BioSB
research school

Machine Learning for Bioinformatics & Systems Biology

1. Introduction, density estimation & classification

Perry Moerland *Amsterdam UMC, University of Amsterdam*

Marcel Reinders *Delft University of Technology*

Lodewyk Wessels *Netherlands Cancer Institute*

Some material courtesy of Robert Duin, David Tax & Dick de Ridder

Programme

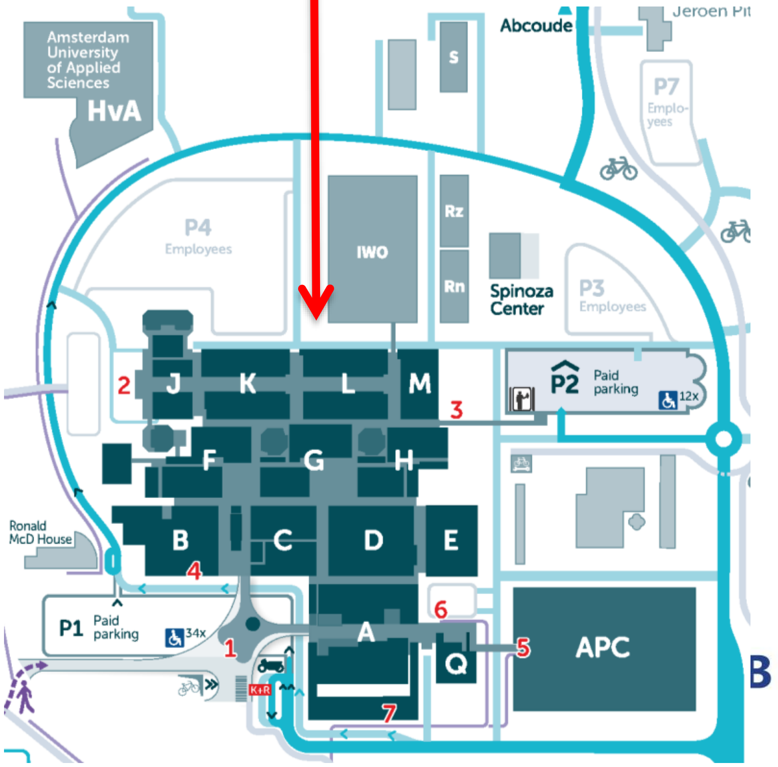
Day	Lecturer	Subjects
Monday 20/1	Perry Moerland	Introduction to machine learning Density estimation Bayesian classification
Tuesday 21/1	Perry Moerland	Parametric and nonparametric classifiers Decision trees & random forests Hierarchical clustering Agglomerative clustering EM and model-based clustering
Wednesday 22/1	Lodewyk Wessels	Feature extraction Embeddings Feature selection Sparse classifiers
Thursday 23/1	Marcel Reinders	Artificial neural networks Support vector machines Classifier ensembles Complexity
Friday 24/1	Marcel Reinders Students Invited speaker	Variational autoencoders Diffusion models Student pitches Invited speaker (application of classification)

Schedule

L0-227

When	What	Where
9.00-12.00	Course	L0-227
12.00-13.00	Lunch break	The Box (G0-114)
13.00-17.00	Course	L0-227

- Coffee/tea etc. and lunch will be provided
- Thursday there will be drinks, bites and a quiz at 17.00 in Miss Scarlett (at 5 minutes walking distance from the AMC)
- **Friday: J1B-223**



Certificates and examination

- To obtain a certificate of successful completion:
  - Analyse a biological dataset (preferably one from your own practice) using the tools provided in the course
  - Write a short report (5-10 pages) on the results
  - Hand this in no later than **February 14, 2025 (3 weeks after end of course)**
- If you have no dataset available, one will be provided
- Grade will be “pass” or “fail”, with at most one resubmission
- If no report or “fail”: certificate of attendance

BioSB: The Netherlands Bioinformatics and Systems Biology research school

- Yearly conference: 20-21 May 2025 (https://www.aanmelder.nl/biosb2025)

- Courses (https://www.dtls.nl/biosb/courses/):
  - Constraint-based modeling, 10-14 February 2025
  - Algorithms for biomolecular networks, 28 April – 2 May 2025
  - Knowledge graphs in the life sciences, Fall 2025
  - Algorithms for genomics, Fall 2025

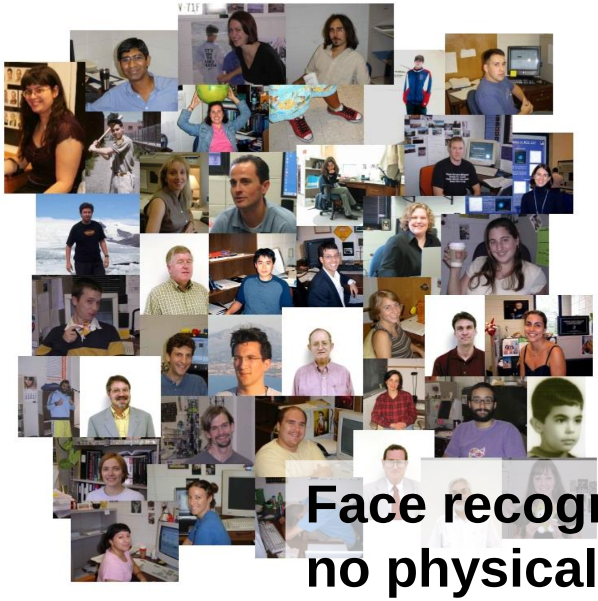
- YoungCB: Regional Student Group (RSG) Netherlands of the International Society of Computational Biology (https://www.dtls.nl/youngcb/)

Course

Machine learning

- The construction of **approximate, generalizing (predictive) models** by **learning from examples**, for problems for which *no full physical model is known* (yet)
- Focus in this course will be on **classification** and **statistical machine learning**, not (so much) on *regression, structural/syntactic* pattern recognition and *reinforcement learning*.

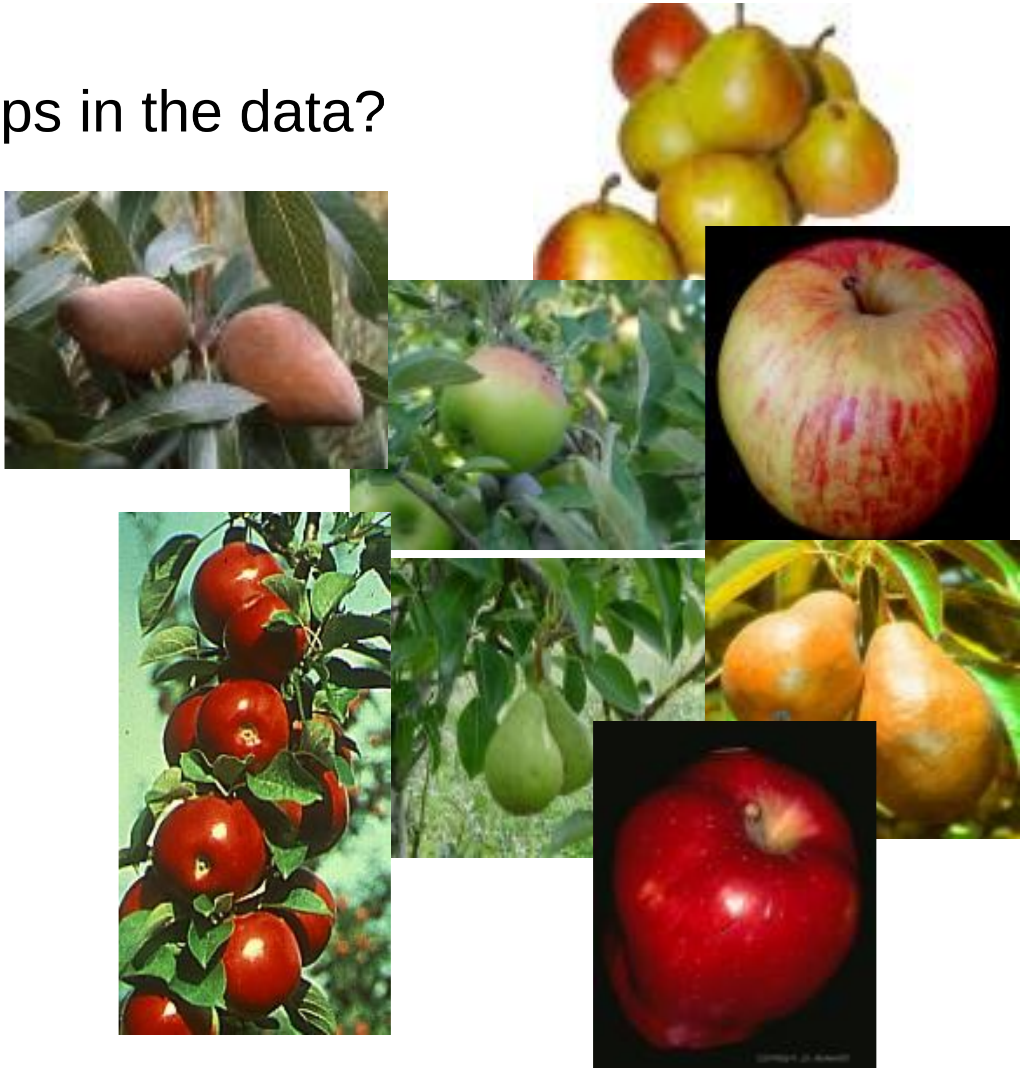
- Related areas
  - Applied statistics
  - Pattern recognition
  - Artificial intelligence
  - Computer vision
  - Data mining



Face recognition, no physical model

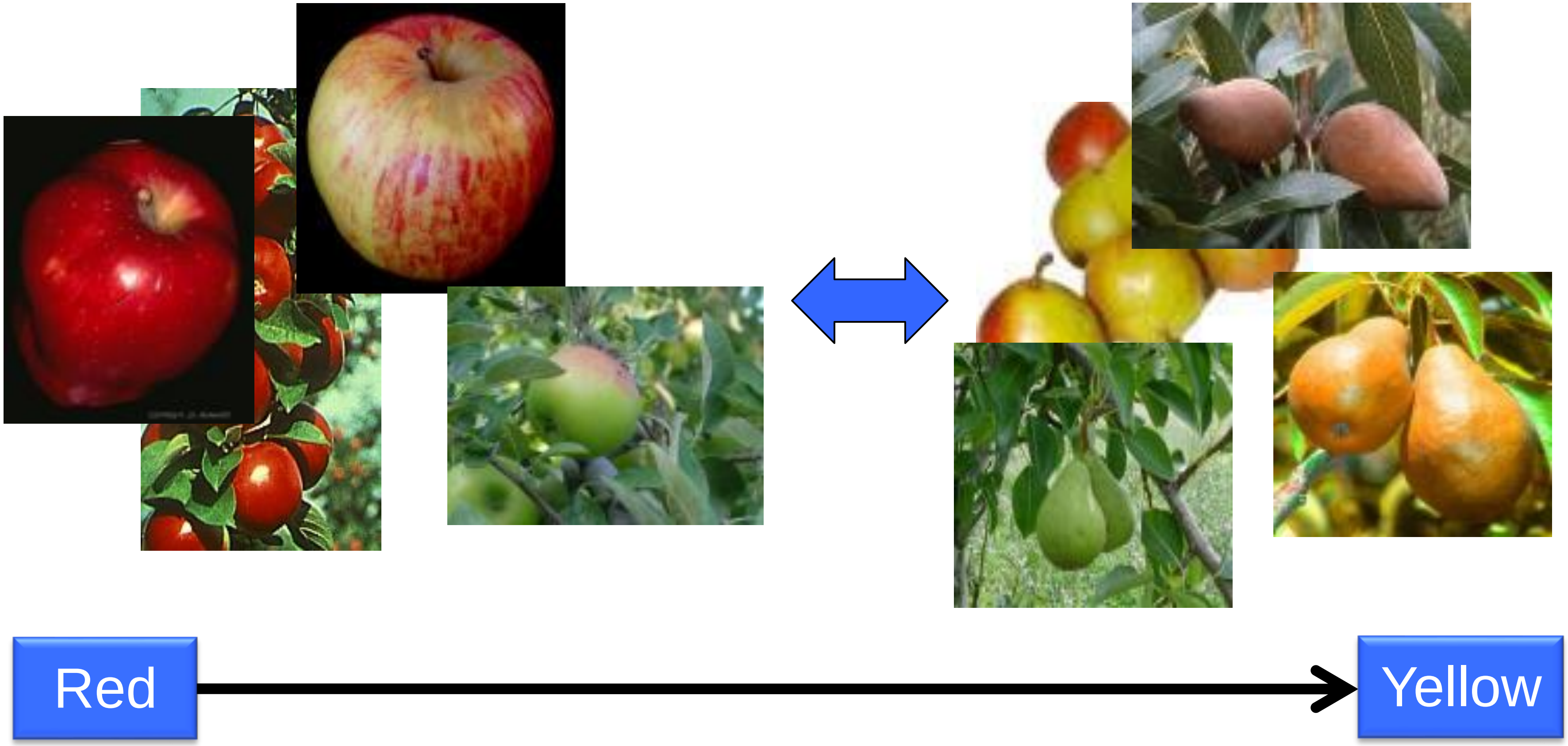
Clustering

- Can we find natural groups in the data?
- E.g. red vs green fruit



Dimensionality reduction

- Can we find predictive features?



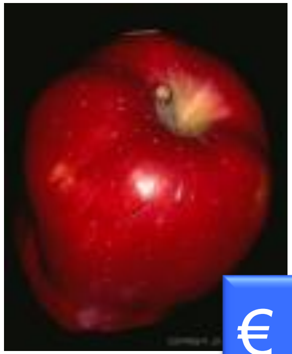
Red

Yellow

Regression

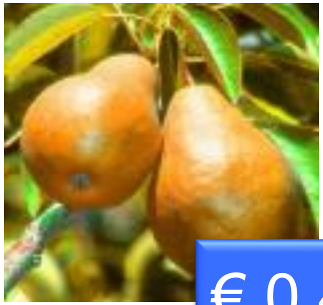
- Can we predict real-valued outputs?

€ 0.25



€ 0.30

€ 0.40

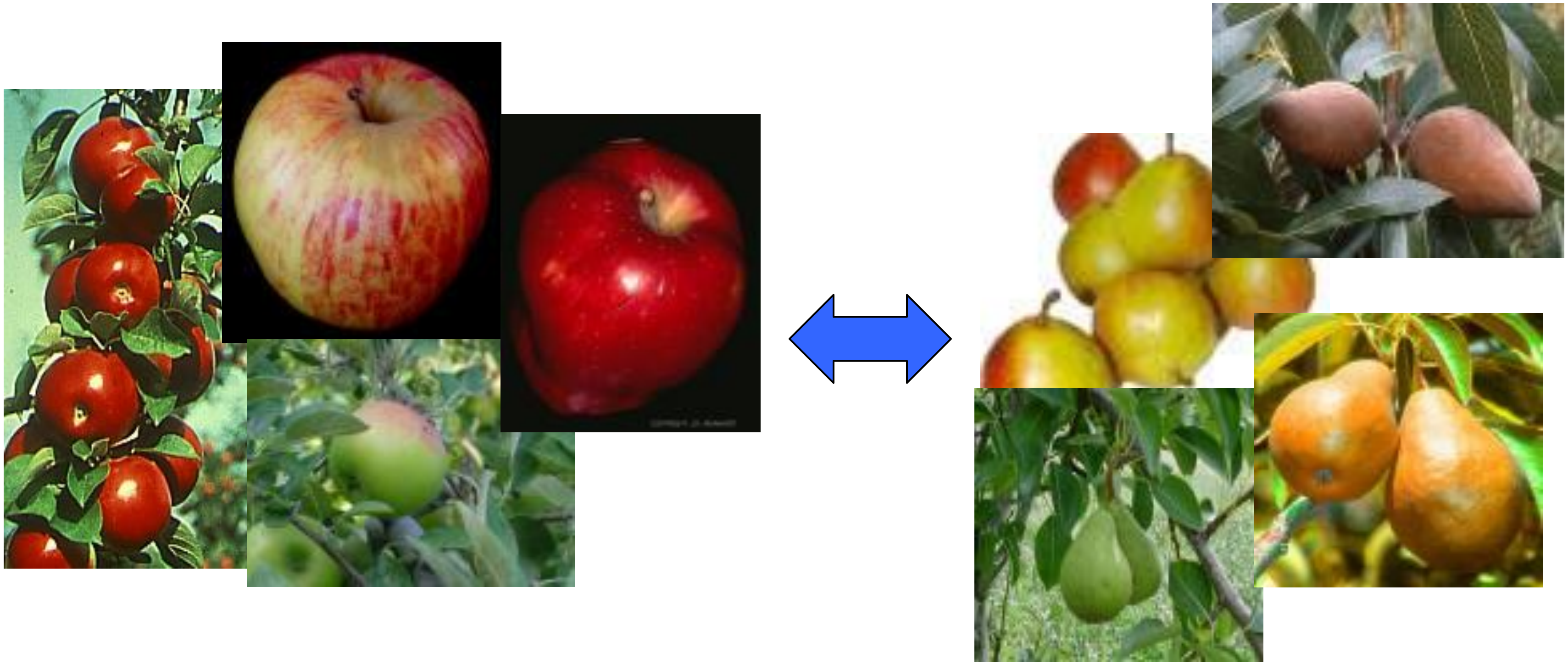


€ 0.45

BioSB

Classification

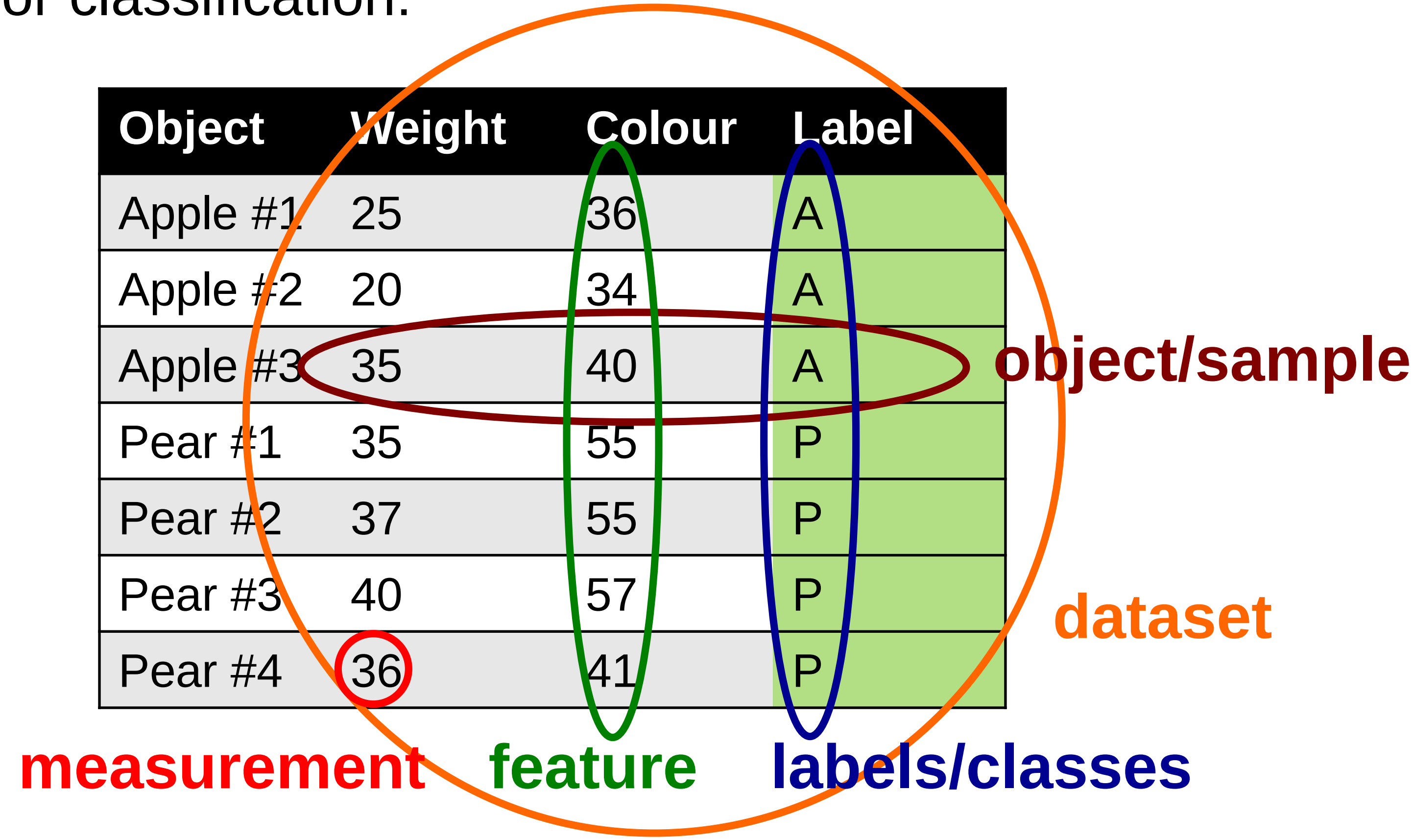
- Can we distinguish apples from pears?



Datasets

- A *dataset* is a set of measurements on many objects

- For classification:



Object	Weight	Colour	Label
Apple #1	25	36	A
Apple #2	20	34	A
Apple #3	35	40	A
Pear #1	35	55	P
Pear #2	37	55	P
Pear #3	40	57	P
Pear #4	36	41	P

object/sample

dataset

measurement

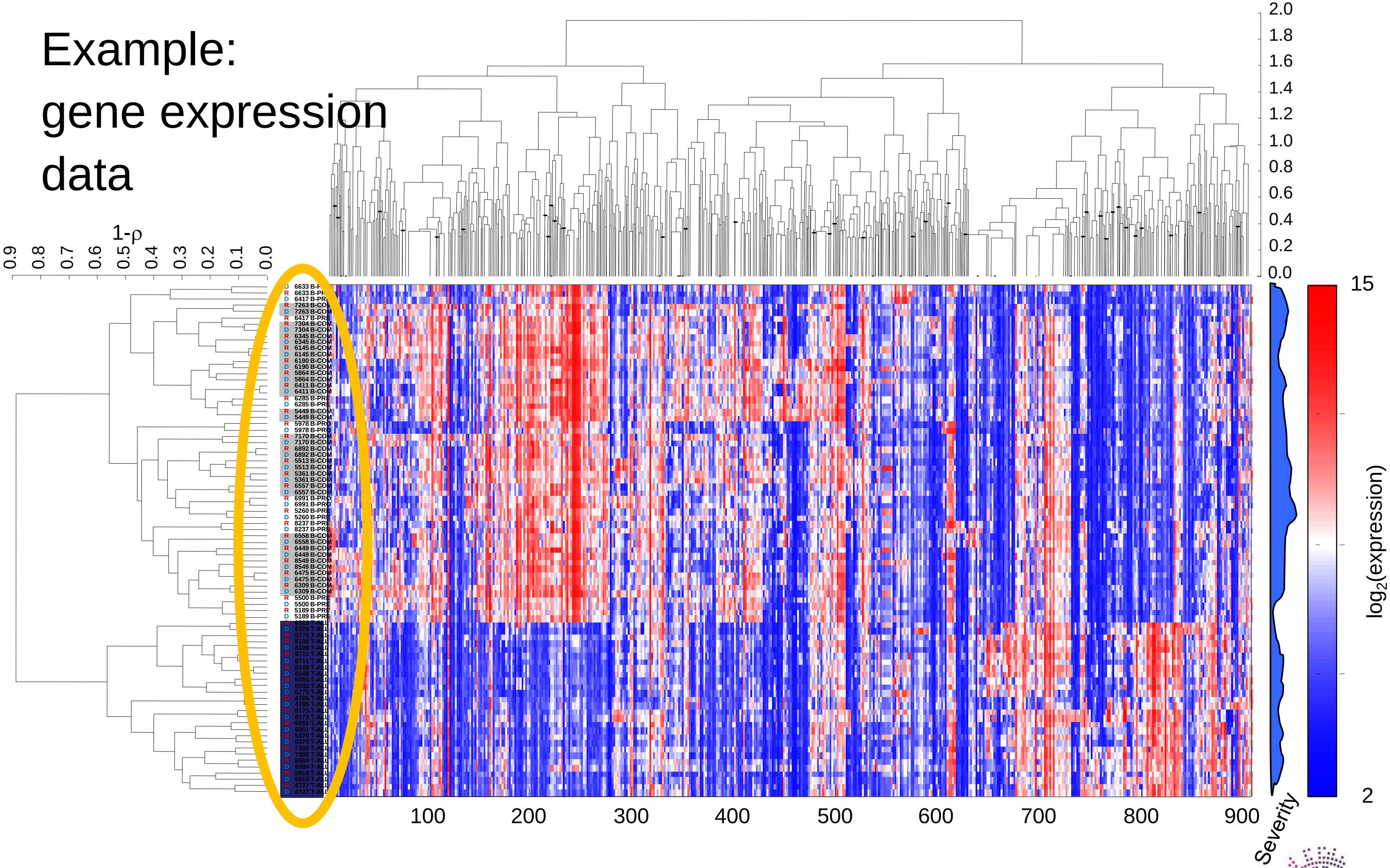
feature

labels/classes

Exercise 1.1-1.9

Classification in bioinformatics

- Example:
  gene expression
  data

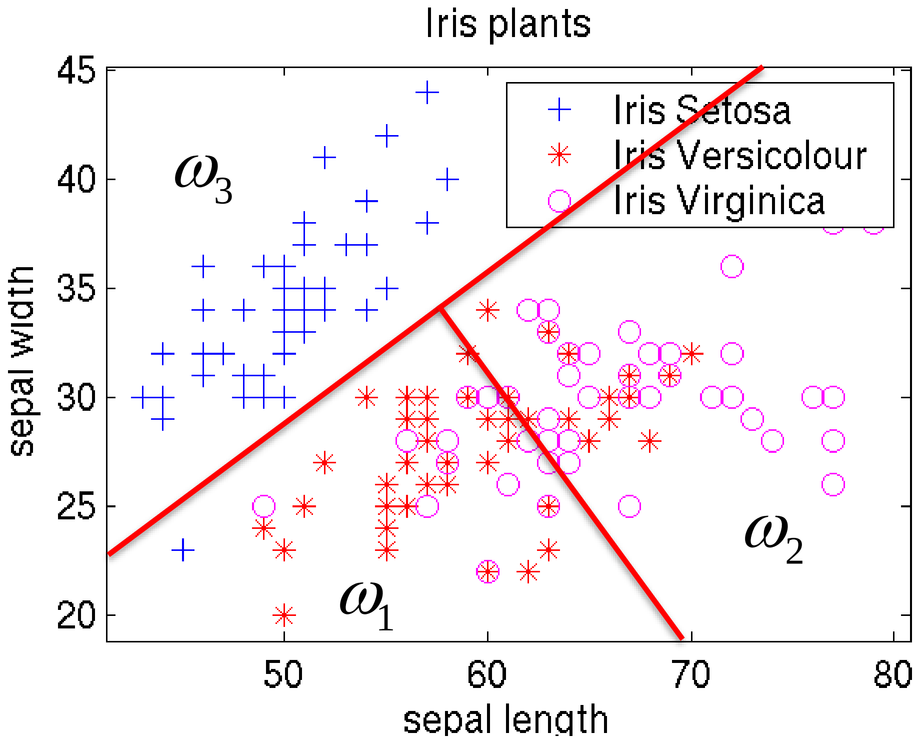


- Note: theory applies to any type of data!

E.g. Predicting metastasis

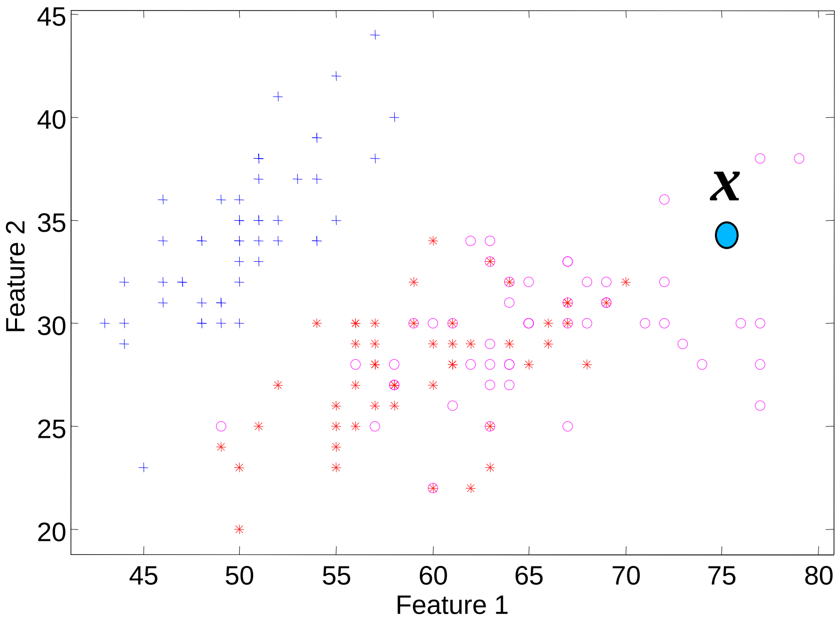
Classification (2)

- Given labeled data ***x***,
  assign each point in feature space to a class ω_i
  (in effect partitioning the feature space)



General model

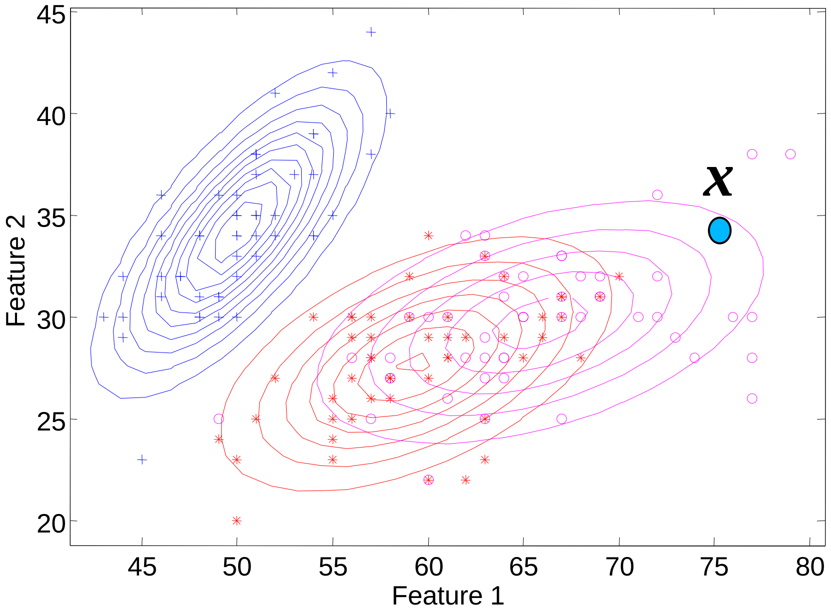
- Construct a model $f(\mathbf{x})$ that outputs ω or y
- This model should be fit to the data



$f(\mathbf{x}) = \omega$ or $f(\mathbf{x}) = y$

General model (2)

- Construct a model $f(\mathbf{x})$ that outputs ω or y
- This model should be fit to the data
- Ideally, we know $p(y \mid \mathbf{x})$ or $p(\omega \mid \mathbf{x})$ over the entire feature space



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

or

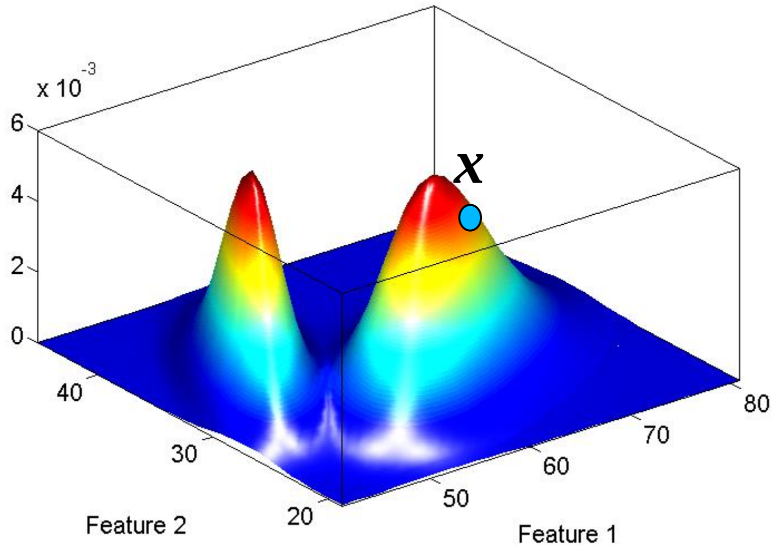
$p(\omega \mid \mathbf{x})$

$f(\mathbf{x}) = \omega$ or $f(\mathbf{x}) = y$

if we know the probability distributions, we can make the most informed decision

General model (3)

- Construct a model $f(\mathbf{x})$ that outputs ω or y
- This model should be fit to the data
- Ideally, we know $p(y \mid \mathbf{x})$ or $p(\omega \mid \mathbf{x})$ over the entire feature space



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

or

$p(\omega \mid \mathbf{x})$

→ $f(\mathbf{x}) = \omega$ or $f(\mathbf{x}) = y$

if we know the probability distributions, we can make the most informed decision

General model (4)

- Clustering: find cluster labels ω given object $\boldsymbol{x}$
  fit model using dataset $\{\boldsymbol{x}_i\}$ $$p(\omega \mid \boldsymbol{x})$$

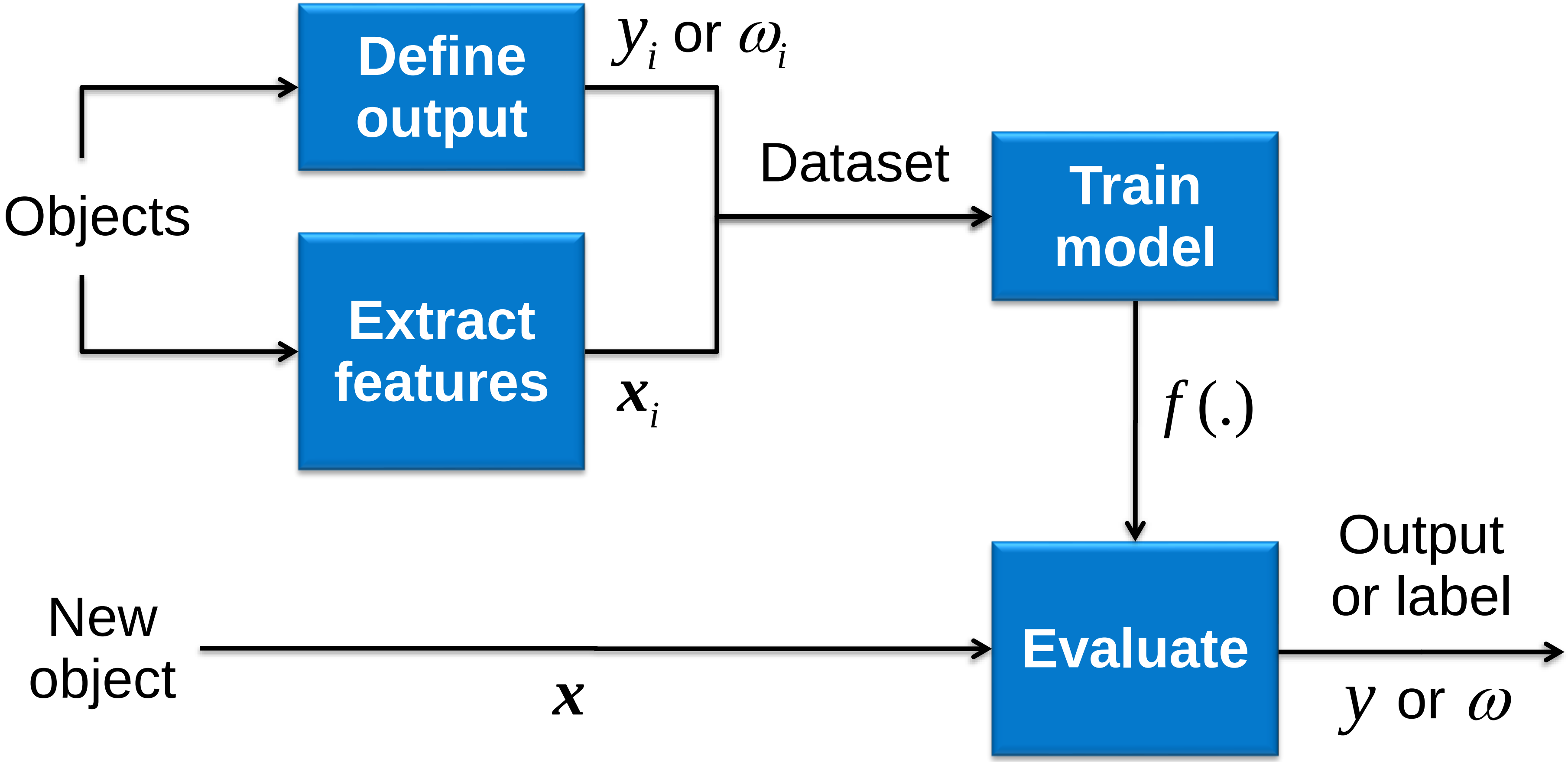
- Dimensionality reduction: find mapping y given object $\boldsymbol{x}$
  fit model using dataset $\{\boldsymbol{x}_i\}$ $$p(y \mid \boldsymbol{x})$$

- Classification: find class labels ω given object $\boldsymbol{x}$
  fit model using dataset $\{\boldsymbol{x}_i, \omega_i\}$ $$p(\omega \mid \boldsymbol{x})$$

- Regression: find target y given object $\boldsymbol{x}$
  fit model using dataset $\{\boldsymbol{x}_i, y_i\}$ $$p(y \mid \boldsymbol{x})$$

Statistical machine learning

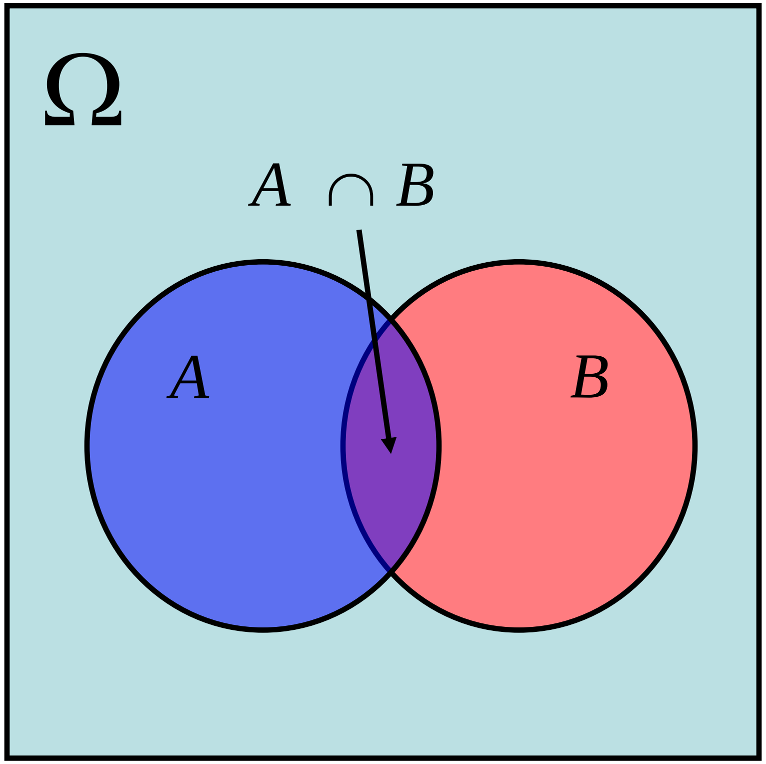
Machine learning pipeline



Statistics and Bayesian estimation

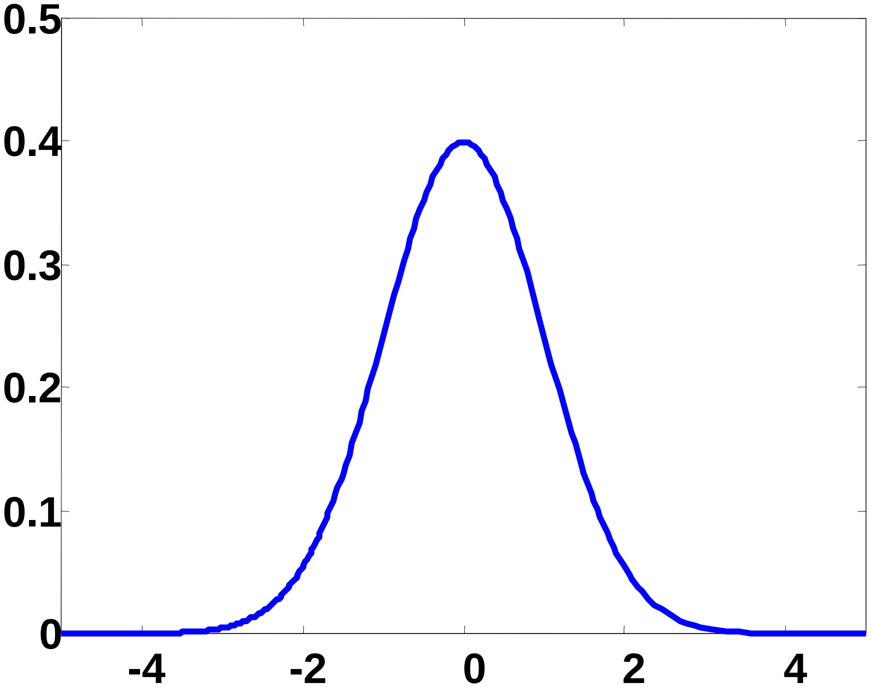
Recall: probability

- Ω : all possible outcomes (sample space)
  e.g. the number of eyes on a dice: 1, 2, 3, 4, 5, 6
- $A \in \Omega$: event
  e.g. “throwing a 3”
- P : probability measure
  - $0 \leq P(A) \leq 1$
  - $P(\Omega) = 1$
  - $P(A \cup B) =$
    $P(A) + P(B) - P(A \cap B)$
  - E.g. $P(A) = 1/6$



Recall: PDFs

- $p(x) = \dfrac{dP(x)}{dx}$: probability density function
  - $p(x) \geq 0$
  - $\int_{-\infty}^{\infty} p(x)dx = 1$
  - $\int_{a}^{b} p(x)dx =$ $P(a \leq x \leq b)$



- ***p*(*x*) is not the probability of *X* being *x* !**

Recall: Bayes' theorem

- Conditional probability of A given B,

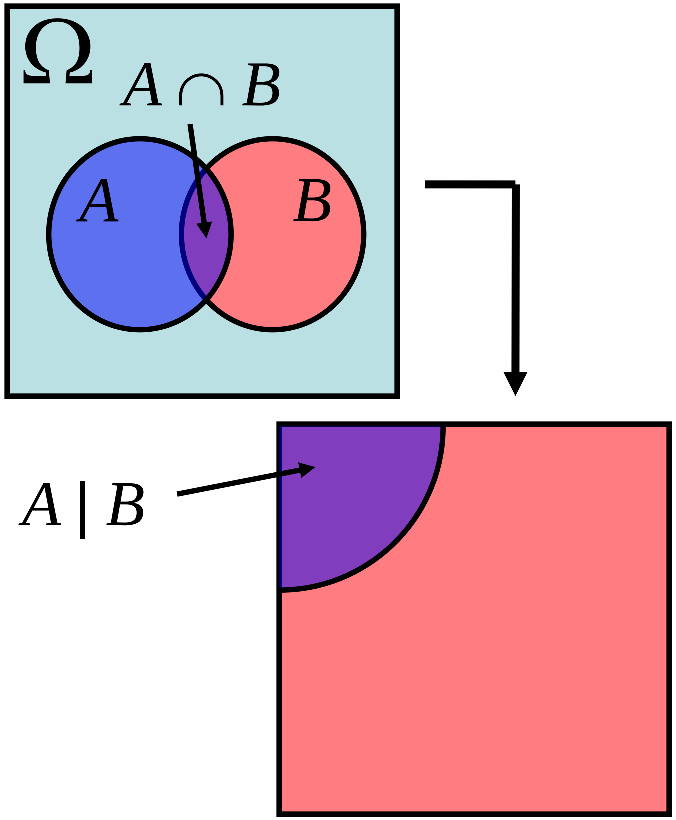
$$P(A|B) = \frac{P(A \cap B)}{P(B)}$$

- As a consequence,

$$\begin{aligned} P(A \cap B) &= P(A|B)P(B) \\ &= P(B|A)P(A) \end{aligned}$$

- Bayes' theorem:

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$



Bayes' theorem (2)

- Bayes' theorem is very useful, but controversial:
  - reverses causality
  - introduces subjective (prior) probabilities

$$P(cause \mid effect) = \frac{P(effect \mid cause)P(cause)}{P(effect)}$$

- **… but the cornerstone of pattern recognition and machine learning**
  - $P(disease|temperature) = \frac{P(temperature|disease)P(disease)}{P(temperature)}$
  - What is P (disease)? How to measure / know?

Bayes' theorem (3)

- In statistical learning, we want to know $p(y \mid x)$ so that we can predict (for example) the most probable output y for a given input x

- Problem: this is often very hard to model or estimate...
  - Predict gender based on height measurement: $p(\text{gender}|\text{height})$?
  - Predict fruit type based on color measurement: $p(\text{fruit}|\text{color})$?
  - Predict temperature based on thermometer reading: $p(\text{temperature}|\text{thermometer reading})$?

problem is that you need to measure too much:
***for every height** you need a number of examples of different genders*
feature = continuous & class label not

Bayes’ theorem (4)

- Solution: combine probabilities
  - y = cause, outcome, target, label (ω), ...
  - x = effect, measurement, feature, ...

$$\underbrace{p(y\,|\,x)}_{\substack{\textit{posterior}\\ \text{probability}}} = \frac{\overbrace{p(x\,|\,y)}^{\substack{\textit{conditional}\\ \text{probability}}}\overbrace{p(y)}^{\substack{\textit{prior}\\ \text{probability}}}}{\underbrace{p(x)}_{\textit{normalisation}}}$$

We update our prior belief (prior) using observations (conditional)

Bayes' theorem (5)

- Classification example $p(\omega \mid \boldsymbol{x})$:
  - $\omega \in$ { 'man', 'woman' } = label
  - $x \in \mathbb{R}^1$ = height measurement(m)
- $p(\omega)$: prior probability of seeing a 'man' or a 'woman' here: ...?
- $p(x|\omega)$: density of x (height) when the person is actually a 'man' or a 'woman'

  $p(x)$: density of height measurement x here (total probability):

$$p(x) = \sum_i p(x \mid \omega_i)\, p(\omega_i)$$

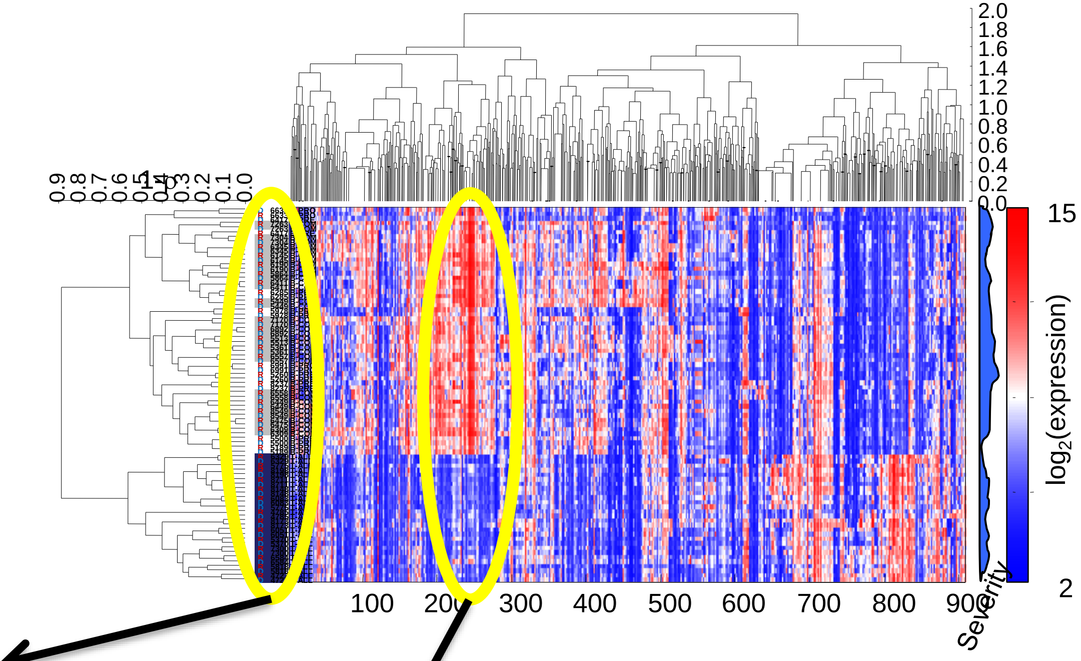
Issue: Prior for man/woman? In NL? In Delft? In classroom?

Bayesian estimation

- Estimate prior, $p(y)$, and conditional, $p(x|y)$
- Use this to obtain posterior, $p(y|x)$
- Construct a cost function $\Lambda(y',y)$:
  the cost of predicting y' when the true outcome is y
  - for classification: cost matrix
  - when all mistakes are equally bad:
    - $\Lambda(y',y) = 0$ when $y' = y$
    - $\Lambda(y',y) = 1$ otherwise

Bayesian classification

Classification

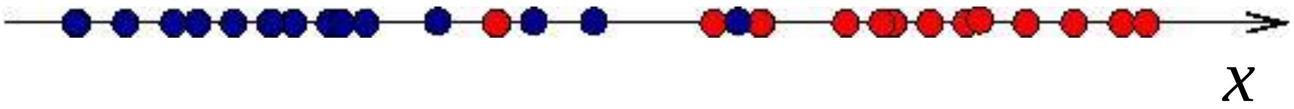


Labels ω

● healthy

● diseased

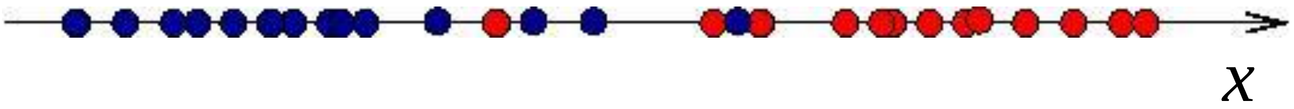
Measurements x



As example, consider a single gene expression measurement x

Posterior probability

- For each object, we have to estimate $p(\omega|x)$ or $p(y|x)$



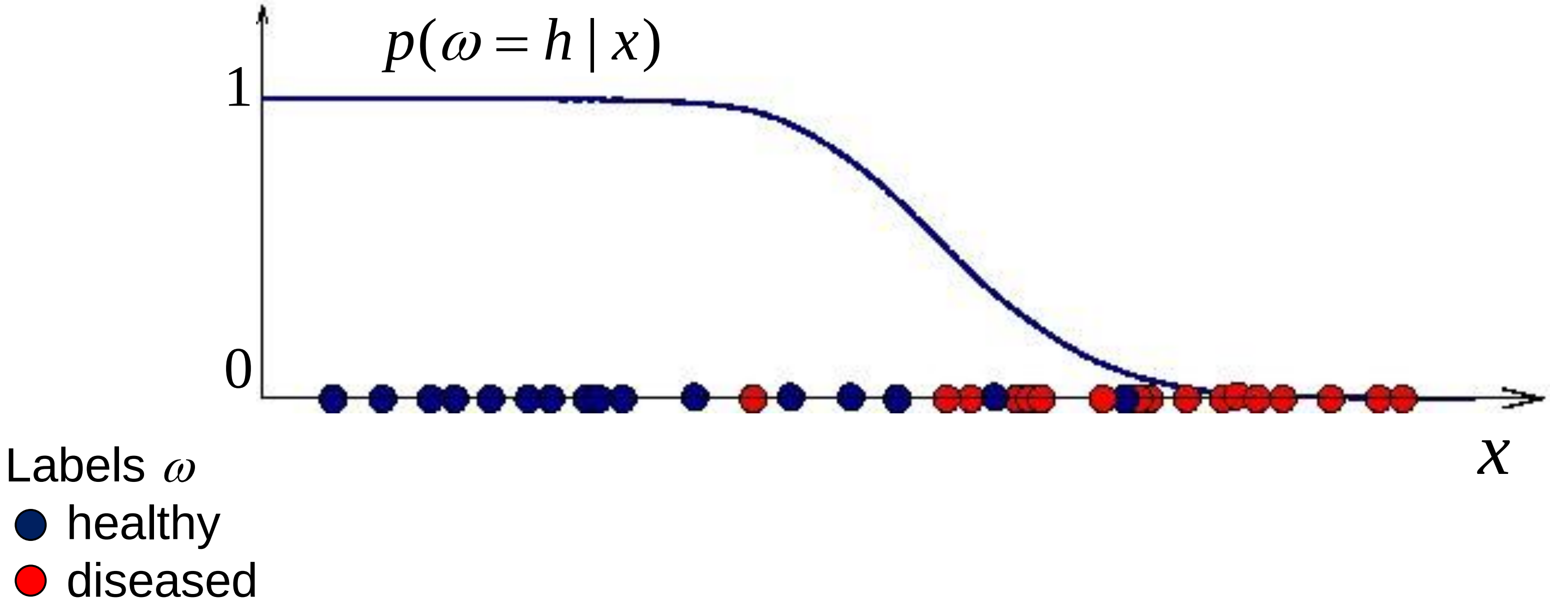
Labels ω

● healthy

● diseased

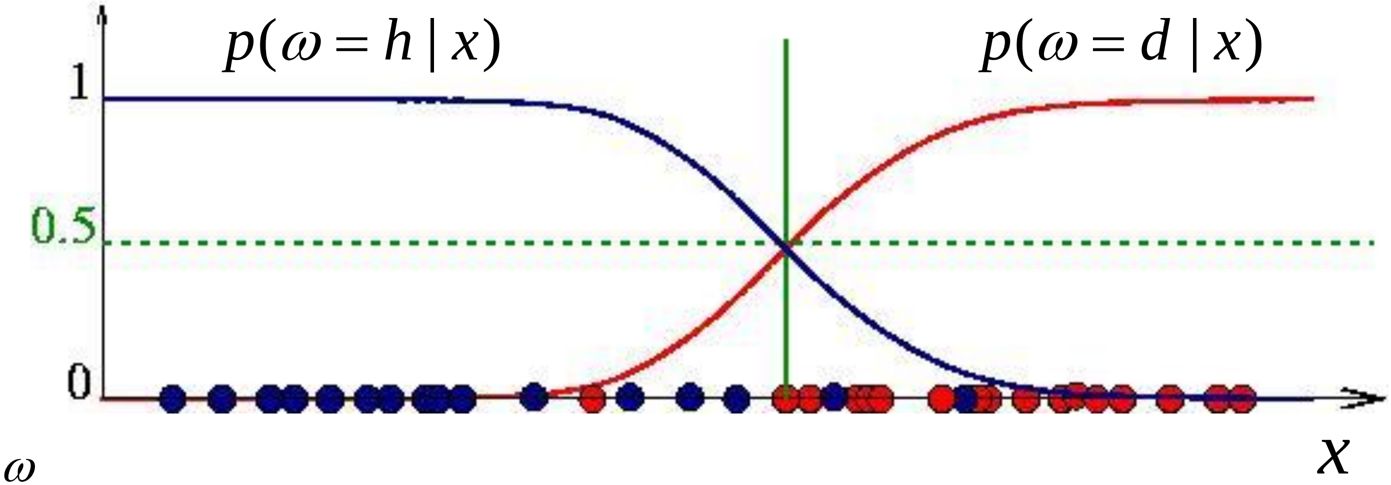
Posterior probability (2)

- For each object, we have to estimate $p(\omega|x)$ or $p(y|x)$



Posterior probability (2)

- For each object, we have to estimate $p(\omega|x)$ or $p(y|x)$



Labels ω

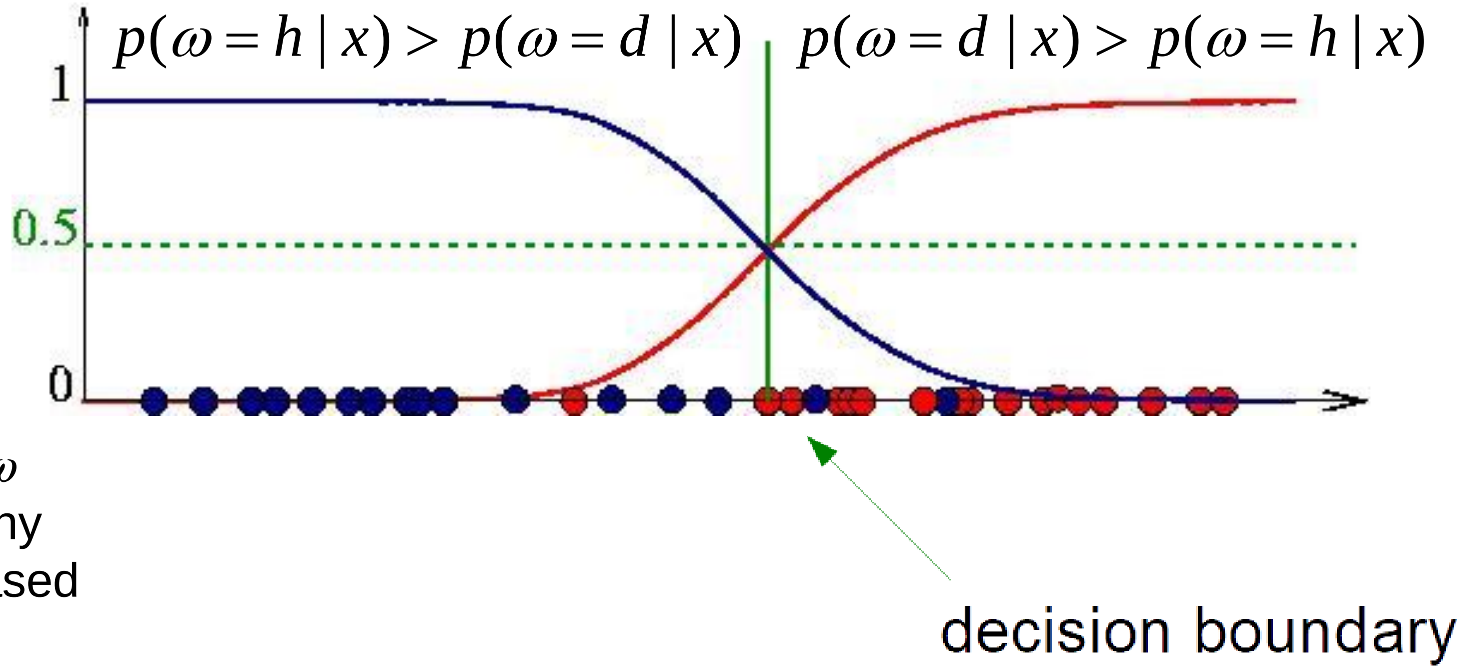
● healthy

● diseased

- Of course: $\sum_{c=1}^{C} p(\omega = c \mid x) = 1$

Posterior probability (3)

- For each object, we have to estimate $p(\omega|x)$ or $p(y|x)$



Labels ω

● healthy

● diseased

Assign label of class with the largest posterior probability

A classifier

- There are several ways to describe a classifier:
  - if $p(\omega = h \mid x) > p(\omega = d \mid x)$ then assign to h
    otherwise to d
  - if $p(\omega = h \mid x) - p(\omega = d \mid x) \geq 0$ then assign to h
    otherwise to d
  - if $\dfrac{p(\omega = h \mid x)}{p(\omega = d \mid x)} \geq 1$ then assign to h
    otherwise to d
  - if $\ln\left[p(\omega = h \mid x)\right] - \ln\left[p(\omega = d \mid x)\right] \geq 0$ then assign to h
    otherwise to d

- A Bayesian classifier is a *threshold* on the difference between *posterior probabilities*

Bayes’ rule

- In many cases, the posterior is hard to estimate
- Often a certain functional form can be assumed for the *class-conditional distributions*
- Use Bayes' theorem to rewrite one into the other:
  - posterior distribution: $$p(\omega = c \mid x) = \frac{p(x \mid \omega = c)\, p(\omega = c)}{p(x)}$$
  - class-conditional distribution: $p(x \mid \omega = c)$
  - prior distribution: $p(\omega)$
  - data distribution: $$p(x) = \sum_{c=1}^{C} p(x \mid \omega = c)\, p(\omega = c)$$

Bayes' rule (2)

- The decision rule becomes

$$p(\omega = h \mid x) > p(\omega = d \mid x)$$

$$\Downarrow$$

$$\frac{p(x \mid \omega = h)\, p(\omega = h)}{p(x)} > \frac{p(x \mid \omega = d)\, p(\omega = d)}{p(x)}$$

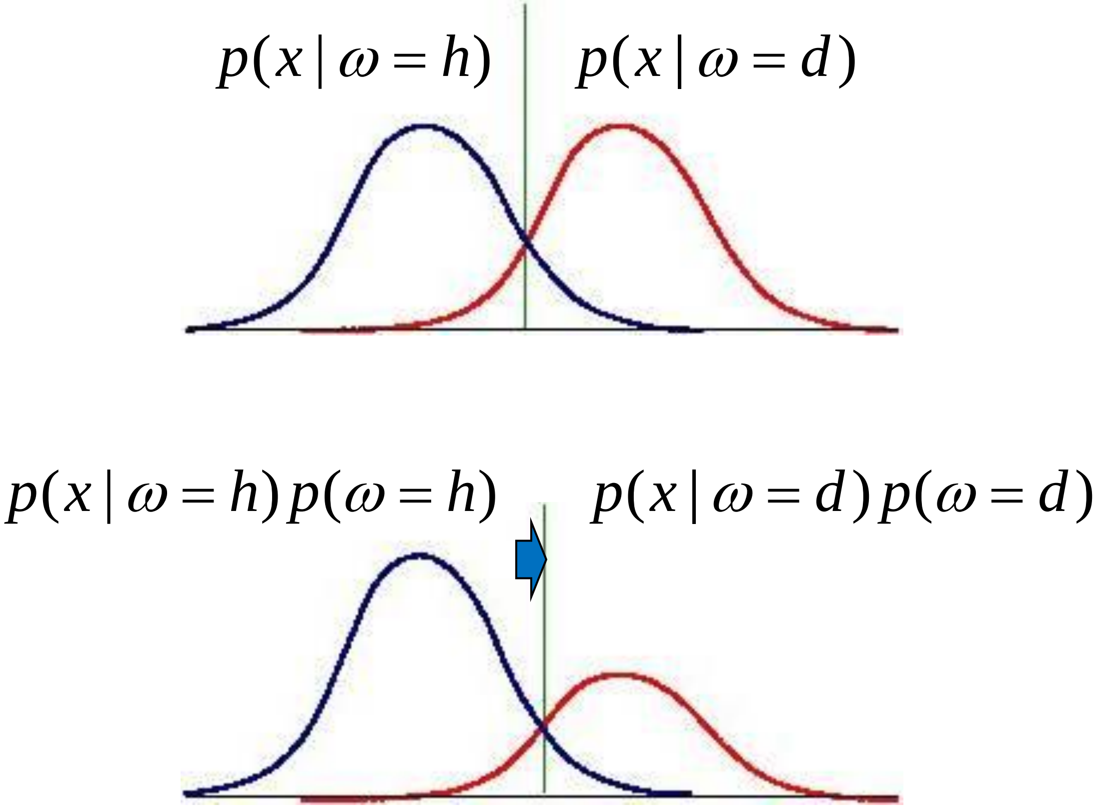
$$\Downarrow$$

$$p(x \mid \omega = h)\, p(\omega = h) > p(x \mid \omega = d)\, p(\omega = d)$$

Seems trivial, but this is something we can measure!

Bayes’ rule (3)

- The effect of the prior:

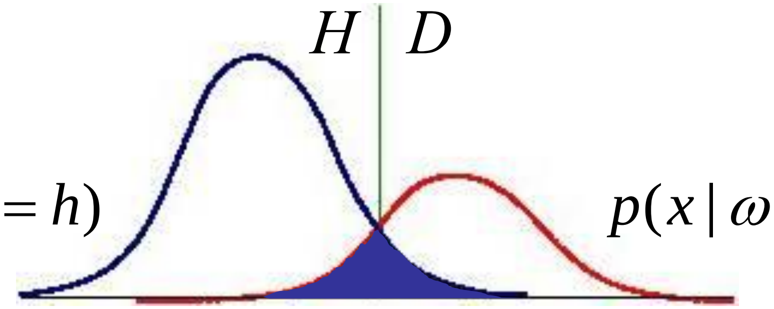


Prior can shift the decision boundary
If one class is very unlikely, we will not make a large error if we misclassify that class

Bayes' rule (4)

- Bayes’ error: ***minimal attainable error***
  (if data follows class-conditional contributions…)

$p(x \mid \omega = h)\, p(\omega = h)$

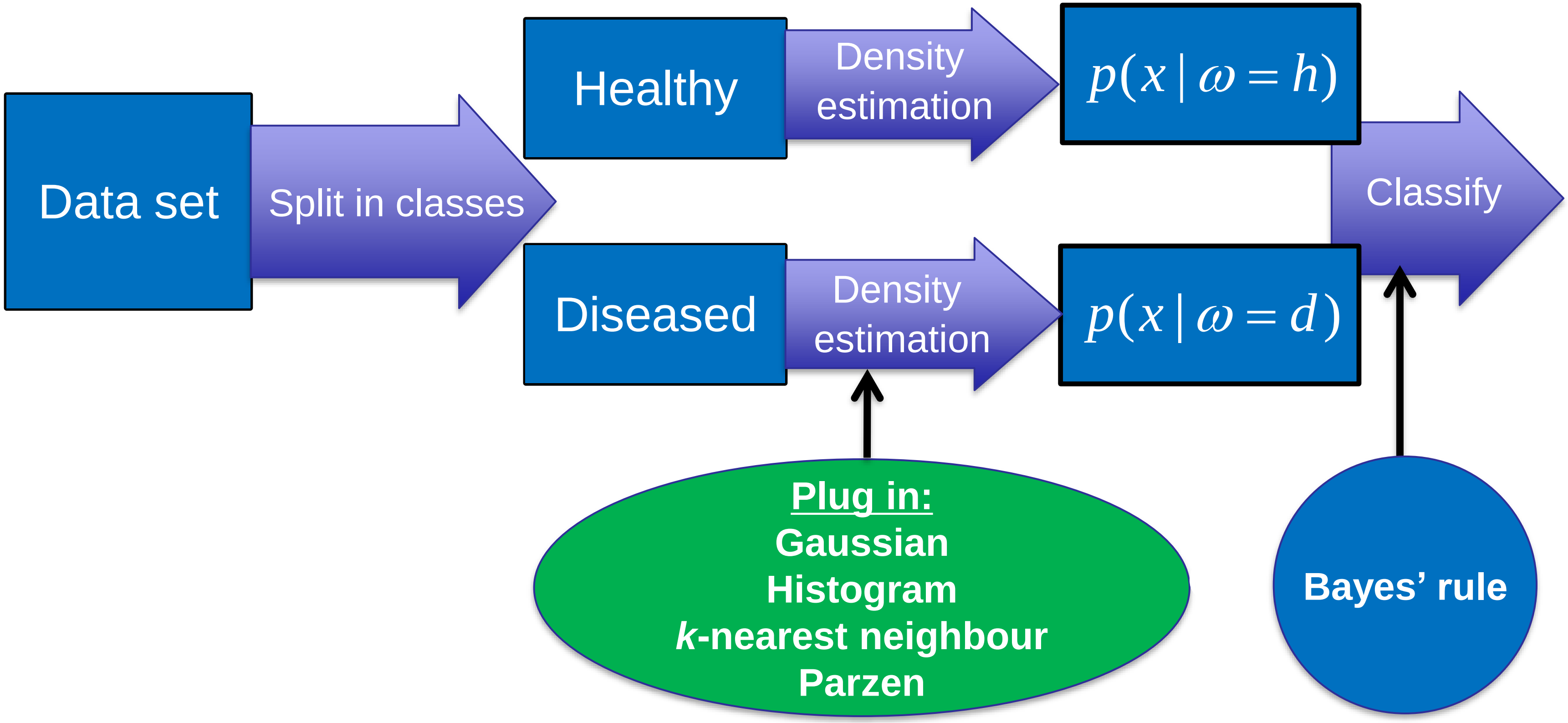


$p(x \mid \omega = d)\, p(\omega = d)$

- $\Lambda(\omega',\omega) = 0$ when $\omega' = \omega$
- $\Lambda(\omega',\omega) = 1$ otherwise

Bayes' rule (5)

- In practice:



Data set

Split in classes

Healthy

Diseased

Density estimation

Density estimation

$p(x \mid \omega = h)$

$p(x \mid \omega = d)$

Classify

Plug in:
Gaussian
Histogram
***k*-nearest neighbour**
Parzen

Bayes' rule

Plug-in Bayes classifier

- Bayes' rule:

$$c_{opt} = \arg\max_c \, p(\omega = c \mid x) = \arg\max_c \, p(x \mid \omega = c)\, p(\omega = c)$$

- Given priors, we only require the class conditional distributions $p(x|\omega=c)$

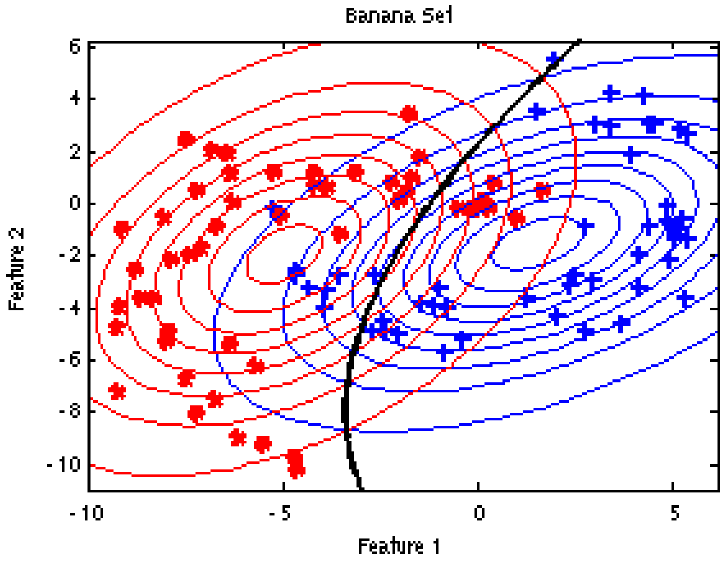
- In practice we will always have to *estimate* $p(x|\omega=c)$ by $\hat{p}(x \mid \omega = c)$ and hope that the resultinh classifier when we *plug in* this approximation will still perform well

- Density estimation is a very hard problem!
- The resulting classifier will be *sub-optimal* and in general will *not* attain Bayes' error

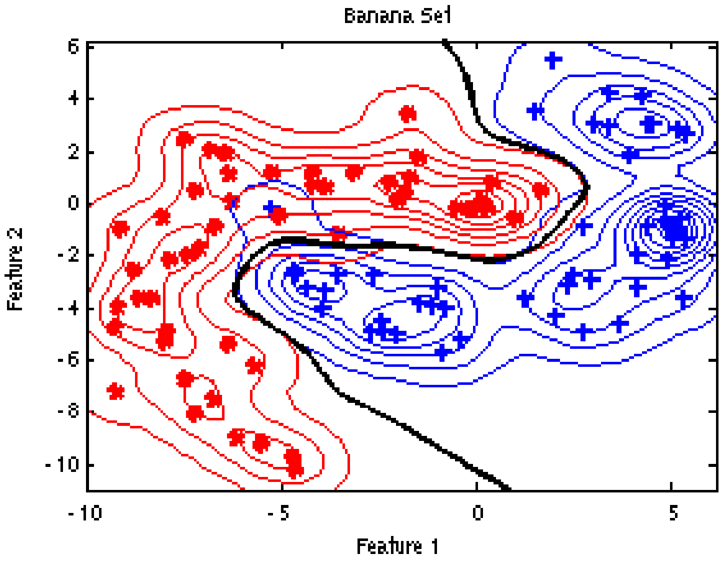
Plug-in Bayes classifier (2)

- Same problem, two different density estimates $\hat{p}(x \mid \omega = c)$

Normal density estimation



Parzen density estimation



Which one is best (Parzen)
Which one is optimal (none: true dist = normal perpendicular to two half-circles)

Density estimation

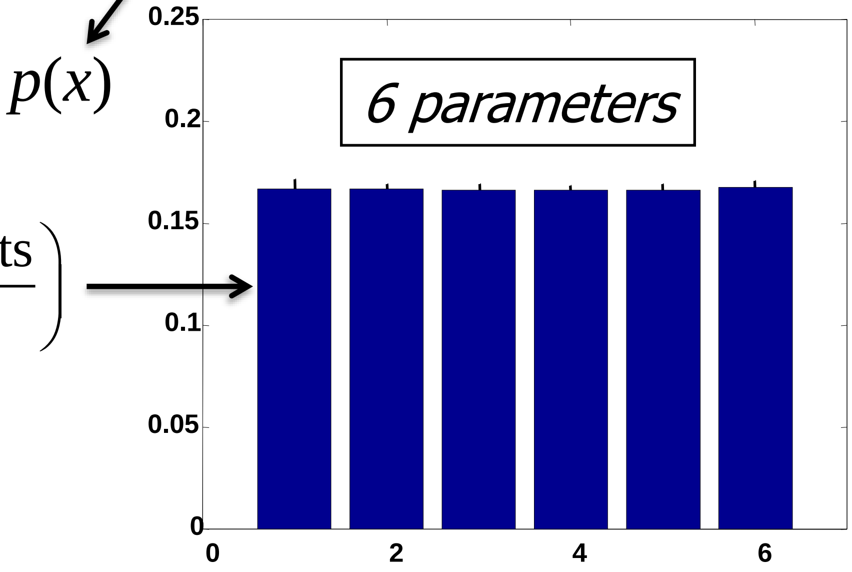
Density estimation

- Simplest approach: approximate density by histogram

  e.g. 10,000 throws
  of a dice

10,000 objects

1 measurement



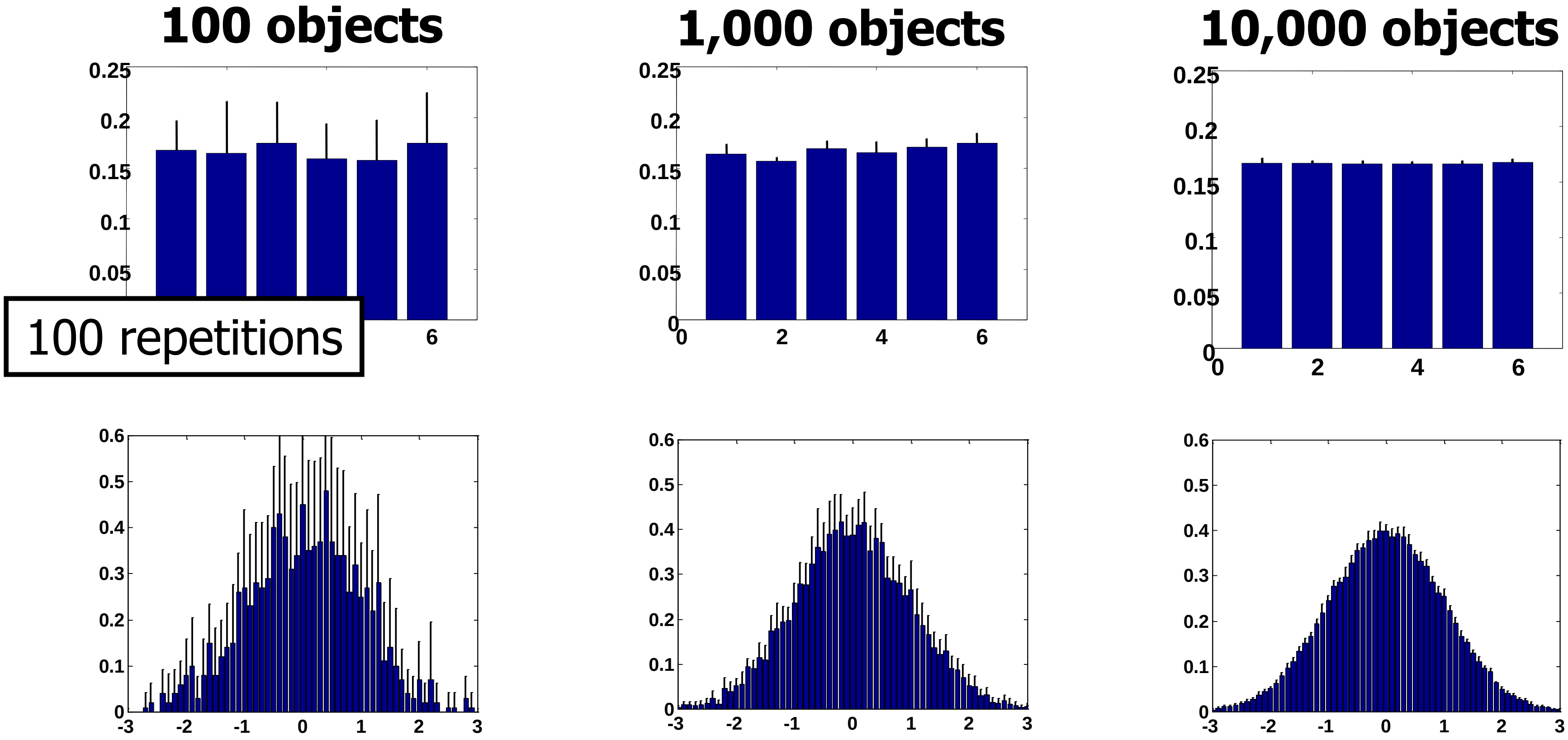
6 parameters

$$\hat{p}(\mathbf{x}) = \frac{dP(\mathbf{x})}{d\mathbf{x}} = \left(\frac{\text{fraction of objects}}{\text{volume}}\right)$$

- But...

Density estimation (2)

- Problem: accuracy



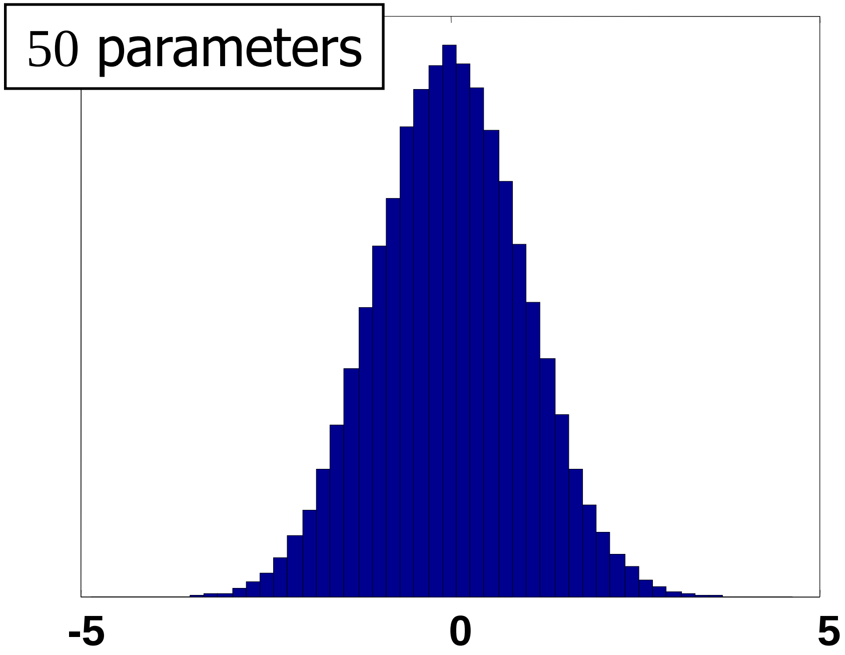
100 repetitions

Gauss: 50 bin -> 50 parameters to estimate

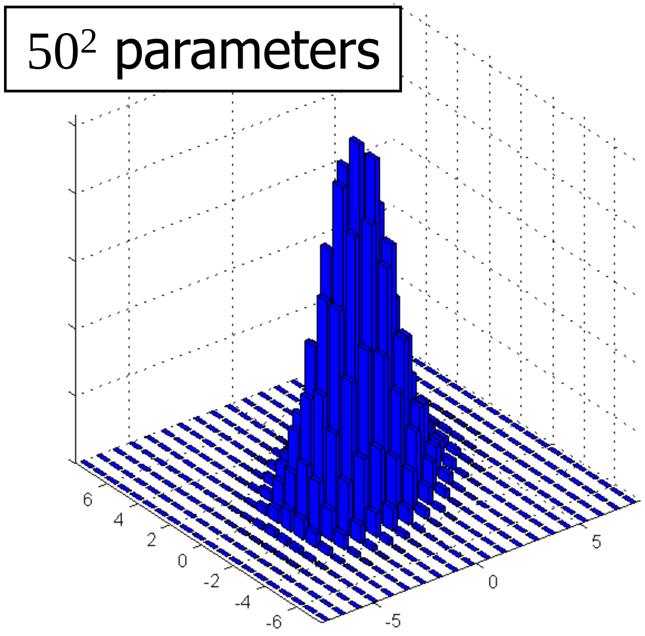
Density estimation (3)

- For 1 - dimensional data, $\pm$ 1000 points needed

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



50 parameters



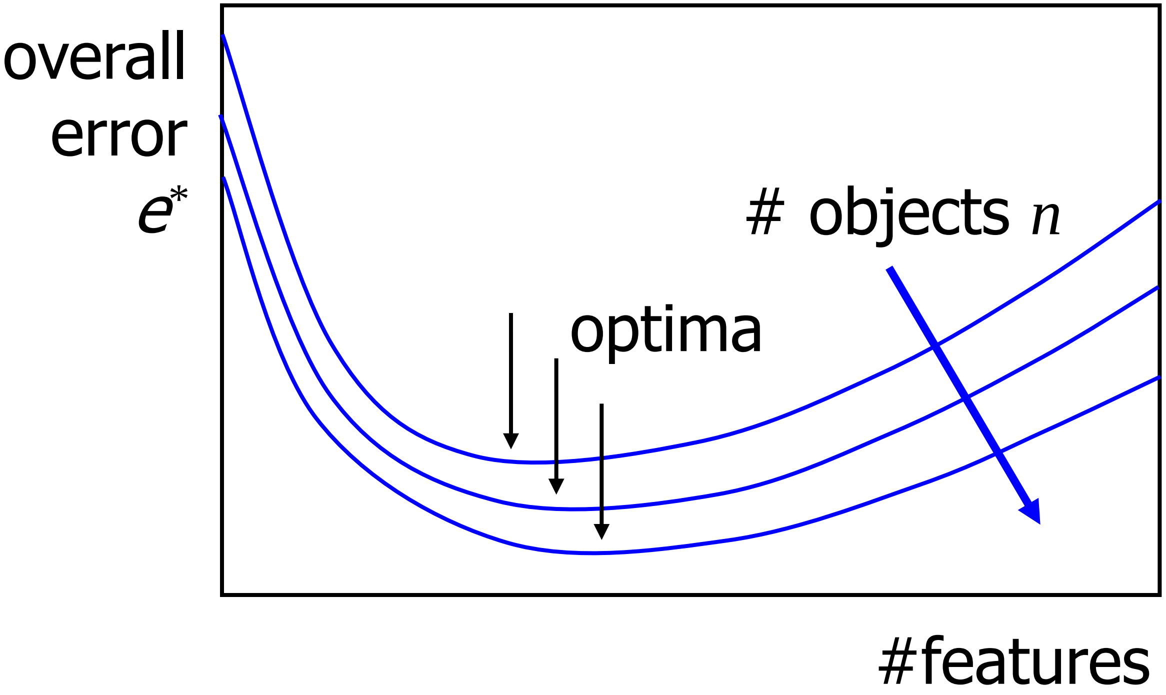
50^2 parameters

- Unworkable for $p > 2$ measurements

Curse of dimensionality

- Intuitively, using more features
  (e.g. width, height, color etc.) should give us
  more information about the outcome to predict
- But we never know the densities, so we have to estimate them
- The 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

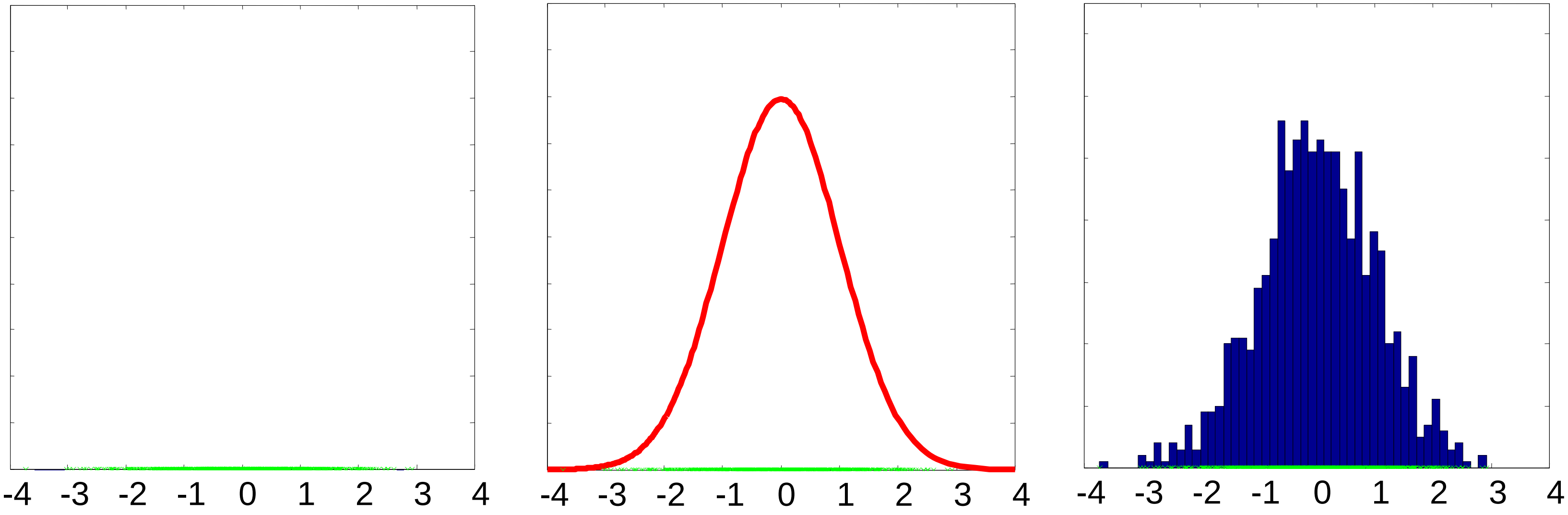
Curse of dimensionality (2)



So, realize if n -> INF than you can have many features

Density estimation (4)

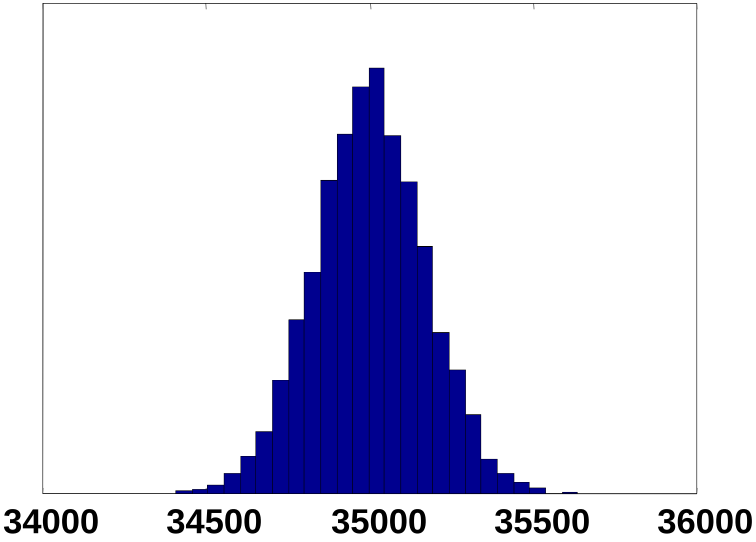
- Two main approaches:
  - *parametric*: assume simple *global* model, e.g. Gaussian, and estimate its parameters
  - *non-parametric:* assume simple *local* model, e.g. uniform, Gaussian, and aggregate



The Gaussian distribution

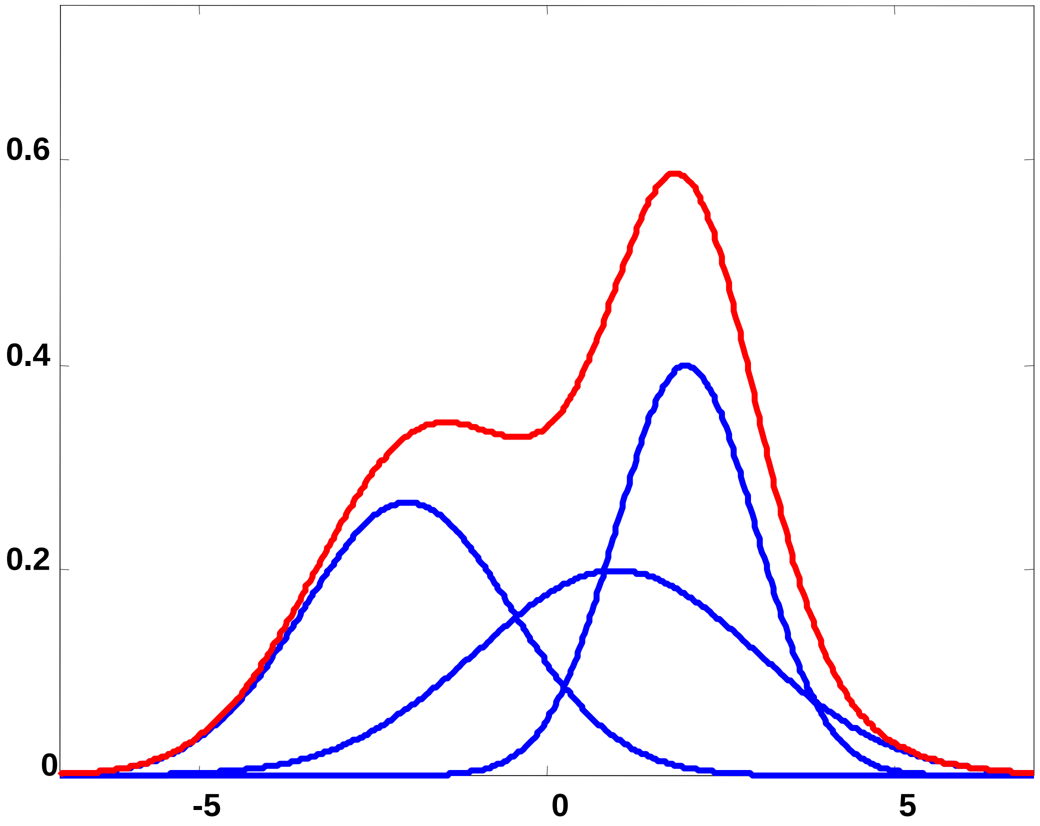
- Why Gaussians?
  - Special distribution: the Central Limit Theorem says that sums of large numbers of i.i.d. (independent, identically distributed) random variables will have a Gaussian distribution
  - Simple, few parameters
  - Often occurs in real life

    e.g. sum of eyes of
    10,000 dice throws
    (expectation = 3.5 per throw)



The Gaussian distribution (2)

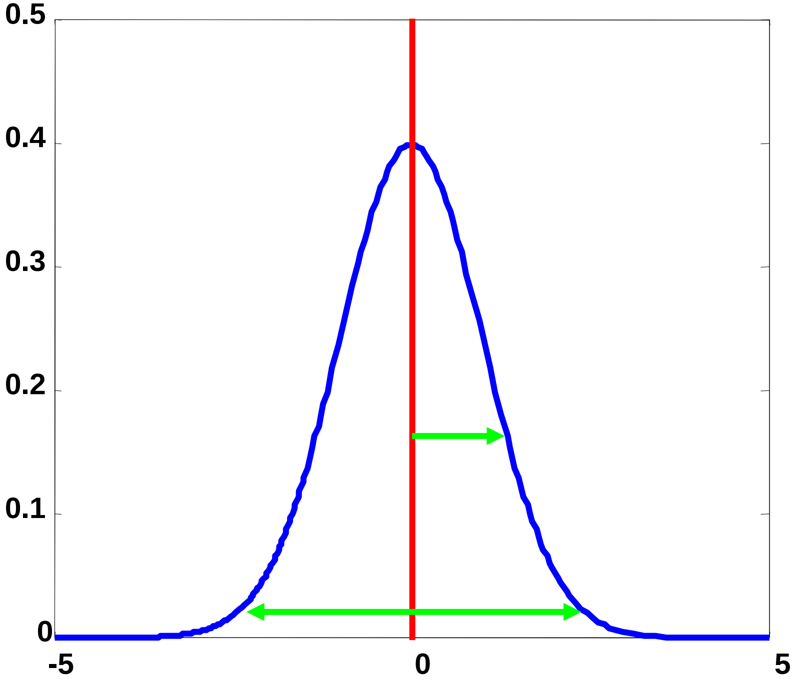
- Not necessarily too restrictive: mixture models (discussed tomorrow)



— Gaussian

— Mixture of Gaussians

The Gaussian distribution (3)



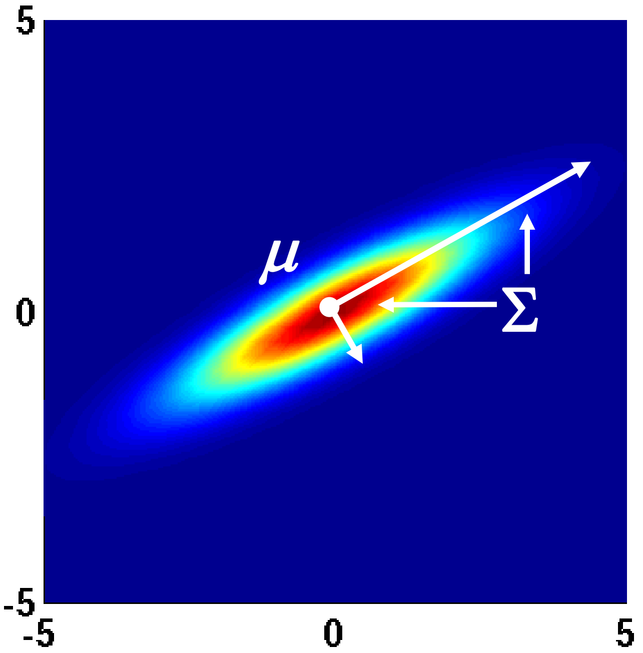
- Normal distribution = Gaussian distribution
- Standard normal distribution: $\mu = 0,\ \sigma^2 = 1$
- 95.45% of data between $[\ \mu - 2\sigma, \mu + 2\sigma\ ]$ (in 1D!)

- 1-dimensional density:

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

μ : mean
σ^2 : variance

Multivariate Gaussian distribution



$$\Sigma = \begin{bmatrix} 3 & 1½ \\ 1½ & 2 \end{bmatrix}$$

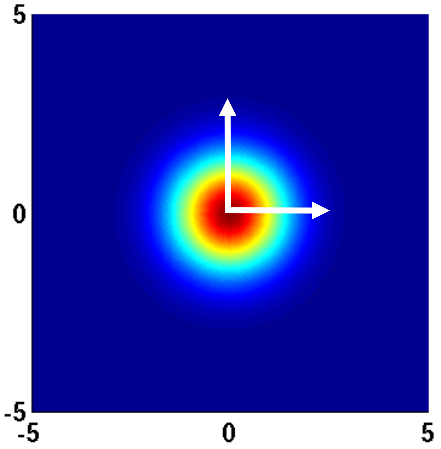
- p - dimensional density:

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

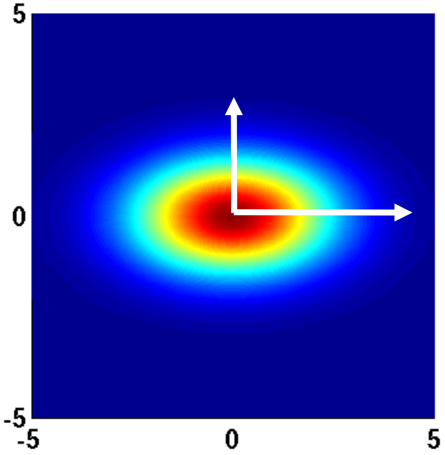
$\boldsymbol{\mu}$: mean

$\boldsymbol{\Sigma}$: covariance matrix

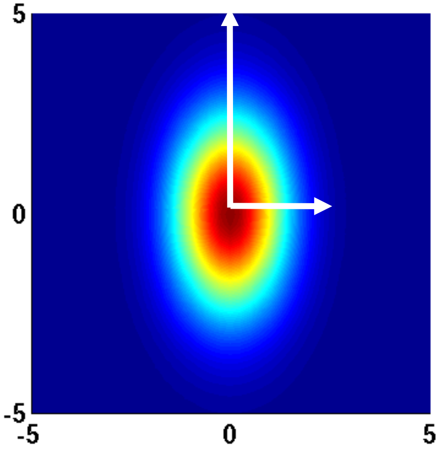
Multivariate Gaussian distribution (2)



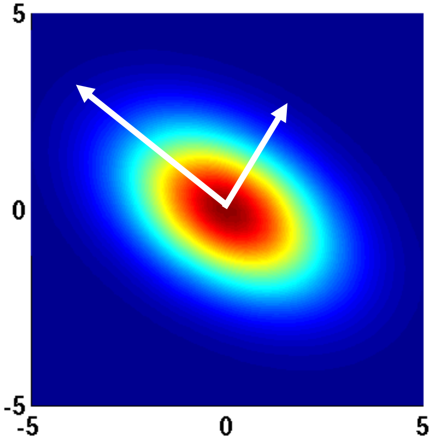
$$\Sigma = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$



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



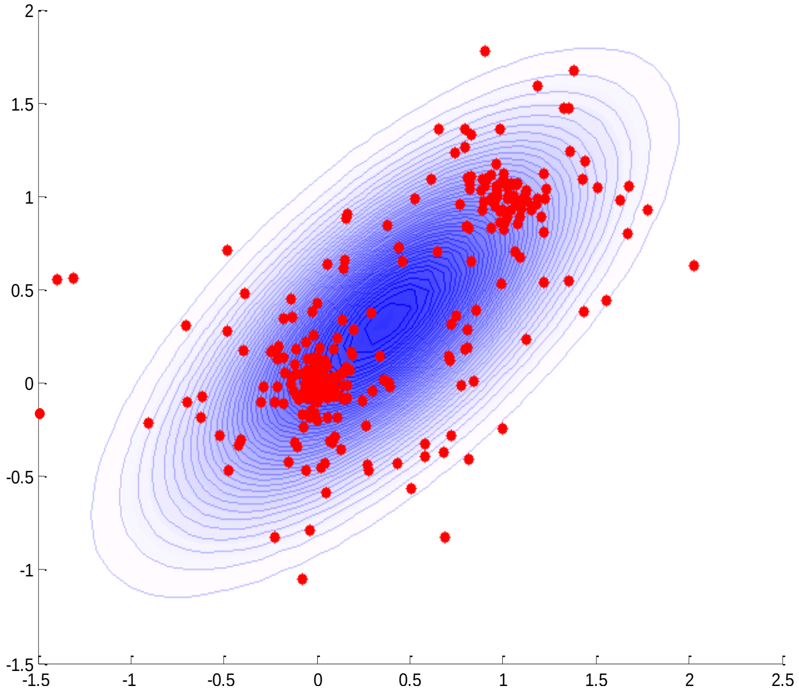
$$\Sigma = \begin{bmatrix} 1 & 0 \\ 0 & 3 \end{bmatrix}$$



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

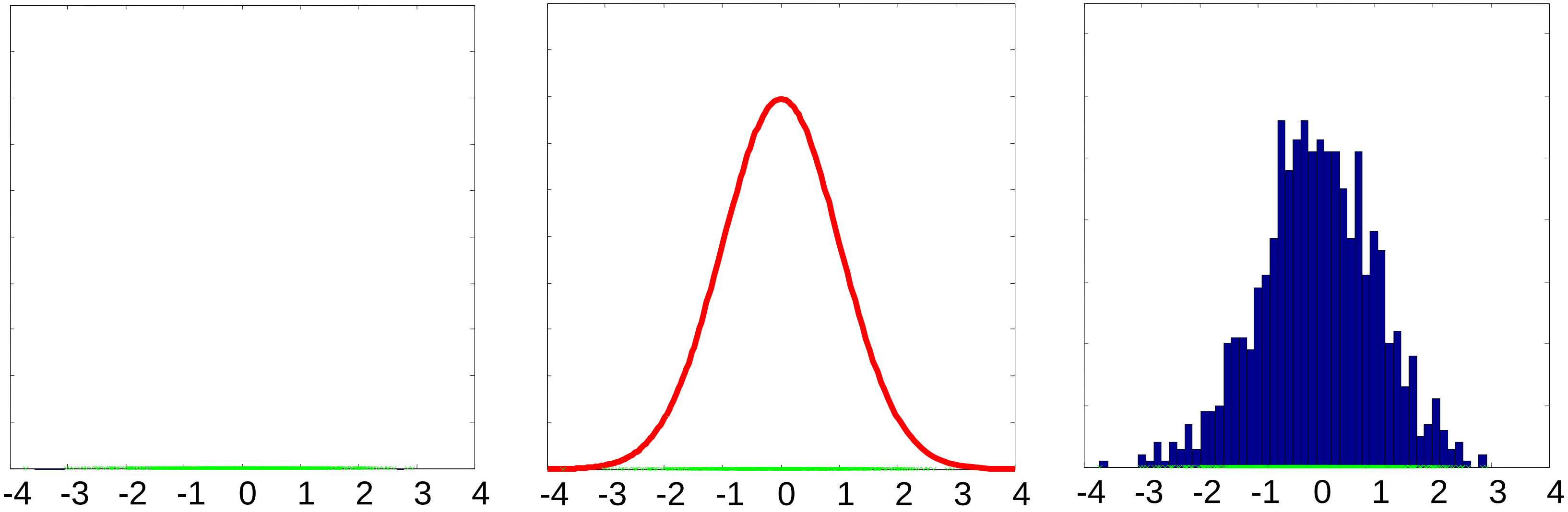
Parametric estimation

- Assume model, e.g. Gaussian and estimate mean μ and covariance Σ from data
- 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 : need *a lot* of data for high-dimensional problems



Density estimation (4)

- Two main approaches:
  - *parametric*: assume simple *global* model, e.g. Gaussian, and estimate its parameters
  - *non-parametric:* assume simple *local* model, e.g. uniform, Gaussian, and aggregate

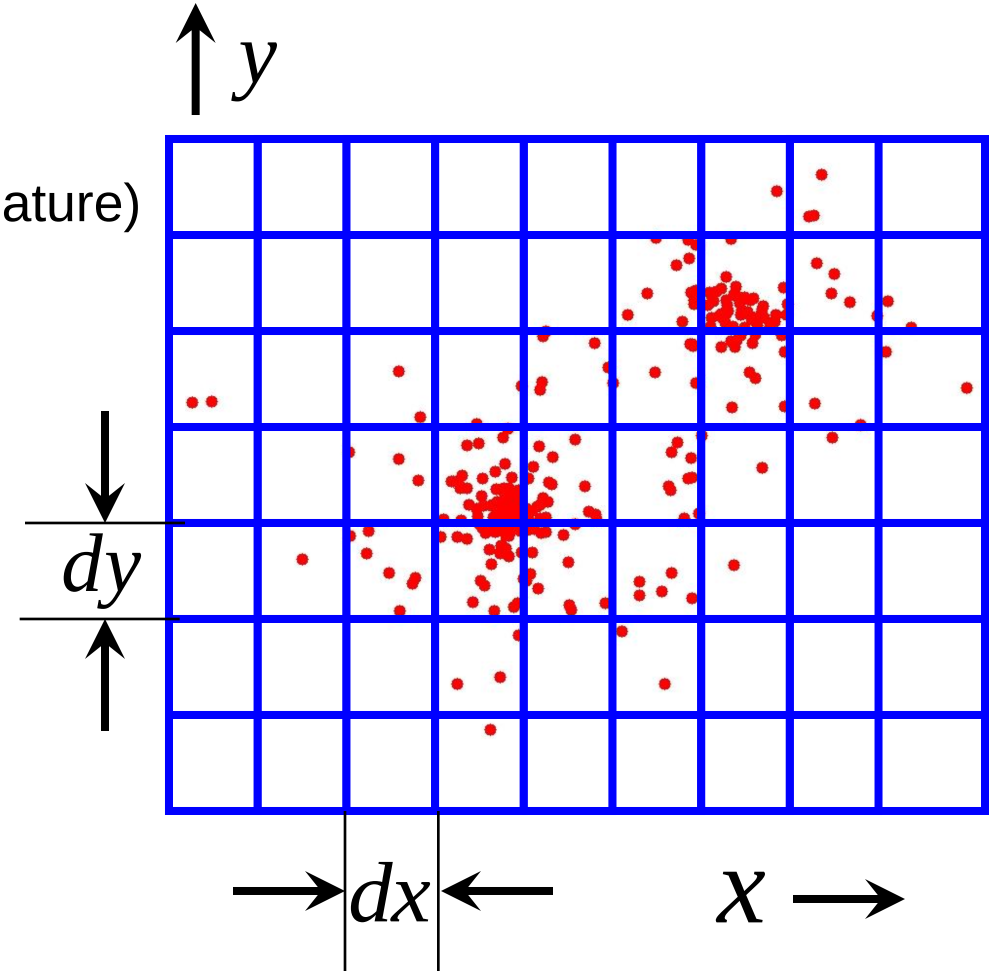


Exercise 1.10-1.14

Histogramming

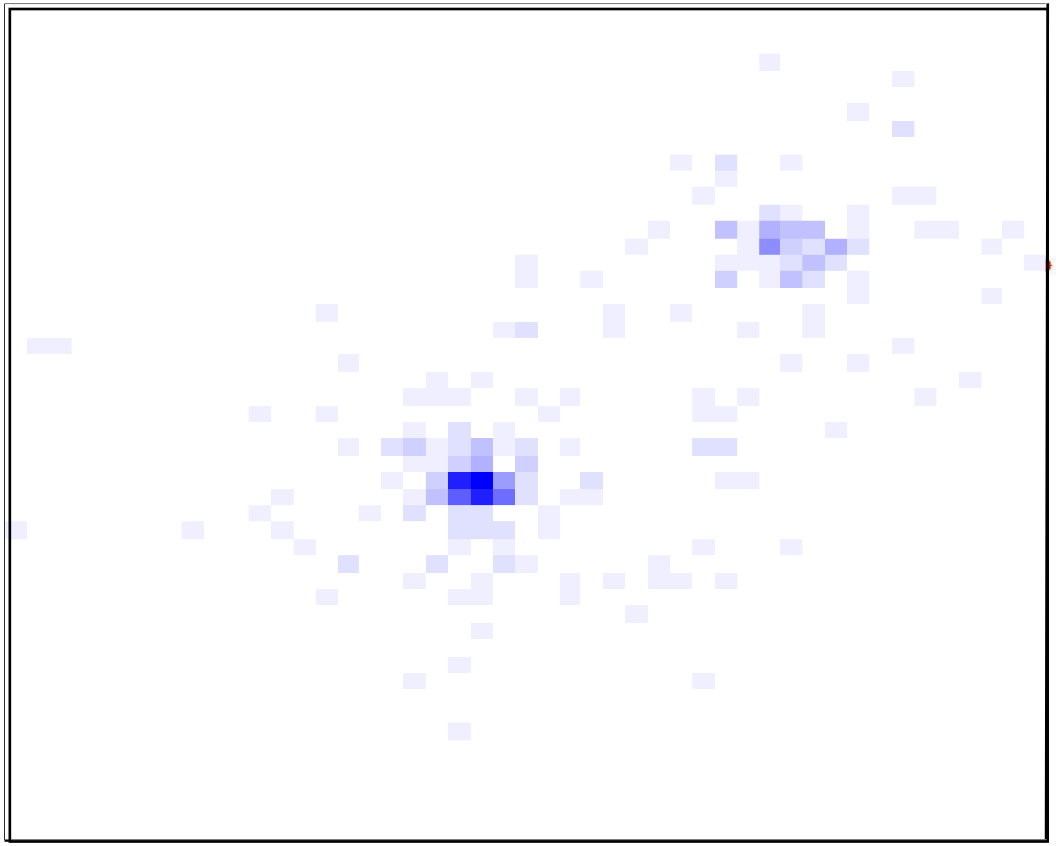
- Histogram method:
  - Divide feature space into N^p bins (N bins per feature)
  - Count number of objects in each bin
  - Normalize:

$$\hat{p}(\boldsymbol{x}) = \frac{n_i}{\sum_{i=1}^{N^p} n_i dxdy}$$



Histogramming (2)

- For example, using N=50 bins per dimension

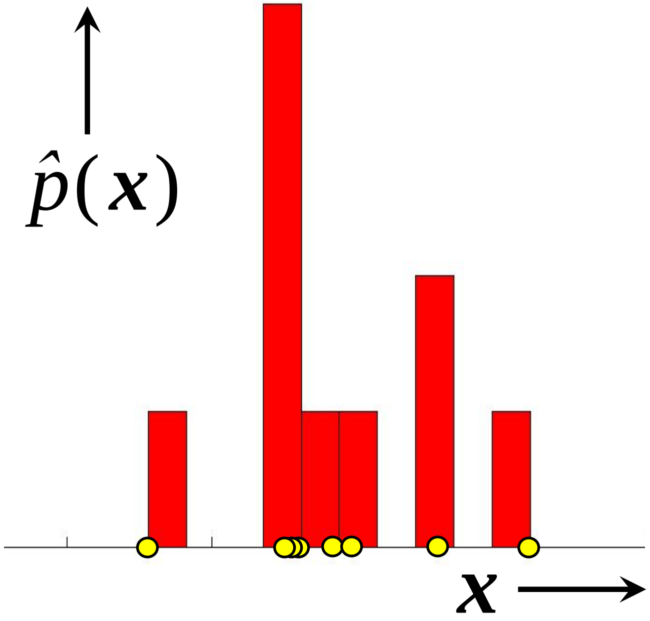


Histogramming (3)

- Histogram density estimate:

$$\hat{p}(\mathbf{x} \mid dx) = \left(\frac{\text{fraction of objects}}{\text{volume}} \right)$$

  - Fix cell size (dx)
  - Count #objects per cell

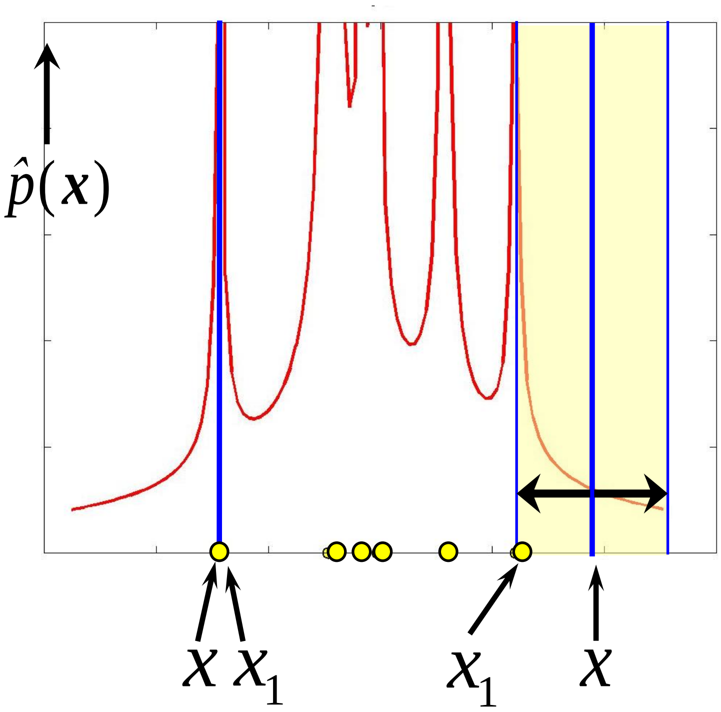


k-nearest neighbor density estimation

- *k*-nearest neighbor estimate:

$$\hat{p}(\mathbf{x}\,|\,k) = \left(\frac{\text{fraction of objects}}{\text{volume}}\right)$$

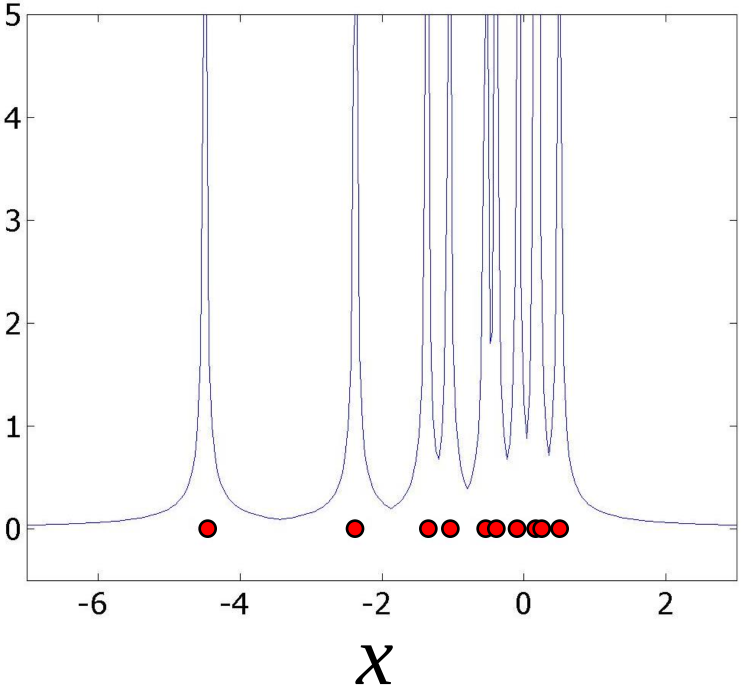
$$= \frac{k}{n\Delta x_k} = \frac{k}{n\|x - x_k\|}$$



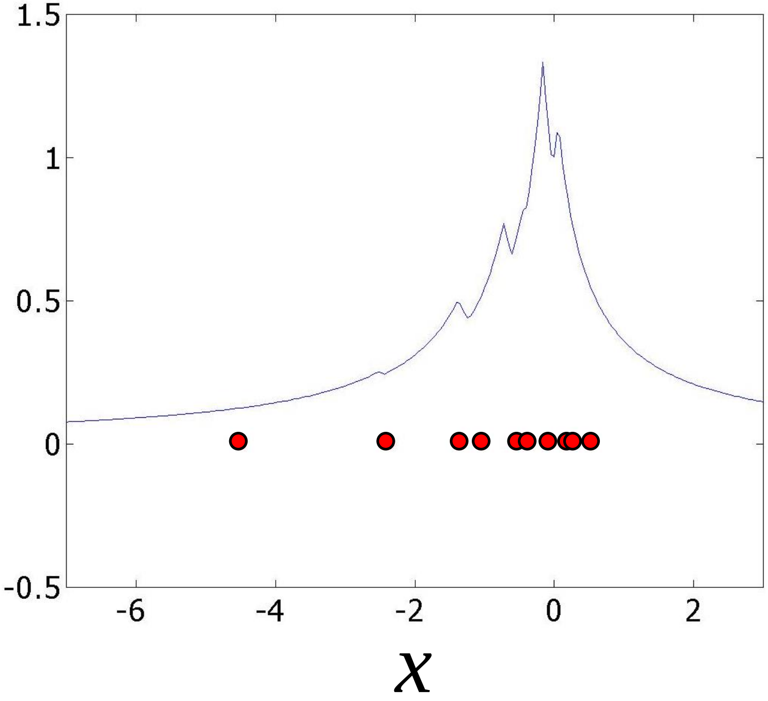
  - Fix #objects per cell (k)
  - Determine cell size (volume)

k-nearest neighbor density estimation (2)

- The density estimate for $k = 1$ contains singularities:



$n = 10, k = 1$



$n = 10, k = 5$

Also known as kernel-based density estimation

Parzen density estimation

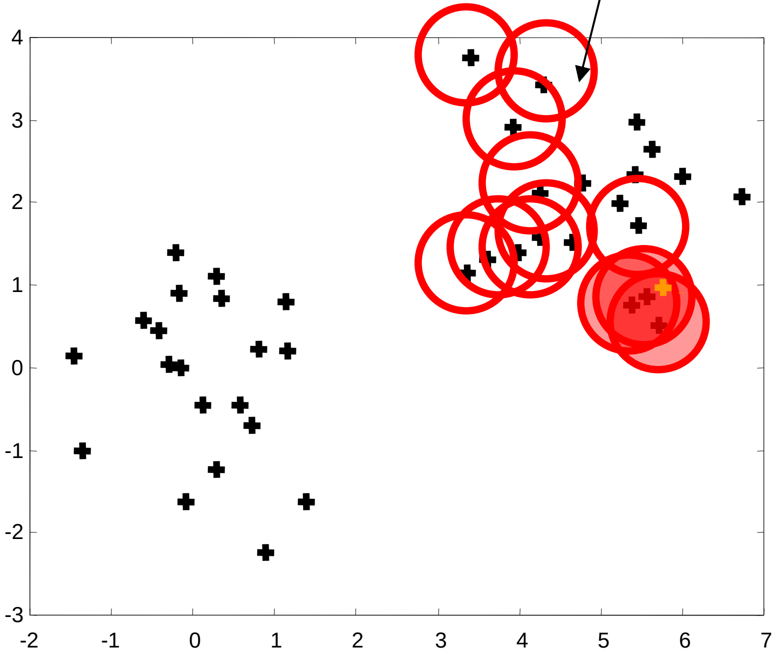
- Procedure:
  - Fix volume of cell
  - Vary positions of cells
  - Add contributions of cells
- Define cell shape (kernel), e.g. uniform

$$K(r,h)=\begin{cases}0 & \text{if } |r|>h\\ 1/V & \text{if } |r|\le h\end{cases}$$

(with V the volume of the kernel)

or Gaussian

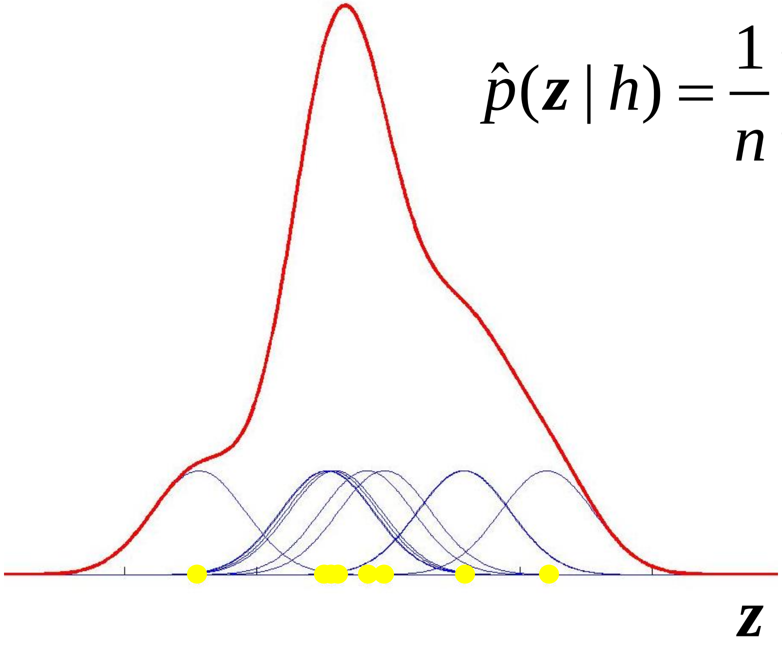
- For test object $\mathbf{z}$, sum all cells: $\hat{p}(\mathbf{z}\,|\,h)=\dfrac{1}{n}\sum_{i=1}^{n}K\left(\|\mathbf{z}-\mathbf{x}_i\|,h\right)$



Parzen density estimation (2)

- With Gaussian kernel: $K(r,h)=\frac{1}{2\pi^{1/2}h}\exp\left(-\frac{1}{2}\frac{r^2}{h^2}\right)$

$$\hat{p}(\mathbf{z}\,|\,h)=\frac{1}{n}\sum_{i=1}^{n}K\left(\left\|\mathbf{z}-\mathbf{x}_i\right\|,h\right)$$

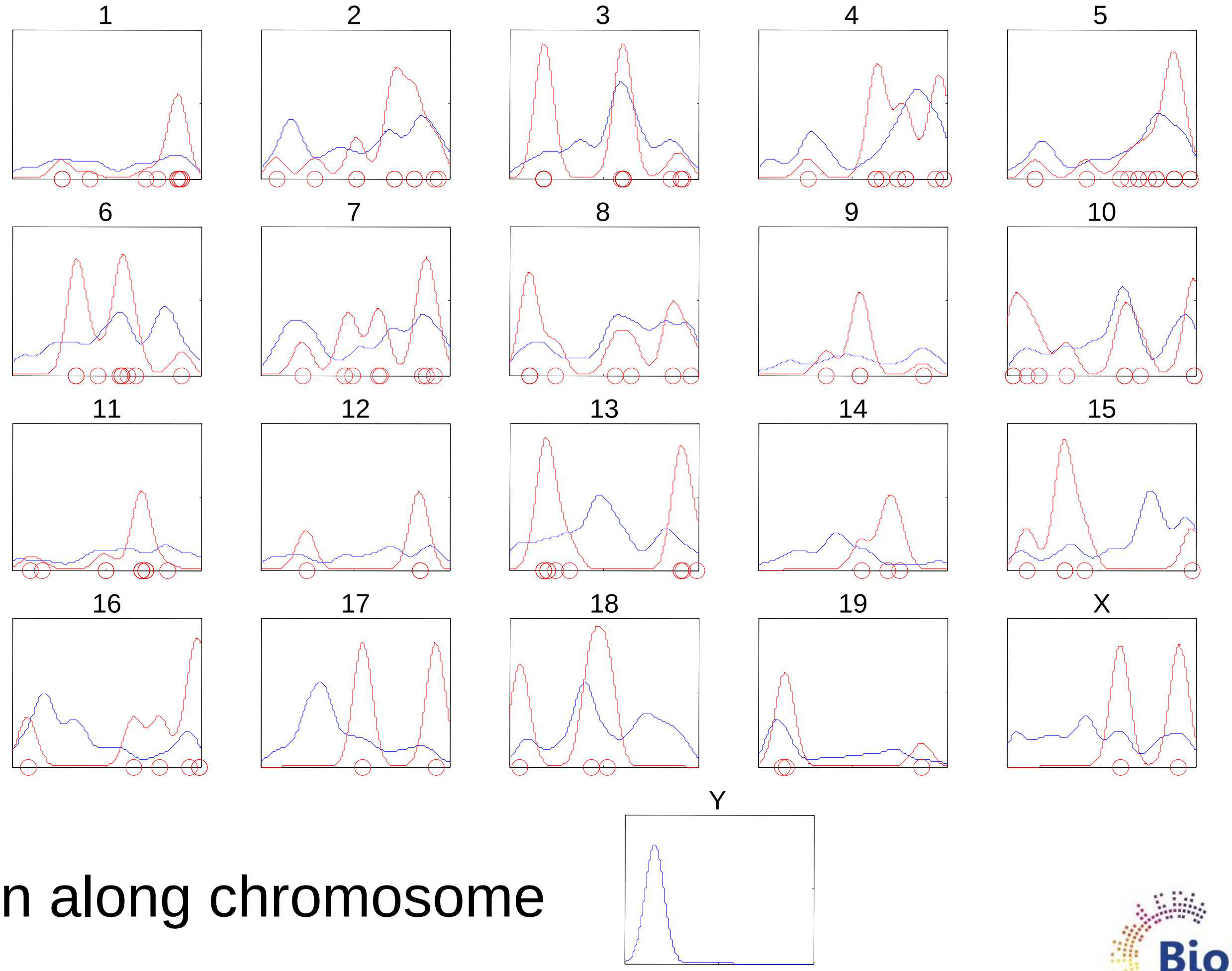


Parzen density estimation (3)

- Example: viral insertions in each chromosome

🟦 Density of highly expressed genes

🟥 Density of viral insertions



- Feature: position along chromosome

Parzen density estimation (4)

- Maximum likelihood (ML) estimate: choose kernel width h such that the probability of the observed data is maximal
  - PDF of observing a point $\boldsymbol{z}$:

$$\hat{p}(\boldsymbol{z} \mid h) = \frac{1}{n} \sum_{i=1}^{n} K\left(\left\| \boldsymbol{z} - \boldsymbol{x}_i \right\|, h \right)$$

  - PDF of observing dataset $\boldsymbol{x}_1$, ..., $\boldsymbol{x}_n$ (likelihood):

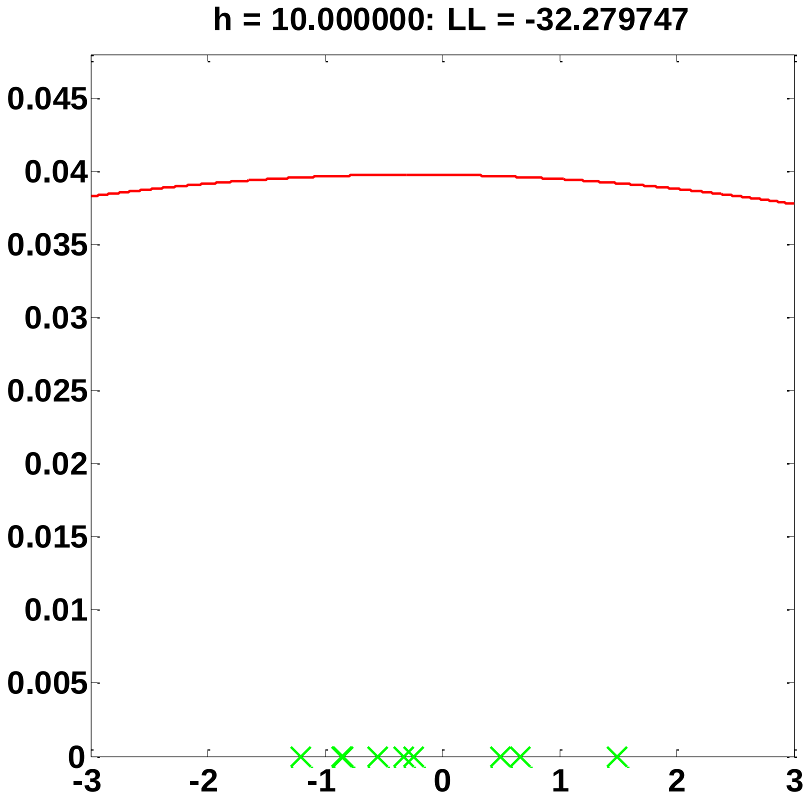
$$\hat{p}(X|h) = \prod_{i-1}^{n} \hat{p}(x_i|h)$$

    (this assumes independence!)
  - **Maximize log-likelihood** w.r.t. h *(convenient to avoid multiplication)*:

$$LL = \log(g(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n)) = \sum_{i=1}^{n} \log(\hat{p}(\boldsymbol{x}_i \mid h))$$

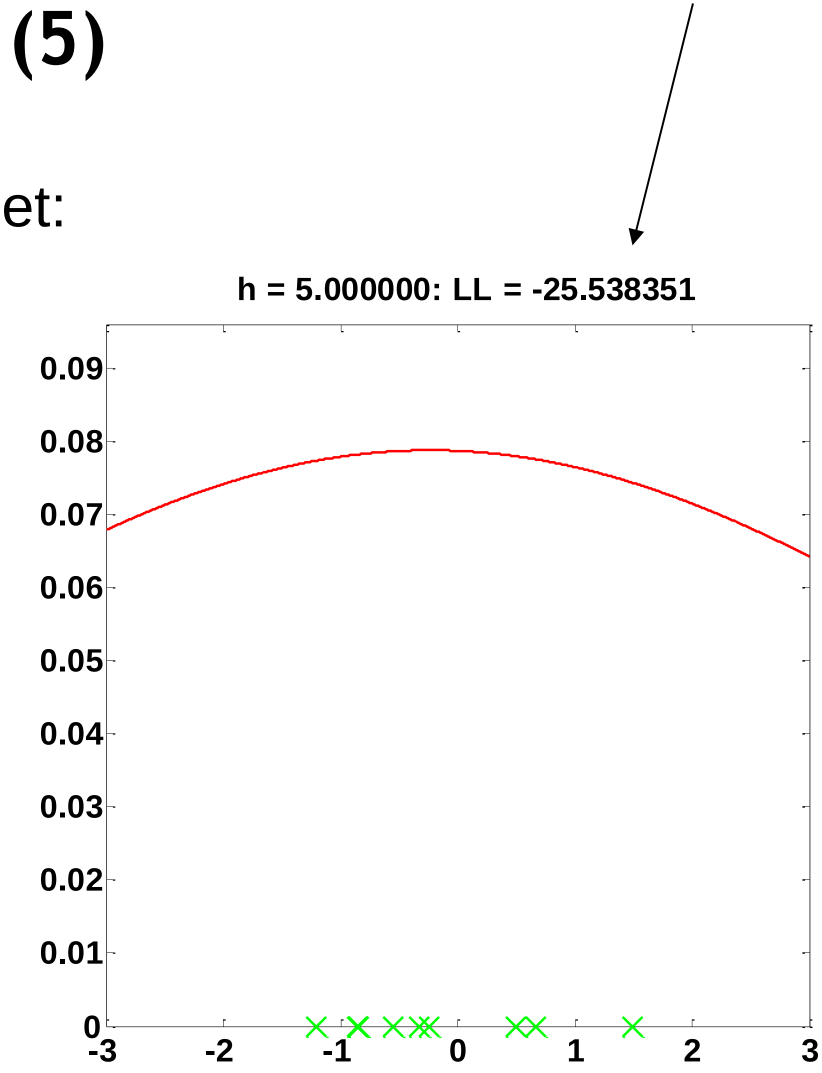
Parzen density estimation (5)

- Maximum likelihood on training set:



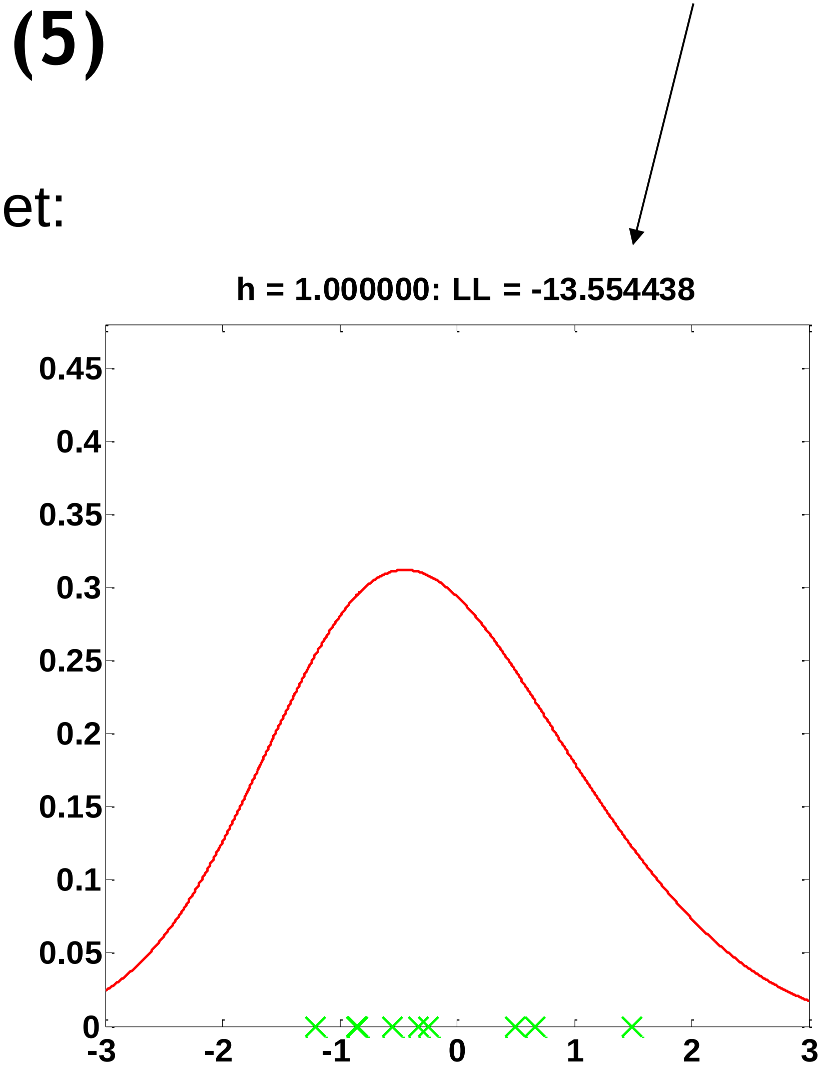
Parzen density estimation (5)

- Maximum likelihood on training set:



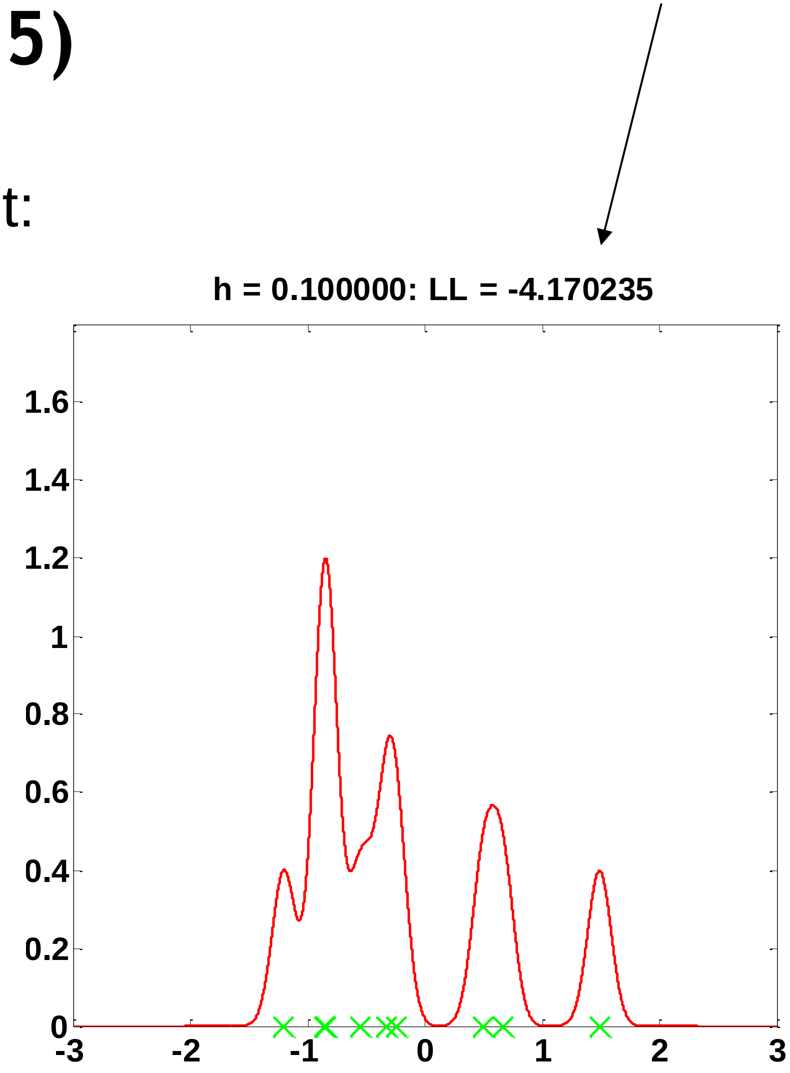
Parzen density estimation (5)

- Maximum likelihood on training set:



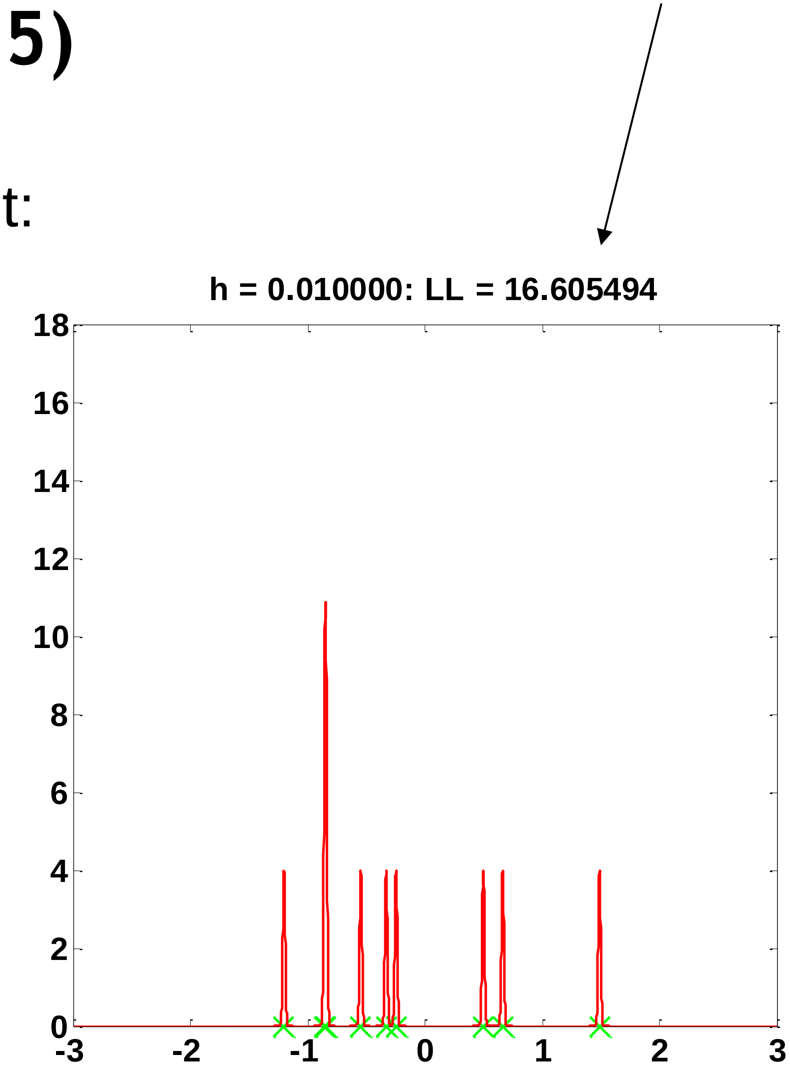
Parzen density estimation (5)

- Maximum likelihood on training set:



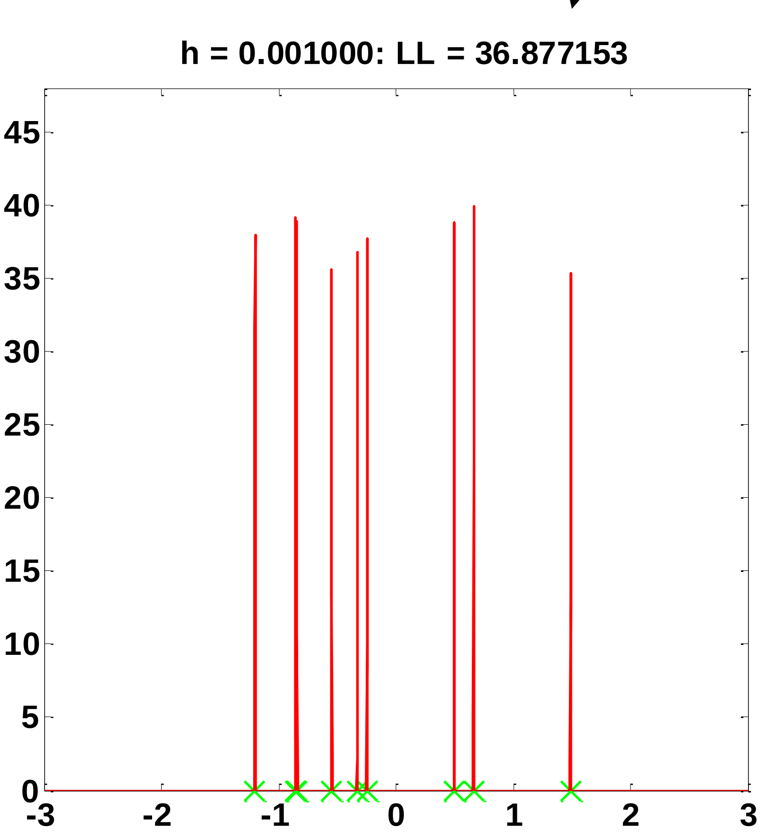
Parzen density estimation (5)

- Maximum likelihood on training set:


h = 0.010000: LL = 16.605494

Parzen density estimation (5)

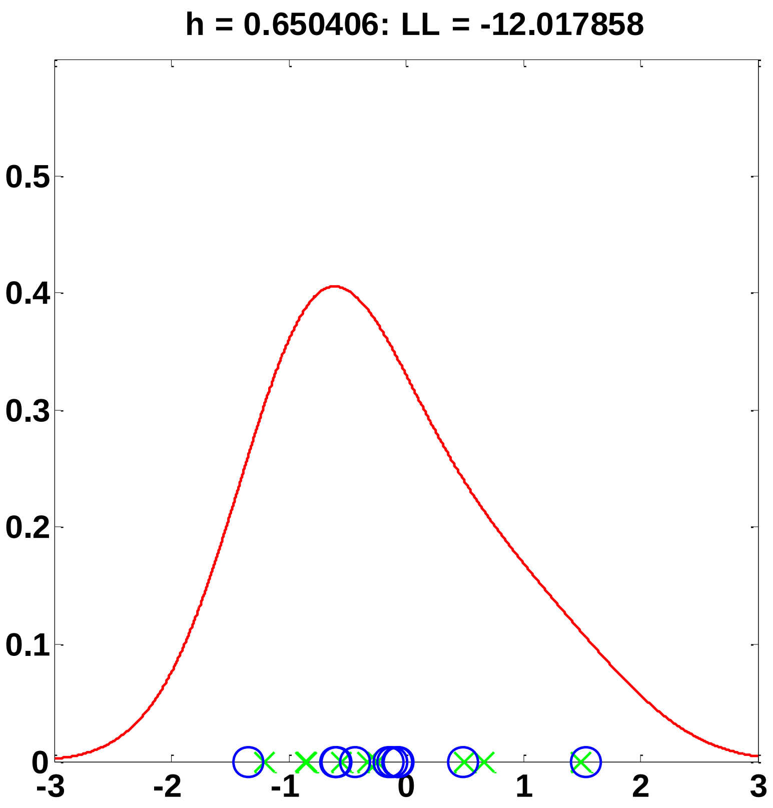
- Maximum likelihood on training set:
  - $h \rightarrow 0$: $LL \rightarrow \infty$
  - **Extreme example of *overtraining* : fitting data too much**



Cross-validation

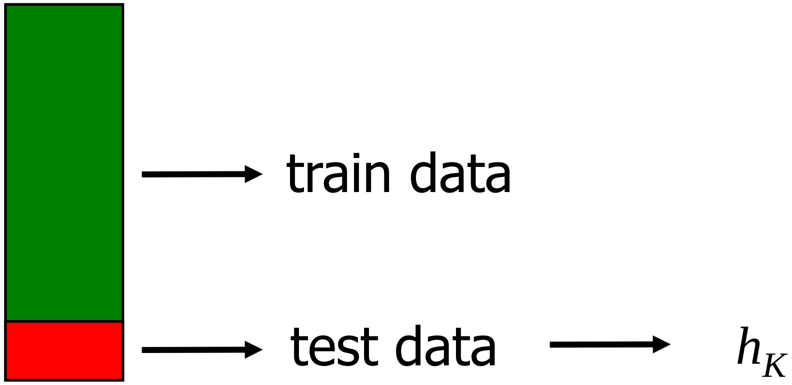
- Solution:
  - Split data into *training set* and *validation set*
  - Optimise h w.r.t. likelihood of validation set, given Parzen model trained on training set

- Problems:
  - Uses a lot of valuable data
  - Sensitive to split of data



Cross-validation (2)

- Better solution: *K*-fold crossvalidation
  - Split data into K parts ($K = n$: leave-one-out)
  - Repeat K times:
    - Find h using $(K - 1)$ parts for training and 1 part for validating
  - Use average of h's as kernel width



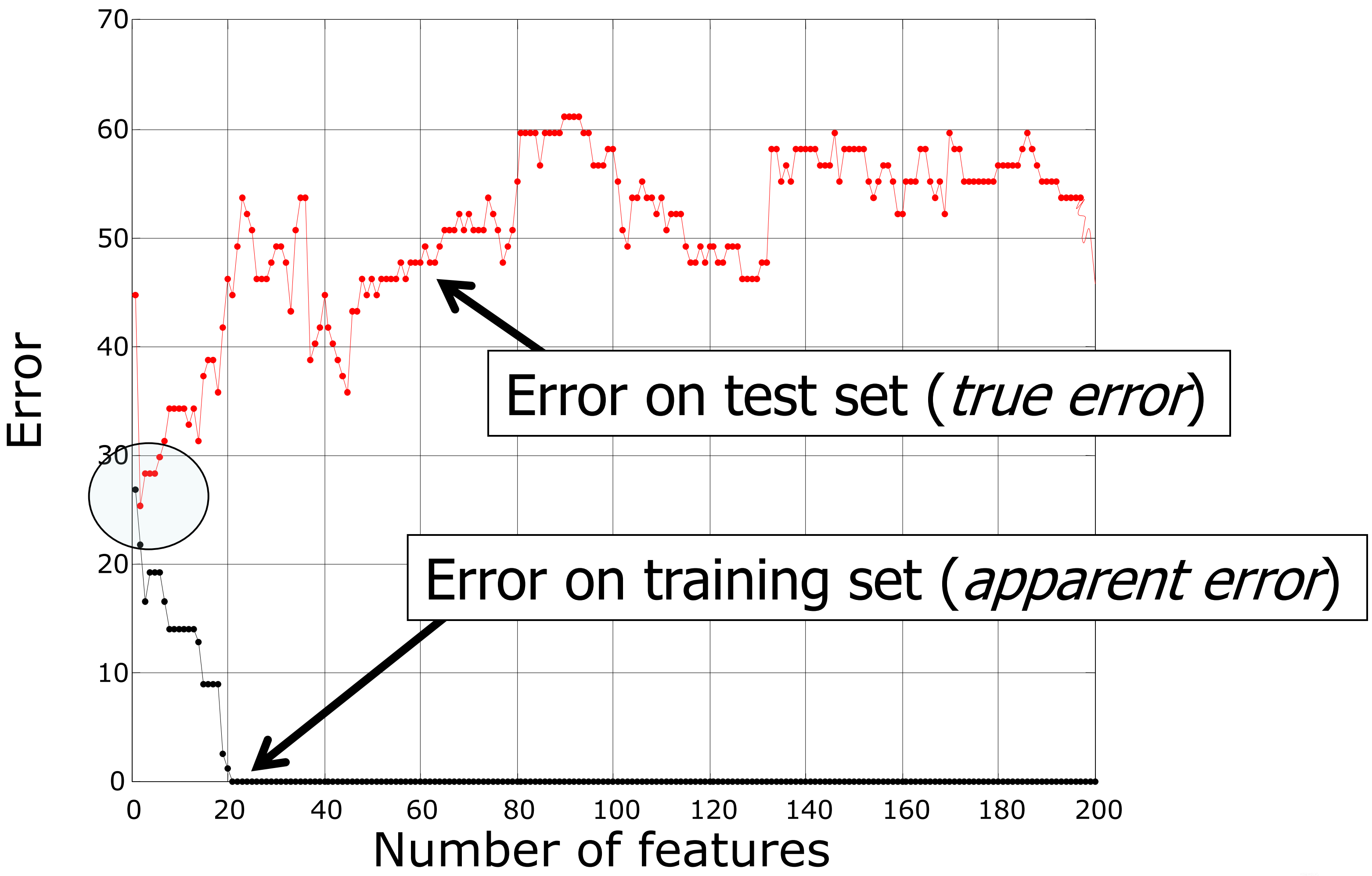
(will return)

Training, test and validation sets

- Terminology:
  - A *training set* is used to estimate parameters
  - An optional *validation set* is used to optimize parameter settings, e.g. by calculating classifier error on this set
  - **A *test set* is only used to judge performance of the entire classifier (only used once!)**

- Error estimates:
  - On training set: *apparent error*
  - On test set: *true error*

Training, test and validation sets (2)

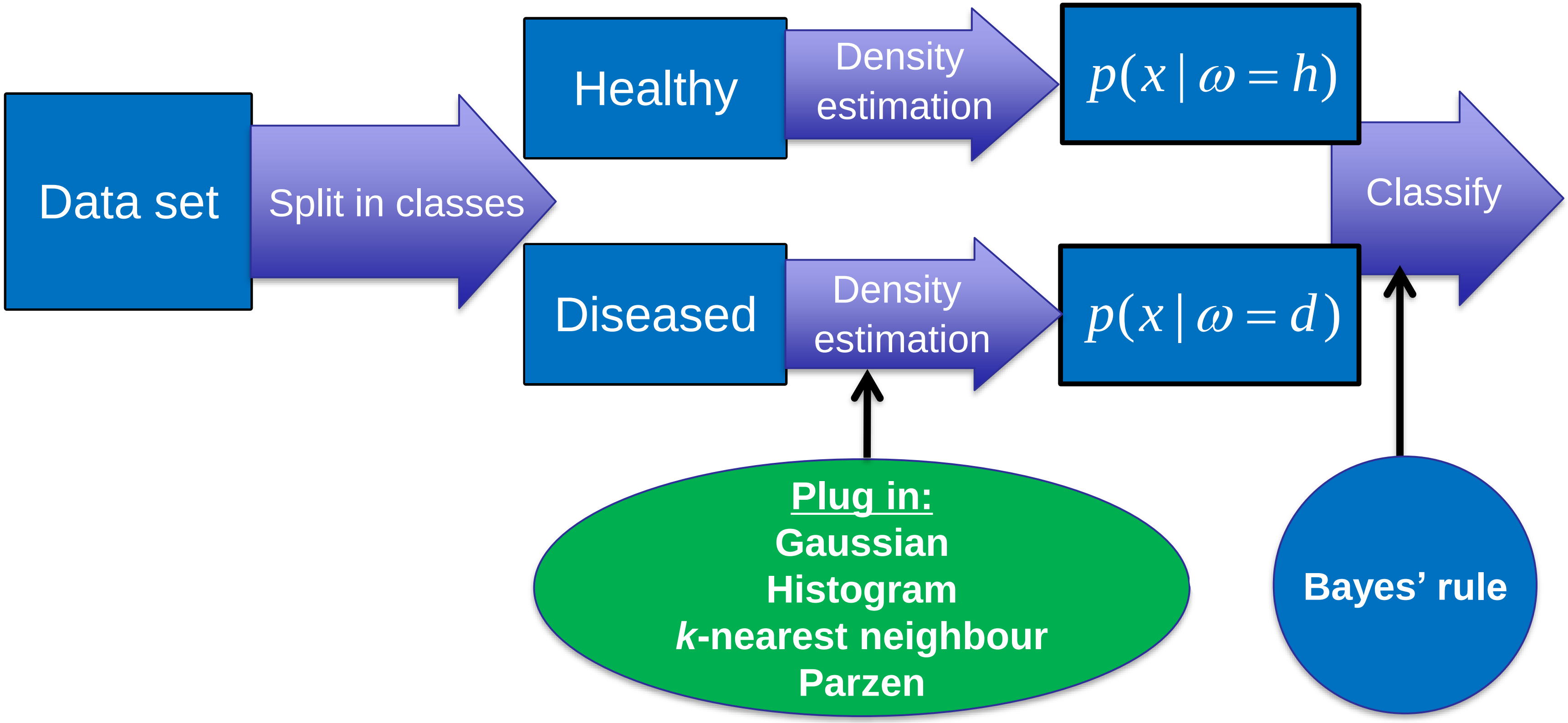


Error on test set (*true error*)

Error on training set (*apparent error*)

Bayesian classification

- In practice:



Recapitulation

- *Bayesian estimation*
  - provides a framework for minimizing cost due to errors
  - combines class-conditional and prior distributions into posterior ones
- We never *know* these distributions, so we have to *estimate* them; this is problematic due to the *curse of dimensionality*
- Possible approaches:
  - *Parametric*: e.g. Gaussian
  - *Nonparametric:* histogramming, *k*-nearest neighbor density estimation, Parzen density estimation

Recapitulation (2)

- *Maximum likelihood estimation* is a method for estimating parameters of density functions
- To optimize parameters, the error should be calculated on a *validation set*
- A completely independent *test set* should only be used to judge performance of the final classifier
- *Cross-validation* and *bootstrapping* can help to estimate performance when little data is available

Exercise 1.15-1.25