

Course

Introduction to Data Analytics in Business

Prof. Dr. Peter Roßbach

WS 2019

Agenda

1 Data Analytics

- 1.1 What is Data Science?
- 1.2 Statistics and Machine Learning
- 1.3 Data Preparation
- 1.4 Exploratory Data Analysis

2 Methods, Algorithms, and Applications

2.1 Classification Methods

- 2.1.1 k-Nearest Neighbors
- 2.1.2 Evaluating Classifier Performance
- 2.1.3 Naive Bayes Classifier
- 2.1.4 Decision Trees
- 2.1.5 Discriminant Analysis
- 2.1.6 Logistic Regression
- 2.1.7 Neural Networks
- 2.1.8 Support Vector Machines
- 2.1.9 Ensemble Methods

Agenda

2.2 Regression

- 2.2.1 Ordinary Least Squares Regression
- 2.2.2 Ridge Regression
- 2.2.3 Support Vector Regression
- 2.2.4 Neural Networks
- 2.2.5 Regression based on Decision Trees

2.3 Interpretable Machine Learning

2.4 Segmentation

- 2.4.1 Clusteranalysis
- 2.4.2 Dimensionality Reduction
 - 2.4.2.1 Self-organizing Maps
 - 2.4.2.2 t-distributed Stochastic Neighbor Embedding (t-SNE)

2.5 Association Analysis

Literature

General Introduction:

Alpaydin, E. (2016): Machine Learning: The New AI, MIT Press Essential Knowledge

Schutt, R.; O'Neil, C. (2013): Doing Data Science, O'Reilly Media

Methods and Algorithms:

Alpaydin, E. (2016): Introduction to Machine Learning, Third Edition, MIT Press

Hastie, T.; Tibshirani, R.; Friedman, J. (2009): The Elements of Statistical Learning, Second Edition, Springer

Implementation:

Aurélien Géron (2017): Hands-On Machine Learning with Scikit-Learn and TensorFlow: Concepts, Tools, and Techniques To Build Intelligent Systems, O'Reilly

Raschka, Sebastian (2015): Python Machine Learning, Packt Publishing

Introduction

Key Statements

- Organizations do not need a big data strategy; they need a business strategy that incorporates big data.
- The days when business leaders could turn data analytics over to IT are over; tomorrow's business leaders must embrace analytics as a business discipline in the same vein as accounting, finance, management science, and marketing.
- The key to data monetization and business transformation lies in unleashing the organization's creative thinking; we have got to get the business users to "think like a data scientist".
- The business potential of big data is only limited by the creative thinking of the business users.
- New technologies do not disrupt business models; it's what organizations do with these new technologies that disrupts business models and enables new ones.

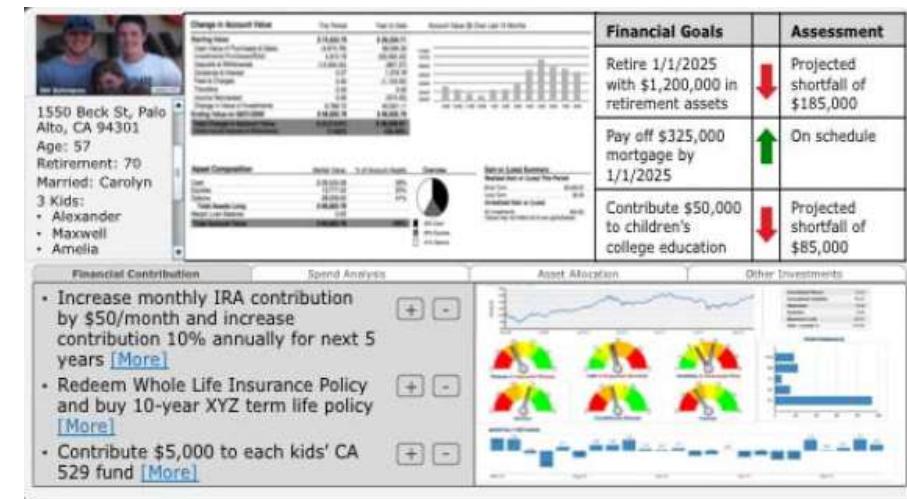
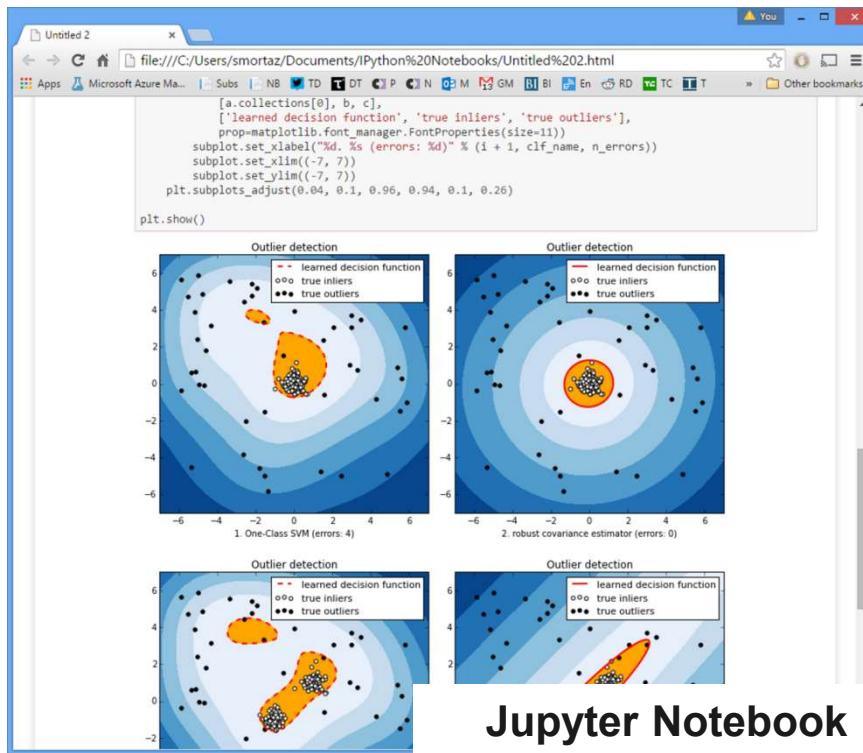
-- Bill Schmarzo

Future of Data Scientists

Data scientist roles are expected to grow over the next 5 years. The use of basic dashboards and notebooks (like Jupyter) will become as common as the use of email and excel, and that a notebook connected to real-time data will kill off the spreadsheet and static powerpoint presentations. Expect business presentations to shift from static powerpoint slide decks to dynamic dashboards connected to real-

time data, one or more predictive models.

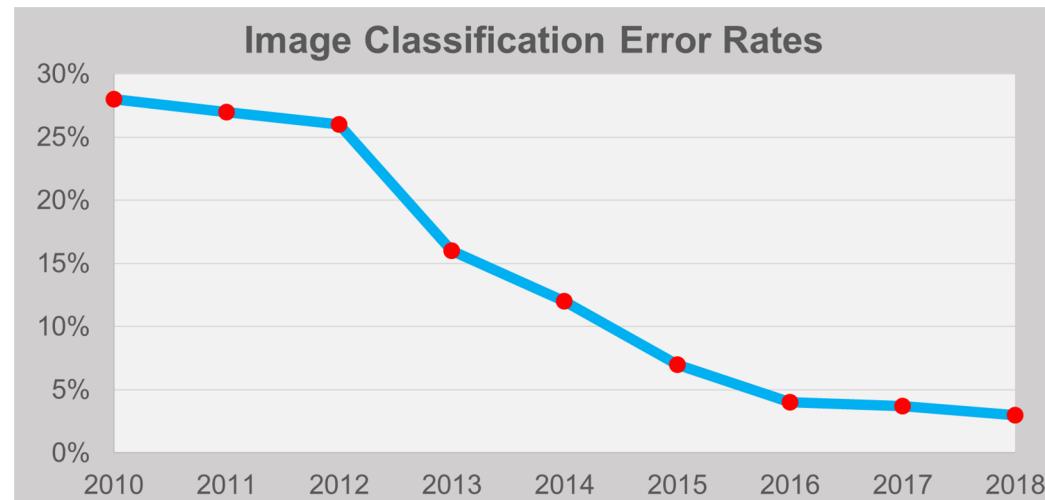
-- IBM



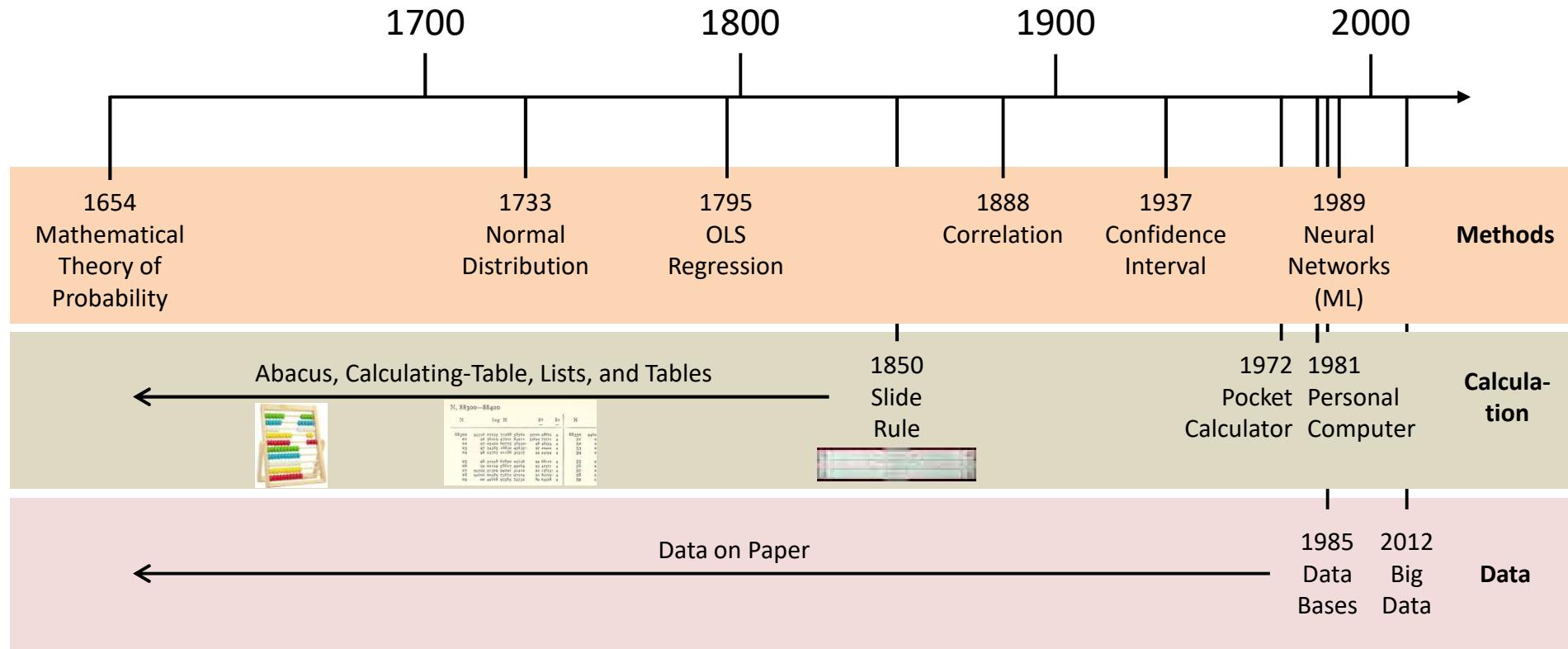
Dashboard

Why is Modern Analytics so Successful?

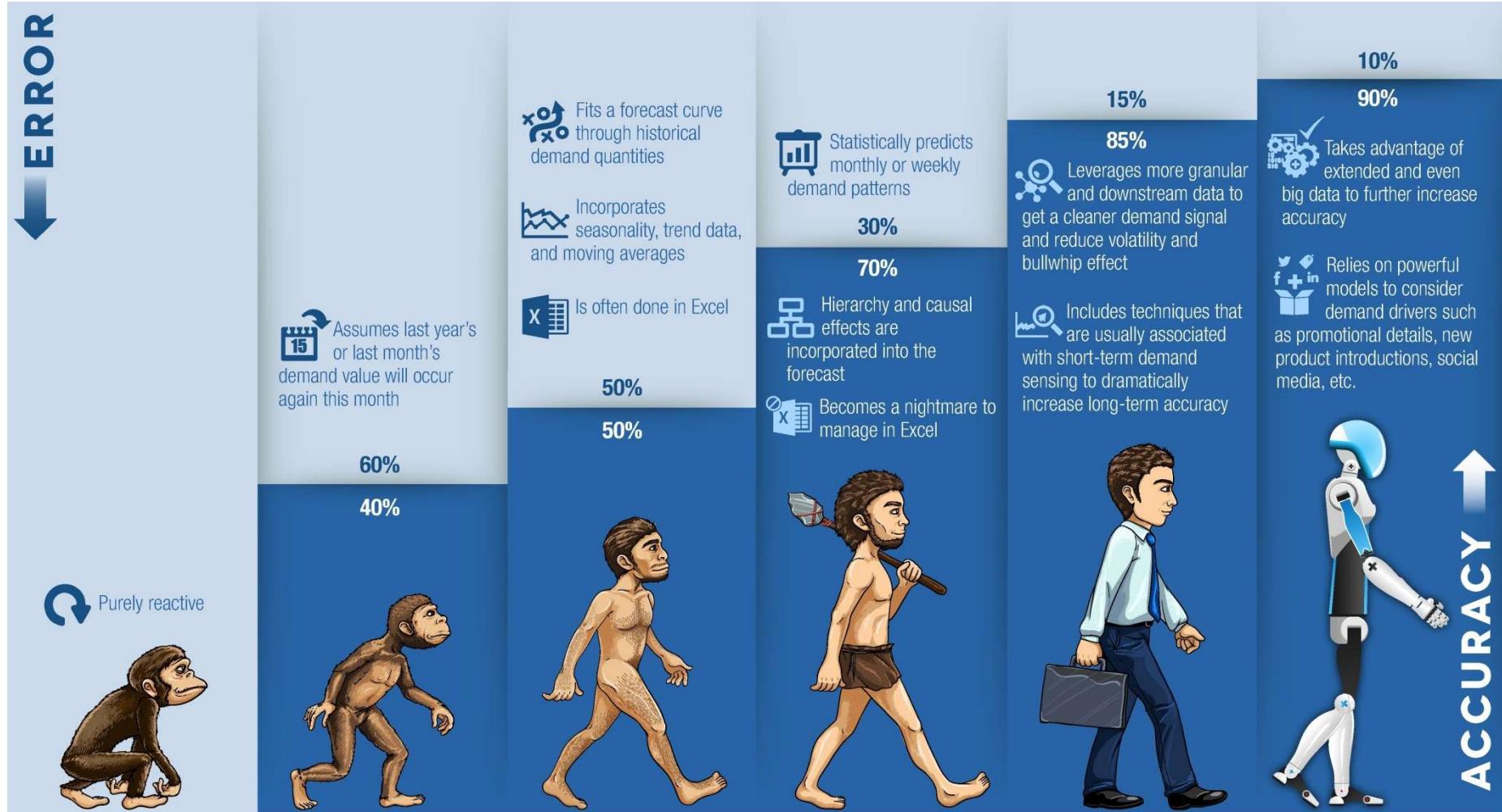
- 1. More Data for the Analysis**
- 2. More Computing Power**
- 3. New Methods and Algorithms**
- 4. New Analysis Processes**



From the Past to the Present



Evolution of Analytics



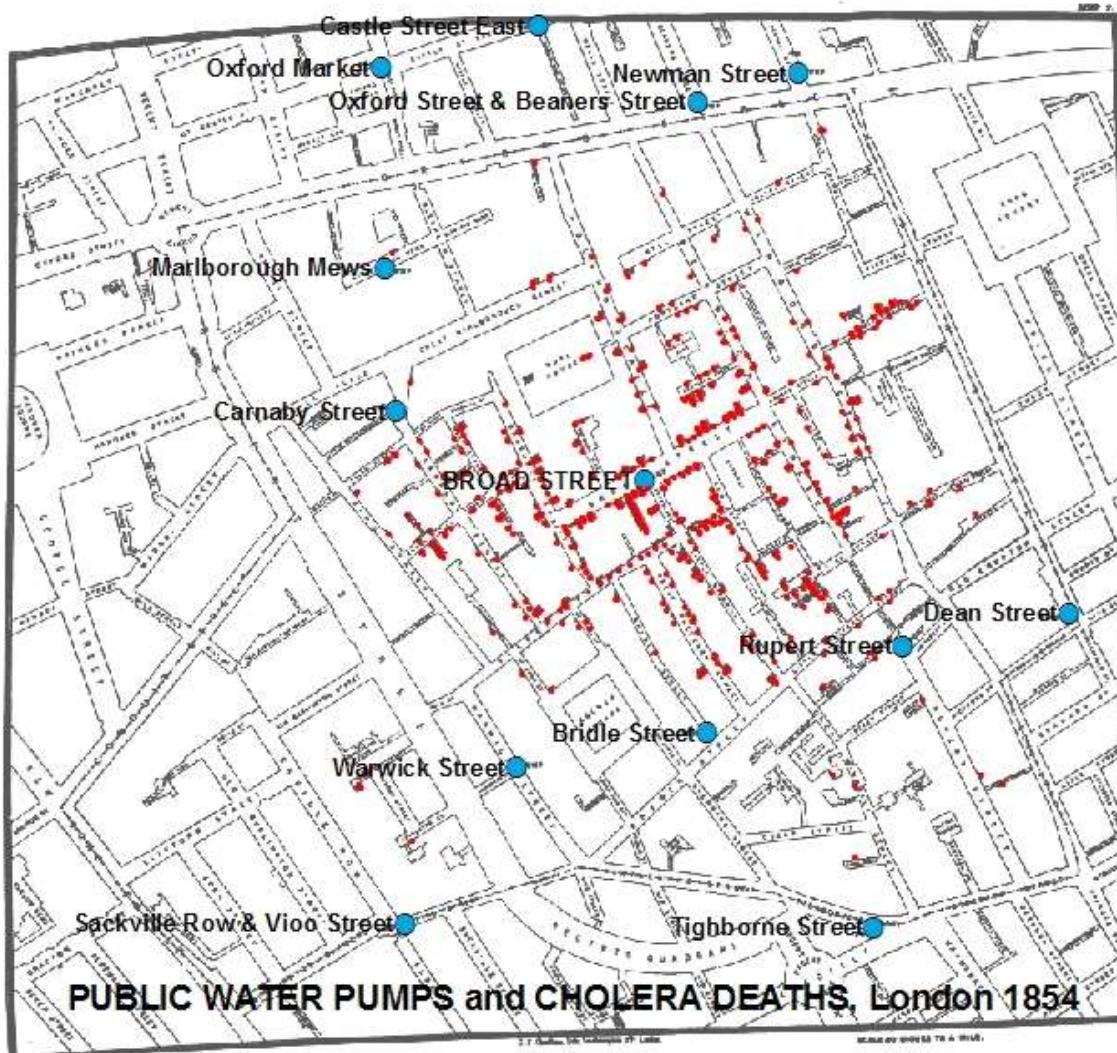
Source: <http://juxt.pro/blog/posts/machine-learning-with-clojure.html>

1 Data Analytics

What is Data Science?

First Application of Data Analytics

Dr. John Snow's Map of the 1854 London Cholera Outbreak



Evolution of the Business Questions

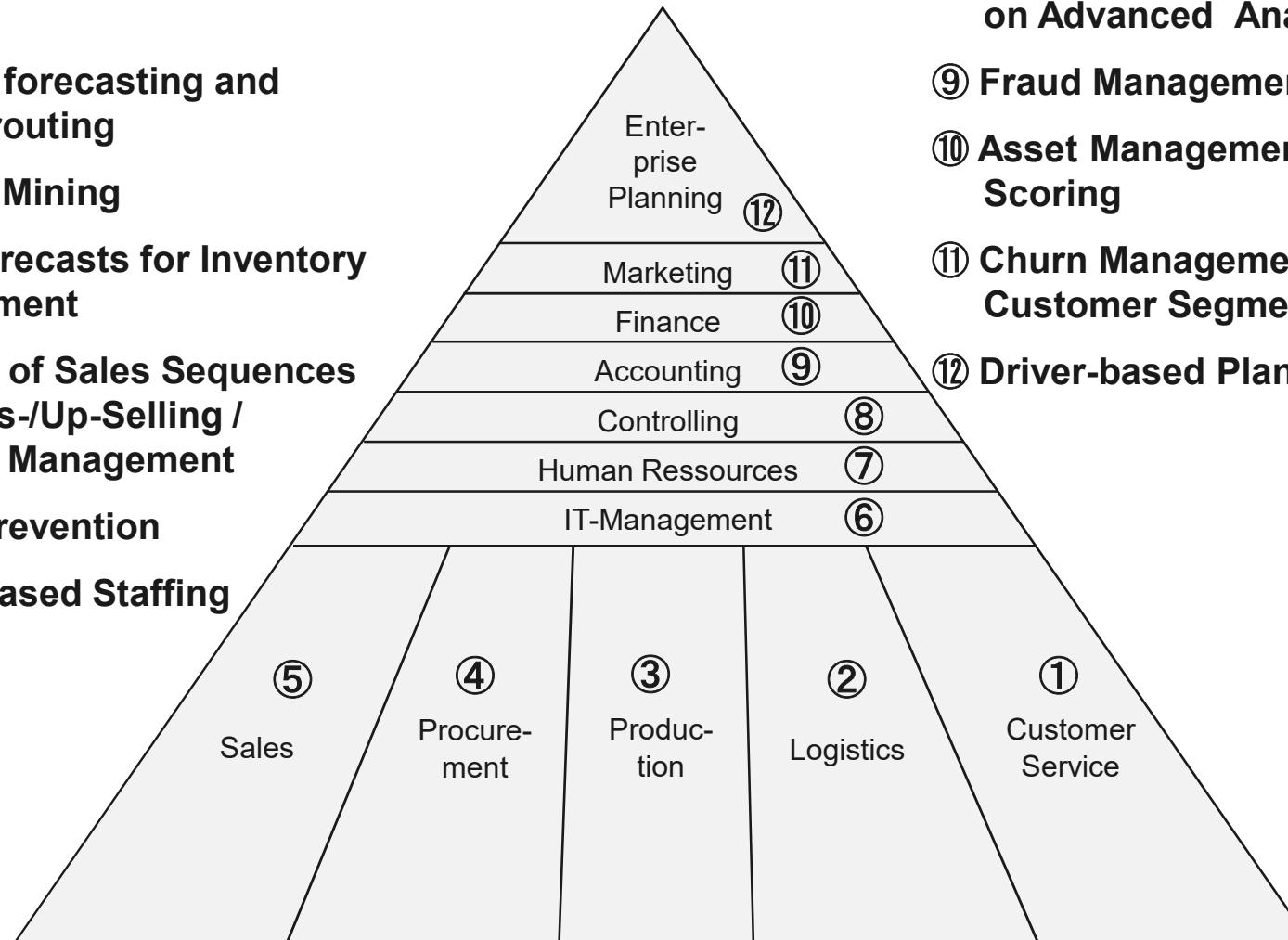
WHAT HAPPENED? (DESCRIPTIVE/BI)	WHAT WILL HAPPEN? (PREDICTIVE ANALYTICS)	WHAT SHOULD I DO? (PRESCRIPTIVE ANALYTICS)
How many widgets did I sell last month?	How many widgets will I sell next month?	Order [5,000] units of Component Z to support widget sales for next month
What were sales by zip code for Christmas last year?	What will be sales by zip code over this Christmas season?	Hire [Y] new sales reps by these zip codes to handle projected Christmas sales
How many of Product X were returned last month?	How many of Product X will be returned next month?	Set aside [\$125K] in financial reserve to cover Product X returns
What were company revenues and profits for the past quarter?	What are projected company revenues and profits for next quarter?	Sell the following product mix to achieve quarterly revenue and margin goals
How many employees did I hire last year?	How many employees will I need to hire next year?	Increase hiring pipeline by 35 percent to achieve hiring goals

Source: Schmarzo (2016): Big Data MBA, p. 13.

Use Cases of Data Science - Examples

- ① Customer Support / Text Mining
- ② Demand forecasting and vehicle routing
- ③ Process Mining
- ④ Sales Forecasts for Inventory Management
- ⑤ Analysis of Sales Sequences for Cross-/Up-Selling / Voucher Management
- ⑥ Attack Prevention
- ⑦ Profile-based Staffing

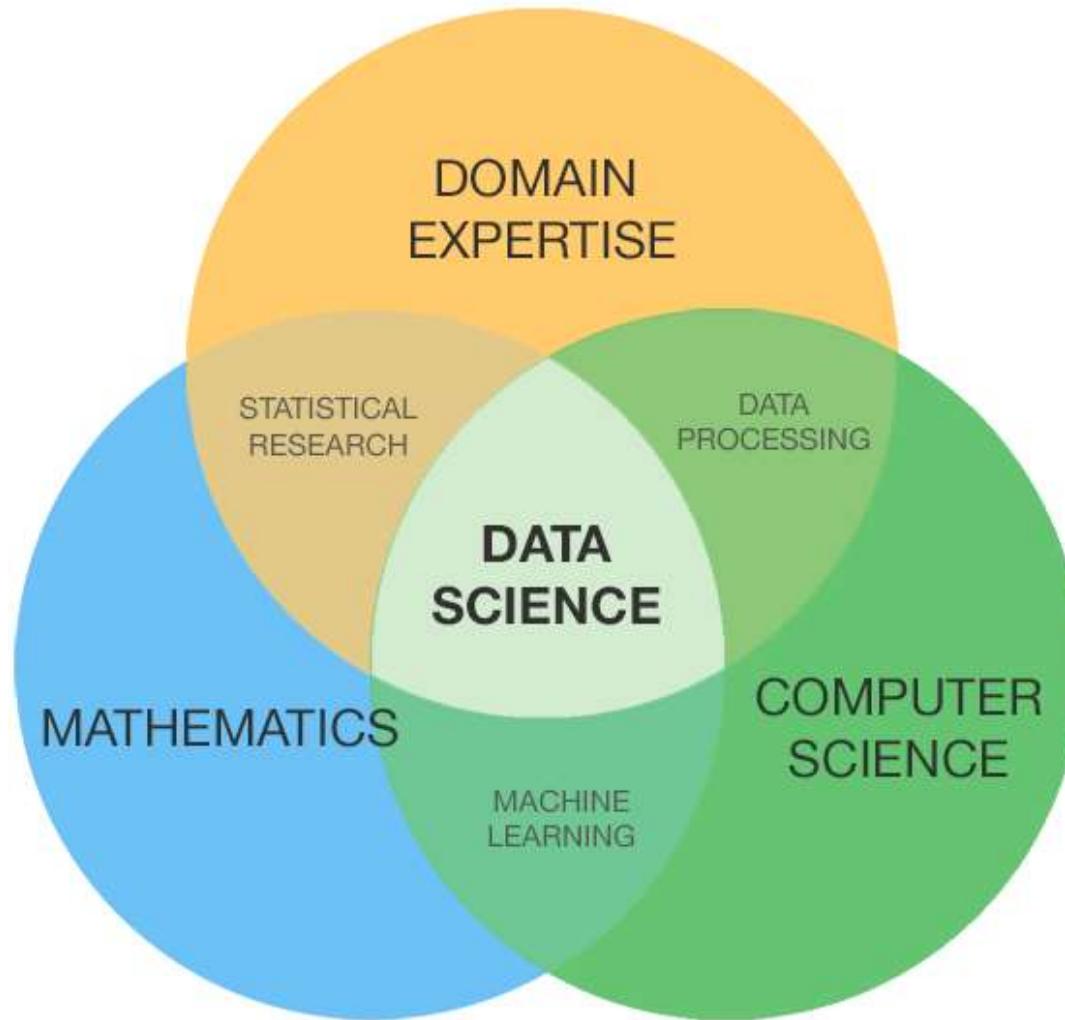
- ⑧ Realtime Controlling based on Advanced Analytics
- ⑨ Fraud Management
- ⑩ Asset Management / Credit Scoring
- ⑪ Churn Management / Customer Segmentation
- ⑫ Driver-based Planning



What is Data Science?

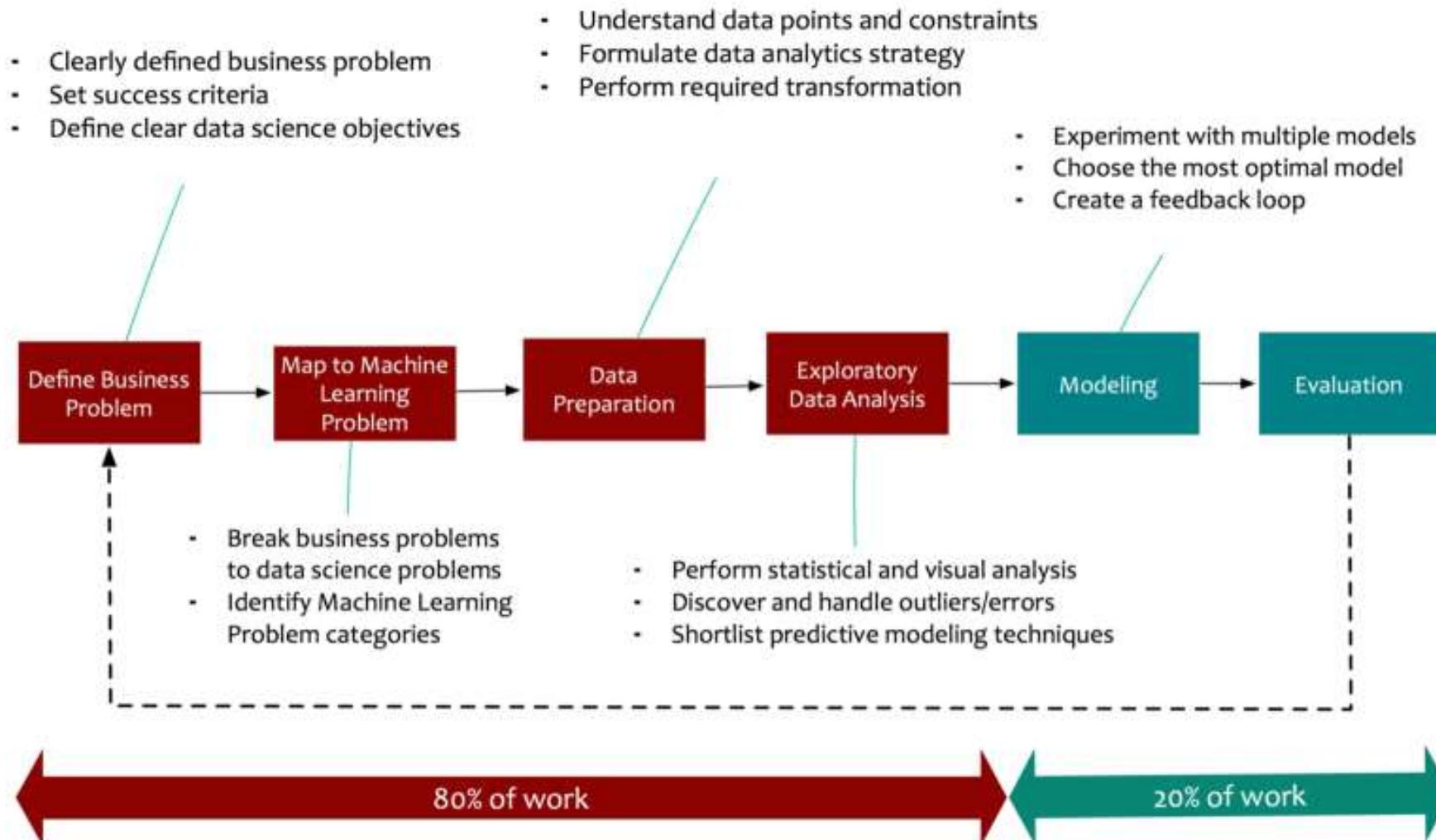
Data science is a collaborative discipline. Specifically, it is the combination of people, data, tools and processes that are used to transform statistical and mathematical skills, information technology, and industry subject matter expertise into actionable insights and business innovation.

Fundamental Skills of Data Science



Source: Palmer, Shelly. *Data Science for the C-Suite*.
New York: Digital Living Press, 2015. Print.

The Data Science Process



Source: <http://www.datasciencecentral.com/profiles/blogs/data-science-simplified-principles-and-process>

Need for Knowledge about the Algorithms

The distance between using Excel and VBA for modeling in credit scoring, for example, and using machine learning algorithms and R or Python to enhance the results, is not that great, compared to the distance between someone running a packaged algorithm they don't really understand and someone who understands the mathematical and statistical operations within an algorithm and can optimize or adapt it as needed – and do so in the context of their deep industry experience.

Source: Dataiku - Data Science for Banking and Insurance

Statistics and Machine Learning

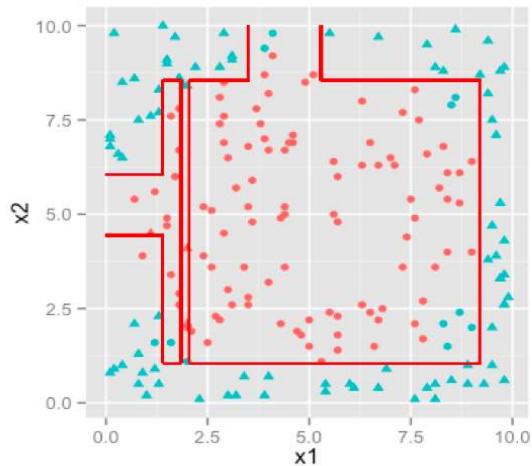
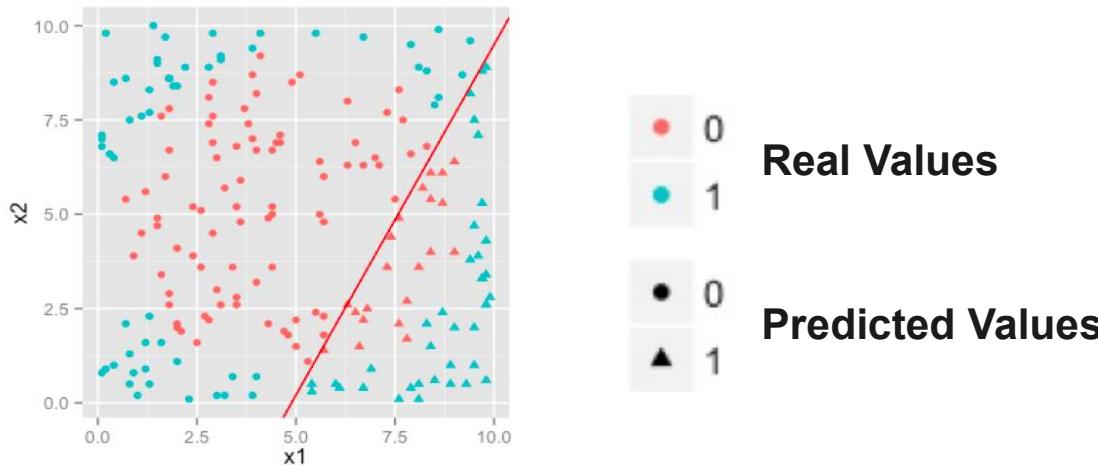
Statistics vs. Machine Learning

Statistics is about finding valid conclusions about the underlying applied theory, and on the interpretation of parameters in their models. It insists on proper and rigorous methodology, and is comfortable with making and noting assumptions. It cares about how the data was collected and the resulting properties of the estimator or experiment (e.g. p-value). The focus is on hypothesis testing.

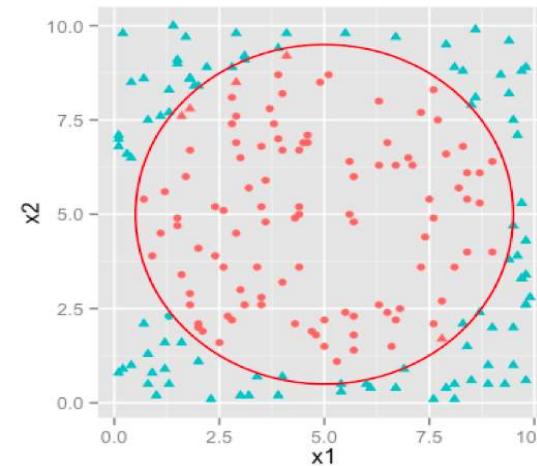
Machine Learning (ML) aims to derive practice-relevant findings from existing data and to apply the trained models to data not previously seen (prediction). It tries to predict or classify with the most accuracy. It cares deeply about scalability and uses the predictions to make decisions. Much of ML is motivated by problems that need to have answers. ML is happy to treat the algorithm as a black box as long as it works.

Statistical Regression vs. Machine Learning Algorithms

Traditional Regression



Decision Tree



Support Vector Machine

Explanation vs. Prediction (I)

Question 1: I have a headache. If I take an aspirin now, will it go away?

Question 2: I had a headache, but it passed. Was it because I took an aspirin two hours ago? Had I not taken such an aspirin, would I still have a headache?

The first case is a typical "predictive" question. You are calculating the effect of a hypothetical intervention.

The second case is a typical "explanatory" question. You are calculating the effect of a counterfactual intervention.

Explanation vs. Prediction (II)

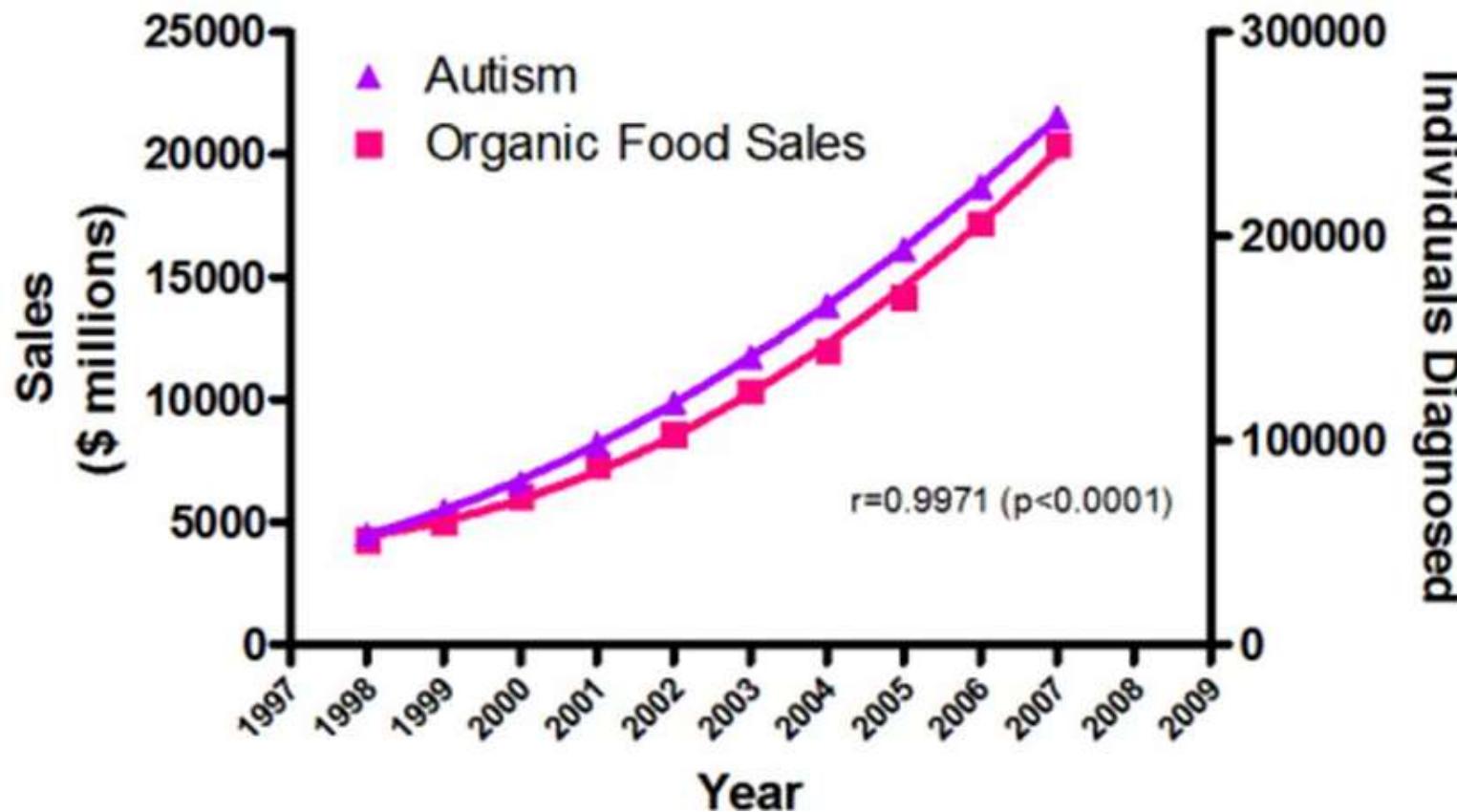
Explanation:

- Explanation is about understanding relationships and why certain things happen and other things not.
- What we need for this is an understanding of cause and effect.
- Tests of causal hypothesis are fundamental.
- Measures of significance are central.
- A good explanatory model may have predictive power.

Prediction:

- Prediction is about anticipating and forecasting what may happen in the future.
- Correlations are important in this context. Correlation does not imply causation.
- Therefore, predictive models may have no real explanatory power.
- To achieve robust predictive models the knowledge of causality is preferable.
- Main Task is to find a model that optimally approximates reality and minimizes overfitting.
- Accuracy is measured using out-of-sample data.

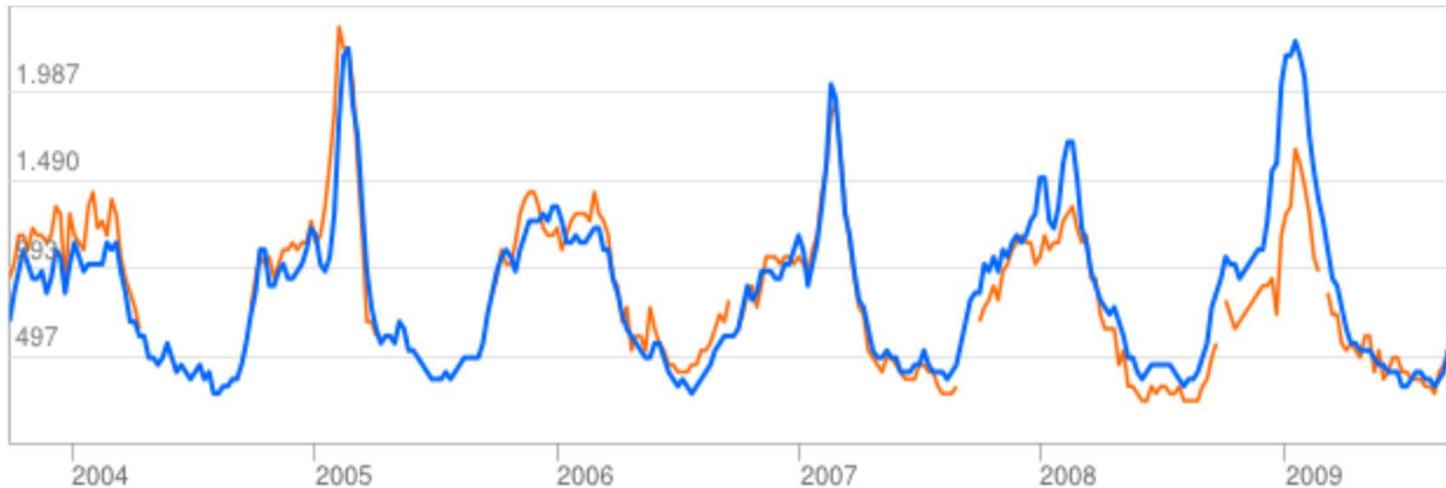
Correlation



Source: Organic Trade Association, 2011 Organic Industry Survey, U.S. Department of Education, Office of Special Education Programs, Data Analysis System (DANS)

Organic food sales and the rate of autism seem to have a very strong correlation, but no one is suggesting that one causes the other!

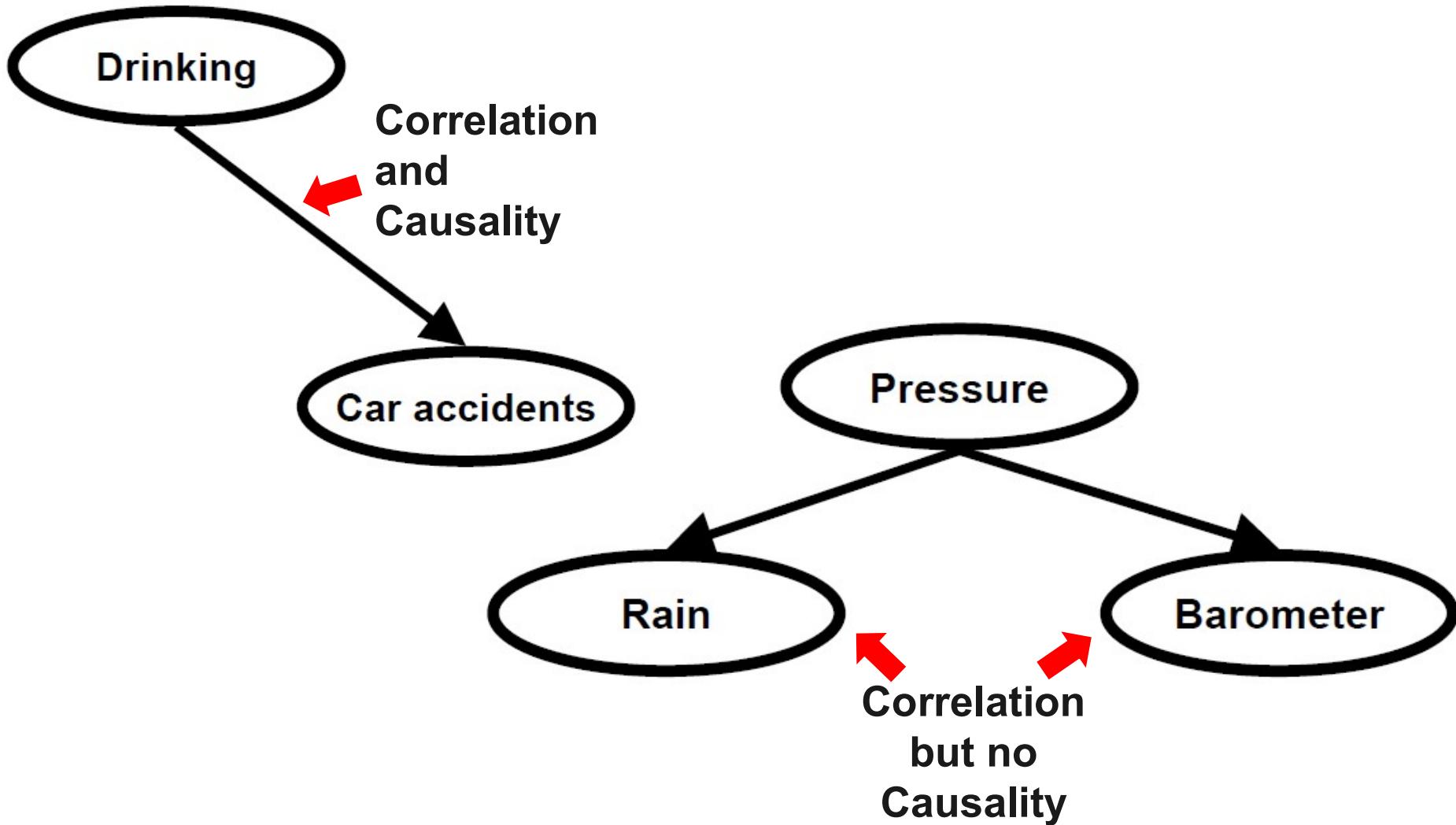
Correlation vs. Causality (I)



Correlation: Two data series behave "similar"

Causality: Principle of Cause and Effect

Correlation vs. Causality (II)



Correlation vs. Causality (III)

But:

**Sometimes it is better to
know/predict something
even if we can not explain it
instead of doing nothing!**

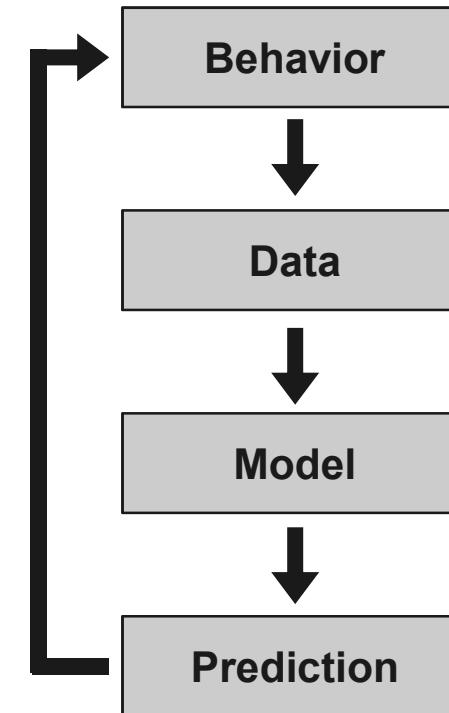
Impact of Predictions

If you predict the weather, independent of the behavior of the people, your model does not influence the outcome .

But if you build a recommendation system, your model gets incorporated back into the real world. Users interact with that product and are affected.

This behavior generates more data, which creates a feedback loop.

Your model is not just predicting the future, but causing it!



Descriptive and Inferential Statistics

Descriptive statistics provides a concise summary of data, e.g. mean, variance, and visualizations. Descriptive statistics does not allow to make conclusions beyond the data.

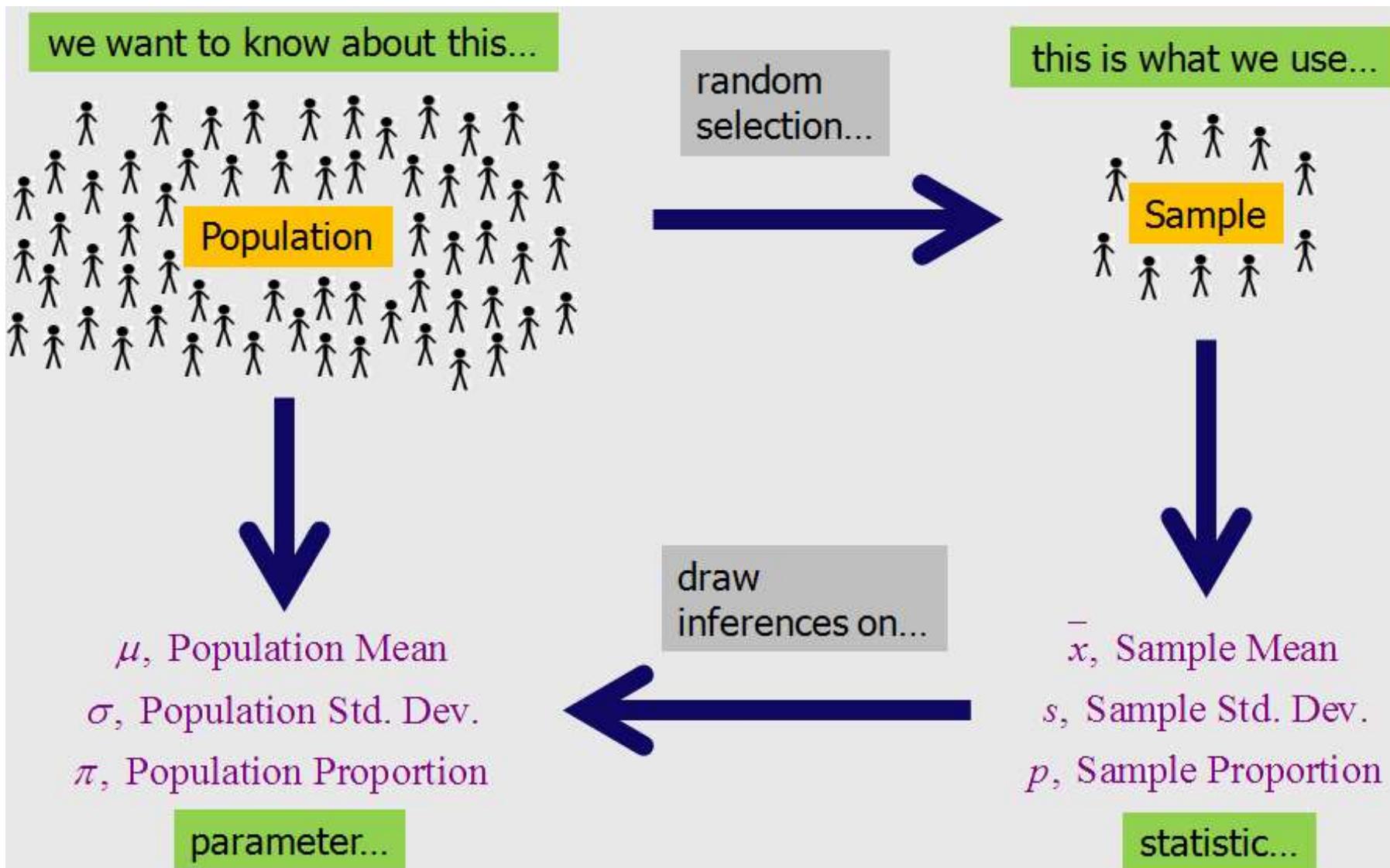
Inferential statistics uses a random sample of data taken from a population to describe and make inferences about the population.

These inferences may take the form of

- estimates of numerical characteristics (estimation),
- answers to yes/no questions (hypothesis testing),
- forecasting of future observations (forecasting),
- descriptions of association (correlation), or
- modeling of relationships (regression).

Source: Bichler (2015): Course Business Analytics, TU München

Statistical Estimation



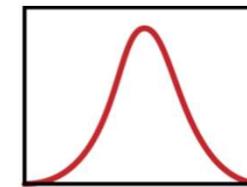
Probability distributions

In statistics we often model the relationship between a population and a sample with an underlying mathematical process.

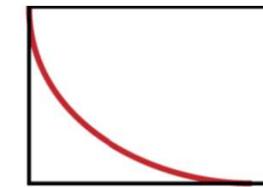
Probability distributions are the foundation of statistical models. In the pre-computer age, scientists observed real-world phenomena, took measurements, and noticed that certain mathematical shapes kept reappearing.

Natural processes tend to generate measurements whose empirical shape could be approximated by mathematical functions with a few parameters that could be estimated from the data. We can use these functions as building blocks of our models.

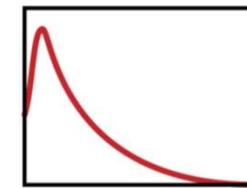
So we make simplifying assumptions about the underlying truth, the mathematical structure, and shape of the underlying generative process that created the data.



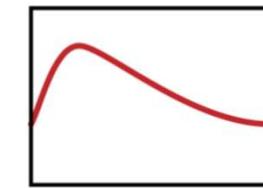
Normal Distribution



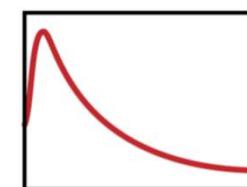
Exponential Distribution



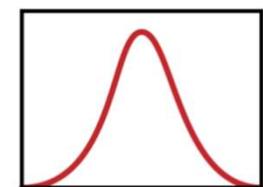
F Distribution



Chi-Square Distribution



Lognormal Distribution



t Distribution

Source: Schutt/O'Neil (2013): Doing Data Science, pp. 30.

Definitions

A **method** is a composition of formalized principles that form the basis for a stringent calculation process.

An **algorithm** is a procedure or set of steps or rules to accomplish a task. It is usually the implementation of a method. Algorithms are used to build models.

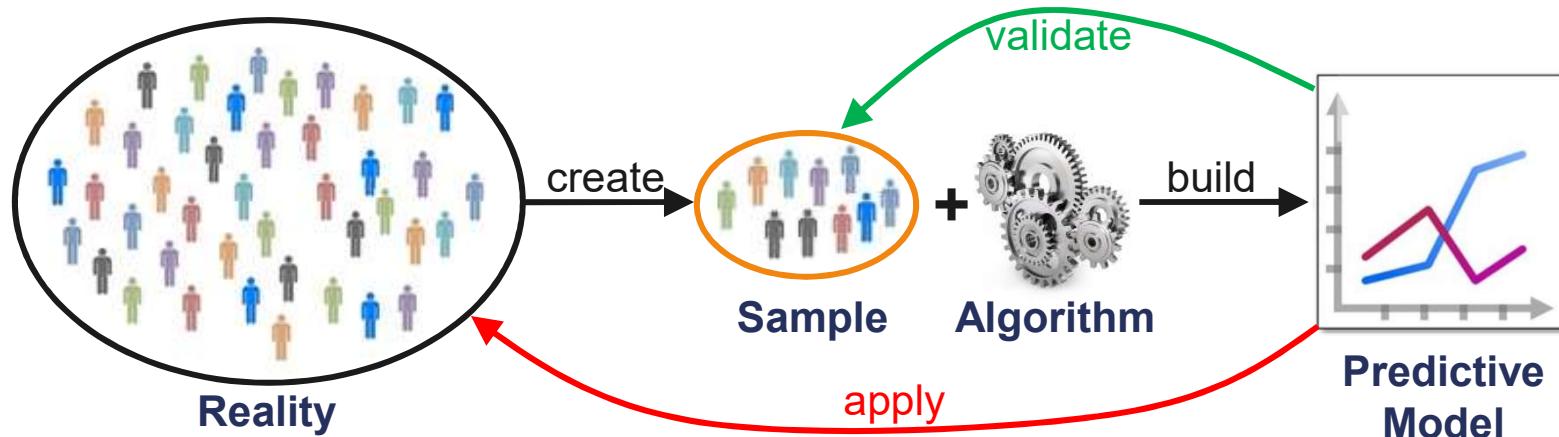
In the given context, a **model** is the description of the relationship between variables. It is used to create output data from given input data, for example to make predictions.

Fitting a model means that you estimate the model using the observed data. You are using your data as evidence to help approximate the real-world mathematical process that generated the data. Fitting the model often involves optimization methods and algorithms, such as maximum likelihood estimation, to help get the parameters.

Overfitting is the term used to mean that you used a dataset to estimate the parameters of your model, but your model isn't that good at capturing reality beyond your sampled data.

Source: Schutt/O'Neil (2013): Doing Data Science.

Traditional Analytics Process



Example Regression - Fitting the model

Age	Rooms	Dist_School	Dist_Mall	Lakeside	Sales_Price
42	6	4.4	1.1	0	392000
35	6	4.7	12.4	0	308000
32	7	1.5	1.5	0	429000
39	5	2.8	10.2	1	338000
18	6	0.1	0.3	0	450000
43	6	2.5	13.5	0	304000
34	5	2.4	3.3	0	349000
49	6	1.4	10.2	0	326000
10	6	0.7	8.8	0	369000
8	6	2.2	1.1	0	413000
22	5	4.9	11.7	0	306000
3	7	4.7	11.5	0	378000
25	4	2.5	3.8	0	348000
38	4	1.4	0.7	0	367000
30	5	2.8	11.6	0	299000
26	6	2.2	8.1	0	350000
38	7	4.2	6.6	0	391000
2	6	4.6	10.4	0	343000
26	5	4.9	1	0	353000
31	5	2.5	5.6	0	365000
44	5	4.9	4.9	1	380000
25	7	1	14.1	0	353000
8	5	0.2	13.3	0	331000
29	4	1.4	2.7	0	351000
19	4	0.8	9.7	0	304000
26	7	2.2	3.2	1	466000
43	4	2.8	0.6	1	379000
24	4	2.6	12.5	1	301000
45	5	1.2	1.6	0	390000
28	6	4.4	3.2	0	383000
35	4	2	0.1	0	371000
48	6	2.4	11.3	0	301000
0	6	2.7	10.7	1	386000
18	4	2.4	0.7	0	387000
30	7	3.8	9.5	1	414000
42	6	0.9	1.4	0	397000
1	4	3	1.9	0	387000
26	5	0.1	1.1	1	391000

SUMMARY OUTPUT					
Regression Statistics					
Multiple R	0.965794107				
R Square	0.932758256				
Adjusted R Square	0.924753287				
Standard Error	11962.63096				
Observations	48				
ANOVA					
	df	SS	MS	F	Significance F
Regression	5	83374421845	16674884369	116.5223999	1.71591E-23
Residual	42	6010390655	143104539.4		
Total	47	89384812500			
	Coefficients	Standard Error	t Stat	P-value	
Intercept	309694.3587	10414.1276	29.73790707	7.64477E-30	
Age	-978.7070888	118.4386216	-8.263411678	2.40669E-10	
Rooms	26247.68844	1788.306679	14.67739776	3.88184E-18	
Dist_School	-6684.949298	1220.872194	-5.475552096	2.24688E-06	
Dist_Mall	-8002.138709	400.8778687	-19.96153775	4.72709E-23	
Lakeside	38025.024	4212.289608	9.027162788	2.17287E-11	

Proportion of variance explained by the model.

Probability that the null hypothesis that *all* of the regression coefficients are equal to zero.

Probability that the null hypothesis $\beta = 0$ is true.
The coefficients are highly likely to be nonzero and therefore significant.

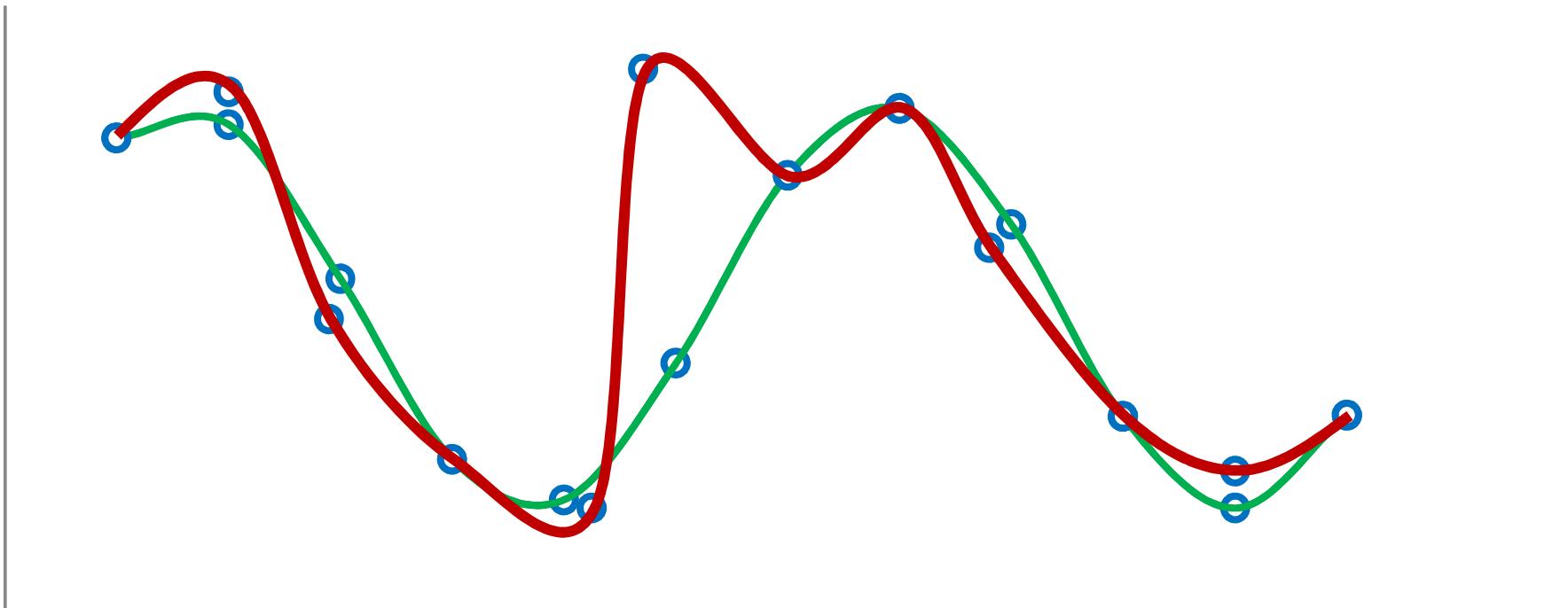
Example Regression - Testing the model

Constant	Age	Rooms	Dist_School	Dist_Mall	Lakeside	Sales_Price_Estimation	Deviation	Dev_Mean
1	16	5	3.3	10.9	0	299000	315989.843	288654761.06
1	31	4	0.6	12.8	0	270000	277906.848	62518240.02
1	20	5	1	1.1	0	420000	405871.357	199618545.66
1	13	5	2	3.8	0	377000	384431.583	55228426.91
1	31	5	3.7	7.1	0	323000	329043.384	36522489.25
1	23	6	3	10	0	329000	344593.991	243172565.38
1	36	7	3.5	6.1	0	340000	385984.354	2114560805.77
1	33	6	5	7.5	1	394000	379467.393	211196677.41
1	47	6	0	0.3	0	429000	418780.615	104435838.65
1	5	5	2.3	1.7	0	390000	407060.246	291052003.12
1	8	4	2.5	12.4	0	299000	290916.563	65341961.39
1	17	4	1.1	0.5	0	380000	386692.578	44790605.45
1	39	5	4.5	6.8	0	295000	318266.409	541325805.81
1	17	7	3	14.6	0	260000	339904.084	6384662676.66
1	6	7	4.8	6.7	1	447000	439878.873	50710445.81
1	2	4	4.3	10	0	289000	303961.029	223832395.48
1	13	4	3.3	12.9	0	289000	276673.998	151930318.16
1	6	4	0.4	10.7	1	371000	358541.03	155225932.53
1	17	5	2.1	7.4	0	336000	351040.56	226218458.07
1	11	5	4.7	5	1	391000	396762.092	33201700.68
1	37	5	2.3	5	1	408000	387359.586	426026702.49
1	19	7	1.3	9.9	1	417000	424946.16	63141455.48
1	41	6	4.9	5.6	1	361000	387510.294	702795708.22
1	18	7	4.1	7.6	0	394000	387586.904	41127801.86
1	4	7	3.2	14.4	0	362000	352890.714	82979086.57
1	48	7	2.8	4.6	1	421000	428947.565	63163796.31
1	47	7	4.3	8.2	0	349000	353066.125	16533374.30
1	25	4	2.8	13.4	0	295000	264270.919	944276448.70
1	47	5	2.9	9	1	357000	341552.99	238610105.71
1	1	6	1.4	11	1	390000	406844.351	283732175.62
1	49	7	3.5	10.1	0	350000	341252.607	76516885.48
							Sum: 14423104194.01	80792496093.75
							Pseudo-R²:	82.1480%

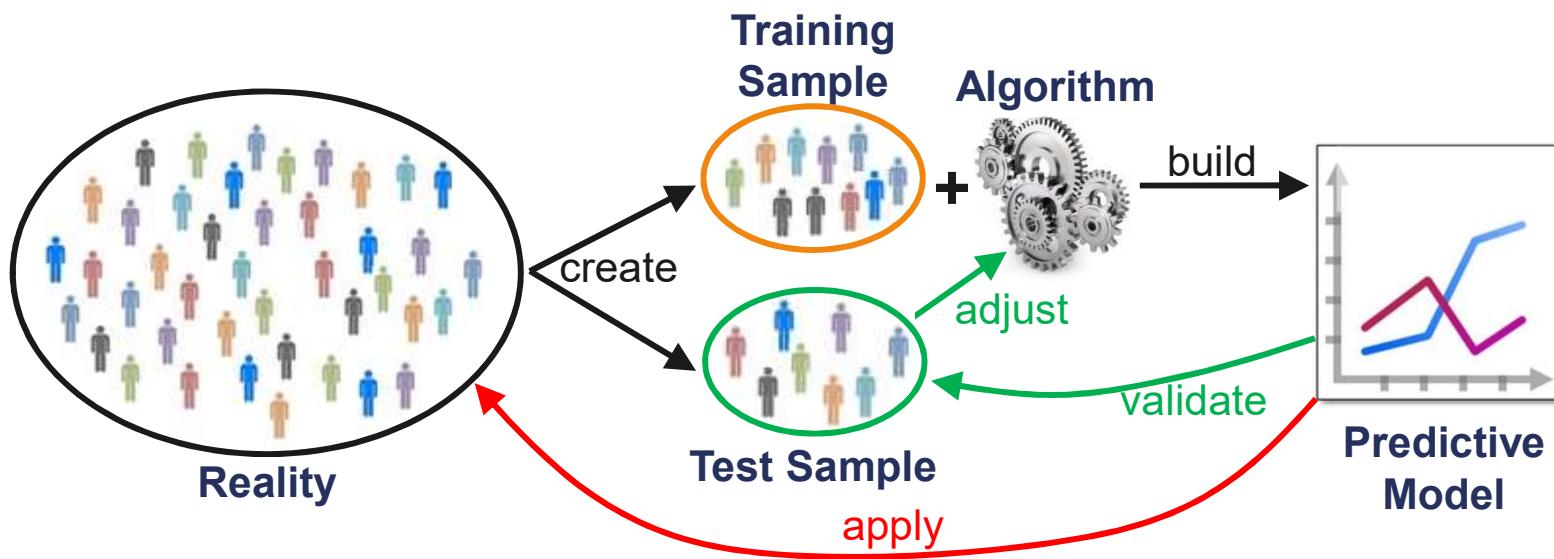
Conclusion:
Model is not robust due to overfitting.



Data Errors and their Consequences

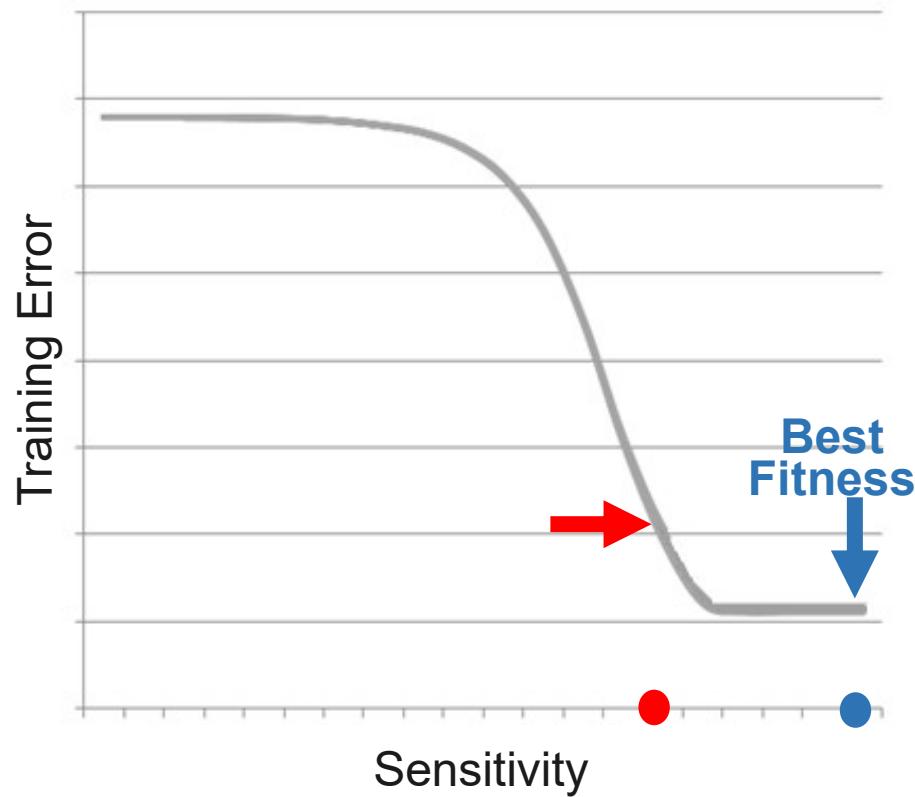


Modern Analytics Process

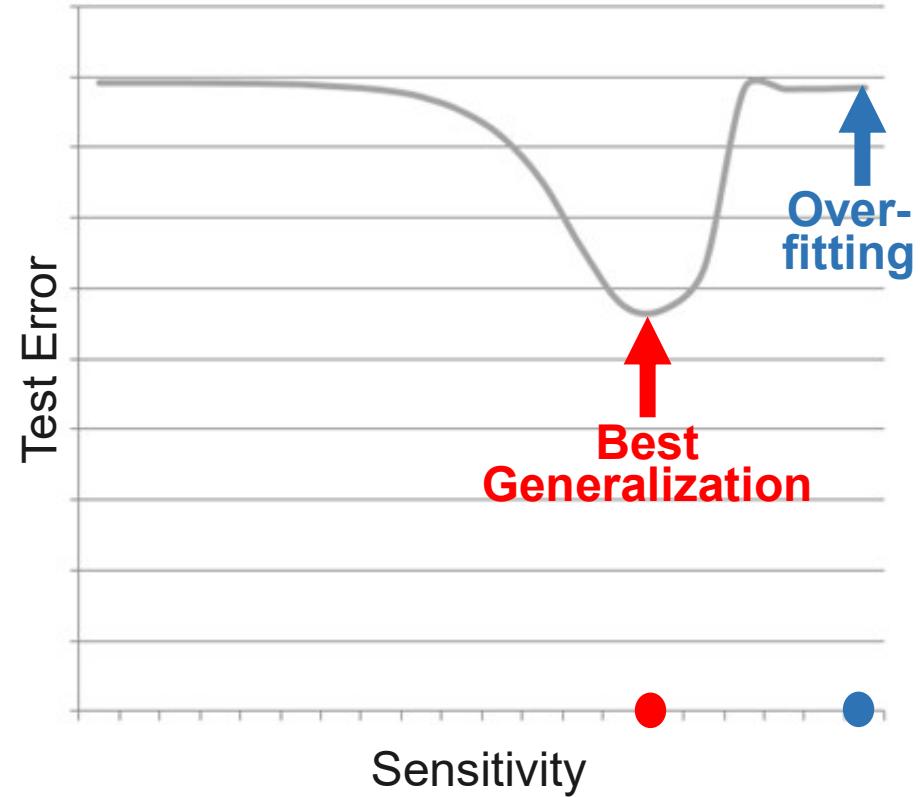


Best Fit vs. Best Generalization

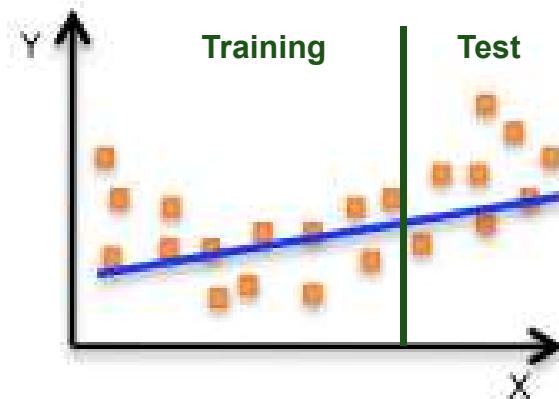
Training (Model Estimation)



Test (Model Application)

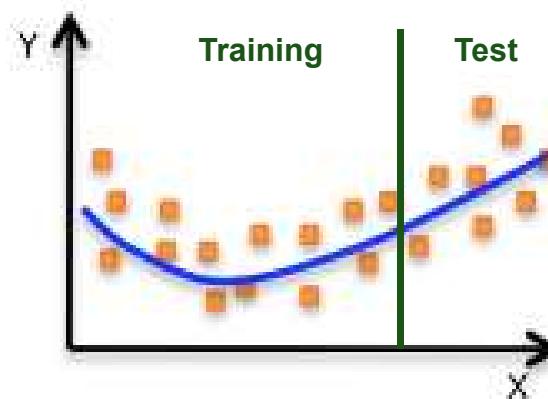


Over- and Underfitting



Underfitting

(model is too simple,
both training and test
errors are large)



Good Fitting



Overfitting

(model is too complex,
training error is small,
but test error is large)

Due to the problem of overfitting, the main goal is to maximize the prediction quality and not to fit the data that is used for the model estimation as well as possible. This is equivalent to minimizing the risk that the model will have weak predictive ability.

The Bias-Variance Tradeoff

The prediction error is influenced by three components:

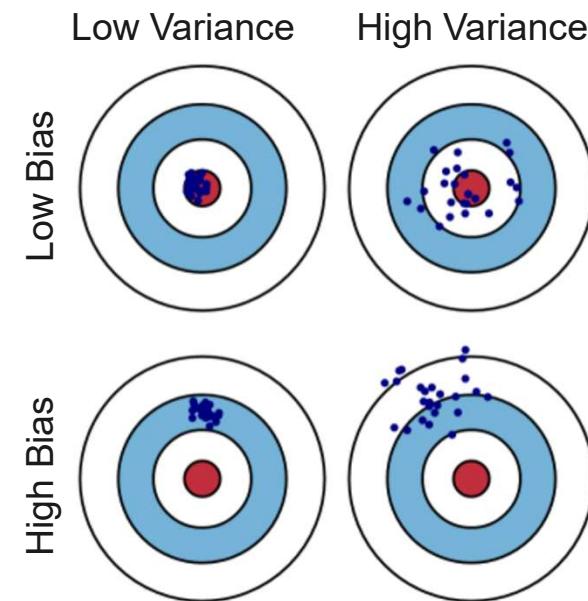
$$\text{Error} = \text{Bias} + \text{Variance} + \text{Noise}$$

Bias is the inability of the used method to learn the relevant relations between the inputs and the outputs. It reflects the method quality, e.g. if a method only produces linear models.

Variance is represents the deviation resulting from the sensitivity of the created model to small fluctuations in the data.

Typically, there is a tradeoff between bias and variance.

Noise is everything that arises from random variations in the data. It cannot be controlled.



The P-value

P-values are used in Null Hypothesis Significance Testing to decide whether to accept or reject a null hypothesis (which typically states that there is no underlying relationship between two variables). If the null hypothesis is rejected, this gives grounds for accepting the alternative hypothesis (that a relationship does exist between two variables).

The P-value quantifies the probability of observing results at least as extreme as the ones observed given that the null hypothesis is true. It is then compared against a pre-determined significance level (α). If the reported P-value is smaller than the result, it is considered statistically significant. Typically, in the social sciences α is set at 0.05.

Source: Vidgen/Yasseri (2016): P-Values: Misunderstood and Misused.

The Problem with the P-value

When Ronald Fisher introduced the P-value in the 1920s, he did not mean it to be a definitive test. His idea was to run an experiment, then see if the results were consistent with what random chance might produce.

Assuming that the null hypothesis was in fact true, he calculates the chances of getting results at least as extreme as what was actually observed. This probability was the P-value. The smaller it was, suggested Fisher, the greater the likelihood that the null hypothesis was false.

Hence, all the P-value can do is summarize the data assuming a specific null hypothesis. The P-value indicates whether a measured result can also be explained by chance. But, it does not answer the really interesting question: What does it say about the correctness of the hypothesis? It cannot work backwards and make statements about the underlying reality.

Source: Nuzzo (2014): Statistical Errors, in: Nature, Vol. 506, pp. 150-152.

Summarizing: Statistics vs. Data Analytics

Statistics

Goals: Explanation, Forecasting

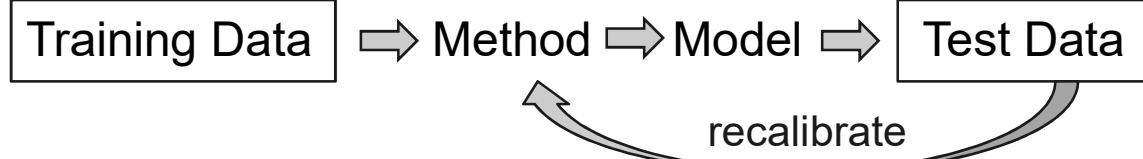
Methods: Statistical Methods

Process:  Data → Method → Model → generalizable?
(Statist. Tests)

Data Analytics

Goals: Forecasting, Finding Structures

Methods: Statistical Methods, Machine Learning, Artificial Intelligence

Process:  Training Data → Method → Model → Test Data
recalibrate

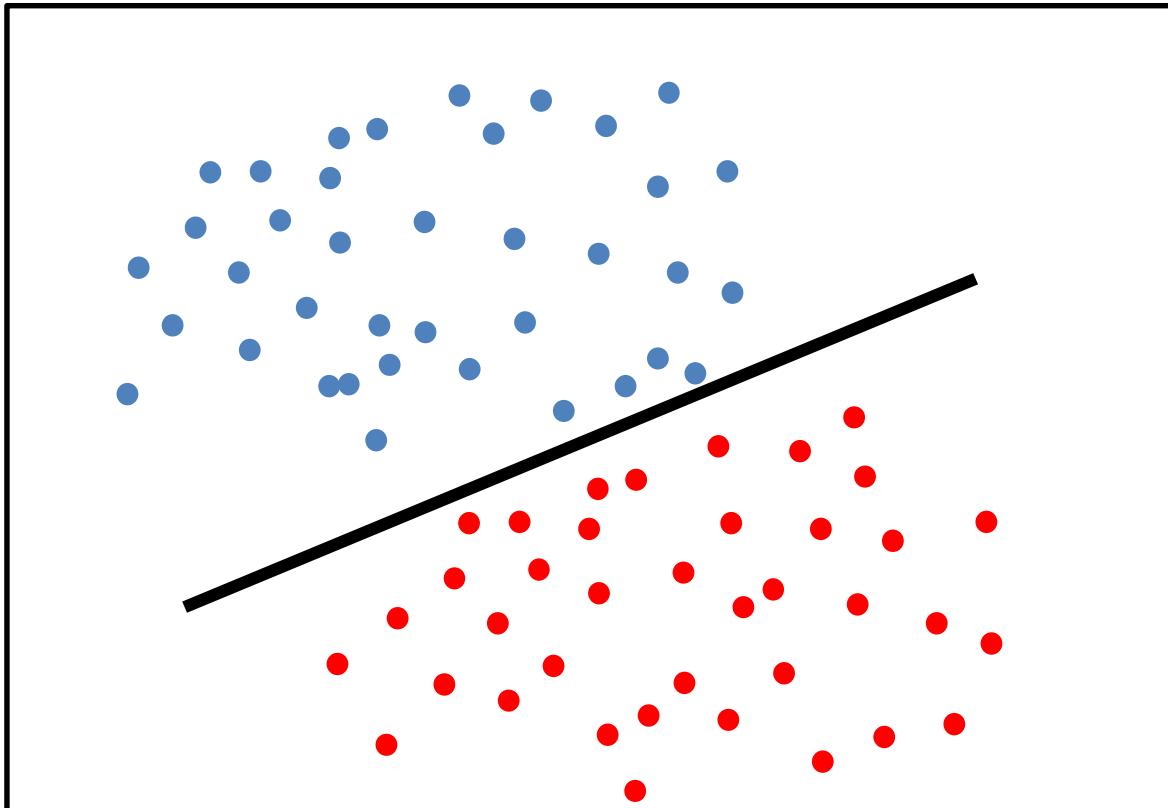
Which Method should I choose?

The choice of the method of data analysis depends on the one hand on the scope of application, but on the other hand on the interrelationships of the data to be analyzed.

In the Big Data area, data spaces are often highly-dimensional, making it difficult to visualize the interrelationships.

For this reason, the choice of the method can often not be made ex ante. In these cases, different methods are competitively tried to select the most suitable one.

Linear World

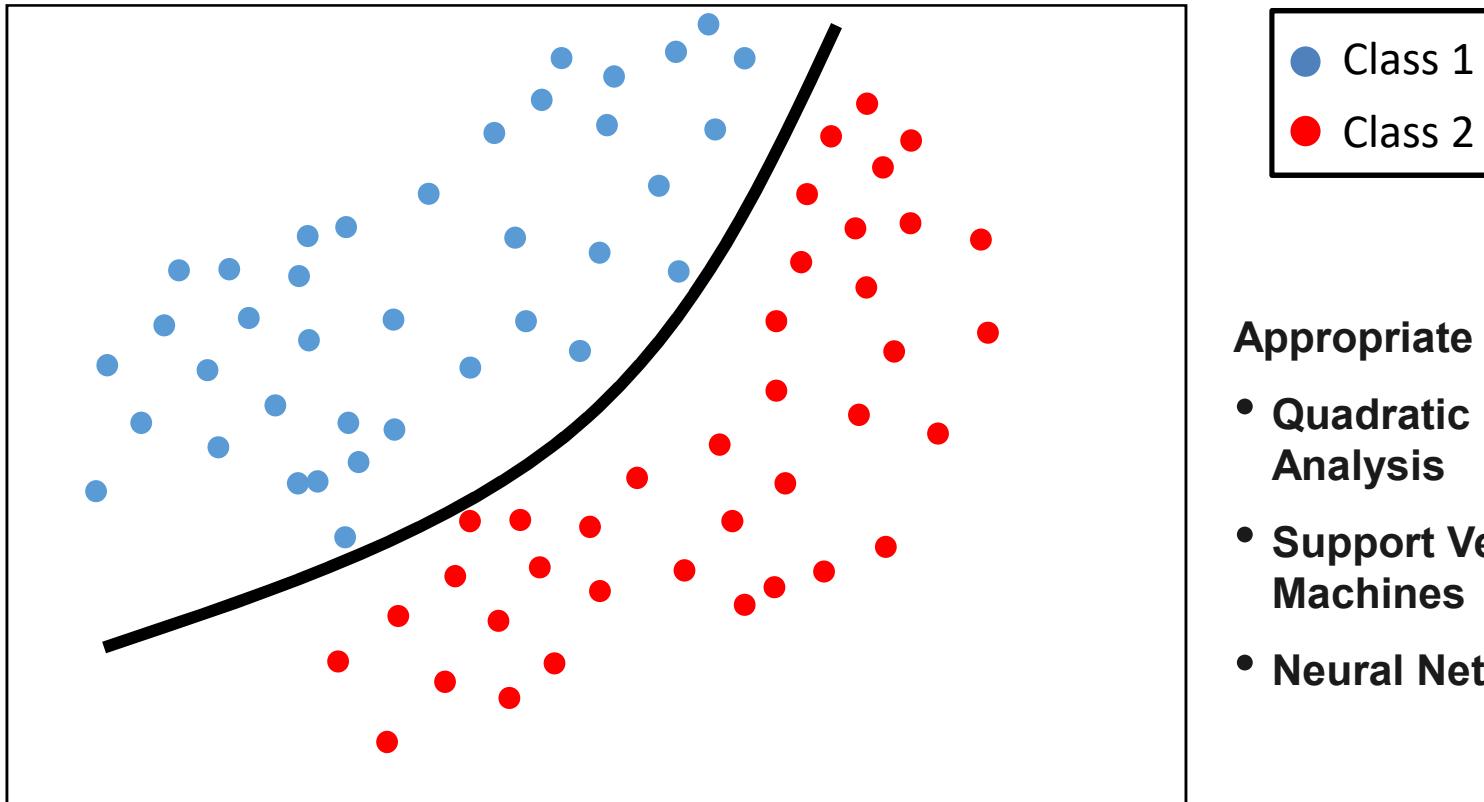


- Class 1 e.g. creditworthy
- Class 2 e.g. not creditworthy

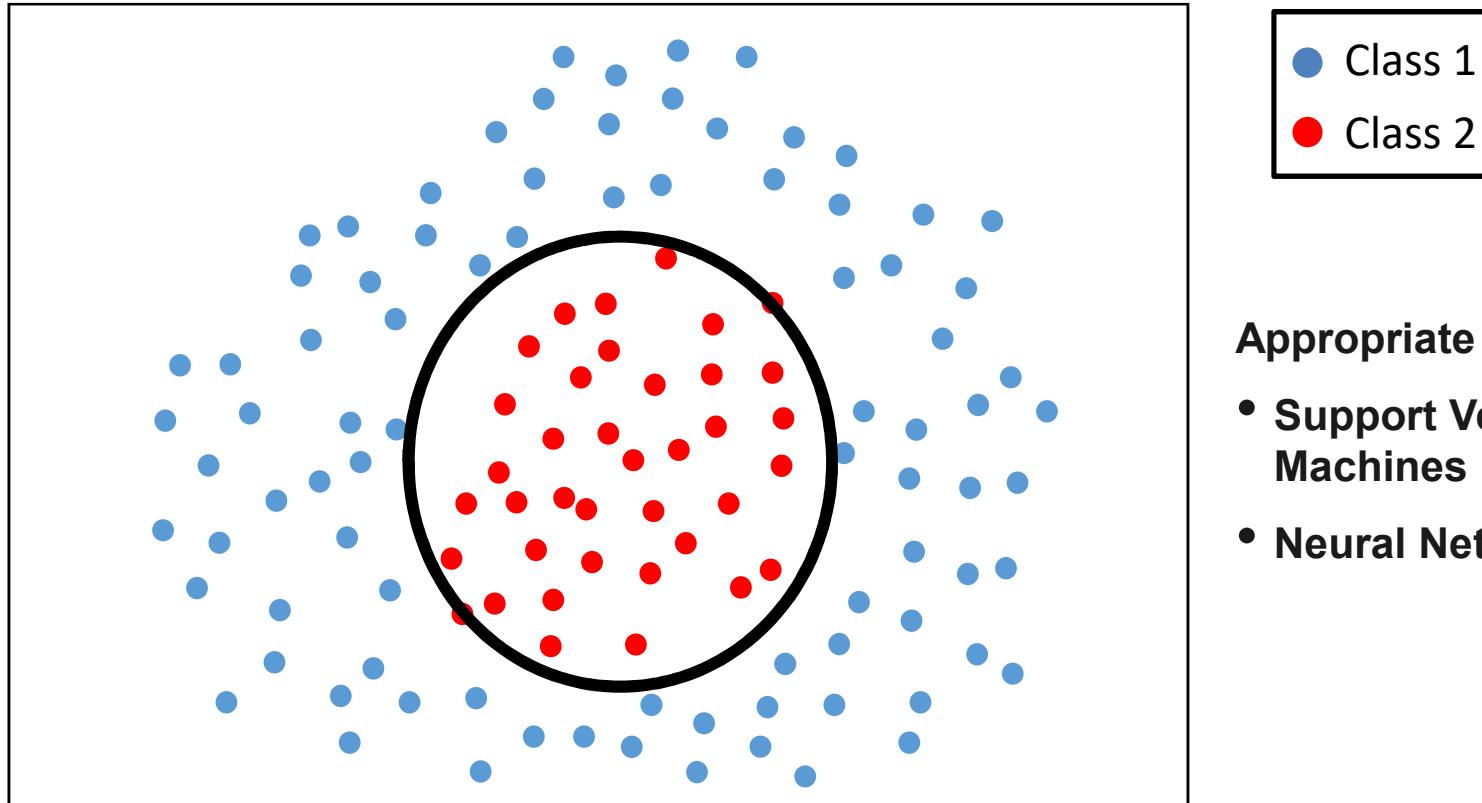
Appropriate Methods:

- Regression
- Linear Discriminant Analysis

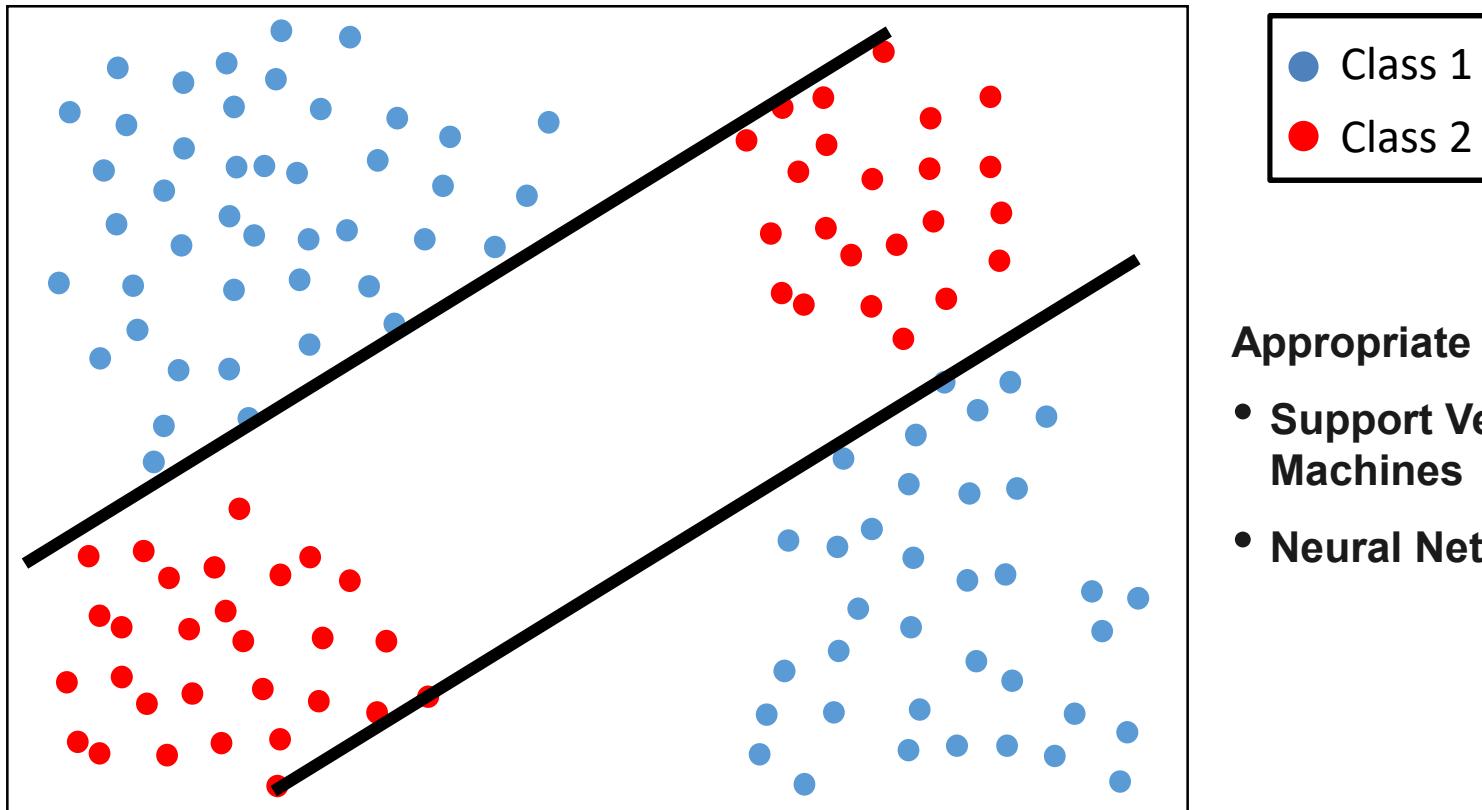
Quadratic World



Nonlinear World (Type 1)



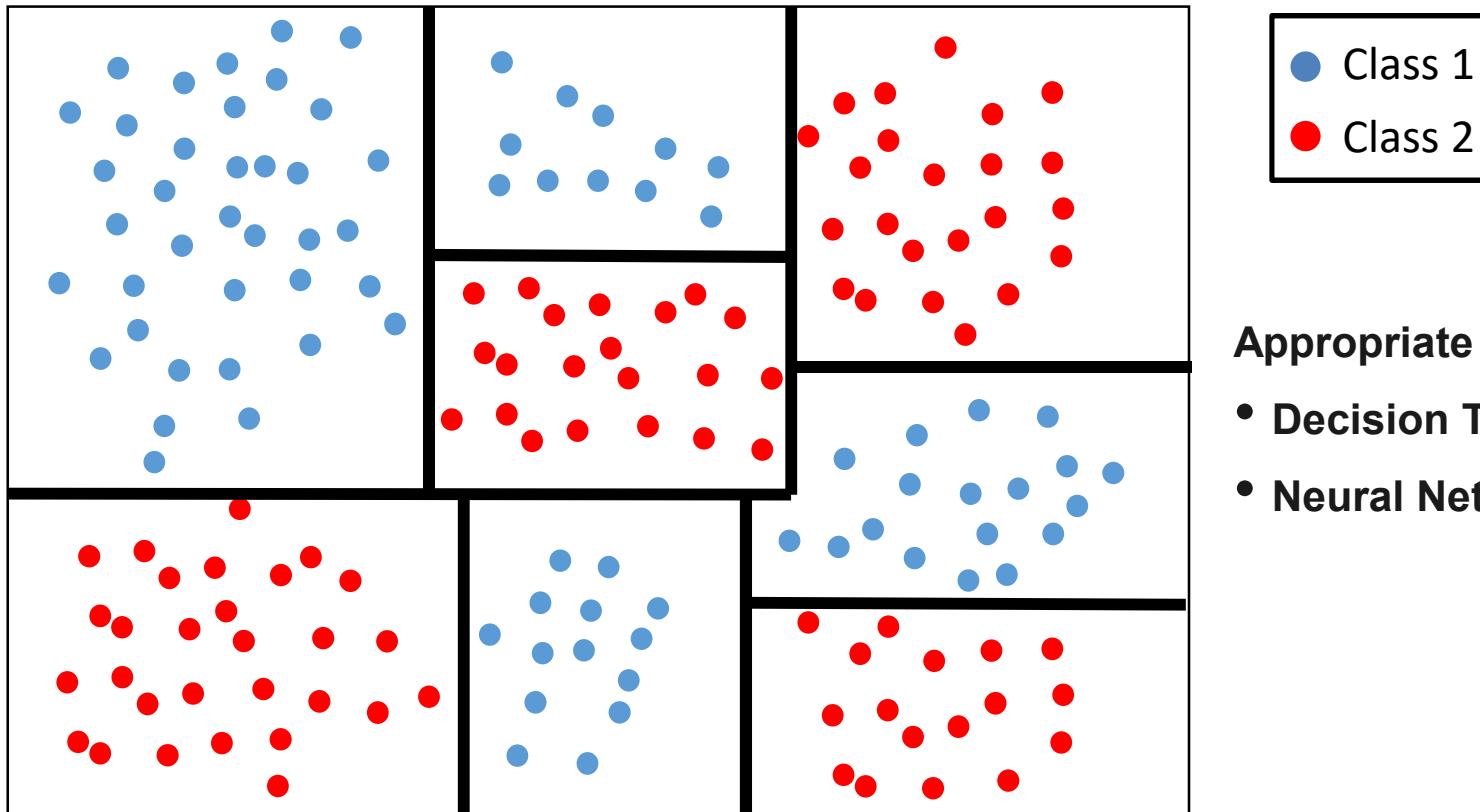
Nonlinear World (Type 2)



Appropriate Methods:

- Support Vector Machines
- Neural Networks

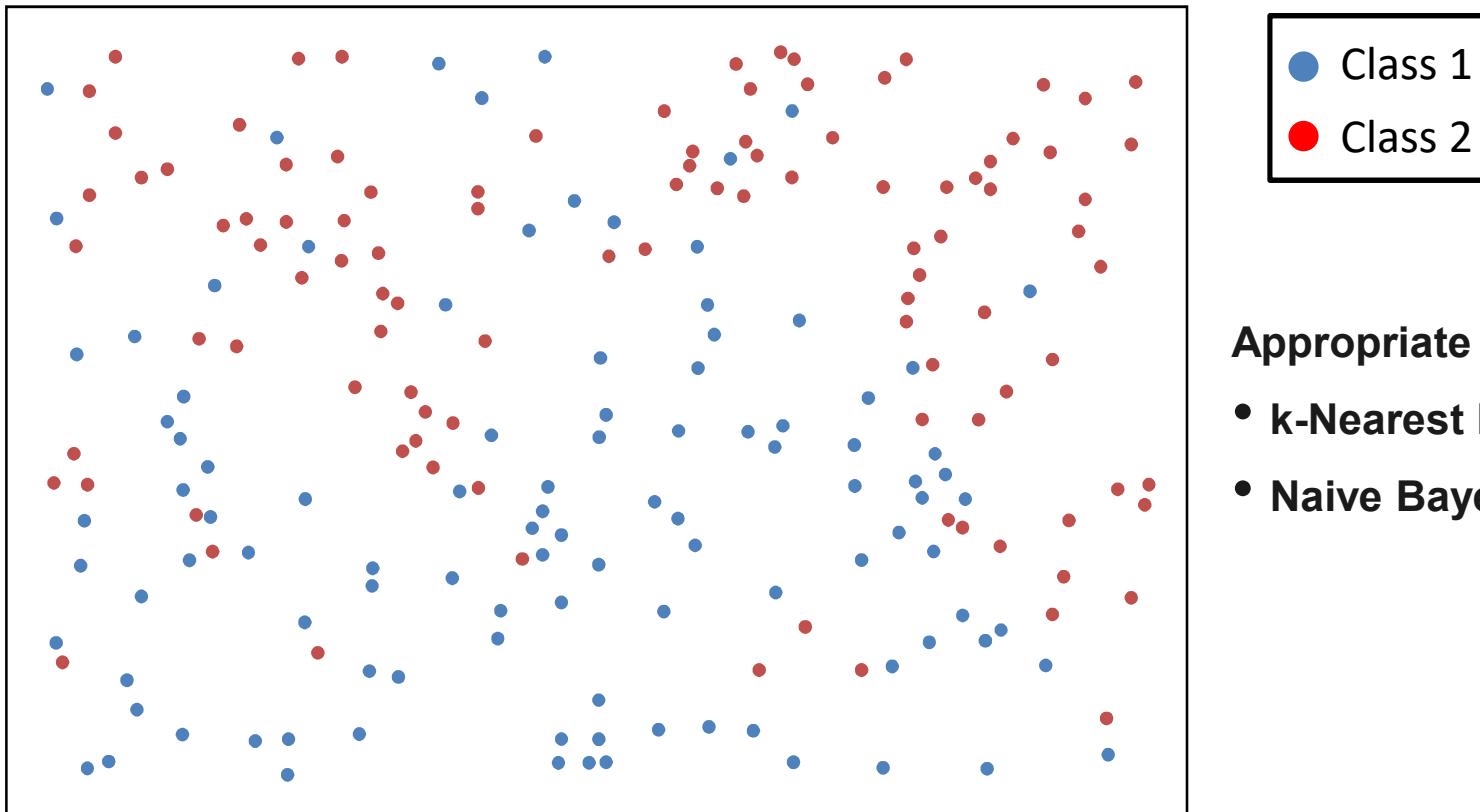
Nonlinear World (Type 3)



Appropriate Methods:

- Decision Trees
- Neural Networks

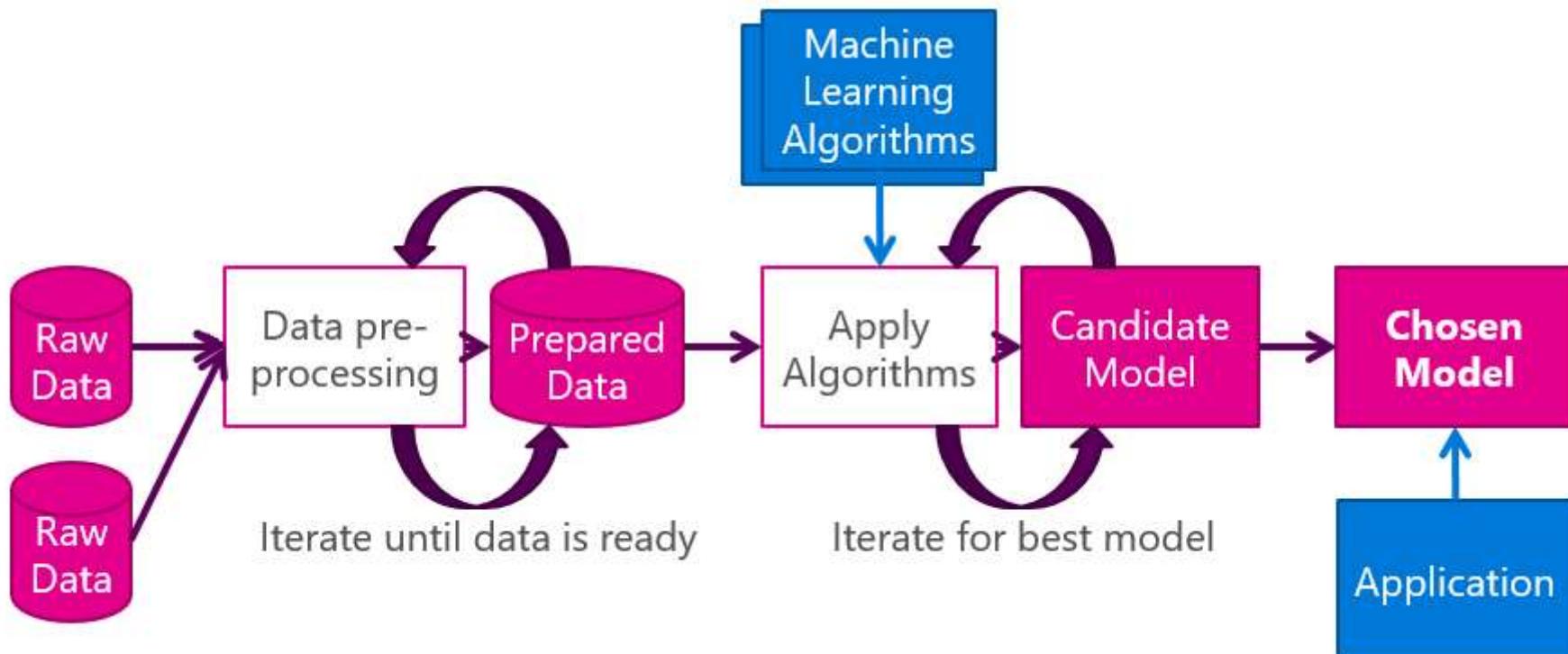
Nonlinear World (Type 4)



Appropriate Methods:

- **k-Nearest Neighbors**
- **Naive Bayes**

The Data Science Process - Technical View

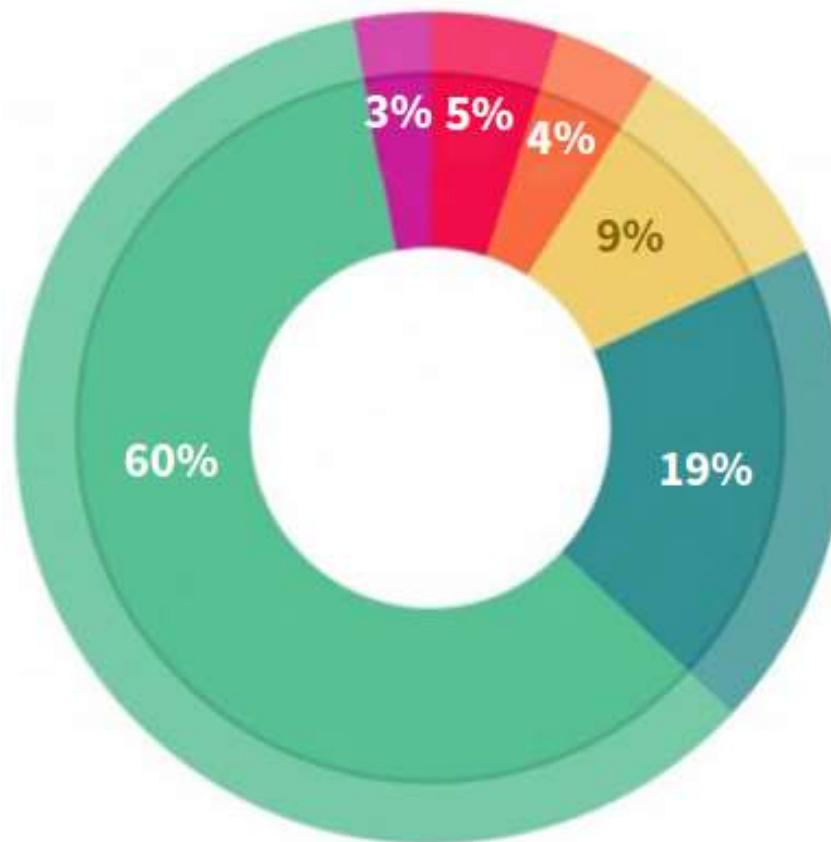


Source: <http://blogs.msdn.microsoft.com/martinkearn/2016/03/01/machine-learning-is-for-muggles-too/>

Data Preparation

Data Preparation and Enrichment

The data collection and preparation phase is the most labor-intensive one, consuming on average between 60-80% of a data scientist's time. It's critical therefore to select a tool that can automate or at least speed the workflows associated with data preparation.



What data scientists spend the most time doing

- Cleaning and organizing data: 60%
- Collecting data sets: 19%
- Mining data for patterns: 9%
- Refining algorithms: 4%
- Other: 5%
- Building training sets: 3%

Source: 2016 Dataiku, Inc.

Data Cleaning

1. Proof of correctness of the data

- examine for irregular outliers (e.g. Age=236)
- examine for typographical errors (e.g. Frankfrut)
- examine for different writing styles (e.g. behavior/behaviour)
- ...

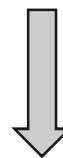
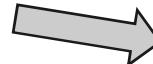
2. Handling missing values

- eliminate all datasets with missing values?
- replace them with their mean values?
- replace them with estimation values (e.g. through interpolation, causal analysis, or k-Nearest Neighbors analysis)?
- treat them as an another variable value in case of categorical data?

Missing Values Strategies

Initial Situation

Variable 1	Variable 2	Variable 3	Variable 4	Variable 5
8.95	0.99	NA	370.53	-34.97
NA	NA	168.02	617.46	-99.49
18.38	0.48	58.06	1177.66	-44.21
16.10	NA	35.39	NA	77.35
23.14	NA	230.44	408.97	-71.79
24.77	0.92	206.83	NA	46.13
20.72	NA	228.74	659.37	-67.77
15.52	0.04	22.46	NA	36.63



Smart Strategy

Variable 1	Variable 2	Variable 3	Variable 4	Variable 5
8.95	0.99	NA	370.53	-34.97
NA	NA	168.02	617.46	-99.49
18.38	0.48	58.06	1177.66	-44.21
16.10	NA	35.39	NA	77.35
23.14	NA	230.44	408.97	-71.79
24.77	0.92	206.83	NA	46.13
20.72	NA	228.74	659.37	-67.77
15.52	0.04	22.46	NA	36.63

Step 1:

Step 2:

Stupid Strategy

Variable 1	Variable 2	Variable 3	Variable 4	Variable 5
8.95	0.99	NA	370.53	-34.97
NA	NA	168.02	617.46	-99.49
18.38	0.48	58.06	1177.66	-44.21
16.10	NA	35.39	NA	77.35
23.14	NA	230.44	408.97	-71.79
24.77	0.92	206.83	NA	46.13
20.72	NA	228.74	659.37	-67.77
15.52	0.04	22.46	NA	36.63

Variable 1	Variable 3	Variable 5
8.95	NA	-34.97
NA	168.02	-99.49
18.38	58.06	-44.21
16.10	35.39	77.35
23.14	230.44	-71.79
24.77	206.83	46.13
20.72	228.74	-67.77
15.52	22.46	36.63

Sampling

A population can be defined as including all people or items with the characteristic one wishes to understand.

Sampling is about to find a representative subset of that population.

Data represents the traces of the real-world processes, and exactly which traces we gather are decided by our sampling method.

There are two sources of randomness and uncertainty:

- (1) the randomness and uncertainty underlying the real-world process itself, and**
- (2) the uncertainty associated with the underlying sampling method.**

Sampling in Times of Big Data

Question:

Is there any need for sampling in times of Big Data? Why not "N=ALL"?

Answer:

Data is not objective! Data does not speak for itself. Data is just a quantitative echo of the events of our society.

Examples:

- When analyzing the probability of customers terminating the relationship, a very small proportion of terminating customers (e.g. 0.2%) on the whole may result in a bias.
- When analyzing political attitudes via social media data, there might be a bias if people with specific attitudes are posting more frequently.
- Ignoring causation can be a flaw. When comparing women and men with the exact same qualifications that have been hired in the past, one must be sure that the company treats female employees in the same way than males. Otherwise, this may result in females tending to leave more often, getting promoted less often, and giving more negative feedback on their environments when compared to the men.

Reasons for Sampling

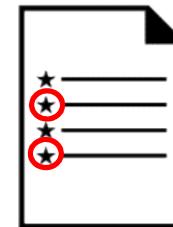
- The volume of data is too large to capture and process
- Design the analytics process using a subset of the data for performance reasons. Later use the complete data set.
- The data set doesn't perfectly represent the target population.
- The data set is imbalanced.
- Use sampling to partition into training and test data.
- ...

Popular Methods of Sampling (I)

Systematic Sampling Simple Random Sampling

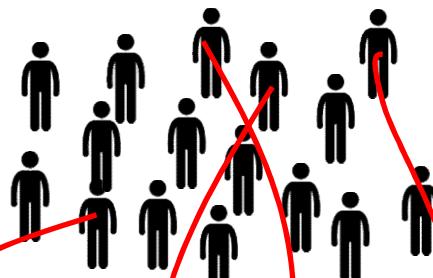


List with names



For example:
Choose every row
with an even number

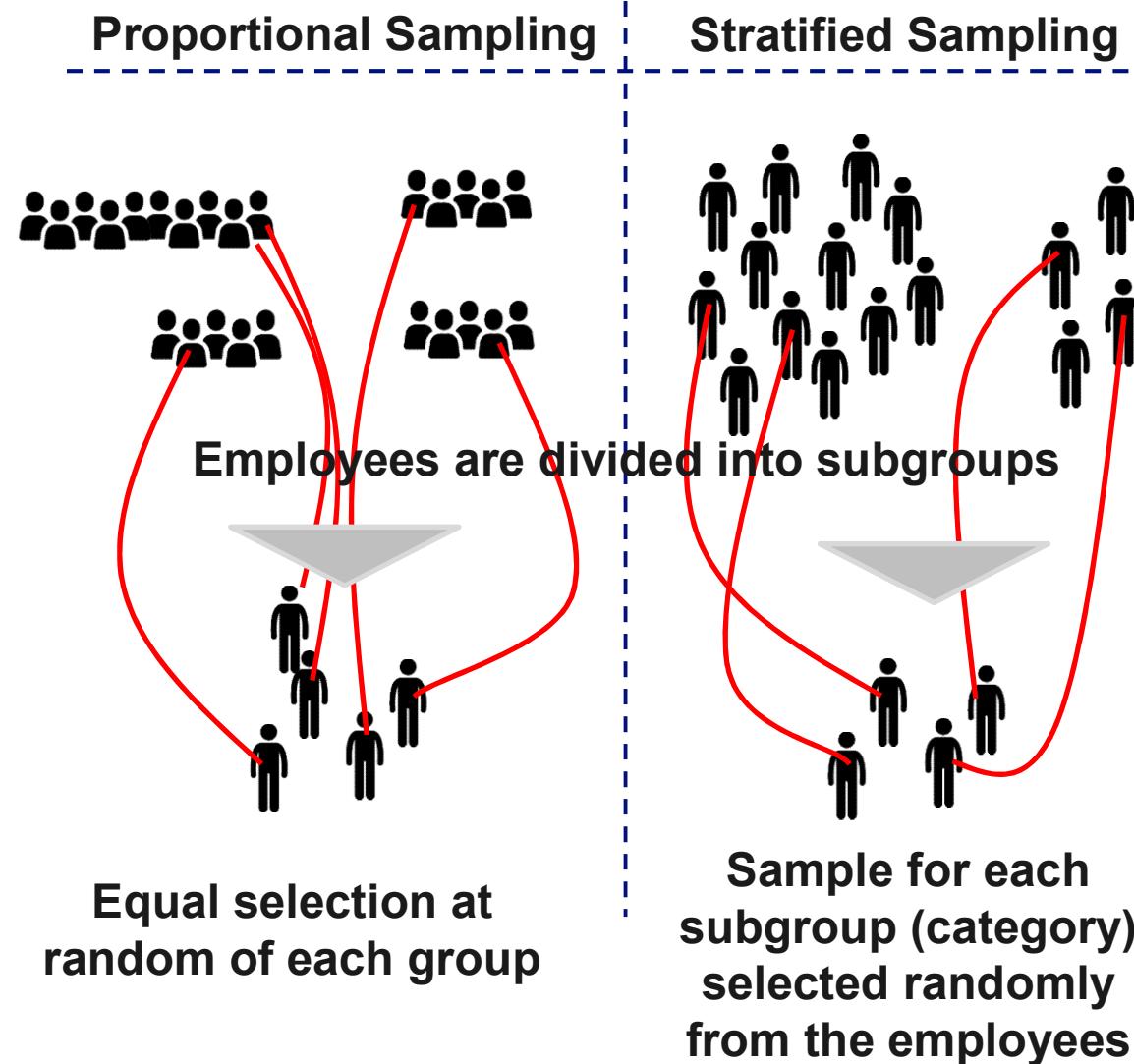
Simple Random Sampling



5000 employees

100 employees
chosen at random

Popular Methods of Sampling (II)



Systematic Sampling

Chooses every row with an even number

day.mins	day.calls	day.charge	intl.mins	intl.calls	intl.charge	churn
157	79	26.69	7.1	6	1.92	No
184.5	97	31.37	8.7	4	2.35	No
258.6	84	43.96	11.2	5	3.02	No
129.1	137	21.95	12.7	6	3.43	Yes
187.7	127	31.91	9.1	5	2.46	No
128.8	96	21.9	11.2	2	3.02	No
156.6	88	26.62	12.3	5	3.32	No
120.7	70	20.52	13.1	6	3.54	No
332.9	67	56.59	5.4	9	1.46	Yes
196.4	139	33.39	13.8	4	3.73	No
190.7	114	32.42	8.1	3	2.19	No
189.7	66	32.25	10	5	2.7	No
224.4	90	38.15	13	2	3.51	No
155.1	117	26.37	10.6	4	2.86	No
62.4	89	10.61	5.7	6	1.54	Yes
84.8	95	14.42	14.2	6	3.83	No
226.1	105	38.44	10.3	5	2.78	No
212	121	36.04	12.6	10	3.4	No
249.6	118	42.43	11.8	3	3.19	Yes
176.8	94	30.06	8.3	4	2.24	No
140.4	94	23.87	11.1	9	3	No
126.3	102	21.47	9.4	2	2.54	No
173.1	85	29.43	14.6	15	3.94	Yes
124.8	82	21.22	10	4	2.7	No
85.8	77	14.59	9.2	4	2.48	No

Yes = 11
No = 11

Random Sampling

Creates totally random samples

Example: 40%

day.mins	day.calls	day.charge	intl.mins	intl.calls	intl.charge	churn
157	79	26.69	7.1	6	1.92	No
184.5	97	31.37	8.7	4	2.35	No
258.6	84	43.96	11.2	5	3.02	No
129.1	137	21.95	12.7	6	3.43	Yes
187.7	127	31.91	9.1	5	2.46	No
128.8	96	21.9	11.2	2	3.02	No
156.6	88	26.62	12.3	5	3.32	No
120.7	70	20.52	13.1	6	3.54	No
332.9	67	56.59	5.4	9	1.46	Yes
196.4	139	33.39	13.8	4	3.73	No
190.7	114	32.42	8.1	3	2.19	No
189.7	66	32.25	10	5	2.7	No
224.4	90	38.15	13	2	3.51	No
155.1	117	26.37	10.6	4	2.86	No
62.4	89	10.61	5.7	6	1.54	Yes
84.8	95	14.42	14.2	6	3.83	No
226.1	105	38.44	10.3	5	2.78	No
212	121	36.04	12.6	10	3.4	No
249.6	118	42.43	11.8	3	3.19	Yes
176.8	94	30.06	8.3	4	2.24	No
140.4	94	23.87	11.1	9	3	No
126.3	102	21.47	9.4	2	2.54	No
173.1	85	29.43	14.6	15	3.94	Yes
124.8	82	21.22	10	4	2.7	No
85.8	77	14.59	9.2	4	2.48	No

Yes = 1
No = 9

Proportional Sampling

Creates proportional samples

Example: 40%

day.mins	day.calls	day.charge	intl.mins	intl.calls	intl.charge	churn
157	79	26.69	7.1	6	1.92	No
184.5	97	31.37	8.7	4	2.35	No
258.6	84	43.96	11.2	5	3.02	No
129.1	137	21.95	12.7	6	3.43	Yes
187.7	127	31.91	9.1	5	2.46	No
128.8	96	21.9	11.2	2	3.02	No
156.6	88	26.62	12.3	5	3.32	No
120.7	70	20.52	13.1	6	3.54	No
332.9	67	56.59	5.4	9	1.46	Yes
196.4	139	33.39	13.8	4	3.73	No
190.7	114	32.42	8.1	3	2.19	No
189.7	66	32.25	10	5	2.7	No
224.4	90	38.15	13	2	3.51	No
155.1	117	26.37	10.6	4	2.86	No
62.4	89	10.61	5.7	6	1.54	Yes
84.8	95	14.42	14.2	6	3.83	No
226.1	105	38.44	10.3	5	2.78	No
212	121	36.04	12.6	10	3.4	No
249.6	118	42.43	11.8	3	3.19	Yes
176.8	94	30.06	8.3	4	2.24	No
140.4	94	23.87	11.1	9	3	No
126.3	102	21.47	9.4	2	2.54	No
173.1	85	29.43	14.6	15	3.94	Yes
124.8	82	21.22	10	4	2.7	No
85.8	77	14.59	9.2	4	2.48	No

Yes = 2
No = 8

Stratified Sampling

Creates balanced samples

Example: 40%

day.mins	day.calls	day.charge	intl.mins	intl.calls	intl.charge	churn
157	79	26.69	7.1	6	1.92	No
184.5	97	31.37	8.7	4	2.35	No
258.6	84	43.96	11.2	5	3.02	No
129.1	137	21.95	12.7	6	3.43	Yes
187.7	127	31.91	9.1	5	2.46	No
128.8	96	21.9	11.2	2	3.02	No
156.6	88	26.62	12.3	5	3.32	No
120.7	70	20.52	13.1	6	3.54	No
332.9	67	56.59	5.4	9	1.46	Yes
196.4	139	33.39	13.8	4	3.73	No
190.7	114	32.42	8.1	3	2.19	No
189.7	66	32.25	10	5	2.7	No
224.4	90	38.15	13	2	3.51	No
155.1	117	26.37	10.6	4	2.86	No
62.4	89	10.61	5.7	6	1.54	Yes
84.8	95	14.42	14.2	6	3.83	No
226.1	105	38.44	10.3	5	2.78	No
212	121	36.04	12.6	10	3.4	No
249.6	118	42.43	11.8	3	3.19	Yes
176.8	94	30.06	8.3	4	2.24	No
140.4	94	23.87	11.1	9	3	No
126.3	102	21.47	9.4	2	2.54	No
173.1	85	29.43	14.6	15	3.94	Yes
124.8	82	21.22	10	4	2.7	No
85.8	77	14.59	9.2	4	2.48	No

Yes = 5
No = 5

Feature Engineering

Feature engineering is the process of using domain knowledge of the data to create features that make machine learning algorithms work. If feature engineering is done correctly, it increases the predictive power of machine learning algorithms by creating features from raw data that help facilitate the machine learning process.

A feature (variable, attribute) is depicted by a column in a dataset. Considering a generic two-dimensional dataset, each observation is depicted by a row and each feature by a column, which will have a specific value for an observation:

		Columns (Features)		
		Feature 1	Feature 2	Feature 3
Rows (Observations) 	0	18.38	58.06	-44.21
	1	16.10	35.39	77.35
	2	23.14	230.44	-71.79
	3	24.77	206.83	46.13

Features can be of two major types. Raw features are obtained directly from the dataset with no extra data manipulation or engineering. Derived features are usually obtained from feature engineering, where we extract features from existing data attributes. A simple example would be creating a new feature "Age" from an employee dataset containing "Birthdate".

Variants of Feature Engineering

1. Transformation

- convert features (e.g. birth date to age)
- build lag structures (e.g. time-lags)
- normalization / standardization / scaling

2. Type Conversion

- if numerical type is needed, transformation categorical into numerical data using dummy features
- if categorical type is needed or more informative, discretize numerical features (e.g. income to income classes like poor and rich)

3. Feature Combination

- create interaction features (e.g. school_score = num_schools x median_school with num_school = number of schools within 5 miles of a property and median_school = median quality score of those schools)
- combine categories (e.g. because they have very few observations or they would result in too many dummy features)

4. Feature Composition

- build ratios (e.g. returns from prices)
- Principal Component Analysis (Dimensionality Reduction)

Scaling

Most datasets contain features highly varying in magnitudes, units and range. Most machine learning algorithms have problems with this because they use distance measures or calculate gradients. The features with high magnitudes will weigh in a lot more in the distance calculations than features with low magnitudes and gradients may end up taking a long time or are not accurately calculable.

To overcome this effect, we scale the features to bring them to the same level of magnitudes. The two most discussed scaling methods are Normalization and Standardization.

Normalization typically means to rescale the values into a range of [0,1]:

$$\hat{x}_i = \frac{x_i - x_{\min}}{x_{\max} - x_{\min}}$$

Standardization typically means to rescale data to have a mean of 0 and a standard deviation of 1 (Z scores):

$$\hat{x}_i = \frac{x_i - \mu}{\sigma}$$

If necessary scaling can be inverted:

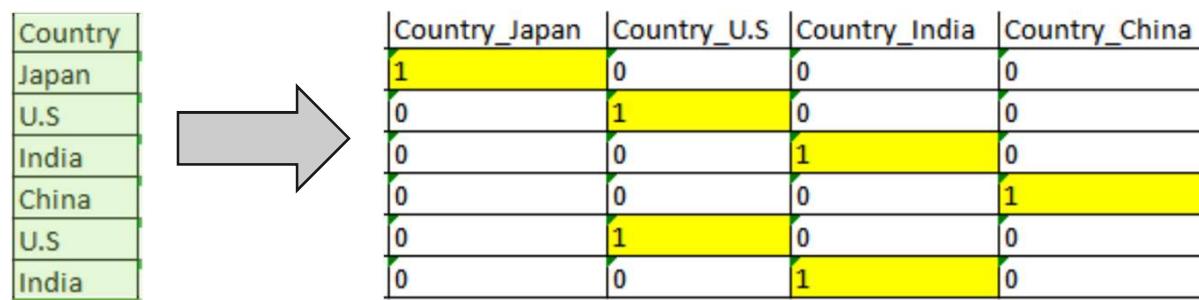
$$x_i = \hat{x}_i \cdot (x_{\max} - x_{\min}) + x_{\min} \quad \text{resp.} \quad x_i = \hat{x}_i \cdot \sigma + \mu$$

Type Conversion (Encoding)

Many machine learning algorithms cannot work with categorical data directly. To convert categorical data to numbers, there exist two variants:

Label encoding refers to transforming the word labels into numerical form so that the algorithms can understand how to operate on them. Every categorical value is assigned to one numerical value, e.g. young -> 1, middle_age -> 2, old -> 3. This only works in specific situations where you have somewhat continuous-like data, e.g. if the categorical feature is ordinal.

One hot encoding is a representation of a categorical variable as binary vectors. Every categorical value is assigned to an artificial binary variable. If the corresponding categorical value occurs in a data row the value of its binary replacement is equal to 1 else 0, e.g.



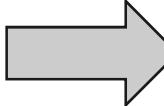
The diagram illustrates the transformation of categorical data into a one-hot encoded matrix. On the left, a vertical list of categories is shown in a light green box, labeled "Country" at the top. The categories listed are Japan, U.S., India, China, U.S., and India. A large gray arrow points from this list to the right, where a 6x4 grid represents the one-hot encoding. The columns are labeled "Country_Japan", "Country_U.S.", "Country_India", and "Country_China". The rows correspond to the categories in the list. The values in the grid indicate the presence (1) or absence (0) of each category in a given row. For example, the first row (Japan) has a 1 in the "Country_Japan" column and 0s in the other three. The second row (U.S.) has 0s in all columns. The third row (India) has 0s in all columns. The fourth row (China) has 0s in all columns. The fifth row (U.S.) has 1 in the "Country_U.S." column and 0s in the others. The sixth row (India) has 0s in all columns.

Country	Country_Japan	Country_U.S.	Country_India	Country_China
Japan	1	0	0	0
U.S.	0	1	0	0
India	0	0	1	0
China	0	0	0	1
U.S.	0	1	0	0
India	0	0	1	0

It is usual when creating dummy variables to have one less variable than the number of categories present to avoid perfect collinearity (dummy variable trap).

Example of Feature Engineering (I)

Data sets often contain date/time features. These features are rarely useful in their original form because they only contain ongoing values. However, they can be useful for extracting cyclical factors, such as weekly or seasonal effects. Suppose, we are given a data "flight date time vs status". Then, given the date-time data, we have to predict the status of the flight.

	Date_Time_Combined	Status		Hour_Of_Day	Status
0	2018-02-14 20:40	Delayed		0	20 Delayed
1	2018-02-15 10:30	On Time		1	10 On Time
2	2018-02-14 07:40	On Time		2	7 On Time
3	2018-02-15 18:10	Delayed		3	18 Delayed
4	2018-02-14 10:20	On Time		4	10 On Time

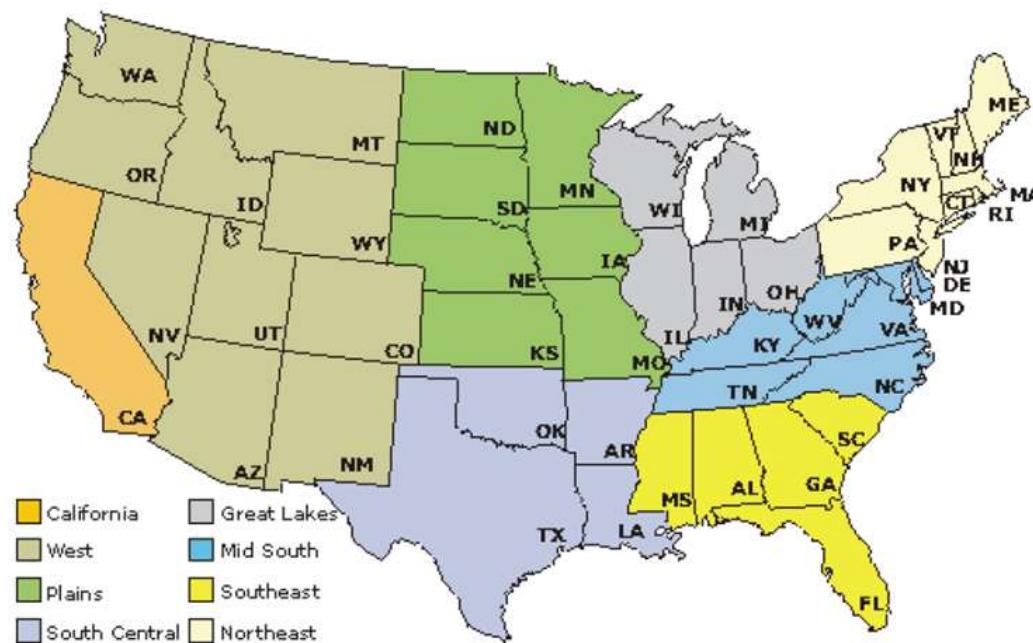
But the status of the flight may depend on the hour of the day, not on the date-time. To analyze this, we will create the new feature "Hour_Of_Day". Using the "Hour_Of_Day" feature, the machine will learn better as this feature is directly related to the status of the flight.

Source: Shekhar, A.: What Is Feature Engineering for Machine Learning?, medium.com.

Example of Feature Engineering (II)

Suppose we are given the latitude, longitude and other data with the objective to predict the target feature "Price_Of_House". Latitude and longitude are not of use in this context if they are alone. So, we will combine the latitude and the longitude to make one feature.

In other cases, it might be appropriate to transform latitude and longitude into categories which reflect regions, for example



Example of Feature Engineering (III)

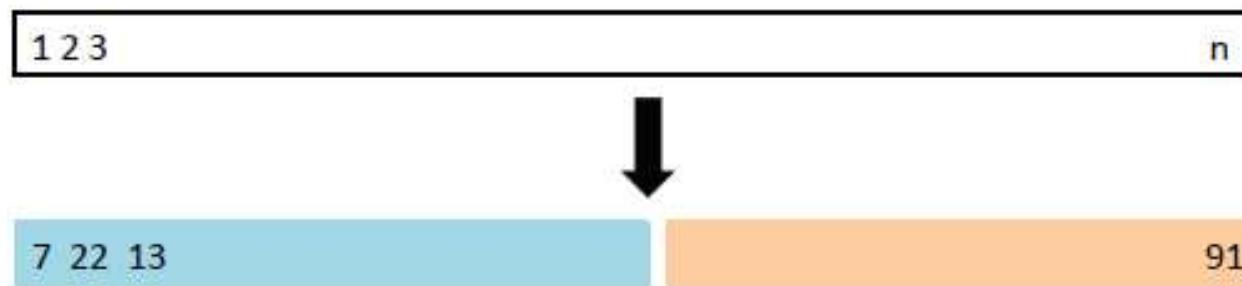
Suppose we are given a feature "Marital_Status" and other data with the objective to classify customers into "Creditworthy" and "Not_Creditworthy". In the data set the marital status has many different values, for example

- single living alone
- single living with his parents
- married living together
- married living separately
- divorced
- divorced but living together
- registered partnerships
- living in marriage-like community
- widowed
- ...

To avoid a transformation into too many and maybe dominating dummy features, we can group the similar classes, e.g. in single, married, widowed. If there exist some remaining sparse classes which cannot be assigned in a meaningful way they can be joined into a single "Other" class.

Partitioning the Data

The partitioning of the data in **Training and Test Data** has the aim to proof if the analytical results can be generalized. The analysis (e.g. the development of a classifier) is carried out on the basis of training data. Subsequently, the results are applied to the test data. If the results are significantly worse than the training data, the model is not generalizable, which is called overfitting.



The partitioning of the data in training and test data can be carried out in the following ways:

- By random/stratified/... sampling (problem with the repeatability)
- according to a list
- according to rules (e.g. the first/last 50 records or every twelfth)

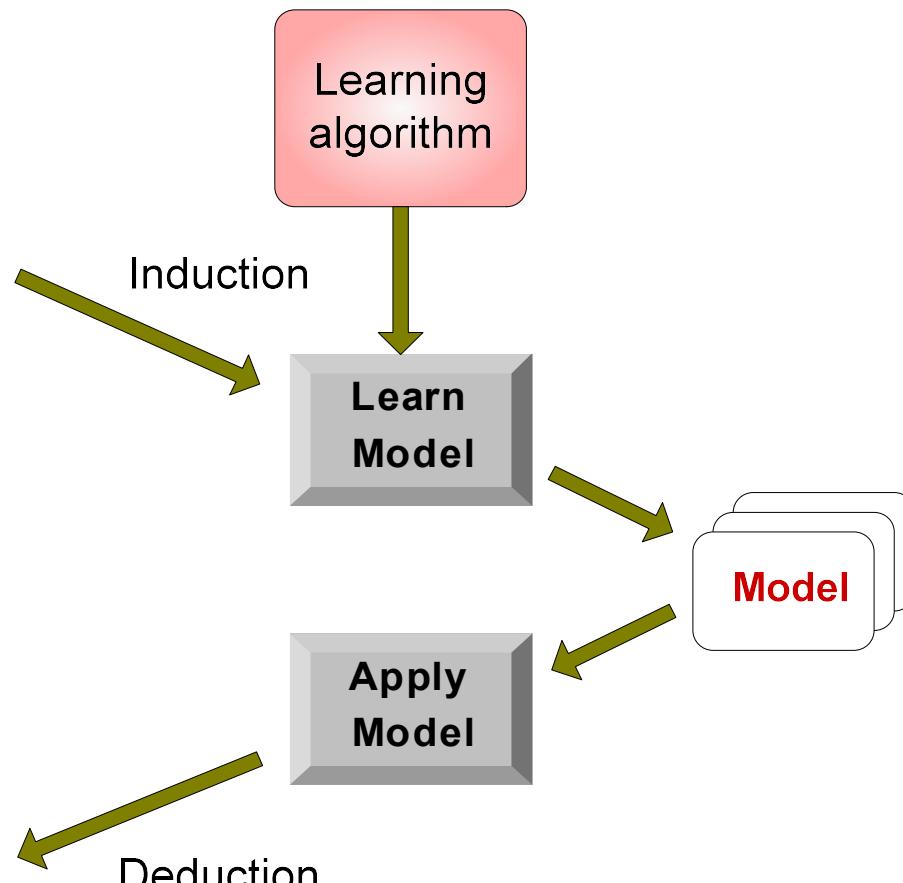
Applying Training and Test Data

Tid	Attrib1	Attrib2	Attrib3	Class
1	Yes	Large	125K	No
2	No	Medium	100K	No
3	No	Small	70K	No
4	Yes	Medium	120K	No
5	No	Large	95K	Yes
6	No	Medium	60K	No
7	Yes	Large	220K	No
8	No	Small	85K	Yes
9	No	Medium	75K	No
10	No	Small	90K	Yes

Training Set

Tid	Attrib1	Attrib2	Attrib3	Class
11	No	Small	55K	?
12	Yes	Medium	80K	?
13	Yes	Large	110K	?
14	No	Small	95K	?
15	No	Large	67K	?

Test Set



Source: <http://www.cs.kent.edu/~jin/BigData/Lecture10-ML-Classification.pptx>

Partitioning

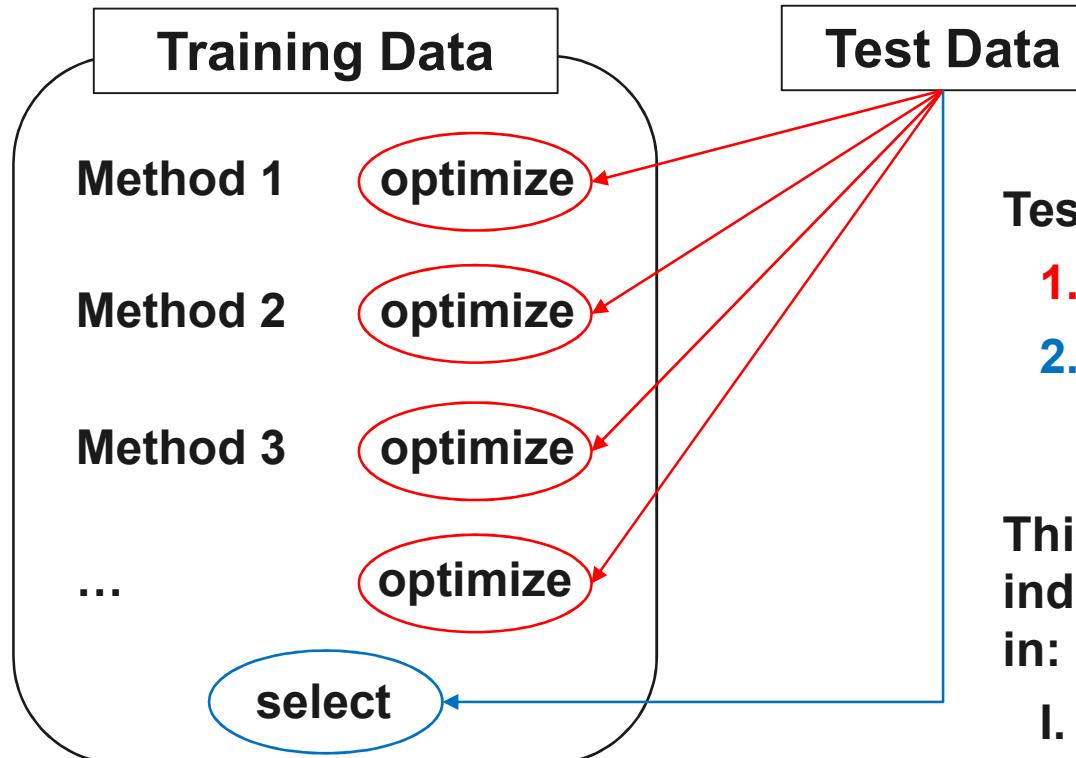
**Partition into
training and test
data**

Example:
Training = 60%
Test = 40%

day.mins	day.calls	day.charge	intl.mins	intl.calls	intl.charge	churn
258.6	84	43.96	11.2	5	3.02	No
129.1	137	21.95	12.7	6	3.43	Yes
156.6	88	26.62	12.3	5	3.32	No
332.9	67	56.59	5.4	9	1.46	Yes
190.7	114	32.42	8.1	3	2.19	No
62.4	89	10.61	5.7	6	1.54	Yes
226.1	105	38.44	10.3	5	2.78	No
249.6	118	42.43	11.8	3	3.19	Yes
126.3	102	21.47	9.4	2	2.54	No
173.1	85	29.43	14.6	15	3.94	Yes



Problems with fixed Training and Test Samples



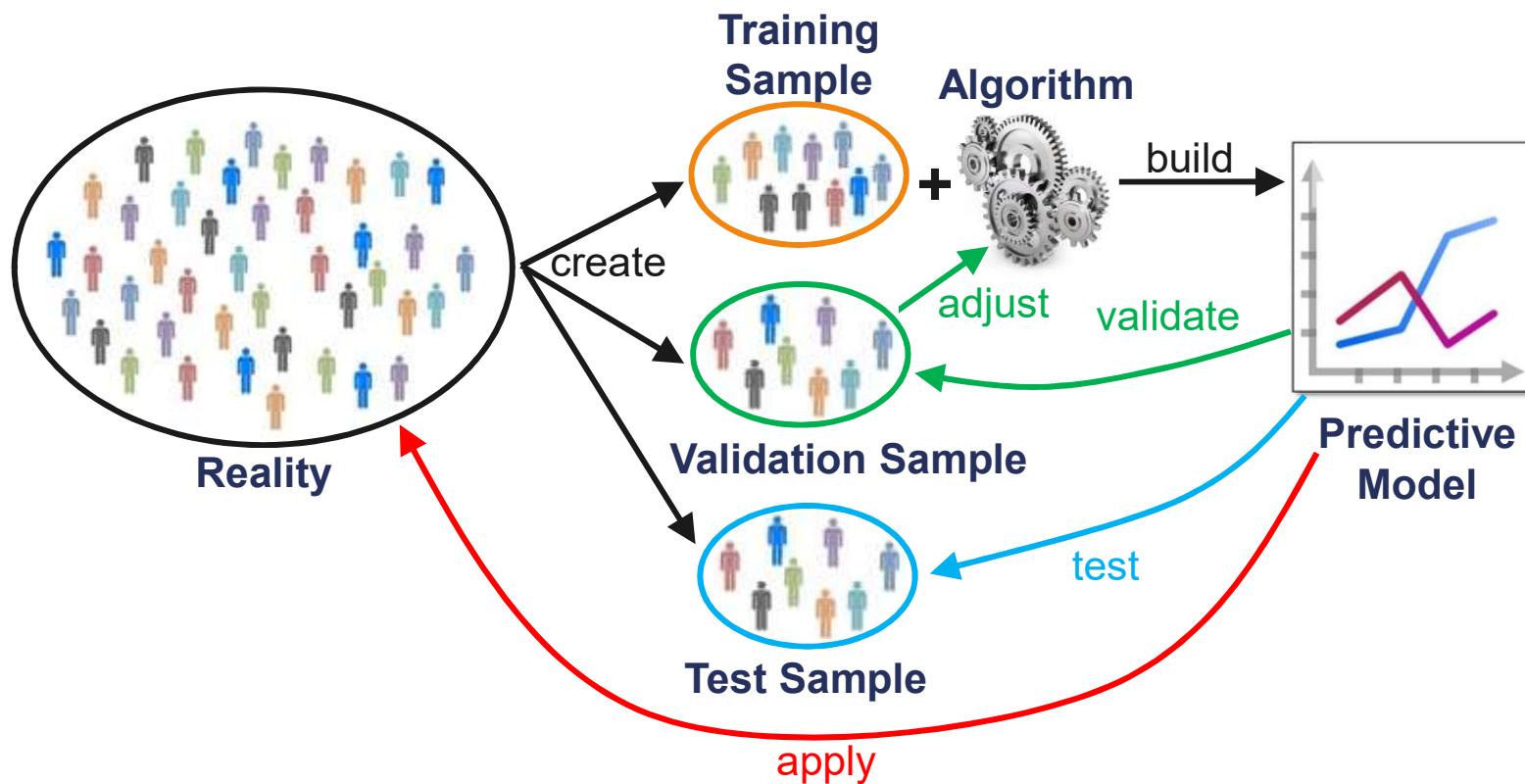
Test data is used for two things:

- 1. Optimize the model training**
- 2. Select the best model via testing the model quality**

This contradicts the idea of independent testing and results in:

- I. Endogenization of the test data**
- II. Selection Bias**

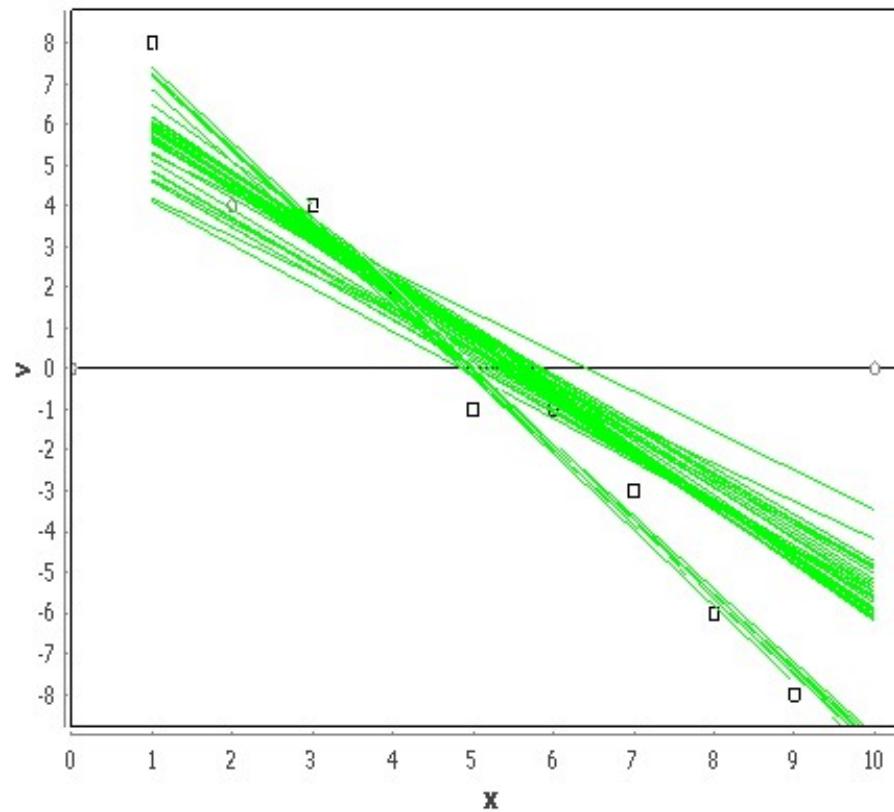
Addressing the Endogeneity Problem



Selection Bias

Training and test error can be highly variable, depending on precisely which observations are included in the training set and which observations are included in the validation set (**Selection Bias**).

Example of different OLS models as a result of different samples:



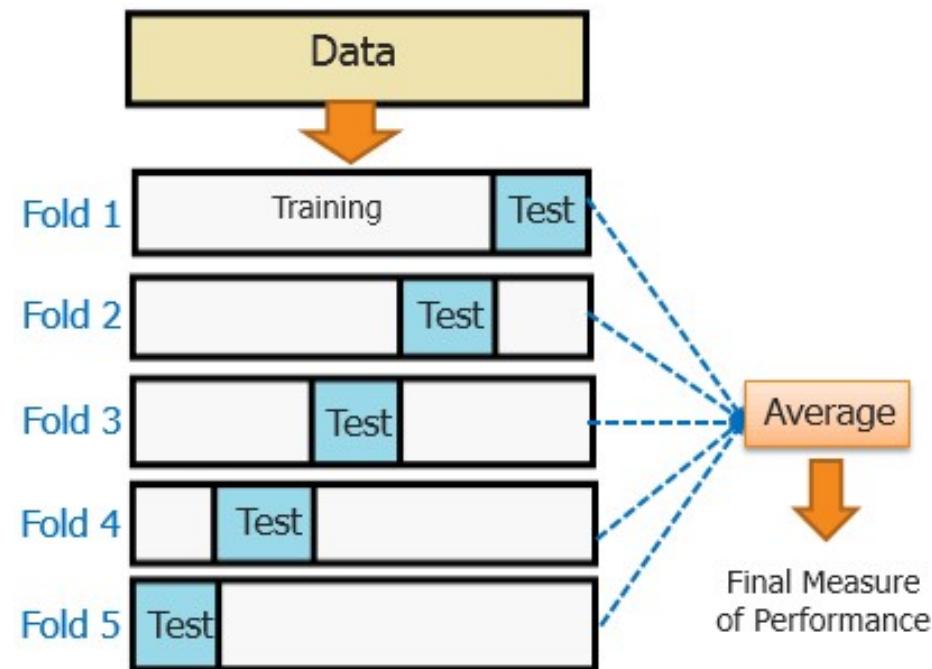
To avoid such problems, one can use so-called resampling methods.

Cross Validation

Cross Validation is a resampling technique for estimating the performance of a predictive model.

The idea is to randomly divide the data into k equal-sized parts. We leave out part k , fit the model to the other $k-1$ parts (combined), and then obtain predictions for the left-out k -th part. This is done in turn for each part ($i = 1, 2, \dots, k$) and then the results are combined.

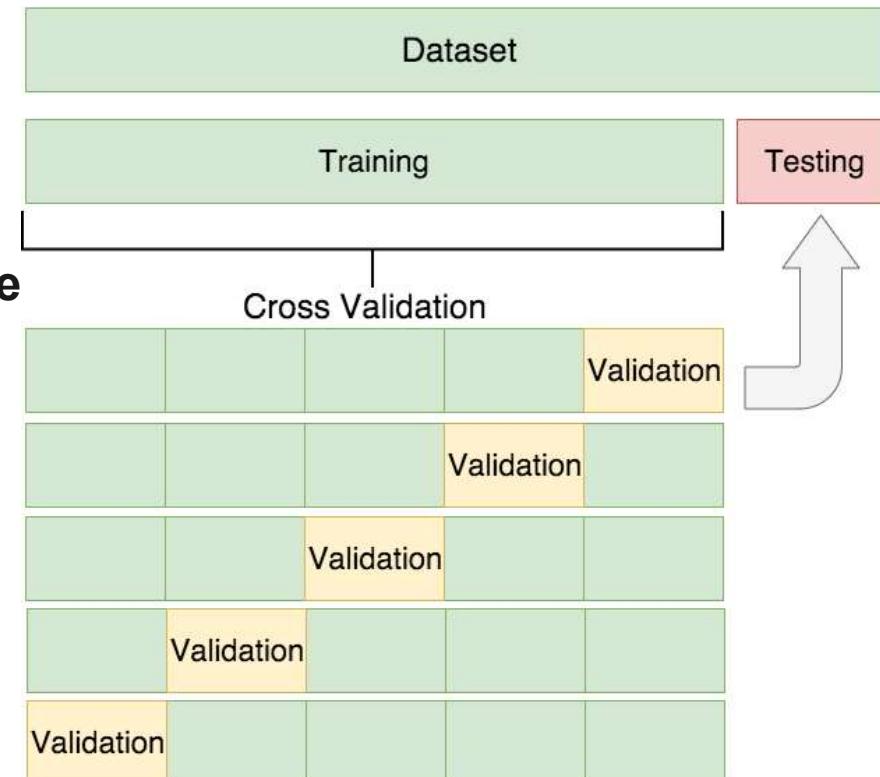
K = 5 or 10 are typical values.



Cross Validation

In Data Science cross validation can be used for model selection and adjustment. In these cases, cross validation is applied to the training data set. For every iteration, $k-1$ folds are used for model fitting and the remaining fold for testing the model (Validation). Every time, the quality measure (e.g. accuracy) for the validation fold is captured. At the end of this step, the average and the standard deviation of the measures are calculated. The best model is the one with the best ratio in high average and low standard deviation.

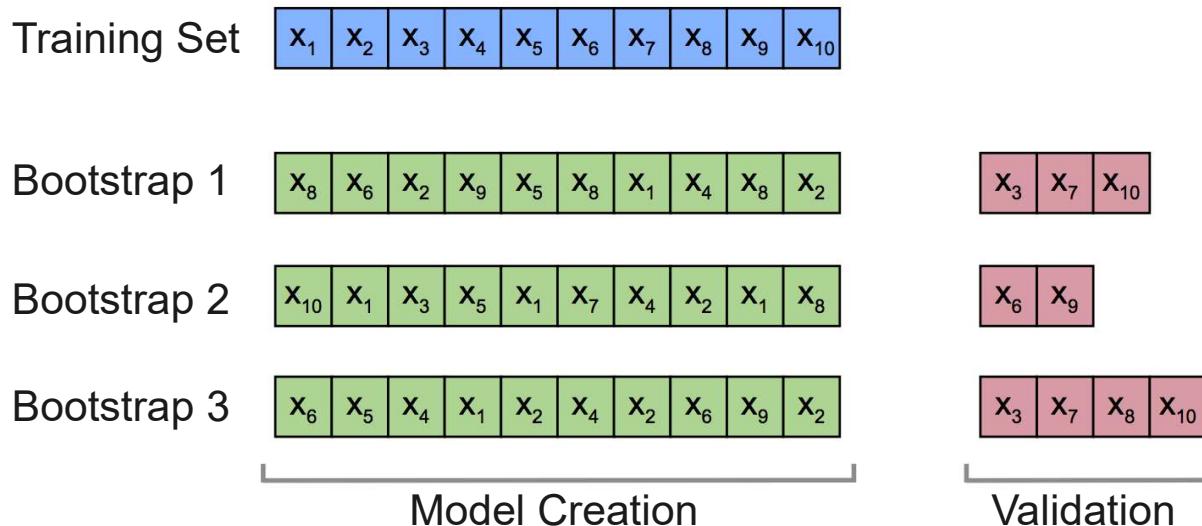
Once the model type and its optimal parameters have been selected, a final model is trained using these hyperparameters on the *full* training set, and the generalization quality is measured on the test set.



Bootstrapping

The bootstrap method involves taking the training set of n objects, and sampling from it to form new samples that are also of size n. The bootstrap sample is taken from the training set using sampling with replacement. This process is repeated a number of times.

For each of these bootstrap samples a model is trained and validated using the data that was not chosen for the subsample set. The final measure of performance is the average of the validation errors of the different models.



Exploratory Data Analysis

Exploratory Data Analysis

In Exploratory Data Analysis (EDA), there is no hypothesis and there is no model.

People are not very good at looking at a column of numbers or a whole data table and then determining important characteristics of the data. EDA techniques have been devised as an aid in this situation.

Reasons for EDA:

- gain intuition about the data
- make comparisons between distributions
- sanity checking (making sure the data is on the scale you expect, in the format you thought it should be)
- find out where data is missing or if there are outliers
- summarize the data

Exploratory data analysis is generally cross-classed in two ways. First, each method is either non-graphical or graphical. And second, each method is either univariate or multivariate.

Univariate Non-Graphical EDA

Non-graphical exploratory data analysis is the first step when beginning to analyze the data. This preliminary data analysis step focuses on four points:

- **measures of central tendency, i.e. mean and median. The median, known as 50th percentile, is more resistant to outliers.**
- **measures of spread, i.e. variance, standard deviation, and interquartile range**
- **the shape of the distribution**
- **the existence of outliers**

The characteristics of interest for a categorical variable are simply the range of values and the frequency of occurrence for each value.

	ACLENGTH	INTPLAN	VMPLAN	NVMAIL	DMIN	DCALL
count	999.000000	999.000000	999.000000	999.000000	999.000000	999.000000
mean	100.831832	0.174174	0.235235	6.992993	191.904404	101.054054
std	39.206659	0.379449	0.424358	13.076044	62.131982	20.346032
min	1.000000	0.000000	0.000000	0.000000	0.000000	0.000000
25%	74.000000	0.000000	0.000000	0.000000	148.900000	88.000000
50%	99.000000	0.000000	0.000000	0.000000	190.700000	102.000000
75%	127.000000	0.000000	0.000000	0.000000	237.000000	115.000000
max	225.000000	1.000000	1.000000	48.000000	350.800000	165.000000

Tests on Outliers

Outlier are data objects, which are clearly different from the others.

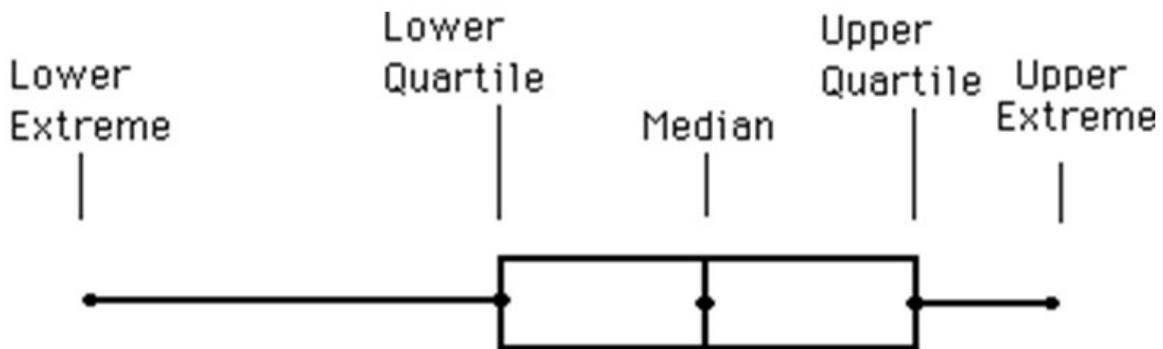
Usually, the detection of outliers is an unsupervised process, because they are not known before analyses.

In the case of **numerical attributes** the Interquartile Range can be used. Here, an outlier is defined if the attribute lies outside the interval

$$[Q_{0.25} - k \cdot (Q_{0.75} - Q_{0.25}), Q_{0.75} + k \cdot (Q_{0.75} - Q_{0.25})]$$

Usually, k has a value between 1.5 and 3. The bigger k, the more different the values must be to be classified as outliers.

Can be visualized by a Box-and-Whisker Plot:



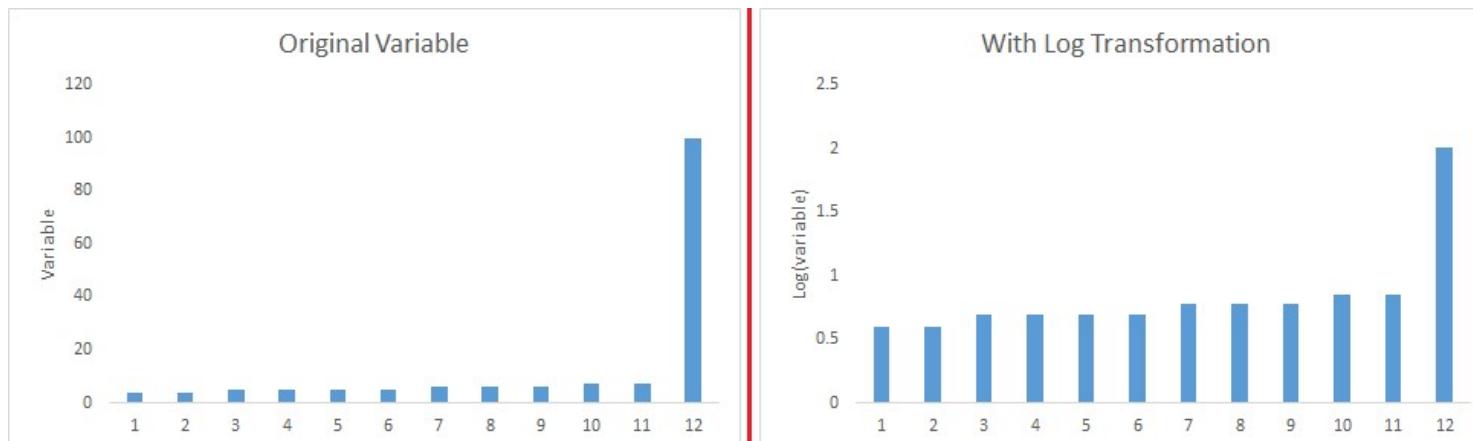
Handling Outliers

Outlier have to be **eliminated** if they

1. would bias the analysis, e.g. if 9 persons have an age between 20 and 30 and the 10th person is 80 years old.
2. are erroneous data, e.g. as a result of input errors or a defect sensor.

It is not always acceptable to drop an observation just because it is an outlier. They can be legitimate observations and are sometimes interesting ones. It's important to investigate the nature of the outlier before deciding.

In those cases where you shouldn't drop the outlier, one option is to try a transformation. **Log transformations** pull in high numbers. This can reduce the impact of a single point if the outlier is an independent variable.



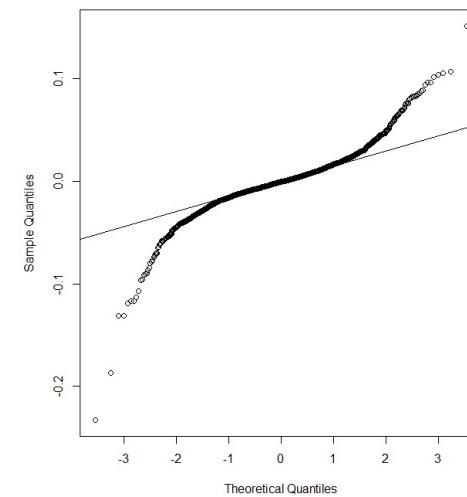
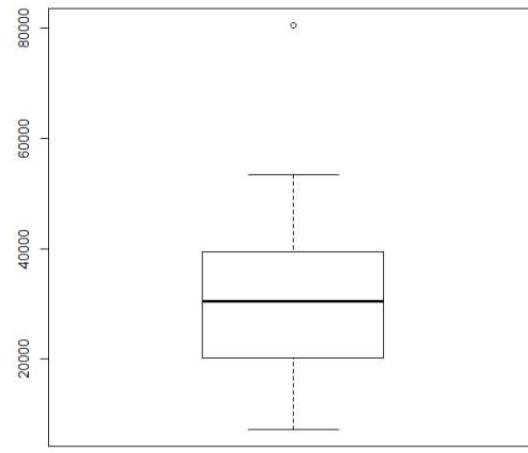
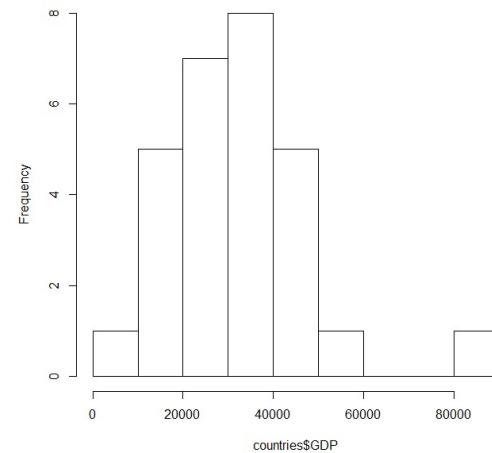
Univariate Graphical EDA

Non-graphical and graphical EDA methods complement each other, they have the same focus. While the non-graphical methods are quantitative and objective, they do not give a full picture of the data. The distribution of a variable tells us what values the variable takes and how often each value occurs.

Types of displays:

for numerical variables: Histograms, Boxplots, Quantile-normal plots, ...

for categorical variables: Pie charts, Bar graphs, ...



Multivariate Non-Graphical EDA

Multivariate non-graphical EDA techniques generally show the relationship between two or more variables in the form of either cross-tabulation for categorical variables or correlation statistics for numerical variables.

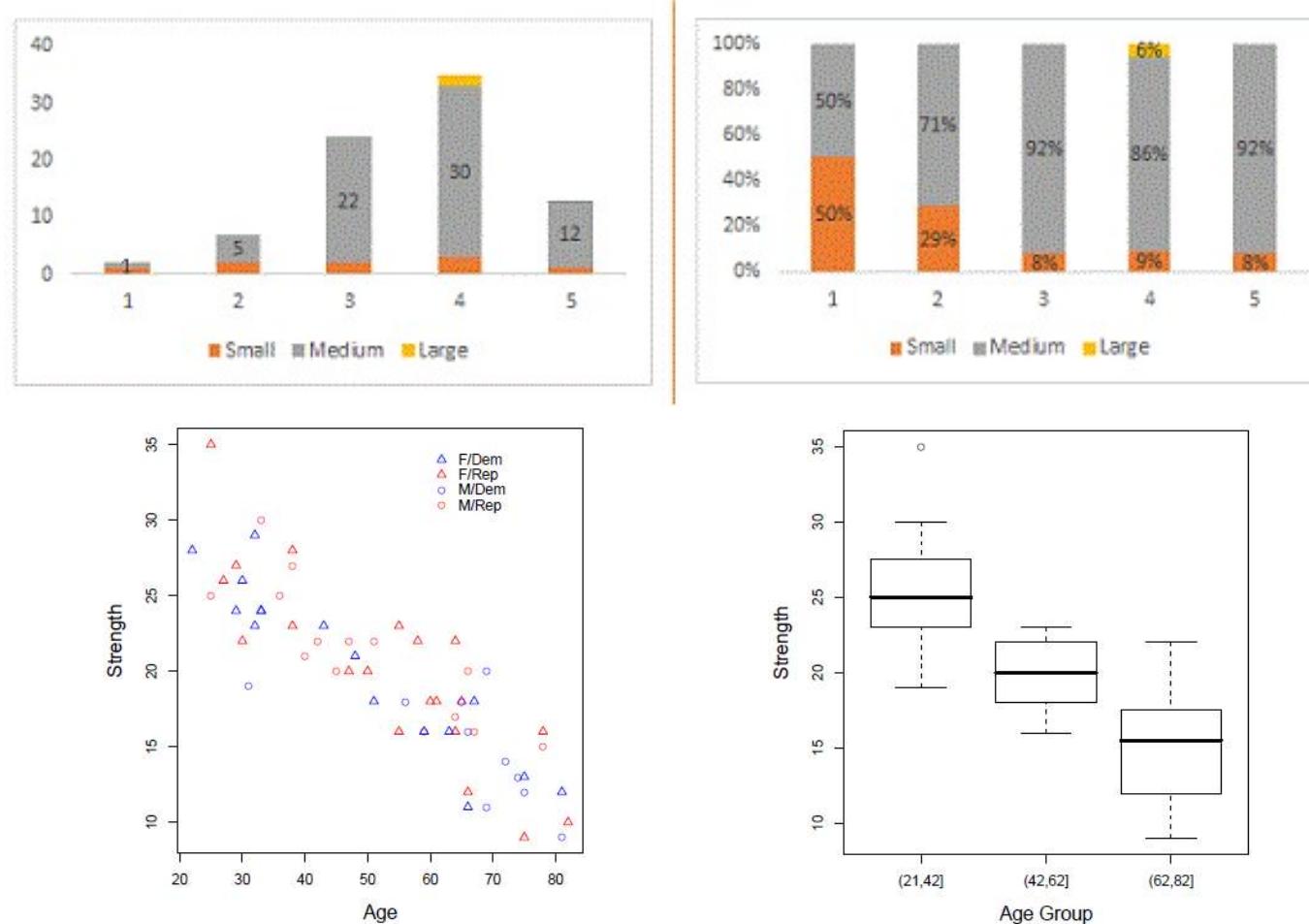
Frequency Row Pct	Product Category					Total
	1	2	3	4	5	
Small	1 11.11	2 22.22	2 22.22	3 33.33	1 11.11	9
Medium	1 1.43	5 7.14	22 31.43	30 42.86	12 17.14	70
Large	0 0.00	0 0.00	0 0.00	2 100.00	0 0.00	2
Total	2	7	24	35	13	81

Frequency Missing = 77

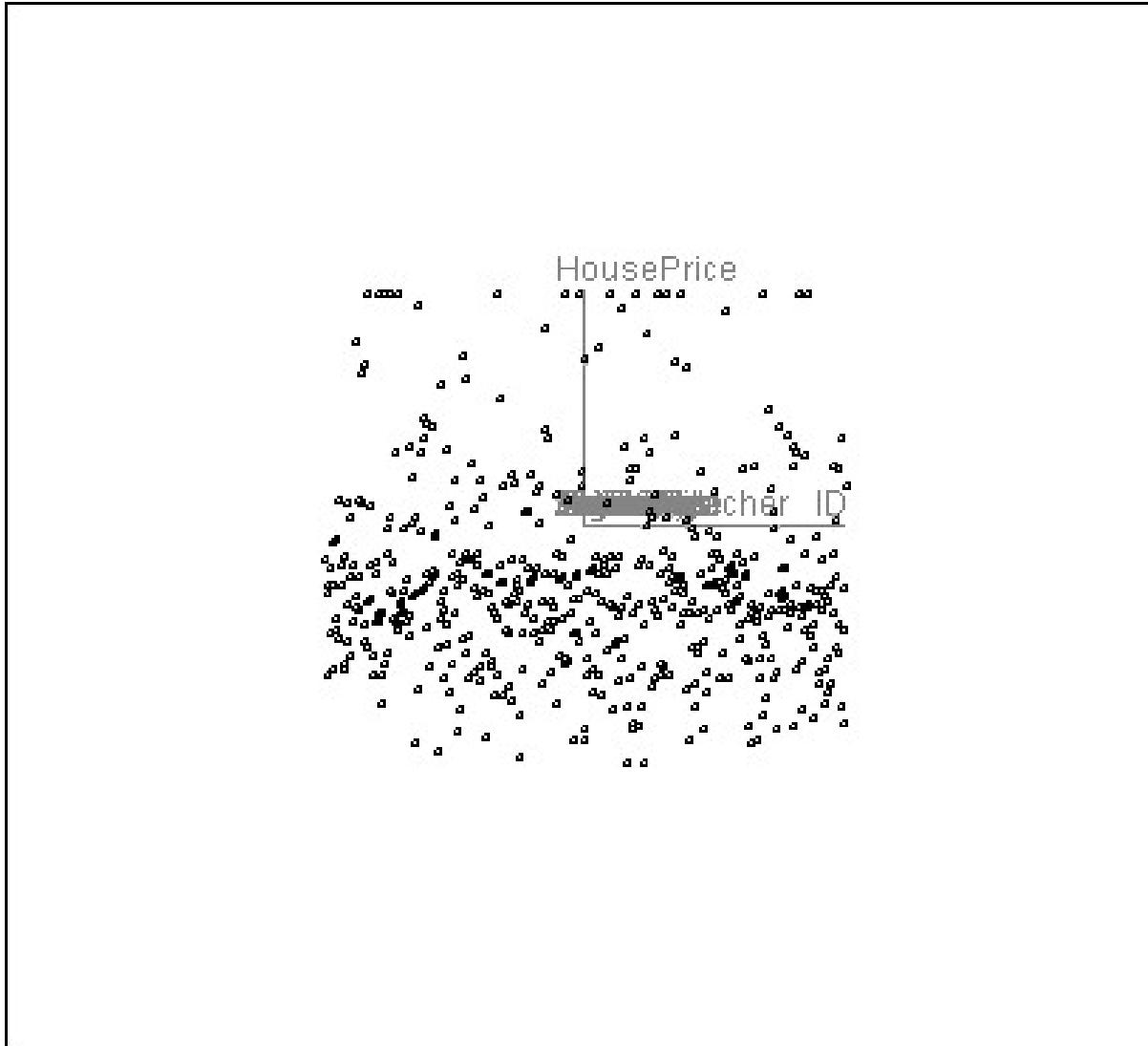
	V2	V3	V4	V5	V6	
V2	-0.388					
V3	0.549	0.189				
V4	0.824	0.758	-0.619			
V5	0.802	0.833	-0.469	-0.946		
V6	-0.186	-0.659	0.349	0.555	0.649	

Multivariate Graphical EDA

Multivariate graphical EDA techniques are scatterplots for numerical variables, Barcharts for categorical variables, or Boxplots for mixed types.



Touring Diagram



Dimensionality Reduction

In times of Big Data, we are often confronted with high-dimensional data. This results in some problems:

- **Curse of dimensionality:** If there are too many dimensions, even the closest neighbors are too far away from each other to realistically be considered "close." Using a measure such as a Euclidean distance results in little differences in the distances between different data objects.
- **Correlated features:** If there are many features, there is a high probability, that they are highly correlated.
- In many cases, the measured data vectors are high-dimensional but we may have reason to believe that the data lie near a lower-dimensional manifold. Learning a suitable low-dimensional manifold from high-dimensional data is essentially the same as learning this underlying source.

To handle those cases Dimensionality Reduction techniques are developed. They can be used for:

- **Data dimensionality reduction:** Produce a compact low-dimensional encoding of a given high-dimensional data set.
- **Data visualization:** Provide a graphical interpretation of a given data set.

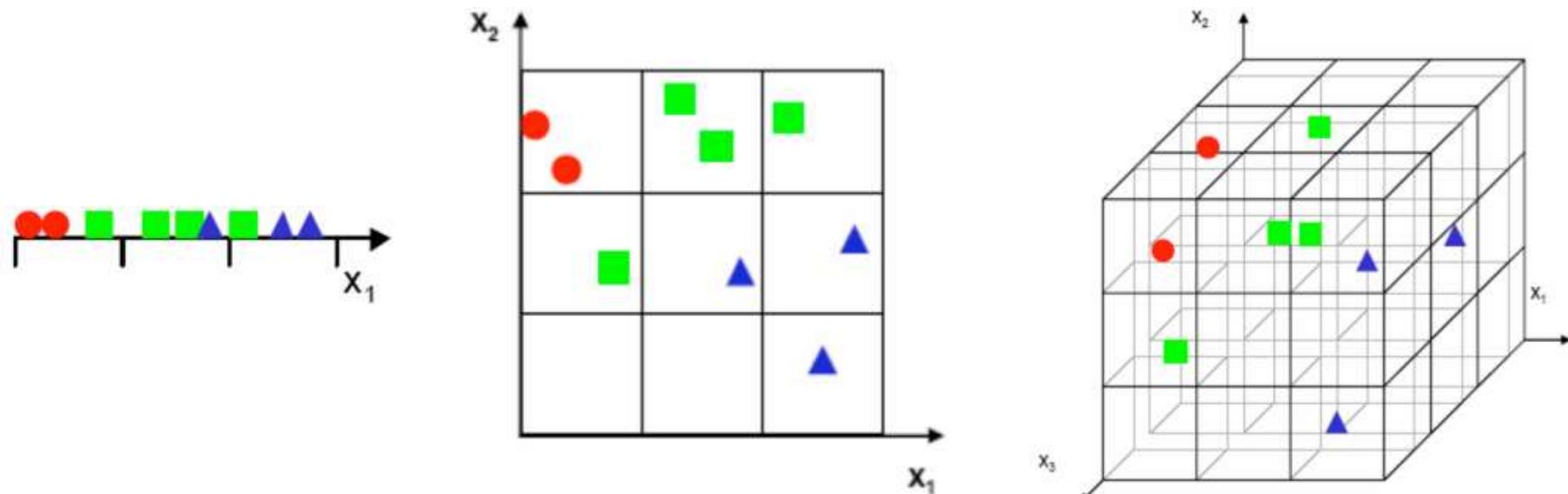
Curse of Dimensionality

Watching TV (in min):

	Ch 1	Ch 2
Customer 1	48	53
Customer 2	128	12
Customer 3	21	106
Customer 4	89	66
Customer 5	65	128
Customer 6	139	101
Customer 7	78	47

	Ch 1	Ch 2	Ch 3	Ch 4	Ch 5	Ch 6	CH 7	CH 8
Customer 1	53	0	62	0	0	0	142	0
Customer 2	0	0	55	0	0	57	0	0
Customer 3	134	0	0	0	112	0	53	133
Customer 4	80	0	112	26	82	112	0	60
Customer 5	0	82	0	0	0	0	119	0
Customer 6	87	109	120	0	0	0	0	0
Customer 7	0	90	0	68	47	0	0	0

MANY possible combinations are NEVER observed ---> sparse



Source: Youtube: Basics of the curse of dimensionality, https://youtu.be/JMmuVyDZ_XA
 Introduction to Data Analytics in Business

Reduction of Features

Reduction of features might not change the overall meaning, but may reduce the complexity significantly.

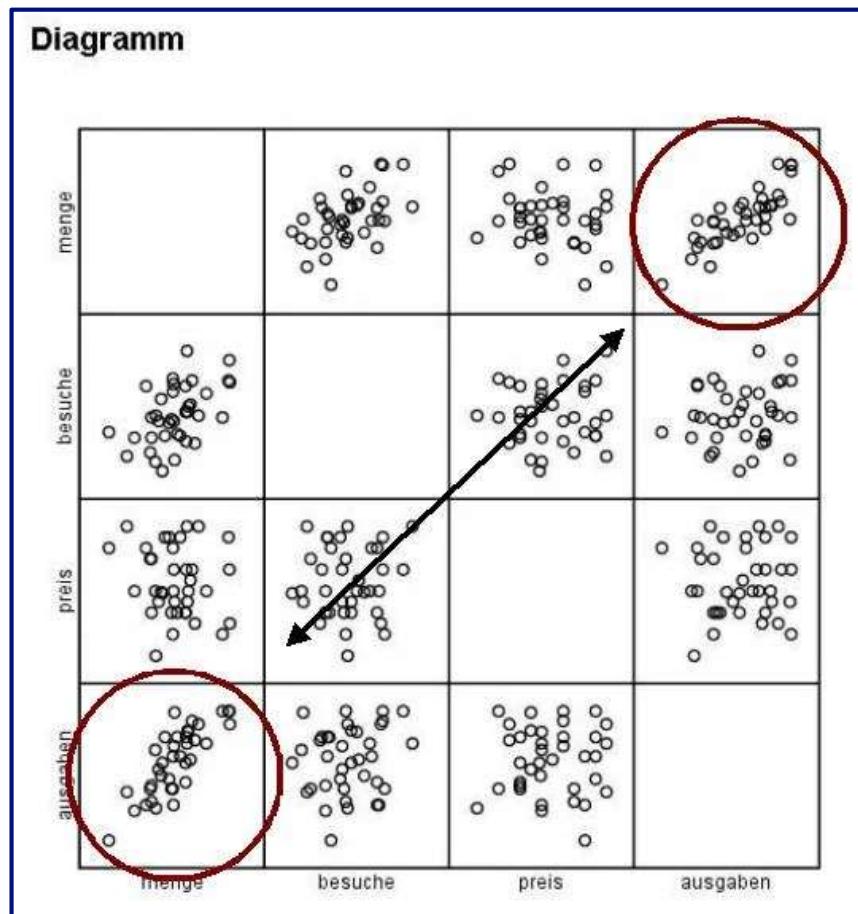


	A	B	C	D	E	F	...	XX
1								
2								
3								
4								
5								

	A	B	C	D	E	F	...	XX
1			X			X		
2			X			X		
3								
4								
5								

The challenge is to reduce without loosing important information. Usually, this is not done by simply deleting features, but by transforming the features in a way that the correlations (multiple presence of information) are eliminated.

Multicollinearity (I)



Multicollinearity describes high correlation between variables.

Some models are susceptible to multicollinearity (high correlations between predictors). Linear models, neural networks and other models can have poor performance in these situations or may generate unstable solutions. Other models, such as decision trees, might be resistant to highly correlated predictors, but multicollinearity may negatively affect interpretability of the model.

Multicollinearity (II)

If there is a need to minimize the effect of multicollinearity, there are a few options. First, principal component analysis can be used to reduce the number of dimensions in a way that removes correlations.

Alternatively, we can identify and remove predictors that contribute the most to the correlations.

We can compute the correlation matrix of the predictors and use an algorithm to remove a subset of the problematic predictors such that all of the pairwise correlations are below a threshold:

While no correlations are above the threshold

Find the pair of predictors with the largest absolute correlation;

For both predictors, compute the average correlation between each predictor and all of the other variables;

Flag the variable with the largest mean correlation for removal;

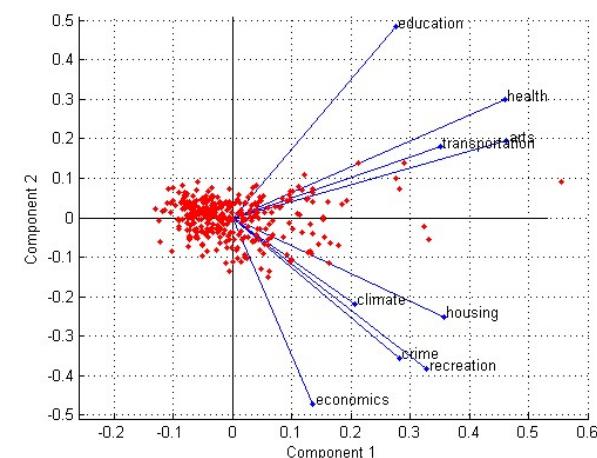
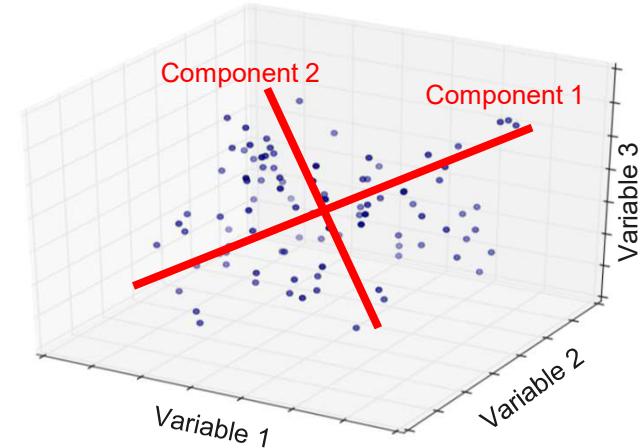
Remove this row and column from the correlation matrix;

Principal Component Analysis (I)

Principal Component Analysis (PCA) is a method of extracting important variables (named components) from a large set of variables available in a data set. It extracts low dimensional set of features from a high dimensional data set with a motive to capture as much information as possible.

Assume you have n observations of p different variables. Define X to be a $n \times p$ matrix where the i^{th} column contains the observations of the i^{th} variable. Each row can be represented as a point in a p -dimensional space. Therefore, X contains n points in a p -dimensional space.

PCA projects p -dimensional data into a q -dimensional sub-space $q \leq p$ in a way that minimizes the residual sum of squares (RSS) of the projection.



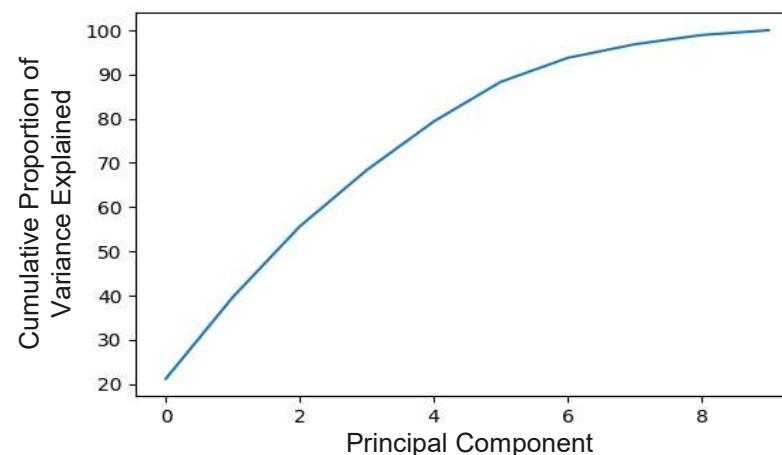
Principal Component Analysis (II)

Every principal component is a linear combination of the original predictor variables X. The principal component i can be written as

$$z_i = \Phi_{1i}x_1 + \Phi_{2i}x_2 + \Phi_{3i}x_3 + \dots + \Phi_{pi}x_p$$

The first principal component is created in a way that it captures the maximum variance in the data set. It determines the direction of highest variability in the data. The larger the variability captured in first component, the larger the information captured by component.

The other components cover step-wise descending the rest of the variance. One can select the k first components until an acceptable share of variance is covered.



Principal Component Analysis (III)

Mathematically, the goal is to minimize the sum of squared distances from the points to their projections. This is equivalent to maximizing the covariance matrix of the projected data.

Assume Σ to be the covariance matrix associated with X . Since Σ is a non-negative definite matrix, it has an eigendecomposition $\Sigma = C \Lambda C^{-1}$, where $\Lambda = \text{diag}(\lambda_1, \dots, \lambda_p)$ is a diagonal matrix of (non-negative) eigenvalues in decreasing order, and C is a matrix where its columns are formed by the eigenvectors of Σ .

We want the first principal component z_1 to be a linear combination of the columns of X , $z_1 = \Phi X$. In addition, we want z_1 to have the highest possible variance $\text{Var}(z_1) = \Phi^T \Sigma \Phi$. It turns out that a will be given by the column eigenvector corresponding with the largest eigenvalue of Σ .

The second principal component z_2 is calculated using the column eigenvector corresponding with the second largest eigenvalue of Σ and so on.

If we pick the first q principal components, we have projected our p -dimensional data into a q -dimensional sub-space with uncorrelated principal components.

Principal Component Analysis (IV)

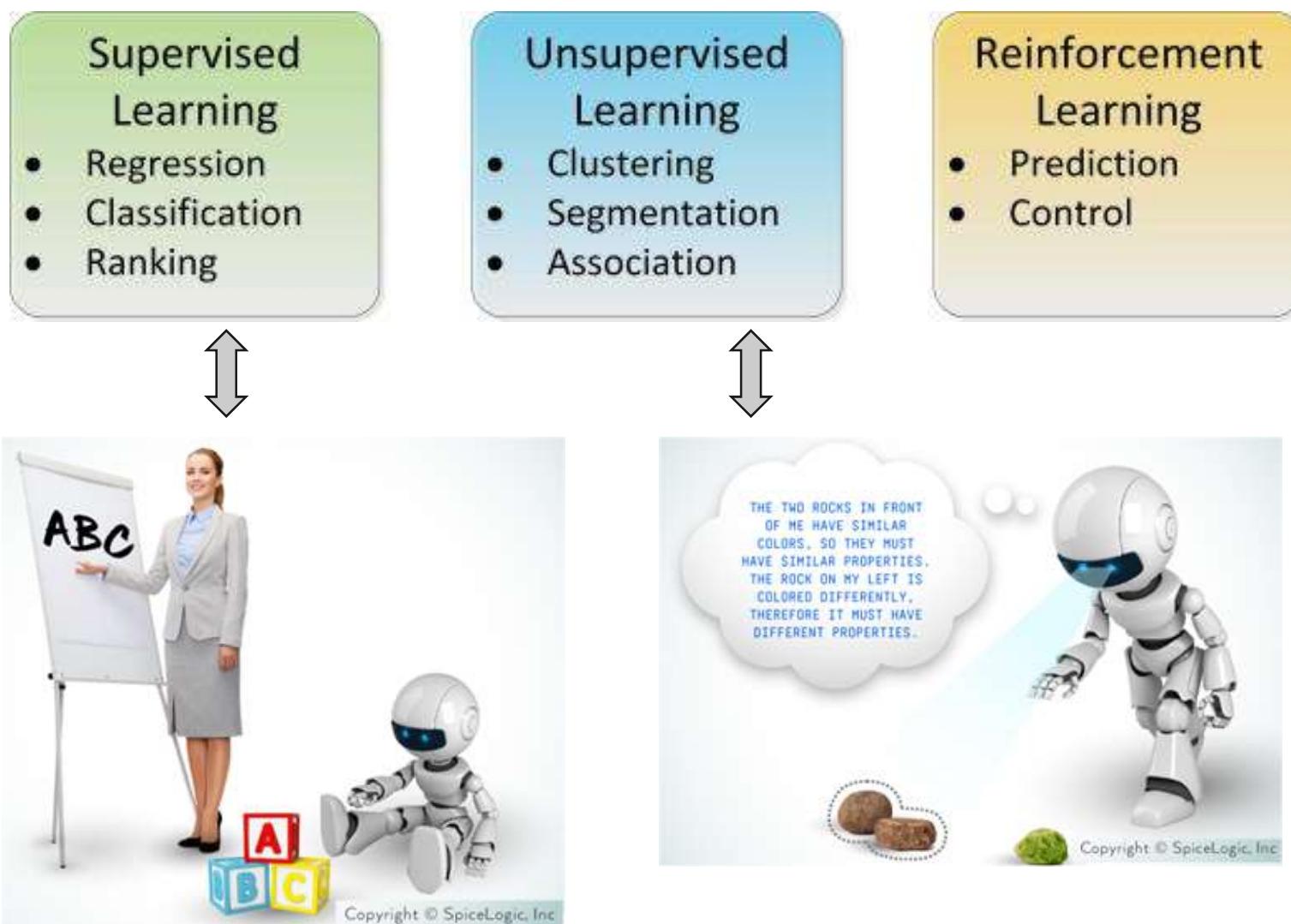
Some general advice:

- PCA is not scale invariant, so it is highly recommended to normalize all the p variables before applying PCA.
- How many principal components to retain will depend on the specific application.
- Two or three principal components can be used for visualization purposes.
- PCA is an orthogonal linear transformation with some flaws. First, there may be no single linear projection that gives a good view of the data: in such a case, all linear projection methods will fail. Second, PCA may not preserve local proximities of points

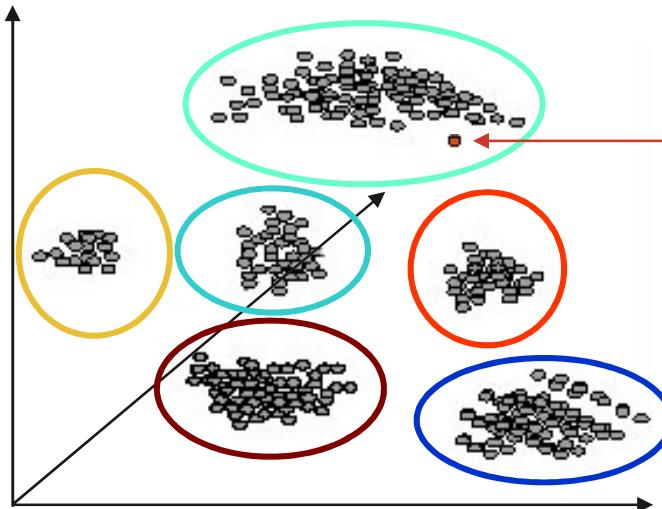
Source: <http://tgmstat.wordpress.com/2013/11/21/introduction-to-principal-component-analysis-pca>

2 Methods, Algorithms, and Applications

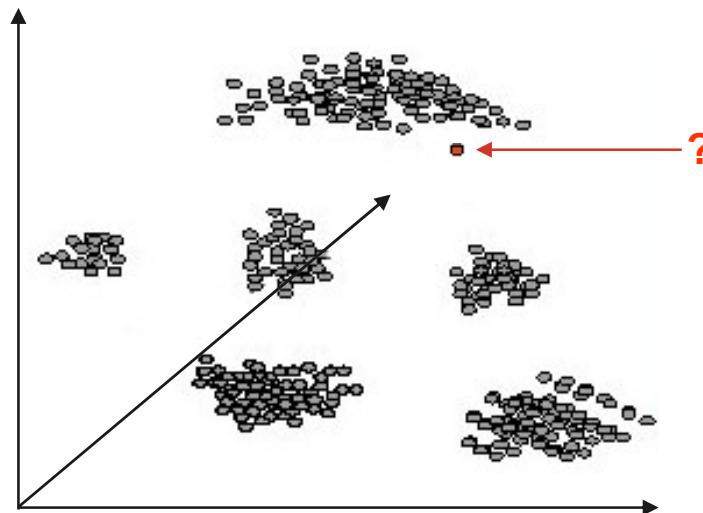
Categories in Machine Learning



Supervised Learning

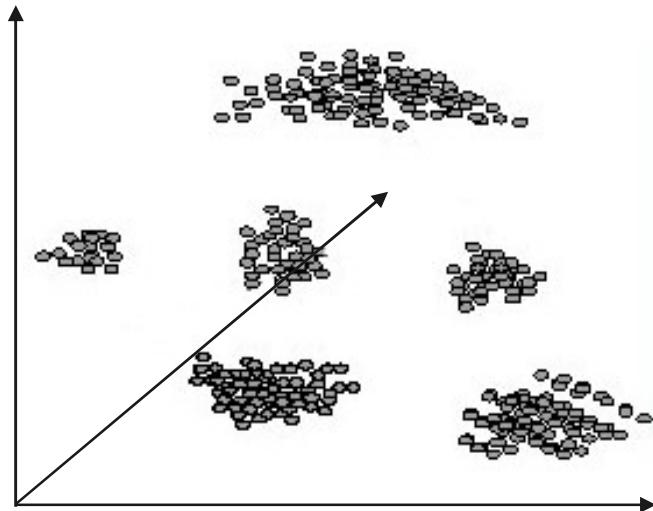


Classification:
„To which segment
does the new
object belong to?“

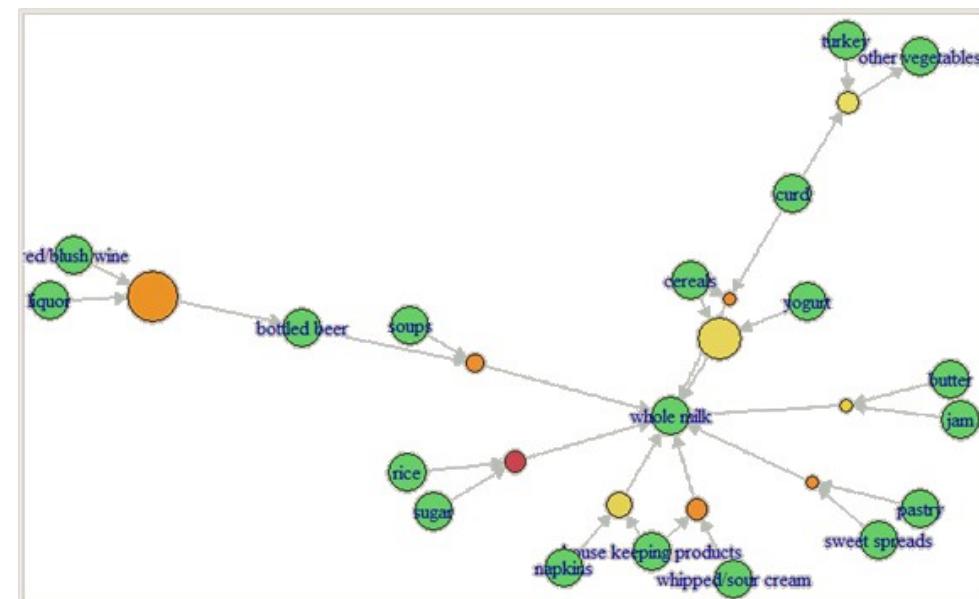


Regression:
"What is the
target value of
the object."

Unsupervised Learning



Segmentation (Clustering):
"In how many and which segments can the data be partitioned?"



Association:
"Detect the strength and the direction of relationships in my data."

Supervised and Unsupervised Learning

Supervised Learning:

- Goal: predict a **target** or **outcome** variable
- Training data where target value is known
- Score to test data where value is not known

DEP_TIME	DEST	DISTANCE	FL_DATE	ORIGIN	Weather	DAY_WEEK	DAY_OF_MONTH	FlightStatus
1455	JFK	184	01.01.2004	BWI	0	4	1	ontime
1640	JFK	213	01.01.2004	DCA	0	4	1	ontime
1245	LGA	229	01.01.2004	IAD	0	4	1	ontime
1709	LGA	229	01.01.2004	IAD	0	4	1	ontime
1035	LGA	229	01.01.2004	IAD	0	4	1	ontime
839	JFK	228	01.01.2004	IAD	0	4	1	ontime
1243	JFK	228	01.01.2004	IAD	0	4	1	ontime
1644	JFK	228	01.01.2004	IAD	0	4	1	ontime
1710	JFK	228	01.01.2004	IAD	0	4	1	ontime
2129	JFK	228	01.01.2004	IAD	0	4	1	ontime
2114	LGA	229	01.01.2004	IAD	0	4	1	ontime
1458	JFK	213	01.01.2004	DCA	0	4	1	ontime
932	LGA	214	01.01.2004	DCA	0	4	1	ontime
1228	LGA	214	01.01.2004	DCA	0	4	1	ontime

HousePrice	HouseSize	HouseAge	LotSize	RiverSide	CrimeRate	Industrial
244	6.064	59.1	0	1	.13587	10.59
212	6.176	72.5	0	0	.13158	10.01
222	6.358	52.9	30	0	.1029	4.93
134	6.103	85.1	0	0	705.042	18.1
238	5.877	79.2	0	0	180.028	19.58
174	5.594	36.8	12.5	0	.13554	6.07
222	6.316	38.1	0	0	.05083	5.19
262	6.718	17.5	22	0	.19073	5.86
140	6.174	93.6	0	0	.2909	21.89
190	5.985	45.4	0	0	.05497	5.19
178	7.393	99.3	0	0	824.809	18.1

Unsupervised Learning:

- Goal: detect patterns
- Grouping of cases based on similarities in input values
- There is no target (outcome) variable

Use Cases Quiz

Online Shop for Electronic Articles:

- Management of Customer Termination
- Returns Management
- Identify similar Regions for Advertising
- Sales Forecasts for Supply and Inventory Management
- Automatically assign new Products into existing Product Categories
- Grouping of similar Customers
- Voucher Management
- Analysis of frequently sold Product Combinations
- Fraud Detection
- Structuring the Competitors
- Analysis of Sales Sequences
- ...

Reinforcement Learning

The solution to many of the problems in our lives cannot be automated. This is not because current computers are too slow, but simply because it is too difficult for humans to determine what the program should do.

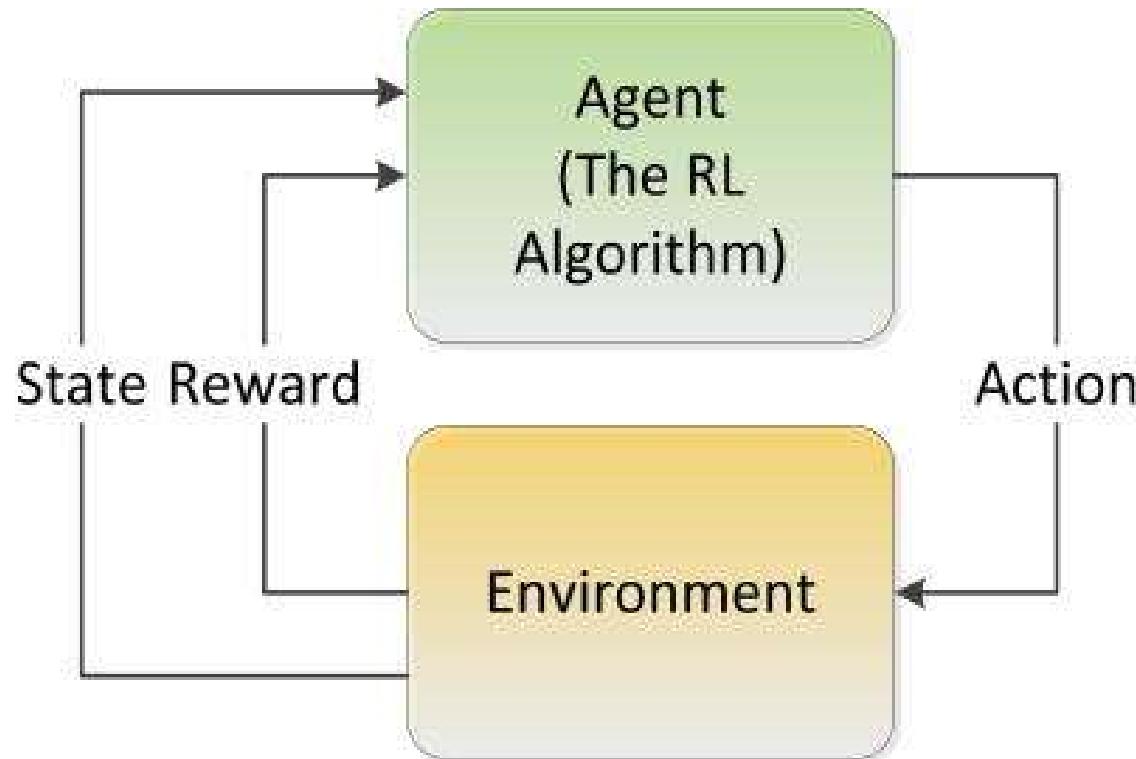
Supervised learning is a general method for training an approximator. However, supervised learning requires sample input-output pairs from the domain to be learned.

For example, we might not know the best way to program a computer to recognize an infrared picture of a tank, but we do have a large collection of infrared pictures, and we do know whether each picture contains a tank or not. Supervised learning could look at all the examples with answers, and learn how to recognize tanks in general.

Unfortunately, there are many situations where we don't know the correct answers that supervised learning requires. For example, in a self-driving car, the question would be the set of all sensor readings at a given time, and the answer would be how the controls should react during the next millisecond.

For these cases there exist a different approach known as reinforcement learning.

Reinforcement Learning



The agent learns how to achieve a given goal by trial-and-error interactions with its environment by maximizing a reward.

AlphaGo

Go is one of the hardest games in the world for AI because of the huge number of different game scenarios and moves. The number of potential legal board positions is greater than the number of atoms in the universe.

The core of AlphaGo is a deep neural network. It was initially trained to learn playing by using a database of around 30 million recorded historical moves. After the training, the system was cloned and it was trained further playing large numbers of games against other instances of itself, using reinforcement learning to improve its play. During this training AlphaGo learned new strategies which were never played by humans.

A newer version named AlphaGo Zero skips the step of being trained and learns to play simply by playing games against itself, starting from completely random play.



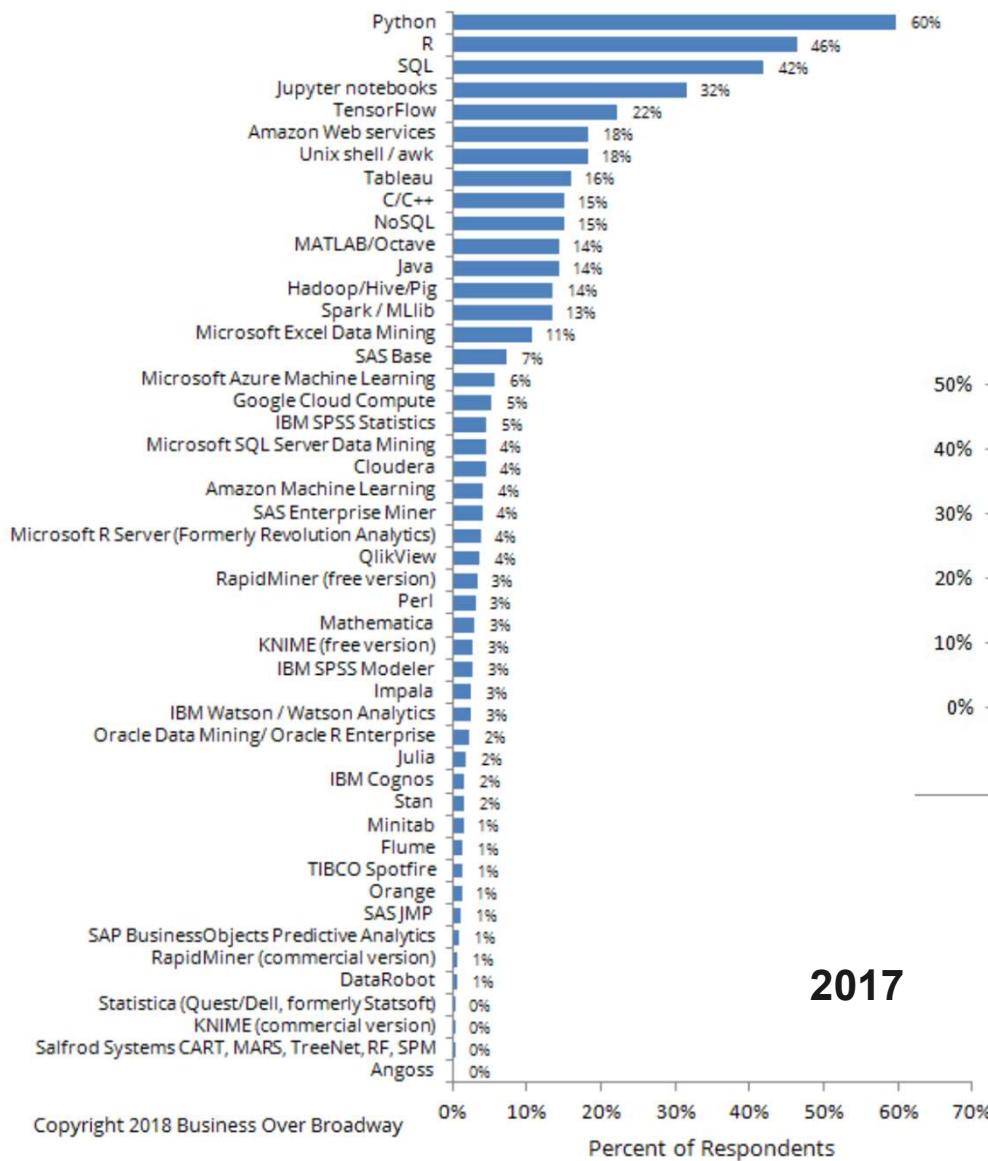
An artificial intelligence called Libratus has beaten four of the world's best poker players in a grueling 20-day tournament in January 2017.

Poker is more difficult because it's a game with imperfect information. With chess and Go, each player can see the entire board, but with poker, players don't get to see each other's hands. Furthermore, the AI is required to bluff and correctly interpret misleading information in order to win.

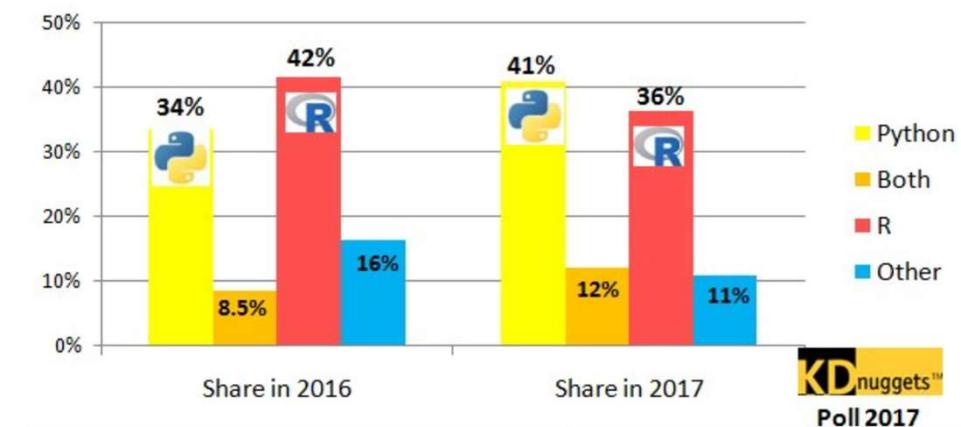
"We didn't tell Libratus how to play poker. We gave it the rules of poker and said 'learn on your own'." The AI started playing randomly but over the course of playing trillions of hands was able to refine its approach and arrive at a winning strategy.



Tools used in Data Science

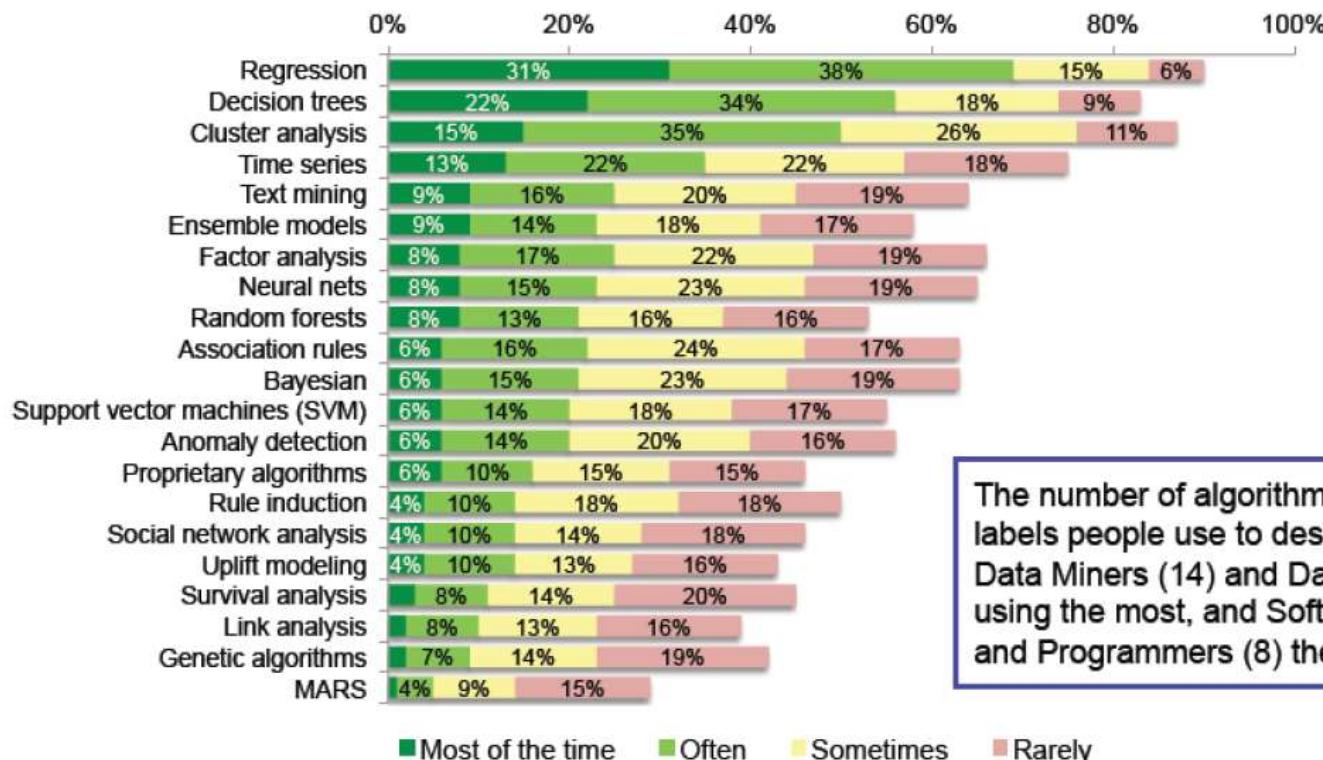


2017



Methods used in Data Science

- Regression, decision trees, and cluster analysis continue to form a triad of core algorithms for most data miners. This has been consistent since the first Data Miner Survey in 2007.
- The average respondent reports typically using 12 algorithms. People with more years of experience use more algorithms, and consultants use more algorithms (13) than people working in other settings (11).



The number of algorithms used varies by the labels people use to describe themselves, with Data Miners (14) and Data Scientists (14) using the most, and Software Developers (9) and Programmers (8) the fewest.

Question: What algorithms / analytic methods do you TYPICALLY use? (Select all that apply)

Classification Methods

k-Nearest Neighbors

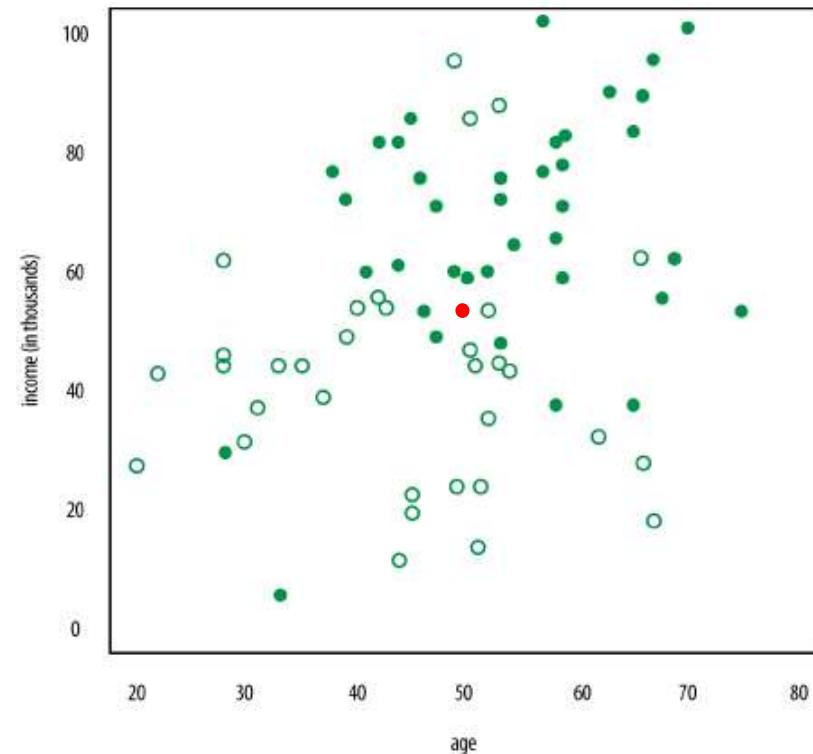
Introductory Example

Credit-Scoring is a typical example for a classification problem. A bank wants to determine the creditworthiness of a customer.

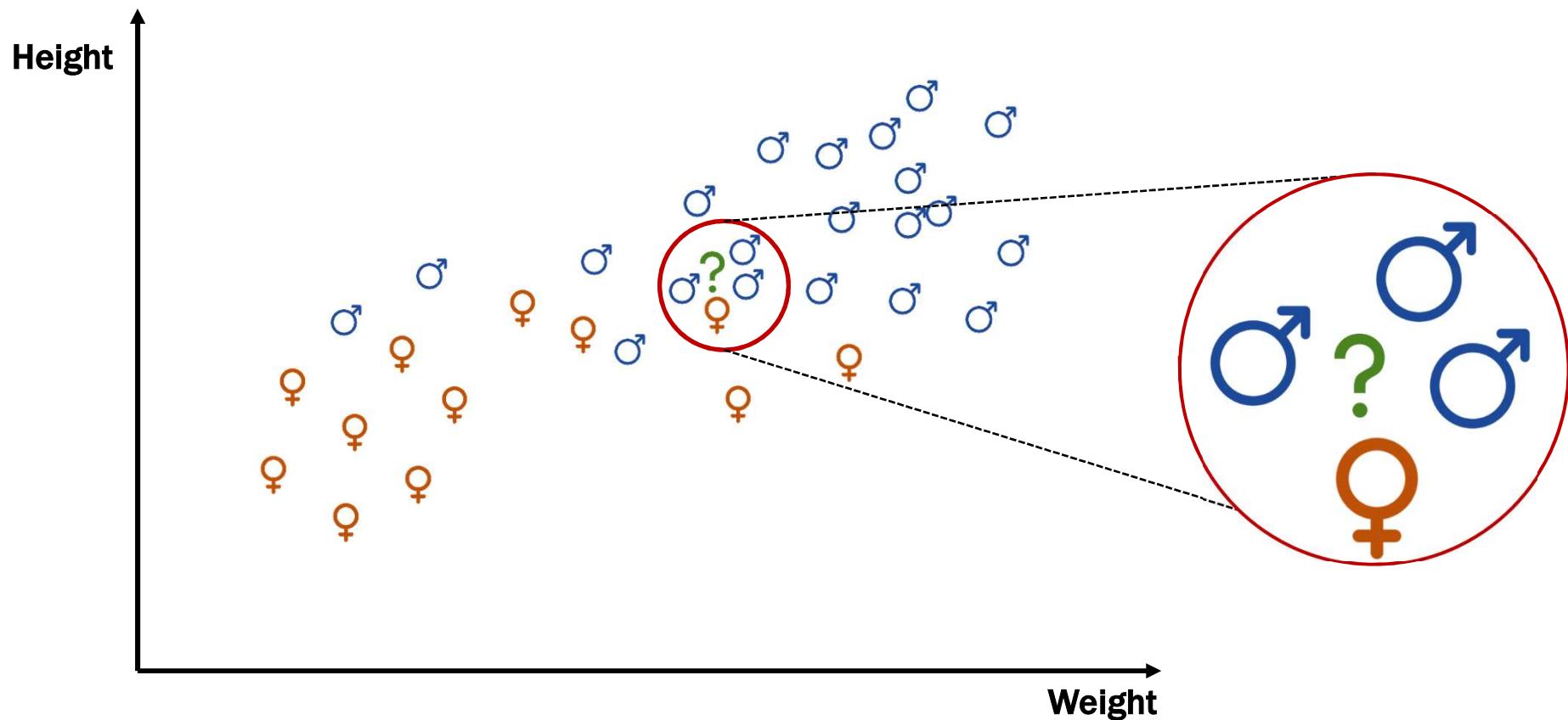
Assume you have the age, income, and a creditworthiness category of "yes" or "no" for a bunch of people and you want to use the age and income to predict the creditworthiness for a new person.

You can plot people as points on the plane and label people with an empty circle if they have low credit ratings.

What if a new guy comes in who is 49 years old and who makes 53,000 Euro? What is his likely credit rating label?



k-Nearest Neighbors (I)



k-Nearest Neighbors (II)

k-Nearest Neighbors (k-NN) is an algorithm that can be used when you have a bunch of objects that have been classified or labeled in some way, and other similar objects that have not gotten classified or labeled yet, and you want a way to automatically label them.

The intuition behind k-NN is to consider the most similar other items defined in terms of their attributes, look at their labels, and give the unassigned item the majority vote. If there's a tie, you randomly select among the labels that have tied for first.

Procedure of k-NN:

1. Determine parameter k (= number of nearest neighbors)
2. Calculate the distances between the new object and all known labeled objects.
3. Choose the k objects from all known labeled objects having the smallest distance to the new object as nearest neighbors.
4. Count the frequencies of the classes of the nearest neighbors.
5. Assign the new object to the most frequent class.

Measuring Similarity

The purpose of a measure of similarity is to compare two vectors of data, and compute a single number which evaluates their similarity.

Usually the similarity $s(x,y)$ of two objects x and y is measured using the inverse of some distance measure $d(x,y)$:

$$s(x,y) = 1 - d(x,y)$$

Euclidean distance is most often used for numerical data vectors:

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

with n as size of the vectors.

The variables must be scaled to avoid unwanted weighting resulting from different level of measurement.

Unnormalized vs. Normalized

Unnormalized:

Objects \ Features	Age	Income	Rent	Years in Job
Objects				
Peter	47	4000	1400	22
Thomas	33	2700	1050	15

$$d = \sqrt{(47 - 33)^2 + (4000 - 2700)^2 + (1400 - 1050)^2 + (22 - 15)^2} = 1346.38$$

Normalized:

Objects \ Features	Age	Income	Rent	Years in Job
Objects				
Peter	0.32	0.21	0.85	0.53
Thomas	0.26	0.13	0.54	0.34

$$d = \sqrt{(0.32 - 0.26)^2 + (0.21 - 0.13)^2 + (0.85 - 0.54)^2 + (0.53 - 0.34)^2} = 0.3771$$

Example (I)

Data:

Customer	Age	Monthly Income	Monthly Costs	Creditworthy
A	24	1800	1400	yes
B	26	1700	1300	no
C	30	2800	2200	yes
D	33	4200	3800	yes
E	35	2300	1800	no
F	39	3000	2800	no
G	41	2400	1600	yes
H	47	5000	2500	yes
I	53	4700	3200	no
J	59	2100	1700	yes

New Customer:

X	32	2900	1800	?
---	----	------	------	---

Example (II)

1. **k = 5 is chosen**

2. Calculate the distances (Normalization and then Euclidian Distance)

Customer	Age	Monthly Income	Monthly Costs	Creditworthy	Distance
A	0.0000	0.0303	0.0400	yes	0.4347
B	0.0571	0.0000	0.0000	no	0.4490
C	0.1714	0.3333	0.3600	yes	0.1726
D	0.2571	0.7576	1.0000	yes	0.8922
E	0.3143	0.1818	0.2000	no	0.2010
F	0.4286	0.3939	0.6000	no	0.4482
G	0.4857	0.2121	0.1200	yes	0.3090
H	0.6571	1.0000	0.4800	yes	0.8167
I	0.8286	0.9091	0.7600	no	0.9855
J	1.0000	0.1212	0.1600	yes	0.8096
X	0.2286	0.3636	0.2000	?	

Example (III)

3. Choose the k nearest neighbors

Customer	Age	Monthly Income	Monthly Costs	Creditworthy	Distance
A	0.0000	0.0303	0.0400	yes	0.4347
C	0.1714	0.3333	0.3600	yes	0.1726
E	0.3143	0.1818	0.2000	no	0.2010
F	0.4286	0.3939	0.6000	no	0.4482
G	0.4857	0.2121	0.1200	yes	0.3090
X	0.2286	0.3636	0.2000	?	

4. Count the numbers of class members

3 x yes; 2 x no

5. Assign object to most frequent class

Customer is creditworthy!

Calculating Accuracies

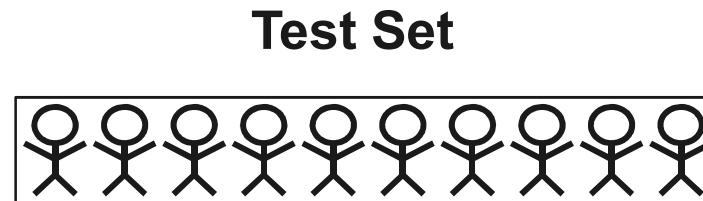


+ + - + - - + - + -

+ - - + - + + - + -

The training set is used to calculate the classifier model.

The accuracy of the training set indicates the goodness of fit of the model.



- + - - + + + - + - real

- + + - + - + - + + predicted

The test set is used to evaluate the classifier model.

The accuracy of the test set indicates the robustness/predictability of the model.

$$\text{Accuracy} = \frac{\#\text{Match}}{\#\text{All}}$$

Determining Parameter k

- 1. Split the original labeled dataset into training and test data.**
- 2. Pick an evaluation metric. Misclassification rate or accuracy are good ones.**
- 3. Run k-NN a few times, changing k and checking the evaluation measure.**
- 4. Optimize k by picking the one with the best evaluation measure.**

| k | Accuracy |
|----|----------|
| 1 | 0.720 |
| 2 | 0.685 |
| 3 | 0.740 |
| 4 | 0.745 |
| 5 | 0.770 |
| 6 | 0.740 |
| 7 | 0.750 |
| 8 | 0.750 |
| 9 | 0.765 |
| 10 | 0.760 |

Evaluating Classifier Performance

Classification

- Let $X = \langle x_1, \dots, x_m \rangle$ denote the **feature space**, consisting of m vectors.
- Let $C = \{c_1, \dots, c_k\}$ denote the **class space**, which represents k possible classes.
- A **classifier** K is a function $K : X \rightarrow C$, that maps an object $x \in X$ to a class $c \in C$.

Procedure of classification:

1. Construction of a classifier
 - a) Starting with a training set $Tr \subset X \times C$, where for each object its related class is known (i.e. the value of the target variable is known), a classifier is build.
 - b) The quality of the classifier is determined using a test set of objects, where also the class of each object is known, and that is independent from the training set.

2. Application of the classifier

Use the classifier to determine the class for each object $x_1, x_2, \dots \in X$ from a set of objects, where the related class for each object is unknown.

Example: Credit-Scoring

| Customer ID | Martial status | Income | Occupational group | Creditworthiness |
|-------------|----------------|--------|--------------------|------------------|
| 1 | single | 1500 | employed | yes |
| 2 | single | 5000 | self-employed | yes |
| 3 | divorced | 3200 | employed | yes |
| 4 | married | 1700 | employed | no |

The feature space consists of three attributes: $X = X_1 \times X_2 \times X_3$ where
 $X_1 = \{ \text{single, married, divorced} \}$, $X_2 = \mathbb{R}_+$, $X_3 = \{ \text{employed, self-employed} \}$
and two classes: $C = \{ \text{yes, no} \}$

Questions:

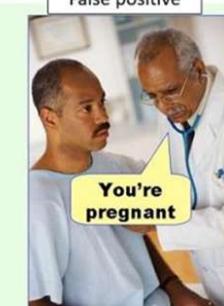
- How to handle non-numerical attributes?
- What is the objective? Minimizing the misclassification rate or the misclassification costs?

Evaluating the quality of Classification (I)

True positives (TP), true negatives (TN), false positives (FP), and false negatives (FN), are the four different possible outcomes of a single prediction for a two-class case. A false positive is when the outcome is incorrectly classified as "yes", when it is in fact "no". A false negative is when the outcome is incorrectly classified as negative, when it is in fact positive. True positives and true negatives are obviously correct classifications.

Confusion Matrix:

| | | Actual Class | |
|-----------------|-----|--------------|----|
| | | Yes | No |
| Predicted Class | Yes | TP | FP |
| | No | FN | TN |



False positive
You're pregnant



False negative
You're not pregnant

Evaluating the quality of Classification (II)

Test metrics are used to assess how accurately the model predicts the known values:

$$\text{accuracy} = \frac{TN + TP}{TP + FP + TN + FN}$$

$$\text{misclassification rate} = \frac{FN + FP}{TP + FP + TN + FN} = 1 - \text{accuracy}$$

Most classification algorithms pursue to minimize the misclassification rate. They implicitly assume that all misclassification errors cost equally. In many real-world applications, this assumption is not true. Cost-sensitive learning takes costs, such as the misclassification cost, into consideration. Using costs, the error rate can be calculated via:

$$\text{cost sensitive misclassification rate} = \frac{FN \cdot C_{FN} + FP \cdot C_{FP}}{TP + FP + TN + FN}$$

Evaluating the quality of Classification (III)

Misclassification rate and accuracy can be misleading, for example in the case of imbalanced samples. Extreme case:

| | yes | no | |
|-----|------|----|--|
| yes | 1000 | 3 | |
| no | 45 | 2 | |

→ $\frac{1000 + 2}{1000 + 45 + 3 + 2} = 0.954$

For problems like, this additional measures are required to evaluate a classifier.

Sensitivity (true positive rate, **recall**) measures the proportion of positives that are correctly identified as such. **Specificity** (true negative rate) measures the proportion of negatives that are correctly identified as such:

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

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

Using both measures, we can compute the **Balanced Accuracy**:

$$balanced\ accuracy = \frac{sensitivity + specificity}{2}$$

Problem of Imbalancing and Accuracy

Assume the following case: A credit card company wants to create a fraud detection system to include it into their transactional systems. The outcomes should be "Accept" (Y) and "Reject" (N). Because fraud rarely occurs, the data set consists of 320 observations for Y and 139 for N. They are partitioned into training and test set. Finally, the model is trained and tested.

Because of the majority of the Y class, the training process concentrates on these cases because their correct classification promises the highest accuracy.

The results of the test of the model is consequently:

| | N | Y |
|---|----|----|
| N | 21 | 6 |
| Y | 32 | 94 |

Thus, the model is blind for the N cases. But these are the ones of primary interest for the company.

Evaluating the quality of Classification (IV)

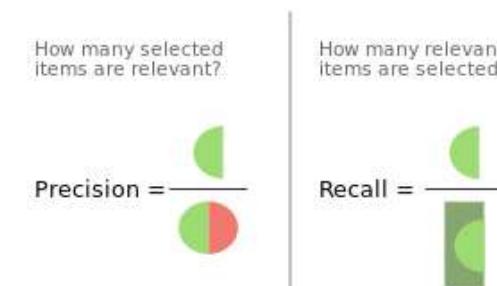
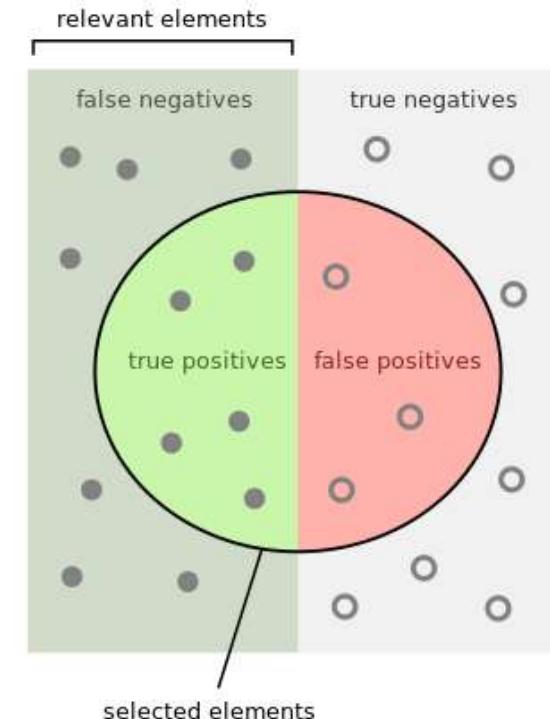
Another measure for the quality of classification that can also be used in the case of imbalance is the F1 score. The F1 score considers both the precision and the recall (sensitivity) to compute the quality.

Precision is the number of correct positive results divided by the number of all positive results returned by the classifier, and **Recall** is the number of correct positive results divided by the number of all relevant positives:

$$\text{precision} = \frac{TP}{TP + FP} \quad \text{recall} = \frac{TP}{TP + FN}$$

The F1 score is the harmonic mean of the precision and recall, where an F1 score reaches its best value at 1 (perfect precision and recall) and worst at 0

$$F1 = 2 * \frac{\text{precision} * \text{recall}}{\text{precision} + \text{recall}}$$



Source: Wikipedia

ROC and AUC (I)

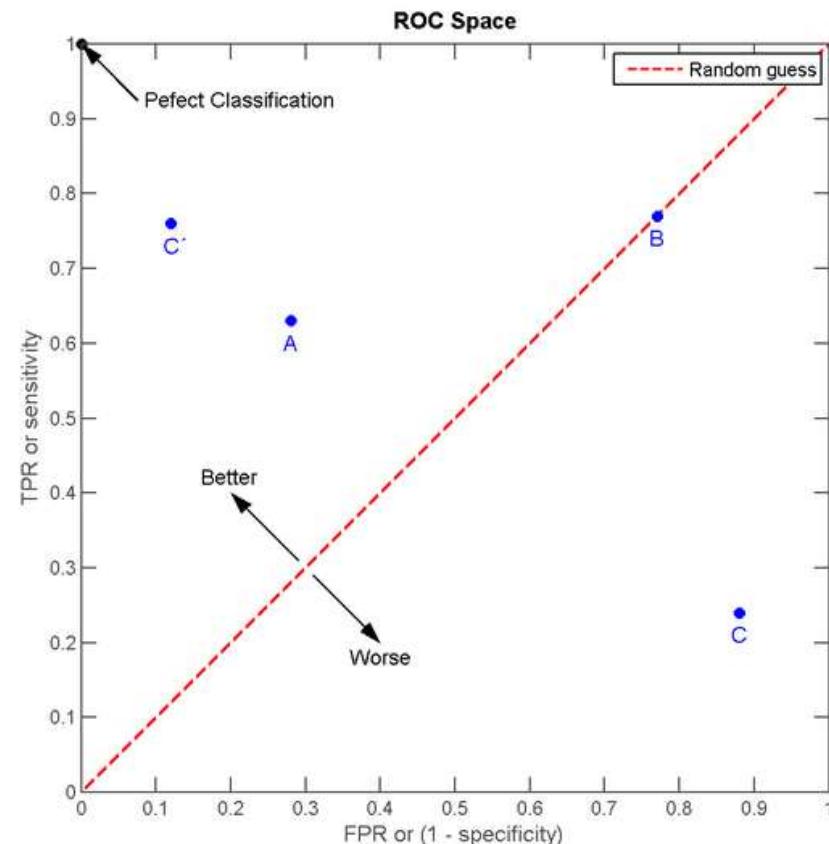
A **ROC plot** is a two-dimensional graph in which the true positive rate (TPR) is plotted on the Y axis and the false positive rate (FPR) is plotted on the X axis.

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

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

The TPR is the percentage of correctly positive classified observations and the FPR the percentage of false positive classified observations.

A ROC plot depicts relative tradeoffs between benefits (true positives) and costs (false positives). The figure shows an ROC plot with five classifiers labeled A through E.



Source: Wikipedia: Receiver operating characteristic

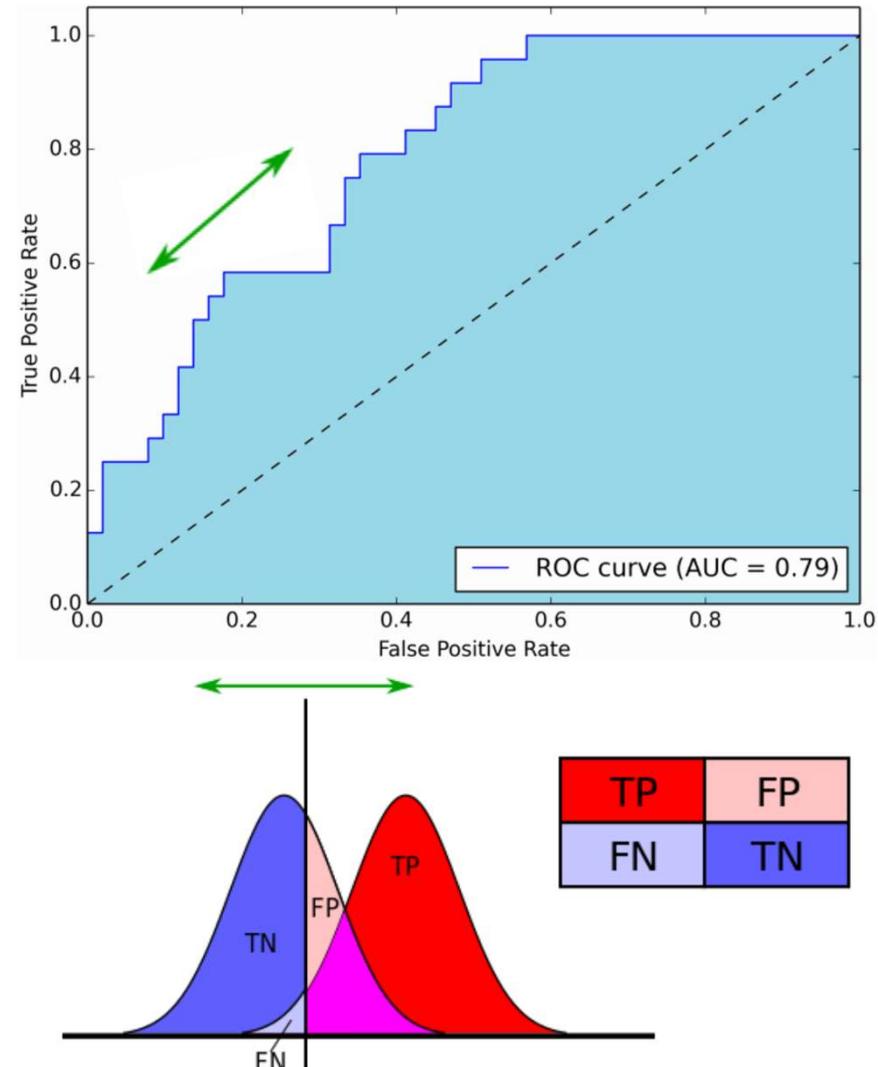
ROC and AUC (II)

If the classifier outputs labels then every model in the ROC plot is represented by a point.

If the classifier outputs probabilities then one must decide about the threshold value. Every probability smaller than this value is assigned to the negative class and the others to the positive class.

Simulating thresholds from 0 to 1 and calculating TPR and FPR for every threshold value results in a **ROC curve**.

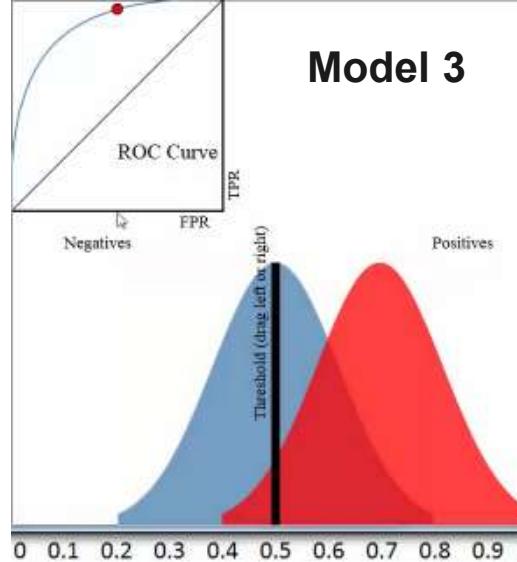
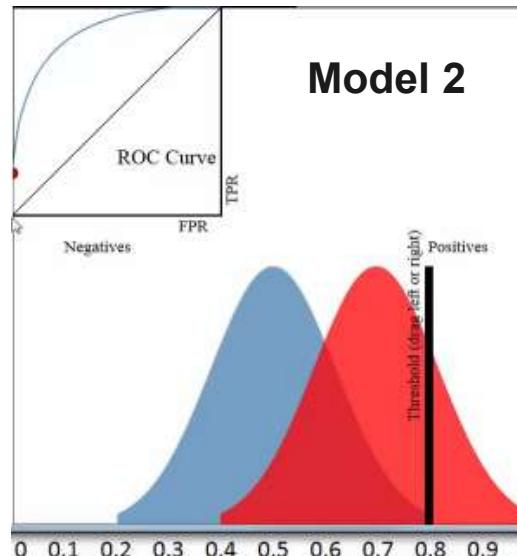
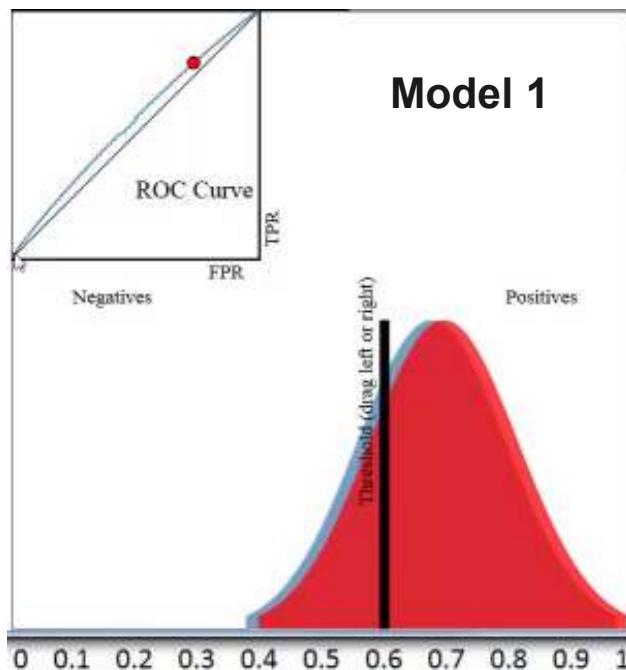
The blue area corresponds to the Area Under the curve (AUC) of the ROC. The AUC of a classifier is equal to the probability that the classifier will rank a randomly chosen positive example higher than a randomly chosen negative example.



Source: Wikipedia: Receiver operating characteristic

ROC and AUC (III)

ROC and AUC can be used in classification analysis in order to determine which of the used models predicts the classes best.



$$\text{True Positive Rate} = 50 / 250 = 0.2$$

$$\text{False Positive Rate} = 0 / 250 = 0$$

$$\text{Plot on ROC Curve: } (x, y) = (0, 0.2)$$

$$\text{True Positive Rate} = 235 / 250 = 0.94$$

$$\text{False Positive Rate} = 125 / 250 = 0.5$$

$$\text{Plot on ROC Curve: } (x, y) = (0.5, 0.94)$$

How to select a Model (I)

A good model must have two properties:

- (1) A high test quality value reflecting a high application quality
- (2) A small difference between training and test quality to avoid overfitting

Often there is a trade-off between the two.

A measure of model quality must incorporate both. A way to do this is to use the test quality as initial point and to adjust it by the ration between test and training quality:

$$\text{Modelquality} = \text{Testquality} * s * \frac{\text{Testquality}}{\text{Trainingquality}}$$

where parameter s can be set as sensitivity of the importance of the ratio.

Limitation: The method does not work if Trainingquality < Testquality.

How to select a Model (II)

Example:

Model 1: Trainingquality=0.95; Testquality=0.89

Model 2: Trainingquality=0.91; Testquality=0.88

$$Modelquality_1 = 0.89 * 1 * \frac{0.89}{0.95} = 0.83$$

s = 1:

$$Modelquality_2 = 0.88 * 1 * \frac{0.88}{0.91} = 0.85$$

$$Modelquality_1 = 0.89 * 0.5 * \frac{0.89}{0.95} = 0.42$$

s = 0.5:

$$Modelquality_2 = 0.88 * 0.5 * \frac{0.88}{0.91} = 0.35$$

Naive Bayes Classifier

Introductory Example

| | A | B | C | D | | BB | BC | BD |
|----|---------|---------|------------|--------|-----|-----------|----------|------|
| 1 | Message | WF_make | WF_address | WF_all | | CF_dollar | CF_pound | Spam |
| 2 | 21 | 0 | 0 | 0 | | 0 | 0 | no |
| 3 | 22 | 0 | 0 | 1,69 | | 0 | 0 | no |
| 4 | 23 | 0 | 0 | 0 | | 0 | 0 | no |
| 5 | 24 | 0 | 0 | 0 | | 0 | 0 | yes |
| 6 | 25 | 0,74 | 0 | 0 | | 0 | 0 | no |
| 7 | 26 | 0 | 0 | 0 | ... | 0 | 0,041 | no |
| 8 | 27 | 0 | 0 | 0 | | 0 | 0 | yes |
| 9 | 28 | 0 | 0 | 0,32 | | 0 | 0,055 | no |
| 10 | 29 | 0,22 | 0 | 0 | | 0 | 0 | yes |
| 11 | 30 | 0 | 0 | 0 | | 0 | 0 | no |
| 12 | 31 | 0 | 0 | 0 | | 0 | 0 | no |
| 13 | 32 | 0 | 0 | 0,32 | | 0 | 0 | no |
| 14 | 33 | 0 | 0 | 0 | | 0 | 0,07 | yes |
| 15 | 34 | 0 | 0 | 0,94 | | 0,388 | 0,097 | yes |
| 16 | 35 | 0,09 | 0,49 | 0,59 | | 0,979 | 0 | yes |
| 17 | 36 | 0 | 0 | 1,2 | | 0 | 0 | no |
| 18 | 37 | 0 | 0 | 0 | | 0 | 0 | yes |
| 19 | 38 | 0 | 0 | 0 | | 0 | 0 | no |
| 20 | 39 | 0 | 0 | 0 | | 0 | 0 | no |
| 21 | 40 | 0 | 0 | 0 | | 0 | 0 | no |

Spam Classification:

48 continuous real [0,100] attributes of type word frequency (WF_...)

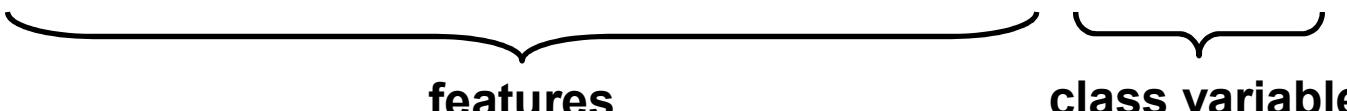
6 continuous real [0,100] attributes of type character frequency (CF_...)

1 nominal {0,1} class attribute of type spam

| | |
|-----------------------|------|
| Number of Instances: | 4601 |
| Number of Attributes: | 57 |

Introductory Example

| Day | outlook | temperature | humidity | wind | play tennis |
|-----|----------|-------------|----------|--------|-------------|
| 1 | sunny | hot | high | light | no |
| 2 | sunny | hot | high | strong | no |
| 3 | overcast | hot | high | light | yes |
| 4 | rainy | mild | high | light | yes |
| 5 | rainy | cold | normal | light | yes |
| 6 | rainy | cold | normal | strong | no |
| 7 | overcast | cold | normal | strong | yes |
| 8 | sunny | mild | high | light | no |
| 9 | sunny | cold | normal | light | yes |
| 10 | rainy | mild | normal | light | yes |
| 11 | sunny | mild | normal | strong | yes |
| 12 | overcast | mild | high | strong | yes |
| 13 | overcast | hot | normal | light | yes |
| 14 | rainy | mild | high | strong | no |



features

class variable

Naive Bayes Classifier (I)

The Naive Bayes Classifier is a classification technique based on Bayes' Theorem with an assumption of independence among predictors. It assumes that the presence of a particular feature in a class is unrelated to the presence of any other feature.

Bayes theorem provides a way of calculating posterior probability via

$$P(B_i|A) = \frac{P(B_i) \cdot P(A|B_i)}{P(A)}$$

$P(B_i|A)$ is the posterior probability of class i given predictor x.

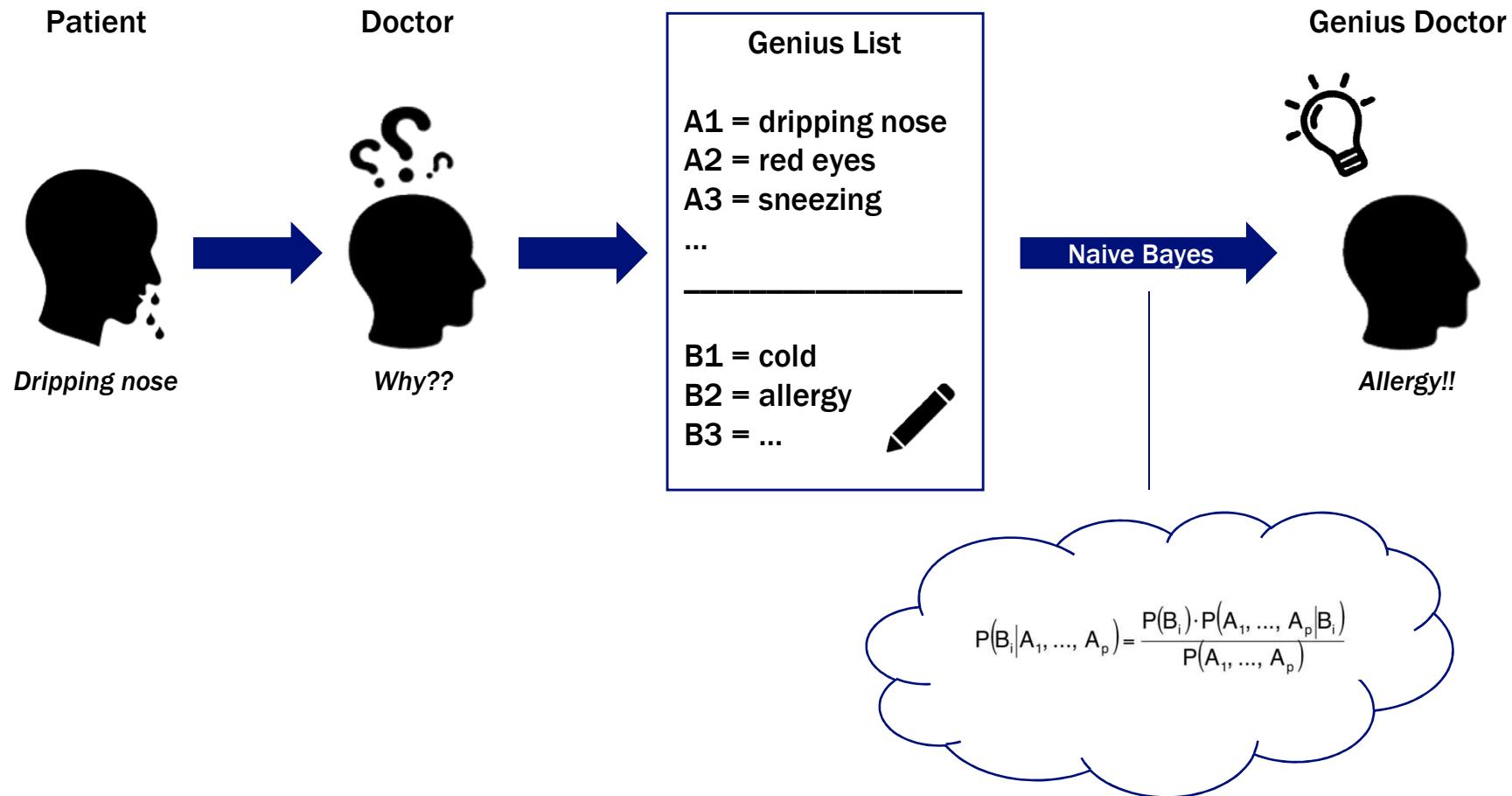
$P(B_i)$ is the prior probability of class i.

$P(A|B_i)$ is the conditional probability of the predictor given class i.

$P(A)$ is the prior probability of the predictor.

Despite its simplicity the Naive Bayes classifier often achieves better results than the more complex methods. It is especially appropriate when the dimension p of the feature space is high. It performs well in case of categorical input variables compared to numerical variables. For numerical variables the normal distribution is assumed.

Naive Bayes Classifier (II)



Naive Bayes Classifier (III)

$P(B_i|A_1, \dots, A_p)$ stands for the conditional probability of the occurrence of event B_i under the condition, that the events A_1, \dots, A_n have been occurred. It can be calculated using Bayes's theorem

$$P(B_i|A_1, \dots, A_p) = \frac{P(B_i) \cdot P(A_1, \dots, A_p|B_i)}{P(A_1, \dots, A_p)}$$

assuming that the data needed for the calculation of $P(A_1, \dots, A_p|B_i)$ can be determined.

This is often very difficult to measure in practice. To solve the problem a naive assumption is made:

"Based on the event B_i the events A_1, \dots, A_p are stochastically independent."

Therefore, it follows:

$$P(A_1, \dots, A_p|B_i) = \prod_{j=1}^p P(A_j|B_i) = P(A_1|B_i) \cdot P(A_2|B_i) \cdot \dots \cdot P(A_p|B_i)$$

Naive Bayes Classifier (IV)

The classification of a new object x with the Naive Bayes Classifier is done by calculating the conditional probabilities $P(B_i|A_1, \dots, A_p)$ for all classes i and the selection of the class, which has the highest probability:

$$P(B_i|A_1, \dots, A_p) = \operatorname{argmax}_{B_i \in \{B_1, \dots, B_k\}} \frac{P(B_i) \cdot \prod_{j=1}^p P(A_j|B_i)}{P(A_1, \dots, A_p)}$$

Since the probability $P(A_1, \dots, A_p)$ is a constant the formula can be simplified to:

$$P(B_i|A_1, \dots, A_p) = \operatorname{argmax}_{B_i \in \{B_1, \dots, B_k\}} P(B_i) \cdot \prod_{j=1}^p P(A_j|B_i)$$

The construction of a Naive-Bayes-Classifier is based on the estimation of the a priori probabilities $P(B_i)$ and the conditional probabilities $P(A_j|B_i)$. These probabilities are then used for the classification of new objects.

Application of the Naive Bayes Classifier

Object $x = (\text{Outlook}=\text{sunny}, \text{Temperature}=\text{cold}, \text{Humidity}=\text{high}, \text{Wind}=\text{strong})$

We need the Probabilities:

$$P(\text{play} = \text{yes}) = \frac{9}{14} = 0,64$$

$$P(\text{play} = \text{no}) = \frac{5}{14} = 0,36$$

$$P(\text{wind} = \text{strong} | \text{play} = \text{yes}) = \frac{3}{9} = 0,33$$

$$P(\text{wind} = \text{strong} | \text{play} = \text{no}) = \frac{3}{5} = 0,6$$

$$P(\text{outlook} = \text{sunny} | \text{play} = \text{yes}) = \frac{2}{9} = 0,22$$

$$P(\text{outlook} = \text{sunny} | \text{play} = \text{no}) = \frac{3}{5} = 0,6$$

$$P(\text{humidity} = \text{high} | \text{play} = \text{yes}) = \frac{3}{9} = 0,33$$

$$P(\text{humidity} = \text{high} | \text{play} = \text{no}) = \frac{4}{5} = 0,8$$

$$P(\text{temperature} = \text{cold} | \text{play} = \text{yes}) = \frac{3}{9} = 0,33$$

$$P(\text{temperature} = \text{cold} | \text{play} = \text{no}) = \frac{1}{5} = 0,2$$

Based on this it follows:

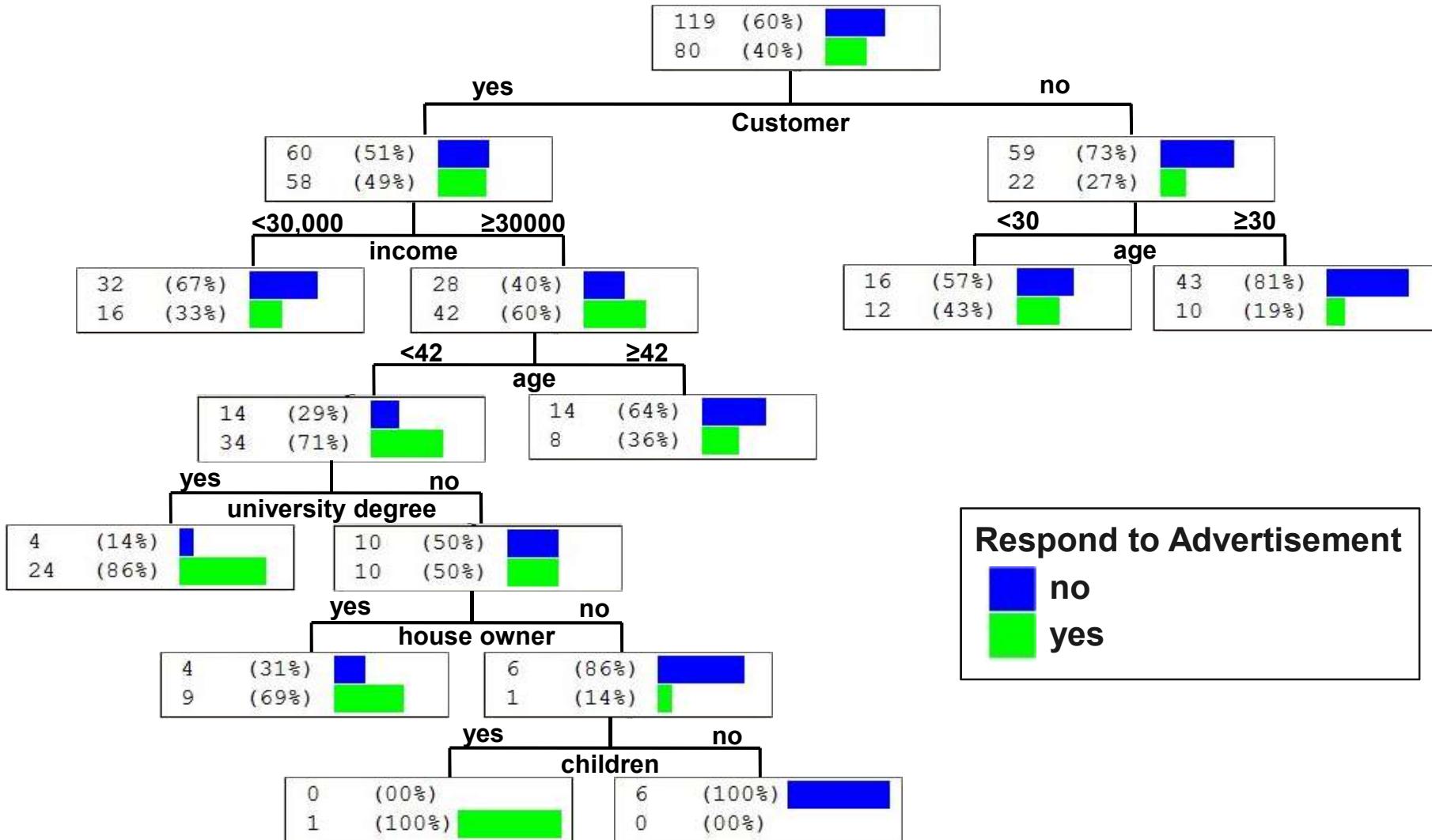
$$P(\text{play} = \text{yes}) \cdot \prod_{j=1, A_j \in x}^p P(A_j | \text{play} = \text{yes}) = 0,64 \cdot 0,33 \cdot 0,22 \cdot 0,33 \cdot 0,33 = 0,0053$$

$$P(\text{play} = \text{no}) \cdot \prod_{j=1, A_j \in x}^p P(A_j | \text{play} = \text{no}) = 0,36 \cdot 0,6 \cdot 0,6 \cdot 0,8 \cdot 0,2 = 0,0207$$

We choose the class $\text{play} = \text{no}$ with the probability: $\frac{0,0207}{0,0053 + 0,0207} = 0,796$

Decision Trees

Introductory Example



Decision Trees (I)

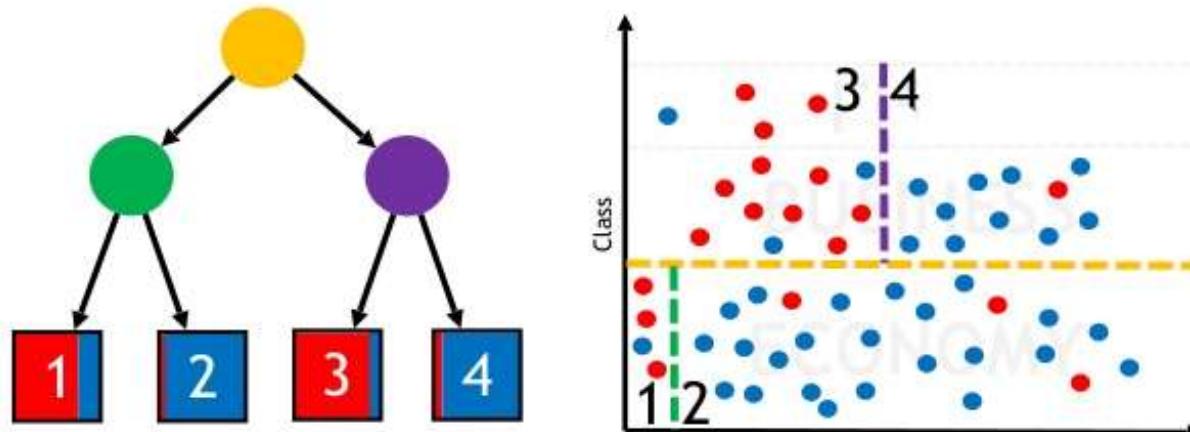
Decision trees belong to the hierarchical methods of classification. They analyze step-by-step (recursive partitioning).

A decision tree consists of nodes and borders. The topmost node (without any parent node) is called "root". A node without a child node is called "leaf". Nodes that have parent and child nodes are called "interior nodes". The interior nodes represent the splitting of the included object sets. An interior node has at least two child nodes (sons). If every interior node has exactly two child nodes, the tree is called a "binary tree".

A decision tree method starts at the root, which includes all objects. The different features are compared (with an adequate measure) regarding their suitability of classification. The most appropriate feature determines the branching of the current set of objects: regarding this feature, the current set of objects is divided into disjoint subsets (partitioning). This method is now used recursively to the created child nodes (subsets).

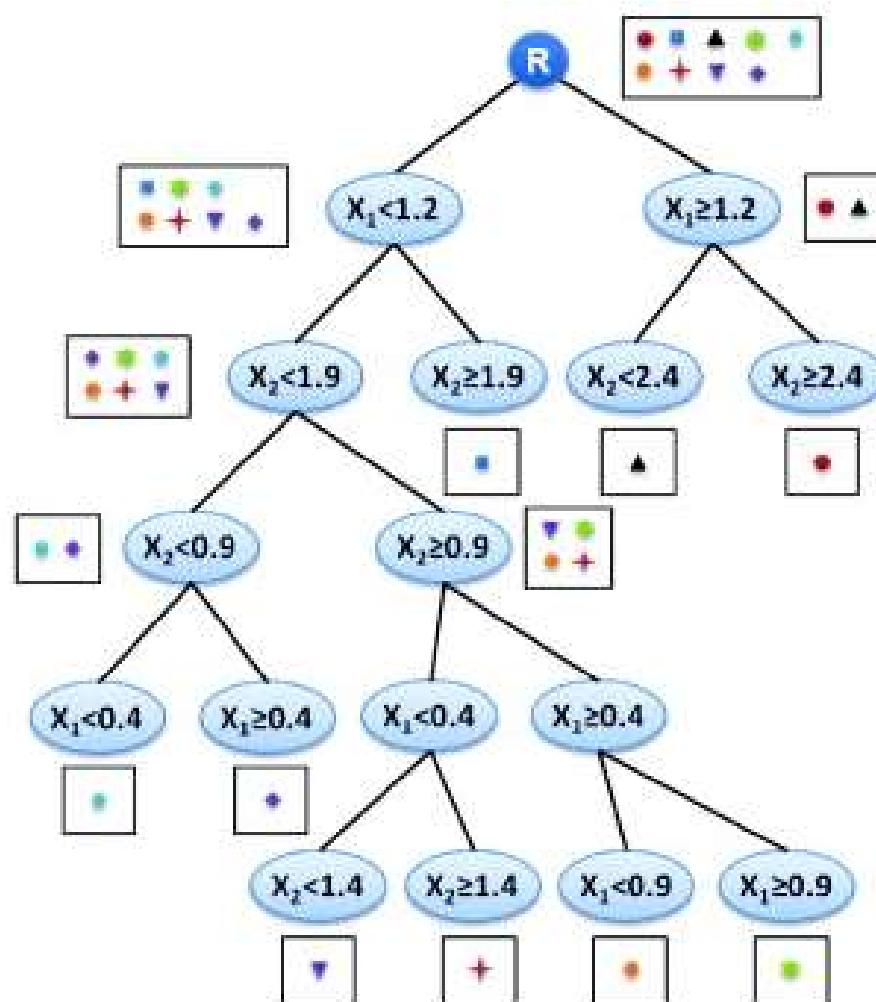
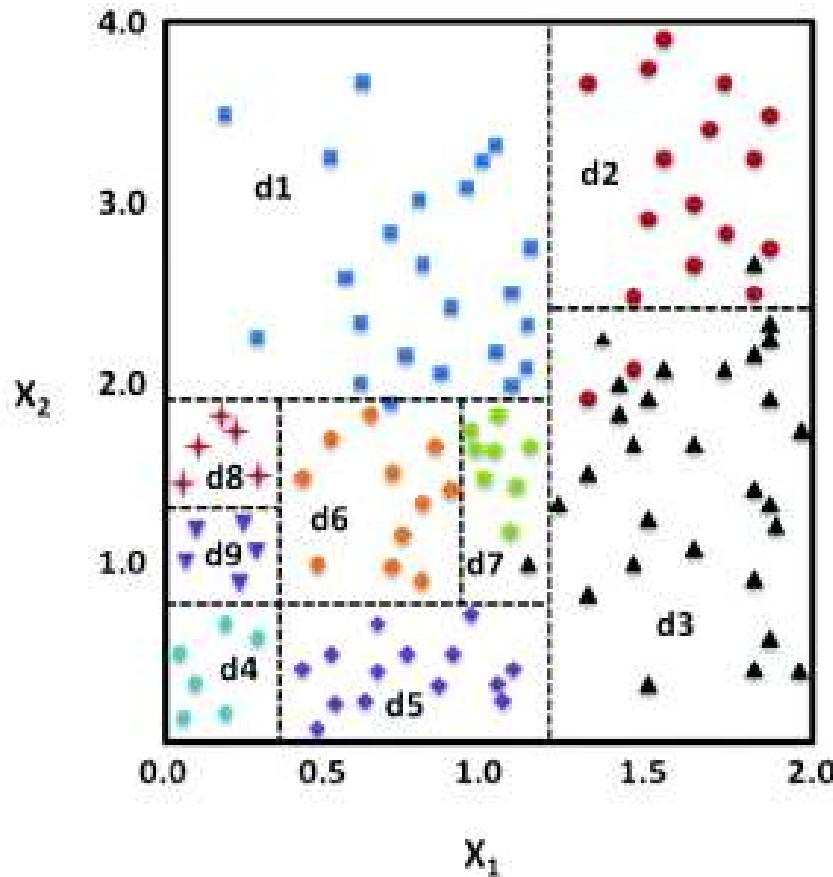
Decision Trees (II)

Graphically, decision tree models divide the dataspace in a large number of subspaces and search for the variables which are able to split the dataspace with the greatest homogeneity. We can think of the decision tree as a map of different path. For a distinct combination of predictor variables and their observed values, we would enter a specific path, which gives the classification in the leaf of the decision tree.



The decision tree approach does not require any assumption about the functional form of variables or distributions. Furthermore in contrast to parametric models like linear regressions, the decision tree algorithm can model multiple structures as well as complex relationships within the data, which would be difficult to replicate in a linear model.

Decision Trees (III)



Source: <http://iopscience.iop.org/article/10.1088/1749-4699/5/1/015004>

Overview of important Decision Tree Methods

| Name | CART | ID3 | C5.0 | CHAID | Random Forests |
|-------------------|---|---|---|--|--|
| Idea | Choose the attribute with the highest information content | One of the first methods from Quinlan; uses the concept of information gain | Like ID3 based on the concept of information gain | Choose the attribute that is most dependent from the target variable | Construct many trees with different sets of features and samples (randomly). Result by voting. |
| Measure used | Gini-Index | Information gain (entropy) | Ratio of information gain | Chi-square split | optional, mostly Gini-Index |
| Type of Splitting | binary | complete, pruning | complete, pruning | complete, pruning | complete |

Splitting with Entropy in ID3

| Day | wind | play tennis |
|-----|--------|-------------|
| 1 | light | no |
| 2 | strong | no |
| 3 | light | yes |
| 4 | light | yes |
| 5 | light | yes |
| 6 | strong | no |
| 7 | strong | yes |
| 8 | light | no |
| 9 | light | yes |
| 10 | light | yes |
| 11 | strong | yes |
| 12 | strong | yes |
| 13 | light | yes |
| 14 | strong | no |

ID3 uses the entropy as a measure for the impurity of a node t:

$$\text{entropy}(t) = - \sum_{i=1}^k p_i \cdot \log_2 p_i$$

where p_i = relative frequency of cases in a node, that belong to a class i

Attribute Wind:

light: 6 x yes, 2 x no

$$\Rightarrow \text{entropy}(\text{light}) = - \left(\frac{6}{8} \cdot \log_2 \frac{6}{8} + \frac{2}{8} \cdot \log_2 \frac{2}{8} \right) = 0.811$$

strong: 3 x yes, 3 x no

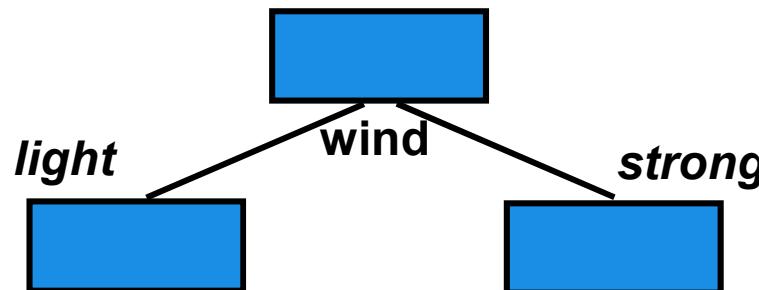
$$\Rightarrow \text{entropy}(\text{strong}) = - \left(\frac{3}{6} \cdot \log_2 \frac{3}{6} + \frac{3}{6} \cdot \log_2 \frac{3}{6} \right) = 1.0$$

Calculating the Information Gain

The information gain is a measure, that shows (by combination of the entropies) the appropriateness of an attribute for splitting:

$$\text{Information gain} = \text{entropy}(t) - \sum_{i=1}^m \frac{t_i}{t} \cdot \text{entropy}(t_i)$$

where m = number of values (here two: light, strong), t_i = number of data sets with strong or light wind (8 resp. 6), t = total number of data sets (14) and $\text{entropy}(t)$ = entropy before splitting.



$$\text{Information gain} = 0.94 - \left(\frac{8}{14} \cdot 0.811 + \frac{6}{14} \cdot 1.0 \right) = 0.048$$

Decision using ID3

Information gain (outlook) = 0.246

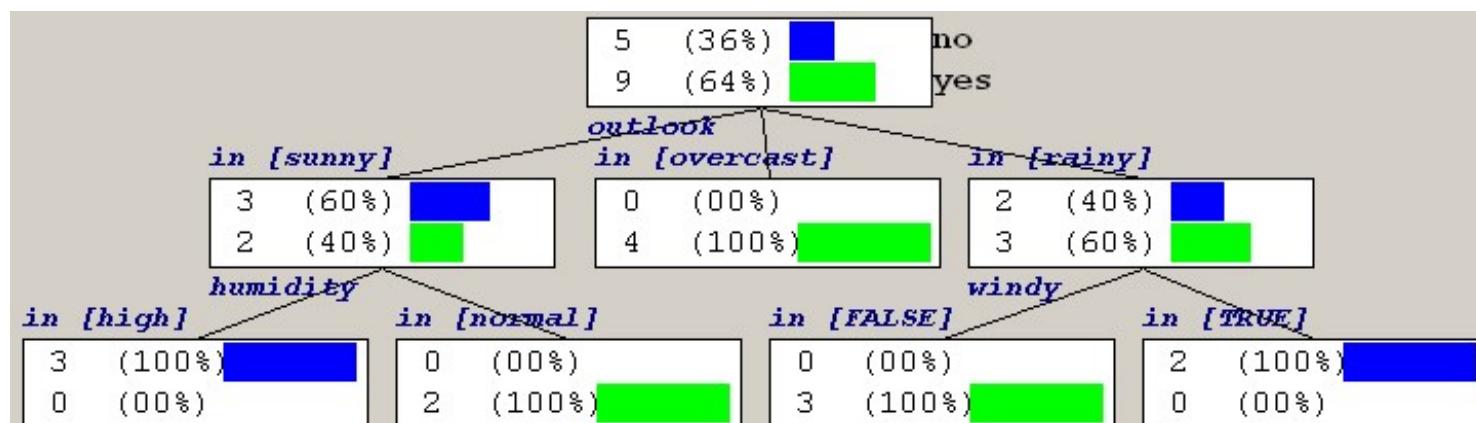
Information gain (humidity) = 0.151

Information gain (wind) = 0.048

Information gain (temperature) = 0.029

We choose the attribute with the largest information gain (here: outlook) for the first splitting.

As solution we obtain the following tree:



Decision using C5.0

ID3 tends to favor attributes that have a large number of values, resulting in larger trees. For example, if we have an attribute that has a distinct value for each record, then the entropy is 0, thus the information gain is maximal.

To compensate for this, C5.0 is a further development that uses the information gain ratio as a splitting criterion:

$$\text{Information gain ratio} = \frac{\text{entropy}(t) - \sum_{i=1}^m \frac{t_i}{t} \cdot \text{entropy}(t_i)}{- \sum_{i=1}^m \frac{t_i}{t} \cdot \log_2 \frac{t_i}{t}}$$

In the case of our example the GainRatio of Windy is

$$\text{Information gain ratio(Windy)} = \frac{0.048}{-\frac{6}{14} \cdot \log_2 \frac{6}{14} - \frac{8}{14} \cdot \log_2 \frac{8}{14}} = 0.049$$

and the GainRatio of Outlook is

$$\text{Information gain ratio(Outlook)} = \frac{0.246}{-\frac{5}{14} \cdot \log_2 \frac{5}{14} - \frac{4}{14} \cdot \log_2 \frac{4}{14} - \frac{5}{14} \cdot \log_2 \frac{5}{14}} = 0.156$$

Handling Numerical Attributes

Numerical attributes are usually splitted binary. In contrast to categorical attributes many possible splitting points exist .

The splitting point with the highest information gain is looked for. For this, the potential attribute is sorted according to its values first and then all possible splitting point and the corresponding information gains are calculated. In extreme cases there exists $n-1$ possibilities.

| day | temperature | play |
|-----|-------------|------|
| 5 | 64 | yes |
| 6 | 65 | no |
| 7 | 68 | yes |
| 9 | 69 | yes |
| 11 | 70 | yes |
| 8 | 71 | no |
| 4 | 72 | yes |
| 14 | 72 | no |
| 10 | 75 | yes |
| 12 | 75 | yes |
| 2 | 80 | no |
| 3 | 81 | yes |
| 13 | 83 | yes |
| 1 | 85 | no |

Test the splitting point temperature = 71.5:

temperature ≤ 71.5 : 4 x yes, 2 x no

$$\Rightarrow \text{Entropy}(< 71.5) = -\left(\frac{4}{6} \cdot \log_2 \frac{4}{6} + \frac{2}{6} \cdot \log_2 \frac{2}{6}\right) = 0.918$$

temperature > 71.5 : 5 x yes, 3 x no

$$\Rightarrow \text{Entropy(strong)} = -\left(\frac{5}{8} \cdot \log_2 \frac{5}{8} + \frac{3}{8} \cdot \log_2 \frac{3}{8}\right) = 0.954$$

We calculate:

$$\text{Information gain} = 0.94 - \left(\frac{6}{14} \cdot 0.918 + \frac{8}{14} \cdot 0.954\right) = 0.00143$$

The CART Algorithm

The CART algorithm (Classification And Regression Trees) constructs trees that have only binary splits. Like C5.0, it is able to handle categorical and numerical attributes.

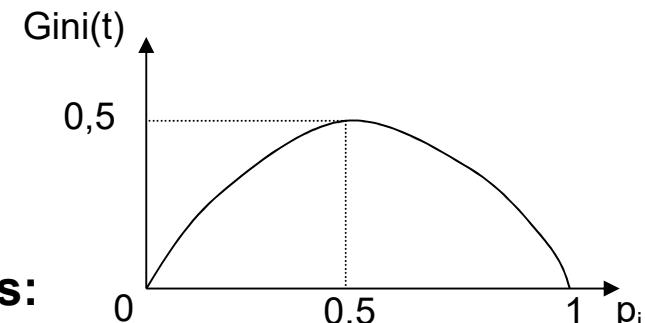
As a measure for the impurity of a node t , CART uses the Gini Index. In the case of two classes the Gini Index is defined as:

$$\text{Gini}(t) = 2 \cdot p_1 \cdot p_2 = 2 \cdot p_1 \cdot (1 - p_1)$$

where p_i = relative frequency of cases in a node, that belong to a class i , $i=1,2$.

A node t is splitted by the attribute that minimizes:

$$\frac{t_1}{t} \cdot \text{Gini}(t_1) + \frac{t_2}{t} \cdot \text{Gini}(t_2) \rightarrow \text{Min}$$



This is equivalent to:

$$\text{Information gain} = \text{Gini}(t) - \sum_{i=1}^2 \frac{t_i}{t} \cdot \text{Gini}(t_i) \rightarrow \text{Max}$$

Splitting in CART

| Day | wind | play tennis |
|-----|--------|-------------|
| 1 | light | no |
| 2 | strong | no |
| 3 | light | yes |
| 4 | light | yes |
| 5 | light | yes |
| 6 | strong | no |
| 7 | strong | yes |
| 8 | light | no |
| 9 | light | yes |
| 10 | light | yes |
| 11 | strong | yes |
| 12 | strong | yes |
| 13 | light | yes |
| 14 | strong | no |

Attribute Wind:

light: 6 x yes, 2 x no

$$\Rightarrow \text{Gini}(\text{light}) = 2 \cdot \frac{6}{8} \cdot \left(1 - \frac{6}{8}\right) = 0.375$$

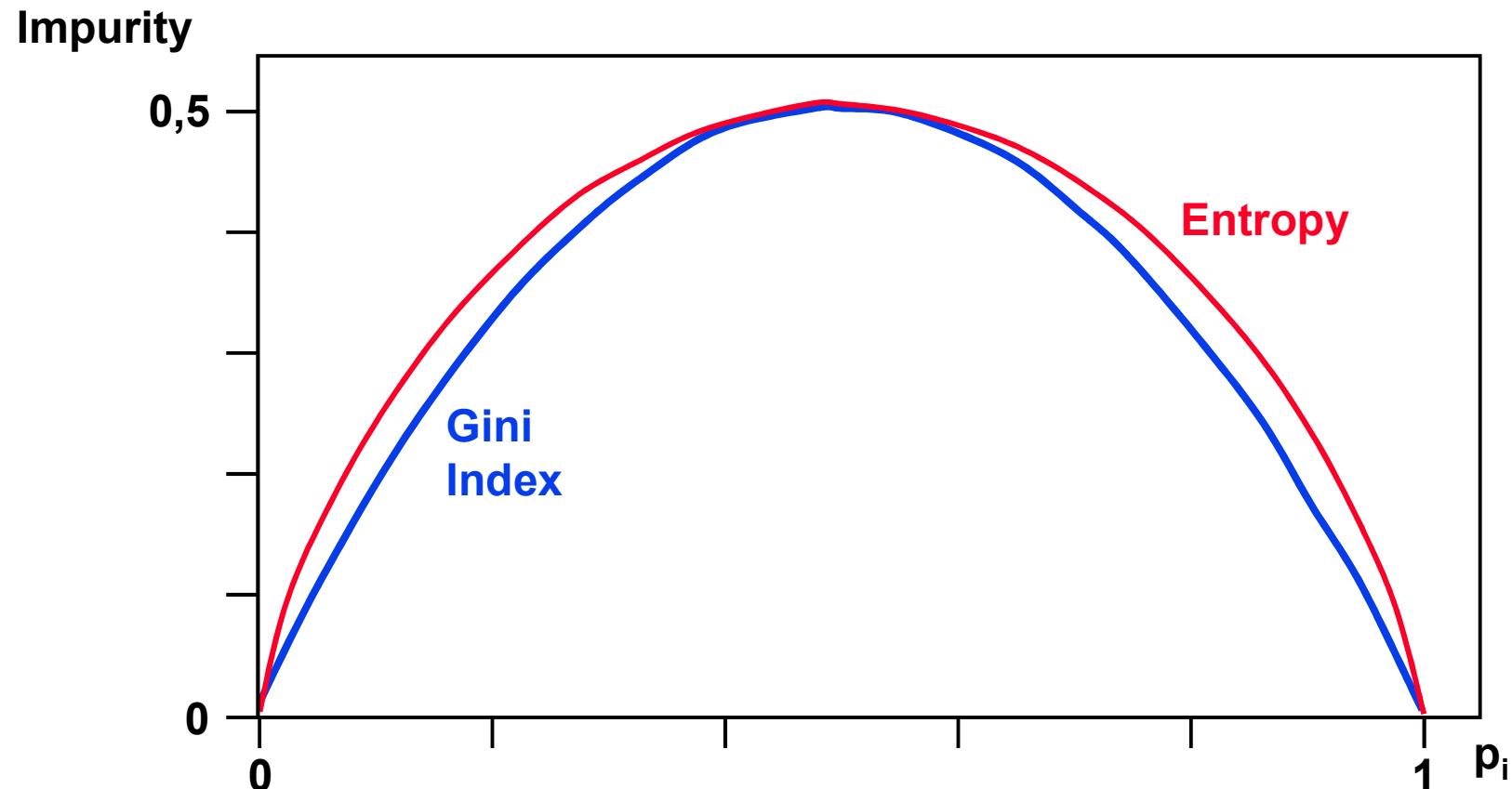
strong: 3 x yes, 3 x no

$$\Rightarrow \text{Gini}(\text{strong}) = 2 \cdot \frac{3}{6} \cdot \left(1 - \frac{3}{6}\right) = 0.5$$

resulting in:

$$\begin{aligned} \text{Information gain} &= 0.46 - \left(\frac{8}{14} \cdot 0.375 + \frac{6}{14} \cdot 0.5 \right) \\ &= 0.0306 \end{aligned}$$

Coherence between Entropy and Gini Index

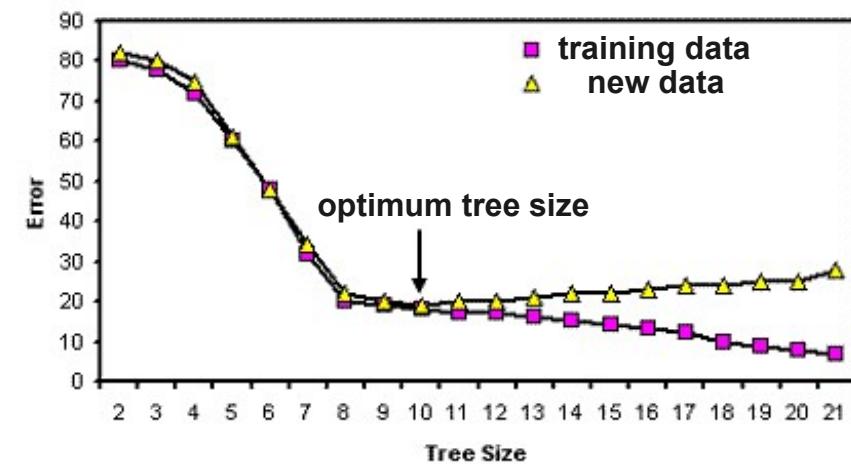


Remark: Entropy has been scaled from (0, 1) to (0, 0.5)!

Pruning Decision Trees

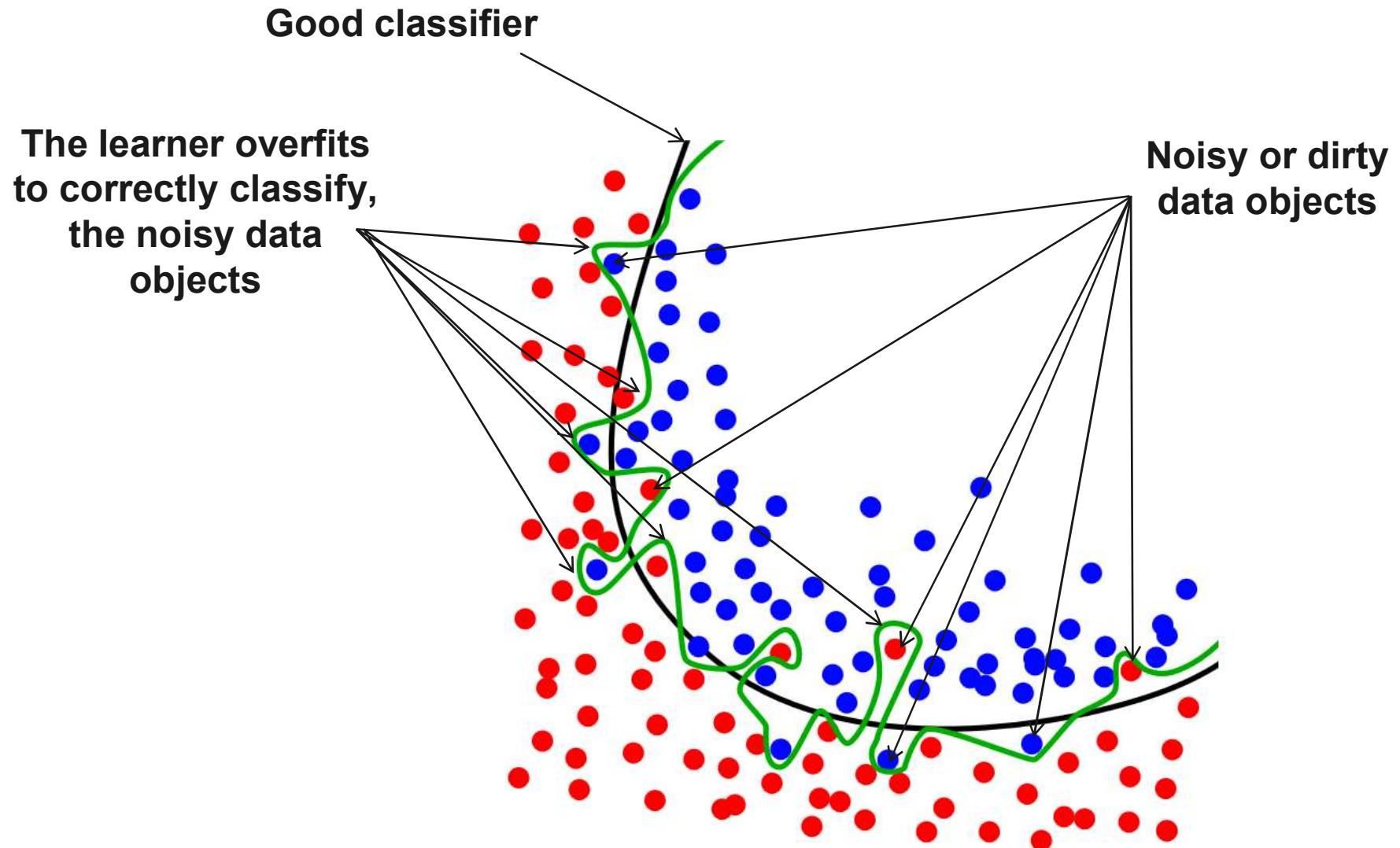
Most decision tree algorithms partition training data until every node contains objects of a single class, or until further partitioning is impossible because two objects have the same value for each attribute but belong to different classes. If there are no such conflicting objects, the decision tree will correctly classify all training objects.

If tree performance is measured from the number of correctly classified cases it is common to find that the training data gives an over-optimistic guide to future performance, i.e. with new data. A tree should exhibit generalization, i.e. work well with data other than those used to generate it. When the tree grows during training it often shows a decrease in generalization. This is because the deeper nodes are fitting noise in the training data not representative over the entire universe from which the training set was sampled. This is called 'overfitting'.



Overfitting can be avoided by pruning the tree so that this will improve generalization. This can be done by using a stopping criterion that prevents some sets of training cases from being subdivided (prepruning), or by removing some of the structure of the tree after it has been produced (postpruning).

Overfitting



Reduced Error Pruning

One of the simplest forms of postpruning is reduced error pruning. This essentially involves growing the tree from a dataset until all possible leaf nodes have been reached and then removing particular subtrees.

Procedure:

- Classify examples in test set – some might be errors
- For each node:
 - Sum the errors over entire subtree
 - Calculate error on same example if converted to a leaf with majority class label
- Prune node with highest reduction in error
- Repeat until error no longer reduced

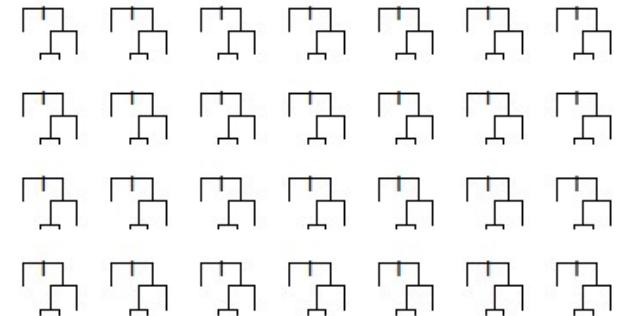
The procedure is then recursed over the freshly pruned tree until there is no possible reduction in error rate at any node.

Random Forest (I)

Random forest is an ensemble classifier that consists of many decision trees.

For every tree a subset of the data objects and at every splitting point a subset of features is randomly chosen. Then the tree is constructed usually using the Gini Index.

In the end, a simple majority vote is taken for prediction.

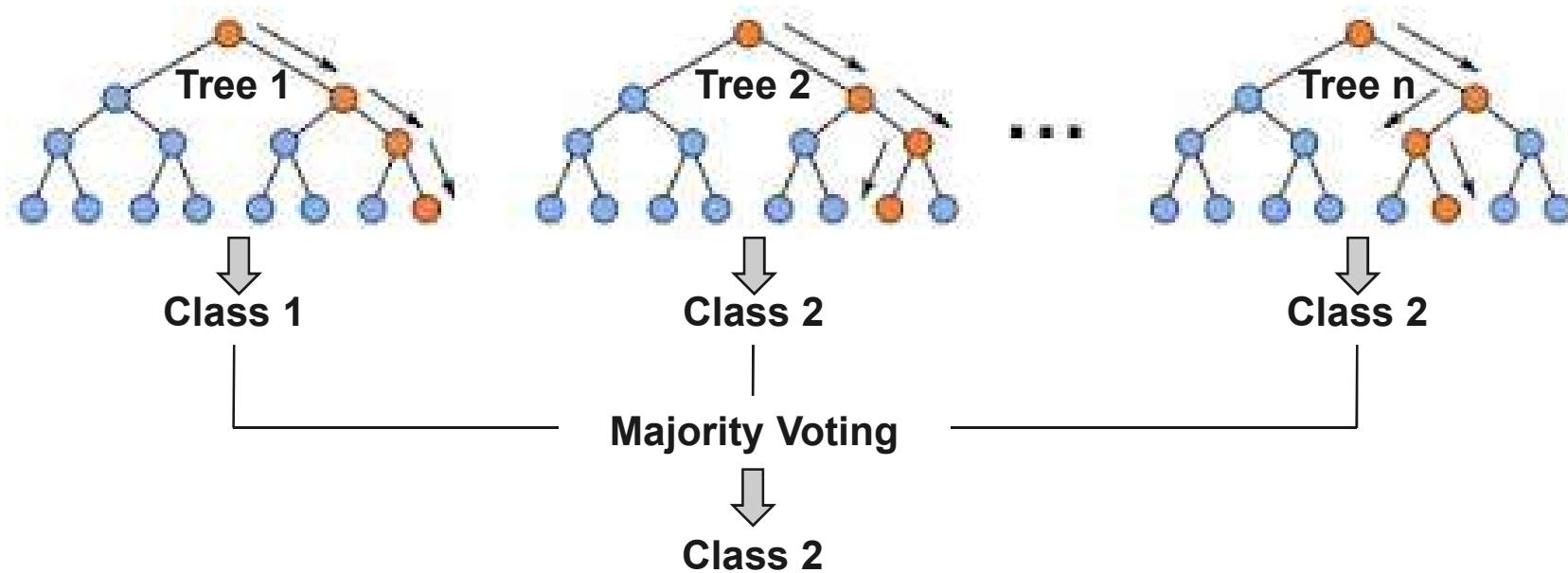


Algorithm:

- 1. Create n samples from the original data. Frequent sample size is 2/3.**
- 2. For each of the samples, grow a tree, with the following modification: at each node, rather than choosing the best split among all predictors, randomly sample m^* of the m predictors and choose the best split from among those variables.**
- 3. Predict by aggregating the predictions of the n trees (majority votes).**

Random Forest (II)

Voting-Principle of Random Forest:

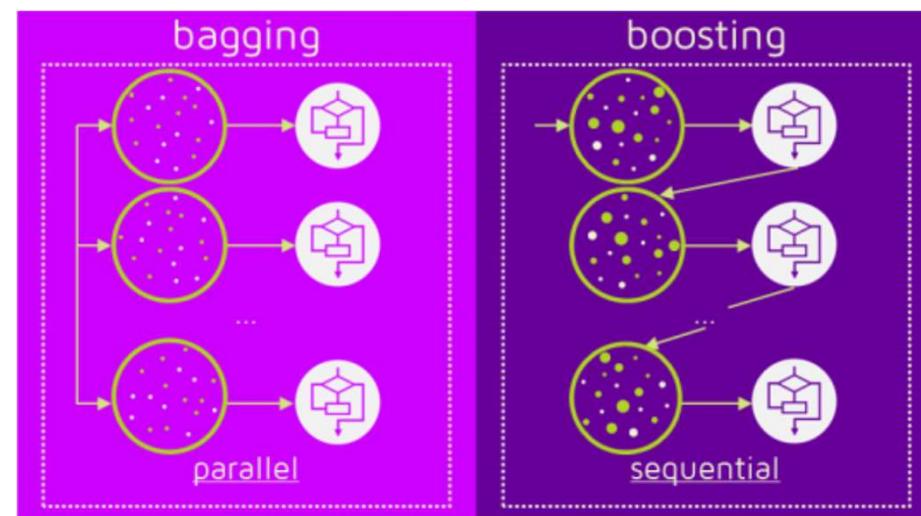


To avoid overfitting effects, the size and the depth of the trees can be restricted.

Boosted Trees (I)

Similar to Random Forest, Boosted Decision Trees use an ensemble of multiple trees to create more powerful classifications. Instead of building the ensemble parallel, Boosting Decision Trees build a series of trees where every tree within this series tries to correct the errors of the previous tree. That means, in boosting algorithms each classifier is trained on data, taking into account the previous classifiers' success.

There exist different approaches to feed the subsequent trees with data for training. Some methods use only the data objects producing errors from the previous tree others combine the errors with a random subsample of the training data set and others use weighting for the data objects. In the last case, the weights for the correct predicted objects are reduced and in the other case increased.



Boosted Trees (II)

Example (AdaBoost):

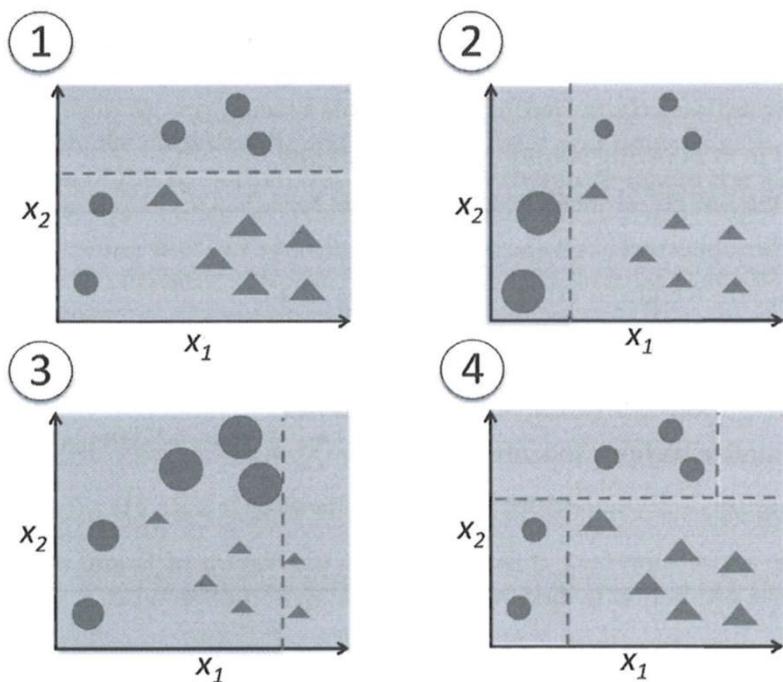


Figure 1, which represents a training set for binary classification where all training samples are assigned equal weights. Based on this training set, we train a decision stump (shown as a dashed line) that tries to classify the samples of the two classes (triangles and circles) as well as possible. Now, we assign a larger weight to the two previously misclassified samples (figure 2) and we lower the weight of the correctly classified samples. The next decision stump will now be more focused on the training samples that have the largest weights. The model shown in figure 2 misclassifies three different samples from the circle-class, which are then assigned a larger weight as shown in figure 3.

We can now combine the three models trained on different reweighted training subsets by a (weighted) majority vote, as shown in figure 4. Even if none of the single models is able to correctly classify all of our training samples, the resulting ensemble classifier classifies everything correctly.

Source: Raschka, S.: Python Machine Learning, pp. 225.

Gradient Boosted Trees (I)

Gradient Boosted Trees use a combination of Boosting, Gradient Descent, and Decision Trees. They build an ensemble based on the boosting principle through minimizing an arbitrary differentiable loss function using the gradient descent approach. They can be applied to classification and regression.

The general form of the loss function is

$$L(y_i, \hat{y}_i) = L(y_i, f(x_i))$$

where \hat{y}_i are the predicted values created by the model $f(x_i)$.

Typical examples are the quadratic loss function

$$\sum_{i=1}^n (y_i - f(x_i))^2$$

or the loss function for a tree f with T terminal nodes

$$L(f) = \sum_{j=1}^T L_j$$

where L_j is the aggregated loss at node j .

Gradient Boosted Trees (II)

Procedure:

1. Train an initial tree $f_0(x)$ based on the training data
2. Calculate the errors and their aggregated value (loss) produced by the tree
3. Use the errors as values for the new output variable (instead of the original y)
4. Train a tree $f_t(x)$ based on the new output variable to minimize the errors (loss function) using gradient descent
5. Repeat step 2 to 4 until $L_t \cong L_{t-1}$ or a number of m trees is created
6. Apply the ensemble using

$$\hat{y} = \sum_{t=0}^m f_t(x) = f_0(x) + f_1(x) + f_2(x) + \cdots + f_m(x)$$

where m =number of boosted trees

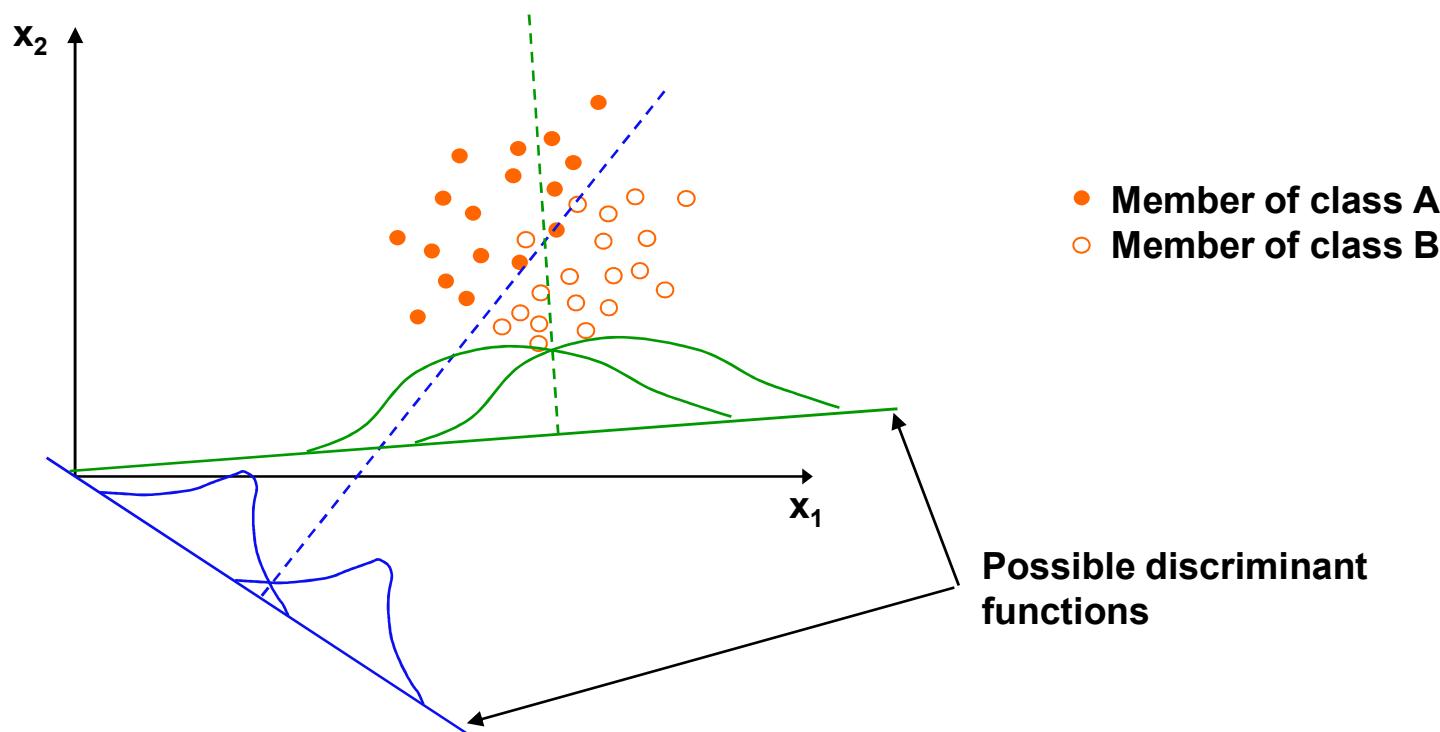
Remark: Even for classification tasks, Gradient Boosted Trees will be using regression trees. The algorithm will compute continuous values between 0 and 1 that are probabilities of belonging to a certain class.

Discriminant Analysis

Discriminant Analysis

We search for a straight line (discriminant function) which best separates the classes. That means, where the variation between the categories is as possible high and the variation within a category is as possible small. This line is found if the projection of the category's elements on the discriminant function shows minimal overlapping.

An example for a two-categories case:



Calculation of the Discriminant Function (I)

Geometrically contemplated, the discriminant function is a linear combination of features. That means, each point d along the line is described via

$$d = \sum_{i=1}^m b_i \cdot x_i$$

With m = number of features.

In the two-features case, we have

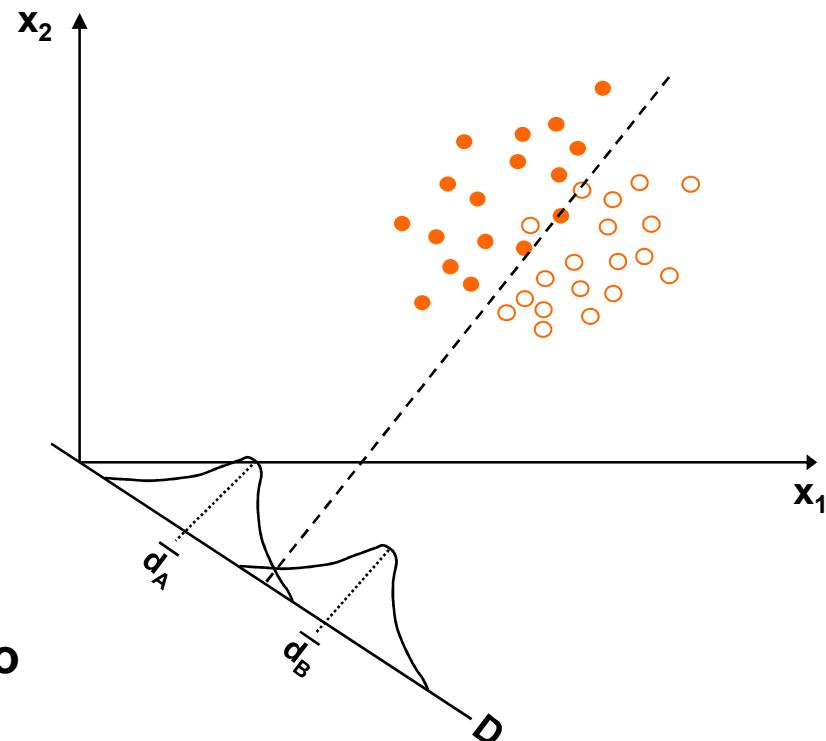
$$d = b_1 \cdot x_1 + b_2 \cdot x_2$$

whereas the **slope** of the discriminant function is defined by the ratio

$$x_2 = \frac{b_2}{b_1} \cdot x_1$$

The discriminant coefficients b_i now have to be estimated via training data.

Annotation: in the k -class case exist $k-1$ discriminant functions!



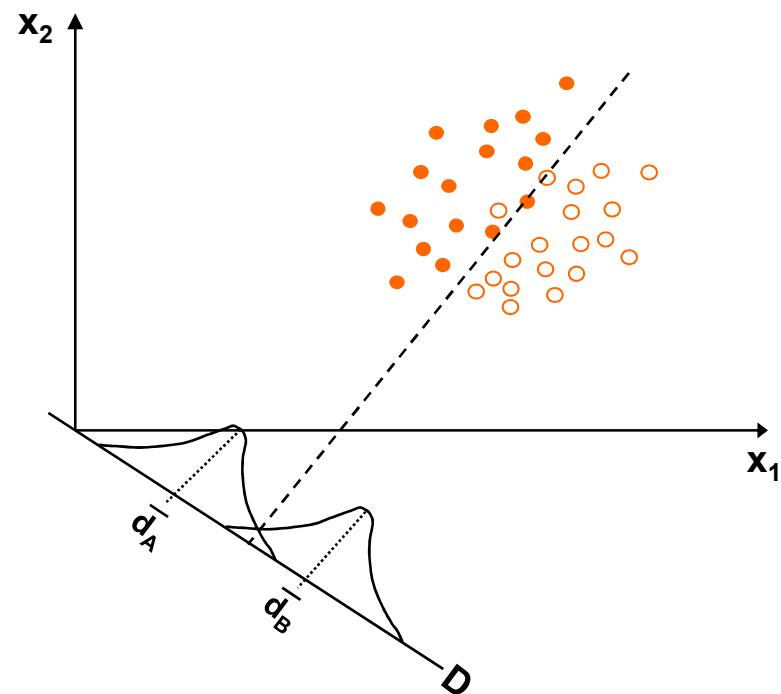
Calculation of the Discriminant Function (II)

An optimal separation exists if the overlapping range of the frequency distribution onto the discriminant function is small. That is why the coefficients are found if

$$\frac{(\bar{d}_A - \bar{d}_B)^2}{\sum_{j=1}^{n_A} (d_{A,j} - \bar{d}_A)^2 + \sum_{j=1}^{n_B} (d_{B,j} - \bar{d}_B)^2} \rightarrow \text{Max!}$$

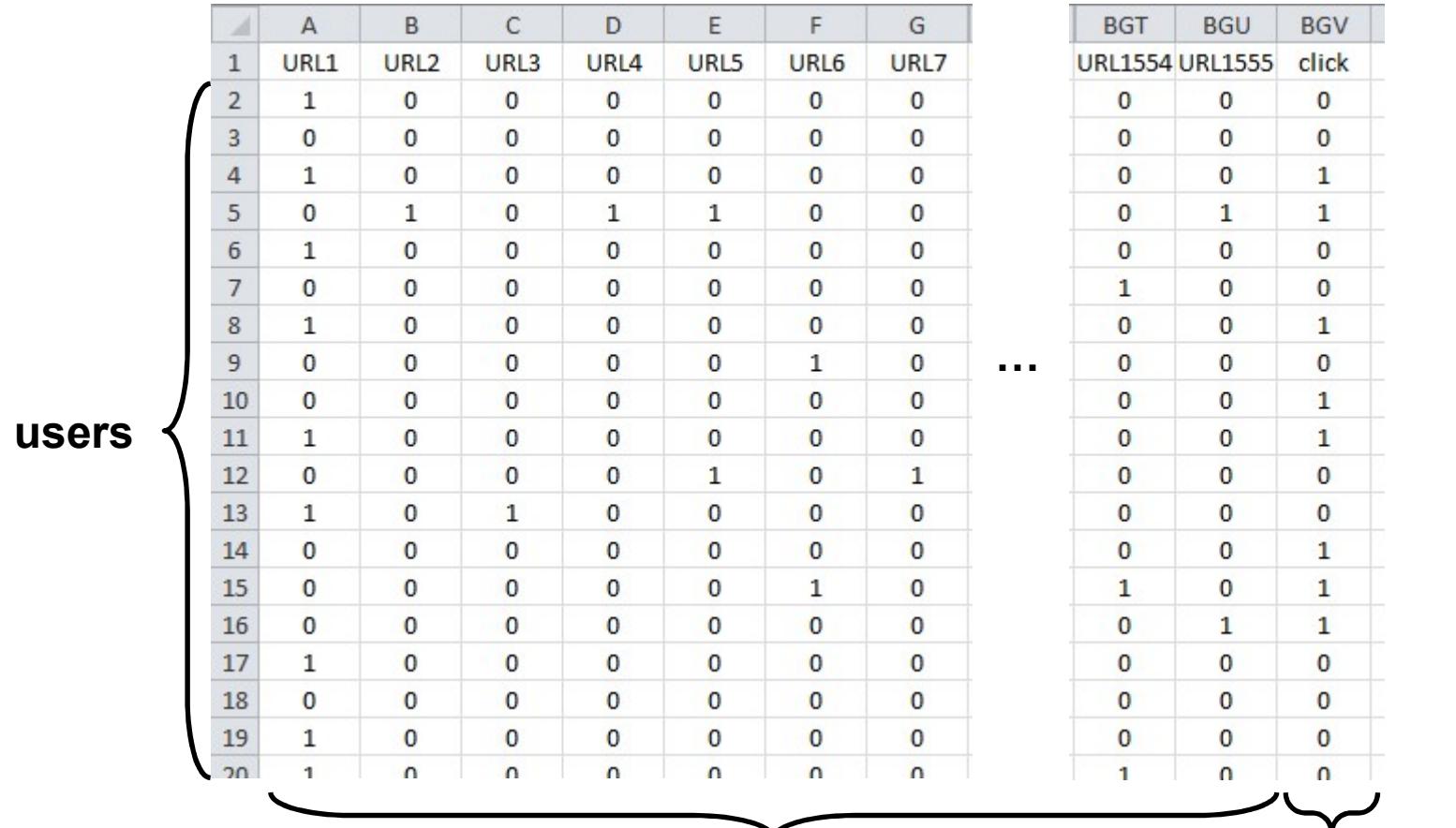
The rule of classification for new objects is:

If the discriminant value d of the object lies closer to the average value \bar{d}_A , it belongs to class A, otherwise to B.



Logistic Regression

Introductory Example



Giant sparse matrix!

One matrix for every ad!

users

websites (features)
1=visited
0=not visited

ad (target)
1=clicked
0=not clicked

| | A | B | C | D | E | F | G |
|----|------|------|------|------|------|------|------|
| 1 | URL1 | URL2 | URL3 | URL4 | URL5 | URL6 | URL7 |
| 2 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 3 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 4 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 5 | 0 | 1 | 0 | 1 | 1 | 0 | 0 |
| 6 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 7 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 8 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 9 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| 10 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 11 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 12 | 0 | 0 | 0 | 0 | 1 | 0 | 1 |
| 13 | 1 | 0 | 1 | 0 | 0 | 0 | 0 |
| 14 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 15 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| 16 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 17 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 18 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 19 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 20 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |

| | BGT | BGU | BGV |
|---------|-----|-----|-----|
| URL1554 | 0 | 0 | 0 |
| URL1555 | 0 | 0 | 0 |
| click | 0 | 0 | 1 |
| | 0 | 1 | 1 |
| | 0 | 0 | 0 |
| | 1 | 0 | 0 |
| | 0 | 0 | 1 |
| | 0 | 0 | 0 |
| | 0 | 0 | 1 |
| | 0 | 0 | 0 |
| | 0 | 0 | 0 |
| | 0 | 0 | 1 |
| | 0 | 0 | 0 |
| | 0 | 0 | 1 |
| | 0 | 0 | 0 |
| | 0 | 0 | 1 |
| | 1 | 0 | 1 |
| | 0 | 1 | 1 |
| | 0 | 0 | 0 |
| | 0 | 0 | 0 |
| | 0 | 0 | 0 |
| | 1 | 0 | 0 |

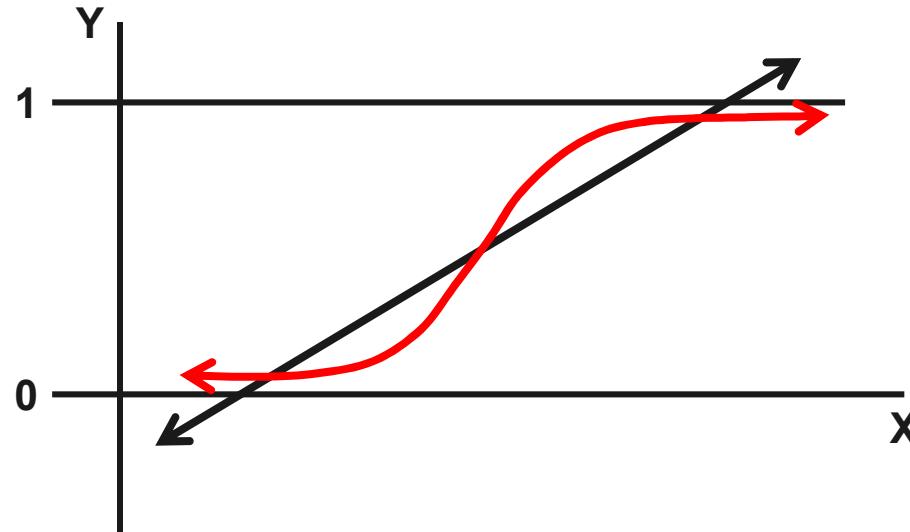
...

Why not classical linear regression?

It is possible to implement a linear regression on such a dataset where $Y=\{0,1\}$.

Problems:

- The predicted values of the linear model can be greater than 1 or less than 0



- ϵ is not normally distributed because Y takes on only two values
- The error terms are heteroscedastic (the error variance is not constant for all values of X)

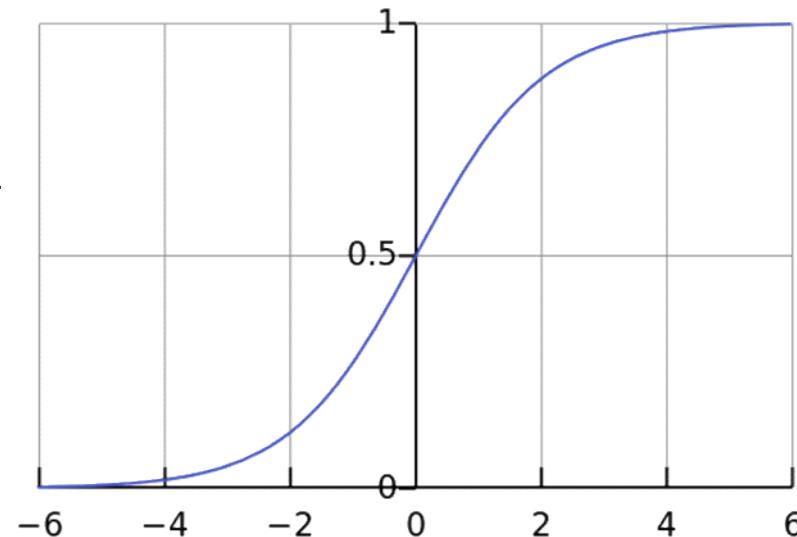
Source: Bichler (2015): Course Business Analytics, TU München

Logistic regression (I)

Logistic regression is a regression model where the dependent variable is categorical. The classical logistic regression is a binary classifier, where the dependent variable has two states. The output of a logistic regression model ranges between 0 and 1.

Logistic regression uses the logistic function (or Sigmoid function) because it can take an input with any value from negative to positive infinity, whereas the output always takes values between zero and one and hence is interpretable as a probability.

It is defined as: $F(x) = \frac{1}{1 + e^{-t}}$



Logistic regression (II)

If we set $t = \beta_0 + \beta'x$

the logistic function can now be written as: $F(x) = \frac{1}{1 + e^{-(\beta_0 + \beta'x)}}$

If we interpret $F(x)$ as the probability that the class attribute has the value 1 with the given input vector x .

The coefficients β_0 and β can be estimated via Maximum Likelihood Estimation.

The parameter β_0 represents the unconditional probability of "1" knowing nothing about the feature vector x .

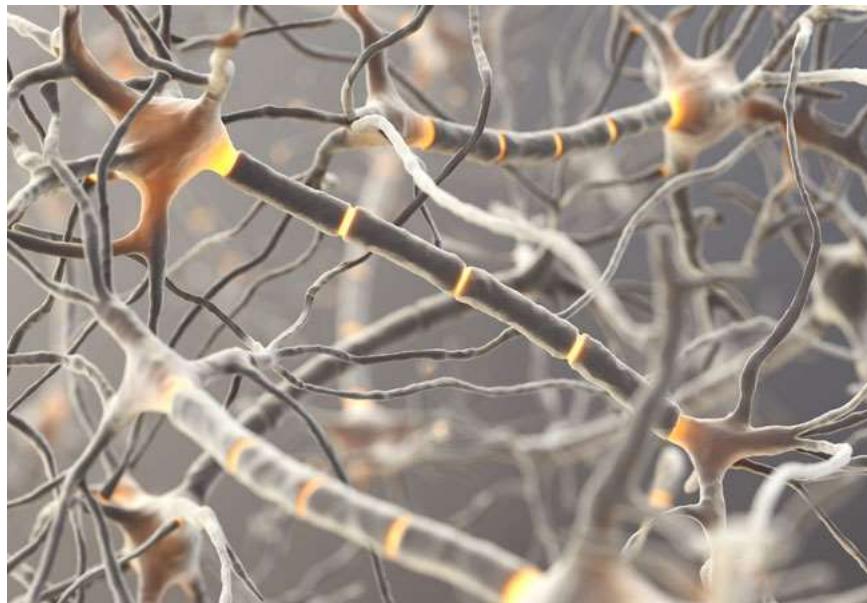
The parameter vector β defines the slope of the logit function. It determines the extent to which certain features contribute for increased or decreased likelihood to "1".

The output of a logistic model is a probability. To use this for classification purposes:

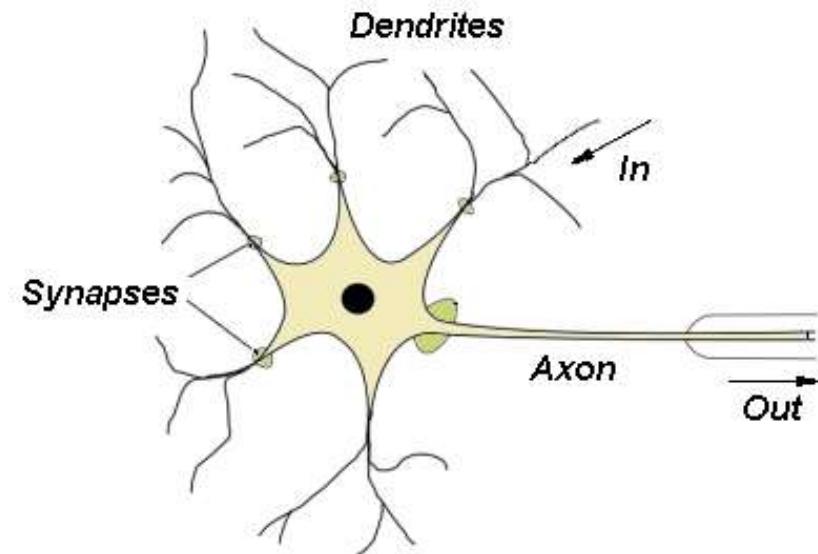
- If the predicted probability is > 0.5 the label is 1
- and otherwise 0.

Neural Networks

Functionality of Human Neurons

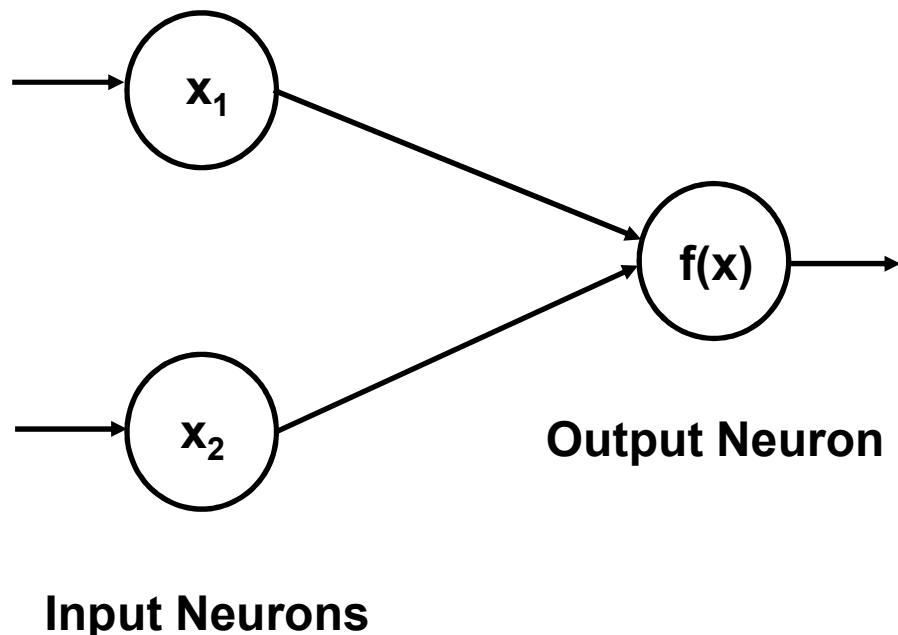


A Look into the Nervous System



Design of a Neuron

An Easy Example (I)



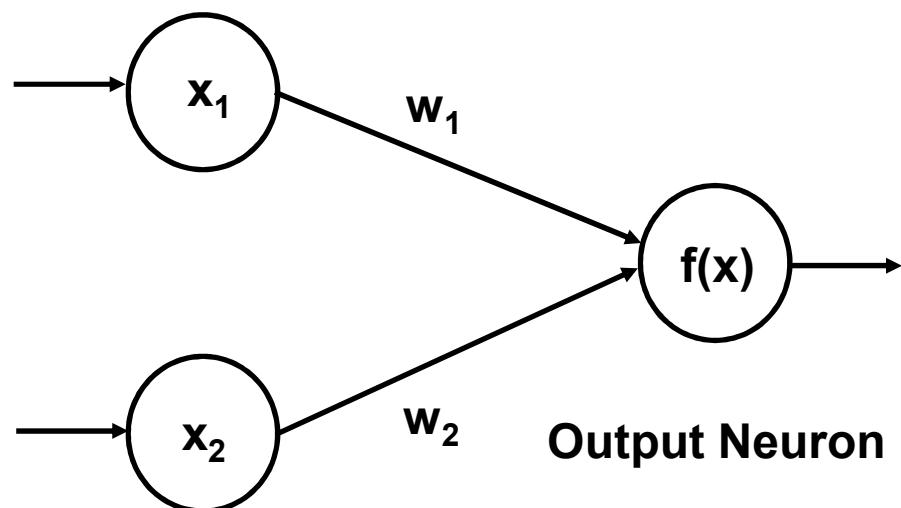
$$s(x) = \sum_{i=1}^2 x_i$$

f(x) = Activation function

e.g. $f(x) = \begin{cases} 1 & , \text{if } s(x) > t \\ 0 & , \text{otherwise} \end{cases}$

where t = Stimulus threshold

An Easy Example (II)



Input Neurons

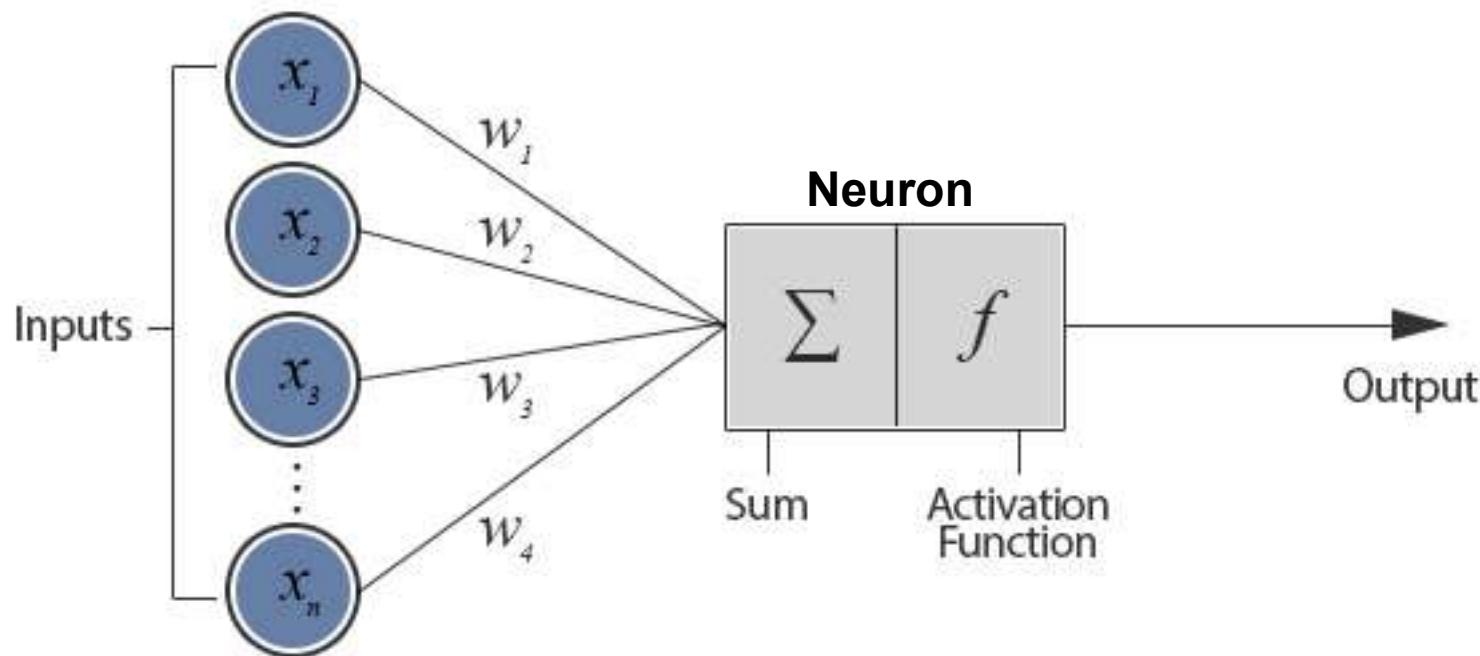
$$s(x) = \sum_{i=1}^2 x_i \cdot w_i$$

$f(x)$ = Activation function

e.g. $f(x) = \begin{cases} 1 & , \text{if } s(x) > t \\ 0 & , \text{otherwise} \end{cases}$

where t = Stimulus threshold

Functionality of a Neuron



The Perceptron

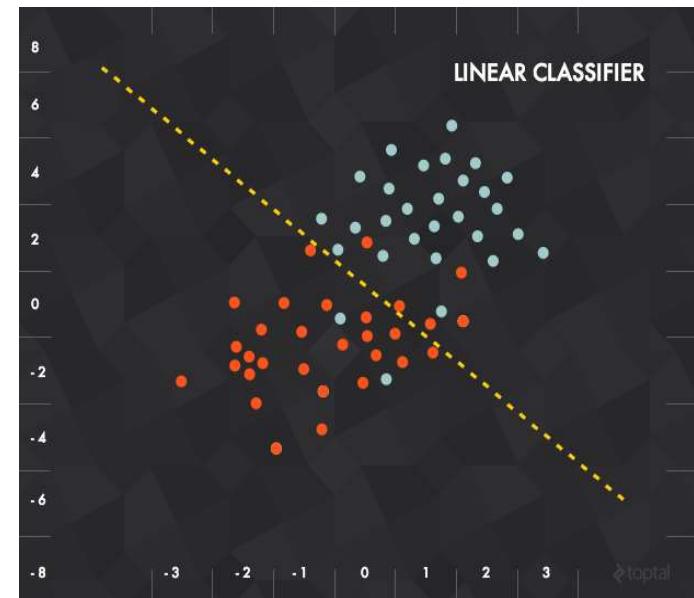
For the case of n inputs, we can rewrite the neuron's function to

$$s(x) = \sum_{i=1}^n x_i \cdot w_i + b = \mathbf{x} \cdot \mathbf{w} + b$$

with $b = -t$. b is known as the perceptron's bias. The result of this function would then be fed into an activation function to produce a labeling

$$f(x) = \begin{cases} 1 & , \text{if } s(x) > 0 \\ 0 & , \text{otherwise} \end{cases}$$

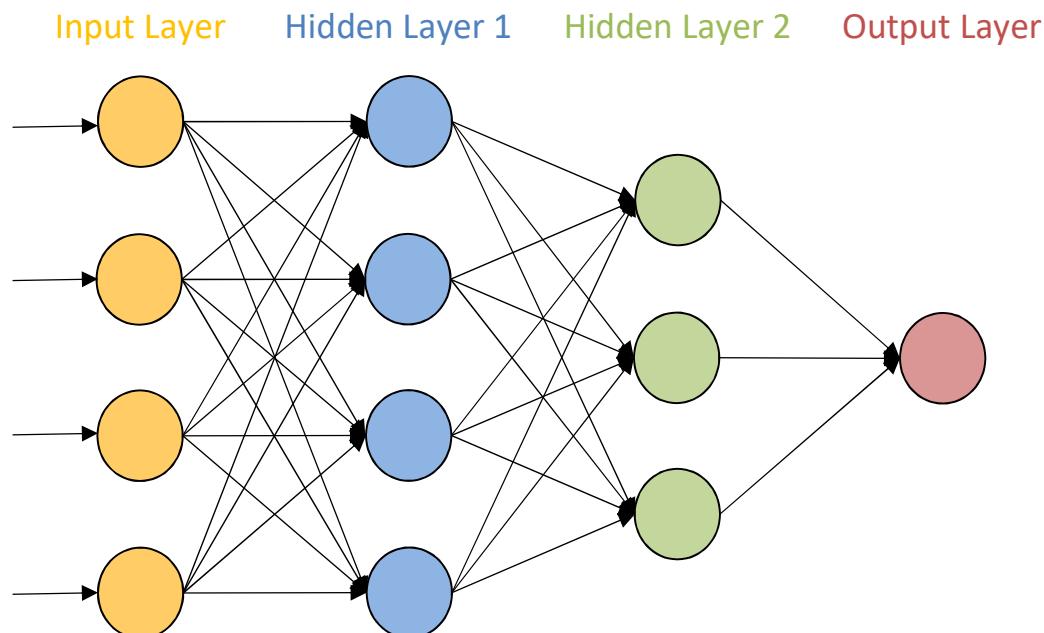
This results in a **linear classifier**. Finally, we have to find the line that best separates the labeled data. The training of the perceptron consists of feeding it multiple training samples and calculating the output for each of them. After each sample, the weights w are adjusted in such a way so as to minimize the output error, defined for example as accuracy or MSE.



Source: <http://www.toptal.com/machine-learning/an-introduction-to-deep-learning-from-perceptrons-to-deep-networks>

The Multilayer Perceptron

The single perceptron approach has a major drawback: it can only learn linear functions. To address this problem, we'll need to use a multilayer perceptron, also known as feedforward neural network. Here, we add layers between the input and the output layer, so-called hidden layers. The hidden layer is where the network stores its internal abstract representation of the training data.

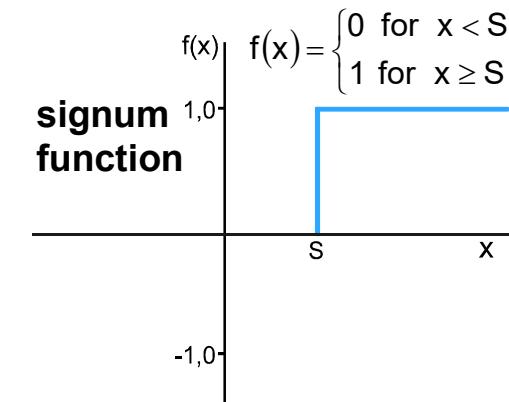
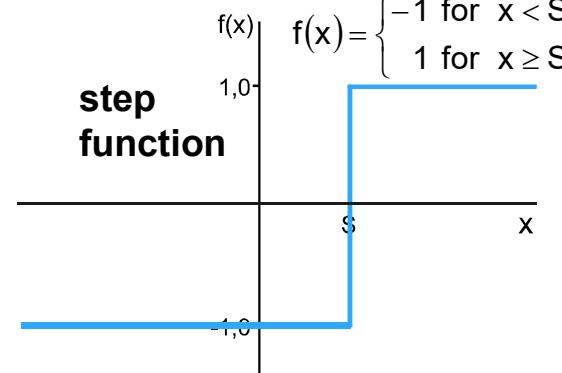
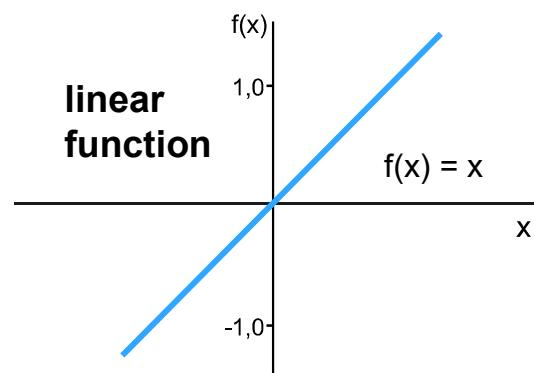


Input Neurons: receive signals from the outer world.

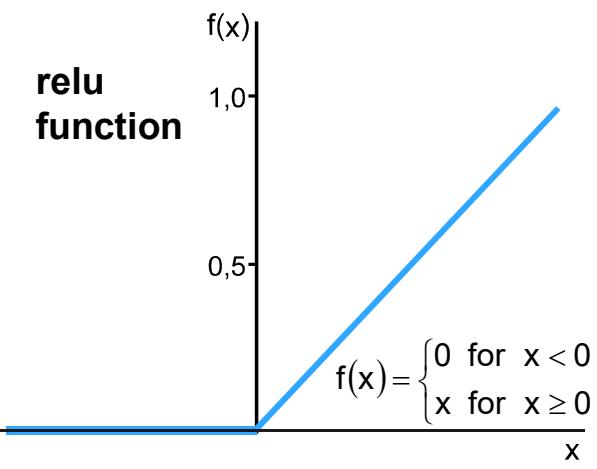
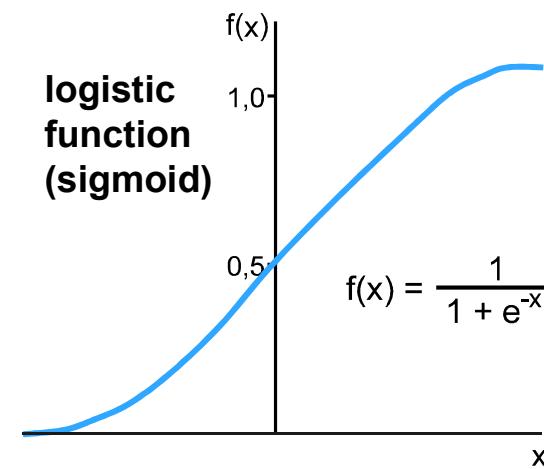
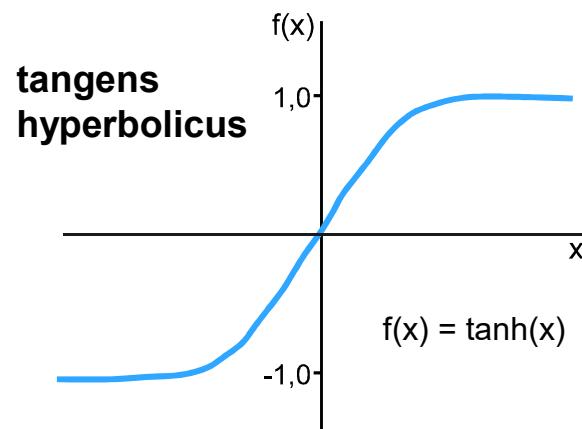
Hidden Neurons: have an internal representation of the outer world.

Output Neurons: pass signals to the outer world.

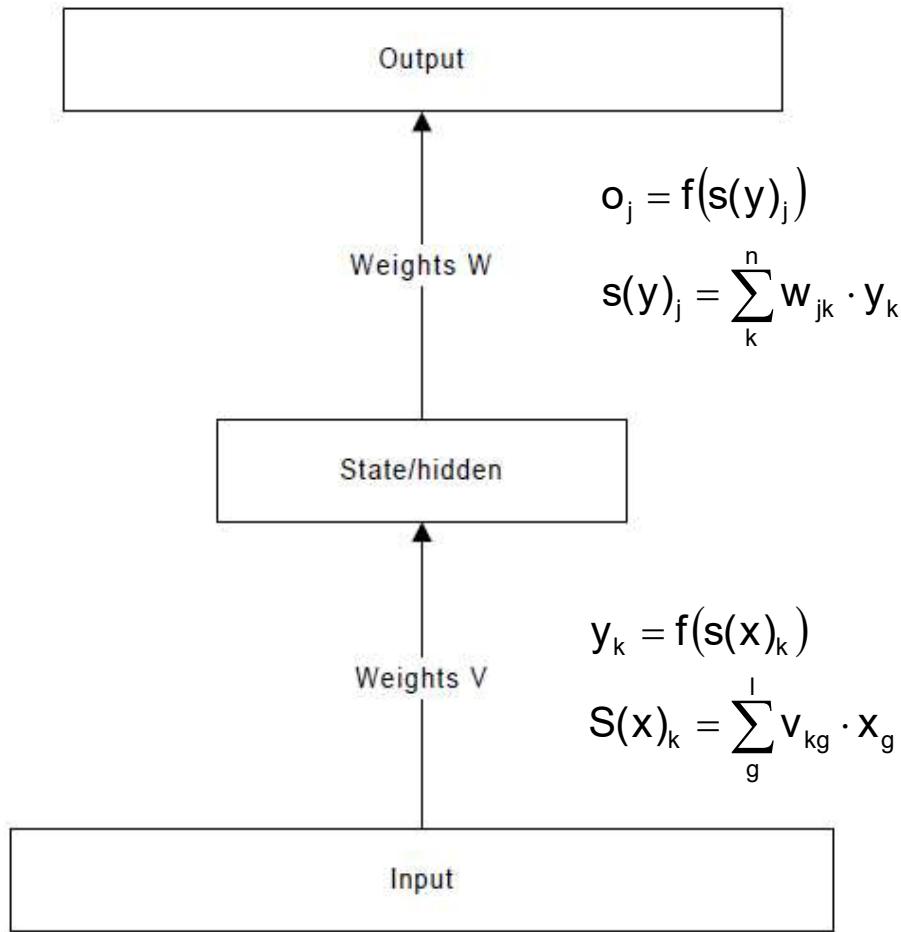
Types of Activation Functions



A linear composition of linear functions is still just a linear function, so most neural networks use non-linear activation functions:



Design of a Feedforward Neural Network



$f()$: Activation Function

n : Number of Hidden Neurons

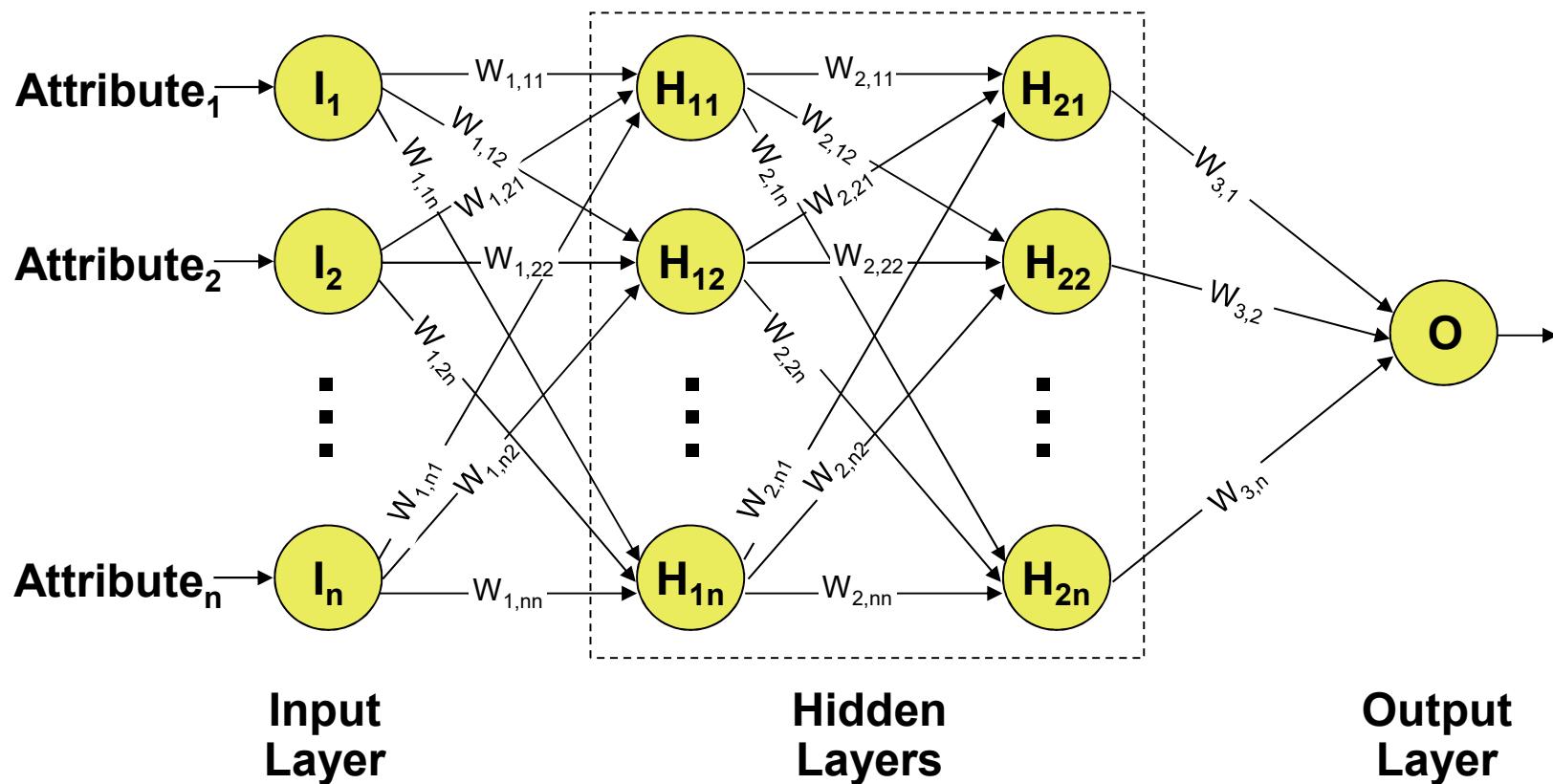
l : Number of Input Neurons

j : Neuron j in the Output Layer

k : Neuron k in the Hidden Layer

g : Neuron g in the Input Layer

Design of a Multilayer Perceptron



Backpropagation

The Backpropagation algorithm is used for calculating the weights. In a training phase, the weights are iteratively calculated using training data sets in such a way that the difference between the calculated and the expected (true) results is minimized. Because the simultaneous calculation of all weights is not possible, they must be found via a learning process. The backpropagation algorithm looks for the minimum of the error function in weight space using the method of gradient descent.

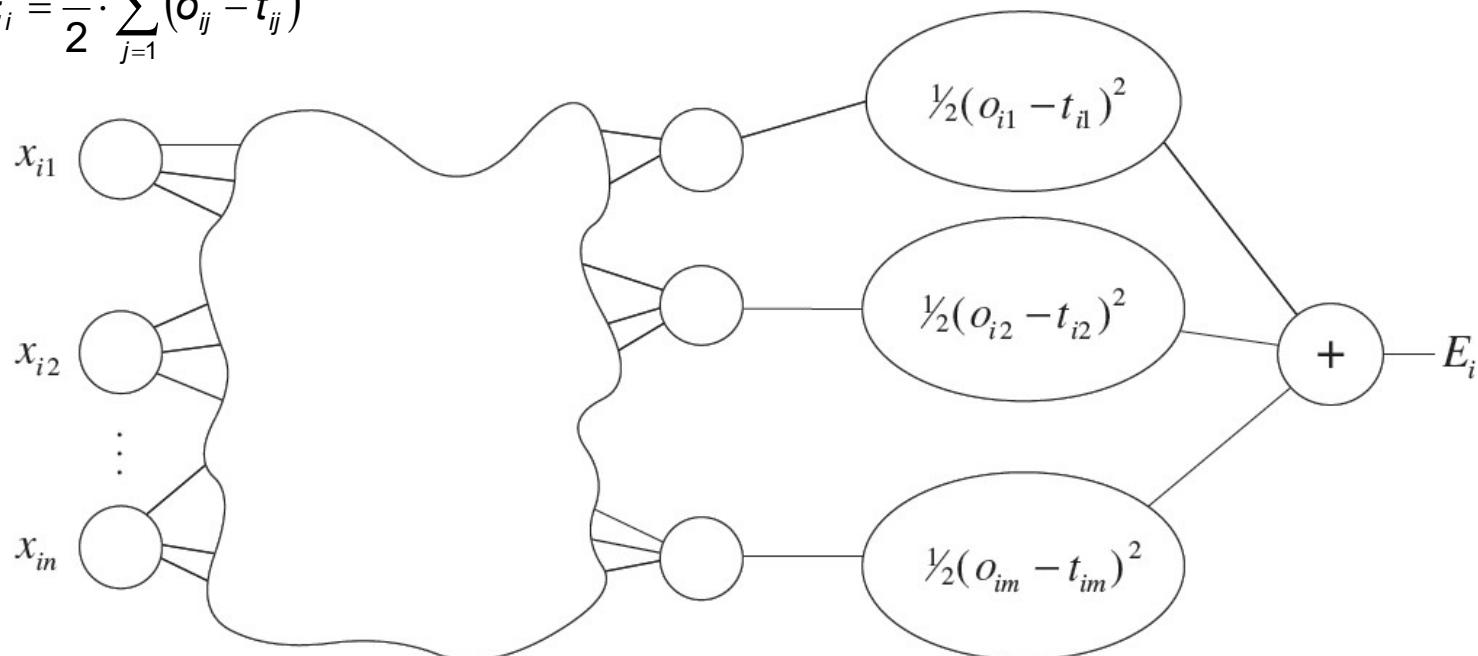
The procedure in principle:

- (1) Define the initial weights
- (2) Put the training set into the input layer
- (3) Calculate the result (value of the output layer) via successive processing one layer after the other
- (4) Compare the output values and target values and calculate the difference
- (5) Iterate steps (2) to (4) for every training set
- (6) Calculate the total error. Adjust the weights beginning with the output layer towards the input layer (backpropagation)
- (7) Iterate steps (2) to (6) until the total error reaches the defined error-level or the number of maximum iterations is reached.

Adjusting the Weights (I)

The error of a training set i is calculated using the quadratic deviation between the values o_{ij} of the neurons of the output layer and their corresponding true values t_{ij} .

$$E_i = \frac{1}{2} \cdot \sum_{j=1}^m (o_{ij} - t_{ij})^2$$



The sum of the errors of all h training objects is the total error value E :

$$E = \sum_{i=1}^h E_i = \sum_{i=1}^h \left(\frac{1}{2} \cdot \sum_{j=1}^m (o_{ij} - t_{ij})^2 \right)$$

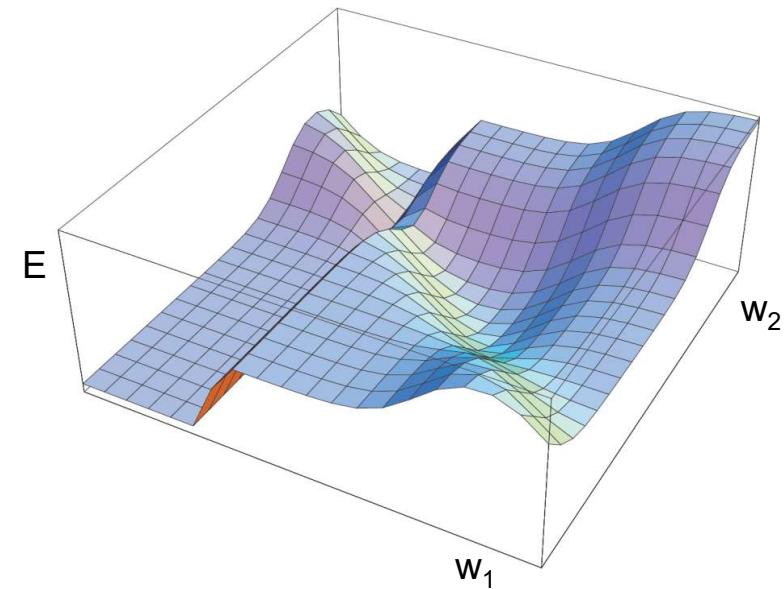
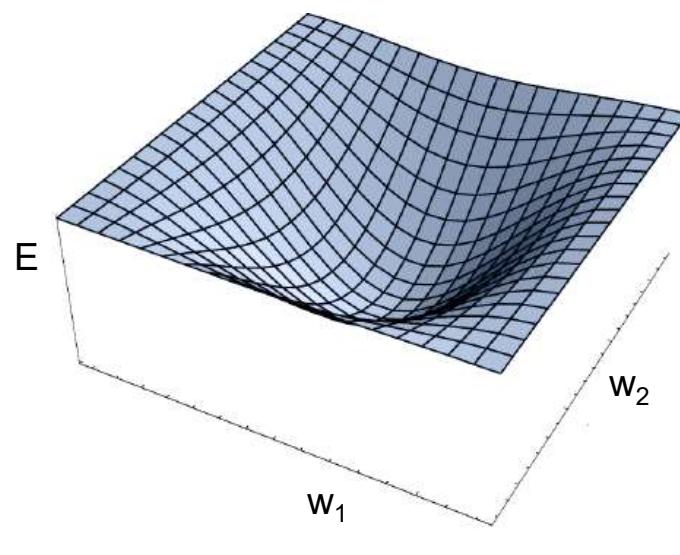
Adjusting the Weights (II)

The function E has to be minimized. Because it depends on the output neurons o_j , it automatically depends on their weights to the precedent layer(s):

$$o_j = f(s(x)_j) \quad \text{with} \quad s(x)_j = \sum_k^n w_{jk} \cdot y_k$$

Thus, the weights have to be found where E is minimal.

Examples of Error functions with two weights:



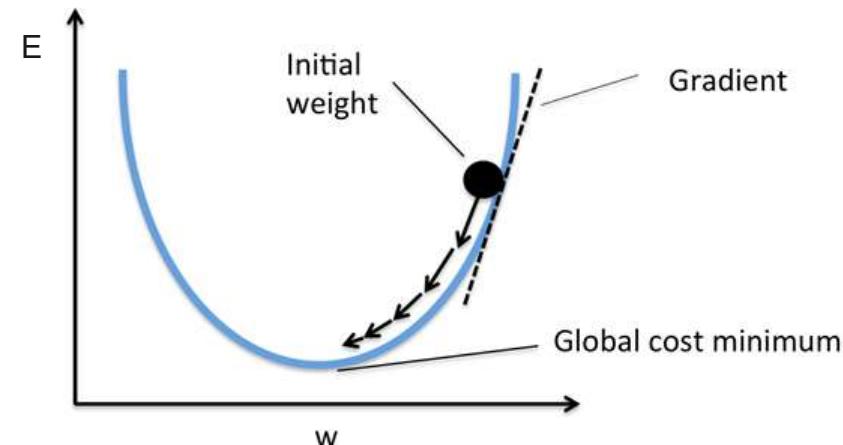
Adjusting the Weights (III)

To minimize the error (cost) function E the backpropagation algorithm uses the method of gradient descent. This method searches those weights, where the vector containing the partial first derivatives of the error function ∇E (gradient) is equal to the zero vector (minimum):

$$\nabla E = \left(\frac{\partial E}{\partial w_1}, \frac{\partial E}{\partial w_2}, \dots, \frac{\partial E}{\partial w_m} \right)$$

To adjust the weight w_{ij} , which connects neurons i to j , the formula is:

$$\begin{aligned}\Delta w_{ij} &= -a \cdot \frac{\partial E}{\partial w_{ij}} \\ &= -a \cdot \sum_{j=1}^m (o_{ij} - t_{ij}) \cdot (-x_i) \\ &= a \cdot \sum_{j=1}^m (o_{ij} - t_{ij}) \cdot x_i\end{aligned}$$



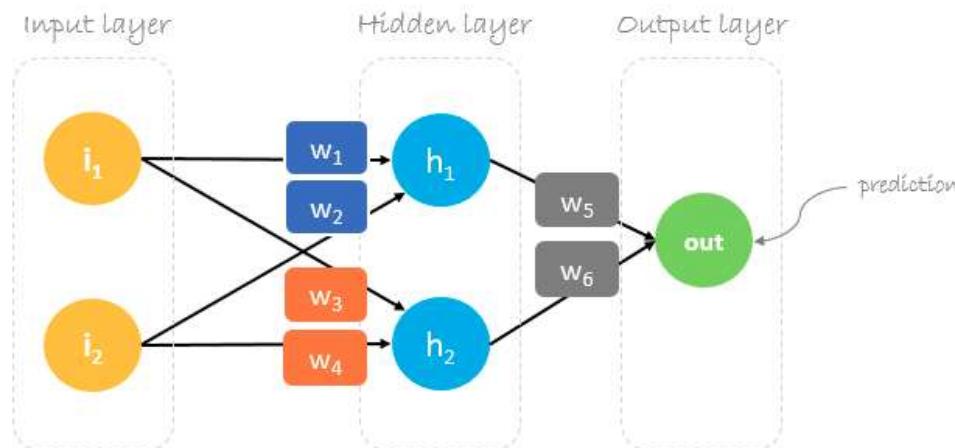
where a represents a predefined learning rate, which defines the step length of each iteration in the negative gradient direction and x_i denote the output value of neuron i .

The adjusted weight is then computed via

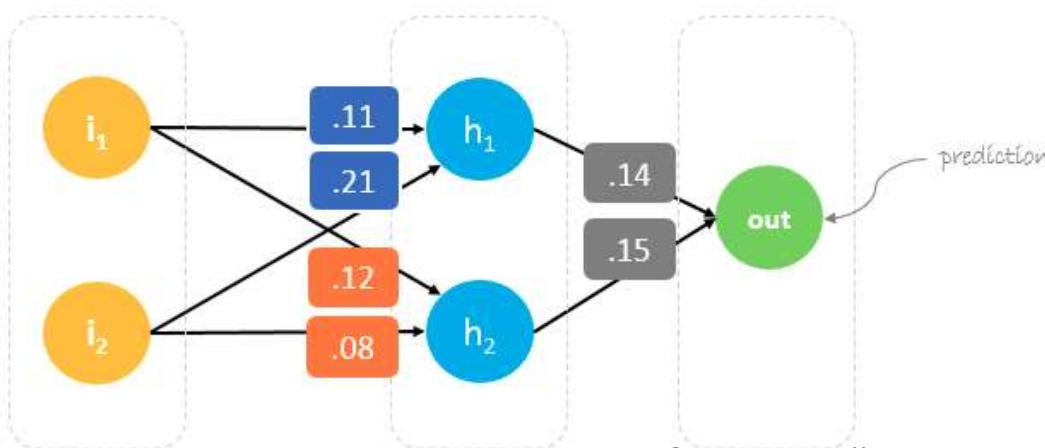
$$w_{ij}^{\text{new}} = w_{ij}^{\text{old}} + \Delta w_{ij}$$

Backpropagation Step by Step (I)

In the following, the backpropagation process will be demonstrated using a simple Neural Network consisting of three layers: Input layer with two inputs neurons, one hidden layer with two neurons, and output layer with a single neuron:

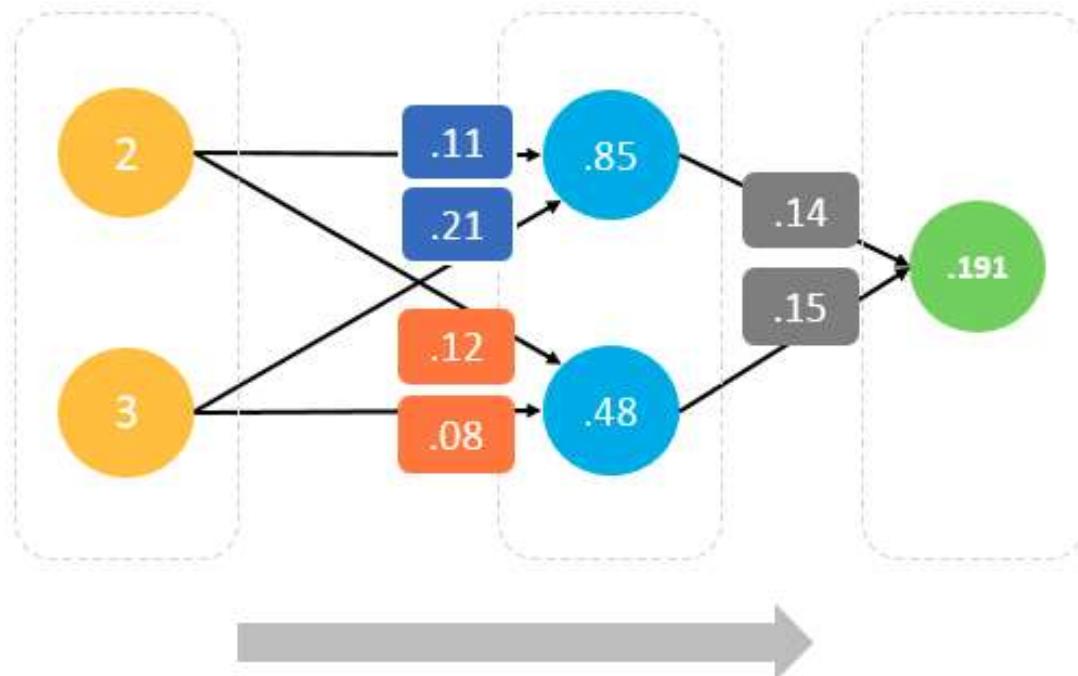


Our initial weights will be: $w_1 = 0.11$, $w_2 = 0.21$, $w_3 = 0.12$, $w_4 = 0.08$, $w_5 = 0.14$ and $w_6 = 0.15$.



Backpropagation Step by Step (II)

Our dataset has one sample with two inputs and one output with the values inputs=[2, 3] and output=[1]. We will use given weights and inputs to predict the output. Inputs are multiplied by weights; the results are then passed forward to next layer:



For reasons of simplification,
no activation function is used
in the neurons.

$$[2 \ 3] \cdot \begin{bmatrix} 0.11 & 0.12 \\ 0.21 & 0.08 \end{bmatrix} \cdot \begin{bmatrix} 0.14 \\ 0.15 \end{bmatrix} = [0.85 \ 0.48] \cdot \begin{bmatrix} 0.14 \\ 0.15 \end{bmatrix} = [0.191]$$

$$2 \times .11 + 3 \times .21 = .85$$

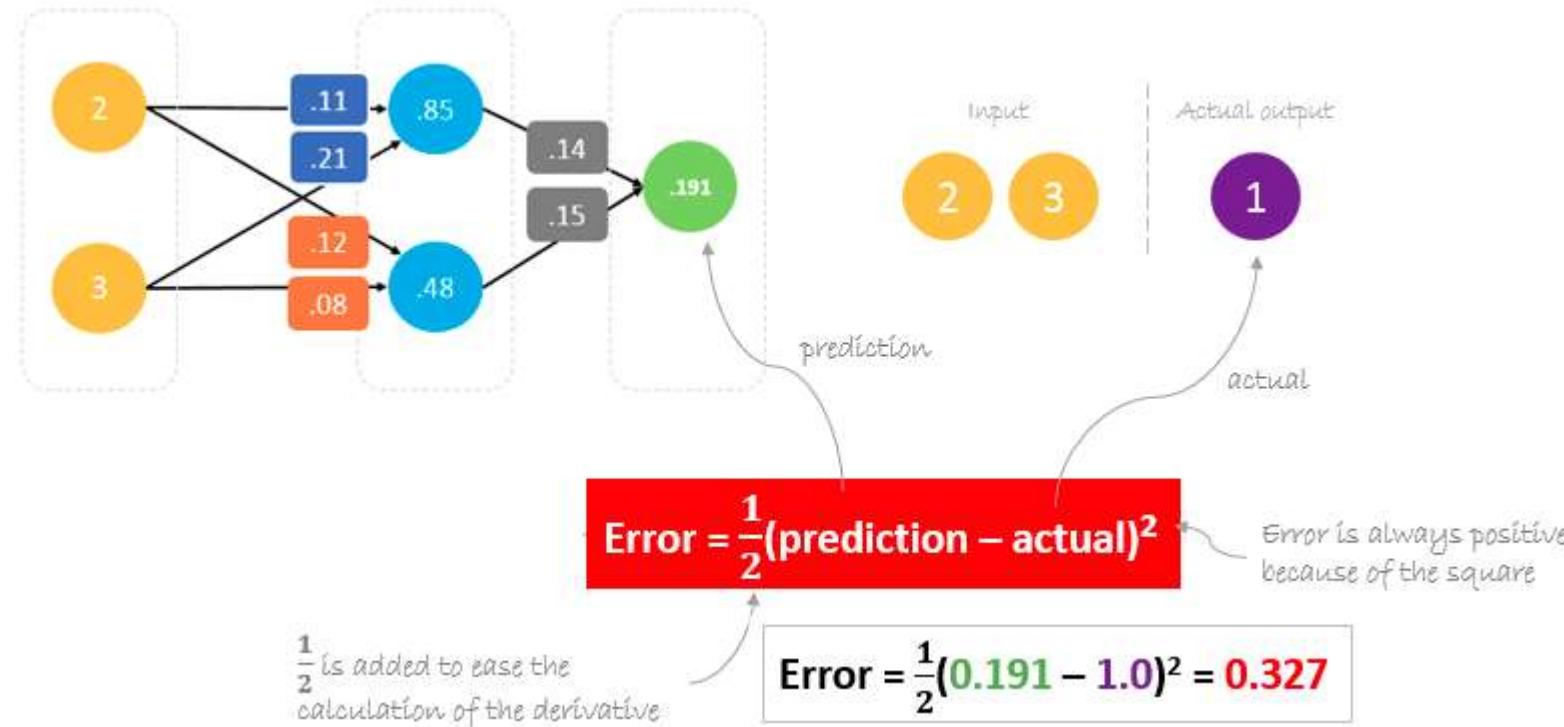
$$.85 \times .14 + .48 \times .15 = .191$$

$$2 \times .12 + 3 \times .08 = .48$$

Source: <http://hmkcode.github.io/ai/backpropagation-step-by-step>

Backpropagation Step by Step (III)

The network output, or prediction, is not even close to actual output. We can calculate the difference or the error as following:



Our main goal of the training is to reduce the error or the difference between prediction and actual output. Since actual output is constant, “not changing”, the only way to reduce the error is to change prediction value. The question now is, how to change prediction value?

Source: <http://hmkcode.github.io/ai/backpropagation-step-by-step>

Backpropagation Step by Step (IV)

By decomposing the prediction into its basic elements we can find that weights are the variable elements affecting prediction value. To change prediction value, we need to adjust the weights:

$$\begin{aligned}
 \text{prediction} &= (\text{h}_1) w_5 + (\text{h}_2) w_6 \\
 &= (\text{i}_1 \text{w}_1 + \text{i}_2 \text{w}_2) w_5 + (\text{i}_1 \text{w}_3 + \text{i}_2 \text{w}_4) w_6
 \end{aligned}$$

We do this using Backpropagation. To find a local minimum of a function using gradient descent, one takes steps proportional to the negative of the gradient of the function at the current point:

$$*W_x = W_x - a \left(\frac{\partial \text{Error}}{\partial W_x} \right)$$

↓ Old weight ↓ Derivative of Error with respect to weight
 New weight ↑ Learning rate

For example, we update w_6 :

$$*W_6 = W_6 - a \left(\frac{\partial \text{Error}}{\partial W_6} \right)$$



We can picture gradient descent optimization as a hiker (the weight coefficient) who wants to climb down a mountain (cost function) into a valley (cost minimum), and each step is determined by the steepness of the slope (gradient) and the leg length of the hiker (learning rate).

Source: <http://hmkcode.github.io/ai/backpropagation-step-by-step>

Backpropagation Step by Step (V)

The derivation of the error function is evaluated by applying the chain rule:

$$\begin{aligned}
 \frac{\partial \text{Error}}{\partial W_6} &= \frac{\partial \text{Error}}{\partial \text{prediction}} * \frac{\partial \text{prediction}}{\partial W_6} \quad \xrightarrow{\text{chain rule}} \\
 &= \frac{\partial \frac{1}{2}(\text{prediction} - \text{actual})^2}{\partial \text{prediction}} * \frac{\partial (\mathbf{i}_1 w_1 + \mathbf{i}_2 w_2) w_5 + (\mathbf{i}_1 w_3 + \mathbf{i}_2 w_4) w_6}{\partial W_6} \\
 &= 2 * \frac{1}{2}(\text{prediction} - \text{actual}) * (\mathbf{i}_1 w_3 + \mathbf{i}_2 w_4) \quad \xleftarrow{\mathbf{h}_2 = \mathbf{i}_1 w_3 + \mathbf{i}_2 w_4} \\
 &= (\text{prediction} - \text{actual}) * (\mathbf{h}_2) \quad \boxed{\Delta = \text{prediction} - \text{actual}} \quad \xleftarrow{\text{delta}} \\
 &= \Delta \mathbf{h}_2
 \end{aligned}$$

To update w_6 we can apply the following formula:

$$*W_6 = W_6 - a \Delta h_2$$

Similarly, we can derive the update formula for w_5 and any other weights existing between the output and the hidden layer:

$$*W_5 = W_5 - a \Delta h_1$$

Source: <http://hmkcode.github.io/ai/backpropagation-step-by-step>

Backpropagation Step by Step (VI)

When moving backward to update w_1 , w_2 , w_3 and w_4 existing between input and hidden layer, the partial derivative for the error function with respect to w_1 , for example, will be as following:

$$\frac{\partial \text{Error}}{\partial w_1} = \frac{\partial \text{Error}}{\partial \text{prediction}} * \frac{\partial \text{prediction}}{\partial h_1} * \frac{\partial h_1}{\partial w_1} \quad \text{chain rule}$$

$$\text{Error} = \frac{1}{2}(\text{prediction} - \text{actual})^2$$

$$= \frac{\partial \frac{1}{2}(\text{prediction} - \text{actual})^2}{\partial \text{prediction}} * \frac{\partial (h_1) w_5 + (h_2) w_6}{\partial h_1} * \frac{\partial i_1 w_1 + i_2 w_2}{\partial w_1}$$

$$\text{prediction} = (h_1) w_5 + (h_2) w_6$$

$$h_1 = i_1 w_1 + i_2 w_2$$

$$= 2 * \frac{1}{2}(\text{prediction} - \text{actual}) * (w_5) * (i_1)$$

$$\Delta = \text{prediction} - \text{actual} \quad \text{delta}$$

$$= \Delta w_5 i_1$$

We can find the update formula for the remaining weights w_2 , w_3 and w_4 in the same way.

Source: <http://hmkcode.github.io/ai/backpropagation-step-by-step>

Backpropagation Step by Step (VII)

In summary, the update formulas for all weights will be:

$$* \mathbf{w}_6 = \mathbf{w}_6 - a (\Delta h_2)$$

$$* \mathbf{w}_5 = \mathbf{w}_5 - a (\Delta h_1)$$

$$* \mathbf{w}_4 = \mathbf{w}_4 - a (i_2 \cdot \Delta w_6)$$

$$* \mathbf{w}_3 = \mathbf{w}_3 - a (i_1 \cdot \Delta w_6)$$

$$* \mathbf{w}_2 = \mathbf{w}_2 - a (i_2 \cdot \Delta w_5)$$

$$* \mathbf{w}_1 = \mathbf{w}_1 - a (i_1 \cdot \Delta w_5)$$

We can rewrite the update formulas in matrices:

$$* \begin{bmatrix} \mathbf{w}_5 \\ \mathbf{w}_6 \end{bmatrix} = \begin{bmatrix} \mathbf{w}_5 \\ \mathbf{w}_6 \end{bmatrix} - a \Delta \begin{bmatrix} h_1 \\ h_2 \end{bmatrix} = \begin{bmatrix} \mathbf{w}_5 \\ \mathbf{w}_6 \end{bmatrix} - \begin{bmatrix} a \Delta h_1 \\ a \Delta h_2 \end{bmatrix}$$

$$* \begin{bmatrix} \mathbf{w}_1 & \mathbf{w}_3 \\ \mathbf{w}_2 & \mathbf{w}_4 \end{bmatrix} = \begin{bmatrix} \mathbf{w}_1 & \mathbf{w}_3 \\ \mathbf{w}_2 & \mathbf{w}_4 \end{bmatrix} - a \Delta \begin{bmatrix} i_1 \\ i_2 \end{bmatrix} \cdot \begin{bmatrix} \mathbf{w}_5 & \mathbf{w}_6 \end{bmatrix} = \begin{bmatrix} \mathbf{w}_1 & \mathbf{w}_3 \\ \mathbf{w}_2 & \mathbf{w}_4 \end{bmatrix} - \begin{bmatrix} a i_1 \Delta w_5 & a i_1 \Delta w_6 \\ a i_2 \Delta w_5 & a i_2 \Delta w_6 \end{bmatrix}$$

Source: <http://hmkcode.github.io/ai/backpropagation-step-by-step>

Backpropagation Step by Step (VIII)

With the derived formulas we can now adjust the weights:

$$\Delta = 0.191 - 1 = -0.809 \quad \text{← Delta} = \text{prediction} - \text{actual}$$

$$a = 0.05$$

$$* \begin{bmatrix} w_5 \\ w_6 \end{bmatrix} = \begin{bmatrix} 0.14 \\ 0.15 \end{bmatrix} - 0.05(-0.809) \begin{bmatrix} 0.85 \\ 0.48 \end{bmatrix} = \begin{bmatrix} 0.14 \\ 0.15 \end{bmatrix} - \begin{bmatrix} -0.034 \\ -0.019 \end{bmatrix} = \begin{bmatrix} 0.17 \\ 0.17 \end{bmatrix}$$

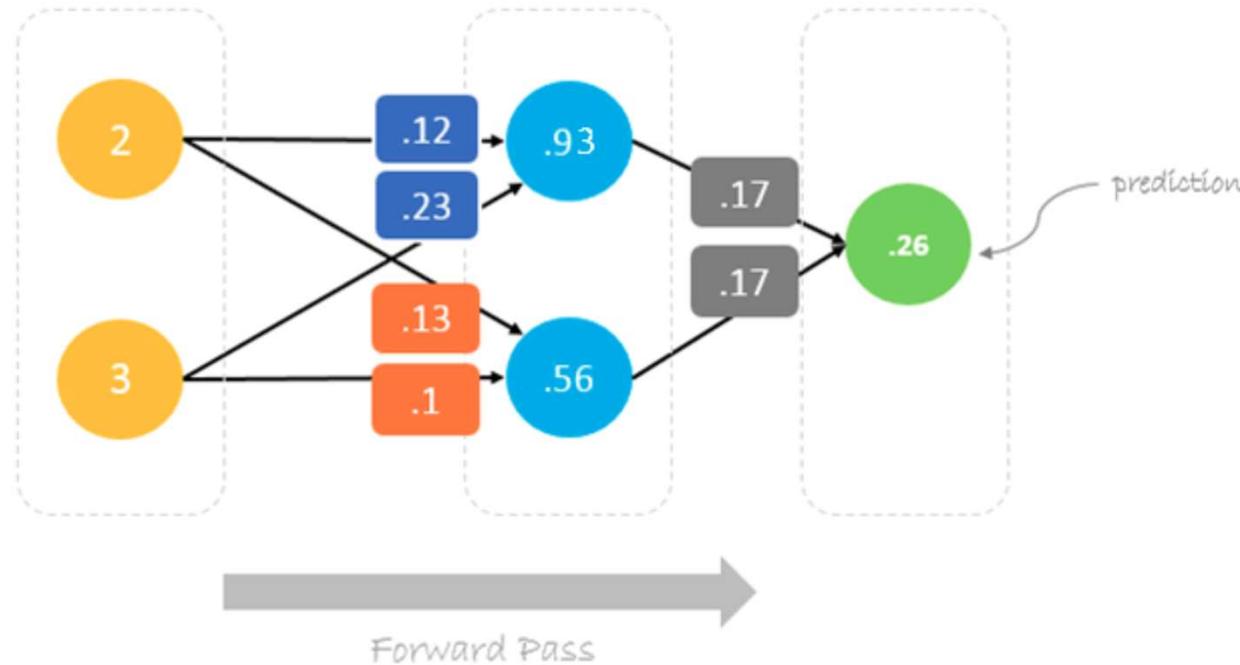
$$* \begin{bmatrix} w_1 & w_3 \\ w_2 & w_4 \end{bmatrix} = \begin{bmatrix} .11 & .12 \\ .21 & .08 \end{bmatrix} - 0.05(-0.809) \begin{bmatrix} 2 \\ 3 \end{bmatrix} \cdot \begin{bmatrix} 0.14 & 0.15 \end{bmatrix}$$

$$= \begin{bmatrix} .11 & .12 \\ .21 & .08 \end{bmatrix} - \begin{bmatrix} -0.011 & -0.012 \\ -0.017 & -0.018 \end{bmatrix} = \begin{bmatrix} .12 & .13 \\ .23 & .10 \end{bmatrix}$$

Source: <http://hmkcode.github.io/ai/backpropagation-step-by-step>

Backpropagation Step by Step (IX)

... and use the new weights to recalculate the example:



$$\begin{bmatrix} 2 & 3 \end{bmatrix} \cdot \begin{bmatrix} 0.12 & 0.13 \\ 0.23 & 0.10 \end{bmatrix} = \begin{bmatrix} 0.93 & 0.56 \end{bmatrix} \cdot \begin{bmatrix} 0.17 \\ 0.17 \end{bmatrix} = [0.26]$$

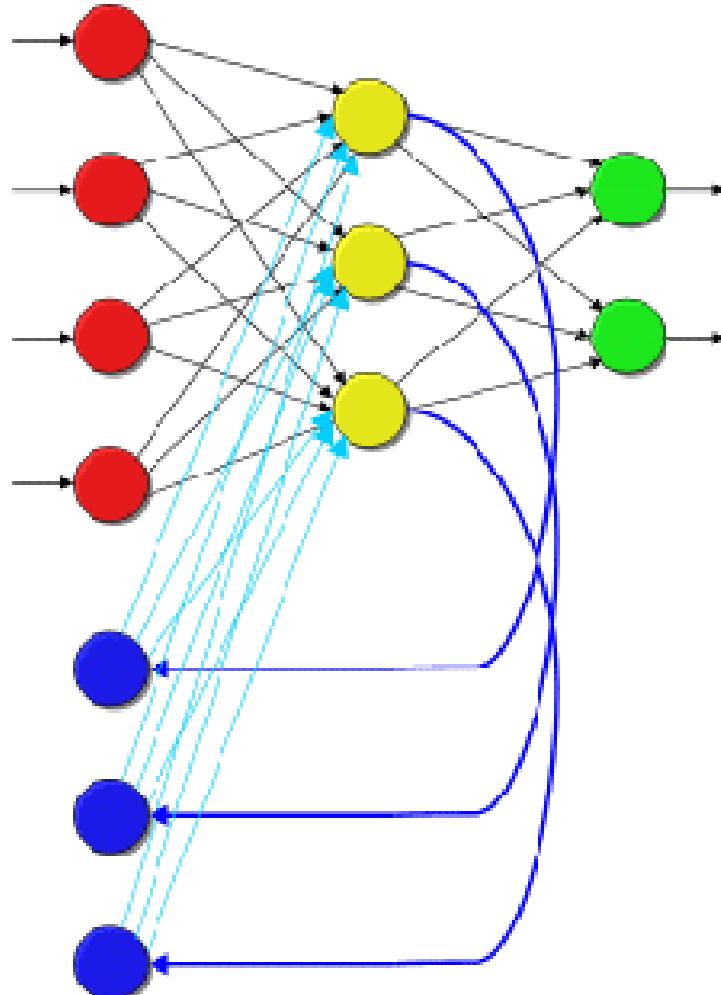
$$2 \times 0.12 + 3 \times 0.23 = 0.93$$

$$0.93 \times 0.17 + 0.56 \times 0.17 = 0.26$$

$$2 \times 0.13 + 3 \times 0.10 = 0.48$$

The new prediction 0.26 is bit closer to the output than the previously predicted one 0.191. We repeat now the same process until error is close or equal to zero.

Design of a Recurrent Neural Network (I)

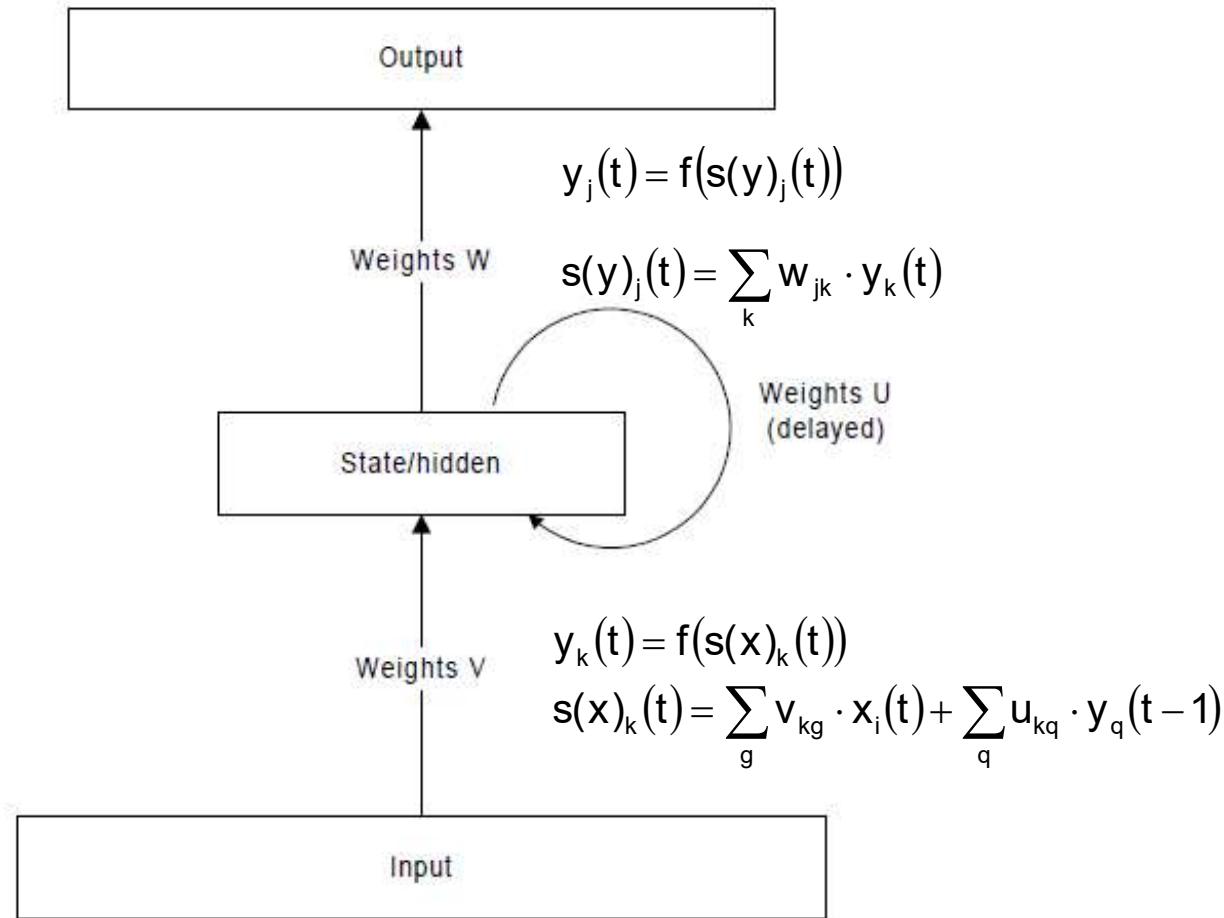


Context Neurons: receive signals from previous iterations.

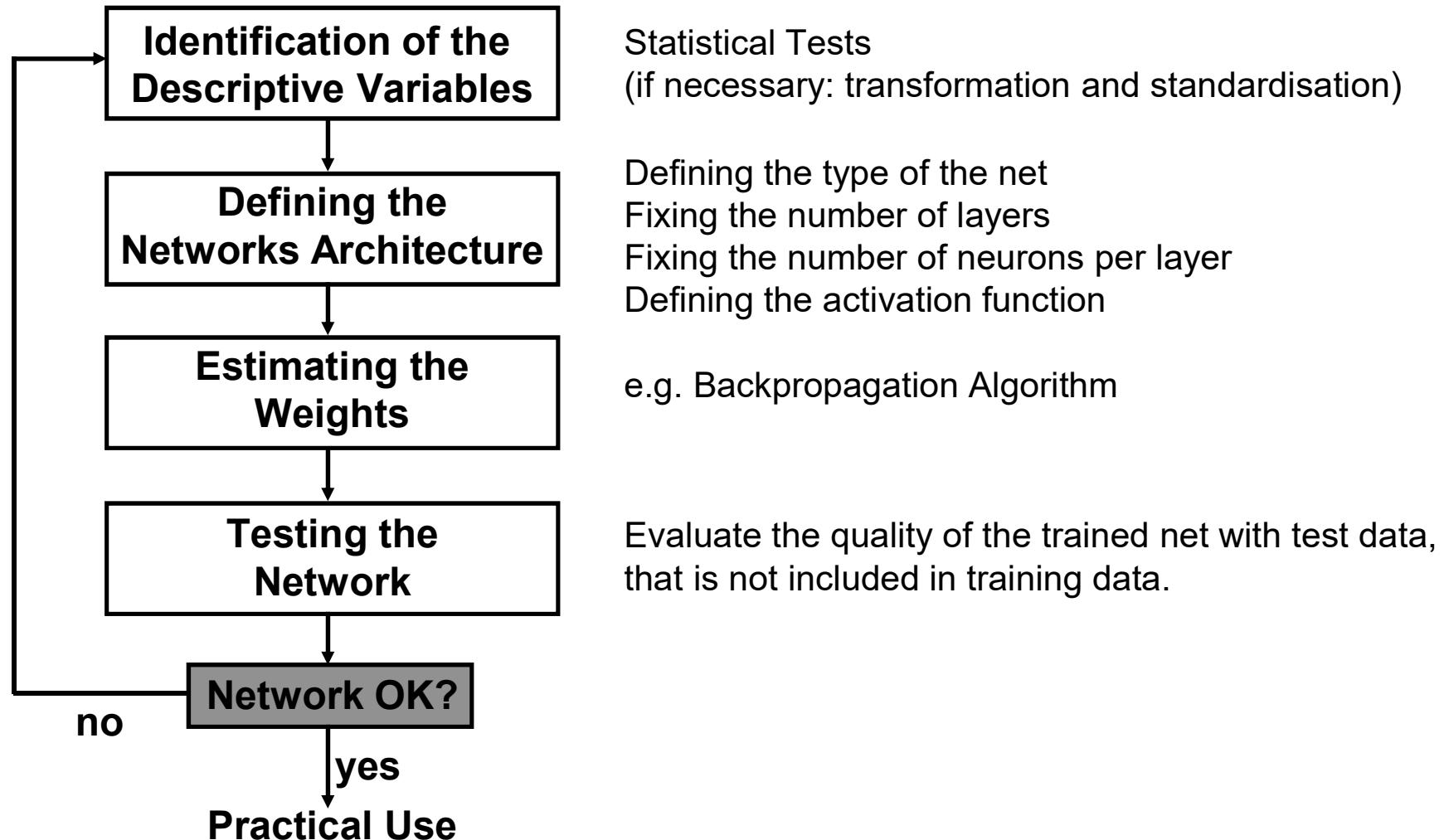
Recurrent neural networks are fundamentally different from feedforward architectures in the sense that they not only operate on an input space but also on an internal state space.

The state space enables the representation (and learning) of temporally/sequentially extended dependencies over unspecified intervals.

Design of a Recurrent Neural Network (II)

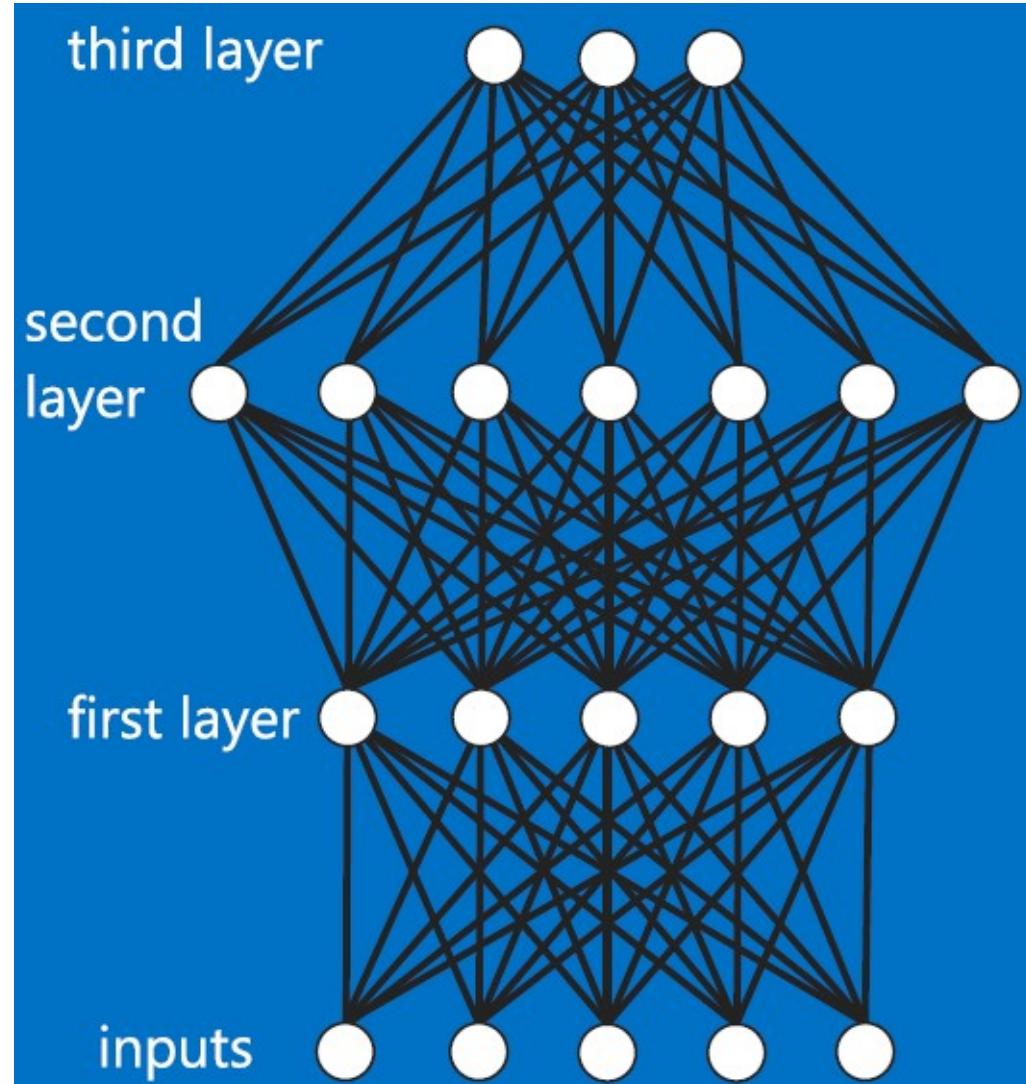


Network Modelling



Deep Learning (I)

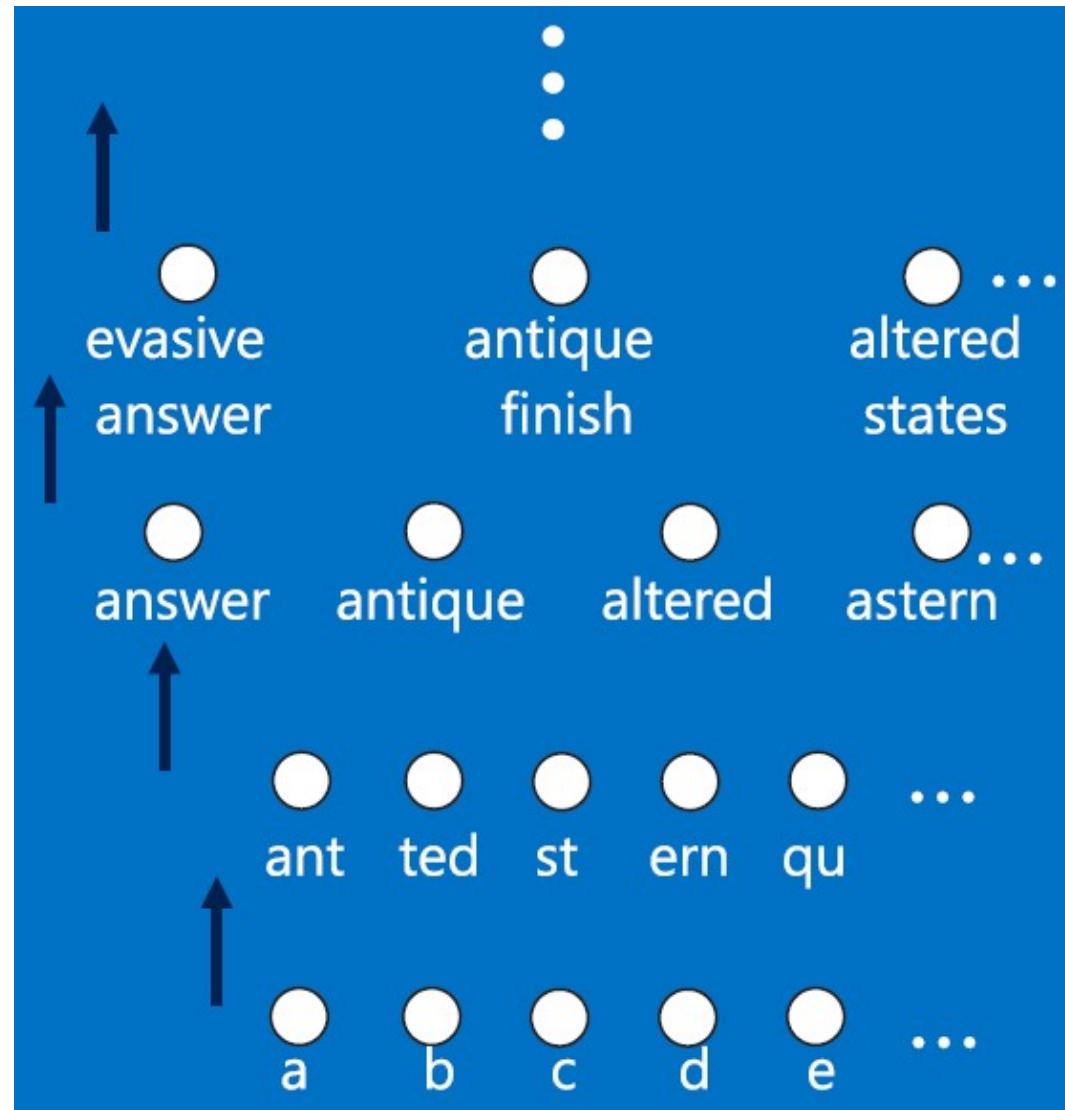
If a network has more layers, it's deep.



Deep Learning (II)

Each node represents a pattern, a combination of the neurons on the previous layer.

Patterns get more complex as the number of layers goes up.



Source: <http://github.com/brohrer/publications/blob/master/Rohrer16DeepLearningDemystified.pdf>
Introduction to Data Analytics in Business

The Problem with Large Networks

A neural network can have more than one hidden layer: in that case, the higher layers are "building" new abstractions on top of previous layers. Larger networks are often more suitable in complex application areas.

But increasing the number of hidden layers leads to two known issues:

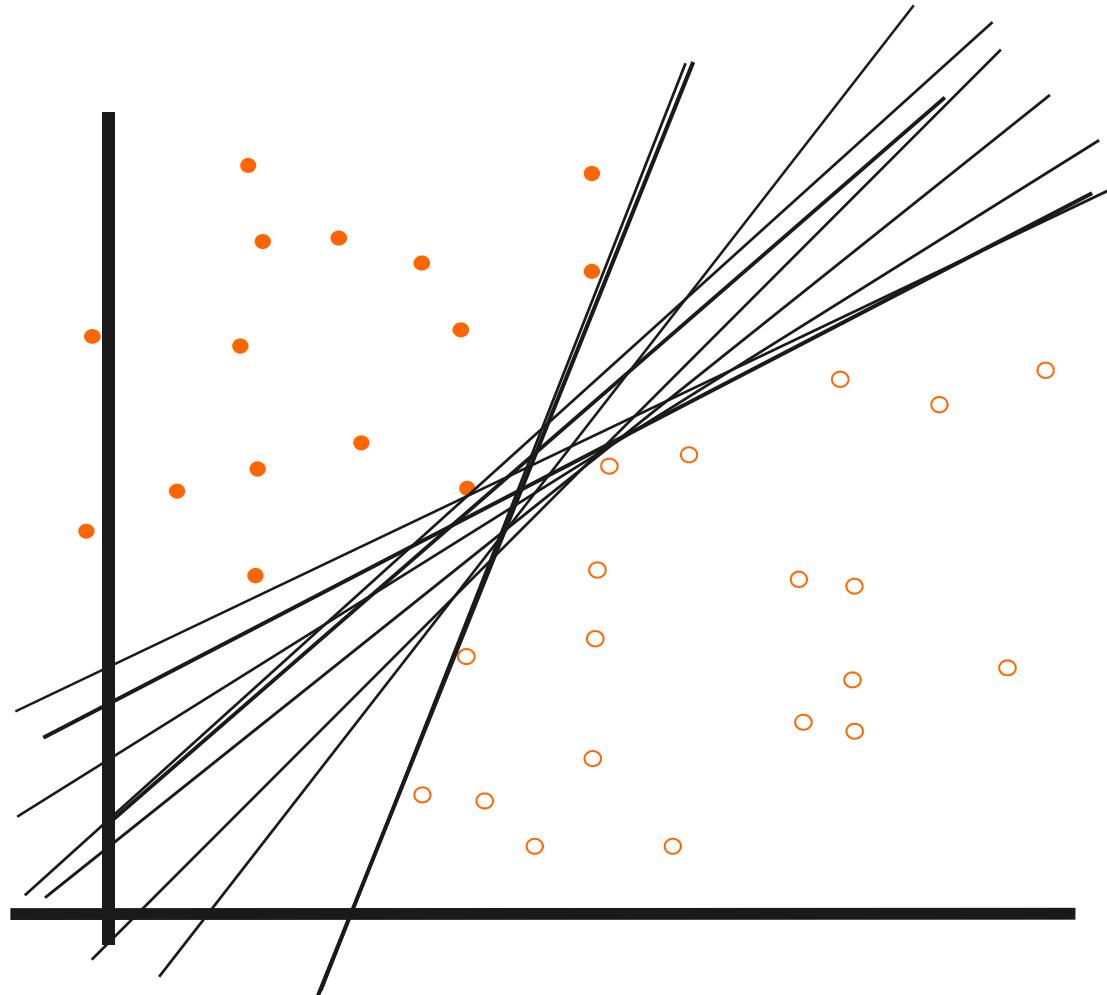
1. **Vanishing gradients:** as we add more and more hidden layers, back-propagation becomes less and less useful in passing information to the lower layers. In effect, as information is passed back, the gradients begin to vanish and become small relative to the weights of the networks.
2. **Overfitting:** the learner ends up fitting the training data really well, but will perform much, much more poorly on real examples.

Many fixes and workarounds have been proposed and investigated, such as alternate weight initialization schemes, unsupervised pre-training, layer-wise training, and variations on gradient descent. One of the most important change is the use of a new activation function ReLU instead of the sigmoid activation function.

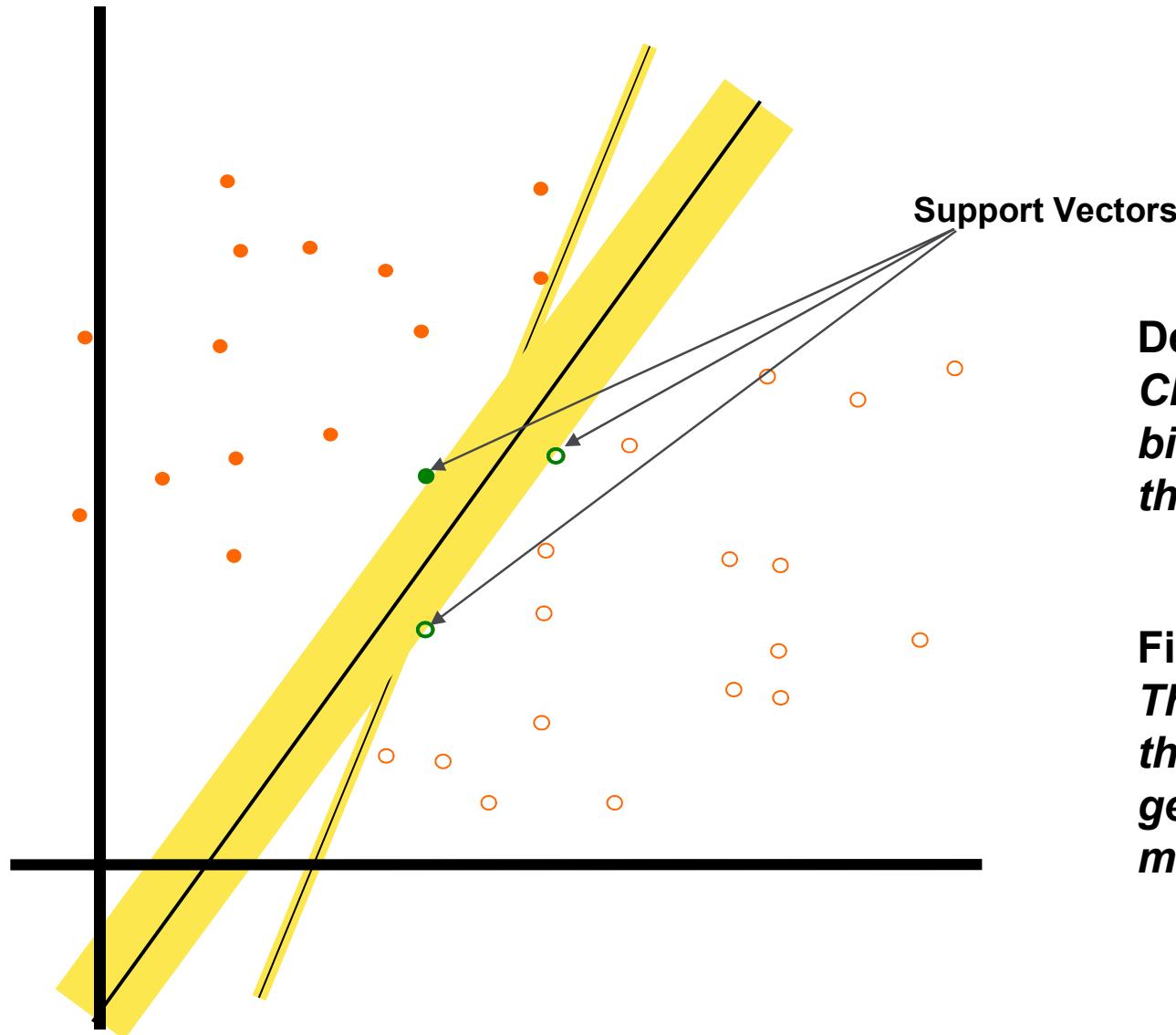
Source: <http://www.toptal.com/machine-learning/an-introduction-to-deep-learning-from-perceptrons-to-deep-networks>

Support Vector Machines

Which one is the optimal Separation Line?



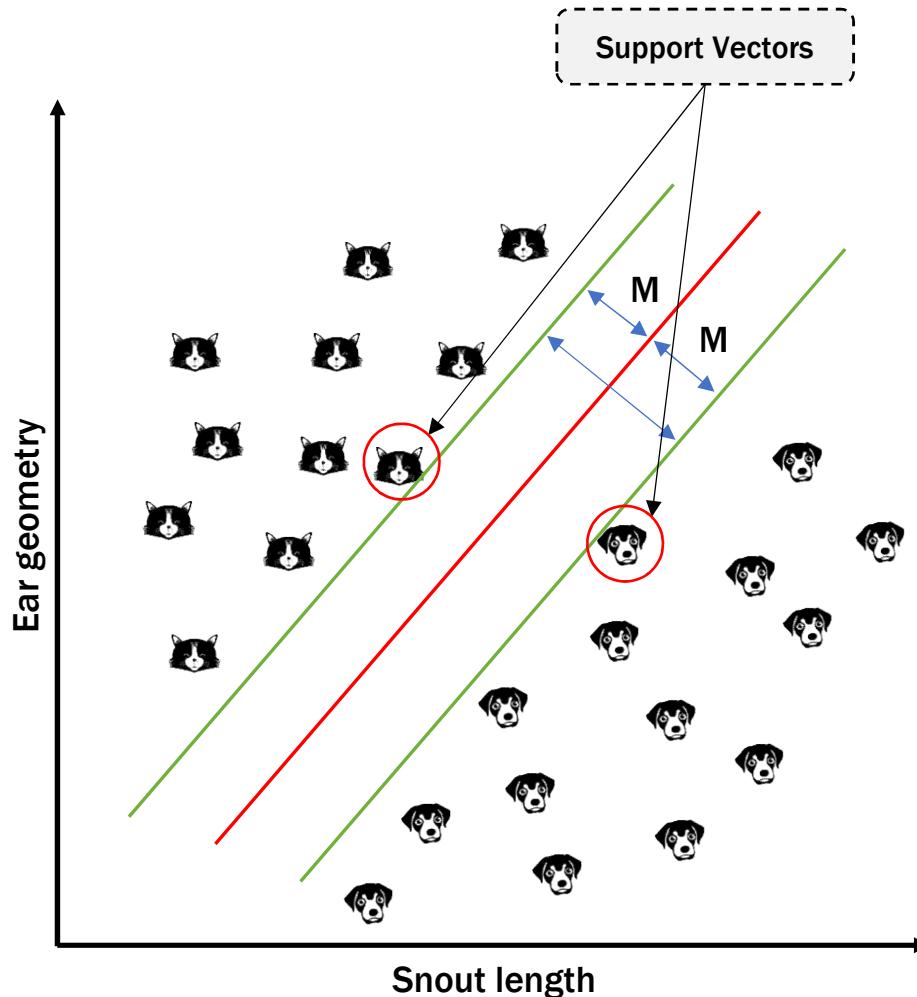
Optimal Separation Line and Support Vectors



Decision criterion:
Chose the line with the biggest distance towards the next objects!

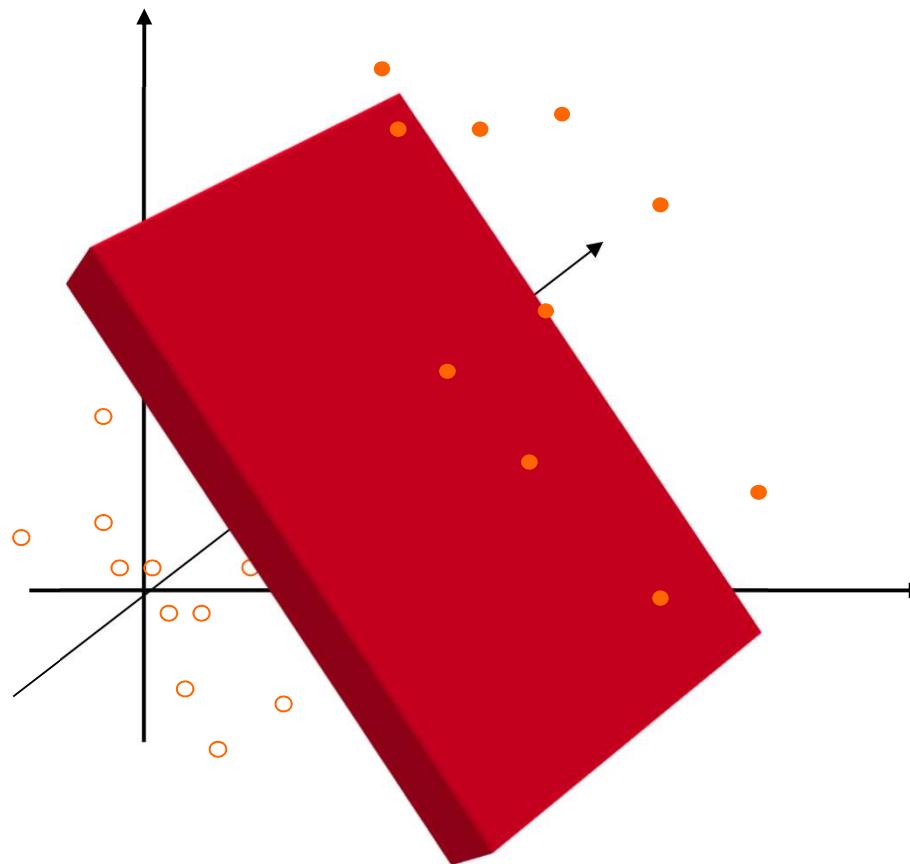
Finding:
The higher the width of the margin the better the generalization of the model!

Example Application



The separating hyperplane

In the multidimensional case, that is if there are more than two features, not a straight line but a hyperplane constitutes the separation medium.

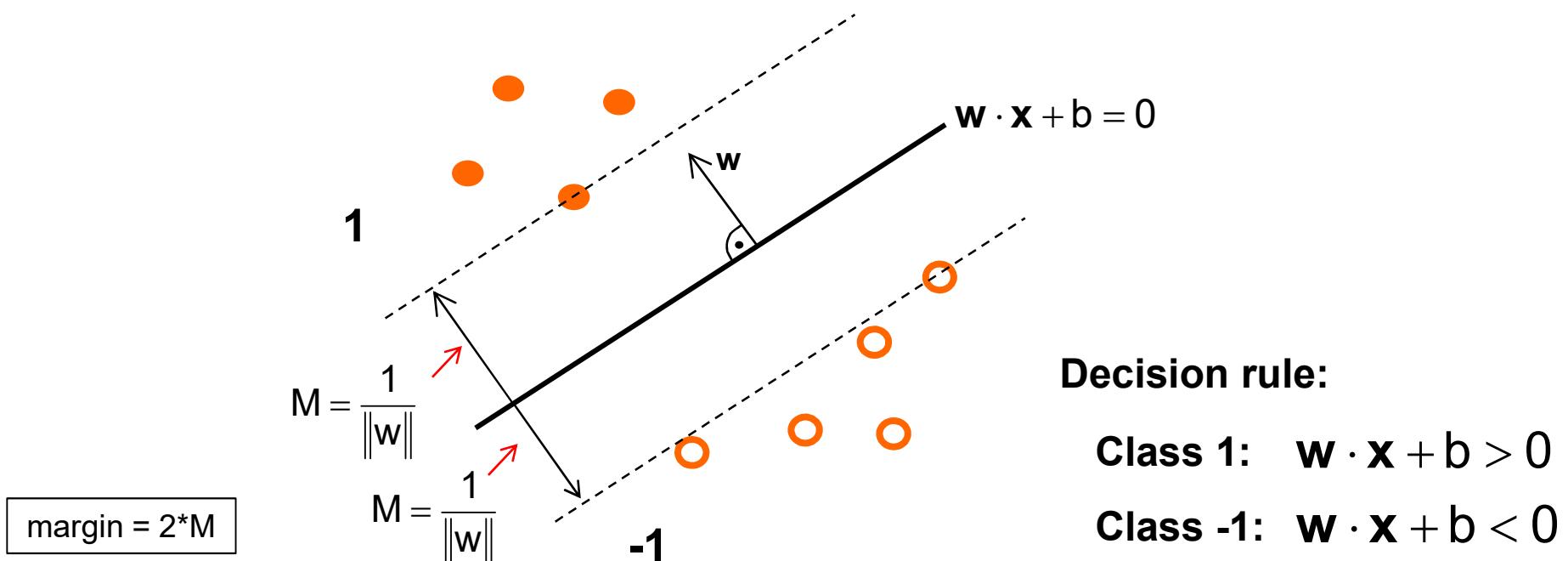


Description of the separating hyperplane

The separating plane can be described through the equation:

$$\mathbf{w} \cdot \mathbf{x} + b = 0$$

w constitutes the weights vector, the dot '·' the scalar product, x the feature vector and b a scalar, which expresses the adjustment in the space.



Estimating the separating hyperplane

During the training procedure, w and b are selected so that the hyperplane with the widest margin is found. This results in the following optimization problem

Objective Function: $M \rightarrow \text{Max!}$

Constraint: $y_i \cdot (w \cdot x_i + b) \geq M$

with y as class attribute (-1, 1).

Because of $M = \frac{1}{\|w\|}$ this problem can be more conveniently rephrased as

Objective Function: $\frac{1}{2} \|w\|^2 \rightarrow \text{Min!}$

Constraint: $y_i \cdot (w \cdot x_i + b) \geq 1$

with y as class attribute (-1, 1).

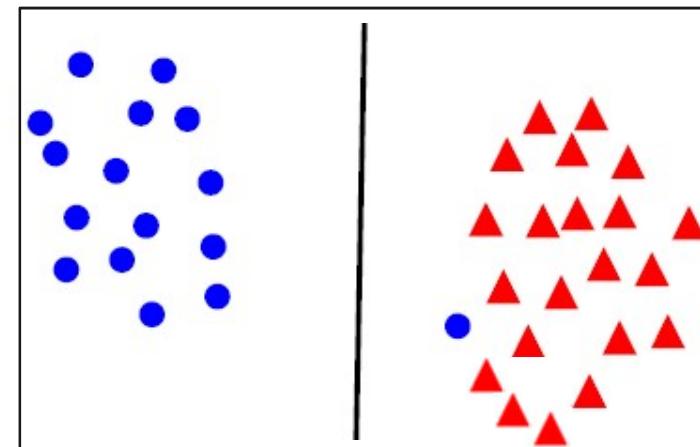
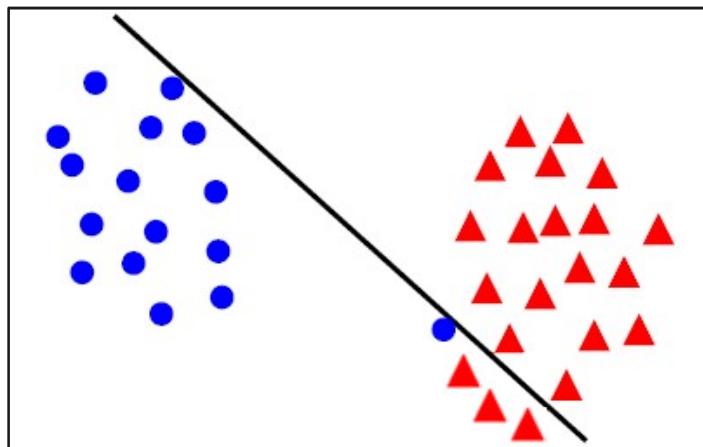
Ambiguous separability (I)

Using a SVM one searches for two things:

- a hyperplane with the widest margin and
- a hyperplane that correctly separates as many instances as possible.

The problem is that you will not always be able to get both things.

Example:



There is a trade off between the margin and the number of mistakes on the training data.

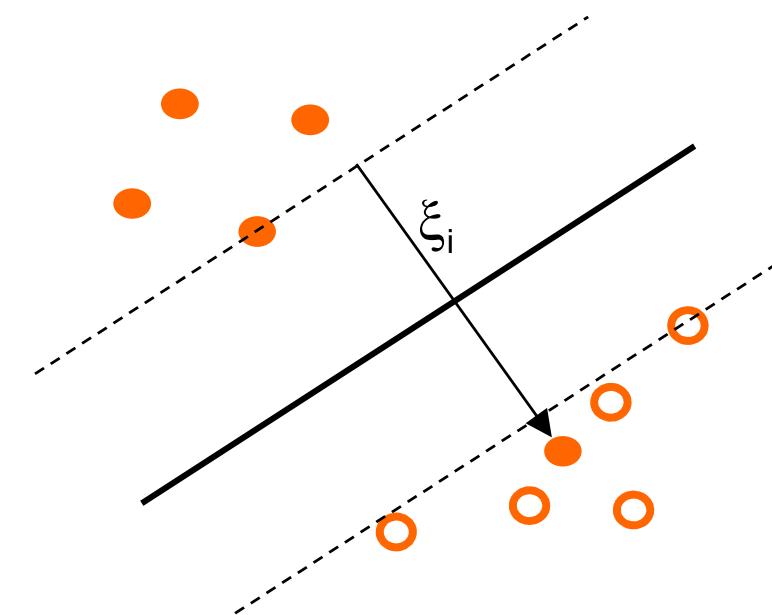
Ambiguous separability (II)

If an exact separability is not advantageous or does not exist, the classification errors (defined as distance ξ to the hyperplane) are included into the minimization problem by an error weight C (Penalty Parameter).

Objective Function: $\frac{1}{2} \|\mathbf{w}\|^2 + C \cdot \sum_{i=1}^n \xi_i \rightarrow \text{Min!}$

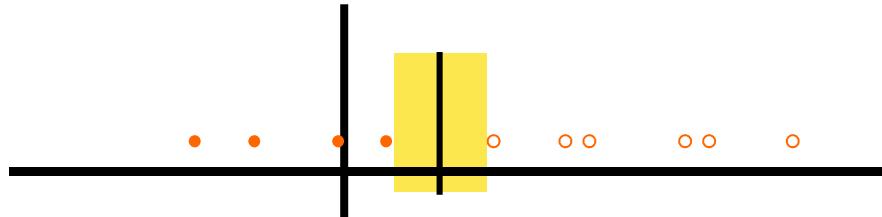
Constraints: $y_i \cdot (\mathbf{w} \cdot \mathbf{x}_i + b) \geq 1 - \xi_i$

$$\xi_i \geq 0$$

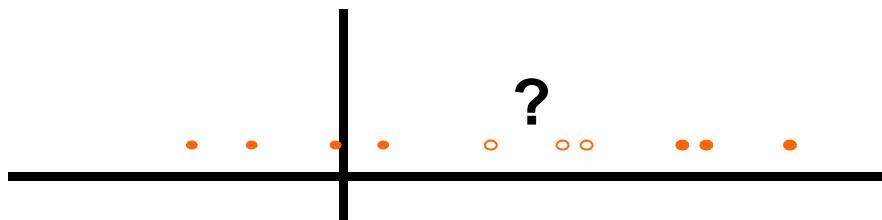


Nonlinearity (I)

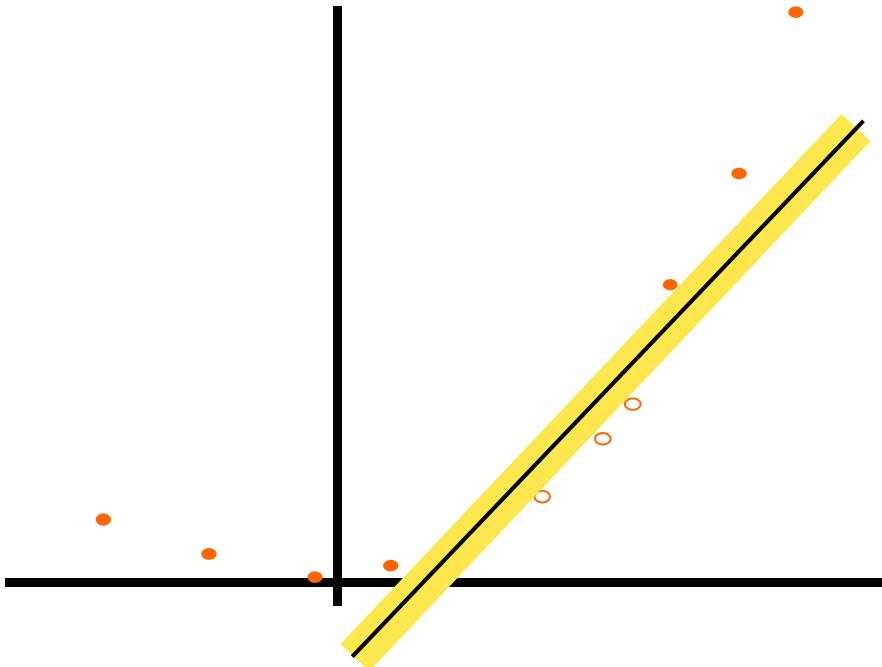
The linear case:



The nonlinear case:



The trick:



Nonlinearity (II)

While other approaches use nonlinear functions in nonlinear cases, Support Vector Machines project the original data into another dimensional space in order to construct the linear classification function there.

The mapping into other dimensional spaces, also named "Feature Space F", is done using a so-called kernel function:

$$\Phi : \mathbb{R}^N \rightarrow F$$

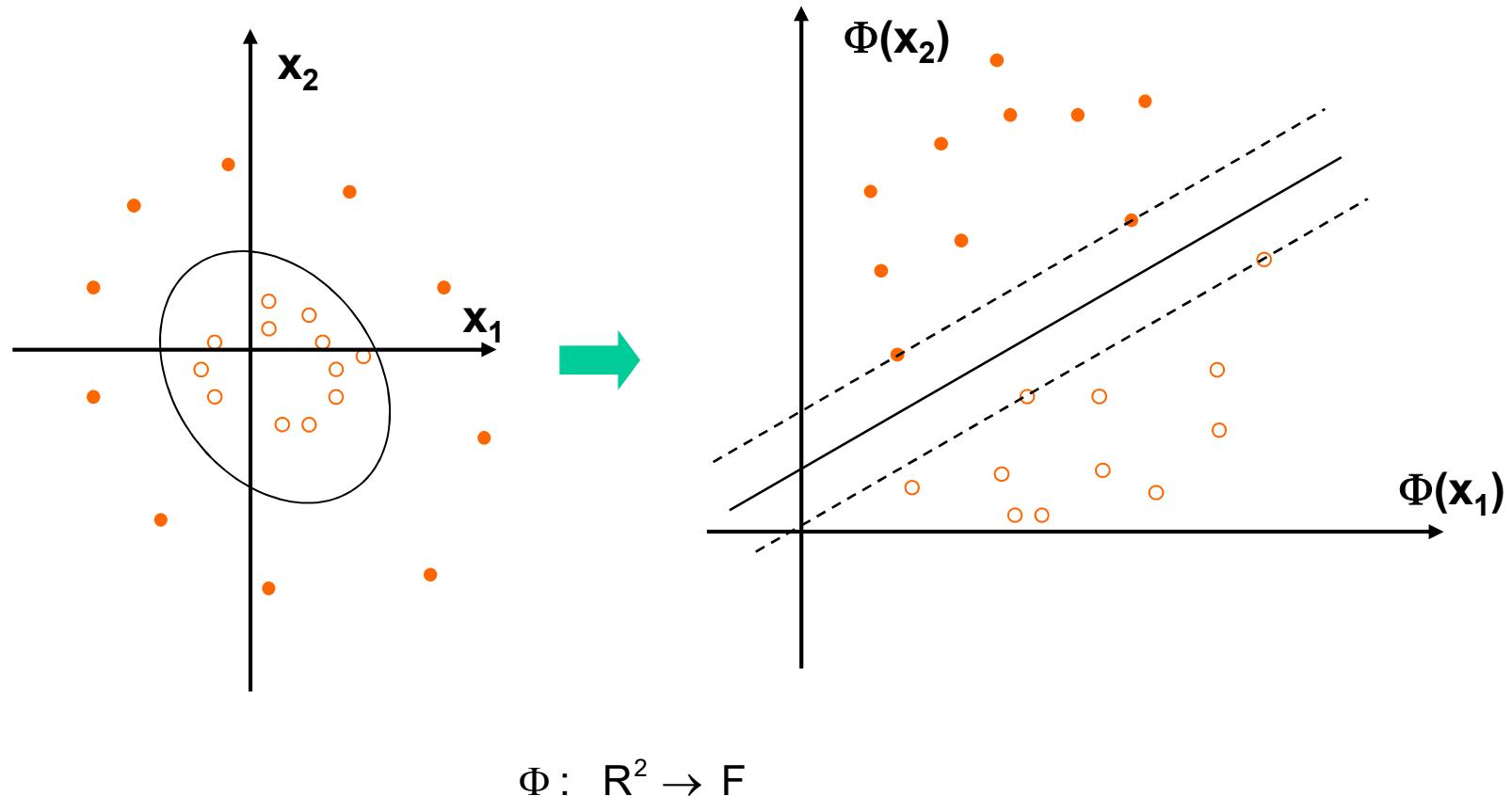
$$x \mapsto \Phi(x)$$

One frequently used kernel function is the Radial Basis Function (RBF):

$$K(x_i, x_j) = \exp\left(-\gamma \|x_i - x_j\|^2\right), \gamma > 0$$

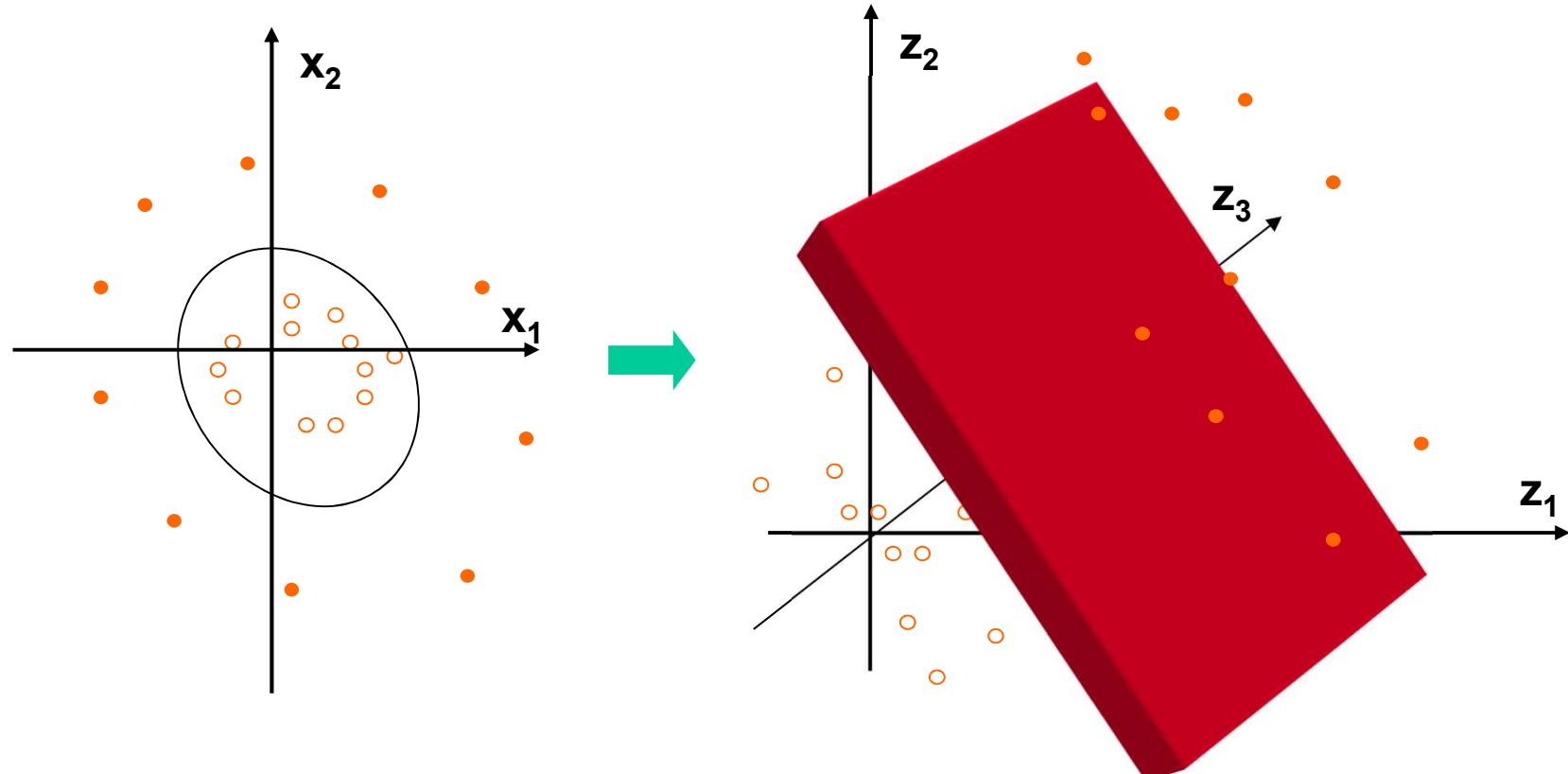
Nonlinearity (III)

Example:



Nonlinearity (IV)

Example:

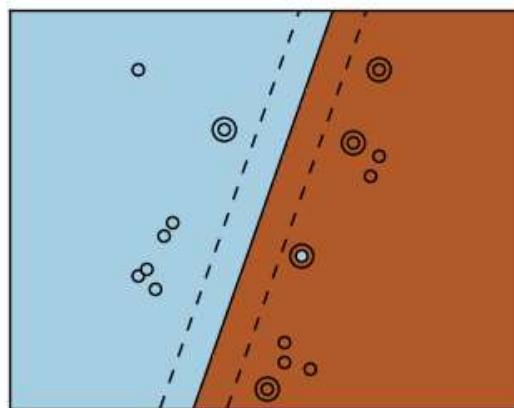


$$\Phi : \mathbb{R}^2 \rightarrow \mathbb{R}^3$$

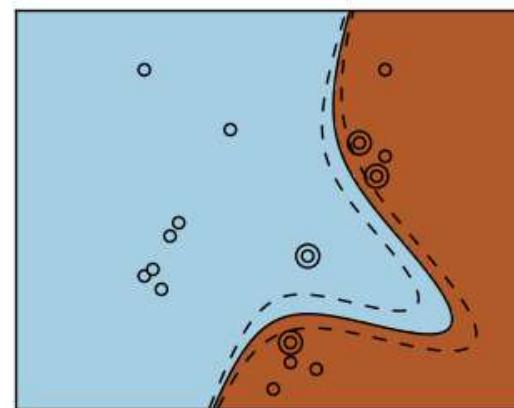
$$(x_1, x_2) \mapsto (z_1, z_2, z_3) := (x_1^2, \sqrt{2}x_1x_2, x_2^2)$$

Different Kernel Functions

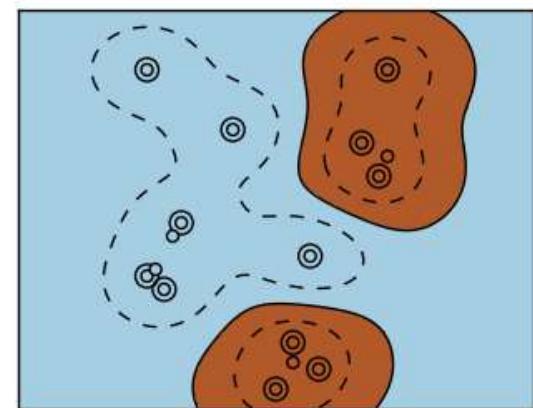
Linear kernel



Polynomial kernel



RBF kernel



Source: http://gael-varoquaux.info/scikit-learn-tutorial/supervised_learning.html

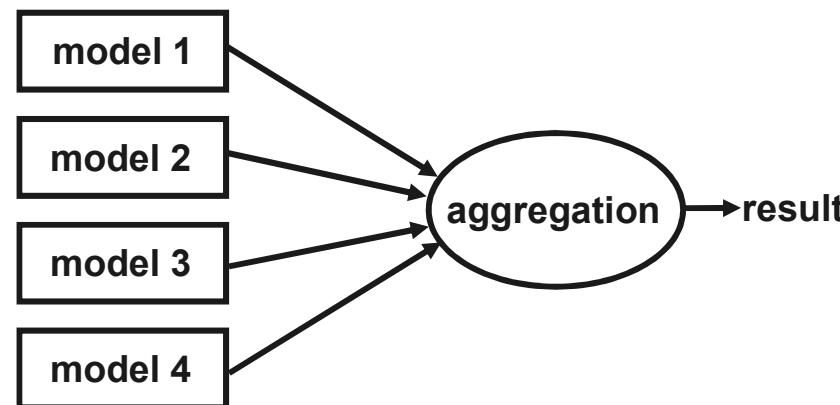
Ensemble Methods

Ensemble Methods

Ensemble methods use different models (created via different data sets, feature sets or methods) that are simultaneously applied to the same problem. The results are sent to an aggregating operation that produces the final result.

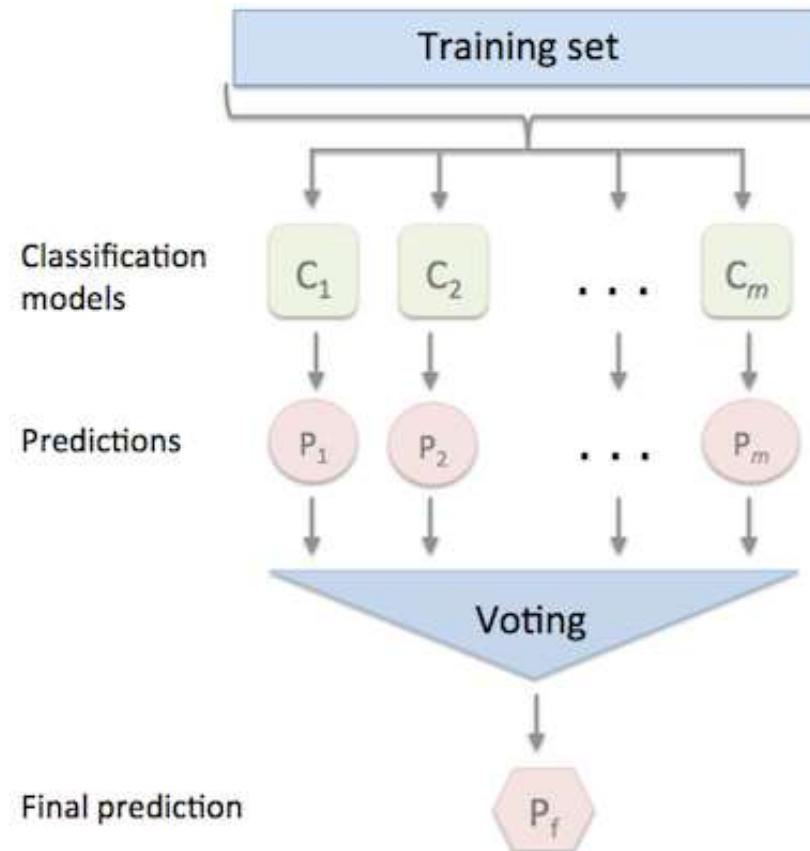
The most widely used classes of ensemble methods are:

- Bagging
- Boosting
- Stacking



Bagging

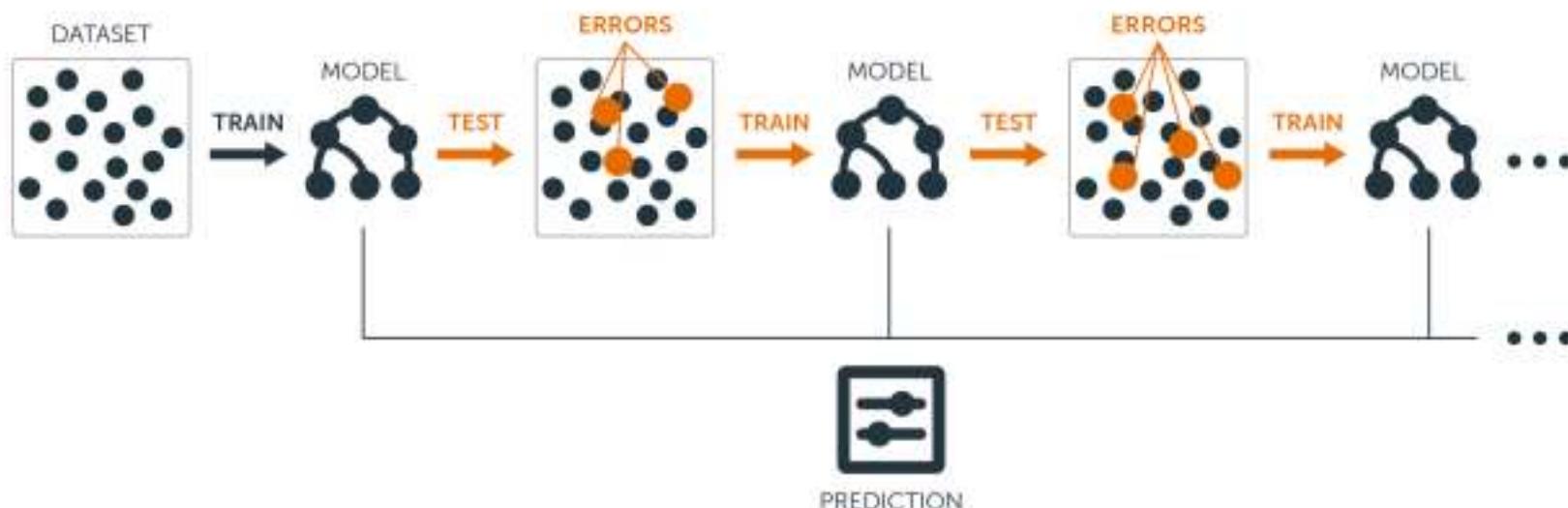
Bagging means to build multiple models from different subsamples of the training dataset and/or with different methods. The results are sent to an (weighted) voting operation that produces the final result.



Source: [http://rasbt.github.io/mlxtend/
user_guide/classifier/EnsembleVoteClassifier/](http://rasbt.github.io/mlxtend/user_guide/classifier/EnsembleVoteClassifier/)

Boosting

Boosting involves sequentially building an ensemble by training each new model instance to emphasize the training instances that previous models mispredict. Different variants exist, mostly based on tree methods. In general, any method can be used. This involves the usage of different methods at the different iterations when building the sequence of models.

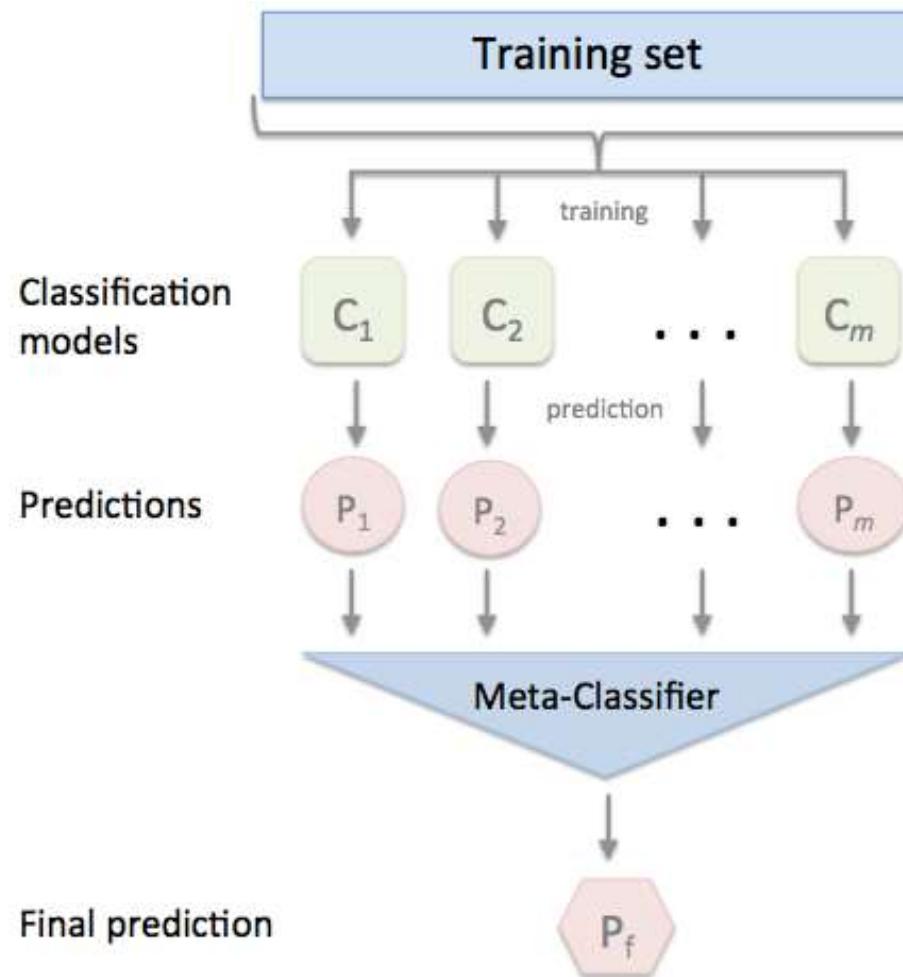


Source: <https://blog.bigml.com/2017/03/14/introduction-to-boosted-trees/>

Stacking

Stacking means to build multiple models (typically of differing types) and a supervisor model that learns how to best combine the predictions of the primary models.

The inputs of the supervisor model (meta-classifier) are the outputs of the other models:



Source: http://rasbt.github.io/mlxtend/user_guide/classifier/StackingClassifier/

Types of Ensembles

Type 1:

- consists of only a few models
- each is a strong model
- like few professional experts



- risk of diverging opinions
- risk of experts being biased to their experiences

Type 2:

- consists of many models
- each is a weak model as a principle
- based on the idea of the wisdom of the masses



- Random Forest and Gradient Boosted Trees are examples

Regression

Predicting using Regression Methods

Example: Predicting House Prices

Groundtruth

↓

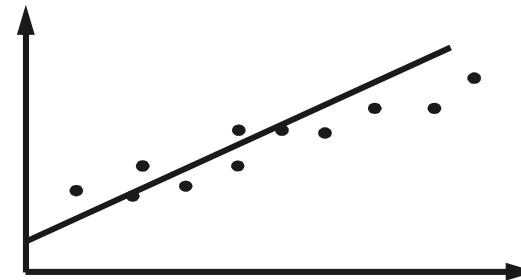
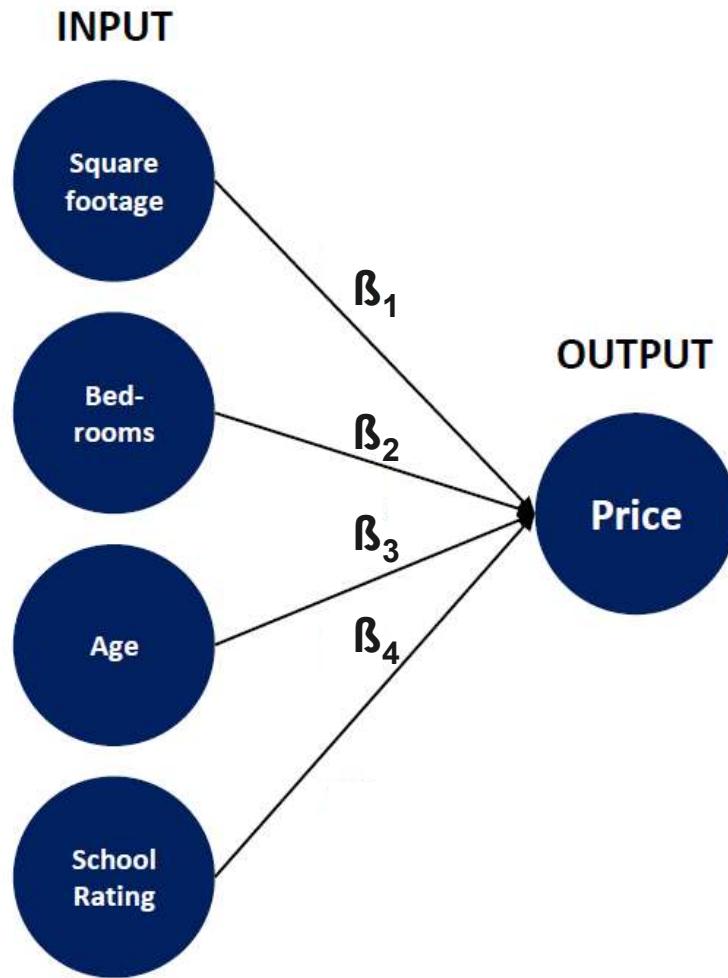
| House No. | Square Footage | Bedrooms | Age | School Rating | Final Price |
|-----------|----------------|----------|-----|---------------|-------------|
| H1 | 1000 | 4 | 3 | 2 | \$100,000 |
| H2 | 800 | 3 | 1 | 4 | \$90,000 |
| H3 | 1200 | 5 | 3 | 5 | \$125,000 |
| H4 | 600 | 2 | 5 | 1 | \$60,000 |
| H5 | 1500 | 6 | 3 | 3 | \$150,000 |

Function: Price = f(SquareFootage, Bedrooms, Age, SchoolRating)

Source: <http://www.sclgsummit.org/uploads/presentation/8934b2d0be055a2261f5d0320f5b59bb.pdf>

Ordinary Least Squares Regression

Traditional OLS Regression Approach



Function:

$$\begin{aligned}
 \text{Price} = & \beta_0 + \beta_1 * \text{SquareFootage} \\
 & + \beta_2 * \text{Bedrooms} + \beta_3 * \text{Age} \\
 & + \beta_4 * \text{SchoolRating}
 \end{aligned}$$

Source: <http://www.sclgsummit.org/uploads/presentation/8934b2d0be055a2261f5d0320f5b59bb.pdf>

Ordinary Least Squares Regression

OLS is a linear approach for predicting a quantitative response Y on the basis of a set of predictor variables X_j

$$y_i = \beta_0 + \beta_1 \cdot x_{1,i} + \beta_2 \cdot x_{2,1} + \cdots + \beta_p \cdot x_{p,i} + v_i = \beta_0 + \boldsymbol{\beta}' \mathbf{x}_i + v_i$$

It assumes that there is approximately a linear relationship between X and Y.

The optimal regression line is found where the sum of squared residuals (RSS)

$$RSS = \sum_{i=1}^n v_i^2 = \sum_{i=1}^n (y_i - \hat{y}_i)^2 = \sum_{i=1}^n (y_i - \beta_0 - \boldsymbol{\beta}' \mathbf{x}_i)^2$$

is minimal. Rewriting

$$y = Xb + v$$

$$\begin{bmatrix} y_1 \\ y_2 \\ \dots \\ y_n \end{bmatrix} = \begin{bmatrix} 1 & x_{1,1} & \dots & x_{m,1} \\ 1 & x_{1,2} & \dots & x_{m,2} \\ \dots & \dots & \dots & \dots \\ 1 & x_{1,n} & \dots & x_{m,n} \end{bmatrix} \cdot \begin{bmatrix} \beta_0 \\ \beta_1 \\ \dots \\ \beta_m \end{bmatrix} + \begin{bmatrix} v_1 \\ v_2 \\ \dots \\ v_n \end{bmatrix}$$

the parameter vector b can be calculated via: $\boldsymbol{\beta} = (X'X)^{-1}X'y$

Measuring the Quality of Fit

Measuring the quality of fit means to measure how well the predictions of a model match the observed data. In the regression setting, the most commonly-used measure is the mean squared error (MSE) which can be calculated for the training and the test sets

$$MSE = \frac{1}{n} \cdot \sum_{i=1}^n (y_i - \hat{y}_i)^2$$

An alternative measure is R², which calculates the proportion of variance explained. It indicates how much better the model predicts compared to the mean \bar{y} as a naïve predictor

$$R^2 = 1 - \frac{\sum_{i=1}^n (y_i - \hat{y}_i)^2}{\sum_{i=1}^n (y_i - \bar{y})^2}$$

To measure the robustness, the formula is applied to the test set but using the mean \bar{y} of the training set. This measure is named Pseudo-R²

$$Pseudo-R^2 = 1 - \frac{\sum_{i=1}^m (y_i^* - \hat{y}_i^*)^2}{\sum_{i=1}^m (y_i^* - \bar{y})^2}$$

Ridge Regression

Ridge Regression

Ridge Regression is an enhancement of OLS using a technique that constrains or regularizes the coefficient estimates. This is done by shrinking the coefficient estimates towards zero resulting in a gain in robustness and a higher generalization ability.

Similar to OLS, Ridge Regression estimates the parameters of the function

$$y_i = \beta_0 + \boldsymbol{\beta}' \mathbf{x}_i + v_i$$

but instead of minimizing the residual sum of squares (v^2), the function to minimize is

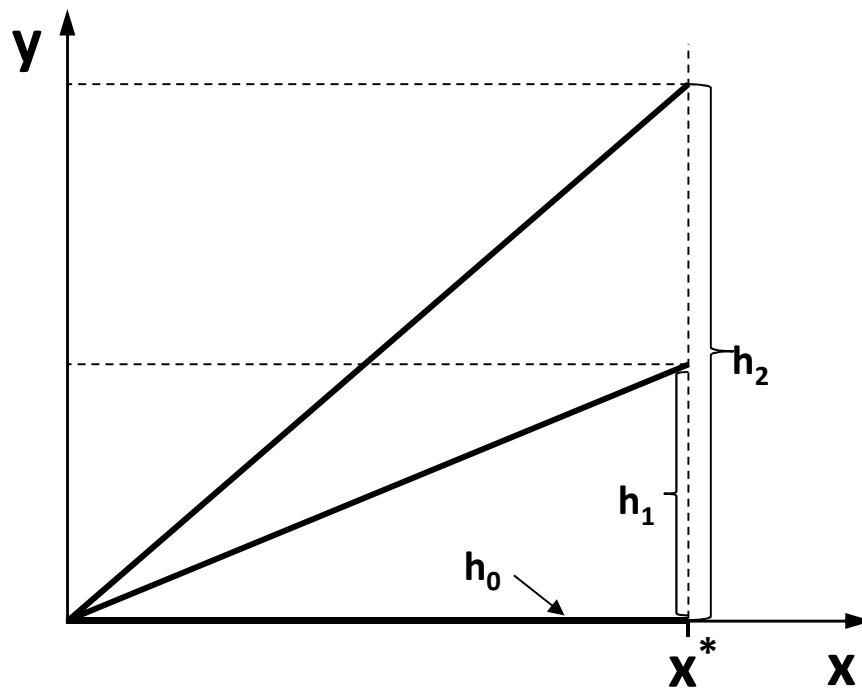
$$\sum_{i=0}^n v_i^2 + \lambda \cdot \sum_{j=1}^p \beta_j^2 = \sum_{i=0}^n (y_i - \beta_0 - \boldsymbol{\beta}' \mathbf{x}_i)^2 + \lambda \cdot \sum_{j=1}^p \beta_j^2$$

where $\lambda \geq 0$ is a tuning parameter, to be determined separately. It is called a shrinkage penalty and has the effect of shrinking the estimates of β_j towards zero. This is equivalent to reducing complexity.

Remark: λ is not applied to the intercept β_0 !

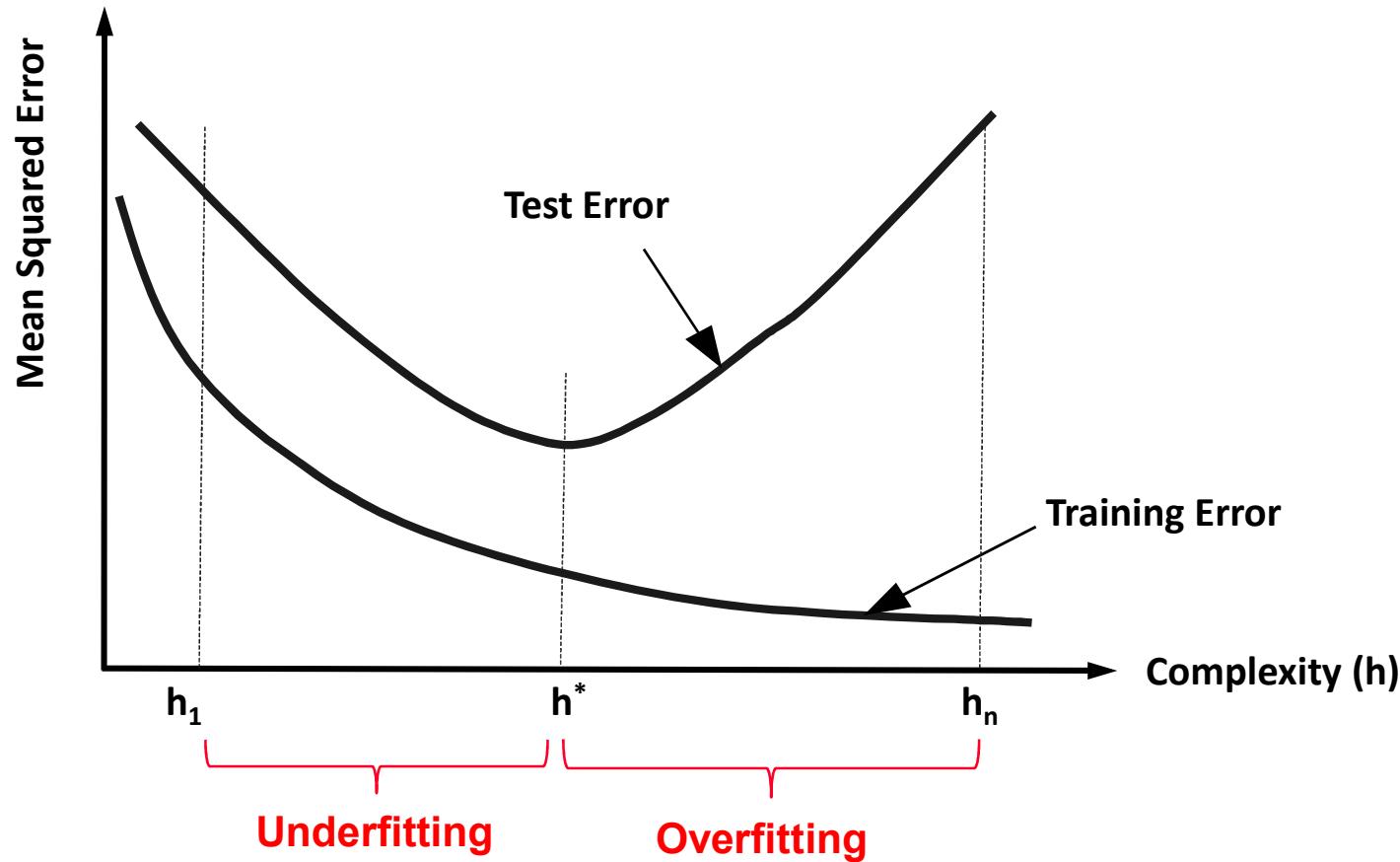
Complexity

Complexity can be measured as the size of the set of possible outputs for a given set of inputs.

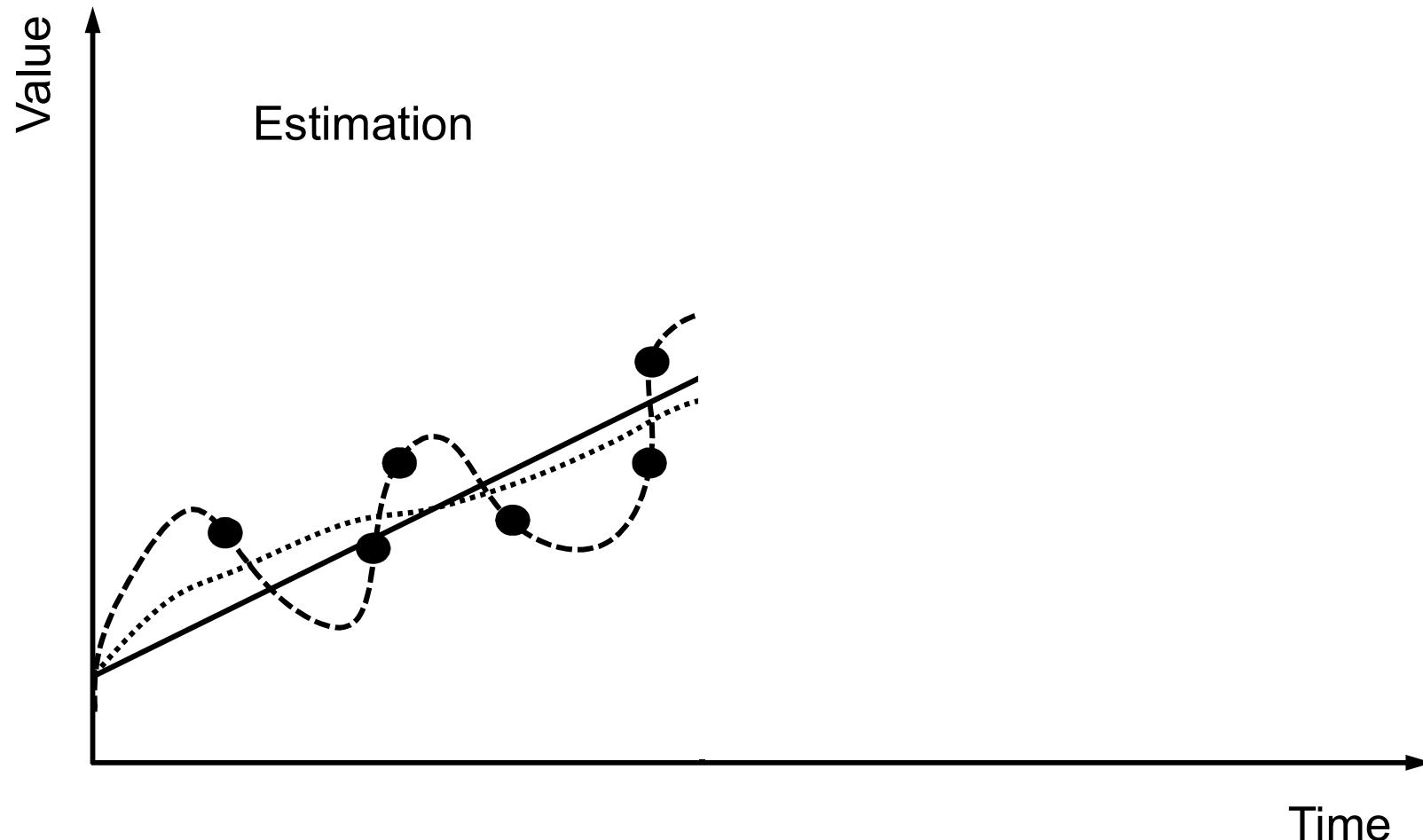


In this example the interval 0 to x^* represents the set of possible inputs. Function h_0 has the lowest complexity because there is just one output independent of the inputs. h_2 has the highest complexity because here the set of possible outputs is the biggest one.

Complexity und Generalisation



Different Complexities



Relationship of λ and Complexity

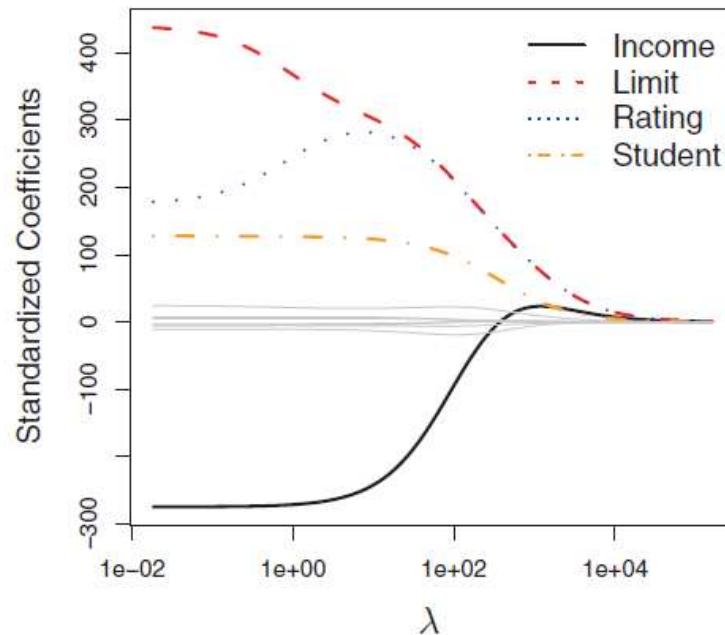
$\lambda \rightarrow \infty$: Lowest Complexity

the ridge regression coefficients are equal to zero. For every input, the result is β_0 .

$\lambda = 0$: Relative High Complexity (linear Model)

the penalty term has no effect, and ridge regression will produce the least squares estimates.

Example:

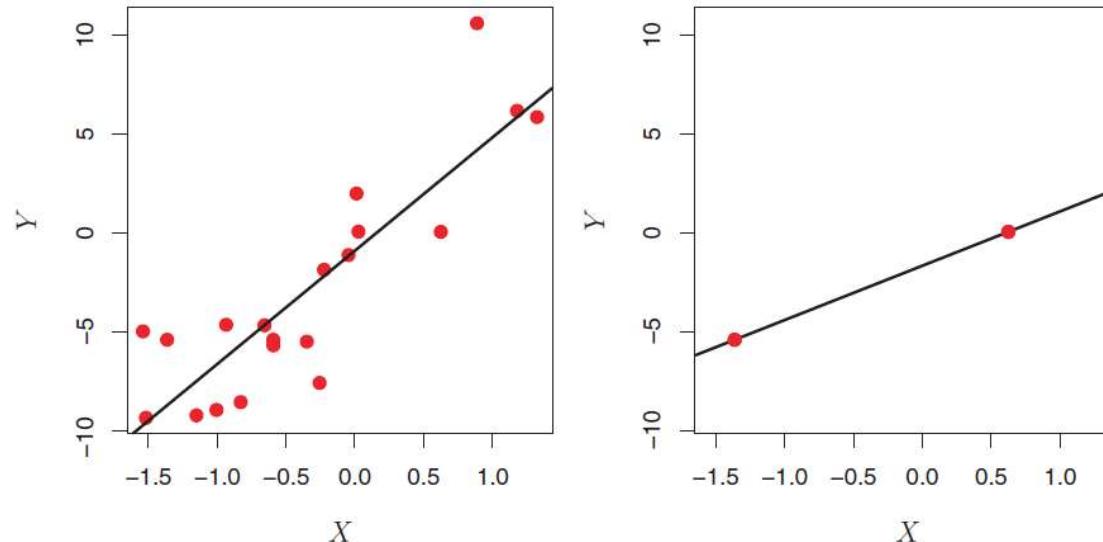


Source:

James et al. (2013): An Introduction to Statistical Learning with R Applications, p. 215f.

Handling High-Dimensionality (I)

OLS is not suitable for high-dimensional data. Especially when the number of features p is as large as, or larger than, the number of observations, OLS cannot be applied. Regardless of whether or not there truly is a relationship between the features and the response, OLS will yield a set of coefficient estimates that result in a perfect fit to the data, such that the residuals are zero.



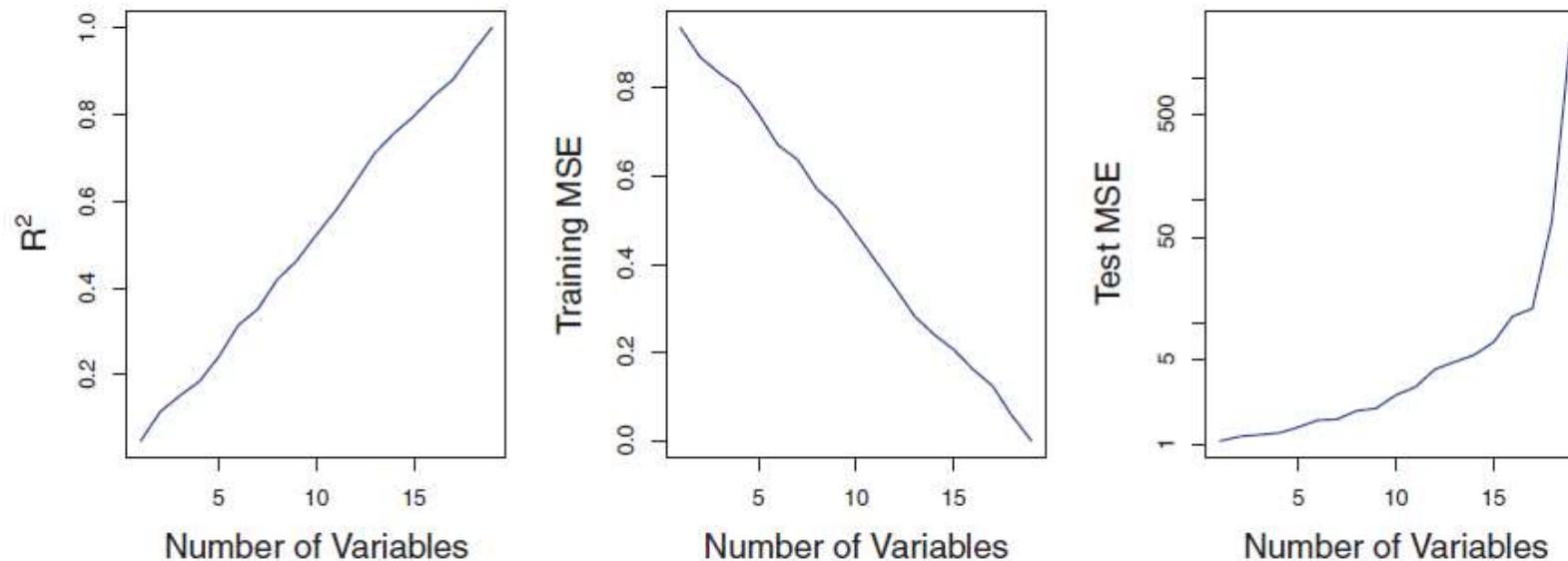
Source:

James et al. (2013): An Introduction to Statistical Learning with R Applications, p. 239f.

The figure shows two cases. When there are 20 observations, $n > p$ and the OLS line does not perfectly fit the data. When there are only two observations, then regardless of the values of those observations, the regression line will fit the data exactly. This is problematic because this perfect fit will almost certainly lead to overfitting of the data.

Handling High-Dimensionality (II)

The figure illustrates the risk of applying OLS when the number of features p is large. The model R^2 increases to 1 as the number of features increases, and the training set MSE decreases to 0. At the same time, the MSE on a test set becomes extremely large as the number of features increases.



In contrast, methods like ridge regression are particularly useful for performing regression in the high-dimensional setting. Essentially, these approaches avoid overfitting by using a less flexible fitting approach than least squares.

Source: James et al. (2013): An Introduction to Statistical Learning with R Applications, p. 240f.

Support Vector Regression

Support Vector Regression

Similar to the other regression approaches, Support Vector Regression estimates the parameters of the function:

$$y_i = \beta_0 + \beta' x_i + v_i$$

The Goal is to find a robust model with a high generalization ability.

SVR regards two sources of Robustness:

- 1. Eliminating Noise**
- 2. Handling Complexity**

Insensitive Loss Function (I)

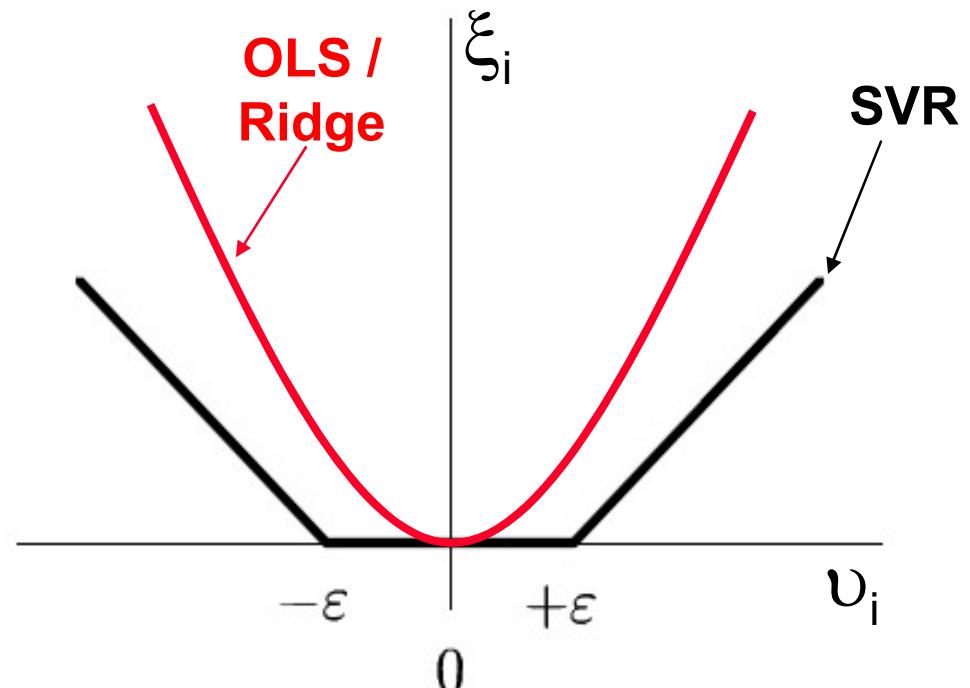
ε -insensitive Loss

$$\xi_i = \begin{cases} |v_i| - \varepsilon, & \text{if } |v_i| > \varepsilon \\ 0, & \text{otherwise} \end{cases}$$

does not penalize acceptable deviations (defined by ε)

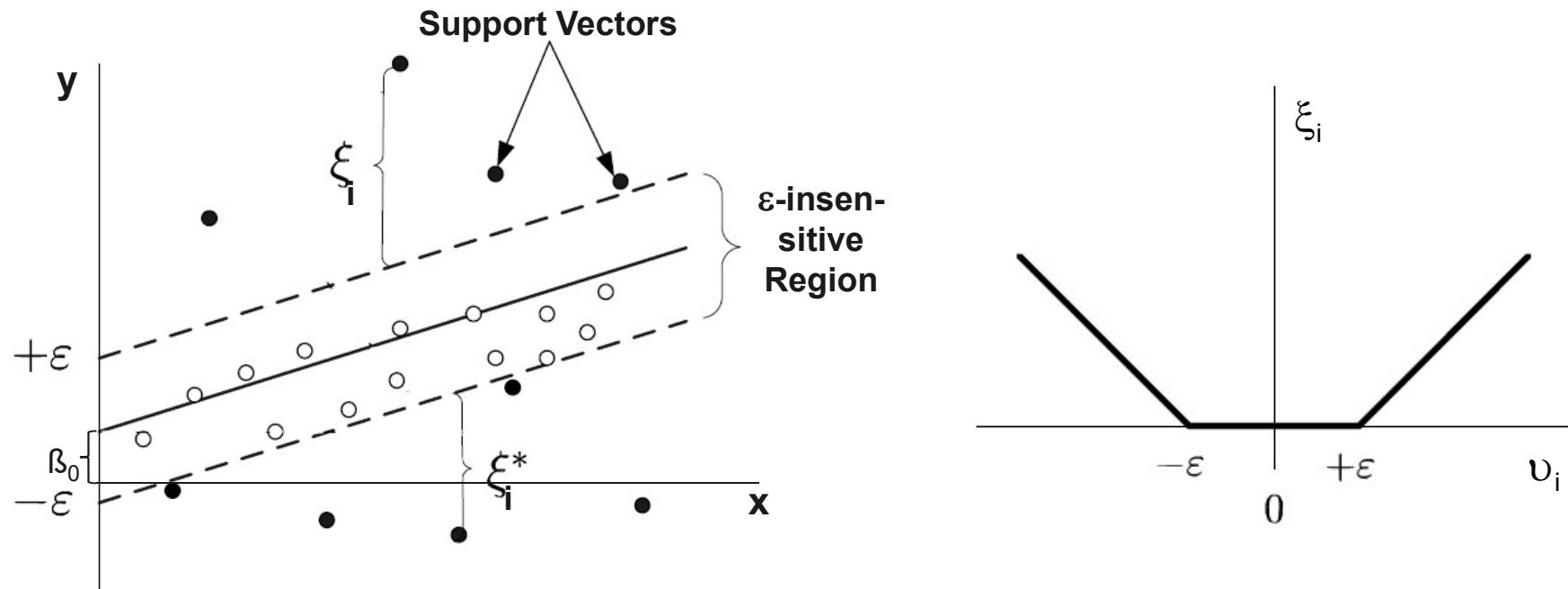
OLS / Ridge

$$\xi_i = v_i^2$$



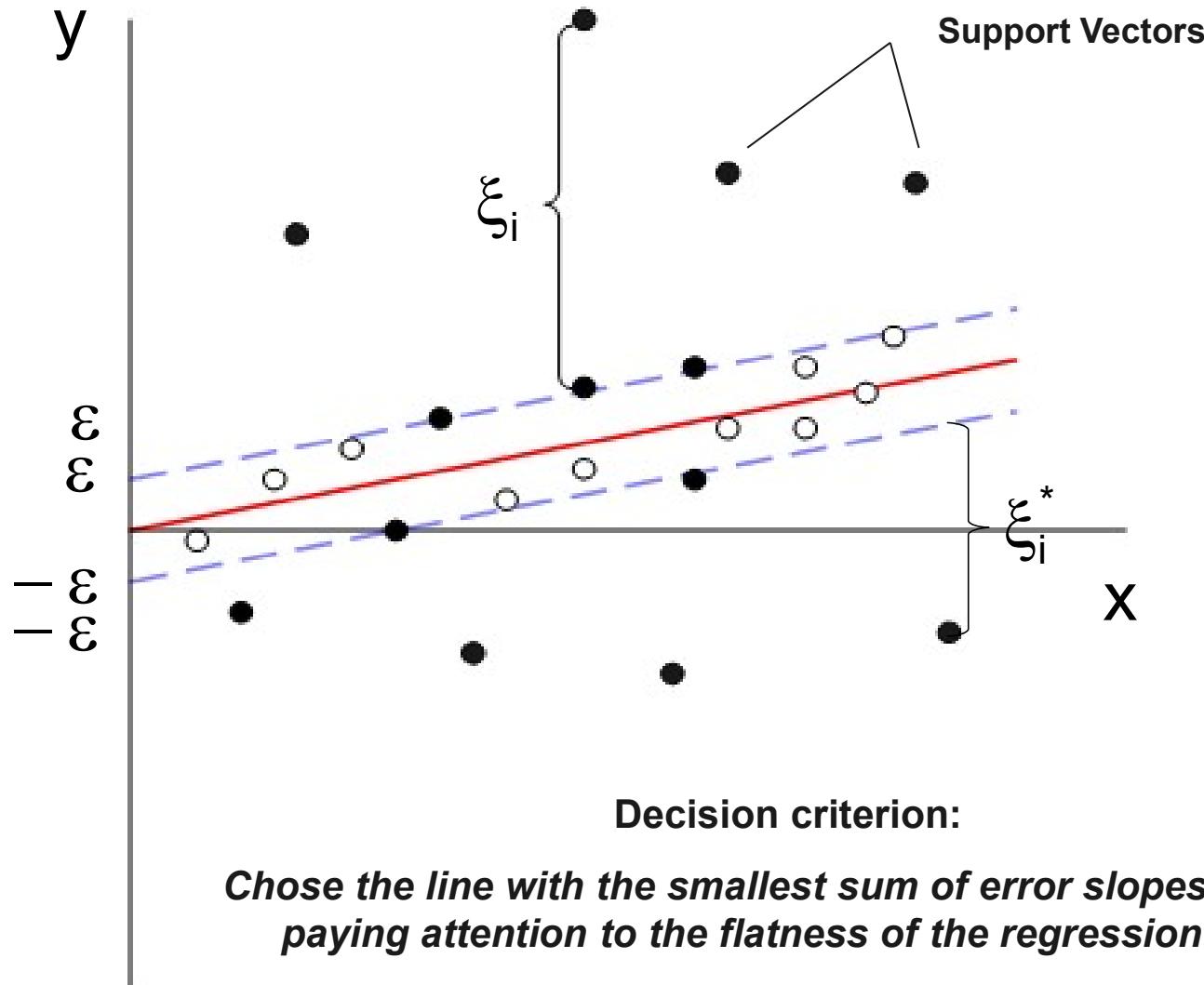
Insensitive Loss Function (II)

Using the ε -insensitive loss function, only those data objects are considered in the estimation, which have a distance greater than ε from the regression function:



Every object inside the ε -insensitive region is ignored. It is regarded as noise.

Support Vector Regression (I)



Estimating the SVR (Linear Case)

The function that has to be estimated is

$$C \cdot \underbrace{\frac{1}{n} \sum_{i=0}^n |y_i - \beta_0 - \beta' x_i|_\varepsilon}_{\text{blue bracket}} + \underbrace{\frac{1}{2} \cdot \|\beta\|^2}_{\text{red bracket}}$$

$$\|\beta\|^2 = \sum_{j=1}^p \beta_j^2$$

where the penalty parameter C controls the trade-off between **approximation** and **generalization**

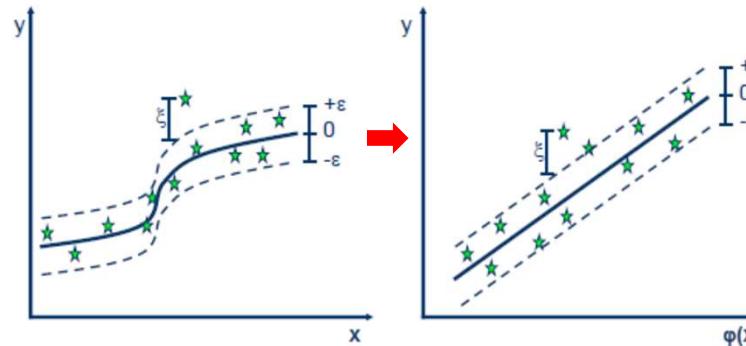
The resulting minimization problem can be formulated as

$$\begin{aligned} \underset{w, b, \xi, \xi^*}{\text{minimize}} \quad & C \sum_{i=1}^n (\xi_i + \xi_i^*) + \frac{1}{2} \|\beta\|^2 \\ \text{with} \quad & y_i - \beta_0 - \beta' x_i \leq \varepsilon + \xi_i \\ & \beta_0 + \beta' x_i - y_i \leq \varepsilon + \xi_i^* \\ & \xi_i, \xi_i^* \geq 0 \end{aligned}$$

Estimating the SVR (Nonlinear Case)

In the nonlinear case Support Vector Regression projects the original data into another dimensional space in order to construct the linear function there

$$\begin{aligned}\phi: \mathbb{R}^N &\rightarrow F \\ x &\mapsto \phi(x)\end{aligned}$$



Analogous to the SVM, kernel functions are used for the projection.

The minimization problem can be formulated as

$$\begin{aligned}\text{minimize}_{w,b,\xi,\xi^*} \quad & C \sum_{i=1}^n (\xi_i + \xi_i^*) + \frac{1}{2} \|\beta\|^2 \\ \text{with} \quad & y_i - \beta' \phi(x_i) - \beta_0 \leq \varepsilon + \xi_i \\ & \beta_0 + \beta' \phi(x_i) - y_i \leq \varepsilon + \xi_i^* \\ & \xi_i, \xi_i^* \geq 0\end{aligned}$$

For application the model can be re-projected to the original space via ϕ^{-1} .

Determining the SVR Parameter

The challenge when applying SVR is to find the best values for ε and C. In literature one can find various approaches.

One very common used approach is k-fold cross-validation. Here the **training** data set is separated into k equal sized subsets. Now, for both parameter ε and C appropriate ranges of values are specified and subdivided into steps. The result is a two-dimensional parameter grid with all the combinations of ε and C.

For every combination the following procedure is performed:

for j=1 to k

choose the j-th subset as validation set

estimate the model using the remaining subsets as training set

apply the validation set to the model and calculate the error $e_l = \sum_{i=1}^{n_k} (y_i - \hat{y}_i)^2$

next j

calculate the total error $e = \sum_{l=1}^k e_l$

The parameter combination with the smallest total error is chosen.

Finally, this parameter combination is applied to the complete training set and tested using the test set.

Neural Networks

Using Neural Network for Regression

Artificial neural networks are often used for classification because of the relationship to logistic regression. Neural networks typically use a logistic activation function and output values from 0 to 1 like logistic regression.

But the continuous output of a net must not be interpreted as a probability, so neural networks can be used too for regression, to model complex and non-linear relationships.

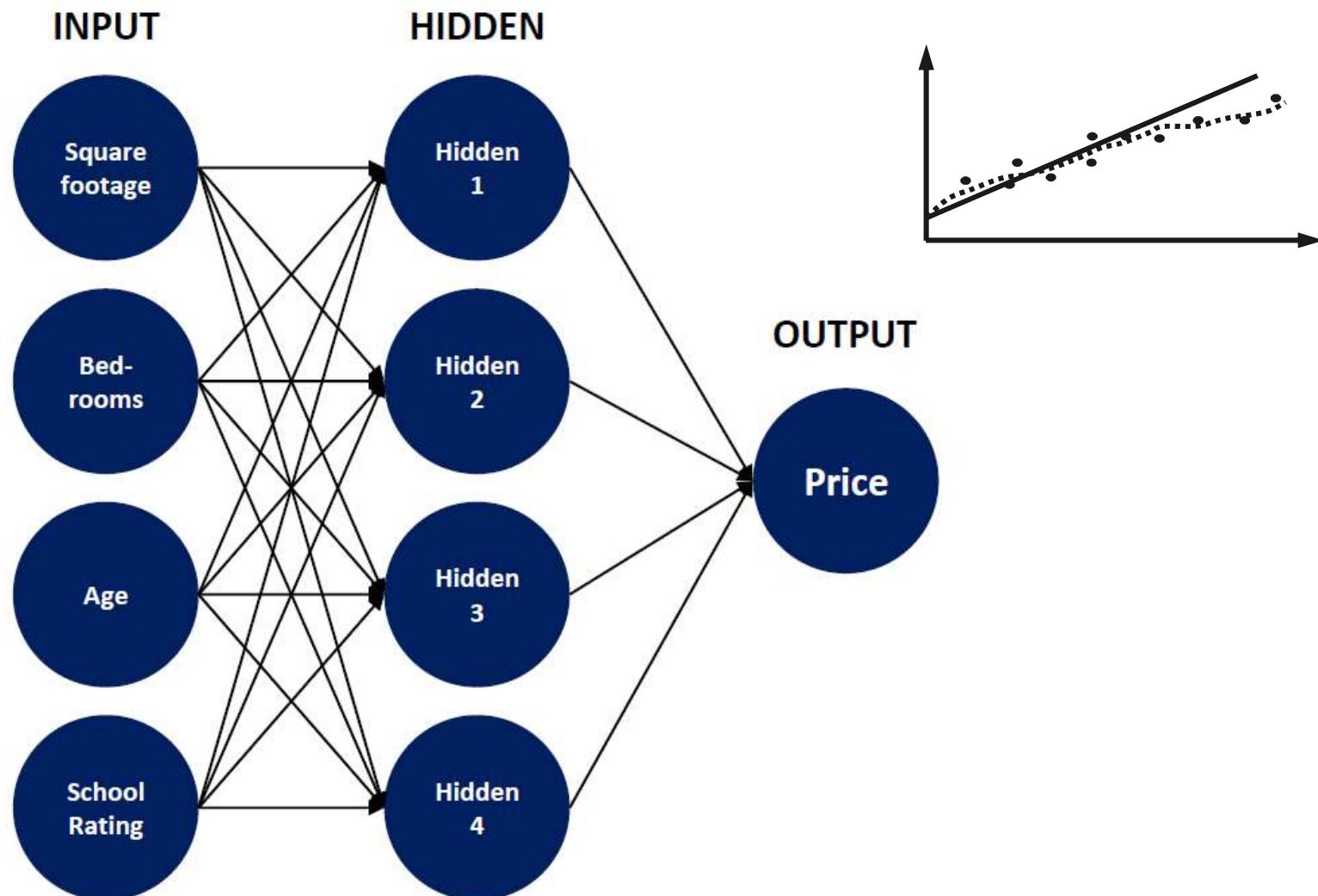
The Singlelayer Perceptron corresponds to a linear regression while a Multilayer Perceptron is able to approximate nearly any function regardless of the complexity and nonlinearity.

Because of the high complexity of the MLP, the models are usually very sensitive and have a tendency to overfitting.

There exist regularization methods, which make the networks better at generalizing beyond the training data.

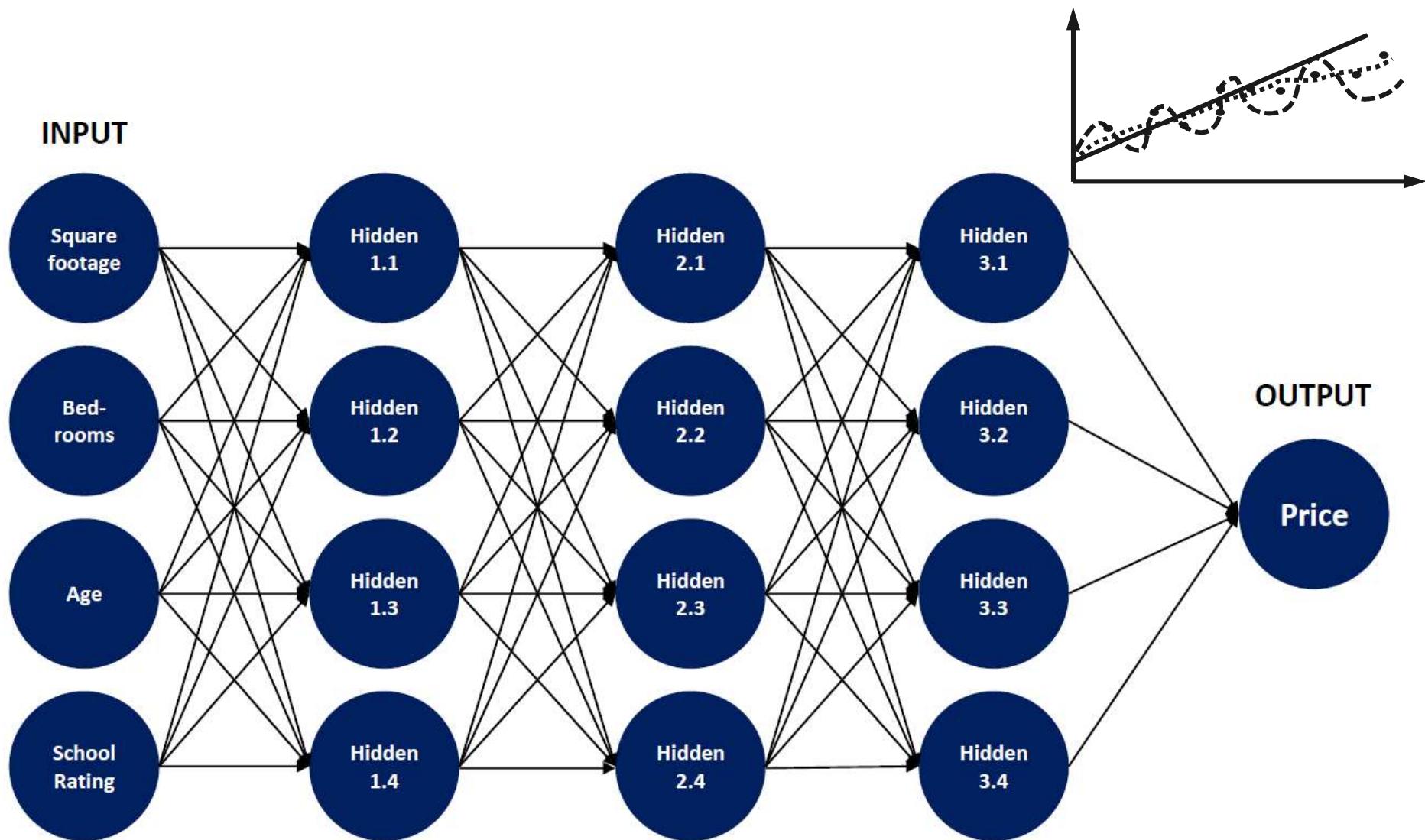
(see <http://neuralnetworksanddeeplearning.com/chap3.html>)

Neural Network (Multilayer Perceptron)



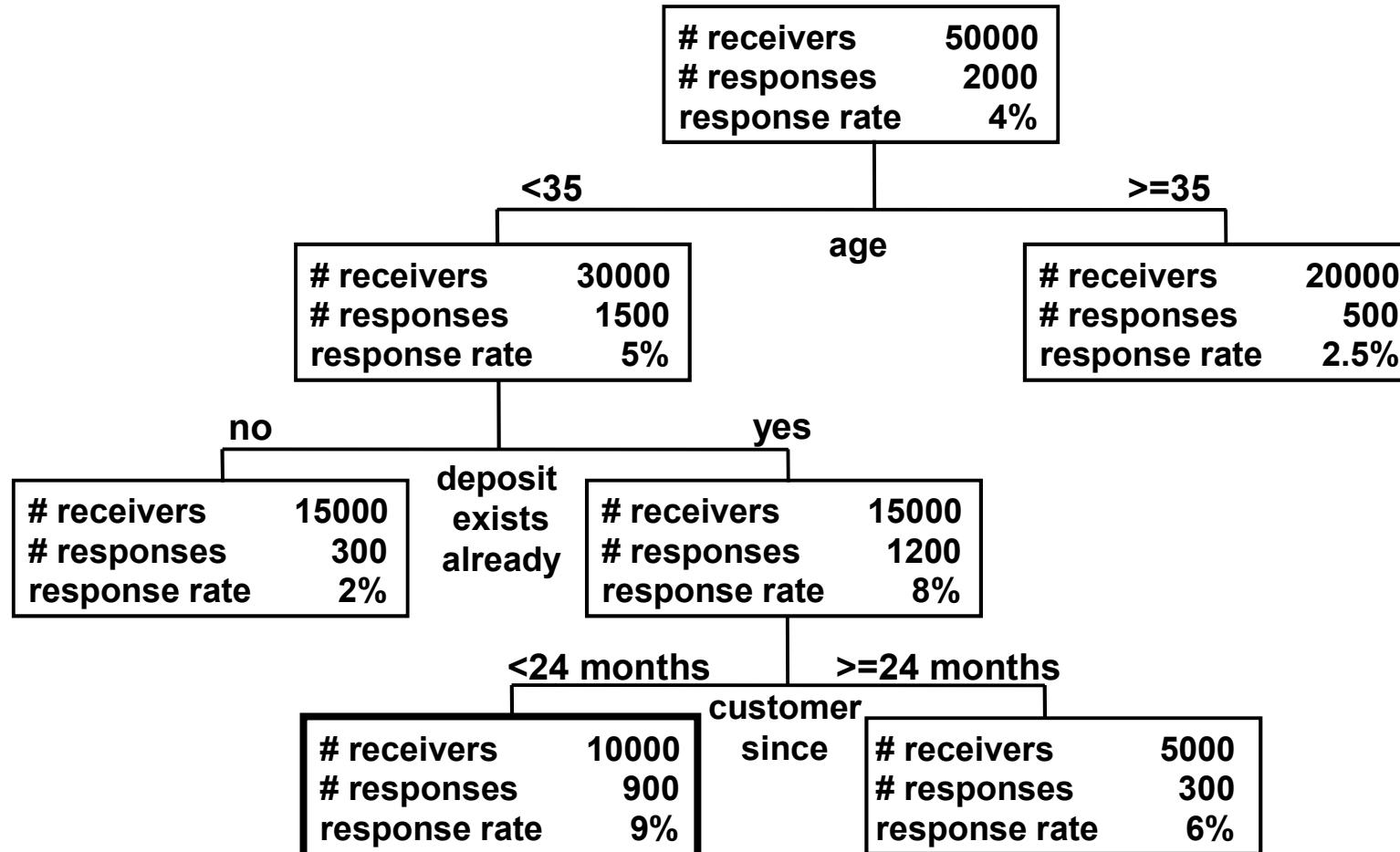
Source: <http://www.sclgsummit.org/uploads/presentation/8934b2d0be055a2261f5d0320f5b59bb.pdf>

Deep Learning



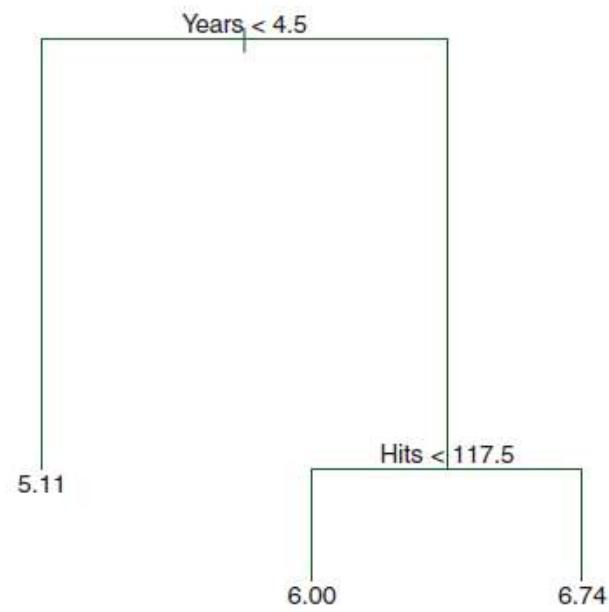
Regression based on Decision Trees

Introductory Example



Regression Trees

Some of the tree approaches can be used for regression too. They can be used for nonlinear multiple regression. The output must be numerical.



The figure shows a CART regression tree for predicting the salary of a baseball player, based on the number of years that he has played in the major leagues and the number of hits that he made in the previous year.

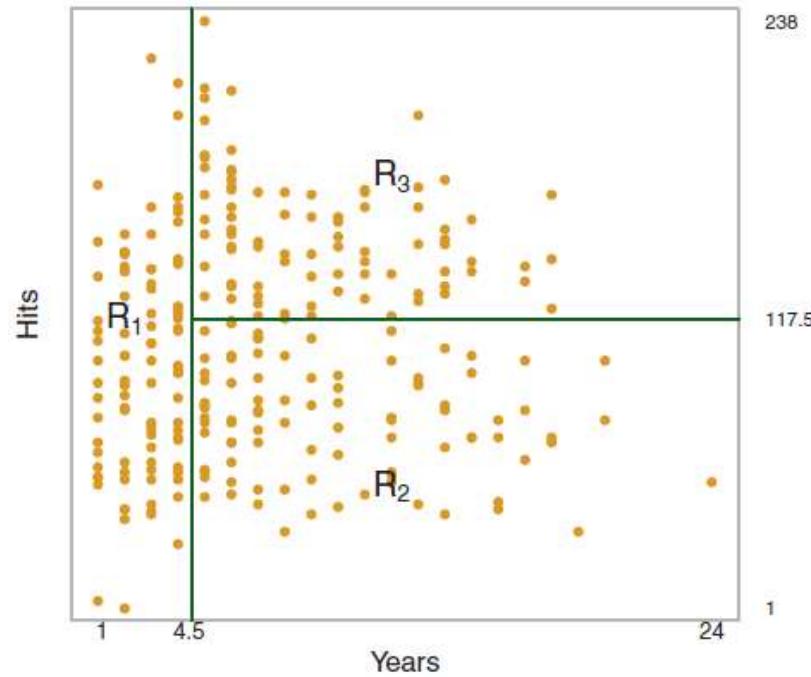
The predicted salary is given by the mean value of the salaries in the corresponding leaf, e.g. for the players in the data set with Years<4.5, the mean (log-scaled) salary is 5.11, and so we make a prediction of $e^{5.11}$ thousands of dollars, i.e. \$165,670, for these players.

Players with Years ≥ 4.5 are assigned to the right branch, and then that group is further subdivided by Hits. The predicted salaries for the resulting two groups are $1,000 \cdot e^{6.00} = \$403,428$ and $1,000 \cdot e^{6.74} = \$845,346$.

Source: James et al. (2013): An Introduction to Statistical Learning with R Applications, p. 304f.

Constructing a Regression Tree (I)

A regression tree partitions the data objects in the feature space into regions R_j , represented by the leafs of the tree. To construct the tree, the goal is to find regions that minimize the RSS, given by



$$RSS = \sum_{j=1}^J \sum_{i \in R_j} (y_i - \bar{y}_{R_j})^2$$

where \bar{y}_{R_j} is the mean value for the training observations within the j^{th} region with $\hat{y}_i = \bar{y}_{R_j}$.

To perform the *recursive binary splitting*, we first select the feature X_j and the cutpoint s such that splitting the feature space into the regions $X_j < s$ and $X_j \geq s$

leads to the greatest possible reduction in RSS.

We consider all features and all possible values of the cutpoint s for each of the features, and then choose the feature and cutpoint such that the resulting tree has the lowest RSS.

Source: James et al. (2013): An Introduction to Statistical Learning with R Applications, p. 305f.

Constructing a Regression Tree (II)

For any feature j and cutpoint s , we seek the value of j and s that minimizes

$$\sum_{i: x_i \in R_1(j,s)}^n (y_i - \bar{y}_{R_1})^2 + \sum_{i: x_i \in R_2(j,s)}^n (y_i - \bar{y}_{R_2})^2$$

where \bar{y}_{R_1} is the mean value for the training observations in $R_1(j,s)$, and \bar{y}_{R_2} is the mean value for the training observations in $R_2(j,s)$.

After splitting, we repeat the process, looking for the best feature and best cutpoint in order to split the data further so as to minimize the RSS within each of the resulting regions.

The process continues until a stopping criterion is reached; for instance, we may continue until no region contains more than five observations.

Once the regions R_1, \dots, R_j have been created, we predict the objects from the test set.

To handle the problem of overfitting, pruning methods exist even for regression trees. For further information see:

James et al. (2013): An Introduction to Statistical Learning with R Applications, p. 307f.

Random Forests for Regression

Due to the usage of means as predictors a regression tree usually simplifies the true relationship between the inputs and the output. The advantage over traditional statistical methods is, that it can give valuable insights about which variables are important and where. But the prediction ability is poor compared to other regression approaches.

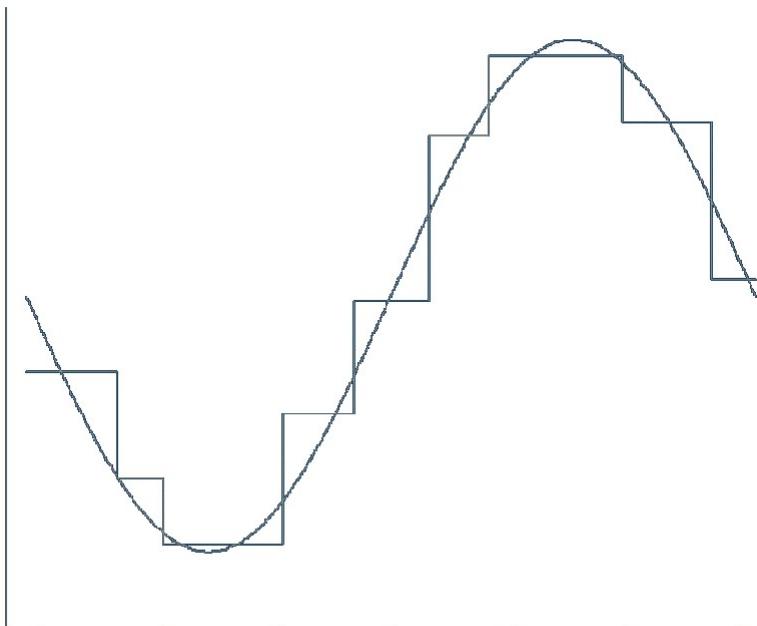
A much better prediction quality can be achieved with the creation of an ensemble of trees, use them for prediction and averaging their results. This is done, when applying the Random Forests approach to a regression task.

Regression Forests are an ensemble of different regression trees and are used for nonlinear multiple regression. The principle is the same as in classification, except that the output is not the result of a voting but instead of an averaging process.

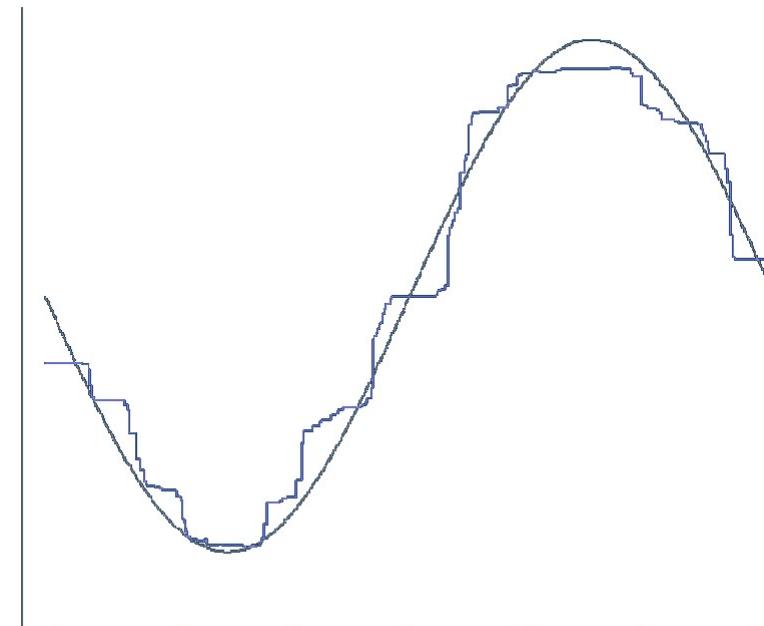
The disadvantage of Random Forests is that the analysis, which aggregates over the results of many bootstrap trees, does not produce a single, easily interpretable tree diagram.

Comparing the Fitting Ability of one vs. many Regression Trees

Single Regression Tree



Average of 100 Regression Trees

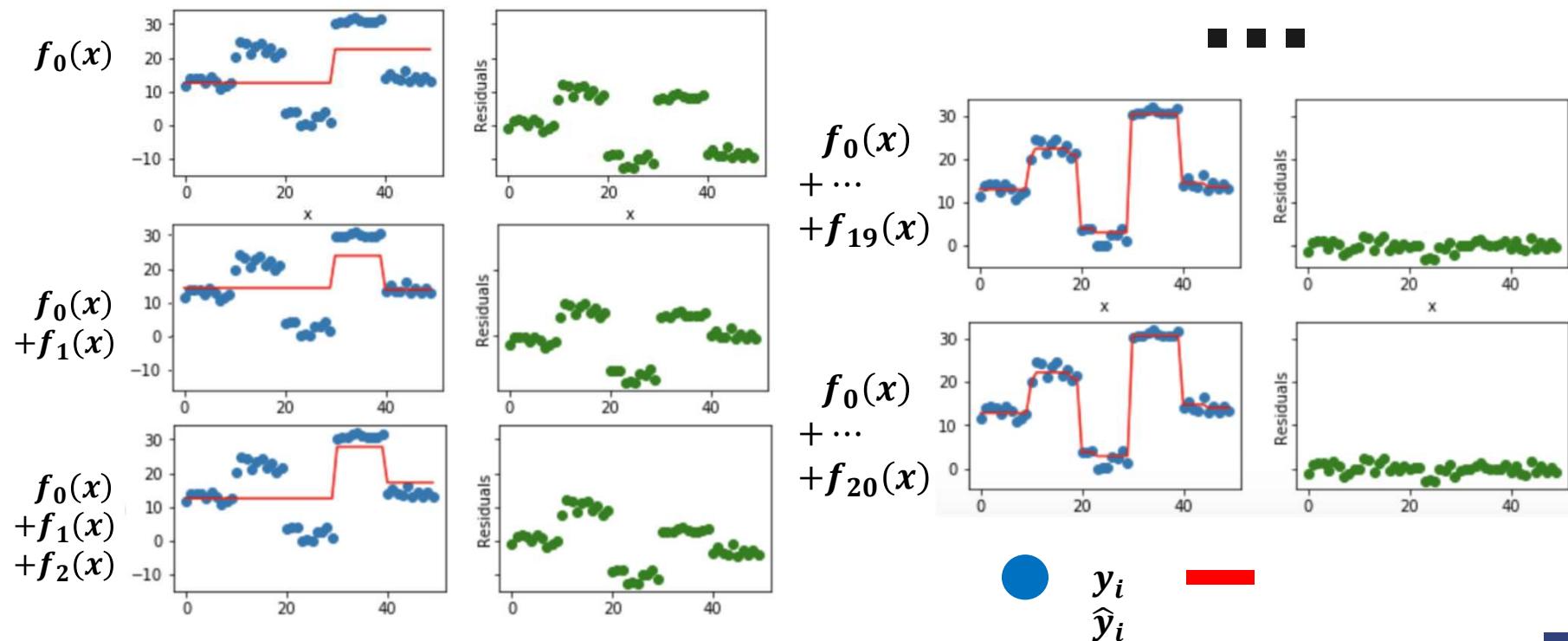


Gradient Boosted Trees for Regression

Gradient Boosted Trees for regression apply the idea of boosting to regression tasks. Typically the MSE ($\sum_{i=1}^n (y_i - \hat{y}_i)^2$) is used as quadratic loss function.

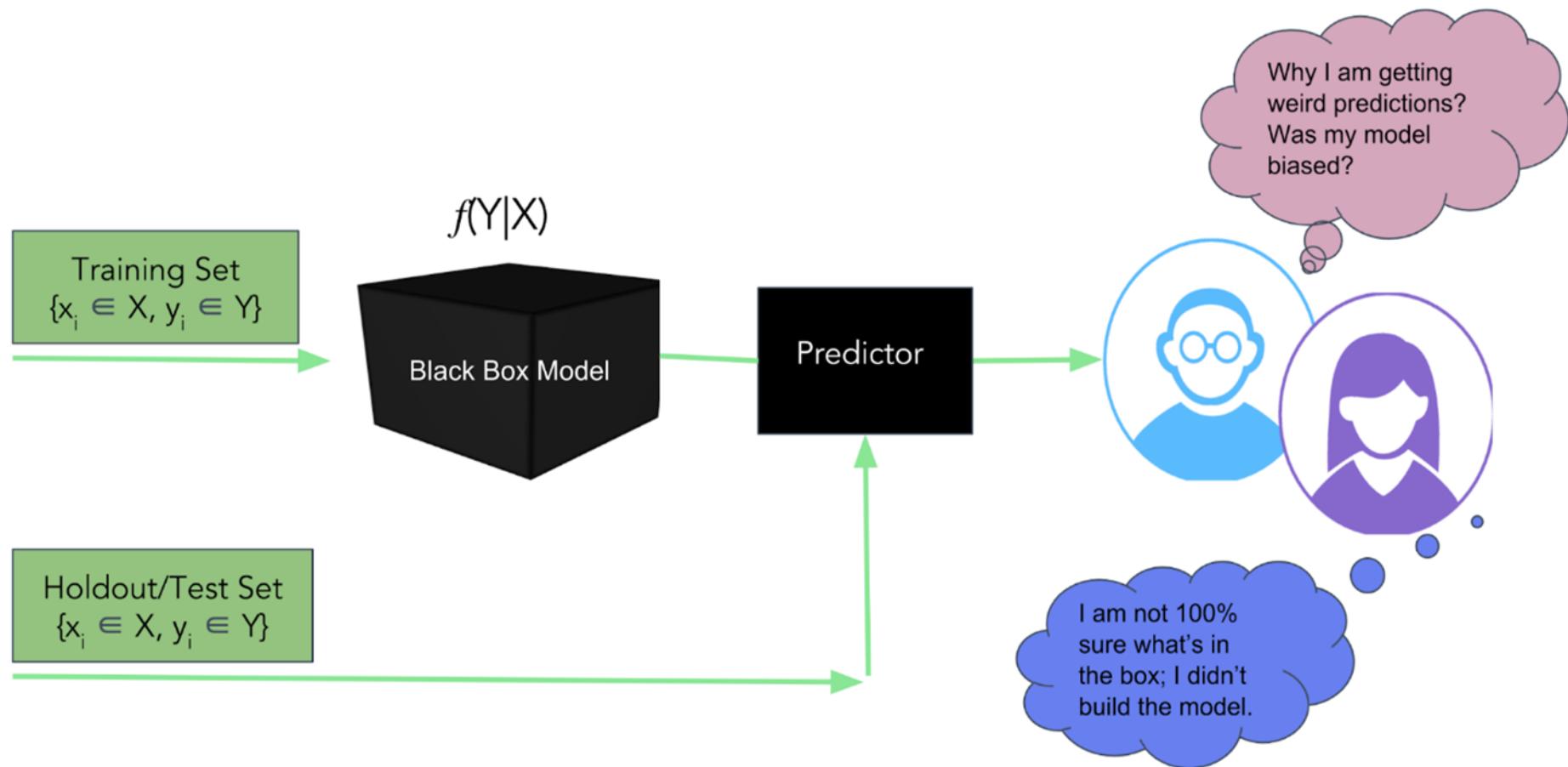
Using the same procedure as described in the classification part, the ensemble is created: $\hat{y} = \sum_{t=0}^m f_t(x) = f_0(x) + f_1(x) + f_2(x) + \dots + f_m(x)$

Graphical example:



Interpretable Machine Learning

Why Interpretable Machine Learning?



<https://www.oreilly.com/ideas/interpreting-predictive-models-with-skater-unboxing-model-opacity>

Why Interpretable Machine Learning?

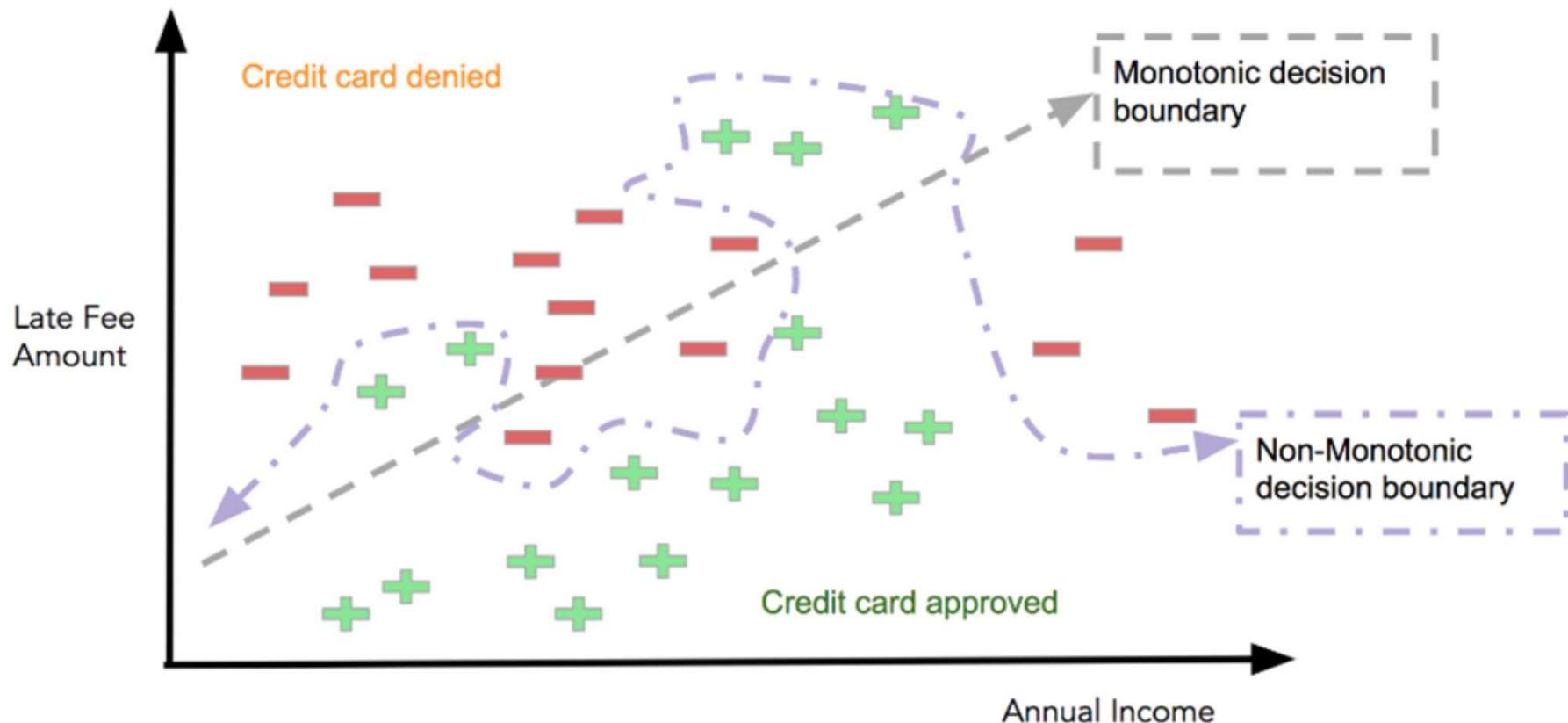
Practitioners often choose linear models over complex ones, compromising performance for interpretability, which might be fine for many use cases where the cost of an incorrect prediction is not high. But, in some scenarios, such as credit scoring or the judicial system, models have to be both highly accurate and understandable.

Model interpretation is the ability to explain and validate the decisions of a predictive model to enable fairness, accountability, and transparency in the algorithmic decision-making.

Interpretability is the degree to which a human can understand the cause of a decision. The higher the interpretability of a machine learning model, the easier it is for someone to comprehend why certain decisions or predictions have been made.

<https://www.oreilly.com/ideas/interpreting-predictive-models-with-skater-unboxing-model-opacity>

Model Performance versus Interpretability



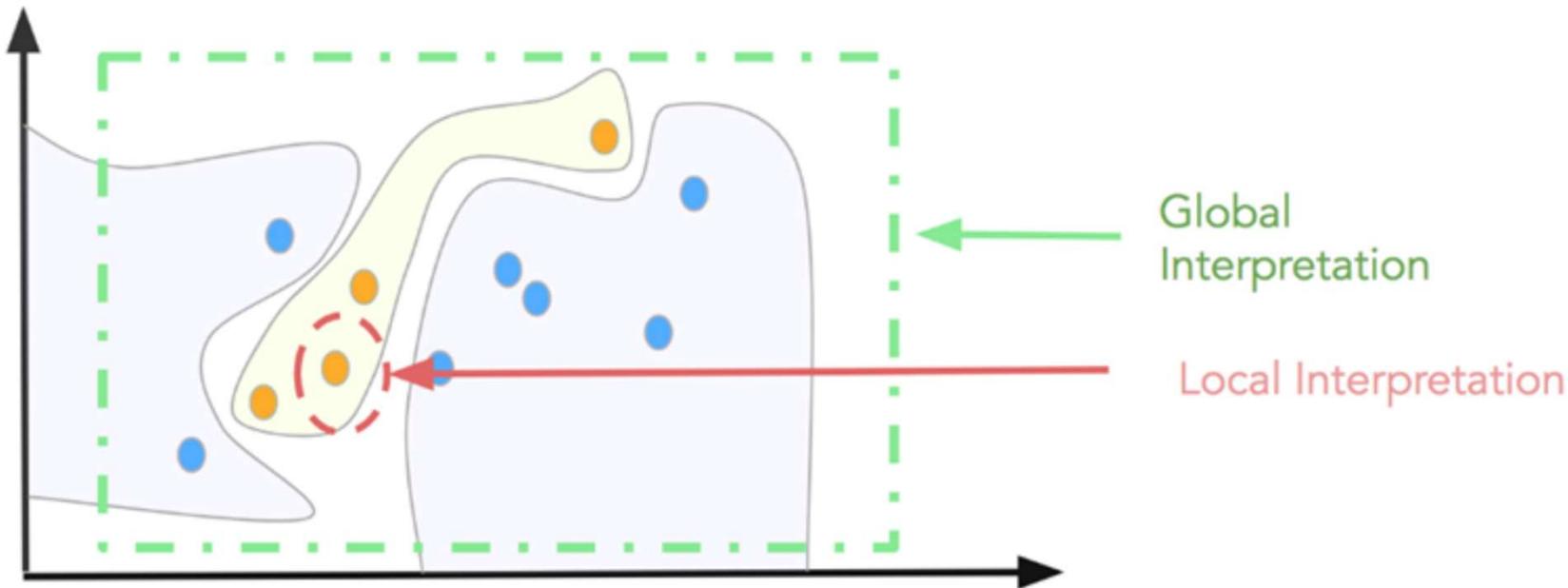
Linear models are simpler to interpret because the relationship between the input variables and the model's output could be quantified in magnitude and direction using the model's coefficient weights. In order to capture the non-monotonic relationship between the independent variable and the model's response function, one generally has to make use of more complex models.

Scope of Interpretability

- **Algorithm Transparency:** Algorithm transparency is about how the algorithm learns a model from the data and what kind of relationships it can learn.
- **Global, Holistic Model Interpretability:** How does the trained model make predictions? Which features are important and what kind of interactions between them take place?
- **Global Model Interpretability on a Modular Level:** How do parts of the model affect predictions?
- **Local Interpretability for a Single Prediction:** Why did the model make a certain prediction for an instance?

Christoph Molnar: Interpretable Machine Learning, 2019, Leanpub

Global versus Local Interpretation



Global interpretation is based on the complete data set and local interpretation on an individual prediction.

<https://www.oreilly.com/ideas/interpreting-predictive-models-with-skater-unboxing-model-opacity>

Categories of Methods for Interpreting

Methods for interpreting machine learning models can basically be categorized into model-specific and model-agnostic methods.

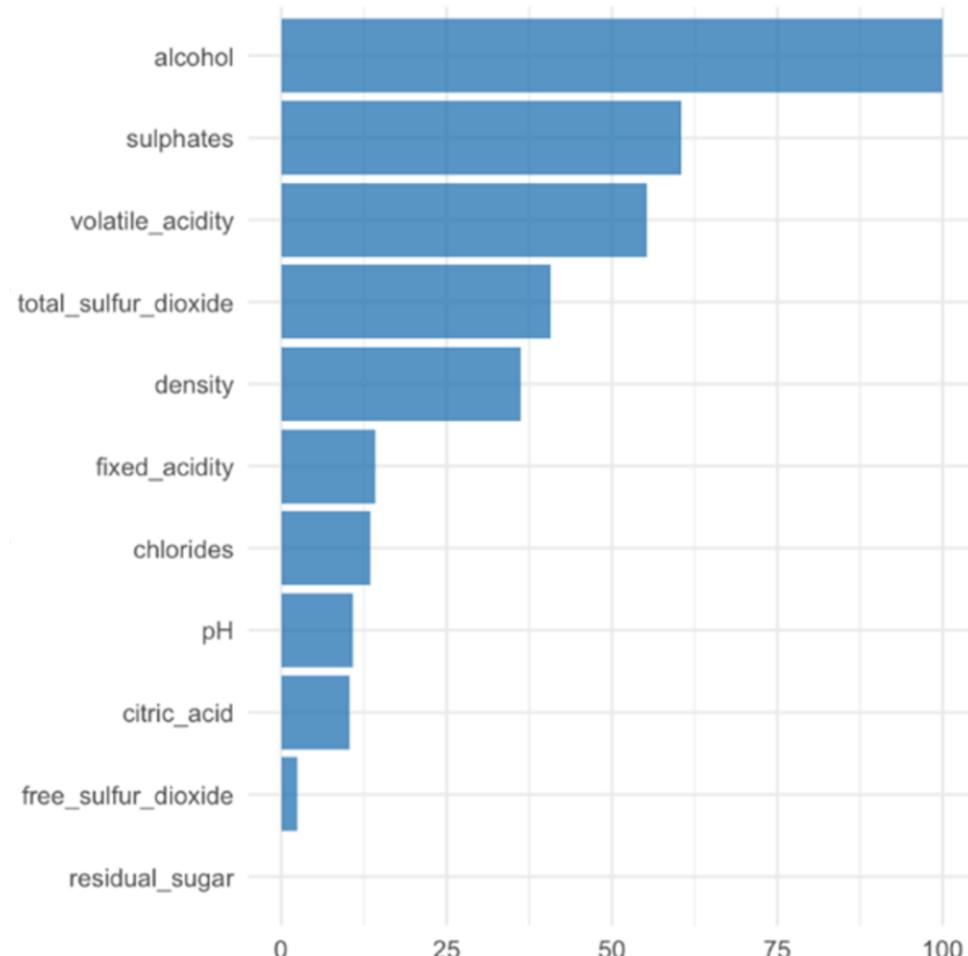
While model-specific methods can only be applied to models generated with a specific methodology (e.g. Random Forest), model-agnostic methods can in principle be applied to any monitored model of machine learning.

Model-agnostic methods often work by changing the inputs of the machine learning model and measuring changes in the model outputs.

Feature Importance

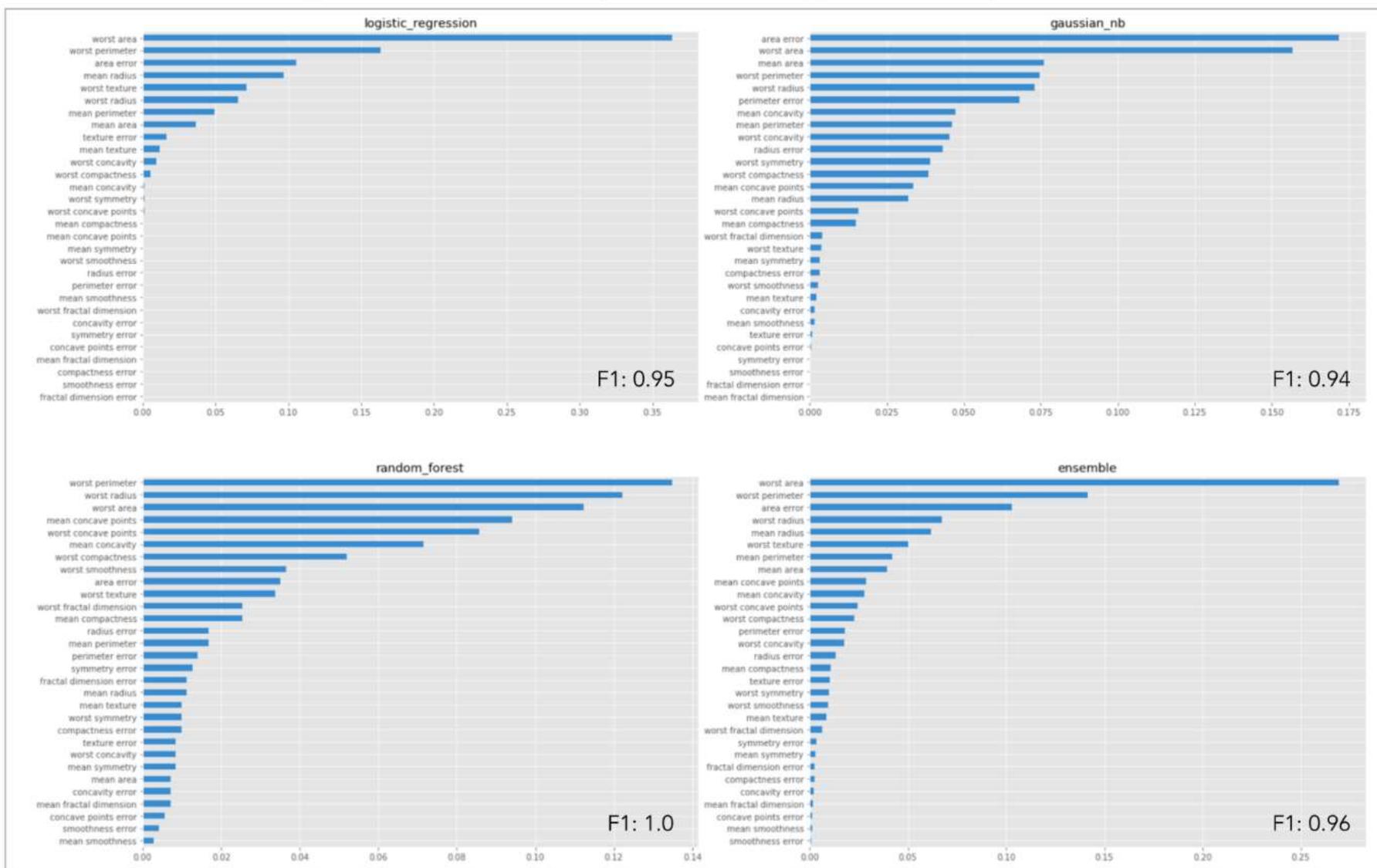
Feature Importance expresses the relative contribution of each feature to the predictions.

We measure the importance of a feature by calculating the increase in the model's prediction error after permuting the feature. A feature is "important" if shuffling its values increases the model error, because in this case the model relied on the feature for the prediction. A feature is "unimportant" if shuffling its values leaves the model error unchanged, because in this case the model ignored the feature for the prediction.



Christoph Molnar: Interpretable Machine Learning, 2019, Leanpub

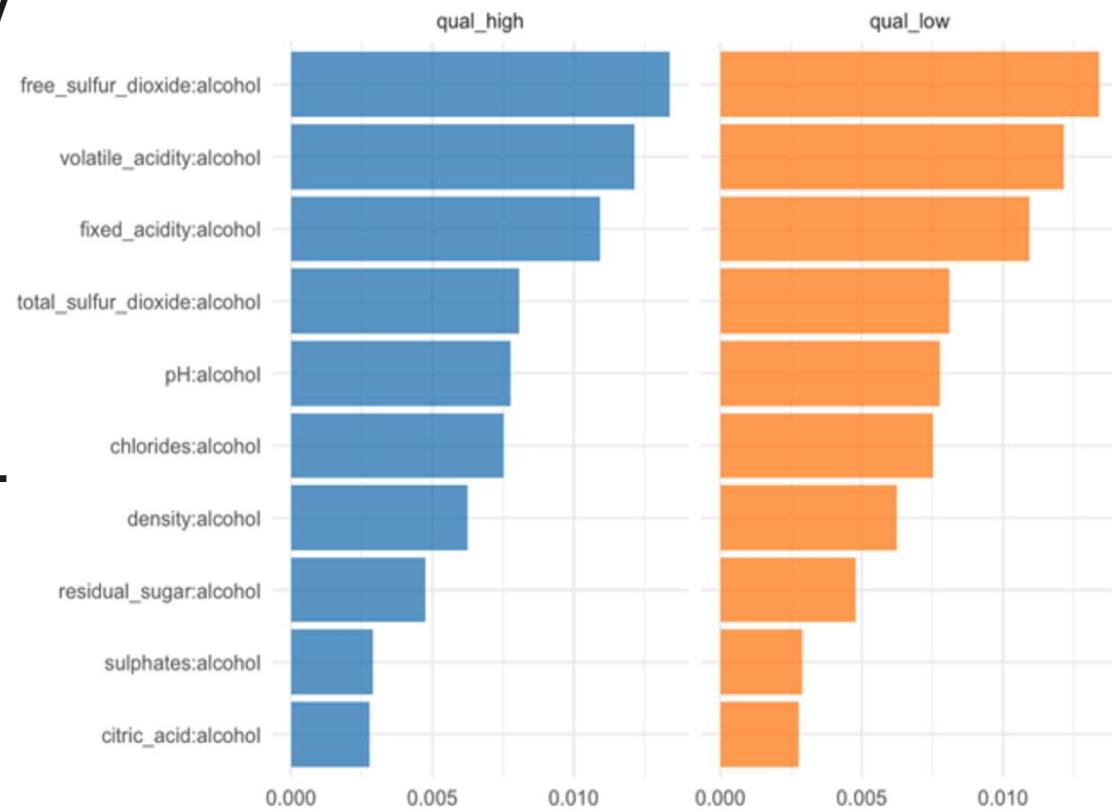
Feature Importance



<https://www.oreilly.com/ideas/interpreting-predictive-models-with-skater-unboxing-model-opacity>

Feature Interaction

Feature interaction is measured by decomposing the prediction function of the model: If a feature j has no interaction with another feature, the prediction function can be expressed as the sum of the sub-function that depends only on j and the sub-function that depends on the remaining other features. If the deviations of the model are fully explained by the addition of the sub-functions, there is no interaction between feature j and the other features. Any variance that is not explained can therefore be traced back to an interaction. It serves as a measure of interaction strength.



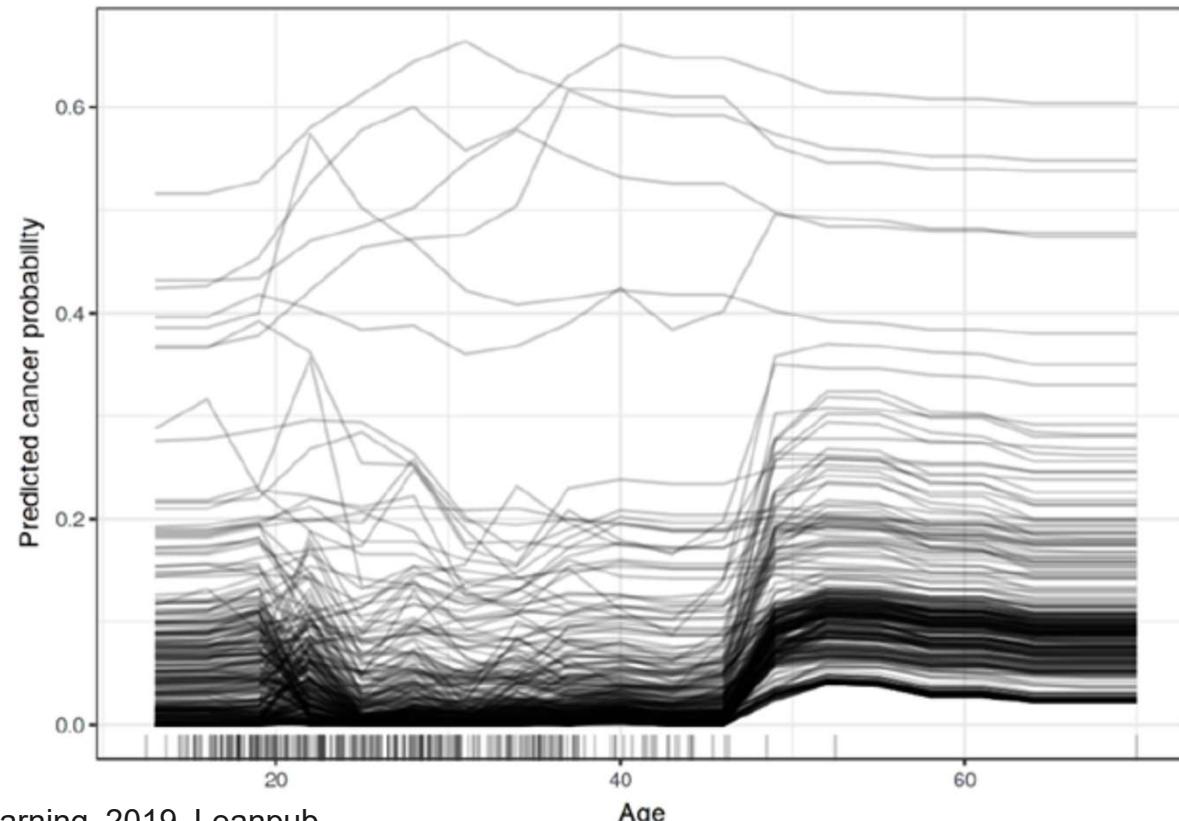
Christoph Molnar: Interpretable Machine Learning, 2019, Leanpub

Individual Conditional Expectation (ICE)

Individual Conditional Expectation (ICE) plots display one line per instance that shows how the instance's prediction changes when a feature changes.

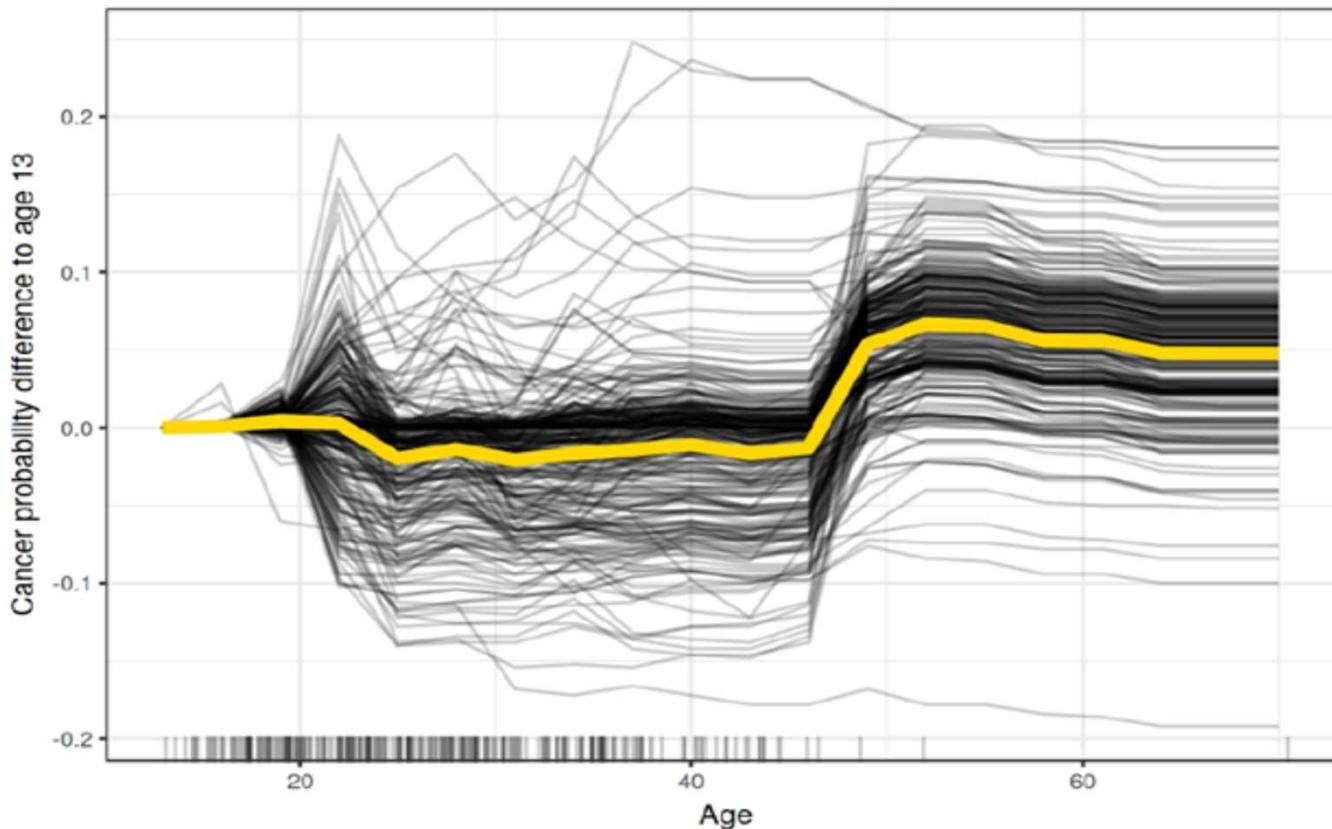
An ICE plot visualizes the dependence of the prediction on a feature for each instance separately, resulting in one line per instance. The values for a line can be computed by keeping all other features the same, creating variants of this instance by replacing the feature's value with other values from the features continuum and making predictions with the black box model for these newly created instances.

The marks on the x-axis indicate the data distribution.



Individual Conditional Expectation (ICE)

There is a problem with ICE plots: Sometimes it can be hard to tell whether the ICE curves differ between individuals because they start at different predictions. A simple solution is to center the curves at a certain point in the feature and display only the difference in the prediction to this point.

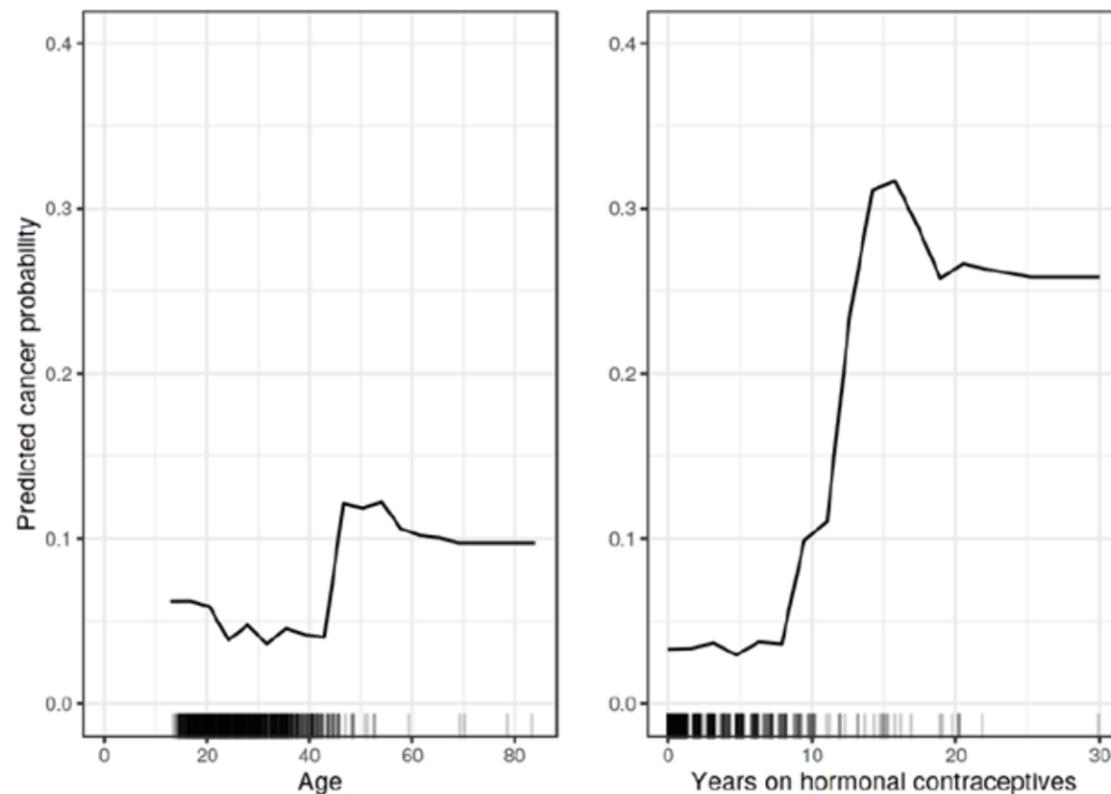


Christoph Molnar: Interpretable Machine Learning, 2019, Leanpub

Partial Dependence Plot (PDP)

The Partial Dependence Plot (PDP) shows the marginal effect one or two features have on the predicted outcome of a machine learning model. A PDP can show whether the relationship between the target and a feature is linear, monotonous or more complex. A PDP is the average of the lines of an ICE plot.

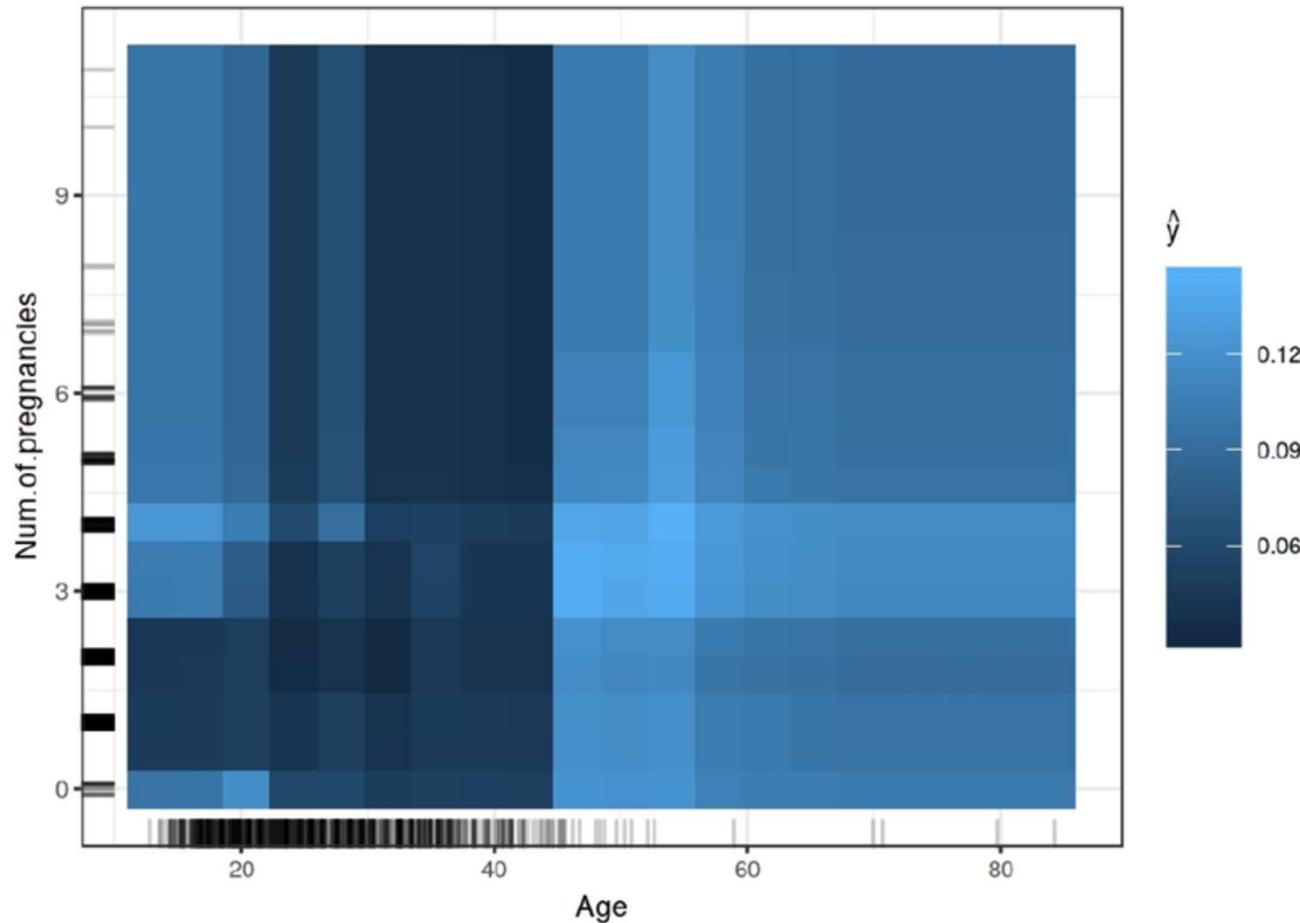
An assumption of the PDP and ICE is that the features are not correlated. If this assumption is violated, the averages calculated for the partial dependence plot will include data points that are very unlikely or even impossible, e.g. a 30 qm flat with 10 rooms.



Christoph Molnar: Interpretable Machine Learning, 2019, Leanpub

Partial Dependence Plot (PDP)

We can also visualize the partial dependence of two features at once:



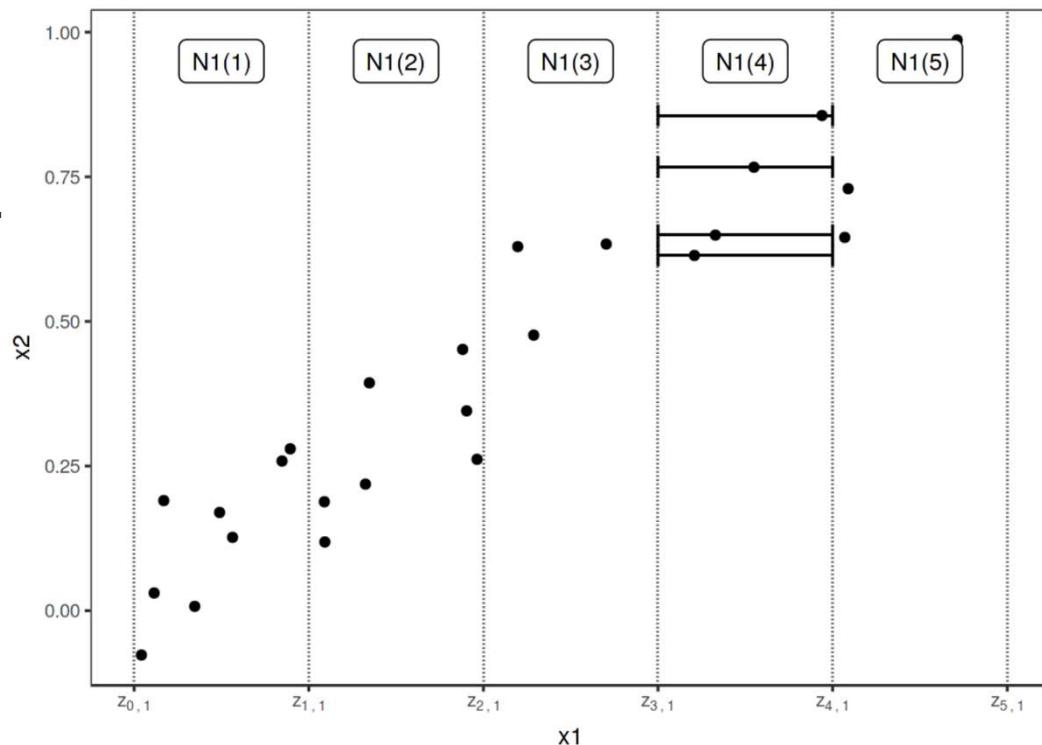
Christoph Molnar: Interpretable Machine Learning, 2019, Leanpub

Accumulated Local Effects (ALE) Plot

Accumulated Local Effects are an alternative to PDPs that can also be used for cored features. PDPs show what the model predicts on average when each data instance has the value v for that feature. It is ignored whether the value v makes sense for all data instances. ALE plots show how the model predictions change in a small "window" of the feature around v for data instances in that window.

Calculation of ALE for feature x_1 , which is correlated with x_2 . First, we divide the feature into intervals (vertical lines). For the data instances (points) in an interval, we calculate the difference in the prediction when we replace the feature with the upper and lower limit of the interval (horizontal lines).

These differences are later accumulated and centered, resulting in the ALE curve.



Global Surrogate

A Global Surrogate model is an interpretable model that is trained to approximate the predictions of a black box model. We can draw conclusions about the black box model by interpreting the surrogate model.

Training a surrogate model is a model-agnostic method, since it does not require any information about the inner workings of the black box model, only access to data and the prediction function is necessary.

Steps to obtain a surrogate model:

- 1. Select a dataset X. This can be the same dataset that was used for training the black box model or a new dataset from the same distribution.**
- 2. For the selected dataset X, get the predictions of the black box model.**
- 3. Select an interpretable model type (linear model, decision tree,...).**
- 4. Train the interpretable model on the dataset X and its predictions.**
- 5. Measure how well the surrogate model replicates the predictions of the black box model.**
- 6. Interpret the surrogate model.**

Christoph Molnar: Interpretable Machine Learning, 2019, Leanpub

Black Box Explanations through Transparent Approximations (BETA)

BETA (Black Box Explanations through Transparent Approximations) is an approach that is model-agnostic, but is limited to the field of classification. The aim is to explain the behavior of a black box classifier globally.

The basic assumption is that the feature space can be divided into subspaces in which similar relationships exist. To describe these, a small number of compact decision rules (If-Then rules) are constructed. Each rule set thus captures the behavior of the black box model in a specific part of the feature space.

Lakkaraju et al.:
Interpretable &
Explorable
Approximations of
Black Box Models,
2017

```
If Age < 50 and Male =Yes:  
  
    If Past-Depression =Yes and Insomnia =No and Melancholy =No, then Healthy  
  
    If Past-Depression =Yes and Insomnia =Yes and Melancholy =Yes and Tiredness =Yes, then Depression  
  
If Age ≥ 50 and Male =No:  
  
    If Family-Depression =Yes and Insomnia =No and Melancholy =Yes and Tiredness =Yes, then Depression  
  
    If Family-Depression =No and Insomnia =No and Melancholy =No and Tiredness =No, then Healthy  
  
Default:  
  
    If Past-Depression =Yes and Tiredness =No and Exercise =No and Insomnia =Yes, then Depression  
  
    If Past-Depression =No and Weight-Gain =Yes and Tiredness =Yes and Melancholy =Yes, then Depression  
  
    If Family-Depression =Yes and Insomnia =Yes and Melancholy =Yes and Tiredness =Yes, then Depression
```

Black Box Explanations through Transparent Approximations (BETA)

To ensure that the generated rules are interpretable, not only a rule-based representation is employed, but also the complexity regarding the number of rules, predicates etc. is minimized. Rule generation is performed using an optimization approach based on an approximate local search.

The subspaces make the complexity of classification decisions more transparent. The evaluation of the behavior in the subspaces requires expert knowledge in the form of context understanding, which is why the involvement of technical experts makes sense. In addition, experts can interactively investigate how the black box model behaves in interesting subspaces.

Human experts are often able not only to explore interesting areas based on their technical understanding, but also to correct them if necessary. Therefore, when searching for explanations, the accuracy of the original model can be improved at the same time.

Lakkaraju et al.: Interpretable & Explorable Approximations of Black Box Models, 2017

Local interpretable model-agnostic explanations (LIME)

Local surrogate models are interpretable models that are used on individual predictions of black box models. Instead of training a global surrogate model, LIME focuses on training local surrogate models to explain individual predictions and thus the prediction results of individual data instances.

LIME tests what happens to the predictions when the input data is varied. For each selected data instance, LIME generates a new data set consisting of multiple data instances composed of permuted values of the features and the resulting predictions of the black box model.

LIME then trains an *interpretable* model on this new data set, whereby each of the generated data instances is weighted with its similarity to the original data instance.

Christoph Molnar: Interpretable Machine Learning, 2019, Leanpub

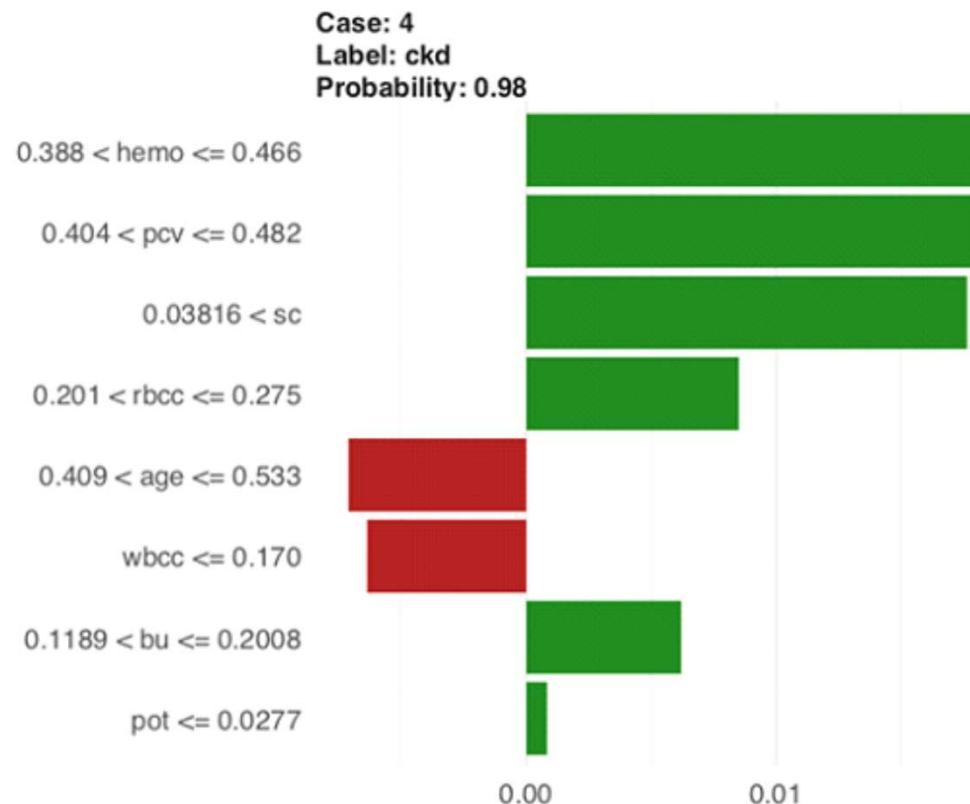
Local interpretable model-agnostic explanations (LIME)

The procedure is as follows:

- 1. Selection of the data instances of interest**
- 2. For each of these instances: Generation of a new data set by (light) permutation of the input values, calculation of the similarity of the new permuted instance to the original instance, and calculation of the corresponding predictions using the black box model.**
- 3. For each of the new datasets: Train a interpretable model (usually a linear model, e.g. OLS, Ridge or Lasso) using the similarities as weights.**
- 4. Interpretation of the resulting local models, if necessary concentrating on the features with the greatest parameter weights in the regression.**

Christoph Molnar: Interpretable Machine Learning, 2019, Leanpub

Local interpretable model-agnostic explanations (LIME)



<https://blog.codecentric.de/en/2018/01/look-beyond-accuracy-improve-trust-machine-learning/>

The facet shows the explanation for the prediction of an individual patient or instance. The header of the facet gives the case number (here the patient ID), which class label was predicted and with what probability. In this case, the instance describes case number 4 which was classified as “ckd” with 98 % probability. Below the header we find a bar-plot for the top 8 most important features; the length of each bar shows the weight of the feature, positive weights support a prediction, negative weights contradict it. The bar-plot shows that the hemoglobin value was between 0.388 and 0.466, which supports the classification as “ckd”; packed cell volume (pcv), serum creatinine (sc), etc. similarly support the classification as “ckd”.

Shapley Values

The Shapley Values are the average contributions that the features of a data instance have in the corresponding prediction.

The Shapley value is a method from coalitional game theory. In this terminology, a prediction can be explained by assuming that each feature value of the instance is a player in a game where the prediction is the payout. The Shapley value tells us how to fairly distribute the "payout" among the features.

The "game" is the prediction task for a single instance of the dataset. The "gain" is the actual prediction for this instance minus the average prediction for all instances. The "players" are the feature values of the instance that collaborate to receive the gain.

The Shapley value is the average marginal contribution of a feature value across all possible coalitions.

Christoph Molnar: Interpretable Machine Learning, 2019, Leanpub

Shapley Values

Let $I=\{1,2,\dots, n\}$ be the set of all n features of a model. Each non-empty subset of I is a coalition in this context.

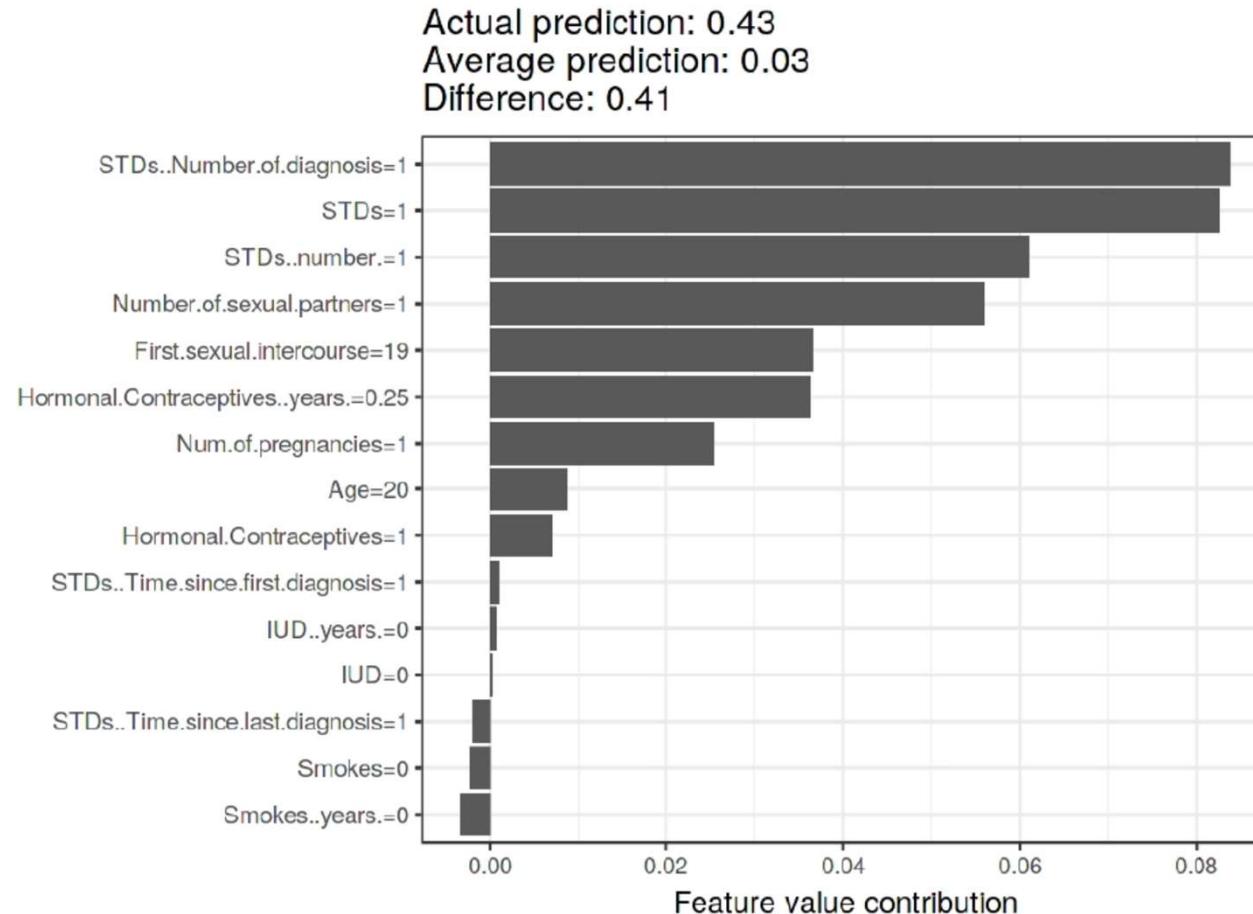
For each of these coalitions the prediction value is calculated and then the differences to the average are taken to obtain the marginal contribution.

The Shapley value is the average of all the marginal contributions to all possible coalitions.

The values of features that are not in a coalition are replaced by random feature values from the data set to enable a prediction from the model.

Christoph Molnar: Interpretable Machine Learning, 2019, Leanpub

Shapley Values



Christoph Molnar: Interpretable Machine Learning, 2019, Leanpub

Shapley values for a woman in the cervical cancer dataset. With a prediction of 0.43, this woman's cancer probability is 0.41 above the average prediction of 0.03. The number of diagnosed STDs increased the probability the most. The sum of contributions yields the difference between actual and average prediction (0.41).

Segmentation

Introductory Example

Assume you are a wholesale distributor and each row of your dataset corresponds to a customer showing the following attributes:

- 1) FRESH: annual spending on fresh products (Continuous);
- 2) MILK: annual spending on milk products (Continuous);
- 3) GROCERY: annual spending on grocery products (Continuous);
- 4) FROZEN: annual spending on frozen products (Continuous)
- 5) DETERGENTS_PAPER: annual spending on detergents and paper products (Continuous)
- 6) DELICATESSEN: annual spending on delicatessen products (Continuous);
- 7) CHANNEL: customers buying channel (Nominal)
- 8) REGION: customers region (Nominal)

Your goal is to segment the users. That means finding similar types of users and bunching them together.

Why would you want to do this?

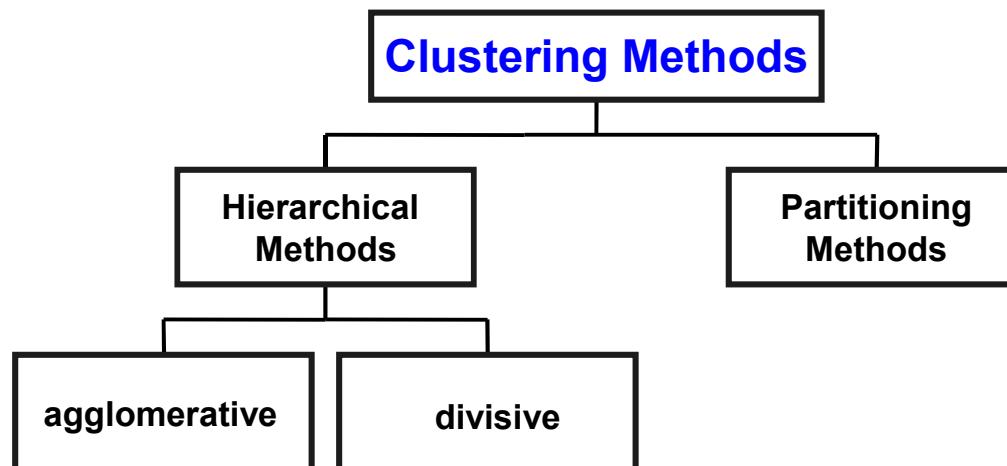
- You might want to give different users different experiences. Marketing often does this; for example, to offer toner to people who are known to own printers.
- You might have a model that works better for specific groups. Or you might have different models for different groups.

Clusteranalysis

Cluster Analysis

Cluster analysis is a type of multivariate statistical analysis. It is used to group data into separate clusters. The main objective of clustering is to find similarities between data objects, and then group similar objects together to assist in understanding relationships that might exist among them. Cluster analysis is based on a mathematical formulation of a measure of similarity.

There are different types of cluster analysis methods:



Partitioning Cluster Methods

The partitioning cluster methods divide the data into a predetermined number of clusters. The most popular technique is the K-Means algorithm.

Given a set of observations (x_1, x_2, \dots, x_n) , where each observation is a m -dimensional real vector, k -means clustering aims to partition the n observations into $(k \leq n)$ segments $S = \{S_1, S_2, \dots, S_k\}$ so as to minimize the within-cluster sum of squares (WCSS).

The objective is to find

$$\operatorname{argmin}_S \sum_{i=1}^k \sum_{x_j \in S_i} (x_j - \bar{x}_i)^2$$

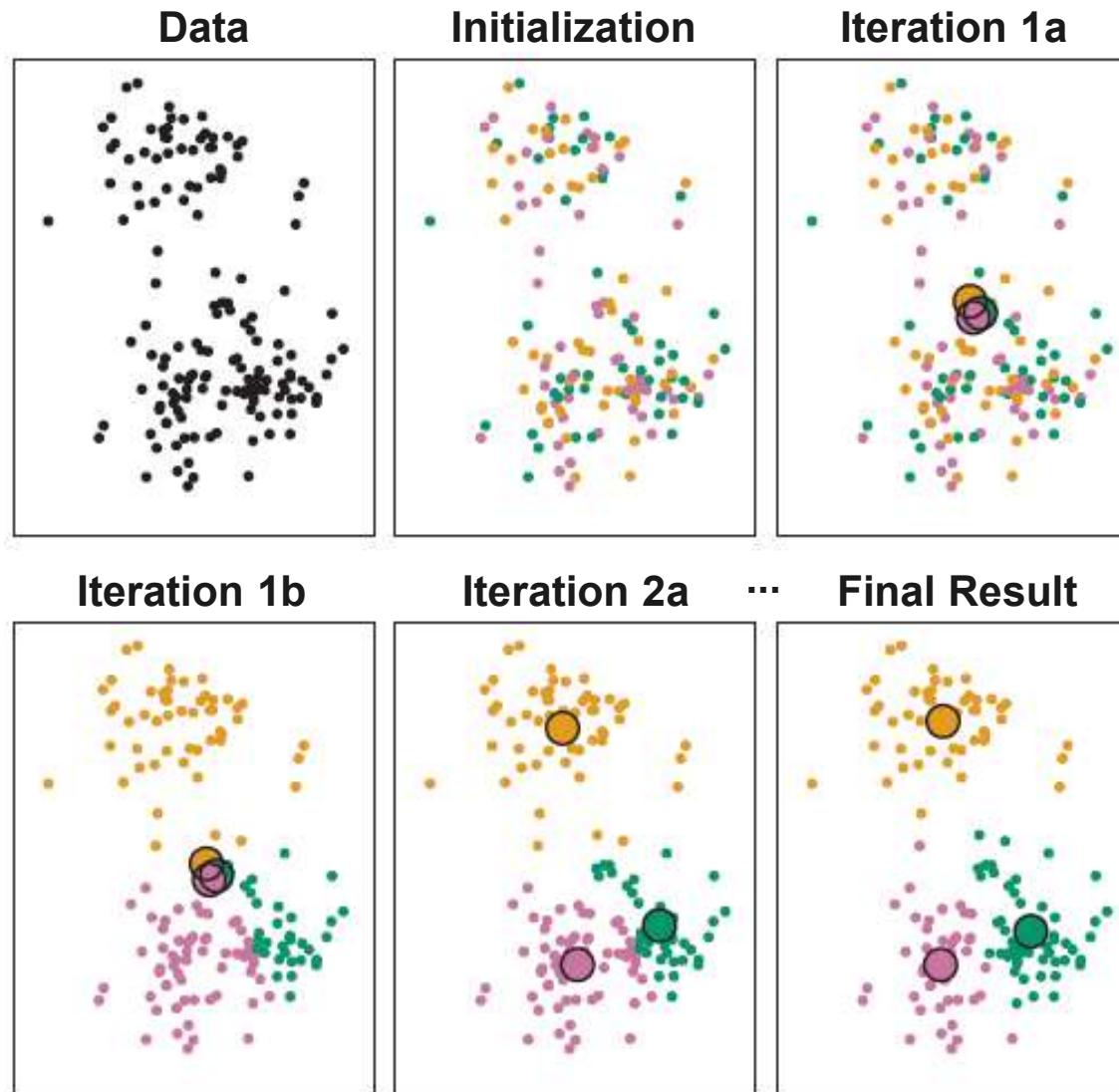
where \bar{x}_i is the mean of points in S_i .

K-Means

Procedure of K-Means:

- Step 1: Randomly partition the data objects into k clusters.**
- Step 2: Calculate the cluster centroids.**
- Step 3: Calculate the distance from every data point to all centroids**
- Step 4: If a data point is closest to its own centroid, leave it where it is. If the data point is not closest to its own centroid, assign it to the cluster with the closest centroid.**
- Step 5: Repeat the step 2 to 4 until a complete pass through of all the data points results in no data point changing from one cluster to another.**

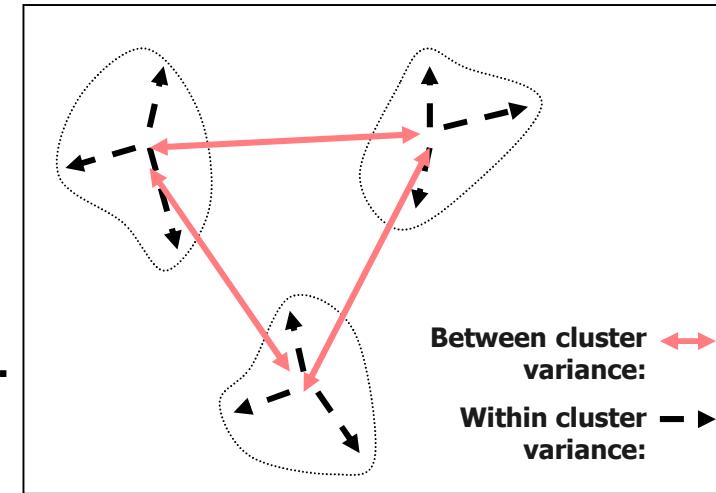
Example of a K-Means Cluster Analysis



Finding the Optimal Number of Clusters (I)

The aim of the cluster analysis is the segmentation of objects into clusters, which are preferably homogeneous in it selves and heterogeneous to each other. The less variance exists within the clusters and the more variance exists between the clusters, the better is the number of clusters.

$$\text{Total variance: } V_{\text{tot}} = \sum_{j=1}^n \sum_{i=1}^m (x_{ij} - \bar{x}_i)^2$$



$$\text{Accumulated variance within the } k \text{ clusters: } V_{\text{in}}^k = \sum_{i=1}^m \sum_{j=1}^k \sum_{l=1, l \in c_k}^{n_k} (x_{lj} - \bar{x}_{c_k, i})^2$$

This results in the variance between the clusters: $V_{\text{betw}}^k = V_{\text{tot}} - V_{\text{in}}^k$

with n = number of objects

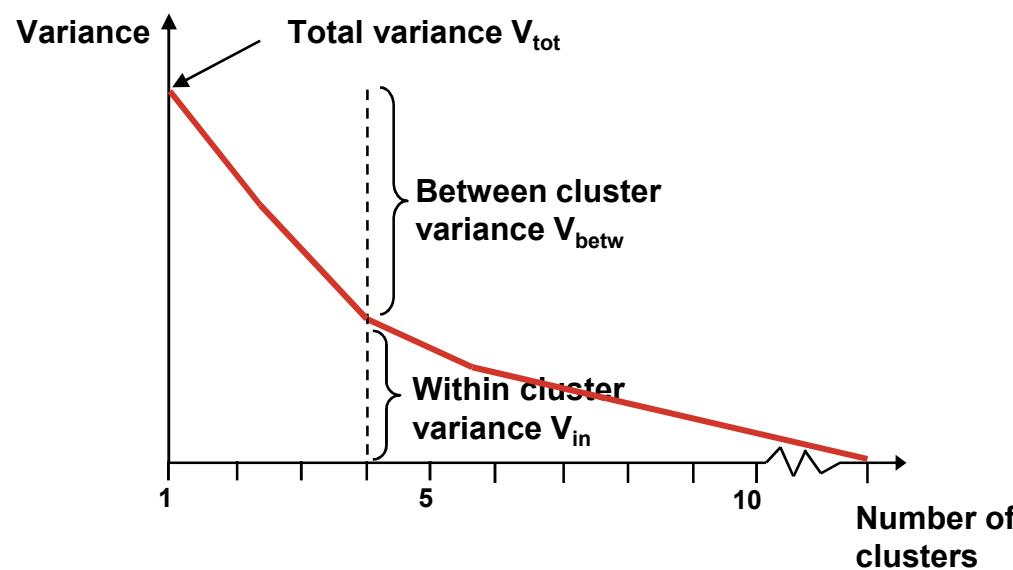
m = number of attributes

n_k = number of objects in cluster k

c_k = cluster k

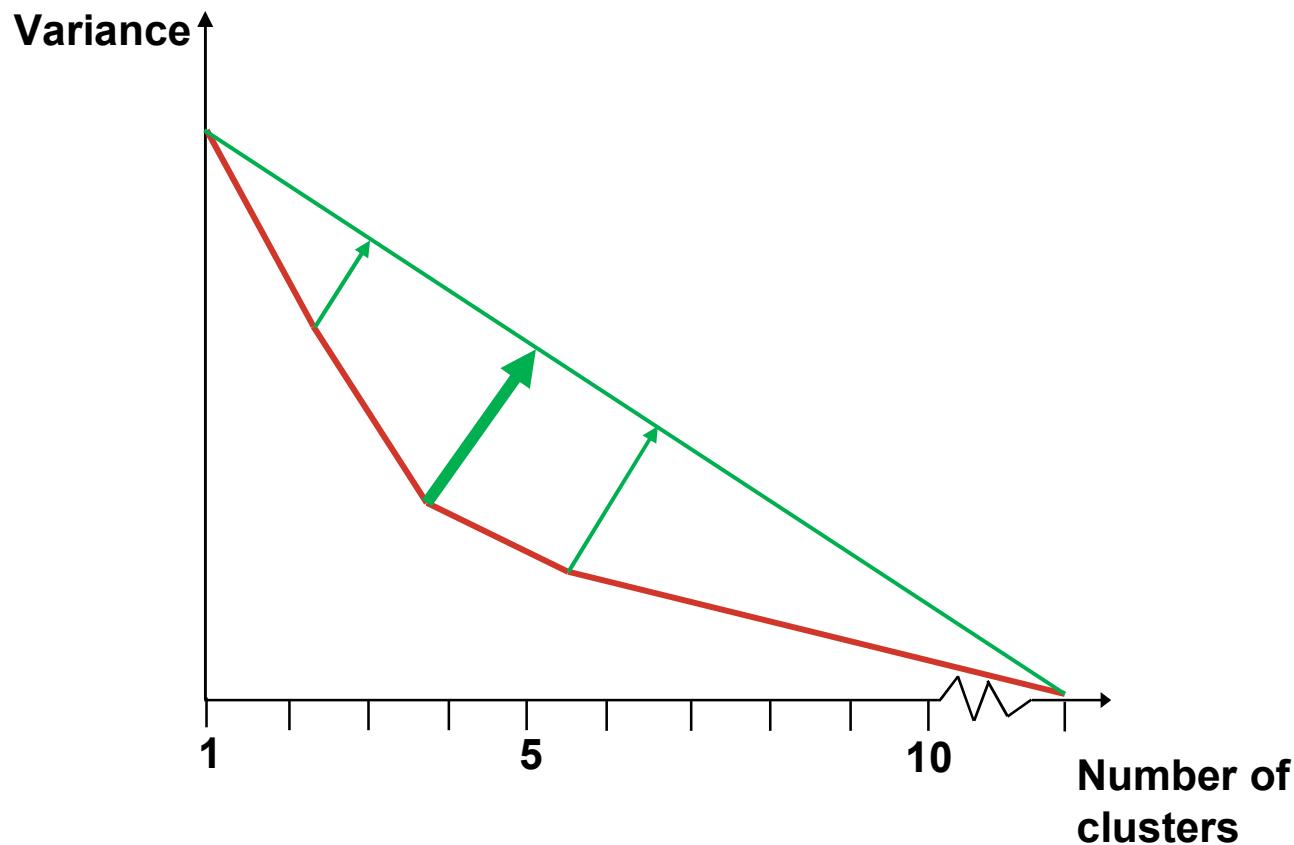
Finding the Optimal Number of Clusters (II)

If you put V_{in} on the ordinate and the number of cluster k on the abscissa, it often results in a curve with one or several kinks. At the point where exists the (first) significant kink, you can find the optimal number of clusters:



Finding the Optimal Number of Clusters (III)

Instead of visually identifying the optimal cluster number, we can calculate the distances from the points on the elbow curve to a straight line linking the first and the last point on the curve. The cluster number with the largest distance is then chosen as the one with the strongest kink.

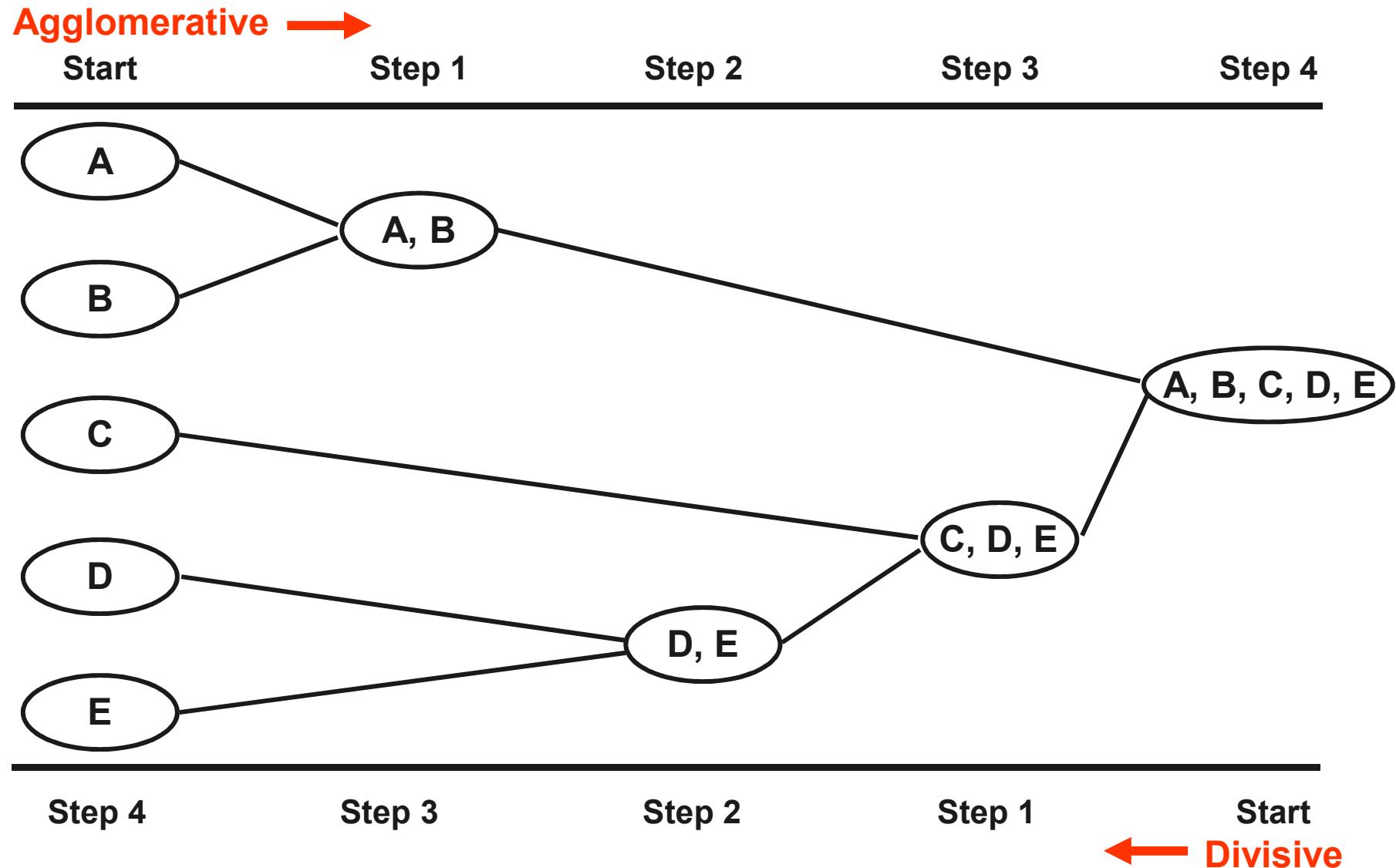


Hierarchical Cluster Methods

There are two types of hierarchical cluster methods:

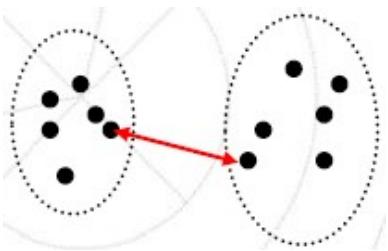
- **Agglomerative hierarchical clustering** is a bottom-up clustering method. It starts with every single data object in a single cluster. Then, in each iteration, it agglomerates (merges) the closest pair of clusters by satisfying some similarity criteria, until all of the data is in one cluster.
- **Divisive hierarchical clustering** is a top-down clustering method. It works in a similar way to agglomerative clustering but in the opposite direction. This method starts with a single cluster containing all data objects, and then successively splits resulting clusters until only clusters of individual data objects remain.

Process of the Hierarchical Cluster Analysis



Measuring Similarity between Clusters (I)

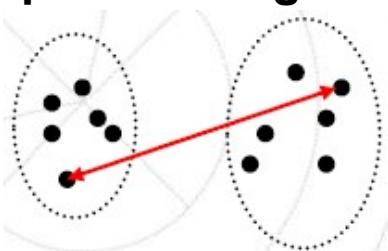
Single Linkage:



Distance between two clusters is the distance between the closest points:

$$d(C_i, C_j) = \min_{x \in C_i, y \in C_j} \{d(x, y)\}$$

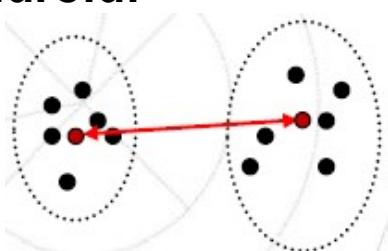
Complete Linkage:



Distance between two clusters is the distance between the farthest pair of points:

$$d(C_i, C_j) = \max_{x \in C_i, y \in C_j} \{d(x, y)\}$$

Cendroid:

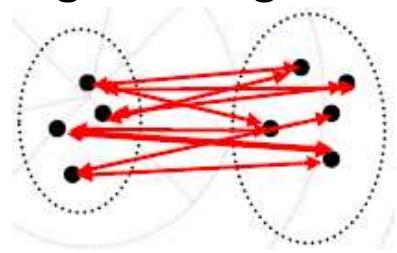


Distance between two clusters i and j is the distance between their cendroids:

$$d(C_i, C_j) = d(\bar{x}_i, \bar{x}_j)$$

Measuring Similarity between Clusters (II)

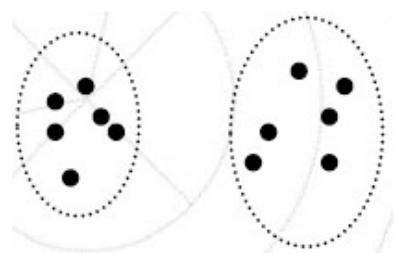
Average Linkage:



Distance between clusters is the average distance between the cluster points:

$$d(C_i, C_j) = \frac{1}{n_i \cdot n_j} \sum_{x \in C_i} \sum_{y \in C_j} d(x, y)$$

Ward's Method / Minimum Variance Method (only Agglomerative):



Ward's minimum variance criterion minimizes the total within-cluster variance. At each step the pair of clusters is merged that leads to minimum increase in total within-cluster variance after merging. This can be calculated as the square of the distance between cluster means divided by the sum of the reciprocals of the number of observations in each cluster:

$$d(C_i, C_j) = \frac{1}{\frac{1}{n_i} + \frac{1}{n_j}} (d(\bar{x}_i, \bar{x}_j))^2$$

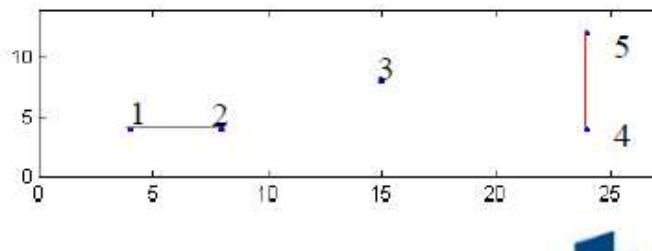
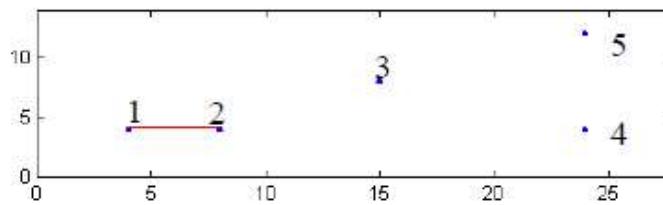
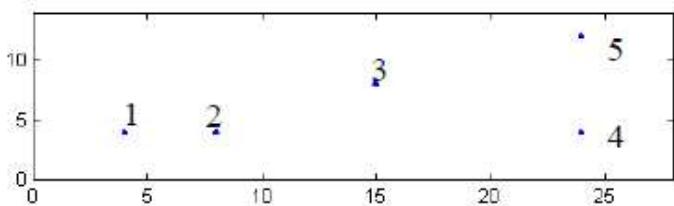
For a comparison of the methods see: Ferreira, L.; Hitchcock, D. B. (2009): A Comparison of Hierarchical Methods for Clustering Functional Data, http://people.stat.sc.edu/Hitchcock/compare_hier_fda.pdf

Single Linkage Example (I)

| | x | y | | | |
|---|----|---|--|--|--|
| 1 | 4 | 4 | | | |
| 2 | 8 | 4 | | | |
| 3 | 15 | 8 | | | |
| 4 | 24 | 4 | | | |

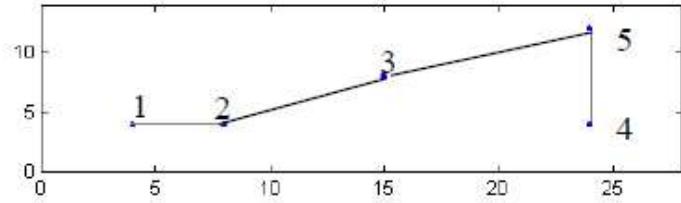
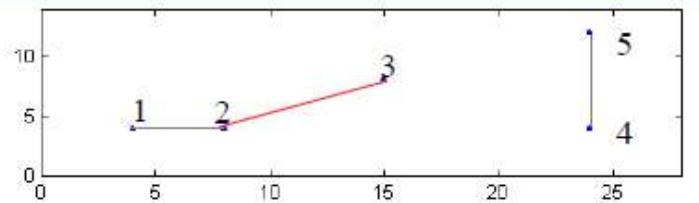
| | 1 | 2 | 3 | 4 | 5 |
|---|------|-----|------|-----|------|
| 1 | - | 4 | 11.7 | 20 | 21.5 |
| 2 | 4 | - | 8.1 | 16 | 17.9 |
| 3 | 11.7 | 8.1 | - | 9.8 | 9.8 |

| | 1,2 | 3 | 4 | 5 |
|-----|------|-----|-----|------|
| 1,2 | - | 8.1 | 16 | 17.9 |
| 3 | 8.1 | - | 9.8 | 9.8 |
| 4 | 16 | 9.8 | - | 8 |
| 5 | 17.9 | 9.8 | 8 | - |

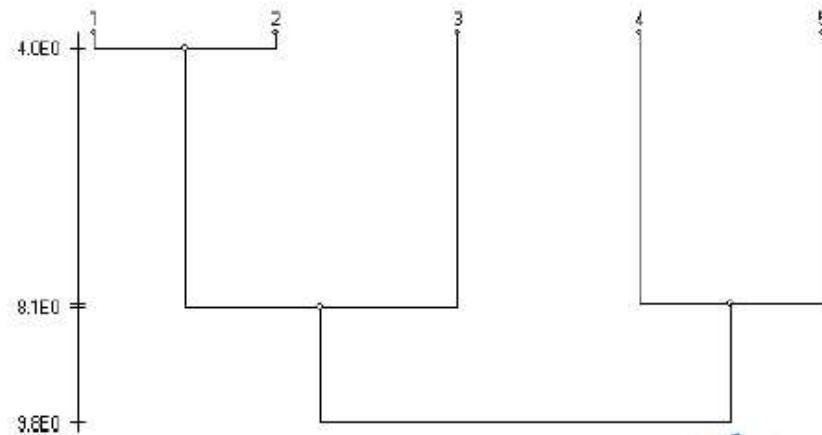


Source: Fred, Ana: Unsupervised Learning, Universidade Técnica de Lisboa

Single Linkage Example (II)



Dendrogram

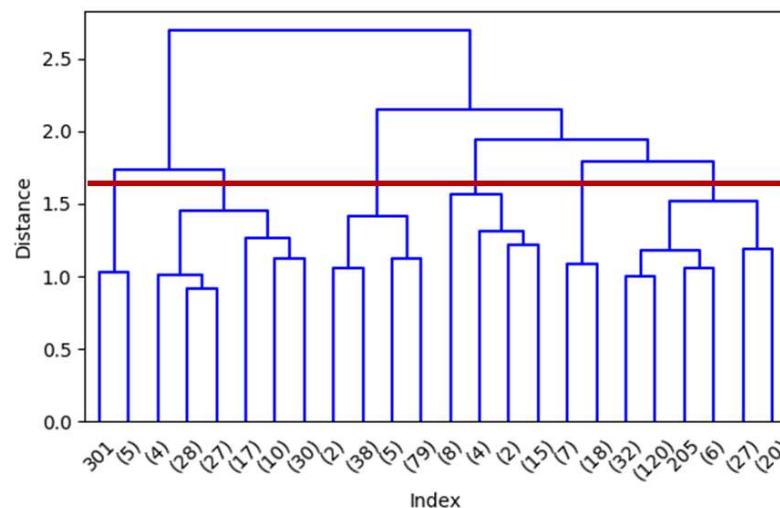


Source: Fred, Ana: Unsupervised Learning, Universidade Técnica de Lisboa

Dendrogram

A dendrogram is a tree diagram frequently used to illustrate the arrangement of the clusters produced by hierarchical clustering. The y-axis represents the value of this distance metric (e.g. euclidean distance) between the clusters.

In a dendrogram the widths of the horizontal lines give an impression about the dissimilarity of the merging object. Thus, a good cluster number might be at a point from where the width of the following horizontal lines is significantly smaller in length. The red line in the graph below shows such a point:



Counting the points that cut this line might be a good answer for the number of clusters the data can have. It is the number 6 in this case.

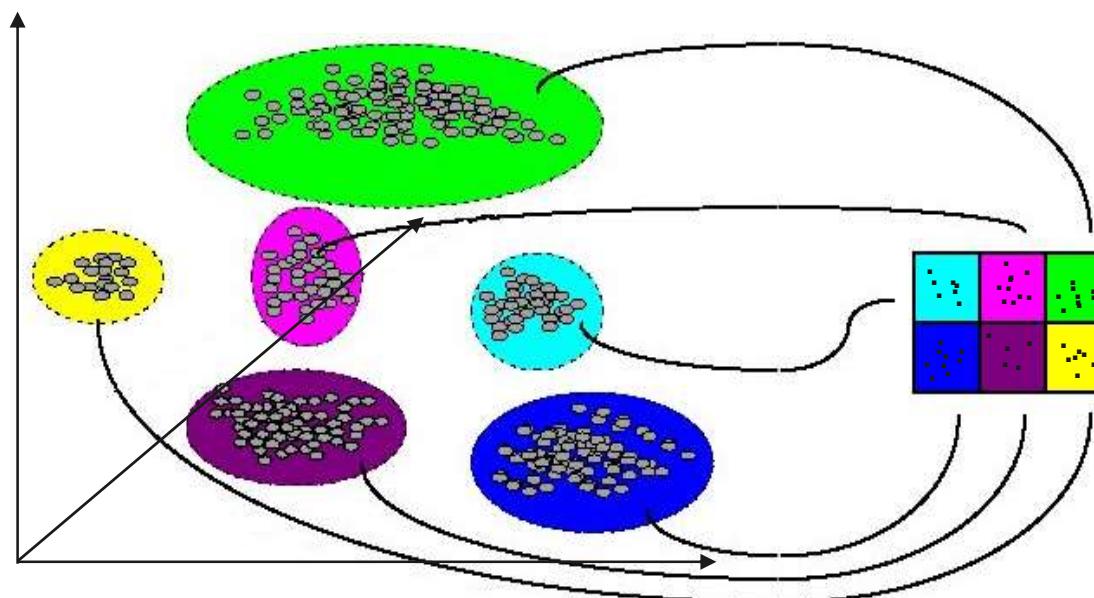
Dimensionality Reduction

Self-organizing Maps

Self-Organizing Maps

A Self-Organizing Map (SOM) is a type of artificial neural network that is trained using unsupervised learning to produce a low-dimensional representation of the input space of high-dimensional data, called a map.

It defines an ordered mapping, a kind of projection from a set of given data items onto a regular, usually two-dimensional grid.



The Training Process

- 1. Generation of a grid with randomly chosen vectors as nodes**
- 2. Random selection of a (real) input object**
- 3. Calculation of the winner node**

$$\|x - m_w\| = \min_i (\|x - m_i\|)$$

m_i = vector of attributes of neuron i
 x = vector of attributes of object
 $h_{wi}(t)$ = neighborhood function

- 4. Adjustment of the winner node**

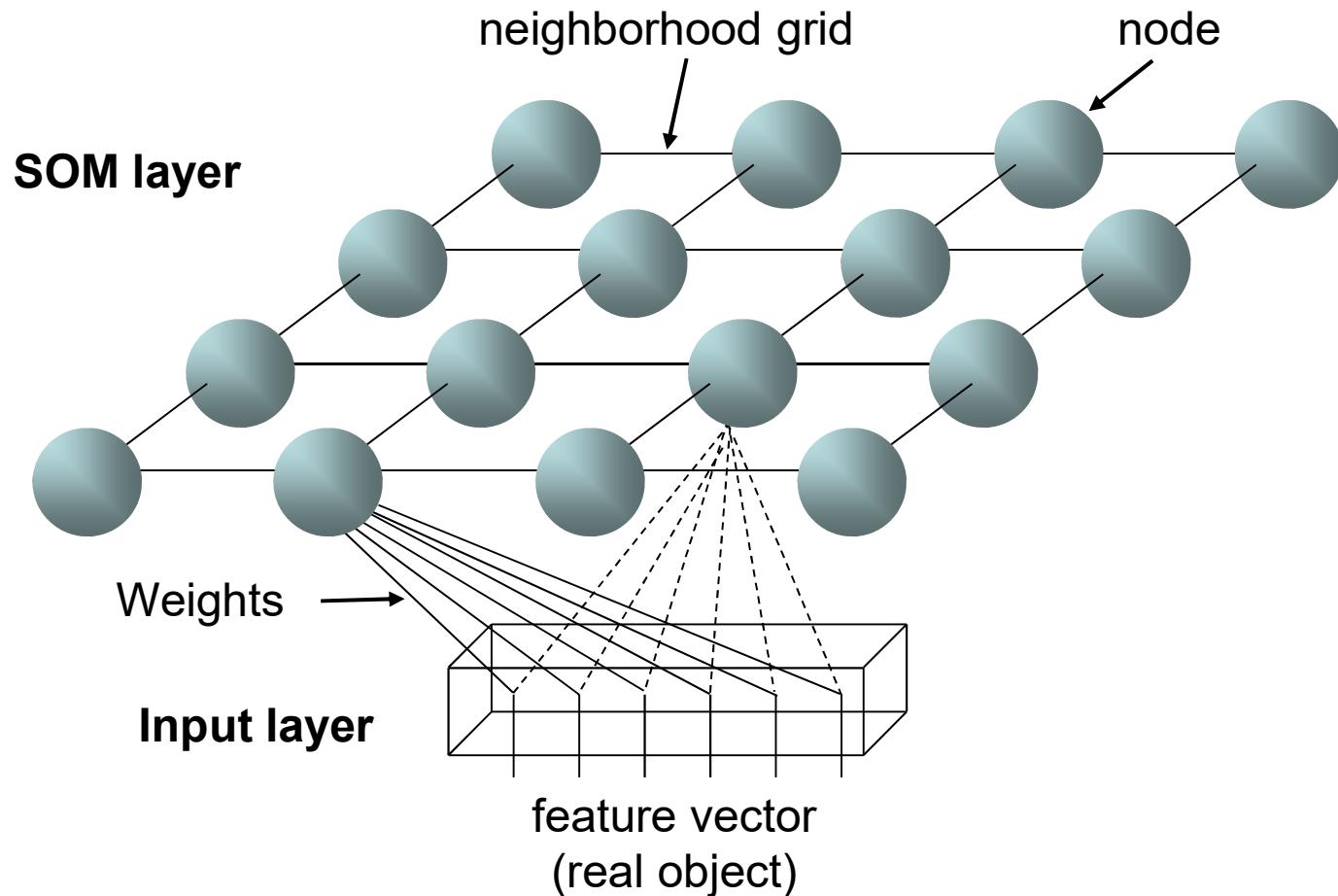
$$m_i(t+1) = m_i(t) + h_{wi}(t) \cdot [x(t) - m_i(t)]$$

- 5. Identification of the neighbors**

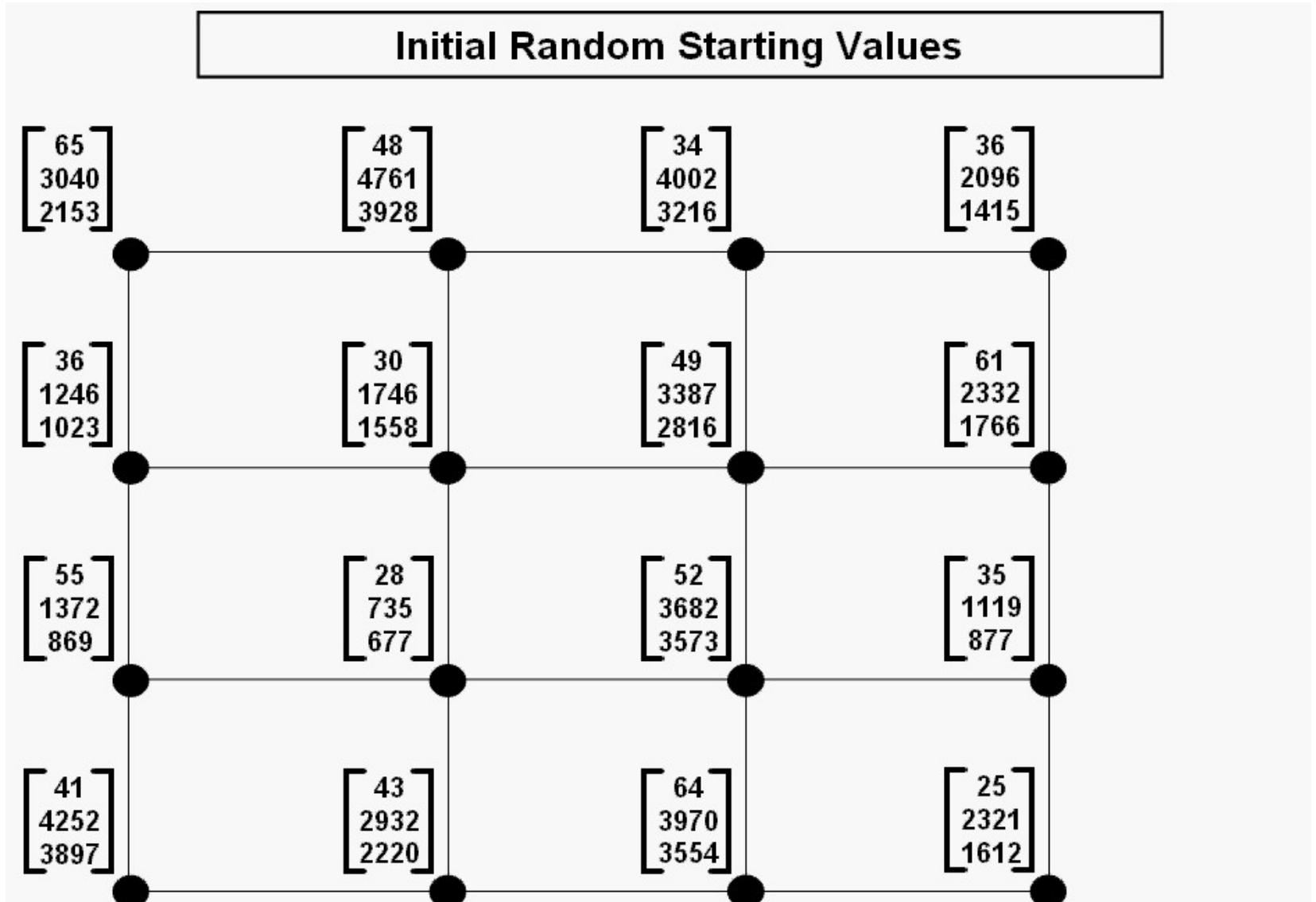
- 6. Adjustment of the neighbors**

- 7. Repeat with 2. or stop if the maximal number of iterations is performed**

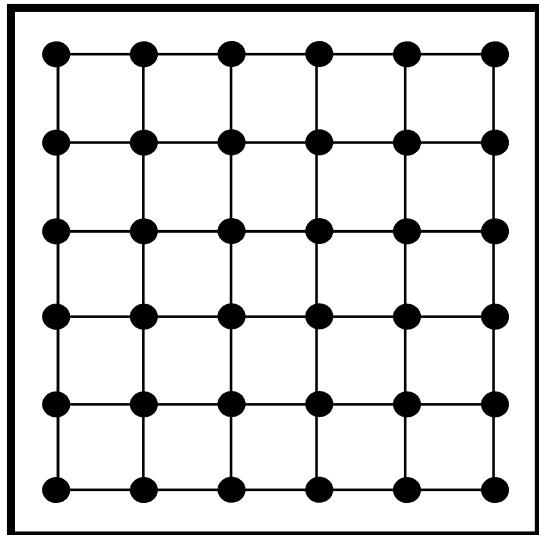
Architecture of a SOM



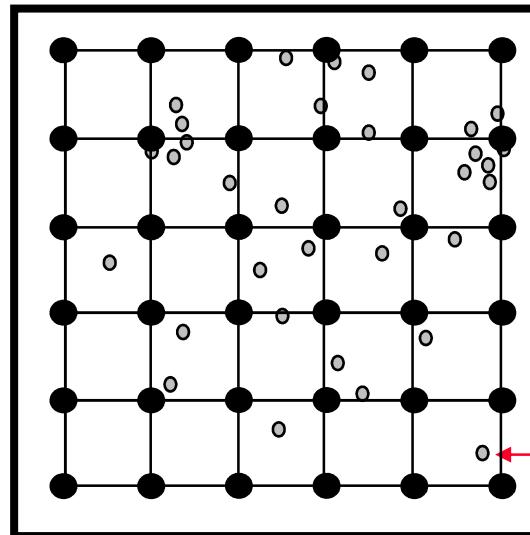
Visualization of the Algorithm



Start and Result

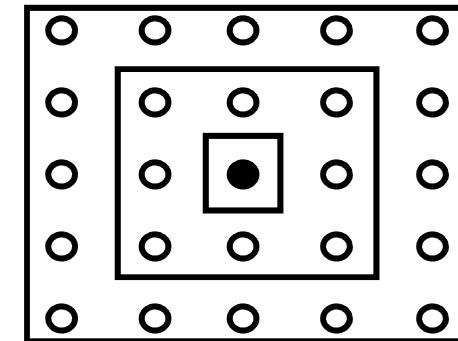
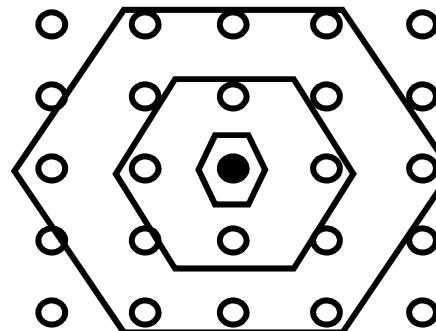
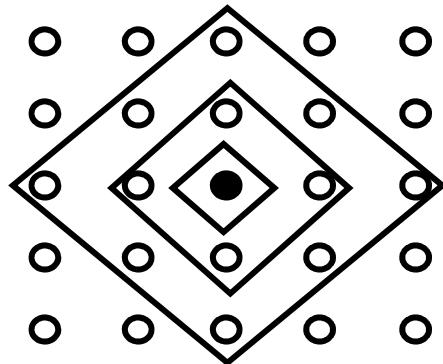


After Initialization



After Training

Types of Neighborhood Functions



Bubble Kernel: $h_{Gi}(t) = \begin{cases} \alpha(t) & \text{if } \|r_G - r_i\| < \sigma^2(t) \\ 0 & \text{otherwise} \end{cases}$

Gaussian Kernel: $h_{Wi}(t) = \alpha(t) \cdot \exp\left(-\frac{\|r_w - r_i\|^2}{2 \cdot \sigma^2(t)}\right)$

r_i = position of neurons i in the map

$\alpha(t)$ = learning rate (decreasing over time)

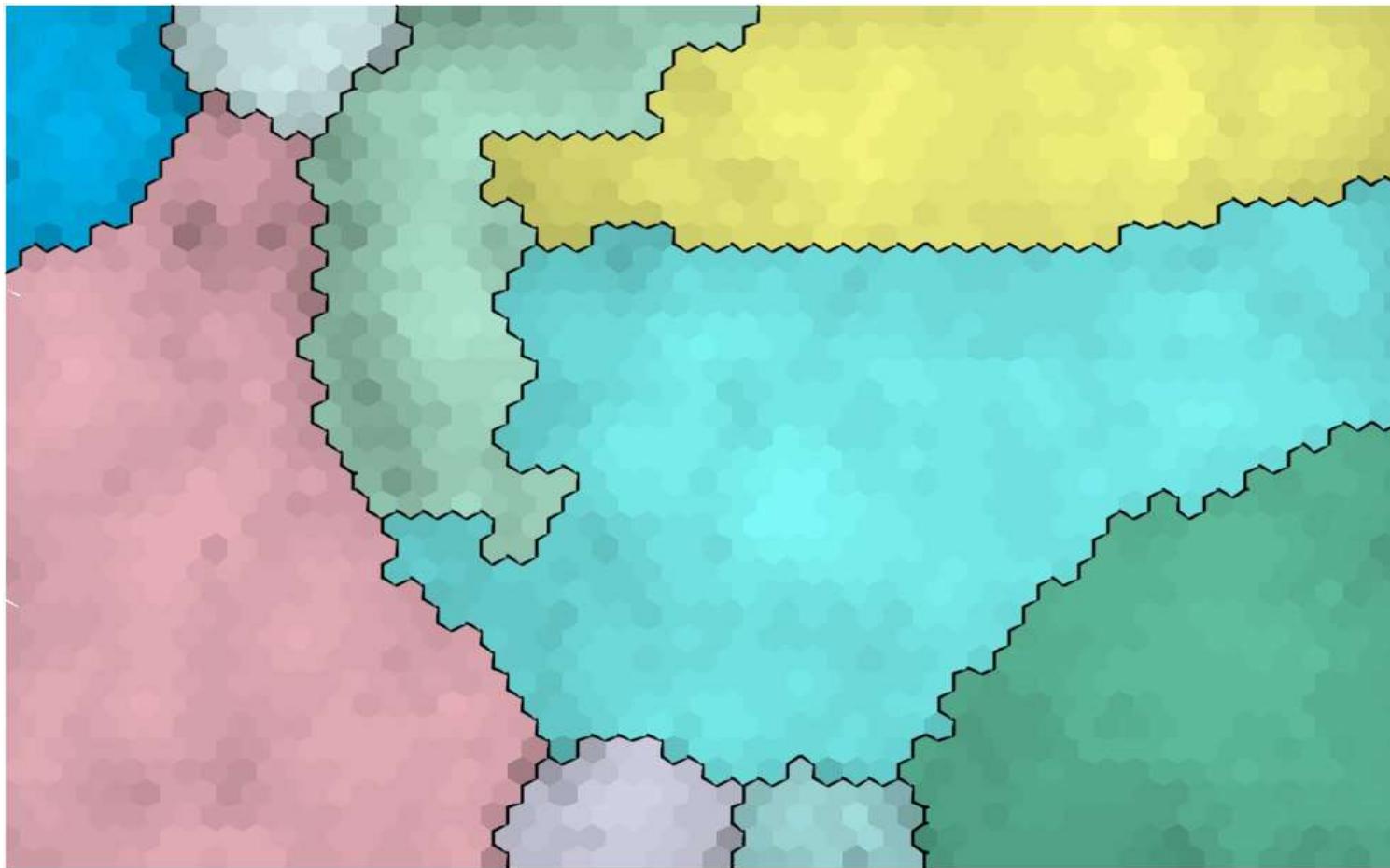
$\sigma(t)$ = neighbourhood radius (decreasing over time)

Clustering of the Self-Organizing Map

In principle, a trained SOM can be interpreted as a system of clusters, where each node represents one segment. In this case, the number of nodes must be predetermined according to the desired number of segments. The usually few number of nodes in this case involve the risk of information loss.

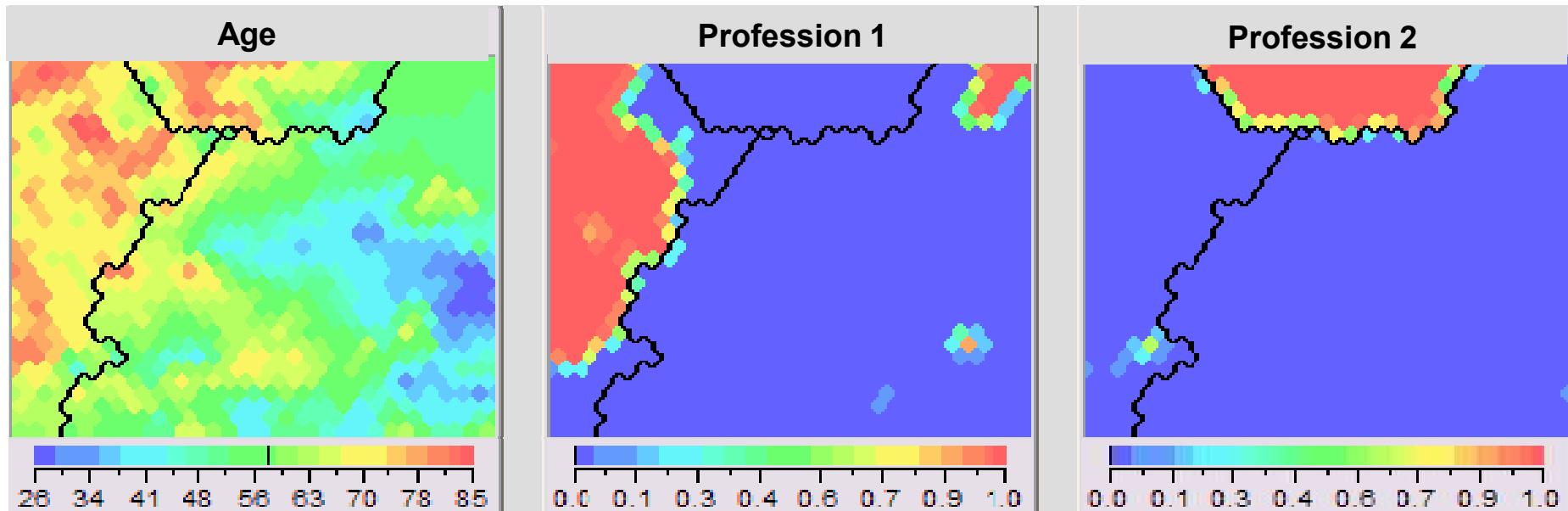
To overcome this problem, the clustering can be carried out using a two-stage approach. Here the data set is first segmented using the SOM, and then, the SOM is clustered using a traditional clustering algorithm. The most important benefit of this procedure is that computational load decreases considerably, making it possible to cluster large data sets in a limited time having better visual properties.

Example of a trained Map with Segments



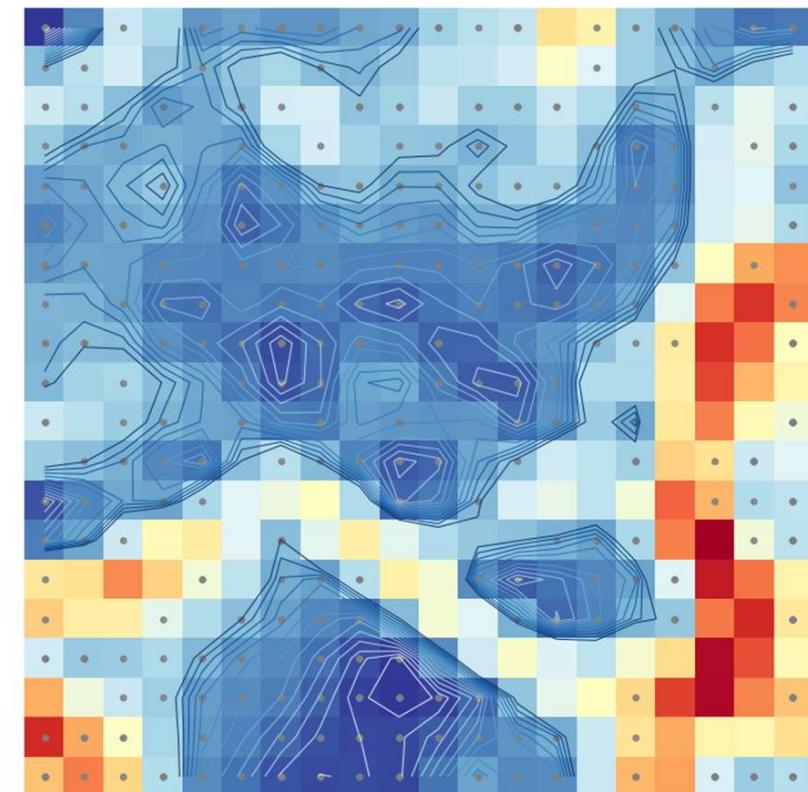
Heatmaps

Visualization of the separability of features:

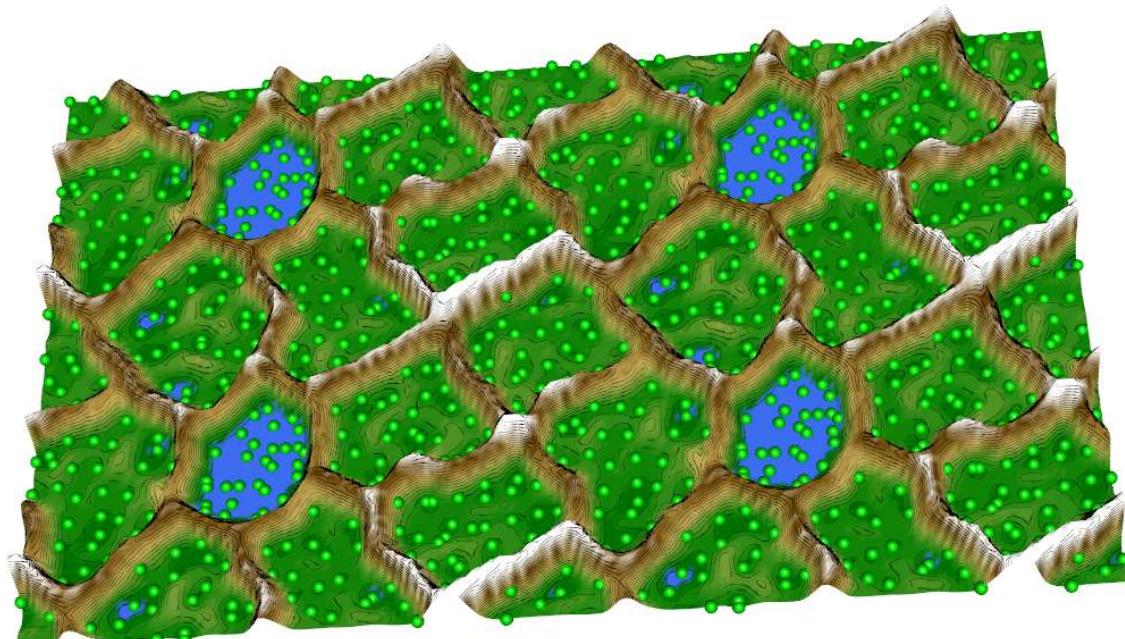
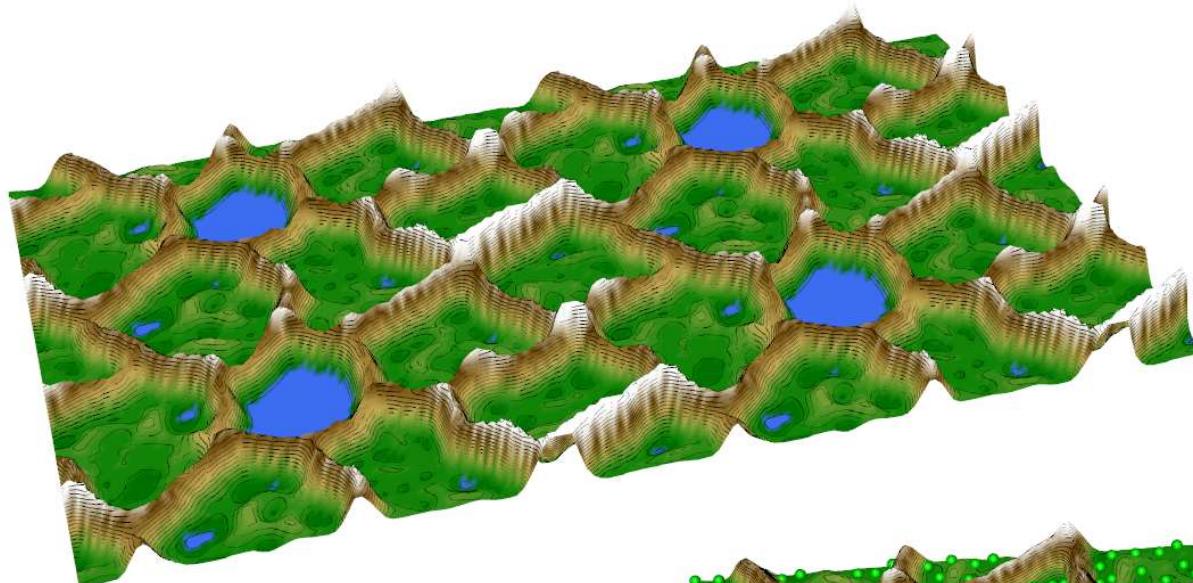


U-Matrix

A U-matrix (unified distance matrix) representation of the Self-Organizing Map visualizes the distances between the nodes. The distance between the adjacent nodes is calculated and presented with different colorings. A dark coloring between corresponds to a large distance and thus a gap between the real objects in the input space. A light coloring between the nodes signifies that the real objects are close to each other in the input space. Dark colors can be interpreted as hills and light colors as valleys. Thus, light areas can be thought as clusters and dark areas as cluster separators. This can be a helpful presentation when one tries to find clusters in the input data without having any a priori information about the clusters.

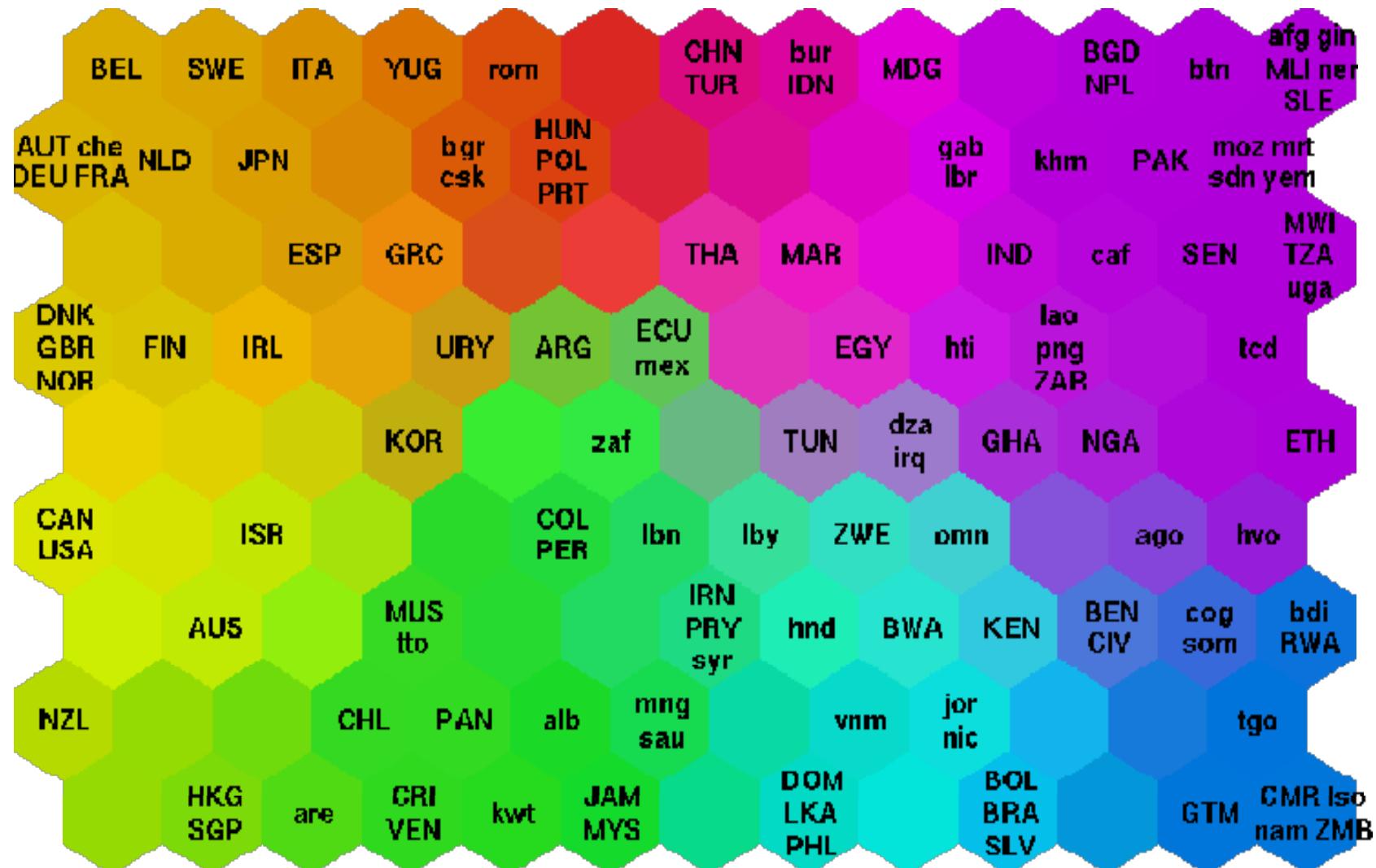


3D-Representation of a U-Matrix

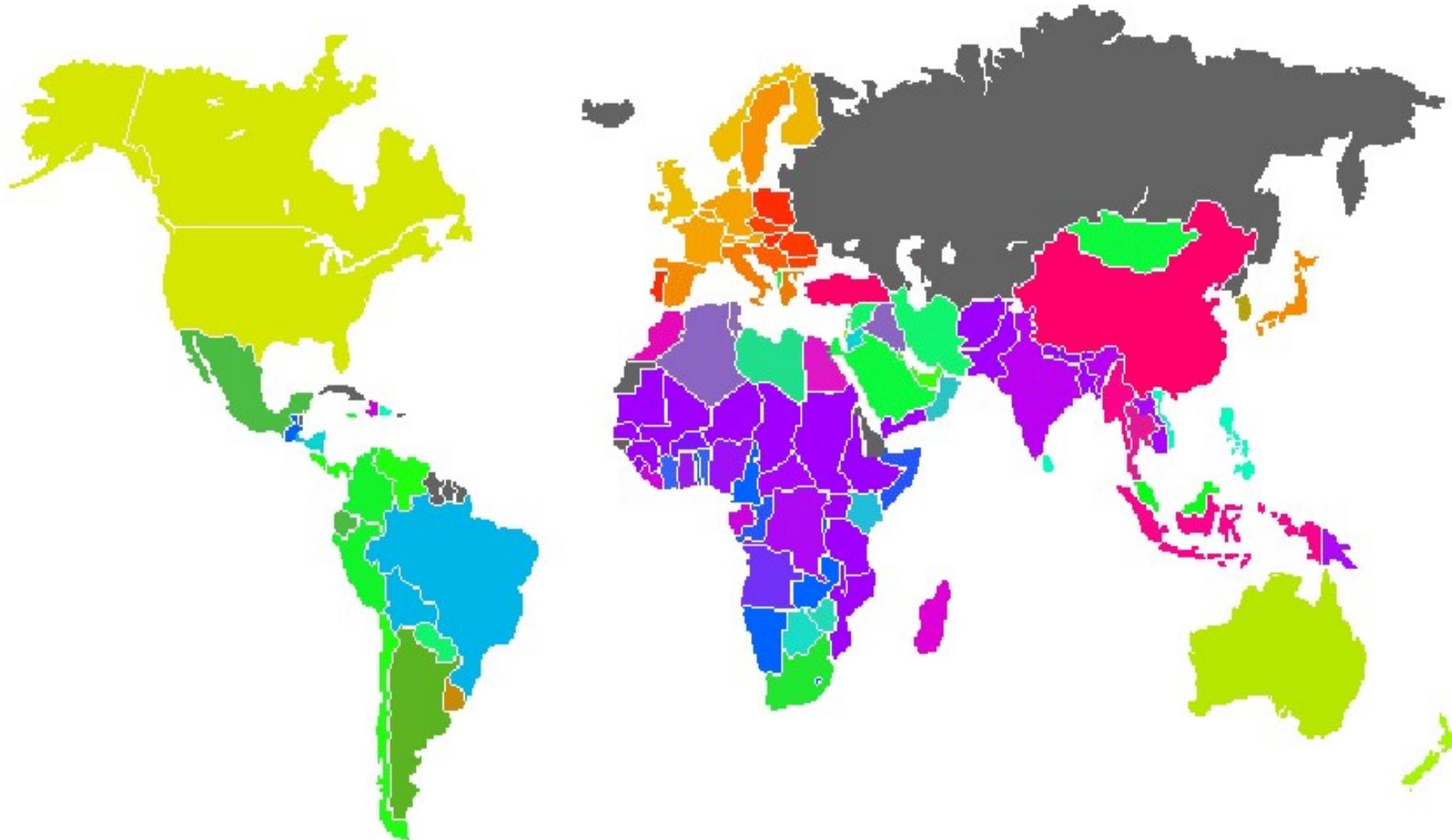


including the real objects:

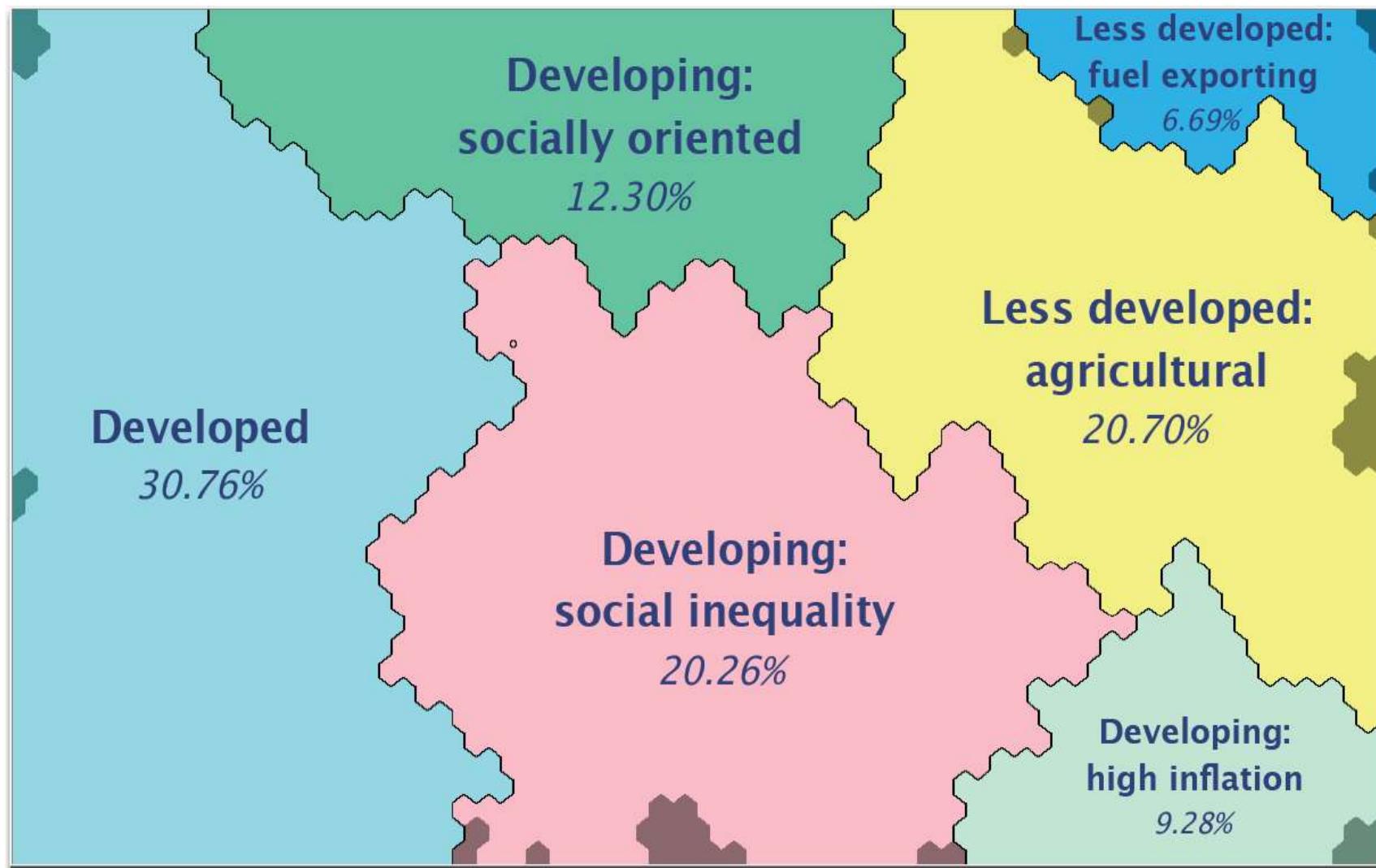
Representation of Countries by Quality of Life (I)



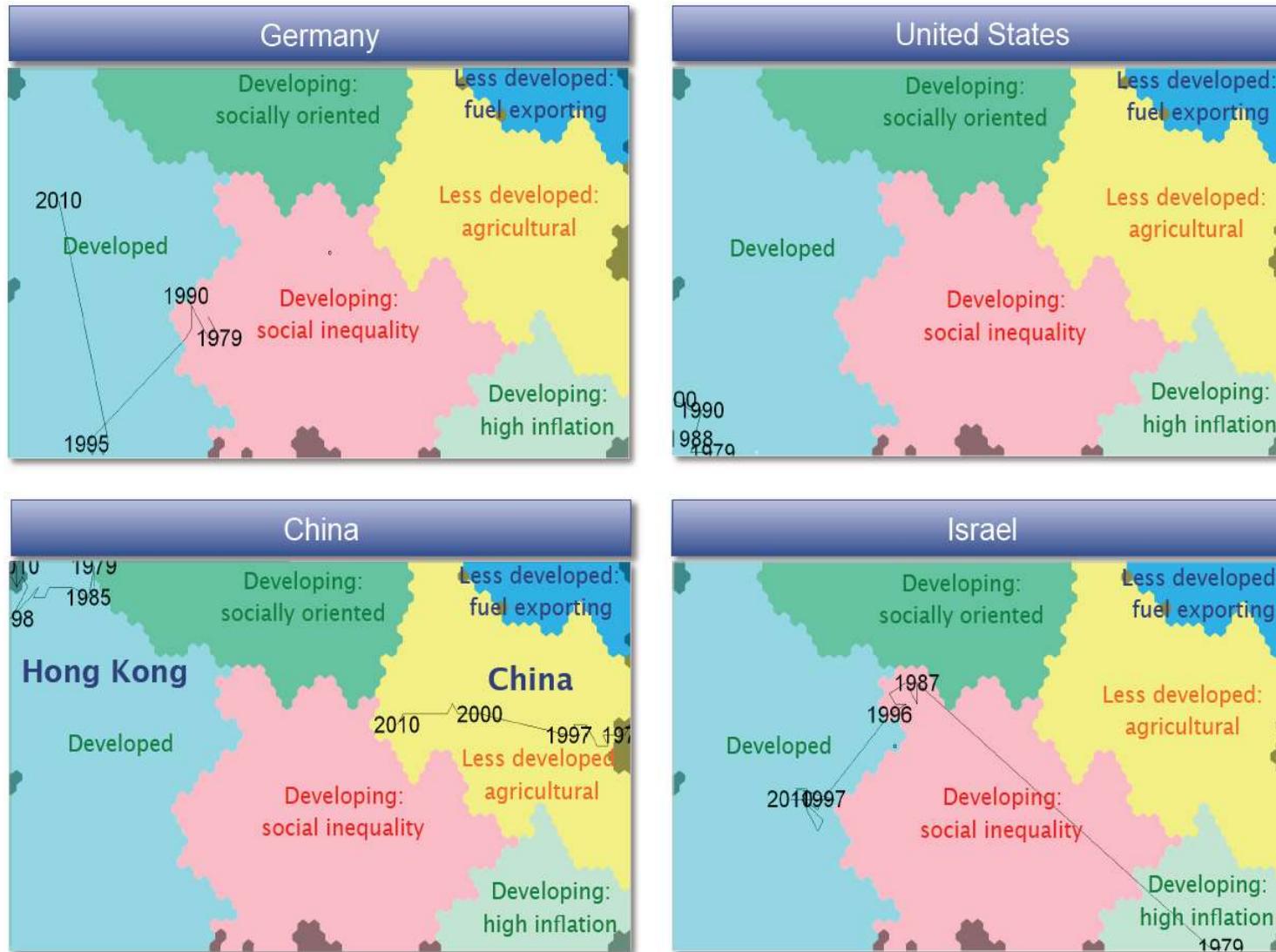
Representation of Countries by Quality of Life (II)



Segmentation of Economies (I)



Segmentation of Economies (II)



Segmentation of Ethic Investment Funds (I)

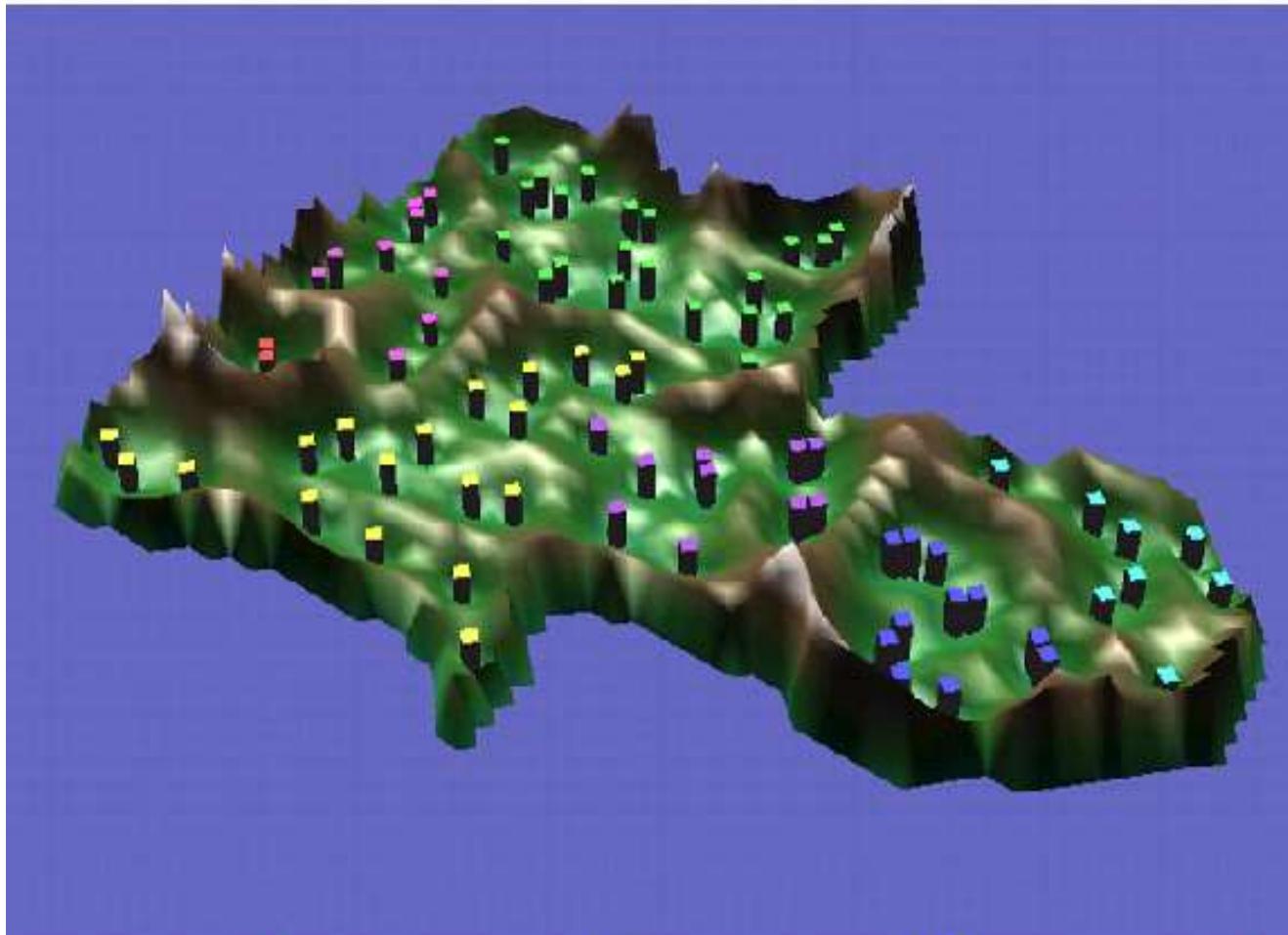
Procedure:

- (1) Choosing possible input variables**
- (2) Normalization of variables**
- (3) Correlation analysis**
- (4) Selecting the used input variables**

| Category | Description |
|-------------|--|
| Return | total return over considered time period |
| Return | return of the last 5 years |
| Return | return of the last 3 years |
| Risk | standard deviation of the last 3 years |
| Cost | offering premium |
| Volume | number of stocks in the fund |
| Transaction | frequency of shifting |
| Experience | retention period of managers over considered time period |

- (5) Training and Interpretation**

Segmentation of Ethic Investment Funds (II)



Cluster 1 (2 funds) ■ Cluster 2 (19 funds) □ Cluster 3 (11 funds) ■ Cluster 4 (10 funds) □
Cluster 5 (23 funds) ■ Cluster 6 (8 funds) ■ Cluster 7 (11 funds) ■

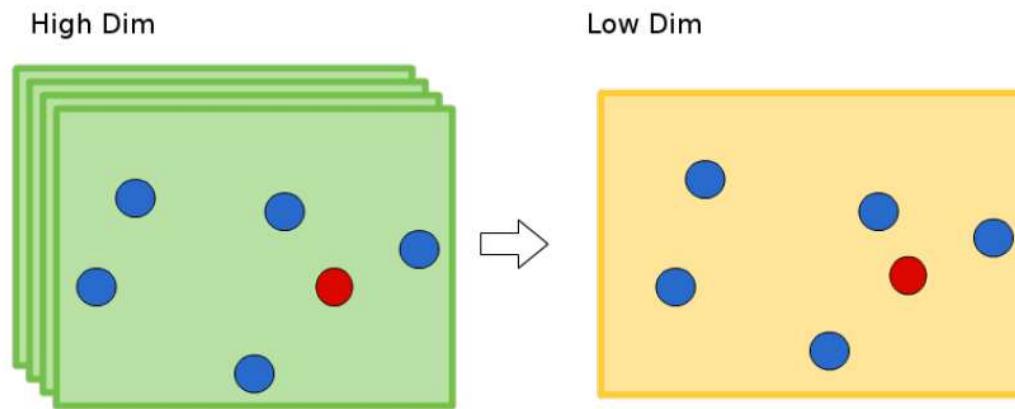
| Cluster | Interpretation |
|---------|----------------|
| 1 | Outliners |
| 2 | Big Family |
| 3 | Secure Players |
| 4 | Growth Players |
| 5 | Young & Loaded |
| 6 | Big & Popular |
| 7 | Blended Pearls |

t-distributed Stochastic Neighbor Embedding (t-SNE)

t-statistic Stochastic Neighbor Embedding

t-statistic Stochastic Neighbor Embedding (t-SNE) is a probabilistic approach to visualize the structure of complex data sets, preserving neighbor similarities. It converts high-dimensional distances between data points into probabilities that represent similarities.

It models each high-dimensional object by a two- or three-dimensional point in such a way that similar objects are modeled by nearby points and dissimilar objects are modeled by distant points.



These visualizations are useful to examine if there is any data-inherent structure that a clustering algorithm can exploit. It can also be used to estimate the number of clusters in the data.

Principle of t-SNE (I)

t-SNE converts high-dimensional Euclidean distances between data points into probabilities that represent similarities. The conditional probability $p_{j|i}$ that a data point x_i would pick x_j as its neighbor is given by

$$p_{j|i} = \frac{e^{-\frac{\|x_i - x_j\|^2}{2\sigma_i^2}}}{\sum_{k \neq i} e^{-\frac{\|x_i - x_k\|^2}{2\sigma_i^2}}}$$

k = number of the local neighbors
 σ_i = variance of the Gaussian distribution
 that is centered on datapoint x_i

This measures how close x_j is from x_i , considering a Gaussian distribution around x_i with a given variance σ_i^2 . This variance is different for every point.

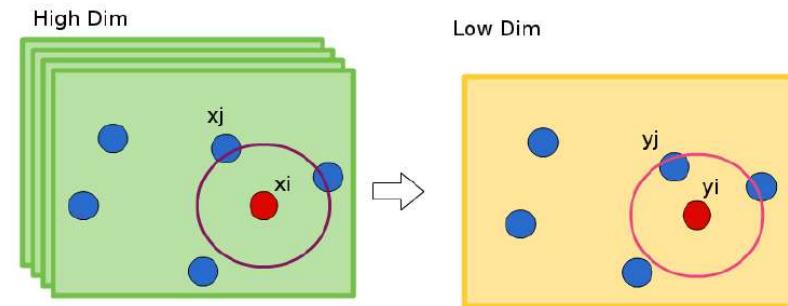
The joint probabilities p_{ij} that measure the pairwise similarity between objects x_i and x_j are calculated by symmetrizing the two conditional probabilities

$$p_{ij} = \frac{p_{j|i} + p_{i|j}}{2n}$$

Principle of t-SNE (II)

t-SNE searches for low-dimensional counterparts with the same similarities. The low-dimensional counterparts y_i and y_j of the high-dimensional data points x_i and x_j are modeled by similar probabilities

$$q_{ij} = \frac{\left(1 + \|y_i - y_j\|^2\right)^{-1}}{\sum_{k \neq i} \left(1 + \|y_k - y_i\|^2\right)^{-1}}$$

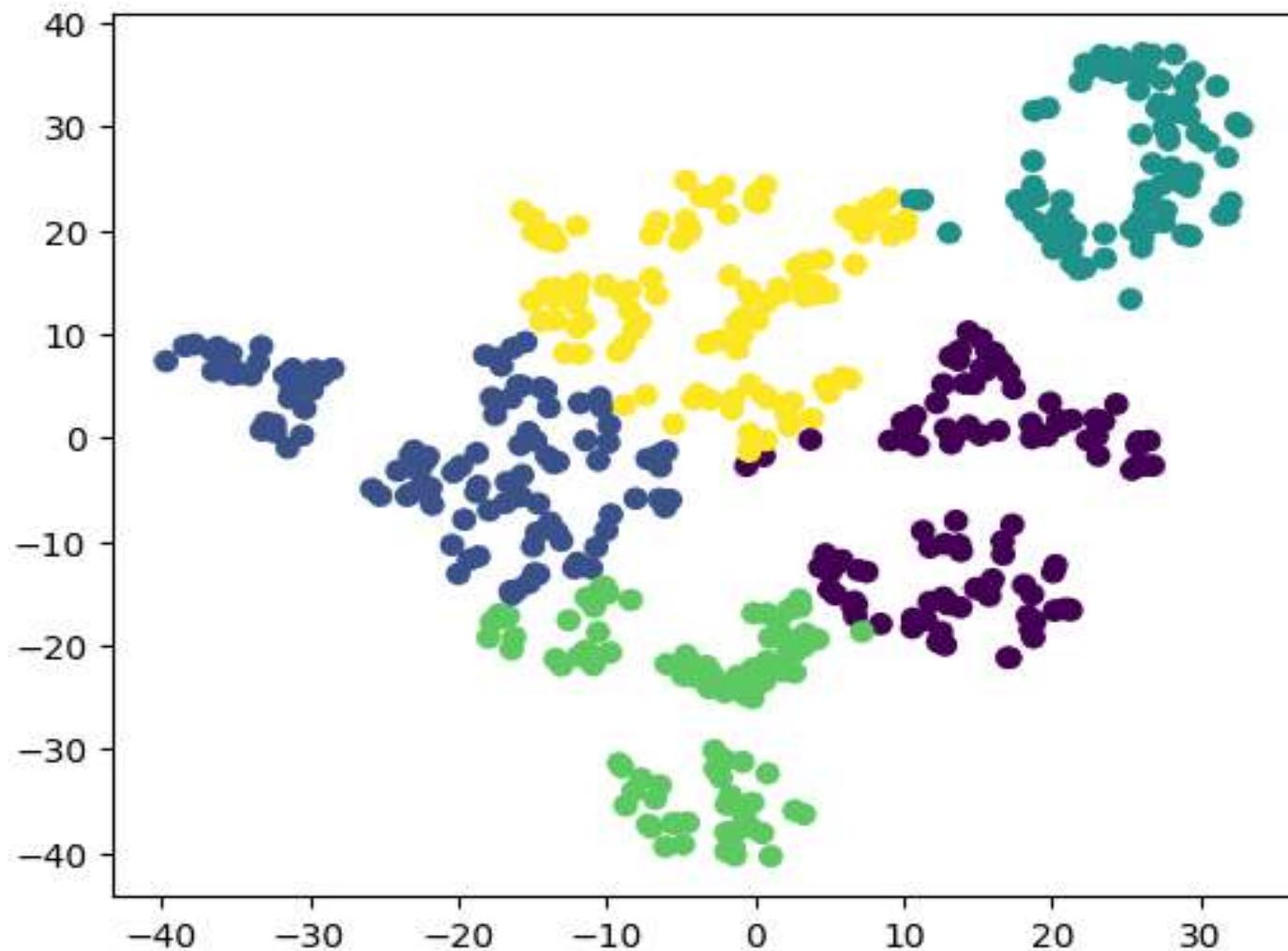


t-SNE aims to find a low-dimensional data representation which minimizes the mismatch between the joint probabilities p_{ij} and q_{ij} by minimizing the cost function

$$C = \sum_i \sum_j p_{ij} \log \frac{p_{ij}}{q_{ij}}$$

The method mainly preserves local similarity structure of the data. Large p_{ij} modeled by small q_{ij} result in a large penalty and small p_{ij} modeled by large q_{ij} result in a small penalty.

Application of t-SNE



Association Analysis

Market Basket Analysis

A classic application of market basket analysis addresses this question:

Which items are likely to be purchased together?

- If product A and product B often go together, then placing a more expensive alternative to B near the display for A can create an up-sell opportunity.
- If product A and B are often purchased together, putting them on sale at different times can drive purchases continually.

Source: SAS

Subject of Association Analysis

Association analysis identifies relationships or affinities between entities and/or between variables. These relationships are then expressed as a collection of association rules.

A typical application field is the market basket analysis. The aim is to detect purchase patterns that affect marketing decisions, e.g. frequently together or in sequence bought products.

The results of the analysis are formulated using so called association rules, e.g. "If item(set) A, then item(set) B". These rules describe the correlations between the entities.

The challenge is to find efficient algorithms which are able to discover the associations in huge data sets very fast, such as data from purchases of shops, product recommendations (like Amazon), music recommendations (like Last FM), medical diagnosis, or content optimization in magazine websites or blogs.

Basic Idea of the Association Analysis

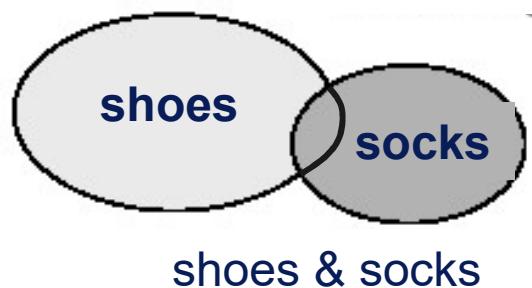
transaction chart

| | |
|-----------|--------------------|
| 1,000,000 | transactions total |
| 200,000 | shoes |
| 50,000 | socks |
| 20,000 | shoes & socks |

rule

If a customer has purchased shoes, then he also purchased socks in 10% of the cases.

venn diagram



evaluation criteria

Confidence: $20,000/200,000 = 10\%$
 Support: $20,000/1,000,000 = 2\%$

Source: Hettich; Hippner (2001): Assoziationsanalyse,
 in: Hippner et al. (Ed.): Handbuch Data Mining im Marketing, p. 460.

The Database

We can represent the items as an item set as

$$I = \{ i_1, i_2, \dots, i_n \}$$

Thus, a transaction is represented as

$$t_n = \{ i_j, i_k, \dots, i_n \}$$

We can represent a rule now as

$$\{i_1, i_2\} \rightarrow \{i_k\}$$

Which can be read as "if a user buys the item set on the left hand side, then the user will likely buy the item on the right hand side too".

Example: $\{\text{coffee}, \text{sugar}\} \rightarrow \{\text{milk}\}$

If a customer buys coffee and sugar, then they are also likely to buy milk.

Confidence and Support (I)

The strength of association is measured by support and confidence.

Support represents the frequency of the item-set in the database:

$$Support(A) = \frac{\text{transactions containing item } A}{\text{all transactions}}$$

$$Support(A \rightarrow B) = \frac{\text{transactions containing items } A \text{ and } B}{\text{all transactions}}$$

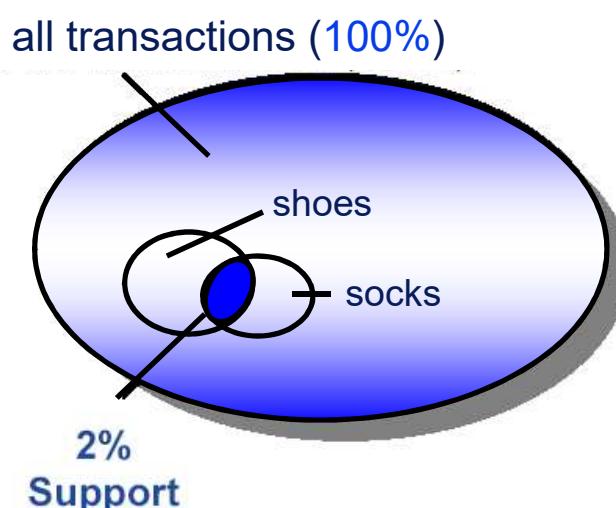
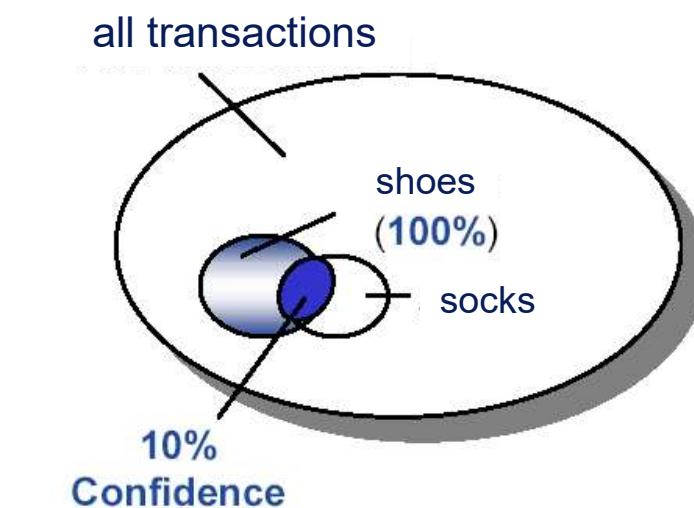
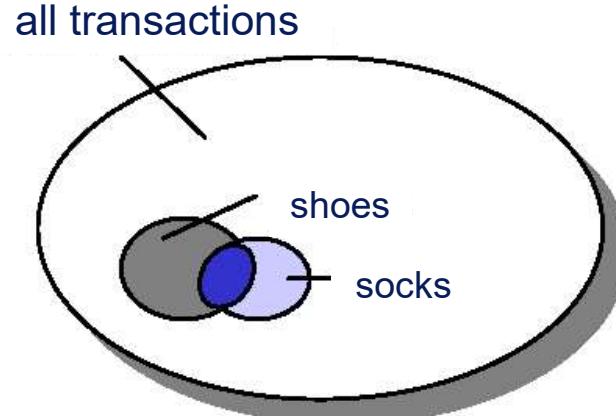
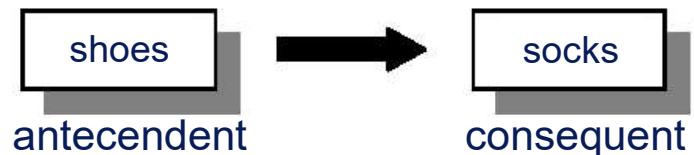
Confidence is an indication of how often the rule has been found to be true:

$$Confidence(A \rightarrow B) = \frac{\text{transactions containing every item in } A \text{ and } B}{\text{transactions containing the items in } A}$$

$$= \frac{Support(A \rightarrow B)}{Support(A)}$$

Confidence and Support (II)

In 10% of the cases where shoes were purchased, socks were also purchased.
Both products occur in 2% of all transactions.



Source: Hettich; Hippner (2001): Assoziationsanalyse, in: Hippner et al. (Ed.): Handbuch Data Mining im Marketing, p. 460.

Example Calculation of Support and Confidence

Database:

```
t1 : bread, honey, cheese
t2 : cheese
t3 : bread, butter, cheese, jam
t4 : bread, sausage
t5 : bread, butter
t6 : bread, sausage
t7 : bread, cheese
t8 : bread, butter, cheese
```

$$\text{sup}(\{\text{bread}\}) = 7/8 = 0.875$$

$$\text{sup}(\{\text{bread, cheese}\}) = 4/8 = 0.5$$

$$\text{sup}(\{\text{bread}\} \rightarrow \{\text{cheese}\}) = 4/8 = 0.5$$

$$\text{sup}(\{\text{bread, cheese}\} \rightarrow \{\text{butter}\}) = 2/8 = 0.25$$

$$\text{conf}(\{\text{bread}\} \rightarrow \{\text{cheese}\}) = 4/7 = 0.57$$

$$\text{conf}(\{\text{cheese}\} \rightarrow \{\text{bread}\}) = 4/5 = 0.8$$

$$\text{conf}(\{\text{bread, cheese}\} \rightarrow \{\text{butter}\}) = 2/4 = 0.5$$

Support is expressed as a percentage of the total number of records in the database.

Confidence is the ratio of the number of transactions that include all items in the consequent, as well as the antecedent (support) to the number of transactions that include all items in the antecedent.

Mining for Association Rules

When mining for association rules, especially those item combinations are of interest, which arise frequently.

For this reason it is reasonable to indicate a minimum value for the support (supmin) a rule must have and likewise a lower barrier for the Confidence (confmin) of a rule. With this, the number of generated rules and the effort for interpreting the results is reduced.

The algorithm for the search of association rules can now be formulated this way:

For a given set D of transactions, a minimum support supmin and a minimum confidence confmin. Find all association rules $A \rightarrow B$ with $\text{sup}(A \rightarrow B) \geq \text{supmin}$ and $\text{conf}(A \rightarrow B) \geq \text{confmin}$.

Source: Hettich; Hippner (2001): Assoziationsanalyse,
in: Hippner et al. (Ed.): Handbuch Data Mining im Marketing, p. 461.

Basic Idea of the Apriori Algorithmus

The Apriori algorithm is the original association rule algorithm. The goal is to find all frequent itemsets even in huge data sets very fast.

The Apriori algorithm is divided in two phases:

1. Within the first phase all frequent itemsets and their frequency are identified. This means to find all itemsets with a support value greater than the minimum support.
2. Within the second phase from the found itemsets all rules are generated, whose confidence value is greater than the minimum confidence.

Procedure of Phase 1

- 1. Find all frequent 1-itemsets ($k=1$) with a support value greater than the minimum support.**
- 2. Check all $(k+1)$ -itemsets containing the $(k-1)$ -itemsets if their support value greater than the minimum support.**

The Apriori algorithm takes advantage of the simple observation that all subsets of a frequent itemset must also be frequent. This observation allows the algorithm to consider a significantly reduced search space by starting with frequent individual items (eliminating rare items). Only those items can have frequent supersets. This idea is applied with every increase of k .

-
-
- 3. Stop if there is no further frequent $(k+1)$ -itemset. In the other case, increase k by 1 and repeat step 2.**

Example (I)

Database:

```
t1 : bread, honey, cheese
t2 : cheese
t3 : bread, butter, cheese, jam
t4 : bread, sausage
t5 : bread, butter
t6 : bread, sausage
t7 : bread, cheese
t8 : bread, butter, cheese
```

Parameter:
Minimum support: 25%,
that means 2/8 transactions

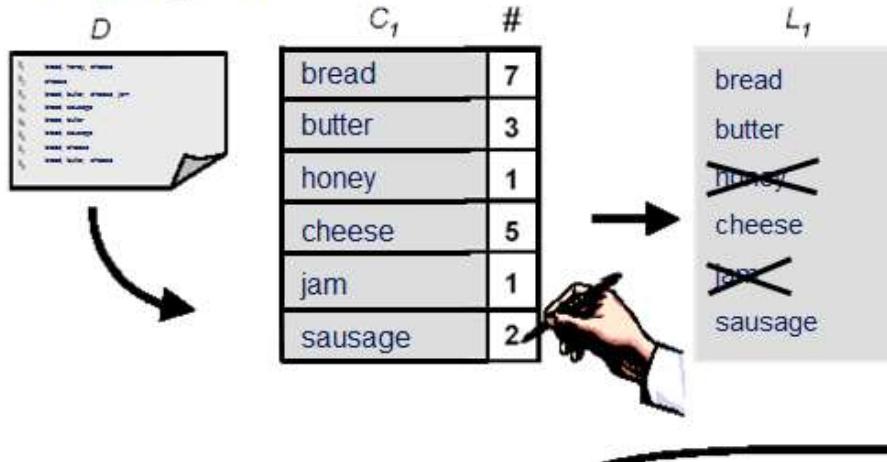
Number of transactions: 8

Source: Hettich; Hippner (2001): Assoziationsanalyse,
in: Hippner et al. (Ed.): Handbuch Data Mining im Marketing, p. 465.

Example (II)

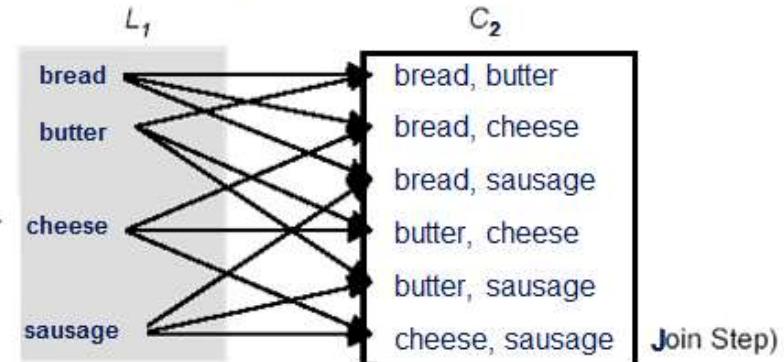
Apriori Algorithm:

1. Counting 1-itemsets



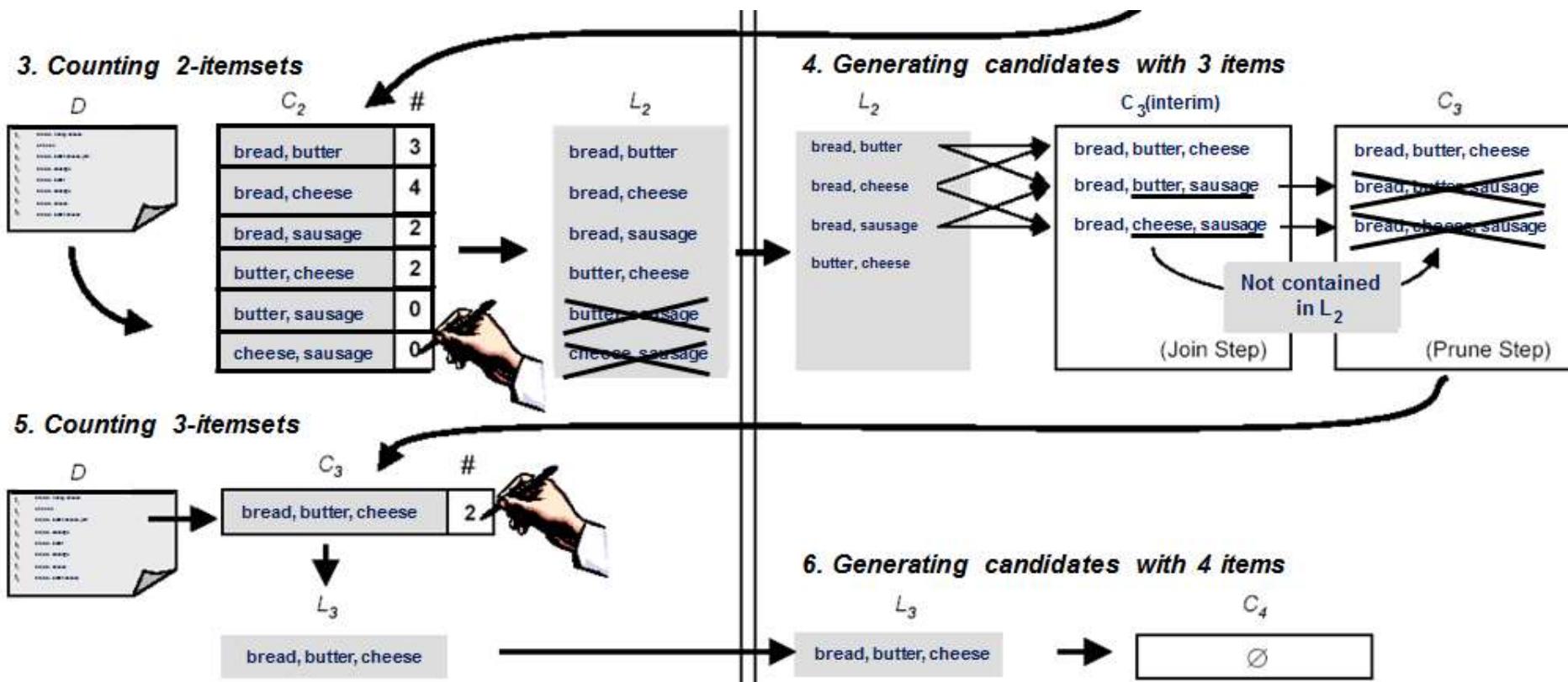
Apriori-gen Function:

2. Generating candidates with 2 items



Source: Hettich; Hippner (2001): Assoziationsanalyse,
 in: Hippner et al. (Ed.): Handbuch Data Mining im Marketing, p. 465.

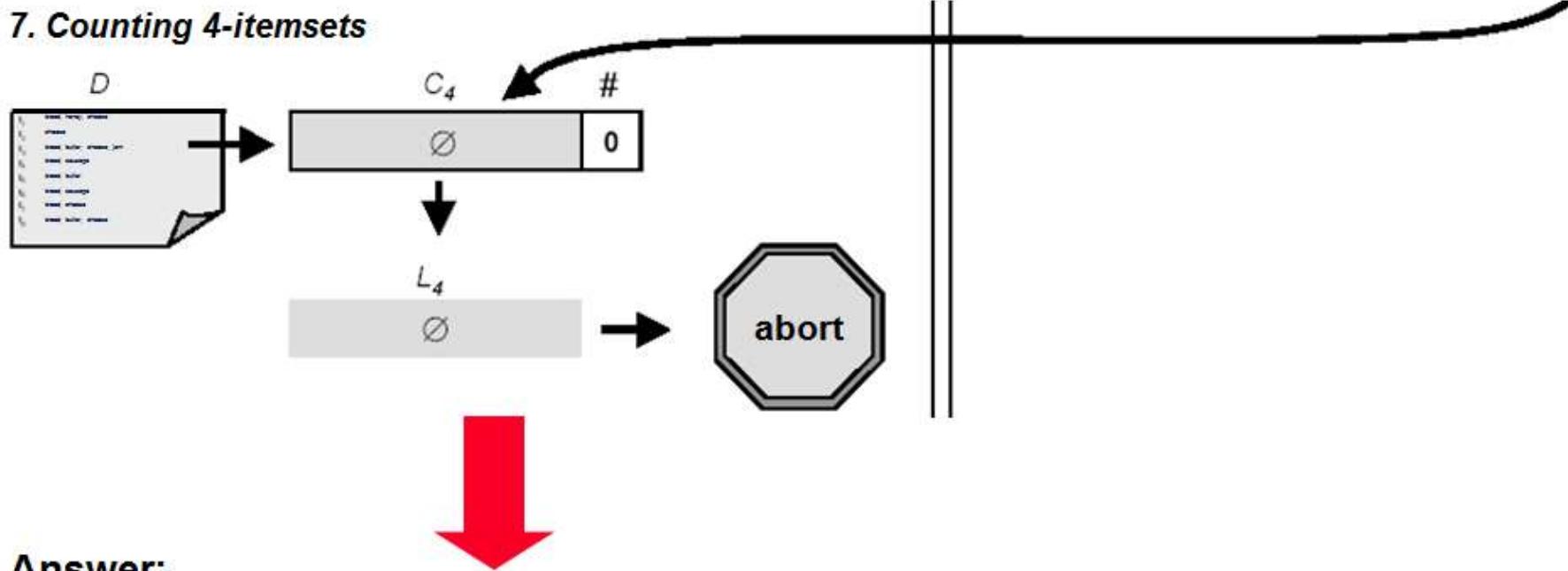
Example (III)



Source: Hettich; Hippner (2001): Assoziationsanalyse,
in: Hippner et al. (Ed.): Handbuch Data Mining im Marketing, p. 465.

Example (IV)

7. Counting 4-itemsets



Answer:

| | L_1 | L_2 | L_3 |
|---------|-------|----------------|-------|
| bread | 7 | bread, butter | 3 |
| butter | 3 | bread, cheese | 4 |
| cheese | 5 | bread, sausage | 2 |
| sausage | 2 | butter, cheese | 2 |

Source: Hettich; Hippner (2001): Assoziationsanalyse,
 in: Hippner et al. (Ed.): Handbuch Data Mining im Marketing, p. 465.

Procedure of Phase 2

After the identification of the frequent itemsets, the association rules are generated in the second phase.

Only rules having a confidence value greater than the minimum confidence are accepted.

Given a frequent itemset $\{a, b, c\}$, we then need to compute confidences of rules such as $\{a, b\} \rightarrow c$, $\{a, c\} \rightarrow b$, and $c \rightarrow \{a, b\}$, and then compare the confidences against the minimum confidence. This has to be done with all frequent itemsets.

Example (V)

Procedure of Rules Generation

Parameter:
Minimum confidence: 60%

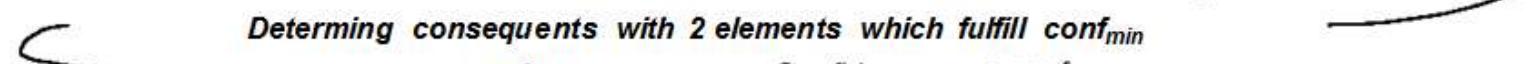
1. Generating rules from L_2 ($k=2$)

| | rules | Confidence | $>conf_{min}$ | 2 item rules | conf |
|-------|--|--------------------------|---------------|-----------------|------|
| L_2 | butter → <u>bread</u>
bread → <u>butter</u> | 3/3 = 1.00
3/7 = 0.43 | ✓ | butter → bread | 1.00 |
| | cheese → <u>bread</u>
bread → <u>cheese</u> | 4/5 = 0.80
4/7 = 0.57 | ✓ | cheese → bread | 0.80 |
| | sausage → <u>bread</u>
bread → <u>sausage</u> | 2/2 = 1.00
2/7 = 0.29 | ✓ | sausage → bread | 1.00 |
| | cheese → <u>butter</u>
butter → <u>cheese</u> | 2/5 = 0.40
2/3 = 0.67 | ✓ | butter → cheese | 0.67 |



2. Generating rules from L_3 ($k=3$)

| | rules | Confidence | $>conf_{min}$ | 3 item rules | conf |
|-------|--|--|---------------|--|--------------|
| L_3 | bread, butter, cheese → <u>bread</u>
bread, cheese → <u>butter</u>
bread, butter → <u>cheese</u> | 2/2 = 1.00
2/4 = 0.50
2/3 = 0.67 | ✓ | butter, cheese → bread
bread, butter → cheese | 1.00
0.67 |

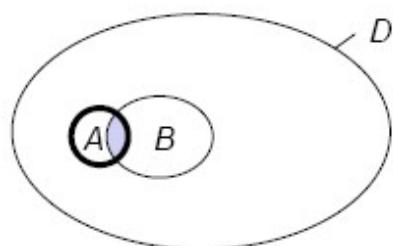
Determining consequents with 2 elements which fulfill $conf_{min}$

| rules | Confidence | $>conf_{min}$ |
|-------------------------------|------------|---------------|
| butter → <u>bread, cheese</u> | 2/3 = 0.67 | ✓ |

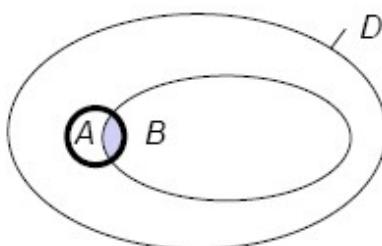
Source: Hettich; Hippner (2001): Assoziationsanalyse,
in: Hippner et al. (Ed.): Handbuch Data Mining im Marketing, p. 469.
Introduction to Data Analytics in Business

Problems with Confidence

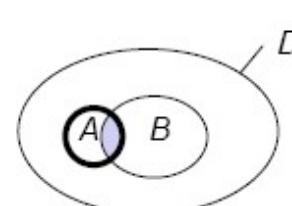
The confidence of a rule could additionally remain unchanged in some important cases from a marketing point of view:



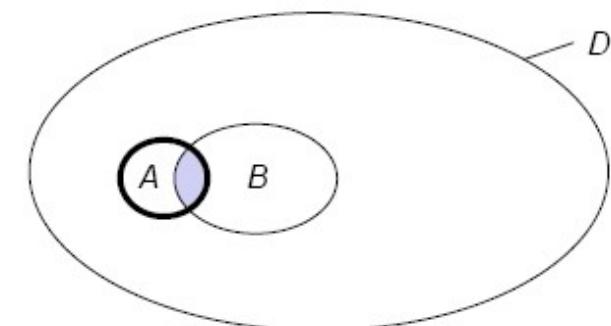
Origin



Frequency of
B rises



Frequency of
D declines



All frequencies are
rising proportional

Source: Hettich; Hippner (2001): Assoziationsanalyse,
in: Hippner et al. (Ed.): Handbuch Data Mining im Marketing, p. 478.

The lift indicates the number of times of the observed frequency greater than the joint frequency of two independent item sets:

$$\text{lift}(A \rightarrow B) = \frac{\text{Confidence}(A \rightarrow B)}{\text{Support}(B)} = \frac{\text{Support}(A \rightarrow B)}{\text{Support}(A) \cdot \text{Support}(B)}$$

A rule is interesting if its lift is significantly greater than 1.

If some rule had a lift of 1, it would imply that the probability of occurrence of the antecedent and that of the consequent are independent of each other. When two events are independent of each other, no rule can be drawn involving those two events.

If the lift is > 1 , that lets us know the degree to which those two occurrences are dependent on one another, and makes those rules potentially useful for predicting the consequent in future data sets.

Problems evaluating the Rules

A disadvantage of the association analysis is that it often produces an amount of rules which is too time-consuming to interpret.

Helpful measures can be:

- Sort the rules according to their quality criteria (e.g. support, confidence or lift).
- Use graphical presentation to achieve an overview.
- Apply filters, e.g. the exclusion of some items from the consequent, to reduce the amount of generated rules.

Graphical Display of Association Rules

