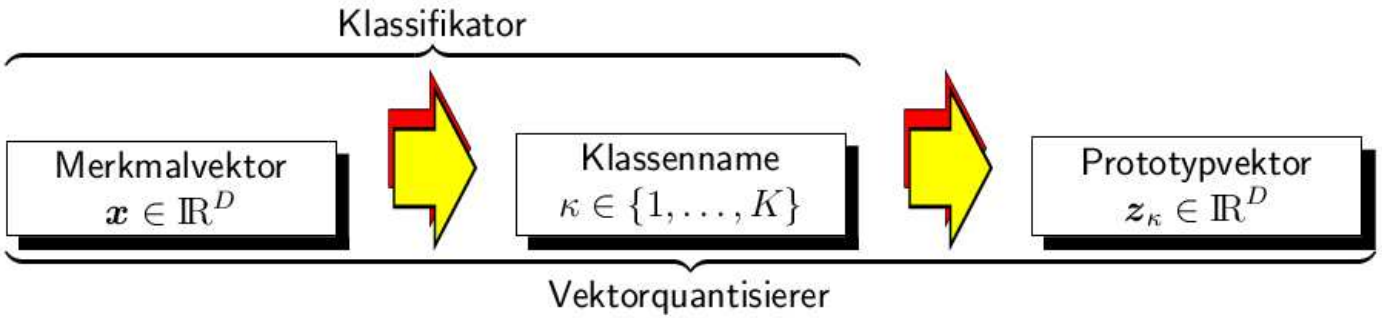


Uniforme Quantisierung der Ebene



Nicht–uniforme Quantisierung

Klassifikation und Vektorquantisierung



Unüberwachtes Lernen: Vektorquantisierung

Ein ***Vektorquantisierer*** ist ein Operator

$$q : \begin{cases} \mathbb{R}^D & \rightarrow & \mathcal{Z} = \{\boldsymbol{z}_1, \ldots, \boldsymbol{z}_K\} \\ \boldsymbol{x} & \mapsto & q(\boldsymbol{x}) \end{cases}$$

- $\boldsymbol{z}_\kappa$ ist der κ-te ***Prototypvektor***
- $\mathcal{Z}$ ist das (endliche) ***Kodebuch*** des Quantisierers
- K ist die ***Kodebuchgröße***

Disjunkte Zerlegung von $\mathbb{R}^D$ („Partition“)

$$\mathbb{R}^D = \mathcal{Y}_1 \cup \cdots \cup \mathcal{Y}_K$$

in ***Zellen*** der Form

$$\mathcal{Y}_\kappa \stackrel{\text{def}}{=} \{\boldsymbol{x} \mid q(\boldsymbol{x}) = \boldsymbol{z}_\kappa\}$$

Der ***optimale*** Vektorquantisierer minimiert die ***Verzerrung***

$$\varepsilon = \mathcal{E}[d(\boldsymbol{X}, q(\boldsymbol{X})] = \sum_{\kappa=1}^{K} \int_{x \in \mathcal{Y}_\kappa} d(\boldsymbol{x}, \boldsymbol{z}_\kappa) \, d\boldsymbol{x}$$

„Quantisierungsfehler“ bzgl. einer geeigneten Abstandsmetrik $d(\cdot, \cdot)$

- Bedingungen für das Kodebuch und die Zellenstruktur des minimal verzerrenden Quantisierers:
  - Der optimale VQ wählt stets den *nächstliegenden* Prototypen zur Klassifikation aus.

$$q(\cdot) \quad \rightsquigarrow \quad \hat{\mathcal{Y}}_\kappa(\mathcal{Z}) \;=\; \{\boldsymbol{x} \in \mathbb{R}^D \mid d(\boldsymbol{x}, \boldsymbol{z}_\kappa) = \min_\lambda d(\boldsymbol{x}, \boldsymbol{z}_\lambda)\}$$

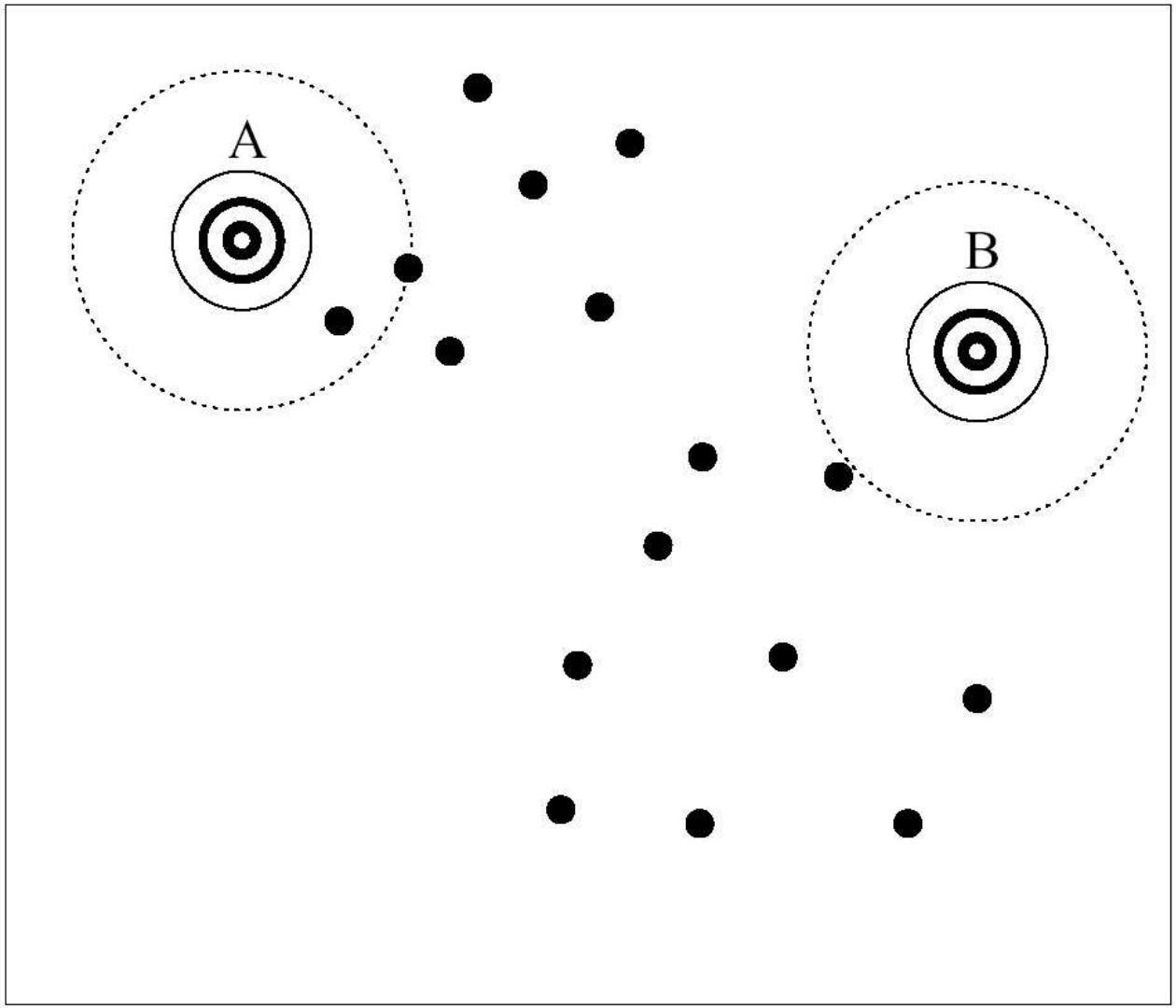
Für ein gegebenes Kodebuch $\mathcal{Z}$ ruft die Klassenzerlegung $\hat{\mathcal{Y}}_1(\mathcal{Z}), \ldots, \hat{\mathcal{Y}}_K(\mathcal{Z})$ den kleinsten Quantisierungsfehler hervor.

  - Prototypvektor $\boldsymbol{z}_\kappa$ ist immer das *Klassenzentroid* $\boldsymbol{z}(\mathcal{Y}_\kappa)$

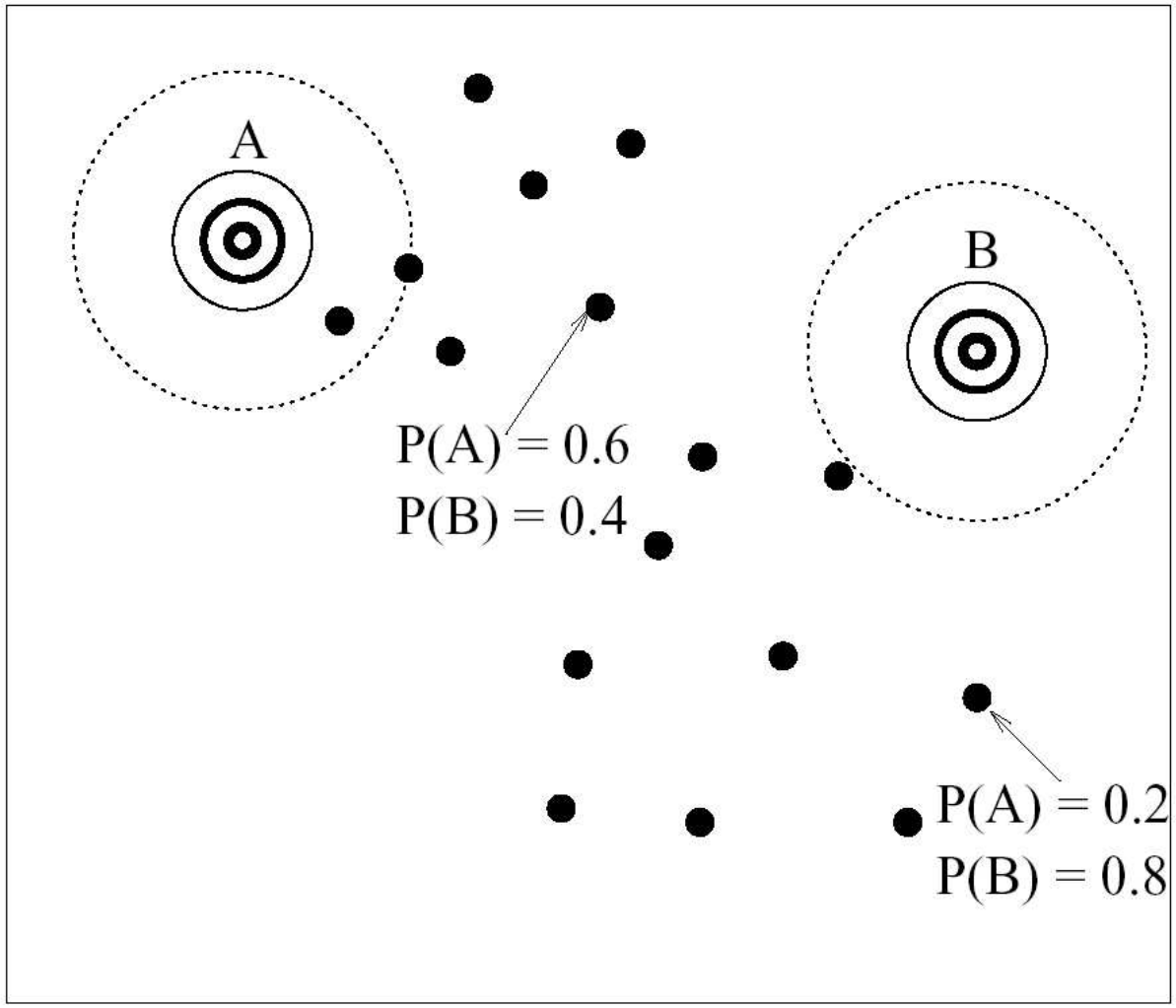
- Vorgangsweise:
  1) Startwerte für Klassenzentren wählen (Anzahl der Cluster definieren),
  2) Zuweisen eines Objektes zum nächsten Klassenzentrum (z.B.: Euklidischer Abstand),
  3) Aufgrund der zugewiesenen Punkt Klassenzentren neu berechnen,
  4) Zurück zu 2) solange bis Klassenzentren konstant bleiben.

- Algorithmen:
  - k-means Clustering (Kriterium: Least Squared Distance)
  - Expectation-Maximization (EM) Algorithmus (Kriterium: Maximum Likelihood)

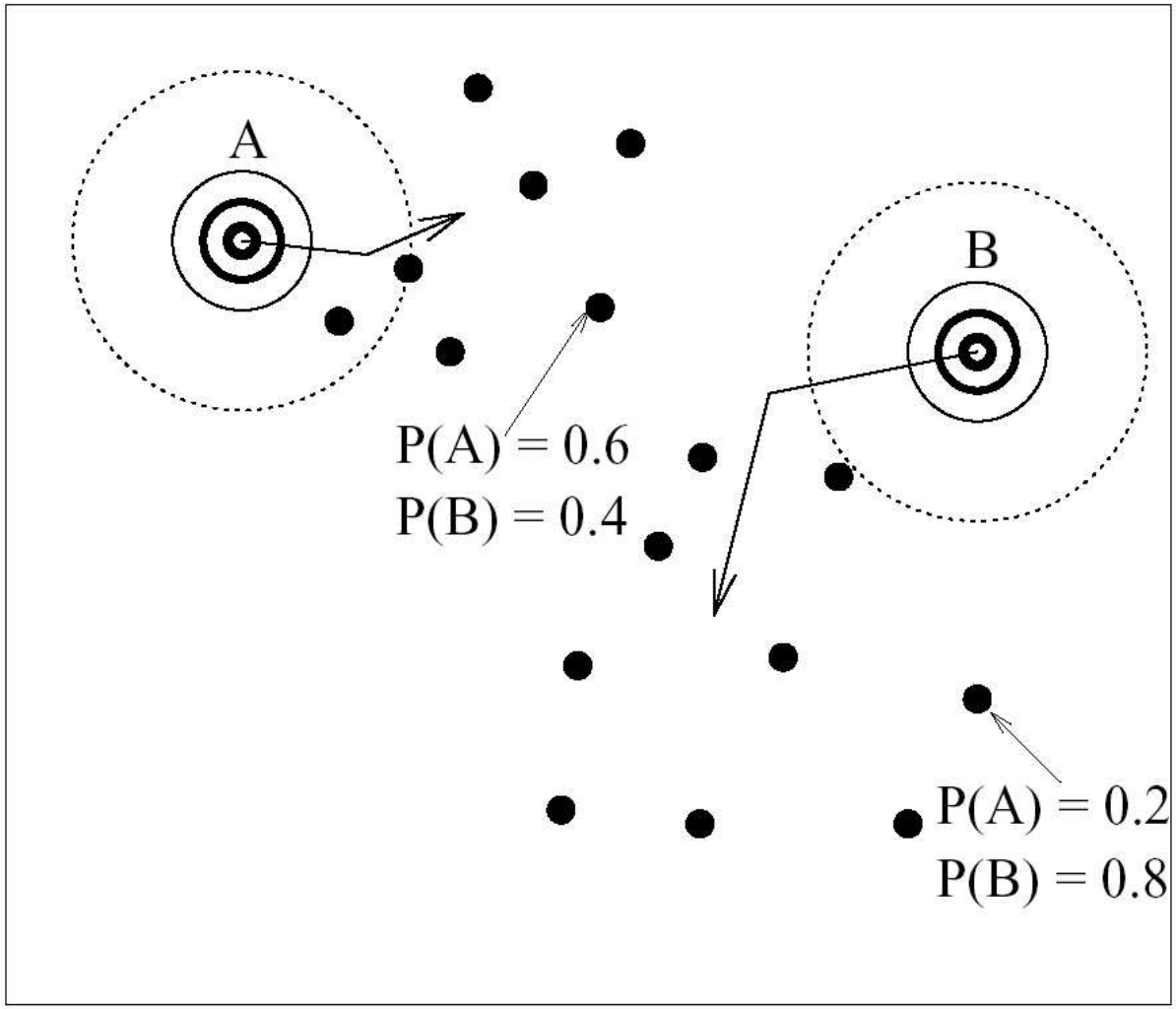
Hidden variable: for each point, which Gaussian generated it?



E-Step: for each point, estimate the probability the each Gaussian generated it



M-Step: modify the parameters according to the hidden variable to maximize the likelihood of the data (and the hidden variable)

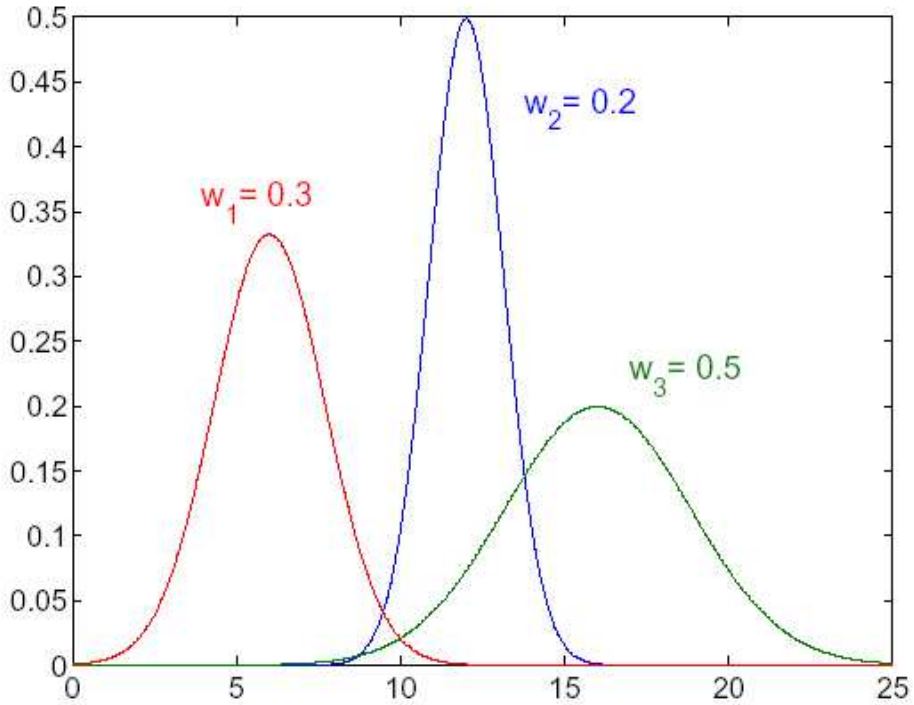


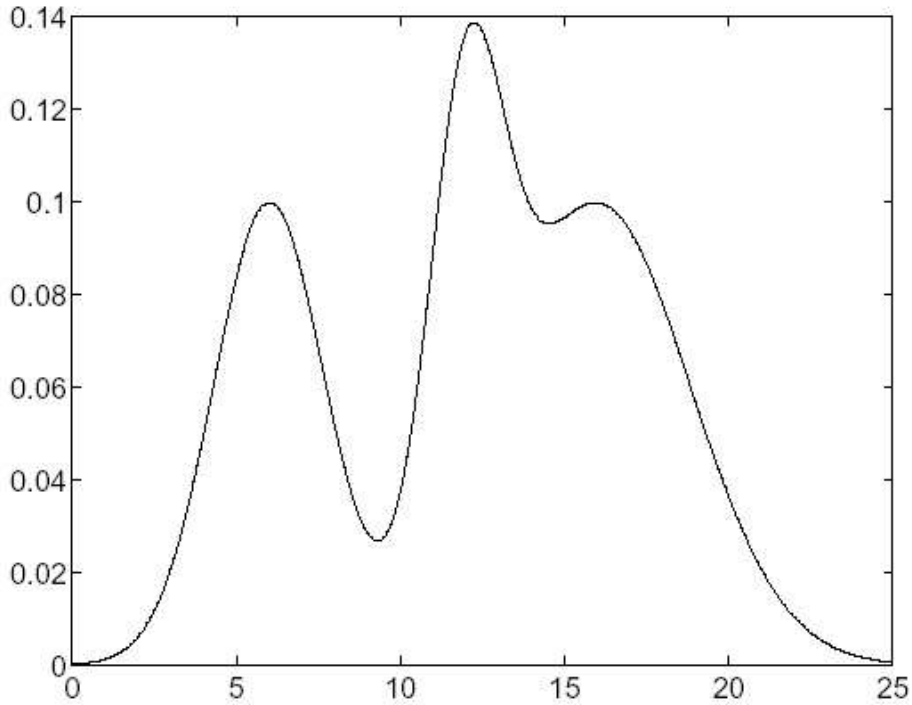
- Objective: maximize the likelihood $p(X;\theta)$ of the data X drawn from an unknown distribution, given the model parameterized by θ:

$$\theta^* = \arg\max_\theta p(X|\theta) = \arg\max_\theta \prod_{p=1}^{n} p(x_p|\theta)$$

- Basic ideas of EM:
  - Introduce a hidden variable such that *its knowledge would simplify the maximization of* $p(X;\theta)$
  - At each iteration of the algorithm:
    - E-Step: estimate the distribution of the hidden variable given the data and the current value of the parameters
    - M-Step: modify the parameters in order to maximize the joint distribution of the data and the hidden variable

- Zur Modellierung von nicht uni-modalen Verteilungen
- Gaussian Mixture = Summe mehrerer Gauss-Verteilungen





- A Gaussian Mixture Model (GMM) is a distribution
- The likelihood given a Gaussian distribution is

$$\mathcal{N}(x;\mu,\Sigma) = \frac{1}{(2\pi)^{\frac{|x|}{2}}\sqrt{|\Sigma|}} \exp\left(-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)\right)$$

  where μ is the mean and Σ is the covariance matrix of the Gaussian. Σ is often diagonal.

- The likelihood given a GMM is

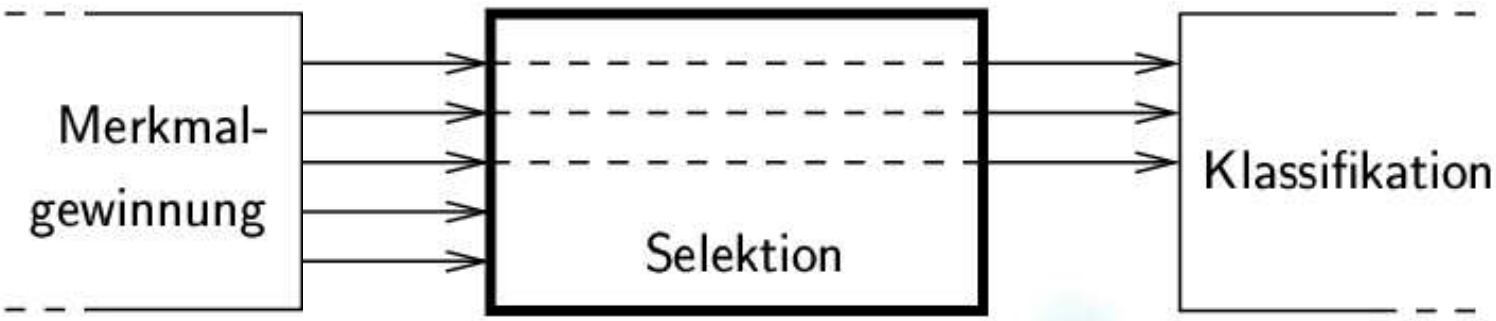
$$p(x) = \sum_{i=1}^{N} w_i \cdot \mathcal{N}(x;\mu_i,\Sigma_i)$$

  where N is the number of Gaussians and w_i is the weight of Gaussian i, with

$$\sum_i w_i = 1 \text{ and } \forall i : w_i \geq 0$$

Feature Selection

- Reduction of the number of features to a set of a few significant ones which optimize the classification performance.



- Basic steps of a *genetic algorithm* for automatic selection of the "best" features:

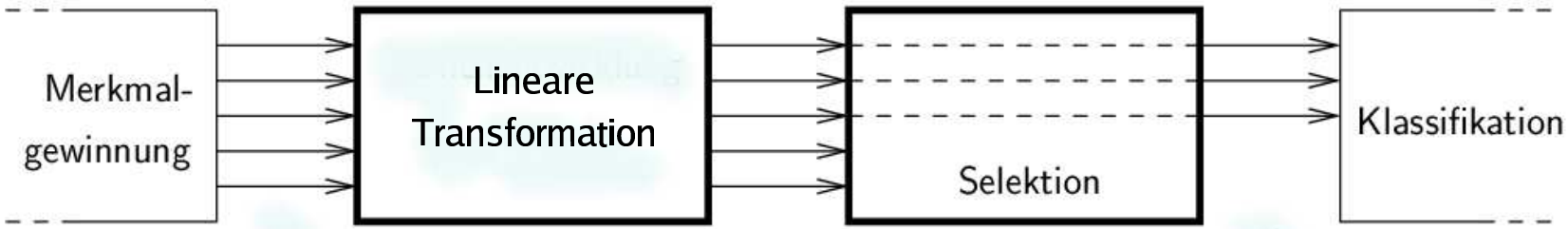
1. A generation procedure to generate the next subset of features X.
2. An evaluation criterion J to evaluate the quality of X.
3. A stopping criterion for concluding the search. In can either be based on the generation procedure or on the evaluation function.
4. A validation procedure for verifying the validity of the selected subset.

Feature Selection: Merkmalraumtransformation

- Reduktion der Vektordimension
  - Geringerer Klassifikationsaufwand
  - Robustere Schätzung statistischer Parameter

Lineare Transformationsvorschrift

$$\varphi : \begin{cases} \mathbb{R}^D & \rightarrow & \mathbb{R}^d \\ \boldsymbol{x} & \mapsto & \boldsymbol{y} = \varphi(\boldsymbol{x}) = \boldsymbol{\Phi}^\top \boldsymbol{x} \end{cases} \qquad \text{mit } d \leq D$$



- Gütekriterien für die Transformation: $\varphi(\cdot)$
  - Varianzmaximierung $\Rightarrow$ Hauptkomponentenanalyse (PCA, Karhunen-Loève Transform)
  - Klassenseparation $\Rightarrow$ Lineare Diskriminanzanalyse (LDA)

Feature Selection: Normierung der Merkmale

- Kompensieren unterschiedlicher Bereiche von Merkmalen. Um im Merkmalsraum Distanzmaße zum Klassifizieren verwenden zu können
- Musterraum

$$\begin{bmatrix} x_{11} & \cdots & x_{1j} & \cdots & x_{1M} \\ \vdots & & \vdots & & \vdots \\ x_{i1} & \cdots & x_{ij} & \cdots & x_{iM} \\ \vdots & & \vdots & & \vdots \\ x_{N1} & \cdots & x_{Nj} & \cdots & x_{NM} \end{bmatrix}$$

- Mittelwert für jede Spalte

$$m(j) = \frac{1}{N} \sum_{i=1}^{N} x_{ij}$$

- Standardabweichung für jede Spalte

$$s(j) = \sqrt{\frac{1}{N-1} \sum_{i=1}^{N} (x_{ij} - m(j))^2}$$

Feature Selection: Normierung der Merkmale

- Mean absolute deviation (Einfluss von Outliern wird unterdrückt)

$$\bar{s}(j) = \frac{1}{N} \sum_{i=1}^{N} |x_{ij} - m(j)|$$

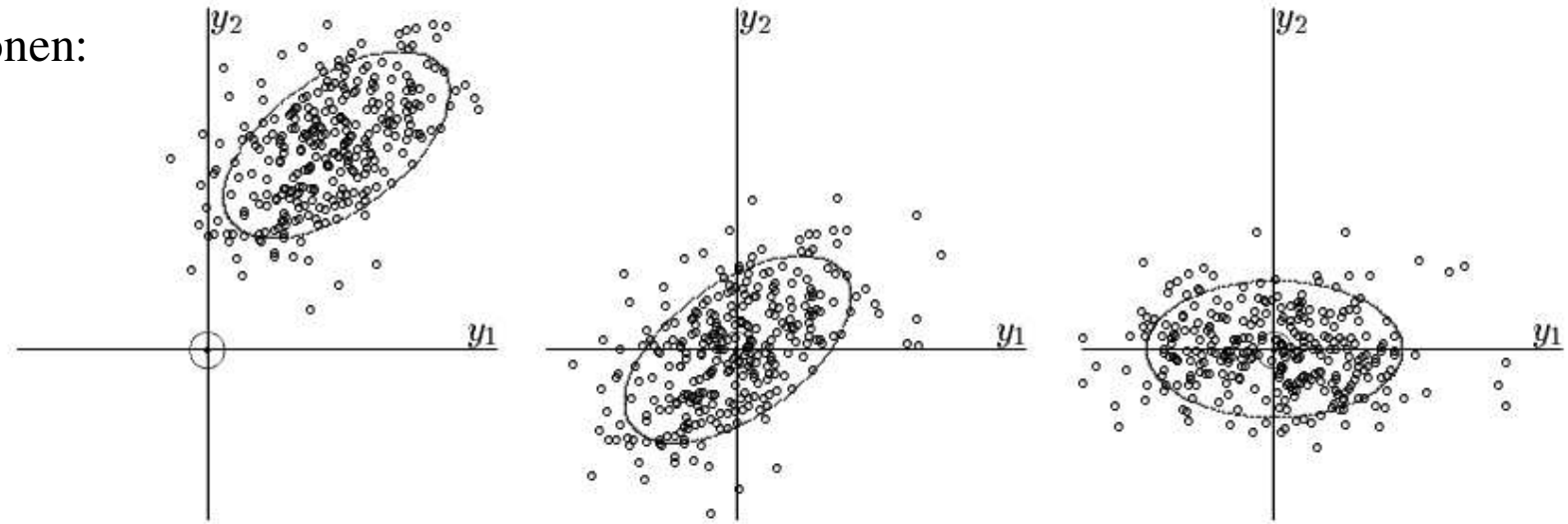
- Normierung: The following normalization of the feature values x_{ij}

$$z_{ij} = \frac{x_{ij} - m(j)}{\bar{s}(j)}$$

yields feature values z_{ij} with $m(j) = 0$ and $\bar{s}(j) = 1$.

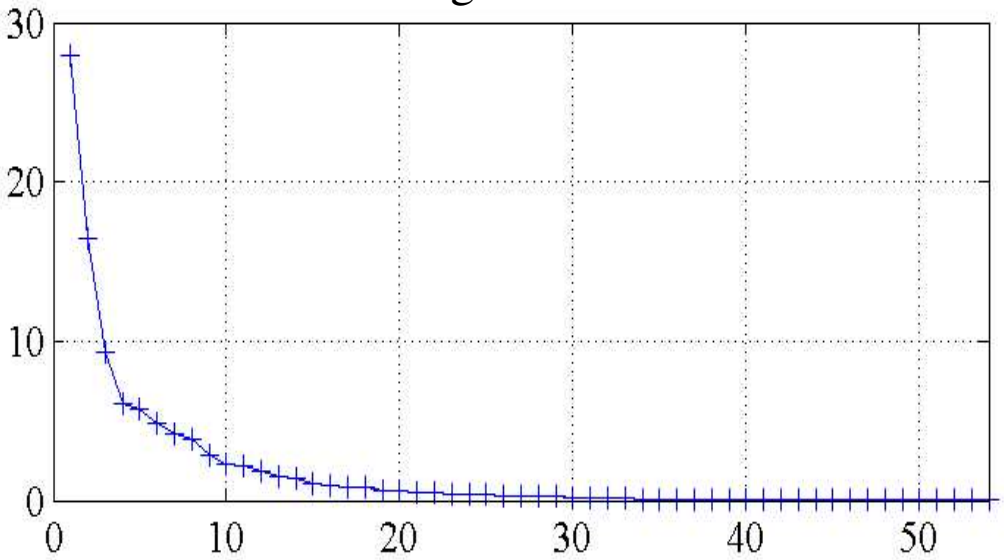
Feature Selection: Hauptkomponentenanalyse

2 Dimensionen:

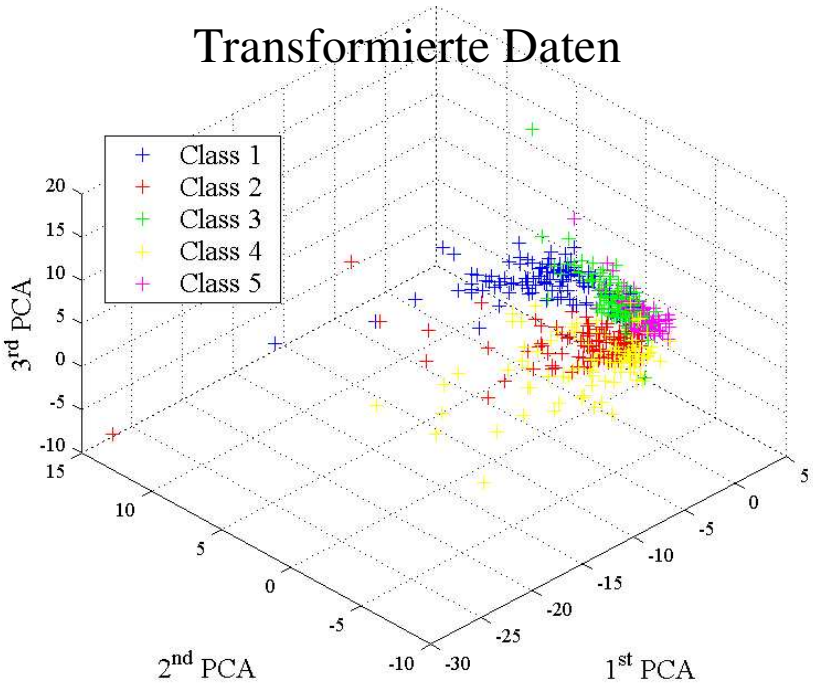


54 Dimensionen:

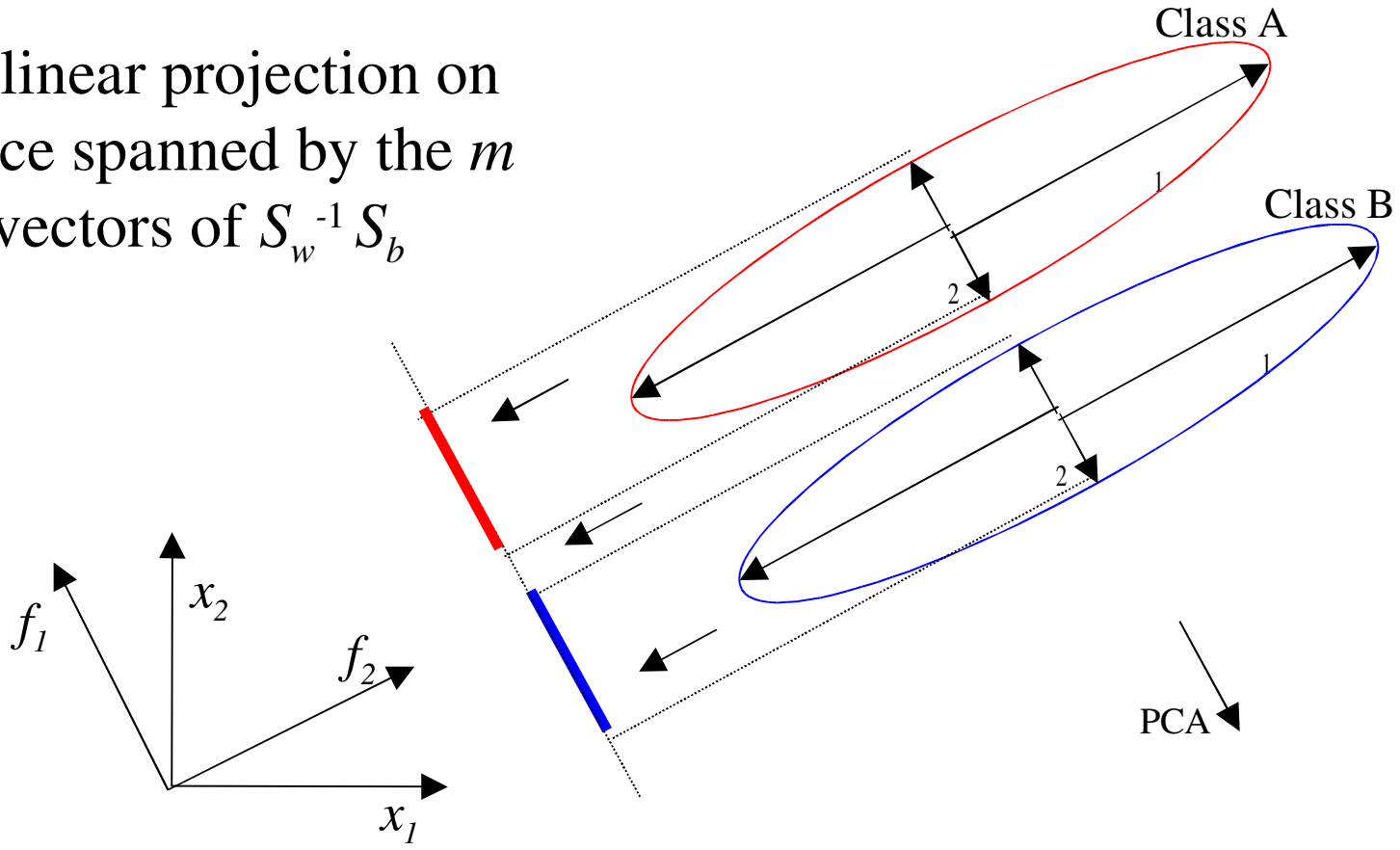
Eigenwerte



Transformierte Daten



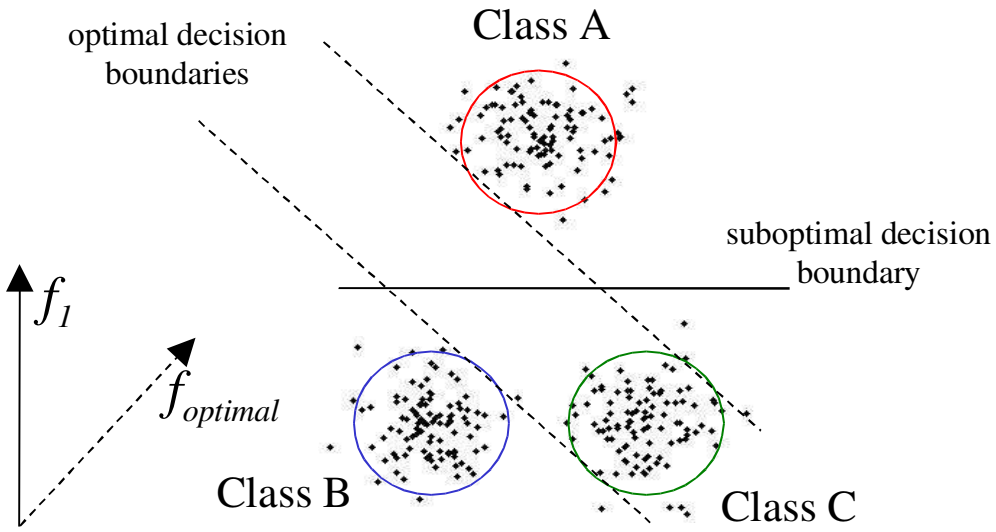
LDA finds a linear projection on to the subspace spanned by the m largest eigenvectors of $S_w^{-1} S_b$

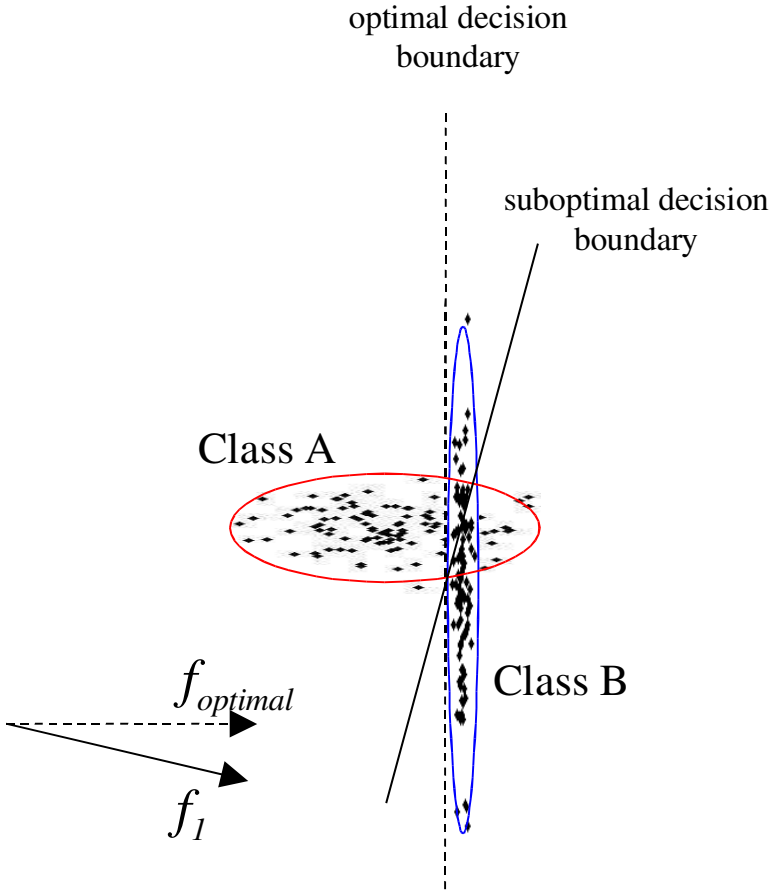


Optimal for m+1 linear separable classes c

Feature Selection: Limitations of LDA

- Suboptimal for more than m+1 classes
  (m is the number of extracted features)
- Classes should be linear separable





Referenzen

- R.O. Duda and P.E. Hart, *Pattern Classification and Scene Analysis*. Wiley&Sons, Inc., 1973.
- R. Bolter, *Bildverarbeitung und Mustererkennung,* Vorlesung ICG Graz, 2000.
- S. Bengio, *An Introduction to Statistical Machine Learning – EM for GMMs,* Dalle Molle Institute for Perceptual Artificial Intelligence.
- E.G. Schukat-Talamazzini, *Automatische Spracherkennung,* Vieweg-Verlag, 1995.
- F. Pernkopf, *Automatic Visual Inspection of Metallic Surfaces,* PhD Thesis, Leoben 2002.
- C.R. Houck, J.A. Joines, G.M. Kay, *A Genetic Algorithm for Function Optimization: A Matlab Implementation,* North Carolina State University.
- M. Obikito, *Introduction to Genetic Algorithms,* Hochschule für Technik und Wirtschaft Dresden, 1998.