Basic concepts

Lecture 1a

Course leader: Oleg Sysoev

Course topics

Block 1

- Basic concepts in machine learning. Software for ML. Classification and regression
- Dimensionality reduction and model selection
- Kernel methods (SVM) and neural networks

Block 2

- Mixture models and ensemble methods

Course organization

- 1 topic= 3-4 lectures (campus) +1 lab (2h* 3, campus)+seminar (zoom)

- Course given as
  - 732A99 (9 ECTS): Block 1+Block 2
  - 732A68 (9 ECTS): Block 1+Block 2
  - TDDE01 (6 ECTS): Block 1

- **Labs**
  - Sign-up at LISAM, **exactly 3 persons**! (otherwise group may be split)
  - Takes around 8h , group report
  - Published a day in advance – try doing before attending the first lab session!
  - **Statement of Contribution:** describe clearly how each member contributed to the group report (what exactly was done by each person). Without it lab is automatically failed.
  - Offline short question answering on LISAM
  - Deadlines
  - To pass exam, **each student needs to have experience of solving all lab tasks** → make sure to try all tasks before the exam!
  - Submission via LISAM

Course organization

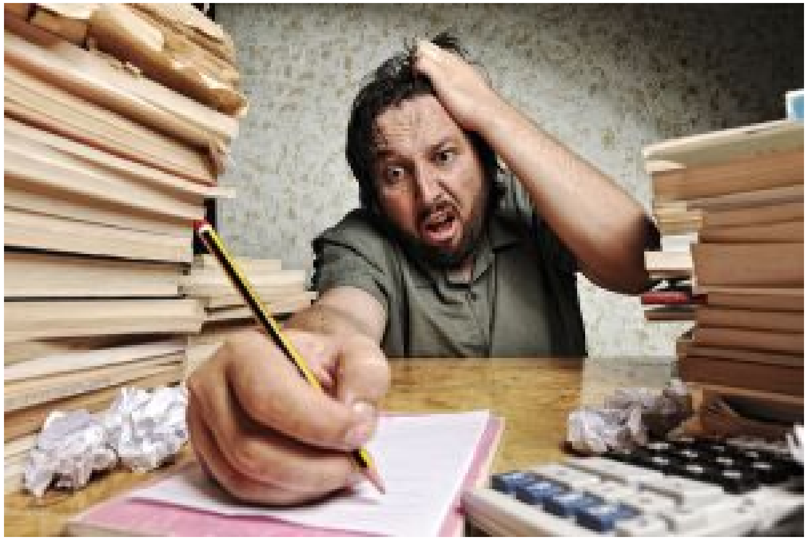
- Lectures
  - Available as PowerPoint or PDF, normally at LISAM

- Tutorials
  - Topic 1 and 2 block 1 have tutorials = basic exercises with answers. Go through **before** the respective lab!

- Seminars
  - Obligatory attendance of all seminars
  - Zoom
  - Speaker and opponent groups
  - Discussion of the latest lab
  - Presentation schedule will be published on LISAM (Seminars.PDF)

Course organization

- Examination
  - laboratory part + computer-based exam



http://www.swagseduction.com/wp-content/uploads/2014/11/stressful.jpg

- Lecture 1b is ’Basic Statistics’

- Lecture 1c is ’Introduction to R’

What is Machine Learning ?

- *Machine learning (ML) is a field of study in artificial intelligence concerned with the development and study of statistical algorithms that can learn from data and generalize to unseen data, and thus perform tasks without explicit instructions.*

Wikipedia (2024).

Machine Learning and Statistics

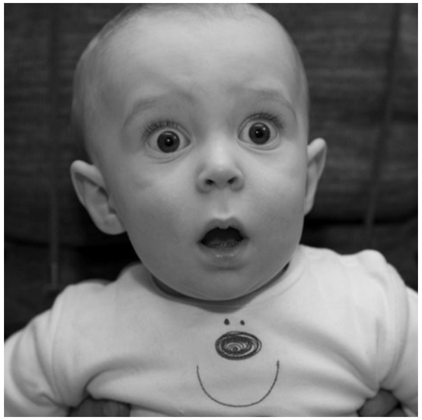
- ML **combines** of **computer science** and **statistics**.
  - Related: **data mining**, **knowledge discovery** and **data science**.
- ML often uses **statistical (probabilistic) models for analyzing data** .
  - Data mining and knowledge discovery tend to use less rigorous, but often effective, algorithms.
  - ML is not a discovery of a hidden information (Data Mining)
- ML vs Statistics: ML has a **heavier focus on prediction** , and lesser on interpretation.
- ML applications often involve large sets →**computational complexity** of algorithms is important.
  - Statistics often does not care about runtime

Why probability models?

- Probability models and statistical inference provide a **framework**

- A principled **way to think** about any problem in machine learning
  - Probabilistic model →Estimation →Prediction

- Probabilistic models **quantify uncertainties**.
  - Deterministic answers may often be inappropriate



http://lolnada.org/t/src/1454993210255.jpg

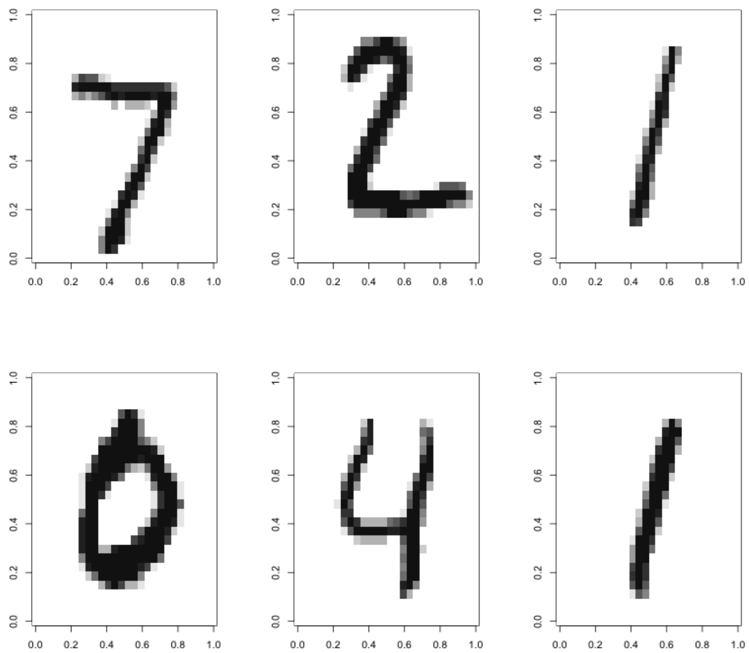
The currency exchange rate tomorrow will be 10.41!

Why probability models?

As robotics is now moving into the open world, the issue of ***uncertainty*** *has become a major stumbling block for the design of capable robot systems. Managing uncertainty is possibly the most important step towards robust real-world robot systems.*

from the book Probabilistic Robotics by Thrun et al.

Example: classifying hadwritten digits



Example: classifying hadwritten digits

Training data: 60000 images.

Test data: 10000 images.

Features: intensities (0-255, scaled to 0-1) in the $28 \times 28 = 784$ pixels as features.

Methods:

- Multinomial classification with LASSO regularization
- Support vector machines
- Neural Networks (deep?)

Example: classifying hadwritten digits

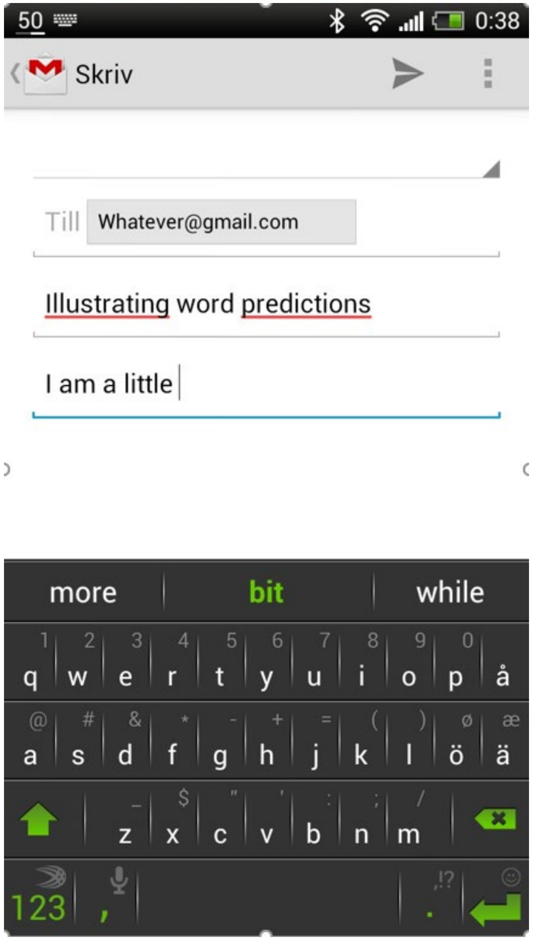
- Confusion matrix

PREDICTION

TRUTH

	0	1	2	3	4	5	6	7	8	9
0	966	0	8	1	1	7	9	2	4	6
1	0	1121	1	1	0	2	3	13	7	7
2	2	2	957	13	5	4	4	21	7	0
3	0	2	9	947	0	29	1	3	12	10
4	0	0	12	1	940	5	5	9	8	32
5	6	1	3	19	1	816	9	1	24	9
6	4	4	13	1	7	12	926	0	10	1
7	1	0	9	10	2	2	0	954	5	13
8	1	4	17	11	2	10	1	3	892	4
9	0	1	3	6	24	5	0	22	5	927

Example: smartfone typing predictions



Example: smartfone typing predictions

- Markov Model of the sentence and Bayes theorem:

$$p(w_n|w_1,\dots,w_{n-1}) = \frac{p(w_1)p(w_2|w_1)\dots p(w_n|w_{n-1})}{p(w_n)}$$

- Intuition:
  - $p(person|intelligent) = 0.1$
  - $p(tree|intelligent) = 0.0001$

Highest P(?|Donald) ?

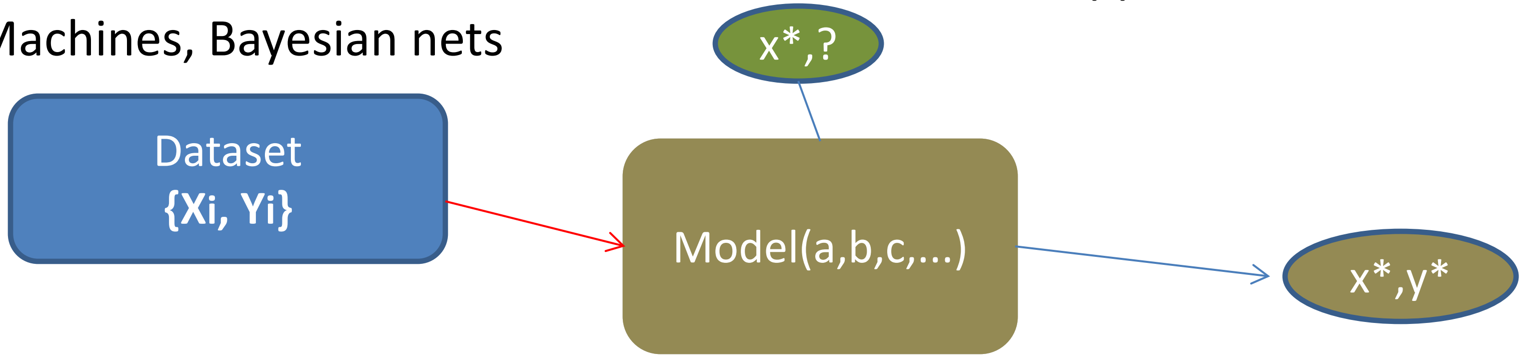
- Probability for sentence depends only on $p(w_n|w_{n-1})$
- How to compute ? Investigate a lot of data!

$$p(w_k|w_{k-1}) = \frac{\#\ cases\ w_k\ follows\ w_{k-1}}{\#\ cases\ w_{k-1}}$$

- In practice, more advanced model used
  - Neural networks for ex.

Types of learning

- **Supervised learning** (classification, regression)
  - Compute parameters from data
  - Given features of a new object, predict target (generalize beyond seen training data)
  - **Classification** (Y=categorical), **Regression** (Y=continuous)
- Most of ML models: Neural Nets, Decision Trees, Support Vector Machines, Bayesian nets



Types of learning

- Unsupervised learning (→Data Mining)
  - No target
  - Aim is to extract interesting information about
    - Relations of parameters to each other
    - Grouping of objects

Ex: clustering, density estimation, association analysis

x1<-> x2<-> x3...

Types of learning

- **Semi-supervised**: targets are known only for some observations.
- **Active learning**. Strategies for deciding which observations to label
- **Reinforcement learning**. Find suitable actions to maximize the reward. True targets are discovered by trial and error. (ex. ChatGPT)
- **Transfer learning:** use knowledge from some domain to train better models in a similar domain

Basic ML ingridients

- **Data** T: observations (cases)
  - Features $x_1, .. x_p$
  - Targets $y_1, \dots, y_r$

Case	x_1	x_2	y
1			
2			
…			

- **Mathematical Model** $P(x|\, w_1, \dots w_k)$ or $P(y|x, w_1, \dots w_k)$
  - Example: Linear regression $\mathrm{p}(y|x, w_o, w_1, \sigma^2) = N(w_0 + w_1 x, \sigma^2)$
- **Learning algorithm** (data→get parameters $\widehat{w}$ or $p(w|D)$)
  - Maximum likelihood, Bayesian estimation…
- **Prediction** of new data x_* by using the fitted model

Types of data sets

- **Training data** (training set T): used for learning the model
  - Supervised learning: w_i in $P(y|\boldsymbol{x}, w_1, \dots w_k)$ estimated using T

X	Y
1.1	M
2.3	F

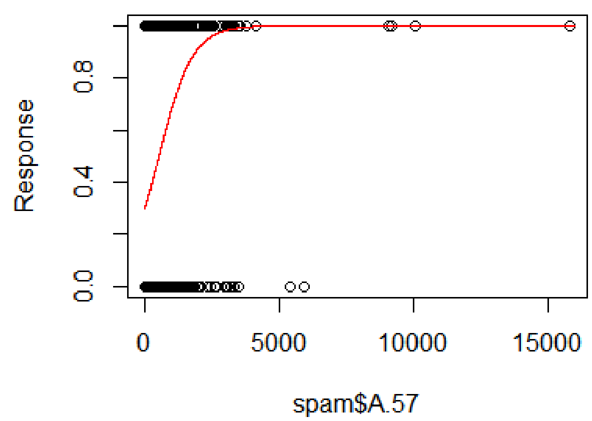
- **Test data** (test set T*): used for predictions
  - Supervised learning: estimate $p(y_*)$ or $\hat{y}_*$ for new $\boldsymbol{x}_*$

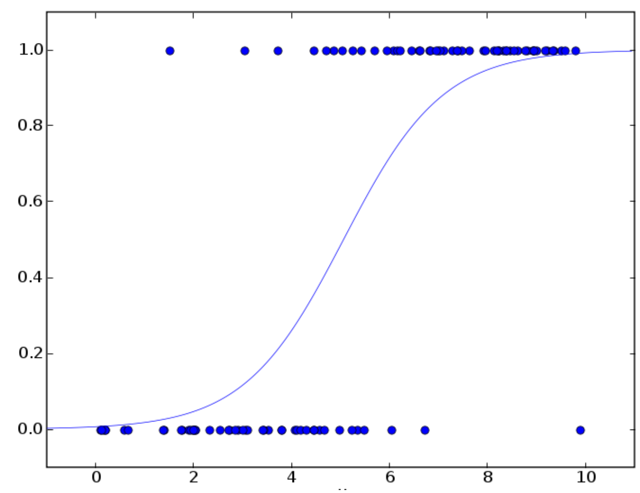
X	Y
1.3	?
2.9	?

Logistic regression

- **Data** $y_i \in \{Spam, Not\ Spam\}, x_i = \#of\ a\ word$
- **Model**: $p(y = Spam|w, x) = \frac{1}{1+e^{-w_0-w_1x}}$
- **Learning algorithm**: maximum likelihood
- **Prediction** : $p(spam) = p(Y = spam|x_*)$

We can also make point predictions
-how?

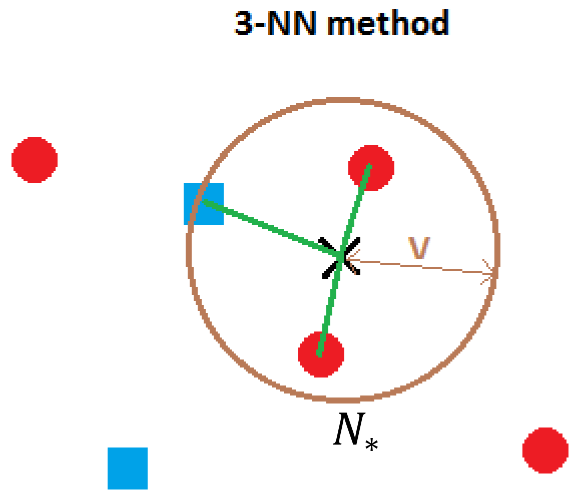




K-nearest neighbor model

- Can be classification or regression

- Basic idea:
  - For given x_*, find K nearest observations
  - Classification: majority voting
  - Regression: compute mean



- K is called **hyperparameter**

K-nearest neighbor algorithm

Data: Training data $\{\mathbf{x}_i, y_i\}_{i=1}^{n}$ and test input $\mathbf{x}_\star$
Result: Predicted test output $\widehat{y}(\mathbf{x}_\star)$

Compute the distances $\|\mathbf{x}_i - \mathbf{x}_\star\|_2$ for all training data points $i = 1, \ldots, n$

Let $\mathcal{N}_\star = \{i : \mathbf{x}_i$ is one of the k data points closest to $\mathbf{x}_\star\}$

Compute the prediction $\widehat{y}(\mathbf{x}_\star)$ as

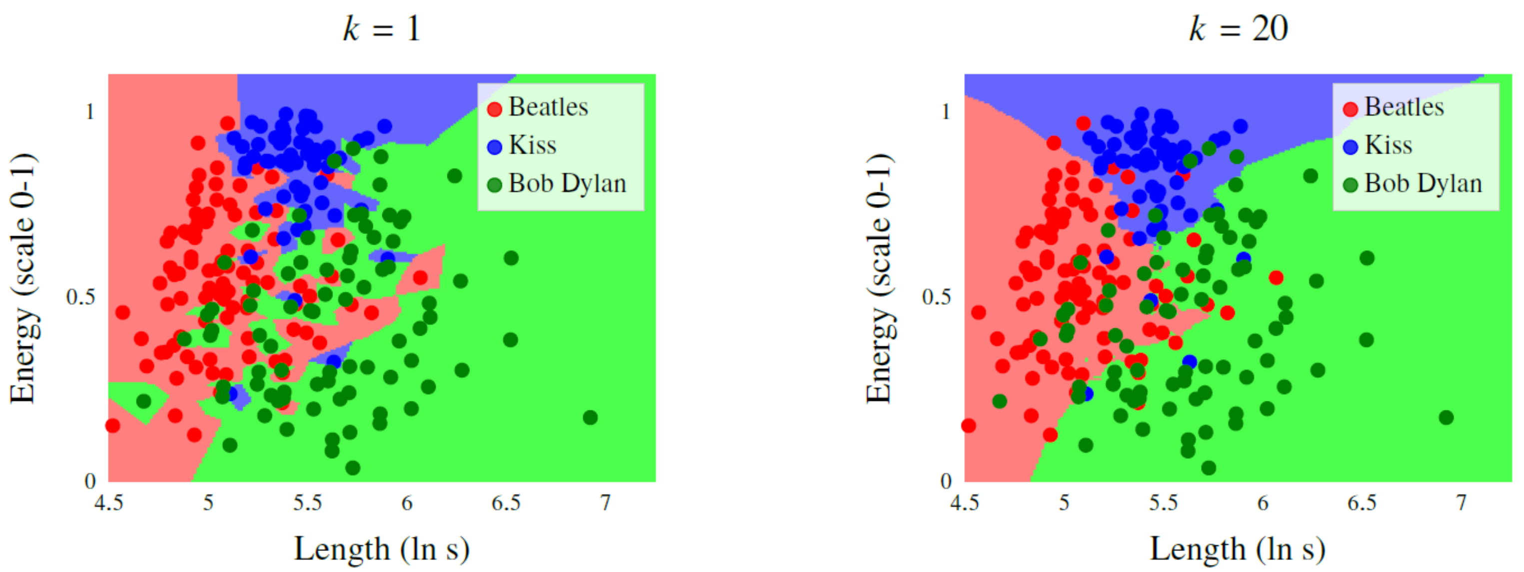
$$\widehat{y}(\mathbf{x}_\star) = \begin{cases} \text{Average}\{y_j : j \in \mathcal{N}_\star\} & \text{(Regression problems)} \\ \text{MajorityVote}\{y_j : j \in \mathcal{N}_\star\} & \text{(Classification problems)} \end{cases}$$

K-nearest neighbor model

- **Data** $\mathrm{T} = \{(\boldsymbol{x}_i, y_i), i = 1, \ldots, n\}$
- **Model**: W same size as T
- **Learning algorithm**: Set W=T, compute distances in W
- **Prediction**:
  - $y_* = \frac{1}{|N_*|} \sum_{i \in N_*} y_i$
  - $y_* = MajorityVote_{j \in N_*}(y_j)$

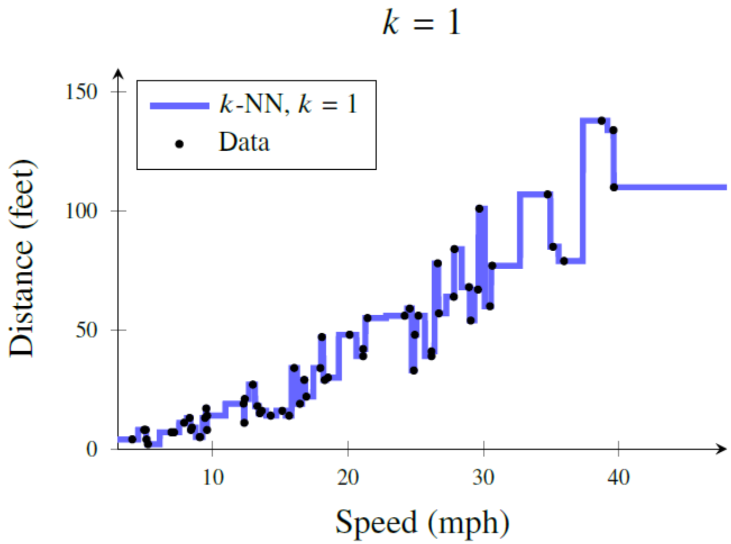
K-nearest neigbor example

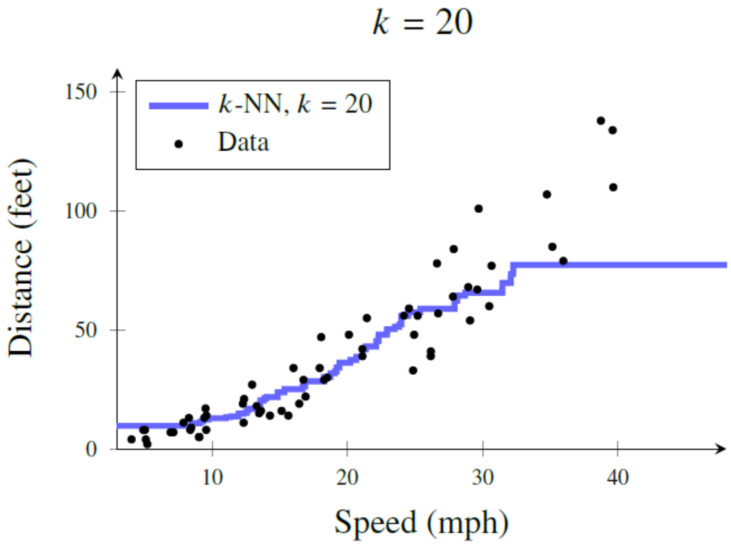
- Classification
  - Music data, x1=song length, x2=a signal processing characteristic



K-nearest neighbor example

- Regression
  - Car data: x1 speed when brake signal given, x2 distance until full stop

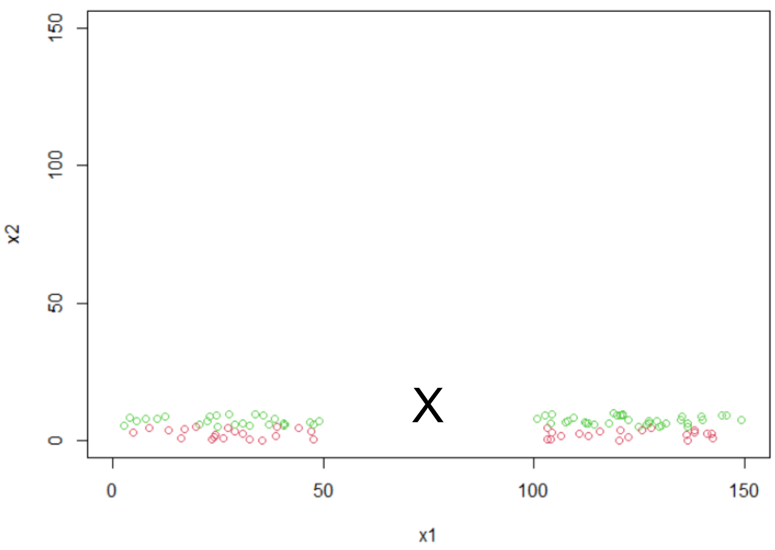


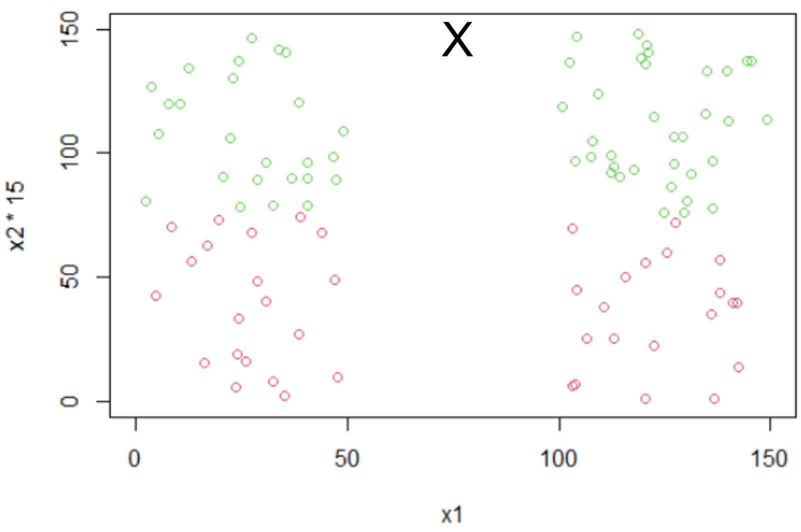


How to choose K?

K-nearest neighbor model

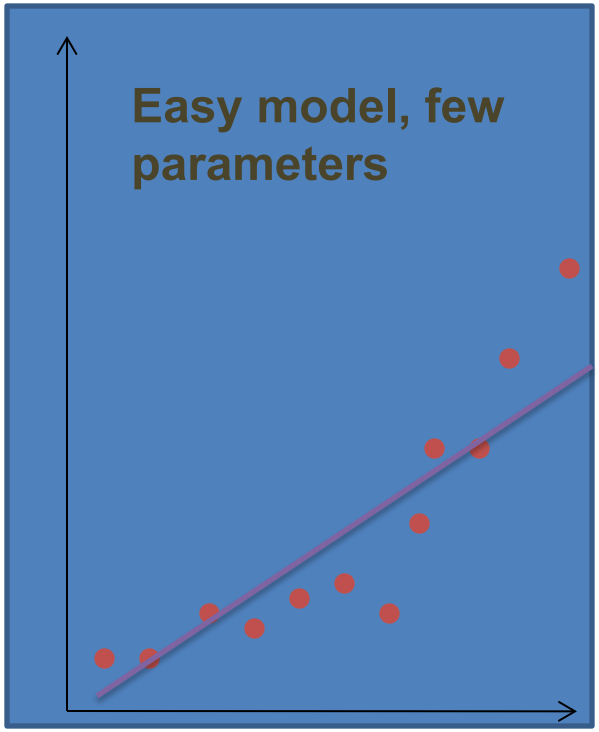
- **Feature preprocessing**: scaling
  - Important, for ex when defining distance
  - Usual preprocessing: $x'_{ij} = \frac{x_{ij} - mean(x_{ij}|\ i=1,\dots n)}{std(x_{ij}|i=1,\dots n)}$

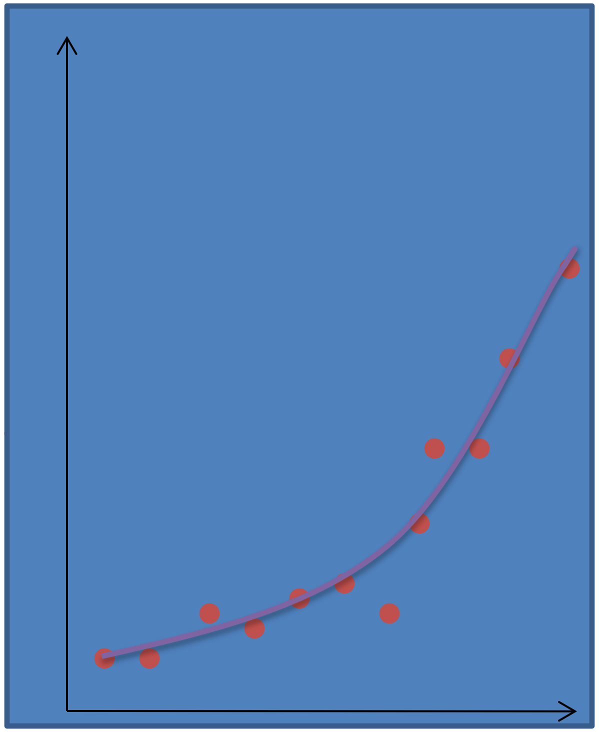


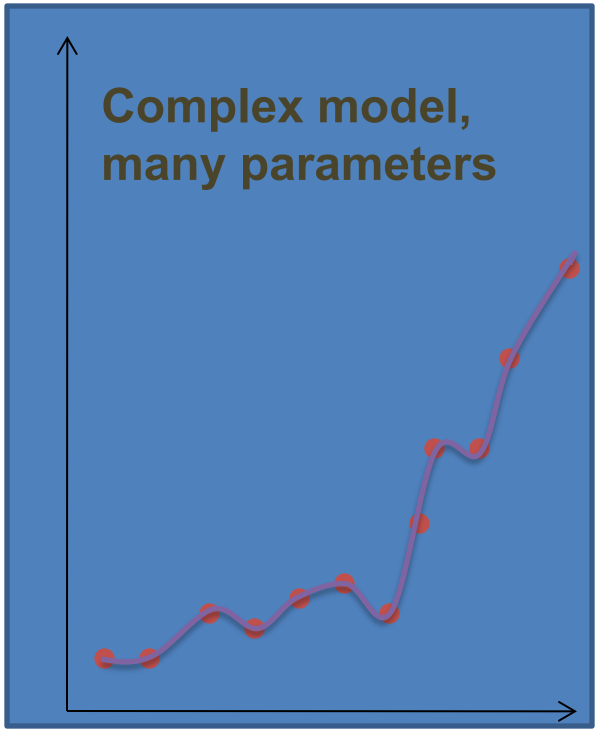


Overfitting

- Which model feels appropriate?

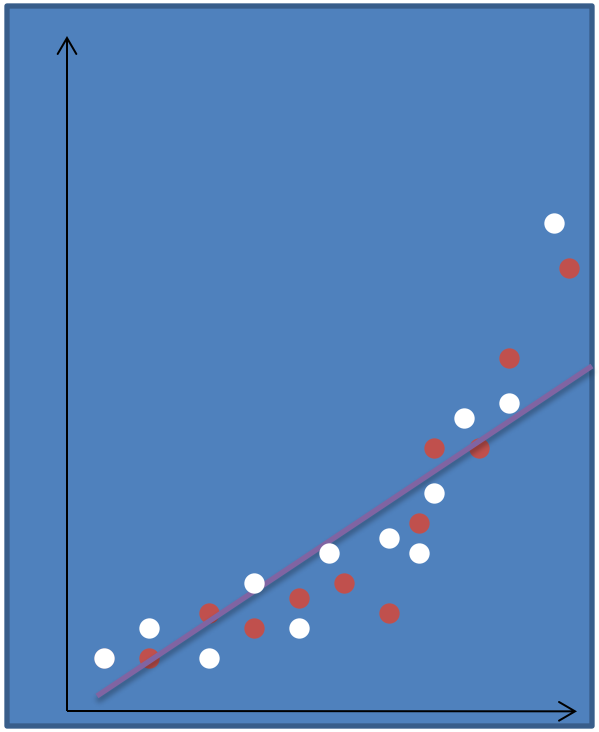


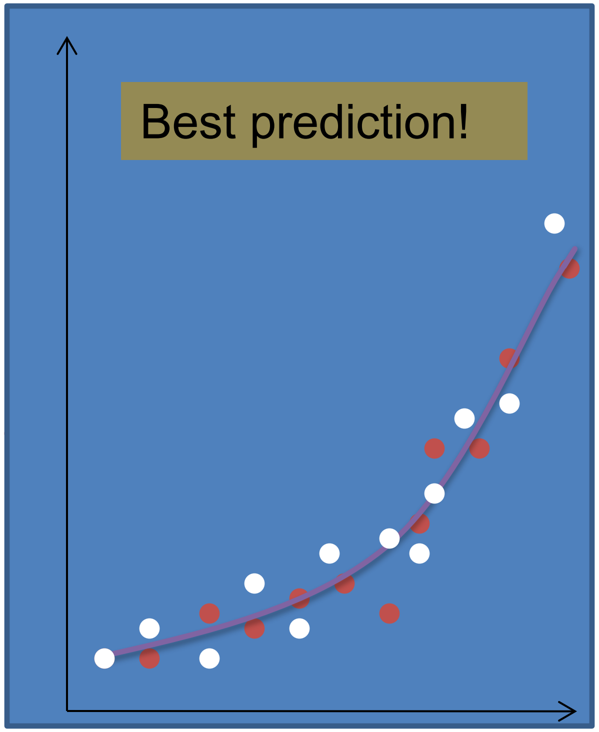


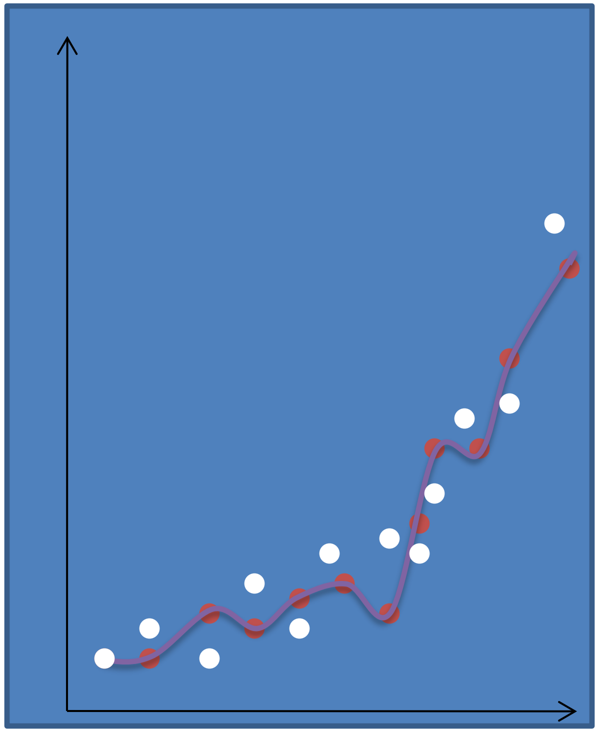


Overfitting

Now new data from the same process

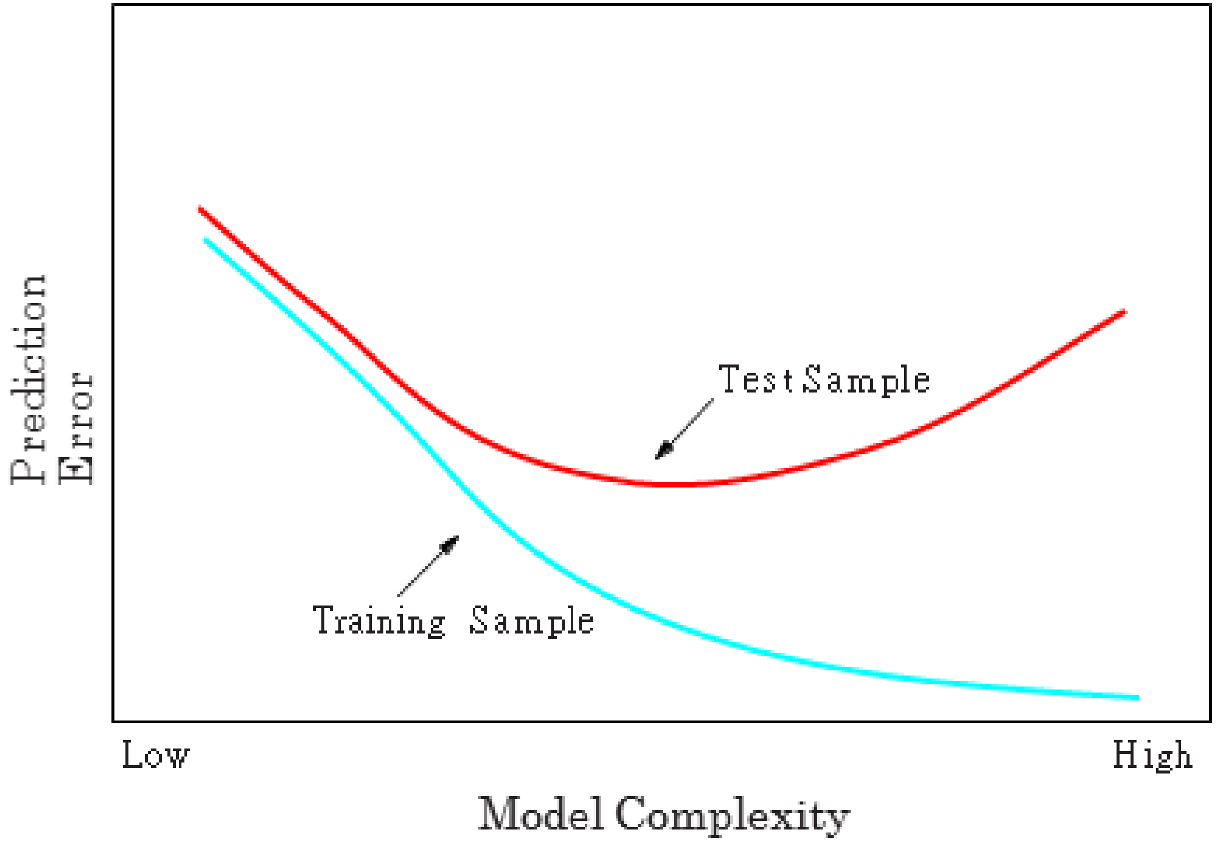






Overfitting

- Observed:



Model selection

- Given several models $M_1, \ldots M_m$
- Divide data set into **training** and **test** data

Training	Test

- Fit models M_i to training data→get parameter values
- Use estimated models to predict test data and compare **test errors** $R(M_1), \ldots R(M_m)$
- Model with lowest prediction error is best

Comment:

- Approach works well for moderate/large data

Holdout method

Divide into training, validation and test sets

Training	Validation	Test

- Choose proportions in some way
- Test set is used to test a performance on a new data

Holdout in R

- How to partition into train/test?
  - Use set.seed(12345) in the labs to get identical results

```
n=dim(data)[1]
set.seed(12345)
id=sample(1:n, floor(n*0.7))
train=data[id,]
test=data[-id,]
```

- How to partition into train/valid/test?

```
n=dim(data)[1]
set.seed(12345)
id=sample(1:n, floor(n*0.4))
train=data[id,]

id1=setdiff(1:n, id)
set.seed(12345)
id2=sample(id1, floor(n*0.3))
valid=data[id2,]

id3=setdiff(id1,id2)
test=data[id3,]
```

Typical error functions

- Regression, **MSE** :

$$R(Y,\hat{Y}) = \frac{1}{N}\sum_{i=1}^{N}(Y_i - \hat{Y}_i)^2$$

- Classification, **misclassification rate**

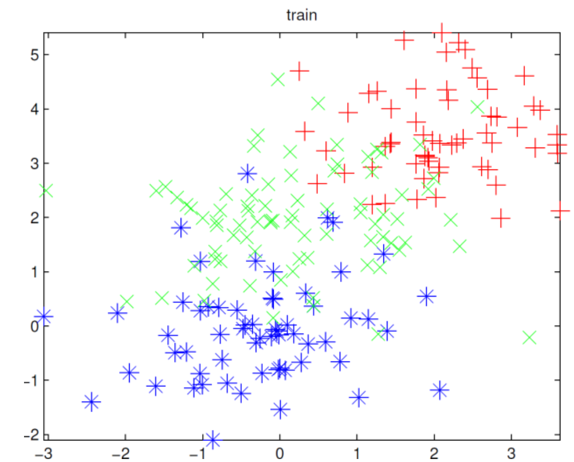
$$R(Y,\hat{Y}) = \frac{1}{N}\sum_{i=1}^{N} I(Y_i \neq \hat{Y}_i)$$

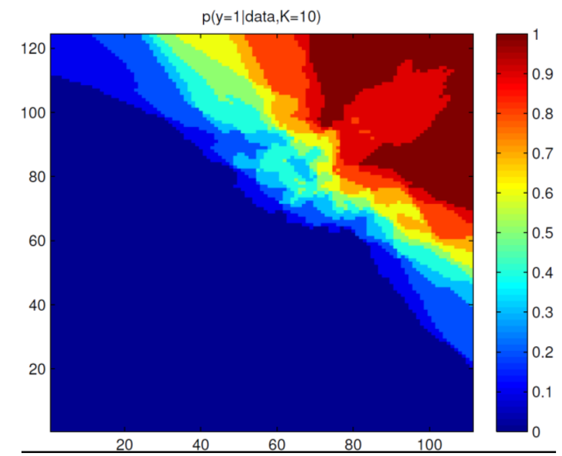
- Classification, cross-entropy for M classes $C_1, \ldots, C_M$:

$$R(Y,\hat{p}(Y)) = -\sum_{i=1}^{N}\sum_{m=1}^{M} I(Y_i = C_m)\log\hat{p}(Y_i = C_m)$$

Model types

- Parametric models
  - Have certain number of parameters independently of the size of training data
  - Assumption about of the data distribution
  - Ex: logistic regression
- Nonparametric models
  - Number of parameters (complexity) changes with training data
    - Example: K-NN classifier

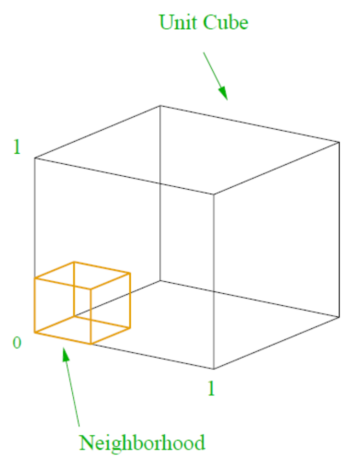


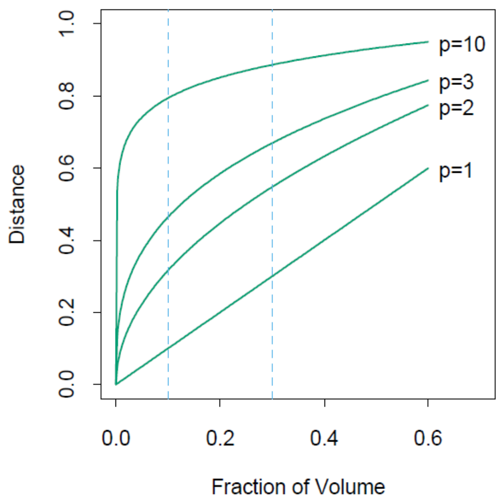


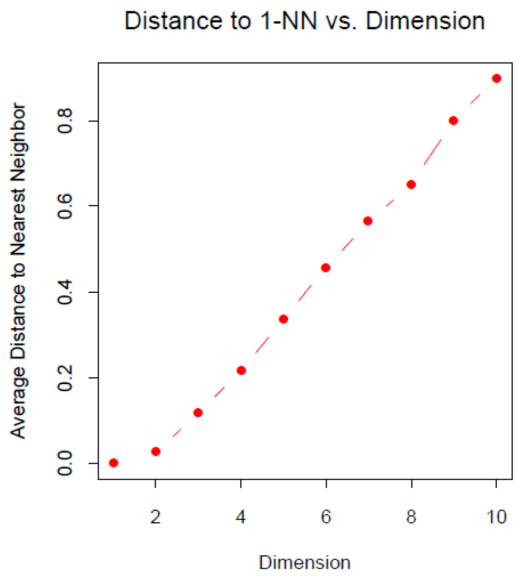
Curse of dimensionality

- Given data T:
  - Features $x_1, .. x_p$
  - Targets $y_1, \dots, y_r$
- When p increases models using ”proximity” measures work badly
- Curse of dimensionality: A point has no ”near neighbors” in high dimensions → using class labels of a neighbor can be misleadning
  - Distance-based methods affected

Curse of dimensionality







Curse of dimensionality

- Hopeless? No!
- Real data normally has much lower effective dimension
  - Dimensionality reduction techniques
- Smoothness assumption
  - small change in one of x's should lead to small change in y→interpolation