Lecture

Speech and Audio Signal Processing

Lecture 9: Bayes decision methods and Gaussian mixture models

TECHNISCHE UNIVERSITÄT DARMSTADT

Content

TECHNISCHE UNIVERSITÄT DARMSTADT

- *Introduction:* From feature values to decisions / detections
- Basics of the Bayes decision theory
- Scalar and multi-dimensional Gaussians models
- Detection / classification: Motivation for…
- … Gaussian mixture models (GMMs)
- GMM parameter estimation based on training data:
  => Iterative procedure
- GMM comparison with codebooks
- Alternative methods, e.g., decision trees, k-nearest neighbors, LSVM, DNN / MLPs
- Application on speaker detection

Detection / decision theory: The basic understanding

TECHNISCHE UNIVERSITÄT DARMSTADT

- **Basis for a decision:** Feature value (scalar feature) or several feature values, combined in a vector (feature vector).

- **One Example** for a decision based on a **scalar feature value,** $x_0(n)$:
  - **Voiced / unvoiced decision** based on the **feature value "total frame energy"** (s. lecture 8)

$$pow(n) = \sum_{n_0=n}^{n+L-1} x^2(n_0)$$

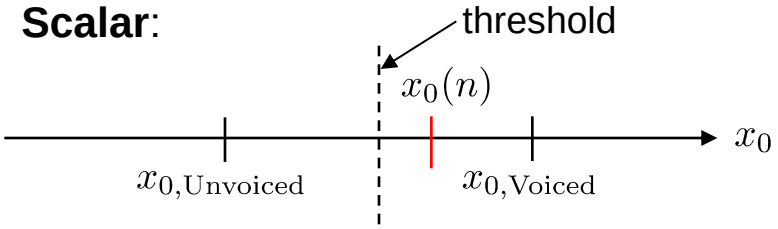
- **One Example** for a decision based on a **feature vector,** $\boldsymbol{x}(n)$ **:**
  - Speaker recognition based on the MFCC vector: $\boldsymbol{y}_{\text{mfcc}}(n)$

$$\boldsymbol{y}_{\text{abs}}(n) = \left[\left|Y(e^{j\Omega_0}, n)\right|, ..., \left|Y(e^{j\Omega_{N-1}}, n)\right|\right]^{\text{T}} \qquad \boldsymbol{y}_{\text{log}}(n) = \log_e\{\boldsymbol{y}_{\text{mel}}(n)\}$$

$$\boldsymbol{y}_{\text{mel}}(n) = \boldsymbol{M}\,\boldsymbol{y}_{\text{abs}}(n) \qquad \boldsymbol{y}_{\text{mfcc}}(n) = \boldsymbol{P}\,\boldsymbol{T}_{\cos}\boldsymbol{y}_{\text{log}}(n)$$

Two binary decision principles: 1) mean value

- Based on the **mean feature values** for each hypothesis:
  - **Scalar**:

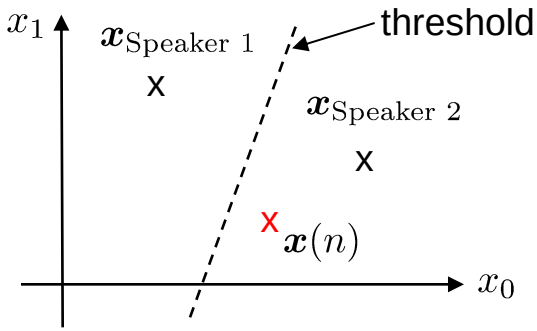


$x_{0,\mathrm{Unvoiced}}$: mean values for each
$x_{0,\mathrm{Voiced}}$ hypothesis determined based on training data

Detection for the hypothesis whose mean value has the lower distance to the current feature value:

$$|x_0(n) - x_{0,\mathrm{Unvoiced}}| \underset{\text{Unvoiced}}{\overset{\text{Voiced}}{\gtrless}} |x_0(n) - x_{0,\mathrm{Voiced}}|$$

  - **Vector** (s. vector quantization):



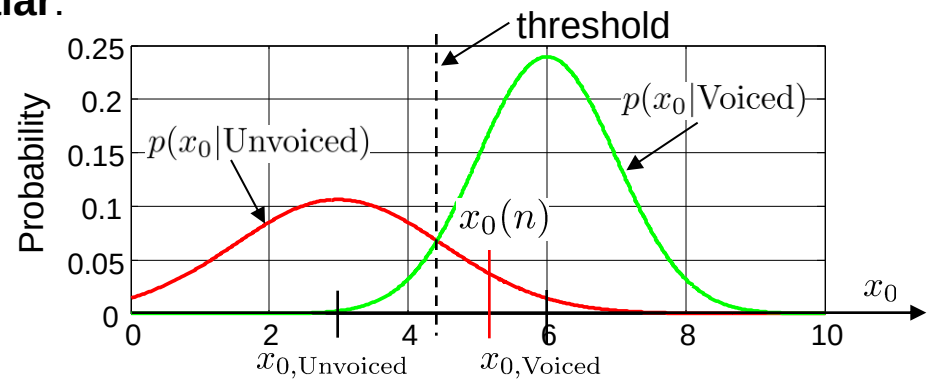
$\boldsymbol{x} = [x_0, x_1]^{\mathrm{T}}$
$\boldsymbol{x}_{\mathrm{Speaker\ 1}}$: mean vectors for each
$\boldsymbol{x}_{\mathrm{Speaker\ 2}}$ hypothesis determined based on training data

Detection based on the distance:

$$\|\boldsymbol{x}(n) - \boldsymbol{x}_{\mathrm{Speaker\ 1}}\| \underset{\text{Speaker 1}}{\overset{\text{Speaker 2}}{\gtrless}} \|\boldsymbol{x}(n) - \boldsymbol{x}_{\mathrm{Speaker\ 2}}\|$$

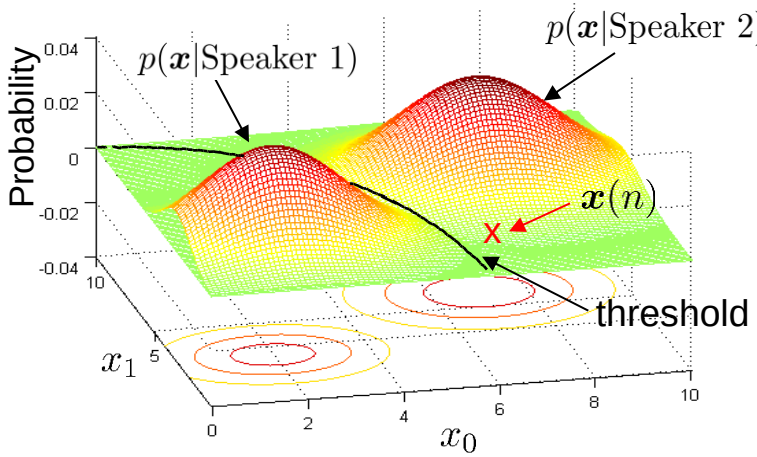
Two binary decision principles: 2) probability density (pdf)

- Based on the **probability density** for each hypothesis:
  - **Scalar**:



Detection based on the probabilities of the feature values for each hypothesis

  - **Vector**:



Equivalent detection principle

Bayes decision theory

- Decision for a hypothesis H_i with: $i \in [1, \ldots, M]$

  based on a feature vector $\boldsymbol{x} = [x_0, x_1, \ldots, x_{D-1}]^{\mathrm{T}}$

  with the *largest* **a posteriori** probability: *MAP (max. a posteriori) detection*,

  $$p(H_i|\boldsymbol{x})$$

  which is the probability for the “class” / “hypothesis” H_i **after** observing the feature vector $\boldsymbol{x} = [x_0, x_1, \ldots, x_{D-1}]^{\mathrm{T}}$

- The *a posteriori* probability can be written – according to Bayes – as follows:

$$p(H_i|\boldsymbol{x}) = \frac{p(\boldsymbol{x}|H_i)\,p(H_i)}{p(\boldsymbol{x})} \quad \text{with:} \quad p(\boldsymbol{x}) = \sum_{i=1}^{M} p(\boldsymbol{x}|H_i)\,p(H_i)$$

Bayes decision theory

- Decision for the largest a posteriori probability $p(H_i|\boldsymbol{x})$

$$\begin{aligned} H_d &= \arg\max_i p(H_i|\boldsymbol{x}) \\ &= \arg\max_i \left[p(\boldsymbol{x}|H_i)\, p(H_i)\right] \end{aligned}$$

- In case of a two-class (binary) decision $M = 2$ one can write:

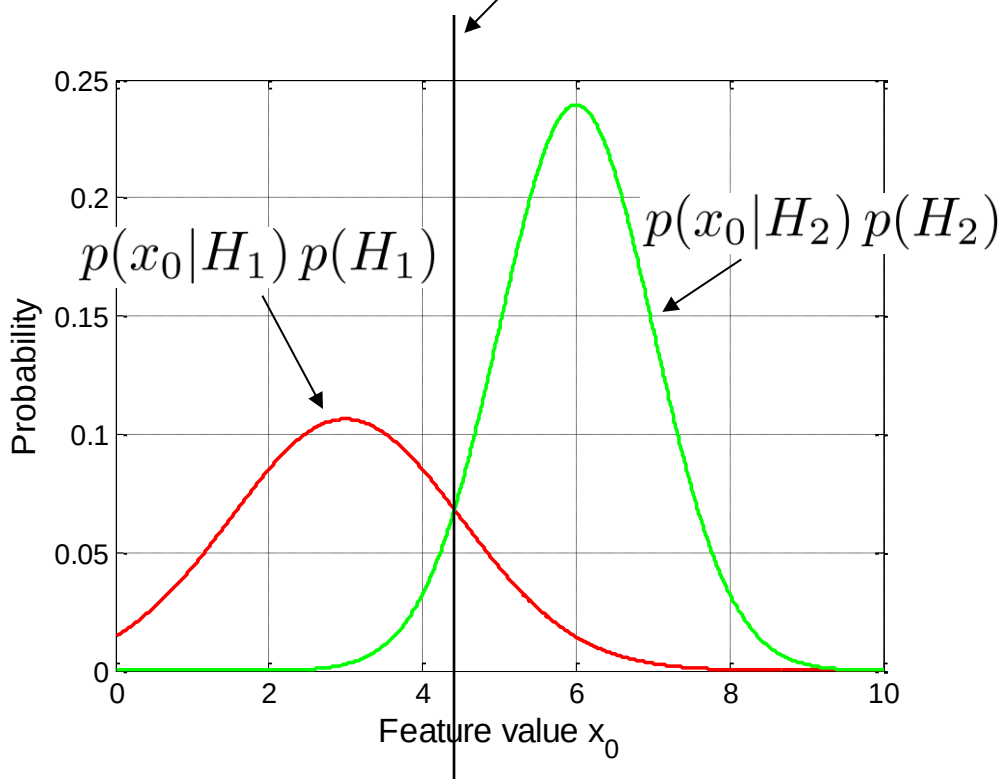
$$p(\boldsymbol{x}|H_1)\, p(H_1) \underset{H_2}{\overset{H_1}{\gtrless}} p(\boldsymbol{x}|H_2)\, p(H_2)$$

Bayes decision theory

- Scalar feature value:

Decision bound:

$$p(x_0|H_1)\,p(H_1) \underset{H_2}{\overset{H_1}{\gtrless}} p(x_0|H_2)\,p(H_2)$$



Bayes decision theory

TECHNISCHE UNIVERSITÄT DARMSTADT

- Likelihood ratio:

$$l(\boldsymbol{x}) = \frac{p(\boldsymbol{x}|H_1)\,p(H_1)}{p(\boldsymbol{x}|H_2)\,p(H_2)} \begin{array}{c} H_1 \\ > \\ < \\ H_2 \end{array} 1$$

- Log Likelihood ratio:

$$llr(\boldsymbol{x}) = \log p(\boldsymbol{x}|H_1) - \log p(\boldsymbol{x}|H_2) + \log\, p(H_1) - \log\, p(H_2) \begin{array}{c} H_1 \\ > \\ < \\ H_2 \end{array} 0$$

- Discriminant functions:

$$g_i(\boldsymbol{x}) = \log p(\boldsymbol{x}|H_i) + \log\, p(H_i)$$

Gaussian models

- **Discriminant functions for Gaussian distributions:**

$$g_i(\boldsymbol{x}) = \log p(\boldsymbol{x}|H_i) + \log\, p(H_i)$$

For scalar values: $p(x_0|H_i) = \dfrac{1}{\sigma_i\sqrt{2\pi}}\, e^{-\frac{(x_0-\mu_i)^2}{2\sigma_i^2}}$

$$g_i(x_0) = -\log(\sigma_i\sqrt{2\pi}) - \frac{(x_0-\mu_i)^2}{2\sigma_i^2} + \log p(H_i)$$

In general, this is a non-linear function.

- **Log likelihood ratio:** $llr(\boldsymbol{x}) = g_1(\boldsymbol{x}) - g_2(\boldsymbol{x})$

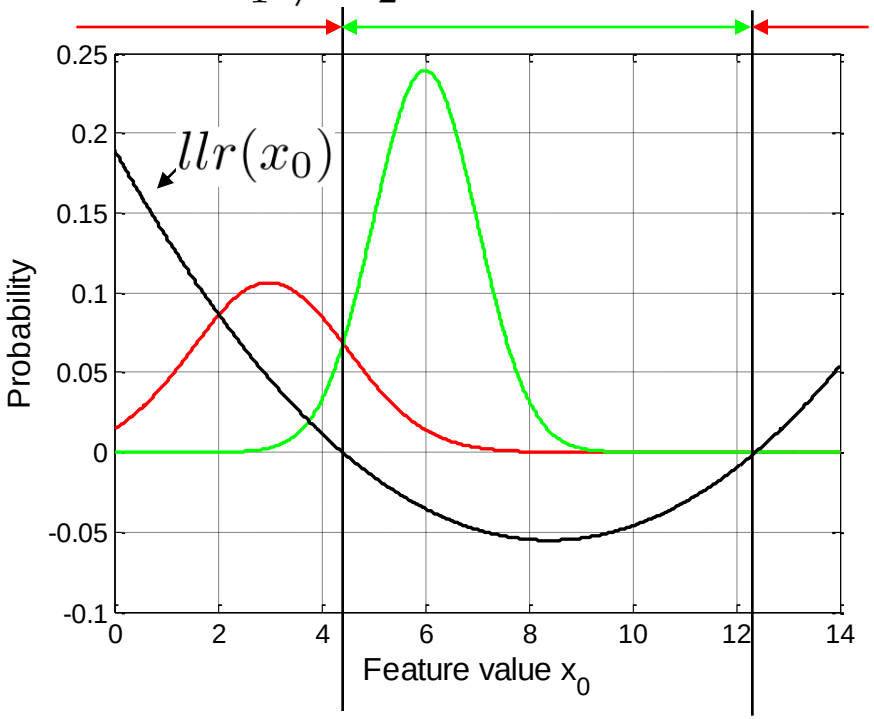
For Gaussian distributions with $\sigma = \sigma_1 = \sigma_2$ one can formulate a modified log likelihood ratio which is linear:

$$llr(x_0) = (\mu_1 - \mu_2)\left[x_0 - \left(\frac{\mu_1+\mu_2}{2} - \frac{\sigma^2}{\mu_1-\mu_2}\log\frac{p(H_1)}{p(H_2)}\right)\right]$$

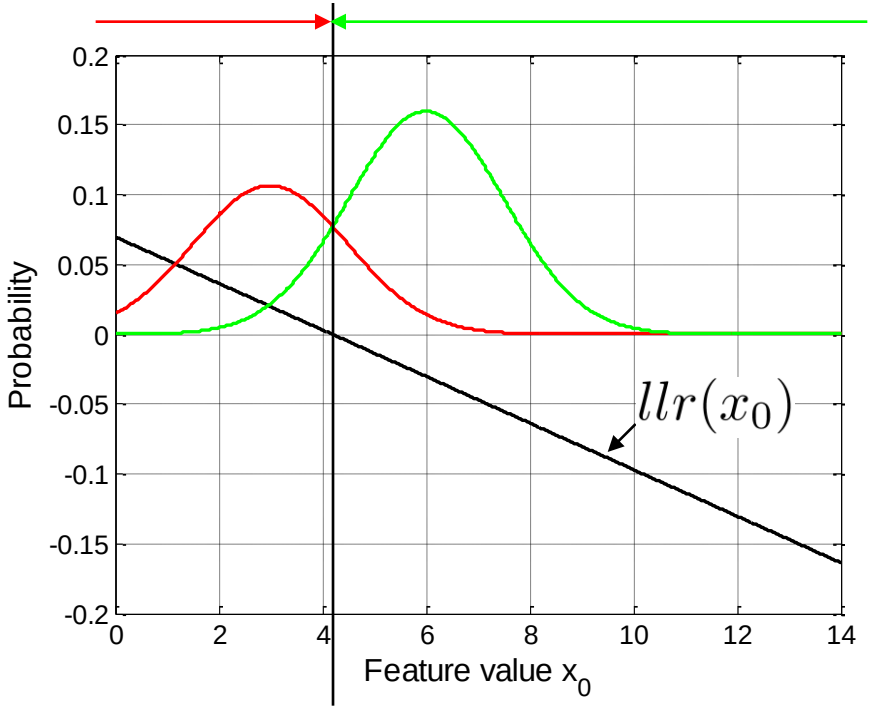
Gaussian models

- Log likelihood ratio: $llr(x_0) = g_1(x_0) - g_2(x_0)$

A posteriori probabilities and llr's for $\sigma_1 \neq \sigma_2$



A posteriori probabilities and llr's for $\sigma = \sigma_1 = \sigma_2$



Multidimensional Gaussian models

TECHNISCHE UNIVERSITÄT DARMSTADT

- Gaussian probability density function (pdf) **for feature vectors**:

$$p(\boldsymbol{x}|H_i) = \frac{1}{\sqrt{(2\pi)^D \, |\boldsymbol{\Sigma}_i|}} \, e^{-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu}_i)^{\mathrm{T}} \, \boldsymbol{\Sigma}_i^{-1} (\boldsymbol{x}-\boldsymbol{\mu}_i)}$$

with: $\boldsymbol{\mu}_i = \begin{bmatrix} \mu_{0,i} \\ \mu_{1,i} \\ \vdots \\ \mu_{D-1,i} \end{bmatrix}$ $\boldsymbol{\Sigma}_i = \begin{bmatrix} \sigma_{0,i}^2 & \sigma_{1\,0,i} & \cdots & \sigma_{D-1\,0,i} \\ \sigma_{0\,1,i} & \sigma_{1,i}^2 & \cdots & \sigma_{D-1\,1,i} \\ \vdots & \vdots & \ddots & \vdots \\ \sigma_{0\,D-1,i} & \sigma_{1\,D-1,i} & \cdots & \sigma_{D-1,i}^2 \end{bmatrix}$

$|\boldsymbol{\Sigma}_i|$: determinant

- With:

$$g_i(\boldsymbol{x}) = \log p(\boldsymbol{x}|H_i) + \log \, p(H_i)$$

- one obtains:

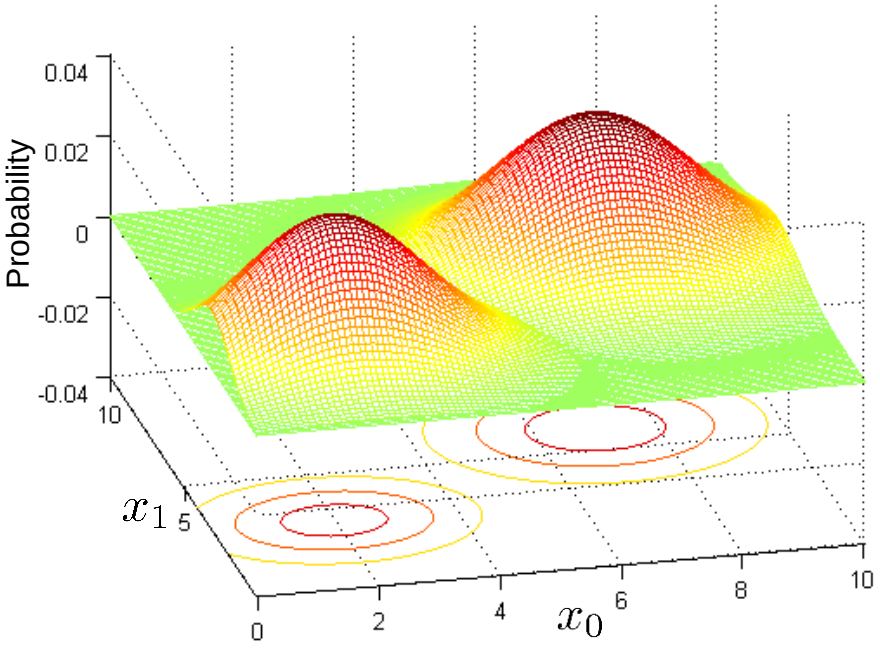
$$g_i(\boldsymbol{x}) = -\frac{1}{2} \, \log(|\boldsymbol{\Sigma}_i| \, (2\pi)^D) - \frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu}_i)^{\mathrm{T}} \, \boldsymbol{\Sigma}_i^{-1} (\boldsymbol{x}-\boldsymbol{\mu}_i) + \log \, p(H_i)$$

$$llr(\boldsymbol{x}) = g_1(\boldsymbol{x}) - g_2(\boldsymbol{x})$$

Multidimensional Gaussian models

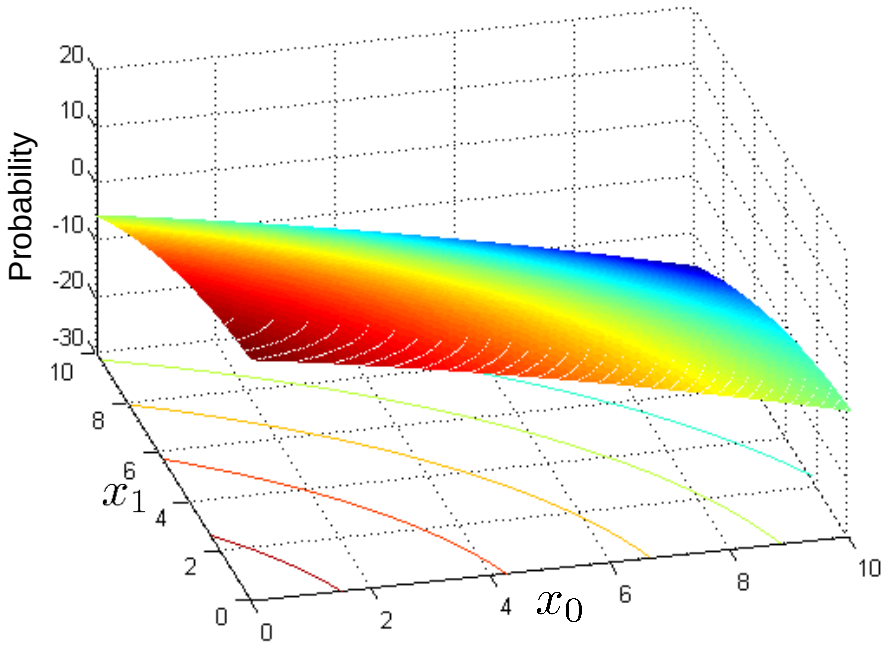
- In general, the *llr* is a non-linear function, resulting in a non-linear separation of the classifier:

- Gaussian pdfs:



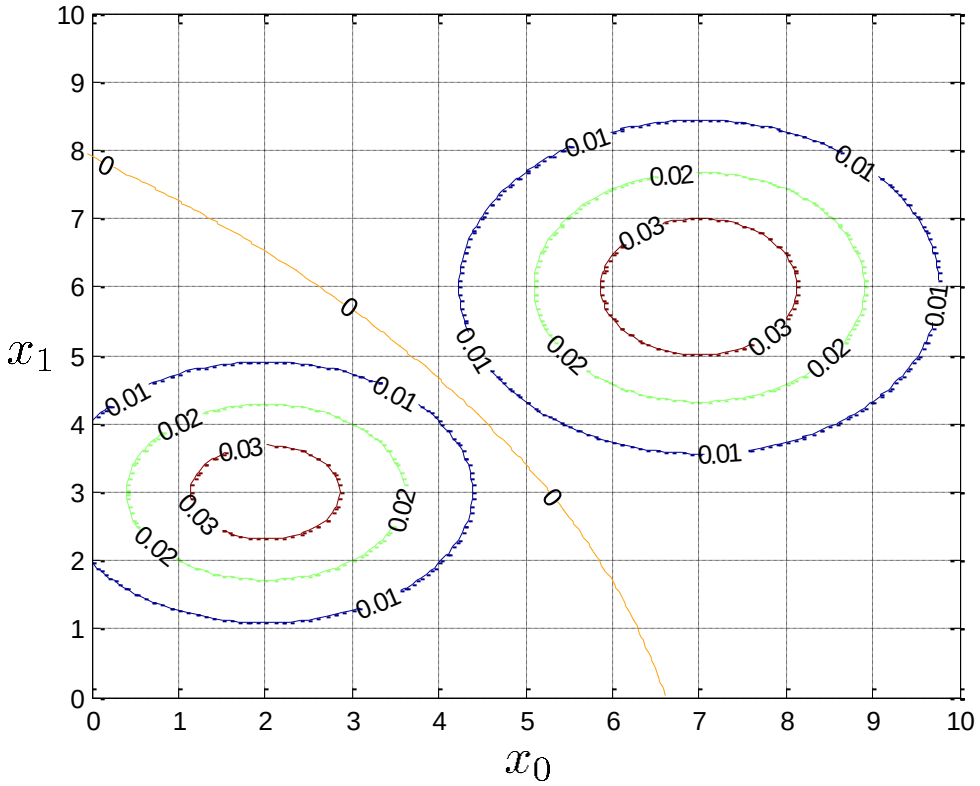
- Difference function (*llr*):

$$llr(\boldsymbol{x}) = g_1(\boldsymbol{x}) - g_2(\boldsymbol{x})$$



Multidimensional Gaussian models

- Non-linear separation between the Gaussian distributions:



The non-linear separation is a difference compared to separation in the case of vector quantization.

Multidimensional Gaussian models

- Linear separation between the Gaussian distributions in case of identical variance matrices: $\boldsymbol{\Sigma} = \boldsymbol{\Sigma}_1 = \boldsymbol{\Sigma}_2$

- With:

$$g_i(\boldsymbol{x}) = -\frac{1}{2}\log(|\boldsymbol{\Sigma}_i|\,(2\pi)^D) - \frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu}_i)^{\mathrm{T}}\,\boldsymbol{\Sigma}_i^{-1}(\boldsymbol{x}-\boldsymbol{\mu}_i) + \log\,p(H_i)$$

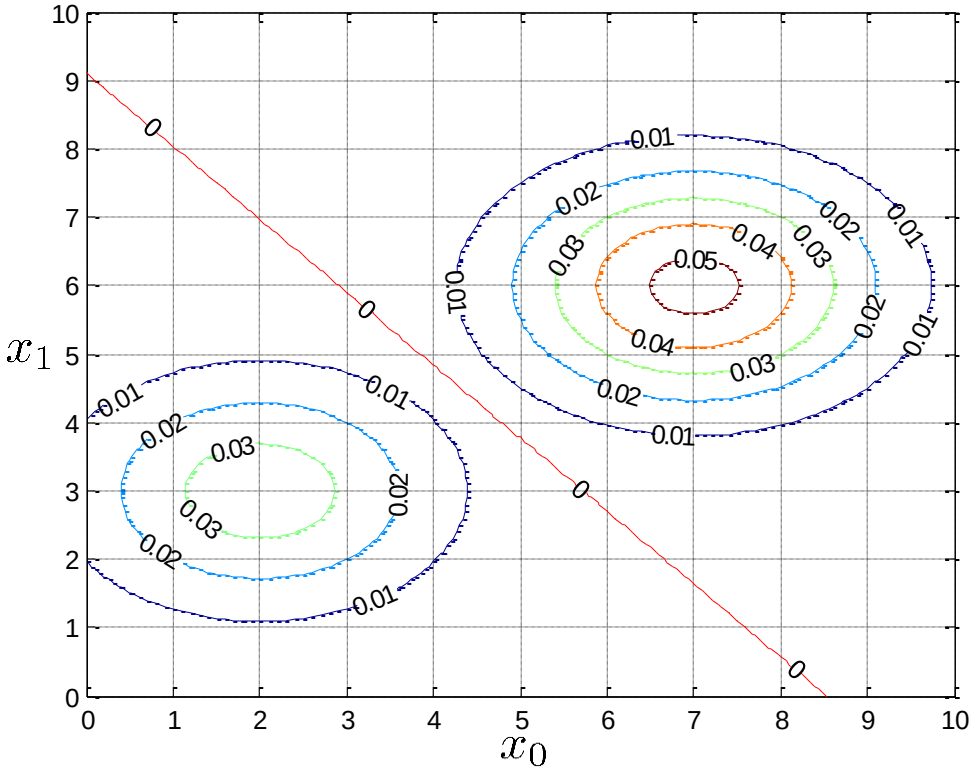
$$llr(\boldsymbol{x}) = g_1(\boldsymbol{x}) - g_2(\boldsymbol{x})$$

- one obtains a linear relation with ***x*** :

$$\begin{aligned} llr(\boldsymbol{x}) \quad = \quad & [\boldsymbol{\Sigma}^{-1}(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_1)]^{\mathrm{T}}\,\boldsymbol{x} \\ & + \left(-\frac{1}{2}\boldsymbol{\mu}_1^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_1 + \frac{1}{2}\boldsymbol{\mu}_2^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_2 + \log\,p(H_1) - \log\,p(H_2)\right) \end{aligned}$$

Multidimensional Gaussian models

- Linear separation between the Gaussian distributions with identical variances:



Separation line is a tangent to the contour curves.

In case of identical variances on the diagonal, the separation line is orthogonal to the connection line between the two Gaussian distributions.

Detection / classification

TECHNISCHE UNIVERSITÄT DARMSTADT

- In real applications the conditional distributions must be modeled for each class H_i:

$$p(\boldsymbol{x}|H_i)$$

- Also, the a priori probabilities $p(H_i)$ for each class should be known or have to be assumed. In case of no a priori knowledge they are chosen identically.

- In case of one feature (element) only, the conditional distributions may be estimated by histogram data.

- When no correlation between the features is considered, the conditional vector probability can be determined as follows:

$$p(\boldsymbol{x}|H_i) = \prod_{j=0}^{D-1} p(x_j|H_i)$$

Detection / classification

- However, typically the feature values are correlated and a multiplication is a too strong simplification.

$$p(\boldsymbol{x}|H_i) = \prod_{j=0}^{D-1} p(x_j|H_i)$$

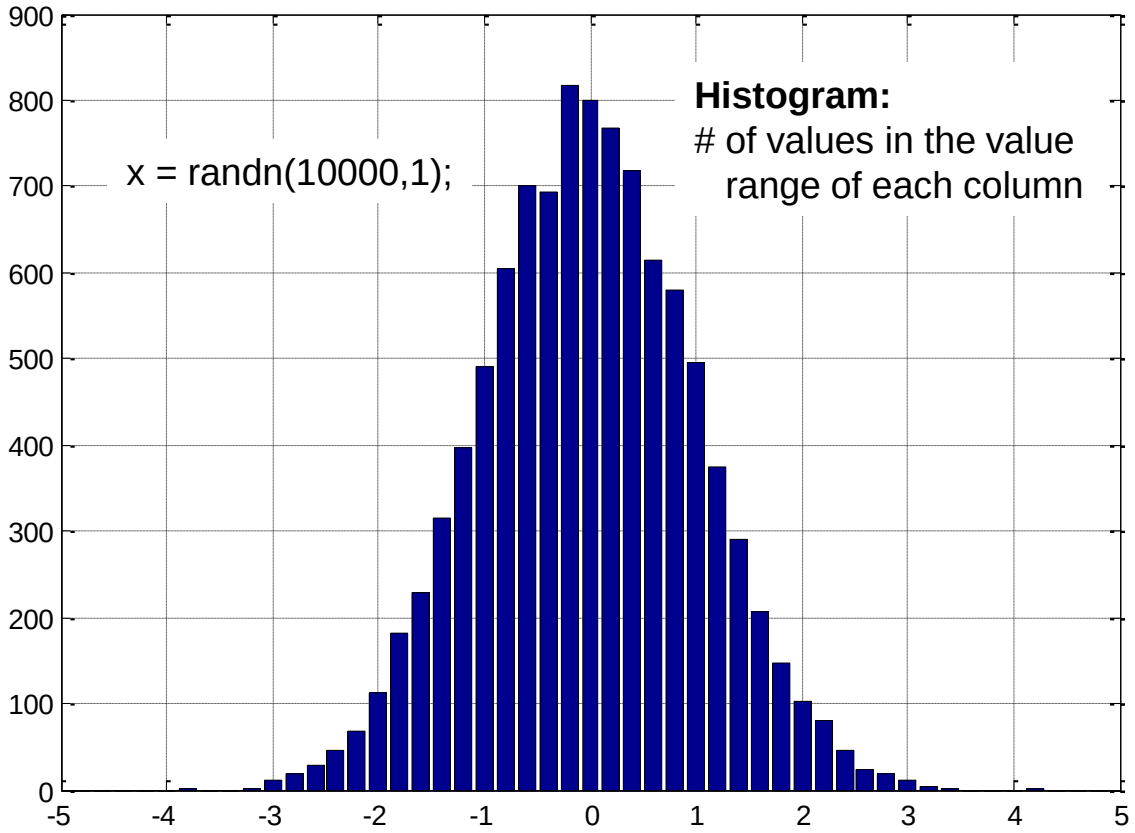
- In order to estimate the *D*-dimensional distributions, a number of data values at the power of *D* is necessary compared to an estimation of one dimension.

$$\tilde{N} = N^D$$

  This is the case if one wants to estimate the distributions with histograms.

Histogram of a Gaussian distribution

- Width of each column is a parameter of the histogram; here chosen as 0.2
- To generate an estimated pdf: Divide the histogram values by their widths multiplied with the number of random values => 0.2*10000 = 2000




```
y = -5:0.2:5;
z = hist(x,y);
```

Plot of a histogram:
=> bar(y,z)

Plot of the estimated pdf:
=> bar(y,z/2000);

Gaussian mixture models (GMM)

- An alternative is to model the joint distribution by Gaussian mixture models.
- The sum of several Gaussian distributions allows to approximate arbitrary distributions

$$\mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu},\boldsymbol{\Sigma}) = \frac{1}{\sqrt{(2\pi)^D\,|\boldsymbol{\Sigma}|}}\, e^{-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^{\mathrm{T}}\,\boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu})}$$

One multivariate Gaussian pdf

$$\boldsymbol{x} = [x_0, x_1, \ldots, x_{D-1}]^{\mathrm{T}} \quad \boldsymbol{\mu} = \begin{bmatrix} \mu_0 \\ \mu_1 \\ \vdots \\ \mu_{D-1} \end{bmatrix} \quad \boldsymbol{\Sigma} = \begin{bmatrix} \sigma_0^2 & \sigma_{1\,0} & \cdots & \sigma_{D-1\,0} \\ \sigma_{0\,1} & \sigma_1^2 & \cdots & \sigma_{D-1\,1} \\ \vdots & \vdots & \ddots & \vdots \\ \sigma_{0\,D-1} & \sigma_{1\,D-1} & \cdots & \sigma_{D-1}^2 \end{bmatrix}$$

$|\boldsymbol{\Sigma}|$: determinant

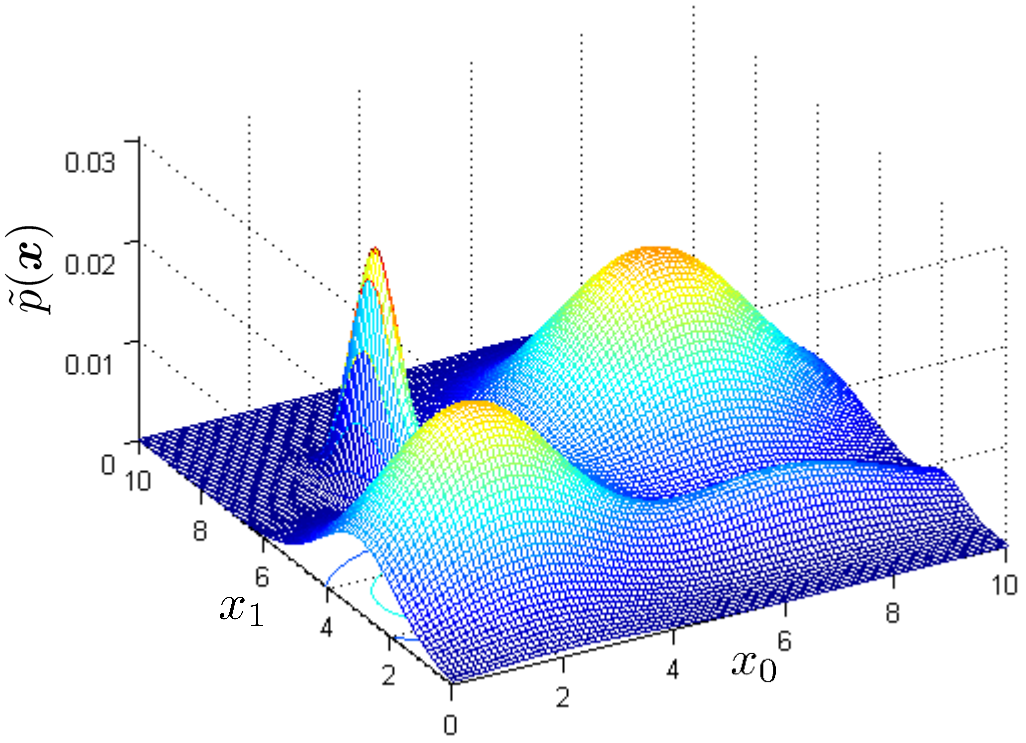
- Approximation of the probability density function (pdf) by a weighted sum of Gaussians (GMM):

$$p(\boldsymbol{x}|H_i) = \tilde{p}(\boldsymbol{x}) = \sum_{k=0}^{K-1} g_k\, \mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \qquad \text{with:} \quad \sum_{k=0}^{K-1} g_k = 1$$

Gaussian mixture models (GMM)

- For each class such a Gaussian mixture must be determined or trained.
- Compared to a D-dimensional histogram, severely less parameters must be determined: K weights, mean vectors and variance matrices.
- Example: Sum of K = 4 Gaussians

$$\tilde{p}(\boldsymbol{x}) = \sum_{k=0}^{3} g_k \, \mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$$

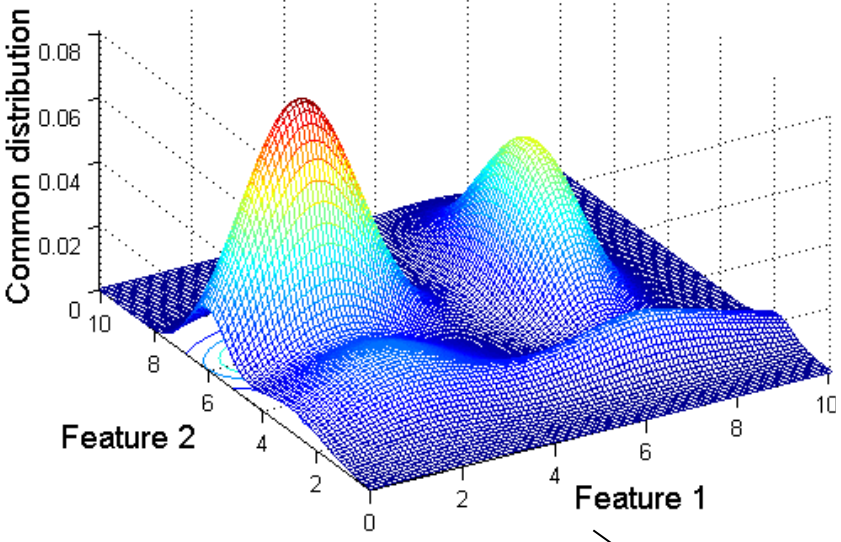


Gaussian mixture models (GMM)

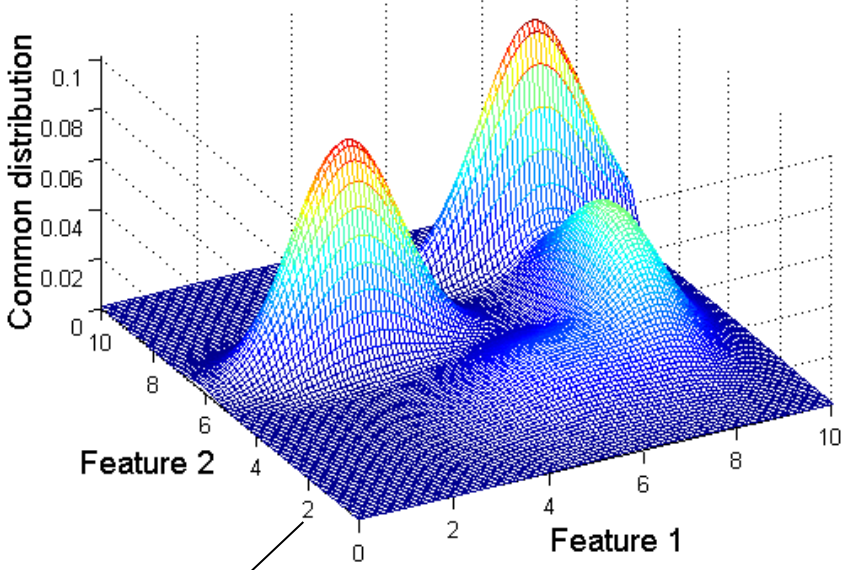
- Classification (i.e. select 1 of N), here N = 2.

Observed data:

Probability model trained with data for class 1:



Probability model trained with data for class 2:



Decision for the model with the higher probability.

GMM parameter estimation

TECHNISCHE UNIVERSITÄT DARMSTADT

- Assuming a training set of feature vectors:

$$\boldsymbol{X} = [\boldsymbol{x}(0), \boldsymbol{x}(1), \cdots, \boldsymbol{x}(N-1)]$$

- **Target:**
  Optimize / maximize the probability of the training set of feature vectors:

$$p(\boldsymbol{X}|\boldsymbol{g}, \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \prod_{n=0}^{N-1} \left[\sum_{k=0}^{K-1} g_k \, \mathcal{N}(\boldsymbol{x}(n)|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \right]$$

- Since the logarithm is a monotonic increasing function, the logarithm of the probability function can alternatively be optimized:

$$\log p(\boldsymbol{X}|\boldsymbol{g}, \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \sum_{n=0}^{N-1} \log \left\{ \sum_{k=0}^{K-1} g_k \, \mathcal{N}(\boldsymbol{x}(n)|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \right\}$$

GMM parameter estimation

- Assuming a current / initial model set is available which has to optimized, an “affiliation” value for each of the Gaussian distributions can determined:
  Affiliation value: Relative contribution of the k-th Gaussian to the pdf.

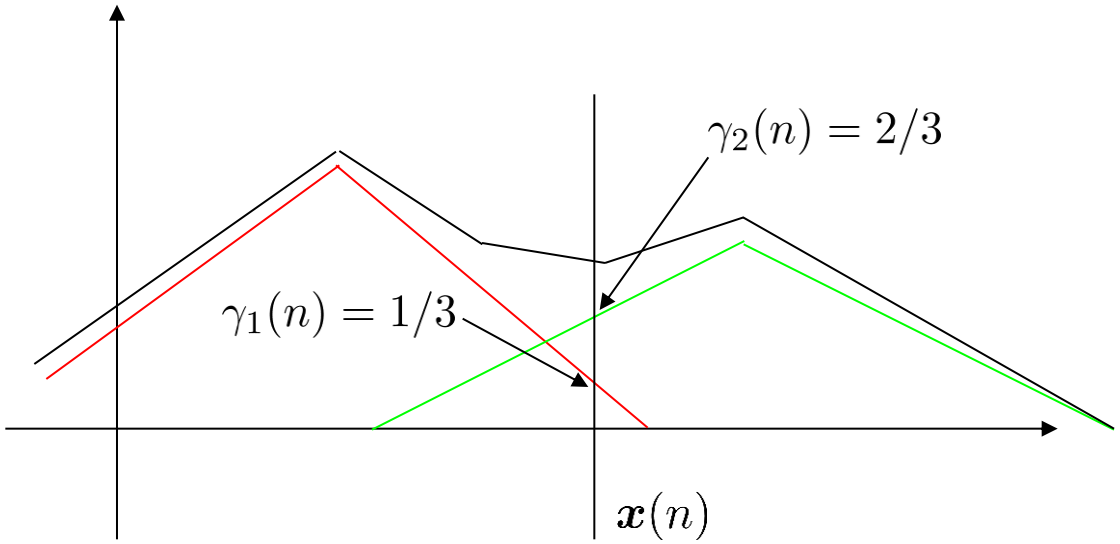
$$\gamma_k(n) = \frac{g_k \, \mathcal{N}(\boldsymbol{x}(n)|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{j=0}^{K-1} g_j \, \mathcal{N}(\boldsymbol{x}(n)|\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)} \qquad \text{with:} \quad \sum_{k=0}^{K-1} \gamma_k(n) = 1$$

- Derivation of the cost function with respect to the mean values:

$$\frac{d}{d\boldsymbol{\mu}_k} \log p(\boldsymbol{X}|\boldsymbol{g}, \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \frac{d}{d\boldsymbol{\mu}_k} \sum_{n=0}^{N-1} \log \left\{ \sum_{l=0}^{K-1} g_l \, \mathcal{N}(\boldsymbol{x}(n)|\boldsymbol{\mu}_l, \boldsymbol{\Sigma}_l) \right\}$$

Affiliation

- Example with two Gaussians modelling a pdf.
- At the value $\boldsymbol{x}(n)$ the first Gaussian contributes relatively 1/3 and the second 2/3.



GMM parameter estimation

- Setting the derivation to zero:

$$
\begin{aligned}
0 &= \frac{d}{d\boldsymbol{\mu}_k} \sum_{n=0}^{N-1} \log \left\{ \sum_{l=0}^{K-1} g_l \, \mathcal{N}(\boldsymbol{x}(n)|\boldsymbol{\mu}_l, \boldsymbol{\Sigma}_l) \right\} \\
&= \sum_{n=0}^{N-1} \frac{g_k \, \frac{d}{d\boldsymbol{\mu}_k} \mathcal{N}(\boldsymbol{x}(n)|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{l=0}^{K-1} g_l \, \mathcal{N}(\boldsymbol{x}(n)|\boldsymbol{\mu}_l, \boldsymbol{\Sigma}_l)} \\
&= \sum_{n=0}^{N-1} \frac{g_k \, \mathcal{N}(\boldsymbol{x}(n)|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{l=0}^{K-1} g_l \, \mathcal{N}(\boldsymbol{x}(n)|\boldsymbol{\mu}_l, \boldsymbol{\Sigma}_l)} \boldsymbol{\Sigma}_k^{-1} \left(\boldsymbol{x}(n) - \boldsymbol{\mu}_k\right)
\end{aligned}
$$

GMM parameter estimation

- Current result:

$$0 = \sum_{n=0}^{N-1} \frac{g_k \, \mathcal{N}(\boldsymbol{x}(n)|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{l=0}^{K-1} g_l \, \mathcal{N}(\boldsymbol{x}(n)|\boldsymbol{\mu}_l, \boldsymbol{\Sigma}_l)} \boldsymbol{\Sigma}_k^{-1} \left(\boldsymbol{x}(n) - \boldsymbol{\mu}_k\right)$$

using: $\gamma_k(n) = \frac{g_k \, \mathcal{N}(\boldsymbol{x}(n)|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{j=0}^{K-1} g_j \, \mathcal{N}(\boldsymbol{x}(n)|\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)}$

- Leads to:

$$0 = \sum_{n=0}^{N-1} \gamma_k(n) \, \boldsymbol{\Sigma}_k^{-1} \left(\boldsymbol{x}(n) - \boldsymbol{\mu}_k\right)$$

Multiplication with: $\boldsymbol{\Sigma}_k$

$$\boldsymbol{\mu}_k \sum_{n=0}^{N-1} \gamma_k(n) = \sum_{n=0}^{N-1} \gamma_k(n) \, \boldsymbol{x}(n)$$

GMM parameter estimation

TECHNISCHE UNIVERSITÄT DARMSTADT

- Current result:

$$\boldsymbol{\mu}_k \sum_{n=0}^{N-1} \gamma_k(n) \;=\; \sum_{n=0}^{N-1} \gamma_k(n)\, \boldsymbol{x}(n)$$

Leads to:

$$\boldsymbol{\mu}_k \;=\; \frac{\sum_{n=0}^{N-1} \gamma_k(n)\, \boldsymbol{x}(n)}{\sum_{n=0}^{N-1} \gamma_k(n)}$$

Mean of the training data weighted with the corresponding affiliation values.

- A comparable approach can be used to update the covariance matrix:

$$0 \;=\; \frac{d}{d\boldsymbol{\Sigma}_k} \sum_{n=0}^{N-1} \log \left\{ \sum_{l=0}^{K-1} g_l\, \mathcal{N}(\boldsymbol{x}(n) | \boldsymbol{\mu}_l, \boldsymbol{\Sigma}_l) \right\}$$

GMM parameter estimation

- A comparable approach can be used to update the covariance matrix:

$$0 = \frac{d}{d\boldsymbol{\Sigma}_k} \sum_{n=0}^{N-1} \log \left\{ \sum_{l=0}^{K-1} g_l \, \mathcal{N}(\boldsymbol{x}(n) | \boldsymbol{\mu}_l, \boldsymbol{\Sigma}_l) \right\}$$

Leads to:

$$\boldsymbol{\Sigma}_k = \frac{\sum_{n=0}^{N-1} \gamma_k(n) \left(\boldsymbol{x}(n) - \boldsymbol{\mu}_k\right) \left(\boldsymbol{x}(n) - \boldsymbol{\mu}_k\right)^{\mathrm{T}}}{\sum_{n=0}^{N-1} \gamma_k(n)}$$

Mean of the training data variance weighted with the corresponding affiliation values.

GMM parameter estimation

- The gains are calculated based on a Lagrange approach:

$$0 = \frac{d}{dg_k}\left[\sum_{n=0}^{N-1}\log\left\{\sum_{l=0}^{K-1} g_l\,\mathcal{N}(\boldsymbol{x}(n)|\boldsymbol{\mu}_l,\boldsymbol{\Sigma}_l)\right\} + \lambda\left(\sum_{l=0}^{K-1} g_l - 1\right)\right]$$

Constraint:
Sum of Gaussian gains (weights) has to be 1.

- The derivation leads to:

$$0 = \sum_{n=0}^{N-1}\frac{\mathcal{N}(\boldsymbol{x}(n)|\boldsymbol{\mu}_k,\boldsymbol{\Sigma}_k)}{\sum_{l=0}^{K-1} g_l\,\mathcal{N}(\boldsymbol{x}(n)|\boldsymbol{\mu}_l,\boldsymbol{\Sigma}_l)} + \lambda$$

Multiplication with g_k

$$0 = \sum_{n=0}^{N-1}\frac{g_k\,\mathcal{N}(\boldsymbol{x}(n)|\boldsymbol{\mu}_k,\boldsymbol{\Sigma}_k)}{\sum_{l=0}^{K-1} g_l\,\mathcal{N}(\boldsymbol{x}(n)|\boldsymbol{\mu}_l,\boldsymbol{\Sigma}_l)} + \lambda\, g_k$$

$$= \sum_{n=0}^{N-1}\gamma_k(n) + \lambda\, g_k$$

GMM parameter estimation

- Current result:

$$
\begin{aligned}
0 &= \sum_{n=0}^{N-1} \gamma_k(n) + \lambda\, g_k \\
0 &= \sum_{n=0}^{N-1} \underbrace{\sum_{k=0}^{K-1} \gamma_k(n)}_{=1} + \lambda \underbrace{\sum_{k=0}^{K-1} g_k}_{=1} \\
0 &= N + \lambda \\
\lambda &= -N
\end{aligned}
$$

Summation over k

- Results in:

$$
g_k = \frac{1}{N} \sum_{n=0}^{N-1} \gamma_k(n)
$$

Mean of the affiliation values of the training data.

GMM parameter estimation (summary)

TECHNISCHE UNIVERSITÄT DARMSTADT

- Description of the joint probability density by Gaussian mixture models:

$$\tilde{p}(\boldsymbol{x}) = \sum_{k=0}^{K-1} g_k \, \frac{1}{\sqrt{(2\pi)^D \, |\boldsymbol{\Sigma}_k|}} \, e^{-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu}_k)^{\mathrm{T}} \, \boldsymbol{\Sigma}_k^{-1} (\boldsymbol{x}-\boldsymbol{\mu}_k)}$$

- **Iterative procedure or EM (estimation / maximization) procedure for the estimation of the GMM parameters $g_k,\ \boldsymbol{\mu}_k,\ \boldsymbol{\Sigma}_k$ based on N training data values $\boldsymbol{x}(n)$:**
  - 1) Allocation of the data vectors $\boldsymbol{x}(n)$, **E-step**:

$$\gamma_k(n) = \frac{g_k \, \mathcal{N}(\boldsymbol{x}(n)|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{j=0}^{K-1} g_j \, \mathcal{N}(\boldsymbol{x}(n)|\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)}$$

  - 2) Update of the Gaussian parameters, **M-step**:

$$\boldsymbol{\mu}_k = \frac{\sum_{n=0}^{N-1} \gamma_k(n) \, \boldsymbol{x}(n)}{\sum_{n=0}^{N-1} \gamma_k(n)} \qquad g_k = \frac{1}{N} \sum_{n=0}^{N-1} \gamma_k(n)$$

$$\boldsymbol{\Sigma}_k = \frac{\sum_{n=0}^{N-1} \gamma_k(n) \, (\boldsymbol{x}(n) - \boldsymbol{\mu}_k) \, (\boldsymbol{x}(n) - \boldsymbol{\mu}_k)^{\mathrm{T}}}{\sum_{n=0}^{N-1} \gamma_k(n)}$$

GMM parameter estimation Iteration stop criterion

- **Stop criterion for the iteration:**
  In case the joint probability for the training data is only marginally increased by one adaptation step, the iteration is stopped.

$$\frac{\sum_{n=0}^{N-1} \log \left\{ \sum_{k=0}^{K-1} g_k^{\text{new}} \mathcal{N}(\boldsymbol{x}(n) | \boldsymbol{\mu}_k^{\text{new}}, \boldsymbol{\Sigma}_k^{\text{new}}) \right\}}{\sum_{n=0}^{N-1} \log \left\{ \sum_{k=0}^{K-1} g_k^{\text{old}} \mathcal{N}(\boldsymbol{x}(n) | \boldsymbol{\mu}_k^{\text{old}}, \boldsymbol{\Sigma}_k^{\text{old}}) \right\}} < 1 + \epsilon$$

GMM parameter estimation: Critical issues for the iteration

TECHNISCHE UNIVERSITÄT DARMSTADT

- When the cost function is optimized

$$\log p(\boldsymbol{X}|\boldsymbol{g},\boldsymbol{\mu},\boldsymbol{\Sigma}) = \sum_{n=0}^{N-1} \log \left\{ \sum_{k=0}^{K-1} g_k \mathcal{N}(\boldsymbol{x}(n)|\boldsymbol{\mu}_k,\boldsymbol{\Sigma}_k) \right\} \longrightarrow \text{max}$$

  one has to be careful when only few training vectors are allocated to one Gaussian.

- **Example:**

  Assume one had only diagonal covariance matrices with the same elements on the diagonal: $\boldsymbol{\Sigma}_k = \sigma_k^2 \boldsymbol{I}$

  and one feature vector would be identical to one Gaussian mean:

$$\boldsymbol{x}(n_0) = \boldsymbol{\mu}_{k_0}$$

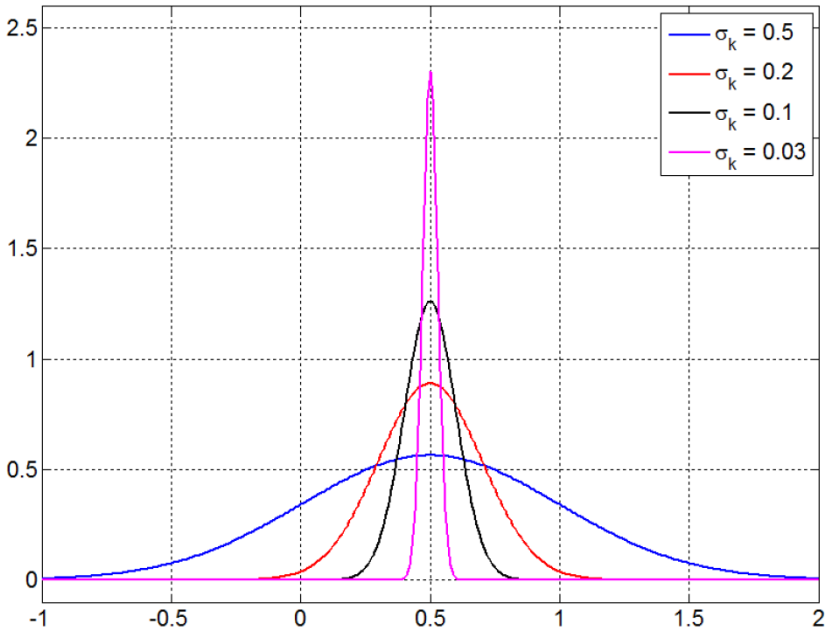
GMM parameter estimation: Critical issues for the iteration

TECHNISCHE UNIVERSITÄT DARMSTADT

- **Example (cont.):**

  Then the variance would approach zero and would lead the value of the Gaussian to infinity at the feature vector value.

  => in that case the optimization criterion is reached.



- In order to avoid these cases, the diagonal elements of the covariance matrices are lower limited corresponding to a lower limit of the width of the Gaussians.

GMM parameter estimation: Initialization of the iteration

- **Initialization with the k-means of LBG algorithms:**

  Typically, a reasonable initialization is recommended for the GMM iteration.

  Therefore, a **codebook is trained** and the codebook vectors are used as initialization for the **mean Gaussian values**.

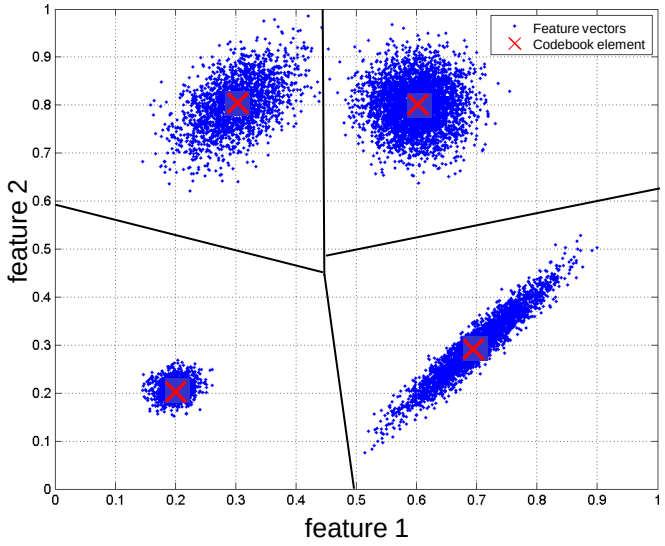
  The **covariance matrices** are estimated based on the covariance of the data vectors allocated to the respective codebook vector.

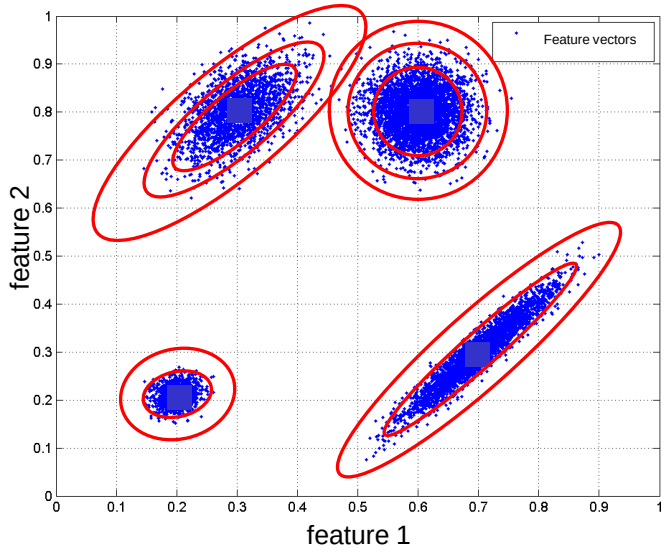
  The **weights** are initialized by the number of elements allocated to each codebook entry.

Gaussian mixture models vs. Codebooks

- **Codebooks:**
  Training data for quantized vector description.

- **GMMs:**
  Training data for pdf estimation. Pdf modeled by Gaussian functions.





Gaussian mixture models vs. Codebooks

TECHNISCHE UNIVERSITÄT DARMSTADT

- 1) Allocation of the data vectors $\boldsymbol{x}(n)$, E-step:

$$\gamma_k(n) = \frac{g_k \, \mathcal{N}(\boldsymbol{x}(n)|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{j=0}^{K-1} g_j \, \mathcal{N}(\boldsymbol{x}(n)|\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)}$$

In the codebook training (LBG algorithm) a training data value is allocated to one cluster center („hard allocation“). The GMM allocation is soft.

- 2) Update of the Gaussian parameters, M-step:
  - Adaptation of the means:

$$\boldsymbol{\mu}_k = \frac{\sum_{n=0}^{N-1} \gamma_k(n) \, \boldsymbol{x}(n)}{\sum_{n=0}^{N-1} \gamma_k(n)}$$

The adaptation of the mean is the same for GMMs and Codebooks. However, with a binary allocation for codebooks.

Gaussian mixture models vs. Codebooks

TECHNISCHE UNIVERSITÄT DARMSTADT

- 2) Update of the Gaussian parameters, M-step:
  - Adaptation of the covariance matrices:

$$\boldsymbol{\Sigma}_k = \frac{\sum_{n=0}^{N-1} \gamma_k(n) \left(\boldsymbol{x}(n) - \boldsymbol{\mu}_k\right) \left(\boldsymbol{x}(n) - \boldsymbol{\mu}_k\right)^{\mathrm{T}}}{\sum_{n=0}^{N-1} \gamma_k(n)}$$

The sum of the diagonal elements of the covariance matrix is sometimes used for an evaluation of the codebook training.

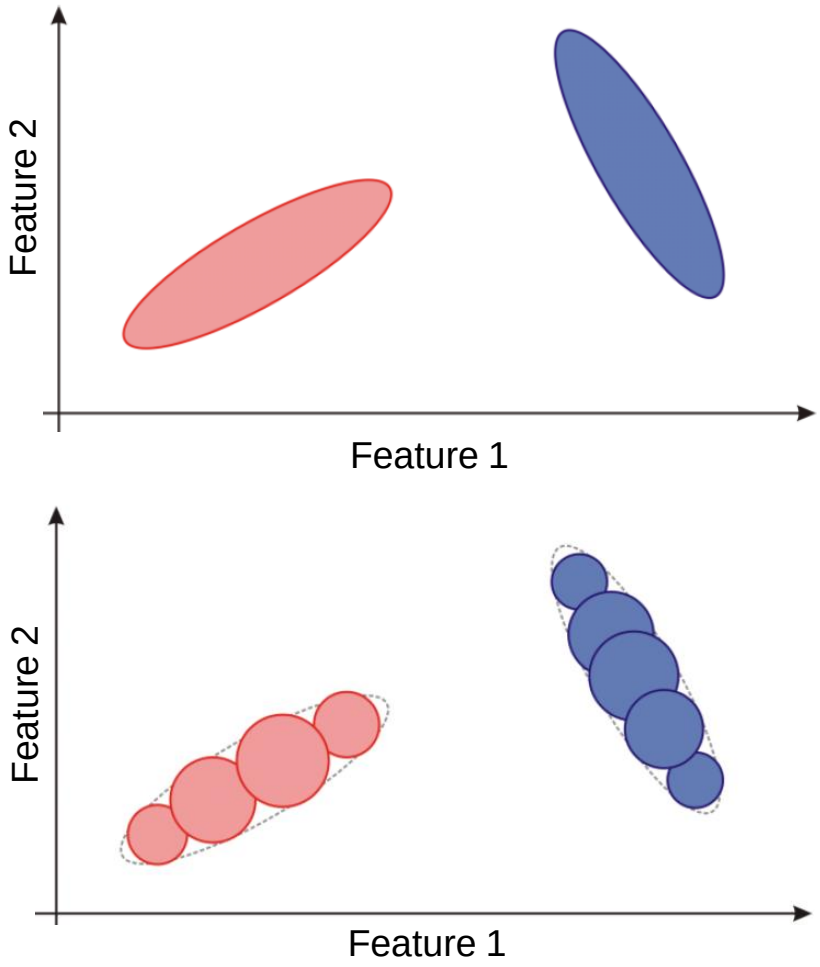
  - Adaptation of the weights:

$$g_k = \frac{1}{N} \sum_{n=0}^{N-1} \gamma_k(n)$$

The weights of the codebook entries are typically chosen identically.
For a MAP (max. a posteriori) estimation, also a weighting according to the number of allocated elements is possible (which is g_k)

Complexity reduction

- In the general case DxD elements of the covariance matrices have to be estimated.
- In case one would only consider the diagonal elements, the complexity can be significantly reduced.
- However, in order to obtain the same approximation quality, one has to model more Gaussians.



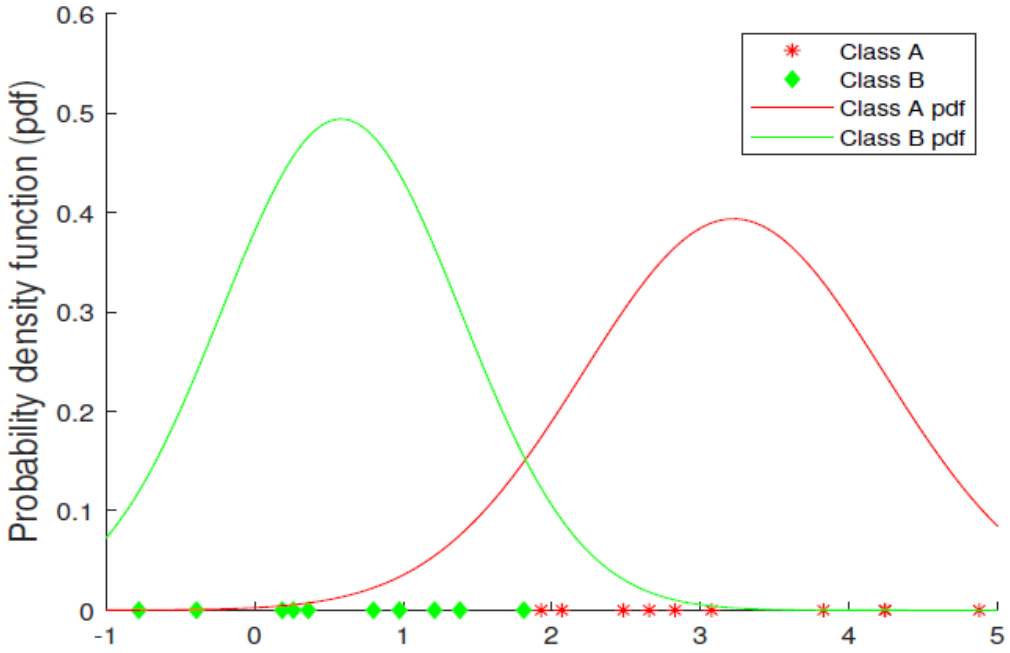
Other Classification Methods

- Naïve Bayes
- Decision tree
- Random Forest
- k-Nearest Neighbor
- Linear Support Vector Machine (SVM)
- DNNs: Multi-Layer Perceptron

Other Classification Methods: Naïve Bayes

- Simpler concept than GMMs: Model the probabilities of each class with one Gaussian only and not a weighted some as in the GMMs.

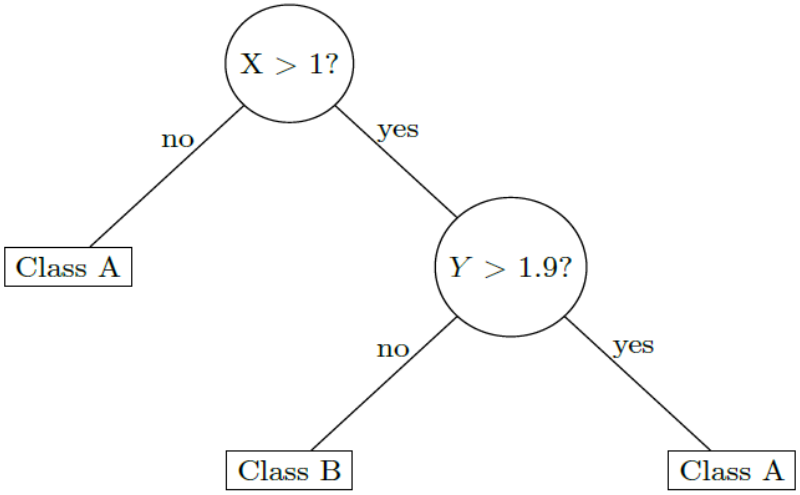
Example for a one-dimentional feature:

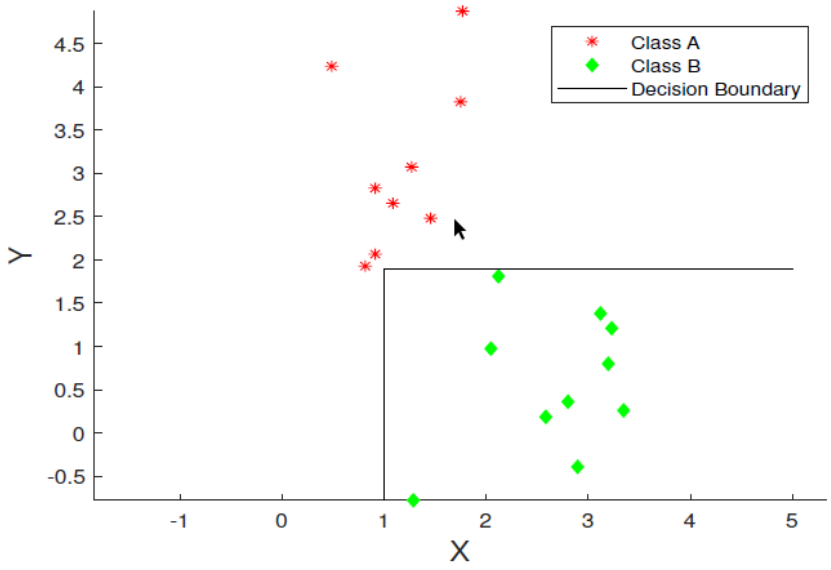


Other Classification Methods: Decision tree

- Application of hierarchical decision rules for partitioning the feature space.

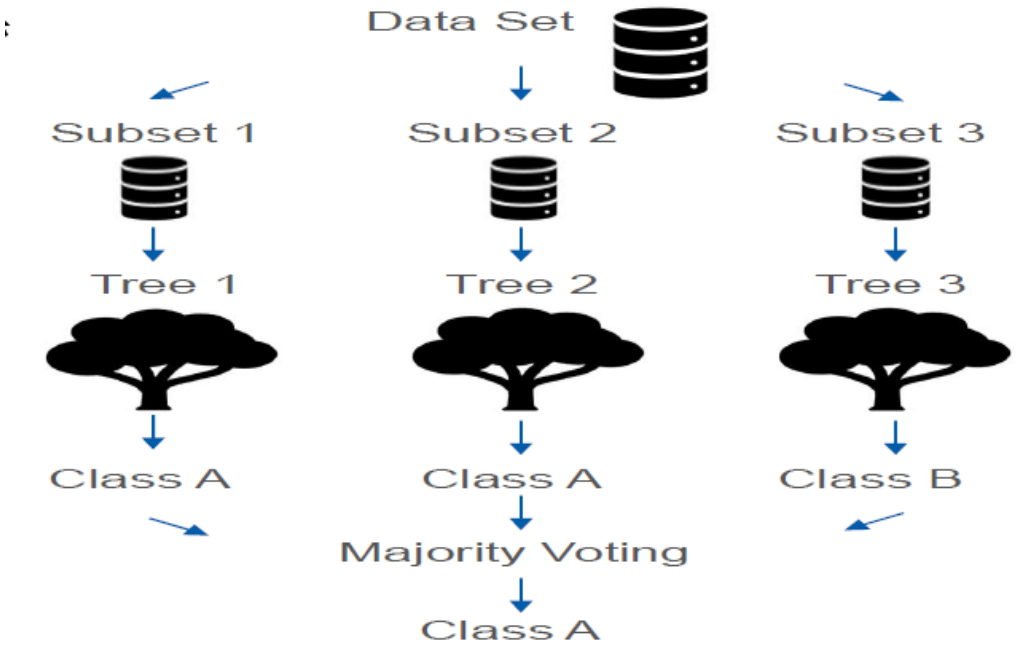
Example for a two-dimentional feature set:





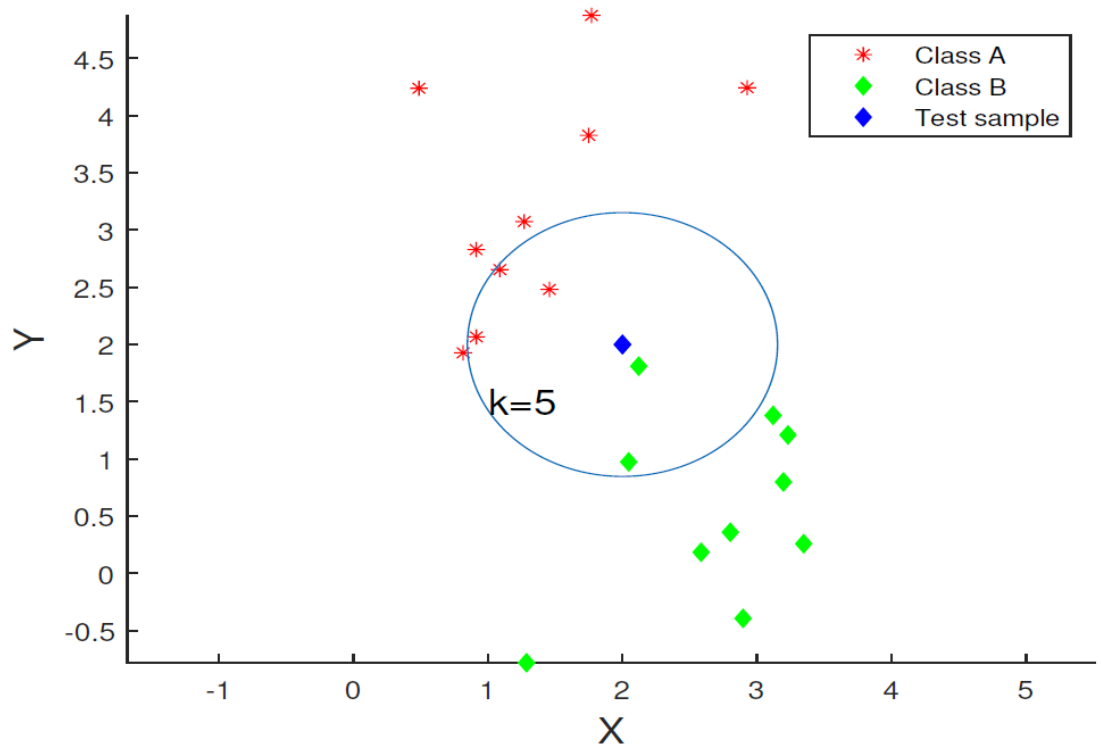
Other Classification Methods: Random Forest

- Train several decision trees on different subsets of the whole data set.
  Each subset produces an individual decision. Overall decision by majority voting.



Other Classification Methods: k-Nearest Neighbor

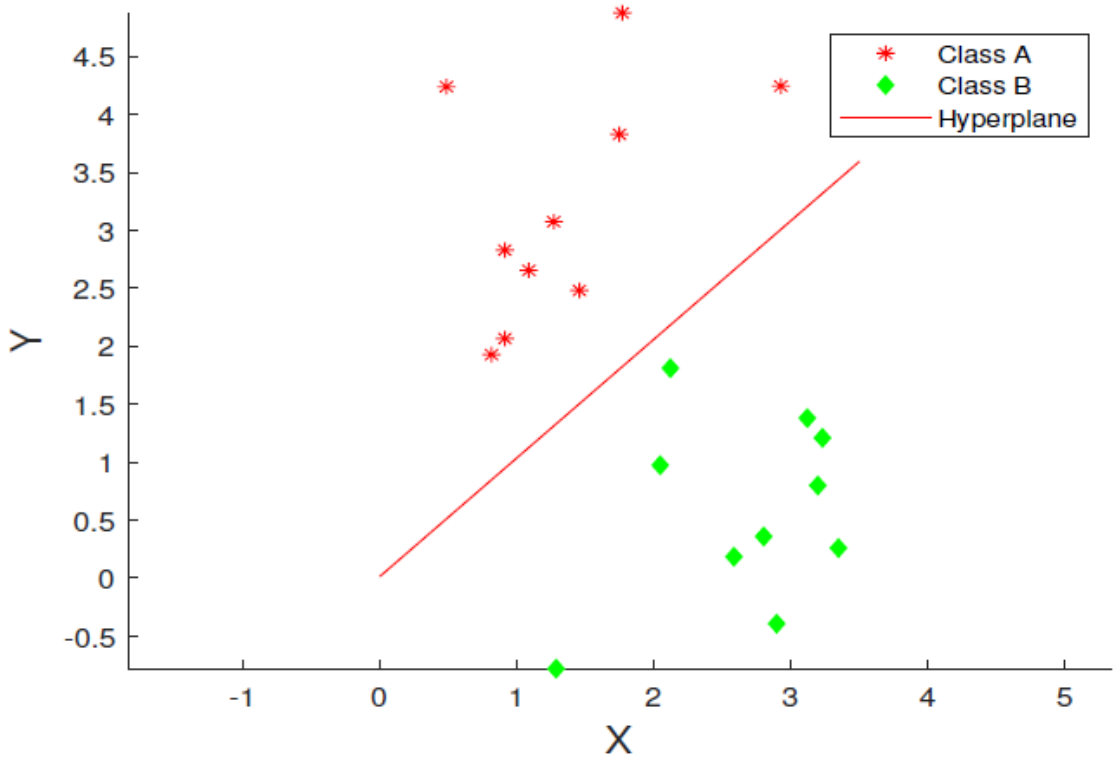
- Number of training samples per class closest to the test sample (current observation to classify determines the decision class.



Other Classification Methods: Linear Support Vector Machine (SVM)

TECHNISCHE UNIVERSITÄT DARMSTADT

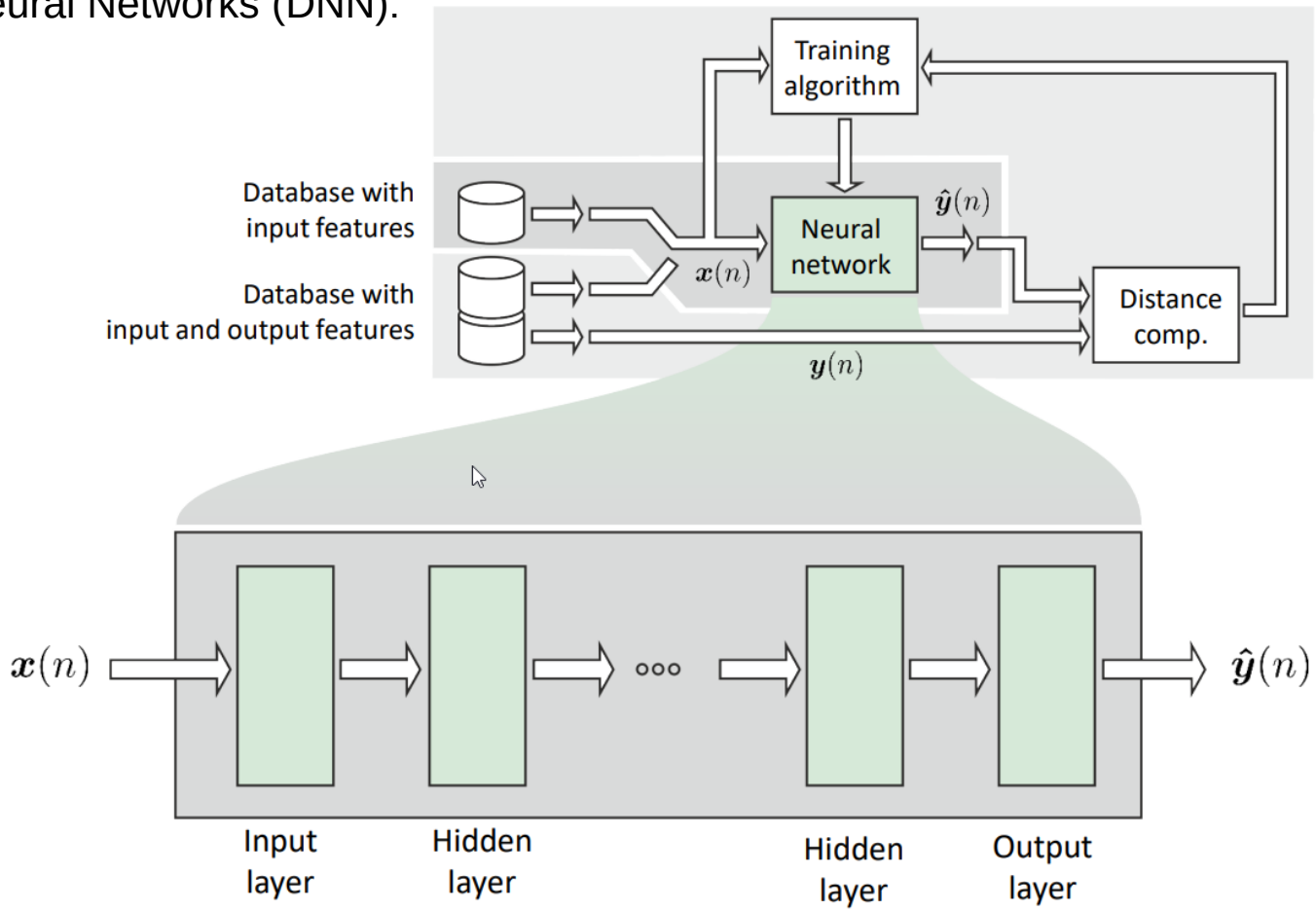
- A hyperplane in the n-dimensional space separates the different classes. More classes to differentiate, more hyperplanes.



Other Classification Methods: DNNs: Multi-Layer Perceptron

TECHNISCHE UNIVERSITÄT DARMSTADT

- Deep Neural Networks (DNN):



Other Classification Methods: DNNs: Multi-Layer Perceptron

- **Input Layer:**

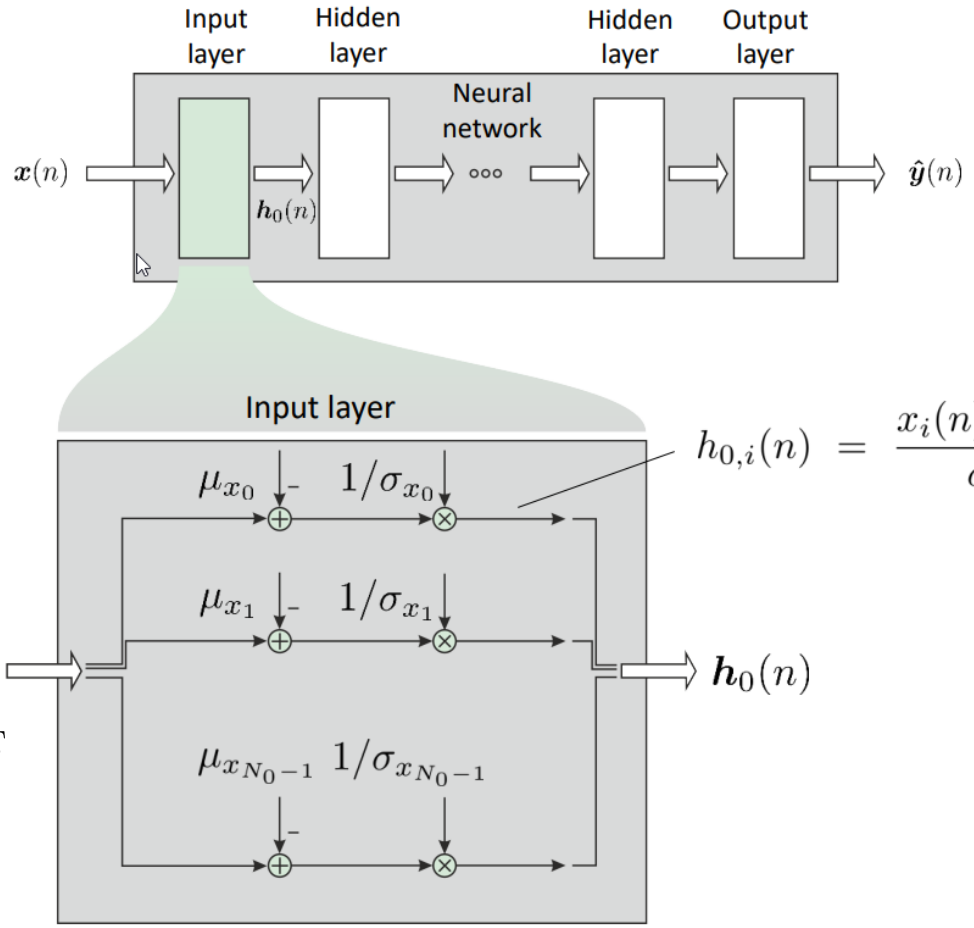
  - Sometimes input data is propagated unchanged to the next layer:

$$\boldsymbol{h}_0(n) = \boldsymbol{x}(n)$$

  - sometime changed to zero mean and normalized variance:

$$h_{0,i}(n) = \frac{x_i(n) - \mu_{x_i}}{\sigma_{x_i}}$$

$$\boldsymbol{h}_0(n) = [h_{0,0}(n), \ldots, h_{0,N_0-1}(n)]^{\mathrm{T}}$$



Other Classification Methods: DNNs: Multi-Layer Perceptron

- **Hidden Layer**, i.e., Neural processing: Linearly weighted sum with bias:

$$x_{m,i}(n) = \boldsymbol{w}_{m,i}^{\mathrm{T}}\, \boldsymbol{h}_m(n) + b_{m,i}$$

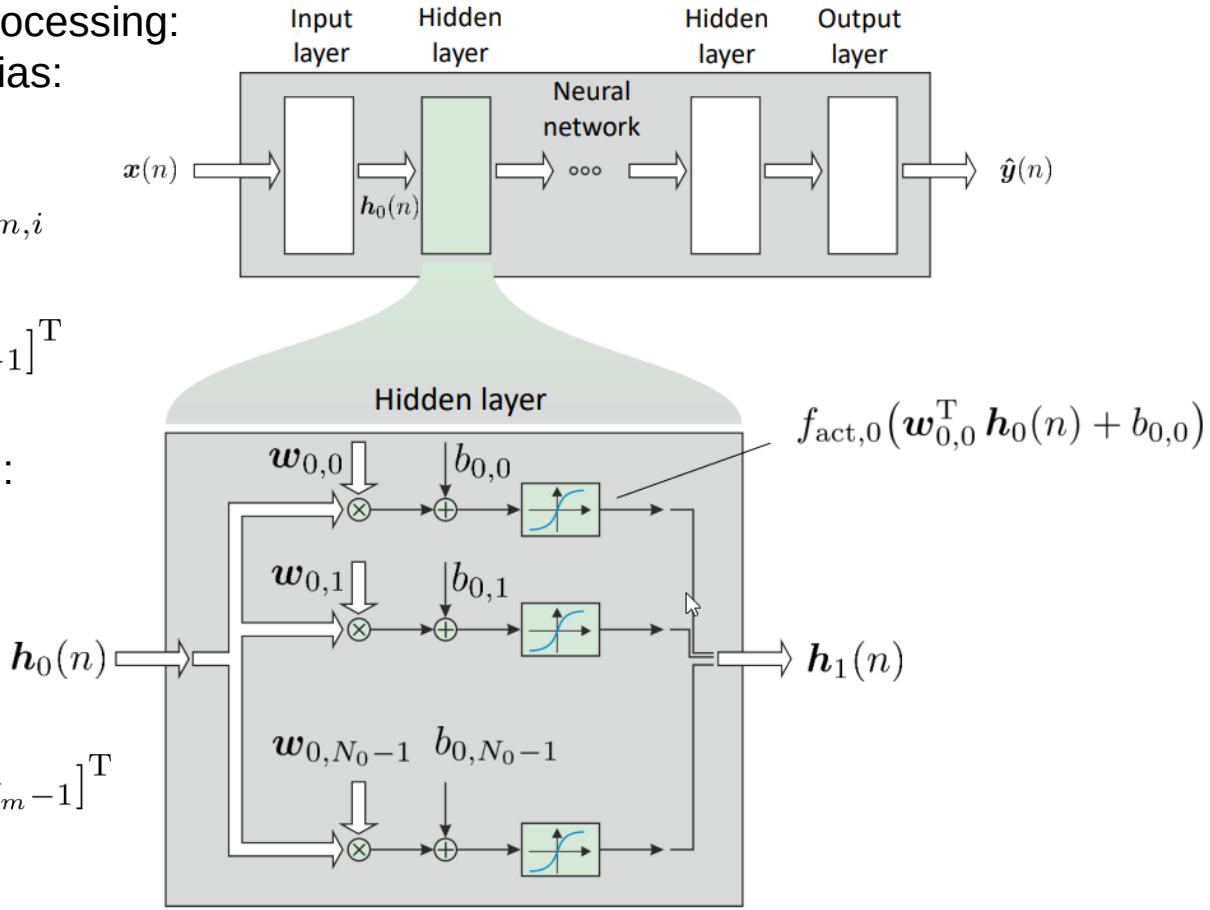
$$\boldsymbol{w}_{m,i} = [w_{m,0}, \ldots, w_{m,N_m-1}]^{\mathrm{T}}$$

- Non-linear activation function:

$$y_{m,i}(n) = f_{\mathrm{act},m}\left(x_{m,i}(n)\right)$$

- Combination to a vector:

$$\boldsymbol{h}_{m+1}(n) = [y_{m,0}, \ldots, y_{m,N_m-1}]^{\mathrm{T}}$$



Other Classification Methods: DNNs: Multi-Layer Perceptron

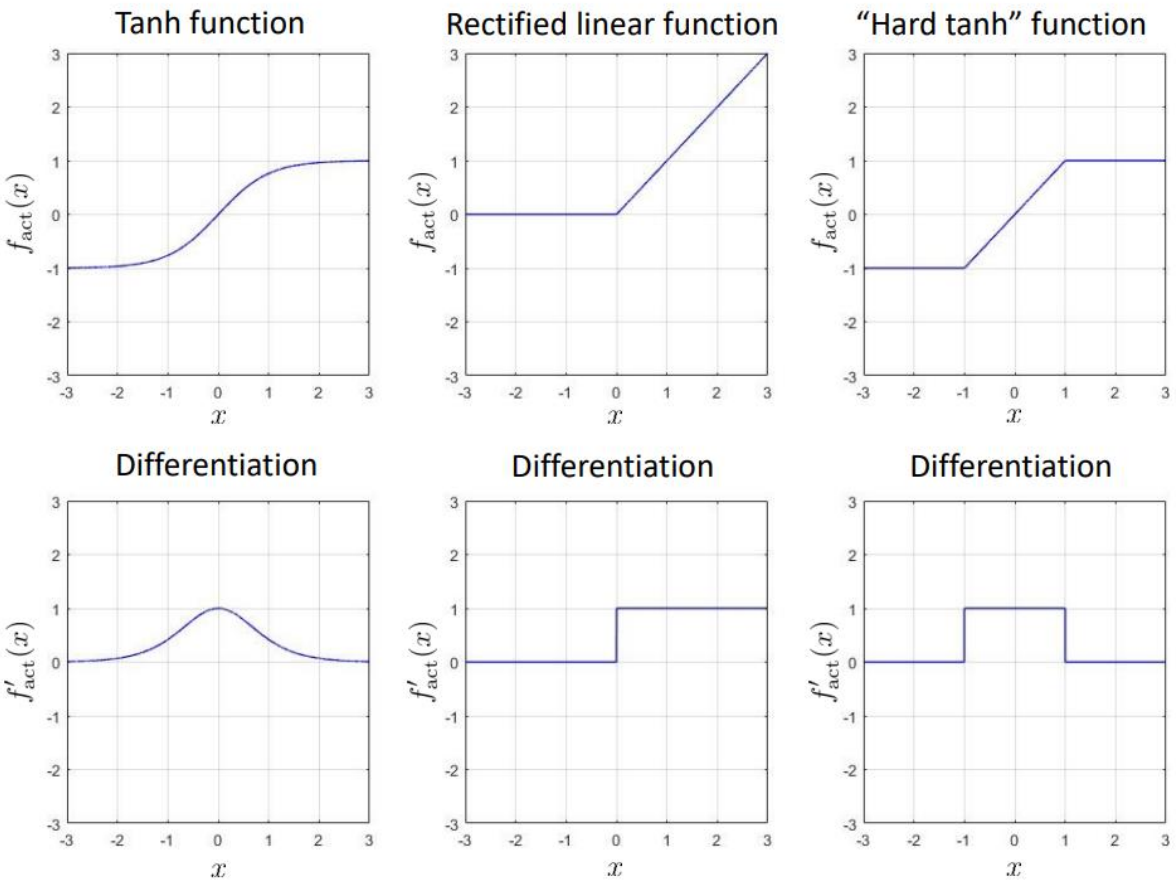
- **Activation functions:**

- Most used:
- the **Tanh** function:

$$f_{\text{act}}\left(x(n)\right) = \frac{e^{2x(n)} - 1}{e^{2x(n)} + 1}$$

- the Rectified Linear function (or Unit; **ReLU**):

$$f_{\text{act}}\left(x(n)\right) = \max\left\{0,\, x(n)\right\}$$



TECHNISCHE UNIVERSITÄT DARMSTADT

- **Output Layer:**
  - Sometimes a pass-through layer only:

$$\hat{\boldsymbol{y}}(n) = \boldsymbol{h}_M(n)$$

  - sometime a limitation, and/or:

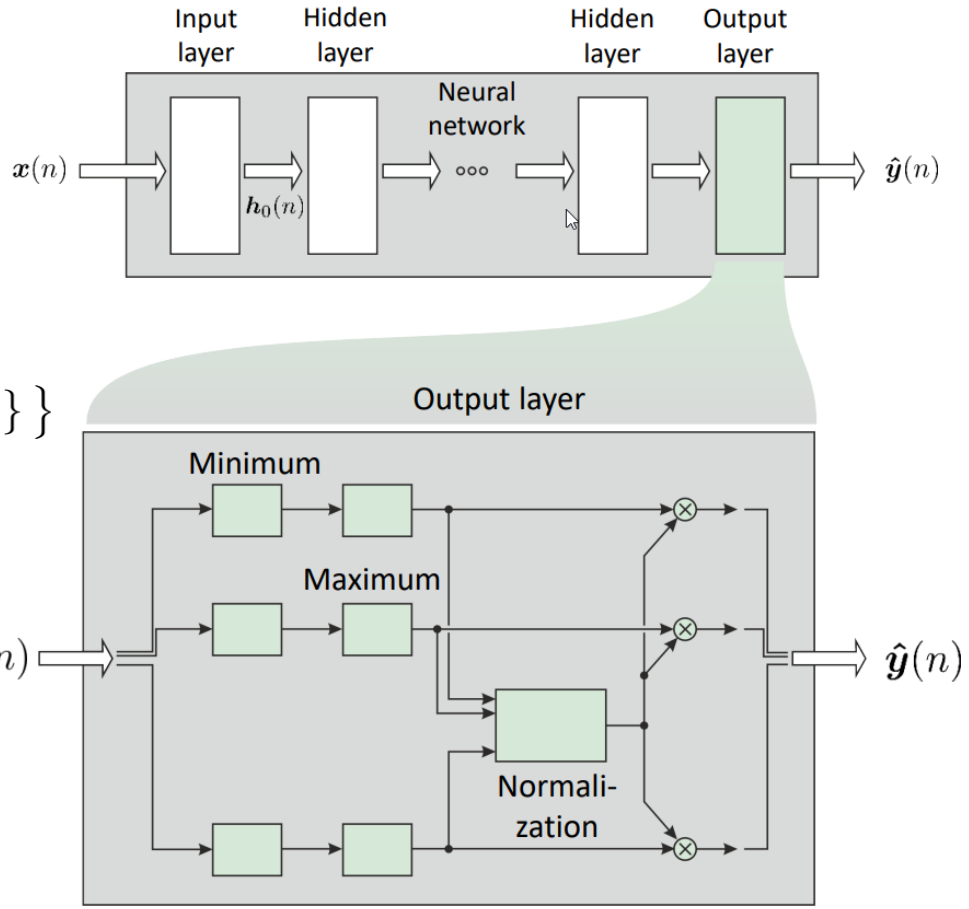
$$\hat{y}_i(n) = \max\left\{\hat{y}_{\min}, \min\left\{\hat{y}_{\max}, h_{M,i}(n)\right\}\right\}$$

  - a normalization, and/or:

$$\hat{y}_i(n) = \frac{h_{M,i}(n)}{\sum_{i=0}^{N_M-1} h_{M,i}(n)}$$

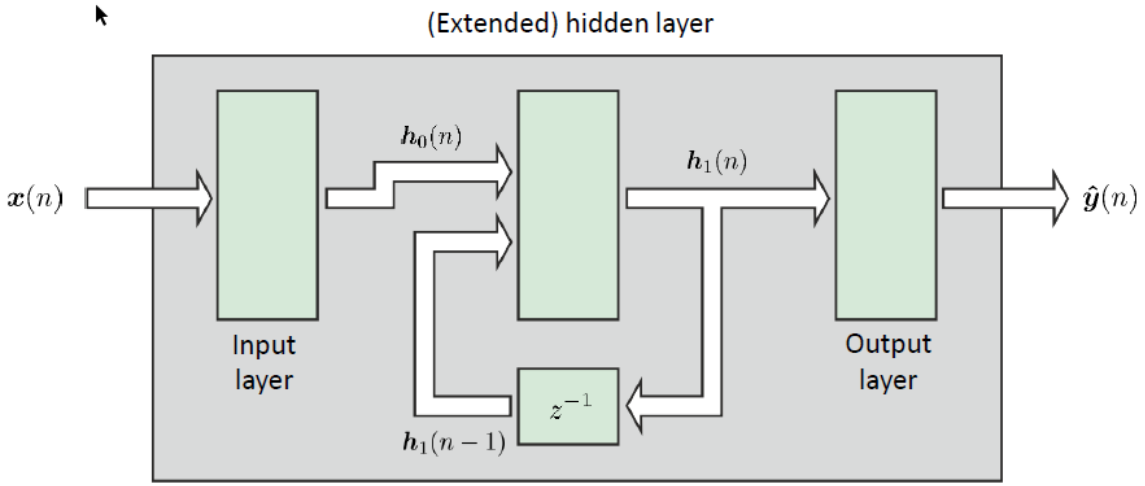
  - or a quantization / decision is performed:

$$\hat{y}_i(n) = \begin{cases} 1 & : \quad h_{M,i}(n) > \text{limit} \\ 0 & : \quad \text{else} \end{cases}$$



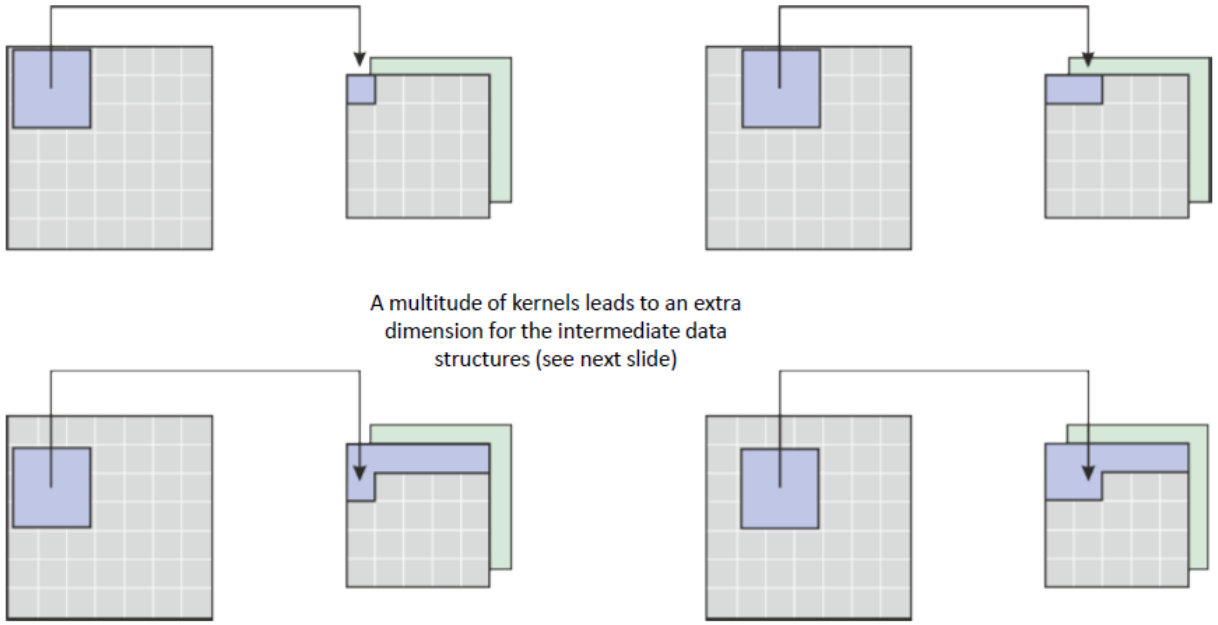
Other Classification Methods using DNNs

- **Some general comments:**
  - Multi-Layer Perceptrons (MLPs) realize complex and learned decision rules. Typically, it makes sense to use a feature vector and not raw data as input.
  - In order to realize a **memory dependent decision** (typically time correlated data where memory to consider makes sense) recurrent networks with memory elements should be used, such as **Long short time memory networks (LSTMs)**:



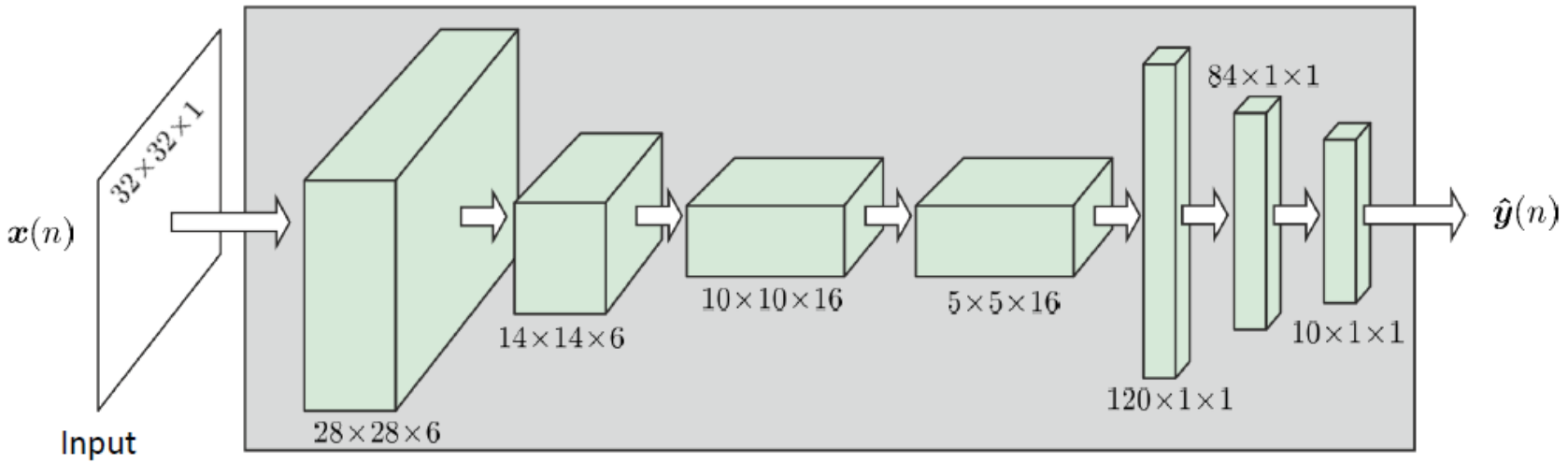
Other Classification Methods using DNNs

- **Some general comments:**
  - If not feature should be used but **raw data as input,** usually a **convolutional layer net** is used as first processing stage: feature learning followed by a detection stage with MLPs or LSTMs.
  - Convolution performed by weighted (according **kernel** weights) sum of the input data.
    Might be done several times with different kernels => more dimensions.



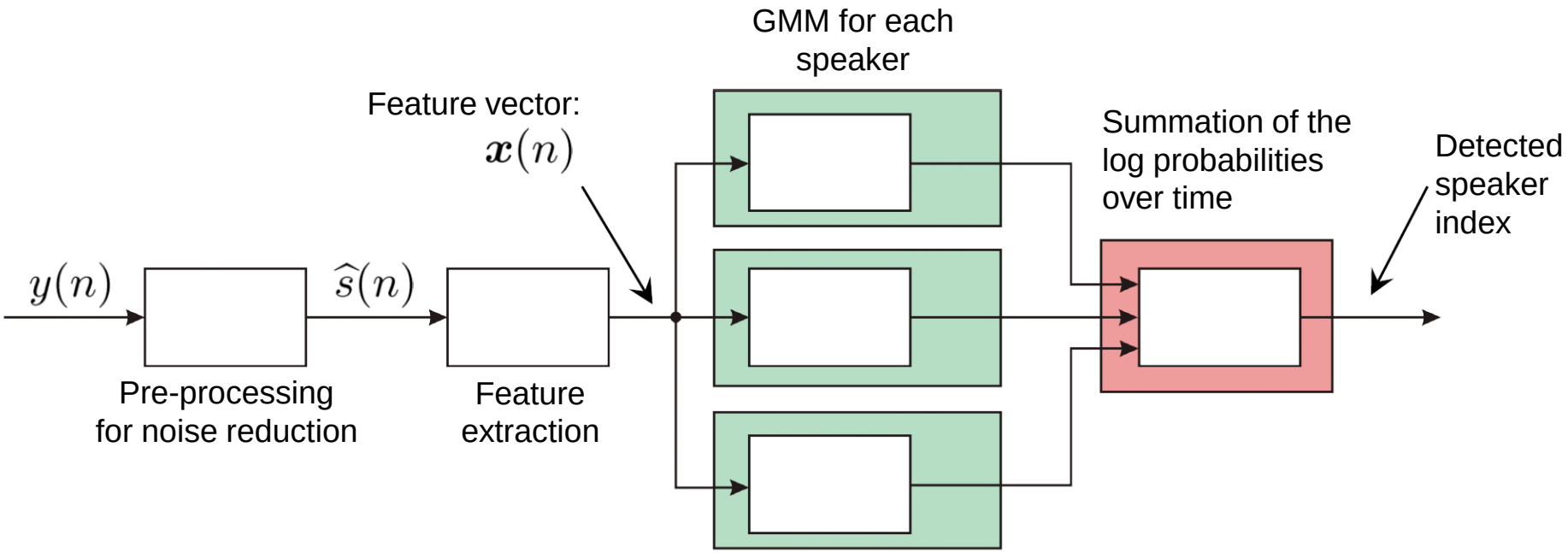
Other Classification Methods using DNNs

- **Typical example structure of a convolutional network:**



Application – separate lecture

- Speaker recognition with GMMs:



More, about speaker recognition, next lecture…

Summary

TECHNISCHE UNIVERSITÄT DARMSTADT

- Introduction to decision problems.
- Basis: Gaussian distributions: single and multi-dimensional.
- Motivation for GMMs based on the estimation of multi-dimensional distributions.
- GMM description:
  - Estimation of GMM parameters based on training data.
  - Derivation of an iterative estimation procedure.
  - Simplifications and applications.
- Alternative detection methods
- Next lecture:
  - Speaker recognition based on MFCCs and GMMs.

References

TECHNISCHE UNIVERSITÄT DARMSTADT

Gaussian mixture models:

[1] Nuno Vasconcelos: The Gaussian classifier
http://svcl.ucsd.edu/~courses/ece271A-F10/handouts/GC.pdf

[2] Frédéric Bimbot, et al.: *A Tutorial on Text-Independent Speaker Verification,*
Eurasip Journal on Applied Signal Processing 2004:4, 430-451

[3] C. M. Bishop: *Pattern Recognition and Machine Learning*, Springer, 2006

[4] L. Rabiner, B.H. Juang: *Fundamentals of Speech Recognition*, Prentice Hall, 1993

[5] B. Gold, N. Morgan: *Speech and Audio Signal Processing*, Wiley, 2000