Basic concepts

Lecture 1a

Course leader: Oleg Sysoev

Course topics

Block 1

- Basic concepts in machine learning. Software for ML.
- Regression, regularization and model selection
- Classification methods
- Dimensionality reduction and uncertainty estimation
- Support vector machines and kernel methods
- Neural networks and deep learning

Block 2

- Splines and additive models. High-dimensional problems
- Mixture models and online learning. Ensemble methods

Course organization

- 1 topic= 4-5 lectures +1 lab (2h* 3)+seminar

- Course given as
  - 732A99 (9 ECTS): Block 1+Block2
  - 732A68 (9 ECTS): Block 1+Block2
  - TDDE01 (6 ECTS): Block 1

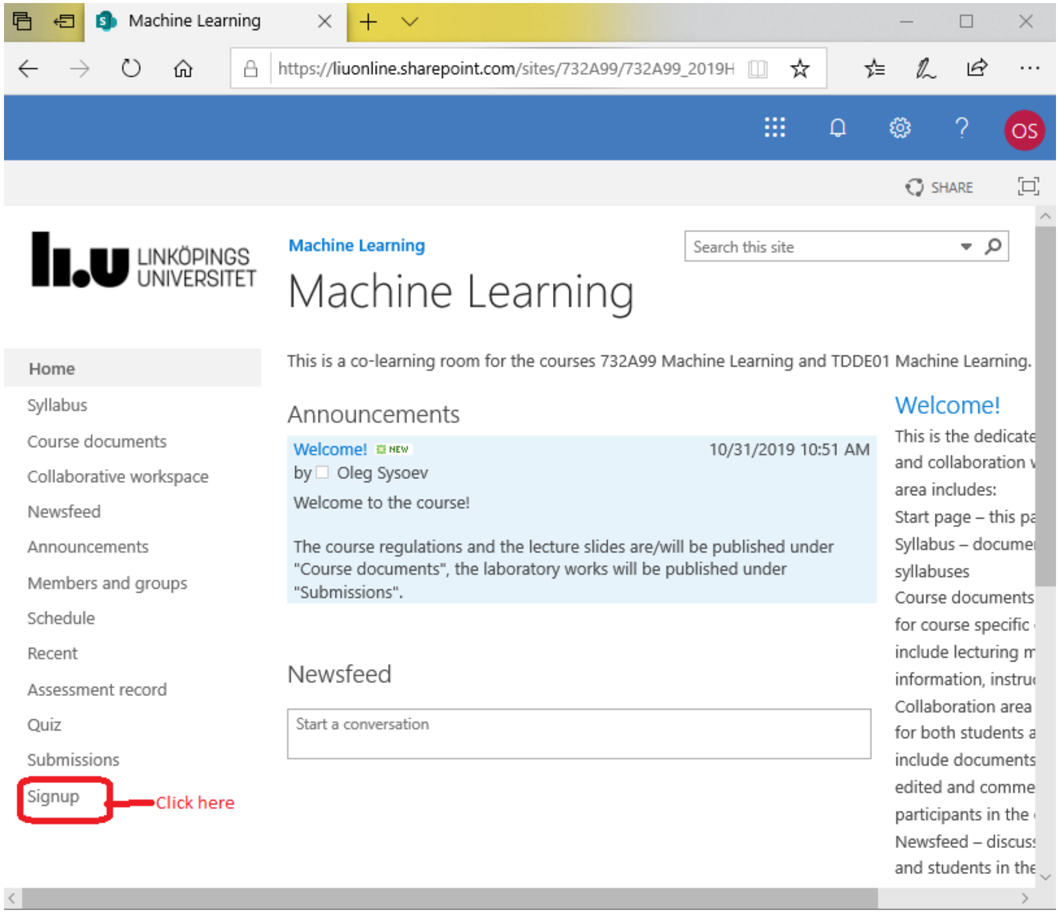
- **Labs**
  - SU rooms used
  - Take around 8h
  - Individual and group reports
  - Sharing only ideas in the group, not text or codes
  - Bring your own laptop if you have – limited amount of computers in the rooms
  - Deadlines
  - Individual Special tasks (optional)– if you solve all of them and get at least 14 points at the exam, you get 2 points more.
  - Published a couple of days in advance – try doing before coming to the first lab session!
  - Submission via LISAM

Course organization

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

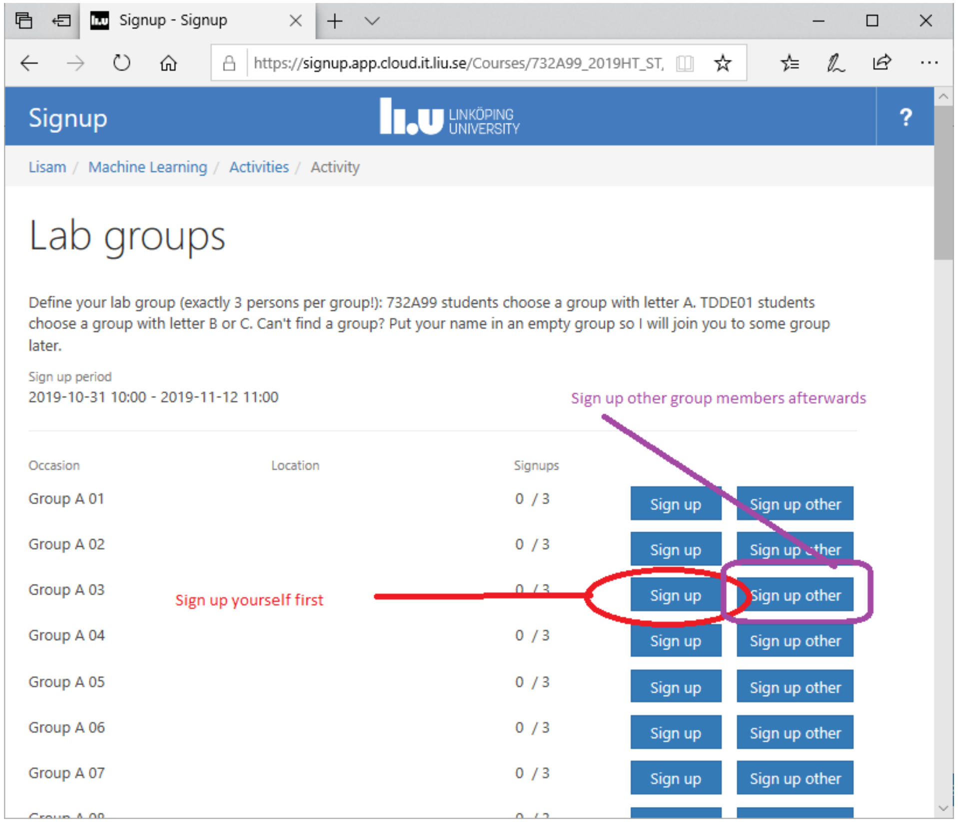
- Seminars
  - Speaker and opponent groups
  - Is a laboratory part, obligatory attendance for speakers and opponents
  - Discussion of the latest lab.
    - Note: lab assignments are slighlty different for TDDDE01/732A99 but all kinds of assignments may appear at the exam!
  - Define your group (3 persons) as soon as possible via Lisam (see next two slides)
    - **Difficult to find a group? Put your name in some empty group item**

Define your group



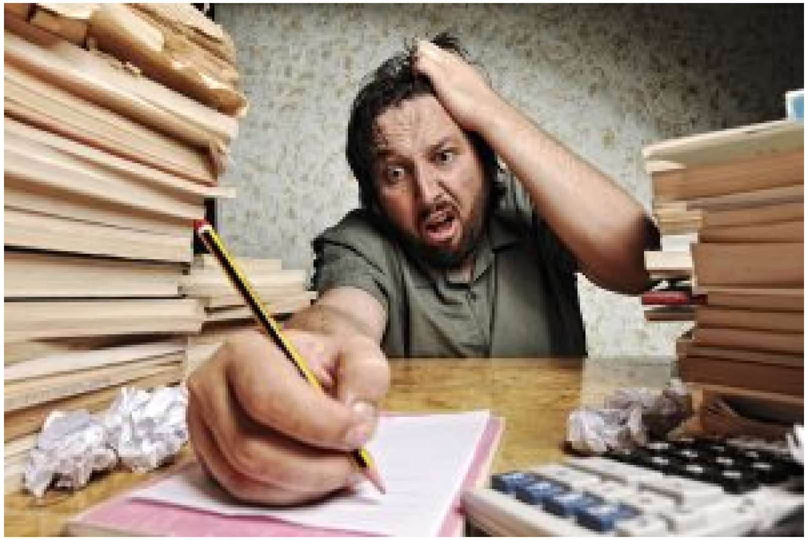
Define your group

- 732A99: Use any empty group with letter **A**
- TDDE01: Use any empty group with letter **B** or **C**



Course organization

- Examination
  - TDDE01, 732A99: laboratory part + computer-based exam



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

- Lecture 1c is 'Introduction to R'
- Lecture 1b is 'Basic Statistics'

What is Machine Learning ?

- *Machine learning is a subfield of* **computer science** *that evolved from the study of* ***pattern recognition*** *and computational learning theory in* ***artificial intelligence***.

- *Machine learning explores the study and construction of* ***algorithms*** *that can* ***learn*** *from and make* ***predictions*** *on* ***data***. *Such algorithms operate by building a model from example inputs in order to make data-driven predictions or* ***decisions***, *rather than following strictly static program instructions.*

Wikipedia (Oct 15, 2016).

Machine Learning and Statistics

- ML=**intersection** of **computer science**, **statistics** and **artificial intelligence**.
  - Related: **data mining**, **knowledge discovery** and **data science**.

- ML uses mainly **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.

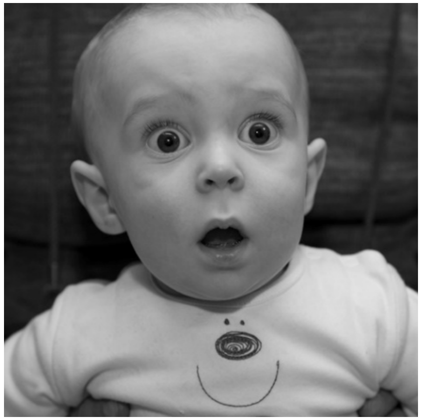
- ML applications often involve large sets →**computational complexity** of algorithms is important.
  - Statistics often does not care about runtime

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

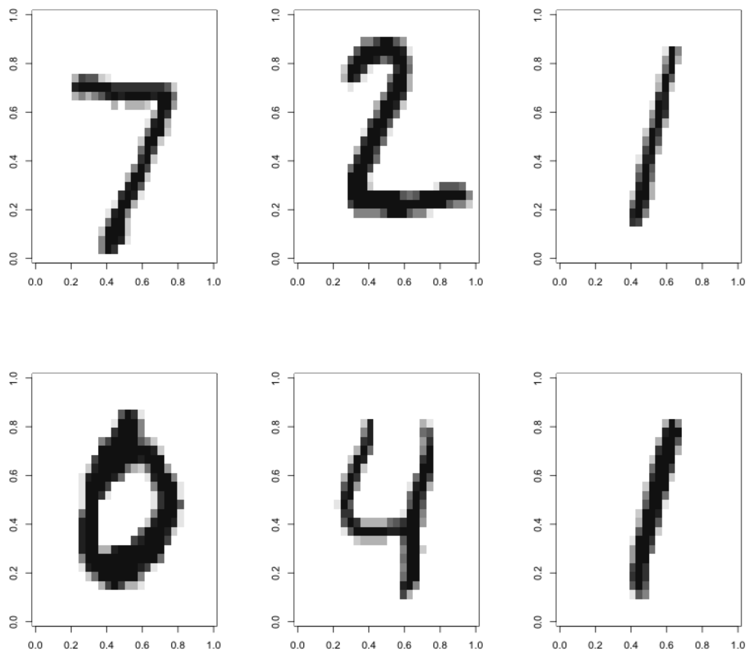
The currency exchange rate tomorrow will be 10.41!

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 regression with LASSO prior
- Support vector machines
- Neural Networks (deep?)

Example: classifying hadwritten digits

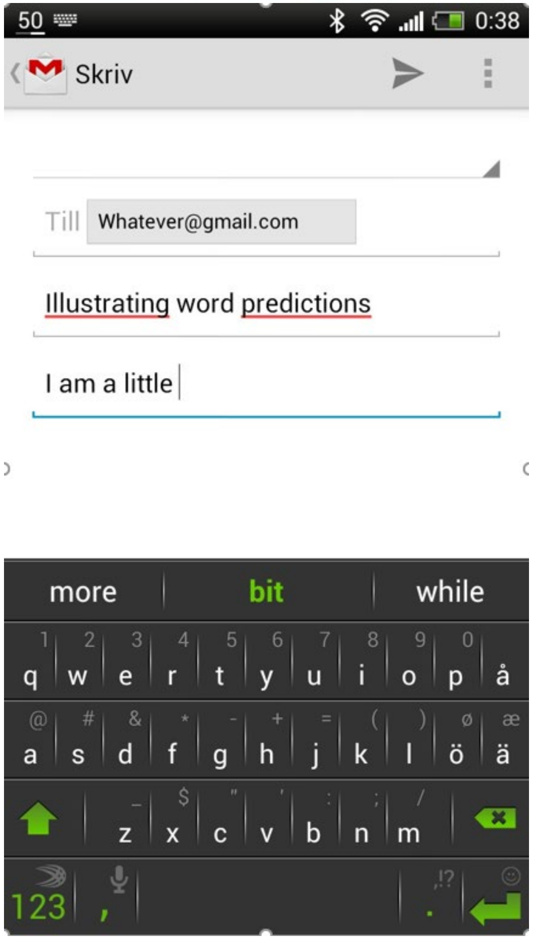
- Confusion matrix

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

- Assume a simple (Markov) model of a sentence:
$$p(w_1, \ldots, w_n) = p(w_1)p(w_2|w_1) \ldots p(w_n|w_{n-1})$$

- Intuition:
  - $p(person|crazy) = 0.1$
  - $p(horse|crazy) = 0.0001$

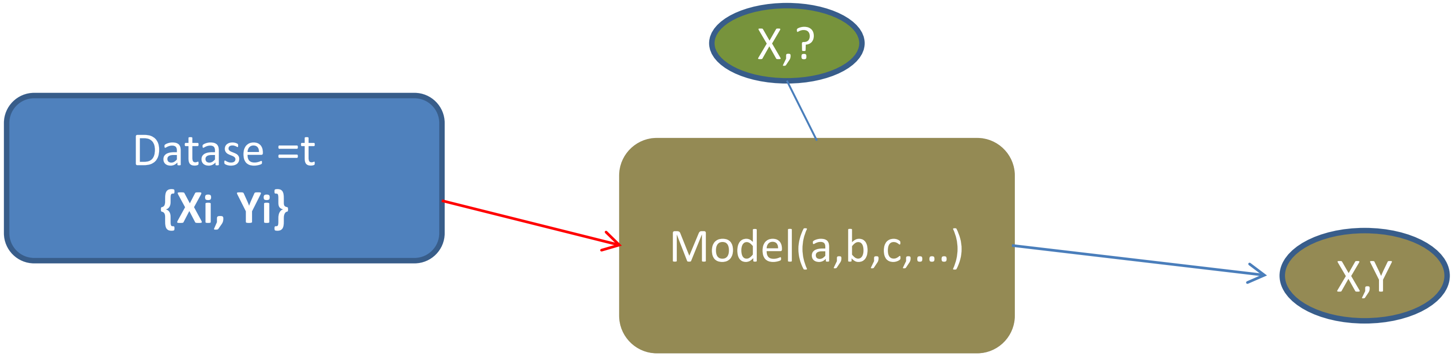
Highest P(?|Donald) ?

- Probability for sentence depends only on $p(w_n|w_{n-1})$
- How to compute ? Investigate a lot of data!
$$p(w_k|w_{k-1}) = \frac{\#\ cases\ w_k\ follows\ w_{k-1}}{\#\ cases\ w_k}$$

- 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
  - **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.

Basic ML ingridients

- **Data** D: observations (cases)
  - Features $X_1, .. X_p$
  - Targets $Y_1, \dots, Y_r$

Case	X_1	X_2	Y
1			
2			
…			

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

Types of data sets

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

X	Y
1.1	M
2.3	F

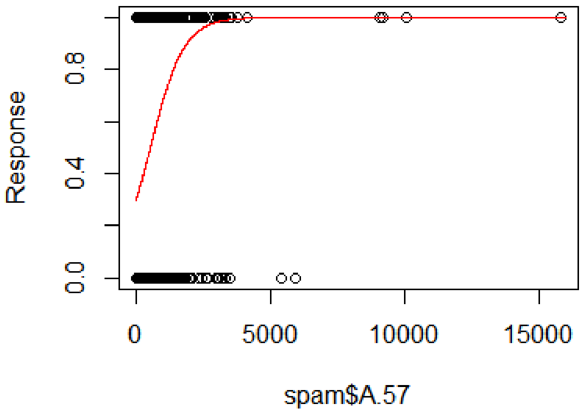
- **Test data** (test set T): used for predictions
  - Supervised learning: estimate $p(Y)$ or $\hat{Y}$ for new $\boldsymbol{x}$

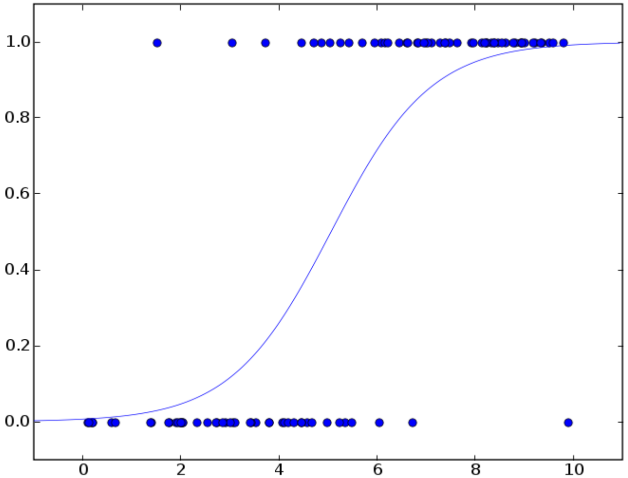
X	Y
1.3	?
2.9	?

Logistic regression

- Data $Y_i \in \{Spam, Not\ Spam\}, X_i = \#of\ a\ word$
- Model: $p(Y = Spam|w, x) = \frac{1}{1+e^{-w_0-w_1 X}}$
- Fitting: maximum likelihood
- Prediction : $p(spam) = p(Y = spam|x)$

We can also make point predictions
-how?



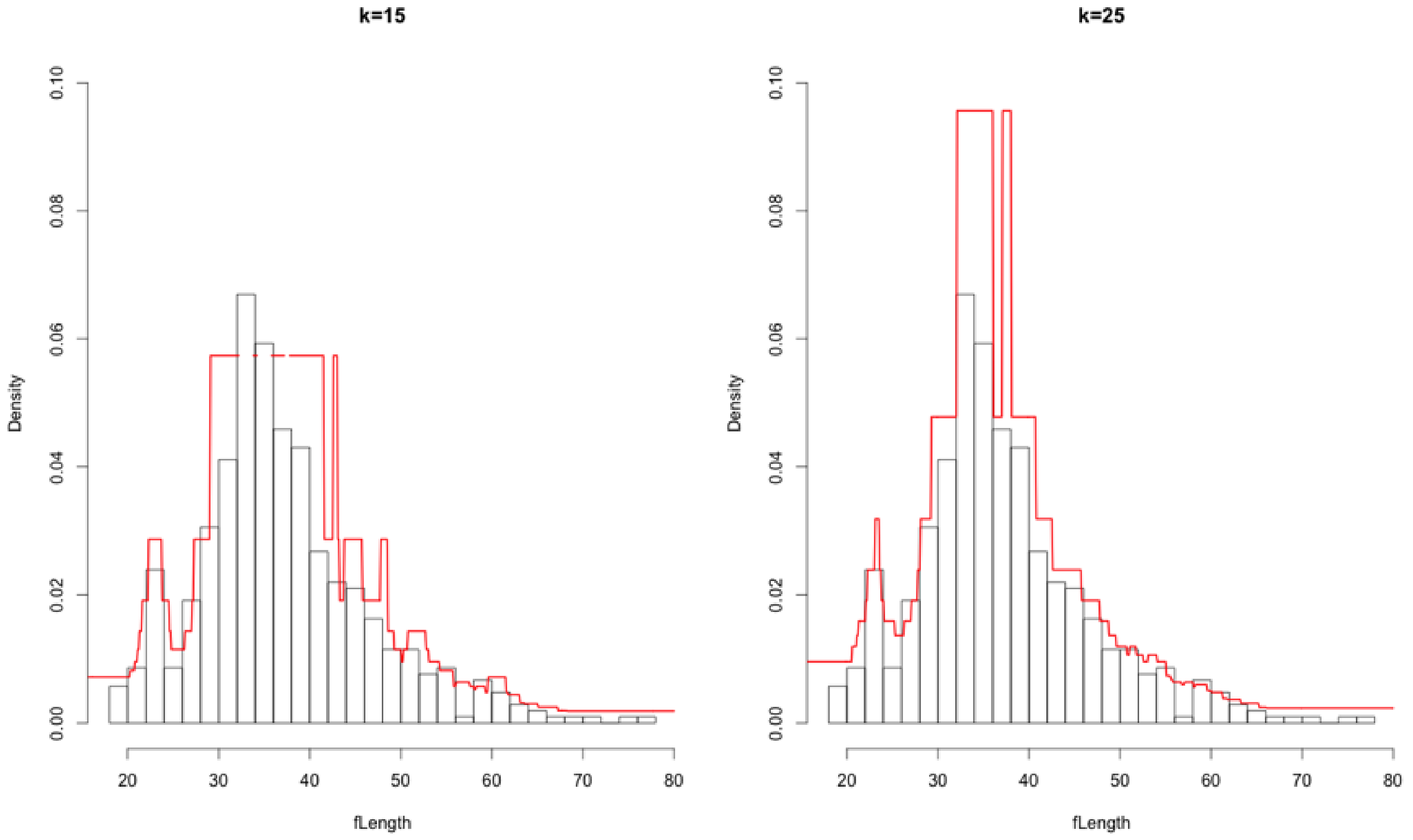


K-nearest neighbor density estimation

- Data: Fish length $X_1, \dots X_N$
- Model $p(x|K) = \frac{K}{N \cdot \Delta}$
  - K: #neighbors in training data
  - Δ: length of the interval containing K neighbors

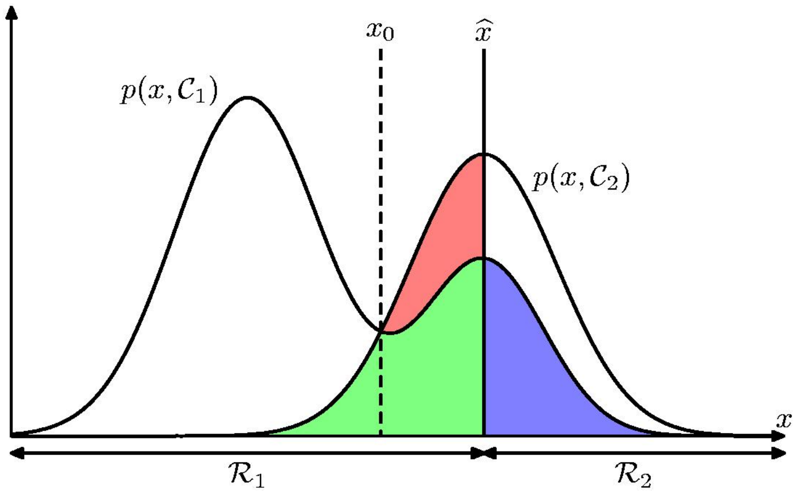
- Learning: Fix some K or find an appropriate K
- Prediction: predict $p(x|K)$

K-nearest neighbor density estimation



K-nearest neighbor density estimation

- Why estimating a density can be interesting:
  1. Estimate **class-conditional densities** $p(x|y = C_i)$
  2. Predict



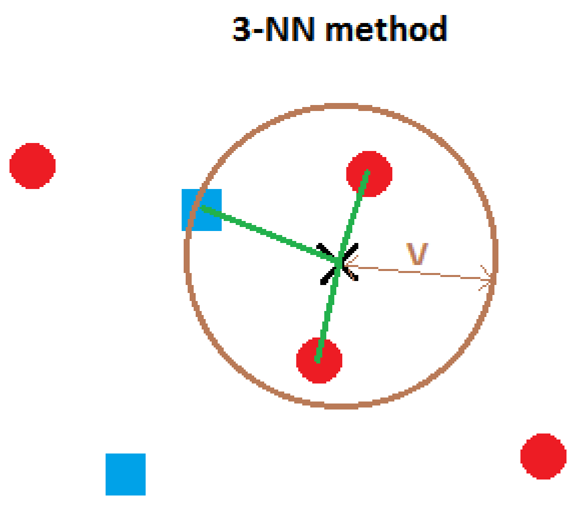
K-nearest neighbor classification

- Given N observations $(\boldsymbol{X}_j, Y_j)$
  - $Y_j = C_i$, where $C_1, \ldots C_m$ are possible class values

- Model assumptions
  - Apply K-NN density estimation:

$$p(X = x | Y = C_i) = \frac{K_i}{N_i V}, p(C_i) = \frac{N_i}{N}$$

  - V: volume of the sphere
  - K_i: #obs from training data of $Y = C_i$ in the sphere
  - N_i: #obs from training data of $Y = C_i$



Bayesian classification

- Prediction $\hat{Y}(\boldsymbol{x}) = C_l$

$$l = \arg \max_{i \in \{1,\ldots,m\}} p(C_i | \boldsymbol{x})$$

- Bayes theorem

$$p(C_i | \boldsymbol{x}) = \frac{p(\boldsymbol{x} | C_i) p(C_i)}{p(\boldsymbol{x})}$$

- We get

$$p(C_i | x) \propto \frac{K_i}{K}$$

K-nearest neighbor classification

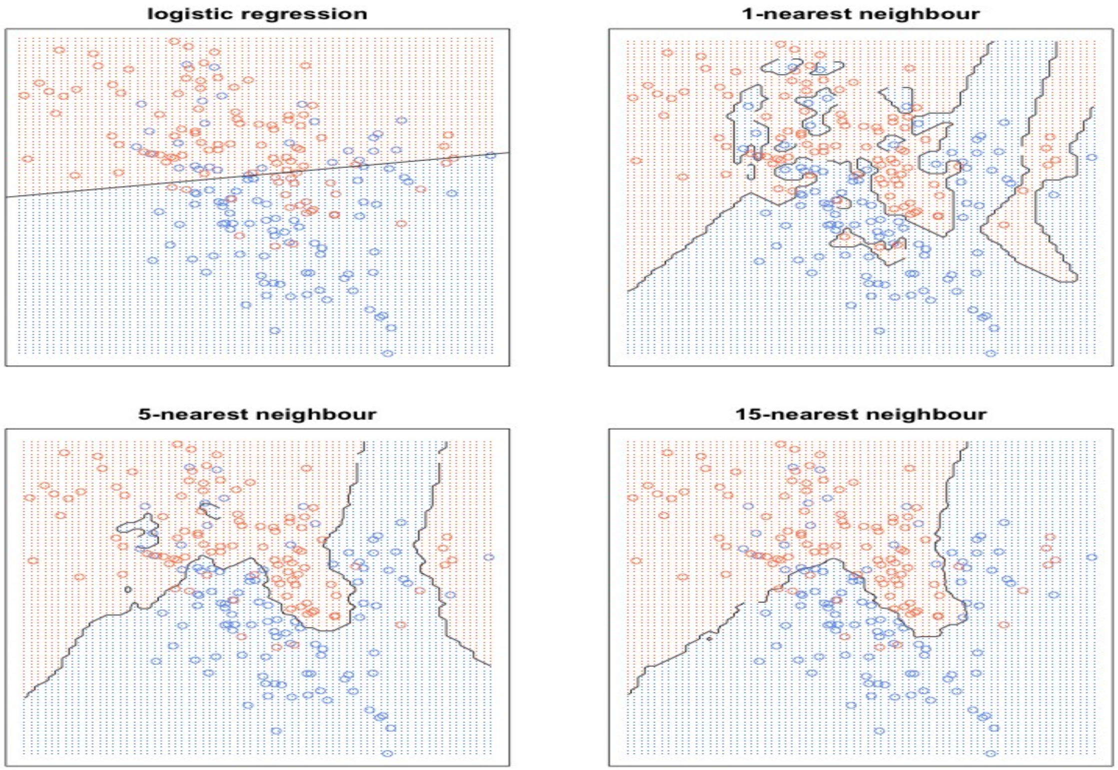
Algorithm

1. Given training set D,number K, and test set T
2. For each $x \in T$
   1. For each $i = 1, \ldots M$
      1. $p'(C_i|x) = \frac{K_i}{K}$
   2. Compute $l = \arg \max_{i \in \{1,\ldots,m\}} p'(C_i|\boldsymbol{x})$
   3. Predict $\hat{Y}(x) = C_l$

Majority voting: prediction for x is defined by majority voting of K neighbors

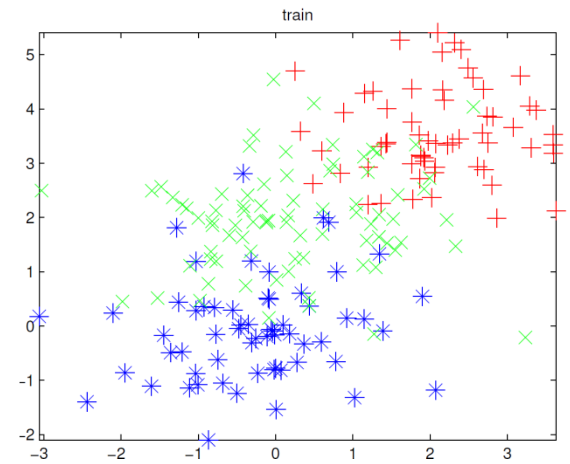
K-nearest neigbor example

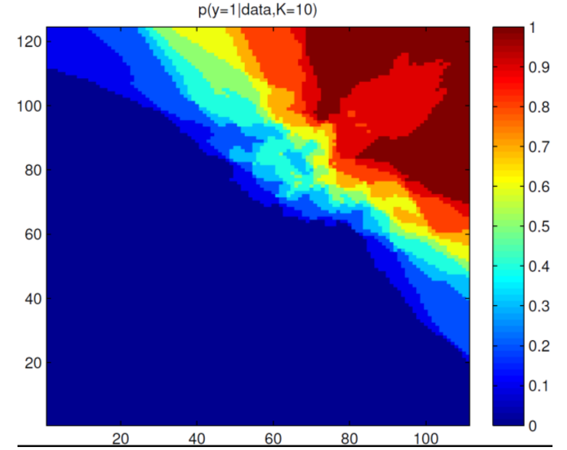
Why classification results are so different for K-NN?



Model types

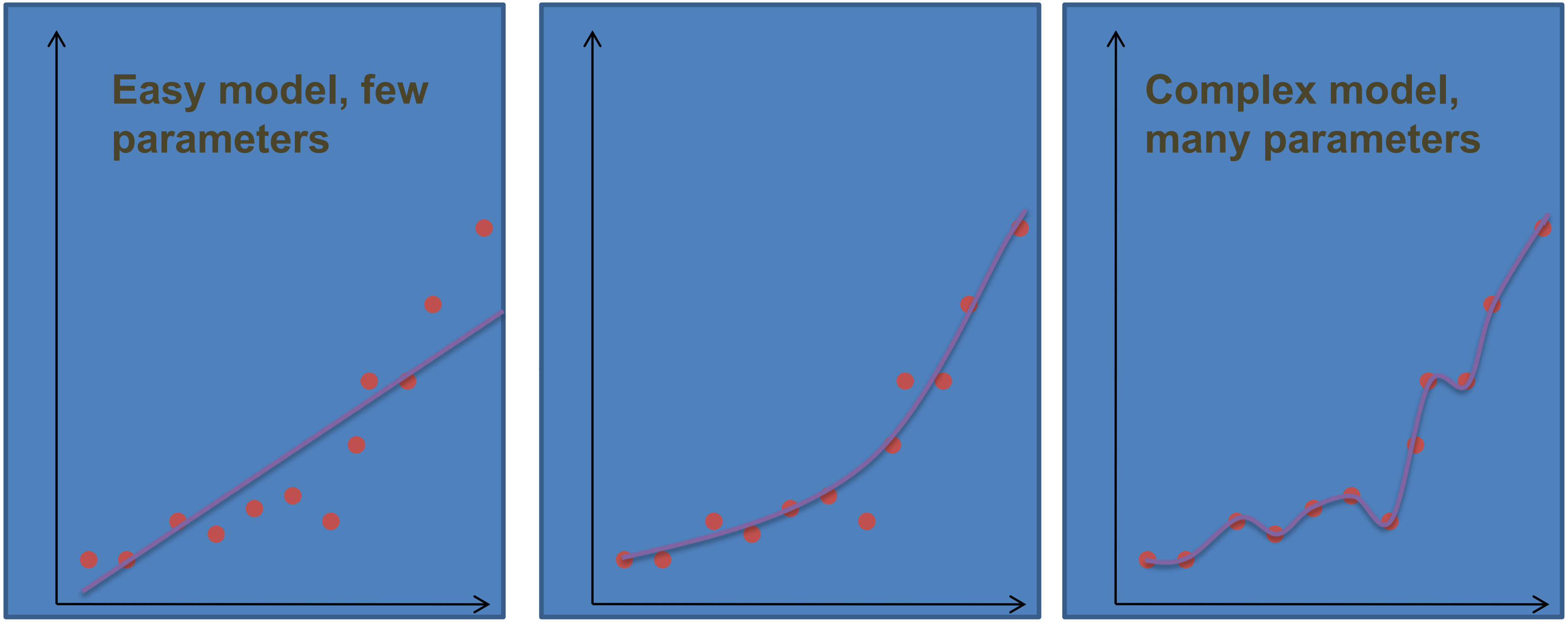
- Parametric models
  - Have certain number of parameters independently of the size of training data
  - Assumption about of the data distribution
  - Ex: logistic regression
- Nonparametric models
  - Number of parameters (complexity) grows with training data
    - Example: K-NN classifier





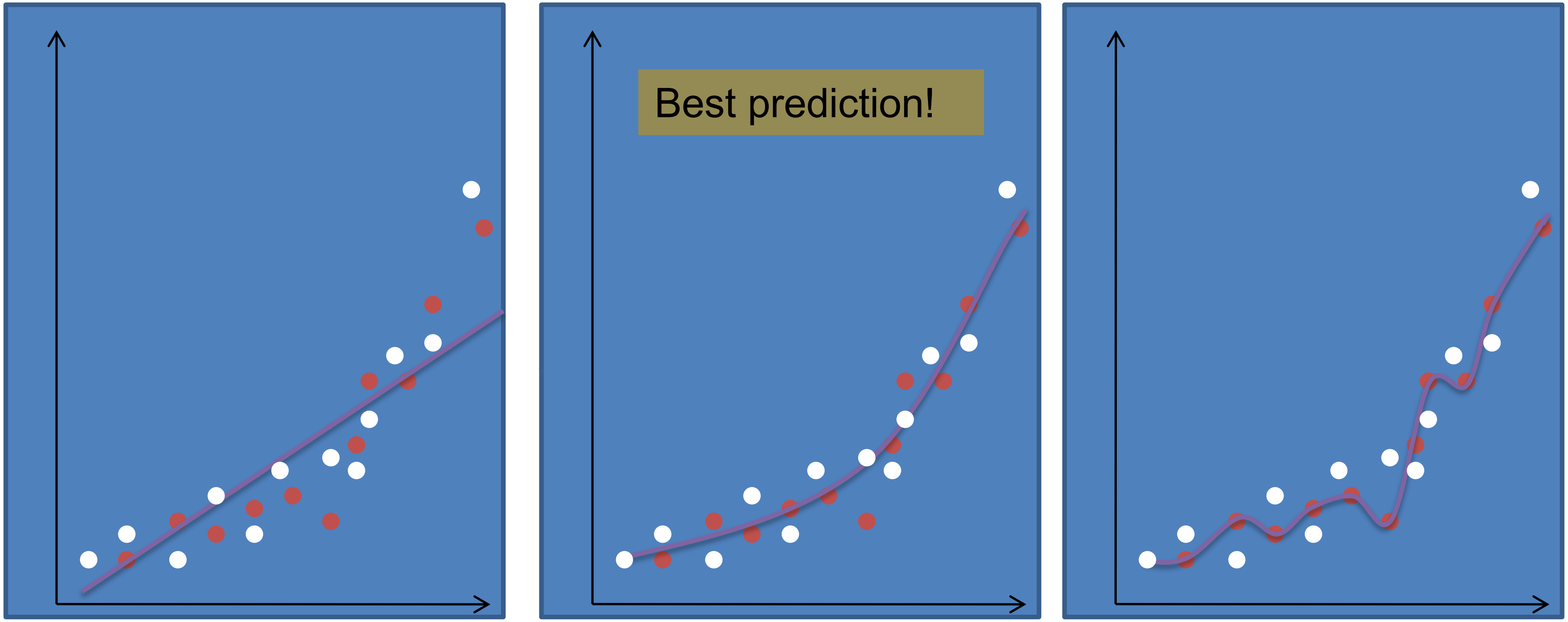
Overfitting

- Which model feels appropriate?



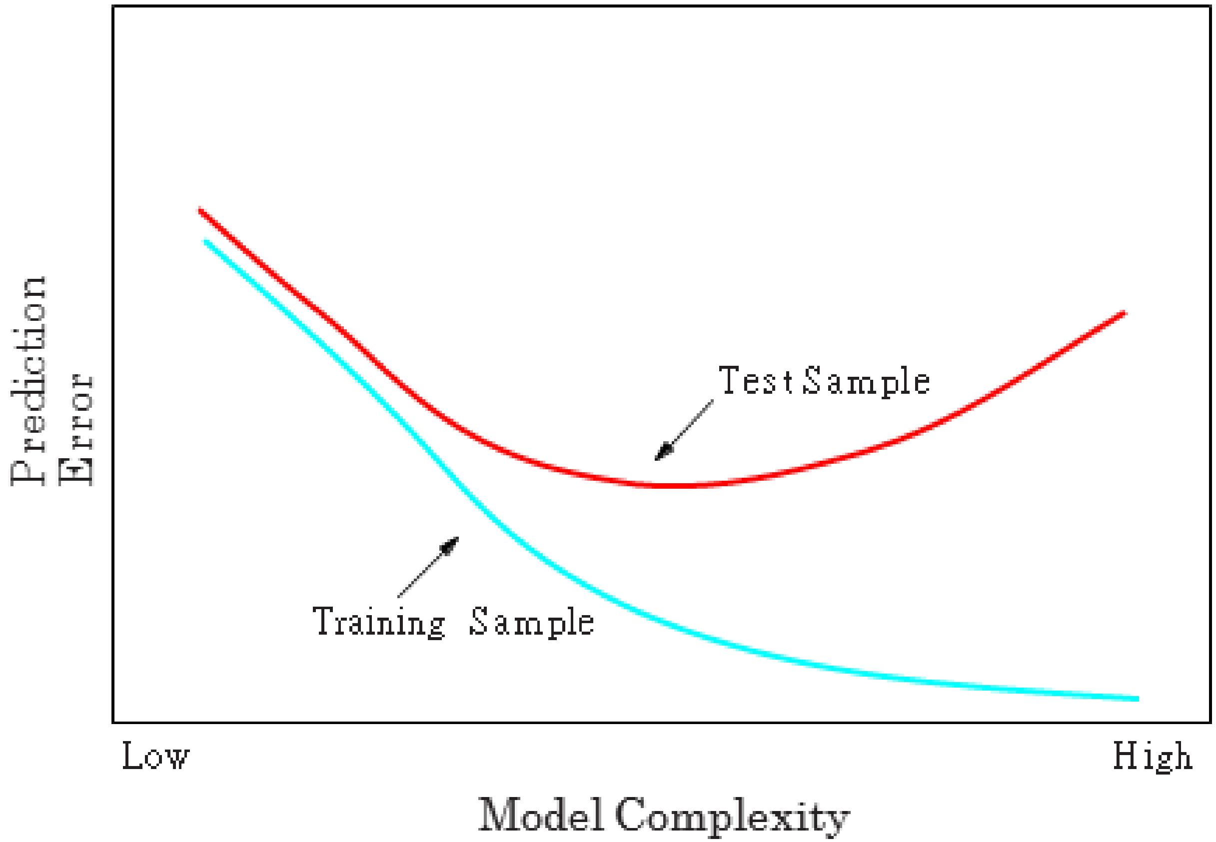
Overfitting

Now new data from the same process



Overfitting

- Observed:



Model selection

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

Training	Test

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

Comment:

- Approach works well for moderate/large data

Typical error functions

- Regression, **MSE** :

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

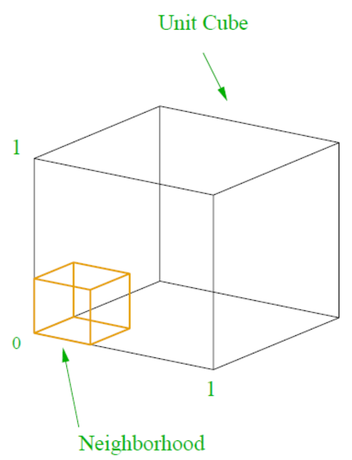
- Classification, **misclassification rate**

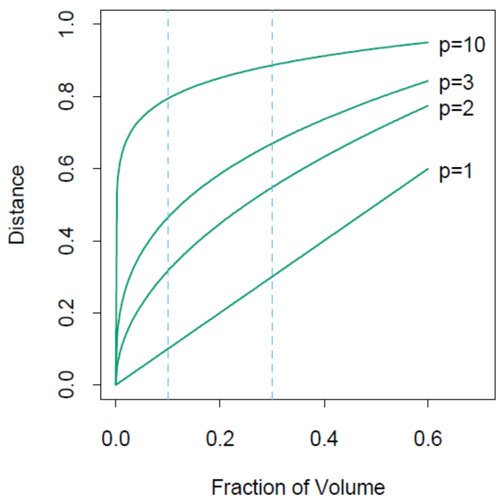
$$R(Y,\hat{Y}) = \frac{1}{N}\sum_{i=1}^{N} I\left(Y_i \neq \hat{Y}_i\right)$$

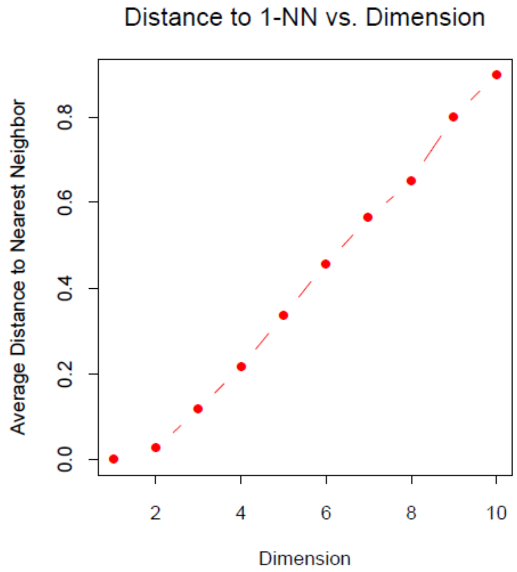
Curse of dimensionality

- Given data D:
  - 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 Xs should lead to small change in Y→interpolation