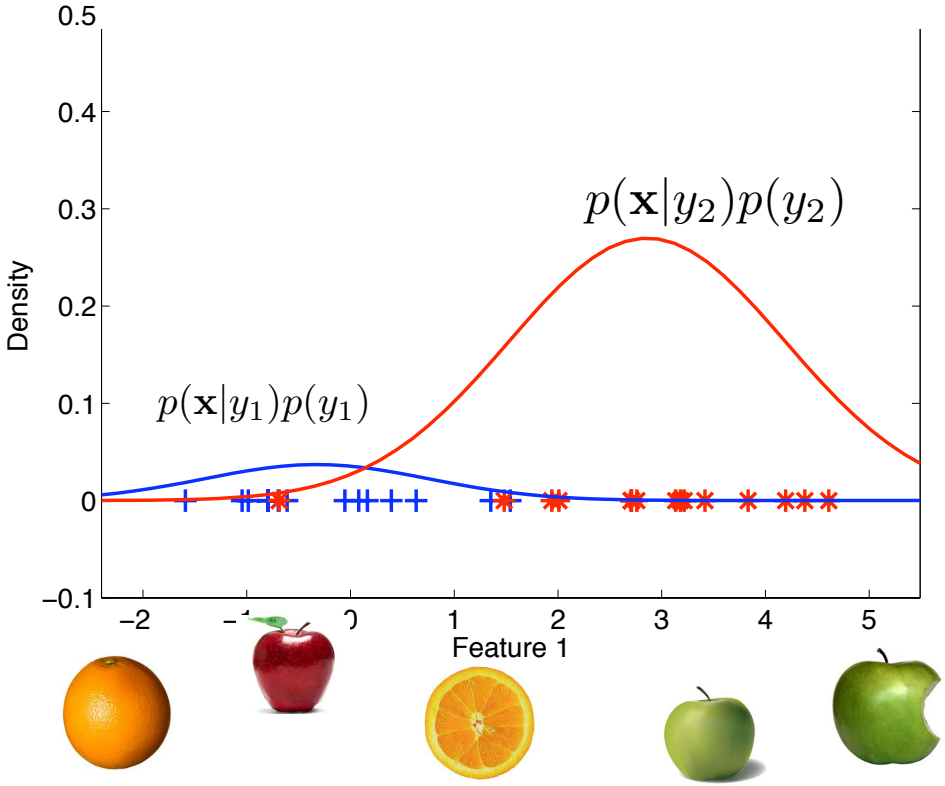
Parametric Density-based Classifiers

CSE2510 Machine Learning



TU Delft

Contents this week:

- Recap of last time:
  objects, feature vectors, feature space, decision boundary, class conditional probability, posterior probability, Bayes classifier, Bayes rule

- Classifiers based on probability estimates
- The curse of dimensionality
- Classifiers based on Gaussian density estimates (Quadratic, linear, nearest mean)
- Scaling of features

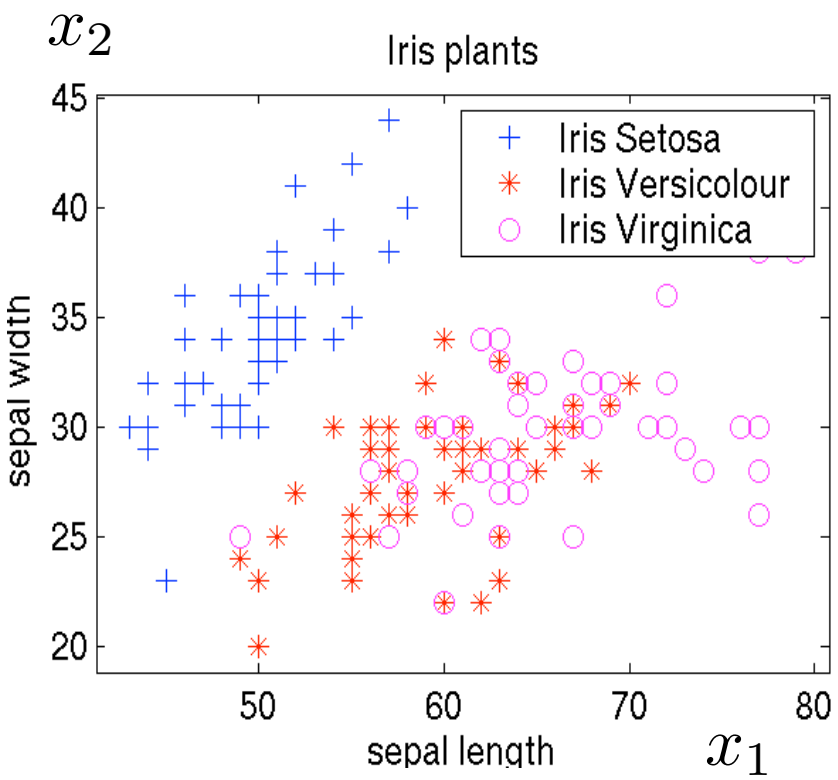
Objects in feature space

- We can interpret the measurements as a vector in a vector space:

$$\mathbf{x} = (x_1, x_2, ..., x_p)$$

- This originates, in principle, from a probability density over the whole feature space

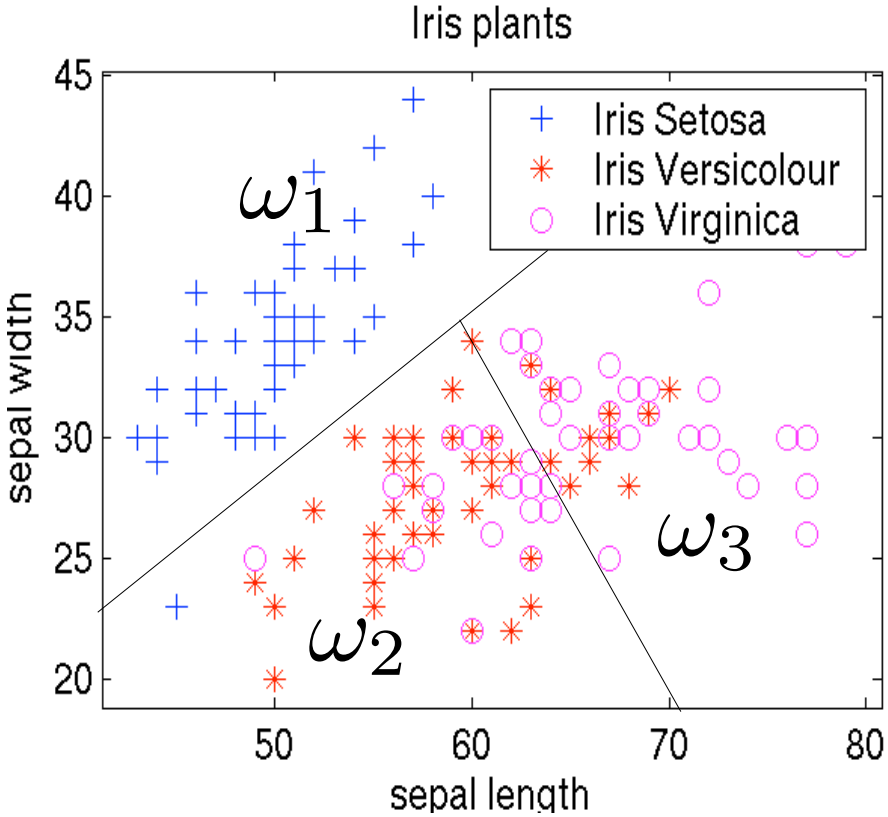
$$p(\mathbf{x}, y)$$



TU Delft

Classification

- Given labeled data: $\mathbf{x}$
- Assign to each object a class label ω
- In effect splits the feature space in separate regions



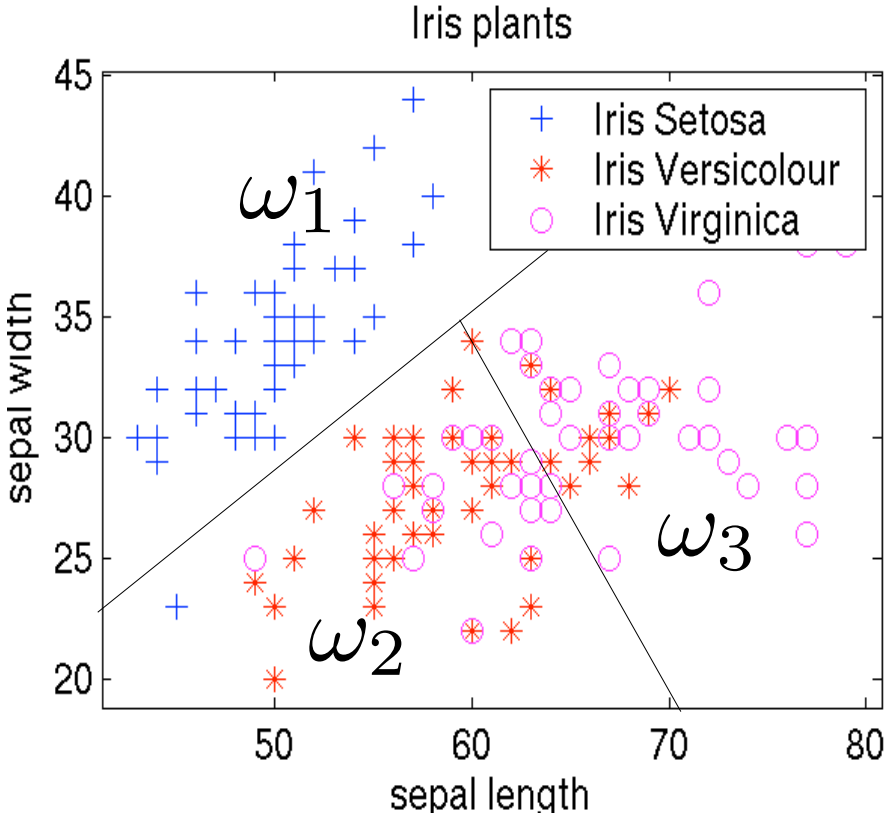
TU Delft

Classification: NOTE

- Given labeled data: $\mathbf{x}$
- Assign to each object a class label ω

- Note that I've used ω to indicate the class, and not y

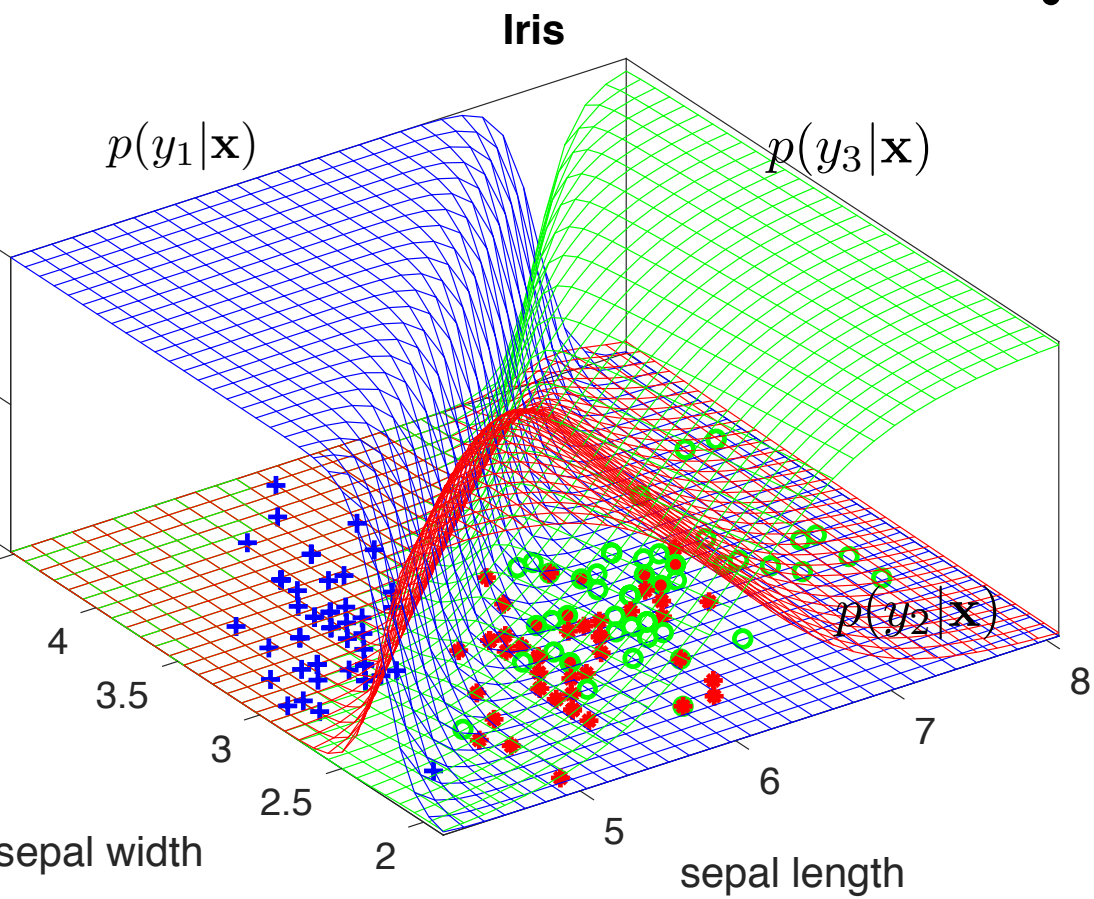
- This is what the book is using



TU Delft

$$p(y_1|\mathbf{x}) > p(y_2|\mathbf{x})$$

Output of the model



- For each object in the feature space, we should find:

$$p(y|\mathbf{x})$$

In practice, we approximate: $\hat{p}(y|\mathbf{x})$

or we fit a function:

$$f(\mathbf{x})$$

TU Delft

Bayes' theorem

- In many cases the posterior is hard to estimate
- Often a functional form of the class distributions can be assumed
- Use Bayes' theorem to rewrite one into the other:

$$p(y|\mathbf{x}) = \frac{p(\mathbf{x}|y)p(y)}{p(\mathbf{x})}$$

class (conditional) distribution	$p(\mathbf{x}\|y)$
class prior	$p(y)$
(unconditional) data distribution	$p(\mathbf{x})$

A note on notation

- Mathematically speaking, there is a difference between

$$p(\mathbf{x}) \quad \text{and} \quad P(\mathbf{x})$$

- (Actually, hard-core mathematicians may use

$$f_X(\mathbf{x}) \quad \text{and} \quad P_X(\mathbf{x}) \quad)$$

A note on notation

- Mathematically speaking, there is a difference between

$$p(\mathbf{x}) \quad \text{and} \quad P(\mathbf{x})$$

probability density
continuous variable

probability mass
discrete variable

- (Actually, hard-core mathematicians may use

$$f_X(\mathbf{x}) \quad \text{and} \quad P_X(\mathbf{x}) \quad)$$

A note on notation

- Mathematically speaking, there is a difference between

$p(\mathbf{x})$ and $P(\mathbf{x})$

probability density
continuous variable

probability mass
discrete variable

- Machine Learners are a bit more sloppy:
  always use $p(\mathbf{x})$

Another note on notation

- We are working with feature **vectors**
- Column or row, that is the question?

- In mathematics, typically a column: $p(\mathbf{x})$

- In data matrix, one object in a row:

$$X = \begin{bmatrix} -1 & -1 \\ -1 & +1 \\ 2 & 0 \\ 3 & 0 \end{bmatrix}$$

(Two-dimensional dataset with four objects)

Data distribution

- In Bayes rule you see $p(\mathbf{x})$. How to get it?
- You can explicitly compute it:

$$p(\mathbf{x}) = p(\mathbf{x}|y_1)p(y_1) + p(\mathbf{x}|y_2)p(y_2)$$

- But if you just find the largest posterior:

$$\frac{p(\mathbf{x}|y_1)p(y_1)}{\cancel{p(\mathbf{x})}} > \frac{p(\mathbf{x}|y_2)p(y_2)}{\cancel{p(\mathbf{x})}}$$

Description of a classifier

There are several ways to describe a classifier:

- if $p(y_1|\mathbf{x}) > p(y_2|\mathbf{x})$ then assign to y_1
  otherwise y_2

- if $p(y_1|\mathbf{x}) - p(y_2|\mathbf{x}) > 0$ then assign to y_1

- or $\dfrac{p(y_1|\mathbf{x})}{p(y_2|\mathbf{x})} > 1$

- or $\log(p(y_1|\mathbf{x})) - \log(p(y_2|\mathbf{x})) > 0$

NOTE!

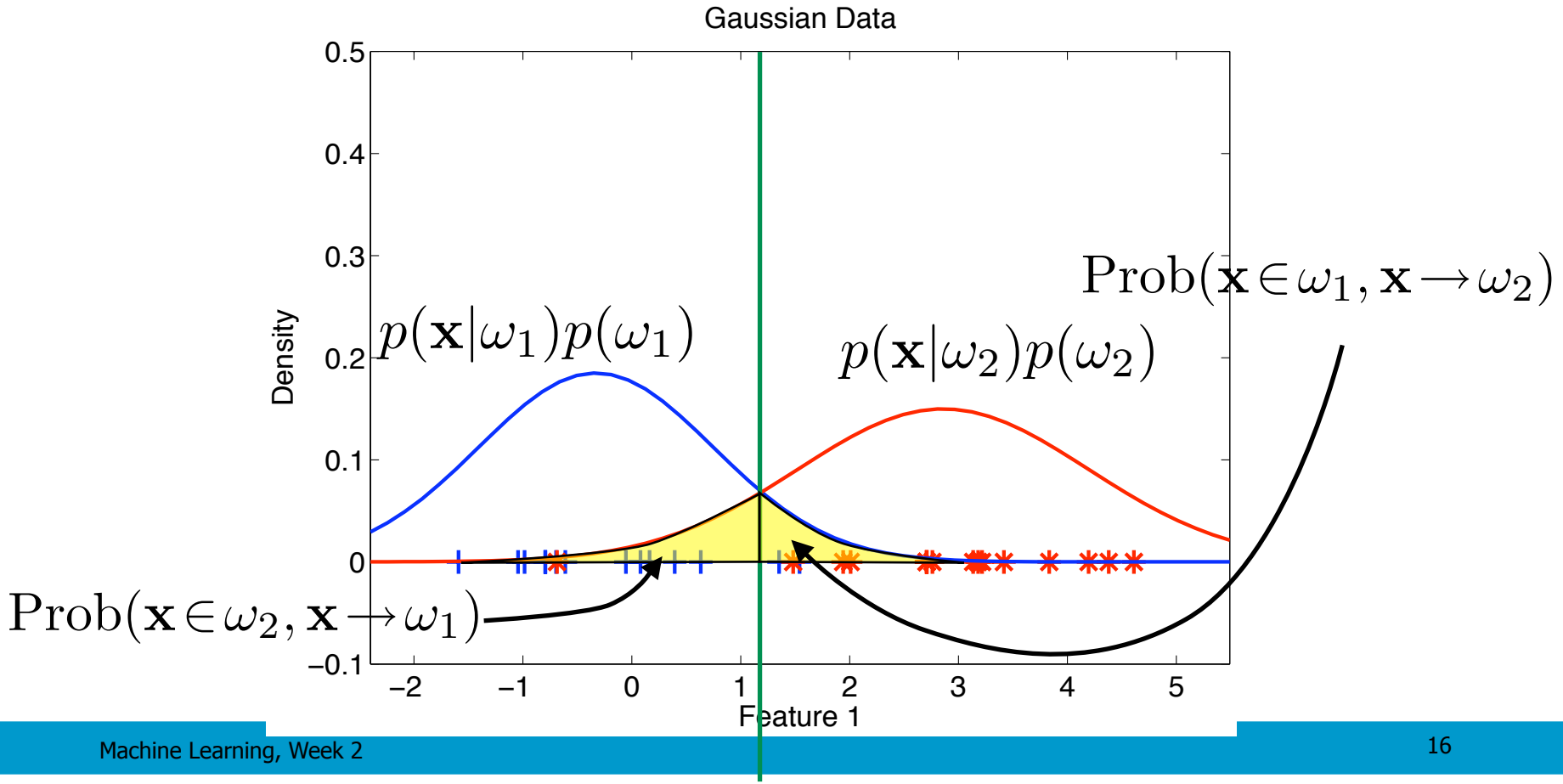
- Up to now:
- Assume that we know the true $p(y|\mathbf{x})$

  or $p(\mathbf{x}|y),\ p(y)$

- In practice:
- We only get a sample (training set), drawn from this distribution

Bayes error ε^*

Bayes error is the **minimum** error: typically >0 !!



TU Delft

Models is What We Need!

- We don't have the true distributions, only a sampled dataset from it
- We have to approximate $p(y|\mathbf{x})$ or $p(\mathbf{x}|y), p(y)$

- We will consider
  - Discriminative and generative models
  - Parametric and nonparametric models

Discriminative Models

$$p(y|\mathbf{x})$$

- When we know the posterior probability densities, we can directly classify objects
- Hard problem : given measurements, e.g. women's height, how can we estimate $p(height|woman)$?
- Strong approximations are needed

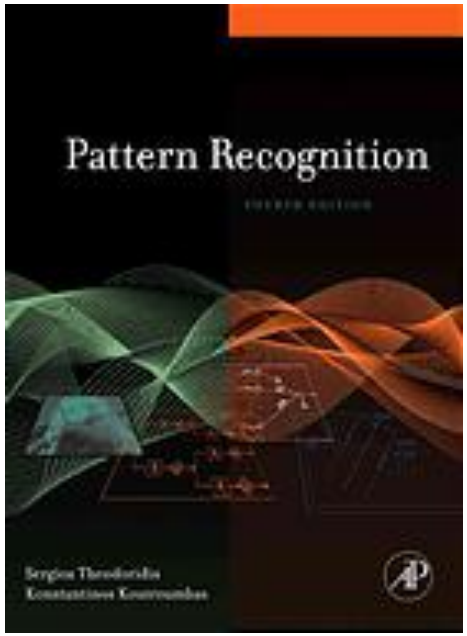
Generative Models

$$p(y|\mathbf{x}) \propto p(y)p(\mathbf{x}|y)$$

- When we know the prior and conditional densities, we know everything about the data for classification
- Density has to be estimated
- Given examples from different classes, 'standard' density estimation is sufficient

Literature

- Sections 2.1 - 2.4 from this book:
- Pattern recognition,
  by Sergios Theodoridis, Konstantinos Koutroumbas
  (2009)



- An eBook is available
  from the TU Library

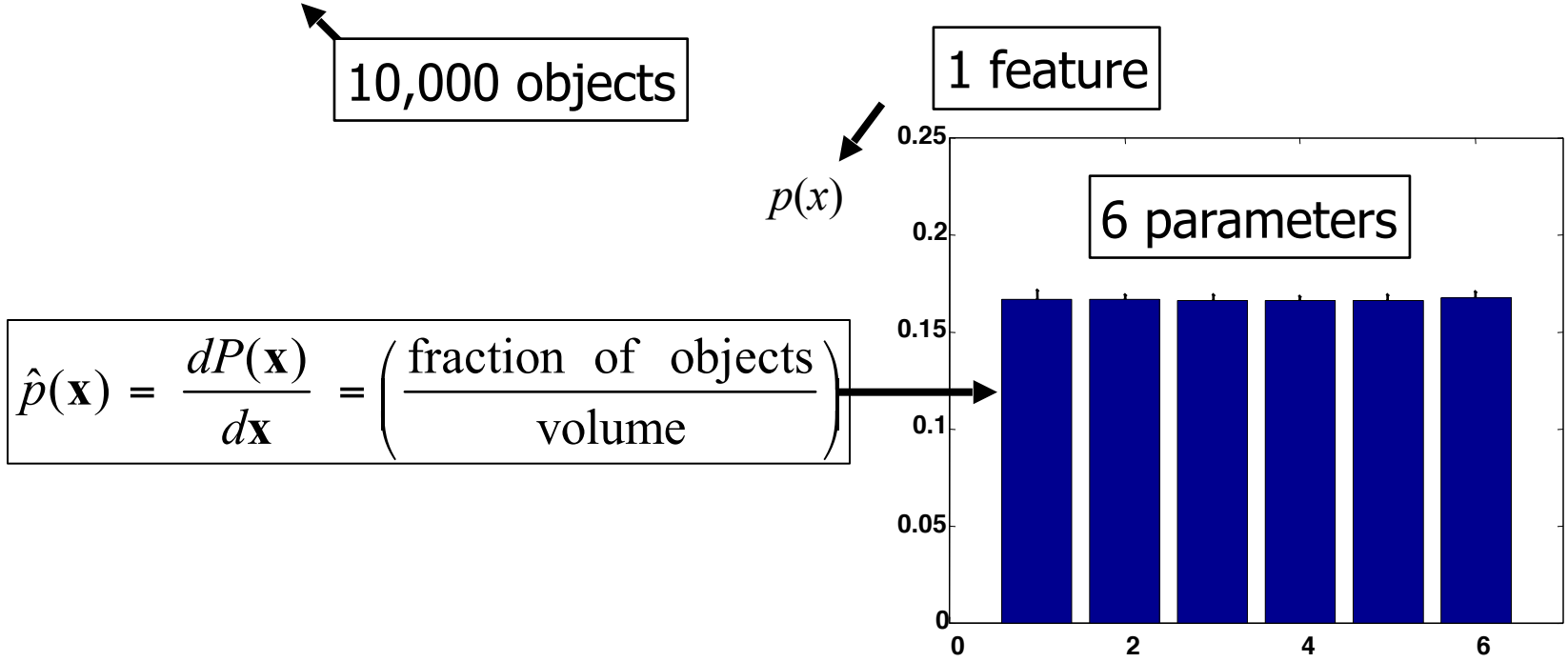
Parametric Modeling & Estimation

- Density estimation
  - Simple nonparametric approach
  - Curse of dimensionality
  - Parametric models

- Some related topics
  - Sphering
  - Properties of Gaussian
  - Mixture modeling

Histogram-based Density Estimation

- Relatively simple approach : approximate density by histogram
  E.g. 10,000 throws of a dice

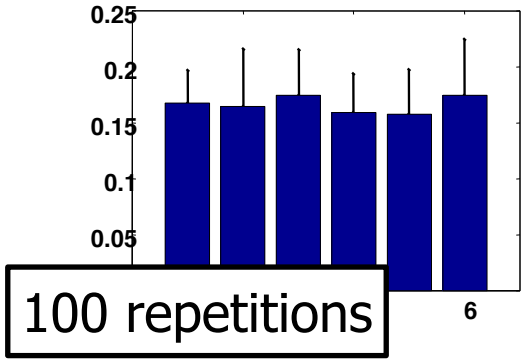


TU Delft

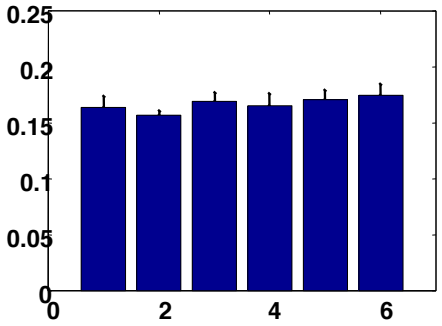
Histogram-based Density Estimation

- Problem : accuracy

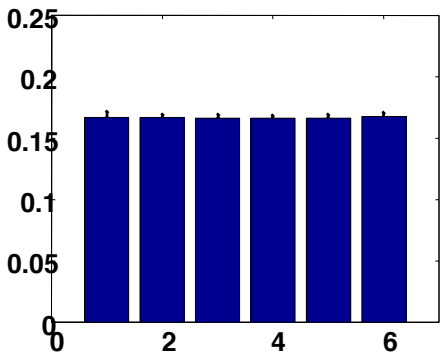
100 objects



1,000 objects



10,000 objects

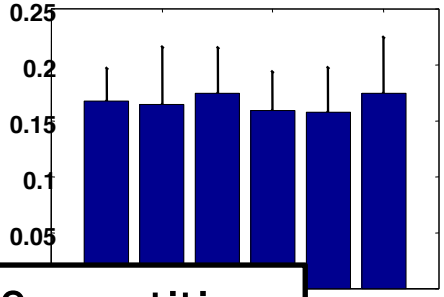


TU Delft

Histogram-based Density Estimation

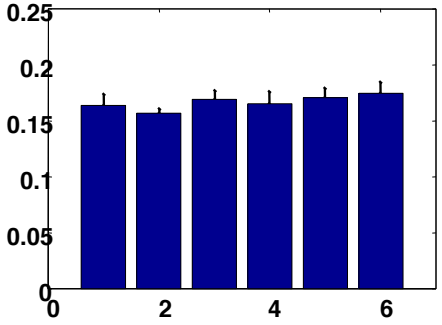
- Problem : accuracy

100 objects

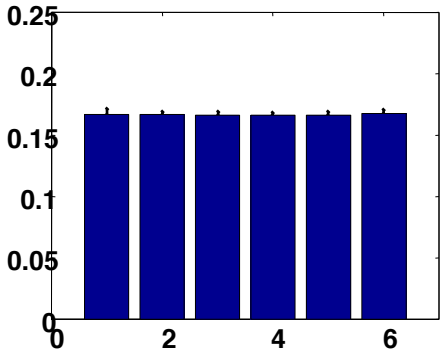


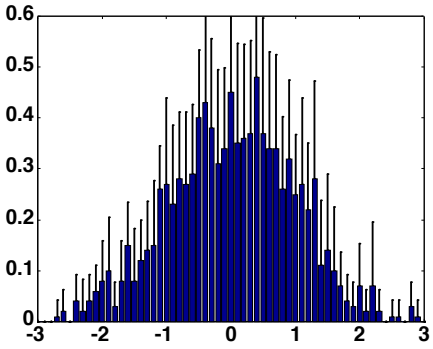
100 repetitions

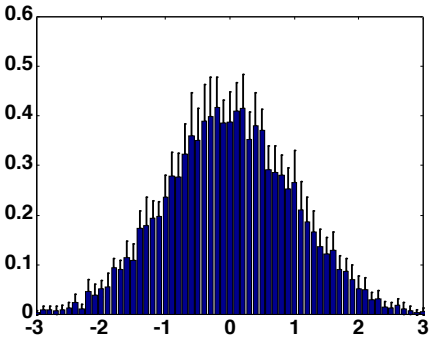
1,000 objects

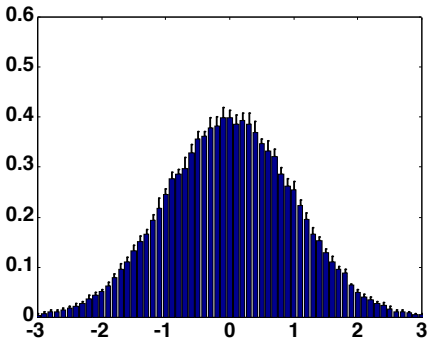


10,000 objects





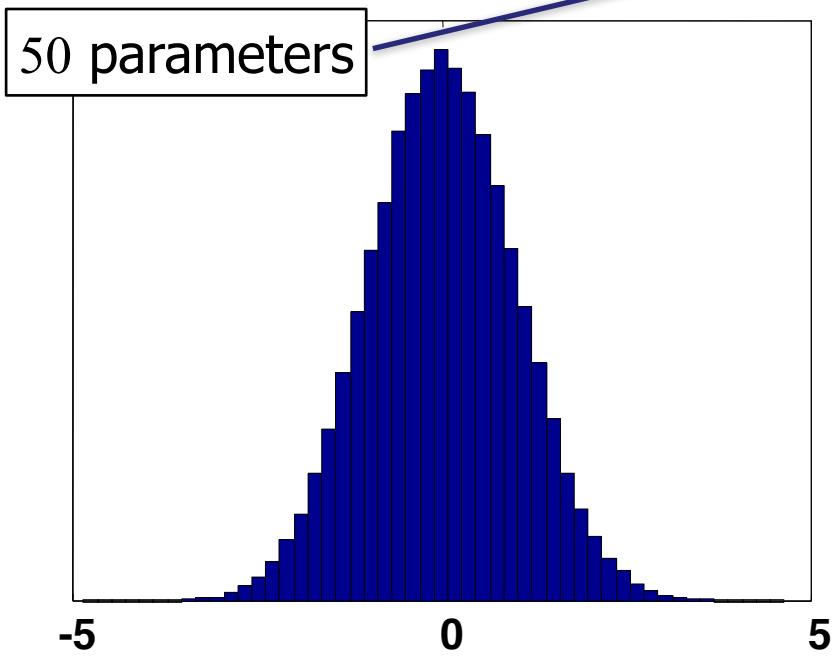




TU Delft

Histogram-based Density Estimation

For 1-dimensional data,
± 1000 objects needed



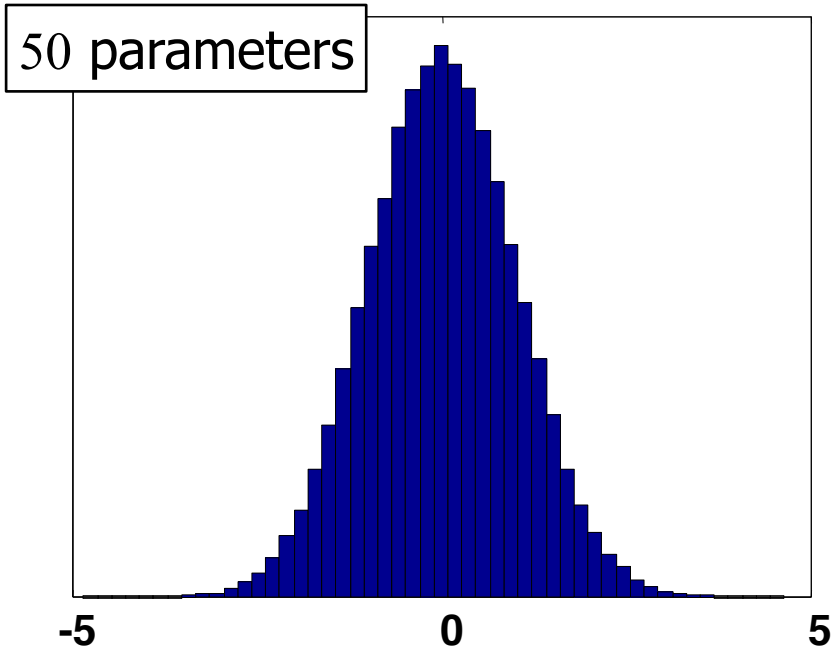
For each bin we estimate one value:
with 50 bins we have 50 parameters

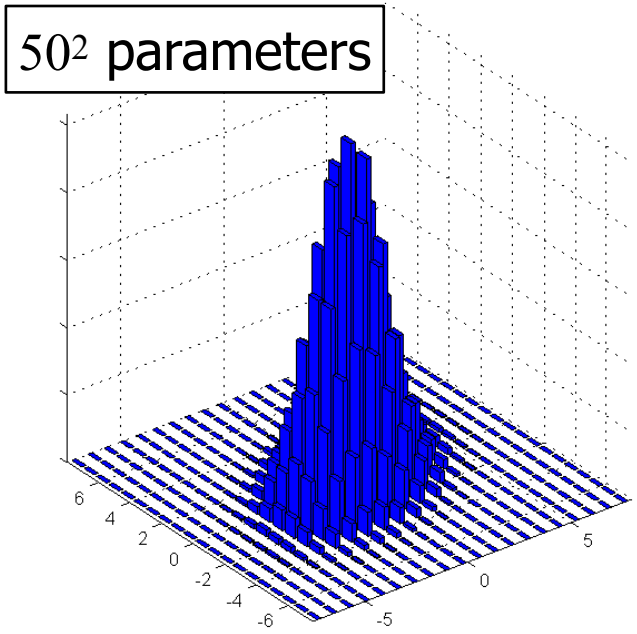
TU Delft

Histogram-based Density Estimation

For 1-dimensional data,
± 1000 objects needed

For p-dimensional data,
$\pm 1000^{p}$ objects needed





- Unworkable for $p > 2$ features

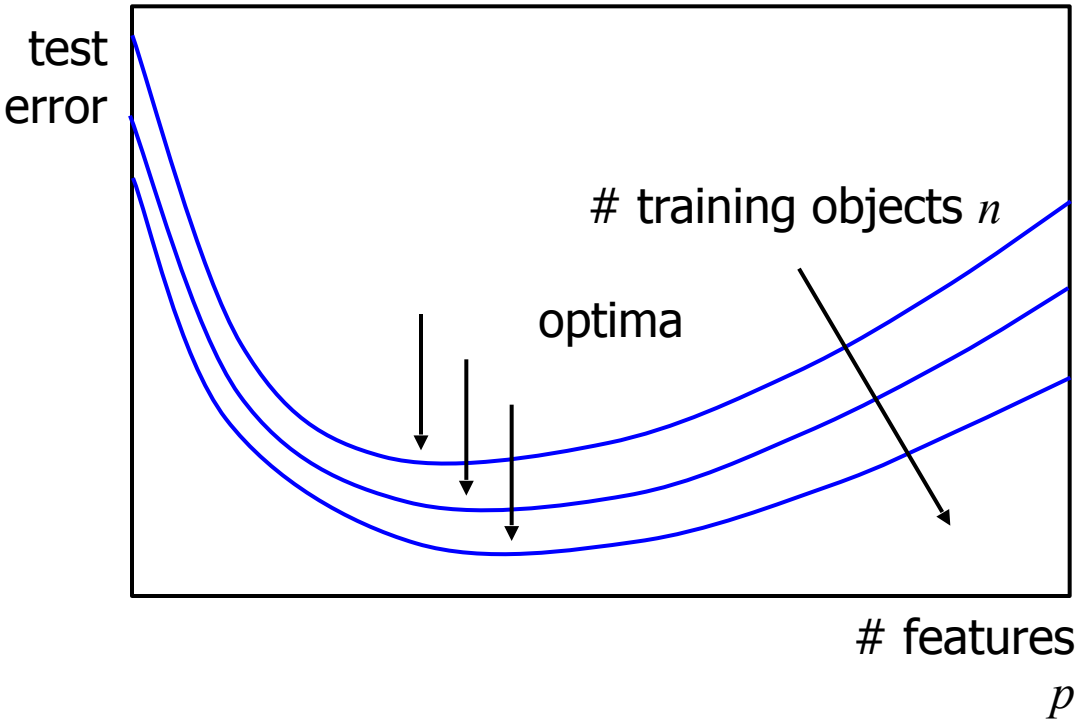
TU Delft

Curse of Dimensionality

- Intuitively, using more features [e.g. width, height, color etc.] should give us more information about the outcome to predict
- But never know densities, so have to estimate
- Number of parameters [e.g. histogram bins] to estimate increases with the number of features
- To estimate these well, you need more objects

- Consequence : there is an optimal number of features to use

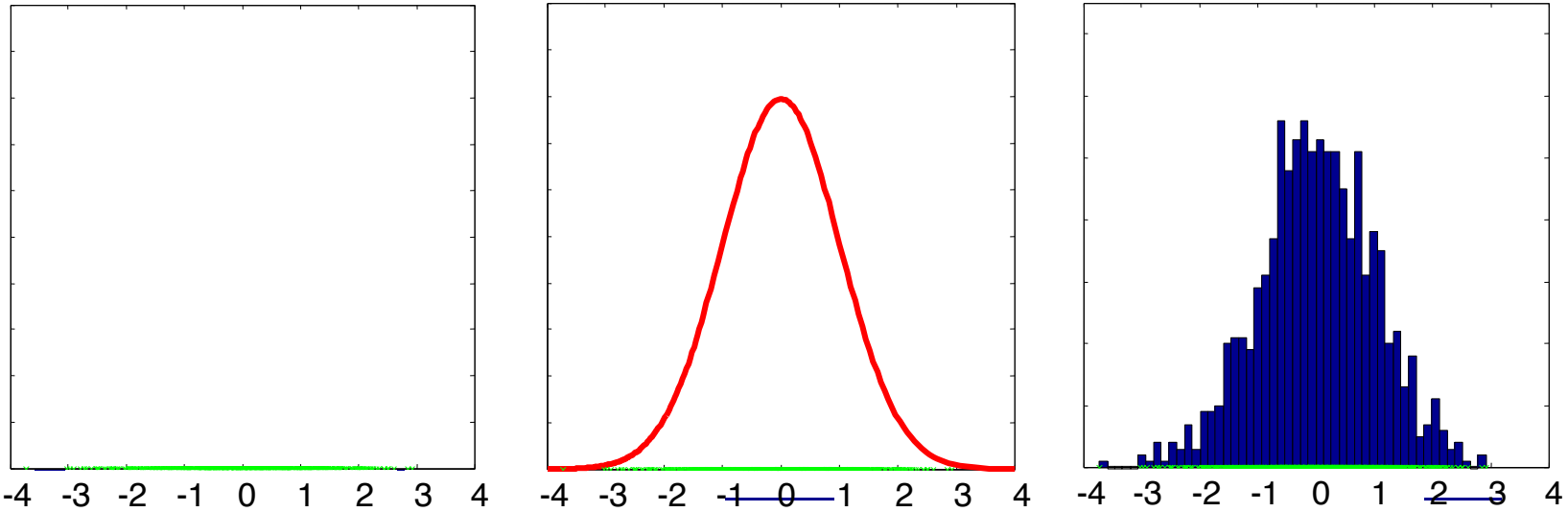
Curse of Dimensionality




f(x)
0,4
0,3
0,2
0,1
$f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$
0
1
2
3
4
5
6
x
$\mu - \sigma$
μ
$\mu + \sigma$
10
GL890

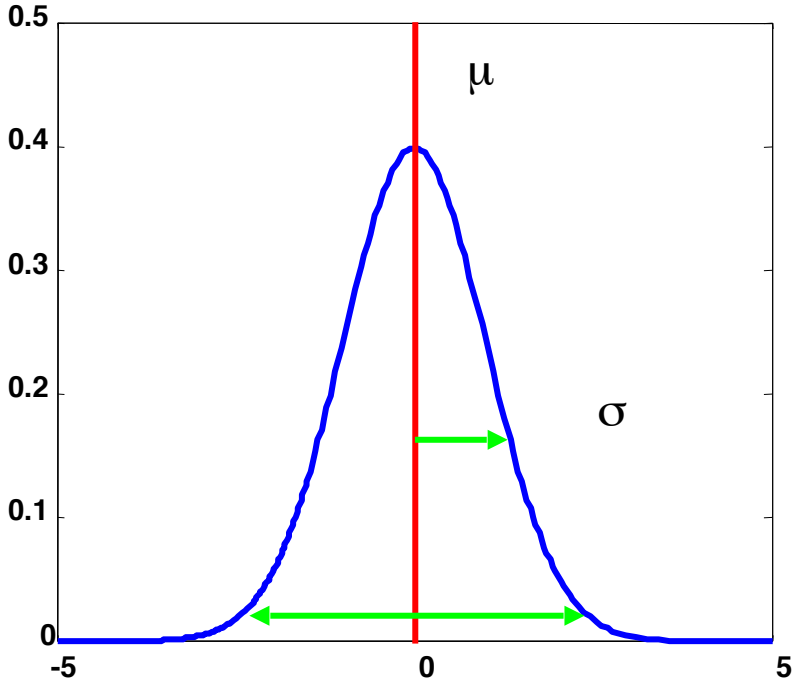
Density Estimation Approaches

- Parametric :
  assume simple global model, e.g. Gaussian,
  and estimate its parameters
- Non-parametric :
  assume simple local model, e.g. uniform, Gaussian



TU Delft

Gaussian Distribution



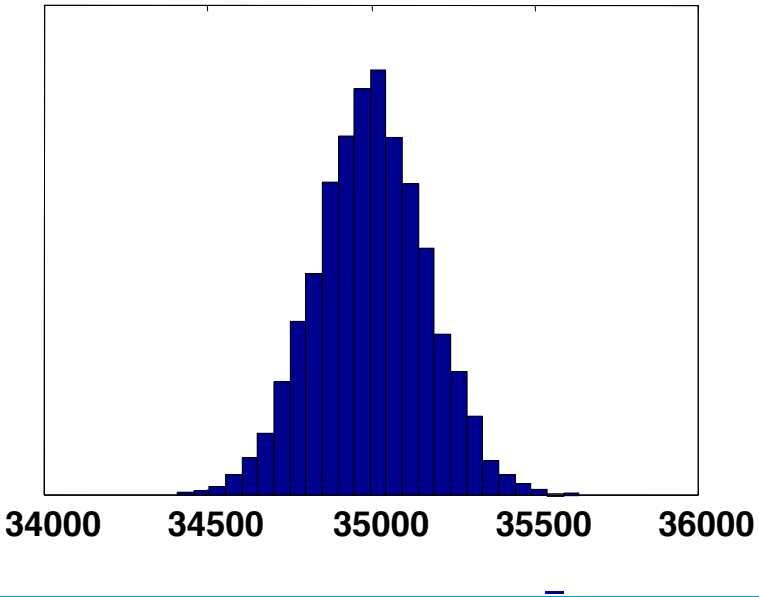
- Normal distribution = Gaussian distribution
- Standard normaldistribution: $\mu = 0, \sigma^2 = 1$
- 95% of data between $[\ \mu - 2\sigma, \mu + 2\sigma\ ]$ [in 1D!]

- 1D : $$p(x) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{1}{2} \frac{(x-\mu)^2}{\sigma^2} \right)$$

Gaussian Distribution

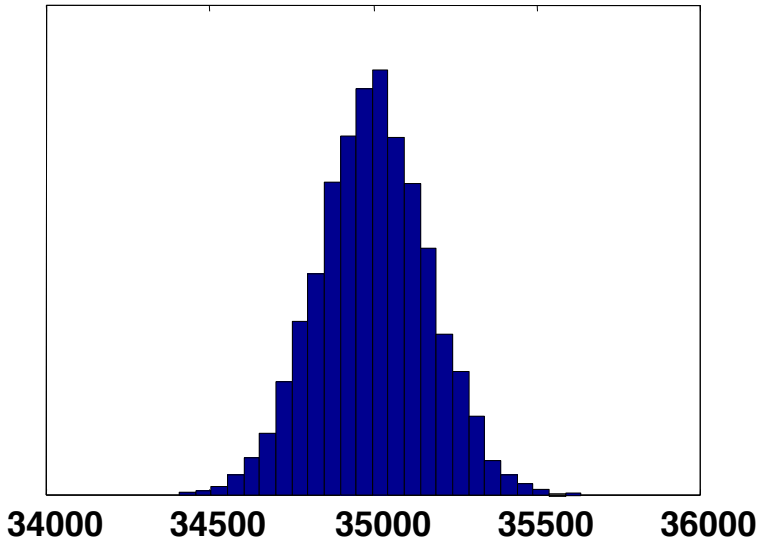
- Why Gaussians?
  - Special distribution : Central limit theorem says that sums of large numbers of i.i.d. [independent, identically distributed] random variables will have a Gaussian distribution
  - May actually occur in real life [approximately]

    E.g. sum of eyes of 10000 dice throws



Gaussian Distribution

- Why Gaussians [continued]?
  - Simple, few parameters
  - Easy to estimate these parameters when using maximum likelihood!



Multivariate Gaussians

- p - dimensional density :

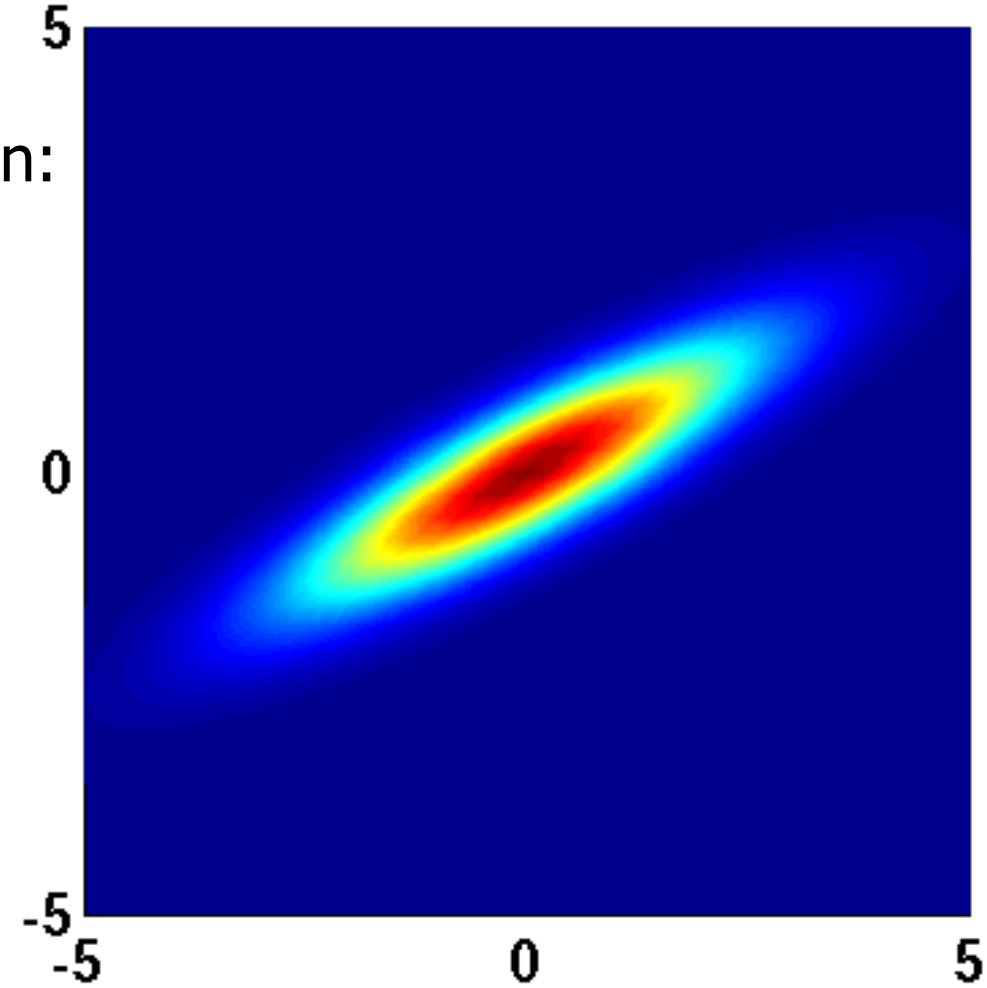
$$p(\mathbf{x}) = \frac{1}{\sqrt{(2\pi)^p \det(\Sigma)}} \exp\left(-\frac{1}{2}(\mathbf{x}-\mu)^T \Sigma^{-1} (\mathbf{x}-\mu) \right)$$

Example 2D Gaussian

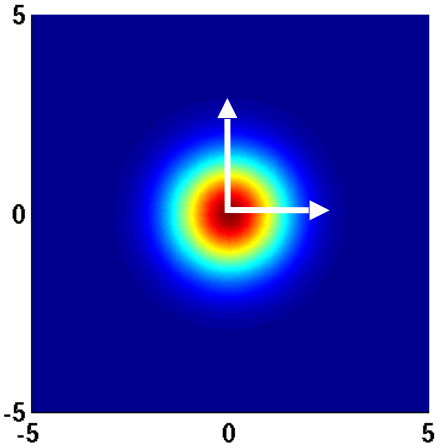
- Top view on an example 2D Gaussian:

$$\mu = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

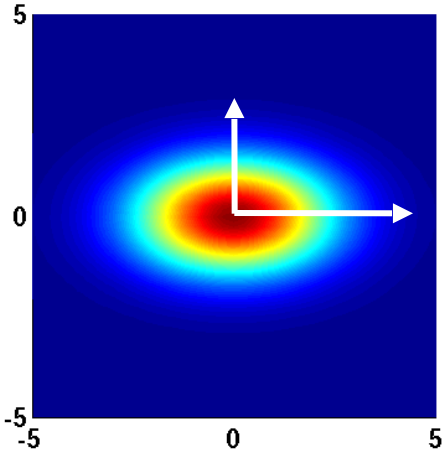
$$\Sigma = \begin{bmatrix} 3 & 1.5 \\ 1.5 & 2 \end{bmatrix}$$



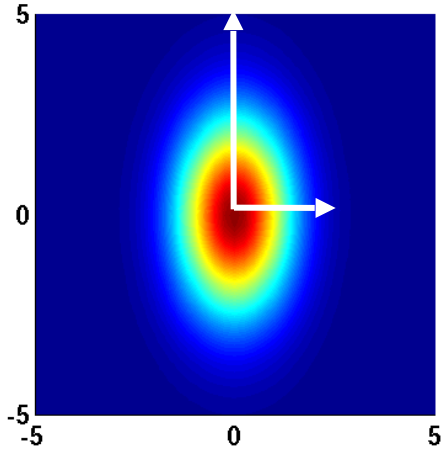
More examples of 2D Gaussians



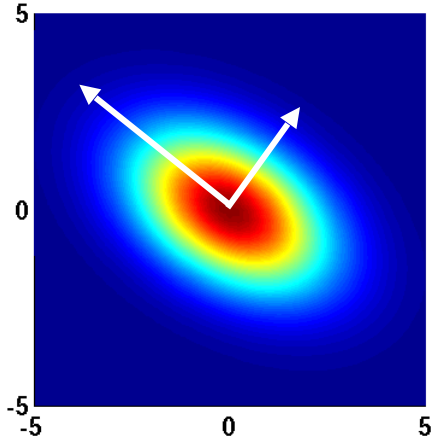
$$\Sigma = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$



$$\Sigma = \begin{bmatrix} 3 & 0 \\ 0 & 1 \end{bmatrix}$$



$$\Sigma = \begin{bmatrix} 1 & 0 \\ 0 & 3 \end{bmatrix}$$



$$\Sigma = \begin{bmatrix} 3 & -1 \\ -1 & 1 \end{bmatrix}$$

Maximum likelihood estimates?

- What are the maximum likelihood estimators for
  the mean
  the covariance matrix?

Maximum likelihood estimates?

- What are the maximum likelihood estimators for the mean
  the covariance matrix?

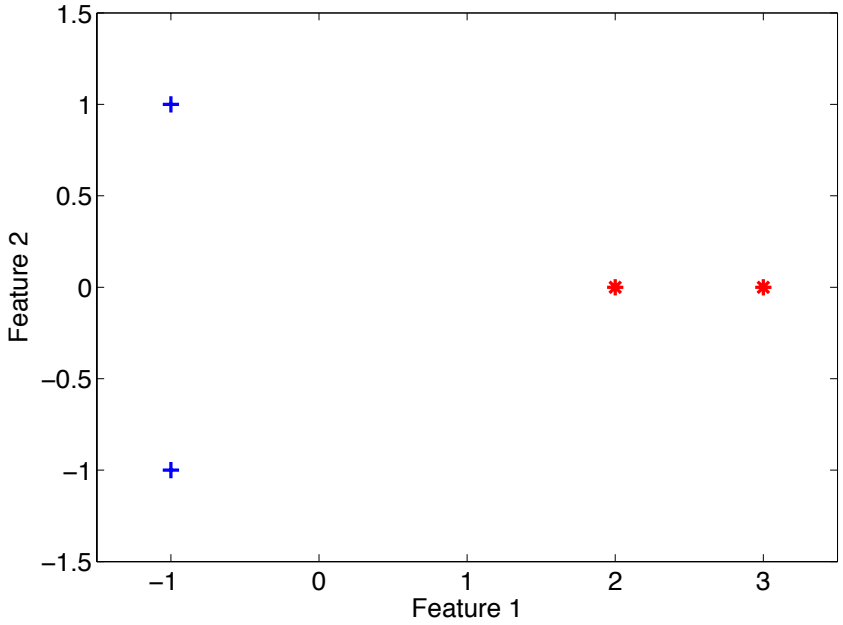
$$\hat{\mu} = \frac{1}{n}\sum_{i=1}^{n}\mathbf{x}_i$$

$$\hat{\Sigma} = \frac{1}{n}\sum_{i=1}^{n}(\mathbf{x}_i - \hat{\mu})(\mathbf{x}_i - \hat{\mu})^T$$

Estimating the covariance matrix

$$X = \begin{bmatrix} -1 & -1 \\ -1 & +1 \\ 2 & 0 \\ 3 & 0 \end{bmatrix}$$

$$\mathbf{y} = \begin{bmatrix} -1 \\ -1 \\ +1 \\ +1 \end{bmatrix}$$



- Mean of class +1?
- Covariance matrix of class +1? And its inverse?

Estimating the mean

- Objects of class +1: stored in rows:

$$\mathbf{x}_1 = \begin{bmatrix} 2 \\ 0 \end{bmatrix} \qquad \mathbf{x}_2 = \begin{bmatrix} 3 \\ 0 \end{bmatrix}$$

- Then:

$$\hat{\mu} = \frac{1}{n} \sum_{i=1}^{n} \mathbf{x}_i = \frac{1}{2} \left(\begin{bmatrix} 2 \\ 0 \end{bmatrix} + \begin{bmatrix} 3 \\ 0 \end{bmatrix} \right) = \begin{bmatrix} 2.5 \\ 0 \end{bmatrix}$$

Estimating the covariance matrix

- Make new, centered, dataset:

$$\hat{\Sigma} = \frac{1}{n}\sum_{i=1}^{n}(\mathbf{x}_i - \hat{\mu})(\mathbf{x}_i - \hat{\mu})^T = \frac{1}{2}\sum_{i=1}^{n}\tilde{\mathbf{x}}_i\tilde{\mathbf{x}}_i^T$$

- Then:

$$\tilde{\mathbf{x}}_1 = \begin{bmatrix} -0.5 \\ 0 \end{bmatrix} \quad \tilde{\mathbf{x}}_2 = \begin{bmatrix} 0.5 \\ 0 \end{bmatrix}$$

- And:

$$\tilde{\mathbf{x}}_1\tilde{\mathbf{x}}_1^T = \begin{bmatrix} 0.25 & 0 \\ 0 & 0 \end{bmatrix}$$

Estimating the covariance matrix

- The same for $\tilde{\mathbf{x}}_2 \tilde{\mathbf{x}}_2^T = \begin{bmatrix} 0.25 & 0 \\ 0 & 0 \end{bmatrix}$

- So in total:

$$\hat{\Sigma} = \frac{1}{2} \sum_{i=1}^{n} \tilde{\mathbf{x}}_i \tilde{\mathbf{x}}_i^T = \frac{1}{2} \left(\begin{bmatrix} 0.25 & 0 \\ 0 & 0 \end{bmatrix} + \begin{bmatrix} 0.25 & 0 \\ 0 & 0 \end{bmatrix} \right)$$

$$= \begin{bmatrix} 0.25 & 0 \\ 0 & 0 \end{bmatrix}$$

Inverse of the covariance matrix?

- Inverse of?: $\hat{\Sigma} = \begin{bmatrix} 0.25 & 0 \\ 0 & 0 \end{bmatrix}$
- It should hold that: $\hat{\Sigma}\hat{\Sigma}^{-1} = \mathbb{I} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$
- Can you find a,b,c, such that?

$$\begin{bmatrix} 0.25 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} a & b \\ b & c \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

Inverse of the covariance matrix?

- Inverse of?: $\hat{\Sigma} = \begin{bmatrix} 0.25 & 0 \\ 0 & 0 \end{bmatrix}$
- It should hold that: $\hat{\Sigma}\hat{\Sigma}^{-1} = \mathbb{I} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$
- Can you find a,b,c, such that?

$$\begin{bmatrix} 0.25 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} a & b \\ b & c \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

NO, not possible
Not invertible

Inverse of the covariance matrix?

- To solve $\begin{bmatrix} 0.25 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} a & b \\ b & c \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$

  you need to get some 'c' for which:

$$0 \times c = 1$$

- Not possible!
- Number of objects is insufficient to find the inverse
- Two objects in a 2-dimensional feature space: a degenerate Gaussian distribution

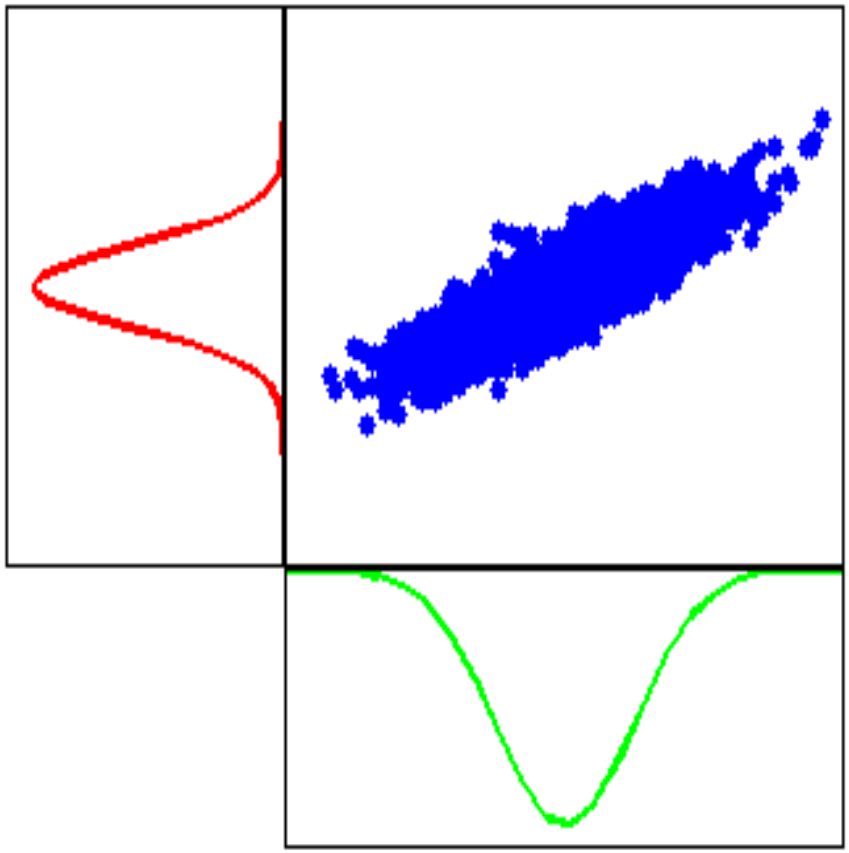
Parametric Estimation

- Sounds simple, but for p-dimensional data set :
  - μ : vector with p elements
  - Σ : matrix with $0.5\ p\ (p+1)$ elements

- Number of parameters increases quadratically with p : might still need lots of data for high-dimensional data

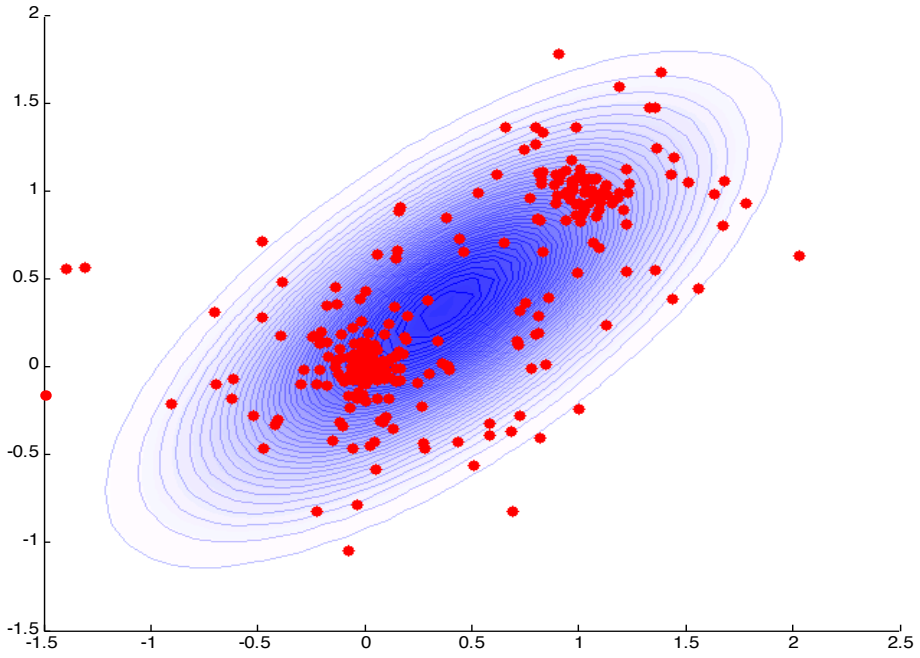
Special Properties

- Any projection of a high-dimensional Gaussian is itself again Gaussian



Parametric Estimation

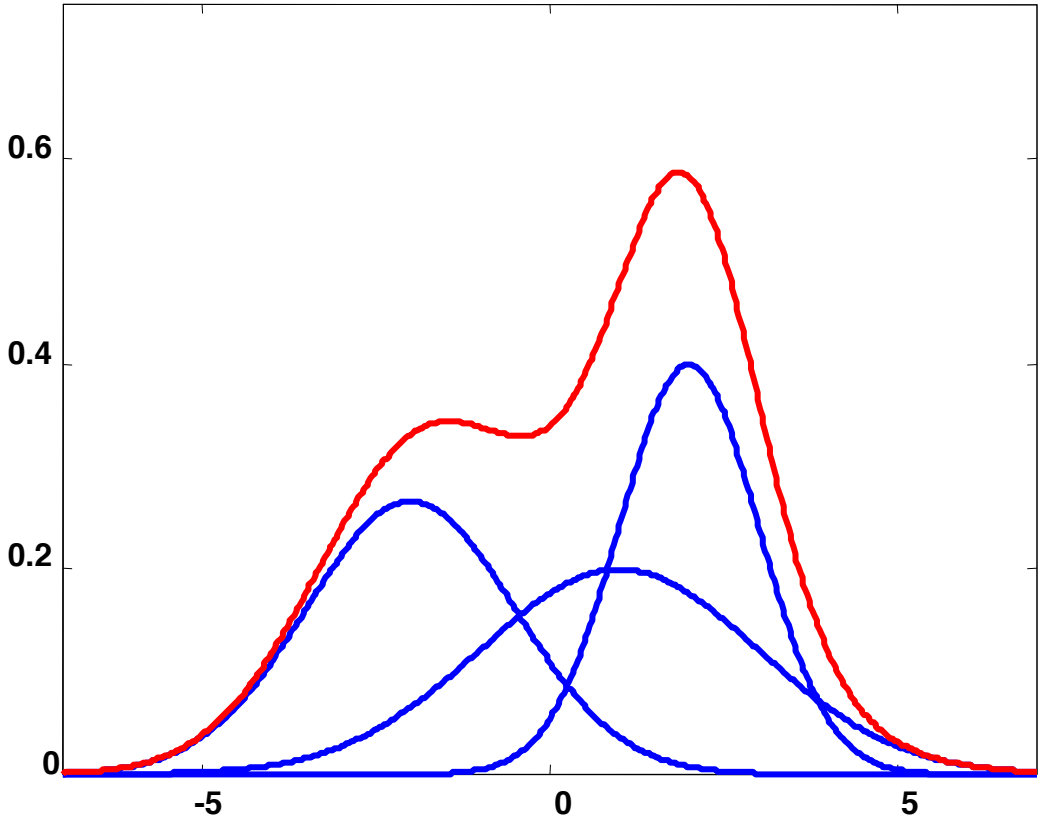
- Assume model, e.g. Gaussian
- Estimate mean μ and covariance Σ from data



TU Delft

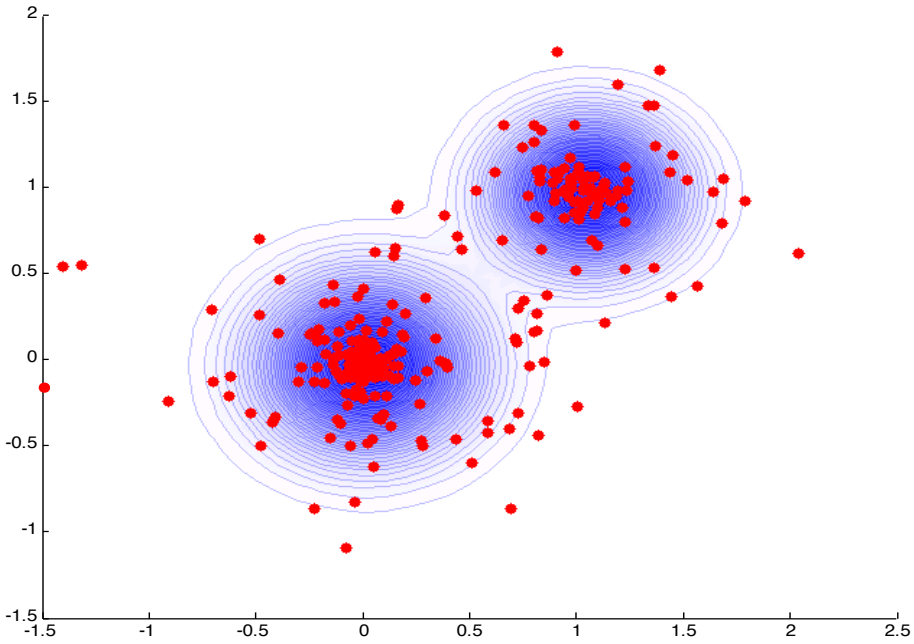
Gaussian Distribution

- Not necessarily too restrictive : mixture models



Mixture models

- Model : mixture of 2 Gaussians
- More parameters: π_i, $\boldsymbol{\mu}_i$ and $\boldsymbol{\Sigma}_i$, $i = 1,2$



Mixture models

- Estimating parameters not straightforward:

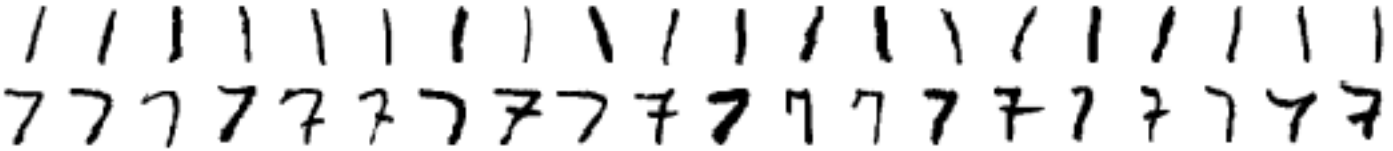
$$p(\mathbf{x}) = \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}; \mu_k, \Sigma_k)$$

- If you know memberships of objects to clusters, you can estimate means+cov.'s
- You only know the memberships if you have the means+cov.'s

- EM-algorithm: see course CS4220 Machine Learning 1

One may Wonder…

- In parametric case, why only consider multivariate Gaussian?
- When using histograms, what seems one of the underlying assumptions?
- With increasing dimensions / # measurements, does Gaussian or histogram need more data?
- When does one have enough data?
- What if one class has multiple modes, and the other not?



Recapitulation

- Up to now:
  - Without models no generalization
  - Density estimation: core of statistical learning, really hard problem
  - Curse of dimensionality: adding measurements does not always help
  - Parametric density estimation: Gaussian

- Next:
  - Classification using Gaussian densities

After this week you should be able to:

- explain how you obtain a classifier using a Gaussian (multivariate) distribution for each class
- implement a simple univariate classifier in Python
- explain what the 'curse of dimensionality' is

- explain the advantages and disadvantages are of the Quadratic classifier, the LDA and the nearest mean classifier
- identify when scaling of the features is important and how to cope with feature scaling