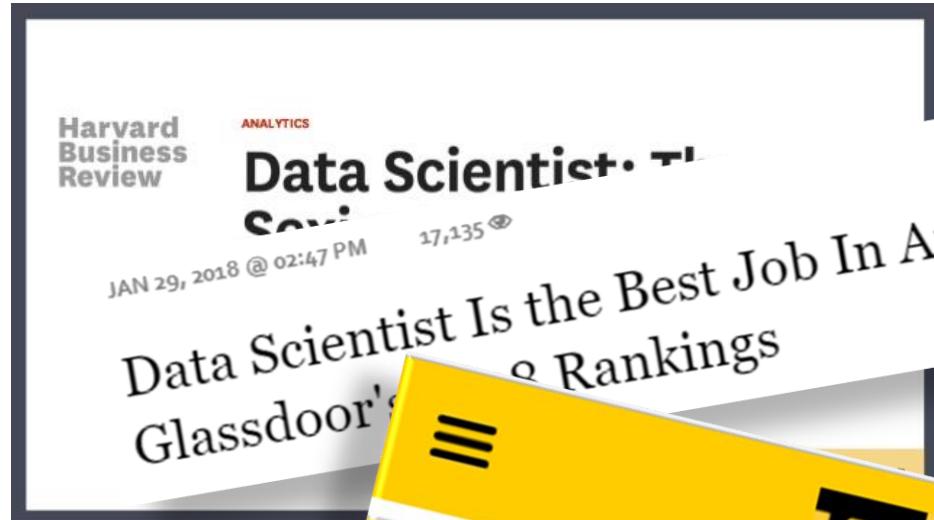


Worth learning data science



GLASSDOOR'S BEST JOBS IN AMERICA 2017

1. Data Scientist

5. Analytics Manager

6. HR Manager

7. Database Administrator

Strategy Manager

Designer

Solutions Architect

SOURCE: GLASSDOOR 50 BEST JOBS IN AMERICA

 CAREER SHERPA

50 Best Jobs in America for 2022

[Best Places to Work](#)[Top CEOs](#)[Best Jobs](#)[Best Cities for Jobs](#)[Highest Paying Jobs](#)[!\[\]\(cbe2492b119e39e02a1dab2af4a4b296_img.jpg\) Share](#)

2022 ▾

United States ▾

	Job Title	Median Base Salary	Job Satisfaction	Job Openings	
#1	Enterprise Architect	\$144,997	4.1/5	14,021	View Jobs
#2	Full Stack Engineer	\$101,794	4.3/5	11,252	View Jobs
#3	Data Scientist	\$120,000	4.1/5	10,071	View Jobs
#4	Devops Engineer	\$120,095	4.2/5	8,548	View Jobs
#5	Strategy Manager	\$140,000	4.2/5	6,977	View Jobs
#6	Machine Learning Engineer	\$130,489	4.3/5	6,801	View Jobs
#7	Data Engineer	\$113,960	4.0/5	11,821	View Jobs
#8	Software Engineer	\$116,638	3.9/5	64,155	View Jobs
#9	Java Developer	\$107,099	4.1/5	10,201	View Jobs
#10	Product Manager	\$125,317	4.0/5	17,725	View Jobs

[Advisor > Education](#)

What Are The Fastest-Growing Jobs Of 2023?

By Cecilia Seiter
Contributor

Forbes ADVISOR

Advertiser Disclosure



Reviewed By
Editor

Veronica Freeman

Published: Feb 1, 2023, 10:30am



Fastest-Growing Tech Careers

Data Scientists

Growth Rate (2021-31): +36%

~~Median Pay: \$100,910 per year~~

Education Requirements: Bachelor's degree

Career Overview: Data scientists extract insights and knowledge from large, complex data sets. They leverage that data to make intelligent, informed decisions to help organizations improve their performance and achieve their goals.

Conducting surveys or scraping the web to collect data is a key component of a data scientist's job. From there, data scientists clean and classify raw data, using machine learning and data visualization software to demonstrate their findings. It's paramount that data scientists know how to communicate their findings effectively and in a way that's accessible to a general audience.

Information Security Analysts

Growth Rate (2021-31): +35%

~~Median Pay: \$102,600 per year~~

Education Requirements: Bachelor's degree in cybersecurity or a related field

Career Overview: Information security analysts are responsible for ensuring the safety and security of an organization's sensitive information and computer systems. They rigorously monitor networks for security breaches and investigate any attacks that may occur.

Information security analysts use software like firewalls and data encryption programs to safeguard sensitive assets. They are also responsible for documenting metrics and reporting attempted attacks. Information security analysts recommend security enhancements to management or senior IT staff, and they help other employees gain their footing with new security products and procedures.

Cybersecurity analysts are a type of information security analyst. For more information, check out our guides on [information security vs. cybersecurity](#) and [how to become a cybersecurity analyst](#).

Web Developers

Growth Rate (2021-31): +30%

~~Median Pay: \$77,030 per year~~

Education Requirements: Bachelor's degree

Career Overview: What is web development? Web development is a multidisciplinary field that involves a combination of technical and creative skills.

Web developers build websites that align with a client's vision and business goals. These professionals must understand how to write code using programming languages such as HTML or XML. They also create and test website applications, interfaces and navigation menus, as well as collaborate with designers to determine a website's layout and functionality.

Some web developers build the entire site; others specialize in building out particular components. For example, back-end web developers create the basic architecture of a site. Front-end developers are responsible for a site's layout and visual features. For more information, see our guide on [how to become a web developer](#).

Basic course information

- Credits: 4
- Contact hours: TU 4-6 (pm) + FR 9-11 (am)
- Instructor: Dr. Roland Molontay
 - 2015: MSc in Applied Mathematics (BME)
 - 2015 - 18: PhD student in Network Theory (BME)
 - 2016: visiting PhD student at Brown University
 - 2021 - : founder and leader of HSDSLab
 - 2022: visiting researcher at Indiana University
 - 2018 - 2020 : research fellow at BME
 - 2021 - 2022: assistant professor at BME
 - 2023 - ssociate professor at BMR
- E-mail: molontayr@gmail.com, data.science.ait@gmail.com
- Teaching lab session + grading homework: Kate Barnes
- Team project mentors: Donát Kóller, Marcell Nagy, József Pintér





Donát Kóller



Marcell Nagy



József Pintér



Kate Barnes

Teaching assistants /
graders / mentors

Syllabus



Some keywords: data types, data preparation, explanatory data analysis, supervised learning, classification, regression, model evaluation, clustering, recommender system, data visualization, case studies



Form of teaching: mostly lectures (with presentation), problem-solving sessions, computer-assisted problem solving (mostly in form of homework problems)



Course material: presented lecture slides (with oral explanation) + problem sheets + iPython notebooks



Plenty of useful materials are available online

Aim of the course

- Provide a broad overview of the field
- Learn about theory and use the methods in exciting real-life datasets
- Excel in your interview for a junior data scientist position
 - Both in oral interview and take-home assignments

The screenshot shows a web page from 'towards data science' featuring an article titled 'Top 30 data science interview questions'. The article is authored by Nitin Panwar, with a profile picture and a 'Follow' button. It was published on Dec 31, 2018, and has a 17 min read duration. The page includes navigation links for DATA SCIENCE, MACHINE LEARNING, PROGRAMMING, VISUALIZATION, AI, PICKS, MORE, and CONTRIBUTE. A sidebar on the left lists '10 Highly Probable Data Scientist Interview Questions' and includes links for Machine learning, Python, and SQL, along with user Soner Yildirim's profile and a 6 min read duration.

10 Highly Probable Data Scientist Interview Questions

Machine learning, Python, and SQL

Soner Yildirim · 2 days ago · 6 min read *

Nitin Panwar [Follow](#)

Dec 31, 2018 · 17 min read

towards
data science

DATA SCIENCE MACHINE LEARNING PROGRAMMING VISUALIZATION AI PICKS MORE | CONTRIBUTE

Twitter Facebook LinkedIn

Recommended literature

- Tan, Pang-Ning, Michael Steinbach, and Vipin Kumar. *Introduction to data mining*. 2005.
- James, Gareth, et al. *An introduction to statistical learning*. Vol. 112. New York: Springer, 2013.
- Leskovec, Jure, Anand Rajaraman, and Jeffrey David Ullman. *Mining of massive datasets*. Cambridge University Press, 2014.
- Sammut, Claude, and Geoffrey I. Webb, eds. *Encyclopedia of machine learning and data mining*. Springer, 2016.

The books are uploaded in Moodle in pdf form!

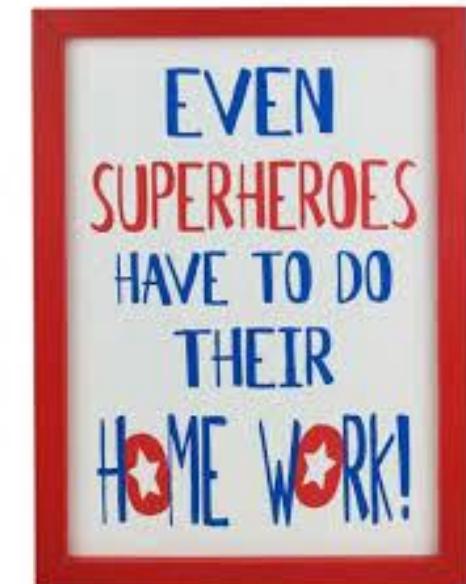
Requirements

- Attendance: there will be sign-up sheets every time
- MIDTERM (25%)
 - On Week 7
 - You can use your own „cheat sheets”/ formula sheets / notes
- FINAL (25%)
 - On Week 14
 - You can use your own „cheat sheets”/ formula sheets /notes
- HOMEWORK problems (25%)
 - There will be 4 HW sheets (roughly in every two weeks)
 - Programming problems
- PROJECT in teams (25%)
 - In teams of 3 (2-4) students



Homework policy

- The homework must be your own work.
 - You can look up books / search for help online
 - If you use longer code snippets from online sources, you must refer to the source
 - The same applies to ChatGPT or similar
 - You are encouraged to help each other if one of you gets stuck, share some ideas with each other
 - You must not send your entire homework to your peers
 - Copying is forbidden but giving assistance is encouraged
- Homework related questions:
data.science.ait@gmail.com
(Kate, Donát or József will help you!)



Homework – late submission policy

- Homework may be submitted after the deadline, but late submission will result in the following points deductions:
 - within 2 hours: 100%
 - within 24 hours: 95%
 - within 48 hours: 90%
 - within 72 hours: 85%
 - within 120 hours: 70%
 - within 168 hours: 60%
 - within 240 hours: 50%
 - otherwise: 0%



Midterm / Final

- On week 7 / on week 14
- 100 minutes long each
- 25% each
- Theoretical questions and numerical examples
- You can use your own „cheat sheets”/ formula sheets
 - As many of **your own** notes as you wish



Projects

- A data science project in teams
 - Team: 3 (2-4) students
 - All the team members will get the same grade point
 - It is recommended to use a version control tool to share your codes with each other, e.g. <https://github.com/>
 - It also looks nice from a recruiter's point of view
 - A teaching assistant (mentor) will be assigned for each team to help you/ provide guidance.
- A list of potential project ideas
 - Coming soon
 - Encouraged to choose from this list
 - If you don't find a topic that interests you, you can also come up with another project that all the team members are interested in
- Schedule
 - W4: forming teams
 - W5: project is chosen, you have talked to the assigned TA, the project plan is ready
 - W9: milestone 1
 - W12: milestone 2
 - W14/15: classroom presentation

Expectations

- Delivering a sophisticated enough data science project
- Studying related work (related papers, projects)
 - TAs will help you finding the relevant literature
 - Goal: Understand what others have done, attempt to not only reproduce the results but improve them in some respects
- Implementing techniques that we have covered in class
- Try more models, evaluate them, find the best models
- Nice and shiny data visualization
- Optional but appreciated: using models/techniques that we have not covered in class

Deliverables

- W4: Team name + list of team members + indicating project preference
- W5: **Project plan**
 - After consulting with assigned TA
 - One-page long report answering the following questions:
 - What is the vision? Why is the problem interesting?
 - What is the purpose of the project? What results do you expect?
 - What data do you plan to use? How do you plan to gather the data?
 - Are the data big enough and of suitable quality?
 - What data preparation steps do you plan to take?
 - What methodology, what models do you plan to use?
 - How would you visualize the results?

Deliverables II.

- **W9: Milestone 1**
 - Two-page long report covering the followings:
 - Have you managed to gather the data? Do you have enough data of appropriate quality?
 - Did you collect the relevant related works? What useful information could you discover?
 - Initial data analysis steps
 - What next steps do you plan to take?
- **W12: Milestone 2**
 - Three-page long report covering the followings:
 - Reviewing the related works
 - Data understanding and data preparation steps
 - More data analysis steps, implementing some models and evaluating them

Final deliverable: classroom presentation

- W14/15: oral presentation should include
 - Description of the problem, motivation
 - Some review of related works
 - Description of the data set
 - Data preparation steps
 - Modeling steps (what models, parameters of the models)
 - Evaluation of the models
 - Sophisticated visualization
 - Interpreting the results, conclusion



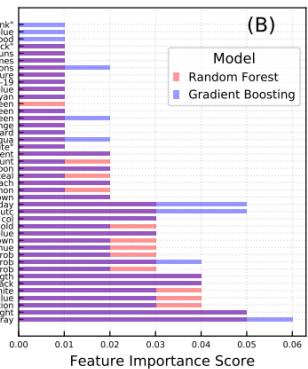
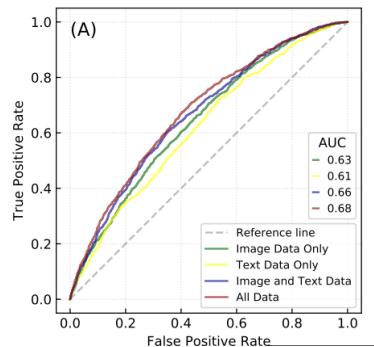
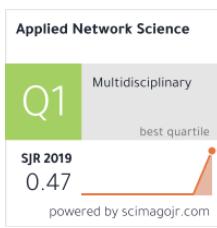
Final deliverable II.: codes

- W14: well-written codes (preferably an Ipython notebook)
 - Comments are necessary



Projects from previous years

- Content-based analysis of memes
 - Predicting virality
 - Engineering image-based and text-based features



Barnes,, Riesenmy, Trinh,, Lleshi, Balogh, & Molontay, R. (2021). Dank or Not?--Analyzing and Predicting the Popularity of Memes on Reddit. *Applied Network Science*



Projects from previous years II.

- March Madness bracket predictions
- Detect Parkinson Disease form Voice Recording
- Song popularity prediction on Spotify
- Sarcasm Detection



List of project ideas: coming soon

- Next week I will present some ideas
- Encouraged to choose from this list
- BUT: you may choose your own project
 - Find your own data set that you are interested in
 - Various data sources available online (e.g Kaggle)
 - Use own measurements
 - Independent data collection (e.g. web scraping techniques)



Grading

- MIDTERM (25%) + FINAL (25%) + HOMEWORK (25%) + PROJECT (25%)
- Cutoffs for letter grades:
 - 90% - A+
 - 85% - A
 - 80% - A-
 - 75% - B+
 - 70% - B
 - 65% - B-
 - 60% - C+
 - 55% - C
 - And so on...



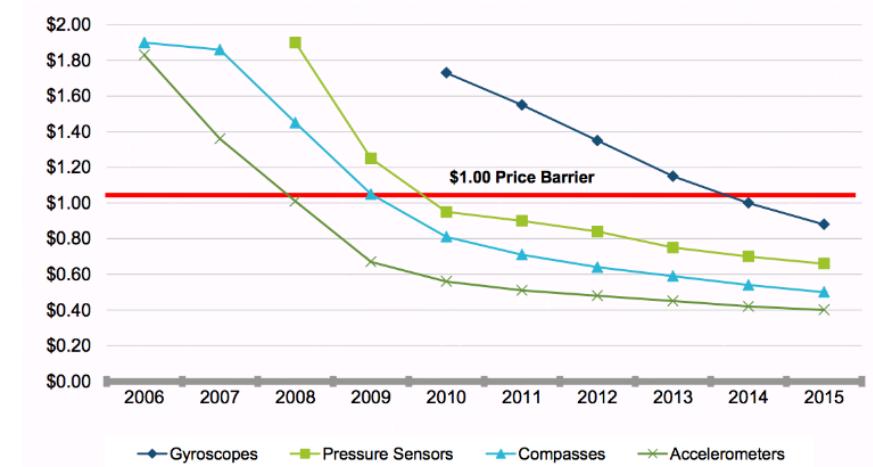
KEEP
CALM
AND
GET GOOD
GRADES

Schedule of the semester

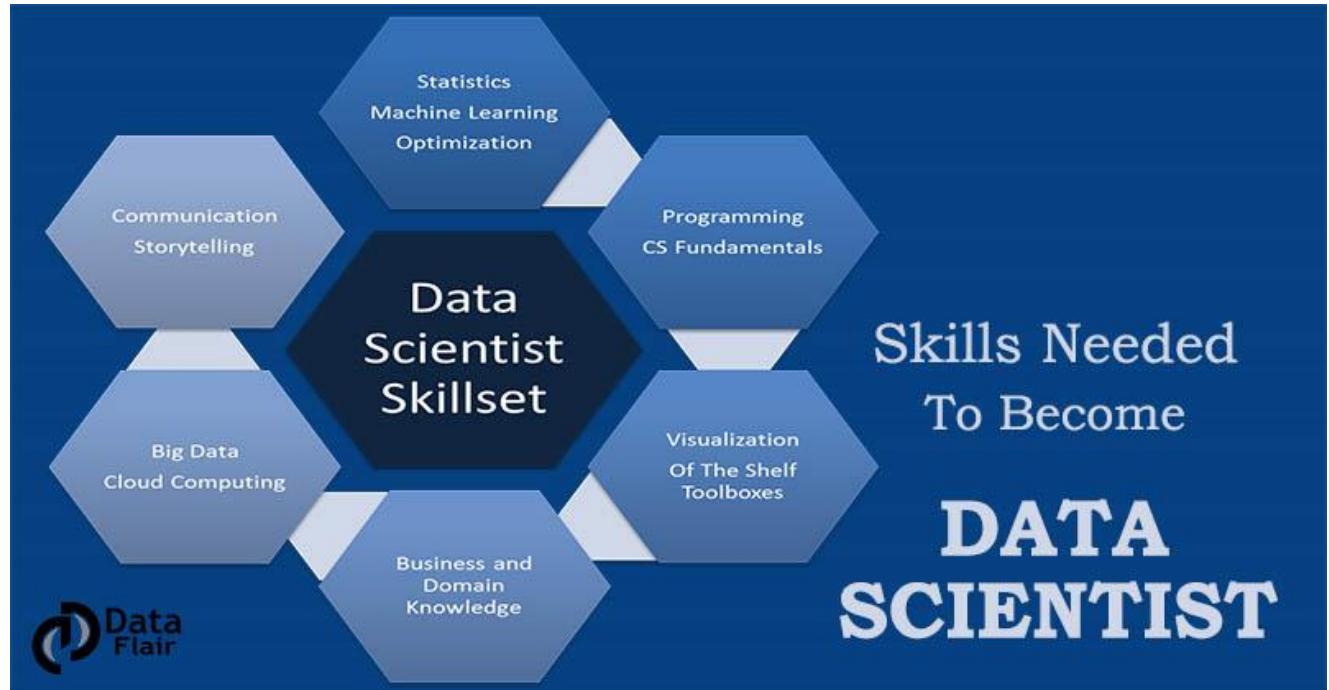
	<i>Monday midnight</i>	<i>Tuesday class</i>	<i>Friday class</i>
W1 (02/06)			
W2 (02/13)		HW1 out	
W3 (02/20)			
W4 (02/27)	HW1 deadline + TEAMS	HW2 out	
W5 (03/06)	PROJECT PLAN		
W6 (03/13)	HW2 deadline	HW3 out	
W7 (03/20)			MIDTERM
SPRING BREAK		SPRING BREAK	SPRING BREAK
W8 (04/03)	HW3 deadline	HW4 out	GOOD FRIDAY
W9 (04/10)	MILESTONE 1		
W10 (04/17)	HW4 deadline		
W11 (04/24)			
W12 (05/01)	MILESTONE 2		
W13 (05/08)			
W14 (05/15)		FINAL	PROJECT presentations
W15 (05/22)		PROJECT presentations	

Historical overview

- John Tukey: *The Future of Data Analysis, Annals of Mathematical Statistics*, 1962
 - Before his time, he predicted the emergence of a new scientific discipline about data
- 80s, 90s: storage capacity increases rapidly + prices decrease → data accumulation (data tomb)
 - Even exceeding Moore's law (the number of transistors in a dense integrated circuit doubles about every two years)— a similar observation is true for storage capacity
- „We are drowning in information,
but starving for knowledge”
John Naisbitt, 1982
- New sophisticated methods were needed to retrieve information from large databases → new algorithms
 - Initially heuristics (without proper theory)
 - In the new millennium it receives more research interest → theoretical support
- Nowadays: the price of sensors are decreasing + large text corpora → more data → the challenge is continuous



What does a data scientist know?



MODERN DATA SCIENTIST

Data Scientist, the sexiest job of the 21th century, requires a mixture of multidisciplinary skills ranging from an intersection of mathematics, statistics, computer science, communication and business. Finding a data scientist is hard. Finding people who understand who a data scientist is, is equally hard. So here is a little cheat sheet on who the modern data scientist really is.

MATH & STATISTICS

- ★ Machine learning
- ★ Statistical modeling
- ★ Experiment design
- ★ Bayesian inference
- ★ Supervised learning: decision trees, random forests, logistic regression
- ★ Unsupervised learning: clustering, dimensionality reduction
- ★ Optimization: gradient descent and variants



PROGRAMMING & DATABASE

- ★ Computer science fundamentals
- ★ Scripting language e.g. Python
- ★ Statistical computing packages, e.g., R
- ★ Databases: SQL and NoSQL
- ★ Relational algebra
- ★ Parallel databases and parallel query processing
- ★ MapReduce concepts
- ★ Hadoop and Hive/Pig
- ★ Custom reducers
- ★ Experience with xaaS like AWS

DOMAIN KNOWLEDGE & SOFT SKILLS

- ★ Passionate about the business
- ★ Curious about data
- ★ Influence without authority
- ★ Hacker mindset
- ★ Problem solver
- ★ Strategic, proactive, creative, innovative and collaborative

COMMUNICATION & VISUALIZATION

- ★ Able to engage with senior management
- ★ Story telling skills
- ★ Translate data-driven insights into decisions and actions
- ★ Visual art design
- ★ R packages like ggplot or lattice
- ★ Knowledge of any visualization tools e.g. Flare, D3.js, Tableau

Who is a data scientist?

„I think data scientist is a sexed-up term for a statistician.”

Nate Silver

„A data scientist is someone who is better at statistics than any software engineer and better at software engineering than any statistician.”

Josh Willis

„A data scientist is a statistician who lives in San Francisco.”

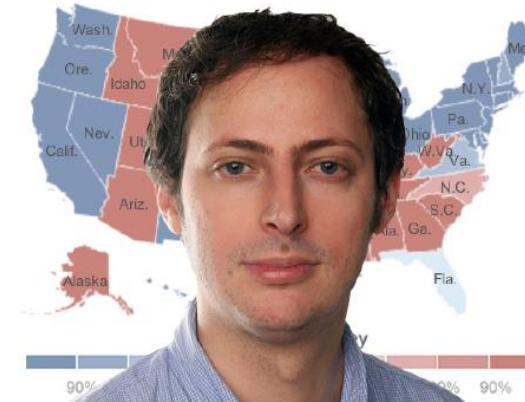
„Data Science is statistics on a Mac”

Twitter



Outlook – Nate Silver

- American statistician, the founder and editor in chief of FiveThirtyEight
- He accurately predicted the result of 49 states on 2008 presidential election and got all the 50 states right in 2012



#natesilverfacts

„When Alexander Bell invented the telephone he had 3 missed calls from Nate Silver.”

„Nate Silver can hear sign language.”

„For Nate Silver, asymptotic theory kicks in at N=1.”

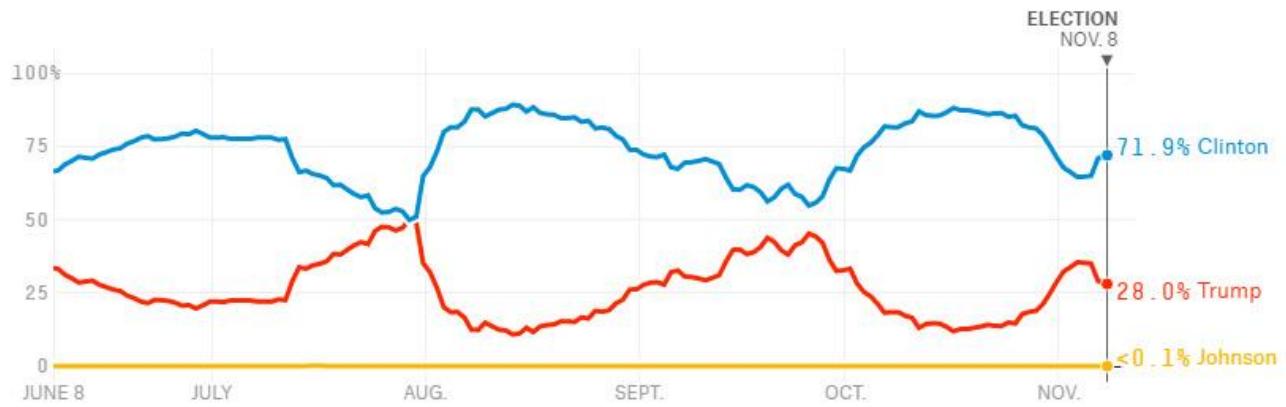
„Fearing Nate Silver, the Null Hypothesis rejected itself.”

„Nate Silver’s model fit the test data even better than the training data.”



Outlook - USA elections, 2016

- 2016: Nate Silver claims that Clinton has much more chance to win (he gives 72% chance to this scenario)
 - Other statisticians gave even less chance for Trump
 - Trump won the election
 - Big data also played a big role in the presidential campaign



Donald Trump's campaign shifted odds by making big data personal

Social media surveys helped to target thousands of individuals in swing states

“Gillian Tett

CAMBRIDGE ANALYTICA

Facebook fined £500,000 over Cambridge Analytica scandal

Oct 25, 2018

UK data watchdog says social media giant failed to safeguard its users' personal information



Justin Sullivan/Getty Images
Mark Zuckerberg has been given until the end of the month to respond

Facebook has been fined £500,000 by the UK's data watchdog for allowing political consulting firm Cambridge Analytica to harvest the information of millions of people without their consent.

Data mining firm behind Trump election built psych political profiles of nearly every American voter

Data in political

Zuckerberg apologises for book's 'mistakes' over Cambridge

silence, CEO announces Facebook will change with third-party apps and admits 'we made

it go either way, in deciding which of his key political segmented voter groups. Once the campaign was sending out



Cambridge Analytica: how 50m Facebook records were hijacked

1

Approx. 320,000 US voters ('seeders') were paid \$2-5 to take a detailed personality/political test that required them to log in with their Facebook account

2

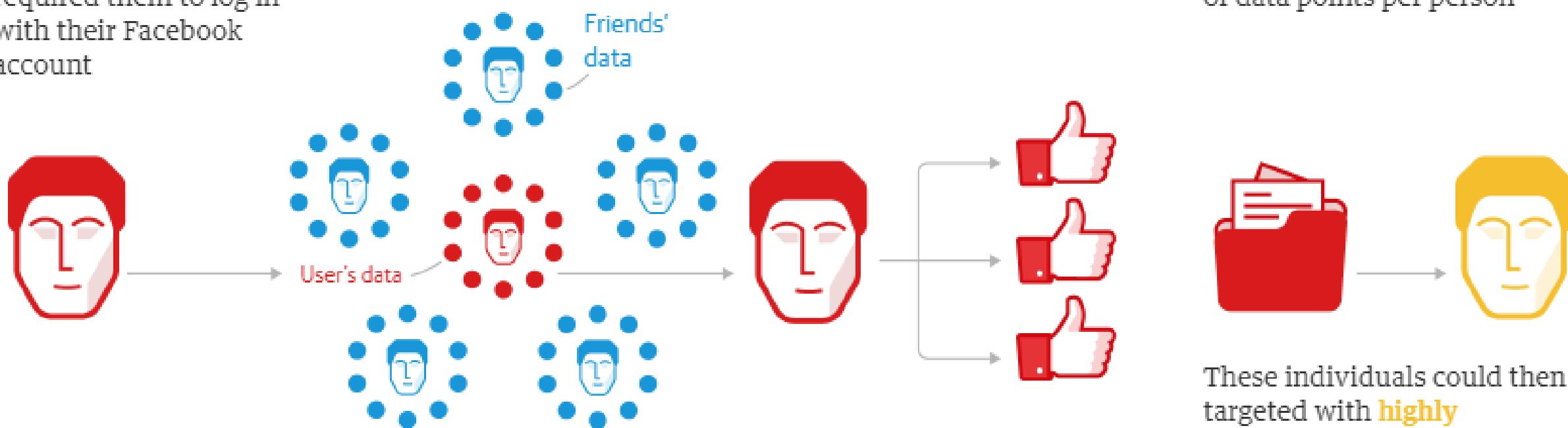
The app also collected data such as likes and personal information from the test-taker's Facebook account ...

3

The personality quiz results were paired with their Facebook data - such as likes - to seek out psychological patterns

4

Algorithms combined the data with other sources such as voter records to create a superior set of records (initially 2m people in 11 key states*), with hundreds of data points per person



... as well their friends' data, amounting to over 50m people's raw Facebook data

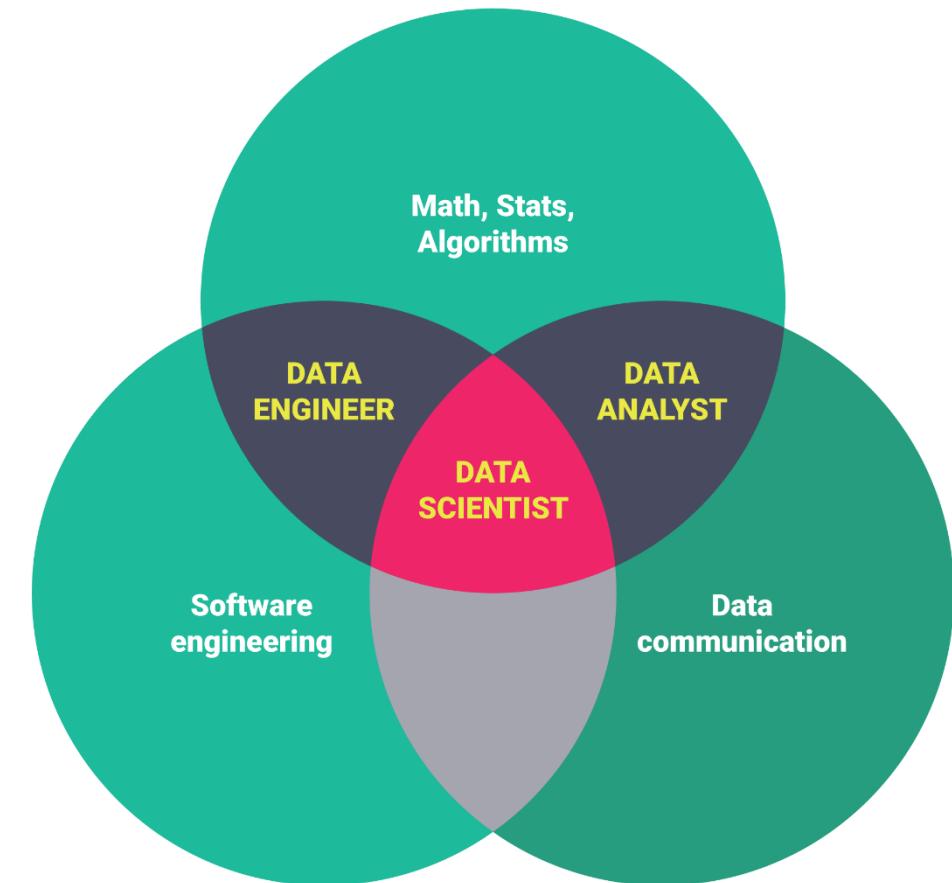
These individuals could then be targeted with highly personalised advertising based on their personality data

Statistics vs. Data science

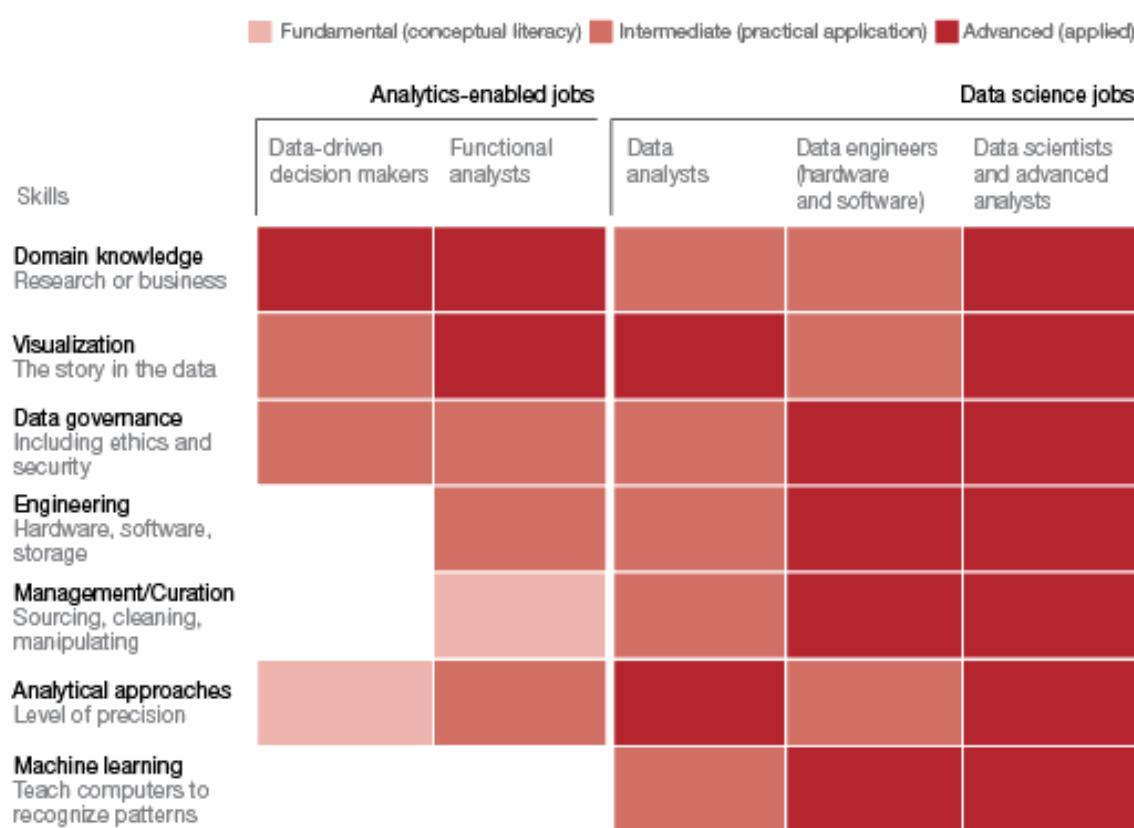
- Different aspects/approaches: testing hypothesis (statistical tests) vs. Finding hypothesis (more general question)
- Studying DNA sequences
 - Statistician: Is there a significant connection between a certain DNA subsequent and a certain disease?
 - Data scientist: What are the connections between certain diseases and certain DNA subsequents?
- Studying smoking habits
 - Statistician: Is there a significant difference regarding smoking ratios between males and females?
 - Data scientist: What are the typical groups regarding smoking habits?

Roles in data science

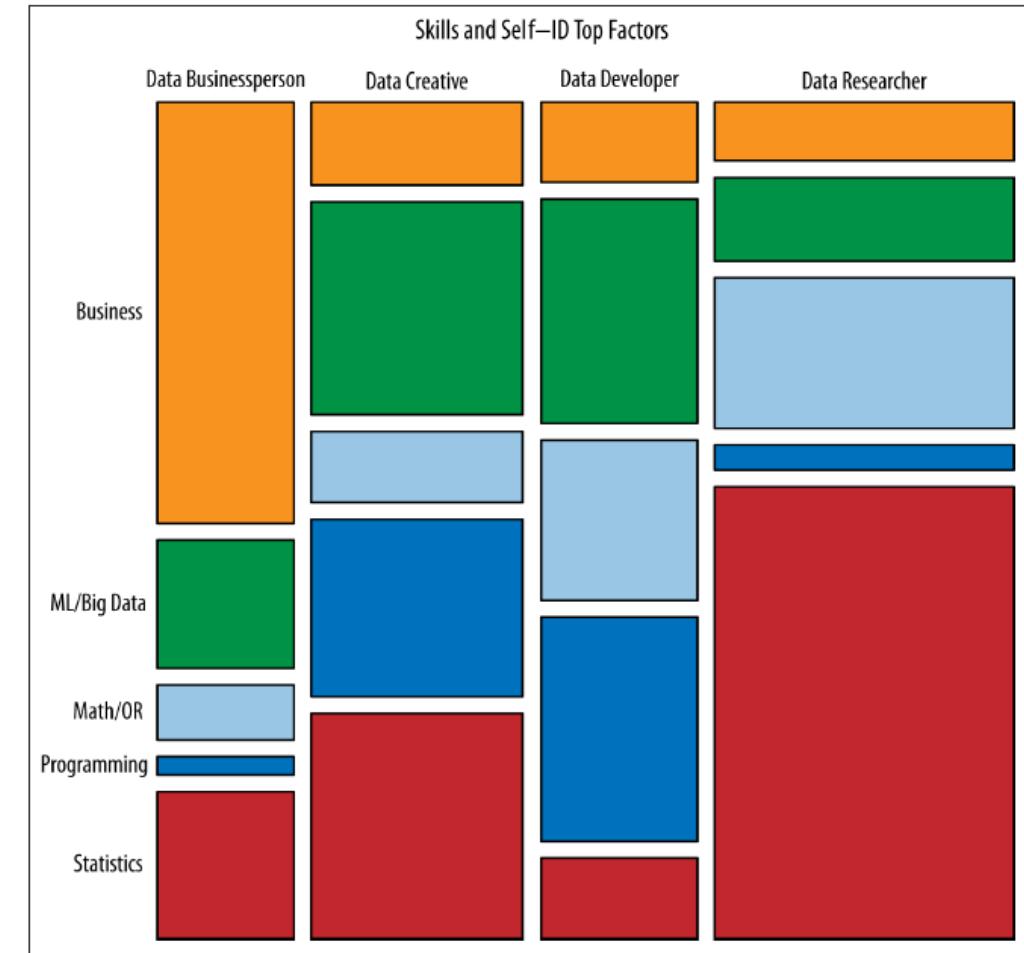
- There are no strict roles, it depends on the company, on the project
- Job listings are also ambiguous
 - Same positions may cover totally different job descriptions



Positions and required skills



Source: PwC analysis based on Burning Glass Technologies data, January 2017.





Data Engineers

Data Analysts

Machine Learning Engineers

Data Scientists

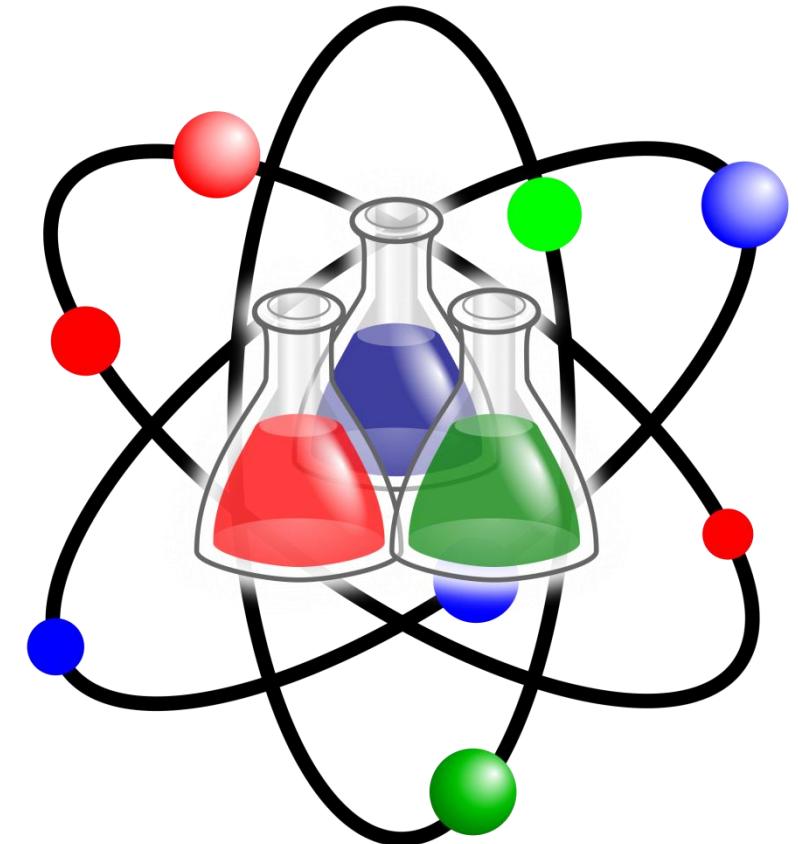
Some application areas– business world

- Telecommunication (optimal pricing, churn detection)
 - Customer history, phone logs
- Retail (up-selling, cross-selling, improving customer satisfaction)
 - Credit card transactions, data from online purchase
- Banks (credit assessment, fraud detection)
 - Customer history, credit card transactions
- Several other domains (stock exchange, social media, websites)



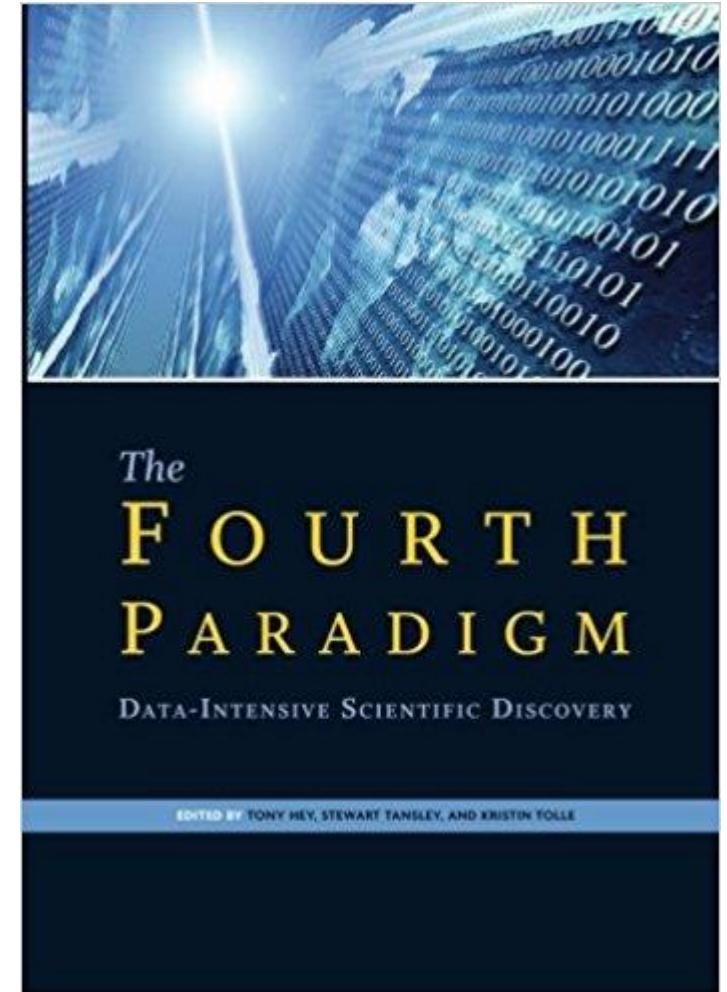
Some application areas– science

- Particle physics
 - Finding new phenomena, validating theories
- Astronomy
 - Analyzing data space telescopes
 - Classifying photos automatically (without a human)
- Drug development
 - Finding drug substance, take out experiments
- Medicine
 - Supporting diagnostic
 - Monitoring systems
- Several other areas (brain research, gene map)



Scientific paradigm shift?

- A thousand years ago: empirical science
 - Describing nature
- Last few hundred years: theoretical approach
 - Introducing models, generalizations
- Last few decades: computational approach
 - Simulation of complex phenomena
- Nowadays: data-driven approach
 - Merging experiments, theory and simulations

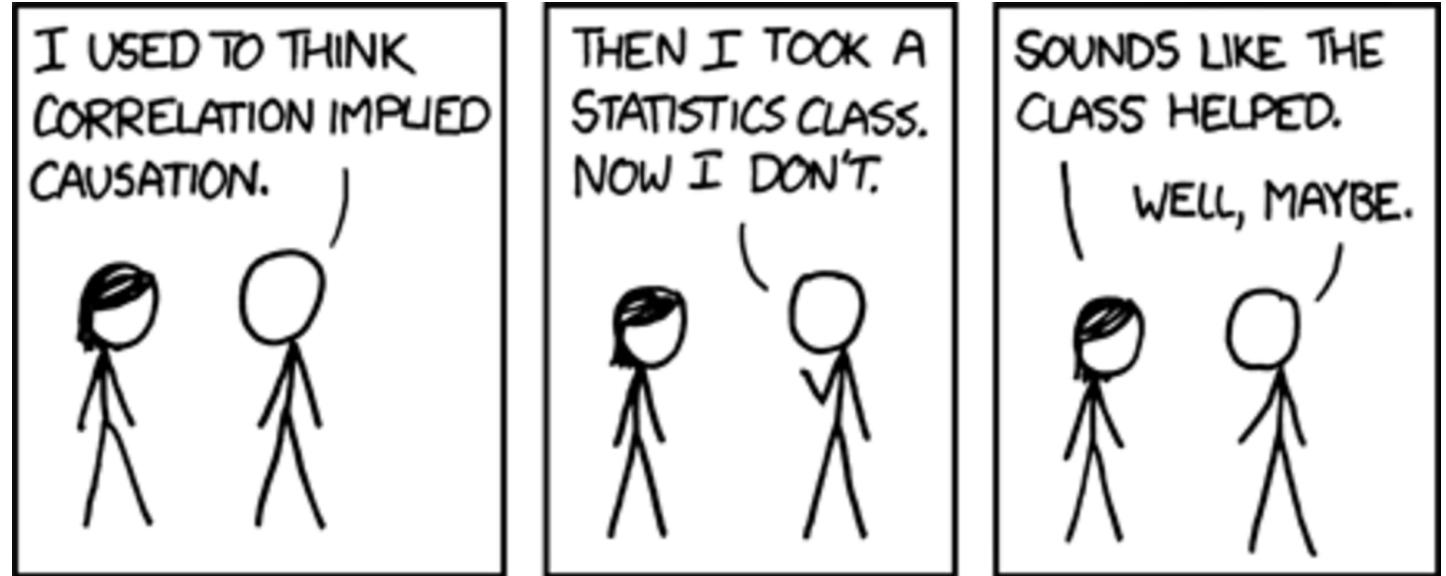


What is not data science?

- Processing and analyzing data are not always considered to be data science
 - Descriptive analysis (e.g. a summary of a population census) is not data science by itself
- Data science looks for patterns, correlations in data BUT correlation DOES NOT imply causation
- Exploring cause and effect relationship is usually out of scope of data science
 - Need for randomized groups
 - Using control groups in verified environments
 - Econometrics – finding causal relations in economic data

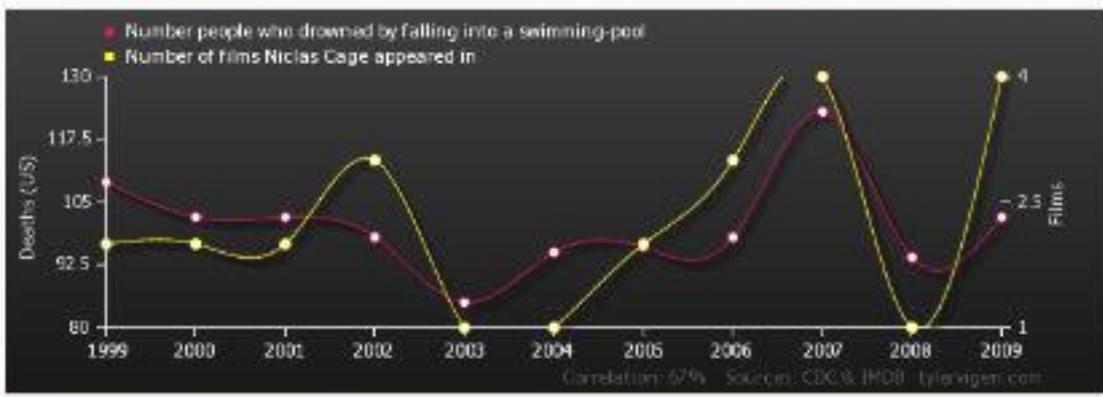
Correlation ≠ causation

- Strong correlation:
 - Height and hair length
 - Ice cream sales and number of drownings

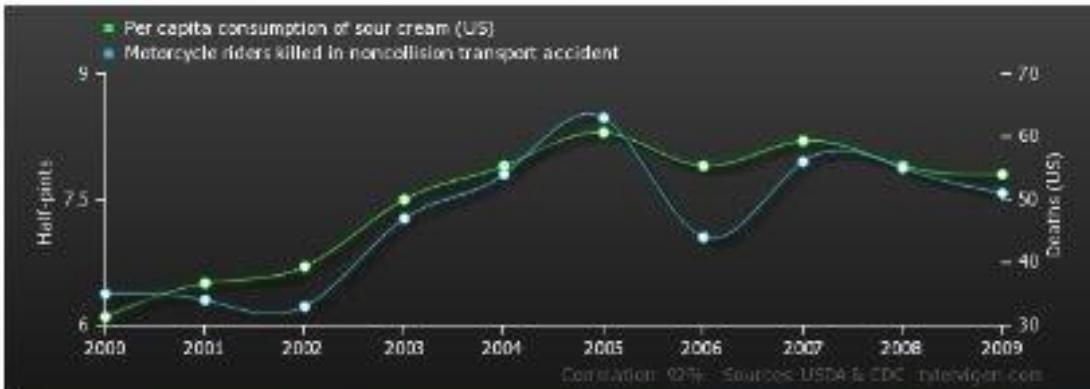


- Be careful! From data one can retrieve connections that is just there due to chance
 - You can't generalize them!
 - See next slides!

Number people who drowned by falling into a swimming-pool
 correlates with
Number of films Nicolas Cage appeared in



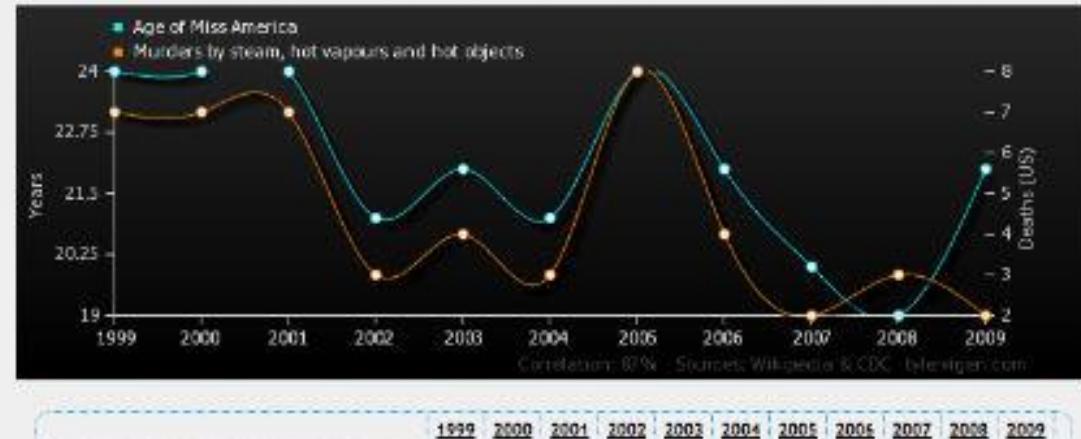
Per capita consumption of sour cream (US)
 correlates with
Motorcycle riders killed in noncollision transport accident



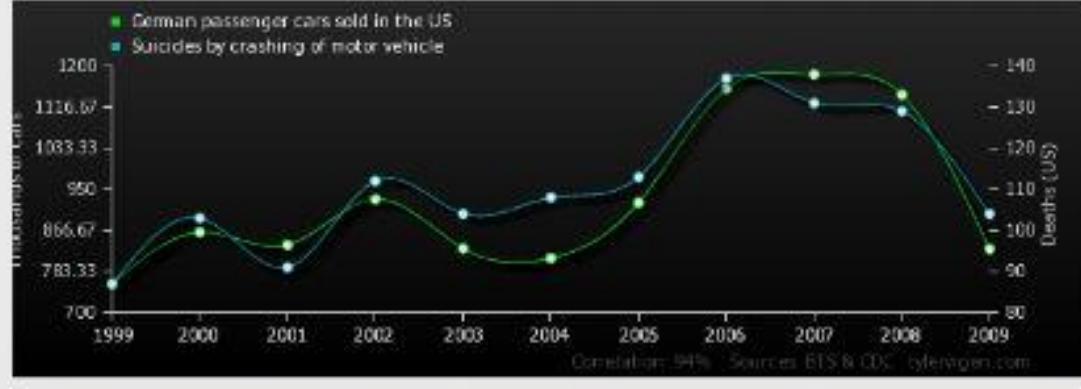
	2000	2001	2002	2003	2004	2005	2006	2007	2008	2009
Per capita consumption of sour cream (US) Half-pints (USDA)	6.1	6.5	6.7	7.5	7.9	8.3	7.9	8.2	7.9	7.8
Motorcycle riders killed in noncollision transport accident Deaths (US) (CDC)	35	34	33	47	54	63	44	56	55	51

Correlation: 0.916391

Age of Miss America
 correlates with
Murders by steam, hot vapours and hot objects



German passenger cars sold in the US
 correlates with
Suicides by crashing of motor vehicle



	1999	2000	2001	2002	2003	2004	2005	2006	2007	2008	2009
German passenger cars sold in the US Thousands of cars (BIS)	758	863	837	930	830	810	923	1,154	1,183	1,142	829
Suicides by crashing of motor vehicle Deaths (US) (CDC)	87	103	91	112	104	108	113	137	131	129	104

Correlation: 0.935701

Python vs. R

The graphic features a blue hexagonal logo with a white brain icon and the text "DataCamp Learn data analysis for free". Below it, the title "DATA SCIENCE WARS" is displayed in large, white, sans-serif letters against a background of colorful lightning bolts. At the bottom, there's a comparison between R and Python. On the left, the R logo is shown with the text "VS." next to it. On the right, the Python logo is shown with the word "python" next to it. Two quotes are presented: "The closer you are to statistics, research and data science, the more you might prefer R." and "The closer you are to working in an engineering environment, the more you might prefer Python." Below each quote is a brief explanatory sentence.

"The closer you are to statistics, research and data science, the more you might prefer R."

R has a steep learning curve at start. Once you know the basics, you can easily learn advanced stuff.

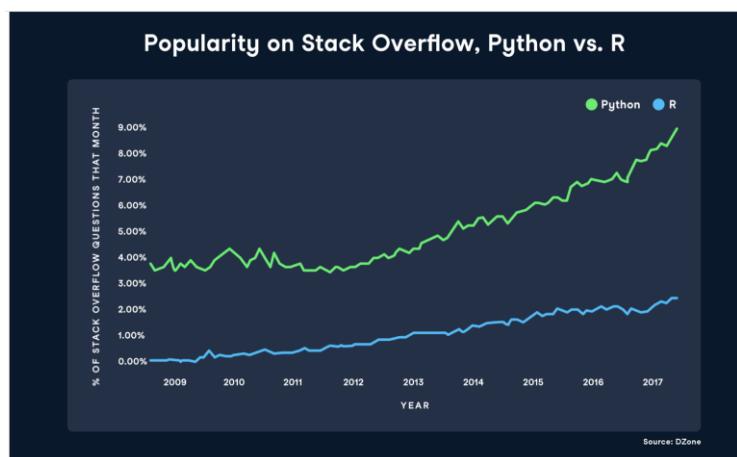
R is not hard for experienced programmers.

"The closer you are to working in an engineering environment, the more you might prefer Python."

Python's focus on readability and simplicity makes that its learning curve is relatively low and gradual.

Python is considered a good language for starting programmers.

Difference Between and Python		
Features	R	Python
Scope	Used mainly for statistical modeling	Used for a variety of purposes like web-application development and data analysis
Used By	Statisticians, Analyst & Data Scientist	Developer, Data Engineers & Data Scientist
Suitable For	People with no prior experience in programming	Newbies to experienced IT professionals
Package Distribution	CRAN	PyPi
Visualization Tools	ggplot2, plotly, ggiraph	Matplotlib, bokeh, seaborn



Some other interesting comparison:

https://www.datacamp.com/community/tutorials/r-or-python-for-data-analysis#gs.fC_rDHo

Refreshing Python

- I really encourage everybody to refresh their knowledge in Python
 - A good tutorial
 - https://www.tutorialspoint.com/python/python_quick_guide.htm
 - Until „Directories in Python”
 - From „Creating Classes” to „Bulit-In Class Attributes”
 - If you need more than a quick refresh I recommend the following short, free interactive online courses:
 - <https://www.datacamp.com/courses/intro-to-python-for-data-science/>
 - <https://www.codeschool.com/courses/try-python>
 - Some other useful materials are uploaded in Moodle



Python preparation

- We will use Jupyter IPython with Anaconda distribution
 - Runs in a web browser
 - Interactive
 - Simple
 - Scenic
- Download it by following this link: <https://www.anaconda.com/download/>
- Other useful links:
 - <https://ipython.org/install.html>
 - <https://www.continuum.io/downloads#windows>



Using Ipython notebooks

- From Anaconda Navigator you can launch Jupyter Notebook or JupyterLab

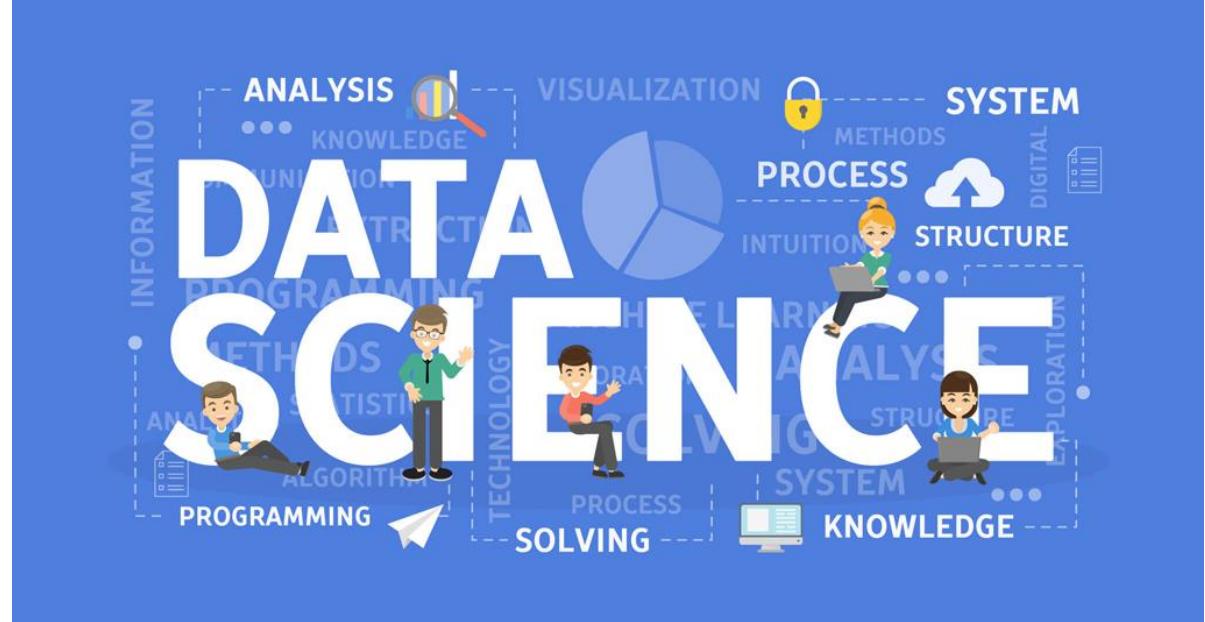
- The notebook will start in your default web browser



- More information:
<https://jupyter.readthedocs.io/en/latest/running.html#running>

What are we going to learn about?

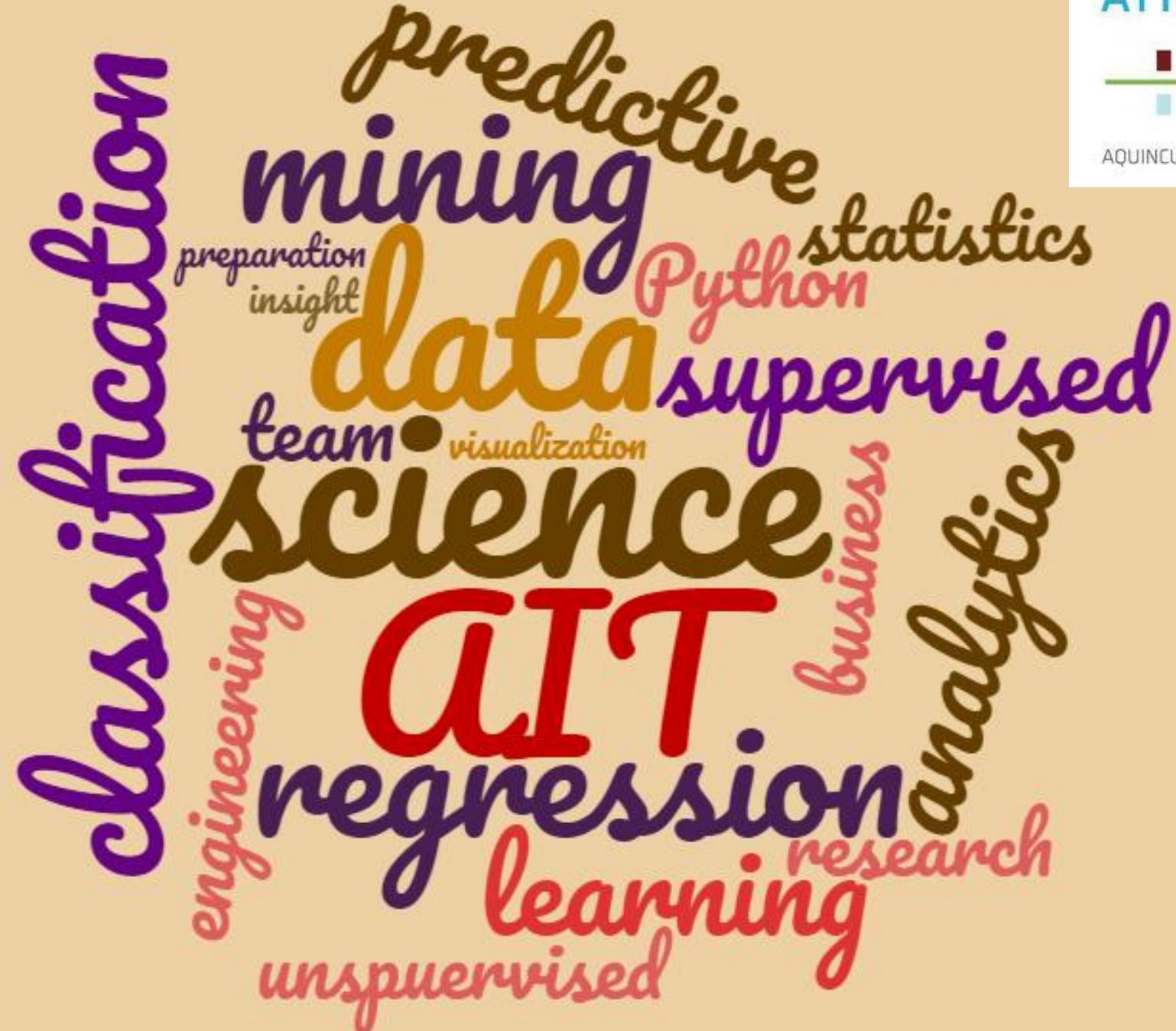
- Data types, data processing
 - Classification (kNN, decision tree, naive Bayes, logistic regression, SVM, neural networks)
 - Hybrid classification (bagging, boosting, ensemble)
 - Regression (linear, polynomial)
 - Evaluating models
 - Clustering (k-means, hierarchical, density based)
 - Recommender systems
 - Networks, PageRank algorithm
 - Data visualization
 - Case studies



Acknowledgement

- András Benczúr, Róbert Pálovics, SZTAKI-AIT, DM1-2
- Krisztián Buza, MTA-BME, VISZJV68
- Bálint Daróczy, SZTAKI-BME, VISZAMA01
- Judit Csima, BME, VISZM185
- Gábor Horváth, Péter Antal, BME, VIMMD294, VIMIA313
- Lukács András, ELTE, MM1C1AB6E
- Tim Kraska, Brown University, CS195
- Dan Potter, Carsten Binnig, Eli Upfal, Brown University, CS1951A
- Erik Sudderth, Brown University, CS142
- Joe Blitzstein, Hanspeter Pfister, Verena Kaynig-Fittkau, Harvard University, CS109
- Rajan Patel, Stanford University, STAT202
- Andrew Ng, John Duchi, Stanford University, CS229





Schedule of the semester

	<i>Monday midnight</i>	<i>Tuesday class</i>	<i>Friday class</i>
W1 (02/06)			
W2 (02/13)		HW1 out	
W3 (02/20)			
W4 (02/27)	HW1 deadline	HW2 out	
W5 (03/06)	PROJECT PLAN		
W6 (03/13)	HW2 deadline	HW3 out	
W7 (03/20)			MIDTERM
SPRING BREAK		SPRING BREAK	SPRING BREAK
W8 (04/03)	HW3 deadline	HW4 out	GOOD FRIDAY
W9 (04/10)	MILESTONE 1		
W10 (04/17)	HW4 deadline		
W11 (04/24)			
W12 (05/01)	MILESTONE 2		
W13 (05/08)			
W14 (05/15)		FINAL	PROJECT presentations
W15 (05/22)		PROJECT presentations	

Reminder



**Hey!! Don't
forget to...**

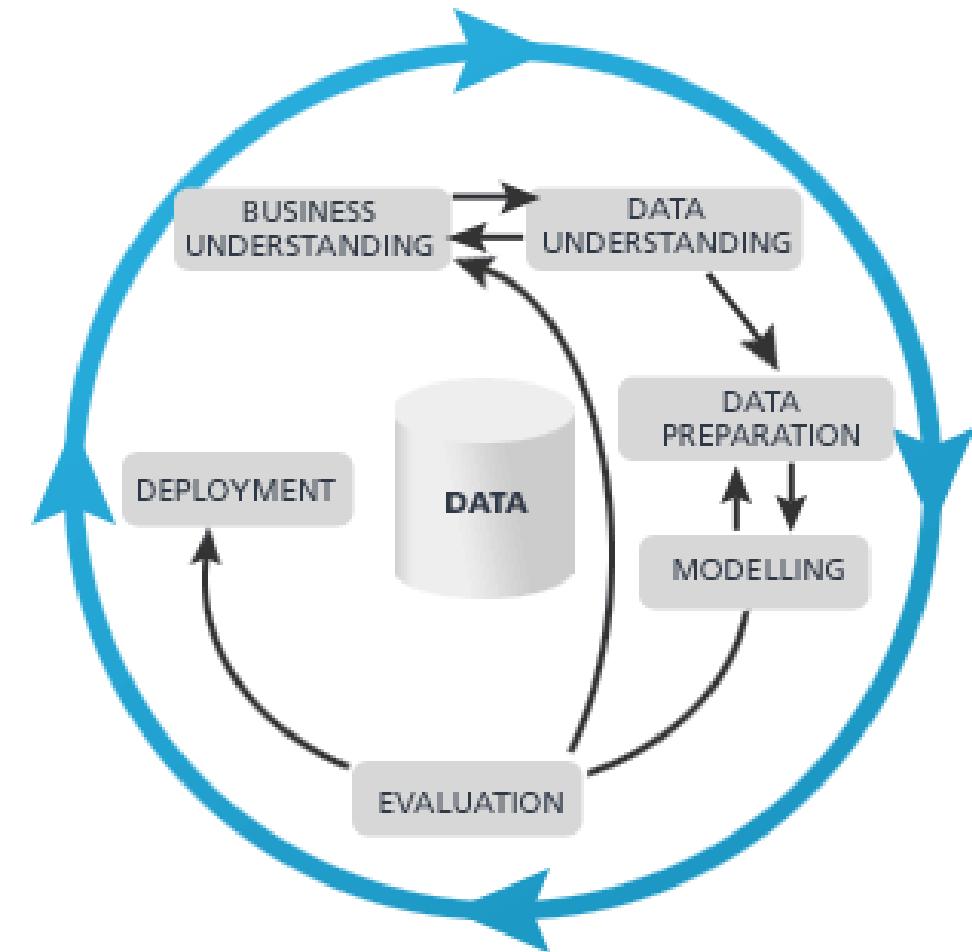
- Please refresh Python and download Jupyter Ipython notebook with Anaconda distribution
 - See the last slides of Lecture 01

Process for data mining / data science

- CRISP-DM: CRoss-Industry Standard Process for Data Mining

- A technical standard with its own limitations but worth following
- Back and forth effect
- Cyclic

CRISP-DM
CROSS INDUSTRY STANDARD PROCESS FOR DATA MINING



BU - Business understanding



What is the aim of the project?



What is its business relevance?



What is the research question?



How can it be translated to a data science question?



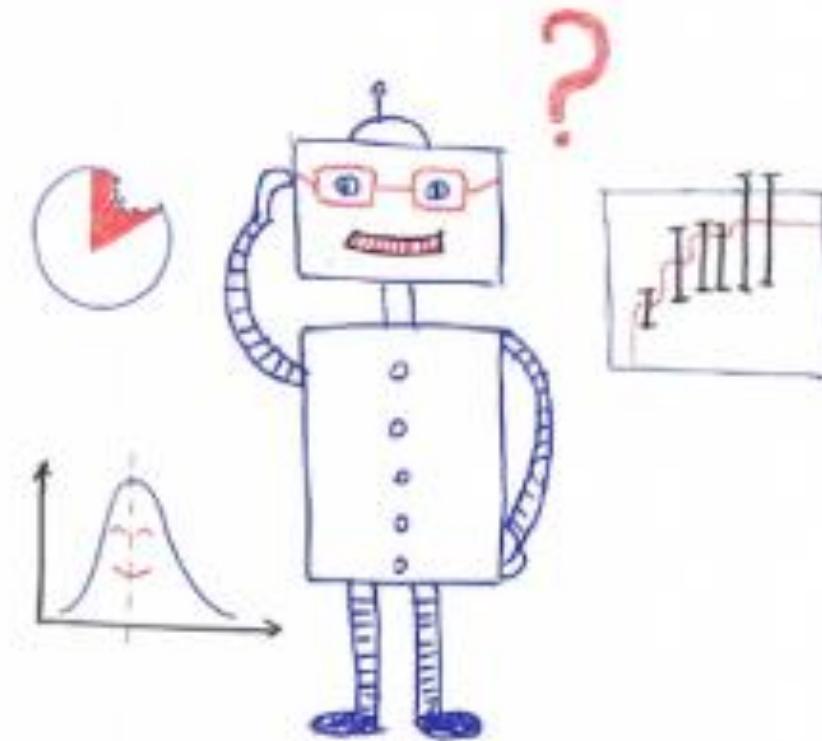
DU - Data understanding

What data do we have?

Can we collect more data?

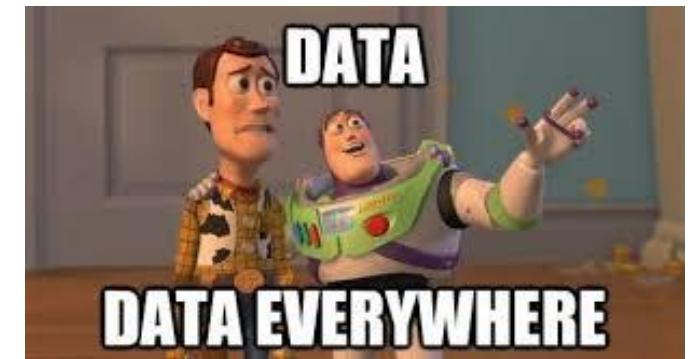
What is the quality of the data?

What do the features mean?



Dataset

- Everything that carries information, and we would like to extract insights from
- In the simplest case the data is structured, i.e., it is like a table / data frame
 - Rows: records, observations, data points, instances
 - Columns: attributes, features
 - A record is described by the values of the attributes in its row
- The data can be inherently unstructured but in many cases we pursue to make it structured



Representation of data

- Rows: record, object, data point, observation, entity, representative, item
- Columns: attribute, feature, dimension
 - Regarding regression, also called: explanatory variable, independent variable
- Target variable/output (for supervised learning):
 - For classification problems: label, class
 - For regression problems: response variable, dependent variable

The diagram illustrates the representation of data as a table. A curly brace on the left side groups the rows of the table, labeled "records". A curly brace at the top groups the columns, labeled "attributes". The table has 10 rows and 5 columns. The columns are labeled Tid, Refund, Marital Status, Taxable Income, and Cheat. The data is as follows:

Tid	Refund	Marital Status	Taxable Income	Cheat
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

Attribute types

- Continuous: real-valued (in most cases it is also considered to be „continuous” if it can take countably infinitely many values)
 - E.g.: temperature, height, weight
- Discrete: can take finitely many values (sometimes variables with countably infinitely many possible values also)
 - Usually represented with integer values or category names
 - E.g.: ZIP code, marital status, (quantity)
- Binary: a special discrete attribute – possible values 0 and 1
 - Sometimes has asymmetric meaning: 0 may mean that something is not true, something is missing
 - Sometimes they can be found in sparse data matrices where the vast majority of the elements are 0
 - E.g. document-term matrices
 - Sparse data structures need special methods

Attribute types – another partition

- Categorical / nominal variables
 - E.g. gender, marital status, place of birth, got treatment?, is overweight?
 - Reasonable operations: frequencies, mode
- Ordinal variables
 - May seem to be categorial, but can be ordered in a quantitative manner
 - E.g. stages (inchoative, advanced), military ranks (admiral, captain, commander), letter grades
 - Reasonable operations: median (but average is not), percentile, rank-correlation
- Quantitative (numerical) variables
 - Interval variables
 - The numerical values show both the ordinal relationship and the extent of deviation
 - E.g.: temperature ($^{\circ}\text{C}$, $^{\circ}\text{F}$), IQ score
 - Reasonable operations: average, difference, variance, correlation
 - Ratio variables
 - They have all the properties of an interval variable, and also have a clear definition of 0.0 (none of that variable)
 - E.g.: temperature ($^{\circ}\text{K}$), height, weight, pieces
 - Reasonable operations: any operations that are defined for real numbers

What to compute?

OK to compute....	Nominal	Ordinal	Interval	Ratio
frequency distribution.	Yes	Yes	Yes	Yes
median and percentiles.	No	Yes	Yes	Yes
add or subtract.	No	No	Yes	Yes
mean, standard deviation	No	No	Yes	Yes
ratio.	No	No	No	Yes

Determine the type of the following attributes in two ways:

1: Continuous, discrete, binary? 2: Nominal, ordinal, quantitative (interval, ratio)?

1. Altitude
2. Total number of rooms in a hotel
3. Military ranks
4. Distance from the center of Heroes Square
5. International Standard Book Number (ISBN)
6. Degree: measurement of plain angle (between 0 and 360)
7. Degree of transparency:
transparent, translucent, opaque
8. Cloakroom ticket numbers
9. Grades (from F to A+)
10. Medals (bronze, silver, gold)
11. Sex (male, female)
12. Age (in years)
13. pH (acidity or basicity of an aqueous solution)

Data exploration

- What are the important features? Are there any interesting relations or redundancy?
 - Are there any apparent problems with the data that need action?
 - Scaling, missing data, outliers
 - Are there patterns that can be recognized using data visualization?
-
- Methods:
 - Summary statistics / descriptive statistics
 - Simple data visualization, plots



Summary statistics

- Purpose: to summarize the variables with numerical values
 - Easy to compute and informative
 - What are the typical values, how scattered are they, what are the frequencies?
 - They can be queried by a simple command in any programs
- Categorical variables: frequencies
- Numerical variables:
 - Percentiles: indicating the value below which a given percentage of observations in a group of observations (e.g. values in a column) falls, e.g. p -percentile denotes the x_p value below which $p\%$ of the observations may be found
 - Usually considered values: min, 25, 50, 75, max
 - Mean: arithmetical average of the values: $mean(x) = \bar{x} = \frac{1}{m} \sum_{i=1}^m x_i$
 - Sensitive to outliers median is more robust
 - Median: the value separating the higher half from the lower half of a data sample
 - Similar to the 50-percentile but not the same

$$median(x) = \begin{cases} x_{r+1} & \text{if } m = 2r + 1 \\ \frac{1}{2}(x_r + x_{r+1}) & \text{if } m = 2r \end{cases}$$

Values describing deviation

- Range: what is the range of the possible values: $\max - \min$
- Sample variance:
 - Sensitive to outliers

$$S_X^2 = \frac{1}{m-1} \sum_{i=1}^m (X_i - \bar{X})^2$$

- Standard deviation: root of the variance
- Average absolute deviation:

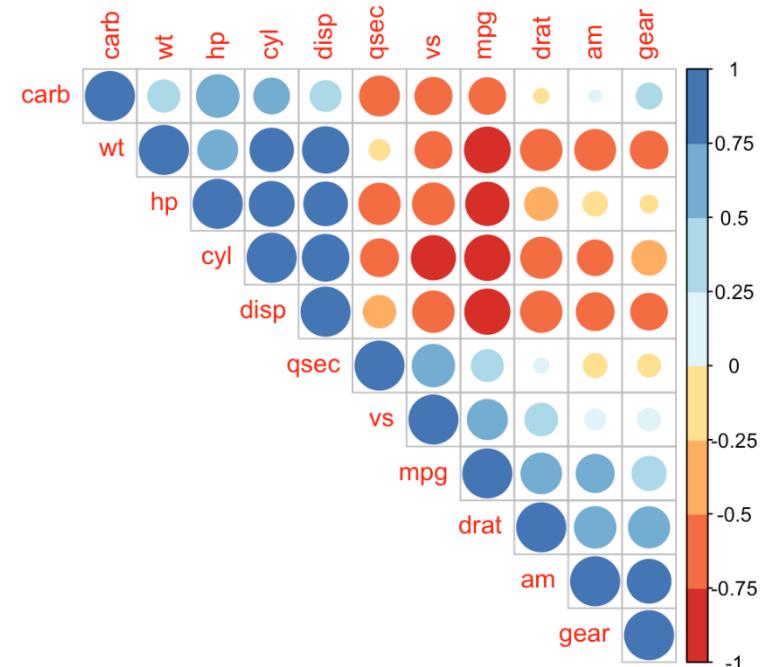
$$\frac{1}{m} \sum_{i=1}^m |X_i - \bar{X}|$$

Covariance and correlation

- Sample covariance between values of j th and k th columns (attributes)

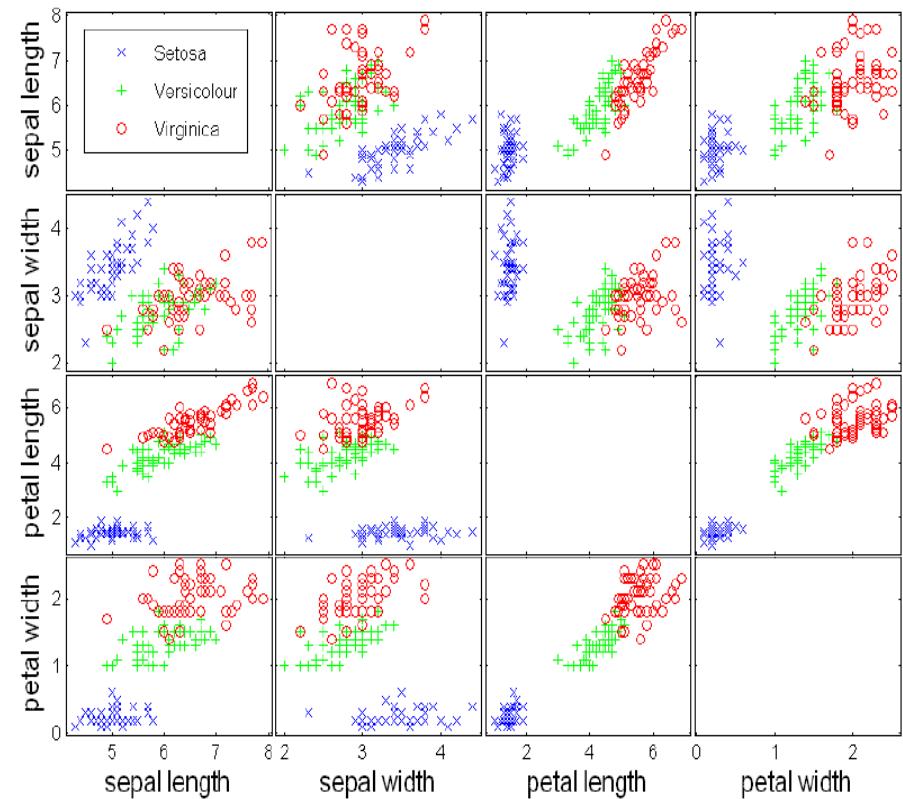
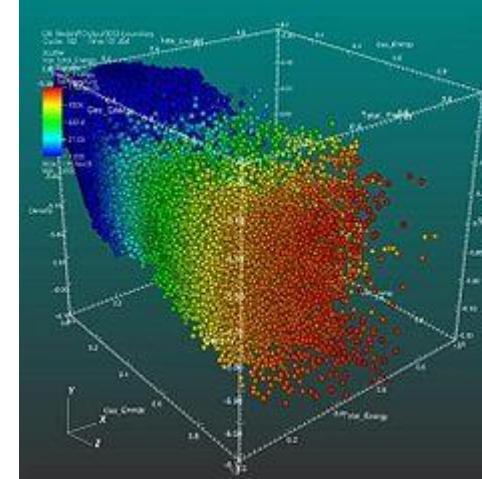
$$q_{jk} = \frac{1}{m-1} \sum_{i=1}^m (X_{ij} - \bar{X}_j)(X_{ik} - \bar{X}_k)$$

- We can form a matrix: sample covariance matrix (symmetric)
 - Sample correlation: $r_{jk} = \frac{q_{jk}}{s_j s_k}$
 - A sample correlation matrix can be formed
 - Sensitive (not robust) against outliers
 - Measure the strength of the linear relationship between variables



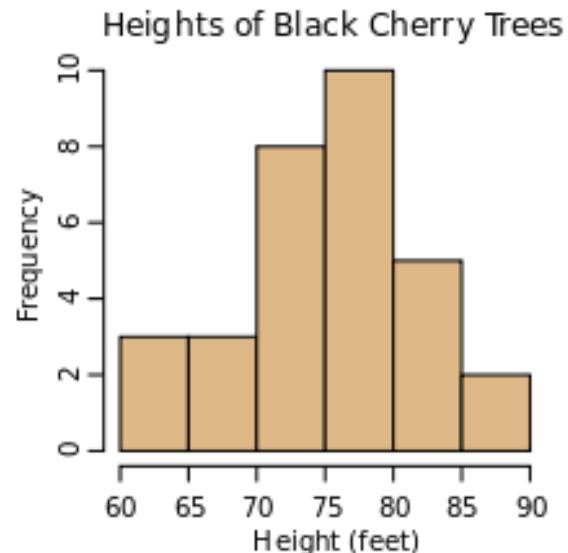
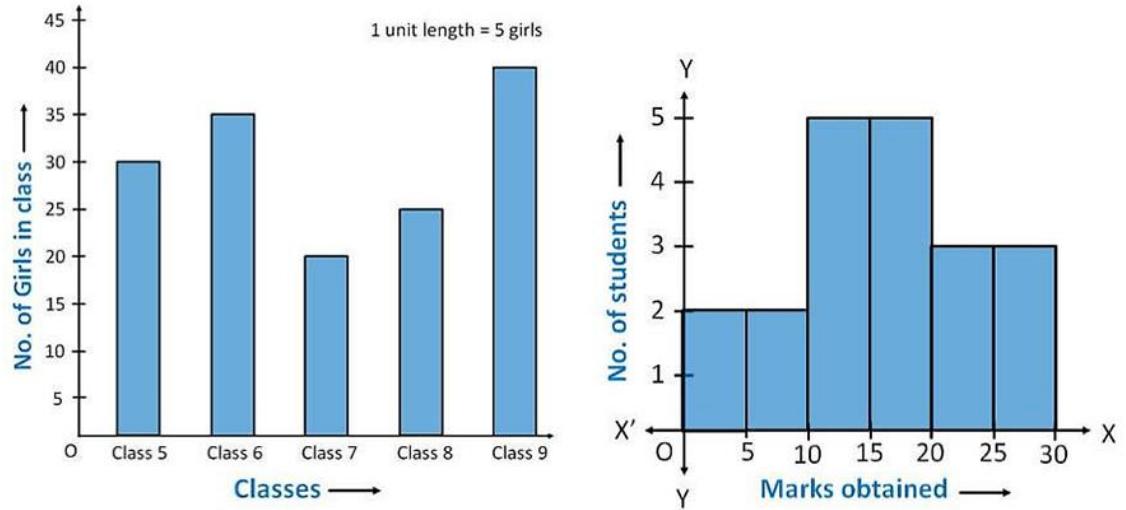
Scatterplot

- The objects correspond to points on the plane / in the space
- The coordinates of the points correspond to the values of two/three attributes of the object
- Beyond the (max) three dimensions the point can also have color/shape/size
 - So we can visualize 5-6 dimension all together
 - It is hard to interpret above 4 dimensions



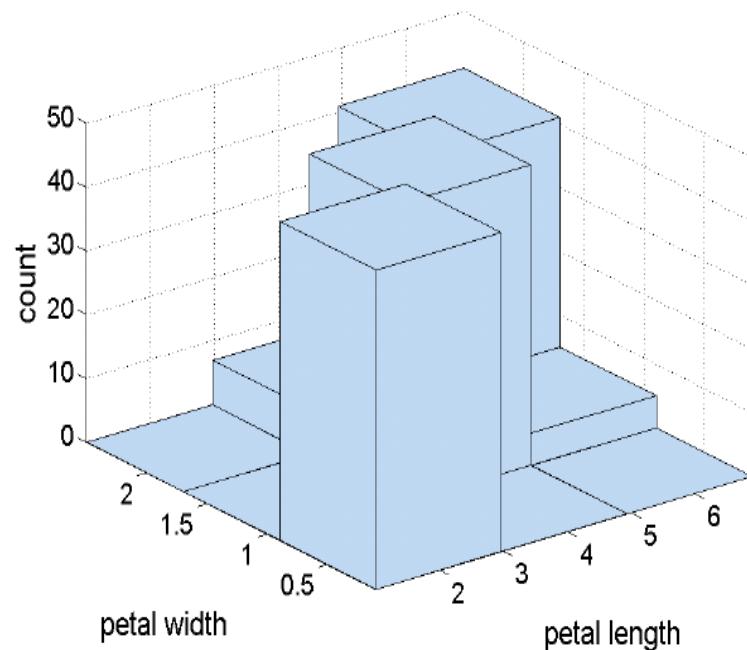
Histogram

- Representation of the distribution of numerical data
- It is an estimate of the probability distribution of a continuous variable
 - Empirical distribution
- Binning the range of values: dividing the entire range of values into a series of intervals
- Counting how many values fall into each interval
 - A rectangle is erected over the bin with height proportional to the frequency

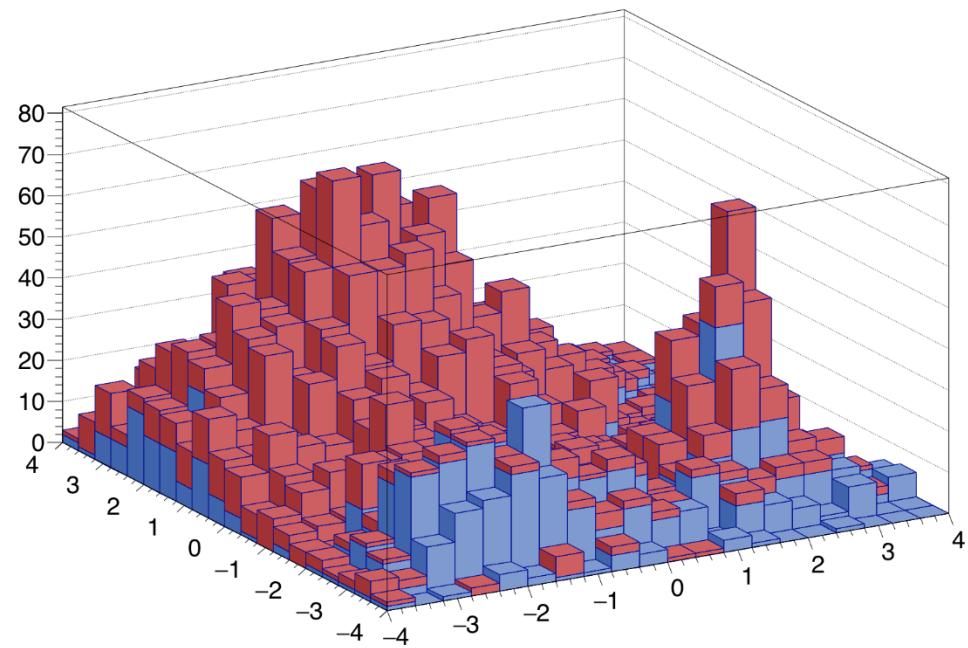


Two-dimensional histogram

- It estimates the joint distribution of two variables

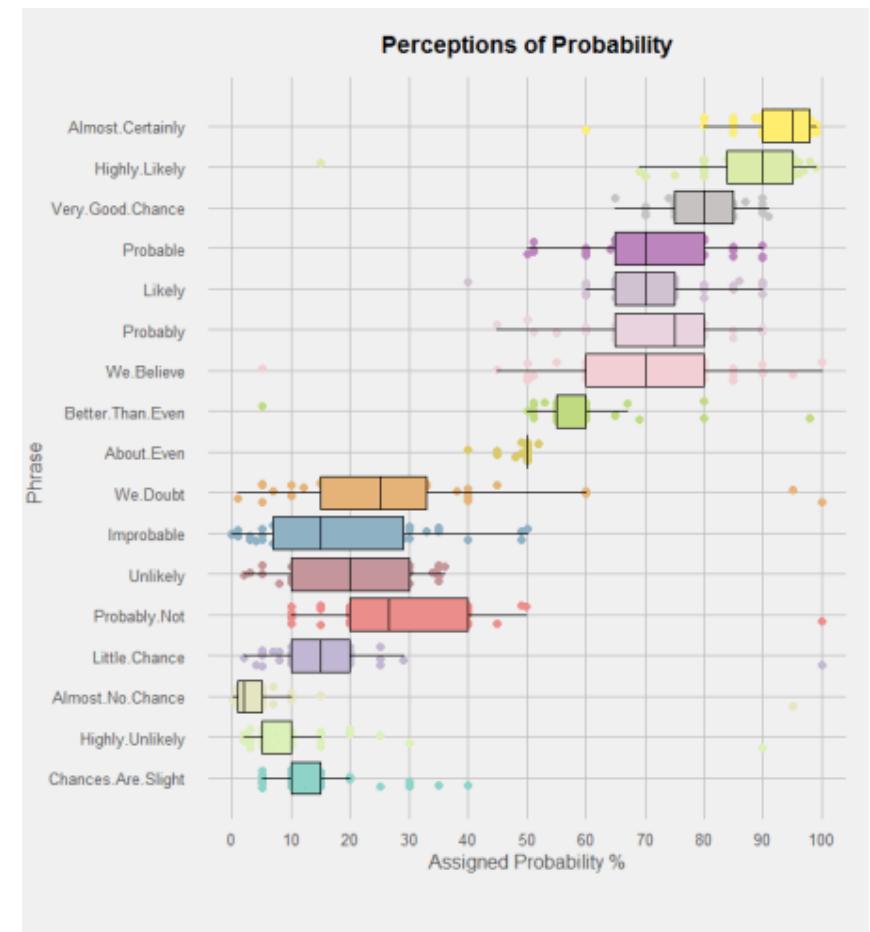
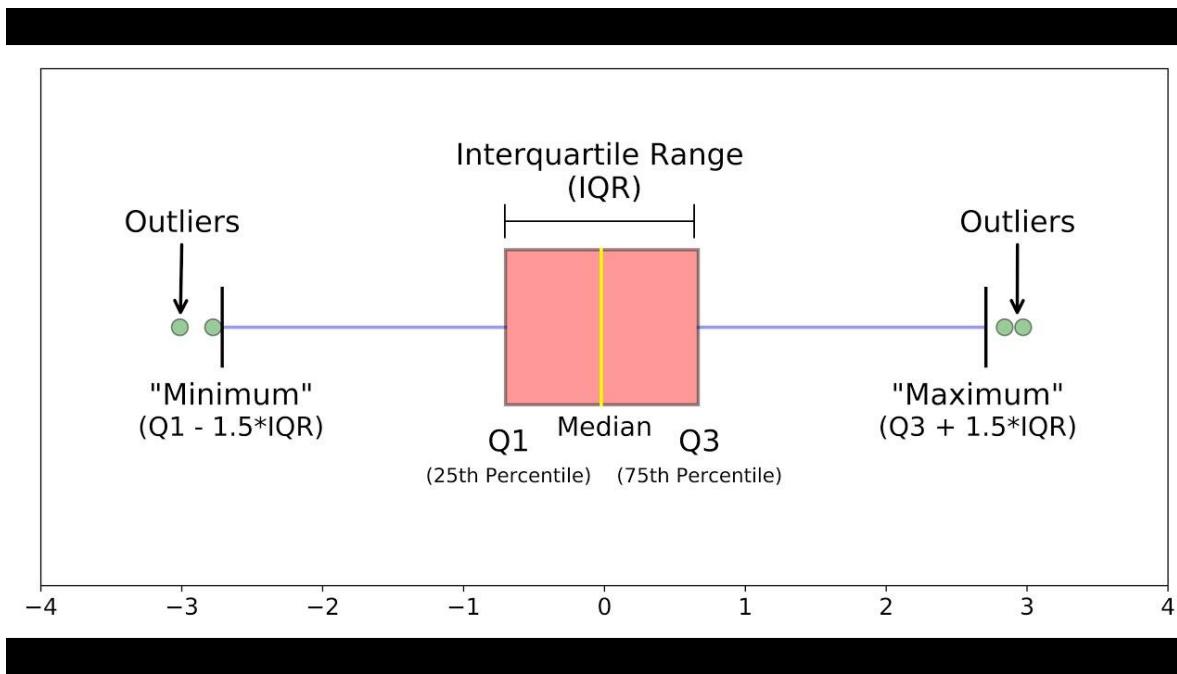


Stacked 2D histograms



Boxplot

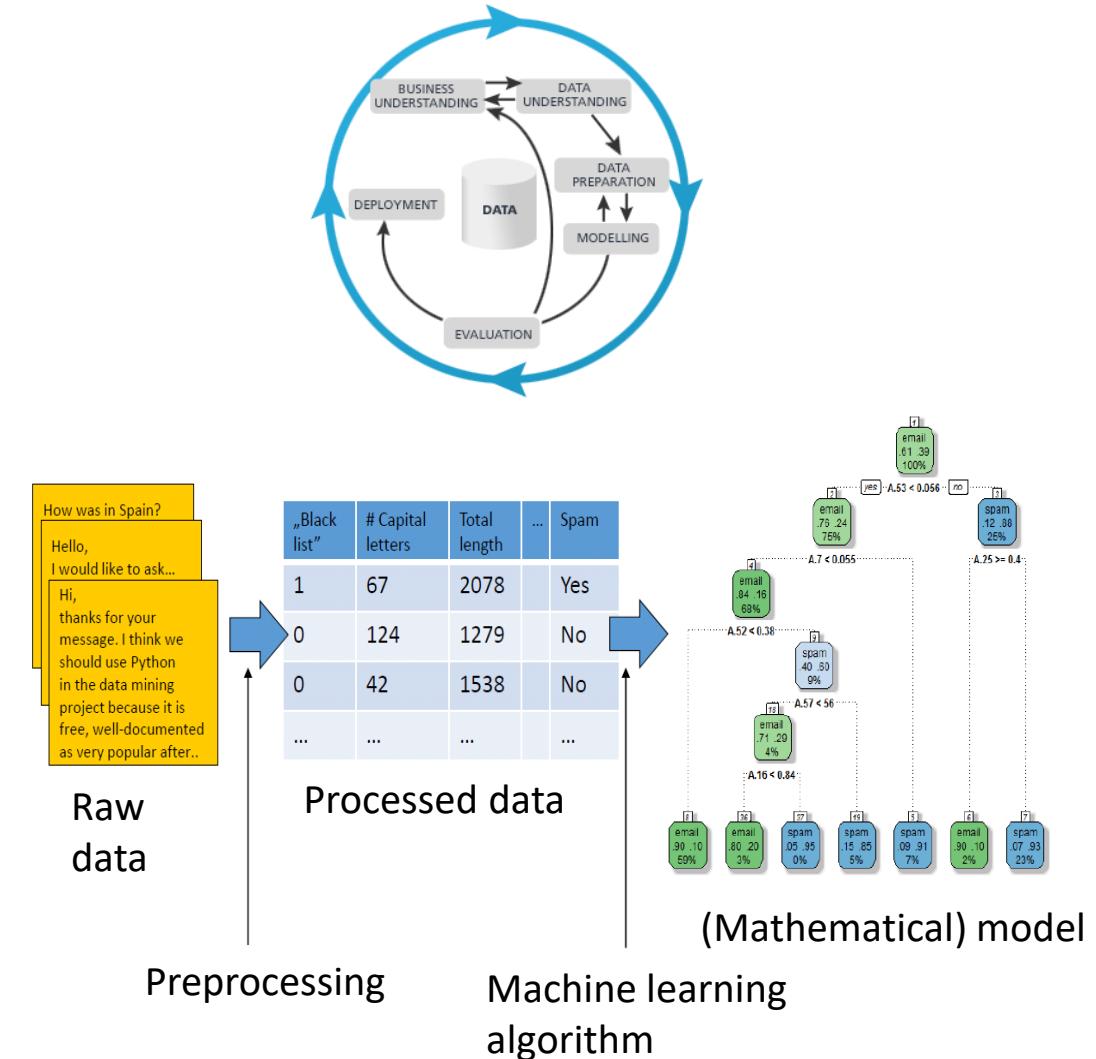
- Another method to visualize distribution
 - Attributed to J. Tukey



DP – Data preparation

The process of transforming and mapping data from one "raw" data form into another format with the intent of making it more appropriate and valuable for analytics.

We can estimate that 70% of resources (time, technology, personnel) used in the whole data science project are committed to DU + DP phases.



Data preparation

Values

Rows

Columns

Identifying
and
correcting
errors

Imputing
missing
values

Remove
duplicates

Detect
outliers

Feature
selection /
dimension
reduction

Introducing
new features

Transforming
features

Scaling
attributes

Discretization

One-hot-
encoding

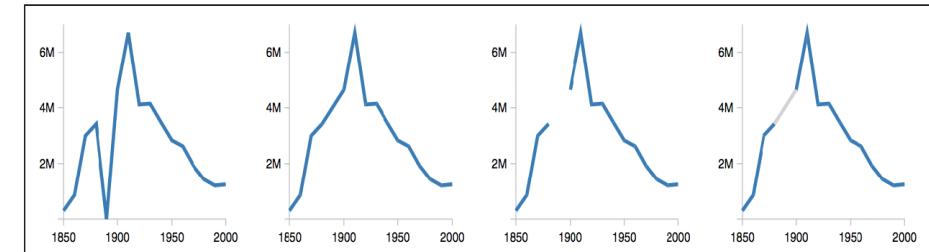
Common problems with values and rows

- Measurement errors
- Inconsistency (mile, m, km; Budapest, Bpest)
- Not plausible data
 - Everybody has a six-figure salary
 - Everybody gets an A+ from Data Science at AIT ☺
- No header
- Missing apostrophe from text fields
- Missing data
- Duplicates (recurring rows)
 - Sometimes not completely identical, e.g. same person with more very similar addresses
- Outliers: point that is distant from other observations
 - Not a problem by itself, but may be



How can the problems be fixed?

- Measurement error: can't be fixed but can be excluded from data if it is detected
- Missing values (data imputation):
 - Not necessarily a problem (perhaps that attribute is not interpreted/defined for every row)
 - We can remove the entire row of the missing value (not a good solution if we have many missing values)
 - We introduce a new global constant, e.g. an „unknown” label
 - We substitute the missing value with the column average (global column average or average with a given label)
 - We impute the missing value with a smart guess based on a machine learning model
- Duplicates: to detect the (nearly) identical observations
- Outlier:
 - Detecting the outlier can be the aim of the project (e.g. freud detection)
 - Sometimes outliers should be excluded
Sometimes outliers are organic part of the data and they should remain in the data



Data preparation

Values

Rows

Columns

Identifying
and
correcting
errors

Imputing
missing
values

Remove
duplicates

Detect
outliers

Feature
selection /
dimension
reduction

Introducing
new features

Transforming
features

Scaling
attributes

Discretization

One-hot-
encoding

Reducing the number of attributes

Aim: to have fewer columns

Why?

- Achieve faster running time
- Need less storage capacity
- Easier to visualize
- The results are easier to interpret
- In high dimension most of the models perform poorly (due to curse of dimensionality)

How?

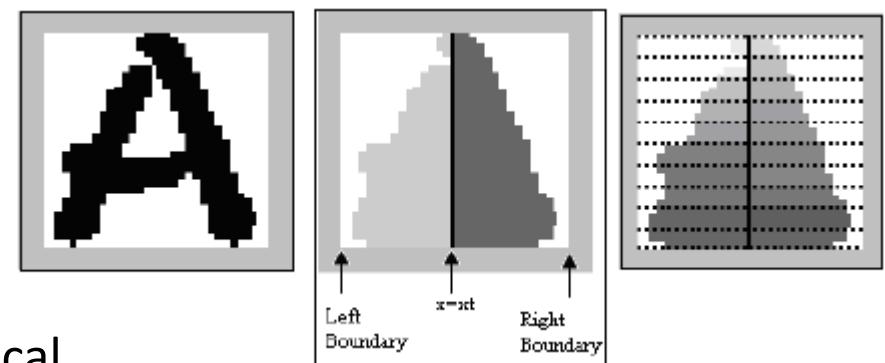
- Omit redundant columns
- Merging columns
- Introducing new (better) attributes and omitting the old ones
- Advanced dimension reduction methods (such as PCA)

Methods for reducing the dimensionality

- Finding redundant columns
 - E.g. Column A: price of the product, Column B: paid VAT
- Finding irrelevant columns
 - E.g. the phone number of a person regarding their creditworthiness (**are you sure?**)
- Automatic filtering
 - If the correlation of two columns is too high, we omit one of them
- Embedded methods
 - The used machine learning method chooses the relevant variables itself (later)
- Advanced dimension reduction methods (such as PCA)

Introducing new attributes

- Sometimes we don't necessarily want to reduce the number of attributes but we want better, more expressive attributes
- Sometimes domain knowledge is needed
- Examples:
 - To extract features from pixel series of images
 - Number of „on” pixels, average of the horizontal coordinates of the „on” pixels, variance of the vertical coordinates, correlation between the horizontal and vertical positions of „on” pixels
 - Combining attributes based on domain knowledge
 - Introducing density instead of volume and mass



Scaling attributes

- Feature scaling: sometimes it is necessary to standardize/normalize the range of independent variables (e.g. for visualization, for some machine learning algorithms)
 - Rescaling the range in $[0, 1]$ (min-max normalization) : $x' = \frac{x - \min(x)}{\max(x) - \min(x)}$
 - Affected by outliers
 - Useful when we don't know about the distribution
 - Standardization (zero-mean, unit-variance): $x' = \frac{x - \bar{x}}{s_x}$
 - Much less affected by outliers
 - Useful when the feature distribution is Normal

Other transformations – logarithmic transformation

- In some application we can take the logarithm of the attribute
 - Especially salary/income/wealth-related
 - It makes more sense to measure „percent” changes in wage rather than absolute changes
 - It is usually more normally distributed)
 - „Diminishing marginal utility”
 - Other (bijective) mappings of the attributes might also be useful



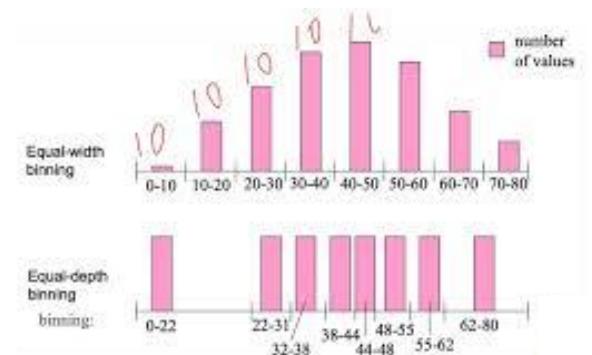
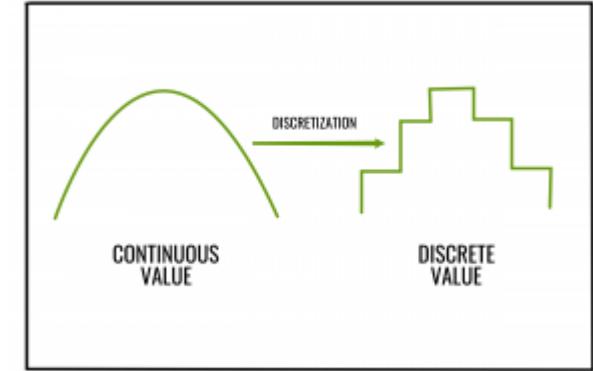
$$\log_b(MN) = \log_b(M) + \log_b(N)$$

$$\log_b\left(\frac{M}{N}\right) = \log_b(M) - \log_b(N)$$

$$\log_b(M^p) = p \log_b(M)$$

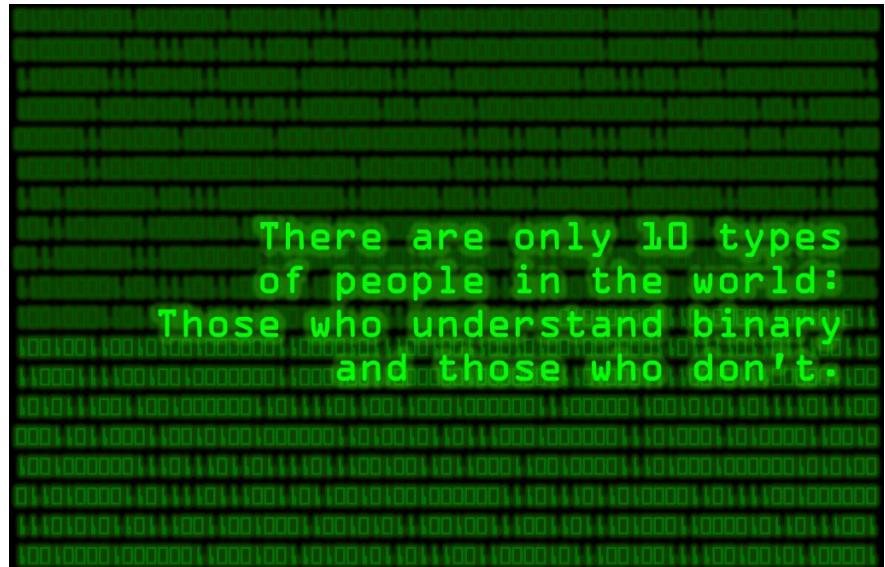
Discretization

- Goal: transferring continuous variables into discrete counterparts
- Why?
 - Some algorithms need discrete variables
 - We need to store less values, for some algorithms the running time is much faster
 - Sometimes a rougher scale is sufficient, e.g. high, medium, low values, the data could be more clear-out
- How to partition the data? What are the discretization cut points?
 - Divide the range of the continuous variable into intervals of the same length (equidistant division)
 - Dividing the range into intervals with the same number of observations (along quantile values) – equal frequency
 - Dividing along quantile values creating groups with differing size, e.g. along quantiles 10, 30, 70, 90
 - If there are some natural cut points where the data is rare, it is reasonable to choose these cut points



From categorial to binary: one-hot-encoding

- If a nominal attribute has k possible values, it is replaced by k synthetic binary attributes (one attribute-per-value approach or one-hot encoding)
 - The i th being 1 if and only if the original value corresponds to the i th group



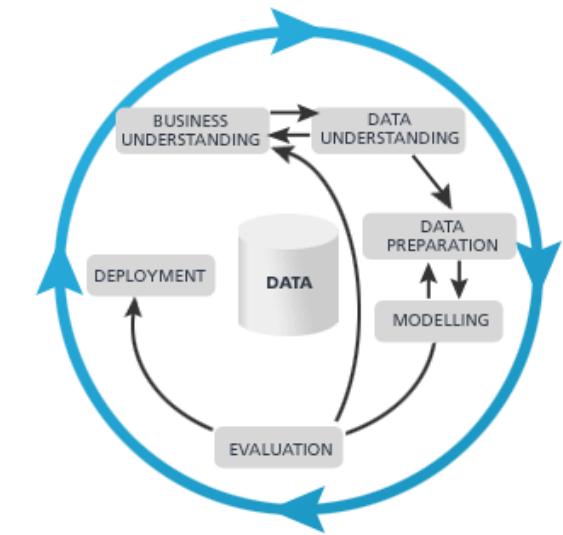
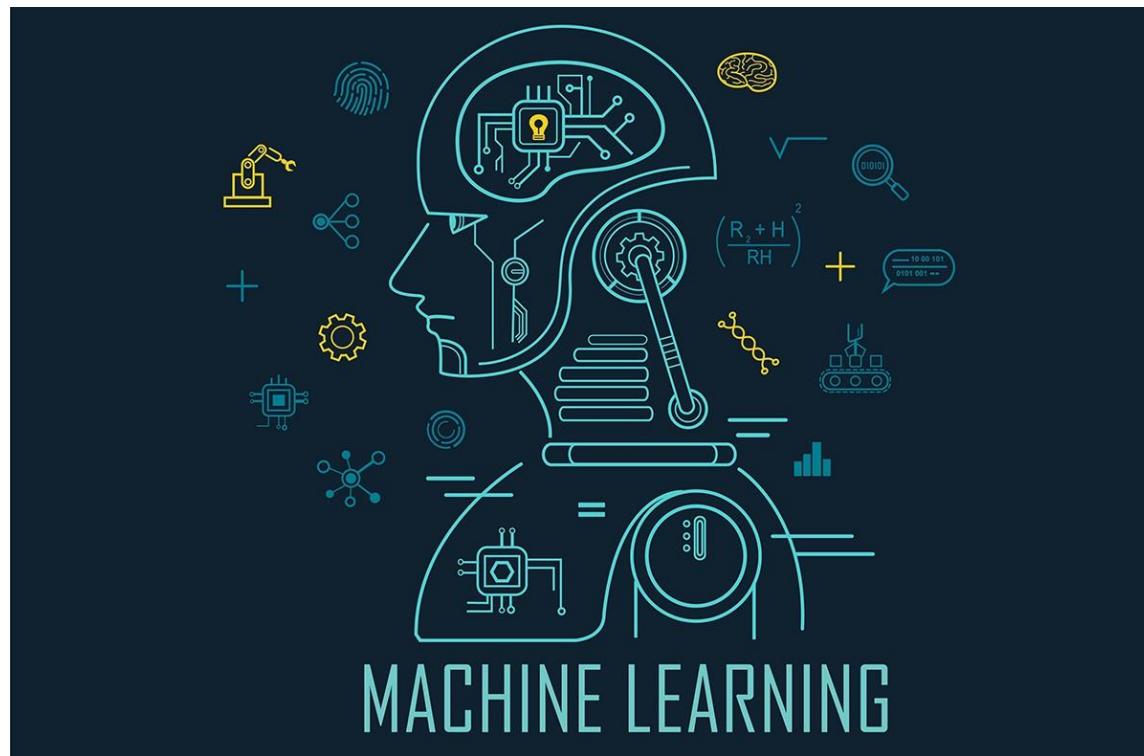
id	color
1	red
2	blue
3	green
4	blue

One Hot Encoding

id	color_red	color_blue	color_green
1	1	0	0
2	0	1	0
3	0	0	1
4	0	1	0

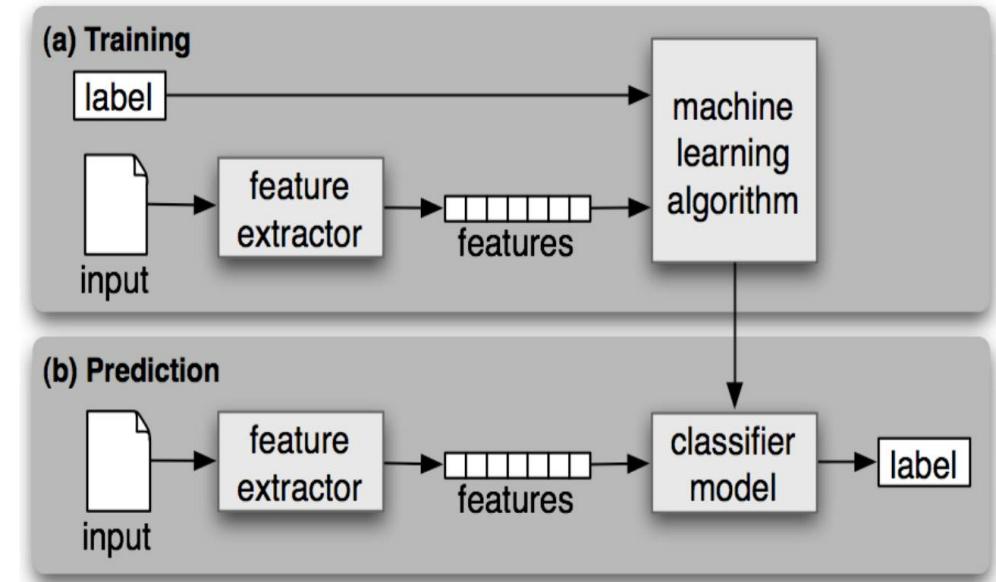
M-Modeling

Finding the best performing model, fitting the parameters



Supervised / unsupervised learning

- Supervised learning
 - We have a training set where the value of the target variable is known
 - Aim: based on the attributes predict the target when it is not known
 - Example: classification, regression
- Unsupervised learning
 - The target (label) is not known for any records (latent labels)
 - Aim: to associate useful labels to the records based on the attributes
 - In many cases our aim is to gain better understanding of the data or visualize the data
 - Example: clustering



Regression

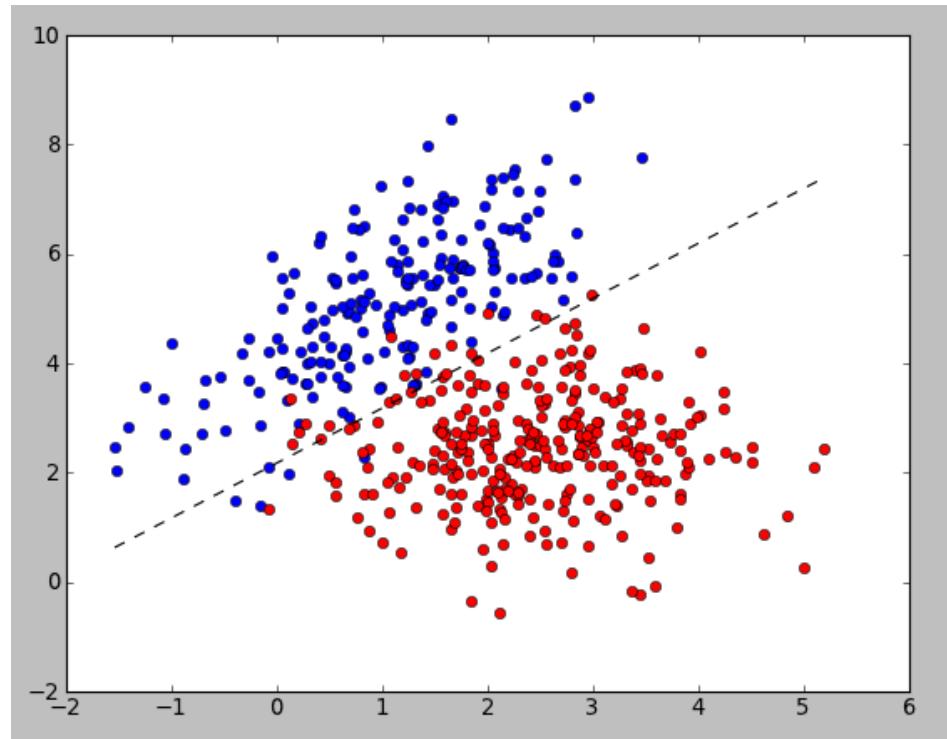
- We try to predict continuous valued output based on the values of other variables (via supervised learning)
- Examples:

Explanatory (input) variables	Target (output)
Number of rooms, size, location (ZIP code), ...	Market value of a house
Movie budget, film genre, popularity of the actors (based on their IMDB pages)	Box office result of a movie
Major, admission point score, gender, age, GMAT scores,	GPA

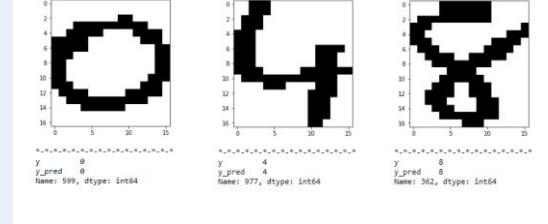
- Challenges: finding the right explanatory variables, the most suitable functional form/modelling approach

Classification

- We try to predict discrete (sometimes binary) valued output based on the values of other variables (via supervised learning)
- It is also possible to do „classification via regression”
- Challenges: finding the right explanatory variables, the most suitable modeling approach, fitting the parameters of the model



Classification - examples

Input variables (features)	Target variable
Purchase history, age, gender	Should we send a targeted advertisement message to a customer? (0/1)
Number of „on” pixels, average of the horizontal coordinates of the „on” pixels, variance of the horizontal coordinates, correlation between the horizontal and vertical positions of „on” pixels, ...	Handwritten digit recognition (0/1/2/3/4/5/6/7/8/9) 
Salary, marital status, address, profession, qualification, ...	Is the customer creditworthy? (0/1)
Words/n-grams appearing in the e-mail, subject of the mail, sender, number of receivers, ...	Is the email spam? (0/1)
Age, gender, profession, qualification, contents liked on Facebook, ...	Psychological profiles/ temperaments (e.g.: sanguine, phlegmatic, choleric, and melancholic)

Fundamental task of regression

Let $X = (X_1, X_2, \dots, X_p)$ be the feature vector and Y is the target variable.

Regression: we suppose that there is a relationship between X and Y , in general: $Y = f(X) + \epsilon$, where ϵ (the random error) is independent from X and has zero mean

Aim: giving prediction: $\hat{Y} = \hat{f}(X)$

In reality \hat{f} is sometimes considered to be a black-box, we are not interested in the functional form, but in giving accurate enough prediction for Y

Learning: On the labeled data of the training set we estimate the function f , minimizing the „prediction error” on the training set

Prediction: using \hat{f} for data that we have not seen before $\hat{Y} = \hat{f}(X)$

Fundamental task for classification

Let $X = (X_1, X_2, \dots, X_p)$ be the feature vector and Y is the target variable.

For classification problems: $Y \in \{c_1, c_2, \dots, c_k\}$

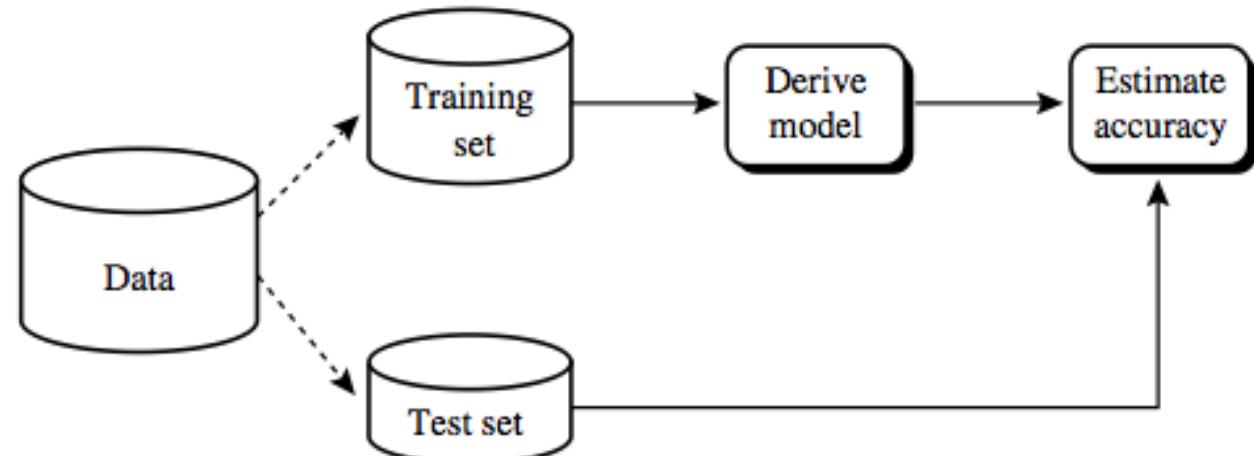
The real $p(X, Y)$ joint distribution (background distribution) is not known

Aim: finding f such that $P(Y = f(X))$ is maximal.

Learning: On the labeled data of the training set (independent identically distributed sample from the $p(X, Y)$) we estimate the function f , minimizing the „classification error” on the training set

Generalization ability

- Purpose: to build a model that predicts the target variable well in general not just on the available data set → good generalization ability
- Dataset is divided into two (or later three) parts
- Cross validation: later
- To evaluate models, a numerical „goodness” notion is needed



Training and test set

Spliting the data set into two parts

- Training set: fitting the model (i.e. optimizing its parameters) on the training set in such a way that it has a good performance on the training set and has a good generalization ability
- Test set: we test the model performance on data that were not seen by the model before
 - We choose the model that has the best performance on the test set



Training Data

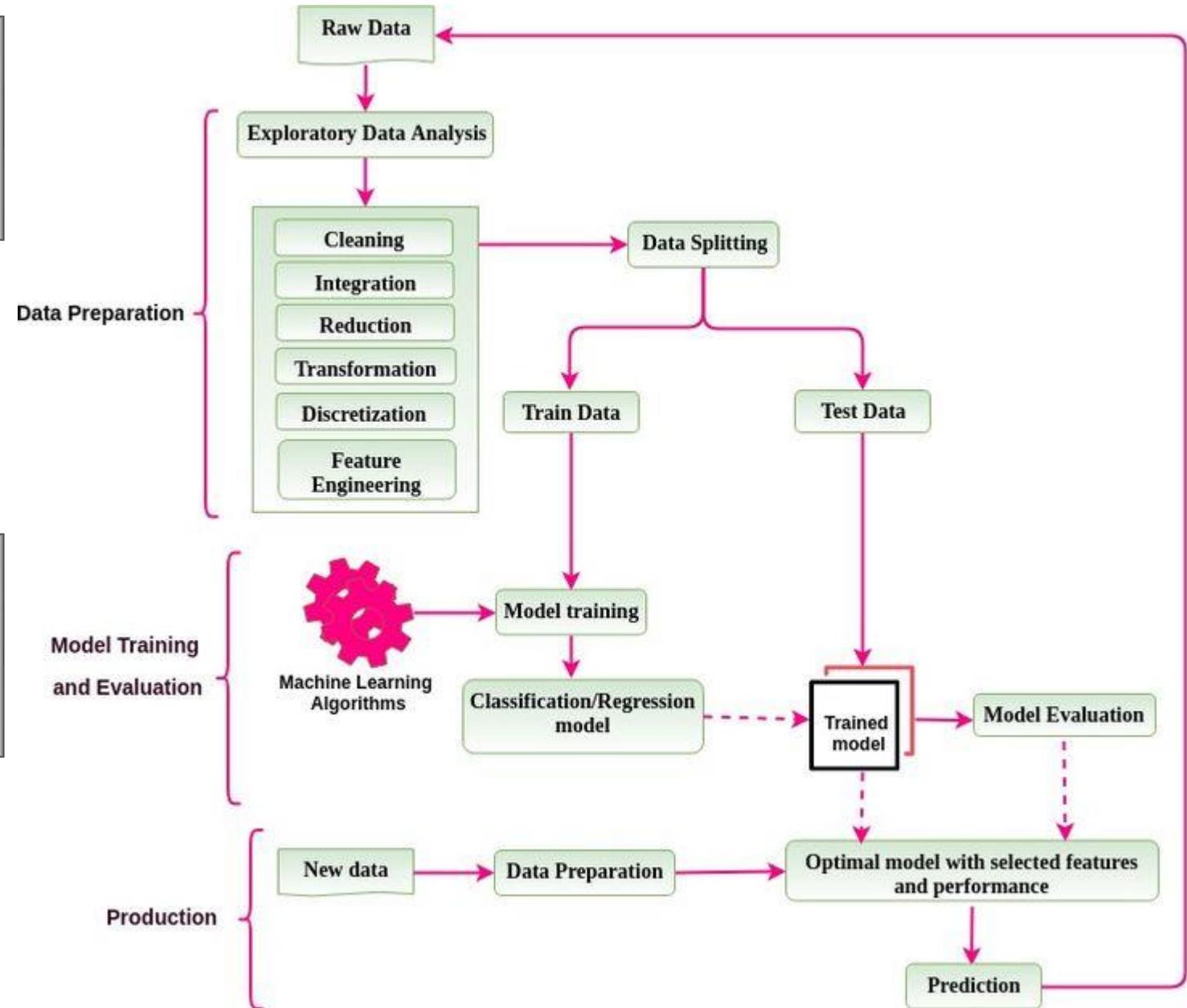
Test Data

E -Evaluation

- Evaluating the model. How does it perform? Is it good enough to achieve our goal?

D -Deployment

- Implementing the model, embedding it to the system. Communicating the results. Writing the report/research paper.



Requirements for successful data science projects

- Having domain knowledge or consulting with domain experts
- Big data (many observations)
 - Less likely to retrieve connections that is just in the data due to chance
 - (It can be computationally expensive!)
- Many features
 - Simple analytics bears with few features
- Clean data
 - Bad data encumber data analysis or leads to false results
 - GIGO: garbage in, garbage out

Requirements for successful data science projects II.

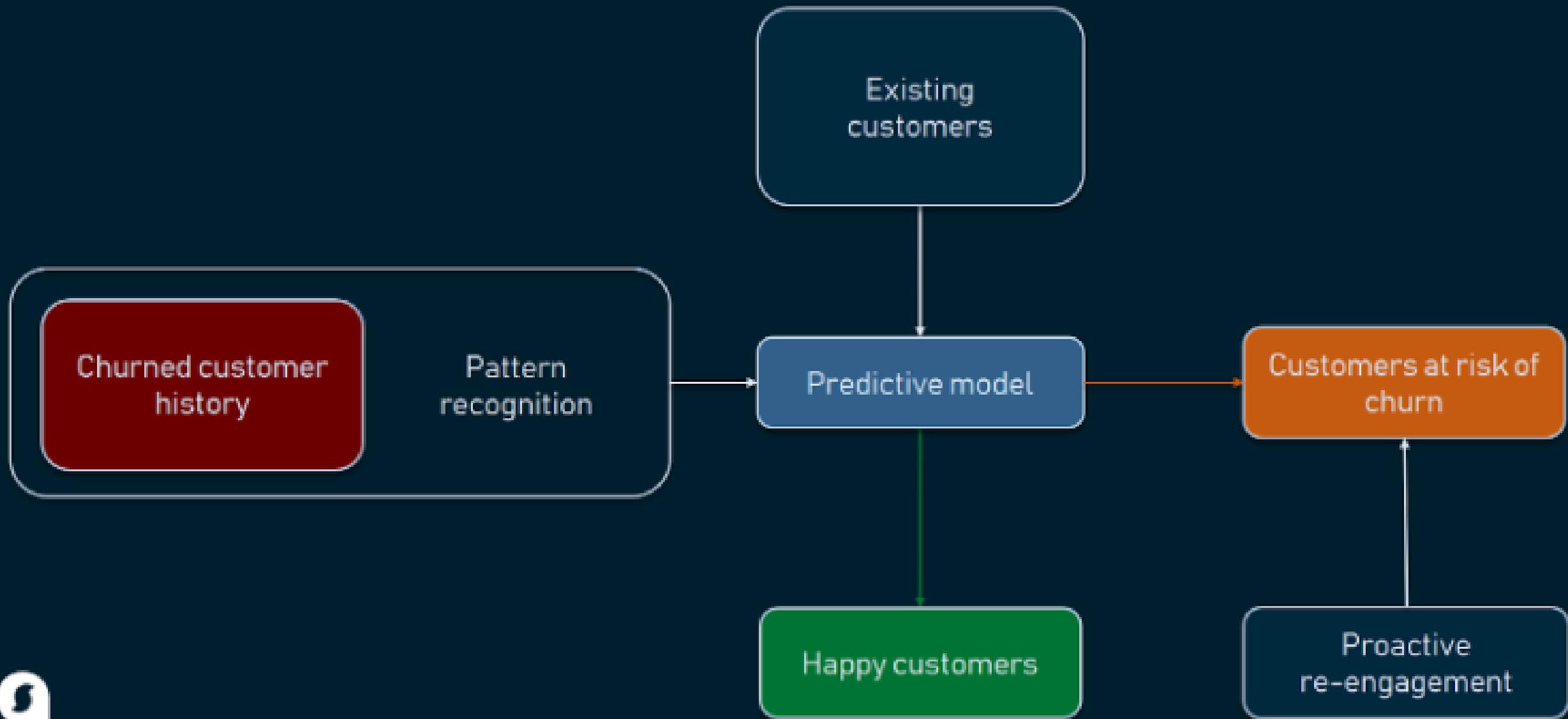
- Unbiased data
 - The data (the sample) should be representative to the population itself
 - BIBO: bias in, bias out
- The capacity of act
 - Sometimes the knowledge is discovered, but it will not go into action (high costs, too rigid system)
- Measurability of Return of Investment (ROI)
 - It defines the success of a project

Case study – customer churn detection in the telecommunication sector

- Churn: occurs when customers unsubscribed or cancel their service contract
- A telecommunication company approached our (imaginary) data science consulting company to predict which customers are at risk of leaving our business
 - Customer retention campaign targeted on at-risk customers
 - Offering coupons or discounts to those most likely to churn

1. How would you formulate the task as a data science problem?
2. Plan the analysis based on the CRISP-DM methodology!
3. Do you think that the requirements of a successful data science project are met?

CHURN RATE PREDICTION WITH MACHINE LEARNING



Customer churn prediction

- BU
 - business objective is reducing customer churn by identifying potential churn candidates beforehand, and take proactive actions to make them stay
- DU
 - Personal data about the customers (age, address, ...)
 - Information about their subscription plan
 - Call/text/data logs (who?, when? how much? etc.)
- DP
 - Feature engineering, transforming features etc.
- M
 - Binary classification problem (supervised learning)
- E
 - Test the performance of the model. Is it good enough to deploy?
- D
 - Design a retention campaign (probably with A/B testing)

What about the success requirements?

Acknowledgement

- András Benczúr, Róbert Pálovics, SZTAKI-AIT, DM1-2
- Krisztián Buza, MTA-BME, VISZJV68
- Bálint Daróczy, SZTAKI-BME, VISZAMA01
- Judit Csima, BME, VISZM185
- Gábor Horváth, Péter Antal, BME, VIMMD294, VIMIA313
- Lukács András, ELTE, MM1C1AB6E
- Tim Kraska, Brown University, CS195
- Dan Potter, Carsten Binnig, Eli Upfal, Brown University, CS1951A
- Erik Sudderth, Brown University, CS142
- Joe Blitzstein, Hanspeter Pfister, Verena Kaynig-Fittkau, Harvard University, CS109
- Rajan Patel, Stanford University, STAT202
- Andrew Ng, John Duchi, Stanford University, CS229



Schedule of the semester

	<i>Monday midnight</i>	<i>Tuesday class</i>	<i>Friday class</i>
W1 (02/06)			
W2 (02/13)		HW1 out	
W3 (02/20)			
W4 (02/27)	HW1 deadline	HW2 out	
W5 (03/06)	PROJECT PLAN		
W6 (03/13)	HW2 deadline	HW3 out	
W7 (03/20)			MIDTERM
SPRING BREAK		SPRING BREAK	SPRING BREAK
W8 (04/03)	HW3 deadline	HW4 out	GOOD FRIDAY
W9 (04/10)	MILESTONE 1		
W10 (04/17)	HW4 deadline		
W11 (04/24)			
W12 (05/01)	MILESTONE 2		
W13 (05/08)			
W14 (05/15)		FINAL	PROJECT presentations
W15 (05/22)		PROJECT presentations	

Similarity, dissimilarity

- For many algorithms, it is necessary to quantify the (dis)similarity of two records (rows)
 - E.g. the goal of clustering is to group similar objects together
- There are several (dis)similarity measures
- The used metric depends on the feature vector, what type of attributes (continuous, categorical, etc.) the row consists of

Similarity vs. dissimilarity

Similarity

The higher it is, the more similar the records are

Usually symmetric

Typically takes values from $[0, 1]$ (or $[0, \infty]$)

Dissimilarity

The higher it is, the more dissimilar the records are

Usually symmetric

Usually the value is 0 if the records are identical

Distance metric

- A special case of dissimilarity measures is distance (or metric):
 $d : X \times X \rightarrow [0, \infty)$

1. $d(x, y) \geq 0$ (non-negativity)
2. $d(x, y) = 0 \Leftrightarrow x = y$ (identity of indiscernibles)
3. $d(x, y) = d(y, x)$ (symmetry)
4. $d(x, z) \leq d(x, y) + d(y, z)$ (triangular inequality or subadditivity)

where $x, y, z \in X$

- We don't necessarily require dissimilarity measures to satisfy the above axioms of distance metric but in many cases we work with distance functions (i.e. measures that satisfy the above conditions)

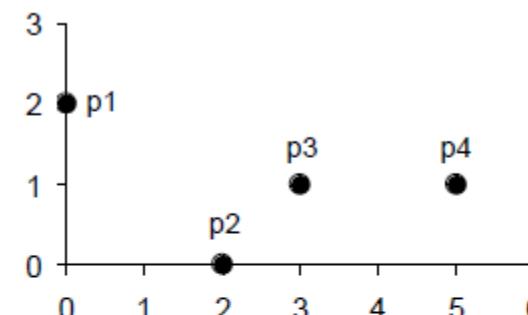
Categorical attributes

- Similarity: 1 if they are equal, 0 otherwise
 - Dissimilarity: just the opposite (0 if they are equal, 1 otherwise)
- Sometimes there are categories that are „closer” to each other than others, so the similarity function can be non-binary as well
 - We can use a scoring matrix that codes the similarities between the categories
 - E.g. professions

Euclidean distance

- The most frequently used distance if the records can be interpreted as points in the n -dimensional space (numerical attributes)
- $p = (p_1, \dots, p_n)$ and $q = (q_1, \dots, q_n)$ are two points in the space, their Euclidean distance:

$$d(p, q) = \sqrt{\sum_{i=1}^n (p_i - q_i)^2}$$



point	x	y
p1	0	2
p2	2	0
p3	3	1
p4	5	1

- It is practical to rescale the range (max-min normalization, standardization)

	p1	p2	p3	p4
p1	0	2.828	3.162	5.099
p2	2.828	0	1.414	3.162
p3	3.162	1.414	0	2
p4	5.099	3.162	2	0

Minkowski distance

- Minkowski-distance (L_p distance): generalization of Euclidean distance
- $p = (p_1, \dots, p_n)$ and $q = (q_1, \dots, q_n)$ are two points, their Minkowski distance:

$$d(p, q) = \sqrt[r]{\sum_{i=1}^n |p_i - q_i|^r}$$

- If $r \geq 1$ it is a distance metric
- The greater the r parameter is, the more important feature scaling is
- Special cases:
 - $r = 1$, Manhattan distance
 - $r = 2$, Euclidean distance
 - $r = \infty$, Chebisev distance

It can also be weighted:

$$d(p, q) = \sqrt[r]{\sum_{i=1}^n w_i |p_i - q_i|^r}$$

Special cases of Minkowski distance

- Manhattan distance (L_1 distance, taxicab distance):

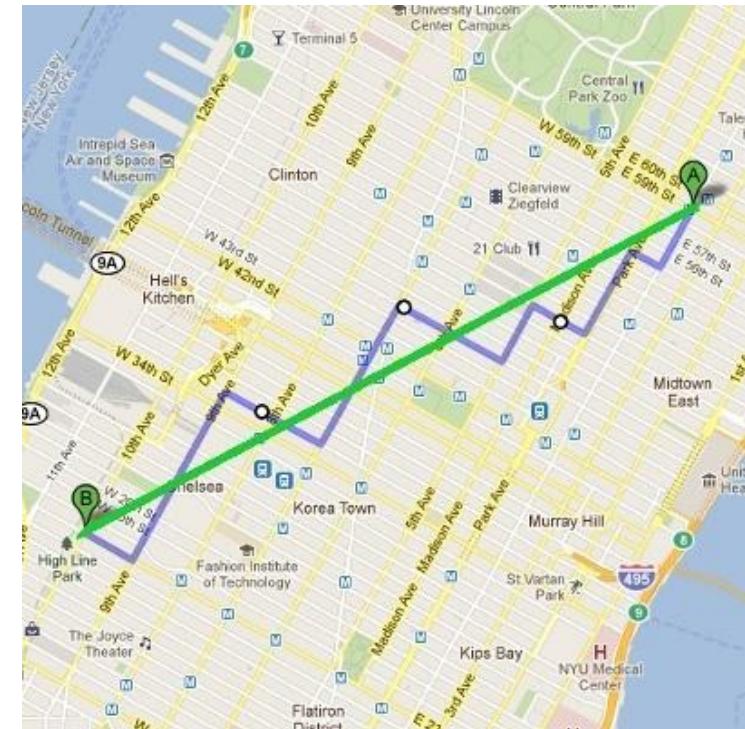
- The Manhattan distance of points (1,2) and (7,0) is 8, they are 8 „blocks” from each other

$$\sum_{i=1}^n |p_i - q_i|$$

- Euclidean distance (L_2)
- Chebyshev distance (L_{\max} , L_{∞})
 - Two equivalent definitions:

$$\lim_{r \rightarrow \infty} \sqrt[r]{\sum_{i=1}^n |p_i - q_i|^r}$$

$$\max_{i \in \{1, \dots, n\}} |p_i - q_i|$$



Problem

- Prove the following statements:

a) $L_1(p, q) = \sum_{i=1}^d |p_i - q_i|$ is a distance metric

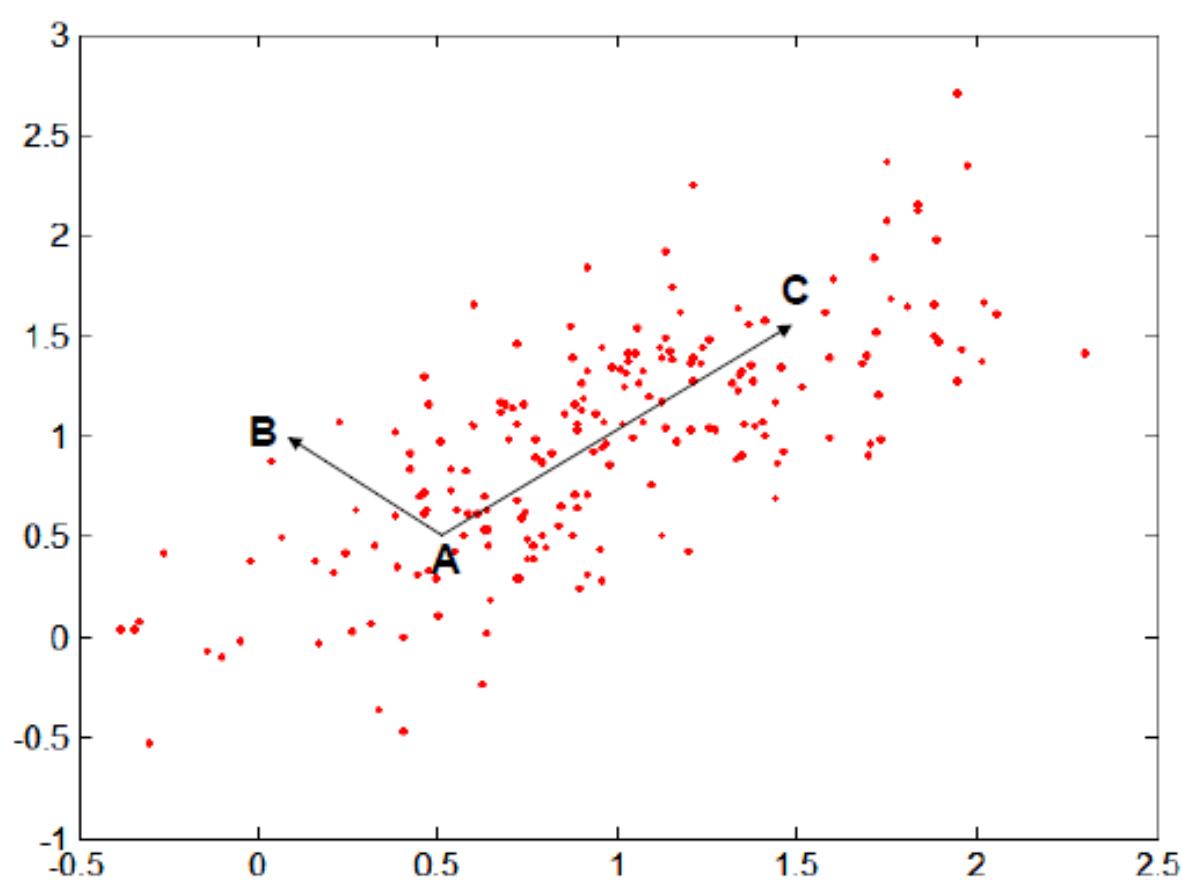
b) $L_2^2(p, q) = \sum_{i=1}^d (p_i - q_i)^2$ is NOT a distance metric,

Mahalanobis distance

- Minkowski distance does not take into consideration that attributes are not necessarily independent from each other
 - In the extreme case, we can have two identical columns, then it counts double in the distance
 - A possible solution is that we filter out these attributes (reducing the number attributes)
 - Or we can use a distance that compensate for the bias coming from the correlatedness of the attributes
- $Mahal(p, q) = \sqrt{(p - q)\Sigma^{-1}(p - q)^T}$
- Where Σ is the sample covariance matrix, that is:

$$\Sigma_{jk} = \frac{1}{m - 1} \sum_{i=1}^m (X_{ij} - \bar{X}_j)(X_{ik} - \bar{X}_k)$$

Mahalanobis distance, an example



- Covariance matrix:
- $\Sigma = \begin{pmatrix} 0.3 & 0.2 \\ 0.2 & 0.3 \end{pmatrix}$
- A: (0.5, 0.5)
- B: (0, 1)
- C: (1.5, 1.5)
- $Mahal(A, B) = \sqrt{5}$
- $Mahal(A, C) = \sqrt{4}$

Problem

We are given a dataset with two attributes (X and Y) and the following covariance matrix:

$$\Sigma = \begin{bmatrix} 0.3 & 0.2 \\ 0.2 & 0.3 \end{bmatrix}$$

1. Calculate the $\text{Corr}(X, Y)$ correlation between the attributes!
2. Calculate the $\text{Mahal}(A, B)$ Mahalanobis distance between two data points A and B , where $A = [0.5 \ 0.5]$, $B = [0 \ 1]$

Problem

Let two feature vectors contain the following attributes:

- person's height (between 5 and 6 feet)
- person's weight (between 90 and 260 lbs)
- person's annual income (between 10,000 and 1 Million dollars)

How would you calculate the distance between the two vectors? What kind of transformations would you apply, and which distance would you chose?

Cosine similarity

- Compares the direction of vectors (the magnitude does not play a role here)
 - The similarity of two vectors in the same direction is 1
 - The similarity of two orthogonal (perpendicular) vector is 0
 - The similarity of two vectors in opposite direction is -1

$$\cos(p, q) = \frac{p \cdot q}{\|p\| \cdot \|q\|}$$

- It is mostly used for high dimensional positive vectors
 - E.g. for document-term matrices with frequencies
- We can form dissimilarity as well: $1 - \cos(p, q)$
 - It is not a distant metric (triangular inequality is not satisfied)

Cosine similarity - example

$$d_1 = 3 \ 2 \ 0 \ 5 \ 0 \ 0 \ 0 \ 2 \ 0 \ 0$$

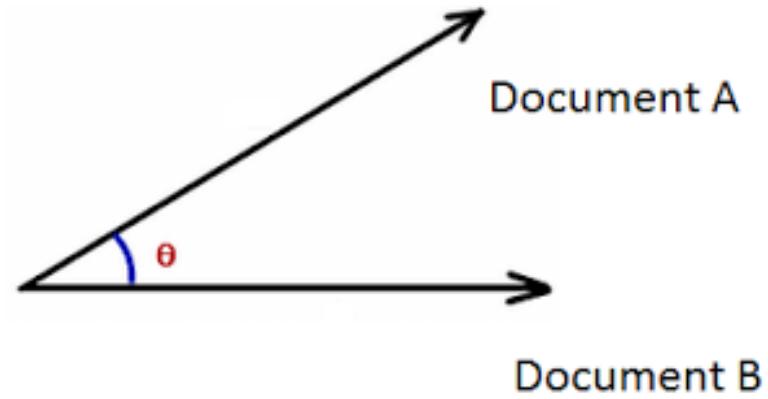
$$d_2 = 1 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 1 \ 0 \ 2$$

$$d_1 \cdot d_2 = 3 \cdot 1 + 2 \cdot 0 + 0 \cdot 0 + \dots + 2 \cdot 1 + 0 \cdot 0 + 0 \cdot 2 = 5$$

$$\|d_1\| = (3^2 + 2^2 + 5^2 + 2^2)^{0.5} = 42^{0.5} = 6.48$$

$$\|d_2\| = (1^2 + 1^2 + 2^2)^{0.5} = 6^{0.5} = 2.45$$

$$\cos(d_1, d_2) = \frac{d_1 \cdot d_2}{\|d_1\| \cdot \|d_2\|} = \frac{5}{6.48 \cdot 2.45} = 0.31$$



Similarity of binary vectors

- Binary data frequently correspond to sparse data matrices (almost all entries are 0s and there are few 1s)
 - E.g. document term matrix, transaction matrix
- The previously studied (dis)similarities are not informative in this case, every record looks similar
 - Special (dis)similarity notions are needed
- Let p and q be binary vectors of length n (the entries are 0 or 1)

Simple Matching Coefficient (SMC)

- SMC: simple matching coefficient

$$\begin{aligned} \text{SMC} &= \frac{\text{number of matching attributes}}{\text{number of attributes}} = \\ &= \frac{M_{00} + M_{11}}{M_{00} + M_{01} + M_{10} + M_{11}} \end{aligned}$$

		<i>p</i>	
		0	1
<i>q</i>	0	<i>M</i> ₀₀	<i>M</i> ₁₀
	1	<i>M</i> ₀₁	<i>M</i> ₁₁

- SMD (simple matching distance) = $1 - \text{SMC}$
- Practical to use if the „information content” of 0 and 1 are equivalent (symmetric), i.e. the data matrix is not sparse

Jaccard index

- Jaccard index (similarity coefficient)

$$J = \frac{M_{11}}{M_{01} + M_{10} + M_{11}}$$

		<i>p</i>	
		0	1
<i>q</i>	0	M_{00}	M_{10}
	1	M_{01}	M_{11}

- The common 0s do not play a role
- It is practical for sparse data matrix
- Jaccard distance = $1 - J$

Example: SMC vs. Jaccard

$p = 1000000000$

$q = 0000001001$

$M_{00} = 7$ (the number of attributes where p is 0 and q is 0)

$M_{01} = 2$ (the number of attributes where p is 0 and q is 1)

$M_{10} = 1$ (the number of attributes where p is 1 and q was 0)

$M_{11} = 0$ (the number of attributes where p is 1 and q was 1)

$$\text{SMC} = \frac{M_{00} + M_{11}}{M_{00} + M_{01} + M_{10} + M_{11}} = \frac{7}{10} = 0.7 \quad J = \frac{M_{11}}{M_{01} + M_{10} + M_{11}} = \frac{0}{3} = 0$$

Consider the following three documents:

- d_1 : “ant bee”
- d_2 : “dog bee hog ant”
- d_3 : “cat gnu dog eel fox”

Problem

These documents can be represented by 8-dimensional vectors in a so-called *document-term matrix*, which describes the frequency of terms that occur in a collection of documents:

	ant	bee	cat	dog	eel	fox	gnu	hog
d_1	1	1	0	0	0	0	0	0
d_2	1	1	0	1	0	0	0	1
d_3	0	0	1	1	1	1	1	0

1. Calculate simple matching coefficient (SMC) and Jaccard-coefficient of d_1 and d_2 .
2. Determine distances derived from these coefficients. Which is the better coefficient to handle the problem? Why?

Note.: Of course, this approach cannot distinguish “John is quicker than Mary” and “Mary is quicker than John” documents.

Problem

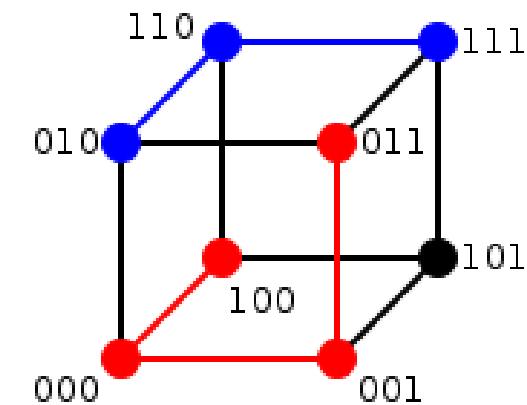
The following table's rows correspond to customers (A, B, C), and the columns correspond to products (a, b, \dots, h). The table contains 1 if a given costumer bought the given item, 0 otherwise. Determine the Jaccard similarity and the Cosine similarity of A and B .

	a	b	c	d	e	f	g	h
A	1	1	0	1	1	0	1	1
B	0	1	1	1	1	1	1	0
C	1	0	1	1	0	1	1	1

Hamming distance

- $H = M_{01} + M_{10}$
 - It is equal to the number of ones in $(p \text{ XOR } q)$
- If p and q are considered to be points in R^n , then p and q are the vertices of an n -dimensional hypercube
- The Hamming distance corresponds to the Manhattan distance of the vertices
 - The Hamming distance of 010 and 111 is 2
 - The Hamming distance of 100 and 011 is 3

		p
	0	1
q	0	M_{00}
1	M_{01}	M_{11}



Problem

Let $tf_{i,j}$ denote the entry in the i -th row and j -th column of the document-term matrix from the previous task. For instance, $tf_{1,1} = 1$, where row = d_1 and column = ant. Consider the following *tf–idf* transformation.

- Let m be the number of documents.
- Let df_j denote the *document frequency*, i.e. the number of non-zero elements in the j -th column (number of documents containing the j -th word). E.g. $df_1 = 2$.
- Let idf_j denote the *inverse document frequency*, which is defined as follows:

$$idf_j = \log \left(\frac{m}{df_j} \right)$$

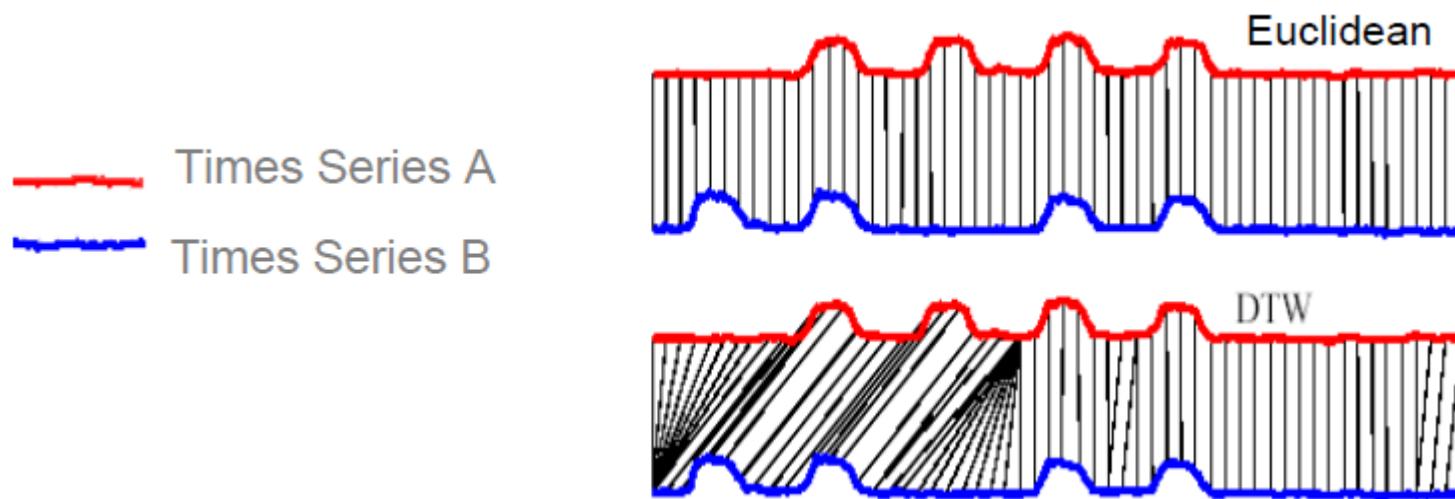
With these notations the *tf–idf* transformation is defined as follows:

$$tf\text{-}idf_{i,j} = tf_{i,j} \cdot idf_j = tf_{i,j} \cdot \log \left(\frac{m}{df_j} \right)$$

The tf–idf abbreviation stands for term frequency – inverse document frequency. What is the impact of this transformation? In case of a concrete, real document-term matrix, what could be the purpose of this transformation?

Distance of time series

- (In this course there will be little emphasis on time series.)
- Time series: a series of data points indexed in time order (a sequence taken at successive equally spaced points in time)
- First approach: we compare the time series index wise (e.g. using Minkowski distance where the coordinates correspond to indices)
 - A problem arises if there is any discrepancy in the alignment of the signals
 - E.g the signal is stretched or compressed compared to the other
 - How do you decide which points to compare with each other?
- A more sophisticated method: dynamic time warping (DTW)



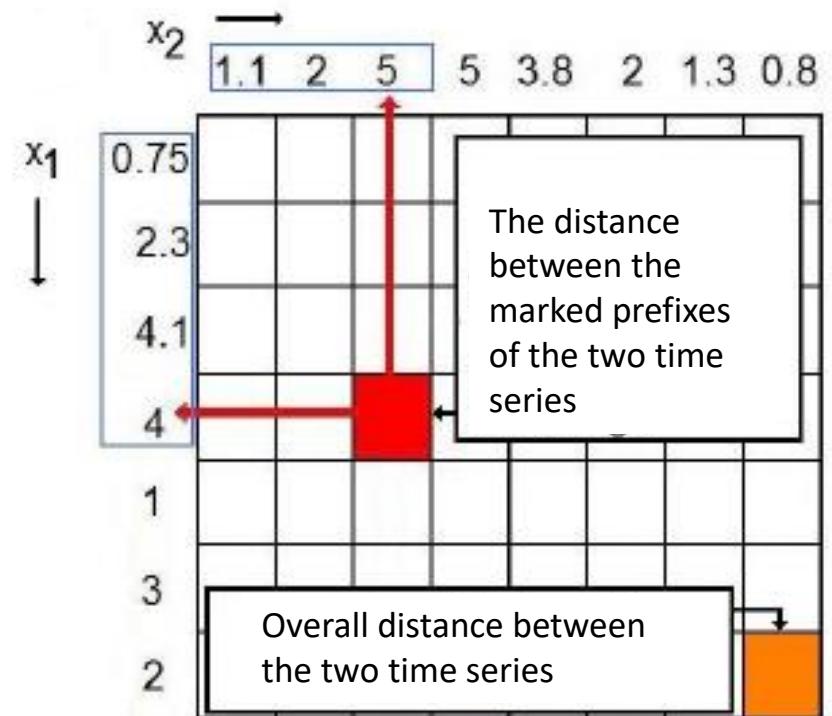
Dynamic Time Warping (DTW)

- In text mining: Levenshtein distance, in bioinformatics Smith-Watermann distance
- It is insensitive to local compression and stretches
- An edit distance that calculates the „cost” to transform a time series to another one
- First a distance measure is needed, comparing the corresponding elements of the two sequences, in the simplest form:

$$c(x[i], y[j]) = |x[i] - y[j]|$$

DTW II.

- We have to define the cost of insertion/deletion: c_i, c_d
 - In the simplest case $c_i = c_d = 0$
 - To calculate the distance, we fill the entries of a matrix (using dynamic programming)
 - We write one of the time series above the first row of the matrix and write the second time series next to the first column
 - The values in each entry of the matrix correspond to DTW distance of the corresponding prefixes of the time series



DTW III.

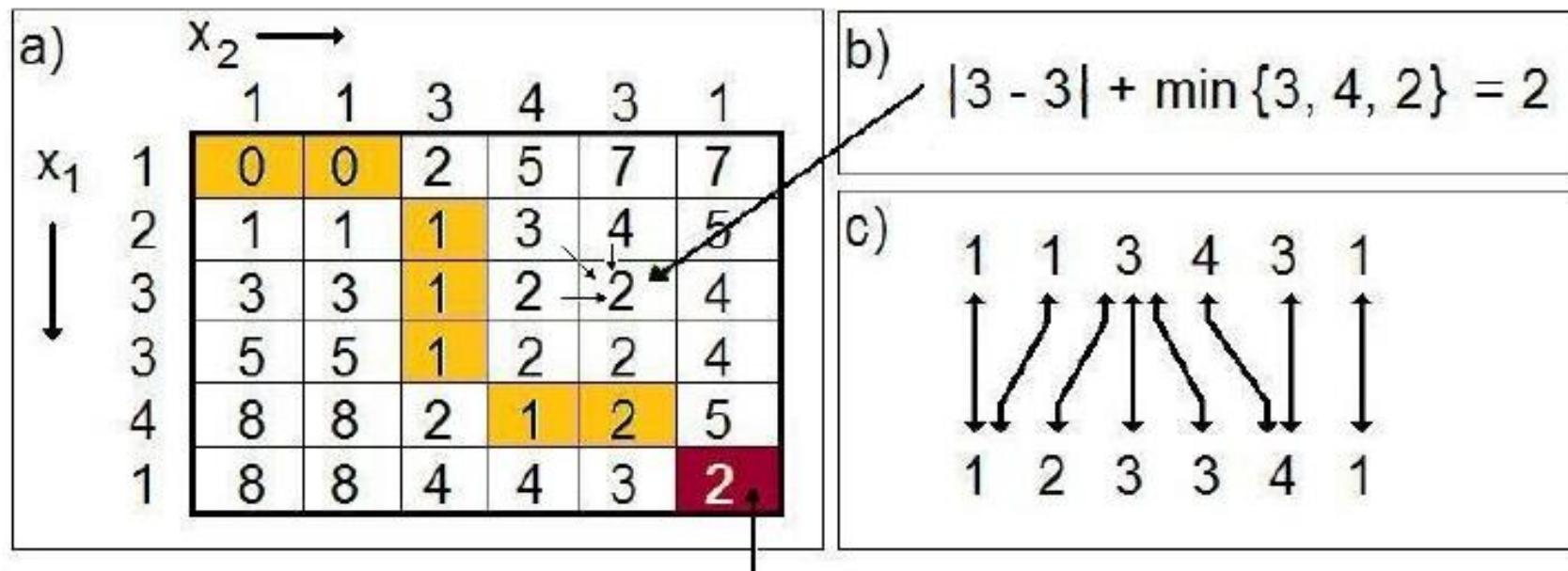
- Let $\text{DTW}(i, j)$ denote the value in the i th row and j th column of the matrix
- The entry in the upper left corner of the matrix: $\text{DTW}(0,0) = c(x[0], y[0])$
- Calculating other elements using:

$$\text{DTW}(i, j) = c(x[i], y[j]) + \min \begin{cases} \text{DTW}(i, j - 1) + c_d \\ \text{DTW}(i - 1, j) + c_i \\ \text{DTW}(i - 1, j - 1) \end{cases}$$

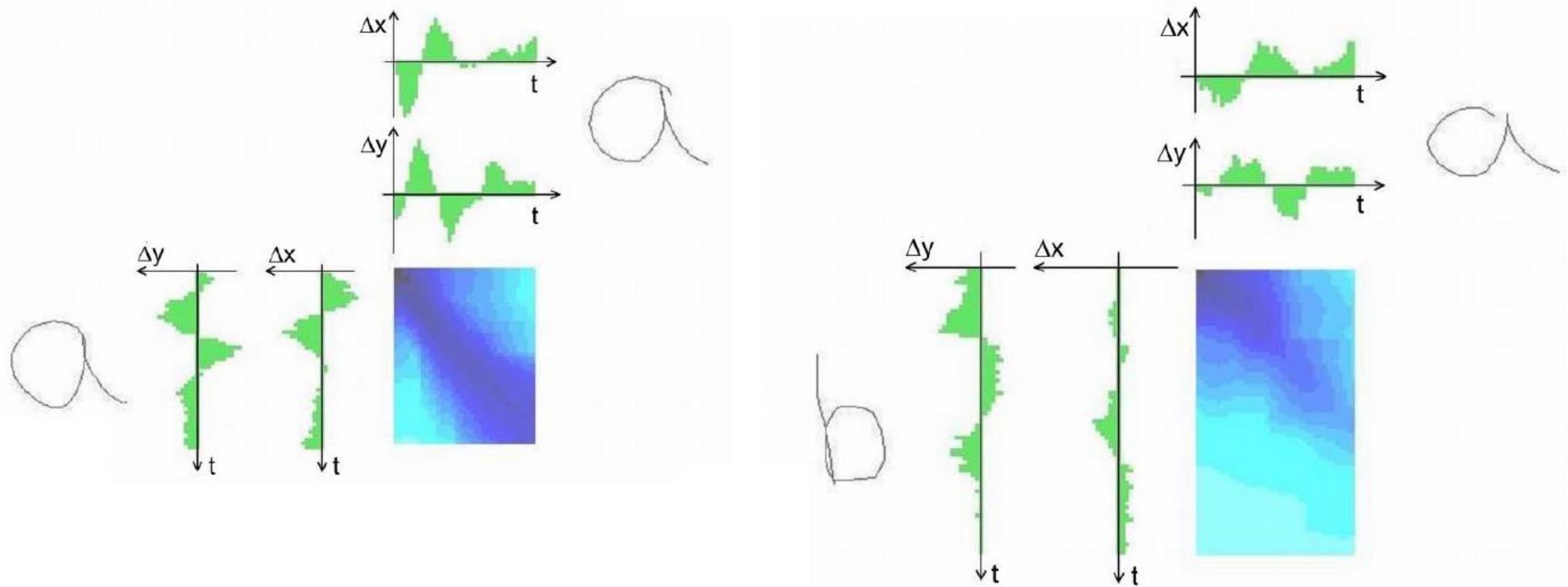
- The terms in the minimum expression respectively corresponds to
 - deletion (compression),
 - insertion (stretch),
 - match

DTW - example

- Warping path: which optimally deforms one of the two input series onto the other
- It works for time series of differing length as well
- Running time: $O(nm)$ / n and m: the length of the time series/
 - It can be faster if we only calculate elements near the main diagonal



Handwritten character recognition using DTW



Problem

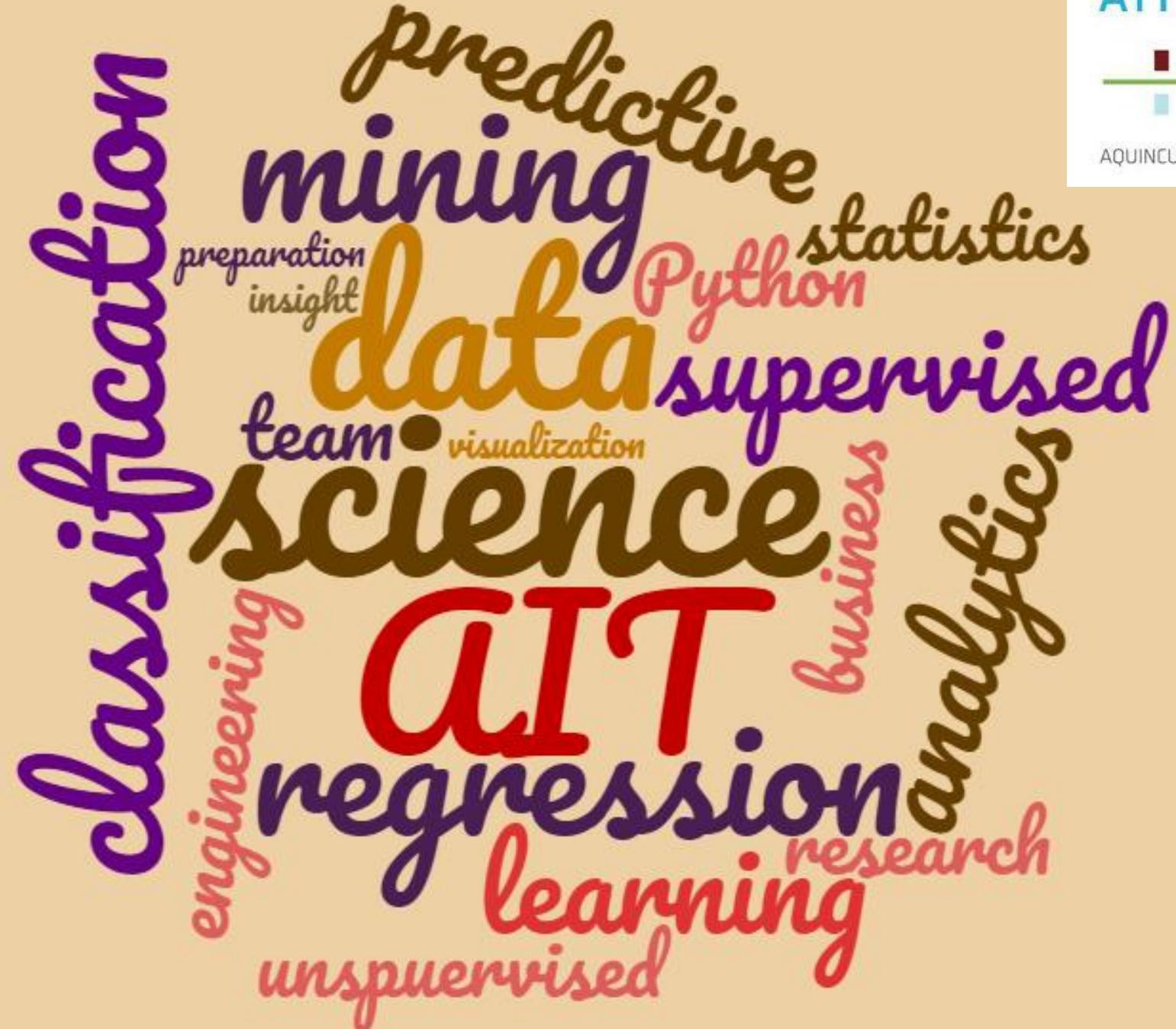
Assuming that the cost of compression/stretching is 0, determine the DTW distance of the following time series (let the inner distance function be the absolute distance)! Find optimal alignment between two time series (the warping path)!

$$t_1 = (3, 2, 5, 7, 8, 9), \quad t_2 = (2, 3, 2, 3, 6, 8)$$

Acknowledgement

- András Benczúr, Róbert Pálovics, SZTAKI-AIT, DM1-2
- Krisztián Buza, MTA-BME, VISZJV68
- Bálint Daróczy, SZTAKI-BME, VISZAMA01
- Judit Csima, BME, VISZM185
- Gábor Horváth, Péter Antal, BME, VIMMD294, VIMIA313
- Lukács András, ELTE, MM1C1AB6E
- Tim Kraska, Brown University, CS195
- Dan Potter, Carsten Binnig, Eli Upfal, Brown University, CS1951A
- Erik Sudderth, Brown University, CS142
- Joe Blitzstein, Hanspeter Pfister, Verena Kaynig-Fittkau, Harvard University, CS109
- Rajan Patel, Stanford University, STAT202
- Andrew Ng, John Duchi, Stanford University, CS229



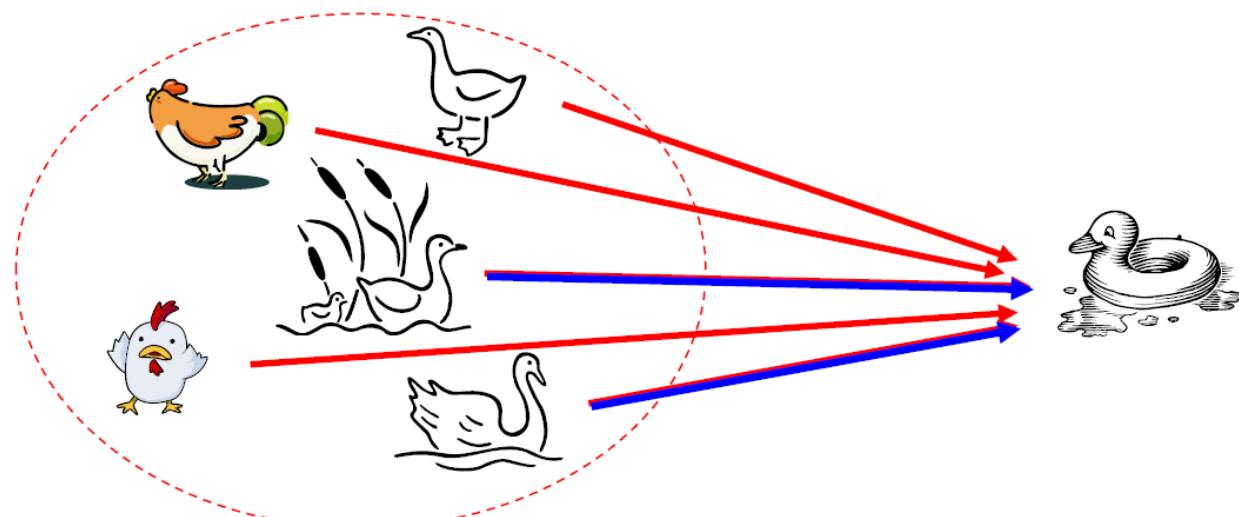


Schedule of the semester

	<i>Monday midnight</i>	<i>Tuesday class</i>	<i>Friday class</i>
W1 (02/06)			
W2 (02/13)		HW1 out	
W3 (02/20)			
W4 (02/27)	HW1 deadline + TEAMS	HW2 out	
W5 (03/06)	PROJECT PLAN		
W6 (03/13)	HW2 deadline	HW3 out	
W7 (03/20)			MIDTERM
SPRING BREAK		SPRING BREAK	SPRING BREAK
W8 (04/03)	HW3 deadline		GOOD FRIDAY
W9 (04/10)	MILESTONE 1	HW4 out	
W10 (04/17)			
W11 (04/24)	HW4 deadline		
W12 (05/01)	MILESTONE 2		
W13 (05/08)			
W14 (05/15)		FINAL	PROJECT presentations
W15 (05/22)		PROJECT presentations	

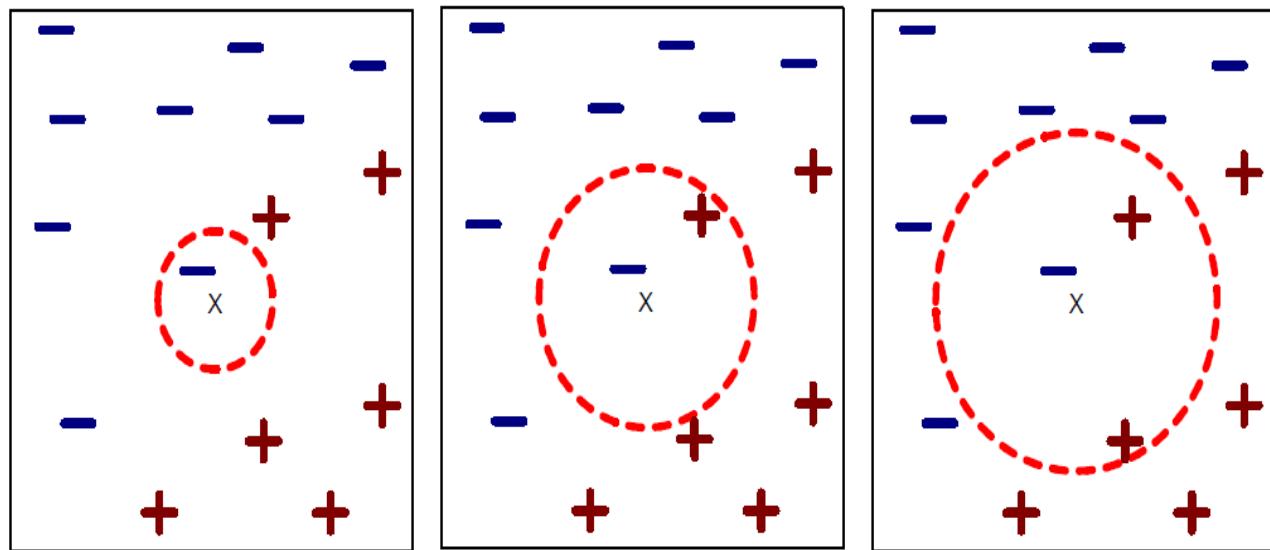
K Nearest Neighbors (kNN)

- kNN: k Nearest Neighbors
- Principle: „If it looks like a duck, swims like a duck, and quacks like a duck, then it probably *is* a duck.”
- It can be used for classification and regression as well



kNN approach

- Records: points in the d -dimensional space (where d is the number of attributes excluding the label)
- The label of a new record is determined by the labels of its k nearest neighbors in the training set
 - What similarity measure to use?
 - How to choose k ?
 - How to decide on the target variable?



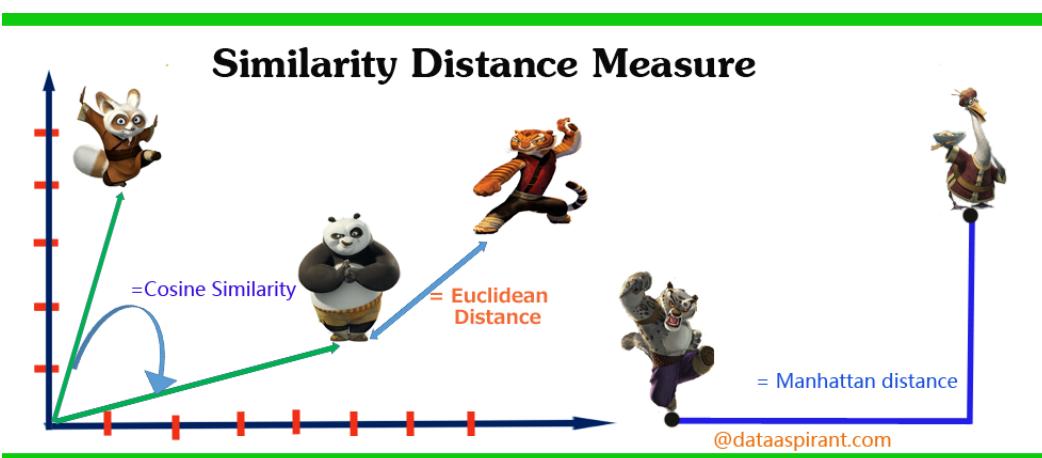
(a) 1-nearest neighbor

(b) 2-nearest neighbor

(c) 3-nearest neighbor

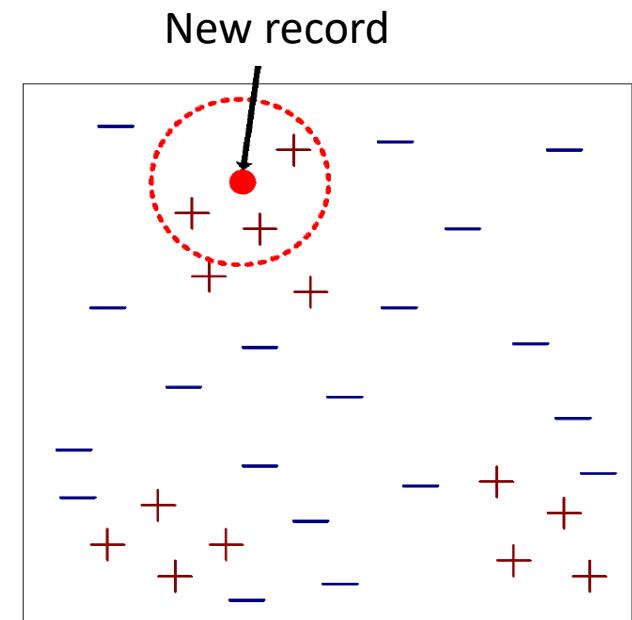
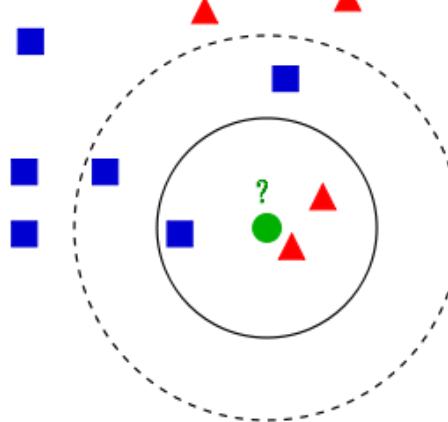
How to choose the right (dis)similarity measure?

- Choose the (dis)similarity that is most suitable for the problem in hand
 - Sometimes we choose the suitable measure intuitively, ensuring that those objects are close according to the chosen measure that we think are similar indeed
 - We can try out more measures and test which has the best performance
 - For possible (dis)similarity measure, see Lecture 03



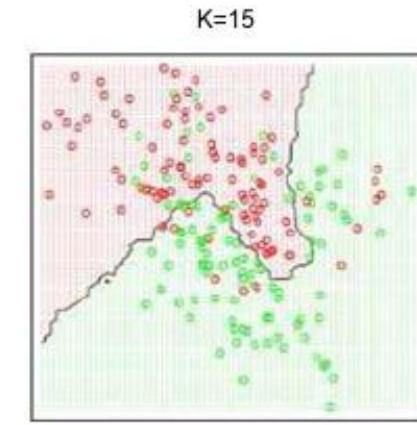
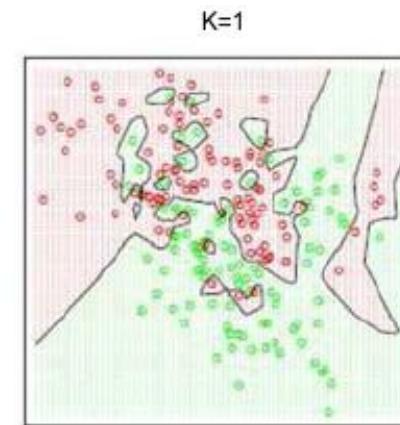
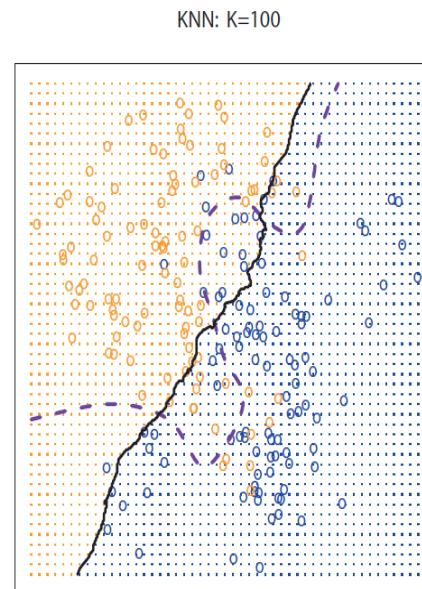
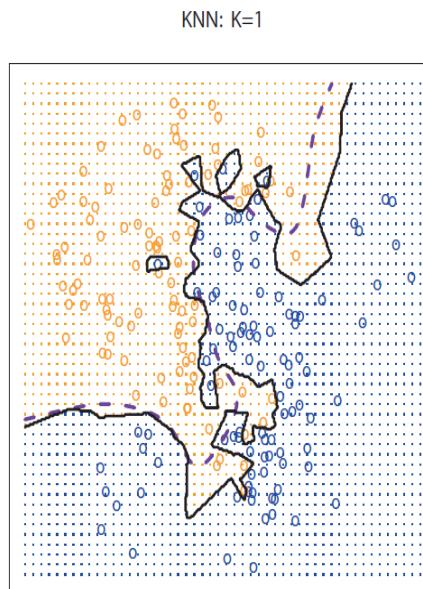
How to choose k ?

- If k is too small, it is too sensitive for noise, for local errors
 - It is sensitive to the training set itself: „high variance”
- If k is too large, then too dissimilar objects are also taken into consideration
 - The model is not flexible enough: „high bias”



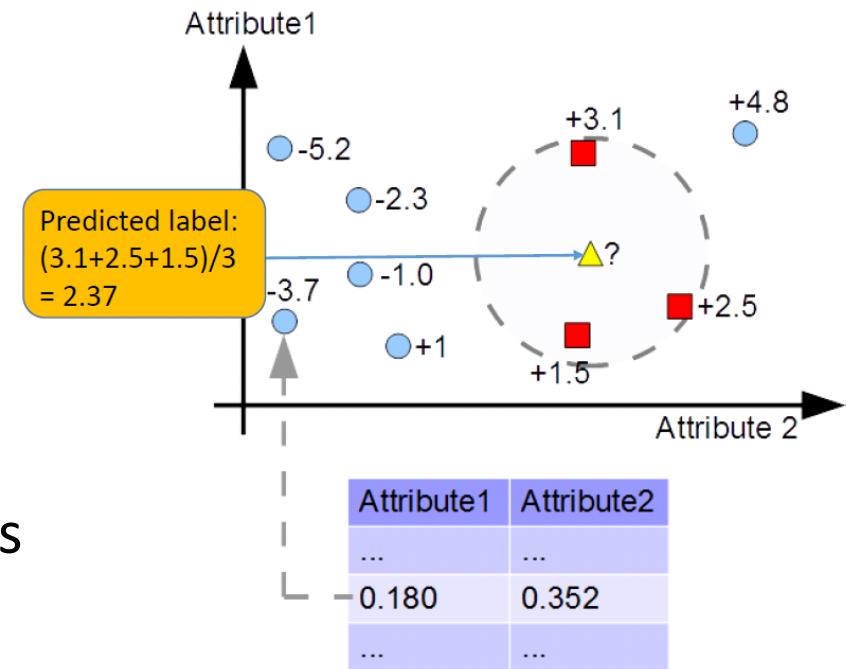
Effect of choosing k

- The bigger the k value is the smoother the boundary is, the smaller the effect of noise is
- If k is too large, the objects that are far away also play a role
 - E.g. if $k=N$, we predict the majority class for every instance



How to decide on the output?

- For classification problems:
 - Majority voting among the labels of the k neighbors
 - For binary classification k should be odd
 - Weighted voting, a possible weight:
 $w_i = \frac{1}{d_i^2}$, where d_i is the distance of i th neighbor
- For regression problems:
 - Averaging the target variables of the k neighbors
 - Weighted averaging (similarly inversely proportional to the distance)



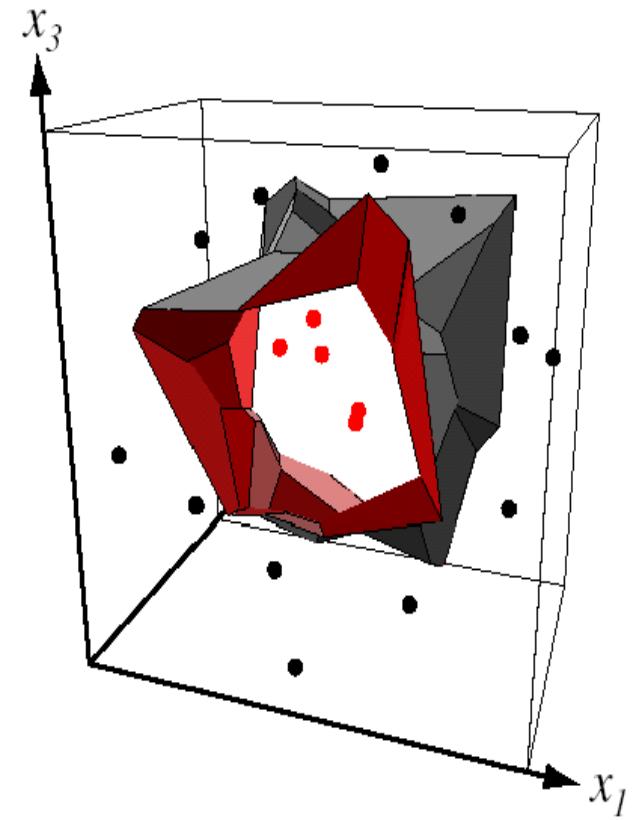
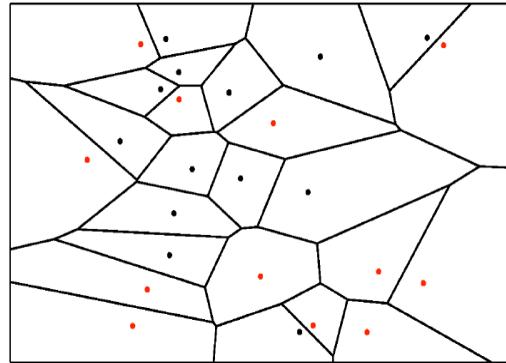
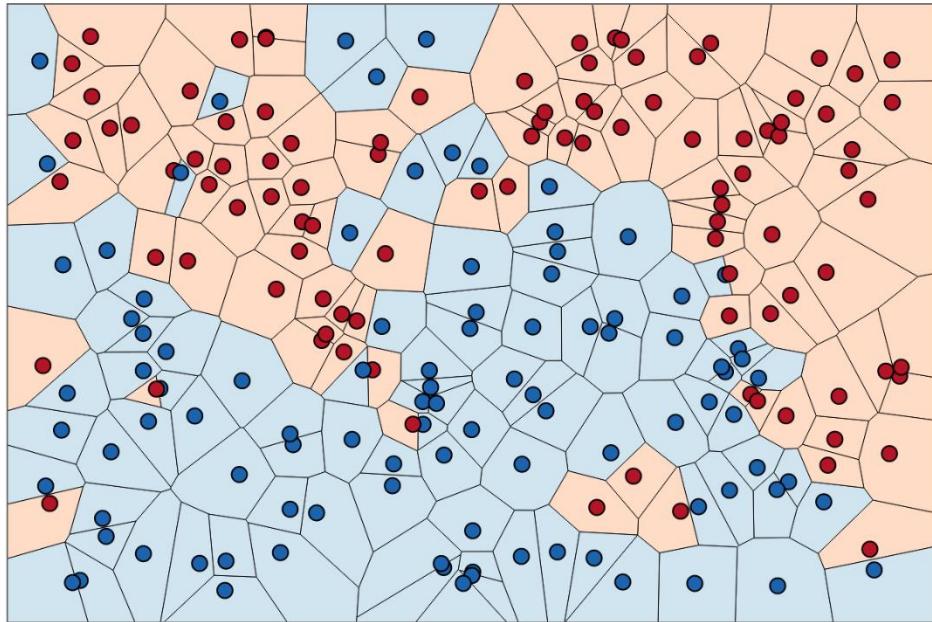
Advantages / disadvantages

- Simple, easy to understand, easy to implement
- Widespread
- Theoretically efficient (for infinitely many training data points)
- „lazy learner“: it works only if a new unlabeled data point arrives
 - There is no explicitly built models, no long model building phase
- For $k=1$ we can make the prediction very fast using some preparation
 - Voronoi diagram (Voronoi cells): Partitioning the space into regions based on distance to some seed points. A cell consists of all points closest to its seed. The seeds are the training points.

- Classifying one record is relatively slow for large training set
- For good performance an arbitrary point should have enough number of training points in its neighborhood
- For good performance, in increased dimension the number of training points should increase exponentially
- Sensitive for irrelevant and correlated attributes
 - Its sensitivity also depends on the chosen dissimilarity (distance) measure

Voronoi diagram (cells)

- Space partition by 1NN classifier



How to find the nearest neighbors?

- Naive method: we calculate the distances between the record (that we aim to classify) and all the training points
 - Time complexity: $O(dN)$ (N : number of training points, d : number of attributes, dimension)
- Using space-partitioning data structures for organizing points
 - E.g. using k - d tree
 - Average complexity of nearest neighbor search: $O(\log N)$, in case of randomly distributed points and constant dimension
- There are also approximation algorithms
 - We can improve running time if we accept „good guesses” of the nearest neighbor. It doesn't guarantee to return the actual nearest neighbor
 - For real-world problems, the approximation algorithms work (almost) as good as exact ones

K - d tree

- Build a k - d tree based on the training set
 - $\{(1,9), (2,3), (3,7), (4,1), (5,4), (6,8), (7,2), (8,8), (7,9), (9,6)\}$
 - Choose a random coordinate, calculate the median, divide the data into two parts, repeat the procedure on both branches as long as we need
- Find the nearest neighbor of a new point: $(7,4)$
 - The point is consisted by which range?
 - Calculate the distance to the points in the range
 - We also check the points in the neighboring ranges



Classification error rate

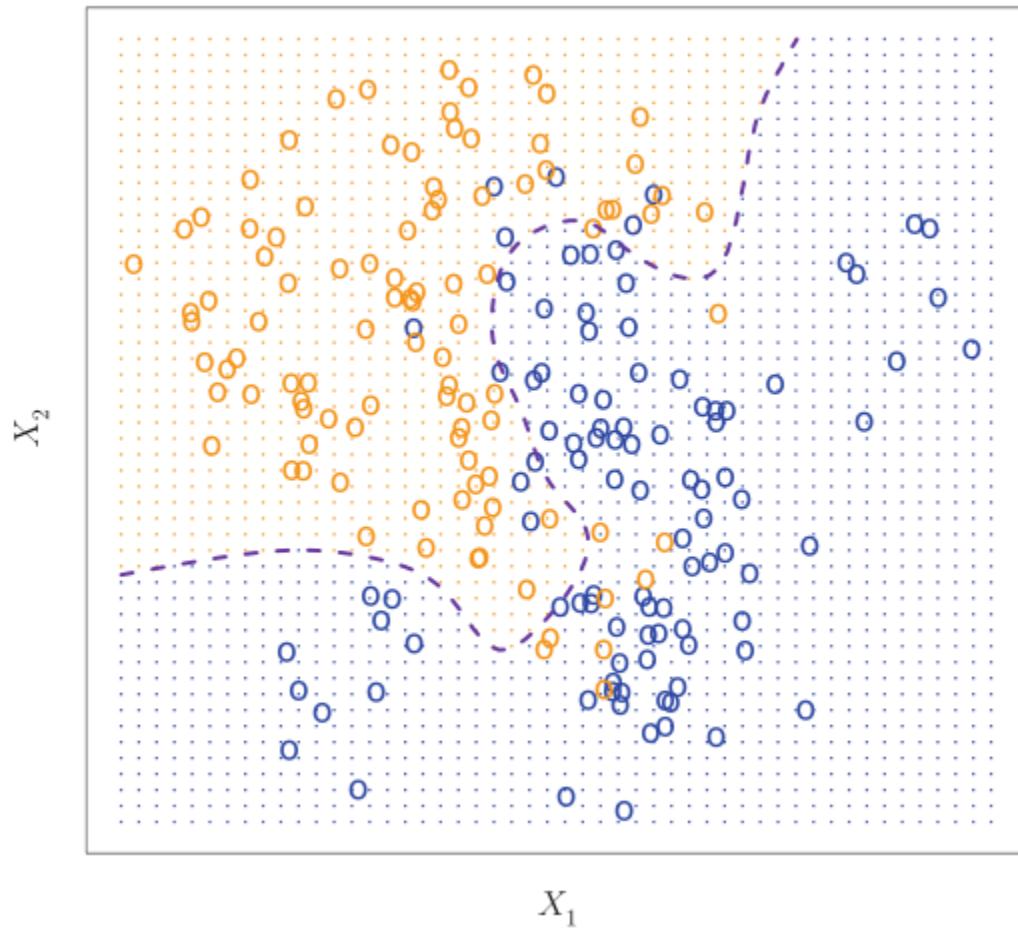
- How can we measure the „goodness” of a classifier?
 - There are several methods (later)
 - The simplest method is the error rate
 - Error rate in the training set
 - Employ the model to the training points: $\{(x_1, y_1), \dots, (x_n, y_n)\}$
 - Error rate is the ratio of misclassified data points:
$$\frac{1}{n} \sum_{i=1}^n I(y_i \neq \hat{y}_i)$$
 - Error rate in the test set
 - We calculate the error rate for new data points that were not used for training
 - A good classifier has a small error rate on the test set

Bayes classifier

- The classifier that minimizes the test error rate is such a classifier that assigns the label to the observation with the highest probability given the attributes
- In other words: to an observation with x_0 feature-vector such a j class is assigned for which the following conditional probability is maximal:
$$P(Y = j|X = x_0)$$
 - For binary classification, label 1 is assigned if: $P(Y = 1|X = x_0) > 0.5$
- The conditional probabilities are not known, unless the data is generated from a given $p(X, Y)$ joint distribution (background distribution)

Bayes decision boundary

- For binary classification problems: it is the region of the attribute space in which the conditional probabilities are equal ($0.5 - 0.5$)
 - The classifier will classify all the points on one side of the decision boundary as belonging to one class and all those on the other side as belonging to the other class
- In the figure it is the purple dashed line



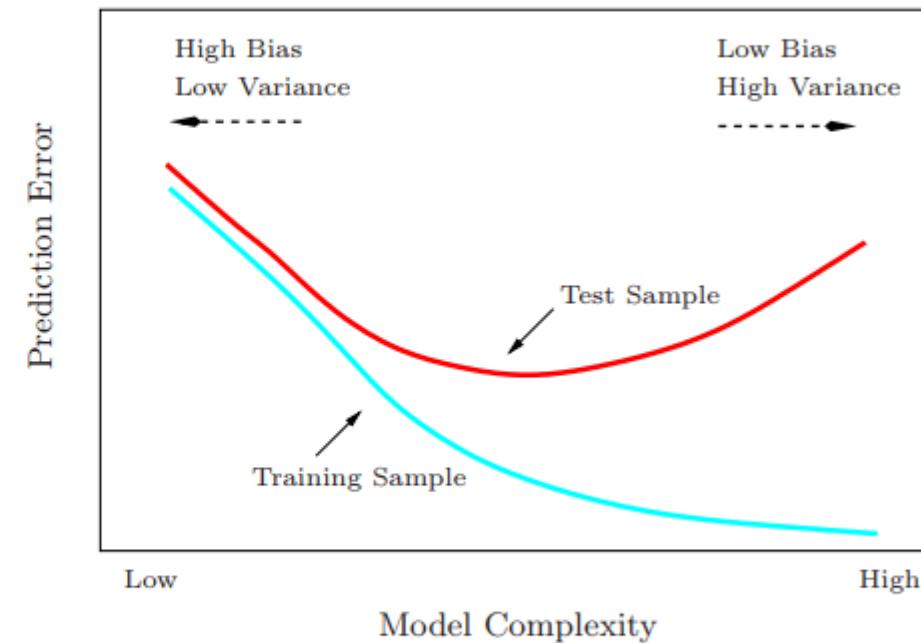
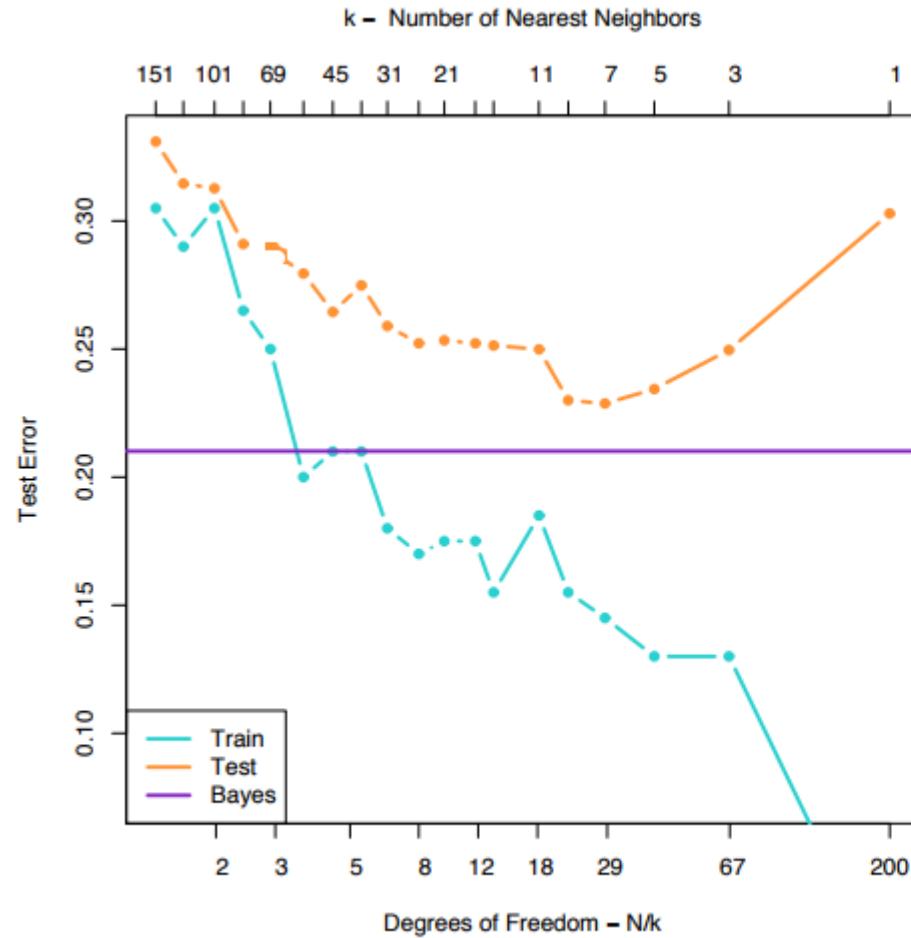
Bayes error rate

- The error rate of the Bayes classifier
- The lowest possible error rate for any classifier on the test set

$$1 - \mathbb{E}(\max_j P(Y = j|X))$$

- Expectation is taken with respect to all possible values of X

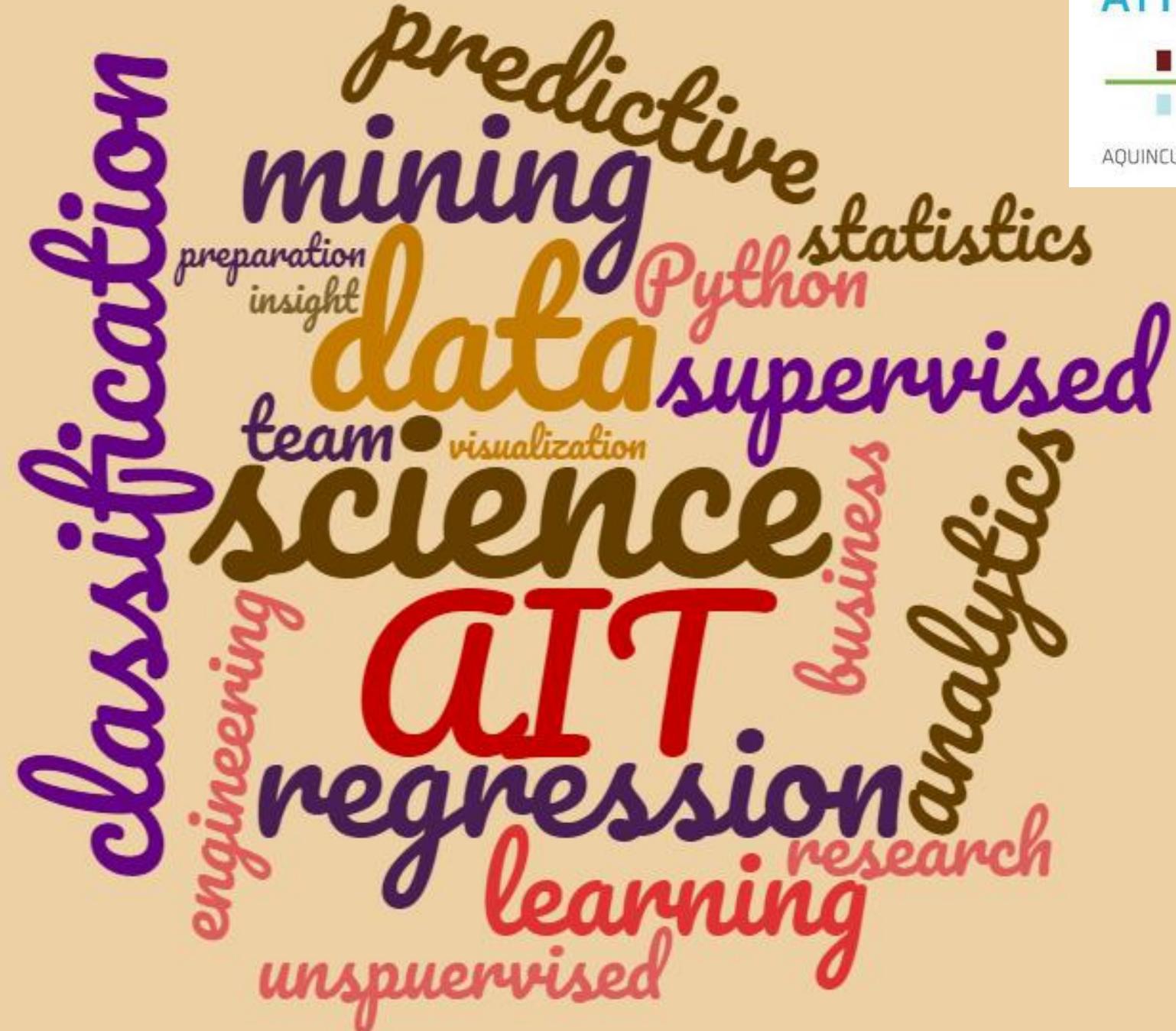
Error rates for k NN algorithm



Acknowledgement

- András Benczúr, Róbert Pálovics, SZTAKI-AIT, DM1-2
- Krisztián Buza, MTA-BME, VISZJV68
- Bálint Daróczy, SZTAKI-BME, VISZAMA01
- Judit Csima, BME, VISZM185
- Gábor Horváth, Péter Antal, BME, VIMMD294, VIMIA313
- Lukács András, ELTE, MM1C1AB6E
- Tim Kraska, Brown University, CS195
- Dan Potter, Carsten Binnig, Eli Upfal, Brown University, CS1951A
- Erik Sudderth, Brown University, CS142
- Joe Blitzstein, Hanspeter Pfister, Verena Kaynig-Fittkau, Harvard University, CS109
- Rajan Patel, Stanford University, STAT202
- Andrew Ng, John Duchi, Stanford University, CS229



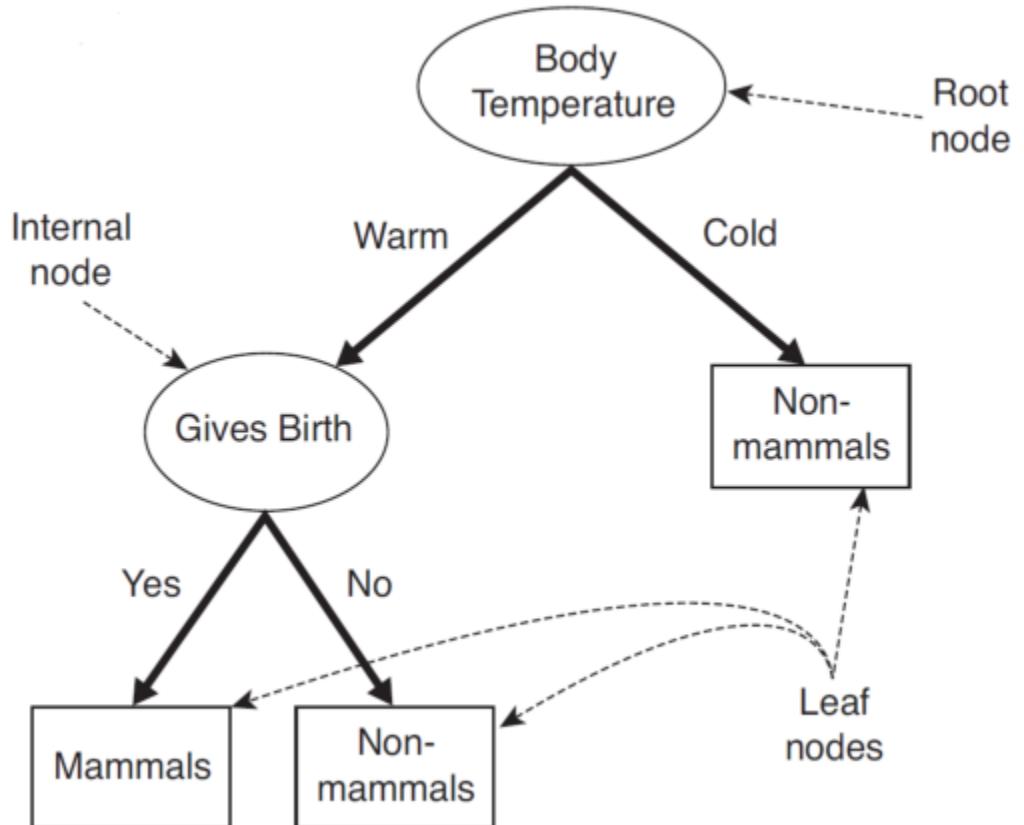


Schedule of the semester

	<i>Monday midnight</i>	<i>Tuesday class</i>	<i>Friday class</i>
W1 (02/06)			
W2 (02/13)		HW1 out	
W3 (02/20)			
W4 (02/27)	HW1 deadline + TEAMS	HW2 out	
W5 (03/06)	PROJECT PLAN		
W6 (03/13)	HW2 deadline	HW3 out	
W7 (03/20)			MIDTERM
SPRING BREAK		SPRING BREAK	SPRING BREAK
W8 (04/03)	HW3 deadline		GOOD FRIDAY
W9 (04/10)	MILESTONE 1	HW4 out	
W10 (04/17)			
W11 (04/24)	HW4 deadline		
W12 (05/01)	MILESTONE 2		
W13 (05/08)			
W14 (05/15)		FINAL	PROJECT presentations
W15 (05/22)		PROJECT presentations	

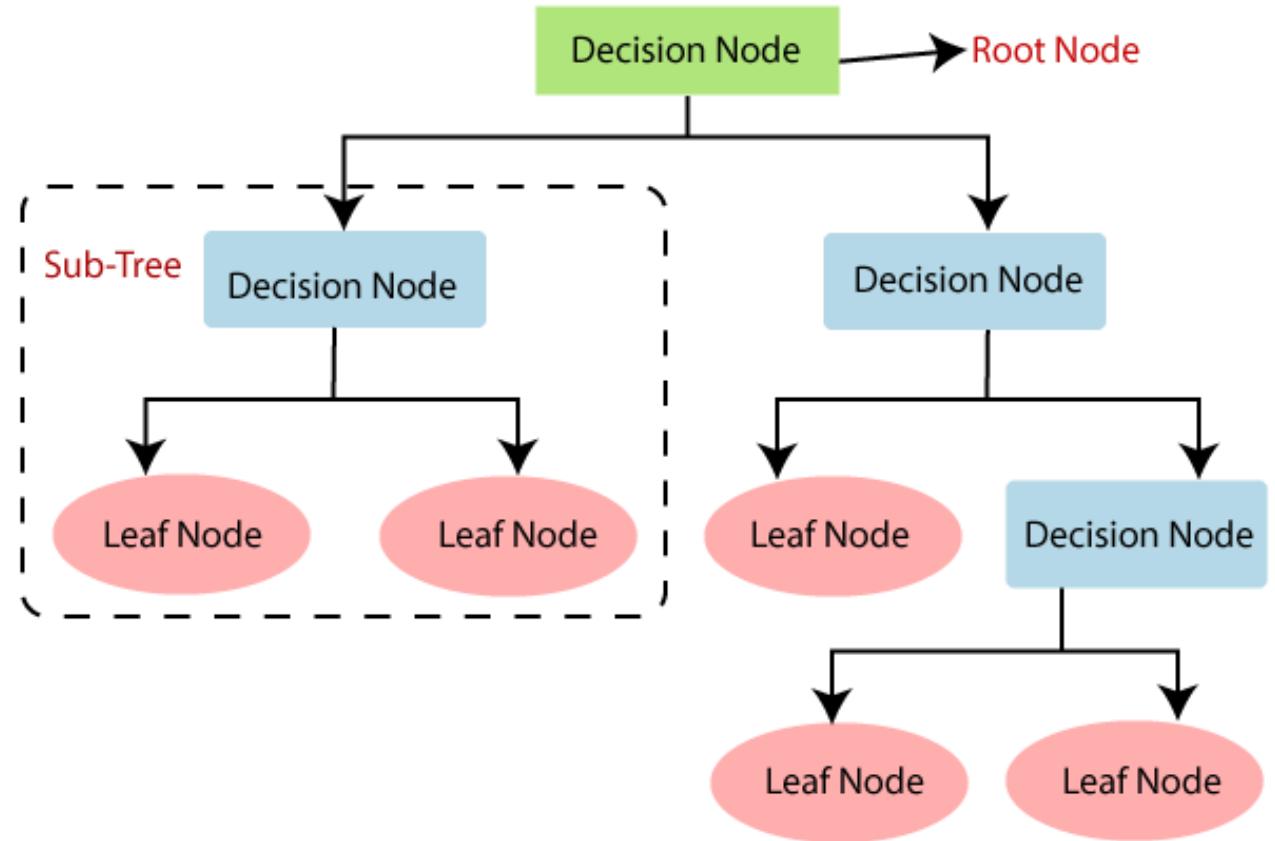
Decision tree

- Rooted, directed (sometimes binary) tree
- Each inner node has a decision rule that assigns instances uniquely to child nodes of the actual node
- Each leaf node has a class label



Prediction with decision tree

- Decision tree is one of the oldest predictive algorithm
 - We start at root node
 - At each interior node we evaluate the decision rule, branch to the child node picked by the decision rule
 - Once a leaf node is reached, predict the label assigned to that node
- It is mainly used for classification, but also suitable for regression problems

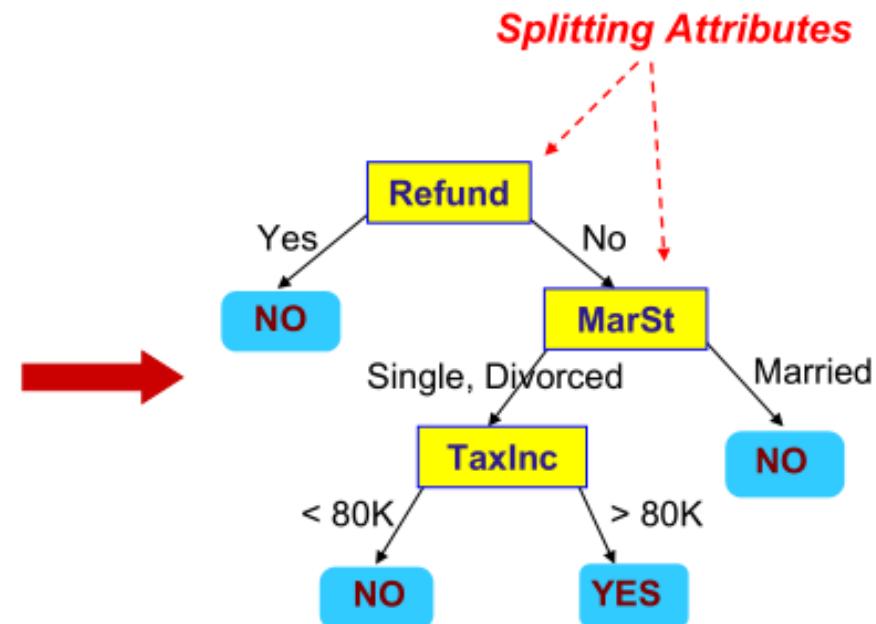


Decision tree - learning phase

It works with continuous and categorical attributes

Tid	Refund	Marital Status	Taxable Income	Cheat	
				categorical	categorical
				continuous	class
1	Yes	Single	125K	No	
2	No	Married	100K	No	
3	No	Single	70K	No	
4	Yes	Married	120K	No	
5	No	Divorced	95K	Yes	
6	No	Married	60K	No	
7	Yes	Divorced	220K	No	
8	No	Single	85K	Yes	
9	No	Married	75K	No	
10	No	Single	90K	Yes	

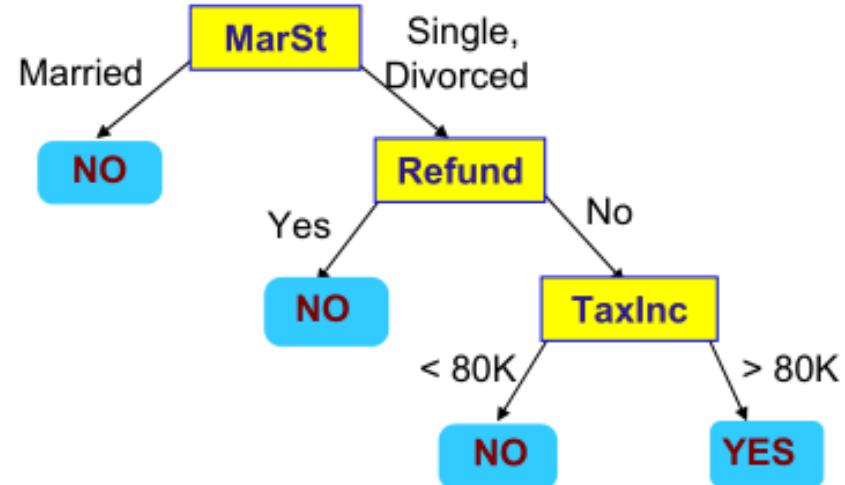
Training Data



Model: Decision Tree

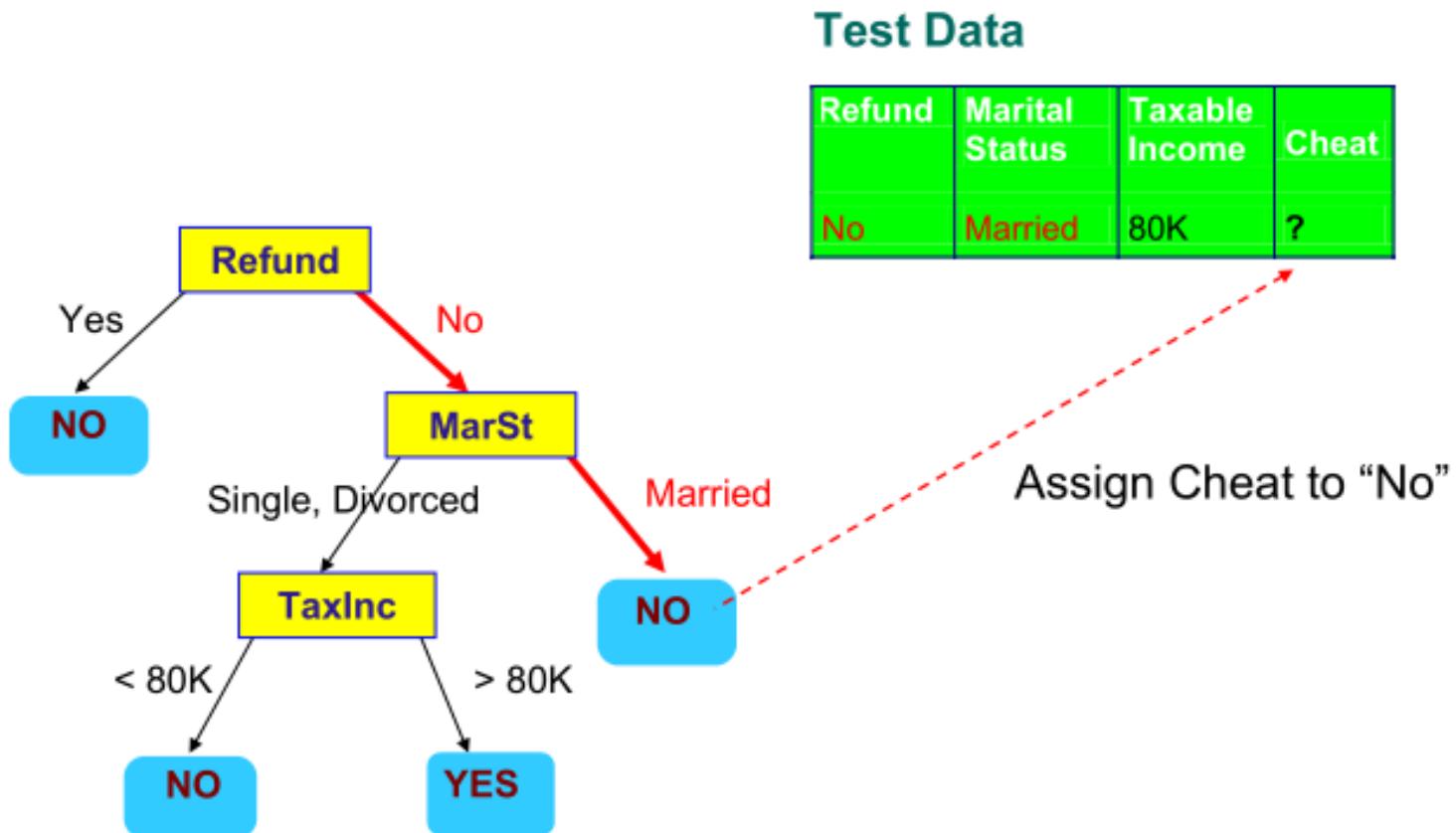
Another tree for the same data

<i>Tid</i>	Refund	Marital Status	Taxable Income	Cheat
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes



There could be more than one tree that fits the same data!

Applying the model to new data



Problems

Ex 1

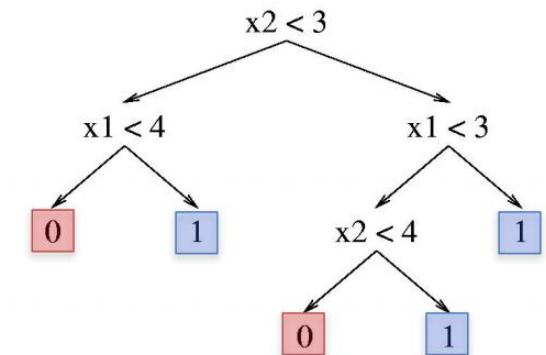
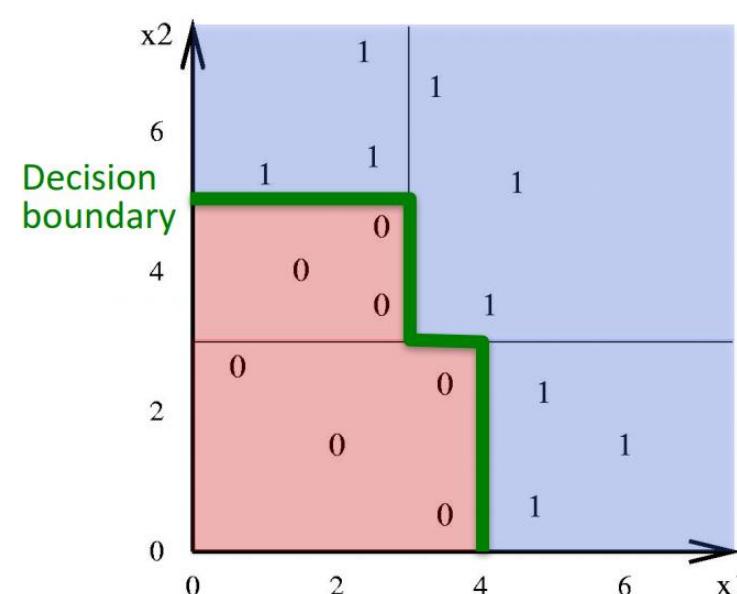
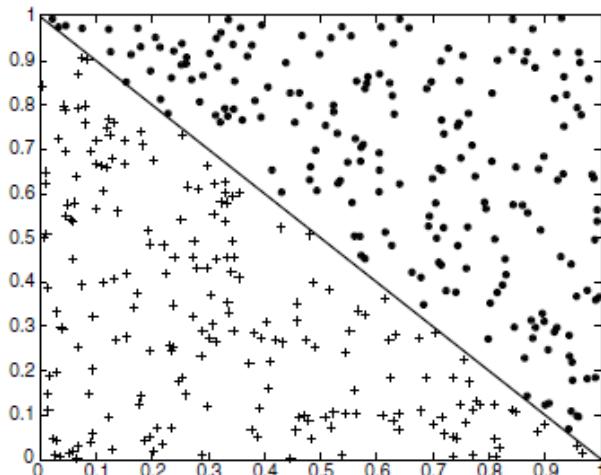
How many logical (Boolean, $f : \{0, 1\}^N \rightarrow \{0, 1\}$) functions can be generated on N binary attributes? What are the possible functions for $N = 2$?

Ex 2

Can decision trees learn logical (Boolean) functions? How to represent the following functions with a decision tree: A **OR** B, A **AND** B, A **XOR** B, where A and B are logical variables.

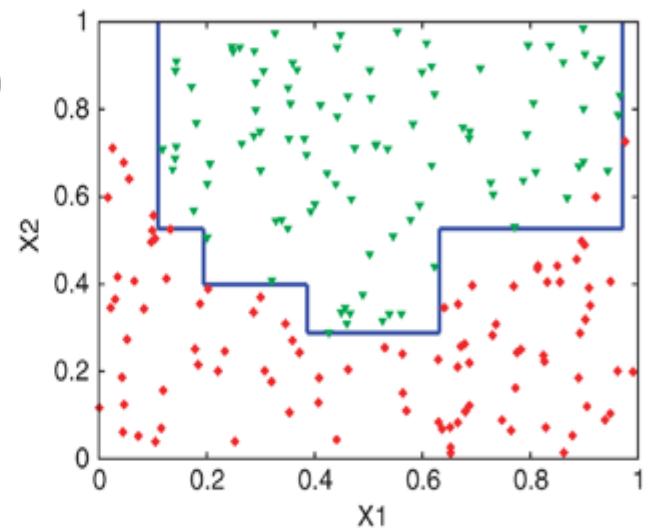
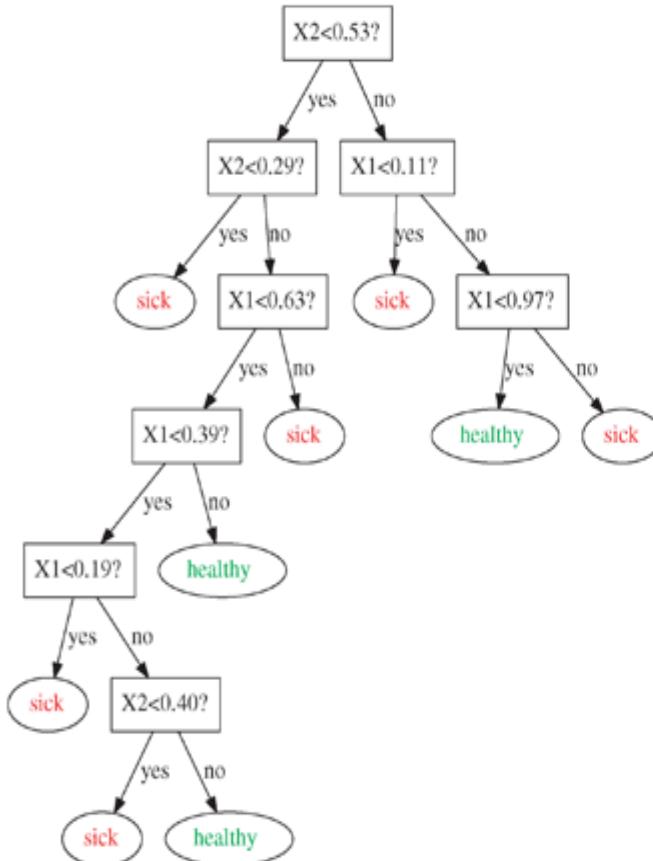
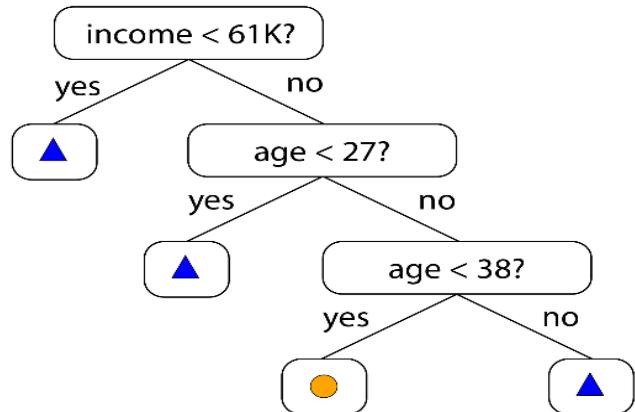
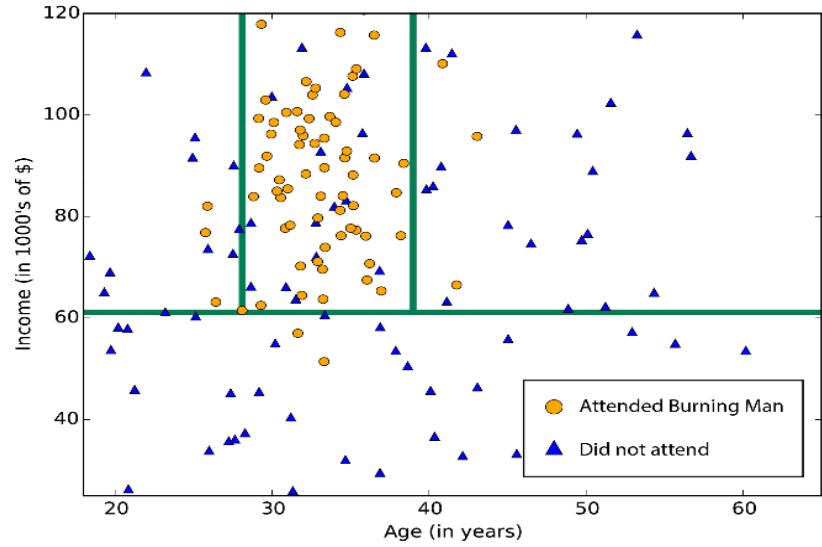
Partitioning the feature space

The boundaries are always parallel to the axes → the decision regions are always unions of (hyper-)rectangles



How could a decision tree represent a data set on the left?

Partitioning the feature space - examples



Algorithm for building decision trees

- We consider a general algorithm
 - Special versions of the algorithm are implemented in various programs
- We do not aim to find the best tree (it is underdefined what is best) but a good enough tree
 - There are exponentially many possible trees
- With a greedy method, deciding locally, quickly
- There are various algorithms (and their variants)
 - Hunt algorithm
 - CART
 - ID3, C4.5 (J48)
 - SLIQ, SPRINT

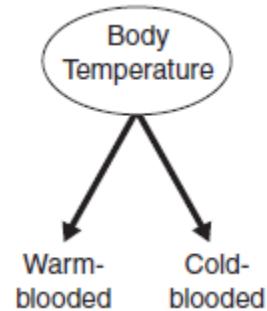
Sketch of Hunt algorithm

- We start with one node (the root), all the records are there
 - Its label is the majority label
- Further steps: we choose a node that is worth splitting on
- End: until there is no node worth splitting on
 - In each node all the records have the same label
 - There are no good splits, e.g. in all nodes, all the records agree in each attribute (aside from the label)

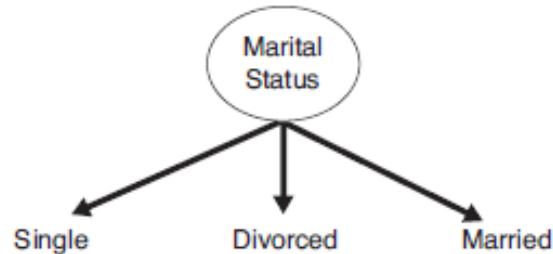
Hunt algorithm - questions

- When does it terminate?
 - When no more nodes left that are worth splitting on
- When is it not worth splitting on?
 - In each node all the records have the same label
 - There are no good splits, e.g. in all nodes, all the records agree in each attribute (aside from the label)
 - If we want to avoid to have a too deep tree (more details later)
- Which node to split on if there are more possibilities?
 - E.g. traversing nodes according to BFS (breadth-first search), DFS (depth-first search)
- How to split?
 - Splitting a node should increase the homogeneity (with respect to the label) of resultant sub-nodes
- How to decide on the label?
 - Using majority voting

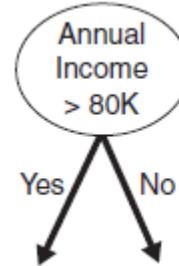
Possible splits



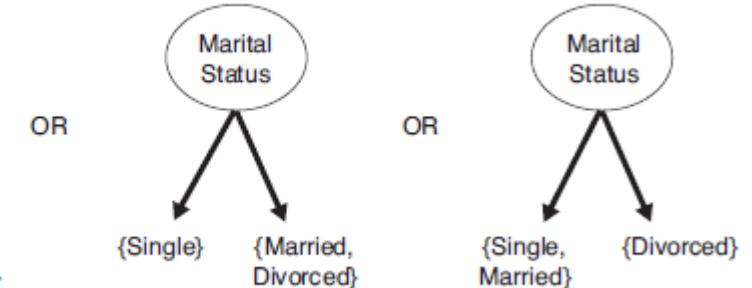
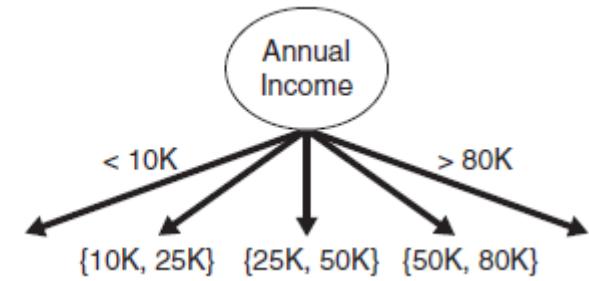
Binary



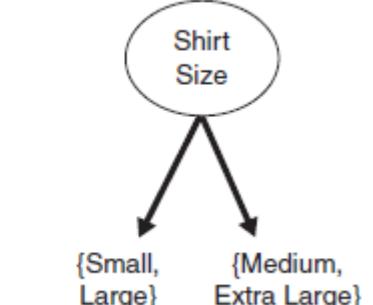
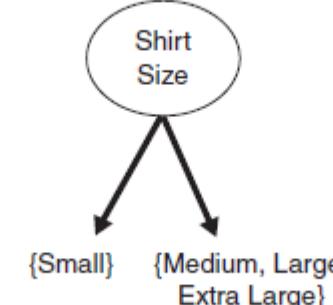
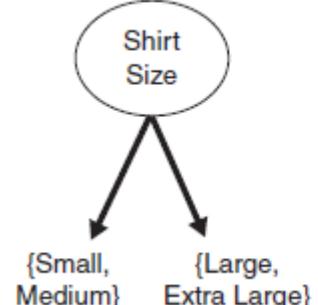
Nominal



Numerical

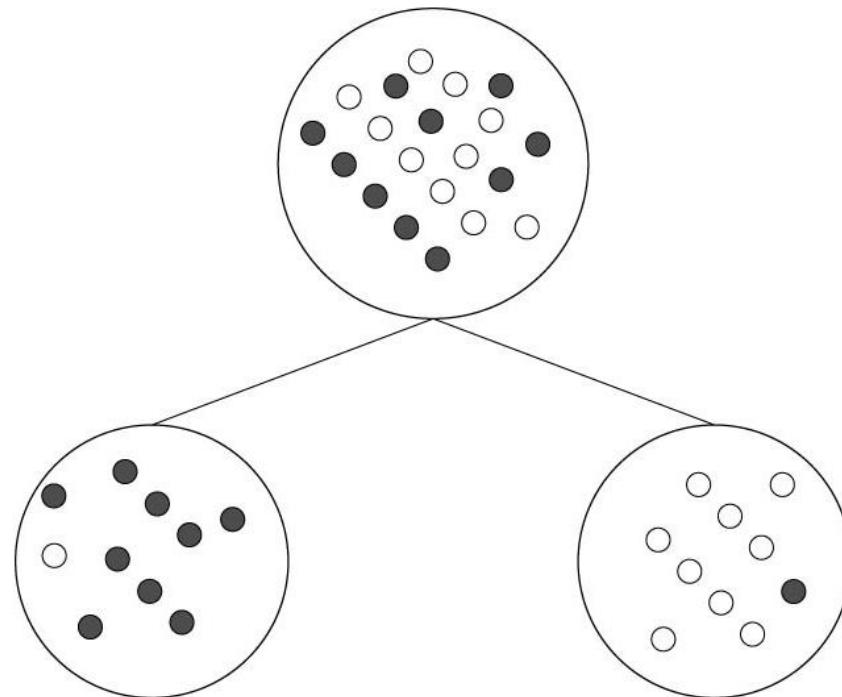


Ordinal



What defines a good split?

- It increases the homogeneity („purity”) of the target variable in the emerging child nodes compared to the parent node
- A good split creates child nodes of similar size (or at least does not create very small nodes)



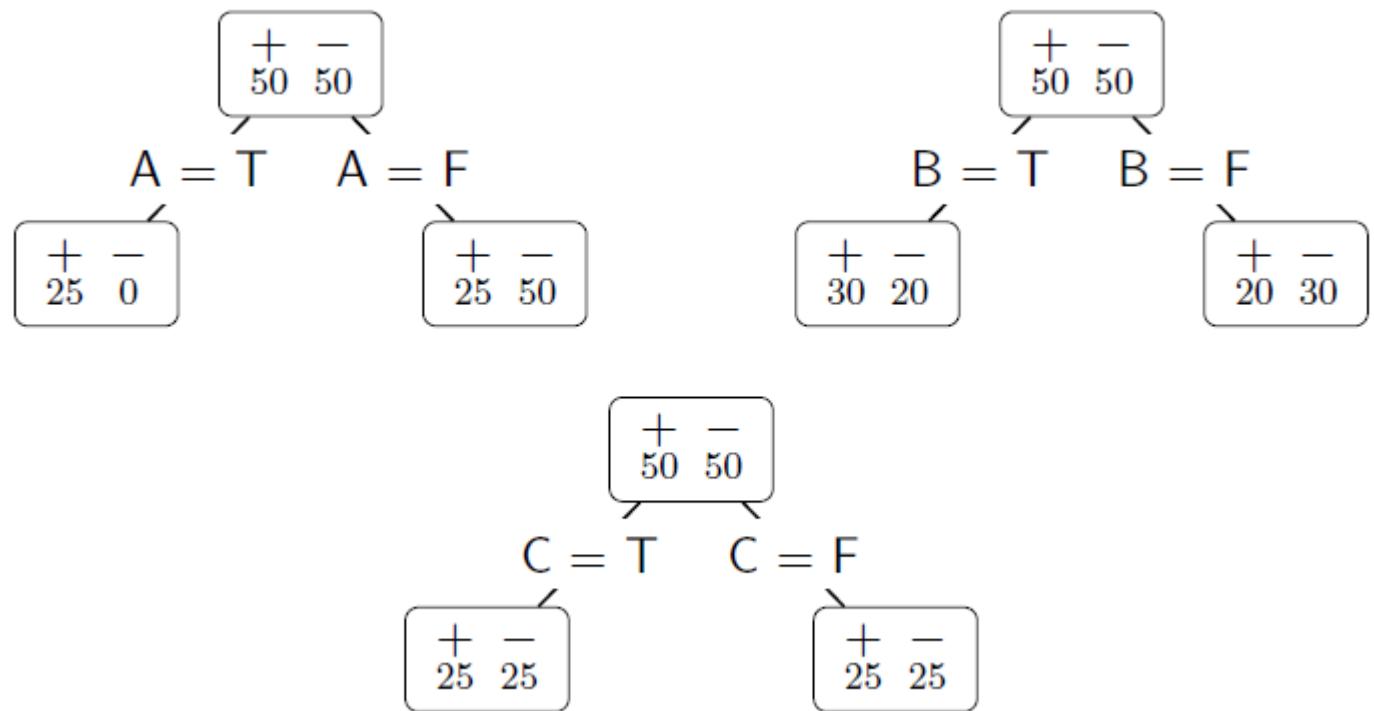
Example

- The following table summarizes a data set with three binary attributes (A, B, C) and two class labels (+, -).
- Which is the best split? What attribute would you split on?

A	B	C	Number of instances	
			class: +	class: -
T	T	T	5	0
F	T	T	0	20
T	F	T	20	0
F	F	T	0	5
T	T	F	0	0
F	T	F	25	0
T	F	F	0	0
F	F	F	0	25

Possible splits

A	B	C	Number of instances	
			class: +	class: -
T	T	T	5	0
F	T	T	0	20
T	F	T	20	0
F	F	T	0	5
T	T	F	0	0
F	T	F	25	0
T	F	F	0	0
F	F	F	0	25

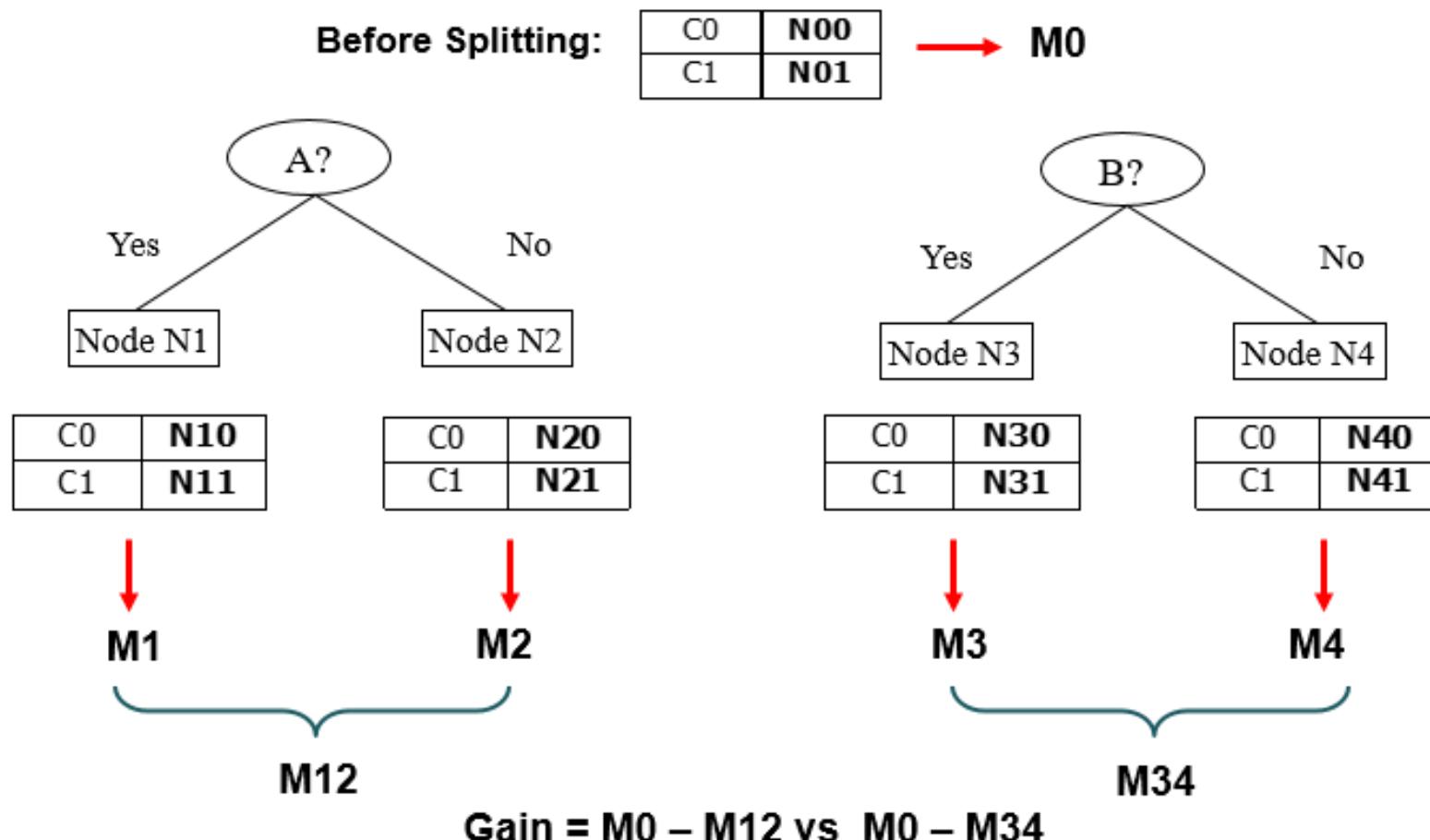


A is the best split!

How to measure the goodness of a split?

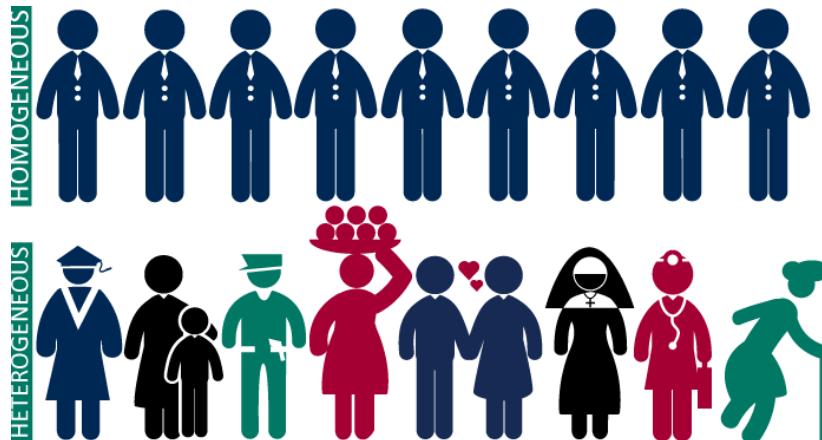
- We consider three possible metrics
- The main principle is the same for all:
 - We define a metric for a set of records that measures the degree of homogeneity (purity) of the target variable within that set
 - We consider three metrics: Gini coefficient, entropy, misclassification error
 - The goodness of a split is measured by the difference between the impurity of the parent node and the emergent child nodes
 - How much do we gain if we split the node? What is the degree of homogeneity increase?
 - We can also consider the size of the emerging child nodes, punishing the too small splits

Which split is better?



Measuring inhomogeneity

- Let t be the node of a decision tree (i.e. a set of records), we aim to measure its homogeneity with respect to a target variable that has c possible values
- Let $p(i|t)$ denote the relative frequency of records with label i in node t



Misclassification error

- Misclassification error or classification error:

$$\text{Classification error } (t) = 1 - \max_i [p(i|t)]$$

- Its maximal value is: $1 - 1/c$, when the records are distributed equally in all classes
- Its minimal value is 0, when all the records have the same label

C1	0
C2	6

$$P(C1) = 0/6 = 0 \quad P(C2) = 6/6 = 1$$
$$\text{Error} = 1 - \max(0, 1) = 1 - 1 = 0$$

C1	1
C2	5

$$P(C1) = 1/6 \quad P(C2) = 5/6$$
$$\text{Error} = 1 - \max(1/6, 5/6) = 1 - 5/6 = 1/6$$

C1	2
C2	4

$$P(C1) = 2/6 \quad P(C2) = 4/6$$
$$\text{Error} = 1 - \max(2/6, 4/6) = 1 - 4/6 = 1/3$$

Gini coefficient

- Gini coefficient

$$\text{Gini}(t) = 1 - \sum_{i=1}^c p(i|t)^2$$

C1	0
C2	6

$$\begin{aligned}\mathbf{P(C1)} &= 0/6 = 0 & \mathbf{P(C2)} &= 6/6 = 1 \\ \mathbf{\text{Gini}} &= 1 - \mathbf{P(C1)^2} - \mathbf{P(C2)^2} = 1 - 0 - 1 = 0\end{aligned}$$

- Its value ranges between 0 and $1 - 1/c$
- Usually Gini is default setting for splitting criterion

C1	1
C2	5

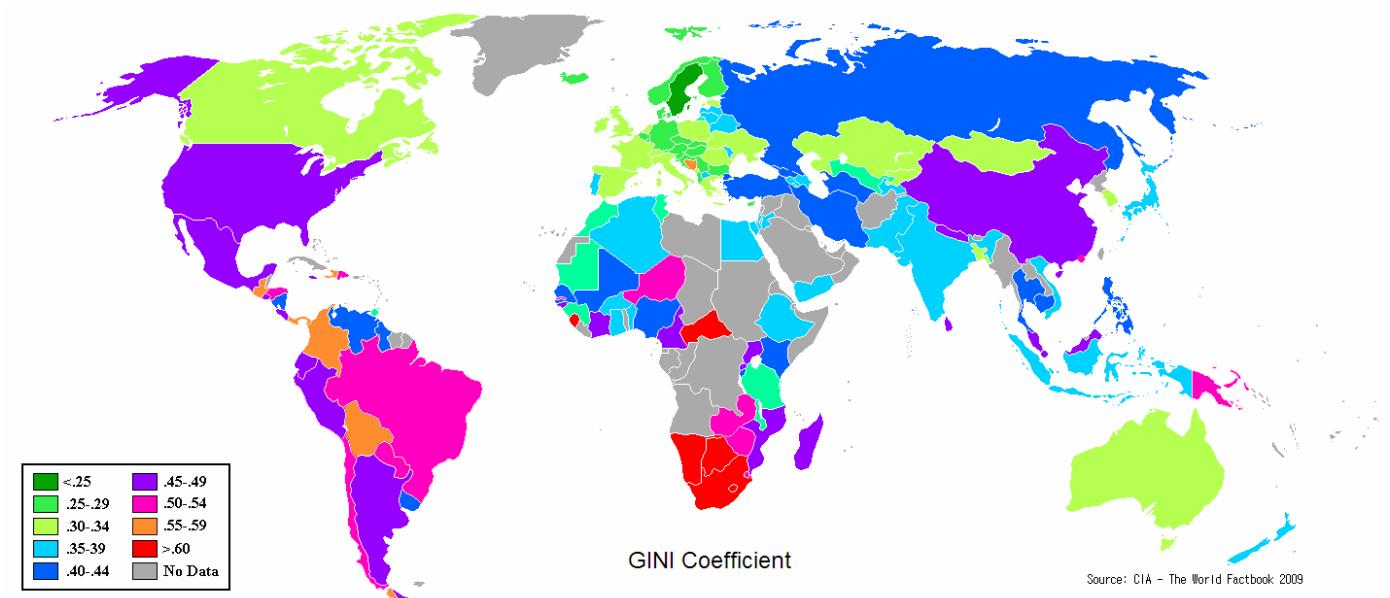
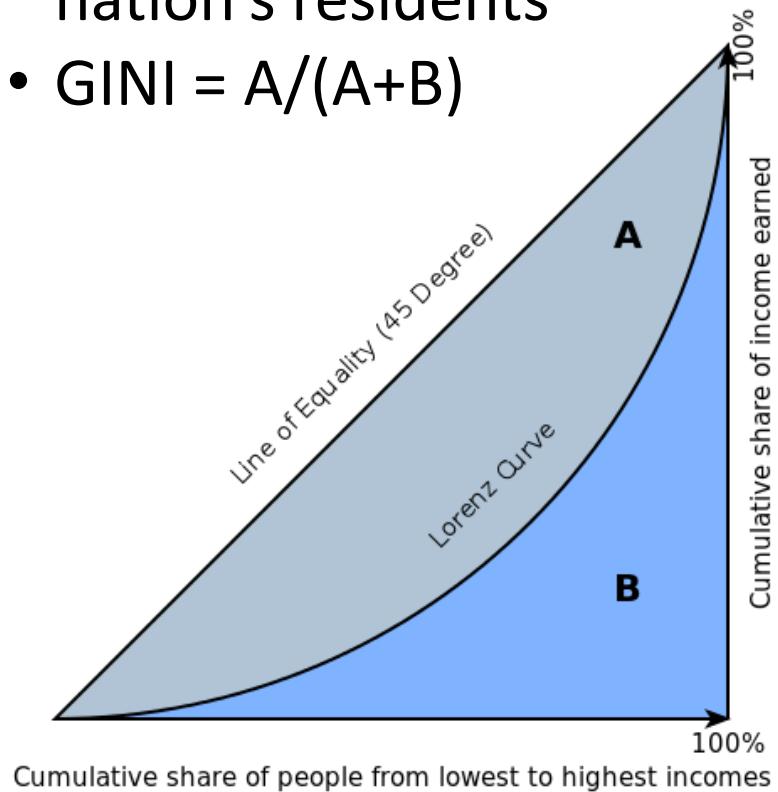
$$\begin{aligned}\mathbf{P(C1)} &= 1/6 & \mathbf{P(C2)} &= 5/6 \\ \mathbf{\text{Gini}} &= 1 - (1/6)^2 - (5/6)^2 = 0.278\end{aligned}$$

C1	2
C2	4

$$\begin{aligned}\mathbf{P(C1)} &= 2/6 & \mathbf{P(C2)} &= 4/6 \\ \mathbf{\text{Gini}} &= 1 - (2/6)^2 - (4/6)^2 = 0.444\end{aligned}$$

Outlook – Gini coefficient in economics

- A similar concept but not the same, do not mix them
- It is used to measure the inequality in income or wealth distribution of a nation's residents
- $\text{GINI} = A/(A+B)$



GINI Coefficient

Source: CIA - The World Factbook 2009

Entropy

$$\text{Entropy}(t) = - \sum_{i=1}^c p(i|t) \log_2 p(i|t)$$

C1	0
C2	6

$$P(C1) = 0/6 = 0 \quad P(C2) = 6/6 = 1$$

$$\text{Entropy} = -0 \log 0 - 1 \log 1 = -0 - 0 = 0$$

- Its value ranges between 0 and $\log_2 c$

C1	1
C2	5

$$P(C1) = 1/6 \quad P(C2) = 5/6$$

$$\text{Entropy} = -(1/6) \log_2 (1/6) - (5/6) \log_2 (5/6) = 0.65$$

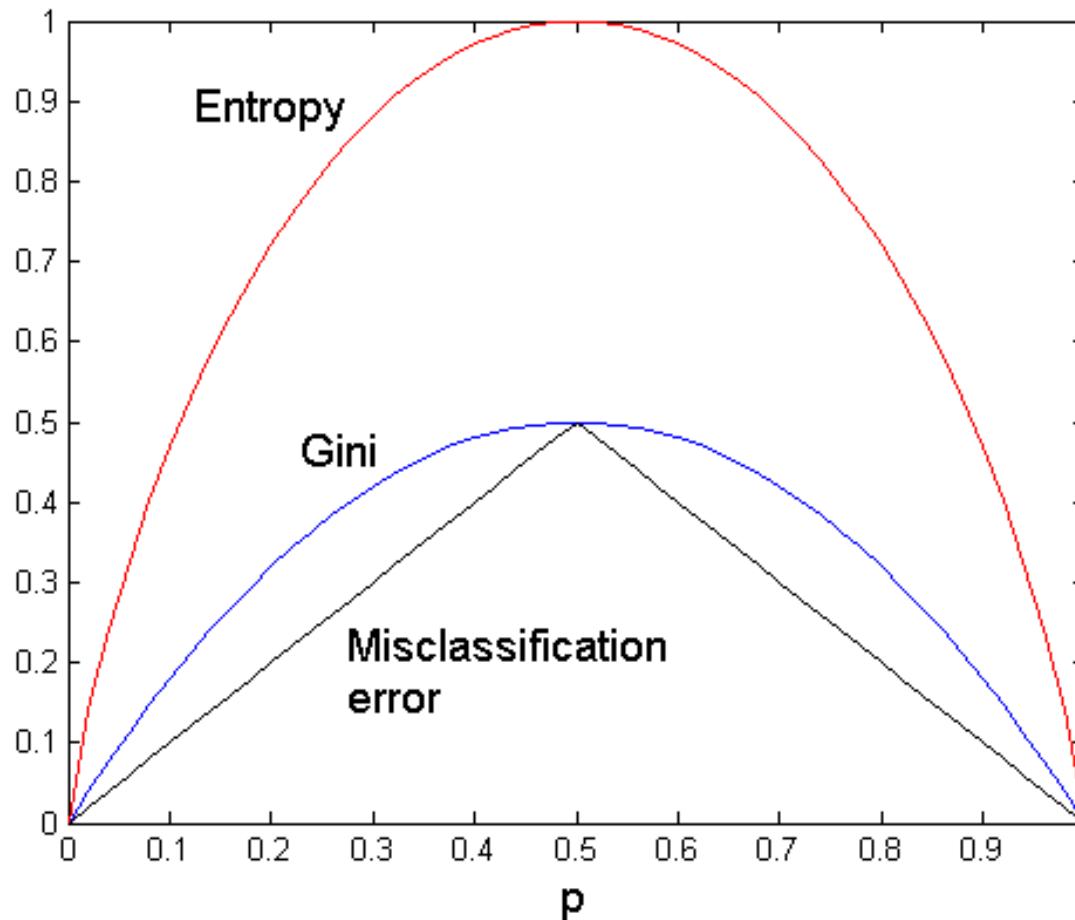
C1	2
C2	4

$$P(C1) = 2/6 \quad P(C2) = 4/6$$

$$\text{Entropy} = -(2/6) \log_2 (2/6) - (4/6) \log_2 (4/6) = 0.92$$

Comparing homogeneity metrics

For binary target variable ($c=2$)



The goodness of a split – the gain

- How much do we gain by the split?
 - Gain (Δ)

$$\Delta = I(\text{parent}) - \sum_{i=1}^k \frac{n_i}{n} I(\text{child}_i)$$

- $I()$: one of the three inhomogeneity metrics
- n_i : number of records in child node i , n : number of records in the parent node
 - We weight the inhomogeneity of the child nodes by their relative size

Problem

- The following table summarizes a data set with three binary attributes (A, B, C) and two class labels (+, -).
- Using misclassification error as inhomogeneity measure, calculate the gains for splitting on each attribute. Which attribute gives the best split?

A	B	C	Number of instances	
			class: +	class: -
T	T	T	5	0
F	T	T	0	20
T	F	T	20	0
F	F	T	0	5
T	T	F	0	0
F	T	F	25	0
T	F	F	0	0
F	F	F	0	25

Splitting by a continuous attribute

- Where to split?
 - Sort the records according to the attribute values
 - Calculate the inhomogeneity metric one by one increasing the value of the cut point
 - Which cut-point corresponds to the highest gain?

Class	No	No	No	Yes	Yes	Yes	No	No	No	No		
Annual Income												
Sorted Values →	60	70	75	85	90	95	100	120	125	220		
Split Positions →	55	65	72	80	87	92	97	110	122	172	230	
	<=	>	<=	>	<=	>	<=	>	<=	>	<=	>
Yes	0	3	0	3	0	3	1	2	2	1	3	0
No	0	7	1	6	2	5	3	4	3	4	4	3
Gini	0.420	0.400	0.375	0.343	0.417	0.400	0.300	0.343	0.375	0.400	0.420	

Should we split on ID?

- Δ gain would be the highest if we split on ID
- The method thus prefers to split the parent node to many small nodes
- It is usually not favorable
 - E.g. if we split on ID, that is useless
 - If the emergent child nodes are too small the model is more likely to have a worse generalization ability
- Possible solutions
 - Only allow for binary splits
 - Filter out those attributes that are not reasonable to split on
 - Instead of Δ gain use another method to measure the goodness of split: gain ratio

Building a decision tree

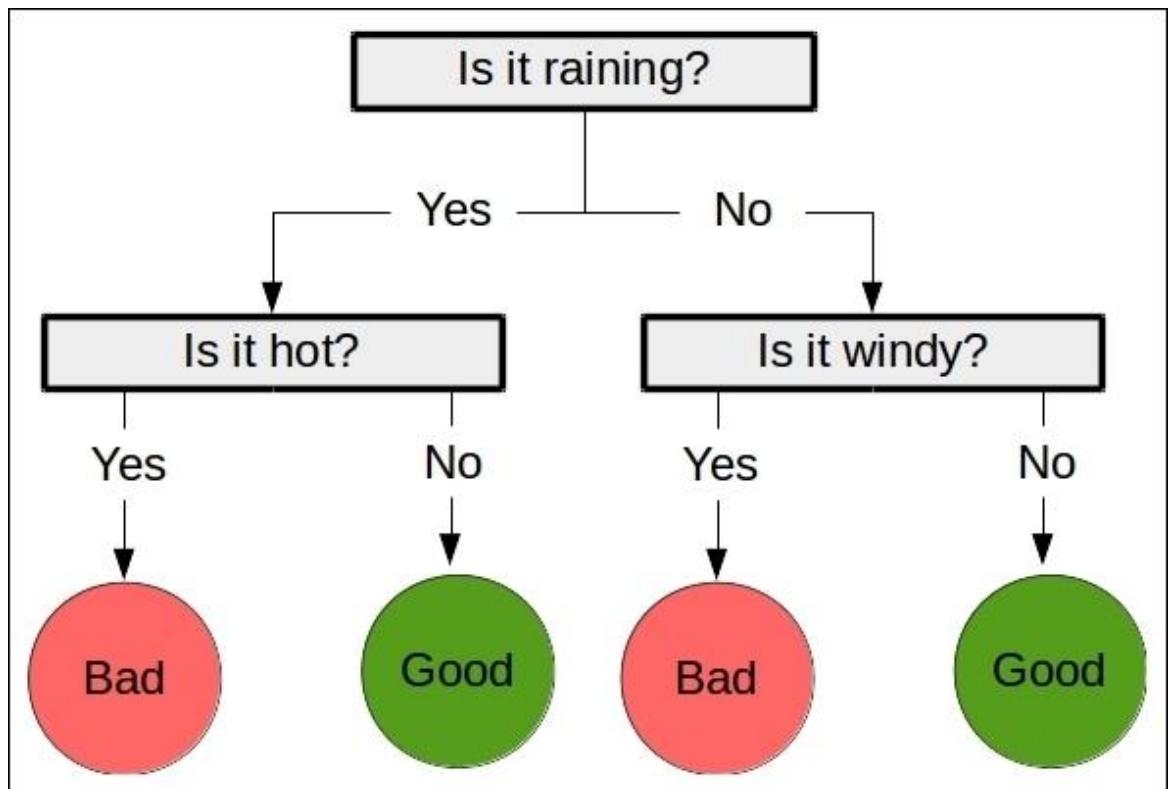
- Relatively fast, easy to interpret
- After building the decision tree, it is fast to predict new labels for unseen data points
- There are not many hyperparameter to set (but there are some)
 - What is the measure of inhomogeneity?
 - Do we allow for multi-split or just for binary?
 - How to traverse nodes?

Termination criteria

- If every node is homogeneous
 - Or every node is quasi-homogeneous (almost homogeneous)
- All the records agree in each attribute (aside from the label)
- Global termination criteria:
 - Bound for the number of leaf nodes
 - Bound for the number of levels (depth of the tree)
- We do not search for the „best” tree
 - It is not practical since it is just the „best” on the training data set

Advantages of decision tree

- Easy to interpret
- It works with a similar approach as human decision process
- Easy to visualize
- Insensitive to irrelevant attributes
- It can be used for a wide variety of problems (for classification/regression, with numerical/categorical attributes)



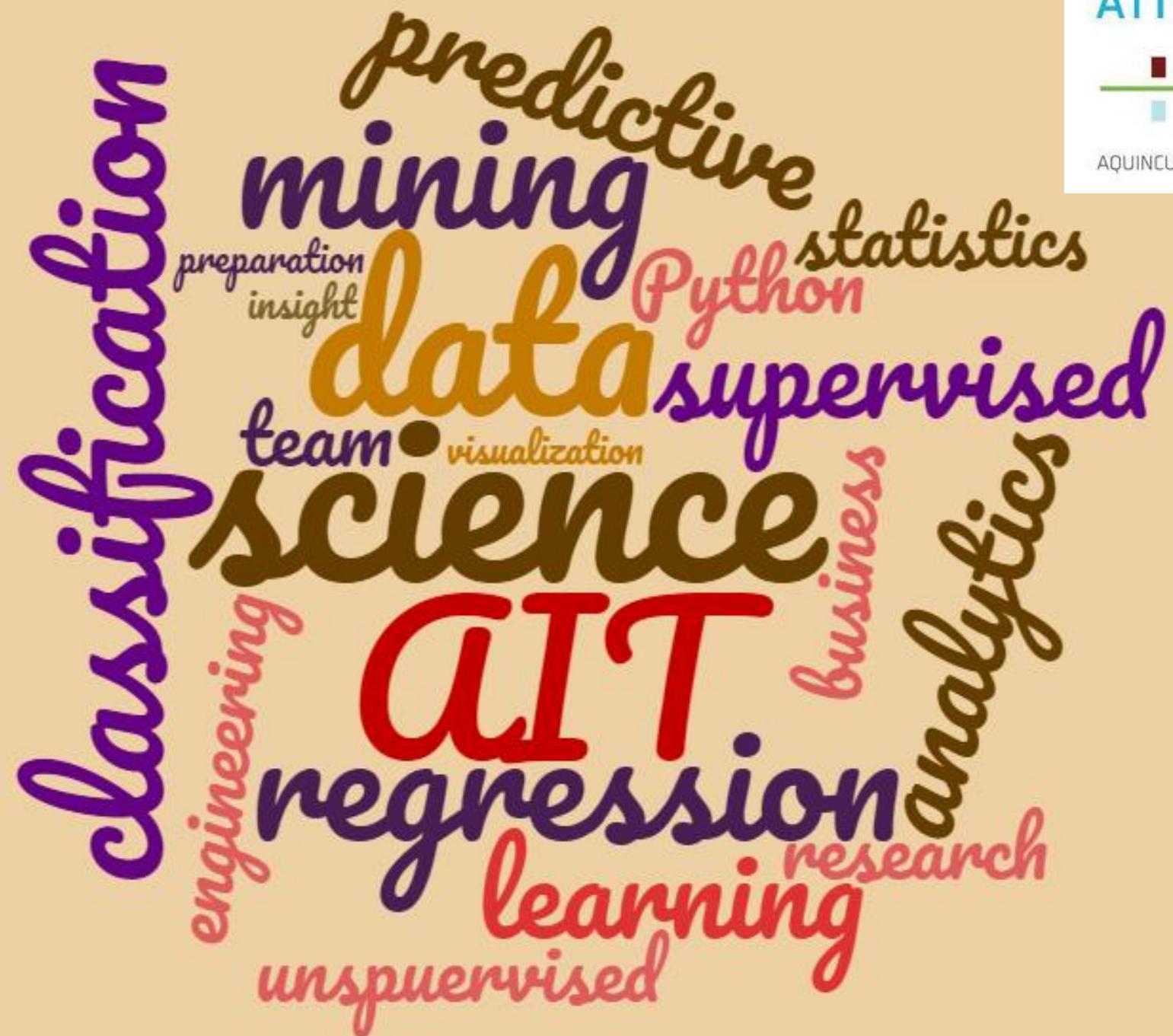
Acknowledgement

- András Benczúr, Róbert Pálovics, SZTAKI-AIT, DM1-2
- Krisztián Buza, MTA-BME, VISZJV68
- Bálint Daróczy, SZTAKI-BME, VISZAMA01
- Judit Csima, BME, VISZM185
- Gábor Horváth, Péter Antal, BME, VIMMD294, VIMIA313
- Lukács András, ELTE, MM1C1AB6E
- Tim Kraska, Brown University, CS195
- Dan Potter, Carsten Binnig, Eli Upfal, Brown University, CS1951A
- Erik Sudderth, Brown University, CS142
- Joe Blitzstein, Hanspeter Pfister, Verena Kaynig-Fittkau, Harvard University, CS109
- Rajan Patel, Stanford University, STAT202
- Andrew Ng, John Duchi, Stanford University, CS229



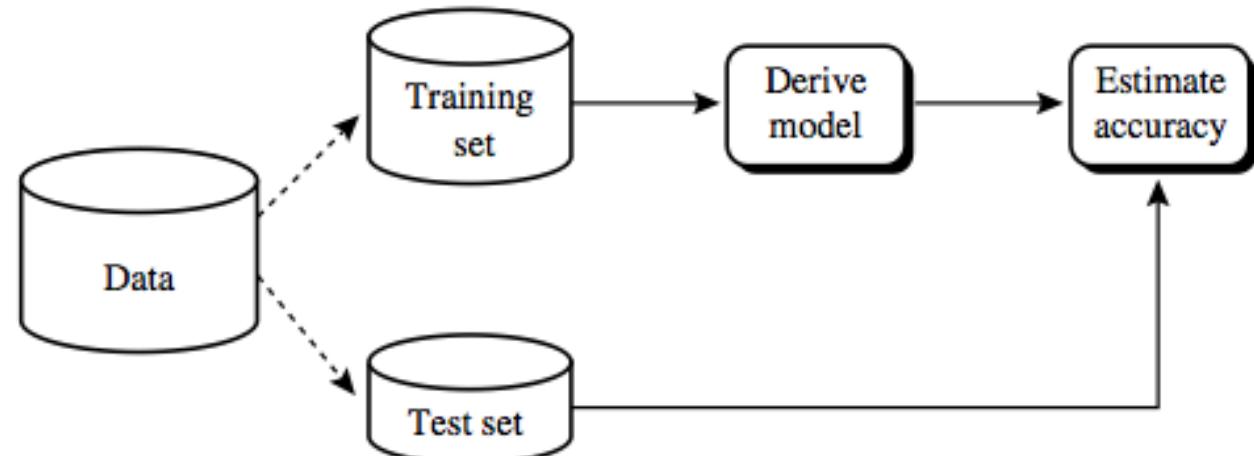
Data Science

March 7, 2023
Overfitting, model evaluation



Generalization ability

- Purpose: to build a model that predicts the target variable well in general not just on the available data set → good generalization ability
- Dataset is divided into two (or three) parts
- Cross validation
- To evaluate models, a numerical „goodness” notion is needed



Which model is the best?

- Scenario:
 - We aim to solve a supervised learning task
 - We have training data
 - We aim to give prediction on a target variable
 - We have candidate algorithms (models) to solve the problem
- Question:
 - How to choose which model (algorithm) to use?

Algorithm of Success

```
while(noSuccess)
{
    tryAgain();
    if(Dead)
        break;
}
```

Training and test set

First approach: split the data set into two parts

- Training set: fitting the model (i.e. optimizing its parameters) on the training set in such a way that it has a good performance on the training set and has a good generalization ability
- Test set: we test the model performance on data that were not seen by the model before
 - We choose the model that has the best performance on the test set

Problems:

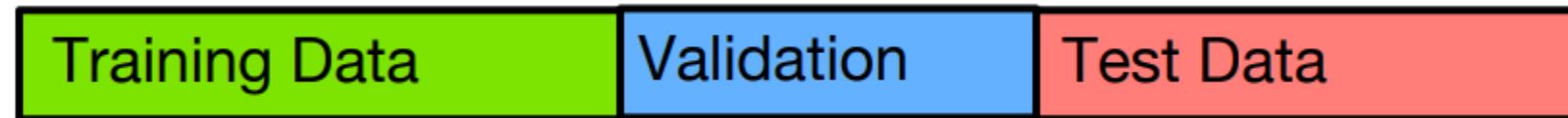


- It overestimates the test performance of the best model („lucky” model)
- The algorithm never has access to the test data

Using validation set

Second approach: split the data into three parts

- Training, validation and test data
- Fitting the models on the training data
- Evaluating the models on the validation data
 - Choosing the best performing model
- The chosen model is also evaluated on the test data

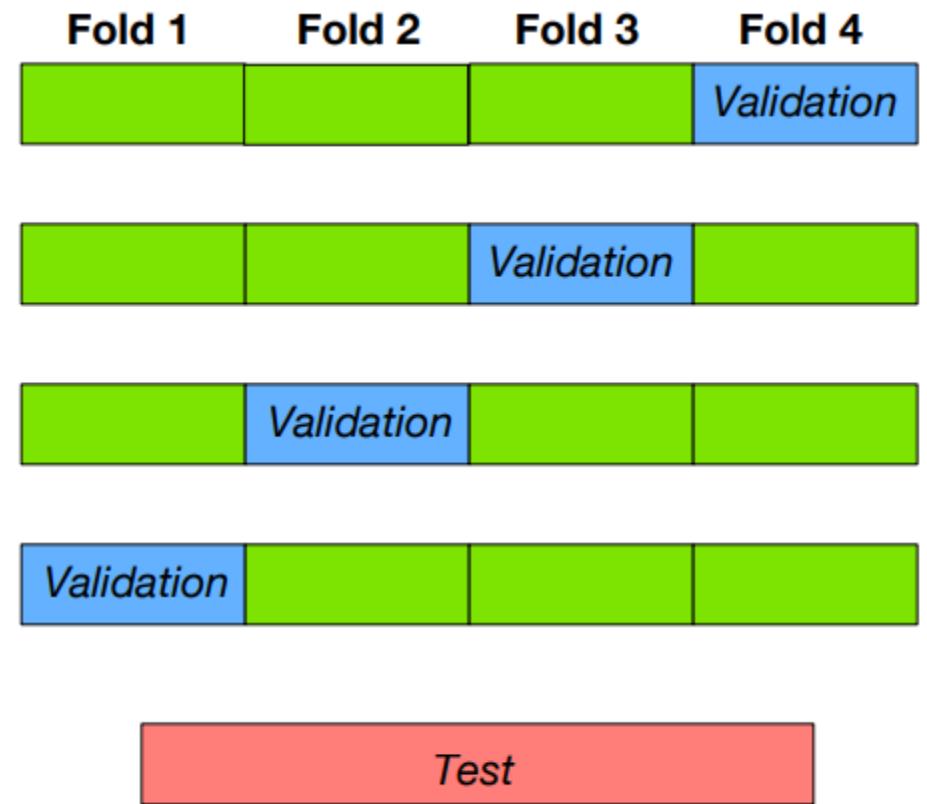


- Problems:

- Wasting training data (neither the validation set nor the test set can be used for training)

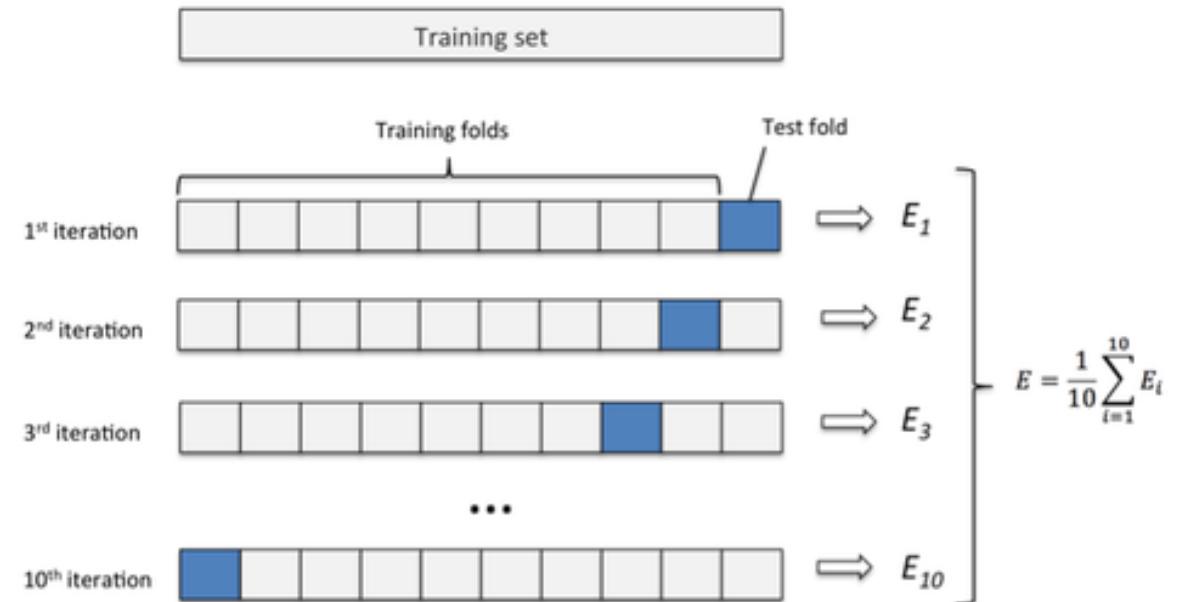
Cross validation

- Split the training data for K equal parts (folds)
- Fit the models on $K-1$ parts and evaluate them on the rest
 - Do it in all possible K -ways
- We choose the model according to the best average performance of the K evaluations
- The chosen model can be evaluated again on a separate test set



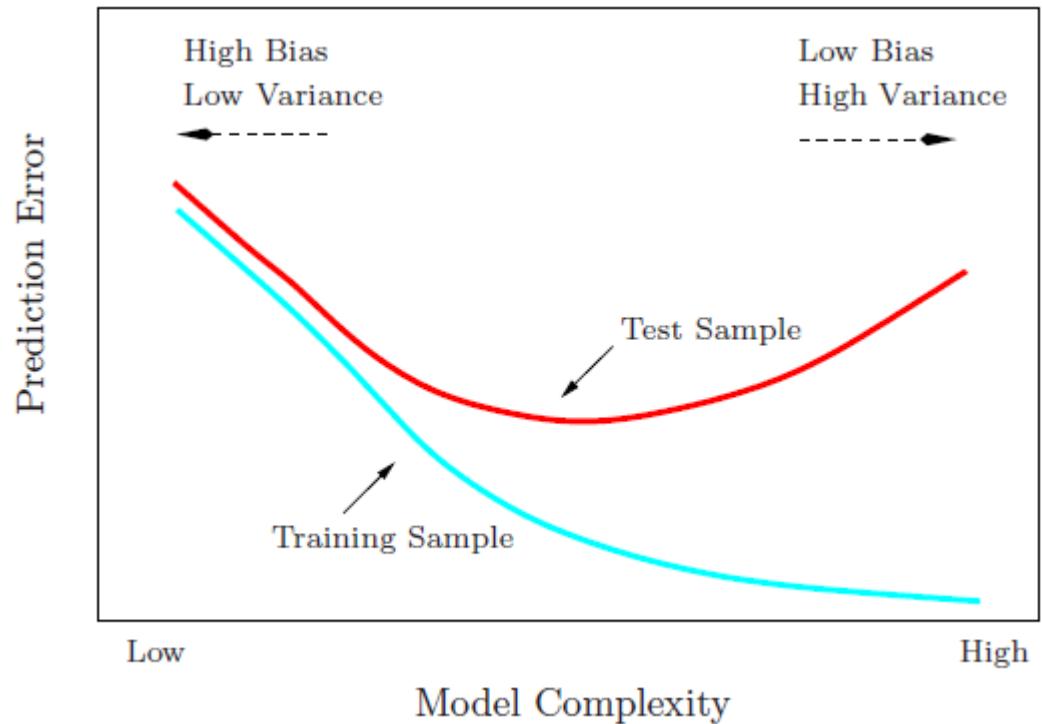
K-fold cross validation

- How to choose K ?
 - If K is too small: the model is fitted on a small data set that leads to high bias
 - If K is too large: Few data points are used for evaluation, the performance estimation is noisy that leads to high variance
 - $K=N$: „leave-one-out“ cross-validation
 - In practice: 5- or 10- fold cross validation is typical
- The learning algorithm has to be run K times but it can be parallelized



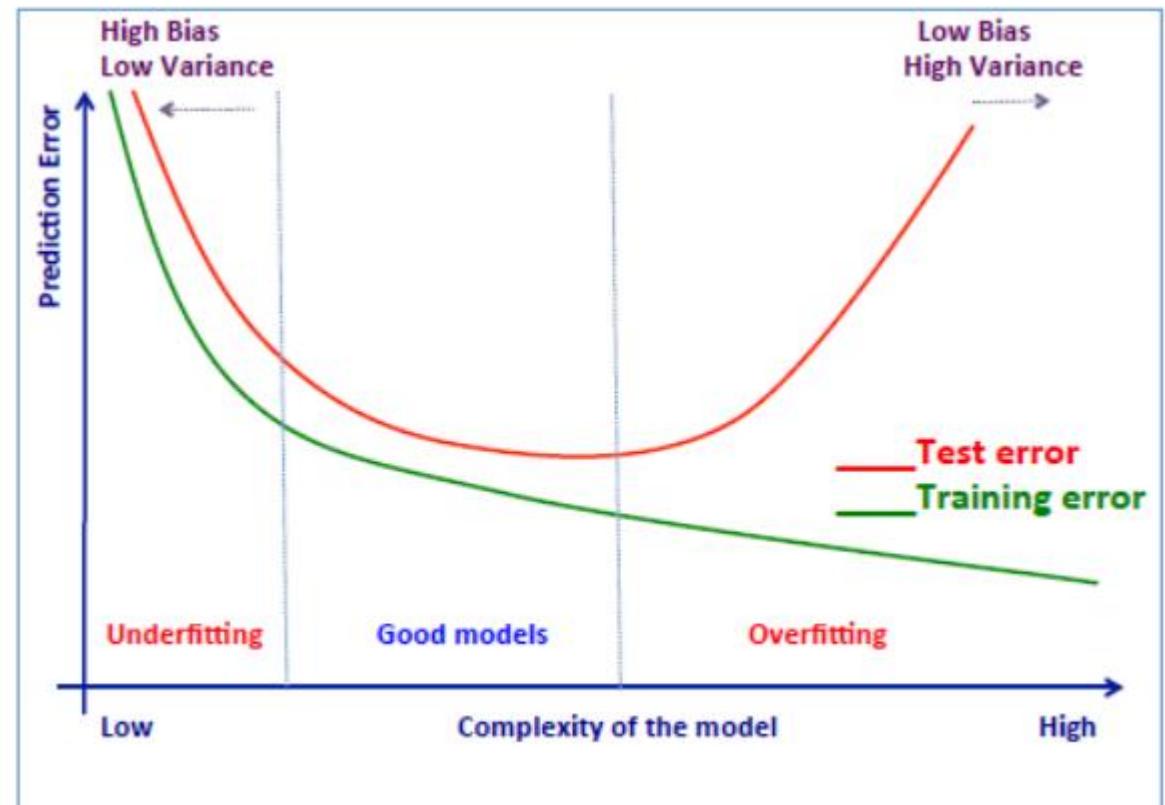
Underfitting - overfitting

- Underfitting: the model is not sophisticated enough to make good predictions
 - Error rate is high both on training data and on test data
 - We should build a more complex model (maybe more training data is needed)
- Overfitting: the model corresponds too closely to a particular set of data and doesn't generalize well
 - Reduce the complexity of the model



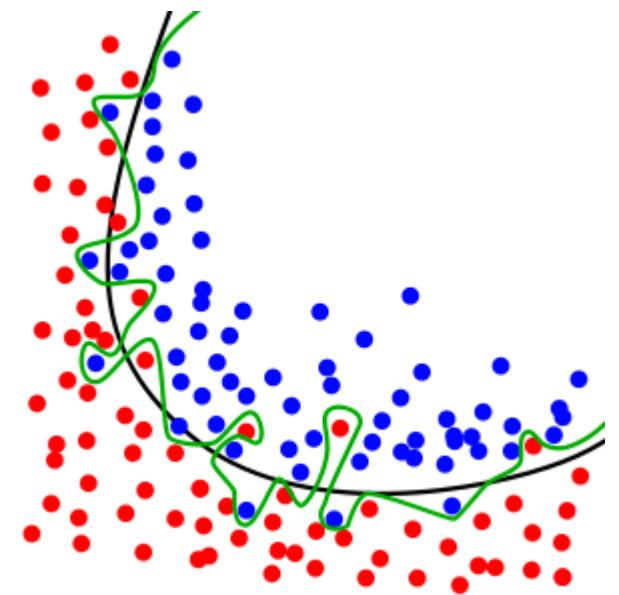
Under- and overfitting

- We say that model M underfits, if $\exists M'$ such that:
 - $\text{error}_{\text{training}}(M) > \text{error}_{\text{training}}(M')$
 - $\text{error}_{\text{test}}(M) > \text{error}_{\text{test}}(M')$
- We say that model M overfits if $\exists M'$ such that
 - $\text{error}_{\text{training}}(M) < \text{error}_{\text{training}}(M')$
 - $\text{error}_{\text{test}}(M) > \text{error}_{\text{test}}(M')$



Causes of overfitting

- Indirect cause: the model is too sophisticated, too complex, contains more parameters than can be justified by the data
- Direct cause:
 - Non-representative data (great number of special cases)
 - Noisy data
 - We choose from too many models: the best will have a good performance on the data set that we evaluate on with no warranty that it will have a similarly good performance on other data



Occam's razor



- Simpler solutions are more likely to be correct than complex ones
- Among competing hypothesis one should select the solution with the fewest assumption
- The principle is also used in machine learning:
Build models in such a way that it penalizes too complex models

The principle is attributed to the medieval philosopher William of Occam

Confusion matrix

- Confusion matrix for binary classification problems:

		Predicted class	
		P	N
		True Positives (TP)	False Negatives (FN)
P	Actual Class	True Negatives (TN)	False Positives (FP)
	N	False Negatives (FN)	True Positives (TP)

Performance indicators for binary classification

- Accuracy: $A = \frac{TP + TN}{TP + TN + FP + FN}$
 - It can be weighted as well
 - Not a good measure for unbalanced classes
- Precision: $P = \frac{TP}{TP + FP}$
 - The ratio of correctly predicted positive observations to the total predicted positive observations
- Recall (sensitivity, TPR): $R = \frac{TP}{TP + FN}$
 - The ratio of correctly predicted positive observations to the all observations in actual positive class
- F1 score (F-measure): $F = \frac{2 \cdot P \cdot R}{P + R} = \frac{2 \cdot TP}{2 \cdot TP + FP + FN}$
 - Harmonic mean of Precision and Recall

Further performance indicators

		Predicted condition		Prevalence $= \frac{\sum \text{Condition positive}}{\sum \text{Total population}}$	
Total population		Predicted Condition positive	Predicted Condition negative		
True condition	condition positive	True positive	False Negative (Type II error)	True positive rate (TPR), Sensitivity, Recall $= \frac{\sum \text{True positive}}{\sum \text{Condition positive}}$	False negative rate (FNR), Miss rate $= \frac{\sum \text{False negative}}{\sum \text{Condition positive}}$
	condition negative	False Positive (Type I error)	True negative	False positive rate (FPR), Fall-out $= \frac{\sum \text{False positive}}{\sum \text{Condition negative}}$	True negative rate (TNR), Specificity (SPC) $= \frac{\sum \text{True negative}}{\sum \text{Condition negative}}$
Accuracy (ACC) = $\frac{\sum \text{True positive} + \sum \text{True negative}}{\sum \text{Total population}}$		Positive predictive value (PPV), Precision $= \frac{\sum \text{True positive}}{\sum \text{Test outcome positive}}$	False omission rate (FOR) $= \frac{\sum \text{False negative}}{\sum \text{Test outcome negative}}$	Positive likelihood ratio (LR+) $= \frac{\text{TPR}}{\text{FPR}}$	Diagnostic odds ratio (DOR) $= \frac{\text{LR+}}{\text{LR-}}$
		False discovery rate (FDR) $= \frac{\sum \text{False positive}}{\sum \text{Test outcome positive}}$	Negative predictive value (NPV) $= \frac{\sum \text{True negative}}{\sum \text{Test outcome negative}}$	Negative likelihood ratio (LR-) $= \frac{\text{FNR}}{\text{TNR}}$	

Problem

The following table summarizes a data set with three attributes (A, B, C) and two class labels (+, -). Build a two-level decision tree.

1. Use misclassification error as inhomogeneity measure. Calculate the gains for each attribute. Which attribute gives the best split?
2. Repeat the previous step for the two children of the root node. Which nodes should be split, and which is the second splitting attribute?
3. Calculate Accuracy, Error rate, Precision, Recall, and F-measure!
4. Choose C as the first splitting attribute and continue building the tree! How a tree of depth 2 would look like in that case?

A	B	C	Number of instances	
			class: +	class: -
T	T	T	5	0
F	T	T	0	20
T	F	T	20	0
F	F	T	0	5
T	T	F	0	0
F	T	F	25	0
T	F	F	0	0
F	F	F	0	25

Binary classification using confidence scores

- Until now we emphasized that the classifier predicts the label (0/1)
- In most cases we can supplement the prediction with a probability score (confidence score)
 - For k NN algorithm: it is the ratio of observations with label 1 among the k nearest neighbor of the data point
 - If we use the weighted version, then the weights are taken into consideration as well
 - For decision tree: the ratio of observations with label 1 among the observations in the leaf of the data point
- Usually we predict label 1 if the score is above 0.5
 - The threshold can be set other values, it depends if we value Type 1 error / Type 2 error more

ROC

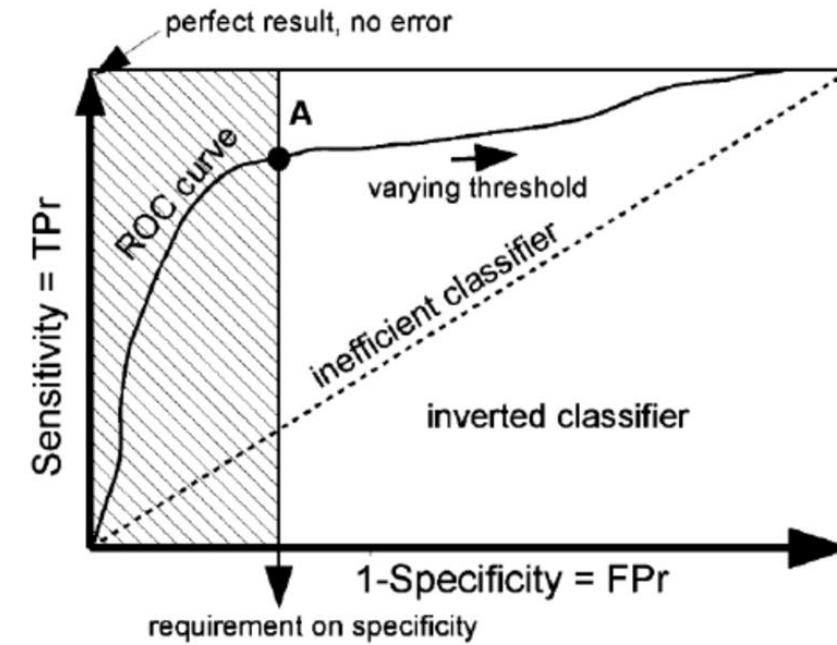
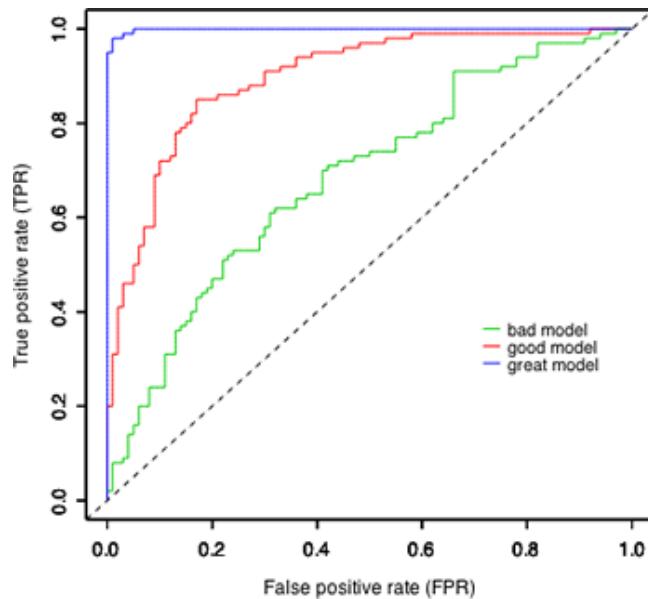
- Receiver Operating Characteristic
 - It has its origins in signal processing (1950s)
- Sort the test data according to their probability scores
 - Above a certain threshold we predict 1 (below that we predict 0)
- For all possible threshold values we determine the FPR and TPR rates
- ROC curve: plotting TPR against FPR at various threshold settings

$$TPR = \frac{TP}{TP + FN}$$

$$FPR = \frac{FP}{TN + FP}$$

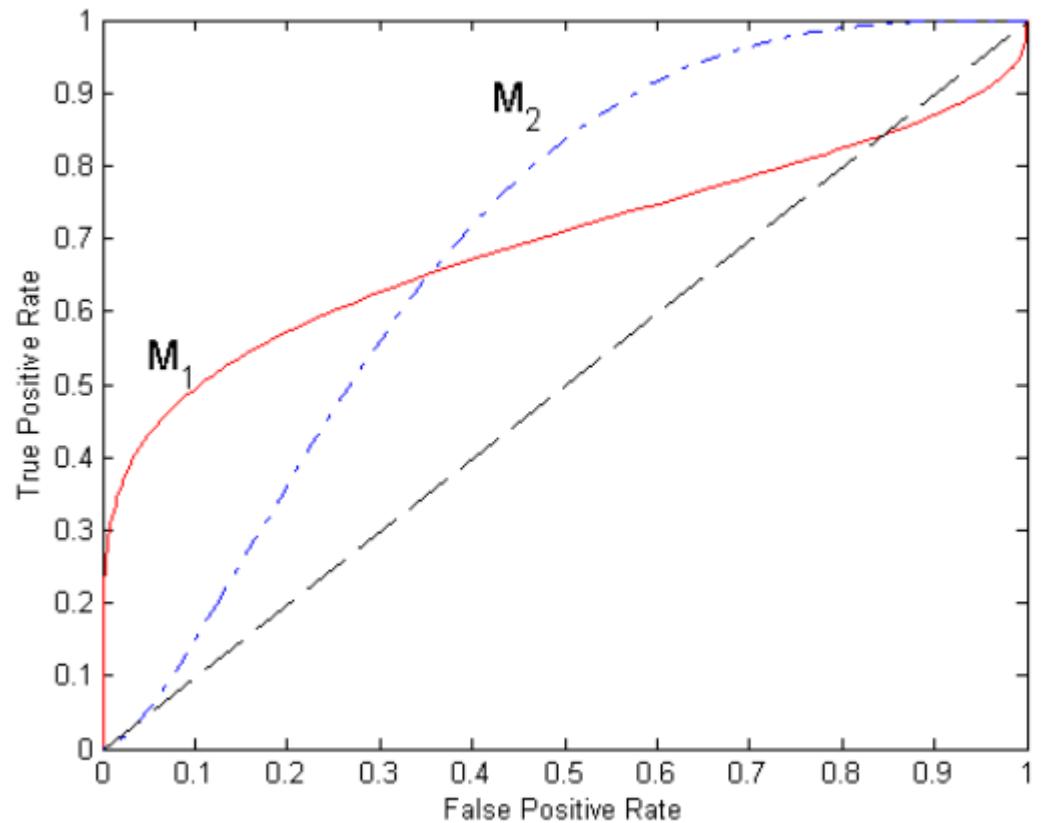
ROC curve

- (1,1): every instance is classified as positive
- (0,0): every instance is classified as negative
- (0,1): perfect classifier
- Diagonal line ($y=x$): random classifier
 - Below the diagonal: even worse than random classifier
 - The predicted labels should be inverted



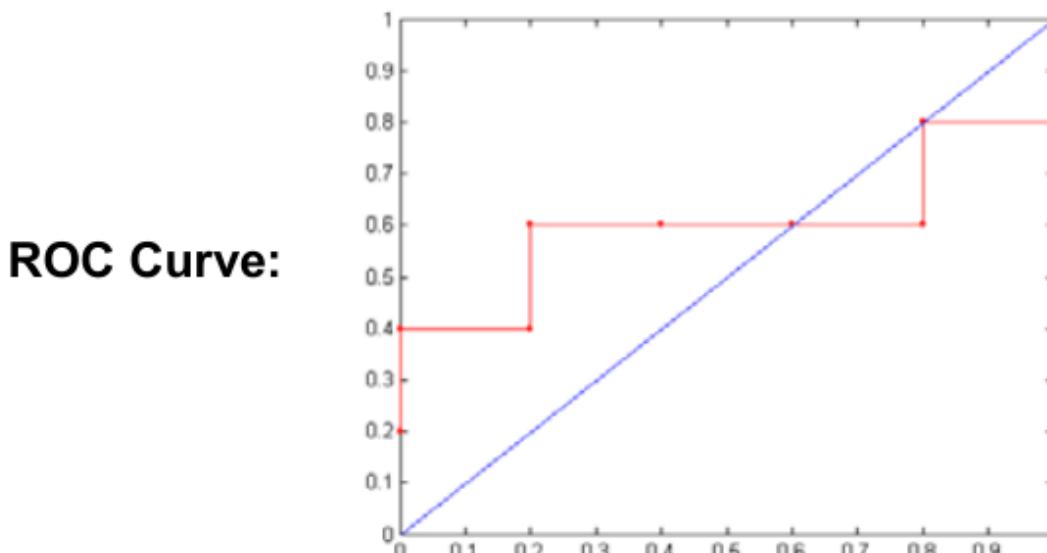
ROC for comparing models

- None of the models can be considered consistently better than the other
 - M_1 is better for small FPR values
 - M_2 is better for large FPR values



Designing the ROC curve

Class	+	-	+	-	-	-	+	-	+	+	
Threshold >=	0.25	0.43	0.53	0.76	0.85	0.85	0.85	0.87	0.93	0.95	1.00
TP	5	4	4	3	3	3	3	2	2	1	0
FP	5	5	4	4	3	2	1	1	0	0	0
TN	0	0	1	1	2	3	4	4	5	5	5
FN	0	1	1	2	2	2	2	3	3	4	5
→ TPR	1	0.8	0.8	0.6	0.6	0.6	0.6	0.4	0.4	0.2	0
→ FPR	1	1	0.8	0.8	0.6	0.4	0.2	0.2	0	0	0

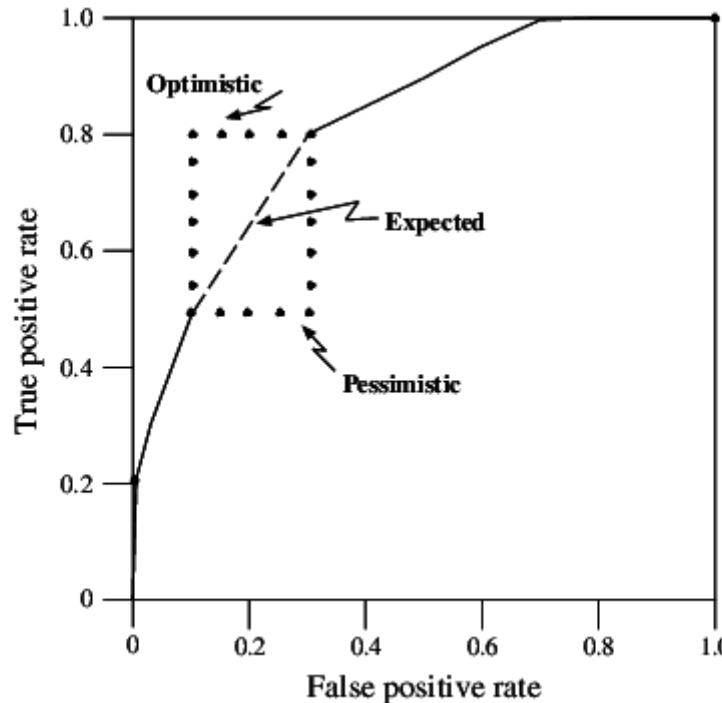


$$TPR = \frac{TP}{TP + FN}$$

$$FPR = \frac{FP}{TN + FP}$$

ROC – equal probability scores with different labels

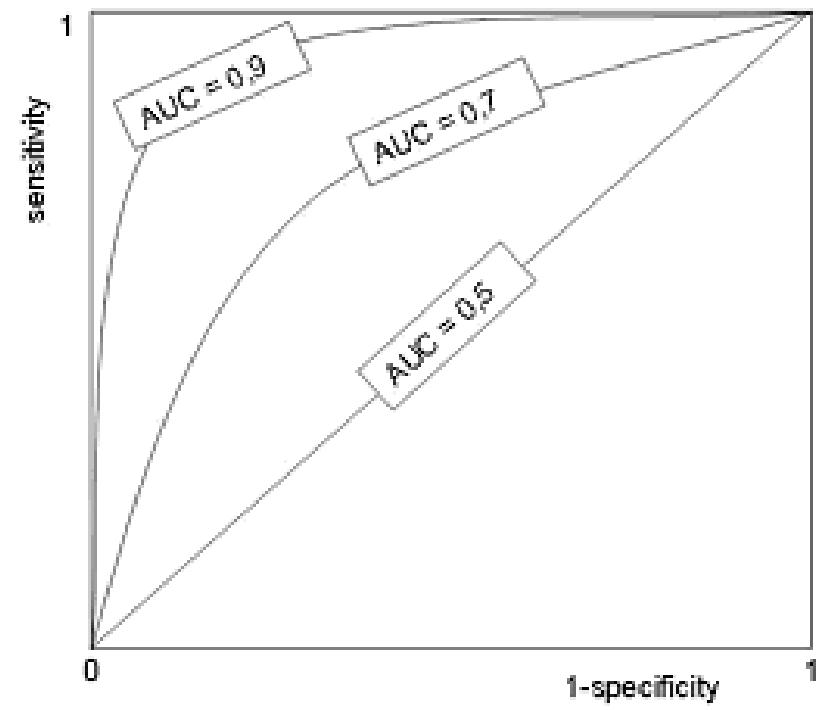
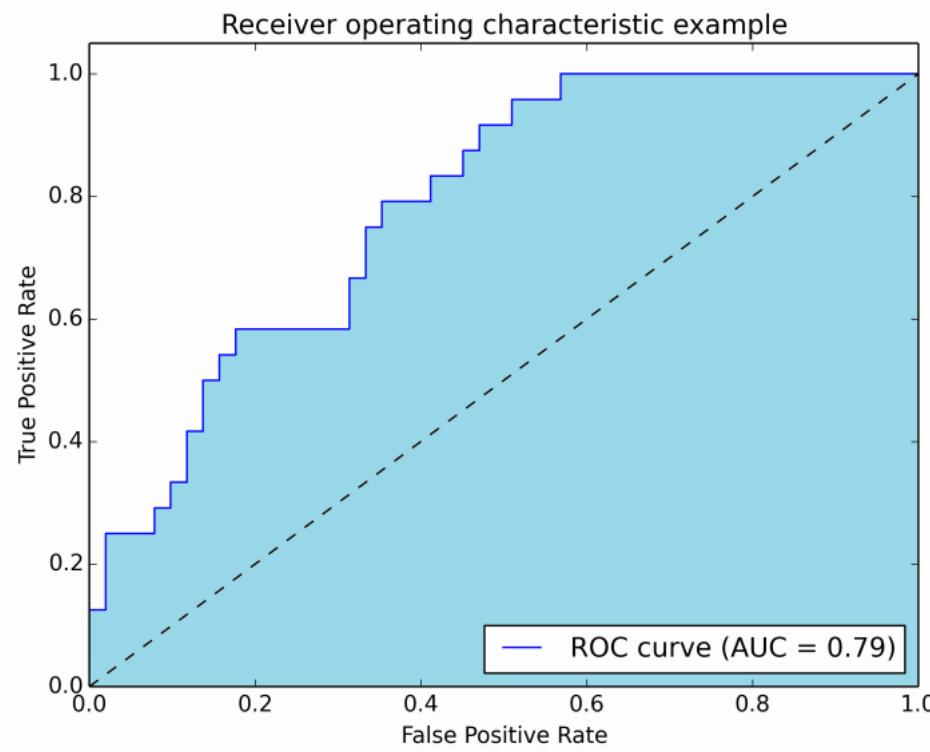
- It may happen that records with different labels have equal probability scores
 - In this case the ROC curve has diagonal segments



AUC

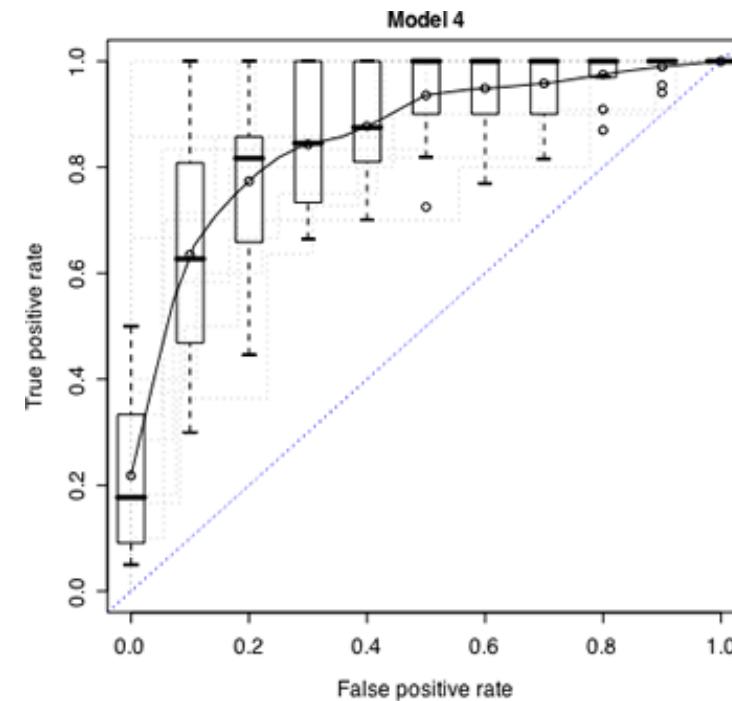
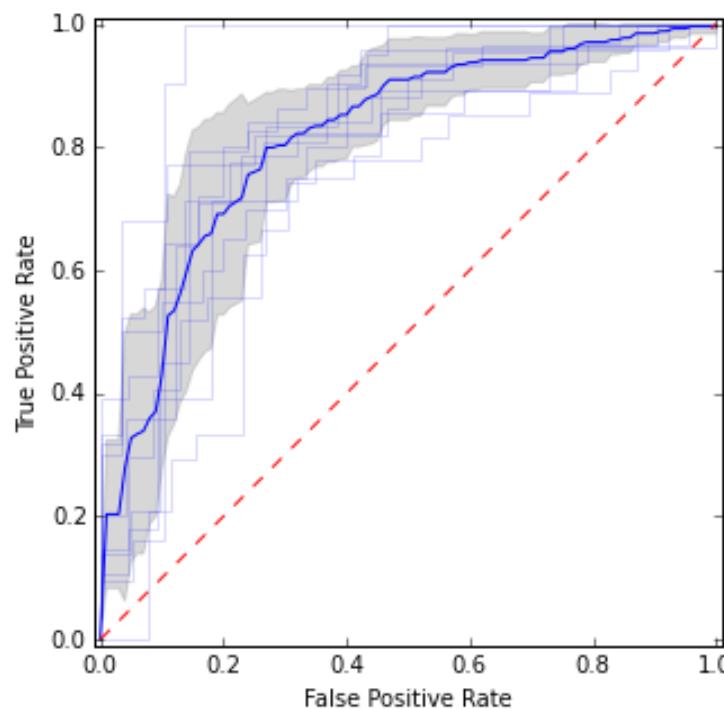
- **Area Under ROC Curve**
 - Its maximal value: 1 (perfect classifier)
 - For random classifier: 0.5
- It is the probability that a randomly-chosen positive record is ranked more highly than a randomly chosen negative record
- It is only defined for binary classification
- It is independent from the threshold – more stable performance indicator

ROC, AUC



Cross-validation with ROC, AUC

- If we cross-validate the data we have more validation sets, so ROC can be determined each time
 - Mean values and variation can be calculated



ROC for multi-class classification

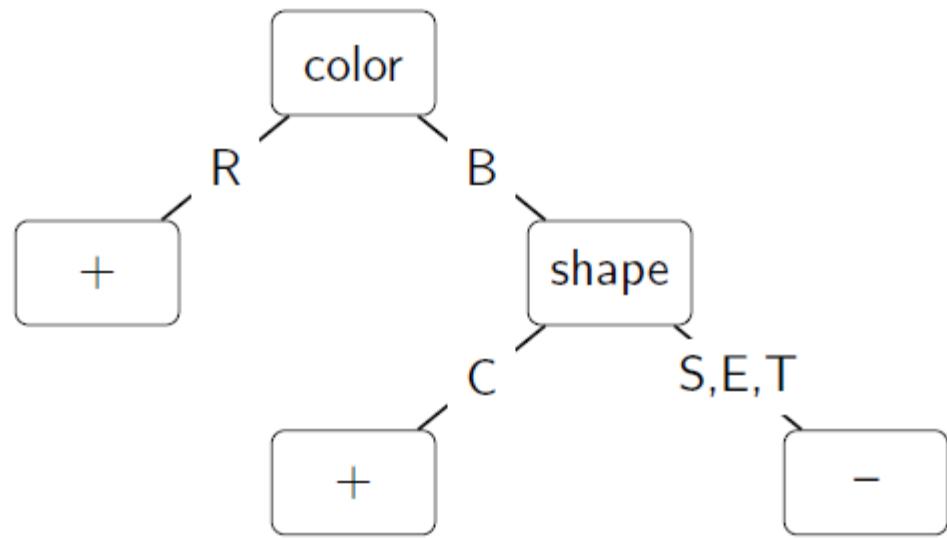
- There are various generalizations of the ROC concept for multi-class classification
 - No canonical approach
 - We omit this here

Problem

Construct the ROC curve of the following classifier and calculate the AUC. How do you interpret the AUC score? What would you suggest in terms of results? In the table confidence scores increase from left to right.

You can see a schematic diagram of a possible decision tree built on training data below.

1. Determine the confidence scores (ratio of positive observations) of the leaves based on the training data (train1, train2, ..., train7).
2. Sort the confidence scores of first three test instances (test1, test2, test3) in ascending order.
3. Construct an ROC curve using the first three instances of the test data (test1, test2, test3) and calculate the AUC.
4. Construct the ROC curve after adding two new test data (test4, test5). If more instances have the same confidence scores ROC curve may change diagonally!



ID	Shape	Color	Size	Class
train1	S	R	L	+
train2	C	R	H	+
train3	C	B	H	+
train4	T	R	L	+
train5	S	B	M	-
train6	E	B	L	-
train7	C	R	M	-

ID	Shape	Color	Size	Class
test1	C	R	H	+
test2	C	B	L	-
test3	E	B	H	-
test4	C	R	L	+
test5	E	R	H	-

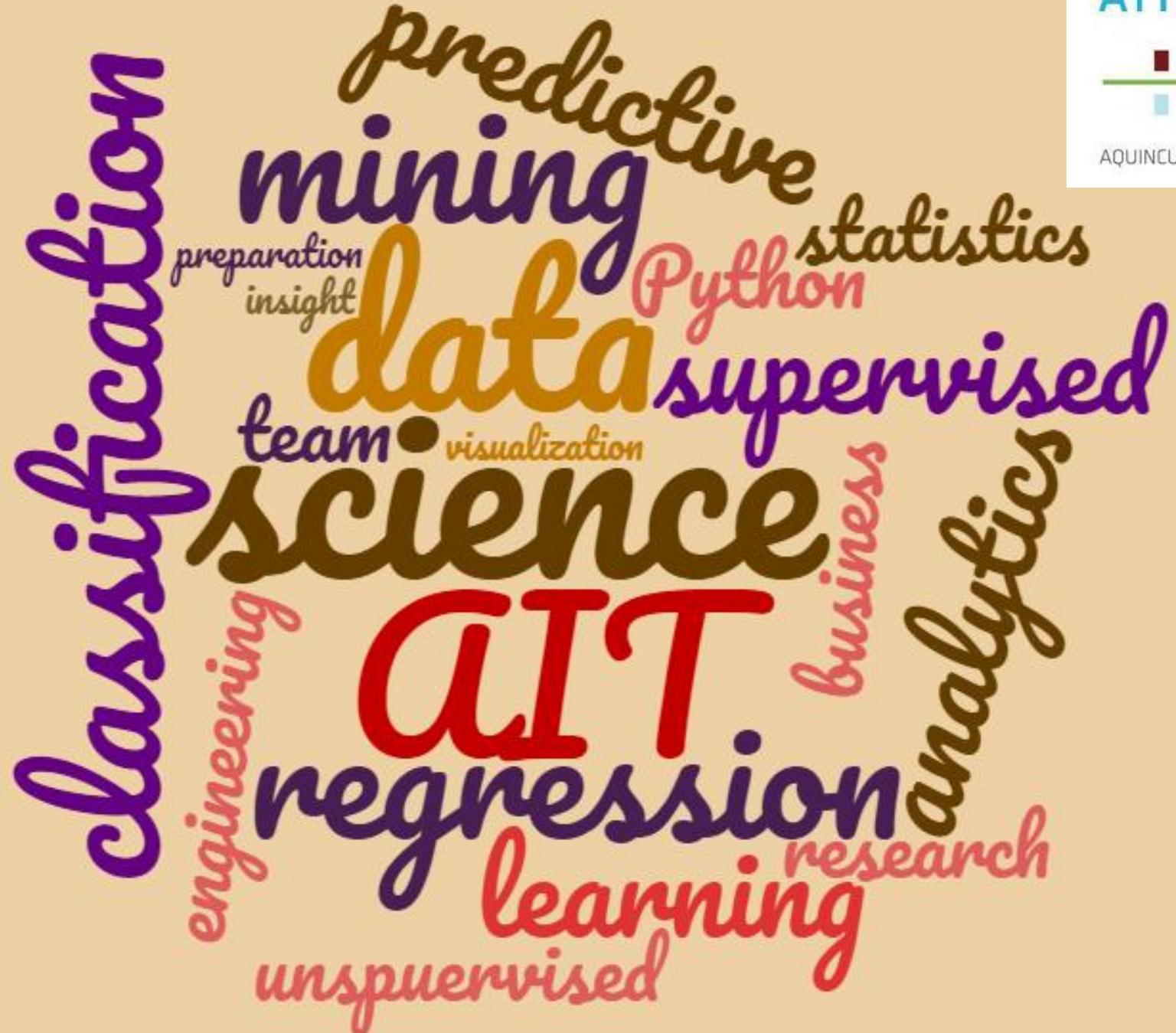
Data leakage

- The use of information in the model training process which would not be expected to be available at prediction time → predictive metrics overestimate the model's utility when run in a production environment
 - Feature leakage
 - The inclusion of columns that will not be available when the model is used for predictions, and result in leakage if included when the model is trained
 - E.g.: "MonthlySalary" column when predicting "YearlySalary"; or "MinutesLate" when predicting "IsLate"; or more subtly "NumOfLatePayments" when predicting "ShouldGiveLoan"
 - Premature featureization (e.g. feature scaling parameters calculation before train-test split)
 - Duplicate rows between train/validation/test (e.g. oversampling before train-test split)
 - Group leakage
 - E.g. 3 X-ray images per patient, all images of a patient should be in the same split, therefore the model cannot memorize the patients instead of learning to recognize covid-19
 - Time leakage
 - Splitting a time-series dataset randomly instead of newer data in test set

Acknowledgement

- András Benczúr, Róbert Pálovics, SZTAKI-AIT, DM1-2
- Krisztián Buza, MTA-BME, VISZJV68
- Bálint Daróczy, SZTAKI-BME, VISZAMA01
- Judit Csima, BME, VISZM185
- Gábor Horváth, Péter Antal, BME, VIMMD294, VIMIA313
- Lukács András, ELTE, MM1C1AB6E
- Tim Kraska, Brown University, CS195
- Dan Potter, Carsten Binnig, Eli Upfal, Brown University, CS1951A
- Erik Sudderth, Brown University, CS142
- Joe Blitzstein, Hanspeter Pfister, Verena Kaynig-Fittkau, Harvard University, CS109
- Rajan Patel, Stanford University, STAT202
- Andrew Ng, John Duchi, Stanford University, CS229





Schedule of the semester

	<i>Monday midnight</i>	<i>Tuesday class</i>	<i>Friday class</i>
W1 (02/06)			
W2 (02/13)		HW1 out	
W3 (02/20)			
W4 (02/27)	HW1 deadline + TEAMS	HW2 out	
W5 (03/06)			PROJECT PLAN
W6 (03/13)	HW2 deadline	HW3 out	
W7 (03/20)			MIDTERM
SPRING BREAK		SPRING BREAK	SPRING BREAK
W8 (04/03)	HW3 deadline		GOOD FRIDAY
W9 (04/10)	MILESTONE 1	HW4 out	
W10 (04/17)			
W11 (04/24)	HW4 deadline		
W12 (05/01)	MILESTONE 2		
W13 (05/08)			
W14 (05/15)		FINAL	PROJECT presentations
W15 (05/22)		PROJECT presentations	

Requirements

- Attendance: there will be sign-up sheets every time
- MIDTERM (25%)
 - On Week 7
 - You can use your own „cheat sheets”/ formula sheets / notes
- FINAL (25%)
 - On Week 13
 - You can use your own „cheat sheets”/ formula sheets /notes
- HOMEWORK problems (25%)
 - There will be 4 HW sheets (in every two weeks)
 - Mostly programming problems
- PROJECTS in teams (25%)
 - In teams of 3 (2-4) students



Reminder

- W4: Team name + list of team members ✓
- W6: **Project plan**
 - One page long report answering the following questions:
 - What is the vision? Why is the problem interesting?
 - What is the purpose of the project? What results do you expect?
 - What data do you plan to use? How do you plan to gather the data?
 - Are the data big enough and of suitable quality?
 - What data preparation steps do you plan to take?
 - What methodology, what models do you plan to use?
 - How would you visualize the results?

Principle of Bayes classifier

- The attributes and class label are considered as random variables
 - Let C denote the class label and A_1, A_2, \dots, A_d denote the attributes
 - Earlier we use Y for the target variable (label) and X_i for the attributes
 - First, for the sake of simplicity we consider the attributes as categorical variables
- We learnt that the misclassification error is minimal if we choose the class label to maximize $P(C | A_1, A_2, \dots, A_d)$ conditional probability
 - $c^* = \operatorname{argmax}_c P(C = c | \underline{A} = \underline{a})$
 - Maximum *a posteriori estimation*
- Can we estimate the value of $P(C | A_1, A_2, \dots, A_d)$ from the data?

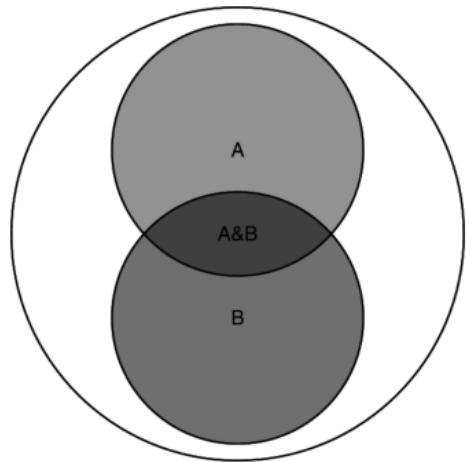
Quick revision of necessary notions

- Conditional probability

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

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

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



- Bayes theorem:

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

- Law of total probability ($\{B_i : i = 1, 2, 3, \dots\}$ is a partition of the sample space)

$$P(A) = \sum_i P(A, B_i) = \sum_i P(A|B_i)P(B_i)$$

Using Bayes theorem for classification

- Use Bayes theorem:

$$c^* = \operatorname{argmax}_c P(C = c \mid \underline{A} = \underline{a}) = \operatorname{argmax}_c \frac{P(\underline{A} = \underline{a} \mid C = c) \cdot P(C = c)}{P(\underline{A} = \underline{a})} = \\ = \operatorname{argmax}_c P(\underline{A} = \underline{a} \mid C = c) \cdot P(C = c)$$

- The first identity uses Bayes theorem. The second identity is valid because the denominator does not depend on the class label, it is the same for any C values
- If $P(C = c)$ is constant for any class label, i.e. the *a priori* probabilities agree for all labels, then

$$c^* = \operatorname{argmax}_c P(C = c \mid \underline{A} = \underline{a}) = \operatorname{argmax}_c P(\underline{A} = \underline{a} \mid C = c), \text{ thus}$$

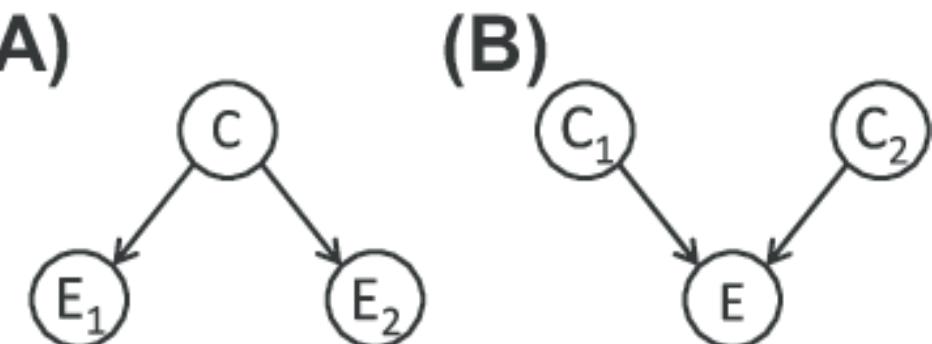
maximum *a posteriori* (the first) and maximum likelihood (the second) estimations are equal

Estimating probabilities from data

- Based on Bayes theorem: $c^* = \operatorname{argmax}_c P(\underline{A} = \underline{a} \mid C = c) \cdot P(C = c)$
- How do we estimate the probabilities?
- $P(C = c_j) = \frac{n_j}{n}$, i.e. the relative frequency of records with label c_j to the number of total records
- To estimate $P(A_1, A_2, \dots, A_d \mid C)$ we have to estimate a lot of parameters
 - For binary attributes: $\#c \cdot (2^d - 1)$ parameters
 - It is only possible if we have very big data
- Naive Bayes (Naïve Bayes) approach:
we assume that A_1, A_2, \dots, A_d random variables are conditionally independent of each other given the class variable C
$$P(A_1, A_2, \dots, A_d \mid C) = P(A_1 \mid C) \cdot P(A_2 \mid C) \cdot \dots \cdot P(A_d \mid C)$$
 - We have to estimate $P(A_i = a_i \mid C = c_j)$ values for any possible i, j pairs
 - Using Naive Bayes assumption the complexity of the problem is reduced
 - For binary attributes there are $\#c \cdot d$ parameters to fit

Naive Bayes assumption

- Naive Bayes assumption: the features are conditionally independent of each other given the class variable
- Conditionally independent but not independent variables:
 - The reading ability of a child is conditionally independent of his/her height given his/her age (common cause – Fig. A)
- Independent but not conditionally independent variables:
 - Basketball game is independent of rain but they are not conditionally independent given the traffic (common effect – Fig. B)



Estimating probabilities from data II.

- Calculating (estimating) $P(A_i = a_i | C = c_j)$ for categorical attributes $P(A_i = a_i | C = c_j) = \frac{n_{ij}}{n_j}$, i.e. number of records with label c_j and with i th attribute a_i divided by the number of records with label c_j
- For continuous attributes
 - Transforming it to discrete
 - Distribution based estimation
 - Approximation by normal distribution

Approximation by normal distribution

- We assume that

$$P(A_i = a_i \mid C = c_j) = \frac{1}{\sqrt{2\pi}\sigma_{ij}} e^{-\frac{(a_i - \mu_{ij})^2}{2\sigma_{ij}^2}}$$

- We have to estimate the value of σ_{ij} and μ_{ij} from the data
 - Sample mean and sample standard deviation
- μ_{ij} is the mean of attribute A_i considering the records with class c_j
- σ_{ij} is the standard deviation of attribute A_i among records with class c_j
- Be careful: the formula above is quite sloppy from a mathematical point of view, regarding continuous variables the question itself is meaningless (instead of a concrete a_i value it would be meaningful to ask for an interval)
 - However, in practice it is used that way!

Example for probability estimation

Tid	Refund	Marital Status	Taxable Income	Evade
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

- Estimating the probability of labels

$$P(\text{No}) = 7/10$$

$$P(\text{Yes}) = 3/10$$

- Estimating the probability of attribute values (discrete case)

$$P(\text{Status} = \text{Married} | \text{No}) = 4/7$$

$$P(\text{Refund} = \text{Yes} | \text{Yes}) = 0$$

Example for probability estimation II.

Tid	Refund	Marital Status	Taxable Income	Evade
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

- Estimating the probability of attribute values (continuous case, normal approximation)

- If Class = No than for Income
 - Mean = 110
 - Sample variance = 2975

$$P(\text{Income} = 120 | \text{No}) = \frac{1}{\sqrt{2\pi} \cdot 54.54} e^{-\frac{(120-110)^2}{2 \cdot 2975}} = 0.0072$$

Prediction

- After we estimated the conditional probabilities based on the training set then we can classify a new observation based on its (a_1, a_2, \dots, a_d) feature vector
- For every c_j label we calculate the following probability:

$$P(\underline{A} = (a_1, a_2, \dots, a_d) \mid C = c_j) \cdot P(C = c_j) =$$

$$P(A_1 = a_1 \mid C = c_j) \cdot P(A_2 = a_2 \mid C = c_j) \cdot \dots \cdot P(A_d = a_d \mid C = c_j) \cdot P(C = c_j)$$

- We predict the c_j class label for which the probability above is maximal, that was our aim to find the maximum *a posteriori* estimation:

$$c^* = \operatorname{argmax}_c P(\underline{A} = \underline{a} \mid C = c) \cdot P(C = c)$$

Example for classification

Name	Give Birth	Can Fly	Live in Water	Have Legs	Class
human	yes	no	no	yes	mammals
python	no	no	no	no	non-mammals
salmon	no	no	yes	no	non-mammals
whale	yes	no	yes	no	mammals
frog	no	no	sometimes	yes	non-mammals
komodo	no	no	no	yes	non-mammals
bat	yes	yes	no	yes	mammals
pigeon	no	yes	no	yes	non-mammals
cat	yes	no	no	yes	mammals
leopard shark	yes	no	yes	no	non-mammals
turtle	no	no	sometimes	yes	non-mammals
penguin	no	no	sometimes	yes	non-mammals
porcupine	yes	no	no	yes	mammals
eel	no	no	yes	no	non-mammals
salamander	no	no	sometimes	yes	non-mammals
gila monster	no	no	no	yes	non-mammals
platypus	no	no	no	yes	mammals
owl	no	yes	no	yes	non-mammals
dolphin	yes	no	yes	no	mammals
eagle	no	yes	no	yes	non-mammals

A: Attributes

M: Mammals

N: Non-Mammals

$$P(A|M) = \frac{6}{7} \cdot \frac{6}{7} \cdot \frac{2}{7} \cdot \frac{2}{7} = 0.06$$

$$P(A|N) = \frac{1}{13} \cdot \frac{10}{13} \cdot \frac{3}{13} \cdot \frac{4}{13} = 0.0042$$

$$P(A|M)P(M) = 0.06 \cdot \frac{7}{20} = 0.021$$

$$P(A|N)P(N) = 0.004 \cdot \frac{13}{20} = 0.0027$$

$$P(A|M)P(M) > P(A|N)P(N) \Rightarrow \text{Mammals}$$

Give Birth	Can Fly	Live in Water	Have Legs	Class
yes	no	yes	no	?

Example for classification II.

$$X = (\text{Refund} = \text{No}, \text{Married}, \text{Income} = 120\text{K})$$

Tid	Refund	Marital Status	Taxable Income	Evade
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

naive Bayes Classifier:

$P(\text{Refund}=\text{Yes}|\text{No}) = 3/7$
 $P(\text{Refund}=\text{No}|\text{No}) = 4/7$
 $P(\text{Refund}=\text{Yes}|\text{Yes}) = 0$
 $P(\text{Refund}=\text{No}|\text{Yes}) = 1$
 $P(\text{Marital Status}=\text{Single}|\text{No}) = 2/7$
 $P(\text{Marital Status}=\text{Divorced}|\text{No}) = 1/7$
 $P(\text{Marital Status}=\text{Married}|\text{No}) = 4/7$
 $P(\text{Marital Status}=\text{Single}|\text{Yes}) = 2/7$
 $P(\text{Marital Status}=\text{Divorced}|\text{Yes}) = 1/7$
 $P(\text{Marital Status}=\text{Married}|\text{Yes}) = 0$

For taxable income:
If class=No: sample mean=110
sample variance=2975
If class=Yes: sample mean=90
sample variance=25

- $P(X|\text{Class}=\text{No}) = P(\text{Refund}=\text{No}|\text{Class}=\text{No}) \times P(\text{Married}|\text{Class}=\text{No}) \times P(\text{Income}=120\text{K}|\text{Class}=\text{No}) = 4/7 \times 4/7 \times 0.0072 = 0.0024$

- $P(X|\text{Class}=\text{Yes}) = P(\text{Refund}=\text{No}|\text{Class}=\text{Yes}) \times P(\text{Married}|\text{Class}=\text{Yes}) \times P(\text{Income}=120\text{K}|\text{Class}=\text{Yes}) = 1 \times 0 \times 1.2 \times 10^{-9} = 0$

Since $P(X|\text{No})P(\text{No}) > P(X|\text{Yes})P(\text{Yes})$

Therefore $P(\text{No}|X) > P(\text{Yes}|X)$
 $\Rightarrow \text{Class} = \text{No}$

What happens if a conditional probability is 0?

- It may occur that for an i, j pair the estimated probability is $P(A_i = a_i | C = c_j) = 0$, since there are no such records in the training set
 - Based on the other attributes a certain class may look a good choice but if a conditional probability is estimated to be 0 that makes the whole expression to be 0
 - Solution: using different estimation methods with smoothing!
- Original: $P(A_i = a_i | C = c_j) = \frac{n_{ij}}{n_j}$
- Laplace smoothing: $P(A_i = a_i | C = c_j) = \frac{n_{ij} + 1}{n_j + \#c}$
- m-smoothing: $P(A_i = a_i | C = c_j) = \frac{n_{ij} + mp}{n_j + m}$
 - Where: n_{ij} : number of records with label c_j and with i th attribute a_i ; n_j : the number of records with label c_j ; $\#c$: number of possible labels, p : prior expectation for probability, m : a parameter

Example: classifying mammals without Laplace smoothing

Name	Give Birth	Can Fly	Live in Water	Have Legs	Class
human	yes	no	no	yes	mammals
python	no	no	no	no	non-mammals
salmon	no	no	yes	no	non-mammals
frog	no	no	sometimes	yes	non-mammals
komodo	no	no	no	yes	non-mammals
bat	yes	yes	no	yes	mammals
pigeon	no	yes	no	yes	non-mammals
cat	yes	no	no	yes	mammals
leopard shark	yes	no	yes	no	non-mammals
turtle	no	no	sometimes	yes	non-mammals
penguin	no	no	sometimes	yes	non-mammals
porcupine	yes	no	no	yes	mammals
eel	no	no	yes	no	non-mammals
salamander	no	no	sometimes	yes	non-mammals
gila monster	no	no	no	yes	non-mammals
platypus	no	no	no	yes	mammals
owl	no	yes	no	yes	non-mammals
eagle	no	yes	no	yes	non-mammals

Give Birth	Can Fly	Live in Water	Have Legs	Class
yes	no	yes	no	?

$$P(A|M) = \frac{4}{5} \cdot \frac{4}{5} \cdot \frac{0}{5} \cdot \frac{0}{5} = 0$$

$$P(A|N) = \frac{1}{13} \cdot \frac{10}{13} \cdot \frac{3}{13} \cdot \frac{4}{13} = 0.0042$$

$$P(A|M)P(M) = 0$$

$$P(A|N)P(N) = 0.004 \cdot \frac{13}{18} = 0.0029$$

$$P(A|M)P(M) < P(A|N)P(N)$$

⇒ Non-Mammals

??

..

Example: classifying mammals with Laplace smoothing

Name	Give Birth	Can Fly	Live in Water	Have Legs	Class
human	yes	no	no	yes	mammals
python	no	no	no	no	non-mammals
salmon	no	no	yes	no	non-mammals
frog	no	no	sometimes	yes	non-mammals
komodo	no	no	no	yes	non-mammals
bat	yes	yes	no	yes	mammals
pigeon	no	yes	no	yes	non-mammals
cat	yes	no	no	yes	mammals
leopard shark	yes	no	yes	no	non-mammals
turtle	no	no	sometimes	yes	non-mammals
penguin	no	no	sometimes	yes	non-mammals
porcupine	yes	no	no	yes	mammals
eel	no	no	yes	no	non-mammals
salamander	no	no	sometimes	yes	non-mammals
gila monster	no	no	no	yes	non-mammals
platypus	no	no	no	yes	mammals
owl	no	yes	no	yes	non-mammals
eagle	no	yes	no	yes	non-mammals

Give Birth	Can Fly	Live in Water	Have Legs	Class	
yes	no	yes	no	?	

With Laplace smoothing:

$$P(A|M) = \frac{5}{7} \cdot \frac{5}{7} \cdot \frac{1}{7} \cdot \frac{1}{7} = 0.01$$

$$P(A|N) = \frac{2}{15} \cdot \frac{11}{15} \cdot \frac{4}{15} \cdot \frac{5}{15} = 0.008$$

$$P(A|M)P(M) = 0.01 \cdot \frac{5}{18} = 0.0027$$

$$P(A|N)P(N) = 0.008 \cdot \frac{13}{18} = 0.0057$$

$P(A|M)P(M) < P(A|N)P(N)$
 \Rightarrow Non-Mammals

Evaluation of Naive Bayes

- Robust (insensitive) for irrelevant attributes
- It can handle missing data (it ignores the missing values for the probability estimation)
- The naive Bayes assumption (conditional independence) in many cases is not valid in reality
 - In practice it still works quite well
 - This assumption can be refined (Bayesian Networks, Bayesian Belief Networks)

Problem

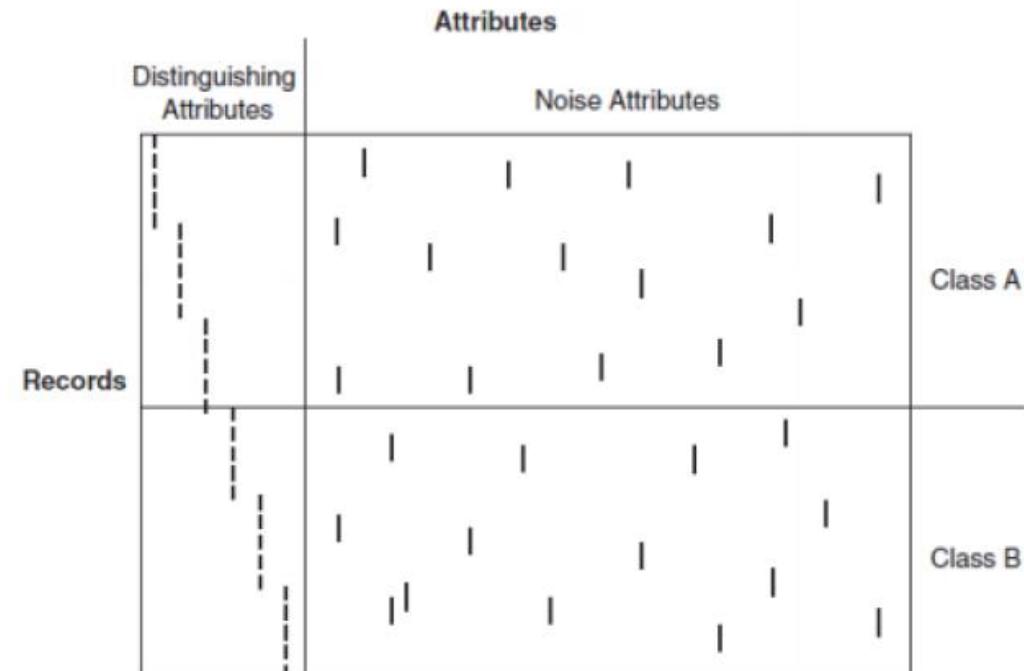
Classify the following record $X = (\text{Marital status} = \text{Single}, \text{Annual income} = 90\text{K})$ using the Naive Bayes classifier based on the training data in the following table, where Default is the class label. Discretize annual income by 20K intervals: [60K, 80K), [80K, 100K), ...!

1. Use the original estimates!
2. Use Laplace smoothing!

Marital status	Annual income	Default
Single	125K	No
Married	95K	No
Single	70K	No
Married	120K	No
Divorced	75K	Yes
Married	60K	No
Divorced	220K	No
Single	85K	Yes
Married	75K	No
Single	90K	Yes

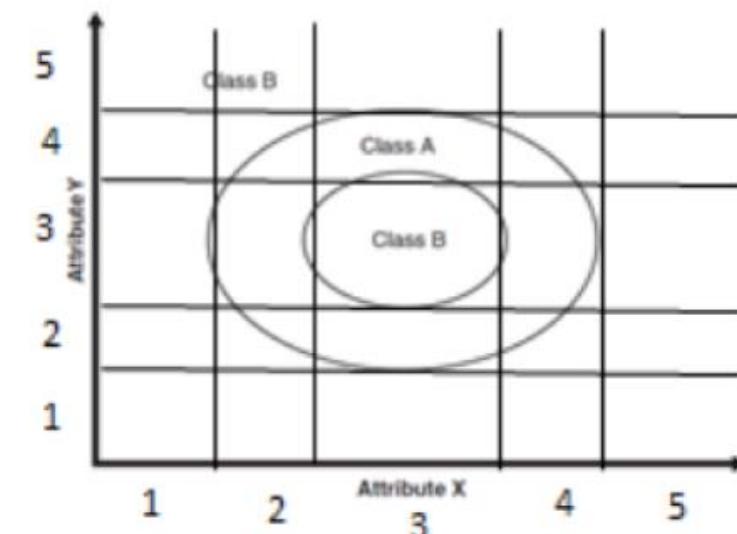
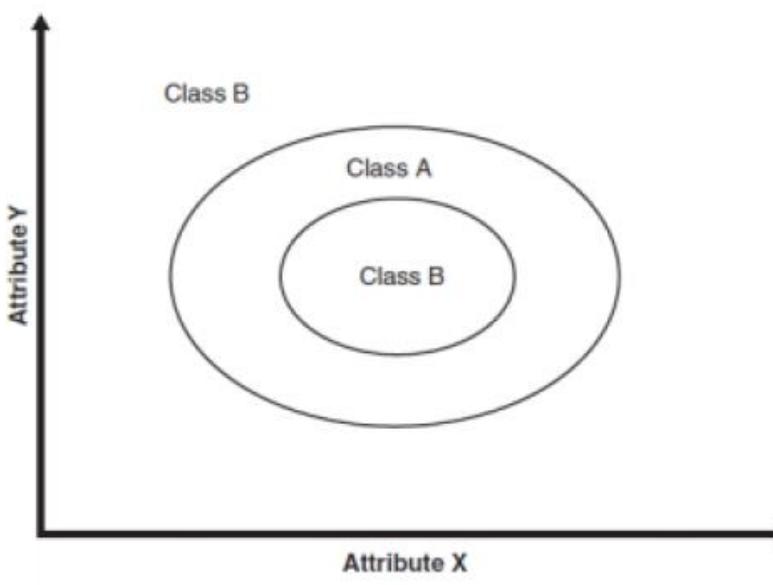
Assume that the following data set contains 1000 records with class label A and 1000 records with label B. There are some binary variables with distinctive power: X_1, X_2, \dots , in addition, there are many noisy binary attributes, that take value 1 or 0 at random.

1. Sketch a decision tree that learns such data! How would the decision tree classifier perform on this data?
2. Which records are close to the first record? How would the k NN classifier perform on this data?
3. Outline the conditional probabilities! How would the Naive Bayes classifier perform on this data? Consider the first row as an example!



Consider the following data with two attributes (X and Y) and two possible labels (A and B). The position of class A and class B records in the X-Y space is illustrated below.

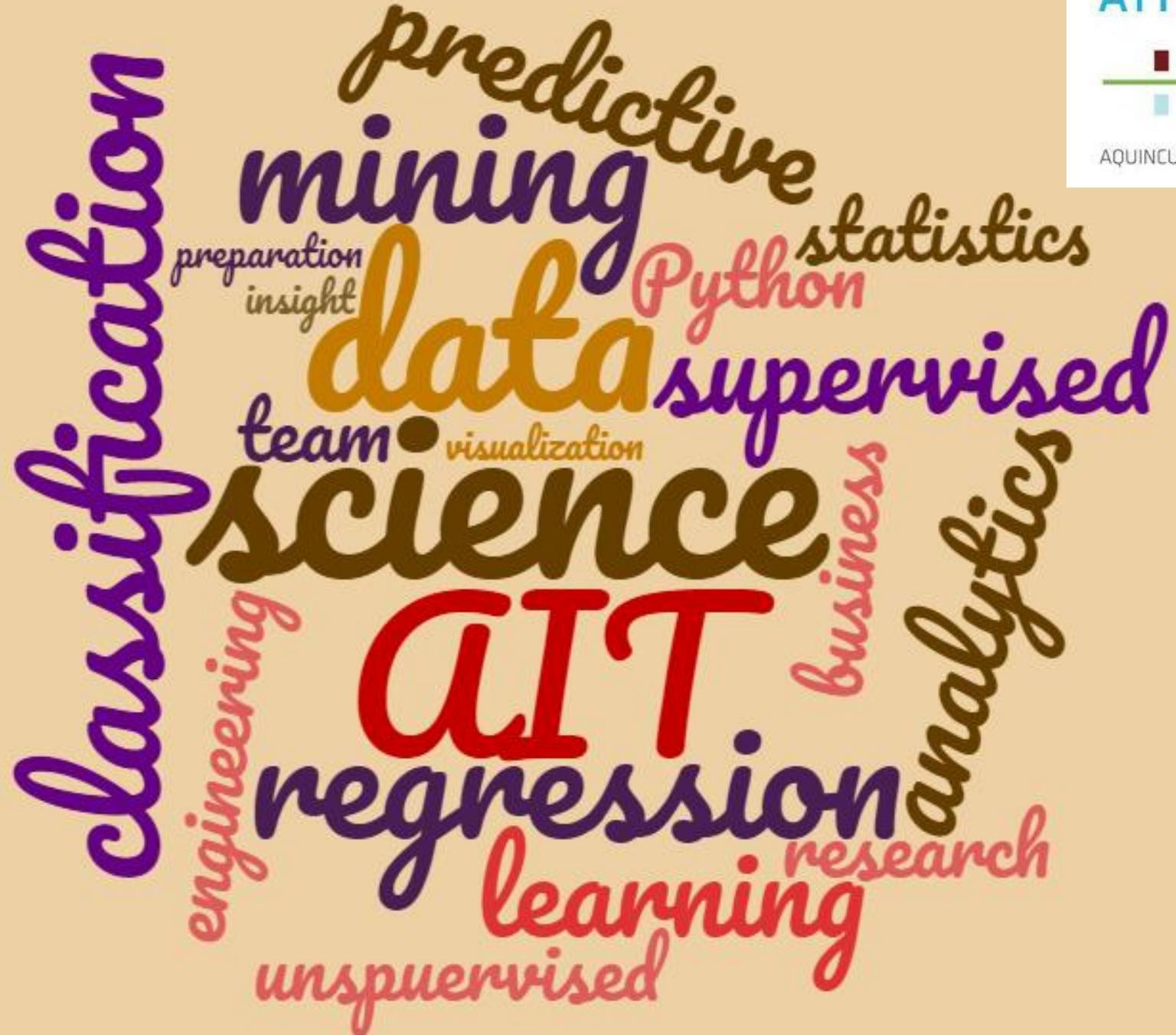
1. How would a decision tree work on such data? Indicate decision boundaries!
2. How would the kNN classifier perform on this data? What does its performance depend on?
3. How would the Naive Bayes classifier perform? Outline the conditional probabilities! We assume that the two classes have the same number of records and that the instances are distributed uniformly. Use the possible discretization given below, i.e. both attribute X and Y are discretized into 5 bins!



Acknowledgement

- András Benczúr, Róbert Pálovics, SZTAKI-AIT, DM1-2
- Krisztián Buza, MTA-BME, VISZJV68
- Bálint Daróczy, SZTAKI-BME, VISZAMA01
- Judit Csima, BME, VISZM185
- Gábor Horváth, Péter Antal, BME, VIMMD294, VIMIA313
- Lukács András, ELTE, MM1C1AB6E
- Tim Kraska, Brown University, CS195
- Dan Potter, Carsten Binnig, Eli Upfal, Brown University, CS1951A
- Erik Sudderth, Brown University, CS142
- Joe Blitzstein, Hanspeter Pfister, Verena Kaynig-Fittkau, Harvard University, CS109
- Rajan Patel, Stanford University, STAT202
- Andrew Ng, John Duchi, Stanford University, CS229



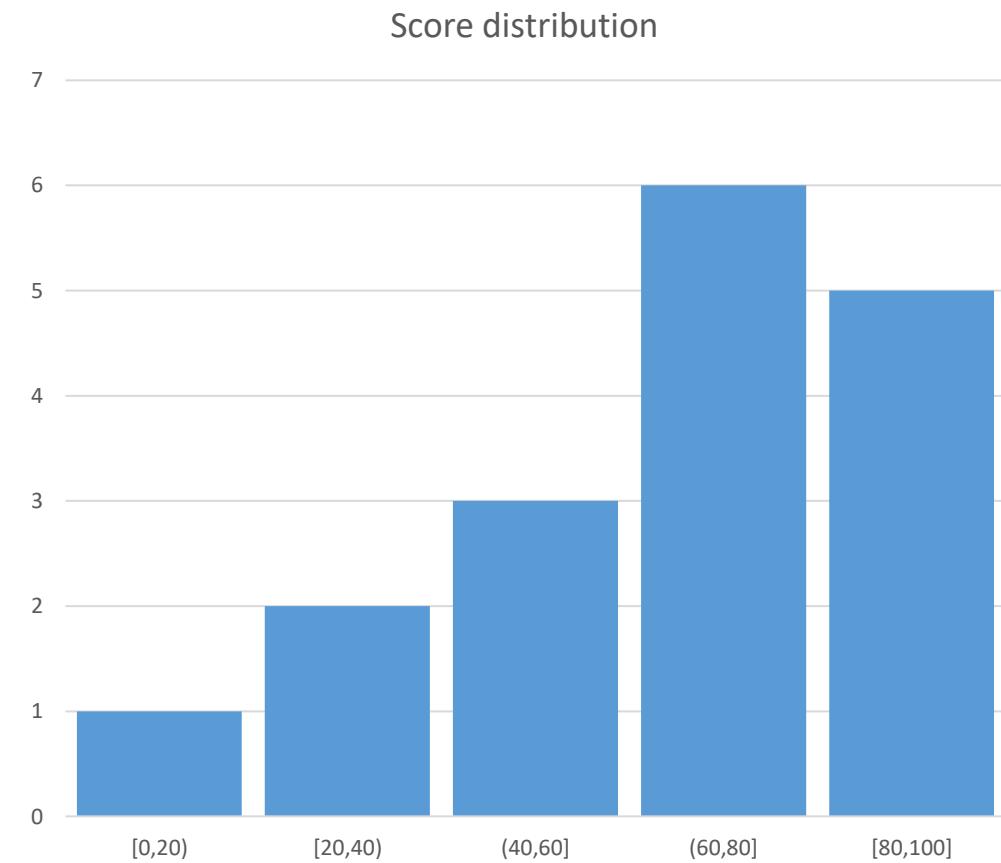


Schedule of the semester

	<i>Monday midnight</i>	<i>Tuesday class</i>	<i>Friday class</i>
W1 (02/06)			
W2 (02/13)		HW1 out	
W3 (02/20)			
W4 (02/27)	HW1 deadline + TEAMS	HW2 out	
W5 (03/06)			PROJECT PLAN
W6 (03/13)	HW2 deadline	HW3 out	
W7 (03/20)			MIDTERM
SPRING BREAK		SPRING BREAK	SPRING BREAK
W8 (04/03)	HW3 deadline		GOOD FRIDAY
W9 (04/10)	MILESTONE 1	HW4 out	
W10 (04/17)			
W11 (04/24)	HW4 deadline		
W12 (05/01)	MILESTONE 2		
W13 (05/08)			
W14 (05/15)		FINAL	PROJECT presentations
W15 (05/22)		PROJECT presentations	

Midterm results

- MIDTERM (25%) + FINAL (25%) + HOMEWORK (25%) + PROJECT (25%)
- Cutoffs for letter grades:
 - 90% - A+
 - 85% - A
 - 80% - A- $(30+40+80+86)/4 \rightarrow 59 \rightarrow C$
 - 75% - B+ $(40+50+85+85)/4 = 65 \rightarrow B-$
 - 70% - B $(70+75+85+90)/4 = 80 \rightarrow A-$
 - 65% - B- $(80+85+95+100)/4 = 90 \rightarrow A+$
 - 60% - C+
 - 55% - C
 - And so on...



Milestone 1

- Two-page long report covering the followings:
 - Have you managed to gather the data? Do you have enough data of appropriate quality?
 - Did you collect the relevant related works? What useful information could you discover?
 - Initial data analysis steps
 - What next steps do you plan to take?



Regression

- We try to predict continuous valued output based on the values of other variables (via supervised learning)
- Examples:

Explanatory (input) variables	Target (output)
Number of rooms, size, location (ZIP code), ...	Market value of a house
Movie budget, film genre, popularity of the actors (based on their IMDB pages)	Box office result of a movie
Major, admission point score, gender, age, GMAT scores,	GPA

- Challenges: finding the right explanatory variables, the most suitable functional form/modelling approach

Fundamental task of regression

Let $X = (X_1, X_2, \dots, X_p)$ be the feature vector and Y is the target variable.

Regression: we suppose that there is a relationship between X and Y , in general: $Y = f(X) + \epsilon$, where ϵ (the random error) is independent from X and has zero mean

Aim: giving prediction: $\hat{Y} = \hat{f}(X)$

In reality \hat{f} is sometimes considered to be a black-box, we are not interested in the functional form, but to give accurate enough prediction to Y

Learning: On the labeled data of the training set we estimate the function f , minimizing the „prediction error” on the training set

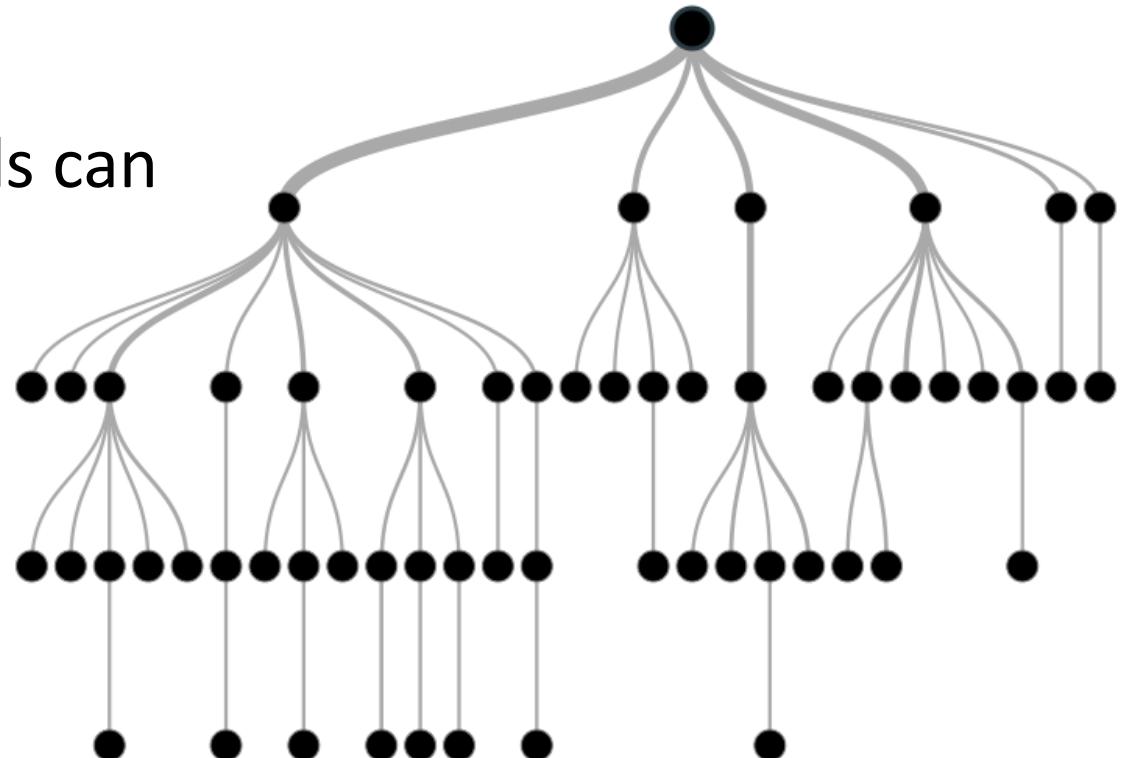
Prediction: using \hat{f} for data that we have not seen before $\hat{Y} = \hat{f}(X)$

Parametric and non-parametric models

- Parametric model
 - We find $f(X)$ in a predefined function family (functional form)
 - E.g. linear, polynomial form
 - The models can be described using a finite number of parameters
 - We optimize for these parameters
- Non-parametric model
 - No functional form is supposed
 - No assumption is needed for the functional form, there is no restriction, the model can better fit the data, the model is more flexible
 - „Infinite dimensional parameter space”
 - To achieve an accurate estimation, more data points are needed than in the parametric case
 - Usually more complex and slower than the parametric case
 - E.g: kNN, decision tree, SVM

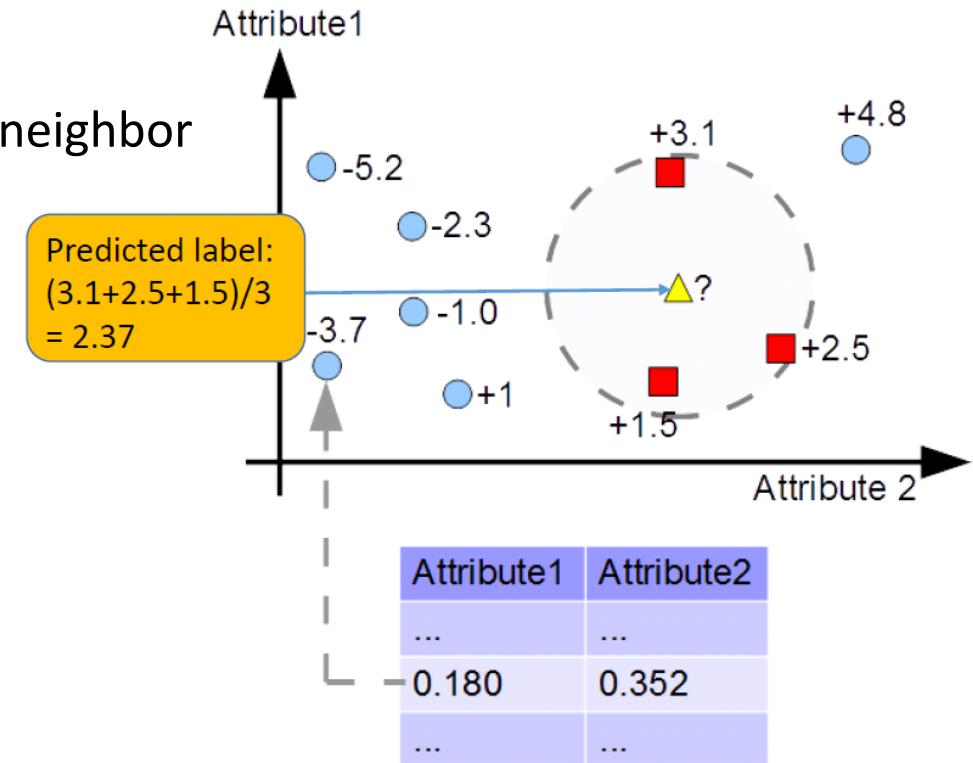
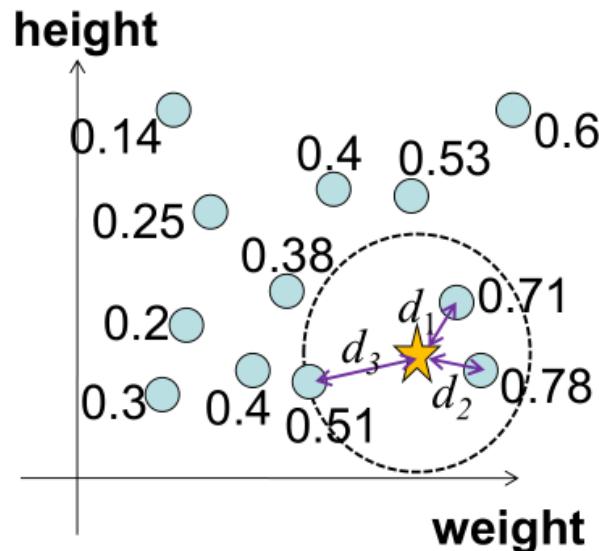
Previously covered methods for regression

- Until now we have focused on classification problems
- We emphasized that some models can also be used to solve regression problems
 - kNN
 - Decision tree



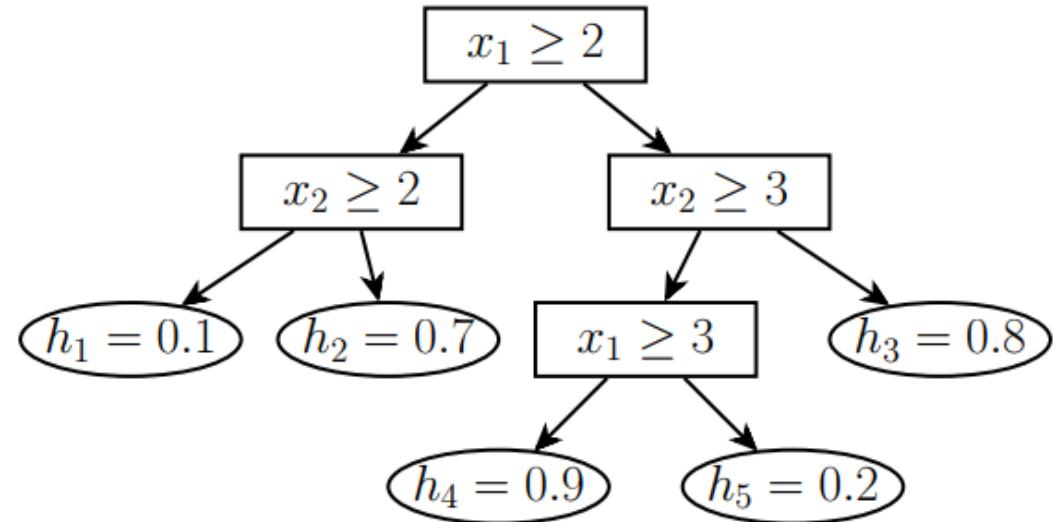
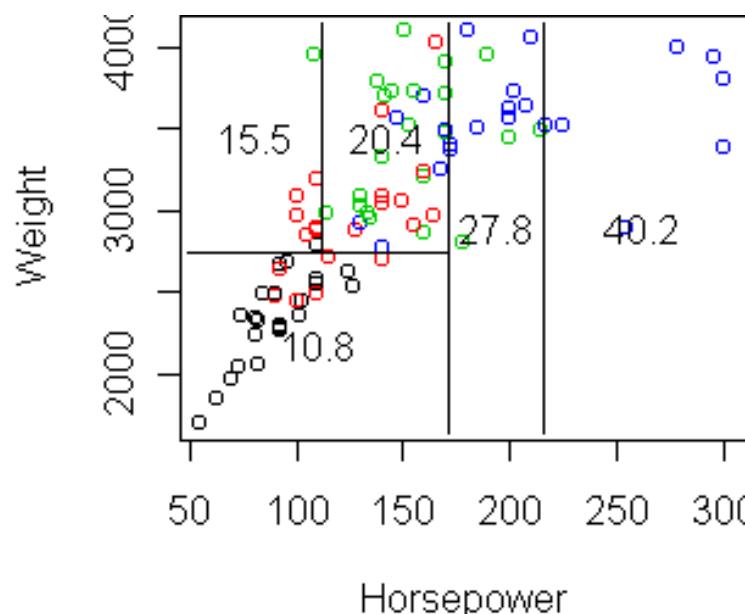
kNN algorithm for regression

- For regression problems:
 - Averaging the target variable of the k nearest neighbors
 - Weighted averaging
 - $w_i = \frac{1}{d_i^2}$, where d_i is the distance of the i th neighbor



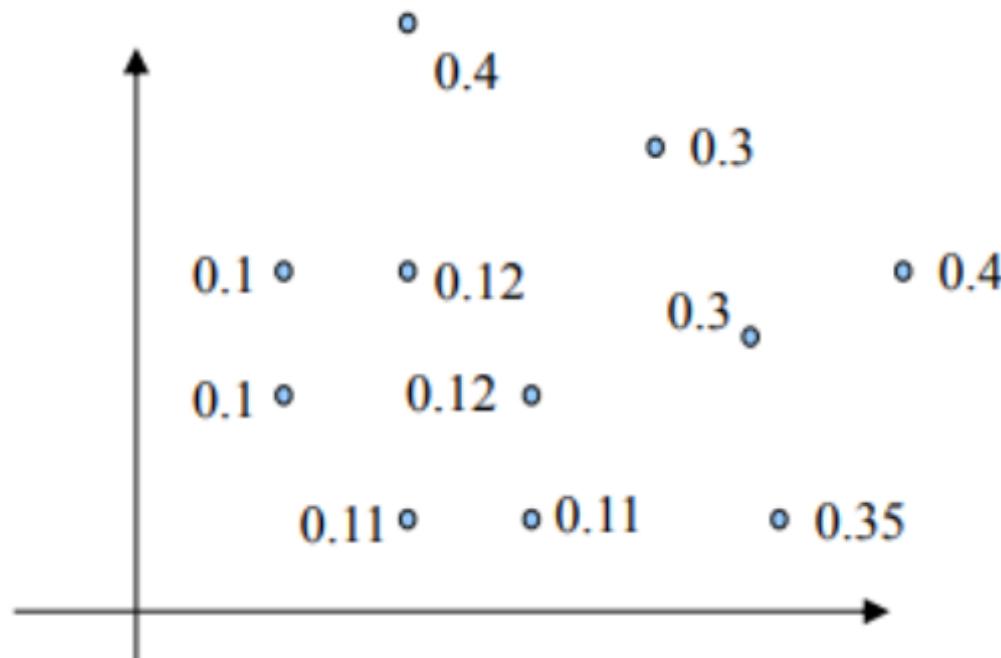
Decision tree for regression

- Prediction: the average of the target variables in each leaf
- Splitting criteria: reducing the variance of the child nodes with respect to the target variable



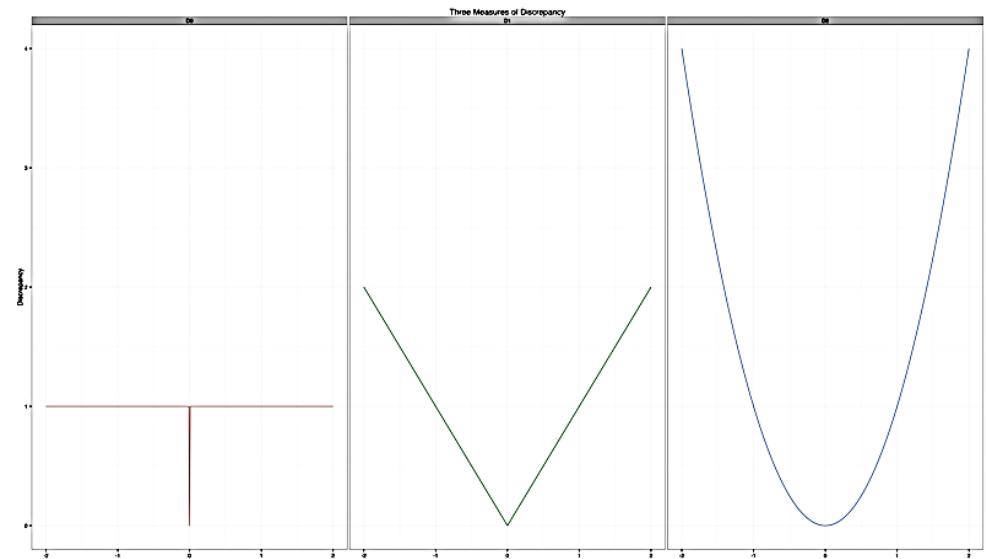
Problem

We would like to solve a regression problem using a decision tree. How would it split the data given in the coordinate system below? The maximum number of leaves is set to 3. Sketch the splits on the figure. (No precise calculation is required.)



Outlook – What is the best guess?

- We have a sample (some observations, real numbers in an ascending order):
 - E.g.: 1.3, 1.8, 2, 2, 2, 2.5, 4.4, 4.6, 5, 5.6, 5.6, 5.9, 6.3, 6.7, 7.1, 8.5, 8.9, 9.3, 9.3, 9.6, 9.8
- What would be the best guess?
- What minimizes the error?
 - It depends on which error?
- Mean, median, mode?
- Mean minimizes the squared error (loss):
 $\bar{x} = \operatorname{argmin}_s \sum_i (x_i - s)^2 \approx 5.44$
- Median minimizes the absolute error (loss):
 $\operatorname{median}(x) = \operatorname{argmin}_s \sum_i |x_i - s| = 5.6$
- Mode minimizes the „equivalence error“:
 $\operatorname{mode}(x) = \operatorname{argmin}_s \sum_i \chi(x_i \neq s) = 2$



Squared error

- For regression problems usually the aim is to minimize the squared error
 - Least square regression, ordinary least square - OLS
- Mean squared error –MSE
Training data: $(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)$

Predicted function: \hat{f}

$$\text{MSE}(\hat{f}) = E\left((\hat{f}(x) - y)^2\right)$$

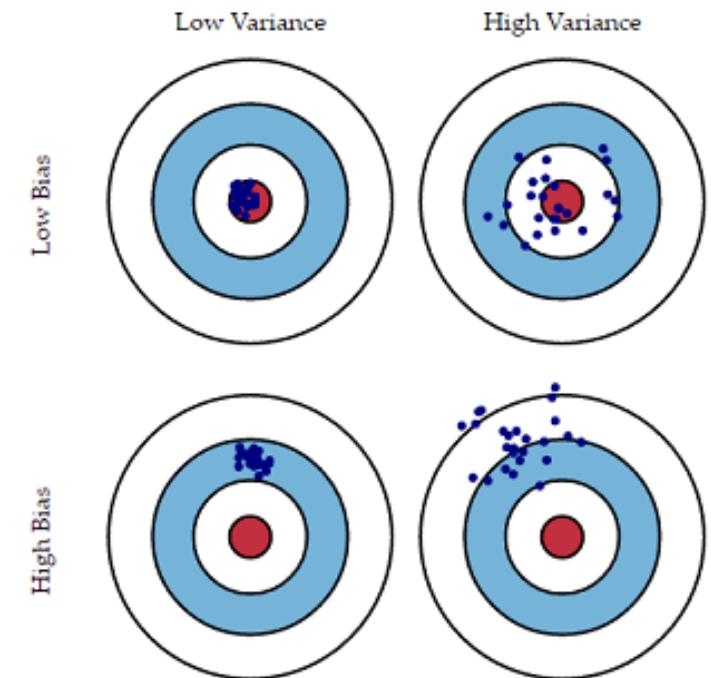
- MSE of the training set

$$\text{MSE}_{train}(\hat{f}) = \frac{1}{n} \sum_{i=1}^n (\hat{f}(x_i) - y_i)^2$$

- MSE of the test set can be calculated similarly

Bias and variance

- **Bias:** The difference between the average prediction of our model and the correct value which we are trying to predict.
- **Variance:** The variability of model prediction for a given data point
 - Usually a model has only one prediction for a given data point but think of it as we can repeat our process of model building to get separate hits on the target



Bias and variance II.

- Bias:

$$\text{Bias}(\hat{f}(x)) = E(\hat{f}(x) - f(x))$$

- Sometimes the squared bias is used

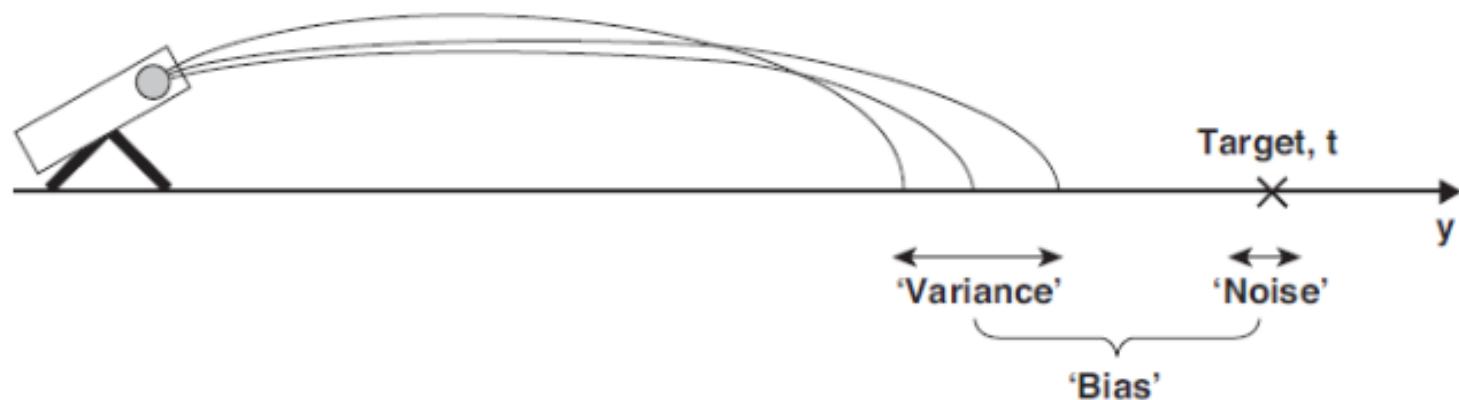
- Variance:

$$\text{Var}(\hat{f}(x)) = E\left(\hat{f}(x) - E\left(\hat{f}(x)\right)\right)^2 = E(\hat{f}(x)^2) - E(\hat{f}(x))^2$$

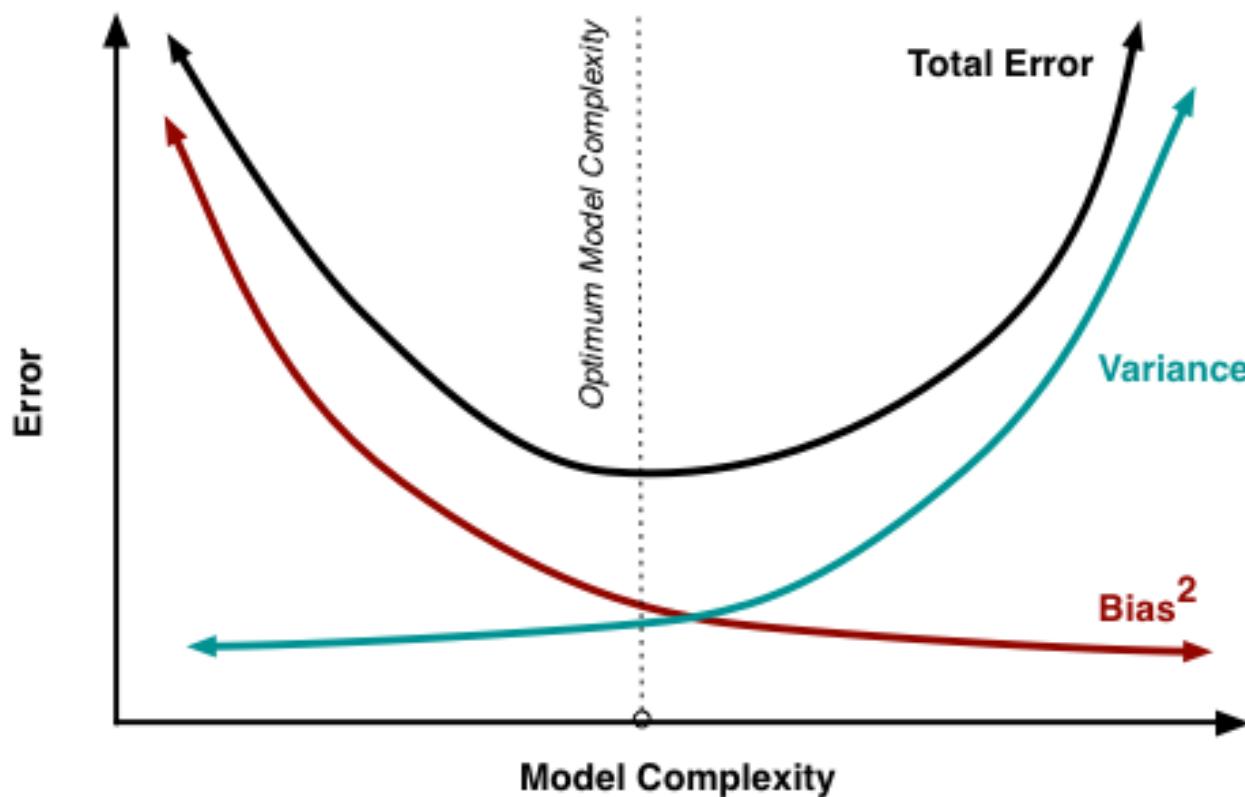
Decomposition of mean squared error

- Claim: For a new record the mean squared error of the predicted value (predicted by \hat{f} function) can be decomposed in the following way:

$$\text{MSE}(\hat{f}(x)) = \text{Var}(\hat{f}(x)) + \text{Bias}^2(\hat{f}(x)) + \text{Var}(\epsilon)$$



Bias-variance tradeoff



Optimal solution: the conditional expected value

- If we aim to minimize the expected (mean) squared error $E \left((f(X_1, X_2, \dots, X_p) - Y)^2 \right)$, then the optimal f function is the conditional expected value:

$$f_{\text{opt}}(x_1, \dots, x_p) = E(Y | X_1 = x_1, X_2 = x_2, \dots, X_p = x_p)$$

- The problem is theoretically solved but to calculate the conditional expected value we should know the joint density function $g(y, x_1, x_2, \dots, x_p)$
 - If the joint distribution is the $(p+1)$ -dimensional normal distribution then f_{opt} is a linear function
 - Due to the multi-dimensional central limit theorem normality is a valid assumption in many cases

- It is reasonable to look for f in the family of linear functions

Linear regression

- We assume that f is linear, i.e. $f(x_1, \dots, x_p) = w_0 + w_1x_1 + w_2x_2 + \dots + w_px_p$
- We are looking for the parameters $w_0, w_1, \dots, w_p \in R$ such that $E(Y - (w_0 + w_1X_1 + w_2X_2 + \dots + w_pX_p))^2$ is minimal
- On a training set with n data points we have to solve the following minimization problem

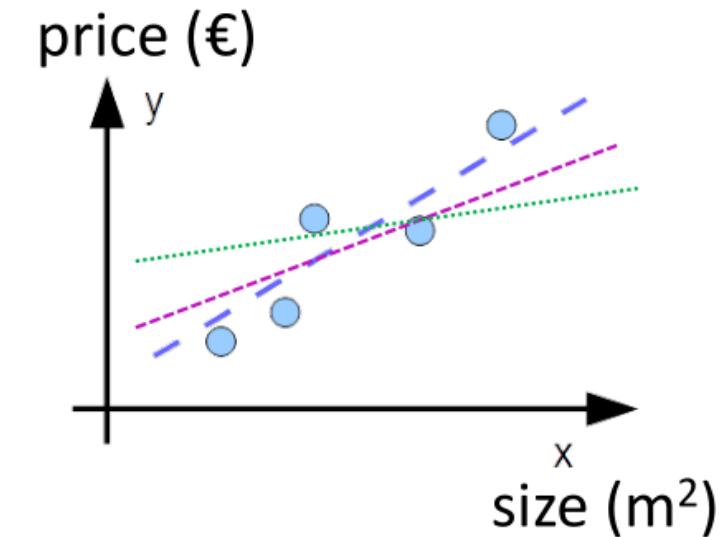
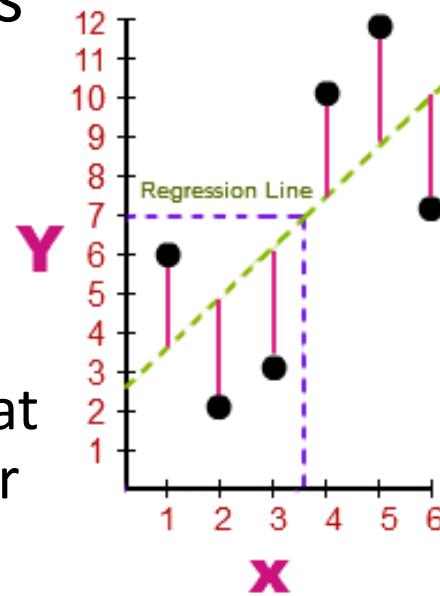
$$\operatorname{argmin}_{w_0, w_1, \dots, w_p} \frac{1}{n} \sum_{i=1}^n (w_0 + \mathbf{w}^T \mathbf{x}_i - y_i)^2$$

where n : number of training data points, $\mathbf{w} = (w_1, w_2, \dots, w_p)$: parameter vector, \mathbf{x}_i : the feature vector of the i th record

- The minimization problem can be solved analytically
 - Huge matrices should be inverted, multiplied with each other (computationally very expensive in practice)
 - Solution: using other optimization methods, e.g. gradient descent

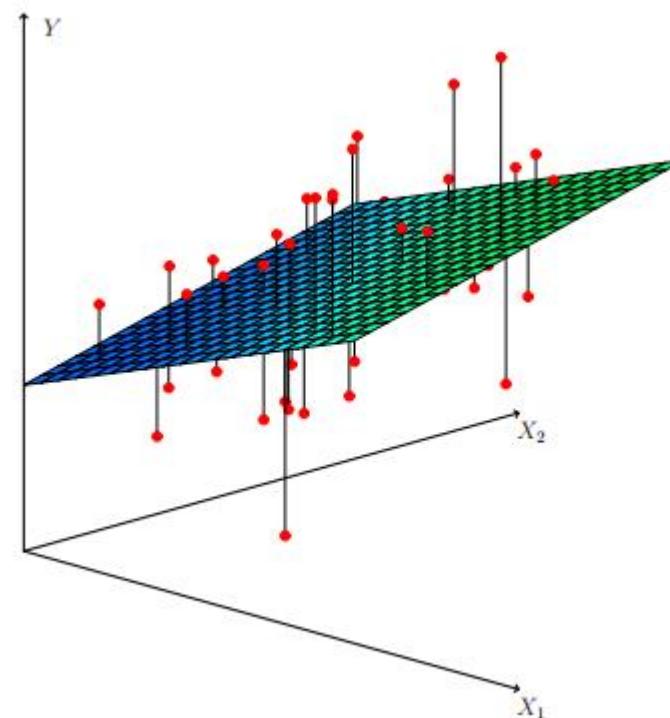
Regression line

- In two dimension linear regression corresponds to a line
- If $n > p+1$, then the system is overdetermined, there is no such a regression that satisfies all the training data
 - We want to find the one that minimizes the squared error



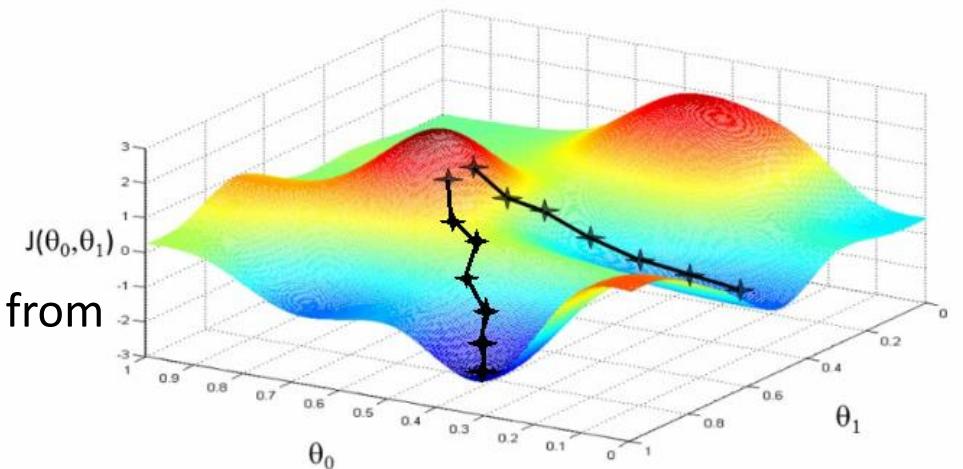
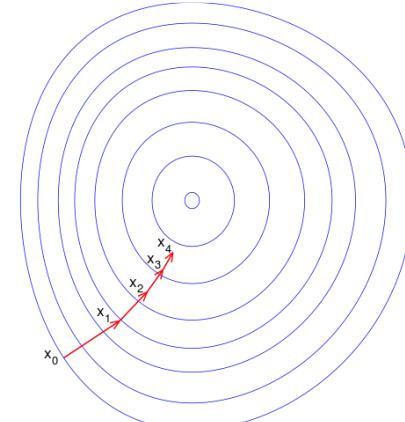
Linear regression for more variables

- Multi-dimensional linear regression defines a hyperplane
 - Red dots are the real observations
 - The plane is the fitted linear regression
 - It minimizes the squared error

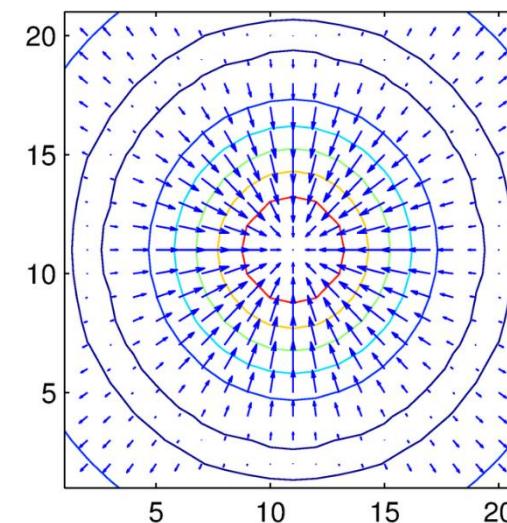
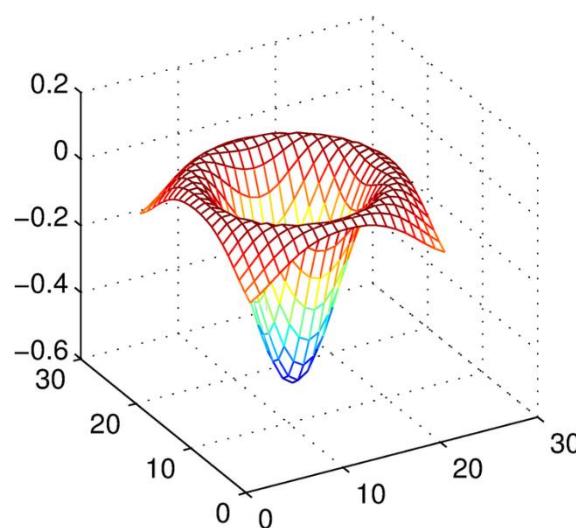
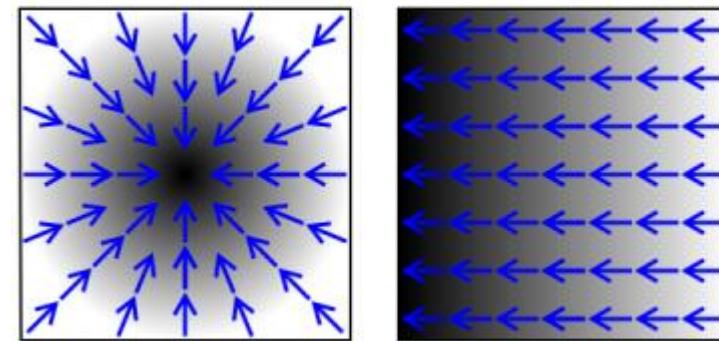
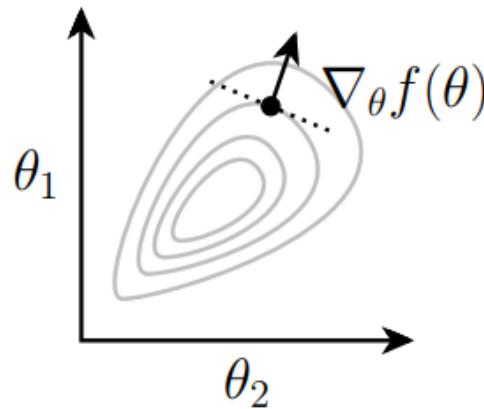


Gradient method (gradient descent/ascent)

- Goal: to find the local minima (maxima) of an f function
- Idea: one takes steps proportional to the negative (positive) of the gradient of the function at the current point
 - Gradient ($\text{grad } f$) is the direction of steepest ascent
 - Negative gradient ($-\text{grad } f$) is the direction of steepest descent
 - The gradient stores all the partial derivatives of a multivariable function
- It will converge to a local minima (maxima), the convergence is quite fast under some regularity assumption
 - It is not necessarily a global minima (maxima)
 - To avoid getting stuck in a local minima we can initialize from more starting points (random restart hill climbing)



Gradient method



Example

Let $y = w_0 + w_1 x_1 + w_2 x_2$ be an equation for regression. We have an observation with $x_1 = 2$ and $x_2 = -1$, its target variable is 3. Write the squared error for this data point as a function of w_0, w_1, w_2 ! Calculate the gradient of the squared error in this point!

$$err = w_0 + w_1 x_1 + w_2 x_2 - 3 = w_0 + w_1 \cdot 2 + w_2 \cdot (-1) - 3$$

$$\frac{d}{dw_1} (err)^2 = \frac{d}{dw_1} (w_0 + w_1 \cdot 2 + w_2 \cdot (-1) - 3)^2 = 2 \cdot 2 \cdot (w_0 + w_1 \cdot 2 + w_2 \cdot (-1) - 3)$$

$$\frac{d}{dw_2} (err)^2 = \frac{d}{dw_2} (w_0 + w_1 \cdot 2 + w_2 \cdot (-1) - 3)^2 = (-1) \cdot 2 \cdot (w_0 + w_1 \cdot 2 + w_2 \cdot (-1) - 3)$$

$$\frac{d}{dw_0} (err)^2 = \frac{d}{dw_0} (w_0 + w_1 \cdot 2 + w_2 \cdot (-1) - 3)^2 = 2 \cdot (w_0 + w_1 \cdot 2 + w_2 \cdot (-1) - 3)$$

Stochastic gradient descent (SGD) – for linear regression

- The function to minimize:

$$Err^2 = ((w_0 + \mathbf{w}^T \mathbf{x}) - y)^2$$

- The gradient is not calculated using the entire dataset but only for one randomly chosen record in each step
- In each step for one randomly chosen training data point

- Calculate the gradient vector (partial derivatives) of the mean-squared error using that single point
- We step to the direction of the negative gradient, i.e. we improve the parameters with the following term:
 $(-1) \cdot \text{gradient} \cdot \text{learning_rate}$

$$\frac{\partial Err^2}{\partial w_0} = 2 \cdot Err$$

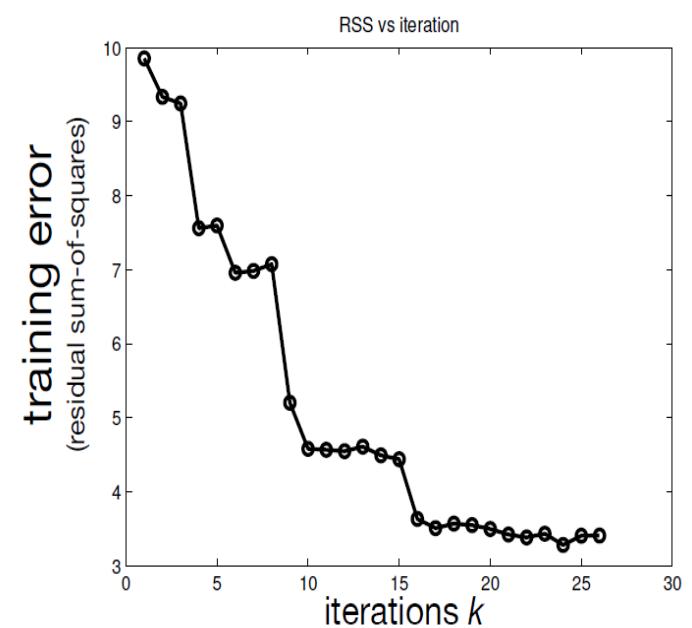
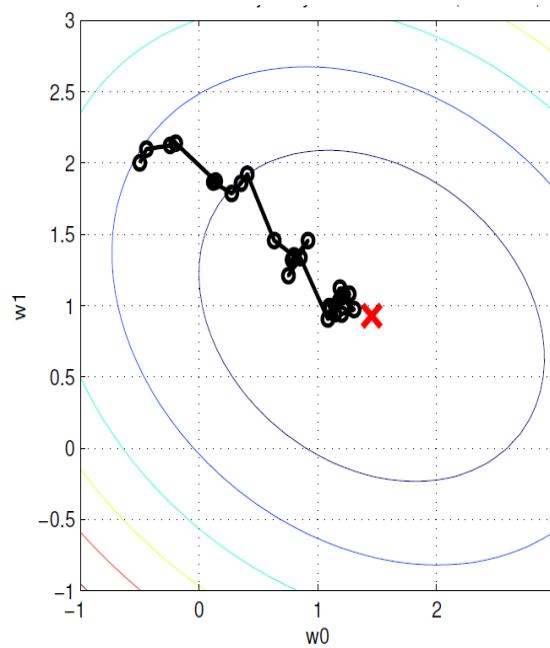
$$\frac{\partial Err^2}{\partial w_i} = 2 \cdot Err \cdot x_i$$

$$w_0 \leftarrow w_0 - \text{lratre} \cdot 2 \cdot Err$$

$$w_i \leftarrow w_i - \text{lratre} \cdot 2 \cdot Err \cdot x_i$$

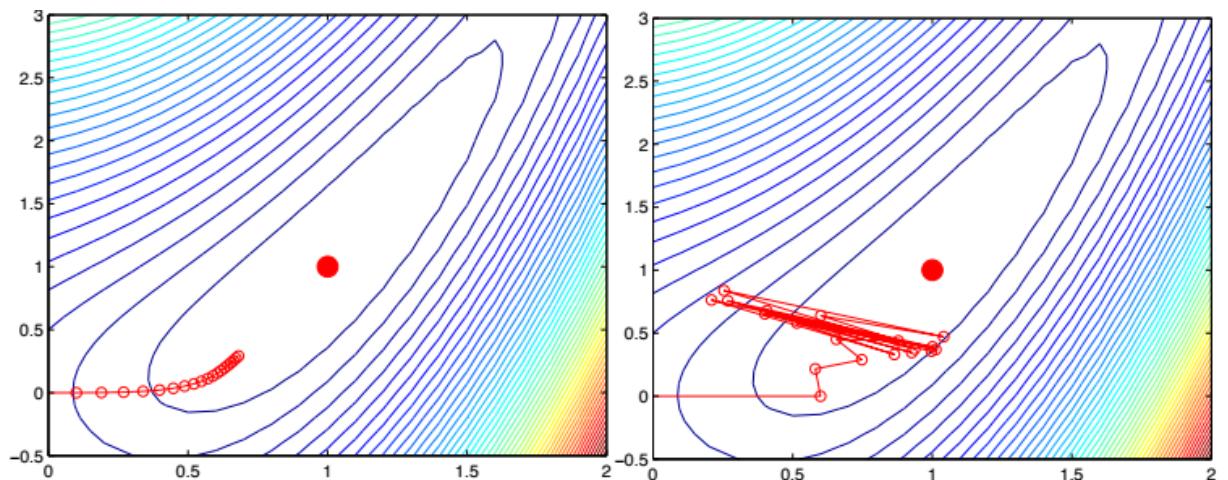
Stochastic gradient descent on training data

- Initialize the parameters (coefficients, weights) of the model randomly
- Iterate the following steps on the training data on randomly chosen data points
 - Calculate the squared error with respect to the chosen data point
 - Calculate the gradient
 - Improve the weights according to the gradient (and the learning rate)



Role of the learning rate

- Learning rate
 - Usual notation: λ
- It regulates the amount by which the weights will change at every step, i.e. how fast the optimization algorithm learns
 - If the rate is too large: the step size is too large, it can (plausibly) „jump over” the minima we are trying to reach
 - If the rate is too small: too many steps are needed to reach the minima (too slow)



Problem

Let $(0, 0, -2); (0, 1, 1); (1, 0, 2)$ be three records on the x_1 - x_2 plane, where the third coordinate is the y target variable. Determine the coefficients of the following linear regression that minimizes the squared error: $y = w_1x_1 + w_2x_2 + w_0$.

- a) Determine the optimal coefficients analytically!
- b) Approximate the optimal coefficients using the gradient descent method (few steps enough).
- c) Approximate the optimal coefficients using the stochastic gradient descent (few steps enough).

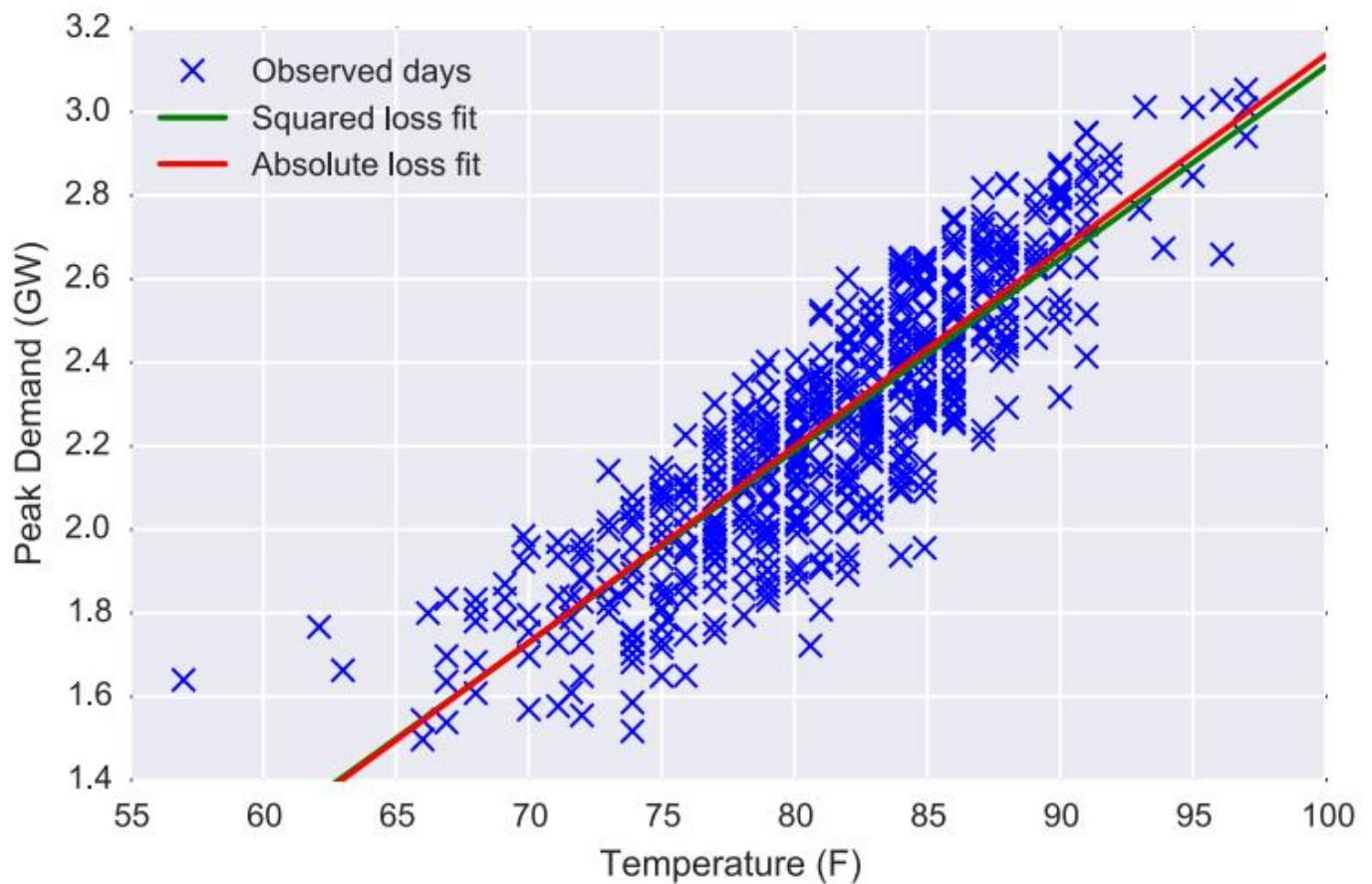
For gradient methods use the following initialization of the weights: $w_1 = w_2 = w_0 = 1$. Let the learning rate be 0.25.

Regularization

- Goal: to avoid overfitting, to have better generalization ability
- How?
 - A regularization term is added to the loss function
 - The regularization term penalizes for too complex models
- For linear regression we add the following regularization term to the squared error: $Err^2 + Reg = (w_0 + \mathbf{w}^T \mathbf{x} - y)^2 + \mu(w_0^2 + |\mathbf{w}|^2)$
 - It penalizes for two large weights (in absolute value) to avoid too large weights and overfitting (μ is the parameter of the regularization term)

Squared error vs. absolute error

- If there are no outliers in the data, then minimizing the squared error and absolute error leads to very similar models



Squared error

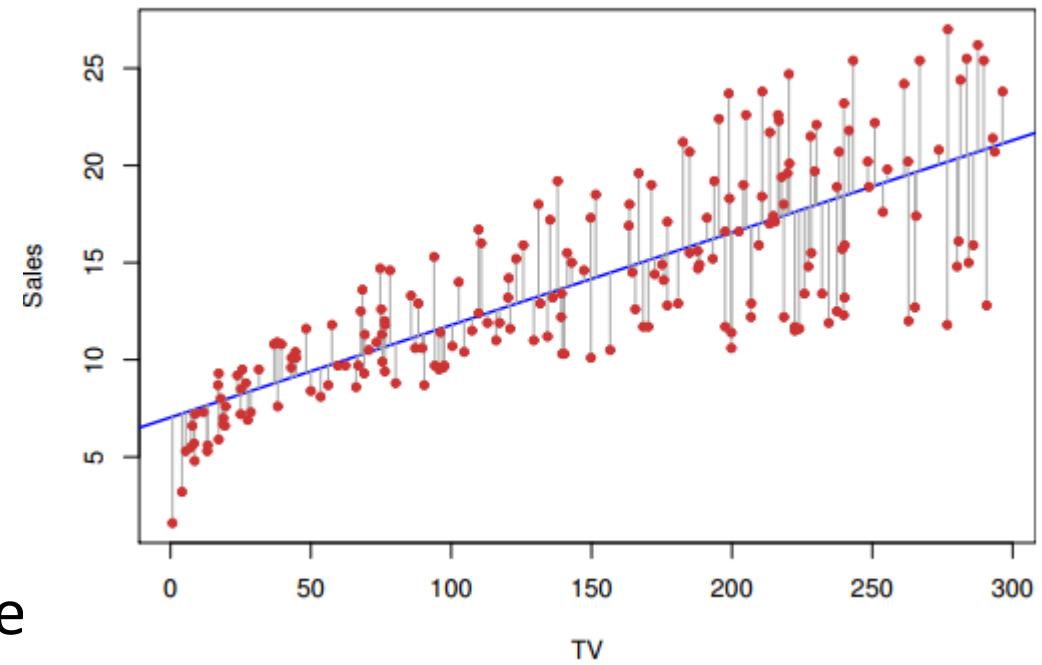
- Residual sum of squares (RSS) - sum of squared errors of prediction (SSE)

$$\text{RSS} = \sum_{i=1}^n (y_i - \hat{y}_i)^2$$

- Root-mean-square error (RMSE)

$$\text{RMSE} = \sqrt{\frac{\sum_{i=1}^n (\hat{y}_i - y_i)^2}{n}}$$

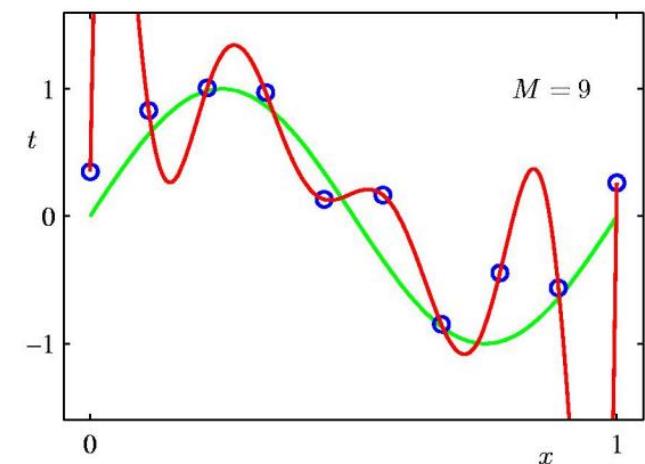
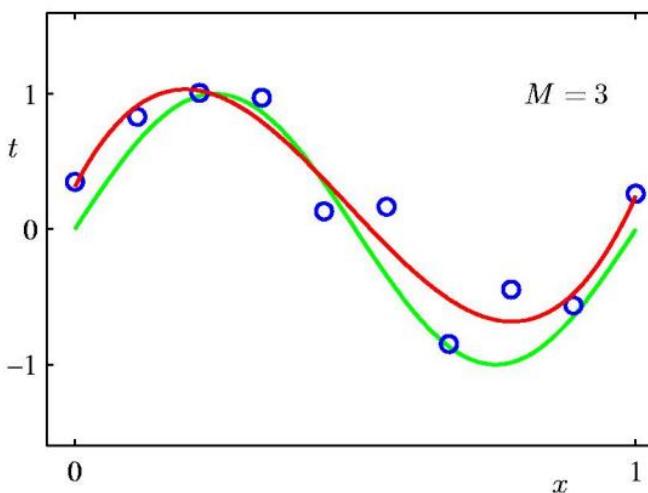
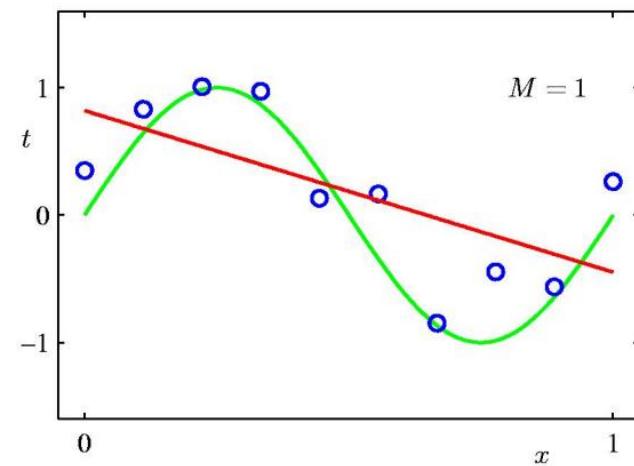
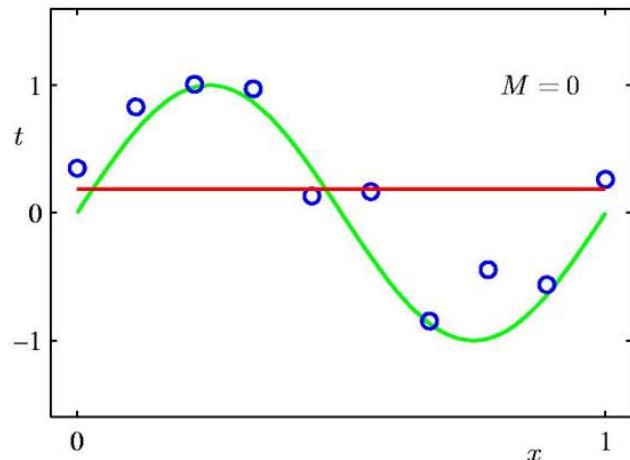
- Minimizing any of them leads to the same model



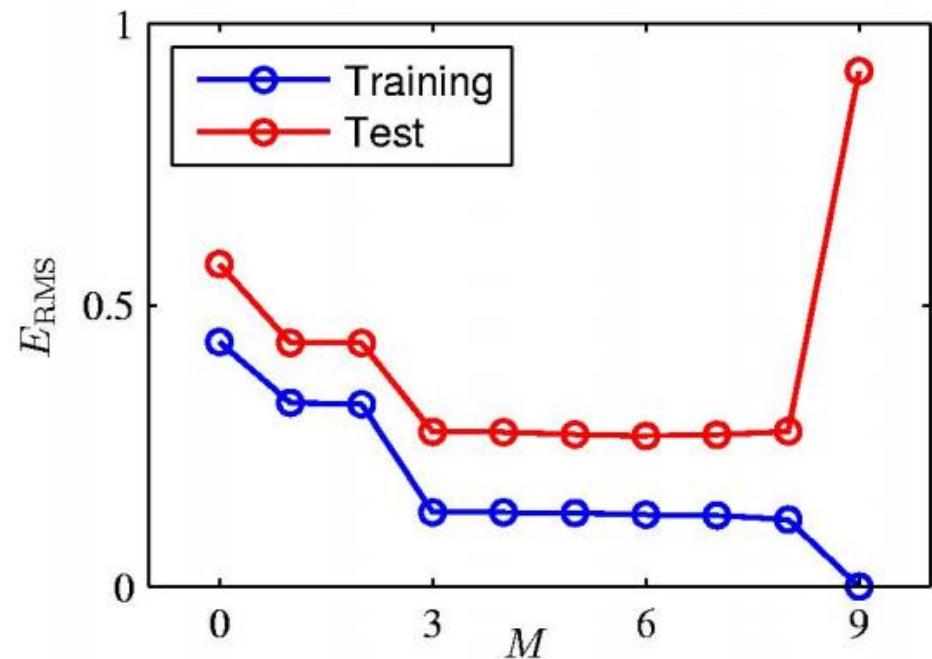
Polynomial regression

- Higher degree polynomials of the attributes are also present in the regression equation
 - Other functions of the attributes may be also considered: log, exp etc.

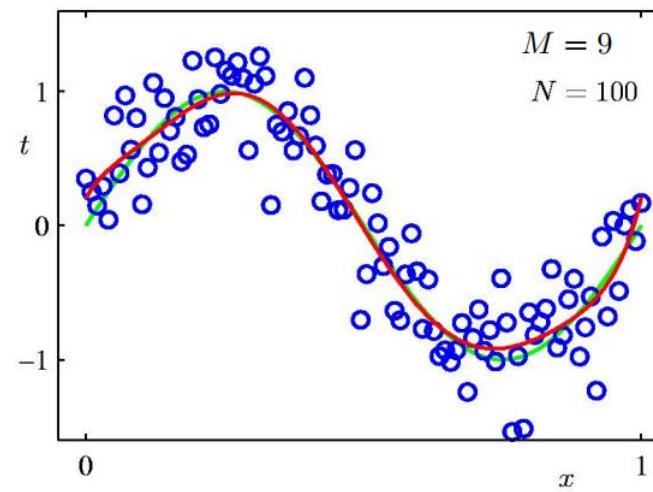
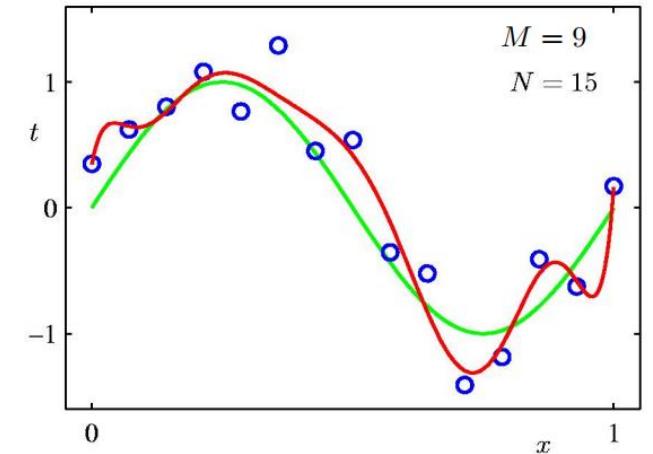
M is the degree of the fitted polynomial



Overfitting



M is the degree of the fitted polynomial, N is the number of data points



Interpretation of coefficients

- We have the following model: $y_i = w_0 + w_1 x_{1i} + w_2 x_{2i} + u_i$
 - u_i is the error term
 - How can w_1 be interpreted?
 - $\frac{\partial y_i}{\partial x_{1i}} = w_1$, i.e. increasing x_{1i} by a unit increases the predicted value of y_i by w_1 units in expectation(if x_{2i} remains constant, i.e. ceteris paribus – other things held constant)
- If the model is: $y_i = w_0 + w_1 x_{1i} + w_2 x_{1i}^2 + u_i$
 - $\frac{\partial y_i}{\partial x_{1i}} = w_1 + 2w_2 x_{1i}$ i.e., the expected effect of increasing x_{1i} on y_i is not constant in x_{1i}

Linear regression with interaction terms

- We can choose a model specification where there are interactions between the explanatory variables
 - It is mainly used if one of the variable is binary (dummy variable)
 - Let the model be: $y_i = w_0 + w_1 x_{1i} + w_2 d_i x_{1i} + u_i$,
 - Where d_i is a binary (dummy) variable
 - $\frac{\partial y_i}{\partial x_{1i}} = w_1 + w_2 d_i$, i.e. the effect also depends on the value of d_i

Stepwise regression

- Automatic procedures to decide which features (explanatory variables) include in the regression
- **Forward selection:** starting with no variables in the model, adding the variable whose inclusion gives the most improvement of the fit, repeating the process until none improves the model to a statistically significant extent
- **Backward elimination:** starting with all candidate variables, deleting the variable whose loss gives the most insignificant decline of the model fit, repeating the process until no further variables can be deleted without a significant loss of fit

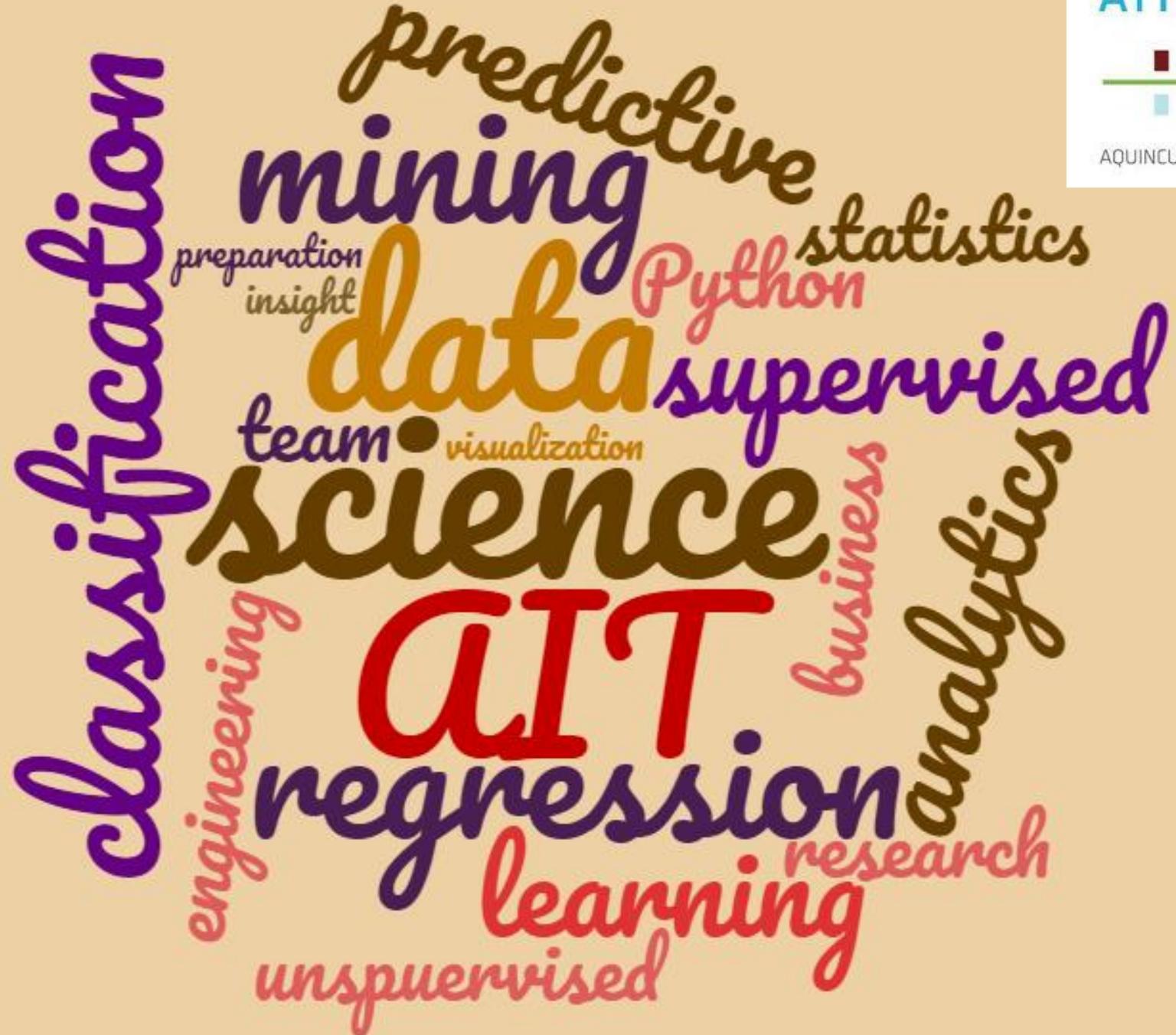
Statistical tests

- Using a t-test we can decide if a coefficient (weight) is significantly different from 0, i.e. the corresponding variable has significant explanatory power in the model
- Using an F-test it can be determined whether the full regression model has significant explanatory variable regarding the target variable

Acknowledgement

- András Benczúr, Róbert Pálovics, SZTAKI-AIT, DM1-2
- Krisztián Buza, MTA-BME, VISZJV68
- Bálint Daróczy, SZTAKI-BME, VISZAMA01
- Judit Csima, BME, VISZM185
- Gábor Horváth, Péter Antal, BME, VIMMD294, VIMIA313
- Lukács András, ELTE, MM1C1AB6E
- Tim Kraska, Brown University, CS195
- Dan Potter, Carsten Binnig, Eli Upfal, Brown University, CS1951A
- Erik Sudderth, Brown University, CS142
- Joe Blitzstein, Hanspeter Pfister, Verena Kaynig-Fittkau, Harvard University, CS109
- Rajan Patel, Stanford University, STAT202
- Andrew Ng, John Duchi, Stanford University, CS229





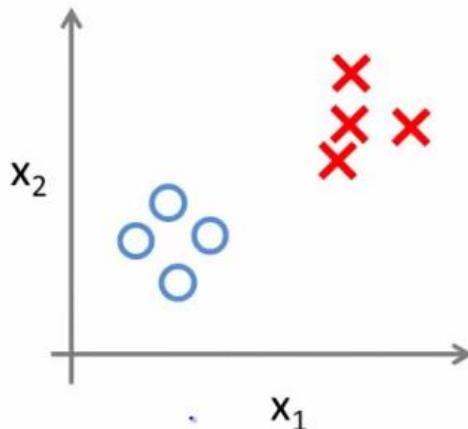
Schedule of the semester

	<i>Monday midnight</i>	<i>Tuesday class</i>	<i>Friday class</i>
W1 (02/06)			
W2 (02/13)		HW1 out	
W3 (02/20)			
W4 (02/27)	HW1 deadline + TEAMS	HW2 out	
W5 (03/06)			PROJECT PLAN
W6 (03/13)	HW2 deadline	HW3 out	
W7 (03/20)			MIDTERM
SPRING BREAK		SPRING BREAK	SPRING BREAK
W8 (04/03)	HW3 deadline		GOOD FRIDAY
W9 (04/10)	MILESTONE 1		
W10 (04/17)		HW4 out	
W11 (04/24)			
W12 (05/01)	HW4 deadline		
W13 (05/08)	MILESTONE 2		
W14 (05/15)		FINAL	PROJECT presentations
W15 (05/22)		PROJECT presentations	

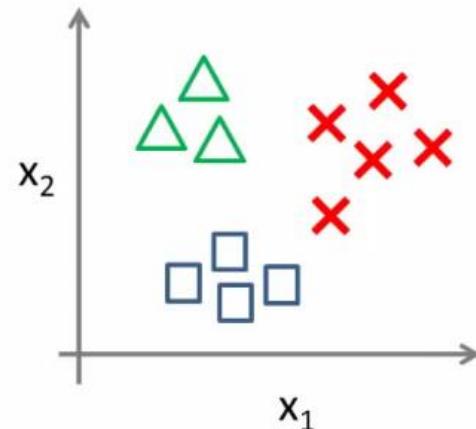
Logistic regression

- It is a classification algorithm not a regression algorithm (despite its name)
- Classification via regression
- Binary classification algorithm

Binary classification:

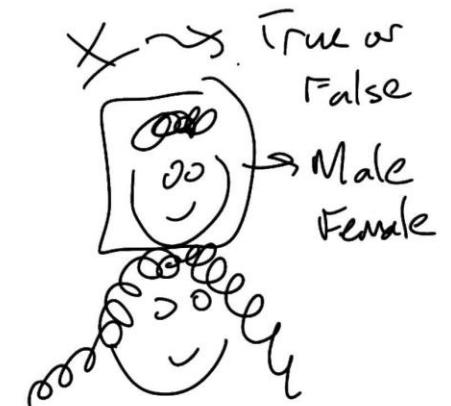


Multi-class classification:



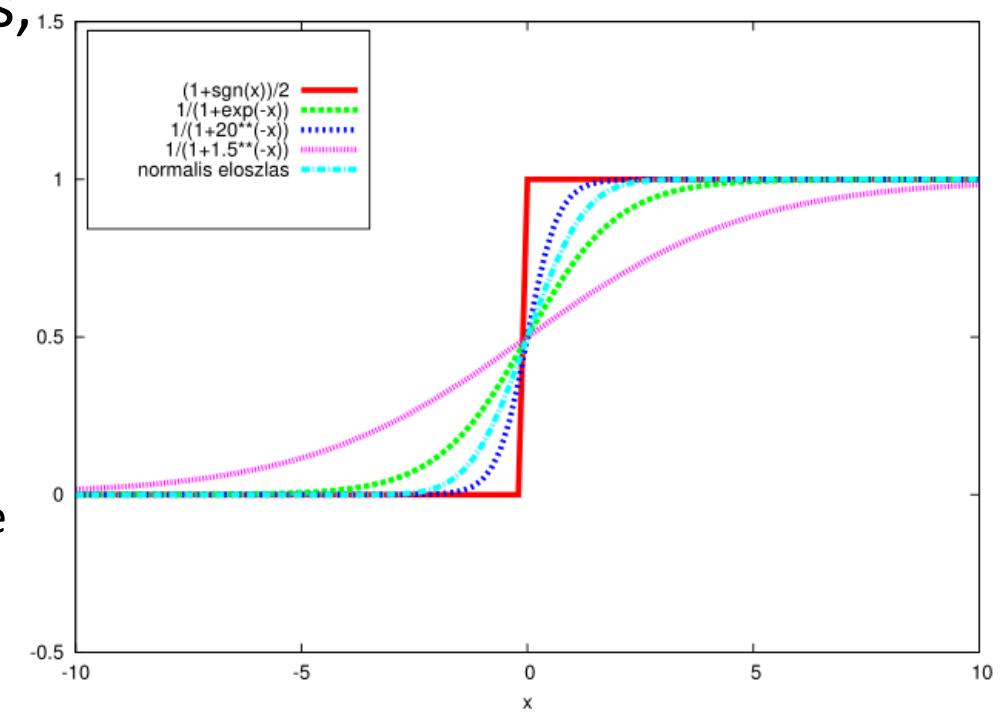
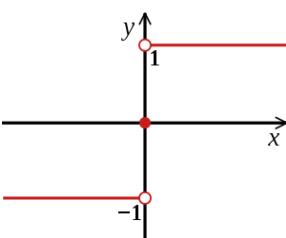
SUPERVISED LEARNING

✓ CLASSIFICATION
✓ REGRESSION



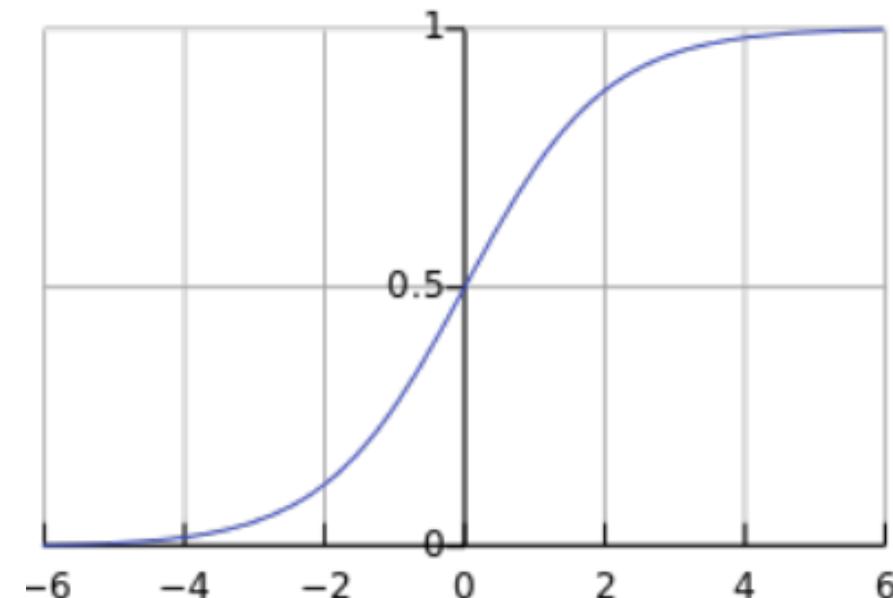
Classification via regression

- We fit a linear regression, if the predicted value is positive we assign the positive class, if the predicted value is negative we assign the negative class to the record
 - It means that we apply a step function on the result of the linear regression
 - We can also smooth the sign function:
 - It is desirable to have smoother function
 - It is a natural approach that the closer we are to the decision boundary, the more uncertain we are in our decision (the boundary is not sharp but rather smooth)
 - We can think of it as we predict how likely the positive label is, not the label itself

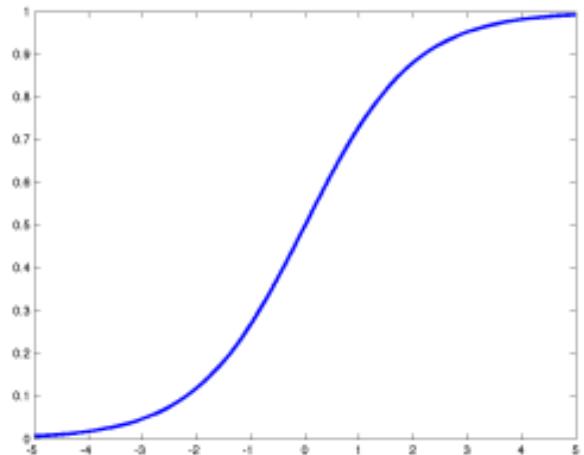


Logistic regression

- Training data: $(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \dots, (\mathbf{x}_n, y_n)$, where $\mathbf{x}_i = (x_{i1}, x_{i2}, \dots, x_{ip})$
 - From now on we use a slightly modified notation as a matter of convenience
Introduce the following notation: $x_{i0} = 1, \forall i$
 - Also: $\mathbf{x}_i = (x_{i0}, x_{i1}, x_{i2}, \dots, x_{ip})$
and $\mathbf{w} = (w_0, w_1, w_2, \dots, w_p)$
 - Using this new notation we can write $\mathbf{w}^T \mathbf{x}_i$
instead of $w_0 + \mathbf{w}^T \mathbf{x}_i$
- Hypothesis for linear regression:
$$\hat{y}_i = h_{\mathbf{w}}(\mathbf{x}_i) = \mathbf{w}^T \mathbf{x}_i$$
- Hypothesis for logistic regression:
$$h_{\mathbf{w}}(\mathbf{x}_i) = \sigma(\mathbf{w}^T \mathbf{x}_i)$$
 - Sigmoid function (logistic function)
$$\sigma(x) = \frac{1}{1+e^{-x}}$$
 - Probabilistic interpretation



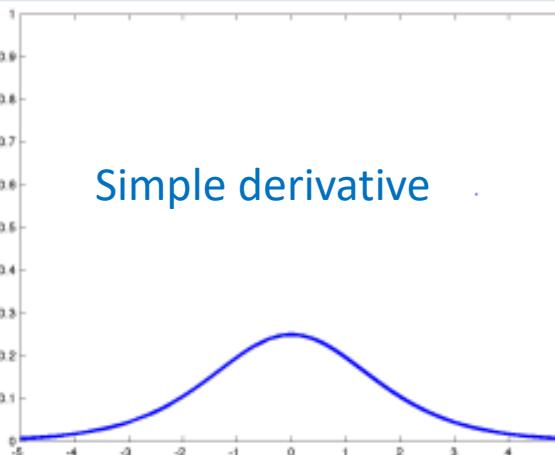
Characteristics of the sigmoid (logistic) function



$$\sigma(z) = \frac{1}{1 + e^{-z}}$$

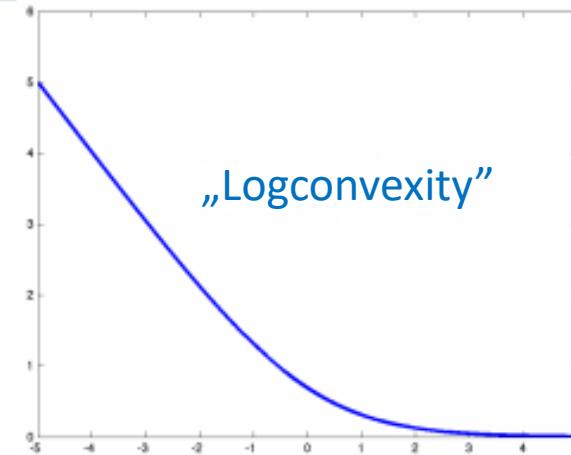
Symmetry:

$$\sigma(-z) = 1 - \sigma(z)$$



Simple derivative

$$\frac{\partial \sigma(z)}{\partial z} = \sigma(z)\sigma(-z)$$



„Logconvexity“

$$-\log \sigma(z) = \log(1 + e^{-z})$$

Log odds ratio:

$$\log \frac{\sigma(z)}{1 - \sigma(z)} = z$$

What is the cost function?

- For linear regression (with regularization)

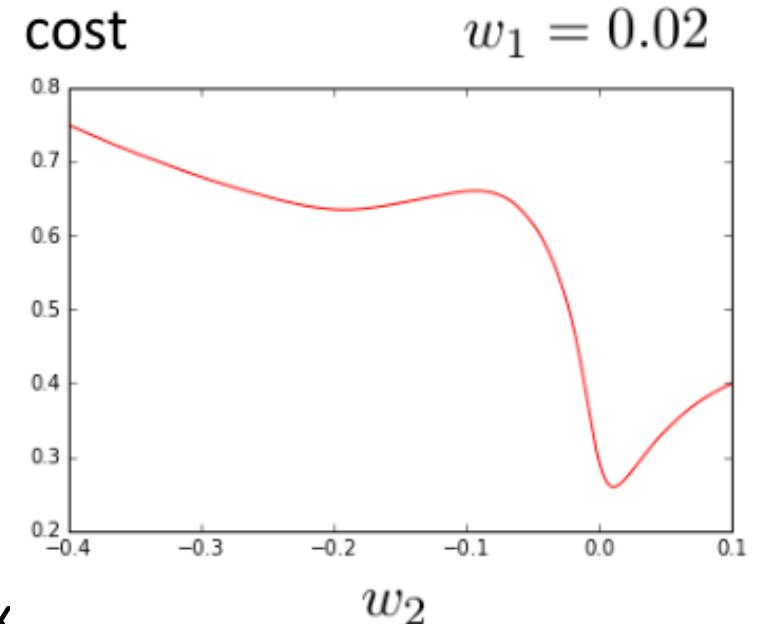
$$\frac{1}{n} \sum_{i=1}^n (y_i - \mathbf{w}^T \mathbf{x}_i)^2 + \lambda |\mathbf{w}|^2$$

- Can we use the „same” for logistic regression?

$$\frac{1}{n} \sum_{i=1}^n (y_i - \sigma(\mathbf{w}^T \mathbf{x}_i))^2 + \lambda |\mathbf{w}|^2$$

- It is not practical because this function is not convex

- Sample data:
 $x=[[1,26], [1,33], [1,34], [25,33], [14,42], [32,56], [32,59], [1,120], [1,76], [1,80], [1,92], [25,135], [1,150], [1,26], [315,35], [218,39], [52,43]]$
 $y=[0, 0, 0, 1, 1, 1, 1, 0, 1, 1, 1, 1, 0, 0, 0, 1]$



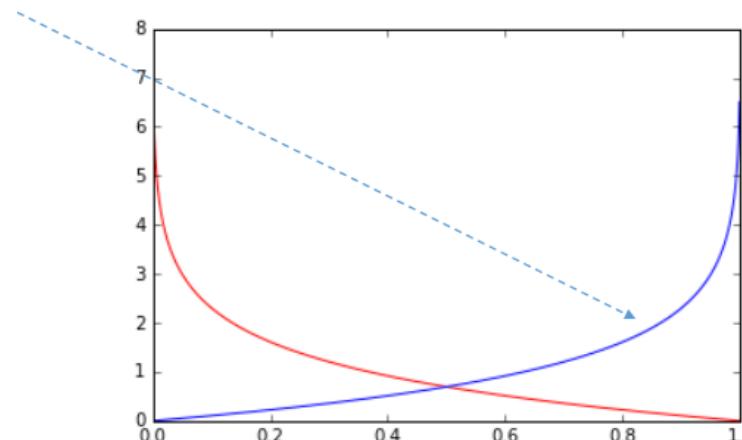
Cost function for logistic regression

- We form the cost function in the following way:
 - Cost for a single record (logarithmic cost)

$$cost(h_w(x), y) = \begin{cases} -\log(h_w(x)), & \text{if } y = 1 \\ -\log(1 - h_w(x)), & \text{if } y = 0 \end{cases}$$

- Total cost for the data set

$$\frac{1}{n} \sum_{i=1}^n cost(h_w(x_i), y_i)$$



Cost function in closed form

- Reminder: $\text{cost}(h_{\mathbf{w}}(\mathbf{x}), y) = \begin{cases} -\log(h_{\mathbf{w}}(\mathbf{x})), & \text{if } y = 1 \\ -\log(1 - h_{\mathbf{w}}(\mathbf{x})), & \text{if } y = 0 \end{cases}$
- With a simple transformation, the cost for a single record:
$$\text{cost}(h_{\mathbf{w}}(\mathbf{x}), y) = -y \log(h_{\mathbf{w}}(\mathbf{x})) - (1 - y) \log(1 - h_{\mathbf{w}}(\mathbf{x}))$$
- Total cost:

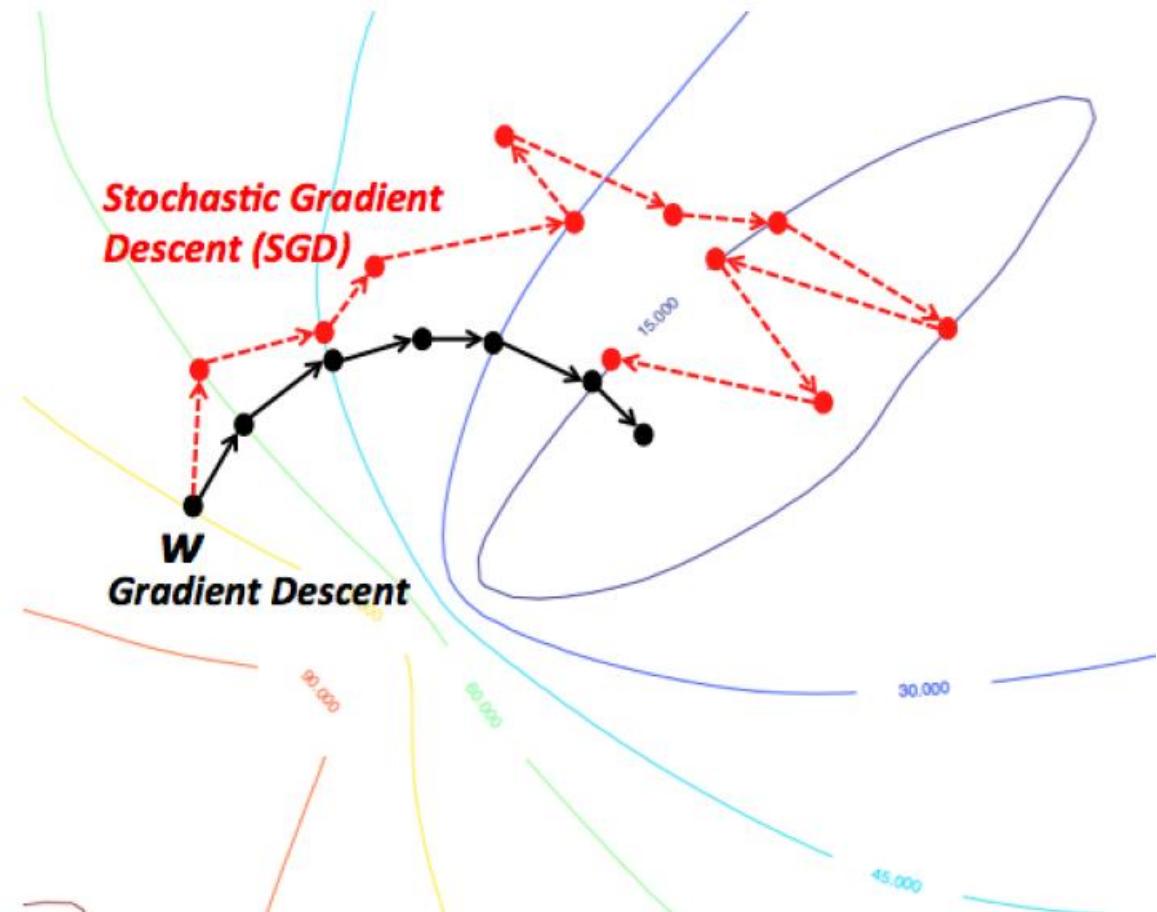
$$C(\mathbf{w}) = -\frac{1}{n} \sum_{i=1}^n (y_i \log(h_{\mathbf{w}}(\mathbf{x}_i)) + (1 - y_i) \log(1 - h_{\mathbf{w}}(\mathbf{x}_i)))$$

- Similarly to linear regression we can add the regularization term:

$$C(\mathbf{w}) + \lambda |\mathbf{w}|^2$$

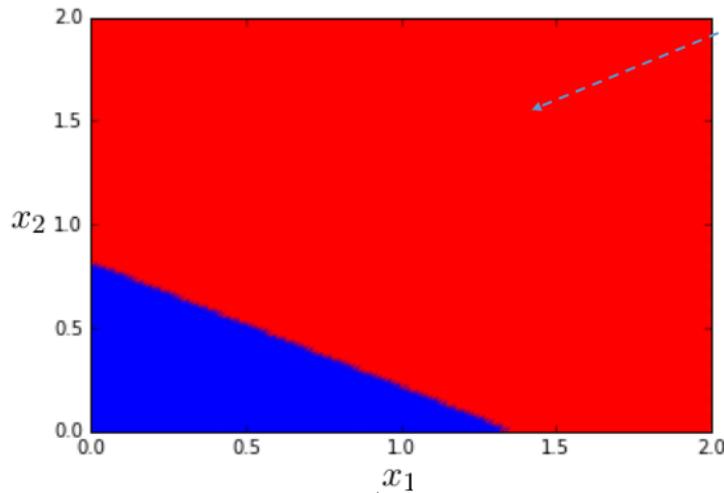
Minimizing the cost function

- It is possible to minimize the cost function
 - Analytically (very slow)
 - With gradient descent
 - With stochastic gradient descent
 - With other optimizing methods



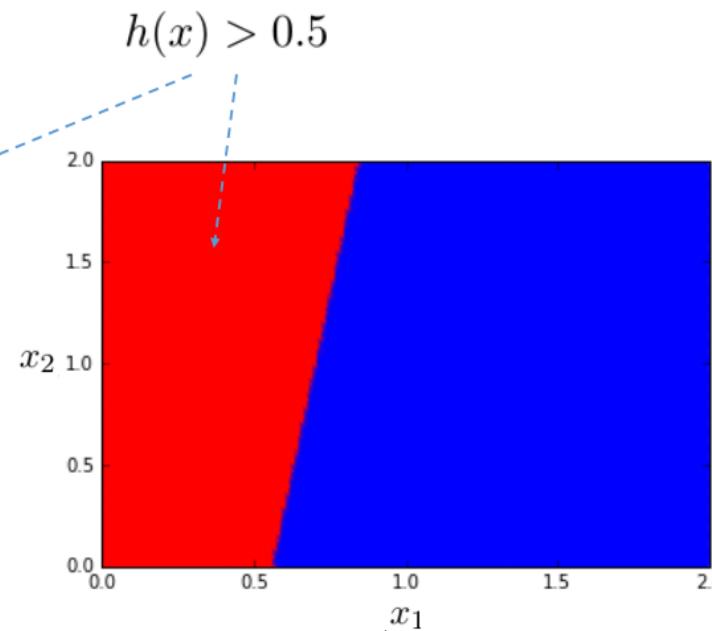
Linear decision boundary

- Having a linear hypothesis ($\mathbf{w}^T \mathbf{x}_i$) the decision boundary of the logistic regression is also linear



$$h(x) = \sigma(-1 + 0.75x_1 + 1.25x_2)$$

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

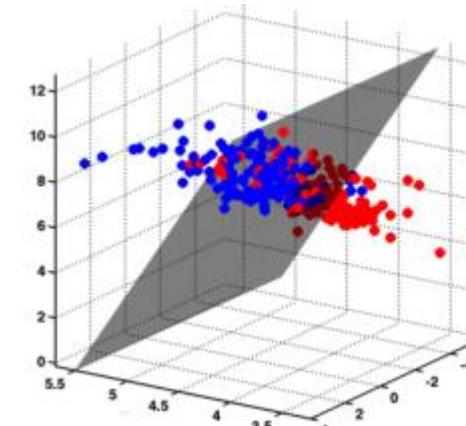
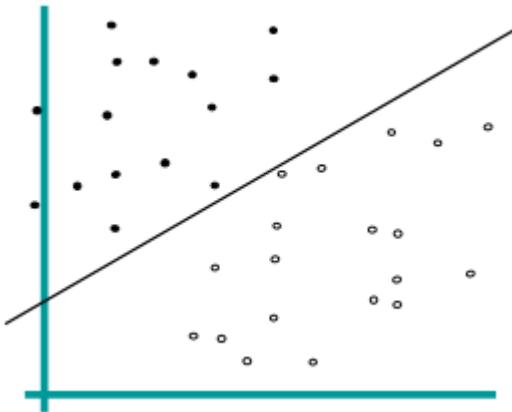


$$h(x) = \sigma(1 - 1.75x_1 + 0.25x_2)$$

Linear separability

- The data is linearly separable if

- In 2D: there exists a line in the plane with all the positive („blue”) points on one side and all the negative „red” points on the other side
- In 3D: there exists such a plane
- In higher dimension: such a hyperplane



- The algorithm is looking for the equation of the separating hyperplane

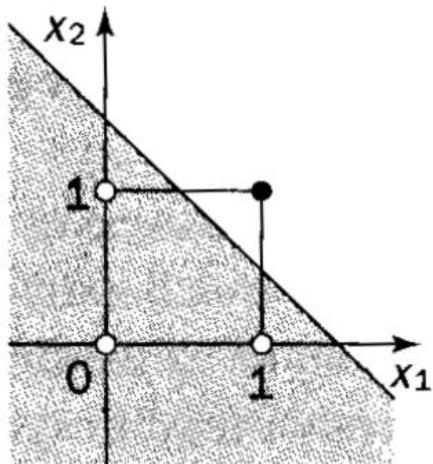
$$\mathbf{w}^T \mathbf{x} = 0$$

$$w_0 + w_1 x_1 + w_2 x_2 + \cdots + w_p x_p = 0$$

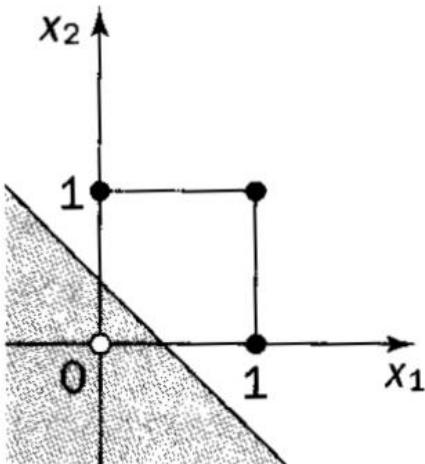
- One side of the hyperplane (positive records): $\mathbf{w}^T \mathbf{x} > 0$
- Other side of the hyperplane (negative records): $\mathbf{w}^T \mathbf{x} < 0$
- Vector \mathbf{w} is orthogonal to the hyperplane it gives the direction of the separation

Linear separability of Boolean functions

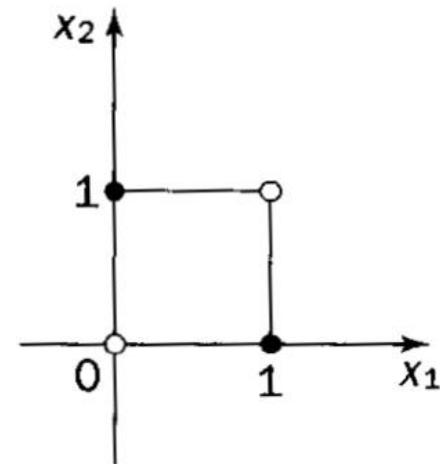
- Some Boolean functions are linearly separable but not all



(a) AND ($x_1 \cap x_2$)



(b) OR ($x_1 \cup x_2$)

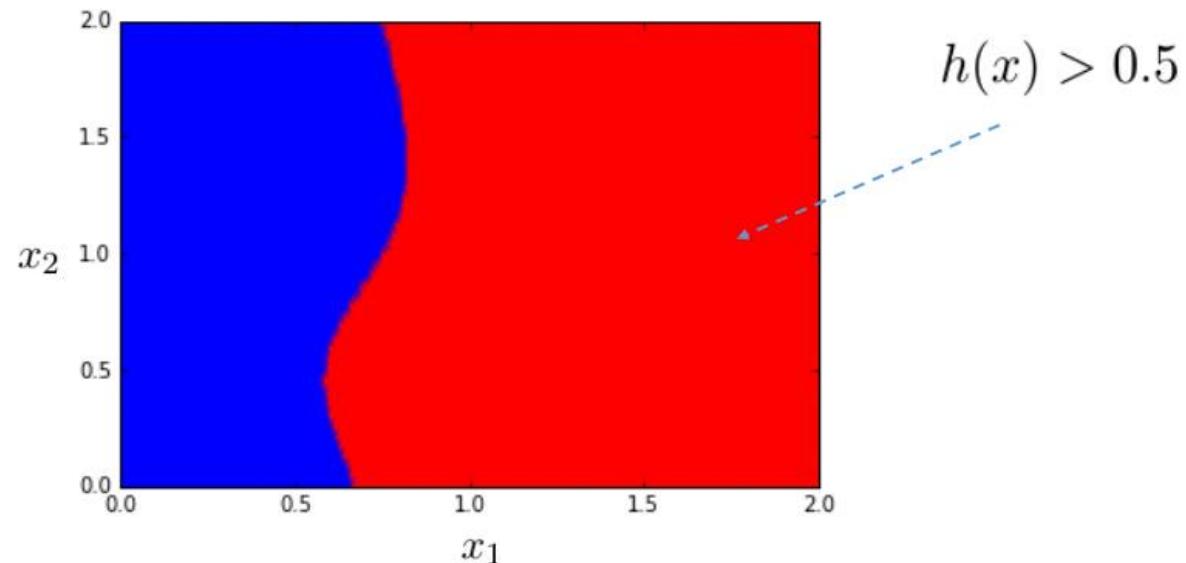


(c) Exclusive-OR
($x_1 \oplus x_2$)



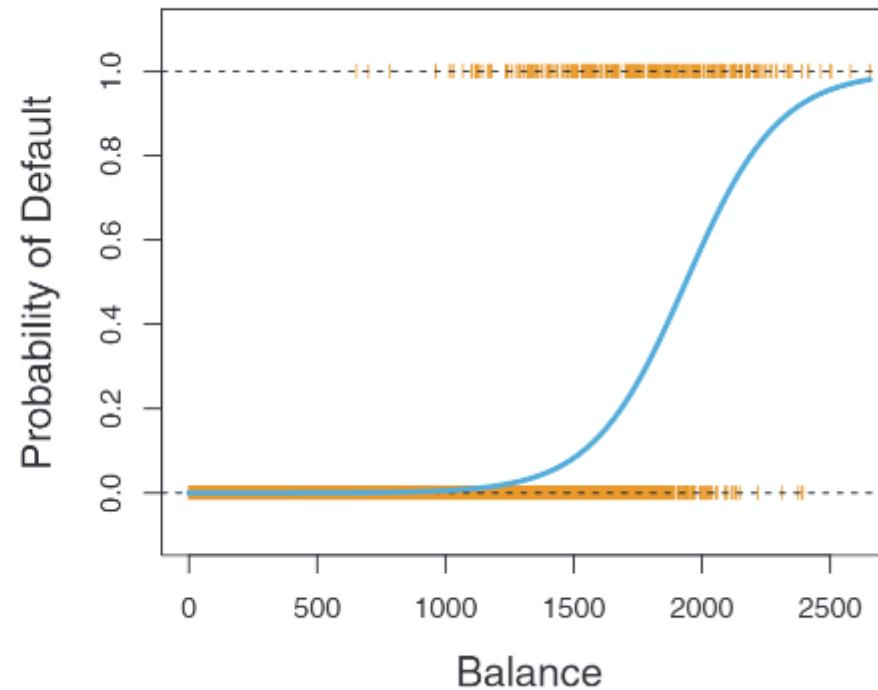
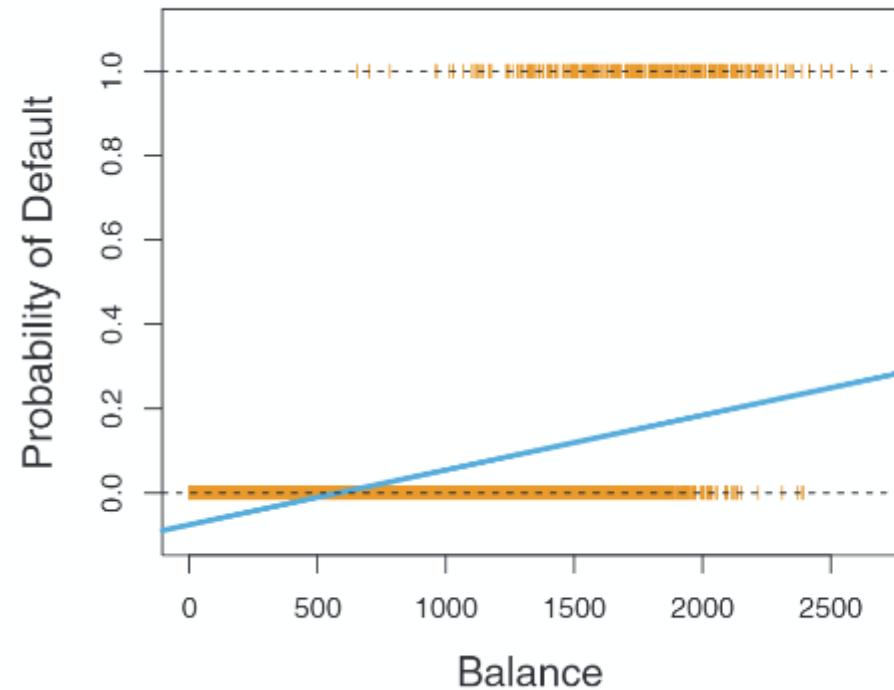
Non-linear decision boundary

- If higher powers of the variables are also allowed in the hypothesis than logistic regression can also create non-linear decision boundary



$$h(x) = \sigma(-2 + 0.3x_1 + 0.25x_1^2 + 6x_1^3 + 2x_2 + 0.5x_2^2 - 4x_2^3 + 2x_1^3x_2^5)$$
$$x = (x_1, x_2)$$

Linear vs. logistic regression

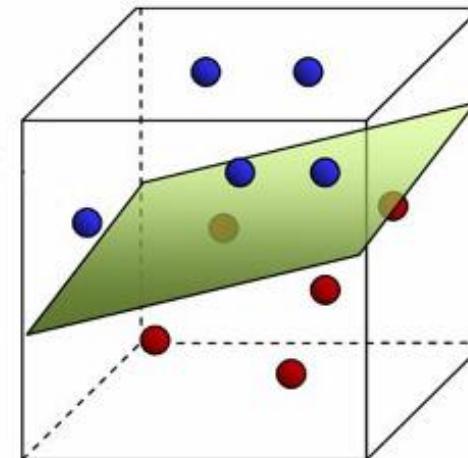


Logit model

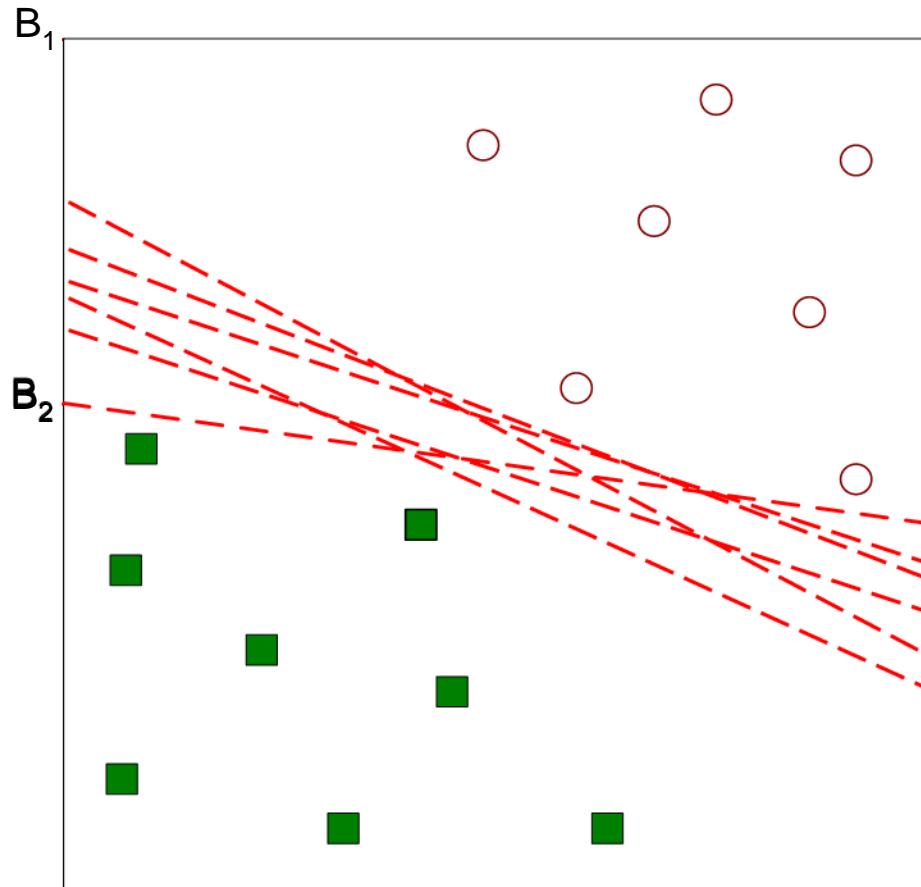
- The inverse function of sigmoid function is the logit function, another way to look at it
- Based on the probabilistic interpretation let p be:
- $p = h_{\mathbf{w}}(\mathbf{x}) = \sigma(\mathbf{w}^T \mathbf{x}) = \frac{1}{1+e^{-\mathbf{w}^T \mathbf{x}}}$
- After inverting : $\ln\left(\frac{p}{1-p}\right) = \mathbf{w}^T \mathbf{x} = \mathbf{w}_0 + \mathbf{w}_1 \mathbf{x}_1 + \cdots + \mathbf{w}_p \mathbf{x}_p$
 - This function is the logit function: $\text{logit}(p) = \ln\left(\frac{p}{1-p}\right)$
 - Based on that logistic regression is also called as logit model

Support Vector Machine (SVM)

- Normally SVM is used to classify (linearly separate) binary data
 - It can be extended for multi-class classification, for non-linear separation and for non-linear regression
 - Goal: to find a separating hyperplane that separate the records of the two classes well
 - What do we mean by „well”?



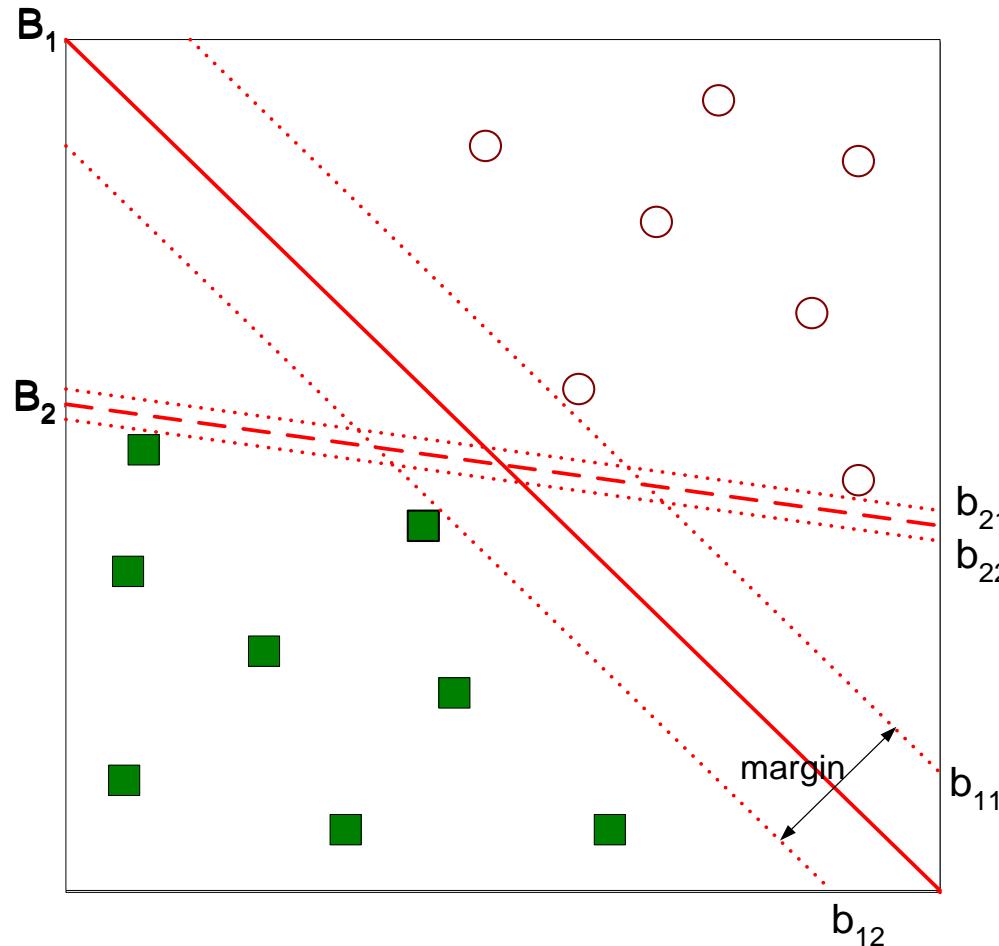
Linear separation, but how?



- If the data is linearly separable there are infinitely many separable line (hyperplane)
- Which to choose?

Which is the better linear separator?

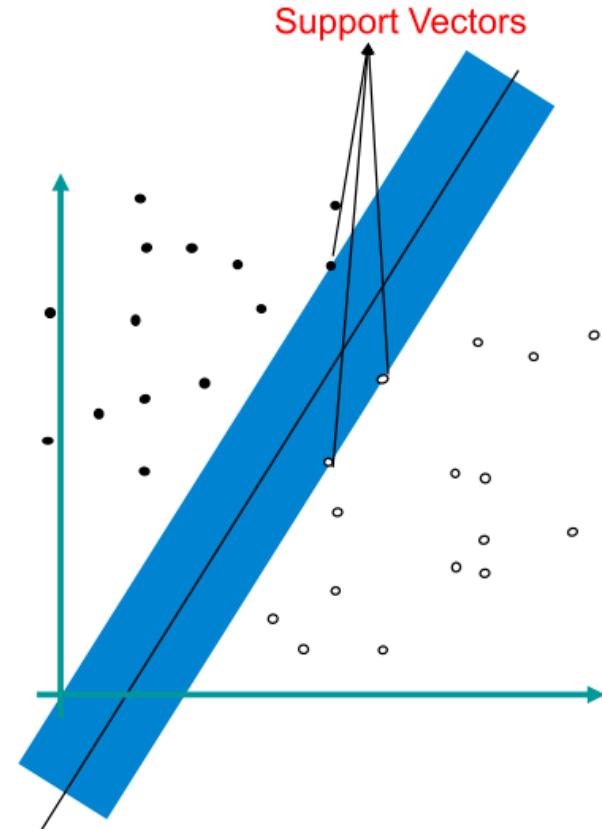
- B_1 or B_2 is the better?
- What do we mean by goodness?
 - The width of margin
 - „Maximizing the margin”
 - So B_1 is better



- Margin: The maximal width of the slice parallel to the separating hyperplane that has no interior data points

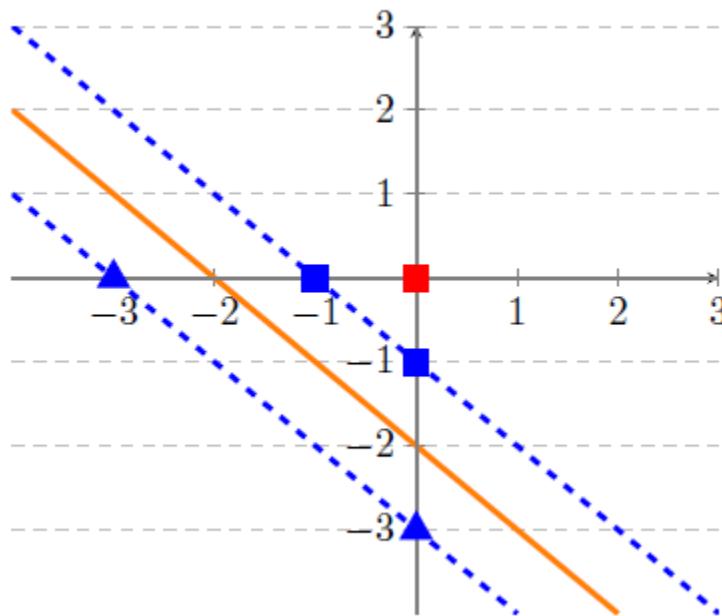
Support vectors

- Support vectors are the data points closest to the separating hyperplane from both classes
- Support vectors are the records of the training set that would change the position of the dividing (separating) hyperplane if removed
 - The „critical” elements



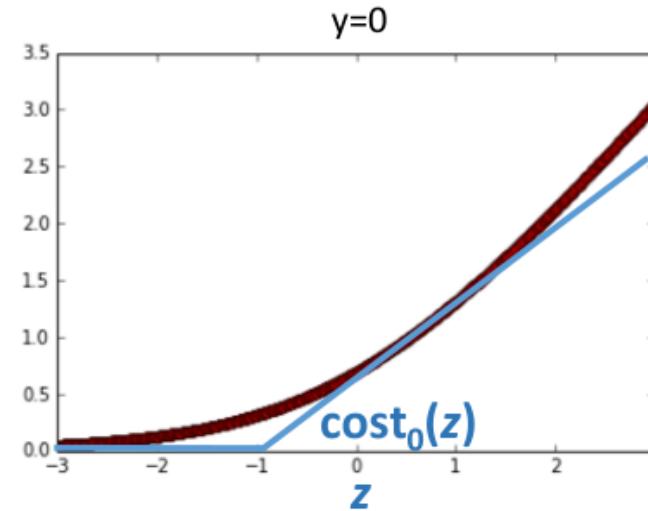
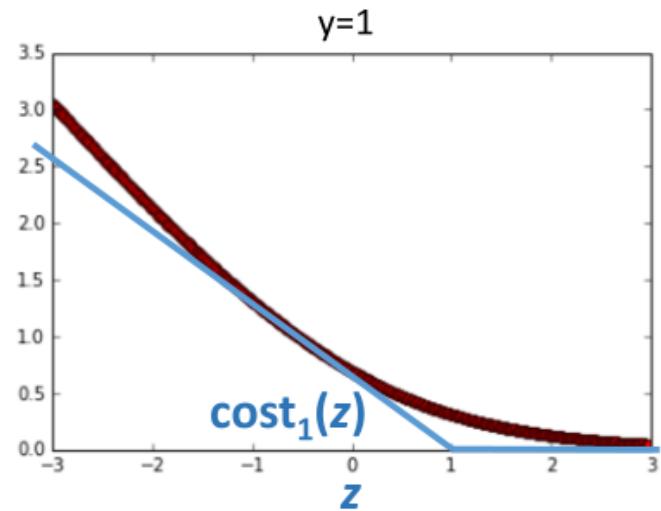
Problem

How does the $2 + x + y = 0$ line separates the 2-dimensional plane? Draw the line in a coordinate system. Which of the following records are support vectors of the given line: $(-3, 0)$; $(0, -3)$; $(-1, 0)$; $(0, -1)$; $(0, 0)$?



Cost function for SVM

- Cost for one single record
 - For logistic regression : $cost(h_{\mathbf{w}}(\mathbf{x}), y) = -y \log(h_{\mathbf{w}}(\mathbf{x})) - (1 - y) \log(1 - h_{\mathbf{w}}(\mathbf{x}))$ where $h_{\mathbf{w}}(\mathbf{x}) = \sigma(\mathbf{w}^T \mathbf{x})$
 - For SVM it is modified a bit (see figure)
 - Black curve: cost function for logistic regression
 - Blue curve: cost function for SVM
 - $z = \mathbf{w}^T \mathbf{x}$



Cost function for SVM II.

- Total cost function for logistic regression:

$$-\frac{1}{n} \sum_{i=1}^n (y_i \log(h_{\mathbf{w}}(\mathbf{x}_i)) + (1 - y_i) \log(1 - h_{\mathbf{w}}(\mathbf{x}_i))) + \lambda \sum_{j=0}^m w_j^2$$

- Total cost function for SVM:

- Here the weights are denoted by $\boldsymbol{\theta}$ instead of \mathbf{w}

$$C \sum_{i=1}^n (y_i cost_1(\boldsymbol{\theta}^T \mathbf{x}_i) + (1 - y_i) cost_0(\boldsymbol{\theta}^T \mathbf{x}_i)) + \frac{1}{2} \sum_{j=0}^m \theta_j^2$$

- Where C is a parameter to set, and the two cost functions are:

$$cost_j(\boldsymbol{\theta}^T \mathbf{x}_i) = \begin{cases} \max \{0, 1 - \boldsymbol{\theta}^T \mathbf{x}_i\}, & \text{if } j = 1 \\ \max \{0, 1 + \boldsymbol{\theta}^T \mathbf{x}_i\}, & \text{if } j = 0 \end{cases}$$

Minimizing the cost function

- If in the total cost function $c \sum_{i=1}^n (y_i \text{cost}_1(\boldsymbol{\theta}^T \mathbf{x}_i) + (1 - y_i) \text{cost}_0(\boldsymbol{\theta}^T \mathbf{x}_i)) + \frac{1}{2} \sum_{j=0}^m \theta_j^2$

C is chosen to be big enough then the minimization task is equivalent with the following constrained optimization problem:

$$\min_{\boldsymbol{\theta}} \frac{1}{2} \sum_{j=0}^m \theta_j^2$$

subject to

$$\begin{aligned}\boldsymbol{\theta}^T \mathbf{x}_i &\geq 1, & \text{if } y_i = 1 \\ \boldsymbol{\theta}^T \mathbf{x}_i &\leq -1, & \text{if } y_i = 0\end{aligned}$$

- Constrained problems can be solved e.g. using Lagrange multipliers

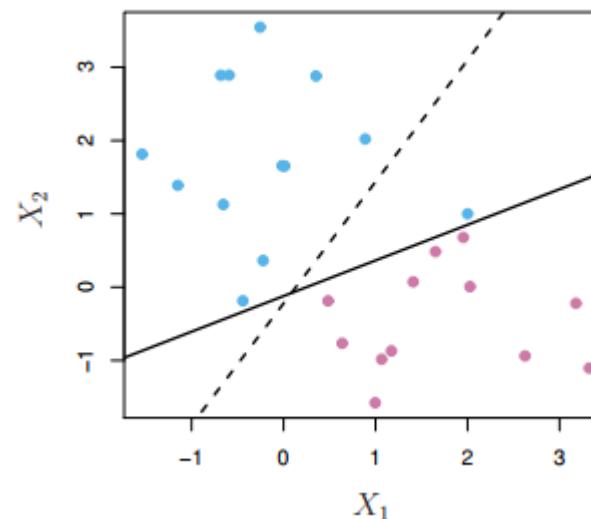
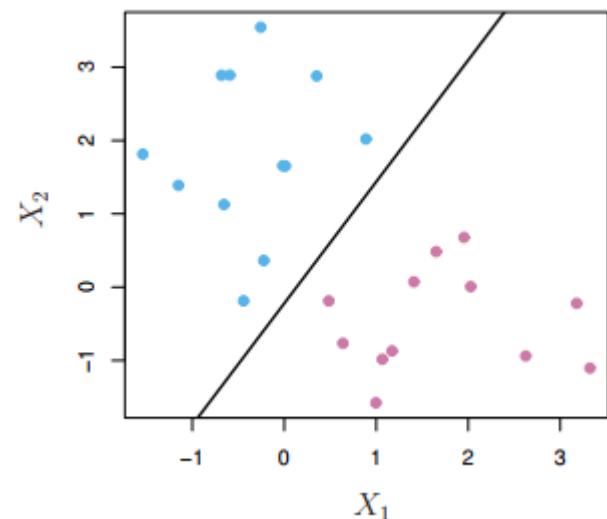
Connection of the minimization problem to the maximal margin

- Solving the previous minimization problem the SVM finds a separating hyperplane that maximizes the margin
- If the data is not linearly separable, then it finds a hyperplane that has a small misclassification error and big margin
 - Which is more important that depends on the constant C
- The equation of the hyperplane: $\theta^T x = 0$

The role of C in the cost function

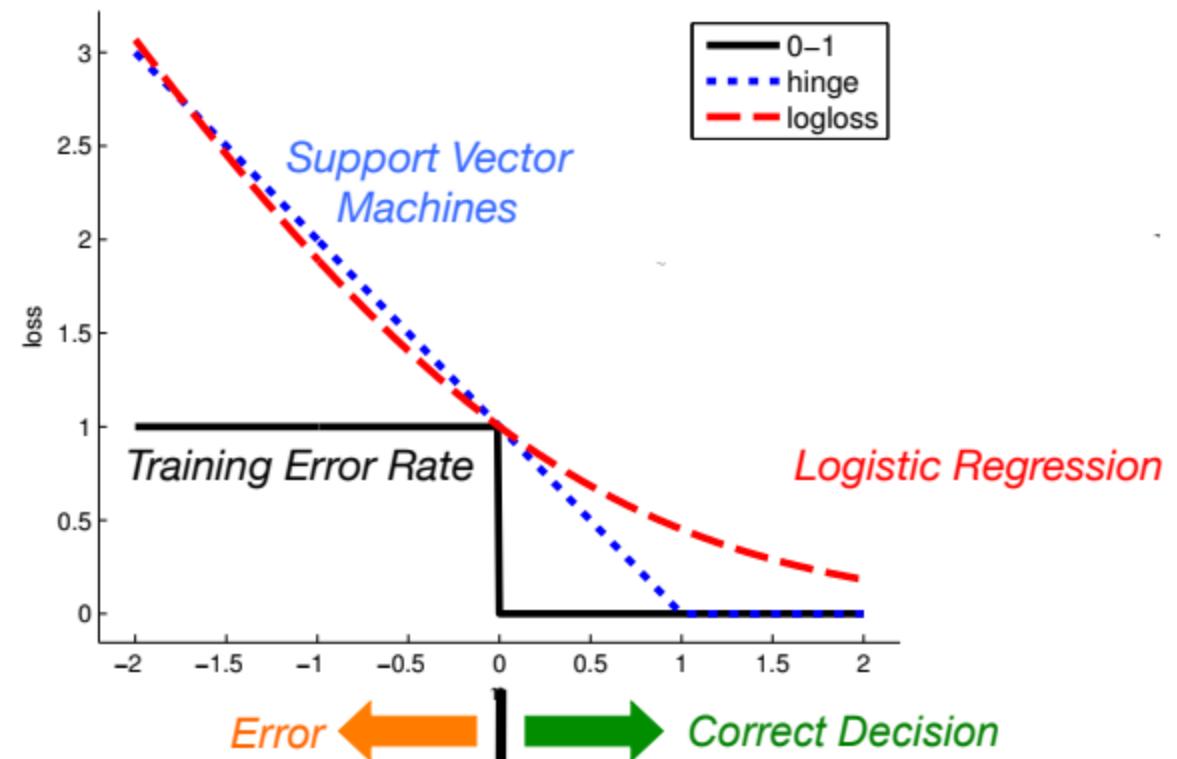
$$C \sum_{i=1}^n (y_i \text{cost}_1(\boldsymbol{\theta}^T \mathbf{x}_i) + (1 - y_i) \text{cost}_0(\boldsymbol{\theta}^T \mathbf{x}_i)) + \frac{1}{2} \sum_{j=1}^m \theta_j^2$$

- If C is large than it is more expensive to make misclassification errors so it gives solution with less misclassification error but a smaller margin
 - Risk of overfitting (high variance)
 - Even the addition of one data point can cause big changes in the result
- If C is small, it focuses more on finding a hyperplane with big margin



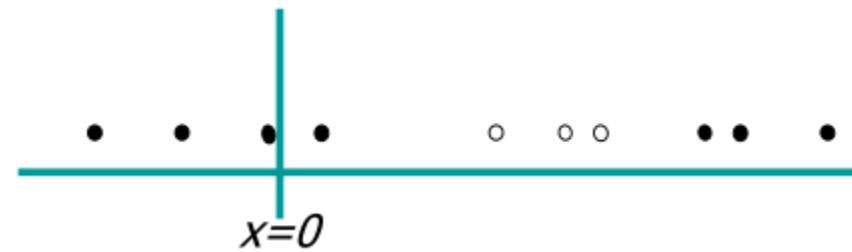
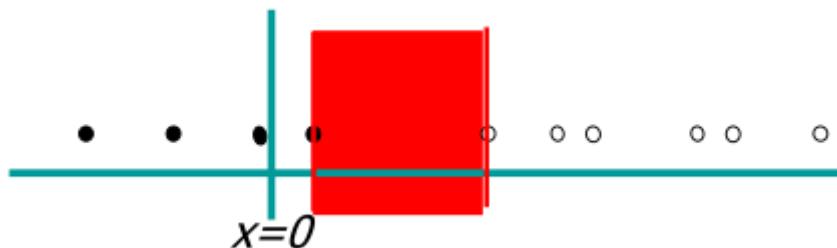
Cost functions for binary classification

- 0-1 loss
 - For some applications that is the important
 - It is difficult to optimize for (not differentiable)
- Logarithmic loss (for logistic regression)
 - Probabilistic interpretation of the output
 - Easy to optimize for (convex)
 - Scalable for big data
- Hinge loss (for SVM)
 - A loss function that maximizes the margin
 - Convex function (usual convex optimizers work)
 - Scalable for big data



Linear separability in 1D

- Are the following data linearly separable?
 - What can be done for the second one? Is it possible to make it linearly separable?

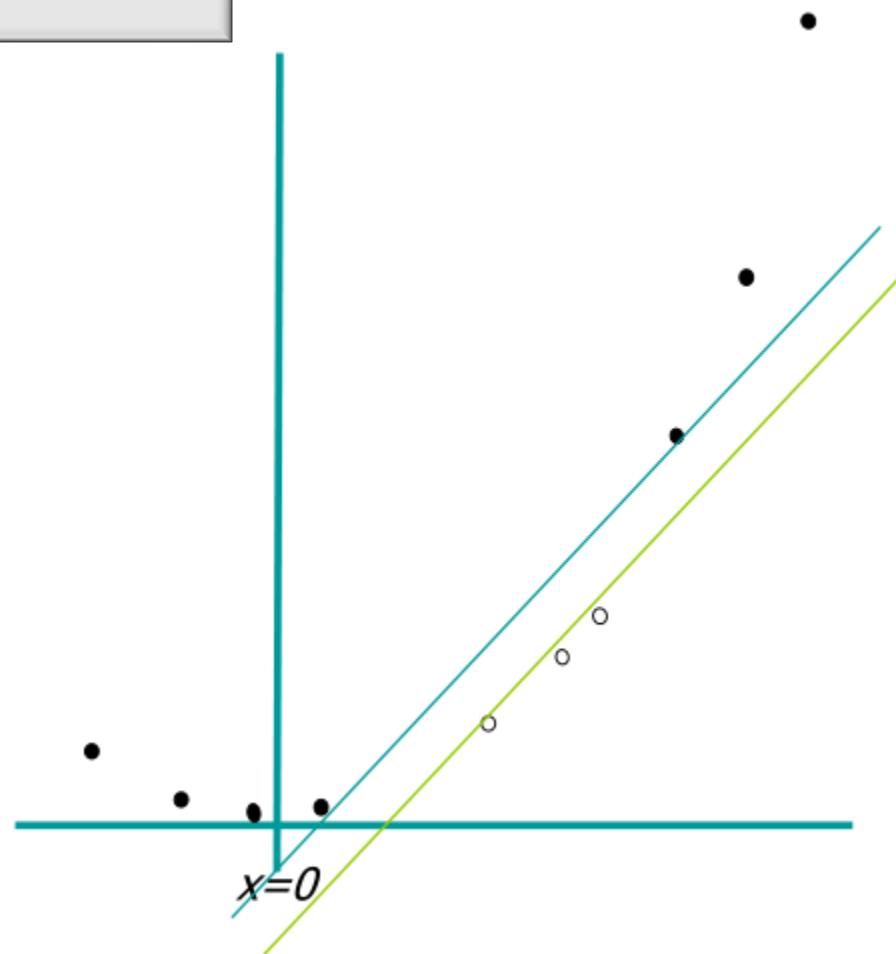


Transformation to higher dimension

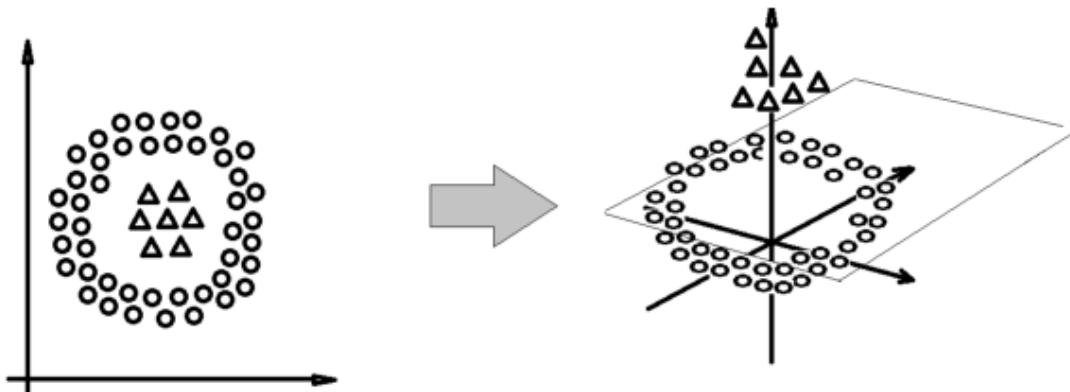
- Apply the following transformation:

$$z_i = (x_i, x_i^2)$$

- Now it is linearly separable!

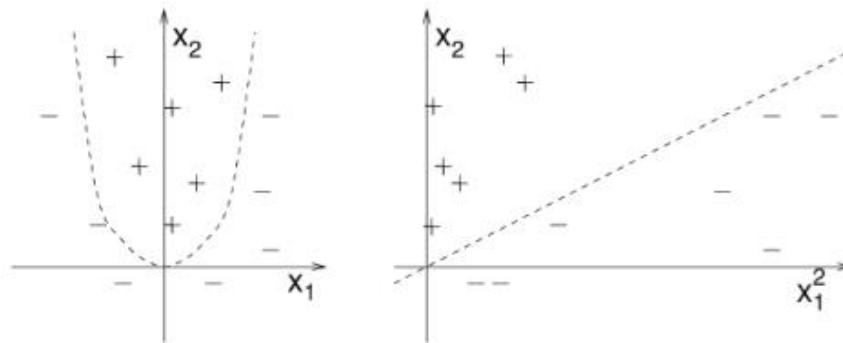


Examples for transformations

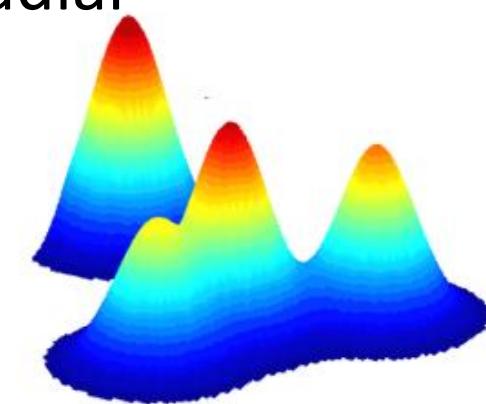


Radial

Quadratic

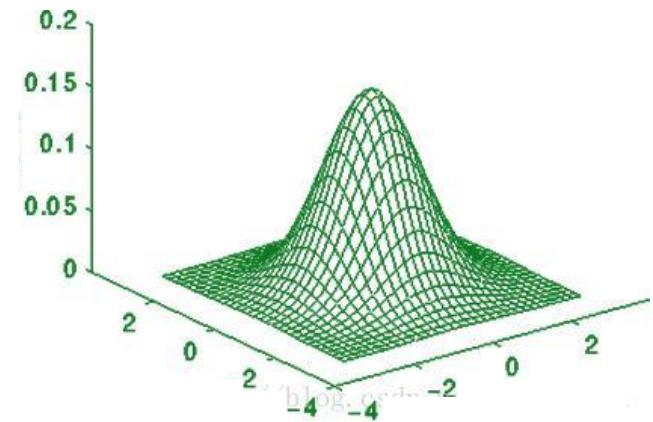


Radial



Examples for transformations II.

- Polynomial
 $\mathbf{z}_i = (\text{some polynomial expressions of } \mathbf{x}_i)$
- Radial
 - Radial function is a function whose value at each point depends only on the distance between that point and a certain point (e.g the origin or \mathbf{c} in the example)
$$\mathbf{z}_i = \exp\left(-\frac{|\mathbf{x}_i - \mathbf{c}|^2}{\sigma^2}\right)$$
- Sigmoid
 $\mathbf{z}_i = (\text{sigmoid functions of } \mathbf{x}_i)$



Problem

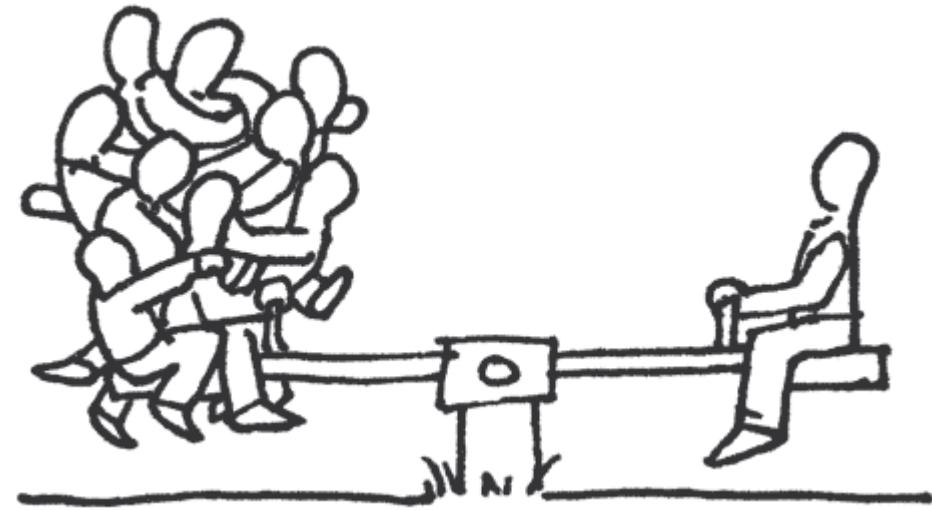
Consider the following data, where the first and the second coordinates are binary attributes, and the third is the class label: $(1, 1, -)$; $(1, -1, +)$; $(-1, 1, +)$; $(-1, -1, -)$. Which Boolean function do you recognize in the data? Is it linearly separable? If so, give the equation of the separating line with maximal margin. If the function is not linearly separable, then transform the data into the following three-dimensional feature space: $(x_1, x_2, x_1 \cdot x_2)$. Furthermore, find the equation of the separating plane with the maximum margin in the transformed space!

Multi-class classification with binary classifiers

- Logistic regression and SVM are binary classifiers
- How can we solve multi-class classification problems with them?
- One-versus-one (OvO) strategy: The problem is reduced to $\binom{C}{2}$ binary classification problems
 - Each receives the samples of a pair of classes from the original training set and must distinguish these two classes
 - At prediction a voting scheme is applied: the class that got the highest number of „votes” (+1 predictions) get predicted by the combined classifier
 - Ambiguity: it is not necessarily unique
 - The result of the combined classifier can be right even if some of the pairwise classifiers were not right
 - Computationally expensive

One-versus-rest (OvR) strategy

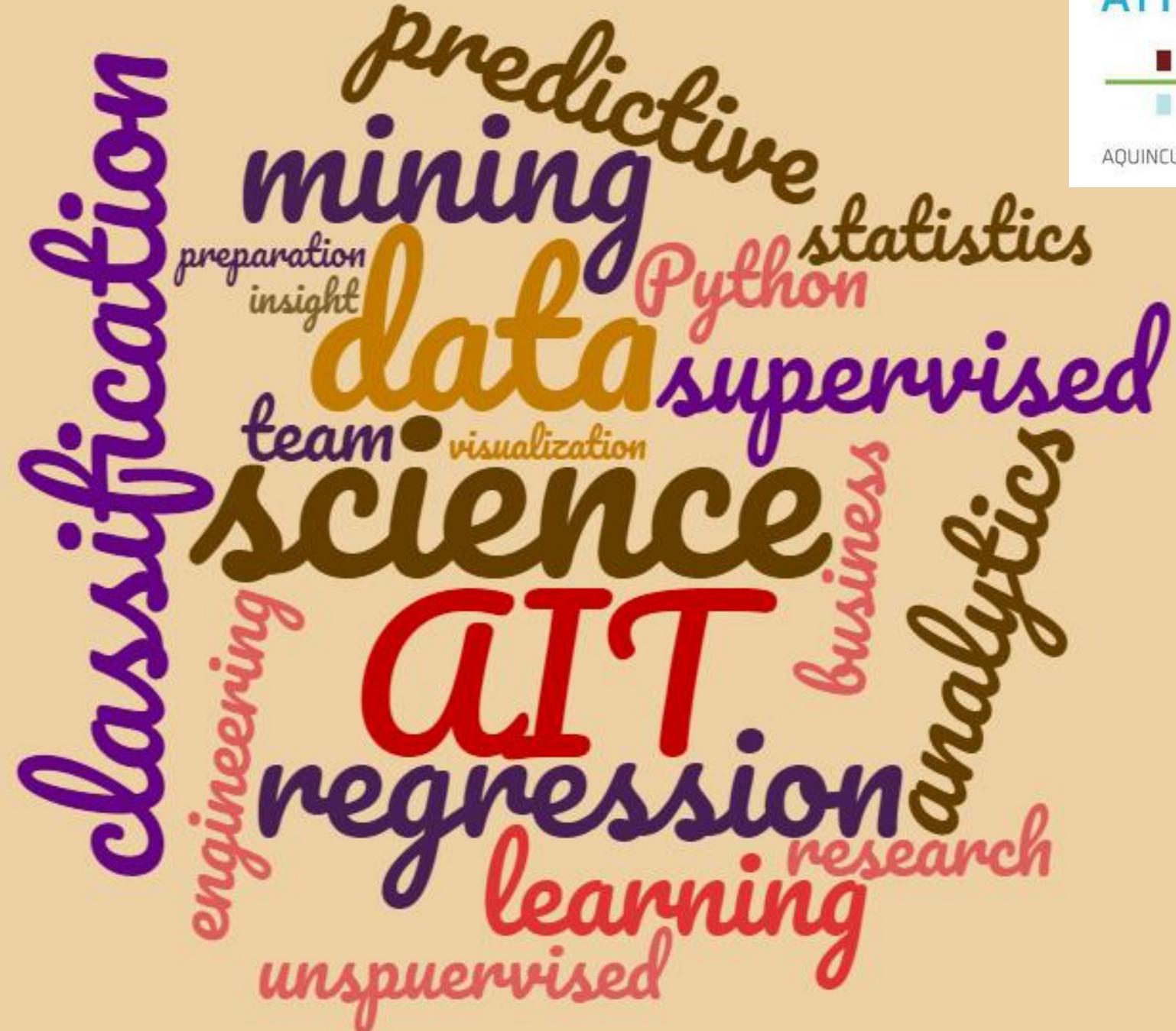
- Training a single classifier per class with the samples of that class as positive samples and the samples of all other classes as negatives (creating a virtual class)
- The base classifier produces a real-valued confidence score for its decision
- Making decisions means applying all classifiers to an unseen sample and predicting the label for which the corresponding classifier reports the highest confidence score



Acknowledgement

- András Benczúr, Róbert Pálovics, SZTAKI-AIT, DM1-2
- Krisztián Buza, MTA-BME, VISZJV68
- Bálint Daróczy, SZTAKI-BME, VISZAMA01
- Judit Csima, BME, VISZM185
- Gábor Horváth, Péter Antal, BME, VIMMD294, VIMIA313
- Lukács András, ELTE, MM1C1AB6E
- Tim Kraska, Brown University, CS195
- Dan Potter, Carsten Binnig, Eli Upfal, Brown University, CS1951A
- Erik Sudderth, Brown University, CS142
- Joe Blitzstein, Hanspeter Pfister, Verena Kaynig-Fittkau, Harvard University, CS109
- Rajan Patel, Stanford University, STAT202
- Andrew Ng, John Duchi, Stanford University, CS229





Schedule of the semester

	<i>Monday midnight</i>	<i>Tuesday class</i>	<i>Friday class</i>
W1 (02/06)			
W2 (02/13)		HW1 out	
W3 (02/20)			
W4 (02/27)	HW1 deadline + TEAMS	HW2 out	
W5 (03/06)			PROJECT PLAN
W6 (03/13)	HW2 deadline	HW3 out	
W7 (03/20)			MIDTERM
SPRING BREAK		SPRING BREAK	SPRING BREAK
W8 (04/03)	HW3 deadline		GOOD FRIDAY
W9 (04/10)	MILESTONE 1		
W10 (04/17)		HW4 out	
W11 (04/24)			
W12 (05/01)	HW4 deadline		
W13 (05/08)	MILESTONE 2		
W14 (05/15)		FINAL	PROJECT presentations
W15 (05/22)		PROJECT presentations	

Artificial Neural Networks (ANN)

- It can be used for approximating any function
 - So it can solve both classification and regression problems
- The algorithm mimics the functioning of interconnected neurons



Machine learning with neural networks

The three main steps of machine learning (in general):

1. Hypothesis: non-linear hypothesis function
2. Cost function: Usual cost (error) function for classification/regression – misclassification error, logarithmic error, hinge, squared error, absolute error
3. Optimization: Gradient descent, stochastic gradient descent, other minimization methods
 - For neural network the cost function is usually non-convex → it is possible to get stuck in local minima

Feature extraction

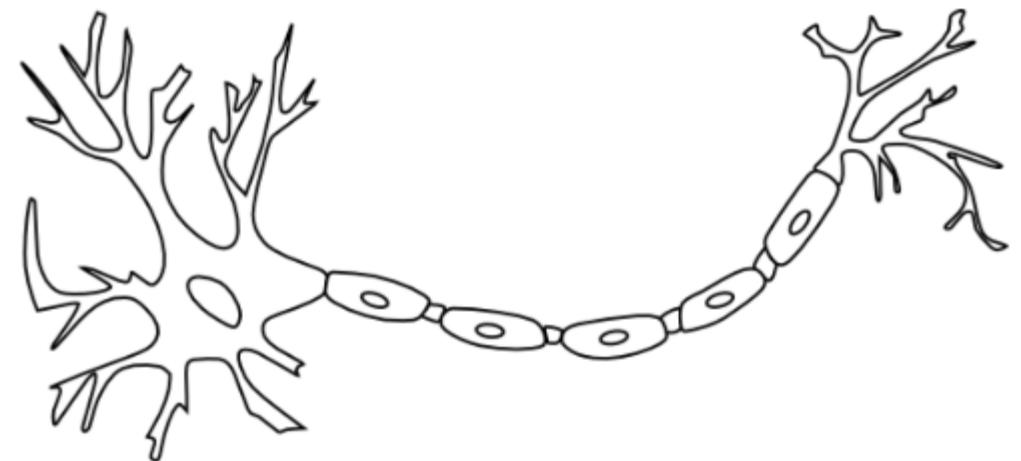
- Until now we defined the features by hand
 - Linear: x_i
 - Polynomial: $x_i^2, x_j^3 x_i^2$
 - Other non-linear functions (pl. radial function, logarithm etc.): $\log(x_i)$ etc.
 - Then the model learned linear hypothesis, the weights Θ (or w):
$$h_{\theta}(x) = \theta^T \phi(x) \quad \phi: \mathbb{R}^n \rightarrow \mathbb{R}^k$$
 - where ϕ is a function that transforms the variables, if $k > n$: transformation to higher dimension
- Goal: having an algorithm that is able to extract features automatically

Expert-defined features for hand-written digit recognition, e.g.:

- number of „on” pixels
- average of the vertical coordinates of the „on” pixels
- variance of the horizontal coordinates
- correlation between the horizontal and vertical positions of „on” pixels
- ...

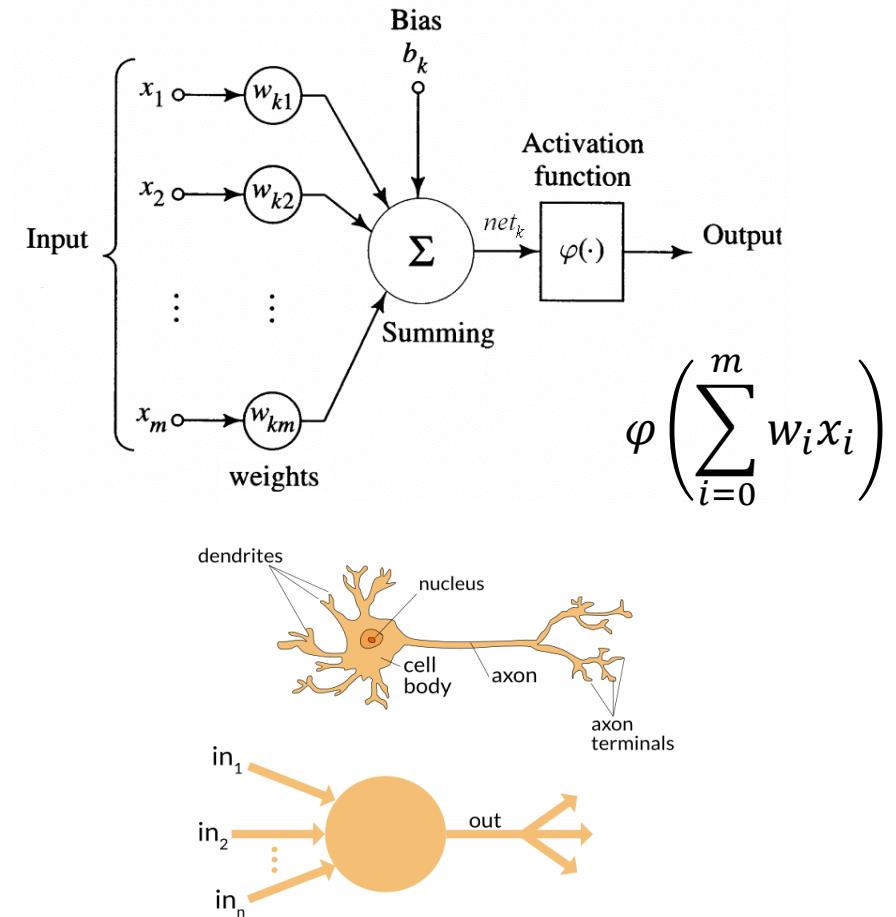
Biological motivation: neuron

- Neurons receive signals via dendrites (usually more dendrites), if the intensity of the stimuli is above a threshold then the neuron generate and propagate an electrical signal (an action potential) and the potential travels along the axon to its connections
 - There are inputs (signals via dendrites)
 - The cell body „process” the inputs
 - The output is the action potential that is transmitted along the axon



Perceptron

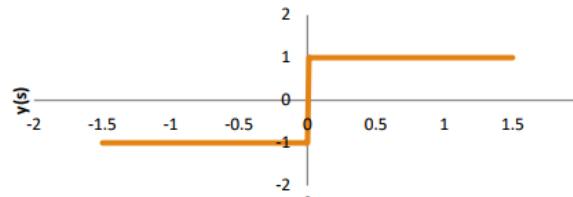
- Input nodes: where the input arrives
 - They correspond to the attributes
 - Special input node is the constant ($x_0 = 1$, bias)
- Weights of the inputs
 - These weights should be learned by the algorithm
- Summing
 - Summation of the weighted inputs
- Activation
 - We apply the activation function to the weighted sum of the inputs
- Output
 - The result of the activation function
 - It corresponds to the class label / target variable



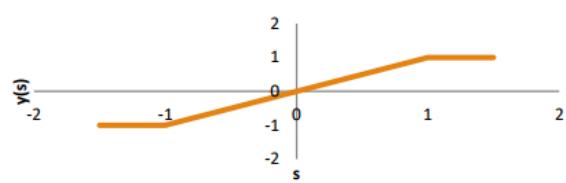
Activation functions

- Sign function
- Piecewise linear function
- Tangent hyperbolic ($K=2$)

$$y = \begin{cases} 1, & s > 0 \\ -1, & s \leq 0 \end{cases}$$



$$y = \begin{cases} 1, & s > 1 \\ s, & -1 \leq s \leq 1 \\ -1, & s \leq -1 \end{cases}$$



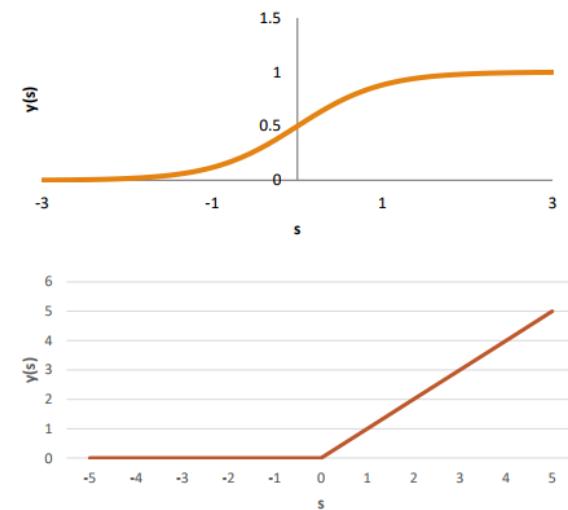
$$y = \frac{1 - e^{-Ks}}{1 + e^{-Ks}}; K > 0$$



- Sigmoid function
- Hinge

$$y = \frac{1}{1 + e^{-Ks}}; K > 0$$

$$y = \max(0, s)$$

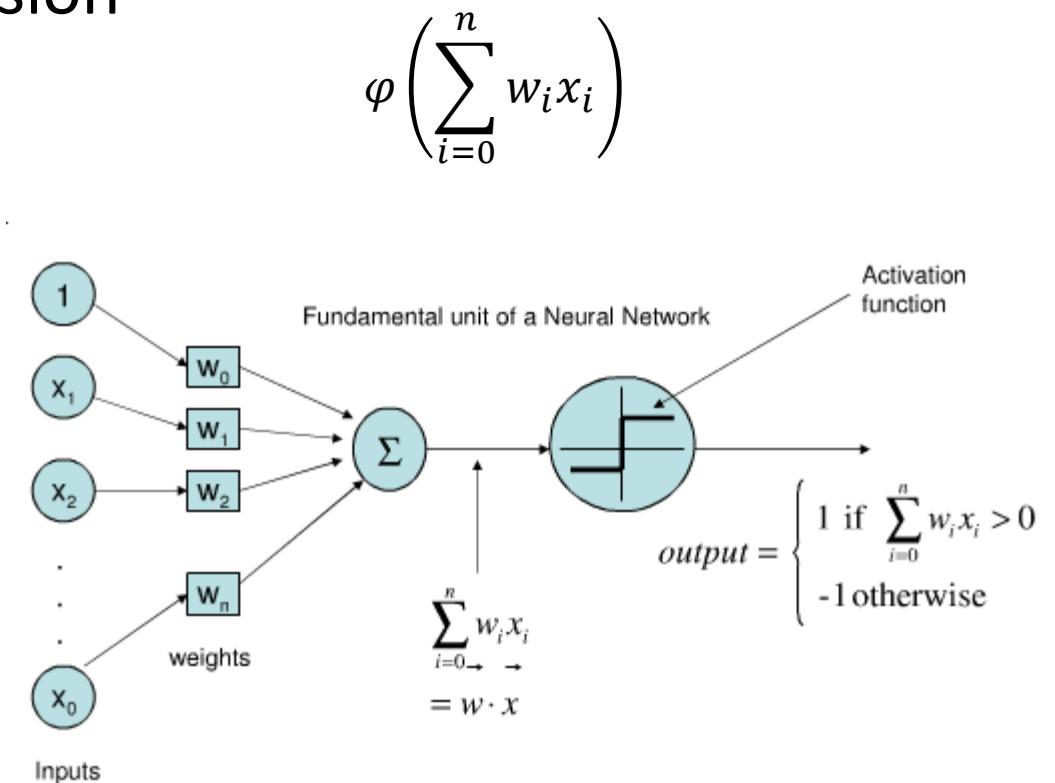


Many other possible activation function

Identity		$f(x) = x$	$f'(x) = 1$
Binary step		$f(x) = \begin{cases} 0 & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$	$f'(x) = \begin{cases} 0 & \text{for } x \neq 0 \\ ? & \text{for } x = 0 \end{cases}$
Logistic (a.k.a Soft step)		$f(x) = \frac{1}{1 + e^{-x}}$	$f'(x) = f(x)(1 - f(x))$
TanH		$f(x) = \tanh(x) = \frac{2}{1 + e^{-2x}} - 1$	$f'(x) = 1 - f(x)^2$
ArcTan		$f(x) = \tan^{-1}(x)$	$f'(x) = \frac{1}{x^2 + 1}$
Rectified Linear		$f(x) = \begin{cases} 0 & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$	$f'(x) = \begin{cases} 0 & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$
SoftPlus		$f(x) = \log_e(1 + e^x)$	$f'(x) = \frac{1}{1 + e^{-x}}$
Bent identity		$f(x) = \frac{\sqrt{x^2 + 1} - 1}{2} + x$	$f'(x) = \frac{x}{2\sqrt{x^2 + 1}} + 1$
SoftExponential		$f(\alpha, x) = \begin{cases} -\frac{\log_e(1 - \alpha(x + \alpha))}{\alpha} & \text{for } \alpha < 0 \\ x & \text{for } \alpha = 0 \\ \frac{e^{\alpha x} - 1}{\alpha} + \alpha & \text{for } \alpha > 0 \end{cases}$	$f'(\alpha, x) = \begin{cases} \frac{1}{1 - \alpha(x + \alpha)} & \text{for } \alpha < 0 \\ e^{\alpha x} & \text{for } \alpha \geq 0 \end{cases}$
Sinusoid		$f(x) = \sin(x)$	$f'(x) = \cos(x)$
Sinc		$f(x) = \begin{cases} 1 & \text{for } x = 0 \\ \frac{\sin(x)}{x} & \text{for } x \neq 0 \end{cases}$	$f'(x) = \begin{cases} 0 & \text{for } x = 0 \\ \frac{\cos(x)}{x} - \frac{\sin(x)}{x^2} & \text{for } x \neq 0 \end{cases}$
Gaussian		$f(x) = e^{-x^2}$	$f'(x) = -2xe^{-x^2}$

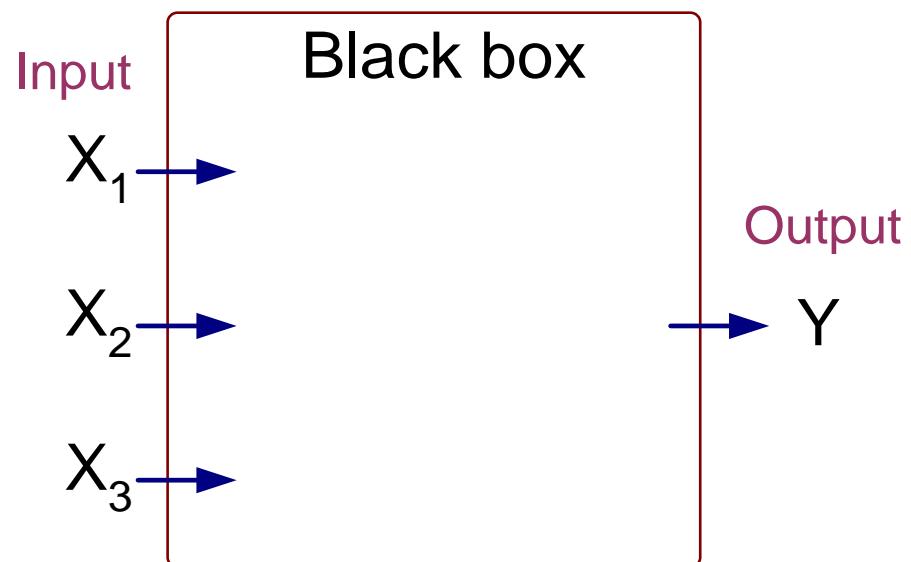
The connection of perceptron to other algorithms

- If the activation function is the sigmoid function then the perceptron algorithm agrees with logistic regression
 - Providing that the cost function is the logarithmic loss
- If the activation function is the identity, then it is the same as linear regression
- Perceptron (with usual activation functions) is a linear algorithm, thus it creates linear decision boundary



Training perceptron to learn logical function

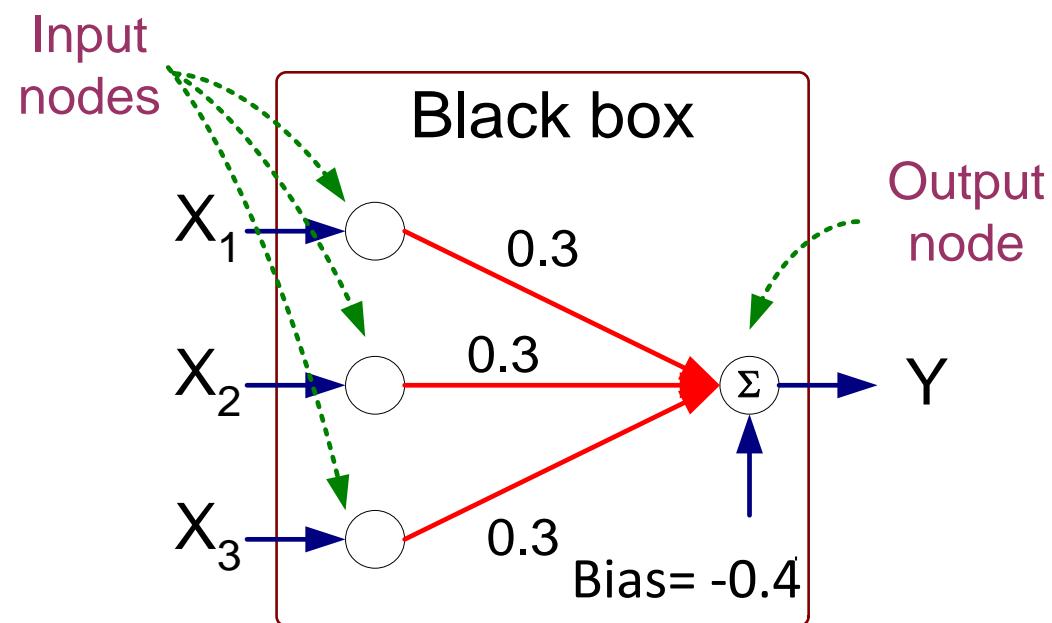
X_1	X_2	X_3	Y
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1
0	0	1	0
0	1	0	0
0	1	1	1
0	0	0	0



- The output (Y) is 1, if and only if at least two input parameters are 1

Training perceptron to learn logical function II.

X_1	X_2	X_3	Y
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1
0	0	1	0
0	1	0	0
0	1	1	1
0	0	0	0



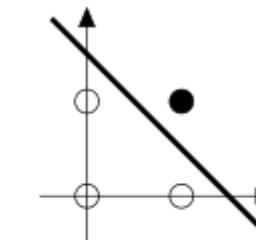
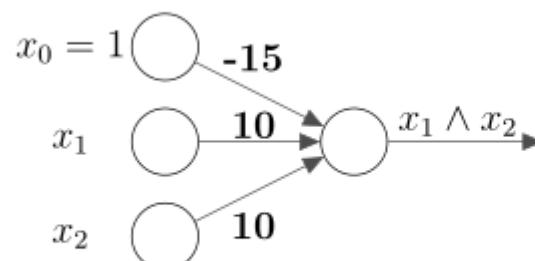
$$Y = I(0.3X_1 + 0.3X_2 + 0.3X_3 - 0.4 > 0)$$

$$\text{where } I(z) = \begin{cases} 1 & \text{if } z \text{ true} \\ 0 & \text{otherwise} \end{cases}$$

Examples – other logical functions

- Similarly to the previous slide, let's find the perceptron that is able to represent AND and OR functions for two input variables
 - x_1 AND x_2
 - x_1 OR x_2
- It is enough to find the equation of the separating line

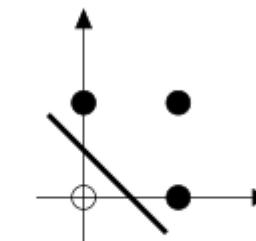
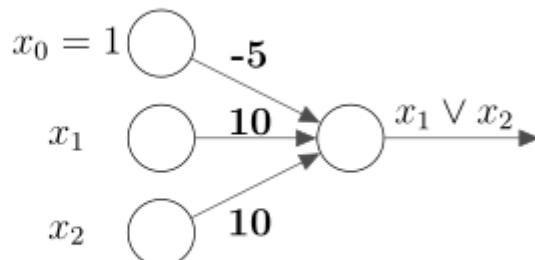
AND



$$x_2 = -x_1 + 1,5$$

$$-1,5x_0 + x_1 + x_2 = 0$$

OR



$$x_2 = -x_1 + 0,5$$

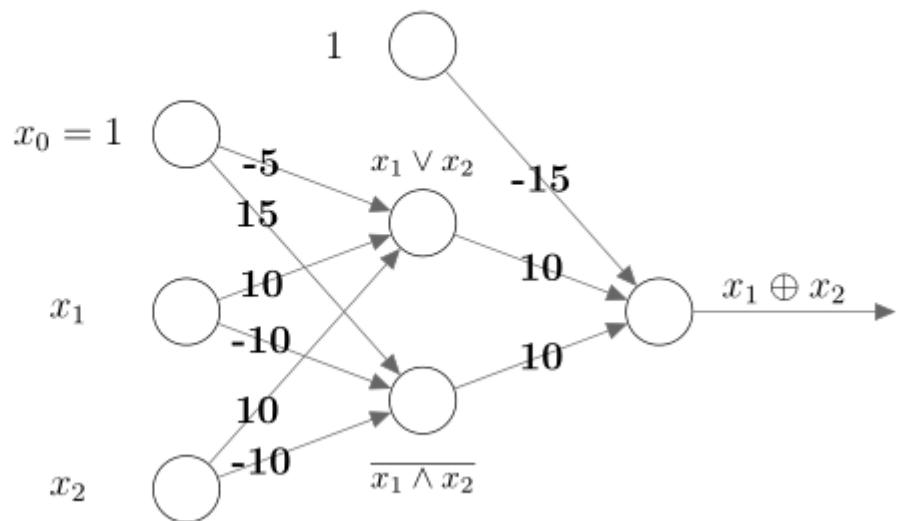
$$-0,5x_0 + x_1 + x_2 = 0$$

Example – XOR function

- We have seen that XOR is not linearly separable
 - One neuron is not able to represent it
 - Using AND and OR functions, it can be represented
 - Let us combine AND and OR to build a neural network by combining the neurons

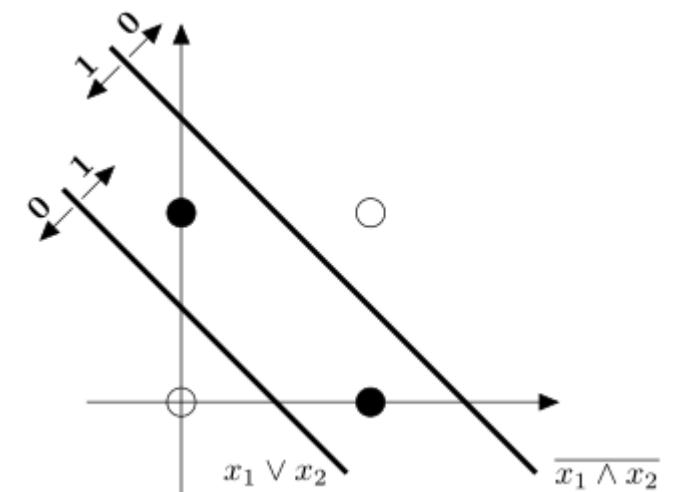
XOR

$$x_1 \oplus x_2 = (x_1 \vee x_2) \wedge \overline{(x_1 \wedge x_2)}$$



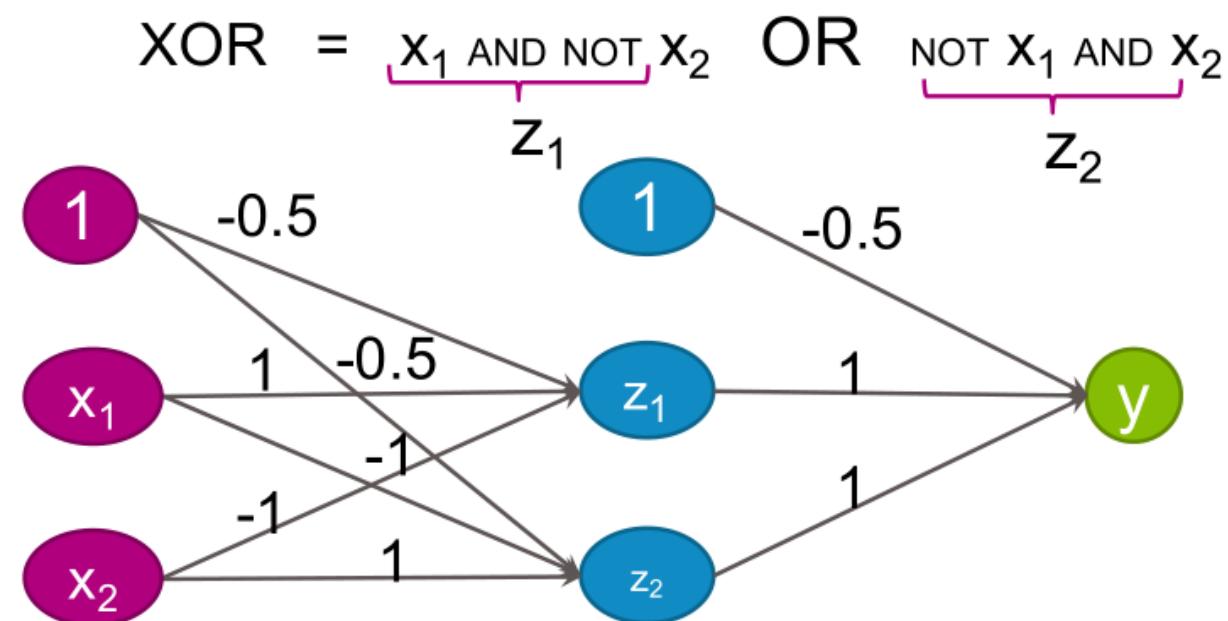
Example—XOR function II. – feature extraction

- One neuron was not enough but by using a so-called hidden layer we were able to represent XOR function
- The hidden layer can be considered as new features ($x_1 \vee x_2$, $\overline{x_1 \wedge x_2}$) extracted from original features (x_1, x_2)
 - Neural networks with hidden layer(s) are able to extract new features automatically



Example – XOR function III.

- XOR function can be represented by other neural networks as well (with the help of a different logical identity)
- The principle is the same



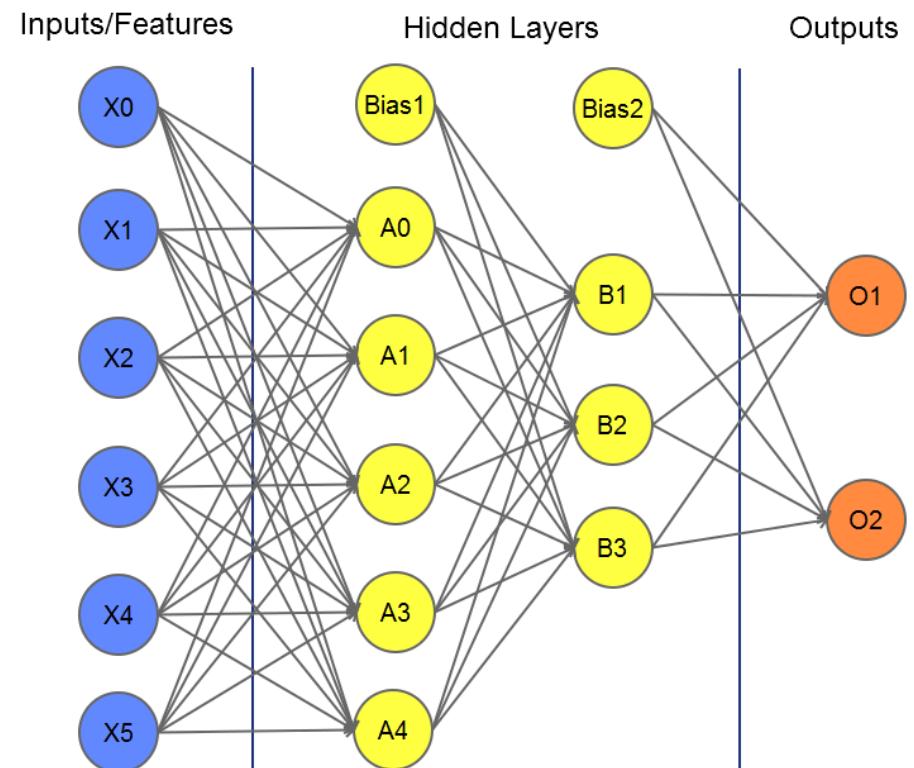
Problem

Represent the following logical functions with a perceptron or show that it is not possible to do so. In the latter case, construct a neural network with one hidden layer.

- a) A AND B AND C
- b) (A XOR B) AND (A OR B)

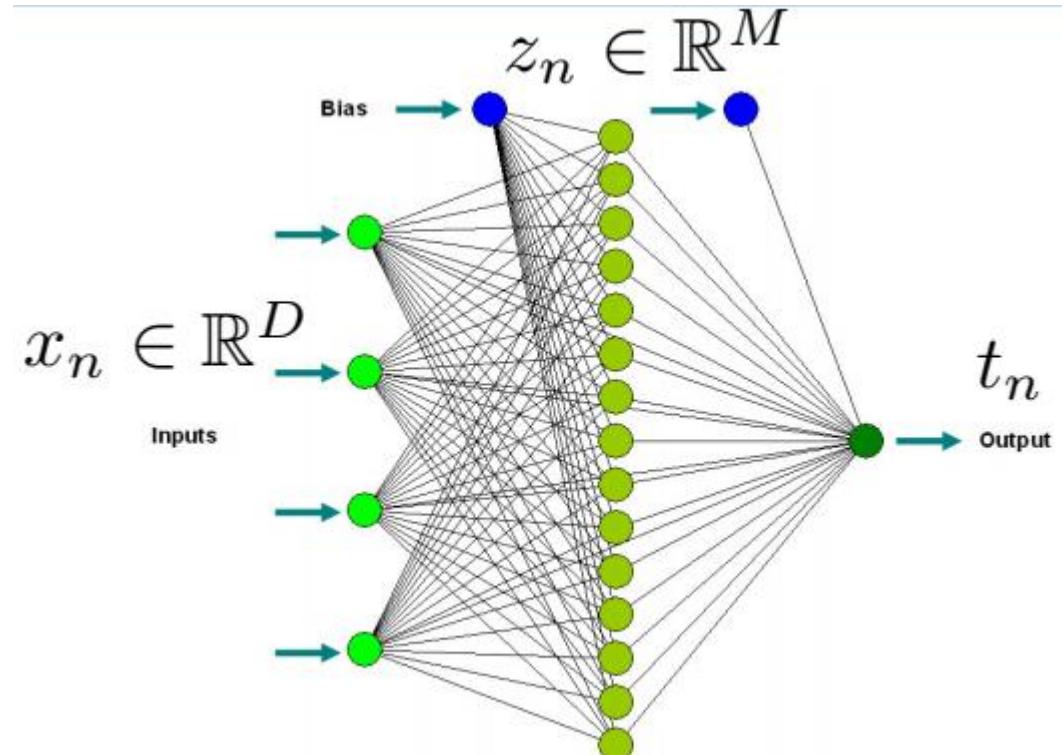
Multi-layer neural networks

- Having more perceptrons (neurons) in multiple layers
- First layer: input nodes
- Last layer: output nodes (one or more)
- Hidden layers: perceptrons
- Perceptrons of neighboring layers are connected
 - The connectedness depends on the architecture of the network (normally: fully connected)
- All perceptrons receive their inputs from the previous layers and propagate their outputs to the next layer
- Weights correspond to the links (edges)
 - These weights should be adjusted by the learning procedure to minimize the cost function



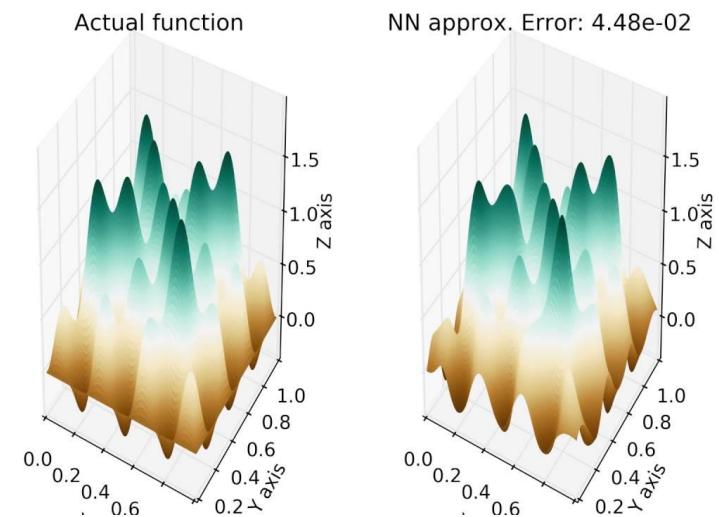
Feed-forward neural network

- Assume that the network is already trained (the weights are set)
- A record arrives to the input nodes and it propagates through the network in a forward direction (from the left to the right) on a layer-by-layer basis
 - The layers process the data
 - It can be considered as extracting new features in each layer
- Finally we obtain the result on the output node(s)



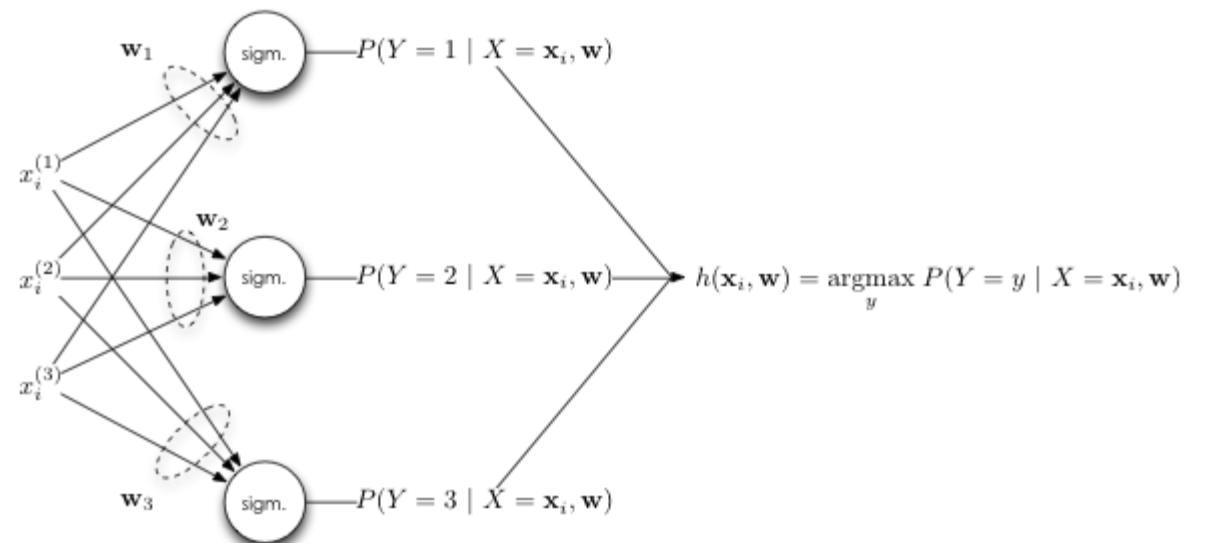
Universal approximation theorem

- A feed-forward network with a single hidden layer containing a finite number of neurons can approximate continuous functions in compact subsets of R^n , under mild assumptions on the activation function
 - It is important to note that the theorem states that simple neural networks can represent a wide variety of functions, however, it does not touch upon the algorithmic learnability of the parameters
 - Non-constructive



Multi-class problem

- If there are more possible class labels (number of labels c)
- There are more output nodes: the same number as the number of possible labels (c)
- An input is assigned to label i if the i th output is 1 and the others are 0
 - if the possible output values are continuous then we assign the label with largest value on the corresponding output node
- E.g. for character recognition, the number of outputs corresponds to the number of characters



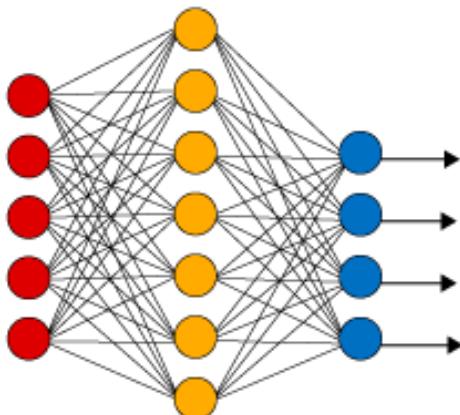
Architecture of the network

- Number of input nodes arises from the nature of the problem
 - Number of attributes, e.g. for character recognition it is the number of pixels
- Number of output nodes arises from the nature of the problem
- The other parameters can be set freely
 - Number of hidden layers
 - Number of neurons in each layer
- The more complex the network is, the more possible that the network will overfit the data
 - Sometimes it is enough to have a simpler network

Deep learning

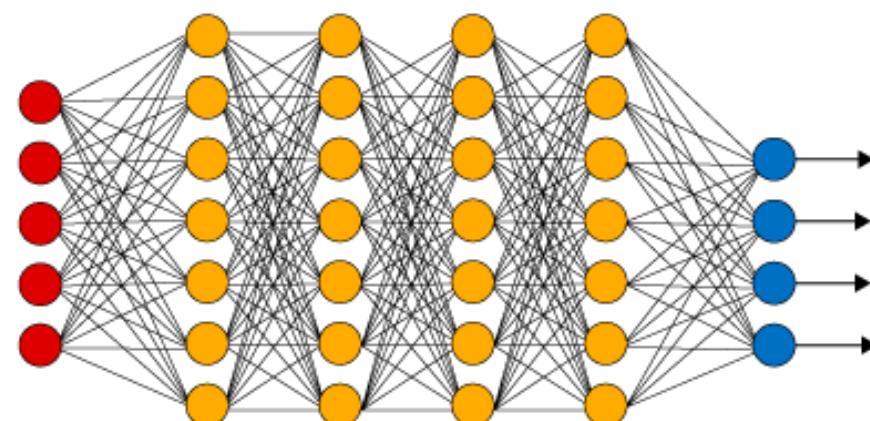
- Deep learning refers to machine learning with deep neural networks, i.e. neural networks with many hidden layers

Simple Neural Network



● Input Layer

Deep Learning Neural Network

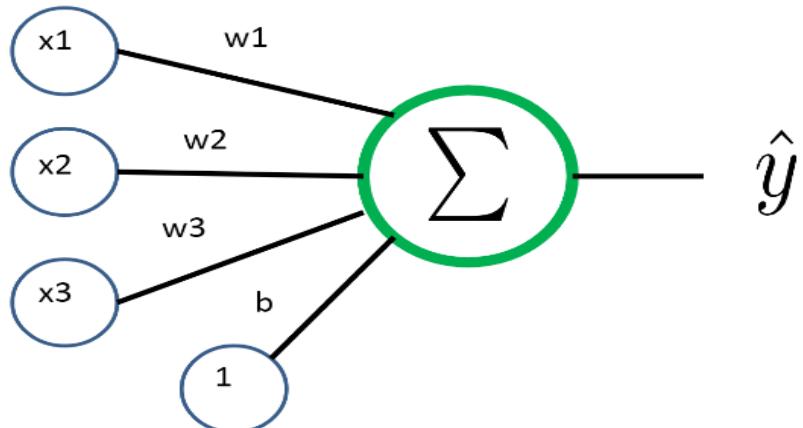


● Hidden Layer

● Output Layer

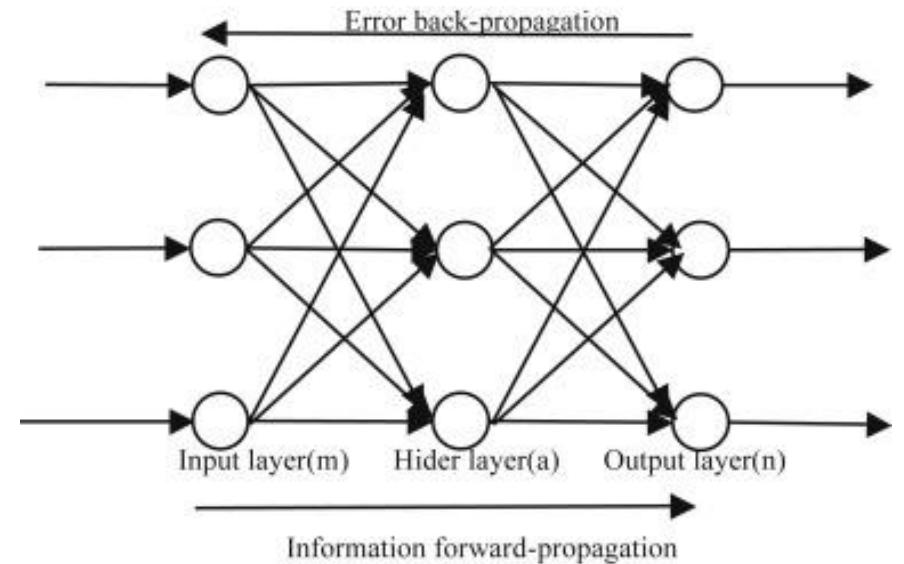
Training a perceptron

- Perceptron is a simple linear model, the training procedure is similar to the usual one:
 - We choose a cost function (MSE, logistic cost)
 - The cost function is minimized (gradient descent, stochastic gradient descent or other optimization method)



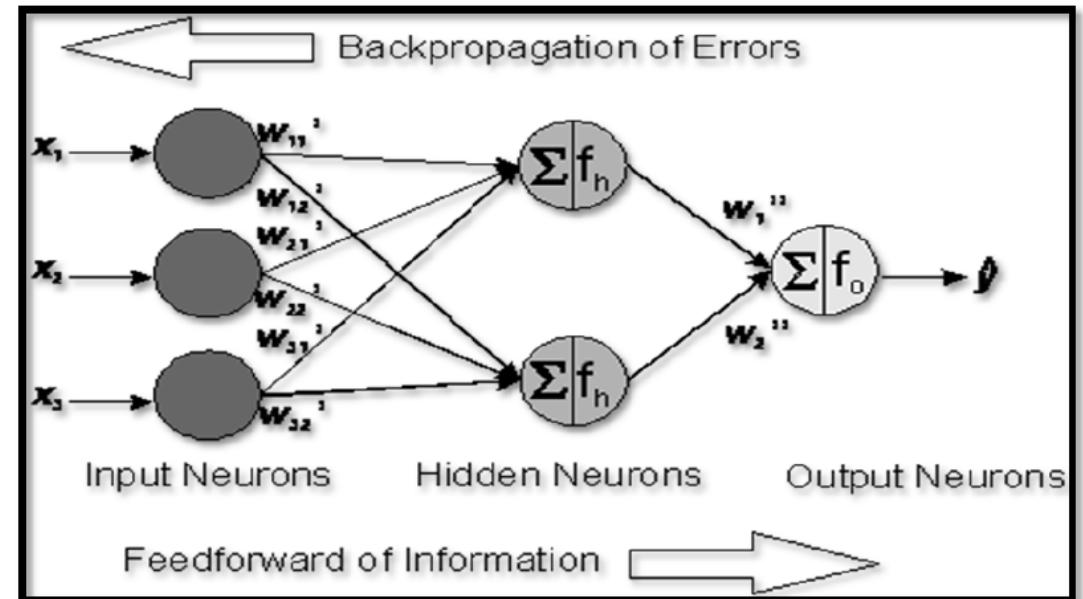
Forward propagation for ANNs

1. The training record arrives on the input nodes
2. Using the current weights on the edges, the record propagates through the network (from the left to the right)
3. We calculate the result on the output node(s) and we compare the output(s) with the actual target value
4. We calculate the cost (e.g. squared error, logarithmic error) based on the difference between the output and the actual target value



Backpropagation

1. Initializing the weights
2. Choosing a (random) record from the training data
3. Using the forward propagation algorithm, the output and the error is calculated
4. Calculating the gradient with respect to the weights
 - See next slides!
5. Updating the weights according to gradient method
6. Start the procedure again with another (random) record and repeat the procedure „until convergence”



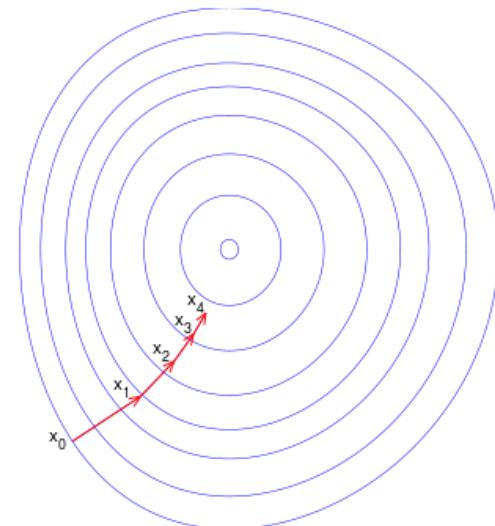
Calculating the gradient – using chain rule

- Reminder: how it worked for simple linear regression:

$$F = (\hat{y} - y)^2$$

$$\hat{y} = \sum_{j=1}^m w_j x_j + w_0$$

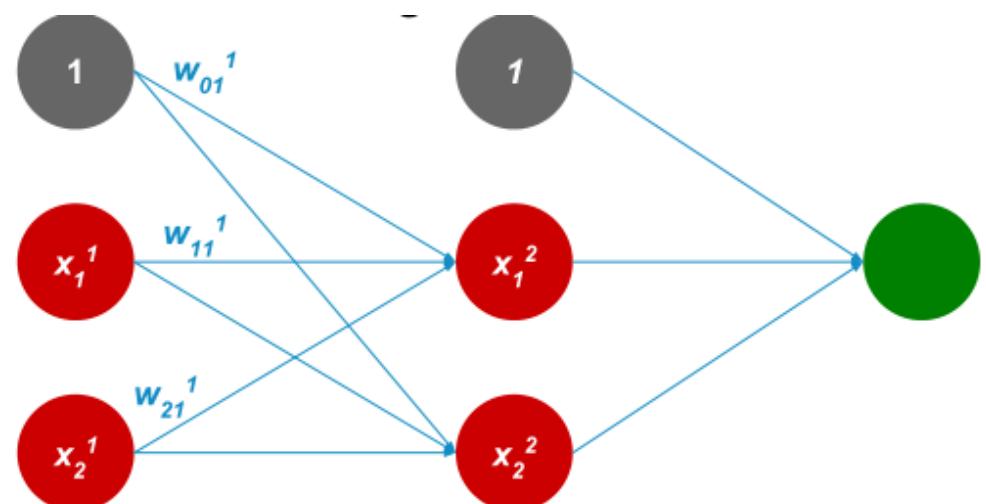
$$\frac{\partial F}{\partial w_j} = \frac{\partial F}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial w_j}$$



Training a multi-layer neural network

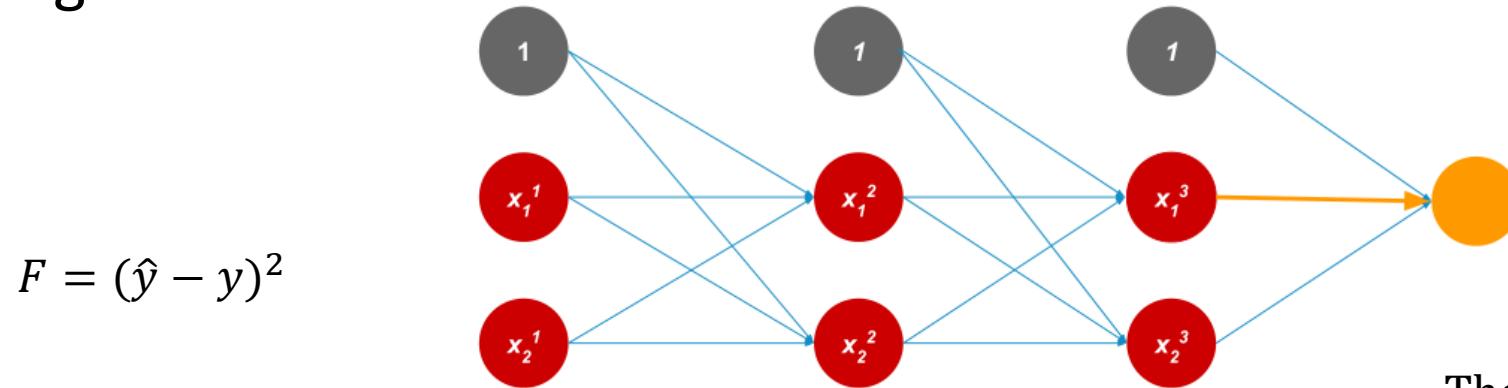
- The output of a layer is the input of the following layer
- The weights on the edges should be learned by the model
- Activation function: g
- Weights on edges: w
- The output of the neurons (nodes) are denoted by x and calculated as:

$$x_i^{l+1} = g \left(\sum_{j=1}^n w_{ji}^l x_j^l + w_0^l \right)$$



Calculating the gradient for a multi-layer neural network

- Using chain-rule:



$$F = (\hat{y} - y)^2$$

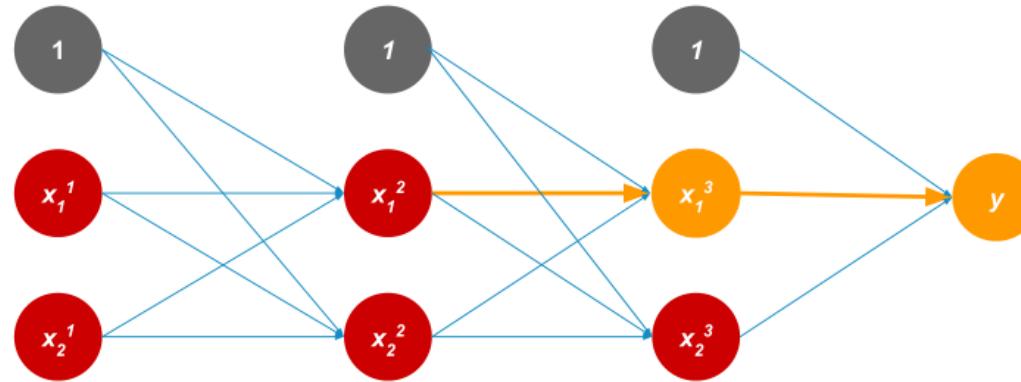
$$\frac{\partial F}{\partial w_{11}^3} = \frac{\partial F}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial w_{11}^3} = 2 \cdot (\hat{y} - y) \cdot g'(w_{01}^3 + w_{11}^3 x_1^3 + w_{21}^3 x_2^3) \cdot x_1^3$$

where g is the activation function, w_{jk}^l is the weight on the edge from x_j^l to x_k^{l+1}

The output:
 $\hat{y} = g(w_{01}^3 + w_{11}^3 x_1^3 + w_{21}^3 x_2^3)$

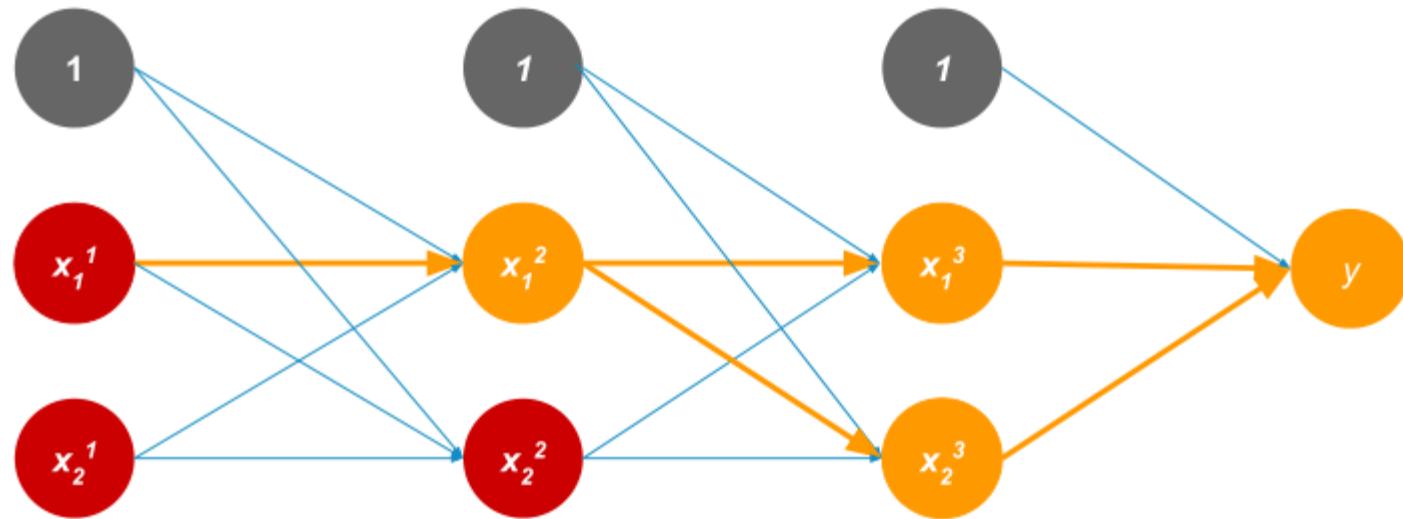
Calculating the gradient for a multi-layer neural network II.

- For „deeper” edges, i.e., for edges that do not effect the output directly, the derivatives calculated recursively



$$\frac{\partial F}{\partial w_{11}^2} = \frac{\partial F}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial x_1^3} \frac{\partial x_1^3}{\partial w_{11}^2}$$

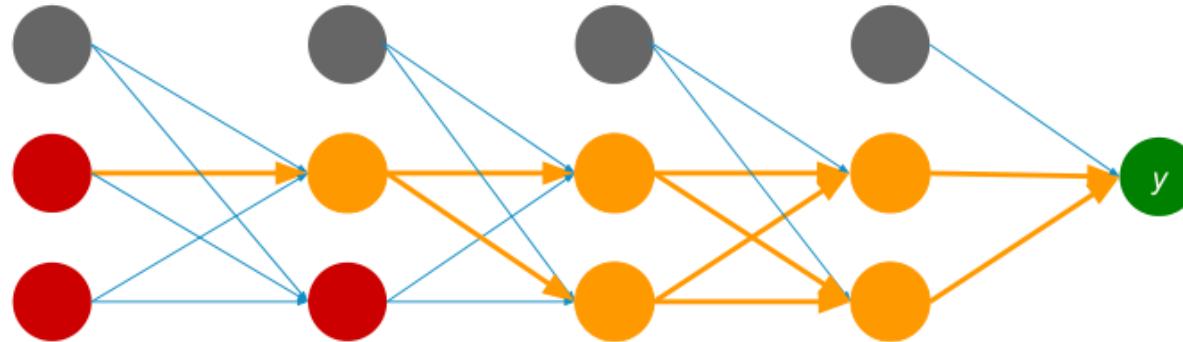
Calculating the gradient for a multi-layer neural network III.



$$\frac{\partial F}{\partial w_{11}^1} = \frac{\partial F}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial x_1^3} \frac{\partial x_1^3}{\partial x_1^2} \frac{\partial x_1^2}{\partial w_{11}^1} + \frac{\partial F}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial x_2^3} \frac{\partial x_2^3}{\partial x_2^2} \frac{\partial x_2^2}{\partial w_{11}^1}$$

Calculating the gradient for a multi-layer neural network IV.

- Generally, we have to sum up the derivatives for all the possible paths that starts from the end node of the examined edge and terminated in the output node



$$\frac{\partial F}{\partial w_{jk}^l} = \sum_{mn\dots q} \frac{\partial F}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial x_m^L} \frac{\partial x_m^L}{\partial x_n^{L-1}} \dots \frac{\partial x_q^{l+2}}{\partial x_k^{l+1}} \frac{\partial x_k^{l+1}}{\partial w_{jk}^l}$$

- Where: L is the number of layers, m, n, \dots, q run through the possible values (i.e. number the possible values are the number of neurons in each layer)

Problem

Consider a chain of two neurons. The input of the first neuron is x_1 and the output is $y_1 = ax_1 + b$. The input of the other neuron is x_2 and the output is $y_2 = cx_2 + d$ (the activation function is the identity function in both cases). Connect the two neurons so that the second neuron's input is y_1 , i.e. $x_2=y_1$.

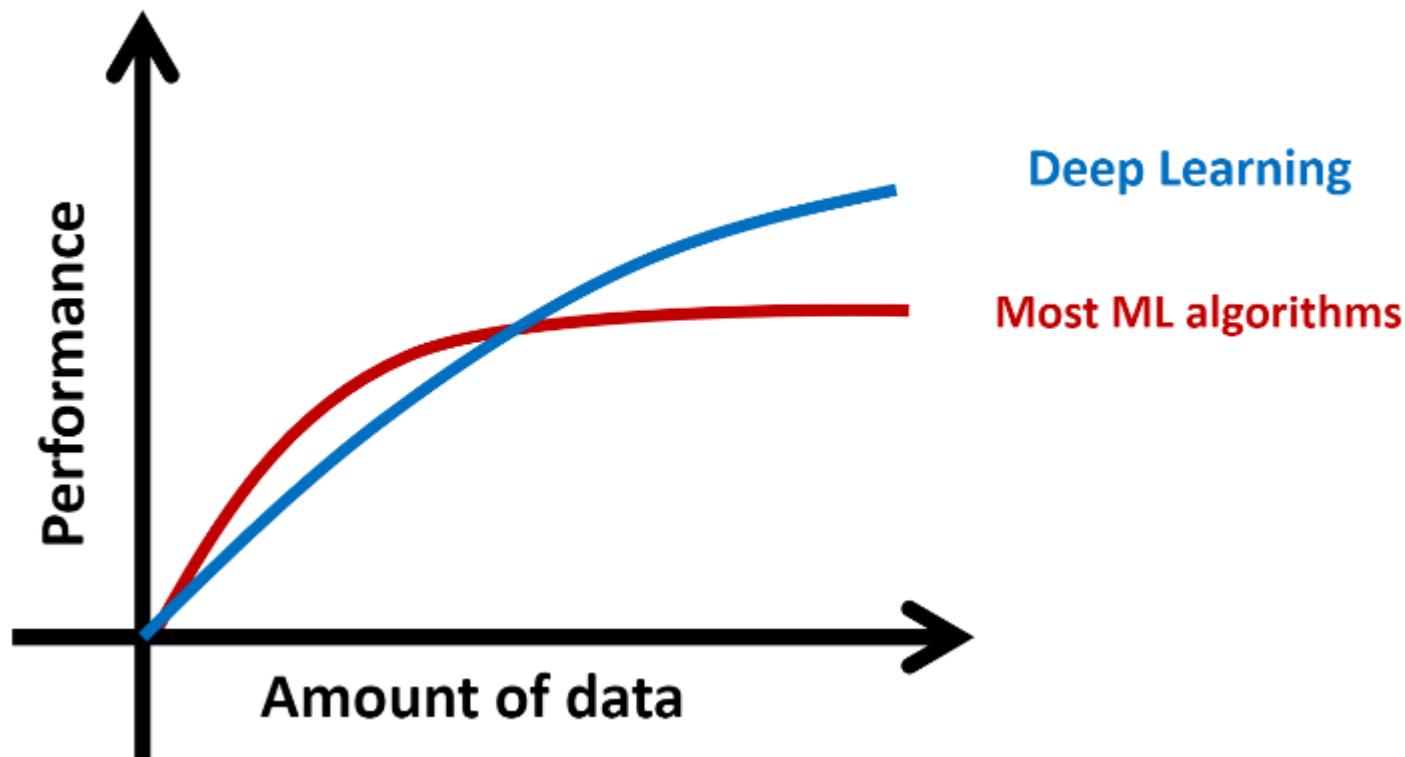
- a) Draw this neural network!
- b) Give the final output y_2 as a function of x_1 !
- c) Let the input of the ANN be x and the output be y . Using the gradient descent method, show how the weights (a, b, c, d) are updated after one training step if the squared error is minimized.

Advantages/disadvantages of neural networks

- They make it possible to extract features automatically
- They mean significant advancement in:
 - Image recognition
 - Speech recognition
 - Text mining
- They can be parallelized
- Potential room for improvement
 - Increasing amount of data
 - Faster computers

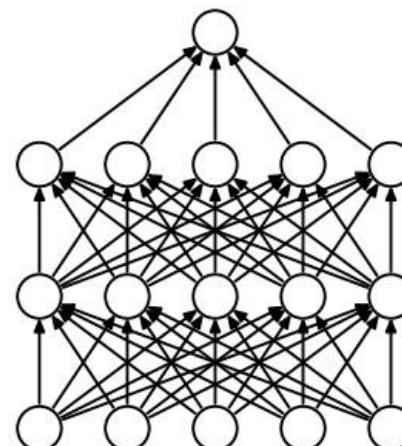
- For good performance (very) big data is necessary
- Computationally expensive
- It is difficult to set the (hyper)parameters, i.e. choose the proper architecture
 - Number of hidden layers, number of neurons, number of edges
- The result is not easily interpretable
 - Difficult to understand what the reason of the prediction is

Deep learning and amount of data

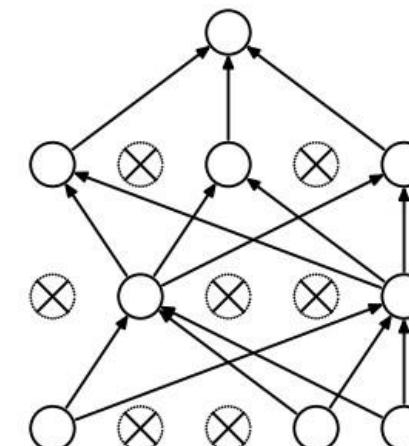


Avoiding overfitting neural networks

- Adding regularization term to cost function
 - Usually the sum of the squared weights (as before)
- Reducing the number of hidden layers
- Avoiding too complex architectures
- Dropout algorithm
(dropping out certain nodes from the network)



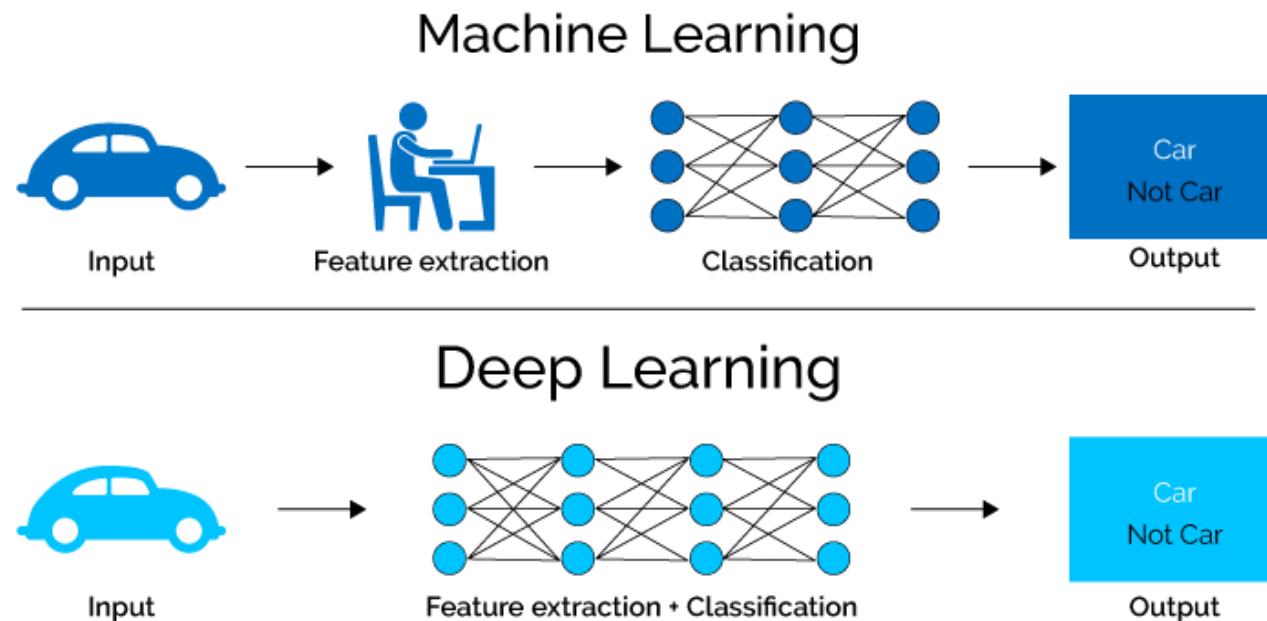
(a) Standard Neural Net



(b) After applying dropout.

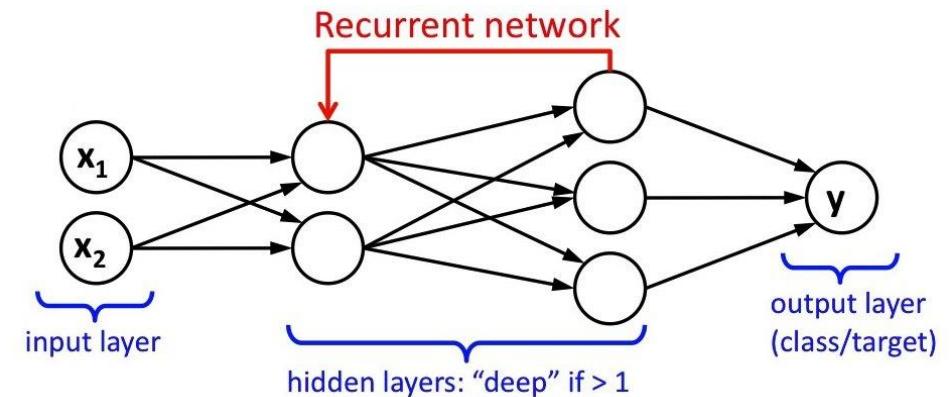
Neural network as a feature extractor

- We can think of the values on the hidden layers as new extracted attributes
- It is also possible to use these new features as inputs of another machine learning algorithm

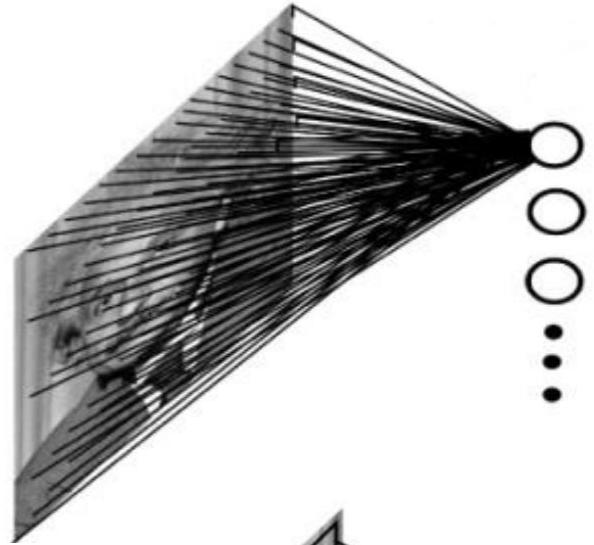


Types of neural networks

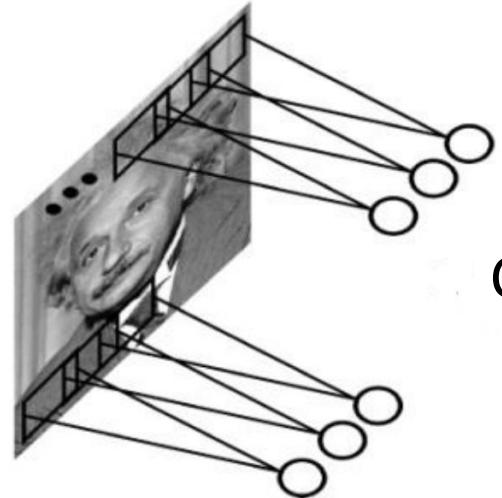
- Fully connected feed-forward neural networks
 - The simplest but not the most popular architecture
 - There are a lot of parameters (edge weights) that we have to optimize for
 - Danger of overfitting
- Recurrent Neural Network – RNN
 - For sequential data it is able to learn the temporal nature of the data
 - E.g. time series prediction, audio mining
 - There is a directed cycle in the network
 - RNN is able to use its own inner memory
- Convolutional Neural Networks – CNN
 - Not all of the connections are drawn, the „proximity”, of the data is also taken into consideration
 - Mainly used for image classification



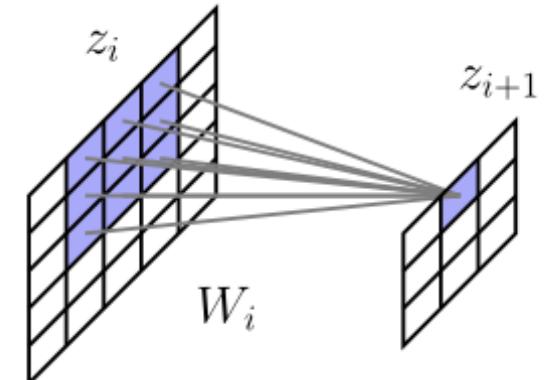
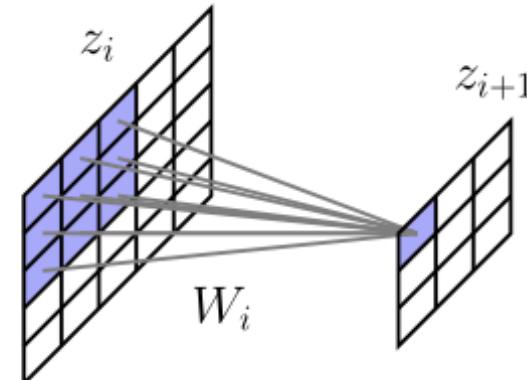
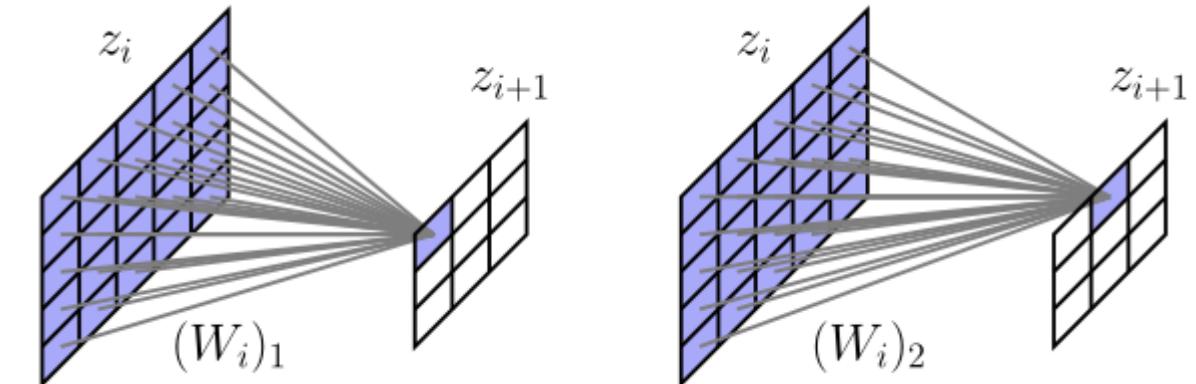
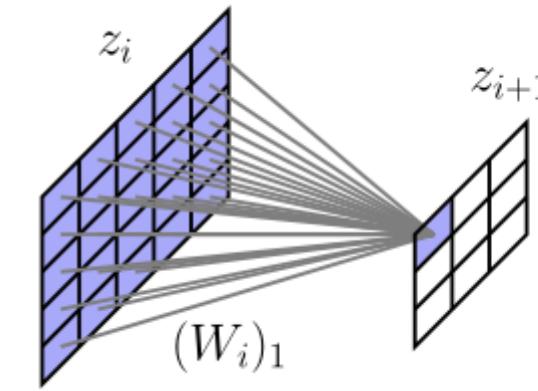
Fully connected vs. convolutional neural networks



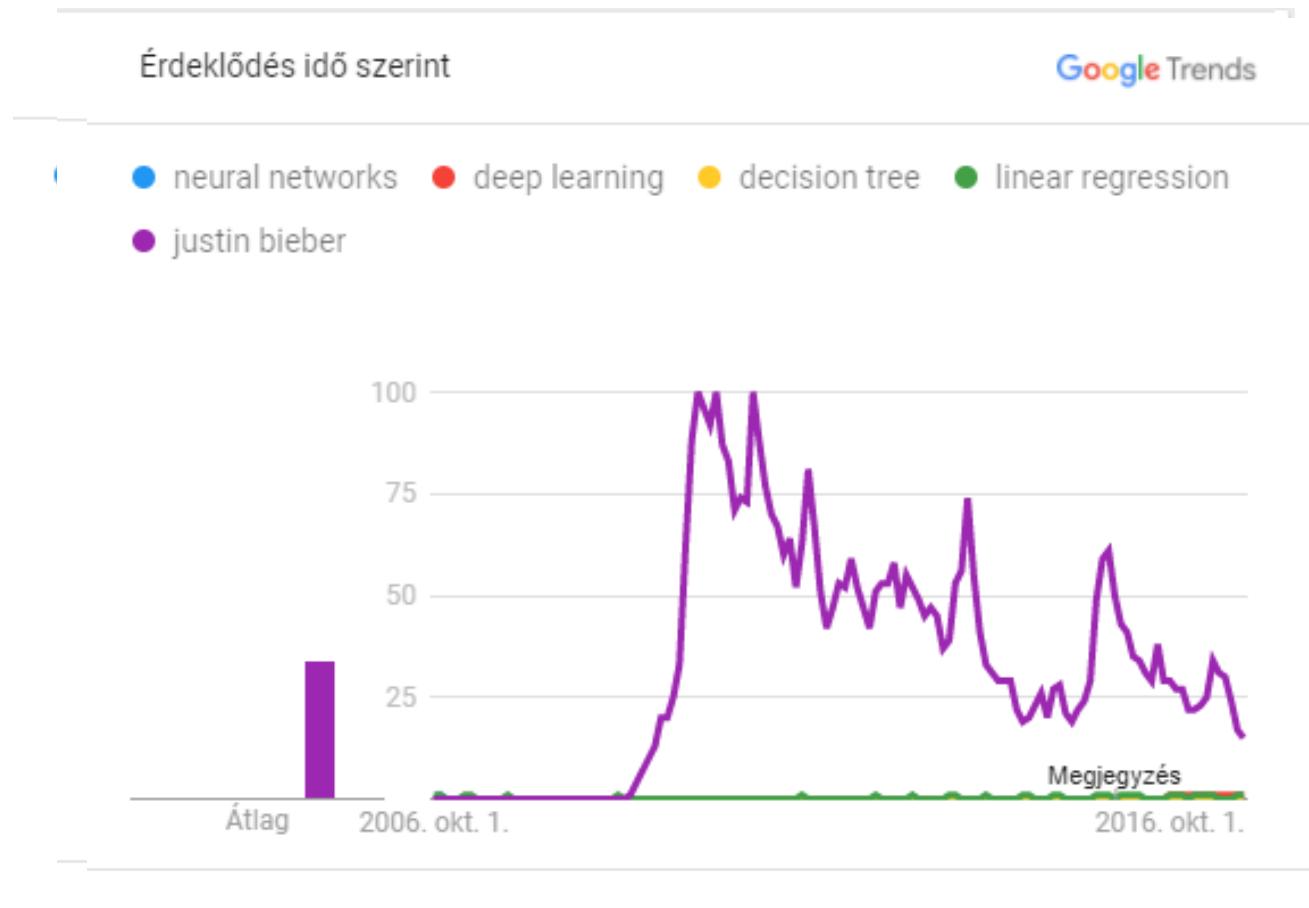
Fully
connected



Convolutional



Change in search interest

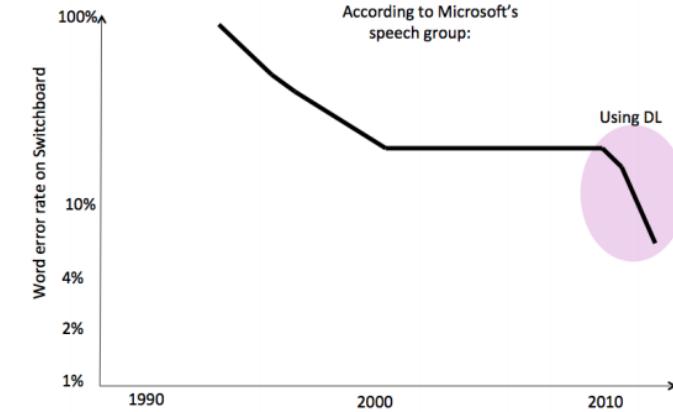


For which problems is it not so practical to use deep learning?

- If we have „traditional” attributes (features that can be easily interpreted by humans), then it is better to start with „traditional” algorithms
 - If we are not satisfied with the results, we can use deep learning to possibly achieve 1-2% performance improvement

For which problem is it practical to use deep learning?

- If the data have attributes that are not easily interpretable (e.g. text, image, audio files) then using deep learning has great potential
 - Image recognition
 - Text mining
 - Audio mining
 - Language processing
- It is practical to use an already trained network for the given problem instead of experimenting with the proper architecture

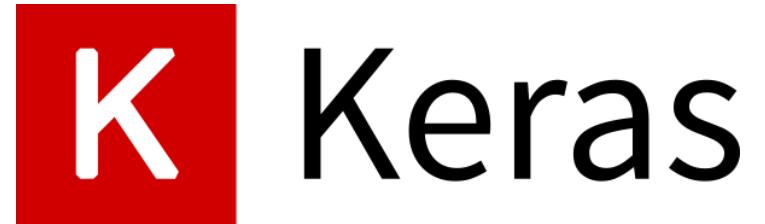


Try it out: <http://deeplearning.cs.toronto.edu>



Softwares for deep learning

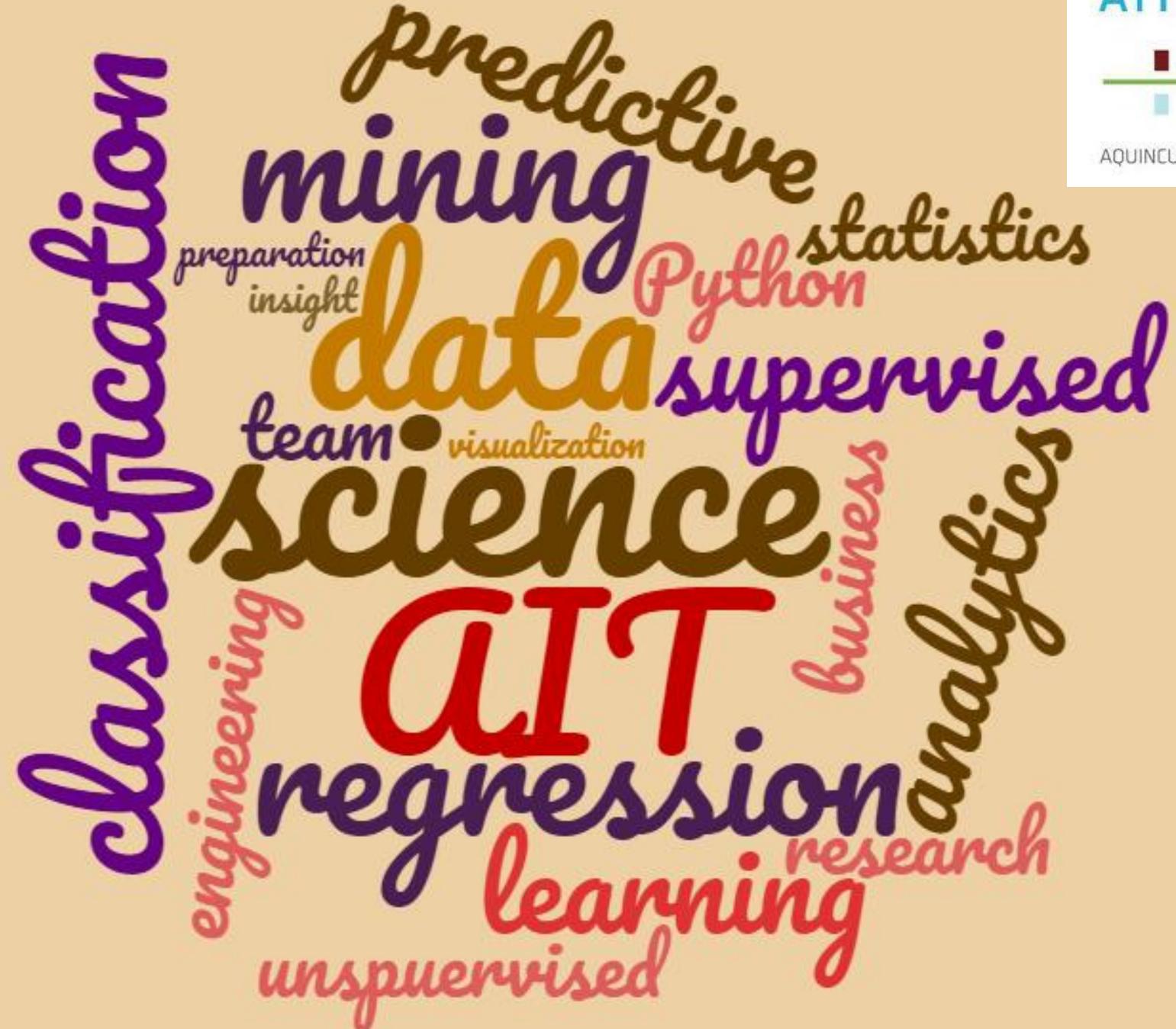
- Keras
 - Based on Python, Theano and TensorFlow backend, easily readable code, originates from Google
- TensorFlow
- Working with Python (and with some other programming languages as well)



Acknowledgement

- András Benczúr, Róbert Pálovics, SZTAKI-AIT, DM1-2
- Krisztián Buza, MTA-BME, VISZJV68
- Bálint Daróczy, SZTAKI-BME, VISZAMA01
- Judit Csima, BME, VISZM185
- Gábor Horváth, Péter Antal, BME, VIMMD294, VIMIA313
- Lukács András, ELTE, MM1C1AB6E
- Tim Kraska, Brown University, CS195
- Dan Potter, Carsten Binnig, Eli Upfal, Brown University, CS1951A
- Erik Sudderth, Brown University, CS142
- Joe Blitzstein, Hanspeter Pfister, Verena Kaynig-Fittkau, Harvard University, CS109
- Rajan Patel, Stanford University, STAT202
- Andrew Ng, John Duchi, Stanford University, CS229





Schedule of the semester

	<i>Monday midnight</i>	<i>Tuesday class</i>	<i>Friday class</i>
W1 (02/06)			
W2 (02/13)		HW1 out	
W3 (02/20)			
W4 (02/27)	HW1 deadline + TEAMS	HW2 out	
W5 (03/06)			PROJECT PLAN
W6 (03/13)	HW2 deadline	HW3 out	
W7 (03/20)			MIDTERM
SPRING BREAK		SPRING BREAK	SPRING BREAK
W8 (04/03)	HW3 deadline		GOOD FRIDAY
W9 (04/10)	MILESTONE 1		
W10 (04/17)		HW4 out	
W11 (04/24)			
W12 (05/01)	HW4 deadline		
W13 (05/08)	MILESTONE 2		
W14 (05/15)		FINAL	PROJECT presentations
W15 (05/22)		PROJECT presentations	

Milestone 2

- Three-page long report including:
 - Reviewing the related works
 - Data understanding and data preparation steps
 - Data analysis steps, implementing some models and evaluating them



Galton and the weight of an ox

- The story of Sir Francis Galton (1906)
 - English statistician, polymath
 - He visited a country fair where 800 people participated in a contest to estimate the weight of an ox
 - To his surprise, the median guess, 1207 pounds was accurate within 1% of the true weight of 1198 pounds
 - None of the experts were as accurate as the crowd itself
 - Several similar experiments have been conducted with similar results



The wisdom of crowds

- Criteria required to form a wise crowd (collective intelligence)
 - Diversity of opinion
 - Independence
 - Decentralization, specialization
 - Appropriate aggregation



WIKIPEDIA



Linux

NEW YORK TIMES BUSINESS BESTSELLER
"An entertaining and thought-provoking ... The Tipping Point by Malcolm Gladwell, ... The Wisdom of Crowds ranges far and wide." —The Boston Globe

THE WISDOM
OF CROWDS
JAMES
SUROWIECKI

WITH A NEW AFTERWORD BY THE AUTHOR



Ensemble methods in machine learning

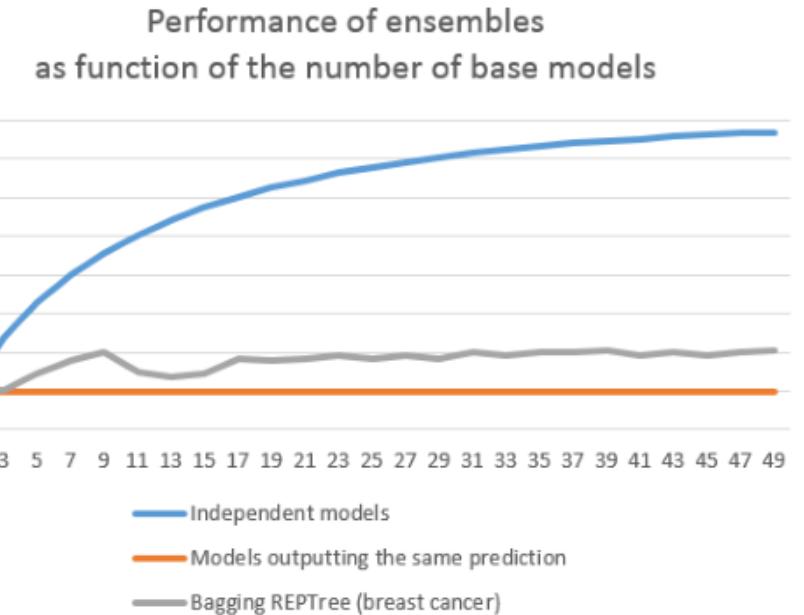
- Main idea: we combine multiple (simple) learning algorithms to have better performance
- Why is it good?
 - We have a binary classification task
 - We have 100 classification models
 - Assume that all of them misclassify a record with probability 0.4 (independently from each other)
 - The probability that more than half of the classifiers misclassify a single record is:

$$\sum_{k=50}^{100} \binom{100}{k} \cdot 0,4^k \cdot 0,6^{100-k} \approx 0,027$$

- Using majority voting we assign the right class with probability 0.973 (or 97.3%)

Performance of ensembles

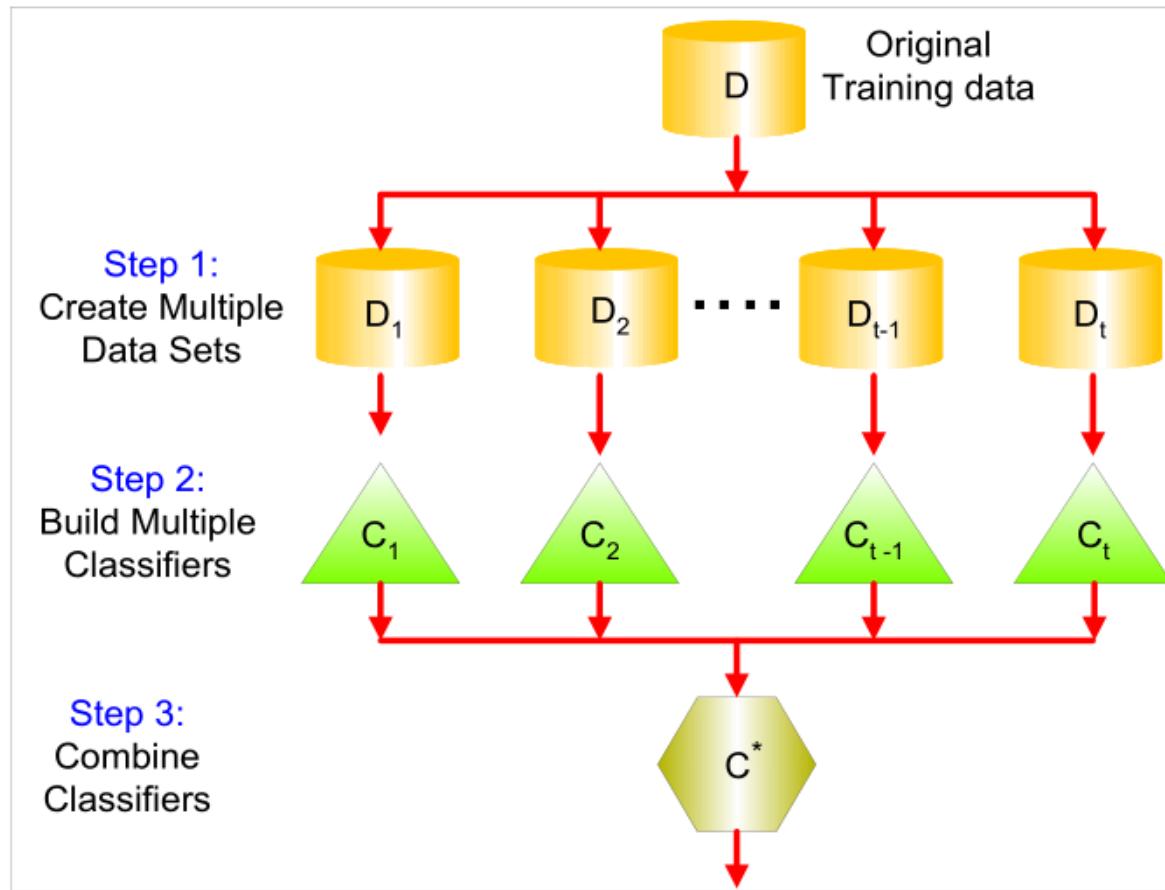
- In practice the increase in performance is not that sharp
 - In reality the models are not independent from each other
- But combining multiple classification algorithms usually results in better predictive performance than could be obtained from any of the composing learning algorithms alone
- It helps to reduce variance
- It helps to avoid overfitting



Motivations for ensemble learning

- Statistical motivation
 - Using ensemble methods, the algorithm can average the different hypotheses and reduce the risk of choosing the wrong classifier
- Computational motivation:
 - Individual classifiers may be stuck in local optima, an ensemble algorithm may provide a better result
- Representational motivation:
 - Ensemble methods can expand the space of hypotheses searched by the individual learning algorithms

Main idea of ensemble learning



Multiple classifiers on a training set

- If the training set is very large: we partition the dataset into disjoint subsets, and we train an individual classifier on each subset
 - Usually the training set is not large enough to do that
- Bagging: random sampling with replacement
 - We create the samples independently from each other
 - The selection of all record is with equal probability
- Boosting: creating the sample in a sequential way, taking into account the previous classifier's success
 - After each training step the selection probabilities (weights) are redistributed
 - Misclassified data increases its weights to emphasize the most difficult cases

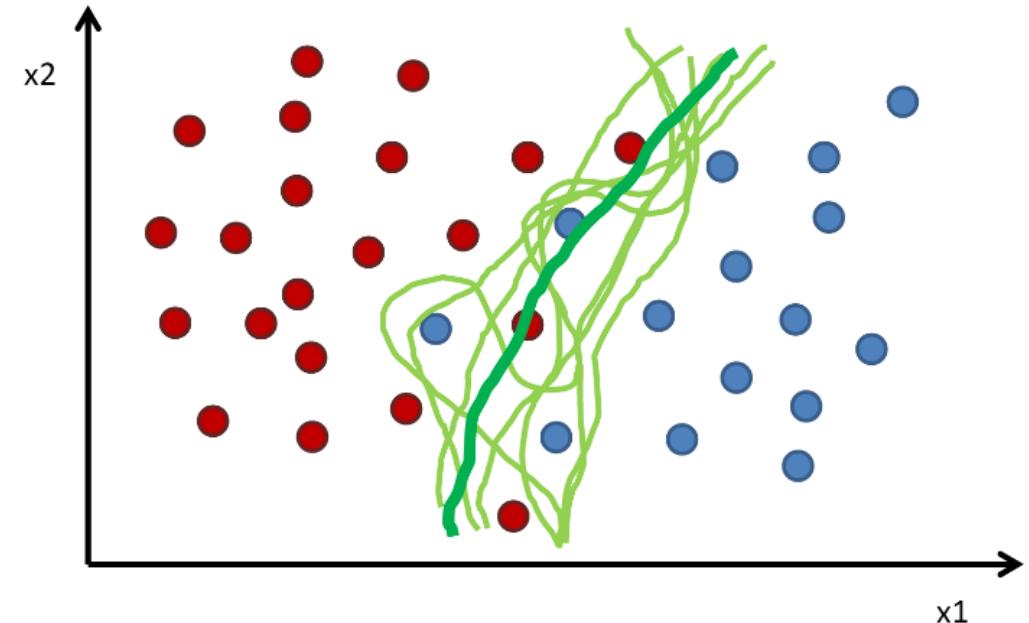
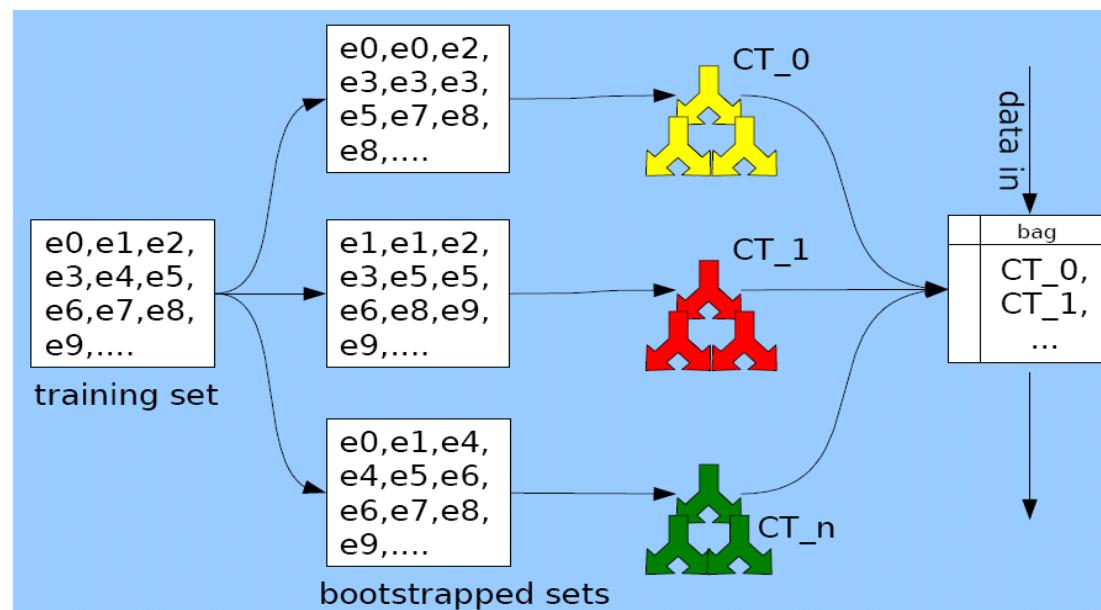
Bagging

- Bagging (**Bootstrap Aggregating**)
- We train the individual models on random samples sampled with replacement

Original Data	1	2	3	4	5	6	7	8	9	10
Bagging (Round 1)	7	8	10	8	2	5	10	10	5	9
Bagging (Round 2)	1	4	9	1	2	3	2	7	3	2
Bagging (Round 3)	1	8	5	10	5	5	9	6	3	7

- The probability of selecting a record: $1 - \left(1 - \frac{1}{n}\right)^n \rightarrow 1 - \frac{1}{e} \approx 0.632$
- The final result is aggregated using majority voting or averaging (for regression)

Bagging - examples



Bagging for attributes

- Attribute bagging (random subspace method)
- Here the difference between the models is not due to sampling the training set but sampling the feature set
- We create (not necessarily disjoint) random samples of the features and train individual models using the sampled feature set

Using meta-models (stacking)

- The results of the individual (base-level) models are combined using a meta-classifier or meta-regressor
 - More sophisticated than just using majority voting or averaging
- The training set is partitioned into two disjoint sets: S_1 and S_2
 - On S_1 the base classifiers are trained
 - The records of set S_2 are classified using the base models, the meta-model is trained on the outputs of the base level models as features

Boosting

- Boosting: creating the sample in a sequential way, taking into account the previous classifier's success
 - After each training step the selection probabilities (weights) are redistributed
 - Misclassified data increases its weights to emphasize the most difficult cases
 - In the example record 4 seems to be a difficult one

Original Data	1	2	3	4	5	6	7	8	9	10
Boosting (Round 1)	7	3	2	8	7	9	4	10	6	3
Boosting (Round 2)	5	4	9	4	2	5	1	7	4	2
Boosting (Round 3)	4	4	8	10	4	5	4	6	3	4

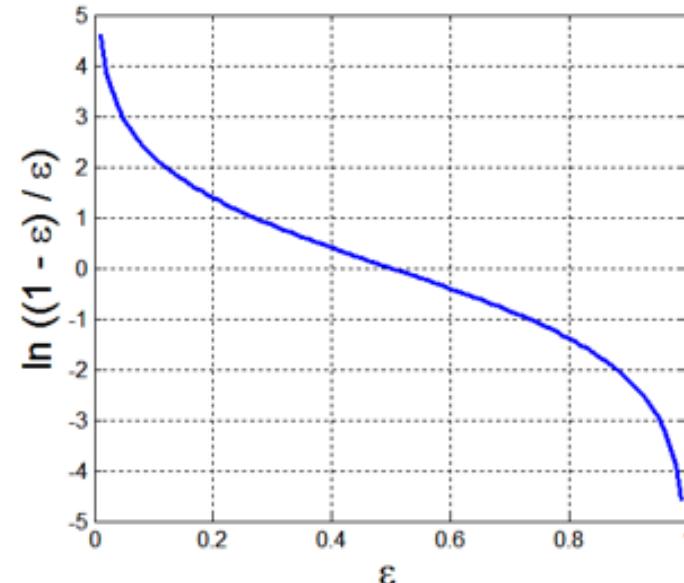
AdaBoost

- Adaptive boosting algorithms
- Base classifiers: C_1, C_2, \dots, C_T
- In step j classifier C_j is trained, and in step j the record i has weight $w_i^{(j)}$
- The error rate of classifier C_j :

$$\varepsilon_j = \frac{1}{N} \sum_{i=1}^N w_i \delta(C_j(x_i) \neq y_i)$$

- The importance of classifier C_j :

$$\alpha_j = \frac{1}{2} \ln \left(\frac{1 - \varepsilon_j}{\varepsilon_j} \right)$$



AdaBoost II.

- Updating the weights

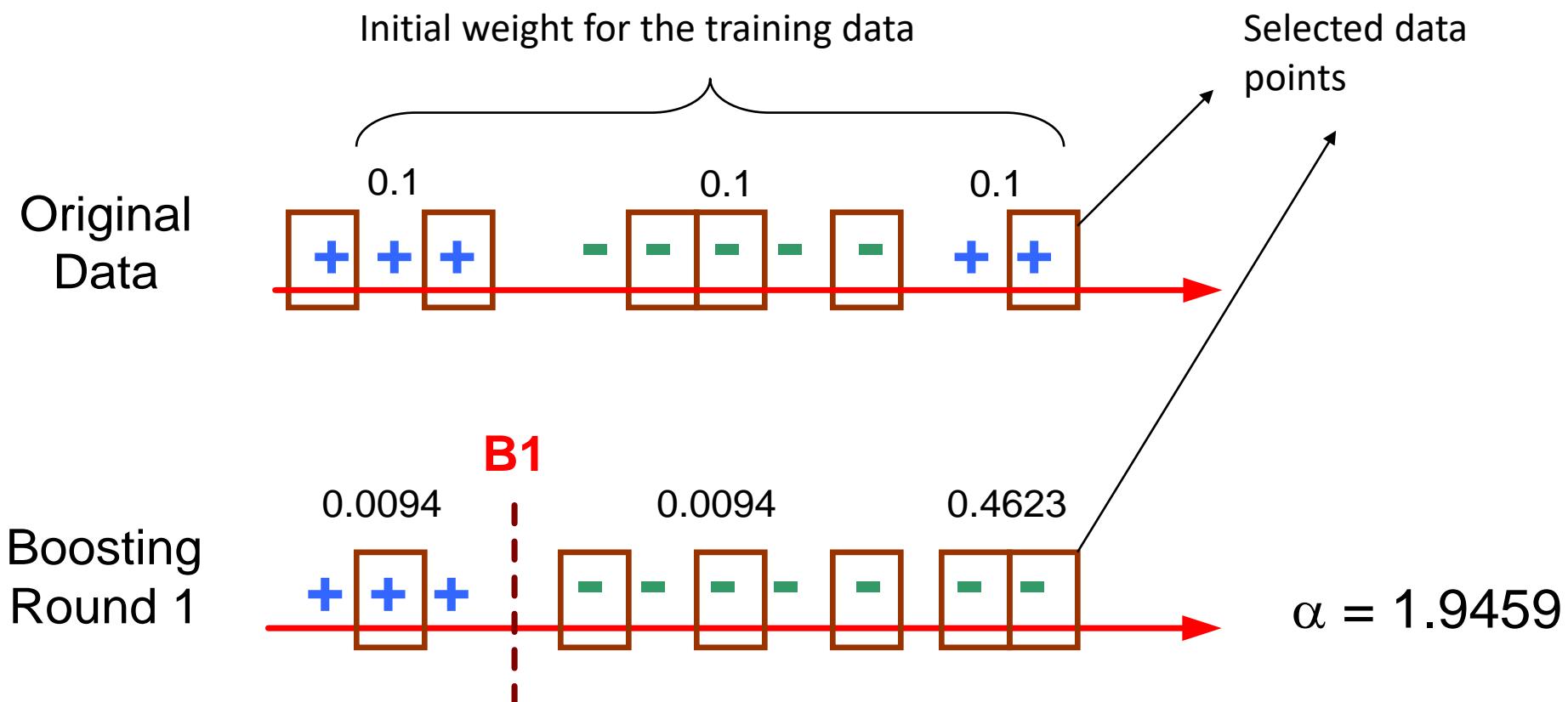
$$w_i^{(j+1)} = \frac{w_i^{(j)}}{Z_j} \begin{cases} e^{-\alpha_j} & \text{if } C_j(x_i) = y_i \\ e^{\alpha_j} & \text{if } C_j(x_i) \neq y_i \end{cases}$$

where Z_j is the normalizing factor to ensure that the sum of the weights is equal to 1

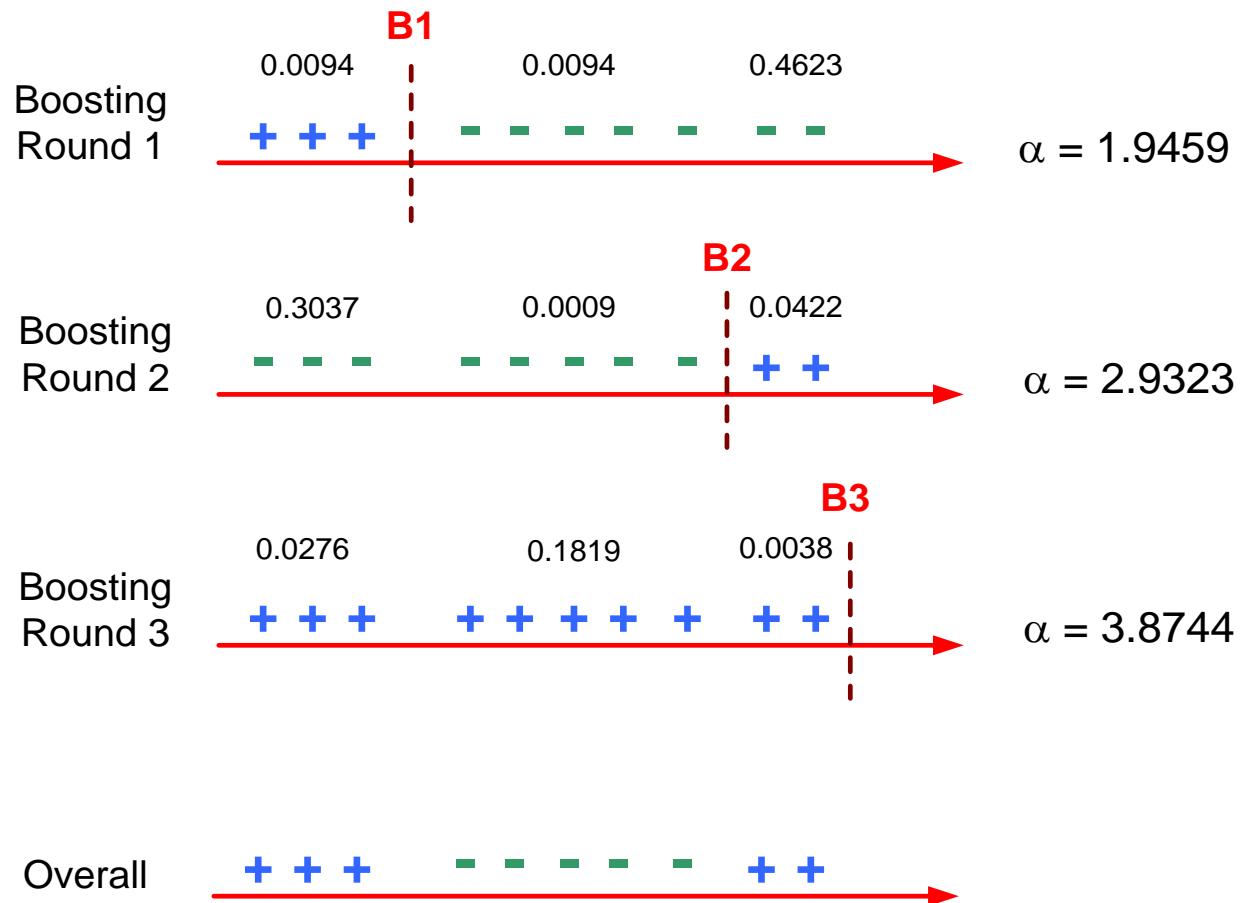
- Classification

$$C^*(x) = \arg \max_y \sum_{j=1}^T \alpha_j \delta(C_j(x) = y)$$

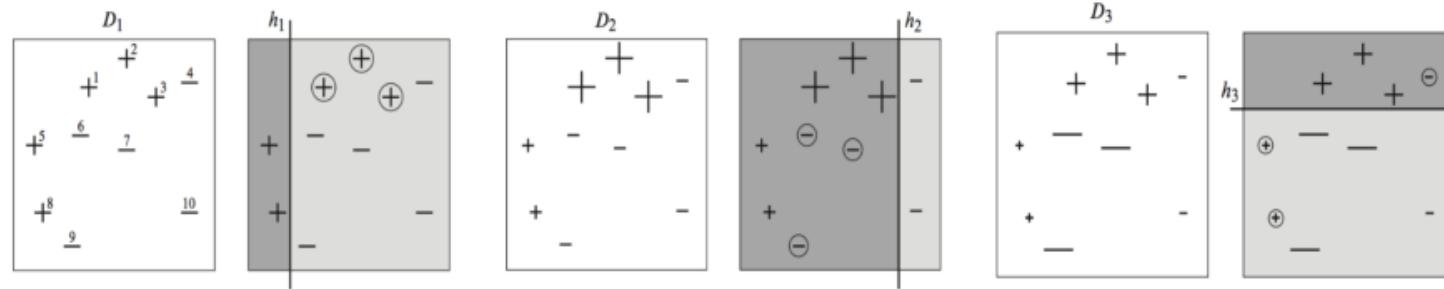
AdaBoost - example



AdaBoost – example II.



AdaBoost – another example

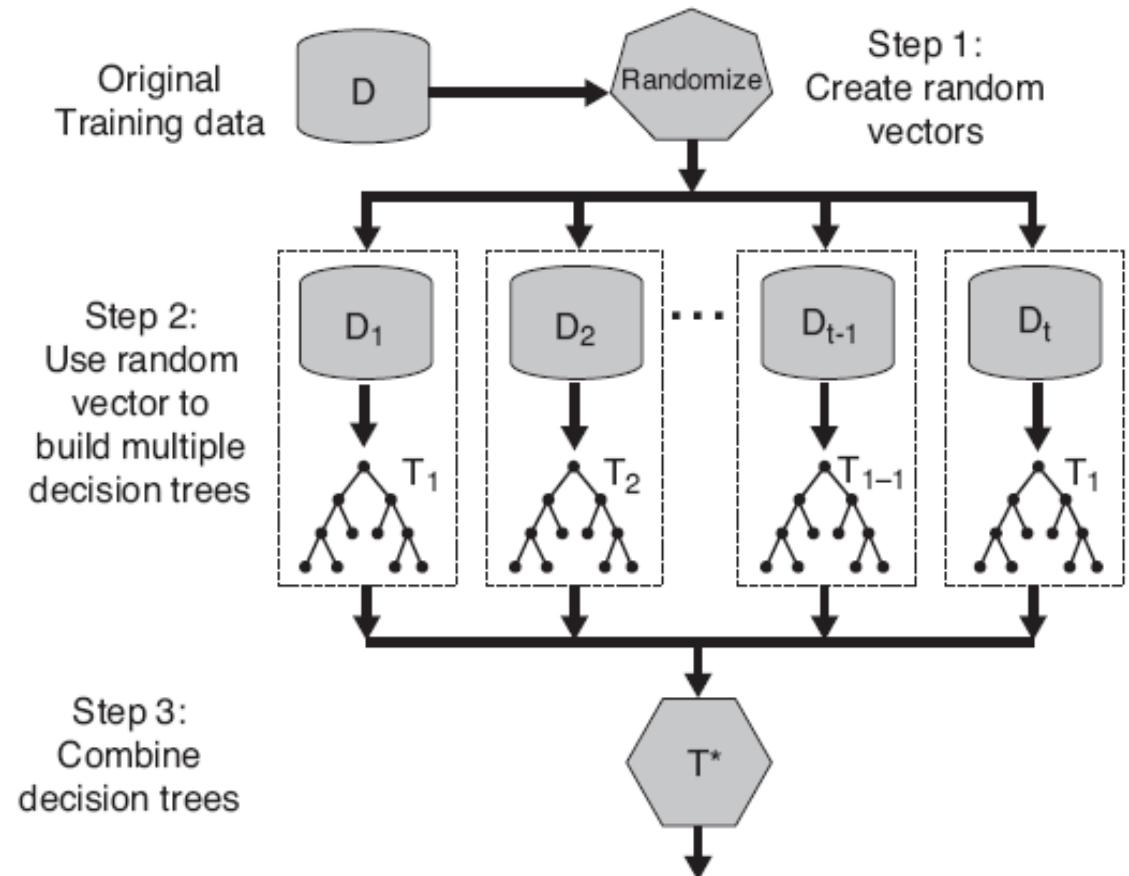


$$H = \text{sign} \left(0.42 \begin{array}{|c|c|} \hline \text{+} & \text{-} \\ \hline \text{+} & \text{-} \\ \hline \end{array} + 0.65 \begin{array}{|c|c|} \hline \text{+} & \text{-} \\ \hline \text{-} & \text{-} \\ \hline \end{array} + 0.92 \begin{array}{|c|c|} \hline \text{+} & \text{-} \\ \hline \text{-} & \text{-} \\ \hline \end{array} \right)$$

$$= \begin{array}{|c|c|} \hline \text{+} & \text{+} \\ \hline \text{-} & \text{-} \\ \hline \text{+} & \text{-} \\ \hline \text{-} & \text{-} \\ \hline \end{array}$$

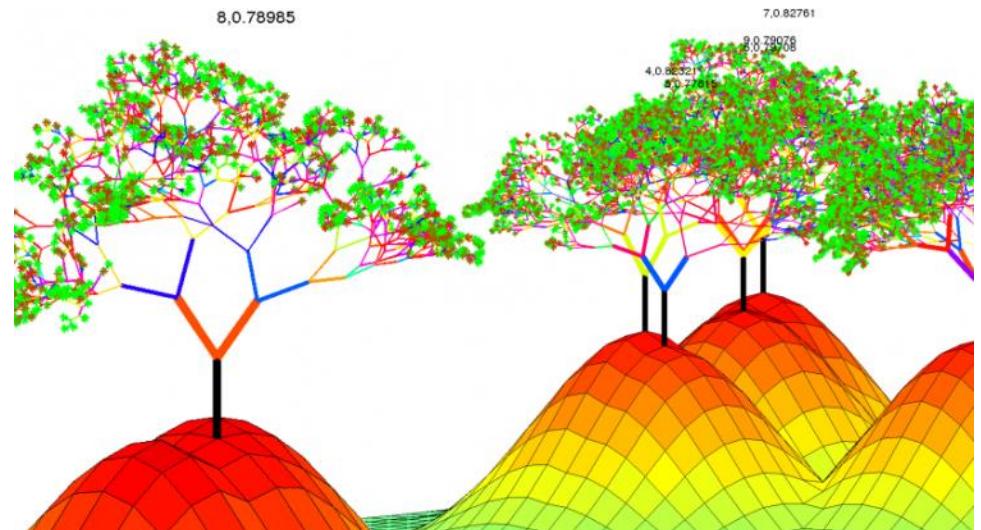
Random forest

- We build multiple decision trees in such a way that
 - We create random samples from the training set using bagging method
 - We use attribute bagging for the features for training the individual trees
 - the feature set for an individual tree is much smaller than the original feature set



Random forest II.

- The base classifiers are decision trees
 - Trained on random samples (bagging)
 - For an individual tree we restrict the feature space for a much smaller number of features
 - The individual trees grow according to a decision tree algorithm
- The random forest uses majority voting on the results of the individual trees

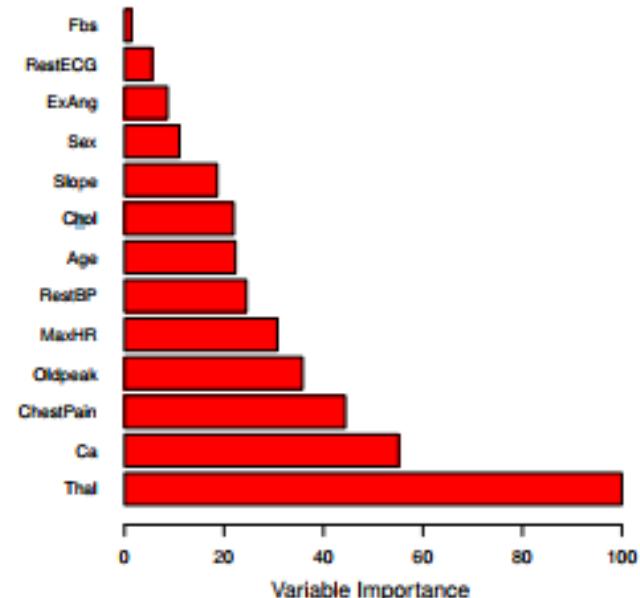


Efficiency of random forest

- Usually we generate a high number of decision trees
- The efficiency of RF model depends on the followings:
 - Number of generated trees (the more trees, the better result)
 - The correlation between the trees (the higher correlation, the worse result)
 - BUT: The more tree we have, the higher the correlation is
 - The more attributes we let the decision trees to split on, the higher the correlation between the trees is
 - We can optimize for these parameters:
 - number of trees
 - used number of features for each tree

Evaluation of random forest

- The interpretability of the model is worse than for a simple decision tree
- Fast, easy to parallelize
- It can be used to measure the feature importance
 - How many times the decision trees use a certain attribute to split on
- Reduce overfitting
- Can handle high dimension
- Small number of parameters (number of trees, cardinality of the reduced feature space)

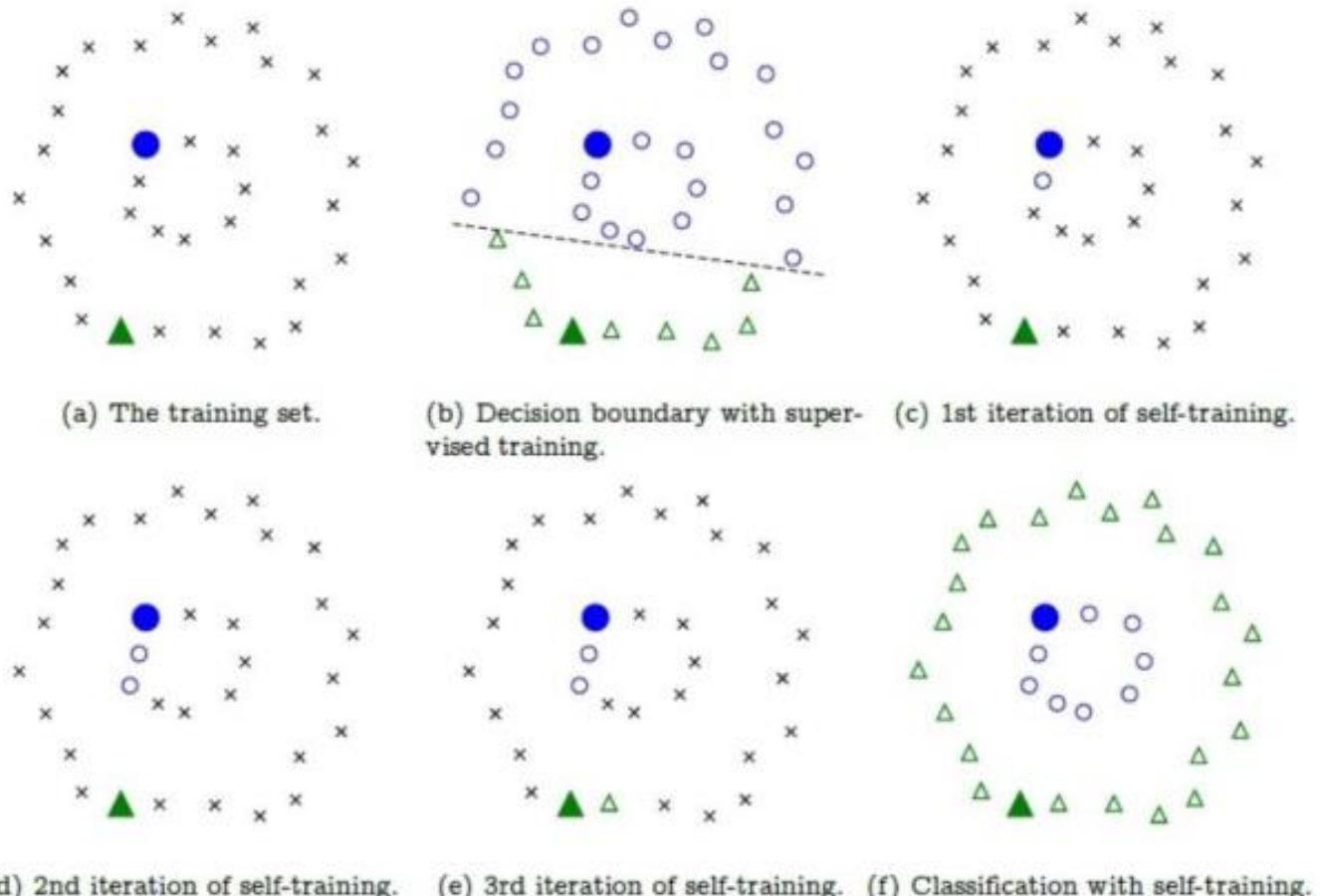


Semi-supervised learning

- We use this method if the training set is small or non-representative
- Procedure
 - We classify the object whose label we are the most confident in
 - We add the previously classified object (with the predicted label) to the training set
 - We retrain the model, and repeat the procedure on the unlabeled data points
- Computationally expensive
- The algorithm is able to take into consideration the structure of the unlabeled data points

Example – semi supervised learning

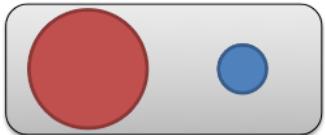
- kNN classifier
 - Two classes
 - The filled objects mark the records with known labels



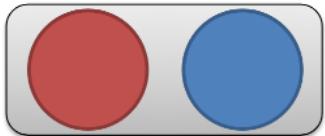
Classifying imbalanced data

- Goal: balancing the class distribution
- Methods:
 - Under-sampling, sub-sampling:
we reduce the size of the frequent class by
eliminating the objects that are easy to
classify (that are far away from the decision
boundary)
 - Disadvantage: we lose data
 - Over-sampling: synthetically supplementing the training data of the minority
class
 - We won't have more information on the minority class, still it can help to improve the
performance of classifiers

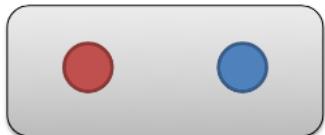
The Problem:



Oversample:

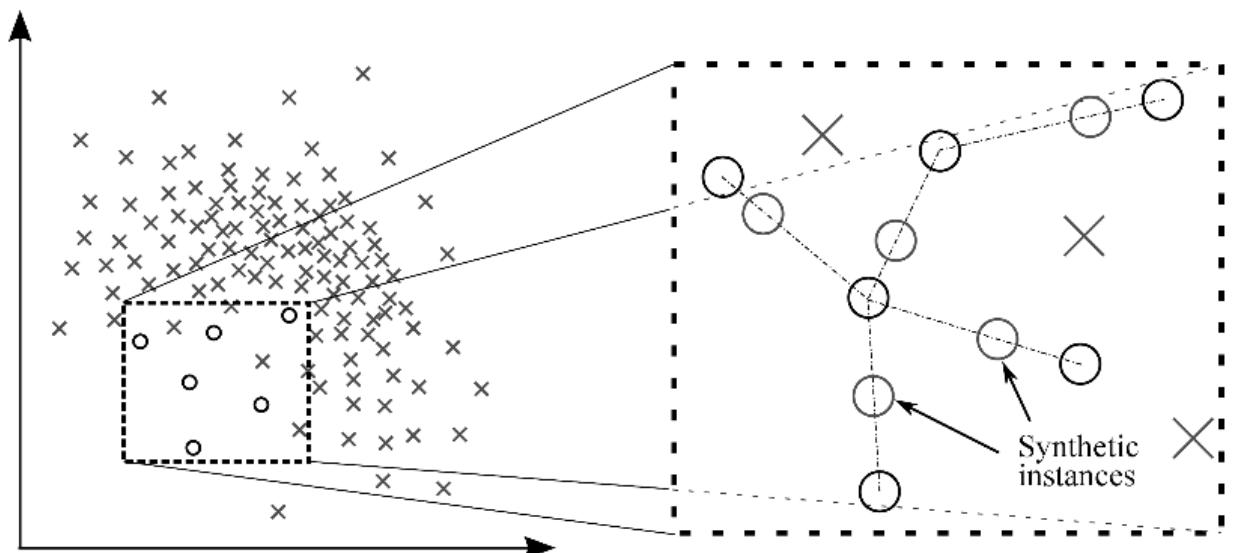


Subsample:



SMOTE

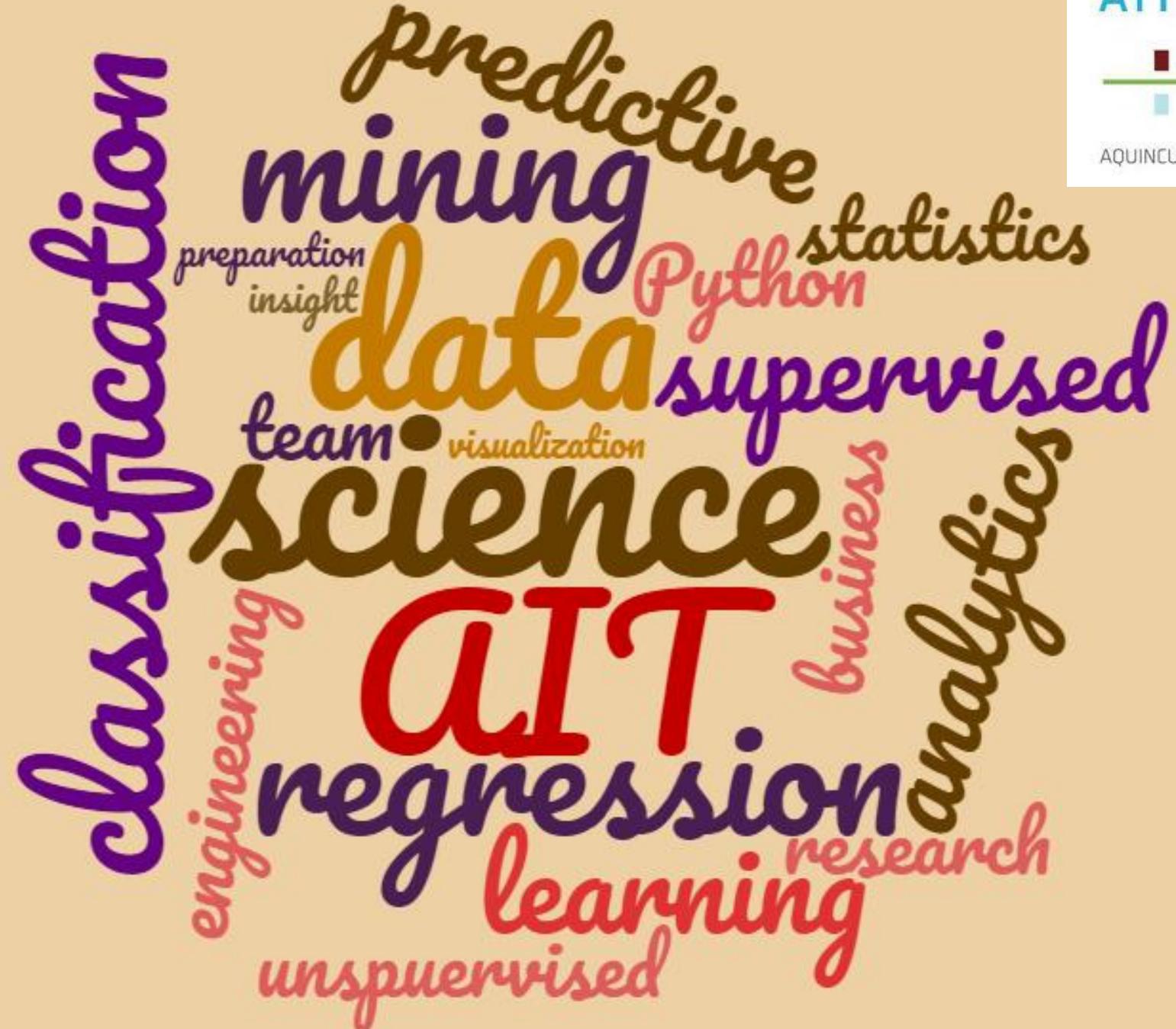
- SMOTE (Synthetic Minority Over-sampling Technique)
- The most common oversampling procedure
- For all instances from the minority class we consider their k nearest neighbors from the minority class.
- To create a synthetic data point by taking a random point on the segment between the neighbor and current data point



Acknowledgement

- András Benczúr, Róbert Pálovics, SZTAKI-AIT, DM1-2
- Krisztián Buza, MTA-BME, VISZJV68
- Bálint Daróczy, SZTAKI-BME, VISZAMA01
- Judit Csima, BME, VISZM185
- Gábor Horváth, Péter Antal, BME, VIMMD294, VIMIA313
- Lukács András, ELTE, MM1C1AB6E
- Tim Kraska, Brown University, CS195
- Dan Potter, Carsten Binnig, Eli Upfal, Brown University, CS1951A
- Erik Sudderth, Brown University, CS142
- Joe Blitzstein, Hanspeter Pfister, Verena Kaynig-Fittkau, Harvard University, CS109
- Rajan Patel, Stanford University, STAT202
- Andrew Ng, John Duchi, Stanford University, CS229





Schedule of the semester

	<i>Monday midnight</i>	<i>Tuesday class</i>	<i>Friday class</i>
W1 (02/06)			
W2 (02/13)		HW1 out	
W3 (02/20)			
W4 (02/27)	HW1 deadline + TEAMS	HW2 out	
W5 (03/06)			PROJECT PLAN
W6 (03/13)	HW2 deadline	HW3 out	
W7 (03/20)			MIDTERM
SPRING BREAK		SPRING BREAK	SPRING BREAK
W8 (04/03)	HW3 deadline		GOOD FRIDAY
W9 (04/10)	MILESTONE 1		
W10 (04/17)		HW4 out	
W11 (04/24)			
W12 (05/01)	HW4 deadline		
W13 (05/08)	MILESTONE 2		
W14 (05/15)		FINAL	PROJECT presentations
W15 (05/22)		PROJECT presentations	

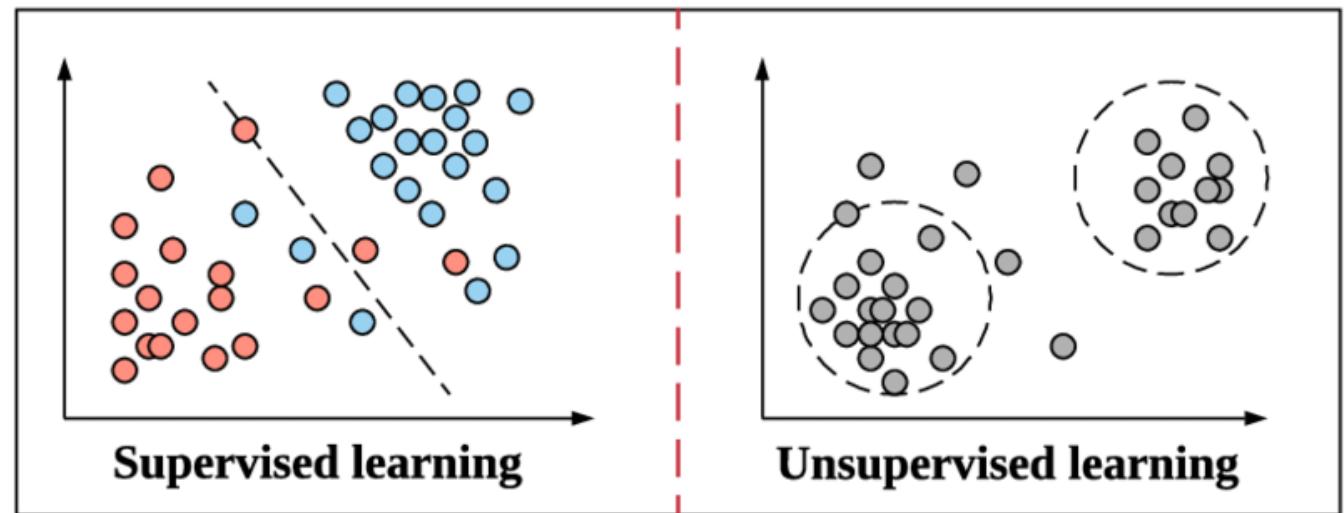
Milestone 2

- Three-page long report including:
 - Reviewing the related works
 - Data understanding and data preparation steps
 - Data analysis steps, implementing some models and evaluating them



Unsupervised learning

- The target (label) is not known for any records (latent labels)
- Aim: to associate useful labels to the records based on the attributes
- In many cases our aim is to gain better understanding of the data or visualize the data

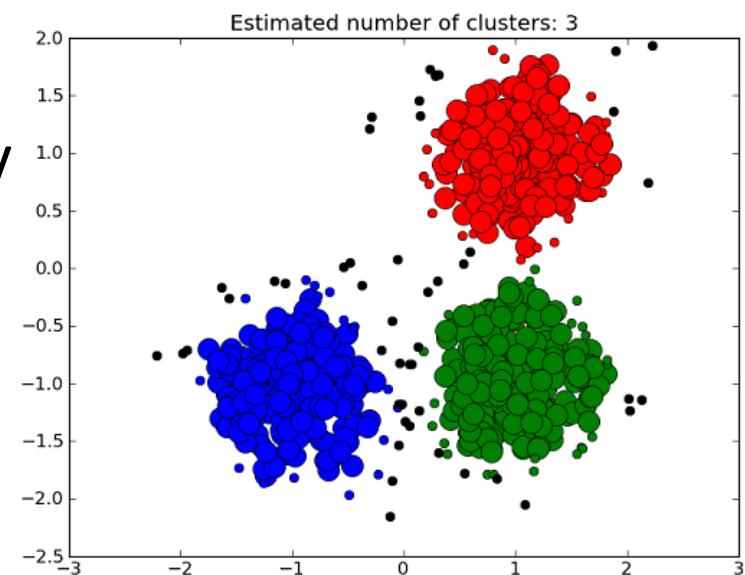


Clustering

- Unsupervised learning
- Grouping similar objects together
 - Aim: objects within a group should be more similar to each other compared to objects from different groups
- How to measure similarity? → similarity measures
- Challenges: What features is the clustering based on? How to measure similarity? How many clusters do we want? How to evaluate a clustering? How to visualize it?

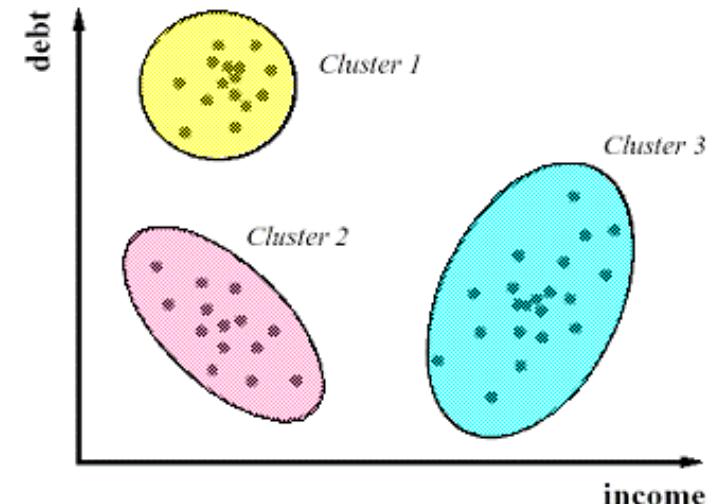
Goal of clustering

- For some data science projects the goal is to solve a clustering task (group similar objects together)
- Sometimes clustering is part of the explanatory analysis
 - Recognize some (hidden) pattern in the data
 - Visualization, gain a better understanding
 - Reducing the complexity of the data by clustering
- The evaluation of clustering and the (optimal) number of clusters depend on the application domain



Clustering - examples

- Customer segmentation
 - Features: Customer data (sex, age, address, profession, ...), purchase history
- Grouping documents based on their content
 - Features: words, n-grams appearing in the text
- Grouping pictures
 - Features: extracted from the pixels



Evaluation of clustering

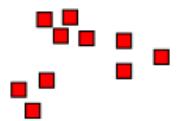
- The evaluation is ambiguous, there are no obvious evaluation methods
- It is difficult to judge how good a clustering is (there is no ground truth)



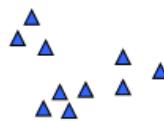
How many clusters?



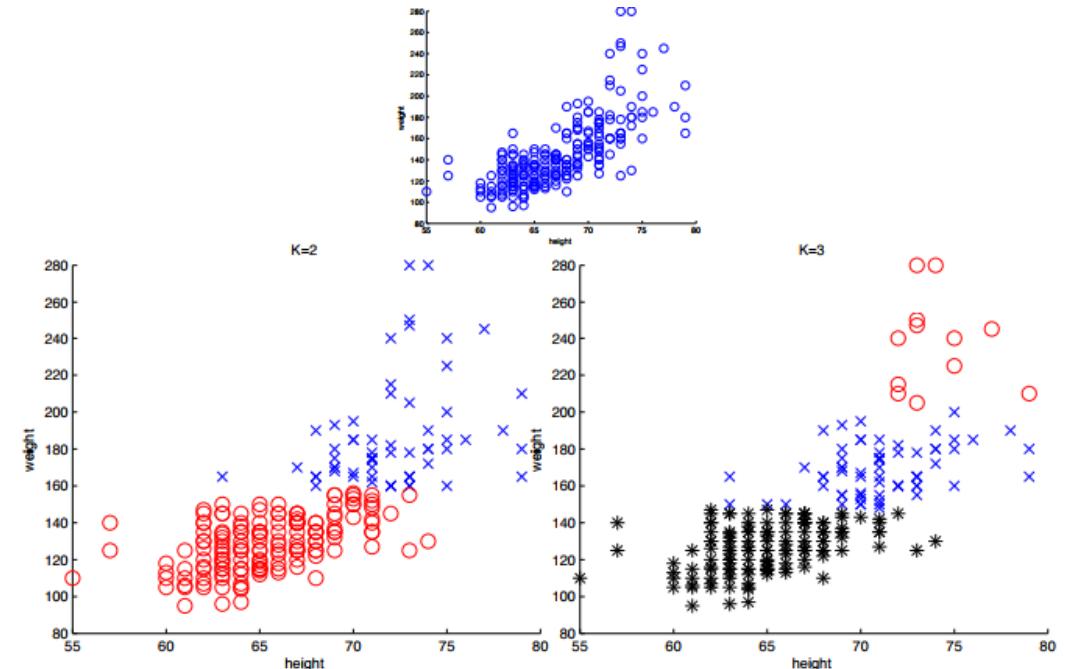
Six Clusters



Two Clusters



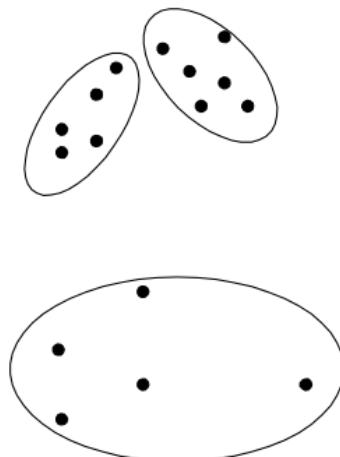
Four Clusters



Types of clustering algorithms

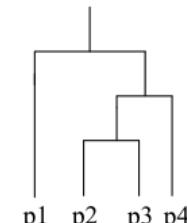
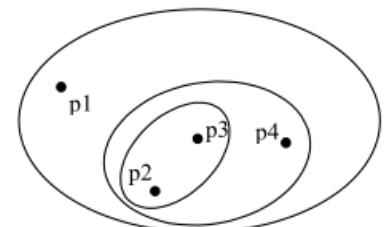
Partitional

- We partition the records into non-overlapping subsets (clusters)
- Each record is in exactly one cluster



Hierarchical

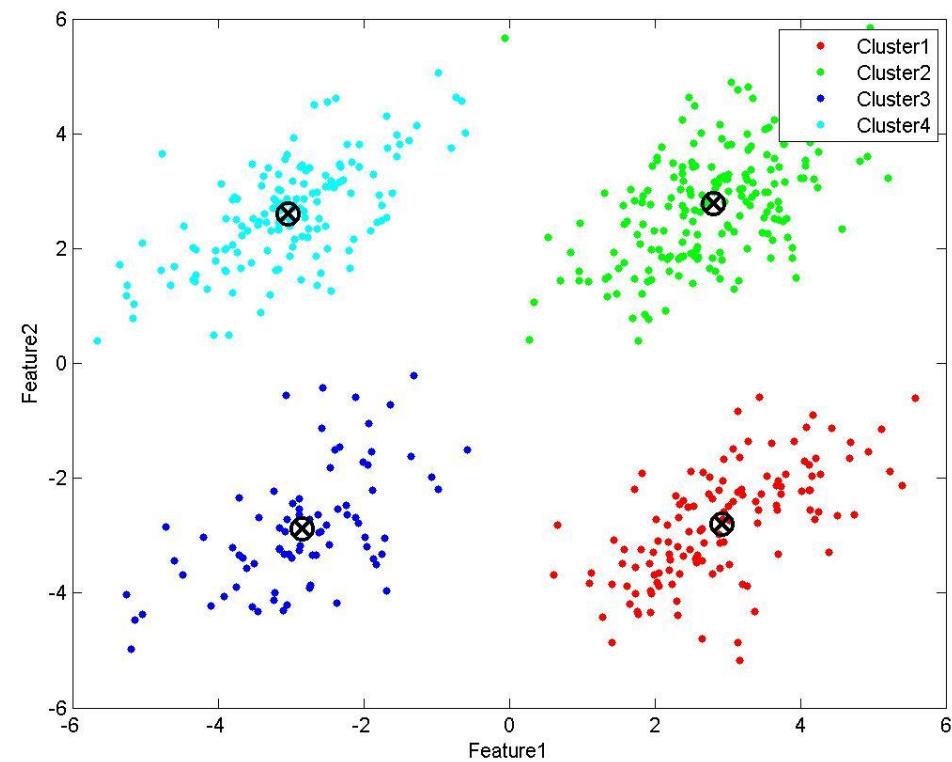
- The clusters are nested
- A record corresponds to a nested hierarchy of clusters



dendrogram

Center-based clustering

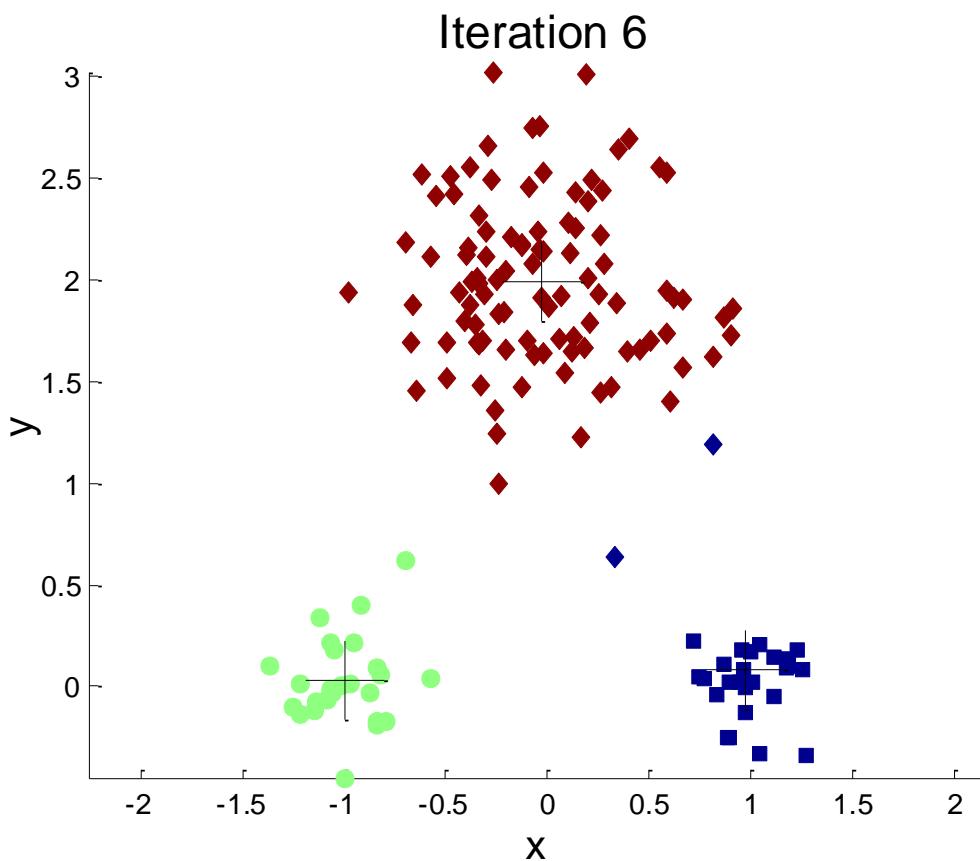
- Center-based or prototype-based
- Every cluster has a representative center point
- Every record corresponds to the cluster whose center point is the closest to the record
 - The center point can be the centroid (the average of the records) or the medoid (the most „representative” datapoint)



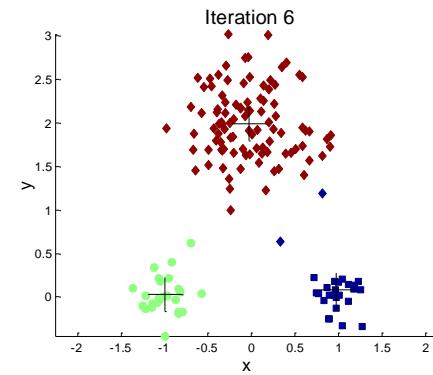
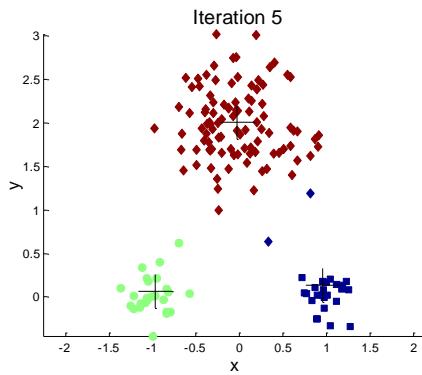
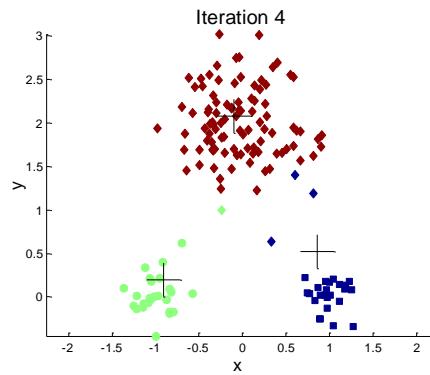
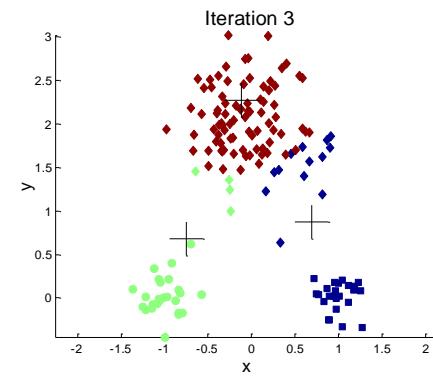
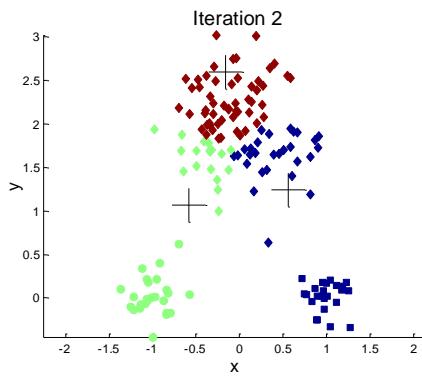
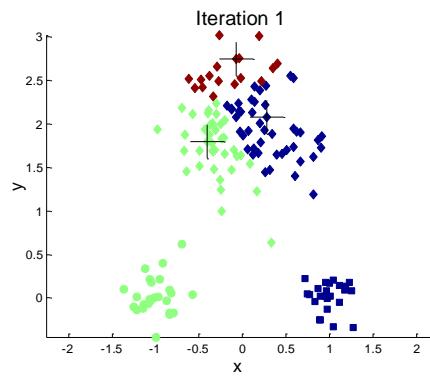
K-means algorithm

- Partitional, center-based clustering algorithms that creates a given K number of clusters
- For a given K number
 1. Choose K initial centroids in the n -dimensional space (we have n attributes), the centroids are not necessarily data points themselves
 2. Assign each record to the closest centroid to form clusters
 3. Calculate the new centroids (means of the records) of the clusters
 4. Repeat point 2 and 3 until convergence (when the assignments no longer change)

K-means algorithm - iterations

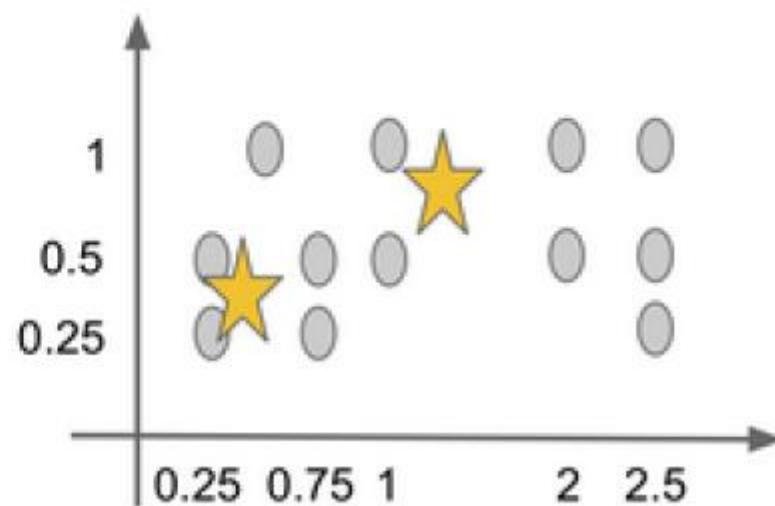


K-means algorithm – iterations II.



Problem

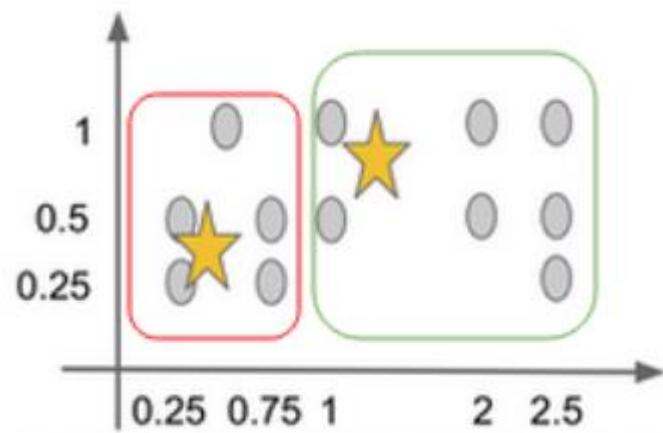
We initialize k-means clustering algorithm with the centroids marked by stars in the figure. Perform an iteration step! Calculate the positions of the new centroids!



Solution

We assign each data point to the nearest centroid. Then we calculate the mean of the coordinates of the data points in each cluster. The means of the coordinates are the new coordinates of the centroids.

Right (green) box:



$$x = \frac{1 + 1 + 2 + 2 + 2.5 + 2.5 + 2.5}{7} = 1.9$$
$$y = \frac{0.25 + 0.5 + 0.5 + 0.5 + 1 + 1 + 1}{7} = 0.68$$

The corresponding centroid: (1.92, 0.68)

Left (red) box:

$$x = \frac{0.25 + 0.25 + 0.5 + 0.75 + 0.75}{5} = 0.5$$
$$y = \frac{0.25 + 0.5 + 1 + 0.25 + 0.5}{5} = 0.5$$

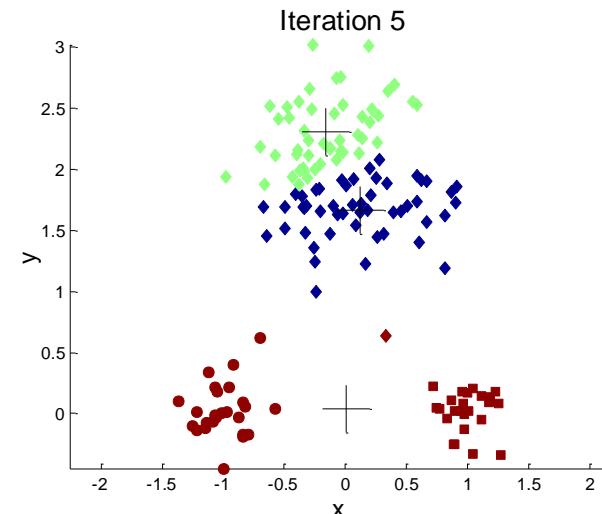
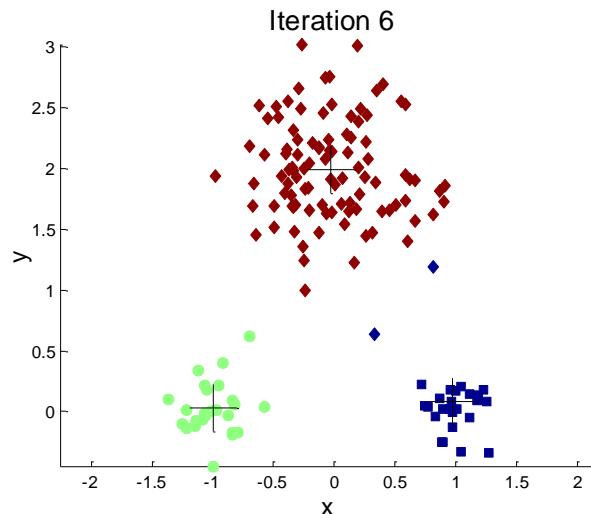
The corresponding centroid: (0.5, 0.5)

K-means - specification

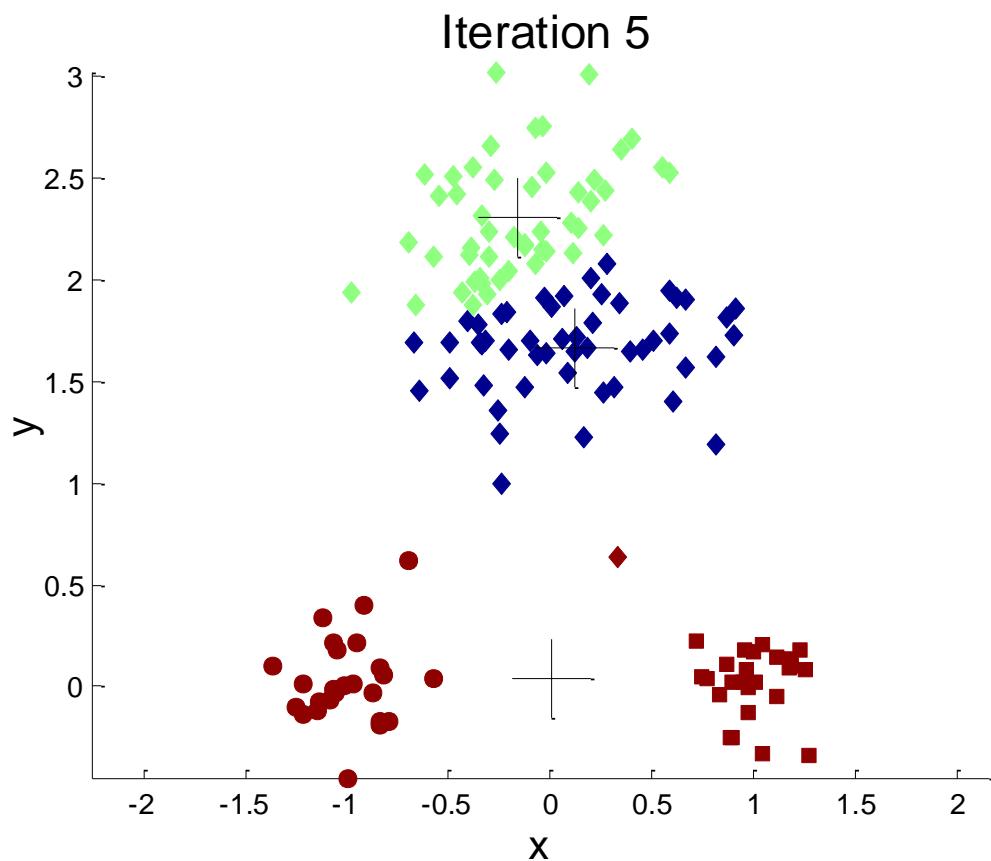
- What (dis)similarity measure to use?
 - Depends on the data, choose the most suitable (dis)similarity
 - Most cases: Euclidean distance
- How to calculate the new centroids?
 - Usually the aim is to minimize the squared distance between records and their centroids (sum of squared „errors”)
 - The mean minimizes the squared error
- Does it converge?
 - Usually yes, for Euclidean distance always
 - The convergence is usually fast
 - Does not guarantee to find the optimum (just local optimum)

Choosing initial centroids

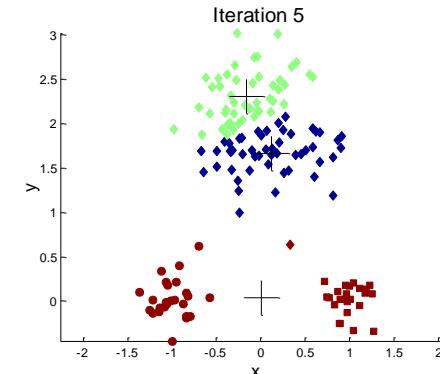
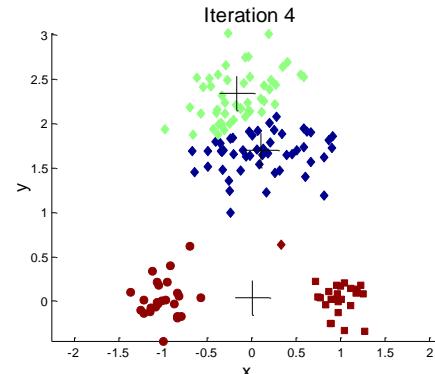
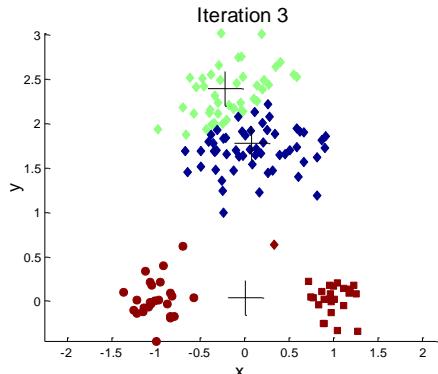
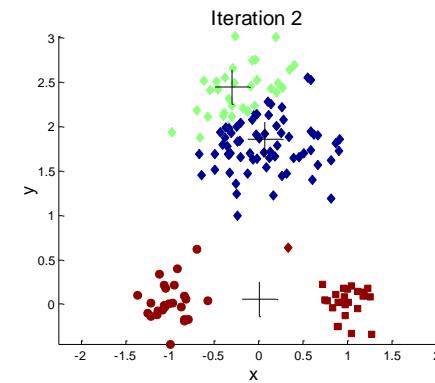
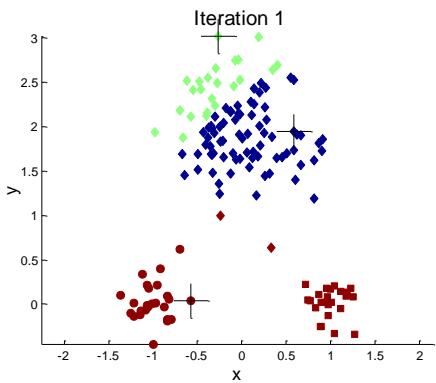
- The final clustering also highly depends on the choice of the initial centroids
- With a bad first initialization we can obtain bad clusters even if there were nice natural clusters



Importance of the initialization



Importance of the initialization II.



Choosing initial data points randomly

- Running the algorithms more times with random initializations
 - Choose the final clustering the lowest sum of squared errors (SSE):
 - SSE or also called total within-cluster sum of squares (WSS)
 - C_i is the i th cluster with centroid c_i

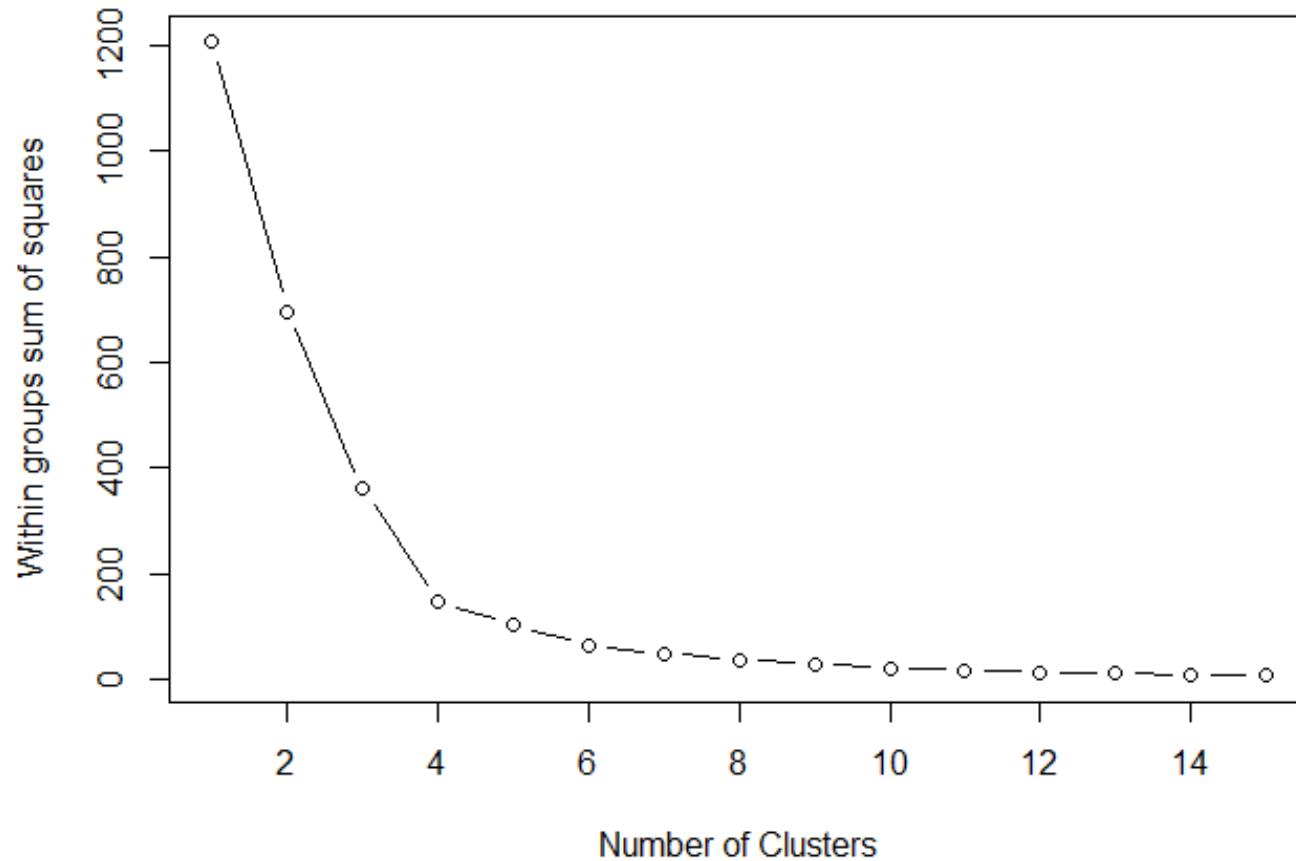
$$\text{SSE} = \sum_{i=1}^K \sum_{x \in C_i} \text{dist}(x, c_i)^2$$

Determining the optimal number of clusters

- The optimal value of K depends on the application domain
- We can try several K values (each with more random initializations)
- We choose the K that gives the most desirable result
- If we solely decide the K value based on the SSE
 - Be careful! The more clusters we have the less the SSE value is
 - If $K=N$, then SSE=0, but it is meaningless
 - The „elbow” method: choose a number of clusters so that adding another cluster doesn't improve much better the SEE

The elbow method

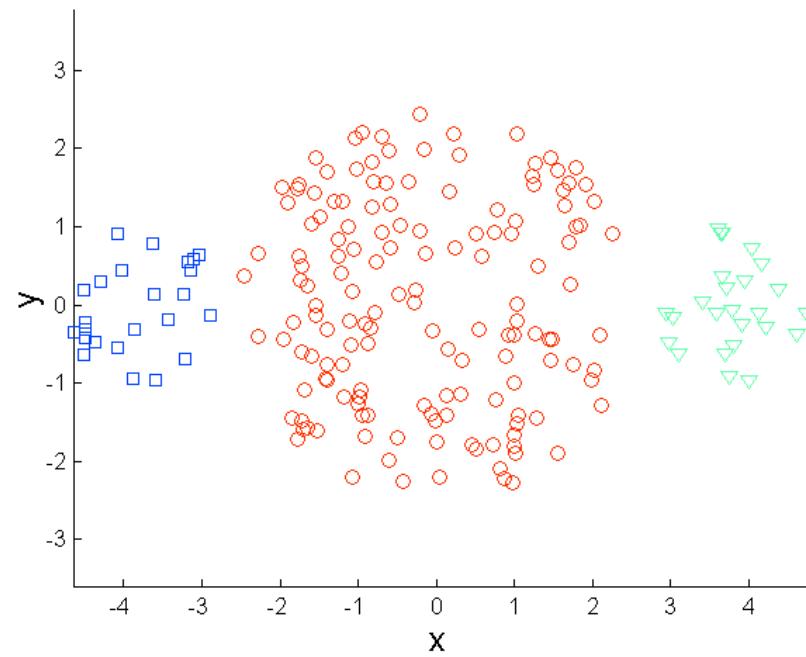
- Scree plot



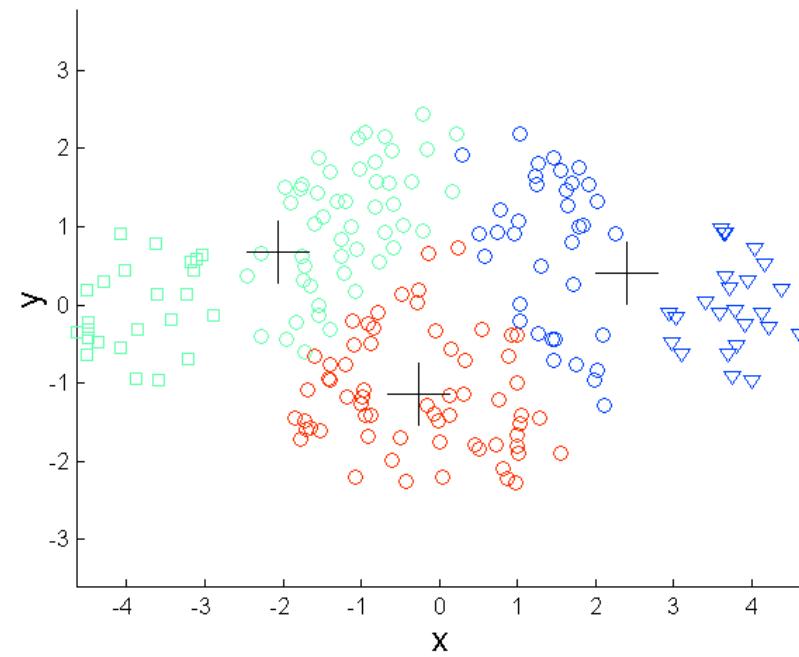
Drawbacks of K-means algorithm

- One should decide on the number of clusters before
- Initial seeds have a strong impact on the final result
- Always construct spherical shapes around the centroids
 - If the natural clusters are not spherical, K-means will not perform well
- K-means cluster tend to be of the same size
 - If the natural clusters differ in size, it will not perform well
- K-means algorithm does not perform well for clusters with differing density

Clusters with differing size

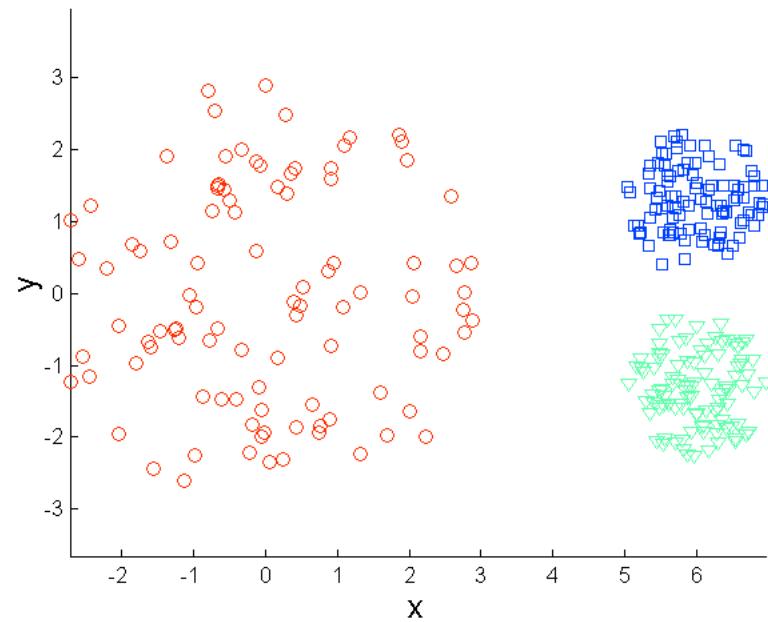


Natural clusters

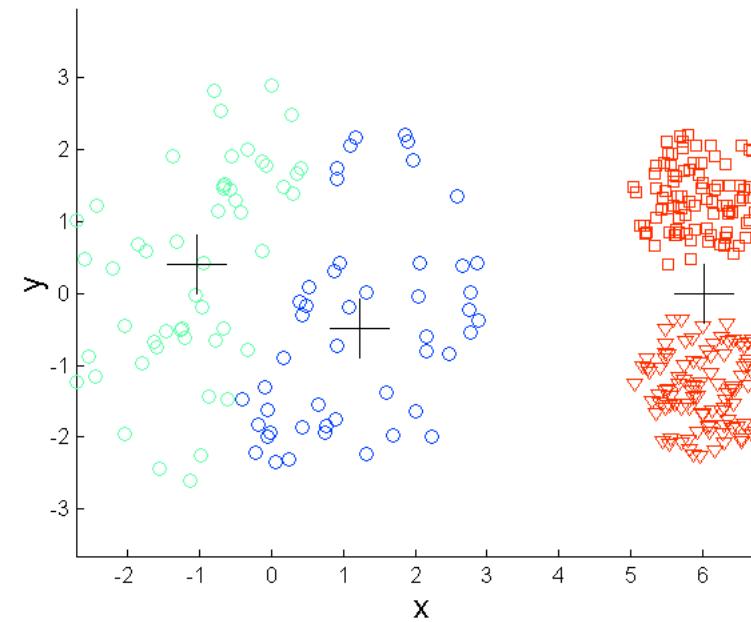


Result of K-means algorithm ($K=3$)

Clusters with differing density

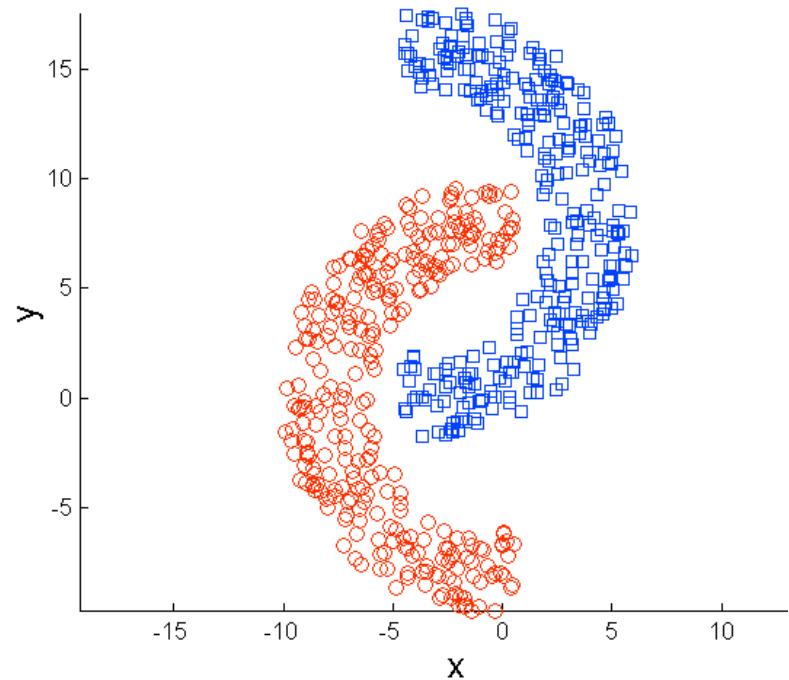


Natural clusters

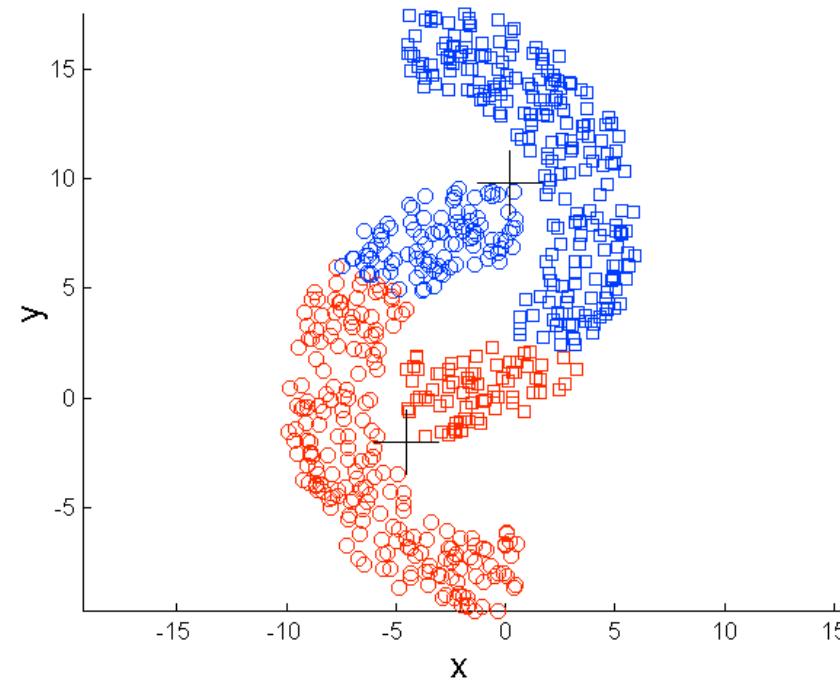


Result of K-means algorithm (K=3)

Non-spherical clusters



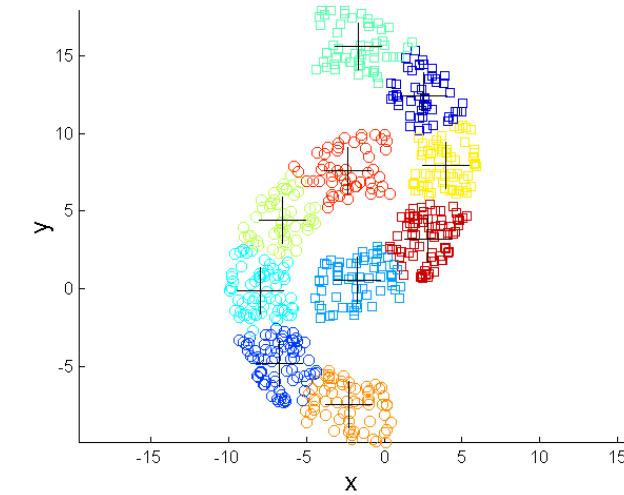
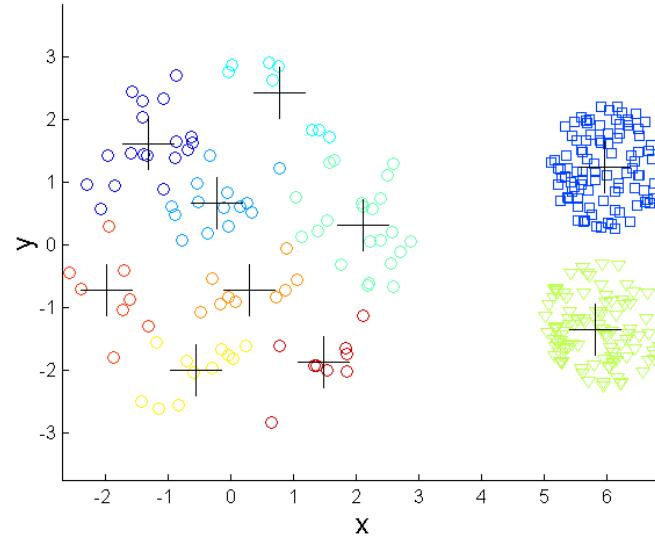
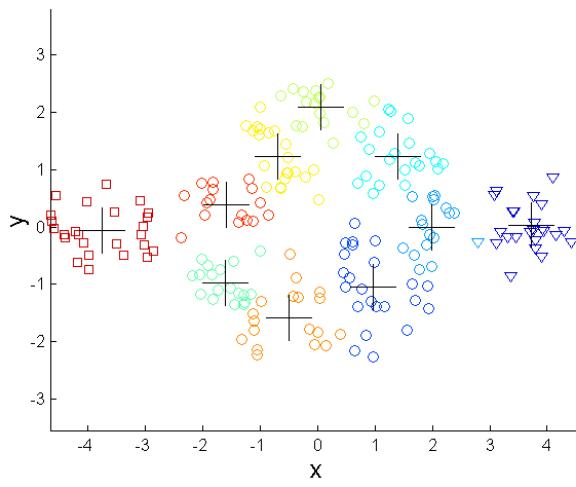
Natural clusters



Result of K-means algorithm (K=2)

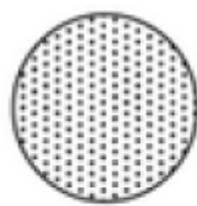
Advantages of K-means

- Easy to implement
- Computationally relatively fast (for small K)
- For the presented drawbacks, a possible solution can be to choose higher K, then the natural clusters will be the union of small clusters

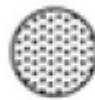


Problem

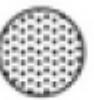
Consider the following sets of two-dimensional points. For the given number of clusters, provide a sketch of the resulting clusters found by k-means algorithm, also indicate the positions of centroids. Assume that Euclidean distance is used and the points are uniformly distributed. If you think that there are more than one possible solutions, then indicate whether a solution is a global or local minimum (i.e. in which case the SSE is smaller) (a) $k = 2$ (b) $k = 3$ (c) $k = 3$ (d) $k = 2$ (e) $k = 3$ (Note that the label of each diagram below matches the corresponding part of this question.)



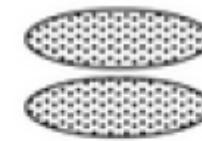
(a)



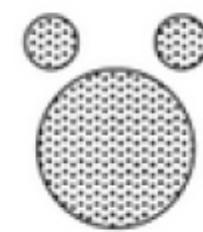
(b)



(c)

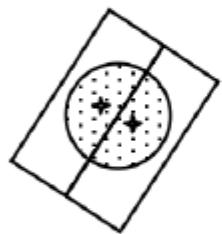


(d)

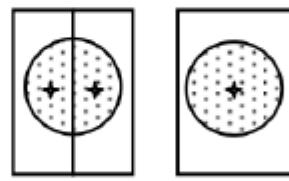


(e)

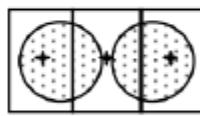
Solution



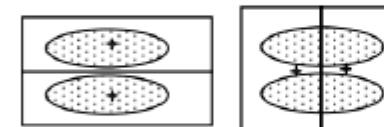
(a)



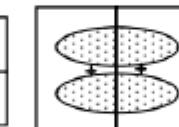
(b)



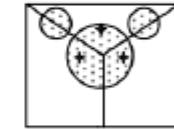
(c)



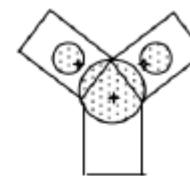
Local minimum



Global minimum



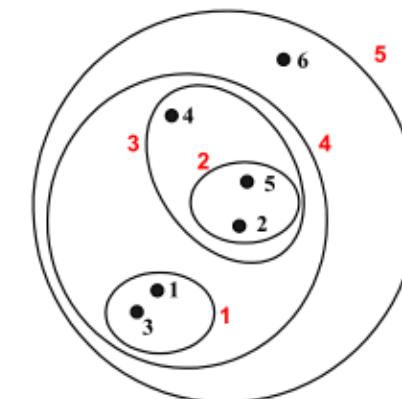
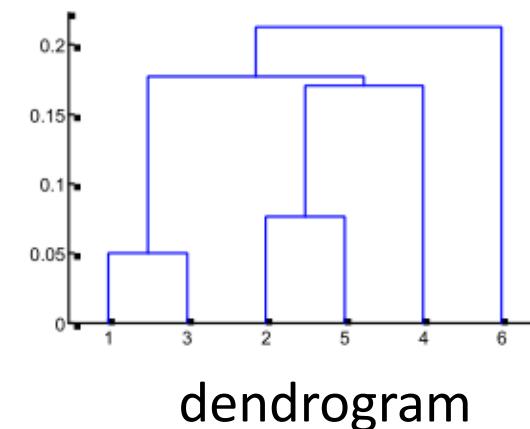
Local minimum



Global minimum

Hierarchical clustering

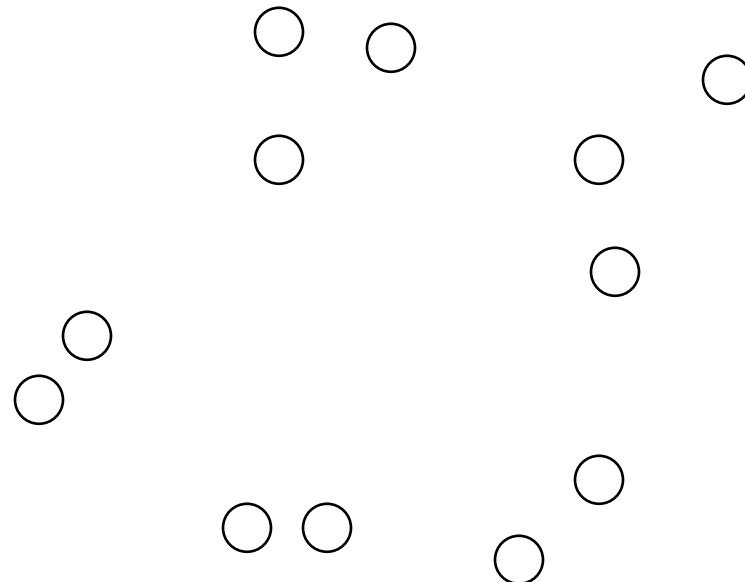
- Construct nested hierarchy of structures
- One doesn't have to determine the number of clusters before, after running the algorithm it is enough to decide on the number of clusters (with the help of the dendrogram)
- Two main strategies
 - Agglomerative
 - „Bottom-up”
 - Divisive
 - „Top-down”



Hierarchical agglomerative clustering

- Each observation starts in its own cluster
- Pairs of clusters are merged as one moves up the hierarchy
 - Until all the observations are merged into one cluster
- We need
 - (dis)similarity measure between observations
 - (dis)similarity measure between clusters
 - How to define the (dis)similarity (or distance) between an observation and a cluster or between two clusters?

Example

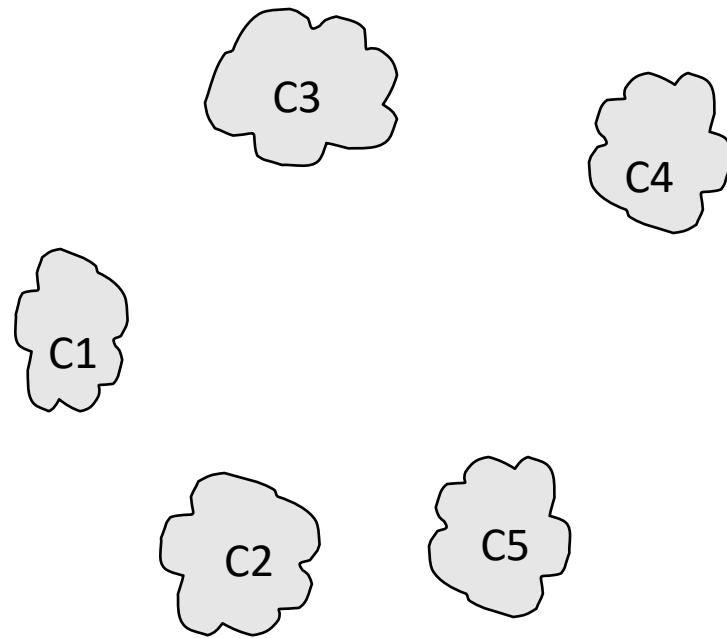


	p1	p2	p3	p4	p5	...
p1						
p2						
p3						
p4						
p5						
.						
.						
.						

Proximity Matrix

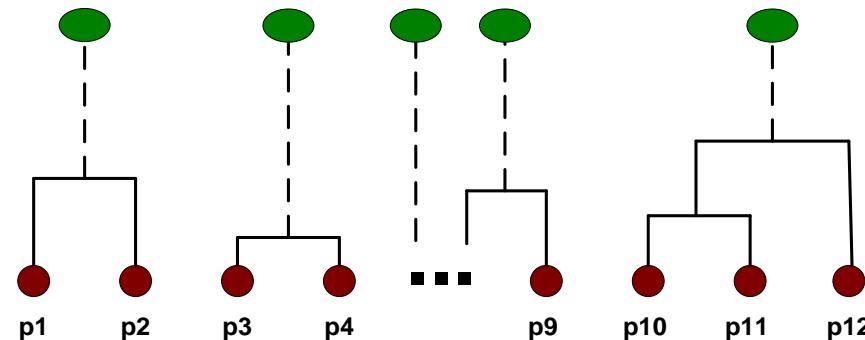
p1 p2 p3 p4 ... p9 p10 p11 p12

After some merging steps



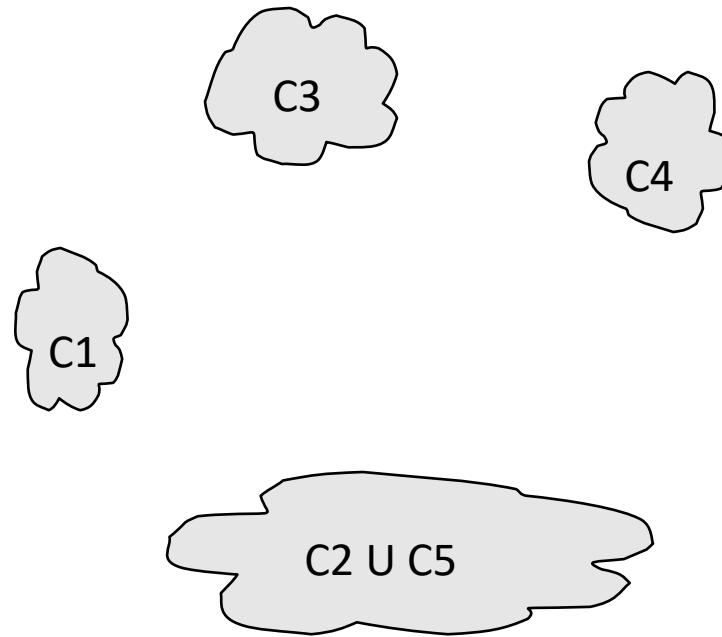
	C1	C2	C3	C4	C5
C1					
C2					
C3					
C4					
C5					

Proximity Matrix



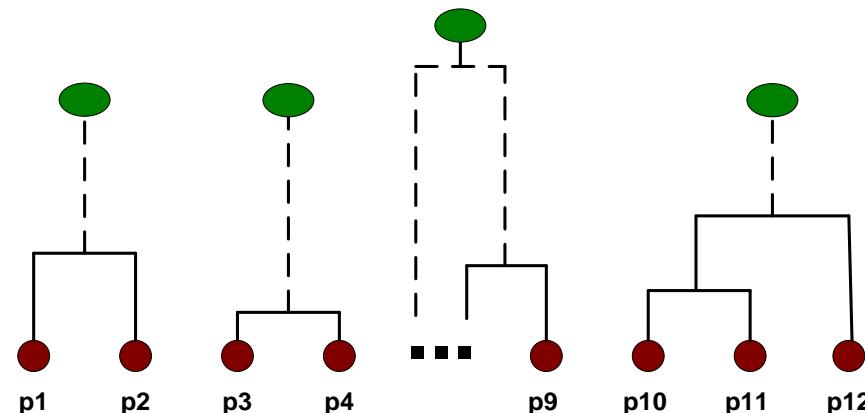
Merging clusters

- How to define the new (dis)similarities?



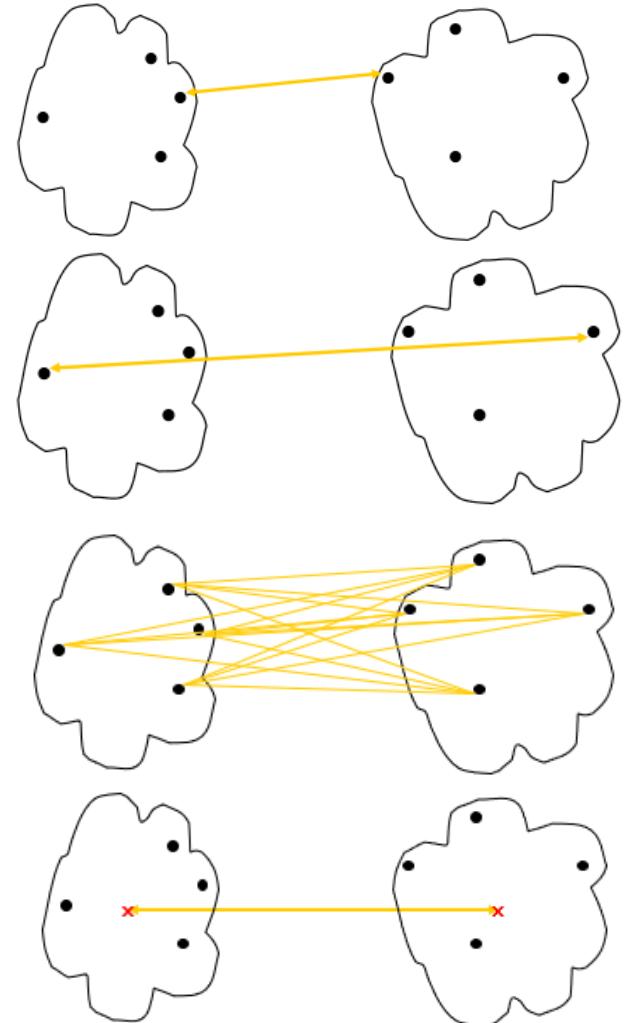
		C1	U C5	C3	C4
C1		?			
C2 U C5	?	?	?	?	
C3		?			
C4		?			

Proximity Matrix



Distance of clusters

- Single linkage (MIN): the distance of two clusters is the minimum pairwise distance of their data points
- Complete linkage (MAX): the distance of two clusters is the maximum pairwise distance of their data points
- Average linkage: taking the average of the distance values between pairs of cases
- Centroid method: the distance between two clusters is the distance between their centroids



Example (single vs. complete linkage)

- Be careful! Here these matrices are similarity matrices not distance matrices! Similarity and dissimilarity (or distance) work oppositely!

	I1	I2	I3	I4	I5
I1	1.00	0.90	0.10	0.65	0.20
I2	0.90	1.00	0.70	0.60	0.50
I3	0.10	0.70	1.00	0.40	0.30
I4	0.65	0.60	0.40	1.00	0.80
I5	0.20	0.50	0.30	0.80	1.00

Single

	I1	I2	I3	I4	I5
I1	1.00	0.90	0.10	0.65	0.20
I2	0.90	1.00	0.70	0.60	0.50
I3	0.10	0.70	1.00	0.40	0.30
I4	0.65	0.60	0.40	1.00	0.80
I5	0.20	0.50	0.30	0.80	1.00

Complete

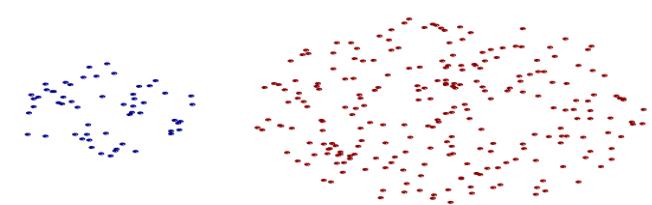
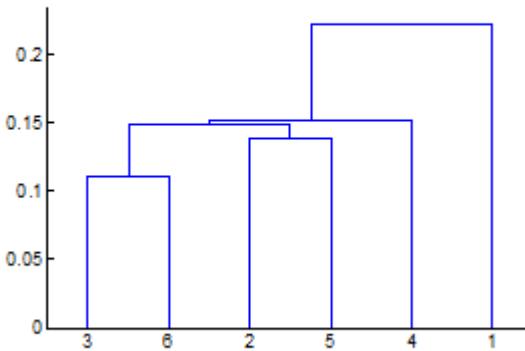
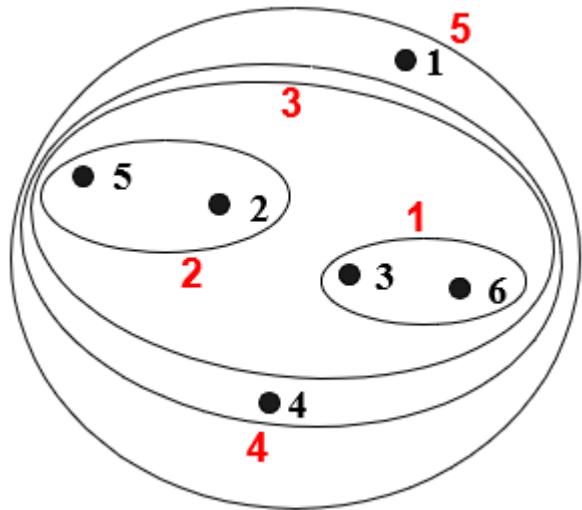
Problem

Use the following similarity matrix to perform single and complete linkage hierarchical clustering by drawing two dendrograms!

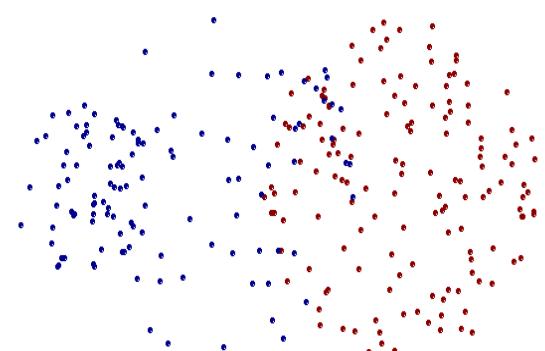
	1	2	3	4	5
1	1	0.15	0.6	0.15	0.95
2	0.15	1	0.5	0.2	0.2
3	0.6	0.5	1	0.05	0.7
4	0.15	0.2	0.05	1	0.85
5	0.95	0.2	0.7	0.85	1

Evaluating single linkage

- Handle elliptical shapes well
- Sensitive for noise and outliers



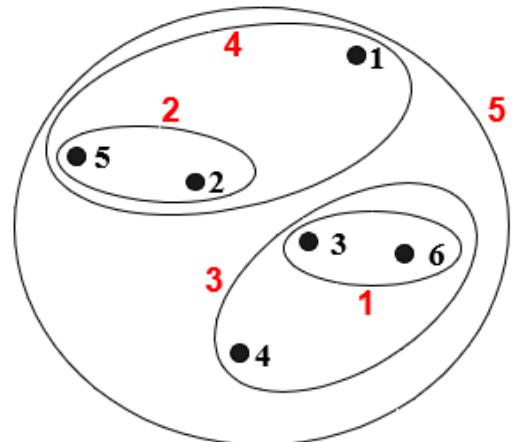
Two clusters



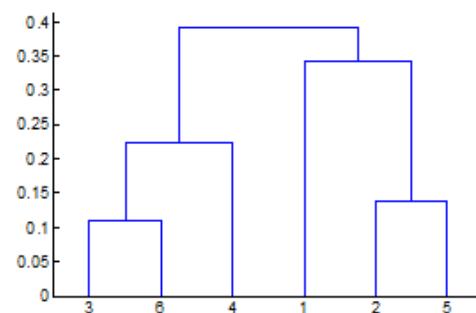
Two clusters

Evaluating complete linkage

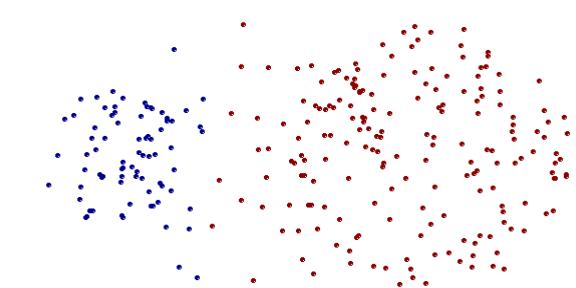
- Less sensitive for noise and outliers
- Tend to divide large clusters
- Tend to construct spherical clusters



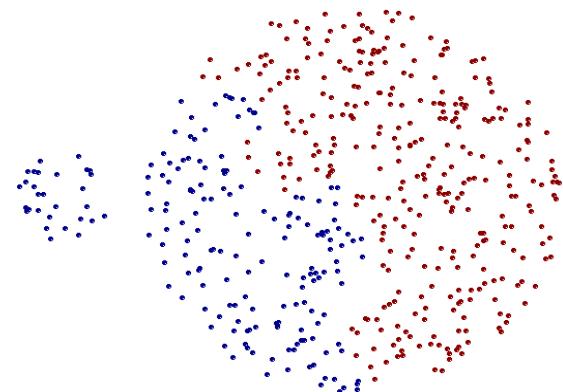
Nested Clusters



Dendrogram



Two clusters



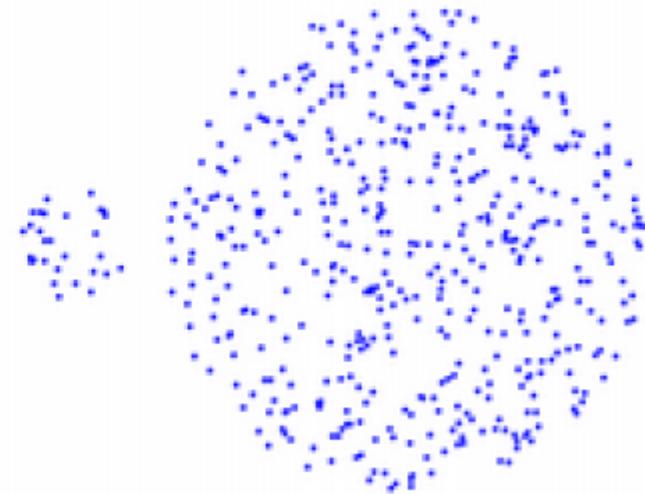
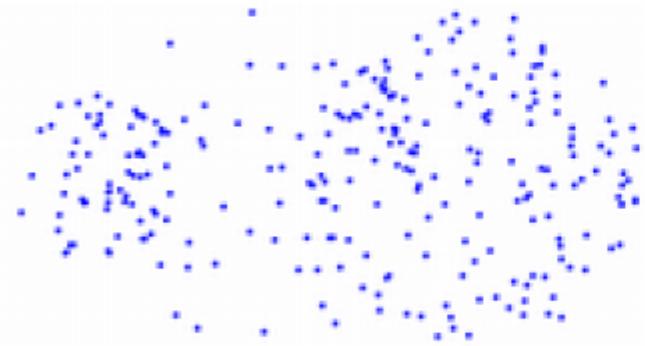
Two clusters

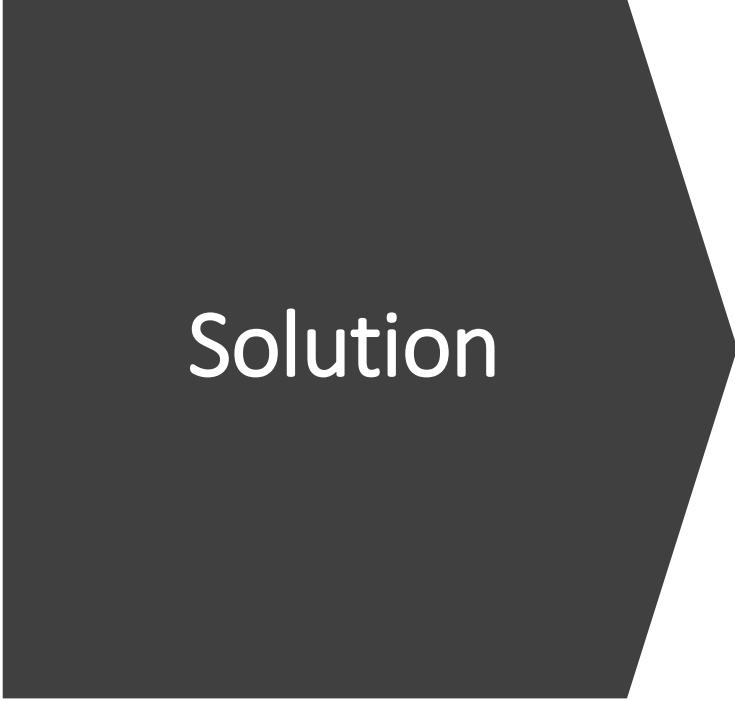
Evaluating hierarchical clustering

- If two clusters are merged during the algorithm, they remain merged: sensitive for noise and outliers
- Complexity: for n datapoints $O(n^3)$, since we have at most n steps with complexity $O(n^2)$ (determining the new distances)
- It outputs a hierarchy or taxonomy that is more informative than the unstructured set of flat clusters (returned by K-means)
- Easier to decide on the number of clusters by looking at the dendrogram
- Easy to implement

Problem

Consider the two-dimensional data sets below. How would the following clustering algorithms split the data into two clusters: k-means, single-linkage and complete-linkage hierarchical clustering?





Solution

Training Data

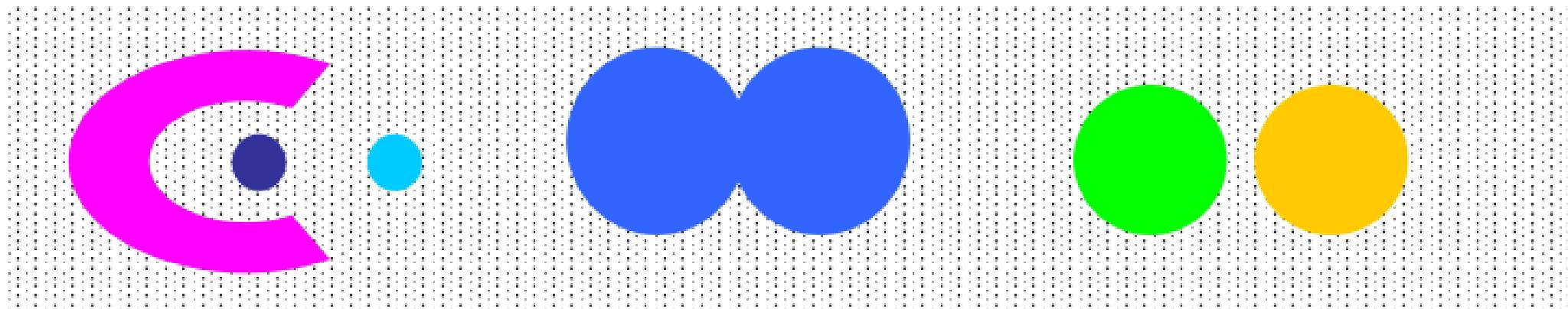
Test Data

Training Data

Test Data

Density based clustering

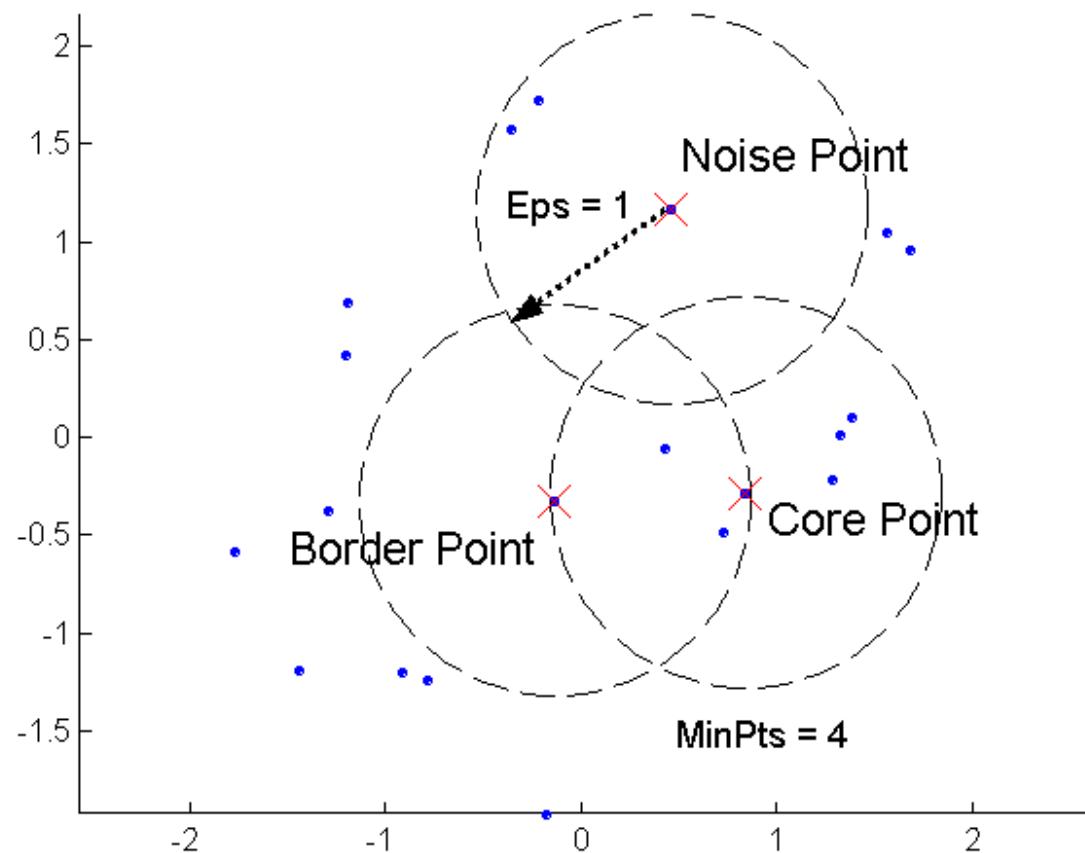
- Clusters are the areas with high density, between clusters the data is less dense
 - 6 density based clusters:



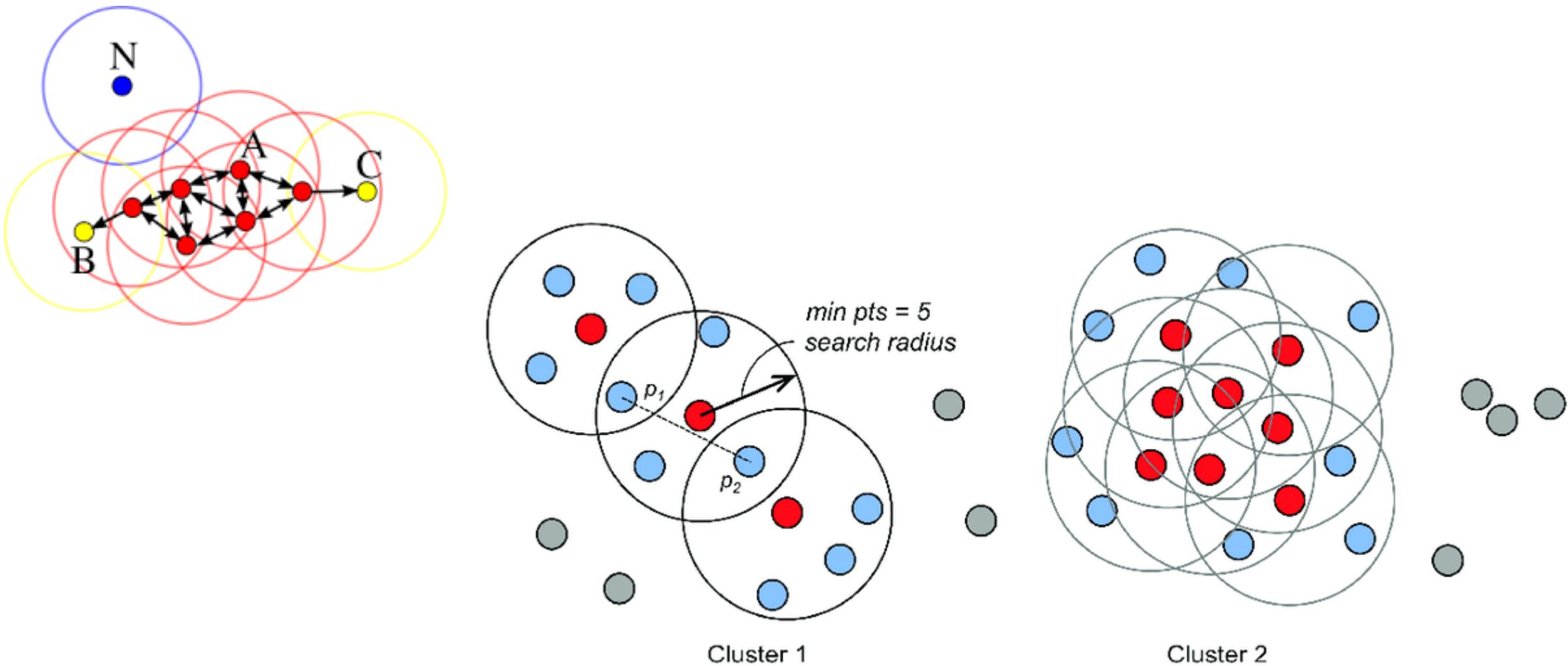
DBSCAN algorithm

- DBSCAN: **Density-based spatial clustering of applications with noise**
 - Density: number of data points within a certain radius (Eps)
 - **Core points**: data points that have at least $MinPts$ points within distance Eps of them (in their Eps -neighborhood)
 - The core points form the interior of the clusters
 - **Border points**: data points that have fewer than $MinPts$ points in their Eps -neighborhood, but they themselves are in the Eps -neighborhood of a core point
 - The border points form the borders of the clusters
 - **Noise points (outliers)**: data points that are neither core points nor border points
 - Noise points are not clustered

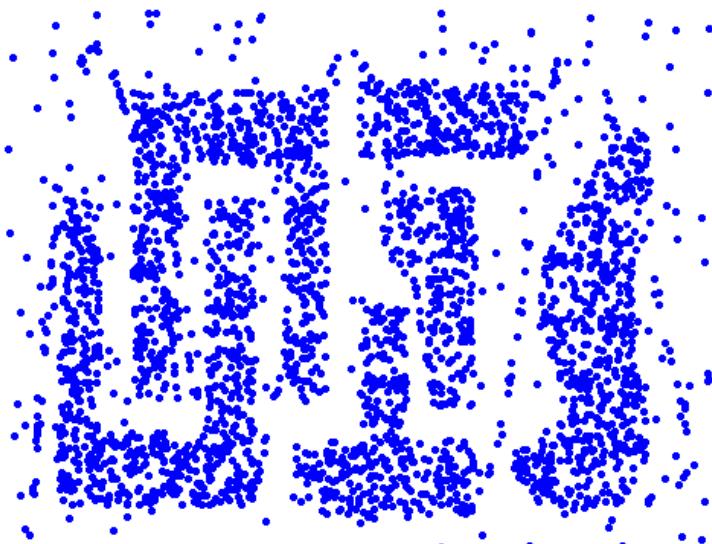
Core points, border points, noise points



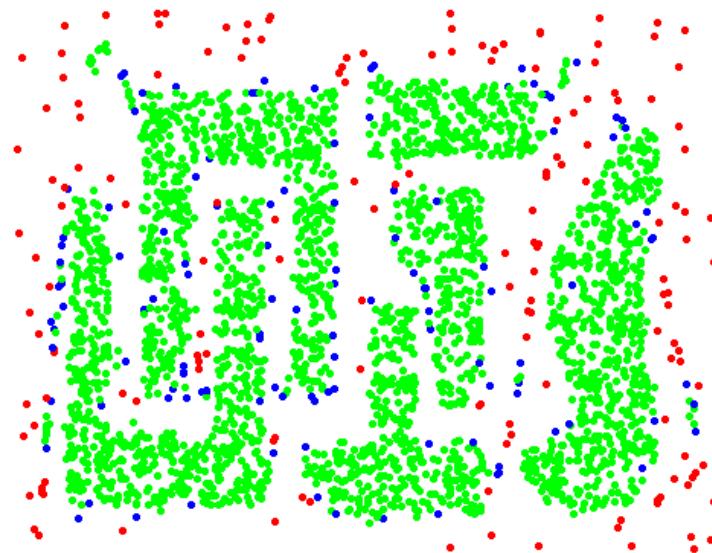
DBSCAN algorithm



DBSCAN – types of data points



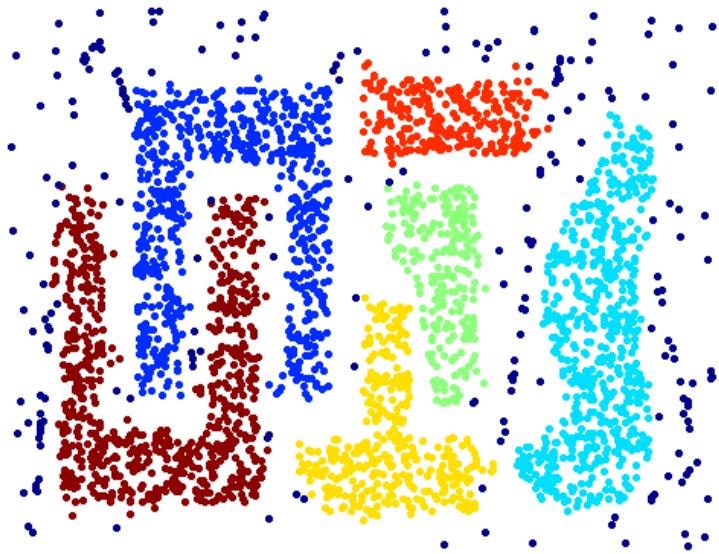
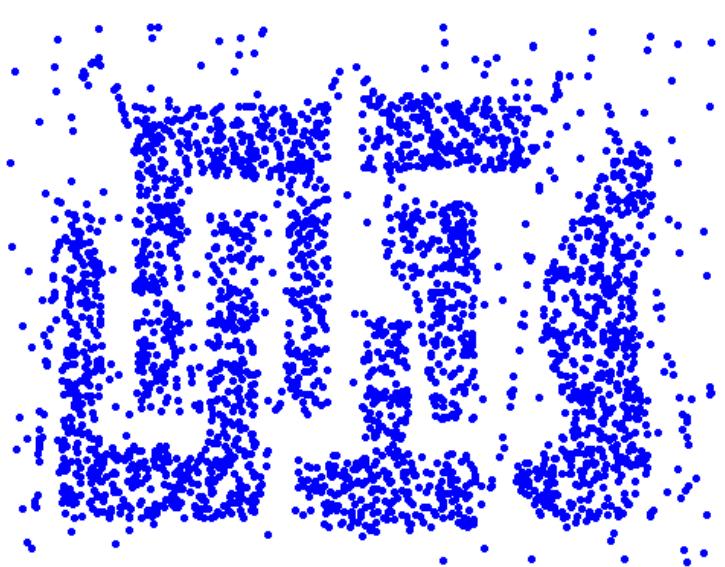
The data



Types of points: **core**, **border** and **noise**

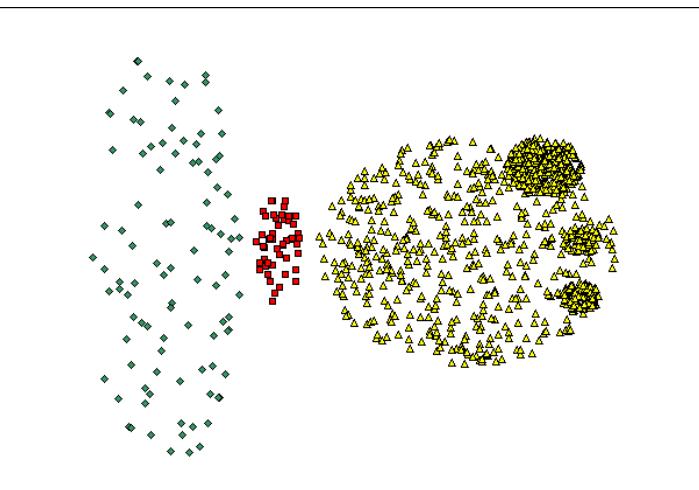
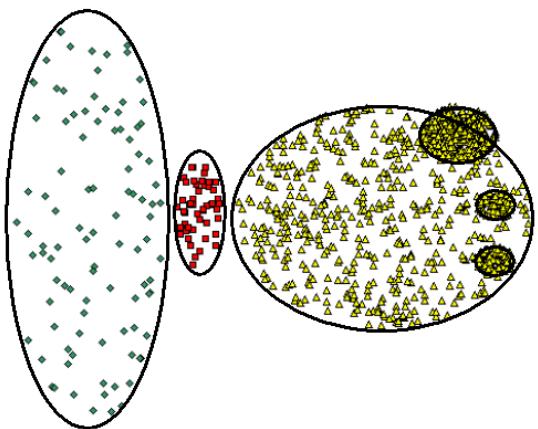
Eps = 10, MinPts = 4

DBSCAN - example

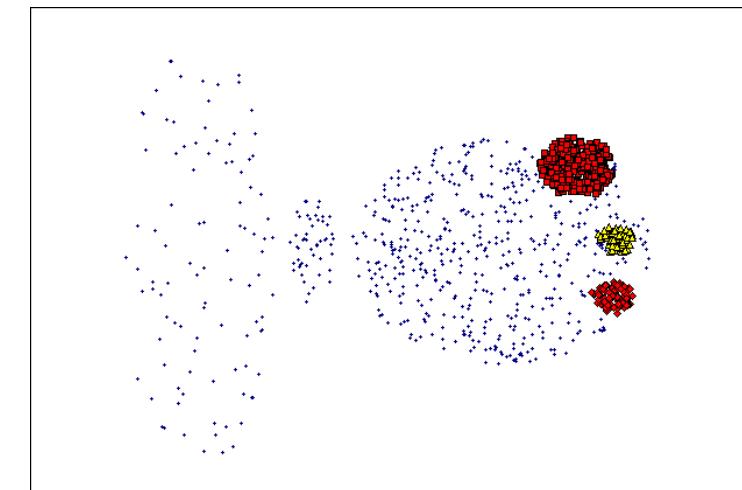


Clusters

DBSCAN - example



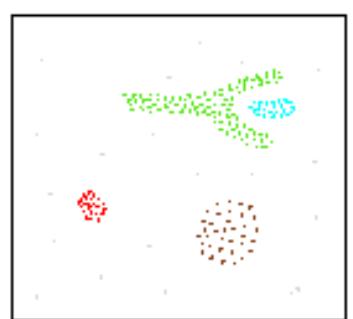
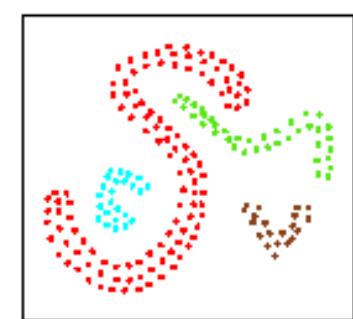
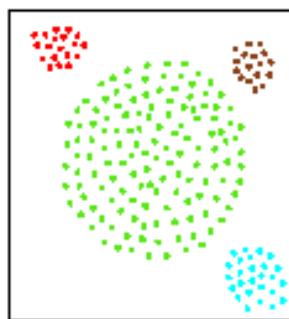
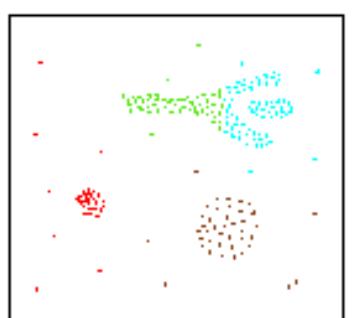
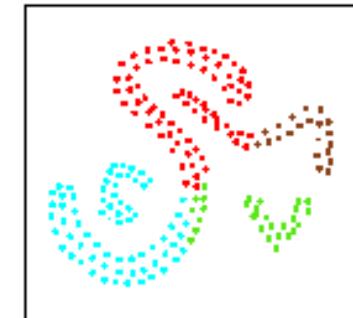
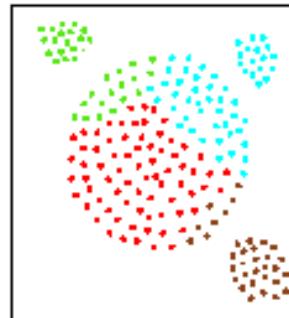
(MinPts=4, Eps=9.75)



(MinPts=4, Eps=9.62)

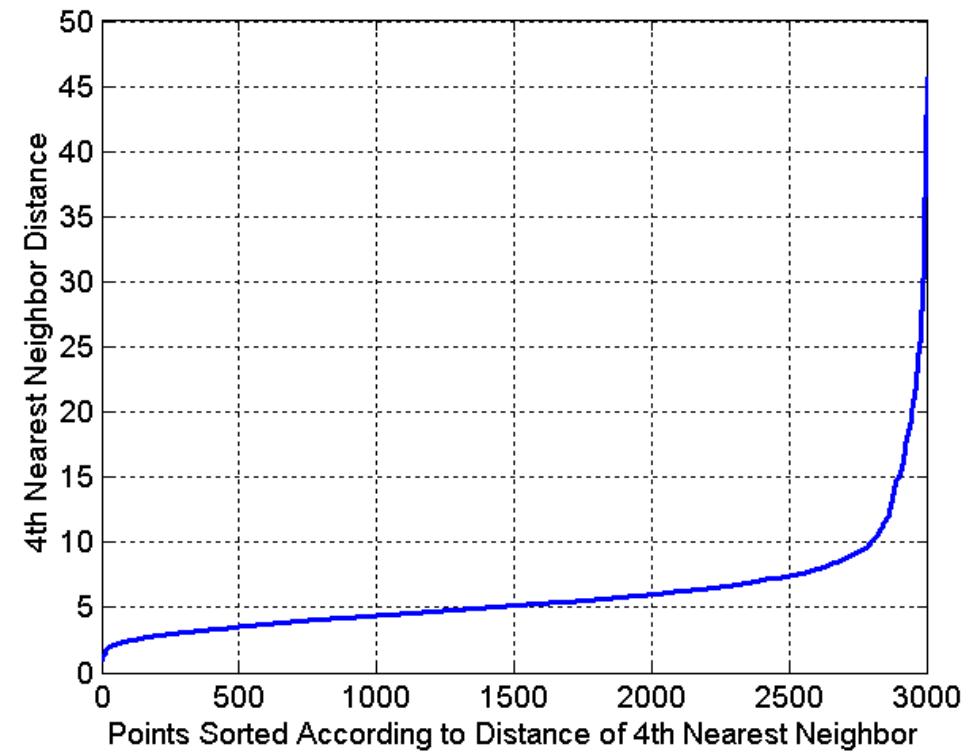
Evaluation of DBSCAN

- Insensitive to noise
- Not all the data points are clustered
- It can automatically handle outliers (by ignoring them)
- Treats clusters with differing size and shapes well
- Can't handle clusters with differing density
- Sensitive for the choice of hyperparameters ($MinPts$, Eps)



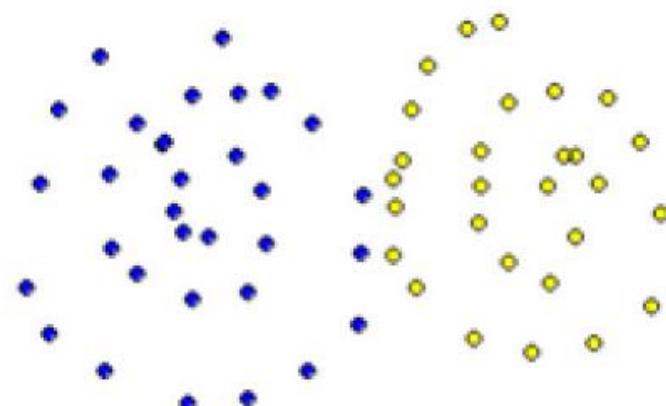
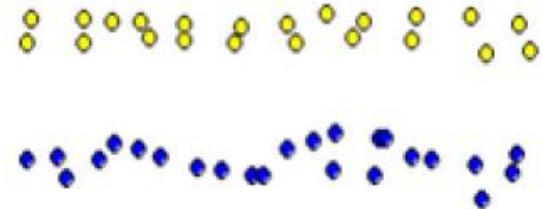
Determining the hyperparameters

- Plot the distance of the k th nearest neighbor for the data points
 - Idea: in dense areas the k th nearest neighbors are almost constants but the k th nearest neighbor of a noise point is much further away
 - Can we observe an angle in the graph?
 - The corresponding distance value is a reasonable choice for Eps



Problem

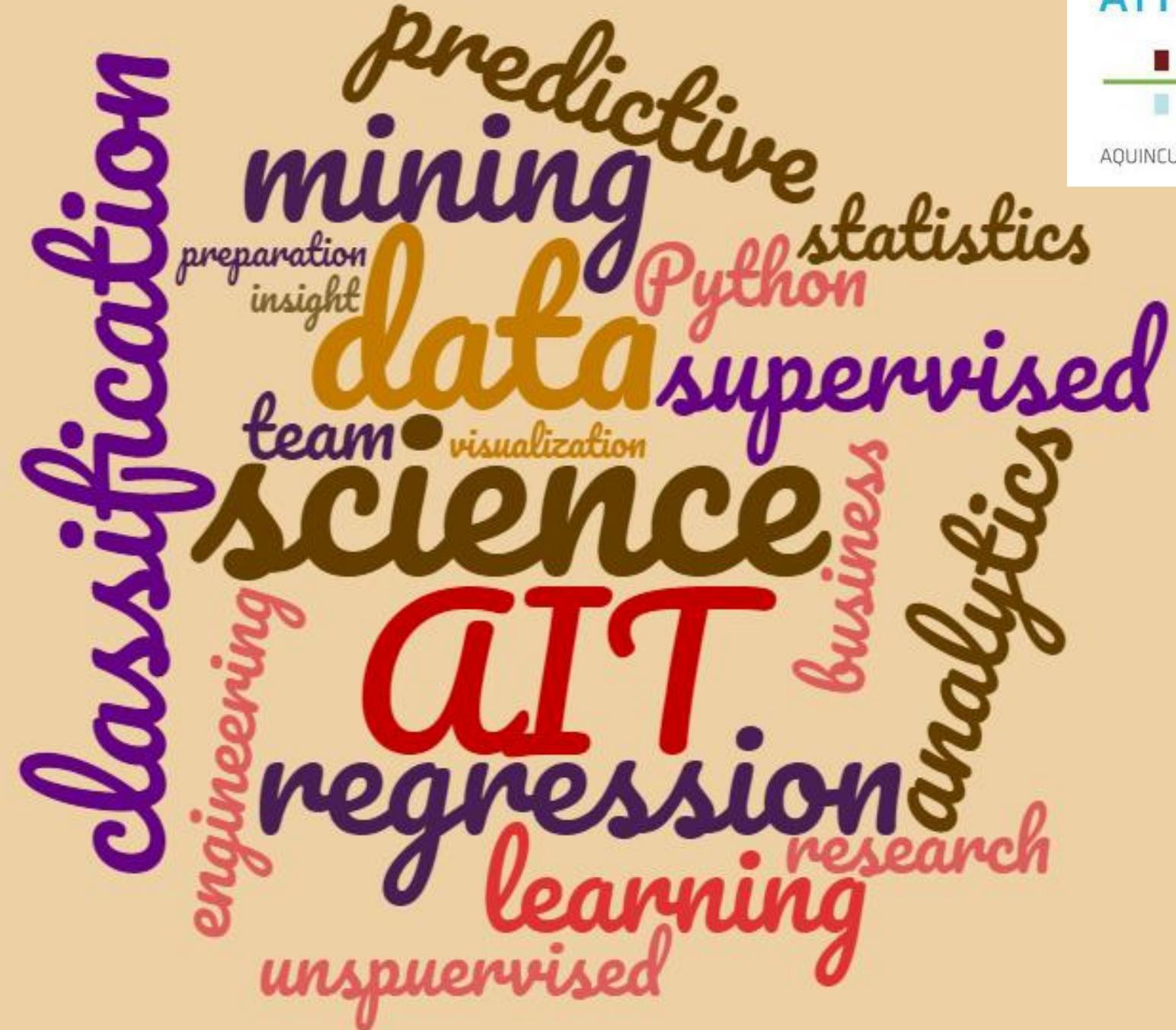
Which clustering algorithm would you use if the goal was to find the two natural clusters (marked by blue and yellow colors)? Consider the following algorithms: k-means, hierarchical clustering (both single and complete linkage) and DBSCAN.



Acknowledgement

- András Benczúr, Róbert Pálovics, SZTAKI-AIT, DM1-2
- Krisztián Buza, MTA-BME, VISZJV68
- Bálint Daróczy, SZTAKI-BME, VISZAMA01
- Judit Csima, BME, VISZM185
- Gábor Horváth, Péter Antal, BME, VIMMD294, VIMIA313
- Lukács András, ELTE, MM1C1AB6E
- Tim Kraska, Brown University, CS195
- Dan Potter, Carsten Binnig, Eli Upfal, Brown University, CS1951A
- Erik Sudderth, Brown University, CS142
- Joe Blitzstein, Hanspeter Pfister, Verena Kaynig-Fittkau, Harvard University, CS109
- Rajan Patel, Stanford University, STAT202
- Andrew Ng, John Duchi, Stanford University, CS229





Schedule of the semester

	<i>Monday midnight</i>	<i>Tuesday class</i>	<i>Friday class</i>
W1 (02/06)			
W2 (02/13)		HW1 out	
W3 (02/20)			
W4 (02/27)	HW1 deadline + TEAMS	HW2 out	
W5 (03/06)			PROJECT PLAN
W6 (03/13)	HW2 deadline	HW3 out	
W7 (03/20)			MIDTERM
SPRING BREAK		SPRING BREAK	SPRING BREAK
W8 (04/03)	HW3 deadline		GOOD FRIDAY
W9 (04/10)	MILESTONE 1		
W10 (04/17)		HW4 out	
W11 (04/24)			
W12 (05/01)	HW4 deadline		
W13 (05/08)	MILESTONE 2		
W14 (05/15)		FINAL	PROJECT presentations
W15 (05/22)		PROJECT presentations	

Recommender systems everywhere



Recommender systems

- Users/customers and products/items are given
 - We know some features of the users and/or items
 - We know the ratings given by the users for some items
 - The rating can be given explicitly in the form of likes/dislikes, evaluation scores
 - Or implicitly by the fact of purchase or clicks
- Goal: to assist the user by recommending useful/interesting items and assist the business to realize higher profit

	2			4	5	2.94*
	5		4			1
			5		2	2.48*
		1		5		4
			4			2
	4	5		1		1.12*

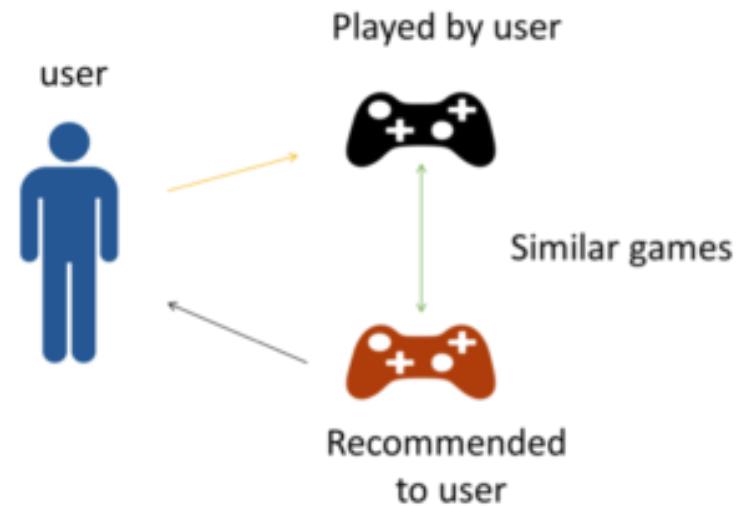


What information the recommendation is based on?

- User related data
 - Age
 - Location
 - Profession
- Item related data (e.g. regarding a movie)
 - Budget
 - Genre
 - Director, actors
- User-item ratings
 - Ratings given by the users for some items

Content based approach

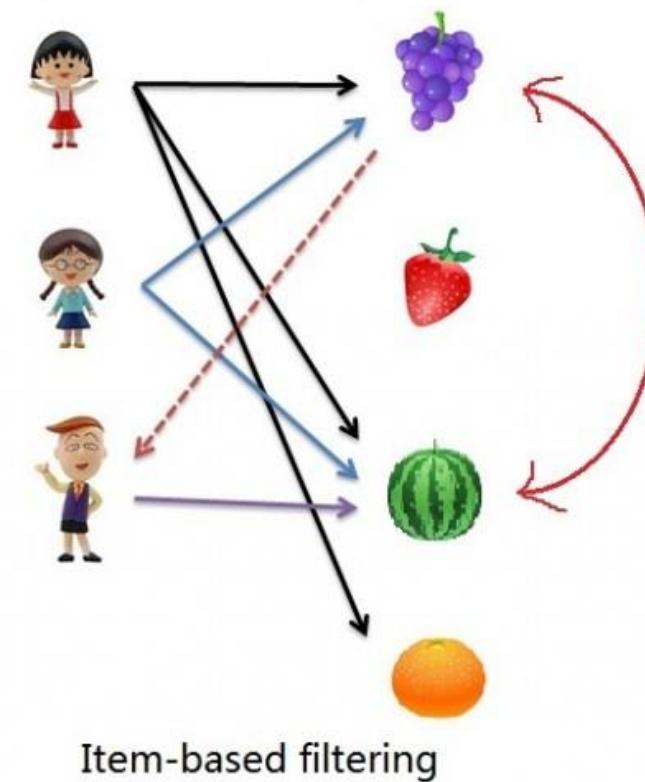
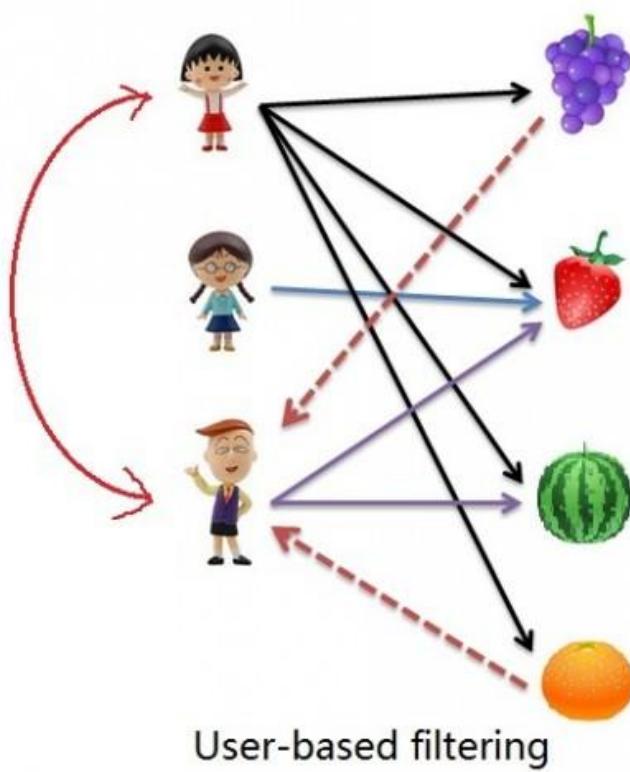
- The recommendation is based on the previous ratings of the users and on the similarity of items
- The items are characterized by some attributes
 - We define similarity between the items based on the attributes
- An item is recommended to a user if it is similar to an item that the user rated highly
- Limitations
 - What about new users?
 - Do not include attributes of users
 - What metric use for similarity between items?



Collaborative filtering

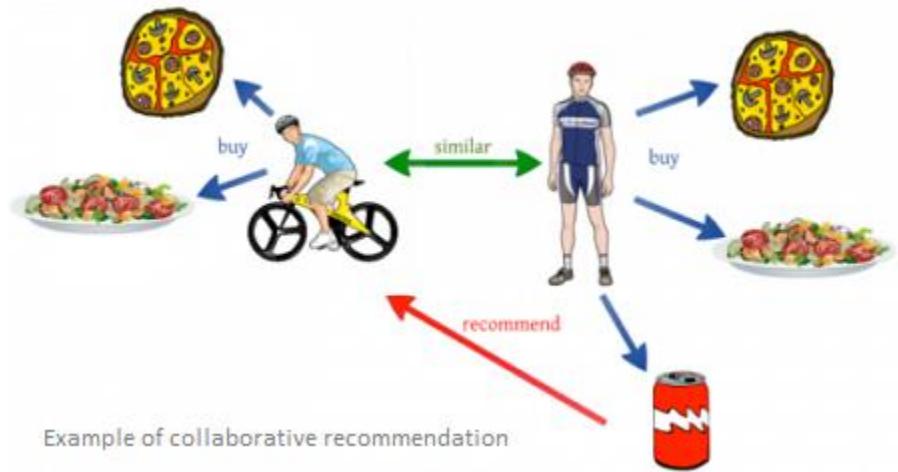
- Principle: Individual preferences are correlated
 - If Alice likes X and Y, and Bob likes X, Y and Z, then it is more likely that Alice likes Z
- Collaborative filtering does not rely on the characteristics of users or items, solely on the ratings
 - We don't need any additional information on the users/items
- User-based collaborative filtering
 - The recommendation is based on the ratings of users with similar taste
- Item-based collaborative filtering
 - Prediction is based on the similarity between items using user's ratings on those items
- Hybrid method
 - Combine the two approaches

Collaborative filtering II.



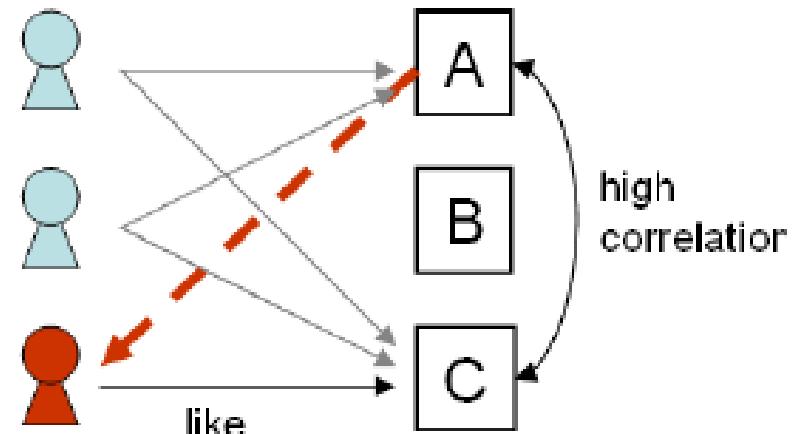
User-based collaborative filtering

- Principle: A user likes the items that are liked by users who are similar to him/her
 - Similarity is based on the ratings
- The predicted rating may be determined by a kNN approach
 - The predicted rating of a given item from a certain user is the average of the ratings of the k most similar users who rated the given item



Item-based collaborative filtering

- We are looking for similar items
 - The similarity is based on the ratings of users
- We recommend items that are similar to the ones that were liked by the user
- It is advantageous if there are more users than items



The model of collaborative filtering

- Collaborative filtering is solely based on the ratings stored in the rating matrix (user-item interaction matrix)
- This matrix is a sparse matrix with lots of missing entries

$$X = \begin{bmatrix} 1 & & 3 \\ & 2 & 5 \\ 3 & & 5 \\ 4 & & 4 \end{bmatrix}$$

Rows correspond to users

Columns correspond to items

The entries correspond to the known ratings of the items given by the users

How to fill in missing entries?

- Goal: predicting the missing entries, i.e. predicting the ratings of the users
 - If the predicted rating is high, we recommend the item

	Movie 1	Movie 2	Movie 3	Movie 4	Movie 5	...
User 1	5	?	1	?	?	...
User 2	?	?	5	?	4	...
User 3	5	4	2	?	?	...
User 4	?	3	?	2	5	...
User 5	1	?	5	?	4	...
User 6	5	4	?	?	2	...
...

$$X = \begin{bmatrix} 1 & ? & ? & 3 \\ ? & 2 & 5 & ? \\ ? & 3 & ? & 5 \\ 4 & ? & 4 & ? \end{bmatrix}$$

Nearest neighbor based methods

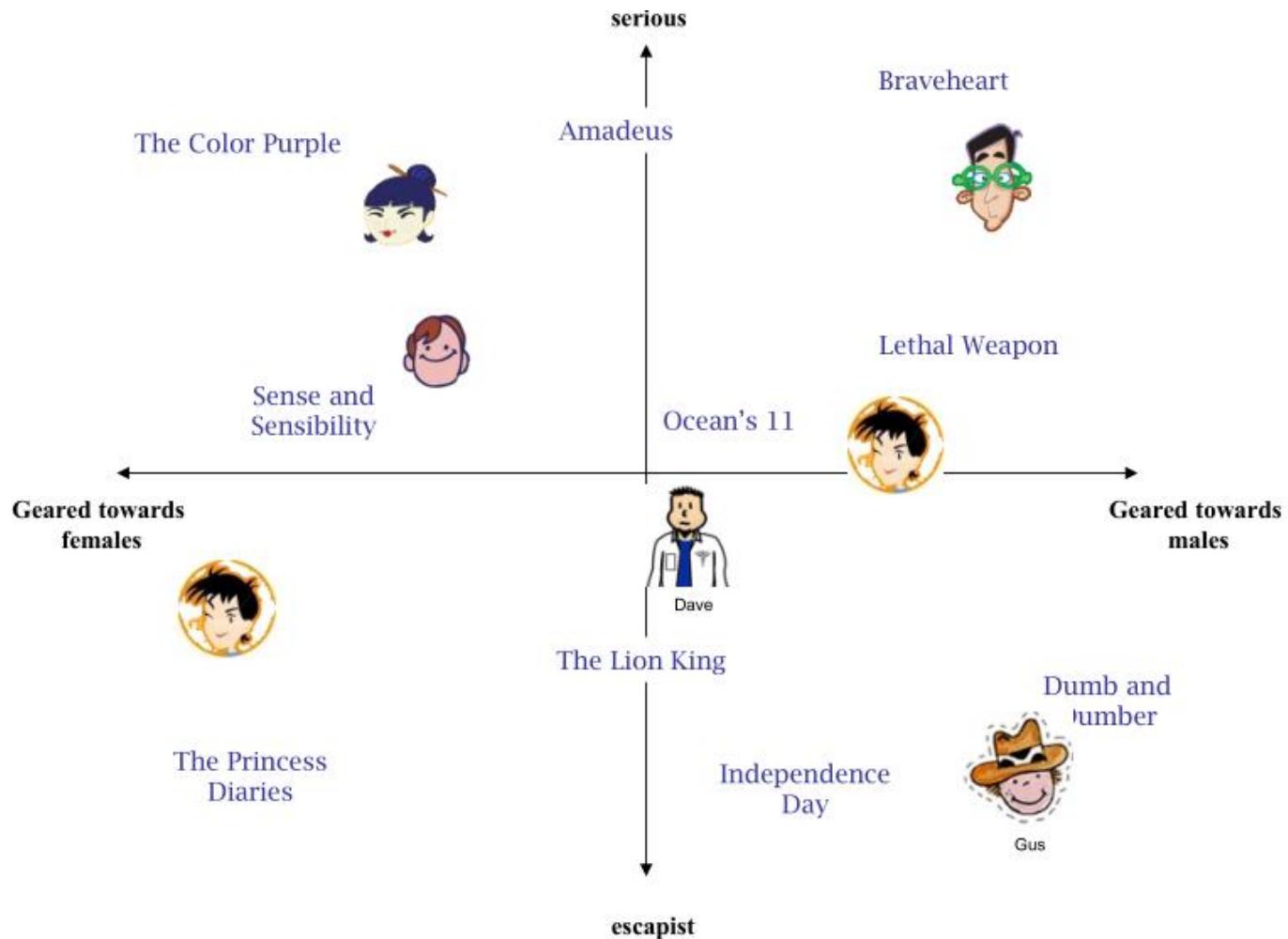
- User-based and items-based approaches are both possible
- User-based approach:
 - User is represented by an (incomplete) row vector
 - We consider his/her k nearest neighbors (with a chosen dissimilarity) who rated the given item
 - We take the (weighted) average of the ratings of k users
 - This is basically a kNN regression
 - Other regression methods are also possible

	Movie 1	Movie 2	Movie 3	Movie 4	Movie 5	...
User 1	5	?	1	?	?	...
User 2	?	?	5	?	4	...
User 3	5	4	2	?	?	...
User 4	?	3	?	2	5	...
User 5	1	?	5	?	4	...
User 6	5	4	?	?	2	...
...

Latent factors

- We transform the items and users to a smaller dimensional space spanned by some latent factors
- Idea: The ratings of the items depend on these latent factors
- For each item and rating, the model assigns weights to the latent factors
 - The weights are extracted from the user-item interaction matrix
- The latent factors (e.g. for movies) can be interpreted as how romantic, how adventurous they are

Illustrating latent factors



Matrix factorization

- We estimate the rating matrix X as the product of two matrices
- Based on the known entries of X we are looking for U and V in such a way that their product approximates the known elements of X as closely as possible

$$\begin{matrix} & \begin{matrix} 2 & 1 \\ 2 & 2 \\ 3 & 2 \\ 1 & 1 \\ \dots & \dots \end{matrix} & \times & \begin{matrix} 2 & 2 & 1 & 3 & \dots \\ 1 & 0 & 3 & 3 & \dots \end{matrix} & \approx & \begin{matrix} 5 & ? & 4 & ? & \dots \\ ? & 4 & ? & ? & \dots \\ ? & 5 & 4 & ? & \dots \\ 4 & ? & 4 & 5 & \dots \\ \dots & \dots & \dots & \dots & \dots \end{matrix} \\ U & & V & & & X \end{matrix}$$

Matrix factorization

- Number of latent factors: K
- Hypothesis : $X \approx UV$, $U \in \mathbb{R}^{m \times K}, V \in \mathbb{R}^{K \times n}$

$$U = \begin{bmatrix} -u_1^T & - \\ \vdots & \\ -u_m^T \end{bmatrix}, \quad V = \begin{bmatrix} | & & | \\ v_1 & \dots & v_n \\ | & & | \end{bmatrix}$$

- Cost function (MSE):

$$\sum_{i,j} \left(x_{i,j} - \sum_{k=0}^K u_{i,k} v_{k,j} \right)^2$$

- Optimization method: (stochastic) gradient descent method
- We can add the usual regularization term: $+ \lambda (\sum_{i,j} (u_{i,j}^2 + v_{i,j}^2))$

Netflix



Most Loved Movies

	Avg rating	Count
The Shawshank Redemption	4.593	137812
Lord of the Rings :The Return of the King	4.545	133597
The Green Mile	4.306	180883
Lord of the Rings :The Two Towers	4.460	150676
Finding Nemo	4.415	139050
Raiders of the Lost Ark	4.504	117456

Most Rated Movies

Miss Congeniality
Independence Day
The Patriot
The Day After Tomorrow
Pretty Woman
Pirates of the Caribbean

Highest Variance

The Royal Tenenbaums
Lost In Translation
Pearl Harbor
Miss Congeniality
Napolean Dynamite
Fahrenheit 9/11



Netflix Prize

- Build a recommendation system
- Started in 2006
- Finished in 2009
- Prize: 1 million \$
- The most famous data mining competition
- Goal: to improve the recommendation system of Netflix by 10%
- More than 2700 R&D teams started to work on the problem

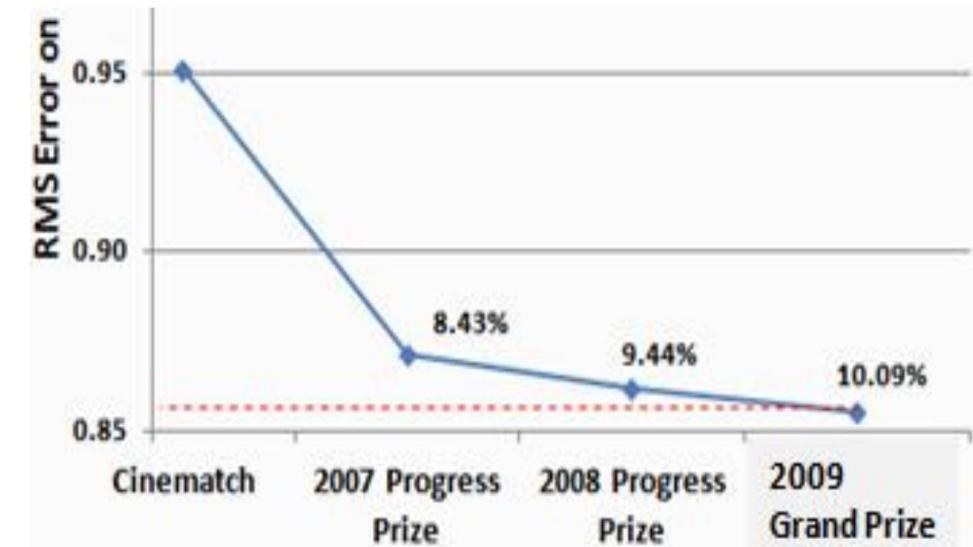
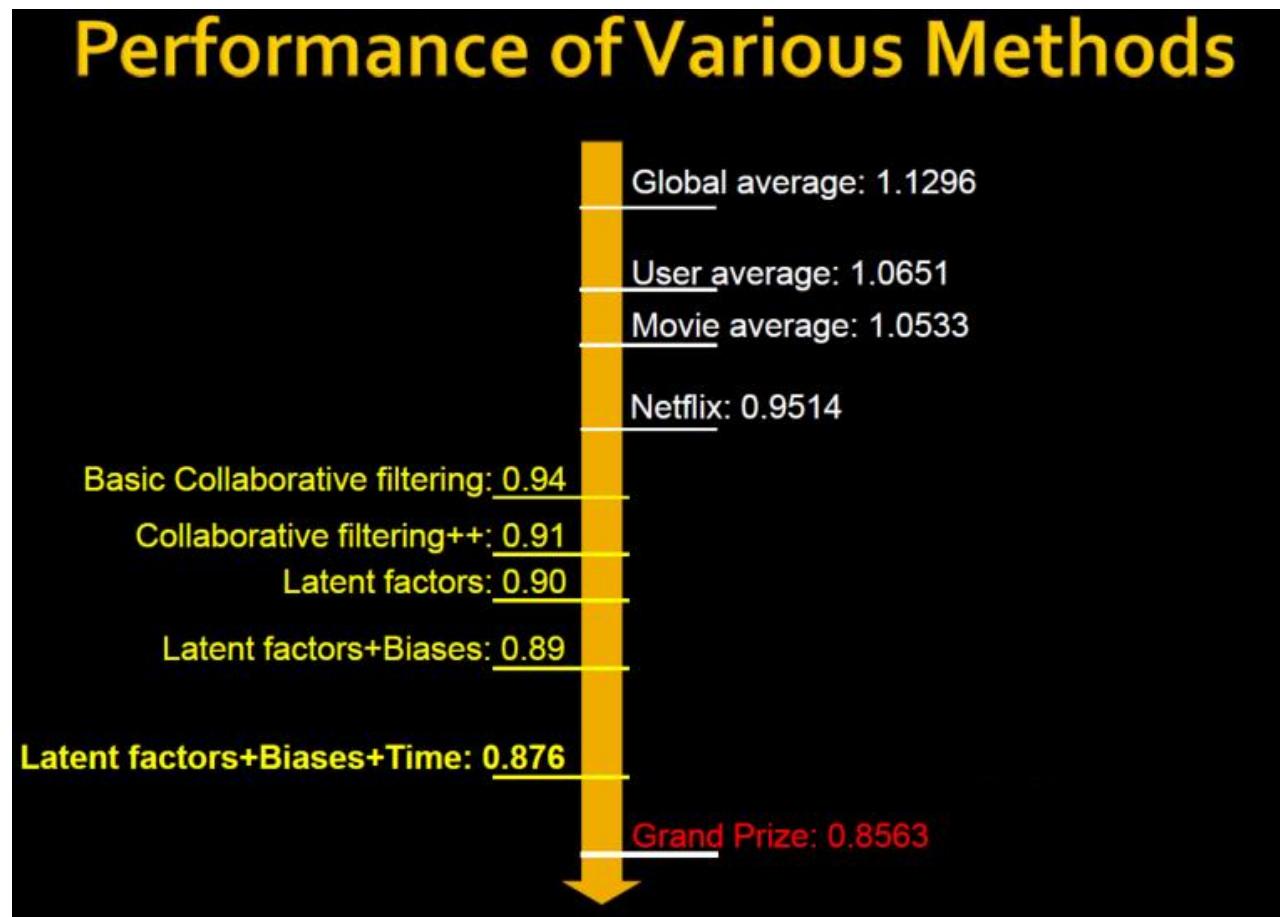


Netflix data

- Training data:
 - 100 million movie ratings
 - 18 thousand movies and 480 thousand users
 - In average \sim 5600 ratings/movie
 - In average \sim 208 ratings/user
 - 6 years data (2000-2005)
 - Ratings are integers between 1 and 5
- Validation data:
 - The last few ratings for every user (2.8 million ratings)
 - Evaluation criteria: RMSE
 - Using the algorithm of Netflix - RMSE: 0.9514
 - 10% improvement would be an RMSE of 0.856
 - The true labels of the validation data are naturally hidden from the teams
 - The teams upload their predicted labels and the system evaluate their results



Improving results...



The leaderboard 30 days before the end date

The screenshot shows the Netflix Prize Leaderboard page. At the top, there's a red header with the Netflix logo and a yellow banner featuring three stars and the text "Netflix Prize". Below the banner is a navigation bar with links: Home, Rules, Leaderboard, Register, Update, Submit, and Download. The main title "Leaderboard" is in blue at the top left. To its right is a button labeled "Display top 20 leaders." A horizontal line separates the header from the table below.

Rank	Team Name	Best Score	Improvement	Last Submit Time
1	BellKor's Pragmatic Chaos	0.8558	10.05	2009-06-26 18:42:37
Grand Prize - RMSE <= 0.8563				
2	PragmaticTheory	0.8582	9.80	2009-06-25 22:15:51
3	BellKor in BigChaos	0.8590	9.71	2009-05-13 08:14:09
4	Grand Prize Team	0.8593	9.68	2009-06-12 08:20:24
5	Dace	0.8604	9.56	2009-04-22 05:57:03
6	BigChaos	0.8613	9.47	2009-06-23 23:06:52
Progress Prize 2008 - RMSE = 0.8616 - Winning Team: BellKor in BigChaos				
7	BellKor	0.8620	9.40	2009-06-24 07:16:02
8	Gravity	0.8634	9.25	2009-04-22 18:31:32
9	Opera Solutions	0.8638	9.21	2009-06-26 23:18:13
10	BruceDengDaiCiYiYou	0.8638	9.21	2009-06-27 00:55:55
11	pengpengzhou	0.8638	9.21	2009-06-27 01:06:43
12	xvector	0.8639	9.20	2009-06-26 13:49:04
13	xiangliang	0.8639	9.20	2009-06-26 07:47:34

The last 30 days

- A new team is formed called Ensemble
 - By the merger of a few teams from the top 10
 - They combine their achievements and try to beat the leader, BellKor
- BellKor
 - Manage to have further little improvements
 - Also realize that Ensemble is a dangerous competitor
- Strategy
 - Both teams have their eyes on the leaderboard
 - The only way they can check whether a new method improves the result is to upload, but then the other teams also get informed



The last 24 hours

- New rule at the end: 1 upload/day
- It means that in the last 24 hours a team can only upload once
- 24 hours to the end, BellKor realizes that the Ensemble team has better result
- Crazy 24 hours have started, both teams try to hurry
- 1 hour before the deadline both teams are ready
 - At what time should they upload their results?
 - Bellkor upload their results 40 min before the deadline
 - Ensemble upload their results 20 min before the deadline



And everybody is waiting...

The final results

NETFLIX

Netflix Prize

COMPLETED

Home Rules Leaderboard Update Download

Leaderboard

Showing Test Score. [Click here to show quiz score](#)

Display top 20 leaders.

Rank	Team Name	Best Test Score	% Improvement	Best Submit Time
1	BellKor's Pragmatic Chaos	0.8567	10.06	2009-07-26 18:18:28
2	The Ensemble	0.8567	10.06	2009-07-26 18:38:22
3	Grand Prize Team	0.8568	9.99	2009-07-26 18:38:22
4	Opera Solutions and Vandelay United	0.8588	9.84	2009-07-10 01:12:31
5	Vandelay Industries!	0.8591	9.81	2009-07-10 00:32:20
6	PragmaticTheory	0.8594	9.77	2009-06-24 12:06:56
7	BellKor in BigChaos	0.8601	9.70	2009-05-13 08:14:09
8	Dace	0.8612	9.59	2009-07-24 17:18:43
9	Feeds2	0.8622	9.48	2009-07-12 13:11:51
10	BioChaos	0.8623	9.47	2009-04-07 12:33:59
11	Opera Solutions	0.8623	9.47	2009-07-24 00:34:07
12	BellKor	0.8624	9.46	2009-07-26 17:19:11

Progress Prize 2008 - RMSE = 0.8627 - Winning Team: BellKor in BigChaos

13	xiangliang	0.8642	9.27	2009-07-15 14:53:22
14	Gravity	0.8643	9.26	2009-04-22 18:31:32
15	Ces	0.8651	9.18	2009-06-21 19:24:53
16	Invisible Ideas	0.8653	9.15	2009-07-15 15:53:04
17	Just a guy in a garage	0.8662	9.06	2009-05-24 10:02:54
18	J Dennis Su	0.8666	9.02	2009-03-07 17:16:17
19	Craig Carmichael	0.8666	9.02	2009-07-25 16:00:54
20	acmehill	0.8668	9.00	2009-03-21 16:20:50

Progress Prize 2007 - RMSE = 0.8627 - Winning Team: BellKor in BigChaos

jure leskovec, Stanford C246: Mining Massive Datasets

And the prize goes to...

Million \$ Awarded Sept 21st 2009



Was it worth it?



The screenshot shows a web browser window displaying a blog post from [techblog.netflix.com](https://techblog.netflix.com/2012/04/netflix-recommendations-beyond-5.html). The title of the post is "3 Years Later...". The main text of the post discusses the evaluation of new recommendation methods and concludes with the following quote:

"We evaluated some of the new methods offline but the additional accuracy gains that we measured did not seem to justify the engineering effort needed to bring them into a production environment."

The post is dated Friday, April 6, 2012, and is written by Xavier Amatriain and Justin Basilio. It mentions the Netflix Prize competition and the transition from Cinematch to Matrix Factorization. The sidebar on the right includes links to other Netflix blogs like "Netflix India Blog" and "Netflix DACH Blog".

Acknowledgement

- András Benczúr, Róbert Pálovics, SZTAKI-AIT, DM1-2
- Krisztián Buza, MTA-BME, VISZJV68
- Bálint Daróczy, SZTAKI-BME, VISZAMA01
- Judit Csima, BME, VISZM185
- Gábor Horváth, Péter Antal, BME, VIMMD294, VIMIA313
- Lukács András, ELTE, MM1C1AB6E
- Tim Kraska, Brown University, CS195
- Dan Potter, Carsten Binnig, Eli Upfal, Brown University, CS1951A
- Erik Sudderth, Brown University, CS142
- Joe Blitzstein, Hanspeter Pfister, Verena Kaynig-Fittkau, Harvard University, CS109
- Rajan Patel, Stanford University, STAT202
- Andrew Ng, John Duchi, Stanford University, CS229

