

# Machine Learning and Data Mining

## V00: Organization

Lecturer: Dr. Andreas Weiler



# About me

- Dr. Andreas Weiler (short: wele)
- Email: [andreas.weiler@zhaw.ch](mailto:andreas.weiler@zhaw.ch)
- Office: InIT, TD O3.12 (Obere Kirchgasse 2)
- Senior Lecturer ZHAW since June 2018
- Director of Studies in Data Science
- CV:
  - 2018: Lecturer (Data Warehousing and OLAP), University of Konstanz
  - 2016-2018: Lead Data Scientist / Data Engineer, coliquio GmbH, Konstanz
  - 2016: Dr. rer. nat., Computer Science, University of Konstanz
    - Topics: Data Streaming, Twitter Analysis, (Visual) Event Detection
  - 2010: Master of Science, Information Engineering, University of Konstanz
    - Topic: Unstructured Data, XML Databases, Client/Server Architecture
  - 2006: Bachelor of Science, Information Engineering, University of Konstanz
    - Topics: Data Warehousing, (Visual) OLAP, Business Intelligence

# Organization

- Material, Organizational Stuff, Labs, ... can be found on Moodle
  - URL: <https://moodle.zhaw.ch/>
  - module description needs to be accepted

Machine Learning und Data Mining FS2022

Dashboard / Meine Kurse / MLDMFS2022

<b>General</b>	<input checked="" type="checkbox"/>
Ankündigungen	<input checked="" type="checkbox"/>
Modulvereinbarung	<input checked="" type="checkbox"/>
<b>Introduction Python / Pandas</b>	<input checked="" type="checkbox"/>
Notebooks	<input checked="" type="checkbox"/>
<b>Slides</b>	<input checked="" type="checkbox"/>
Lectures Dömer	<input checked="" type="checkbox"/>



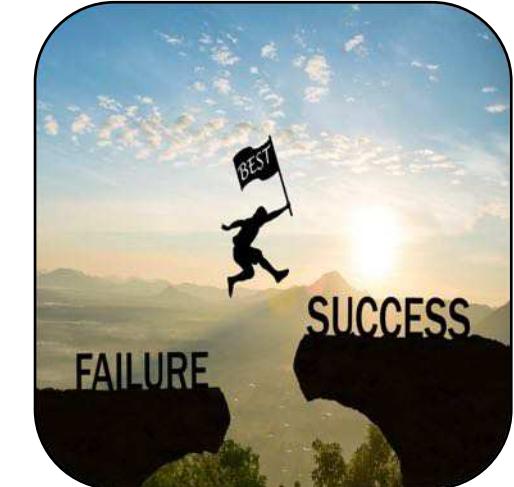
# Organization

- Lectures
  - the lecture begins and ends on time
  - notebooks etc. are just used for teaching
- Extent
  - $2 + 2 \text{ ECTS} = 90 \text{ mins lecture}$   
 $90 \text{ mins lab}$   
ca.180 mins self-study a week
- Module Mark
  - end-of-semester (oral or written) exam (90 mins) → 80%
  - 2 graded labs (each 2 week) → 20%



# Organization

- 2 graded labs (20% of final grade):
  - 2 weeks long
  - build groups of two students
  - plagiarism: If we catch you, you fail the entire module (or worse). We run plagiarism checks.
  - deadlines are deadlines. Even 1 second late is too late. Only exception: serious illness. If you submit a deliverable later than the deadline, you will receive 0 marks for that deliverable
  - let us know about (potential) problems as soon as possible. Telling us one day before a deadline that you had a cold three weeks ago, is no excuse.



# Literature

- Maschinelles Lernen: Grundlagen und Algorithmen in Python, J. Frochte
- Data Mining – Practical Machine Learning Tools and Techniques, Ian H. Witten
- Machine Learning, Tom Mitchell, McGraw-Hill Education Ltd, 1997
- Machine Learning: The Art and Science of Algorithms that Make Sense of Data, Peter Flach
- Guide to Intelligent Data Analysis: How to Intelligently Make Sense of Real Data: Making Practical Sense of Real Data, Berthold M. R., Borgelt C., Höppner F., Klawonn F.



# Further Material

- Online Courses:
  - <https://www.udemy.com/machinelearning/>
  - <https://www.coursera.org/learn/machine-learning>
  - <https://www.edx.org/course/foundations-of-data-science-computational-thinking>
- The World Wide Web
  - Kaggle
  - KD Nuggets
  - Google AI Blog
  - Vaultanalytics <https://vaulitanalytics.com/ai-applications/>
  - O'Reilly AI Newsletter <https://www.oreilly.com/ai/newsletter.html>
  - TechCrunch
  - Machine Learning Yearning, Andrew Ng, Book In Progress  
<http://www.mlyearning.org/>

# Course Content IT20a\_WIN

Week	Lecture	Lab
KW08	Introduction	Lab 01
KW09	Data Processing Part 01	Lab 02
KW10	Carnival Monday	
KW11	Data Processing Part 02	Lab 03
KW12	Evaluation	Lab 04
KW13	Recommender Systems	<b>Graded Lab 01</b> <span style="border: 2px solid red; padding: 2px;">IMPORTANT</span>
KW14	Association Rules	<b>Graded Lab 01</b> <span style="border: 2px solid red; padding: 2px;">IMPORTANT</span>
KW15	Linear and Logistic Regression	Lab 05
KW16	Easter Monday	
KW17	Support Vector Machines	<b>Graded Lab 02</b> <span style="border: 2px solid red; padding: 2px;">IMPORTANT</span>
KW18	Decision Trees and Naïve Bayes	<b>Graded Lab 02</b> <span style="border: 2px solid red; padding: 2px;">IMPORTANT</span>
KW19	Partition Clustering	Lab 06
KW20	Hierarchical and Density Clustering	Lab 07
KW21	Neural Networks and Closing	Review and Questions for Exam

# Questions and Answers



# Machine Learning and Data Mining

## V01: Introduction

Lecturer: **Dr. Andreas Weiler**



based on material of  
Andreas Weiler (wele),  
Mark Cieliebak (ciel), and  
Thilo Stadelmann (stdm)

# Learning Objectives

- fundamental understanding of the topics data mining, machine learning, deep learning and reinforcement learning
- process of a typical data science project
- understanding of the different types of machine learning
- examples of data science projects from the InIT

“We are drowning in data,  
but starving for knowledge!”

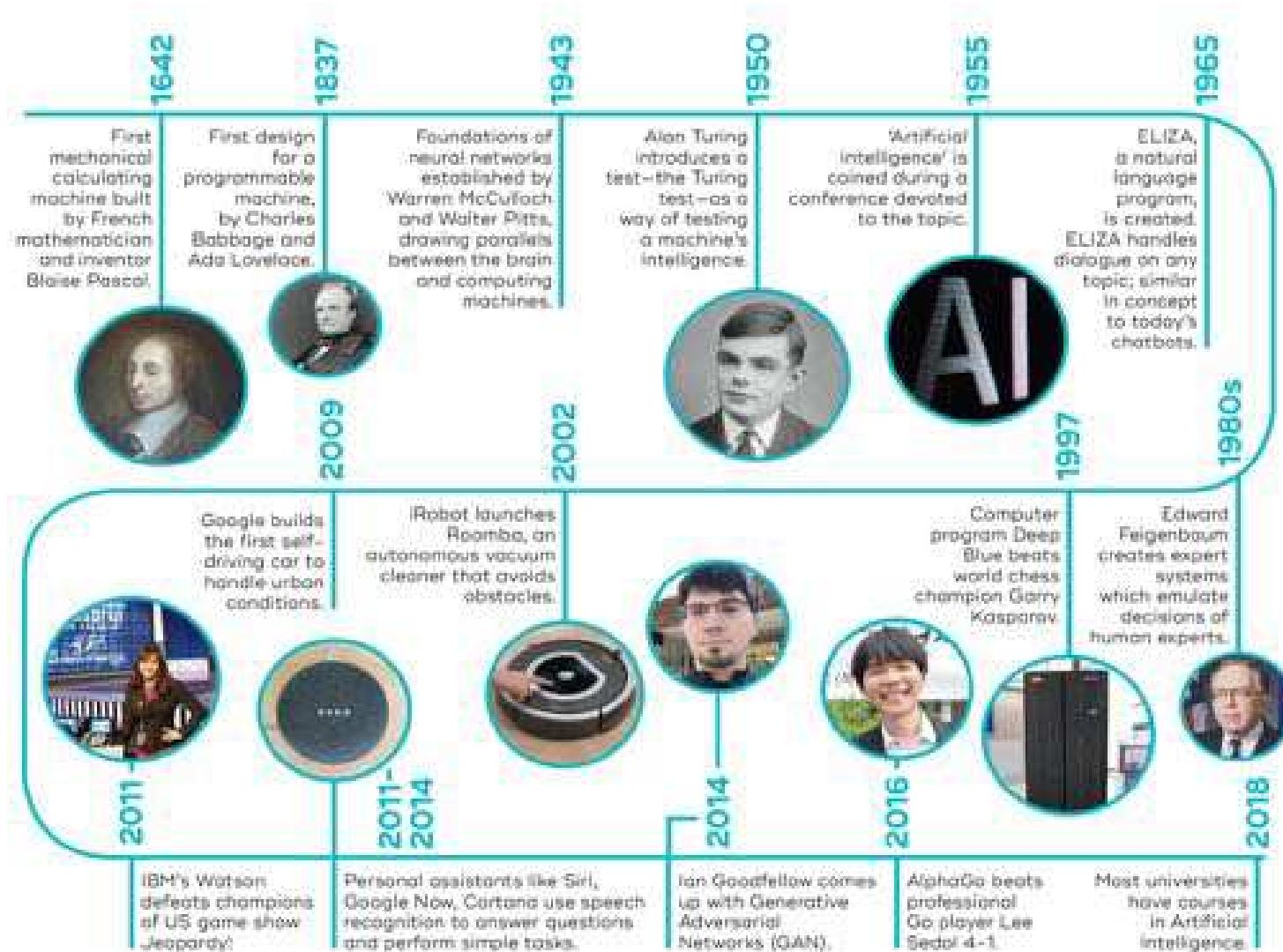
- John Naisbitt, 1982



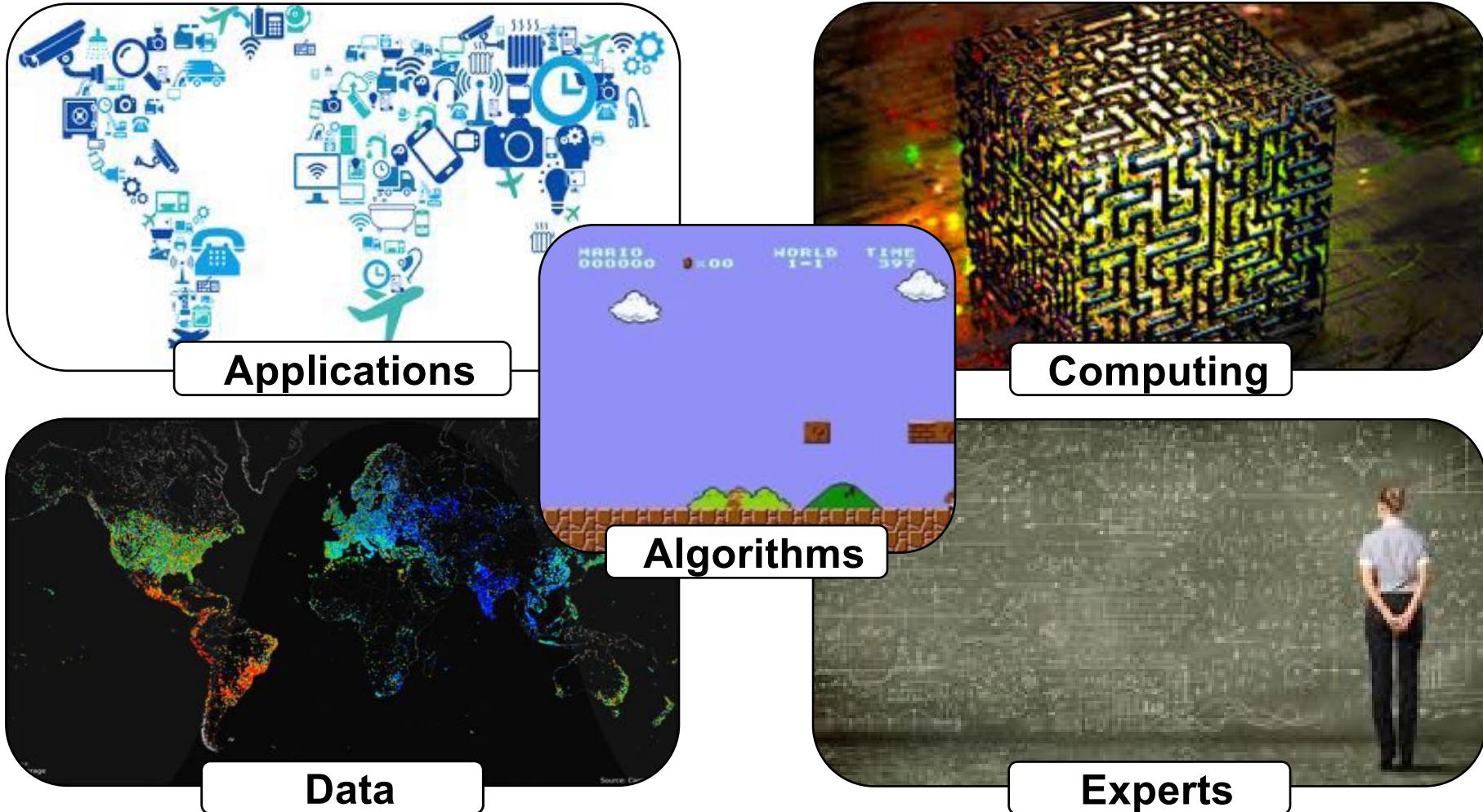
# Origin of a hype about ...



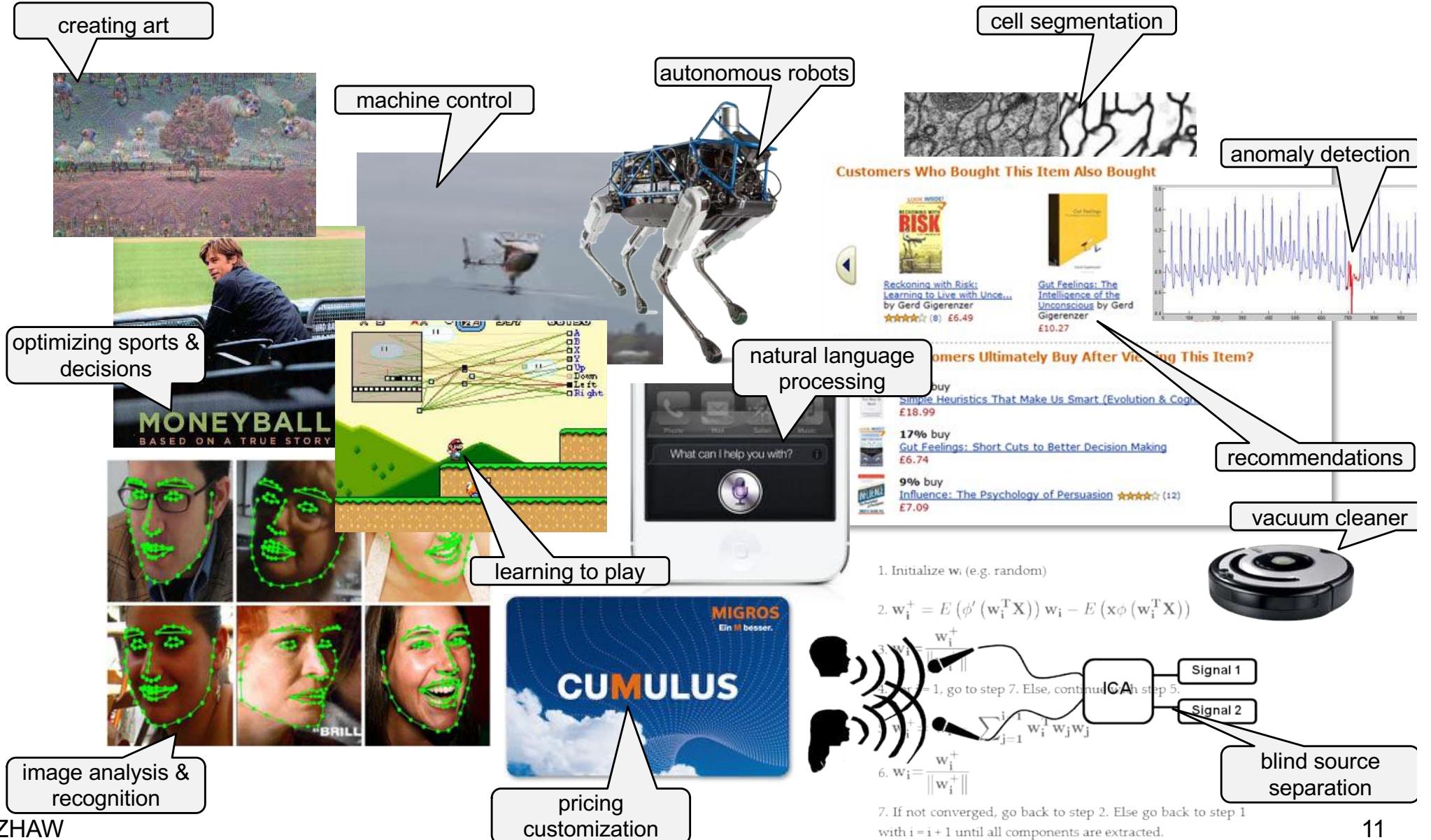
# Origin of a hype about ...



# Origin of a hype about ...



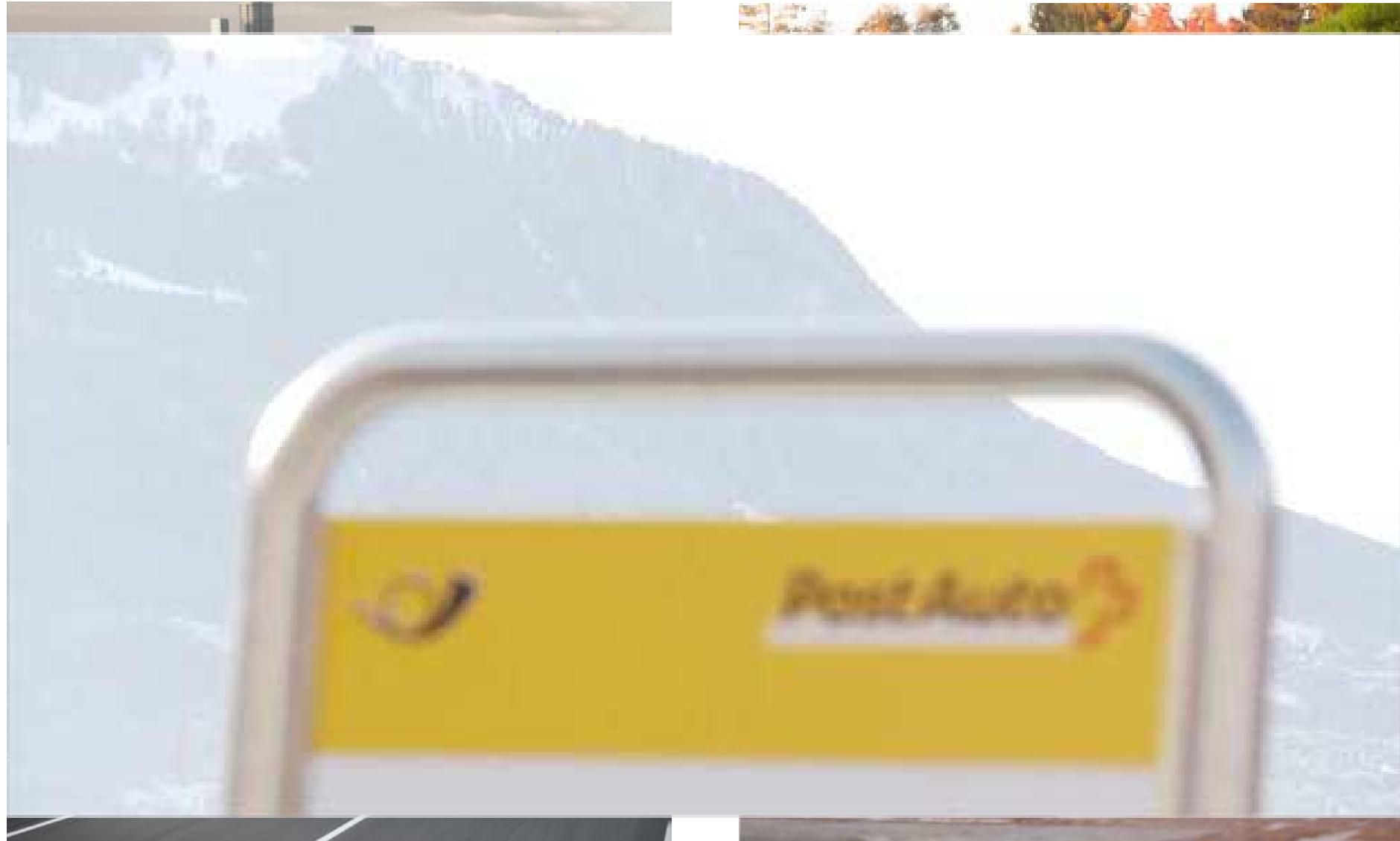
# ... about Applications ...



# ... about Applications ...



# ... about Applications ...



# ... about Applications ...

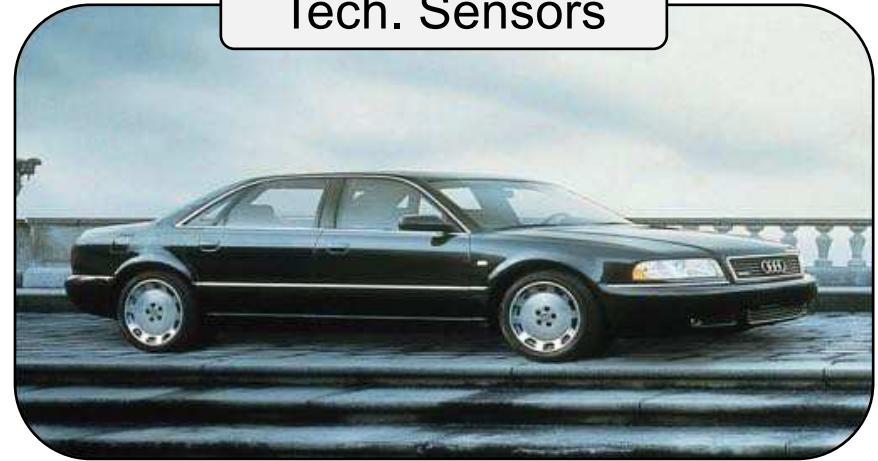


# ... about Data ...

Social Sensors

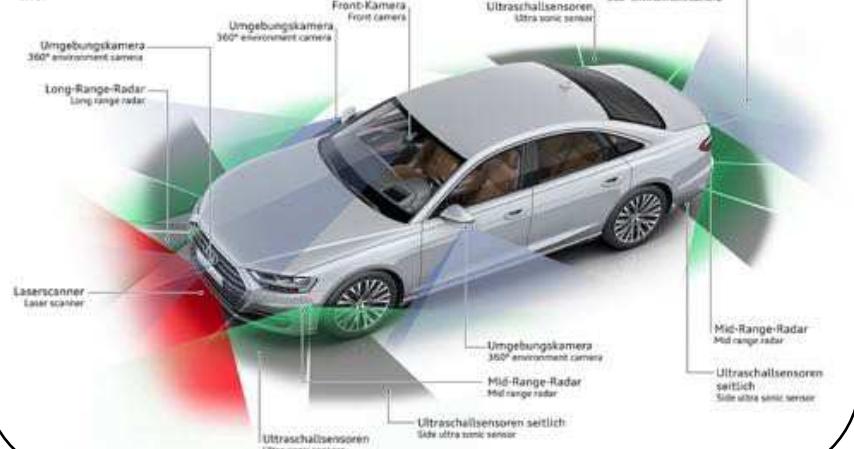


Tech. Sensors



Audi A8

Sensorfelder der Umfeldüberwachung  
Sensor areas for environment observation  
07/17



# ... about Data ...



# ... about Data ...



Tweets  
**30.5K**

Following  
**298K**

Followers  
**106M**



# ... about Data ...



# ... about Computing ...

ASCI Red 1997



Size: tennis court, Costs: \$55  
Mio., Power: 1.3 tF

Sony PS3 2006



Costs: \$ 499 Power: 2.1 tF

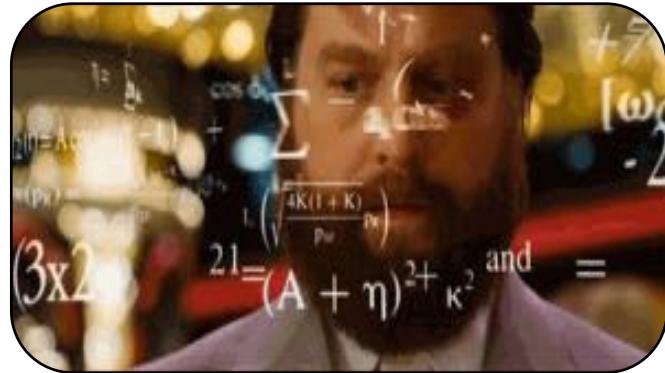
NVIDIA DGX2 2018



Costs: \$ 399.000, Power: 2 pF



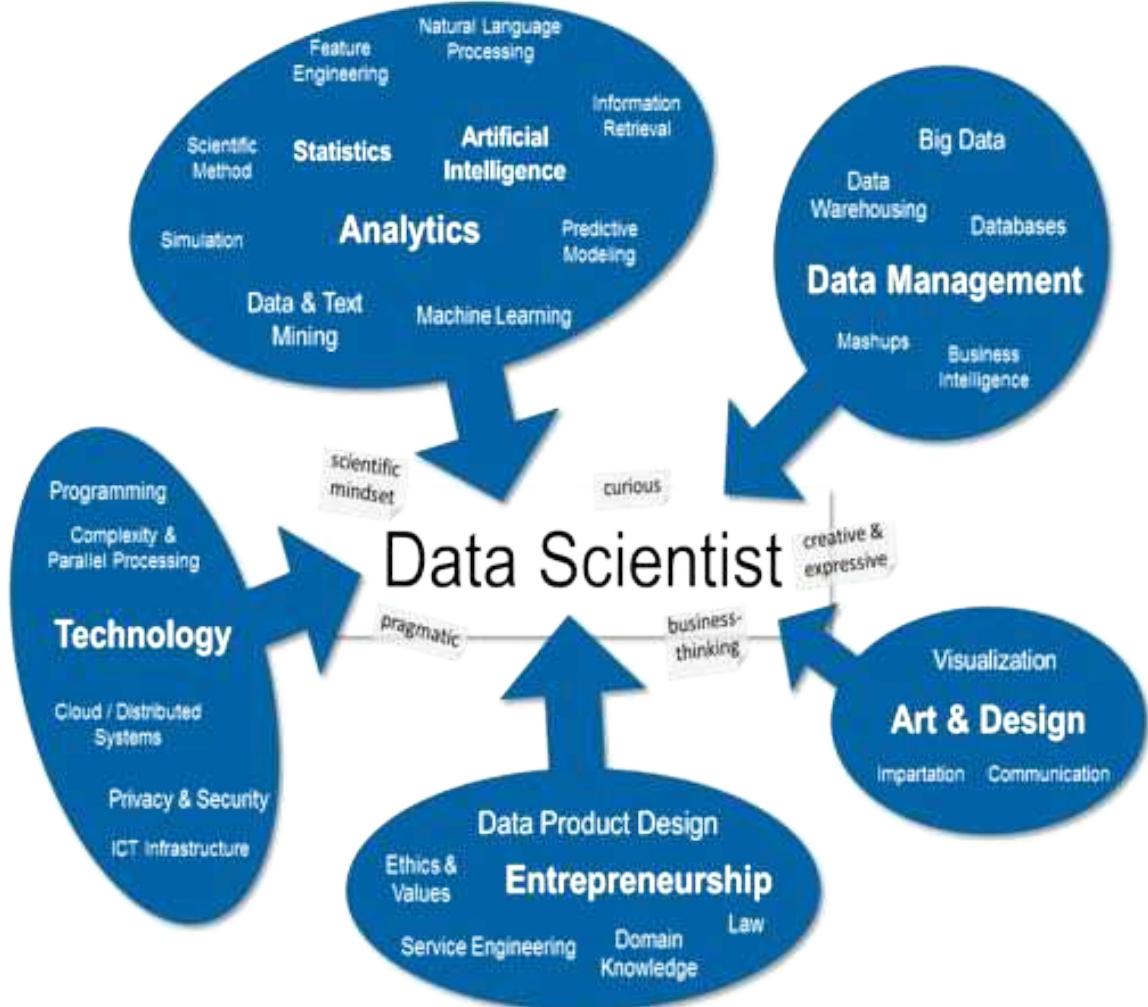
# ... about Experts ...



Data Scientist Is the Best Job In America According Glassdoor's 2018 Rankings

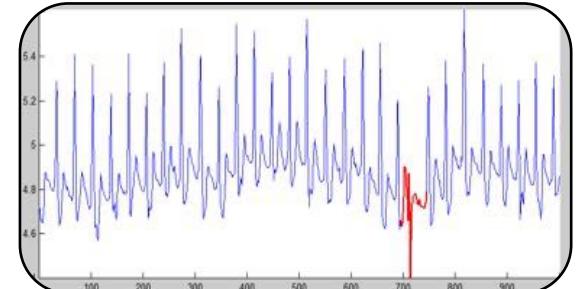
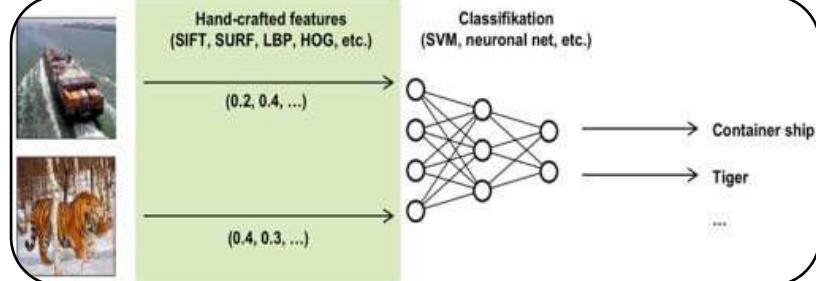
## Data Scientist: The Sexiest Job of the 21st Century

Want the people who have made the most out of money, unstructured data, by Thomas M. Davenport and G.E. Poff  
**W**hen I was a kid, I used to read the old encyclopedias. They were great, but they were also very heavy and took up a lot of space. So I decided to make my own. I used to collect old books and magazines, and then I would cut them up and paste them into my own book. It was a lot of work, but it was also a lot of fun. I learned a lot about how things work, and I also learned a lot about how people think. I think that's why I became a Data Scientist.

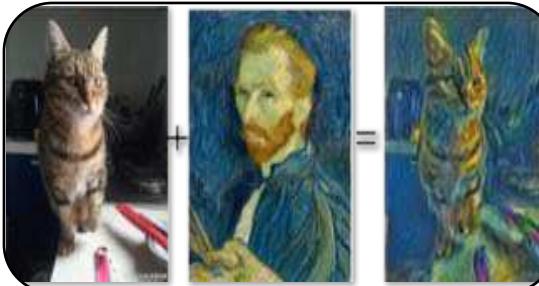
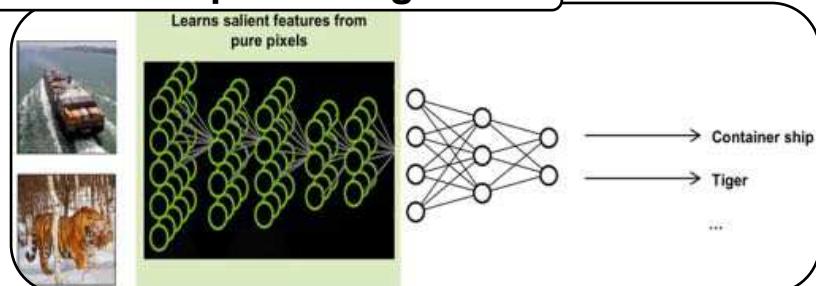


# ... and about Algorithms.

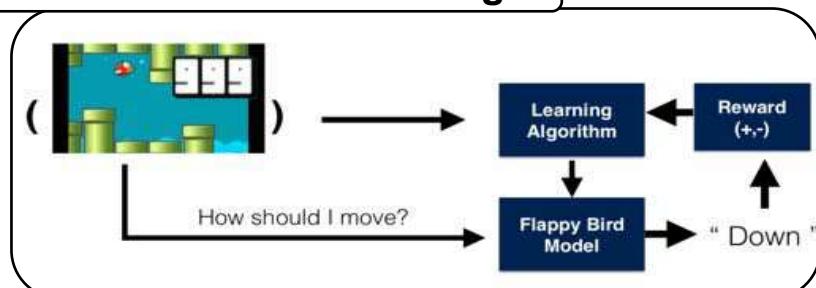
## Machine Learning



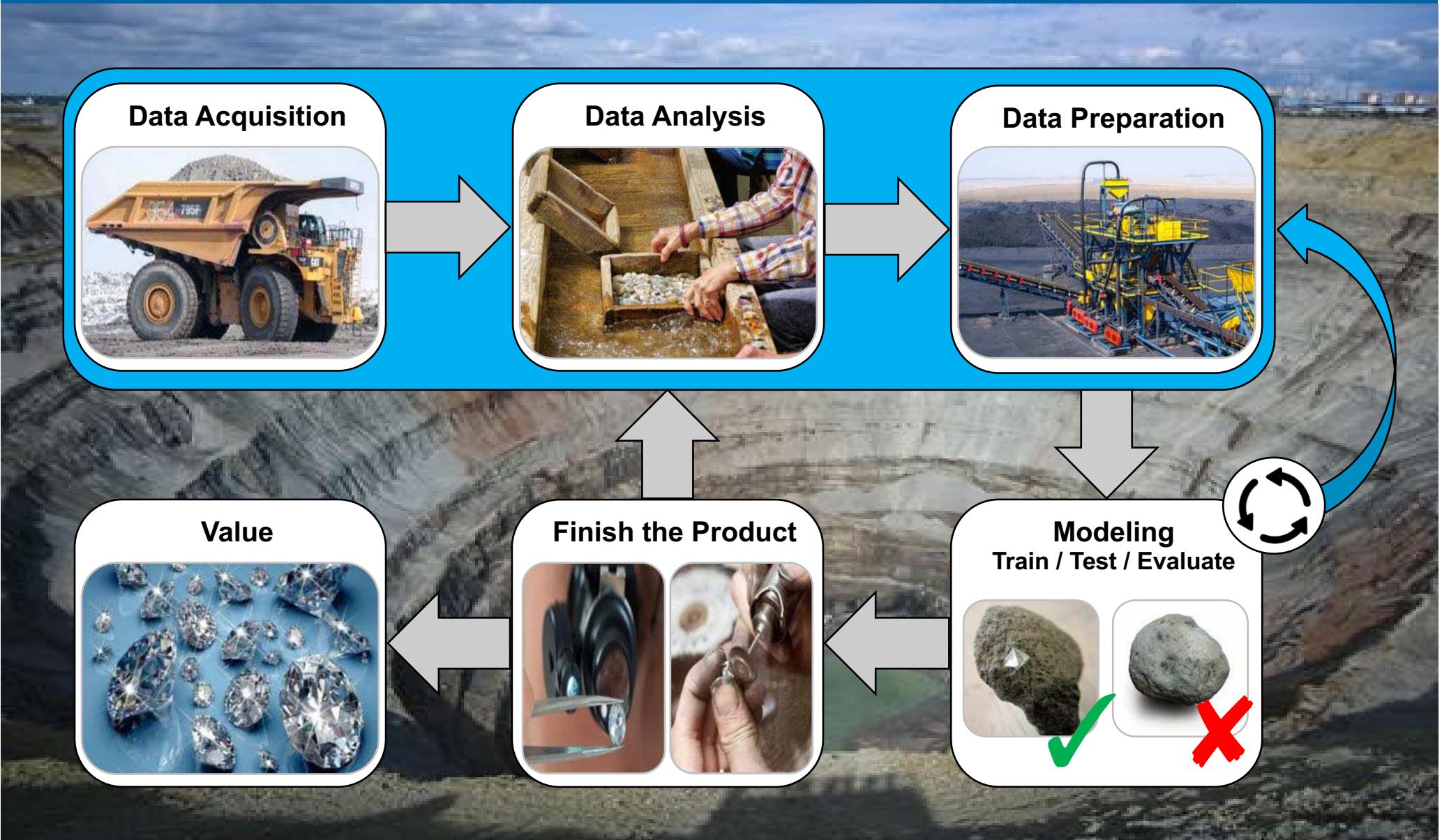
## Deep Learning



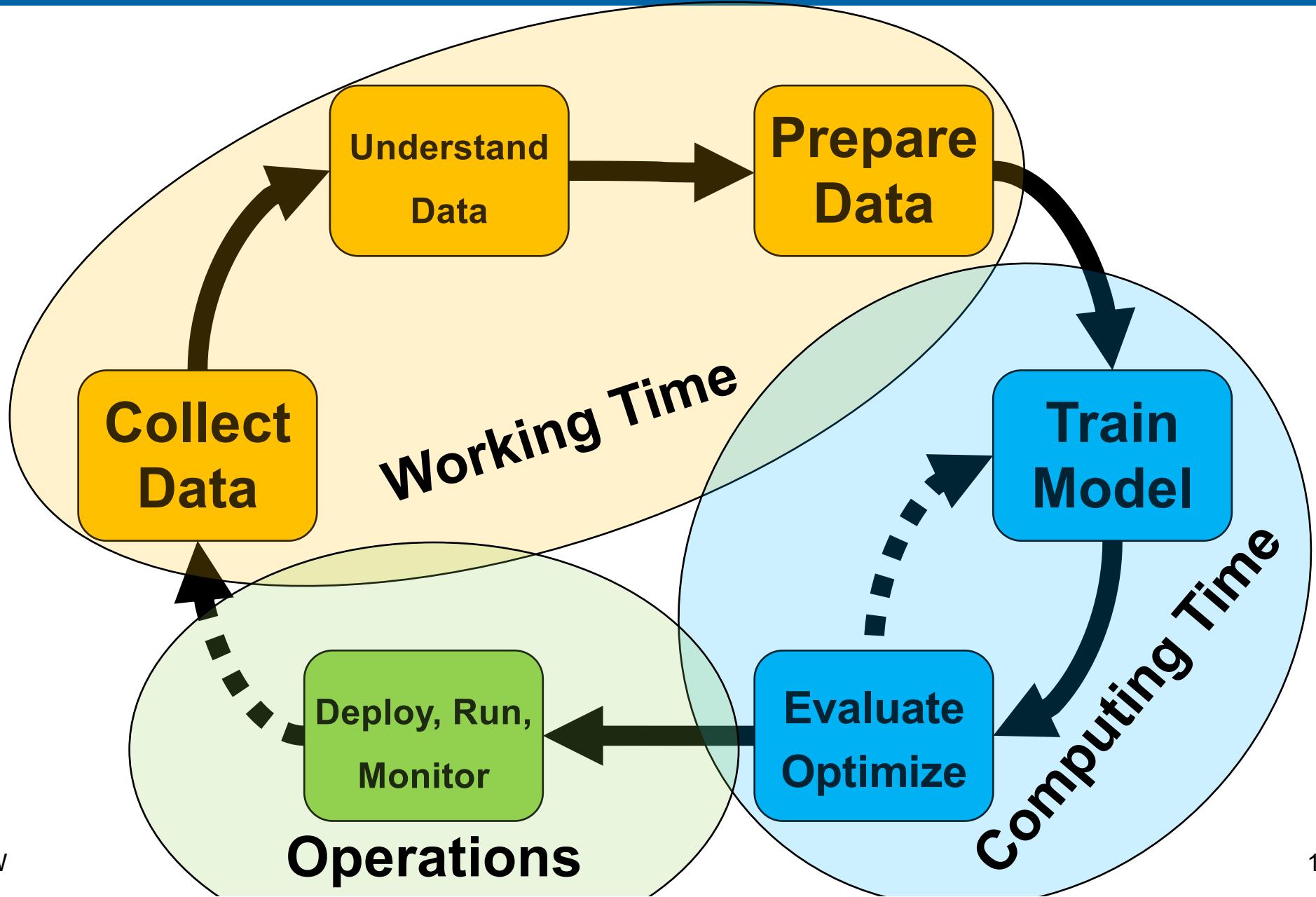
## Reinforcement Learning



# Typical Data Science Project

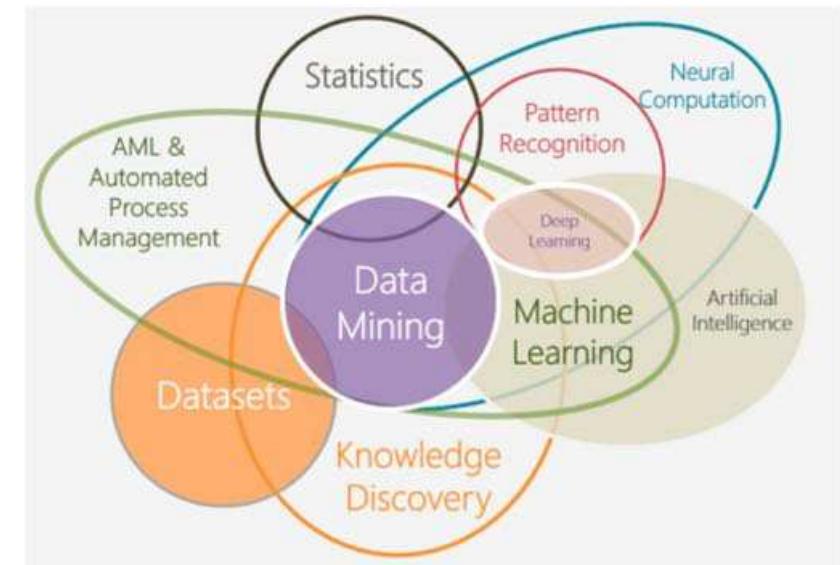


# Typical Data Science Project

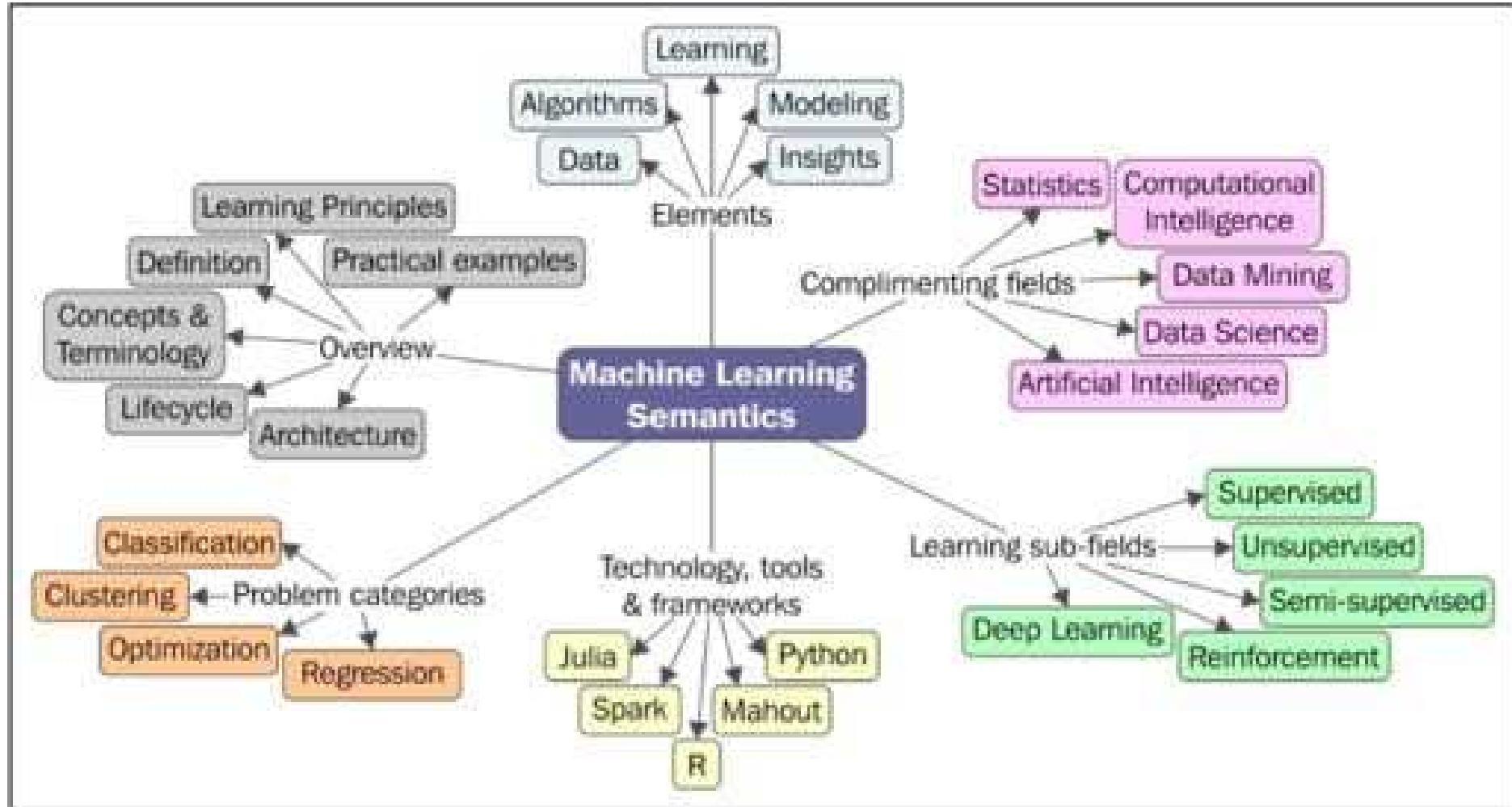


# Wikipedia Definitions

- „**Data mining** is the process of discovering patterns in large data sets involving methods at the intersection of machine learning, statistics, and database systems. The term "data mining" is a misnomer, because the goal is the extraction of patterns and knowledge from large amounts of data, not the extraction (mining) of data itself“
- „**Machine learning** is the scientific study of algorithms and statistical models that computer systems use to perform a specific task without using explicit instructions, relying on patterns and inference instead..“



# Machine Learning and Data Mining



# Data Mining

- is the analysis step of the Knowledge Discovery and Data Mining process (short KDD)
- extracting hidden information
- an interdisciplinary subfield of computer science
- the computational process of discovering patterns in large data sets
- involving methods at the intersection of artificial intelligence, machine learning, statistics, and database systems
- complexity considerations, post-processing of discovered structure and knowledge, information visualization

# Questions and Answers



# Data Mining - Example

Transaction	Items
1	{Bread, Milk}
2	{Bread, Diapers, Beer, Eggs}
3	{Milk, Diapers, Beer, Cola}
4	{Bread, Milk, Diapers, Beer}
5	{Bread, Milk, Diapers, Cola}
6	{Bread}

- Example of a frequent item set: {Diapers, Beer}
- Example of an association rule: {Diapers} → {Beer}
- Support {Beer} = transaction with {Beer} / transactions
- Confidence {Diapers → Beer} = 
$$\frac{\text{Support } \{\text{Diapers, Beer}\}}{\text{Support } \{\text{Diapers}\}}$$

# Machine Learning

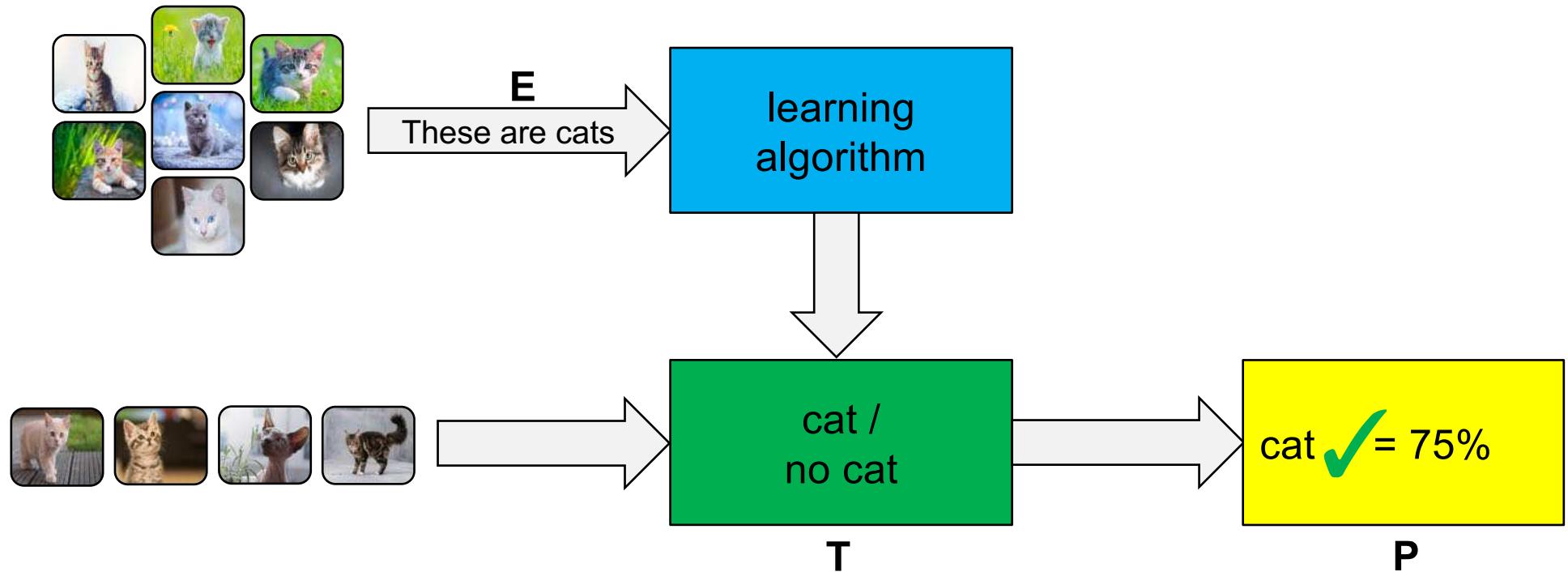
- Arthur Samuel (1959):
  - „*Machine Learning is the field of study that gives computers the ability to **learn** without being explicitly programmed.*“
- Herbert Simon (1983):
  - „*Learning denotes changes in the system that are **adaptive** in the sense that they enable the system to do the same task or tasks drawn from the same population more efficiently and more effectively the next time.*“
- Tom Mitchell (1997):
  - „*Machine Learning is the study of computer algorithms that improve **automatically** through experience.*“
  - „*A computer program is said to learn from experience **E** with respect to some class of tasks **T** and performance measure **P**, if its performance at tasks in **T**, as measured by **P**, improves with experience **E**“*



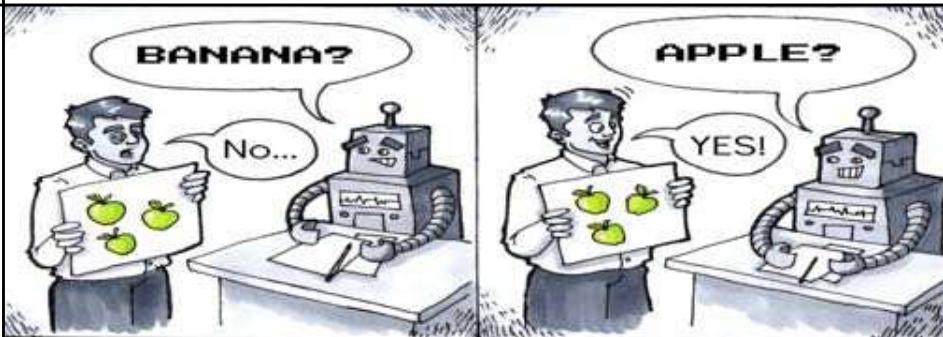
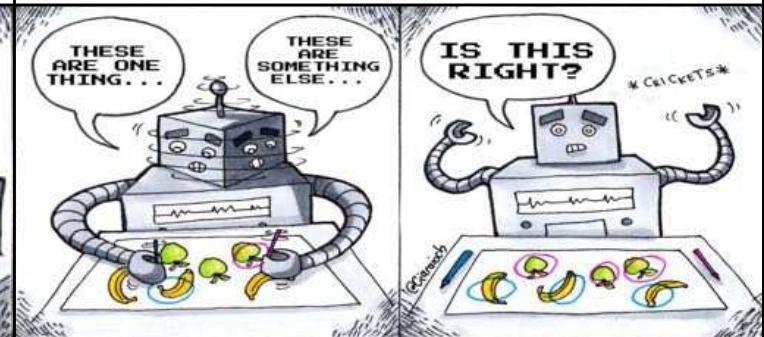
# Machine Learning

- Tom Mitchell\* (1997):

- „A computer program is said to learn from experience  $E$  with respect to some class of tasks  $T$  and performance measure  $P$ , if its performance at tasks in  $T$ , as measured by  $P$ , improves with experience  $E$ .“



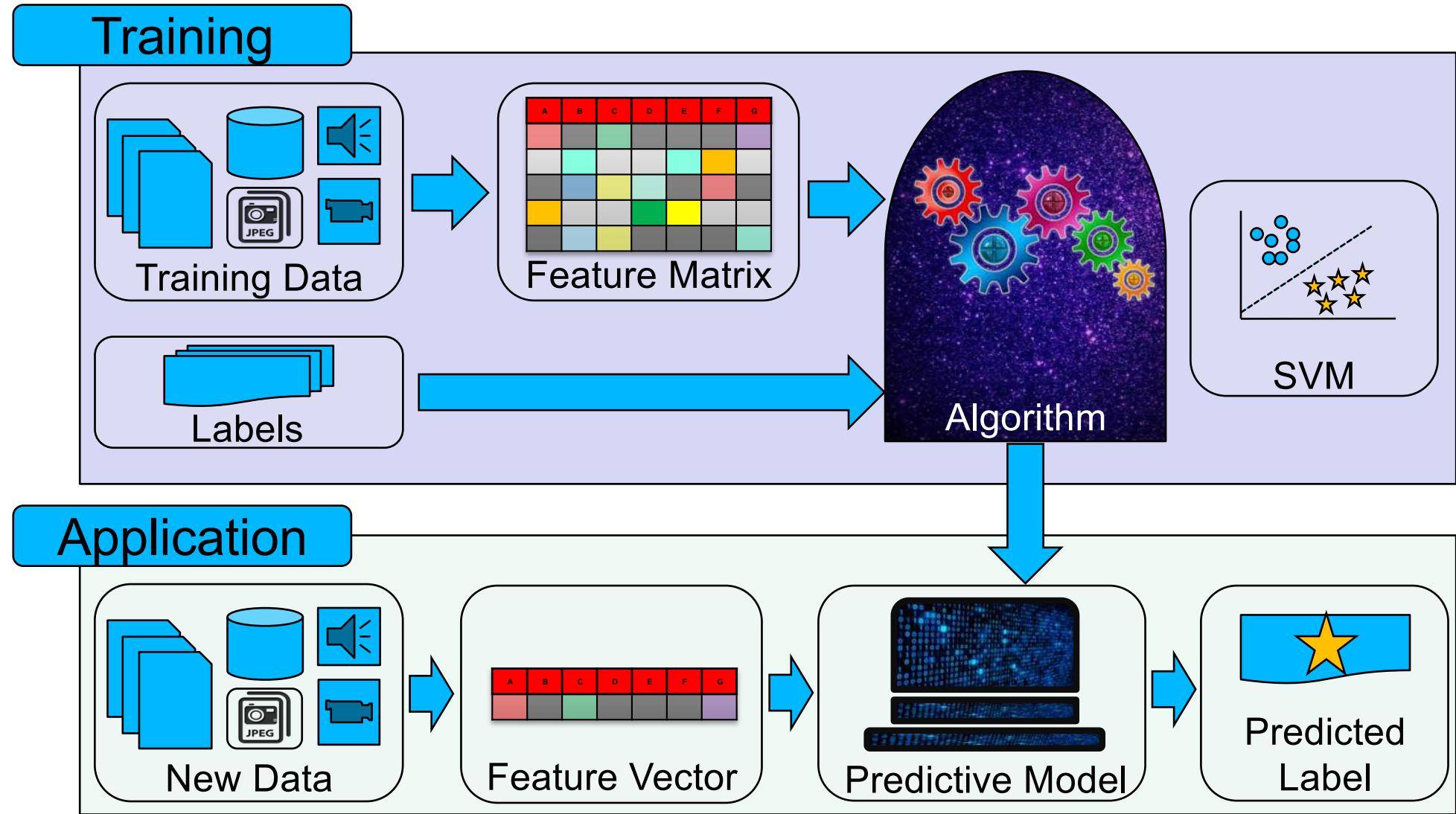
# Machine Learning - Types

	Supervised	Unsupervised
Description	the algorithm is presented with inputs (independent variables) and associated labels indicating the class of the observation (dependent variable). The algorithm attempts to learn the rule that maps inputs to each class. New data is classified based on the rule learned by the algorithm.	the algorithm is presented only with inputs (independent variables). The algorithm attempts to classify things based on similarity or dissimilarity to each other.
Data	is labeled with a class or value	is unlabeled
Goal	predict class or value	determine data patterns / groupings
Algorithms	SVM, Linear Regression, Decision Trees	K-Means, DBScan, Hierarchical
Task		

# Questions and Answers

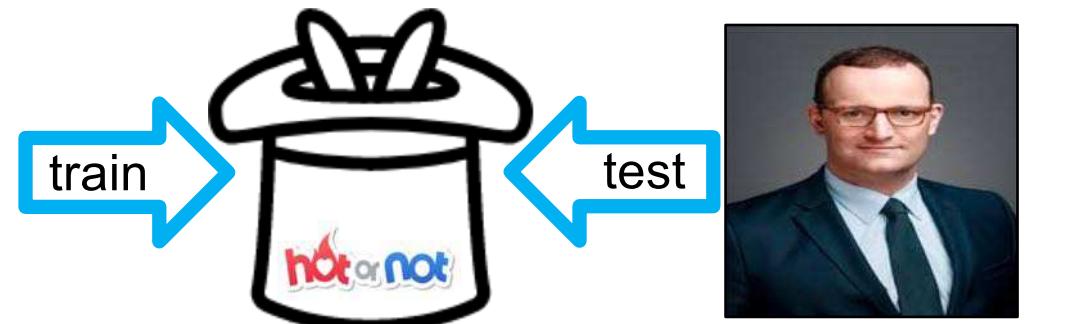
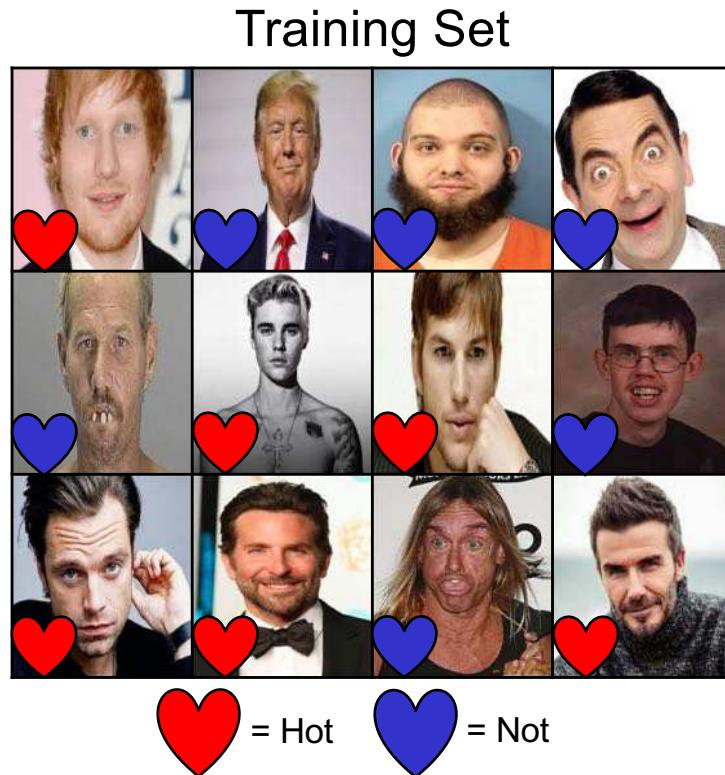


# Machine Learning - Supervised



# Machine Learning - Supervised

- Classification is the problem of identifying to which of a set of categories a new observation belongs, on the basis of a training set of data containing observations whose category is known.



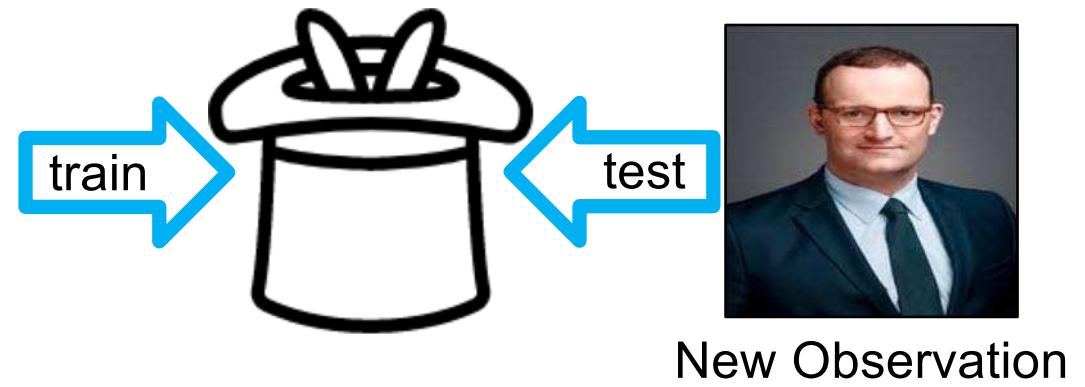
# Machine Learning - Supervised

- Regression is the problem of predicting and forecasting a concrete number based on a set of data containing observations whose category is known.

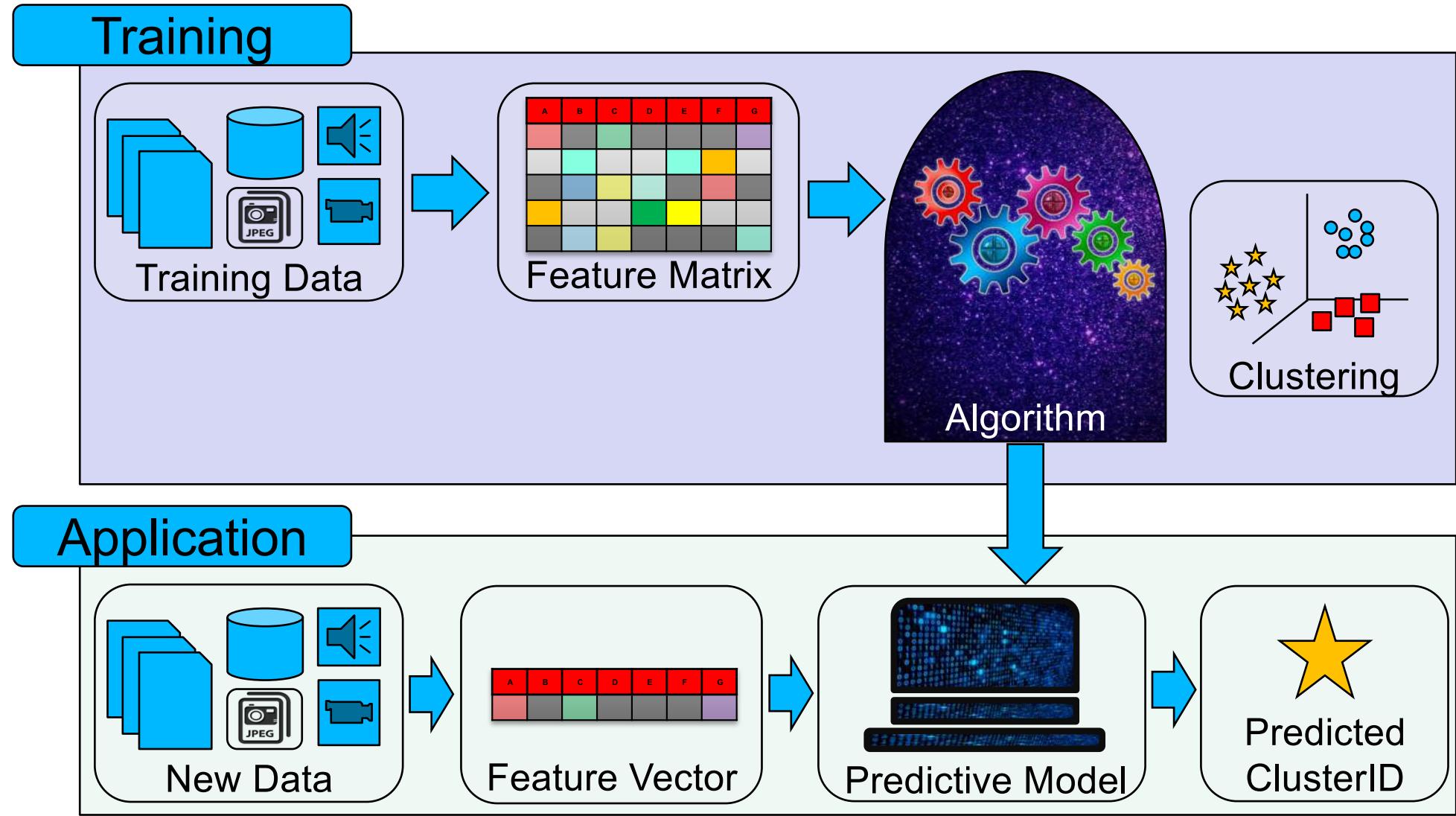
Training Set



37 = Age

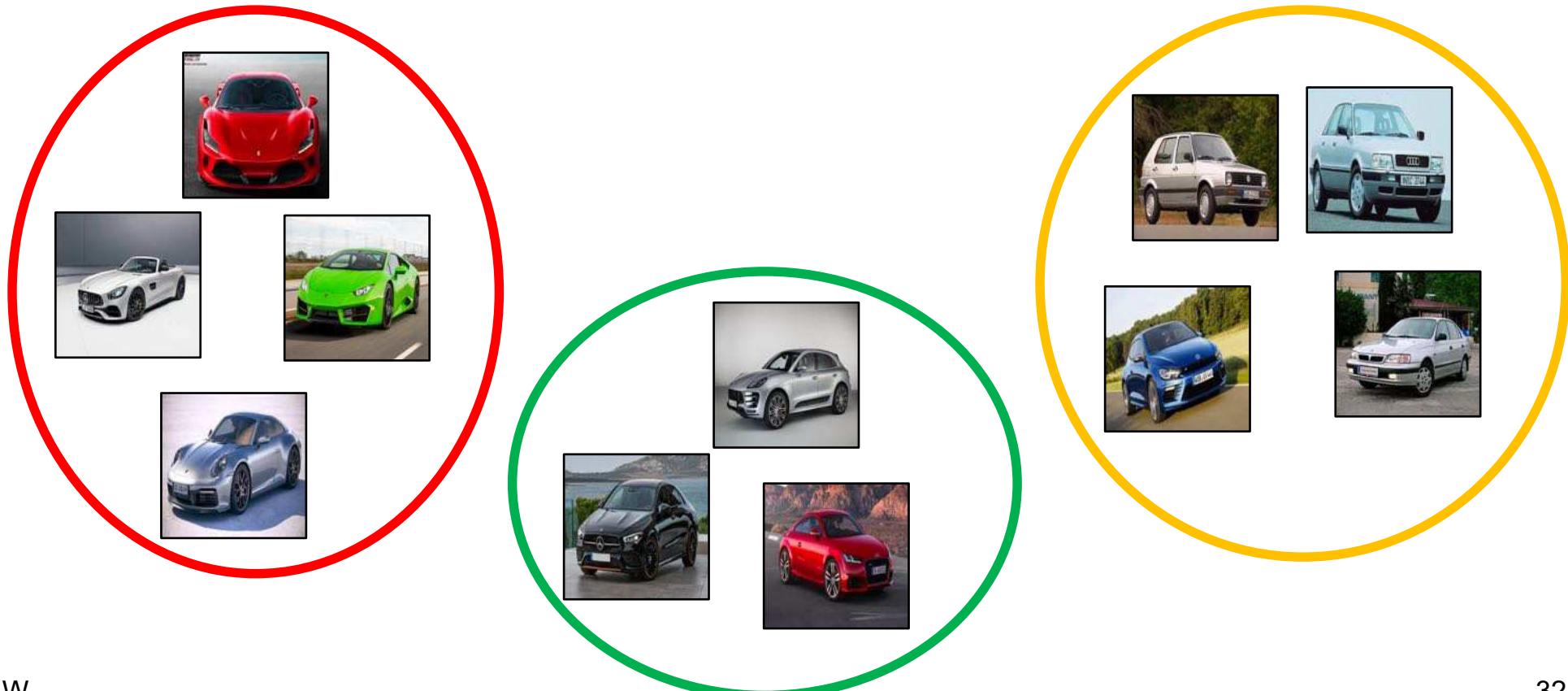


# Machine Learning - Unsupervised



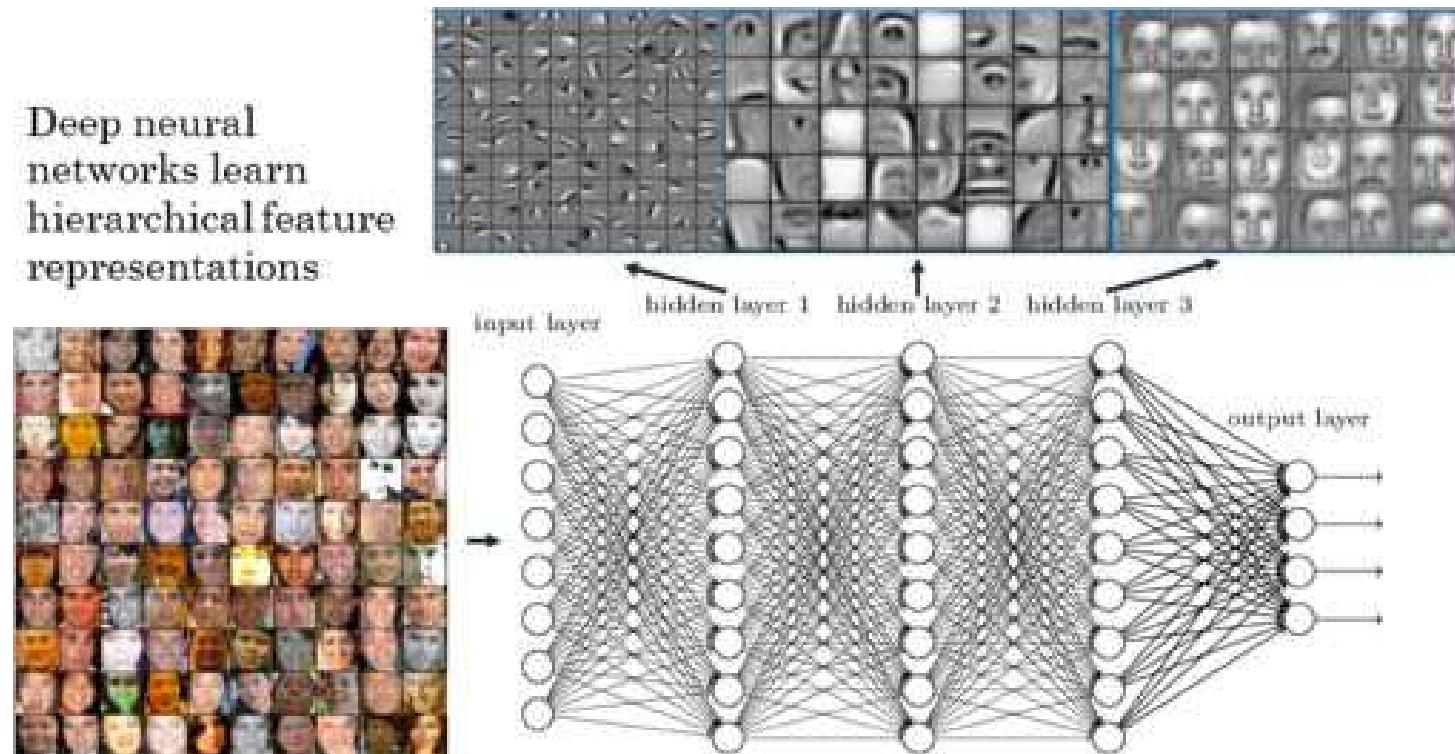
# Machine Learning - Unsupervised

- Clustering is the task of grouping a set of objects in such a way that objects in the same group (called a cluster) are more similar (in some sense) to each other than to those in other groups (clusters)

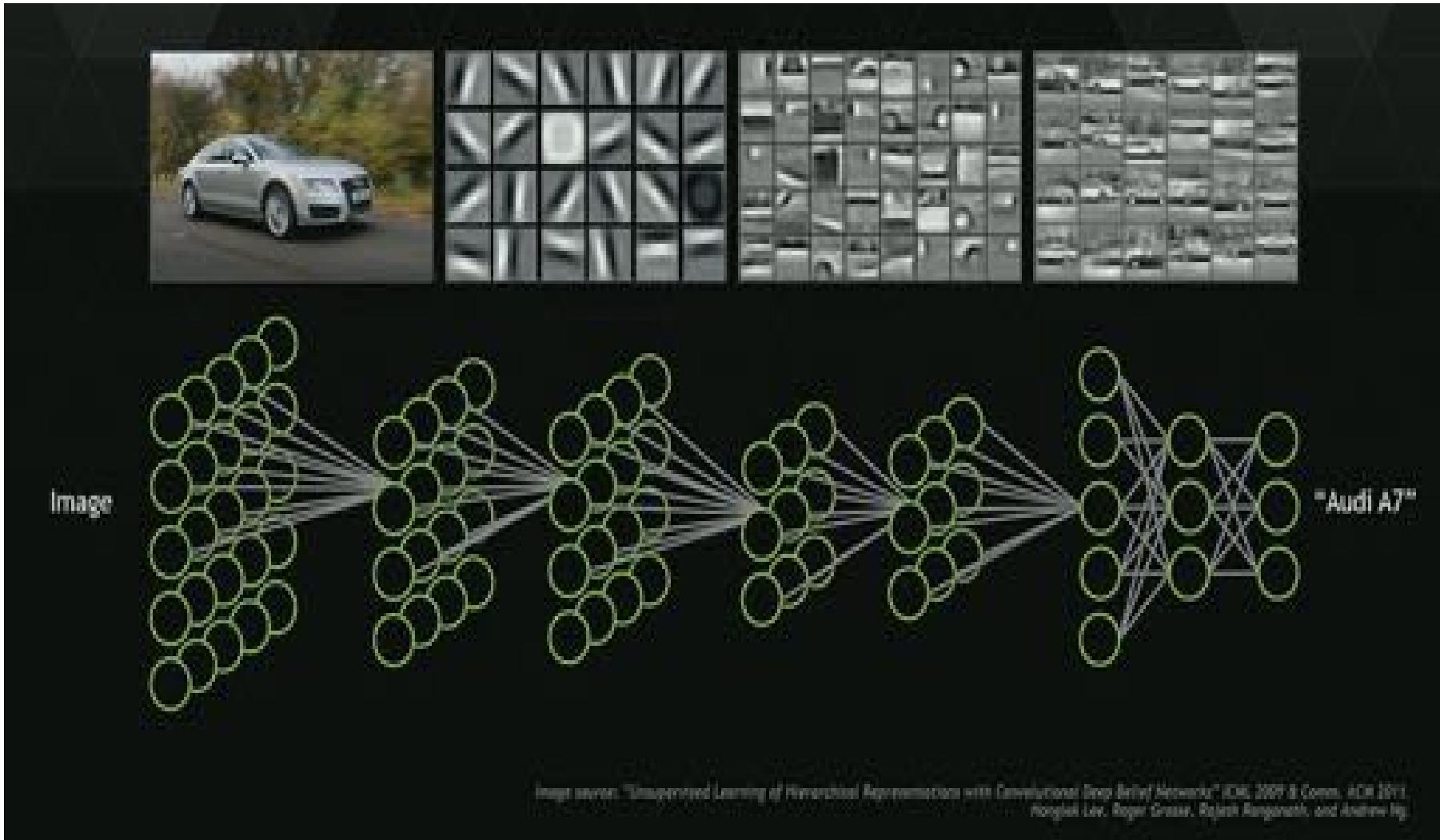


# Excursion: Deep Learning

- Deep Learning uses multiple layers to progressively extract higher level features from the raw input. For example, in image processing, lower layers may identify edges, while higher layers may identify the concepts relevant to a human such as digits or letters or faces.



# Excursion: Deep Learning



# Machine Learning vs. Deep Learning

## Machine Learning



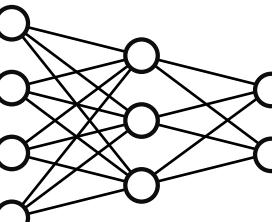
Feature Extraction  
(SIFT, SURF, LBP, HOG, etc.)

(0.2, 0.4, ...)

Classification  
(SVM, Neuronal Network, etc.)



(0.4, 0.3, ...)



→ Container Ship

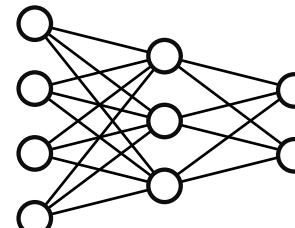
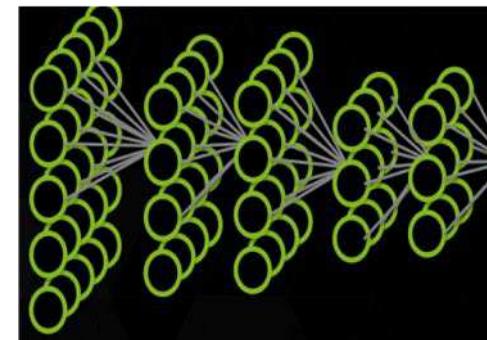
→ Tiger

...

## Deep Learning



takes raw pixels, features are  
learned by the network!



→ Container Ship

→ Tiger

...

# Machine Learning vs. Deep Learning

	Machine Learning	Deep Learning
Process	cannot perform automatic feature extraction, requires labelled parameters	performs automatic feature extraction without the need of human intervention
Structure	can be defined as a shallow neural network which consists one input and one output, with barely one hidden layer	to be qualified as deep learning, there has to be at least three layers
Data requirements	can train on lesser data	requires large data
Accuracy	gives lesser accuracy	provides high accuracy
Training time	takes less time to train	takes longer to train
Hardware	trains on CPU	requires GPU to train properly
Hyperparameter tuning	limited tuning capabilities	can be tuned in various different ways
Transparency	more transparent: able to more easily explain decisions made by the model	more opaque: difficult to explain why the model made a given decision

# Excursion: Reinforcement Learning

- Reinforcement Learning is concerned with how agents ought to take actions in an environment in order to maximize the notion of cumulative reward.

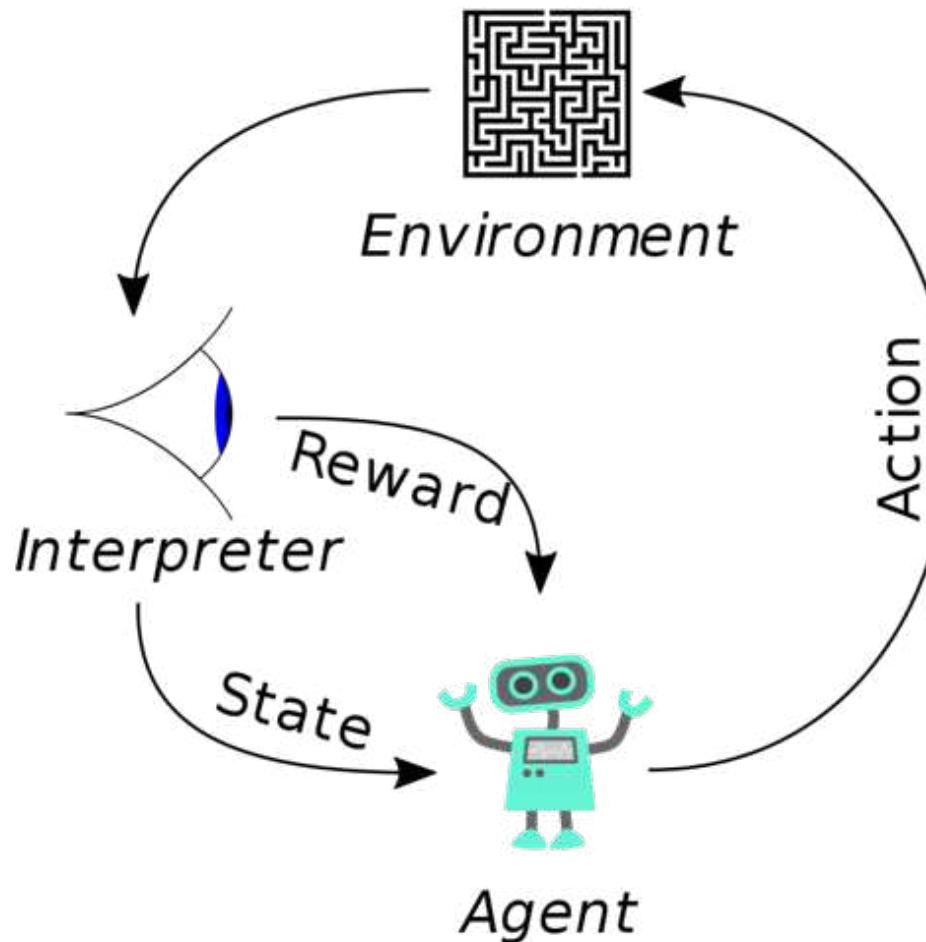


Single Agent



Multi Agents

# Excursion: Reinforcement Learning



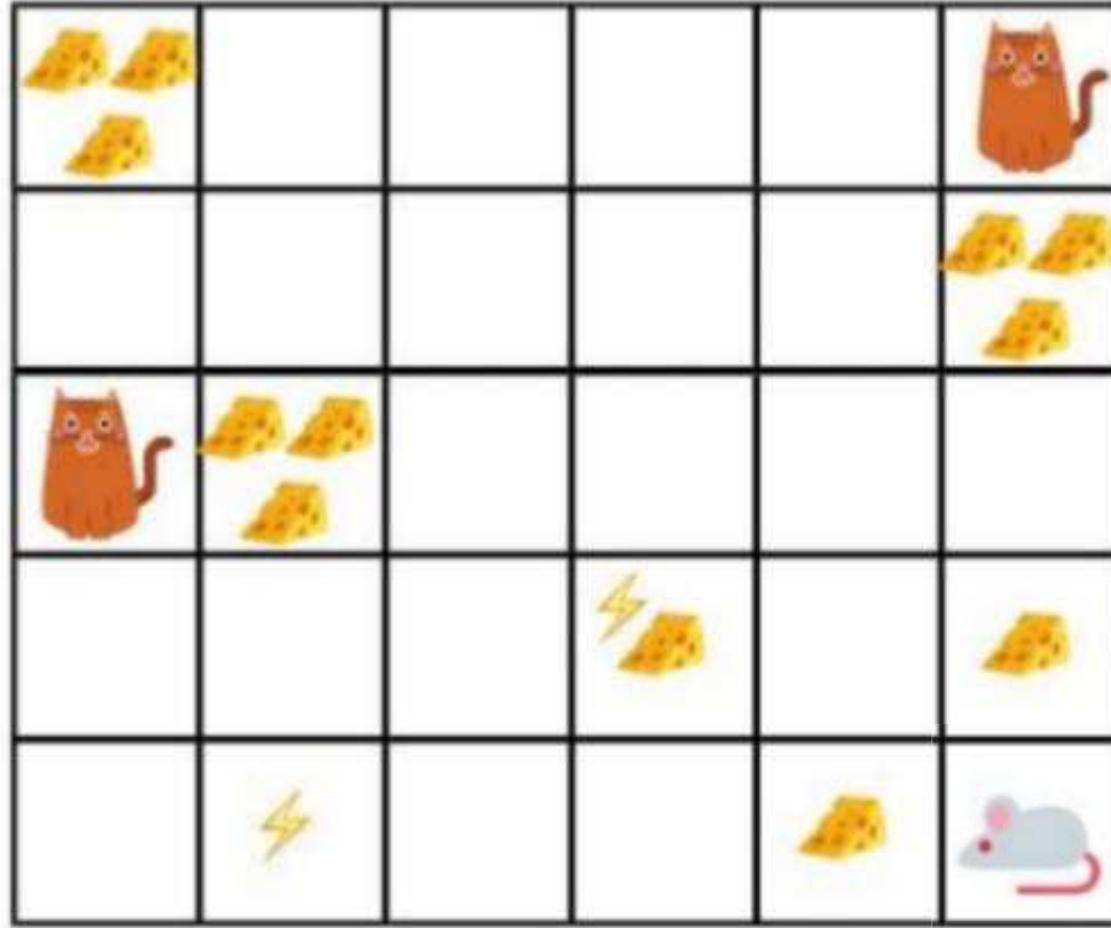
- At each step  $t$  the agent:
  - Executes action  $A_t$
  - Receives observation  $O_t$
  - Receives scalar reward  $R_t$
- The environment:
  - Receives action  $A_t$
  - Emits observation  $O_{t+1}$
  - Emits scalar reward  $R_{t+1}$
- $t$  increments at env. step

# Excursion: Reinforcement Learning

- Fly stunt manoeuvres in a helicopter
- Defeat the world champion at Backgammon
- Manage an investment portfolio
- Control a power station
- Make a humanoid robot walk
- Play Atari games better than humans
- Autonomous cars



# Excursion: Reinforcement Learning



<https://medium.com/@julsimon/talk-an-introduction-to-reinforcement-learning-e26177338787>

**Agent** is the mouse

**Actions**: go up, down, left, right

**Reward**:

- zero reward for blank cell
- positive for finding cheese
- negative for electric shock
- very negative for meeting cat

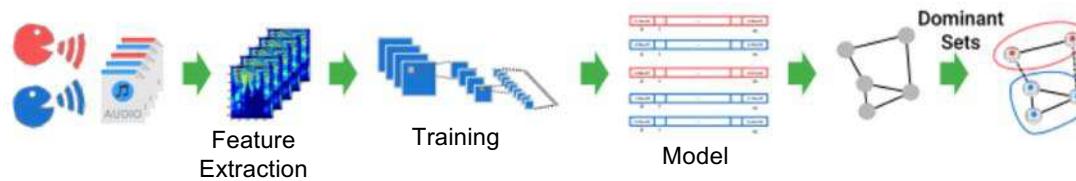
**Episode**: until mouse dies or max 50 steps

# Projects: Speaker and Face Recognition

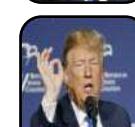
## Challenge: Detection of speakers



**Method:** Neural Network trained with speeches



## Result

	vs		= 0.00
	vs		= 0.23
	vs		= 0.78
	vs		= 0.81

# Projects: Speaker and Face Recognition

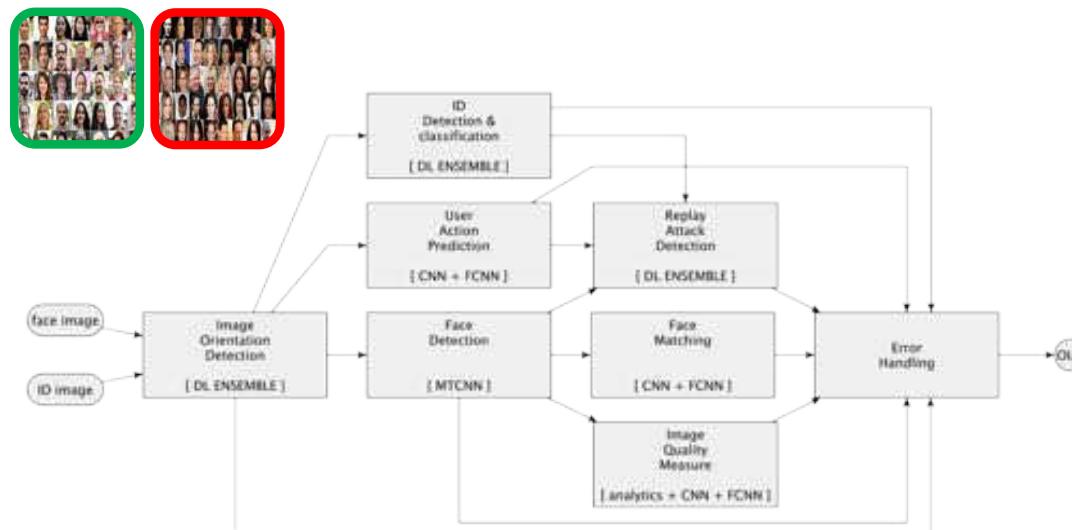
## Challenge: Detection of “fake” faces



## Result



**Method:** Neural Network trained with images of faces



«Deep Learning in the Wild». ANNPR’2018. Stadelmann, Amirian, Arabaci, Arnold, Duivesteijn, Elezi, Geiger, Lörwald, Meier, Rombach & Tuggener (2018).

# Projects: Advertisement Detection



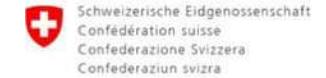
# Projects: Skillue – digital marketplace

## Talents

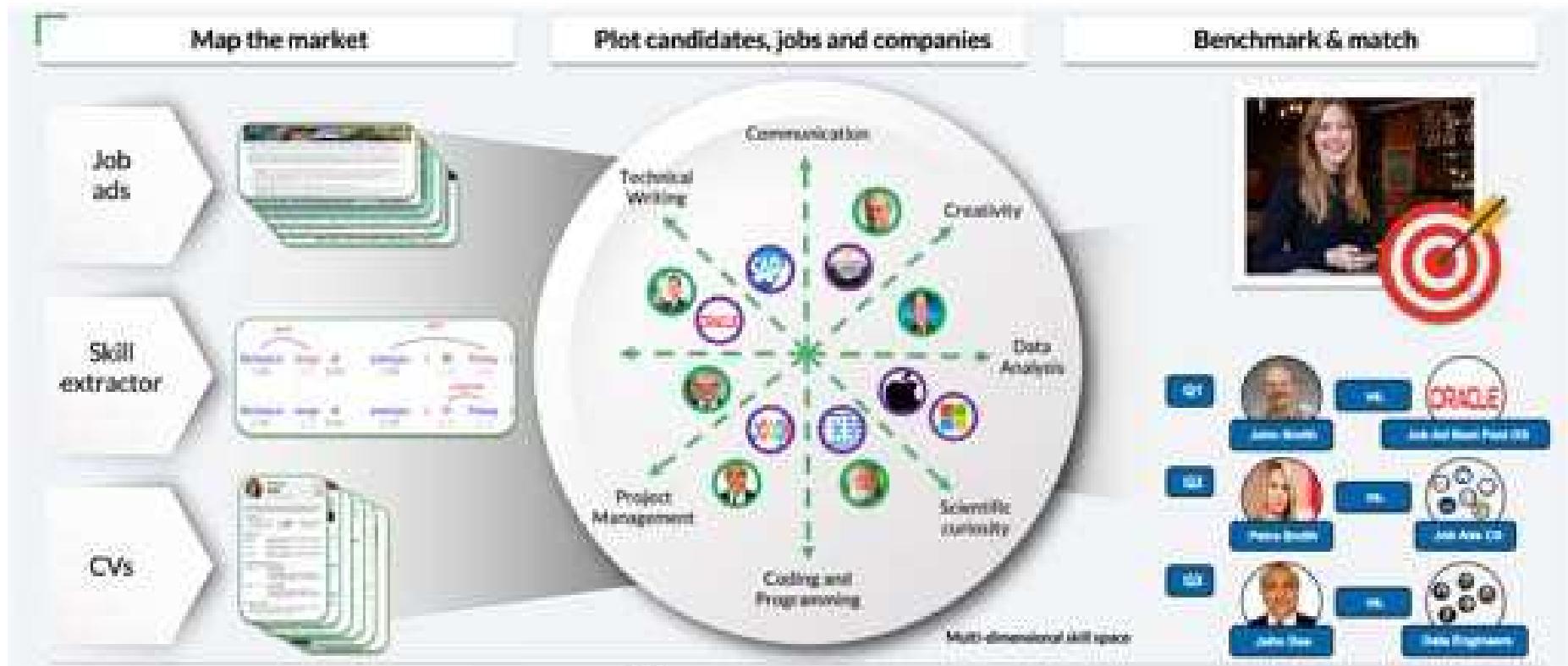
- ✓ compare yourself to the market
- ✓ create your career path
- ✓ find the perfect job

## Companies

- ✓ find top talents
- ✓ compare vacancies to talents
- ✓ detect internal skill gaps



Innosuisse – Schweizerische Agentur  
für Innovationsförderung



# Lessons Learned

- fundamental understanding of the topics data mining, machine learning, deep learning and reinforcement learning
- process of a typical data science project
- understanding of the different types of machine learning
- examples of data science projects from the InIT



# Questions and Answers



# Machine Learning and Data Mining

## V02: Data Processing Part 01

Lecturer: Dr. Andreas Weiler



# Repetition

... REPETITION ...

→ IS THE MOTHER →

of learning

# Course Content IT20a\_WIN

Week	Lecture	Lab
KW08	Introduction	Lab 01
KW09	Data Processing Part 01	Lab 02
KW10	Carnival Monday	
KW11	Data Processing Part 02	Lab 03
KW12	Evaluation	Lab 04
KW13	Recommender Systems	<b>Graded Lab 01</b> <span style="border: 1px solid red; padding: 2px;">IMPORTANT</span>
KW14	Association Rules	<b>Graded Lab 01</b> <span style="border: 1px solid red; padding: 2px;">IMPORTANT</span>
KW15	Linear and Logistic Regression	Lab 05
KW16	Easter Monday	
KW17	Support Vector Machines	<b>Graded Lab 02</b> <span style="border: 1px solid red; padding: 2px;">IMPORTANT</span>
KW18	Decision Trees and Naïve Bayes	<b>Graded Lab 02</b> <span style="border: 1px solid red; padding: 2px;">IMPORTANT</span>
KW19	Partition Clustering	Lab 06
KW20	Hierarchical and Density Clustering	Lab 07
KW21	Neural Networks and Closing	Review and Questions for Exam

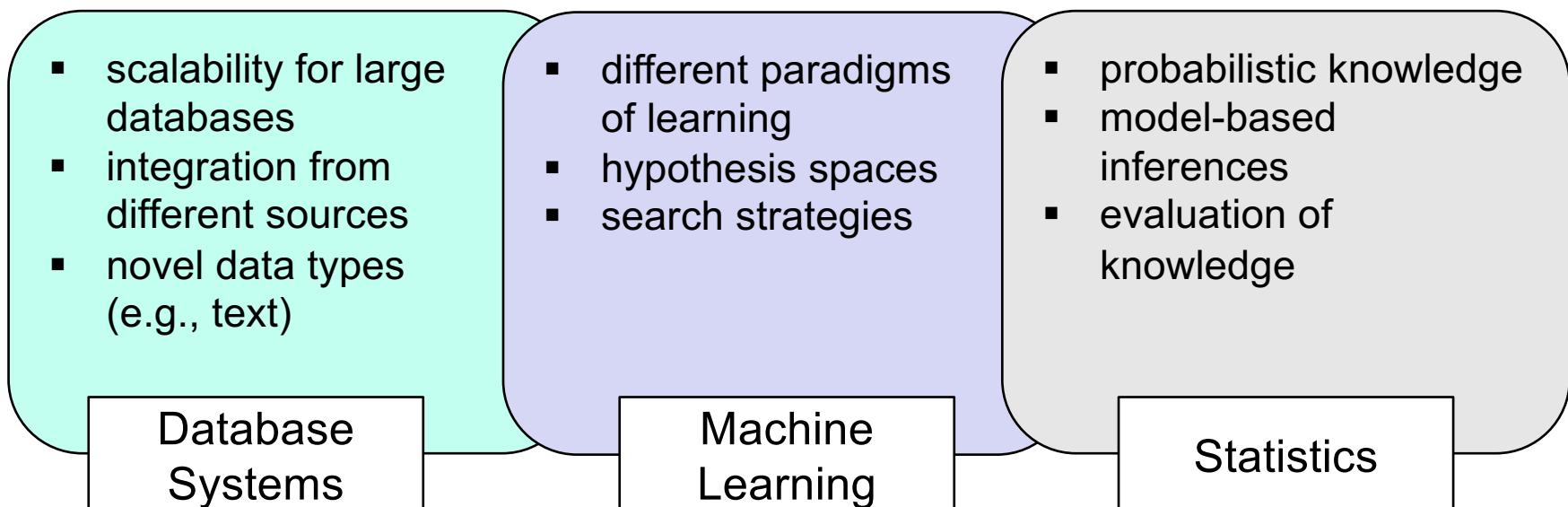
# Learning Objectives

- understanding of the KDD process
- distinguish between different data types and data classes
- understanding of data preprocessing, cleaning, and exploration



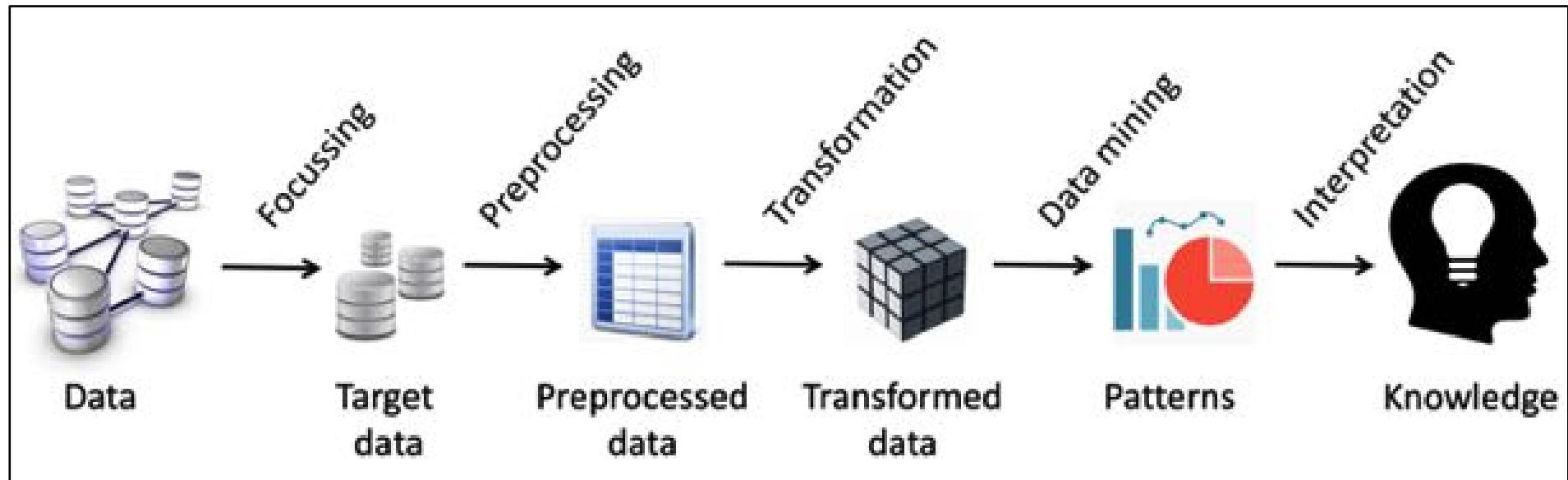
# Knowledge Discovery in Databases

- KDD is the process of (semi-) automatic extraction of knowledge from databases which is
  - valid
  - previously unknown
  - and potentially useful
- Interdisciplinary field



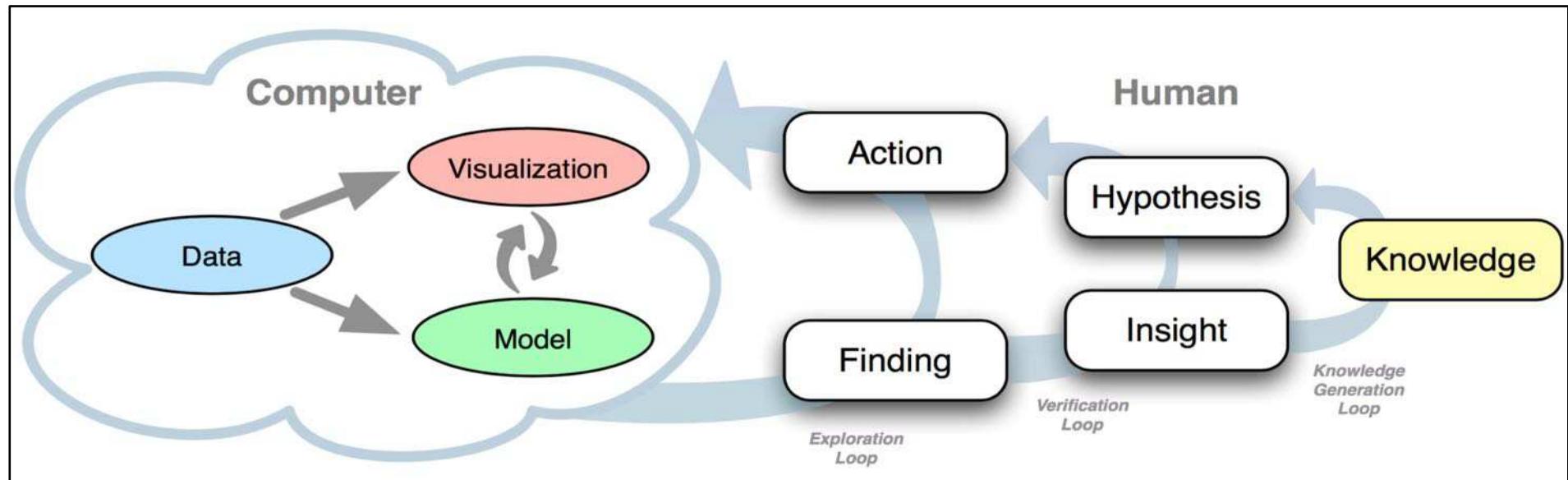
# Knowledge Discovery in Databases

- interactive and iterative process
- continuous optimization of the different tasks



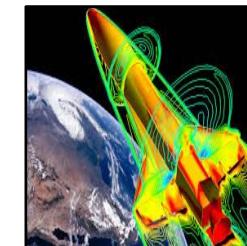
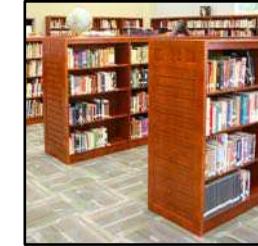
# Knowledge Discovery in Databases

- interactive and iterative process
- continuous optimization of the different tasks

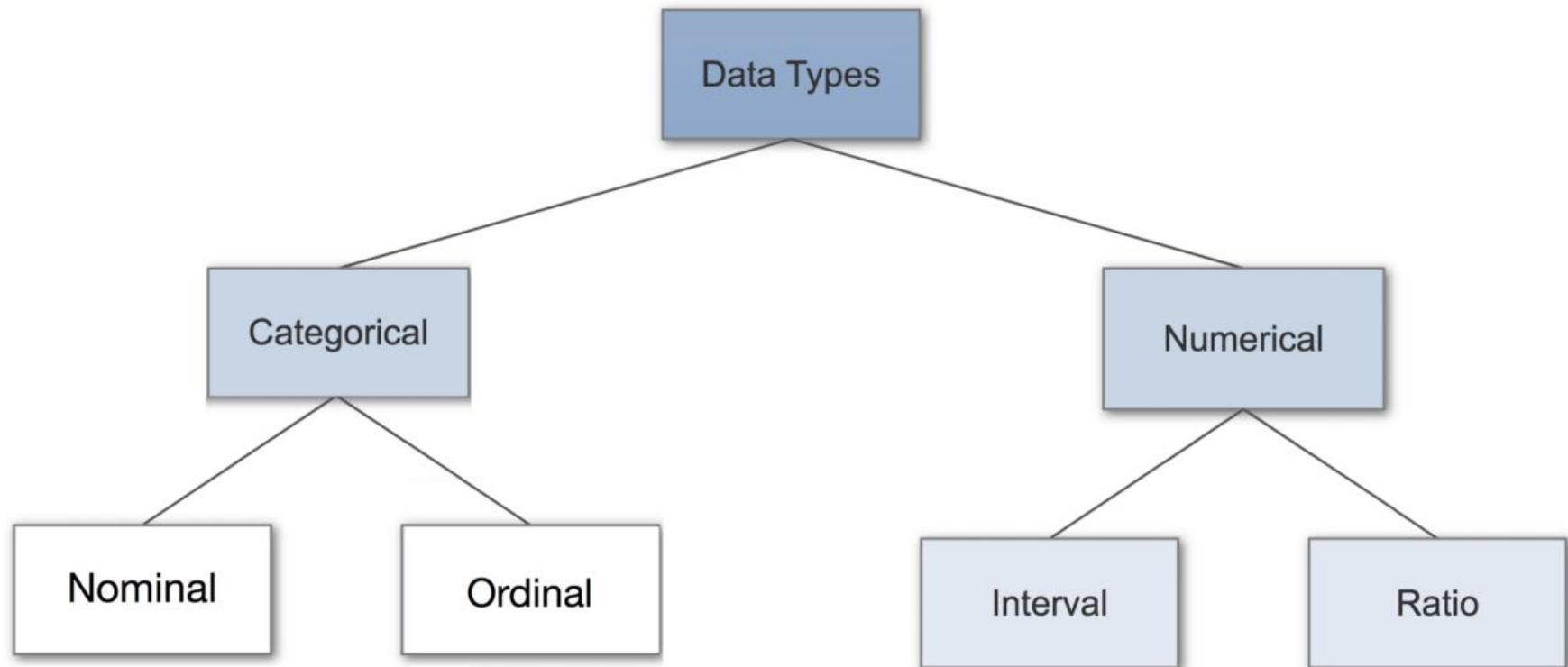


# Data

- has many sources, e.g.
  - sensor data
  - survey data
  - simulation data
  - social media data
  - textual data
  - financial data
  - multimedia data
  - ERP systems data
- but independent of the data source, each data point has a data type



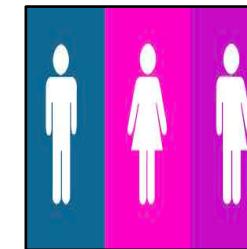
# Data Types



# Data Types

- **nominal**

- nominal scales are used for **labeling variables**, without **any quantitative value**
- **no numerical significance**
- nominal data that has **no order**
- scales could simply be called “**labels**”
- examples:
  - gender
  - hair color
  - race
  - marital status



# Data Types

- **ordinal**

- represent discrete and ordered units
- nearly the same as nominal data, except that it's ordering matters
- no distance between the different categories
- examples:
  - military rank
  - movie rating by number of stars
  - educational background
  - difficulty of cooking recipe

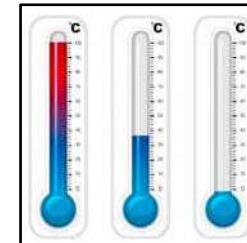
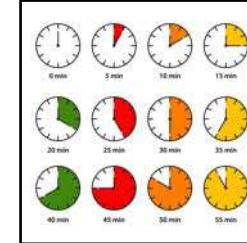


# Data Types

- numeric

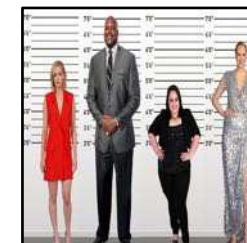
- interval

- numeric scales in which we know both the order and the exact differences between the values
    - don't have a „true zero“
    - we can add and subtract, but we cannot multiply, divide or calculate ratios



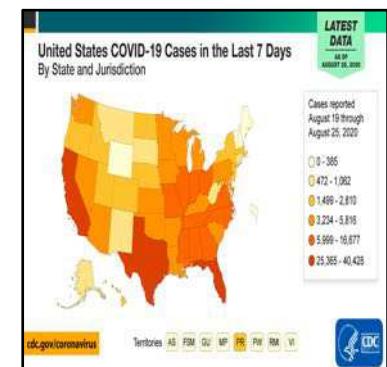
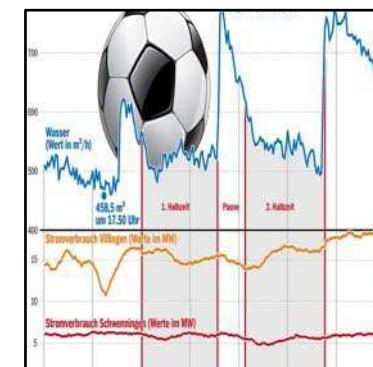
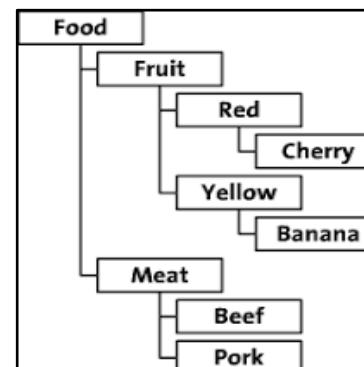
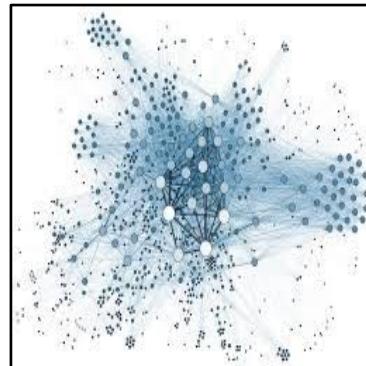
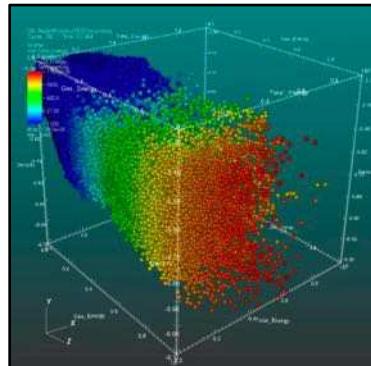
- ratio:

- are also ordered units that have the same difference
    - the same as interval values, with the difference that they do have an absolute zero



# Typical Data Classes

- multidimensional data
- network data
- hierarchical data
- time-series data
- geographic data



# Question

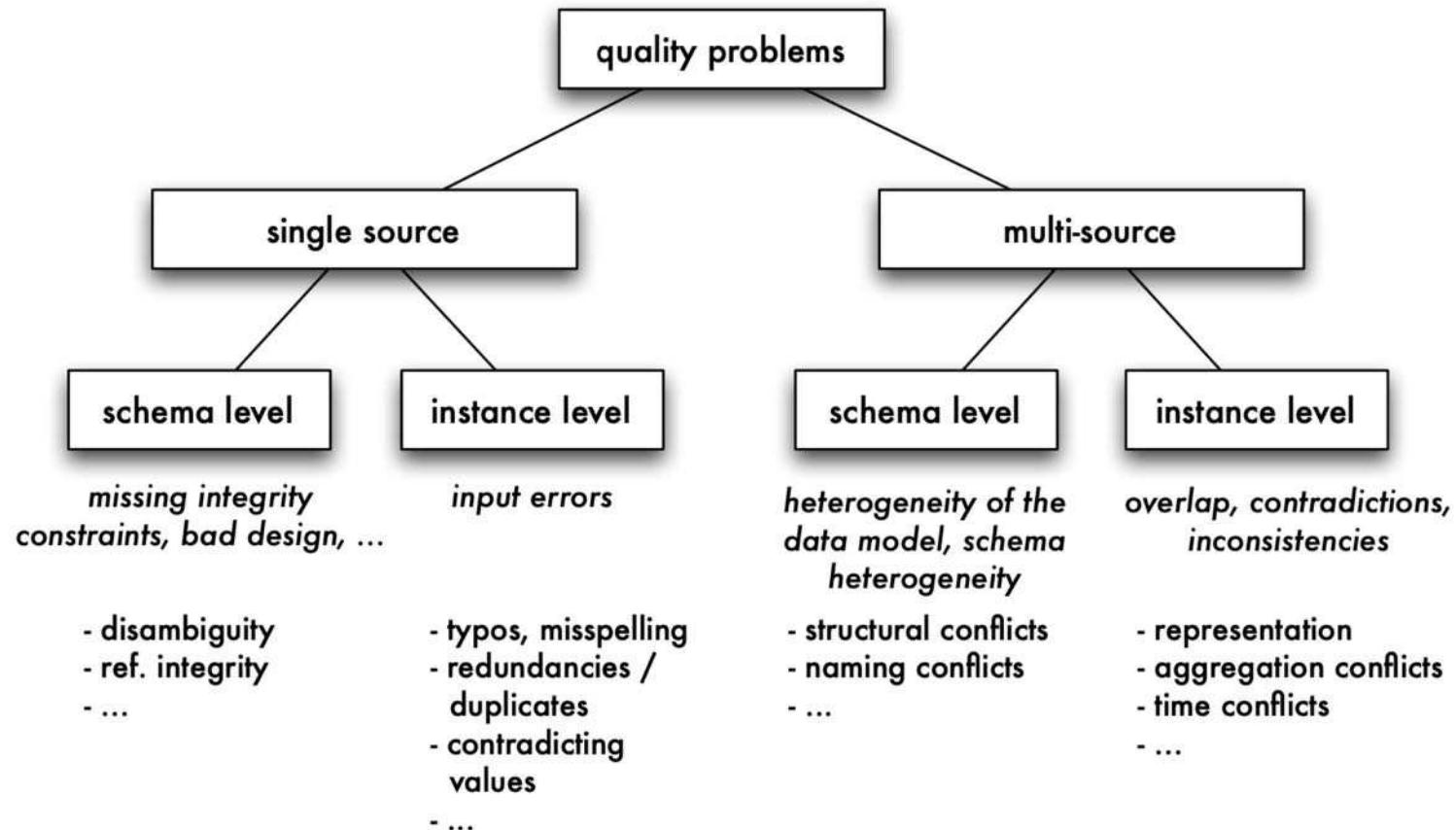
- Give examples for all of the data types / classes that you would use to describe the movement of a flock of lions.



- Nominal?
- Ordinal?
- Numeric?
- Network Data?
- Hierarchical?
- Time-Series?
- Geographic?

# Data Cleaning

- quality problems



# Data Cleaning

- process of improving the data quality
- low-quality data will lead to low-quality mining results
- removing or modifying incorrect or improperly formatted data
- example dataset “breaking bad”:

<b>title</b>	<b>date</b>	<b>rating</b>	<b>episode</b>	<b>season</b>	<b>length</b>
live free or die	01.11.2012	9.3	1	5	43
madrigal	06.12.2013	8.9	2	5	48
sunset	25.10.2011	9.3	6	3	47
grilled	20.12.2009	9.3	2	2	46

# Data Cleaning

- example dataset “breaking bad”:

<b>id</b>	<b>title</b>	<b>date</b>	<b>rating</b>	<b>episode</b>	<b>season</b>	<b>length</b>
1	live free or die	01.11.2012	9.3	1	5	43
2	madrigal	06.12.2013	89	2	5	48
3	sunset	25.10.2011	9.3	6	3	47
4	grilled	20.12.2009	9.3	2	2	46
5	down	42.12.2009	8.3	4	2	333
6	cancer man	19.10.2008	8.3	4	1	48
7	live fre or die	01.11.2012	9.3	1	5	43

<b>firstname</b>	<b>lastname</b>	<b>bdate</b>	<b>age</b>	<b>gender</b>	<b>phone</b>
bryan	cranston	03-07-1956	65	1	999-9999
aaron	paul	27-08-1979	44	m	777-53474
anna, gunn	gunn	08-11-1968	52	0	040-15627

# Data Cleaning

- example dataset “breaking bad”:

<b>id</b>	<b>title</b>	<b>date</b>	<b>rating</b>	<b>episode</b>	<b>season</b>	<b>length</b>
1	live free or die	01.11.2012	9.3	1	5	43
2	madrigal	06.12.2013	<b>89</b>	2	5	48
3	sunset	25.10.2011	9.3	6	3	47
4	grilled	20.12.2009	9.3	2	2	46
5	down	<b>42.12.2009</b>	8.3	4	2	<b>333</b>
6	cancer man	19.10.2008	8.3	4	1	48
7	<b>live fre or die</b>	01.11.2012	9.3	1	5	43

<b>firstname</b>	<b>lastname</b>	<b>bdate</b>	<b>age</b>	<b>gender</b>	<b>phone</b>
bryan	cranston	03-07-1956	65	male	<b>999-99999</b>
aaron	paul	<b>27-08-1979</b>	<b>24</b>	<b>0</b>	777-53474
<b>anna, gunn</b>	gunn	08-11-1968	52	female	040-15627
betsy	brandt	03-14-1973	47	<b>femalle</b>	333-49858

# Data Cleaning: (near) duplicates

- example dataset “breaking bad”:

<b>id</b>	<b>title</b>	<b>date</b>	<b>rating</b>	<b>episode</b>	<b>season</b>	<b>length</b>
1	live free or die	01.11.2012	9.3	1	5	43
2	madrigal	06.12.2013	89	2	5	48
3	sunset	25.10.2011	9.3	6	3	47
4	grilled	20.12.2009	9.3	2	2	46
5	down	42.12.2009	8.3	4	2	333
6	cancer man	19.10.2008	8.3	4	1	48
7	<b>live fre or die</b>	01.11.2012	9.3	1	5	43

# Data Cleaning: (near) duplicates

- compare the attributes of the tuple
- compare the content of the attributes
  - e.g., calculate levenshtein distance between the content

	1	2	3	4	5	6	7
live free or die	0	15	15	14	15	13	<b>1</b>
madrigal	15	0	8	7	7	7	14
sunset	15	8	0	6	6	8	14
grilled	14	7	6	0	7	10	13
down	15	7	6	7	0	9	14
cancer man	13	7	8	10	9	0	12
live fre or die	<b>1</b>	14	14	13	14	12	0

# Data Cleaning: missing values

- example dataset “breaking bad”:

title	date	rating	episode	season	length
live free or die	01.11.2012	9.3	1	5	43
madrigal	06.12.2013	8.9	2	5	48
sunset	25.10.2011		6	3	47
grilled	20.12.2009	9.3	2	2	46
down	22.12.2009	8.3	4	2	
cancer man	19.10.2008	8.3	4	1	48
shotgun	09.11.2012	8.7	5	4	47

# Data Cleaning: missing values

- example dataset “breaking bad”:

title	date	rating	episode	season	length
live free or die	01.11.2012	9.3	1	5	43
madrigal	06.12.2013	8.9	2	5	48
sunset	25.10.2011		0	3	47
grilled	20.12.2009	9.3	2	2	46
down	22.12.2009	8.3	4	2	
cancer man	19.10.2008	8.3	4	1	48
shotgun	09.11.2012	8.7	5	4	47

ignore the tuple

# Data Cleaning: missing values

- example dataset “breaking bad”:

title	date	rating	episode	season	length
live free or die	01.11.2012	9.3	1	5	43
madrigal	06.12.2013	8.9	2	5	48
sunset	25.10.2011	9.3	6	3	47
grilled	20.12.2009	9.3	2	2	46
down	22.12.2009	8.3	4	2	47
cancer man	19.10.2008	8.3	4	1	48
shotgun	09.11.2012	8.7	5	4	47

fill in the missing value manually

# Data Cleaning: missing values

- example dataset “breaking bad”:

title	date	rating	episode	season	length
live free or die	01.11.2012	9.3	1	5	43
madrigal	06.12.2013	8.9	2	5	48
sunset	25.10.2011	-1	6	3	47
grilled	20.12.2009	9.3	2	2	46
down	22.12.2009	8.3	4	2	-1
cancer man	19.10.2008	8.3	4	1	48
shotgun	09.11.2012	8.7	5	4	47

use a global constant

# Data Cleaning: missing values

- example dataset “breaking bad”:

title	date	rating	episode	season	length
live free or die	01.11.2012	9.3	1	5	43
madrigal	06.12.2013	8.9	2	5	48
sunset	25.10.2011	8.8	6	3	47
grilled	20.12.2009	9.3	2	2	46
down	22.12.2009	8.3	4	2	46.5
cancer man	19.10.2008	8.3	4	1	48
shotgun	09.11.2012	8.7	5	4	47

use the attribute mean

# Data Cleaning: missing values

- example dataset “breaking bad”:

title	date	rating	episode	season	length
live free or die	01.11.2012	9.3	1	5	43
madrigal	06.12.2013	8.9	2	5	48
sunset	25.10.2011	9.3	6	3	47
grilled	20.12.2009	9.3	2	2	46
down	22.12.2009	8.3	4	2	47
cancer man	19.10.2008	8.3	4	1	48
shotgun	09.11.2012	8.7	5	4	47

use the most probable value

# Data Cleaning: missing values

- + can be easily done
- + no computational effort
- - loss of information
- - unnecessary if the attribute is not needed

ignore the tuple

- + for small datasets effective
- + “real” value
- - not feasible for large datasets
- - time consuming
- - error-prone

enter value manually

- + simple to implement
- - not the most accurate approximation of the value

use attribute mean

- + can be easily done
- + missing values are marked
- - values can not be used in algorithms

use a global constant

- + most accurate approximation of the value
- - most computational effort

use most probable value

# Data Cleaning: noisy data

- example dataset “breaking bad”:

title	date	rating	episode	season	length
live free or die	01.11.2012	9.3	1	5	43
madrigal	06.12.2013	8.9	2	5	48
sunset	25.10.2011	9.3	6	3	47
grilled	20.12.2009	9.3	2	2	46
down	22.12.2009	8.3	4	2	<b>91</b>
cancer man	19.10.2008	8.3	4	1	48
shotgun	09.11.2012	8.7	5	4	47

# Data Cleaning: noisy data

- example dataset “breaking bad”:



# Data Cleaning: noisy data

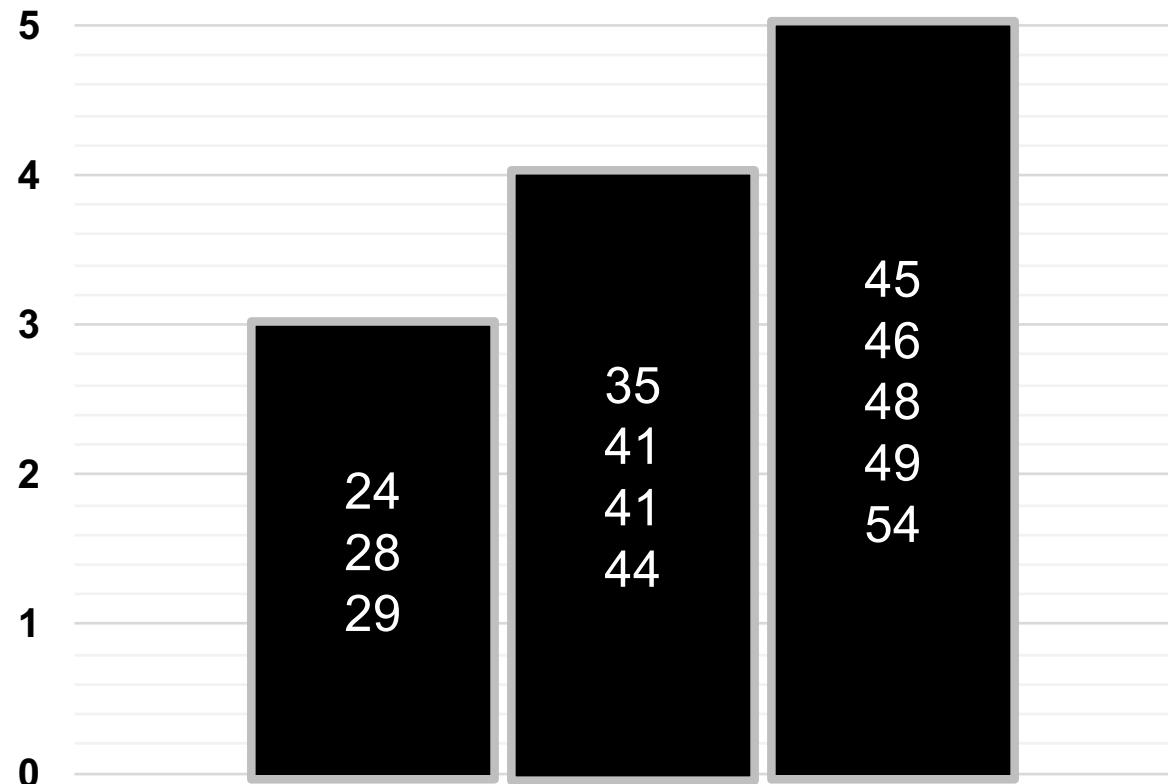
- binning
  - sort data and partition into (equi-depth) bins and then smooth by bin means, bin median, bin boundaries, etc.
- regression
  - smooth by fitting a regression function
- clustering
  - cluster data and remove outliers
  - automatically
  - via manual exploration

# Data Cleaning: noisy data

- **equal-width binning**
  - divides the range into N intervals of equal size
  - width of intervals:  $\text{width} = (\text{max} - \text{min}) / N$
  - simple method
  - outliers may dominate result
- **equal-depth binning**
  - divides the range into N intervals
  - each interval contains approximately the same number of records
  - skewed data is also handled well

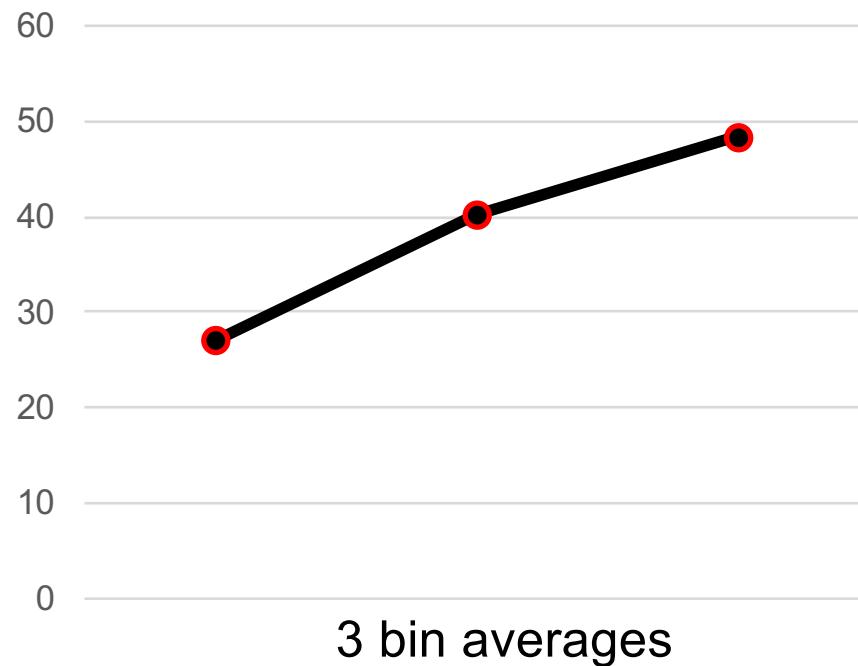
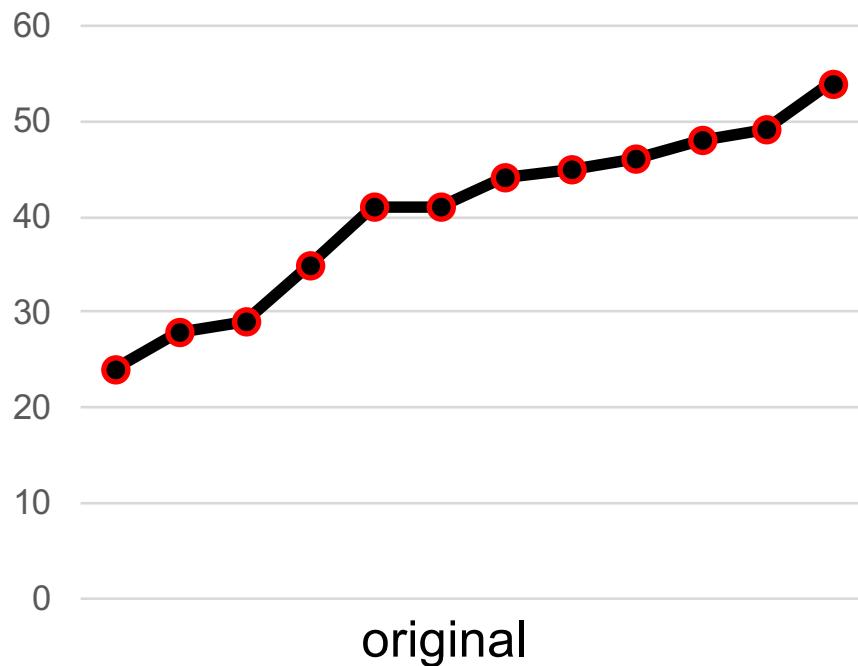
# Data Cleaning: noisy data

- equal-width binning
  - sorted price values: 24, 28, 29, 35, 41, 41, 41, 44, 45, 46, 48, 49, 54
  - $N = 3$  (user-defined)
  - width =  $54 - 24 / 3 = 10$



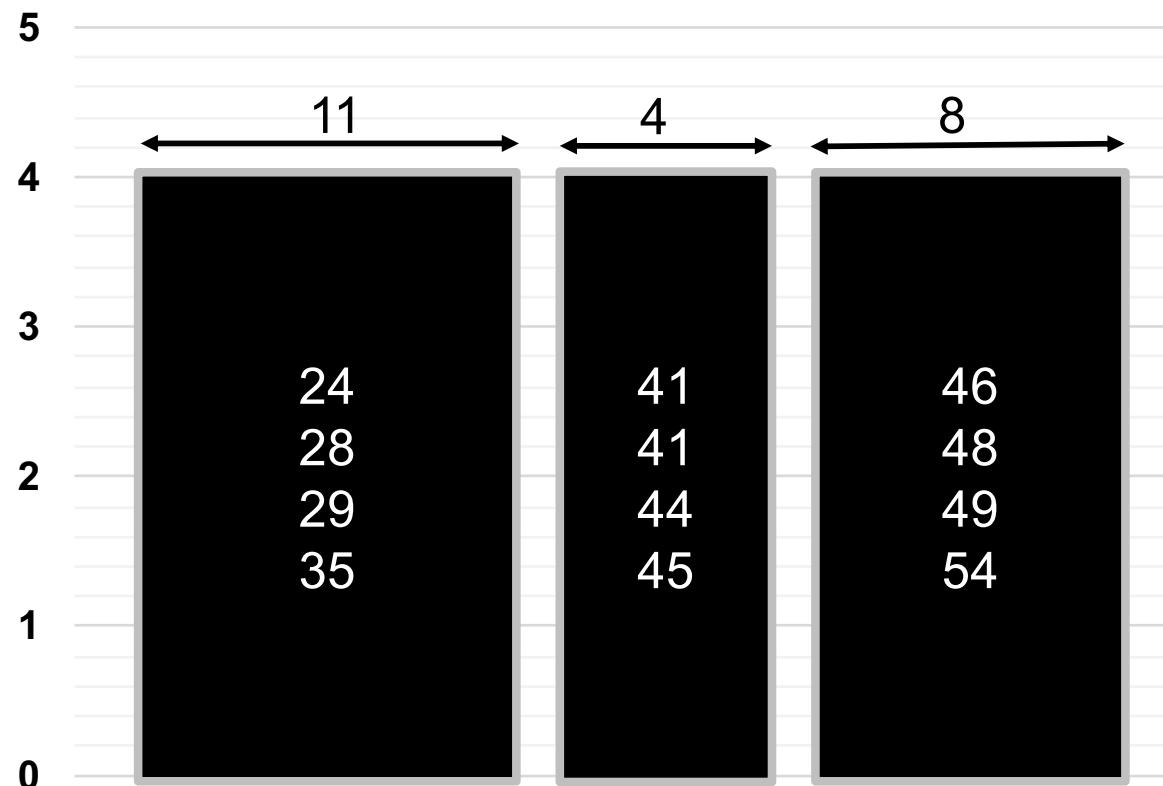
# Data Cleaning: noisy data

- equal-width binning
  - sorted price values: 24, 28, 29, 35, 41, 41, 44, 45, 46, 48, 49, 54
  - $N = 3$  (user-defined)
  - width =  $54 - 24 / 3 = 10$



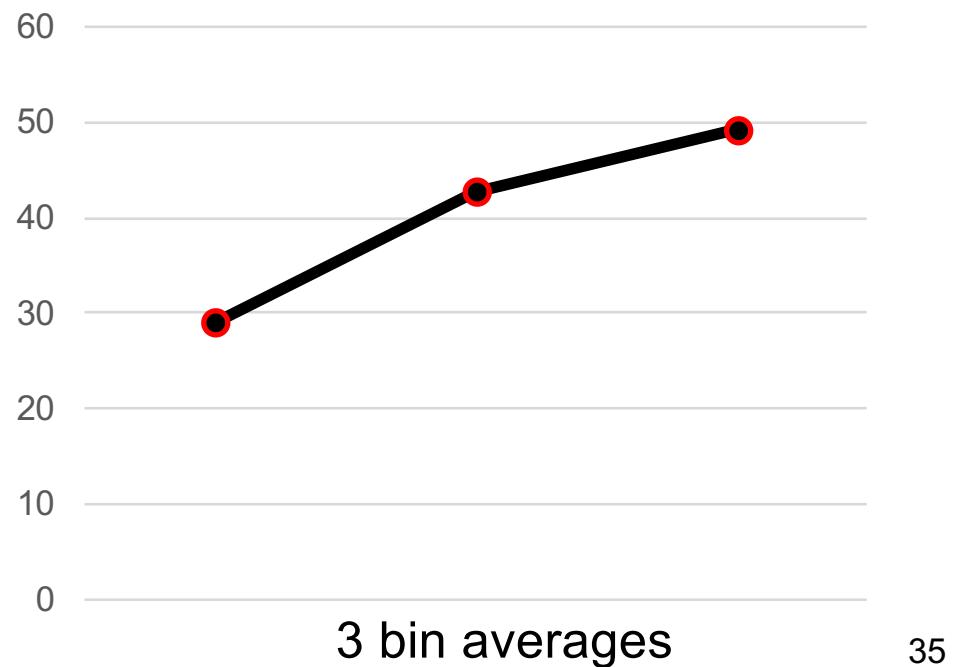
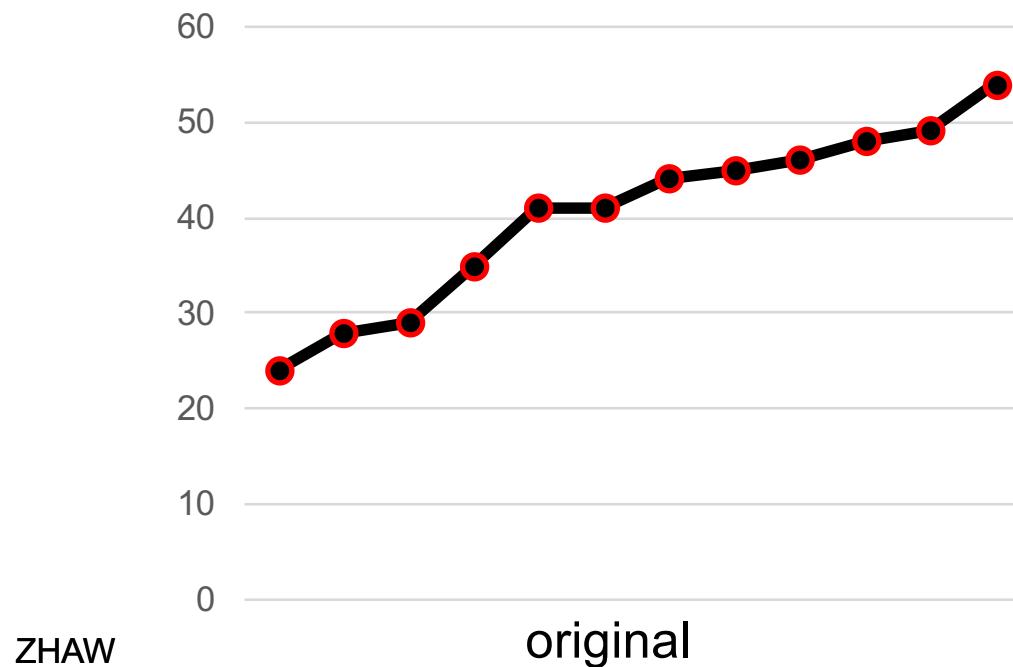
# Data Cleaning: noisy data

- equal-depth binning
  - sorted price values: 24, 28, 29, 35, 41, 41, 44, 45, 46, 48, 49, 54
  - $N = 3$



# Data Cleaning: noisy data

- equal-depth binning
  - sorted price values: 24, 28, 29, 35, 41, 41, 44, 45, 46, 48, 49, 54
  - $N = 3$



# Data Cleaning: noisy data

- equal-depth binning
  - sorted price values: 24, 28, 29, 35, 41, 41, 44, 44, 45, 46, 46, 48, 49, 54
  - $N = 3$
  - partition into 3 equal-depth bins:
    - [24, 28, 29, 35], [41, 41, 44, 45], [46, 48, 49, 54]
  - smoothing by bin means:
    - replace each value by the mean value of the bin)
    - [29, 29, 29, 29], [43, 43, 43, 43], [49, 49, 49, 49]
  - smoothing by bin boundaries:
    - replace each value by closest boundary value
    - [24, 24, 24, 35], [41, 41, 45, 45], [46, 46, 46, 54]

# Lessons Learned

- understanding of the KDD process
- distinguish between different data types and data classes
- understanding of data preprocessing, cleaning, and exploration



# Questions and Answers



# Machine Learning and Data Mining

## V03: Data Processing Part 02

Lecturer: Dr. Andreas Weiler



# Repetition

... REPETITION ...

→ IS THE MOTHER →

of learning

# Course Content IT20a\_WIN

Week	Lecture	Lab
KW08	Introduction	Lab 01
KW09	Data Processing Part 01	Lab 02
KW10	Carnival Monday	
KW11	Data Processing Part 02	Lab 03
KW12	Evaluation	Lab 04
KW13	Recommender Systems	<b>Graded Lab 01</b> <span style="border: 1px solid red; padding: 2px;">IMPORTANT</span>
KW14	Association Rules	<b>Graded Lab 01</b> <span style="border: 1px solid red; padding: 2px;">IMPORTANT</span>
KW15	Linear and Logistic Regression	Lab 05
KW16	Easter Monday	
KW17	Support Vector Machines	<b>Graded Lab 02</b> <span style="border: 1px solid red; padding: 2px;">IMPORTANT</span>
KW18	Decision Trees and Naïve Bayes	<b>Graded Lab 02</b> <span style="border: 1px solid red; padding: 2px;">IMPORTANT</span>
KW19	Partition Clustering	Lab 06
KW20	Hierarchical and Density Clustering	Lab 07
KW21	Neural Networks and Closing	Review and Questions for Exam

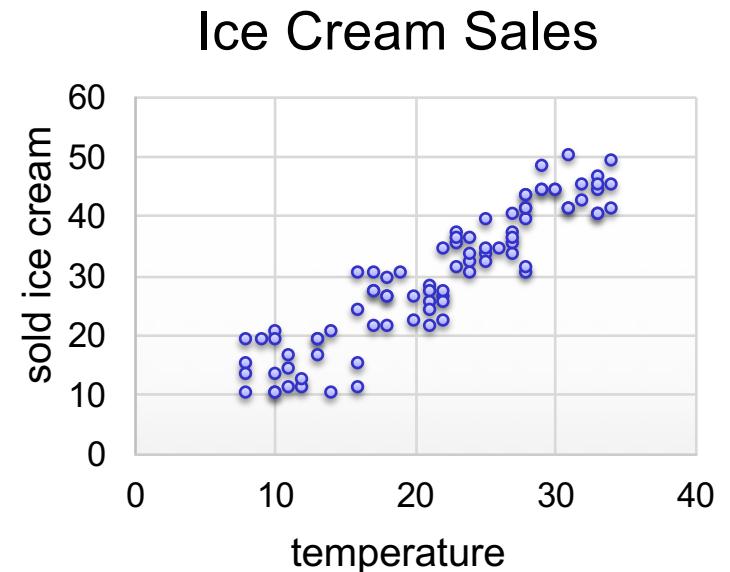
# Learning Objectives

- understanding of data preprocessing and cleaning
- understanding of data normalization, sampling, and partitioning



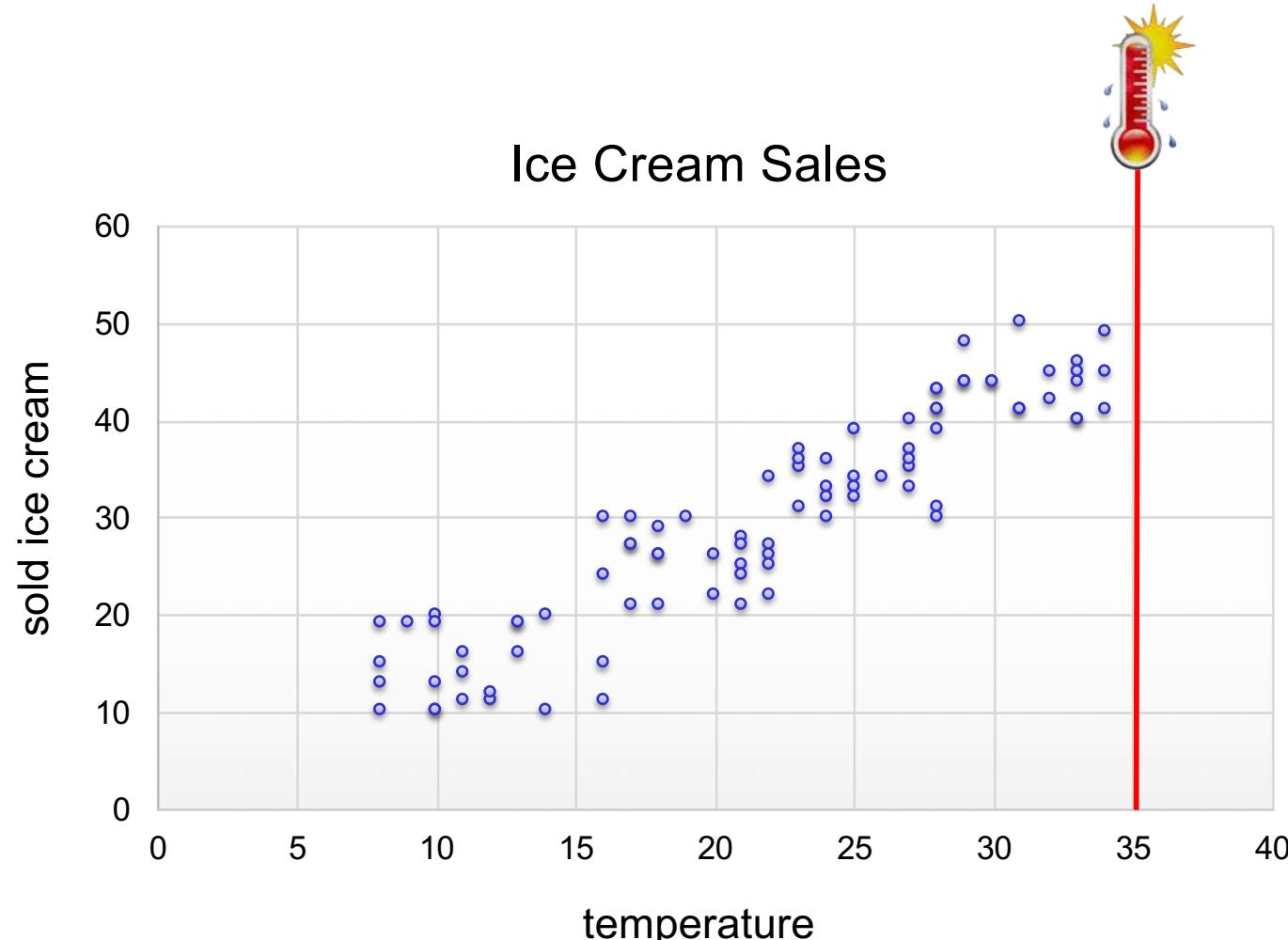
# Data Cleaning: noisy data

- linear regression
  - tries to discover the parameters of the straight line equation that best fits the data point
  - smooth out noise by fitting data to a function
  - can also be used to fill in missing values
  - calculation of relationships between input and output values
  - training of a model with known values
  - prediction of unknown values by using the model



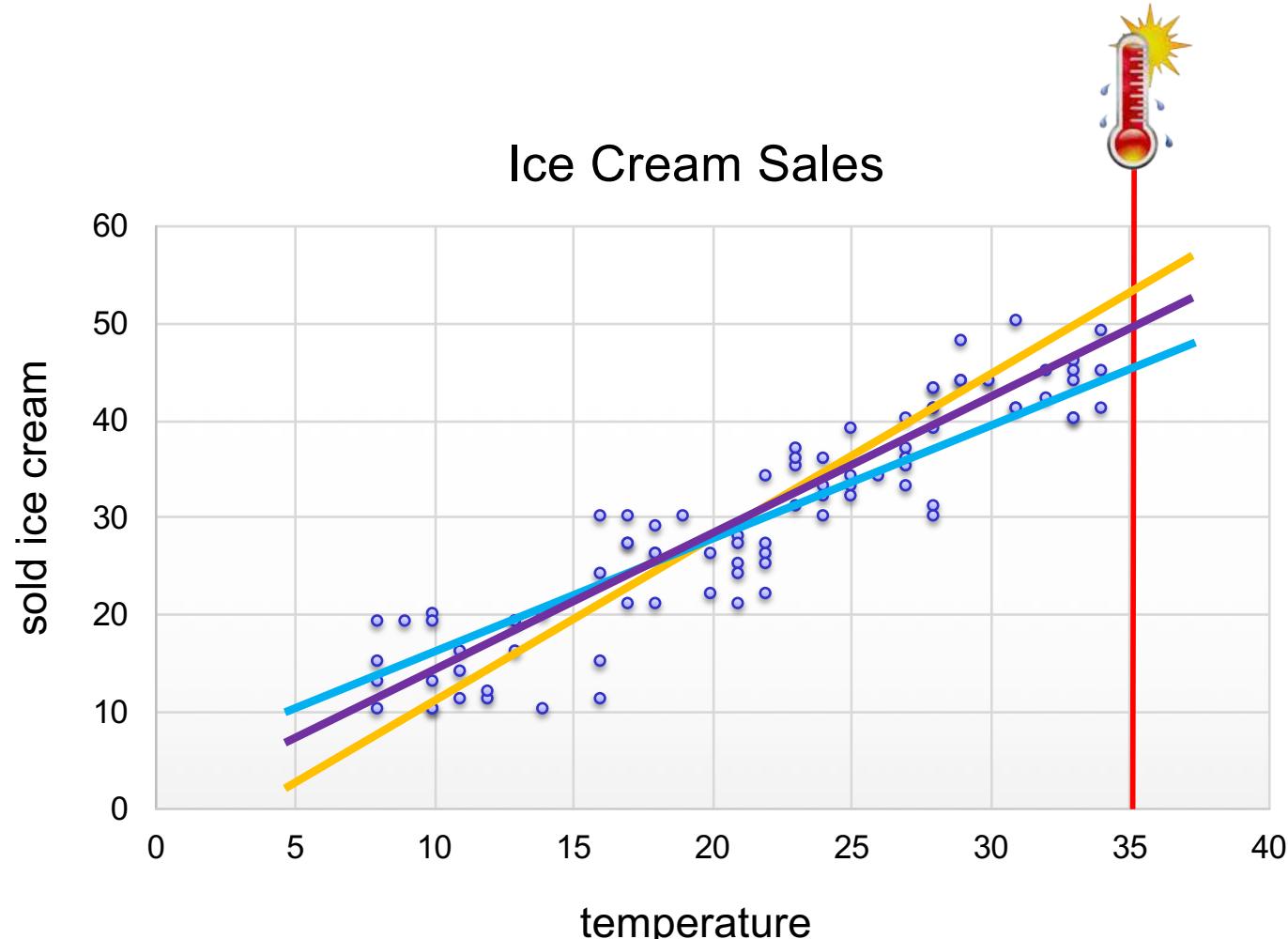
# Data Cleaning: noisy data

- linear regression
  - predicting a "sold ice cream" value for 35 degrees



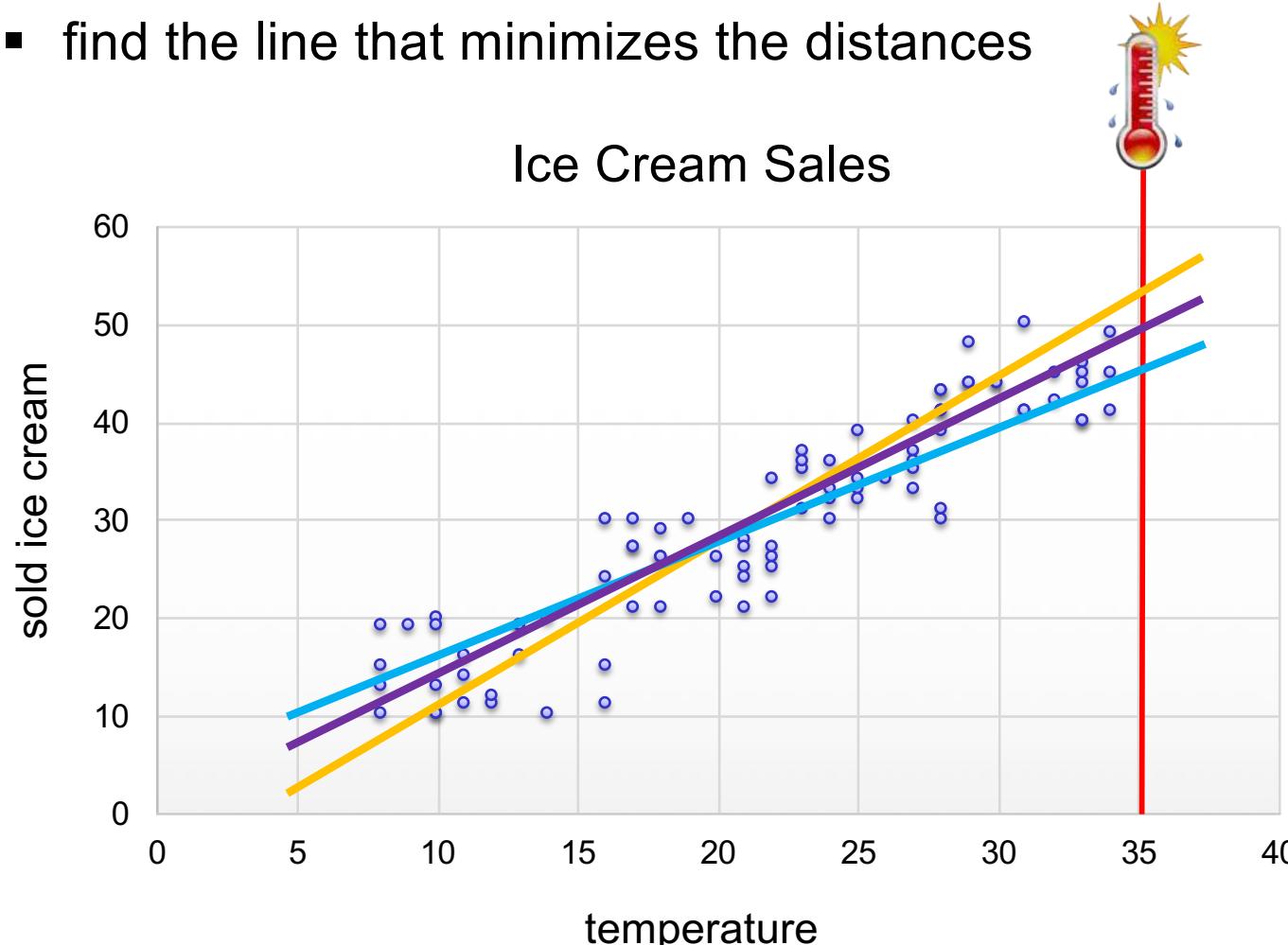
# Data Cleaning: noisy data

- linear regression
    - many possibilities for regression lines



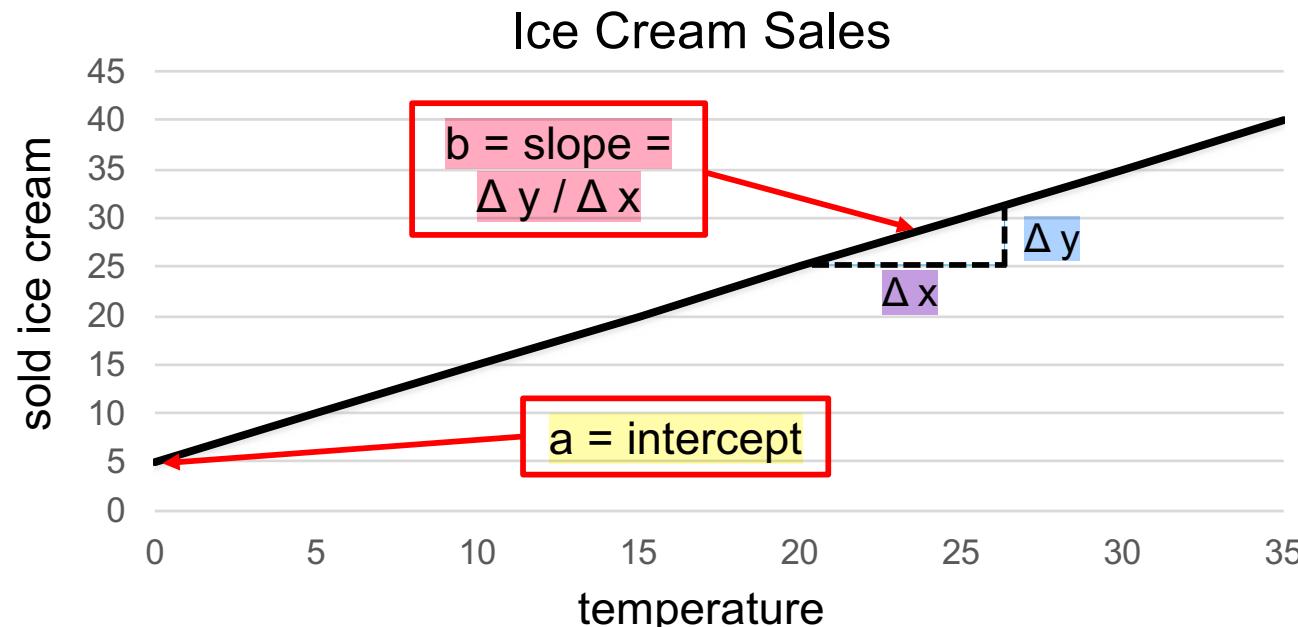
# Data Cleaning: noisy data

- linear regression
  - calculate distance between each data point and the “line”
  - find the line that minimizes the distances



# Data Cleaning: noisy data

- linear regression
  - expression describing a straight line:  $y = a + bx$   
( $y$  = outcome;  $x$  = input)
  - $b$ : indicates the slope of the line
  - $a$ : indicates the intersection with the y-axis



# Data Cleaning: noisy data

- linear regression
  - calculation of the slope

$$b = \frac{\sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^n (x_i - \bar{x})^2}$$

- $\bar{x}$  is the mean value of  $x$
- $\bar{y}$  is the mean value of  $y$
- find  $a$  when  $b$  is known:  $a = \bar{y} - b\bar{x}$

# Data Cleaning: noisy data

- linear regression
  - example

<b>temperature</b>	x	10	15	17	23	25	29	31	32
<b>sold ice cream</b>	y	15	22	28	34	38	44	48	55

- calculation of the slope with  $\bar{x} = 22.75$  and  $\bar{y} = 30.75$
- $b = \frac{(10 - 22.75) * (15 - 30.75) + (15 - 22.75) * (22 - 30.75) + \dots}{(10 - 22.75)^2 + (15 - 22.75)^2 + \dots}$
- $b = 1.6560088202866594$

# Data Cleaning: noisy data

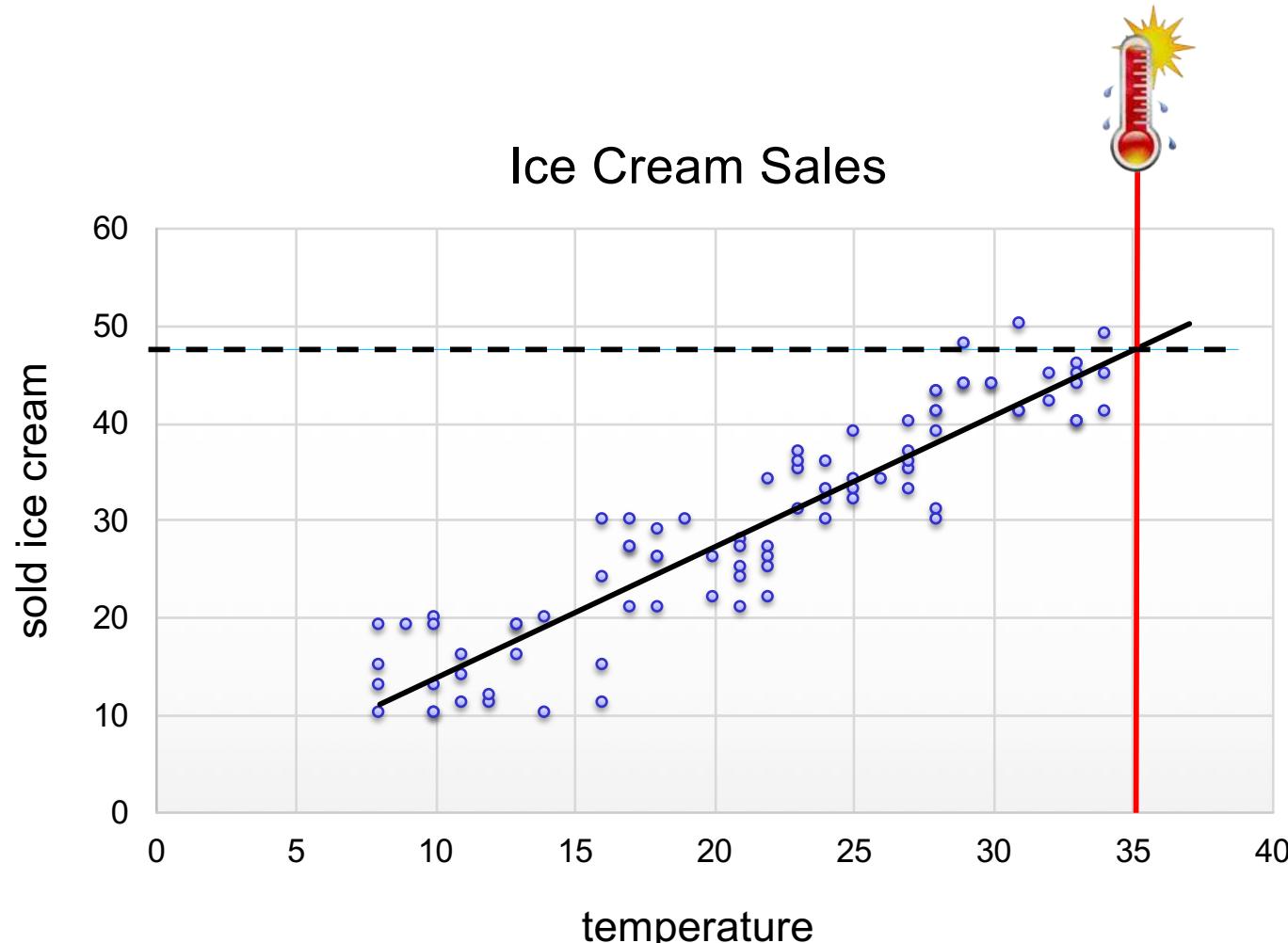
- linear regression
  - example

<b>temperature</b>	x	10	15	17	23	25	29	31	32
<b>sold ice cream</b>	y	15	22	28	34	38	44	48	55

- find a when b is known:
  - $a = 30.75 - 1.656 * 22.75$
  - $a = -2.1742006615215033$
- predicting a "sold ice cream" value for 35 degrees
  - $y = a + bx$
  - $y = -2.1742006615215033 + (1.6560088202866594 * 35)$
  - $y = \underline{\underline{55.786}}$

# Data Cleaning: noisy data

- linear regression
  - predicting a "sold ice cream" value for 35 degrees visually



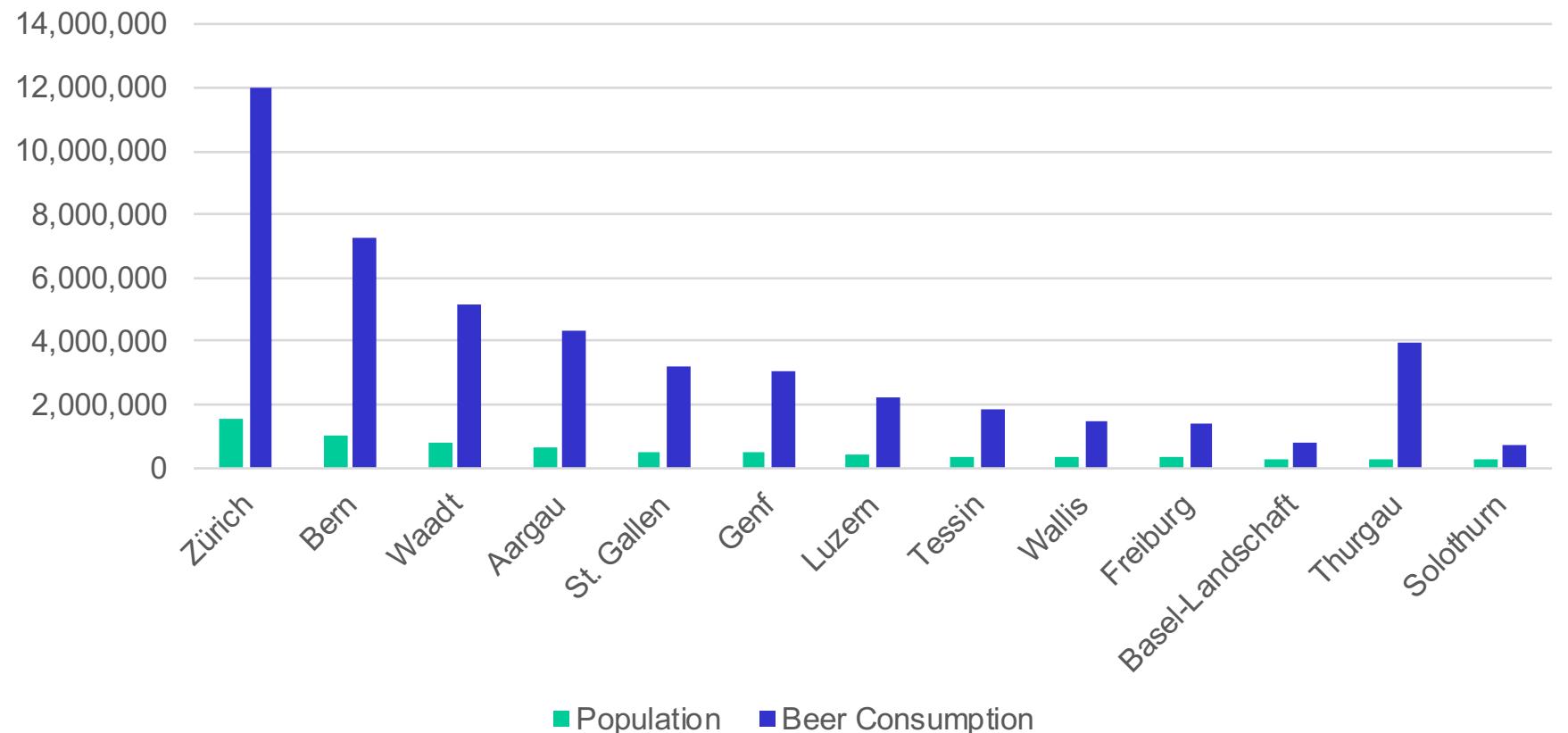
# Data Normalization

- change the values of numeric columns to a common scale, without distorting differences in the ranges of values

Canton	Population	Canton	Beer Consumption
Zürich	1539275	Zürich	12021124
Bern	1039474	Bern	7232844
Waadt	805098	Waadt	5195496
Aargau	685845	Aargau	4342480
St. Gallen	510734	St. Gallen	3195642
Genf	504128	Genf	3031348
Luzern	413120	Luzern	2212584
Tessin	351491	Tessin	1855445
Wallis	345525	Wallis	1522087
Freiburg	321783	Freiburg	1440590
Basel-Landschaft	289468	Basel-Landschaft	807930
Thurgau	279547	Thurgau	3936703
Solothurn	275247	Solothurn	716128

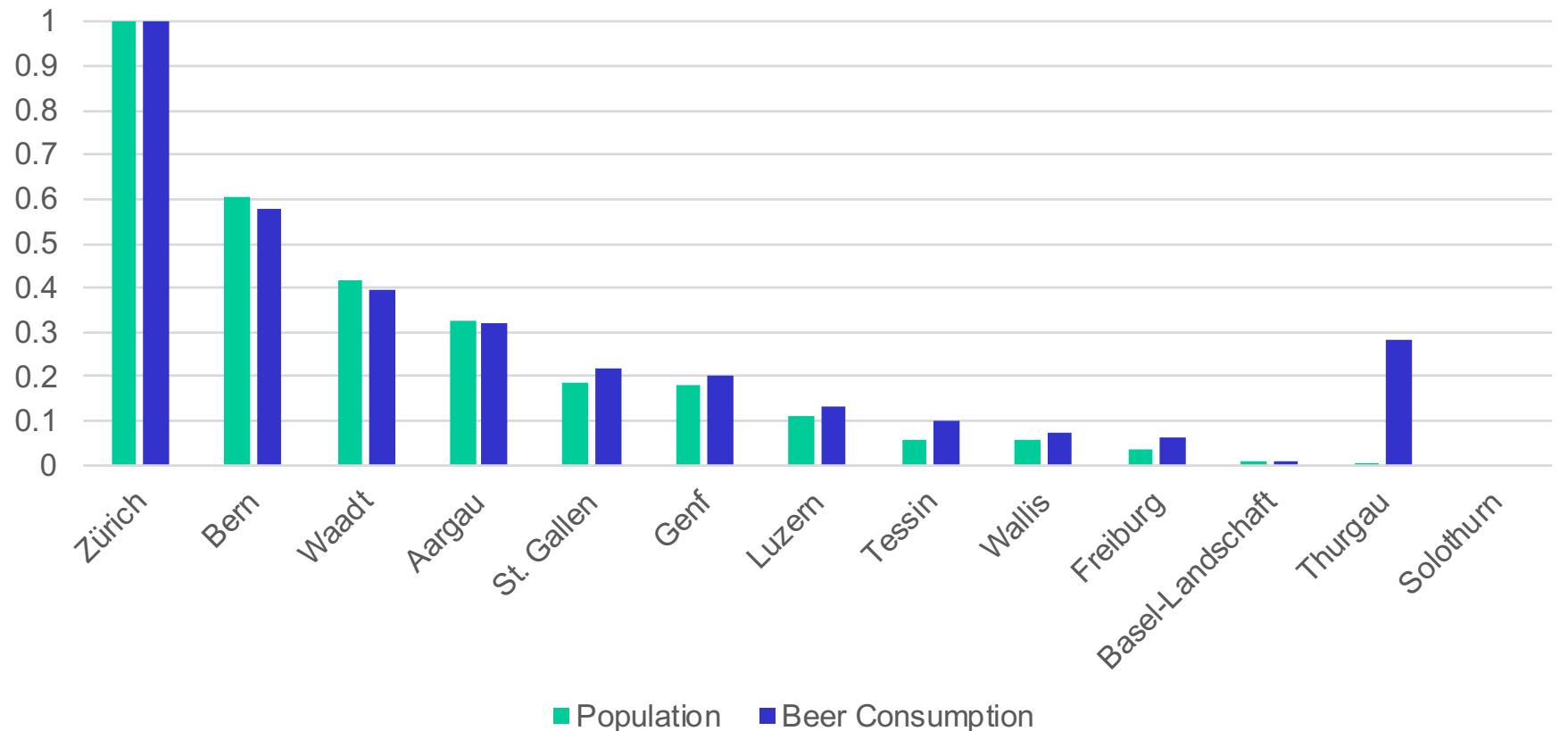
# Data Normalization

- change the values of numeric columns to a common scale,  
without distorting differences in the ranges of values



# Data Normalization

- change the values of numeric columns to a common scale, without distorting differences in the ranges of values



# Data Normalization

- linear normalization

$$f_{lin}(v) = \frac{v - min}{max - min}$$

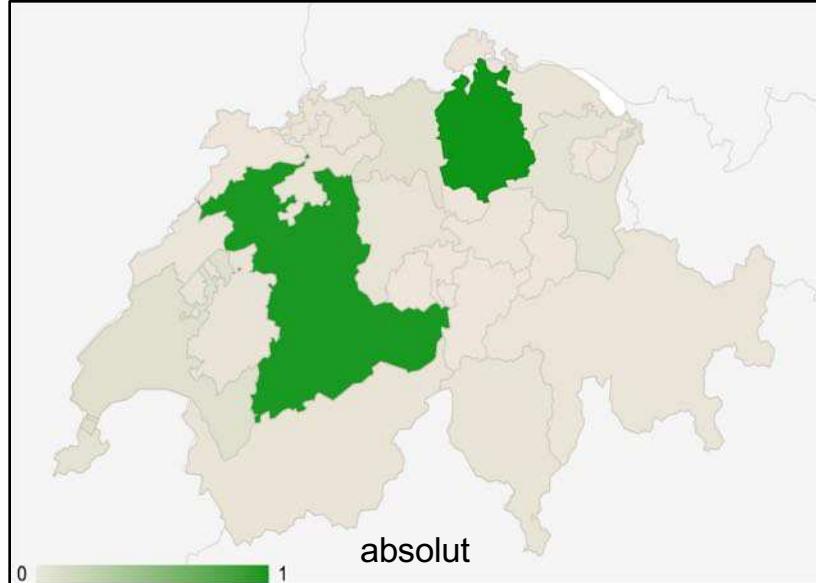
- square root normalization

$$f_{sq}(v) = \frac{\sqrt{v} - \sqrt{min}}{\sqrt{max} - \sqrt{min}}$$

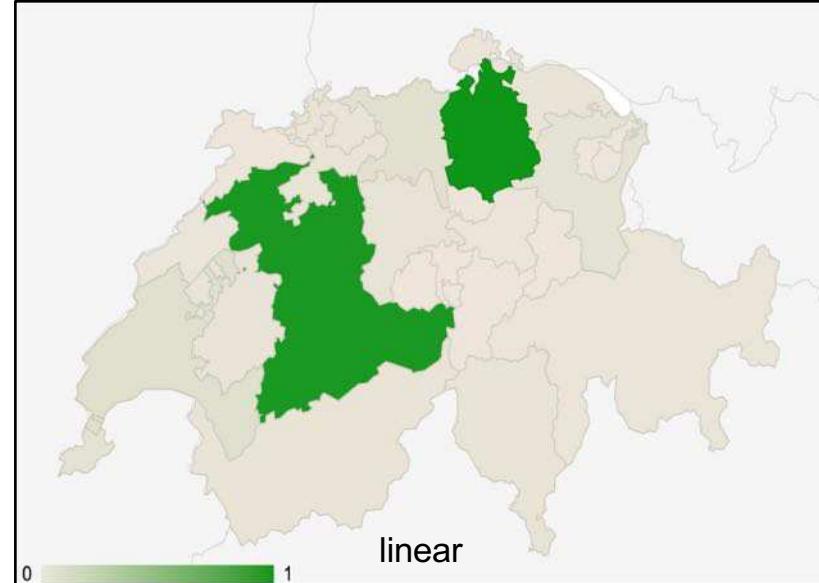
- logarithmic normalization

$$f_{ln}(v) = \frac{\ln(v) - \ln(min)}{\ln(max) - \ln(min)}$$

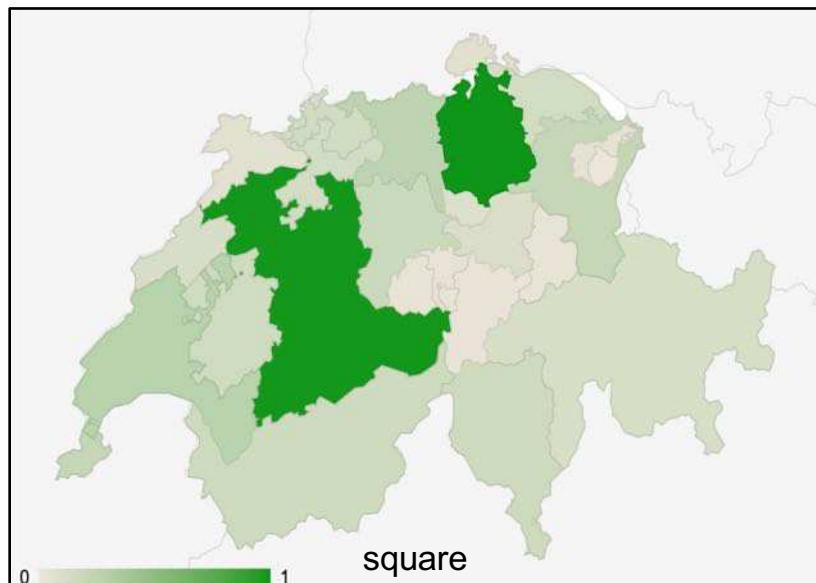
# Data Normalization



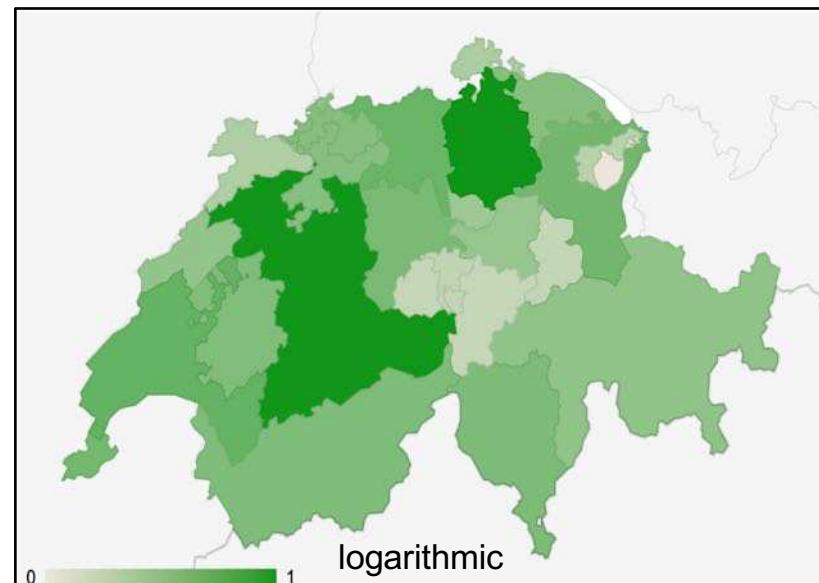
absolut



linear



square



logarithmic

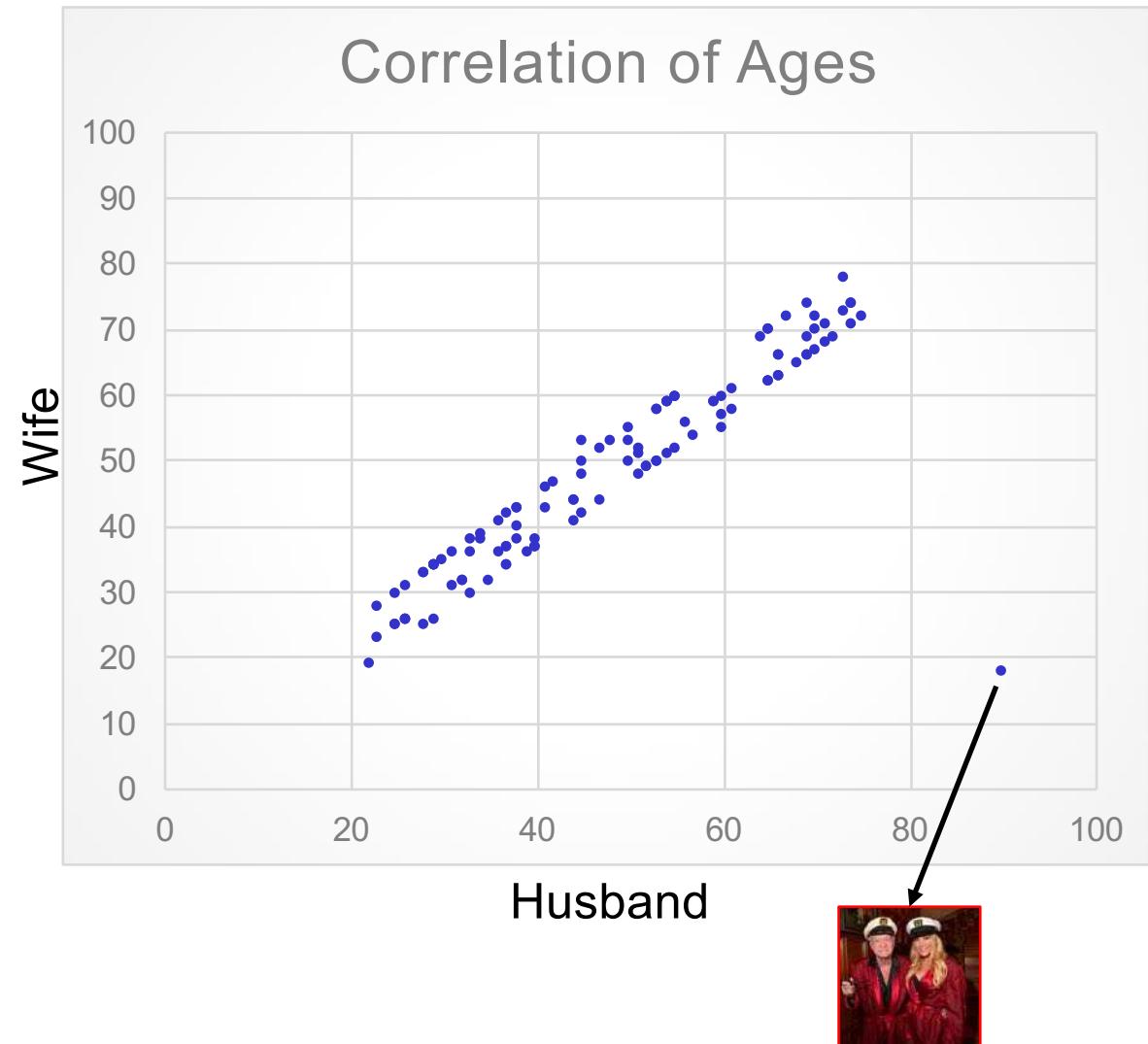
# Data Sampling

- represent a large dataset by a smaller subset
  - speed up automatic calculations
  - the subset should sufficiently represent the whole data set
- 
- example:
    - statistics about drug consumption at an electro festival
    - total of 100.000 electro fans
    - the subset should consist of 100 data points at the end



# Data Sampling

<b>Wife</b>	<b>Husband</b>
45	53
60	55
32	32
33	36
45	48
70	72
50	53
34	38
51	52
40	38
41	43
38	40



# Data Sampling

<b>Wife</b>	<b>Husband</b>
45	32
60	32
32	32
33	32
45	32
70	32
50	32
34	32
51	32
40	32
41	32
38	32

dataset 1

<b>Wife</b>	<b>Husband</b>
45	53
45	55
45	32
45	36
45	48
45	72
45	53
45	38
45	52
45	38
45	43
45	40

dataset 2

<b>Wife</b>	<b>Husband</b>
22	20
24	22
26	24
28	26
30	28
32	30
34	32
36	34
38	36
40	38
42	40
44	42

dataset 3

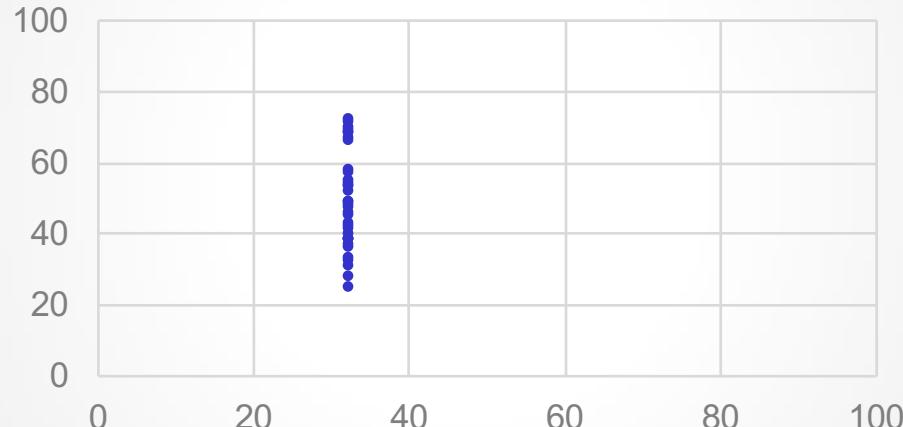
<b>Wife</b>	<b>Husband</b>
20	20
35	25
40	30
25	21
30	23
38	28
45	35
42	32
45	40
42	45
37	50
32	55

dataset 4

# Data Sampling

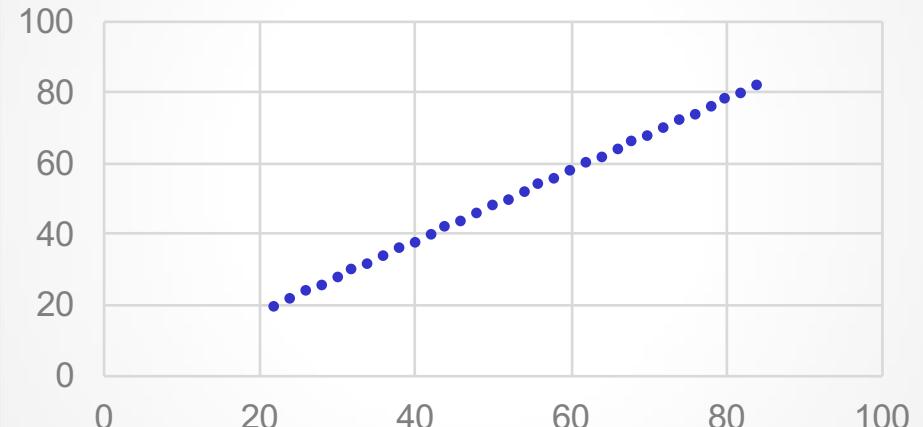
1

Correlation of Ages



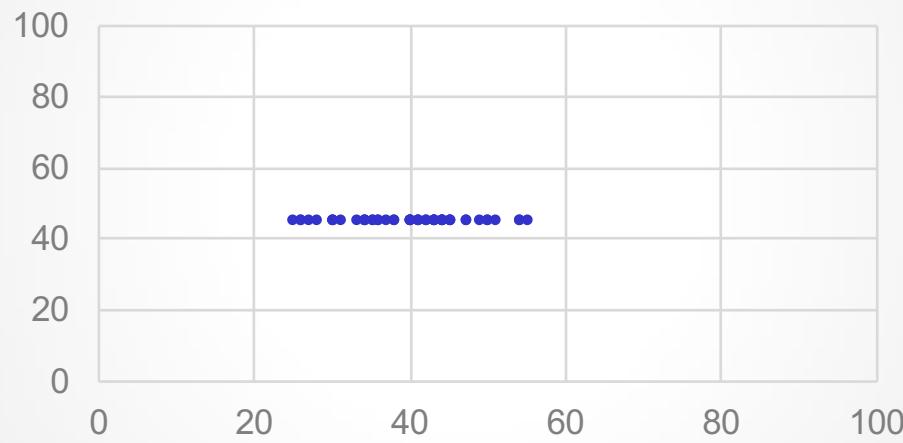
3

Correlation of Ages



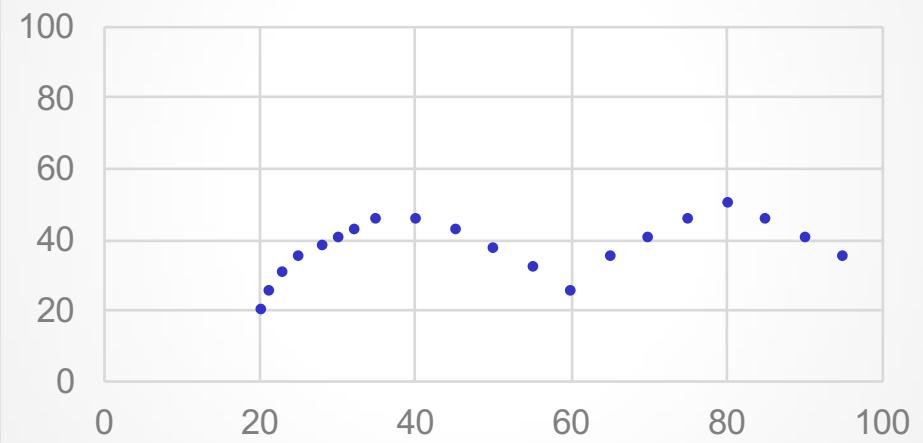
2

Correlation of Ages



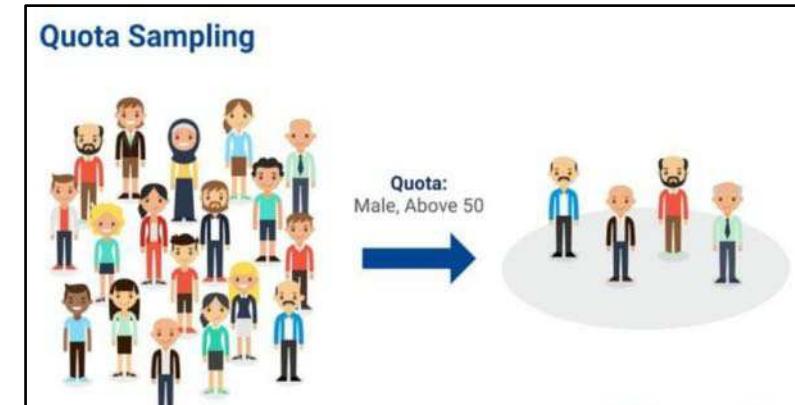
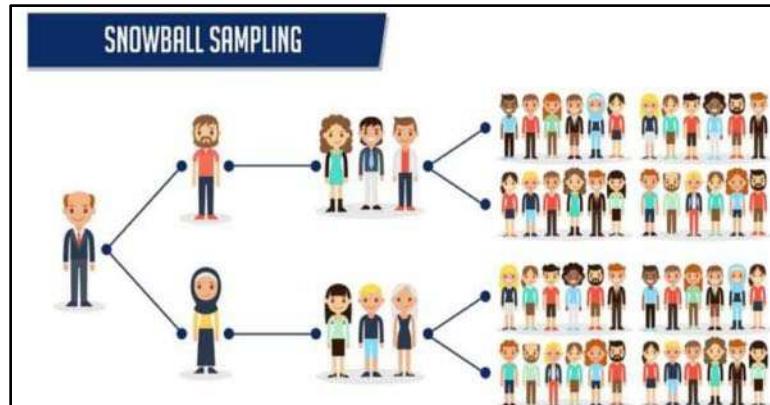
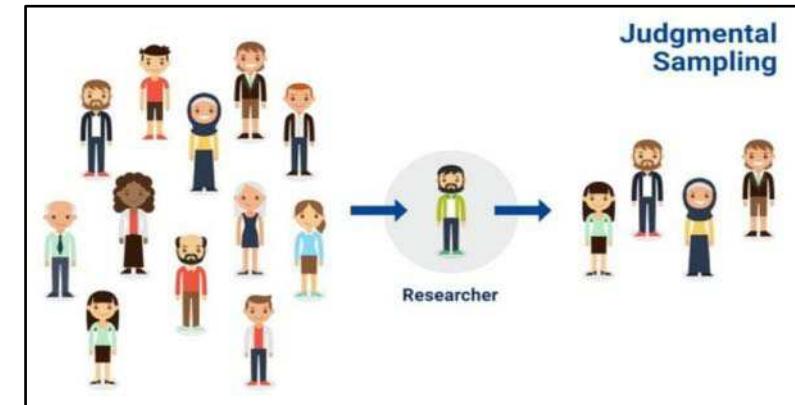
4

Correlation of Ages



# Data Sampling

- non-probabilistic sample
  - sample selected on some non-random basis
  - e.g., convenience, judgement, snowball, quota, volunteer



# Data Sampling

- non-probabilistic sample
  - sample selected on some non-random basis
  - e.g., convenience, judgement, snowball, quota, volunteer

- ease of access
- select haphazardly those cases that are easiest to obtain
- e.g., just take the first 1000 data objects

convenience

- objects are chosen on the basis of the experts knowledge and judgment
- select the cases that will best enable you to solve your task

judgement

- purely based on referrals
- like a snowball increasing in size until enough objects are collected
- e.g., select two objects and all other objects are somehow related to them

snowball

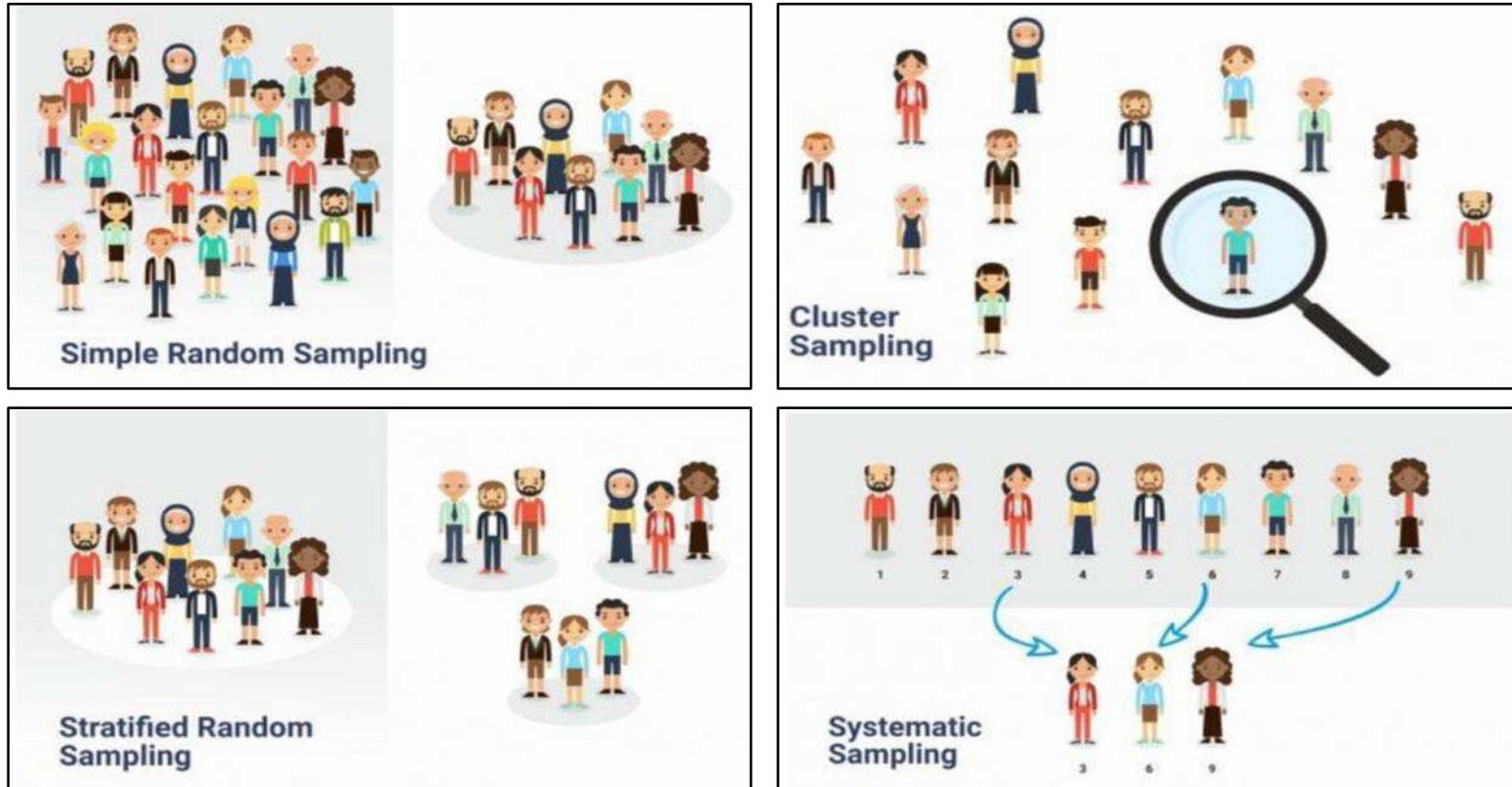
- sample is selected based on attribute values
- type of stratified sampling but with non-randomly chosen objects

e.g. Male, above 50

quota

# Data Sampling

- probabilistic sample
  - every element has an equal chance of being selected
  - e.g., simple, systematic, stratified, or cluster random sampling



# Data Sampling

- probabilistic sample - simple random sampling
  - a random sampling strategy is the least biased sampling method
  - every element can be selected with the same probability
  - chance that the sample does not describe the whole population
  - e.g., randomly choose 4 objects of the below data set

	1	2	3	4	5	6	7	8	9
age	34	22	42	33	55	81	61	24	19
gender	M	F	M	F	F	F	M	M	M
degree	BSc	PhD	High School	BSc	PhD	High School	High School	High School	MSc
nationality	german	german	german	japanese	german	german	german	japanese	japanese
martial status	married	single	single	divorced	single	single	divorced	married	married

# Data Sampling

- probabilistic sample – systematic random sampling
  - order the data according to a certain attribute
  - select any  $k_{th}$  element
  - $k = \text{number of data points} / \text{amount of samples}$

	1	2	3	4	5	6	7	8	9
age	19	22	24	33	34	42	55	61	81
gender	M	F	M	F	M	M	F	M	F
degree	MSc	PhD	High School	BSc	BSc	High School	PhD	High School	High School
nationality	japanese	german	japanese	japanese	german	german	german	german	german
martial status	married	single	married	divorced	married	single	single	divorced	single

# Data Sampling

- probabilistic sample - **stratified random sampling**
  - define **stratas** based on attributes of the data
  - then **select samples** in each strata
  - **homogeneity** within subgroups
  - **heterogeneity** between subgroups

	1	2	3	4	5	6	7	8	9
age	19	33	34	24	61	42	81	55	22
gender	M	F	M	M	M	M	F	F	F
degree	MSc	BSc	BSc	High School	High School	High School	High School	PhD	PhD
nationality	japanese	japanese	german	japanese	german	german	german	german	german
martial status	married	divorced	married	married	divorced	single	single	single	single

# Data Sampling

## ▪ probabilistic sample - cluster random sampling

- create clusters within your data
- randomly choose clusters to take the samples from
- homogeneity between subgroups
- heterogeneity within subgroups

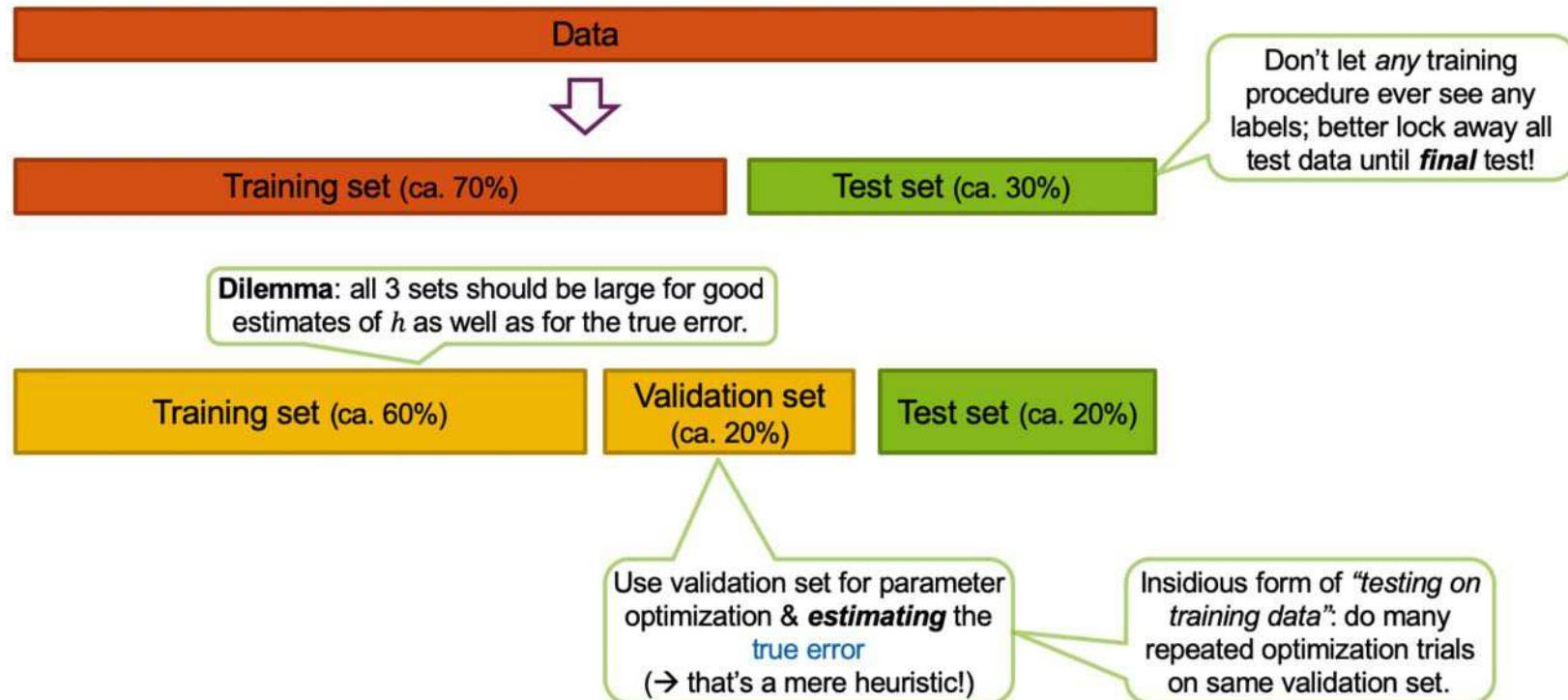
Each Cluster should be as diverse as possible → should represent the whole dataset

Because each cluster should contain everything

	1	2	3	4	5	6	7	8	9
age	34	22	42	33	55	81	61	24	19
gender	M	F	M	F	F	F	M	M	M
degree	BSc	PhD	High School	BSc	PhD	High School	High School	High School	MSc
nationality	german	german	german	japanese	german	german	german	japanese	japanese
martial status	married	single	single	divorced	single	single	divorced	married	married

# Data Partitioning

- default approaches split into training, test, and validation set (if possible)



# Data Partitioning

- k-fold cross validation

1. Divide the data set into  $k$  folds of equal size, here  $k$  is 10



2. Use one fold for testing a model built on all other data folds.



3. Repeat the model building and testing for each of the data folds.



4. Calculate the average of all of the  $k$  test errors and deliver this as result.

$k=5$  is a common choice

# Data Partitioning

- k-fold cross validation

	1	2	3	4	5	6	7	8	9
age	19	33	34	24	61	42	81	55	22
gender	M	F	M	M	M	M	F	F	F
degree	MSc	BSc	BSc	High School	High School	High School	High School	PhD	PhD
nationality	japanese	japanese	german	japanese	german	german	german	german	german
martial status	married	divorced	married	married	divorced	single	single	single	single

	10	11	12	13	14	15	16	17	18
age	29	53	34	34	41	82	41	45	62
gender	M	M	F	M	F	M	F	F	F
degree	MSc	BSc	MSc	High School	High School	High School	High School	BSc	PhD
nationality	german	japanese	german	japanese	german	german	german	german	german
martial status	married	divorced	married	married	divorced	single	single	single	single

# Data Partitioning

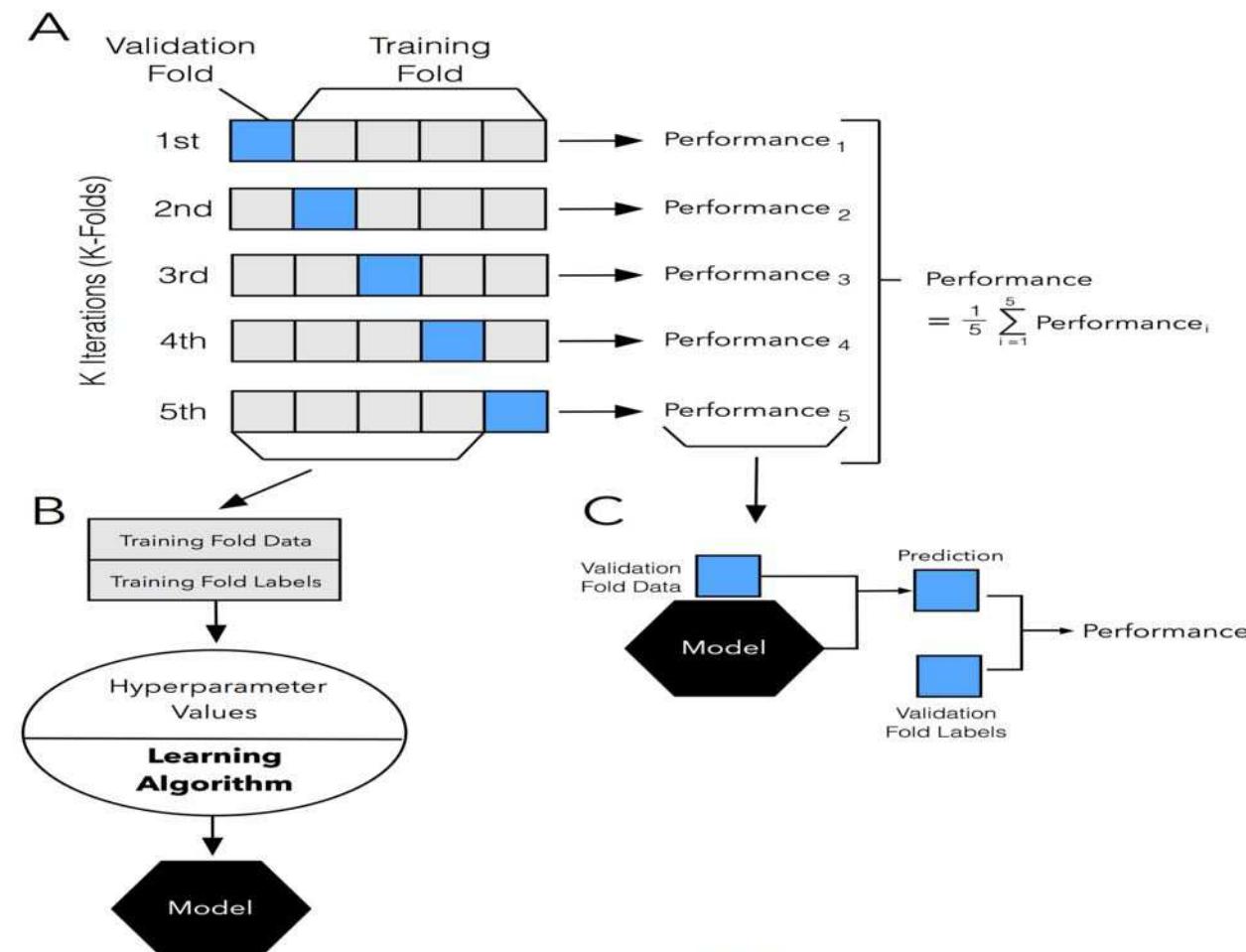
- k-fold cross validation

	1	2	3	4	5	6	7	8	9
age	19	33	34	22	34	61	81	55	22
gender	M	F	M	M	M	M	F	F	F
degree	MSc	BSc	BSc	High School	High School	High School	High School	PhD	PhD
nationality	japanese	japanese	german	japanese	german	german	german	german	german
martial status	married	divorced	married	married	divorced	single	single	single	single

	10	11	12	13	14	15	16	17	18
age	29	53	34	34	41	82	41	45	62
gender	M	M	F	M	F	M	F	F	F
degree	MSc	BSc	MSc	High School	High School	High School	High School	BSc	PhD
nationality	german	japanese	german	japanese	german	german	german	german	german
martial status	married	divorced	married	married	divorced	single	single	single	single

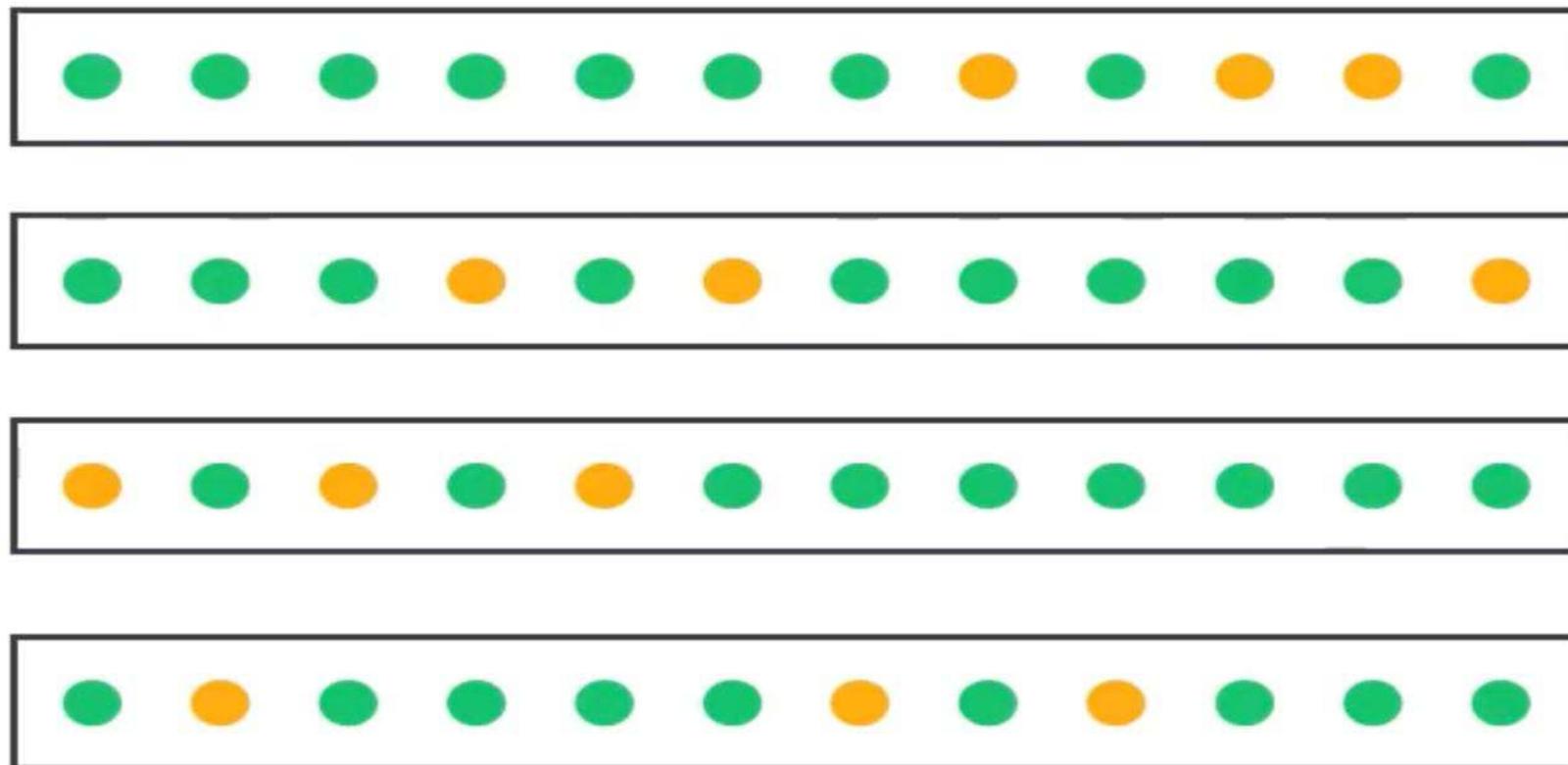
# Data Partitioning

- k-fold cross validation



# Data Partitioning

- k-fold cross validation (randomizing)



# Data Partitioning

- leave-one-out cross validation (LOOCV)

- Train the model on  $n-1$  samples and evaluate it on the single data point
- Computing expensive
- Useful especially for small datasets



# Lessons Learned

- understanding of data preprocessing and cleaning
- understanding of data normalization, sampling, and partitioning



# Questions and Answers



# Machine Learning and Data Mining

## V04: Evaluation

Lecturer: **Dr. Andreas Weiler**



# Repetition

... REPETITION ...

→ IS THE MOTHER →

of learning

# Course Content IT20a\_WIN

Week	Lecture	Lab
KW08	Introduction	Lab 01
KW09	Data Processing Part 01	Lab 02
KW10	Carnival Monday	
KW11	Data Processing Part 02	Lab 03
KW12	Evaluation	Lab 04
KW13	Recommender Systems	<b>Graded Lab 01</b> <span style="border: 1px solid red; padding: 2px;">IMPORTANT</span>
KW14	Association Rules	<b>Graded Lab 01</b> <span style="border: 1px solid red; padding: 2px;">IMPORTANT</span>
KW15	Linear and Logistic Regression	Lab 05
KW16	Easter Monday	
KW17	Support Vector Machines	<b>Graded Lab 02</b> <span style="border: 1px solid red; padding: 2px;">IMPORTANT</span>
KW18	Decision Trees and Naïve Bayes	<b>Graded Lab 02</b> <span style="border: 1px solid red; padding: 2px;">IMPORTANT</span>
KW19	Partition Clustering	Lab 06
KW20	Hierarchical and Density Clustering	Lab 07
KW21	Neural Networks and Closing	Review and Questions for Exam

# Learning Objectives

- understanding of training, testing, and validation data
- understanding of evaluation scenarios



# Evaluation

- necessary for evaluating the systems performance
  - systems performance with different parameters
  - comparison of different algorithms and settings
- a serious evaluation needs a clear task and design
  - motivation, goals, and method
  - qualitative vs. quantitative results
  - given repeatability and reproducibility
- evaluation of separate components if necessary

# Evaluation Measures

$\hat{y}$  = predicted  $y$

- standard error measure

- error  $E = \frac{1}{N} \sum_{i=1}^N (1 - id(\hat{y}_i, y_i))$ , where  $id(a, b) = \begin{cases} 1 & \text{if } a = b \\ 0 & \text{else} \end{cases}$  is the identity function

- components of  $E$

- contingency table:

		actual prediction	
		$\downarrow y, \hat{y} \rightarrow$	
		1	0
1		true positive (TP)	false negative (FN)
0		false positive (FP)	true negative (TN)

- True Positives (TP): when the actual class of the data point was 1(True) and the predicted is also 1(True)
- True Negatives (TN): when the actual class of the data point was 0(False) and the predicted is also 0(False)
- False Positives (FP): when the actual class of the data point was 0(False) and the predicted is 1(True).
- False Negatives (FN): When the actual class of the data point was 1(True) and the predicted is 0(False).

# Evaluation Measures

- components of  $E$ 
  - contingency table:

$\downarrow y, \hat{y} \rightarrow$	1	0
1	true positive (TP)	false negative (FN)
0	false positive (FP)	true negative (TN)

= hit

= miss,  
type-II error

= false alarm,  
type-I error

- classification with costs
  - attach costs to each cell in the matrix above («cost matrix»)
  - replace sum of errors with sum of costs per actual prediction



# Evaluation Measures

- example for standard error measure

- error  $E = \frac{1}{N} \sum_{i=1}^N (1 - id(\hat{y}_i, y_i))$ , where  $id(a, b) = \begin{cases} 1 & \text{if } a = b \\ 0 & \text{else} \end{cases}$  is the identity function
- $E = \frac{1}{9} * 3 = 0.3333$



color	size	surface	label	prediction
2,3,4,5	133	576	fruit	fruit
2,3,4,5	145	576	fruit	fruit
6,7,8	999	333	veggie	veggie
3,7,8	888	333	veggie	veggie
1,4,3	888	334	veggie	veggie
2,3,4	100	588	fruit	fruit
6,7	999	334	veggie	fruit
3,4	100	555	fruit	veggie
4,5	99	555	fruit	veggie

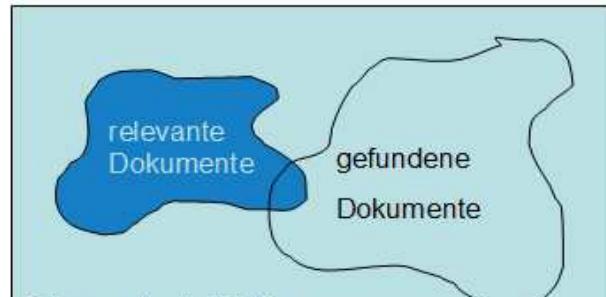
# Evaluation Measures

- **accuracy**
  - standard measure that doesn't regard different «costs» of errors
  - formula:  $\frac{TP+TN}{TP+TN+FP+FN}$
- **recall**
  - how many of the relevant documents (i.e.,  $y = 1$ ) have been returned (i.e.,  $\hat{y} = 1$ )?
  - formula:  $\frac{TP}{TP+FN}$
- **precision**
  - how many of the returned documents are actually relevant?
  - formula:  $\frac{TP}{TP+FP}$
- **F-measure**
  - combination of **recall** & **precision** via their harmonic mean
  - formula:  $\frac{2 * (recall * precision)}{recall + precision}$

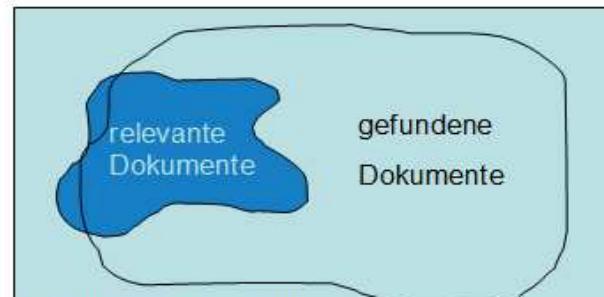
# Evaluation Measures

- precision and recall
  - there's a trade-off between recall and precision because they show the two different types of error

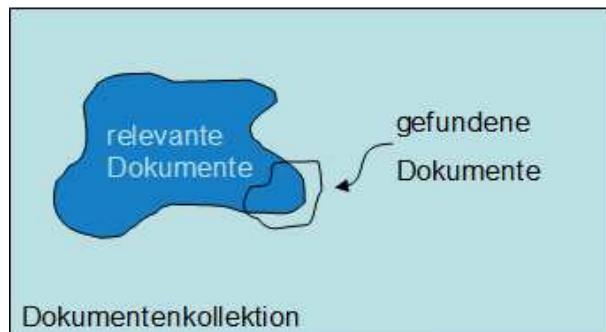
recall = Ausbeute



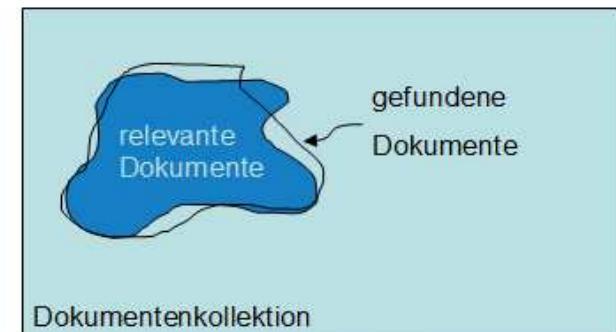
Geringe Ausbeute, geringe Präzision



Hohe Ausbeute, geringe Präzision



Geringe Ausbeute, hohe Präzision



Hohe Ausbeute, hohe Präzision

Quelle: <http://www.svisseduc.ch/informatik/sozialkult/doocs/2005-03-issep-suchmaschinen.pdf>

# Evaluation Measures

- **precision**
  - part of the returned objects that are relevant

$$\text{precision} = \frac{\text{count(relevant returned documents)}}{\text{count(returned documents)}}$$

- example
  - query: fruits with yellow fruit pulp
  - relevant: lemon, banana, pineapple, mango
  - result: banana, plum, mango
  - precision = 2/3



# Evaluation Measures

- **recall**
  - part of the relevant objects that are returned

$$\text{recall} = \frac{\text{count(relevant returned documents)}}{\text{count(relevant documents)}}$$

- example
  - query: fruits with yellow fruit pulp
  - relevant: lemon, banana, pineapple, mango
  - result: banana, plum, mango
  - recall = 2/4



# Evaluation Measures

- precision and recall

	<b>relevant</b>	<b>not relevant</b>
<b>returned</b>	true positives (TP)	false positives (FP)
<b>not returned</b>	false negatives (FN)	true negatives (TN)

$$precision = \frac{TP}{(TP + FP)}$$

$$recall = \frac{TP}{(TP + FN)}$$

	<b>relevant</b>	<b>not relevant</b>
<b>returned</b>	lemon, banana	plum
<b>not returned</b>	pineapple, mango	kiwi, apple, ...



# Evaluation Measures

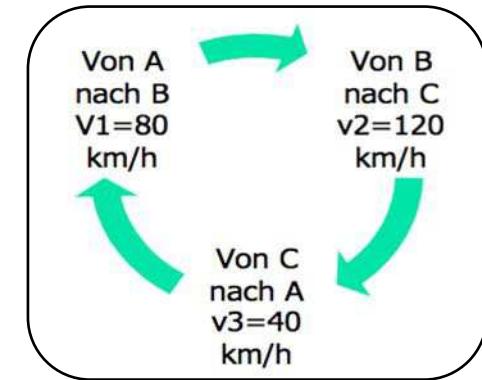
- precision vs recall
  - recall can be increased by returning more documents
    - example
      - Q: fruits with yellow fruit pulp
      - result: lemon, plum, banana, pineapple, kiwi, cherry, orange, apple
      - relevant: lemon, banana, pineapple, mango
      - precision = 3/8, recall = 3/4
    - a system which returns all documents has 100% recall
    - the opposite can also be achieved
      - example
        - Q: fruits with yellow fruit pulp
        - result: lemon
        - relevant: lemon, banana, pineapple, mango
        - precision = 1/1, recall = 1/4



# Evaluation Measures

## ■ F-measure

- calculates the balance (harmonic mean) between precision and recall
- excursion: harmonic mean
  - example: speed calculation
  - from A to B to A with 10 km distance each
  - arithmetic mean:  $(40+80+120)/3 = 80$
  - harmonic mean:  $\frac{3}{\frac{1}{80} + \frac{1}{40} + \frac{1}{120}} = 60$



## ■ formula

$$F_{measure} = \frac{2 * \frac{precision * recall}{precision + recall}}{precision + recall}$$

# Evaluation Measures

	relevant	not relevant
returned	20	40
not returned	60	1,000,000

- precision =  $20 / (20 + 40) = 1/3$
- recall =  $20 / (20 + 60) = 1/4$
- $F_{measure} = 2 * (1/3 * 1/4) / (1/3 + 1/4) = 0.286$

	relevant	not relevant
returned	50	10
not returned	30	1,000,000

- precision =  $50 / (50 + 10) = 5/6$
- recall =  $50 / (50 + 30) = 5/8$
- $F_{measure} = 2 * (5/6 * 5/8) / (5/6 + 5/8) = 0.714$

# Evaluation Measures

	relevant	not relevant
returned	10	2
not returned	20	1,000,000

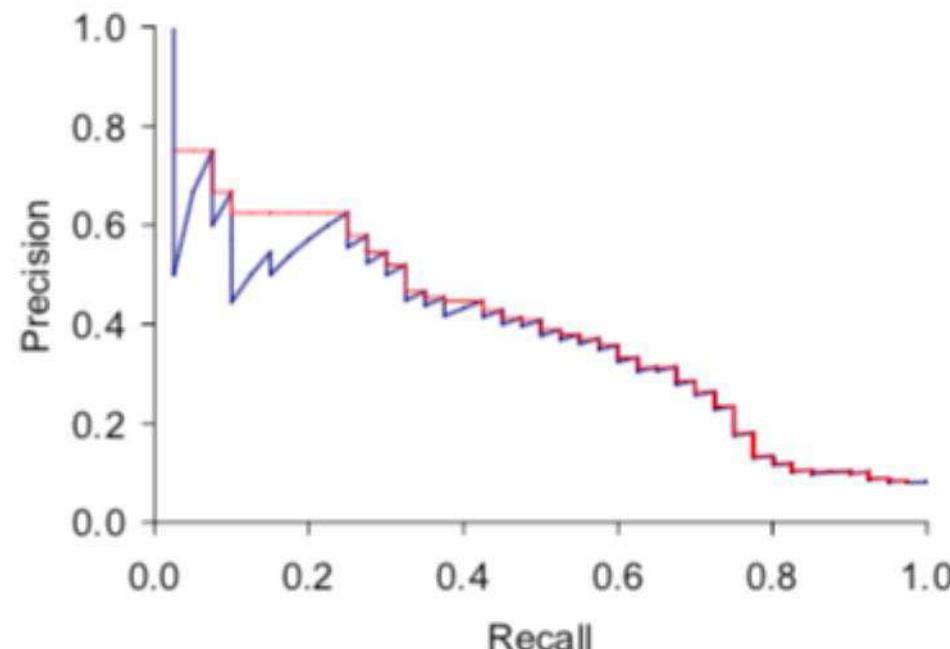
- precision =  $10 / (10 + 2) = 0.833$
- recall =  $10 / (10 + 20) = 0.333$
- $F_{\text{measure}} = 2 * (0.833 * 0.333) / (0.833 + 0.333) = 0.476$

	relevant	not relevant
returned	28	60
not returned	2	1,000,000

- precision =  $28 / (28 + 60) = 0.318$
- recall =  $28 / (28 + 2) = 0.933$
- $F_{\text{measure}} = 2 * (0.318 * 0.933) / (0.318 + 0.933) = 0.475$

# Evaluation Measures

- precision recall curve
  - precision, recall and  $F_{measure}$  are scores for results without a ranking
  - calculate results for several TopN (e.g., TopN n in 1 to 10).
  - shows the course of the results for different settings
  - each data point shows one precision recall combination



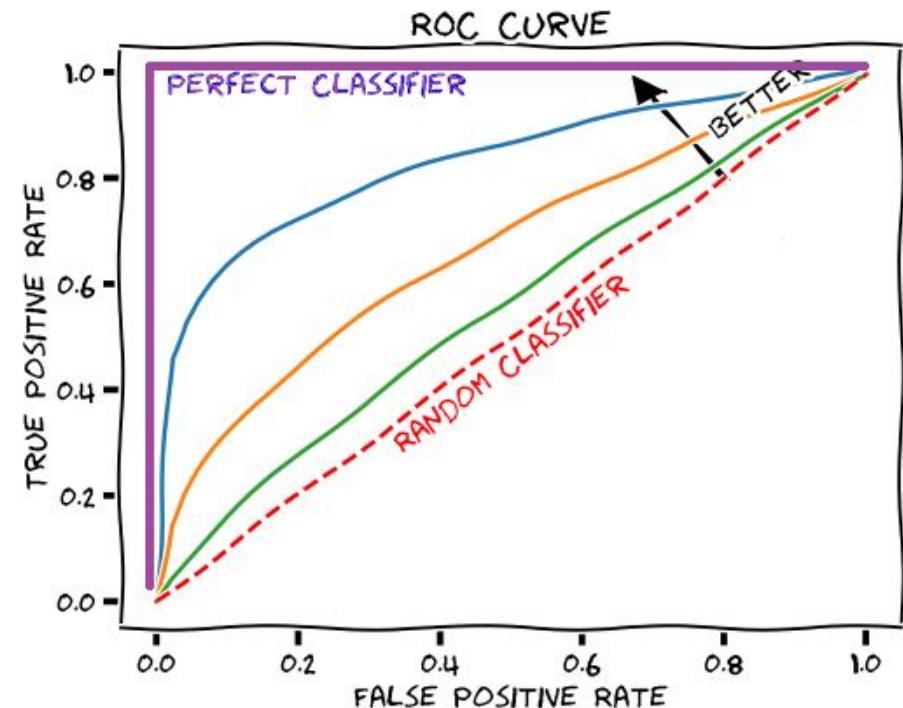
# Evaluation Measures

- area under roc curve
  - roc = receiver operating characteristic curve
  - true positive eqv. with hit
  - false positive eqv. with false alarm, type I error or underestimation
  - the farther away from a straight line they are, the better

$\downarrow y, \hat{y} \rightarrow$	1	0
1	true positive (TP)	false negative (FN)
0	false positive (FP)	true negative (TN)

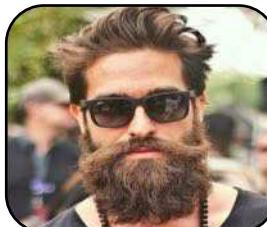
$$TPR = \frac{TP}{P} = \frac{TP}{TP + FN} = 1 - FNR$$

$$FPR = \frac{FP}{N} = \frac{FP}{FP + TN} = 1 - TNR$$



# Evaluation Measures

- relevance assessment and labelling
  - relevance assessments are only useful when consistent
  - inconsistent assessments lead to »untruth« results
  - kappa value for inter-rater agreement
  - goal: consistent and true gold standard or ground-truth



# Evaluation Measures

- the kappa value measures the score of inter-rater agreement or disagreement
- developed for categorical ratings
- formula:
  - $P(A)$  = proportion of observed agreements of the raters
  - $P(E)$  = agreement, which we could get randomly

$$K = \frac{P(A) - P(E)}{1 - P(E)}$$

- values for  $k$  in the interval  $[0.2, 1.0]$  are acceptable
- for lower values of  $k$  repeat the ratings with different raters

# Evaluation Measures

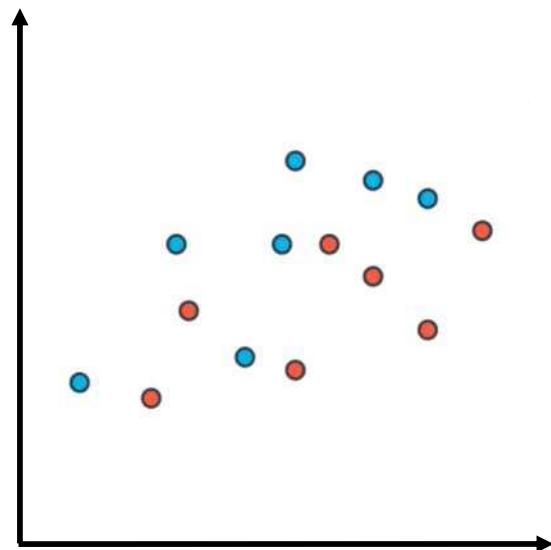
- example for kappa calculation

		rater2 relevant		Total
		Yes	No	
rater1 relevant	Yes	300	20	320
	No	10	70	80
		Total	90	400

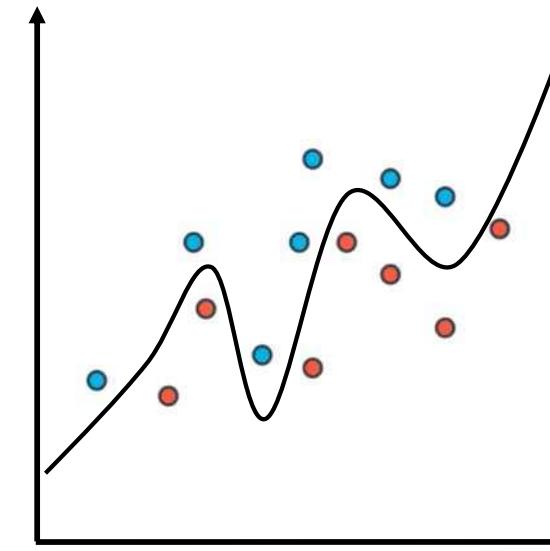
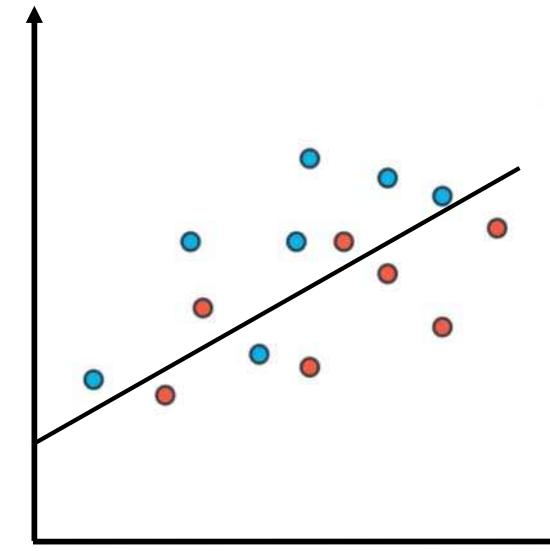
$$K = \frac{P(A) - P(E)}{1 - P(E)}$$

- $P(A) = (300 + 70) / 400 = 0.925$
- $P(\text{no}) = (80 + 90) / (400 + 400) = 170/800 = 0.2125$
- $P(\text{yes}) = (320 + 310) / (400 + 400) = 630/800 = 0.7878$
- $P(E) = P(\text{no})^2 + P(\text{yes})^2 = 0.045 + 0.62 = 0.665$
- Kappa  $K = (0.925 - 0.665) / (1 - 0.665) = 0.776$  (acceptable)

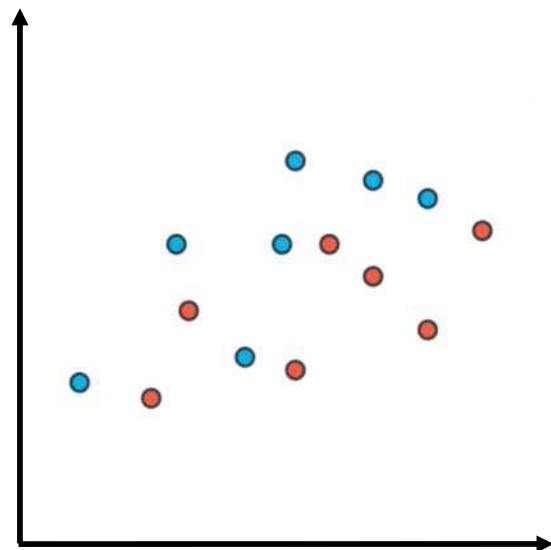
# Model Evaluation



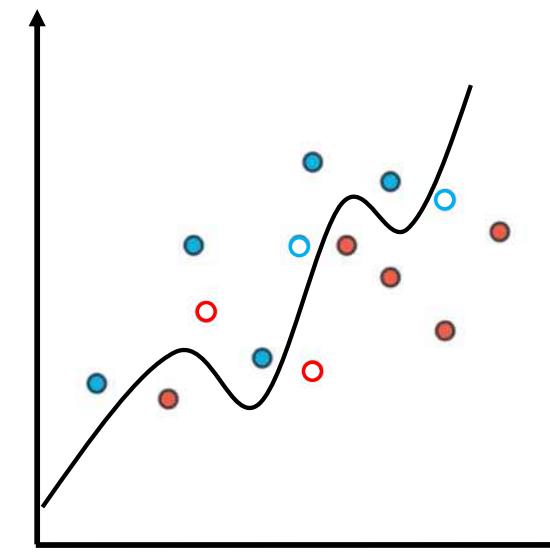
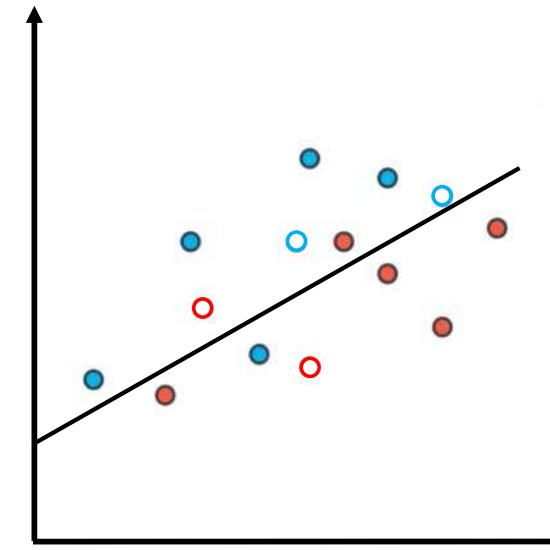
?



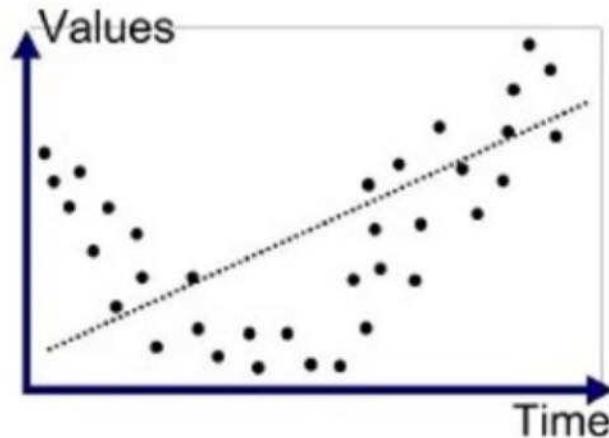
# Model Evaluation



?

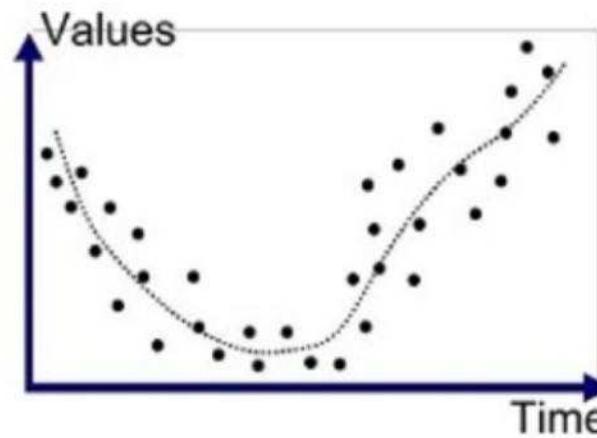


# Model Evaluation



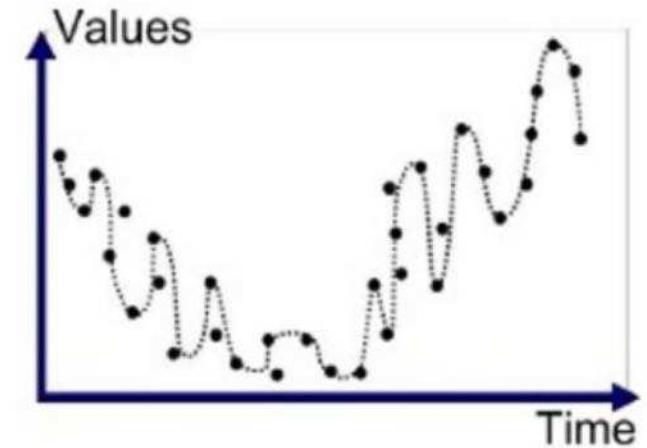
Underfitted

Bad on Training Set  
Bad on Testing Set



Good Fit/Robust

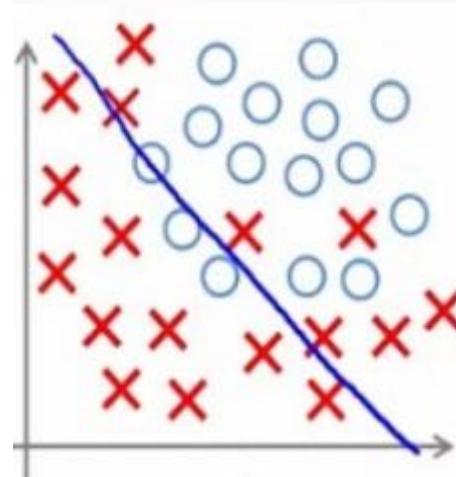
Good on Training Set  
Good on Testing Set



Overfitted

Great on Training Set  
Bad on Testing Set

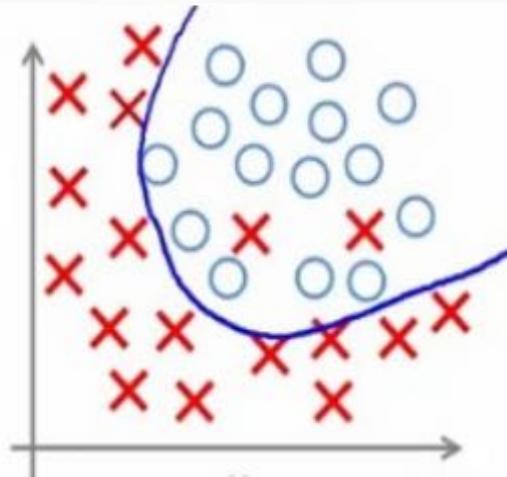
# Model Evaluation



**Under-fitting**

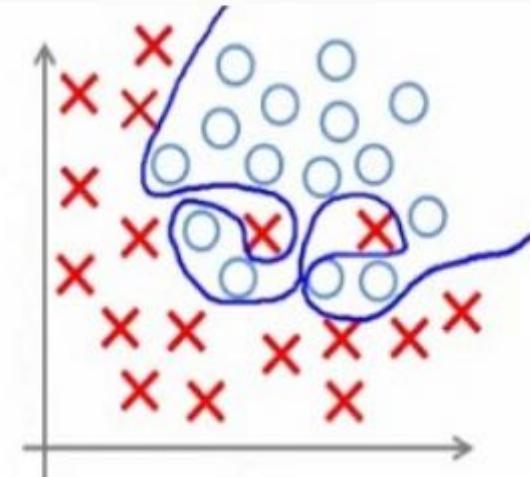
(too simple to  
explain the  
variance)

Bad on Training Set  
Bad on Testing Set



**Appropriate-fitting**

Good on Training Set  
Good on Testing Set



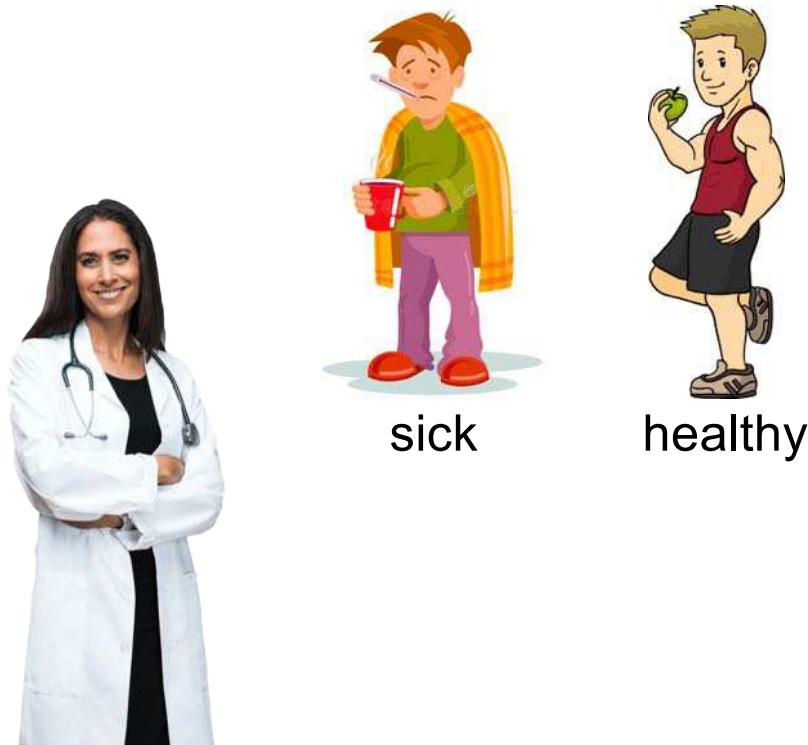
**Over-fitting**

(forcefitting – too  
good to be true)

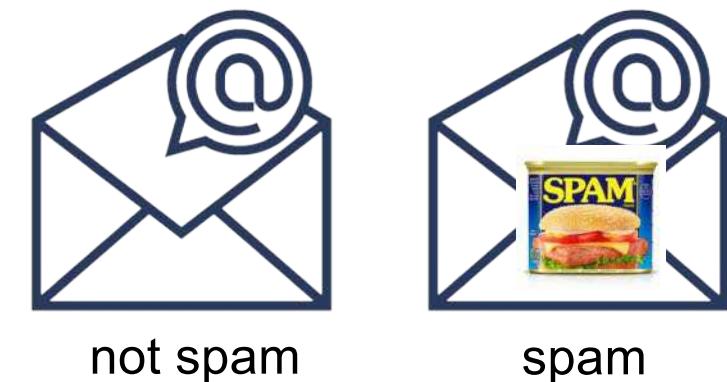
Great on Training Set  
Bad on Testing Set

# Model Evaluation: Two cases...

## Medical Model



## Spam Classifier Model



# Model Evaluation: Two cases...

## Medical Model

	diagnosed sick	diagnosed healthy
sick	 	 
healthy	 	 

## Spam Classifier Model

	sent to spam folder	sent to inbox
spam	 	 
not spam	 	 

# Model Evaluation: Two cases...

## Medical Model

	diagnosed sick	diagnosed healthy
sick	1000 true positives 200 false negatives	
healthy	800 false positives 8000 true negatives	

## Spam Classifier Model

	sent to spam folder	sent to inbox
spam	100 true positives 170 false negatives	
not spam	30 false positives 700 true negatives	

# Model Evaluation: Two cases...

## Medical Model

accuracy: 90%

	diagnosed sick	diagnosed healthy
sick	1000 true positives 200 false negatives	
healthy	800 false positives 8000 true negatives	

## Spam Classifier Model

accuracy: 80%

	sent to spam folder	sent to inbox
spam	100 true positives 170 false negatives	
not spam	30 false positives 700 true negatives	

# Model Evaluation: Two cases...

## Medical Model

accuracy: 90%

- false positives ok
- **false negatives not ok**
- optimized for **high recall**
- find **all** the sick people



## Spam Classifier Model

accuracy: 80%

- **false positives not ok**
- false negatives ok
- optimized for **high precision**



# Model Evaluation: Two cases...

## Medical Model

precision: 55,7%

	diagnosed sick	diagnosed healthy
sick	1000 true positives 200 false negatives	
healthy	800 false positives 8000 true negatives	

## Spam Classifier Model

precision: 76,9%

	sent to spam folder	sent to inbox
spam	100 true positives 170 false negatives	
not spam	30 false positives 700 true negatives	

# Model Evaluation: Two cases...

## Medical Model

recall: 83,3%

	diagnosed sick	diagnosed healthy
sick	1000 true positives	200 false negatives
healthy	800 false positives	800 true negatives

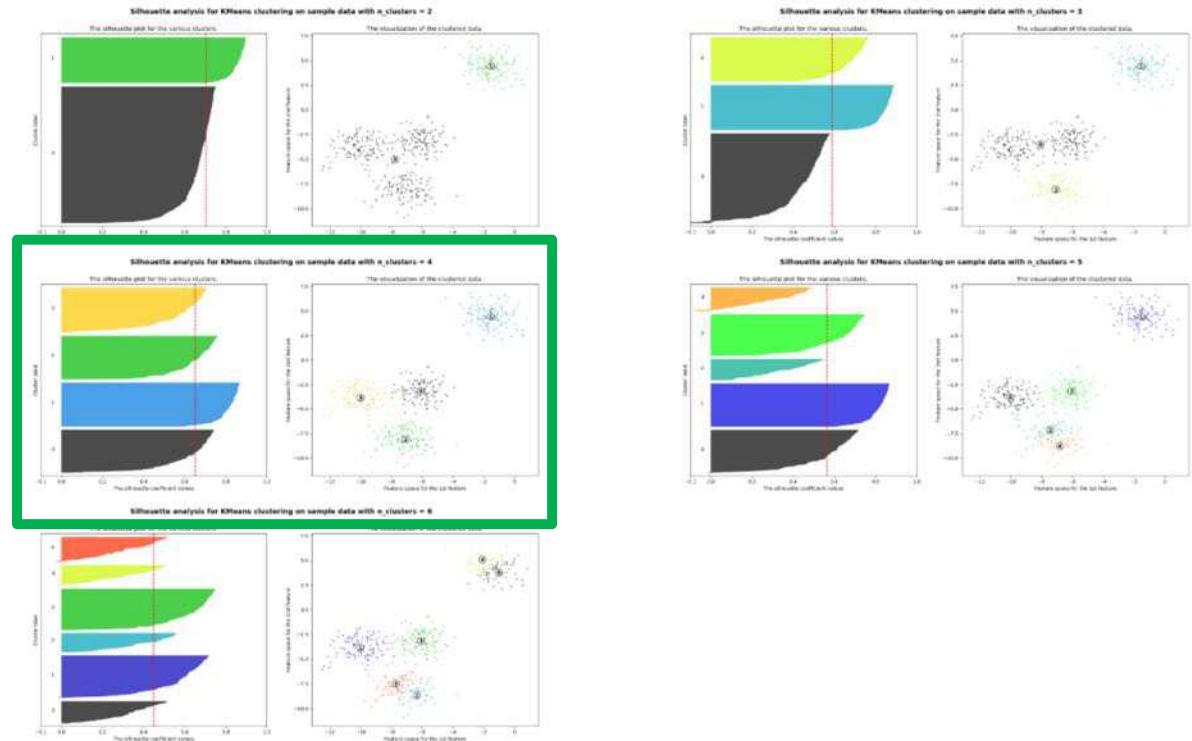
## Spam Classifier Model

recall: 37%

	sent to spam folder	sent to inbox
spam	100 true positives	170 false negatives
not spam	30 false positives	700 true negatives

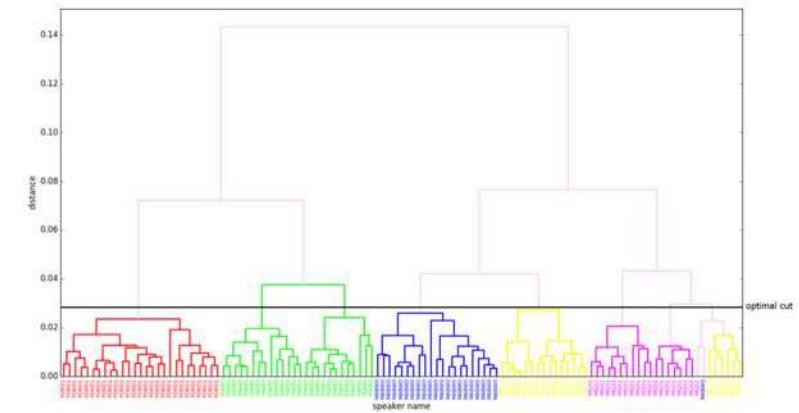
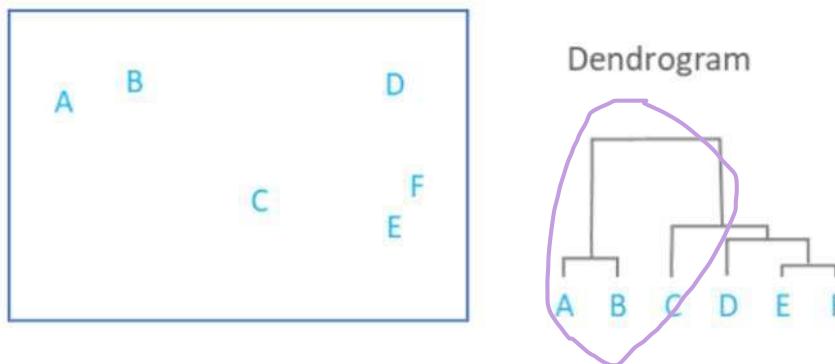
# Model Evaluation: Clustering

- Silhouette coefficients (as these values are referred to as) near +1 indicate that the sample is far away from the neighboring clusters
- A value of 0 indicates that the sample is on or very close to the decision boundary between two neighboring clusters and negative values indicate that those samples might have been assigned to the wrong cluster
- n = 2 and n = 4 are > then the average silhouette\_score
- n = 4 the clusters have more similar sickness then for n = 2



# Model Evaluation: Clustering

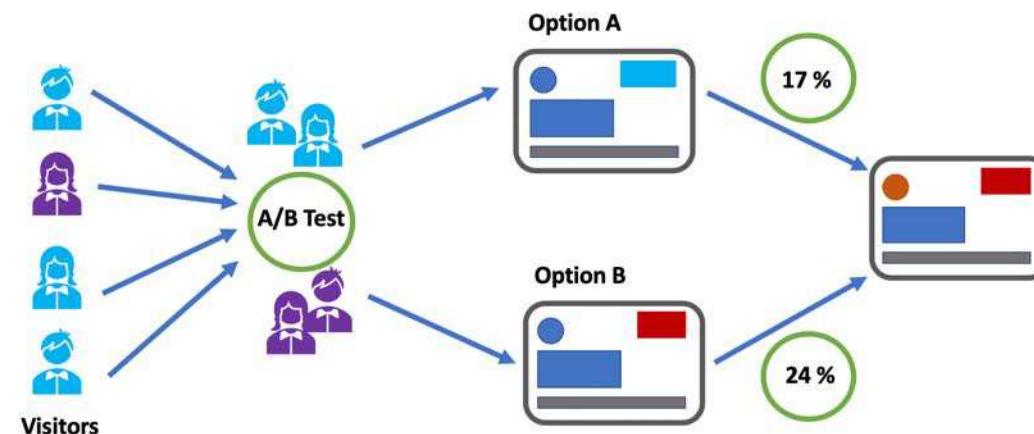
- shows the distribution of data points in a tree like fashion
- the heights reflect the distance between the clusters
- the dendrogram below shows us that the big difference between clusters is between the cluster of A and B versus that of C, D, E, and F



Dendrogram of utterances of 5 speakers, colored by speaker id;  
from [Lukic et al., MLSP 2016].

# A/B Testing

- testing of components or features on target groups
- example:
  - a older version of a recommender system provides recommendations for the main group of users
  - a selected group of users get recommendations from a new version
  - comparative evaluation of the performance of both versions
- direct evaluation after a certain time
- dynamic and flexible development
- reflects generally the reality most suitable



# Evaluation in the wild: Twitter Event Detection

- definition of an event
  - „a real-world occurrence that takes place in a certain geographical location and over a certain time period“ (Allan, 2002)
  - „a significant thing that happens at some specific time and place“ (McMinn et al., 2013)
  - „a behavior associated to a topic within a time interval in which it has been extensively treated but rarely before and after“ (Guille et al., 2013)



Hudson Plane Crash (2009)



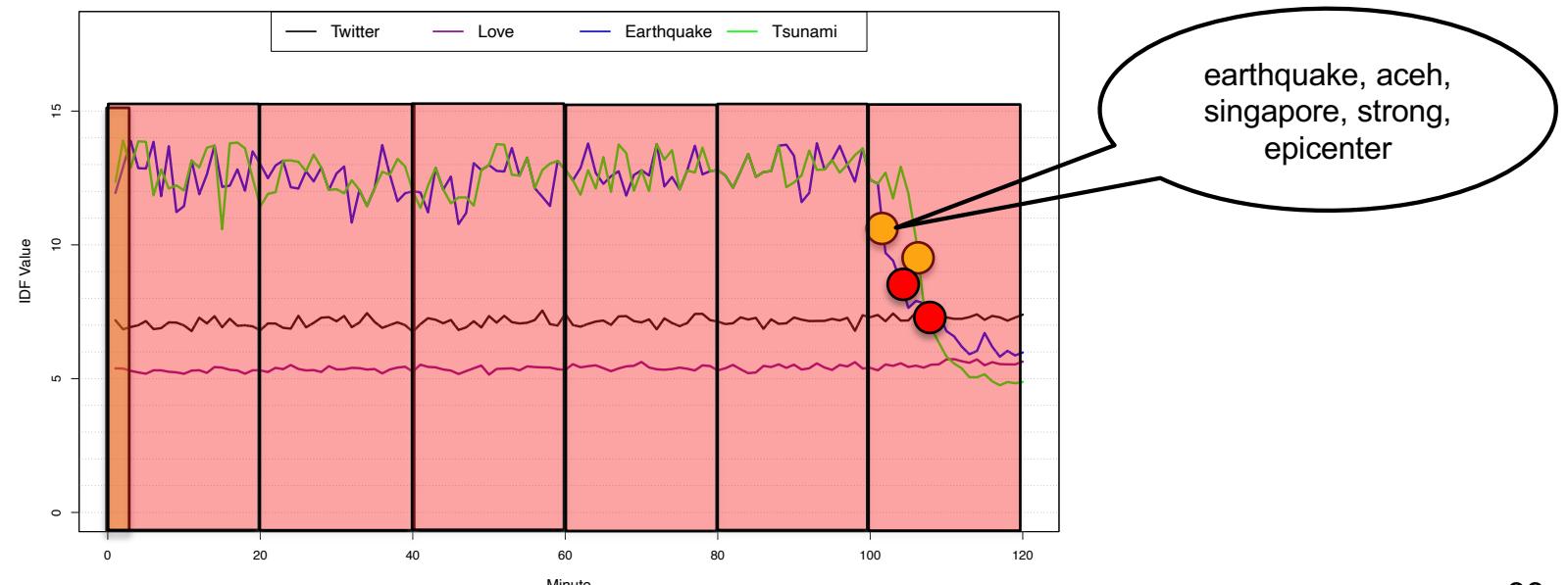
Boston Marathon Bombing (2013)



Winning Goal World Cup (2014)

# Evaluation in the wild: Twitter Event Detection

- analysis of frequencies of terms for detecting the „unexpected“
- four steps of event detection
  - pre-processing of the tweets
  - detection algorithm
  - creation of the event description
  - alarming or reporting of events



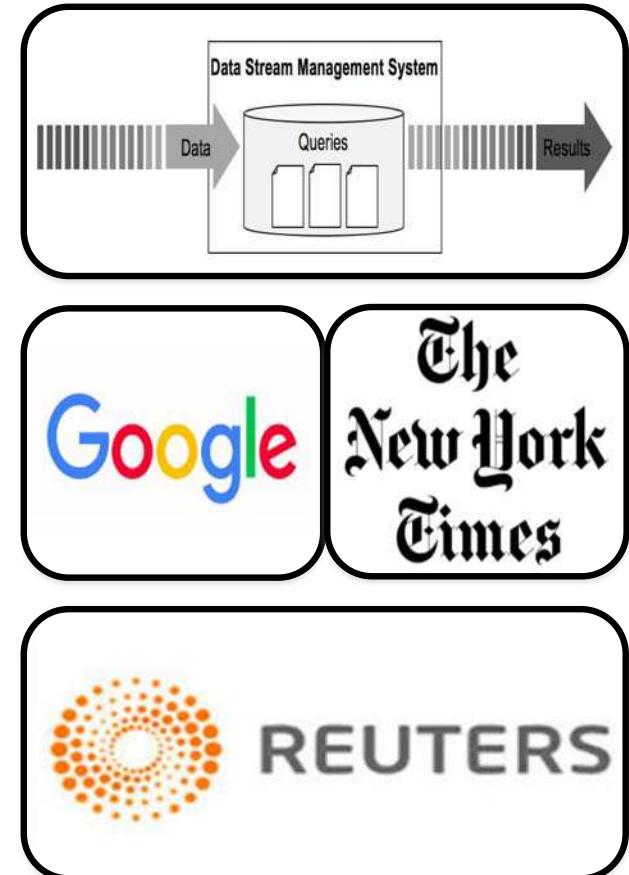
# Evaluation in the wild: Twitter Event Detection

- large diversity of event detection methods
  - very varying preprocessing steps
  - very varying detection methods
  - very varying event descriptions
  - many parameters and settings
- varying definitions of the events
  - social, sport, or political events
  - natural disasters or criminal events
  - country-specific events
  - events in events
- no data set for testing available
  - not allowed to publish twitter data
- no data set for evaluation available
  - relevance assessment very difficult
  - no ground-truth or gold standard



# Evaluation in the wild: Twitter Event Detection

- goal: a common evaluation framework
  - implementation of the methods in a DSMS
  - common preprocessing steps
  - common event descriptions as output
- a common data set for testing
  - input: selected days with events
  - output: similar number of events
- a common data set for evaluation
  - precision: events verified with Google or NYTimes
  - recall: events liste in Reuters News Archive



# Evaluation in the wild: Twitter Event Detection

The image displays two search results pages for the query "marines, usa, weapons, yemen, today".

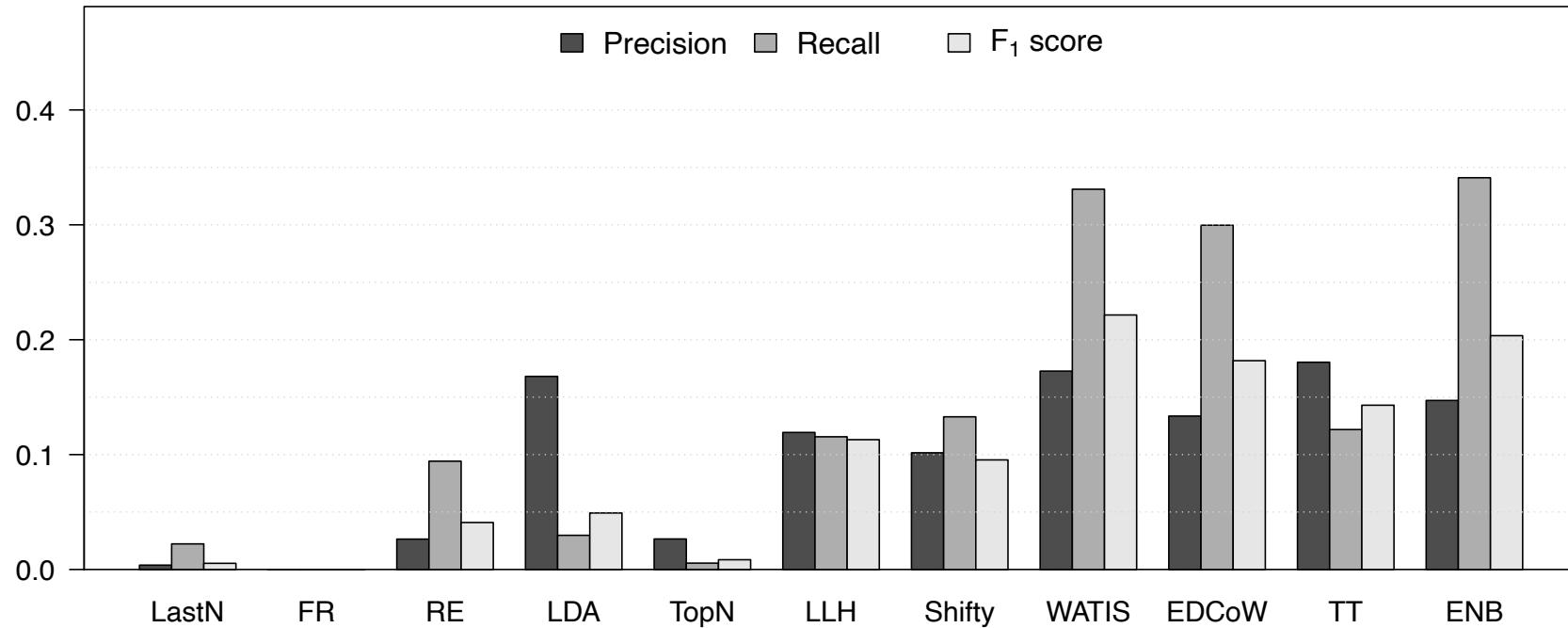
**Left Side (Google Search Results):**

- World News Headlines**
  - Leaders to sign document after Minsk talks on Ukraine: source
  - Thousands protest against Houthi rule in Yemen after embassies close
  - U.S. Marines say they destroyed weapons before leaving Yemen**
  - Minsk talks on Ukraine crisis could last six more hours: Kiev presidential aide
  - Sri Lanka seeks delay in U.N. war-crimes report
  - Australia seeks last-ditch deal to save pair from execution in Indonesia
  - Leaders hold Ukraine peace talks as fighting surges
  - Euro zone, Greece fail to agree way forward following meeting
  - Greece's Varoufakis seeks 'healing deal' on Monday
  - Canada 'inclined' to extend mandate of forces in Iraq: minister
  - Suicide bombers strike Niger town; Chadians kill 13 Boko Haram militants
- Adam Scott Out**
  - Adam Scott Out
  - 2013 Masters -
  - Bilder zu master
  - Weitere Bilder zu
  - Masters: Adam
  - The Masters 20
  - Adam Scott wins
  - Adam Scott beats

**Right Side (Search Engine Interface):**

- Search bar: marines, usa, weapons, yemen, today
- Results:
  - You Can't Sleep | World of Psychology
  - Sleepy Life - HTC - Android phones - Whirlpool
  - Social Media When You Can't Sleep ...
  - Baby Now Dangerous (and Illegal)? | The ...
  - g-tube suggestions? - BabyCenter
  - TeamFortress.TV

# Evaluation in the wild: Twitter Event Detection



- generally very low values
- but very good to compare the different methods

# Lessons Learned

- understanding of training, testing, and validation data
- understanding of evaluation scenarios



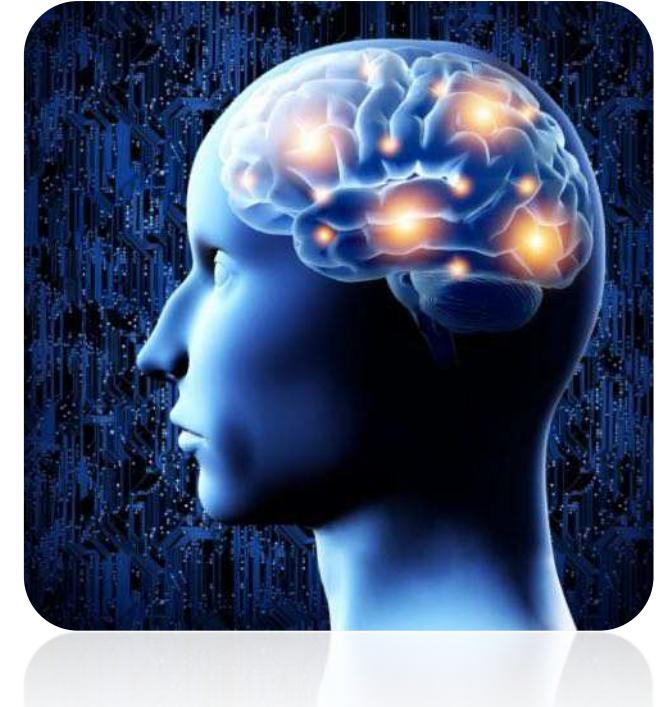
# Questions and Answers



# Machine Learning and Data Mining

## V05: Recommender Systems

Lecturer: Dr. Andreas Weiler



# Repetition

... REPETITION ...

→ IS THE MOTHER →  
*of learning*

# Course Content IT20a\_WIN

Week	Lecture	Lab
KW08	Introduction	Lab 01
KW09	Data Processing Part 01	Lab 02
KW10	Carnival Monday	
KW11	Data Processing Part 02	Lab 03
KW12	Evaluation	Lab 04
KW13	Recommender Systems	<b>Graded Lab 01</b> <span style="border: 2px solid red; padding: 2px;">IMPORTANT</span>
KW14	Association Rules	<b>Graded Lab 01</b> <span style="border: 2px solid red; padding: 2px;">IMPORTANT</span>
KW15	Linear and Logistic Regression	Lab 05
KW16	Easter Monday	
KW17	Support Vector Machines	<b>Graded Lab 02</b> <span style="border: 2px solid red; padding: 2px;">IMPORTANT</span>
KW18	Decision Trees and Naïve Bayes	<b>Graded Lab 02</b> <span style="border: 2px solid red; padding: 2px;">IMPORTANT</span>
KW19	Partition Clustering	Lab 06
KW20	Hierarchical and Density Clustering	Lab 07
KW21	Neural Networks and Closing	Review and Questions for Exam

# Learning Objectives

- fundamental understanding of the different types of recommender systems
- understanding of problems and solutions within recommender systems
- examples of recommender systems in productive systems

“Already, 35 percent of what consumers purchase on Amazon and 75 percent of what they watch on Netflix come from product recommendations based on algorithms.”  
- McKinsey, 2013

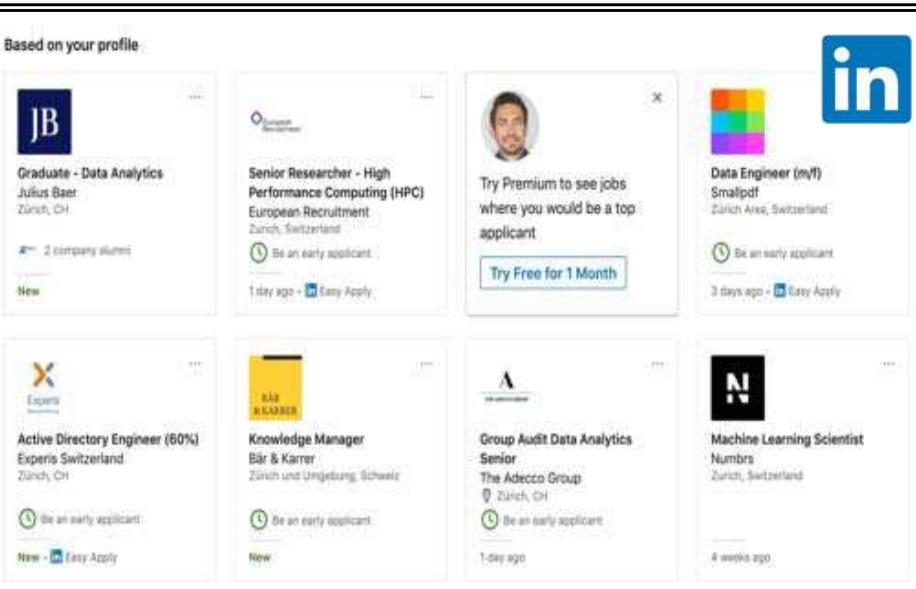


# From Scarcity to Abundance



- shelf space is a scarce commodity for traditional retailers
- web enables near-zero-cost dissemination of information about products
- more choice necessitates better filters, like recommendation systems

# Recommender Systems



# Amazons Recommendations

Recommended for you, Thomas

Literature & Fiction	Cancer & Fitness Equipment	Health, Fitness & Dining Books	Tableware
CATASTROPHE	Coffee, Tea & Espresso	Diagnoses & Memoirs	Engineering Books

Frequently Bought Together

Total price: \$94.90

Add both to Cart  
Add both to List

This item: Rumble Roller - Textured Muscle Foam Roller Manipulates Soft Tissue Like A Massage Therapist \$69.95

Rumble Roller X-Firm Beastie and Base - Extra Firm Spiky Massage Ball - Comes With Base For... \$24.95 (\$3.12 / oz)

Your Recently Viewed Items and Featured Recommendations

Inspired by your browsing history

Navya Gymnastic Rings for Full Body Strength and Circuit Training	Champion Sports NCAI-NHGS Certified Lacrosse Ball	Rumble Roller Beastie Hook - Hand Held Massage Tool - Use With RumbleRoller Beastie	Muscle Roller Ball Set :: FOR TRIGGER POINT & Myofascial Release	Foam Roller, Luxe Premium High Density Firm Roller - Extra Firm With 1 Year Warranty	2X LACROSSE BALLS FOR TRIGGER POINT MASSAGE - Fine-Toned plus	The Dead Wedge - Deadlift Jack Alternative for Your Gym Bag - Raises loaded barbell & plates...
1,274 \$25.87 Prime	573 \$15.00 - \$11.19	1,131 \$14.97 Prime	237 \$10.75 - \$24.95	1,421 \$10.75 Prime	43 \$17.95 Prime	355

Departments · Browsing History · Thomas's Amazon.com · Today's Deals · Gift Cards & Registry · Sell · Help ·

Your Browsing History [View and Edit](#)

Rumble Roller Full Size Extra Firm Black...	Kindle Paperwhite, 6" High Resolution Display...	Rumble Roller Beastie Hook - Hand Held...	Amazon Prime (One Year Membership)	Harney & Sons Black Tea Hot Cinnamon Sunse...
Today	Thu, May 26	Wed, May 25		

Related to items you've viewed [See more](#)

calvino

WILD SARDINES IN EXTRA VIRGIN OLIVE OIL

Customers Who Bought This Item Also Bought

Rumble Roller X-Firm Beastie and Base	Rumble Roller Beastie 1+1 Firm Spiky Massage Ball - Comes With Base For...	Rumble Roller Beastie Bar + Stands - Hands Free	Rumble Roller Original Beastie + Base - Massage Roller Beastie - Comes With Base For Stabilized...	Rumble Roller - Textured Muscle Foam Roller Manipulates Soft Tissue Like A Massage Therapist	Rumble Roller Foam Rollers - Textured Muscle Foam Roller Manipulates Soft Tissue Like A...
112 \$24.95 Prime	73 \$34.95 Prime	27 \$59.95 Prime	37 \$24.95 Prime	43 \$44.95 - \$67.48	1,010 \$35.55

There is a newer version of this item:

Kindle Paperwhite E-reader, 6" High-Resolution Display (300 ppi) with Built-in Light, Wi-Fi - Includes Special Offers

\$119.99

5 stars (19,163)

In Stock.

Recommended for You Based on Kindle Paperwhite, 6" High Resolution Display w...

Moko Case for Kindle Paperwhite, Premium Thinnest and Lightest Leather Cover with...	Swees Ultra Slim Leather Case Cover for Amazon All-New Kindle Paperwhite (Both 2012...	Fintie SmartShell Case for Kindle Paperwhite - The Thinnest and Lightest Leather Cover for...	Kindle Paperwhite, 6" High Resolution Display (212 ppi) with Built-in Light, Free 3G
598 \$9.99 Prime	273 \$3.99 Prime	7,015 \$14.99 Prime	45,265 \$159.99 Prime

Best-selling emerging technology [See more](#)

# Recommender Systems in a nutshell



User X

User Y



buys



buys



buys



buys



buys



recommended

# Recommender Systems in a nutshell



User X



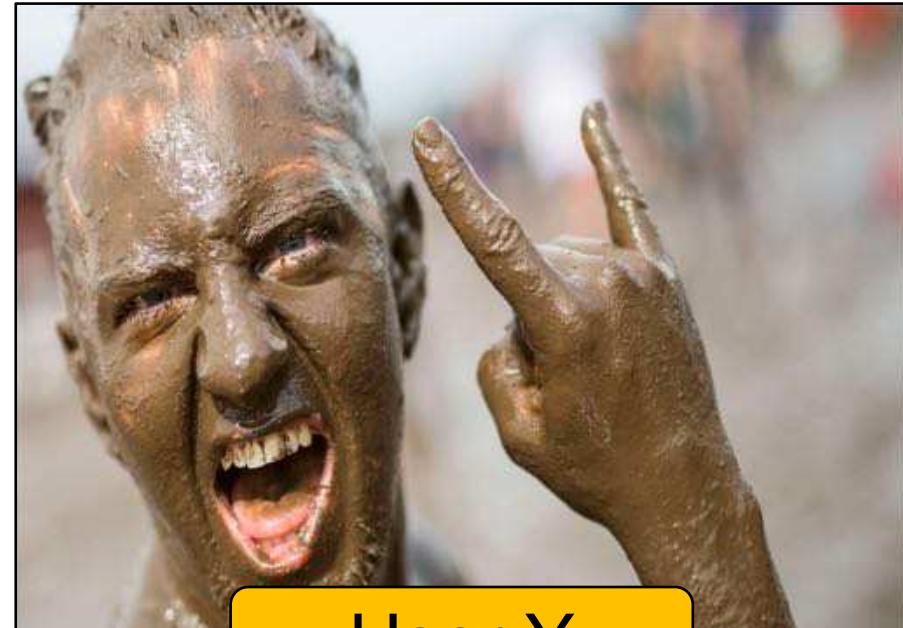
buys



buys



buys



User Y



buys



buys

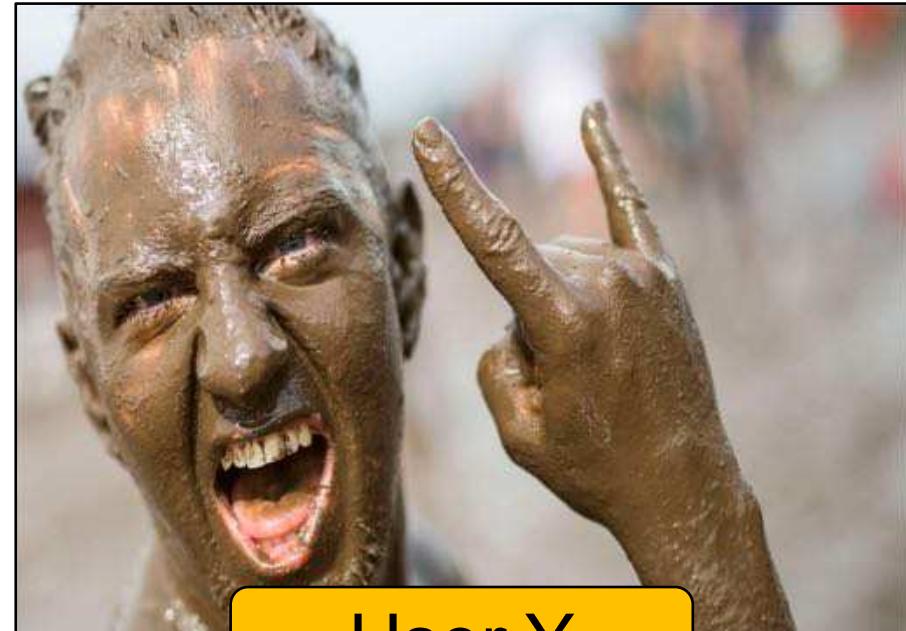


recommended

# Recommender Systems in a nutshell



Users



User Y



buy



buys



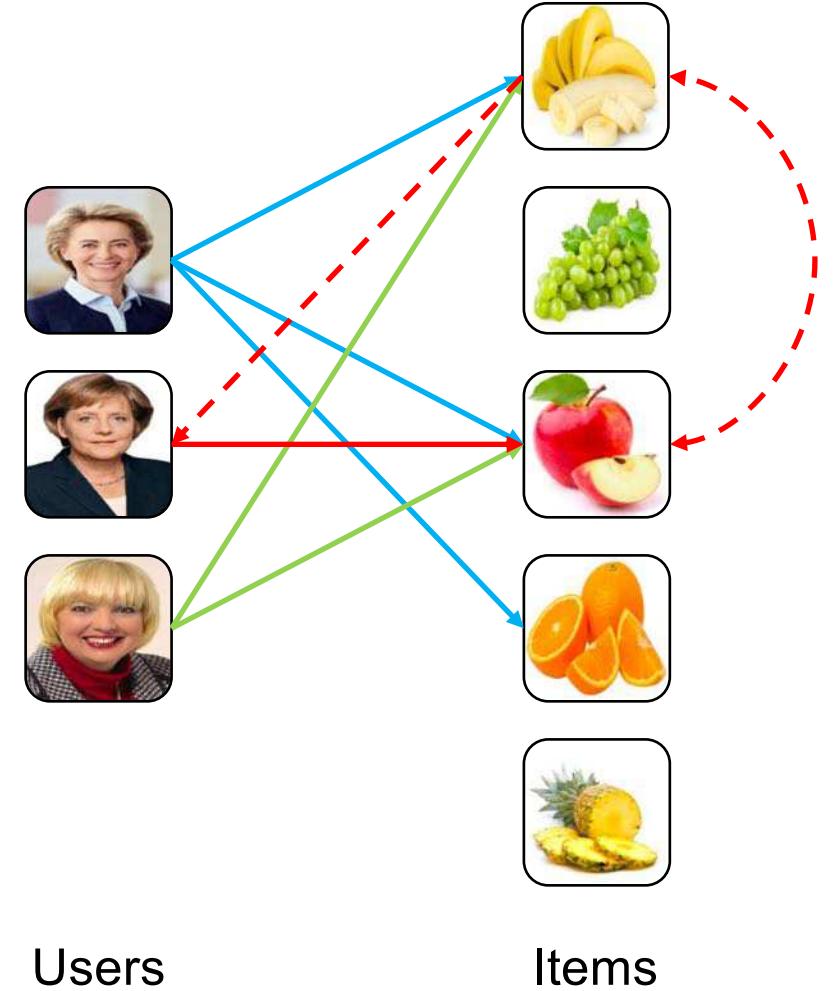
buys



recommended

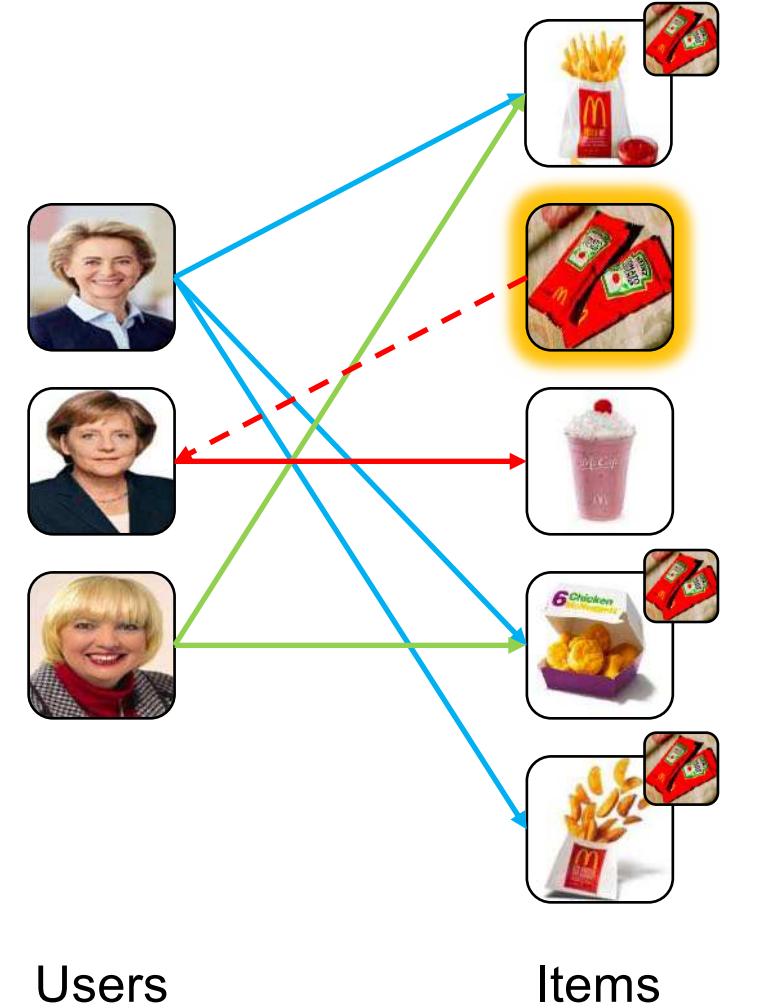
# Types of Recommendations

- **editorial and hand curated**
  - lists of favorites
  - lists of “essential” items
- **simple aggregates**
  - Top 10
  - most popular
  - most recent
- **tailored to individual users**
  - content based
  - collaborative filtering
  - latent factor based
  - deep social collaborative filtering



# Types of Recommendations

- editorial and hand curated
  - lists of favorites
  - lists of “essential” items
- simple aggregates
  - Top 10
  - **most popular**
  - most recent
- tailored to individual users
  - content based
  - collaborative filtering
  - latent factor based
  - deep social collaborative filtering



# Formal Model

- $C := \{\text{users}\}$  (set of users)
- $S := \{\text{items}\}$  (set of items)
- $u := \text{rating function}$  of items for users
  - $u : C \times S \rightarrow R$
  - $R := \{\text{recommended items}\}$  (set of recommended items)
- the rating function  $u$  chooses for every user  $c$  in the set  $C$  the most valuable items  $s$  from the set  $S$ :

$$c \in C \quad s'_c = \operatorname{argmax}_u u(c, s)$$

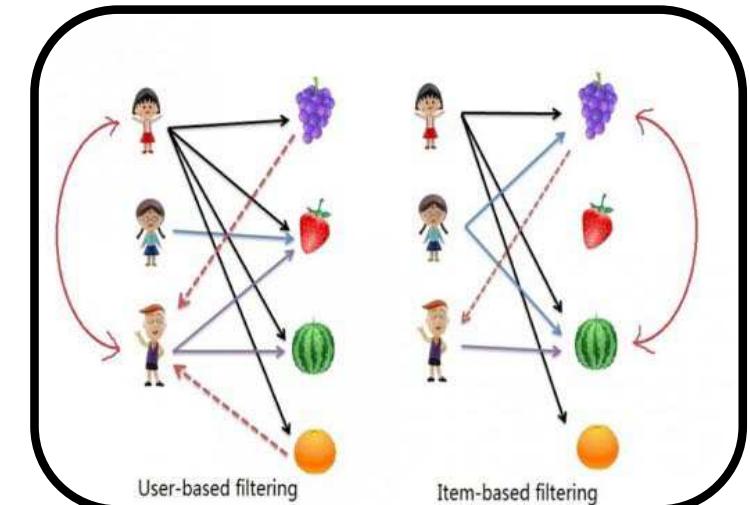
# A good recommender ...

- recommends personalized and valuable items within the current context for the user
- recommends items from all different and possible areas of interest for the user
- does not recommend items which are already known by the user
- extends the areas of interest of the user (serendipity effect)



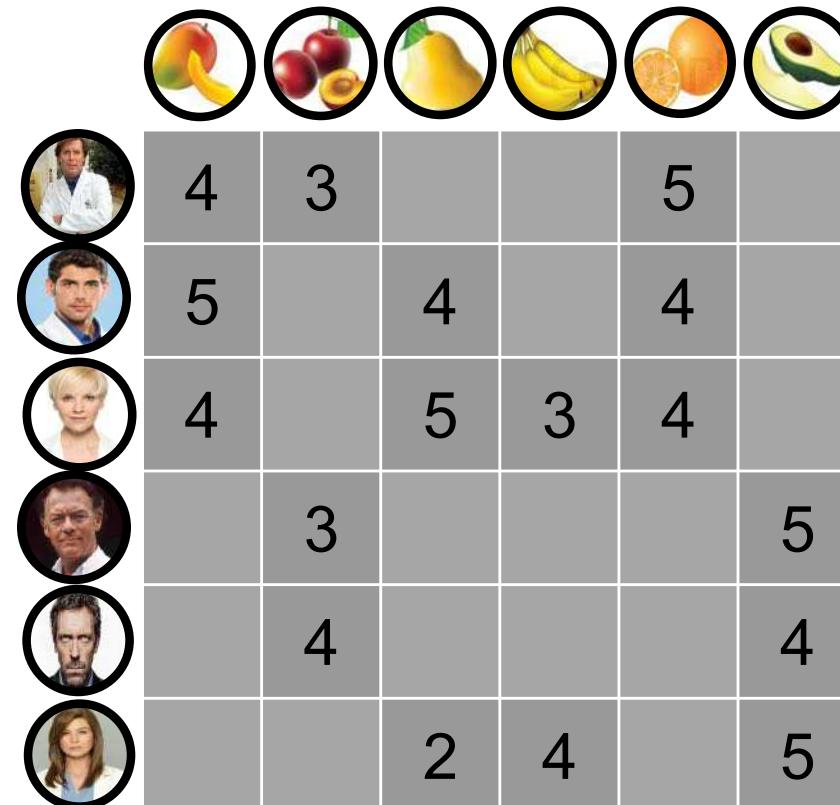
# Collaborative Filtering

- analysis of behavior pattern of user groups
- recommends items to a user based on the comparison of his/her behavior against the users with a "similar" behavior
- two alternatives: user-based or item-based
- challenges:
  - many items to recommend
  - only a few can be recommended
  - mostly sparsely data per user
  - interest measure can be very diverse
    - explicit: rating, stars, like
    - implicit: purchases, clicks, reading time
  - no data for new users or items



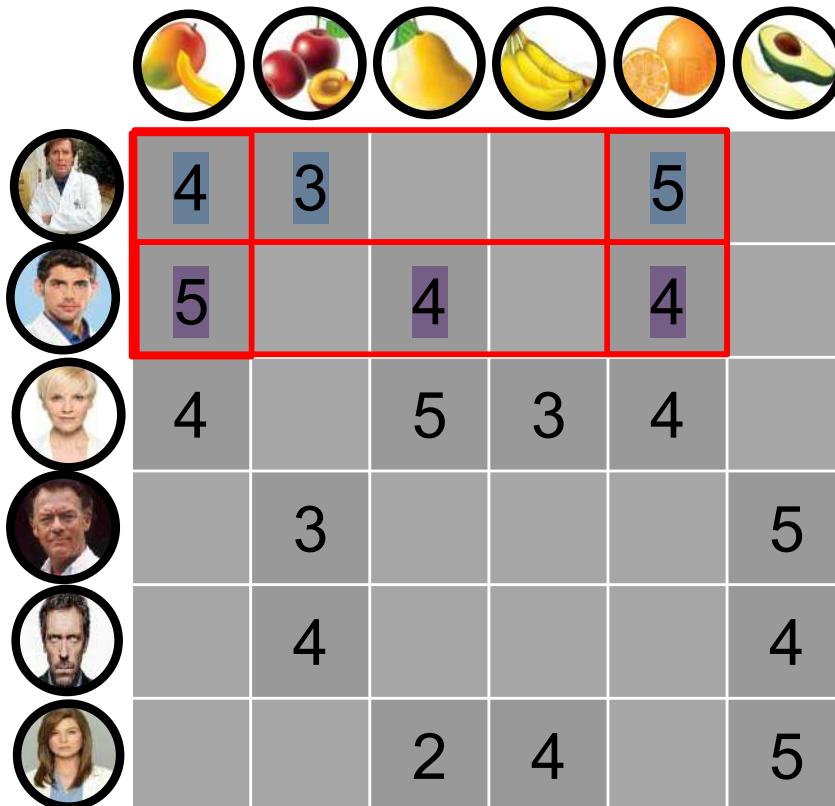
# Collaborative Filtering

- example for user-based collaborative filtering
  - calculation of the similarity between the users based on their rating behavior

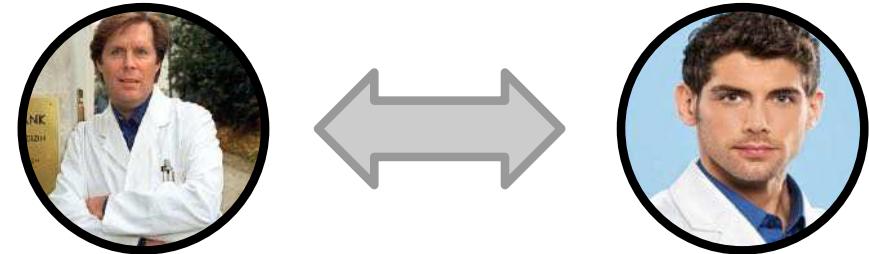


# Collaborative Filtering

- example for user-based collaborative filtering
  - calculation of the similarity between the users based on their rating behavior



$$\text{similarity} = \cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\| \|\mathbf{B}\|} = \frac{\sum_{i=1}^n A_i B_i}{\sqrt{\sum_{i=1}^n A_i^2} \sqrt{\sum_{i=1}^n B_i^2}}$$

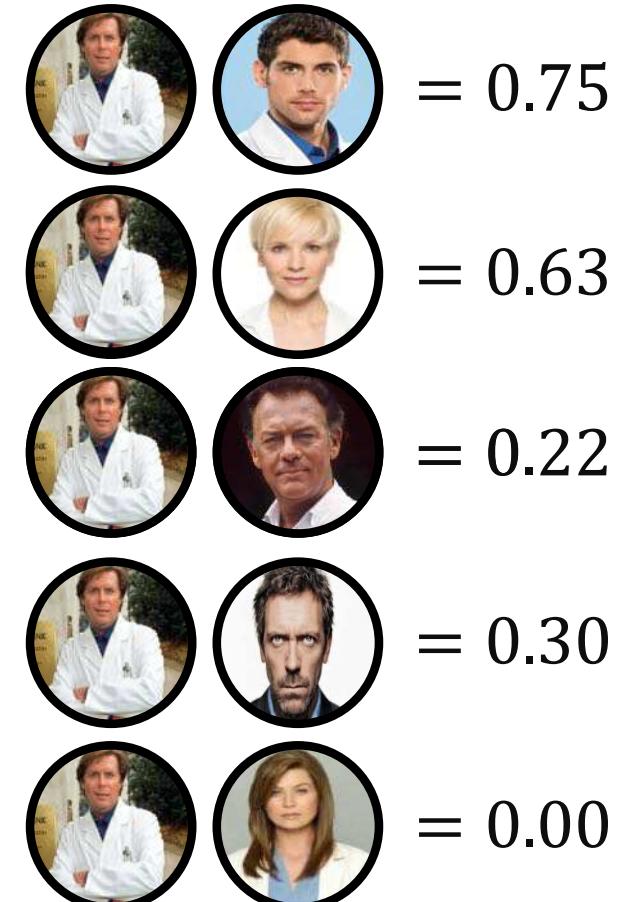


$$\frac{(4 * 5) + (5 * 4)}{\sqrt{16 + 9 + 25} * \sqrt{25 + 16 + 16}} = 0.75$$

# Collaborative Filtering

- example for user-based collaborative filtering
  - calculation of the similarity between the users based on their rating behavior

	Mango	Apples	Pear	Banana	Orange	Avocado
Greg	4	3			5	
Greg	5		4		4	
Greg	4		5	3	4	
Greg		3			5	
Greg	4				4	
Greg		2	4		5	



# Collaborative Filtering

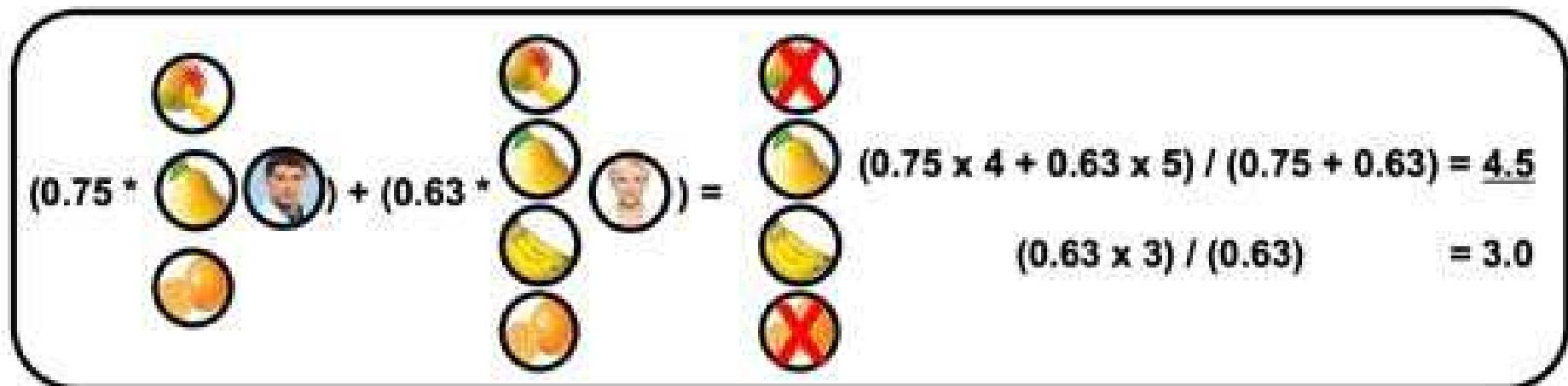
- example for user-based collaborative filtering
  - calculation of the **similarity** between the users based on their rating behavior

	4	3			5	
	5		4		4	
	4		5	3	4	
		3				5
		4				4
			2	4		5

		0.75	0.63	0.22	0.30	0.00
	0.75		0.91	0.00	0.00	0.16
	0.63	0.91		0.00	0.00	0.40
	0.22	0.00	0.00		0.97	0.64
	0.30	0.00	0.00	0.97		0.53
	0.00	0.16	0.40	0.64	0.53	

# Collaborative Filtering

- example for user-based collaborative filtering
  - calculation of recommendations for the target user

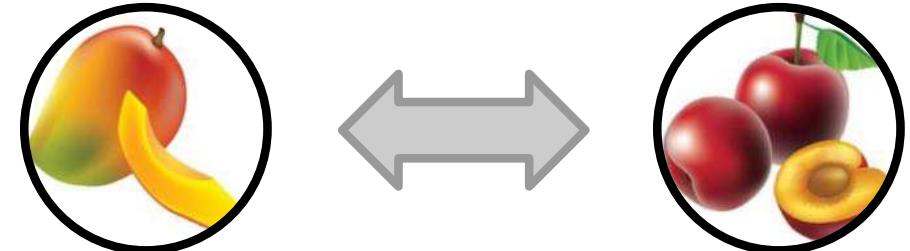


# Collaborative Filtering

- example for item-based collaborative filtering
  - calculation of the similarity between the items based on the rating behavior of the users

	4	3			5	
	5		4		4	
	4		5	3	4	
		3			5	
		4				4
			2	4		5

$$\text{similarity} = \cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\| \|\mathbf{B}\|} = \frac{\sum_{i=1}^n A_i B_i}{\sqrt{\sum_{i=1}^n A_i^2} \sqrt{\sum_{i=1}^n B_i^2}}$$

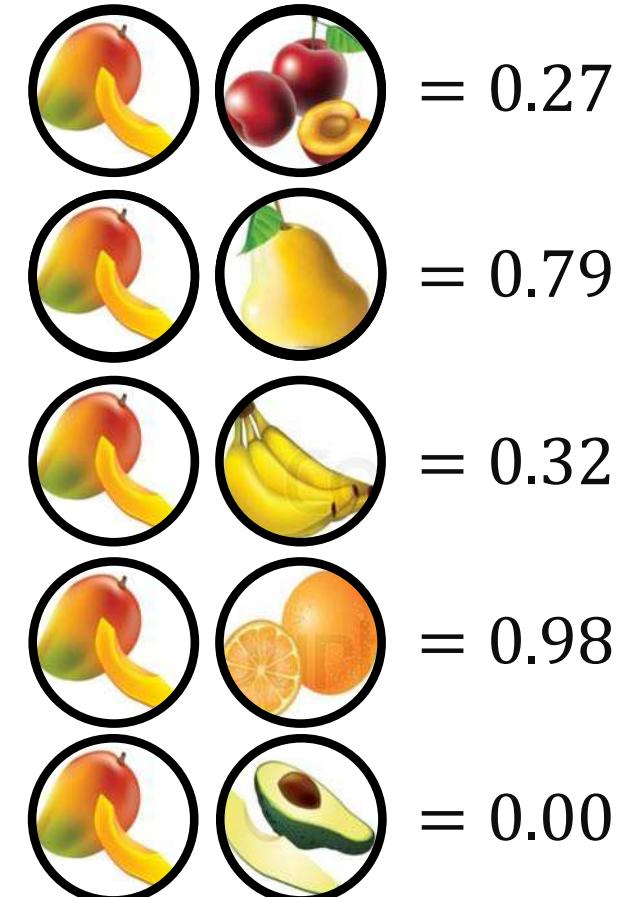


$$\begin{aligned} & \frac{(4 * 3)}{\sqrt{16 + 25 + 16} * \sqrt{9 + 9 + 16}} \\ &= 0.27 \end{aligned}$$

# Collaborative Filtering

- example for item-based collaborative filtering
  - calculation of the similarity between the items based on the rating behavior of the users

	Mango	Apples	Pear	Banana	Orange	Avocado
House M.D.	4	3			5	
Dr. Cox	5		4		4	
Chase	4		5	3	4	
Cameron		3			5	
Gregory		4			4	
Wilson			2	4		5



# Collaborative Filtering

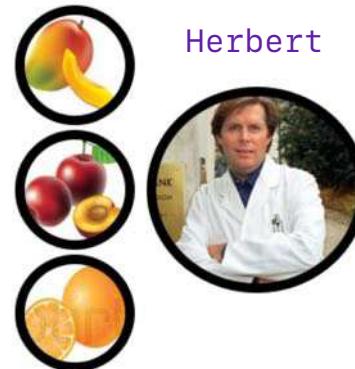
- example for item-based collaborative filtering
  - calculation of the similarity between the items based on the rating behavior of the users

	4	3			5	
	5		4		4	
	4		5	3	4	
		3				5
		4				4
			2	4		5

		0.27	0.79	0.32	0.98	0.00
	0.27		0.00	0.00	0.34	0.65
	0.79	0.00		0.69	0.71	0.18
	0.32	0.00	0.69		0.32	0.49
	0.98	0.34	0.71	0.32		0.00
	0.00	0.65	0.18	0.49	0.00	

# Collaborative Filtering

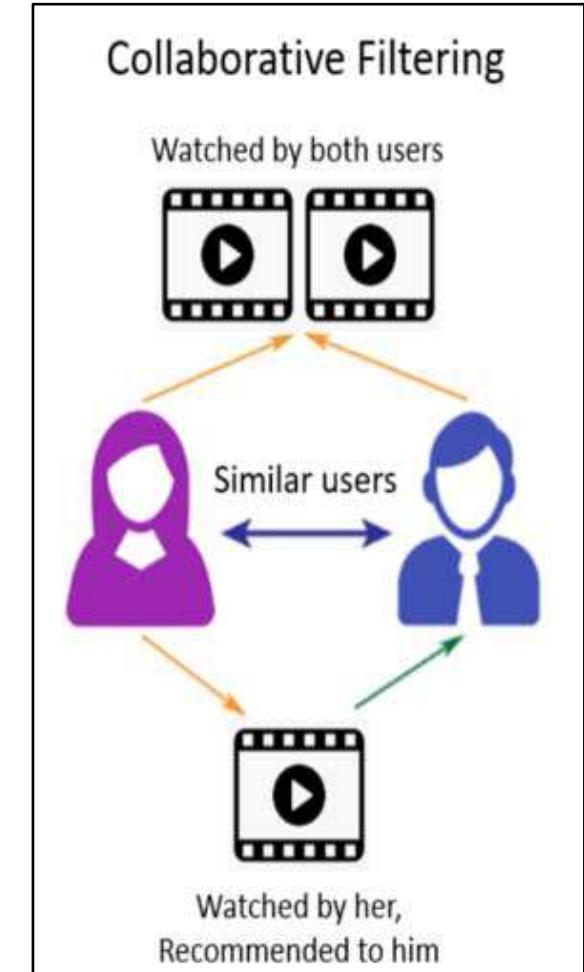
- example for item-based collaborative filtering
  - calculation of recommendations for the target user



$$(4 \cdot \text{apple}) + (3 \cdot \text{orange}) + (5 \cdot \text{mango}) = \frac{(0.79 * 4 + 0.71 * 5)}{(0.79 + 0.71)} = \underline{\underline{4.5}}$$
$$(\text{apple}) \quad (\text{orange}) \quad (\text{mango})$$
$$(0.65 * 3) / 0.65 = 3.0$$

# Collaborative Filtering

- pros:
  - relatively easy to implement
  - creates fairly good results for most cases
  - the results are produced by a data driven algorithm
- cons:
  - needs interest measure of items from users
  - users need to be active
  - assumes that current behavior pursues historic behavior of the user Funktioniert nicht, wenn sich Interessen des user ändern
  - sparsity and cold start problems
  - the results are produced by a data driven algorithm

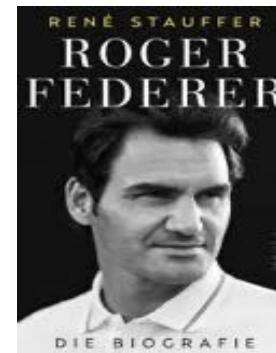


# Problems

- **Sparsity Problem**
  - mostly we have **very large object sets** (e.g., products, contents, people) and **just a few ratings**
  - the **majority** of ratings are **negative**
  - the probability that **various users rated the same objects** and are interested in further common objects is **mostly very low**
- **Cold Start Problem**
  - new user problem: the system needs to **become familiar** with the new user and to **learn the behavior and preferences**
  - new item problem: **without any rating for an object** it is **impossible to recommend it to any users**
- Solution: **hybrid recommender** with **collaborative** and **content-based methods**

# Content-based Filtering

- recommendations are based on the content or the attributes of the objects instead of the user behavior and ratings
- explicit attributes (meta-data) of objects
  - movies: genre, actors, duration, year, location
  - holidays: kids club, country, temperature
- content of objects
  - textual content (e.g., topics)
  - characterization of the content (e.g., storybook, biography)



# Content-based Filtering

- ideal to use for text-based products (e.g., webpages, books)
- objects are described by extracted features (e.g., keywords, topics)
- users are also described by extracted features of their behavior or profile
- recommendations are calculated by distance measures between the feature sets

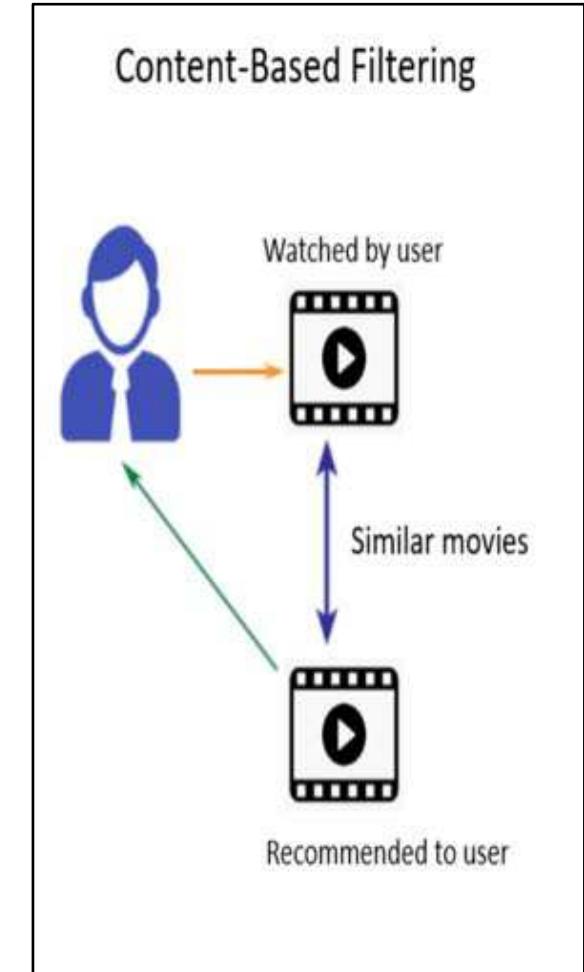


watched by user

recommended

# Content-based Filtering

- pros:
  - no need for other user data
  - no cold-start or sparsity problem
  - individual interests are more considered
  - new and less famous objects are also recommended
  - easier to interpret, cause of the feature representation and matching
- cons:
  - only usable for objects, which can be meaningfully described by features
  - serendipity effect is not really supported



# Content-based Filtering

- example:
  - a user watched the series Breaking Bad



# Content-based Filtering

- example:
  - a user watched the series Breaking Bad



creator: Vince Gilligan



recommended

# Content-based Filtering

- example:
  - a user watched the series Breaking Bad



actor: Bryan Cranston



recommended

# Content-based Filtering

- example:

- a customer buys the book with the title „Building **data mining** applications for **CRM**“
- 7 Books are possible candidates for a recommendation:
  - „Accelerating Customer Relationships: Using **CRM** and Relationship Technologies“
  - „Mastering Data Mining: The Art and Science of **Customer Relationship Management**“
  - „**Data Mining** Your Website“
  - „Introduction to marketing“
  - „**Consumer** behaviour“
  - „Marketing research, a handbook“
  - „**Customer** knowledge management“

# Content-based Filtering

## ■ example

Count	Author	Title	Genre	Category	Sub-Category	Format	Language	Price	Rating	Reviews	Stock Status	Last Update
1	D. J. Patil	Building data mining applications for CRM: Accelerating customer relationships using CRM and relationship technologies	Business	Data Mining	Marketing	Handbook	English	Rs. 1200	4.5	10	In Stock	2023-10-15
1	John W穷	Marketing Data Mining: the art and science of Customer Relationship Management	Business	Data Mining	Marketing	Handbook	English	Rs. 1500	4.8	12	In Stock	2023-10-15
1	John W穷	Data Mining your website	Business	Data Mining	Marketing	Handbook	English	Rs. 1000	4.2	8	In Stock	2023-10-15
1	John W穷	Introduction to Marketing	Business	Data Mining	Marketing	Handbook	English	Rs. 800	4.0	6	In Stock	2023-10-15
1	John W穷	Consumer behavior	Business	Data Mining	Marketing	Handbook	English	Rs. 900	4.3	7	In Stock	2023-10-15
1	John W穷	Marketing Research: a Handbook	Business	Data Mining	Marketing	Handbook	English	Rs. 1100	4.6	9	In Stock	2023-10-15
1	John W穷	Customer Knowledge Management	Business	Data Mining	Marketing	Handbook	English	Rs. 1300	4.9	11	In Stock	2023-10-15

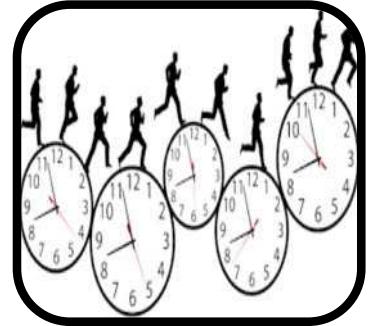
# Content-based Filtering

## ■ example

Count	Website	Category	Author	Title	Published	Language	Series	Format	Price	Rating	Reviews
1	Marketing	Marketing	John Haider	Building data mining applications for CRM: Accelerating customer relationships using CRM and relationship technologies	2010	English	Marketing	Book	\$29.99	4.5	12
1	Marketing	Marketing	John Haider	Mastering Data Mining: the art and science of Customer Relationship Management	2010	English	Marketing	Book	\$29.99	4.5	12
1	Marketing	Marketing	John Haider	Data Mining your website	2010	English	Marketing	Book	\$29.99	4.5	12
1	Marketing	Marketing	John Haider	Introduction to Marketing	2010	English	Marketing	Book	\$29.99	4.5	12
1	Marketing	Marketing	John Haider	Consumer behavior	2010	English	Marketing	Book	\$29.99	4.5	12
1	Marketing	Marketing	John Haider	Marketing Research: a Handbook	2010	English	Marketing	Book	\$29.99	4.5	12
1	Marketing	Marketing	John Haider	Customer Knowledge Management	2010	English	Marketing	Book	\$29.99	4.5	12

# Context-aware Filtering

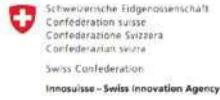
- context is a dynamic collection of factors, which describe the status of a user
- context factors can change very fast and influence the decision of a user about the relevance of a recommendation
- time factor: time of day, weekday, weekend, month
- space factor: at home, work, outdoor, indoor
- social factor: family, friends, party
- context-aware recommendations should include these factors



# Context-aware Filtering

- **pre-filtering**
  - filtering of context-relevant information within the data
  - relatively easy to implement
  - reduces the considering data sets
  - leads to an increasing sparsity problem
- **post-filtering**
  - filtering of recommendations after the calculation due to relevance to the context
  - does not consider the context directly
  - context irrelevant data can influence the recommendations
  - calculation of the recommendations must be done on the whole data set

# Real-world project: Skillue HR Reco

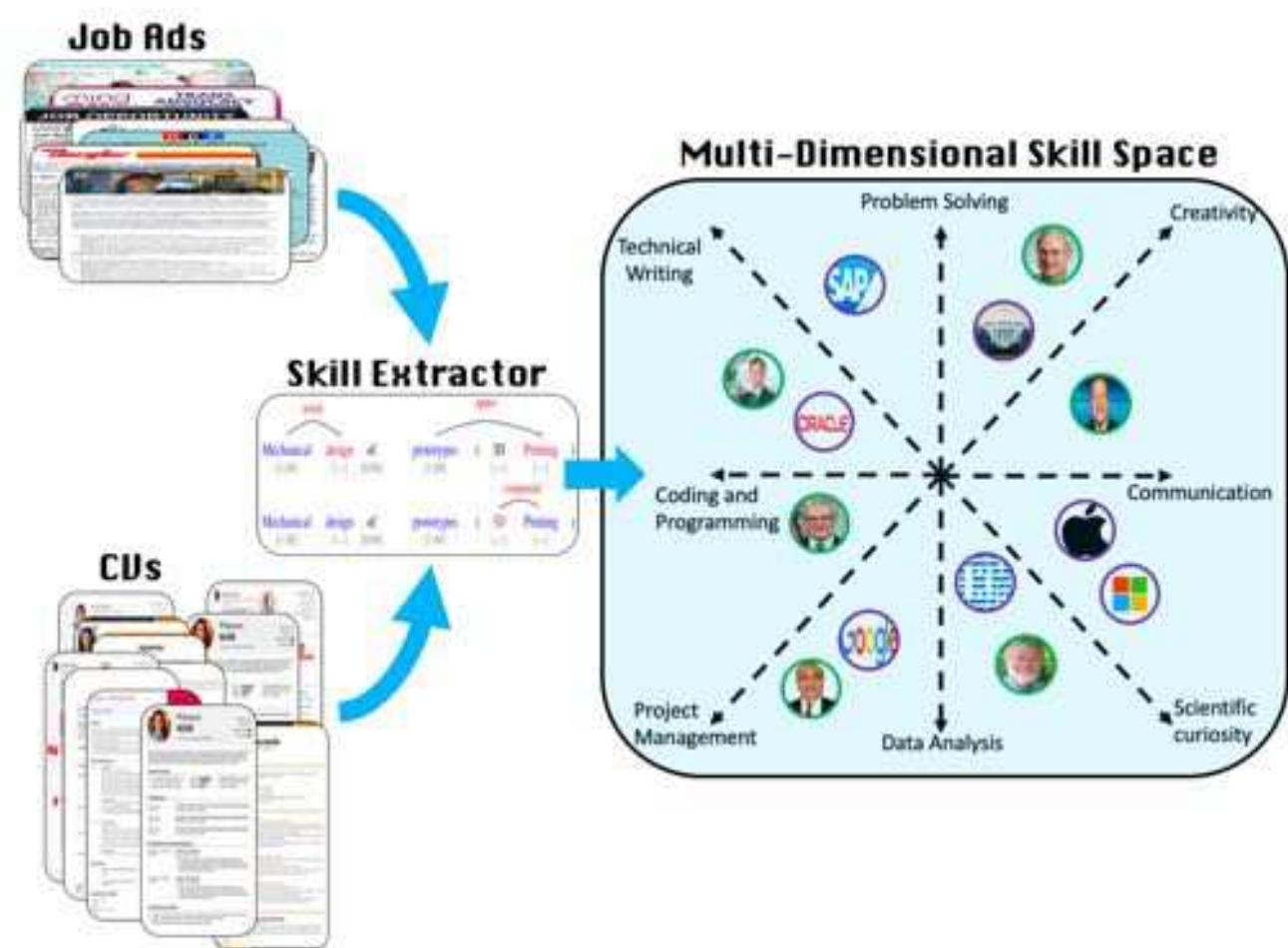


**SKILLUE**  
Job Recommendation System

**Challenge:** A system for recommending jobs for persons, persons for companies and skills for persons.

**Method:** Very large search space with extracted skills from job advertisements and cvs.

«Syntax-based Skill Extractor for Job Advertisements». SDS'2019. Smith, Braschler, Haberthuer, Weiler (2019).



# Lessons Learned

- user-based collaborative filtering
- item-based collaborative filtering
- content-based filtering
- problems of recommender systems
- examples of real-world recommender systems



# Questions and Answers



# Machine Learning and Data Mining

## V06: Association Rules

Lecturer: **Dr. Andreas Weiler**



# Repetition

... REPETITION ...  
→ IS THE MOTHER  
→ of learning

# Course Content IT20a\_WIN

Week	Lecture	Lab
KW08	Introduction	Lab 01
KW09	Data Processing Part 01	Lab 02
KW10	Carnival Monday	
KW11	Data Processing Part 02	Lab 03
KW12	Evaluation	Lab 04
KW13	Recommender Systems	<b>Graded Lab 01</b> <span style="border: 1px solid red; padding: 2px;">IMPORTANT</span>
KW14	Association Rules	<b>Graded Lab 01</b> <span style="border: 1px solid red; padding: 2px;">IMPORTANT</span>
KW15	Linear and Logistic Regression	Lab 05
KW16	Easter Monday	
KW17	Support Vector Machines	<b>Graded Lab 02</b> <span style="border: 1px solid red; padding: 2px;">IMPORTANT</span>
KW18	Decision Trees and Naïve Bayes	<b>Graded Lab 02</b> <span style="border: 1px solid red; padding: 2px;">IMPORTANT</span>
KW19	Partition Clustering	Lab 06
KW20	Hierarchical and Density Clustering	Lab 07
KW21	Neural Networks and Closing	Review and Questions for Exam

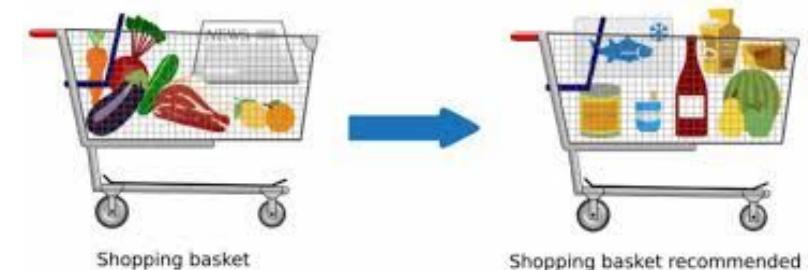
# Learning Objectives

- understanding of frequent itemset mining
- understanding of mining various kinds of association rules



# Introduction

- find associations between the different items that customers place in their shopping baskets or carts
- develop marketing strategies according to which items are frequently purchased together
  - where should products be placed in the store to optimize their sales?
  - which products are bought together? change over time?
  - does the brand make a difference?
  - how are the demographics of the neighborhood affecting what customers are buying?



# Mining frequent patterns

- find all rules that correlate the presence of one set of items with that of another set of items
- itemset
  - a collection of one or more items
  - K-itemset, an itemset that contains k items
- support count ( $\sigma$ )
  - frequency of occurrence
  - $\sigma\{\text{milk, bread}\} = 5$
  - $\sigma\{\text{coke, chips}\} = 4$
- support (s)
  - fraction of transactions that contain an itemset
  - $s(\{\text{milk, bread}\}) = 5/8$
  - $s(\{\text{coke, chips}\}) = 4/8$

T#	items
1	bread, peanuts, milk, apple, honey
2	bread, honey, coke, chips, milk
3	steak, honey, coke, chips, bread
4	honey, coke, peanuts, milk, apple
5	honey, coke, chips, milk, bread
6	apple, coke, chips, milk
7	apple, coke, peanuts, milk, bread
8	apple, peanuts, cheese, milk, bread

# Mining frequent patterns

- **frequent itemset**
  - an itemset whose support is greater than or equal to a minimum support threshold
- **association rule**
  - an implication expression of the form X → Y, where X and Y are frequent itemsets
  - example: {milk, bread} → {honey}
- **rule evaluation metrics**
  - **support:** fraction of transactions that contain both X and Y
  - **confidence:** measures how often items in Y appear in transactions that contain X

T#	items
1	bread, peanuts, milk, apple, honey
2	bread, honey, coke, chips, milk
3	steak, honey, coke, chips, bread
4	honey, coke, peanuts, milk, apple
5	honey, coke, chips, milk, bread
6	apple, coke, chips, milk
7	apple, coke, peanuts, milk, bread
8	apple, peanuts, cheese, milk, bread

# Mining frequent patterns

- rule evaluation metrics
  - support:  $s = \sigma(X \cup Y) / |T|$
  - confidence:  $c = \sigma(X \cup Y) / \sigma(X)$
- example
  - $\{\text{milk, bread}\} \rightarrow \{\text{honey}\}$
  - $s = \frac{\sigma(\text{milk, bread, honey})}{|T|} = 3/8 = 0.375$
  - $c = \frac{\sigma(\text{milk, bread, honey})}{\sigma(\text{milk, bread})} = 3/5 = 0.6$
- given a set of transactions T, the goal of association rule mining is to find all rules having
  - support  $\geq$  minsup threshold
  - confidence  $\geq$  minconf threshold

T#	items
1	bread, peanuts, milk, apple, honey
2	bread, honey, coke, chips, milk
3	steak, honey, coke, chips, bread
4	honey, coke, peanuts, milk, apple
5	honey, coke, chips, milk, bread
6	apple, coke, chips, milk
7	apple, coke, peanuts, milk, bread
8	apple, peanuts, cheese, milk, bread

# Types of association rules

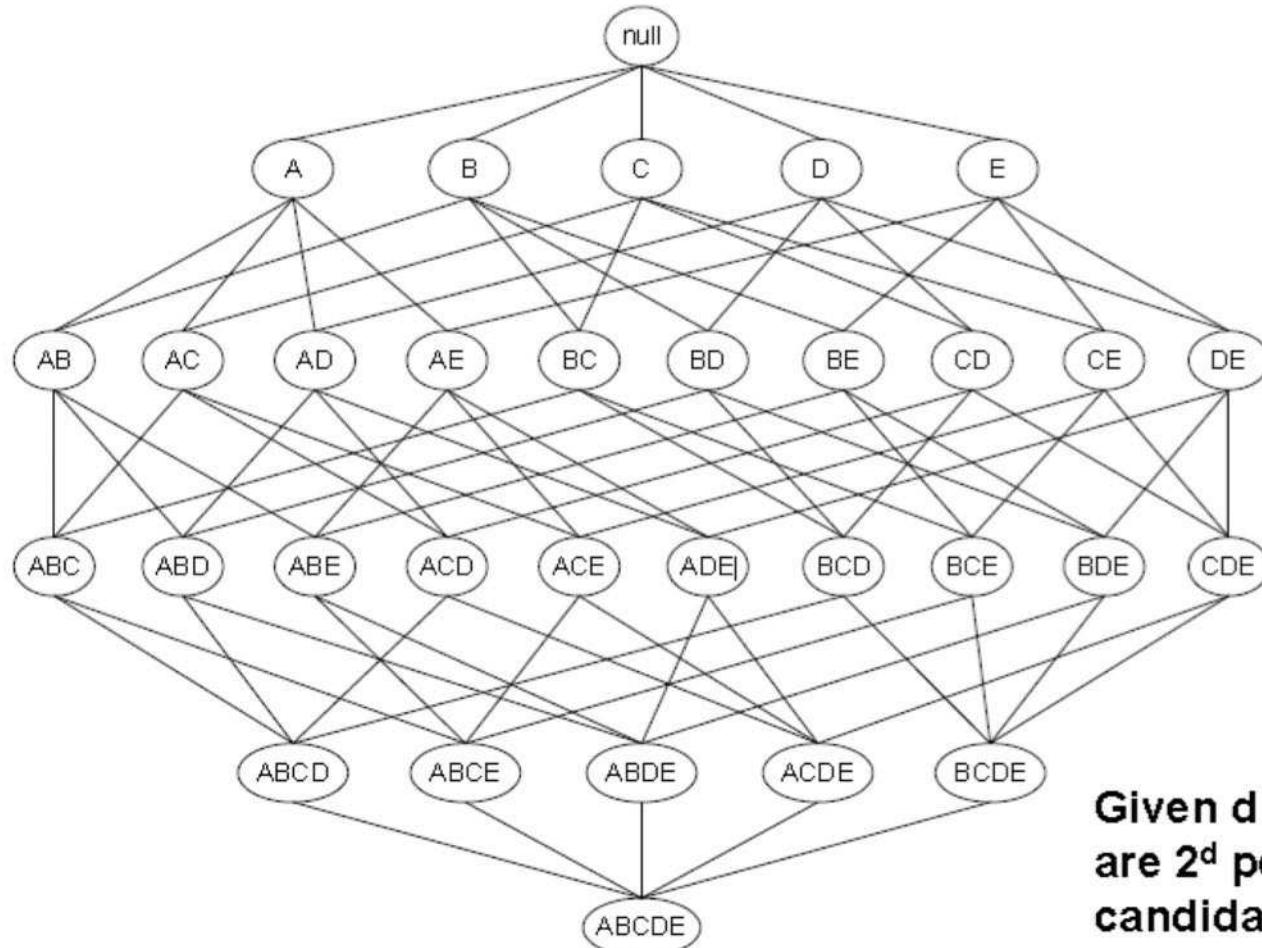
- one-dimensional
  - $\text{buys}(\text{X}, \text{"iPhone 12"}) \rightarrow \text{buys}(\text{X}, \text{"charging cable"})$
  - [support = 18%, confidence = 63%]
  - 18% of all transactions showed that an iPhone 12 and a charging cable were bought together
  - if a customer buys an iPhone 12 there is a chance of 63% that he/she will buy a charging cable as well
- multi-dimensional
  - $\text{age}(\text{X}, \text{"[20,...,29]"}) \wedge \text{income}(\text{X}, \text{"[40K,...,49K"]}) \rightarrow \text{buys}(\text{X}, \text{"iPhone 12 Pro"})$
  - [support = 2%, confidence = 60%]

# Association Rule Mining

- two step process
  - find all frequent itemsets that meet a defined minimum support threshold
  - generate strong association rules that meet a defined minimum support threshold and a defined minimum confidence threshold
- the possible number of frequent itemsets is huge
  - if an itemset is frequent, each of its subsets is frequent as well
  - for example a frequent itemset of length 100 contains  $\binom{100}{1} = 100$  frequent 1-itemsets,  $\binom{100}{2} = 100$  frequent 2-itemsets, ...  
 $\rightarrow 2^{100} - 1$  possible frequent itemsets

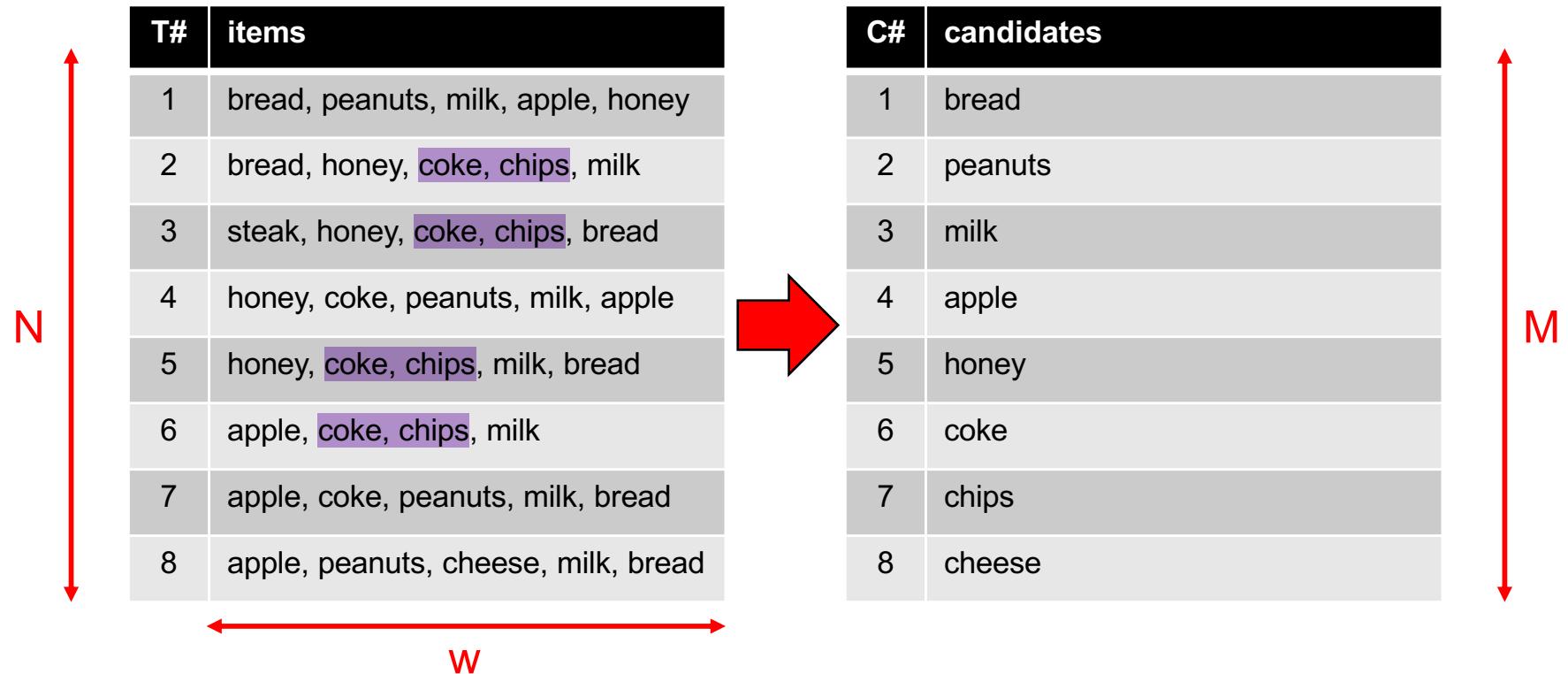
# Association Rule Mining

- number of itemsets



# Frequent Item Set Mining

- brute-force approach for frequent itemset generation
  - each itemset is a candidate frequent itemset
  - count the support of each candidate
  - complexity:  $O(NMw)$

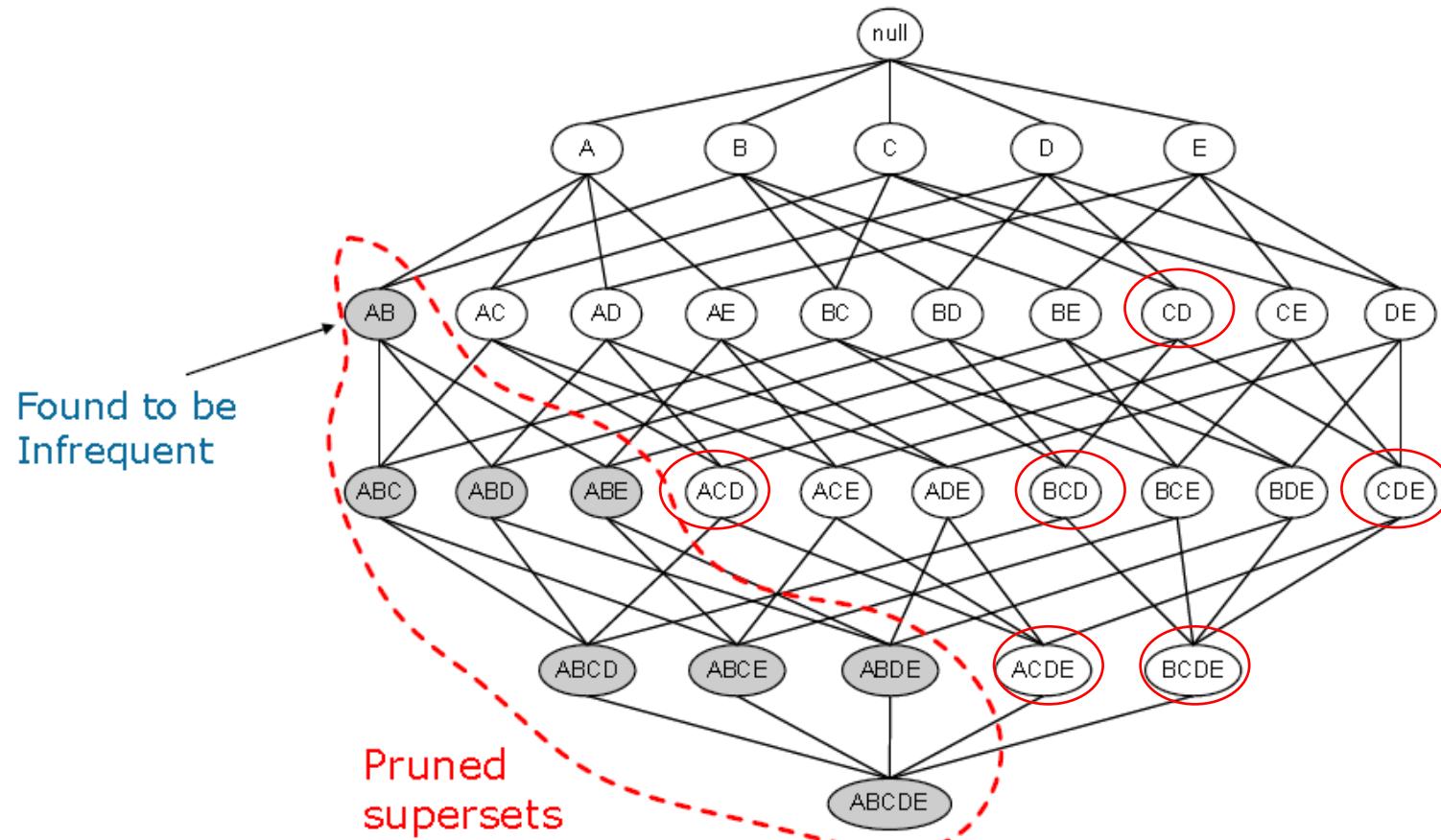


# Frequent Item Set Mining

- **improvements**
  - reduce the number of candidates ( $M$ ) by using pruning techniques
  - reduce the number of transactions ( $N$ ) by reducing the size of  $N$  as the size of itemset increases
  - reduce the number of comparisons ( $NM$ ) by using efficient data structures to store the candidates or transactions
- **idea of apriori principle**
  - if an itemset is frequent, than all of its subsets must also be frequent
  - so if an itemset is infrequent, its superset must not be tested

# Frequent Item Set Mining

- idea of apriori principle



# The Apriori algorithm

- initially, scan DB once to get frequent 1-itemset
- generate length  $(k+1)$  candidate itemsets from length  $k$  frequent itemsets
- test the candidates against DB
- terminate when no frequent or candidate set can be generated

```
from efficient_apriori import apriori
transactions = [('eggs', 'bacon', 'soup'),
                 ('eggs', 'bacon', 'apple'),
                 ('soup', 'bacon', 'banana')]
itemsets, rules = apriori(transactions, min_support=0.5, min_confidence=1)
print(rules) # [{eggs} -> {bacon}, {soup} -> {bacon}]
```

# The Apriori algorithm

- pseudo code of the apriori algorithm

$C_k$ : Candidate itemset of size k

$L_k$  : frequent itemset of size k

$L_1 = \{\text{frequent items}\};$

**for** ( $k = 1; L_k \neq \emptyset; k++$ ) **do begin**

$C_{k+1} = \text{candidates generated from } L_k;$  //Join & Prune

**for each** transaction  $t$  in database **do**

increment the count of all candidates in  $C_{k+1}$   
that are contained in  $t$

$L_{k+1} = \text{candidates in } C_{k+1} \text{ with min\_support}$  //Prune minSup

**end**

**return**  $\cup_k L_k;$

# Association Rule Mining: Example

- consider a database D, consisting of 9 transactions
- minimum support count required is 2 (i.e.,  $\text{min\_sup} = 2/9 = 22\%$ )
- minimum confidence required is 70%
- we first find out the frequent itemset using the Apriori algorithm
- then, association rules will be generated using minimum support and minimum confidence thresholds

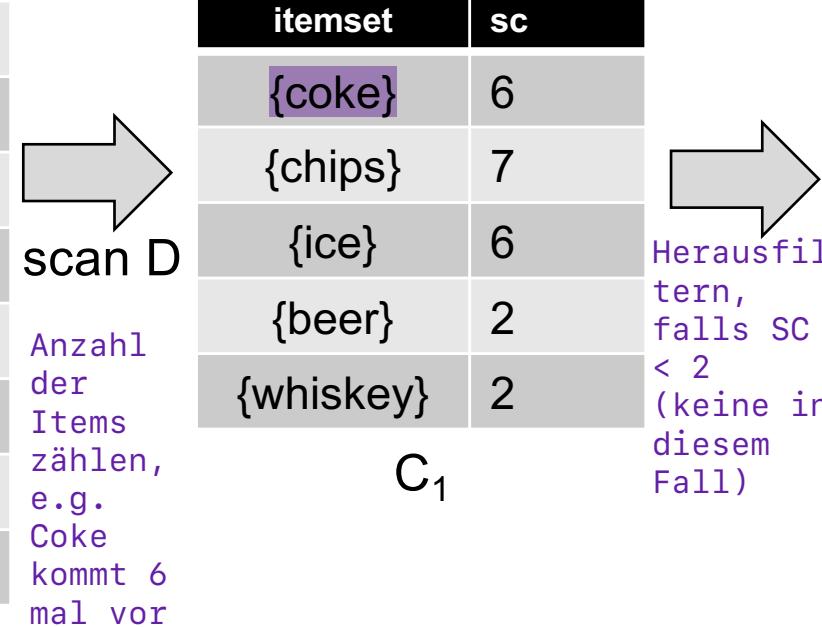
T#	items
1	coke, chips, whiskey
2	chips, beer
3	chips, ice
4	coke, chips, beer
5	coke, ice
6	chips, ice
7	coke, ice
8	coke, chips, ice, whiskey
9	coke, chips, ice

database D

# Association Rule Mining: Example

- step 1: generating 1-itemset frequent pattern
  - $L_1$  consists of the candidate 1-itemsets satisfying minimum support
  - in the first iteration each item is a member of the set of candidate.

T#	items
1	coke, chips, whiskey
2	chips, beer
3	chips, ice
4	coke, chips, beer
5	coke, ice
6	chips, ice
7	coke, ice
8	coke, chips, ice, whiskey
9	coke, chips, ice



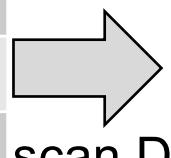
# Association Rule Mining: Example

T#	items
1	coke, chips, whiskey
2	chips, beer
3	chips, ice
4	coke, chips, beer
5	coke, ice
6	chips, ice
7	coke, ice
8	coke, chips, ice, whiskey
9	coke, chips, ice

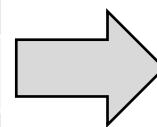
database D

SC = Support Count

itemset
{coke, chips}
{coke, ice}
{coke, beer}
{coke, whiskey}
{chips, ice}
{chips, beer}
{chips, whiskey}
{ice, beer}
{ice, whiskey}
{beer, whiskey}



itemset	sc
{coke, chips}	4
{coke, ice}	4
{coke, beer}	1
{coke, whiskey}	2
{chips, ice}	4
{chips, beer}	2
{chips, whiskey}	2
{ice, beer}	0
{ice, whiskey}	1
{beer, whiskey}	0



itemset	sc
{coke, chips}	4
{coke, ice}	4
{coke, whiskey}	2
{chips, ice}	4
{chips, beer}	2
{chips, whiskey}	2

L<sub>2</sub>

# Association Rule Mining: Example

- step 3: generating 3-itemset frequent pattern
  - in order to find  $C_3$ , we compute  $L_2 \text{ Join } L_2$
  - $C_3 = L_2 \text{ Join } L_2 = \{\{\text{coke, chips, ice}\}, \{\text{coke, chips, whiskey}\}, \{\text{coke, ice, whiskey}\}, \{\text{chips, ice, beer}\}, \{\text{chips, ice, whiskey}\}, \{\text{chips, beer, whiskey}\}\}$
  - now, join step is complete and Prune step will be used to reduce the size of  $C_3$ . Prune step helps to avoid heavy computation due to large  $C_k$
  - we can determine that four candidates cannot possibly be frequent (Apriori property)
  - for example, lets take  $\{\text{coke, chips, ice}\}$ . The 2-item subsets of it are  $\{\text{coke, chips}\}$ ,  $\{\text{coke, ice}\}$ , and  $\{\text{chips, ice}\}$ . Since all 2-item subsets of  $\{\text{coke, chips, ice}\}$  are members of  $L_2$ , We will keep  $\{\text{coke, chips, ice}\}$  in  $C_3$ .

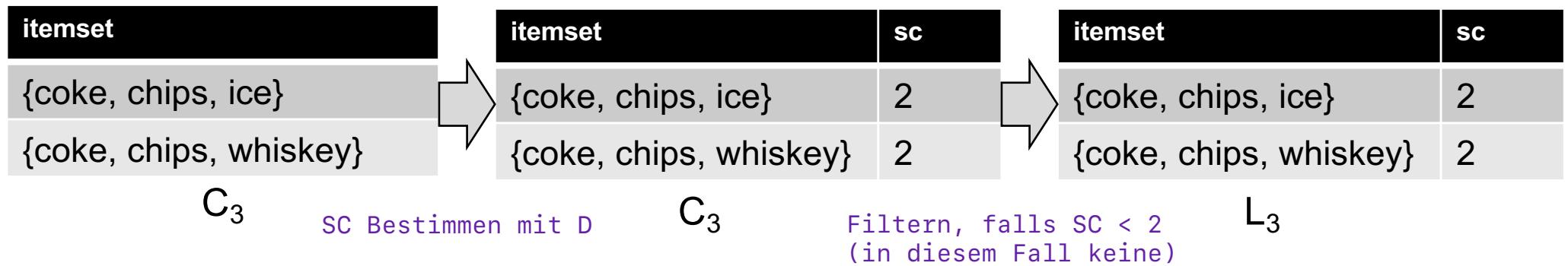
Zürcher Hochschule  
für Angewandte  
Wissenschaften

T#	items
1	coke, chips, whiskey
2	chips, beer
3	chips, ice
4	coke, chips, beer
5	coke, ice
6	chips, ice
7	coke, ice
8	coke, chips, ice, whiskey
9	coke, chips, ice

database D

# Association Rule Mining: Example

- step 3: generating 3-itemset frequent pattern
  - lets take another example of {chips, ice, whiskey} which shows how the pruning is performed. The 2-item subsets are {chips, ice}, {chips, whiskey}, and {ice, whiskey}
  - but, {ice, whiskey} is not a member of  $L_2$  and hence it is not frequent violating Apriori property. Thus we will have to remove {chips, ice, whiskey} from  $C_3$



# Association Rule Mining: Example

- step 4: generating 4-itemset frequent pattern
  - the algorithm uses  $L_3$  Join  $L_3$  to generate a candidate set of 4-itemsets,  $C_4$ . Although the join results in {coke, chips, ice, beer}, this itemset is pruned since its subset {chips, ice, whiskey} is not frequent  $C_4 = \text{empty} = \emptyset = \{\} \rightarrow C_4 \text{ ist leer (Null)}$
  - thus,  $C_4 = \emptyset$ , and the algorithm terminates, having found all of the frequent items. This completes the Apriori algorithm
  - these frequent itemsets will be used to generate strong association rules (where strong association rules satisfy both minimum support and minimum confidence)

# Association Rule Mining: Example

- step 5: generating association rules from frequent itemsets
  - for each frequent itemset  $I$ , generate all nonempty subsets of  $I$
  - for every nonempty subset  $s$  of  $I$ , output the rule " $s \rightarrow (I-s)$ " if  $\text{support\_count}(I) / \text{support\_count}(s) \geq \text{min\_conf}$
- we had  $L = \{\{\text{coke}\}, \{\text{chips}\}, \{\text{ice}\}, \{\text{beer}\}, \{\text{whiskey}\}, \{\text{coke, chips}\}, \{\text{coke, ice}\}, \{\text{coke, whiskey}\}, \{\text{chips, ice}\}, \{\text{chips, beer}\}, \{\text{chips, whiskey}\}, \{\text{coke, chips, ice}\}, \{\text{coke, chips, whiskey}\}\}$
- lets take  $I = \{\text{coke, chips, whiskey}\}$
- its all nonempty subsets are  $\{\text{coke, chips}\}, \{\text{coke, whiskey}\}, \{\text{chips, whiskey}\}, \{\text{coke}\}, \{\text{chips}\}, \{\text{whiskey}\}$

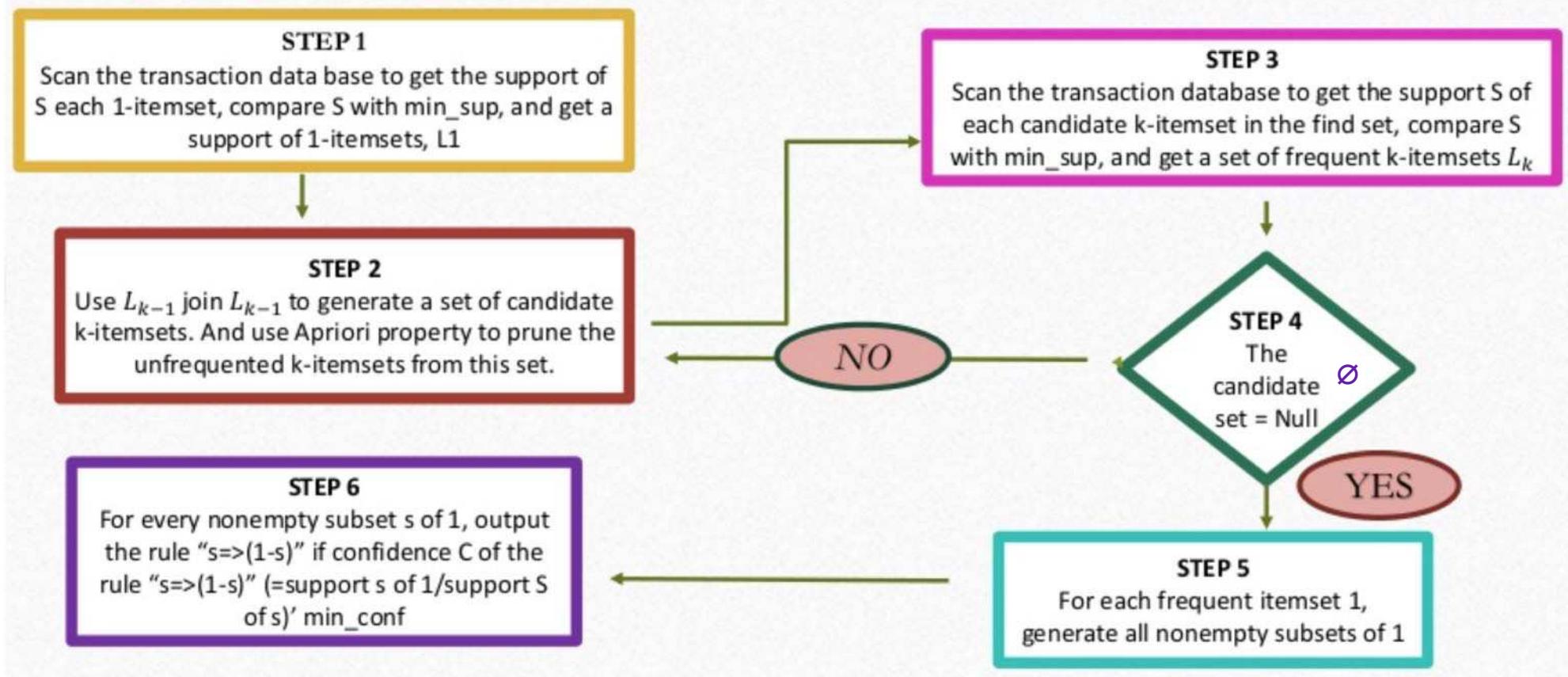
# Association Rule Mining: Example

- step 5: generating association rules from frequent itemsets
  - let minimum confidence threshold be 70%
  - the resulting association rules are as follows
    - rule1: {coke, chips} → {whiskey}
      - confidence =  $sc\{\text{coke, chips, whiskey}\} / sc\{\text{coke, chips}\} = 2/4 = 50\%$
      - rule1 is rejected
    - rule2: {coke, whiskey} → {chips}
      - confidence =  $sc\{\text{coke, chips, whiskey}\} / sc\{\text{coke, whiskey}\} = 2/2 = 100\%$
      - rule2 is selected
    - rule3: {chips, whiskey} → {coke}
      - confidence =  $sc\{\text{coke, chips, whiskey}\} / sc\{\text{chips, whiskey}\} = 2/2 = 100\%$
      - rule3 is selected

# Association Rule Mining: Example

- step 5: generating association rules from frequent itemsets
  - let minimum confidence threshold be 70%
  - the resulting association rules are as follows
    - rule4: {coke} → {chips, whiskey}
      - confidence =  $sc\{\text{coke, chips, whiskey}\} / sc\{\text{coke}\} = 2/6 = 33\%$
      - rule4 is rejected
    - rule5: {chips} → {coke, whiskey}
      - confidence =  $sc\{\text{coke, chips, whiskey}\} / sc\{\text{chips}\} = 2/7 = 29\%$
      - rule5 is rejected
    - rule6: {whiskey} → {coke, chips}
      - confidence =  $sc\{\text{coke, chips, whiskey}\} / sc\{\text{whiskey}\} = 2/2 = 100\%$
      - rule6 is selected

# Association Rule Mining: Summary



# Association Rule Mining: Summary

- challenges
    - multiple scans of transaction database is necessary
    - problematic with a huge amount of candidates
  - improving the apriori algorithm
    - reduce database scans
    - shrink number of candidates
- neues Thema ab hier? -----
- frequent pattern trees
    - an efficient and scalable method to generate sets of frequent patterns  
*das, worüber die vorherigen Folien waren*
    - allows frequent itemset discovery without candidate itemset generation
    - two step approach:
      - build a compact data structure called FP-Tree
      - extract frequent itemsets from the FP-Tree

# FP-Tree: Example

- step 1: calculate min support count

ID	A	B	C	D	E	List
Transaction Items						
T1	coke	chips			ice	{A, B, E}
T2		chips		beer		{B, D}
T3		chips	whiskey			{B, C}
T4	coke	chips		beer		{A, B, D}
T5	coke		whiskey			{A, C}
T6		chips	whiskey			{B, C}
T7	coke		whiskey			{A, C}
T8	coke	chips	whiskey		ice	{A, B, C, E}
T9	coke	chips	whiskey			{A, B, C}

$$\text{minsupport} = 22\%$$

$$\text{min support count} = (22/100 * 9) = 1.98 \rightarrow 2$$

# FP-Tree: Example

- step 2: find frequency of occurrence of each item

ID	Transaction Items				List
T1	coke	chips			ice {A, B, E}
T2		chips		beer	{B, D}
T3		chips	whiskey		{B, C}
T4	coke	chips		beer	{A, B, D}
T5	coke		whiskey		{A, C}
T6		chips	whiskey		{B, C}
T7	coke		whiskey		{A, C}
T8	coke	chips	whiskey		ice {A, B, C, E}
T9	coke	chips	whiskey		{A, B, C}

Item	Freq	Prio
coke A	6	2
chips B	7	1
whiskey C	6	3
beer D	2	4
ice E	2	5

Prio = Highest Freq → Lowest Prio

frequent item list:  
 $L = \{B:7, A:6, C:6, D:2, E:2\}$

# FP-Tree: Example

- step 3: order the items according to priority

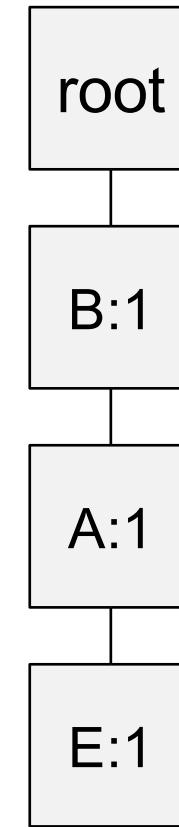
Geordnet nach  
Prio → B immer  
zuerst, dann A,  
C, D, E

A	B	C	D	E	
ID	Transaction Items				Ordered List
T1	coke	chips			ice {B, A, E}
T2		chips		beer	{B, D}
T3		chips	whiskey		{B, C}
T4	coke	chips		beer	{B, A, D}
T5	coke		whiskey		{A, C}
T6		chips	whiskey		{B, C}
T7	coke		whiskey		{A, C}
T8	coke	chips	whiskey		ice {B, A, C, E}
T9	coke	chips	whiskey		{B, A, C}

# FP-Tree: Example

- step 4: build the FP-Tree

Ordered Items
{B, A, E}
{B, D}
{B, C}
{B, A, D}
{A, C}
{B, C}
{A, C}
{B, A, C, E}
{B, A, C}

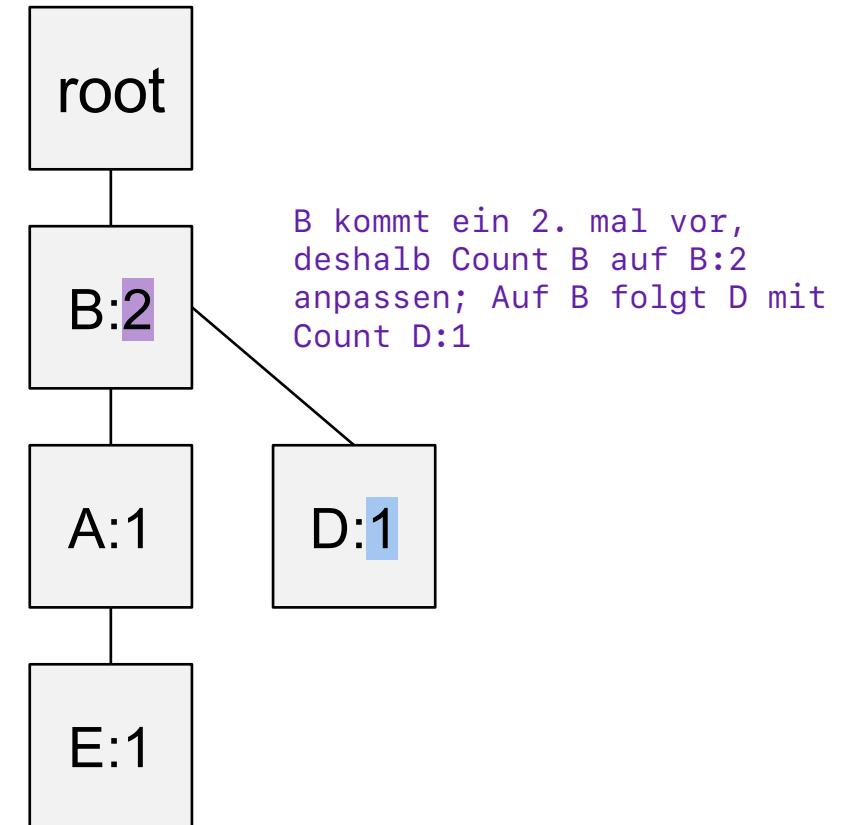


Alles kommt 1 mal vor im 1.  
Schritt → :1

# FP-Tree: Example

- step 4: build the FP-Tree

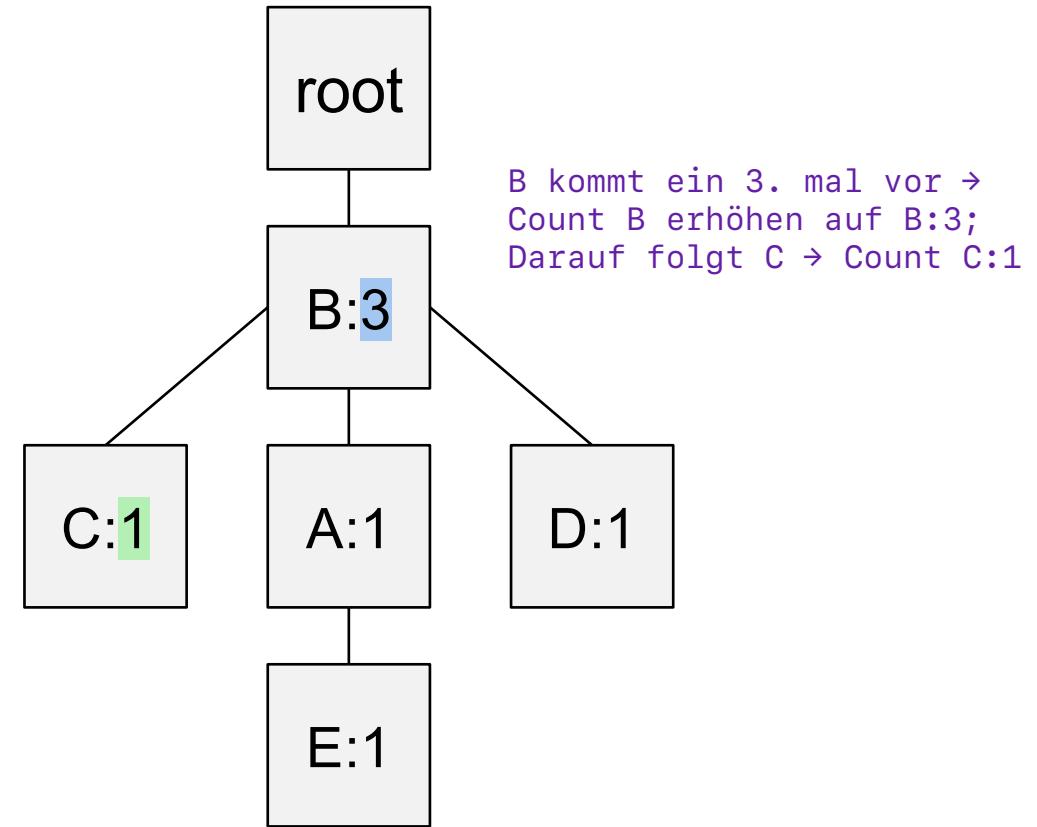
Ordered Items
{B, A, E}
{B, D}
{B, C}
{B, A, D}
{A, C}
{B, C}
{A, C}
{B, A, C, E}
{B, A, C}



# FP-Tree: Example

- step 4: build the FP-Tree

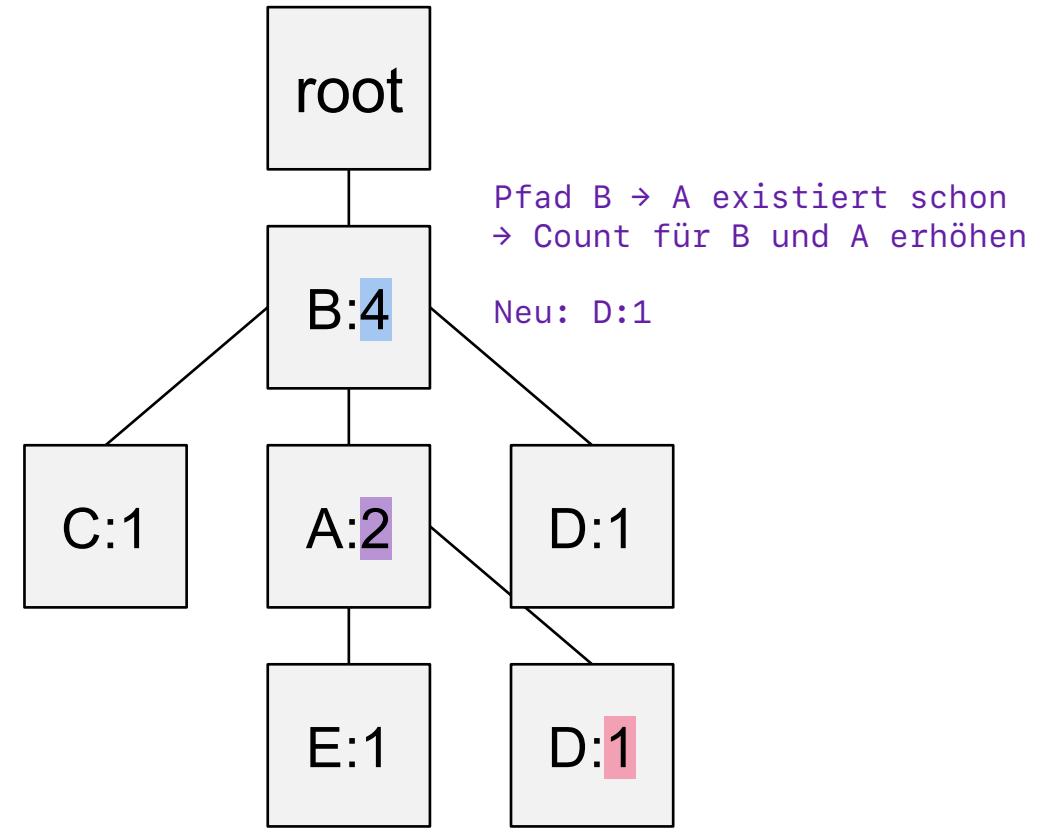
Ordered Items
{B, A, E}
{B, D}
{B, C}
{B, A, D}
{A, C}
{B, C}
{A, C}
{B, A, C, E}
{B, A, C}



# FP-Tree: Example

- step 4: build the FP-Tree

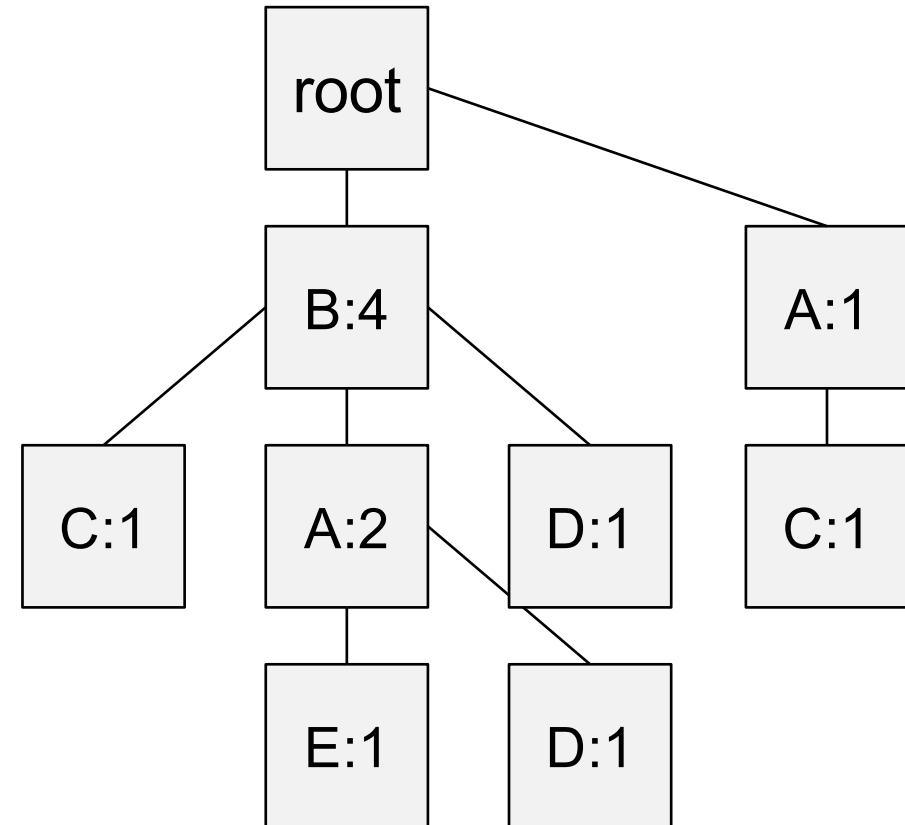
Ordered Items
{B, A, E}
{B, D}
{B, C}
{B, A, D}
{A, C}
{B, C}
{A, C}
{B, A, C, E}
{B, A, C}



# FP-Tree: Example

- step 4: build the FP-Tree

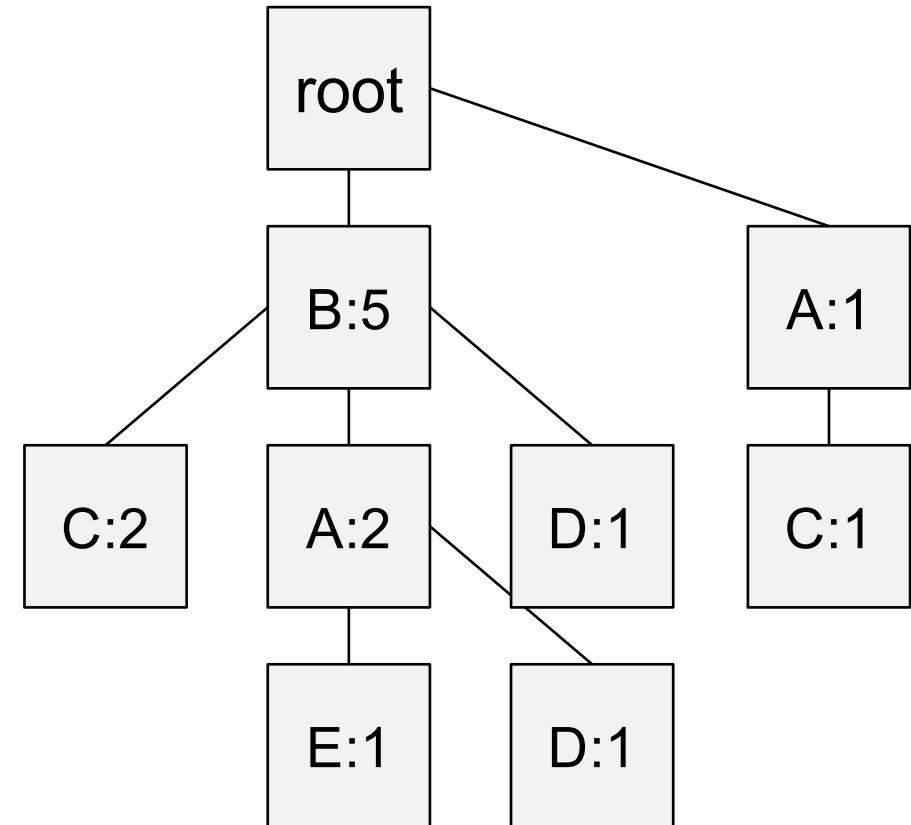
Ordered Items
{B, A, E}
{B, D}
{B, C}
{B, A, D}
<b>{A, C}</b>
{B, C}
{A, C}
{B, A, C, E}
{B, A, C}



# FP-Tree: Example

- step 4: build the FP-Tree

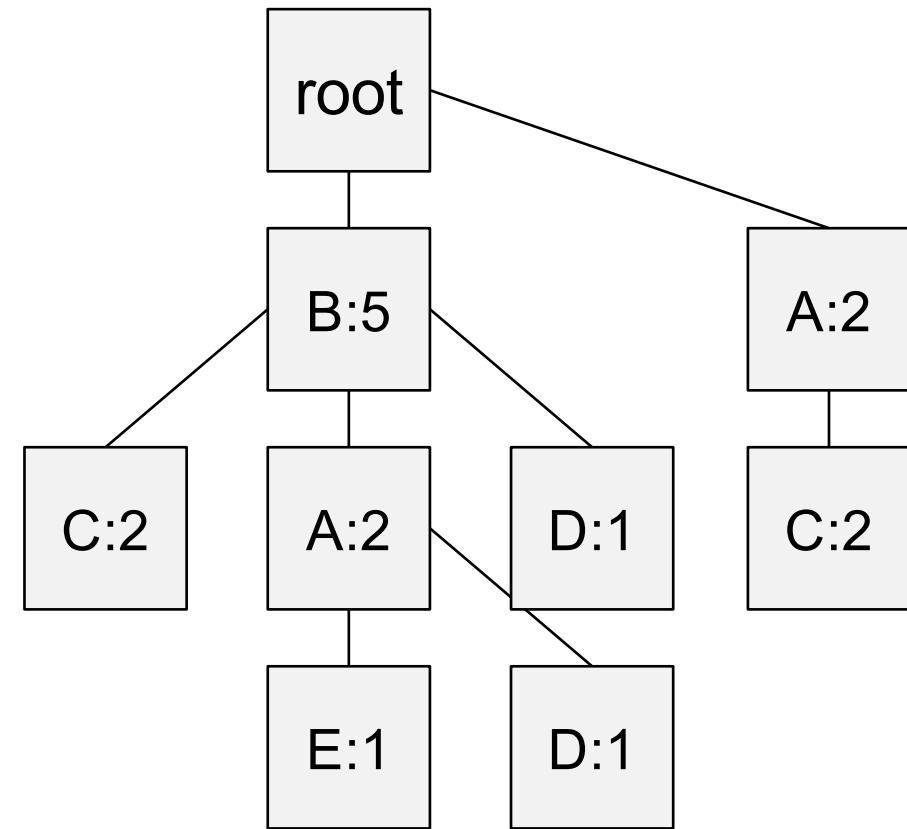
Ordered Items
{B, A, E}
{B, D}
{B, C}
{B, A, D}
{A, C}
{B, C}
{A, C}
{B, A, C, E}
{B, A, C}



# FP-Tree: Example

- step 4: build the FP-Tree

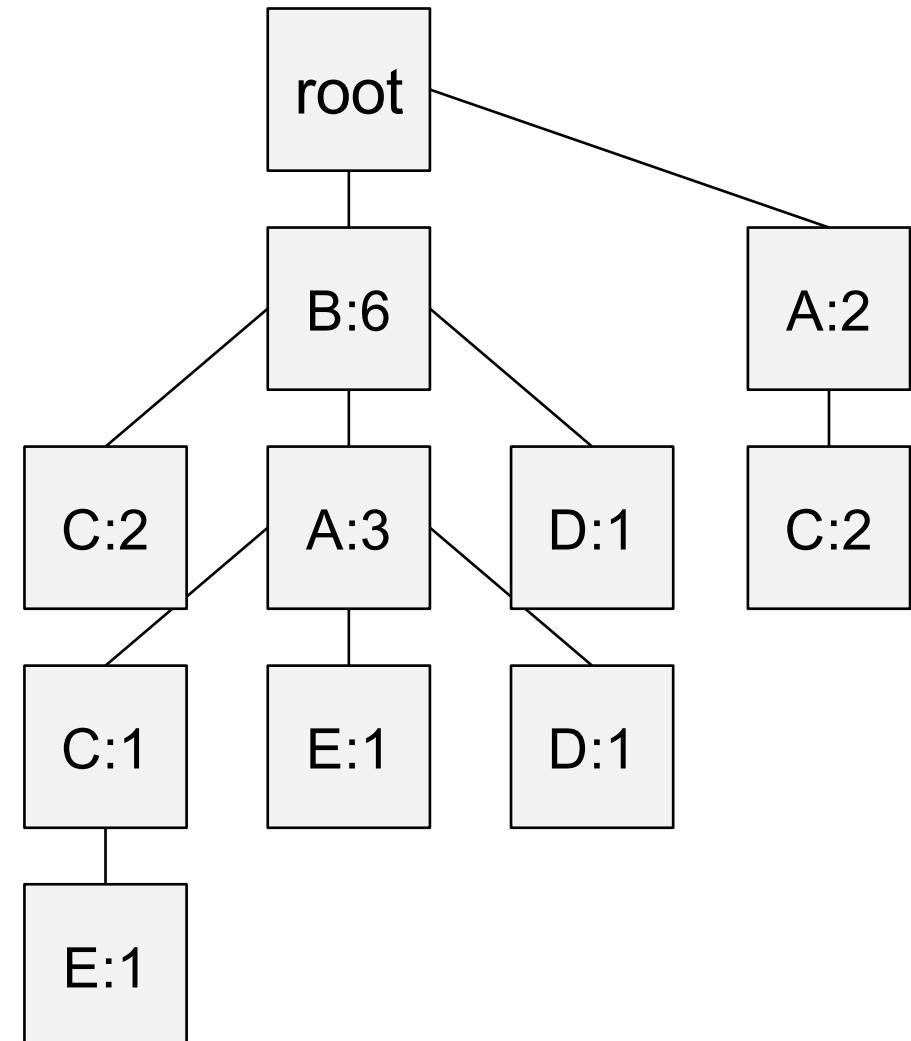
Ordered Items
{B, A, E}
{B, D}
{B, C}
{B, A, D}
{A, C}
{B, C}
<b>{A, C}</b>
{B, A, C, E}
{B, A, C}



# FP-Tree: Example

- step 4: build the FP-Tree

Ordered Items
{B, A, E}
{B, D}
{B, C}
{B, A, D}
{A, C}
{B, C}
{A, C}
{B, A, C, E}
{B, A, C}

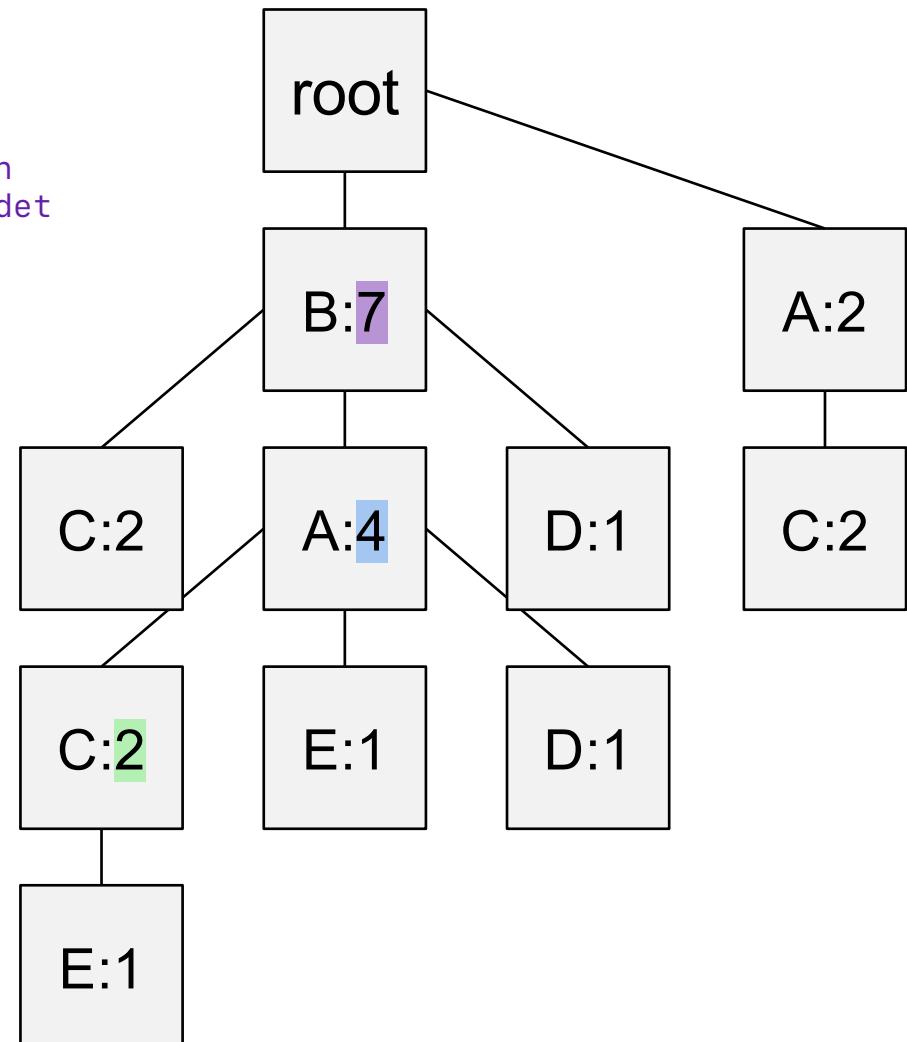


# FP-Tree: Example

- step 4: build the FP-Tree

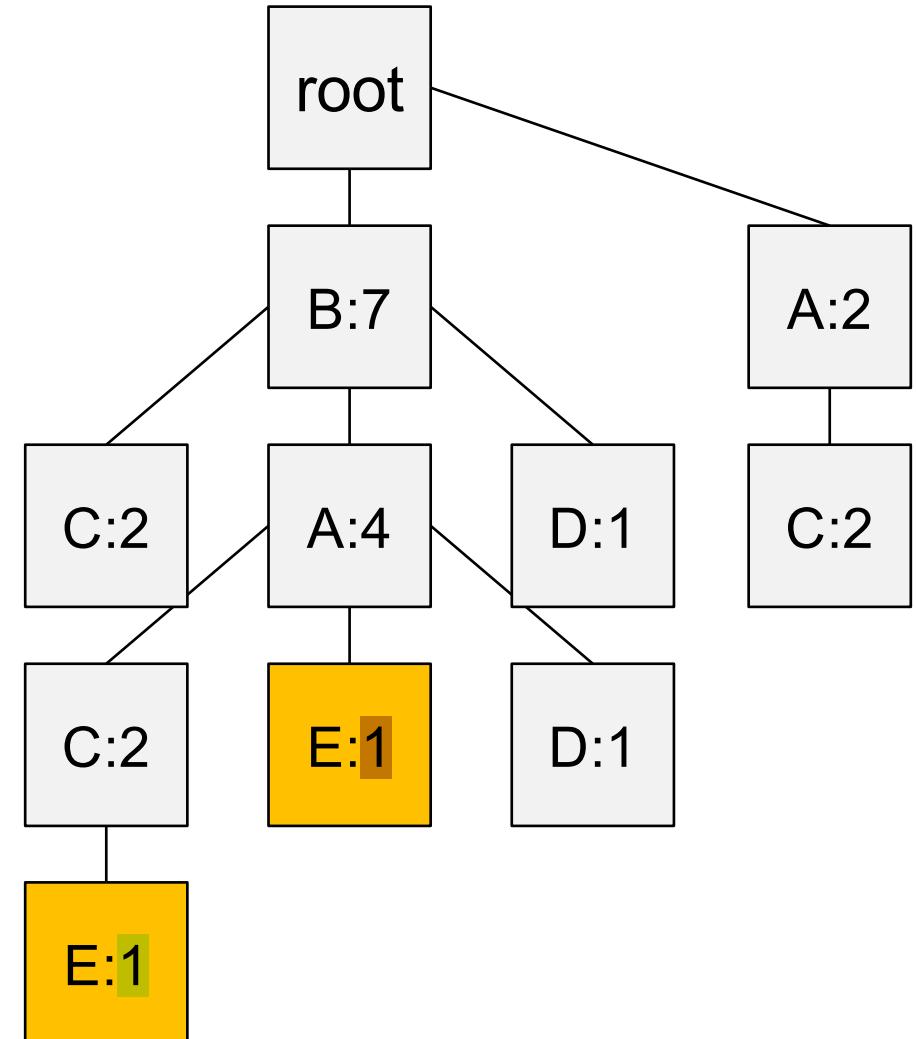
Ordered Items
{B, A, E}
{B, D}
{B, C}
{B, A, D}
{A, C}
{B, C}
{A, C}
{B, A, C, E}
<b>{B, A, C}</b>

alle Items in  
Baum abgebildet



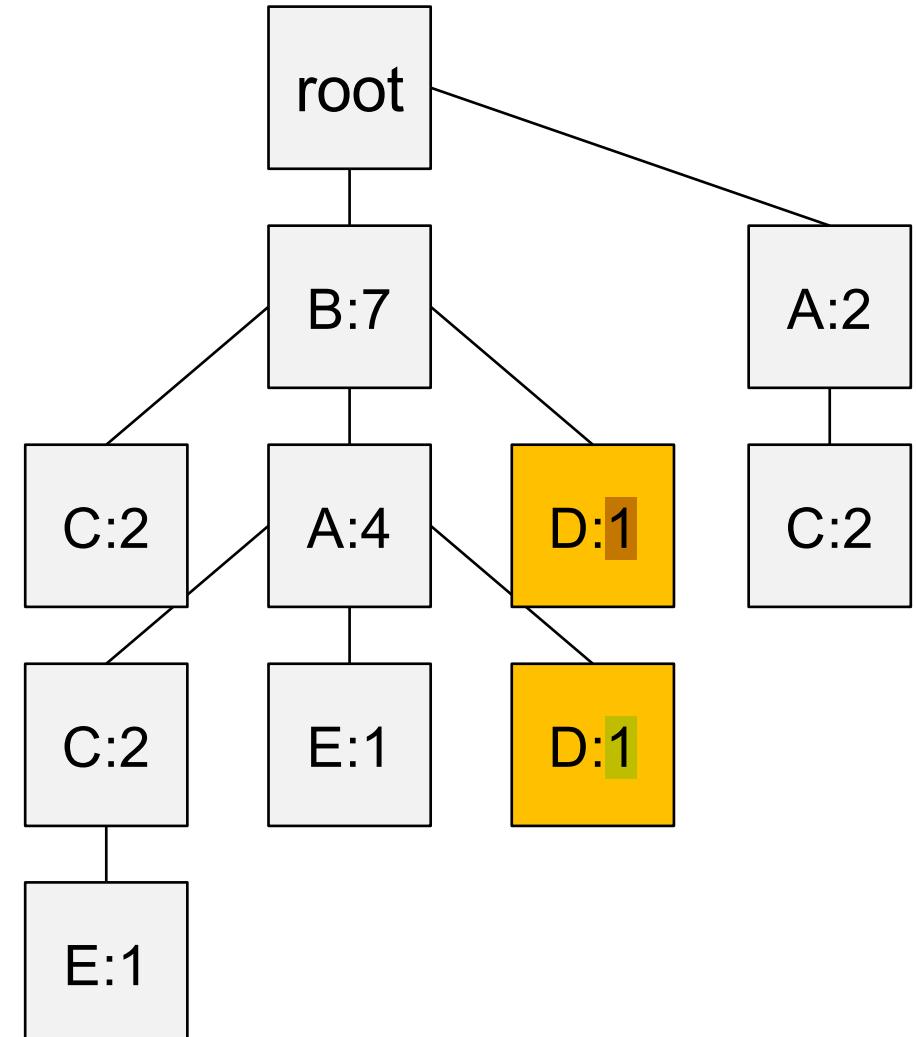
# FP-Tree: Example

- step 5: find conditional pattern base and conditional FP-Tree for each item *in this case: E*
  - conditional pattern base for E:  $\{\{B, A: 1\}, \{B, A, C: 1\}\}$
  - conditional FP-Tree for E: {B, A: 2} *Längster gemeinsamer Baum bis E: A → B (2 mal)*
  - frequent patterns generated: {B, E: 2}, {A, E: 2}, {B, A, E: 2} *Wie viel mal kann über Subsets E erreicht werden:  
über B: 2 mal  
über A → E: 2 mal  
über B → A: 2 mal*



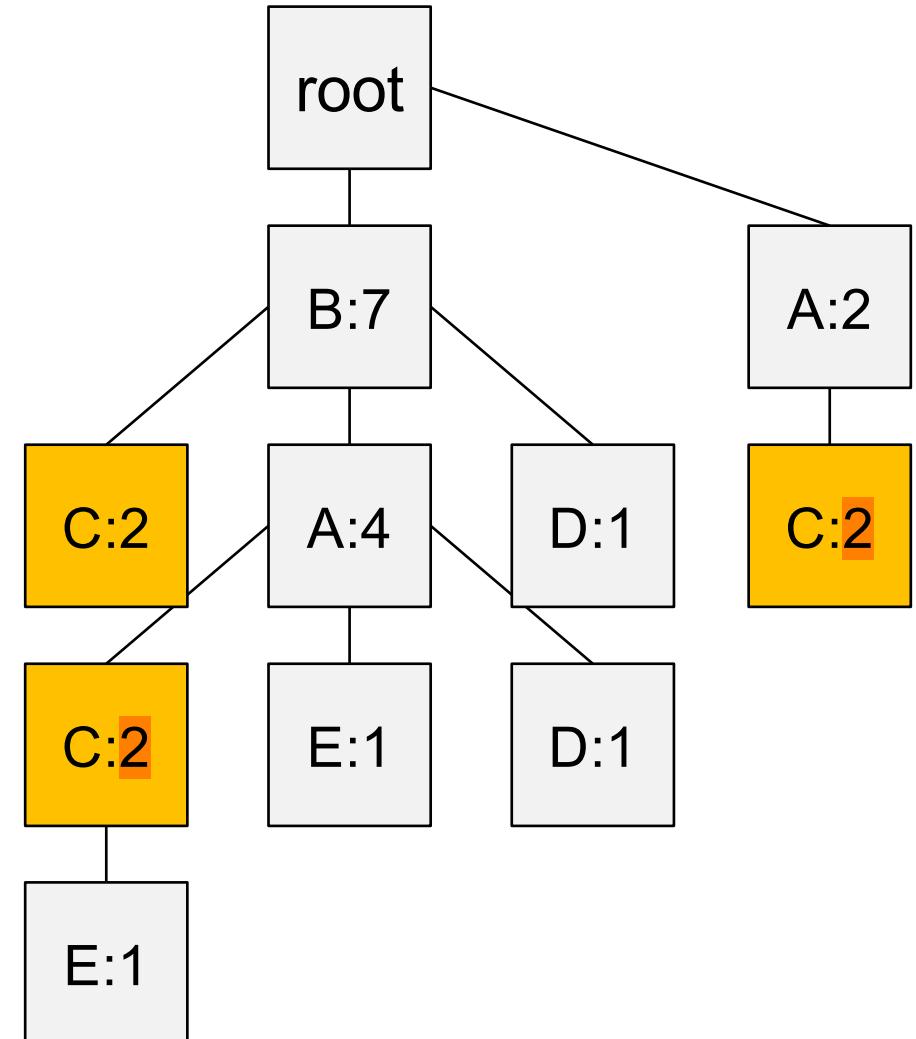
# FP-Tree: Example

- step 5: find conditional pattern base and conditional FP-Tree for each item
  - conditional pattern base for D: {{B: 1}, {B, A: 1}}
  - conditional FP-Tree for D: {B: 2} *Längster gemeinsamer Baum ist B*
  - frequent patterns generated: {B, D: 2}  
  
*Wie viel mal kann über Subsets D erreicht werden:  
über B: 2 mal  
über B → A: 1 mal → liegt unter min. Support Count → wird ignoriert.*



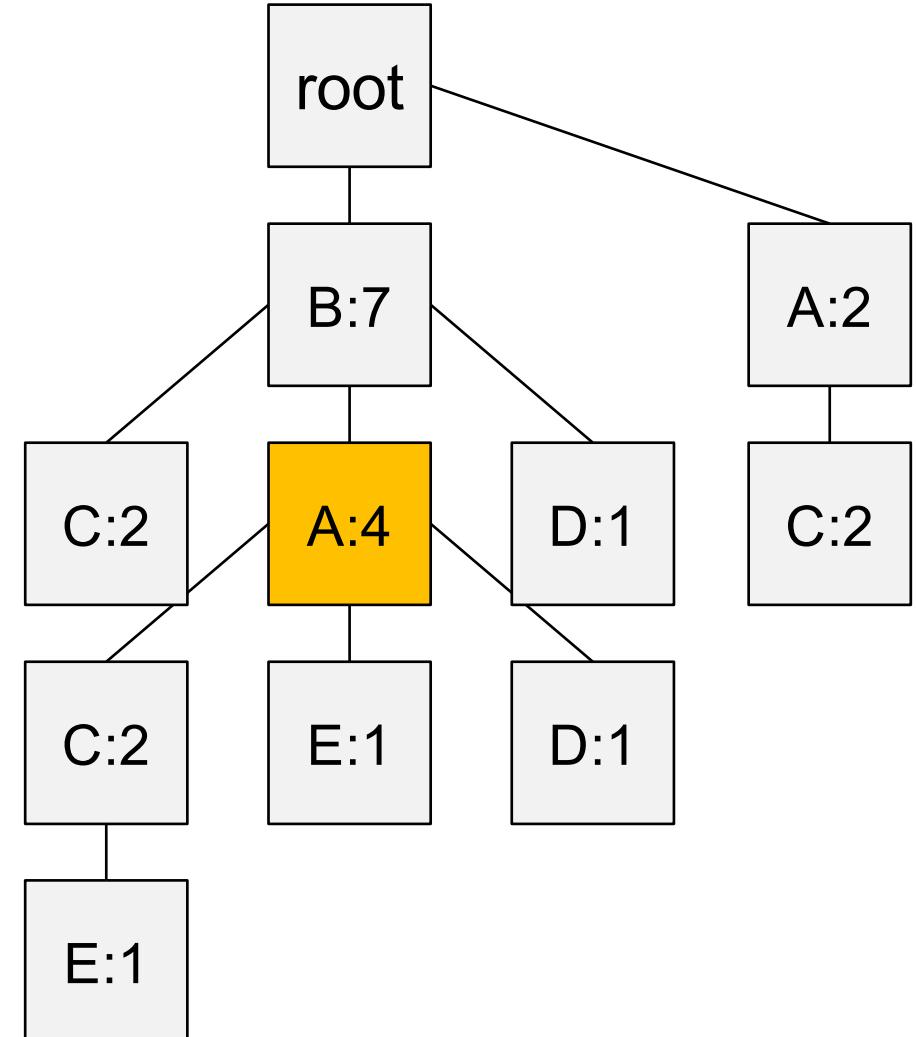
# FP-Tree: Example

- step 5: find conditional pattern base and conditional FP-Tree for each item
  - conditional pattern base for C:  $\{\{B, A: 2\}, \{B: 2\}, \{A: 2\}\}$
  - conditional FP-Tree for C:  $\{B: 4, A: 2\}, \{A: 2\}$  Es gibt mehrere Pfade von Root, deshalb 2 Sets
  - frequent patterns generated:  $\{B, C: 4\}, \{A, C: 4\}, \{B, A, C: 2\}$  C kann via A 4 mal erreicht werden  
→ 2 mal via Root → B & 2 mal via Root → A



# FP-Tree: Example

- step 5: find conditional pattern base and conditional FP-Tree for each item
  - conditional pattern base for A: {B: 4}
  - conditional FP-Tree for A: {B: 4}
  - frequent patterns generated: {B, A: 4}



# FP-Tree: Example

- step 6: find generated frequent patterns

- {chips, coke: 4} {B, A:4}
- {chips, whiskey: 4} {B, C:4}
- {coke, whiskey: 4} {A, C:4}
- {chips, coke, whiskey: 2} {B, A, C:2}
- {chips, beer: 2} {B, D:2}
- {chips, ice: 2} {B, E:2}
- {coke, ice: 2} {A, E:2}
- {chips, coke, ice: 2} {B, A, E:2}

ID	A	B	C	D	E	Transaction Items	Ordered List
T1		coke	chips			ice	{B, A, E}
T2			chips		beer		{B, D}
T3			chips	whiskey			{B, C}
T4	coke	chips			beer		{B, A, D}
T5	coke			whiskey			{A, C}
T6			chips	whiskey			{B, C}
T7	coke			whiskey			{A, C}
T8	coke	chips	whiskey			ice	{B, A, C, E}
T9	coke	chips	whiskey				{B, A, C}

# Lift measure

- measure for how likely item Y is purchased when item X is purchased, while controlling for how popular item Y is
  - $\text{lift } (x \rightarrow y) = \text{support}(x, y) / \text{support}(x) * \text{support}(y)$

Transaction	Support	Confidence	Lift
Canned Beer → Soda	1%	20%	1.0
Canned Beer → Berries	0.1%	1%	0.3
Canned Beer → Male Cosmetics	0.1%	1%	2.6

- lift value of 1 implies no association between beer and soda
- lift value below 1 implies that if someone buys berries, he would likely be averse to beer      *averse = abgeneigt sein*
- high lift value of 2.6 implies whenever someone does buy male cosmetics, he is very likely to buy beer as well

# Lessons Learned

- understanding of frequent itemset mining
- understanding of mining various kinds of association rules



# Questions and Answers



# Machine Learning and Data Mining

## V07: Linear and Logistic Regression

Lecturer: Dr. Andreas Weiler



# Repetition

... REPETITION ...

→ IS THE MOTHER →

of learning

# Course Content IT20a\_WIN

Week	Lecture	Lab
KW08	Introduction	Lab 01
KW09	Data Processing Part 01	Lab 02
KW10	Carnival Monday	
KW11	Data Processing Part 02	Lab 03
KW12	Evaluation	Lab 04
KW13	Recommender Systems	<b>Graded Lab 01</b> <span style="border: 1px solid red; padding: 2px;">IMPORTANT</span>
KW14	Association Rules	<b>Graded Lab 01</b> <span style="border: 1px solid red; padding: 2px;">IMPORTANT</span>
KW15	Linear and Logistic Regression	Lab 05
KW16	Easter Monday	
KW17	Support Vector Machines	<b>Graded Lab 02</b> <span style="border: 1px solid red; padding: 2px;">IMPORTANT</span>
KW18	Decision Trees and Naïve Bayes	<b>Graded Lab 02</b> <span style="border: 1px solid red; padding: 2px;">IMPORTANT</span>
KW19	Partition Clustering	Lab 06
KW20	Hierarchical and Density Clustering	Lab 07
KW21	Neural Networks and Closing	Review and Questions for Exam

# Learning Objectives

- fundamental understanding of the gradient descent method for linear regression
- fundamental understanding of logistic regression



# Linear Regression

- linear approach to model the relationship between a scalar response and one or more explanatory variables
  - one explanatory variable is called simple linear regression
  - for more than one, the process is called multivariate linear regression

car					
horsepower	150	200	250	300	350
price	\$150K	\$200K	???	\$300K	\$350K

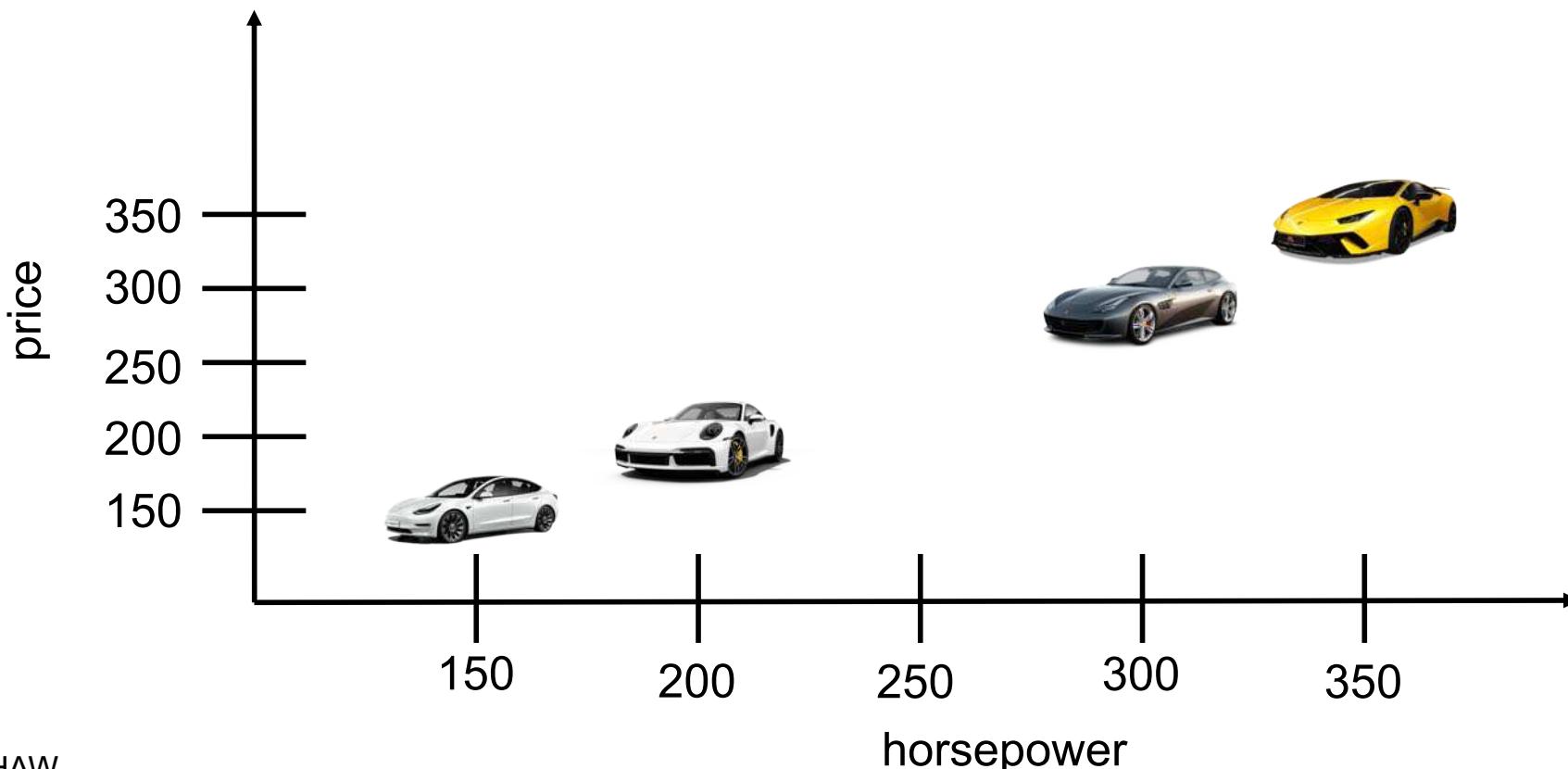
# Linear Regression

- linear approach to model the relationship between a scalar response and one or more explanatory variables
  - one explanatory variable is called simple linear regression
  - for more than one, the process is called multivariate linear regression

car						
horsepower	150	200	250	300	350	400
price	\$150K	\$200K	\$250K	\$300K	\$350K	\$400K
type	female	male	female	male	female	???

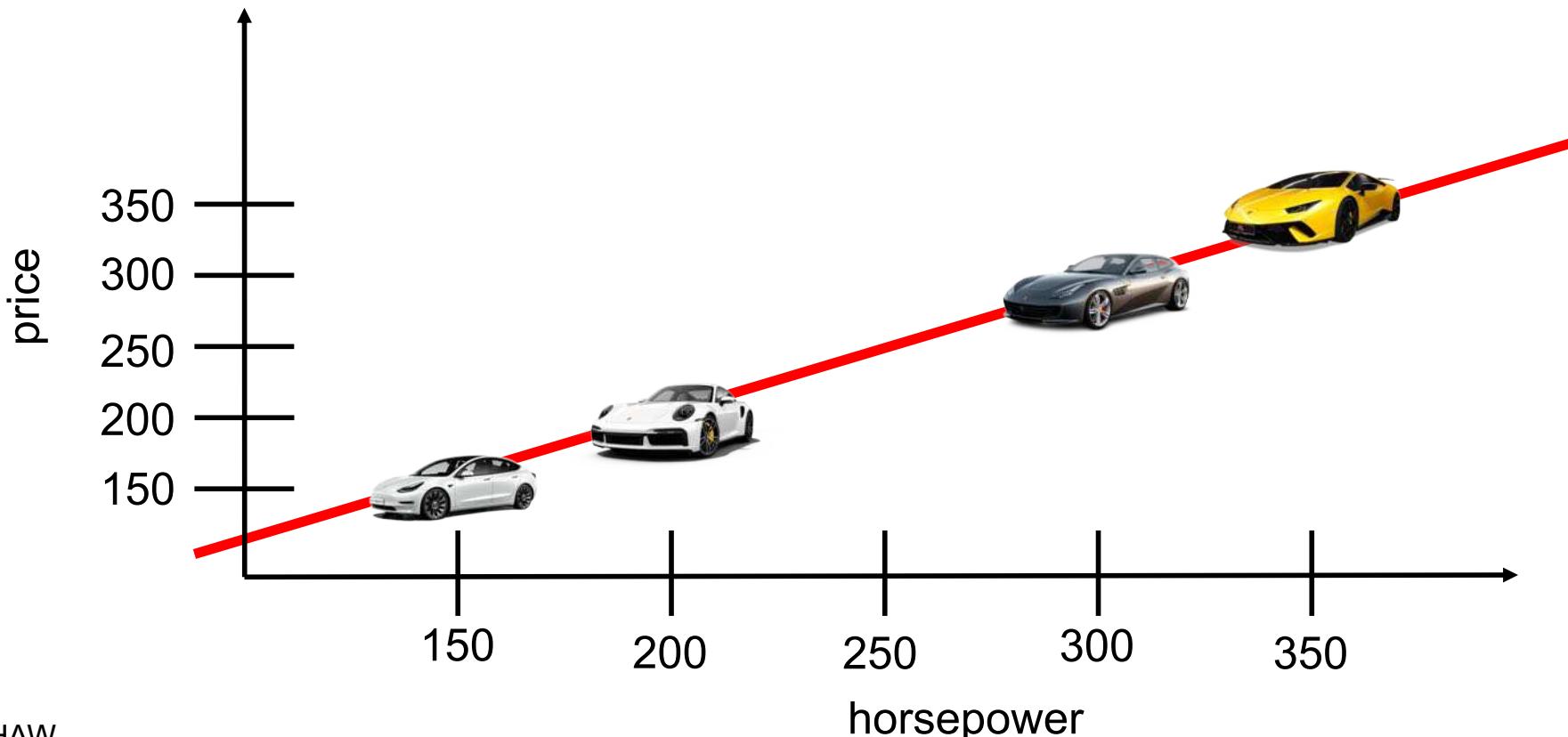
# Linear Regression

- linear approach to model the relationship between a scalar response and one or more explanatory variables



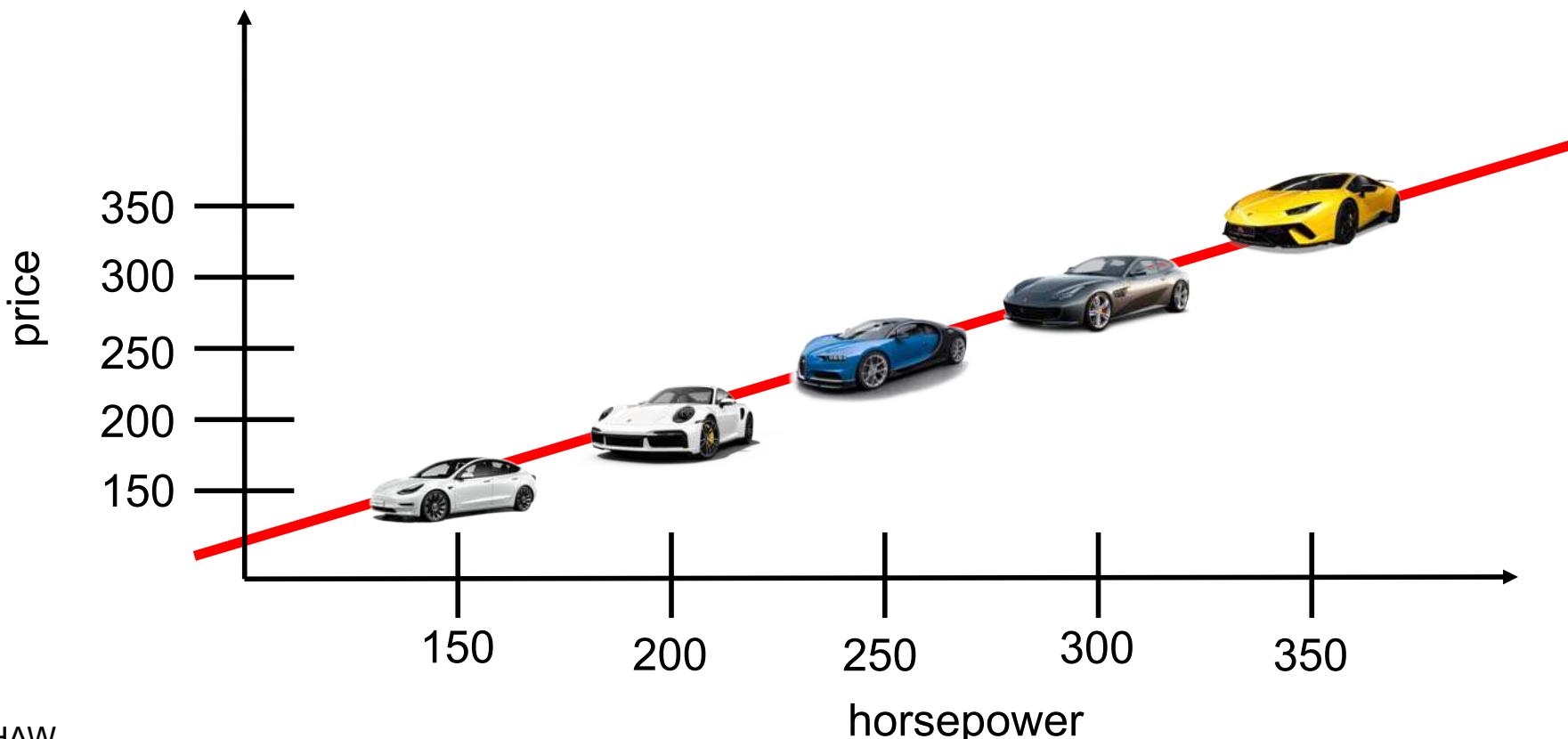
# Linear Regression

- linear approach to model the relationship between a scalar response and one or more explanatory variables



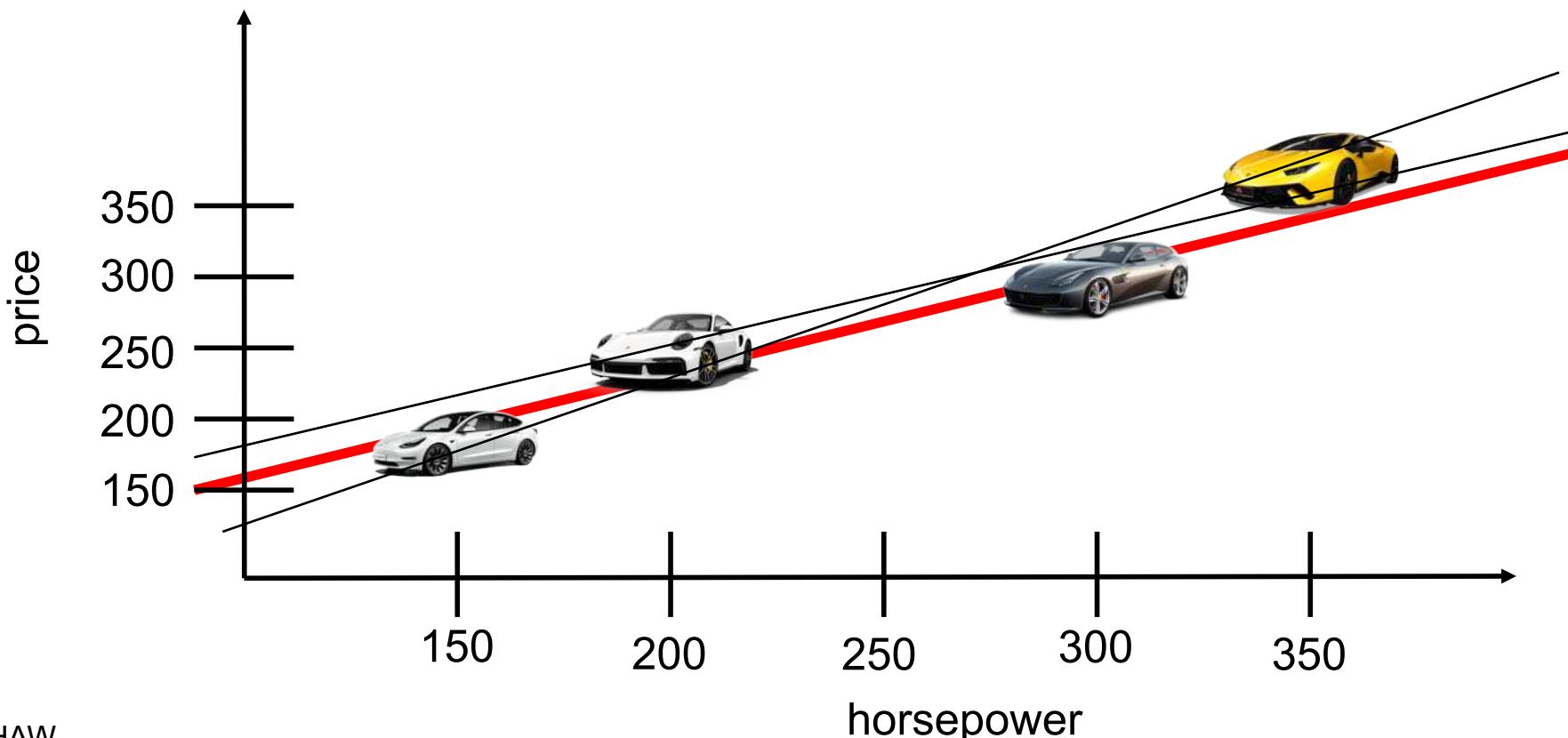
# Linear Regression

- linear approach to model the relationship between a scalar response and one or more explanatory variables



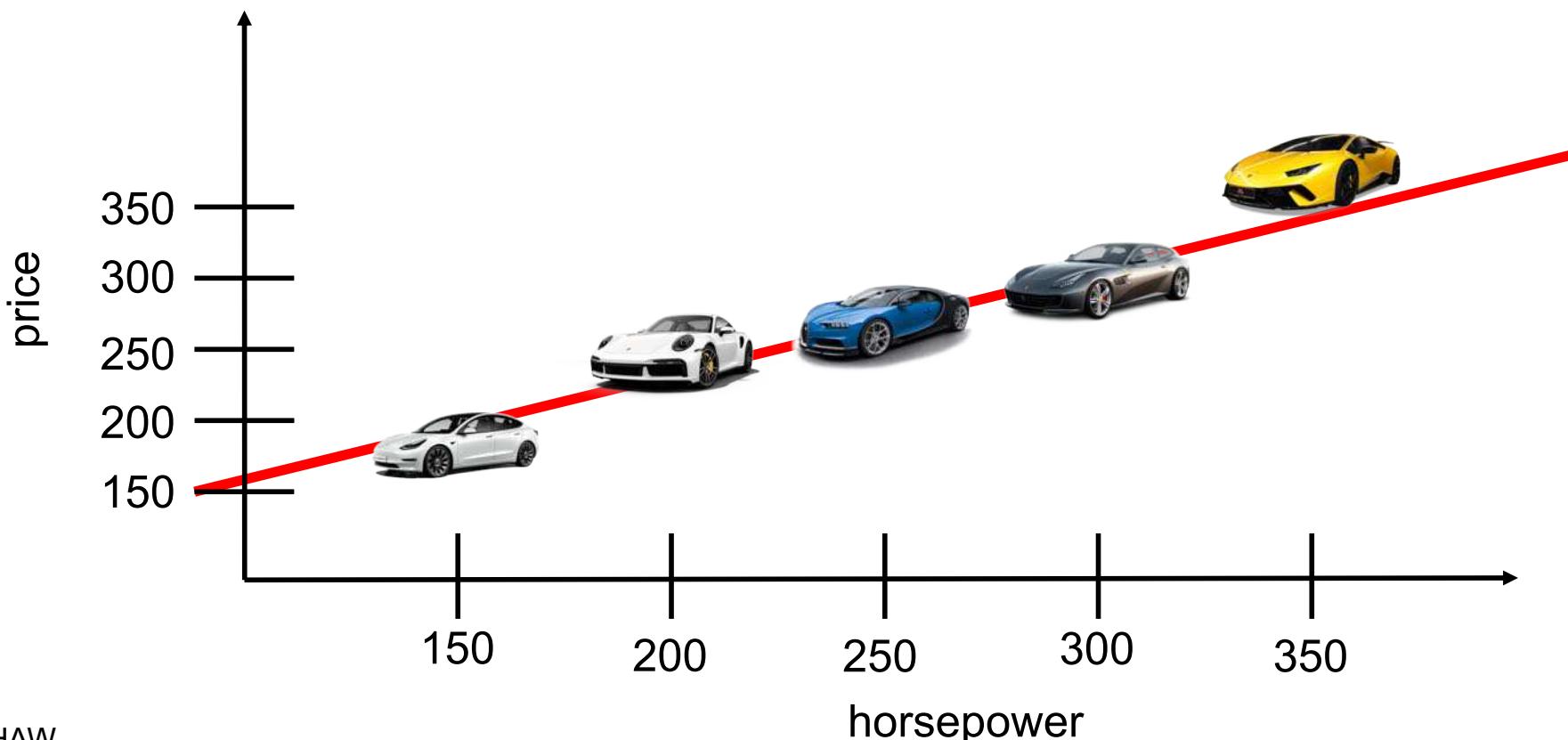
# Linear Regression

- linear approach to model the relationship between a scalar response and one or more explanatory variables



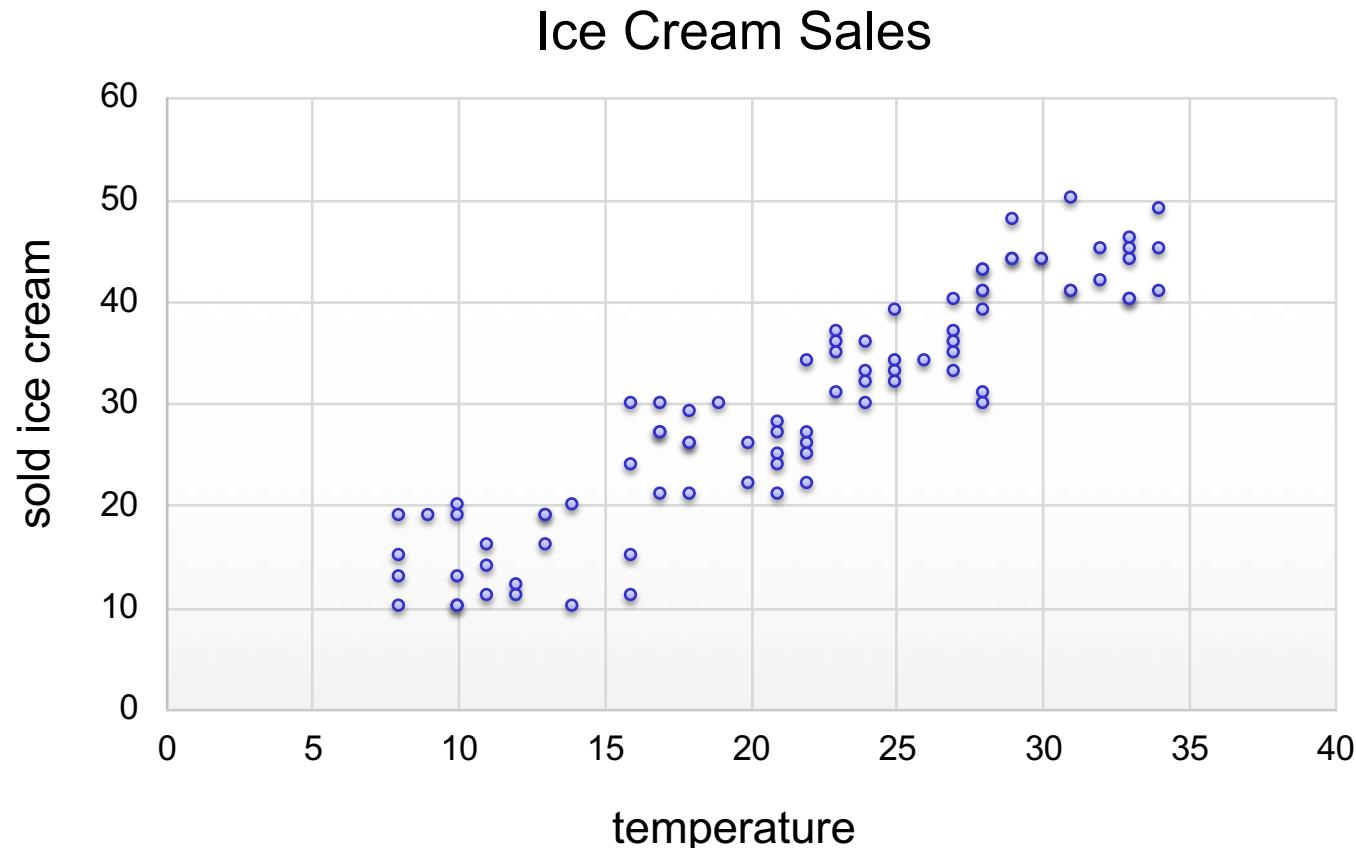
# Linear Regression

- linear approach to model the relationship between a scalar response and one or more explanatory variables



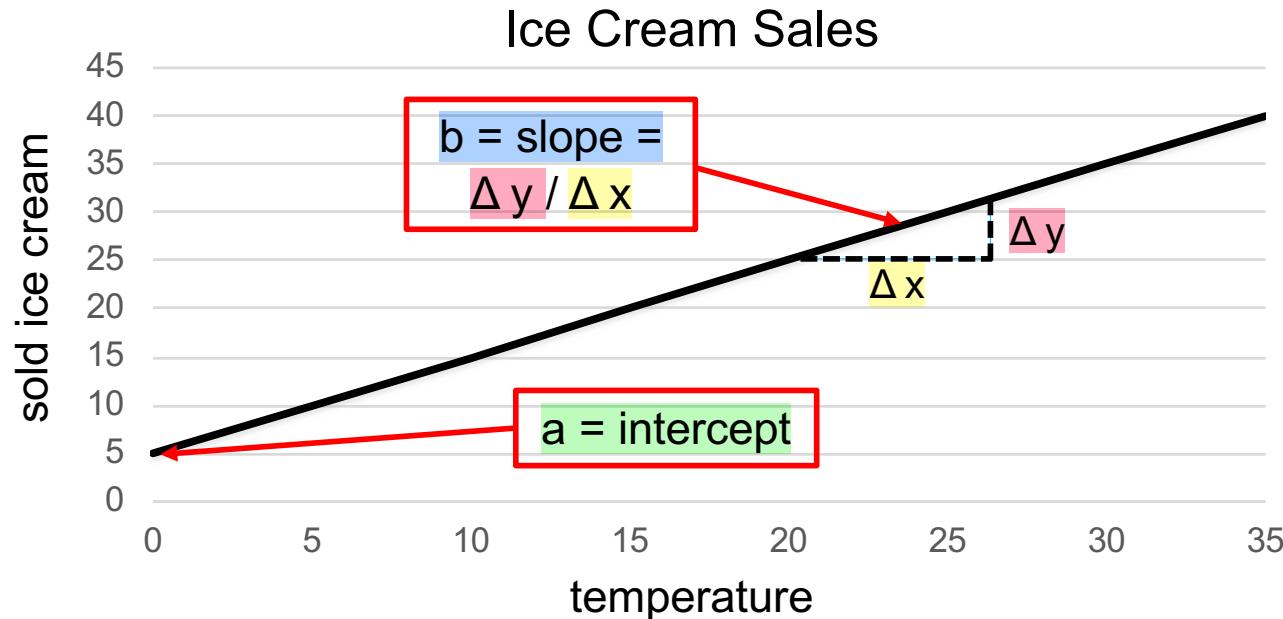
# Linear Regression

- how to find the best fitting line?



# Linear Regression

- ordinary least squares method
  - expression describing a straight line:  $y = a + bx$   
( $y$  = outcome;  $x$  = input)
  - $b$ : indicates the slope of the line
  - $a$ : indicates the intersection with the y-axis



# Linear Regression

- ordinary least squares method
  - calculation of the slope

$$b = \frac{\sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^n (x_i - \bar{x})^2}$$

- $\bar{x}$  is the mean value of  $x$
- $\bar{y}$  is the mean value of  $y$
- find  $a$  when  $b$  is known:  $a = \bar{y} - b\bar{x}$

# Linear Regression

- ordinary least squares method
  - example

<b>temperature</b>	x	10	15	17	23	25	29	31	32
<b>sold ice cream</b>	y	15	22	28	34	38	44	48	55

- calculation of the slope with  $\bar{x} = 22.75$  and  $\bar{y} = 30.75$
- $b = \frac{(10 - 22.75) * (15 - 30.75) + (15 - 22.75) * (22 - 30.75) + \dots}{(10 - 22.75)^2 + (15 - 22.75)^2 + \dots}$
- $b = 1.6560088202866594$

# Linear Regression

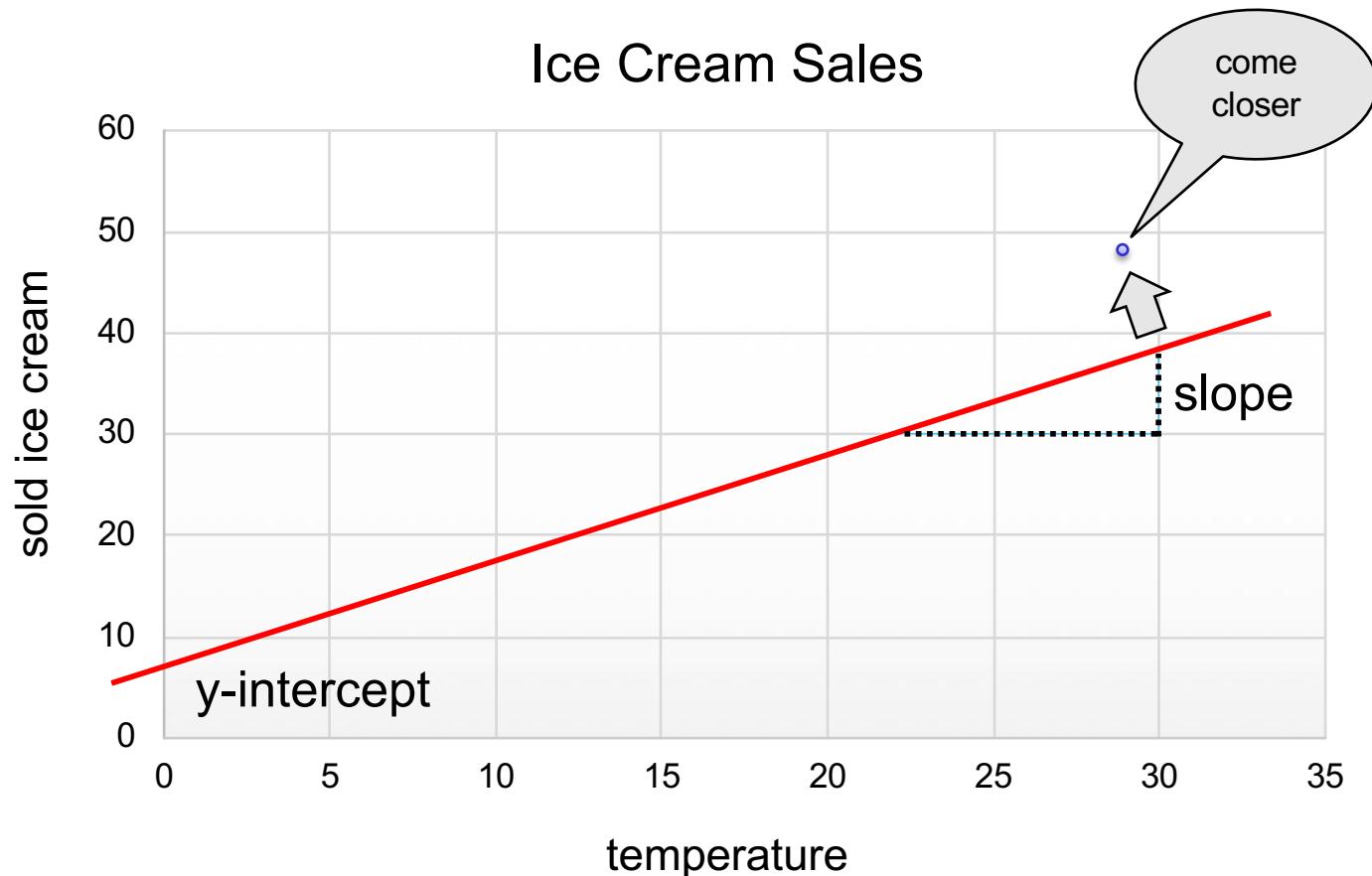
- ordinary least squares method
  - example

<b>temperature</b>	x	10	15	17	23	25	29	31	32
<b>sold ice cream</b>	y	15	22	28	34	38	44	48	55

- find a when b is known:
  - $a = 30.75 - 1.656 * 22.75$
  - $a = -2.1742006615215033$
- predicting a "sold ice cream" value for 35 degrees
  - $y = a + bx$
  - $y = -2.1742006615215033 + (1.6560088202866594 * 35)$
  - $y = 55.786$

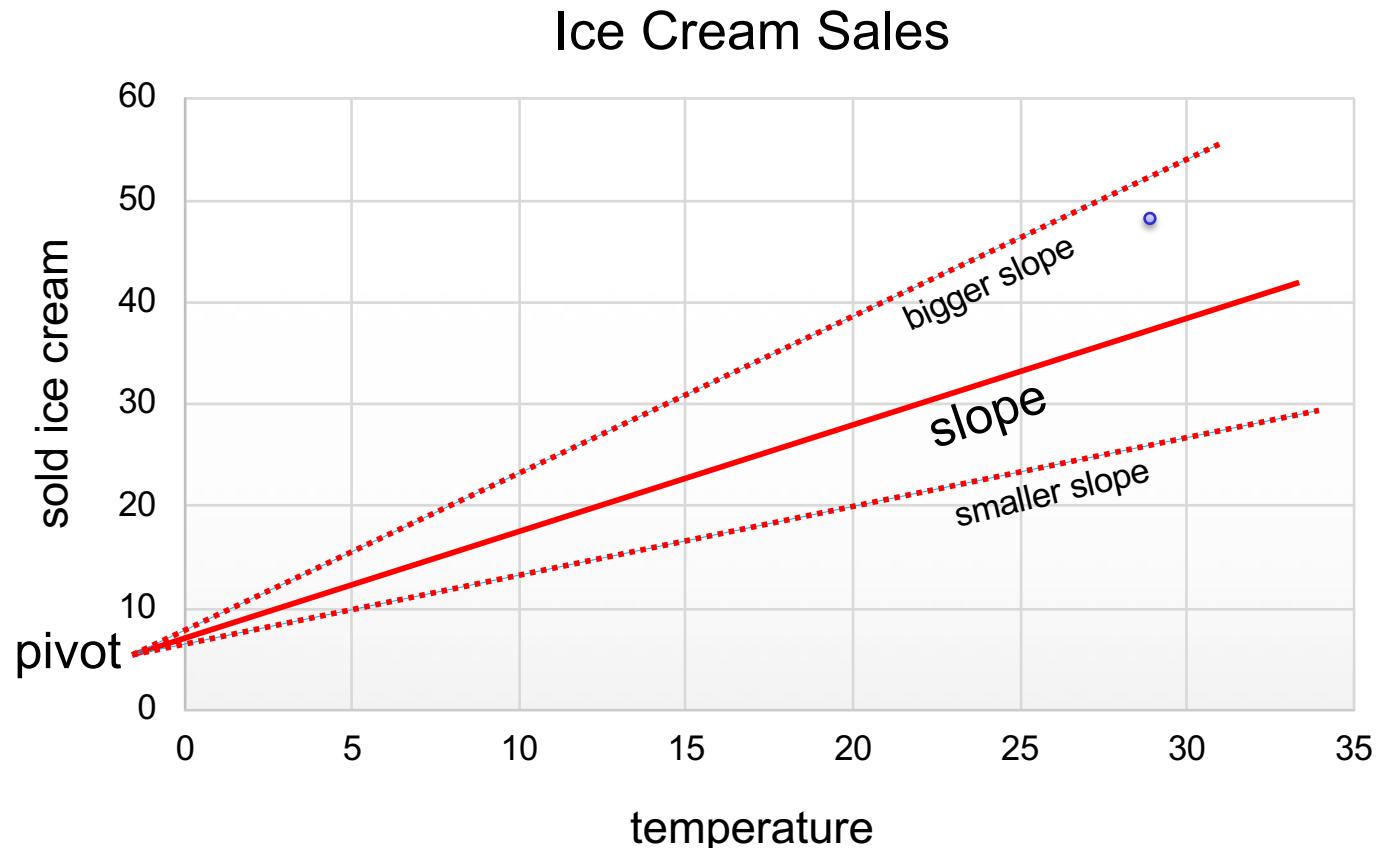
# Linear Regression

- gradient descent method



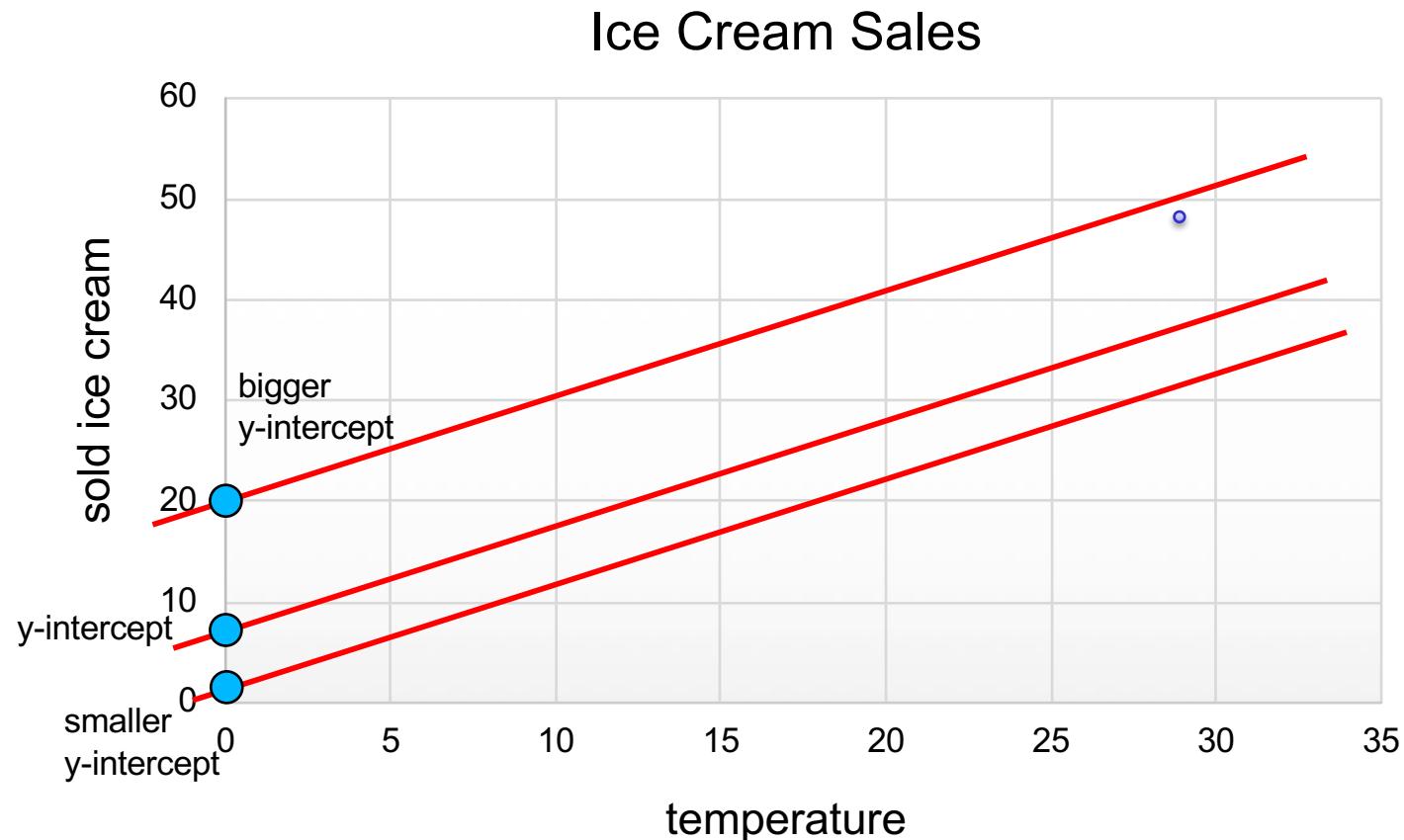
# Linear Regression

- gradient descent method
  - adjust the slope with rotation



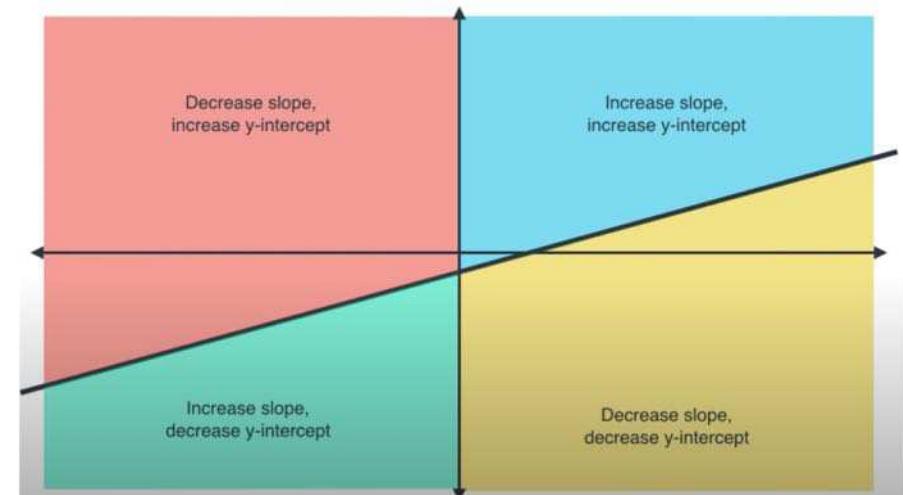
# Linear Regression

- gradient descent method
  - adjust the y-intercept with translation



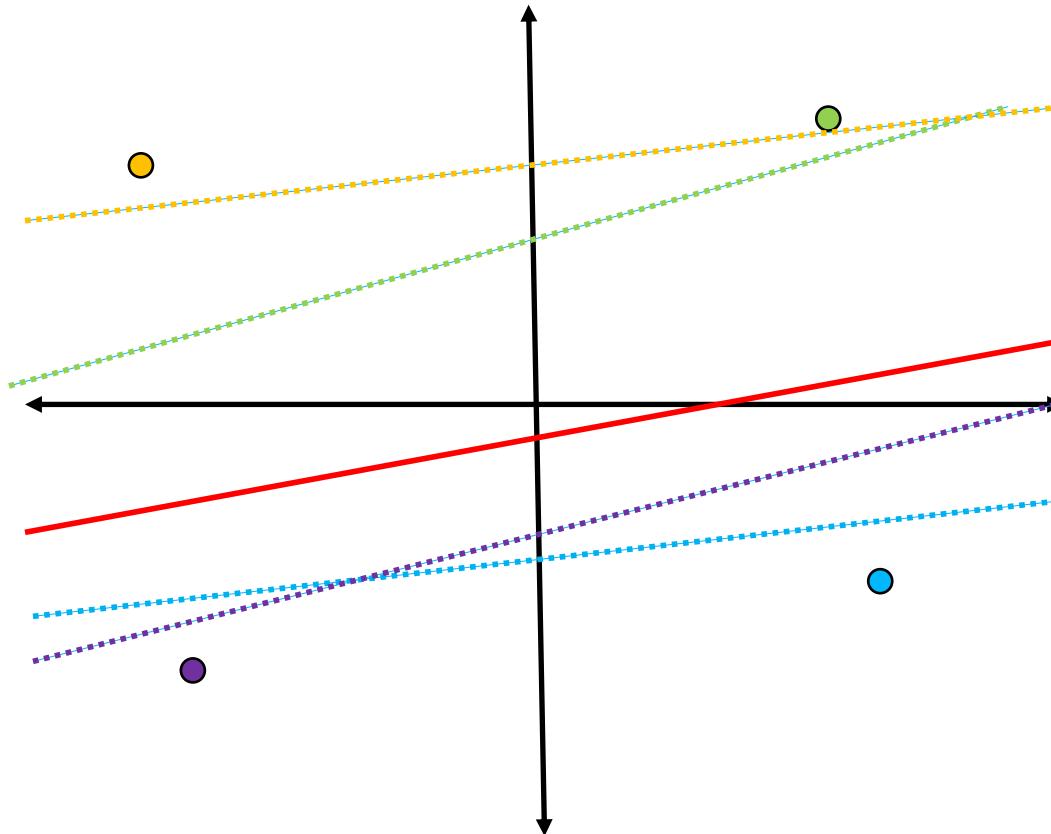
# Linear Regression

- gradient descent method
  - four cases of moving a line
    - rotate counter-clockwise and translate up → increase slope and increase the y-intercept
    - rotate clockwise and translate up → decrease the slope and increase the y-intercept
    - rotate counter-clockwise and translate down → increase the slope and decrease the y-intercept
    - rotate clockwise and translate down → decrease the slope and decrease the y-intercept



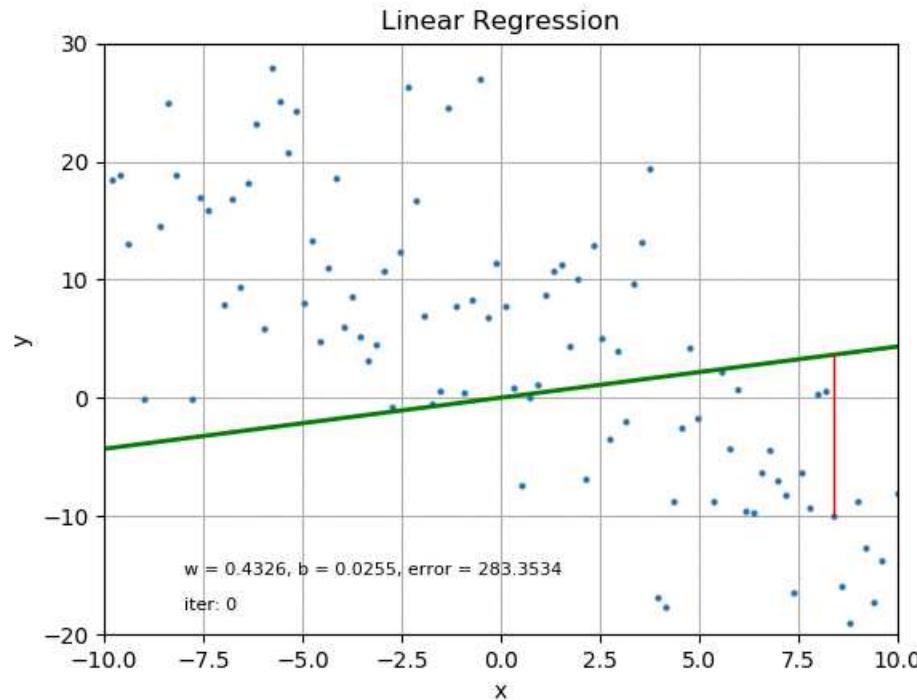
# Linear Regression

- gradient descent method



# Linear Regression

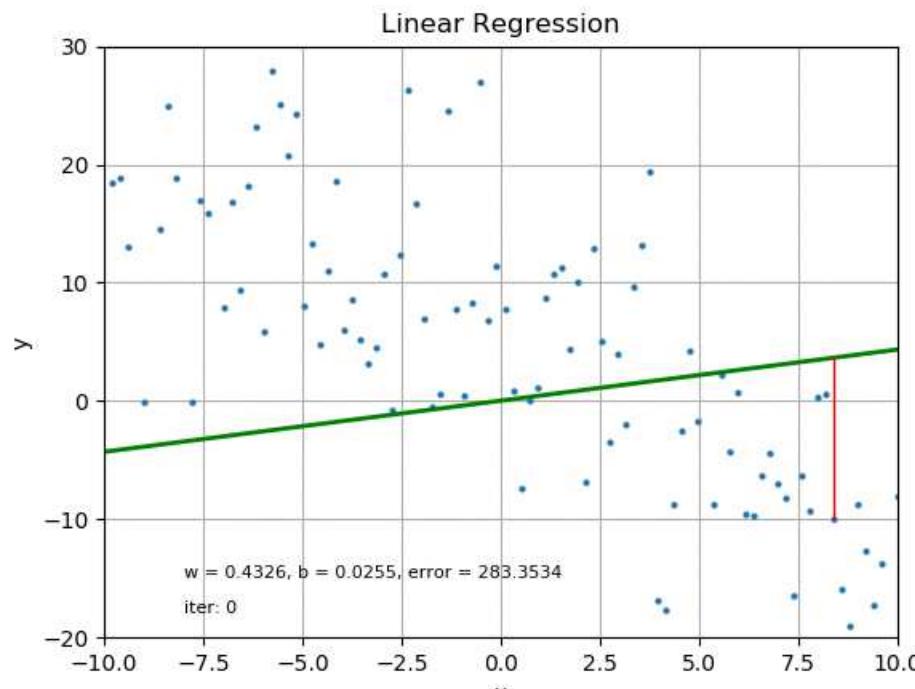
- gradient descent method



- step 1: start with a random line (e.g.,  $y = 2x + 3$ )
- step 2: pick a large number for repetitions or epochs (e.g., 1000)
- step 3: pick a small number for the learning rate  $lr$  (e.g., 0.01)
- step 4: pick random point (repeat epoch times)
  - if point above line and to the right of the y-axis, add  $lr$  to slope and add  $lr$  to y-intercept
  - if point above line and to the left of the y-axis, subtract  $lr$  to slope and add  $lr$  to y-intercept
  - if point below line and to the right of the y-axis, subtract  $lr$  to slope and subtract  $lr$  to the y-intercept
  - if point below line and to the left of the y-axis, add  $lr$  to the slope and subtract  $lr$  to the y-intercept

# Linear Regression

- gradient descent method: optimization

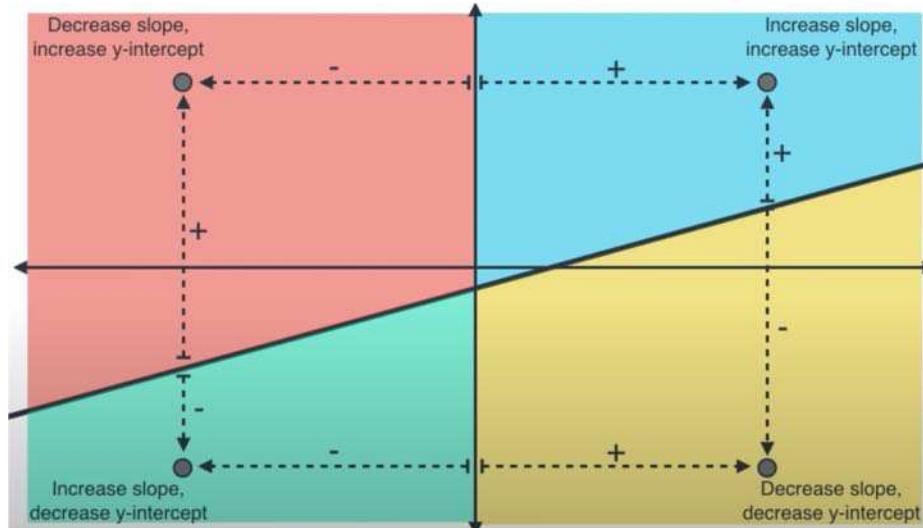


4 cases! → 1 case?

- step 1: start with a random line (e.g.,  $y = 2x + 3$ )
- step 2: pick a large number for repetitions or epochs (e.g., 1000)
- step 3: pick a small number for the learning rate  $lr$  (e.g., 0.01)
- step 4: pick random point (repeat epoch times)
  - if point above line and to the right of the y-axis, add  $lr$  to slope and add  $lr$  to y-intercept
  - if point above line and to the left of the y-axis, subtract  $lr$  to slope and add  $lr$  to y-intercept
  - if point below line and to the right of the y-axis, subtract  $lr$  to slope and subtract  $lr$  to the y-intercept
  - if point below line and to the left of the y-axis, add  $lr$  to the slope and subtract  $lr$  to the y-intercept

# Linear Regression

- gradient descent method: optimization



- step 1: start with a **random line** (e.g.,  $y = 2x + 3$ )
- step 2: pick a **large number** for **repetitions or epochs** (e.g., 1000)
- step 3: pick a **small number** for the **learning rate** (e.g., 0.01)
- step 4: pick **random point** (repeat epoch times)
  - add (**learning rate**) \* (vertical distance) \* (horizontal distance) to slope
  - add (**learning rate**) \* (vertical distance) to y-intercept

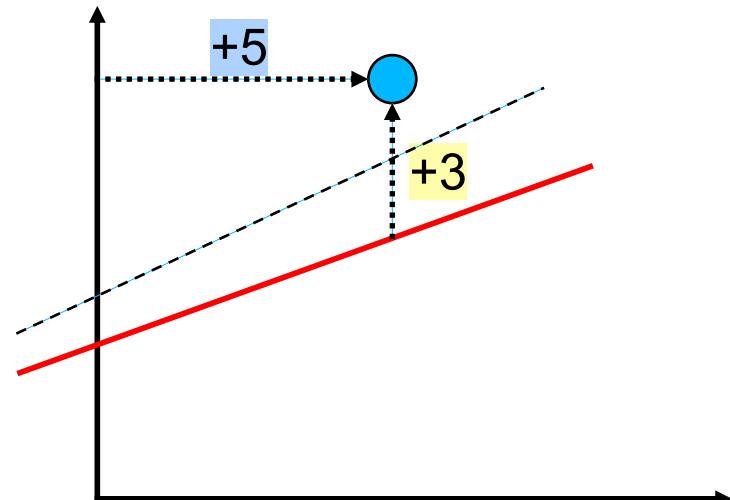
# Linear Regression

- gradient descent method: optimization

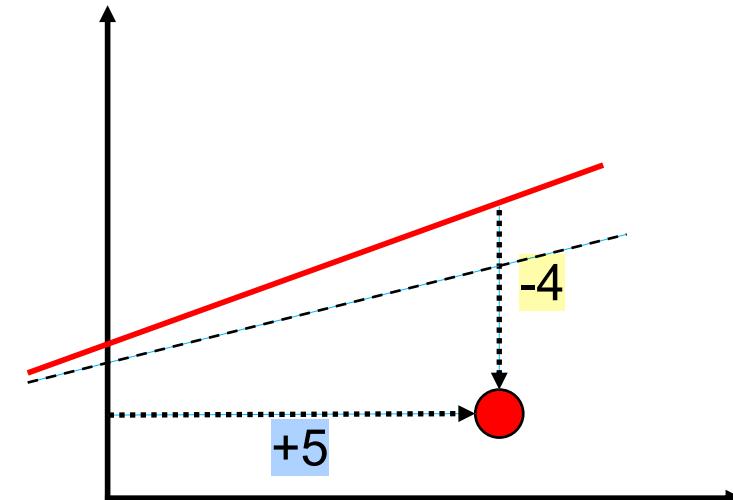
- example:

- $y = 2x + 3$

- learning rate = 0.01



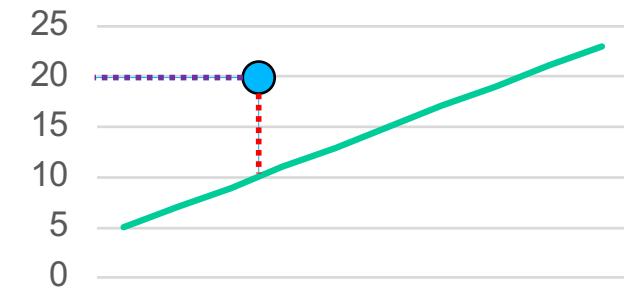
- add  $0.01 * 3 * 5 = 0.15$  to slope
      - add  $0.01 * 3 = 0.03$  to y-intercept
      - $y = 2.15x + 3.03$



- add  $0.01 * (-4) * 5 = -0.2$  to slope
      - add  $0.01 * (-4) = -0.04$  to y-intercept
      - $y = 1.8x + 2.96$

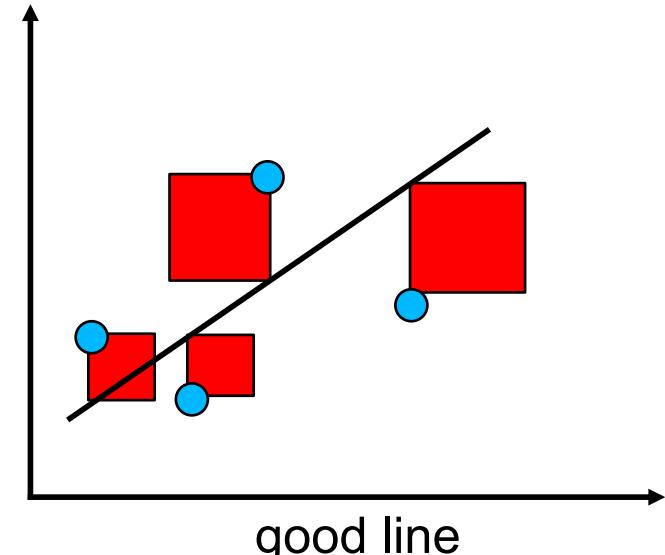
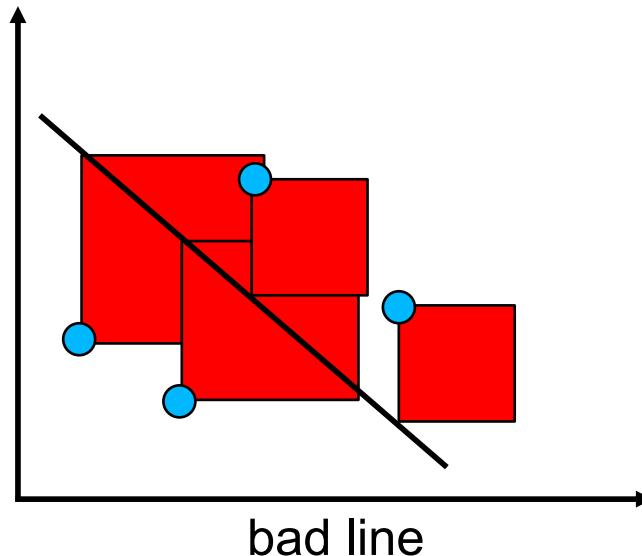
# Linear Regression

- gradient descent method: optimization
  - example calculating distance:
    - $y = 2x + 3$  (slope = 2; yintercept = 3)
    - random data point 1:  $cx = 5$ ;  $cy = 20$
    - vertical distance:
      - $vd = 20 - (2 * 5) + 3$
      - $vd = 7$
    - horizontal distance:
      - $hd = 5$
    - new slope =  $0.01 * 7 * 5 + 2 = 2.35$
    - new yintercept =  $0.01 * 7 + 3 = 3.07$
    - $y = 2.35x + 3.07$



# Linear Regression

- mean squared error (MSE)
  - step 1: find the regression line
  - step 2: insert your X values into the linear regression equation to find the new Y values ( $\hat{Y}$ )
  - step 3: subtract the new  $\hat{Y}$  value from the original Y to get the error
  - step 4: square the errors, add up the errors, calculate the mean



# Linear Regression

- gradient descent method: mean squared error (MSE)
  - step 1: find the regression line
  - step 2: insert your X values into the linear regression equation to find the new Y values ( $\hat{Y}$ )
  - step 3: subtract the new  $\hat{Y}$  value from the original Y to get the error
  - step 4: square the errors, add up the errors, calculate the mean

X	Y	$\hat{Y}_1'$	$e_1$	$e_1^2$	$\hat{Y}_2'$	$e_2$	$e_2^2$
43	41	43.6	-2.6	6.76	41.35	0.35	0.12
44	45	44.4	0.6	0.36	42.1	-2.9	8.41
45	49	45.2	3.8	14.44	42.85	-6.15	37.8
46	47	46	1	1	43.6	-3.4	11.56
47	44	46.8	-2.8	7.84	44.35	0.35	0.12

- regression line 1
  - $y = 9.2 + 0.8x$
  - MSE = 6.08
- regression line 2
  - $y = 9.1 + 0.75x$
  - MSE = 11.61

# Multivariate Linear Regression

- linear regression with multiple features
  - find regression equation:  $\hat{y} = b_0 + b_1x_1 + b_2x_2$
  - example: find regression equation to predict test score, based on IQ and the number of hours that the person studied

person	score	IQ	study hours
1	100	110	40
2	90	120	30
3	80	100	20
4	70	90	0
5	60	80	10

- $\hat{y}$  is the predicted test score;  $b_0$ ,  $b_1$ , and  $b_2$  are regression coefficients;  $x_1$  is an IQ score; and  $x_2$  is the number of hours that the person studied

# Multivariate Linear Regression

- linear regression with multiple features
  - to define the regression coefficients, we use the following equation:

$$\mathbf{b} = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{Y}$$

- so we need to complete the following steps: define  $\mathbf{X}$ , define  $\mathbf{X}'$ , compute  $\mathbf{X}'\mathbf{X}$ , find the inverse of  $\mathbf{X}'\mathbf{X}$ , define  $\mathbf{Y}$

$$\mathbf{X} = \begin{bmatrix} 1 & 110 & 40 \\ 1 & 120 & 30 \\ 1 & 100 & 20 \\ 1 & 90 & 0 \\ 1 & 80 & 10 \end{bmatrix} \quad \mathbf{X}' = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 110 & 120 & 100 & 90 & 80 \\ 40 & 30 & 20 & 0 & 10 \end{bmatrix}$$

- compute  $\mathbf{X}'\mathbf{X} \rightarrow$  find the inverse of  $\mathbf{X}'\mathbf{X}$

$$\mathbf{X}'\mathbf{X} = \begin{bmatrix} 5 & 500 & 100 \\ 500 & 51,000 & 10,800 \\ 100 & 10,800 & 3,000 \end{bmatrix} \quad (\mathbf{X}'\mathbf{X})^{-1} = \begin{bmatrix} 101/5 & -7/30 & 1/6 \\ -7/30 & 1/360 & -1/450 \\ 1/6 & -1/450 & 1/360 \end{bmatrix}$$

# Multivariate Linear Regression

- linear regression with multiple features
  - to define the regression coefficients, we use the following equation:

$$\mathbf{b} = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{Y} \quad \mathbf{X}' = \mathbf{X} \cdot \mathbf{T} \text{ (transformiert)}$$

- so we need to complete the following steps: define  $\mathbf{X}$ , define  $\mathbf{X}'$

$$\mathbf{X} = \begin{bmatrix} 1 & 110 & 40 \\ 1 & 120 & 30 \\ 1 & 100 & 20 \\ 1 & 90 & 0 \\ 1 & 80 & 10 \end{bmatrix} \quad \mathbf{X}' = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 110 & 120 & 100 & 90 & 80 \\ 40 & 30 & 20 & 0 & 10 \end{bmatrix}$$

- compute  $\mathbf{X}'\mathbf{X}$ , find the inverse of  $\mathbf{X}'\mathbf{X}$ , and define  $\mathbf{Y}$

$$\mathbf{X}'\mathbf{X} = \begin{bmatrix} 5 & 500 & 100 \\ 500 & 51,000 & 10,800 \\ 100 & 10,800 & 3,000 \end{bmatrix} \quad (\mathbf{X}'\mathbf{X})^{-1} = \begin{bmatrix} 101/5 & -7/30 & 1/6 \\ -7/30 & 1/360 & -1/450 \\ 1/6 & -1/450 & 1/360 \end{bmatrix} \quad \mathbf{Y} = \begin{bmatrix} 100 \\ 90 \\ 80 \\ 70 \\ 60 \end{bmatrix}$$

# Multivariate Linear Regression

- linear regression with multiple features
  - with all of the essential matrices defined, we are ready to compute the least squares regression coefficients  $\mathbf{b} = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{Y}$

$$\mathbf{X}' = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 110 & 120 & 100 & 90 & 80 \\ 40 & 30 & 20 & 0 & 10 \end{bmatrix} * \mathbf{Y} = \begin{bmatrix} 100 \\ 90 \\ 80 \\ 70 \\ 60 \end{bmatrix} = \begin{bmatrix} 400 \\ 40900 \\ 8900 \end{bmatrix}$$

$$(\mathbf{X}'\mathbf{X})^{-1} = \begin{bmatrix} 101/5 & -7/30 & 1/6 \\ -7/30 & 1/360 & -1/450 \\ 1/6 & -1/450 & 1/360 \end{bmatrix} * \begin{bmatrix} 400 \\ 40900 \\ 8900 \end{bmatrix} = \begin{bmatrix} 20 \\ 0.5 \\ 0.5 \end{bmatrix}$$

↓

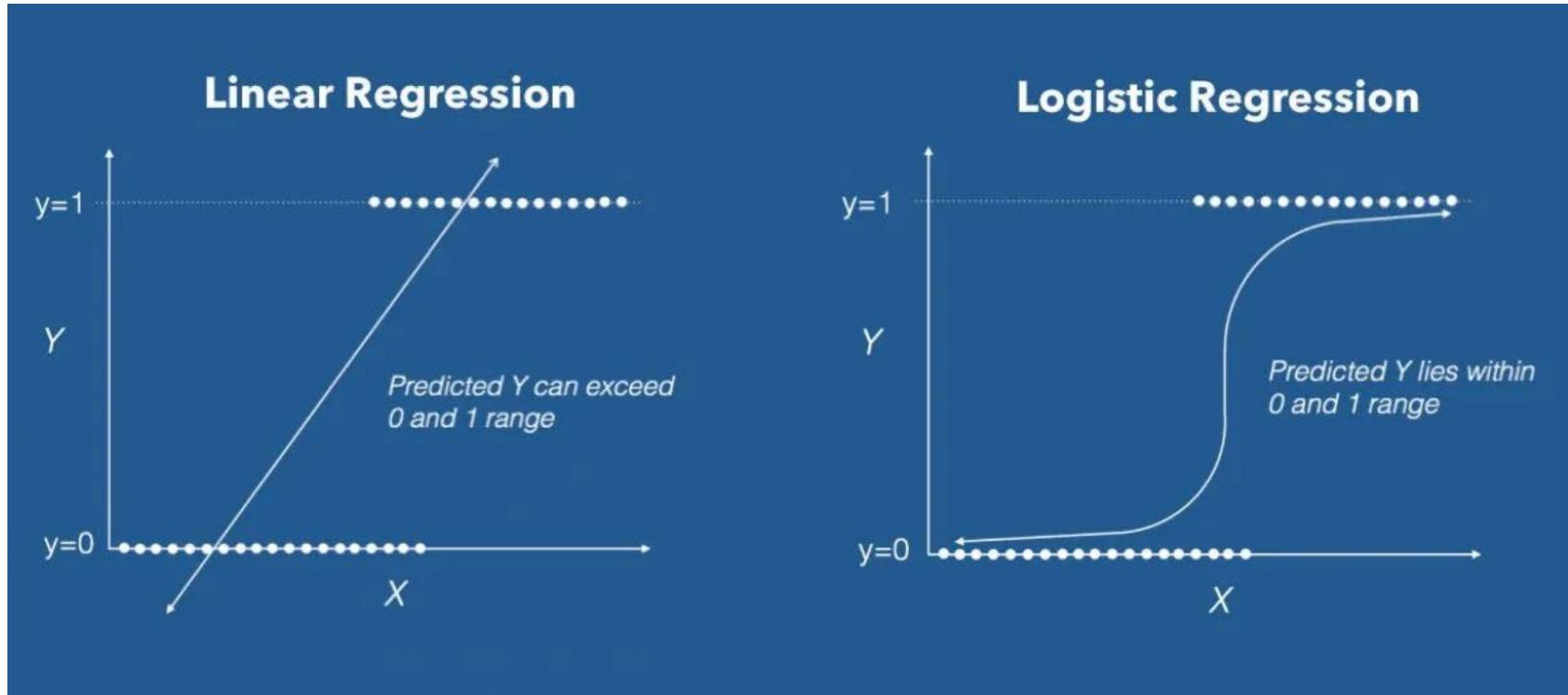
$$\mathbf{b} = \begin{bmatrix} b_0 \\ b_1 \\ b_2 \end{bmatrix} = \begin{bmatrix} 20 \\ 0.5 \\ 0.5 \end{bmatrix} \rightarrow \hat{y} = 20 + 0.5x_1 + 0.5x_2$$

# Logistic Regression

- predicts if something is true or false
- provides probabilities and classifies new samples using continuous and discrete measurements
- examples:
  - predict if a person is prone to a disease, the dataset contains different medical features about persons and a label if the person is prone or not
  - predict if person would elect a political party, the dataset contains different features about persons and a label if the person would elect the political party or not

# Logistic Regression

- why not linear regression?



# Logistic Regression

- example

- a group of 20 students spends between 0 and 6 hours studying for an exam. How does the number of hours spent studying affect the probability of the student passing the exam?
- data

Hours	0.50	0.75	1.00	1.25	1.50	1.75	1.75	2.00	2.25	2.50	2.75	3.00	3.25	3.50	4.00	4.25	4.50	4.75	5.00	5.50
Pass	0	0	0	0	0	0	1	0	1	0	1	0	1	0	1	0	1	1	1	1

- results of logistic regression analysis

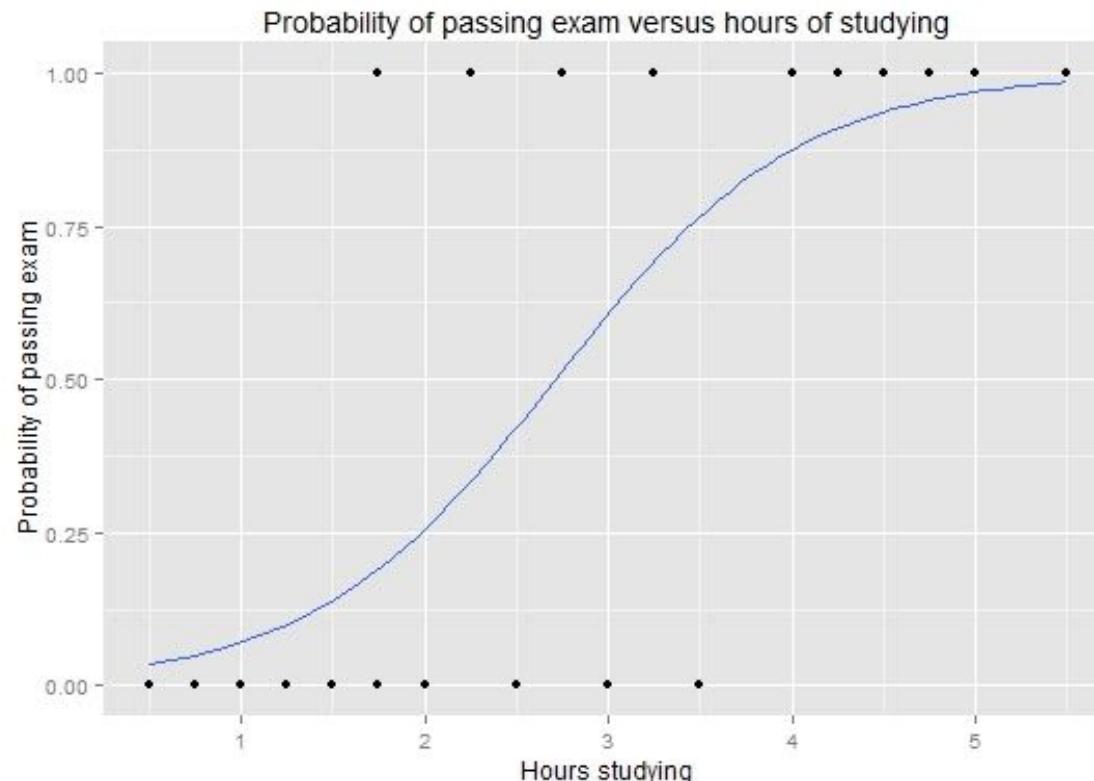
	Coefficient	Std.Error	z-value	P-value (Wald)
Intercept	-4.0777	1.7610	-2.316	0.0206
Hours	1.5046	0.6287	2.393	0.0167

- hours of studying is significantly associated with the probability of passing the exam (p-value 0.0167)

# Logistic Regression

- example

- a group of 20 students spends between 0 and 6 hours studying for an exam. How does the number of hours spent studying affect the probability of the student passing the exam?



# Logistic Regression

- example
  - results of logistic regression analysis

	Coefficient	Std.Error	z-value	P-value (Wald)
Intercept	-4.0777	1.7610	-2.316	0.0206
Hours	1.5046	0.6287	2.393	0.0167

- logistic model

$$p = \frac{1}{1 + e^{-(b_0 + b_1 x)}}$$

- probability of passing exam  $\frac{1}{1 + \exp(-(1.5046 \cdot \text{Hours} - 4.0777))}$
- studying 2 hours  $\rightarrow 0.26$  chance to pass the exam
- studying 4 hours  $\rightarrow 0.87$  chance to pass the exam

# Logistic Regression

- example

- whether or not college basketball players got drafted into the NBA (draft: 0 = no, 1 = yes) based on their average points, rebounds, and assists in the previous season

<b>draft</b>	<b>points</b>	<b>rebounds</b>	<b>assists</b>
0	12	3	6
1	13	4	4
0	13	4	6
1	12	9	9
1	14	4	5
0	14	4	4
0	17	2	2
1	17	6	5
1	21	5	7
0	21	9	3
1	24	11	11
0	24	4	5

# Logistic Regression

- example

- $b_0 = 0.001; b_1 = 0.001; b_2 = 0.001; b_3 = 0.001$
- $\text{logit} = b_0 + b_1 * \text{points} + b_2 * \text{rebounds} + b_3 * \text{assists}$

<b>draft</b>	<b>points</b>	<b>rebounds</b>	<b>assists</b>	<b>logit</b>
0	12	3	6	0.022
1	13	4	4	0.022
0	13	4	6	0.024
1	12	9	9	0.031
1	14	4	5	0.024
0	14	4	4	0.023
0	17	2	2	0.022
1	17	6	5	0.029
1	21	5	7	0.034
0	21	9	3	0.034
1	24	11	11	0.047
0	24	4	5	0.034

# Logistic Regression

- example

- $b_0 = 0.001; b_1 = 0.001; b_2 = 0.001; b_3 = 0.001$
- calculate  $e^{\text{logit}}$

<b>draft</b>	<b>points</b>	<b>rebounds</b>	<b>assists</b>	<b>logit</b>	<b><math>e^{\text{logit}}</math></b>
0	12	3	6	0.022	1.0222
1	13	4	4	0.022	1.0222
0	13	4	6	0.024	1.0243
1	12	9	9	0.031	1.0315
1	14	4	5	0.024	1.0243
0	14	4	4	0.023	1.0233
0	17	2	2	0.022	1.0222
1	17	6	5	0.029	1.0294
1	21	5	7	0.034	1.0346
0	21	9	3	0.034	1.0346
1	24	11	11	0.047	1.0481
0	24	4	5	0.034	1.0346

# Logistic Regression

- example

- $b_0 = 0.001; b_1 = 0.001; b_2 = 0.001; b_3 = 0.001$
- calculate probability =  $1 / (1 + e^{\text{logit}})$

<b>draft</b>	<b>points</b>	<b>rebounds</b>	<b>assists</b>	<b>logit</b>	<b><math>e^{\text{logit}}</math></b>	<b>probability</b>
0	12	3	6	0.022	1.0222	0.4945
1	13	4	4	0.022	1.0222	0.4945
0	13	4	6	0.024	1.0243	0.4940
1	12	9	9	0.031	1.0315	0.4923
1	14	4	5	0.024	1.0243	0.4940
0	14	4	4	0.023	1.0233	0.4943
0	17	2	2	0.022	1.0222	0.4945
1	17	6	5	0.029	1.0294	0.4928
1	21	5	7	0.034	1.0346	0.4915
0	21	9	3	0.034	1.0346	0.4915
1	24	11	11	0.047	1.0481	0.4883
0	24	4	5	0.034	1.0346	0.4915

# Logistic Regression

- example

- $b_0 = 0.001; b_1 = 0.001; b_2 = 0.001; b_3 = 0.001$
- calculate log likelihood =  $\ln(\text{probability})$

<b>draft</b>	<b>points</b>	<b>rebounds</b>	<b>assists</b>	<b>logit</b>	<b><math>e^{\text{logit}}</math></b>	<b>probability</b>	<b>log likelihood</b>
0	12	3	6	0.022	1.0222	0.4945	-0.6822
1	13	4	4	0.022	1.0222	0.4945	-0.7042
0	13	4	6	0.024	1.0243	0.4940	-0.6812
1	12	9	9	0.031	1.0315	0.4923	-0.7088
1	14	4	5	0.024	1.0243	0.4940	-0.7052
0	14	4	4	0.023	1.0233	0.4943	-0.6817
0	17	2	2	0.022	1.0222	0.4945	-0.6822
1	17	6	5	0.029	1.0294	0.4928	-0.7078
1	21	5	7	0.034	1.0346	0.4915	-0.7103
0	21	9	3	0.034	1.0346	0.4915	-0.6763
1	24	11	11	0.047	1.0481	0.4883	-0.7169
0	24	4	5	0.034	1.0346	0.4915	-0.6763

# Logistic Regression

- example

- $b_0 = 0.001; b_1 = 0.001; b_2 = 0.001; b_3 = 0.001$
- calculate the sum of the log likelihood = -8.3331

<b>draft</b>	<b>points</b>	<b>rebounds</b>	<b>assists</b>	<b>logit</b>	<b><math>e^{\text{logit}}</math></b>	<b>probability</b>	<b>log likelihood</b>
0	12	3	6	0.022	1.0222	0.4945	-0.6822
1	13	4	4	0.022	1.0222	0.4945	-0.7042
0	13	4	6	0.024	1.0243	0.4940	-0.6812
1	12	9	9	0.031	1.0315	0.4923	-0.7088
1	14	4	5	0.024	1.0243	0.4940	-0.7052
0	14	4	4	0.023	1.0233	0.4943	-0.6817
0	17	2	2	0.022	1.0222	0.4945	-0.6822
1	17	6	5	0.029	1.0294	0.4928	-0.7078
1	21	5	7	0.034	1.0346	0.4915	-0.7103
0	21	9	3	0.034	1.0346	0.4915	-0.6763
1	24	11	11	0.047	1.0481	0.4883	-0.7169
0	24	4	5	0.034	1.0346	0.4915	-0.6763

# Logistic Regression

- example

- repeat until “best” sum of the log likelihood is detected
- $b_0 = 3.681193; b_1 = 0.112827; b_2 = -0.39568; b_3 = -0.67954$
- calculate the sum of the log likelihood = -5.9428

draft	points	rebounds	assists	logit	e <sup>logit</sup>	probability	log likelihood
0	12	3	6	-0.2291	0.7952	0.5570	-0.8143
1	13	4	4	0.84705	2.3328	0.3001	-1.2038
0	13	4	6	-0.5120	0.5993	0.6253	-0.9816
1	12	9	9	-4.6418	0.0096	0.9905	-0.0096
1	14	4	5	0.28033	1.3236	0.4304	-0.8431
0	14	4	4	0.95987	2.6114	0.2769	-0.3242
0	17	2	2	3.44880	31.4627	0.0308	-0.0313
1	17	6	5	-0.1722	0.8415	0.5430	-0.6106
1	21	5	7	-0.6846	0.5043	0.6648	-0.4083
0	21	9	3	0.45078	1.5696	0.3892	-0.4929
1	24	11	11	-5.4384	0.0043	0.9957	-0.0043
0	24	4	5	1.40860	4.0902	0.1965	-0.2187

# Logistic Regression

- example

- calculate for a player with 14 points, 4 rebounds, and 5 assists

- $b_0 = 3.681193; b_1 = 0.112827; b_2 = -0.39568; b_3 = -0.67954$

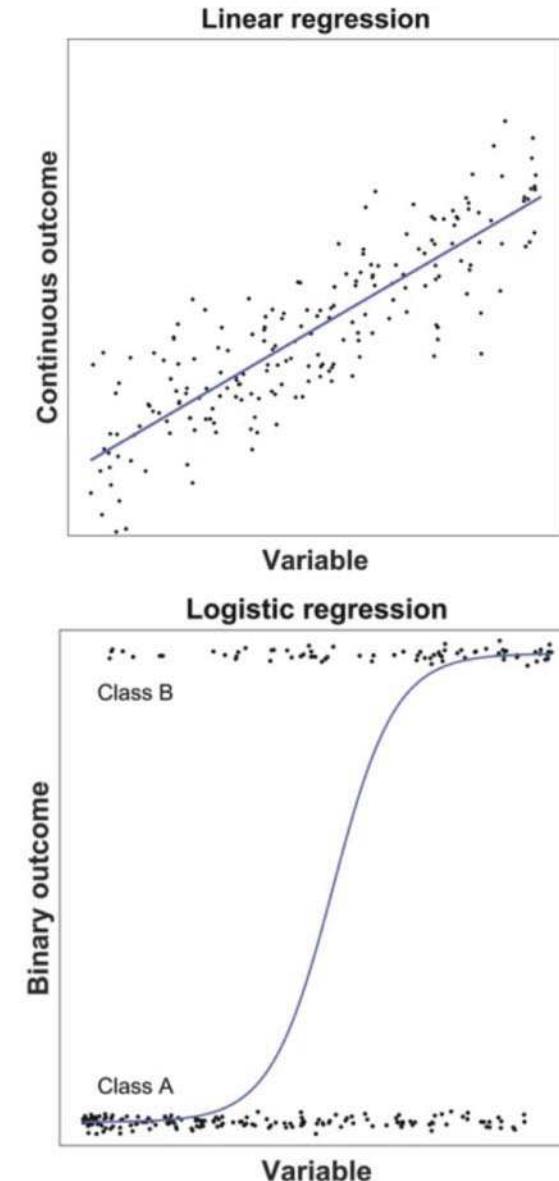
- $p(\text{draft} = 1) =$

$$\begin{aligned} & 1 / \\ & (1 + e^{3.681193 + 0.112827 * 14 - 0.39568 * 4 - 0.67954 * 5}) \\ & = 0.57 \end{aligned}$$

- result  $0.57 \geq 0.5$  so we can predict that the player will be drafted into the NBA

# Linear vs Logistic Regression

LINEAR REGRESSION VERSUS LOGISTIC REGRESSION	
<b>LINEAR REGRESSION</b>	<b>LOGISTIC REGRESSION</b>
A linear approach that models the relationship between a dependent variable and one or more independent variables	A statistical model that predicts the probability of an outcome that can only have two values
Used to solve regression problems	Used to solve classification problems (binary classification)
Estimates the dependent variable when there is a change in the independent variable	Calculates the possibility of an event occurring
Output value is continuous	Output value is discrete
Uses a straight line	Uses an S curve or sigmoid function
Ex: predicting the GDP of a country, predicting product price, predicting the house selling price, score prediction	Ex: predicting whether an email is spam or not, predicting whether the credit card transaction is fraud or not, predicting whether a customer will take a loan or not
Visit <a href="http://www.PEDIAA.com">www.PEDIAA.com</a>	



# Lessons Learned

- fundamental understanding of the gradient descent method for linear regression
- fundamental understanding of logistic regression



# Questions and Answers



# Machine Learning and Data Mining

## V08: Support Vector Machines

Lecturer: Dr. Andreas Weiler



# Repetition

... REPETITION ...  
→ IS THE MOTHER  
→ of learning

# Course Content IT20a\_WIN

Week	Lecture	Lab
KW08	Introduction	Lab 01
KW09	Data Processing Part 01	Lab 02
KW10	Carnival Monday	
KW11	Data Processing Part 02	Lab 03
KW12	Evaluation	Lab 04
KW13	Recommender Systems	<b>Graded Lab 01</b> <span style="border: 1px solid red; padding: 2px;">IMPORTANT</span>
KW14	Association Rules	<b>Graded Lab 01</b> <span style="border: 1px solid red; padding: 2px;">IMPORTANT</span>
KW15	Linear and Logistic Regression	Lab 05
KW16	Easter Monday	
KW17	Support Vector Machines	Lab 06
KW18	Decision Trees and Naïve Bayes	<b>Graded Lab 02</b> <span style="border: 1px solid red; padding: 2px;">IMPORTANT</span>
KW19	Partition Clustering	<b>Graded Lab 02</b> <span style="border: 1px solid red; padding: 2px;">IMPORTANT</span>
KW20	Hierarchical and Density Clustering	Lab 07
KW21	Neural Networks and Closing	Review and Questions for Exam

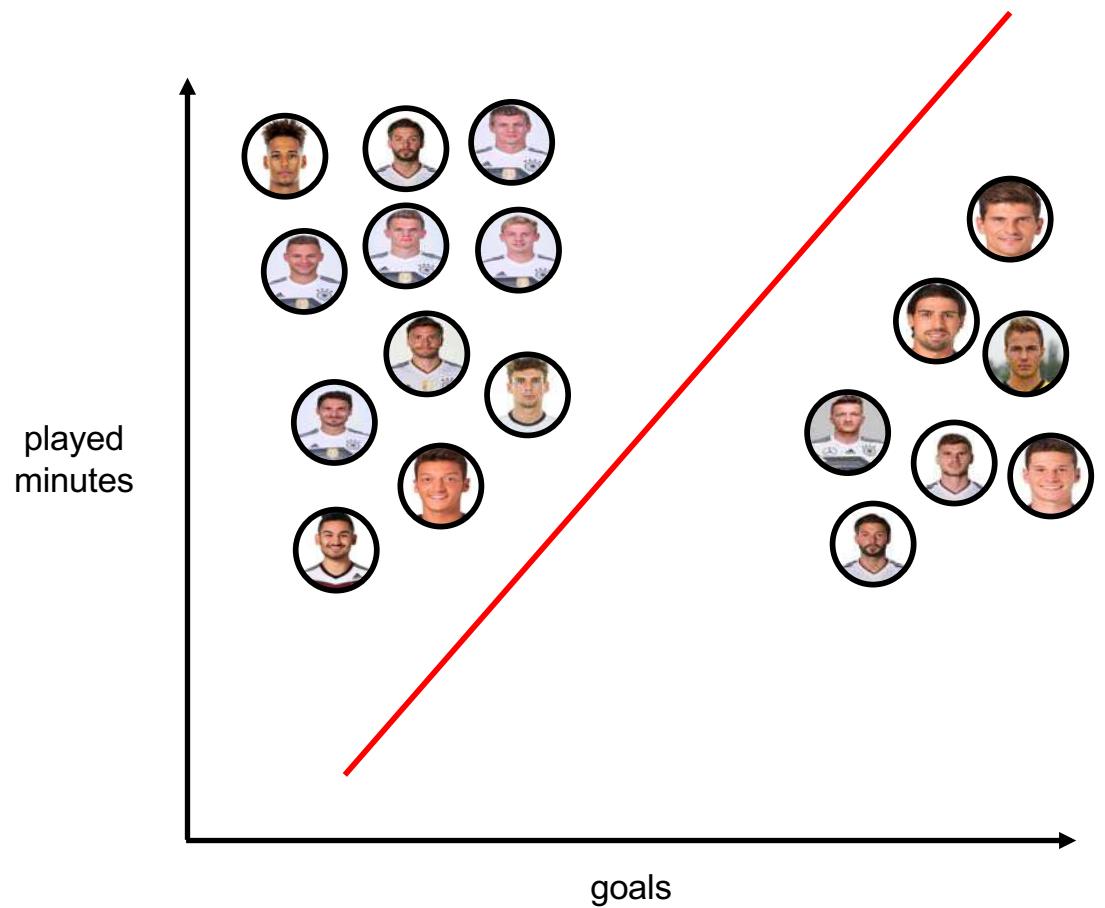
# Learning Objectives

- understanding of classification
- understanding of the support vector machine
- understanding of multiclass classification



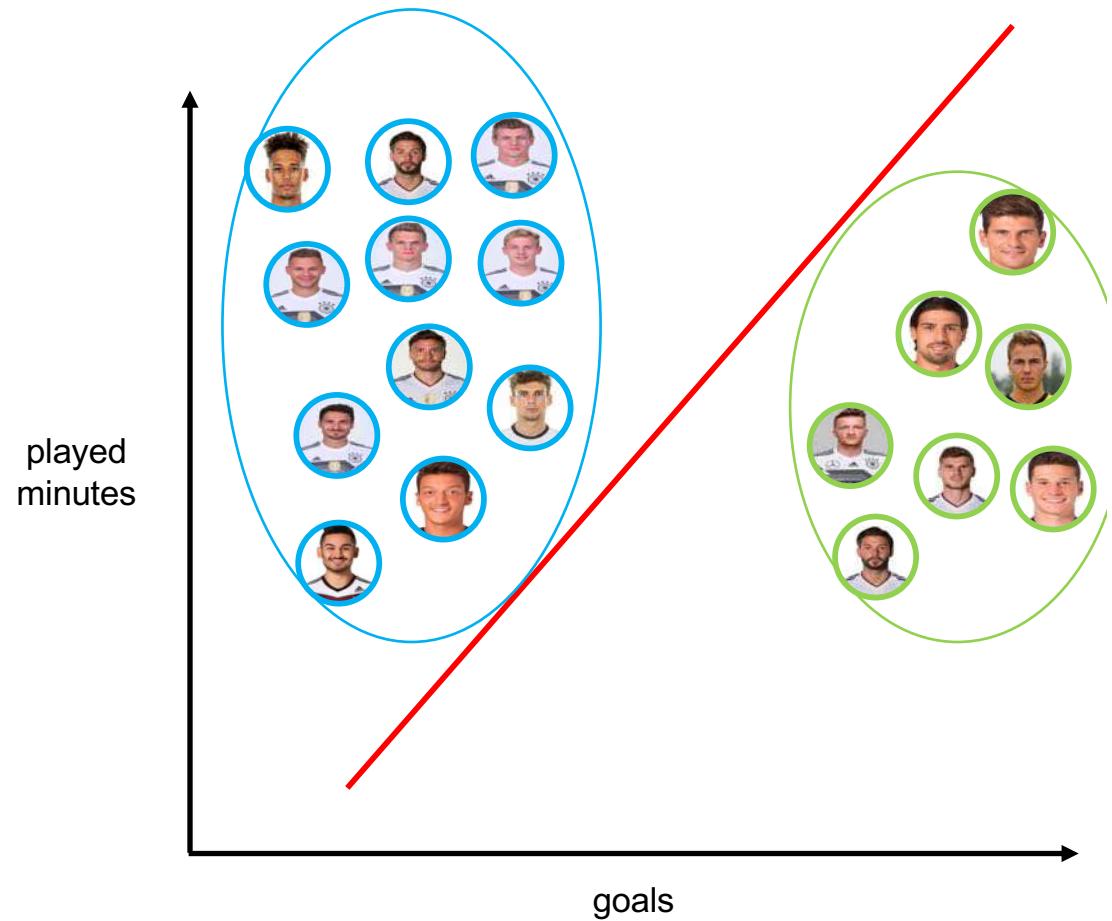
# Classification

- separate one or more classes
- predict class for data points with unknown class
- many algorithms exist
  - fisher linear discriminant
  - K-nearest neighbor
  - logistic regression
  - decision tree
  - neural networks
  - SVM
  - naïve bayes
  - Adaboost
  - ... and many many more!



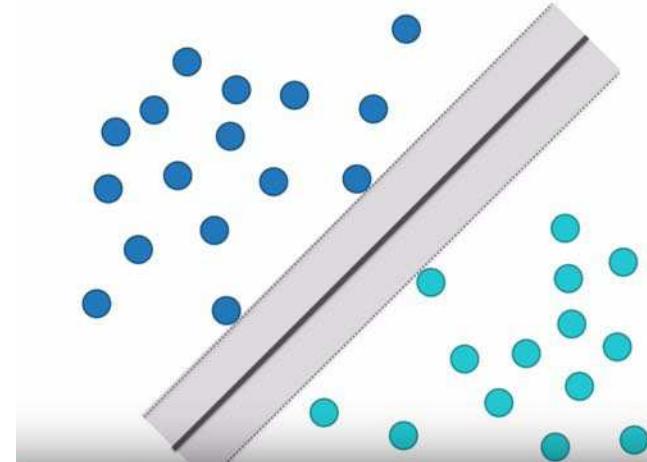
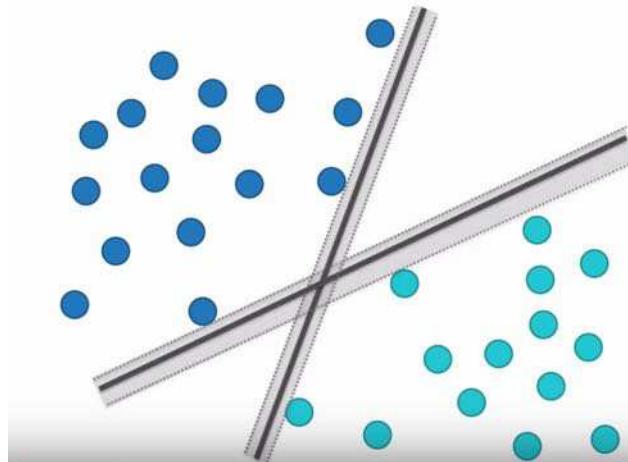
# Classification

- separate one or more classes
- predict class for data points with unknown class
- many algorithms exist
  - fisher linear discriminant
  - K-nearest neighbor
  - logistic regression
  - decision tree
  - neural networks
  - SVM
  - naïve bayes
  - Adaboost
  - ... and many many more!



# Support Vector Machines

- goal: find a hyperplane that separates the two classes of input points with maximum margin



Vladimir Vapnik  
(1963)

- points on the decision boundary (the "gutter of the street") are called support vectors
- only "some" points influence the decision boundary, in contrast to other methods

# Support Vector Machines

- family of classifiers
  - (maximal margin) hyperplane classifier (for linearly separable data)
  - support vector classifier (for almost linearly separable data)
  - support vector machine (for non-linearly separable data)

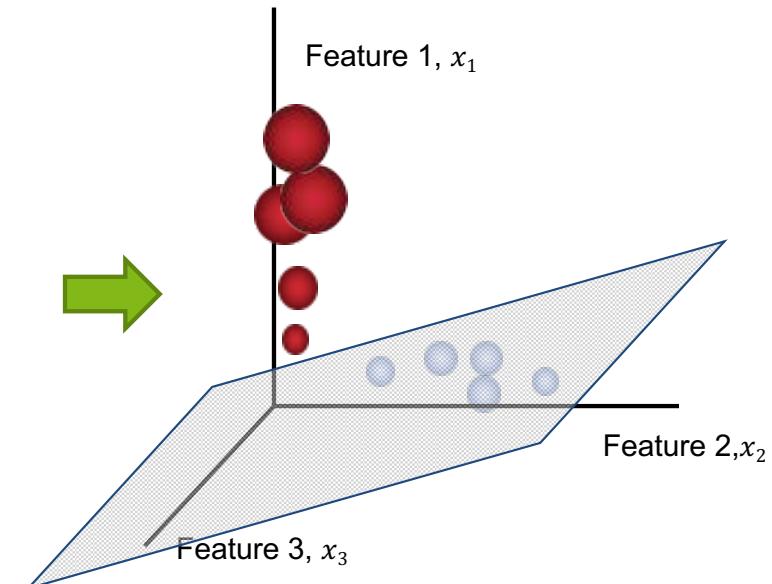
Feature vectors being column vectors here (as opposed to row vectors usually) makes the math more convenient later

$N$  observations

$p$  features

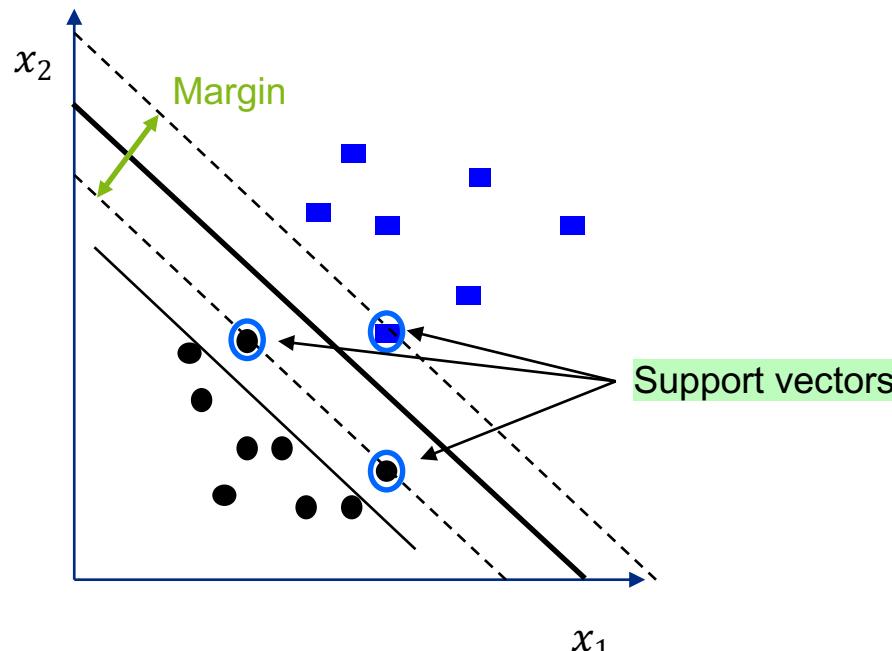
	1	2	3	4	5	6	7	8	9	10
1	1	2	1	3	4	4	5	7	6	6
2	2	4	2	6	8	10	2	1	2	1
3	3	1	5	6	6	2	1	2	2	1

Class 1      Class 2



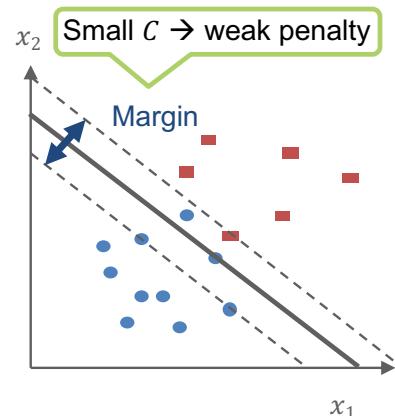
# Support Vector Machines

- support vectors for the maximum margin classifier
  - close (to the hyperplane) training examples determine the hyperplane ( $\rightarrow$  Far away training examples do not participate in its specification)
  - these examples are called the support vectors  $\rightarrow$  removing them would change the location of the separating hyperplane

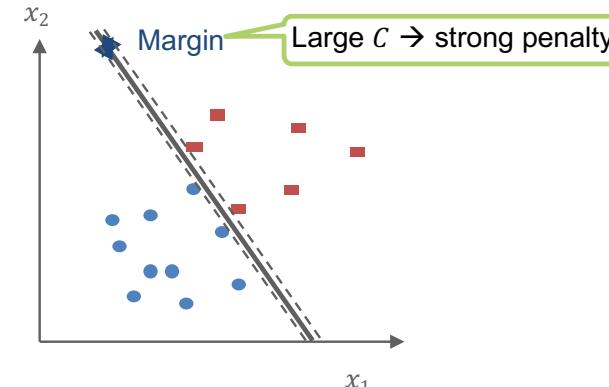


# Support Vector Machines

- idea of the  $C$  penalty factor on misclassifications
- new assumption: data is not linearly separable any more
  - idea of soft margins that allows for some misclassifications
  - data may contain untypical or mislabeled samples; process might really be (moderately) non-linear
  - the number of misclassifications is controlled by a penalty factor  $C$
- sometimes a larger margin is worth having some misclassified observations

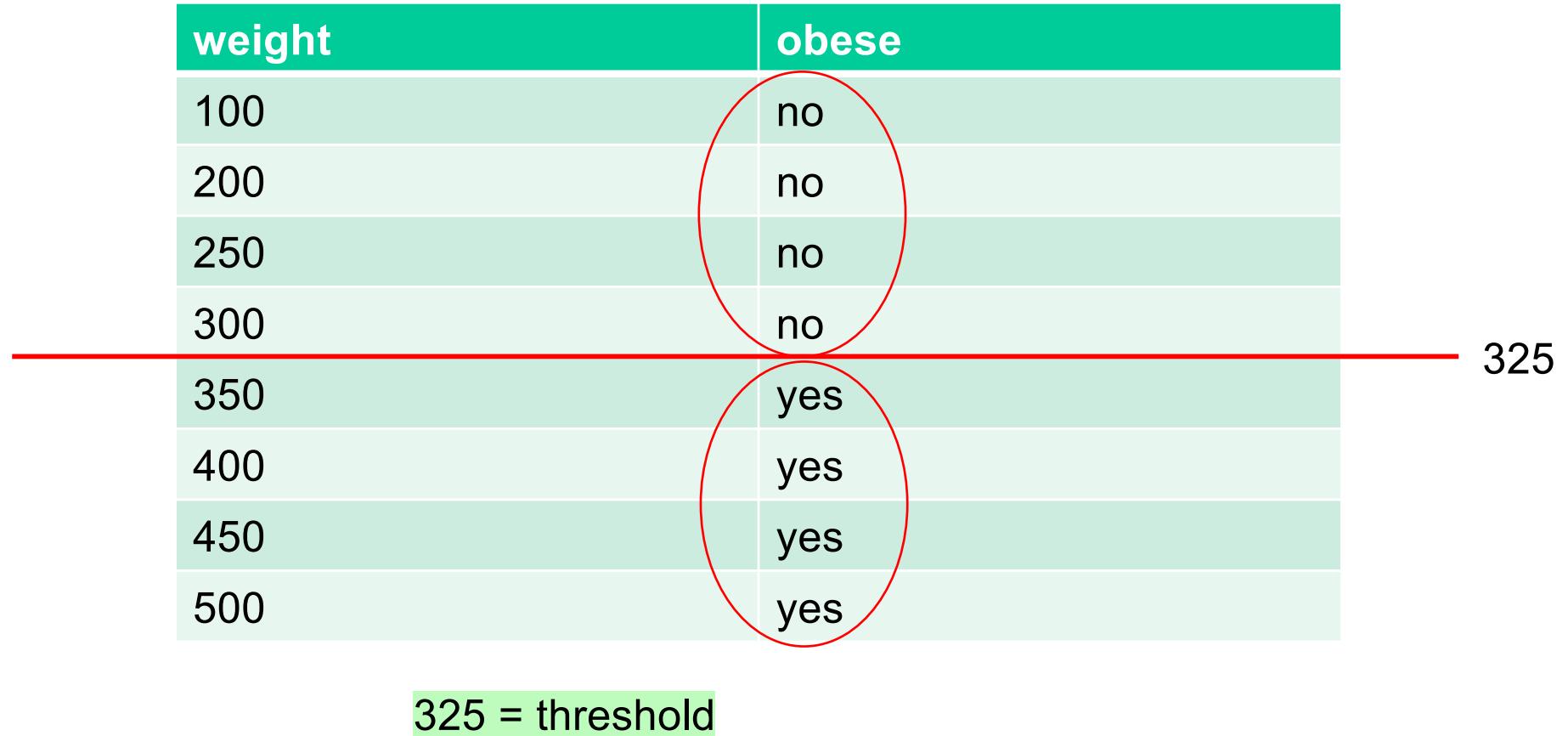


Low penalty: high # of misclassified training examples



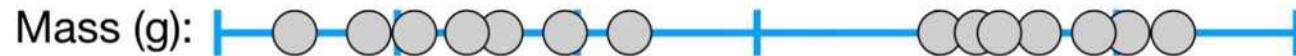
High penalty: low # of misclassified training examples

# Support Vector Machines



# Support Vector Machines

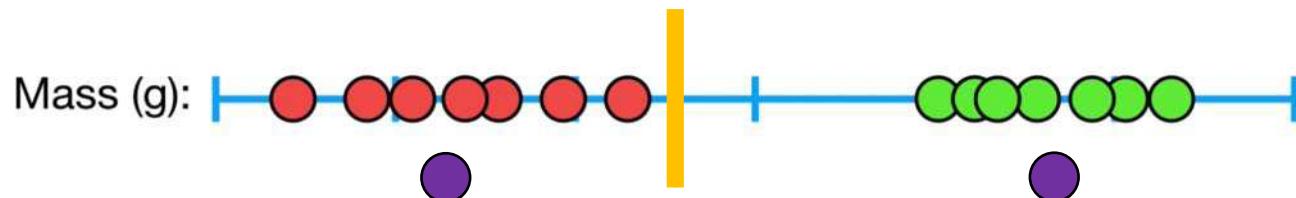
- lets imagine we measured the mass of a bunch of mice



- the red dots present mice that are not obese and the green dots present mice that are obese

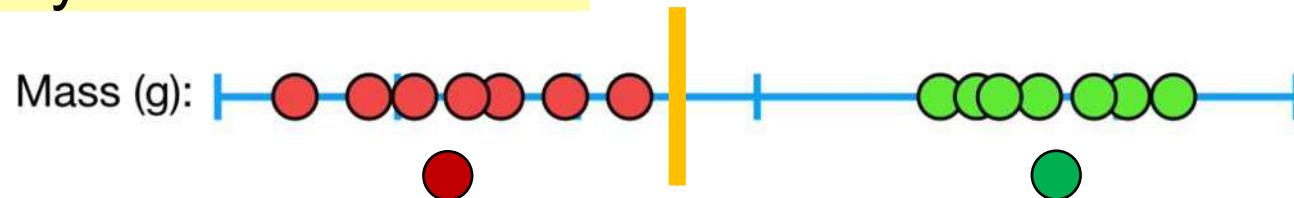


- based on these observations we can pick a threshold and classify new observations

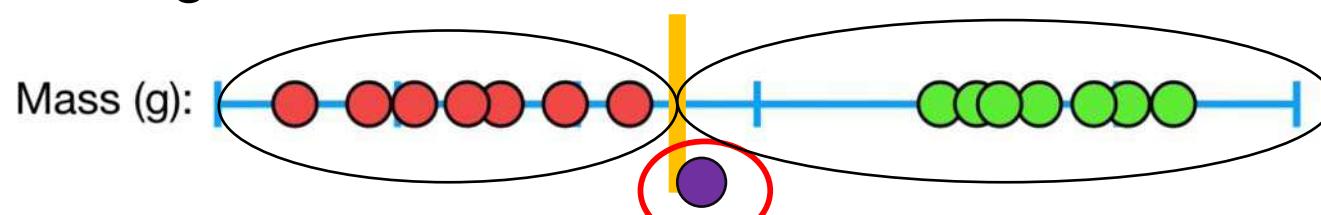


# Support Vector Machines

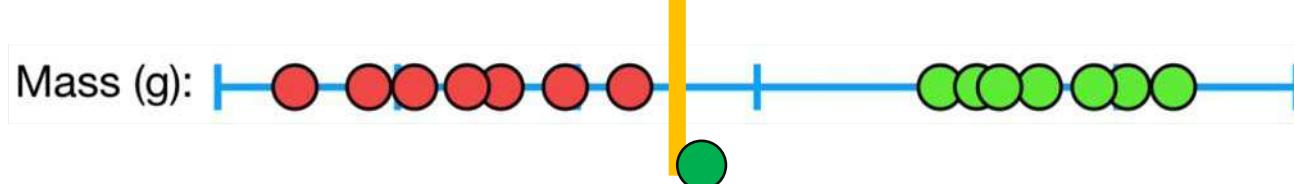
- based on these observations we can pick a threshold and classify new observations



- what if we get a new observation here?



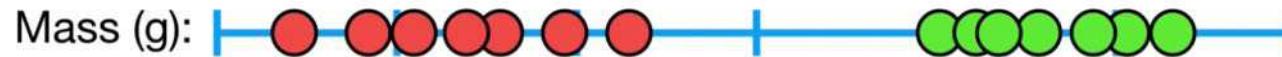
- we classify it as obese



- but isn't it closer to the observations that are not obese?

# Support Vector Machines

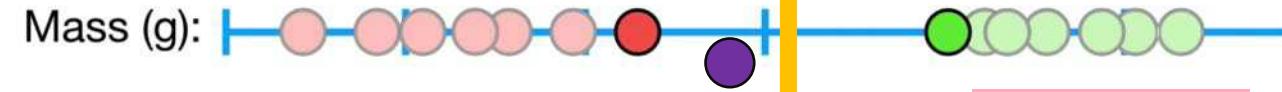
- going back to the original data set



- we can focus on the observations on the edges

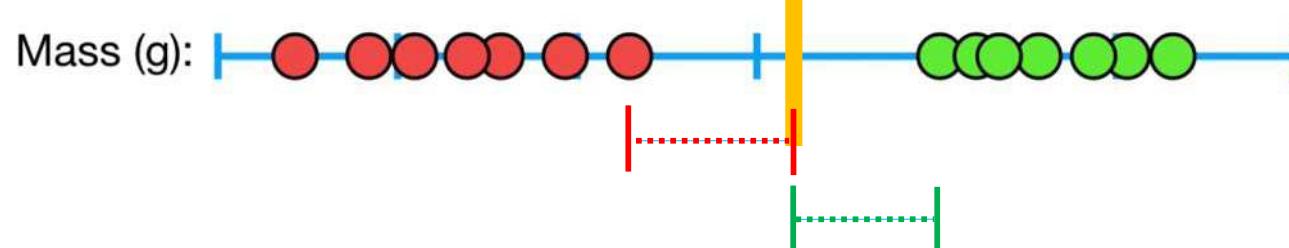


- and use the midpoint between them as threshold



- now the observation falls into the class **not obese**

- the **shortest distance** between the **closest observations** and the threshold is called the **margin** (maximal margin classifier)

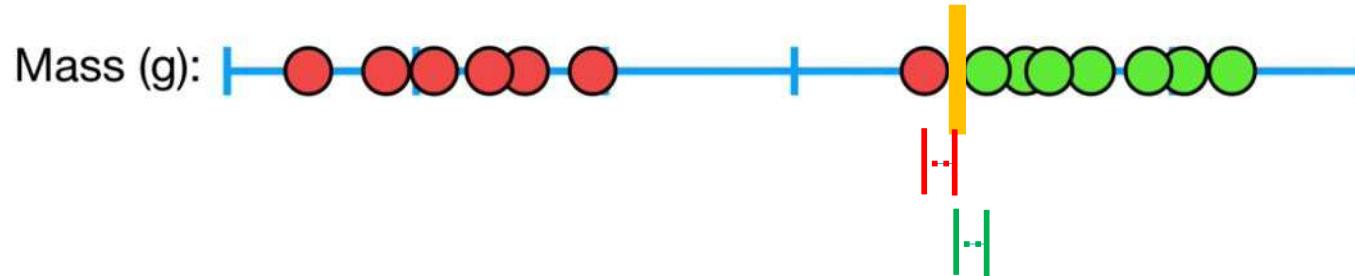


# Support Vector Machines

- problem solved?



- the maximum margin classifier is be really close



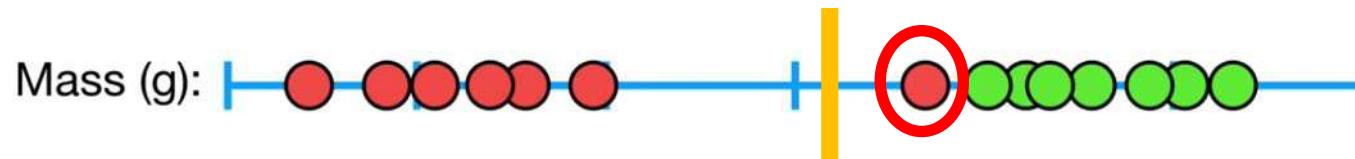
- so a new observation would be classified as not obese



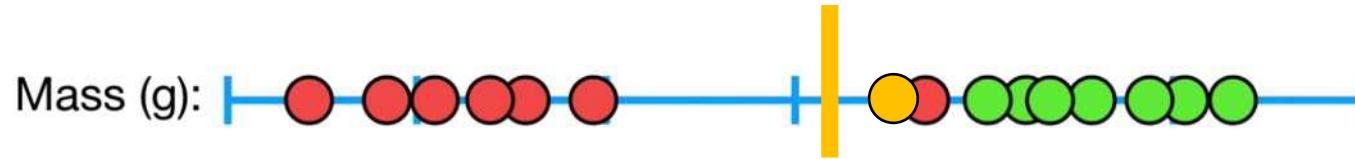
- maximal margin classifiers are super sensitive to outliers

# Support Vector Machines

- to make a threshold that is not so sensitive to outliers we must allow misclassifications



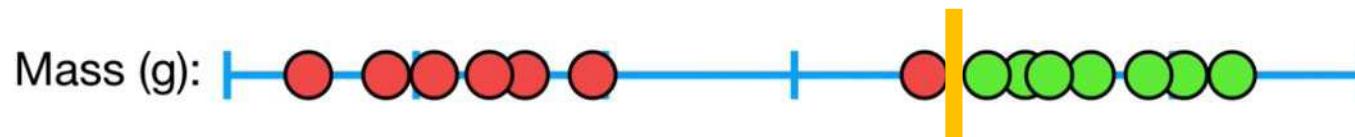
- with this threshold we will misclassify a training data point



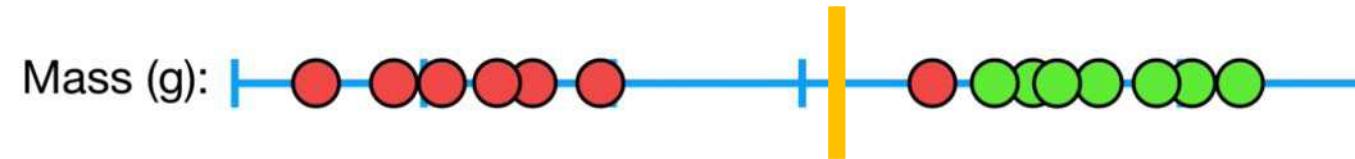
- but we will correctly classify this new observation
- choosing a threshold which allows misclassifications is an example of the bias/variance tradeoff

# Support Vector Machines

- threshold very sensitive to training data (low bias) but performed poorly for new observations (high variance)



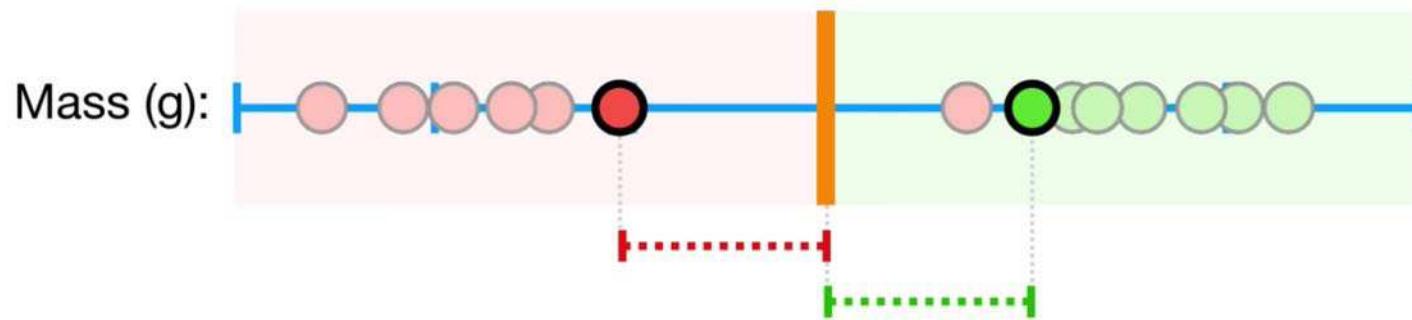
- threshold less sensitive to training data and allowed misclassifications (higher bias) but performed better for new observations (lower variance)



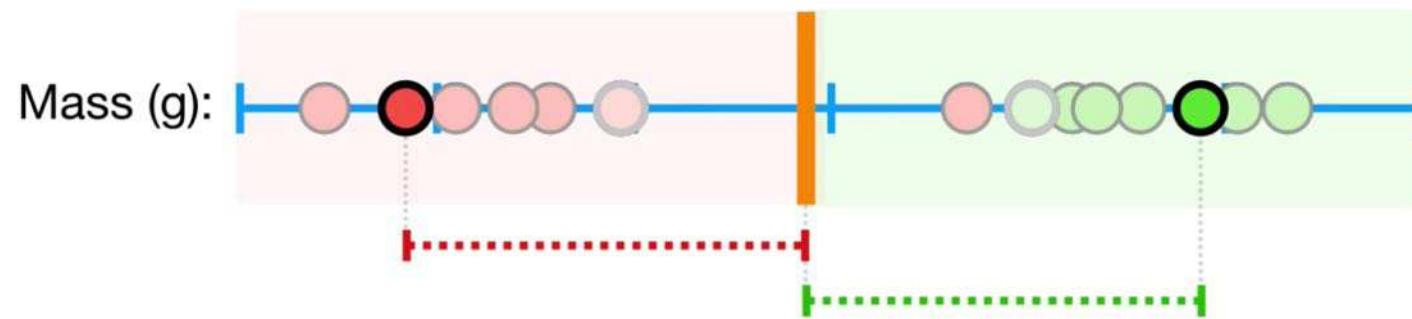
- when we allow misclassification between the threshold and the observations this is called soft margin

# Support Vector Machines

- how do we know which is the best soft margin?



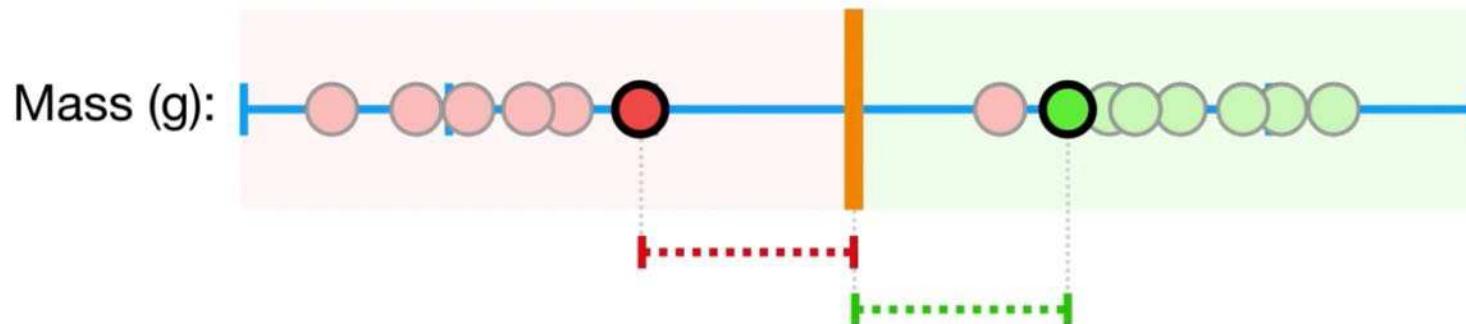
vs.



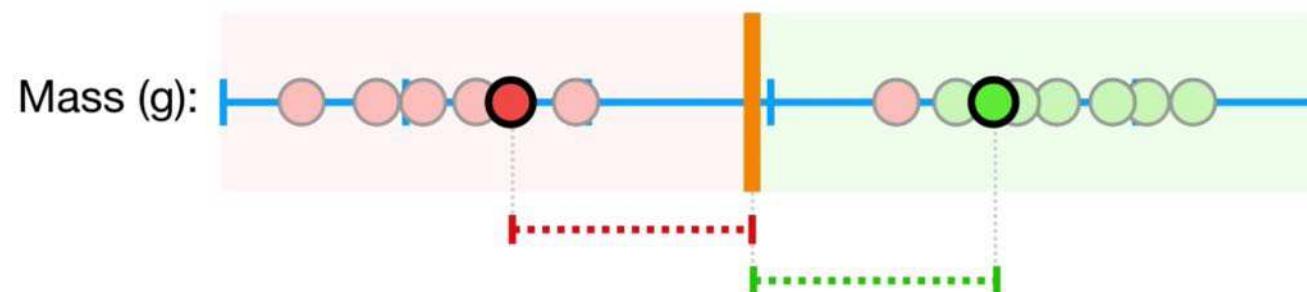
- use cross-validation to determine the ratio between misclassifications in the training vs. test data

# Support Vector Machines

- how do we know which is the best soft margin?
  - use cross-validation to determine the ratio between misclassifications in the training vs. test data



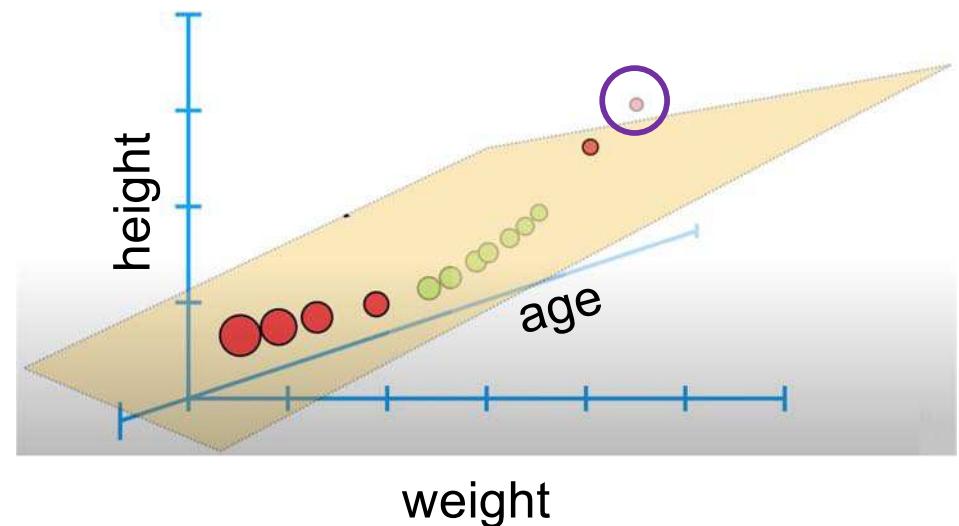
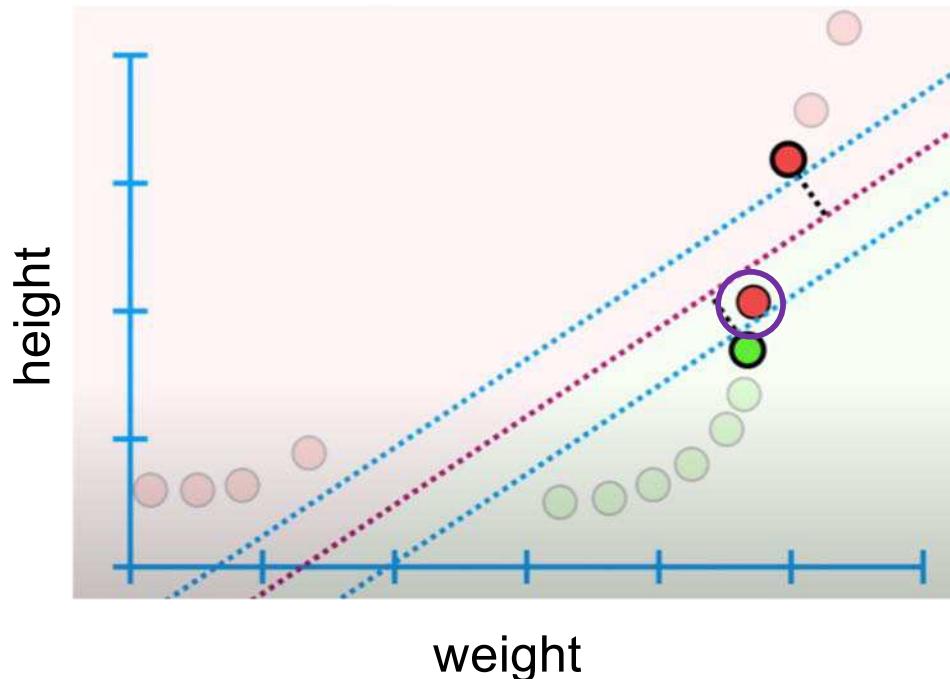
- soft margin classifier aka support vector classifier



- allow one misclassification and two observations that are correctly classified within the soft margin

# Support Vector Machines

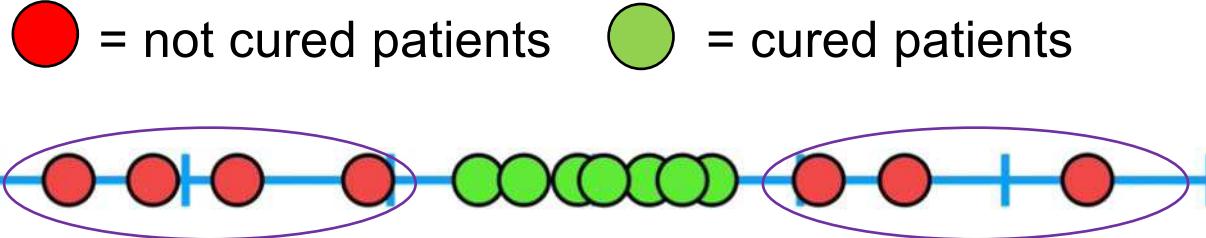
- more dimensions?



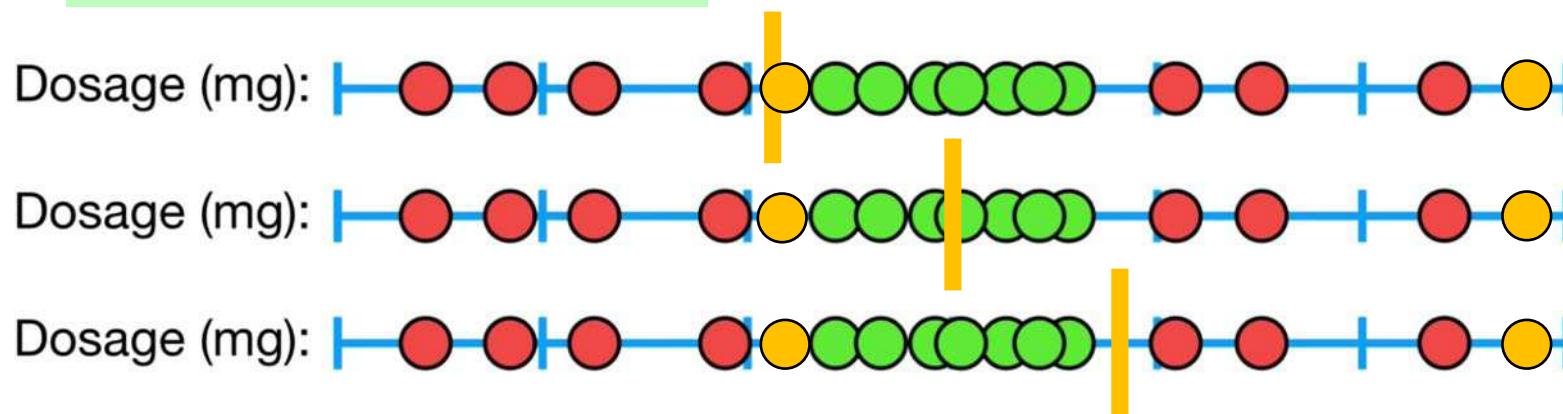
- when the data is 3-dimensional the support vector classifier forms a plane instead of a line
- for more than 3-dimensions it is called hyperplane

# Support Vector Machines

classifiers?

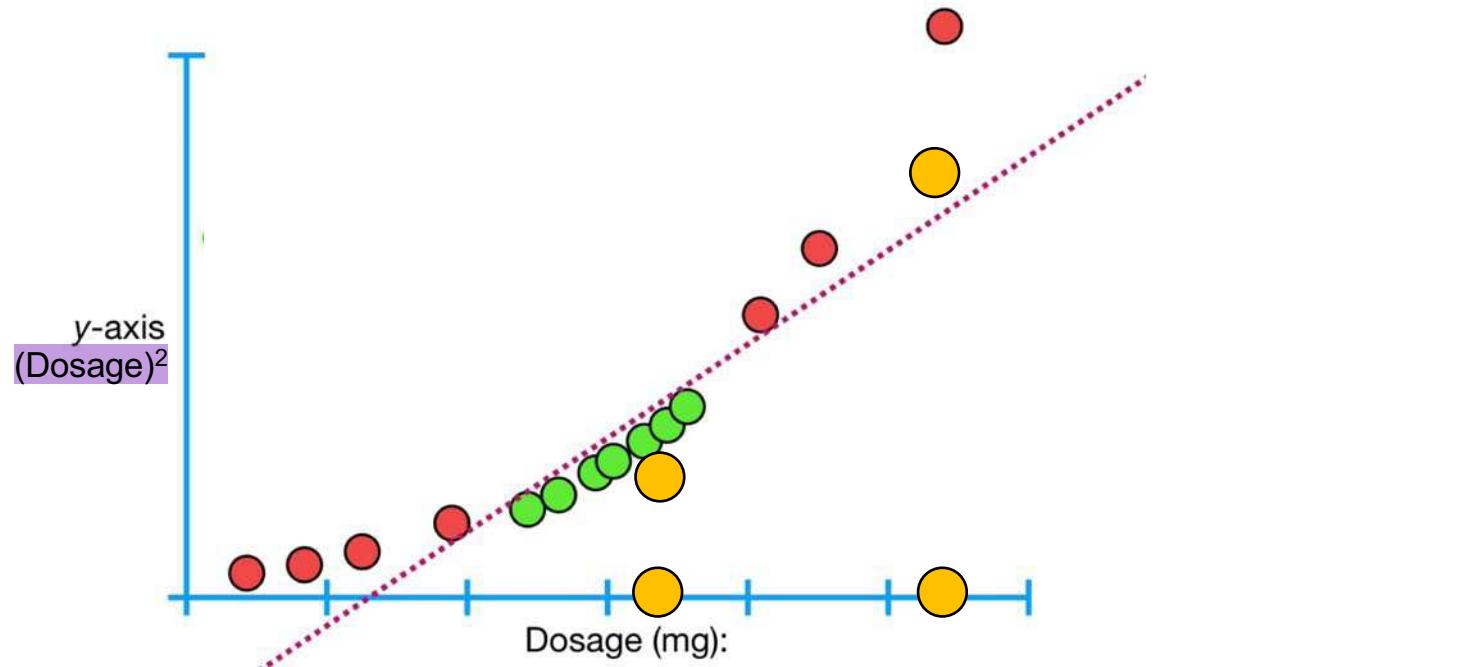
- support vector machines can handle outliers and overlapping classifications
- but what if we have tons of overlap → example drug dosage  


- no matter where we put the threshold, we will make a lot of misclassifications



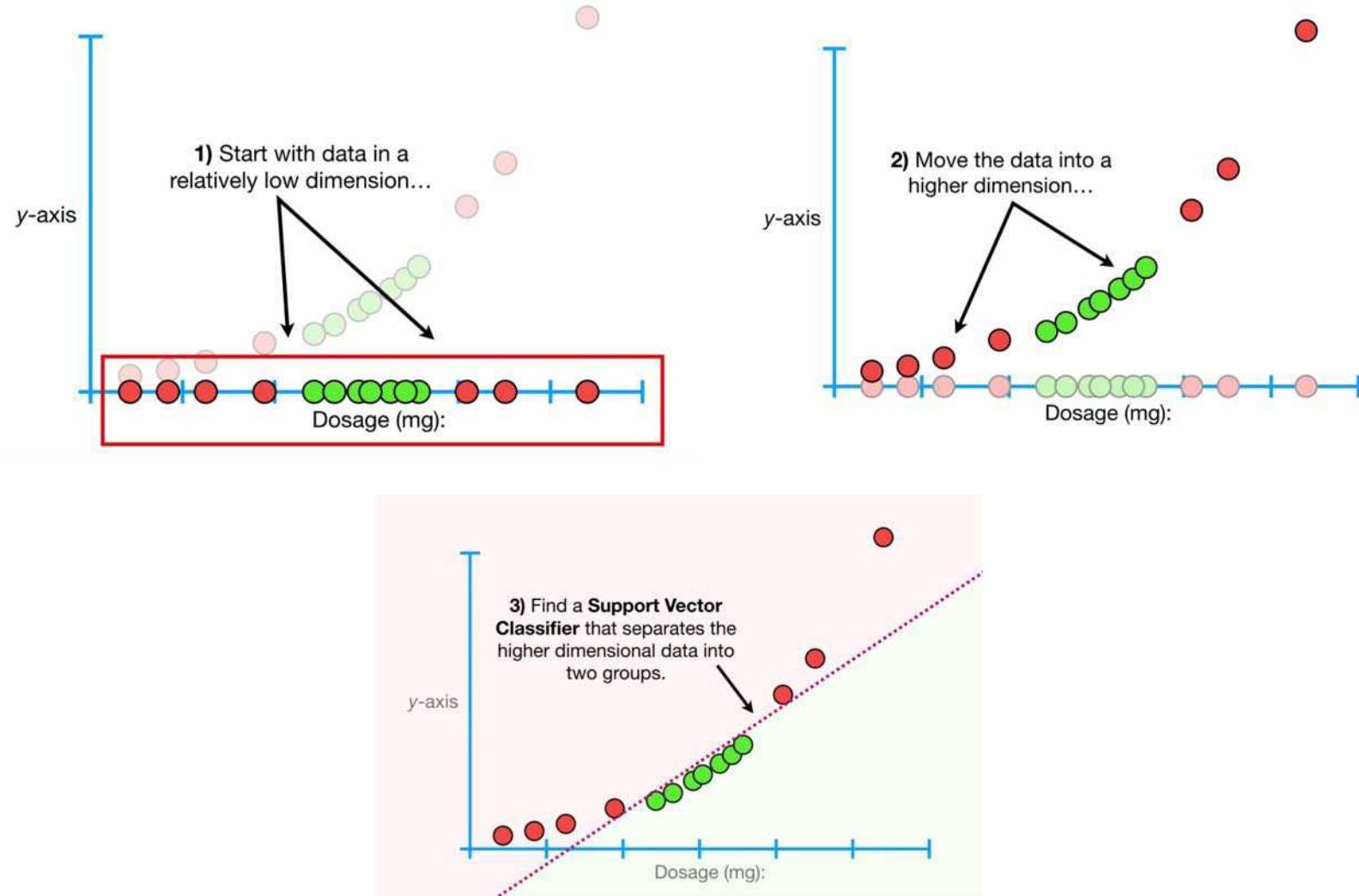
# Support Vector Machines

- can we do better than maximal margin classifiers and support vector classifiers? → Support Vector Machines
- we add a y-axis with the  $(\text{Dosage})^2$  as value
- now we are able to use a support vector classifier to separate the two classes and to classify new observations ●



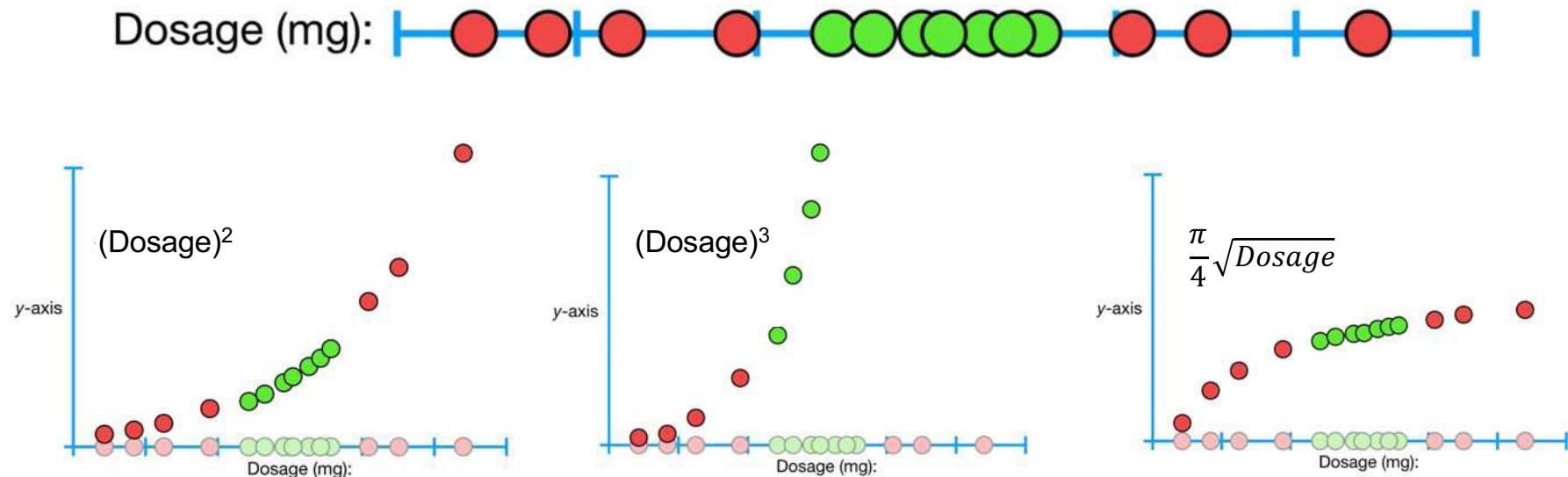
# Support Vector Machines

- the main ideas behind support vector machines are



# Support Vector Machines

- how do we decide how to transform the data?

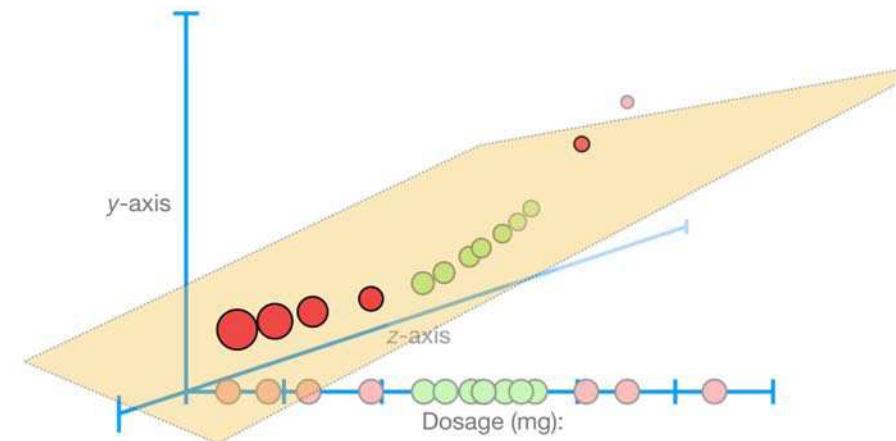
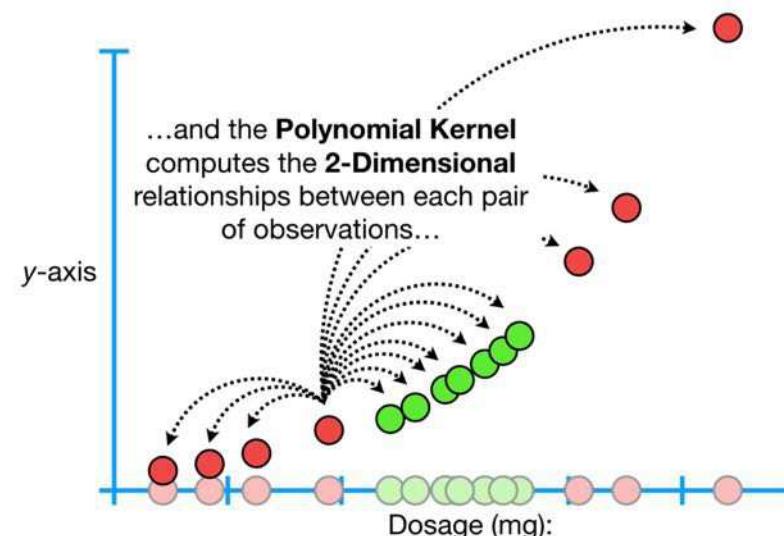


## Support Vector Machines

- SVMs use **kernel functions** to systematically find support vector classifiers in higher dimensions

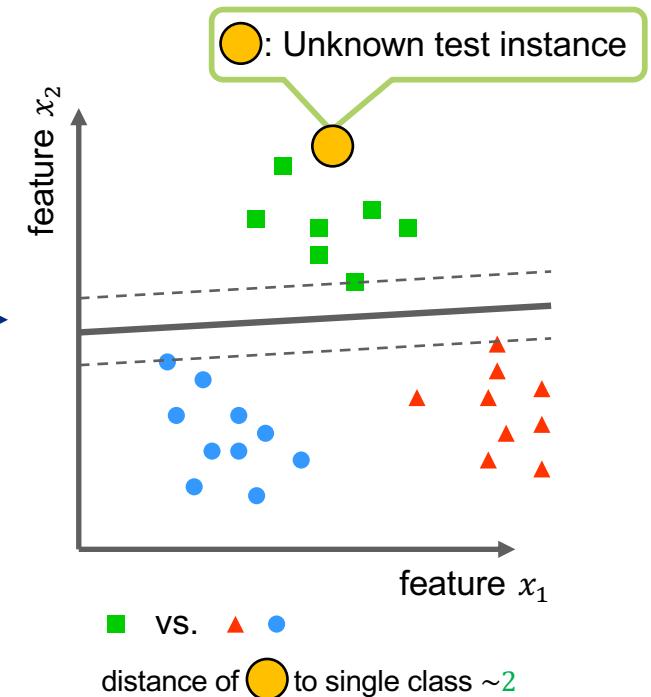
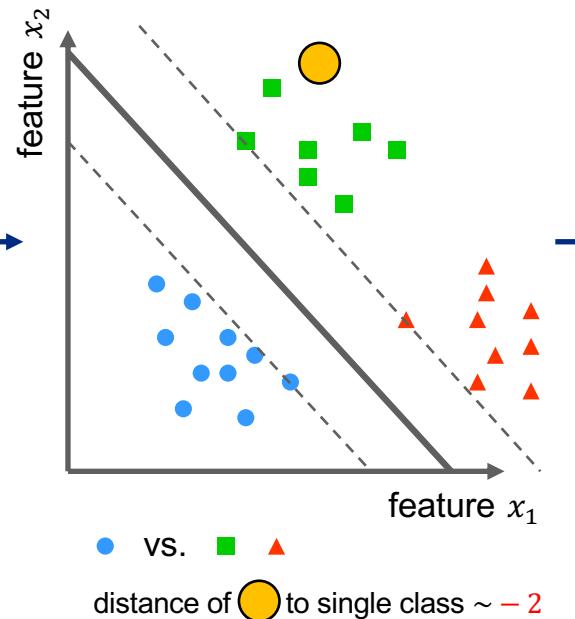
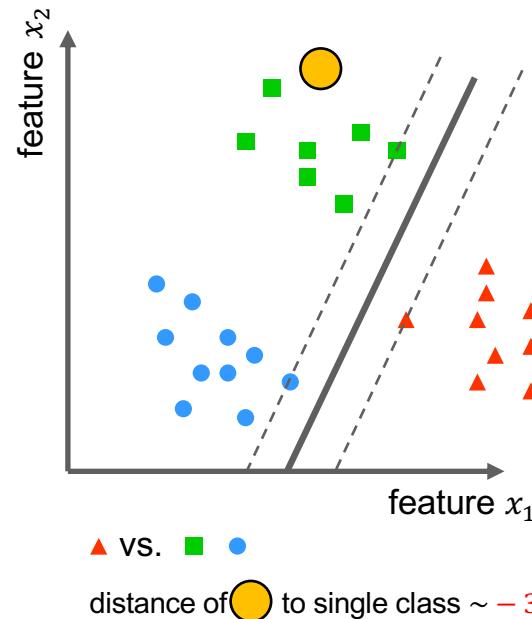
# Support Vector Machines

- kernel functions and trick
  - polynomial kernel with parameter  $d$  for the degree
  - with  $d = 1, 2, 3, \dots, n$  the polynomial kernel computes the relationships between each pair of observation in  $n$ -dimension
  - these relationships are used to find a support vector classifier
  - the kernel functions only calculate the relationships between every pair of points as if they are in the higher dimensions



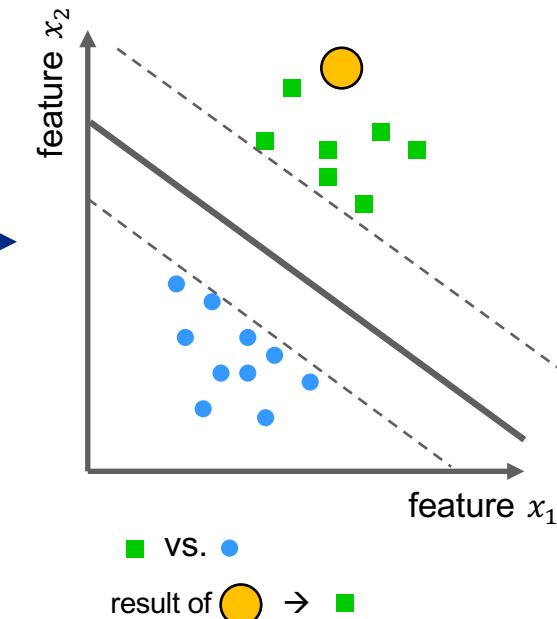
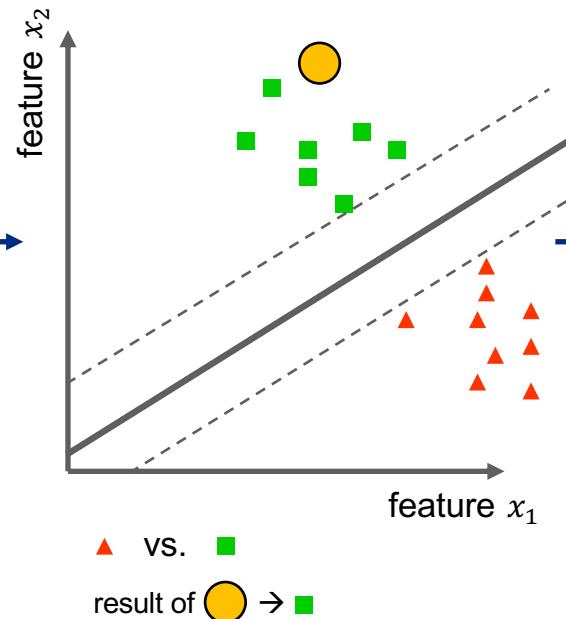
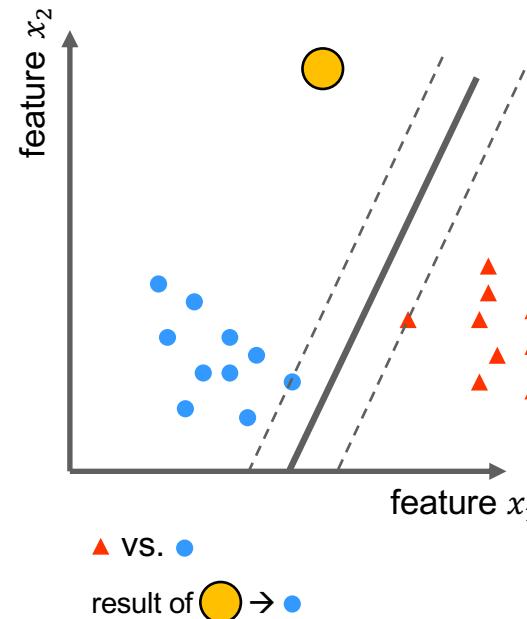
# Multiclass Classification

- one vs. rest (or one vs. all) classification
- SVM is a binary classifier, so it can only separate two classes
- more than 2 classes? → #classes times 'one vs. rest'
- ● has highest positive distance (confidence) in green case



# Multiclass Classification

- one vs. one classification
- assume  $k$  classes and learn all  $k(k-1)/2$  pairwise comparisons
- classify unknown instance by majority vote among all pairwise comparisons



# Lessons Learned

- understanding of classification
- understanding of the support vector machine
- understanding of multiclass classification



# Questions and Answers



# Machine Learning and Data Mining

## V09: Decision Trees and Naïve Bayes

Lecturer: Dr. Andreas Weiler



# Repetition

... REPETITION ...

→ IS THE MOTHER →

of learning

# Course Content IT20a\_WIN

Week	Lecture	Lab
KW08	Introduction	Lab 01
KW09	Data Processing Part 01	Lab 02
KW10	Carnival Monday	
KW11	Data Processing Part 02	Lab 04
KW12	Evaluation	Lab 05
KW13	Recommender Systems	<b>Graded Lab 01</b> <span style="border: 1px solid red; padding: 2px;">IMPORTANT</span>
KW14	Association Rules	<b>Graded Lab 01</b> <span style="border: 1px solid red; padding: 2px;">IMPORTANT</span>
KW15	Linear and Logistic Regression	Lab 06
KW16	Easter Monday	
KW17	Support Vector Machines	<b>Graded Lab 02</b> <span style="border: 1px solid red; padding: 2px;">IMPORTANT</span>
KW18	Decision Trees and Naïve Bayes	<b>Graded Lab 02</b> <span style="border: 1px solid red; padding: 2px;">IMPORTANT</span>
KW19	Partition Clustering	Lab 08
KW20	Hierarchical and Density Clustering	Lab 09
KW21	Neural Networks and Closing	Review and Questions for Exam

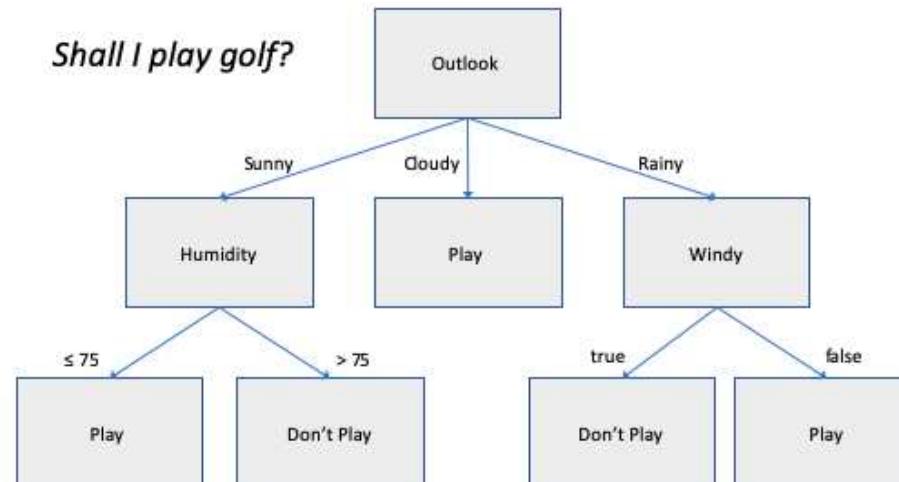
# Learning Objectives

- understanding of the principle functionality of decision trees
- understanding of the principle functionality of the naïve bayes method

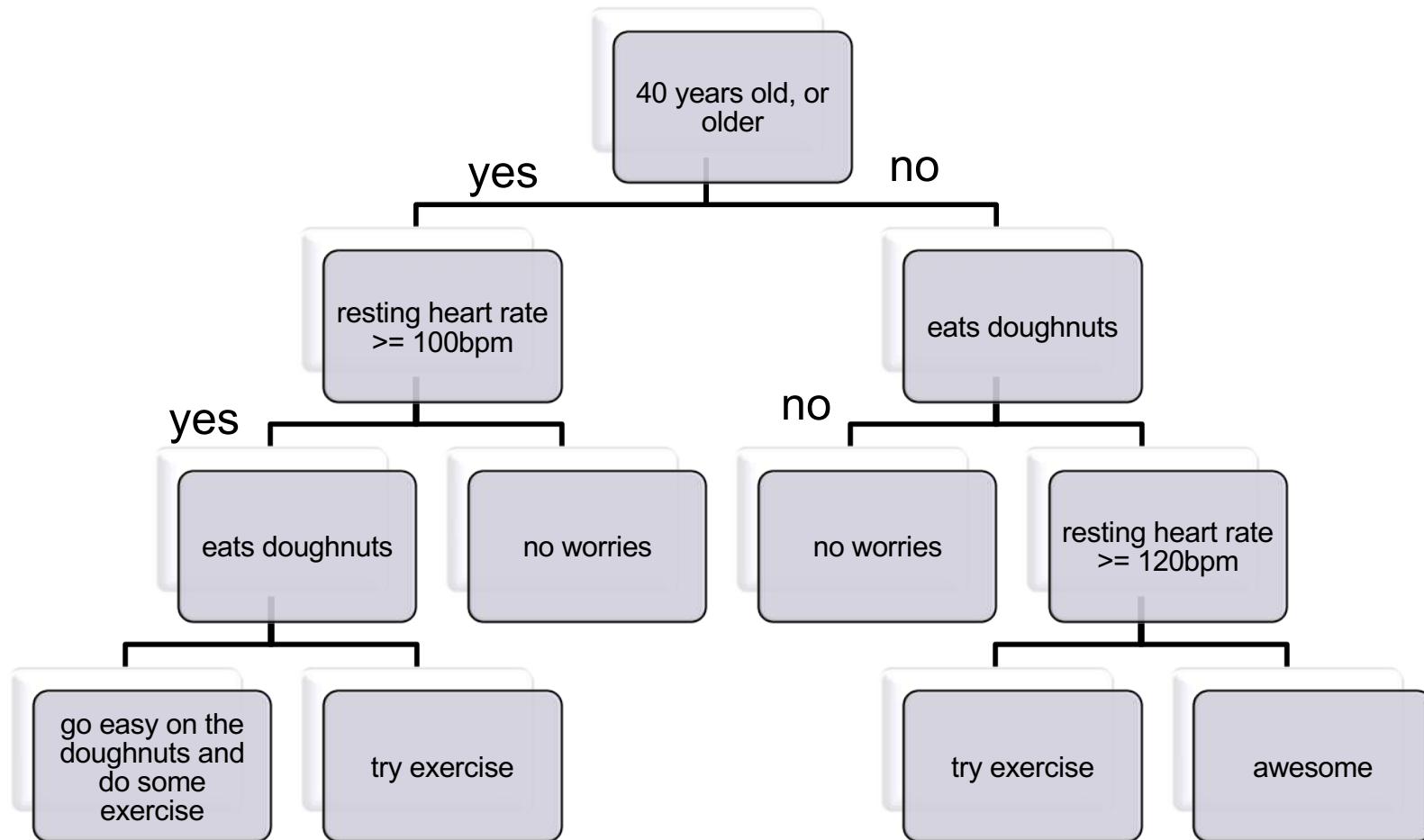


# Decision Trees

- a decision tree is a **flowchart-like structure**
  - each internal node represents a "test" on an attribute
  - each branch represents the outcome of the test
  - each leaf node represents a class label
  - the paths from root to leaf represent classification rules
- decision trees can be build on binary, categorical, numeric or ranked data

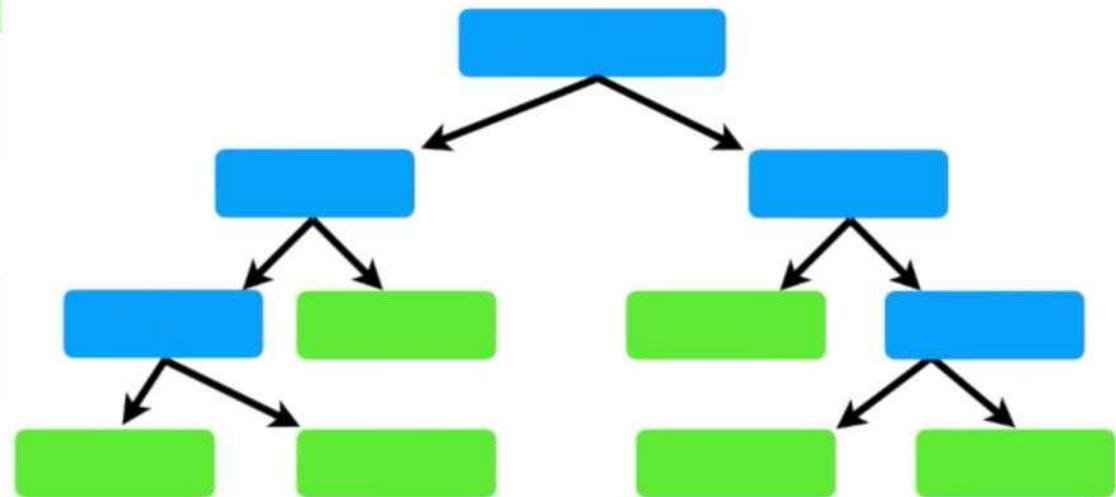


# Decision Trees



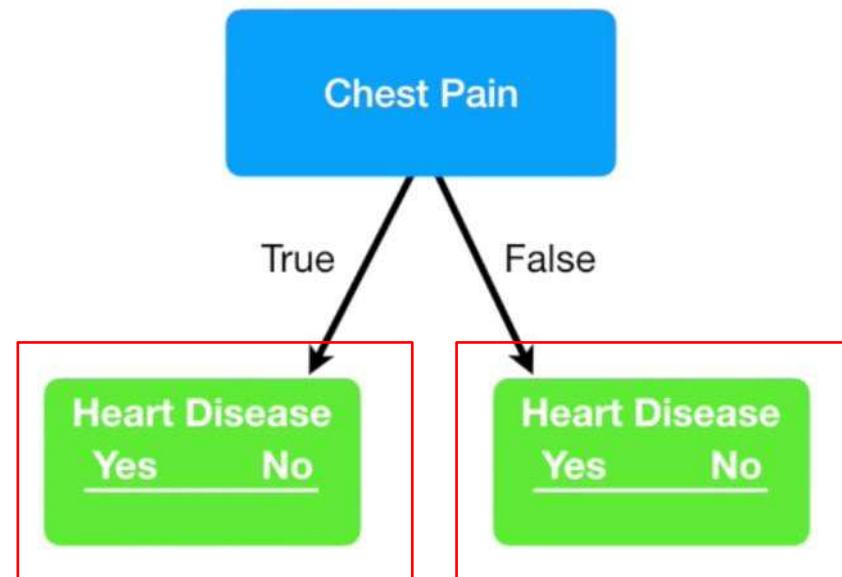
# Decision Trees

Chest Pain	Good Blood Circulation	Blocked Arteries	Heart Disease
No	No	No	No
Yes	Yes	Yes	Yes
Yes	Yes	No	No
Yes	No	???	Yes
etc...	etc...	etc...	etc...



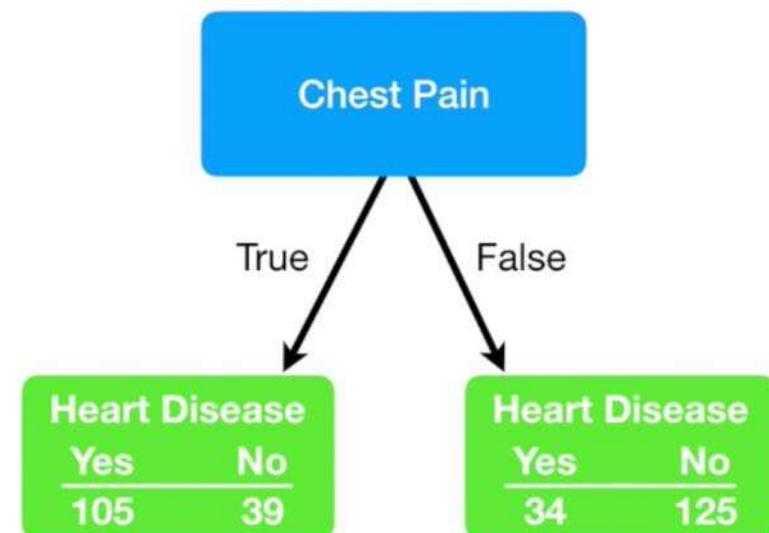
# Decision Trees

Chest Pain	Good Blood Circulation	Blocked Arteries	Heart Disease
No	No	No	No
Yes	Yes	Yes	Yes
Yes	Yes	No	No
Yes	No	???	Yes
etc...	etc...	etc...	etc...



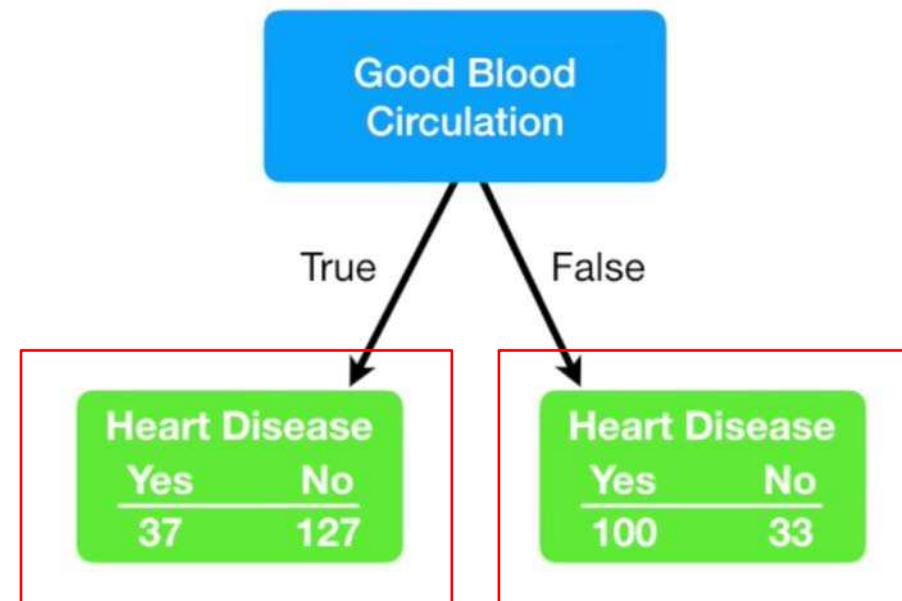
# Decision Trees

Chest Pain	Good Blood Circulation	Blocked Arteries	Heart Disease
No	No	No	No
Yes	Yes	Yes	Yes
Yes	Yes	No	No
Yes	No	???	Yes
etc...	etc...	etc...	etc...



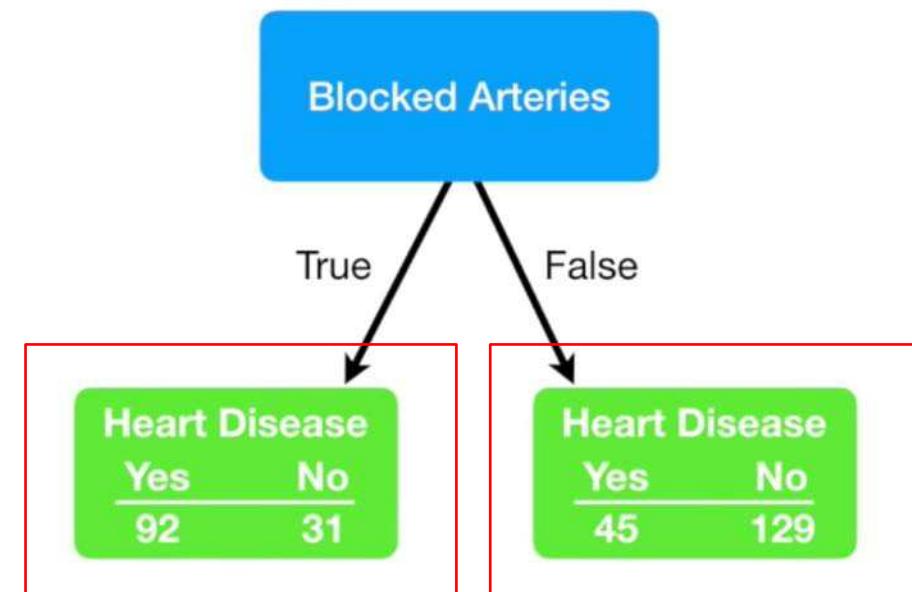
# Decision Trees

Chest Pain	Good Blood Circulation	Blocked Arteries	Heart Disease
No	No	No	No
Yes	Yes	Yes	Yes
Yes	Yes	No	No
Yes	No	???	Yes
etc...	etc...	etc...	etc...



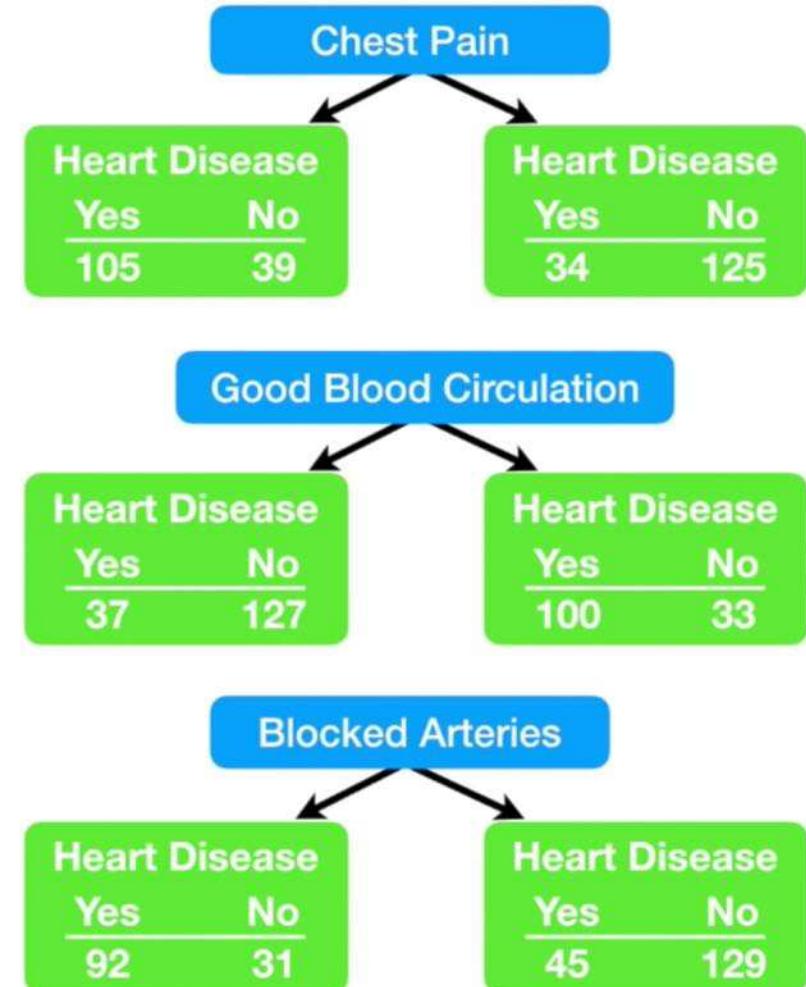
# Decision Trees

Chest Pain	Good Blood Circulation	Blocked Arteries	Heart Disease
No	No	No	No
Yes	Yes	Yes	Yes
Yes	Yes	No	No
Yes	No	???	Yes
etc...	etc...	etc...	etc...



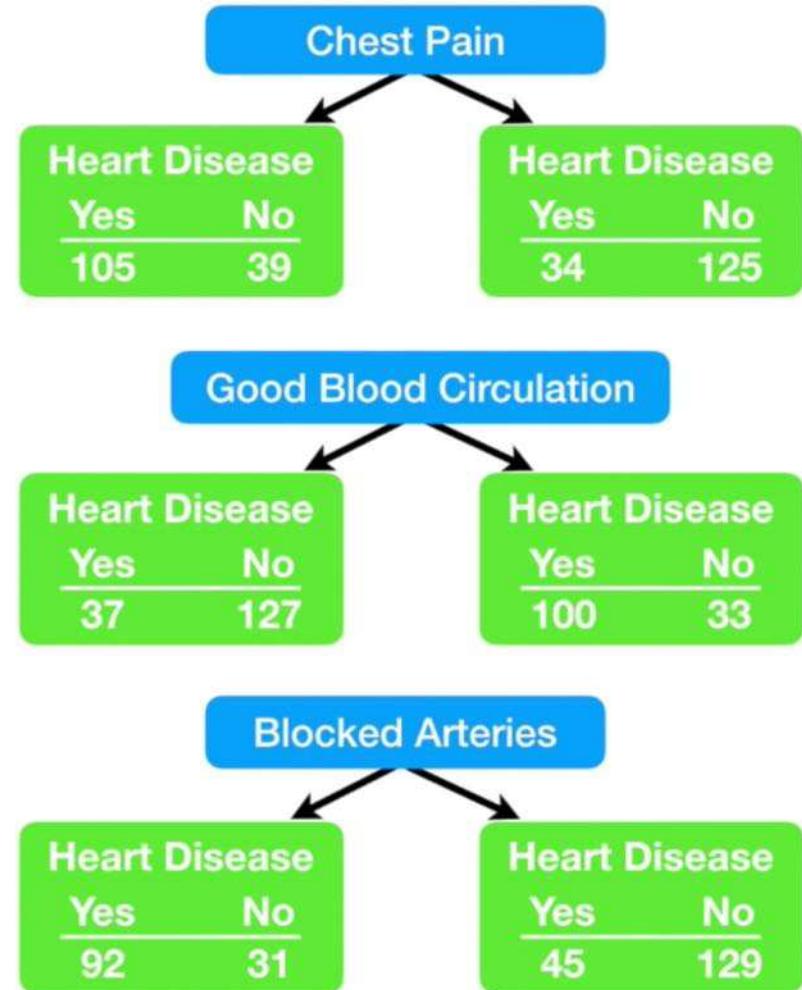
# Decision Trees

Chest Pain	Good Blood Circulation	Blocked Arteries	Heart Disease
No	No	No	No
Yes	Yes	Yes	Yes
Yes	Yes	No	No
Yes	No	???	Yes
etc...	etc...	etc...	etc...



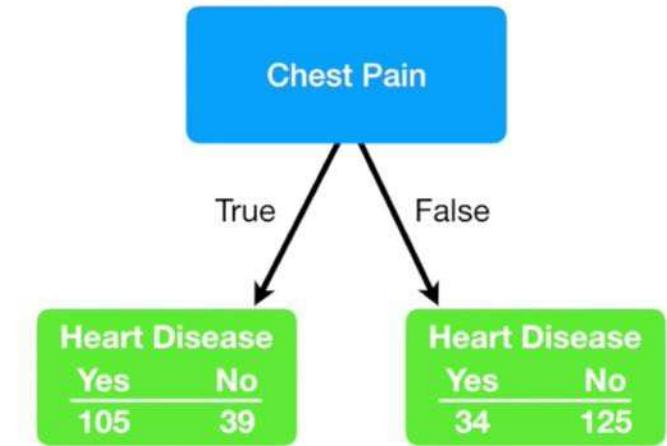
# Decision Trees

- because none of the leaf nodes is 100% “yes heart disease” or 100% “no heart disease”, they are all considered impure
- to determine which separation is best, we need a way to measure and compare impurity
- there are a bunch of ways to measure impurity, we focus on the very popular one called Gini



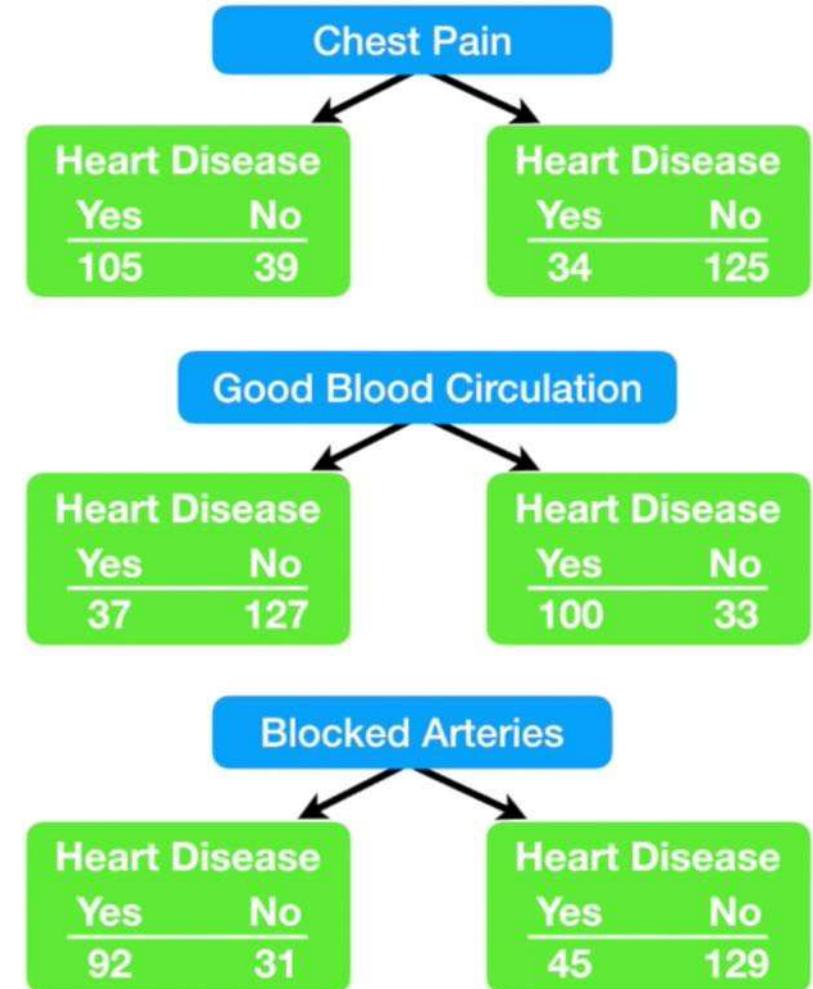
# Decision Trees

- Gini impurity =  $1 - (\text{the probability of "yes"})^2 - (\text{the probability of "no"})^2$
- for chest pain, true:
  - $1 - (105 / (105 + 39))^2 - (39 / (39 + 105))^2$
  - = 0.395
- for chest pain, false:
  - $1 - (34 / (34 + 125))^2 - (125 / (125 + 34))^2$
  - = 0.336
- total gini impurity of chest pain (weighted average)
  - $((144 / 144 + 159) * 0.395) + ((159 / 159 + 144) * 0.336)$
  - = 0.364



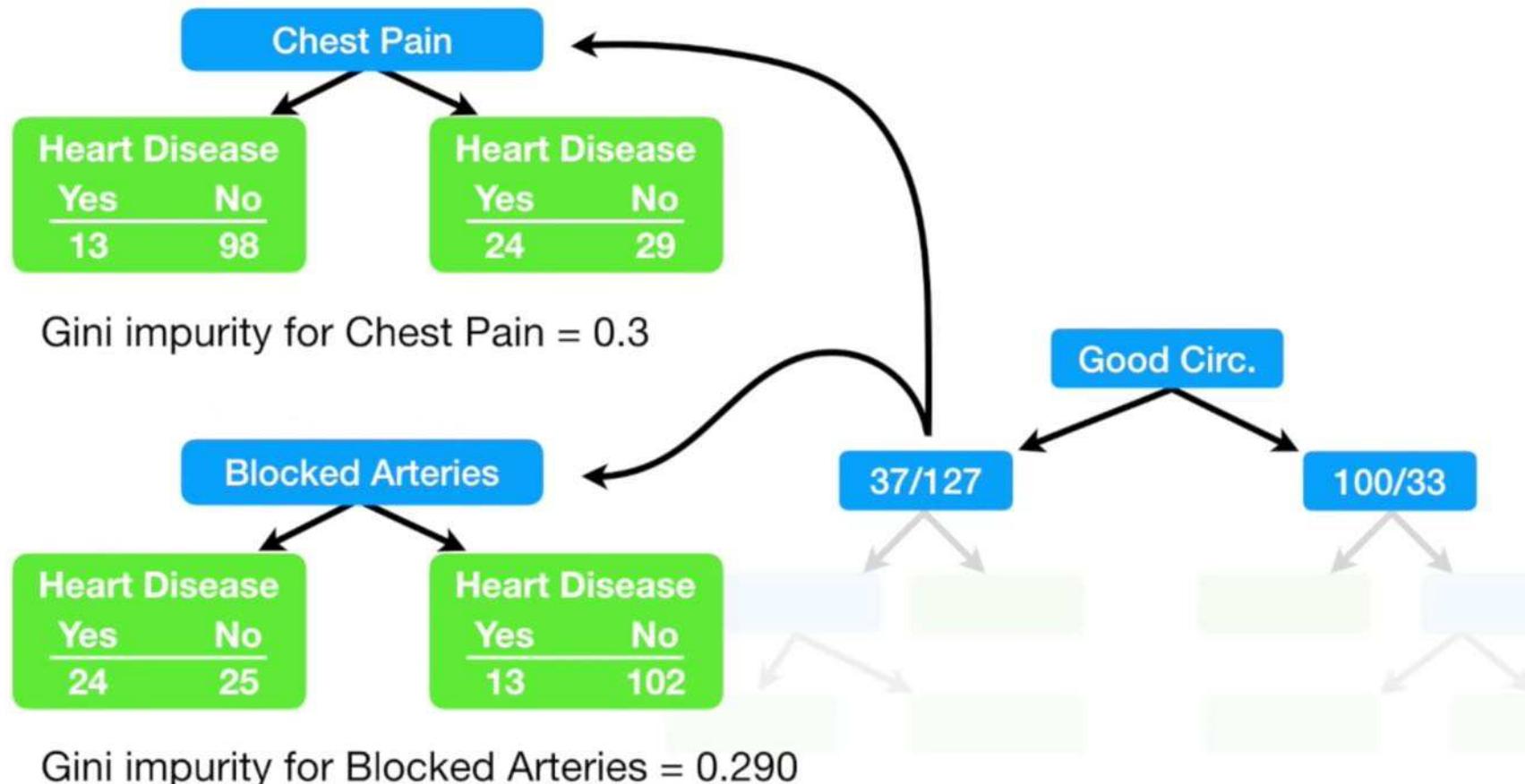
# Decision Trees

- gini impurity of chest pain
  - = 0.364
- gini impurity of good blood circulation
  - = 0.360
- gini impurity of blocked arteries
  - = 0.381
- good blood circulation has lowest impurity → best divider and root of the tree



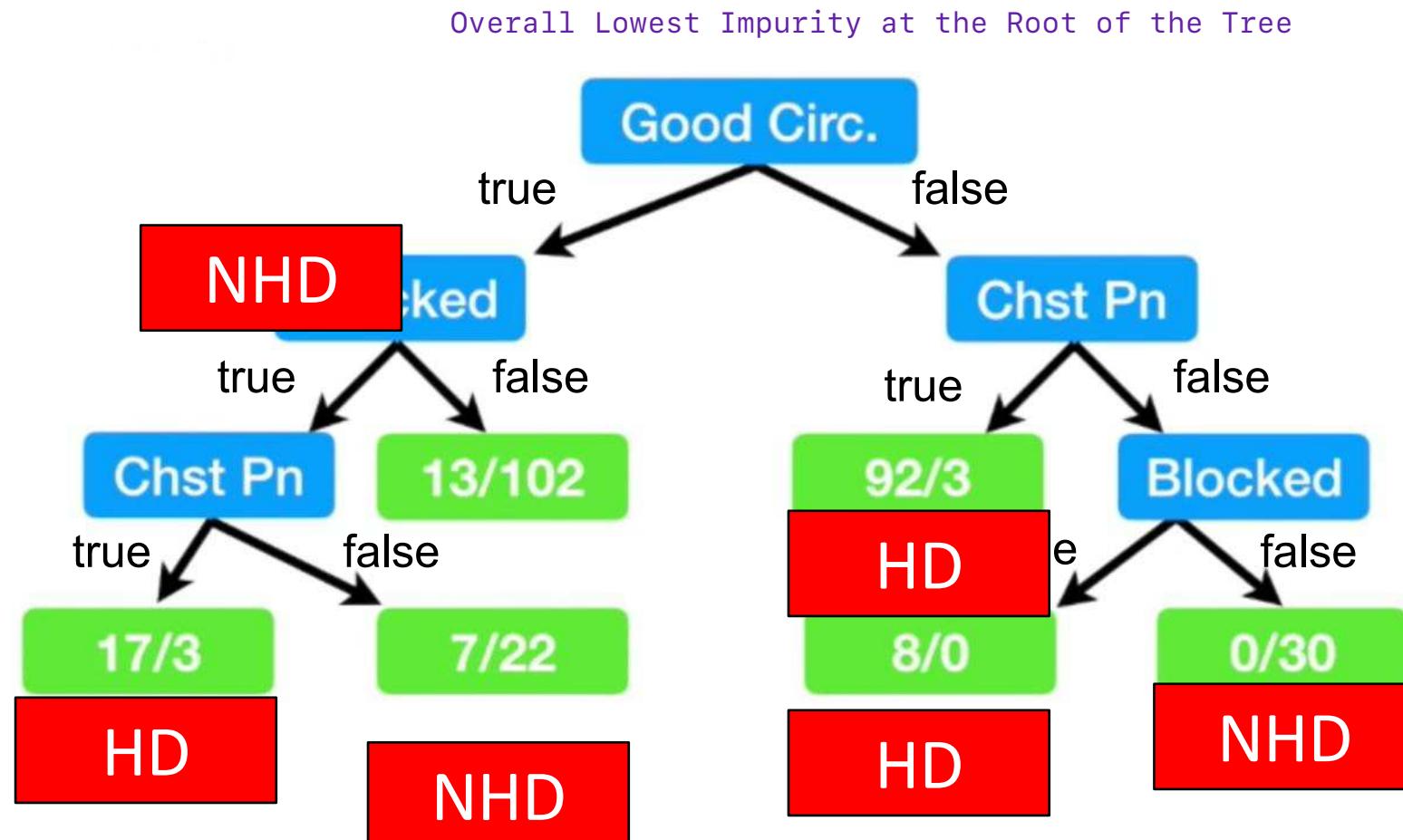
# Decision Trees

- repeat for next levels



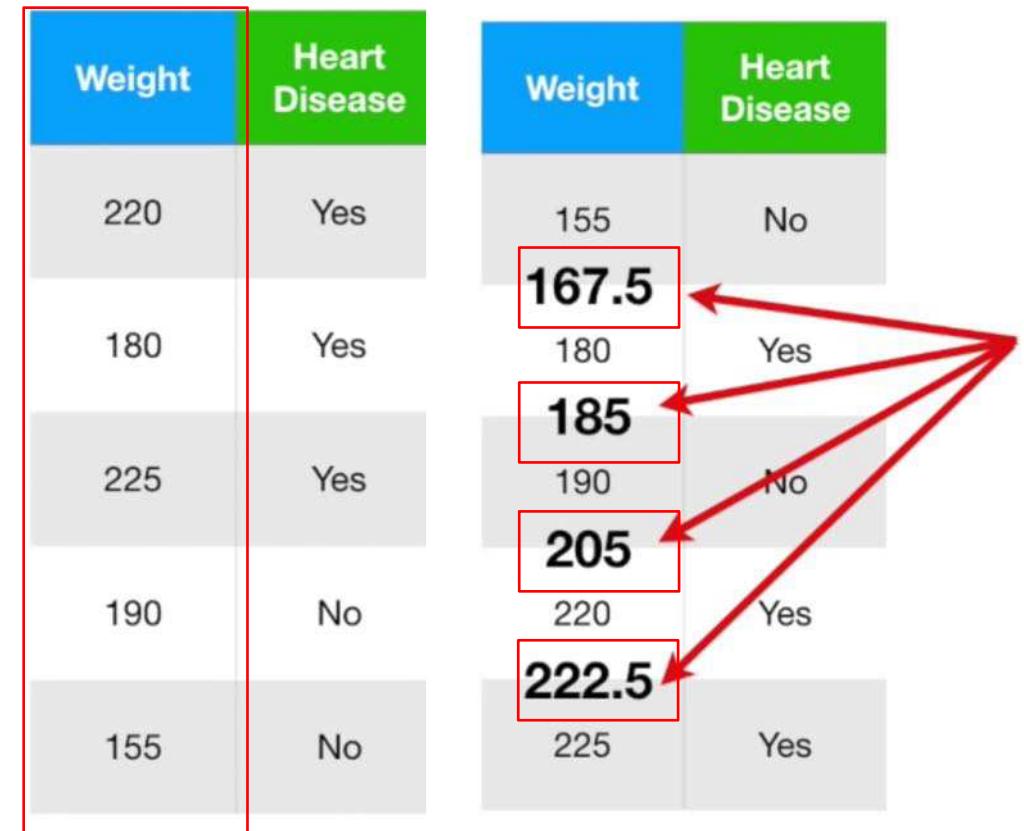
# Decision Trees

- final decision tree for binary data



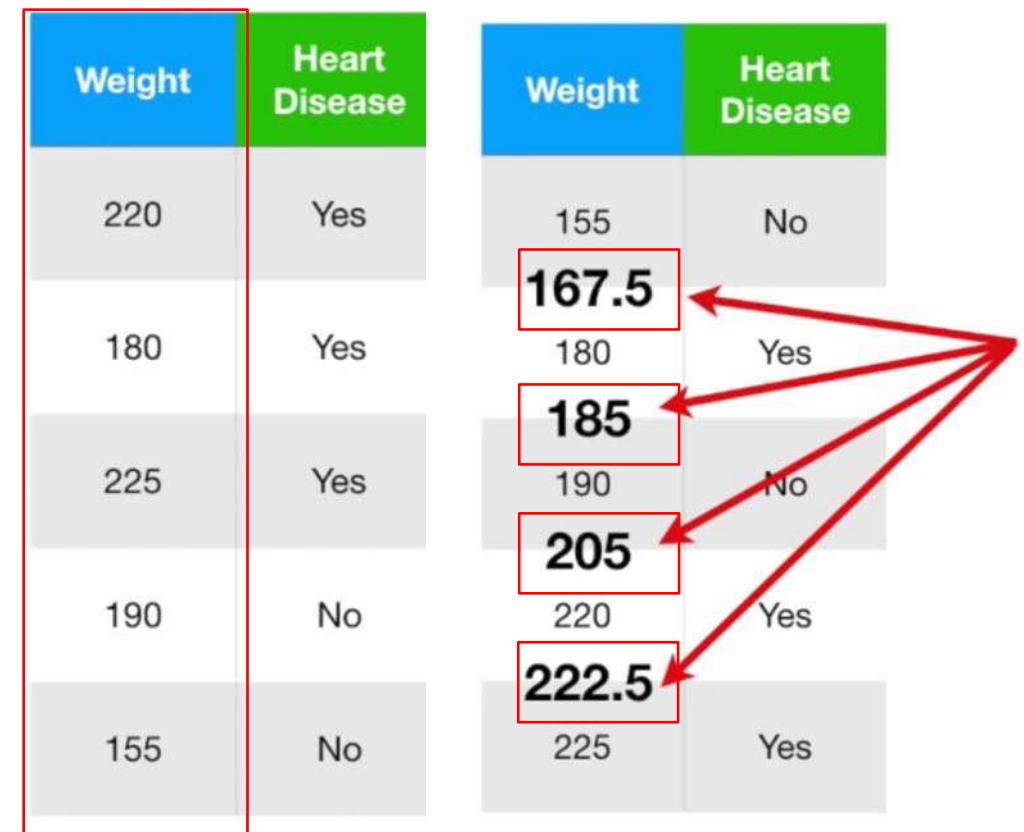
# Decision Trees

- how about numeric data?
  - sort the data from low to high
  - calculate the average weight for all adjacent patients
  - calculate the impurity values for each average weight



# Decision Trees

- 1. Klasse 167.5
  - Linke seite  $\leq 167.5$ 
    - 1 No, 0 Yes
  - Rechte seite  $> 167.5$ 
    - 20 No, 33 Yes
- 2. Klasse 185
  - Linke Seite  $\leq 185$ 
    - 1 No, 1 Yes
  - Rechte Seite  $> 185$ 
    - 20 No, 32 Yes



# Decision Trees

Weight	Heart Disease
155	No
<b>167.5</b>	Yes
180	Yes
<b>185</b>	No
<b>205</b>	Yes
220	Yes
<b>222.5</b>	Yes
225	Yes

Red arrows point from each row's weight to its Gini impurity value:

- 167.5 → Gini impurity = 0.3
- 185 → Gini impurity = 0.47
- 205 → Gini impurity = 0.27
- 222.5 → Gini impurity = 0.4

The lowest impurity occurs when we separate using **weight < 205...**

...so this is the cutoff and impurity value we will use when we compare weight to chest pain or blocked arteries.

# Decision Trees

- how about ranked data?
  - calculate the impurity scores for all possible ranks
  - e.g., for weather ranks 1 to 4 we get weather rank  $\leq 1$ , weather rank  $\leq 2$ , weather rank  $\leq 3$

rank the weather	play golf
1	yes
1	no
3	yes
1	yes
...	...

# Decision Trees

- how about multiple choice data?
  - calculate the **impurity score** for each one and all combinations
  - e.g., for **color choices blue, green and red** we get **color choice: blue, color choice: green, color choice: red, color choice: blue or green, color choice: blue or red and color choice: green or red**

color choice	play golf
green	yes
blue	no
red	yes
green	yes
...	...

# Decision Trees

- advantages
  - simple to understand and interpret
  - have value even with little hard data
  - support to determine worst, best and expected values for different scenarios
  - can be combined with other decision techniques and to form random forests
- disadvantages
  - they are unstable, a small change in the data can lead to a large change in the structure of the optimal decision tree
  - they are often relatively inaccurate, other predictors perform better with similar data
  - calculations can get very complex, particularly if many values are uncertain and/or if many outcomes are linked

# Naïve Bayes

- Naïve Bayes classifiers are a family of simple "probabilistic classifiers" based on applying Bayes' theorem with strong (naïve) independence assumptions between the features
- Bayes' theorem
  - $\mathbf{X}$  is a data tuple
  - $\mathbf{H}$  is some hypothesis, such as that  $\mathbf{X}$  belongs to a specified class  $\mathbf{C}$
  - $P(\mathbf{H}|\mathbf{X})$  is the posterior probability of  $\mathbf{H}$  conditioned on  $\mathbf{X}$
  - $P(\mathbf{C}_i|\mathbf{X})$  is the probability that tuple  $\mathbf{X}$  belongs to class  $\mathbf{C}_i$ , given that we know the attribute description of  $\mathbf{X}$



Thomas Bayes

# Naïve Bayes

- Bayes' theorem

- data tuple has attribute age and income
- $X$  is a 35-year-old customer with an income of 40.000 €
- $H$  is the hypothesis that our customer will buy a computer
- $P(H|X)$  reflects the probability that customer  $X$  will buy a computer given that we know his age and income
- $P(H)$  is the prior probability of  $H$ . The probability that any given customer will buy a computer, regardless of age or income
- $P(X)$  is the probability that a customer is 35-years-old and earns 40.000 €
- $P(X|H)$  reflects the probability that customer  $X$  is 35-years-old and earns 40.000 € given that we know he will buy a computer

# Naïve Bayes

- Bayes' theorem
  - provides a way of calculating the posterior probability  $P(H|X)$  from  $P(H)$ ,  $P(X|H)$ ,  $P(X)$

$$P(H|X) = \frac{P(X|H)P(H)}{P(X)}$$

- example: posterior probability
  - a school with 60% boys, 40% girls
  - 50% of girls wear trousers the others skirts
  - 100% of boys wear trousers
  - what is the probability of a random student wearing trousers to be a girl?

# Naïve Bayes

- example: posterior probability
  - $P(G)$ , probability that the student is a girl => 0.4 (40%)
  - $P(T)$ , probability that the student is wearing trousers =>  
 $0.5 * 0.4 + 1 * 0.6 = 0.8$
  - $P(T|G)$ , probability of the student wearing trousers given that the student is a girl => 0.5 (50%)
  - probability of a random student wearing trousers to be a girl:

$$P(G|T) = \frac{P(T|G)P(G)}{P(T)} = \frac{0.5 * 0.4}{0.8} = 0.25$$

# Naïve Bayes

- Bayes classifier

- let  $D$  be a training set of tuples and their associated class labels
- each tuple is represented by an  $n$ -dimensional attribute vector  $X$
- there are  $m$  classes  $C_1, C_2, \dots, C_m$
- given  $X$ , the classifier will predict that  $X$  belongs to the class having the highest posterior probability;  $P(C_i|X) > P(C_j|X)$

$$P(H|X) = \frac{P(X|H)P(H)}{P(X)} \rightarrow P(C_i|X) = \frac{P(X|C_i)P(C_i)}{P(X)}$$

# Naïve Bayes

- Bayes classifier
  - given  $X$ , the classifier will predict that  $X$  belongs to the class having the highest posterior probability;  $P(C_i|X) > P(C_j|X)$

$$\frac{P(X|C_i)P(C_i)}{P(X)} > \frac{P(X|C_j)P(C_j)}{P(X)}$$

- assumptions
  - $P(X)$  is constant for all classes =>  $P(X | C_i) * P(C_i)$  must be maximized
  - if the class prior probabilities are not known we assume they are equally likely =>  $P(C_1) = P(C_2) = \dots = P(C_m)$
  - therefore, we maximize  $P(X | C_i)$
  - for availability, computational, and storage reasons, we assume class conditional independence

$$P(X|C_i) = P(x_1|C_i) \times P(x_2|C_i) \times \dots \times P(x_n|C_i)$$

# Naïve Bayes

- Bayes classifier
  - assumptions
    - $P(X)$  is constant for all classes =>  $P(X|C_i)P(C_i)$  must be maximized
    - if the class prior probabilities are not known we assume they are equally likely =>  $P(C_1) = P(C_2) = \dots = P(C_m)$
    - therefore, we maximize  $P(X|C_i)$
    - for availability, computational, and storage reasons, we assume class conditional independence

$$P(X | C_i) = P(x_1 | C_i) \times P(x_2 | C_i) \times \dots \times P(x_n | C_i)$$

- $P(x_1 | C_i)$  is the number of tuples of class  $C_i$  in  $D$  having the value  $x_k$  for  $A_k$  divided by  $|C_i, D|$ , the number of tuples of class  $C_i$  in  $D$

# Naïve Bayes

- example

RID	age	income	student	credit_rating	buys_computer
1	youth	high	no	fair	no
2	youth	high	no	excellent	no
3	middle	high	no	fair	yes
4	senior	medium	no	fair	yes
5	senior	low	yes	fair	yes
6	senior	low	yes	excellent	no
7	middle	low	yes	excellent	yes
8	youth	medium	no	fair	no
9	youth	low	yes	fair	yes
10	senior	medium	yes	fair	yes
11	youth	medium	yes	excellent	yes
12	middle	medium	no	excellent	yes
13	middle	high	yes	fair	yes
14	senior	medium	no	excellent	no

# Naïve Bayes

- example

- step 1 (compute  $P(C_i)$ ):
  - $C_1: \text{buys\_computer} = \text{yes}; C_2: \text{buys\_computer} = \text{no}$
  - $X = (\text{age} = \text{youth}; \text{income} = \text{medium}; \text{student} = \text{yes}; \text{credit\_rating} = \text{fair})$
  - need to maximize  $P(X | C_i) * P(C_i)$ , for  $i = 1, 2$
  - compute  $P(C_i)$ 
    - $P(\text{buys\_computer} = \text{yes}) = 9/14 = 0.643$
    - $P(\text{buys\_computer} = \text{no}) = 5/14 = 0.357$

# Naïve Bayes

- example

- step 2 (compute  $P(X|C_i)$ ):
  - $X = (\text{age} = \text{youth}; \text{income} = \text{medium}; \text{student} = \text{yes}; \text{credit\_rating} = \text{fair})$
  - $P(\text{age} = \text{youth}|\text{buys\_computer} = \text{yes}) = 2/9 = 0.222$
  - $P(\text{age} = \text{youth}|\text{buys\_computer} = \text{no}) = 3/5 = 0.600$
  - $P(\text{income} = \text{medium}|\text{buys\_computer} = \text{yes}) = 4/9 = 0.444$
  - $P(\text{income} = \text{medium}|\text{buys\_computer} = \text{no}) = 2/5 = 0.400$
  - $P(\text{student} = \text{yes}|\text{buys\_computer} = \text{yes}) = 6/9 = 0.667$
  - $P(\text{student} = \text{yes}|\text{buys\_computer} = \text{no}) = 1/5 = 0.200$
  - $P(\text{credit\_rating} = \text{fair}|\text{buys\_computer} = \text{yes}) = 6/9 = 0.667$
  - $P(\text{credit\_rating} = \text{fair}|\text{buys\_computer} = \text{no}) = 2/5 = 0.400$

# Naïve Bayes

- example

- step 2 (compute  $P(X|C_i)$ ):

- $P(X|buys\_computer = yes) =$   
 $P(age = youth|buys\_computer = yes) \times$   
 $P(income = medium|buys\_computer=yes) \times$   
 $P(student = yes|buys\_computer=yes) \times$   
 $P(credit\_rating = fair|buys\_computer=yes)$

$$= 0.044$$

- $P(X|buys\_computer = no) =$   
 $P(age = youth|buys\_computer=no) \times$   
 $P(income = medium|buys\_computer=no) \times$   
 $P(student = yes|buys\_computer=no) \times$   
 $P(credit\_rating = fair|buys\_computer=no)$

$$= 0.019$$

diese Werte wurden  
im vorherigen  
Schritt berechnet

# Naïve Bayes

- example

- step 3 (compute  $P(X|C_i)P(C_i)$ ):
  - $P(X|buys\_computer = yes) \times P(buys\_computer = yes) = 0.044 \times 0.643 = 0.028$
  - $P(X|buys\_computer = no) \times P(buys\_computer = no) = 0.019 \times 0.357 = 0.007$
  - the classifier predicts  $buys\_computer = yes$  for tuple X

# Naïve Bayes

- advantages
  - fast to train and classify
  - performance is similar to decision trees and neural networks
  - easy to implement
  - handles numeric and categorical data
  - useful for very large data sets (no iterative parameter estimation)
- disadvantages
  - assumes class conditional independence, therefore, loss of accuracy
  - model is difficult to interpret

# Lessons Learned

- understanding of the principle functionality of decision trees
- understanding of the principle functionality of the naïve bayes method



# Questions and Answers



# Machine Learning and Data Mining

## V10: Partitioning Clustering

Lecturer: Dr. Andreas Weiler



# Repetition

... REPETITION ...

→ IS THE MOTHER →

of learning

# Course Content IT20a\_WIN

Week	Lecture	Lab
KW08	Introduction	Lab 01
KW09	Data Processing Part 01	Lab 02
KW10	Carnival Monday	
KW11	Data Processing Part 02	Lab 04
KW12	Evaluation	Lab 05
KW13	Recommender Systems	<b>Graded Lab 01</b> <span style="border: 1px solid red; padding: 2px;">IMPORTANT</span>
KW14	Association Rules	<b>Graded Lab 01</b> <span style="border: 1px solid red; padding: 2px;">IMPORTANT</span>
KW15	Linear and Logistic Regression	Lab 06
KW16	Easter Monday	
KW17	Support Vector Machines	<b>Graded Lab 02</b> <span style="border: 1px solid red; padding: 2px;">IMPORTANT</span>
KW18	Decision Trees and Naïve Bayes	<b>Graded Lab 02</b> <span style="border: 1px solid red; padding: 2px;">IMPORTANT</span>
KW19	Partition Clustering	Lab 08
KW20	Hierarchical and Density Clustering	Lab 09
KW21	Neural Networks and Closing	Review and Questions for Exam

# Learning Objectives

- understanding of different partitioning clustering techniques
- understanding of measuring the quality of a clustering result

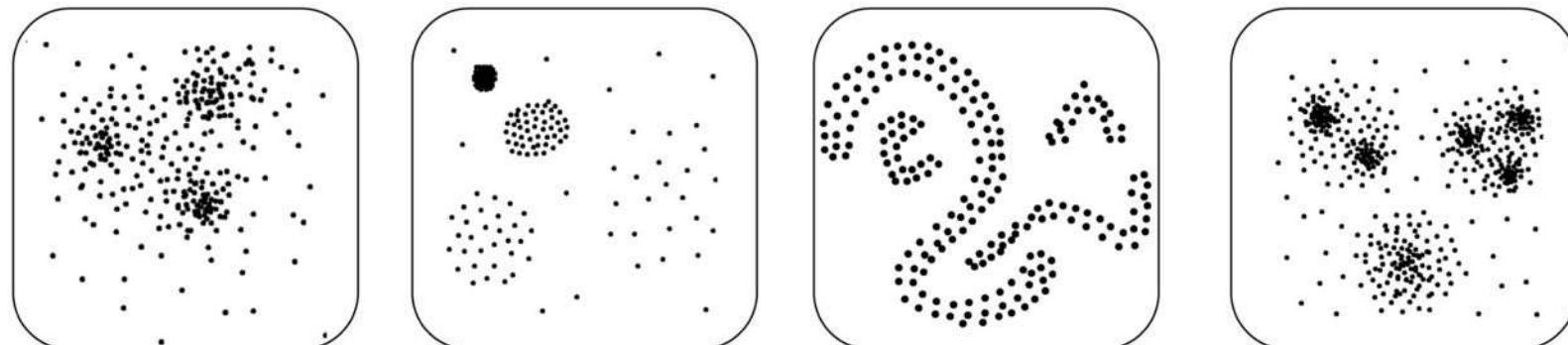


# Clustering

- clustering is the task of grouping a set of objects in such a way that objects in the same group (cluster) are more similar (in some sense) to each other than to those in other groups
- typical cluster models include:
  - connectivity models: for example, hierarchical clustering builds models based on distance connectivity
  - centroid models: for example, the k-means algorithm represents each cluster by a single mean vector
  - distribution models: clusters are modeled using statistical distributions, such as multivariate normal distributions used by the expectation-maximization algorithm
  - density models: for example, DBSCAN and OPTICS defines clusters as connected dense regions in the data space

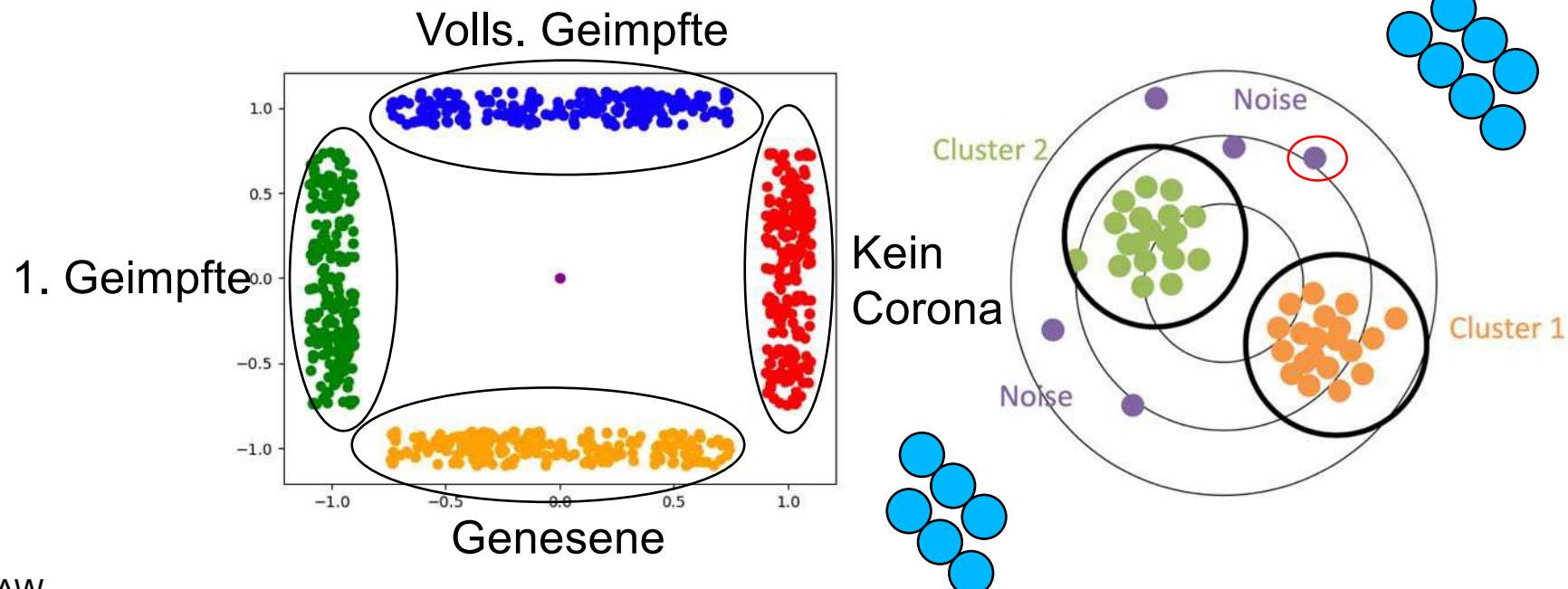
# Clustering

- clustering identifies a finite set of categories, classes or groups  $C_1 \dots C_k$  in the dataset such that:
  - objects within the same cluster  $C_i$  shall be as similar as possible
  - objects of different clusters  $C_i, C_j$  shall be as dissimilar as possible
- properties of clusters
  - clusters may have different sizes, shapes, densities
  - clusters may form a hierarchy
  - clusters may be overlapping or disjoint



# Clustering

- goals of cluster analysis
  - data **understanding**: find „natural“ clusters and **describe their properties**
  - data **class identification**: find **useful** and **suitable groupings**
  - data **reduction**: find **representatives** for **homogenous groups**
  - **outlier detection**: find **unusual data objects**
  - **noise detection**: find **random perturbation** of the data

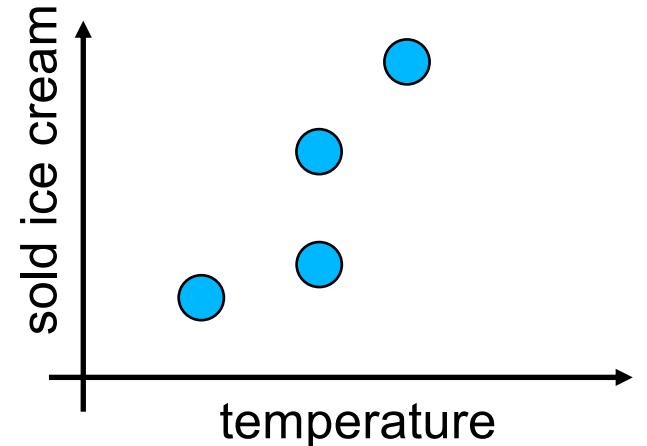


# Clustering

- application scenarios
  - marketing
    - large database with customer data containing their properties and past buying records
    - find groups of customers with similar behavior
  - CAD databases
    - large database of CAD data containing abstract feature vectors
    - find homogenous groups of similar CAD parts
    - determine standard parts for each group
    - find standard parts instead of special parts
  - text analysis
    - structuring large corpora of text documents
    - use a hierarchical clustering of the text documents

# Clustering

- **similarity of clusters**
  - distance function calculates  $\text{dist}(o_1, o_2)$  for pairs of objects
  - small distance => similar objects
  - large distance => dissimilar objects
- **requirements for distance functions**
  - $\text{dist}(o_1, o_2) = d \in \mathbb{R} \geq 0$
  - $\text{dist}(o_1, o_2) = 0$  iff  $o_1 = o_2$
  - $\text{dist}(o_1, o_2) = \text{dist}(o_2, o_1)$
  - for metric distance functions  $\text{dist}(o_1, o_3) \leq \text{dist}(o_1, o_2) + \text{dist}(o_2, o_3)$

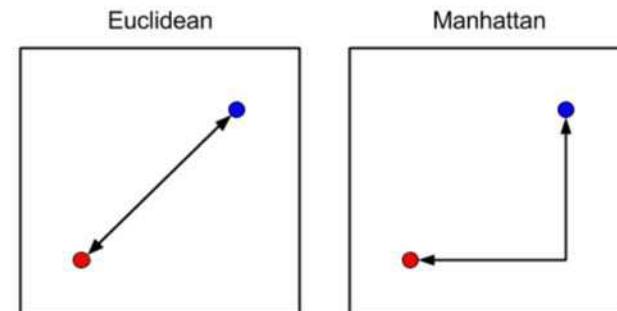


euclidean

$$\text{dist}(x, y) = \sqrt{\sum_{i=1}^d (x_i - y_i)^2}$$

manhattan

$$\text{dist}(x, y) = \sum_{i=1}^d |x_i - y_i|$$



# K-means clustering

- aims to partition  $n$  observations into  $k$  clusters in which each observation belongs to the cluster with the nearest mean

**Input:**  $D = \{p_1, \dots, p_n\}$  (Points to be clustered),  
 $k$  (number of clusters)

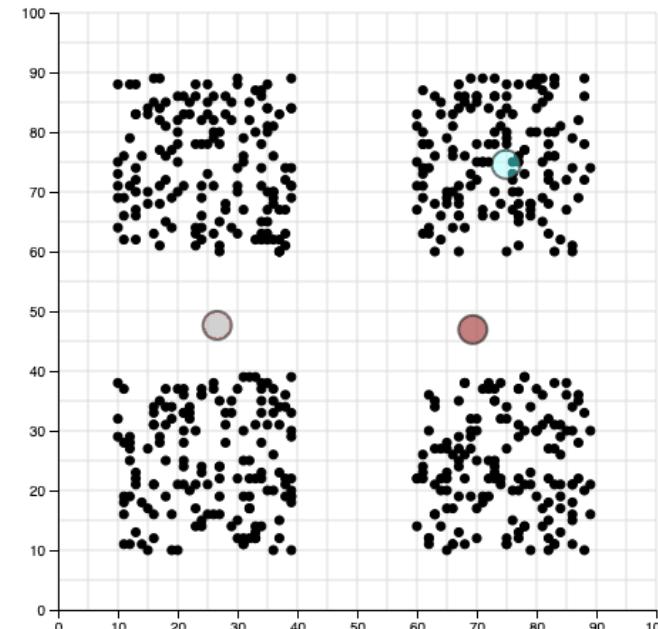
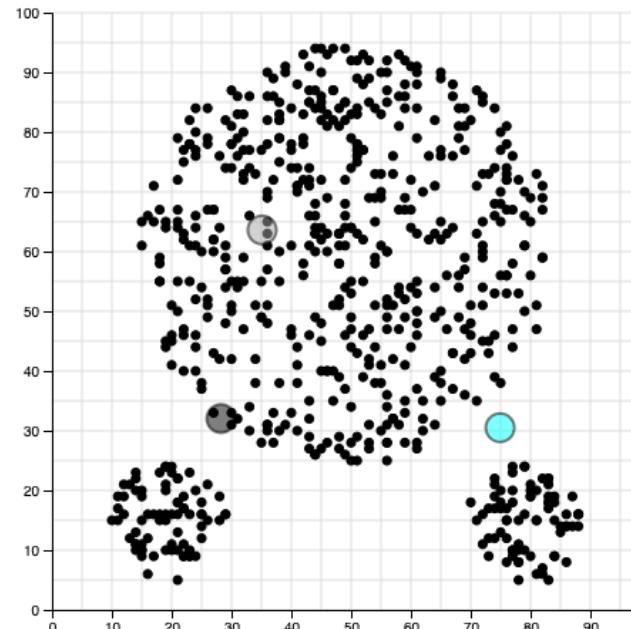
**Output:**  $C = \{c_1, \dots, c_k\}$  (cluster centroids)  
 $m: D \rightarrow \{1, \dots, k\}$  (cluster membership)

## Method

- (1) Arbitrarily choose  $k$  objects from  $D$  as the initial cluster centers;
- (2) repeat**
- (3) (re)assign each object to the cluster to which the object is the most similar, based on the mean value of the objects in the cluster;
- (4) update the cluster means, i.e., calculate the mean value of the objects for each cluster;
- (5) until** no change;

# K-means clustering

- examples



## Pseudocode: k-means

**Complexity Range:**  $O(k \times n \times t)$

**Input:**  $k$  clusters

**Output:**  $k$  clusters

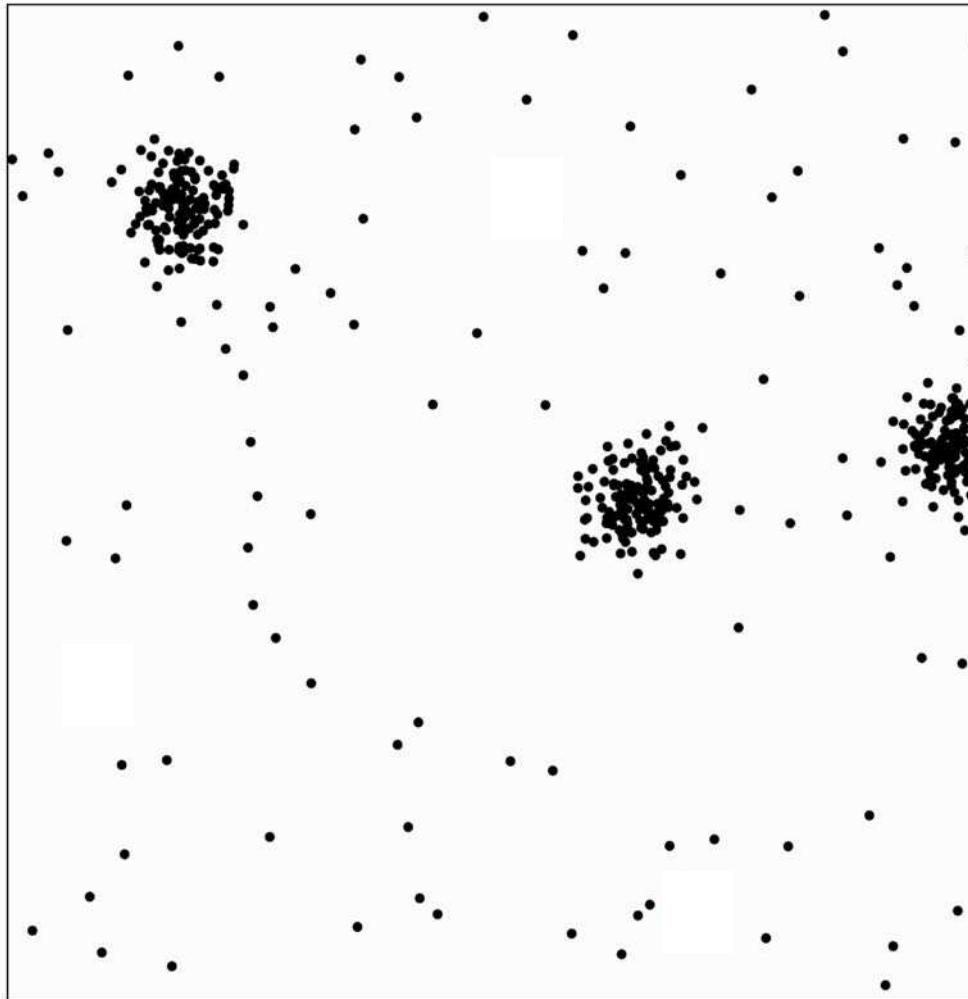
### Pseudocode:

1. Choose  $k$  objects as initial **cluster centers**.
2. Assign each data point to the cluster which has the closest **mean point (centroid)** under chosen distance metric.
3. When all data points have been assigned, recalculate the positions of  $k$  **centroids (mean points)**.
4. Repeat steps 2 and 3 until the **centroids** do not change any more. All data points remain in their most recently assigned cluster.

<https://educlust.dbvis.de/>

# K-means clustering

- example



## Pseudocode: k-means

**Complexity Range:**  $O(k \times n \times t)$

**Input:**  $k$  clusters

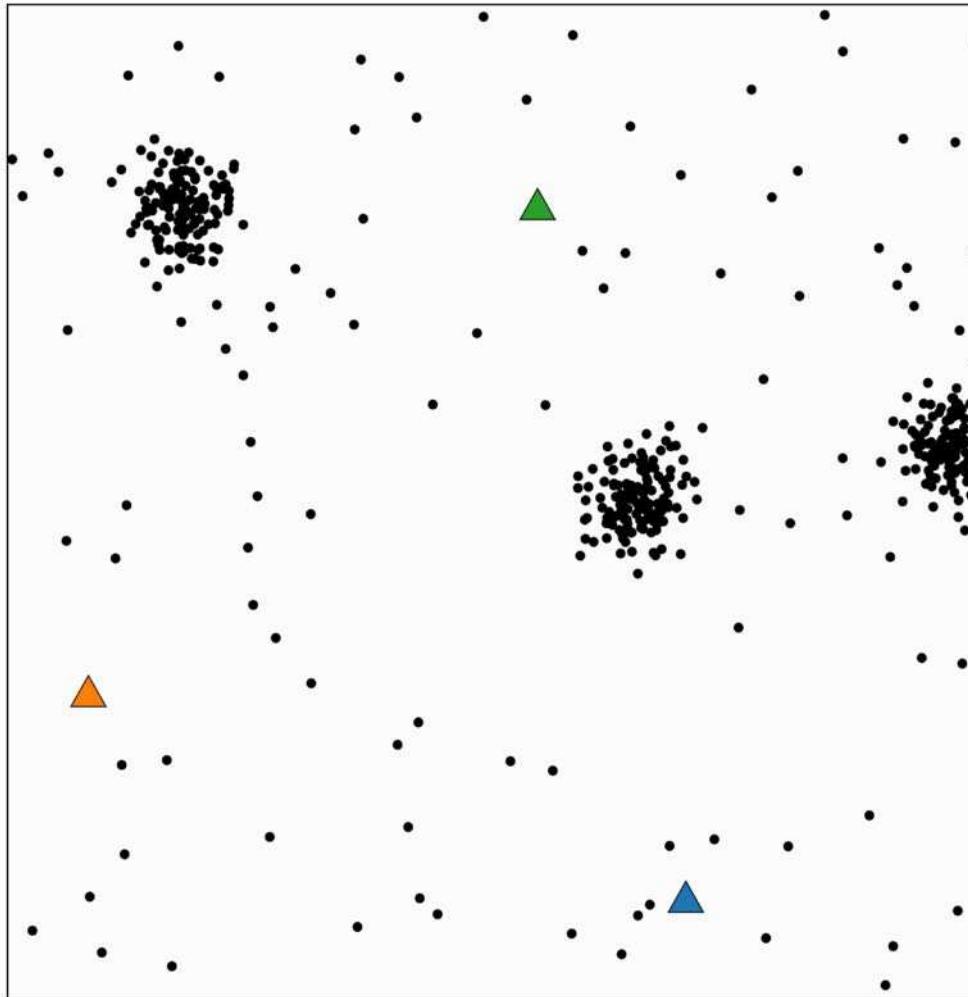
**Output:**  $k$  clusters

### Pseudocode:

1. Choose  $k$  objects as initial **cluster centers**.
2. Assign each data point to the cluster which has the closest **mean point (centroid)** under chosen distance metric.
3. When all data points have been assigned, recalculate the positions of  **$k$  centroids (mean points)**.
4. Repeat steps 2 and 3 until the **centroids** do not change any more. All data points remain in their most recently assigned cluster.

# K-means clustering

- example



## Pseudocode: k-means

**Complexity Range:**  $O(k \times n \times t)$

**Input:**  $k$  clusters

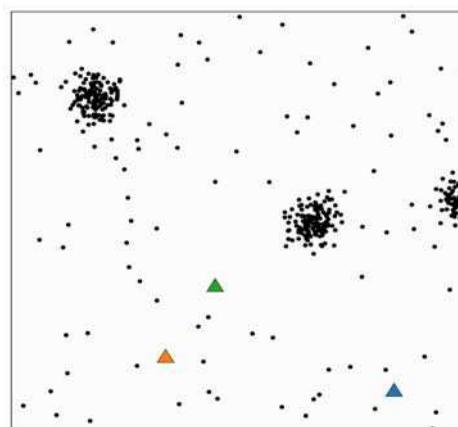
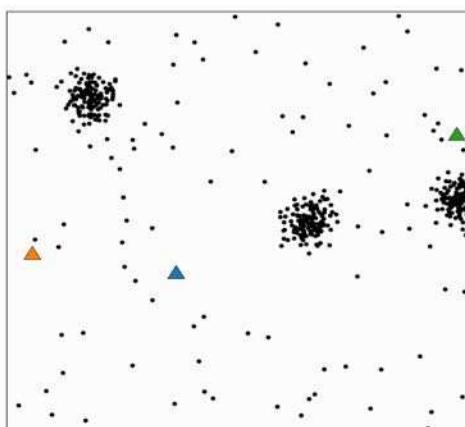
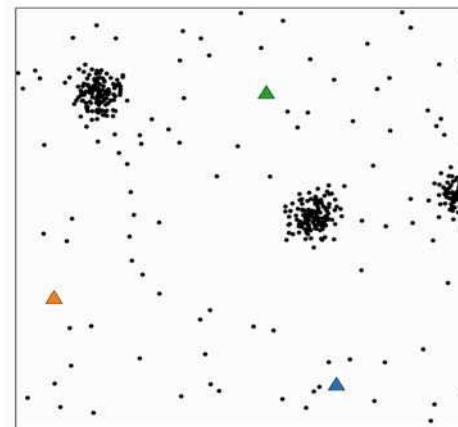
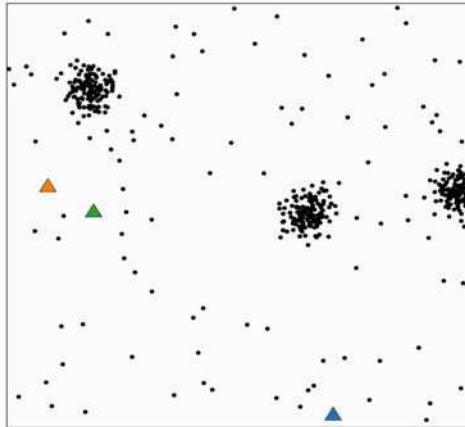
**Output:**  $k$  clusters

### Pseudocode:

1. Choose  $k$  objects as initial **cluster centers**.
2. Assign each data point to the cluster which has the closest **mean point (centroid)** under chosen distance metric.
3. When all data points have been assigned, recalculate the positions of  **$k$  centroids (mean points)**.
4. Repeat steps 2 and 3 until the **centroids** do not change any more. All data points remain in their most recently assigned cluster.

# K-means clustering

- example



## Pseudocode: k-means

**Complexity Range:**  $O(k \times n \times t)$

**Input:**  $k$  clusters

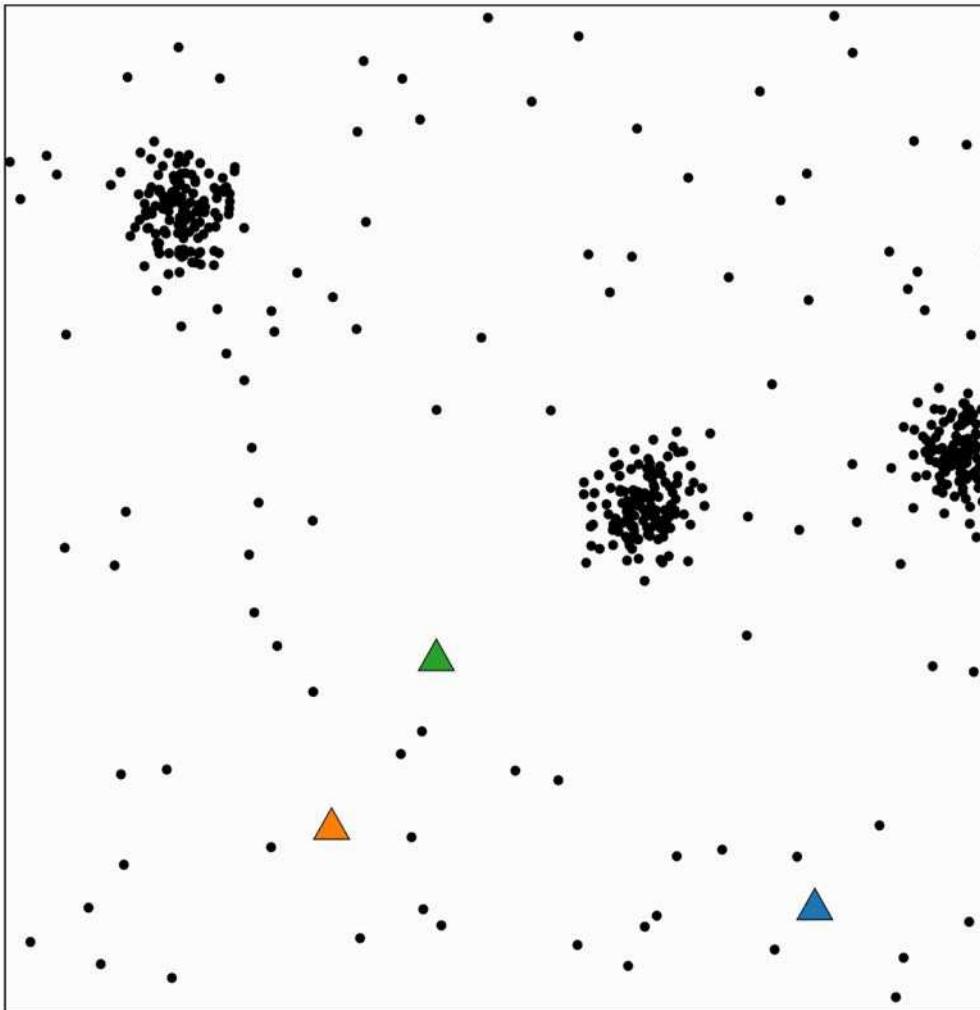
**Output:**  $k$  clusters

### Pseudocode:

1. Choose  $k$  objects as initial **cluster centers**.
2. Assign each data point to the cluster which has the closest **mean point (centroid)** under chosen distance metric.
3. When all data points have been assigned, recalculate the positions of  **$k$  centroids (mean points)**.
4. Repeat steps 2 and 3 until the **centroids** do not change any more. All data points remain in their most recently assigned cluster.

# K-means clustering

- example



## Pseudocode: k-means

**Complexity Range:**  $O(k \times n \times t)$

**Input:**  $k$  clusters

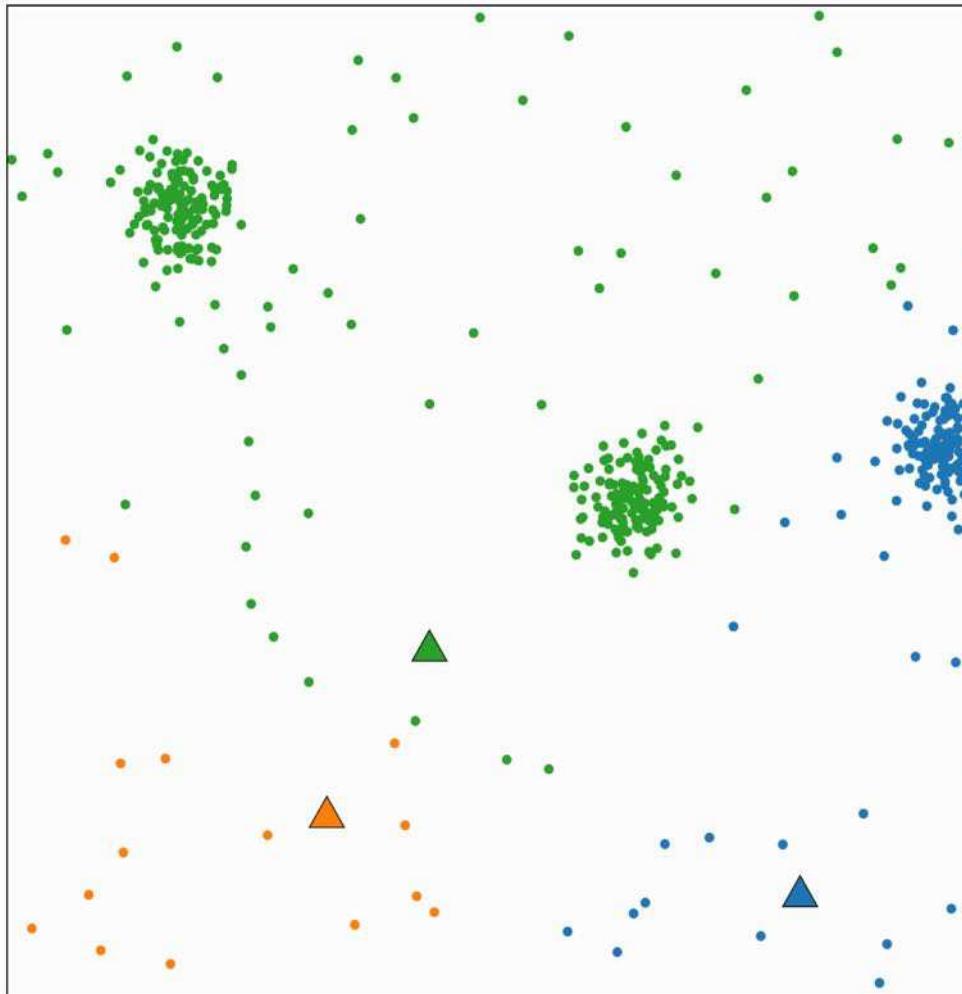
**Output:**  $k$  clusters

### Pseudocode:

1. Choose  $k$  objects as initial **cluster centers**.
2. Assign each data point to the cluster which has the closest **mean point (centroid)** under chosen distance metric.
3. When all data points have been assigned, recalculate the positions of  **$k$  centroids (mean points)**.
4. Repeat steps 2 and 3 until the **centroids** do not change any more. All data points remain in their most recently assigned cluster.

# K-means clustering

- example



## Pseudocode: k-means

**Complexity Range:**  $O(k \times n \times t)$

**Input:**  $k$  clusters

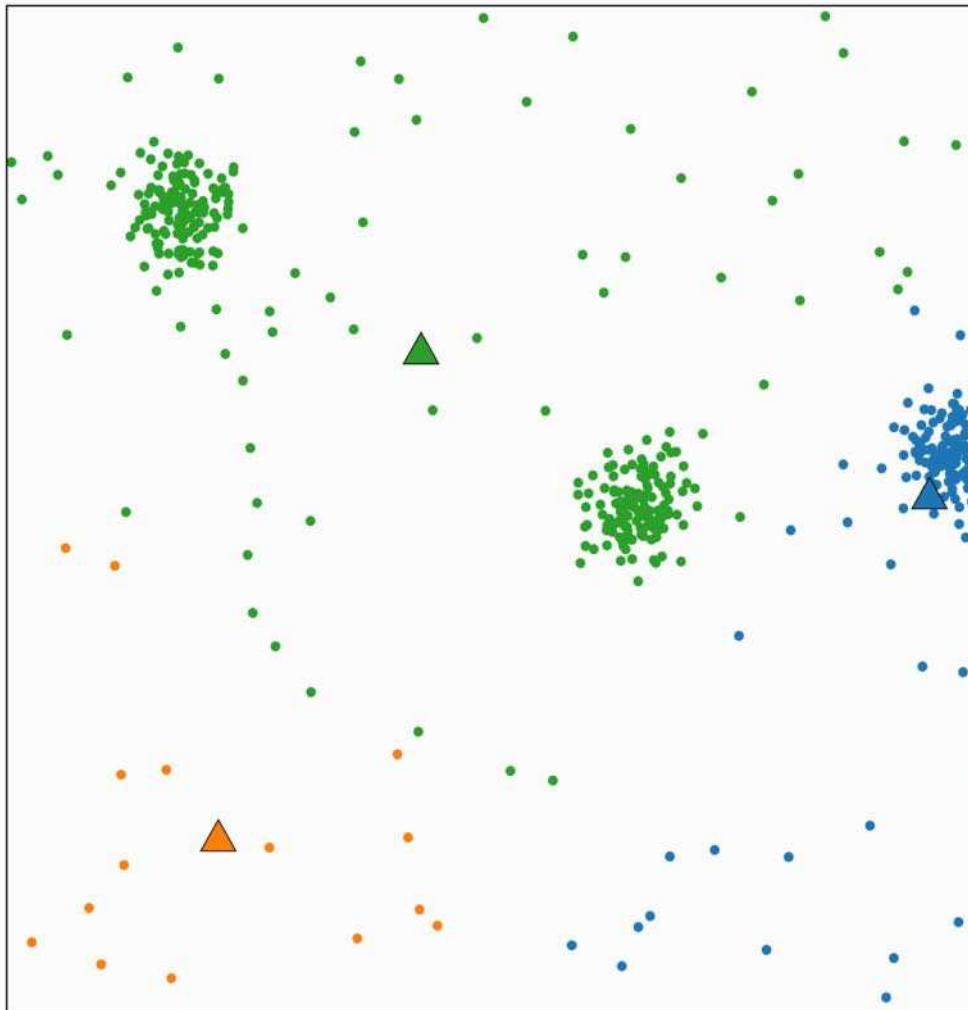
**Output:**  $k$  clusters

### Pseudocode:

1. Choose  $k$  objects as initial **cluster centers**.
2. Assign each data point to the cluster which has the closest **mean point (centroid)** under chosen distance metric.
3. When all data points have been assigned, recalculate the positions of  **$k$  centroids (mean points)**.
4. Repeat steps 2 and 3 until the **centroids** do not change any more. All data points remain in their most recently assigned cluster.

# K-means clustering

- example



## Pseudocode: k-means

**Complexity Range:**  $O(k \times n \times t)$

**Input:**  $k$  clusters

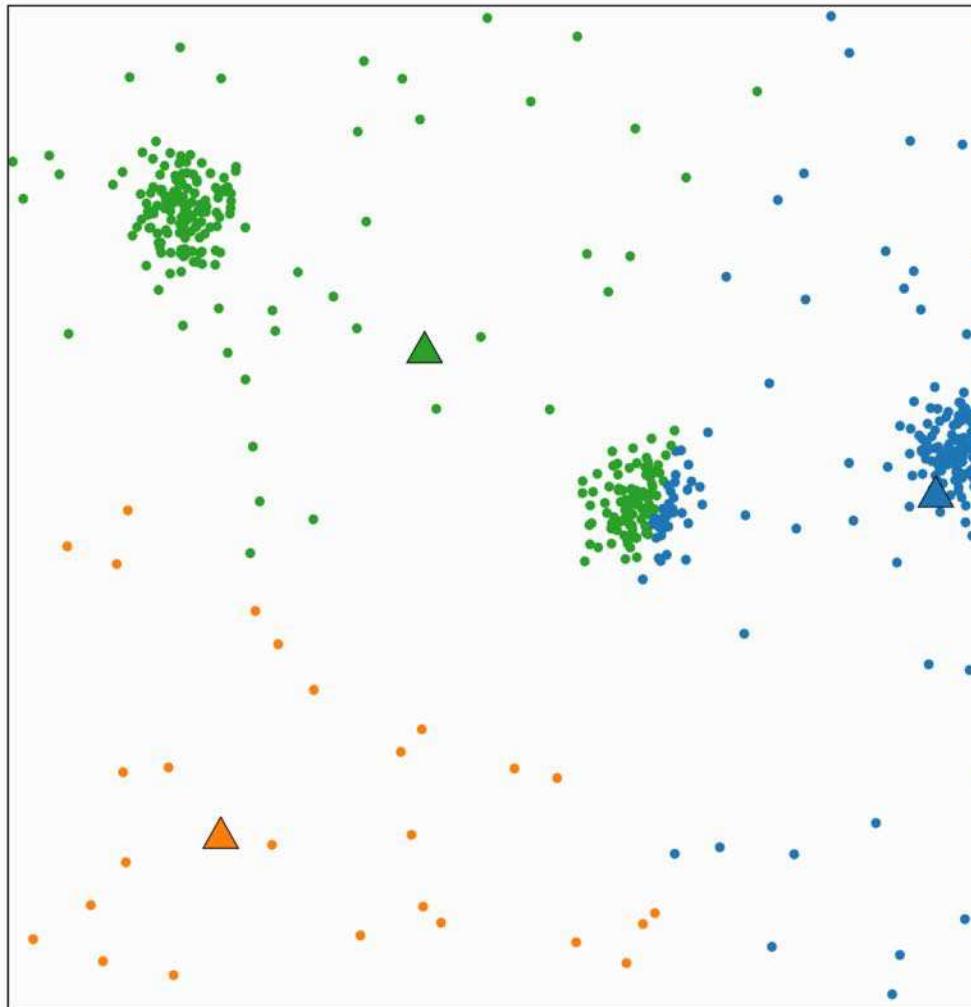
**Output:**  $k$  clusters

### Pseudocode:

1. Choose  $k$  objects as initial **cluster centers**.
2. Assign each data point to the cluster which has the closest **mean point (centroid)** under chosen distance metric.
3. When all data points have been assigned, recalculate the positions of  **$k$  centroids (mean points)**. (highlighted)
4. Repeat steps 2 and 3 until the **centroids** do not change any more. All data points remain in their most recently assigned cluster.

# K-means clustering

- example



## Pseudocode: k-means

**Complexity Range:**  $O(k \times n \times t)$

**Input:**  $k$  clusters

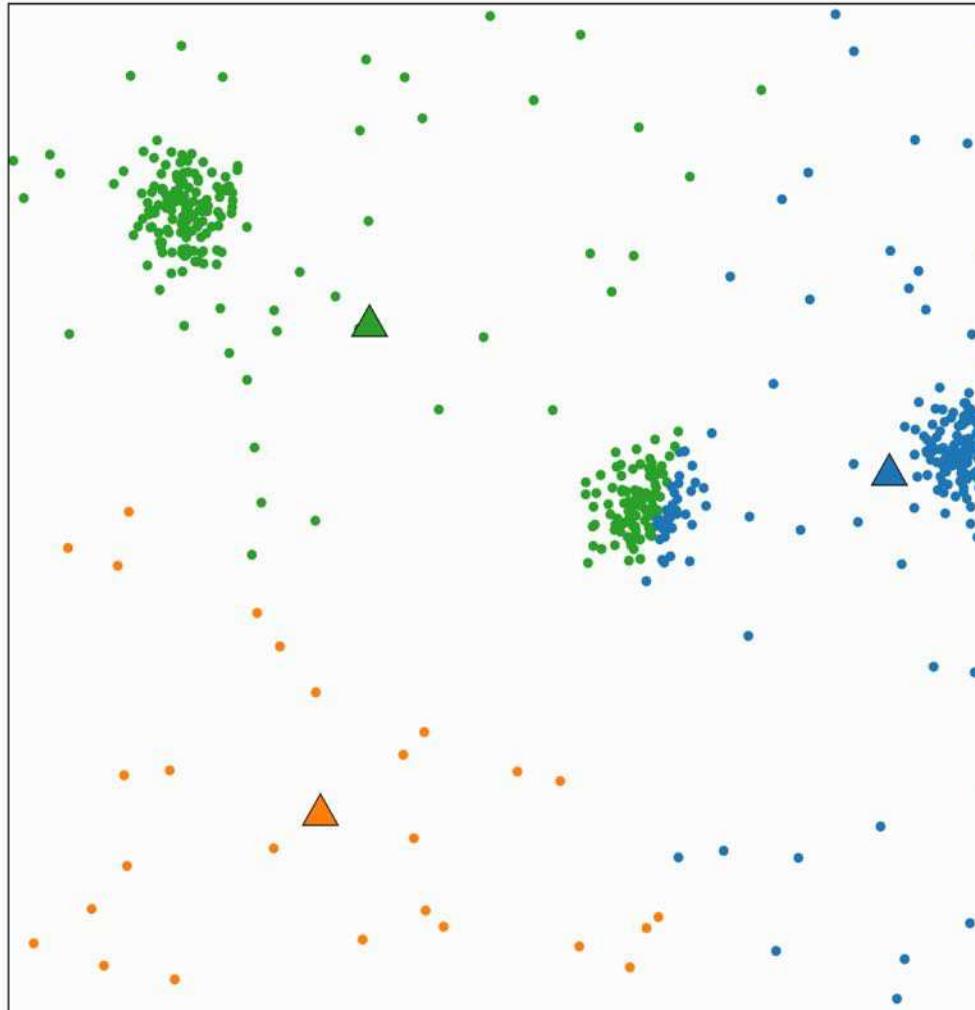
**Output:**  $k$  clusters

### Pseudocode:

1. Choose  $k$  objects as initial **cluster centers**.
2. Assign each data point to the cluster which has the closest **mean point (centroid)** under chosen distance metric.
3. When all data points have been assigned, recalculate the positions of  **$k$  centroids (mean points)**.
4. Repeat steps 2 and 3 until the **centroids** do not change any more. All data points remain in their most recently assigned cluster.

# K-means clustering

- example



## Pseudocode: k-means

**Complexity Range:**  $O(k \times n \times t)$

**Input:**  $k$  clusters

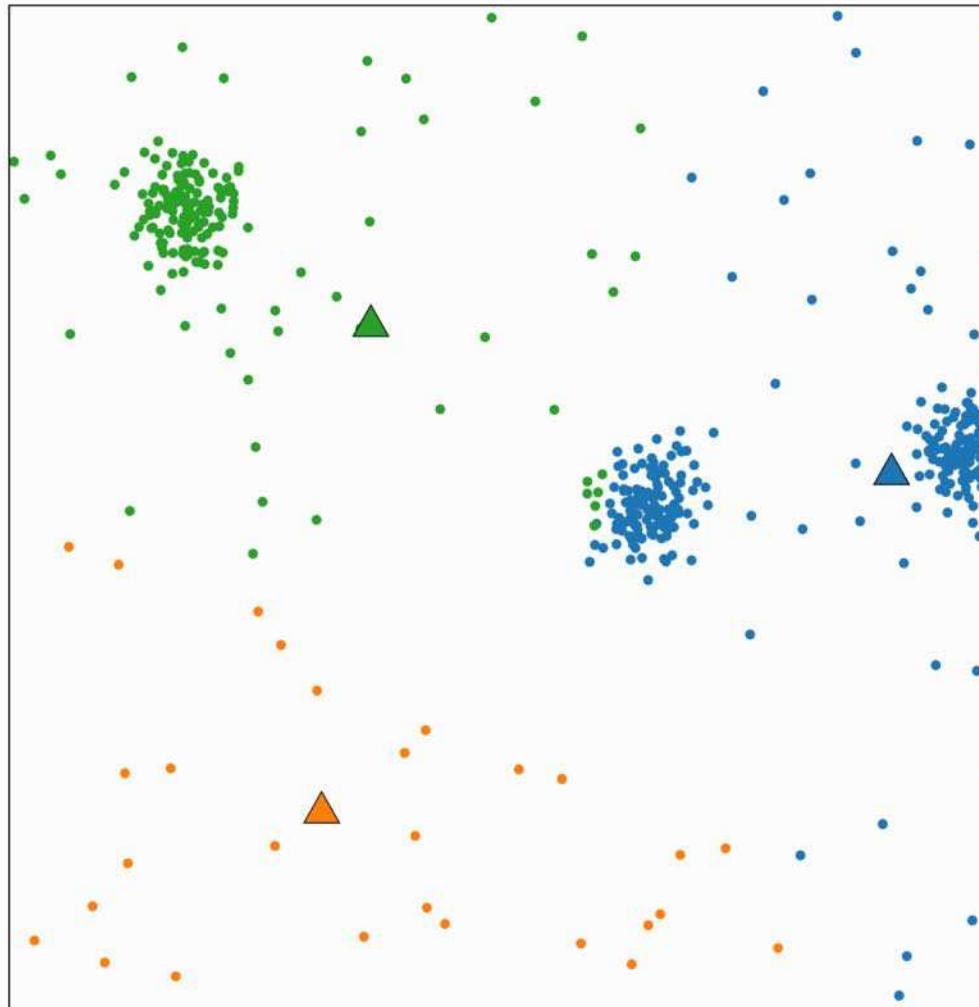
**Output:**  $k$  clusters

### Pseudocode:

1. Choose  $k$  objects as initial **cluster centers**.
2. Assign each data point to the cluster which has the closest **mean point (centroid)** under chosen distance metric.
3. When all data points have been assigned, recalculate the positions of  **$k$  centroids (mean points)**. (highlighted)
4. Repeat steps 2 and 3 until the **centroids** do not change any more. All data points remain in their most recently assigned cluster.

# K-means clustering

- example



## Pseudocode: k-means

**Complexity Range:**  $O(k \times n \times t)$

**Input:**  $k$  clusters

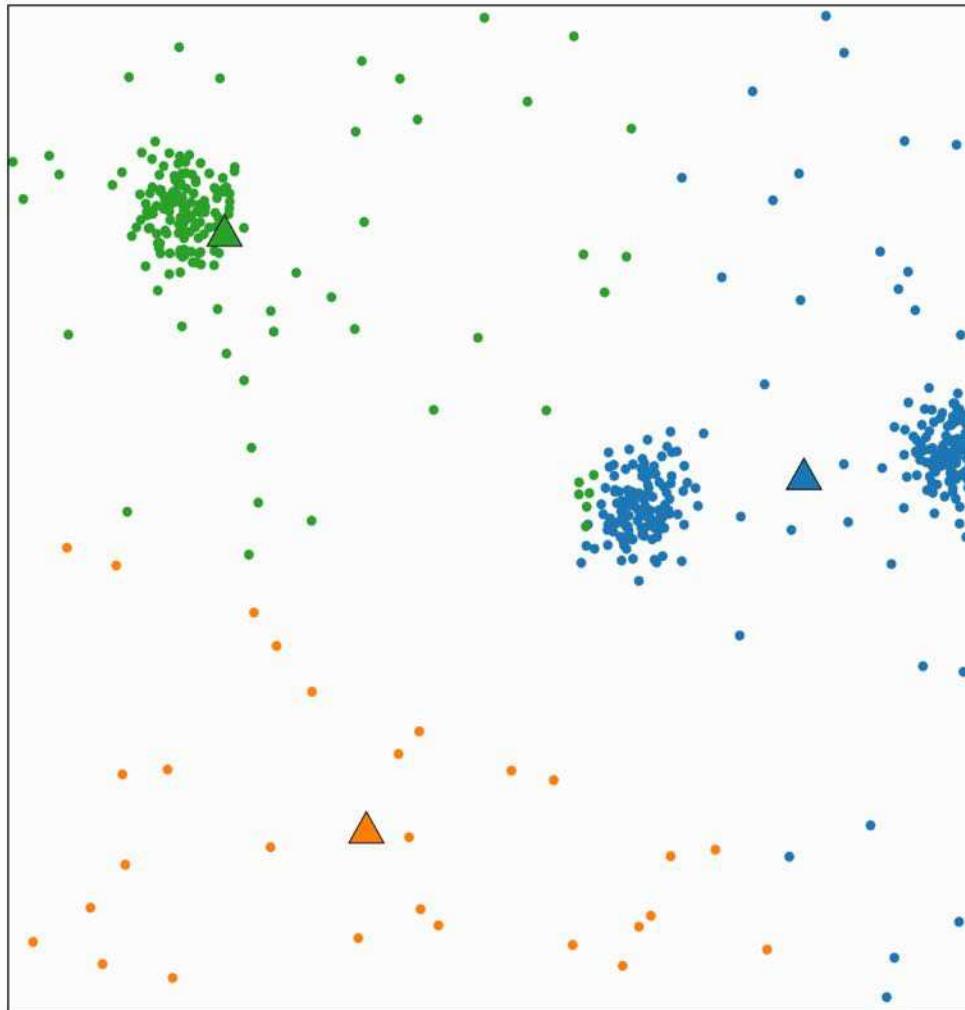
**Output:**  $k$  clusters

### Pseudocode:

1. Choose  $k$  objects as initial **cluster centers**.
2. Assign each data point to the cluster which has the closest **mean point (centroid)** under chosen distance metric.
3. When all data points have been assigned, recalculate the positions of  **$k$  centroids (mean points)**.
4. Repeat steps 2 and 3 until the **centroids** do not change any more. All data points remain in their most recently assigned cluster.

# K-means clustering

- example



## Pseudocode: k-means

**Complexity Range:**  $O(k \times n \times t)$

**Input:**  $k$  clusters

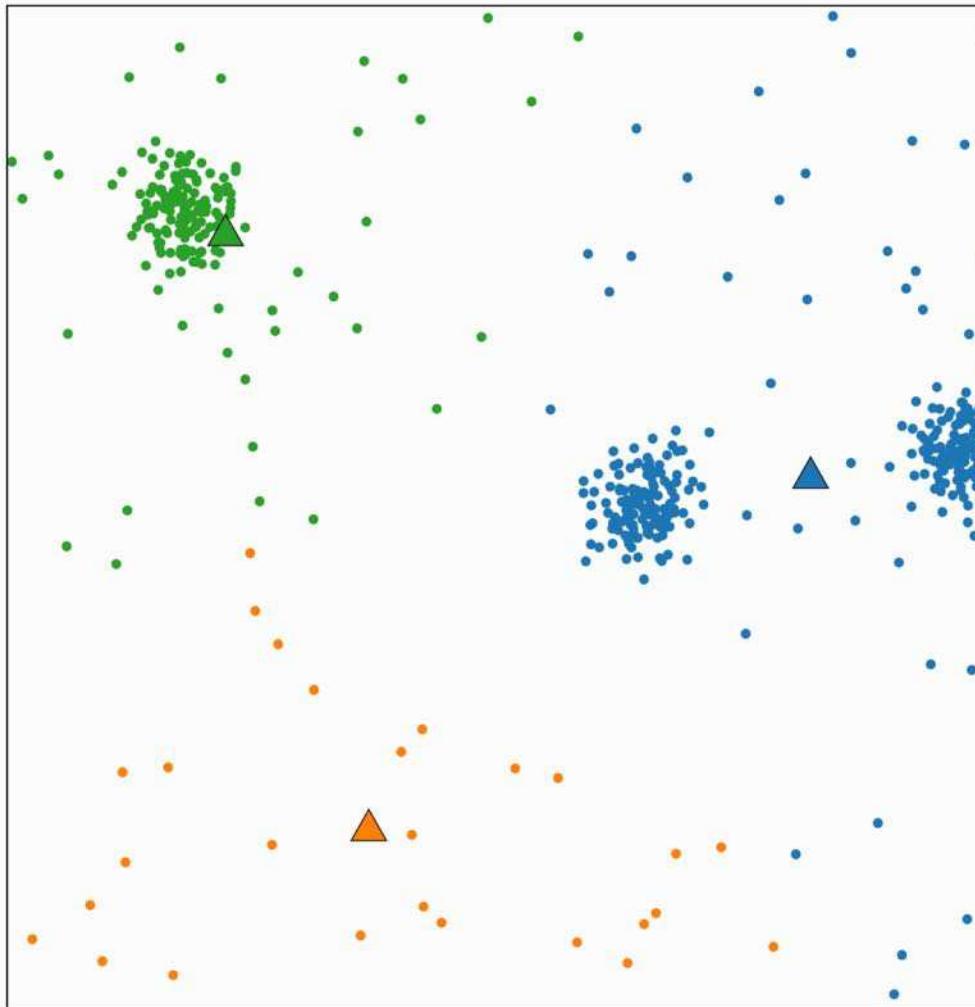
**Output:**  $k$  clusters

### Pseudocode:

1. Choose  $k$  objects as initial **cluster centers**.
2. Assign each data point to the cluster which has the closest **mean point (centroid)** under chosen distance metric.
3. When all data points have been assigned, recalculate the positions of  **$k$  centroids (mean points)**. (highlighted)
4. Repeat steps 2 and 3 until the **centroids** do not change any more. All data points remain in their most recently assigned cluster.

# K-means clustering

- example



## Pseudocode: k-means

**Complexity Range:**  $O(k \times n \times t)$

**Input:**  $k$  clusters

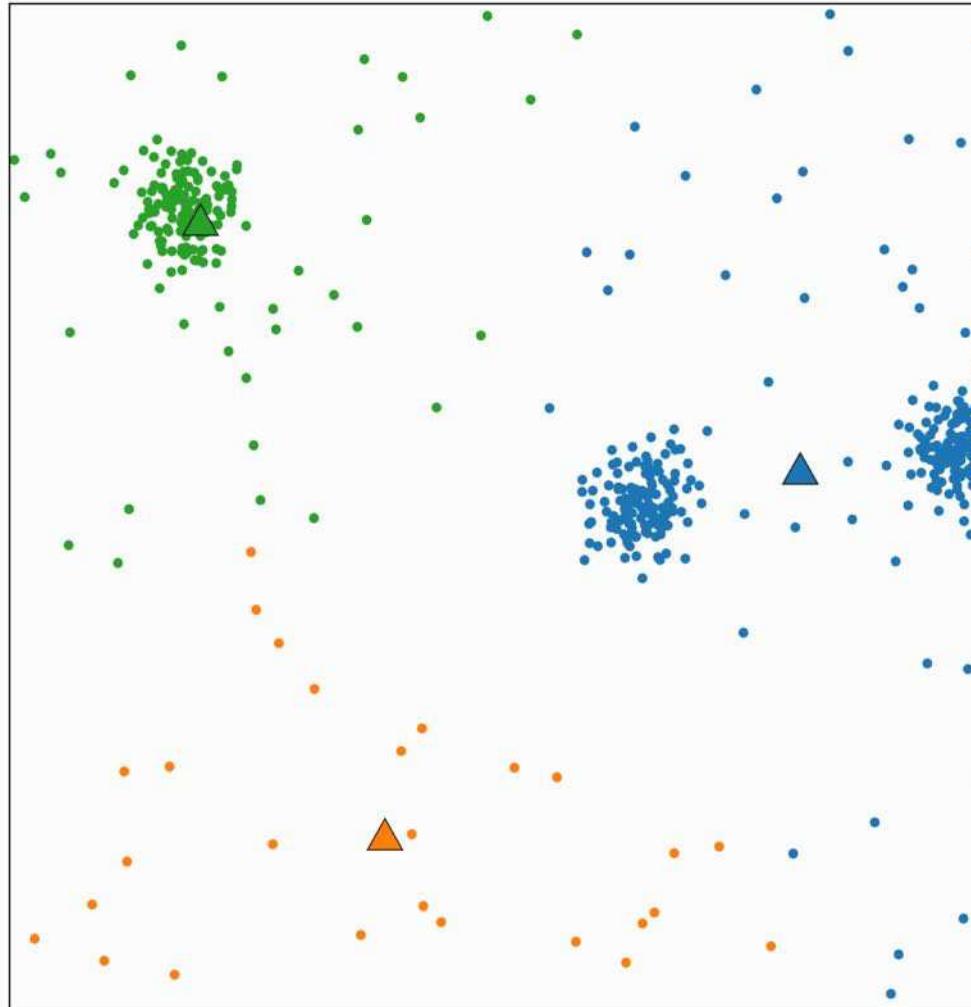
**Output:**  $k$  clusters

### Pseudocode:

1. Choose  $k$  objects as initial **cluster centers**.
2. Assign each data point to the cluster which has the closest **mean point (centroid)** under chosen distance metric.
3. When all data points have been assigned, recalculate the positions of  **$k$  centroids (mean points)**.
4. Repeat steps 2 and 3 until the **centroids** do not change any more. All data points remain in their most recently assigned cluster.

# K-means clustering

- example



## Pseudocode: k-means

**Complexity Range:**  $O(k \times n \times t)$

**Input:**  $k$  clusters

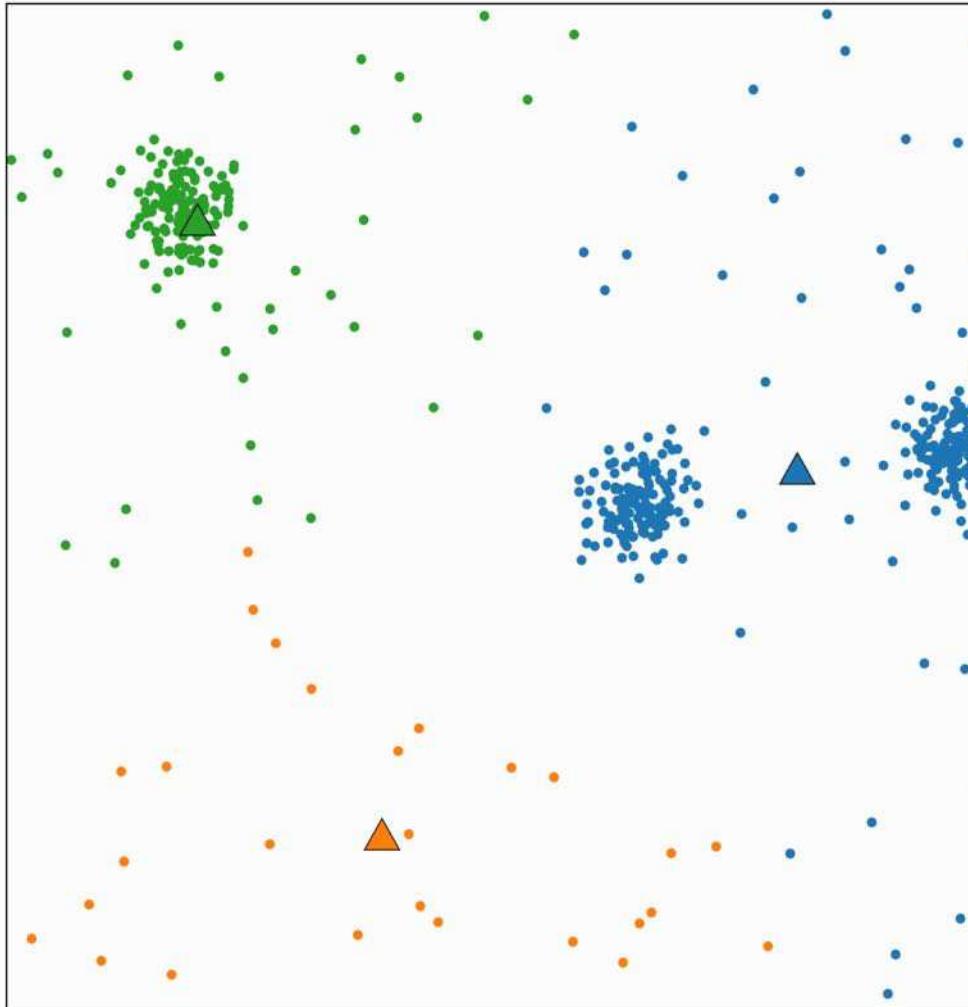
**Output:**  $k$  clusters

### Pseudocode:

1. Choose  $k$  objects as initial **cluster centers**.
2. Assign each data point to the cluster which has the closest **mean point (centroid)** under chosen distance metric.
3. When all data points have been assigned, recalculate the positions of  **$k$  centroids (mean points)**. (mean points)
4. Repeat steps 2 and 3 until the **centroids** do not change any more. All data points remain in their most recently assigned cluster.

# K-means clustering

- example



## Pseudocode: k-means

**Complexity Range:**  $O(k \times n \times t)$

**Input:**  $k$  clusters

**Output:**  $k$  clusters

### Pseudocode:

1. Choose  $k$  objects as initial **cluster centers**.
2. Assign each data point to the cluster which has the closest **mean point (centroid)** under chosen distance metric.
3. When all data points have been assigned, recalculate the positions of  **$k$  centroids (mean points)**.
4. Repeat steps 2 and 3 until the **centroids** do not change any more. All data points remain in their most recently assigned cluster.

# K-means clustering

- advantages
  - relatively efficient with  $O(tkn)$ 
    - $t = \# \text{ iterations}$
    - $k = \# \text{ clusters}$
    - $n = \# \text{ data points}$
  - simple implementation
- disadvantages
  - need to specify number of clusters  $k$
  - unable to handle noisy data and outliers
  - cannot detect clusters with non-convex shapes
  - applicable only when mean is defined
  - often terminates at a local optimum

# ISODATA

## ■ K-means refinement

- don't need to know the number of clusters
- clusters are merged if either the number of members in a cluster is less than a certain threshold or if the centers of two clusters are closer than a certain threshold
- clusters are split into two different clusters if the cluster standard deviation exceeds a predefined value and the number of members is twice the threshold for the minimum number of members
- user has to provide several additional parameter values

## Pseudocode: ISODATA

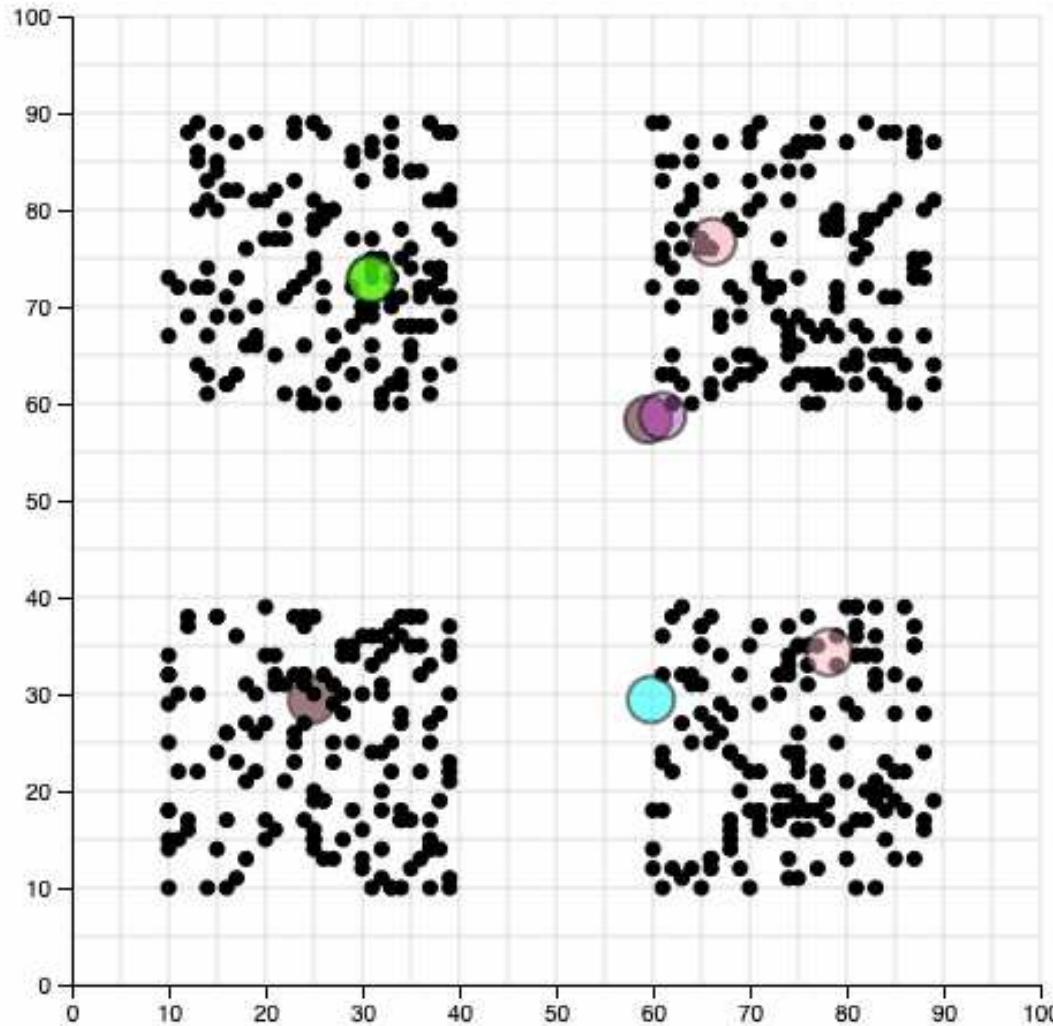
Complexity Range:  $O(k \times n \times t)$

Input:  $k$ , *MaxIter*, *MinPts*, *MinDist*, *StdDev*  
Output:  $k'$  clusters

### Pseudocode:

1. Choose  $k$  objects as initial **cluster centers**.
2. Assign each data point to the cluster which has the closest **mean point (centroid)** under chosen distance metric.
3. When all data points have been assigned, recalculate the positions of  **$k$  centroids (mean points)**.
4. Reassignment of data points do not change clusters. Clusters are splitted, merged or eliminated, if requirements (parameters) are not met.
5. Repeat steps 2 and 3 until the **centroids** do not change any more. All data points remain in their most recently assigned cluster and parameter requirements of cluster condition are fulfilled.

# ISODATA



## Pseudocode: ISODATA

**Complexity Range:**  $O(k \times n \times t)$

**Input:**  $k$ , **MaxIter**, **MinPts**, **MinDist**, **StdDev**

**Output:**  $k'$  clusters

**Pseudocode:**

1. Choose  $k$  objects as initial **cluster centers**.
2. Assign each data point to the cluster which has the closest **mean point (centroid)** under chosen distance metric.
3. When all data points have been assigned, recalculate the positions of  $k$  **centroids (mean points)**.
4. Reassignment of data points do not change clusters. Clusters are splitted, merged or eliminated, if requirements (parameters) are not met.
5. Repeat steps 2 and 3 until the **centroids** do not change any more. All data points remain in their most recently assigned cluster and parameter requirements of cluster condition are fulfilled.

# Clustering evaluation

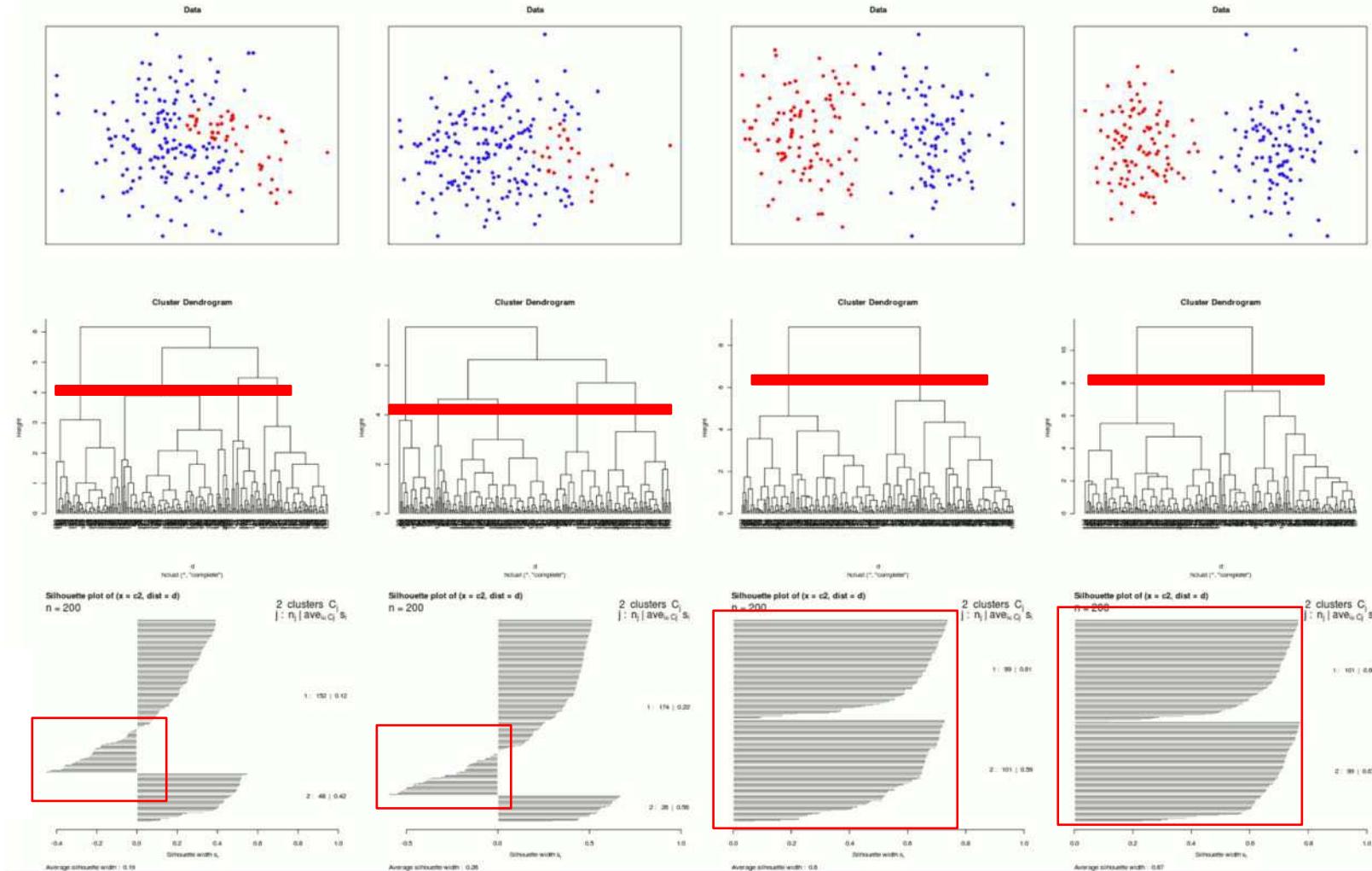
- without labels
  - elbow method: a heuristic used in determining the number of clusters in a data set
  - silhouette coefficient: measure for cluster validity/consistency
  - dendrogram: visual inspection of distances and balancing in hierarchical clusterings
- with labels
  - purity: a simple & transparent measure of correctness of clustering
  - rand index: compares two clusterings (e.g., own scheme with random partition)
  - missclassification rate:  $MR = \frac{1}{N} \sum_{j=1}^C e_j$  ( $N$  number of samples,  $C$  number of true clusters,  $e_j$  number of wrongly assigned samples of true cluster  $j$ , i.e. spread to multiple clusters or mixed in not pure clusters)
  - precision and recall can be also applied

# Clustering evaluation

- silhouette coefficient: measure for cluster validity/consistency
  - measure of clustering quality for k-means
  - silhouette score for a set of sample data points is used to measure how dense and well-separated the clusters are.
  - silhouette score takes into consideration the intra-cluster distance between the sample and other data points within the same cluster and inter-cluster distance between the sample and the next nearest cluster
  - the silhouette score falls within the range [-1, 1]
  - the silhouette score of 1 means that the clusters are very dense and nicely separated. The score of 0 means that clusters are overlapping. The score of less than 0 means that data belonging to clusters may be wrong/incorrect
  - the silhouette plots can be used to select the most optimal value of the K (no. of cluster) in K-means clustering

# Clustering evaluation

- silhouette coefficient: measure for cluster validity/consistency



# Lessons Learned

- understanding of different partitioning clustering techniques
- understanding of measuring the quality of a clustering result



# Questions and Answers



# Machine Learning and Data Mining

## V11: Hierarchical and Density Clustering

Lecturer: Dr. Andreas Weiler



# Repetition

... REPETITION ...

→ IS THE MOTHER →

of learning

# Course Content IT20a\_WIN

Week	Lecture	Lab
KW08	Introduction	Lab 01
KW09	Data Processing Part 01	Lab 02
KW10	Carnival Monday	
KW11	Data Processing Part 02	Lab 03
KW12	Evaluation	Lab 04
KW13	Recommender Systems	<b>Graded Lab 01</b> <span style="border: 1px solid red; padding: 2px;">IMPORTANT</span>
KW14	Association Rules	<b>Graded Lab 01</b> <span style="border: 1px solid red; padding: 2px;">IMPORTANT</span>
KW15	Linear and Logistic Regression	Lab 05
KW16	Easter Monday	
KW17	Support Vector Machines	Lab 06
KW18	Decision Trees and Naïve Bayes	<b>Graded Lab 02</b> <span style="border: 1px solid red; padding: 2px;">IMPORTANT</span>
KW19	Partition Clustering	<b>Graded Lab 02</b> <span style="border: 1px solid red; padding: 2px;">IMPORTANT</span>
KW20	Hierarchical and Density Clustering	Lab 07 / <b>Graded Lab 02</b> <span style="border: 1px solid red; padding: 2px;">IMPORTANT</span>
KW21	Neural Networks and Closing	Review and Questions for Exam

# Learning Objectives

- understanding of hierarchical clustering techniques
- understanding of density based clustering techniques



# Clustering

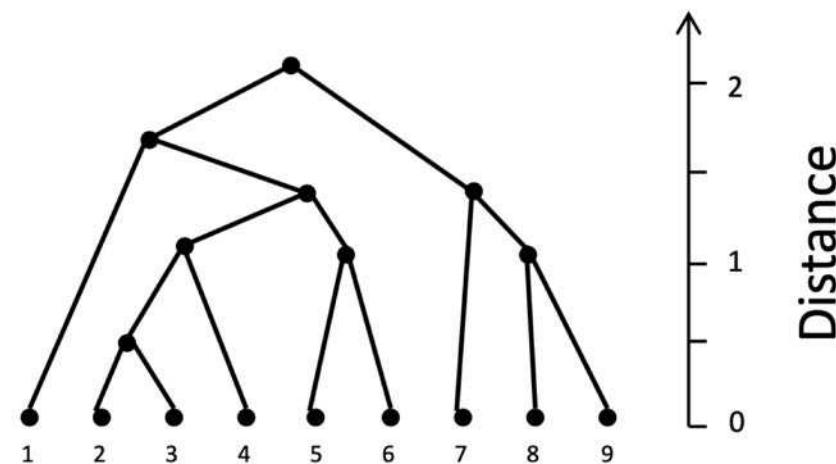
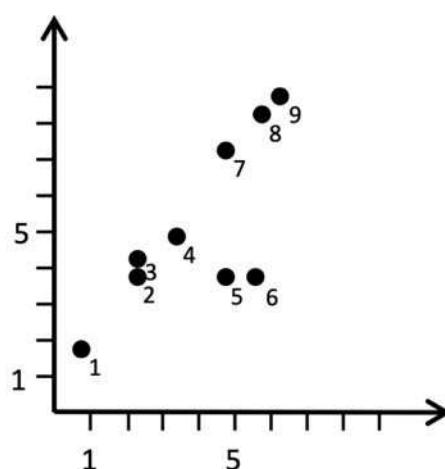
- clustering is the task of grouping a set of objects in such a way that objects in the same group (cluster) are more similar (in some sense) to each other than to those in other groups
- typical cluster models include:
  - connectivity models: for example, hierarchical clustering builds models based on distance connectivity
  - centroid models: for example, the k-means algorithm represents each cluster by a single mean vector
  - distribution models: clusters are modeled using statistical distributions, such as multivariate normal distributions used by the expectation-maximization algorithm
  - density models: for example, DBSCAN and OPTICS defines clusters as connected dense regions in the data space

# Hierarchical Clustering

- hierarchical clustering builds models based on distance connectivity and merging of clusters with minimum distance
- a typical clustering analysis approach via partitioning data set sequentially
- construct nested partitions layer by layer via grouping objects into a tree of clusters called dendrogram
- bottom-up strategy
  - initially each data object is in its own (atomic) cluster
  - then merge these atomic clusters into larger and larger clusters

# Hierarchical Clustering

- a **dendrogram** is a **tree of nodes** representing clusters
- satisfying the following properties
  - root represents the **whole data set**
  - leaf nodes represent **clusters** containing a **single object**
  - inner nodes represent the **union of all objects contained in its corresponding subtrees**
- dendrogram example

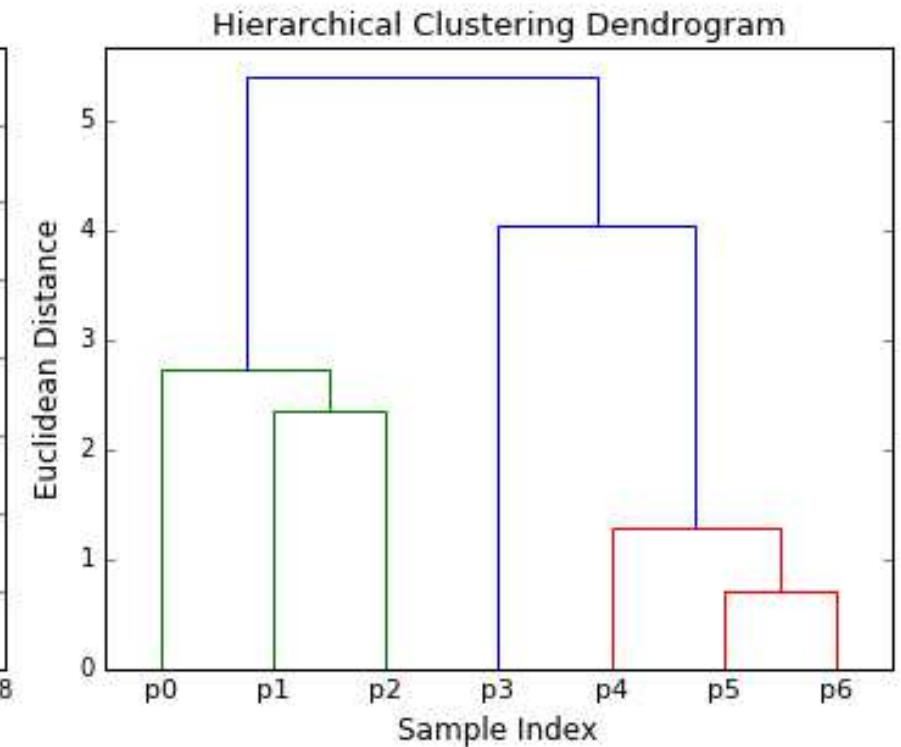
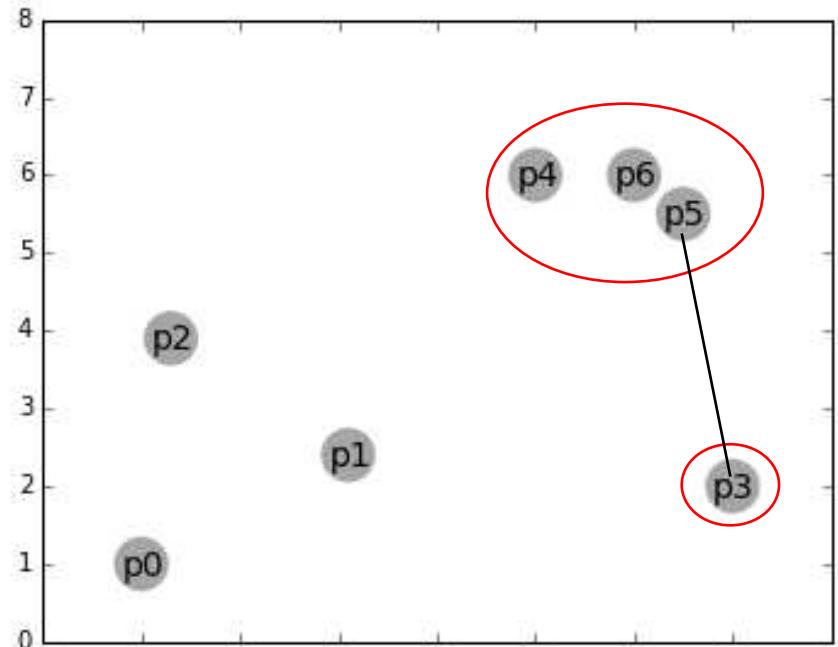


# Hierarchical Clustering

- method
  - step 1: form **initial clusters** consisting of a **single object** and compute **the distance between each pair of clusters**
  - step 2: merge the two clusters having **minimum distance**
  - step 3: calculate the **distance between the new cluster and all other clusters**
  - step 4: if there is **only one cluster** containing all objects: **STOP**, otherwise go to **step 2**
- **no input parameter is needed**
- **different methods** can be applied
  - single, complete, average, median and centroid linkage

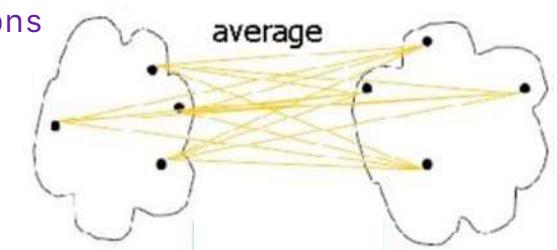
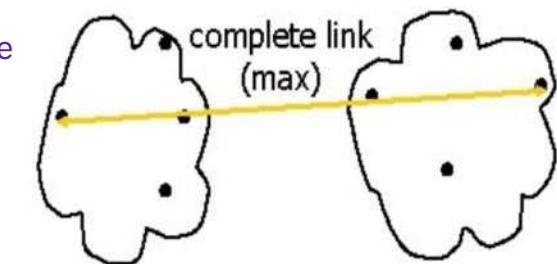
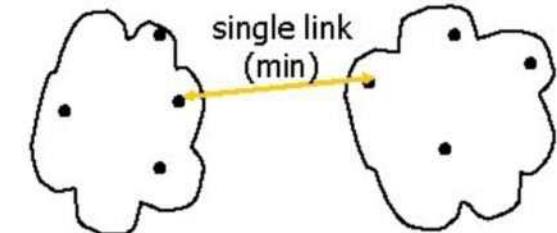
# Hierarchical Clustering

- example



# Hierarchical Clustering

- **single linkage**      distance between 2 elements which are as close as possible
  - shortest distance single link (min) between an element in one cluster and an element in the other, i.e.,  $d(C_i, C_j) = \min\{d(x_{ip}, x_{jq})\}$
- **complete linkage**      distance between 2 elements which are as far away as possible
  - largest distance between an element in one cluster complete link (max) and an element in the other, i.e.,  $d(C_i, C_j) = \max\{d(x_{ip}, x_{jq})\}$
- **average linkage**      Average of all pair-wise connections of elements of both clusters
  - avg distance between average elements in one cluster and elements in the other, i.e.,  $d(C_i, C_j) = \text{avg}\{d(x_{ip}, x_{jq})\}$

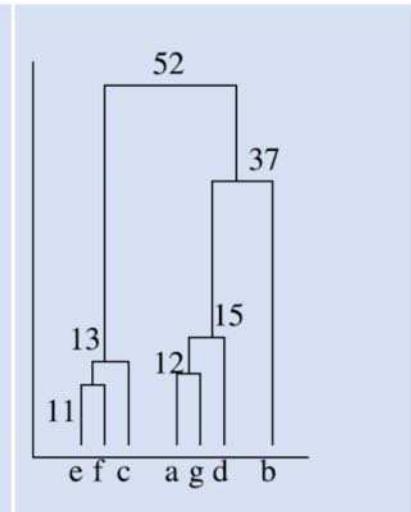


# Single Linkage

- example

	a	b	c	d	e	f	g
a	0	50	63	17	72	81	12
b		0	49	41	42	54	37
c			0	52	13	16	61
d				0	56	66	15
e					0	<b>11</b>	64
f						0	74
g							0

	a	b	c	d	ef	g
a	0	50	63	17	72	<b>12</b>
b		0	49	41	42	37
c			0	52	13	61
d				0	56	15
ef					0	<b>74</b>
g						0



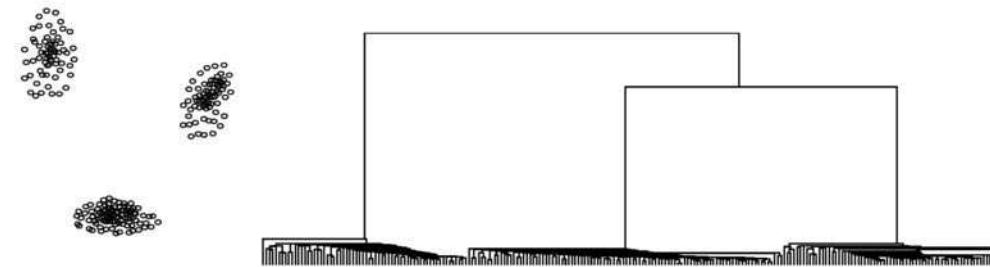
	ag	b	c	d	ef
ag	0	37	61	15	72
b		0	49	41	42
c			0	52	<b>13</b>
d				0	56
ef					0

	ag	b	d	efc
ag	0	37	<b>15</b>	61
b		0	41	42
d			0	<b>52</b>
efc				0

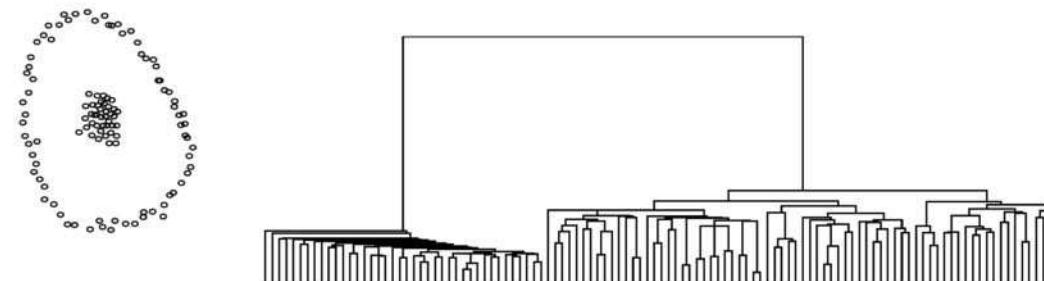
	agd	b	efc
agd	0	<b>37</b>	52
b		0	54
efc			0

# Single Linkage

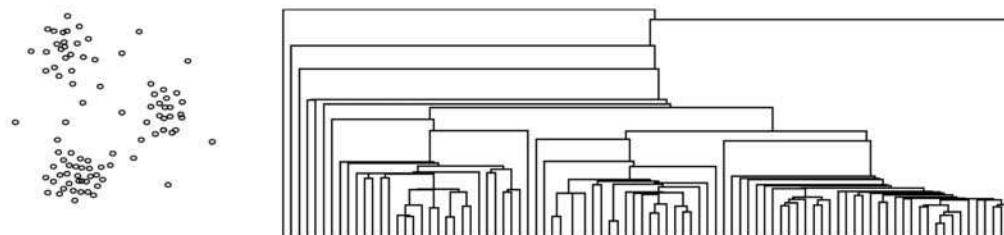
- three well-separated clusters



- two well-separated clusters

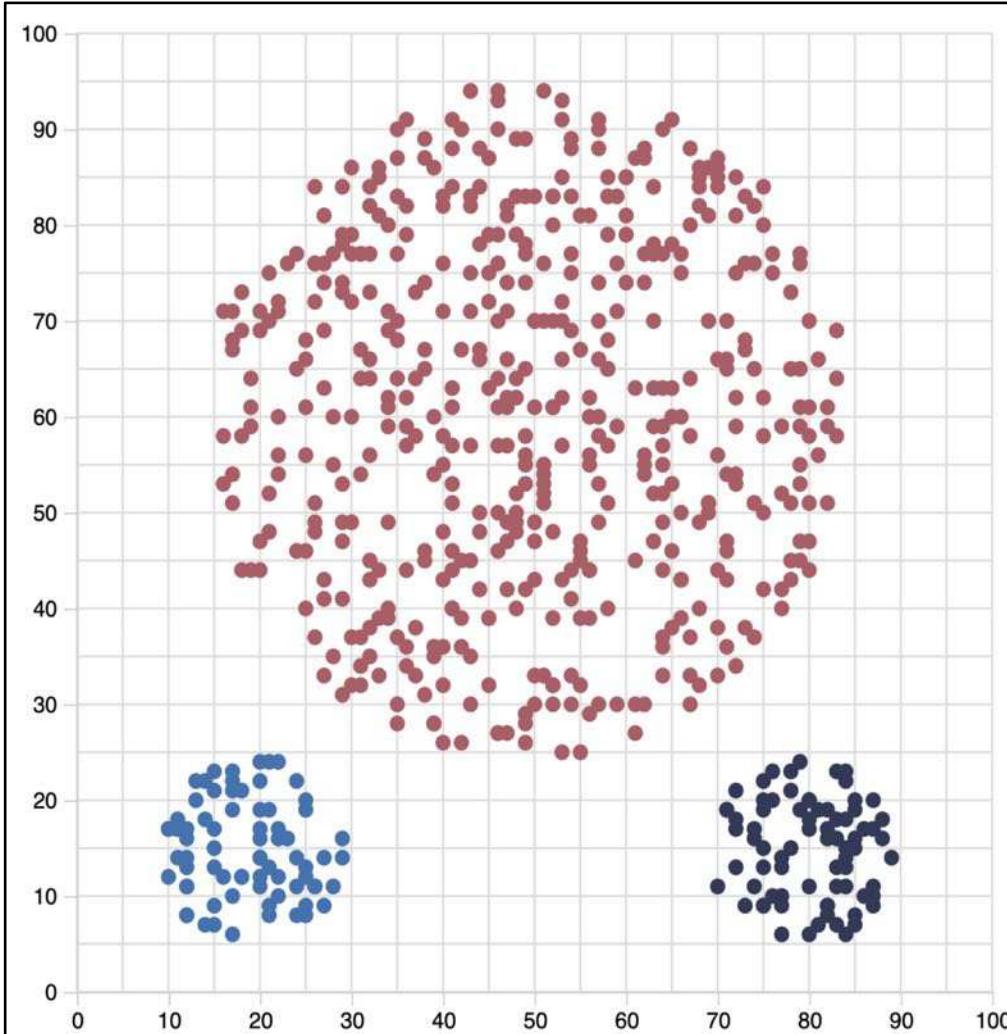


- small noises to clusters



# Single Linkage

- example



## Pseudocode: Single Linkage

**Complexity Range:**  $O(n^3)$

**Input:** No user-adjustable parameters used.

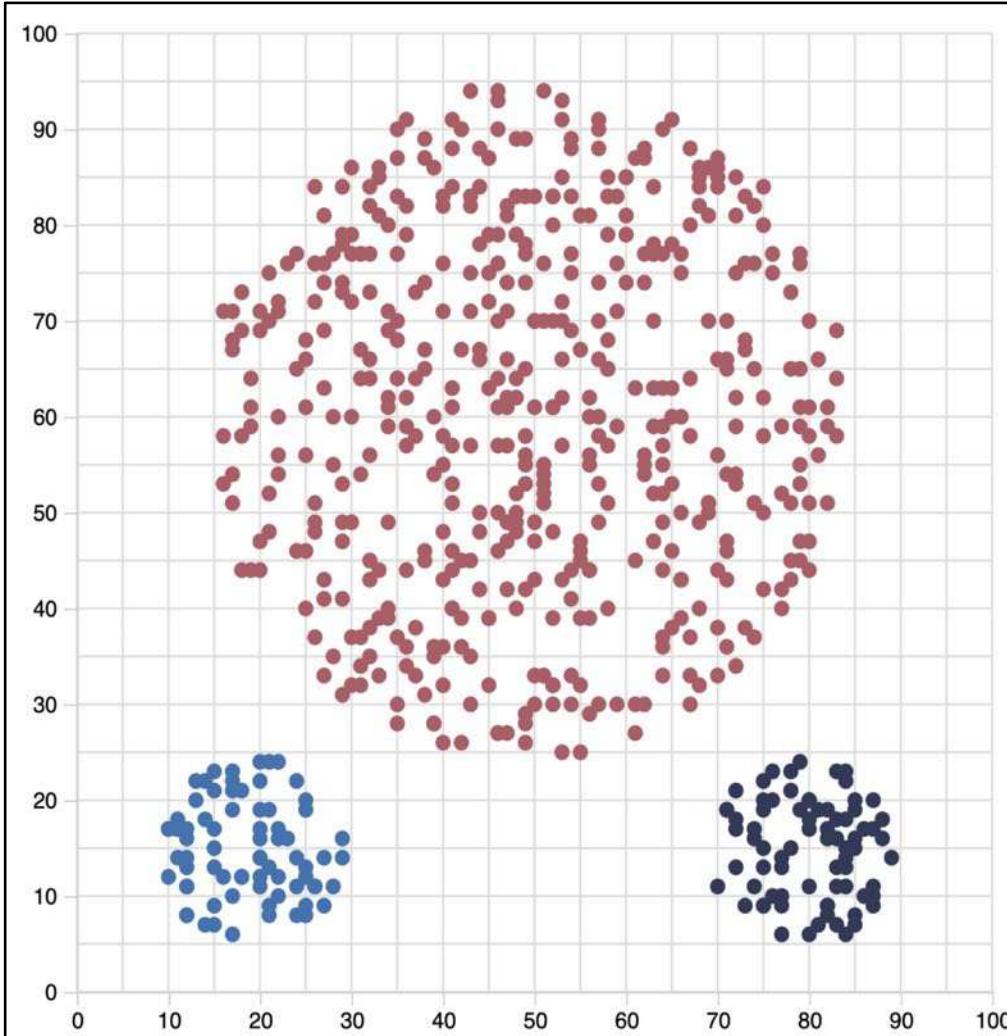
**Output:** Dendrogram, Leveled Cluster Coloring

### Pseudocode:

1. Compute **Distance (Proximity) Matrix** based on the chosen method (**Single Linkage**)
2. Join clusters having the minimal distance (depends on the chosen method (Single))
3. Go back to **step 1** until all points are **processed** and **dendrogram** is finished.

# Complete Linkage

- example



## Pseudocode: Complete Linkage

**Complexity Range:**  $O(n^3)$

**Input:** No user-adjustable parameters used.

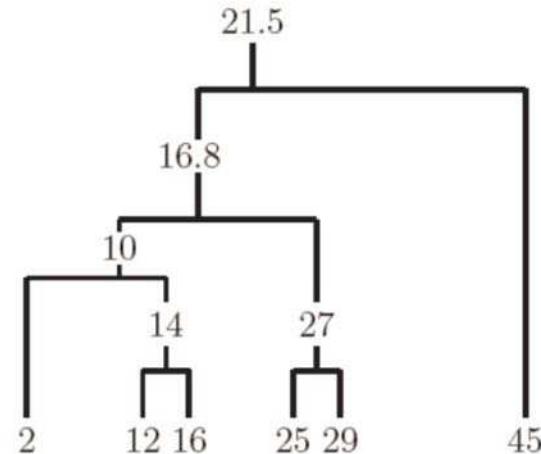
**Output:** Dendrogram, Leveled Cluster Coloring

### Pseudocode:

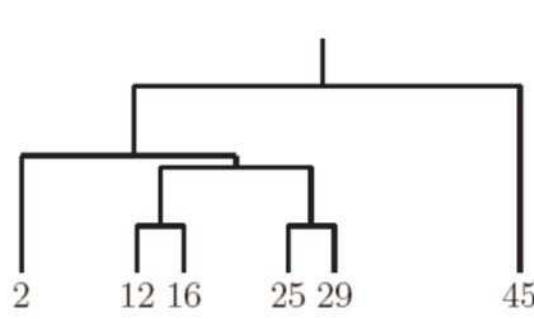
1. Compute **Distance (Proximity) Matrix** based on the chosen method (**Complete Linkage**)
2. Join clusters having the minimal distance (depends on the chosen method (Complete))
3. Go back to **step 1** until all points are processed and dendrogram is finished.

# Single Linkage

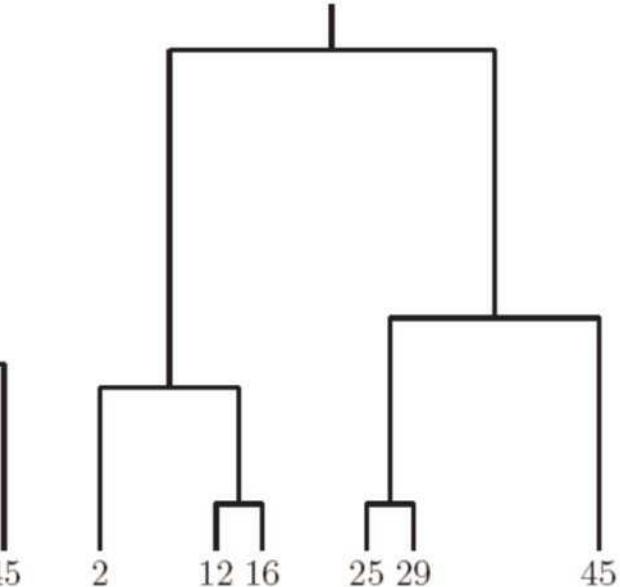
- clustering of the 1-dimensional data set {2, 12, 16, 25, 29, 45}
- all three approaches to measure the distance between clusters lead to different dendograms



Centroid



Single linkage



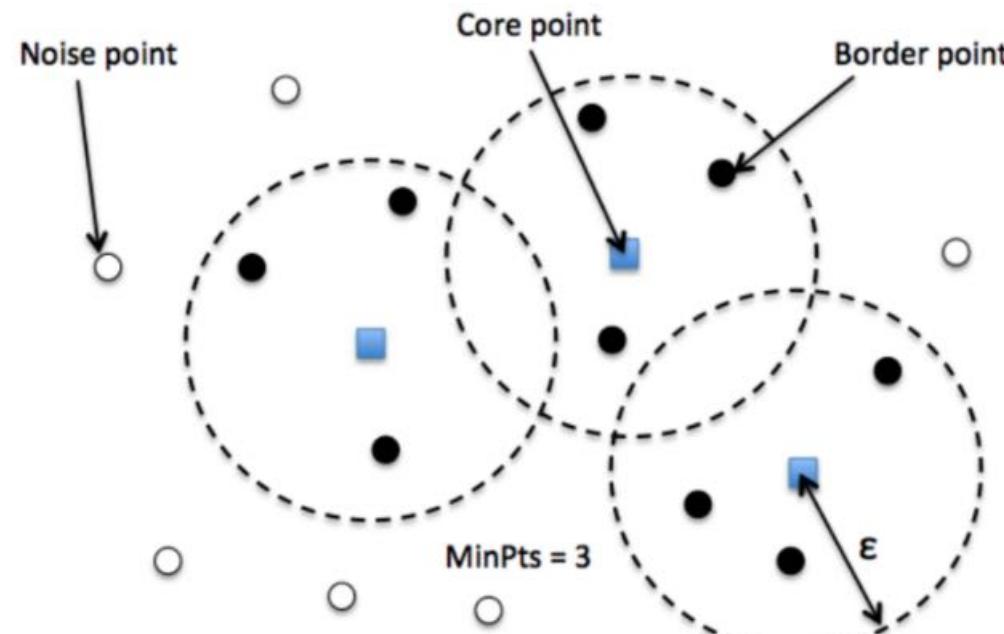
Complete linkage

# Density based Clustering

- density based clustering defines clusters as connected dense regions in the data space
- no need to specify the number of clusters
- DBSCAN (Density-Based Spatial Clustering of Applications with Noise)
  - minPts: the minimum number of points (a threshold) clustered together for a region to be considered dense
  - eps ( $\varepsilon$ ): a distance measure that will be used to locate the points in the neighborhood of any point
  - reachability in terms of density establishes a point to be reachable from another if it lies within a particular distance (eps) from it
  - connectivity, on the other hand, involves a transitivity based chaining-approach to determine whether points are located in a particular cluster

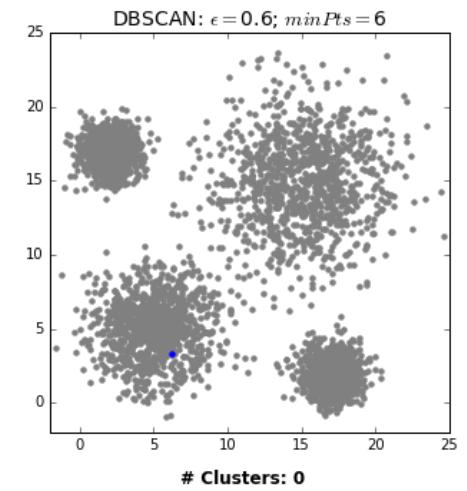
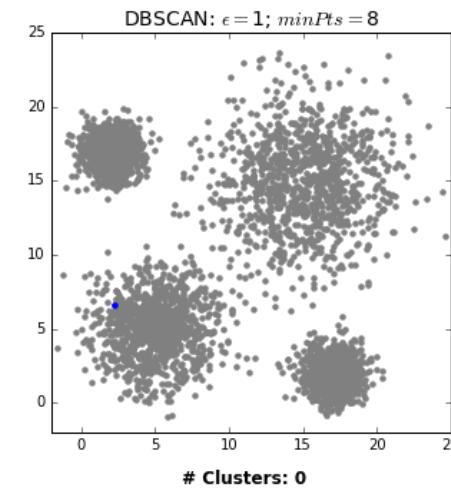
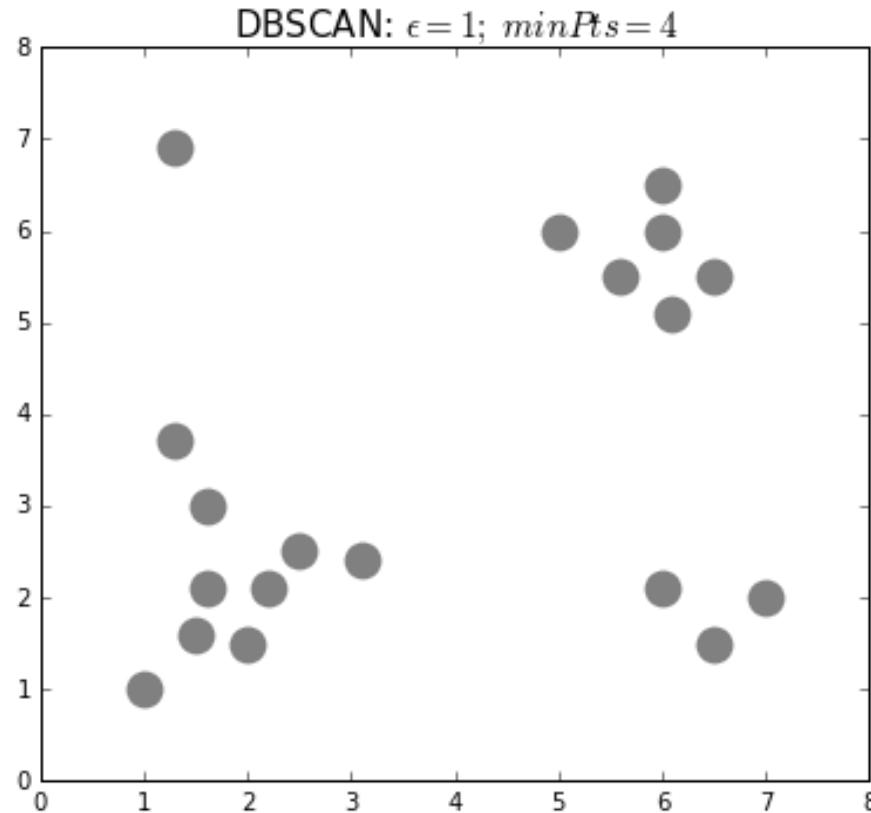
# Density based Clustering

- DBSCAN (Density-Based Spatial Clustering of Applications with Noise)
  - core: this is a point that has at least m points within distance n from itself
  - border: this a point that has at least one core point at a distance n
  - noise: this is a point that is neither a core nor a border. And it has less than m points within distance n from itself



# Density based Clustering

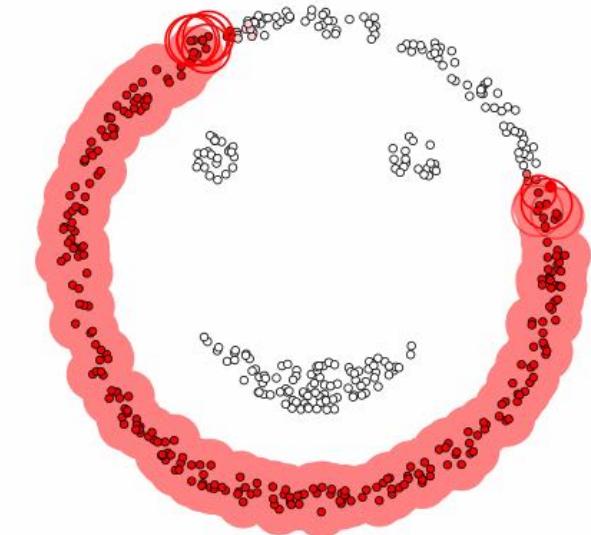
- DBSCAN (Density-Based Spatial Clustering of Applications with Noise)



# Density based Clustering

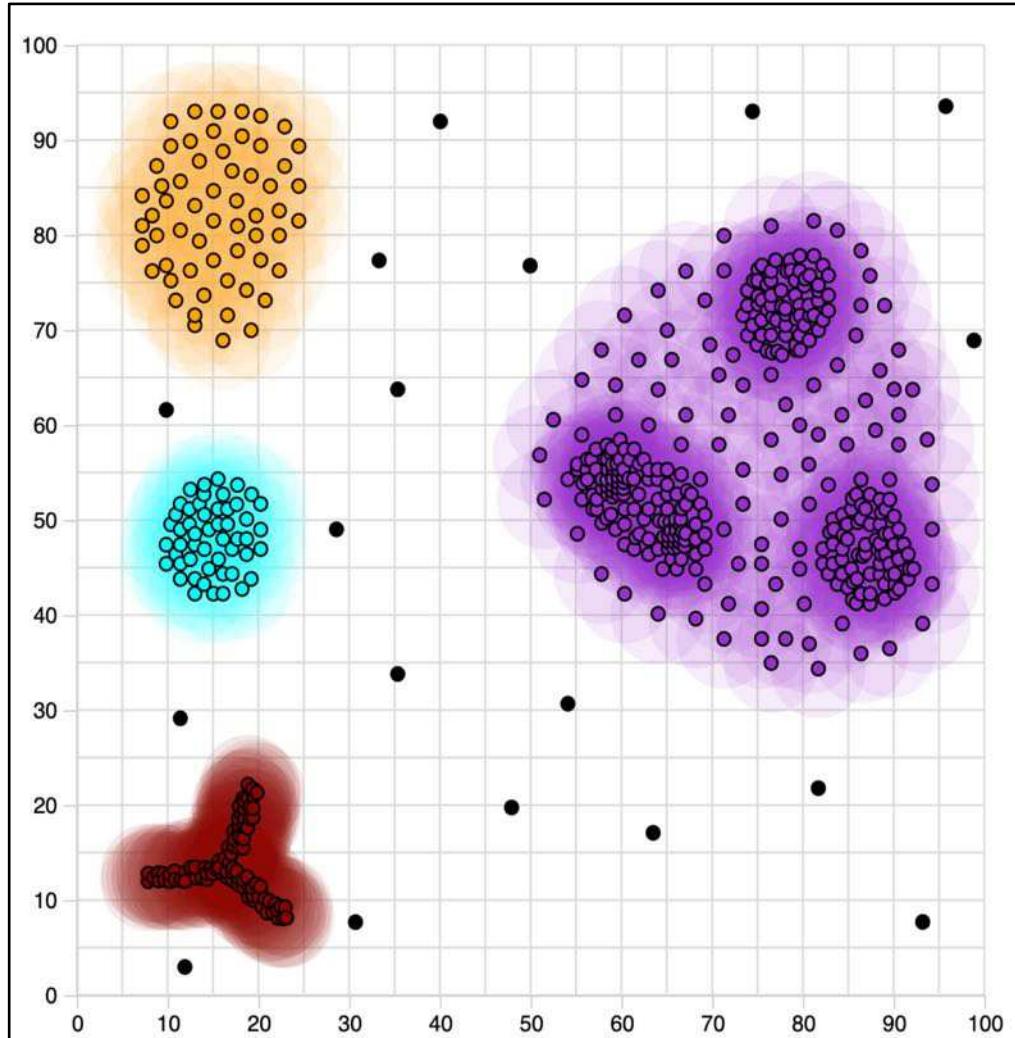
## ■ DBSCAN

- initially, it starts with a random unvisited starting data point. All points within a distance  $\epsilon$  classify as neighborhood points
- it needs a minimum number of 'minPts' points within the neighborhood to start the clustering process. Otherwise, the point gets labeled as 'Noise.'
- all points within the distance  $\epsilon$  become part of the same cluster. Repeat the procedure for all the new points added to the cluster group. continue till it visits and labels each point within the  $\epsilon$  neighborhood of the cluster.
- on completion of the process, it starts again with a new unvisited point thereby leading to the discovery of more clusters or noise. At the end of the process, you ensure that you mark each point as either cluster or noise



# DBSCAN

## ■ example



### Pseudocode: DBSCAN

**Complexity Range:**  $O(n \log n) - O(n^2)$

**Input:**  $\epsilon, MinPts$

**Output:** Cluster of data points with noise.

#### Pseudocode:

1. Select an unprocessed data point  $P$  and retrieve all points density-reachable (neighbors) from  $P$  wrt  $\epsilon$  and  $MinPts$ .
2. If  $P$  is a core point, a cluster is formed. Otherwise, classify  $P$  as noise.
3. Retrieve all points density-reachable from neighbor points and assign them to current cluster.
4. If  $P$  or neighbor is a border point, no points are density-reachable from  $P$ . Go to step 1.
5. Continue algorithm until all data points have been processed.

# DBSCAN

- advantages
  - no need to specify the number of clusters in advance
  - able to find arbitrarily shaped clusters
  - able to detect noise
- disadvantages
  - cannot cluster data sets well with large differences in densities

# Lessons Learned

- understanding of hierarchical clustering techniques
- understanding of density based clustering techniques



# Questions and Answers



## V12: Neural Networks

Lecturer: **Dr. Andreas Weiler**

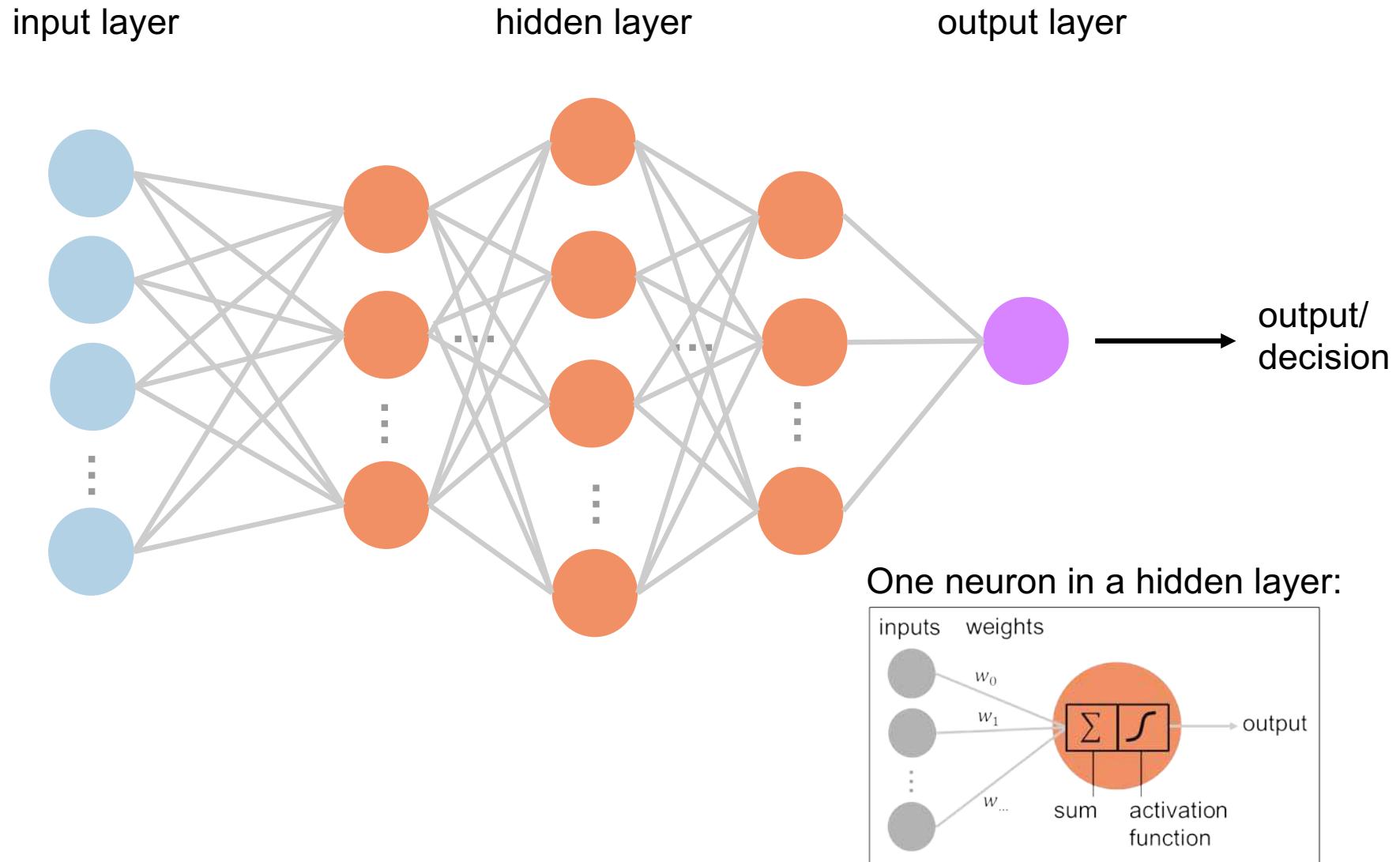


# Learning Objectives

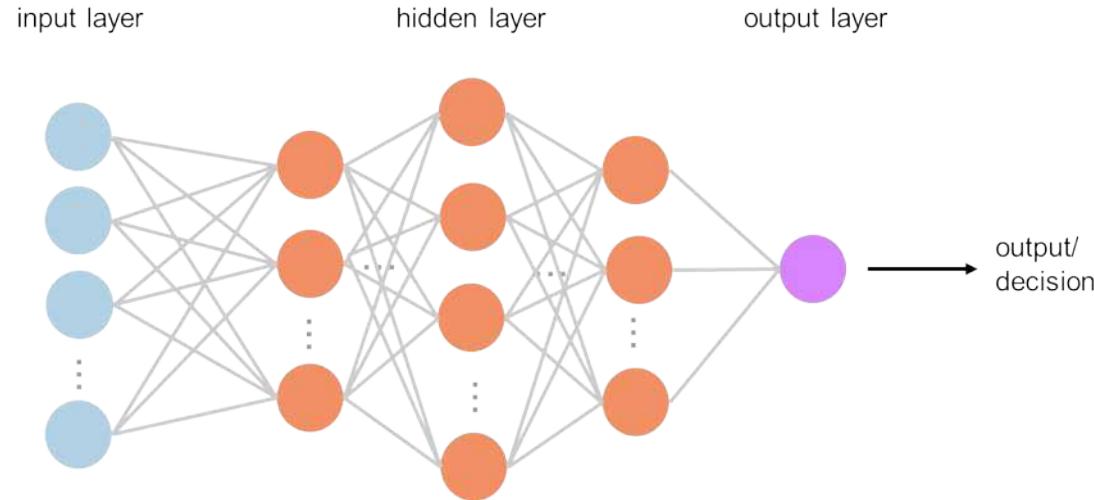
- fundamental understanding of feed forward neural networks
- fundamental understanding of questions and topics for the SEP



A Neural Network has an Input Layer, one or several Hidden Layers and an Output Layer



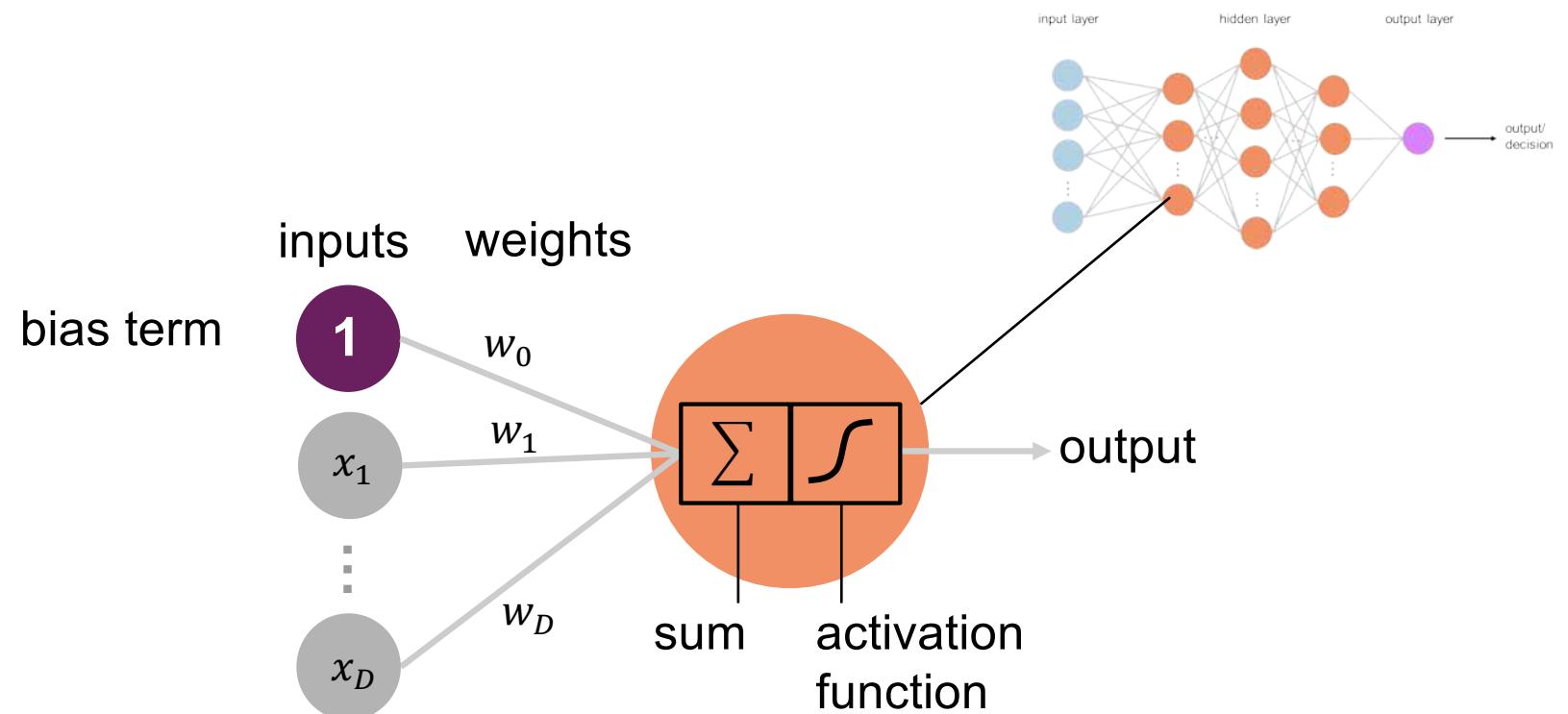
# Feedforward Neural Network



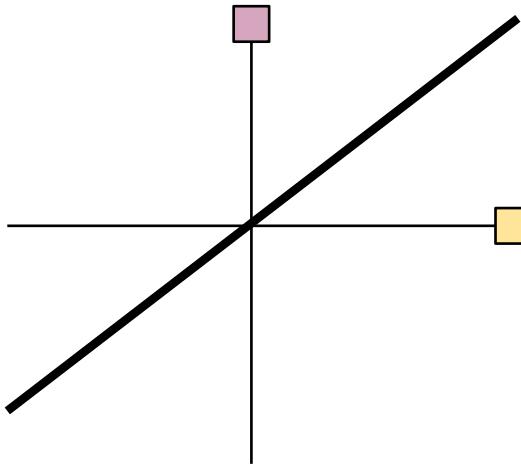
- The information moves in **only one direction**—forward—from the input nodes, through the hidden nodes (if any) and to the output nodes. There are **no cycles or loops** in the network. → Each neuron in one layer has **directed connections** to the **neurons of the subsequent layer**
- **No connections** between nodes **within the same layer**
- Vanilla Neural Network Structure: Subsequent layers are **fully connected**
- **Parameters:** Weights connecting the layer

# A Neuron

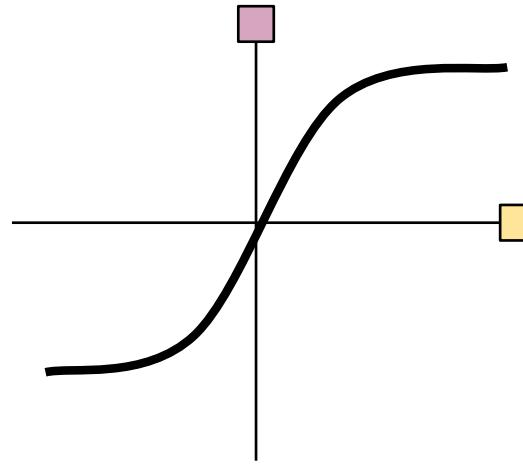
... applies an activation function to the weighted sum of its inputs



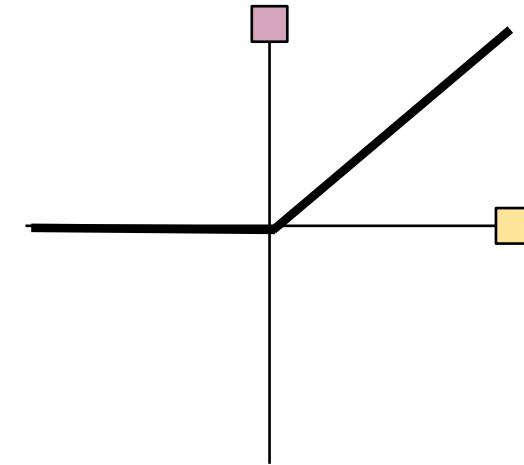
# Activation Functions



**LINEAR**  
like linear regression  
(only used in output layer)



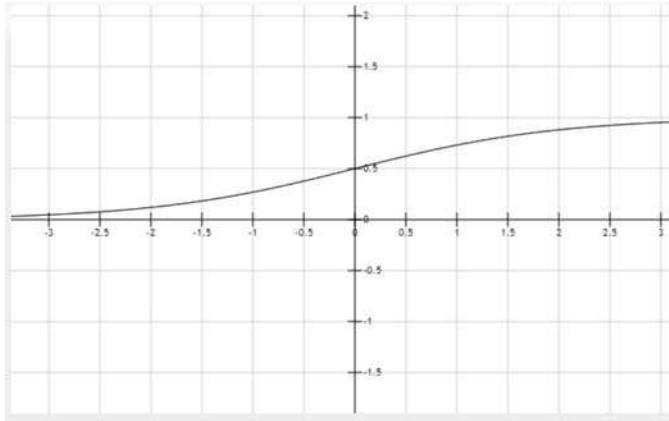
**LOGISTIC / SIGMOIDAL / TANH**  
Smooth, differentiable, saturating functions



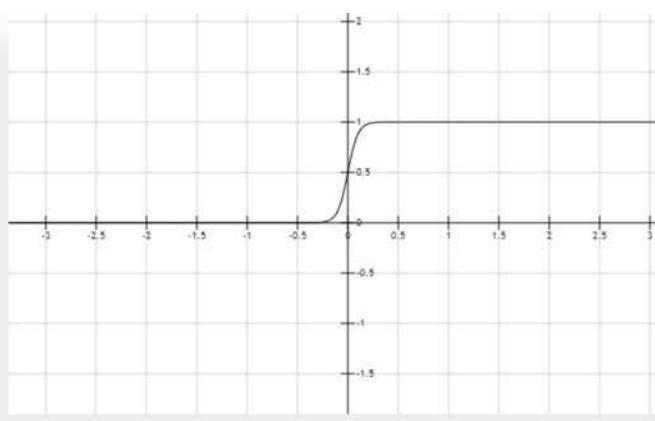
**RECTIFIED LINEAR (ReLU)**  
Cheap to compute, popular lately

# Sigmoid Function

Sigmoid function approximates a step function for large weights and approximate offset

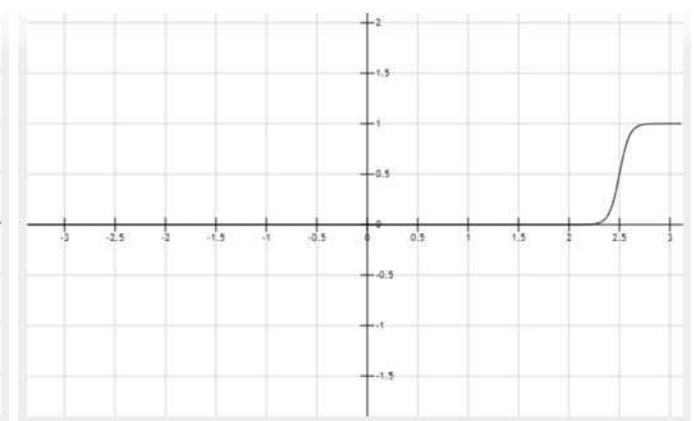


$$\frac{1}{1+e^{-x}}$$



$$\frac{1}{1+e^{-20x}}$$

<http://fooplot.com/>



$$\frac{1}{1+e^{-(20x-50)}}$$

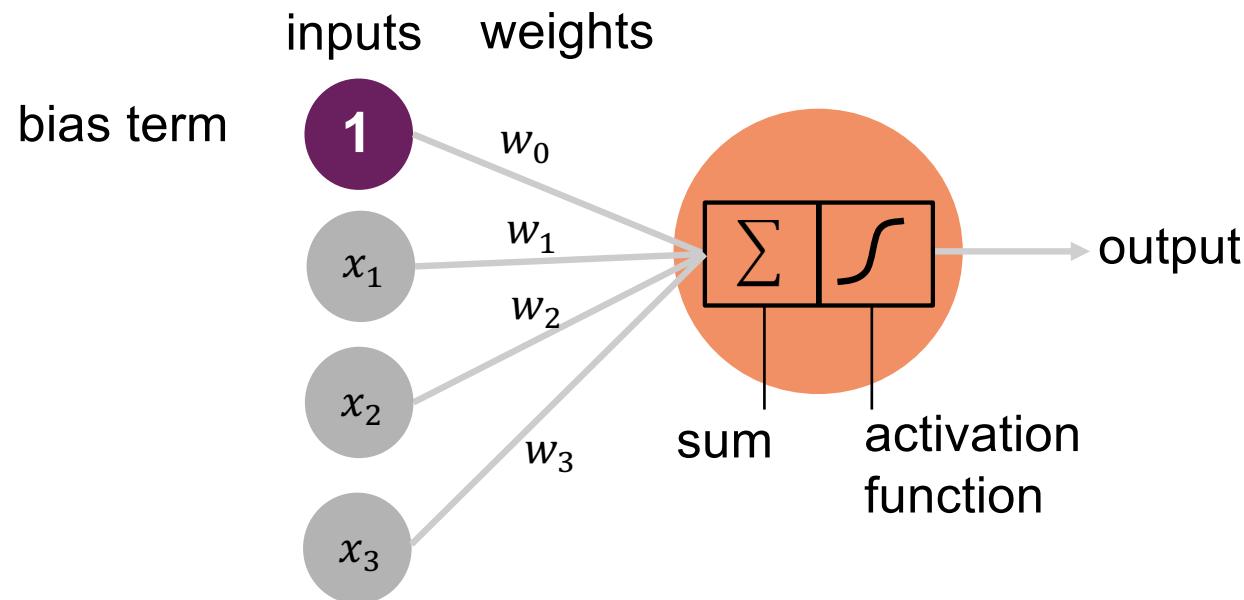
# Activation Functions

Name	Plot	Equation	Derivative
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 Unit (ReLU) <sup>[2]</sup>		$f(x) = \begin{cases} 0 & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$	$f'(x) = \begin{cases} 0 & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$
Parameteric Rectified Linear Unit (PReLU) <sup>[2]</sup>		$f(x) = \begin{cases} \alpha x & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$	$f'(x) = \begin{cases} \alpha & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$
Exponential Linear Unit (ELU) <sup>[3]</sup>		$f(x) = \begin{cases} \alpha(e^x - 1) & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$	$f'(x) = \begin{cases} f(x) + \alpha & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$
SoftPlus		$f(x) = \log_e(1 + e^x)$	$f'(x) = \frac{1}{1 + e^{-x}}$

[https://cdn-images-1.medium.com/max/1600/1\\*p\\_hyqAtyl8pbI2kEl6siOQ.png](https://cdn-images-1.medium.com/max/1600/1*p_hyqAtyl8pbI2kEl6siOQ.png)

# A Neuron

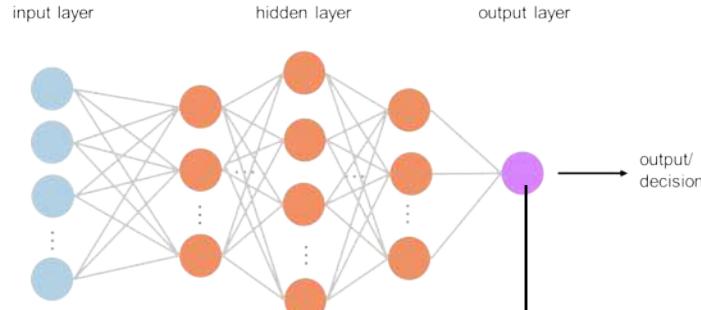
... applies an activation function to the weighted sum of its inputs



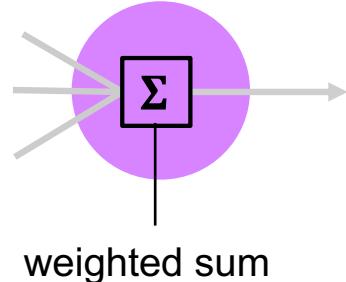
$$\text{sum: } z = w_0 + w_1 x_1 + w_2 x_2 + w_3 x_3$$

$$\text{output: } \hat{y} = \text{act}(z) = \begin{cases} 1 & \text{if } z \geq 0 \\ 0 & \text{otherwise} \end{cases}$$

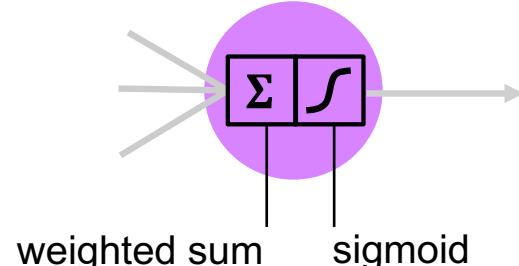
# The Output Layer



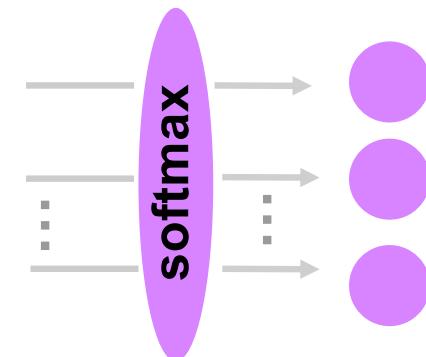
Regression:  
Linear Function



Binary Classification:  
Sigmoid



Multinomial Classification:  
Softmax



$$\hat{y}_k^{(i)} = \text{softmax}(z)_k = \frac{e^{z_k}}{\sum_{k'=1}^K e^{z_{k'}}}$$

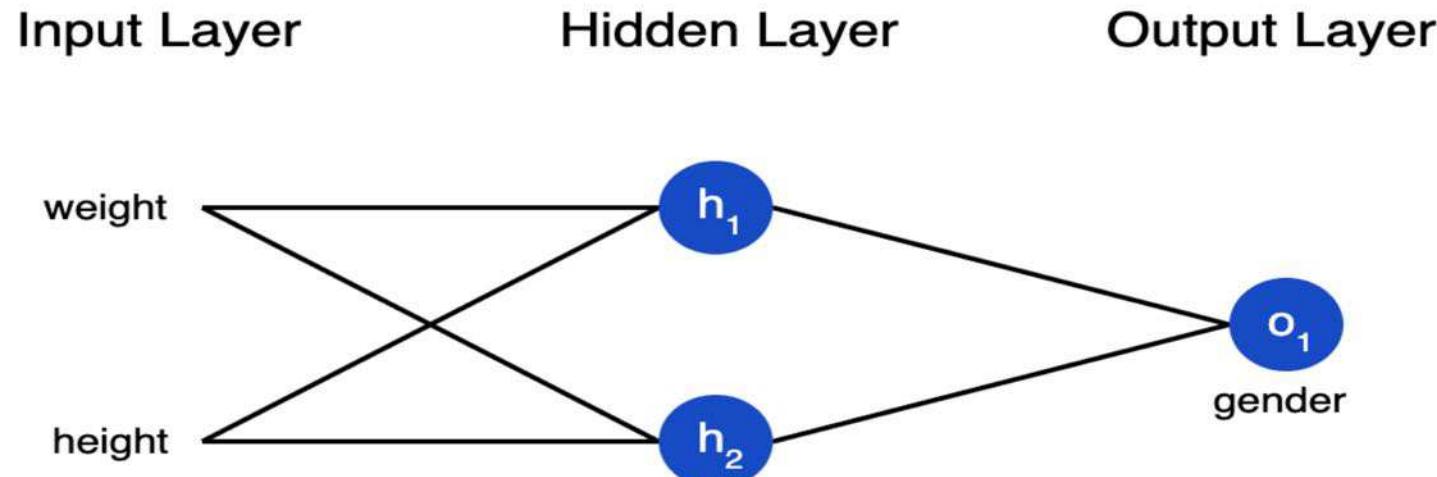
# A Simple Example

- imagine we have the below measurements

Name	Weight (lb)	Height (in)	Gender
Alice	133	65	F
Bob	160	72	M
Charlie	152	70	M
Diana	120	60	F

# A Simple Example

- let's train our network to predict someone's gender given their weight and height



# A Simple Example

- we'll represent Male with a 0 and Female with a 1, and we'll also shift the data to make it easier to use

Name	Weight (minus 135)	Height (minus 66)	Gender
Alice	-2	-1	1
Bob	25	6	0
Charlie	17	4	0
Diana	-15	-6	1

- I arbitrarily chose the shift amounts (135 and 66) to make the numbers look nice. Normally, you'd shift by the mean

# A Simple Example

- before we train our network, we first need a way to quantify how “good” it’s doing so that it can try to do “better”
  - we’ll use the mean squared error (MSE) loss 
$$\text{MSE} = \frac{1}{n} \sum_{i=1}^n (y_{true} - y_{pred})^2$$
  - $n$  is the number of samples, which is 4 (Alice, Bob, Charlie, Diana).
  - $y$  represents the variable being predicted, which is Gender.
  - $y_{true}$  is the true value of the variable (the “correct answer”). For example,  $y_{true}$  for Alice would be 1 (Female).
  - $y_{pred}$  is the predicted value of the variable. It’s whatever our network outputs.
  - $(y_{true} - y_{pred})^2$  is known as the squared error. Our loss function is simply taking the average over all squared errors (hence the name *mean squared error*). The better our predictions are, the lower our loss will be!

# A Simple Example

- Let's say our network always outputs 00 - in other words, it's confident all humans are Male 🤔. What would our loss be?

Name	$y_{true}$	$y_{pred}$	$(y_{true} - y_{pred})^2$
Alice	1	0	1
Bob	0	0	0
Charlie	0	0	0
Diana	1	0	1

$$\text{MSE} = \frac{1}{4}(1 + 0 + 0 + 1) = \boxed{0.5}$$

# A Simple Example

- we now have a clear goal: minimize the loss of the neural network. We know we can change the network's weights and biases to influence its predictions, but how do we do so in a way that decreases loss?
  - for simplicity, let's pretend we only have Alice in our dataset

Name	Weight (minus 135)	Height (minus 66)	Gender
Alice	-2	-1	1

# A Simple Example

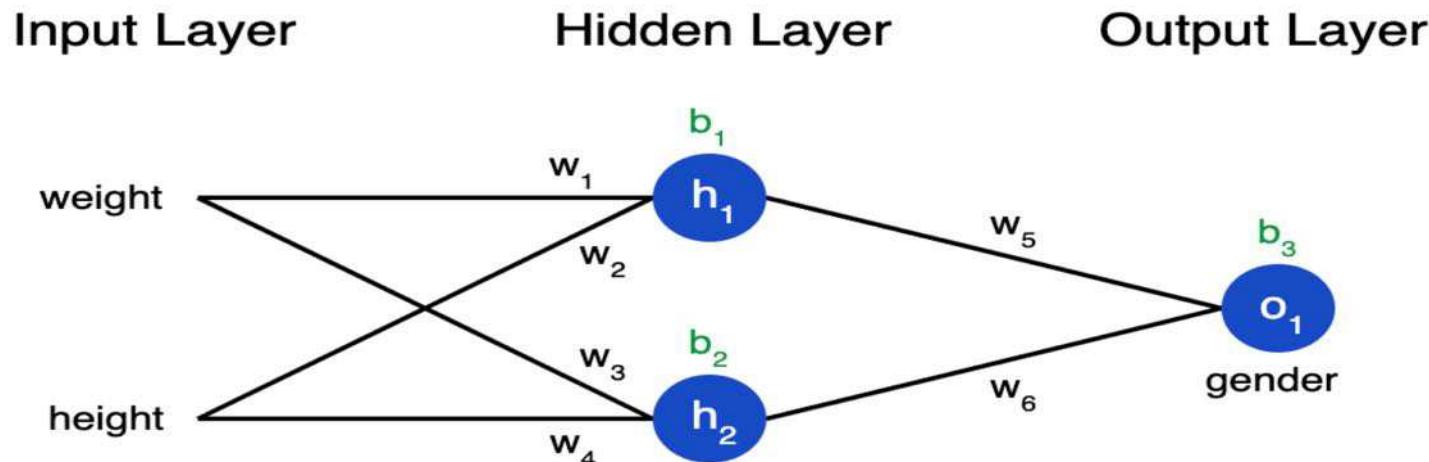
- we now have a clear goal: minimize the loss of the neural network. We know we can change the network's weights and biases to influence its predictions, but how do we do so in a way that decreases loss?
  - for simplicity, let's pretend we only have Alice in our dataset

Name	Weight (minus 135)	Height (minus 66)	Gender
Alice	-2	-1	1

- then the mean squared error loss is just Alice's squared error

# A Simple Example

- another way to think about loss is as a function of weights and biases. Let's label each weight and bias in our network



- then, we can write loss as a multivariable function

$$L(w_1, w_2, w_3, w_4, w_5, w_6, b_1, b_2, b_3)$$

# A Simple Example

- calculating the Partial Derivative, we're going to continue pretending only Alice is in our dataset

Name	Weight (minus 135)	Height (minus 66)	Gender
Alice	-2	-1	1

- let's initialize all the weights to 1 and all the biases to 0. If we do a feedforward pass through the network, we get

$$\begin{aligned} h_1 &= f(w_1x_1 + w_2x_2 + b_1) \\ &= f(-2 + -1 + 0) \\ &= 0.0474 \end{aligned}$$

$$h_2 = f(w_3x_1 + w_4x_2 + b_2) = 0.0474$$

$$\begin{aligned} o_1 &= f(w_5h_1 + w_6h_2 + b_3) \\ &= f(0.0474 + 0.0474 + 0) \\ &= 0.524 \end{aligned}$$

# A Simple Example

- The network outputs  $y_{pred} = 0.524$ , which doesn't strongly favor Male (0) or Female (1). Let's calculate  $\partial L / \partial w_1$ :

$$\frac{\partial L}{\partial w_1} = \frac{\partial L}{\partial y_{pred}} * \frac{\partial y_{pred}}{\partial h_1} * \frac{\partial h_1}{\partial w_1}$$

$$\begin{aligned}\frac{\partial L}{\partial y_{pred}} &= -2(1 - y_{pred}) \\ &= -2(1 - 0.524) \\ &= -0.952\end{aligned}$$

$$\begin{aligned}\frac{\partial y_{pred}}{\partial h_1} &= w_5 * f'(w_5 h_1 + w_6 h_2 + b_3) \\ &= 1 * f'(0.0474 + 0.0474 + 0) \\ &= f(0.0948) * (1 - f(0.0948)) \\ &= 0.249\end{aligned}$$

$$\begin{aligned}\frac{\partial h_1}{\partial w_1} &= x_1 * f'(w_1 x_1 + w_2 x_2 + b_1) \\ &= -2 * f'(-2 + -1 + 0) \\ &= -2 * f(-3) * (1 - f(-3)) \\ &= -0.0904\end{aligned}$$

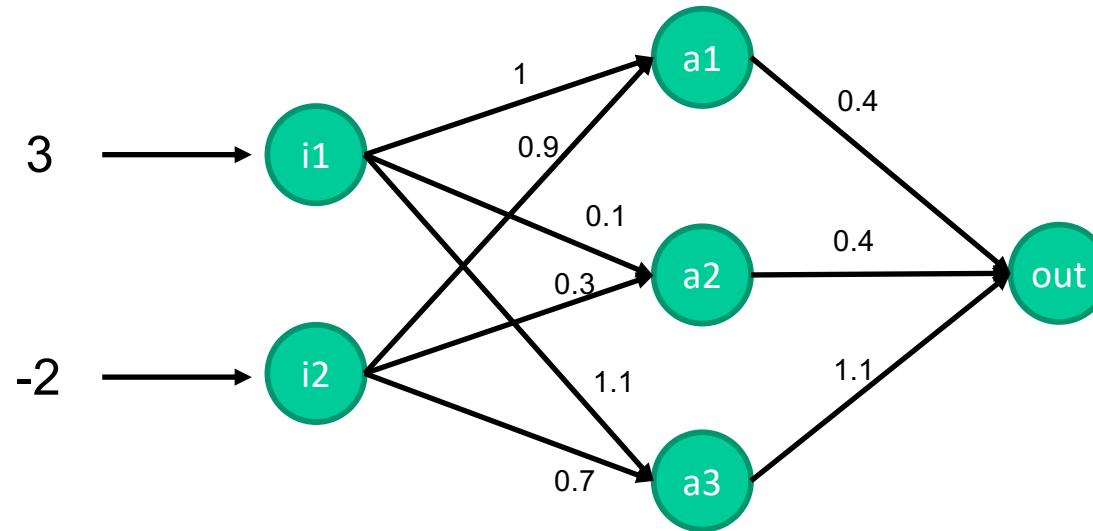
$$\begin{aligned}\frac{\partial L}{\partial w_1} &= -0.952 * 0.249 * -0.0904 \\ &= 0.0214\end{aligned}$$

- We did it! This tells us that if we were to increase  $w_1$ ,  $L$  would increase a tiny bit as a result

# A Simple Example

- our training process will look like this
  - 1. choose **one sample** from our dataset. This is what makes it **stochastic gradient descent** - we **only operate on one sample at a time**.
  - 2. **calculate all the partial derivatives** of loss with **respect to weights or biases**
  - 3. Use the **update equation** to update each weight and bias.
  - 4. **Go back to step 1.**
- See V12\_NeuronalNetwork.ipynb for more an implementation and more details...

# Self-Study: Another Example

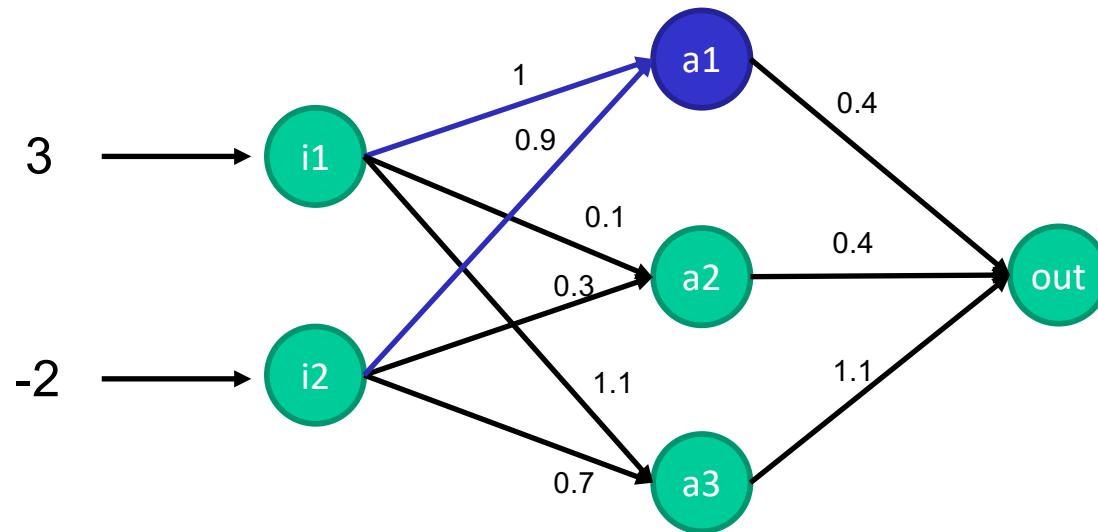


Activation function  
for all nodes (also out):

$$\text{act}(z) = \begin{cases} 1 & \text{if } z \geq 0 \\ 0 & \text{otherwise} \end{cases}$$

The bias ( $w_0$ ) for all nodes is the same:  $w_{a1,0} = w_{a2,0} = w_{a3,0} = w_{\text{out},0} = 0.2$

# Self-Study: Another Example



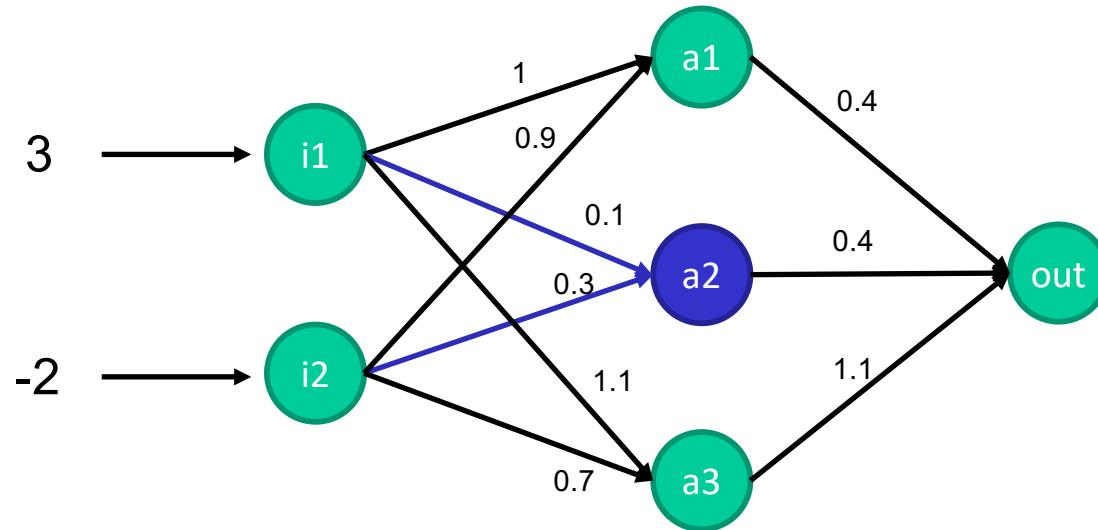
$$\text{act}(z) = \begin{cases} 1 & \text{if } z \geq 0 \\ 0 & \text{otherwise} \end{cases}$$

The bias ( $w_0$ ) for all nodes is the same:  $w_{a1,0} = w_{a2,0} = w_{a3,0} = w_{out,0} = 0.2$

$$z_1 = i_1 \cdot 1 + i_2 \cdot 0.9 + w_{a1,0} = 3 \cdot 1 - 2 \cdot 0.9 + 0.2 = 1.4$$

$$a_1 = \text{act}(z_1) = \text{act}(1.4) = 1.4$$

# Self-Study: Another Example



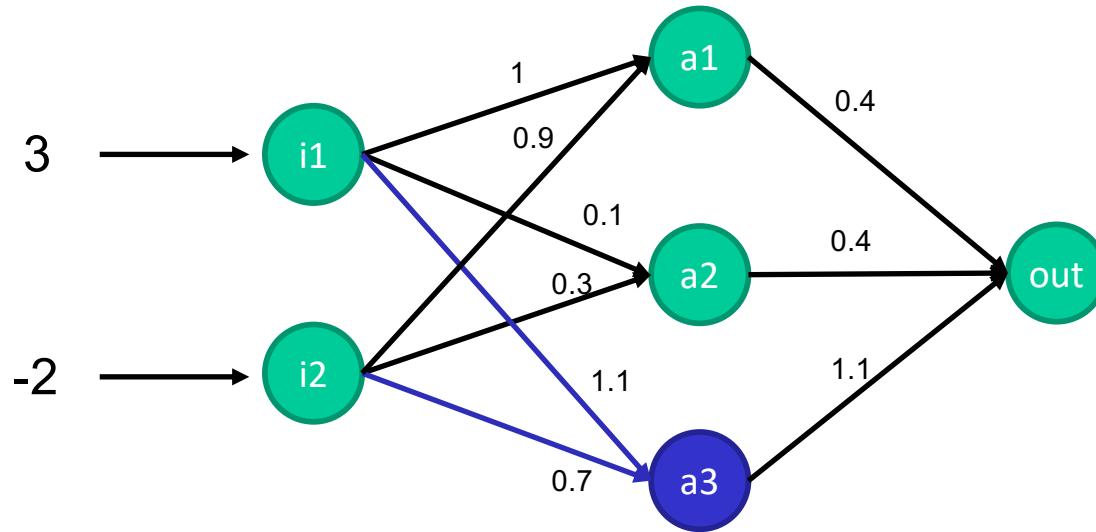
$$\text{act}(z) = \begin{cases} 1 & \text{if } z \geq 0 \\ 0 & \text{otherwise} \end{cases}$$

The bias ( $w_0$ ) for all nodes is the same:  $w_{a1,0} = w_{a2,0} = w_{a3,0} = w_{out,0} = 0.2$

$$z_2 = i_1 \cdot 0.1 + i_2 \cdot 0.3 + w_{a2,0} = 3 \cdot 0.1 - 2 \cdot 0.3 + 0.2 = -0.1$$

$$a_2 = \text{act}(z_2) = \text{act}(-0.1) = 0$$

# Self-Study: Another Example



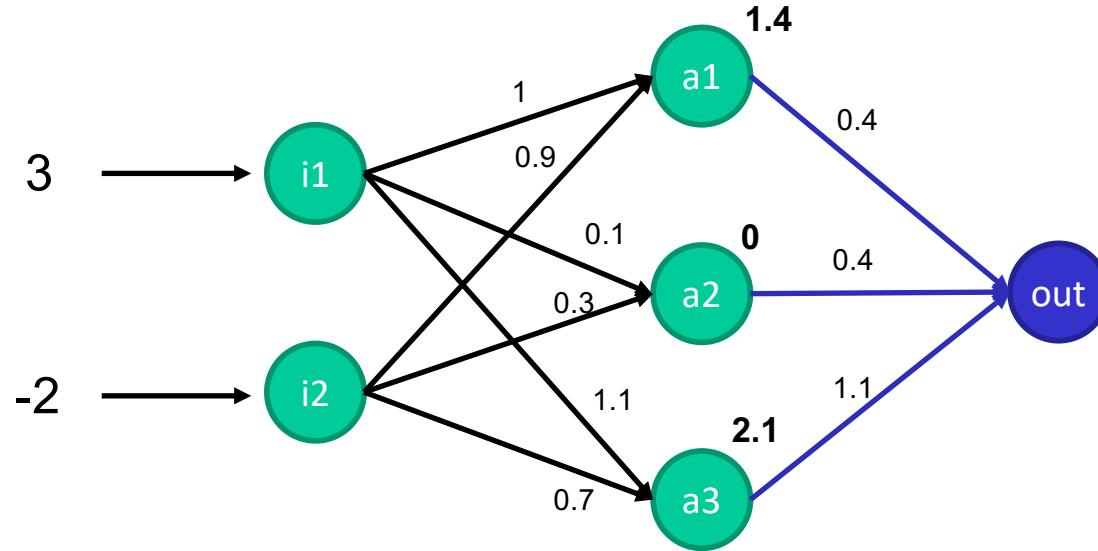
$$\text{act}(z) = \begin{cases} 1 & \text{if } z \geq 0 \\ 0 & \text{otherwise} \end{cases}$$

The bias ( $w_0$ ) for all nodes is the same:  $w_{a1,0} = w_{a2,0} = w_{a3,0} = w_{out,0} = 0.2$

$$z_3 = i_1 \cdot 1.1 + i_2 \cdot 0.7 + w_{a3,0} = 3 \cdot 1.1 - 2 \cdot 0.7 + 0.2 = 2.1$$

$$a_3 = \text{act}(z_3) = \text{act}(2.1) = 2.1$$

# Self-Study: Another Example



$$\text{act}(z) = \begin{cases} 1 & z \geq 0 \\ 0 & \text{otherwise} \end{cases}$$

The bias ( $w_0$ ) for all nodes is the same:  $w_{a1,0} = w_{a2,0} = w_{a3,0} = w_{out,0} = 0.2$

$$\begin{aligned} z_{out} &= a_1 \cdot 0.4 + a_2 \cdot 0.4 + a_3 \cdot 2.1 + w_{out,0} = 1.4 \cdot 0.4 + 0 \cdot 0.4 + 2.1 \cdot 1.1 + 0.2 = 3.07 \\ \text{out} &= \text{act}(z_{out}) = \text{act}(3.07) = 3.07 \end{aligned}$$

# The Cost Function

## Regression: Mean Squared Error

$$MSE = \frac{1}{N} \sum_{i=1}^N (y^{(i)} - \hat{y}^{(i)})^2$$

## Classification: Cross Entropy Error

(formula for one sample  $x^{(i)}$ )

- Binary (logistic):

$$\mathcal{L}_{\text{CE}}(\hat{y}^{(i)}, y^{(i)}) = -[y^{(i)} \log \hat{y}^{(i)} + (1 - y^{(i)}) \log(1 - \hat{y}^{(i)})]$$

- Multinomial (softmax):

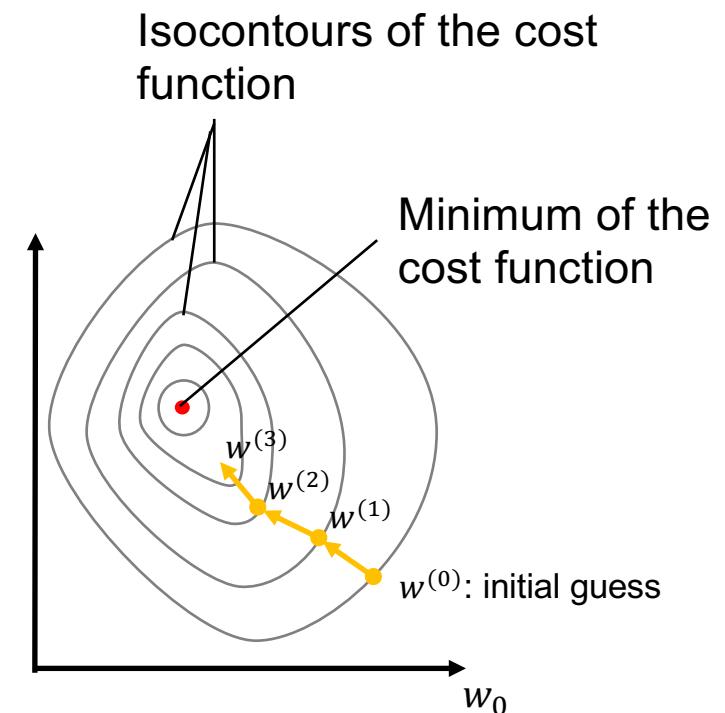
$$\mathcal{L}_{\text{CE}}(\hat{\mathbf{y}}^{(i)}, \mathbf{y}^{(i)}) = -\sum_{k=1}^K y^{(i)}_k \log \hat{y}^{(i)}_k = -\log \frac{\exp(w_c^T x^{(i)})}{\sum_{k=1}^K \exp(w_k^T x^{(i)})} \quad (\text{where } c \text{ is the correct class})$$

The total cross entropy error is the sum over all training samples (in the batch)

## Minimisation by Gradient Descent

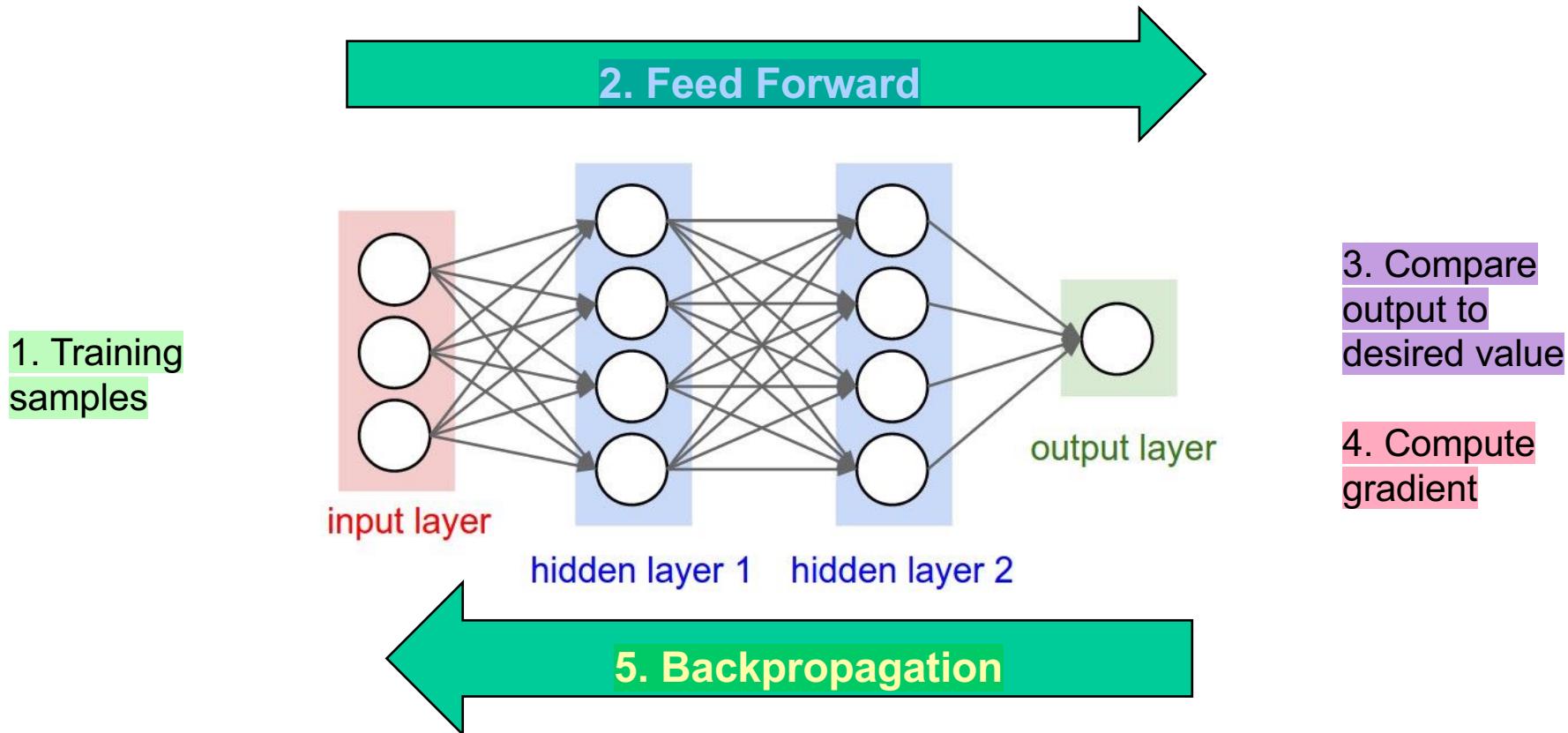
- Initial guess for the model parameters: the weights
- Repeat until stopping criterium is reached:
  - compute gradient of the cost function with respect to parameters
  - adjust the parameters in the opposite direction of the gradient by a small step, i.e. the learning rate  $\alpha$

$$w \leftarrow w - \alpha \frac{1}{N} \frac{\partial \mathcal{L}_{CE}}{\partial w}$$



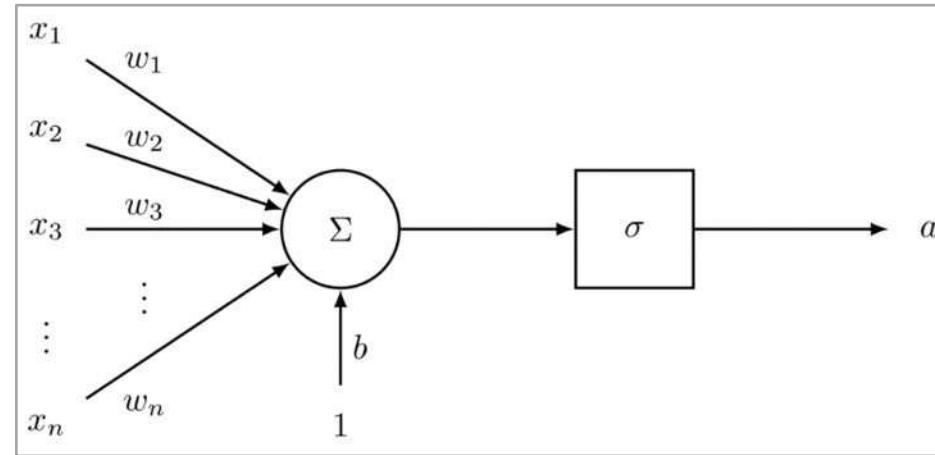
# Training in Feedforward Neural Networks

Parameter (the weights) optimisation by gradient descent



[https://www.pyimagesearch.com/wp-content/uploads/2016/08/simple\\_neural\\_network\\_header.jpg](https://www.pyimagesearch.com/wp-content/uploads/2016/08/simple_neural_network_header.jpg)

# Key Factor of Success: Differentiability

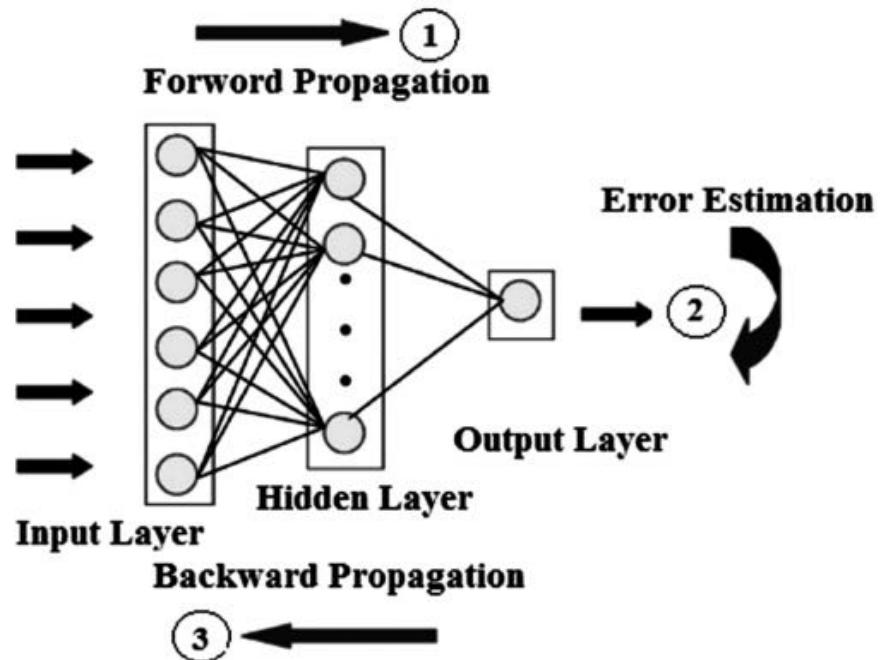


If **all components** in a neural network are **differentiable**, we can efficiently compute the gradient of the **cost function** with respect to the **weights** and optimise using **gradient descent**

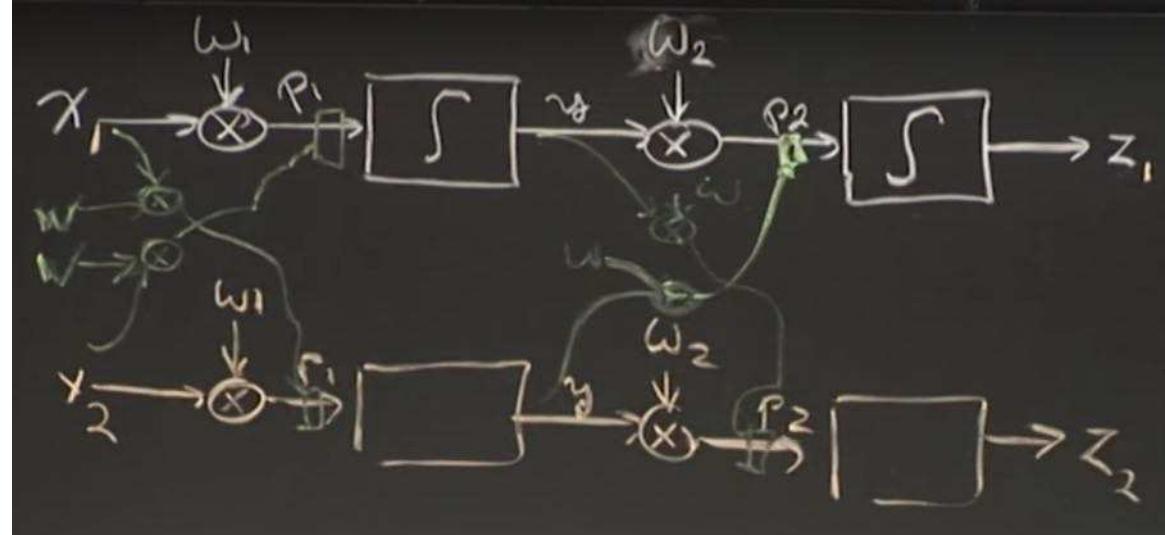
# Backpropagation

- First described by Werbos 1974
- Re-discovered by Parker 1982 and **Rumelhart et al. 1986**

Idea: Compute gradient of cost function and propagate changes back to the weights.



# Backpropagation: Re-use of Partial Derivatives



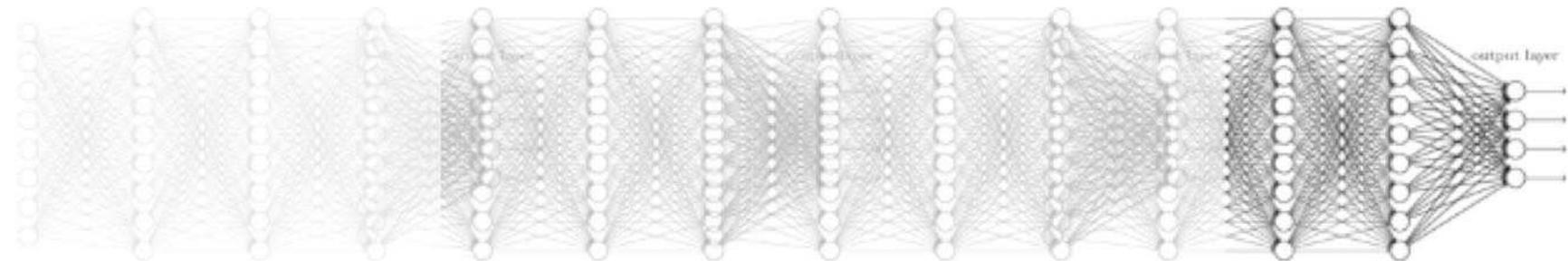
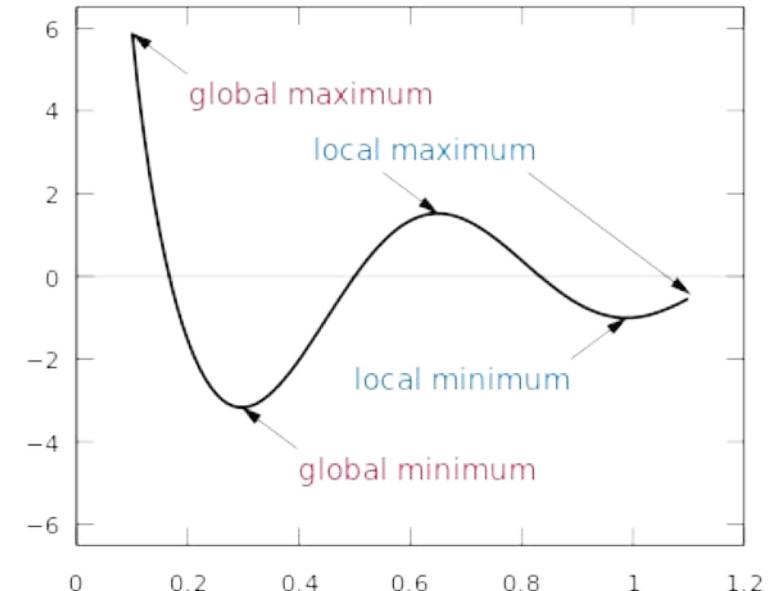
Main Idea: Re-use partial derivatives to compute gradients efficiently

$$\frac{\partial P}{\partial w_2} = \frac{\partial P}{\partial z} * \frac{\partial z}{\partial p_2} * \frac{\partial p_2}{\partial w_2} = \frac{\partial p_2}{\partial w_2} * \frac{\partial z}{\partial p_2} * \frac{\partial P}{\partial z}$$

$$\frac{\partial P}{\partial w_1} = \dots = \frac{\partial p_1}{\partial w_1} * \frac{\partial y}{\partial p_1} * \frac{\partial p_2}{\partial y} * \frac{\partial z}{\partial p_2} * \frac{\partial P}{\partial z}$$

# Disadvantages of Backpropagation

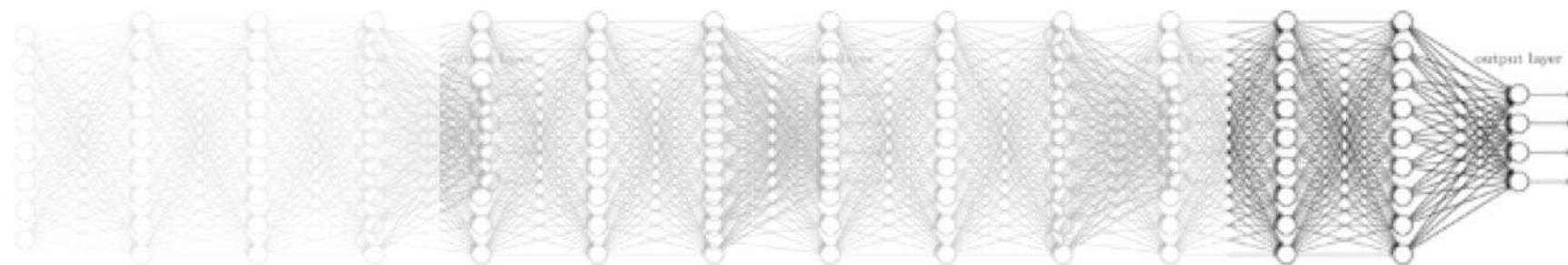
- Requires large amount of labeled training data
- Learning time is slow for multiple hidden layers
- Can get stuck in local optima
- Vanishing Gradient Problem



# Vanishing Gradient Problem

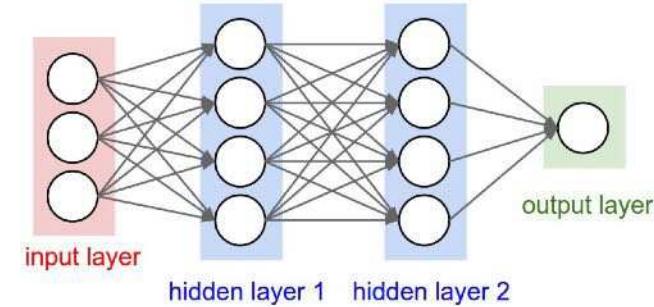
Sepp Hochreiter 1991

- Observation: if we have many hidden layers, then changes in the first layers will diminish since the gradient becomes extremely small
- Reason: if all partial derivatives are between -1 and 1, then multiplying them makes them exponentially small
- Depths of a network can be easily up to 150 layers
- In practice, this results in very slowly changing weights, thus, very long training times
- Solution: do not use randomly initialized weights, but rather "carefully pre-trained weights"

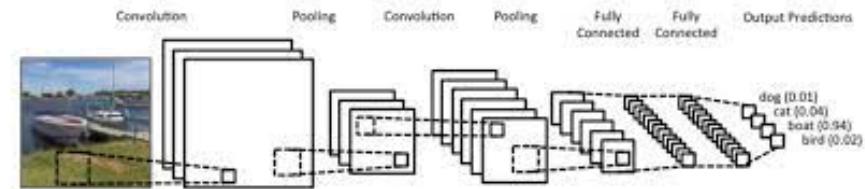


# Architectures for Neural Networks

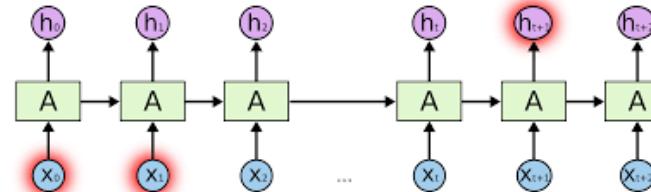
## Vanilla Neural Networks



## Convolutional Neural Networks



## Recurrent Neural Networks



## Transformers

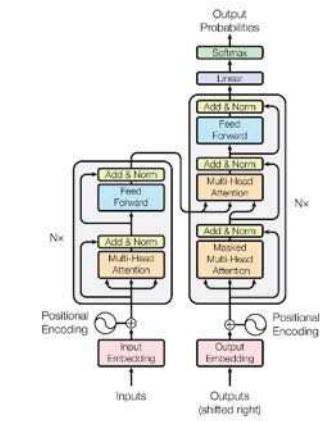


Figure 1: The Transformer - model architecture.

# Frameworks for Neural Networks



Hugging Face



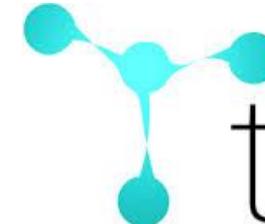
Keras



TensorFlow



theano



DEEPMLEARNING4J

Caffe

## Semester-Endprüfung SEP

### Prüfungsmodus:

- 90 Minuten in Moodle
- Prüfungsstoff: Vorlesung und Praktika
- Open book
- Keine Kommunikation während der Prüfung

### **Note:**

- 20% Labs + 80% SEP

## Dual-Choice Tasks

Einige Aufgaben sind als **Dual-Choice-Aufgaben** gestellt.

Kreuzen Sie in der Tabelle an, ob die jeweiligen Aussagen wahr oder falsch sind.

Jede richtige Antwort gibt 0.5 Punkte; jede falsche Antwort gibt 0.5 Punkte Abzug; keine Antwort gibt keinen Punkt.

Ein negatives Punktesaldo in einer Aufgabe wird auf 0 Punkte aufgerundet.

# SEP Sample Questions

<b>temperature</b>	x	10	15	17	23	25	29	31	32
<b>sold ice cream</b>	y	15	22	28	35.914	38	44	48	55

→ Füllen Sie den fehlenden Wert mit der mean Methode und sagen Sie mit der linearen regression den Wert für "sold ice cream" bei einer Temperatur von 20 Grad voraus.

30.946 bei 20 Grad

# SEP Sample Questions

<b>id</b>	<b>title</b>	<b>date</b>	<b>rating</b>	<b>episode</b>	<b>season</b>	<b>length</b>
1	live free or die	01.11.2012	9.3	1	5	43
2	madrigal	06.12.2013	89 faktor 10	2	5	48
3	sunset	25.10.2011	9.3	6	3	47
4	grilled	20.12.2009	9.3	2	2	46
5	down	42.ungültig	42.12.2009	8.3	4	2
						333 viel länger
6	cancer man	19.10.2008	8.3	4	1	48
7	live fre or die	01.11.2012	9.3	1	5	43

<b>firstname</b>	<b>lastname</b>	<b>bdate</b>	<b>age</b>	<b>gender</b>	<b>phone</b>
bryan	cranston	03-07-1956	65	1	999-9999 vermutlich placeholder
aaron	paul	27-08-1979	44	m nicht nummer	777-53474
anna, gunn	gunn	08-11-1968	52	0	040-15627

vorname, nachname

→ Finden Sie 5 Fehler in den Daten und beschreiben Sie die Fehler.

# SEP Sample Questions

TID	Artikel_1	Artikel_2	Artikel_3	Artikel_4	Artikel_5	Artikel_6
1	gin	tonic	chips	bier	rum	cola
2	chips	gin	cola	mehl	tonic	seife
3	kerzen	cola	rum	mehl	seife	eier
4	gin	kerzen	tonic	rum	seife	bier
5	rum	cola	bier	kerzen	eier	mehl
6	tonic	kerzen	eier	seife	mehl	cola
7	tonic	gin	erbsen	seife	eier	brot
8	seife	rum	butter	cola	brot	mehl
9	seife	kerzen	butter	brot	cola	käse
10	gin	tonic	brot	mehl	eier	cola
11	bier	eier	tonic	gin	erbsen	cola
12	fanta	chips	pizza	honig	brot	eier
13	fanta	tonic	gin	honig	cola	chips
14	pizza	fanta	chips	cola	tonic	gin
15	pizza	chips	fanta	honig	gin	cola

# SEP Sample Questions

$$\text{precision} = \text{TP} / (\text{TP} + \text{FP}) = 20 / (20 + 60) = 20 / 60 = 2/6 = 1/3$$

$$\text{recall} = \text{TP} / (\text{TP} + \text{FN}) = 20 / (20 + 60) = 20 / 80 = 1/4$$

$$\text{F-Score} = 2 * ((\text{precision} * \text{recall}) / (\text{precision} + \text{recall})) = 2 * ((1/12) / (7/12)) = 2 * (1/7) = 2/7$$

	relevant	not relevant
returned	20	40
not returned	60	1,000,000

	relevant	not relevant
returned	50	10
not returned	30	1,000,000

$$\text{precision} = 50 / (50 + 10) = 50 / 60 = 5/6$$

$$\text{recall} = 50 / (50 + 30) = 50 / 80 = 5/8$$

$$\text{F-Score} = 2 * ((\text{precision} * \text{recall}) / (\text{precision} + \text{recall})) = 2 * ((5/48) / (70/48)) = 2 * (25/70) = 10/14$$

→ Berechnen Sie Precision, Recall und FScore für die Werte in den Tabellen.

# SEP Sample Questions

possible items to recommend to 2nd person (not yet rated by him):

- cherry
- banana
- avocado

cherry:  
 $(0.749 * 4) / (0.749) = 4$

banana:  
 $(0.913 * 3) / (0.913) = 3$

avocado:  
 none = 0

2nd most sim  
 $(0.749)$

wanted

most sim  
 $(0.913)$



4	3			5		
5		4		4		
4		5	3	4		
	3				5	
	4				4	
		2	4		5	

Similarity Matrix User-Based:

```
[[1. 0.749 0.627 0.218 0.3 0. ]
 [0.749 1. 0.913 0. 0. 0.158]
 [0.627 0.913 1. 0. 0. 0.404]
 [0.218 0. 0. 1. 0.97 0.639]
 [0.3 0. 0. 0.97 1. 0.527]
 [0. 0.158 0.404 0.639 0.527 1. ]]
```

→ Welche Produkte werden der Person in der 2. Zeile vorgeschlagen wenn man user-based collaborative Filtering mit den 2 ähnlichsten Personen verwendet?

# SEP Sample Questions

ID	Transaction Items				List
T1	coke	chips			ice {A, B, E}
T2		chips		beer	{B, D}
T3		chips	whiskey		{B, C}
T4	coke	chips		beer	{A, B, D}
T5	coke		whiskey		{A, C}
T6		chips	whiskey		{B, C}
T7	coke		whiskey		{A, C}
T8	coke	chips	whiskey		ice {A, B, C, E}
T9	coke	chips	whiskey		{A, B, C}

→ Erstellen Sie den FP-Tree zu den Daten in der Tabelle.

# SEP Sample Questions

Gegeben sind die Abstände zwischen den Datenpunkten a, b, c, d, e, f, g.

Geben Sie die Reihenfolge des Ablaufes (inklusive den Distanzen) des Hierarchical Clustering mit Single Linkage an.  
schritt 1: merge(a,g) → distanz(11)

Beispiel:

- schritt 1: merge(e,d) --> distanz(17)
- schritt 2: merge(ed, f) --> distanz(27)
- ....

a	b	c	d	e	f	g
a	0	50	63	17	72	81
b		0	49	41	42	54
c			0	52	13	16
d				0	56	66
e					0	12
f						0
g						0

ag

schritt 2: merge(e,f) → distanz(12)

ef

schritt 3: merge(ef,c) → distanz(13)

efc

schritt 4: merge(d, ag) → distanz(15)

dag

schritt 5: merge(dag, b) → distanz(37)

dagb

schritt 6: merge(dagb, efc) → distanz(42)

dagbefc

# SEP Sample Questions

Nehmen Sie an sie haben den folgenden Datensatz: (2,2), (4,4), (5,5), (6,6), (9,9) (0,4), (4,0) und der Clustering Algorithmus KMeans wird mit k=3 ausgeführt um die Daten zu clustern. Weiterhin wird die Manhattan Distanz (siehe Abbildung) als Distanzfunktion verwendet um die Distanz zwischen den Zentroiden und den Datenpunkten zu berechnen. Die gebildeten Cluster C1, C2, C3 nach der 1. Iteration sind wie folgt:

C1: {(2,2), (4,4), (6,6)}

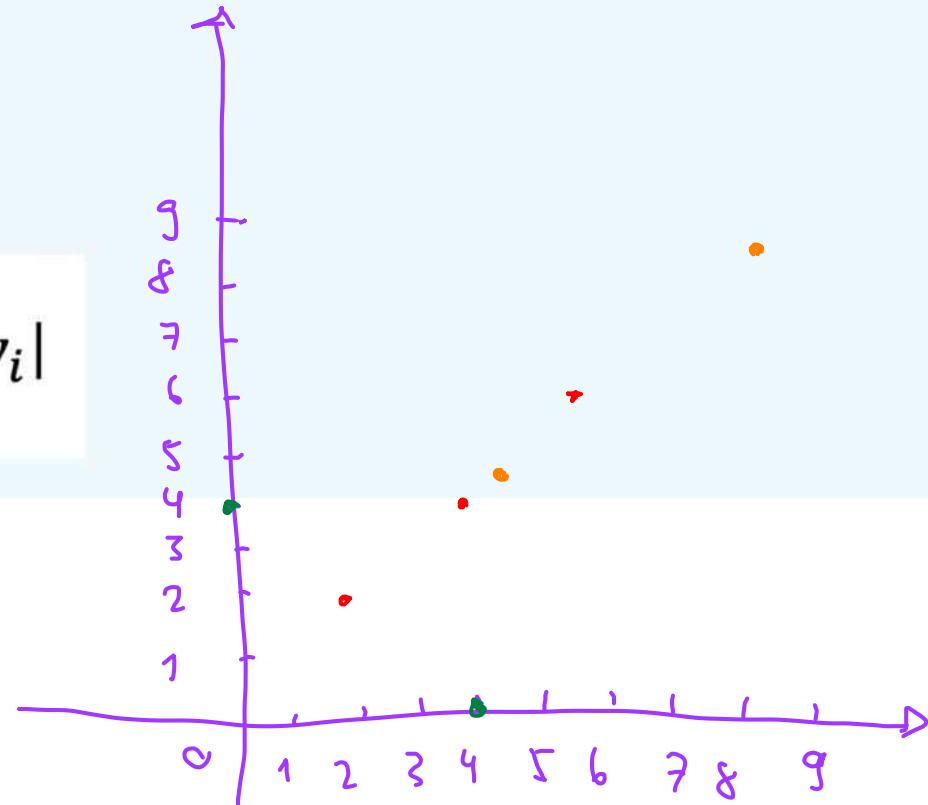
C2: {(0,4), (4,0)}

C3: {(5,5), (9,9)}

Geben Sie die Zentroide zu den 3 Clustern an.

$$dist(x, y) = \sum_{i=1}^d |x_i - y_i|$$

k1 = (4, 4)  
k2 = (2, 2)  
k3 = (7, 7)



# SEP Sample Questions

Sie haben die folgenden Daten und Distanzen zwischen den Punkten gegeben.

Wählen Sie das korrekte Dendrogramm, welches durch den Einsatz des Single Linkage Verfahrens entstanden ist.

point	x coordinate	y coordinate
p1	0.4005	0.5306
p2	0.2148	0.3854
p3	0.3457	0.3156
p4	0.2652	0.1875
p5	0.0789	0.4139
p6	0.4548	0.3022

	p1	p2	p3	p4	p5	p6
p1	0.0000	0.2357	0.2218	0.3688	0.3421	0.2347
p2	0.2357	0.0000	0.1483	0.2042	0.1388	0.2540
p3	0.2218	0.1483	0.0000	0.1513	0.2843	0.1100
p4	0.3688	0.2042	0.1513	0.0000	0.2932	0.2216
p5	0.3421	0.1388	0.2843	0.2932	0.0000	0.3921
p6	0.2347	0.2540	0.1100	0.2216	0.3921	0.0000

1.  $(p_3 \quad p_6)$

2.  $(p_2 \quad p_5)$

3.  $(p_2 \quad p_5 \quad p_3 \quad p_6)$

4.  $(p_2 \quad p_5 \quad p_3 \quad p_4 \quad p_6)$

5.  $(p_2 \quad p_5 \quad p_3 \quad p_4 \quad p_6)$

6.  $(p_2 \quad p_5 \quad p_3 \quad p_4 \quad p_6 \quad p_1)$

