

Machine Learning (ML) Concepts & Algorithms

Status: 1 December 2020

DHBW – Fakultät Technik-Informatik, Stuttgart, Autumn 2020

Dr. Hermann Völlinger, Mathematics & IT Architecture

<http://www.dhbw-stuttgart.de/~hvoellin/>

www.dhbw-stuttgart.de

Lecture Information in Moodle Account of Course 18c:

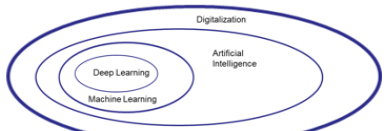
<https://elearning.dhbw-stuttgart.de/moodle/course/view.php?id=8210>

ML0 – General Remarks and Goals of Lecture (ML)

Dr. Hermann Völlinger
Privatier, ex-IBM-SWG Services
Mathematiker, IT-Architekt
Data Warehousing (DWH), Business Intelligence (BI) &
Data Mining (DM), Machine Learning (ML)



- Tel.: +49(0)-7159-42246
- Mobile: +49(0)-177-4117949
- Email: hermann.vollinger@gmail.com
- Dr. Hermann Völlinger's [Biography](#)
- [Content of the Lecture Data Warehouse I \(WS 2001\)](#)
- [Results of the examination of Data Warehouse I lecture](#)
- [Content & Literature List of the Lecture Data Warehouse II \(SS 2002\)](#)
- [Content of the Lecture Data Warehouse & Data Mining \(WS2003 - WS2015\)](#)
- [Script of the Data Warehouse & Data Mining Lecture, Sample "Grocery", Access DB & Flat File](#)
- [Exercises & Solutions of the Lecture Data Warehouse & Data Mining](#)
- [Script of the Lecture "Introduction to Data Warehousing" \(WS2019\)](#)
- [Exercises & Solutions of the Lecture "Introduction to Data Warehousing"](#)
- [List of Themes for Seminar-Papers of the Lecture "Introduction to Data Warehousing"](#)
- [MindMap of the Lecture "Machine Learning \(ML\) Concepts & Algorithms" \(WS2020\)](#)
- [Script of the Lecture "Machine Learning \(ML\) Concepts & Algorithms" \(WS2020\)](#)
- [Exercises & Solutions of the Lecture "Machine Learning \(ML\) Concepts & Algorithms"](#)
- [Video of Monitoring of the Lecture "Machine Learning \(ML\) Concepts & Algorithms"](#)
- [Video of Solution/Homework 2.3 of the Lecture "Machine Learning \(ML\) Concepts & Algorithms"](#)
- [List of Themes for Seminar-Papers of the Lecture "Machine Learning \(ML\) Concepts & Algorithms"](#)
- [List of Themes for DHBW-Diploma Paper \(Data Warehouse / Business Intelligence\)](#)

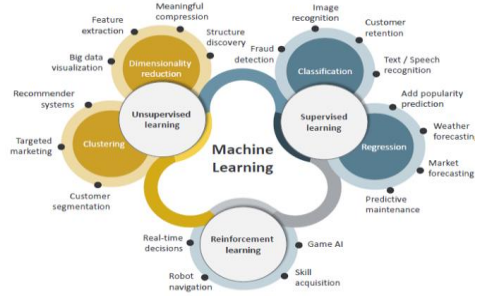


Homework / Exercises to Lecture "ML-Concepts & Algorithms"

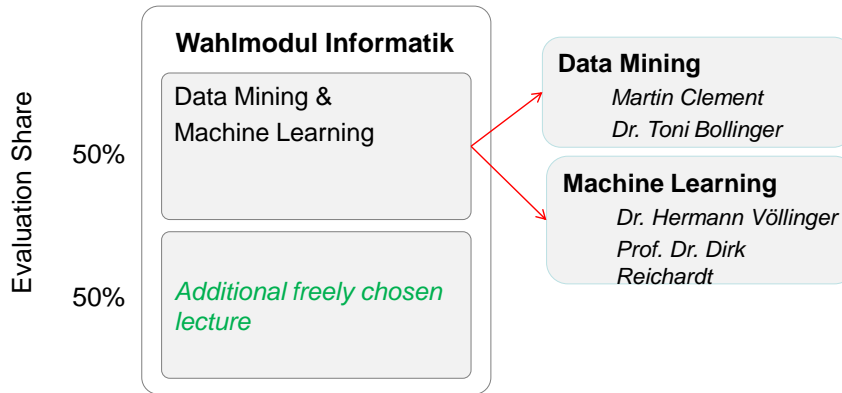
by Dr. Hermann Völlinger and Other

Goal: Documentation of all solutions to the homework/exercises in the lecture "ML-Concepts & Algorithms".

Chapter	Title of Chapter	Number of Homework	Incl. Advanced Homework
ML0	General Remarks and Goals of Lecture (ML)	1	0
ML1	Introduction to Machine Learning (ML)	3	0
ML2	Classical Learning: k-Nearest & Conf. Elm. Algos.	2	0
ML3	Supervised and Unsupervised Learning	5	0
ML4	Decision Tree Learning	3	3
ML5	Simple Linear Regression (SLR) & Multiple Linear Regression (MLR)	3	1
ML6	Neural Networks: Convolutional	6	2
ML7	Neural Networks: Backpropagation Algorithm	2	0
ML8	ML: Support Vector Machines	4	0
sum		33	6



General Remarks to Lecture Machine Learning (ML) (1/3)



“Wahlmodul Informatik II (T3INF4902)” (Elective Module Computer Science), see:

https://www.dhbw.de/fileadmin/user/public/SP/STG/Informatik/Angewandte_Informatik.pdf

Modul Description: „Data Mining und Grundlagen des Maschinellen Lernens“: Machine Learning

- Einführung in das Maschinelle Lernen
- Symbolische Lernverfahren
- Grundlagen Neuronaler Netze
- Probabilistische Lernmodelle
- Erweiterte Konzepte und Deep Learning
- Entwurf und Implementierung ausgewählter Techniken für eine Anwendung

Data Mining:

- Daten und Datenanalyse
- Clustering
- Classification
- Assoziationsanalyse
- Weitere Verfahren, z.B.:
 - Regression
 - Deviation Detection
- Visualisierung

General Remarks to Lecture Machine Learning (ML) (2/3)

- Time: online, Tuesday (6 x) starting 29.9. – 3.11.20, 10:00 – 12:30, i.e.: 105 minutes lecture, 30 minutes exercises / homework (everyone should present his homework at least one time), 10-15 minutes break, more details see my DHBW homepage).
- Examination: Seminar Paper: 2 persons, 8-10 pages/English/11.12.2020,18:00. Grading together with Data Mining will be seen on DHBW Bachelor Certificate.
- My DHBW Lecture Homepage: <http://www.dhbw-stuttgart.de/~hvoellin/> contains a script of the lecture and also solutions („Musterlösungen“) to the homeworks/exercises. The homepage includes also sample data for exercises and a lot of other interesting facts about Machine Learning.



Dr. Hermann Voellinger

Privatier, ex.IBM-SWG Services
Mathematiker, IT Architekt
Data Warehousing (DWH), Business Intelligence (BI) &
Data Mining (DM), Machine Learning (ML)

Tel.: +49(0)-7159-42246,
Mobile: +49(0)-177-4117949
Email: hermann.voellinger@gmail.com

Dr. Hermann Voellinger's [Biography](#)

[Content](#) of the Lecture Data Warehouse I (WS 2001)
[Results of the examination](#) of Data Warehouse I lecture
[Content & Literature List](#) of the Lecture Data Warehouse II (SS 2002)
[Concept](#) of the Lecture Data Warehouse & Data Mining (WS2002 - WS2015)
[Script](#) of the Data Warehouse & Data Mining Lecture, Sample "Grocery", [Access DB](#) & Flat File [FF](#)
[Exercises & Solutions](#) of the Lecture Data Warehouse & Data Mining

[Script](#) of the Lecture "Introduction to Data Warehousing" (WS2019)
[Exercises & Solutions](#) of the Lecture "Introduction to Data Warehousing"
[List of Themes for Seminar-Papers](#) of the Lecture "Introduction to Data Warehousing"

[MindMap](#) of the Lecture "Machine Learning (ML): Concepts & Algorithms" (WS2020)
[Script](#) of the Lecture "Machine Learning (ML): Concepts & Algorithms" (WS2020)
[Exercises & Solutions](#) of the Lecture "Machine Learning (ML): Concepts & Algorithms"
[Video of Motivation](#) of the Lecture "Machine Learning (ML): Concepts & Algorithms"
[Video of Solution/Homework 3.2](#) of the Lecture "Machine Learning (ML): Concepts & Algorithms"
[List of Themes for Seminar-Papers](#) of the Lecture "Machine Learning (ML): Concepts & Algorithms"

List of [Themes for DHBW Diploma Paper](#) (Data Warehouse / Business Intelligence)



General Remarks to Lecture Machine Learning (ML) (3/3)

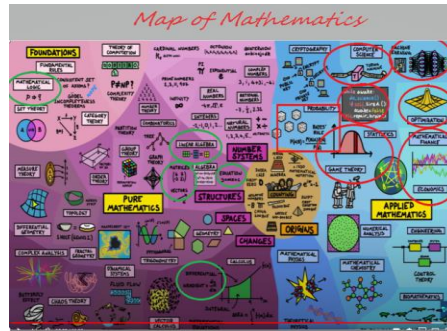
- The solutions of homework ("Musterlösungen") are published in the Homepage. More than 30 homeworks are included. If the student agrees, his solutions can also be included in the script of solutions.
- The lecture script is in **English**, since the common IT language is English. Some dedicated slides/pages are in German, not to lose "Look and Feel" of the slide.
- Prerequisites: Python or R Programming (i.e. usage of Jupyter Notebooks), Lecture: "Datenbanken1", Lecture: "Mathematik 1" and "Mathematik 2"; some basis knowledge about Data Warehousing (fifth semester) and Data Mining (visited in parallel).
- In WS2020 lecture we have only 18 SW (lecture hours), so we will skip some chapters in this script, which is designed for 36 SW (see next slide). But feel free to have also a look on this chapters.

Homework / Exercises to Lecture "ML- Concepts & Algorithms"

Dr. Hermann Völlinger and Other

Goal: Documentation of all Solutions to the Homework/Exercises in the Lecture "ML- Concepts & Algorithms".

Chapter	Title of Chapter	Number of Homeworks	Incl. Advanced Homeworks
ML0	General Remarks and Goals of Lecture (ML)	1	0
ML1	Introduction to Machine Learning (ML)	5	0
ML2	Concept Learning: V-Spaces & Conf. Estm. Algo.	2	0
ML3	Supervised and Unsupervised Learning	5	2
ML4	Decision Tree Learning	5	3
ML5	Simple Linear Regression (SLR) & Multiple Linear Regression (MLR)	5	2
ML6	Neural Networks: Convolutional Neural Networks	4	2
ML7	Backpropagation Algorithm	2	0
ML8	ML: Support Vector Machines	4	0
sum		33	9



Mathematik2 / Lehr- und Lerneinheiten:

Angewandte Mathematik

- Grundlagen der Differential und Integralrechnung reeller Funktionen mit mehreren Veränderlichen sowie von Differentialgleichungen und Differentialgleichungssystemen
- Numerische Methoden und weitere Beispiele mathematischer Anwendungen in der Informatik

Statistik

- Deskriptive Statistik
- Zufallsexperimente, Wahrscheinlichkeiten und Spezielle Verteilungen – Induktive Statistik
- Anwendungen in der Informatik

Data Warehouse & Data Mining:

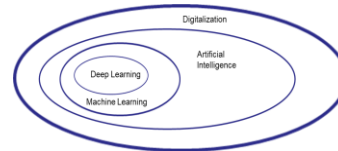
Data Mining / Grundlagen

- Daten und Datenanalyse
- Clustering
- Classification
- Assoziationsanalyse
- Weitere Verfahren,
 - z.B.: Regression, Deviation Detection
- Visualisierung

Goals of Lecture (1/3)

The lecture's aim is to introduce Machine Learning (ML) as part of Artificial Intelligence. We learn the most important methods that are used in Machine Learning (ML) and they are presented with their essential features. Several references are given to in-depth applications or information through internet-links or further literature. In many places concrete implementation examples with *Jupyter Notebooks** are defined. Especially we see the following "List of Topics":

- Introduction of ML (with Categorization of ML) and Ethical aspects of AI and ML.
- Concept Learning: Version Spaces & Candidate Elimination Algorithm.
- Classification with Bayes Learning and K-Means Clustering (i.e. for Iris dataset).
- Build Decision Trees with GINI-Index method or with ID3 Algorithm (incl. Python Jupyter Notebook).
- Decision Trees - Usage for Predictive Maintenance in Production (Use Case: „Gießerei“).



* The name Jupyter is a reference to the three core programming languages supported by Jupyter, which are [Julia](#), [Python](#) and [R](#).

Machine Learning, Deep Learning, Cognitive Computing - Artificial Intelligence technologies are spreading rapidly. The background is that today the computing and storage capacities are available that make AI scenarios possible.

What is Artificial Intelligence?

The term **Artificial Intelligence (AI)** was coined more than 60 years ago by the US computer scientist John McCarthy. He submitted an application for a research project on machines that played chess, solved mathematical problems and learned on their own. In the summer of 1956, he presented his findings to other scientists. The British mathematician Alan Turing had already developed the "Turing Test" six years earlier, which can determine whether the other person is a human or a machine posing as a human being.

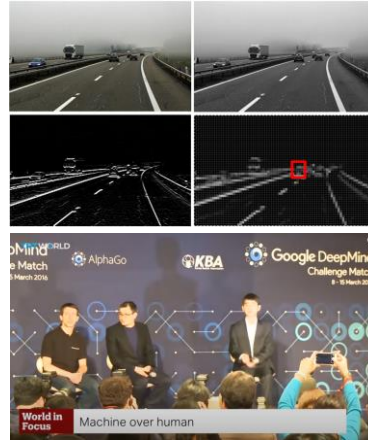
But it took decades to advance AI research because it required significantly more computational power. In the mid-1990s, the time had come and scientists devoted themselves to tasks such as image recognition. Gradually, artificial neural networks have been developed that capture information such as language and images, recognize patterns and develop their own solutions.

Artificial intelligence is now a part of everyday life, whether as search engines, as smart language assistants, in medical diagnoses or in self-driving vehicles. Military also discusses to use AI for military issues, like war robots. The United Nations (UN) debated how to handle killer robots that autonomously decide on military operations.

Goals of Lecture (2/3)

Continue the "List of Topics":

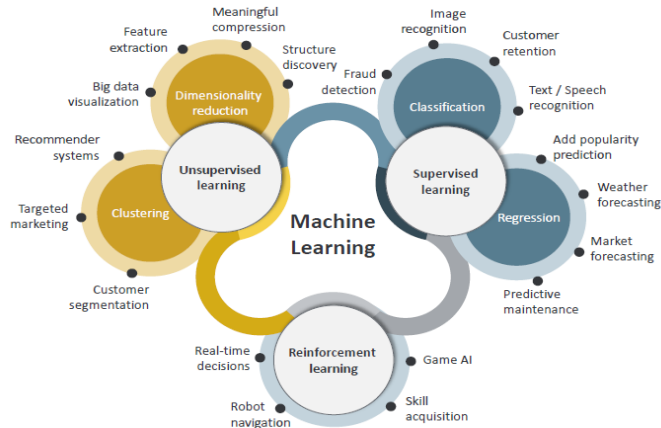
- Applications of Regression Methods: simple Linear- (sLR) & multiple Linear Regression (mLR).
- Convolutional Neural Networks Concepts and Algorithms (i.e. for image recognition) and several other Use Cases.
- Deep Learning in Google AlphaGo (with Self Learning AI) & Autonomous Driving.
- Back Propagation for Neural Networks (incl. Definition and Implementation with Jupyter Notebook).
- Support Vector Machines (SVM) and detailed concrete examples with Jupyter Notebook.



See Ref. [HVö-3]: [http://www.lehre.dhbw-stuttgart.de/~hvoellin/MindMap_Machine_Learning_\(ML\)_Lecture.pdf](http://www.lehre.dhbw-stuttgart.de/~hvoellin/MindMap_Machine_Learning_(ML)_Lecture.pdf)

Goals of Lecture (2/3) – Understand 3 Categories of ML

See Ref. [HHeid]: <https://towardsdatascience.com/what-are-the-types-of-machine-learning-e2b9e5d1756f>



Supervised Learning

Supervised learning is the most popular paradigm for machine learning. It is the easiest to understand and the simplest to implement. It is very similar to teaching a child with the use of flash cards.

Given data in the form of examples with labels, we can feed a learning algorithm these example-label pairs one by one, allowing the algorithm to predict the label for each example, and giving it feedback as to whether it predicted the right answer or not.

Unsupervised Learning

Unsupervised learning is very much the opposite of supervised learning. It features no labels. Instead, our algorithm would be fed a lot of data and given the tools to understand the properties of the data. From there, it can learn to group, cluster, and/or organize the data in a way such that a human (or other intelligent algorithm) can come in and make sense of the newly organized data.

Reinforcement Learning

Reinforcement learning is learning from mistakes. Place a reinforcement learning algorithm into any environment and it will make a lot of mistakes in the beginning. So long as we provide some sort of signal to the algorithm that associates good behaviors with a positive signal and bad behaviors with a negative one, we can reinforce our algorithm to prefer good behaviors over bad ones. Over time, our learning algorithm learns to make less mistakes than it used to.

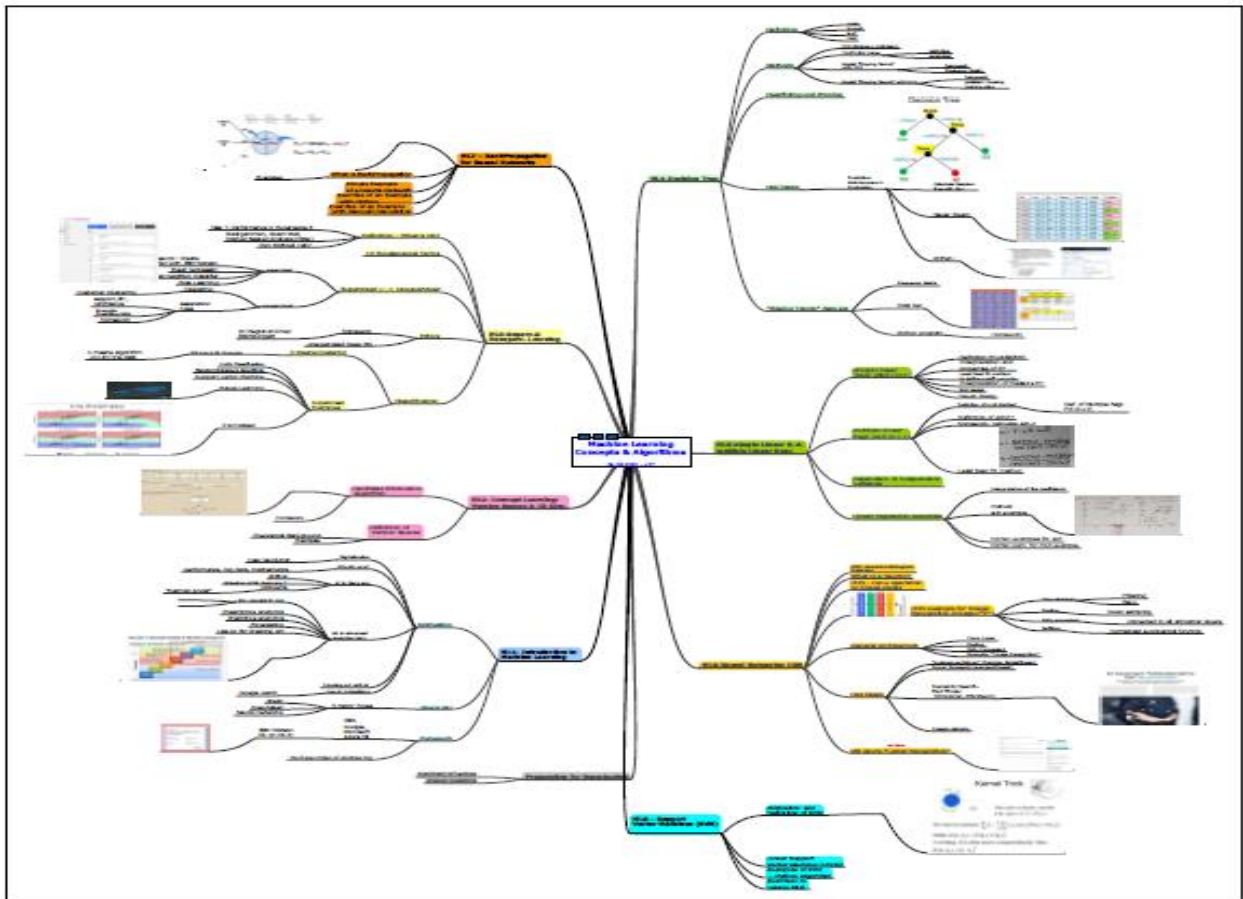
Overview of the Lessons in Machine Learning (ML)

We have 8 chapters, 3 are skipped (see gray color font). The date can change if necessary:

1. ML1: Introduction to Machine Learning - 29.09.2020, 10:00-12:30
2. ML2: Concept Learning: Version Spaces & Candidate Elim. Algorithm
3. ML3: Supervised and Unsupervised Learning - 06.&13.10.20, 10:00-12:30
4. ML4: Decision Tree Learning - 20.10.2020, 10:00-12:30
5. ML5: Simple Linear - & Multiple Regression - 27.10.2020, 10:00-12:30
6. ML6: Neural Networks: Convolutional NN - 03.11.2020, 10:00-12:30
7. ML7: Neural Network: BackPropagation Algorithm
8. ML8: Support Vector Machines (SVM)

Actual time plan:

<https://rapla.dhbw-stuttgart.de/rapla?key=txB1FOi5xd1wUJBWuX8lJoG0cr9RVifzB7e5WYTczgq3qJZTian3jkGZb9hTVbzP>



References to Machine Learning (1/3)

1. **[EAlp]** Ethem Alpaydin, "Maschinelles Lernen", De Gruyter (Oldenbourg), 2. Edition (May 2019)
2. **[Bei+Kern]** Christoph Beierle, Gabriele Kern-Isberner, "Methoden Wissensbasierter Systeme - Grundlagen - Algorithmen - Anwendungen", Springer-Vieweg Verlag, 6. Edition (August 2019)
3. **[BERT]**: Jacob Devlin and Other: "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding"; Google(USA); 2019
4. **[CWJ]**: COMPUTERWOCHE – Digitalisierung/ Machine Learning „Teil 1: Die Grundlagen - darum geht es“, „Teil 2: Die Pläne der Anwender - das haben deutsche Unternehmen vor“, Teil 3: Anwendungen und Plattformen - die Technik“: <https://www.computerwoche.de/a/machine-learning-darum-geht-s.3330413>
5. **[FAZ]**: FAZ Artikel „TECHNIK DER ZUKUNFT: Die Hälfte der Deutschen weiß nicht, was KI ist“: http://www.xing-news.co/reader/news/articles/1857263?newsletter_id=39283&toolbar=true&xng_share_origin=web
6. **[GI-RgStg]**: Gesellschaft für Informatik-Arbeitskreis Künstliche Intelligenz, Regionalgruppe Stuttgart: <http://rg-stuttgart.gi.de/veranstaltungen/mo-14012019-themenabend-kuenstliche-intelligenz.html>
7. **[HHeid]**: Hunter Heidenreich: "What are the Types of Machine Learning"; Internet- Blog: <https://towardsdatascience.com/what-are-the-types-of-machine-learning-e2b9e5d1756f>

References to Machine Learning (2/3)

8. **[HVö-1]:** Hermann Völlinger: [Script](#) of the Lecture "Introduction to Data Warehousing"; DHBW Stuttgart; WS2019
9. **[HVö-2]:** Hermann Völlinger and Other: [Exercises & Solutions](#) of the Lecture "Introduction to Data Warehousing"; DHBW Stuttgart; WS2019
10. **[HVö-3]:** Hermann Völlinger: [MindMap](#) of the Lecture "Machine Learning: Concepts & Algorithms"; DHBW Stuttgart; WS2020
11. **[HVö-4]:** Hermann Völlinger and Other: [Exercises & Solutions](#) of the Lecture "Machine Learning: Concepts & Algorithms"; DHBW Stuttgart; WS2020
12. **[HVö-5]:** Hermann Völlinger: [Script](#) of the Lecture "Machine Learning: Concepts & Algorithms"; DHBW Stuttgart; WS2020
13. **[HVö-6]:** Hermann Völlinger: GitHub to the Lecture "Machine Learning: Concepts & Algorithms"; see in: <https://github.com/HVoellinger/Lecture-Notes-to-ML-WS2020>
14. **[MatLab]:** MatLab eBook: "Reinforcement Learning with MATLAB - Understanding Training and Deployment"; MathWorks 2019;
<https://www.slideshare.net/HiteshMohapatra/reinforcement-learning-ebook-part3>
15. **[SfUni-1]:** Stanford University (USA) - [Machine Learning](#) Course, by Andrey Ng:
<https://www.coursera.org/learn/machine-learning>

References to Machine Learning (3/3)

16. **[SfUni-2]:** Stanford University (USA) – Series of 112 YouTube Videos about Machine Learning, by Andrew Ng (see above):
https://www.youtube.com/playlist?list=PLLssT5z_DsK-h9vYZkQkYNWcltqhiRJLN
17. **[Sift]:** Sift Science Engineering – Article: “Deep Learning for Fraud Detection - Introduction”:
<https://engineering.siftscience.com/deep-learning-fraud-detection/>
18. **[TDWI]:** TDWI eBook: “PROJEKTE UND ARTIFICIAL-INTELLIGENCE WIRTSCHAFTLICHKEIT”; SIGS DATACOM GmbH, 53842 Troisdorf, info@sigs-datacom.de; 2020
19. **[TMitch]:** Tom Mitchell „Machine Learning”, McGraw Hill, 1997:
<https://www.cs.cmu.edu/~tom/mlbook.html>
20. **[TMun]:** Toshinori Munakata: "Fundamentals of the new Artificial Intelligence", Springer Verlag, 2. Edition (January 2011)
21. **[TU-Darm]:** TU Darmstadt: Data Mining und Maschinelles Lernen - WS 17/18 „Einführung in maschinelles Lernen und Data Mining“: <http://www.ke.tu-darmstadt.de/lehre/ws-17-18/mldm>

Beside the information you get from the literature, you have also the chance to learn from other ML experts in your town. For example you can visit a **Meetup** meeting (**i.e** under the logo “**Cyber Valley**”) in Stuttgart/Tübingen.

Exercises to Lesson ML0

Homework H0.1 – “Three Categories of Machine Learning”

Groupwork (2 Persons). Compare the differences of the three categories, see slide “goal of lecture (2/2)”:

1. **Supervised** - (SVL)
2. **Unsupervised** - (USL)
3. **Reinforcement - Learning** (RFL)

See the information in internet, for example the following link:

<https://towardsdatascience.com/what-are-the-types-of-machine-learning-e2b9e5d1756f>

Give of short descriptions of the categories and explain the differences (~5 minutes for each category).

Complete solutions are found in “Exercises2Lecture.pdf”

1. **ML1: Introduction to Machine Learning (ML)**
2. ML2: Concept Learning: VSspaces & Cand. Elim. Algo.
3. ML3: Supervised and Unsupervised Learning
4. ML4: Decision Tree Learning
5. ML5: simple Linear Regression (sLR) & multiple Linear Regression (mLR)
6. ML6: Neural Networks: Convolutional NN
7. ML7: Neural Network: BackPropagation Algorithm
8. ML8: Support Vector Machines (SVM)

ML1 - Introduction to Machine Learning (ML)



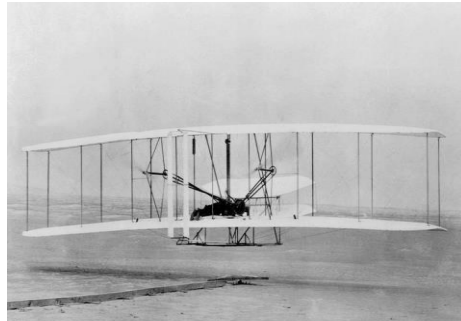
<https://www.youtube.com/watch?v=5dLG3JDk2VU>

<https://www.youtube.com/watch?v=XHjIqQBSPjk>

Status: 1. Dezember 2020

Page 14

Motivation1: the New Revolution through Digitalization



Like the dream of flying, artificial intelligence (AI) has long been a dream and vision. Muscle power was not enough to enable man to fly (see "Schneiderlein von Ulm"). Even the Wright brothers would not have gotten so far using a steam engine weighing tons. Only the discovery of the light diesel engine brought the breakthrough. Similar in computing only the quantum jump in performance enabled the new digital revolution. See next slide.

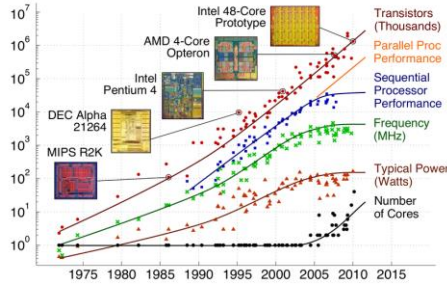
Status: 1. Dezember 2020

Page 15

Motivation2: Why Machine Learning (ML) now?

There are three **main drivers** for Digitalization (i.e. ML), compared to end of the last century (1997-1998) - only 20 years ago:

- ~ 100-times **better CPU power** available using cloud-based infrastructure
 - ~ 100-times **more data** (and more use cases) available than 20 years ago
 - ~ 100-times **better mathematical** algorithms and models
- ====> ~ 1 million better possibilities



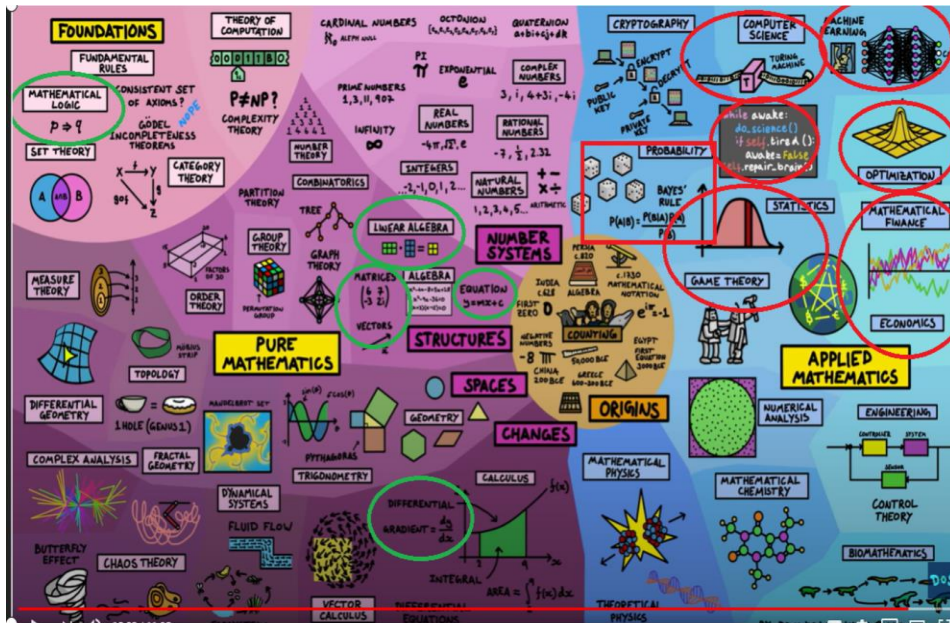
Moore's law is the observation that the number of transistors in a dense integrated circuit doubles about every two years.

https://en.wikipedia.org/wiki/Moore%27s_law

Status: 1. Dezember 2020

Page 16

<https://www.youtube.com/watch?v=OmJ-4B-mS-Y>



Motivation3: Germans are skeptical about the use AI

In den letzten Jahren konnten Forscher und Ingenieure große technologische Fortschritte im Bereich künstliche Intelligenz machen. Künstliche Intelligenz wird in immer mehr Produkten und Dienstleistungen eingesetzt. Manche Experten sehen darin einen großen Fortschritt, andere Experten warnen vor möglichen Risiken.
Ist der Einsatz von künstlicher Intelligenz Ihrer Meinung nach im Allgemeinen mit mehr Risiko oder mehr Nutzen verbunden - oder sind Risiko und Nutzen ungefähr gleich groß?

Repräsentative Befragung in Deutschland (ab 18 Jahre), n=2.000 Befragte, Feldzeit: 24. bis 28. August 2018



The main obstacles that prevent ML in Germany to really take of are:

1. It is not clear to many what ML and digitization is and how it will change their daily work.
2. There is a "fear" of new technologies. Some persons fear to loose familiar procedures others even to loose their working place entirely.
3. German companies (in the majority of the industries) are technologically, organizationally and intellectually unprepared for ML (i.e. digitization)

In the most cases where persons are interviewed about the usage of KI, it was not used in production-processes in the German companies (see chemistry, travel, logistics, machine building, etc.). Only in Automobile (20%) or Finance (~10%) we see some progress in this area (compare slide 23) . Currently the topic of artificial intelligence is dividing the spirits. On the one hand, it brings great progress, on the other, it carries risks that are difficult to assess. This becomes evident in the discussion about self-driving cars which would make road traffic much safer but are highly debated and are feared to be partly unpredictable. Tesla founder Elon Musk warns against the use of artificial intelligence, which could be more dangerous for humanity than nuclear weapons. The Germans are also skeptical about the use of Artificial Intelligence in general as above YouGov survey clearly shows.

Understanding - What is Machine Learning (ML)?



What is machine learning?

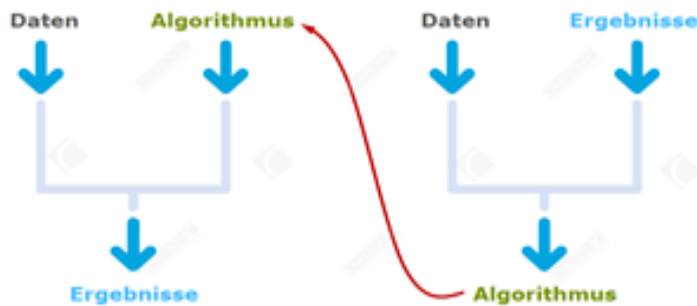
The answer is, in one word, **algorithms**.

Machine learning is, more or less, a way for computers to learn things without being specifically programmed. But how does that actually happen?

Algorithms are sets of rules that a computer is able to follow. Think about how you learned to do long division – maybe you learned to take the denominator and divide it into the first digits of the numerator, then subtracting the subtotal and continuing with the next digits until you were left with a remainder.

Well, that's an algorithm, and it's the sort of thing we can program into a computer, which can perform these sorts of calculations much, much faster than we can.

Traditional Software vs. Machine Learning:



Definition of Machine Learning (ML), see Ref. [TMitch]

▪ Definition (Mitchell 1997)

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

▪ Given:

- a task T
- a performance measure P
- some experience E with the task

▪ Goal:

- generalize the experience in a way that allows to improve your performance on the task

See the book „Machine Learning“ from Tom Mitchell, McGraw Hill, 1997:

<https://www.cs.cmu.edu/~tom/mlbook.html>

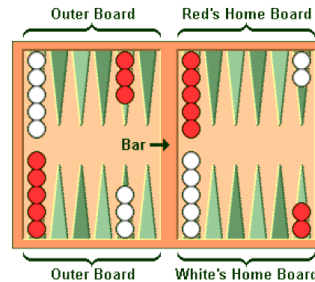
Machine Learning is the study of computer algorithms that improve automatically through experience. Applications range from datamining programs that discover general rules in large data sets, to information filtering systems that automatically learn users' interests. This book provides a single source introduction to the field. It is written for advanced undergraduate and graduate students, and for developers and researchers in the field. No prior background in artificial intelligence or statistics is assumed.

See the following **two examples**:

1. **Chess Playing**, where Task T is playing chess. Performance measure P is percent of games won against opponents and Training experience E is playing practice games against itself.
2. **Robot Driving**, where Task T is driving on public four-lane highways using vision sensors. Performance measure P is average distance traveled before an error (as judged by human overseer) and Training experience E is a sequence of images and steering commands recorded while observing a human driver.

Example1 of ML - Learning to Play Backgammon

- Task:
 - play backgammon
- Performance Measure:
 - percentage of games won
- Experience:
 - previous games played



TD-Gammon:

- learned a neural network for evaluating backgammon boards
- from playing millions of games against itself
- successively improved to world-champion strength
- <http://www.research.ibm.com/massive/tld.html>
GNU Backgammon: <http://www.gnu.org/software/gnubg/>

GNU Backgammon (GNUbg) plays and analyzes backgammon games and matches.

It is able to play and analyze both money games and tournament matches, evaluate and roll out positions, and more.

Driven by a command-line interface, it displays an ASCII rendering of a board on text-only terminals, but also allows the user to play games and manipulate positions with a [graphical GTK+ interface](#).

GNU Backgammon is extensible on platforms which support Python.

Example2 of ML - Recognizing Spam-Mail

- Task:
 - sort E-mails into categories (e.g., Regular / Spam)
- Performance Measure:
 - Weighted Sum of Mistakes (letting spam through is not so bad as misclassifying regular E-mail as spam)
- Experience:
 - Handsorted E-mail messages in your folder



In Practice:

- Many Spam-Filters (e.g., Mozilla) use Bayesian Learning for recognizing spam mails

Example3 of ML - Market Basket Analysis

- Task:
 - discover items that are frequently bought together
- Performance Measure:
 - ? (revenue by making use of the discovered patterns)
- Experience:
 - Supermarket check-out data

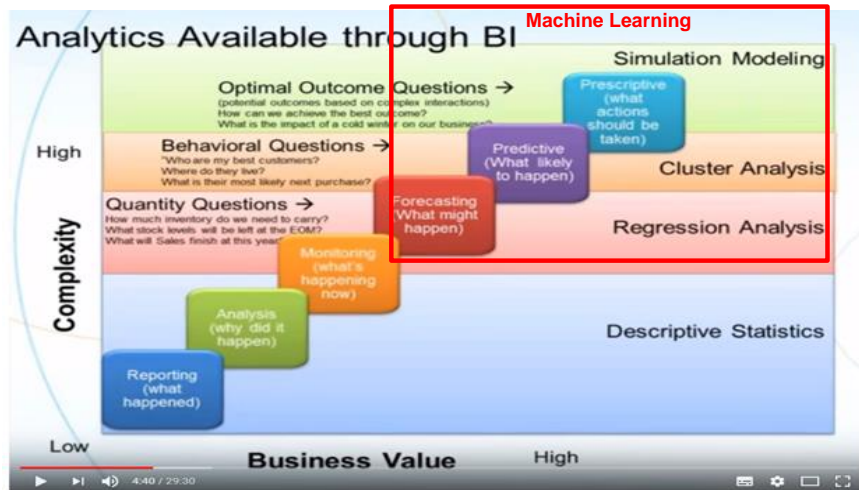
Myth:

- The most frequently cited result is:

diapers → beer



Use-Case 1: Advanced Analytics & Machine Learning (1/3)



Status: 1. Dezember 2020

Page 23

"Advanced Analytics (AA)":

The basis of the technical and technical content was a YouTube video that can be found on the Internet on this topic:

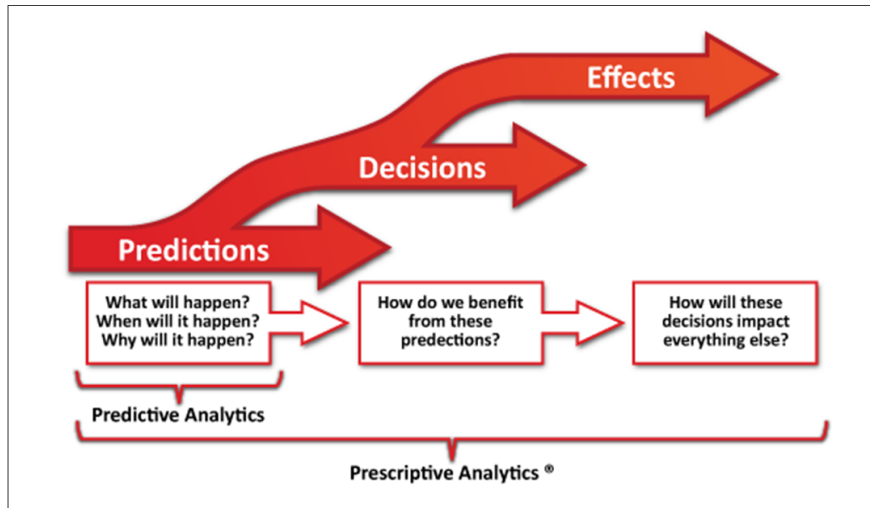
<https://www.youtube.com/watch?v=oNNk9-tmsZY>

As an introductory slide for a better understanding and overview of AA, we are used to mention the six most important methods of "Advanced Analytics". Their complexity increases from bottom to top and at the same time the professional added value that can be achieved with it increases from left to right. The simplest and least valuable subject is thus the reporting or also called ("reporting") (bottom left).

The most difficult to implement but also the most technically valuable method is the descriptive predictive model "Prescriptive" (top right).

For Forecasting, Predictive Analysis and Precrptive Analysis we will use ML.

Use-Case 1: Advanced Analytics & Machine Learning (2/3)



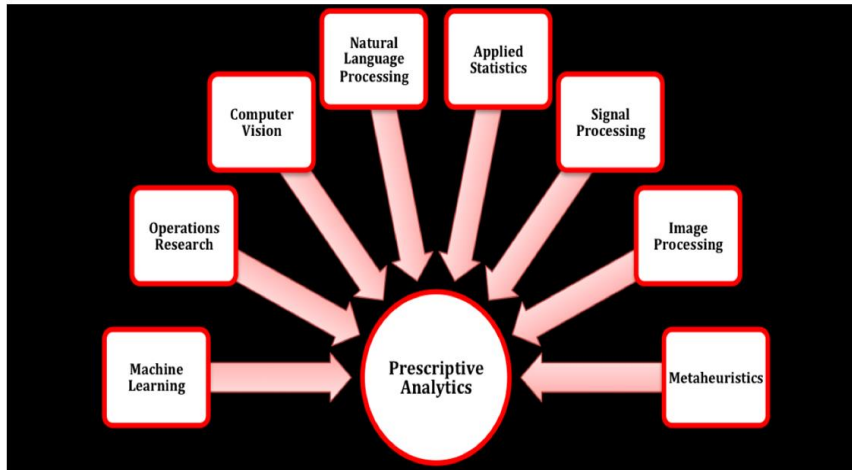
Prescriptive analytics incorporates both structured and unstructured data, and uses a combination of advanced analytic techniques and disciplines to predict, prescribe, and adapt. While the term prescriptive analytics was first coined by IBM and later trademarked by Ayata, the underlying concepts have been around for hundreds of years.

The technology behind prescriptive analytics synergistically combines hybrid data, business rules with mathematical models and computational models.

The data inputs to prescriptive analytics may come from multiple sources: internal, such as inside a corporation; and external, also known as environmental data. The data may be structured, which includes numbers and categories, as well as unstructured data, such as texts, images, sounds, and videos.

Unstructured data differs from structured data in that its format varies widely and cannot be stored in traditional relational databases without significant effort at data transformation.^[10] More than 80% of the world's data today is unstructured, according to IBM.

Use-Case 1: Advanced Analytics & Machine Learning (3/3)



In addition to this variety of data types and growing data volume, incoming data can also evolve with respect to velocity, that is, more data being generated at a faster or a variable pace. Business rules define the [business process](#) and include objectives constraints, preferences, policies, best practices, and boundaries. Mathematical models and computational models are techniques derived from mathematical sciences, computer science and related disciplines such as applied statistics, machine learning, operations research, natural language processing, computer vision, pattern recognition, image processing, speech recognition, and signal processing.

The correct application of all these methods and the verification of their results implies the need for resources on a massive scale including human, computational and temporal for every Prescriptive Analytic project.

In order to spare the expense of dozens of people, high performance machines and weeks of work one must consider the reduction of resources and therefore a reduction in the accuracy or reliability of the outcome. The preferable route is a reduction that produces a probabilistic result within acceptable limits.

Use-Case 2: Use AI for creating Art (ex. V. van Gogh) (1/3)

<https://youtu.be/ARotWjhRpjE>

Status: 1. Dezember 2020

Page 26

Cambridge Consultants — A new artificial intelligence system can turn simple sketches into paintings reminiscent of works by great artists of the 19th and 20th centuries, researchers say.

The [artificial intelligence](#) (AI) system, dubbed Vincent, learned to paint by "studying" 8,000 works of art from the Renaissance up to the 20th century. According to the system's creators — engineers from the United Kingdom-based research and innovation company Cambridge Consultants — Vincent is unique not only in its ability to make art that is actually enjoyable but also in its capability to respond promptly to human input. "Vincent allows you to draw edges with a pen, edges of a picture you can imagine in your mind, and from those pictures, it produces a possible painting based on its training," said Monty Barlow, director of [machine learning](#) at Cambridge Consultants, who led the project. "There is this concern that artificial intelligence will start replacing people doing things for them, but Vincent allows humans to take part in the decisions of the creativity of artificial intelligence." [[Super-Intelligent Machines: 7 Robotic Futures](#)]

Use-Case 2: Use AI for creating Art (ex. V. van Gogh) (2/3)

<https://blog.netapp.com/how-vincent-ai-learned-to-paint/>

Status: 1. Dezember 2020

Page 27

Teaching Vincent

Barlow said that using only 8,000 works of art to train Vincent is by itself a major achievement. Previously, a similar system would have needed millions, or even billions, of [samples to learn to paint](#).

"Most machine learning deployed today has been about classifying and feeding lots and lots of examples into a system," Barlow said. "It's called supervised learning. You show a million photos of a face, for example, and a million photos of not a face, and it learns to detect faces."

Vincent uses a more sophisticated technique that allows the machine to teach itself automatically, without constant human input. The system behind Vincent's abilities is based on the so-called generative adversarial network, which was first described in 2014.

The technique uses two [neural networks](#) that compete with each other. At the beginning, both networks are trained, for example, on images of birds. Subsequently, one network is tasked with producing more images of birds that would persuade the other network that they are real. Gradually, the first network gets better at producing realistic images, while the second one gets better at spotting fakes, according to the researchers.

Use-Case 2: Use AI for creating Art (ex. V. van Gogh) (3/3)

TURN ANY PHOTO INTO AN ARTWORK – FOR FREE!

We use an algorithm inspired by the human brain. It uses the stylistic elements of one image to draw the content of another. Get your own artwork in just three steps.

1 Upload photo

The first picture defines the scene you would like to have painted.



2 Choose style

Choose among predefined styles or upload your own style image.



3 Submit

Our servers paint the image for you. You get an email when it's done.



[Try it now](#)

<https://deepart.io/>

Use-Case 3: Use AI/Digitalization in the Medicine of the Future

Of course, when studying the current applications of AI, medicine is a large and broad field of AI applications. Three important fields of application are:

1. **Monitoring:** When the smartphone notices the disease before it breaks out.



Apple Watch 4: With the new product generation the tech company is expanding the device into a health guard. With a software update, Apple released a new ECG function. Two sensors on the underside of the clock record the electrical impulses of the heartbeat. The measurement results are recorded in an app on the iPhone. The application will tell you if it detects any signs of cardiac arrhythmia after about 30 seconds. The results of ECG features stores the app on the iPhone. The owner can later share it with the doctor via PDF.

2. **Diagnosis:** When computer and doctor are reading x-rays together.

Above the strongly enlarged, purple-colored cells, light green circles float in the picture. They mark where Google's Artificial Intelligence found cancerous tissue in the sample. As Bob McDonald, Technical Program Manager at the software company, moves the small glass plate under the microscope, he scours through the next segment of the cell...

3. **Therapy:** When every patient gets his own medication

Not only for prophylaxis and diagnosis, but also for completely new therapies, the IT revolution in medicine can pave the way. This is shown, for example, by a project that is currently being driven by the biotech company Biontech in Mainz with high investments. In cooperation with the US company Genentech, a subsidiary of the Roche Group, Biontech is working on the development of completely individualized vaccines designed to trigger a body immune reaction against tumor cells.

The *Handelsblatt* has written a good report. "Digitization and Health - The Medicine of the Future - How AI protects us against cancer and heart attack": <http://a.msn.com/05/de-de/BBT0ICR?ocid=se>

Status: 1. Dezember 2020

Page 29

1. **Monitoring:** Significant deviations in essential body data will be automatically analyzed in the future: is the increased heart rate a normal consequence of the staircase that has just been climbed? Or does it indicate cardiovascular disease when combined with other data and the patient's history? Thus, diseases can be detected in early stages and treated accordingly effectively.

2. **Diagnostics:** Where it depends today almost exclusively on the knowledge and the analysis ability of the physician, whether for example the cancer metastasis on the X-ray image is recognized as such, support the physician in the future artificially intelligent systems, which become a little smarter with each analyzed X-ray image. The probability of error in the diagnosis decreases, the accuracy in the subsequent treatment increases.

3. **Therapy:** Big data & AI have the potential to make the search for new medicines and other treatment methods significantly more efficient. Today, countless combinations of molecules must first be tested for their effectiveness in the Petri dish, then in animal experiments, and finally in clinical trials, until at the end there may be a new drug. A billion dollar roulette game in which the chances of winning can be significantly increased by computer-aided forecasting, which in turn access an unprecedented wealth of research data. **Examples:**

- „KI entwickelt Medikament“: <https://t3n.de/news/vollstaendig-ki-entwickeltes-1248880/> ;
- “Corona Virus”: <https://futurezone.at/science/coronavirus-ki-warnte-schon-2019-vor-dem-ausbruch/400738542>

What does Machine Learning look like?

What does **machine learning** look like?

In machine learning, our goal is either prediction or clustering. Prediction is a process where, from a set of input variables, we estimate the value of an output variable. For example, using a set of characteristics of a house, we can predict its sale price. Prediction problems are divided into two main categories:

0 1 2 3 4
5 6 7 8 9 6 8 7 9 5
4 3 2 1 0

Regression problems, where the variable to predict is numerical (e.g., the price of a house).



Classification problems, where the variable to predict is part of one of some number of pre-defined categories, which can be as simple as "yes" or "no." (for example, predict whether a certain piece of equipment will experience a mechanical failure)

The most prominent and common algorithms used in machine learning historically and today come in three groups: **linear models, tree-based models, and neural networks.**

The 3 major Types of Machine Learning algorithms (1/2)

Linear Model Approach



A **linear model** uses a simple formula to find the "best fit" line through a set of data points. This methodology dates back over 200 years, and it has been used widely throughout statistics and machine learning. It is useful for statistics because of its simplicity – the variable you want to predict (the dependent variable) is represented as an equation of variables you know (independent variables), and so prediction is just a matter of inputting the independent variables and having the equation spit out the answer.

For example, you might want to know how long it will take to bake a cake, and your regression analysis might yield an equation $t = 0.5x + 0.25y$, where t is the baking time in hours, x is the weight of the cake batter in kg, and y is a variable which is 1 if it is chocolate and 0 if it is not. If you have 1 kg of chocolate cake batter (we love cake), then you plug your variables into our equation, and you get $t = (0.5 \times 1) + (0.25 \times 1) = 0.75$ hours, or 45 minutes.

Tree-Based Model Approach



When you hear tree-based, think decision trees, i.e., a sequence of branching operations.

A **decision tree** is a graph that uses a branching method to show each possible outcome of a decision. Like if you're ordering a salad, you first decide the type of lettuce, then the toppings, then the dressing. We can represent all possible outcomes in a decision tree. In machine learning, the branches used are binary yes/no answers.

Neural Networks









Neural networks refer to a biological phenomenon comprised of interconnected neurons that exchange messages with each other. This idea has now been adapted to the world of machine learning and is called ANN (Artificial Neural Networks).

Deep learning, which you've heard a lot about, can be done with several layers of neural networks put one after the other.

ANNs are a family of models that are taught to adopt cognitive skills.

The 3 major Types of Machine Learning algorithms (2/2)

TYPE	NAME	DESCRIPTION	ADVANTAGES	DISADVANTAGES
Linear	 Linear regression	The "best fit" line through all data points. Predictions are numerical.	Easy to understand -- you clearly see what the biggest drivers of the model are.	<ul style="list-style-type: none"> X Sometimes too simple to capture complex relationships between variables. X Does poorly with correlated features.
	 Logistic regression	The adaptation of linear regression to problems of classification (e.g., yes/no questions, groups, etc.)	Also easy to understand.	<ul style="list-style-type: none"> X Sometimes too simple to capture complex relationships between variables. X Does poorly with correlated features.
Tree-based	 Decision tree	A series of yes/no rules based on the features, forming a tree, to match all possible outcomes of a decision.	Easy to understand.	<ul style="list-style-type: none"> X Not often used on its own for prediction because it's also often too simple and not powerful enough for complex data.
	 Random Forest	Takes advantage of many decision trees, with rules created from subsamples of features. Each tree is weaker than a full decision tree, but by combining them we get better overall performance.	A sort of "wisdom of the crowd". Tends to result in very high quality models. Fast to train.	<ul style="list-style-type: none"> X Models can get very large. X Not easy to understand predictions.
	 Gradient Boosting	Uses even weaker decision trees, that are increasingly focused on "hard" examples.	High-performing.	<ul style="list-style-type: none"> X A small change in the feature set or training set can create radical changes in the model. X Not easy to understand predictions.
Neural networks	 Neural networks	Interconnected «neurons» that pass messages to each other. Deep learning uses several layers of neural networks stacked on top of one another.	Can handle extremely complex tasks - no other algorithm comes close in image recognition.	<ul style="list-style-type: none"> X Very slow to train, because they often have a very complex architecture. X Almost impossible to understand predictions.

10 Fundamental Terms of Machine Learning (1/2)



model (n)

[ˈmɒdəl] / noun

1. a mathematical representation of a real world process; a predictive model forecasts a future outcome based on past behaviors.



algorithm (n)

[ˈælgərɪðəm] / noun

1. a set of rules used to make a calculation or solve a problem.



training (v)

[ˈtreɪnɪŋ] / verb

1. the process of creating a model from the training data. The data is fed into the training algorithm, which learns a representation for the problem, and produces a model. Also called "learning".



regression (n)

[rɪˈgrɛʃən] / noun

1. a prediction method whose output is a real number, that is, a value that represents a quantity along a line. Example: predicting the temperature of an engine or the revenue of a company.



classification (n)

[ˌklæsəfəˈkeɪʃən] / noun

1. a prediction method that assigns each data point to a predefined category, e.g., a type of operating system.



target (n)

[ˈtɑrɡət] / noun

1. in statistics, it is called the dependent variable; it is the output of the model or the variable you wish to predict.

10 Fundamental Terms of Machine Learning (2/2)



training set (n)

[ˈtreɪnɪŋ set] / noun

1. a dataset used to find potentially predictive relationships that will be used to create a model.



test set (n)

[test set] / noun

1. a dataset, separate from the training set but with the same structure, used to measure and benchmark the performance of various models.



feature (n)

[ˈfi:tʃə] / noun

1. also known as an independent variable or a predictor variable, a feature is an observable quantity, recorded and used by a prediction model. You can also engineer features by combining them or adding new information to them.

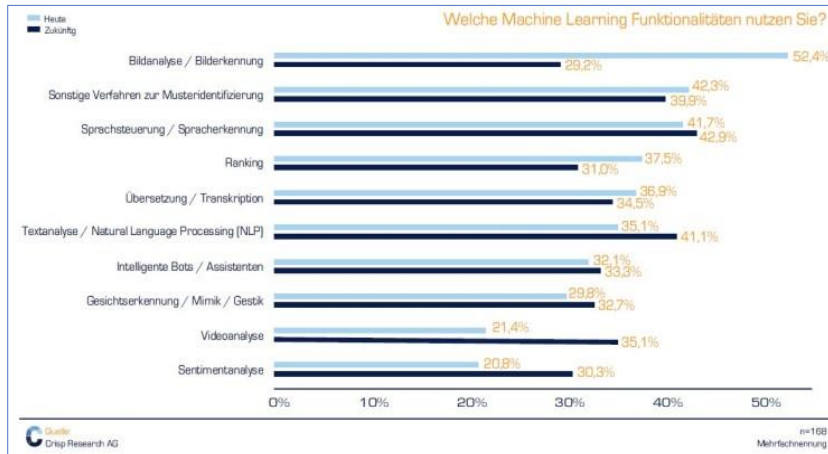


overfitting (v)

[ˌoʊvərˈfi:tɪŋ] / verb

1. a situation in which a model that is too complex for the data has been trained to predict the target. This leads to an overly specialized model, which makes predictions that do not reflect the reality of the underlying relationship between the features and target.

What is the status of ML and what are future plans



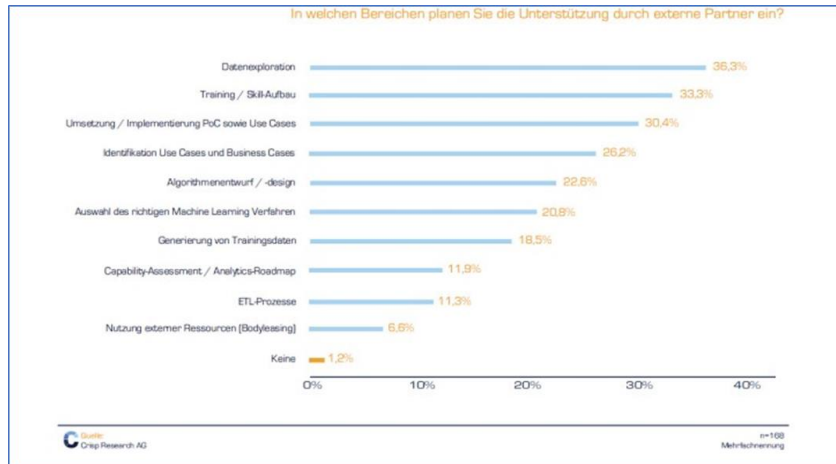
Today, **machine-learning algorithms** are mainly used in the field of image analysis and recognition. In the future, speech recognition and processing will become more important. The processing and analysis of large amounts of data is a core task of such a digital infrastructure platform.

Therefore, IT managers must ensure that their IT can handle different artificial intelligence processes. Server, storage and network infrastructures must be designed for new ML-based workloads. Data management must also be prepared so that ML-as-a-Service offerings in the cloud can be used. In the context of ML, alternative hardware components such as GPU-based clusters from Nvidia, Google's Tensor Processing Unit (TPU) or IBM's TrueNorth processor have become popular in recent months.

Companies have to decide whether they want to invest themselves or use the services of corresponding cloud providers. One of the major uses for ML is speech recognition and processing. Amazon Alexa is currently moving into households, Microsoft, Google, Facebook and IBM have invested here a large part of their research and development funds and purchased specialist firms.

It can be foreseen that natural language communication becomes more natural at the customer interface. The operation of digital products and enterprise IT solutions will also be possible via voice command. This affects both the customer frontend and the IT backend.

In what ML Tasks customers need external skills



With large cloud providers including ML services and products in their service portfolio, it's relatively easy for users to get started. Amazon Machine Learning, Microsoft Azure Machine Learning, IBM Bluemix, and Google Machine Learning provide cost-effective access to related services through the public cloud.

So users do not need their own supercomputers, statistical experts or dedicated infrastructure management. You can start with a few commands about the APIs of the big public cloud providers.

They will find various machine learning techniques, services and tools such as graphical programming models and storage services.

The more they get involved, the greater the risk of vendor lock-ins. Therefore, users should think about their strategy before starting. IT service providers and managed service providers can also deploy and operate ML systems and infrastructures, making independence from the public cloud providers and their SLAs equally possible.

Status of Industries in the Usage of ML Technologies

	In Evaluierung / Planung	Erste Erfahrungen & Prototypen	Nutzung in ausgewählten Einsatzbereichen	Einsatz im Produktivbetrieb in weiten Teilen des Unternehmens
Automobil und Automobilzulieferer	60,0%	20,0%	0,0%	20,0%
Maschinen- und Anlagenbau	52,9%	17,7%	29,4%	0,0%
Chemische Industrie	30,0%	40,0%	30,0%	0,0%
Metallverarbeitende Industrie	38,5%	46,2%	15,4%	0,0%
Konsumgüter und Handel	18,8%	43,8%	25,0%	12,5%
Logistik und Verkehr	16,7%	41,7%	41,7%	0,0%
Banken und Versicherungen	28,6%	33,3%	28,6%	9,5%

The **usage behavior of ML** is very different not only between, but also within the industries. In the automotive industry, for example, there are big gaps between the pioneers and the latecomers. Real-time image and video analysis and statistical methods and mathematical models from machine learning and deep learning are widely used for the development and production of self-driving cars. Some methods are also used to detect manufacturing defects. The share of innovators, who already use ML to a large extent, is the largest in the automotive industry at around 20 percent. In contrast, however, there are 60 percent who deal with ML but are still in the evaluation and planning phase. Thus, it turns out that in the automotive industry, some lighthouses shape the picture, but from a nationwide adaptation, there is no question. The mechanical and plant engineering companies also have half (53 percent) in the evaluation and planning phase. Nearly one-third use ML productively in selected applications and 18 percent are currently building prototypes. Next are the commercial and consumer goods companies, which are 44 percent to test ML in initial projects and prototypes. This is not surprising given that these companies usually have well-maintained data sets and a lot of experience in business intelligence and data warehouses. If they succeed in measurably improving pricing strategies, product availability or marketing campaigns, ML is seen as a welcome innovation tool for existing big-data strategies. The same applies to the IT, telecoms and media industries: there ML processes have long been used, for example, for playing online advertising, calculating purchase probabilities (conversion rates) or personalizing web content and shopping recommendations. For professional service providers, measuring and improving customer loyalty, quality of service and on-time delivery play an important role, as these are the competitive differentiating factors.

ML platforms & ML products



Status: 1. Dezember 2020

Page 38

When it comes to **selecting platforms and products**, solutions from the public cloud play an increasingly important role (ML as a Service). In order to avoid complexity and because the major cloud providers are also the leading innovators in this field, many users are choosing these cloud solutions. While 38.1 of the respondents prefer solutions from the public cloud, 19.1 percent choose proprietary solutions from selected providers and 18.5 percent open source alternatives. The rest either follow a hybrid strategy (15.5 percent) or have not yet formed an opinion (8.8 percent).

Among the **cloud-based solutions** AWS (<https://aws.amazon.com/de/machine-learning/>) has the highest level of awareness: 71 percent of the decision makers indicate that they Amazon in this context is known. Even Microsoft, Google and IBM are the survey respondents to more than two-thirds in the ML environment. Interestingly enough, only 17 percent of respondents use AWS cloud services in the context of evaluation, design and production operations for ML. About one third of the respondents in each case deal with IBM Watson ML (<https://www.ibm.com/cloud/machine-learning>), Microsoft Azure ML Studio (<https://azure.microsoft.com/en-us/services/machine-learning-studio>) or the Google Cloud ML Platform (<https://cloud.google.com/ml-engine/docs/tensorflow/technical-overview>). The analysts believe that this has a lot to do with the manufacturers' marketing efforts. Accordingly, IBM and Microsoft are investing heavily in their cognitive and AI strategies. Both have strong SME and wholesale distribution and a large network of partners. Google, however, owes its position to the image as a huge data and analytics machine, which drives the market through many innovations - such as Tensorflow, many ML-APIs and their own hardware. After all, HP Enterprise with "Haven on Demand" is also one of the relevant ML players and is used by 14%.

Ethics in Artificial Intelligence (AI)

Lecture on GI Regional Group Stg./BB from 14.1.19: "How 'ethics' can be integrated into AI programs" by Michael Mörike, (board member Integrata-Stiftung Tübingen):

The development of artificial intelligence is rapid and explosive at the same time: Algorithms that decide for us, and the capabilities of machines that surpass us humans already raise many ethical questions: How should an autonomous car behave in everyday life? Which rules will apply to robots in the future? Is it possible to include ethics in AI?

The latter question is at the center of the lecture. The author assumes this could work and tries to explain how to do that. He differentiates between human ethics in dealing with AI, which he calls external ethics, and ethics, which he calls machinery morality. In the lecture the necessity and the benefit as well as the feasibility are discussed.

In the Special Interest Group AI at bwcon, we address ethical issues and challenges in the rapid development of artificial intelligence with business representatives from southern Germany.

[More about Special Interest Group](#)

See also Stanley Kubrick's famous movie from 1968: "2001: A Space Odyssey"

<https://www.youtube.com/watch?v=XHjIqQBSPjk>

HAL 9000: "I'm sorry Dave, I'm afraid I can't do that" & "Deactivation of HAL 9000"

<https://www.youtube.com/watch?v=ARJ8cAGm6JE&t=42s>

https://www.youtube.com/watch?v=c8N72t7aScY&list=PLawr1rgf_CvSiNsWPbL0OrMKbcZRHJud7&index=25

Ethics of AI and ML is very important and should also be mentioned in a lecture about ML. Better understanding ML methods give the students also the ability to better understand and evaluate the ethical aspects of the technology.

Examples such as these cause skepticism as to whether algorithms judge more objectively than humans. They are only as good as the developers who program them - and how the data they are fed with. Therefore, they must be understandable, because AI does not always have to be intelligent, let alone meaningful. On the other hand, people are mistaken. Especially in the area of personnel decisions on the stomach are made or the applicants are preferred, who are similar to the hiring person. Intelligent programs can counteract if they have been cleverly designed and scrutinized. You can give applicants a chance that would never have been invited. So there is no simple solution. Which program is clever and which tasks it should and is supposed to take over - that is why people still have to decide.

Consider an Interview with Carsten Kraus: „Deep Neural Networks könnten eigene Moralvorstellungen entwickeln“

<https://ecommerce-news-magazin.de/e-commerce-news/e-commerce-interviews/interview-mit-carsten-kraus-deep-neural-networks-koennten-eigene-moralvorstellungen-entwickeln/>

Exercises to Lesson ML1

Homework H1.1 – “Most Popular ML Technologies + Products”

Groupwork (3 Persons). Look on the three most used ML technologies/products (see information in internet):

1. IBM Watson Machine Learning - <https://www.ibm.com/cloud/machine-learning>
2. Microsoft Azure ML Studio - <https://azure.microsoft.com/en-us/services/machine-learning-studio/>
3. Google Cloud Machine Learning Plattform - <https://cloud.google.com/ml-engine/docs/tensorflow/technical-overview>

Give of short overview about the products and its features (~10 minutes for each) und give a comparison matrix of the 3 products and an evaluation. What is your favorite product (~ 5 minutes).

Complete solutions are found in “Exercises2Lecture.pdf”

Exercises to Lesson ML1

Homework H1.2 - “Ethics in Artificial Intelligence”

Groupwork (2 Persons) - evaluate the interview with Carsten Kraus (Founder Omikron/Pforzheim, Germany): „Deep Neural Networks könnten eigene Moralvorstellungen entwickeln“.

<https://ecommerce-news-magazin.de/e-commerce-news/e-commerce-interviews/interview-mit-carsten-kraus-deep-neural-networks-koennten-eigene-moralvorstellungen-entwickeln/>

The victory of Google-developed DeepMind-Software AlphaGo against South Korean Go-world champion Lee Sedol does not simply ring in the next round of industrial revolution. According to IT expert Carsten Kraus, the time of superiority of Deep Neural Networks (DNN) with respect to human intelligence has now began.

Homework H1.3 (optional) – “Create Painting with DeepArt”

1 Person – Create your own painting by using DeepArt company in Tübingen (<https://deepart.io/>). What ML method did you use to create “paintings”?



Not only in Silicon Valley, but also in the northern Black Forest, the artificial intelligence is driven forward. The medium-sized company Omikron wrestles with Google for specialists and sells its search technology to the greats: Renault, Fresenius and Siemens rely on the machine learning from Pforzheim.

Omikron founder Carsten Kraus understood early on how to make calculating machines more efficient and intelligent. As a gifted prodigy, he founded the company during school hours. The startup mentality he has preserved until today.

Complete solutions are found in “Exercises2Lecture.pdf”

Exercises to Lesson ML1

Homework H1.4 (optional) – Summary of video “*What is ML?*”

1 Person - summarizes the results of the first YouTube Video “What is Machine Learning” by Andrew Ng in a report of 10 minutes. Create a small PowerPoint presentation. See:

https://www.youtube.com/playlist?list=PLLsT5z_DsK-h9vYZkQkYNWcItqhIRJLN

Homework H1.5 (optional)– Summary of video “*Supervised - & Unsupervised Learning*”

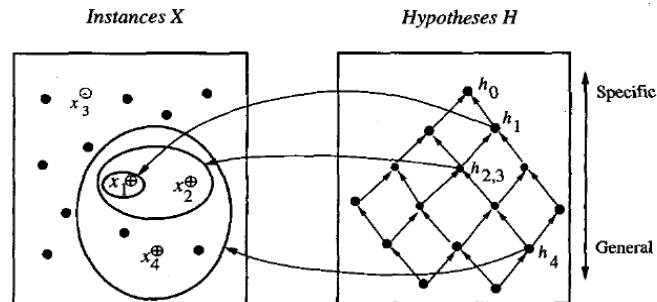
Groupwork (2 Persons) - summarizes the results of the second and third YouTube Video “Supervised Learning” and “Unsupervised Learning” by Andrew Ng in a Report of 15 Minutes. Create a small PowerPoint presentation. See:

https://www.youtube.com/playlist?list=PLLsT5z_DsK-h9vYZkQkYNWcItqhIRJLN

Complete solutions are found in “Exercises2Lecture.pdf”

1. ML1: Introduction to Machine Learning (ML)
2. ML2: Concept Learning: VSspaces & Cand. Elim. Algo.
3. ML3: Supervised and Unsupervised Learning
4. ML4: Decision Tree Learning
5. ML5: simple Linear Regression (sLR) & multiple Linear Regression (mLR)
6. ML6: Neural Networks: Convolutional NN
7. ML7: Neural Network: BackPropagation Algorithm
8. ML8: Support Vector Machines (SVM)

ML2 – Concept Learning: Version Spaces & Candidate Elimination Algorithm



References:

- T. Mitchell, 1997, Chapter 2.
- P. Winston, "Learning by Managing Multiple Models", in P. Winston, *Artificial Intelligence*, Addison-Wesley Publishing Company, 1992, pp. 411-422.

See also:

http://www2.cs.uregina.ca/~dbd/cs831/notes/ml/vspace/3_vspace.html

Motivation

***** placeholder *****

Introduction

***** placeholder *****

Definition of AQ Learning

Definition

AQ learning is a form of supervised machine learning of rules from examples and background knowledge performed by the well-known AQ family of programs and other machine learning methods. AQ learning pioneered separate-and-conquer approach to rule learning in which examples are sequentially covered until a complete class description is formed. Derived knowledge is represented in a highly expressive form of attributional rules.

Theoretical Background

The core of AQ learning is a simple version of A^q (algorithm quasi-optimal) covering algorithm, developed by Ryszard S. Michalski in the late 1960s (Michalski [1969](#)). The algorithm was initially developed for the purpose of minimization of logic functions, and later adapted for rule learning and other machine learning applications.

Simple A^q Algorithm

A^q algorithm realizes a form of supervised learning. Given a set of positive events (examples) P , a set of negative events N , and a...

Example of Candidate Elimination Algorithm (1/3)

Learning the concept of "Japanese Economy Car"

Features: (Country of Origin, Manufacturer, Color, Decade, Type)

Origin	Manufacturer	Color	Decade	Type	Example Type
Japan	Honda	Blue	1990	Economy	Positive
Japan	Toyota	Green	1970	Sports	Negative
Japan	Toyota	Blue	1990	Economy	Positive
USA	Chrysler	Red	1980	Economy	Negative
Japan	Honda	White	1980	Economy	Positive

Solution:

1. Positive Example: (Japan, Honda, Blue, 1990, Economy)

Initialize G to a singleton set that includes everything

Initialize S to a singleton set that includes the first positive example

G = { (?, ?, ?, ?, ?) }

S = { (Japan, Honda, Blue, 1990, Economy) }

(Japan, Honda, Blue, 1990, Economy)

These models represent the most general and the most specific heuristics one might learn.
The actual heuristic to be learned, "Japanese Economy Car", probably lies between them somewhere within the version space.

2. Negative Example: (Japan, Toyota, Green, 1970, Sports)

Specialize G to exclude the negative example.

G = { (?, Honda, ?, ?, ?),
(?, ?, Blue, ?, ?),
(?, ?, ?, 1980, ?),
(?, ?, ?, Economy) }

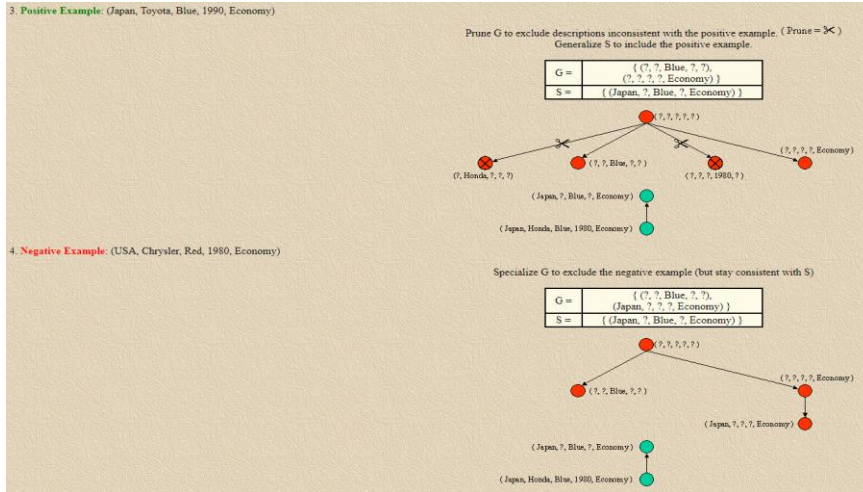
S = { (Japan, Honda, Blue, 1990, Economy) }

(Japan, Honda, Blue, 1990, Economy)

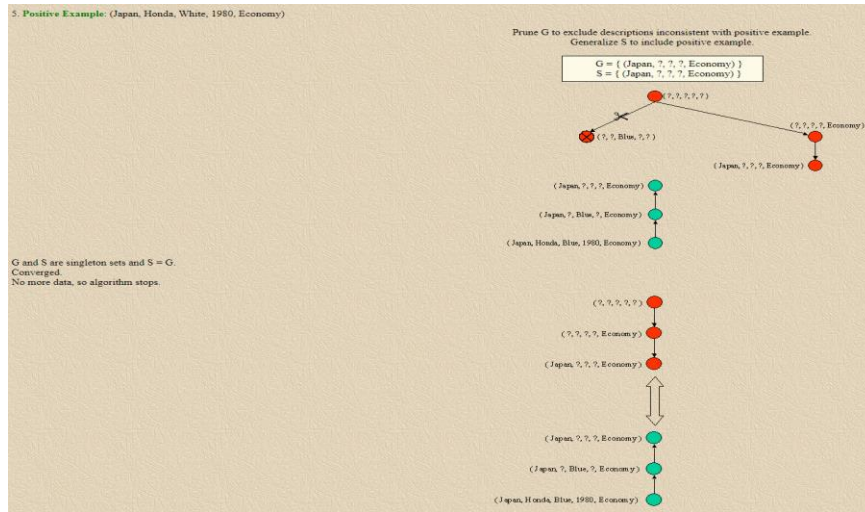
Refinement occurs by generalizing S or specializing G, until the heuristic hopefully converges to one that works well.

<https://onlinelibrary.wiley.com/doi/abs/10.1002/wics.78>

Example of Candidate Elimination Algorithm (2/3)



Example of Candidate Elimination Algorithm (3/3)



Definition of AQ Learning

***** placeholder *****

Exercises to Lesson ML2

Homework H2.1 – “Version Space for “EnjoySport”:

Create the Version Space for the *EnjoySport* concept learning problem with training examples in the following table; see [TMitch], Ch.2 or <https://www.youtube.com/watch?v=cW03t3aZkmE>

Example	<i>Sky</i>	<i>AirTemp</i>	<i>Humidity</i>	<i>Wind</i>	<i>Water</i>	<i>Forecast</i>	<i>EnjoySport</i>
1	Sunny	Warm	Normal	Strong	Warm	Same	Yes
2	Sunny	Warm	High	Strong	Warm	Same	Yes
3	Rainy	Cold	High	Strong	Warm	Change	No
4	Sunny	Warm	High	Strong	Cool	Change	Yes

Homework H2.2 – “ VS ---second example*****

***** placeholder *****

Complete solutions are found in “Exercises2Lecture.pdf”

Exercises to Lesson ML2

Homework 2.3 – “Exercise of an Example with Python”

*****Platzhalter*****

Homework 2.4 – “Exercise of an Example with Python”

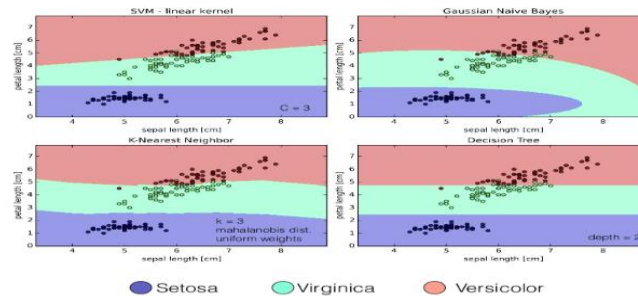
*****Platzhalter*****

Solutions are found in Ref. **[HVö-4]**: “Exercises2Lecture.pdf”

1. ML1: Introduction to Machine Learning (ML)
2. ML2: Concept Learning: VSpaces & Cand. Elim. Algo.
3. **ML3: Supervised and Unsupervised Learning**
4. ML4: Decision Tree Learning
5. ML5: simple Linear Regression (sLR) & multiple Linear Regression (mLR)
6. ML6: Neural Networks: Convolutional NN
7. ML7: Neural Network: BackPropogation Algorithm
8. ML8: Support Vector Machines (SVM)

ML3 – Supervised- and Unsupervised Learning

Iris Example



<https://images.app.goo.gl/ZiyTSGRYcnSzBsB59>

Status: 1. Dezember 2020

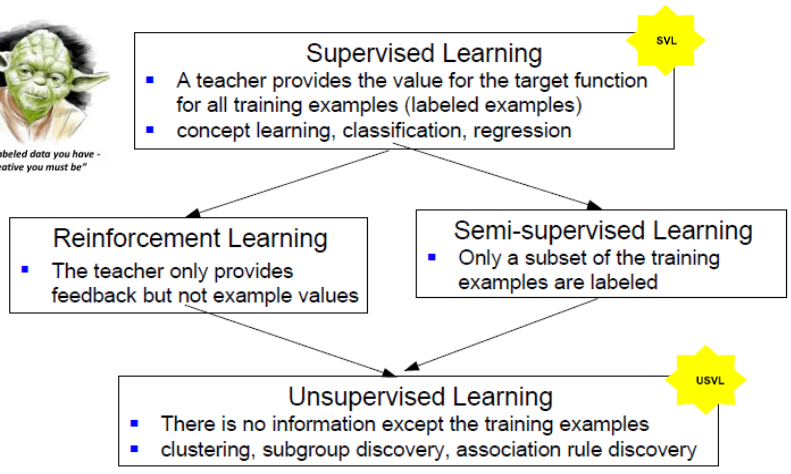
Page 53

In the ML3 Chapter we list the most common concepts and algorithms of SuperVised- (SVL) and UnSuperVised Learning (USVL):
 In esp. under SVL we see Classification methods like Lazy Learning (Rote Learning, kNN Algorithm, etc.) and Bayes Learning for Text Classification and also Regression methods (i.e. simple linear regression).
 In USVL we discuss Clustering- (i.e. K-Means Clustering) and Association Algorithms (i.e. Predictive Market Basket Analysis).

Different Learning Scenarios – Supervised v. Unsupervised



"No labeled data you have - creative you must be"



https://en.wikipedia.org/wiki/Machine_learning#Supervised_and_semi-supervised_learning

Supervised: All data is labeled and the algorithms learn to predict the output from the input data.

Unsupervised: All data is unlabeled and the algorithms learn to inherent structure from the input data.

Semi-supervised: Some data is labeled but most of it is unlabeled and a mixture of supervised and unsupervised techniques can be used.

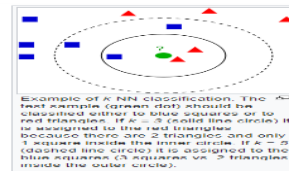
SVL Classification (1/8): Lazy Learning / k-Nearest Neighbors

- No model is learned
 - The stored training instances themselves represent the knowledge
 - Training instances are searched for instance that most closely resembles new instance
- lazy learning

Rote Learning Example

Date	Temperature	Outlook	Humidity	Windy	Play Golf?
07-05	hot	sunny	high	false	no
07-06	hot	sunny	high	true	no
07-07	hot	overcast	high	false	yes
07-09	cool	rain	normal	false	yes
07-10	cool	overcast	normal	true	yes
07-12	mild	sunny	high	false	no
07-14	cool	sunny	normal	false	yes
07-16	mild	rain	normal	false	yes
07-20	mild	sunny	normal	true	yes
07-21	mild	overcast	high	true	yes
07-22	hot	overcast	normal	false	yes
07-23	mild	rain	high	true	no
07-26	cool	rain	normal	true	no
07-30	mild	rain	high	false	yes
today	cool	sunny	normal	false	yes

- Examples:
 - Rote-learner
 - Memorizes entire training data and performs classification only if attributes of record match one of the training examples exactly
 - Nearest-neighbor classifier
 - Uses k "closest" points (nearest neighbors) for performing classification



In machine learning, Lazy Learning is a learning method in which generalization of the training data is, in theory, delayed until a query is made to the system, as opposed to in eager learning, where the system tries to generalize the training data before receiving queries.

Rote learning is a memorization technique based on repetition. The idea is that one will be able to quickly recall the meaning of the material the more one repeats it. Some of the alternatives to rote learning include meaningful learning, associative learning, and active learning.

The primary motivation for employing lazy learning, as in the K-nearest neighbors algorithm, used by online recommendation systems ("people who viewed/purchased/listened to this movie/item/tune also ...") is that the data set is continuously updated with new entries (e.g., new items for sale at Amazon, new movies to view at Netflix, new clips at Youtube, new music at Spotify or Pandora). Because of the continuous update, the "training data" would be rendered obsolete in a relatively short time especially in areas like books and movies, where new best-sellers or hit movies/music are published/released continuously. Therefore, one cannot really talk of a "training phase".

SVL Classification (2/8): Value Difference Metric (VDM)

Value Difference Metric (VDM): The VDM metrics described here address the problem of classification. A system is presented with a training set T , which consists of n instances. Each instance has an input vector x , and an output class c . The problem of classification is to decide what the output class of a new input vector y should be, based on what was learned from the training set, even if y did not appear in the training set.

$$vdm_a(x, y) = \sum_{c=1}^C \left| \frac{N_{a,x,c}}{N_{a,x}} - \frac{N_{a,y,c}}{N_{a,y}} \right|^q = \sum_{c=1}^C |P_{a,x,c} - P_{a,y,c}|^q \quad (3)$$

where

- $N_{a,x}$ is the number of instances in the training set T that had value x for attribute a ;
- $N_{a,x,c}$ is the number of instances in T that had value x for attribute a and an output class c ;
- C is the number of output classes in the problem domain;
- q is a constant, usually 1 or 2; and
- $P_{a,x,c}$ is the conditional probability that the output class is c given that attribute a has the value x . As can be seen from (3), $P_{a,x,c}$ is defined as:

$$P_{a,x,c} = \frac{N_{a,x,c}}{N_{a,x}} \quad (4)$$

Note that $N_{a,x}$ is the sum of $N_{a,x,c}$ over all classes, i.e.,

$$N_{a,x} = \sum_{c=1}^C N_{a,x,c} \quad (5)$$

Business Meaning: The vdm metrics describes the **distance in the behavior of different instances** of the attribute a . See for example the outcome of the concrete calculations in the homework ML3.1

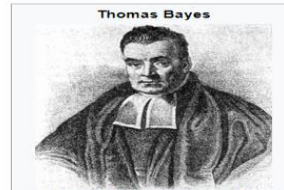
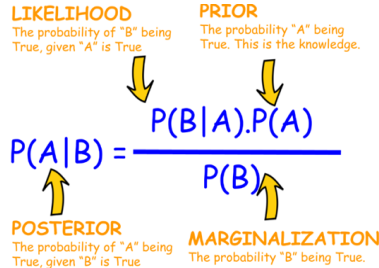
<https://pdfs.semanticscholar.org/f72c/bf9f16f244f5643273fa04c25e2697fe66b9.pdf>

SVL Classification (3/8): Bayes Learning Concept

Bayes Learning (Conditional Probability):

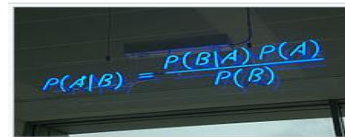
In probability theory and statistics, **Bayes' theorem** describes the probability of an event, based on prior knowledge of conditions that might be related to the event. Bayes theorem is named after Reverend Thomas Bayes (/beɪz/; 1701?–1761).

More details later in this chapter



Portrait purportedly of Bayes used in a 1936 book,^[1] but it is doubtful whether the portrait is actually of him.^[2] No earlier portrait or claimed portrait survives.

Born c. 1701
London, England
Died 7 April 1761 (aged 59)
Tunbridge Wells, Kent, England



A blue neon sign showing the simple statement of Bayes's theorem at the offices of HP Autonomy

Example: see https://en.wikipedia.org/wiki/Bayes%27_theorem

A particular test for whether someone has been using cannabis is 90% sensitive and 80% specific, meaning it leads to 90% true "positive" results (meaning, "Yes he used cannabis") for cannabis users and 80% true negative results for non-users-- but also generates 20% false positives for non-users. **Q.:** Assuming 5% of people actually do use cannabis, what is the probability that a random person who tests positive is really a cannabis user?

Solution: Let P(User | Positive) mean "the probability that someone is a cannabis user given that he tests positive". Then we can write with the formula above:

$$\begin{aligned}
 P(\text{User} | \text{Positive}) &= \frac{P(\text{Positive} | \text{User})P(\text{User})}{P(\text{Positive})} \\
 &= \frac{P(\text{Positive} | \text{User})P(\text{User})}{P(\text{Positive} | \text{User})P(\text{User}) + P(\text{Positive} | \text{Non-user})P(\text{Non-user})} \\
 &= \frac{0.90 \times 0.05}{0.90 \times 0.05 + 0.20 \times 0.95} \approx 19\%
 \end{aligned}$$

Test with Confusion-Matrix

Test \ Actual	User	Non-user	Total
Positive	45	190	235
Negative	5	760	765
Total	50	950	1000

90% sensitive, 80% specific: 45/235 ≈ 19%

Remark: Bayes Learning is called **Naive Bayes** when the value of a particular feature is independent of the value of any other feature, given the class variable. For example, a fruit may be considered to be an apple if it is red, round, and about 10 cm in diameter. A naive Bayes classifier considers each of these features to contribute independently to the probability that this fruit is an apple, regardless of any possible correlations between the color, roundness, and diameter features.

SVL Classification (4/8): Bayes L. for Text Classification

Task: Let's see how this works in practice with a simple example. Suppose we are **building a classifier that says whether a text is about sports or not**. Our training data has 5 sentences:

Training-Text and Target-Text:

No.	Training-Text	Label
1	"A great game"	Sports
2	"The election was over"	Not Sports
2	"Very clean match"	Sports
4	"A clean but forgettable game"	Sports
5	"It was a close election"	Not Sports
Target-Text		
new	"A very close game"	?????????

Bayes' Theorem is useful when working with conditional probabilities (like we are doing here), because it provides us with a way to reverse them. In our case, we have, so using this theorem we can reverse the conditional probability:

$$P(\text{sports}|\text{a very close game}) = \frac{P(\text{a very close game}|\text{sports}) \times P(\text{sports})}{P(\text{a very close game})}$$

Since for our classifier we're just trying to find out which tag has a bigger probability, we can discard the divisor — which is the same for both tags — and just compare

$$P(\text{a very close game}|\text{Sports}) \times P(\text{Sports})$$

with

$$P(\text{a very close game}|\text{Not Sports}) \times P(\text{Not Sports})$$

Page 58

More details:

<https://medium.com/analytics-vidhya/naive-bayes-classifier-for-text-classification-556fabaf252b#:~:text=The%20Naive%20Bayes%20classifier%20is,time%20and%20less%20training%20data.>

The Naive Bayes classifier is a simple classifier that classifies based on probabilities of events. It is the applied commonly to text classification. Though it is a simple algorithm, it performs well in many text classification problems.

Other Pros include less training time and less training data. That is, less CPU and Memory consumption.

As with any machine learning model, we need to have an existing set of examples (training set) for each category (class).

Let us consider sentence classification to classify a sentence to either 'Sports' or 'Not Sports'. In this case, there are two classes ("Sports" and "Not Sports"). With the training set, we can train a Naive Bayes classifier which we can use to automatically categorize a new sentence.

Important Use-Case: **Recognizing Spam-Mail**

Consider the problem of classifying documents by their content, for example into [spam](#) and non-spam [e-mails](#). See also: *Example2 of the ML Definition* by Tom Mitchel (Chapter ML1).

SVL Classification (5/8): Bayes L. for Text Classification

So here comes the *Naive* part: we assume that every word in a sentence is **independent** of the other ones. This means that we're no longer looking at entire sentences, but rather at individual words. So for our purposes, "this was a fun party" is the same as "this party was fun" and "party fun was this".

We write this as:

$$P(a \text{ very close game}) = P(a) \times P(\text{very}) \times P(\text{close}) \times P(\text{game})$$

This assumption is very strong but super useful. It's what makes this model work well with little data or data that may be mislabeled. The next step is just applying this to what we had before:

$$P(a \text{ very close game} | \text{Sports}) = P(a | \text{Sports}) \times P(\text{very} | \text{Sports}) \times P(\text{close} | \text{Sports}) \times P(\text{game} | \text{Sports})$$

Calculating Probabilities:

The final step is just to calculate every probability and see which one turns out to be larger. Calculating a probability is just counting in our training data. First, we calculate the *a priori* probability of each tag: for a given sentence in our training data, the probability that it is *Sports* = $P(\text{Sports}) = 3/5$. Then, $P(\text{Not Sports}) = 2/5$. That's easy enough.

Then, calculating $P(\text{game} | \text{Sports})$ means counting how many times the word "game" appears in *Sports* texts (2) divided by the total number of words in *sports* (11). Therefore, $P(\text{game} | \text{Sports}) = 2/11$.

However, we run into a problem here: "close" doesn't appear in any *Sports* text! That means that $P(\text{close} | \text{Sports}) = 0$. This is rather inconvenient since we are going to be multiplying it with the other probabilities, so we'll end up with zero.

One can also write a **Jupyter Notebook** to automat the calculations:

To find the total number of times a word appears in a class, we can use `CountVectorizer` from `sklearn`. `CountVectorizer` gives Term-Document Matrix (TDM) for each class. A term-document matrix (TDM) consists of a list of word frequencies appearing in a set of documents. Next, let's compute the Term-Document Matrix (TDM) for 'Sports' class.

We follow the description of the above link (page before):

```

1 from sklearn.feature_extraction.text import CountVectorizer
2
3 stmt_docs = [row['sent'] for index,row in training_data.iterrows() if row['class'] == 'stmt']
4
5 vec_s = CountVectorizer()
6 X_s = vec_s.fit_transform(stmt_docs)
7 tdm_s = pd.DataFrame(X_s.toarray(), columns=vec_s.get_feature_names())
8
9 tdm_s
    
```

See also: <https://towardsdatascience.com/introduction-to-na%C3%AFve-bayes-classifier-fa59e3e24aaf>

SVL Classification (6/8): Bayes L. for Text Classification

How do we do it? By using something called [Laplace smoothing](#): we add 1 to every count so it's never zero. To balance this, we add the number of possible words to the divisor, so the division will never be greater than 1. In our case, the possible words are (see notespage):

'a' 'great' 'very' 'over' 'it' 'but' 'game' 'election' 'clean' 'close' 'the' 'was' 'forgettable' 'match'.

Since the number of possible words is 14 (I counted them!), applying smoothing we get that $P(\text{game}|\text{Sports})=(2+1)/(11+14)=3/25$. The full results are:

P(Sports)=3/5
P(Not Sports)=2/5
Anzahl (Words|Sports)=11
(Words|Not Sports)=9
(possible words)=14 <-- see notes

Word	P(word Sports)	P(word Not Sports)
a	3/25	2/23
very	2/25	1/23
close	1/25	2/23
game	3/25	1/23

=> $P(a|\text{Sports}) \cdot P(\text{very}|\text{Sports}) \cdot P(\text{close}|\text{Sports}) \cdot P(\text{game}|\text{Sports}) \cdot P(\text{Sports}) = 3/25 \cdot 2/25 \cdot 1/25 \cdot 3/25 \cdot 3/5 = 18/25 \cdot 1/25 \cdot 3/5 = 54/125 \cdot 1/25 = 2,7648e-5$

Analog: $P(a|\text{Not Sports}) \cdot P(\text{very}|\text{Not Sports}) \cdot P(\text{close}|\text{Not Sports}) \cdot P(\text{game}|\text{Not Sports}) \cdot P(\text{Not Sports}) = 2/23 \cdot 1/23 \cdot 2/23 \cdot 1/23 \cdot 2/5 = 8/115 \cdot 1/23 = 0,57176e-5$

=> the Sentence 'a very close game' is tagged as Sports.

<https://www.youtube.com/watch?v=exHwuy9kVcg>

From Wikipedia, the free encyclopedia

In statistics, **additive smoothing**, also called **Laplace smoothing**^[1] (not to be confused with **Laplacian smoothing** as used in **image processing**), or **Lidstone smoothing**, is a technique used to **smooth categorical data**. Given an observation $\mathbf{x} = (x_1, x_2, \dots, x_d)$ from a **multinomial distribution** with N trials, a "smoothed" version of the data gives the estimator:

$$\hat{\theta}_i = \frac{x_i + \alpha}{N + \alpha d} \quad (i = 1, \dots, d),$$

where the "pseudocount" $\alpha > 0$ is a smoothing **parameter**. $\alpha = 0$ corresponds to no smoothing. (This parameter is explained in § **Pseudocount** below.) Additive smoothing is a type of **shrinkage estimator**, as the resulting estimate will be between the **empirical probability (relative frequency)** x_i/N , and the **uniform probability** $1/d$. Invoking Laplace's **rule of succession**, some authors have argued^[citation needed] that α should be 1 (in which case the term **add-one smoothing**^{[2][3]} is also used)^[further explanation needed], though in practice a smaller value is typically chosen.

To calculate the number of positive words, see the following table:

No.	word	#word>1	sum
1	a	3	3
2	great		4
3	very		5
4	over		6
5	it		7
6	but		8
7	game	2	10
8	election	2	12
9	clean	2	14
10	close		15
11	the		16
12	was	2	18
13	forgettable		19
14	match		20

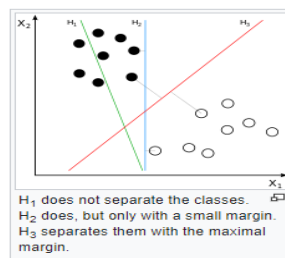
SVL Classification (7/8): Support Vector Machines (SVM)

In [machine learning](#), support-vector machines (SVMs, also support-vector networks^[1]) are [supervised learning](#) models with associated learning [algorithms](#) that analyze data used for [classification](#) and [regression analysis](#). The Support Vector Machine (SVM) algorithm is a popular machine learning tool that offers solutions for both classification and regression

Motivation [\[edit\]](#)

Classifying data is a common task in [machine learning](#).

Suppose some given data points each belong to one of two classes, and the goal is to decide which class a *new data point* will be in. In the case of support-vector machines, a data point is viewed as a p -dimensional vector (a list of p numbers), and we want to know whether we can separate such points with a $(p - 1)$ -dimensional [hyperplane](#). This is called a [linear classifier](#). There are many hyperplanes that might classify the data. One reasonable choice as the best hyperplane is the one that represents the largest separation, or [margin](#), between the two classes. So we choose the hyperplane so that the distance from it to the nearest data point on each side is maximized. If such a hyperplane exists, it is known as the [maximum-margin hyperplane](#) and the linear classifier it defines is known as a [maximum-margin classifier](#), or equivalently, the [perceptron of optimal stability](#).^[citation needed]



H_1 does not separate the classes.
 H_2 does, but only with a small margin.
 H_3 separates them with the maximal margin.

See: Chapter **ML8**

Page 61

In [machine learning](#), **support-vector machines (SVMs, also support-vector networks^[1])** are [supervised learning](#) models with associated learning [algorithms](#) that analyze data used for [classification](#) and [regression analysis](#). The Support Vector Machine (SVM) algorithm is a popular machine learning tool that offers solutions for both classification and regression problems. Developed at AT&T Bell Laboratories by Vapnik with colleagues (Boser et al., 1992, Guyon et al., 1993, Vapnik et al., 1997), it presents one of the most robust prediction methods, based on the statistical learning framework or VC theory proposed by Vapnik and Chervonekis (1974) and Vapnik (1982, 1995). Given a set of training examples, each marked as belonging to one or the other of two categories, an SVM training algorithm builds a model that assigns new examples to one category or the other, making it a non-[probabilistic binary linear classifier](#) (although methods such as [Platt scaling](#) exist to use SVM in a probabilistic classification setting). An SVM model is a representation of the examples as points in space, mapped so that the examples of the separate categories are divided by a clear gap that is as wide as possible. New examples are then mapped into that same space and predicted to belong to a category based on the side of the gap on which they fall.

In addition to performing [linear classification](#), SVMs can efficiently perform a non-linear classification using what is called the [kernel trick](#), implicitly mapping their inputs into high-dimensional feature spaces....

More details: https://en.wikipedia.org/wiki/Support_vector_machine

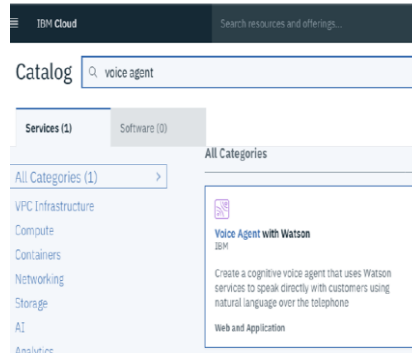
SVL Classification (8/8): Natural Language Processing (NLP)

Computer Linguistics (also called Natural Language Processing – NLP) one learned from many training examples to understand Human Speech and transform it to text or vice versa. For example today speech assistants (ex. Alexa etc.) are very common.

For example:

- Build a “Voice Assistant” by using the Text / Speech Recognition services of the IBM Cloud : platform:
 1. Watson Voice Agent
 2. Watson Assistant
 3. Speech to Text
 4. Text to Speech

See also Homework ML3.3 for more details about building the services.



Log in into IBM Cloud and follow the tutorial descriptions (see links):

Create a “Voice Agent” by running the following steps:

- Set up the requires IBM Cloud Services
- Configure the TWILIO Account
- Configure the Voice Agent on the IBM Cloud and import Skill by uploading either *skill-banking-balance-enquiry.json* or *skill-pizza-order-book-table.json*

Supervised Learning (SVL) – List of Regression Concepts

Models

Linear regression · Simple regression · Polynomial regression · General linear model · Generalized linear model · Discrete choice · Logistic regression · Multinomial logit · Mixed logit · Probit · Multinomial probit · Ordered logit · Ordered probit · Poisson · Multilevel model · Fixed effects · Random effects · Mixed model · Nonlinear regression · Nonparametric · Semiparametric · Robust · Quantile · Isotonic · Principal components · Least angle · Local · Segmented · Errors-in-variables

Estimation

Least squares · Linear · Non-linear · Ordinary · Weighted · Generalized · Partial · Total · Non-negative · Ridge regression · Regularized · Least absolute deviations · Iteratively reweighted · Bayesian · Bayesian multivariate

Background

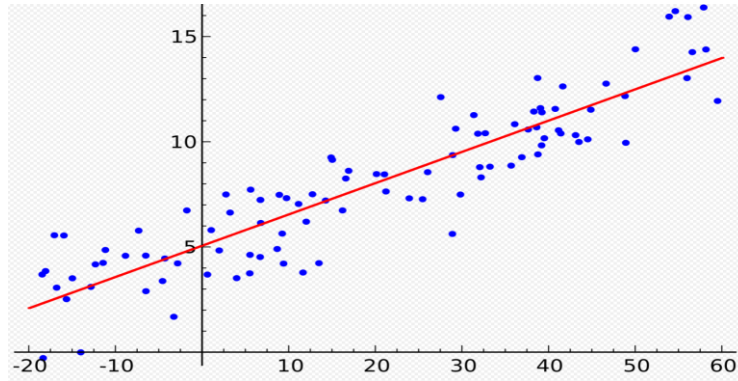
Regression model validation · Mean and predicted response · Errors and residuals · Goodness of fit · Studentized residual · Gauss–Markov theorem

Link: https://en.wikipedia.org/wiki/Regression_analysis

In statistical modeling, regression analysis is a set of statistical processes for estimating the relationships among variables. It includes many techniques for modeling and analyzing several variables, when the focus is on the relationship between a dependent variable and one or more independent variables (or 'predictors'). More specifically, regression analysis helps one understand how the typical value of the dependent variable (or 'criterion variable') changes when any one of the independent variables is varied, while the other independent variables are held fixed.

Supervised Learning (SVL) – Simple Linear Regression Example

Given a [data](#) set $\{y_1, x_1, \dots, x_p\}$ of n [statistical units](#), a linear regression model assumes that the relationship between the dependent variable y and the [p-vector](#) of regressors x is [linear](#). This relationship is modeled through a *disturbance term* or *error variable* ε — an unobserved [random variable](#) that adds "noise" to the linear relationship between the dependent variable and regressors.



See more details to this in chapter ML5.

See chapter ML5

Definition of Unsupervised Learning (USVL)

What do we mean by **unsupervised learning**?

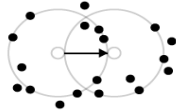
By unsupervised, it means that we're not trying to predict a variable; instead, we want to discover hidden patterns within our data that will let us to identify groups, or clusters, within that data.



Previously, we've discussed regression and classification algorithms. These are types of supervised learning; that is, we have a specific variable we are looking to predict based on other, independent variables. Here we'll discuss clustering models, which is one of the simplest types of unsupervised learning.

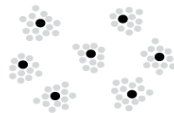
First and most common USVL Example - Clustering (1/4)

Clustering is often used in marketing in order to group users according to multiple characteristics, such as location, purchasing behavior, age, and gender. It can also be used in scientific research; for example, to find population clusters within DNA data.



One simple clustering algorithm is DBSCAN. In DBSCAN, you select a distance for your radius, and you select a point within your dataset -- then, all other data points within the radius's distance from your initial point are added to the cluster. You then repeat the process for each new point added to the cluster, and you repeat until no new points are within the radii of the most recently added points. Then, you choose another point within the dataset and build another cluster using the same approach.

DBSCAN is intuitive, but its effectiveness and output rely heavily on what you choose for a radius, and there are certain types of distributions that it won't react well to.



The most widespread clustering algorithm is called k-means clustering. In k-means clustering, we pre-define the number of clusters we want to create -- the number we choose is the k, and it is always a positive integer.

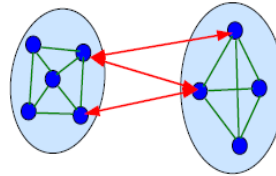
To run k-means clustering, we begin by randomly placing k starting points within our dataset. These points are called centroids, and they become the prototypes for our k clusters. We create these initial clusters by assigning every point within the dataset to its nearest centroid. Then, with these initial clusters created, we calculate the midpoint of each of them, and we move each centroid to the midpoint of its respective cluster. After that, since the centroids have moved, we can then reassign each data point to

a centroid, create an updated set of clusters, and calculate updated midpoints. **We continue iterating for a predetermined number of times -- 300 is pretty standard. By the time we get to the end, the centroids should move minimally, if at all.**

First USVL Example – Clustering Concept (2/4)

Repetition –
Data Mining

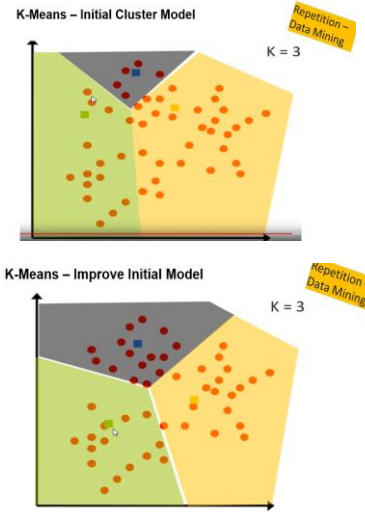
- Given:
 - a set of examples
 - in some description language (e.g., attribute-value)
 - no labels (→ unsupervised)
- Find:
 - a grouping of the examples into meaningful *clusters*
 - so that we have a high
 - **intra-class similarity:**
similarity between objects
in same cluster
 - **inter-class dissimilarity:**
dissimilarity between objects
in different clusters



First USVL Example: K-Means-Clustering Algorithm (3/4)

K-Means Learning Algorithm:

1. Define an initial (random) solution as vectors of means
 $\mathbf{m}(t=0) = [m_1, m_2, \dots, m_K]^T$
2. Classify each input data according to $\mathbf{m}(t)$
3. Use the classification obtained in step 2 to recompute the vectors of means $\mathbf{m}(t+1)$
4. Update $t = t+1$
5. If $\|\mathbf{m}(t) - \mathbf{m}(t-1)\| < \zeta$ (convergence)
 Use $\mathbf{m}(t)$ as the solution
 Else
 Go back to step 2



K-means Clustering is one of the simplest and popular unsupervised machine learning algorithms. Typically, unsupervised algorithms make inferences from datasets using only input vectors without referring to known, or labelled, outcomes. The objective of K-means is simple: group similar data points together and discover underlying patterns. To achieve this objective, K-means looks for a fixed number (k) of clusters in a dataset. A cluster refers to a collection of data points aggregated together because of certain similarities. You'll define a target number k, which refers to the number of centroids you need in the dataset. A centroid is the imaginary or real location representing the center of the cluster. Every data point is allocated to each of the clusters through reducing the in-cluster sum of squares.

In other words, the **K-means algorithm identifies k number of centroids**, and then allocates every data point to the nearest cluster, while keeping the centroids as small as possible. The 'means' in the K-means refers to averaging of the data; that is, finding the centroid. More details:

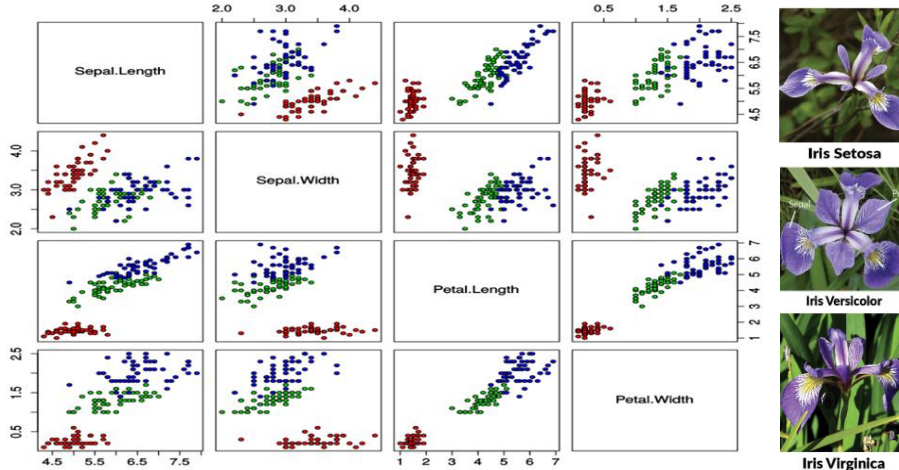
K-means algorithm: Let's see the steps on how the K-means machine learning algorithm works using the Python. We'll use the Scikit-learn library and some random data to illustrate a K-means clusteringSee more details under:

<https://towardsdatascience.com/understanding-k-means-clustering-in-machine-learning-6a6e67336aa1>

Repetition –
Data Mining

Cluster Ex. & K-Means Clusters of IRIS Dataset (4/4)

Iris Data (red=setosa,green=versicolor,blue=virginica)



https://github.com/bhattbhavesh91/k_means_iris_dataset/blob/master/K_in_K_means_Clustering.ipynb

Status: 1. Dezember 2020

Page 69

How the K-means algorithm works: To process the learning data, the K-means algorithm in data mining starts with a first group of randomly selected centroids, which are used as the beginning points for every cluster, and then performs iterative (repetitive) calculations to optimize the positions of the centroids. It halts creating and optimizing clusters when either: The centroids have stabilized — there is no change in their values because the clustering has been successful. The defined number of iterations has been achieved.

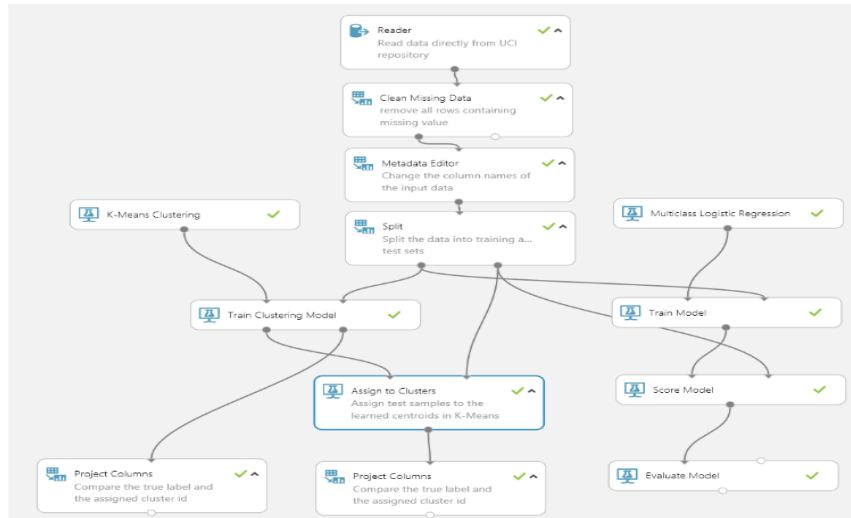
K-means Clusters of IRIS Dataset: The Iris dataset contains the data for 50 flowers from each of the 3 species - Setosa, Versicolor and Virginica.

<http://www.lac.inpe.br/~rafael.santos/Docs/CAP394/WholeStory-Iris.html>

The data gives the measurements in centimeters of the variables sepal length and width and petal length and width for each of the flowers. Goal of the study is to perform exploratory analysis on the data and build a K-means clustering model to cluster them into groups. Here we have assumed we do not have the species column to form clusters and then used it to check our model performance. Since we are not using the species column we have an unsupervised learning method. Develop a Python program by using the Scikit-learn library can be seen under:

https://github.com/bhattbhavesh91/k_means_iris_dataset/blob/master/K_in_K_means_Clustering.ipynb

UseCase MS Azure ML: k-Means Algorithm on UCI Iris data



Status: 1. Dezember 2020

Page 70

YouTube videos:

<https://www.youtube.com/watch?v=Cifl6cuEwMw>

<https://www.youtube.com/watch?v=nnp77iFxrE>

<https://www.youtube.com/watch?v=pH3hQc585WQ>

Use Case Description:

<https://gallery.azure.ai/Experiment/a7299de725a141388f373e9d74ef2f86>

This sample demonstrates how to perform clustering using k-means algorithm on the UCI Iris data set. Also we apply multi-class Logistic regression to perform multi-class classification and compare its performance with k-means clustering.

Clustering: Group Iris Data

This sample demonstrates how to perform clustering using the k-means algorithm on the UCI Iris data set. In this experiment, we perform k-means clustering using all the features in the dataset, and then compare the clustering results with the true class label for all samples. We also use the **Multiclass Logistic Regression** module to perform multiclass classification and compare its performance with that of k-means clustering.

Data

We used the Iris data set, a well-known benchmark dataset for multiclass classification from the [UCI repository](#). This dataset has 150 samples with 4 features and 1 label (the last column). All features are numeric except that the label, which is a string.

2. Ex. of USVL: Association Learning – General Form (1/4)

- General Form:

$$A_1, A_2, \dots, A_n \rightarrow B_1, B_2, \dots, B_m$$

- Interpretation:

- When items A_i appear, items B_j also appear with a certain probability

- Examples:

- Bread, Cheese** → **RedWine**.
Customers that buy bread and cheese, also tend to buy red wine.
- MachineLearning** → **WebMining, MLPraktikum**.
Students that take 'Machine Learning' also take 'Web Mining' and the 'Machine Learning Praktikum'

Repetition –
Data Mining

What is Association Learning?

Association learning is a rule based [machine learning](#) and data mining technique that finds important relations between variables or features in a data set. Unlike conventional association algorithms measuring degrees of similarity, association rule learning identifies hidden correlations in databases by applying some measure of interestingness to generate an association rule for new searches.

Practical Uses of Association Learning

- **Basket data analysis** – Whether planning product placement in a storefront, running a marketing campaign or designing a business catalog, association mining is a useful tool to take the guesswork out of what your customers are looking for.
- **Web usage mining and intrusion detection** – Finding these hidden correlations is a powerful predictive tool to discover brand new security threats and network performance issues that haven't been analyzed first by a human.
- **Bioinformatics** – From biology to engineering and everything in between, association mining is one of the go-to foundational tools for spotting overlooked and potentially useful techniques.

2. Ex. USVL: Association Rules – Repeat Measure Def's (2/4)

Rule: $X \Rightarrow Y$

$$\text{Support} = \frac{\text{freq}(X, Y)}{N}$$

$$\text{Confidence} = \frac{\text{freq}(X, Y)}{\text{freq}(X)}$$

$$\text{Lift} = \frac{\text{Support}}{\text{Supp}(X) \times \text{Supp}(Y)}$$

Repetition –
Data Mining

Properties:

- $\text{Sup}(X \Rightarrow Y) = \text{Sup}(Y \Rightarrow X)$
- $\text{Lift}(X \Rightarrow Y) = \text{Lift}(Y \Rightarrow X)$

Question:

- How many rules have you to consider in this example?
- Prove the answer: **You have to consider 80 rules (40 for Support and Lift).**



Rule	Support	Confidence	Lift
$A \Rightarrow D$	2/5	2/3	10/9
$C \Rightarrow A$	2/5	2/4	5/6
$A \Rightarrow C$	2/5	2/3	5/6
$B \& C \Rightarrow D$	1/5	1/3	5/9

N=5

Support ($A \Rightarrow D$):= $\text{freq}(A, D) / 5 = 2/5$

Support ($C \Rightarrow A$):= $\text{freq}(C, A) / 5 = 2/5$

Support ($A \Rightarrow C$):= $\text{freq}(A, C) / 5 = 2/5$

Support ($B \& C \Rightarrow D$):= $\text{freq}(B \& C, D) / 5 = 1/5$

Confidence($A \Rightarrow D$):= $\text{freq}(A, D) / \text{freq}(A) = (2/5) / (3/5) = 2/3$

Confidence($C \Rightarrow A$):= $\text{freq}(C, A) / \text{freq}(C) = (2/5) / (4/5) = 2/4 = 1/2$

Confidence($A \Rightarrow C$):= $\text{freq}(A, C) / \text{freq}(A) = (2/5) / (3/5) = 2/3$

Confidence($B \& C \Rightarrow D$):= $\text{freq}(B \& C, D) / \text{freq}(B \& C) = (1/5) / (3/5) = 1/3$

Lift($A \Rightarrow D$):= $\text{Sup}(A \Rightarrow D) / (\text{Sup}(A) * \text{Sup}(D)) = (2/5) / (3/5 * 3/5) = (2/5) / (9/25) = 2 / (9/5) = 10/9$

Lift($C \Rightarrow A$):= $\text{Sup}(C \Rightarrow A) / (\text{Sup}(C) * \text{Sup}(A)) = (2/5) / (4/5 * 3/5) = (2/5) / (12/25) = 2 / (12/5) = 10/12 = 5/6$

Lift($A \Rightarrow C$):= $\text{Sup}(A \Rightarrow C) / (\text{Sup}(A) * \text{Sup}(C)) = (2/5) / (3/5 * 4/5) = (2/5) / (12/25) = 2 / (12/5) = 10/12 = 5/6$

Lift($B \& C \Rightarrow D$):= $\text{Sup}(B \& C \Rightarrow D) / (\text{Sup}(B \& C) * \text{Sup}(D)) = (1/5) / (3/5 * 3/5) = (1/5) / (9/25) = 1 / (9/5) = 5/9$

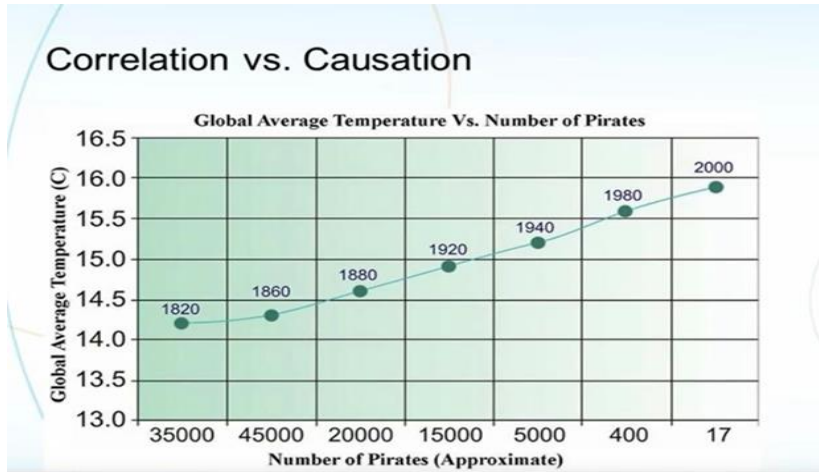
2. Ex. USVL: Use-Case: Predictive Market Basket Analysis (3/4)

Recommender Systems



The screenshot shows the Amazon.com interface for a product page. At the top, there's the Amazon logo and navigation links like 'Hello, Sign in to get personalized recommendations. New customer? Start here.', 'Your Amazon.com', 'Today's Deals', 'Gifts & Wish Lists', and 'Gift Cards'. Below this is a search bar with 'Health & Personal Care' entered. The main product is 'Adult Reusable Cotton/Poly Snap Diaper - Large - Fits 32" - 46" - Each' by Comfort Concepts. It has a price of \$15.05 and is marked as 'In stock'. Below the product image, there's a 'Frequently Bought Together' section showing the diaper and a game, 'Call of Duty 4: Modern Warfare Game of the Year Edition by Activision', with a combined price of \$40.11. Buttons for 'Add both to Cart' and 'Add both to Wish List' are visible.

2. Ex. USVL: Check Associations for Business Context (4/4)



Status: 1. Dezember 2020

Page: 74

The diagram (example) suggests that the temperature of the ocean depends on the number of pirates on the oceans. One might come to the conclusion: "Let's increase the number of pirates and the temperature of the oceans will drop again".

It is therefore always to check by business experts, which metrics are only in a correlation and which have a causal business context.

Exercises to Lesson ML3

Homework H3.1 – “Calculate Value Difference Metric”

Calculate $d := \text{Value Difference Metric (VDM)}$ for the fields “Refund” and “Marital Status”. Remember the following formula and see also details of VDM in internet (1 person, 10 minutes):

$$d_A(v_1, v_2) = \sum_c \left| \frac{n_{1,c}}{n_1} - \frac{n_{2,c}}{n_2} \right|^k$$

k is a user-settable parameter (e.g., $k=2$)

$n_{1,c}$ = die Häufigkeit von Attributwert 1 in Klasse c
 n_1 = die Häufigkeit von Attributwert 1 über alle Klassen
 Da keine numerischen Werte vorhanden sind, setze $k=1$

With data table:

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

Repetition –
Data Mining

Hint: $d(\text{single, married})$, $d(\text{single, divorced})$, $d(\text{married, divorced})$; $d(\text{refund=yes, refund=no})$

$d(\text{single, married})$, $d(\text{single, divorced})$, $d(\text{married, divorced})$;

Class	Marital Status		
	Single	Married	Divorced
Yes	2	0	1
No	2	4	1

$d(\text{refund=yes; refund=no})$

Complete solutions are found in “Exercises2Lecture.pdf”

Exercises to Lesson ML3

Homework H3.2 – “Bayes Learning for Text Classification”

1 Person: Review the example about Bayes Learning in this lesson. Use the same training data as in the lesson together with the new lagged text. Run the Bayes -Text Classification calculation for the sentence “*Hermann plays a TT match*” and tag this sentence.

No.	Training-Text	Label
1	“A great game”	Sports
2	“The election was over”	Not Sports
2	“Very clean match”	Sports
4	“A clean but forgettable game”	Sports
5	“It was a close election”	Not Sports
6	“A very close game”	Sports
Target-Text		
new	“Hermann plays a TT match”	?????????

Additional Question: What will happen if we change the target to “*Hermann plays a very clean game*”

Optional*(1 P): Define an algorithm in Python (use Jupyter Notebook) to automate the calculations.

Use description under: <https://medium.com/analytics-vidhya/naive-bayes-classifier-for-text-classification-556fabaf252b#:~:text=The%20Naive%20Bayes%20classifier%20is,time%20and%20less%20training%20data.>

Exercises to Lesson ML3

Homework H3.3 – Create in IBM Cloud two services “Voice Agent” and “Watson Assistant Search Skill” with IBM Watson Services

Homework for 2 person: Log in into IBM Cloud and follow the tutorial descriptions (see links):

1. “Voice Agent” (1 person):

- a. Set up the requires IBM Cloud Services
- b. Configure the TWILIO Account
- c. Configure the Voice Agent on the IBM Cloud and import Skill by uploading either
 - ▶ skill-banking-balance-enquiry.json or
 - ▶ skill-pizza-order-book-table.json

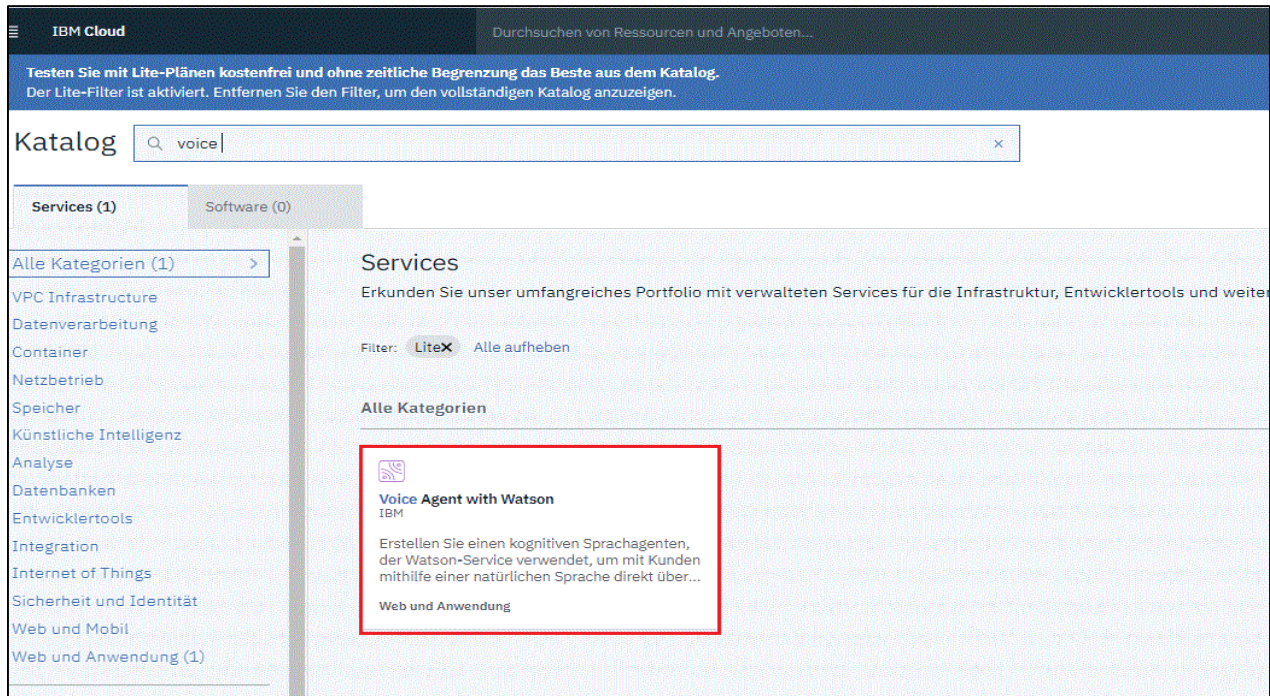
See tutorial: <https://github.com/FelixAugenstein/digital-tech-tutorial-voice-agent>

2. “Assistant Search Skill” (1 person):

- a. Configuring Watson Assistant & Discovery Service on the IBM Cloud
- b. Configuring Watson Assistant & Search Skill on the IBM Cloud
- c. Deploy the Assistant with Search Skill

See tutorial: <https://github.com/FelixAugenstein/digital-tech-tutorial-watson-assistant-search-skill>

Remark: You can integrate the two skills, such that when the dialog skill has no answer you show the search results. The reading of texts from the search results of the search skill is unfortunately not (yet) possible. Watson can only display the search result with title/description etc. as on Google. The tutorial in the cloud docs on the same topic is also helpful: <https://cloud.ibm.com/docs/assistant?topic=assistant-skill-search-add>



A further example of a Finance Chatbot is **BOTTO** which runs von MS Azure from Fiducia AG (Karlsruhe). See a presentation from Fiducia and Adesso (Dortmund):

<https://www.adesso.de/adesso/adesso-de/branchen/banken-finanzdienstleister/sonderthemen/forum-banken/presentation-chatbot-botto-g-weber-fiducia-gad-it-ag.pdf>

Exercises to Lesson ML3

Repetition –
Data Mining

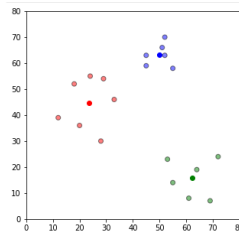
Homework H3.4 – Create a K-Means Clustering in Python”

Homework for 2 Persons: Create a python algorithm (in Jupyter Notebook) which clusters the following points:

```
df = pd.DataFrame({
    'x': [12, 20, 28, 18, 29, 33, 24, 45, 45, 52, 51, 52, 55, 53, 55, 61, 64, 69, 72],
    'y': [39, 36, 30, 52, 54, 46, 55, 59, 63, 70, 66, 63, 58, 23, 14, 8, 19, 7, 24]
})
```

Following the description of: <https://benalexkeen.com/k-means-clustering-in-python/> to come to 3 clear clusters with 3 means at the center of these clusters:

We'll do this manually first (1 person), then show how it's done using scikit-learn (1 person)



Repeat *K-Means* clustering (DM lesson or internet). Describe and explain the 4 necessary steps to reach the final cluster

1. The centroids.
2. Assigning the first clusters.
3. Calculating the center of gravity and interacting.
4. The final clusters.

Exercises to Lesson ML3

Repetition –
Data Mining

Homework H3.5 – “Repeat + Calculate of *Measures for Association*”

Remember and give explanations of the **Measures for Association**: *support*, *confidence* and *lift* (1 person, 10 min):

Calculate these measures for following 8 item sets of a shopping basket (1 person, 10 minutes):

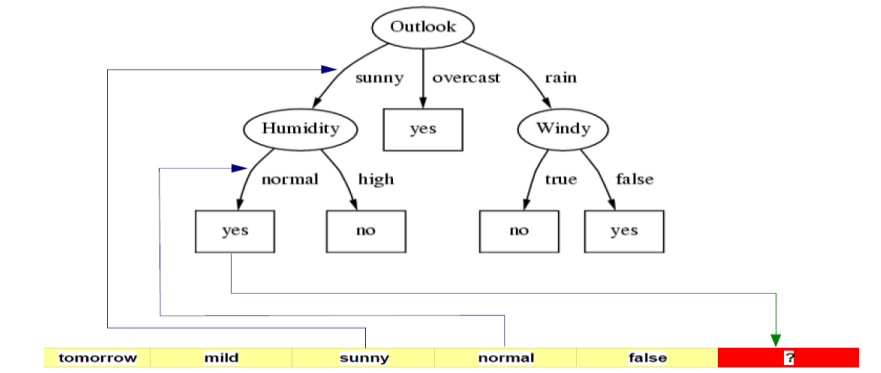
{ Milch, Limonade, Bier }; { Milch, Apfelsaft, Bier }; { Milch, Apfelsaft, Orangensaft };
{ Milch, Bier, Orangensaft, Apfelsaft }; { Milch, Bier }; { Limonade, Bier, Orangensaft };
{ Orangensaft }; { Bier, Apfelsaft }

1. What is the support of the item set { Bier, Orangensaft }?
2. What is the confidence of { Bier } \rightarrow { Milch } ?
3. Which association rules have support and confidence of at least 50%?

Complete solutions are found in “Exercises2Lecture.pdf”

1. ML1: Introduction to Machine Learning (ML)
2. ML2: Concept Learning: VSpaces & Cand. Elim. Algo.
3. ML3: Supervised and Unsupervised Learning
4. **ML4: Decision Tree Learning**
5. ML5: simple Linear Regression (sLR) & multiple Linear Regression (mLR)
6. ML6: Neural Networks: Convolutional NN
7. ML7: Neural Network: BackPropagation Algorithm
8. ML8: Support Vector Machines (SVM)

ML4 – Decision Tree Learning



Status: 1. Dezember 2020

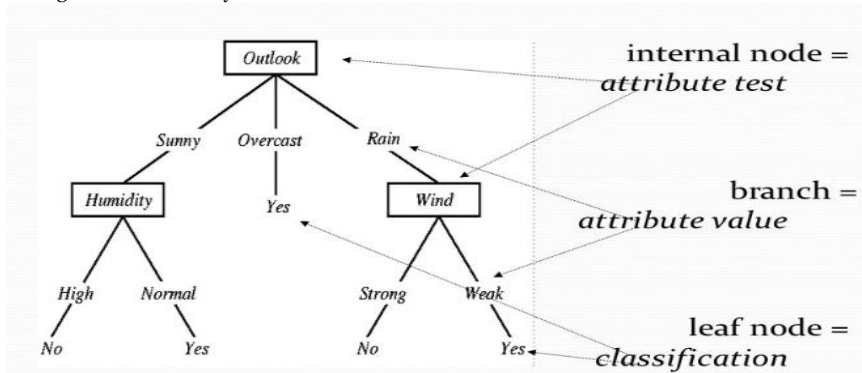
Page 80

Decision tree learning uses a **decision tree** as a **predictive model** which maps observations about an item to conclusions about the item's target value. It is one of the predictive modelling approaches used in **statistics**, **data mining** and **machine learning**. Tree models where the target variable can take a finite set of values are called **classification trees**. In these tree structures, leaves represent class labels and branches represent conjunctions of features that lead to those class labels. Decision trees where the target variable can take continuous values (typically real numbers) are called **regression trees**. In decision analysis, a decision tree can be used to visually and explicitly represent decisions and decision making. In data mining, a decision tree describes data but not decisions; rather the resulting classification tree can be an input for decision making.

Definition of Decision Trees

A decision tree is a tree where each node represents a feature(attribute), each link(branch) represents a decision(rule) and each leaf represents an outcome (categorical or continues value).

The whole idea is to create a tree like this *play tennis* example or the entire data and process a single outcome at every leaf .



Usage of Decision Trees

- Decision tree is one of the **most popular** machine learning algorithms used all along.
- Decision trees are used for both **classification and regression** problems, in this this lesson we will talk about classification.
- We have couple of other algorithms in ML, so why do we have to choose Decision Trees? Here are a **few arguments** why we use Decision Trees:
 - ✓ Decision tress often mimic the human level thinking so it's so **simple to understand** the data and make some good interpretations.
 - ✓ Decision trees actually make you **see the logic for the data** to interpret (not like black box algorithms like SVM, NN, etc..)

Build a Decision Tree with Different Methods

There are couple of algorithms there to build a decision tree, the most important are:

1. **ID₃** (Iterative Dichotomiser 3) → uses **Entropy function** and **Information gain** as metrics.
2. **CART** (Classification and Regression Trees) → uses **Gini Index (Classification)** as metric.

We will build a two decision trees with this two methods using the *Playing Tennis Game* data set (see next foil).

- **Information entropy** is the average rate at which information is produced by a stochastic source of data.
- **Information Gain** is the change in information entropy H from a prior state to a state that takes some information as given:
 $IG(T, a) = H(T) - H(T|a)$,
 where $H(T|a)$ is the conditional entropy of T given the value of attribute a .

GINI Index

$$Gini = \sum_{i \neq j} p(i)p(j)$$

i and j are levels of the target variable

- Consider **Gini Index with Binary Target variables** (Yes and No), then we have 4 possible combinations (Probabilities) with sum =1:
 $P(\text{Target}=1).P(\text{Target}=1) + P(\text{Target}=1).P(\text{Target}=0) + P(\text{Target}=0).P(\text{Target}=1) + P(\text{Target}=0).P(\text{Target}=0) = 1$
 $P(\text{Target}=1).P(\text{Target}=0) + P(\text{Target}=0).P(\text{Target}=1) = 1 - P^2(\text{Target}=0) - P^2(\text{Target}=1)$.
 Therefore **Gini Index** = $1 - P^2(\text{Target}=0) - P^2(\text{Target}=1)$.

What are the differences between ID3, C4.5 and CART?

- **ID3**, or Iterative Dichotomizer, was the first of three Decision Tree implementations developed by Ross Quinlan (Quinlan, J. R. 1986. Induction of Decision Trees. Mach. Learn. 1, 1 (Mar. 1986), 81-106.)
- **CART**, or *Classification And Regression Trees* is often used as a generic acronym for the term Decision Tree, though it apparently has a more specific meaning. In sum, the CART implementation is very similar to C4.5; the one notable difference is that CART constructs the tree based on a numerical splitting criterion recursively applied to the data, whereas C4.5 includes the intermediate step of constructing rule sets.
- **C4.5**, Quinlan's next iteration. The new features (versus ID3) are: (i) accepts both continuous and discrete features; (ii) handles incomplete data points; (iii) solves over-fitting problem by (very clever) bottom-up technique usually known as "pruning"; and (iv) different weights can be applied the features that comprise the training data. Of these, the first *three* are very important--and (i) would suggest that any DT implementation you choose have all three. The fourth (differential weighting) is much less important.

What is Gini index (source wikipedia): In economics, the **Gini coefficient**, sometimes called the **Gini index** or **Gini ratio**, is a measure of statistical dispersion intended to represent the income or wealth distribution of a nation's residents, and is the most commonly used measurement of inequality. It was developed by the Italian statistician and sociologist Corrado Gini (1864-1965) and published in his 1912 paper "*Variability and Mutability*".^{[1][2]}

„Playing Tennis Game“ data set

Let's just take a famous dataset in the machine learning world which is weather dataset (playing game Y or N based on weather condition). For smarter work, count the frequencies of all feature values in dependency of YES/NO.

outlook	temp.	humidity	windy	play
sunny	hot	high	false	no
sunny	hot	high	true	no
overcast	hot	high	false	yes
rainy	mild	high	false	yes
rainy	cool	normal	false	yes
rainy	cool	normal	true	no
overcast	cool	normal	true	yes
sunny	mild	high	false	no
sunny	cool	normal	false	yes
rainy	mild	normal	false	yes
sunny	mild	normal	true	yes
overcast	mild	high	true	yes
overcast	hot	normal	false	yes
rainy	mild	high	true	no

Definition of "Frequency Table" $X \rightarrow y$

feature	feature value 1 (v1)	feature value 2 (v2)	feature value 3 (v3)	sum
YES	#v1&YES	#v2&YES	#v3&YES	#YES
NO	#v1&NO	#v2&NO	#v3&NO	#NO
sum	#value 1	#value 2	#value 3	#counts

Example: **feature** = outlook

outlook	overcast	sunny	rainy	sum
YES	4	2	3	9
NO	0	3	2	5
sum	4	5	5	14

We have four X values (also called "features")= {outlook, temp, humidity, windy} being categorical and one y value ("target")= {play Y or N} also being categorical. So we need to learn the mapping (what machine learning always does) between X and y.

To better work with the data set we have first to summaries the number of results (yes/no) in the target variable y="play" for each value of all features (attributes) X={outlook; temp.; humidity; windy}:

humidity	high	normal	sum
YES	3	6	9
NO	4	1	5
sum	7	7	14

temperature	hot	mild	cool	sum
YES	2	4	3	9
NO	2	2	1	5
sum	4	6	4	14

windy	FALSE	TRUE	sum
YES	6	3	9
NO	2	3	5
sum	8	6	14

Build the Tree using the ID₃ algorithm (1/7)

To create a tree, we need to have a root node first and we know that nodes are features/attributes (outlook, temp, humidity and windy),
So, which one do we need to pick first??

Answer: determine the attribute that best classifies the training data; use this attribute at the root of the tree. Repeat this process at for each branch.

This means we are performing top-down, greedy search through the space of possible decision trees.

Okay, so how do we choose the best attribute?

Answer: use the attribute with the highest **information gain** in ID₃

*In order to define information gain precisely, we begin by defining two measures commonly used in information theory, called **entropy** and **information gain***

Lets first start with the ID₃ algorithm:

1. Definition of the Root value: defined by the highest Information Gain in ID₃.
 - a) We need the two measures "Entropy of a Dataset S "= $H(S)$ & "Information Gain"= IG .
 - b) see next slides

ID3 algorithm (2/7) – Measures (see Wikipedia)

Entropy

Entropy $H(S)$ is a measure of the amount of uncertainty in the (data) set S (i.e. entropy characterizes the (data) set S).

$$H(S) = - \sum_{c \in C} p(c) \log_2 p(c)$$

Where,

- S – The current (data) set for which entropy is being calculated (changes every iteration of the ID3 algorithm)
- C – Set of classes in S $C = \{ \text{yes, no} \}$
- $p(c)$ – The proportion of the number of elements in class c to the number of elements in set S

When $H(S) = 0$, the set S is perfectly classified (i.e. all elements in S are of the same class).

In ID3, entropy is calculated for each remaining attribute. The attribute with the **smallest** entropy is used to split the set S on this iteration. The higher the entropy, the higher the potential to improve the classification here.

Information gain

Information gain $IG(A)$ is the measure of the difference in entropy from before to after the set S is split on an attribute A . In other words, how much uncertainty in S was reduced after splitting set S on attribute A .

$$IG(A, S) = H(S) - \sum_{t \in T} p(t) H(t)$$

Where,

- $H(S)$ – Entropy of set S
- T – The subsets created from splitting set S by attribute A such that $S = \bigcup_{t \in T} t$
- $p(t)$ – The proportion of the number of elements in t to the number of elements in set S
- $H(t)$ – Entropy of subset t

In ID3, information gain can be calculated (instead of entropy) for each remaining attribute. The attribute with the **largest** information gain is used to split the set S on this iteration.

Remarks: Entropy for a binary classification problem

- If all examples are positive or all are negative then entropy will be **zero** i.e. low.
- If half of the examples are of positive class and half are of negative class then entropy is **one** i.e. high

ID3 algorithm (3/7) – Steps to Build Tree

1. Compute the Entropy for Dataset = $H(S)$ (use the above formula) and fill the Frequency Tables for all features.
2. For every attribute/feature:
 - i. Calculate entropy for all categorical values
 - ii. Take average information entropy for the current attribute
 - iii. Calculate gain for the current attribute
3. Pick the highest gain attribute.
4. Repeat until we get the tree we desired.

Example: Compute the entropy for the weather data set

$$H(S) = \sum_{c \in C} -p(c) \log_2 p(c)$$

$$C = \{\text{yes, no}\}$$

Out of 14 instances, 9 are classified as yes,
and 5 as no

$$p_{\text{yes}} = -\left(\frac{9}{14}\right) \log_2 \left(\frac{9}{14}\right) = 0.41$$

$$p_{\text{no}} = -\left(\frac{5}{14}\right) \log_2 \left(\frac{5}{14}\right) = 0.53$$

$$H(S) = p_{\text{yes}} + p_{\text{no}} = 0.94$$

See the calculation of the measure $IG(S, \text{Outlook})$ for the “Playing Tennis” data:

1. Step: Calculate total entropy
For this, the total number of yes/no events must be counted.

$$H(S) = -\left(\frac{9}{14} \log_2 \left(\frac{9}{14}\right) + \frac{5}{14} \log_2 \left(\frac{5}{14}\right)\right) \approx 0.940$$

2. Step: Calculate Information Gain for each feature
Calculate entropy for each classification::

outlook	overcast	sunny	rainy	sum
YES	4	2	3	9
NO	0	3	2	5
sum	4	5	5	14

$$H(\text{outlook} = \text{overcast}) = -\left(\frac{4}{4} \log_2 \left(\frac{4}{4}\right) + 0 \log_2 (0)\right) = 0$$

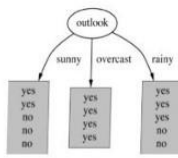
$$H(\text{outlook} = \text{sunny}) = -\left(\frac{2}{5} \log_2 \left(\frac{2}{5}\right) + \frac{3}{5} \log_2 \left(\frac{3}{5}\right)\right) \approx 0.971$$

$$H(\text{outlook} = \text{rainy}) = -\left(\frac{3}{5} \log_2 \left(\frac{3}{5}\right) + \frac{2}{5} \log_2 \left(\frac{2}{5}\right)\right) \approx 0.971$$

feature's information gain:

$$IG(S, A_{\text{outlook}}) = 0.94 - \left(\frac{4}{14} \cdot 0 + \frac{5}{14} \cdot 0.971 + \frac{5}{14} \cdot 0.971\right) = 0.246$$

ID3 algorithm (4/7) – Entropy & Information Gain

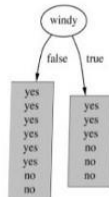


$$\left. \begin{aligned} E(\text{Outlook}=\text{sunny}) &= -\frac{2}{5} \log\left(\frac{2}{5}\right) - \frac{3}{5} \log\left(\frac{3}{5}\right) = 0.971 \\ E(\text{Outlook}=\text{overcast}) &= -1 \log(1) - 0 \log(0) = 0 \\ E(\text{Outlook}=\text{rainy}) &= -\frac{3}{5} \log\left(\frac{3}{5}\right) - \frac{2}{5} \log\left(\frac{2}{5}\right) = 0.971 \end{aligned} \right\} H(S, \text{Outlook})$$

Average Entropy information for Outlook

$$I(\text{Outlook}) = \frac{5}{14} * 0.971 + \frac{4}{14} * 0 + \frac{5}{14} * 0.971 = 0.693 \quad \left. \begin{aligned} & \sum_{t \in T} p(t) H(t) \end{aligned} \right\}$$

$$\text{Gain}(\text{Outlook}) = E(S) - I(\text{outlook}) = 0.94 - 0.693 = 0.247 \quad \Rightarrow \quad IG(A, S) = H(S) - \sum_{t \in T} p(t) H(t)$$



$$E(\text{Windy}=\text{false}) = -\frac{6}{14} \log\left(\frac{6}{14}\right) - \frac{2}{14} \log\left(\frac{2}{14}\right) = 0.811$$

$$E(\text{Windy}=\text{true}) = -\frac{2}{6} \log\left(\frac{2}{6}\right) - \frac{2}{6} \log\left(\frac{2}{6}\right) = 1$$

Average entropy information for Windy

$$I(\text{Windy}) = \frac{8}{14} * 0.811 + \frac{6}{14} * 1 = 0.892$$

$$\text{Gain}(\text{Windy}) = E(S) - I(\text{Windy}) = 0.94 - 0.892 = 0.048$$

Similarly we can calculate for other two attributes (Humidity and Temp).

Detailed calculations are found in “Exercises2Lecture.pdf”
See for example the calculation of IG(S,T Temperature):

temperature	hot	mild	cool	sum
YES	2	4	3	9
NO	2	2	1	5
sum	4	6	4	14

$$H(\text{temp} = \text{hot}) = -\left(\frac{2}{4} \log_2\left(\frac{2}{4}\right) + \frac{2}{4} \log_2\left(\frac{2}{4}\right)\right) = 1$$

$$H(\text{temp} = \text{mild}) = -\left(\frac{4}{6} \log_2\left(\frac{4}{6}\right) + \frac{2}{6} \log_2\left(\frac{2}{6}\right)\right) \approx 0.918$$

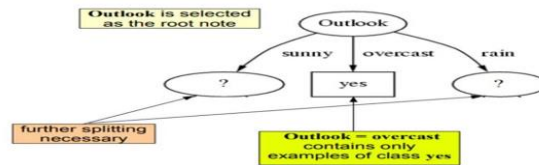
$$H(\text{temp} = \text{cool}) = -\left(\frac{3}{4} \log_2\left(\frac{3}{4}\right) + \frac{1}{4} \log_2\left(\frac{1}{4}\right)\right) \approx 0.811$$

Which results in: $IG(S, \text{Temp}) = 0.94 - (4/14)*1 - (6/14)*0.918 - (4/14)*0.811 \approx 0,02914$

ID3 algorithm (5/7) – Results for Root Node

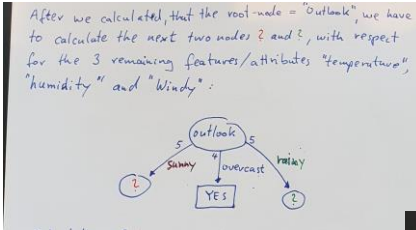
Outlook	Temperature
Info: 0.940-0.693	Info: 0.940-0.911
Gain: 0.940-0.693	Gain: 0.940-0.911
Humidity	Windy
Info: 0.940-0.788	Info: 0.940-0.892
Gain: 0.940-0.788	Gain: 0.940-0.892

Pick the highest gain attribute as root node.

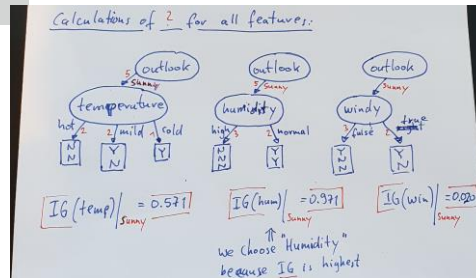


Detailed calculations are found in "Exercises2Lecture.pdf"

ID3 algorithm (6/7) – Calculate Node|outlook=sunny

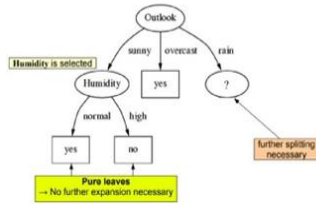


Remark: Details to the calculation of **InfoGain (temperature|outlook=sunny)**, etc. can be found in the Notes-Page to this slide or in the solution of the exercise ML3.1.

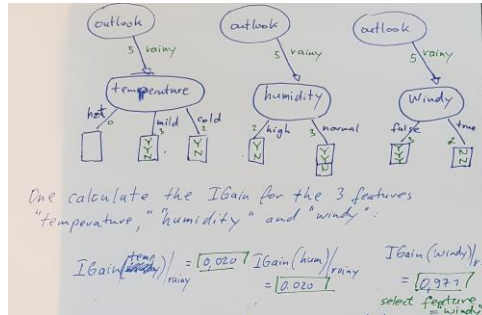


Detailed calculations are found in "Exercises2Lecture.pdf"

ID3 algorithm (7/7) – Final decision tree

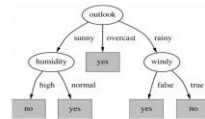


Remark: Details to the calculation of **InfoGain (temperature)|outlook=rainy**, etc. can be found in the Notes-Page to this slide or in the solution of the exercise ML3.1.



Final decision tree

Finally we get the tree something like his:



Detailed calculations are found in "Exercises2Lecture.pdf"

Build Tree with Gini Index (1/8) - Definitions

In CART we use Gini index as a metric. We use the Gini Index as our cost function used to evaluate splits in the dataset.

Gini Index for Binary Target variable $= 1 - \sum_{t=0}^1 P_t^2$

A Gini score gives an idea of how good a split is by how mixed the classes are in the two groups created by the split. A perfect separation results in a Gini score of 0, whereas the worst case split that results in 50/50 classes:

For Binary Target variable, **Max Gini Index** value $= 1 - (1/2)^2 - (1/2)^2$
 $= 1 - 2 \cdot (1/2)^2 = 1 - 2 \cdot (1/4) = 1 - 0.5 = 0.5$

Similarly if Target Variable is categorical variable with multiple levels, the Gini Index will be still similar. If Target variable takes k different values, the Gini

Index will be $1 - \sum_{t=0}^{k-1} P_t^2$ (Maximum value Gini Index is $= 1 - 1/k$)

Minimum value of Gini Index will be 0 when all observations belong to one label.

GINI Index is also used in Data Mining lecture and is therefore a repeating of the concepts. By using a different example like the DM lecture the GINI measure is better understood.

Compare also the following YouTube video “Gini index based Decision Tree” about the calculation of a Gini index based Decision Tree:

<https://www.youtube.com/watch?v=2IEcfRuHFV4>

Build Tree with Gini Index (2/8) – Continuous Attributes

- For efficient computation: for each attribute,
 - Sort the attribute on values
 - Linearly scan these values, each time updating the count matrix and computing gini index
 - Choose the split position that has the least gini index

Cheat	No	No	No	Yes	Yes	Yes	No	No	No	No												
Taxable Income																						
	60	70	75	85	90	95	100	120	125	220												
	55	65	72	80	87	92	97	110	122	172	230											
	<=	>	<=	>	<=	>	<=	>	<=	>	<=	>	<=	>								
Yes	0	3	0	3	0	3	0	3	1	2	2	1	3	0	3	0	3	0	3	0		
No	0	7	1	6	2	5	3	4	3	4	3	4	3	4	4	3	5	2	6	1	7	0
Gini	0.420	0.400	0.375	0.343	0.417	0.400	<u>0.300</u>	0.343	0.375	0.400	0.420	0.420	0.375	0.400	0.420	0.420	0.420	0.420	0.420	0.420	0.420	

Remark: For the calculation of the **Gini index** for each cell see Notes-Page of this slide.

First calculate the **Gini-Index** of the first cell, we write: **Gini (55):**

Gini(55) = Frq(<=55)*Gini(<=55)+Frq(>55)*Gini(>55) where $Frq(X) = \frac{\#(\text{values in } X)}{\#(\text{values in cell})}$. We calculate: $Gini(<=55) = 1 - 0^2 - 0^2 = 1$

$Gini(>55) = 1 - (3/10)^2 - (7/10)^2 = (100 - 9 - 49)/100 = 42/100 = 0.42$

$\Rightarrow Gini(55) = 0/10 * 1 + 10/10 * 0.42 = 0.420$

Second cell: **Gini (65):** We calculate: $Gini(<=65) = 1 - 0^2 - 1^2 = 0$

$Gini(>65) = 1 - (3/9)^2 - (6/9)^2 = (81 - 9 - 36)/81 = 36/81 = 4/9$

$\Rightarrow Gini(65) = 1/10 * 0 + 9/10 * 4/9 = 0.400$

Similar: **Gini(72) = 2/10*(1-0^2-1^2)+8/10*(1-(3/8)^2-(5/8)^2) = 4/5*(64-9-25)/64 = 4/5*30/64 = 3/8 = 0.375**

Gini(80) = 3/10*(1-0-1^2)+7/10*(1-(3/7)^2-(4/7)^2) = 7/10*(49-9-16)/49 = 7/10*24/49 = 12/35 = 0.343

Gini(87) = 4/10*(1-(1/4)^2-(3/4)^2)+6/10*(1-(2/6)^2-(4/6)^2) = 4/10*(16-1-9)/16+6/10*(36-4-16)/36 = 3/20+4/15 = (9+16)/60 = 25/60 = 5/12 = 0.417

Gini(92) = 5/10*(1-(2/5)^2-(3/5)^2)+5/10*(1-(1/5)^2-(4/5)^2) = 5/10*(25-4-9)/25+5/10*(25-1-16)/25 = 5/10*12/25+5/10*8/25 = 6/25+4/25 = 10/25 = 40/100 = 0.400

Gini(97) = 6/10*(1-(3/6)^2-(3/6)^2)+4/10*(1-(0/4)^2-(4/4)^2) = 6/10*(1-1/4-1/4)+4/10*0 = 6/10*1/2 = 3/10 = 0.300

Per symmetry of the cell values you can see, that: **Gini(110) = Gini(80) = 0.343;**

Gini(122) = Gini(72) = 0.375; Gini(172) = Gini(65) = 0.400 and Gini(230) = Gini(55) = 0.420

Result: the best split value is 97, as this has the lowest Gini-Index.

Build Tree with Gini Index (3/8) – Gini (Node)

Gini Index for a given node t :

$$GINI(t) = 1 - \sum_j [P(j | t)]^2$$

(NOTE: $p(j | t)$ is the relative frequency of class j at node t).

- Maximum ($1 - 1/n_c$) when records are equally distributed among all classes, implying least interesting information
- Minimum (0.0) when all records belong to one class, implying most interesting information

C1	0
C2	6

$P(C1) = 0/6 = 0$ $P(C2) = 6/6 = 1$
Gini = $1 - P(C1)^2 - P(C2)^2 = 1 - 0 - 1 = 0$

C1	1
C2	5

$P(C1) = 1/6$ $P(C2) = 5/6$
Gini = $1 - (1/6)^2 - (5/6)^2 = 0.278$

C1	2
C2	4

$P(C1) = 2/6$ $P(C2) = 4/6$
Gini = $1 - (2/6)^2 - (4/6)^2 = 0.444$

Calculate the **Gini Index** of each of the **attributes/features** of “Play Tennis” example. Use the Frequency Table $X \rightarrow y$:

$$\begin{aligned} \text{Gini (outlook)} &= 4/14 * \text{Gini(overcast)} + 5/14 * \text{Gini(sunny)} + 5/14 * \text{Gini(rainy)} = 4/14 * (1 - (4/4)^2 - (0/4)^2) + 5/14 * (1 - (2/5)^2 - (3/5)^2) + 5/14 * (1 - (3/5)^2 - (2/5)^2) \\ &= 4/14 * (1 - 1 - 0) + 5/14 * (25/25 - 4/25 - 9/25) + 5/14 * (25/25 - 9/25 - 4/25) \\ &= 4/14 * 0 + 5/14 * (12/25) + 5/14 * 12/25 = 10/14 * 12/25 = 2/7 * 6/5 = 12/35 = \sim 0.343 \end{aligned}$$

$$\begin{aligned} \text{Gini (temp)} &= 4/14 * \text{Gini(hot)} + 6/14 * \text{Gini(mild)} + 4/14 * \text{Gini(cold)} = 4/14 * (1 - (2/4)^2 - (2/4)^2) + 6/14 * (1 - (4/6)^2 - (2/6)^2) + 4/14 * (1 - (3/4)^2 - (1/4)^2) \\ &= 4/14 * (1 - 1/4 - 1/4) + 6/14 * (36/36 - 16/36 - 4/36) + 4/14 * ((16 - 9 - 1)/16) \\ &= 4/14 * 1/2 + 6/14 * (16/36) + 4/14 * 6/16 = 1/7 + 4/21 + 3/28 = 37/84 = \sim 0.44 \end{aligned}$$

Analogous: **Gini(windy)** $\sim 0,428$; **Gini(humidity)** $\sim 0,367$ (see slide p.68)

Build Tree with Gini Index (4/8) – Algorithm Steps

1. Compute the gini index for data-set. This value is not needed for the building(modeling) the tree, but to have a quality measure for the tree. Fill the frequency-table for all features (all values).
2. For every attribute/feature:
 - i. Calculate gini index for all categorical values
 - ii. Take average information entropy for the current attribute
 - iii. Calculate the gini gain (lowest gini is best)
3. Pick the best gini gain attribute.
4. Repeat until we get the tree we desired.

We calculate it for every row and split the data accordingly in our binary tree.
We repeat this process recursively.

Build Tree with Gini Index (5/8) – Step1

The calculations are similar to ID3 ,except the formula changes.

Compute **Gini index** for dataset = Gini(S):

Fill Frequency Table:

$$= 1 - \sum_{t=0}^{t=1} P_t^2$$

Out of 14 instances ,
yes=9,no=5
 $1 - (9/14)^2 - (5/14)^2$

$1 - 0.413 - 0.127 = 0.46$
Gini = 0.46

outlook	overcast	sunny	rainy	sum
YES	4	2	3	9
NO	0	3	2	5
sum	4	5	5	14

humidity	high	normal	sum
YES	3	6	9
NO	4	1	5
sum	7	7	14

temperature	hot	mild	cool	sum
YES	2	4	3	9
NO	2	2	1	5
sum	4	6	4	14

windy	FALSE	TRUE	sum
YES	6	3	9
NO	2	3	5
sum	8	6	14

Build Tree with Gini Index (6/8) – Step2 (Gini_{Split})

- GINI based Splitting: used in CART, SLIQ, SPRINT.
- When a node p is split into k partitions (children), the quality of split is computed as,

$$GINI_{split} = \sum_{i=1}^k \frac{n_i}{n} GINI(i)$$

where, n_i = number of records at child i,
 n = number of records at node p.

“Playing Tennis” example -Calculate **Gini(outlook)** (use Frequency Table):

Fill Frequency Table for **outlook**:

outlook	overcast	sunny	rainy	sum
YES	4	2	3	9
NO	0	3	2	5
sum	4	5	5	14

$$\begin{aligned} \text{Gini (outlook)} &= 4/14 * \text{Gini(overcast)} + 5/14 * \text{Gini(sunny)} + 5/14 * \text{Gini(rainy)} \\ &= 4/14 * (1 - (4/4)^2 - (0/4)^2) + 5/14 * (1 - (2/5)^2 - (3/5)^2) + 5/14 * (1 - (3/5)^2 - (2/5)^2) \\ &= 4/14 * (1 - 1 - 0) + 5/14 * (25/25 - 4/25 - 9/25) + 5/14 * (25/25 - 9/25 - 4/25) \\ &= 4/14 * 0 + 5/14 * (12/25) + 5/14 * (12/25) = 10/14 * 12/25 = 2/7 * 6/5 = 12/35 = 0.343 \end{aligned}$$

Similar: **Gini(windy)=0.429**; **Gini(temp)=0.44**; **Gini(humidity)=0.367**
=> choose **outlook** = root node

Calculate the **Gini Index** of each of the attributes/features of “Play Tennis” example. Use the Frequency Table $X \rightarrow y$:

Gini (temp)

$$\begin{aligned} &= 4/14 * \text{Gini(hot)} + 6/14 * \text{Gini(mild)} + 4/14 * \text{Gini(cold)} \\ &= 4/14 * (1 - (2/4)^2 - (2/4)^2) + 6/14 * (1 - (4/6)^2 - (2/6)^2) + 4/14 * (1 - (3/4)^2 - (1/4)^2) \\ &= 4/14 * (1 - 1/4 - 1/4) + 6/14 * (36/36 - 16/36 - 4/36) + 4/14 * (16/16 - 9/16 - 1/16) = 4/14 * 1/2 \\ &+ 6/14 * (16/36) + 4/14 * 6/16 = 1/7 + 4/21 + 3/28 = 37/84 = \sim 0.44 \end{aligned}$$

Gini(windy)

$$\begin{aligned} &= 8/14 * \text{Gini(false)} + 6/14 * \text{Gini(true)} \\ &= 8/14 * (1 - (6/8)^2 - (2/8)^2) + 6/14 * (1 - (3/6)^2 - (3/6)^2) \\ &= 8/14 * (16/16 - 9/16 - 4/16) + 6/14 * (1 - 1/4 - 1/4) = 3/14 * 10/16 + 6/14 * 1/2 \\ &= 3/14 + 3/14 = 6/14 = 3/7 = \sim 0.429 \end{aligned}$$

Gini(humidity)

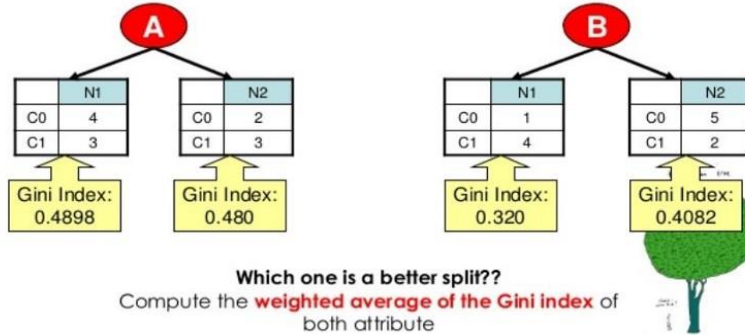
$$\begin{aligned} &= 7/14 * \text{Gini(high)} + 7/14 * \text{Gini(normal)} \\ &= 7/14 * (1 - (3/7)^2 - (4/7)^2) + 7/14 * (1 - (6/7)^2 - (1/7)^2) \\ &= 7/14 * (49/49 - 9/49 - 16/49) + 7/14 * (49/49 - 36/49 - 1/49) = 7/14 * 24/49 + 7/14 * 12/49 = 12/49 \\ &+ 6/49 = 18/49 = \sim 0.367 \end{aligned}$$

=> **outlook** = root node

Build Tree with Gini Index (7/8) – Step2-4

In the same procedure, we can follow other steps to build the tree with Gini index

Suppose there are two ways (A and B) to split the data into smaller subset.



Gini(A) = 17/35 **Gini(B) = 13/35**
→ the option **B** is the better choice

Calculate the above numbers:

Case A:

$$\text{Gini (N1)} = 1 - (4/7)^2 + (3/7)^2 = 49/49 - 16/49 - 9/49 = 24/49 \sim 0.4898$$

$$\text{Gini (N2)} = 1 - (2/5)^2 + (3/5)^2 = 25/25 - 4/25 - 9/25 = 12/25 = 0.48$$

$$\Rightarrow \text{Gini(A)} = 7/12 * 24/49 + 5/12 * 12/25 = \text{shorten} = 2/7 + 1/5 = \mathbf{17/35}$$

Case B:

$$\text{Gini (N1)} = 1 - (1/5)^2 + (4/5)^2 = 25/25 - 1/25 - 16/25 = 8/25 = 0.32$$

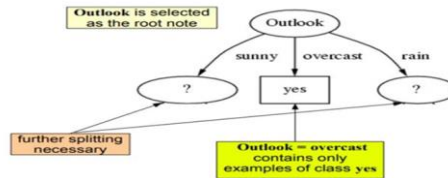
$$\text{Gini (N2)} = 1 - (5/7)^2 + (2/7)^2 = 49/49 - 25/49 - 4/49 = 20/49 \sim 0.4082$$

$$\Rightarrow \text{Gini(B)} = 5/12 * 8/25 + 7/12 * 20/49 = \text{shorten} = 2/15 + 5/21 = 14/105 + 25/105 = 39/105 = \mathbf{13/35}$$

==> option B is the better choice.

Build Tree with Gini Index (8/8) – Step4

Calculate the left node for outlook feature=sunny.



Frequency Table (temp|sunny)

temperature with outlook=sunny	hot	mild	cool	sum
YES	0	1	1	2
NO	2	1	0	3
sum	2	2	1	5

Gini(temp|outlook=sunny)

$$= 2/5 * (1 - (0/2)^2 - (2/2)^2) + 2/5 * (1 - (1/2)^2 - (1/2)^2) + 1/5 * (1 - (1/1)^2 - (0/1)^2)$$

$$= 2/5 * (1 - 0 - 1) + 2/5 * (1 - 1/4 - 1/4) + 1/5 * (1 - 1 - 0)$$

$$= 2/5 * 1/2 = 1/5 = 0.2$$

Similar: Gini(windy|sunny) ~ 0,267; Gini(humidity|sunny) = 0
=> choose **humidity** = left node

windy with outlook=sunny	FALSE	TRUE	sum
YES	1	1	2
NO	2	1	3
sum	3	2	5

humidity with outlook=sunny	high	normal	sum
YES	0	2	2
NO	3	0	3
sum	3	2	5

Gini(windy|sunny)

$$= 3/5 * (1 - (1/3)^2 - (2/3)^2) + 2/5 * (1 - (1/2)^2 - (1/2)^2)$$

$$= 3/5 * (9/9 - 1/9 - 4/9) + 2/5 * (1 - 1/4 - 1/4) = 3/5 * 4/9 + 2/5 * 1/2 = 7/15 = \sim 0.267$$

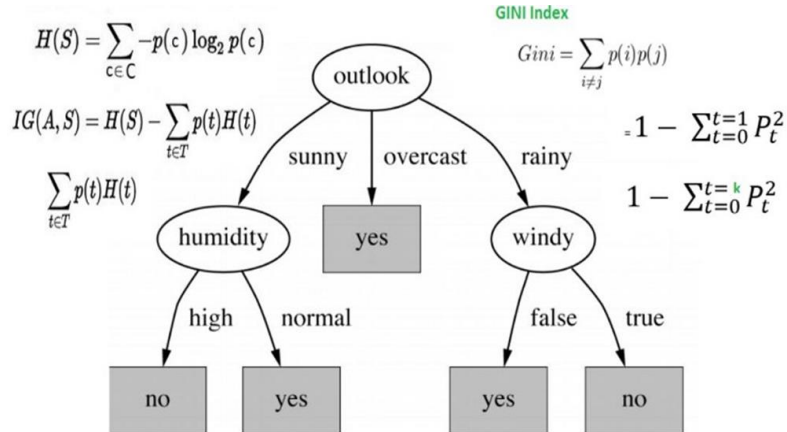
Gini(humidity|sunny)

$$= 3/5 * (1 - (0/3)^2 - (3/3)^2) + 2/5 * (1 - (2/2)^2 - (0/2)^2) = 3/5 * (1 - 0 - 1) + 2/5 * (1 - 1 - 0) = 0$$

➔ **Humidity** is the left node

Final Decision Tree - Summary

Calculate $Gini(temp|rainy) = 0.467$ and $Gini(windy|rainy) = 0 \rightarrow$ choose **windy** = right node



temperature with outlook=rainy	hot	mild	cool	sum
YES	0	2	1	2
NO	0	1	1	3
sum	0	3	2	5

Gini(temp|rainy)

$$= 0 * 3/5 + 3/5 * (1 - (2/3)^2 - (1/3)^2) + 2/5 * (1 - (1/2)^2 - (1/2)^2) = 3/5 * (4/9) + 2/5 * 1/2$$

$$= 4/15 + 3/15 = 7/15 = \sim 0.467$$

Remark*: there is no data record for the value "hot" --> Training Set is too small.

windy with outlook=rainy	FALSE	TRUE	sum
YES	3	0	2
NO	0	2	3
sum	3	2	5

Gini(windy|rainy)

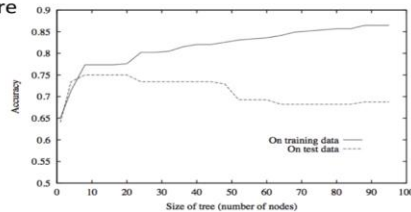
$$= 3/5 * (1 - (3/3)^2 - (0/3)^2) + 2/5 * (1 - (0/2)^2 - (2/2)^2) = 3/5 * (1 - 1 - 0) + 2/5 * (1 - 0 - 1) = 3/5 * 0 + 2/5 * 0 = 0$$

→ Windy is the right node

Overfitting in Decision Trees

One of the questions that arises in a decision tree algorithm is the optimal size of the final tree. A tree that is too large risks overfitting the training data and poorly generalizing to new samples. A small tree might not capture important structural information about the sample space. However, it is hard to tell when a tree algorithm should stop because it is impossible to tell if the addition of a single extra node will dramatically decrease error. This problem is known as the horizon effect. A common strategy is to grow the tree until each node contains a small number of instances then use pruning to remove nodes that do not provide additional information.^[1]

- Can always classify training examples perfectly
 - keep splitting until each node contains 1 example
 - singleton = pure
- Doesn't work on new data



Overfitting and Pruning shows limitations of ML Decisions Tree methods and give the reason to consider special ML methods for special problems. A ML method with fits all applications did not exist.

In statistics, **overfitting** is "the production of an analysis that corresponds too closely or exactly to a particular set of data, and may therefore fail to fit additional data or predict future observations reliably".^[1]

An **overfitted model** is a statistical model that contains more parameters than can be justified by the data.^[2]

The essence of overfitting is to have unknowingly extracted some of the residual variation (i.e. the noise) as if that variation represented underlying model structure.

See also for more details:

https://www.youtube.com/watch?v=i_0-5rdxsfg

Decision Tree Pruning

Pruning is a technique in [machine learning](#) that reduces the size of [decision trees](#) by removing sections of the tree that provide little power to classify instances. Pruning reduces the complexity of the final [classifier](#), and hence improves predictive accuracy by the reduction of [overfitting](#). Pruning should reduce the size of a learning tree without reducing predictive accuracy as measured by a [cross-validation](#) set. There are many techniques for tree pruning that differ in the measurement that is used to optimize performance.

Common Method: Cost Complexity Pruning

Cost complexity pruning generates a series of trees $T_0 \dots T_m$ where T_0 is the initial tree and T_m is the root alone. At step i , the tree is created by removing a subtree from tree $i - 1$ and replacing it with a leaf node with value chosen as in the tree building algorithm. The subtree that is removed is chosen as follows:

1. Define the error rate of tree T over data set S as $\text{err}(T, S)$.
2. The subtree that minimizes $\frac{\text{err}(\text{prune}(T, t), S) - \text{err}(T, S)}{|\text{leaves}(T)| - |\text{leaves}(\text{prune}(T, t))|}$ is chosen for removal.

The function $\text{prune}(T, t)$ defines the tree gotten by pruning the subtrees t from the tree T . Once the series of trees has been created, the best tree is chosen by generalized accuracy as measured by a training set or cross-validation.

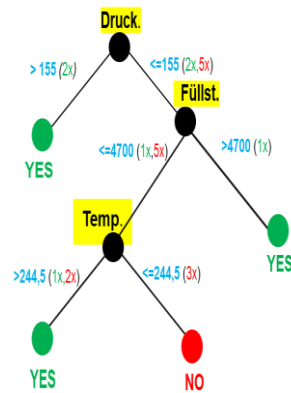
To lessen the chance of, or amount of, overfitting, several techniques are available (e.g. [model comparison](#), [cross-validation](#), [regularization](#), [early stopping](#), [pruning](#), [Bayesian priors](#), or [dropout](#)).

The basis of some techniques is either (1) to explicitly penalize overly complex models or (2) to test the model's ability to generalize by evaluating its performance on a set of data not used for training, which is assumed to approximate the typical unseen data that a model will encounter.

UseCase: Predictive Maintenance in Production (1/10)

Nr.	Anl	Typ	Temp.	Druck	Füllst.	Fehler
1001	123	TN	244	140	4600	nein
1002	123	TO	200	130	4300	nein
1009	128	TSW	245	108	4100	ja
1028	128	TS	250	112	4100	nein
1043	128	TSW	200	107	4200	nein
1088	128	TO	272	170	4400	ja
1102	128	TSW	265	105	4100	nein
1119	123	TN	248	138	4800	ja
1122	123	TM	200	194	4500	ja

Decision Tree



The use case for Predictive Analysis (Maintenance) is a concrete project of IBM and BMW with the goal to improve the quality of production processes.

UseCase: Predictive Maintenance in Production (2/10)

You can either do the calculation of the Decision Tree (using GINI Index) by hand (manually), see Homework H4.2 or by using Python Program with a SKLearn Library - see Homework H4.3:

Define a Decision Tree for a Predictive Maintenance Problem (Homework 4.3 of lesson ML05)

Powered by: Dr. Hermann Völlinger, DHBW Stuttgart (Germany); August 2020, following ideas from Seminarpaper (DHBW SS2020): "Calculation of Decision Trees using GINI-Index" from Heike Filtzke and Paul Mäder.

The solution is part of seminarpaper SW07 in the list of seminarpapers (http://www.lehre.dhbw-stuttgart.de/~hvoellin/Themes_ML_Seminar_Paper.pdf) as part of the Machine Learning lecture by Hermann Völlinger at DHBW Stuttgart (SS2020).

To see more details pls. check JP Notebook with name "Homework-H4_3.ipynb" or Python Pgm. "Homework-H4_3.py" in GitHub Account from H. Völlinger: <https://github.com/HVoellinger/Lecture-Notes-to-ML-WS2020>

The here used algorithms and methods are from Lecture: "ML_Concept&Algorithm (WS2020)"; Chapter ML4. See slides with the titles: "Build Tree with Gini Index (1/8)" until "Build Tree with Gini Index (8/8)".

There are four basic steps when you're implementing this solution:

1. Import libraries and load and prepare training data.
2. Define the Decision Tree for the example data ("Training Data")
3. Calculation of the es GINI Indices and Definition of the Nodes.
4. Define the DTree and print the results (incl. Feature values and Nodes)

Step 1: Import libraries and Load & prepare Training Data

1. Import Libraries and check the versions.
2. Import the data from csv-file: "HomeworkH3_4-data.csv".
3. Define the value "Yes" of column "Fehler" as "1" else set it to "0".
4. Overwrite the column "Fehler" with the new values.
5. Print now the data to check it (omit not needed columns).

UseCase: Predictive Maintenance in Production (3/10)

```
In [1]: # Imports of needed Libraries
import pandas as pd
import numpy as np
import matplotlib as mp
import sklearn as sk
import matplotlib.pyplot as plt
from sklearn.tree import plot_tree
from sklearn.tree import DecisionTreeClassifier

# to check the time of execution, import function time
import time

# check the actual versions of the imported Libraries
print(pd.__version__)
print(np.__version__)
print(mp.__version__)
print(sk.__version__)

1.0.3
1.18.3
3.2.1
0.22.2.post1
```

```
In [2]: # Prepare and Print Training Data
print('This is the list of 3 features and one target column ("Training Data"):')
data = pd.read_csv('Homesork-H4-3-Data.csv')
data['Fehler'] = pd.Series(np.where(data.Fehler.values == 'YES', 1, 0), data.index)
data.drop(['Typ', 'Anl', 'Nr.'], axis=1, inplace=True)
data

This is the list of 3 features and one target column ("Training Data"):
```

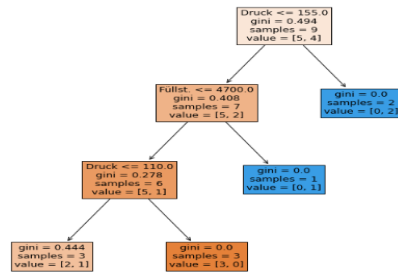
	Temp	Druck	Füllst.	Fehler
0	244	140	4900	0
1	200	130	4300	0
2	246	108	4100	1
3	250	112	4100	0
4	200	107	4200	0
5	272	170	4400	1
6	205	105	4100	0
7	248	138	4800	1
8	200	184	4500	1

UseCase: Predictive Maintenance in Production (4/10)

Step 2: Define the Decision Tree & Calculate GINI Indices

1. Define the features and the target value ("Fehler")
2. Call Function DecisionTreeClassifier with parameters
3. Fit the Decision Tree (DT) model
4. Plot the Dec. Tree

```
In [4]: features = ['Temp.', 'Druck', 'Füllst.']
X = data[features]
y = data.Fehler
crv = DecisionTreeClassifier(max_depth=3, criterion='gini')
crv.fit(X,y)
y_pred = crv.predict(X)
fig = plt.figure()
fig.set_size_inches(10,10)
tree_plot = plot_tree(crv, filled=True,
                      feature_names=features, fontsize=13)
plt.show()
```



UseCase: Predictive Maintenance in Production (5/10)

Step 3: Calculation of the GINI Indices and Definition of the Nodes

1. Calculates the Gini indices and returns them as a list for the specified columns.
2. Finds the next node, outputs it and returns the value and column of the affected value.

In [5]: # Calculates the Gini indices and returns them as a list for the specified columns.

```
def gini(data, split_points, col):
    ges = len(data.index)
    gini_ind = []
    for x in split_points.index:
        high = data[data[col] >= split_points[col][x]].count()[col]
        high_n = data[(data[col] >= split_points[col][x]) &
                      (data['Fehler'] == 0)].count()[col]
        low = data[data[col] < split_points[col][x]].count()[col]
        low_n = data[(data[col] < split_points[col][x]) &
                     (data['Fehler'] == 0)].count()[col]
        if (low != 0):
            g_low = low/ges*(1-((low-low_n)/low)**2-(low_n/low)**2)
        else:
            g_low = 0
        g_high = high/ges*(1-((high-high_n)/high)**2-(high_n/high)**2)
        gini_ind.append(g_high_g_low)
    return(gini_ind)
```

In [6]: # Finds the next node, outputs it and returns the value and column of the affected value.

```
def get_node(data, test_col):
    gini_table = pd.DataFrame()
    split_points = pd.DataFrame()
    low_gini = 1
    for col in data.columns:
        if (col != test_col):
            sorted_data = data.sort_values(by=col, ignore_index=True)
            for x in range(1, len(sorted_data)):
                split_points.at[x-1, col] = (sorted_data[col][x-1] +
                                             sorted_data[col][x]) / 2
                gini_table[col] = gini(sorted_data, split_points, col)
            if (gini_table[col].min() < low_gini):
                low_gini = gini_table[col].min()
                node_col = col
                node_val = split_points[col][gini_table[col].idxmin()]
    print(split_points)
    print(gini_table)
    print(node_col, node_val)
    return (node_val, node_col)
```

UseCase: Predictive Maintenance in Production (6/10)

Step 4: Define the tree and print the results (inclusive all feature-values and nodes) ¶

1. Define the tree with it nodes by running the logic of teh lesson
2. Print the data for all Values of the features
3. Print and show the node values foe all three features

```
In [7]: def tree(data, test_col):
         l_data = data.copy()
         while(len(l_data.columns) > 1 and not l_data.empty):
             node = get_node(l_data, test_col)
             l_data.drop(index = l_data[l_data[node[1]] >=
                 node[0]].index, inplace = True)
             l_data.drop(columns = node[1], inplace = True)
             l_data.reset_index(drop = True, inplace = True)
         return
```

Print the result, i.e.: -> a. Print all steps with it results. -> b. Print the nodea and its values.

```
In [8]: # Print all steps with it results
         # Print the node and its value
         tree(data, 'Fehler')

         Temp.  Druck  Füllst.
0    200.0    186.0    4100.0
1    200.0    197.5    4100.0
2    222.0    110.0    4150.0
3    244.5    121.0    4250.0
4    246.5    134.0    4350.0
5    249.0    129.0    4450.0
6    257.5    155.0    4550.0
7    268.5    182.0    4700.0
```

UseCase: Predictive Maintenance in Production (7/10)

```

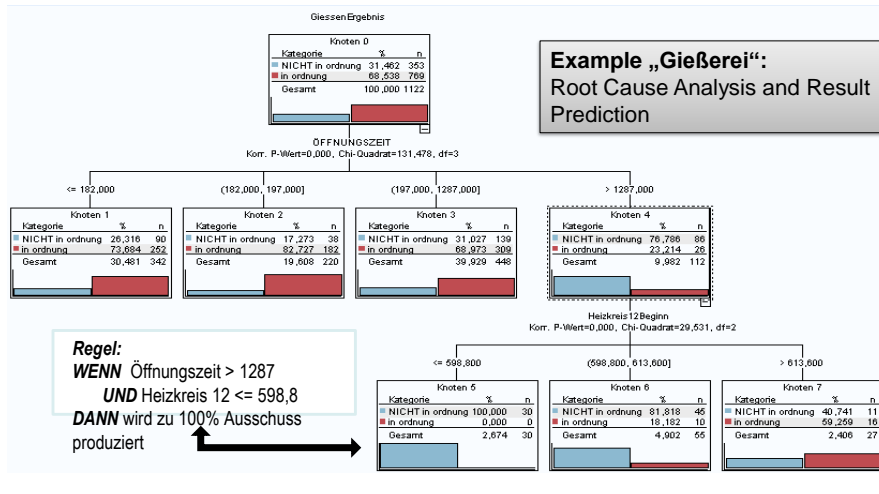
Temp.      Druck      Fullst.
0 0.493827  0.444444  0.493827
1 0.493827  0.380952  0.493827
2 0.481481  0.481481  0.481481
3 0.433333  0.433333  0.433333
4 0.480909  0.344444  0.344444
5 0.481481  0.444444  0.444444
6 0.492063  0.317460  0.492063
7 0.416667  0.416667  0.416667
Druck 155.0
Temp.      Fullst.
0 200.0     4100.0
1 222.0     4100.0
2 244.5     4150.0
3 246.5     4250.0
4 249.0     4450.0
5 257.5     4700.0
Temp.      Fullst.
0 0.408163  0.408163
1 0.342857  0.408163
2 0.285714  0.404762
3 0.484762  0.404762
4 0.342857  0.371429
5 0.380952  0.238095
Fullst. 4700.0
Temp.
0 200.0
1 222.0
2 244.5
3 247.5
4 257.5
Temp.
0 0.277778
1 0.250000
2 0.222222
3 0.250000
4 0.266667
Temp. 244.5

[9]: # print current date and time
print("date",time.strftime("%d.%m.%Y %H:%M:%S"))
print("***** end of Homework H4.3 *****")

date 07.08.2020 22:57:32
***** end of Homework H4.3 *****

```

UseCase: Predictive Maintenance in Production (8/10)



Status: 1. Dezember 2020

Page
110

Another link to the paper you will find in:

<https://nbn-resolving.org/urn:nbn:de:gbv:ilm1-2008000255>

UseCase: Predictive Maintenance in Production (9/10)

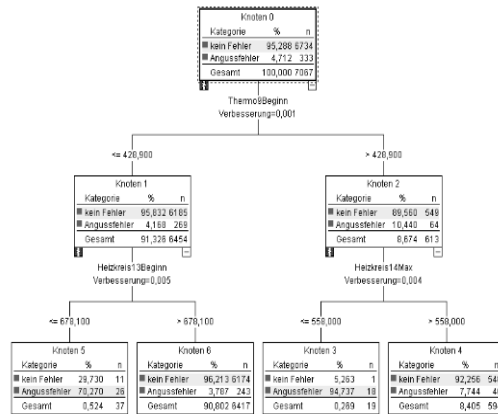


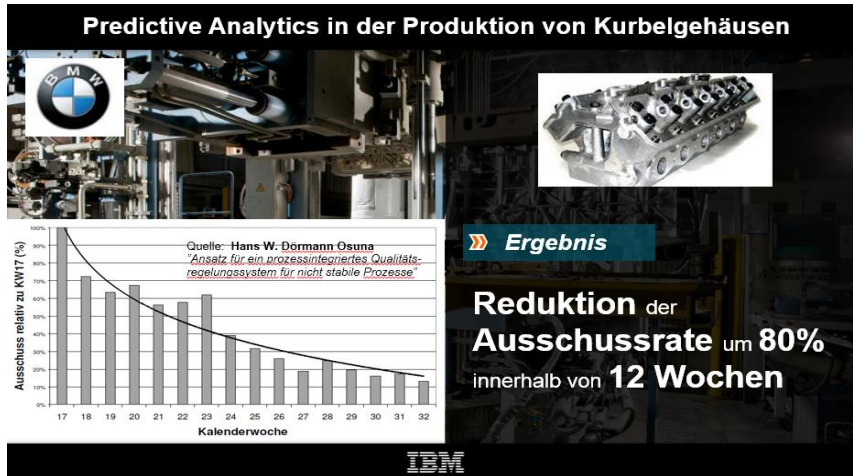
Bild 7.11: CART Baum

Hans W. Dörmann Osuna

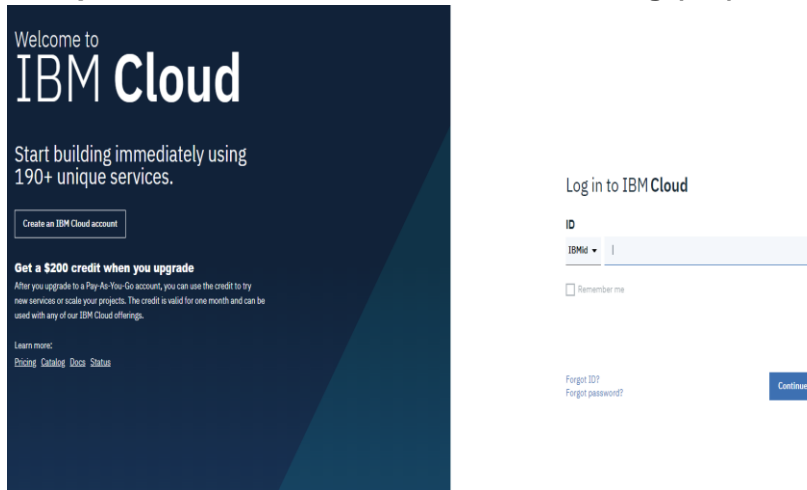
Ansatz für ein prozessintegriertes
Qualitätsregelungssystem für nicht stabile
Prozesse

<http://d-nb.info/992620961/34>

UseCase: Predictive Maintenance in Production (10/10)



Example from IBM Watson- Machine Learning (1/5)



Welcome to
IBM Cloud

Start building immediately using
190+ unique services.

Create an IBM Cloud account

Get a \$200 credit when you upgrade
After you upgrade to a Pay-As-You-Go account, you can use the credit to try new services or scale your projects. The credit is valid for one month and can be used with any of our IBM Cloud offerings.

Learn more
[Pricing](#) [Catalog](#) [Docs](#) [Status](#)

Log in to IBM Cloud

ID
IBMId |

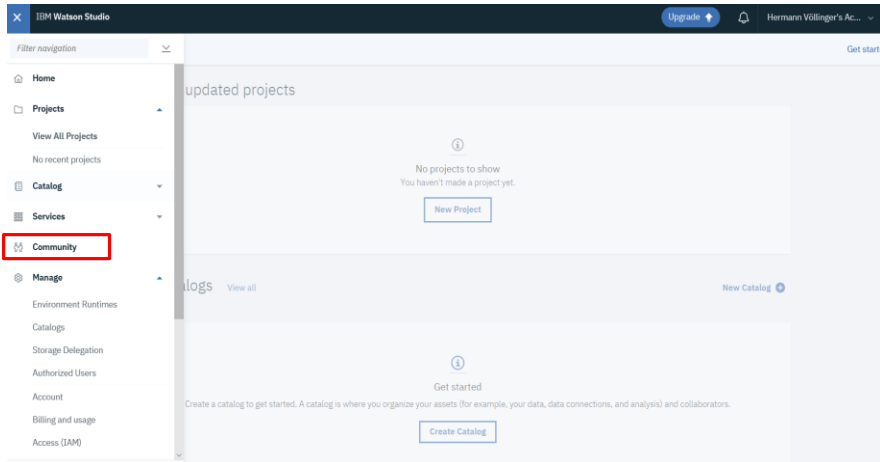
Remember me

[Forgot ID?](#)
[Forgot password?](#)

Continue

[https://console.bluemix.net/dashboard/apps/?cm_mmc=Email Nurture-Cloud Platform-WW WW-LoginBodyLoginButton&cm_mmca1=000002FP&cm_mmca2=10001675&cm_mmca3=M00010245&cvosrc=email.Nurture.M00010245&cv_campaign=Cloud Platform-WW WW](https://console.bluemix.net/dashboard/apps/?cm_mmc=Email+Nurture-Cloud+Platform-WW-WW-LoginBodyLoginButton&cm_mmca1=000002FP&cm_mmca2=10001675&cm_mmca3=M00010245&cvosrc=email.Nurture.M00010245&cv_campaign=Cloud+Platform-WW+WW)

Example from IBM Watson- Machine Learning (2/5)



Example from IBM Watson- Machine Learning (3/5)

All filters

Popular filters: Spark Deep Learning Brunel

Search results (8) Sort by: Most Related

<p>TUTORIAL</p> <p>Ensure loan fairness</p> <p>AUTHOR: IBM Developer DATE: Jan 09, 2019</p> <p>LEVEL: Beginner TOPIC: Notebook +1</p>	<p>DATA SET</p> <p>Country Statistics: Central Bank Discount...</p> <p>AUTHOR: IBM DATE: May 22, 2016</p> <p>TOPIC: Economy & Business</p>	<p>TUTORIAL</p> <p>Predict loan eligibility using IBM Watson...</p> <p>AUTHOR: IBM Developer DATE: Feb 01, 2019</p> <p>LEVEL: Beginner TOPIC: Machine Learning +1</p>	<p>DATA SET</p> <p>Country Statistics: Commercial Bank Prime...</p> <p>AUTHOR: IBM DATE: May 22, 2016</p> <p>TOPIC: Economy & Business</p>
<p>DATA SET</p> <p>External debt stocks, total (DOD, current...</p> <p>AUTHOR: IBM DATE: May 22, 2016</p> <p>TOPIC: Economy & Business</p>	<p>DATA SET</p> <p>Population below national poverty line...</p> <p>AUTHOR: IBM DATE: May 22, 2016</p> <p>TOPIC: Law & Government</p>	<p>NOTEBOOK</p> <p>Predict Loan Applicant Behavior with...</p> <p>AUTHOR: IBM DATE: Mar 23, 2018</p> <p>TOPIC: Economy & Business</p>	<p>DATA SET</p> <p>Ratio (% of population) at national poverty...</p> <p>AUTHOR: IBM DATE: Jun 30, 2016</p> <p>TOPIC: Society</p>

Example from IBM Watson- Machine Learning (4/5)

CODE
Models
Code Patterns
Open Projects

CONTENT
Announcements
Articles
Courses
Series
Tutorials
Videos

COMMUNITY
Blogs
Events

RELATED
Artificial intelligence
Deep learning
Machine learning

TUTORIAL

Predict loan eligibility using IBM Watson Studio

Build a predictive model to automate the process of targeting the right applicants

Hassan AlMuneef | Published January 18, 2019

Cloud Data Science Machine Learning Object Storage Predictive Analytics

Loans are the core business of loan companies. The main profit comes directly from the loan's interest. The loan companies grant a loan after an intensive process of verification and validation. However, they still don't have assurance if the applicant is able to repay the loan with no difficulties.

In this tutorial, we'll build a predictive model to predict if an applicant is able to repay the lending company or not. We will prepare the data using Jupyter Notebook and then build the model using SPSS Modeler.

Learning objectives

After completing this tutorial, you'll understand how to:

- Add and prepare your data
- Build a machine learning model
- Save the model

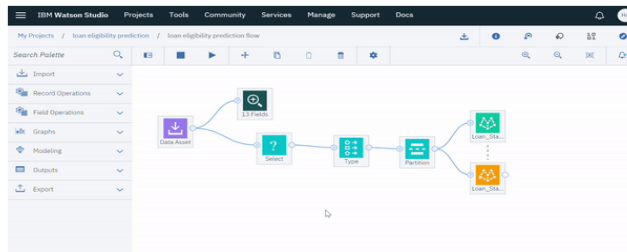
SOCIAL
f t in

CONTENTS
Learning objectives
Prerequisites
Estimated time
Steps
Dataset
Step 1. Create a project in Watson Studio
Step 2. Upload the dataset to Watson Studio
Step 3. Create SPSS modeler flow
Step 4. Add and prepare data
Step 5. Configure variables type
Step 6. Build a machine learning model
Step 7. View the model
Step 8. Evaluate the performance of the model
Step 9. Save the Model
Summary

Example from IBM Watson- Machine Learning (5/5)

Step 8. Evaluate the performance of the model

1. Drag and drop the Analysis node from the Output section, and connect it with the model.
2. After running the node, you can see your analysis report on the right side panel.



The analysis report shows we have achieved 82.3% accuracy on our test data set with this model. At the end, you can build more models within the same canvas until you get the result you want.

Exercises to Lesson ML4

Homework H4.1 - “Calculate ID3 and CART Measures”

Groupwork (2 Persons) - Calculate the measures of decision tree “Playing Tennis Game”:

1. ID3 (Iterative Dichotomiser 3) method using *Entropy Fct. & Information Gain*.
2. CART (Classification) → using *Gini Index(Classification)* as metric.

Homework H4.2 - “Define the Decision Tree for UseCase *Predictive Maintenance* (see slide p.77) by calculating the GINI Indexes”

Groupwork (3 Persons): Calculate the Decision Tree for UseCase “Predictive Maintenance” on slide p.77. Do the following steps (one person per step):

1. Calculate the **Frequency Matrices** for the features „Temp.“, „Druck“ and „Füllst.“
2. Define the **Root-node** by calculating the GINI-Index for all values of the three features. Define the optimal **split-value for the root-node** (see slide p.88)
3. **Finalize the decision tree** by calculation the GINI-Index for the remaining values for the features “Temp.” and “Füllst.”

Solutions are found in Ref. [HVö-4]: “Exercises2Lecture.pdf”

See also for Jupyter Notebooks for Homework 4.2 in [HVö-6]: GitHub/HVoellinger:

<https://github.com/HVoellinger/Lecture-Notes-to-ML-WS2020>

Exercises to Lesson ML4

Homework H4.3 (advanced)*-“Create and describe the algorithm to automate the calculation of the Decision Tree for the Use Case “Predictive Maintenance”

Groupwork (2 Persons): Create and describe the **algorithm to automate the calculation** of steps 1. to 3. of homework H4.2. See more detailed description of the steps in the lecture:

1. Calculate the **Frequency Matrices** for the features „Temp.“, „Druck“ and „Füllst.“
2. Def. **Root-node** by calculating GINI-Index of the three features & find the optimal **split-value** for the root-node.
3. **Finalize the decision tree** by calculation the GINI-Index for the remaining values for the features “Temp.” and “Füllst.”

Homework H4.4* - “Summary of the Article ...*prozessintegriertes Qualitätsregelungssystem...*”

Groupwork (2 Persons) – read and create a short summary about a special part of article/dissertation from Hans W. Dörmann Osuna: “[Ansatz für ein prozessintegriertes Qualitätsregelungssystem für nicht stabile Prozesse](http://d-nb.info/992620961/34)“. Link to article: <http://d-nb.info/992620961/34>

For the two chapters (1 Person each Chapter, 15 Minutes):

- Chapter 7.1 „Aufbau des klassischen Qualitätsregelkreises“
- Chapter 7.2. “Prädiktive dynamische Prüfung”

Solutions are found in Ref. [HVö-4]: “Exercises2Lecture.pdf”

Hint to H4.3: See also for Jupyter Notebooks for Homework H4.3 with the name “*ML4-Homework-H4_3.ipynb*” in [HVö-6]: GitHUb/HVoellinger: <https://github.com/HVoellinger/Lecture-Notes-to-ML-WS2020>

Hint to H4.4*: Another link to the paper you will find in: <https://nbn-resolving.org/urn:nbn:de:gbv:ilm1-2008000255>

Exercises to Lesson ML4

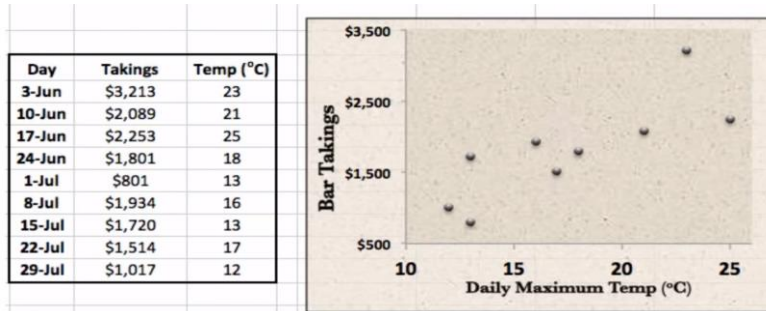
Homework H4.5* - “Create and describe the algorithm to automate the calculation of the Decision Tree for the Use Case “Playing Tennis” using ID₃ method”

Groupwork (2 Persons) - Calculate the measures of decision tree “Playing Tennis Game” by creating a Python Program (i.e. using Jupyter Notebook) with “ID₃ (Iterative Dichotomiser 3)” method using Entropy Fct. & Information Gain

Solutions are found in Ref. **[HVö-4]**: “Exercises2Lecture.pdf”

1. ML1: Introduction to Machine Learning (ML)
2. ML2: Concept Learning: VSpaces & Cand. Elim. Algo.
3. ML3: Supervised and Unsupervised Learning
4. ML4: Decision Tree Learning
5. **ML5: simple Linear Regression (sLR) & multiple Linear Regression (mLR)**
6. ML6: Neural Networks: Convolutional NN
7. ML7: Neural Network: BackPropagation Algorithm
8. ML8: Support Vector Machines (SVM)

ML5 – simple Linear Regression (sLR) & multiple Linear Regression (mLR)



<https://github.com/HVoellinger/Lecture-Notes-to-ML-WS2020/blob/master/QYulc.gif>

Status: 1. Dezember 2020

Page
121

Regression methods are supporting the concept of building a "good" model, when you only know test data (or measurement points). It also shows how to calculate the error and to give you therefore a measure for the quality of the model. The concepts use well-known mathematics (Linear Algebra and n-dim. Geometry).

For more details see also the following YouTube video:
<https://www.youtube.com/watch?v=aq8VU5KlmkY>

Simple Linear Regression: Variable's Roles and sLR Model

You may remember one of these:

$$y = a + b x$$

$$y = m x + b$$

In the stats world, we just use a different notation:

$$y = \beta_0 + \beta_1 x$$

$$y = \beta_0 + \beta_1 x + \varepsilon$$

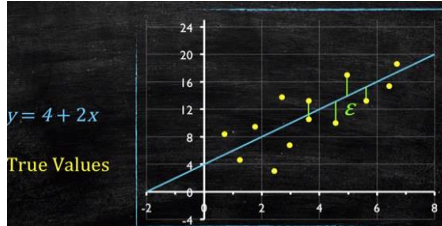
- y is the dependent variable
- x is the independent variable
- β_0 is the constant or intercept
- β_1 is x 's slope or coefficient
- ε is the error term

Establish if there is a **relationship** between two variables.

- More specifically, establish if there is a statistically significant relationship between the two.
- Examples: Income and spending, wage and gender, student height and exam scores.

Forecast new observations.

- Can we use what we know about the relationship to forecast unobserved values?
- Examples: What will be sales be over the next quarter? What will the ROI of a new store opening be contingent on store attributes?

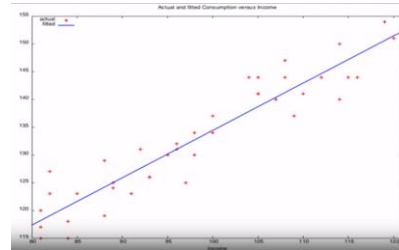
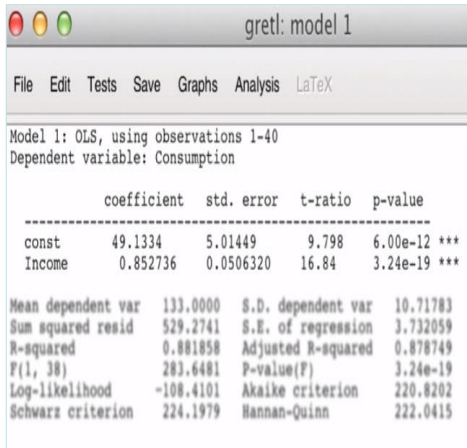


sLR Example: Training-Data $x=income$, $y=consumption$

Business Question: What Properties could explain a family's Consumption?

ID	Income	Consumption	ID	Income	Consumption
1	119	154	21	116	144
2	85	123	22	115	144
3	97	125	23	93	126
4	95	130	24	105	141
5	120	151	25	89	124
6	92	131	26	104	144
7	105	141	27	108	144
8	110	141	28	88	129
9	98	130	29	109	137
10	98	134	30	112	144
11	81	115	31	96	132
12	81	117	32	89	125
13	91	123	33	93	126
14	105	144	34	114	140
15	100	137	35	81	120
16	107	140	36	84	118
17	82	123	37	88	119
18	84	115	38	96	131
19	100	134	39	82	127
20	108	147	40	114	150

sLR Example: Get the LR Result with a Tool



Interpreting the Coefficients

- The estimated model is $Consumption = 49.13 + 0.85 Income + \epsilon$
- 49.13 could be interpreted as the consumption level of a family with 0 income.
- Most generally, intercept doesn't have intuitive interpretation.
- 0.85 is the marginal effect of one unit of income on consumption: for every unit more of income a family has, we estimate its consumption grows by 0.85 units.
- The slope always has an intuitive interpretation.

Definition of the Measure of the sLR Method: “R-Squared” R^2

https://en.wikipedia.org/wiki/Coefficient_of_determination

For the definition of R^2 we need some measures of the training-set (set of observations points). These measures are Sum of Squares Total (SST), Sum of Squares Error (SSE) and Sum of Squares Regression (SSR). SSR is not needed for the definition of R^2 , but we will use it later in the chapter:

SST, SSE and SSR are given by the following definitions (D5.x). Without loss of generality (w.l.o.g) we can assume that SST is greater zero:

$$(D5.1): \text{ Sum of Squares Total (SST) } = \sum_{i=1}^n (y_i - \bar{y})^2$$

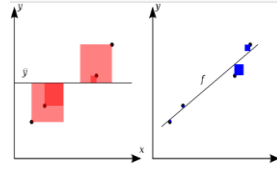
$$(D5.2): \text{ Sum of Squares Error (SSE) } = \sum_{i=1}^n (y_i - f_i)^2$$

$$(D5.3): \text{ Sum of Squares Regression (SSR) } = \sum_{i=1}^n (f_i - \bar{y})^2$$

For the definition of R^2 we need the Sum of Squares Error (SSE) and the Sum of Squares Total (SST). The definition of R^2 is given by the difference of 1 and the quotient of SSE/SST (D5.4):

$$R^2 = 1 - \text{SSE/SST} \quad (D5.4)$$

Notation: We say a line is an “optimal” simple Linear regression (sLF) - line if R^2 is maximal (which is the same as SSE is minimal).



The better the linear regression (on the right) fits the data in comparison to the simple average (on the left graph), the closer the value of R^2 is to 1. The areas of the blue squares represent the squared errors (SSE) with respect to the linear regression. The areas of the red squares represent the squared totals with respect to the average:

$$R^2 = 1 - \text{SSE/SST}$$

Properties & Theorems about the Measure R^2 (1/3)

We see the following theorems:

Theorem (T.S.1): "Prop. of R^2 "

(i) $0 \leq R^2 \leq 1$ "Limitation"
 (ii) $R^2 = 0 \Leftrightarrow SSE = SST$ "Minimum"
 (iii) $R^2 = 1 \Leftrightarrow SSE = 0$ "Maximum"

Define:
 $SST = \sum (y_i - \text{Mean}(y_i))^2$
 $SSE = \sum (y_i - \hat{y}_i)^2$
 $SSR = \sum (\hat{y}_i - \text{Mean}(y_i))^2$

Without limitation of generality let $SST \neq 0$

Proof: is trivial. We will write it down anyway to practice and repeat mathematical proofs

(i) $R^2 \geq 0 \Leftrightarrow 1 - \frac{SSE}{SST} \geq 0 \Leftrightarrow 1 \geq \frac{SSE}{SST} \Leftrightarrow SST \geq SSE$ (per def.)
 $R^2 \leq 1 \Leftrightarrow 1 \geq 1 - \frac{SSE}{SST} \Leftrightarrow 0 \geq -\frac{SSE}{SST} \Leftrightarrow \frac{SSE}{SST} \geq 0$ (p. def.)

(ii) $0 = 1 - \frac{SSE}{SST} \Leftrightarrow \frac{SSE}{SST} = 1 \Leftrightarrow SSE = SST$
 (iii) $1 = 1 - \frac{SSE}{SST} \Leftrightarrow \frac{SSE}{SST} = 0 \Leftrightarrow SSE = 0$ q.e.d.

Properties/Conclusion out of (T.S.1):

for this two examples the "green" $f(x)$ is a better s/in. Repr. than the red $f(x)$

The question is how well the independent variables are suited to explain the variance of the dependent or to predict their values. This is where the R^2 comes into play. It is a measure that can not be less than 0 and not greater than 1. Since the R^2 is a share value, it is also often given in percent.

If a regression has an R^2 close to 0, it means that the chosen independent variables are not well suited to predict the dependent variable. One speaks then also of a bad model adaptation ("poor model fit").

Properties & Theorems about the Measure R^2 (2/3)

To find the optimal sLR-Regression-Line $y(x) = a + b \cdot x$, you have to find the maximum of function $R^2 = R^2(a, b)$. R^2 is a function in the two variables a (= "intercept") and b (= "slope").

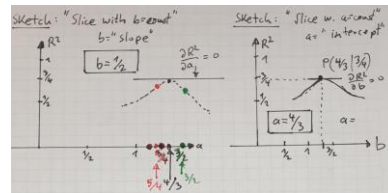
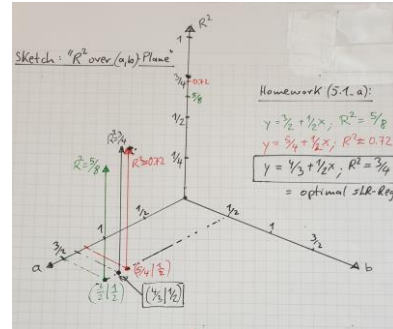
Geometrically this is a "surface" of degree 2 over the (a, b) -plane; see sketch "R² over (a, b) -plane". The maximum in the R^2 -surface looks like the "summit in a hill landscape" (see below):



You get a better impression of the surface, when you check the slices where intercept or slope are constant. See the sketches: "Slice-const. slope" & "Slice-const. intercept" (see the right side).

What are the conditions for R^2 to be maximal? From Calculus we know that a function in variables a, b has *extreme values*, if the differential $dR^2 = (dR^2/da) \cdot da + (dR^2/db) \cdot db = 0$; *max.* if $d(d(R^2)) < 0$.

See the tangential lines in the sketches on the right side. To calculate the two variables a and b you have to solve the two equations: $dR^2/da = 0$ and $dR^2/db = 0$ & check maximum condition. We will use these conditions for the calculations of coefficients a, b in $y = a + b \cdot x$ (see "Least Square Fitting" method).



In the **homework** we will try to visualize the "R²-mountain landscape

Properties & Theorems about the Measure R² (3/3)

Corollary (C5.1): "Simple linear regression without an intercept term (a = 0)"

Sometimes it is appropriate to force the optimal SLR-line to pass through the origin, because x and y are assumed to be proportional, then: $b = \text{Mean}(x \cdot y) / \text{Mean}(x^2)$

Proof:

$$\text{Let } y = b \cdot x \Rightarrow R^2 = 1 - \text{SSE} / \text{SST} = 1 - (1 / \text{SST}) \cdot \sum [y_i - b \cdot x_i]^2 \rightarrow 0 = dR^2 / db = -2 \sum [(y_i - b \cdot x_i) \cdot x_i] = -2 \sum [x_i \cdot y_i - b \cdot x_i^2] \rightarrow \sum [x_i \cdot y_i] = b \cdot \sum [x_i^2] \Rightarrow b = \sum [x_i \cdot y_i] / \sum [x_i^2] = n \cdot \text{Mean}(x \cdot y) / n \cdot \text{Mean}(x^2) \rightarrow b = \text{Mean}(x \cdot y) / \text{Mean}(x^2). \quad \text{q.e.d.}$$

We need now some helpful formulas about sums and mean-values, because we need them later in the calculation of "optimal" coefficients a, b (see: "Least Square Fit" (LSF) method):

Proposition (P5.1): Easily you can proof, that the following equations are valid (let $M(x) := \text{Mean}(x)$).

- (i) $\sum [(x_i - M(x))^2] = \sum [x_i^2] - n \cdot M(x)^2$
- (ii) $\sum [(y_i - M(y))^2] = \sum [y_i^2] - n \cdot M(y)^2$
- (iii) $\sum [(x_i - M(x)) \cdot (y_i - M(y))] = \sum [x_i \cdot y_i] - n \cdot M(x) \cdot M(y)$

Proof: "straightforward"

- (i) $\sum [(x_i - M(x))^2] = \sum [x_i^2 - 2 \cdot M(x) \cdot x_i + M(x)^2]$ (binominal formula)
 $= \sum [x_i^2] - 2 \cdot M(x) \cdot \sum [x_i] + \sum [M(x)^2] = \sum [x_i^2] - 2 \cdot n \cdot M(x)^2 + n \cdot M(x)^2$ because $\sum [x_i] = n \cdot M(x)$
- (ii) analog: $\sum [(y_i - M(y))^2] = \sum [y_i^2] - 2 \cdot M(y) \cdot \sum [y_i] + \sum [M(y)^2] = \sum [y_i^2] - 2 \cdot n \cdot M(y)^2 + n \cdot M(y)^2 = \sum [y_i^2] - n \cdot M(y)^2$
- (iii) analog: $\sum [(x_i - M(x)) \cdot (y_i - M(y))] = \sum [x_i \cdot y_i - M(x) \cdot y_i - x_i \cdot M(y) + M(x) \cdot M(y)]$ ("multiply out all factors")
 $= \sum [x_i \cdot y_i] - n \cdot M(x) \cdot M(y) - n \cdot M(x) \cdot M(y) + n \cdot M(x) \cdot M(y) = \sum [x_i \cdot y_i] - n \cdot M(x) \cdot M(y) \quad \text{q.e.d.}$

Umformungen für Prop. (P5.1):

in Prop. (P5.1) we have seen:

- (i): $\sum (x_i - \bar{x})^2 = \sum [x_i^2] - n \cdot \bar{x}^2$
- (ii): $\sum (y_i - \bar{y})^2 = \sum [y_i^2] - n \cdot \bar{y}^2$
- (iii): $\sum [(x_i - \bar{x})(y_i - \bar{y})] = \sum [x_i \cdot y_i] - n \cdot \bar{x} \cdot \bar{y}$

Per definition:

- (a): $\sum [x_i^2] = n \cdot \overline{x^2}$
- (b): $\sum [y_i^2] = n \cdot \overline{y^2}$
- (c): $\sum [x_i \cdot y_i] = n \cdot \overline{xy}$

Conclusion of Comparison:

- (i) & (a): $n \cdot \overline{x^2} = n \cdot \bar{x}^2 + \sum (x_i - \bar{x})^2 = n \cdot \bar{x}^2 + \overline{(x_i - \bar{x})^2} \cdot n$
 $\Rightarrow \overline{x^2} = \bar{x}^2 + \overline{(x_i - \bar{x})^2} \quad \text{(I)}$
- (ii) & (b): $\overline{y^2} = \bar{y}^2 + \overline{(y_i - \bar{y})^2} \quad \text{(II)}$
- (iii) & (c): $\overline{xy} = \bar{x} \cdot \bar{y} + \overline{(x_i - \bar{x})(y_i - \bar{y})} \quad \text{(III)}$

“Least Square Fitting” of a sLR-Line $y = a + b \cdot x$ (1/4)

<https://www.analyzemath.com/calculus/multivariable/linear-least-squares-fitting.html>

Task: Calculate for a simple Linear Regression (sLR)-line $f(x) = a + b \cdot x$ the coefficients a and b such that $f(x)$ is an optimal.

Solution:

The condition for $f(x)$ -optimal is equivalent (“ \Leftrightarrow ”) $R^2 = 1 - \text{SSE}/\text{SST} = \text{max}$. $\leftarrow (T5.1) \Rightarrow \text{SSE} = \text{min}$. We know from the initial lecture mathematics 1 (“extreme value problems”) that the first derivation of S (“ dS ”) must be zero and for a minimum additionally the second derivation (d^2S) is less zero. Start with first derivation: $dS = dS/da \cdot da + dS/db \cdot db = 0 \Leftrightarrow dS/da = 0$ (1) & $dS/db = 0$ (2)

We write “sum” for the symbol: $\sum_{i=1}^n$

Execute now formula (1): $0 = d/da(S) = d/da[\sum(y_i - a - b \cdot x_i)^2] = 2 \cdot \sum(y_i - a - b \cdot x_i) \cdot (-1)$
 $= -2 \cdot [\sum(y_i) - \sum(a) - \sum(b \cdot x_i)]$
 $\Rightarrow 0 = \sum(y_i) - n \cdot a - b \cdot \sum(x_i) = n \cdot M(y) - n \cdot a - n \cdot b \cdot M(x)$
 $\Rightarrow a + b \cdot M(x) = M(y)$ (3)

Similar for formula (2): $0 = d/db(S) = d/db[\sum(y_i - a - b \cdot x_i)^2] = 2 \cdot \sum(y_i - a - b \cdot x_i) \cdot (-x_i)$
 $= -2 \cdot [\sum(x_i \cdot y_i) - \sum(a \cdot x_i) - \sum(b \cdot x_i^2)]$
 $\Rightarrow 0 = \sum(x_i \cdot y_i) - n \cdot a \cdot M(x) - b \cdot \sum(x_i^2)$
 $\Rightarrow a \cdot n \cdot M(x) + b \cdot \sum(x_i^2) = \sum(x_i \cdot y_i)$ (4)

1. $\text{adj}(A) = C^T$ “Adjugate Matrix”

$C = ((-1)^{i+j} M_{ij})_{1 \leq i, j \leq n}$; $M_{ij} = \det.$ of $(n \times n) \times (n-1)$ Matrix from deleting row i & column j

Bsp: $A = 3 \times 3$ Matrix
symmetrisch

$$A = \begin{pmatrix} a & b & c \\ b & d & e \\ c & e & f \end{pmatrix}$$

$$C = \begin{pmatrix} +|d e| & -|b e| & +|b d| \\ -|b c| & +|a c| & -|a b| \\ +|b c| & -|b e| & +|b d| \end{pmatrix}$$

$$= \begin{pmatrix} d e - e^2 & e c - b f & b e - d d \\ e c - b f & a f - c^2 & b c - a e \\ b e - c d & b c - a e & a d - b^2 \end{pmatrix} \begin{matrix} \text{symmetrisch} \\ \parallel \\ C^T \end{matrix}$$

2. $A^{-1} = \frac{1}{\det A} (\text{adj} A)$; A wie bei 1.

$$\det A = a \cdot \begin{vmatrix} d & e \\ e & f \end{vmatrix} - b \begin{vmatrix} b & c \\ e & f \end{vmatrix} + c \begin{vmatrix} b & c \\ d & e \end{vmatrix}$$

$$= a(d f - e^2) - b(b f - e c) + c(b e - c d)$$

“Least Square Fitting” of a sLR-Line $y = a + b \cdot x$ (2/4)

We can rewrite (3) & (4) in matrix form:

$$(3) \Rightarrow \begin{pmatrix} 1 & \bar{x} \\ n\bar{x} & \sum x_i^2 \end{pmatrix} \begin{pmatrix} a \\ b \end{pmatrix} = \begin{pmatrix} \bar{y} \\ \sum x_i y_i \end{pmatrix}$$

To solve this with Matrix A, we have to invert $A := A^{-1}$

From Linear Algebra (Math. I): $A^{-1} := \frac{1}{\det A} \cdot \text{adj}(A)$; $\text{adj}(A) = C^T$ Repeat Math.

From Math. we get $A^{-1} = \frac{1}{\det A} \begin{pmatrix} \sum x_i^2 & -\bar{x} \\ -n\bar{x} & 1 \end{pmatrix}$; $C := \begin{bmatrix} (-1)^{i+j} H_{ij} \end{bmatrix}_{1 \leq i, j \leq n}$
 $H_{ij} = \text{determinant of a } (n-1) \times (n-1) \text{ matrix, that results from del. row } i \text{ and column } j \text{ of } A$

$$\Rightarrow \begin{cases} a = \frac{1}{\det A} \cdot [\bar{y} \sum x_i^2 - \bar{x} \sum x_i y_i] \\ b = \frac{1}{\det A} [\sum x_i y_i - n\bar{x} \bar{y}] \end{cases}; \det A = \sum x_i^2 - n\bar{x}^2$$

So we get the final result:

$$a = \frac{\bar{y} \cdot \sum x_i^2 - \bar{x} \cdot \sum x_i y_i}{\sum x_i^2 - n\bar{x}^2} \quad (I)$$

$$b = \frac{\sum x_i y_i - n\bar{x} \bar{y}}{\sum x_i^2 - n\bar{x}^2} \quad (II)$$

Formula for the Coefficients of an opt. sLR-line $y = a + b \cdot x$

Check that above Formulas define a Maximum:

We have seen that $R^2 = \text{max.} \Leftrightarrow \text{SSE} := S = \text{min}$

We see: $\frac{\partial}{\partial a}(S) := S_a = \frac{\partial}{\partial a} [\sum (y_i - a - b x_i)^2]$

$$= -2 \cdot \sum (y_i - a - b x_i)$$

$$\frac{\partial}{\partial b}(S) := S_b = \frac{\partial}{\partial b} [\sum (y_i - a - b x_i)^2]$$

$$= -2 \cdot \sum (x_i y_i - a x_i - b x_i^2)$$

To proof that $S = \text{min.}$ you have to show:

$$S_{aa} \cdot S_{bb} = S_{ba} \cdot S_{ab} > 0; \quad S_{aa} := \frac{\partial}{\partial a}(S_a) = \frac{\partial}{\partial a} \left(\frac{\partial S}{\partial a} \right)$$

$$S_{aa} = \frac{\partial}{\partial a} [(-2) \cdot \sum (y_i - a - b x_i)] = \frac{\partial}{\partial a} (-2 \cdot \sum y_i + 2a n + 2b \sum x_i) = 0 + 2n + 0 > 0 \quad \checkmark$$

$$S_{bb} = \frac{\partial}{\partial b} [(-2) \sum (x_i y_i - a x_i - b x_i^2)] = \frac{\partial}{\partial b} [(-2) \sum x_i y_i + 2a \sum x_i + 2b \sum x_i^2] = 0 + 2 \sum x_i^2 + 0 = 2 \sum x_i^2 > 0 \quad \checkmark$$

P5.1(i)

$$S_{ab} = \frac{\partial}{\partial a} [(-2) \sum (x_i y_i - a x_i - b x_i^2)] = 0 + 2 \sum x_i + 0 = 2n \bar{x} > 0$$

$$S_{ba} = \frac{\partial}{\partial b} [(-2) \cdot \sum (y_i - a - b x_i)] = 0 + 0 + 2 \sum x_i = 2n \bar{x} > 0$$

"Least Square Fitting" of a sLR-Line $y = a + b \cdot x$ (3/4)

Example of a "Least Squares Fitting" calculation for simple LR

Find the "least square fit" $y = b_0 + b_1x$ for the experimental data points: $\{(1, 2), (3, 4), (2, 6), (4, 8), (5, 12), (6, 13), (7, 15)\}$

Solution:

Number of Point $N=7$ Mean-Values ("Mittelwerte"): $[M(x), M(y)] = [28/7, 60/7] \sim [4; 8,5714]$
Set up a table with the quantities included in the above formulas for b_0 and b_1 and also the quantities for the calculation of R^2 :

i	x_i	y_i	needed for calculation of b_0 and b_1		needed for calculation of R^2			SST = SSE + SSR ?
			$x_i \cdot y_i$	x_i^2	$y(x_i)$	$SSE = \sum (y_i - y(x_i))^2$	$SST = \sum (y_i - M(y))^2$	$SSR = \sum (y(x_i) - M(y))^2$
1	1	2	2	1	2,0357	0,001274	43,3833	42,7101
2	3	4	12	9	6,3929	5,726	20,8977	4,7459
3	2	6	12	4	4,2143	3,1887	6,6121	18,9843
4	4	8	32	16	8,5715	0,3266	0,3265	0
5	5	12	60	25	10,7501	1,5623	11,7553	4,7467
6	6	13	78	36	12,9287	0,00661	19,6125	18,9861
7	7	15	105	49	15,1073	0,01151	41,3269	42,718
sum	28	60	301	140		10,822994	143,7143	132,8911

Substitute these values into Formula I and II:

$b_0 = (140 \cdot 60 / 7 - 4 \cdot 301) / (140 - 7 \cdot 16) = -28 / 196 = -1/7 \sim -0,14286$

$b_1 = (301 - 7 \cdot 4 \cdot (60/7)) / 28 = 61 / 28 \sim 2,1786$

----> Regression-Line: $y = -1/7 + (61/28) \cdot x$

$R^2 = 1 - \text{Sum}((y_i - y(x_i))^2) / \text{Sum}((y_i - M(y))^2) = 1 - (10,822994 / 143,7143) \sim 0,9247$

Compare with Python
intercept: -0.14285714285714057
slope: [2.17857143]

coef. determination: 0.9247017892644135

Rewrite the LSF-formula (I), (II):

(I): $a = \frac{\bar{y} \cdot \sum x_i^2 - \bar{x} \cdot \sum x_i y_i}{\sum x_i^2 - n \bar{x}^2} = \frac{\bar{y} [\sum (x_i - \bar{x})^2 + n \bar{x}^2] - \bar{x} \sum (x_i y_i)}{\sum [(x_i - \bar{x})^2]}$

"intercept"

$= \frac{\bar{y} [\sum (x_i - \bar{x})^2 + n \bar{x}^2] - \bar{x} [n \bar{x} \bar{y} + \sum (x_i - \bar{x})(y_i - \bar{y})]}{\sum (x_i - \bar{x})^2} = \bar{y} - \bar{x} \frac{\sum (x_i - \bar{x})(y_i - \bar{y})}{\sum (x_i - \bar{x})^2}$

(P5.1, iii)

$a = \bar{y} - b \bar{x}$

(II): $b = \frac{\sum (x_i y_i) - n \bar{x} \bar{y}}{\sum (x_i - \bar{x})^2} = \frac{\sum [(x_i - \bar{x})(y_i - \bar{y})] + n \bar{x} \bar{y} - n \bar{x} \bar{y}}{\sum (x_i - \bar{x})^2} = \frac{\sum [(x_i - \bar{x})(y_i - \bar{y})]}{\sum (x_i - \bar{x})^2}$

"slope"

With the definition of $\tilde{x}_i := x_i - \bar{x}$ & $\tilde{y}_i := y_i - \bar{y}$, we can rewrite: (I)' & (II)':

$a = \bar{y} - \bar{x} \cdot \frac{\sum [\tilde{x}_i \cdot \tilde{y}_i]}{\sum [\tilde{x}_i^2]} = \bar{y} - b \bar{x}$ (I)''

$b = \frac{\sum [\tilde{x}_i \cdot \tilde{y}_i]}{\sum [\tilde{x}_i^2]}$ (II)''

“Least Square Fitting” of a sLR-Line $y = a + b \cdot x$ (4/4)

Corollary (C5.2): “LSF Model based Properties: i.e. center of mass point”

Let $y = a + b \cdot x$ be an optimal SLR line, then we have the following properties:

- i. Let $M(x)$ and $M(y)$ the mean-values for all x_i and $y_i \rightarrow$ the sLR line goes through the *center of mass* (“*Schwerpunkt*”) point = $(M(x), M(y))$ if the model includes an intercept term (i.e. is not forced to go through the origin).
- ii. The sum of errors $e_i = y_i - y(x_i)$ is zero if the model includes an intercept term: $\sum(e_i) = 0$
- iii. Values e_i and x_i are uncorrelated (whether or not there is an intercept term model): $\sum(x_i \cdot e_i) = 0$

Proof:

Part (i): Use the equations of “least best fit” method for sLR, then take the formulas (I) and (II) for the coeff. a and b and set the values into regression line to proof: $y(M(x)) = a + b \cdot M(x) = M(y)$:

(I), (II) $\Rightarrow y(M(x)) = a + b \cdot M(x) = \frac{1}{\det} [(M(y) \cdot \sum(x_i^2) - M(x) \cdot \sum(x_i \cdot y_i))] + \frac{1}{\det} [(\sum(x_i \cdot y_i) - n \cdot M(x) \cdot M(y))] \cdot M(x)$ (1)
 where $\det = \sum(x_i^2) - n \cdot M(x)^2$ (2)

The **red parts** are vanishing, (1) $\Rightarrow y(M(x)) = \frac{M(y)}{\det} [\sum(x_i^2) - n \cdot M(x)^2] = \frac{M(y)}{\det} \cdot \det = M(y)$ using (2).

Part (ii) and (iii): see equation (2) and (3) in *Theorem (T5.2)*

q.e.d.

Remark: With the “center of mass” condition, we have a “quick check” for a line, if it is a candidate for an optimal sLR-model. Since only one of two conditions for the determination of an optimal regression line is fulfilled with the “center of mass” it’s therefore a **necessary** but **not a sufficient** condition (“*..notwendig aber nicht hinreichend*”)

Remark & Theorem to the “SST=SSR+SSE” Condition (1/3)

<https://math.stackexchange.com/questions/709419/prove-sst-sse-ssr>

Sometimes in the literature or in YouTube videos you see the formula: “SST=SSR+SSE” (SSE,SST see slides before, and SSR := Sum((f(xi) – Mean(yi))²). Wikipedia (https://en.wikipedia.org/wiki/Coefficient_of_determination): “... In some cases the total sum of squares equals the sum of the two other sums of squares defined above...”. We can prove that his formula is true, if we have the “optimal” Regression-Line. (Take Care!).

Theorem (T5.2): Let $y(x) = a + bx$ be a regression-line with a =intercept and b =slope, with $b < 0$ then:

$$y(x) \text{ is optimal} \implies SST = SSR + SSE \quad (\text{Equation (E5.1)})$$

Proof: Proof that $SST=SSR+SSE$ under the condition: $y(x)$ is optimal $\iff dR^2/da=0$ and $dR^2/db=0$ for $y(x)$.

Let $\bar{y} := \frac{1}{n} \sum y_i$ ("Mean-value"); $f_i = a + b \cdot x_i$ (0) is a regression-line
 We calculate / transform SST using mathematics ("Analysis"):

$$SST = \sum (y_i - \bar{y})^2 = \sum (y_i - f_i + f_i - \bar{y})^2 = \sum (y_i - f_i)^2 + 2 \sum (y_i - f_i)(f_i - \bar{y}) + \sum (f_i - \bar{y})^2$$

 So we have to proof: (1) $\sum (y_i - f_i)(f_i - \bar{y}) = 0$:= SSE
 $R^2 = \max \iff SSE = \min$; We write: $SSE =: E$; $E = \min \iff \frac{\partial E}{\partial a} = 0 = \frac{\partial E}{\partial b}$
 $0 = \frac{\partial E}{\partial a} = \frac{\partial}{\partial a} \left(\sum (y_i - f_i)^2 \right) \Rightarrow 0 = 2 \cdot \sum (y_i - a - bx_i) \cdot (-1) = -2 \left(\sum (y_i - f_i) \right) \Rightarrow \sum (y_i - f_i) = 0 \quad (2)$
 $0 = \frac{\partial E}{\partial b} = 2 \cdot \sum (y_i - a - bx_i) \cdot (-x_i) \Rightarrow 0 = \sum x_i (y_i - f_i) \stackrel{b \neq 0}{\iff} \sum \left(\frac{f_i - a}{b} - \frac{a}{b} \right) (y_i - f_i)$
 $\Rightarrow \sum f_i (y_i - f_i) = 0 \quad (3) \Rightarrow (1) = 0 \quad = \frac{1}{b} \cdot \sum f_i (y_i - f_i) \stackrel{b \neq 0}{\iff} \sum (y_i - f_i) = \frac{1}{b} \sum f_i (y_i - f_i) \stackrel{=0}{=} 0 \quad (2)$

Details to Theorem(T5.2) see: <https://www.youtube.com/watch?v=-fP7VasT1Oc>

See also: <https://math.stackexchange.com/questions/709419/prove-sst-sse-ssr>

Mathematik1 / Lehr- und Lerneinheiten

Angewandte Mathematik

- Grundlagen der Differential und Integralrechnung reeller Funktionen mit mehreren Veränderlichen sowie von Differentialgleichungen und Differentialgleichungssystemen
- Numerische Methoden und weitere Beispiele mathematischer Anwendungen in der Informatik

Statistik

- Deskriptive Statistik
- Zufallsexperimente, Wahrscheinlichkeiten und Spezielle Verteilungen
- Induktive Statistik
- Anwendungen in der Informatik

Remark & Theorem to the "SST=SSR+SSE" Condition (2/3)

Remark (R5.1):

Whit the statement: "SST = SSR + SSE" (*) you could rewrite $R^2 = 1 - SSE/SST = (SST - SSE)/SST = SSR/SST$ (*). You could now think $R^2 = SSR/SST$ is also a good metric, but this is wrong (see below: example of Homework 5H.1_a).

ATTENTION: In some documents or YouTube videos you see this as metric. **Please don't use this metric !!!**
Example: See Homework (H5.1):

Manuel calculation of two sLR-lines (green, red) (Homework (H5.1_a) + Compare with optimal sLR-line (homework (H5.1_b) + Check Results with the new metric $R^2 = SSR/SST$

Decide what is the "better" sLR-Line: $y = 1.5 + 0.5 \cdot x$ or $y = 1.25 + 0.5 \cdot x$?

Solution:
 Number of Point N=3 Mean-Values ("Mittelwerte"): $[M(x), M(y)] = [2; (7/3)]$
 Set up a table with the quantities included in the above formulas for a and b and also the quantities for the calculation of R^2 :

With the definition $R^2 = SSR/SST$ we get the result that the green line is the best sLR-line of the three -> With $R^2 = 1 - SSE/SST$ it was the yellow line => the red metric is not applicable!

needed for calculation of a and b					Needed for calculation of R^2			SST = SSE + SSR ?		
i	x_i	y_i	$x_i \cdot y_i$	x_i^2	$y(x_i)$	$SSE = \sum(y_i - y(x_i))^2$	$SST = \sum(y_i - M(y))^2$	R^2	$SSR = \sum(y(x_i) - M(y))^2$	R^2
1	1	2	2	1	2,00	0,000000	0,111111		0,111111	
2	3	3	9	9	3,00	0,000000	0,444444		0,444444	
3	2	2	4	4	2,50	0,250000	0,111111		0,027778	
sum	6	7	15	14		0,250000	0,666667	0,625000	0,583333	0,875000

$y = 1.5 + 0.5 \cdot x$
 $y = 1.25 + 0.5 \cdot x$

Which estimation (red or green) is better?

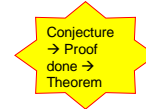
needed for calculation of R^2					SST = SSE + SSR ?	
$y(x_i)$	$SSE = \sum(y_i - y(x_i))^2$	$SST = \sum(y_i - M(y))^2$	R^2	$SSR = \sum(y(x_i) - M(y))^2$	R^2	
1,75	0,0625000	0,111111		0,3402778		
2,75	0,0625000	0,444444		0,1736111		
2,25	0,0625000	0,111111		0,0068444		
0,1875000	0,6666667	0,7187500	0,7187500	0,2068333	0,7812500	

From Homework (H5.1_b) we get the data for the "optimal" sLR-line:

needed for calculation of R^2					SST = SSE + SSR ?	
$y(x_i)$	$SSE = \sum(y_i - y(x_i))^2$	$SST = \sum(y_i - M(y))^2$	R^2	$SSR = \sum(y(x_i) - M(y))^2$	R^2	
13/6	$(1/6)^2 = 1/36$	$(1/3)^2 = 1/9$		1/4		
17/6	$(1/6)^2 = 1/36$	$(2/3)^2 = 4/9$		3/4		
14/6	$(-2/6)^2 = 1/9$	$(1/3)^2 = 1/9$		0		
42/6=7	0/36=0	2/3	0,7500000	1/2	0,7500000	

Remark & Theorem to the “SST=SSR+SSE” Condition (3/3)

I guess the following conjecture. It is a conjecture, since I can't prove it so far.



Conjecture (Conj5.1): “SST=SSR+SSE” is not sufficient for an optimal sLR-line
 (“...necessary but not sufficient...”). This would prove that the direction “ \Leftarrow ” of Theorem (T5.2) is wrong !!

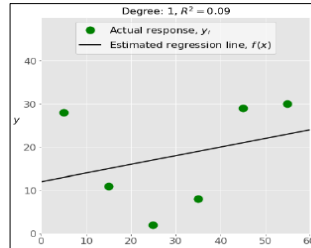
Idea of a proof:

If you do not believe that “ \Leftarrow ” is correct, you can for example refute this by constructing a counterexample. This would mean you have to find a *Trainings-Set TS*, with a sLR-line with (SST=SST+SSE) which is not optimal. To check this, is a sub-task in Homework (H5.5)*.

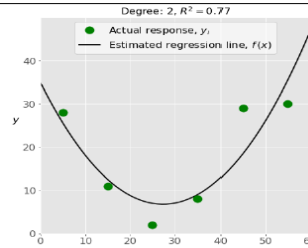
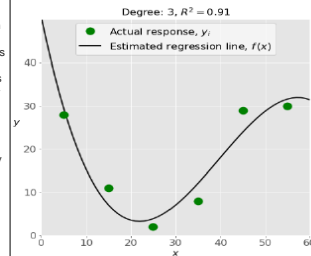
***** see Homework (H5.5)* *****

Plots of underfitted, well-fitted, and overfitted Models

This plot shows a linear regression line that has a low R^2 . It might also be important that a straight line can't take into account the fact that the actual response increases as x moves away from 25 towards zero. This is likely an example of **underfitting**.

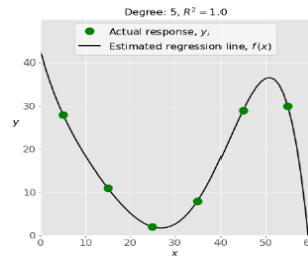


This plot presents polynomial regression with the degree equal to 3. The value of R^2 is higher than in the preceding cases. This model behaves better with known data than the previous ones. However, it shows some signs of **overfitting**, especially for the input values close to 60 where the line starts decreasing, although actual data don't show that.



This plot illustrates polynomial regression with the degree equal to 2. In this instance, this might be the optimal degree for modeling this data. The model has a value of R^2 that is satisfactory in many cases and **shows trends nicely (well-fitted)**.

In this plot, you may see the perfect fit: six points and the polynomial line of the degree 5 (or higher) yield $R^2 = 1$. Each actual response equals its corresponding prediction. In some situations, this might be exactly what you're looking for. In many cases, however, this is an **overfitted model**. It is likely to have poor behavior with unseen data, esp. with the inputs larger than 50. For example, it assumes, without any evidence, that there is a significant drop in responses for $x > 50$ and that y reaches zero for x near 60. Such behavior is the consequence of excessive effort to learn and fit the existing data.



Status: 1. Dezember 2020

Page
136

Underfitting and Overfitting

Underfitting occurs when a model can't accurately capture the dependencies among data, usually as a consequence of its own simplicity. It often yields a low R^2 with known data and bad generalization capabilities when applied with new data.

Overfitting happens when a model learns both dependencies among data and random fluctuations. In other words, a model learns the existing data too well.

Complex models, which have many features or terms, are often prone to overfitting. When applied to known data, such models usually yield high R^2 . However, they often don't generalize well and have significantly lower R^2 when used with new data.

Polynomial Regression: You can regard this as a generalized case of linear regression. In other words, in addition to linear terms like b_1x_1 , your regression function f can include non-linear terms such as $b_2x_1^2$, $b_3x_1^3$, or even $b_4x_1x_2$, $b_5x_1^2x_2$, and so on. In the case of two variables and the polynomial of degree 2, the regression function has this form: $f(x_1, x_2) = b_0 + b_1x_1 + b_2x_2 + b_3x_1^2 + b_4x_1x_2 + b_5x_2^2$. The simplest example of such estimated regression function is a polynomial of degree 2: $f(x) = b_0 + b_1x + b_2x^2$.

One very important question that might arise when you're implementing polynomial regression is related to **the choice of the optimal degree of the polynomial regression function**. There is no straightforward rule for doing this. It depends on the case. You should, however, be aware of two problems that might follow the choice of the degree: **underfitting** and **overfitting**.

simple Linear Regression (sLR) with Scikit-learn (1/3)

Following ideas from: "Linear Regression in Python" by Mirko Stojiljkovic, 28.4.2020 (see details: <https://realpython.com/linear-regression-in-python/#what-is-regression>)

There are **five basic steps** when you're implementing linear regression:

1. Import the packages and classes you need.
2. Provide data to work with and eventually do appropriate transformations.
3. Create a regression model and fit it with existing data.
4. Check the results of model fitting to know whether the model is satisfactory.
5. Apply the model for predictions. These steps are more or less general for most of the regression approaches and implementations.

Step 1: Import packages and classes

The first step is to import the package `numpy` and the class `LinearRegression` from `sklearn.linear_model`.

```
In [1]: # Step 1: Import packages and classes
import numpy as np
from sklearn.linear_model import LinearRegression
```

Now, you have all the functionalities you need to implement linear regression.

The fundamental data type of NumPy is the array type called `numpy.ndarray`. The rest of this article uses the term `array` to refer to instances of the type `numpy.ndarray`.

The class `sklearn.linear_model.LinearRegression` will be used to perform linear and polynomial regression and make predictions accordingly.

Scikit-learn (formerly **scikits.learn** and also known as **sklearn**) is a [free software machine learning library](#) for the [Python programming language](#).^[3] It features various [classification](#), [regression](#) and [clustering](#) algorithms including [support vector machines](#), [random forests](#), [gradient boosting](#), [k-means](#) and [DBSCAN](#), and is designed to interoperate with the Python numerical and scientific libraries [NumPy](#) and [SciPy](#).

See also: “scikit-learn *Machine Learning in Python*”: <https://scikit-learn.org/stable/>

Complete solution you will find under:

<https://github.com/HVoellinger/Lecture-Notes-to-ML-WS2020>

“sLR Example_Lecture-ML5.ipynb”

simple Linear Regression (sLR) with Scikit-learn (2/3)

Step 2: Provide data

The second step is defining data to work with. The inputs (regressors, x) and output (predictor, y) should be arrays (the instances of the class `numpy.ndarray`) or similar objects. This is the simplest way of providing data for regression.

```
In [2]: # Step 2: Provide data
x = np.array([ 5, 15, 25, 35, 45, 55]).reshape((-1, 1))
y = np.array([ 5, 20, 14, 32, 22, 38])
```

Now, you have two arrays: the input x and output y . You should call `reshape()` on x because this array is required to be two-dimensional, or to be more precise, to have one column and as many rows as necessary. That's exactly what the argument `(-1, 1)` of `reshape()` specifies.

```
In [3]: print ("This is how x and y look now:")
print("x=",x)
print("y=",y)
```

Step 3: Create a model and fit it

The next step is to create a linear regression model and fit it using the existing data. Let's create an instance of the class `LinearRegression`, which will represent the regression model.

```
In [4]: model = LinearRegression()
```

This statement creates the variable `model` as the instance of `LinearRegression`. You can provide several optional parameters to `LinearRegression`:

----> `fit_intercept` is a Boolean (True by default) that decides whether to calculate the intercept b_0 (True) or consider it equal to zero (False).

----> `normalize` is a Boolean (False by default) that decides whether to normalize the input variables (True) or not (False).

----> `copy_X` is a Boolean (True by default) that decides whether to copy (True) or overwrite the input variables (False).

----> `n_jobs` is an integer or None (default) and represents the number of jobs used in parallel computation. None usually means one job and -1 to use all processors.

This example uses the default values of all parameters.

It's time to start using the model. First, you need to call `fit()` on model:

```
In [5]: model.fit(x, y)
Out[5]: LinearRegression(copy_X=True, fit_intercept=True, n_jobs=None, normalize=False)
```

simple Linear Regression (sLR) with Scikit-Learn (3/3)

Step 4: Get results

Once you have your model fitted, you can get the results to check whether the model works satisfactorily and interpret it.

You can obtain the coefficient of determination (R^2) with `score()` called on model:

```
In [7]: r_sq = model.score(x, y)
print('coefficient of determination:', r_sq)
coefficient of determination: 0.7158756137479542
```

When you're applying `score()`, the arguments are also the predictor x and regressor y , and the return value is R^2 .

The attributes of model are `intercept_` which represents the coefficient b_0 and `coef_` which represents b_1 :

```
In [8]: print('intercept:', model.intercept_)
print('slope:', model.coef_)
intercept: 5.6333333333333329
slope: [0.54]
```

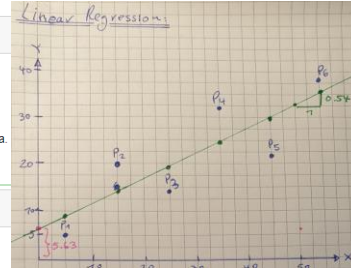
Step 5: Predict response

Once there is a satisfactory model, you can use it for predictions with either existing or new data.

To obtain the predicted response, use `predict()`:

```
In [10]: y_pred = model.predict(x)
print('predicted response:', y_pred, sep='\n')
predicted response:
[ 8.33333333 13.73333333 19.13333333 24.53333333 29.93333333 35.33333333]
```

When applying `predict()`, you pass the regressor as the argument and get the corresponding predicted response.

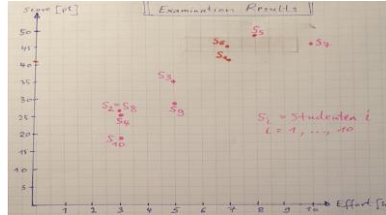


Examples of sLR-lines: $y = a + b \cdot x$ (1/2)

Example (E5.1): Build the SLR (x,y)-Model "Students Examination Results"

Part a: Analyze the y ="Achieved points of the exam [%]" depending on the parameter: x ="Effort exam preparation [h]" Find "least square fit" $y = b_0 + b_1x$ with $\{(exam\ prep.[h],\ score[pt.])\} = \{(7, 41), (3, 27), (5, 35), (3, 26), (8, 48), (7, 45), (10, 46), (3, 27), (5, 29), (3, 19)\}$

needed for calculation of b0 and b1					
student i	exam prep. x_i	points y_i	$x_i \cdot y_i$	x_i^2	grade
1	7	41	287	49	2,0
2	3	27	81	9	3,7
3	5	35	175	25	2,8
4	3	26	78	9	3,8
5	8	48	384	64	1,2
6	7	45	315	49	1,6
7	10	46	460	100	1,4
8	3	27	81	9	3,7
9	5	29	145	25	3,5
10	3	19	57	9	4,7
sum	54	343	2063	348	2,8



Part b: Do the same with a new x ="Effort for homework[h]": $\{(homework[h],\ score[pt.])\} = \{(5, 41), (4, 27), (5, 35), (3, 26), (9, 48), (8, 45), (10, 46), (5, 27), (3, 29), (3, 19)\}$

Task: Build the model sLR(x,y). Compare and check your result with the output of a Python-Program. Answer the following questions:

- Q1: How much points would a student achieve without any preparation or without doing any homework?
- Q2: How much points would a student achieve with (10 hours of preparation for the exam) or (10 hours homework)?
- Q3: How much effort you will need to reach enough points (=25) to get enough points to pass the exam?

Additional Question/Remark: Our calculation use one of the two variables independent from the other variable. What is the difference to mLR-model results. Is R^2 (calculate here) different from $Adj.R^2$ we got in *Example (E5.3)*?

Example (E5.1)-First Part: $\{(exam\ prep.[h],\ score[pt.])\} = \{(7, 41), (3, 27), (5, 35), (3, 26), (8, 48), (7, 45), (10, 46), (3, 27), (5, 29), (3, 19)\}$

needed for calculation of b0 and b1				
student i	exam prep. x_i	points y_i	$x_i \cdot y_i$	x_i^2
1	7	41	287	49
2	3	27	81	9
3	5	35	175	25
4	3	26	78	9
5	8	48	384	64
6	7	45	315	49
7	10	46	460	100
8	3	27	81	9
9	5	29	145	25
10	3	19	57	9
sum	54	343	2063	348

Second Part: similar to first part build the above table with the new data: $\{(homework[h],\ score[pt.])\} = \{(5, 41), (4, 27), (5, 35), (3, 26), (9, 48), (8, 45), (10, 46), (5, 27), (3, 29), (3, 19)\}$

Examples of sLR-lines: $y = a + b \cdot x$ (2/2)

Solution of E5.1-a:

Student i	needed for calculation of b0 and b1				needed for calculation of R ²		
	exam score y_i	points x_i	$x_i \cdot y_i$	x_i^2	y_i^2	$SSE = \sum (y_i - \hat{y}_i)^2$	$SST = \sum (y_i - \bar{y})^2$
1	7	41	287	1681	49	0.314181	44.89
2	3	27	81	729	9	2.809475	51.29
3	5	35	175	1225	25	32.797	4.89
4	8	26	208	676	64	4.83209	68.89
5	8	48	384	2304	64	15.98025	187.69
6	7	65	455	4225	49	22.38261	114.49
7	10	46	460	2116	100	51.477	29.97529
8	9	27	243	729	81	2.809475	51.29
9	5	29	145	841	25	32.797	14.417209
10	3	19	57	361	9	25.925	234.09
Mean	54	343	2063	148		134.217184	922.1

Substitute these values into Formula 1 and 11:

$b_0 = (148 \cdot 34.3 - 54 \cdot 2063) / (343 \cdot 10 - 54^2) = (50862 - 111402) / (3430 - 2916) = -60540 / 514 = -117.78$

$b_1 = (2063 - 10 \cdot 34.3) / (2825 - (1054/5) / (282/5)) = (2063 - 343) / (2825 - 187.2) = 1720 / 2637.8 = 0.65$

→ Regression-Line: $y = -117.78 + 0.65 \cdot x$

$R^2 = 1 - \frac{\sum (y_i - \hat{y}_i)^2}{\sum (y_i - \bar{y})^2} = 1 - \frac{134.2172/922.1}{0.8544} = 0.8544$

Question 1: 14,117 points.
Question 2: 14,117 + 37,58 = 51,497 = 50 points (grade = 1,0).

Check of Proposition (P5.11): $\hat{f}(Mean(x)) = -117.78 + 0.65 \cdot 34.3 = 22.914 \approx \text{Mean}(y)$ (o.e.d.)

Complete Solution: see the file *LR-Calculation_of_Coeff.xlsx* in my GitHub [HVö-6]: https://github.com/HVoellinger/Lecture-Notes-to-ML-WS2020/blob/master/LR-Calculation_of_Coeff.xlsx
Solution of E5.1-b: see also the above GitHub link.
 Q1: 14,117 and 15,164; Q2: in both cases 50 points, which is a grade of 1,0. Q3: 2,91[h] and 2,83[h]

Example (E5.2): “Manual calculation for sLR-model with Iowa Houses Data”
 Take a subset of 10 data-records and calculate manually a, b and R² using the matrices of the lesson. Compare your result with the results of the Homework ML5.3: “Coding with the dataset “Iowa Homes” to predict the “House Price” based on “Square Feet” and a second variable (i.e. “Age of Home”). Compare your result with the output of a Python-Program. See the same link to GitHub as above in E5.1

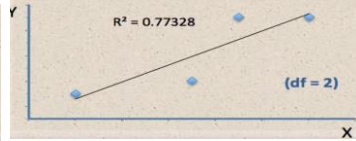
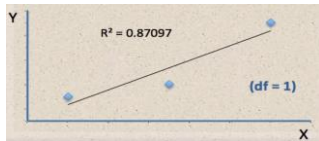
Example (E5.2): See Homework H5.3

Definition of the multiple Linear Regression Function (k=2)

$$Y_i = B_0 + B_1X_i + \epsilon_i$$

Q: What is the minimum number of observations required to estimate this regression?

Answer = 2

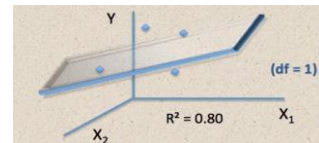


df :=Degrees of Freedom = Number of observations - 2

$$Y_i = B_0 + B_1X_{1i} + B_2X_{2i} + \epsilon_i$$

Q: What is the minimum number of observations required to estimate this regression?

Answer = 3



$$df = n - k - 1$$

k = the number of explanatory (X) variables

n := Number of observations
k := Number of variables

Multiple or multivariate linear regression is a case of linear regression with two or more independent variables.

If there are just two independent variables, the estimated regression function is $f(x_1, x_2) = b_0 + b_1x_1 + b_2x_2$. It represents a regression plane in a three-dimensional space. The goal of regression is to determine the values of the weights b_0 , b_1 , and b_2 such that this plane is as close as possible to the actual responses.

The case of more than two independent variables is similar, but more general. The estimated regression function is $f(x_1, \dots, x_r) = b_0 + b_1x_1 + \dots + b_rx_r$, and there are $r + 1$ weights to be determined when the number of inputs is r .

See also the YouTube Video: "Regression II: Degrees of Freedom EXPLAINED | Adjusted R-Squared" : <https://www.youtube.com/watch?v=4otEcA3gjLk>

Consider also the YouTube video from Andrew Ng: "Lecture 4.1 - Linear Regression With Multiple Variables" and following Lectures 4.x:
https://www.youtube.com/watch?v=Q4GNLhRtZNc&list=PLLsT5z_DsK-h9vYZkQkYNWcltqhIRJLN&index=18

Definition of the “Adjusted R-squared” Measure

Question: How does degrees of freedom relate to R-squared?

As df decreases, (ie. more variables added to a given model) R-squared will ONLY increase

Adjusted $\bar{R}^2 = \left\{ \begin{array}{l} 1 - (1 - R^2) \frac{n - 1}{n - k - 1} \\ \text{OR} \\ 1 - \left(\frac{SSE}{SST} \right) \frac{n - 1}{n - k - 1} \end{array} \right.$

as k increases, Adj \bar{R}^2 will tend to decrease, reflecting the reduced power in the model.

Rule: if $K > 1$ choose the regression model where **Adj. \bar{R}^2 is maximum**

R^2 only works as intended in a **simple linear regression** model with one explanatory variable. With multiple regression made up of several independent variables ($k > 1$), the R-Squared must be adjusted.

The **adjusted R^2** compares the descriptive power of regression models that include diverse numbers of predictors. Every predictor added to a model increases R^2 and never decreases it. Thus, a model with more terms may seem to have a better fit just for the fact that it has more terms, while the adjusted R-squared compensates for the addition of variables and only increases if the new term enhances the model above what would be obtained by probability and decreases when a predictor enhances the model less than what is predicted by chance.

➔ **Rule:** if $K > 1$ choose the regression model where **Adj. \bar{R}^2 is max.**

In an overfitting condition, an incorrectly high value of R^2 , which leads to a decreased ability to predict, is obtained. This is not the case with the adjusted R^2 .

For more **detailed interpretation** about the differences of R^2 and Adj.- R^2 , see also the YouTube Video: “Regression II: Degrees of Freedom EXPLAINED | Adj. R^2 ”: <https://www.youtube.com/watch?v=4otEcA3gjLk>

Properties and Remarks of Adj.R² Measure

Remark (R5.2):

Adj.R² can only be defined when $n-k-1 < 0$ which is same as $k < n-1$.

Also it is clear that the *number of observation-points*(n) $\geq k+2$. If this is not the case, you can't calculate a meaningful Regression-Hyperplane HP. We say "without loss of generality" (w.l.o.g.): $k \leq n-2$.

Theorem (T5.3): "Properties of Adj.R²"

- (i): Adj.R² $\leq R^2 \leq 1$ "Limitation"
 (ii): Adj.R² can become negative "Negativity"

Proof:

(i)
$$\begin{aligned} \text{Adj.R}^2 &= 1 - (1 - R^2) \frac{(n-1)}{(n-1-k)} \\ &\leq 1 - (1-R^2)(n-1) \quad \text{by Remark R5.2: } n-1-k \geq n-1-n+2=1, \text{ which shows } 1/(n-k-1) \leq 1 \Rightarrow (n-1)/(n-1-k) \leq n-1 \\ &= 1 + (1-R^2)(-1)(n-1) \\ &\leq 1 + (1-R^2)(-1) \quad \text{by } n-1 \geq 1 \text{ (} n \geq 2 \text{) we see } -(n-1) \leq -1 \\ &= 1 - 1 + R^2 = R^2 \leq 1 \text{ (by T5.1)} \end{aligned}$$

- (ii) In Homework H5.3 we see for $n=10, k=8: R^2 \sim 0,8$
 Then we see for $\text{Adj.R}^2 = 1 - (1-R^2)(9/1) \sim 1 - (0,2) \cdot 9 = 1 - 1,8 = -0,8$

“Least Square Fitting” for mLR(k=2): $z = a + b \cdot x + c \cdot y$ (1/4)

<https://www.youtube.com/watch?v=XN0yKLRPIQQ>

Task: Calculate for the "regression/plane" $f(x,y) = z = a + bx + cy$ the coefficients a, b and c ; $R^2 = 1 - \frac{SSE}{SST}$; $\bar{x} = \frac{1}{n} \sum x_i$; $\bar{y} = \frac{1}{n} \sum y_i$; $\bar{z} = \frac{1}{n} \sum z_i$; $P_i = (x_i | y_i | z_i)$; $\hat{P}_i = (x_i | y_i | f_i(x,y))$; $f(x,y) = a + bx + cy$

Solution: $R^2 = \text{maximal} \Leftrightarrow SSE = S$ is minimal
 $S = \text{minimal} \Leftrightarrow dS = \frac{\partial S}{\partial a} \cdot da + \frac{\partial S}{\partial b} \cdot db + \frac{\partial S}{\partial c} \cdot dc = 0$
 $(1) 0 = \frac{\partial S}{\partial a} = \frac{\partial}{\partial a} \left[\sum (z_i - f_i(x,y))^2 \right] = 2 \cdot \sum (z_i - a - bx_i - cy_i) \cdot (-1)$
 $\Rightarrow 0 = \sum z_i - n \cdot a - \sum bx_i - c \sum y_i \Rightarrow n \bar{z} = na + b \bar{x} + c \bar{y}$
 $\Rightarrow a = \bar{z} - b \bar{x} - c \bar{y}$ (I)

Now the "trick" is, to substitute formula (I) into the derivations $\frac{\partial S}{\partial b}$ and $\frac{\partial S}{\partial c}$. We reach in this case that we have only to invert a 2×2 -Matrix (instead a 3×3 -Matrix)

(2) $0 = \frac{\partial S}{\partial b} = \frac{\partial}{\partial b} \left(\sum (\hat{z}_i - b \hat{x}_i - c \hat{y}_i)^2 \right)$
 $= \frac{\partial}{\partial b} \left(\sum (\hat{z}_i - b \hat{x}_i - c \hat{y}_i) \cdot (-\hat{x}_i) \right)$
 $= -2 \cdot \sum (\hat{z}_i - b \hat{x}_i - c \hat{y}_i) \cdot \hat{x}_i$
 $= -2 \cdot \sum (\hat{z}_i \cdot \hat{x}_i - b \hat{x}_i^2 - c \hat{y}_i \hat{x}_i)$
 $\Rightarrow 0^{(1)} = \sum (\hat{x}_i \hat{z}_i - b \hat{x}_i^2 - c \hat{x}_i \hat{y}_i)$
 Analog $\frac{\partial S}{\partial c} = 0 \Rightarrow 0^{(2)} = \sum (\hat{y}_i \hat{z}_i - b \hat{x}_i \hat{y}_i - c \hat{y}_i^2)$

We define:
 $\hat{z}_i := z_i - \bar{z}$
 $\hat{x}_i := x_i - \bar{x}$
 $\hat{y}_i := y_i - \bar{y}$ (D3)

“Least Square Fitting” for mLR(k=2): $z = a + b \cdot x + c \cdot y$ (2/4)

We can rewrite this with a matrix:

$$\begin{pmatrix} \sum \hat{x}_i^2 & \sum \hat{x}_i \hat{y}_i \\ \sum \hat{x}_i \hat{y}_i & \sum \hat{y}_i^2 \end{pmatrix} \begin{pmatrix} b \\ c \end{pmatrix} = \begin{pmatrix} \sum \hat{x}_i \hat{z}_i \\ \sum \hat{y}_i \hat{z}_i \end{pmatrix}$$

By Linear Algebra we know, that: $A^{-1} = \frac{1}{\det A} \begin{pmatrix} d & -c \\ -b & a \end{pmatrix}$ (3)

2×2 Matrix (symmetric) $\} z = \begin{pmatrix} a & b \\ c & d \end{pmatrix}$

$$\Rightarrow \begin{pmatrix} b \\ c \end{pmatrix} = \frac{1}{\det} \begin{pmatrix} \sum \hat{y}_i^2 & -\sum \hat{x}_i \hat{y}_i \\ -\sum \hat{x}_i \hat{y}_i & \sum \hat{x}_i^2 \end{pmatrix} \cdot \begin{pmatrix} \sum \hat{x}_i \hat{z}_i \\ \sum \hat{y}_i \hat{z}_i \end{pmatrix}; \det = \sum \hat{x}_i^2 \sum \hat{y}_i^2 - (\sum \hat{x}_i \hat{y}_i)^2$$

Summary

$$b_0 = \bar{z} - b_1 \bar{x} - b_2 \bar{y}$$

$$b_1 = \frac{(\sum \hat{x}_i^2)(\sum \hat{y}_i \hat{z}_i) - (\sum \hat{x}_i \hat{y}_i)(\sum \hat{y}_i \hat{z}_i)}{(\sum \hat{x}_i^2)(\sum \hat{y}_i^2) - (\sum \hat{x}_i \hat{y}_i)^2}$$

$$b_2 = \frac{(\sum \hat{x}_i \hat{y}_i)(\sum \hat{x}_i \hat{z}_i) - (\sum \hat{x}_i^2)(\sum \hat{y}_i \hat{z}_i)}{(\sum \hat{x}_i^2)(\sum \hat{y}_i^2) - (\sum \hat{x}_i \hat{y}_i)^2}$$

$$\alpha = \bar{z} - b \bar{x} - c \bar{y}$$

$$b = (\sum \hat{y}_i^2 \cdot \sum \hat{x}_i \hat{z}_i - \sum \hat{x}_i \hat{y}_i \cdot \sum \hat{y}_i \hat{z}_i) / \det$$

$$c = (\sum \hat{x}_i^2 \cdot \sum \hat{y}_i \hat{z}_i - \sum \hat{x}_i \hat{y}_i \cdot \sum \hat{x}_i \hat{z}_i) / \det$$

Merkregel: "(3) (iii) Vereinfacht"

$$(3) : \begin{pmatrix} \text{II} \\ \text{III} \end{pmatrix} = \begin{pmatrix} a & b \\ b & c \end{pmatrix} \begin{pmatrix} A \\ B \end{pmatrix} \Leftrightarrow \begin{pmatrix} \text{II} \\ \text{III} \end{pmatrix} = a \cdot A + b \cdot B$$

$$\begin{pmatrix} \text{II} \\ \text{III} \end{pmatrix} = c \cdot B + b \cdot A$$

$\hat{x}_i := x_i - \bar{x}$
 $\hat{y}_i := y_i - \bar{y}$
 $\hat{z}_i := z_i - \bar{z}$
 $\bar{x} := \frac{1}{n} \sum x_i$
 $\bar{y} := \frac{1}{n} \sum y_i$
 $\bar{z} := \frac{1}{n} \sum z_i$

<https://github.com/HVoellinger/Lecture-Notes-to-ML-WS2020/blob/master/QYulc.gif>

“Least Square Fitting” for mLR k=2): $z = a + b*x + c*y$ (3/4)

Solution:

Number of Point N=8 Mean-Values ("Mittelwerte") = : [M(x),M(y),M(z)] ~ [240/8, 104/8, 178/8] = [30; 13; 22,25]
Set up a table with the quantities included in the above LSF formulas (I) and (II) for simple LR:

needed for the calculation of a, b and c												
i	x _i	y _i	z _i	X _i :=x _i -M(x)	Y _i :=y _i -M(y)	Z _i :=z _i -M(z)	X _i *Y _i	X _i *Z _i	Y _i *Z _i	X _i ²	Y _i ²	
1	0	1	4	-30	-12	-18,25	360	547,50	219,00	900	144	
2	5	1	5	-25	-12	-17,25	300	431,25	207,00	625	144	
3	15	2	20	-15	-11	-2,25	165	33,75	24,75	225	121	
4	25	5	14	-5	-8	-8,25	40	41,25	66,00	25	64	
5	35	11	32	5	-2	9,75	-10	48,75	-19,50	25	4	
6	45	15	22	15	2	-0,25	30	-3,75	-0,50	225	4	
7	55	34	38	25	21	15,75	525	393,75	330,75	625	441	
8	60	35	43	30	22	20,75	660	622,50	456,50	900	484	
Sum	240	104	178	0	0	0,00	2070	2115,00	1284,00	3550	1406	

Substitute the values to the formulas (I) and (II) of LSF for mLR:

det = sum(X_i²)*sum(Y_i²)-(sum(X_i*Y_i))²=3550*1406-(2070)²=706400
a = Mean(z)-b*Mean(x)-c*Mean(y) ~ 22,25-0,4471*30 + 0,2550*13 ~ 5,522
b = (1/det)*(sum(Y_i²)*sum(X_iZ_i) - sum(X_iY_i)*sum(Y_iZ_i))=(1/det)*(1406*2115-2070*1284) ~ 0,4471
c = (1/det)*(sum(X_i²*sum(Y_iZ_i)- sum(X_iY_i)*sum(X_iZ_i))=(1/det)*(3550*1284-2070*2115) ~ 0,2550

Compare with Python-Pgm (next slides):
intercept: 5.52257927519819
coefficients: [0.44706965 0.25502548]

So we get the optimal mLR line: $z = 5,522 + 0,4471*x + 0,255*y$

R² = 1 - SSE/SST ~ 0,86159 (details see notes-page)

coefficient of determination: 0.8615939258756776

--> Adj.R²= 1 -(1-R²)*(7/5) ~ 0,8062 (details see notepage)

Set up a table with the quantities included in the formulas for R² and Adj.R²:

needed for calculation of R ²			SST = SSE + SSR ?
z(x _i , y _i)	SSE=sum(z _i -z(x _i , y _i)) ²	SST=sum(z _i -M(z)) ²	SSR=sum(z(x _i , y _i)-M(z)) ²
5,777	3,1577	333,0625	271,3597
8,0125	9,0752	297,5625	202,7064
12,7385	52,7294	5,0625	90,4686
17,9745	15,7967	68,0625	18,2799
23,9755	64,3926	95,0625	2,9774
29,4665	55,7486	0,0625	52,0779
38,7825	0,6123	248,0625	273,3236
41,273	2,9825	430,5625	361,8745
	204,4950	1477,5000	1273,0680

sse+ssr: 1477,5630

R² = 1-SSE/SST = 1 - Sum((z_i-z(x_i, y_i))²)/Sum((z_i-M(z))²) = 1 -(204,495/1477,5) ~ 0,86159

--> Adj.R²= 1 -(1-R²)*(7/5) ~ 0,8062

“Least Square Fitting” for mLR(k=2): $z = a + b \cdot x + c \cdot y$ (4/4)

Corollary (C5.3): “mLR(k=2) -Center of mass point”

Let $z = a + b \cdot x + c \cdot y$ be an optimal mLR(k=2)-plane, then we have the following properties:

Let $M(x)$, $M(y)$ and $M(z)$ the mean-values for all x_i , y_i and $z_i \rightarrow$ The *center of mass point* (“Schwerpunkt”) = $(M(x), M(y), M(z))$ is on the mLR(k=2)-plane.

Proof: The statement follows directly for the LSF calculation, formula (I).

Examples for mLR(k=2) Models: $z = a + b \cdot x + c \cdot y$ (1/2)

Example (E5.3): Calculation of mLR(k=2)-model for "Students Examination Results"

Similar to *Example (E5.1)*: Find "least square fit" $z = a + b \cdot x + c \cdot y$ for the z :="Achieved points(score) of exam[pt.]" depending on the two parameter: x :="Effort exam preparation[h]" and y :="Effort for homework [h]". Data from *Training Set TS* = $\{(x, y, z) \mid (\text{exam prep.}[h], \text{homework}[h], \text{score}[pt.])\} = \{(7,5;41), (3,4;27), (5,5;35), (3,3;26), (8,9;48), (7,8;45), (10,10;46), (3,5;27), (5,3;29), (3,3;19)\}$

Task: Build to the model mLR(x,y; z). Compare and check your result with the output of a Python-Program.

Answer the following three Questions:

- Q1: How much points would a student achieve without any preparation and without doing any homework?
- Q2: How much points would a student achieve with (10 hours of preparation for the exam) and (10 hours homework) ?
- Q3: How much effort you will need to reach enough points (=25) to score enough points to pass the exam?

Add. Question/Remark: Our calculation use both variables in the calculation. What is the difference to our sLR-model results? Compare Adj.R² (calculated here) to the two R² you got in *Example (E5.1)*.

Solution: see https://github.com/HVoellinger/Lecture-Notes-to-ML-WS2020/blob/master/LR-Calculation_of_Coeff.xlsx

Step 4: Get results

You can obtain the properties of the model the same way as in the case of simple linear regression:

```
In [4]: r_sq = model.score(x, y)
print('coefficient of determination:', r_sq)
print('intercept:', model.intercept_)
print('coefficients:', model.coef_)

coefficient of determination: 0.8842531690055735
intercept: 13.264053254437865
coefficients: [2.48754931 1.38239645]
```

Set up a table with the quantities included in the formulas for R² and Adj.R²:

needed for calculation of R ²			SST = SSE + SSR ?
$z(x_i, y_i)$	$SSE = \sum (z_i - z(x_i, y_i))^2$	$SST = \sum (z_i - M(z))^2$	$SSR = \sum (z(x_i, y_i) - M(z))^2$
37,59	11,6281	44,8900	10,8241
26,256	0,5535	53,2900	64,7059
32,614	5,6930	0,4900	2,8426
24,874	1,2679	68,8900	88,8495
45,606	5,7312	187,6900	127,8256
41,736	10,6537	114,4900	55,2941
51,964	35,5693	136,8900	312,0169
27,638	0,4070	53,2900	44,3822
29,85	0,7225	28,0900	19,8025
24,874	34,5039	234,0900	88,8495
	106,7302	922,1000	815,3930

sse+ssr: 922,1231

$$R^2 = 1 - SSE/SST = 1 - \frac{\sum (z_i - z(x_i, y_i))^2}{\sum (z_i - M(z))^2} = 1 - \frac{106,7302}{922,1} \sim 0,88425$$

$$\rightarrow \text{Adj.}R^2 = 1 - (1 - R^2) \cdot (9/7) \sim 0,8512$$

Examples for mLR(k=2) Models: $z = a + b \cdot x + c \cdot y$ (2/2)

1. Q1: How much points would a student achieve without any preparation and without doing any homework? 13,264 points.
2. Q2: How much points would a student achieve with (10 hours of preparation for the exam) and (10 hours homework)? 50 points ---> score = 1,0.
3. Q3: How much effort you will need to reach enough points (=25) to score enough points to pass the exam?
Consider the cases: $x=0 \rightarrow y=(25-13,264)/1,382=8,49[h]$; $y=0 \rightarrow x=(25-13,264)/2,488=4,72[h]$. When you do 3 hours for both tasks, you will reach $z = 13,264 + 2,488 \cdot 3 + 1,382 \cdot 3 = 24,874 - 25$ points. So you would pass the examination.

Example (E5.4): "Manual calculation for mLR(k=2)-model with Iowa Houses Data"

Similar to example (E5.2) we take a subset of 10 data-records and calculate manually a, b and $\text{Adj. } R^2$ using the matrices of the lesson. Compare your result with the results of Homework: "Coding with the dataset "Iowa Homes" to predict the "House Price" based on "Square Feet" and a second variable (i.e. "Age of Home"). Compare your result with the output of a Python-Program.

multiple Linear Regression (mLR) with Scikit-learn (1/2)

Following ideas from: "Linear Regression in Python" by Mirko Stojilkovic, 28.4.2020 (see details: <https://realpython.com/linear-regression-in-python/#what-is-regression>) you can obtain the properties of the model the same way as in the case of simple linear regression in **five basic steps** (see before):

Steps 1 and 2: Import packages and classes, and provide data

First, you import numpy and sklearn.linear_model.LinearRegression and provide known inputs and output:

```
In [2]: # First, you import numpy and sklearn.linear_model.LinearRegression and
# provide known inputs and output.

import numpy as np
from sklearn.linear_model import LinearRegression

# That's a simple way to define the input x and output y.

x = [[0, 1], [5, 1], [15, 2], [25, 5], [35, 11], [45, 15], [55, 34], [60, 35]]
y = [4, 5, 20, 14, 32, 22, 38, 43]
x, y = np.array(x), np.array(y)
```

Step 3: Create a model and fit it

The next step is to create the regression model as an instance of LinearRegression and fit it with .fit():

```
In [5]: # The result of this statement is the variable model referring to the object of type LinearRegression.
# It represents the regression model fitted with existing data.

model = LinearRegression().fit(x, y)
```

Scikit-learn (formerly **scikits.learn** and also known as **sklearn**) is a [free software machine learning library](#) for the [Python programming language](#).^[3] It features various [classification](#), [regression](#) and [clustering](#) algorithms including [support vector machines](#), [random forests](#), [gradient boosting](#), [k-means](#) and [DBSCAN](#), and is designed to interoperate with the Python numerical and scientific libraries [NumPy](#) and [SciPy](#).

See also: “scikit-learn **Machine Learning in Python**”: <https://scikit-learn.org/stable/>

Complete solution you will find under:

<https://github.com/HVoellinger/Lecture-Notes-to-ML-WS2020>
 “mLR - Example for ML05.ipynb”

multiple Linear Regression (mLR) with Scikit-learn (2/2)

Step 4: Get results

You can obtain the properties of the model the same way as in the case of simple linear regression:

```
In [4]: r_sq = model.score(x, y)
print('coefficient of determination:', r_sq)
print('intercept:', model.intercept_)
print('coefficients:', model.coef_)
```

```
coefficient of determination: 0.8615939258756776
intercept: 5.52257927519819
coefficients: [0.44786965 0.25502548]
```

You obtain the value of R^2 using `score()` and the values of the estimators of regression coefficients with `intercept_` and `coef_`. Again, `intercept_` holds the bias b_0 , while now `coef_` is an array containing b_1 and b_2 respectively.

In this example, the intercept is approximately 5.52, and this is the value of the predicted response when $x_1 = x_2 = 0$. The increase of x_1 by 1 yields the rise of the predicted response by 0.45. Similarly, when x_2 grows by 1, the response rises by 0.26.

Step 5: Predict response

Predictions also work the same way as in the case of simple linear regression: The predicted response is obtained with `.predict()`

```
In [5]: y_pred = model.predict(x)
print('predicted response:', y_pred, sep='\n')
```

```
predicted response:
[ 5.77760476  8.012953  12.73867497 17.9744479  23.97529728 29.4660957
 38.78227633 41.27265006]
```


Exercises to ML5

Homework H5.1 - "sLR manual calculations of R² & Jupyter Notebook (Python)"

Consider 3 points P1=(1|2), P2=(3|3) and P3=(2|2) in the xy-plane. Part b: 1 Person; Rest: 1 Person;

Part a: Calculate the sLR-Measures R² for the two estimated sLR-lines $y=1,5 + 0,5*x$ and $y=1,25 + 0,5*x$. Which estimation (red or green) is better? Check "center of mass". (Hint: $R^2 = 1 - SSE/SST$).

Part b: Calculate the optimal Regression-Line $y = a + b*x$. By using the formulas developed in the lesson for the coefficients a and b. What is R² for this line?

Part c: Build a Jupyter Notebook (Python) to check the manual calculations of Part b. You can use the approach of the lesson by using the Scikit-learn Python library. Optional*: Pls. plot a picture of the "mountain landscape" for R² over the (a,b)-plane.

Part d: Sometimes in the literature or in YouTube videos you see the formula: " $SST = SSR + SSE$ " (SSE, SST see lesson and $SSR := \sum_i (f(x_i) - \text{Mean}(y_i))^2$). Theorem (ML5-2): "This formula is only true, if we have the optimal Regression-Line. **For all other lines it is wrong!** Check this, for the two lines of Part a (red and green) and the opt. Regression-Line calculated in Part b."

Solutions of Homework 5.1 are found in "Exercises2Lecture.pdf"

Check your results:

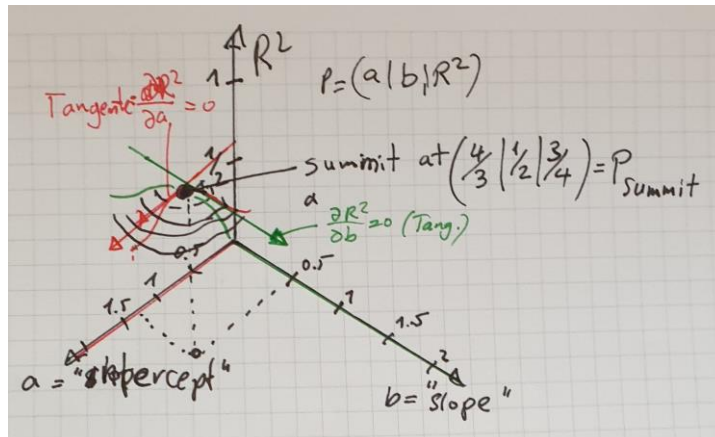
Part a: $R^2 = 5/8 = 0.625$; $R^2 = 23/32 = 0.71875$

⇒ $y = 1,25 + 0,5*x$ is the better regression model.

Part b: $y = 4/3 + 0,5*x$ is the Regression-Line. $R^2 = 3/4$.

Part c: See in GitHub <https://github.com/HVoellinger/Lecture-Notes-to-ML-WS2020>

⇒ the Jupyter Notebook "Homework-ML5_1c-LinReg.ipynb"



Part d:

Exercises to ML5

Homework H5.2*- “Create a Python Pgm. for sLR with. Iowa Houses Data”:

2 Persons: See the video, which shows the coding using Keras library & Python:
<https://www.youtube.com/watch?v=Mcs2x5-7bc0>. Repeat the coding with the dataset “Iowa Homes” to predict the “House Price” based on “Square Feet”. See the result:



Exercises to Lesson ML5

Homework H5.3 – “Calculate Adj.R² for mLR”

See also the YouTube Video: “Regression II: Degrees of Freedom EXPLAINED | Adjusted R-Squared”; <https://www.youtube.com/watch?v=4otEcA3gjLk>

Task:

- Part A: Calculate Adj.R² for given R² for a “Housing Price” example (see table below). Did you see a “trend”?
- Part B: What would be the best model if n=25 and if n=10 (use Adj.R²)?

Multiple Regression
 Example: **Housing Prices**
 Key Predictor Variable: **Sq. Footage**
 Dependent Variable: **Home Sale Price**
 Other factors: **Median neighborhood income**
 • Age of the home
 • size of the lot
 • quality of local schools

number of observations, n	number of variables, k	R ²
25	4	0.71
25	5	0.76
25	6	0.78
25	7	0.79
10	4	0.71
10	5	0.76
10	6	0.78
10	7	0.79

Part A: Check your results:

number of observations, n	number of variables, k	R ²	Adj-R ²
25	4	0.71	0.652
25	5	0.76	0.6968
25	6	0.78	0.7067
25	7	0.79	0.7035
10	4	0.71	0.4780
10	5	0.76	0.4600
10	6	0.78	0.3400
10	7	0.79	0.0550

Complete solutions are found in “Exercises2Lecture.pdf”

Exercises to ML5

Homework H5.4 - "mLR (k=2) manual calculations of Adj.R² & Jupyter Notebook (Python)"

Consider the 4 points $P_1=(1|2|3)$, $P_2=(3|3|4)$, $P_3=(2|2|4)$ and $P_4=(4|3|6)$ in the 3-dimensional space:

Part a: Calculate the sLR-Measures Adj.R² for the two Hyperplanes H_1 : =plane defined by $\{P_1, P_2, P_3\}$ and H_2 : =Plane defined by $\{P_2, P_3, P_4\}$. Which plane (red or green) is a better mLR estimation?

Part b: What is the optimal Regression-Plane $z = a + b \cdot x + c \cdot y$. By using the formulas developed with "Least Square Fit for mLR" method for the coefficients a, b and c. What is Adj.R² for this plane? (Hint: $a=17/4$, $b=3/2$, $c=-3/2$; $R^2=0,9474$ and $\text{Adj.R}^2 \sim 0.8421$)

Part c: Build a Jupyter Notebook(Python) to check the manual calculations of part b. You can use the approach of the lesson by using the Scikit-learn Python library.

Part a: 1 Person, Part b +c: 1 Person

Solutions of Homework 5.4 are found in "Exercises2Lecture.pdf"

Check your results:

Part a: $z = 4 + x - y$, $z = 4 + 2x - 2y$; $R^2 = R^2 = 15/19 \Rightarrow \text{AdjR}^2$ is also equal for both planes=7/19 (no decision about better plan is possible).

Part b: $z = 17/4 + (3/2) \cdot x - (3/2) \cdot y$ is the optimal Regression-Plane. $\text{AdjR}^2 \sim 0.8421$

Part c: See in GitHub <https://github.com/HVoellinger/Lecture-Notes-to-ML-WS2020>
 \Rightarrow the Jupyter Notebook "Homework-ML5_4c-LinReg.ipynb"

Exercises to ML5

Homework H5.5* - Decide ($SST=SSE+SSR$) => optimal sLR-line ?

Examine this direction of the ($SST=SSE+SSR$) condition. We could assume that the condition: " $SST = SSR + SSE$ " (*) also implies that $y(x)$ is an optimal regression line. In many examples this is true! (see homework 5H.1_a).

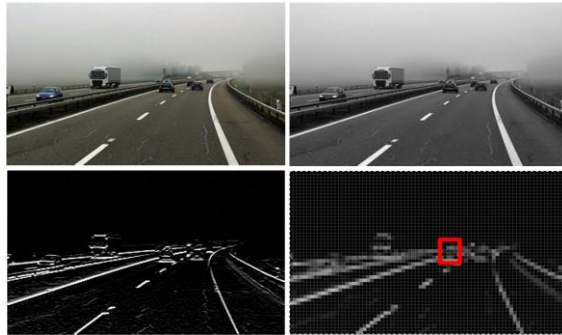
Task: Decide the two possibilities a) and b): (2 Persons, one for each step)

- a. Statement is true, so you have to prove this. I.e. Show that when the "mixed term" of the equation is zero ($\sum[(f_i - y_i) * (f_i - M(y))] = 0$ for all i) implies an optimal sLR-line.
- b. To prove that it's wrong, it's enough to construct a counterexample: define a *Training Set* $TS = \{observation\text{-}points\}$; a sLR-line which has condition (*), but is not an optimal sLR-line.

Solutions of Homework H5.5 are found in "Exercises2Lecture.pdf"

1. ML1: Introduction to Machine Learning (ML)
2. ML2: Concept Learning: VSpaces & Cand. Elim. Algo.
3. ML3: Supervised and Unsupervised Learning
4. ML4: Decision Tree Learning
5. ML5: simple Linear Regression (sLR) & multiple Linear Regression (mLR)
6. **ML6: Neural Networks: Convolutional NN**
7. ML7: Neural Network: BackPropagation Algorithm
8. ML8: Support Vector Machines (SVM)

ML6 – Convolutional Neural Networks (CNN)



<https://www.youtube.com/watch?v=3JQ3hYko51Y>

Status: 1. Dezember 2020

Page
158

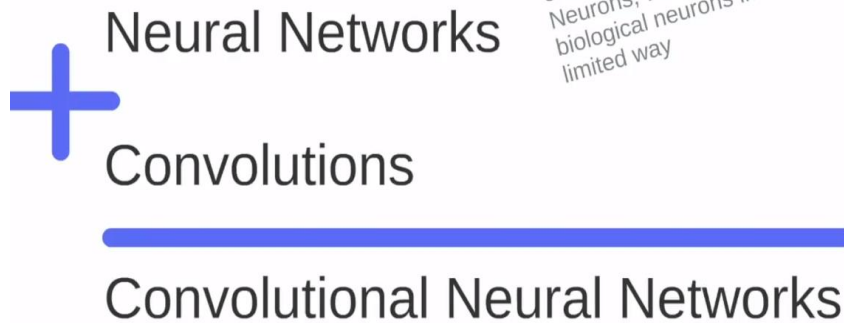
See the also the following documents/scripts:

- Scikit-learn 0.23.1 “ Ch. 1.17. Neural network models (supervised)” https://scikit-learn.org/stable/modules/neural_networks_supervised.html?highlight=back%20propagation

See also YouTube videos:

- Neural Network 3D Simulation: <https://www.youtube.com/watch?v=3JQ3hYko51Y>
- “Convolutional Neural Networks (CNNs) explained” <https://www.youtube.com/watch?v=YRhxdVksls>
- “Convolutional Neural Network Tutorial (CNN) | How CNN Works | Deep Learning Tutorial | Simplilearn” <https://www.youtube.com/watch?v=Jy9-aGMBTE>

CNN simulate Biological Neurons



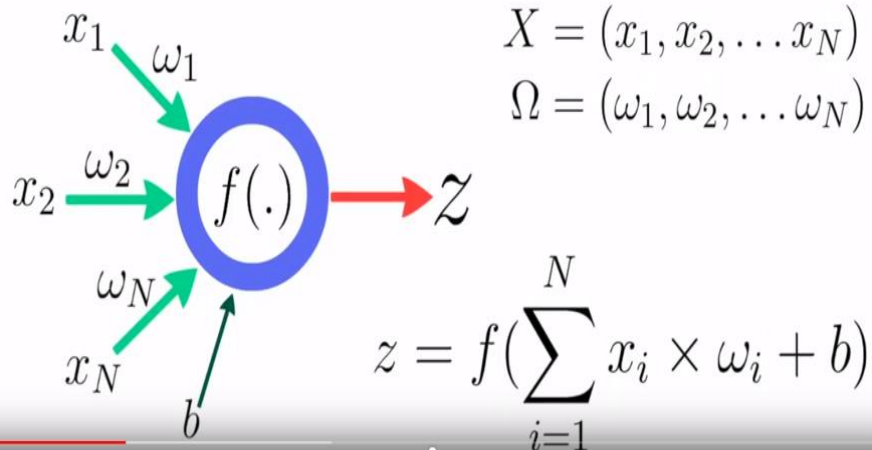
For more details see:

<https://www.youtube.com/watch?v=ol2rvjbzVml>

CNN is the result of two elements: well-known algorithm of artificial neural networks plus a set of operations that we will call convolution. By combining these two kinds of ideas here we get the convolutional neural networks or simply CNN.

Recalling the concept neuro networks are composed of artificial neurons which simulate biological neurons in a limited way.

CNN Concepts – The Artificial Neuron



Let's just take a look at the artificial neuron.

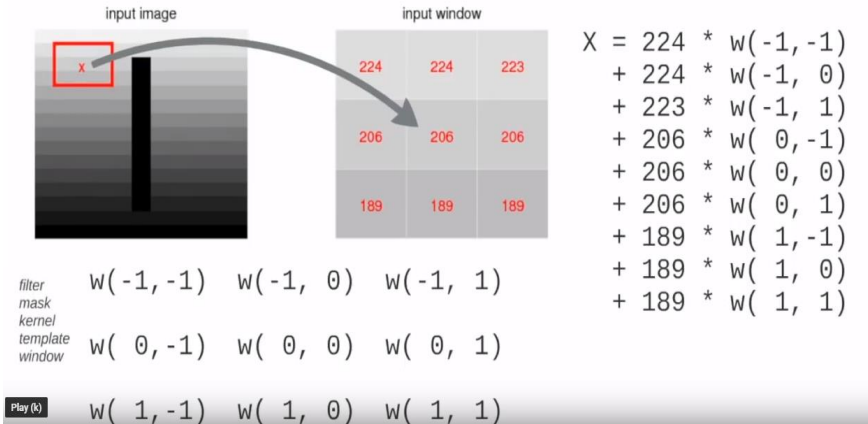
What we have here is a set of elements that are represented by the set of inputs of x_1, x_2 up to x_N which are connected to an activation function $f(\cdot)$.

The connection between the input and the activation function is drawn by a set of weights which are represented by Ω_1, Ω_2 up to Ω_N .

Besides this we have a bias, which is the letter b . After applying the output of the activation function is represented by the letter z . Z is a function applied to the input weighted by all these elements plus the bias. In this we connect some input and have a single output.

Remark: bias is a constant distortion.

CNN – Convolution Operations on Image Pixles



The other concept is the convolution operation: we have a single input image and we apply some filter or can be called mask care node template or even window and by applying this we can see that this red square is represented in this input window so all these values pixels of the image in that position and by applying a filter which we can represent here by Omega at these nine pixels of the image in that position pixels of the image in that position.

For these three by three positions we obtain an output value by combining the input window or the input image weighted by all that values inside the filter. The result of the convolution in this position X will be the result of the sum (see right side of the picture)

CNN Architecture Example for Image Recognition

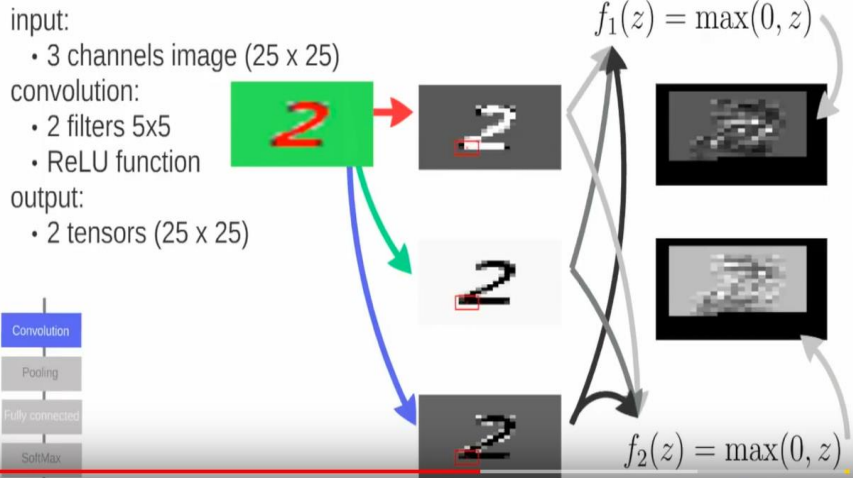


Here is CNN applied to an image classification example. It is an example of a **Supervised Deep Learning** method.

In the example we are using 4 special layers:

The first step or the first layer of the CNN is composed by one or more layers of **convolution** after these we can apply one or more steps of **pooling** after this we design what we call a **fully connected layer** and at the end to obtain a classification we can apply some operation for example **SoftMax**.

Image Recognition - First Convolutional Layer



Here we have the example of image recognition of the number “2”. The slide show the processes (i.e. filters, ReLU) in the first convolutional layer. ReLU layer will apply an elementwise activation function, such as the $\max(0, x)$ thresholding at zero. Output are 2 tensors (25x25).

In general the **convolutional layer** is the core building block of a Convolutional Network that does most of the computational heavy lifting.

In general **ReLU** (Rectified Linear Unit) is an **Activation Function**. The **ReLU** is the most used **activation function** in the world right now. Since, it is used in almost all the convolutional neural networks or deep learning.

Image Recognition - Second Convolutional Layer

input:

- 2 tensors from previous convolution (25 x 25)

convolution:

- 3 filters 3x3
- ReLU function

output:

- 3 tensors (25 x 25)



The second convolutional layer run again filters and ReLu function.
Output are 3 tensors (25 x25)

Image Recognition - Pooling Layer

input:

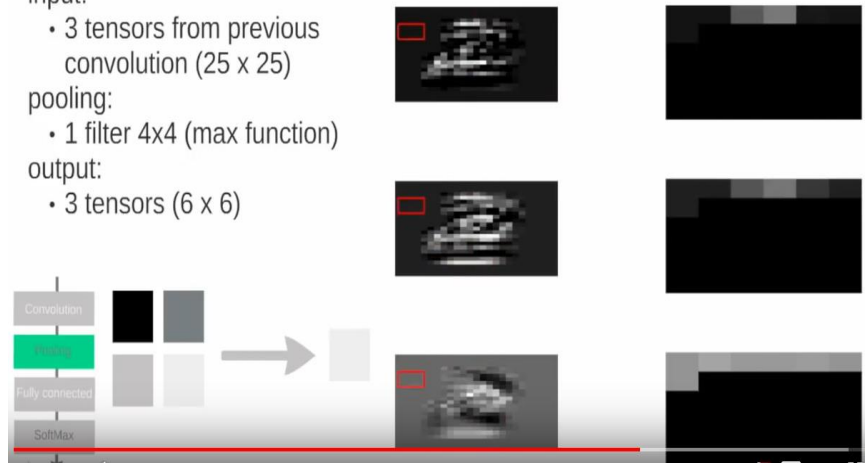
- 3 tensors from previous convolution (25 x 25)

pooling:

- 1 filter 4x4 (max function)

output:

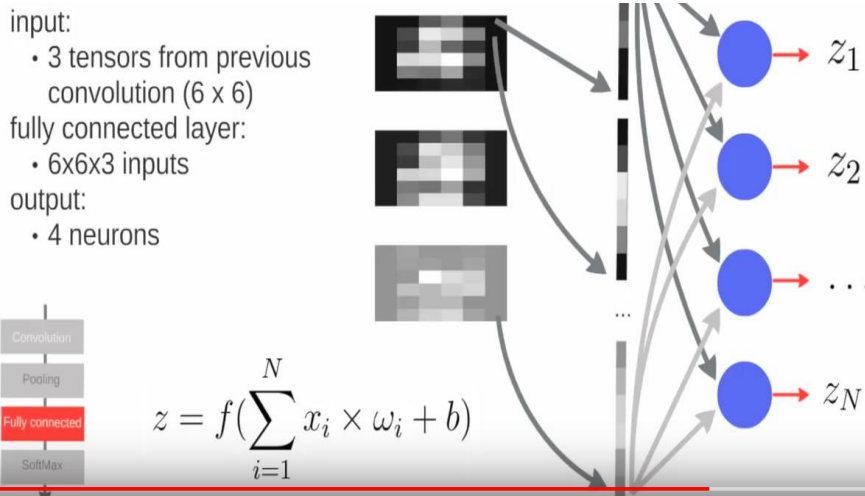
- 3 tensors (6 x 6)



POOLING layer will perform a down-sampling operation along the spatial dimensions (width, height). Here we have a filter 4X4 with max function which is resulting in 3 tensors (6x6).

In general a **pooling layer** is another building block of a **CNN**. Its function is to progressively reduce the spatial size of the representation to reduce the amount of parameters and computation in the network. **Pooling layer** operates on each feature map independently. The most common approach used in **pooling** is max **pooling**.

Image Recognition - Fully Connected Layer



FC (i.e. fully-connected) layer will compute the class scores, resulting in volume of size, where each of the 10 numbers correspond to a class score, such as among the 10 categories of CIFAR-10. As with ordinary Neural Networks and as the name implies, each neuron in this layer will be connected to all the numbers in the previous volume.

Output: 4 neurons.

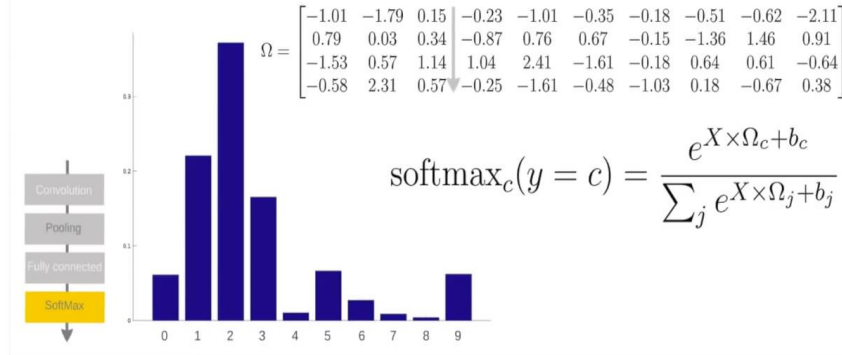
In general the FC is the **fully connected layer** of neurons at the end of CNN. Neurons in a **fully connected layer** have full connections to all activations in the previous **layer**, as seen in regular Neural Networks and **work** in a similar way.

Image Recognition - SoftMax Equation

input:

- 4 neurons from previous layer

$$X = (0.91, 0.51, 0.45, 0.48)$$



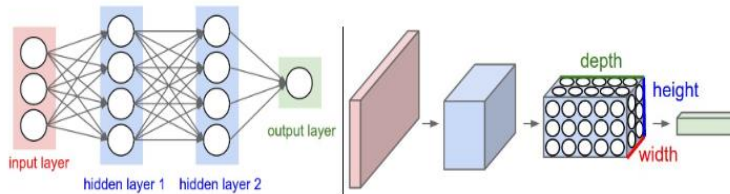
The **SoftMax Layer** (SoftMax Equation) reduces a 2-dimensional matrix (4x10) of results/scores to a 1-dimensional result. The 4 rows of the matrix come from 4 neurons, each delivers 10 numbers/scores. The result are 10 values/scores with values in the interval [0, 1] for the numbers 0,1...9.

In [mathematics](#), the **softmax function**, also known as **softargmax** or **normalized exponential function**, is a function that takes as input a vector of K real numbers, and normalizes it into a [probability distribution](#) consisting of K probabilities. That is, prior to applying softmax, some vector components could be negative, or greater than one; and might not sum to 1; but after applying softmax, each component will be in the [interval](#) [0,1] and the components will add up to 1, so that they can be interpreted as probabilities. Furthermore, the larger input components will correspond to larger probabilities. Softmax is often used in [neural networks](#), to map the non-normalized output of a network to a probability distribution over predicted output classes.

The standard (unit) softmax function $\sigma : \mathbb{R}^K \rightarrow \mathbb{R}^K$ is defined by the formula

$$\sigma(\mathbf{z})_i = \frac{e^{z_i}}{\sum_{j=1}^K e^{z_j}} \quad \text{for } i = 1, \dots, K \text{ and } \mathbf{z} = (z_1, \dots, z_K) \in \mathbb{R}^K$$

CNN – General Architecture Overview



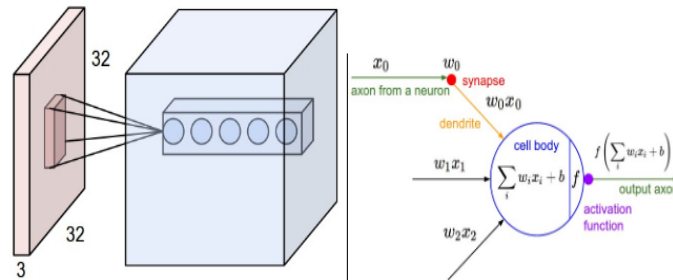
Left: A regular 3-layer Neural Network. Right: A ConvNet arranges its neurons in three dimensions (width, height, depth), as visualized in one of the layers. Every layer of a ConvNet transforms the 3D input volume to a 3D output volume of neuron activations. In this example, the red input layer holds the image, so its width and height would be the dimensions of the image, and the depth would be 3 (Red, Green, Blue channels).

A ConvNet is made up of Layers. Every Layer has a simple API: It transforms an input 3D volume to an output 3D volume with some differentiable function that may or may not have parameters.

Regular Neural Nets: Neural Networks receive an input (a single vector), and transform it through a series of *hidden layers*. Each hidden layer is made up of a set of neurons, where each neuron is fully connected to all neurons in the previous layer, and where neurons in a single layer function completely independently and do not share any connections. The last fully-connected layer is called the “output layer” and in classification settings it represents the class scores. *Regular Neural Nets don’t scale well to full images.*

3D volumes of neurons. Convolutional Neural Networks take advantage of the fact that the input consists of images and they constrain the architecture in a more sensible way. In particular, unlike a regular Neural Network, the layers of a ConvNet have neurons arranged in 3 dimensions: **width, height, depth**. (Note that the word *depth* here refers to the third dimension of an activation volume, not to the depth of a full Neural Network, which can refer to the total number of layers in a network.) For example, the input images in CIFAR-10 are an input volume of activations, and the volume has dimensions 32x32x3 (width, height, depth respectively). As we will soon see, the neurons in a layer will only be connected to a small region of the layer before it, instead of all of the neurons in a fully-connected manner. Moreover, the final output layer would for CIFAR-10 have dimensions 1x1x10, because by the end of the ConvNet architecture we will reduce the full image into a single vector of class scores, arranged along the depth dimension.

Layers used to build ConvNets - Convolutional Layer

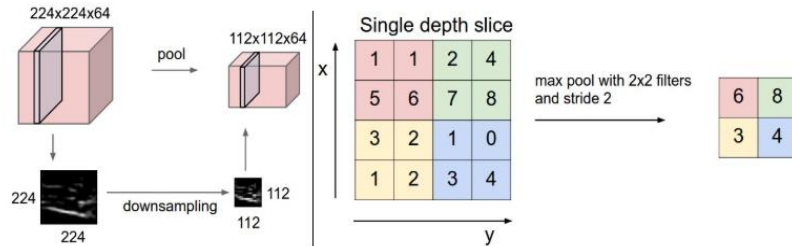


Left: An example input volume in red (e.g. a 32x32x3 CIFAR-10 image), and an example volume of neurons in the first Convolutional layer. Each neuron in the convolutional layer is connected only to a local region in the input volume spatially, but to the full depth (i.e. all color channels). Note, there are multiple neurons (5 in this example) along the depth, all looking at the same region in the input - see discussion of depth columns in text below. **Right:** The neurons from the Neural Network chapter remain unchanged: They still compute a dot product of their weights with the input followed by a non-linearity, but their connectivity is now restricted to be local spatially.

As we described above, a simple ConvNet is a sequence of layers, and every layer of a ConvNet transforms one volume of activations to another through a differentiable function. We use three main types of layers to build ConvNet architectures: **Convolutional Layer**, **Pooling Layer**, and **Fully-Connected Layer** (exactly as seen in regular Neural Networks). We will stack these layers to form a full ConvNet **architecture**.

Layers used to build ConvNets - Pooling Layer

General pooling. In addition to max pooling, the pooling units can also perform other functions, such as *average pooling* or even *L2-norm pooling*. Average pooling was often used historically but has recently fallen out of favor compared to the max pooling operation, which has been shown to work better in practice.



Pooling layer downsamples the volume spatially, independently in each depth slice of the input volume. **Left:** In this example, the input volume of size $[224 \times 224 \times 64]$ is pooled with filter size 2, stride 2 into output volume of size $[112 \times 112 \times 64]$. Notice that the volume depth is preserved. **Right:** The most common downsampling operation is max, giving rise to **max pooling**, here shown with a stride of 2. That is, each max is taken over 4 numbers (little 2×2 square).

Layers used to build ConvNets - Fully-Connected Layer

Fully-connected layer

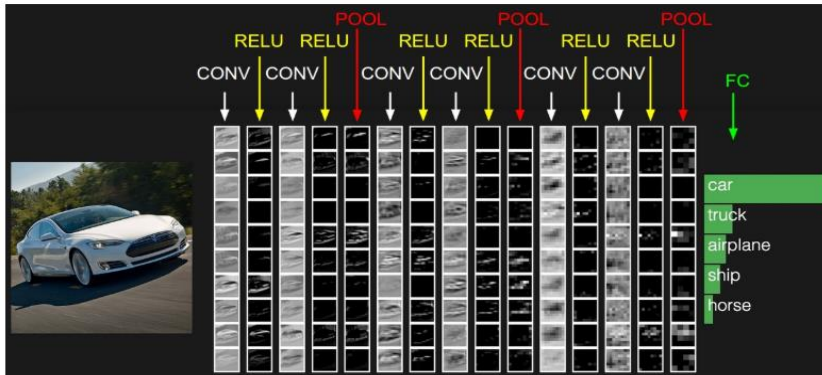
Neurons in a fully connected layer have full connections to all activations in the previous layer, as seen in regular Neural Networks. Their activations can hence be computed with a matrix multiplication followed by a bias offset. See the *Neural Network* section of the notes for more information.

Converting FC layers to CONV layers

It is worth noting that the only difference between FC and CONV layers is that the neurons in the CONV layer are connected only to a local region in the input, and that many of the neurons in a CONV volume share parameters. However, the neurons in both layers still compute dot products, so their functional form is identical. Therefore, it turns out that it's possible to convert between FC and CONV layers:

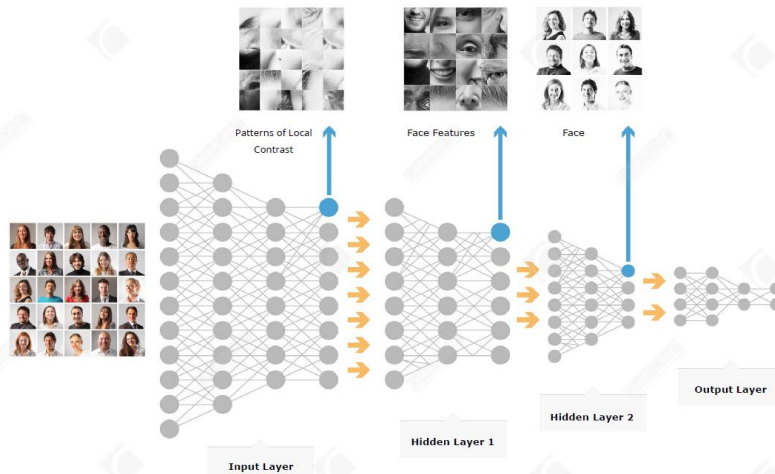
- For any CONV layer there is an FC layer that implements the same forward function. The weight matrix would be a large matrix that is mostly zero except for at certain blocks (due to local connectivity) where the weights in many of the blocks are equal (due to parameter sharing).
- Conversely, any FC layer can be converted to a CONV layer. For example, an FC layer with $K = 4096$ that is looking at some input volume of size $7 \times 7 \times 512$ can be equivalently expressed as a CONV layer with $F = 7, P = 0, S = 1, K = 4096$. In other words, we are setting the filter size to be exactly the size of the input volume, and hence the output will simply be $1 \times 1 \times 4096$ since only a single depth column "fits" across the input volume, giving identical result as the initial FC layer.

Layers to build ConvNets–Example “Image Recognition”



The activations of an example ConvNet architecture. The initial volume stores the raw image pixels (left) and the last volume stores the class scores (right). Each volume of activations along the processing path is shown as a column. Since it's difficult to visualize 3D volumes, we lay out each volume's slices in rows. The last layer volume holds the scores for each class, but here we only visualize the sorted top 5 scores, and print the labels of each one. The full [web-based demo](#) is shown in the header of our website. The architecture shown here is a tiny VGG Net, which we will discuss later.

Deep Neural Networks (DNN)



Status: 1. Dezember 2020

Page
173

DNNs have already made breakthroughs for a number of practical applications, the most widely used being **speech recognition**. Without DNN Siri would not have been possible. DNN is also used for **playing games**: The new with DNN is that systems like Goggle AlphaGo has learned the game strategy independently. The victory of IBM Deep Blue against Kasparov in chess was a victory of ingenious programmers in combination with superior computing power. AlphaGo, on the other hand, has achieved its progress since October - when it beat the European champions - by **playing and learning against itself**. This is the new capability of Deep Neural Networks (DNN). In other games, DNN have already demonstrated how autonomous their learning has become: for example, [Space Invaders](#), where the DNN became a master player just by "watching" the screen pixels and playing around with joystick moves.

See also: <https://www.latimes.com/science/sciencenow/la-sci-sn-computer-learning-space-invaders-20150224-story.html>

However, for many areas, such as autonomous driving or lending, the use of such networks is extremely critical and risky due to their "**black box**" nature, as it is difficult to interpret how or why the models come to particular conclusions. It is an open issue to understand and explain the decision making of deep neural networks.

UC1 - Daimler AG: “Deep-Learning-und-Autonomes-Fahren”

Referent: Eike Rehder (Daimler AG.)

Inhalt: Automatisches Fahren verspricht, die Mobilität neu zu gestalten. Damit ein automatisches Fahrzeug sicher im Straßenverkehr navigieren kann, muss es mit einer künstlichen Intelligenz ausgestattet werden. Diese ermöglicht dem Fahrzeug, seine Umgebung wahrzunehmen, das Verkehrsgeschehen zu analysieren und Entscheidungen zu treffen.

Ein wesentlicher Baustein für diese neue Intelligenz sind künstliche neuronale Netze (KNNs). Obwohl diese bereits seit Jahrzehnten Gegenstand der Forschung sind, haben sie erst in den letzten Jahren eine Qualität erreicht, die ihre Anwendung im Straßenverkehr in greifbare Nähe rücken lässt. Dies geht zurück auf die Entwicklung des so genannten Deep Learning, bei dem KNNs mit mehreren Millionen Parametern erfolgreich trainiert werden können. Diese tiefen Netze können überall dort Anwendung finden, wo eine explizite Modellierung durch den Menschen kaum möglich ist. Insbesondere in der Verarbeitung von Sensordaten sowie dem Verstehen komplexer Verkehrsszenarien werden sie eingesetzt.

Dieser Vortrag soll eine Einführung in das maschinelle Lernen, künstliche Neuronale Netze und Deep Learning geben. Als Beispiele hierfür dienen Anwendungen aus der Wahrnehmung, Situationsinterpretation und Entscheidungsfindung des automatischen Fahrens.

<https://rg-stuttgart.gi.de/veranstaltung/deep-learning-und-autonomes-fahren/>

Automatic driving promises to redesign mobility. For an automatic vehicle to navigate safely on the road, it has to be equipped with artificial intelligence. This allows the vehicle to perceive its surroundings, analyze the traffic and make decisions.

An essential building block for this new intelligence are artificial neural networks (KNNs). Although these have been the subject of research for decades, only in recent years have they achieved a quality that makes their use in road traffic within reach. This goes back to the development of so-called deep learning, in which KNNs with several million parameters can be successfully trained.

These deep networks can be used everywhere where explicit modeling by humans is hardly possible. In particular, they are used in the processing of sensor data and the understanding of complex traffic scenarios.

UC2 – Fraunhofer IEE + enercast GmbH (Kassel, Germany): “Power forecasts for renewable energy with CNN”

By using Big Data & Artificial Intelligence (CNN) Fraunhofer IEE and enercast GmbH have been operating (since 2011) forecast services via distributed and redundant data centers in Germany. The currently 600 TB of Managed Data are processed via a robust, industry-ready Big Data infrastructure – growth per day: 150 GB.

Hundreds of artificial neural networks can be trained on our Artificial Intelligence platform. For each site, we transform data points into high-quality and relevant information via this infrastructure (24/7 calculation, transmission and monitoring). See also [White Paper](#)



Status: 1. Dezember 2020

Page
175

Fraunhofer IEE has more than 15 years of experience in forecasting from volatile energy producers. Enercast GmbH (<https://www.enercast.de/>) has been developing since 2011 together with Fraunhofer in a cooperation project reliable and accurate performance forecasts and extrapolations for wind + photovoltaic systems. The special at the calculations is that these rely on artificial intelligence - through the use of neural networks. All data is processed by an algorithm. For more details see also the whitepaper: https://www.enercast.de/wp-content/uploads/2018/04/whitepaper-prognosen-wind-solar-kuenstliche-intelligenz-neuronale-netze_110418_EN.pdf

Detail: The entire infrastructure was gradually built up - always with the objective of providing a max. to achieve possible forecast quality. Fact is, per system (so for a single wind turbine) are always used multiple networks. This leads for a complete wind farm to hundreds of networks at work. The first training of network is mostly on Cassandra clusters. Only when a certain quality is achieved, the switch takes place in the operational mode - then without Cassandra. The networks then change continuously and adapt to the respective local conditions. That is, if there are changes in the local climate situation, then the grids correct themselves automatically to provide optimal performance prediction. Exactly this adaptability is the big advantage over the classic, static models. The static models describe the situation through a set of fixed parameters in an algorithm. Economically, the static models deliver faster results, whereas the AI-based systems achieve higher forecasting quality in the medium term.

UC3 -Semantic Search: “Predictive Market with Fact-Finder” <https://youtu.be/vSWLafBdHus>

Machine Learning: FACT-Finder sagt voraus, was Kunden brauchen

Kunden bestellen zwar immer wieder die gleichen Verbrauchsartikel, trotzdem ist kein Einkauf wie der andere: Manches wird ständig gekauft (Vitamin-tabletten), manches nur sporadisch (Heuschnupfenspray) und manches einmalig (Nagelschere). FACT-Finder erkennt die Kauf-rhythmen innerhalb eines Shops und kann daher bereits ab dem zweiten Einkauf Vorschläge ausspielen, die mit

hoher Wahrscheinlichkeit gekauft werden – Mehrumsatz vorprogrammiert. Dank Machine-Learning-Algorithmen passt sich der Predictive Basket zudem an das individuelle Kundenverhalten an. Bevor einem Kunden bestimmte Verbrauchsartikel ausgehen – und bevor er sie womöglich woanders kauft –, erinnert FACT-Finder an die Wiederbestellung der Produkte.



Status: 1. Dezember 2020

Page
176

Maximize re-orders. With the Predictive Basket. Online shopping without rummaging and thinking: The Predictive Basket by FACT-Finder shows yours customers are likely to buy those products that they are likely to buy in their current session become.

Whether click, search query or purchase - with every interaction in the shop your customers leave traces in the form of data. With our tracking interface, this data is captured by your FACT-Finder and used for shop optimization. But what exactly does FACT-Finder use the tracking data for? And what does shop tracking bring to your customers and you as a user? In this post you will find the answers. Your box office hits automatically move into the focus of the customers, based on the shop tracking data. FACT-Finder learns which products are most popular with your customers - that is, which ones they most click on, add to the shopping cart and buy. This knowledge can be incorporated into the sorting of your search results - by activating the Automatic Search Optimization. Your bestsellers will automatically move up to the top positions in the result ranking over time. This increases the purchase probability. Because the higher the relevant products appear, the faster they will catch your customers. See also Fact-Finder Webinar: Dr. Holger Schmidt – „Wie Plattformen den eCommerce disrupten“:

<https://www.youtube.com/watch?v=9T9sOxRB9gg&feature=youtu.be>

UC4 – Deep Neural Network – “Google AlphaGo”

<https://storage.googleapis.com/deepmind-media/alphago/AlphaGoNaturePaper.pdf>

The story of AlphaGo so far

AlphaGo is the first computer program to defeat a professional human Go player, the first program to defeat a Go world champion, and arguably the strongest Go player in history.

AlphaGo's first formal match was against the reigning 3-times European Champion, Mr. Fan Hui, in October 2015. Its 5-0 win was the first ever against a Go professional, and the results were published in full technical detail in the international journal, [Nature](#). AlphaGo then went on to compete against legendary player Mr Lee Sedol, winner of 18 world titles and widely considered to be the greatest player of the past decade.

AlphaGo's 4-1 victory in Seoul, South Korea, in March 2016 was watched by over 200 million people worldwide. It was a landmark achievement that experts agreed was a decade ahead of its time, and earned AlphaGo a 9 dan professional ranking (the highest certification) - the first time a computer Go player had ever received the accolade.

During the games, AlphaGo played a handful of [highly inventive winning moves](#), several of which - including move 37 in game two - were so surprising they overturned hundreds of years of received wisdom, and have since been examined extensively by players of all levels. In the course of winning, AlphaGo somehow taught the world completely new knowledge about perhaps the most studied and contemplated game in history.

Since then, AlphaGo has continued to surprise and amaze. In January 2017, an improved AlphaGo version was revealed as the online player "Master" which achieved [60 straight wins in online fast time-control games](#) against top international Go players.



Go behind the scenes in Seoul with the AlphaGo movie

[Watch now on Netflix](#)



Demis Hassabis on 'The Future of Go' Summit in China

Status: 1. Dezember 2020

Page
177

The victory of Google-developed DeepMind AlphaGo software against South-Korean Go world- champion Lee Sedol does not simply ring in the next round of industrial revolution. According to IT expert Carsten Kraus, the superiority of Deep Neural Networks (DNN) with respect to human intelligence has probably even put an end to all the upheavals ...

What's so special about AlphaGo? The novelty is that AlphaGo has learned the game strategy independently. The victory of IBM Deep Blue against Kasparov in chess was a victory of ingenious programmers in combination with superior computing power. AlphaGo, on the other hand, has achieved its progress since October - when it beat the European champions - by playing and learning against itself. This is the new capability of Deep Neural Networks (DNN). In other games, DNN have already demonstrated how autonomous their learning has become: for example, [Space Invaders](#) Space Invaders, where the DNN became a master player just by "watching" the screen pixels and playing around with joystick moves. See also YouTube videos:

- “AlphaGo - The Movie | Full Documentary” (2016):
<https://www.youtube.com/watch?v=WXuK6gekU1Y>
- New DeepMind AI Beats AlphaGo 100-0 | Two Minute Papers #201 (2017):
<https://www.youtube.com/watch?v=9xISy9F5WtE>
- Google's Deep Mind Explained! - Self Learning A.I.
<https://www.youtube.com/watch?v=TnUYcTuZjPM>

Example from Microsoft Azure- Machine Learning

EXPERIMENT

Compare Multi-class Classifiers: Letter recognition

By AzureML Team for Microsoft · September 2, 2014

8 likes



Summary

This sample demonstrates how to compare multiple multi-class classifiers using the letter recognition dataset.

Description

Compare Multi-class Classifiers: Letter Recognition

This sample demonstrates how to create multiclass classifiers and evaluate and compare the performance of multiple models.

Data

For this experiment, we use the letter image recognition data from the UCI repository. The first column is the label, which identifies each row as one of 26 letters, A-Z. The remaining 16 columns are feature columns. The dataset contains 20000 instances. Description and other details about the data can be found at <http://archive.ics.uci.edu/ml/machine-learning-databases/letter-recognition/letter-recognition.names>.

Label	Cut1	Cut2	Cut3	Cut4	Cut5	Cut6	Cut7	Cut8	Cut9	Cut10	Cut11	Cut12	Cut13	Cut14
A	1	2	9	2	5	1	0	10	0	0	0	10	0	0
B	1	9	10	2	7	2	10	0	0	0	0	10	0	0
C	1	1	11	0	0	0	10	0	0	0	10	0	0	0
D	7	10	0	0	0	0	0	0	0	0	10	0	0	0
E	2	1	0	1	1	0	0	0	0	0	0	0	0	1
F	1	11	0	0	0	0	0	0	0	0	0	0	0	0
G	4	2	0	1	0	0	0	0	0	0	0	0	0	0
H	1	1	0	0	0	0	0	0	0	0	0	0	0	0
I	1	1	0	0	0	0	0	0	0	0	0	0	0	0
J	2	2	0	0	0	0	0	0	0	0	0	0	0	0
K	1	1	10	0	0	0	0	0	0	0	0	0	0	0
L	11	10	10	0	0	0	0	0	0	0	0	0	0	0
M	1	1	0	0	0	0	0	0	0	0	0	0	0	0
N	0	0	0	0	0	0	0	0	0	0	0	0	0	0
O	0	0	0	0	0	0	0	0	0	0	0	0	0	0
P	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Q	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0
S	0	0	0	0	0	0	0	0	0	0	0	0	0	0
T	0	0	0	0	0	0	0	0	0	0	0	0	0	0
U	0	0	0	0	0	0	0	0	0	0	0	0	0	0
V	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0
X	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Y	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Z	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Letter	Probability
a	0.7
q	0.2
o	0.1

Open in Studio

Add to Collection

22735 views

5628 downloads

RELATED ITEMS

- Compare Multi-class Classifiers: Letter...
- Compare Multi-class Classifiers: Letter...
- Compare Multi-class Classifiers: Letter...

ALGORITHMS

One-vs-All Multiclass, Two-Class Support Vector Machine, Multiclass Decision Forest, Multiclass Decision Jungle, Multiclass Logistic Regression, Multiclass Neural Network

TAGS

- multi-class
- multi-class classifier
- neural network
- decision forest

<https://gallery.azure.ai/Experiment/Compare-Multi-class-Classifiers-Letter-recognition-2>

Exercises to Lesson ML6

Homework H6.1 – “Power Forecasts with CNN in UC2”

Groupwork (2 Persons): Evaluate and explain in more details the CNN in “UC2-Fraunhofer + enercast: Power forecasts for renewable energy with CNN”, see: https://www.enercast.de/wp-content/uploads/2018/04/whitepaper-prognosen-wind-solar-kuenstliche-intelligenz-neuronale-netze_110418_EN.pdf

Homework H6.2 – “Evaluate AI Technology of UC3”

Groupwork (2 Persons): Evaluate and find the underlying AI technology which is used in “UC3 – Semantic Search: Predictive Basket with Fact-Finder”. See: <https://youtu.be/vSWLafBdHus>

Solutions are found in Ref. **[HVö-4]**: “Exercises2Lecture.pdf”

Exercises to Lesson ML6

Homework H6.3* (advanced) – “Create Summary to GO Article”

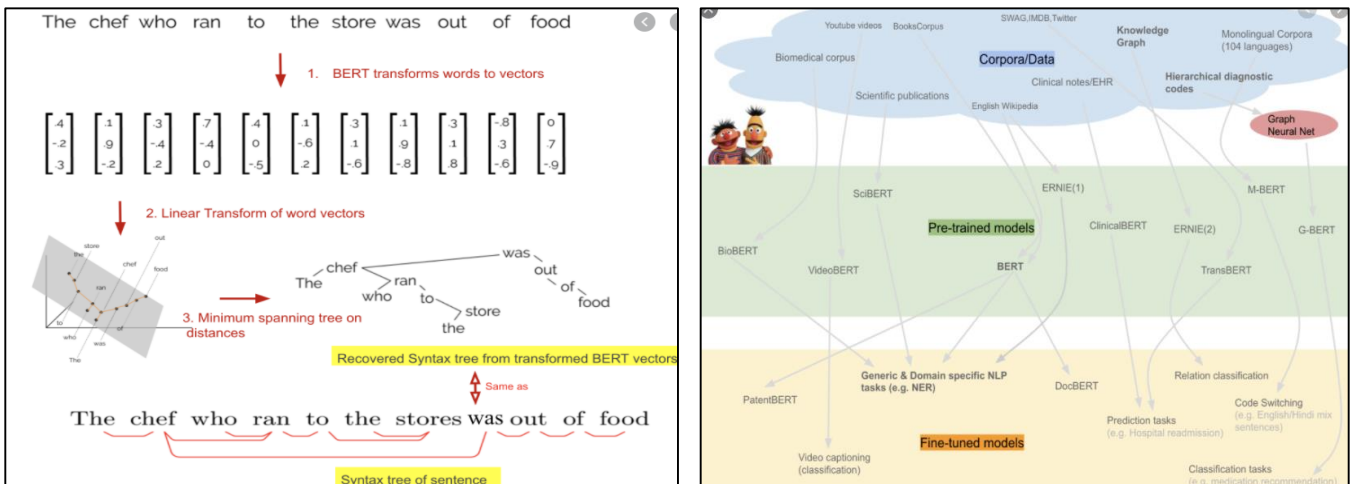
Groupwork (2 Persons): Read and summaries of the main results of the article “Mastering the game of Go with deep neural networks and tree search”. See also:
<https://storage.googleapis.com/deepmind-media/alphago/AlphaGoNaturePaper.pdf>

Homework H6.4* (advanced) – “Create Summary to BERT Article”

Groupwork (2 Persons): Read and summaries of the main results of the article about BERT. See Ref. **[BERT]**: Jacob Devlin and Other: “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding”; Google(USA); 2019; See for introduction to BERT the following video:
<https://www.youtube.com/watch?v=xI0HHN5XKDo>

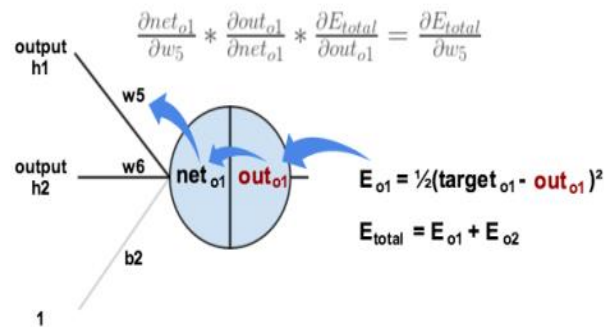
Solutions are found in Ref. **[HVö-4]**: “Exercises2Lecture.pdf”. Details to **Homework-H6.4: Bidirectional Encoder Representations from Transformers (BERT)** is a technique for [natural language processing](#) (NLP) pre-training developed by [Google](#). BERT was created and published in 2018 by Jacob Devlin and his colleagues from Google. [\[1\]\[2\]](#) Google is leveraging BERT to better understand user searches. [\[3\]](#)

The original English-language BERT model comes with two pre-trained general types: [\[1\]](#) (1) the BERT_{BASE} model, a 12-layer, 768-hidden, 12-heads, 110M parameter neural network architecture, and (2) the BERT_{LARGE} model, a 24-layer, 1024-hidden, 16-heads, 340M parameter neural network architecture; both of which were trained on the [BooksCorpus](#)^[4] with 800M words, and a version of the [English Wikipedia](#) with 2,500M words.



1.	ML1: Introduction to Machine Learning (ML)
2.	ML2: Concept Learning: VSpaces & Cand. Elim. Algo.
3.	ML3: Supervised and Unsupervised Learning
4.	ML4: Decision Tree Learning
5.	ML5: simple Linear Regression (sLR) & multiple Linear Regression (mLR)
6.	ML6: Neural Networks: Convolutional NN
7.	ML7: Neural Network: BackPropagation Algorithm
8.	ML8: Support Vector Machines (SVM)

ML7 – BackPropagation for Neural Networks



See the following documents/scripts:

- “A Step by Step Backpropagation Example” by Matt Mazur (2018); <https://mattmazur.com/2015/03/17/a-step-by-step-backpropagation-example/>
- Scikit-learn 0.23.1 “Ch. 1.17. Neural network models (supervised)” https://scikit-learn.org/stable/modules/neural_networks_supervised.html?highlight=back%20propagation

See following YouTube Videos:

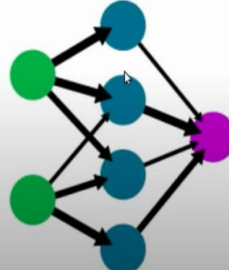
- “Back Propagation in Neural Network with an Example | Machine Learning (2019); <https://www.youtube.com/watch?v=GJXKOrqZauk>
- “Back propagation neural network with example in Hindi and How it works?” (31.12.2018); https://www.youtube.com/watch?v=0_2nA_WoSmE

What is BackPropagation

Backpropagation is a supervised learning algorithm, for training Neural Networks.

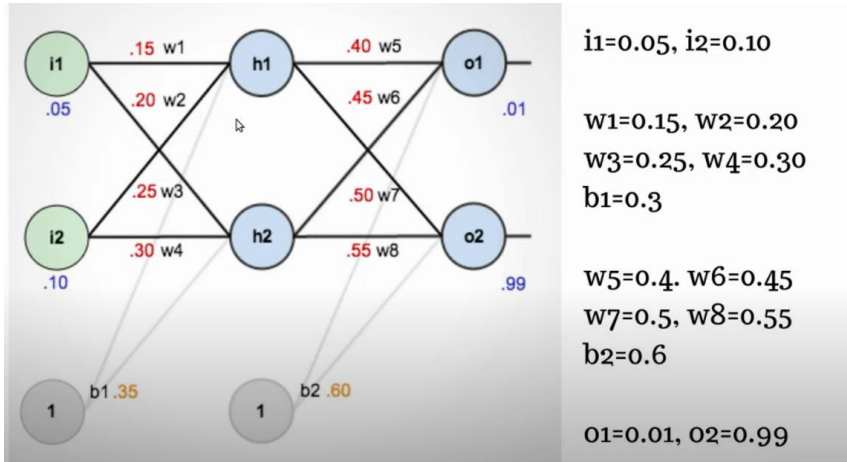
A simple neural network

input layer hidden layer output layer



<https://mattmazur.com/2015/03/17/a-step-by-step-backpropagation-example/>

Simple Example of a Neural Network



$i_1=0.05, i_2=0.10$

$w_1=0.15, w_2=0.20$

$w_3=0.25, w_4=0.30$

$b_1=0.3$

$w_5=0.4, w_6=0.45$

$w_7=0.5, w_8=0.55$

$b_2=0.6$

$O_1=0.01, O_2=0.99$

Exercises to Lesson ML7

Homework 7.1 – “Exercise of an Example with Python”

*****placeholder*****

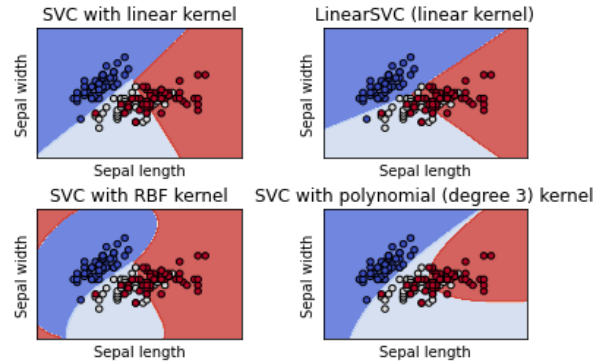
Homework 7.2 – “Exercise of an Example with Python”

*****placeholder*****

Solutions are found in Ref. **[HVö-4]**: “Exercises2Lecture.pdf”

1. ML1: Introduction to Machine Learning (ML)
2. ML2: Concept Learning: VSspaces & Cand. Elim. Algo.
3. ML3: Supervised and Unsupervised Learning
4. ML4: Decision Tree Learning
5. ML5: simple Linear Regression (sLR) & multiple Linear Regression (mLR)
6. ML6: Neural Networks: Convolutional NN
7. ML7: Neural Network: BackPropagation Algorithm
8. ML8: Support Vector Machines (SVM)

ML8 – Support Vector Machines (SVM)



See also:

<https://courses.cs.washington.edu/courses/cse573/05au/support-vector-machines.ppt>

More details: https://en.wikipedia.org/wiki/Support_vector_machine

Motivation and Definition of SVM

*****placeholder*****

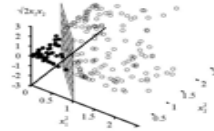
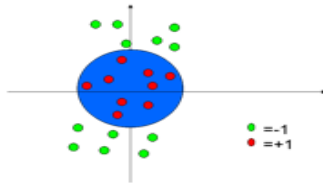
More details: https://en.wikipedia.org/wiki/Support_vector_machine

Linear Support Vector Machine (LSVM)

*****placeholder*****

LSVM – Kernel Trick

Kernel Trick



Data points are linearly separable
in the space $(x_1^2, x_2^2, \sqrt{2}x_1x_2)$

We want to maximize $\sum_i \alpha_i - \frac{1}{2} \sum_{i,j} y_i y_j \alpha_i \alpha_j \langle F(\mathbf{x}_i) \cdot F(\mathbf{x}_j) \rangle$

Define $K(\mathbf{x}_i, \mathbf{x}_j) = \langle F(\mathbf{x}_i) \cdot F(\mathbf{x}_j) \rangle$

Cool thing : K is often easy to compute directly! Here,

$$K(\mathbf{x}_i, \mathbf{x}_j) = \langle \mathbf{x}_i \cdot \mathbf{x}_j \rangle^2$$

Examples of SVM – Python Algorithm

***** placeholder *****

https://scikit-learn.org/stable/auto_examples/svm/plot_iris_svc.html#sphx-glr-auto-examples-svm-plot-iris-svc-py

Exercises to Lesson ML8

Homework H8.1 – “Exercise of an Example with Python”

***** placeholder*****

Homework H8.2 – “Exercise of an Example with Python”

***** placeholder*****

Solutions are found in Ref. **[HVö-4]**: “Exercises2Lecture.pdf”

Exercises to Lesson ML8

Homework 8.3 – “Exercise of an Example with Python”

***** placeholder *****

Homework H8.4 – “Exercise of an Example with Python”

*****placeholder*****

Solutions are found in Ref. **[HVö-4]**: “Exercises2Lecture.pdf”

Appendix

Appendix-1

1. **ML1: Introduction to Machine Learning (ML)**
2. ML2: Concept Learning: VSpaces & Cand. Elim. Algo.
3. **ML3: Supervised and Unsupervised Learning**
4. ML4: Decision Tree Learning
5. ML5: simple Linear Regression (sLR) & multiple Linear Regression (mLR)
6. **ML6: Neural Networks: Convolutional NN**
7. ML7: Neural Network: BackPropagation Algorithm
8. ML8: Support Vector Machines (SVM)

1. ML1: Introduction to Machine Learning (ML)
2. **ML2: Concept Learning: VSpaces & Cand. Elim. Algo.**
3. ML3: Supervised and Unsupervised Learning
4. ML4: Decision Tree Learning
5. ML5: simple Linear Regression (sLR) & multiple Linear Regression (mLR)
6. **ML6: Neural Networks: Convolutional NN**
7. ML7: Neural Network: BackPropagation Algorithm
8. ML8: Support Vector Machines (SVM)

1. ML1: Introduction to Machine Learning (ML)
2. ML2: Concept Learning: VSpaces & Cand. Elim. Algo.
3. **ML3: Supervised and Unsupervised Learning**
4. **ML4: Decision Tree Learning**
5. ML5: simple Linear Regression (sLR) & multiple Linear Regression (mLR)
6. **ML6: Neural Networks: Convolutional NN**
7. ML7: Neural Network: BackPropagation Algorithm
8. ML8: Support Vector Machines (SVM)

1. ML1: Introduction to Machine Learning (ML)
2. ML2: Concept Learning: VSpaces & Cand. Elim. Algo.
3. **ML3: Supervised and Unsupervised Learning**
4. **ML4: Decision Tree Learning**
5. ML5: simple Linear Regression (sLR) & multiple Linear Regression (mLR)
6. **ML6: Neural Networks: Convolutional NN**
7. ML7: Neural Network: BackPropagation Algorithm
8. ML8: Support Vector Machines (SVM)

1. ML1: Introduction to Machine Learning (ML)
2. ML2: Concept Learning: VSpaces & Cand. Elim. Algo.
3. **ML3: Supervised and Unsupervised Learning**
4. ML4: Decision Tree Learning
5. **ML5: simple Linear Regression (sLR) & multiple Linear Regression (mLR)**
6. **ML6: Neural Networks: Convolutional NN**
7. ML7: Neural Network: BackPropagation Algorithm
8. ML8: Support Vector Machines (SVM)

1. ML1: Introduction to Machine Learning (ML)
2. ML2: Concept Learning: VSpaces & Cand. Elim. Algo.
3. **ML3: Supervised and Unsupervised Learning**
4. ML4: Decision Tree Learning
5. ML5: simple Linear Regression (sLR) & multiple Linear Regression (mLR)
6. **ML6: Neural Networks: Convolutional NN**
7. ML7: Neural Network: BackPropagation Algorithm
8. ML8: Support Vector Machines (SVM)