Neural Networks and Learning Systems
TBMI26 / 732A55
2024

Lecture 1
Introduction

Magnus Borga
Department of Biomedical Engineering (IMT)
magnus.borga@liu.se

Course information

- All information will be available on Lisam

- You must register for classes on Lisam (Signup)
  – Chose group A, B , or C for classes
  – Do not change group!

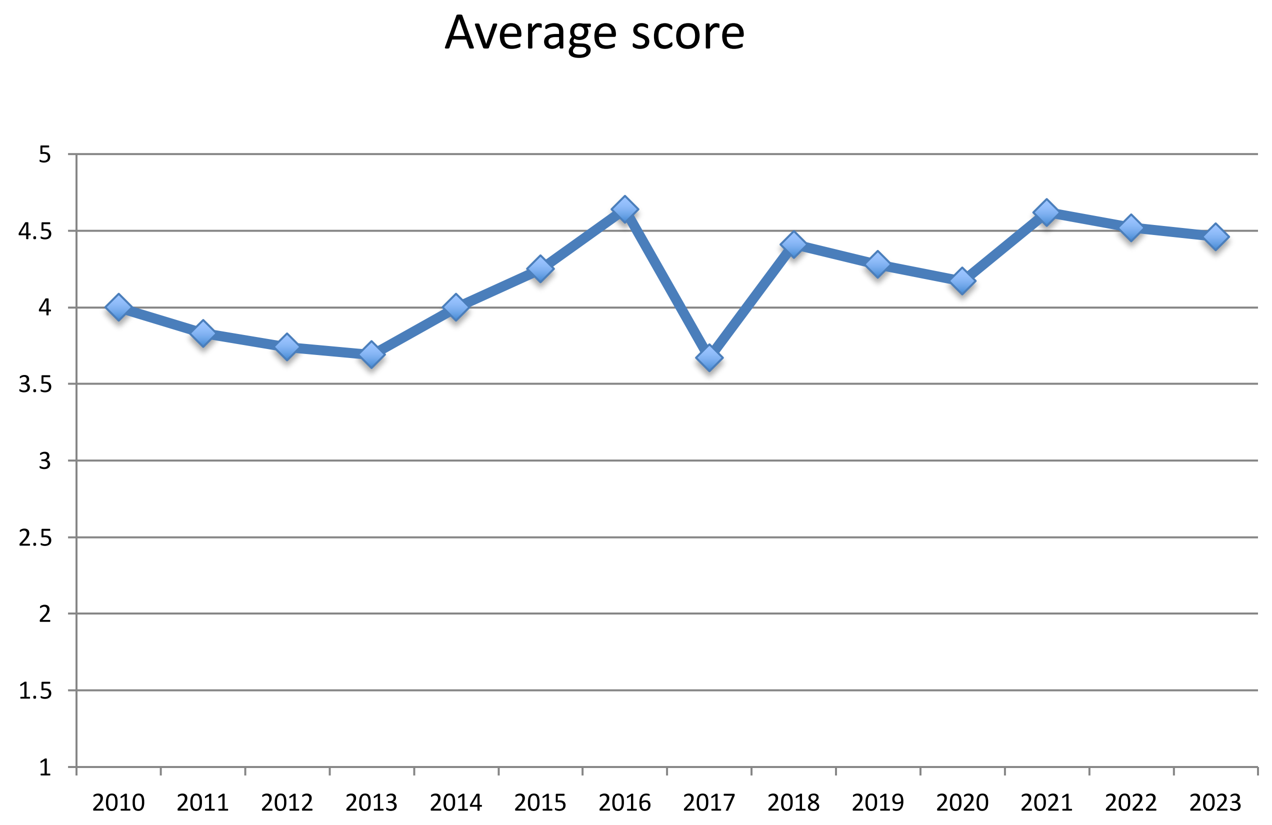
- Laboratory exercises will be done at home with scheduled supervision on Teams
  – Two occasions for each lab. You may choose occasion (Wed or Thu)

Staff

- **Examiner:** Magnus Borga, IMT (magnus.borga@liu.se)
- **Course admin:** Martin Hultman, IMT (martin.o.hultman@liu.se)

- Lectures:
  - Magnus Borga
  - Anders Eklund
- Classes:
  - Martin Hultman
  - Elisabeth Klint
  - Iulian Tampu
- Laboratory exercises:
  - Martin Hultman
  - Elisabeth Klint
  - Iulian Tampu
  - Christoforos Spyretos

Course evaluation and development



2023: 4.46

Changes 2024

Improved instructions regarding lab exercises

The Course - Lectures

PPT lectures, handouts on course page

1. Introduction
2. Supervised learning - Linear classification
3. Neural networks
4. Deep learning
5. Generative Adversarial Networks
6. Ensemble learning & Boosting methods
7. Reinforcement learning
8. Unsupervised learning – Dimensionality reduction, Clustering
9. Kernel methods

The Course - Lessons

- Pen & paper exercises
- Complementary presentations
- Preparations and help with lab assignments
  - In particular lesson 3 is important! – Preparing for the first lab
- Choose group (A,B, or C) on Lisam and follow that group

The Course – Lab assignments

- 4 laboratory exercises/assignments:
  1. Pattern recognition using linear classifiers and neural networks
  2. Deep learning
  3. Face recognition in images using Boosting techniques
  4. Reinforcement learning
- Programming in Python
- Assignments are done in pairs. (Not more than 2 students together!)
- Supervision in S Teams
  - time scheduled (“LA” in schedule)
  - Important to be well prepared <u>before</u> the scheduled supervision!
  - Important to turn on notifications in Teams

The Course – Lab assignments cont.

- Possible oral examination during scheduled supervision <u>if time allows! </u>(The code still needs to be submitted in Lisam.)
- Strongly recommended to hand in the reports during the course, i.e. before the exam in March.
- Late reports may not be corrected until next re-exam (June/August).
- Maximum two re-submissions (3 attempts) for each assignment
- Labs not passed at the August exam will be registered as failed in Ladok and need to be redone next year

Course material

- Lecture notes
- Exercise collection
- Assignments

- Additional links in lecture notes (not required reading)

Prerequisites

- Linear algebra
  - Vectors, scalar products, eigenvectors and eigenvalues
- Multidimensional calculus
  - Gradients, partial derivatives
- Mathematical statistics
  - Mean, variance, covariance, correlation, Gaussian distribution,
- Programming
  - Some programming experience
  - Python experience helps a lot

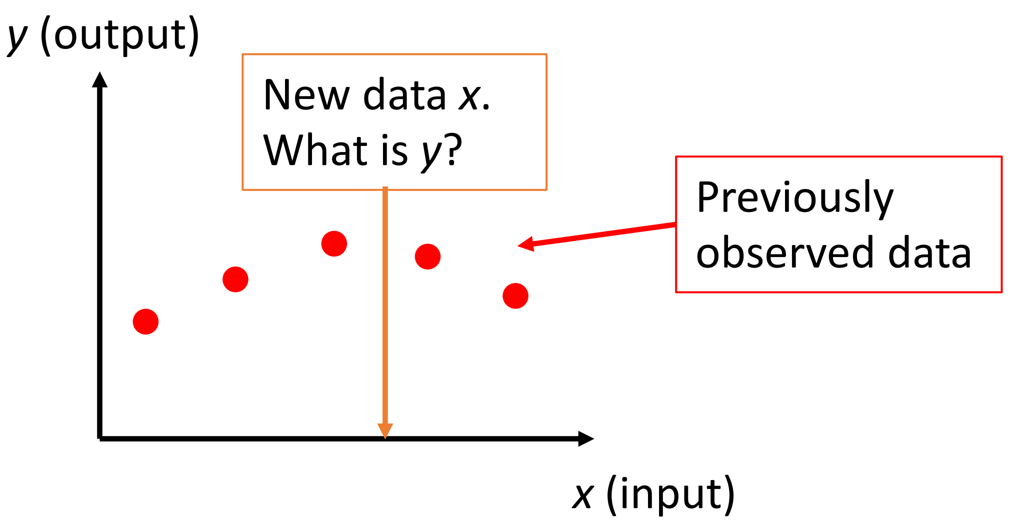
Machine Learning

What is machine learning?

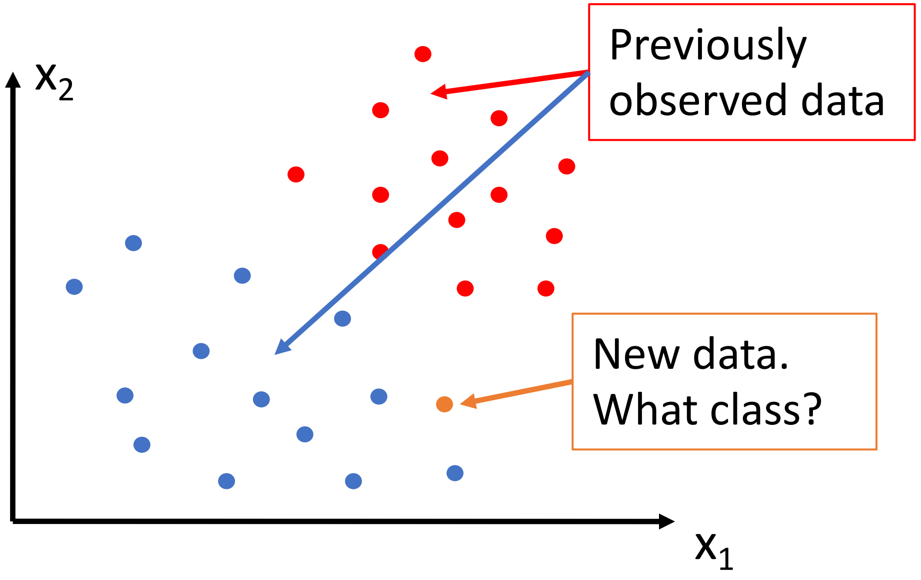
(My) **definition**: The ability of a system to <u>extract relevant information from previous data</u> and <u>generalize to new data</u>

Machine Learning = Data Analysis + Modelling

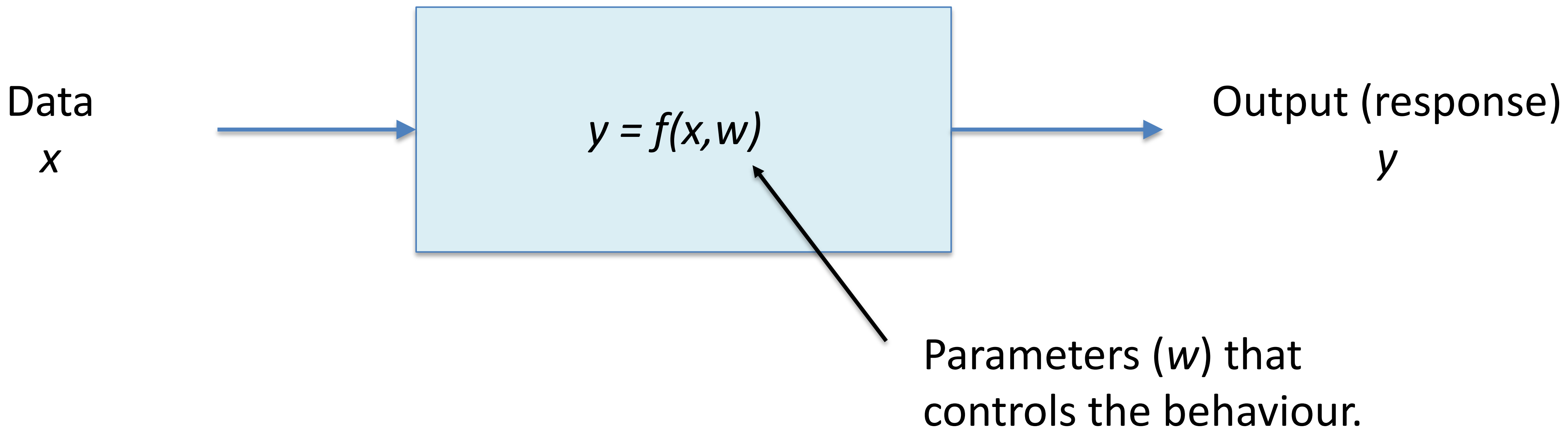
Example: Regression



Example: Classification

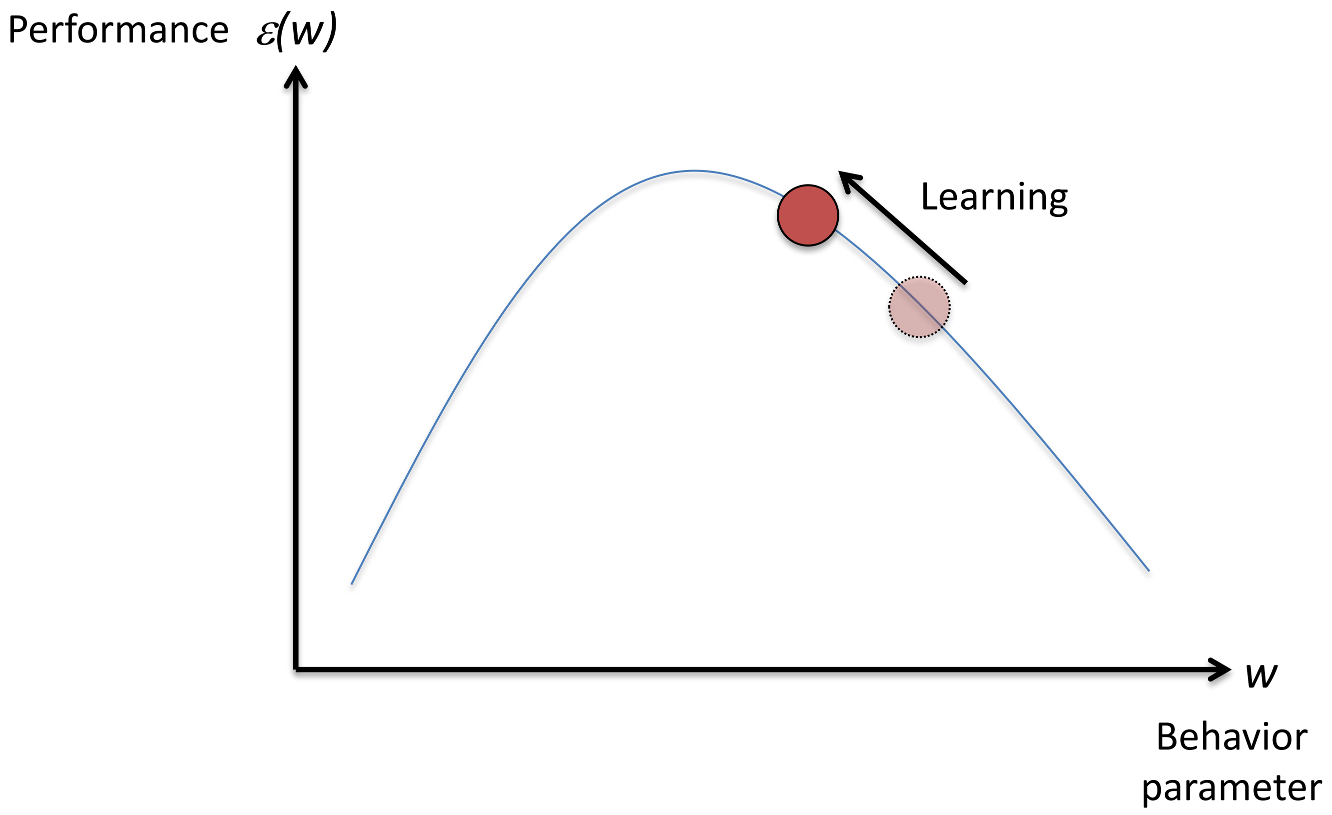


Machine Learning



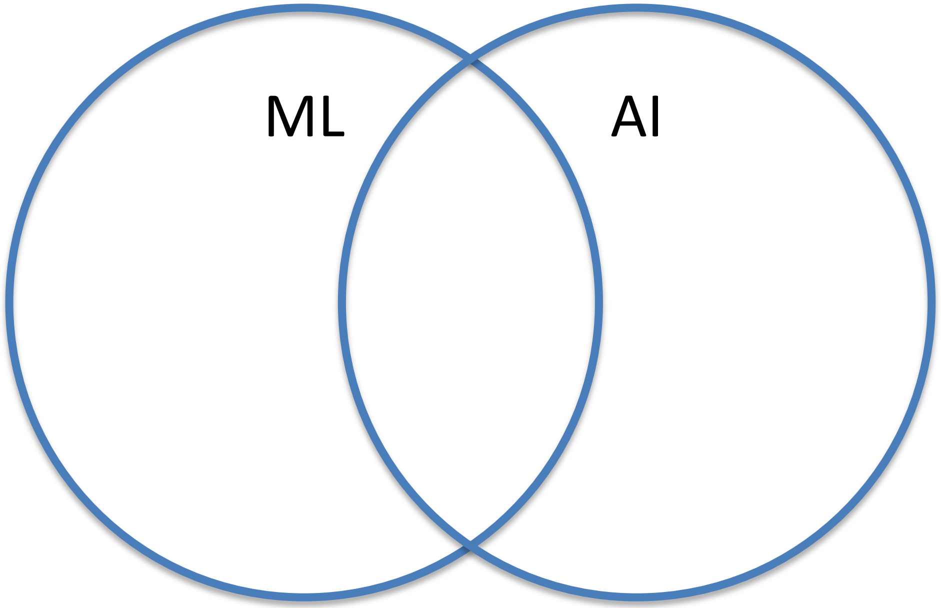
Machine learning: The system adapts the model *f* to some training data to improve its performance on other data

Learning = optimization of performance



What is being learnt depends on how we define performance!

ML vs AI



- AI aims at simulating "intelligent" behaviour, but not necessarily by learning
- ML does not always aim at doing something "intelligent"

Mathematical foundations of machine learning

Machine Learning

Find parameters for optimal behaviour

Optimization theory

Statistics

Model uncertainty

Multidimensional calculus

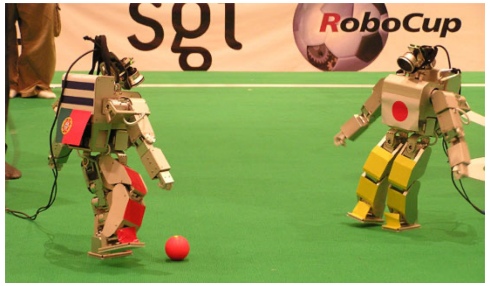
Linear algebra

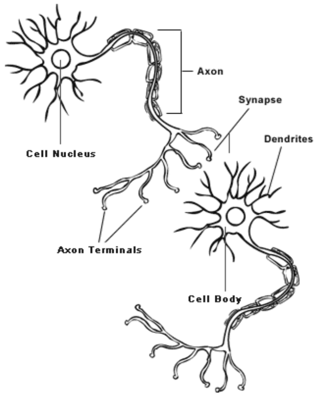
Why machine learning?

- Algorithm too complex for a human to design, but we can easily provide examples of what the algorithm should do.
- Relationships in high-dimensional data too complex for a human to see, but a computer can find these.
- The computer should learn and adapt continuously to new situations.

Applications of machine learning

- Pattern and speech recognition
- Robots & autonomous systems
- Big data
- Expert systems & decision support
- Games
- Models of the brain





Three main categories of machine learning methods

- **Supervised learning (predictive)**
  Learn to generalize and classify new data based on labelled training data.
  - Pattern recognition
  - Classification
  - Regression

- **Unsupervised learning (descriptive**)
  Discover structure and relationships in complex high-dimensional data.
  – Clustering
  – Dimensionality reduction

- **Reinforcement learning (active)**
  Generate policies/strategies that lead to a (possibly delayed) reward. Learning by doing.
  – Sequence optimization

Classification vs. regression vs. ranking

- **Classification**: Select one of a discrete set of classes (the output set *W* is discrete).
  - Which horse is going to win this race?
  - Which letter does this image depict?
  - Is this email spam (yes/no)?

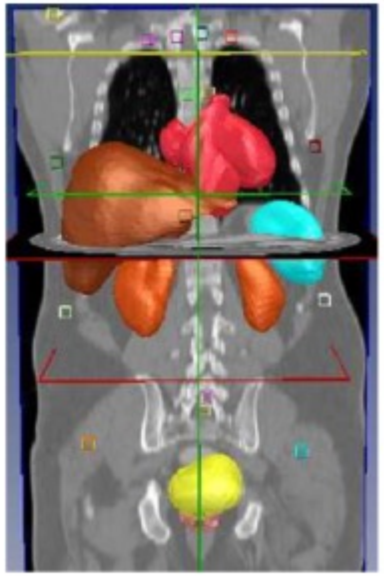
- **Regression**: Learn to predict a continuous value ($W = \mathbb{R}$).
  - Learn to predict the temperature tomorrow.
  - What is the *probability* that this image depicts the letter 'a'?

- **Ranking**: Learn to rank a set of items ($W = \mathbb{R}$).
  - Rank webpages, movies, etc.

Pattern recognition examples

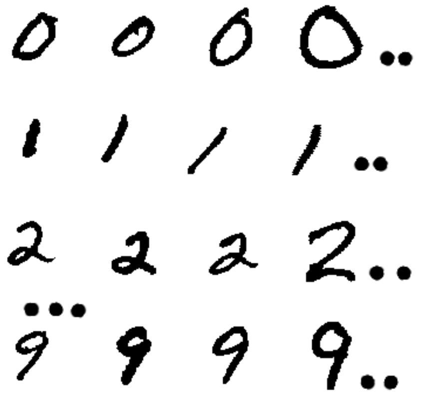
Face recognition

Organ segmentation

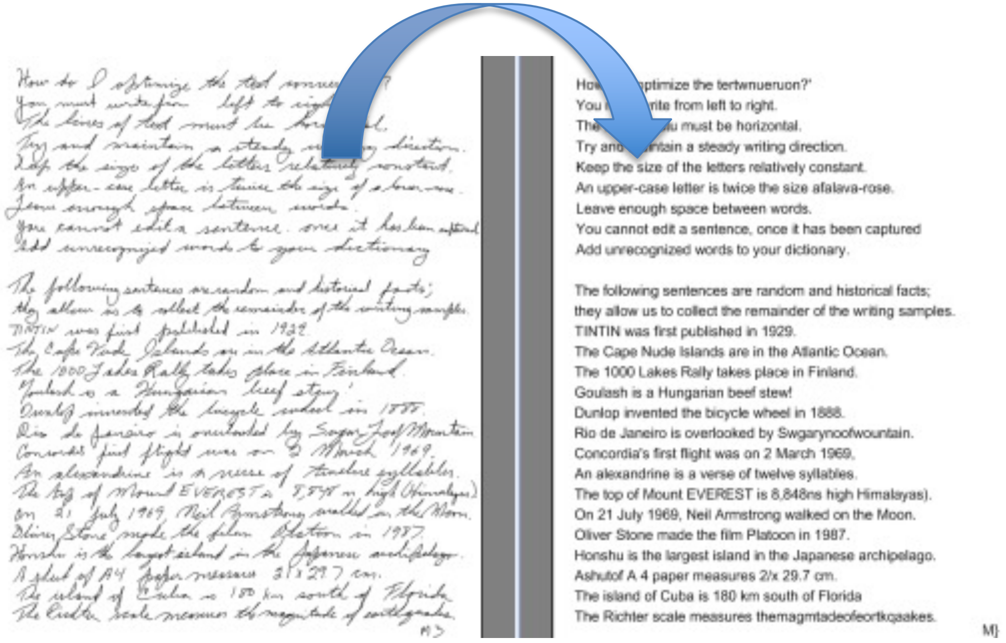


Pattern recognition examples

Optical Character Recognition (OCR)

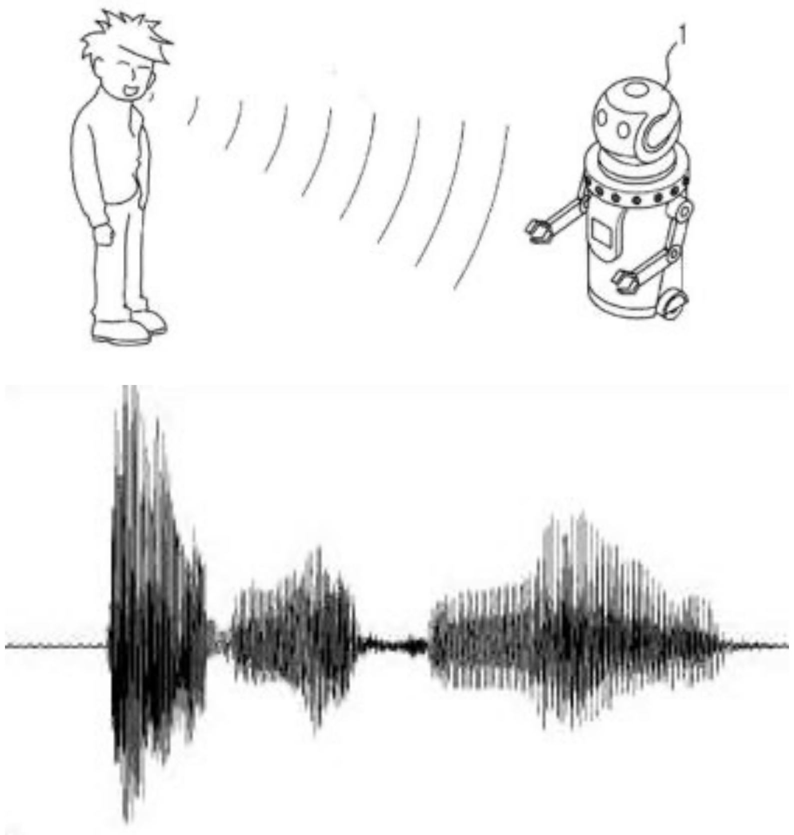


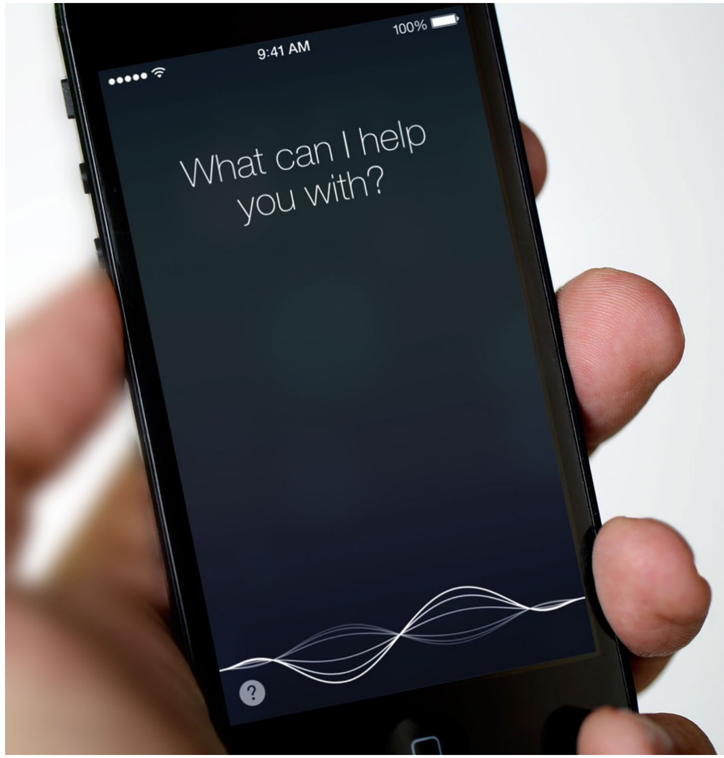
Source: http://blog.damiles.com



Pattern recognition examples

Speech recognition





Pattern recognition examples

Game positions

Spam filters

#1
100% satisfied
4U
Accept credit cards
Act Now!
Additional Income
Affordable
All natural

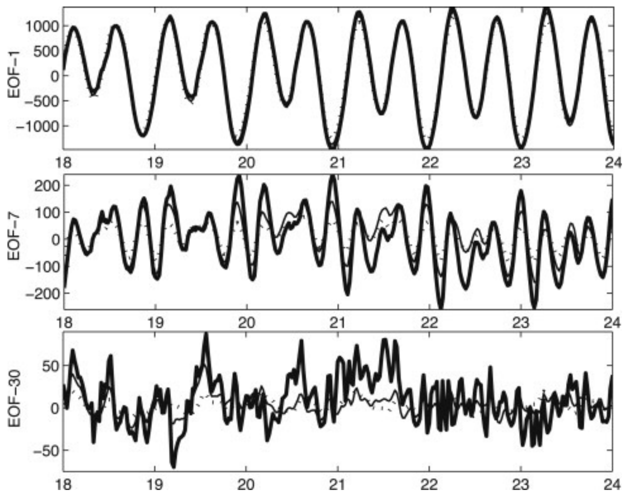
Movie & music recommendation

Spotify

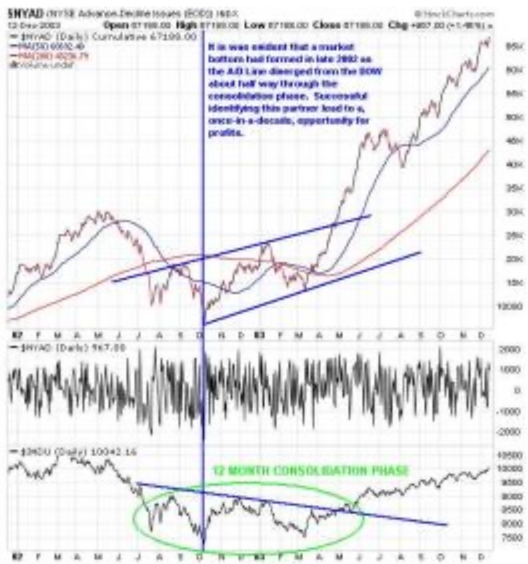
Regression examples

Prediction and forecasting

Weather and natural phenomena



Financial markets



Features

- A <u>feature</u> is a measurement or scalar number that describes some aspect of a phenomenon or object
  - Size, length, shape, velocity
  - Intensity and color (RGB)
  - Position (x,y)
  - Signal frequency
  - Sensor measurements (e.g., temperature)
  - Game piece present at certain location (yes/no)
  - Word present in an email (yes/no)

- <u>Feature extraction </u>is the process of measuring features from data.

Features – Iris dataset

Iris setosa

Iris versicolor

Iris virginica

Fisher's Iris Data

Sepal Length	Sepal Width	Petal Length	Petal Width	Species
5.7	4.4	1.5	0.4	*I. setosa*
5.8	2.6	4.0	1.2	*I. versicolor*
5.8	2.7	5.1	1.9	*I. virginica*

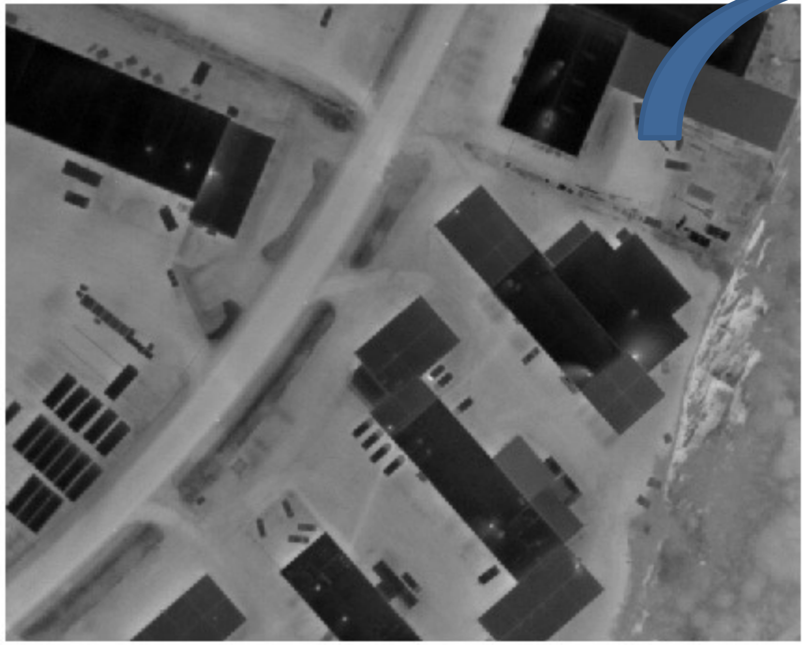
From Wikipedia

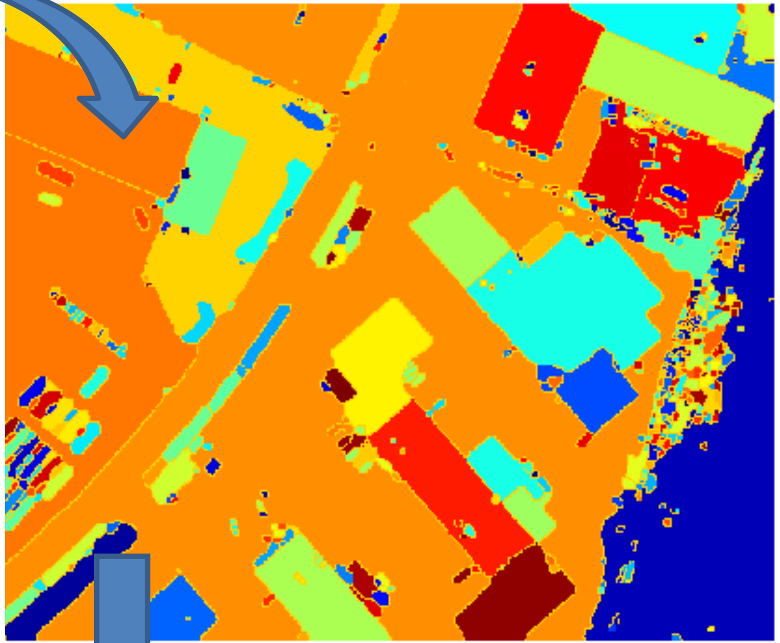
Feature vectors:

$$\mathbf{x}_1 = \begin{bmatrix} 5.7 \\ 4.4 \\ 1.5 \\ 0.4 \end{bmatrix} \quad \mathbf{x}_2 = \begin{bmatrix} 5.8 \\ 2.6 \\ 4.0 \\ 1.2 \end{bmatrix} \quad \mathbf{x}_3 = \begin{bmatrix} 5.8 \\ 2.7 \\ 5.1 \\ 1.9 \end{bmatrix}$$

Features – Image classification

Watershed transform

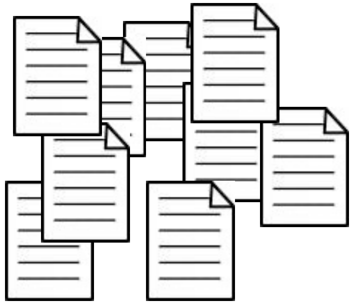




Extract intensity & shape features

$$\mathbf{x} = \begin{bmatrix} \text{Mean intensity} \\ \text{Std intensity} \\ \text{Border regularity} \\ \text{Compactness} \end{bmatrix}$$

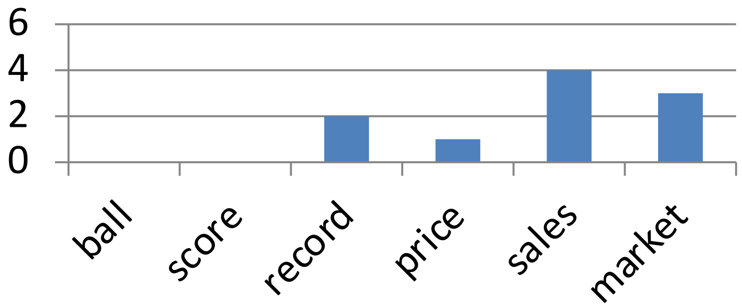
Features – Document analysis



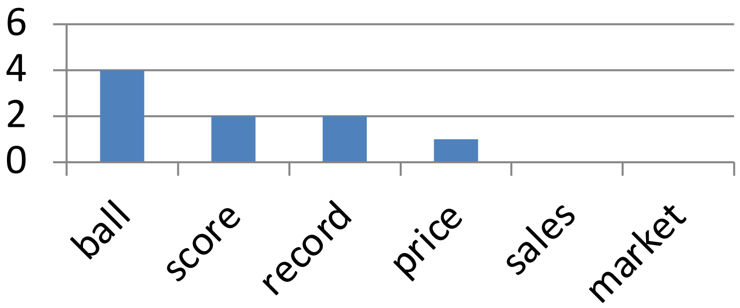
Articles, mail, web pages, ...

Dictionary:
{'ball', 'score', 'record', 'price', 'sales', 'market'}

Financial document



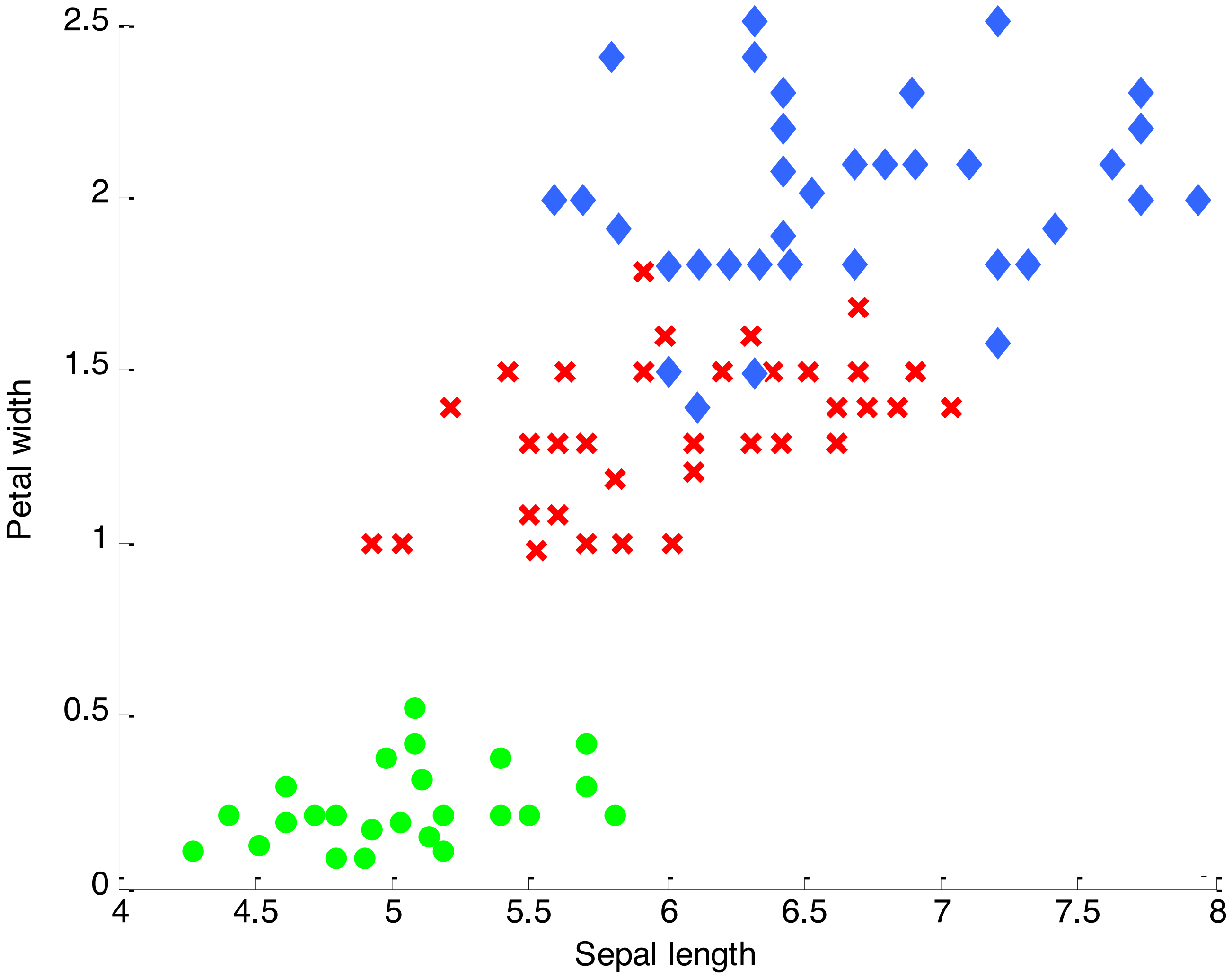
Sports document



$$\mathbf{x} = \begin{bmatrix} \#'ball' \\ \#'score' \\ \#'record' \\ \#'price' \\ \#'sales' \\ \#'market' \end{bmatrix}$$

Bag-of-words model

Feature space



Iris setosa

Iris versicolor

Iris virginica

Knowledge representation

Two alternative ways to represent the gained knowledge:

- Instance-based learning
  - Store all training examples
  - Compare new data with the stored examples

  \+ Fast and easy to train (just need to store the data)

  \+ Transparent (possible to explain the result)

  \- Expensive to store a lot of data

  \- Computational cost to produce results

- Parameterized models
  - Use training data to fit a parameterized model
  - Use the model to predict new data

  \+ Efficient w.r.t. memory (only need to store the parameters)

  \+ Fast to compute the output

  \- Less transparent (difficult to explain the result)

  \- Require a lot of training data and time

Supervised Learning

Supervised learning

- **Task:** Learn to predict/classify new data from labeled examples.
- **Input:** Training data examples $\{\boldsymbol{x}_i, y_i\}$, i=1...K, where $\mathbf{x}_i$ is a feature vector and y_i is an output value in the set W.
- **Output:** A function $f(\boldsymbol{x};w_1,\ldots,w_N) \rightarrow \mathrm{W}$

Find a function f so that new feature vectors are classified correctly. ***Generalization!!***

k-Nearest Neighbours (k-NN)

An example of instance-based learning

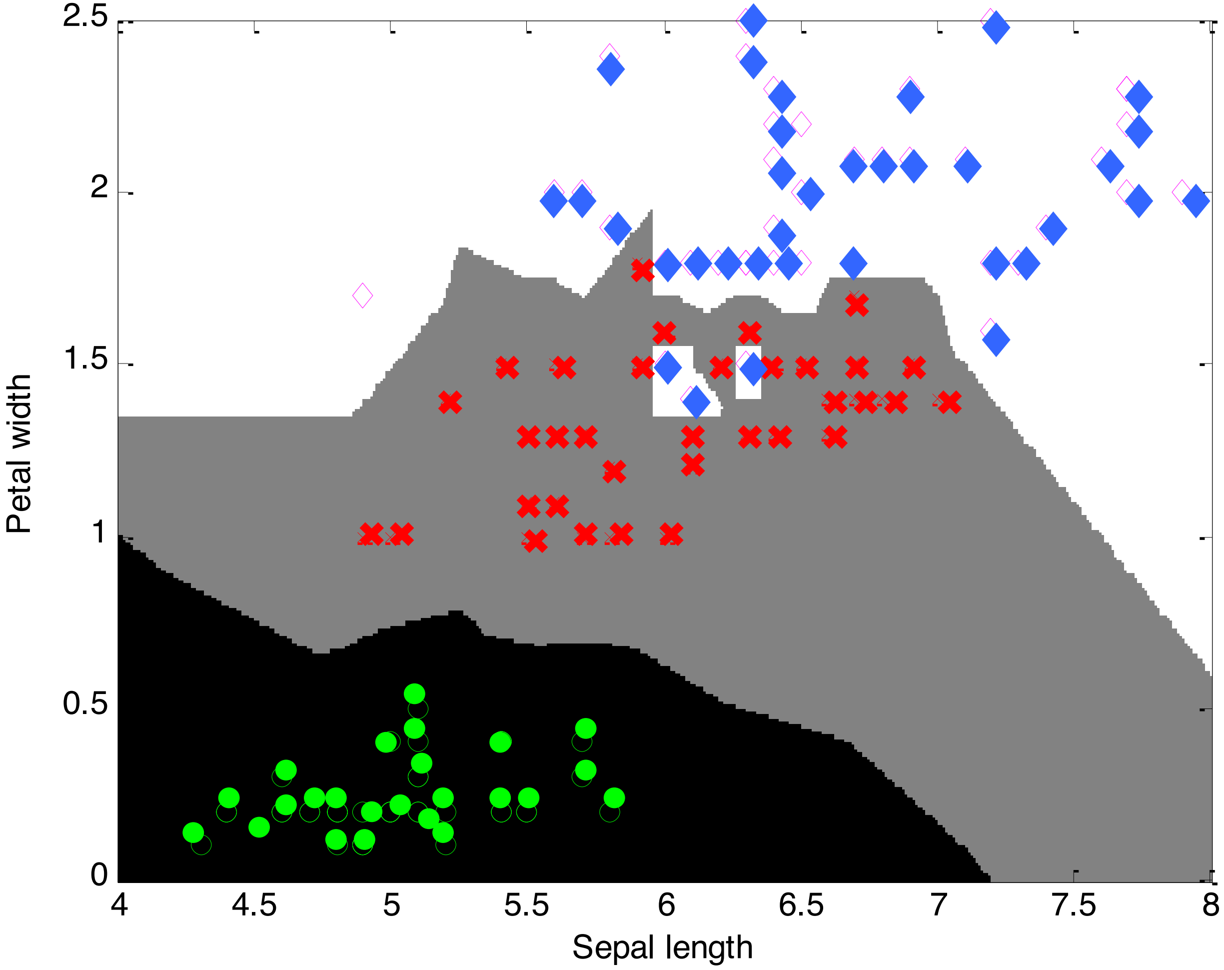
- Save all training data.
- For a new case, find similar examples among the training data.
- Requires a similarity measure (metric), for example the Euclidian distance

$$\|\mathbf{x}-\mathbf{y}\| = \sqrt{\sum_i \left(x_i - y_i\right)^2}$$

- A majority vote among the *k* nearest neighbours decides the class, where *k* can be 1,2,3,4...

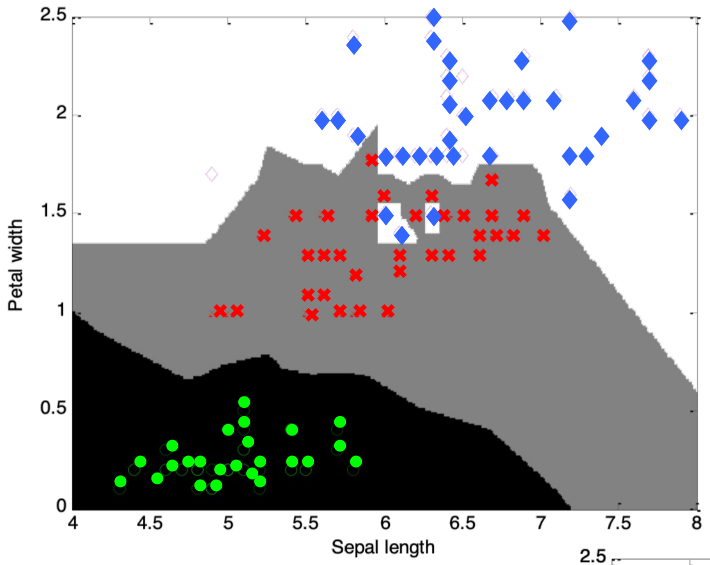
Classification boundaries

$k = 1$

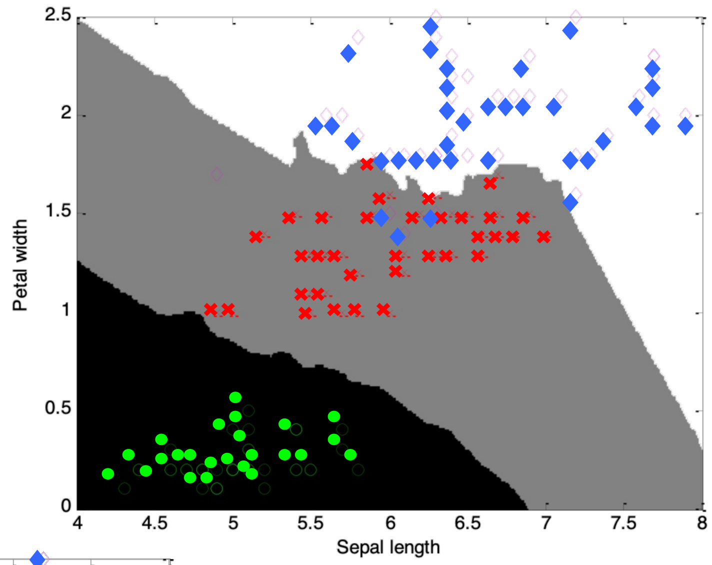


Classification boundaries

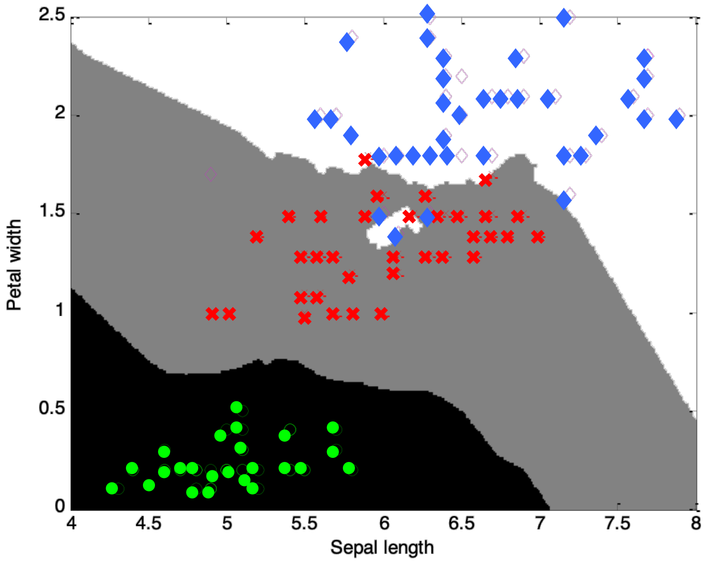
$k = 1$



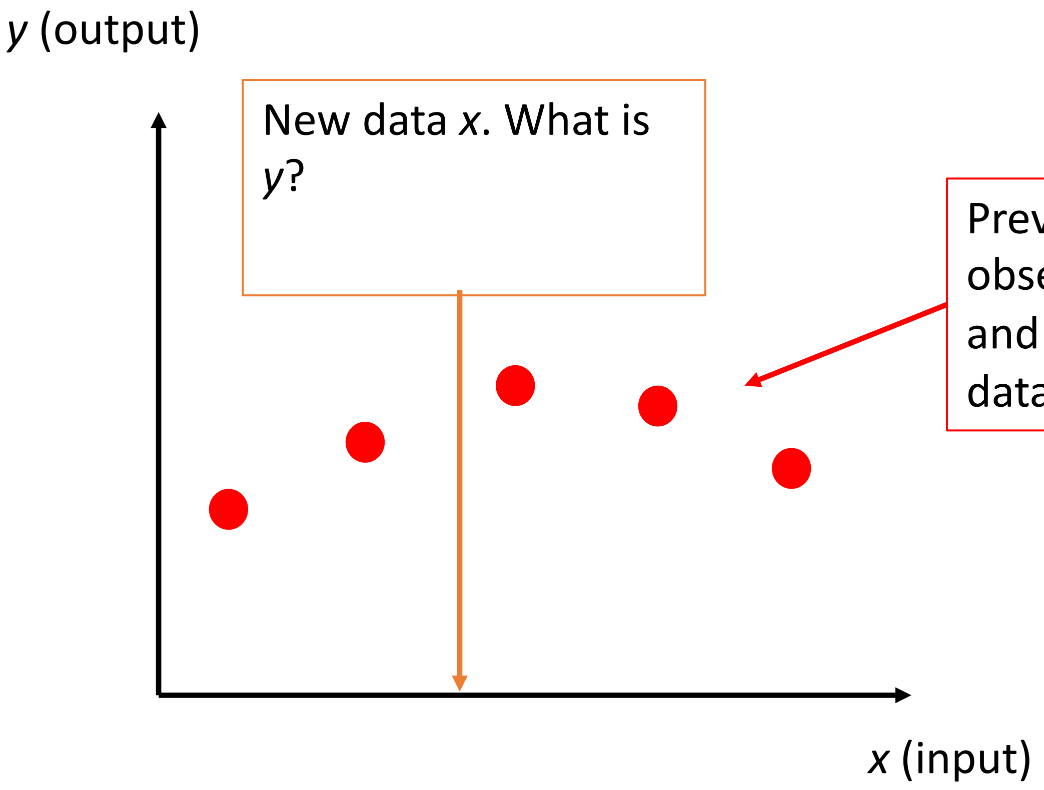
$k = 7$



$k = 3$



k-NN for regression



Local regression model (instead of a global model)

Examples:

- $k = 1$
  - Nearest neighbour interpolation
- $k = 2$
  - Use linear interpolation between the two closest points

Pros and cons of k-NN

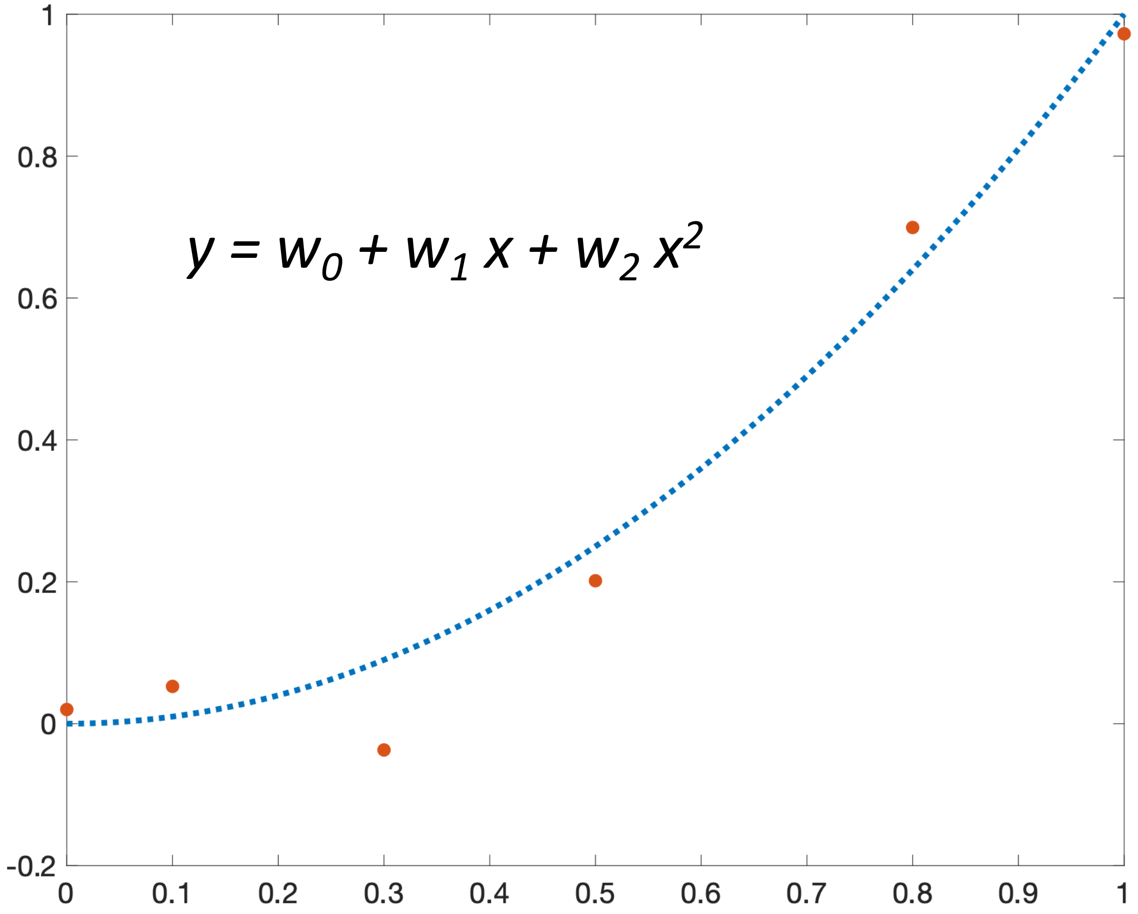
+ Very simple – only one parameter (*k*)
+ Simple to add new data
+ Very fast to "train"

- Must store all training data – expensive for large data sets
- Computationally heavy for large data sets – must compare new samples with all stored samples.

Parametric models

- Instead of storing all training data, a parametric model can be fitted
- Only need to store the parameters
- Very fast to compute the output

- May take long time to train the model



Discriminant functions

- Classification can be seen as a function that takes one value e.g. +1, for one class and another value, e.g. -1, for the other class
- Such function is called a **discriminant function**
- This means that training a classifier becomes a function approximation!

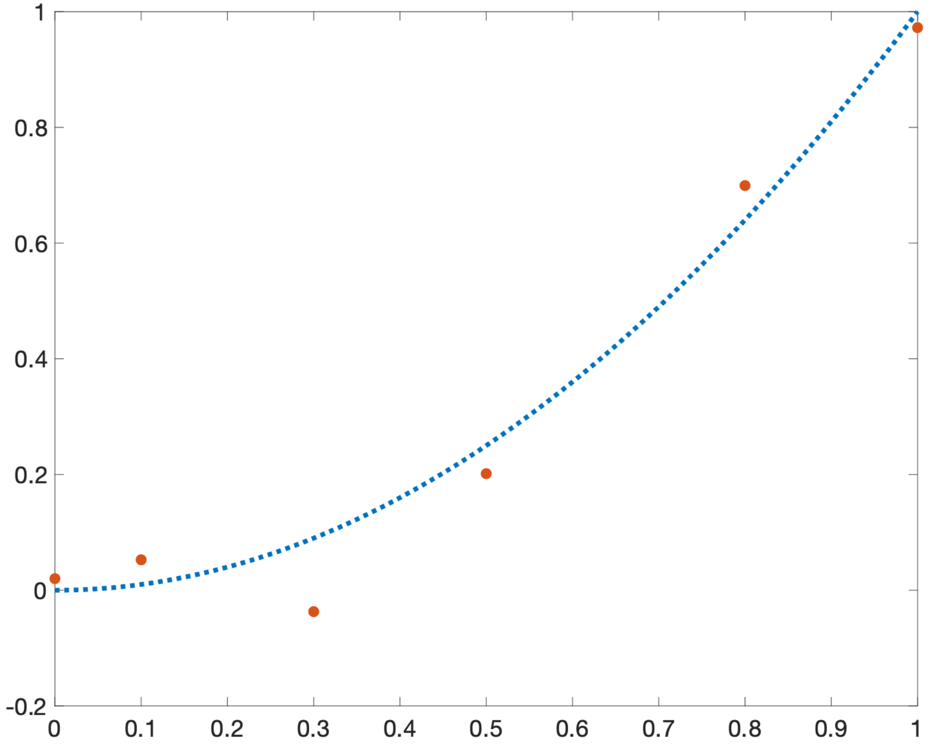
Generalization is the key

- The important question is not how well a model can fit to existing data, but how well it can predict future data!
- Any (finite) data set can be perfectly fitted by a model complex enough.
- Do we need a perfect fit?
- Do we even *want* a perfect fit?!

Overfitting

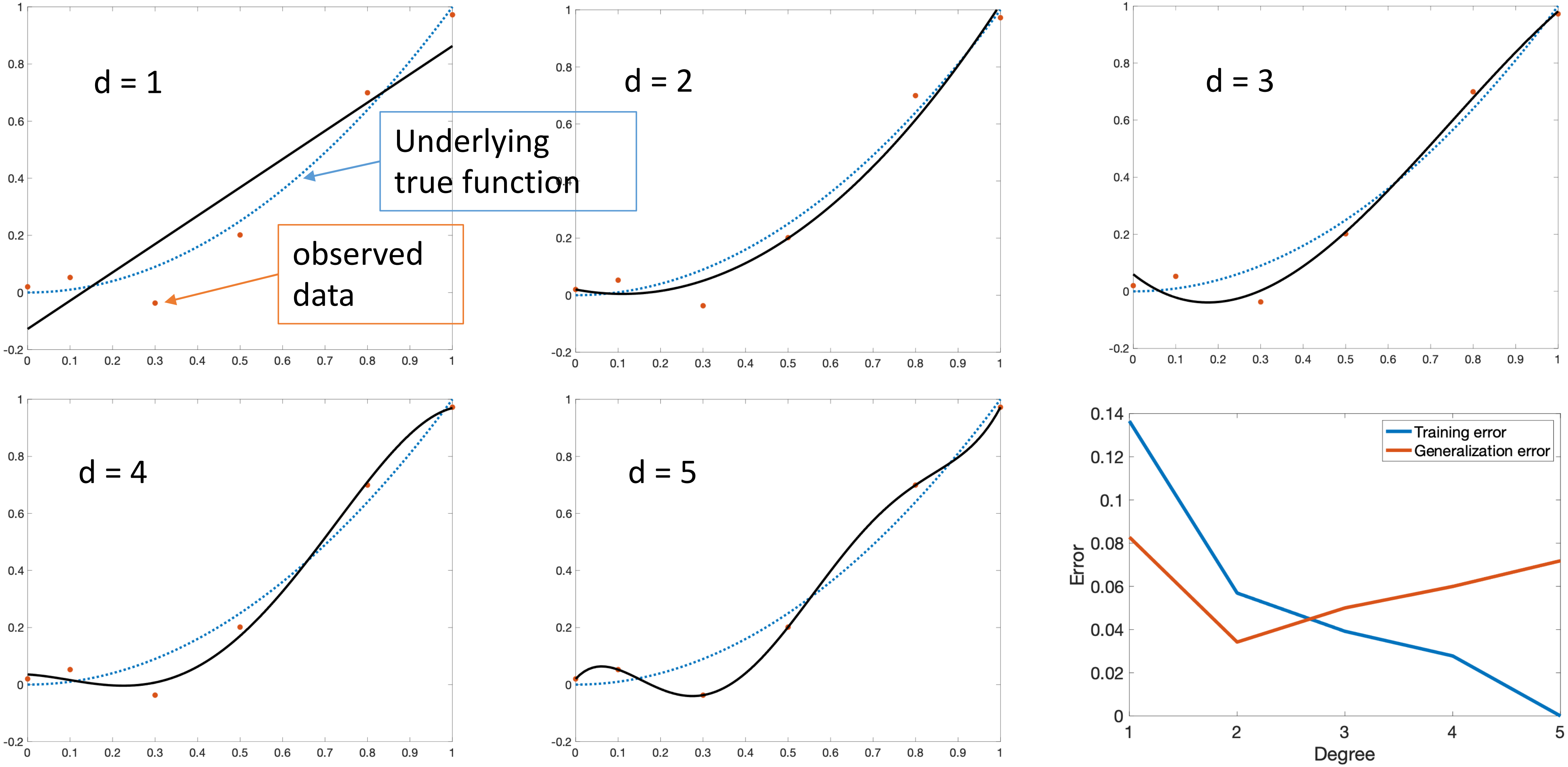
Example:

Assume we want to learn a relationship between *x* and *y* that in reality is quadratic. But we don't know that, we only have these sample points, which (as always) contain some measurement errors or "noise":



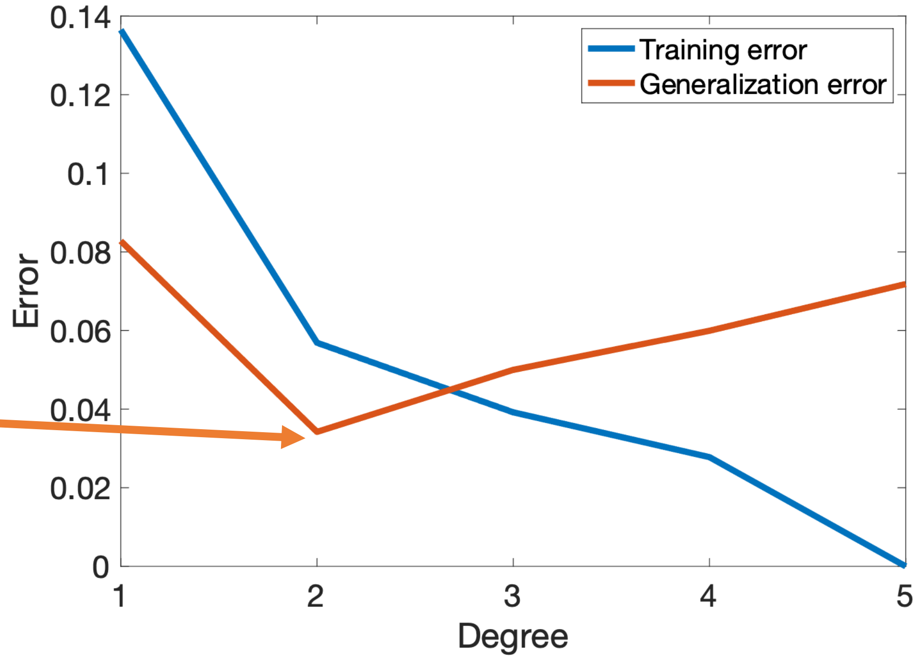
Overfitting

This is how polynomials of different degrees would fit the data:



Generalization error

- The best fit to the data is clearly not the model that is closest to the underlying true relationship!
- We want to find the model that **minimizes the generalization error,** not the error on the training data!
- Hence, we need to train our model on one part of the data and validate the model on another part that has not been used for training.

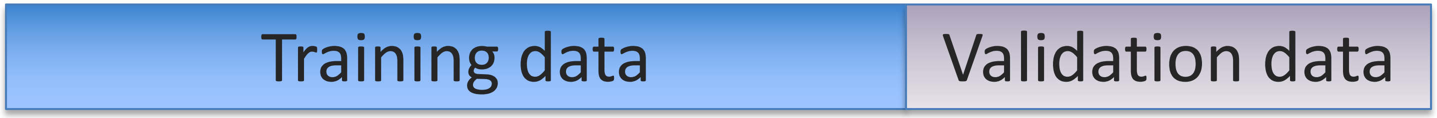


Validation data

- Three ways to perform the validation:
  - Hold out
  - Cross validation
  - Leave one out

Hold out

- Simplest approach, hold out one part of the entire data set as validation data.



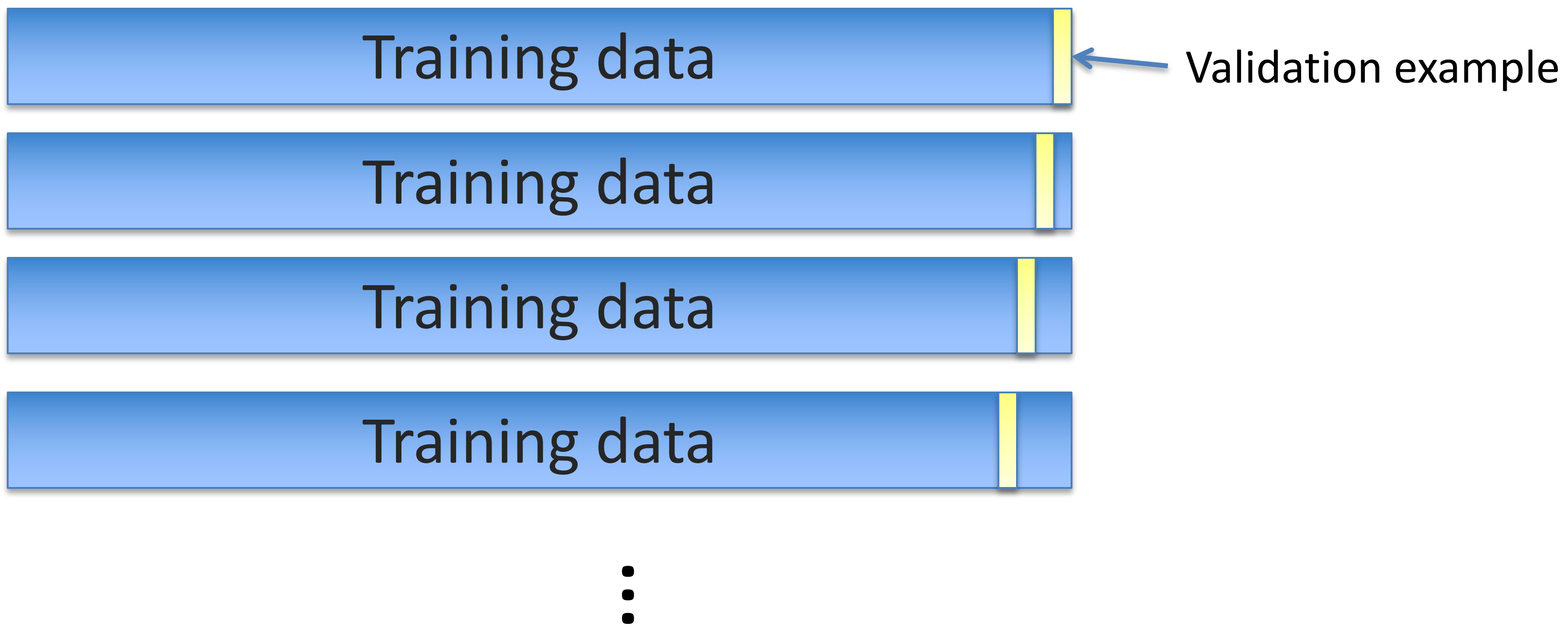
n-fold Cross-Validation

- Divide data set into n segments. Train using n-1 segments and validate using the n:th.
- Example of 3-fold Cross-Validation:

Training data	Training data	Validation data
Training data	Validation data	Training data
Validation data	Training data	Training data

Leave-one-out

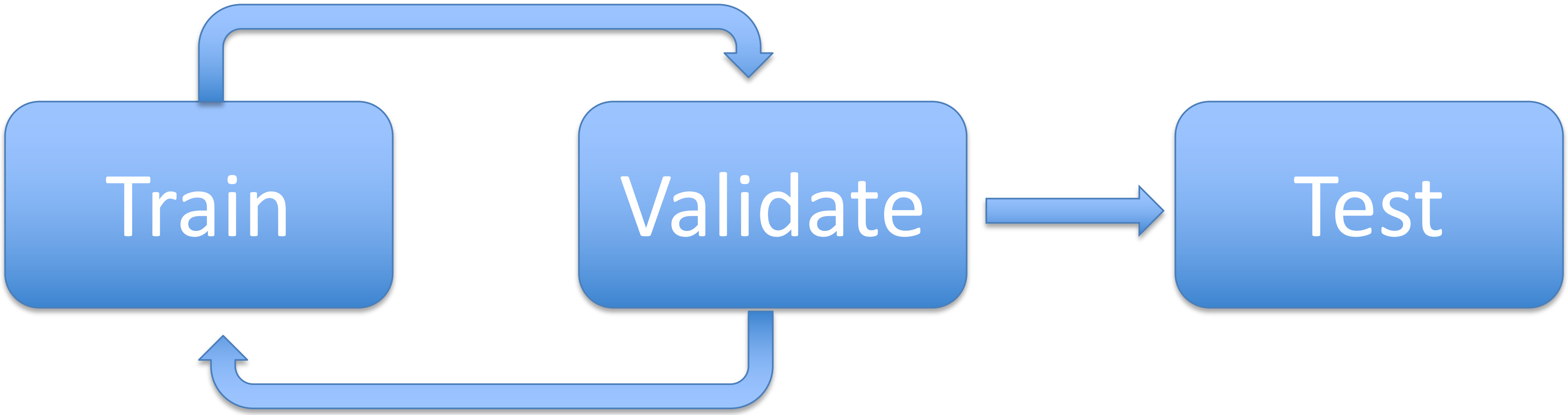
- Extreme case of Cross-Validation: Use all data but one example for training and use the last one to evaluate



How can we find the minimum generalization error?

- What happens if the generalization error is not low enough?
- Modify the classifier (change the model) and train again…
- But – then the validation data is used to select the model!
- How do we know how well the new model generalizes?
- Need new validation data to test the final model – this dataset is called test data.
- Test data must <u>never</u> be used more than once!

Training – Validation –Testing



Evaluating classifiers – The Confusion matrix

Iris setosa

Iris versicolor

Iris virginica

Predicted class (columns); Actual class (rows)

	Setosa	Versicol.	Virginica
Setosa	50	0	0
Versicol.	0	45	5
Virginica	0	7	43

Accuracy: $$\frac{50+45+43}{150}=92\%$$