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

by

Dr. Hermann Völlinger and Other

Status: 22 December 2022

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

Contents

Numbers of Exercises per Chapter

When we count the numbers of the exercises for this document for each chapter of the lesson, we get the following result:

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)	5	0
ML2	*Concept Learning: VSpaces & Cand. Elim. 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	4	2
ML7	*Neural Network: BackPropagation Algorithm*	*2*	*0*
ML8	*ML8: Support Vector Machines*	*4*	*0*
sum		33	9

Links to Further Literature:

1. **[HVö-3]:** Hermann Völlinger: MindMap of the Lecture "Machine Learning: Concepts & Algorithms" "; DHBW Stuttgart; WS2020
2. **[HVö-5]:** Hermann Völlinger: Script of the Lecture "Machine Learning: Concepts & Algorithms"; DHBW Stuttgart; WS2020
3. **[HVö-6]:** Hermann Völlinger: GitHub to the Lecture "Machine Learning: Concepts & Algorithms"; see in: https://github.com/HVoellinger/Lecture-Notes-to-ML-WS2020

Exercises to Lesson ML0: General Remarks and Goals of Lecture (ML)

Homework H0.1- "Three Categories of Machine Learning"

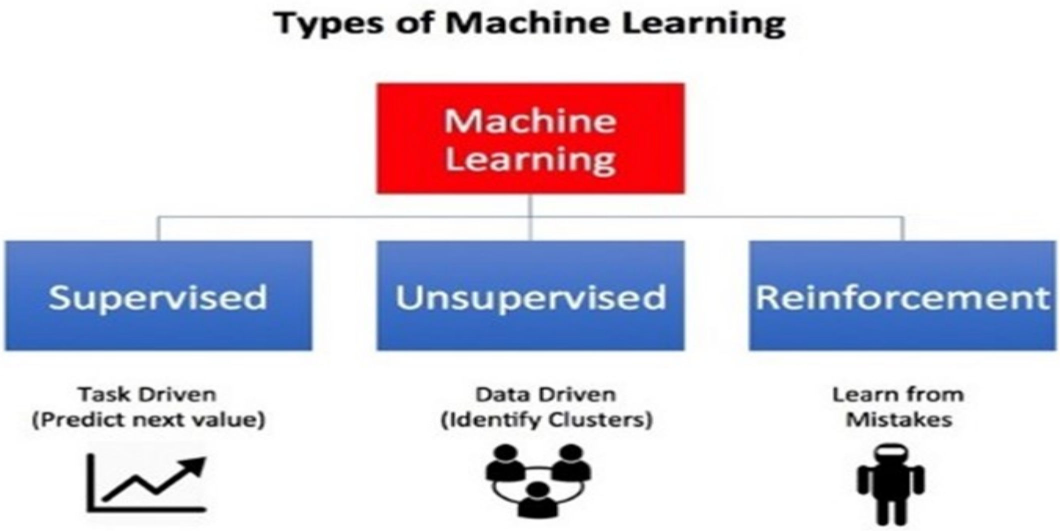
Groupwork (2 Persons). Compare the differences of the three categories, see slide "goal of lecture (2/2)":

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

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).

First Solution:



Supervised	Unsupervised	Reinforcement
Datensatz mit Beschriftung	Reiner Datensatz	Lernt aus Fehlern ➔ viele Fehler am Anfang
Durch üben wird Beschriftung vorhergesagt	Tools lernen die Eigenschaften der Daten zu verstehen	Bewertungen für gute bzw. schlechte Verhaltensweise
Feedback ob die Vorhersage stimmt oder nicht	Tools können die Daten gruppieren, vereinen oder neu anordnen	Perfektionismus über Zeit
Anwendungsgebiet: Entscheidungsfindung für bestimmtes Aufgabengebiet z.B. Gesichtswiedererkennung	Anwendungsgebiet: Mustererkennung z.B. Einkaufsverhalten	Anwendungsgebiet: Abschätzung von Verhaltensmustern z.B. Videospiele

<u>Second Solution</u>: R. Scholz, N. Breuninger; WS2020

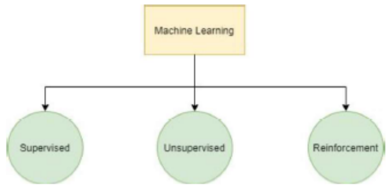
Types of Machine Learning

Homework H0.1.

Agenda

1. Overview
2. Supervised Learning
3. Unsupervised Learning
4. Reinforcement Learning

Overview



Source: own representation

Supervised Learning

- most popular
- easy and simple to implement
- data form: examples with labels
- predict label for example
- feedback if prediction is correct
- trained algorithm predicts label for example
- highly focused on singular task

Source: https://azure.microsoft.com/de-de/services/open-datasets/catalog/mnist/

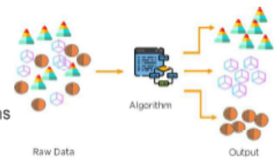
Supervised Learning

Use-Cases:

- Advertisement Popularity
  - search engine
- Spam Classification
  - e-mail
- Face Recognition
  - facebook image tag

Unsupervised Learning

- opposite of supervised learning
- no labels
- group, cluster, and/or organize the data
  - output optimized for humans
- makes suggestions and recommendations
- boost productivity



Source: https://miro.medium.com/max/1198/0*2EFfPMn6BGenVWA

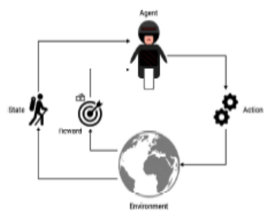
Unsupervised Learning

Use-Cases:

- Recommender Systems
  - video recommendation system
- Buying Habits
  - group customers into similar purchasing segments
- Grouping User Logs
  - group user logs and issues

Reinforcement Learning

- different than previous
- no dataset
- learning by mistakes
- lots of mistakes at beginning
- less errors over time
- signal for positive and negative behavior



Source: https://www.inovex.de/blog/reinforcement-learning-walkthrough-introduction/

Exercises to Lesson ML1: Introduction to Machine Learning (ML)

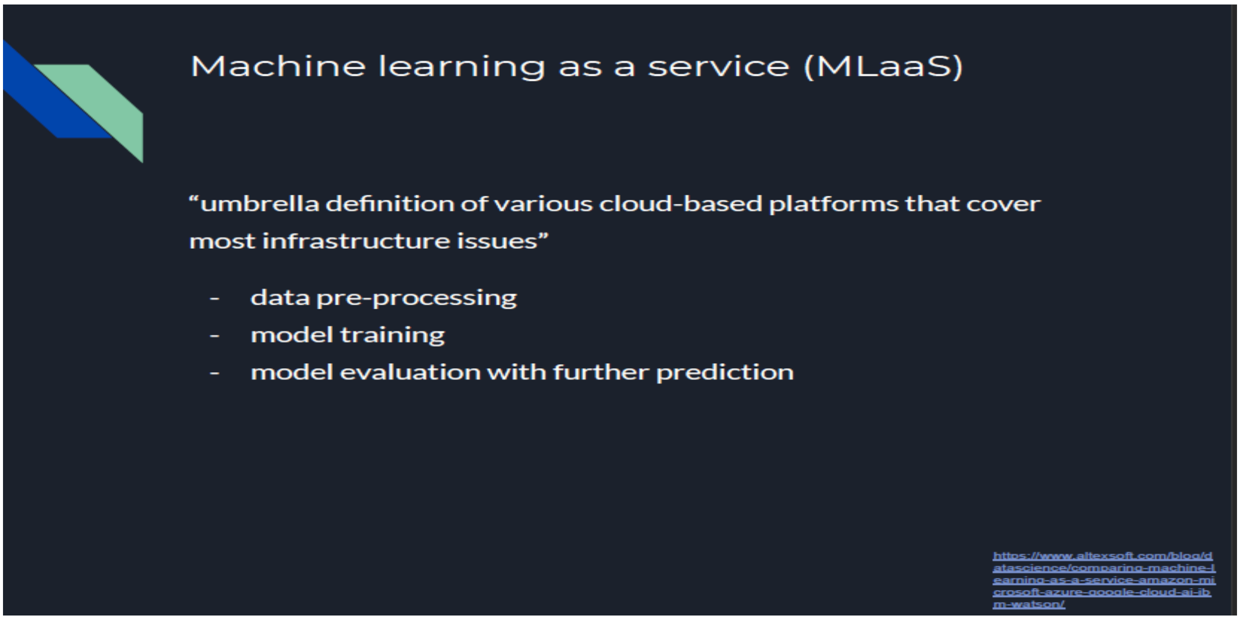
Homework H1.1 - "Most Popular ML Technologies + Products"

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).

First Solution:



Leadingen Service providers

Computerwoche - Teil 3: Anwendungen und Plattformen

- Amazon Machine Learning services
- Azure Machine Learning
- Google Cloud AI
- IBM Watson

CLOUD MACHINE LEARNING SERVICES COMPARISON

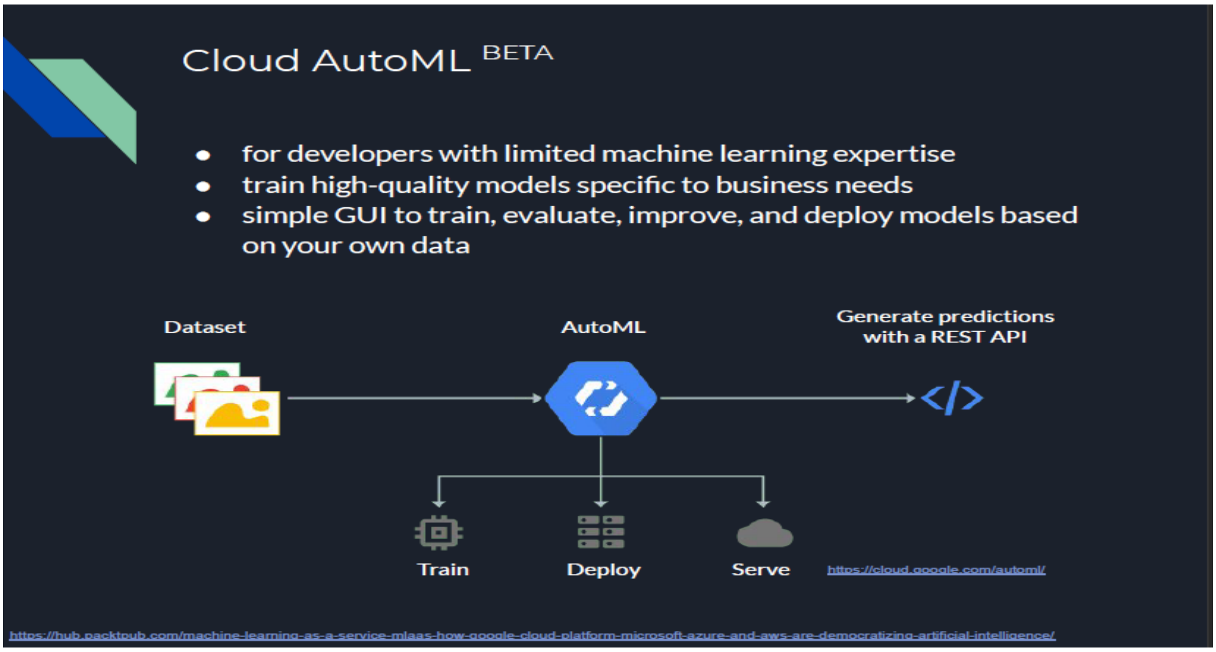
	Amazon	Microsoft	Google	IBM
Automated and semi-automated ML services				
	Amazon ML	Microsoft Azure ML Studio	Google Prediction API	IBM Watson ML Model Builder
Classification	✓	✓	deprecated	✓
Regression	✓	✓		✓
Clustering	✓	✓		✗
Anomaly detection	✗	✓		✗
Recommendation	✗	✓		✗
Ranking	✗	✓		✗
Platforms for custom modeling				
	Amazon SageMaker	Azure ML Services	Google ML Engine	IBM Watson ML Studio
Built-in algorithms	✓	✗	✗	✓
Supported frameworks	TensorFlow, MXNet, Keras, Gluon, Pytorch, Caffe2, Chainer, Torch	TensorFlow, scikit-learn, Microsoft Cognitive Toolkit, Spark ML	TensorFlow, scikit-learn, XGBoost, Keras	TensorFlow, Spark MLlib, scikit-learn, XGBoost, PyTorch, IBM SPSS, PMML

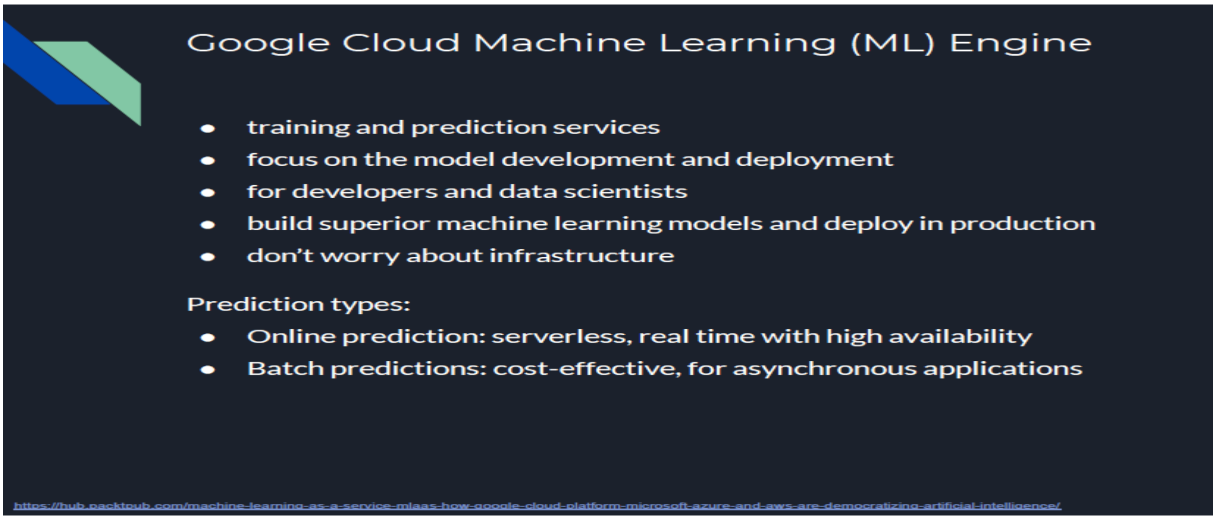
https://www.altexsoft.com/blog/datascience/comparing-machine-learning-as-a-service-amazon-microsoft-azure-google-cloud-ai-ibm-watson/

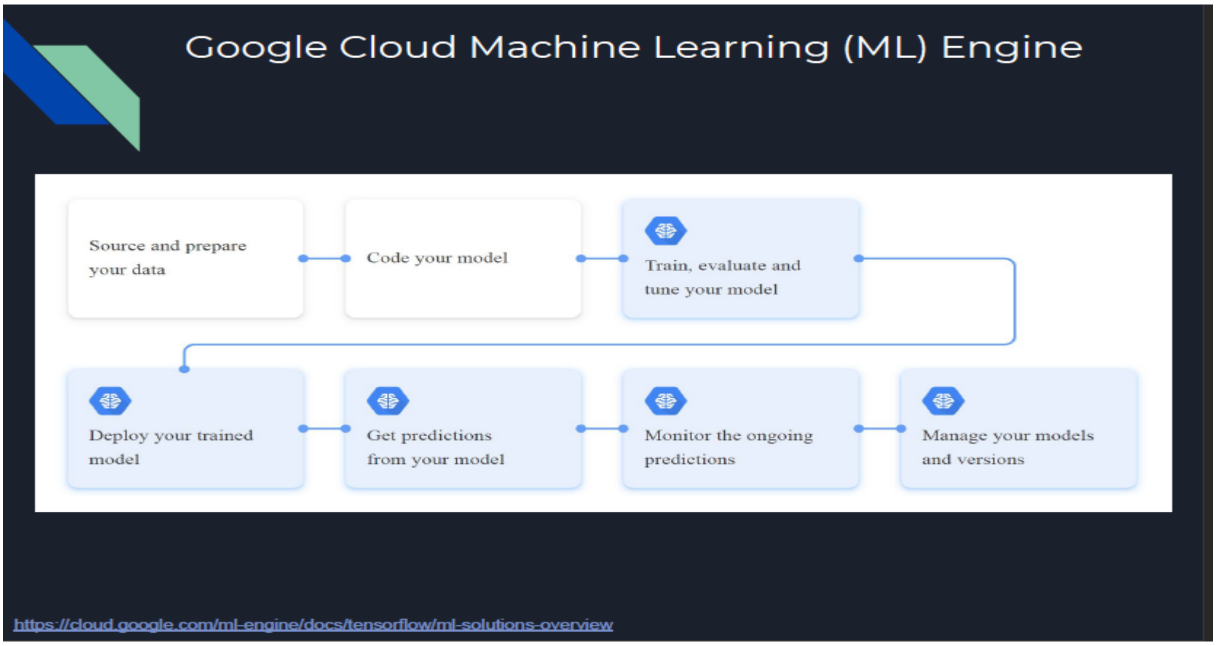
SPEECH AND TEXT PROCESSING APIs COMPARISON

	Amazon	Microsoft	Google	IBM
Speech Recognition (Speech into Text)	✓	✓	✓	✓
Text into Speech Conversion	✓	✓	✓	✓
Entities Extraction	✓	✓	✓	✓
Key Phrase Extraction	✓	✓	✓	✓
Language Recognition	100+ languages	120 languages	120+ languages	60+ languages
Topics Extraction	✓	✓	✓	✓
Spell Check	✗	✓	✗	✗
Autocompletion	✗	✓	✗	✗
Voice Verification	✓	✓	✗	✗
Intention Analysis	✓	✓	✓	✓
Metadata Extraction	✗	✗	✗	✓
Relations Analysis	✗	✓	✗	✓
Sentiment Analysis	✓	✓	✓	✓
Personality Analysis	✗	✗	✗	✓
Syntax Analysis	✗	✓	✓	✓
Tagging Parts of Speech	✗	✓	✓	✗
Filtering Inappropriate Content	✗	✓	✓	✗
Low-quality Audio Handling	✓	✓	✓	✓
Translation	6 languages	60+ languages	100+ languages	21 languages
Chatbot Toolset	✓	✓	✓	✓

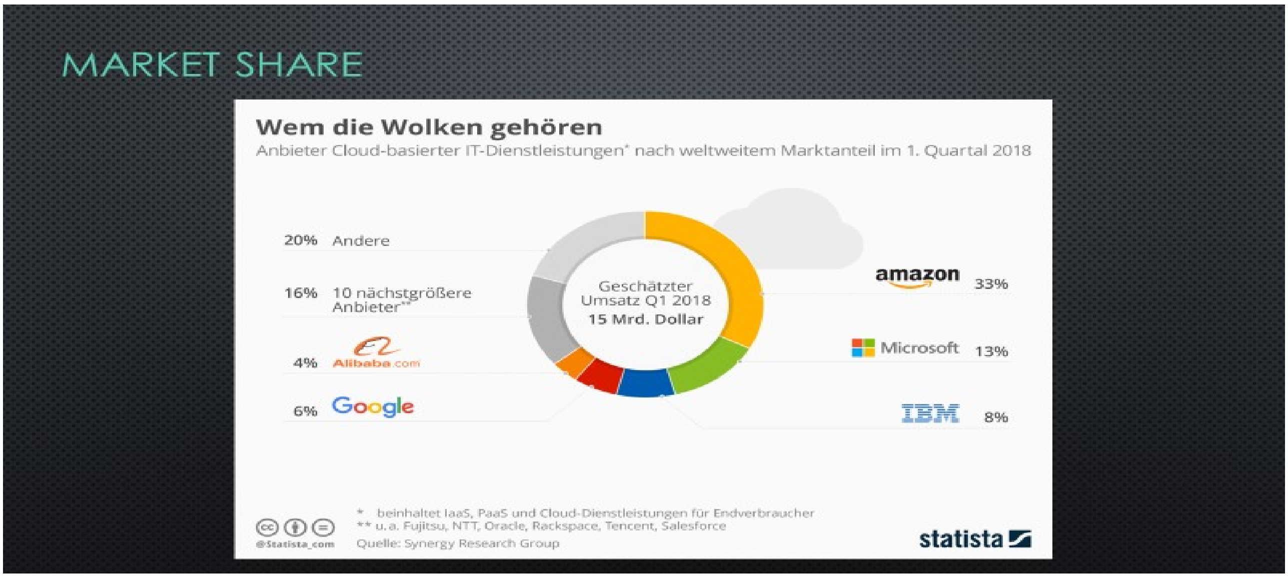
https://www.altexsoft.com/blog/datascience/comparing-machine-learning-as-a-service-amazon-microsoft-azure-google-cloud-ai-ibm-watson/

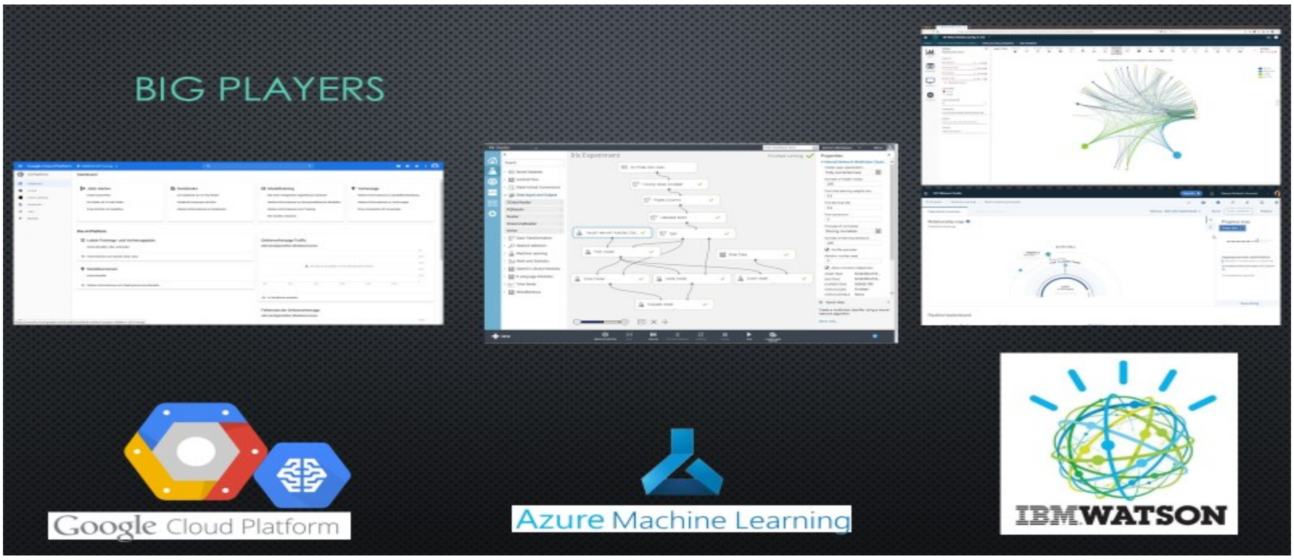

Cloud AutoML BETA
• for developers with limited machine learning expertise
• train high-quality models specific to business needs
• simple GUI to train, evaluate, improve, and deploy models based on your own data
Dataset
AutoML
Generate predictions with a REST API
</>
Train
Deploy
Serve
https://cloud.google.com/automl/
https://hub.packtpub.com/machine-learning-as-a-service-mlaas-how-google-cloud-platform-microsoft-azure-and-aws-are-democratizing-artificial-intelligence/

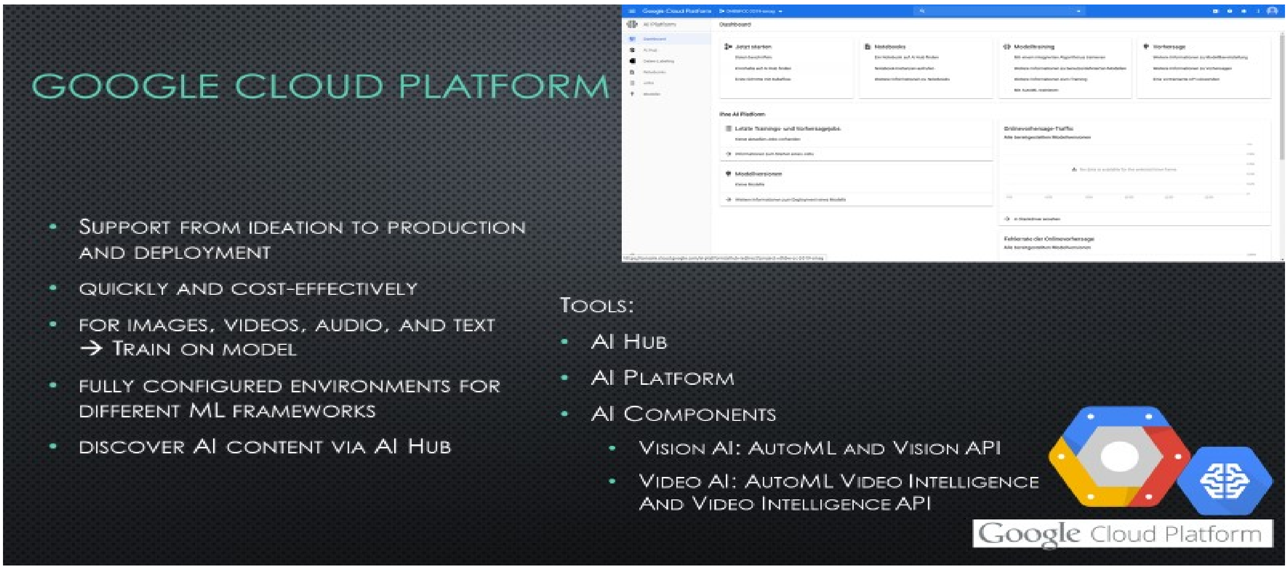

Google Cloud Machine Learning (ML) Engine
• training and prediction services
• focus on the model development and deployment
• for developers and data scientists
• build superior machine learning models and deploy in production
• don't worry about infrastructure
Prediction types:
• Online prediction: serverless, real time with high availability
• Batch predictions: cost-effective, for asynchronous applications
https://hub.packtpub.com/machine-learning-as-a-service-mlaas-how-google-cloud-platform-microsoft-azure-and-aws-are-democratizing-artificial-intelligence/

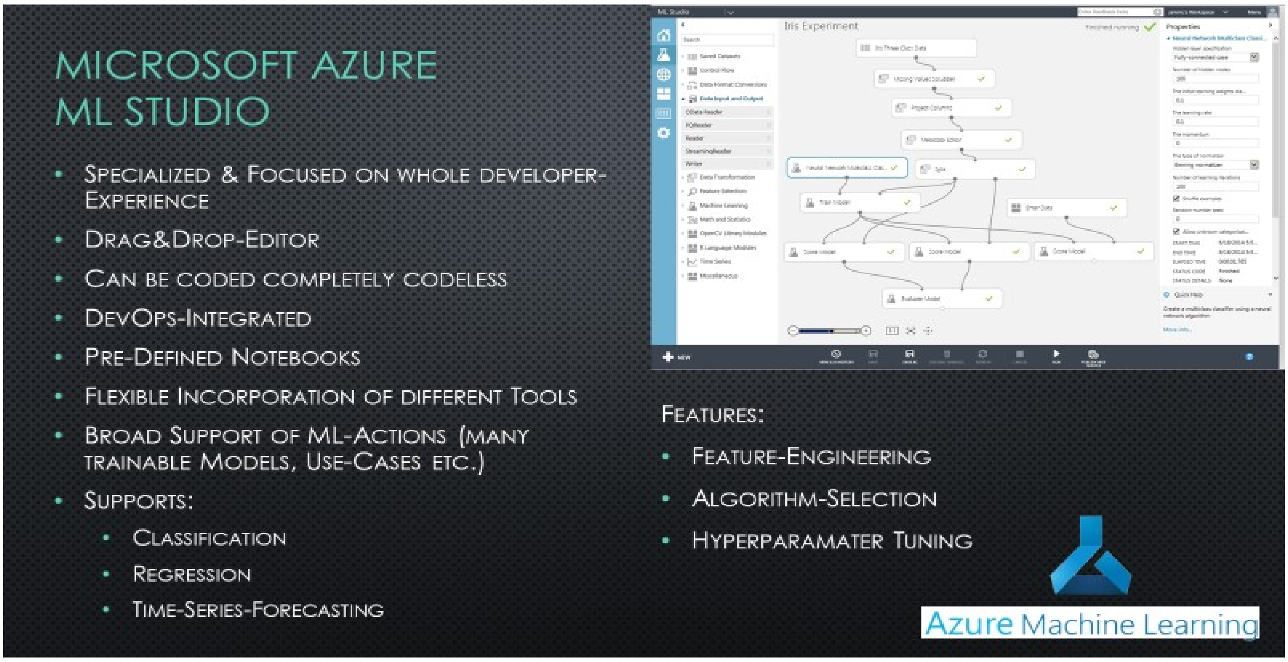

Google Cloud Machine Learning (ML) Engine
Source and prepare your data
Code your model
Train, evaluate and tune your model
Deploy your trained model
Get predictions from your model
Monitor the ongoing predictions
Manage your models and versions
https://cloud.google.com/ml-engine/docs/tensorflow/ml-solutions-overview

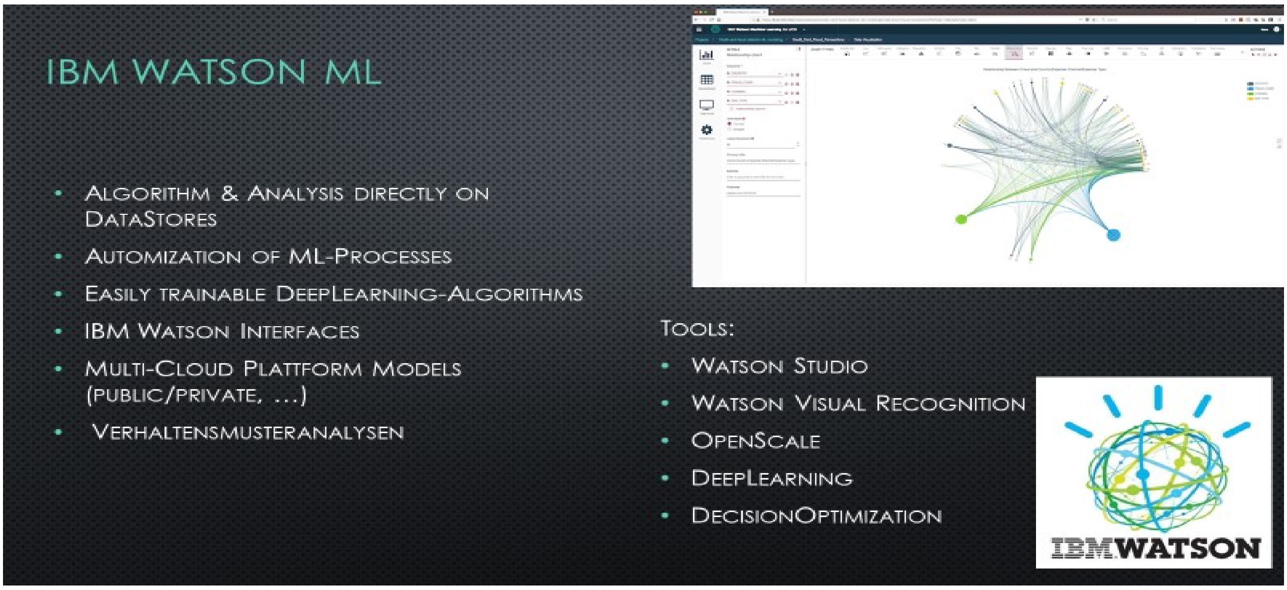
Second Solution:

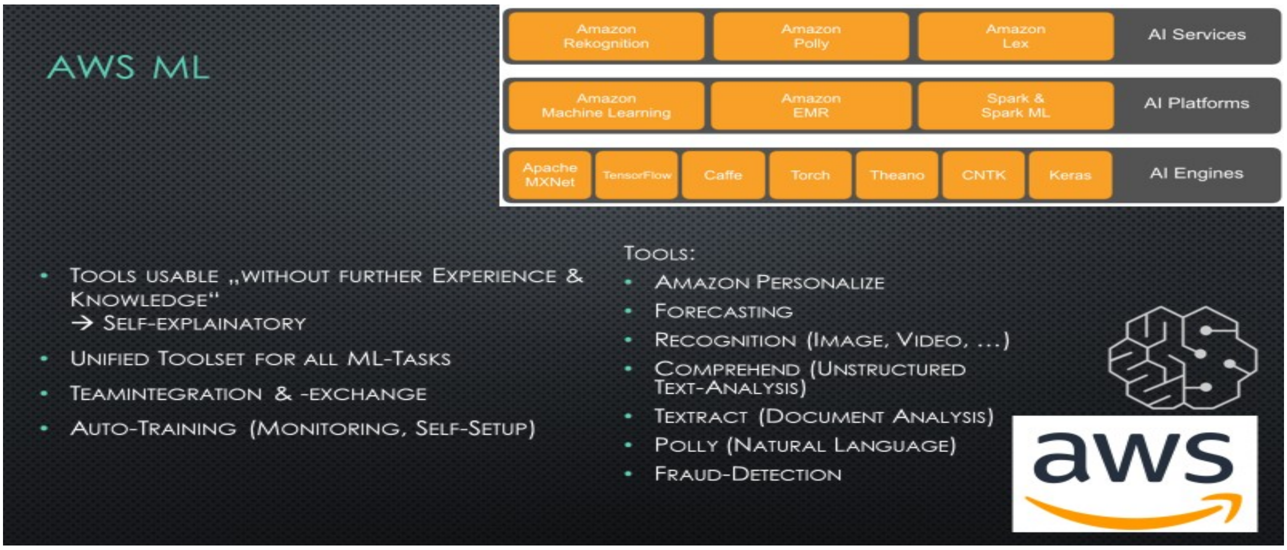


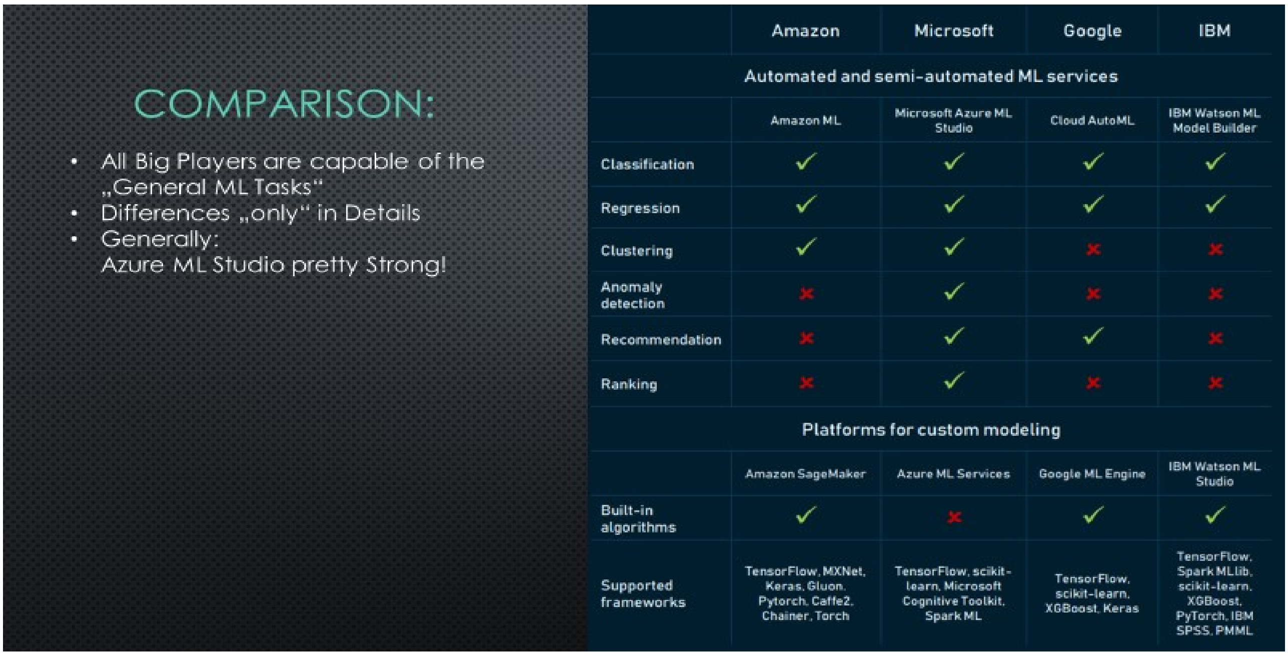

BIG PLAYERS
Google Cloud Platform
Azure Machine Learning
IBM WATSON


GOOGLE CLOUD PLATFORM
• Support from ideation to production and deployment
• quickly and cost-effectively
• for images, videos, audio, and text → Train on model
• fully configured environments for different ML frameworks
• discover AI content via AI Hub
Tools:
• AI Hub
• AI Platform
• AI Components
• Vision AI: AutoML and Vision API
• Video AI: AutoML Video Intelligence and Video Intelligence API
Google Cloud Platform


MICROSOFT AZURE ML STUDIO
• Specialized & Focused on whole developer-Experience
• Drag&Drop-Editor
• Can be coded completely codeless
• DevOps-Integrated
• Pre-Defined Notebooks
• Flexible Incorporation of different Tools
• Broad Support of ML-Actions (many trainable Models, Use-Cases etc.)
• Supports:
• Classification
• Regression
• Time-Series-Forecasting
Features:
• Feature-Engineering
• Algorithm-Selection
• Hyperparamater Tuning
Azure Machine Learning







	Amazon	Microsoft	Google	IBM
Automated and semi-automated ML services				
	Amazon ML	Microsoft Azure ML Studio	Cloud AutoML	IBM Watson ML Model Builder
Classification	✓	✓	✓	✓
Regression	✓	✓	✓	✓
Clustering	✓	✓	✗	✗
Anomaly detection	✗	✓	✗	✗
Recommendation	✗	✓	✓	✗
Ranking	✗	✓	✗	✗
Platforms for custom modeling				
	Amazon SageMaker	Azure ML Services	Google ML Engine	IBM Watson ML Studio
Built-in algorithms	✓	✗	✓	✓
Supported frameworks	TensorFlow, MXNet, Keras, Gluon, Pytorch, Caffe2, Chainer, Torch	TensorFlow, scikit-learn, Microsoft Cognitive Toolkit, Spark ML	TensorFlow, scikit-learn, XGBoost, Keras	TensorFlow, Spark MLlib, scikit-learn, XGBoost, PyTorch, IBM SPSS, PMML

SPEECH AND TEXT PROCESSING APIs COMPARISON

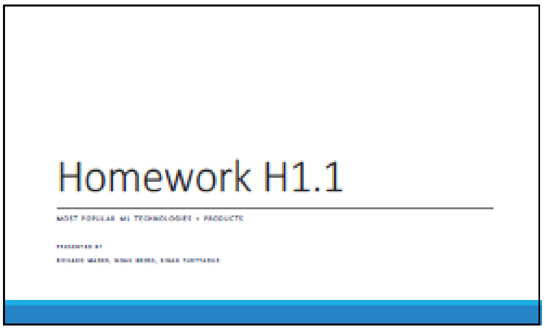
	Amazon	Microsoft	Google	IBM
Speech Recognition (Speech into Text)	✓	✓	✓	✓
Text into Speech Conversion	✓	✓	✓	✓
Entities Extraction	✓	✓	✓	✓
Key Phrase Extraction	✓	✓	✓	✓
Language Recognition	100+ languages	120 languages	120+ languages	60+ languages
Topics Extraction	✓	✓	✓	✓
Spell Check	✗	✓	✗	✗
Autocompletion	✗	✓	✗	✗
Voice Verification	✓	✓	✗	✗
Intention Analysis	✓	✓	✓	✓
Metadata Extraction	✗	✗	✗	✓
Relations Analysis	✗	✓	✗	✓
Sentiment Analysis	✓	✓	✓	✓
Personality Analysis	✗	✗	✗	✓
Syntax Analysis	✗	✓	✓	✓
Tagging Parts of Speech	✗	✓	✓	✗
Filtering Inappropriate Content	✗	✓	✓	✗
Low-quality Audio Handling	✓	✓	✓	✓
Translation	6 languages	60+ languages	100+ languages	21 languages
Chatbot Toolset	✓	✓	✓	✓

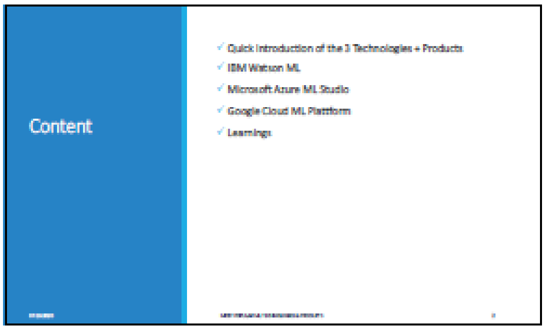
IMAGE ANALYSIS APIs COMPARISON

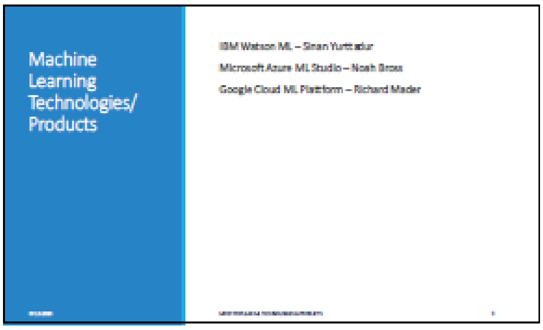
	Amazon	Microsoft	Google	IBM
Object Detection	✓	✓	✓	✓
Scene Detection	✓	✓	✓	✗
Face Detection	✓	✓	✓	✓
Face Recognition (person face identification)	✓	✓	✓	✗
Facial Analysis	✓	✓	✓	✓
Inappropriate Content Detection	✓	✓	✓	✓
Celebrity Recognition	✓	✓	✓	✗
Text Recognition	✓	✓	✓	✓
Written Text Recognition	✓	✓	✓	✗
Search for Similar Images on Web	✗	✓	✓	✗
Logo Detection	✗	✗	✓	✗
Landmark Detection	✗	✓	✓	✗
Food Recognition	✗	✗	✗	✓
Dominant Colors Detection	✗	✓	✓	✗

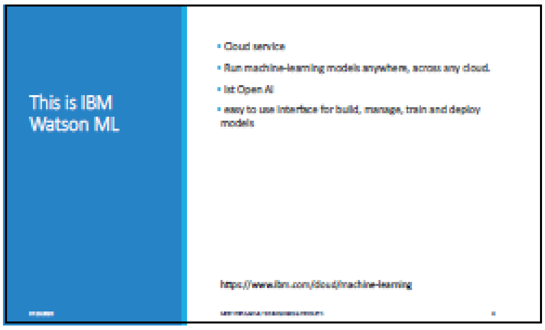
Third Solution: R. Mader, N. Bross, S Yurttadur; WS2020:

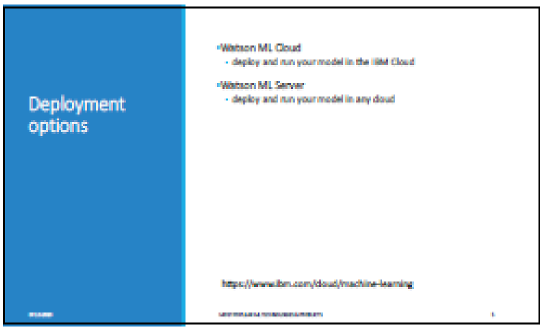
Richard Mader, Noah Bross, Sinan Yurttadur 07.10.2020

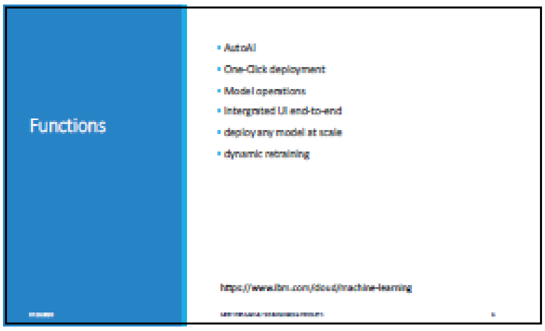


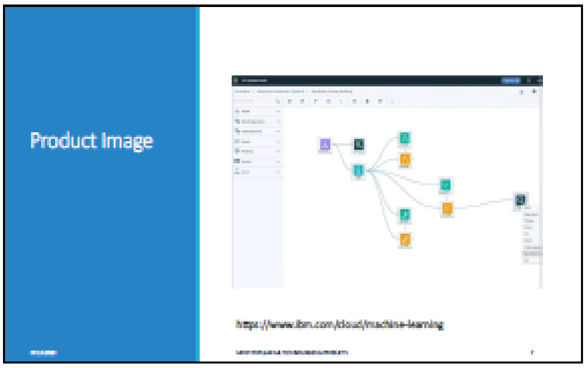




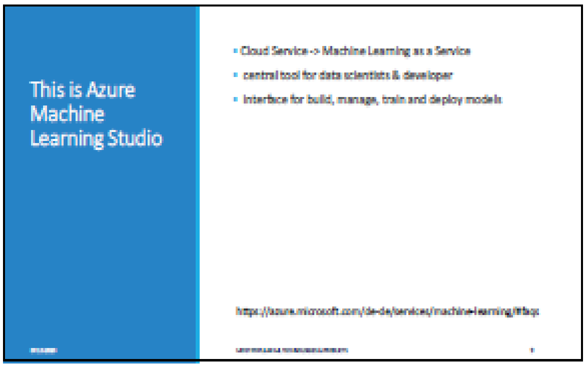


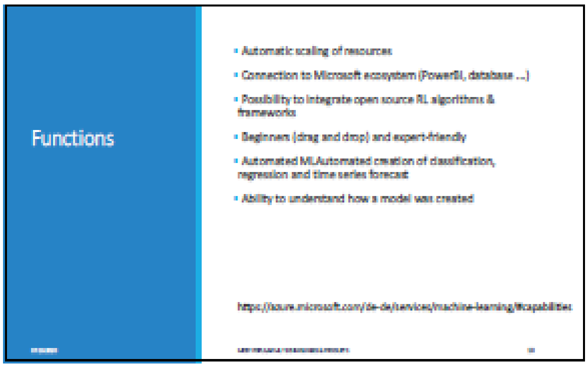


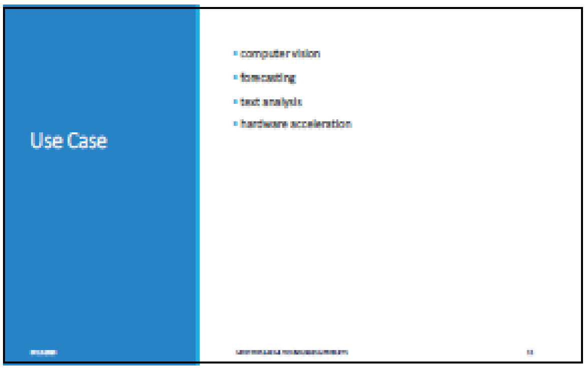


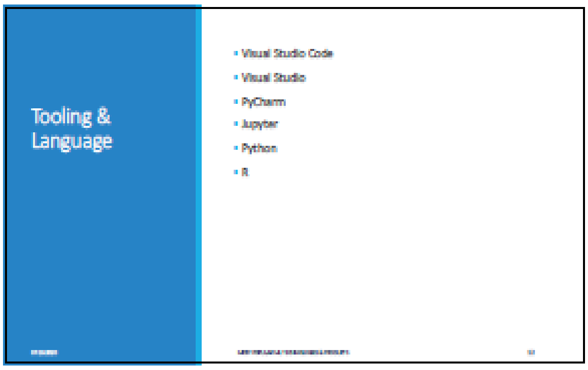

Product Image
https://www.ibm.com/cloud/machine-learning

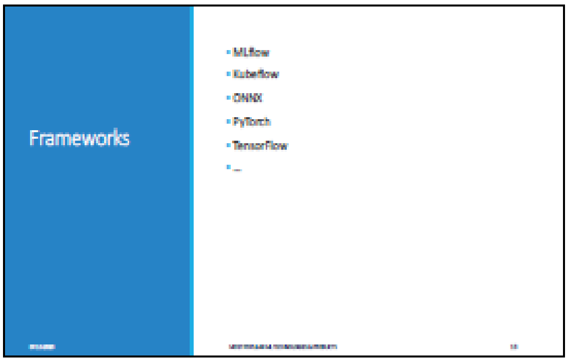

Compare IBM ML
https://www.ibm.com/cloud/machine-learning

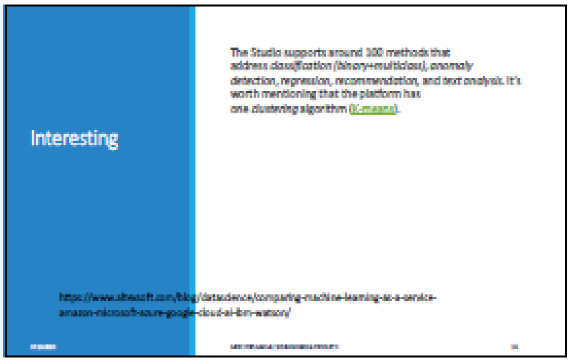

This is Azure Machine Learning Studio
Cloud Service-> Machine Learning as a Service
central tool for data scientists & developer
Interface for build, manage, train and deploy models
https://azure.microsoft.com/de-de/services/machine-learning/#faqs


Functions
Automatic scaling of resources
Connection to Microsoft ecosystem (PowerBI, database ...)
Possibility to integrate open source RL algorithms & frameworks
Beginners (drag and drop) and expert-friendly
Automated ML Automated creation of classification, regression and time series forecast
Ability to understand how a model was created
https://azure.microsoft.com/de-de/services/machine-learning/#capabilities

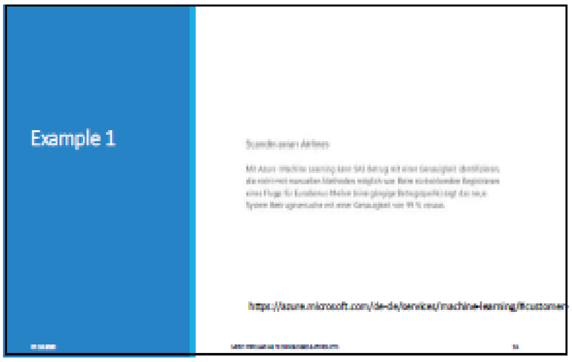

Use Case
computer vision
forecasting
text analysis
hardware acceleration

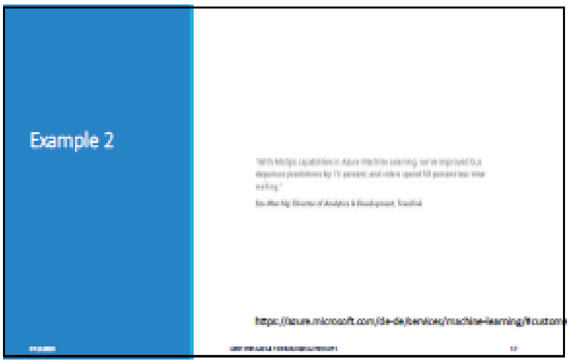

Tooling & Language
Visual Studio Code
Visual Studio
PyCharm
Jupyter
Python
R


Frameworks
MLflow
Kubeflow
ONNX
PyTorch
TensorFlow
…

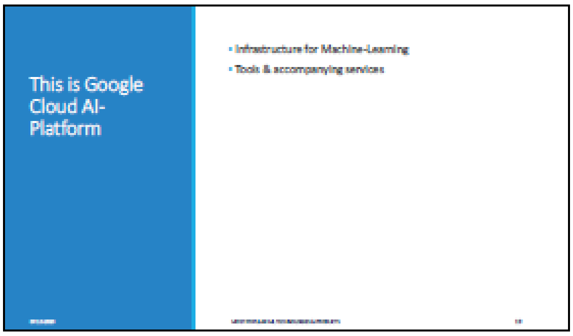

Interesting
The Studio supports around 100 methods that address classification (binary+multiclass), anomaly detection, regression, recommendation, and text analysis. It's worth mentioning that the platform has one clustering algorithm (K-means).
https://www.altexsoft.com/blog/datascience/comparing-machine-learning-as-a-service-amazon-microsoft-azure-google-cloud-ai-ibm-watson/

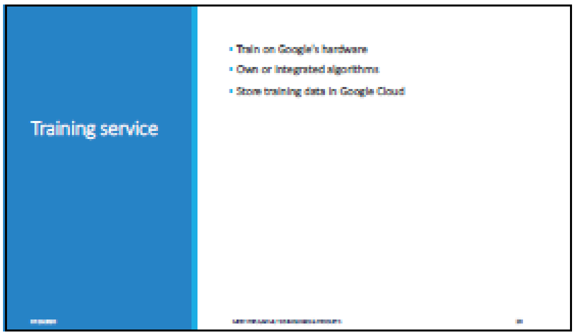

Pricing
https://azure.microsoft.com/de-de/services/machine-learning/#pricing

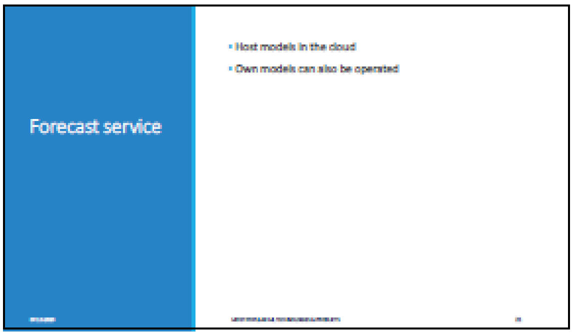

Example 1
https://azure.microsoft.com/de-de/services/machine-learning/#customer

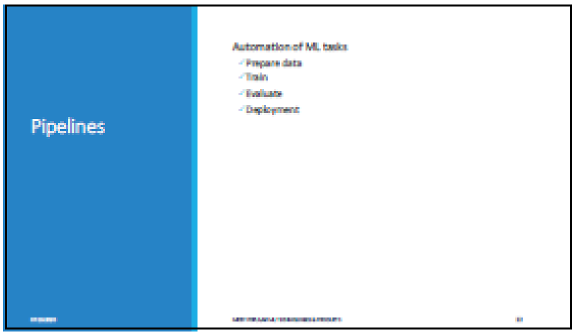

Example 2
https://azure.microsoft.com/de-de/services/machine-learning/#custom

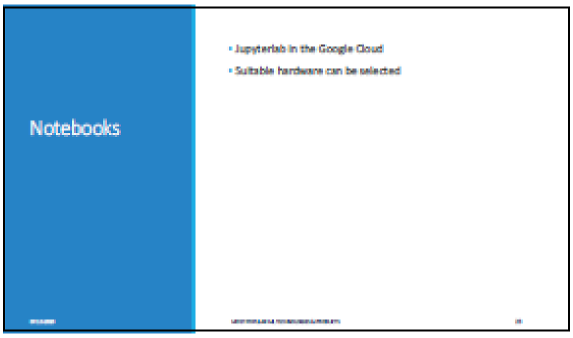

Popularity
Interest over time
https://stackshare.io/stackups/amazon-machine-learning-vs-azure-machine-learning

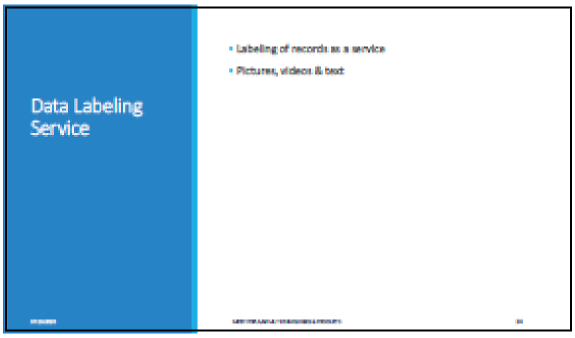

This is Google Cloud AI-Platform
• Infrastructure for Machine-Learning
• Tools & accompanying services

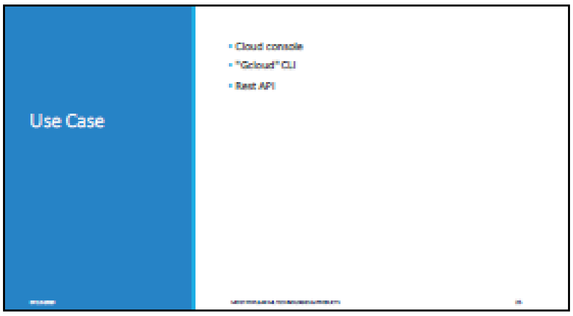

Training service
• Train on Google's hardware
• Own or integrated algorithms
• Store training data in Google Cloud

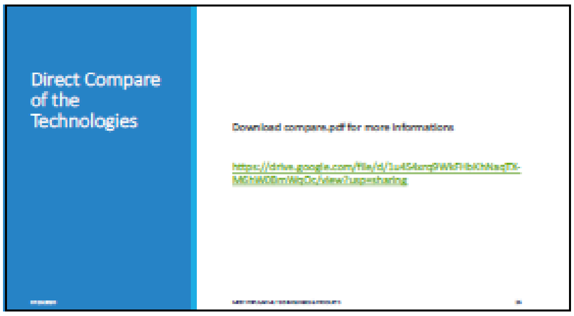

Forecast service
• Host models in the cloud
• Own models can also be operated


Pipelines
Automation of ML tasks
✓ Prepare data
✓ Train
✓ Evaluate
✓ Deployment


Notebooks
• Jupyterlab in the Google Cloud
• Suitable hardware can be selected


Data Labeling Service
• Labeling of records as a service
• Pictures, videos & text


Use Case
• Cloud console
• "Gcloud" CLI
• Rest API

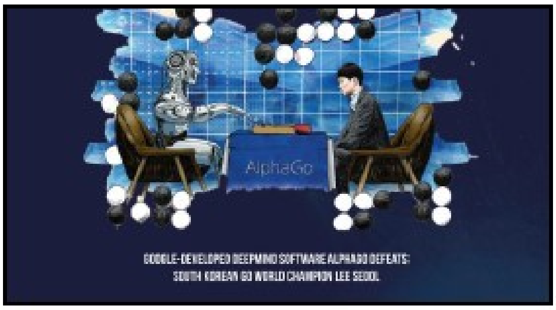

Direct Compare of the Technologies
Download compare.pdf for more informations

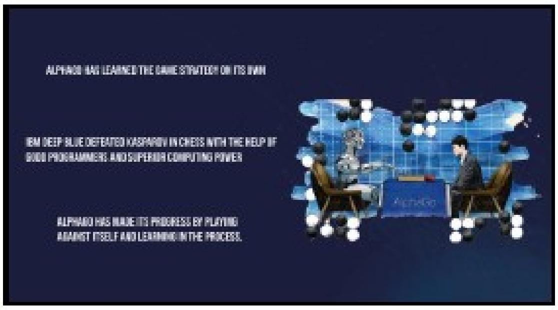
Homework H1.2 - "Ethics in Artificial Intelligence"

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.

Solution: B. Storz, L. Mack; WS2020:


ETHICS IN ARTIFICIAL INTELLIGENCE
HOMEWORK H1.2


AlphaGo
GOOGLE-DEVELOPED DEEPMIND SOFTWARE ALPHAGO DEFEATS:
SOUTH KOREAN GO WORLD CHAMPION LEE SEDOL


ALPHAGO HAS LEARNED THE GAME STRATEGY ON ITS OWN
IBM DEEP BLUE DEFEATED KASPAROV IN CHESS WITH THE HELP OF GOOD PROGRAMMERS AND SUPERIOR COMPUTING POWER
ALPHAGO HAS MADE ITS PROGRESS BY PLAYING AGAINST ITSELF AND LEARNING IN THE PROCESS.


ALPHAGO HAS LEARNED THE GAME STRATEGY ON ITS OWN
IBM DEEP BLUE DEFEATED KASPAROV IN CHESS WITH THE HELP OF GOOD PROGRAMMERS AND SUPERIOR COMPUTING POWER
ALPHAGO HAS MADE ITS PROGRESS BY PLAYING AGAINST ITSELF AND LEARNING IN THE PROCESS.
DEEP NEURAL NETWORKS (DNN)


Go was considered invincible by computers
Based on many levels of abstraction patterns
Players act a lot from intuition
1985
prize money of 40 million Taiwanese dollars
AlphaGo already won against the European Go champion
differences in playing strength are huge


The real world is much more complex
→ there are many different inputs and outputs
→ the results of actions may not only be slow but also ambiguous
Go is a game with simple rules, easy actions and quick feedback "win/loss".
Feedback is essential for DNN

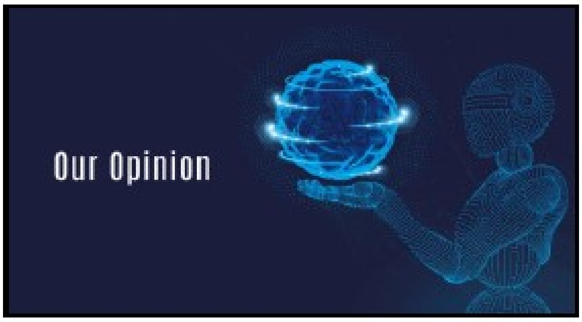

Current vs. Future
CURRENT
Breakthroughs in several practical applications
→ speech recognition
FUTURE
Algo trading → stock market
→ high complexity
→ simple input & output
→ clear measure of success


Whoever controls the money can control people, industries, states
Makes more profits by playing lazily
→ Taking companies to the brink of ruin
→ Buying them cheaply
DNN - ALGO TRADING
DANGEROUS?
Our general moral concepts are not "human nature", but have been learned (Nietzsche)
Autonomously learning DNN can lead to completely different moral concepts


BE CREATIVE
Go Style is kind of "creative" - but there's long way of PC on creating sth. real on their own
Computers have to understand lots of things what we are learning unconsciously since our childhood
First: autonomous scientific research - later they will learn to rethink and create new ideas


He thinks 2035 - if we let them.
We don't have to submit our position as „leaders"
Wise political decisions are needed - quick

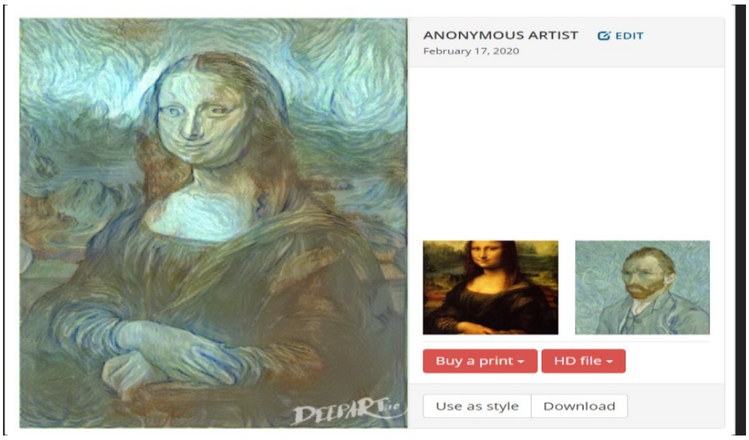

Great danger that DNN will take over the stock market & not hand back the control
01
02
03
Isn't easy to make laws because every country decides on their own
Political Decisions

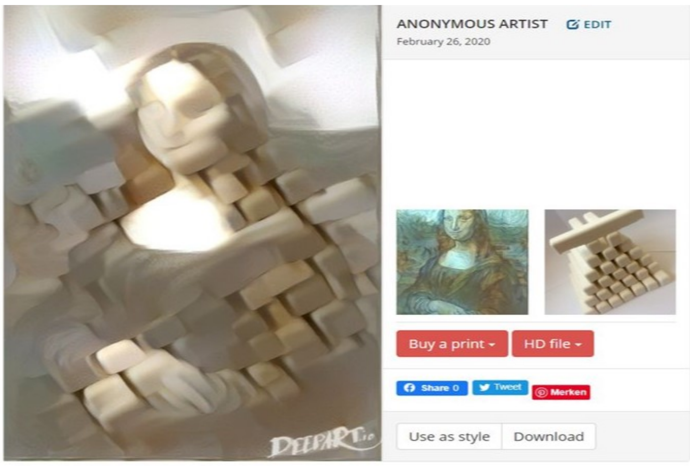

Our Opinion

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”?

Solutions:





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

1 Person - summaries the results of the first YouTupe 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=PLLssT5z_DsK-h9vYZkQkYNWcltqhlRJLN

Solutions:

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

Groupwork (2 Persons) - summaries the results of the second and third YouTupe 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=PLLssT5z_DsK-h9vYZkQkYNWcltqhlRJLN

Solutions:

Supervised-learning
VS
Unsupervised-learning

a glorious presentation by Marc

Agenda

1. Intro
2. Supervised Learning
3. Examples for Supervised Learning
4. Unsupervised Learning
5. Example for Unsupervised Learning
6. SEMI-SUPERVISED LEARNING

Machine Learning

Supervised learning

- Deutsch: Überwachtes lernen
- **Wir haben** strukturierte Daten
- **Wir haben** einen Input X und einen Output y (**KLEIN Y**!!!!!)
- **Wir trainieren** das Netzwerk mit Beispieldaten (X,y)
- **Wir benutzen** das Netzwerk:
  - X reinstecken
  - y kommt raus

X und y

Stellen wir uns vor wir haben 10000 Datensätze

Input	Output	Datenmenge
X_train	y_train	75% (7500 Datensätze)
X_val	y_val	15% (1500 Datensätze)
X_test	X_test	10% (1000 Datensätze)

Supervised Learning - Arten

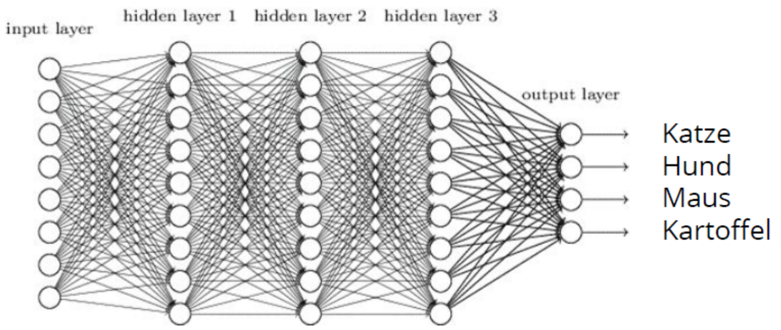
- wir unterscheiden zwischen Categorical und Regression

Categorical Recognition	Regression
- es gibt nur X Lösungsmöglichkeiten - Das Netz soll später zwischen den Lösungsmöglichkeiten unterscheiden	- eine Zahl abhängig von den Input-Daten kommt aus dem Netzwerk

Supervised Learning - Categorical Recognition

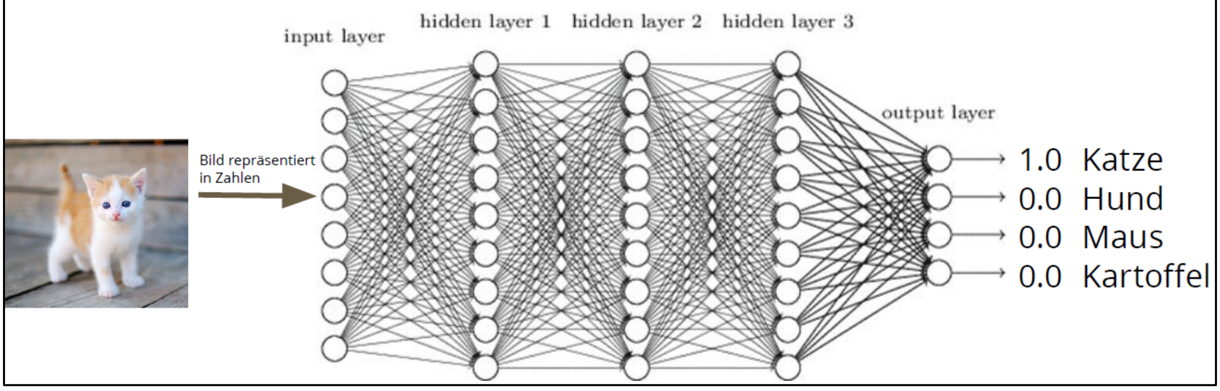
- Wir haben **Bilder aus dem Garten**
- Wir haben **4 Ordner**, mit denen wir zwischen Bildern **unterscheiden**
- Wir wollen unterscheiden zwischen {Katze, Hund, Maus, Kartoffel}

Neuronen-ID: #1 #2 #3 #4



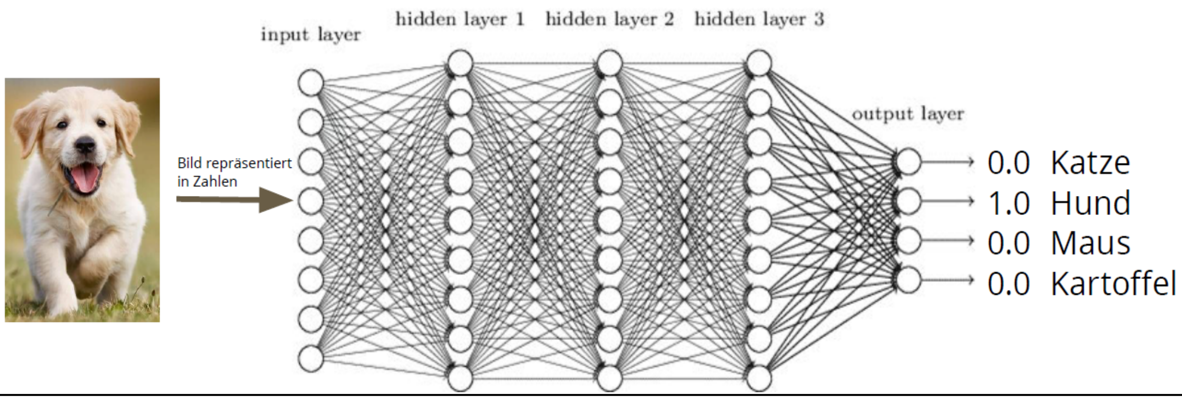
Supervised Learning - Categorical Recognition

- Katze **TRAINING**



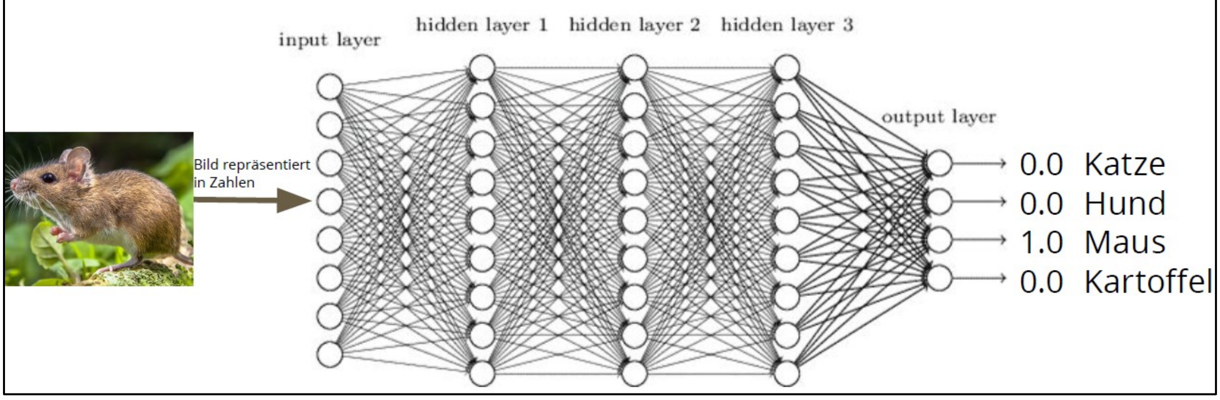
Supervised Learning - Categorical Recognition

- Hund **TRAINING**



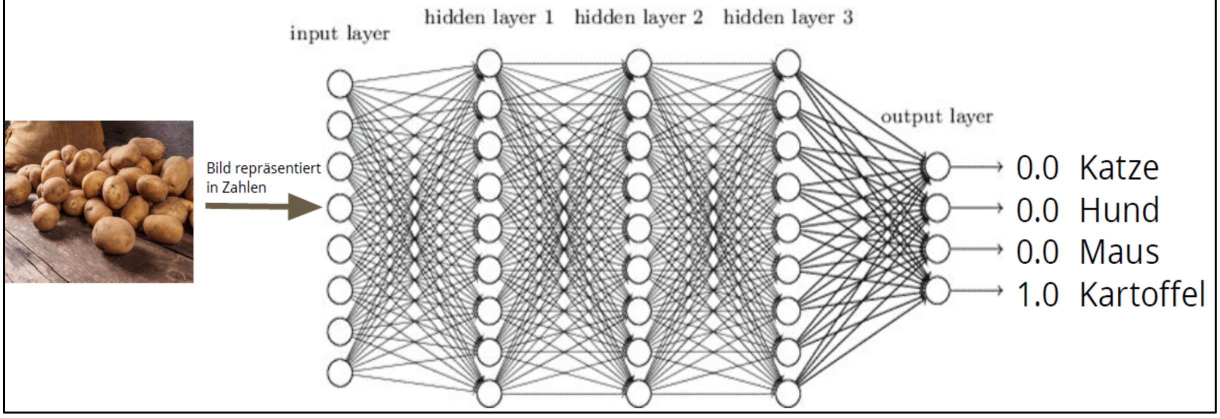
Supervised Learning - Categorical Recognition

- Maus **TRAINING**



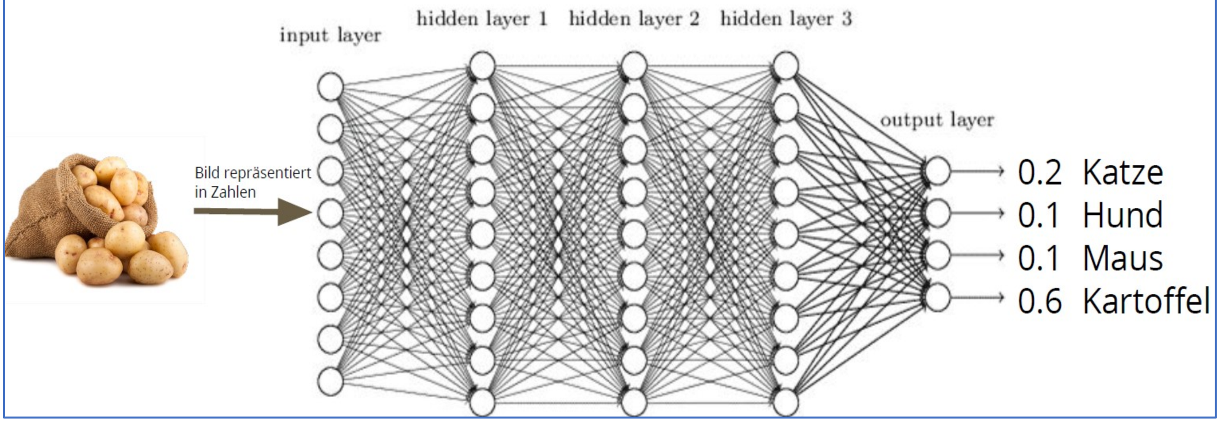
Supervised Learning - Categorical Recognition

- Kartoffel **TRAINING**



Supervised Learning - Categorical Recognition

- Kartoffel **ERKENNUNG**



Supervised Learning - Arten

Categorical Recognition	**Regression**
- es gibt nur X Lösungen - Wir versuchen später zwischen den X-Dingen zu unterscheiden → Das mit der Katze	- eine Zahl abhängig von den Input-Daten kommt aus dem Netzwerk

Supervised Learning - Regression

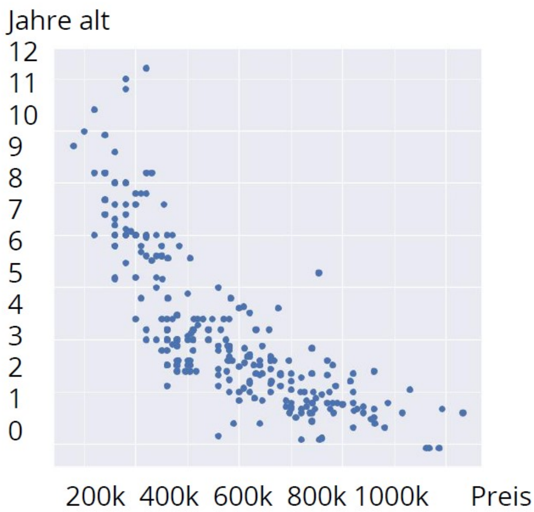
- bei der Regression versuchen wir einen numerischen Wert vorherzusagen
- Beispiel: Price-Prediction
- "Housing Prices Dataset":
    - 80 Spalten/Features (Numerical, String (categorical meistens))
    - 2920 Datensätze

PoolArea
A PoolQC
A Fence
A MiscFeature
MiscVal
MoSold
YrSold
A SaleType
A SaleCondition

Dataset: https://www.kaggle.com/alphaepsilon/housing-prices-dataset

Supervised Learning - Regression - Scikit

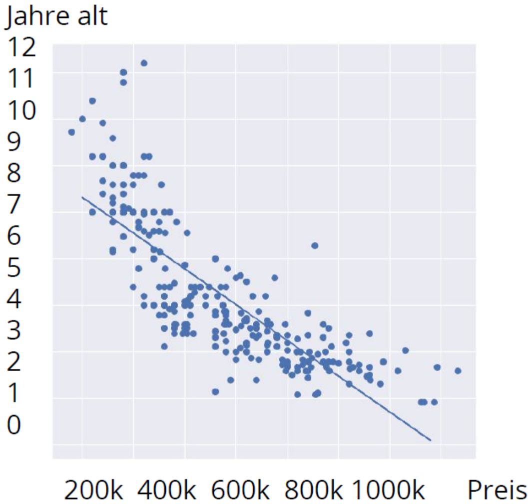
- Wir nehmen an:
  Y-Achse = Alter des Hauses
  X-Achse = Preis



Supervised Learning - Regression - Scikit

- Wir nehmen an:
  Y-Achse = Alter des Hauses
  X-Achse = Preis

Regression Linear

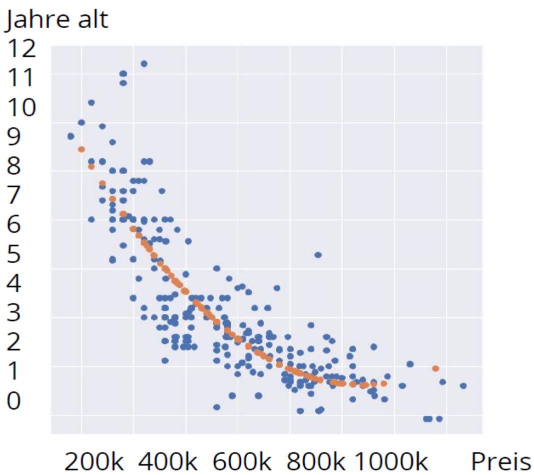


Supervised Learning - Regression - Scikit

- Wir nehmen an:
  Y-Achse = Alter des Hauses
  X-Achse = Preis

Regression Polynomial

(das orangene ist unsere “Kurve”)



Supervised Learning - Regression - Scikit

Vorteile:
- super einfach umzusetzen (5 Zeilen Code in Python)
- einfach zu testen und zu plotten
- sehr schnell “trainiert”

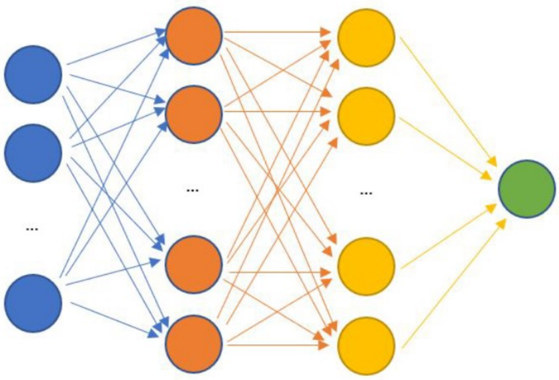
Probleme:
1. wir haben **nur eines** der 80 Input-Daten verwendet
2. wir bilden mehr oder weniger nur einen **Durchschnitt**

Supervised Learning - Regression - DNN

Input Data:

Categorical Input:
1.0 hat Küche
1.0 hat Bad

Numerical Input
0.5 Baujahr
0.4 Fläche



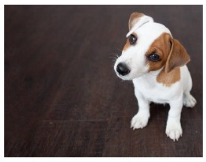
Output Data:

Preis (0.5 = 1000 Euro)

Unsupervised Learning

- Wir wissen nichts/wenig über die Daten ODER
- die Daten sind nicht gelabelt
- Beispieldatensatz:



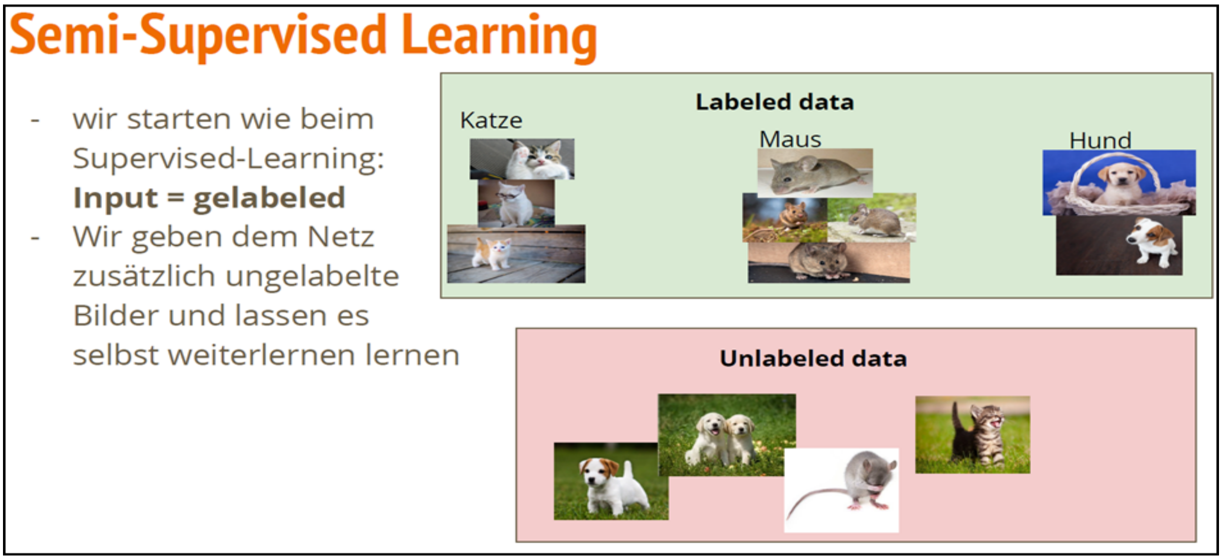





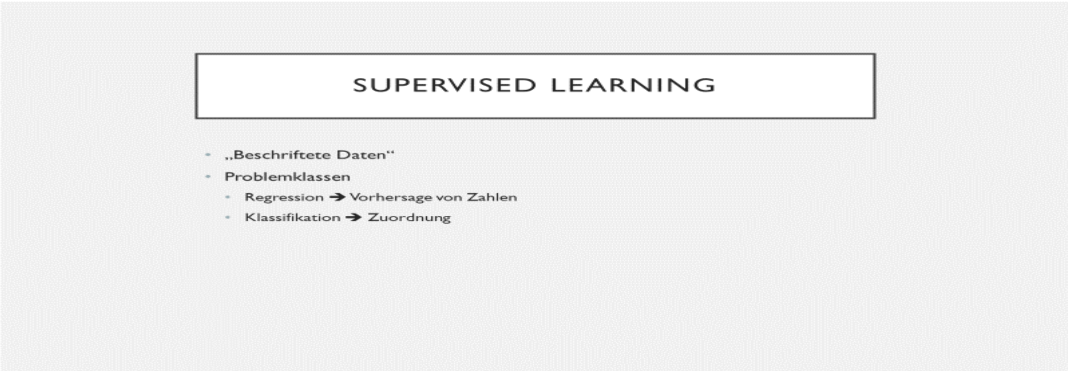


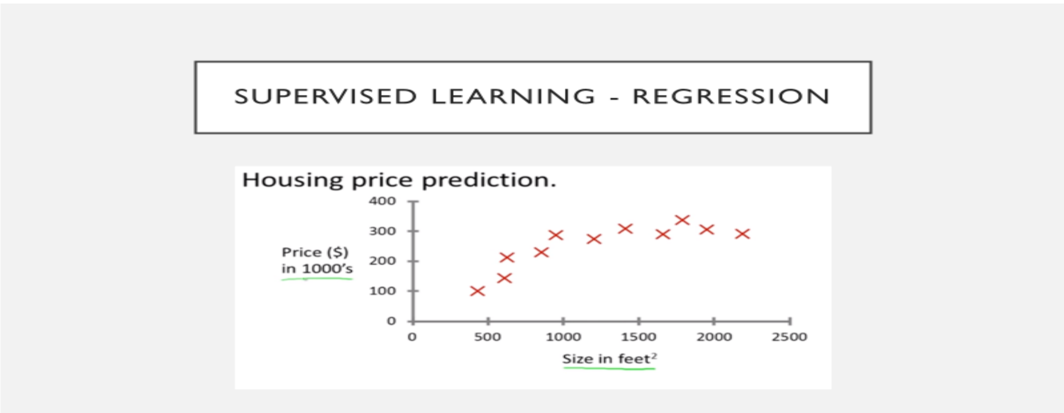
Unsupervised Learning
- Was passieren soll:
Klasse B
Klasse A
Klasse C

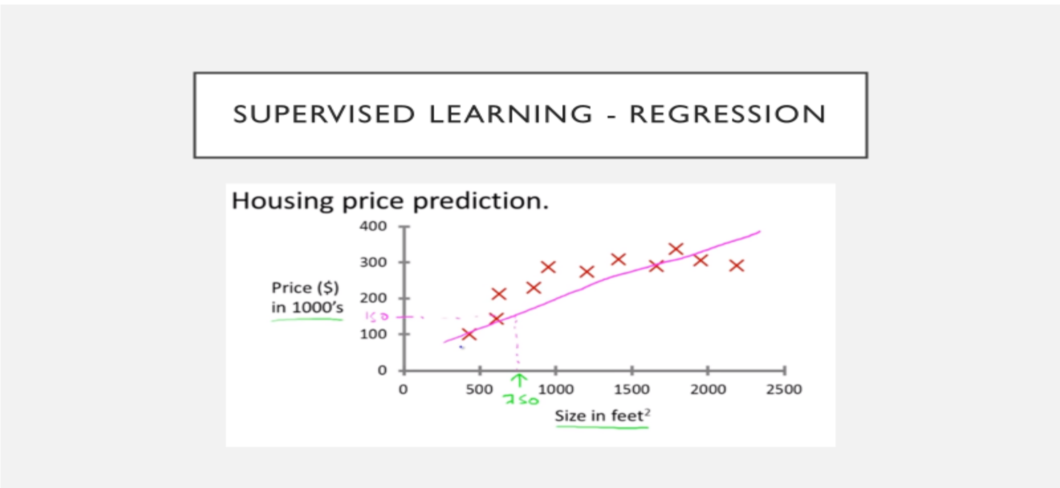

Unsupervised Learning
- Was häufig passiert:
Klasse B
Klasse A
Klasse C


Semi-Supervised Learning
- wir starten wie beim Supervised-Learning: Input = gelabeled
- Wir geben dem Netz zusätzlich ungelabelte Bilder und lassen es selbst weiterlernen lernen
Labeled data
Katze
Maus
Hund
Unlabeled data

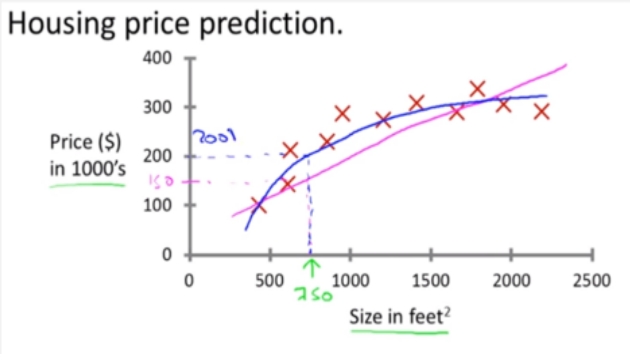
Second Solution:



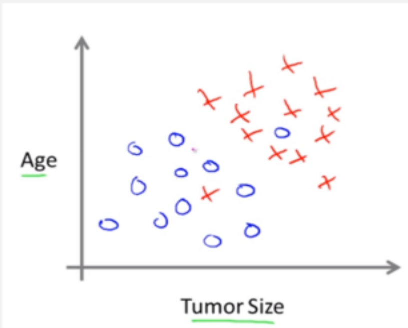




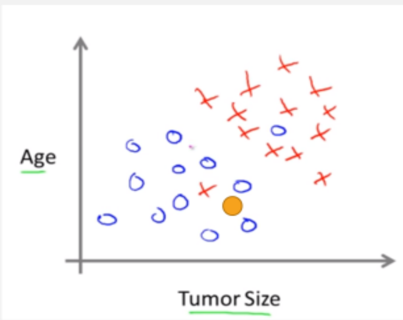
SUPERVISED LEARNING - REGRESSION

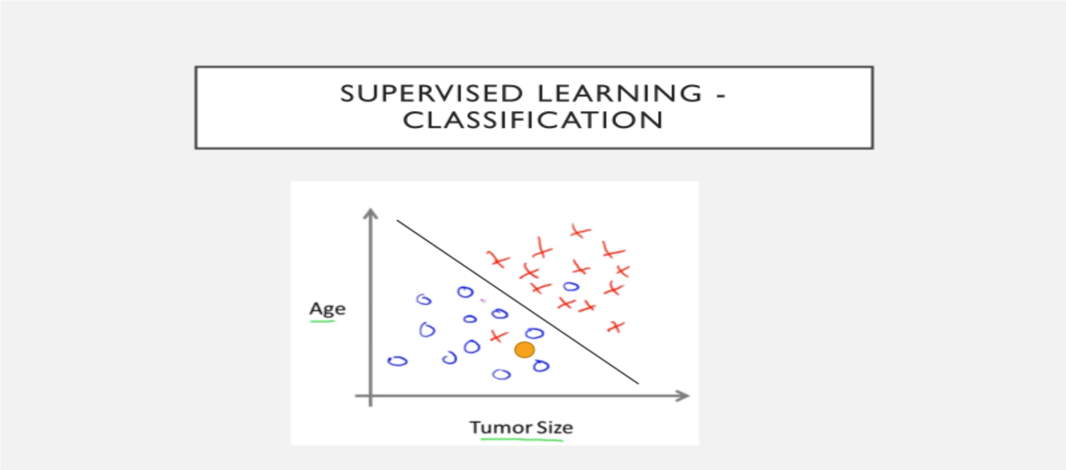


SUPERVISED LEARNING - CLASSIFICATION

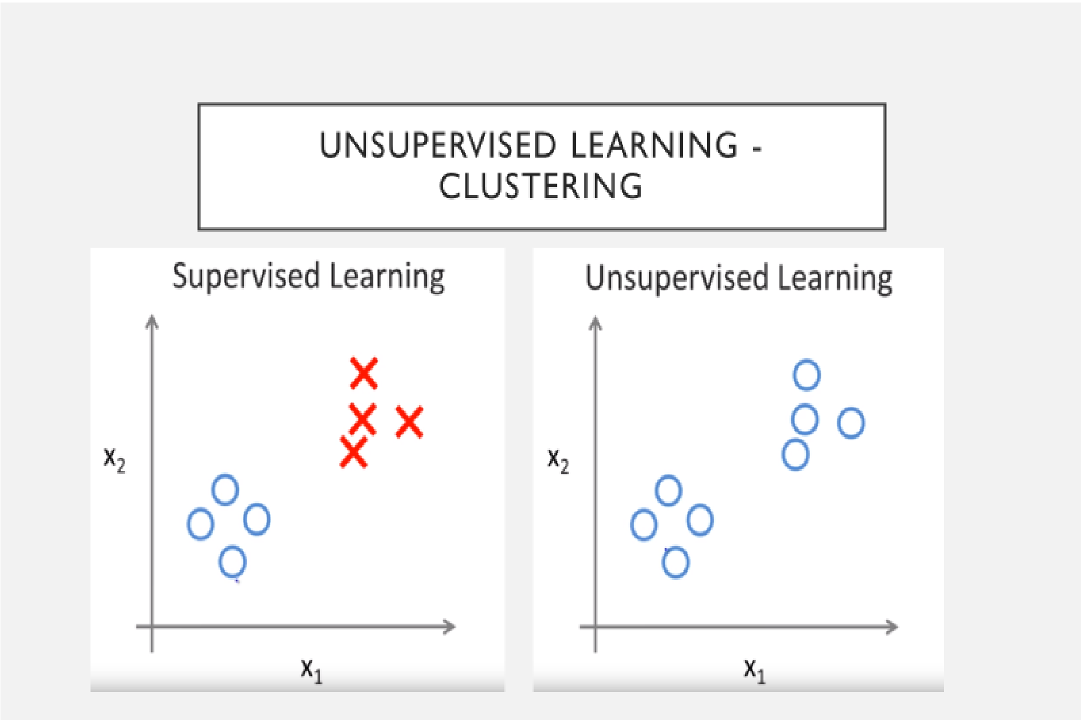


SUPERVISED LEARNING - CLASSIFICATION

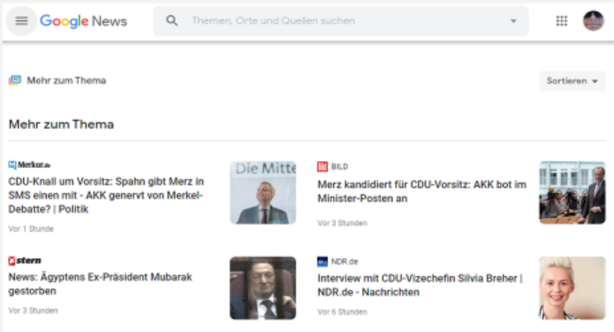



SUPERVISED LEARNING - CLASSIFICATION
Age
Tumor Size


UNSUPERVISED LEARNING
„Unbeschriftete Daten"
Problemklassen
Hauptsächlich Clustering


UNSUPERVISED LEARNING - CLUSTERING
Supervised Learning
Unsupervised Learning
x_2
x_1
x_2
x_1

UNSUPERVISED LEARNING - CLUSTERING



QUIZFRAGEN

- Szenario 1: Wir verkaufen Laptops. Wir haben Verkaufszahlen aus den letzten 5 Jahren und wollen vorhersagen, wie viele Laptops wir in den nächsten 3 Monaten verkaufen werden.
- Szenario 2: Auf Basis einer Kundendatenbank Marktsegmente identifizieren
- Szenario 3: Wir wollen einen Spamfilter erstellen.
- Szenario 4: Nutzergruppen im sozialen Netzwerk analysieren

Supervised oder Unsupervised?
Regression, Klassifikation oder Clustering?

Exercises to Lesson ML2: Concept Learning: Version Spaces & Candidate Elimination

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	*Sky*	*AirTemp*	*Humidity*	*Wind*	*Water*	*Forecast*	*EnjoySport*
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

Solutions:

Homework H2.2– "Version Space – Second example*********"

*********** placeholder*******************

Solutions:

....

Exercises to Lesson ML3: Supervised and Unsupervised Learning

Homework H3.1 - "Calculate *Value Difference Metric*"

Calculate d:= 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

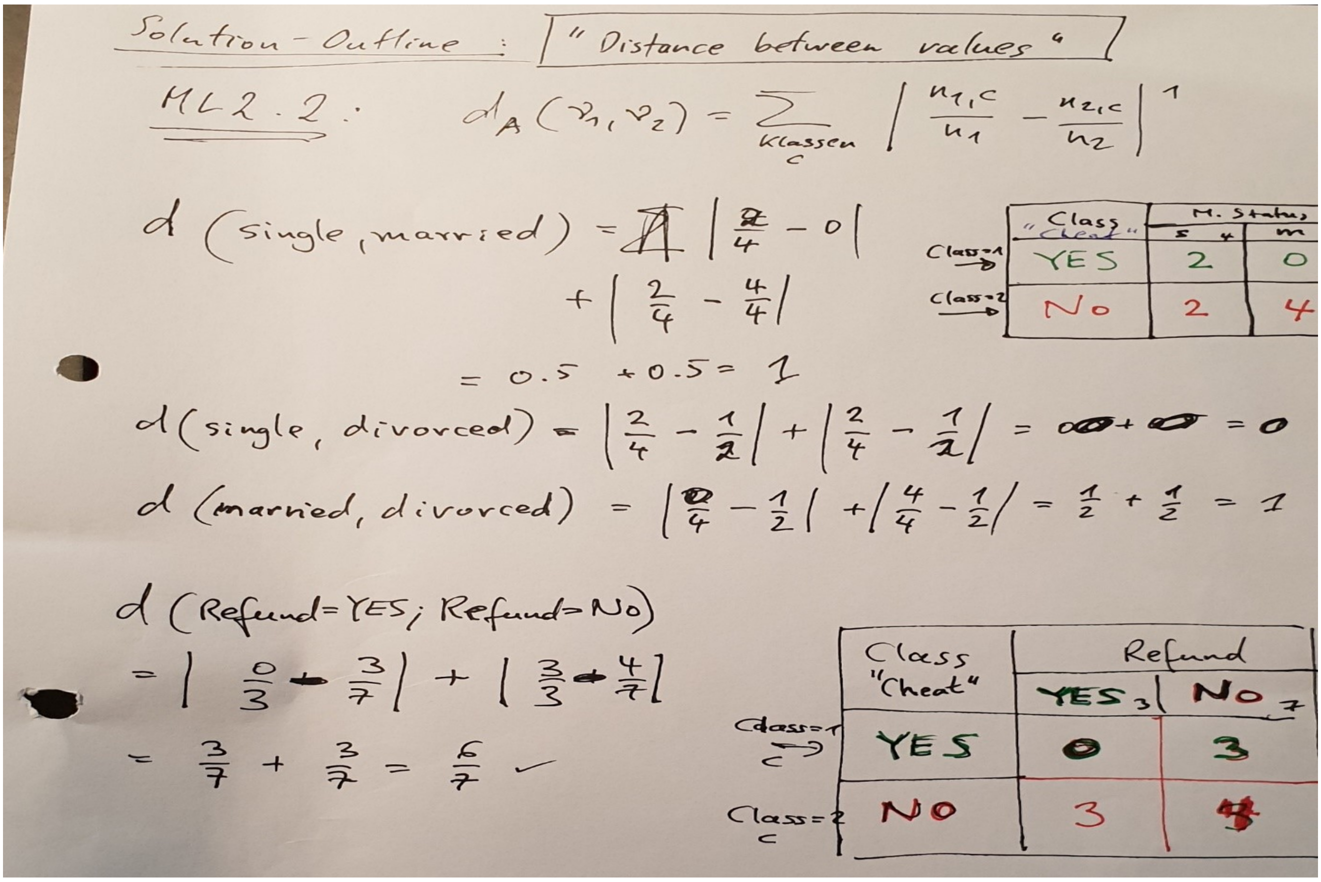
With data table:

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(single, married), d(single, divorced), d(married, divorced); d(refund=yes, refund=no)

Solutions:



Homework H3.2 – "Bayes Learning for Text Classification"

<u>1 Person:</u> 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"***	??????????

<u>Additional Question</u>: What will happen if we change the target to "*Hermann plays a very clean game"*

<u>**Optional***(1 P.):</u> 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.

<u>**Solution**</u>: by A. Gholami, J. Schwarz; ML-Lecture WS2020

Naive Bayes Algorithm

Sentence Classification

What is Bayes Algorithm

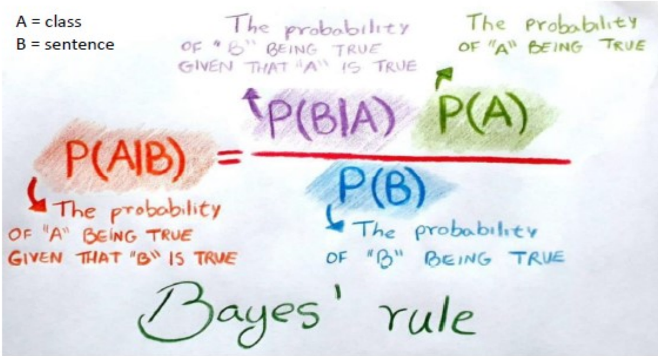
- Simple algorithm to classify text
- Low training time and resources
- Requires a set of labeled training data
- Will be used to classify new sentences

Our data

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

- Training data consists of two classes
  - Sport or not sport

A = class
B = sentence



How does it work?

- Comparing the probabilities:
  - P (Sport| Hermann played a TT match) = $\frac{P(A\ very\ close\ game|Sports) * P(Sports)}{P(A\ very\ close\ game)}$
  - P (Not Sports | Hermann played a TT match) = $\frac{P(A\ very\ close\ game|not\ Sports) * P(Not\ Sports)}{P(A\ very\ close\ game)}$

Probability of a sentence

- Likelyhood a setence is a sport sentence:
  - P(Sport) = $\frac{Number\ of\ sentences\ in\ class\ 'Sports'}{Total\ number\ of\ sentences\ in\ the\ training\ set}$
    - Similarily calculate P(Not Sports)
- „Naive" Bayes because we think each word is indipendent from the other ones
  - Possibility of a sentence „A very close game" is calculated like this:
    - $P(A\ very\ close\ game) = P(A) * P(very) * P(close) * P(game)$

Probability of a sentence in a class

- Applying the probabilites of the words to Bayes formula:

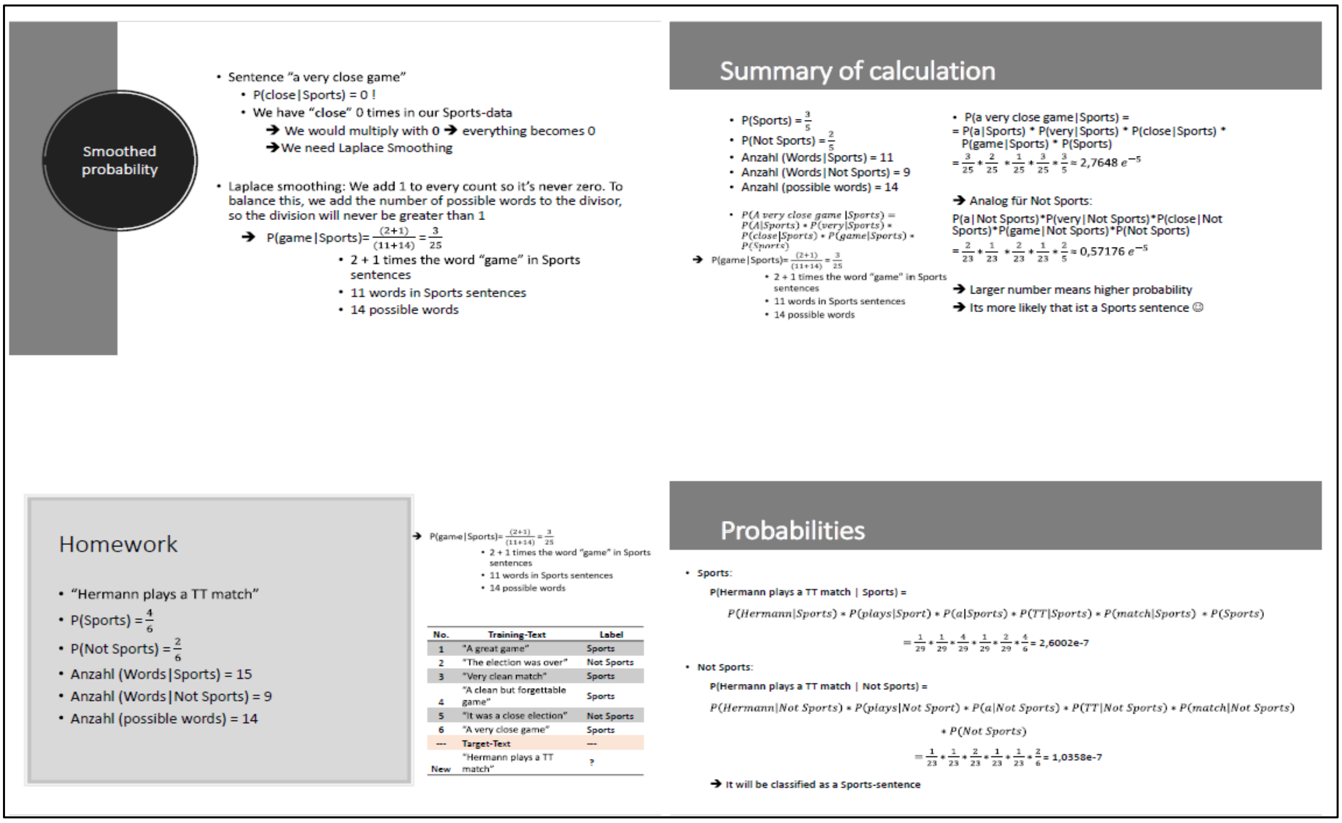
$$P(A\ very\ close\ game\ |Sports) = P(A|Sports) * P(very|Sports) * P(close|Sports) * P(game|Sports) * P(Sports)$$

Calculating the probabilities

- Now all we have to do is calculate all the different probabilites by counting everything in our training data

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

- P(Sports) = 3/5 | P(Not Sports) = 2/5
- Probability of a word in class Sports:
  - $P(game|Sports) = \frac{amount\ of\ 'game'\ in\ Sports\ sentences}{total\ number\ of\ words\ in\ Sports\ sentences} = \frac{2}{11}$
  - Repeat for other words and other classes



Solution to Optional: by A. Gholami, J. Schwarz; ML-Lecture WS2020

1 Naive Bayes Text Classification

We made a simple Algorithm to try and classify sentences into either Sports or Not Sports sentences. We start with a couple sentences either classed "Sports" or "Not Sports" and try to classify new sentences based on that. At the end we make a comparison, which class ("Sports" or "Not Sports") the new sentence is more likely to end up in.

1.1 What happens here:

1. import everything we need
2. Provide training data and do transformations.
3. Create dictionaries and count the words in each class.
4. Calculate probabilities of the words.

To evaluate a new sentence…

5. Vectorize and transform all sentences
6. Count all words
7. Transform new sentence
8. Perform Laplace Smoothing, so we don't multiply with 0
9. Calculate probability of the new sentence for each class
10. Output what's more likely

[1]:
```
# This notebook was created by Alireza Gholami and Jannik Schwarz
# Importing everything we need
```

```
import pandas as pd
from sklearn.feature_extraction.text import CountVectorizer
from nltk.tokenize import word_tokenize
# Import library time to check execution with date + time information
import time
#check versions of libraries
print('pandas version is: {}'.format(pd.__version__))
import sklearn
print('sklearn version is: {}'.format(sklearn.__version__))

[2]: # Naming the columns
columns = ['sentence', 'class']
# Our training data
rows = [['A great game', 'Sports'],
['The election was over', 'Not Sports'],
['Very clean match', 'Sports'],
['A clean but forgettable game', 'Sports'],
['It was a close election', 'Not Sports'],
['A very close game', 'Sports']]

# the data inside a dataframe
training_data = pd.DataFrame(rows, columns=columns)
print('f'The training data:\n{training_data}\n')

[3]: # Turns the data into vectors
def vectorisation(my_class):
# my_docs contains the sentences for a class (sports or not sports)
my_docs = [row['sentence'] for index, row in training_data.iterrows() if row['class'] ==
my_class]
# creates a vector that counts the occurrence of words in a sentence
my_vector = CountVectorizer(token_pattern=r"(?u)¥b¥w+¥b")
# Token-Pattern damit einstellige Wörter wie 'a' gelesen werden
# transform the sentences
my_x = my_vector.fit_transform(my_docs)
# tdm = term_document_matrix_sport | create the matrix with the vectors for a class
tdm = pd.DataFrame(my_x.toarray(), columns=my_vector.get_feature_names())
return tdm, my_vector, my_x

[4]: # Here we are actually creating the matrix for sport and not sport sentences
tdm_sport, vector_sport, X_sport = vectorisation('Sports')
tdm_not_sport, vector_not_sport, X_not_sport = vectorisation('Not Sports')
print (f'Sport sentence matrix: \n{tdm_sport}\n')
```

```
print (f'Not sport sentence matrix: \n{tdm_not_sport}\n')
print (f'Amount of sport sentences: {len(tdm_sport)}')
print (f'Amount of not sport senteces: {len(tdm_not_sport)}')
print (f'Total amount of sentences: {len(rows)}')
```

```
[5]: # creates a dictionary for each class
def make_list(my_vector, my_x):
my_word_list = my_vector.get_feature_names()
my_count_list = my_x.toarray().sum(axis=0)
my_freq = dict(zip(my_word_list, my_count_list))
return my_word_list, my_count_list, my_freq
```

```
[6]: # create lists
# word_list_sport = word list ['a', 'but', 'clean', 'forgettable', 'game', 'great', 'match', 'very']
# count_list_sport = occurence of words [2 1 2 1 2 1 1 1]
# freq_sport = combining the two to create a dictionary
word_list_sport, count_list_sport, freq_sport = make_list(vector_sport, X_sport)
word_list_not_sport, count_list_not_sport, freq_not_sport = make_list(vector_not_sport,
X_not_sport)
print(f'sport dictionary: \n{freq_sport}\n')
print(f'not sport dictionary: \n{freq_not_sport}\n')
```

```
[7]: # calculate the probability of a word in a sentence of a class
def calculate_prob(my_word_list, my_count_list): my_prob = []
for my_word, my_count in zip(my_word_list, my_count_list):
my_prob.append(my_count / len(my_word_list))
prob_dict = dict(zip(my_word_list, my_prob))
return prob_dict
```

```
[8]: # probabilities of the words in a class
prob_sport_dict = calculate_prob(word_list_sport, count_list_sport)
prob_not_sport_dict = calculate_prob(word_list_not_sport, count_list_not_sport)
print(f'probabilites of words in sport sentences: \n{prob_sport_dict}\n')
print(f'probabilites of words in not sport sentences: \n{prob_not_sport_dict}')
```

```
[9]: # all sentences again
docs = [row['sentence'] for index, row in training_data.iterrows()]
# vectorizer
vector = CountVectorizer(token_pattern=r"(?u)¥b¥w+¥b")
```

```
# transform the sentences
X = vector.fit_transform(docs)
# counting the words
total_features = len(vector.get_feature_names())
total_counts_features_sport = count_list_sport.sum(axis=0)
total_counts_features_not_sport = count_list_not_sport.sum(axis=0)
print(f'Amount of distinct words: {total_features}')
print(f'Amount of distinct words in sport sentences: {total_counts_features_sport}')
print(f'Amount of distinct words in not sport sentences:
{total_counts_features_not_sport}')
```

```
[10]: # a new sentence
new_sentence = 'Hermann plays a TT match'
# gets tokenized
new_word_list = word_tokenize(new_sentence)
```

```
[11]: # We're using Laplace smoothing, # if a new word occurs the probability would be 0
# So every word counter gets incremented by one
def laplace(freq, total_count, total_feat): prob_sport_or_not = []
for my_word in new_word_list:
if my_word in freq.keys():
counter = freq[my_word]
else: counter = 0
# total_count is the amount of words in sport sentences and total_feat the total amount of words
prob_sport_or_not.append((counter + 1) / (total_count + total_feat))
return prob_sport_or_not
```

```
[12]: # probability for the new words
prob_new_sport = laplace(freq_sport, total_counts_features_sport, total_features)
prob_new_not_sport = laplace(freq_not_sport, total_counts_features_not_sport,
total_features)
print(f'probability that the word is in a sport sentence: {prob_new_sport}')
print(f'probability that the word is in a not sport sentence: {prob_new_not_sport}')
```

```
[13]: # multiplying the probabilities of each word
new_sport = list(prob_new_sport)
sport_multiply_result = 1
for i in range(0, len(new_sport)): sport_multiply_result *= new_sport[i]

# multiplying the result with the ratio of sports sentences to the total amount of sentences (here: 4/6)
```

```
sport_multiply_result *= ( len(tdm_sport) / len(rows) )

# multiplying the probabilities of each word
new_not_sport = list(prob_new_not_sport)
not_sport_multiply_result = 1
for i in range(0, len(new_not_sport)): not_sport_multiply_result *= new_not_sport[i]
# multiplying the result with the ratio of sports sentences to the total amount of sentences (here: 2/6)
not_sport_multiply_result *= ( len(tdm_not_sport) / len(rows) )
```

```
[14]: # comparing what's more likely
print(f'The probability of the sentence "{new_sentence}":\nSport vs not sport\n
{sport_multiply_result} vs {not_sport_multiply_result}\n\n')
if not_sport_multiply_result < sport_multiply_result: print('Verdict: It\'s probably a sports
sentence!')
else: print('Verdict: It\'s probably not a sport sentence!')
```

```
[15]: # print current date and time
print("Date & Time:",time.strftime("%d.%m.%Y %H:%M:%S"))
print ("*** End of Homework-H3.2_Bayes-Learning... ***")
```

Homework H3.3 (advanced)* – "Create in IBM Cloud two services *Voice Agent* and *Watson Assistant Search Skill* with IBM Watson Services"

Homework for 2 Persons: 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

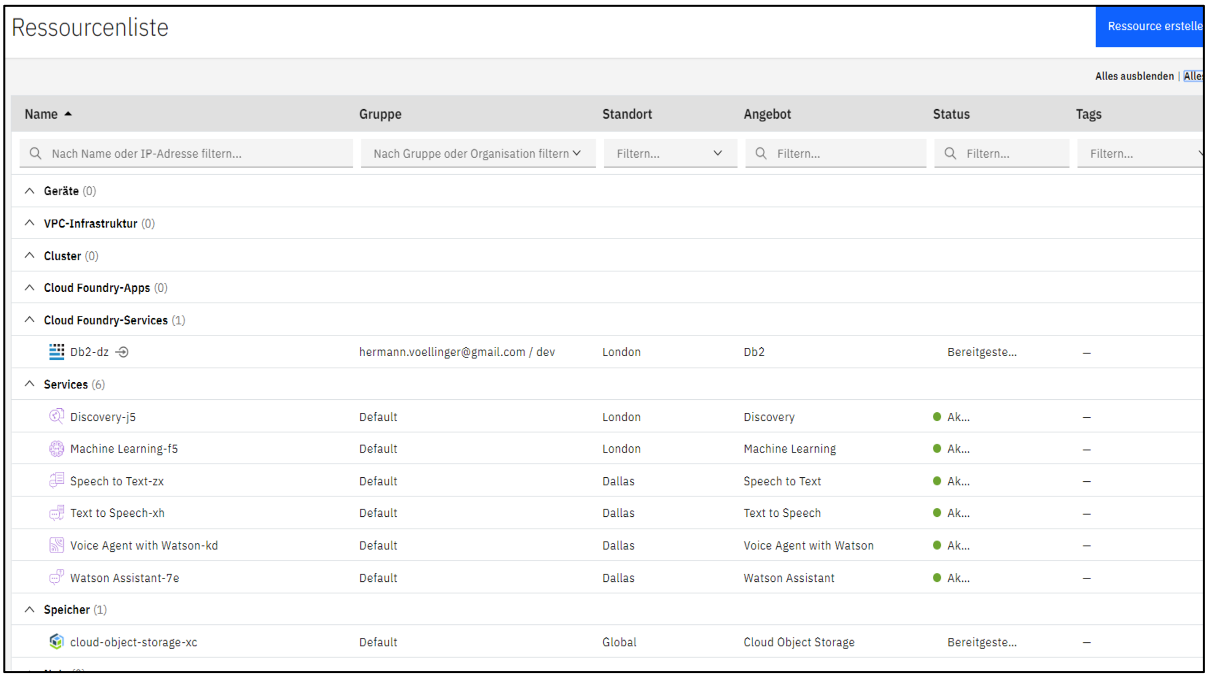
<u>See tutorial:</u>
https://github.com/FelixAugenstein/digital-tech-tutorial-watson- assistant-search-skill

<u>Remark:</u> 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

Solutions:

Ad1: by Hermann Völlinger; 12.3.2020

For creating a “voice agent” I activate the 4 services “Speech2Text”, “Text2Speech”, “Voice Agent” and Watson Assistant” on IBM Watson. See the following screenshot:

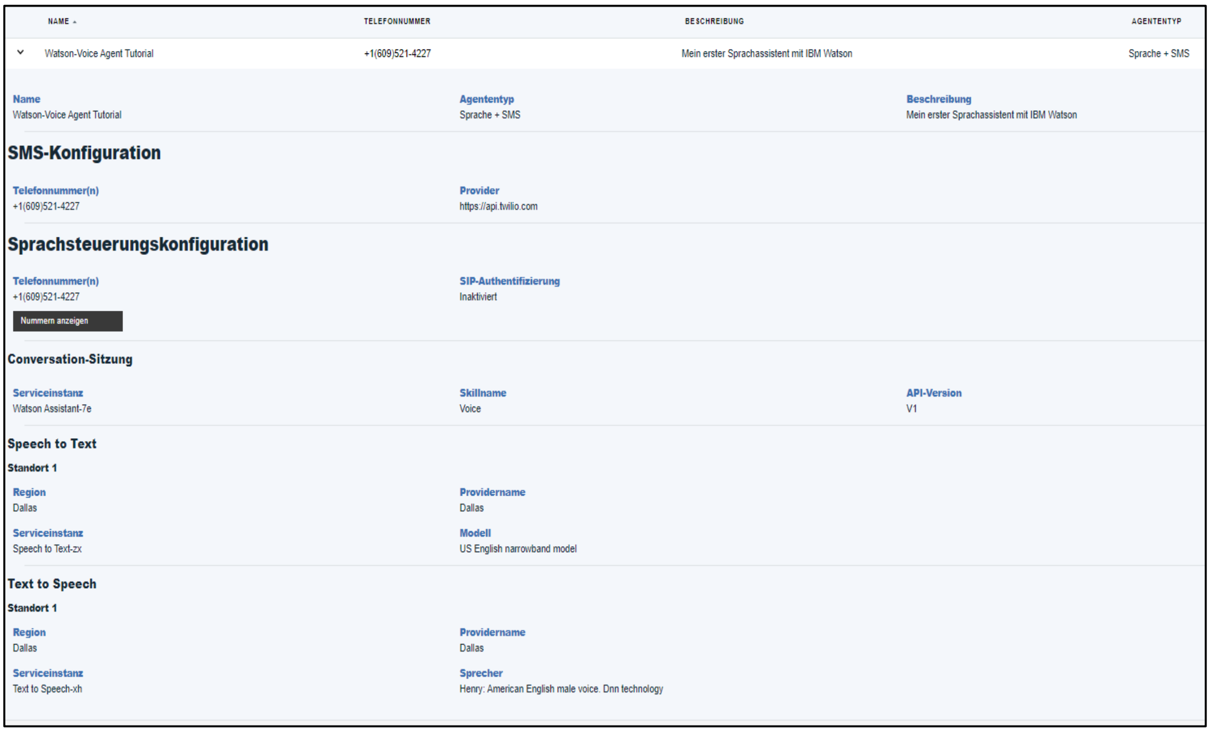


Next to have to do the Configuring of a Twilio Account, including the steps:

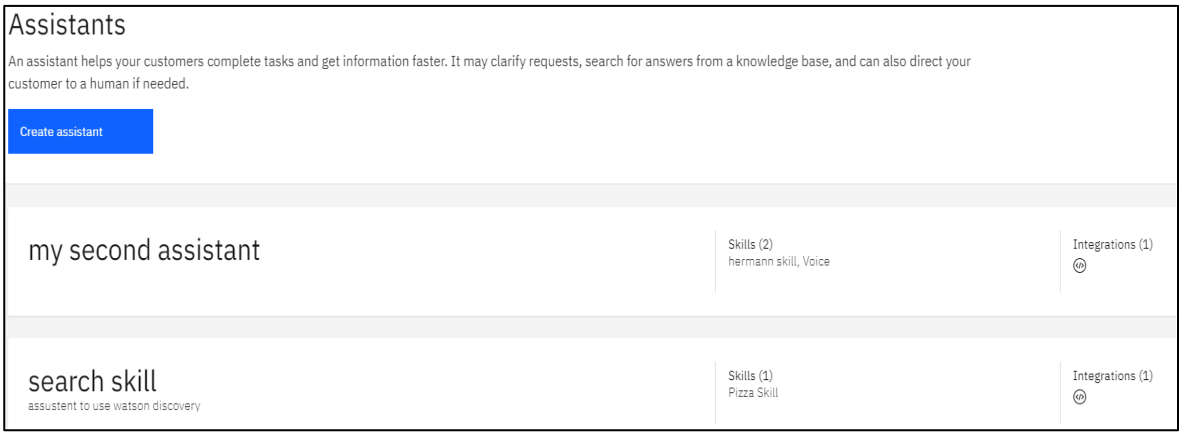
1. Register for Twilio and Start a free Trial.
2. Confirm your email.
3. Verify your phone number. Therefore, use the phone number you will use to call the Watson Voice Agent later on.

You link the phone-number with your solution “Watson-Voice Agent Tutorial”, see:

Finally, you can see the final configuration by opening the service app "Watson-Voice Agent Tutorial". See the following screenshot:



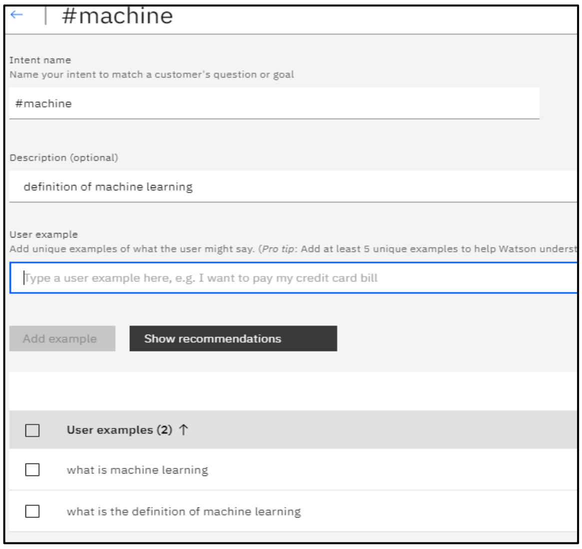
By opening the *Watson Assistant,* we see all available solutions, i.e. dialog- and search skills. Under "my second assistant" we see the two dialog skills "*hermann skill"* and *"voice":*

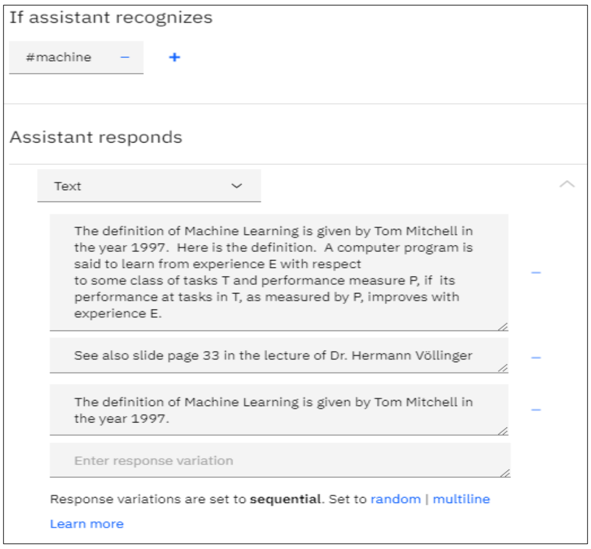


After opening "voice" we see all intents (number=12). Some are imported by the json-file. Other are created by myself, like *#machine*, *#FirstExample* or *#SecondExample:*

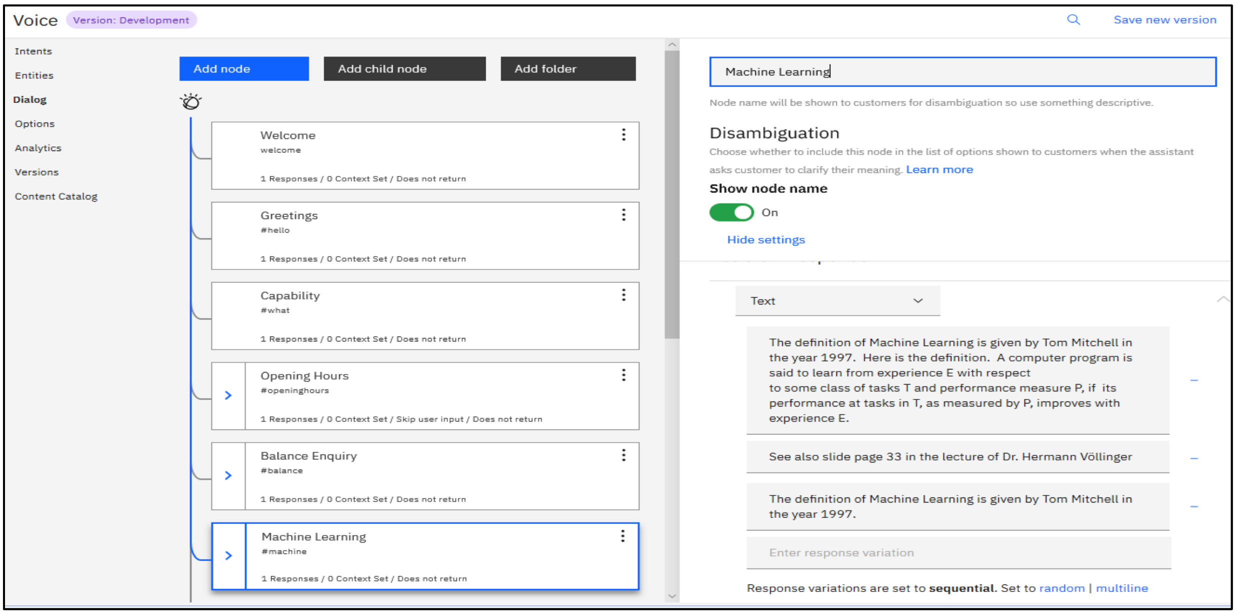
Intents (12) ↑	Description	Modified ⇅	Conflicts ⇅
#balance	Get balance	a month ago	
#FirstExample	First example of ML Definition	a month ago	
#goodbye	Goodbye	a month ago	
#hello	Greetings	2 months ago	
#machine	definition of machine learning	a month ago	
#No	Negative	2 months ago	
#openinghours	What are the opening hours	2 months ago	
#SecondExample	Second examples of ML definition	a month ago	
#TableTennis	support for playing tabel tennis	a month ago	
#time	Ask for Time	2 months ago	
#what	What can you do?	2 months ago	
#Yes	Affirmative	2 months ago	

You can define questions (see *#machine*) and also answers of the voice assistant (“chatbot”):

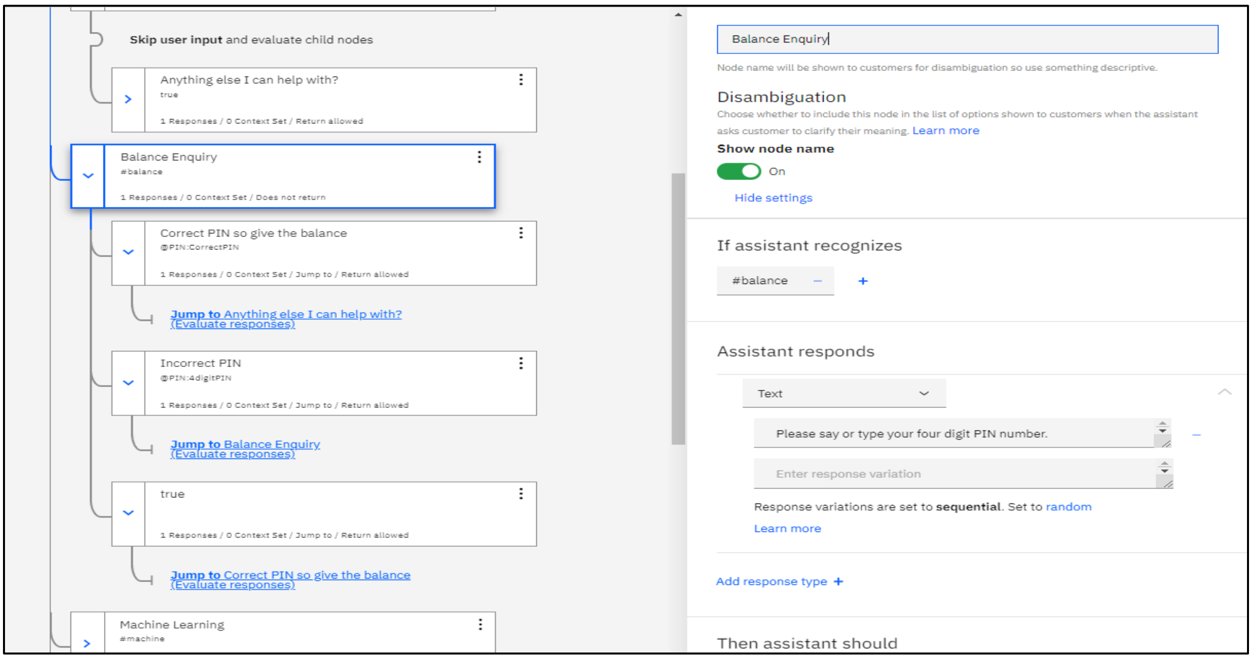




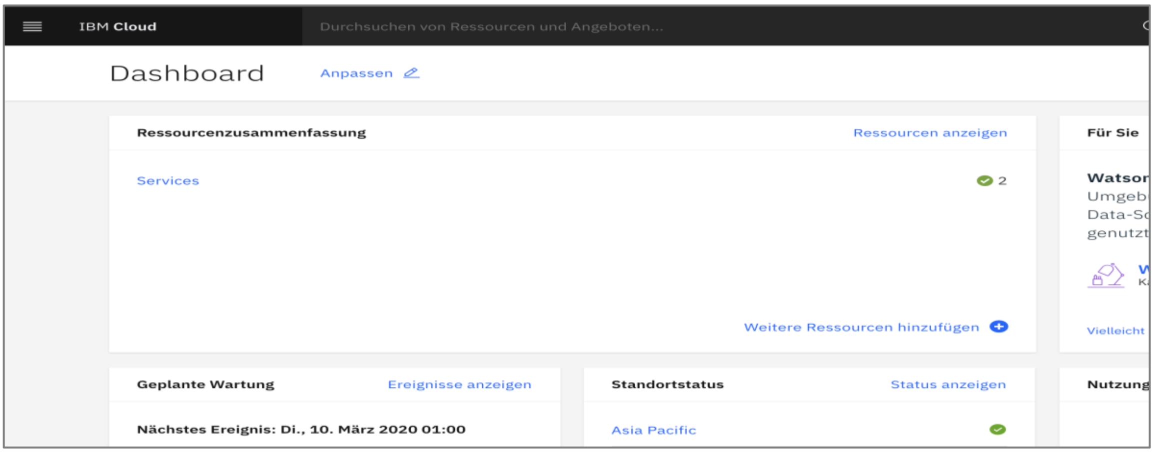
So, one gets the final flow chart of the dialog skill for the Voice-Agent *Voice*. See her the response of the question “*What is Machine Learning*?”:



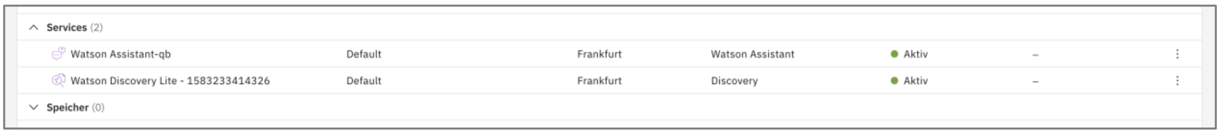
Similar you see her the logic of the question "*What is my Balance*?":

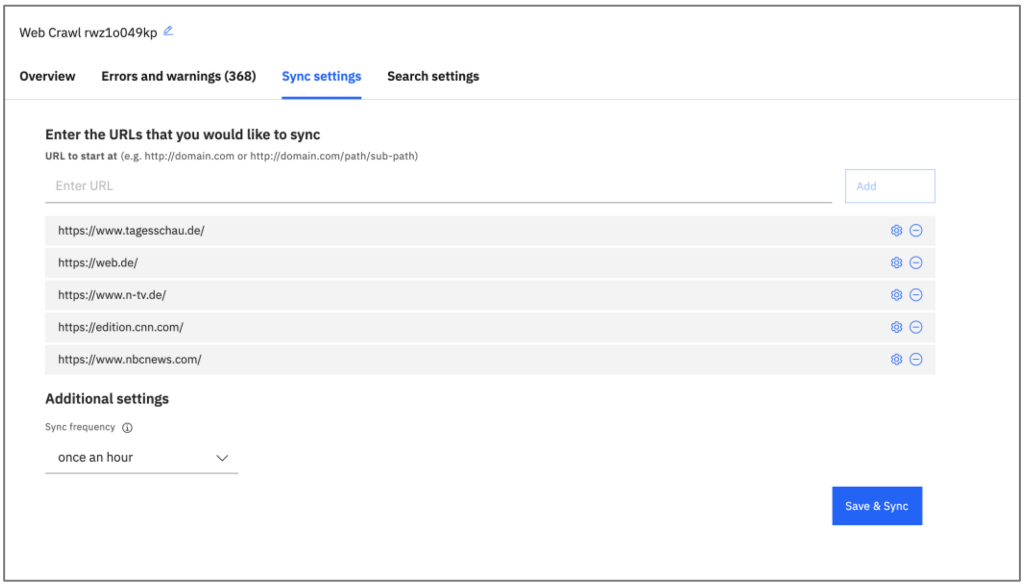


Ad2: By Niklas Gysinn & Maximilian Wegmann, DHBW Stg. SS2020 (4.3.2020)
Creating a Watson Search (Discovery) Skill using the IBM Cloud
Source used: https://github.com/FelixAugenstein/digital-tech-tutorial-watson-assistant-search-skill

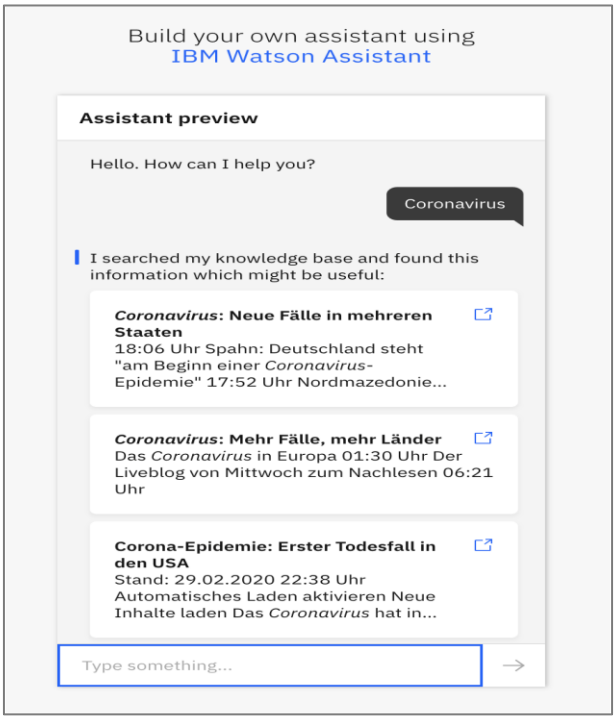


First of all, we created two services. One service for crawling and indexing the website information and one for providing the assistant functionality.





The discovery service uses various news sites (e.g. German “Tagesschau”) to retrieve the latest articles and make them available to the assistant.



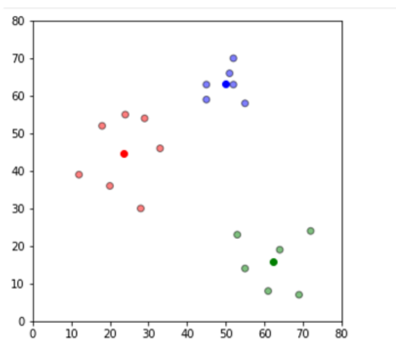
This information can then be accessed via a "chat" provided by the IBM Watson Assistant service.

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

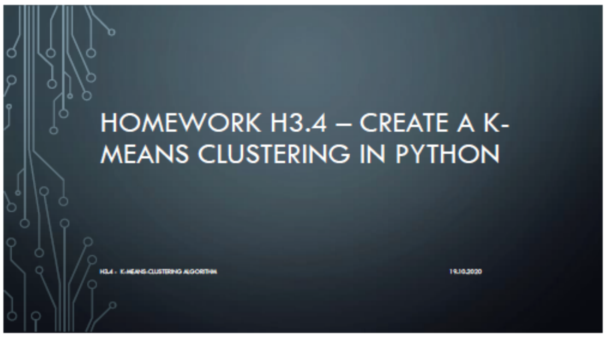
Homework for 2 Persons: Create a python algorithm (in Jupyter Notebook) which clusters the following points:

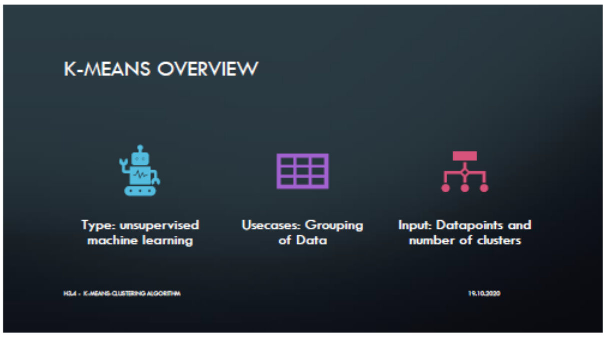
```
df = pd.DataFrame({
    'x': [12, 20, 28, 18, 29, 33, 24, 45, 45, 52, 51, 52, 55, 53, 55, 61, 64, 69, 72],
    'y': [39, 36, 30, 52, 54, 46, 55, 59, 63, 70, 66, 63, 58, 23, 14, 8, 19, 7, 24]
})
```

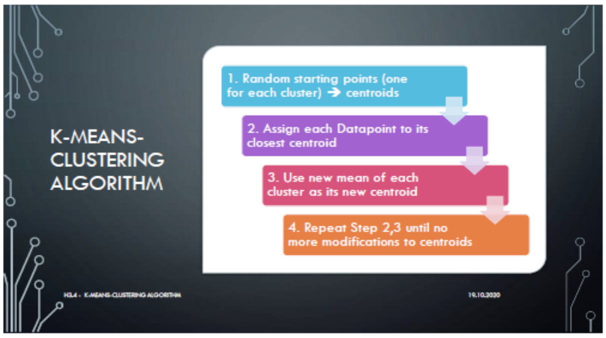
Following the description of: https://benalexkeen.com/k-means-clustering-in-python/ to come to 3 clear clusters with 3 means at the center of these clusters: We'll do this manually first (1 person), then show how it's done using scikit-learn (1 person)

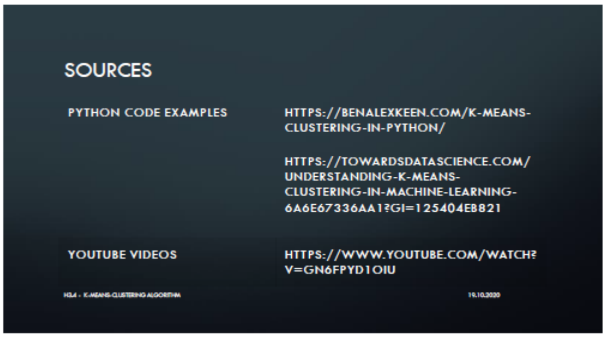


Solutions: by L. Krauter und M. Limbacher; ML Lecture - WS2020









1 Create a K-Means Clustering Algorithm in Python

By: Markus Limbacher & Lucas Krauter; 20. October 2020
This solves Homework H3.4 from Lecture: “Machine Learning - Concepts & Algorithms”, DHBW Stuttgart, WS2020
Following the implementation of Ben Keen (2017) from: “https://benalexkeen.com/k-meansclustering-
in-python/”

1.1 Content

This notebook is split into three parts: 1. Section 1.2 2. Section 1.3: program each step manually 3. Section 1.4: use the scikit library to use the algorithm

1.1.1 Summary K-Means Algorithm:

1. Select Random Starting Points (one for each cluster) = centroids
2. Assign each Datapoint to its closest centroid
3. Use new mean of each cluster as its new centroid
4. Repeat Step 2,3 until mo more modifications to centroids are made

1.2 Preparations

1.2.1 Import of libraries

The first step is to import the necessary library packages.

[1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
```

```
%matplotlib inline
import copy
import sklearn as sk
from sklearn.cluster import KMeans
# to check the time of execution, import function time
import time
# check versions of libraries
print('pandas version is: {}'.format(pd.__version__))
print('numpy version is: {}'.format(np.__version__))
print('sklearn version is: {}'.format(sk.__version__))
```

1.2.2 Dataset
The second step is defining data to work with. The data frame contains two arrays of x and y coordinates. These build several points in a two-dimensional space.

```
[2]: # Definition of Dataset (see Homework H3.4)
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] })
# Check that the definition of dataset is OK
print ("**** data frame ****")
print ("First column = No.")
print (df)
*** data frame ***
First column = No.
    x  y
0 12 39
1 20 36
2 28 30
3 18 52
4 29 54
5 33 46
6 24 55
7 45 59
8 45 63
9 52 70
10 51 66
11 52 63
12 55 58
13 53 23
14 55 14
15 61 8
```

16 64 19

17 69 7

18 72 24

1.3 K-Means manually

Start with selecting the count of clusters **k**. Select one random Starting Point **i** for each cluster. These center points are called **centroids**.

```
[3]: # Number of clusters ==> k
k = 3
np.random.seed(42)
# centroids[i] = [x, y]
centroids = {
i+1: [np.random.randint(0, 80), np.random.randint(0, 80)]
for i in range(k)
}
```

1.3.1 Display dataset

Print the centroids and the values of the data frame in a two-dimensional coordinate system.

```
[4]: fig = plt.figure(figsize=(5, 5))
plt.scatter(df['x'], df['y'], color='k')
colmap = {1: 'r', 2: 'g', 3: 'b'}
for i in centroids.keys():
plt.scatter(*centroids[i], color=colmap[i])
plt.xlim(0, 80)
plt.ylim(0, 80)
plt.show()
```

1.3.2 Assignment Stage

Assign each Datapoint to its closest centroid. Since the step will be repeated, we will program a function. The distance is calculated as the difference between the two points [x1,y1] and [x2,y2] by the following formula: $d=\sqrt{(x1 - x2)^2 - (y1 - y2)^2}$

```
[5]: # Function to determine closest centroid for the dataset df
def assignment(df, centroids):
# Iterating over every centroid in centroids
for i in centroids.keys():
# calculate distance function: sqrt((x1 - x2)^2 - (y1 - y2)^2)
df['distance_from_{}'.format(i)] = (
np.sqrt( (df['x'] - centroids[i][0]) ** 2 + (df['y'] - centroids[i][1]) ** 2) )
```

```
# select and save closest centroid for each datapoint
centroid_distance_cols = ['distance_from_{}'.format(i) for i in centroids.keys()]
df['closest'] = df.loc[:, centroid_distance_cols].idxmin(axis=1)
df['closest'] = df['closest'].map(lambda x: int(x.lstrip('distance_from_')))
# select the color of the cluster depending on the centroid
df['color'] = df['closest'].map(lambda x: colmap[x])
# return data frame with additional information
return df
# call assignment function
df = assignment(df, centroids)
print(df)
x y distance_from_1 distance_from_2 distance_from_3 closest color
0 12 39 46.324939 62.625873 35.902646 3 b
1 20 36 38.013156 56.364883 38.000000 3 b
2 28 30 28.017851 52.430907 44.721360 1 r
3 18 52 50.328918 53.600373 22.090722 3 b
4 29 54 45.650849 42.426407 21.931712 3 b
5 33 46 36.715120 40.496913 30.870698 3 b
6 24 55 49.091751 47.265209 19.416488 3 b
7 45 59 45.398238 26.019224 29.154759 2 g
8 45 63 49.365980 26.172505 27.313001 2 g
9 52 70 56.008928 21.470911 32.249031 2 g
10 51 66 52.000000 20.880613 32.015621 2 g
11 52 63 49.010203 19.235384 33.837849 2 g
12 55 58 44.181444 16.124515 38.483763 2 g
13 53 23 9.219544 41.146081 60.745370 1 r
14 55 14 4.000000 48.703183 69.462220 1 r
15 61 8 11.661904 52.952809 77.698134 1 r
16 64 19 13.928388 41.593269 70.434367 1 r
17 69 7 19.313208 53.037722 83.006024 1 r
18 72 24 23.259407 36.013886 72.138755 1 r
```

1.3.3 Display modified dataset with color assigned to closest centroid.

Create a function to display the new data frame with the additional information. Draw each cluster in a different color.

```
[6]: # Function to display the data frame
def displayDataset(df, centroids):
fig = plt.figure(figsize=(5, 5))
# display data frame
```

```
plt.scatter(df['x'], df['y'], color=df['color'], alpha=0.5, edgecolor='k')
# display each centroid
for i in centroids.keys():
plt.scatter(*centroids[i], color=colmap[i])
plt.xlim(0, 80)
plt.ylim(0, 80)
plt.show()
# invoke display function
displayDataset(df, centroids)
```

1.3.4 Update Stage

Update the position of the centroids of the cluster. For the purpose of tracking the difference between the positions the old positions will be saved in old_centroids. The update function calculates a new mean of each cluster for its new centroid.

```
[7]: # Copies current centroids for demonstration purposes
old_centroids = copy.deepcopy(centroids)
# Calculate mean from each seperate cluster as new centroid positions
def update(k):
# for each centroid
for i in centroids.keys():
# calculate and save new mean
centroids[i][0] = np.mean(df[df['closest'] == i]['x'])
centroids[i][1] = np.mean(df[df['closest'] == i]['y'])
return k
# start update
centroids = update(centroids)
```

1.3.5 Display updated centroids

Display the new positions of the centroids. The change of positions is indicated with arrows.

```
[8]: fig = plt.figure(figsize=(5, 5))
ax = plt.axes()
# draw datapoints
plt.scatter(df['x'], df['y'], color=df['color'], alpha=0.5, edgecolor='k')
# draw centroids
for i in centroids.keys():
plt.scatter(*centroids[i], color=colmap[i])
plt.xlim(0, 80)
plt.ylim(0, 80)
# add arrows
for i in old_centroids.keys():
```

```
old_x = old_centroids[i][0]
old_y = old_centroids[i][1]
dx = (centroids[i][0] - old_centroids[i][0]) * 0.75
dy = (centroids[i][1] - old_centroids[i][1]) * 0.75
ax.arrow(old_x, old_y, dx, dy, head_width=2, head_length=3, fc=colmap[i],ec=colmap[i])
plt.show()
```

1.3.6 Repeat Assignment

Repeat the assignment stage with the new centroid positions.

```
[9]: # assign closest centroid to each point in the dataframe
df = assignment(df, centroids)
# Plot results
displayDataset(df, centroids)
```

1.3.7 Repeat Assignment and Update Steps

Repeat the previous steps until there is no more modification in the assignment of the closest centroids.

```
[10]: # Create endless loop
while True:
# copy old centroid points
closest_centroids = df['closest'].copy(deep=True)
# calculate new means of each cluster
centroids = update(centroids)
# assign each datapoint to nearest centroid
df = assignment(df, centroids)
# if the old centroids equals the new ones => no modification made => exit loop
if closest_centroids.equals(df['closest']):
break

# display result
displayDataset(df, centroids)
```

1.4 K-Means using scikit-learn

Use the scikit k-Means implementation to build the cluster of the data frame.
 ### Preparations
Create the same data frame as above so that it is fresh.

```
[11]: # Dataset
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] })
```

1.4.1 K-Means training

Invoke the imported k-Means constructor with the number of clusters (here 3). Then train the model with the dataset.

```
[12]: # invoke constructor
kmeans = KMeans(n_clusters=3)
# Fitting K-Means model
print(kmeans.fit(df))
```

```
KMeans(algorithm='auto', copy_x=True, init='k-means++', max_iter=300,
n_clusters=3, n_init=10, n_jobs=None, precompute_distances='auto',
random_state=None, tol=0.0001, verbose=0)
```

1.4.2 K-Means prediction

Use the model to calculate a prediction for the same data frame. Each datapoint will be labeled for the chosen cluster.

```
[13]: # create label for each datapoint in data frame
labels = kmeans.predict(df)
# save centroids of each cluster
centroids = kmeans.cluster_centers_
```

1.4.3 Display the result

Display the positions of the centroids and the data frame. The color depends of the assigned label for each datapoint.

```
[14]: # Display result
fig = plt.figure(figsize=(5, 5))
# set color for each datapoint
colmap = {1: 'b', 2: 'g', 3: 'r'}
colors = list( map(lambda x: colmap[x+1], labels))
# draw each datapoint
plt.scatter(df['x'], df['y'],color=colors, alpha=0.5, edgecolor='k')
# draw each centroid
for idx, centroid in enumerate(centroids):
plt.scatter(*centroid, color=colmap[idx+1])
plt.xlim(0, 80)
plt.ylim(0, 80)
plt.show()
```

```
[15]: # print current date and time
print("date & time:",time.strftime("%d.%m.%Y %H:%M:%S"))
print ("*** End of Homework-H3.4_k-Means_Clustering ***")
date & time: 19.10.2020 17:44:45
```

*** End of Homework-H3.4_k-Means_Clustering ***

Homework H3.5 – "Repeat + Calculate *Measures for Association*"

1. Remember and give explanations of the Measures for Association: support, confidence and lift (1 Person, 10 min):
2. Calculate measures for the following 8 item sets of a shopping basket (1 person, 10 min):
   { Milch, Limonade, Bier }; { Milch, Apfelsaft, Bier }; { Milch, Apfelsaft, Orangensaft };{ Milch, Bier, Orangensaft, Apfelsaft };{ Milch, Bier };{ Limonade, Bier, Orangensaft }; { Orangensaft };{ Bier, Apfelsaft }
   a. What is the support of the item set { Bier, Orangensaft }?
   b. What is the confidence of { Bier } ➔ { Milch } ?
   c. Which association rules have support and confidence of at least 50%?

<u>**First Solution**:</u> Dr. Hermann Völlinger DHBW Stuttgart, SS2019

To 2a.:
We have 8 market baskets -→Support(Bier=>Orangensaft)=frq(Bier,Orangensaft)/8
We see two baskets which have Bier and Orangensaft together
--→Support = 2/8=1/4 = 25%
To 2b.:
We see that frq(Bier)=6 und frq(Bier,Milch)=4 -→Conf(Bier=>Milch)=4/6=2/3= 66,7%
To 2c.:
To have a support>=50% we need items/products which occur in more than 4 baskets. We see for example Milch is in 5 baskets (we write: #Milch=5), #Bier=6, #Apfelsaft=4, #Orangensaft=4 and #Limonade=2.
Only the 2-pair #(Milch, Bier)=4 has minimum of 4 occurrences. We see this by calculating the Frequency-Matric(frq(X=>Y)) for all tuples (X,Y):

frq(X,Y)	Bier	Milch	A-Saft	O-Saft	Limo
Bier		4	3	2	2
Milch	4		3	2	1
A-Saft	3	3		2	0
O-Saft	2	2	2		1
Limo	2	1	0	1	

It is easy to see, that there are no 3-pairs with a minimum of 4 occurrences: only Sup(Bier,Milch) is >=50%. But for all X: Sup{Bier,Milch},X)<50% .
We see from the above matric, that: Supp(Milch=>Bier)=Supp(Bier=>Milch)4/8=1/2=50%
We now calculate: Conf(Milch=>Bier)=4/#Milch=4/5=80%
From Question 2, we know that Conf(Bier=>Milch)=66,7%

Solution: Only the two association rules (Bier=>Milch) and (Milch=>Bier) have support and confidence >=50%.

<u>Second Solution</u>: Anna-Lena Volkhardt, DHBW Stuttgart, SS2020 (4.3.2020)

Definition

Support:
It is a measure of how frequently the collection of items occur together as a percentage of all transactions.

Confidence:
It is the ratio of the number of transactions that include all items in {Y} as well as the number of transactions that include all items in {X} to the number of transactions that include all items in {X}.

Lift:
It is the ratio of confidence to expected confidence. The Lift tells us how much better a rule is at predicting the result than just assuming the result in the first place. Greater lift values indicate stronger associations.

Source: https://infocenter.informationbuilders.com/wf80/index.jsp?topic=%2Fpubdocs%2FRStat16%2Fsource%2Ftopic49.htm

Rule

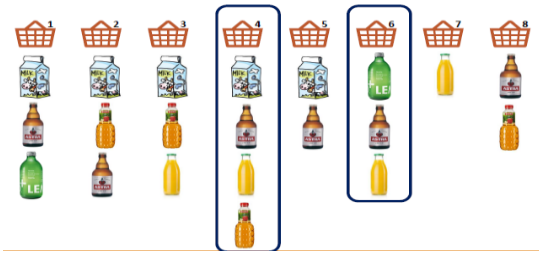
X => Y

Support(X,Y) = **Frq(X,Y)/N**

Confidence(X,Y) = **Frq(X,Y)/Frq(X)**

Lift(X,Y) = **Con(X,Y)/Sup(Y)**

1	Milch	Bier	Limonade	
2	Milch	Apfelsaft	Bier	
3	Milch	Apfelsaft	Orangensaft	
4	Milch	Bier	Orangensaft	Apfelsaft
5	Milch	Bier		
6	Limonade	Bier	Orangensaft	
7	Orangensaft			
8	Bier	Apfelsaft		

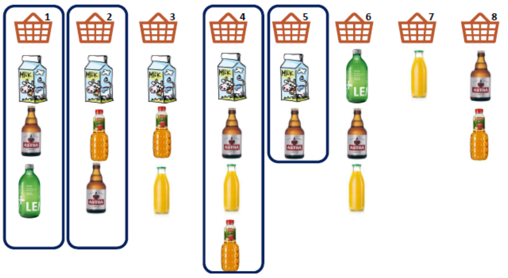


Calculation

Support(Bier, Orangensaft) = Frq(Bier,Orangensaft)/8

Frq(Bier,Orangensaft) = 2

=> Support = 2/8 = ¼ = 25%



Calculation

Confidence(Bier, Milch) = Frq(Bier,Milch)/Frq(Bier)

Frq(Bier,Milch) = 4

Frq(Bier) = 6

=> Confidence = 4/6 = 2/3 = 67%

Overview

1	Milch	Bier	Limonade	
2	Milch	Apfelsaft	Bier	
3	Milch	Apfelsaft	Orangensaft	
4	Milch	Bier	Orangensaft	Apfelsaft
5	Milch	Bier		
6	Limonade	Bier	Orangensaft	
7	Orangensaft			
8	Bier	Apfelsaft		

X	Frq(X)
Milch	5
Bier	6
Limonade	2
Orangensaft	4
Apfelsaft	4

Frequency-Matric

frq(X,Y)	Milch	Bier	Limonade	Orangensaft	Apfelsaft
Milch	x	**4**	1	2	3
Bier	**4**	x	2	2	3
Limonade	1	2	x	1	0
Orangensaft	2	2	1	x	2
Apfelsaft	3	3	0	2	x

Calculation

For support >= 50% we need Frq(X,Y) >= 4. As we can see in the freqency-matric it only appears twice.

Only the pair (Milch,Bier) has 4 occurences and a support of 50%.

For the confidence you can use the result of task 2.2 for Conf(Bier,Milch) = 67% and Conf(Milch,Bier) = 4/5 = 80%.

Thanks to the frequency-matric you can see, that there are no 3-pairs with a minimum of 4 occurrences.

Only the two association rules (Bier=>Milch) and (Milch=>Bier) have support and confidence >=50%.

<u>Third Solution</u>: R. Beer & A. Joukhadar, DHBW Stuttgart, WS2020 (20.10.2020)

REPEAT + CALCULATE OF MEASURES FOR ASSOCIATION

Homework H3.5
Robin Beer – Abdulkarim Joukhadar

Measures for Association

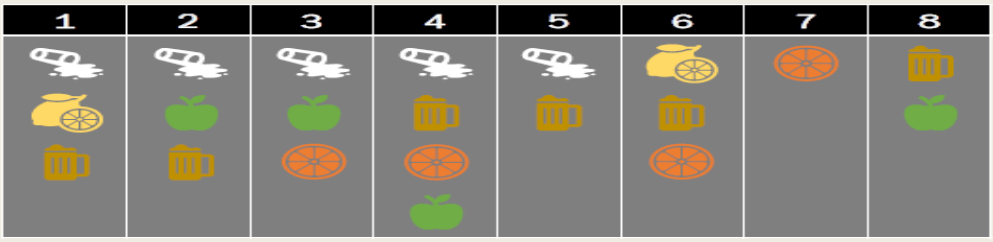
- Support — Percentage of how often an association appears in the whole dataset
- Confidence — How often the rule is found to be true
- Lift — Ratio of how often the association occurs compared to if the values were independent

Association Rules $X \Rightarrow Y$

- Support $Support = \frac{frq\ (X,Y)}{N}$
- Confidence $Confidence = \frac{frq\ (X,Y)}{frq\ (X)}$
- Lift $Lift = \frac{Support}{Support(X) \times Support(Y)}$

15.10.2020 3

Item sets of a shopping basket



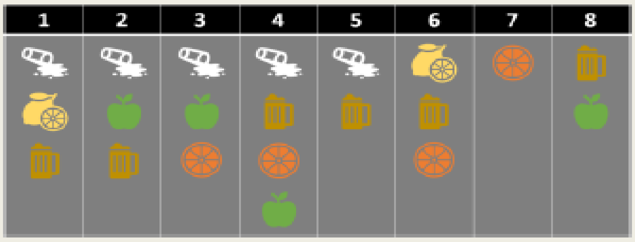
15.10.2020 4

1. What is the support of the item set {Bier, Orangensaft}?

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

$$Support = \frac{frq\ (Bier, Orangensaft)}{N}$$

$$Support = \frac{2}{8} = \frac{1}{4} = 25\%$$



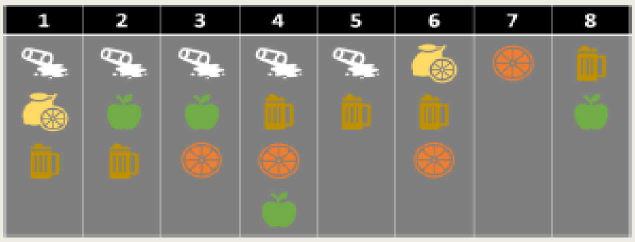
15.10.2020 5

2. What is the confidence of {Bier}➔{Milch}?

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

$$Confidence = \frac{frq\ (Bier, Milch)}{frq\ (Bier)}$$

$$Confidence = \frac{4}{6} = \frac{2}{3} = 66{,}67\%$$



3. Which association rules have support and confidence of at least 50%?

Frequencies:

- Milch: 5
- Limonade: 2
- Bier: 6
- Apfelsaft: 4
- Orangensaft: 4

1	2	3	4	5	6	7	8
Milch	Milch	Milch	Milch	Milch	Limonade	Orangensaft	Bier
Limonade	Apfelsaft	Apfelsaft	Bier	Bier	Bier		Apfelsaft
Bier	Bier	Orangensaft	Orangensaft		Orangensaft		
			Apfelsaft				

15.10.2020 7

3. Which association rules have support and confidence of at least 50%?

X/Y	Milch	Limonade	Bier	Apfelsaft	Orangensaft
Milch	X	s = 1/8 c = 1/5	s = 1/2 c = 4/5	s = 3/8 c = 5/8	s = 1/4 c = 2/5
Limonade	s = 1/8 c = 1/2	X	s = 1/4 c = 1	0	s = 1/8 c = 1/2
Bier	s = 1/2 c = 2/3	s = 1/4 c = 1/3	X	s = 3/8 c = 1/2	s = 1/4 c = 1/3
Apfelsaft	s = 3/8 c = 3/4	0	s = 3/8 c = 3/4	X	s = 1/4 c = 1/2
Orangensaft	s = 1/4 c = 1/2	s = 1/8 c = 1/4	s = 1/4 c = 1/2	s = 1/4 c = 1/2	X

X/Y	Milch	Limonade	Bier	Apfelsaft	Orangensaft
Milch, Limonade	X	X	s = 1/8 c = 1	0	0
Milch, Bier	X	s = 1/8 c = 1/4	X	s = 1/4 c = 1/2	s = 1/8 c = 1/4
Milch, Apfelsaft	X	0	s = 1/4 c = 2/3	X	s = 1/4 c = 2/3
Milch, Orangensaft	X	0	s = 1/8 c = 1/2	s = 1/4 c = 1	X
Limonade, Bier	s = 1/8 c = 1/2	X	X	0	s = 1/8 c = 1/2
Limonade, Apfelsaft	-	X	-	X	-
Limonade, Orangensaft	0	X	s = 1/8 c = 1	0	X
Bier, Apfelsaft	s = 1/4 c = 2/3	0	X	X	s = 1/8 c = 1/3
Bier, Orangensaft	s = 1/8 c = 1/2	s = 1/8 c = 1/2	X	s = 1/8 c = 1/2	X
Apfelsaft, Orangensaft	s = 1/4 c = 1	0	s = 1/8 c = 1/2	X	X

15.10.2020 8

Exercises to Lesson ML4: Decision Tree Learning

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

Groupwork (2 Persons). Calculate the measures of the 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.

First Solution with ID3 (Hermann Völlinger, Feb. 2020): Missing calculations on **ID3 method** (see page number of the corresponding lecture slides on the right top):

Outlook:

Page 59

$$E(outlook=sunny) = -\frac{2}{5}\cdot\log_2\left(\frac{2}{5}\right) - \frac{3}{5}\cdot\log_2\left(\frac{3}{5}\right)$$

$$= -\frac{2}{5}\cdot(-1{,}32) - 0.6\cdot\underbrace{\log_2(0{,}6)}_{-0{,}734}$$

$$= +0{,}528 + 0{,}4421 = +0{,}971$$

$$E(outlook=overcast) = \frac{4}{4}\cdot\log_2(1) - 0\cdot\log_2(0)$$

$$= 1\cdot 0 - 0 = 0.$$

$$E(outlook=rainy) = \frac{3}{5}\cdot\log_2\left(\frac{3}{5}\right) - \frac{2}{5}\log_2\left(\frac{2}{5}\right) = +0{,}971$$

$$\boxed{\sum_{t\in T} p(t)\cdot H(t)} = \sum_{\substack{sunny\\overcast\\rainy}} p(t)\cdot H(t) = \frac{5}{14} * \underbrace{(+0.971)}_{E(outlook=sunny)} + \frac{4}{14}\cdot 0 + \frac{5}{14} * (+0.971)$$

$$= +\frac{10}{14} * 0.971 = +\frac{9{,}71}{14} \boxed{= +0{,}693}$$

$$\boxed{IG(A,S) =} H(S) - \sum_{t\in T} p(t)H(t) = 0.94 - 0{,}693 \boxed{= 0.247}$$

WINDY:

$$E(windy=false) = -\left(\frac{6}{8}\right)\log_2\left(\frac{6}{8}\right) - \frac{2}{8}\cdot\log_2\left(\frac{2}{8}\right) = 0.811$$

$$E(windy=true) = -\frac{3}{6}\log_2\left(\frac{3}{6}\right) - \frac{3}{6}\log_2\left(\frac{3}{6}\right)$$

$$= -\frac{1}{2}\cdot\log_2\left(\frac{1}{2}\right) - \frac{1}{2}\log_2\left(\frac{1}{2}\right) = -\log_2(0.5)$$

$$= -(-1) = 1$$

$$\log_a(b) = x$$
$$\Leftrightarrow b = a^x$$
$$0.5 = 2^{-1} = \frac{1}{2} \checkmark$$

$$\boxed{\sum_{t\in T} p(t)E(\ldots)} = \frac{8}{14}\cdot 0.811 + \frac{6}{14}\cdot 1 = 0{,}463 + 0{,}429$$

$$\boxed{= 0892}$$

$$\boxed{IGain(Windy) = E(S) - \sum_{t\in T} p(t)E(\ldots) = 0.94 - 0.892 = 0{,}048}$$

Humidity

page 60

$$E(Hum = high) = -\frac{3}{7}\log_2\left(\frac{3}{7}\right) - \frac{4}{7}\cdot\log_2\left(\frac{4}{7}\right)$$
$$= +\frac{3}{7}(1{,}222) + \frac{4}{7}(0{,}807)$$
$$= 0.524 + 0{,}461$$
$$\cancel{0.824} = 0.985$$

Humidity

high: 7 — YES, YES, YES (3); NO, NO, NO, NO (4)

normal: 7 — YES, YES, YES, YES, YES, YES (6); NO (1)

$$E(Hum = normal) = -\frac{6}{7}\cdot\log_2\left(\frac{6}{7}\right) - \frac{1}{7}\log_2\left(\frac{1}{7}\right)$$
$$= +\frac{6}{7}\cdot(0{,}222) + \frac{1}{7}\cdot(+2{,}807) = 0{,}190 + 0{,}401$$
$$= 0.591$$

$$\boxed{\sum_{t\in T} p(t)\cdot H(t)} = \frac{7}{14}\cdot 0.985 + \frac{7}{14}(0.591) =$$
$$= \frac{1}{2}(0.985 + 0.591) = \frac{1}{2}(1{,}576) = \boxed{0.788}$$

$$\boxed{IG = 0.940 - 0.788 = 0.152}$$

Temperature: Abzählen: #hot = 4 (YES 2, NO 2); #mild = 6 (Y 4, N 2); #cold = 4 (Y 3, N 1)

$$E(Temp = hot) = -\frac{2}{4}\cdot\log_2\left(\frac{2}{4}\right) - \frac{2}{4}\cdot\log_2\left(\frac{2}{4}\right) = -\frac{2}{2}\log_2\left(\frac{1}{2}\right) = 1$$
$$E(Temp = mild) = -\frac{4}{6}\log_2\left(\frac{4}{6}\right) - \frac{2}{6}\cdot\log_2\left(\frac{2}{6}\right) = -\frac{2}{3}\log_2\left(\frac{2}{3}\right) - \frac{1}{3}\log_2\left(\frac{1}{3}\right) =$$
$$= +0{,}390 + 0{,}528 = 0{,}918$$
$$E(Temp = cold) = -\frac{3}{4}\cdot\log_2\left(\frac{3}{4}\right) - \frac{1}{4}\log_2(0.25) = -0{,}75\cdot\log_2(0.75) - 0.25\cdot\log_2(0.25)$$
$$= 0.311 + 0.5 = 0{,}811$$

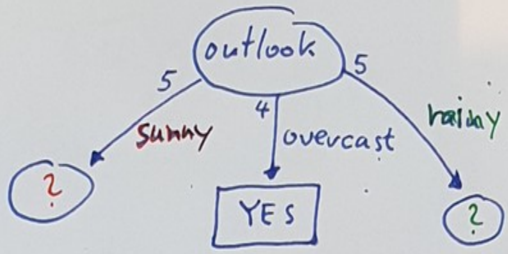
$$\sum_{t\in T} p(t)\cdot H(t) = \frac{4}{14}\cdot 1 + \frac{6}{14}\cdot 0{,}918 + \frac{4}{14}\cdot 0{,}811 = \frac{2}{7} + \frac{3}{7}\cdot 0.918 + \frac{2}{7}\cdot 0.811$$
$$= \boxed{0.911}$$

$$\boxed{IG = 0.940 - 0.911 = 0.029}$$

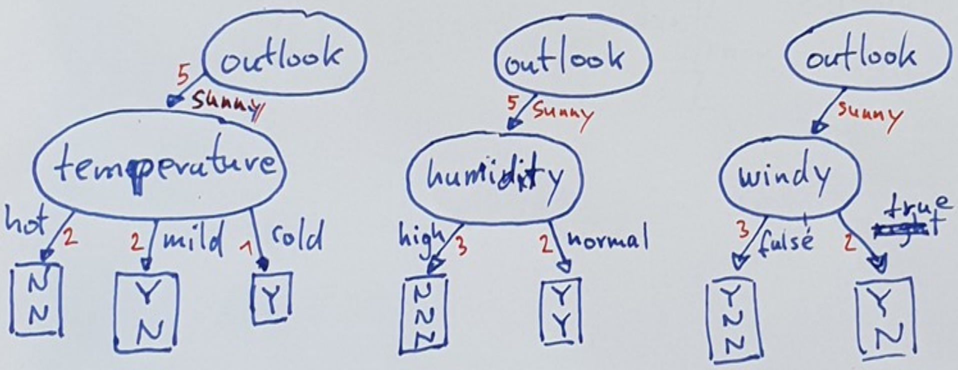
page 61

Model-Calculation of Decision-Tree for "Playing Tennis" (ID3)

After we calculated, that the root-node = "Outlook", we have to calculate the next two nodes ? and ?, with respect for the 3 remaining features/attributes "temperature", "humidity" and "Windy":



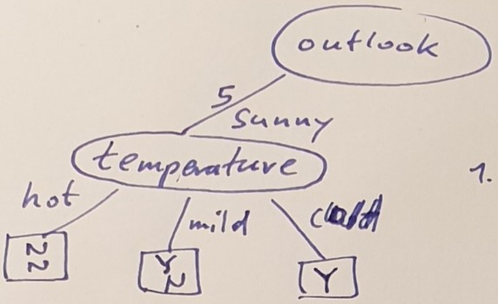
Calculations of ? for all features:



$\boxed{IG(temp)\big|_{Sunny} = 0.571}$ $\boxed{IG(hum)\big|_{Sunny} = 0.971}$ $\boxed{IG(win)\big|_{Sunny} = 0.020}$

⇑ we choose "Humidity" because IG is highest

Page 61



1. Berechne IGain(Temperature)

$E(temp=hot) = -0 \cdot \log_2(0) - \frac{2}{2} \cdot \log_2\left(\frac{2}{2}\right) = -\log_2(1) = 0$

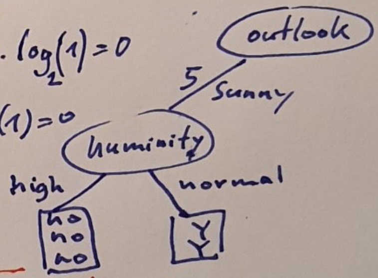
$E(temp=mild) = -\frac{1}{2} \cdot \log_2\left(\frac{1}{2}\right) - \frac{1}{2}\log_2\left(\frac{1}{2}\right) = -\log_2\left(\frac{1}{2}\right) = 1$

$E(temp=cold) = 1 \cdot \log_2(1) = 0$

$\boxed{\sum_{t \in T} p(t) \cdot E(t) = \frac{2}{5} \cdot 0 + \frac{2}{5}(1) + \frac{1}{5} \cdot 0 = \frac{2}{5} = 0.4}$

$\boxed{IGain(Temp.) = E(outlook=sunny) = 0.971 - 0.4 = 0.571}$

2. Berechne IGain(huminity):



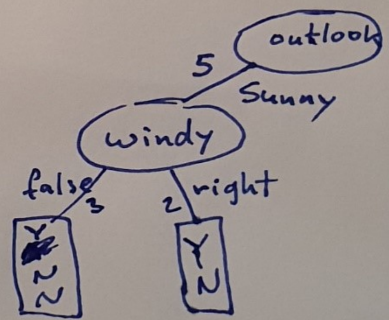
$E(huminity=high) = 0 \cdot \log_2(0) - \frac{3}{3} \cdot \log_2(1) = 0$

$E(hum=normal) = -\frac{2}{2} \cdot \log_2\left(\frac{2}{2}\right) = -\log_2(1) = 0$

$\boxed{\sum_{t \in T} p(t) \cdot E(t) = \frac{3}{5} \cdot 0 + \frac{2}{5} \cdot 0 = 0}$

$\boxed{IGain(Huminity) = 0.971 - 0 = 0.971}$

3. Berechne IGain(windy):



$E(windy=false) = -\frac{1}{3} \cdot \log_2\left(\frac{1}{3}\right) - \frac{2}{3} \cdot \log_2\left(\frac{2}{3}\right)$

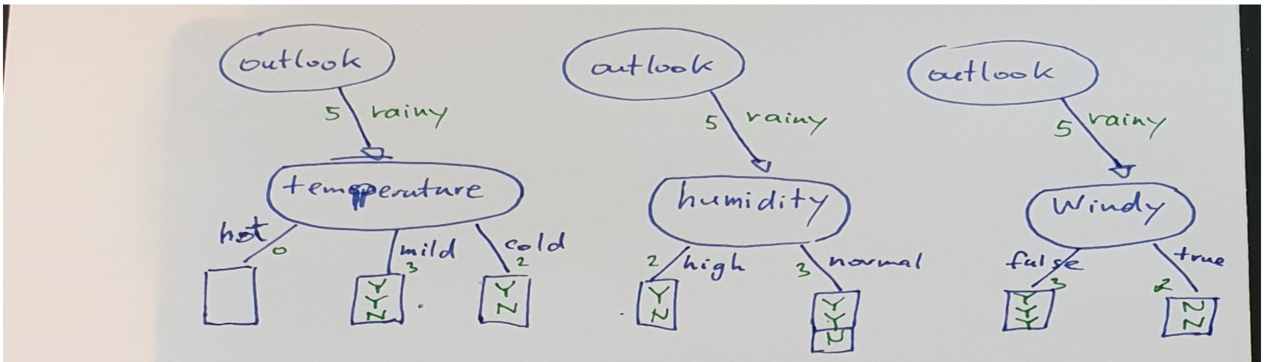
$= 0{,}528 + 0{,}390 = 0{,}918$

$E(windy=right) = -\frac{1}{2} \cdot \log_2\left(\frac{1}{2}\right) - \frac{1}{2}\log_2\left(\frac{1}{2}\right)$

$= -\log_2\left(\frac{1}{2}\right) = 1$

$\boxed{\sum_{t \in T} p(t) \cdot E(t) = \frac{3}{5} \cdot 0.918 + \frac{2}{5} \cdot 1 = 0{,}951}$

$\boxed{IGain(Windy) = 0{,}971 - 0{,}951 = 0{,}020}$



One calculate the IGain for the 3 features "temperature," "humidity" and "windy":

$IGain(temp)|_{rainy} = \boxed{0,020}$ $\quad IGain(hum)|_{rainy} = \boxed{0.020}$ $\quad IGain(windy)|_{r.} = \boxed{0,971}$

select feature = "windy"

page 62

1. Berechne $IGain(temp)|_{rainy}$:

$$E(temp=hot) = \sum_{\substack{yes\\no}} -p(c)\log_2(c) = -p(yes)\cdot\log_2 p(yes) - p(no)\cdot\log_2 p(no)$$

$$= 0\cdot\log_2(p(yes)) - 0\cdot\log_2(p(no)) = 0$$

$$E(temp=mild) = -p(yes)\cdot\log_2(p(yes)) - p(no)\cdot\log_2(p(no))$$

$$= -\frac{2}{3}\cdot\log_2\left(\frac{2}{3}\right) - \frac{1}{3}\cdot\log_2\left(\frac{1}{3}\right) = \frac{2}{3}\cdot(0{,}585) + \frac{1}{3}(1{,}585)$$

$$= 0{,}390 + 0{,}528 = 0{,}918$$

$$E(temp=cold) = -\frac{1}{2}\cdot\log_2\left(\frac{1}{2}\right) - \frac{1}{2}\log_2\left(\frac{1}{2}\right) = -\log_2\left(\frac{1}{2}\right) = 1$$

$$\sum_{t\in\{mild,\ cold\}} p(t)\cdot E(t) = \frac{3}{5}\cdot 0{,}918 + \frac{2}{5}(1) = 0{,}551 + 0{,}4 = \boxed{0{,}951}$$

$$\boxed{IGain(temp)|_{rainy}} = 0.971 - 0{,}951 = \boxed{0{,}020}$$

2. Berechne $IGain(hum)$:

$$E(hum=high) = -\frac{1}{2}\cdot\log_2\left(\frac{1}{2}\right) - \frac{1}{2}\cdot\log_2\left(\frac{1}{2}\right) = -\log_2\left(\frac{1}{2}\right) = 1$$

$$E(hum=normal) = -\frac{2}{3}\cdot\log_2\left(\frac{2}{3}\right) - \frac{1}{3}\left(\log_2\left(\frac{1}{3}\right)\right) = \frac{2}{3}(0{,}585) + \frac{1}{3}(1{,}585) = 0{,}918$$

$$\sum_{t\in T} p(t)E(t) = \frac{2}{5}\cdot 1 + \frac{3}{5}\cdot 0{,}918 = 0{,}4 + 0{,}551 = \boxed{0{,}951}$$

$$IGain(humidity)|_{rainy} = 0.971 - 0{,}951 = \boxed{0{,}020}$$

3. Berechne $IGain(Windy)$:

$$E(windy=false) = -1\cdot\log_2(1) + 0 = 0$$

$$E(windy=right) = 0\cdot\log_2(0) + 1(\log_2(1)) = 0$$

$$\sum_{t\in T} p(t)E(t) = \frac{3}{5}\cdot 0 + \frac{2}{5}\cdot 0 = \boxed{0}$$

$$IGain(windy)|_{rainy} = 0.971 - 0 = \boxed{0{,}971}$$

Second Solution with ID3 (Lars Gerne & Nils Hauschel, 03/31/20):

1 Entropy

1.1 Definition

Entropy indicates the impurity of data. If the value is lower, the data is easier to classify. If the value is higher, the data is more difficult to classify. A high entropy means, that more bits are required to describe the information.

1.2 Formula

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

H - greek E (Eta), represents entropy
S - data set
C - Quantity of all categories
c - category

2 task

Calculate the decision tree for a data set using the ID3 algorithm.

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
rainy	mild	high	true	no

Tabelle 1: Playing Tennis Game - data set

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

$$H(outlook = overcast) = -\left(\frac{4}{4}log_2\left(\frac{4}{4}\right) + 0log_2\left(0\right)\right)$$
$$= 0$$
$$H(outlook = sunny) = -\left(\frac{2}{5}log_2\left(\frac{2}{5}\right) + \frac{3}{5}log_2\left(\frac{3}{5}\right)\right)$$
$$\approx 0.971$$
$$H(outlook = rainy) = -\left(\frac{3}{5}log_2\left(\frac{3}{5}\right) + \frac{2}{5}log_2\left(\frac{2}{5}\right)\right)$$
$$\approx 0.971$$

feature's information gain:

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

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

$$H(temp = hot) = -\left(\frac{2}{4}log_2\left(\frac{2}{4}\right) + \frac{2}{4}log_2\left(\frac{2}{4}\right)\right)$$
$$= 1$$
$$H(temp = mild) = -\left(\frac{4}{6}log_2\left(\frac{4}{6}\right) + \frac{2}{6}log_2\left(\frac{2}{6}\right)\right)$$
$$\approx 0.918$$
$$H(temp = cool) = -\left(\frac{3}{4}log_2\left(\frac{3}{4}\right) + \frac{1}{4}log_2\left(\frac{1}{4}\right)\right)$$
$$\approx 0.811$$

feature's information gain:

$$IG(S, A_{outlook}) = 0.94 - \left(\frac{4}{14}1 + \frac{6}{14}0.918 + \frac{4}{14}0.811\right)$$
$$= 0.029$$

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

$$H(humidity = high) = -\left(\frac{3}{7}log_2\left(\frac{3}{7}\right) + \frac{4}{7}log_2\left(\frac{4}{7}\right)\right)$$
$$\approx 0.985$$
$$H(humidity = normal) = -\left(\frac{6}{7}log_2\left(\frac{6}{7}\right) + \frac{1}{7}log_2\left(\frac{1}{7}\right)\right)$$
$$\approx 0.592$$

feature's information gain:

$$IG(S, A_{outlook}) = 0.94 - \left(\frac{7}{14}0.985 + \frac{7}{14}0.592\right)$$
$$= 0.152$$

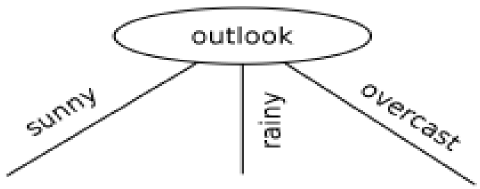
windy	FALSE	TRUE	sum
YES	6	3	9
NO	2	3	5
sum	8	6	14

$$H(windy = TRUE) = -\left(\frac{3}{6}log_2\left(\frac{3}{6}\right) + \frac{3}{6}log_2\left(\frac{3}{6}\right)\right)$$
$$= 1$$
$$H(windy = FALSE) = -\left(\frac{6}{8}log_2\left(\frac{6}{8}\right) + \frac{2}{8}log_2\left(\frac{2}{8}\right)\right)$$
$$\approx 0.811$$

feature's information gain:

$$IG(S, A_{outlook}) = 0.94 - \left(\frac{8}{14}0.811 + \frac{6}{14}1\right)$$
$$= 0.049$$

3. step: The feature with the largest IG will be selected as the root node. This results in the following tree:



A new root node must be determined recursively for each branch.

1. Calculate total entropy:
For the subset S_{sunny} following data set results:

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

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

2. Calculate Information Gain for each feature:

temperature	hot	mild	cool	sum
YES	0	1	1	2
NO	2	1	0	3
sum	2	2	1	5

$$H(temp = hot) = 0$$
$$H(temp = mild) = 1$$
$$H(temp = cool) = 0$$
$$IG(S_{sunny}, A_{temp}) = 0.971 - \left(\frac{2}{5}0 + \frac{2}{5}1 + \frac{1}{5}0\right)$$
$$\approx 0.571$$

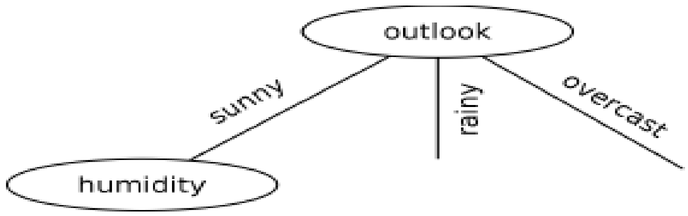
humidity	high	normal	sum
YES	0	2	2
NO	3	0	3
sum	3	2	5

$$H(humidity = high) = 0$$
$$H(humidity = normal) = 0$$
$$IG(S_{sunny}, A_{humidity}) = 0.971 - \left(\frac{3}{5}0 + \frac{2}{5}0\right)$$
$$\approx 0.971$$

windy	FALSE	TRUE	sum
YES	1	1	3
NO	1	2	3
sum	2	3	5

$$H(windy = FALSE) = 1$$
$$H(windy = TRUE) = -\left(\frac{1}{3}log_2\left(\frac{1}{3}\right) + \frac{2}{3}log_2\left(\frac{2}{3}\right)\right)$$
$$\approx 0.918$$
$$IG(S_{sunny}, A_{windy}) = 0.971 - \left(\frac{2}{5}1 + \frac{3}{5}0.918\right)$$
$$\approx 0.020$$

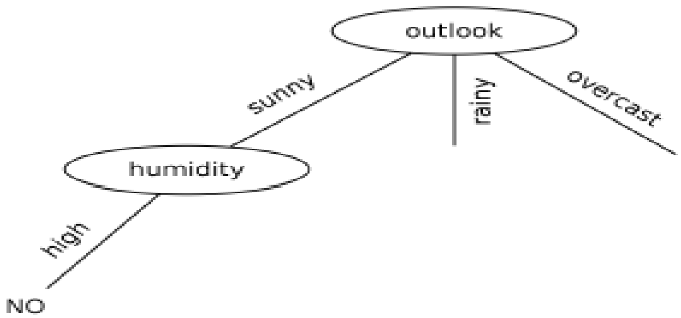
3. step: The feature with the largest IG will be selected as the root node. This results in the following tree:



1. Calculate total entropy:
   For the subset $S_{sunny,high}$ following data set results:

outlook	temp	humidity	windy	play
sunny	hot	high	false	no
sunny	hot	high	true	no
sunny	mild	high	false	no

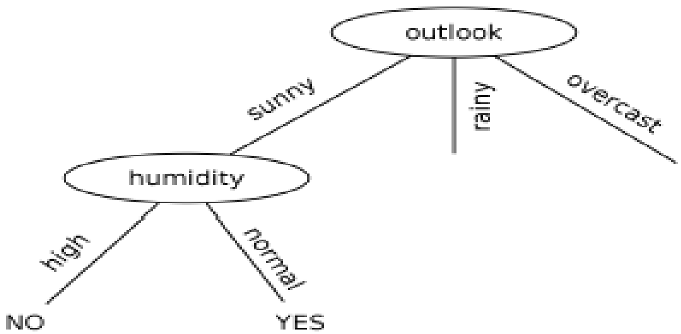
No entropy needs to be calculated, because all entries have the result „no“ .
This results in the following tree:



1. Calculate total entropy:
   For the subset $S_{sunny,normal}$ following data set results:

outlook	temp	humidity	windy	play
sunny	cool	normal	false	yes
sunny	mild	normal	true	yes

No entropy needs to be calculated, because all entries have the result „yes“ .
This results in the following tree:



1. Calculate total entropy:

outlook	temp	humidity	windy	play
rainy	mild	high	false	yes
rainy	cool	normal	false	yes
rainy	cool	normal	true	no
rainy	mild	normal	false	yes
rainy	mild	high	true	no

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

2. Calculate Information Gain for each feature:

temperature	mild	cool	sum
YES	2	1	3
NO	1	1	2
sum	3	2	5

$$H(temp = mild) = -\left(\frac{2}{3}log_2\left(\frac{2}{3}\right) + \frac{1}{3}log_2\left(\frac{1}{3}\right)\right)$$
$$\approx 0.918$$
$$H(temp = cool) = 1$$
$$IG(S_{rainy}, A_{temp}) = 0.971 - \left(\frac{3}{5}0.92 + \frac{2}{5}1\right)$$
$$\approx 0.019$$

humidity	high	normal	sum
YES	1	2	3
NO	1	1	2
sum	2	3	5

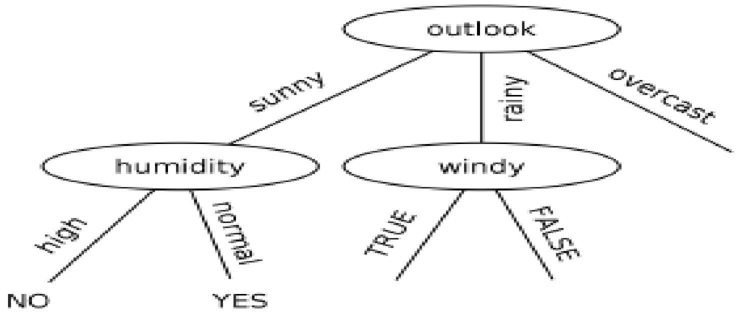
$$H(humidity = high) = 1$$
$$H(humidity = normal) = -\left(\frac{2}{3}log_2\left(\frac{2}{3}\right) + \frac{1}{3}log_2\left(\frac{1}{3}\right)\right)$$
$$\approx 0.918$$

$$IG(S_{rainy}, A_{humidity}) = 0.971 - \left(\frac{3}{5}0.92 + \frac{2}{5}1\right)$$
$$\approx 0.019$$

windy	TRUE	FALSE	sum
YES	0	3	3
NO	2	0	2
sum	2	3	5

$$H(windy = TRUE) = 0$$
$$H(windy = FALSE) = 0$$
$$IG(S_{rainy}, A_{windy}) = 0.971 - \left(\frac{3}{5}0 + \frac{2}{5}0\right)$$
$$\approx 0.971$$

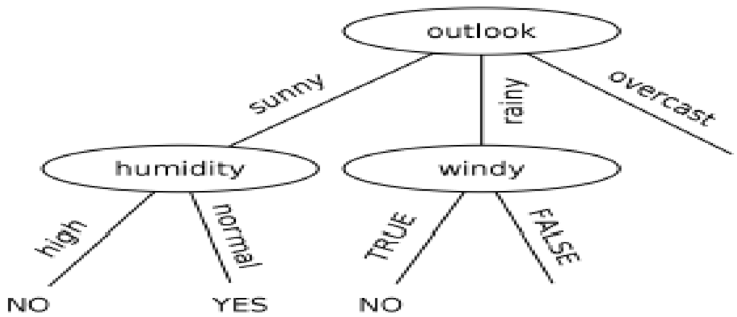
3. step: The feature with the largest IG will be selected as the root node. This results in the following tree:



1. Calculate total entropy:
   For the subset $S_{rainy,TRUE}$ following data set results:

outlook	temp	humidity	windy	play
rainy	cool	normal	true	no
rainy	mild	high	true	no

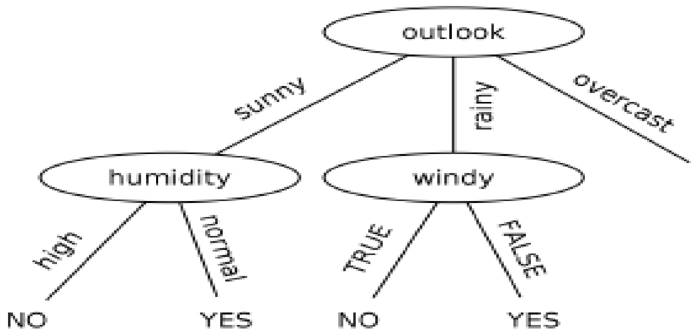
No entropy needs to be calculated, because all entries have the result „no" .
This results in the following tree:



1. Calculate total entropy:
   For the subset $S_{rainy,FALSE}$ following data set results:

outlook	temp	humidity	windy	play
rainy	mild	high	false	yes
rainy	cool	normal	false	yes
rainy	mild	normal	false	yes

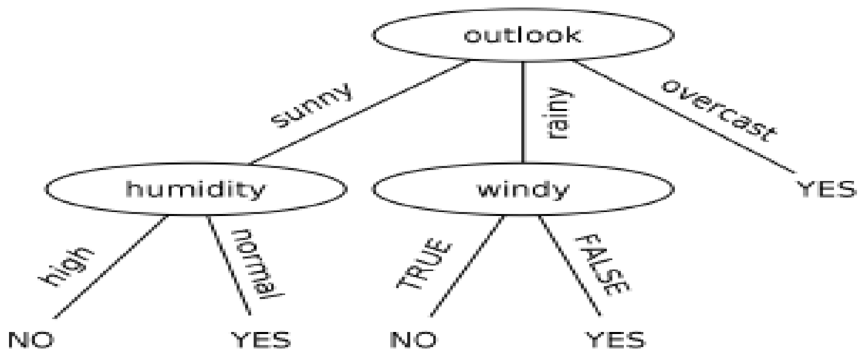
No entropy needs to be calculated, because all entries have the result „yes" .
This results in the following tree:



1. Calculate total entropy:
For the subset $S_{outlook=overcast}$ following data set results:

outlook	temp	humidity	windy	play
overcast	hot	high	false	yes
overcast	cool	normal	true	yes
overcast	mild	high	true	yes

No entropy needs to be calculated, because all entries have the result „yes" .
This results in the following tree:



First Solution with CART: Missing calculations on **CART method** using **GINI Index** as a metric (see page number of the corresponding lecture slides on the right top): see Notes Page in the lecture presentation.

Second Solution with CART (from Heike.Fitzke@de.kaercher.com, SS2020):

Oberster Knoten berechnen:

outlook:

	overcast	sunny	rainy	
Y	4	2	3	9
N	0	3	2	5
	4	5	5	

GINI(outlook) = ~0,343

humidity:

	high	normal	
Y	3	6	9
N	4	1	5
	7	7	

GINI(humidity) = 0,367

temperature:

	hot	mild	cool	
Y	2	4	3	9
N	2	2	1	5
	4	6	4	

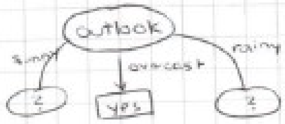
GINI (temperatur) = 0,44

windy:

	FALSE	TRUE	
Y	6	3	9
N	2	3	5
	8	6	

GINI (windy) = 0,429

⇒



For sunny: temperature | sunny

	hot	mild	cool	
Y	0	1	1	2
N	2	1	0	3
				5

GINI(temp | outlook = sunny) = 0,2

For windy: ~~temperature | windy~~ windy | sunny

	FALSE	TRUE	
Y	1	1	2
N	2	1	3
			5

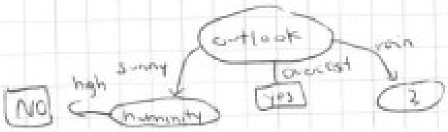
GINI (windy | sunny) = 0,267

For huminity: huminity | sunny

	high	normal	
Y	0	2	2
N	3	0	3
			5

GINI (huminity | sunny) = 0

⇒



Für windy: windy | rain

	FALSE	TRUE	
Y	3	0	3
N	0	2	2
	3	2	5

$$GINI(windy|rain) = \frac{3}{5}\left(1-\frac{3}{3}^2-\frac{0}{3}^2\right)+\frac{2}{5}\left(1-\frac{0}{2}^2-\frac{2}{2}^2\right)=0$$

Für huminity | rain

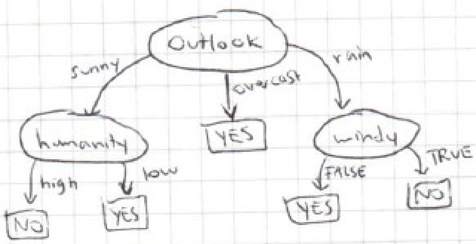
	high	normal	
Y	1	2	3
N	1	1	2
	2	3	5

$$GINI(huminity|rain) = \frac{2}{5}\left(1-\frac{1}{2}^2-\frac{1}{2}^2\right)+\frac{3}{5}\left(1-\frac{2}{3}^2-\frac{1}{3}^2\right)>0$$

Für temp | rain

	hot	mild	cool	
Y	0	2	1	3
N	0	1	1	2
		3	2	5

$$GINI(temp|rain) = \frac{3}{5}\left(1-\frac{2}{3}^2-\frac{1}{3}^2\right)+\frac{2}{5}\left(1-\frac{1}{2}^2-\frac{1}{2}^2\right)>0$$



Homework H4.2 - “Define the Decision Tree for UseCase “Predictive Maintenance” (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.67)
3. **Finalize the decision tree** by calculation the GINI-Index for the remaining values for the features “Temp.” and “Füllst.”

Optional*: Create and describe the **algorithm to automate the calculation** of steps 1. to 3.

First Solution (H.Völlinger):

Ad 1:

We calculate first the matrix for **Druck** by looking on the **Data Table**:

Nr.	Anl	Typ	Temp.	Druck	Füllst.	Fehler
1001	123	TN	244	140	4600	NO
1002	123	TO	200	130	4300	NO
1009	128	TSW	245	108	4100	YES
1028	128	TS	250	112	4100	NO
1043	128	TSW	200	107	4200	NO
1088	128	TO	272	170	4400	YES
1102	128	TSW	265	105	4100	NO
1119	123	TN	248	138	4800	YES
1122	123	TM	200	194	4500	YES

When we follow strictly the approach of slide 67, we have to consider intervals for classes "<= "and ">" and a split-point in the middle of the interval. See the slide p.67:

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	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	

So we get the following matrix:

Druck																				
Values			105		107		108		112		130		138		140		170		194	
Error			NO		NO		YES		NO		NO		YES		NO		YES		YES	
Split-Point	104		106		107,5		110		121		134		139		155		182		206	
Interval	< =	>	< =	>	< =	>	< =	>	< =	>	< =	>	< =	>	< =	>	< =	>	< =	>
NO	0	5	1	4	2	3	2	3	3	2	4	1	4	1	5	0	5	0	5	0
YES	0	4	0	4	0	4	1	3	1	3	1	3	2	2	2	2	3	1	4	0
GINI																				

****

We calculate next the matrix for **Temp.**:

Temp.																
Values			200, 200, 200		244		245		248		250		265		272	
Error			NO, NO,YES		NO		YES		YES		NO		NO		YES	
Split-Point	178		222		244,5		246,5		249		257,5		268,5		275,5	
Interval	< =	>	< =	>	< =	>	< =	>	< =	>	< =	>	< =	>	< =	>
NO	0	5	2	3	3	2	3	2	3	2	4	1	5	0	5	0
YES	0	4	1	3	1	3	2	2	3	1	3	1	3	1	4	0
GINI																

Nr.	Anl	Typ	Temp	Druck	Füllst.	Fehler
1001	123	TN	244	140	4600	NO
1002	123	TO	200	130	4300	NO
1009	128	TSW	245	108	4100	YES
1028	128	TS	250	112	4100	NO
1043	128	TSW	200	107	4200	NO
1088	128	TO	272	170	4400	YES
1102	128	TSW	265	105	4100	NO
1119	123	TN	248	138	4800	YES
1122	123	TM	200	194	4500	YES

****

Finally we calculate the matrix for **Füllst.**:

Füllst.																
Values			4100, 4100,4100		4200		4300		4400		4500		4600		4800	
Error			NO, NO, YES		NO		NO		YES		YES		NO		YES	
Split-Point	4050		4150		4250		4350		4450		4550		4700		4900	
Interval	< =	>	< =	>	< =	>	< =	>	< =	>	< =	>	< =	>	< =	>
NO	0	5	2	3	3	2	4	1	4	1	4	1	5	0	5	0
YES	0	4	1	3	1	3	1	3	2	2	3	1	3	1	4	0
GINI																

Nr.	Anl	Typ	Temp	Druck	Füllst.	Fehler
1001	123	TN	244	140	4600	NO
1002	123	TO	200	130	4300	NO
1009	128	TSW	245	108	4100	YES
1028	128	TS	250	112	4100	NO
1043	128	TSW	200	107	4200	NO
1088	128	TO	272	170	4400	YES
1102	128	TSW	265	105	4100	NO
1119	123	TN	248	138	4800	YES
1122	123	TM	200	194	4500	YES

Ad2:

We calculate first for all values of **Druck** the GINI- Index:

See the following matrix, which shows the results.

Druck										
Values		105	107	108	112	130	138	140	170	194
Error		NO	NO	YES	NO	NO	YES	NO	YES	YES
Split-Point	**104**	**106**	**107,5**	**110**	**121**	**134**	**139**	**155**	**182**	**206**
Interval	< = >	< = >	< = >	< = >	< = >	< = >	< = >	< = >	< = >	< = >
NO	0 5	1 4	2 3	2 3	3 2	4 1	4 1	5 0	5 0	5 0
YES	0 4	0 4	0 4	1 3	1 3	1 3	2 2	2 2	3 1	4 0
GINI	**0.494**	**0.444**	**0.381**	**0.481**	**0.433**	**0.344**	**0.444**	**0.317**	**0.417**	**0.494**

First we calculate Gini (Druck) for the value= 139:

Gini (Druck)

= 6/9***Gini(<=139)** + 3/9***Gini(>139)**'

= 2/3*(1- (4/6)²-(2/6)²) + 1/3*(1-(1/3)²-(2/3)²)'

= 2/3*((36-16-4)/36) + 1/3*((9-1-4)/9) = 8/27 + 4/27 = 4/9 = **~0.444'**

Second we calculate Gini (Druck) for the value= 155:

Gini (Druck)

= 7/9***Gini(<=0155)** + 2/9***Gini(>155)**'

= 7/9*(1- (2/7)²-(5/7)²) + 2/9*(1-(2/2)²-(0/2)²)'

= 7/9*((49-4-25)/49) + 0 = 7/9*(20/49) = 20/63 = **~0.317'**

Third we calculate GINI (Druck) for the value= 182:

Gini (Druck)

= 8/9***Gini(<=182)** + 1/9***Gini(>182)**'

= 8/9*(1- (3/8)²-(5/8)²) + 1/9*(1-(1/1)²-(0/1)²) '

= 8/9*((64-9-25)/49) + 0 = 8/9*(30/64) = 10/24 = 5/12 **~0.417'**

For the rest of the calculations see the following screenshot:

$$Gini(105) = Gini(200) = 0 + 1*\left(1-\left(\tfrac{4}{9}\right)^2-\left(\tfrac{5}{9}\right)^2\right) = \frac{81-25-16}{81} = \frac{40}{81} = 0.494$$

$$Gini(107) = \frac{1}{9}\cdot\underbrace{(1-0-1)}_{=0} + \frac{8}{9}\left(1-\left(\tfrac{1}{2}\right)^2-\left(\tfrac{1}{2}\right)^2\right) = \frac{8}{9}\cdot\frac{1}{2} = \frac{4}{9} = 0{,}444$$

$$Gini(108) = \frac{2}{9}\underbrace{(1-0-1)}_{=0} + \frac{7}{9}\left(1-\left(\tfrac{3}{7}\right)^2-\left(\tfrac{4}{7}\right)^2\right) = \frac{7(49-9-16)}{9\cdot 49} = \frac{24}{63} = \frac{8}{21} = 0{,}381$$

$$Gini(113) = \frac{3}{9}\cdot\left(1-\left(\tfrac{1}{3}\right)^2-\left(\tfrac{2}{3}\right)^2\right) + \frac{2}{3}\left(1-\left(\tfrac{1}{2}\right)^2-\left(\tfrac{1}{2}\right)^2\right) = \frac{3(9-1-4)}{9\cdot 9} + \frac{2}{3}\cdot\frac{1}{2} = \frac{4}{27}+\frac{9}{27} = \frac{13}{27} = 0{,}481$$

$$Gini(130) = \frac{4}{9}\cdot\left(1-\left(\tfrac{3}{4}\right)^2-\left(\tfrac{1}{4}\right)^2\right) + \frac{5}{9}\cdot\left(1-\left(\tfrac{2}{5}\right)^2-\left(\tfrac{3}{5}\right)^2\right) = \frac{4(16-9-1)}{9\cdot 16} + \frac{5}{9}\cdot\frac{(25-4-9)}{25} = \frac{6}{36} = \frac{1}{6} + \frac{12}{45} = \frac{4}{15} = \frac{13}{30} = 0.433$$

We calculate next for all values of **Temp.** the GINI- Index:
See the following matrix, which shows the results:

Temp.								
Values		200, 200, 200	244	245	248	250	265	272
Error		**NO, NO,YES**	**NO**	**YES**	**YES**	**NO**	**NO**	**YES**
Split-Point	**178**	**222**	**244,5**	**246,5**	**249**	**257,5**	**268,5**	**275,5**
Interval	< = >	< = >	< = >	< = >	< = >	< = >	< = >	< = >
NO	0 5	2 3	3 2	3 2	3 2	4 1	5 0	5 0
YES	0 4	1 3	1 3	2 2	3 1	3 1	3 1	4 0
GINI	**0.494**	**0.481**	**0.433**	**0.489**	**0.481**	**0.492**	**0.417**	**0.494**

We see that the value of the **GINI-index** only depends on the distribution of **YES** and **NO's**:
For the values **178, 222, 244,5, 249, 268,5** and **275,5** we can use the GINI of Druck, since the distribution of YES and NO's are same
So we need only to calculate GINI(Temp.) for the values= **246,5 and 257,5**

First we calculate GINI (Temp.) for the value= **246,5:**
Gini (Temp.)
= 5/9***Gini(<=246,5)** + 4/9***Gini(>246,5)'**
= 5/9*(1- (3/5)²-(2/5)²) + 4/9*(1-(2/4)²-(2/4)²)'
= 5/9*((25-9-4)/25) + 4/9*(1-1/4-1/4) = 5/9*(12/25) + 4/9*1/2 = 4/15 + 2/9 = 22/45 **~0.489'**
Second we calculate GINI (Druck) for the value= **257,5**:
Gini (Temp.)
= 7/9***Gini(<=257,5)** + 2/9***Gini(>257,5)'**
= 7/9*(1- (4/7)²-(3/7)²) + 2/9*(1-(1/2)²-(1/2)²) '
= 7/9*((49-16-9)/49) + 1/9 = 7/9*(24/49) + 1/9 = 8/21 + 1/9 = 31/63 **~0.492'**

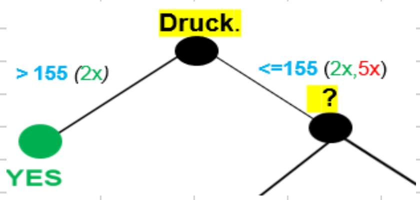
Finally we calculate all values of **Füllst.** the GINI- Index:
See the following matrix, which shows the results:

Füllst.								
Values		4100, 4100,4100	4200	4300	4400	4500	4600	4800
Error		NO, NO, YES	NO	NO	YES	YES	NO	YES
Split-Point	4050	4150	4250	4350	4450	4550	4700	4900
Interval	< = >	< = >	< = >	< = >	< = >	< = >	< = >	< = >
NO	0 5	2 3	3 2	4 1	4 1	4 1	5 0	5 0
YES	0 4	1 3	1 3	1 3	2 2	3 1	3 1	4 0
GINI	**0.494**	**0.481**	**0.433**	**0.344**	**0.444**	**0.492**	**0.417**	**0.494**

All values of GINI- Indexes are calculated above.
For example GINI(Füllst.) for the value= 4450 is the same as GINI(Druck) for the value=139.

****

RESULT: When we consider the lowest GINI we see it with 0.317 for the feature DRUCK for the value 155.
=> DRUCK = Root-Node and the Split-Value is at 155. Our descion tree is now:

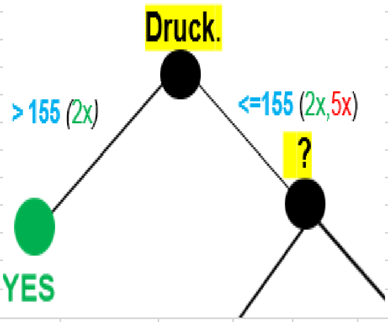


Ad3:

We need to calculate the GINI-Indexes for all remaining 7 values (where Druck < 170) for the Features **Temp.** and **Füllst.**:

We need to calculate the GINI-Indexes for all remaining 7 values (where Druck < =155) for the Features **Temp.** and **Füllst.**:

Nr.	Anl	Typ	Temp.	Druck	Füllst.	Fehler
1001	123	TN	244	140	4600	NO
1002	123	TO	200	130	4300	NO
1009	128	TSW	245	108	4100	YES
1028	128	TS	250	112	4100	NO
1043	128	TSW	200	107	4200	NO
~~1088~~	~~128~~	~~TO~~	~~272~~	~~170~~	~~4400~~	~~YES~~
1102	128	TSW	265	105	4100	NO
1119	123	TN	248	138	4800	YES
~~1122~~	~~123~~	~~TM~~	~~200~~	~~194~~	~~4500~~	~~YES~~



Temp.							
Values		200, 200	244	245	248	250	265
Error		NO, NO	NO	YES	YES	NO	NO
Split-Point	178	222	244,5	246,5	249	257,5	272,5
Interval	< = >	< = >	< = >	< = >	< = >	< = >	< = >
NO	0 5	2 3	3 2	3 2	3 2	4 1	5 0
YES	0 2	0 2	0 2	1 1	2 0	2 0	2 0
GINI	0.408	0.343	0.286	0.405	0.343	0.405	0.408

GINI (178) = GINI (272,5) = $0/7 \cdot (GINI{<=}178) + 7/7 \cdot GINI(>178) = 0 + 1-(5/7)^2-(2/7)^2 = (49-4-25)/49 = 20/49$ ~ **0.408**

GINI (222) = GINI (249) = $2/7 \cdot (1-0-1) + 5/7 \cdot (1-(3/5)^2-(2/5)^2 = 5/7 \cdot ((25-9-4)/25) = 1/7 \cdot (12/5) = 12/35$ ~ **0.343**

GINI (244,5) = $3/7 \cdot (1-0-1) + 4/7 \cdot (1-(1/2)^2-(1/2)^2 =$ **0 +** $4/7 \cdot (1/2) = 4/14 = 2/7$ ~ **0.286**

GINI (246,5) = $4/7 \cdot (1-(3/4)^2-(1/4)^2) + 3/7 \cdot (1-(1/3)^2-(2/3))^2 =$ $4/7 \cdot ((16-9-1)/16) + 3/7 \cdot ((9-1-4)/9) = 6/28 + 4/21 = 17/34$ ~ **0.405**

GINI (257,5) = $6/7 \cdot (1-(4/6)^2-(2/6)^2) + 1/7 \cdot (1-0-1))^2 =$ $6/7 \cdot (1-(2/3)^2-(1/3)^2 + 0 = 6/7 \cdot (4/9) = 6/7 \cdot 4/9 = 8/21$ ~ **0.405**

The final task ist to calculate the table for **Füllst:**

Nr.	Anl	Typ	Temp.	Druck	Füllst.	Fehler
1001	123	TN	244	140	4600	NO
1002	123	TO	200	130	4300	NO
1009	128	TSW	245	108	4100	YES
1028	128	TS	250	112	4100	NO
1043	128	TSW	200	107	4200	NO
~~1088~~	~~128~~	~~TO~~	~~272~~	~~170~~	~~4400~~	~~YES~~
1102	128	TSW	265	105	4100	NO
1119	123	TN	248	138	4800	YES
~~1122~~	~~123~~	~~TM~~	~~200~~	~~194~~	~~4500~~	~~YES~~

Füllst.						
Values		4100, 4100, 4100	4200	4300	4600	4800
Error		NO, NO, YES	NO	NO	NO	YES
Split-Point	4050	4150	4250	4450	4700	4900
Interval	< = >	< = >	< = >	< = >	< = >	< = >
NO	0 5	2 3	3 2	4 1	5 0	5 0
YES	0 2	1 1	1 1	1 1	1 1	2 0
GINI	0.408	0.405	0.405	0.371	0.238	0.408

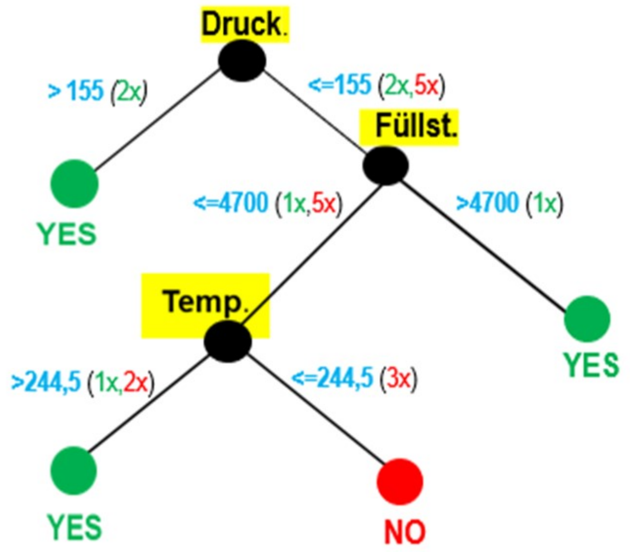
For the Values 4050, 4150, 4250 and 4900 we can use the GINI calculation from Temp.

So we need only to calculate the GINI for **4450** and **4700**:

GINI (4450) = $5/7 \cdot (1-(4/5)^2- (1/5)^2) + 2/7 \cdot (1-(1/2)^2-(1/2)^2) = 5/7 \cdot ((25-16-1)/25) + 2/7 \cdot (1/2) = 8/35 + 1/7 = 13/35 = 12/63 = 4/21$ ~ **0.371**

GINI (4700) = $6/7 \cdot (1-(5/6)^2-(1/6)^2) + 1/7 \cdot (1-0-1) =$ $6/7 \cdot ((36-25-1)/36) = (6/7) \cdot (10/36) = 10/42 = 5/21$ ~ **0.238**

Result: When we compare the lowest GINI values for **Temp**. and **Füllst**., we see **GINI (Temp. = 244,5) = 0.286** and **GINI (Füllst. = 4700)= 0.238**. So we get the following *final decision tree*:



If you look at the number of occurrences per branch ("Zweig"), then you can determine the leaf ("Blatt"). We see that the leaf **(>244,5)** is set to **YES** even if you have two **NO**. This is because the branch **(<=244,5)** is clear. Nevertheless, we will need more data to have a "better" situation in this leaf. Usually in realistic scenarios you have data-sets that have more than several thousands to millions records, such that you get a much clearer decision.

Remark: In this example we have a dataset of only 9 rows. In the **industrial production** (i.e. mechanical engineering) we have much more values (*thousands to millions*). So we need to develop an algorithm to run all the calculations of the GINI-Indexes.
Optional (SW)*: Describe and create the **algorithms to automate the calculation** of the steps 1.to 3.

Homework H4.3* - "Create and describe the algorithm to automate the calculation of the Decision Tree for UseCase "Predictive Maintenance"

Groupwork (2 Persons): Create and describe the **algorithm to automate the calculation** of steps 1. to 3. of homework H4.2. Do the following steps (following the algorithm described in the lecture):

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.67)
3. **Finalize the decision tree** by calculation the GINI-Index for the remaining values for the features "Temp." and "Füllst."

Solution: Created by H. Fritze. & P. Mäder (DHBW, SS2020) and H. Völlinger (DHBW, WS2020). The following screenshot are from a Jupyter Notebook (using Python3):

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 Fitzke and Paul Mäder.

The solution is part of seminarpaper SW07 in the list of seminarpapers (http://wwwlehre.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_3ipynb" 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 Libraies and check the versions.
2. Import the data from csv-file: "Homework-H3_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 (ommit not needed columns).

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('Homework-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"):
```

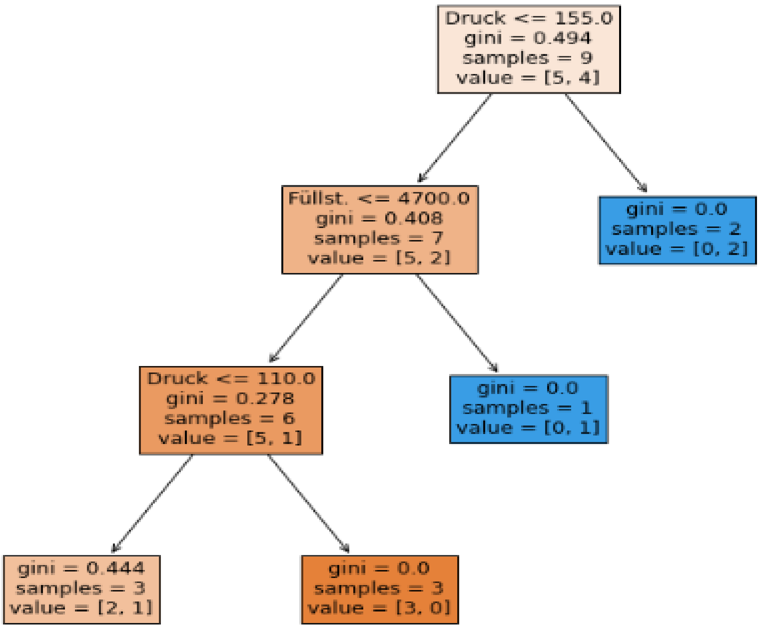
Out[2]:

	Temp.	Druck	Füllst.	Fehler
0	244	140	4600	0
1	200	130	4300	0
2	245	108	4100	1
3	250	112	4100	0
4	200	107	4200	0
5	272	170	4400	1
6	265	105	4100	0
7	248	138	4800	1
8	200	194	4500	1

Step 2: Define the Decision Tree & Calculate GINI Indices

1. Define the features and the target value ("Fehler")
2. Call Function DecisisontreeClassifier with paramters
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()
```



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)
```

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, ie.: -> 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  106.0   4100.0
1  200.0  107.5   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  139.0   4450.0
6  257.5  155.0   4550.0
7  268.5  182.0   4700.0
      Temp.     Druck   Füllst.
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  0.488889  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.  Füllst.
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.   Füllst.
0  0.408163  0.408163
1  0.342857  0.408163
2  0.285714  0.404762
3  0.404762  0.404762
4  0.342857  0.371429
5  0.380952  0.238095
Füllst. 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
```

In [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 ******************
```

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".
Link to article: http://d-nb.info/992620961/34

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

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

<u>First Solution</u>: by Adrian Koslowski; 1.4.2020:

Task: Summary of the chapter „Aufbau des klassischen Qualitätsregelkreises” of Hans W. Dörmann Osuma‘s „Ansatz für ein prozessintegriertes qualitätsregelungssystem für nicht stabile Prozesse“

Subheadings

- „Aufgaben“
- „Voraussetzungen für die Datenerfassung“
- „Datenauswertung“
    - „Data Understanding“
    - „Data Preparation“
    - „Modellierung und Datenanalyse“
    - „Implementierung“

„Aufgaben“ - Functions

During production data is collected and compared to target values. If the values do not match, the system automatically acts to correct itself:

„Voraussetzungen für die Datenerfassung“ -Requirements for data collection

- Process must be formally describable
- Data must be measurable
- Values must be processable

„Datenauswertung“ – Data processing

4 phases:

1. Plan
2. Do
3. Check
4. Act

„Data Understanding“

- What variables are relevant for my process?
- What must be taken into consideration?

„Data Preparation “

- Goal: Creation of a table with which current data can be compared to target values
- Generation of initial target values by testing and measurements as well as opinions of specialists and more

„Modellierung und Datenanalyse“ – Modeling and Data Analysis

- Creation of a model of the real process
- Search for dependencies and causalities

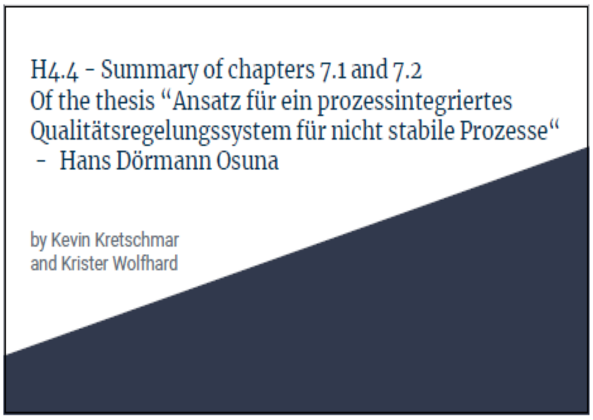
- CART- and CHAID- decision trees as well as rule-based System as possible methods

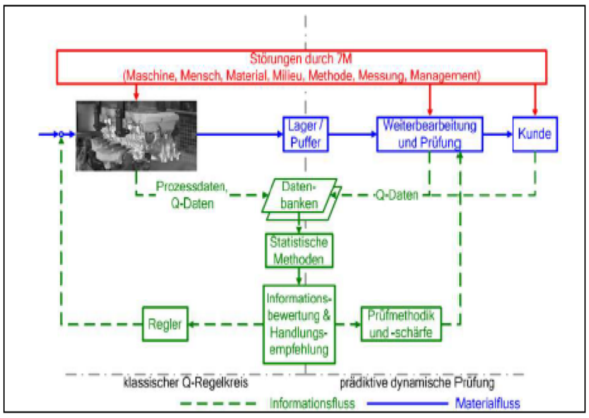
„Implementierung" - Implementation

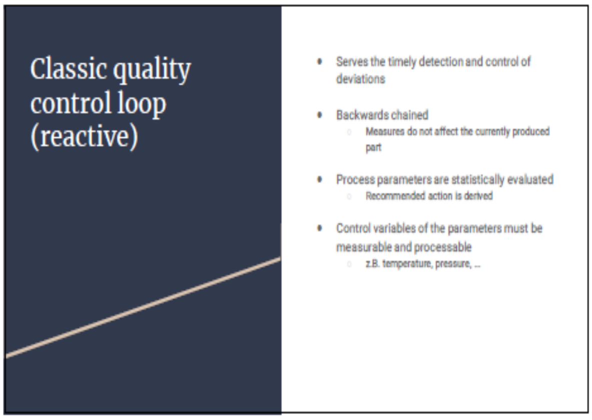
- Creation of new variables and target values based on new solutions
- Adaptation of existing target values to accommodate new knowledge and rules

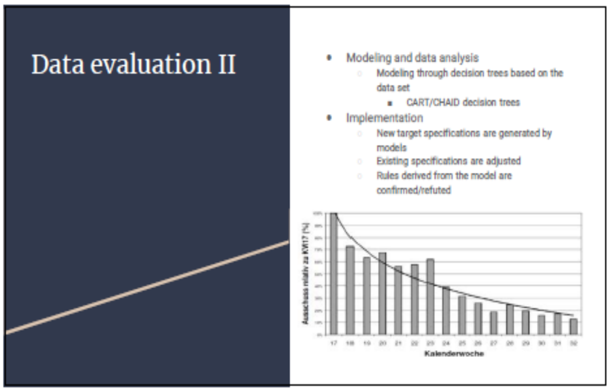
****

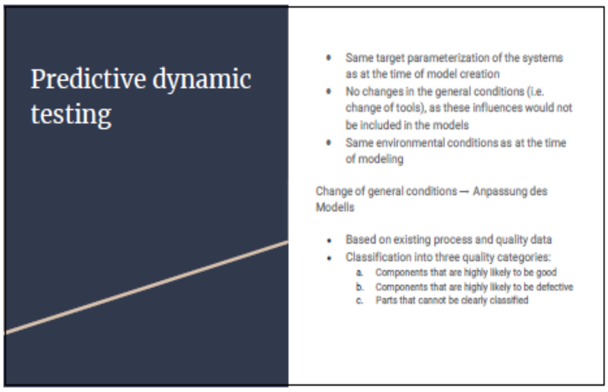
<u>Second Solution</u>: by Kevin Kretschmar & Krister Wolfhard; 27.10.2020:

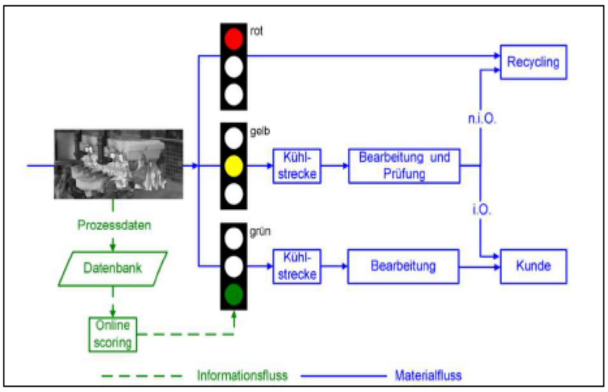


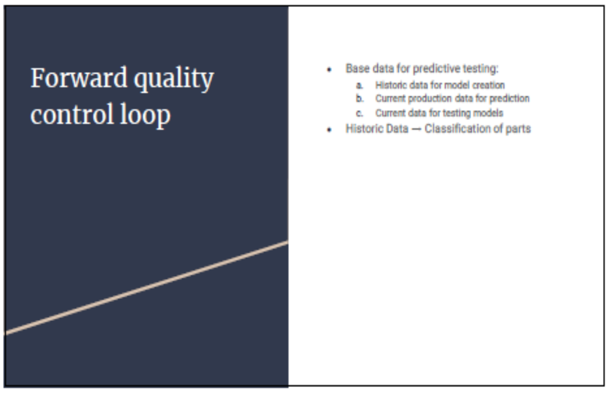


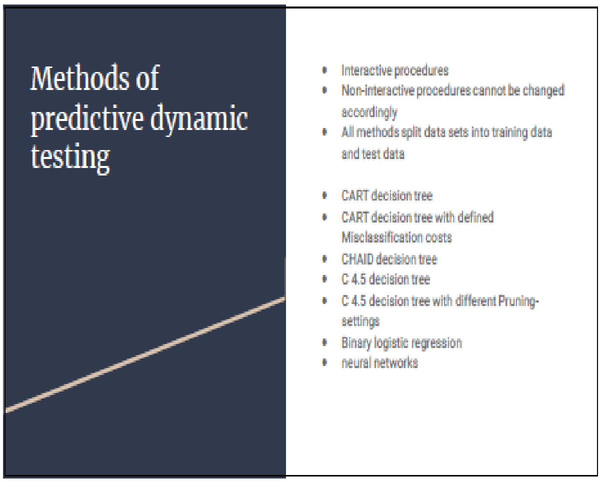


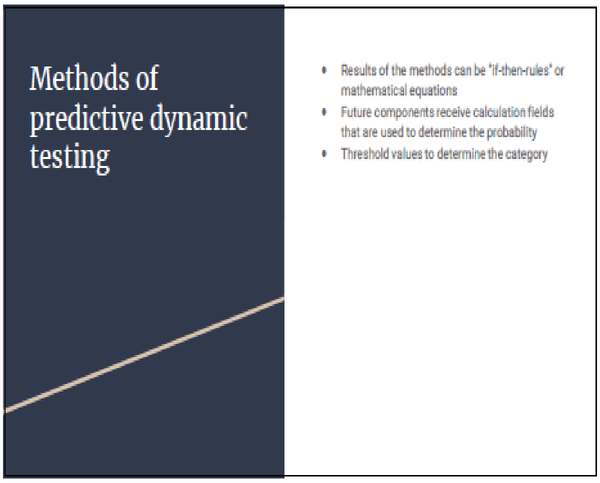








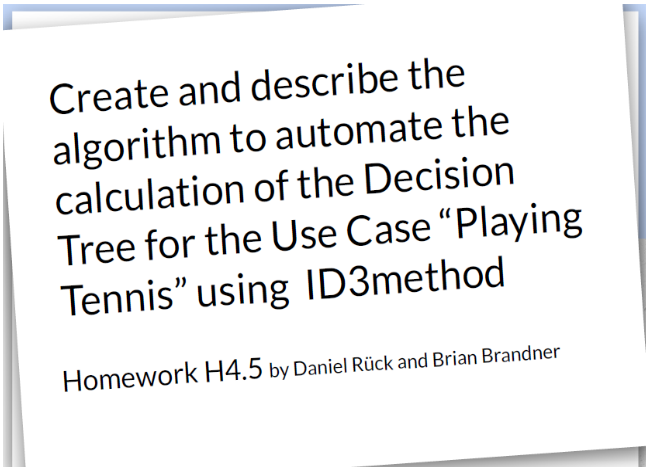


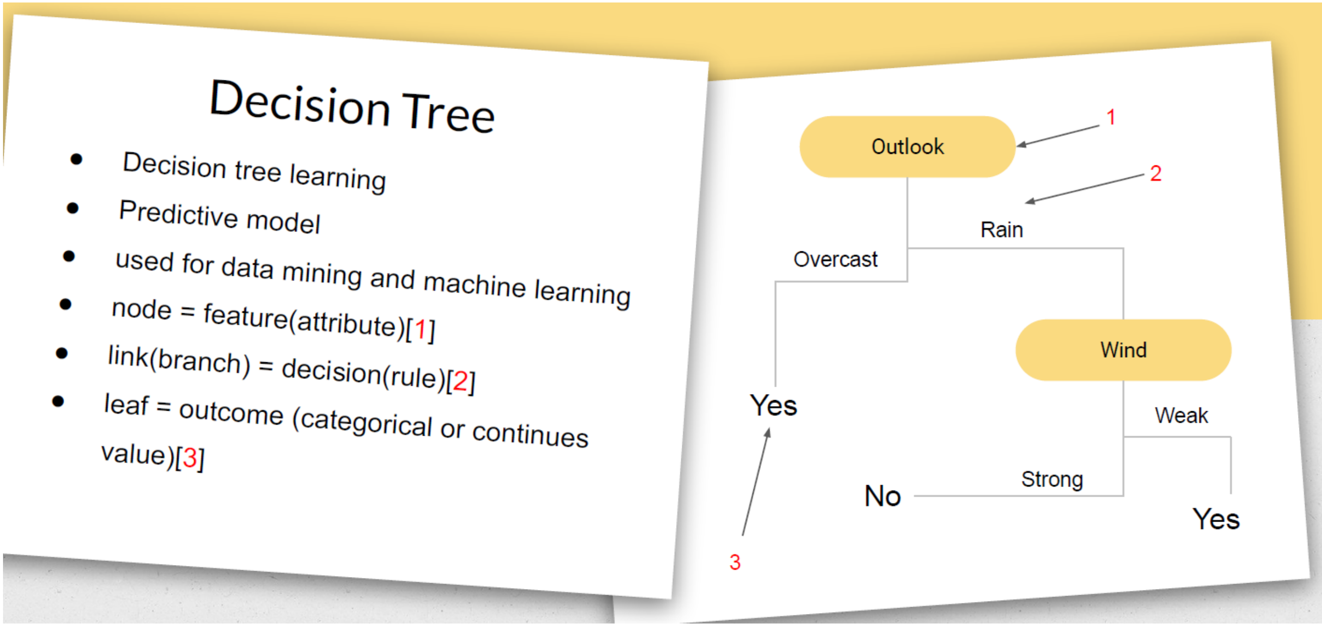


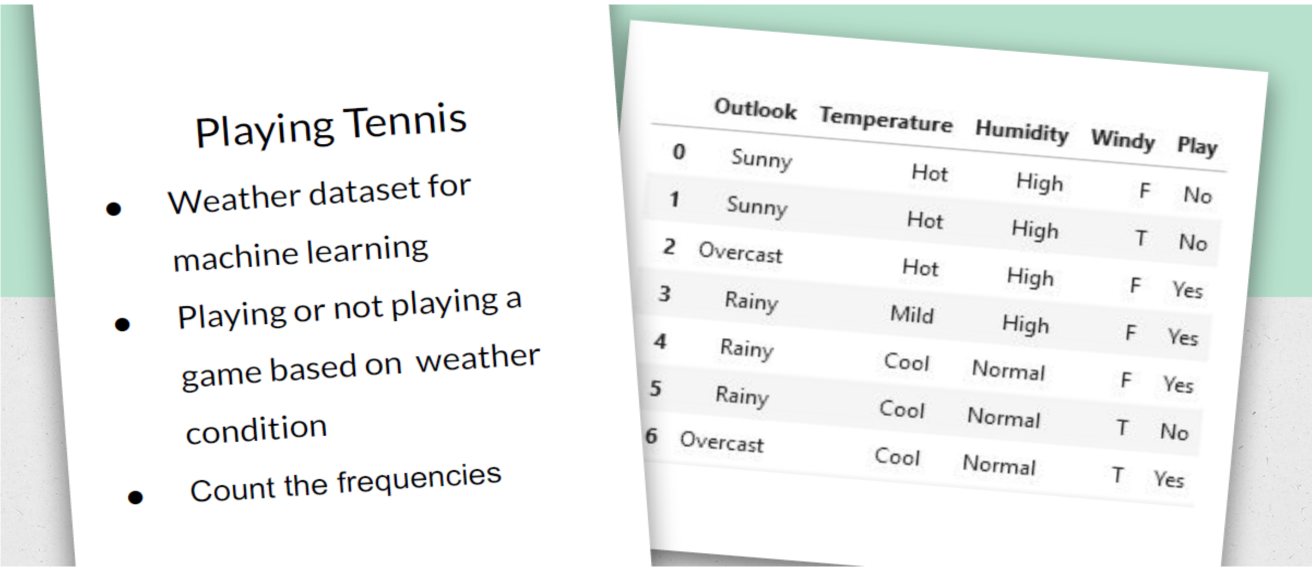
Homework H4.5* - "Create and describe the algorithm to automate the calculation of the Decision Tree for the Use Case "Playing Tennis" using ID3 method"

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

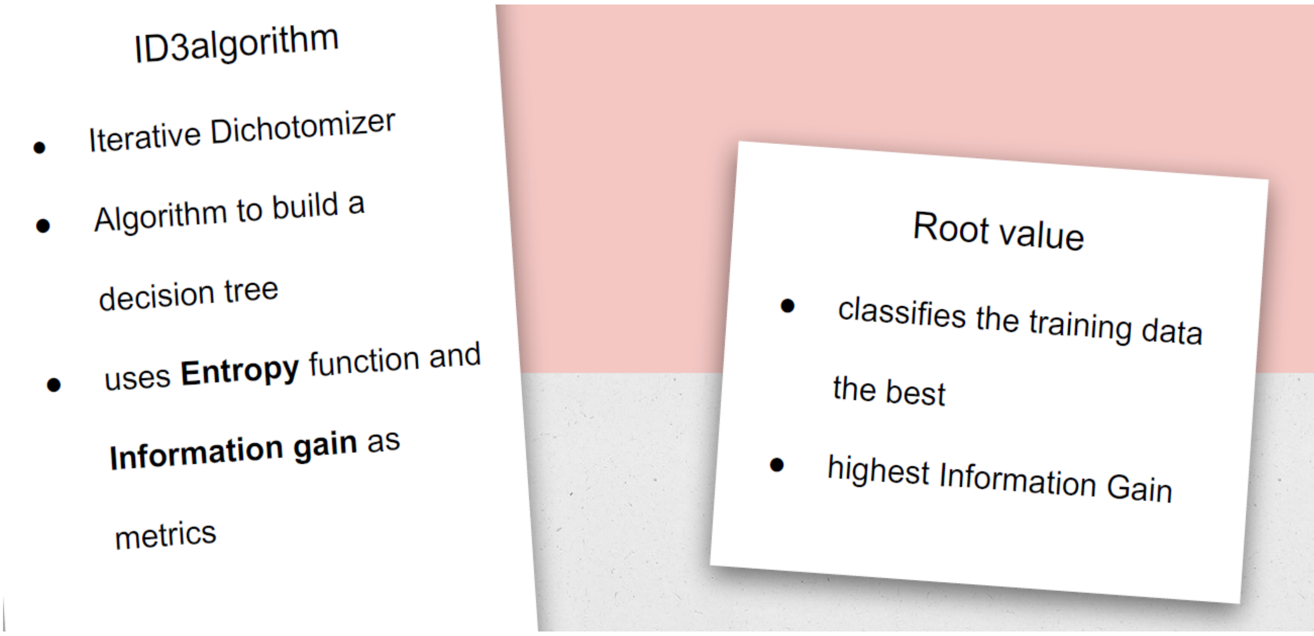
<u>First Solution</u>: by Daniel Rück & Brian Brandner; 27.10.2020:

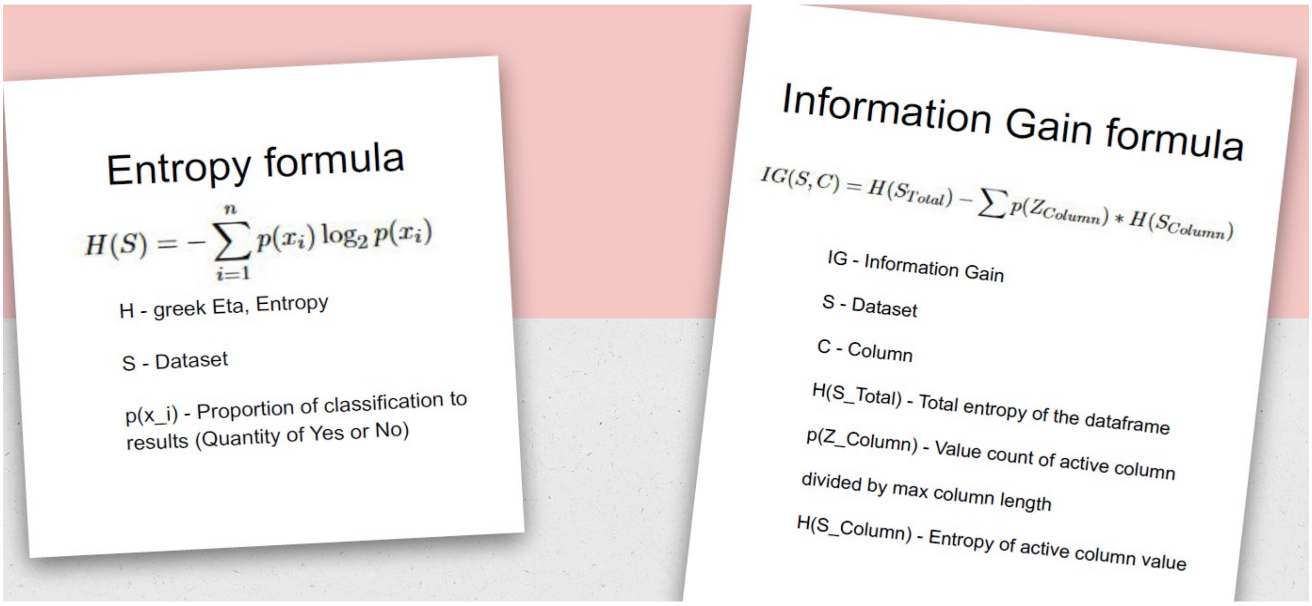


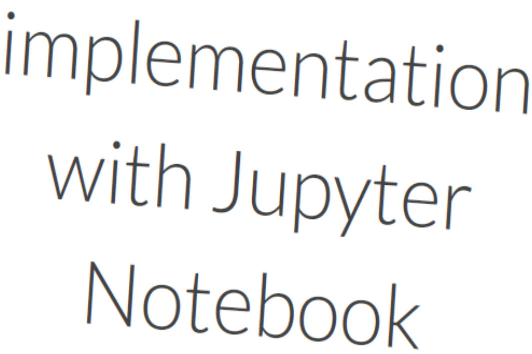




	Outlook	Temperature	Humidity	Windy	Play
0	Sunny	Hot	High	F	No
1	Sunny	Hot	High	T	No
2	Overcast	Hot	High	F	Yes
3	Rainy	Mild	High	F	Yes
4	Rainy	Cool	Normal	F	Yes
5	Rainy	Cool	Normal	T	No
6	Overcast	Cool	Normal	T	Yes


ID3algorithm
Iterative Dichotomizer
Algorithm to build a decision tree
uses Entropy function and Information gain as metrics
Root value
classifies the training data the best
highest Information Gain


Entropy formula
$H(S) = -\sum_{i=1}^{n} p(x_i) \log_2 p(x_i)$
H - greek Eta, Entropy
S - Dataset
p(x_i) - Proportion of classification to results (Quantity of Yes or No)
Information Gain formula
$IG(S, C) = H(S_{Total}) - \sum p(Z_{Column}) * H(S_{Column})$
IG - Information Gain
S - Dataset
C - Column
H(S_Total) - Total entropy of the dataframe
p(Z_Column) - Value count of active column divided by max column length
H(S_Column) - Entropy of active column value


implementation with Jupyter Notebook

H4_5

October 26, 2020

1 Decision Tree for the Use Case "Playing Tennis" using ID3 method

Homework H4.5 from Exercises to Lesson ML4Homework of the lecture "Machine Learning - Concepts & Algorithms". DHBW Stuttgart (WS2020) ***By Brian Brandner and Daniel Rück 26. October 2020***

The ID3 (Iterative Dichotomiser 3) method is used to generate a decision tree from a dataset. To achieve this the algorithm needs the **Entropy** formula to determine impurity of data and the **Information Gain**, which indicates the most relevant dataset attribut

1.1 Import of libraries

- **pandas** - loads the dataset and provids necessary frame details
- **math** - calculates in the alogarithm to the base 2
- **pprint** - prints the dictionary storage
- **IPython** - uses display, Math and Latex to for printing the formula
- **sys** - version information to python

[1]:
```
# libraries to import
import pandas as pd
import math
import pprint
from IPython.display import display, Math, Latex
# python version check library
import sys

# print python version, for some imports this version number is viewed as␣
↪theirs.
print("python {}".format(sys.version))
# version of pandas
print("pandas {}".format(pd.__version__))
```

See the rest of this Jupyter Notebooks H4.3 with the name "*Homework_H4.5-DecTree_ID3.ipynb*" (as PDF: *"Homework_H4.5-DecTree_ID3.pdf")* in **[HVö-6]:** GitHUb/HVoellinger: https://github.com/HVoellinger/Lecture-Notes-to-ML-WS2020

Exercises to Lesson ML5: simple Linear Regression (sLR) & multiple Linear Regression (mLR)

Homework H5.1 - "sLR manual calculations of R² & Jupyter Notebook (Python)"

Consider we have the 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 R-Square R^2 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? (1 Person, 15 minutes). (Hint: R^2-Square= 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 R^2 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 R^2 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 := Sumi(f(xi) – Mean(yi))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:

Part a**:** (H.Völlinger & Sam Matsa, INF17B, 5.4.2020):

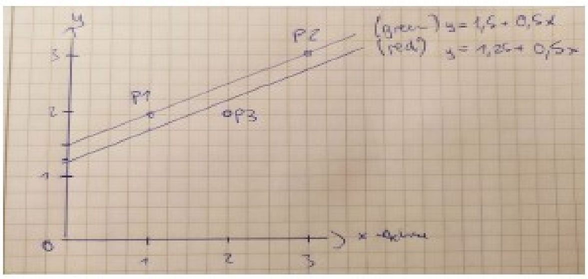
Task

Consider we have the 3 points on the xy-plane

- P1 = (1|2)
- P2 = (3|3)
- P3 = (2|2)

and two estimated SLR-lines:

- y = 1,5 + 0,5*x
- y = 1,25 + 0,5*x



Which estimation (red or green) is better?

Berechne "rote" Gerade:

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

$\bar{y} = \frac{y_1 + y_2 + y_3}{3} = \frac{2+2+3}{3} = \frac{7}{3}$

$SSE := \sum_{i=1}^{3} (y_i - f_i)^2 = (2 - f_1)^2 + (3 - f_2)^2 + (2 - f_3)^2$

$= (2 - 1{,}75)^2 + (3 - 2{,}75)^2 + (2 - 2{,}25)^2$

$= (0.25)^2 + (0.25)^2 + (0{,}25)^2 = 3 \cdot (\frac{1}{4})^2 = \frac{3}{16} = 0{,}1875$

$SST := \sum_{i=1}^{3} (y_i - \bar{y})^2 = (2 - \frac{7}{3})^2 + (3 - \frac{7}{3})^2 + (2 - \frac{7}{3})^2 = (\frac{1}{3})^2 + (\frac{2}{3})^2 + (\frac{1}{3})^2 = \frac{1+4+}{9}$

$= \frac{6}{9} = \frac{2}{3}$

$\Rightarrow R^2 = 1 - \frac{3 \cdot 3}{16 \cdot 2} = 1 - \frac{9}{32} = \frac{32-9}{32} = \frac{23}{32} = 0{,}71875$

Berechnung der "grünen" Gerade:

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

$SSE = \sum_{i=1}^{3} (y_i - f_i)^2 = \underbrace{(2-2)^2}_{=0} + \underbrace{(3-3)^2}_{=0} + (2 - 2{,}5)^2 = (\frac{1}{2})^2 = \frac{1}{4} = 0.2$

$SST = \frac{2}{3}$ (siehe oben

$\Rightarrow R^2 = 1 - \frac{1 \cdot 3}{4 \cdot 2} = \frac{8-3}{8} = \frac{5}{8} = 0.625$

Answer: The regression line $y = 1{,}25 + 0.5 \cdot x$ is better regression, since $R^2 = 0{,}71875$ is greater than $R^2 = 0.625$

We calculate for the "*center of mass*" [M(x), M(y)] = [2, 7/3]:

y(2) =1,5 + 0,5*2 = 2,5 > M(y)

y(2) = 1,25+0,5*2 = 2,25 < M(y)

Make some comments concerning the condition SST = SSE +SSR:

Manuel calculation of two sLR-lines (green ,red) (Homework (H5.1_a) + Compare with optimal sLR-line (homewrk (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*x* or *y = 1,25+0,5*x* ?

With the defintion 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!

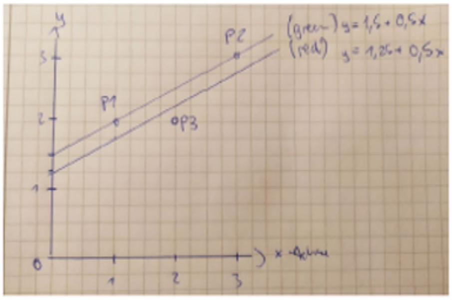
Solution:

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:

	needed for calculation of a and b				Needed for calculation of R^2				SST = SSE + SSR ?	
i	*xi*	*yi*	x_i*y_i	x_i^2	$y(x_i)$	$SSE=sum(yi-y(x_i))^2$	$SST=sum(yi-M(y))^2$	R^2	$SSR=sum(y(xi) - M(y))^2$	R^2
1	1	2	2	1	2,00	0,0000000	0,1111111		0,1111111	
2	3	3	9	9	3,00	0,0000000	0,4444444		0,4444444	
3	2	2	4	4	2,50	0,2500000	0,1111111		0,0277778	
sum	6	7	15	14		0,2500000	0,6666667	0,6250000	0,5833333	0,8750000
									0,8333333	<--SSR + SSE

- y = 1,5 + 0,5*x
- y = 1,25 + 0,5*x



Which estimation (red or green) is better?

Needed for calculation of R^2				SST = SSE + SSR ?	
$y(x_i)$	$SSE=sum(yi-y(x_i))^2$	$SST=sum(yi-M(y))^2$	R^2	$SSR=sum(y(xi) - M(y))^2$	R^2
1,75	0,0625000	0,1111111		0,3402778	
2,75	0,0625000	0,4444444		0,1736111	
2,25	0,0625000	0,1111111		0,0069444	
	0,1875000	0,6666667	0,7187500	0,5208333	0,7812500
				0,7083333	<--SSR + SSE

From Homework (H5.1_b) we get the data for the **"optimal"** sLR-line:

needed for calculation of R^2				SST = SSE + SSR ?	
y(xi)	$SSE=sum(yi-y(xi))^2$	$SST=sum(yi-M(y))^2$	R^2	$SSR=sum(y(xi) - M(y))^2$	R^2
11/6	$(1/6)^2=1/36$	$(-1/3)^2=1/9$		1/4	
17/6	$(1/6)^2=1/36$	$(2/3)^2=4/9$		1/4	
14/6	$(-2/6)^2=4/36$	$(1/3)^2=1/9$		0	
42/6=7	6/36=1/6	2/3	0,7500000	1/2	0,7500000
				0,6666667	<--SSR + SSE

Part b:

Detailed description and Excel document with the integrated formulas for the calculation of the coefficients a, b can be found GitHub/Hvoellinger:

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

The excel name is "LR-Calculation of Coeff.xlsx":

Find "least square fit" *y = b0 + b1x* with {(x, y)}= {(1,2), (3, 3), (2, 2)}

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 b0 and b1 and also the quantities for the calculation of R^2:

	needed for calculation of b0 and b1				needed for calculation of R^2			SST = SSE + SSR ?
student i	*exam prep.* x_i	*points* y_i	x_i*y_i	x_i^2	$y(x_i)$	$SSE=sum(yi-y(x_i))^2$	$SST=sum(yi-M(y))^2$	$SSR=sum(y(xi) - M(y))^2$
1	1	2	2	1	11/6	$(1/6)^2=1/36$	$(-1/3)^2=1/9$	1/4
2	3	3	9	9	17/6	$(1/6)^2=1/36$	$(2/3)^2=4/9$	1/4
3	2	2	4	4	14/6	$(-2/6)^2=4/36$	$(1/3)^2=1/9$	0
sum	6	7	15	14	42/6=7	6/36=1/6	2/3	1/2

Substitute these values into Formula I and II:

b_0 = ((7/3)*14 -2*15)/(14-12) = (8/3)/2= 4/3

b_1 = (15 -3*2*(7/3))/2 = 1/2 = 0.5

----> Regression-Line: y = 4/3 + 1/2*x

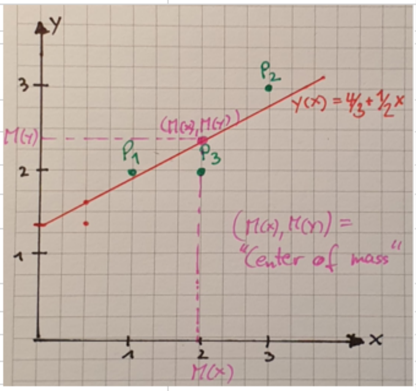
R^2 = 1 - Sum((yi-y(xi))^2)/Sum((yi-M(y))^2) = 1 - (1/6)/(2/3) = 1 -(1*3)/(6*2)=1-3/12= 1-1/4=3/4

Compare with Python

intercept: 1.3333333333333334

slope: [0.5]

coefficient of determination: 0.7499999999999999



Check of Proposition (P5.1): f(Mean(x)) = (4/3) + (1/2)*2 = 7/3 = Mean(y) q.e.d.

y=4/3 + 0.5*x is the Regression-Line. R^2 =3/4.

Part c:

Detailed description and code can be found in GitHub:
https://github.com/HVoellinger/Lecture-Notes-to-ML-WS2020
The Jupyter Notebook has the name "Homework-ML5_1c-LinReg.ipynb":

Homework-ML5_1c_LinReg

July 21, 2020

1 # Simple Linear Regression With scikit-learn (Example from lesson ML05)

Powered by: Dr. Hermann Völlinger, DHBW Stuttgart(Germany); July 2020

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)

The example is from Lecture: "ML_Concept&Algorithm" (WS2020); Homework ML5.1 with title: "Manual calculations of R^2 and find the optimal Regression-Line for a small example" + "Create a Jupyter Notebook (Python) to check the manual calculated results"

Let's start with the simplest case, which is simple linear 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.

2 Step 1: Import packages and classes

The first step is to import the package numpy and the class LinearRegression from sklearn.linear_model:

[3]:
```
# Step 1: Import packages and classes

import numpy as np
import sklearn as sk
from sklearn.linear_model import LinearRegression
```

3 Step 2: Provide data

The second step is defining data to work with. The inputs (regressors,) and output (predictor,) should be arrays (the instances of the class numpy.ndarray) or similar objects. This is the simplest way of providing data for regression:

[4]:
```
# Step 2: Provide data

x = np.array([ 1, 3, 2]).reshape((-1, 1))
y = np.array([ 2, 3, 2])
```

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.

[5]:
```
print ("This is how x and y look now:")
print("x=",x)
print("y=",y)
```

```
This is how x and y look now:
x= [[1]
 [3]
 [2]]
y= [2 3 2]
```

As you can see, x has two dimensions, and x.shape is (3, 1), while y has only a single dimension, and y.shape is (3,).

4 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:

[7]:
```
model = LinearRegression()
```

This statement creates the variable model as the instance of LinearRegression. You can provide several optional parameters to LinearRegression:

[8]:
```
model.fit(x, y)
```

[8]:
```
LinearRegression(copy_X=True, fit_intercept=True, n_jobs=None, normalize=False)
```

With .fit(), you calculate the optimal values of the weights and , using the existing input and output (x and y) as the arguments. In other words, .fit() fits the model. It returns self, which is the variable model itself. That's why you can replace the last two statements with this one:

[9]:
```
# model = LinearRegression().fit(x, y)
```

This statement does the same thing as the previous two. It's just shorter.

5 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 (²) with .score() called on model:

[13]:
```
r_sq = model.score(x, y)
print('coefficient of determination:', r_sq)
```

```
coefficient of determination: 0.7499999999999999
```

When you're applying .score(), the arguments are also the predictor x and regressor y, and the return value is ².

The attributes of model are .intercept_, which represents the coefficient, and .coef_, which represents :

[14]:
```
print('intercept:', model.intercept_)
print('slope:', model.coef_)
```

```
intercept: 1.3333333333333334
slope: [0.5]
```

6 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():

[16]:
```
y_pred = model.predict(x)
print('predicted response:', y_pred, sep='\n')
```

```
predicted response:
[1.83333333 2.83333333 2.33333333]
```

When applying .predict(), you pass the regressor as the argument and get the corresponding predicted response.

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:

Solutions:

Homework H5.3 – "Calculate Adj.R² for MR"

See also the YouTupe 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**)?

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

First Solution (H.Völlinger):

Part A:

1. Row: **Adj-R²** = 1-(1-R²)*(n-1/n-k-1) = 1-(0,29)*24/20 = 1-0,348 = **0,652**

……. Rest analogue……………

You get the final result:

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

Part B:

n=25: you get the best model for k=6 (Adj-R²=0.7067)

n=10: you get best the model for k=4 (Adj-R²=0.4780)

Second Solution (Lukas Petric, 8.4.2020):

Homework 4.2 - "Calculate Adj.R^2 for MR" Lukas Petrič

Part A: Calculate Adj.R^2 for given R^2 for a "Housing Price" example (see table below). Did you see a "trend"?

Task: Calculate Adj. R^2 with $R'^2 = 1 - (1-R^2) * (n-1/n-k-1)$

Number of observations, n	Number of variables, k	R^2	Adj. R^2
25	4	0,71	0,652
25	5	0,76	0,69684211
25	6	0,78	0,70666667
25	7	0,79	0,70352941
10	4	0,71	0,478
10	5	0,76	0,46
10	6	0,78	0,34
10	7	0,79	0,055

In order for Adj. R^2 to get higher, there is a certain threshold of k in relation to n that shouldn't be exceeded.

Part B: What would be the best model if n=25 and if n=10 (use Adj.R^2)?

For n=25 Adj. R^2 is highest for k=6, so n=25 and k=6 is the best model.
For n=10 Adj. R^2 is highest for k=4, so n=10 and k=4 is the best model.

Homework H5.4 - "mLR (k=2) manual calculations of Adj.R^2 & Jupyter Notebook (Python) to check results"

Part a: 1 Person, Part b +c: 1 Person

Consider the 4 points P1=(1|2|3), P2=(3|3|4), P3=(2|2|4) and P4=(4|3|6) in the 3-dimensional space:

Part a: Calculate the mLR-Measures Adj.R^2 for the two Hyperplanes H1:=plane defined by {P1,P2,P3} and H2:=Plane defined bx {P2,P3,P4}. Which plane (red or green) is a better mLR estimation? (Hint: calculate Adj.R^2).

Part b: What is the optimal Regression-Plane z = a + b*x + c*y. By using the formulas developed with "Least Square Fit for mLR" method for the coefficients a, b and c. What is Adj.R^2 for this plane? (Hint: a=17/4, b=3/2, c=-3/2; R^2 ~0.9474 and Adj.R^2=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.

First Solution: by Hermann Völlinger, 29.10.2020

Part a:

$H1: f(x,y) = z = 4 + x - y = \langle P1, P2, P3 \rangle$

$H2: f(x,y) = z = 4 + 2x - 2y = \langle P2, P3 | P4 \rangle$

$P_1 = (1|2|3);\ P_2 = (3|3|4);\ P_3 = (2|2|4),\ P_4 = (4|3|6)$

Berechne $R^2 = 1 - \frac{SSE}{SST}$ für beide Ebenen

$$SST = \sum_{i=1}^{4} (z_i - \bar{z})^2 = \left(3 - \tfrac{17}{4}\right)^2 + 2\cdot\left(4 - \tfrac{17}{4}\right)^2 + \left(6 - \tfrac{17}{4}\right)^2$$

$$= \left(\tfrac{5}{4}\right)^2 + 2\left(\tfrac{1}{4}\right)^2 + \left(\tfrac{7}{4}\right)^2 = \frac{25+2+49}{16} = \frac{76}{16} = \frac{19}{4}$$

$$SSE = \sum_{i=1}^{4} (f(x_i,y_i) - z_i)^2 \underset{\text{nur } P4 \notin \langle P1,P2,P3\rangle}{=} (f(x_4,y_4) - 6)^2 = (4+4-3-6)$$

$$= (-1)^2 = 1$$

$$SSE = \sum_{i=1}^{4} (f(x_i,y_i) - z_i)^2 \underset{P_1 \notin \langle P_2,P_3,P_4\rangle}{=} (f(x_1,y_1) - z_1)^2$$

$$= (4 + 2\cdot 1 - 2\cdot 2 - 3)^2 = (-1)^2 = 1$$

Daraus folgt: R^2 ist gleich für beide Ebenen.

$$R^2 = 1 - \frac{SSE}{SST} = 1 - \frac{4}{19} = \frac{15}{19}$$

$$\Rightarrow \bar{R}^2 = 1 - \left(\frac{4}{19}\right)\cdot\frac{3}{1} = \frac{19-12}{19} = \frac{7}{19}$$

Part c:

1.3 Step 4: Get results

You can obtain the properties of the model the same way as in the case of simple linear regression:

```
[4]: r_sq = model.score(x, y)
     print('coefficient of determination:', r_sq)
     print('intercept:', model.intercept_)
     print('coefficients:', model.coef_)
```

```
coefficient of determination: 0.9473684210526315
intercept: 4.25
coefficients: [ 1.5 -1.5]
```

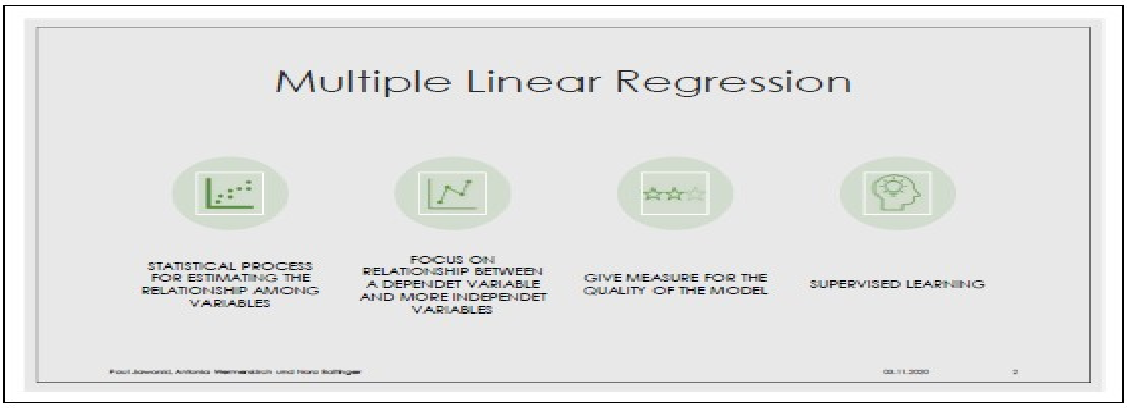
You obtain the value of ² using .score() and the values of the estimators of regression coefficients with .intercept_ and .coef_. Again, .intercept_ holds the bias , while now .coef_ is an array containing and respectively.

In this example, the intercept is approximately 4.25, and this is the value of the predicted response when = = 0. The increase of by 1 yields the rise of the predicted response by 1.5. Similarly, when grows by 1, the response declined by -1.5.

Adj.R^2 := 1 – (1 - R^2) * (3/1) = 1 - (1 - 0,94736)*3 ~ 0,84208

Second Solution: by A. Wermerskirch, N. Baitinger und P. Jaworski, 2.11.2020

Part a+b:



Formulas

Value	Formula
Regression plane	$z = a + b \cdot x + c \cdot y$
det	$\det = \sum (x_i - \bar{x})^2 \sum (y_i - \bar{y})^2 - \left(\sum (x_i - \bar{x}) \cdot (y_i - \bar{y}) \right)^2$
a	$a = \bar{z} - b\bar{x} - c\bar{y}$
b	$b = \frac{\Sigma (y_i - \bar{y})^2 \cdot \Sigma (x_i - \bar{x}) \cdot (z_i - \bar{z}) \; - \; \Sigma (x_i - \bar{x}) \cdot (y_i - \bar{y}) \cdot \Sigma (y_i - \bar{y}) \cdot (z_i - \bar{z})}{det}$
c	$c = \frac{\Sigma (x_i - \bar{x})^2 \cdot \Sigma (y_i - \bar{y}) \cdot (z_i - \bar{z}) \; - \; \Sigma (x_i - \bar{x}) \cdot (y_i - \bar{y}) \cdot \Sigma (x_i - \bar{x}) \cdot (z_i - \bar{z})}{det}$

Formulas

Value	Abbreviation	Formular	Meaning
Sum of Squares Totals	SST	$\sum_{i=1}^{n} (y_i - \bar{y})^2$	Total deviation from the mean value
Sum of Squares Errors	SSE	$\sum_{i=1}^{n} (y_i - \bar{f})^2$	Unexplained deviation from the mean value
Sum of Squares Regression	SSR	$\sum_{i=1}^{n} (f_i - \bar{y})^2$	Explained deviation from the mean value
R-Squared	R^2	$1 - \frac{SSE}{SST}$	The closer the value of R^2 is to 1 the better the regression fits the data

Formulas

Value	Abbreviation	Formular	Meaning
Number of oberservations	n		measured points, number of training set points
Number of variables	k		several independent variables (k > 1) R² must be adjusted
Degrees of freedom	df	$df = n - k - 1$	e.g. df=1; n=4, k=2
Adjusted R-squared	Adj. R²	$1-(1-R^2)\frac{n-1}{n-k-1}$ or $1-\left(\frac{SSE}{SST}\right)\frac{n-1}{n-k-1}$	how well observed outcomes are replicated by the model

Homework H5.4

- Consider the 4 points P1= (1 | 2 | 3), P2=(3 | 3 | 4), P3=(2 | 2 | 4) and P4=(4 | 3 | 6) in the 3-dimensional space:
- Part a: Calculate the sLR Measures Adj.R² for the two Hyperplanes H1:=plane defined by {P1, P2, P3} and H2:=Plane defined by {P2, P3, P4}. Which plane (H1 or H2) 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?
- 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: Adj.R²

- Calculate the sLR Measures Adj.R² for the two Hyperplanes H1:=plane defined by {P1, P2, P3} and H2:=Plane defined by {P2, P3, P4}. Which plane (H1 or H2) is a better mLR estimation?
- P1= (1 | 2 | 3), P2=(3 | 3 | 4), P3=(2 | 2 | 4) and P4=(4 | 3 | 6)
- Step 1: H1 and H2 planes
  H1: $z = 4 + x - y$ and H2: $z = 4 + 2x - 2y$
- Step 2: Mean z
  $M(z) = \frac{3+4+4+6}{4} = \frac{17}{4} = 4{,}25$
- Step 3: Calculate $z(x_i, y_i)$, SSE and SST for H1 and H2
- Step 4: Calculate R² and Adj. R²

Part a: Adj.R²

H1

$z(x_i,y_i)$	$SSE=\sum(z_i-z(x_i,y_i))^2$	$SST=\sum(z_i-M(z))^2$
3	0	1,5625
4	0	0,0625
4	0	0,0625
5	1	3,0625
	1	**4,75**

H2

$z(x_i,y_i)$	$SSE=\sum(z_i-z(x_i,y_i))^2$	$SST=\sum(z_i-M(z))^2$
4	0	0,0625
4	0	0,0625
6	0	3,0625
2	1	1,5625
	1	**4,75**

$$R^2 = 1 - \frac{SSE}{SST} = \frac{1}{4{,}75} = 0{,}7895$$

$$Adj.R^2 = 1-(1-R^2)\frac{n-1}{n-k-1} = 1-(1-0{,}7895)\frac{4-1}{4-2-1} = \frac{7}{19} \approx 0{,}3684$$

no desicion about a better plane possible

Part b: Optimal Regression Plane

- 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?
- P1= (1 | 2 | 3), P2=(3 | 3 | 4), P3=(2 | 2 | 4) and P4=(4 | 3 | 6) → n = 4
- Step 1: Mean-Values

  $M(x) = \frac{1+3+2+4}{4} = \frac{10}{4} = 2{,}5$ $M(y) = \frac{2+3+2+3}{4} = \frac{10}{4} = 2{,}5$ $M(z) = \frac{3+4+4+6}{4} = \frac{17}{4} = 4{,}25$
- Step 2: Calculate X_i, Y_i and Zi

  $X_i = x_i - M(x)$ $Y_i = y_i - M(y)$ $Z_i = z_i - M(z)$
- Step 3: Calculate det = $\sum(X_i)^2 \cdot \sum(Y_i)^2 - (\sum X_i \cdot Y_i)^2$
- Step 4: Calculate a, b and c to get the optimal mLR line $z = a + b \cdot x + c \cdot y$
- Step 5: Calculate R² and Adj. R²

Part b: Optimal Regression Plane

Step 2: Calculate Xi, Yi and Zi

	needed for the calculation of a, b and c										
i	xi	yi	zi	Xi=xi-M(x)	Yi=yi-M(y)	Zi=zi-M(z)	Xi*Yi	Xi*Zi	Yi*Zi	Xi²	Yi²
1	1	2	3	-1,5	-0,5	-1,25	0,75	1,875	0,625	2,25	0,25
2	3	3	4	0,5	0,5	-0,25	0,25	-0,125	-0,125	0,25	0,25
3	2	2	4	-0,5	-0,5	-0,25	0,25	0,125	0,125	0,25	0,25
4	4	3	6	1,5	0,5	1,75	0,75	2,625	0,875	2,25	0,25
sum	10	10	17	0	0	0	2	4,5	1,5	5	1

Step 3: Calculate det = $\sum(Xi)^2 \cdot \sum(Yi)^2 - (\sum Xi \cdot Yi)^2$

$det = 5 * 1 - (2)^2 = 5 - 4 = 1$

Part b: Optimal Regression Plane

Step 4: Calculate a, b and c to get the optimal mLR line $z = a + b \cdot x + c \cdot y$

$$b = \frac{1}{det} \cdot (sum(Yi^2) \cdot sum(Xi \cdot Zi) - sum(Xi \cdot Yi) \cdot sum(Yi \cdot Zi)) = \frac{1}{1} \cdot (1 \cdot 4{,}5 - 2 \cdot 1{,}5) = 1{,}5$$

$$c = \frac{1}{det} \cdot (sum(Xi^2) \cdot sum(Yi \cdot Zi) - sum(Xi \cdot Yi) \cdot sum(Xi \cdot Zi)) = \frac{1}{1} \cdot (5 \cdot 1{,}5 - 2 \cdot 4{,}5) = -1{,}5$$

$$a = M(z) - b \cdot M(x) - c \cdot M(y) = 4{,}25 - 1{,}5 \cdot 2{,}5 - (-1{,}5) \cdot 2{,}5 = 4{,}25$$

So we get the optimal mLR line: $z = 4{,}25 + 1{,}5x - 1{,}5y$

Part b: Optimal Regression Plane

Step 5: Calculate R² and Adj. R²

needed for calculation of R²			SST= SSE + SSR
z(xi,yi)	SSE=sum(zi-z(xi,yi))²	SST=sum(zi-M(z))²	SSR=sum(z(xi,yi)-M(z))²
2,75	0,0625	1,5625	2,25
4,25	0,0625	0,0625	0
4,25	0,0625	0,0625	0
5,75	0,0625	3,0625	2,25
17	0,25	4,75	4,5
		SSE+SSR=	4,75

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

$$R^2 = 1 - \frac{0{,}25}{4{,}75} \approx 0{,}94736 \approx 0{,}9474$$

$$Adjusted\ R^2 = 1 - (1 - R^2) \cdot \left(\frac{n-1}{n-k-1}\right)$$

$$Adj.R^2 = 1 - (1 - 0{,}94736) \cdot \left(\frac{3}{1}\right) = 0{,}84208 \approx 0{,}8421$$

Part c:

multiple Linear Regression (mLR) with scikit-learn

Provided by Nora Baitinger, Antonia Wermerskirch, Paul Jaworski

Location: DHBW Stuttgart, Date: 2.11.2020

Extented by H. Völlinger; DHBW; 2.11.2020

The implementation of mLR is very similar to that of sLR:

1. Import all needed packages
2. Provide data to work with
3. Create and fit regression model with data from previous step
4. Check the fitted model for statisfaction
5. Apply model for predicitions

Step 1: Import all needed dependencies

numpy - uses numerical mathematics

IPython - uses display, Math and Latex to for printing the formula

sklearn - Use/call the LinearRegression module

sys – version information to pythonImport of libraries

Rest see **[HVö-6]:** Dr. Hermann Völlinger: GitHub to the Lecture "Machine Learning: Concepts & Algorithms“; see: https://github.com/HVoellinger/Lecture-Notes-to-ML-WS2020

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[(fi-yi)*(fi-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-points*}; a sLR-line which has condition (*), but is not an optimal sLR-line.

Exercises to Lesson ML6: Convolutional Neural Networks (CNN)

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"
https://www.enercast.de/wp-content/uploads/2018/04/whitepaper-prognosen-wind-solar-kuenstliche-intelligenz-neuronale-netze_110418_EN.pdf

Solutions:
…..

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".
https://youtu.be/vSWLafBdHus

Solutions:
……

Homework H6.3* – "Create Summary to GO Article"

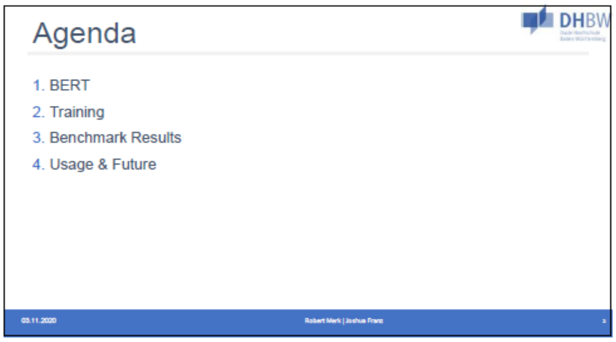
Groupwork (2 Persons) - read and create a summary of the main results of the article "Mastering the game of Go with deep neural networks and tree search"
https://storage.googleapis.com/deepmind-media/alphago/AlphaGoNaturePaper.pdf

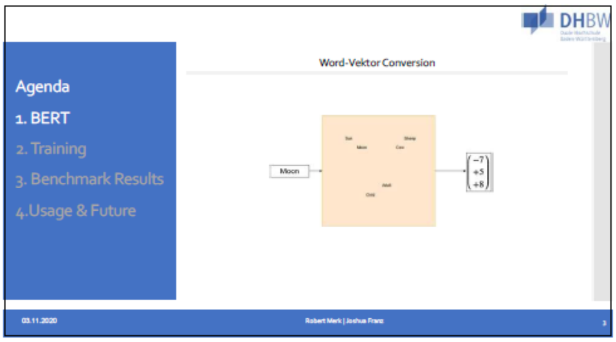
Solutions:
…..

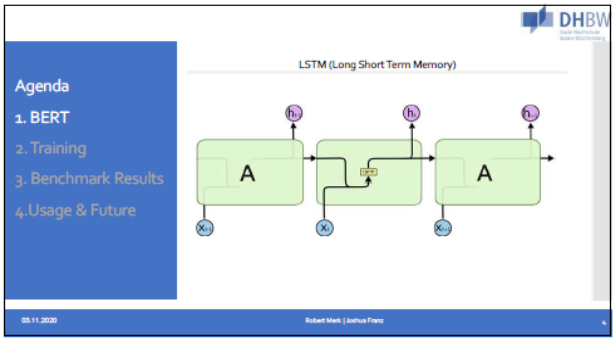
Homework H6.4* – "Create Summary to BERT Article"

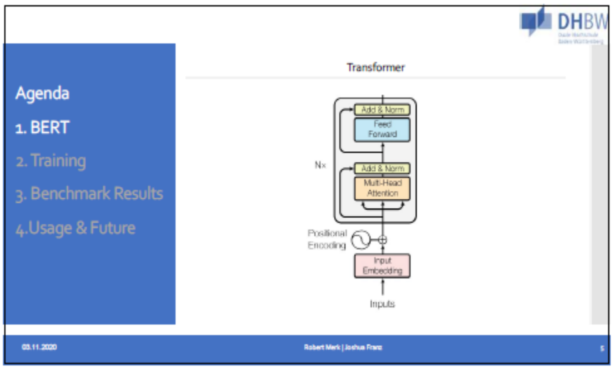
Groupwork (2 Persons): Read and summaries of the main results of the article about BERT. See Ref. [BERT]: Jacob Devlin and Other: "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding"; Google (USA); 2019

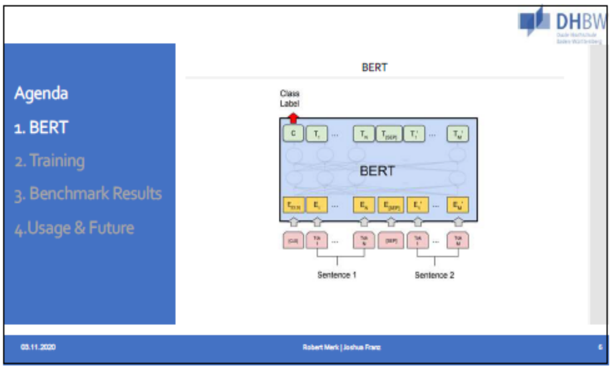
Solutions: by Robert Merk unn Joshua Franz; 3.11.2020

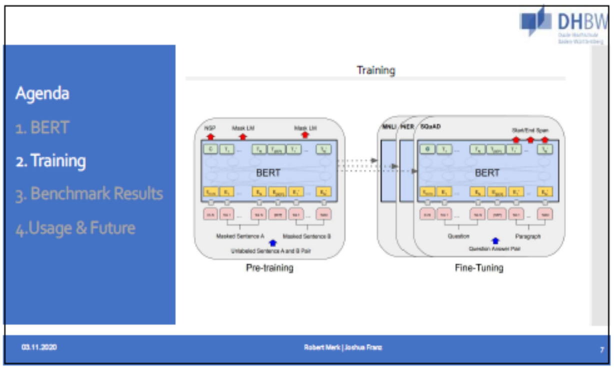


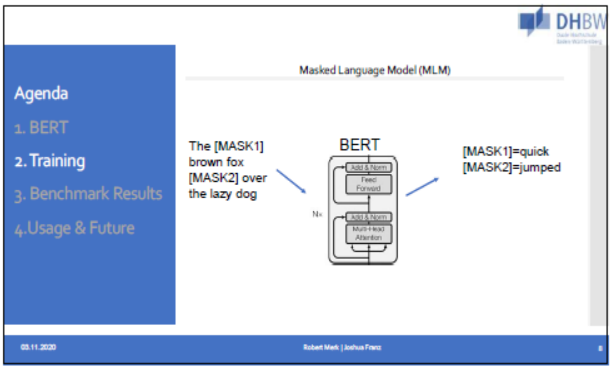


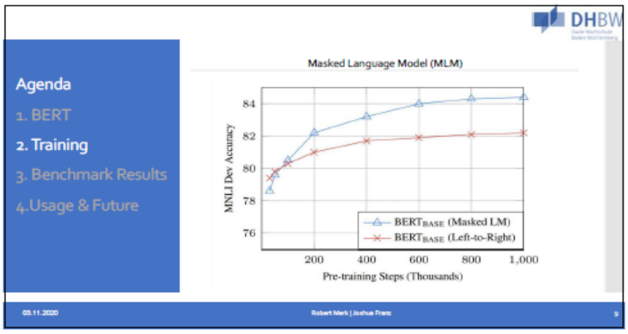


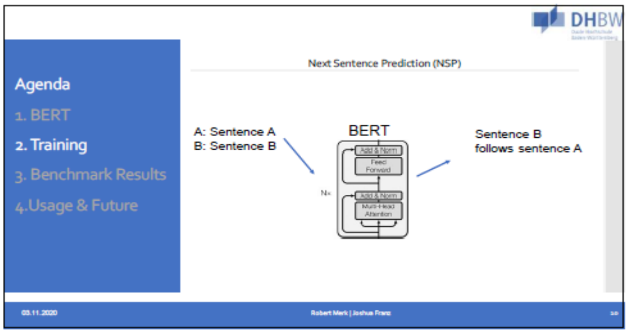

DHBW
Agenda
1. BERT
2. Training
3. Benchmark Results
4. Usage & Future
Transformer
Add & Norm
Feed Forward
Add & Norm
Multi-Head Attention
Nx
Positional Encoding
Input Embedding
Inputs
03.11.2020
Robert Merk | Joshua Franz

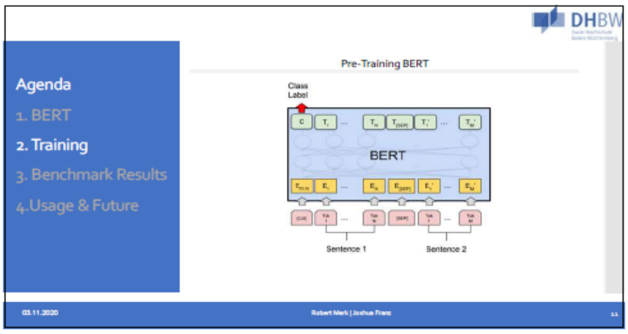

DHBW
Agenda
1. BERT
2. Training
3. Benchmark Results
4. Usage & Future
BERT
Class Label
BERT
Sentence 1
Sentence 2
03.11.2020
Robert Merk | Joshua Franz


DHBW
Agenda
1. BERT
2. Training
3. Benchmark Results
4. Usage & Future
Training
BERT
BERT
Pre-training
Fine-Tuning
03.11.2020
Robert Merk | Joshua Franz

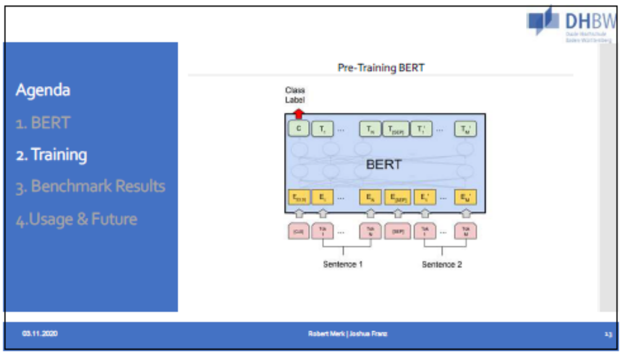

DHBW
Agenda
1. BERT
2. Training
3. Benchmark Results
4. Usage & Future
Masked Language Model (MLM)
The [MASK1] brown fox [MASK2] over the lazy dog
BERT
[MASK1]=quick
[MASK2]=jumped
03.11.2020
Robert Merk | Joshua Franz

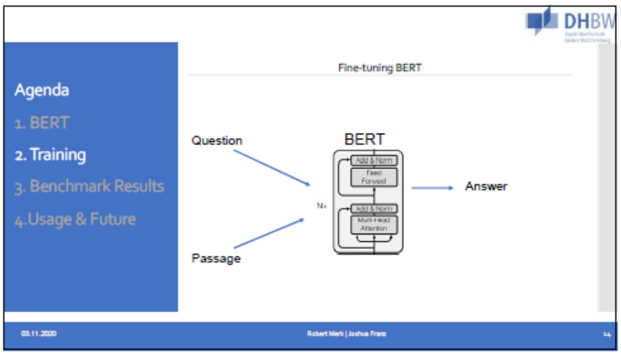

DHBW
Agenda
1. BERT
2. Training
3. Benchmark Results
4. Usage & Future
Masked Language Model (MLM)
MNLI Dev Accuracy
84
82
80
78
76
200
400
600
800
1,000
Pre-training Steps (Thousands)
BERTBASE (Masked LM)
BERTBASE (Left-to-Right)
03.11.2020
Robert Merk | Joshua Franz

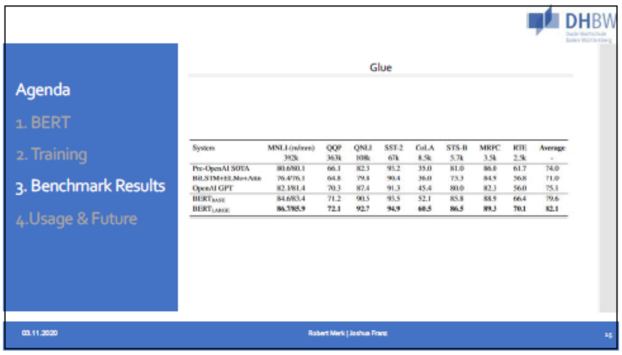

DHBW
Agenda
1. BERT
2. Training
3. Benchmark Results
4. Usage & Future
Next Sentence Prediction (NSP)
A: Sentence A
B: Sentence B
BERT
Sentence B follows sentence A
03.11.2020
Robert Merk | Joshua Franz

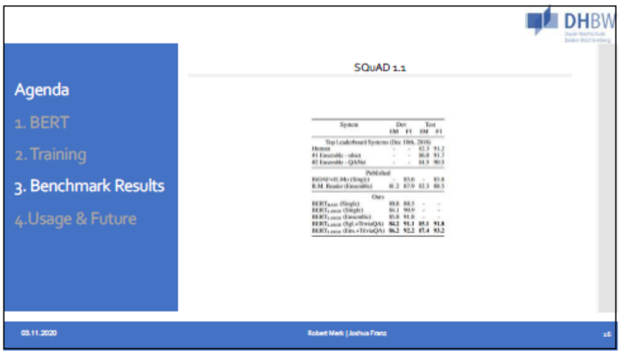

DHBW
Agenda
1. BERT
2. Training
3. Benchmark Results
4. Usage & Future
Pre-Training BERT
Class Label
BERT
Sentence 1
Sentence 2
03.11.2020
Robert Merk | Joshua Franz

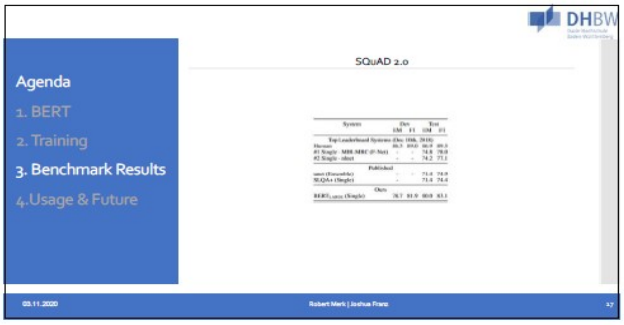

DHBW
Agenda
1. BERT
2. Training
3. Benchmark Results
4. Usage & Future
Pre-Training BERT
Input
Token Embeddings
Segment Embeddings
Position Embeddings
03.11.2020
Robert Merk | Joshua Franz


DHBW
Agenda
1. BERT
2. Training
3. Benchmark Results
4. Usage & Future
Pre-Training BERT
Class Label
BERT
Sentence 1
Sentence 2
03.11.2020
Robert Merk | Joshua Franz

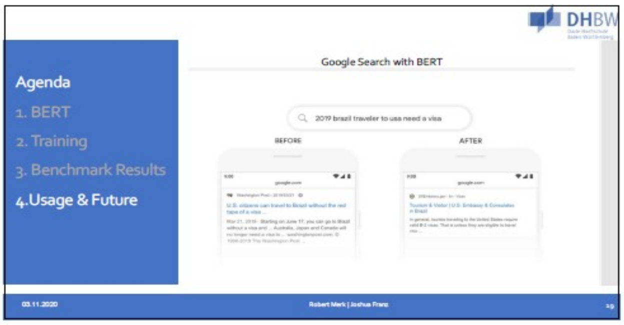

DHBW
Agenda
1. BERT
2. Training
3. Benchmark Results
4. Usage & Future
Fine-tuning BERT
Question
BERT
Answer
Passage
03.11.2020
Robert Merk | Joshua Franz


DHBW
Agenda
1. BERT
2. Training
3. Benchmark Results
4. Usage & Future
Glue
03.11.2020
Robert Merk | Joshua Franz


DHBW
Agenda
1. BERT
2. Training
3. Benchmark Results
4. Usage & Future
SQuAD 1.1
03.11.2020
Robert Merk | Joshua Franz


DHBW
Agenda
1. BERT
2. Training
3. Benchmark Results
4. Usage & Future
SQuAD 2.0
03.11.2020
Robert Merk | Joshua Franz


DHBW
Agenda
1. BERT
2. Training
3. Benchmark Results
4. Usage & Future
Alibaba.com
Google
Bai du 百度
Microsoft 365
HUAWEI
03.11.2020
Robert Merk | Joshua Franz


DHBW
Agenda
1. BERT
2. Training
3. Benchmark Results
4. Usage & Future
Google Search with BERT
2019 brazil traveler to usa need a visa
BEFORE
AFTER
03.11.2020
Robert Merk | Joshua Franz

Exercises to Lesson ML7: BackPropagation for Neural Networks

Homework H7.1 – "Exercise of an Example with Python"

*********** placeholder*******************

Solutions:

….

Homework H7.2 – "Exercise of an Example with Python"

*********** placeholder*******************

Solutions:

….

Exercises to Lesson ML8: Support Vector Machines (SVM)

Homework H8.1 – "Exercise of an Example with Python"

*********** placeholder******************

Solutions:

....

Homework H8.2 – "Exercise of an Example with Python"

*********** placeholder******************

Solutions:

....

Homework H8.3 – "Exercise of an Example with Python"

*********** placeholder******************

Solutions:

....

Homework H8.4 – "Exercise of an Example with Python"

*********** placeholder******************

Solutions:

....