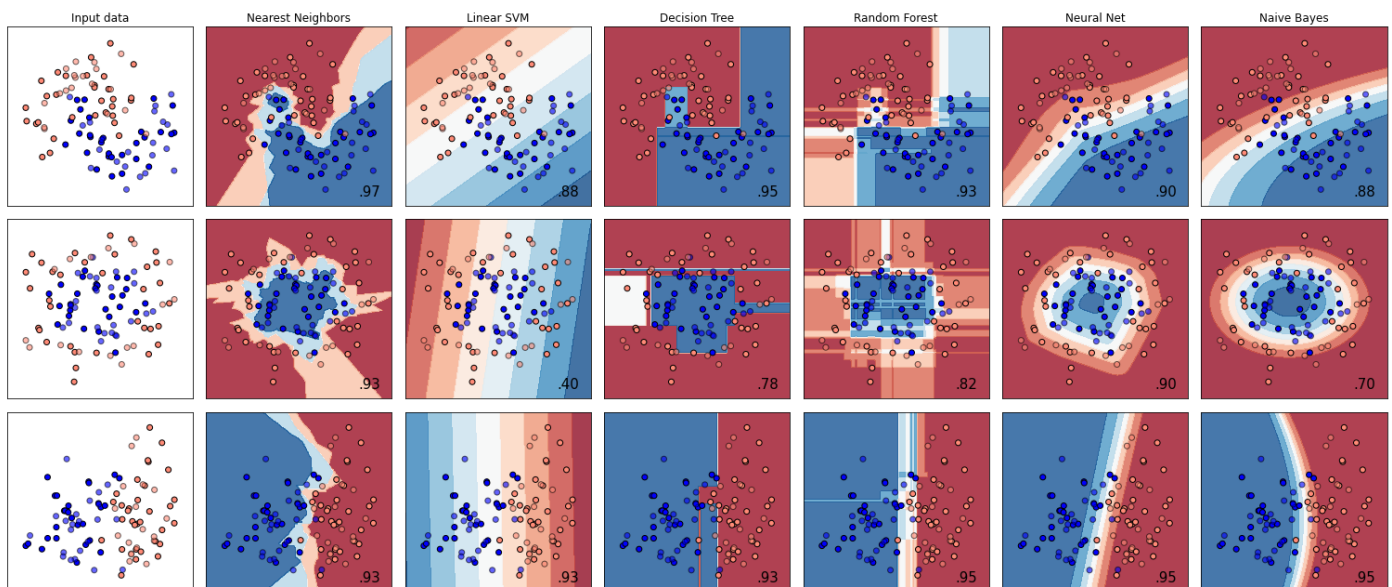


Introduction and overview of artificial intelligence

By Thomas Jörg, thomas@iludis.de



1: In this table you see the classification results of different ML-classifiers used in this document. On the left the original dataset is shown. Derived from https://scikit-learn.org/stable/auto_examples/classification/plot_classifier_comparison.html

A collection of lessons for introducing the basic concepts of artificial intelligence / machine learning in school classes.

Target audience: between 14 and 18 years.

Table of contents

Table of contents	2
Lesson 1: Why should young students study the basic concepts of A.I.?	3
Lesson 2: ML/AI/DL What's behind it all?	4
Lesson 3: Classification & prediction, what about data privacy protection?	6
Lesson 4: ML/AI/DL: What are the similarities and differences?	7
Lesson 5: What is the difference between traditional code and artificial intelligence?	8
Lesson 6: NLP and the GPT-3 on "What is machine learning"?	14
Lesson 7: Teachable Machine and Scratch	16
Lesson 8: ML principles unplugged	20
Some useful resources for the A.I. lessons	24
Lesson 9: Decision Trees	26
Lesson 10: Naive Bayes Classifier	33
Lesson 11: Linear Regression	43
Lesson 12: Distinguish correlation and causation!	50
Lesson 13: k-Means Clustering.	51
Lesson 14: k-nearest neighbour	58
Lesson 15: Support Vector Machines	65
Lesson 16: Orange Data Mining Project: Price prediction of a used car	68
Lesson 17: Orange Data Mining Project, how to survive the titanic?	75
Lesson 18: Summary of the complete lesson	79



Photo DALL-E2

Lesson 1: Why should young students study the basic concepts of A.I.?

Some introductory thoughts for teachers

Artificial intelligence is becoming a technology that is fundamentally changing all of our lives right now since it occupies the core of human self-understanding: The ability to think and evolve as a result. In reality, AI is driving the so-called fourth/fifth industrial revolution (counting depending on how you interpret it). Here, things, processes and machines network and communicate, learning to make decisions independently. In the European industrialized nations, knowledge about AI is thus becoming an existential social issue that affects the lives of every citizen.

Electronic sensors that are everywhere - from temperature sensors in living rooms to RFID sensors in the manufacturing industry to blood pressure sensors in smartwatches - provide continuous streams of data about us and our environment. This data is the 'crude oil' of AI, because AI systems are data-driven: An AI learns from hidden patterns it discovers in the data, captures their meaning, and independently plans resulting actions. Sometimes with, sometimes without human intervention. In this way, what was previously "big data" becomes "smart data."

Robotic systems clean our homes, automobiles learn autonomous driving, medical robots perform surgical operations. Novel expert systems invent new medicines, write journalistic texts and design the clothing fashions we put on - and even acquire patents they have developed themselves.

The younger generation will grow up with these autonomous, intelligent systems. In order not to be at the mercy of these "black box" systems as a simple user, every child should learn as early as possible on which working principles artificial intelligence is based and how to design, use and evaluate them independently. Knowledge of how to deal with AI is thus becoming one of the core competencies of the 21st century. Relevant questions that can be formulated as teaching objectives include, for example:

- How does a machine learn?
- What is the significance of data? And where does the data come from?
- What are the limits of machine learning?
- How can I influence it myself?
- How can I protect myself from it under certain circumstances? Where is this necessary?
- What significance will I have as a human being in a world where machines think?

The first thing a young person should understand: Machine learning is multi-faceted. From a helicopter perspective, it is a process that begins with data ingestion. The collected data is processed in an iterative learning cycle, followed by an assessment of learning success. Once this cycle is complete, the next step is the application of the learned content. The 'heart' of artificial intelligence is the data-processing algorithm used in each case, which is selected and adapted according to the problem.

This collection of AI algorithms behaves much like a large toolbox: here are many different tools that 'fit' a particular task. Both the process and the tools should be taught in the classroom. The goal is for a young person to understand the nature of AI by doing and experiencing it themselves.

And this path starts with classifying and structuring the most important terms and principles of action. And this is where we start in the first chapter.

Lesson 2: ML/AI/DL What's behind it all?

What students should learn in this lesson:

A.I. is all around us, we find it – sometimes invisible, hidden – in many places and applications. It's about predicting things, forecasting, pattern recognition and rule-based synthesis of new unknown things (Style-GANs or Transformer Models). Students should understand the relevance of A.I. in their life.



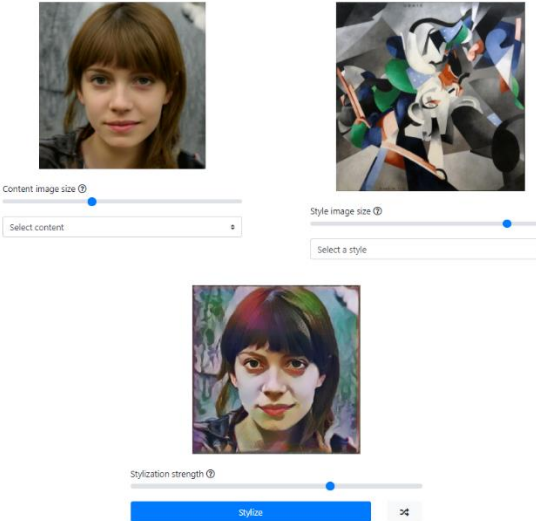
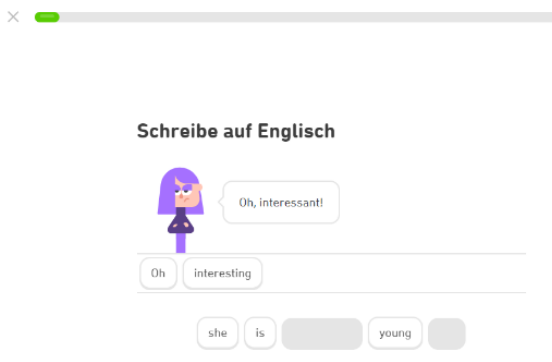
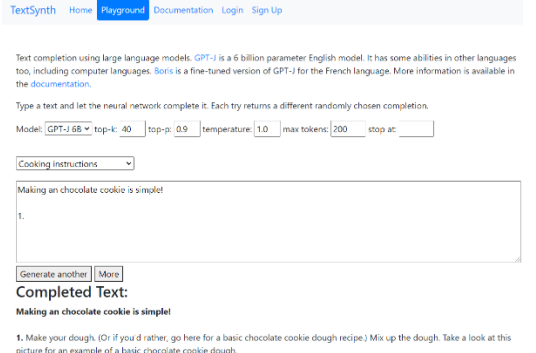
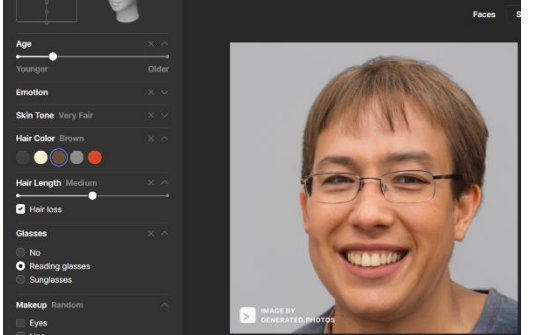
What if I told you these people never existed?
Visualisierung StyleGAN2 / Karras et al. and Nvidia / CC BY-SA 4.0
Visualisierung Owlsmcgee / CC BY-SA 4.0

Possible students' activities and tasks:

The following AI examples on these web pages can be used for short introductory presentations. A group of students can be divided into smaller groups of about two to three students for this purpose. Each group receives one of these websites on the topic. Possible questions include:

1. briefly introduce what the purpose of this AI is.
2. How does this AI work, or what technologies are used here?
3. what problems can it be used to solve?
4. what human jobs could it be used to replace?
5. what are the possibilities and dangers?

Subject	URL	Screenshot
Demo of handwritten digits Recognition	https://www.snaplogic.com/machine-learning-showcase/handwritten-digit-recognition	
Prediction of drawings	https://quickdraw.withgoogle.com/#	

Style Transfer	https://reiinakano.com/arbitrary-image-stylization-tfjs/	
Online language Training (GPT-3)	https://www.duolingo.com/learn	
Text synthesis (GPT-2)	https://textsynth.com/playground.html	
GAN: Synthesis of human faces	https://www.whichfaceisreal.com/generated.photos/	
Text recognition and translation.	https://deepl.com/translator	

Lesson 3: Classification & prediction, what about data privacy protection?



What students should learn

„ Data is collected from me as an individual through many different channels: Via social networks, by means of movements with my smartphone, and even movement profiles within supermarkets. All this data is used by trained artificial intelligences to create personality profiles of me. These make it possible to make my thinking and future trading transparent to outside institutions. And often others know about my habits that I am not aware of myself. This is where you have to learn to protect yourself.

Possible students' activities and tasks:

A.I. can predict your behavior based on your (sometimes unconscious) habits. And A.I. is often invisible, acting in the background – therefore it's possible you are classified and you don't know anything about it.

Discuss the following three examples. First read through the newspaper/ blog articles and summarize:

1. What is the role of artificial intelligence here?
2. Are the people involved aware of it?
3. Is the prediction that the AI makes for the benefit of the person concerned?
4. How would you feel if an AI made predictions about your character or future behavior?
5. What options do I have to influence it?

„How Target Figured Out A Teen Girl Was Pregnant Before Her Father Did“

<https://www.forbes.com/sites/kashmirhill/2012/02/16/how-target-figured-out-a-teen-girl-was-pregnant-before-her-father-did/?sh=226677fa6668>

Gaggle: Realtime-surveillance of high school students by an artificial intelligence in the USA

<https://www.gaggle.net/>

<https://www.the74million.org/article/gaggle-surveillance-minneapolis-families-not-smart-ai-monitoring/>

<https://www.buzzfeednews.com/article/carolinehaskins1/gaggle-school-surveillance-technology-education>

Facebook: The formation of love – prediction of personal relationships between facebook users.

https://m.facebook.com/nt/screen/?params=%7B%22note_id%22%3A10158928005273415%7D&path=%2Fnotes%2Fnote%2F&refsrc=deprecated&_rdr

<https://www.facebook.com/data/posts/10152217010993415>

Lesson 4: ML/AI/DL: What are the similarities and differences?

What students should learn

„Fundamentally, machine learning is a technique for using computers to predict things based on past observations. AI technology (algorithms) give computers the ability to learn without being explicitly programmed. It is a different and independent approach for utilizing computers compared to traditional programming.“

Possible Students activities and tasks

Do you think it is artificial intelligence?	Why do you think so?	what data is needed to make it work?
Image and facial recognition		
E-mail spam filtering		
Analysis for risk of heart disease		
Classification of used cars prices		
Prediction of energy need next winter		
Calculation of failure probability of my car		

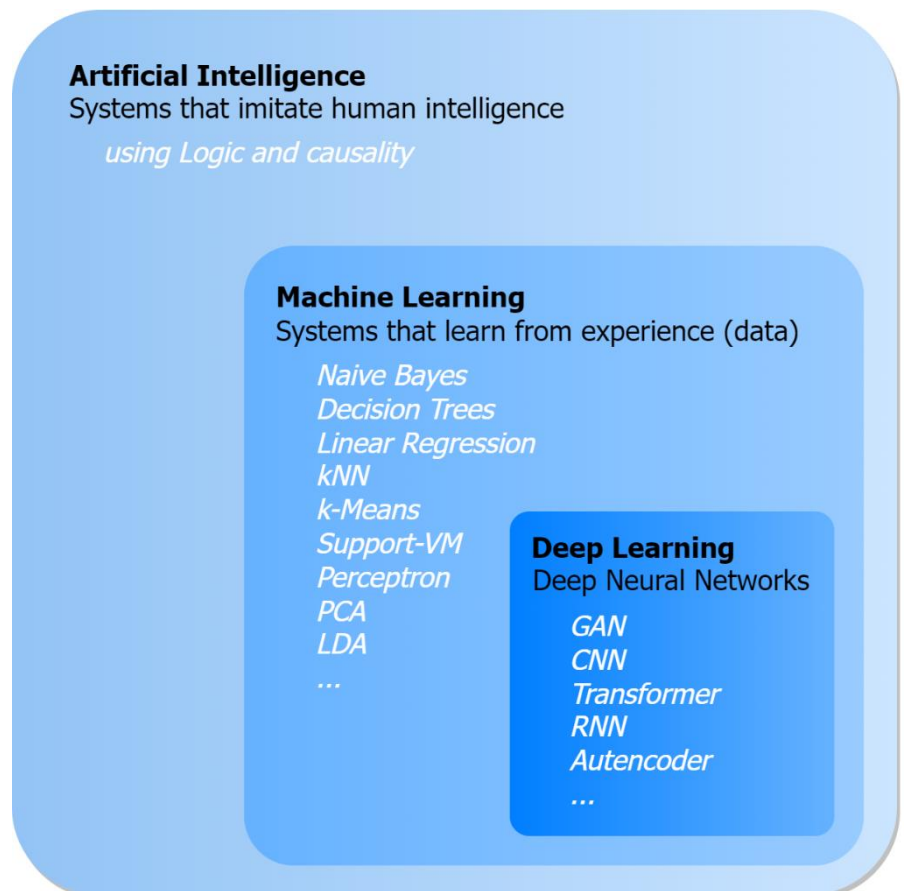
Part I: What's the difference between A.I, machine learning and deep learning?

Artificial intelligence ist the umbrella term for all kinds of machine behaviour that imitates human intelligence.

It can be interpreted as all methods of machine learning adding a layer of logical conclusions.

Machine learning consists of methods (called algorithms) that learn from data. Therefore, data has to be collected and prepared. ML is analysing the data to find patterns and rules.

Deep learning usually is the area of neural networks.



Lesson 5: What is the difference between traditional code and artificial intelligence?

What students should learn

How does computer programming using programming languages (such as Java, Python, or Scratch) differ from machine learning applications? What are the limitations of traditional imperative programming?

Possible students' activities and tasks:

Students should describe to a computer what a cat is.

It has pointed ears, it walks on four paws, it meows when it is hungry and it catches mice. In classical programming languages, attributes and methods are used to describe objects:

- Cat has four paws, two eyes, pointed ears.
- Cat can tap, run, sniff.

To recognize real-world cats, this is not enough, because dogs or can also have most of these properties. Using the well-known Kaggle data collection "Cats and Dogs," students are asked to work out how these two types of animals might be distinguished. The question will likely be difficult to answer.

In artificial intelligence, a computer learns from experience: the (slightly different) characteristics of many, many cats are collected, and the computer evaluates them.

Part I: Brainstorming, describe to a computer what a cat is

Cats and Dogs Dataset:

<https://www.kaggle.com/datasets/tongpython/cat-and-dog>



You are supposed to explain to a computer how to recognize a cat and a dog.

Collect some keywords that describe both types of animals.



the teacher's task here is not to allow generalities, but to pay attention to the peculiarities and limitations of computer programming languages. A thing must be described by properties, not by approximations.

Permissible is a statement such as:

"A cat has fur."

Not permissible are statements like:

"A dog sometimes has to go out in the evening" or "A cat often can't stand dogs".

How can you distinguish these?

Following your descriptions: Can the computer now safely recognize dogs and cats?
And furthermore also distinguish between them?

Collect and formulate problems that a computer might have in recognizing and distinguishing.



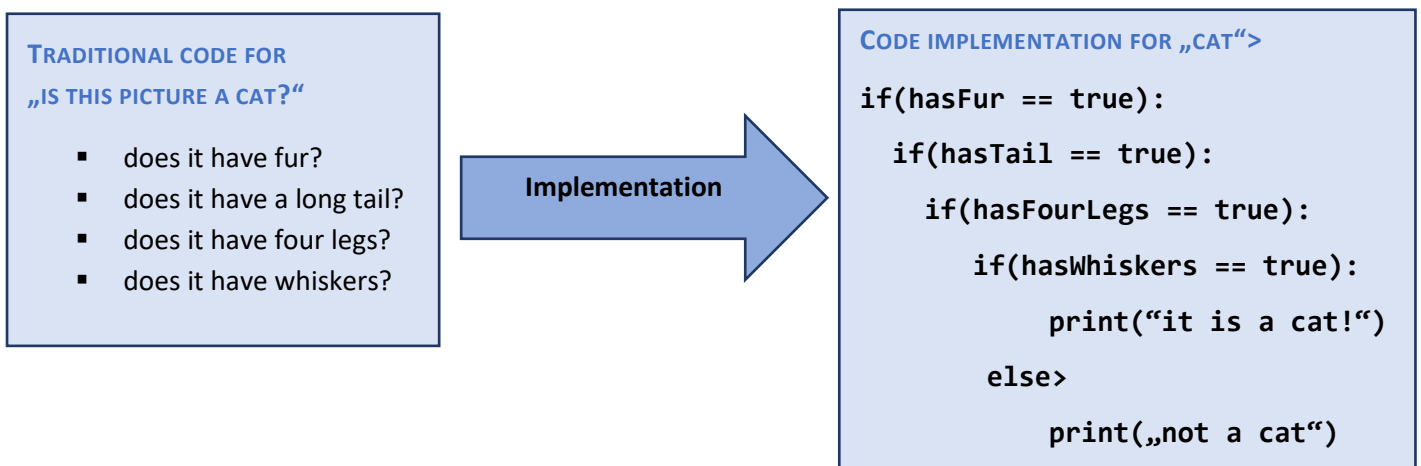
Part II: Analyse the results of the brainstorming session. An explanatory approach

[inspired from “Techgirlz: How computers learn”, <https://www.techgirlz.org/>]



DALL-E2

Let's assume a cat. If we want a computer to identify a cat and distinguish it, for example from a dog or a bear, we need to implement the rules – if we use traditional computer programming

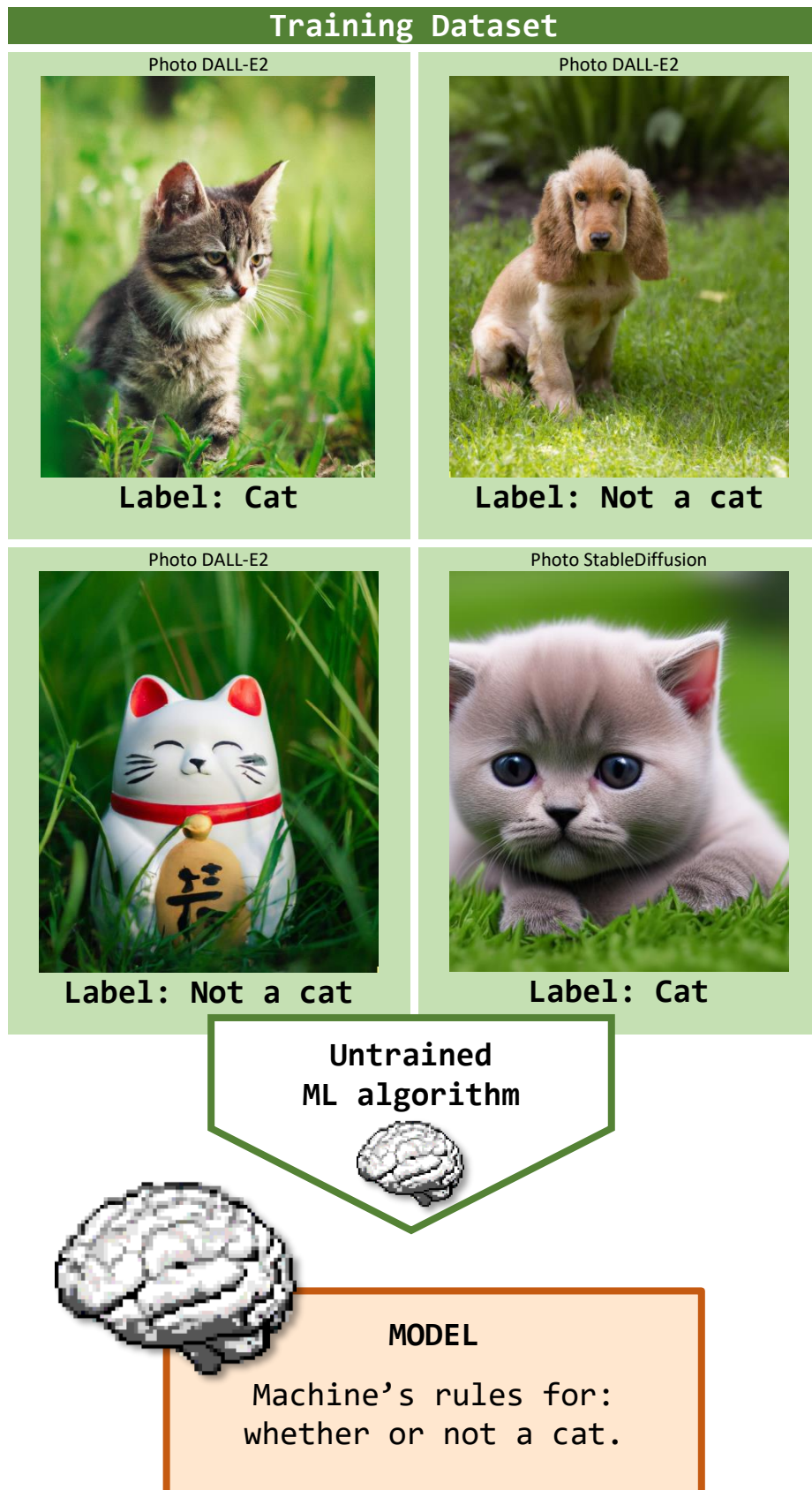


Classical programming would perform classification by exact comparison with tightly specified parameters. In the above example, Boolean operations - i.e. decisions with the two possibilities yes or no - are used. Does the object to be classified have certain sharply defined properties or not? The results are therefore equally sharp, exact, and usually much too narrowly defined. Recognizing an arbitrary cat in an arbitrary perspective cannot work this way.

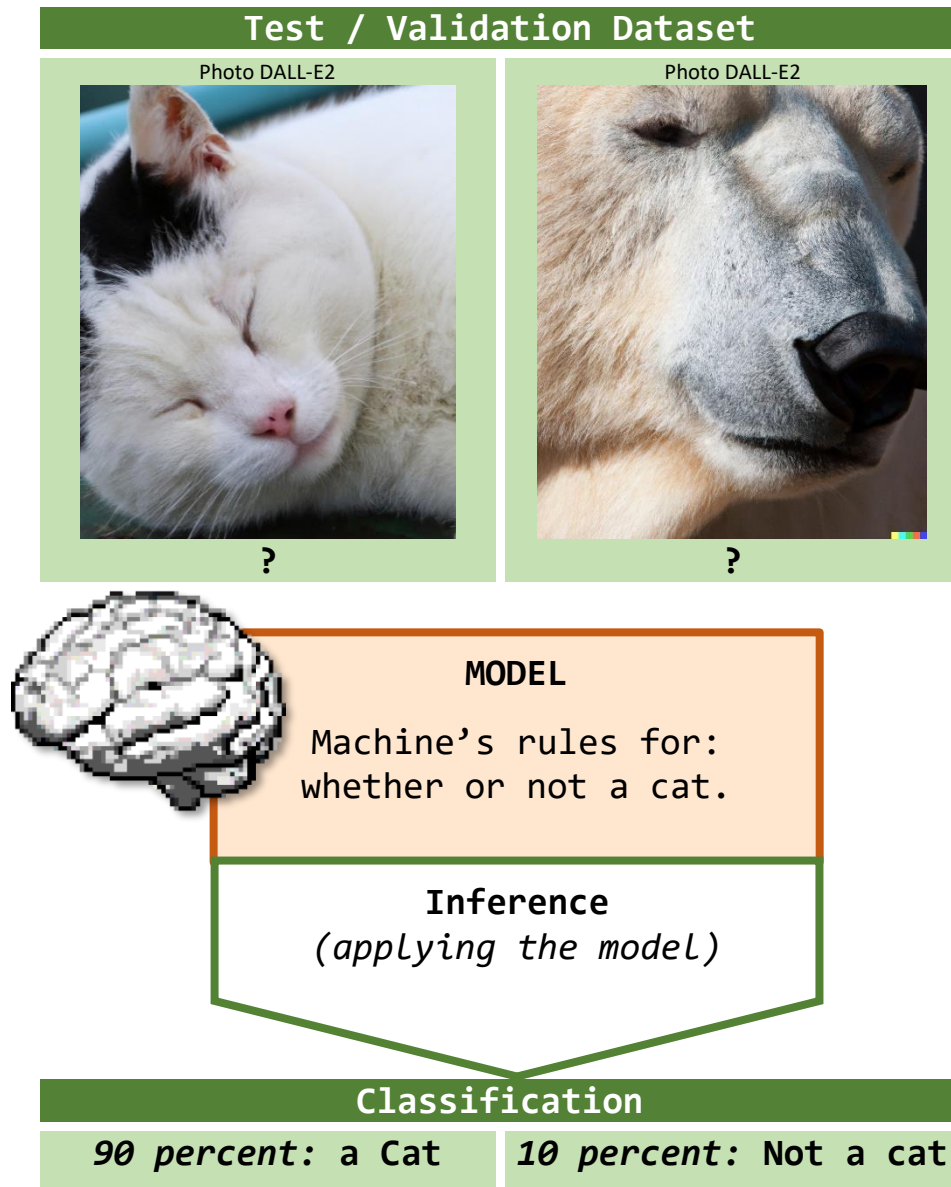
Part III: Training a ML algorithm with data.

We take a training dataset, where all data is pre-classified. This dataset is called a ,labelled dataset ‘.

We apply a machine-learning algorithm to it (*in this special case for example a convolutional neural network*):





Part IV: Testing and validating a pretrained model



A well-trained machine learning model generalizes: It recognizes a certain class of things based on similar features and therefore allows for inaccuracies. ML models mostly work with probabilities: It is not possible to decide with 100% certainty whether a picture shows a cat or a dog. With well-trained models, probability values of 80-95% are common and usually completely sufficient for the respective purpose.

Part V: How can this work?

We need 3 ingredients:

<p>1. Data. LOTS of Data! Even better: Lots of LABELLED Data!</p>  <p>...</p> <p>...</p>	<p>2. Many computers with lots of resources</p>  <p>Illustration by DALL-E2</p>	<p>3. Algorithms to process the data “There is no free lunch”-theorem” *</p> $Y = b_0 + b_1x_1 + b_2x_2 + \dots + b_nx_n$ $P(A B) = \frac{P(B A)P(A)}{P(B)}$ $d(x, y) = \sqrt{\sum_{i=1}^n (x_i - y_i)^2}$ $J = \sum_{i=1}^m \sum_{k=1}^K w_{ik} \ x^i - \mu_k\ ^2$ $y = w_0 + \sum_{i=1}^n x_i w_i$ <p>...</p> <p>...</p>
---	---	--

*NFL-Theorem: <https://machinelearningmastery.com/no-free-lunch-theorem-for-machine-learning/>

Lesson 6: NLP and the GPT-3 on “What is machine learning”?

What students should learn:

GPT-3 is a language processing model developed (the abbreviation stands for ‘generative pretrained transformer’) by the American non-profit organization OpenAI. It uses deep learning to create, summarize, simplify or translate texts. It is one of the first models that passed the turing test (!). It is (according to the current status of 2021) one of the most powerful A.I. models worldwide.

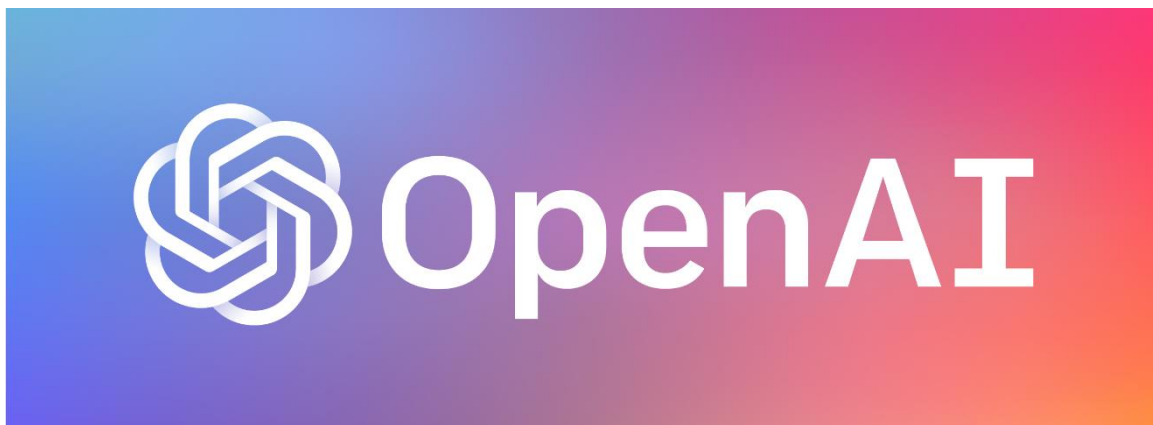
Possible students’ activities and tasks:

Students should read the text carefully. The content is less important (which is very informative, by the way). It is about the evaluation of the linguistic composition. Discuss the following questions:

- Can one still recognize here whether an artificial intelligence has written the text or not?
- And if one can still recognize it: How can you tell?
- Where are possible chances, but also dangers?
- What if, for example, this AI model were used to produce fake news?
- How can something like this kind of misuse be prevented?

Found on R-Bloggers:

<https://www.r-bloggers.com/2021/05/the-most-advanced-ai-in-the-world-explains-what-ai-machine-learning-and-deep-learning-are/>



The text on the following page was fully written by an artificial intelligence, on 12th of May 2021.

Artificial intelligence (AI), machine learning (ML), and deep learning (DL) are related concepts that are often used interchangeably. They are also three distinct and different concepts. In this blog post, we will define artificial intelligence, machine learning, and deep learning and explain why they are all different and how they are related.

Artificial intelligence

AI is a broad and complex concept that has been around for decades. AI is used to describe a concept or a system that mimics the cognitive functions of the human brain. It can be used to describe a situation where machines can act or behave in a way that mimics human behaviour. AI is often used to describe a system that can learn from experience, can use knowledge to perform tasks, to reason, and to make decisions.

There are many different types of AI. For example, there are expert systems, neural networks, and fuzzy logic. In this blog post, we are going to focus on the different types of machine learning. A machine learning model is an AI system that can learn from a dataset and can make predictions or decisions based on the data.

Machine learning

Machine learning is a subset of AI and is a method for algorithms to learn from data. It can be used to build models that can predict future behaviour based on past experience. Machine learning is used to analyse large datasets and to find patterns in the data. An example of a machine learning model is a spam filter that learns to differentiate between spam and non-spam messages. There are three different types of machine learning. Each of them is used for a different type of problem.

Supervised learning

Supervised learning is the most common type of machine learning. It is used to find patterns in data and is used to predict future behaviour based on past experience. In supervised learning, the data is split into two parts, known as the training set and the test set. The training set is used to train the model and the test set is used to evaluate the accuracy of the model.

The goal of supervised learning is to find a relationship between independent variables and dependent variables. The independent variables are the things that we know about the data. For example, the independent variables are the features that are used to describe a customer. The dependent variables are the things we want to know about the data. For example, the dependent variable is the profit made from a specific customer. The relationship between the independent variables and the dependent variables is known as the model. Supervised learning can be used to predict the profit of a customer based on the features of the customer.

Unsupervised learning

Unsupervised learning is used to find structure in the data. In unsupervised learning, there is no training set. The model is learned from the test set. It can also be used to find groups or clusters in the data or to identify anomalies in the data. Unsupervised learning can be used to find groups of similar customers.

Reinforcement learning

Reinforcement learning is a type of machine learning that is used to find good actions or decisions based on the data. Reinforcement learning is used to find an optimal action or decision that will maximize the reward. It is used to find the optimal solution to a problem. The optimal solution depends on the reward function. Reinforcement learning can be used to optimize different types of problems. For example, it can be used to optimize a non-linear function or to find the shortest route in a network (see also Reinforcement Learning: Life is a Maze).

Deep learning

Deep learning is a subset of machine learning that uses artificial neural networks. Artificial neural networks are computational models that are inspired by the architecture of the human brain. They are used to develop algorithms that can learn from data (see also Understanding the Magic of Neural Networks).

Deep learning is used to build models that can classify data or find patterns in the data. Deep learning is used to perform complex tasks such as object recognition, speech recognition, and translation. Deep learning is the most popular type of machine learning.

In this blog post, we explained the difference between artificial intelligence, machine learning, and deep learning



Lesson 7: Teachable Machine and Scratch

What students should learn in this lesson:

Train a model with a self-recorded training dataset: for generalising a class like a cup or a sponge it is necessary to collect many different objects. And those objects need to be slightly unprecise. For example: if you make a cup dataset, you have to wave the cup slowly in front of your camera to produce different perspectives of one object. the algorithm tries to find general features that are common to all objects. Inference – the predictions/classifications a model is performing after training – works with probabilities. In most of the cases, the model will not classify with 100% certainty.

Possible students' activities and tasks:

Students perform a few simple steps: Students pick out several items of the same kind from their immediate environment, such as: several different pens, scissors, cups, books, pencil cases, glasses, etc.

They first group these items into classes, program in that class in the Teachable Machine, and create photos of all the items in that group.

In the process, intuitive attention is already paid to data augmentation: The objects are moved slightly in front of the camera to change the perspective - this prevents overfitting and enables generalization.

After data capture, students train their models and can then start playing around with them. Afterwards, the model can be exported and further processed in Scratch, for example.

Tutorial: Running TensorFlow models in Scratch: <https://dalelane.co.uk/blog/?p=4201>



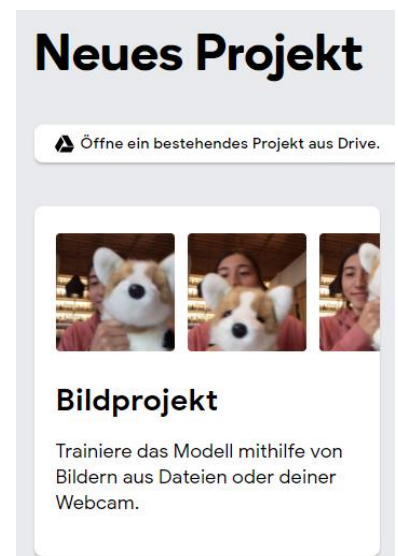
1: call Teachable Machine in Browser

<https://teachablemachine.withgoogle.com/>

2. create a new image project

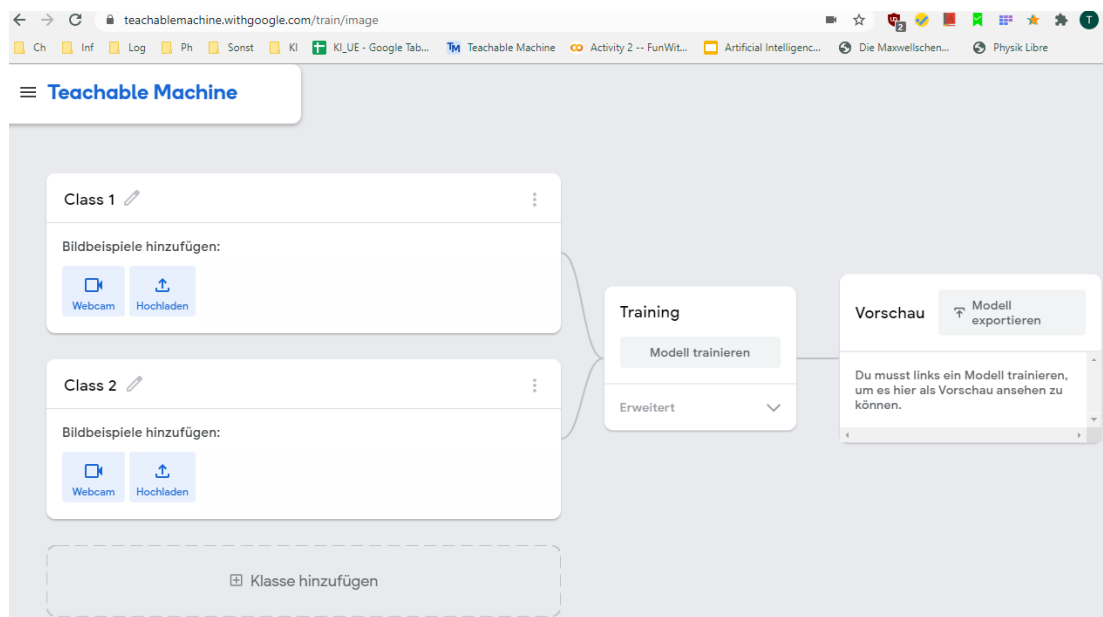
<https://teachablemachine.withgoogle.com/train/image>

There are different possibilities to choose from – we want to train a model for object classification.

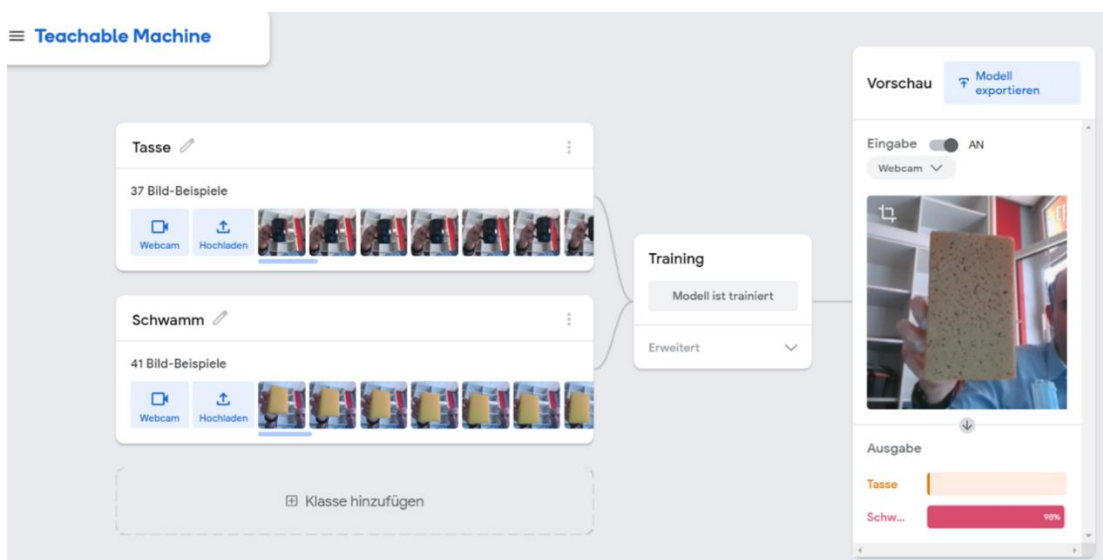


3. record your training data

To do this, click the webcam icon, start the camera, and keep any object in front of it. While holding the mouse button, several images can be taken. You can hold any number of object classes.



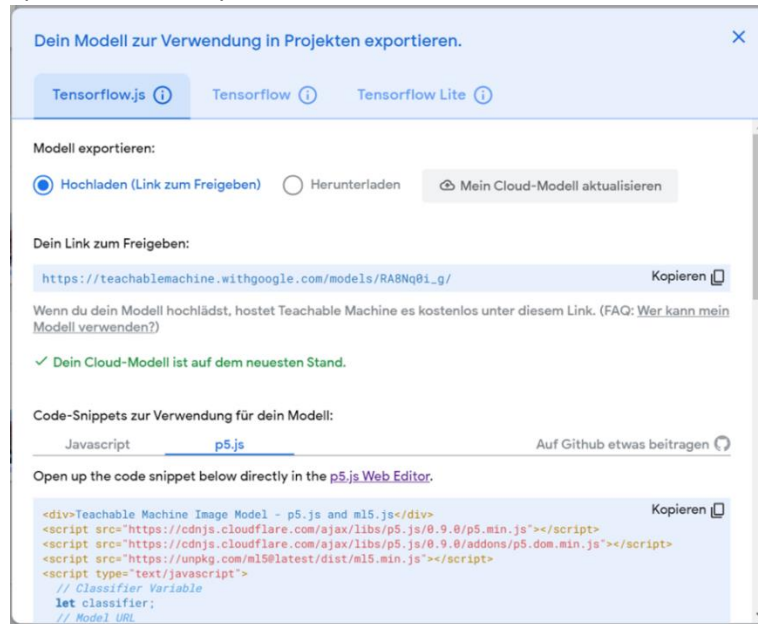
Then you train the neural network and test:



Google will record images like this: 224 x224 Pixel.

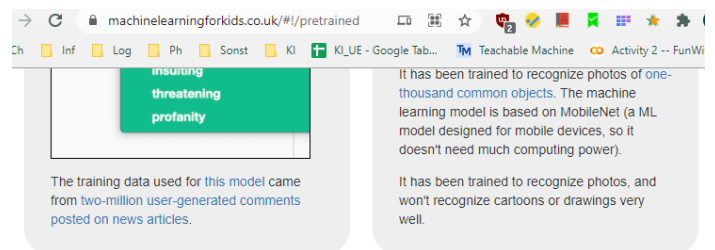


If you export the model, you should save it under the default link from Google:

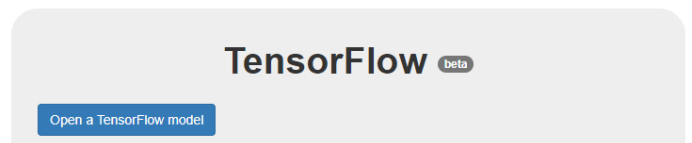


4. You can import the model in scratch

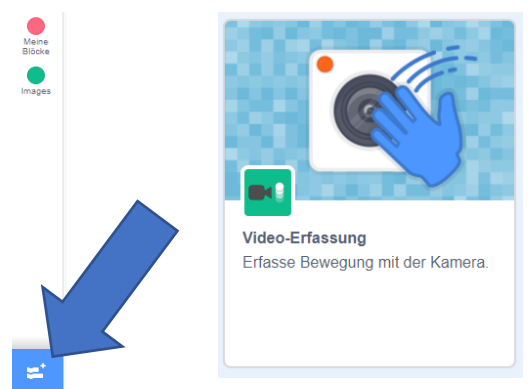
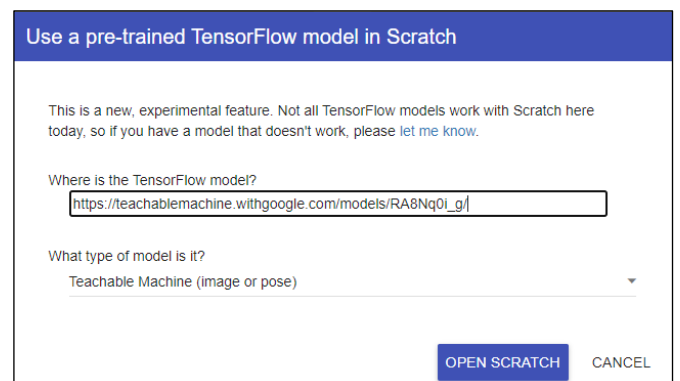
<https://machinelearningforkids.co.uk/#!/pretrained>



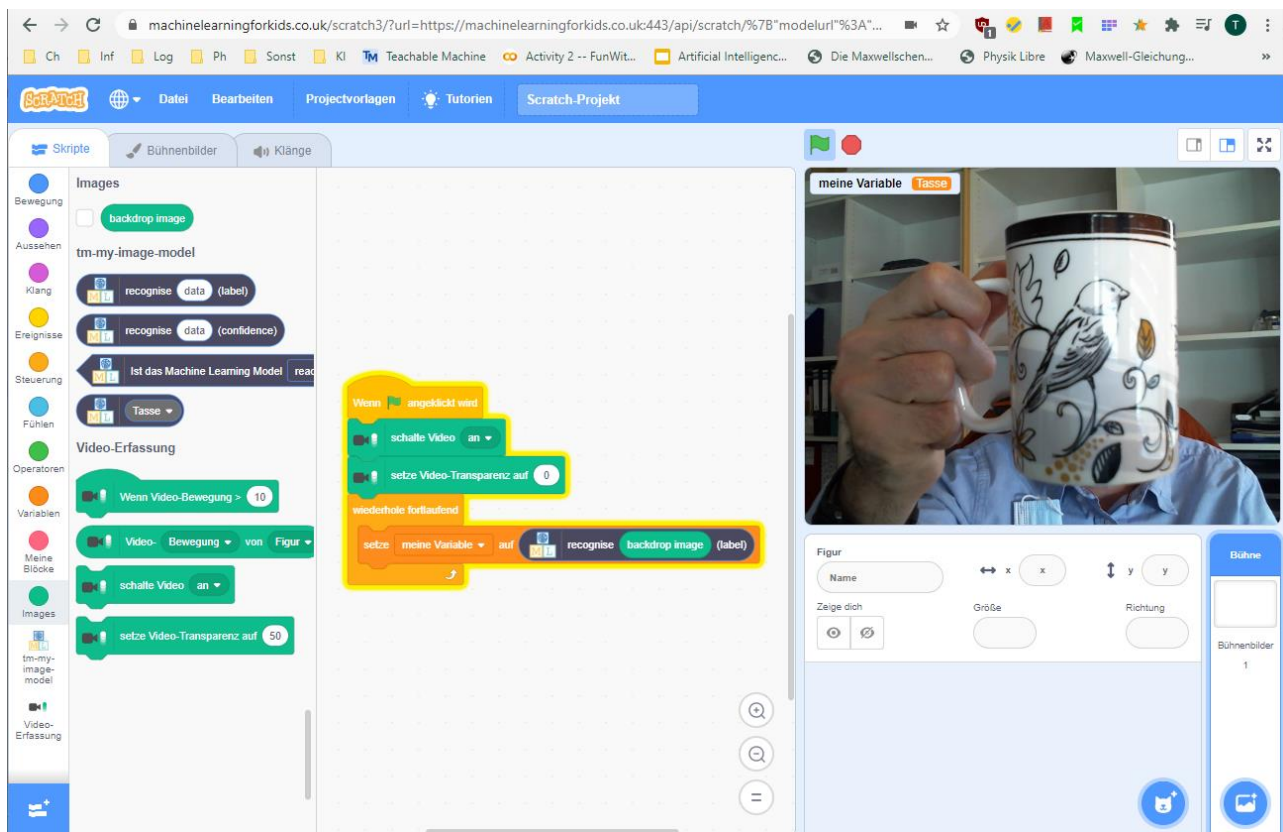
Then you enter the URL of your TeachableMachine model. This will be loaded into Scratch



Now you only need the image acquisition as a plugin, then you can start:



Here is a simple but working example program.



SO MUCH DATA EVERYWHERE



BUT SO HARD TO COLLECT ...

Lesson 8: ML principles unplugged

What students should learn:

A.I. means prediction or classification performed by a model that has been trained with training data before. This Training data needs to be collected and is based on features: numbers or categories that must be measured using sensors, e.g., mass, width/height-ratio, deformability, color, surface properties and many more. The model is generalizing those many different feature values and after this training process it can predict or classify unknown data, which is called inference. It is not sufficient to take one single measurement, since no generalization or rule-extraction is possible with only one sample. Many samples are needed to form a meaningful dataset.

Photo DALL-E2

Possible students' activities and tasks:

the students are asked to act out the process of creating a model. to do this, they solve a simple problem: how can different everyday objects be distinguished from one another? to do this, they are each given an ensemble of some cups or sponges or pens - depending on what is already available in the classroom. the students are asked to develop their own ideas as to which measurable object properties can be used to separate the different classes of objects from one another. a cup, for example, is generally heavier than a pen. a sponge generally has a different height-to-width ratio than a pen. the students are therefore asked to construct and use suitable properties. after all the objects have been measured, the measured values are entered in diagrams. if the properties were well chosen, the point clouds can, for example, be easily divided by separating lines - a classification has been done.

Part I: how to teach a machine to distinguish between a sponge or a cup or a pencil?



Possible features:

1. color
2. mass
3. surface texture
4. plasticity/deformability
5. Ratio height to width

Selection of the best features:

How unique or typical are the characteristics? How easy are they to measure?

Deformability and mass are best suited.

→ determination of these two characteristics; Modeling completed.

Part II: Generating training data:

Students can perform their own measurements based on 3-4 different objects of each class.



Result: labelled training data

	mass (g)	deformability (%)
Cup 1	240	0
Cup 2	400	5
Cup 3	290	2
Cup 4	120	3
sponge 1	12	35
sponge 2	23	20
sponge 3	7	15
sponge 4	30	50

Is this data sufficient (is the meaningfulness robust enough)?



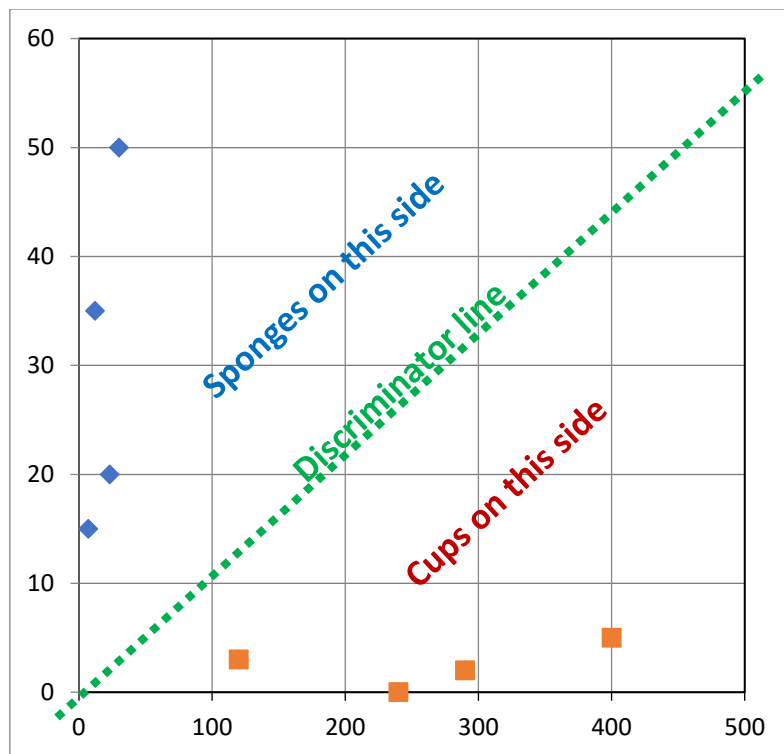
What if a sponge is wet and has a much higher mass?

What if a cup is made of a deformable material?

And: Does the deformability depend on the mass?



Part III: Presented in a diagram like this:

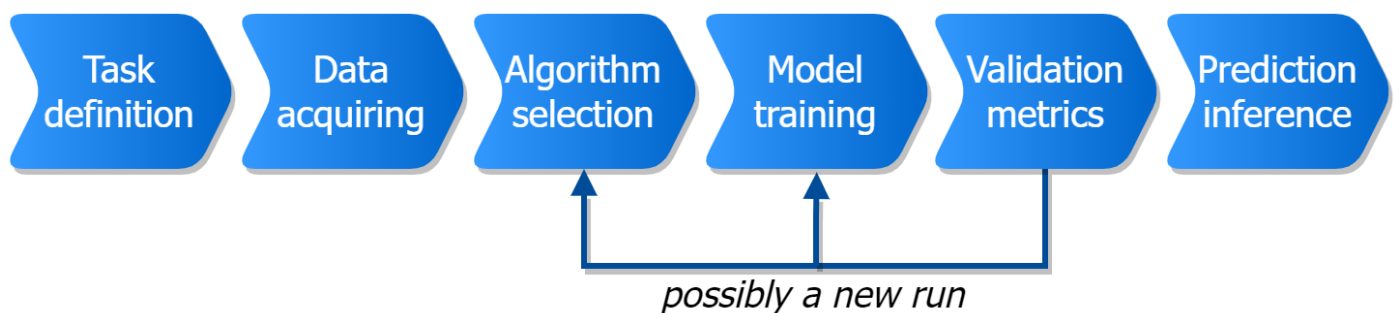


The machine can use this diagram to distinguish the two item classes from each other, and it learned this from labeled training data!

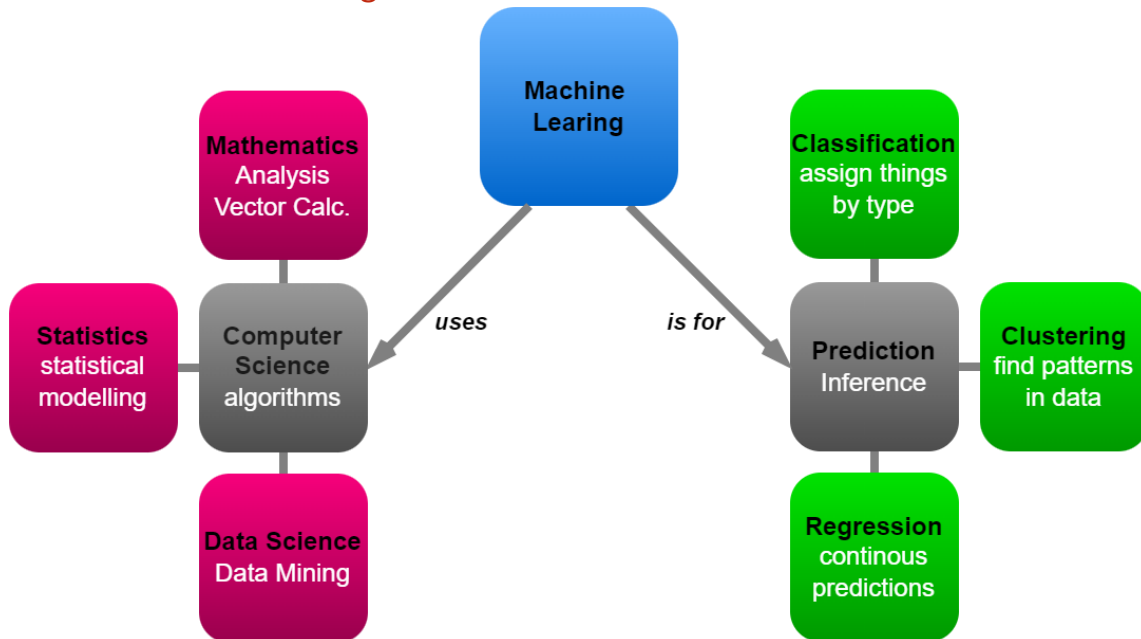
Part IV: Summary and analysis of the lesson

Process of training a machine-learning model:

- We have to define the task precisely so that we can collect the necessary data.
- We need to collect data in order to be able to train an ML algorithm.
- We need to find out an effective ML algorithm that can cope best with the data.
- We need to train the model with the data and the algorithm.
- We need to measure if the model is making good predictions and possibly tweak it.
- We need to use the finished model correctly.



Part V: What is machine learning all about?



AI is about prediction, classification, and clustering.

„Fundamentally, machine learning is a technique for using computers to predict things based on past observations. AI technology (algorithms) give computers the ability to learn without being explicitly programmed.

It is a completely different and independent approach for utilizing computers compared to traditional programming. “

Some useful resources for the A.I. lessons

A: Datasets

No machine learning without data! To be able to work out the basics with students as a teacher, you need pedagogically well-prepared data sets. For this purpose, it is advantageous to know as many different data sources as possible in order to filter out the most suitable ones for teaching.

Common sources for datasets:		
http://archive.ics.uci.edu/ml/index.php		University of Irvine
https://www.kaggle.com/		Biggest Open Source
https://en.wikipedia.org/wiki/List_of_datasets_for_machine-learning_research		List of Dataset-Sources
https://appen.com/datasets-resource-center/		Commercial
https://data.world/		Community free
https://datasetsearch.research.google.com/		Google repositories
http://yann.lecun.com/exdb/mnist/		'Famous' MNIST DB
 Datasets in Orange Data Mining		Built-in the application

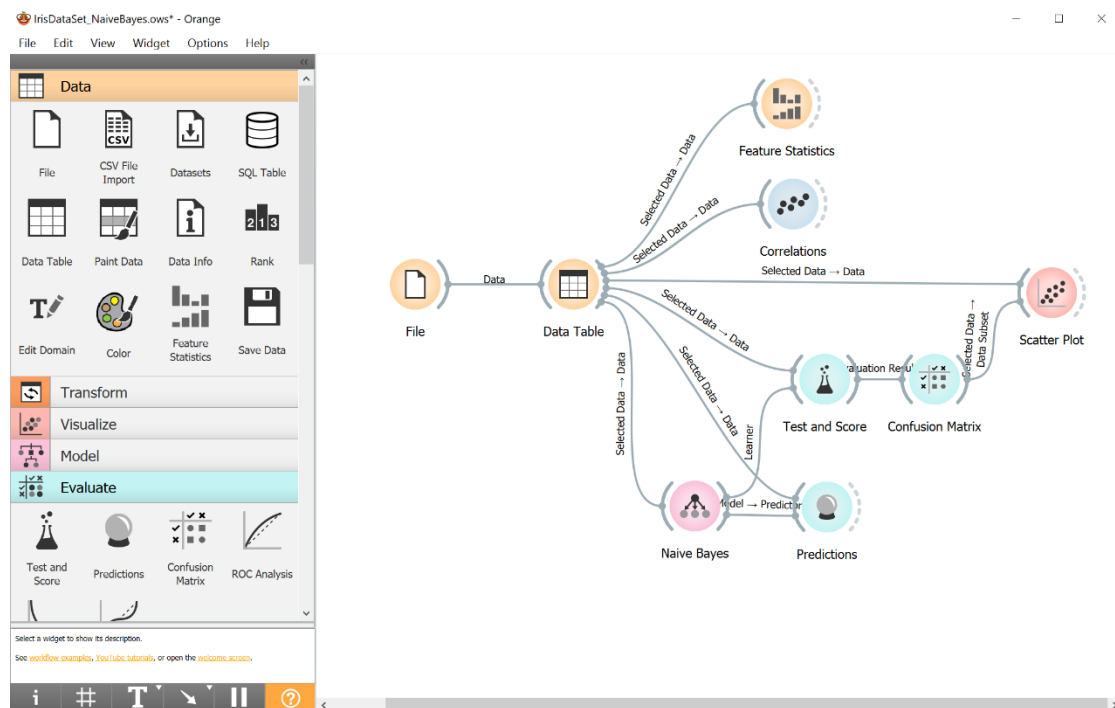
B: Simulations

People process visual information a thousand times faster than written information. Simulations and visualizations refer to this. They are just as important for students as teaching or project work.

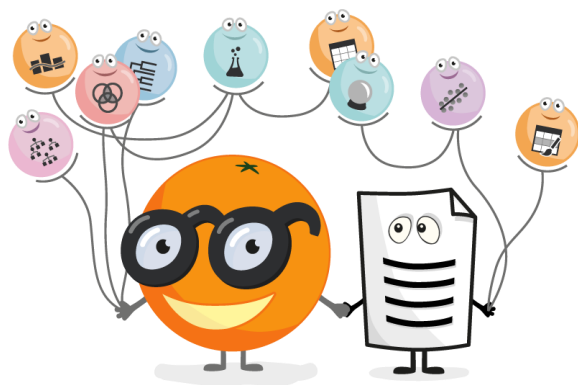
Visualisations, Demonstrations, Simulations	Topic
https://www.cs.ryerson.ca/~aharley/vis/conv/flat.html	CNN-Demo
https://poloclub.github.io/cnn-explainer/	CNN-Explainer
https://lecture-demo.ira.uka.de/convolution-demo/	Convolution Demo
https://cs.stanford.edu/~karpathy/svmjs/demo/demoforest.html	Decision Tree & RandomForest
https://playground.tensorflow.org/	FAMOUS NeuralNetwork Demo
https://iludis.de/kMeansClustering/index.html	k-Means-Clustering
https://user.ceng.metu.edu.tr/~akifakkus/courses/ceng574/k-means/	k-Means-Clustering
https://lecture-demo.ira.uka.de/kmeans-demo/preset=1	k-Means-Clustering
https://iludis.de/kNNDemo/index.html	kNN
https://lecture-demo.ira.uka.de/knn-demo/#	kNN
https://iludis.de/LinearRegression/index.html	Linear Regression
https://phet.colorado.edu/en/simulation/least-squares-regression	Linear Regression
https://lecture-demo.ira.uka.de/neural-network-demo/	Perceptron Demo
https://iludis.de/Perceptron/index.html	Perceptron with Gradient decent
https://ml4a.github.io/demos/	Repository with many demos
https://iludis.de/svmDemo/index.html	Support Vector Machine
https://jgreitemann.github.io/svm-demo	Support Vector Machine

C) Software: Orange Data Mining

In this teaching concept, the software package "Orange Data Mining" is very often used. It allows students who have never programmed before to access the ML workflow in an easy and intuitive way. It is node-based and therefore works visually: Students recognize the work steps already completed, can plan and experiment further based on them, and furthermore always see the big picture.



The software is free, does not require any installation rights and also runs on older computers. (Only with neural networks sometimes larger computing times are needed).



The software package can be downloaded here:
<https://orangedatamining.com/>

Lesson 9: Decision Trees

Or: How to convert data to trees

What students should learn:

Decision trees are the perfect introduction to the world of machine learning algorithms. They illustrate how a machine decomposes a task, structures it, and uses optimization calculations to build the best possible model for predictions. In addition, decision trees can be interpreted very well: The finished model can be easily visualized and comprehended.

Possible Students activities and tasks:

Possible students' activities and tasks:

Students learn about the tree structure of a decision tree using a very simple introductory quiz. Afterwards, they are given a task for group work: An everyday scenario is recorded in tabular form and is to be translated into a decision tree. Once this tree has been intuitively created, the students are directed to optimize this tree. Here, in addition to the optimization, the underlying principles of the Gini index are explained.

Part I: Introductory quiz

Teacher:

"You know the concept of decision trees from everyday life. So, let us start with an example:

[Teacher hands a student a secret note with 8 different animals:]

Teacher:

"He/she now has a list with 8 different animals and randomly chooses one I do not know. I will try to guess which one with only three questions, one after the other – and the only allowed answers are "yes" or "no":

<p><i>List with animals:</i></p> <ul style="list-style-type: none"> • wolf • squirrel, • chicken, • cow, • ladybug, • bee, • silverfish, • bed bug 	<p><i>List with questions:</i></p> <p>1 does it have more than 5 legs?</p> <p>2a does it live inside your home?</p> <p>2b does it live on a farm?</p> <p>3a does it collect pollen?</p> <p>3b does it feed from blood?</p> <p>3c is it cute?</p> <p>3d Is it tall?</p>
--	---

[Teacher draws the decision tree on the blackboard, starting with the just asked questions.]

Teacher: "You still have the secret list. Let's try some two more animals from it."

[Students complete the tree together with the teacher using all 8 animals.]

Teacher: "Now we have every animal at its own end of this tree. What about the following examples?"

[Students classify all missing animals asked by the teacher.]

Sparrow, farm goose, horse, Humvee, centipede, cockroach, head louse etc.

Part II: Visualization of the first tree

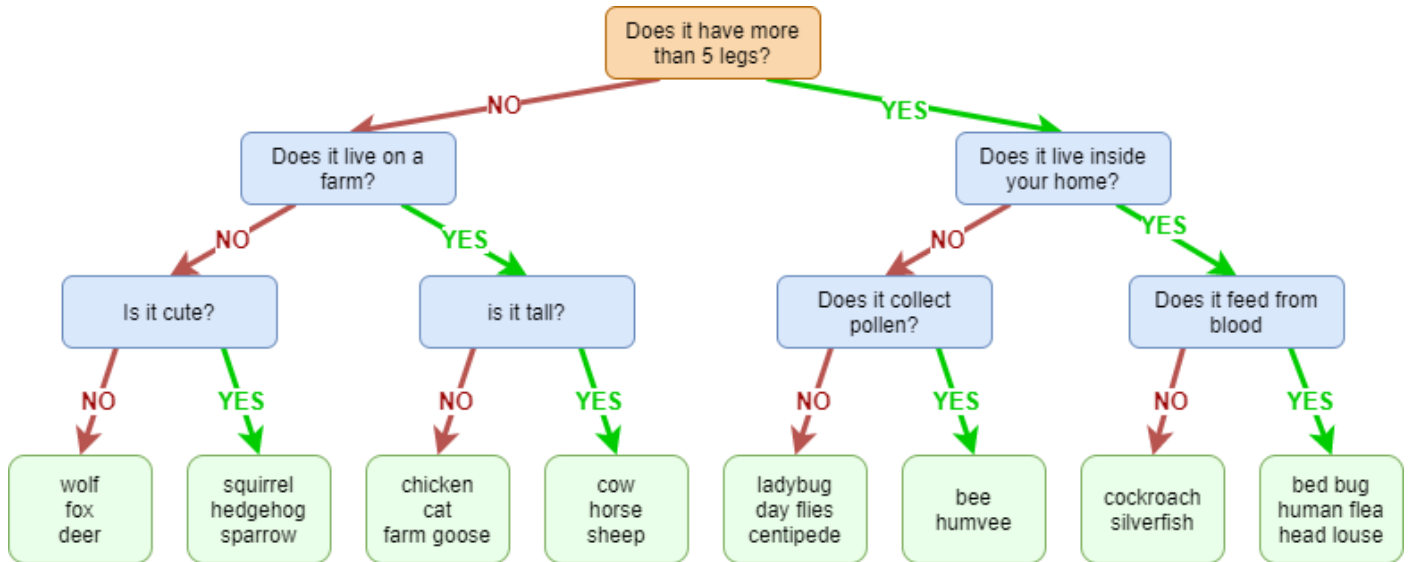


Figure 2: Blackboard sketch: completed decision tree

Part III: Fundamentals and terminology of Trees) Teacher:

“Inside this so-called decision tree, some species are forming their own classes. So, we can use this tree for classification of animals – as long it is useful for us: It is only a model for a specific task.”

“Let’s have a look what a tree in computer science is.

What’s its purpose and how are the different parts named? – ATTENTION: TERMINOLOGY!”

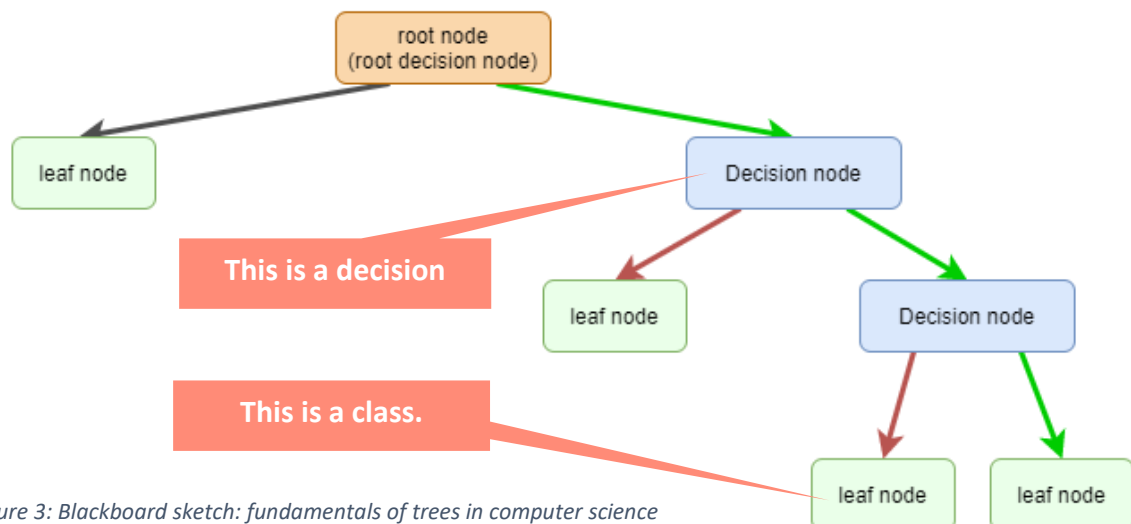


Figure 3: Blackboard sketch: fundamentals of trees in computer science

Part IV: Problem based teamwork: building a tree from a dataframe

Teacher:

“Now for some teamwork. Imagine the following problem: You wonder what your friend probably will do today, and you want to give him a call. But you hesitate – how will he behave? Today the sun is shining till the afternoon and there is a lot of homework to do. You try to predict his decision based on your experience in the past. Therefore, you write down your thoughts:”

Already together with friends?	Sun is shining?	Still some homework to do?	Observed behaviour
No	No	No	Video games
No	No	Yes	Doing homework
No	Yes	No	Play outside
No	Yes	Yes	Play outside
Yes	No	No	Video games
Yes	No	Yes	Video games
Yes	Yes	No	Play outside
Yes	Yes	Yes	Play outside

Figure 4: Work sheet with data frame

[Students convert the training dataset to a decision tree. The first three columns are the data/features, the last column is the target dataset. Different students' approaches are possible]

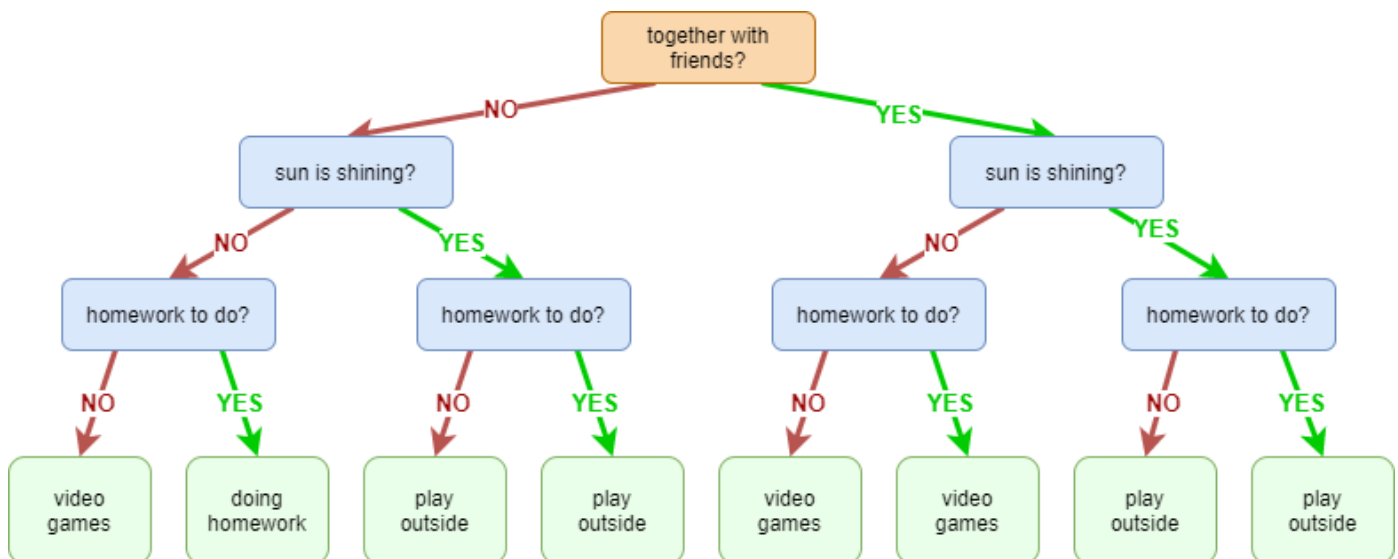


Figure 5: Blackboard sketch for the completed decision tree

Part V: Optimizing a decision tree using simplification.

Teacher:

“There are some decisions that can be simplified. Can you find them? So, please simplify the tree and explain what you are doing.”

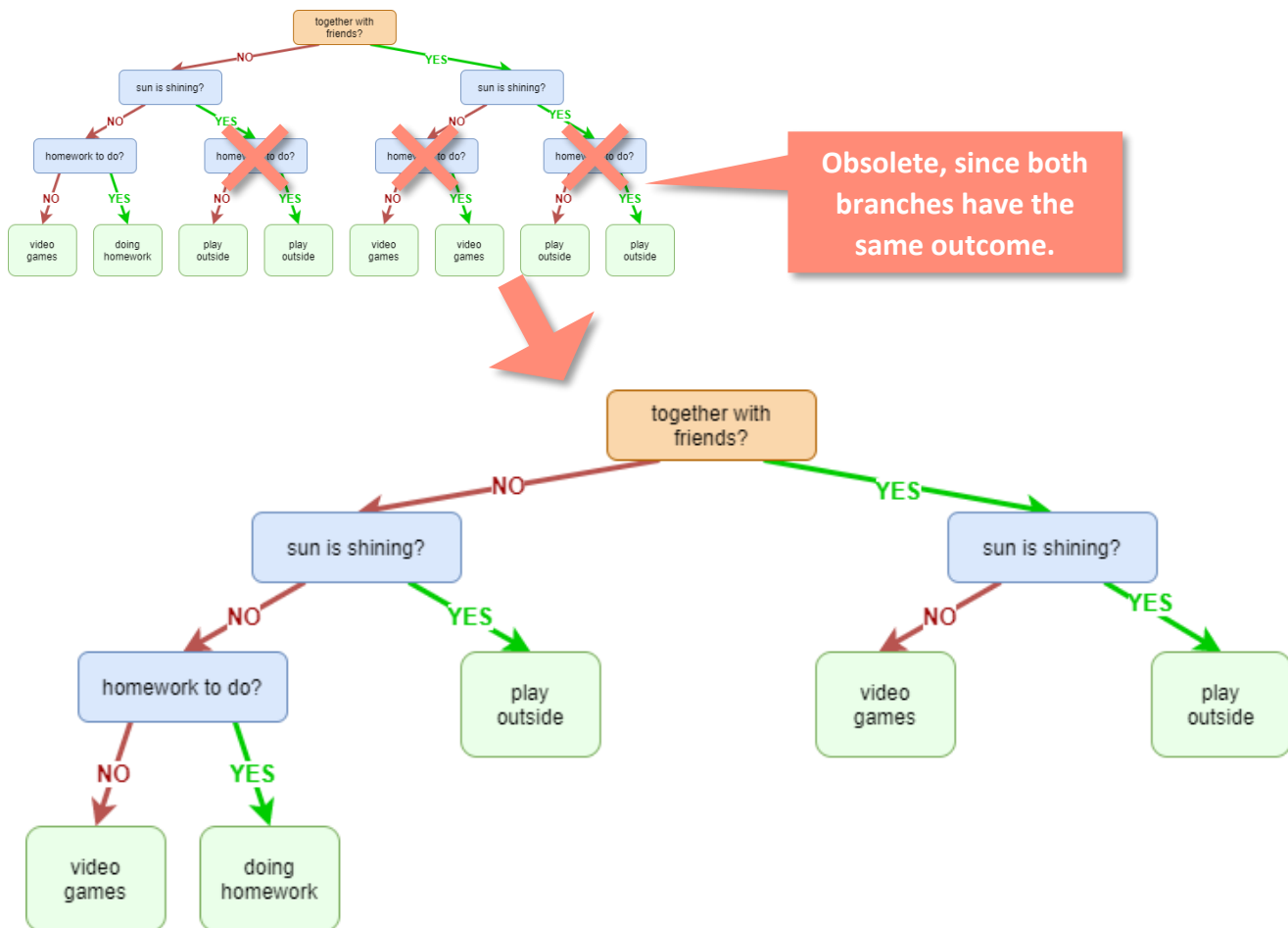


Figure 6: improved, but not optimized decision trees

Teacher:

“Why is it better?”

Students:

“There are less decisions necessary to accomplish the same task. There are in average:
12 decisions divided by 5 outcomes = 2,4 decisions per outcome”

Some more simplified trees are possible, and some of them should be transferred into the students exercise books since it is important to understand: There is no single method to solve the problem.

Part VI: Which is the best decision to start with? Entropy-driven optimization.

Teacher: “We didn’t take into account that we could also vary the sequence of the different questions. Is it important to ask the three questions like we did until yet?”

Students: “No, that’s not necessary.”

Teacher: “So, let’s vary the start question and search for the best solution. How many variations are possible?”

Students: “3 different questions at the beginning, then 2 remaining different questions and a fixed question at last: $3 \times 2 \times 1 = 6$ different decision trees”

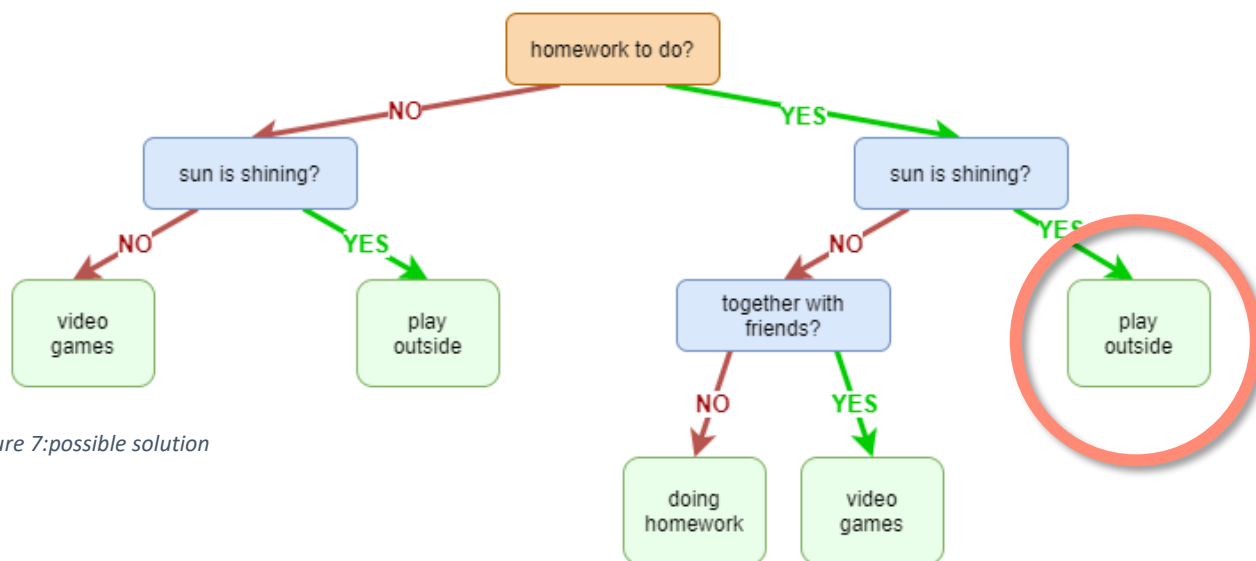
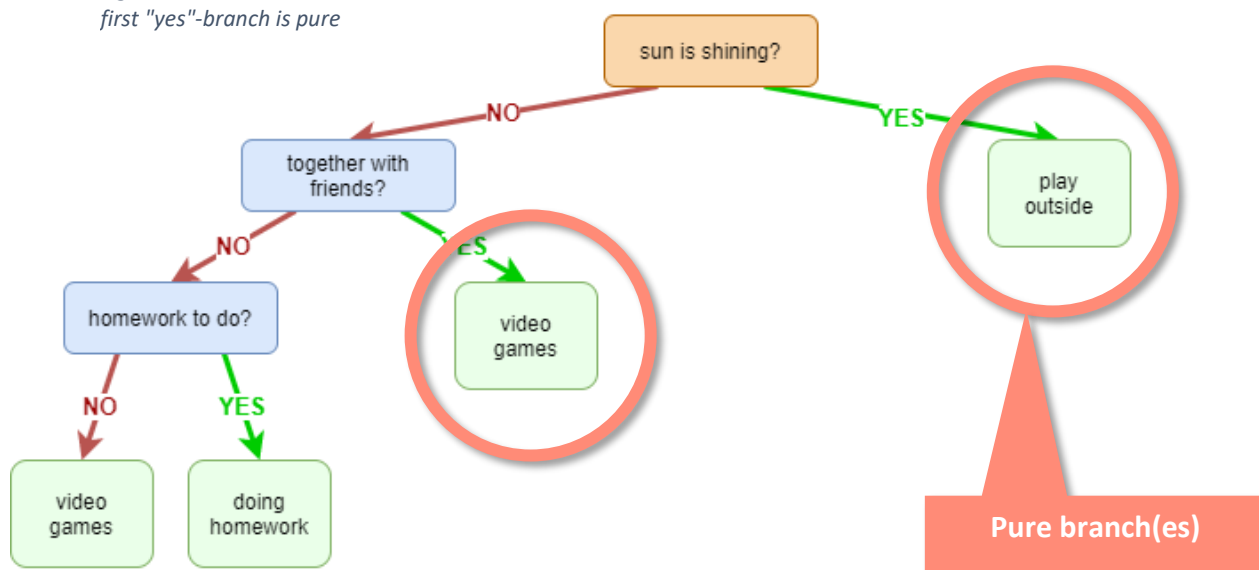


Figure 7: possible solution

Teacher:

“If we can optimize in such a way, that a decision is producing a ready classification without further splitting, the result is called “pure” (TERMINOLOGY). The purer the decision after a split the better. But: What is purity? Can it be measured?”

Figure 8: best solution, since the first “yes”-branch is pure



Part VII: How to imagine what purity is and how can it be measured.







Example: you know the cherry-banana-shake?

It's a mixture of two different juice ingredients.

Relationship to our problem:

There are many ways to measure purity, the most common ones are the so-called GINI-coefficient and the so-called ENTROPY. Both are independent from each other and describe the degree of purity a decision is producing. They vary between 0 and 1.

In general: the lower the GINI-coefficient the more pure the decision – and the better!

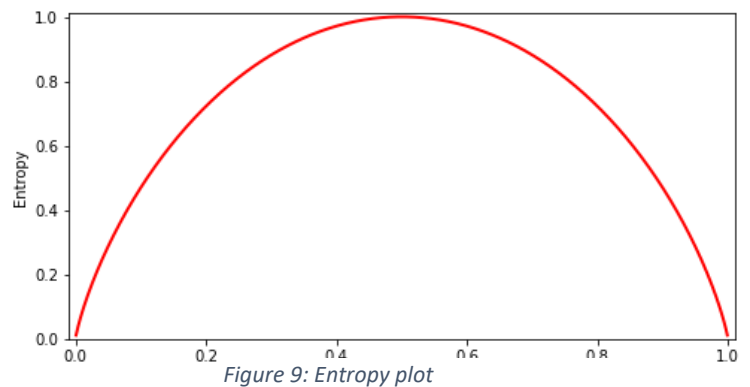
			
no intermixture,	Slight intermixture	heavy intermixture	complete intermixture
Gini = 0	Gini ≈ 0,15	Gini ≈ 0,3	Gini = 0.5
Entropie=0	Entropie=0.3	Entropie=0.7	Entropie=1
Relates to a decision splitting that is:			
Perfect	Nearly perfect	Mostly acceptable	Senseless

```
import numpy as np
import matplotlib.pyplot as plt
import math as math

def entropy(p):
    return - p * math.log2(p) - (1-p)
    * math.log2(1 - p)

x = np.arange(0.001, 1.0, 0.001)
print(x)
ent = [entropy(p) for p in x]
print(ent)

fig = plt.figure(figsize=(8,4))
ax = plt.subplot(1,1,1)
ax.plot(x, ent, color = 'red', lw = 2)
plt.ylim([0, 1.01])
plt.xlim([-0.01, 1.01])
plt.xlabel('p(x)')
plt.ylabel('Entropy')
plt.show()
```



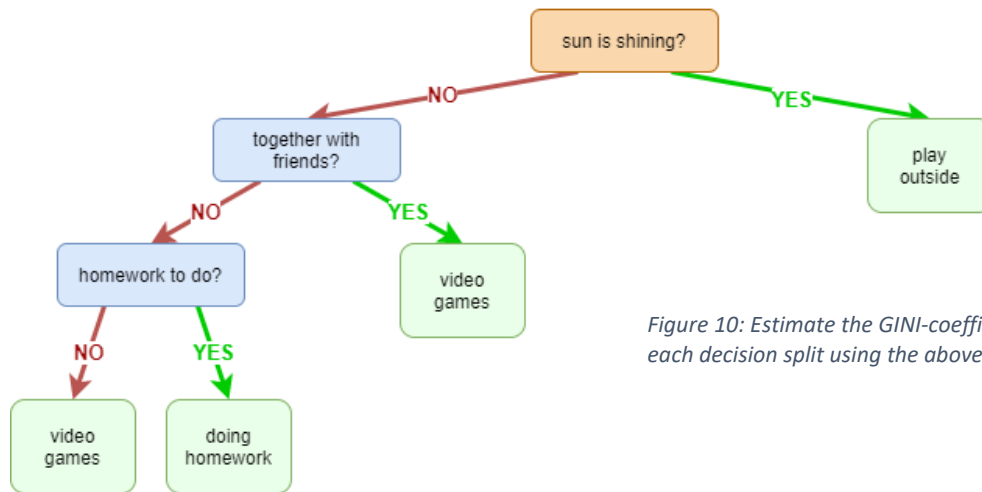


Figure 10: Estimate the GINI-coefficients for each decision split using the above examples.

Part VIII: Summary, “what we have learned today”

Decision trees are an algorithmic tool to predict or classify new appearing data based on collected training data in the past. It structures the way finding the right classification through a hierarchical chain of decision splits. Those splits can be optimized using gini- or entropy-driven measurements. The decision-tree-algorithm takes input-data, calculates the best splits and outputs a tree that can easily be visualized.

Part Part IX: Overfitting

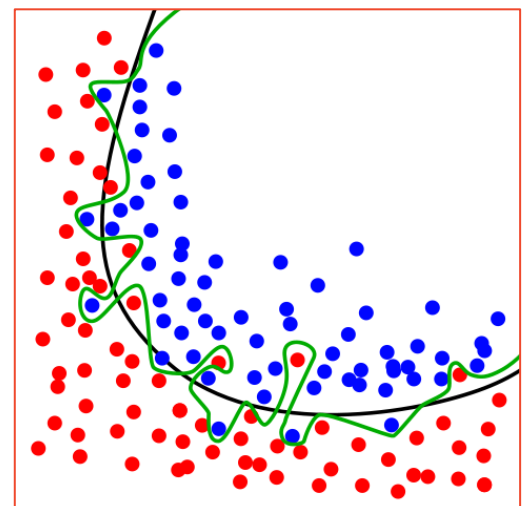


What would happen if we would take every single datapoint into account without generalizing?

In this case, no generalization would have taken place: It would be a case of the so-called ‘overfitting’.

From <https://en.wikipedia.org/wiki/Overfitting>

The green line represents an overfitted model and the black line represents a regularized model. While the green line best follows the training data, it is too dependent on that data and it is likely to have a higher error rate on new unseen data, compared to the black line.

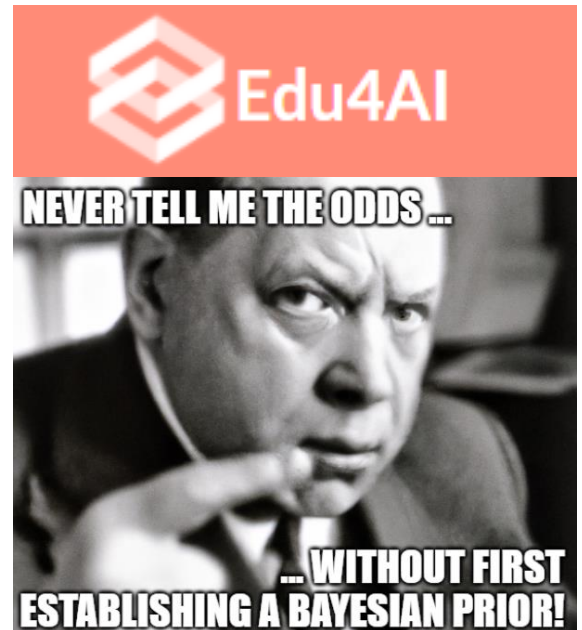


Lesson 10: Naive Bayes Classifier

Or: "Never tell me the odds!"

What students should learn:

Students learn about the concept of conditional probability using the famous psychological test question of "librarian or farmer." The surprising result opens new insights for the students about the limits of their own thinking. In the example that builds on this, the significance of a typical Corona quick test is analyzed concretely, with the students working out the technical terms of true positive/ false negative/ true negative/ false positive. The well-known formula of the Bayes theorem results in direct consequence. In the following introduction of the classifier the students are introduced to the integration of different features. This can be deepened by means of text mining or a small orange data mining project.



Possible students' activities and tasks:

The point here is to learn and apply the most important basic statistical concepts in addition to the fundamentally important Naive Bayes classifier: Using the Coronatest statistic they are familiar with; students work out the branching tree of all four possible test results. By classifying them as True Positive, False Negative, True Negative, and False Positive, they derive the conditional probability equation, Bayes' Theorem. Students apply the concepts of sensitivity (= recall), specificity in their own exercises. Later, this still one-dimensional approach is extended for several features, until one finally arrives at the general version of the classifier. In addition, the confusion matrix and the important metrics such as Precision, Recall and Accuracy emerge.

Part I: Introductory question: Librarian or farmer?

As you consider the next question, please assume that Steve was selected at random from a representative sample:

An individual has been described by a neighbor as follows: "Steve is very shy and withdrawn, invariably helpful but with little interest in people or in the world of reality. A meek and tidy soul, he has a need for order and structure, and a passion for detail." Is Steve more likely to be a librarian or a farmer?

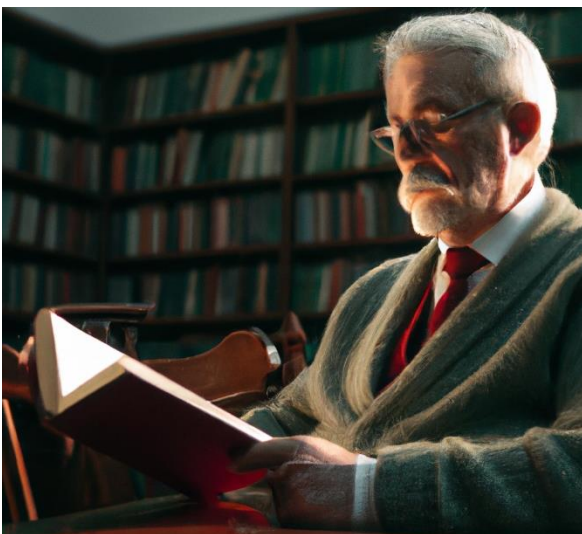
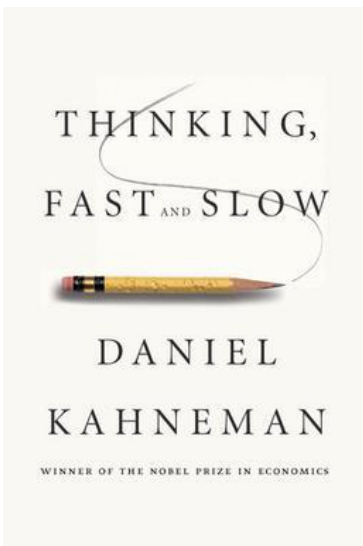


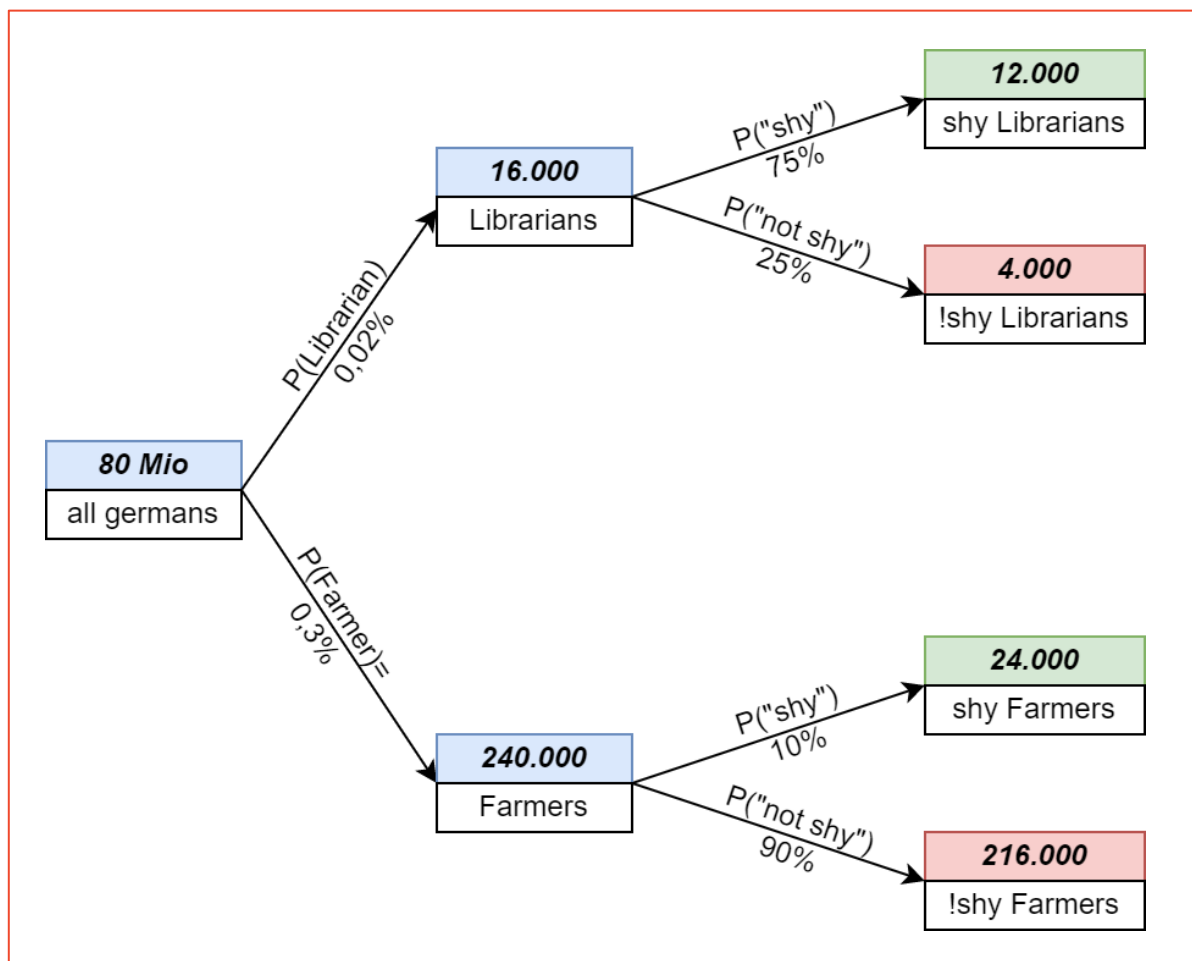
Photo DALL-E2



Photo DALL-E2



This famous example comes from the book "Thinking, Fast and Slow" by Daniel Kahnemann and is used to recognize the limits of one's own thinking. Unless specifically trained, humans tend not to think in terms of conditional probabilities. However, for the Naïve Bayes Classifier, students need to develop this very way of thinking.



On the board, the decision tree is developed with the students. Many calculations can be performed on it and the important concepts of conditional probability can be derived from it:

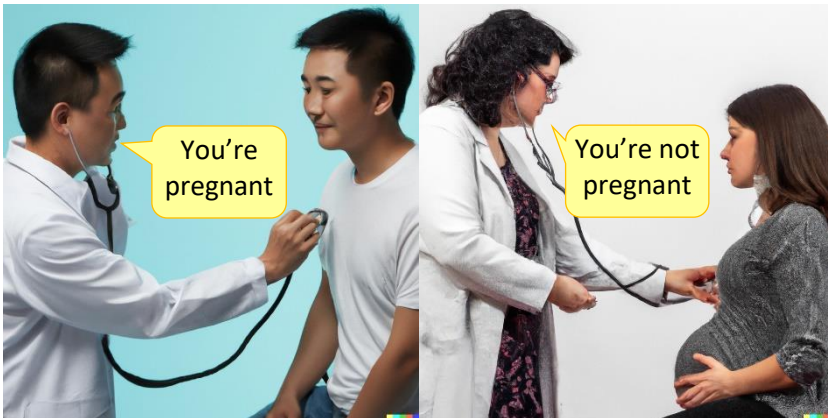
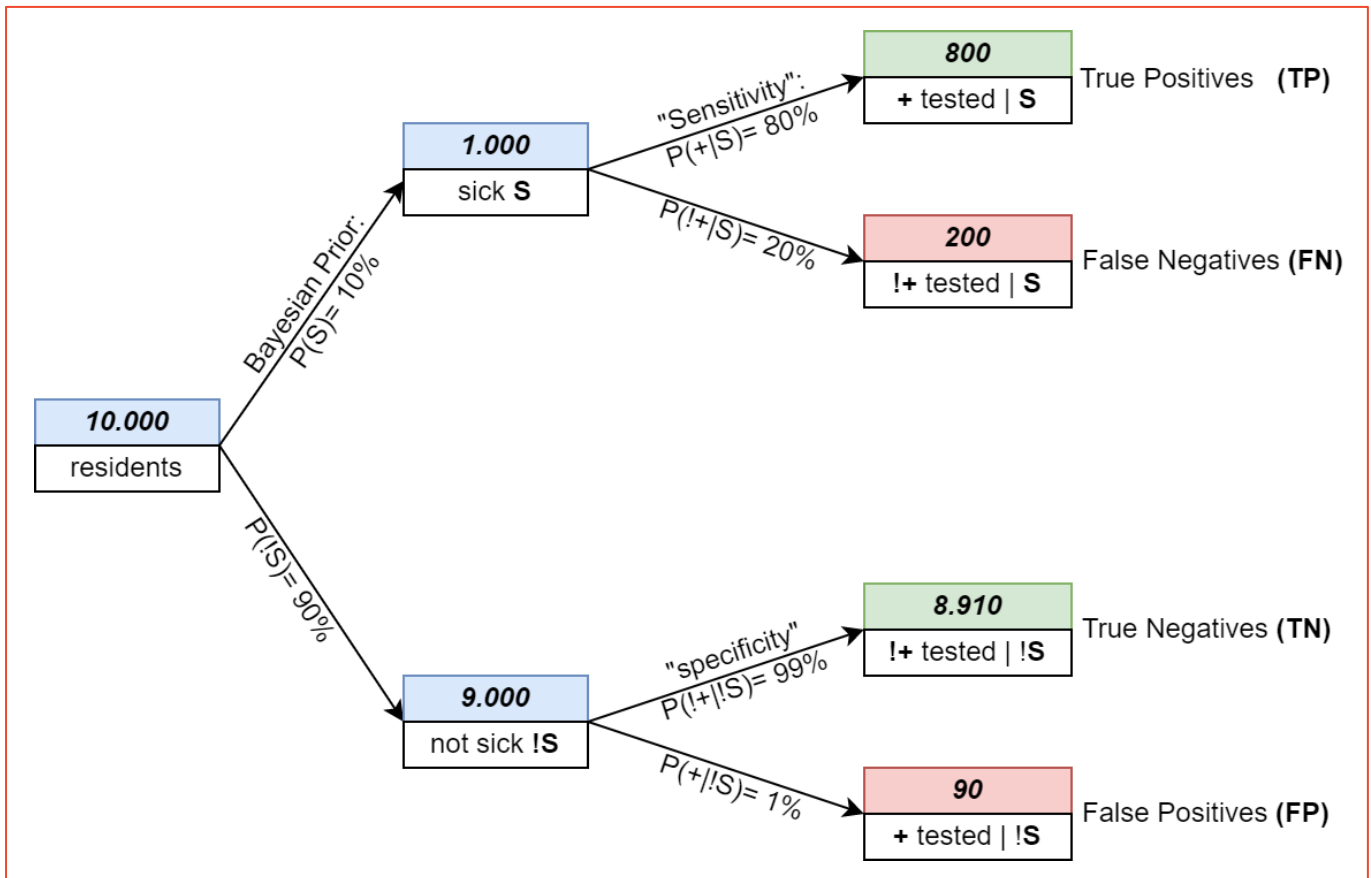
- What is the probability of finding a shy person under the condition that he is a librarian?
- What is the probability of finding a shy person among the set of all farmers?
- Accordingly, what is the probability of finding a librarian among the shy people?

Answer: Very low, because: In general, there are significantly fewer librarians than farmers,

so in absolute terms there are significantly fewer shy librarians than shy farmers.

Part II: The Math of a common “Coronatest”

Most students will be familiar with the typical Corona quick tests either from their home environment or from school: Sticks that you stick up your nose and wait after being treated with test liquid to see if a test strip wetted with it shows the infamous second mark. Which then results in a quarantine period. Students should have a very emotional relationship with this test.



However, these tests are relatively uncertain (which is why PCR was often used in practice to make reliable statements). The test characteristics are:

Sensitivity (recall): this is the probability with which a sick test person tests positive.

Specificity: This is the probability with which a healthy test person tests negative.

In both cases, correctly tested positive and correctly tested negative, there are also the two complementary events, namely a false positive or false negative test result.

Which one is the false positive and which one is the false negative?

Part III: Introduction of the confusion matrix

The results of the decision tree can be entered in the so-called four-field table (mathematics technical term), or the so-called confusion matrix.

Here is the same confusion matrix in duplicate:

In the upper one the numerical values are entered, in the lower one the corresponding probabilities and proportions, or rates.

The symbol for 'Not' was taken from the programming syntax, where it is usually represented as an exclamation mark ("!").

Detailed designation with concrete numerical values

	Actually sick (S)	Actually not Sick (!S)	Sum
tested positive	800 "True Positive" TP	90 "False Positive" FP	890 = 10.000 residents · P(+)
tested negative	200 "False Negative" FN	8.910 "True Negative" TN	9.110 = 10.000 residents · P(!+)
Sum	1.000 = 10.000 · P(S)	9.000 = 10.000 · P(!S)	10.000 residents

Symbol display with rates, or shares (number-independent):

	S	!S	Sum
+	$P(S) \cdot P(+ S) = 0.1 \cdot 0.8$ TP rate	$P(!S) \cdot P(+ !S) = 0.9 \cdot 0.01$ FP rate	$P(+) = 0,089$
!+	$P(S) \cdot P(!+ S) = 0.1 \cdot 0.2$ FN rate	$P(!S) \cdot P(!+ !S) = 0.9 \cdot 0.99$ TN rate	$P(!+) = 0,911$
Sum	$P(S) = 0.1$	$P(!S) = 0.9$	1

Terminology used:

Term	In normal language	Terminology
$P(S)$	Probability of being sick	<i>Prior</i>
$P(+)$	Probability of being positive tested	<i>Evidence</i>
$P(!S)$	Probability of being not sick	
$P(!+)$	Probability of being not positive tested	
$P(+ S)$	Probability of being positive tested given being sick	<i>Sensitivity</i>
$P(!+ S)$	Probability of being not positive tested given being sick	
$P(+ !S)$	Probability of being positive tested given not being sick	
$P(!+ !S)$	Probability of being not positive tested given not being sick	
$P(S +)$	Probability of being sick given positive tested	<i>Posterior</i>

What are we interested in? Deriving the Bayes Theorem:

In the proportion of people who are actually sick under the condition that they have been tested positive, $P(S|+)$:

$$P(S|+) = \frac{TP}{TP + FP}$$

True positives divided by all people that are tested positive (all positives: true positives plus false positives)

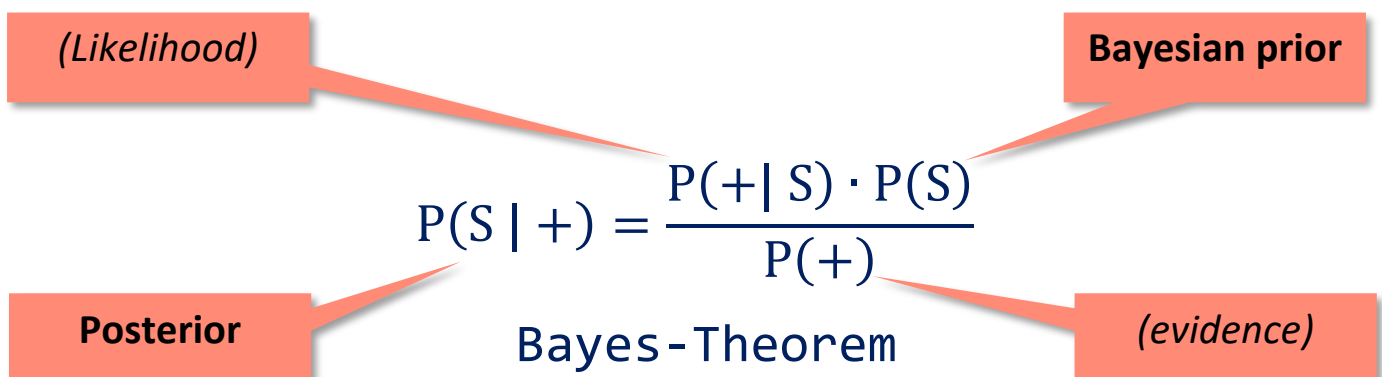
$$P(S|+) = \frac{TP \text{ rate}}{TP \text{ rate} + FP \text{ rate}}$$

Dividing by all residents eliminates the absolute number of people

$$P(S|+) = \frac{P(+|S) \cdot P(S)}{P(+)}$$

Look at the tree above: TP rate = $P(S) \cdot P(+|S)$. TP rate + FP rate = probability of being tested positive, $P(+)$

$$P(S|+) = \frac{800}{800 + 90} = \frac{0.1 \cdot 0.8}{0.089} = 0.898$$



Part VI: Why is the Bayesian prior this important?

Recalculate for a probability of being sick of 30%, $P(S) = 0.3$

	Actually sick (S)	Actually not Sick (!S)	Sum
tested positive	2400 TP	70 FP	2.470
tested negative	600 FN	6.930 TN	7.530
Sum	3.000	7.000	10.000

$$P(S | +) = \frac{2400}{2400 + 70} = \frac{0.3 \cdot 0.8}{0.2470} = 0.971$$

Recalculate for a probability of being sick of 3%, $P(S) = 0.03$

	Actually sick (S)	Actually not Sick (!S)	Sum
tested positive	240 TP	97 FP	337
tested negative	60 FN	9.603 TN	9.663
Sum	300	9.700	10.000

$$P(S | +) = \frac{240}{240 + 97} = \frac{0.03 \cdot 0.8}{0.0337} = 0.712$$

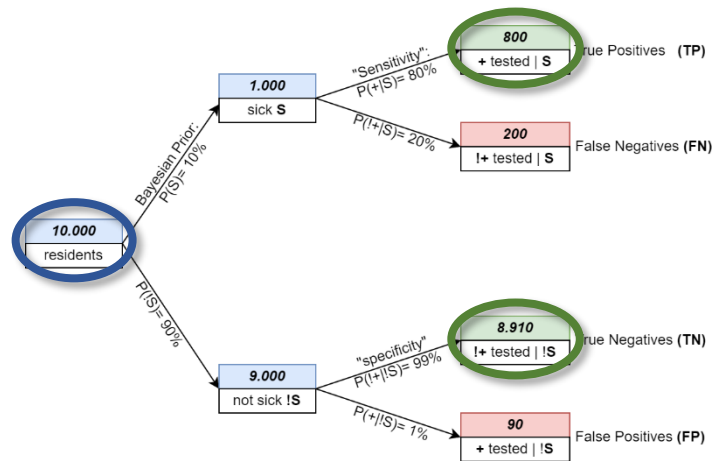
If we compare the three different calculations for the posterior, we see differing expressive power depending on the prior values. The higher the probability of the prior the more expressive power the posterior has.

Part V: Important Terms in Performance Metrics:

Accuracy

$$= \frac{\text{all correct classified items}}{\text{all items}}$$

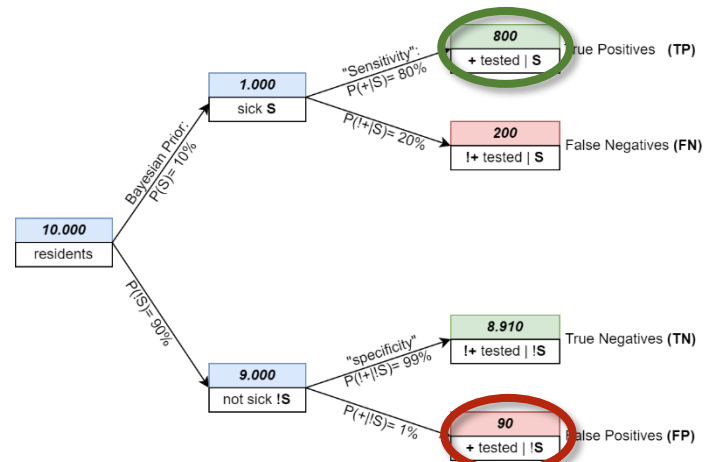
$$= \frac{TP + TN}{TP + TN + FP + FN}$$



Precision

$$= \frac{\text{all correct predicted positive items}}{\text{all positive predicted items}}$$

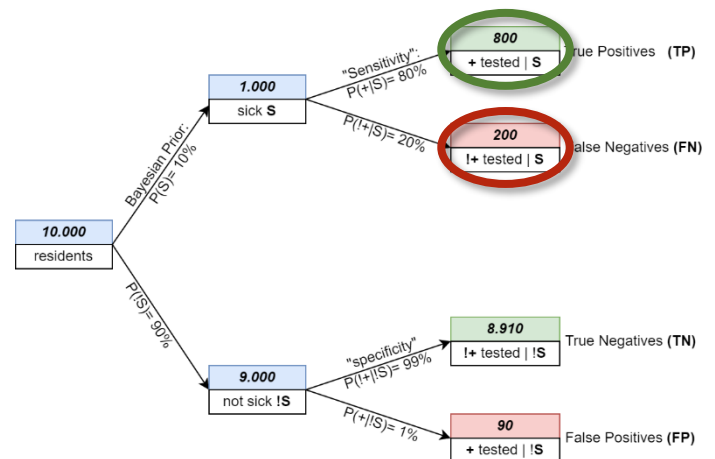
$$= \frac{TP}{TP + FP}$$



Recall (Sensitivity)

$$= \frac{\text{all correct predicted positive items}}{\text{all actual positive items}}$$

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



Part IV: From Bayes-Theorem to Machine-Learning, deriving the Naïve-Bayes Classifier

If we review our simple example in the last lesson, we recognize that the actual outcome is the classification of being sick or not. The prediction is whether or not someone has a positive Corona test result.

Therefore, we call being sick our desired target and the test results our associated feature.

We designate our target with the small letter y and our features with the large letter X , which is a very common and widespread terminology in machine learning. So, we can rewrite our Bayes-Theorem-formula:

$$P(S | +) = \frac{P(+ | S) \cdot P(S)}{P(+)}$$

*"Probability of being sick
given a positive test result."*

$$P(y | X) = \frac{P(X | y) \cdot P(y)}{P(X)}$$


With y as the target designator
and X the feature designator

*"Probability of a specific classification
given a specific feature value."*

Now we can put the naïve bayes classifier to work.

Imagine a police statistic about stolen cars. For each car, the circumstances were recorded and tabulated:

Nr.	Parked	Type	Color	Stolen?
1	on Street	Sport	Red	Yes
2	on Street	Sport	Red	No
3	on Street	Sport	Red	Yes
4	on Street	Sport	Blue	No
5	in Garage	Sport	Blue	Yes
6	in Garage	Compact	Blue	No
7	in Garage	Compact	Red	Yes
8	on Street	Compact	Red	No
9	in Garage	Compact	Red	No
10	in Garage	Sport	Red	Yes



Here we have three features instead of one feature and therefore we need to adapt our formula for the likelihood

$$P(y | X) = \frac{P(X | y) \cdot P(y)}{P(X)}$$

$$P(\text{Stolen} | \text{Parked, Type, Color}) = \frac{P(\text{Parked, Type, Color} | \text{Stolen}) \cdot P(\text{Stolen})}{P(\text{Parked, Type, Color})}$$

We are searching again for the conditional probability on the left side of the equation, the posterior probability:

The probability of a car being stolen given how it was parked, which type it was and which color it had.

And here comes the naïve part of the Naïve Bayes Classifier:

We assume that all features are independent from each other. Which means according to this example:

If a given car is of the type “sports” is completely independent from its color and independent how it was parked. These features can be interpreted as independent as three playing dices thrown simultaneously. So, the probabilities can be rewritten using the chain rule:

$$P(\text{Parked, Type, Color}) = P(\text{Parked}) \cdot P(\text{Type}) \cdot P(\text{Color})$$

$$P(\text{Parked, Type, Color}|\text{Stolen}) = P(\text{Parked}|\text{Stolen}) \cdot P(\text{Type}|\text{Stolen}) \cdot P(\text{Color}|\text{Stolen})$$

Color	Stolen?
Blue	No
Blue	No
Blue	Yes
Red	No
Red	No
Red	No
Red	Yes
Red	Yes
Red	Yes
Red	Yes

	Stolen ("Yes")	Not stolen ("No")	Sum
Red	4	3	7
Blue	1	2	3
Sum	5	5	10

	P(Yes)	P(No)
Red	4/7	3/7
Blue	1/3	2/3

Type	Stolen?
Compact	No
Compact	No
Compact	No
Compact	Yes
Sport	No
Sport	No
Sport	Yes
Sport	Yes
Sport	Yes
Sport	Yes

	Stolen ("Yes")	Not stolen ("No")	Sum
Compact	1	3	4
Sport	4	2	6
Sum	5	5	

	P(Yes)	P(No)
Compact	1/4	3/4
Sport	4/6	2/6

Parked	Stolen?
in Garage	No
in Garage	No
in Garage	Yes
in Garage	Yes
in Garage	Yes
on Street	No
on Street	No
on Street	No
on Street	Yes
on Street	Yes

	Stolen ("Yes")	Not stolen ("No")	Sum
Garage	3	2	5
Street	2	3	5
Sum	5	5	

	P(Yes)	P(No)
Garage	3/5	2/5
Street	2/5	3/5

Before we dive in some calculations, we can simplify the formula once again:

$P(\text{Parked})$, $P(\text{Type})$ and $P(\text{Color})$ don't change, these are constants. Therefore, we can neglect them:

$$P(\text{Parked}) \cdot P(\text{Type}) \cdot P(\text{Color}) = \text{constant}$$

Our Formula changes from...

$$P(\text{Stolen} | \text{Parked, Type, Color}) = \frac{P(\text{Parked, Type, Color} | \text{Stolen}) \cdot P(\text{Stolen})}{P(\text{Parked}) \cdot P(\text{Type}) \cdot P(\text{Color})}$$

...to:

$$P(\text{Stolen} | \text{Parked, Type, Color}) \propto P(\text{Parked, Type, Color} | \text{Stolen}) \cdot P(\text{Stolen})$$

Calculate the probability if a car is stolen or not given red color, compact type and parked on the street:

$$P(\text{Yes} | \text{Street, Compact, Red}) = P(\text{Street} | \text{Yes}) \cdot P(\text{Compact} | \text{Yes}) \cdot P(\text{Red} | \text{Yes}) \cdot P(\text{Yes})$$

$$P(\text{Yes} | \text{Street, Compact, Red}) = \frac{2}{5} \cdot \frac{1}{4} \cdot \frac{4}{7} \cdot \frac{1}{2} = \frac{8}{280}$$

This must be compared to:

$$P(\text{No} | \text{Street, Compact, Red}) = P(\text{Street} | \text{No}) \cdot P(\text{Compact} | \text{No}) \cdot P(\text{Red} | \text{No}) \cdot P(\text{No})$$

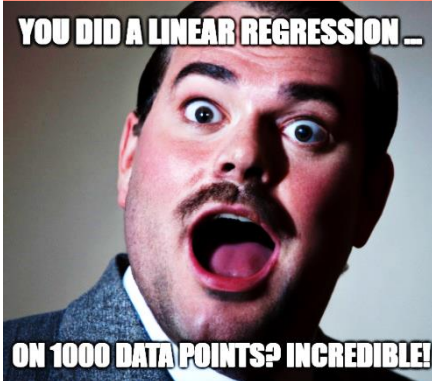
$$P(\text{No} | \text{Street, Compact, Red}) = \frac{3}{5} \cdot \frac{3}{4} \cdot \frac{3}{7} \cdot \frac{1}{2} = \frac{27}{280}$$

Since $27/280 > 8/280$ we classify our car given this specific feature set as NOT stolen.

Lesson 11: Linear Regression

What students should learn?

The second Part of the 'Big Three' Classification, Clustering and Regression: Data analysis of correlated data. Continuous Prediction by using inter- and extrapolations. Therefore, students must do a "trendline" inside a table calculation software like Microsoft Excel, LibreOffice Calc or Google Tables. The quality of the prediction can be estimated using the so-called correlation coefficient.



Possible students' activities and tasks:

When will more ice cream be sold? When it is warm. Can you formulate a The higher the temperature the more ice is sold theorem from this? The data shed light on the hypothesis: a linear regression with a high correlation coefficient means that the model can be used well for prediction. In this context, what do correlation coefficient, residuals, sum of least squares, y-axis intercept, and slope mean? Students work through the basic concepts using a simple ice cream vendor example, learn the mathematical interpretation using a simulation, and later apply it in a small project. There, the meaning of the correlation coefficients can be seen. For the older students, the gradient descent method can be explained; the mathematical basics should be available in grades 10 and up.

Part I: The Story

An ice cream vendor wants to predict the probable sales at the next weekend. The only thing he knows is the weather forecast: it will be a sunny afternoon and the temperature will probably be 13°C on Saturday and 21°C on Sunday. Please help him to organize and plan this weekend. He has a table with values from the past weeks:

temperature (°C)	number of ice cream sales
12	180
14	220
15	330
16	325
17	400
18	425
20	480
13	?
21	?



Part II: Modeling inside GoogleDocs / Microsoft Excel / LibreOffice Calc etc.

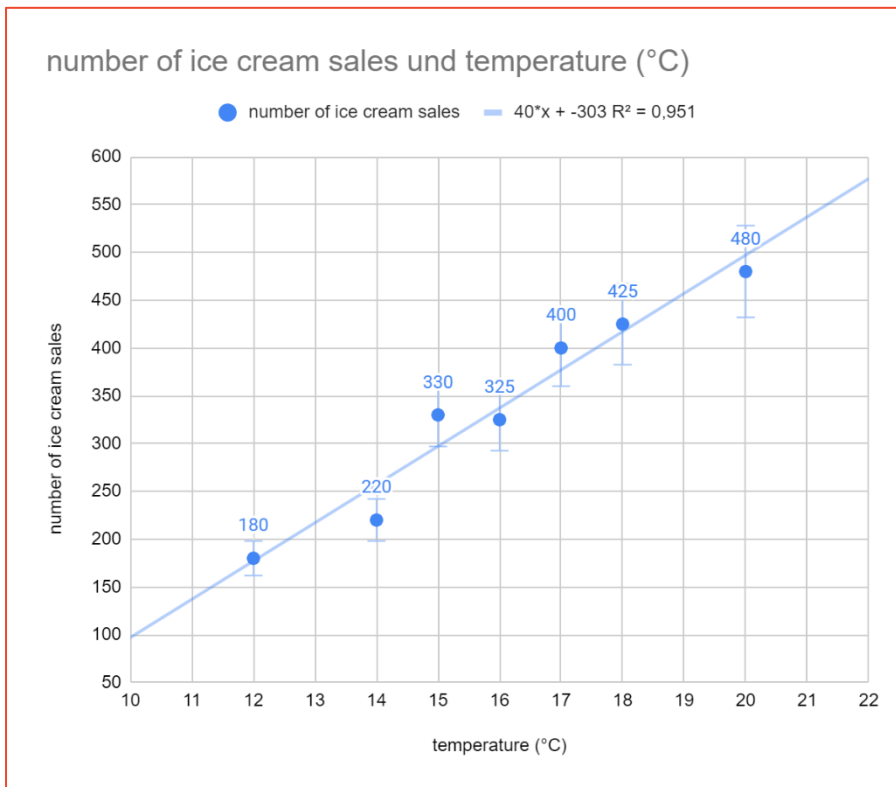


Figure 11: The scatterplot made in Google Tables. The regression calculation is already done, and the equation is shown. This is the starting point for the exercises.

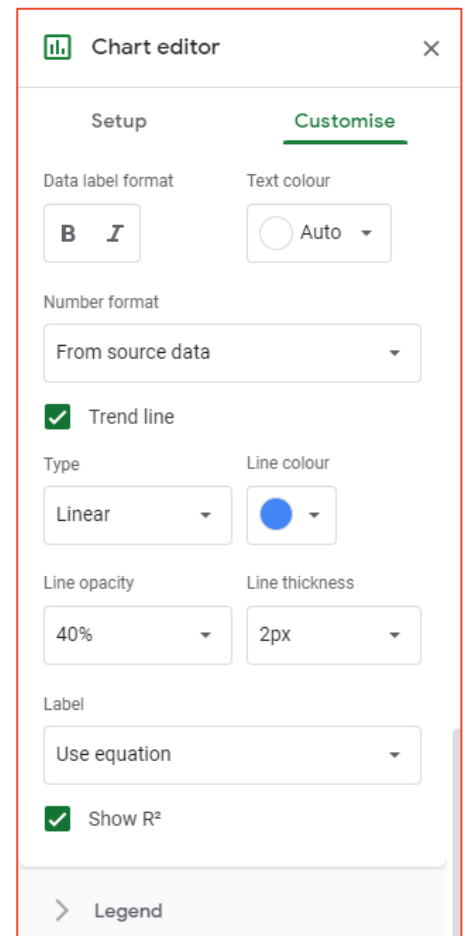


Figure 12: Necessary settings for calculating the regression equation. First you have to build a scatter plot from the data table. Then you have to add a trendline to the plot. At last, you need to format the trendline for displaying equation and squared correlation coefficient.

Regression function:

$$y = 40 \cdot x - 303$$

Part III: Exercises:

Predict the probable icecream sales at the following temperature:

- a) 0°C *Answer: -303 pieces of icecream, does not make sense*
- b) 10°C *Answer: 400 – 303 = 97 pieces of icecream*
- c) 13 and 21°C *Answer: 13°C: 217 icecream | 21°C: 537 icecream*

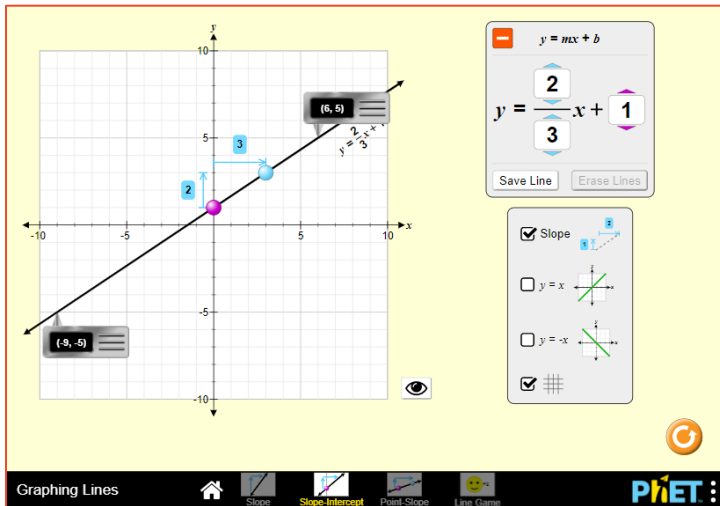
Calculate a **residue** of your own choice. For example, at 15°C:

$$y_{\text{Predicted}} = 40 \cdot 15 - 303 = 600 - 303 = 297$$

$$y_{\text{measured}} = 330$$

$$\text{Residue}_{x=15} = 330 - 297 = 33$$

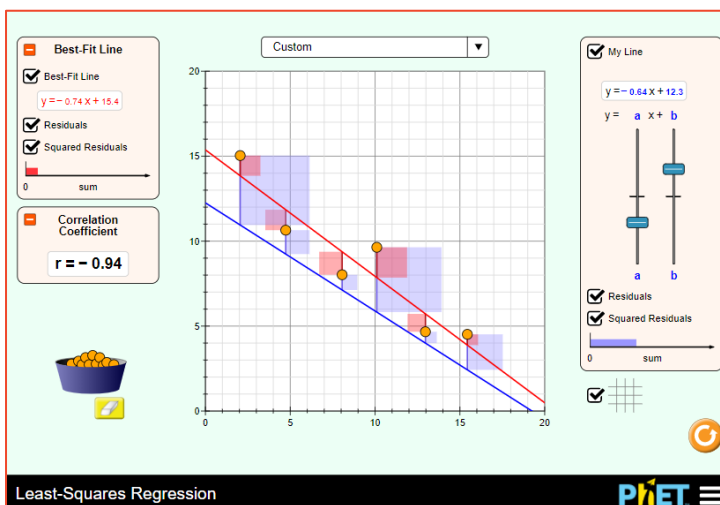
Part IV: How to simulate linear regression. Phet and Gradient Descent



Linear equations

https://phet.colorado.edu/sims/html/graphing-lines/latest/graphing-lines_en.html

If it seems necessary to repeat or deepen the basics of linear equations, you can start with this simulation. The basics of Y-axis intercept and slope are thereby interactively worked out and practiced. An interesting puzzle ("Line Game") rounds off the simulation.



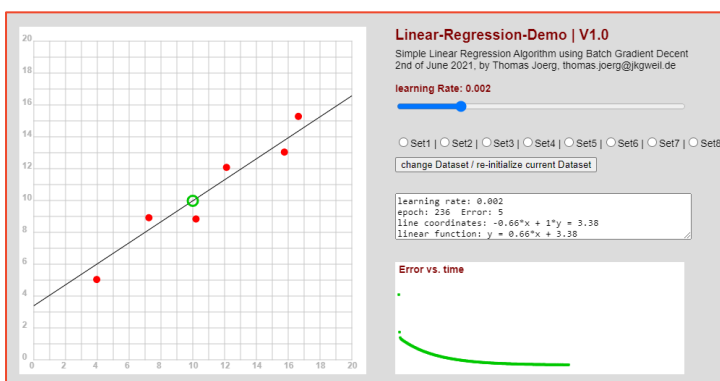
Least squares regression

<https://phet.colorado.edu/en/simulation/least-squares-regression>

Using many built-in data sets, the basics of regression can be worked out, such as:

least sum of squares,
residuals and squared residuals (they are visualized!)
correlation coefficient (including the sign)

The possibility to enter your own data sets and to estimate your own straight lines makes the application enormously versatile



Gradient Descent Linear Regression

<https://iludis.de/LinearRegression/index.html>

The difference between classical linear regression in statistics and ML consists of two components:

1. in statistics one looks for correlations, in machine learning one looks for predictions.
2. statistical mathematics works analytically, i.e., the regression equations are calculated with formulas.

In machine learning, one uses optimization algorithms, such as the gradient descent method. Here, a straight line is first determined at random, and an error function is calculated for the data points. The gradient descent uses the residual squares: The formula for the calculation is derived - the gradient is determined from it. If one wants to optimize the straight-line equation further, one must change the parameters in the direction of the negative gradient. From a mathematical point of view, one runs in exactly that direction in which the residue squares decrease. And this is exactly how one iteratively arrives at the best possible straight line. This is therefore not calculated analytically but optimized iteratively: This is the principle of gradient descent, which is applied again and again in many other areas of machine learning.

Part V (Optional) Derivation of the gradient descent in linear regression

1. Cost function KF() in linear regression: KF() is the mean of all residual squares
2. Memory Mean: If 'n' values exist, must be divided by 'n'
3. Residuum: The difference between data value and regression value ($y = mx + c$)
4. The cost function forms a parabola, and this parabola can be derived.
5. The optimization is carried out by adjusting the slope (m) and Y-axis sections (c).
6. Therefore, it must be derived according to these two components (higher mathematics with so-called partial **derivatives, does not have to be understood**)

$$KF() = \frac{1}{n} \sum_{i=1}^n (y_{i, \text{DataPoint}} - y_{i, \text{RegressionLine}})^2$$

$$KF() = \frac{1}{n} \sum_{i=1}^n (y_{i, \text{DataPoint}} - (mx_i + c))^2$$

Multiplying out, $y_{\text{DataPoint}}$ is now y, Indices, 'i' will be neglected:

$$\begin{aligned} (y_{\text{DataPoint}} - y_{\text{RegressionLine}})^2 &= \\ (y - (mx + c))^2 &= \\ (y - mx - c)^2 &= \\ (y - mx - c) \cdot (y - mx - c) &= \\ y^2 - ymx - yc - ymx + m^2x^2 + mxc - yc + mxc + c^2 &= \\ y^2 - 2ymx - 2yc + m^2x^2 + 2mxc + c^2 & \end{aligned}$$

deriving according to 'm': $\frac{\text{Derivative KF()}}{\text{according to m}} = -2yx + 2mx^2 + 2xc =$

$$\begin{aligned} -2x \cdot (y - mx - c) &= \\ -2x \cdot (y - (mx + c)) &= \\ -2x \cdot (y_{\text{DataPoint}} - y_{\text{RegressionLine}}) &= \\ -2x \cdot \text{ResidualValue} & \end{aligned}$$

deriving according to 'c': $\frac{\text{Derivative KF()}}{\text{according to c}} = -2y + 2mx + 2c =$

$$\begin{aligned} -2 \cdot (y - mx - c) &= \\ -2 \cdot (y - (mx + c)) &= \\ -2 \cdot \text{ResidualValue} & \end{aligned}$$

Iteratively, the LearningRate is then optimized according to the following formula:

$\begin{aligned} c_{\text{new}} &= c_{\text{old}} - 2 \cdot \text{ResidualValue} \cdot \text{LearningRate} \\ m_{\text{new}} &= m_{\text{old}} - 2 \cdot x_i \cdot \text{ResidualValue} \cdot \text{LearningRate} \end{aligned}$
--

Some Formulas for linear regression calculations

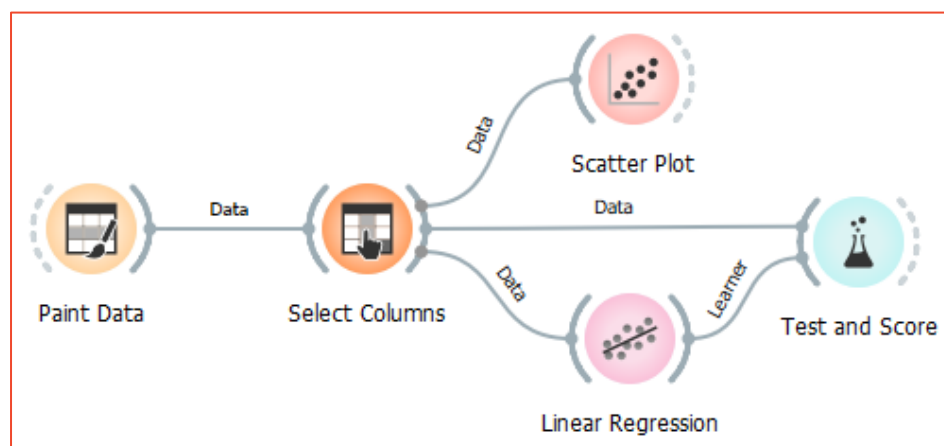
(teachers' secret knowledge, students do not need to know)

Mean value of x: \bar{x} [here: the temperature] $\bar{x} = \frac{1}{n} \sum_{i=1}^n x_i = 16$	Mean value of y: \bar{y} [here: the icecream sales] $\bar{y} = \frac{1}{n} \sum_{i=1}^n y_i = 337,14$
Standard deviation of x: s_x $s_x = \sqrt{\frac{1}{n} \sum_{i=1}^n (x_i - \bar{x})^2} = 2,45$	Standard deviation of y: s_y $s_y = \sqrt{\frac{1}{n} \sum_{i=1}^n (y_i - \bar{y})^2} = 100,49$
Covariance: Cov_{xy} $Cov_{xy} = \sqrt{\frac{1}{n} \sum_{i=1}^n (x_i - \bar{x}) \cdot (y_i - \bar{y})} = 240$	Correlation R_{xy} = standardized covariance Cov_{xy} $R_{xy} = \frac{Cov_{xy}}{s_x \cdot s_y} = 0,97$
Regression coefficients (a: coefficient, b: constant) $y = a \cdot x + b$ $a = \frac{S_y}{S_x} \cdot R_{xy} = 40 \quad b = -\frac{S_y}{S_x} \cdot R_{xy} \cdot \bar{x} + \bar{y} = -303$	Coefficient of determination (R squared): $R^2 = (R_{xy})^2 = 0,951$

Part VI: Interpretation of the correlation coefficient in Orange Data Mining

The better the datapoints fit to the regression line:

- The smaller the sum of squared residuals
- The nearer the correlation coefficient to a value of +1 for a positive slope or -1 for a negative slope

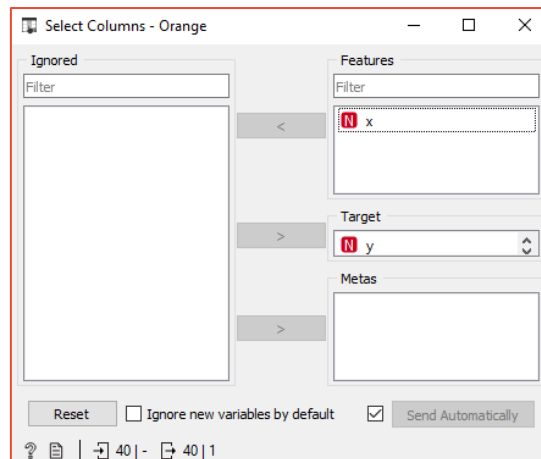
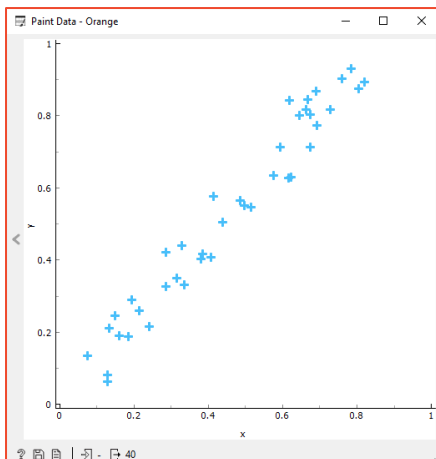


The Complete Setup

of the Project: A scatterplot for visualizing the regression line, a Linear Regression Node together with a Test and Score node for performance metrics.

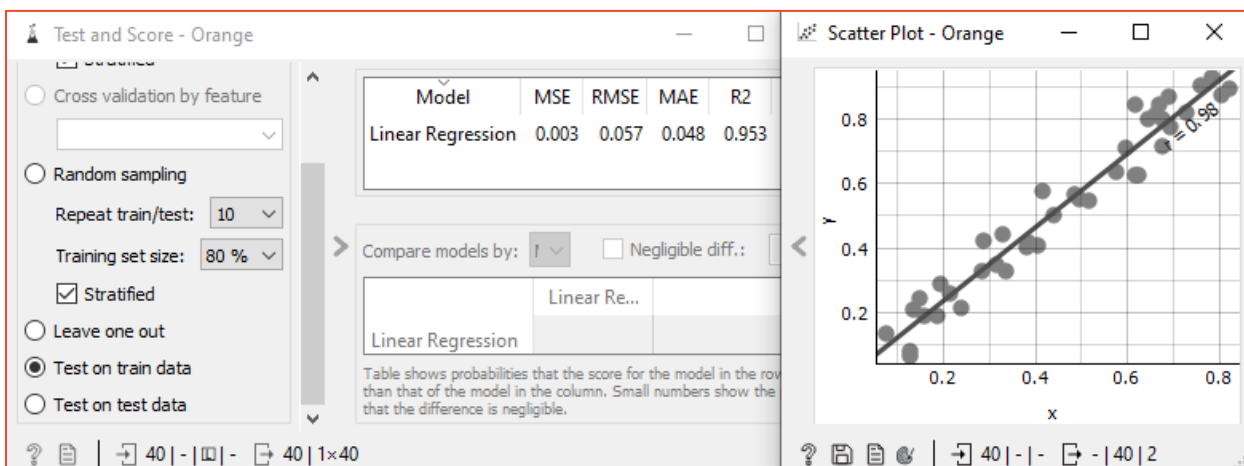
Explaining the Setup and the nodes:

Paint Data and Select Columns: Preparing the test-dataset

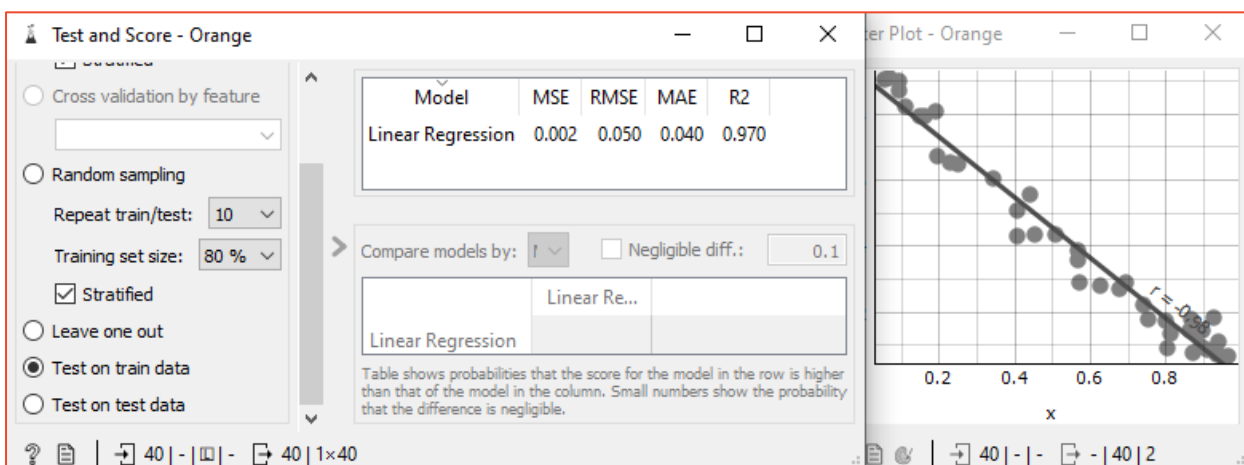


Some synthetic data is painted since we only want to understand the meaning of the value of the correlation coefficient.

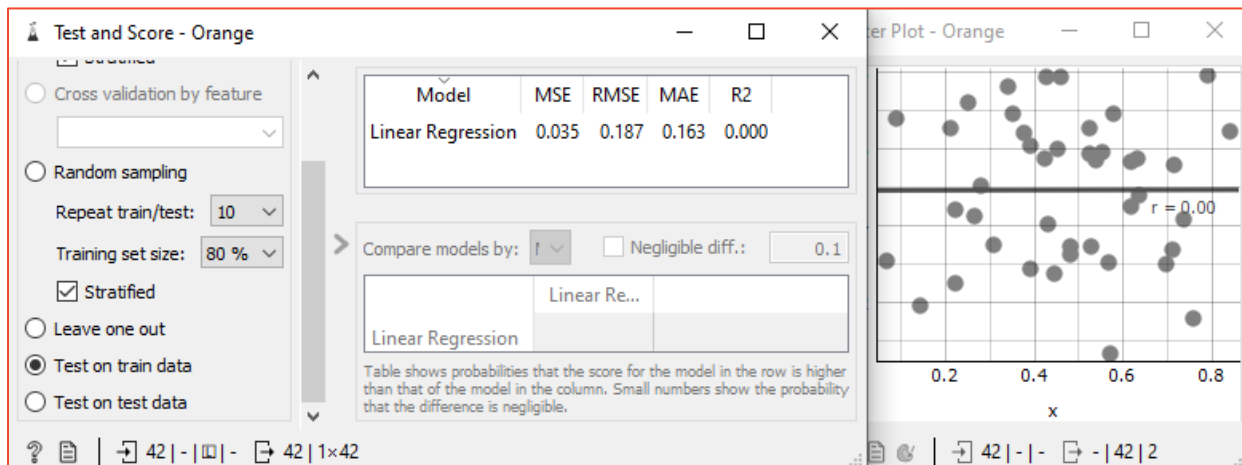
Variation 1: Strong positive correlation of ca. +1 (+0.98)



Variation 2: Strong negative correlation of ca. -1 (-0.98)

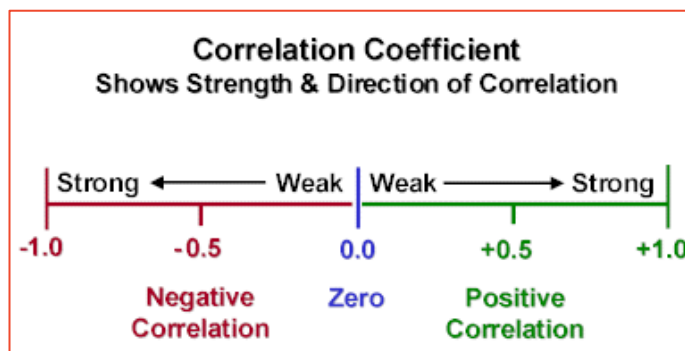


Variation 3: No correlation of ca. 0

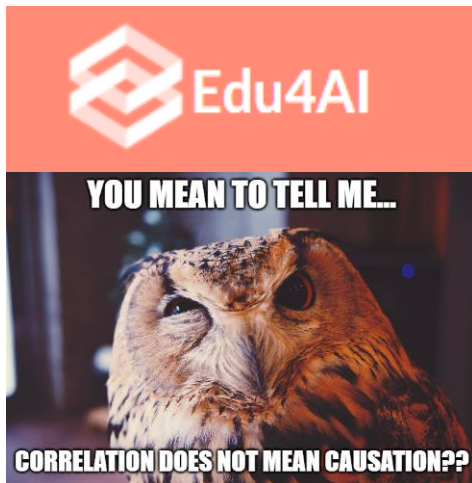


Summary:

Correlation coefficients can vary between the values -1 and +1. A value that is close to 1 in terms of magnitude means a strong correlation between x and y values. A positive sign here means a positive slope, correspondingly complementary is the negative sign. A value close to 0 means that the variables do not correlate with each other.



Lesson 12: Distinguish correlation and causation!



13 Photo by [Joe Green](#) on [Unsplash](#)

What students should learn:

Sometimes you see connections where there are none, and that can be misleading. Here are some examples: Is there a connection between the weather and the river level? Yes, certainly, because it is to be expected that with stronger rainfalls the river levels rise! Is there a connection between the weather and the size of your shoes? Probably not!

Of course, it could be that there is an apparent correlation between your shoe size and, for example, the average annual temperature. After all, we are living in times of climate change, and as your feet grow, the annual temperature may also grow. But this is only a so-called "correlation": Correlation is a measure for relationship between variables. First of all, it says nothing about whether it is a cause-and-effect principle.

Possible students' activities and tasks:

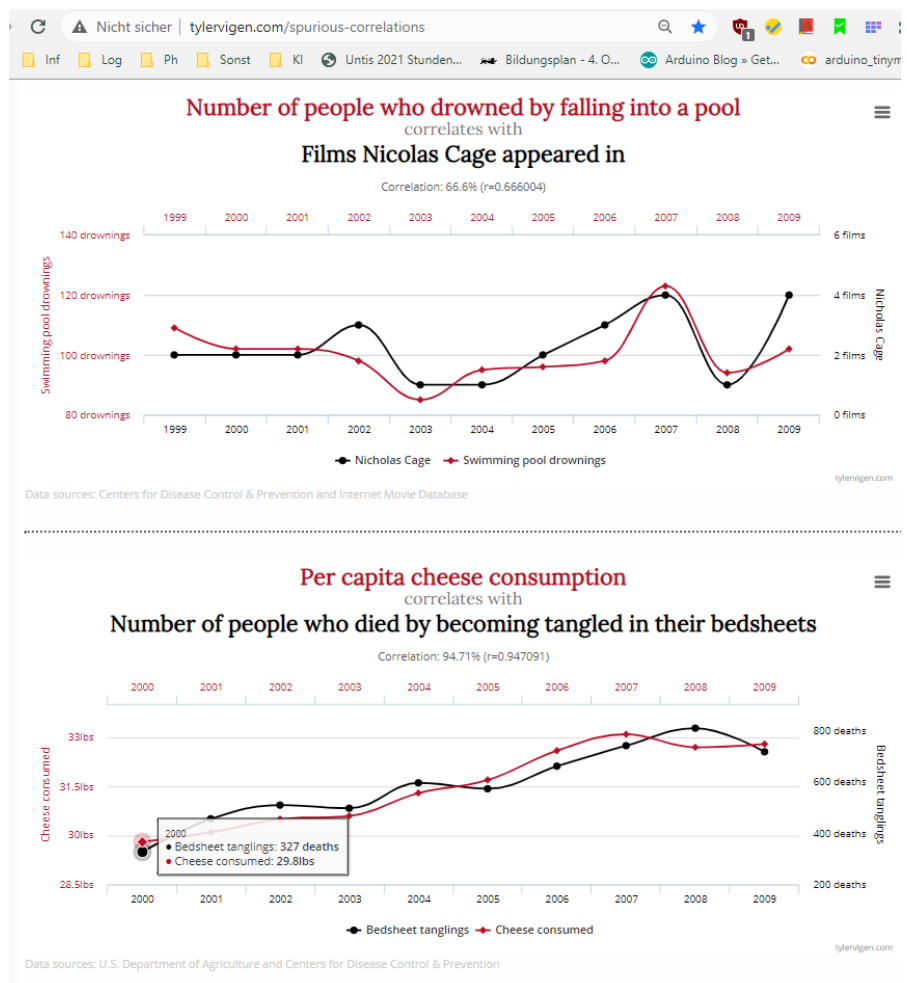
Have students each pick an example from the website and describe why you might be fooled by the diagrams shown here. In this context, the term "**Cum hoc ergo propter hoc**" can also be researched and incorporated. Bogus causalities can reach into the realm of superstition. What gives rise to superstition?

If different variables are dependently related, then we speak of "causality". Shoe size, for example, is causally related to body size - and this is easy to see.

Let's have a look at the website of Tyler Vigen:

<https://tylervigen.com/spurious-correlations>

He is a mathematics professor with a funny hobby: He finds correlations – i.e., traits that seem to be related but do not have a justifiable link.



Lesson 13: k-Means Clustering.

Or: How to handle with unlabeled data?

What students should learn:

When we must handle unlabeled data, unsupervised ml-algorithms are sometimes able to find agglomerations of datapoints and therefore can structure our data by building clusters. The k-means-clustering algorithm is a well-known example of these learning algorithms. But how can we determine how many clusters are appropriate - how to determine the 'k'-value? We can use an elbow plot and find the characteristic kink in the plotline!

Possible students' activities and tasks:

Using an intuitive problem where the solution is obvious, the concept of clustering is introduced. The algorithm is elaborated by observing a step-by-step simulation. Subsequently, using a one-dimensional mealtime scenario, the elaborated algorithm is applied by the students, and the clustering is calculated by hand once. A small project in Orange Data Mining with synthetic, self-painted data rounds off the lesson.

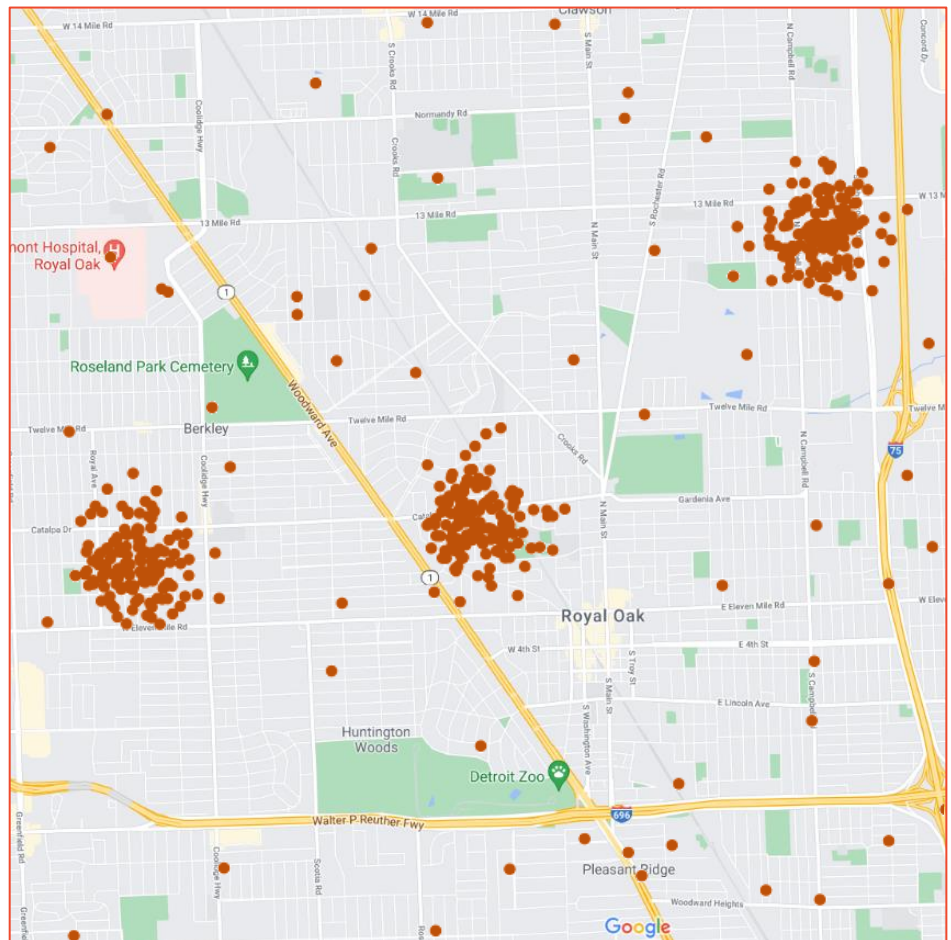
Part I: The story

Imagine you want to build up a pizza delivery service with a small number of stores in your hometown. Since you must deliver your pizzas, you try to keep the ways for the pizza drivers as short as possible.

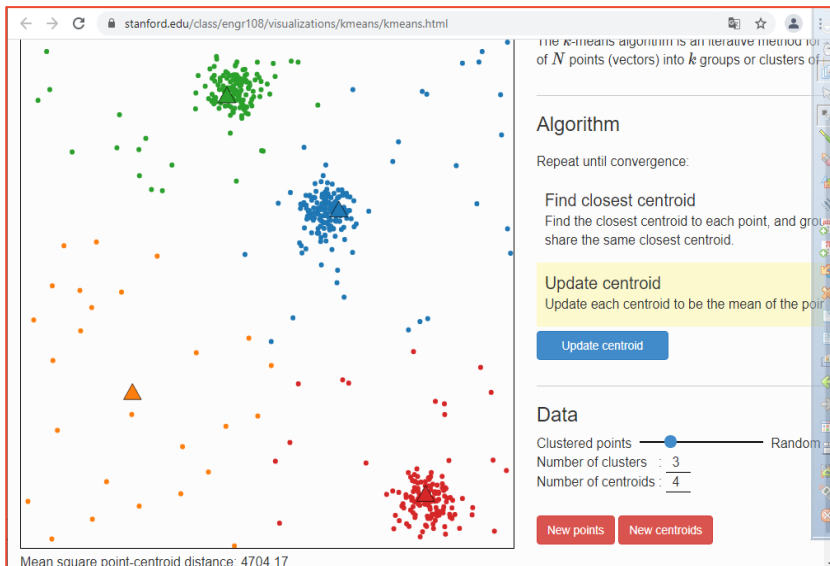
How many stores do you need and where to place them to achieve your goals? All you have is a map of your customers and the locations from which they ordered pizzas.



Figure 14: Imagine the contribution of customers; every point on the map corresponds to a pizza order in the past.

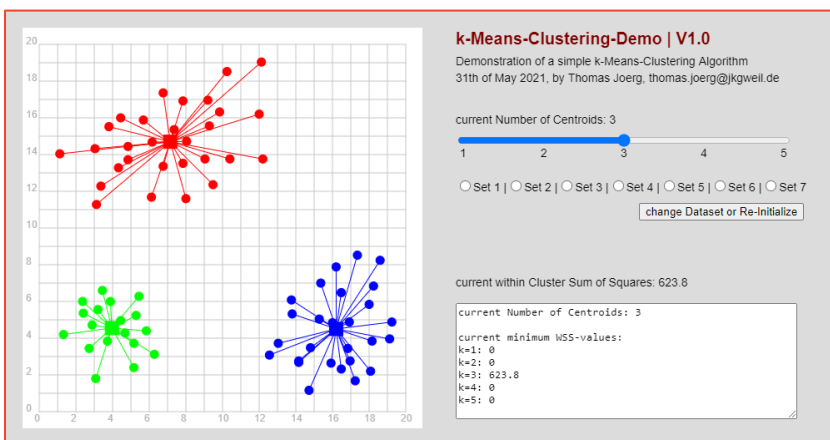


Part II: Visualising the algorithm



<https://github.com/karanveerm/kmeans>

This simulation is hosted on github, since it is not present on an internet-website any more. But you can download the app and start it using the index.html



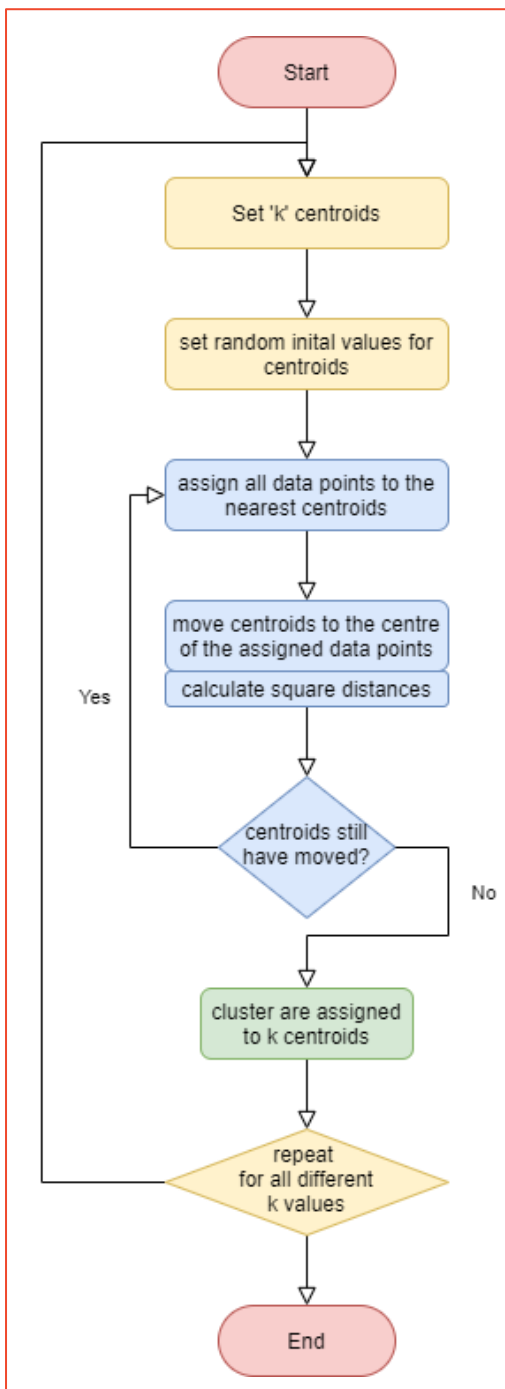
<https://iludis.de/>

[kMeansClustering/index.html](https://iludis.de/kMeansClustering/index.html)

This simulation contains different datasets and will calculate the Sum of squares for each configuration. These values can be copied to an excel-sheet for visualizing the elbow-plot.

Steps for the simulation

- Take as a starting number of clusters 3.
- Start with 3 Centroids and try to converge them to the center of the data point blobs.
- Like many other machine learning-algorithms, you must repeat the procedure many times with different, randomly set starting values.
- The smallest mean-square value you find is the best:
 - a) Write down the algorithm as a flowchart or simply text based
 - b) write down the best value of the mean square distance.



Part III: Formulating the algorithm

The flowchart diagram for the k-means clustering algorithm. Linguistically, it can be formulated a bit simpler like this:

a) Start with an arbitrary number of centroids.

b) Distribute the Centroids randomly between the points.

c) Calculate all distances between all points and the centroids.

d) Assign the points to the closest Centroid.

e) Now move the Centroids to the center of its assigned points.

f) Repeat from point c) until the Centroids stop moving.

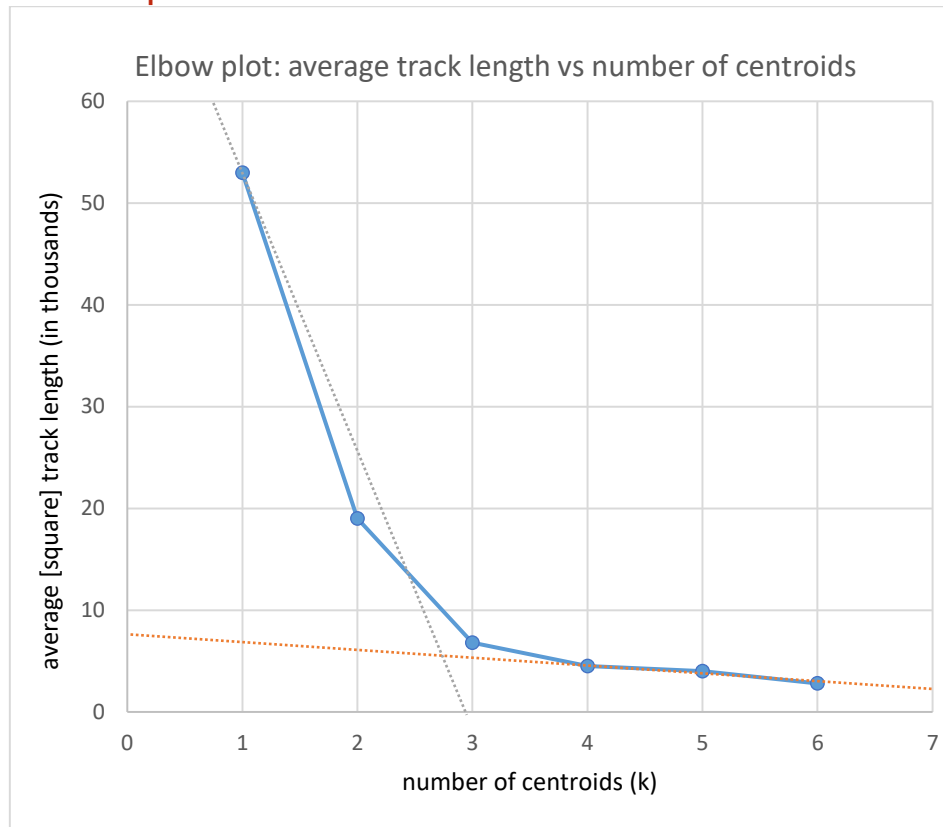
e) Start again at point b) by changing the number of Centroids.

Part IV: Steps for deriving the elbow plot

'k' Centroids	Mean square point-centroid distance
1	53000
2	19000
3	6800
4	4500
5	4000
6	2800

If you collect all optimal distances for all different centroid numbers, you will get a table similar to this. Make an Excel-Sheet with these values to build a diagram:

Part V: Elbow plot and interpretation



at the characteristic bend of the graph, you will find the optimal number of centroids, since there is no significant further change in the mean squared distances. And if there is no more substantial change in these values, then we can do without further centroids.

Part IV: k-Means clustering unplugged.

in the first learning unit, the students have developed an idea of how the algorithm works. To deepen this model idea, they will now run through a simple computational example. Therefore, the teacher picks 3 students and asks them about their habits:

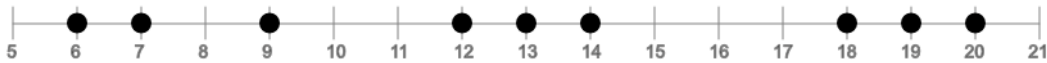
- what time do you have breakfast?
- what time is your lunch,
- and what time is your dinner?

A table like this one will be created, which will be our new dataset for clustering:

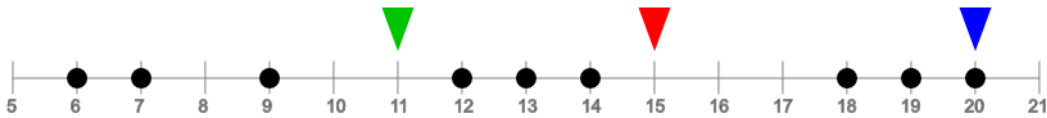
Student	Breakfast time	Lunch time	Dinner time
1	9	14	19
2	7	13	20
3	6	12	18

Complete visualization of the iterations:

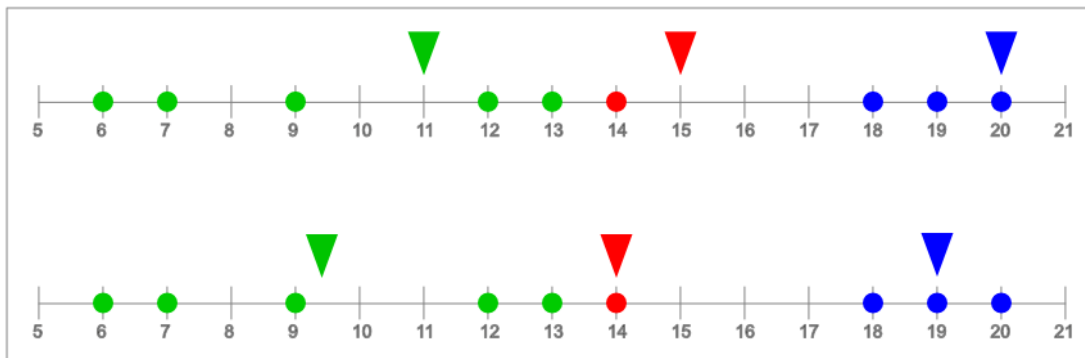
All data is transferred to a number line:



Number line with data points



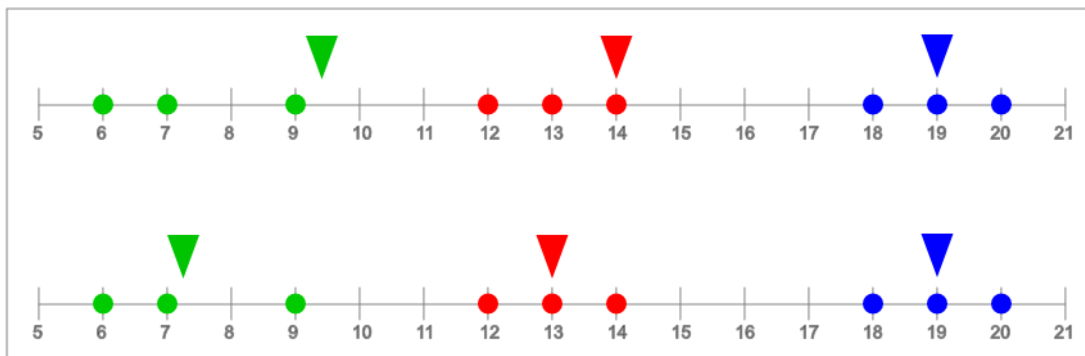
In this example, three centroids are randomly set.



Iteration 1:

Data points are assigned to the nearest centroid.

The centroids are moved in the middle of the data points.



Iteration 2:

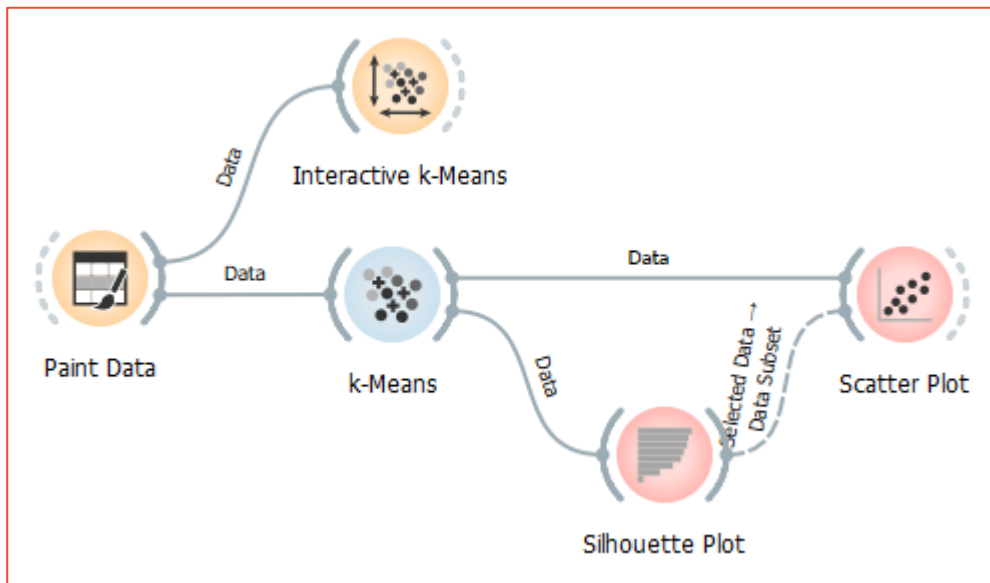
Data points are assigned to the nearest centroid.

The centroids are moved in the middle of the data points.

In iteration 1, the data point at value “13” is set to green. This assignment was randomly chosen – it could also have been of red colour, since the distances to both centroids are equal.

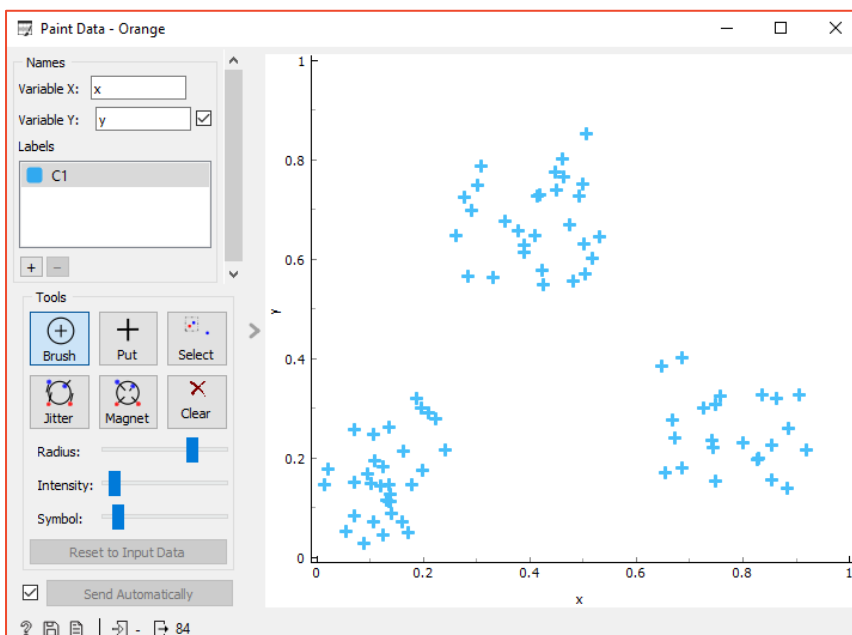
Students can also try to use 2 or 4 different centroids. The number of three for the initial calculation is chosen since the expected outcome is obvious for the students: it can be clearly seen that the algorithm is producing reasonable and robust results.

Part IIV: k-Means-Clustering with Orange Data Mining



The Complete Setup
of the Project: Simple and straight forward can it be used for an introductory lesson for Orange Data Mining.

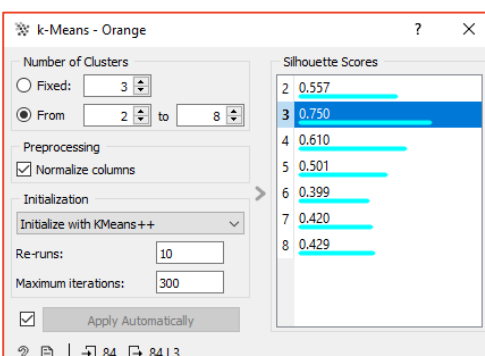
Short explanations for the nodes:



Paint Data:

Datapoints can be drawn randomly by using the brush. These datapoints are stored as values in an associated data-table and can be used for calculations and manipulations.

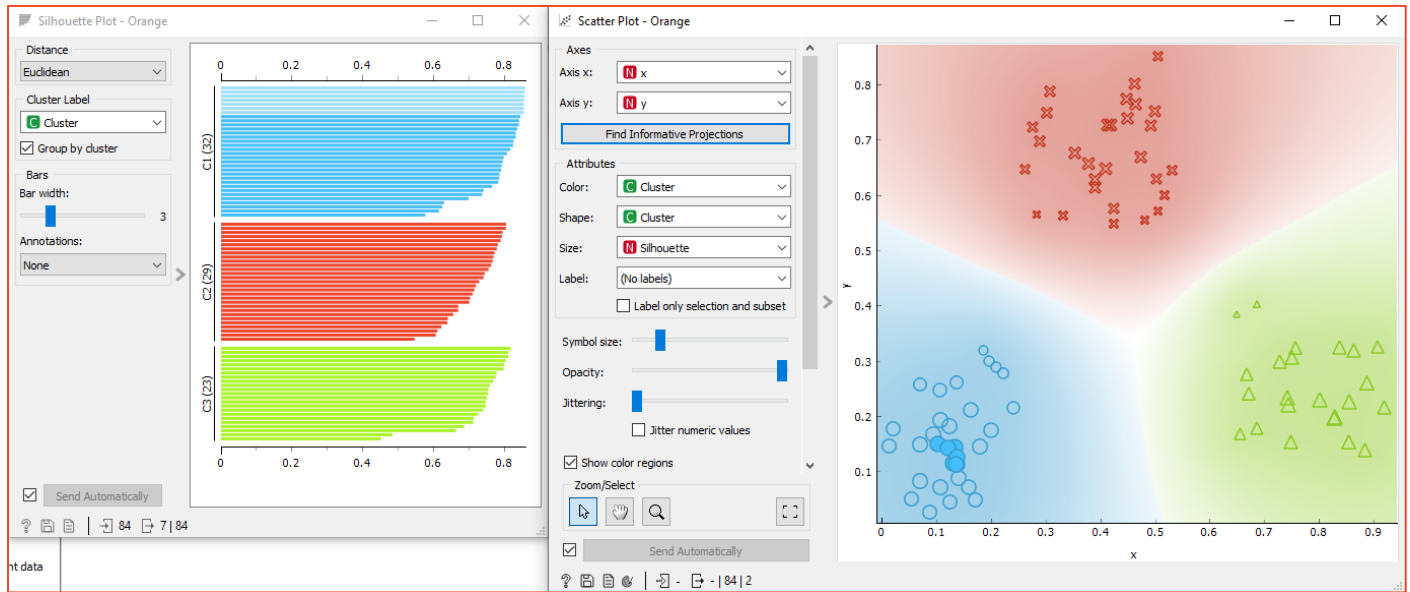
It is a fast, straightforward way for building synthetic data.



k-Means:

To iterate through a range of k-values, choose the second option "from". A calculation for silhouette scores will be done. In short: the higher the silhouette score the better the data is fitted. In this example it is obvious that three clusters will fit the best and therefore it also gets the highest silhouette score.

Shilouette Plot and Scatter Plot



In the scatter Plot the classified data can be visualized. It corresponds tightly with the shilouette plot. This plot shows the correlation between the (invisible) centroid and the individual data point: The bigger the correlation the higher the score – and the larger the bar-length in the bar-graph of the datapoint. If a datapoint is selected in the shilouette plot it will also be selected in the scatter plot!

Add-On, nice to have: The interactive k-Means-Node

Options Help

Settings

Reset Widget Settings...

Add-ons...

Installer - Orange

Filter...

	Name	Version	Action
<input type="checkbox"/>	Associate	1.1.9	
<input type="checkbox"/>	Bioinformatics	4.5.0	
<input checked="" type="checkbox"/>	Educational	0.5.0	

Orange3 Educational

OK Cancel

Interactive k-Means - Orange

Recompute Centroids Step Back Randomize Run Simulation

84 84 | 3

Lesson 14: k-nearest neighbour



Photo by [Phakphoom Srinorajan](#) on [Unsplash](#)

Or: can we develop a kNN from scratch?

What students should learn

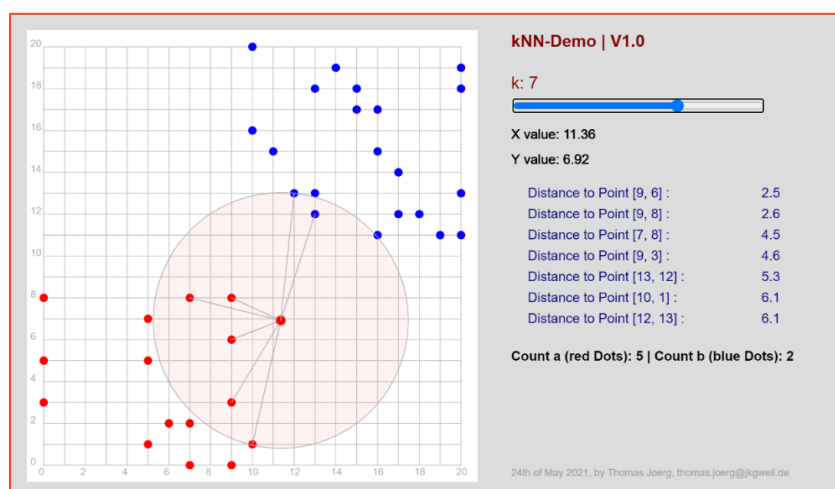
What does an implemented ML algorithm look like "under the hood"? So far, ready-made libraries have been used, and they look like the infamous "black box". In this lesson, one of the simplest ML algorithms will be implemented - here in Python, so that the well-known plotting libraries (e.g., Matplotlib) can be used. However, many other programming languages are conceivable.

Possible students' activities

Students learn about the algorithmic principle of the kNN classifier through a visualization. Afterwards, they first formulate the algorithm linguistically and write it down. Using a very simple example of three meal times per day, they apply the principle to a 1-dimensional problem. Only now can they gradually build up the implementation: The data structure of the array is introduced and practiced using simple examples. Then the programmable control structure of the loop is introduced, with which one can access all elements of an array. When both basic concepts are understood, the minimum search is worked out and programmed as an implementable algorithm. Only now can the 2D array be constructed and practiced as a suitable data structure for the Essens model already worked out. A randomly chosen time point for an unclassified meal is determined and classified using the already implied algorithm. In the process, students will encounter different possible solutions, all of which - if they produce correct results - will be appreciated and discussed. Students thus realize that there is not only one way to solve a programming problem.

Part I: Interactive demo of kNN

<https://editor.p5js.org/iludis40/full/kZsF04gPx>



Part II: A tricky question for the students: What means training of a kNN model?

Answer: There is no real training.

Only the parameters k and later possibly the method for the distance determination can be adjusted. (Monitoring and validation are done with the usual performance metrics Accuracy, Precision and Recall)

Of course, the feature selection can also be added. But the optimization of hyperparameters as for example in linear regression or the decision tree does not take place.

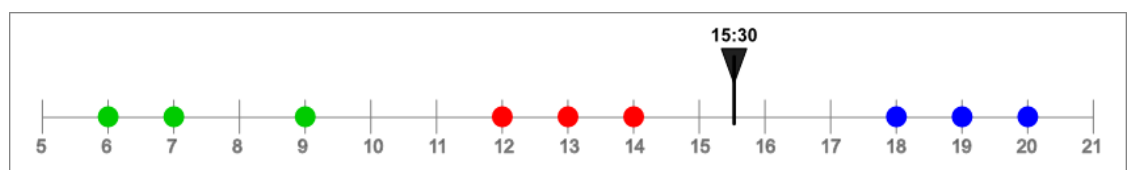
Instead, prediction ALWAYS requires the entire data set in memory, since there is no abstraction from the data.

Part III: kNN unplugged

A) kNN 1D

We start with the full classified meal data from the k-means lesson. Green color is representing breakfast, red codes lunch and the blue dots are representing the lunch times of different students. Now we want to classify the randomly set time for a meal of 15:30. Everybody will immediately recognize that we will probably classify this meal most probably as lunch.

Time	Meal type
6	"B"
7	"B"
9	"B"
12	"L"
13	"L"
14	"L"
18	"D"
19	"D"
20	"D"



Since everything in this task is obvious, we can focus on constructing the algorithm:

1	Determine the distance of our random mealtime to all other points
2	Determine the smallest distance
3	Determine the class of the meal with the smallest distance and return this as result.

Part IV: How should we implement the algorithm in Python?

It is recommended to use Jupyter-Notebooks since programming can be split in single tasks.

The Meal-List can be written as two-dimensional Python-List (2D-Array). We simply have to put every row in a single list of 2 elements and collect all rows together in a containing list. This container is called 'allMeals':

```
## declare and initialize variables
allMeals =
[[6, "B"], [7, "B"], [9, "B"], [12, "L"], [13, "L"], [14, "L"], [18, "D"], [19, "D"], [20, "D"]]
someMeal = 15.5
print(allMeals)
print(allMeals[0])
print(allMeals[0][0])
```

OUTPUT:

```
[[6, 'B'], [7, 'B'], [9, 'B'], [12, 'L'], [13, 'L'], [14, 'L'], [18, 'D'], [19, 'D'], [20, 'D']]

[6, 'B']
6
```

An example for some exercises with the 2D-List to familiarize the students with the calculations:

```
print(allMeals[0][0] + allMeals[4][0]) #Arithmetical sum
print(allMeals[0][1] + allMeals[4][1]) #String-concatenation
```

OUTPUT:

```
19
BL
```

Now we can test to calculate a random distance. But a problem will emerge: How to deal with negative values? Answer: there is no absolute right or wrong; many ways are possible and if these different methods provide correct results, those methods are correct!

```
## calculate a randomly chosen distance:
distance = allMeals[0][0] - someMeal
print(distance)

## case difference
if distance < 0:
    distance = -distance
print(distance)

## absolute value
distance = abs(distance)
print(distance)

## square root from square number
distance = (distance**2)**0.5
print(distance)
```

OUTPUT:

```
-9.5
9.5
9.5
9.5
```

The core algorithm: Minimum Search

```
# minimum-search
### starting with the assumption that the first value could be the lowest:
indexLowest = 0
lowest = abs(allMeals[0][0]-someMeal)

### iterating all other values and compare the
for i in range(0, len(allMeals)):
    distance = abs(allMeals[i][0]-someMeal)
    if(distance<lowest):
        lowest = distance
        indexLowest = i

print("the lowest distance is: ", lowest)
print("it has the index: ", indexLowest)
```

OUTPUT

```
the lowest distance is:  1.5
it has the index:  5
```

The principle of the minimum-search algorithm

it is described in many places in the internet. In short:

- We start at the first list element, which has the index 0: We first assume that the value of this list element is the lowest – as long as we don't know better - and therefore assign this value to the variable "lowest".
- We go through all the list items using the control structure of a loop.
- At each loop pass, this 'lowest' value is compared again and again with the current list value:
- If the current list value is smaller than 'lowest' value, 'lowest' is overwritten with this list value.
- If you have worked through all list elements, you have finally found the smallest value.

```
exampleArray = [7, 12, 2, 5, 9]
lowest = exampleArray[0]

### iterating all other values and compare the
for i in range(0, len(exampleArray)):
    if exampleArray[i] < lowest:
        lowest = exampleArray[i]

print("the lowest value is: ", lowest)
```

OUTPUT

```
the lowest value is:  2
```

The prediction is straightforward: just read and return the class-label of the element with the lowest index:

```
print("The meal-element in the list is: ", allMeals[indexLowest])  
print("The predicted class of someMeal is: ", allMeals[indexLowest][1])
```

OUTPUT

The meal-element in the list is: [14, 'L']

The predicted class of someMeal is: L

Part V: Internal differentiation

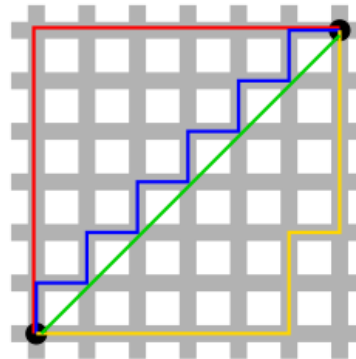
Experience shows that the field of students widens when it comes to programming tasks. Some are faster, some slower. The faster students can be given additional tasks:

- Determine the class membership for $k = 2$ or $k = 3$.
- Determine the class membership in a two-dimensional case.

Here, the Euclidean distance with the Pythagorean theorem can be used or the so-called Manhattan distance, which results from the simple subtraction of x and y values.

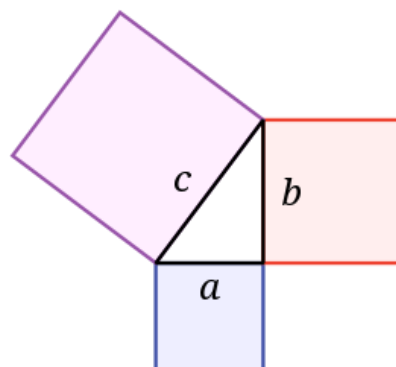
Manhattan-distance:

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



Pythagorean theorem

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

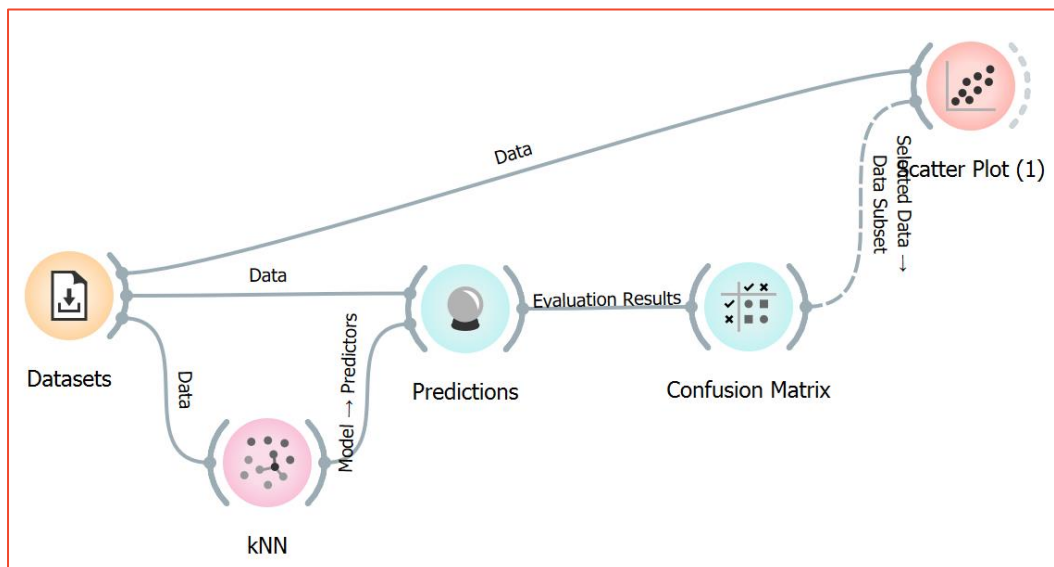
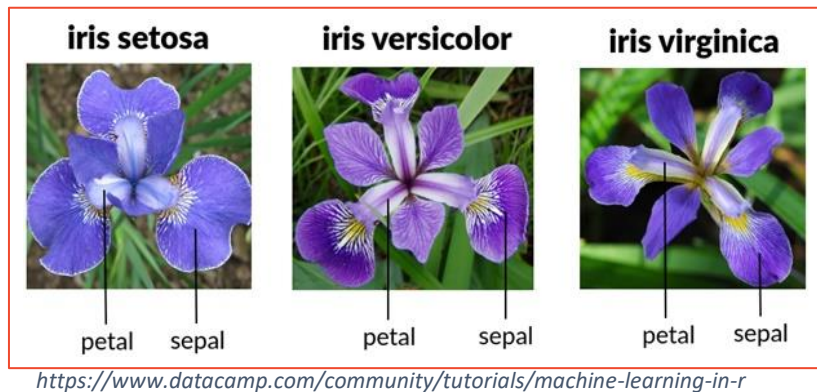


Part IV: kNN with Orange Data Mining

Introducing the Iris Dataset:

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

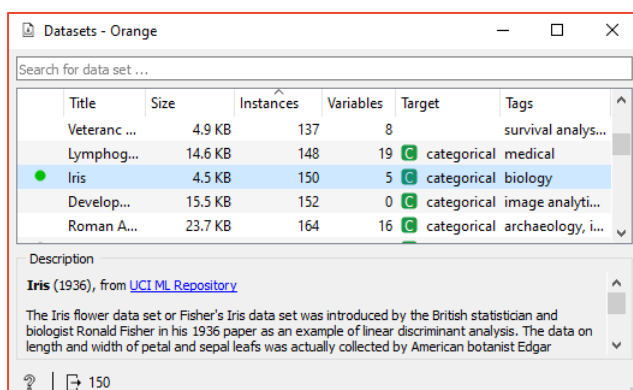
In the year 1936 the mathematician/geneticist Sir Ronald Aylmer Fisher collected data of three different classes of Iris flowers. These data was used for classifying the different species. It is a typical test dataset for ML-algorithms.



The Complete Setup of the Project: It can be used for an introductory lesson for the Iris Dataset.

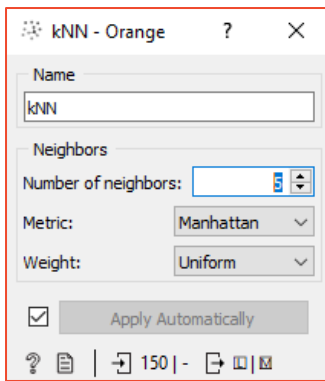
With this setup, many other ML algorithms can be tested and compared. The kNN represents only a starting point here.

Short explanations for the nodes:



Datasets:

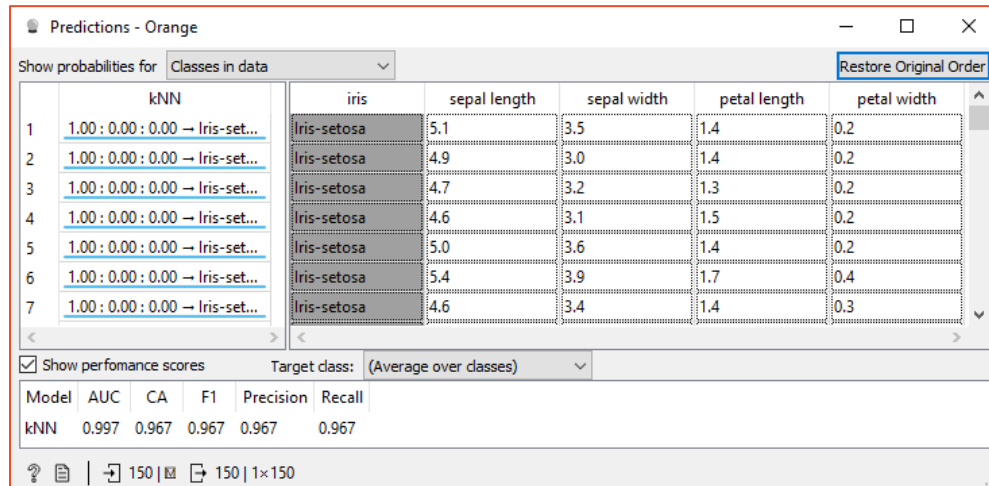
Orange Data Mining comes with a bunch of prebuilt Datasets covering many different aspects of data mining. We will use the Iris flower dataset in this tutorial.



kNN

some different parameters can be chosen for trying to achieve a higher accuracy. For the students it should be sufficient to know the methods of 'Euclidean' and 'Manhattan' distance.

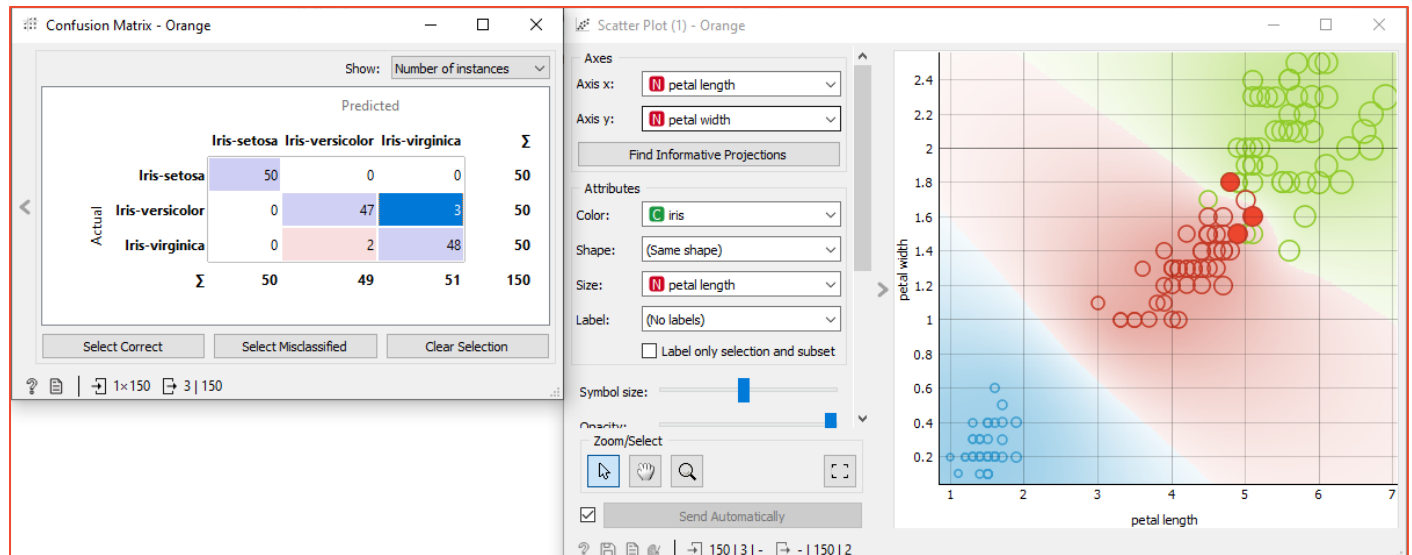
Predictions:



In this process step, the original data labels are discarded and the kNN algorithm is applied to the database. Useful is the display of the performance metric in the lower quarter of the window: CA - which means Accuracy - is at 0.967, the same with Recall and Precision. This indicates that with the selected kNN parameters the result is already quite good.

However, there is one setting where the predictions are 100% correct. Can the students figure out what those are?

Confusion Matrix and Scatter Plot



The scatterplot visualizes the classifications – and some more aspects: If inside the confusion matrix nothing is selected, all data points are not filled. If you select a cell inside the confusion matrix, for example the off-diagonal elements which are misclassified, you can see them as filled dots inside the scatter plot. Now you can fine-tune your model until all off-diagonal matrix-elements will become a zero value and therefore, all data points are well-classified.



Lesson 15: Support Vector Machines

What students should learn

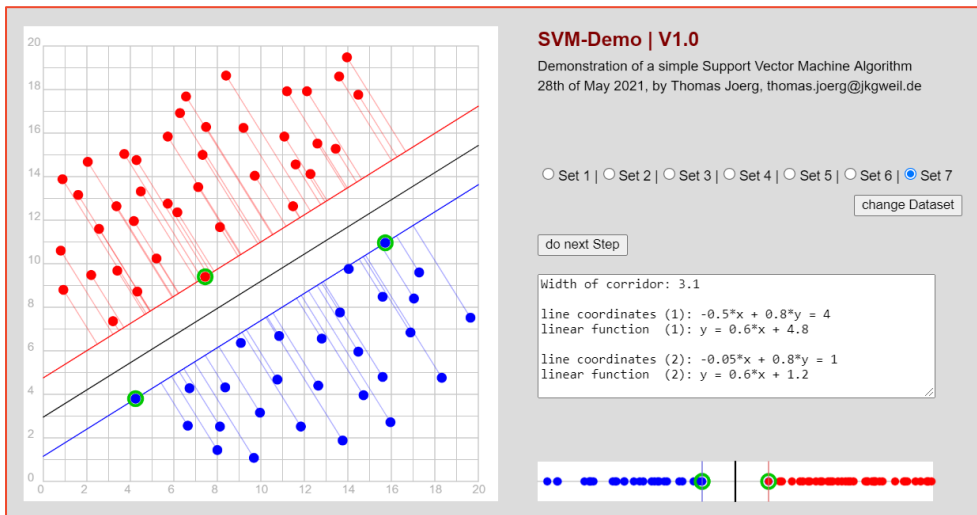
The great antagonist of neural networks: Support Vector Machines were the central and most powerful algorithms during the so-called "neural winter" in the 1970s and 1980s. They are still important today. They work according to the principle of the so-called "large margin classifier": The goal of the algorithm is the broadest possible separation of the individual classes. In this process, the separation plane is called the "hyperplane".

The didactic reduction in this teaching unit is to convey the basic working principle of the support vectors, but to merely address higher concepts like the "kernel trick" as an idea without deepening them. Likewise, we will restrict ourselves to linear separable classes. Polynomial separation functions can be illustrated by a simulation but will not be deepened.

Possible students' activities

What students need to work out: The principle of broadest possible separation: the use of so-called mathematical support vectors to achieve the best possible division of classes. To do this, they start by analyzing a simulation and formulate the algorithm linguistically. Afterwards, they will work through a 1D problem in which the students discuss different limitations and their respective advantages and disadvantages. A clear boundary is drawn to the so-called "kernel trick". Usually, data bases are modified mathematically in such a way that hyperplanes can be inserted to force a separation or classification. This is much too demanding and is deliberately left out here.

Part I: "Build the road as wide as possible"



Different data sets are looked at and discussed on how to make the boundary as thick as possible by using the widest possible channel.

To do this, one must first find exactly the two points from each of the different classes that are closest to each other. Because: the channel must run through these two points.

You can start at the line connecting these two points: The discriminator line is first set at right angles to this connecting line. But now you will probably see some points that lie inside the channel.

Within some further steps the channel will be optimized until no point is inside the channel anymore. The corresponding discriminator lines result mathematically (please only discuss, not calculate!) orthogonal to the support vectors of the data points

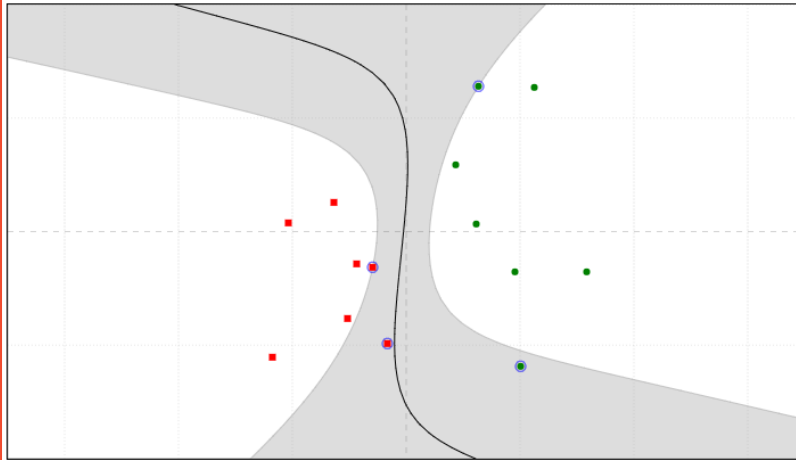
Repeat the procedure for different datasets in the app!

Interactive demo of Support Vector Machines (SVM)

February 12, 2018

tags: c++, machine-learning, svm, wasml

Note: you may have to **disable your adblocker** for this demo to work.



☒ Toggle $\nu = 0.15$
 Kernel: Sigmoid $\gamma = 1.0$ $c_0 = 0.0$
 $K(\mathbf{x}, \mathbf{y}) = \tanh(\gamma \mathbf{x} \cdot \mathbf{y} + c_0)$

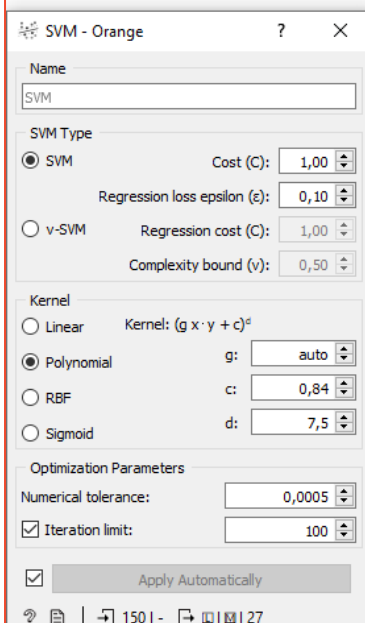
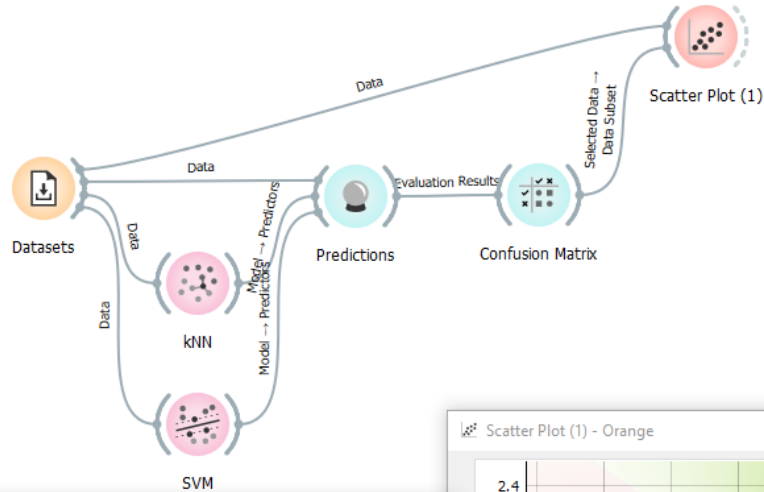
Part II: SVM can get very complicated

<https://jgreitemann.github.io/svm-demo>

SVMs can also be used to separate non-linearly separable data from each other for correct classification. However, this becomes mathematically very demanding very quickly and will not be discussed further here. The simulation example should only serve as an outlook.

Part III: kNN with Orange Data Mining

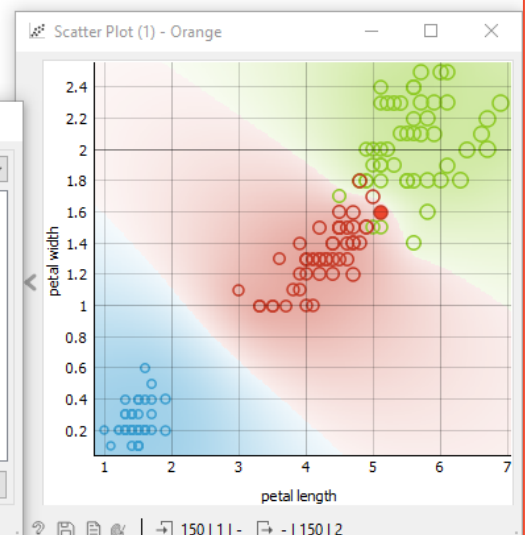
Extend the kNN project from the last chapter with an SVM node. Experiment with the hyperparameters to get the best model possible.



Confusion Matrix - Orange

Show: Number of instances

	Predicted			
	Iris-setosa	Iris-versicolor	Iris-virginica	Σ
Iris-setosa	50	0	0	50
Iris-versicolor	0	49	1	50
Iris-virginica	0	1	49	50
Σ	50	50	50	150



Part IV: SVM 1D unplugged

Imagine how to separate cultivated blueberries from strawberries. Cultivated Blueberries have an average weight of 8 grams, while the mean weight of Strawberries is about 15 grams. But these fruits size and weight can differ very much. So comes the question: How to separate them reliably?



Photo by [Bibhash Banerjee](#) on [Unsplash](#)

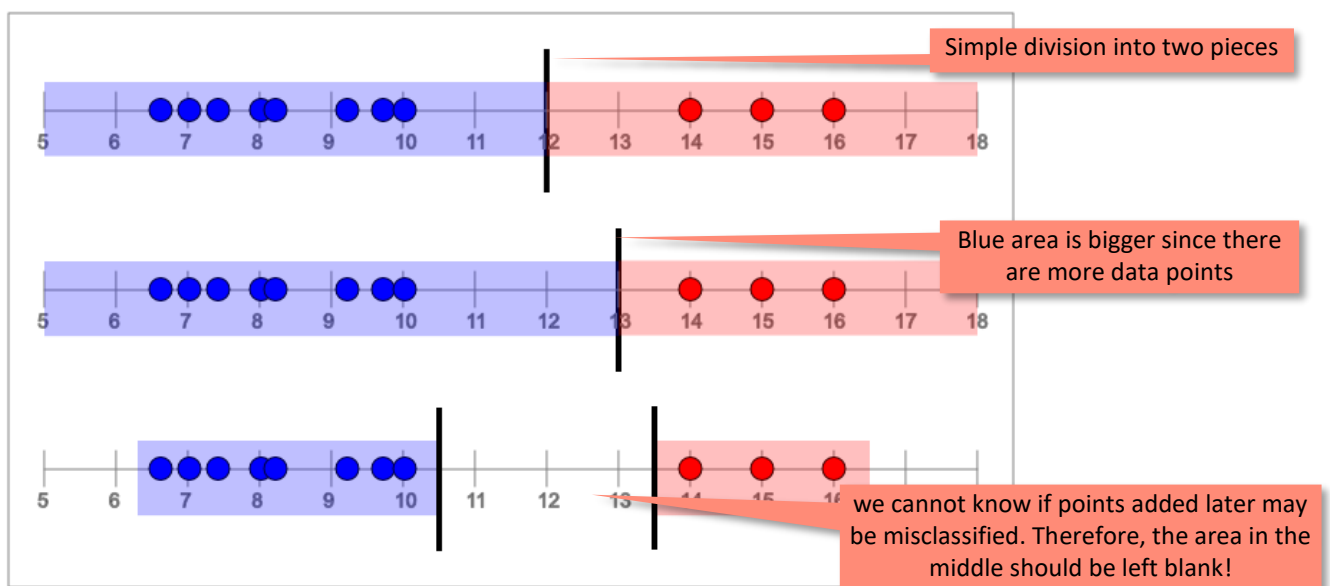


Photo by [Chris Bayer](#) on [Unsplash](#)

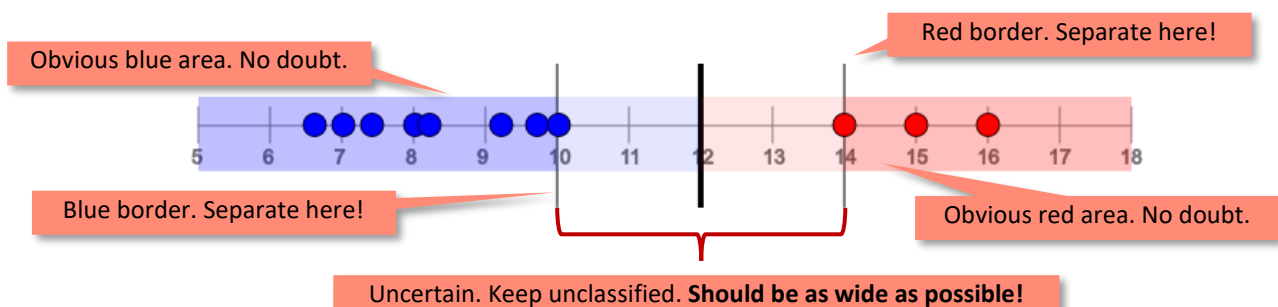
We build a small dataset with some weights of different fruits and visualize them on a number line:



Where is the cleverest way to set a dividing line to define an area for the blue and an area for the red points? Students should think about the problem and present their findings. They could be like these possible solutions:



It seems to be a reasonable approach to maximise the “area of unknown classification” and to consider the uncertainty between the border points of both areas:



Lesson 16: Orange Data Mining Project: Price prediction of a used car

What students should learn?

Students should independently work through the path from data preparation to data analysis to final prediction. It is important for the students to realize that despite all the highly developed algorithms, a lot of manual work is necessary in the adaptation of data and models. The students use the free software "Orange Data Mining" as a tool. "Orange" is Node-based and therefore encapsulates the complex mathematical background, allowing the students to fully focus on the data pipeline.

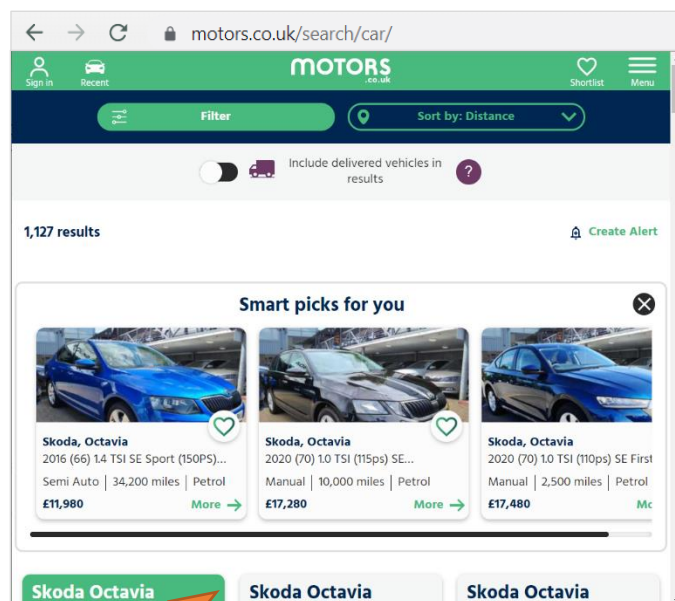
Note: This car is used since multivariate regression analysis happens to apply very well to this data set. The regression yields a correlation coefficient of about 0.9, which gives students a nice sense of accomplishment.

Possible students' activities

Run a complete DataMining project from start to finish: Based on a (hopefully) interesting real data set, the entire data preparation pipeline is set up and various ML models are trained with it. And this goes all the way to the fine-tuning of parameters. Students thus playfully learn how to deal with data-driven projects.

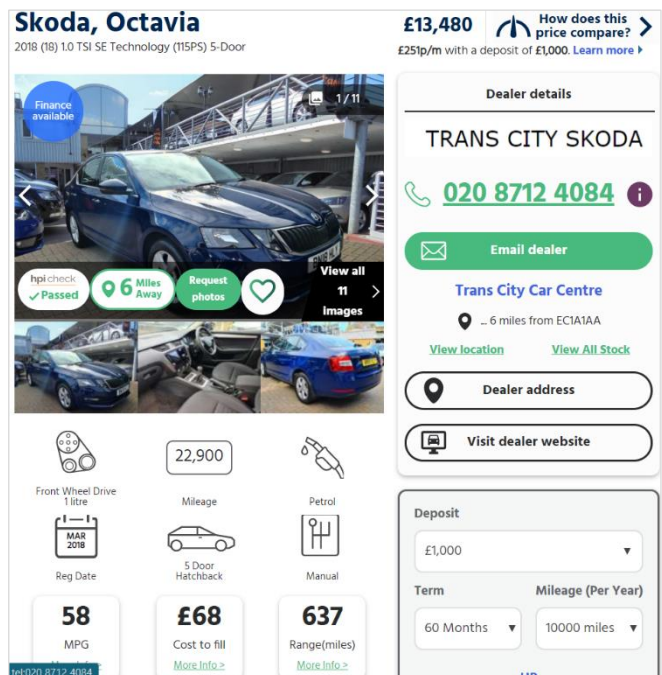
Part I: The story

Let's have a look on a very popular website offering used cars. We are interested in a "Skoda Octavia". A click on the button "How does this price compare" gives us the result: "No information available for this car".



Note: the following basic elements of the lesson can be implemented methodically in different ways. the key words mentioned here are only a description of the way for the teacher.

What can we do to avoid paying a too high price?



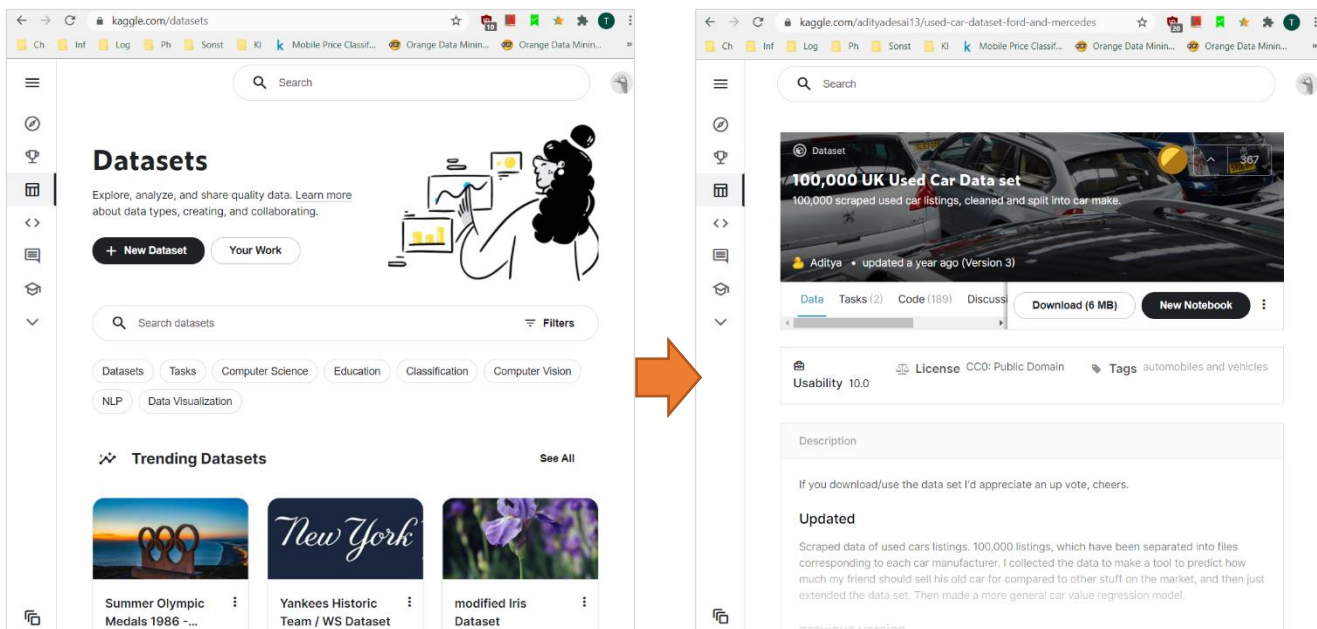
Part II: Feature selection.

Which features are relevant for building the price for this car?

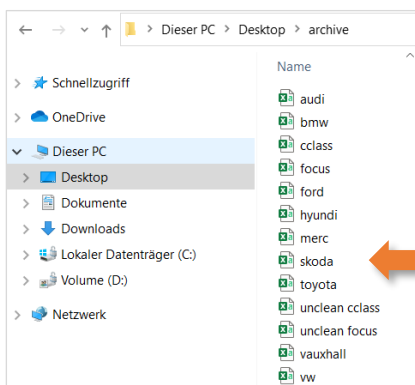
- **“Age:”** the newer the better and therefore the higher the price,
- **“Mileage”:** less mileage means higher price,
- **“Fuel type”:** Petrol cars are more expensive these days than diesel powered cars,
- **“Transmission type”:** Automatic or Manual, usually a manual controlled car is a bit cheaper,
- **“Miles per gallon”:** the higher the better and the more expensive,
- **“Engine size / horsepower”:** Bigger engine sizes usually means more horsepower means more expensive.
- **“Accessories”:** *specific color, air conditioners, specific rims, stereo systems ...*

Part III: Data is necessary

No prediction is possible without some clues - i.e., reliable data. you need as much data as possible from the past. Where do we get this kind of data? We will use Kaggle, a very popular website for data scientists and machine learning specialists:



Download the archive and unzip all files:

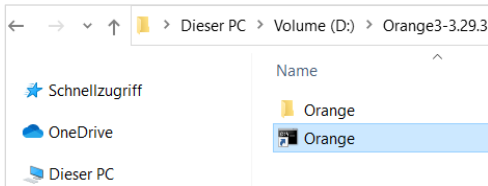


Now we can start working.

We will need this particular dataset:

Part IV: Introducing “Orange Data Mining”

The free software tool “Orange Data Mining” can be downloaded from <https://orangedatamining.com/>



If the portable version is used, it is not necessary to install it. Simply unpack and double-click the black “Orange”-icon.

Very useful information can be found on the YouTube-channel <https://www.youtube.com/c/OrangeDataMining/videos>

Workflow

Click this button

Double click opens dialog

search the file, load it

and set “price” as the target variable

Create a “Data Table”

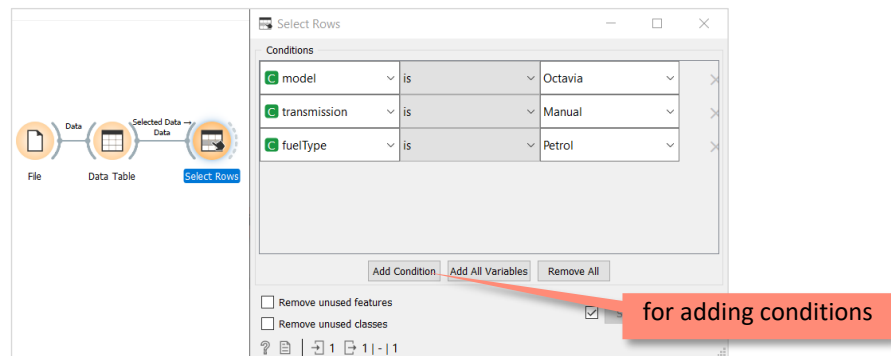
Connect it to “File”

and double click “Data Table”

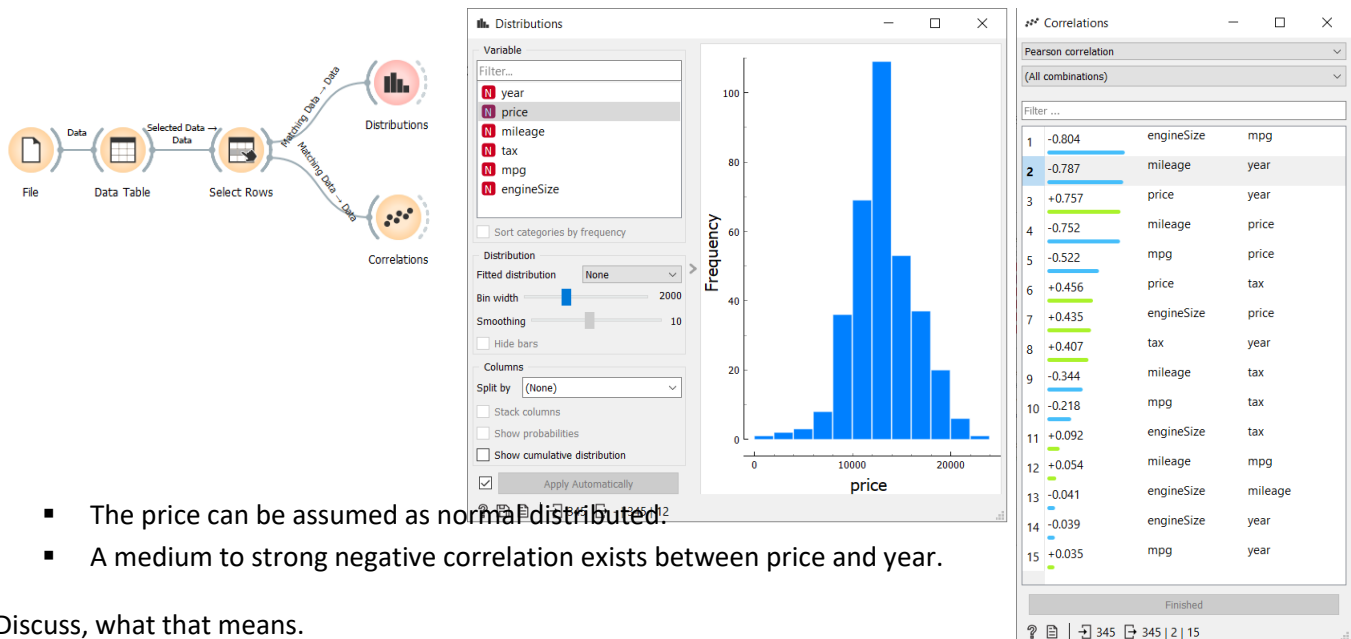
Be sure, NOT to select some specific rows since Orange performs all following steps only on this selected

There is too much data we don't need, so we must **filter** the relevant values.

Place a "Select Rows"-Node, connect it to the "Data Table", double click it and set the relevant filter conditions:



So, let's have a look what we have so far. For **getting information**, we can use "Distributions" and "Correlations":

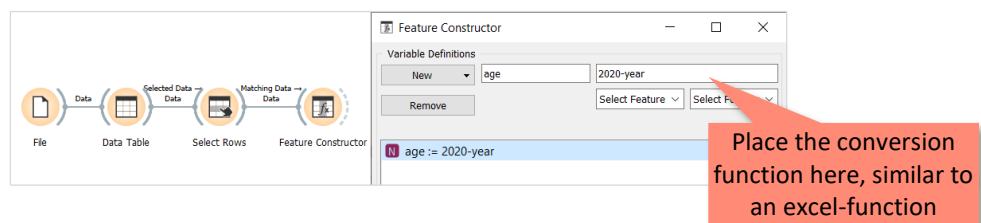


- The price can be assumed as normal distributed.
- A medium to strong negative correlation exists between price and year.

Discuss, what that means.

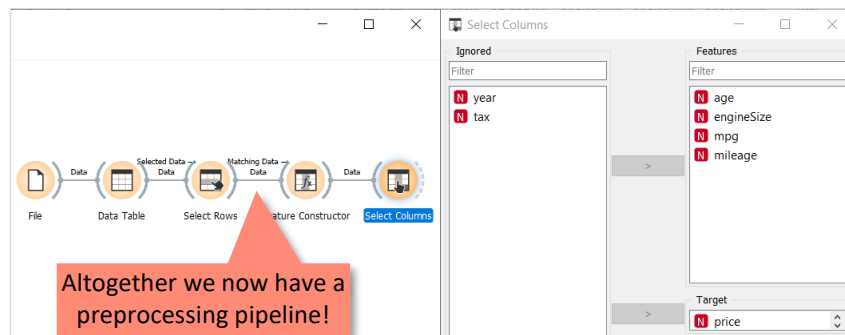
4. "Nice to have" but useful: create features.

The 'year' of a car does not make much sense – working with the "age" of a car would probably be more useful since we can generalize from the specific point of time the dataset was collected. Let's perform a simple conversion of "year" to "age". The dataset was collected in 2020 so we must subtract both values using the **"Feature Constructor"**:



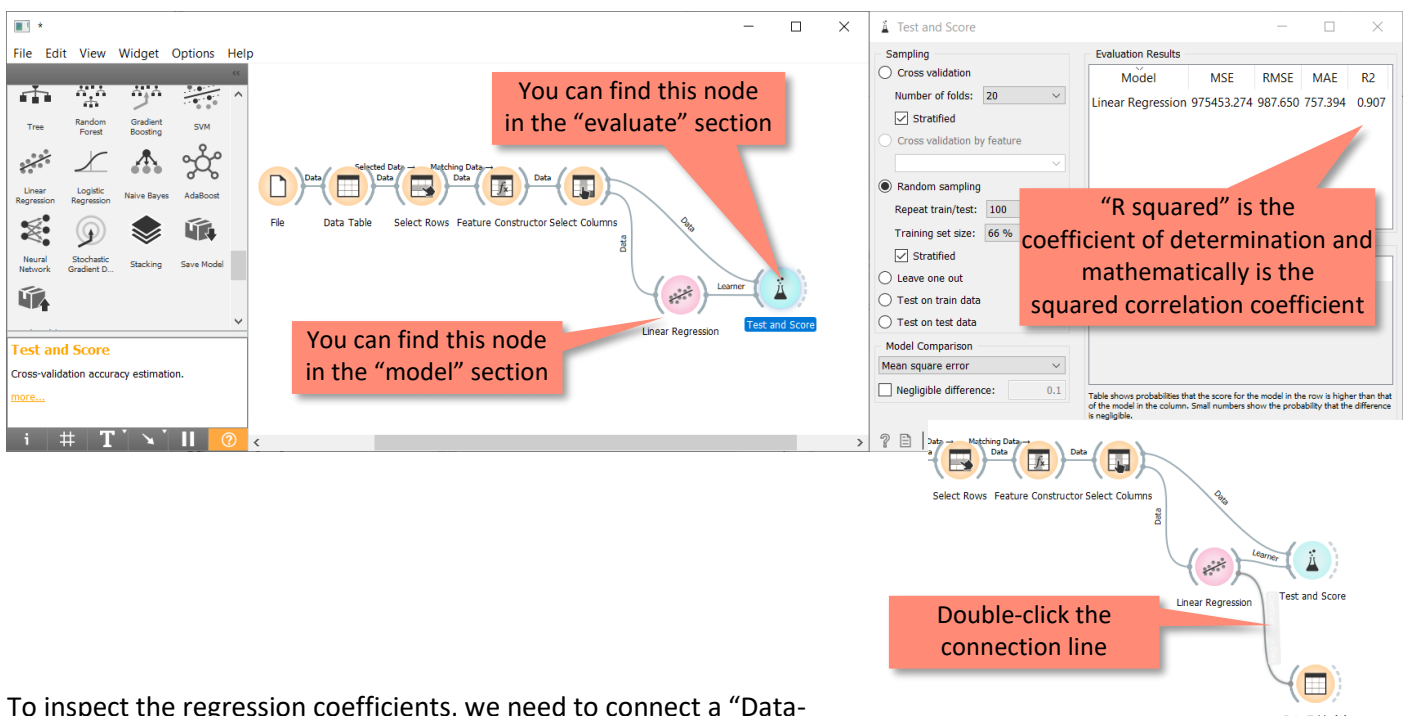
Part V: Now we can start with machine learning.

We must apply our thoughts from the beginning: which features are relevant, and which are not. Therefore, we must **select features** using the “Select Columns”-node:



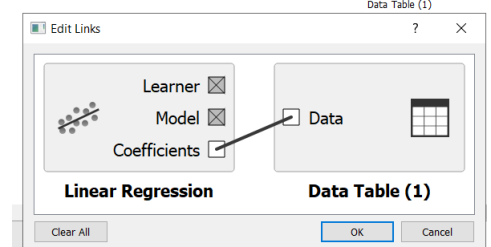
Now we can apply different algorithms to it to build a model. Let's start with a linear regression:

As a starting point, leave the “Linear Regression” at its default values. Make a double click on “Test and Score” to inspect the performance of the regression algorithm.



To inspect the regression coefficients, we need to connect a “Data-Table”-node to the Node of the linear regression. A double click on the connection line assures the correct data pipelining:

	name	coef
1	intercept	30750
2	age	-1016.29
3	engineSize	686.907
4	mpg	-274.135
5	mileage	-0.0553704



A first attempt of prediction “by hand”

This means we can now predict a medium price for our car using the following (slightly rounded!) formula with a $R^2 = 0.907$, which is pretty high:

predicted price

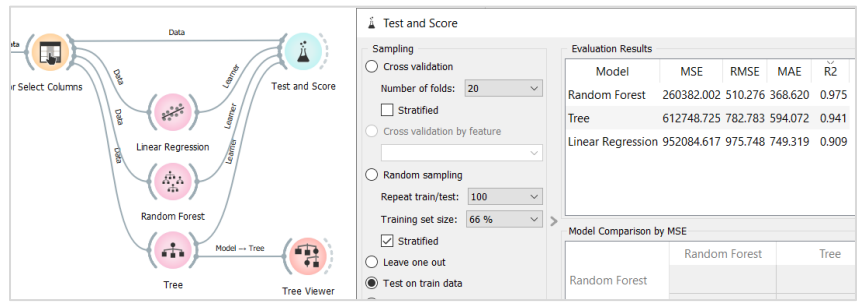
$$= 30750 - 1016 \cdot \text{age} + 687 \cdot \text{engineSize} - 274 \cdot \text{mpg} - 0.0554 \cdot \text{mileage}$$

Our example car had the following values:

Age: 2 (year 2018)	Engine Size: 1.0	Mileage: 22900	Mpg: 58	Price: 13480
--------------------	------------------	----------------	---------	--------------

$$\text{predicted price} = 30750 - 1016 \cdot 2 + 687 \cdot 1 - 274 \cdot 58 - 0.0554 \cdot 22900 = 12244$$

Which means: The car is too expensive at this price!

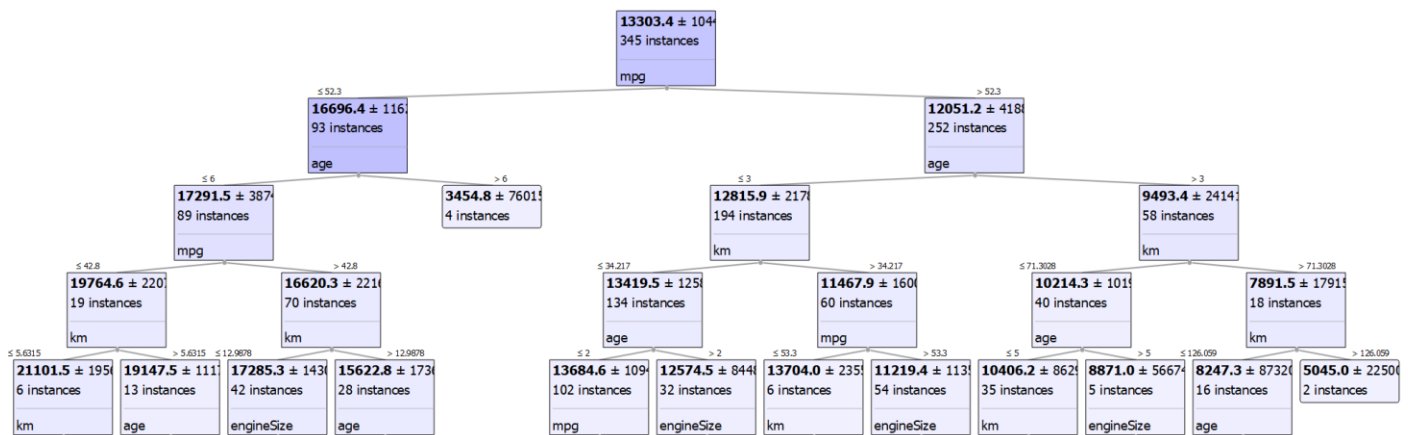


Part VI: Apply additional models

Let's try and compare some additional algorithms for regression: Decision tree and random forest

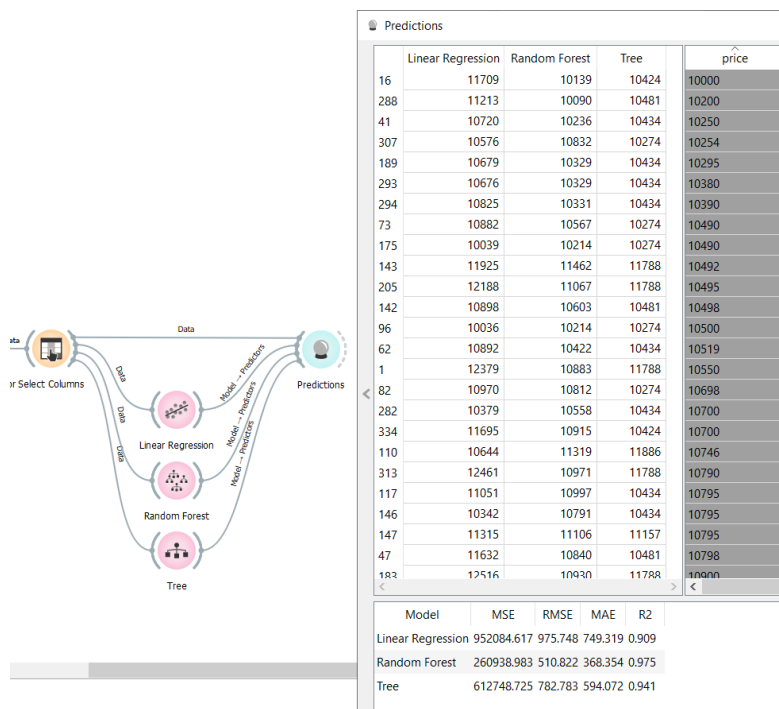
Both tree-based algorithms perform better than the linear regression.

Especially with the “Tree”-node we can use the “Tree Viewer”-node to inspect and visualize:



Automated Prediction using Orange

Replace the “Test and Score”-node with the “Prediction”-node and double-click it. Here you can see all models predicted values compared to the actual values. The “random forest”-model obviously performs as the best:



Part VI: Final thoughts and suggestions

There are some built-in datasets like the “iris”-dataset in orange. And on Kaggle there are many more for own projects – even some competitions, e.g., the prediction of surviving the titanic disaster. Students should play around with classification and regressions.

Lesson 17: Orange Data Mining Project, how to survive the titanic?

Or: Machine learning from disaster



What students should learn?

Students analyze an imperfect original data set. Non-perfect means that a dataset is incomplete, and some features are unusable. Thus, this scenario is close to realistic conditions: Before ML models are trained, students are forced to become familiar with the dataset from scratch. Basic assumptions about the relevance of features must be made before a meaningful selection of data can be made. The results are not always satisfactory. An average precision of the results in the range of 0.8 must be expected. But: quite plausible and surprising causal relationships can be identified again and again. For example, the probability of survival is strongly dependent on the gender or the class in which one was booked in.

Possible students' activities

Run a complete DataMining project from start to finish: Based on a (hopefully) interesting real data set, the entire data preparation pipeline is set up and various ML models are trained with it. And this goes all the way to the fine-tuning of parameters. Students thus playfully learn how to deal with data-driven projects.

It was a famous competition on Kaggle.com: How to predict the probability of surviving the titanic disaster. The original dataset can be found here: <https://www.kaggle.com/competitions/titanic>

Part I: Explanation of the dataset attributes

There are 12 Features, some are more some are less relevant:

Attribute	Explanation
SURVIVED	The target variable, two different classes: 0: not survived, 1: survived
PCLASS	First class (expensive), second and third class: 1, 2 and 3
NAME	The name of the passenger
GENDER	Male and Female
AGE	Age in years
SIBSP	Siblings and spouses: number of relatives plus husband or wife on board
PARCH	Number of parents or children on board
TICKET	ID of the ticket
FARE	Price of the ticket
CABIN	Cabin number
EMBARKED	Port of entry, C: Cherbourg, Q: Queenstown, S: Southampton
PASSENGERID	Sequential number for each row, "primary key"-variable

Part II: Didactic guidance for the teacher on what students should learn and do:

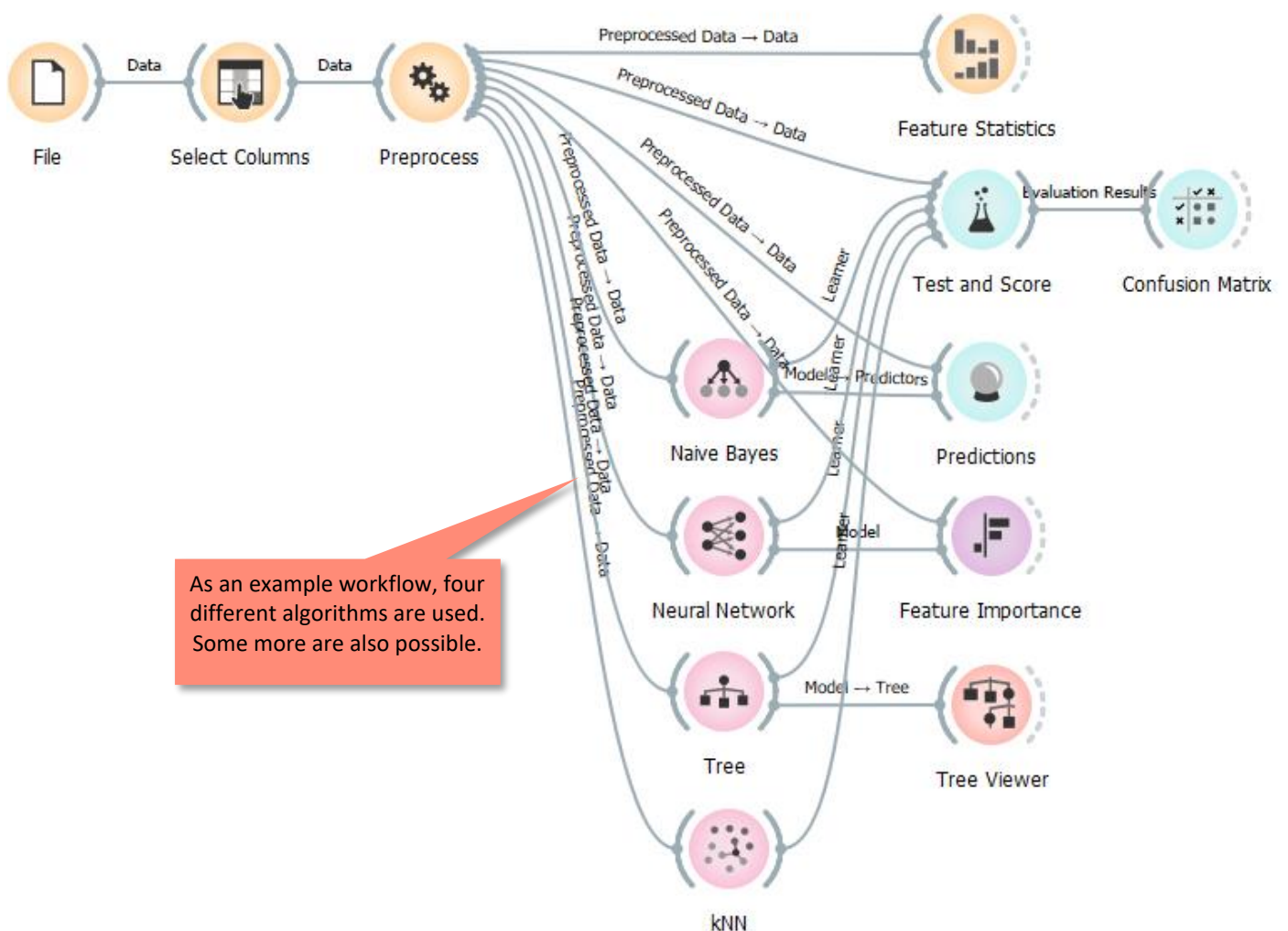
With this data set, the effort that needs to be put in is somewhat higher. Before a model can be created, preparation is necessary:

1. there are quite a few entries where values are missing. Usually, one has to remove these entries first. This is done with the "Preprocess" node.
2. there are some features which do not play a role for a prediction. These are the cabin number, the name, the "Passenger ID" and the ticket. These are removed from the record with the "Select Columns" node.

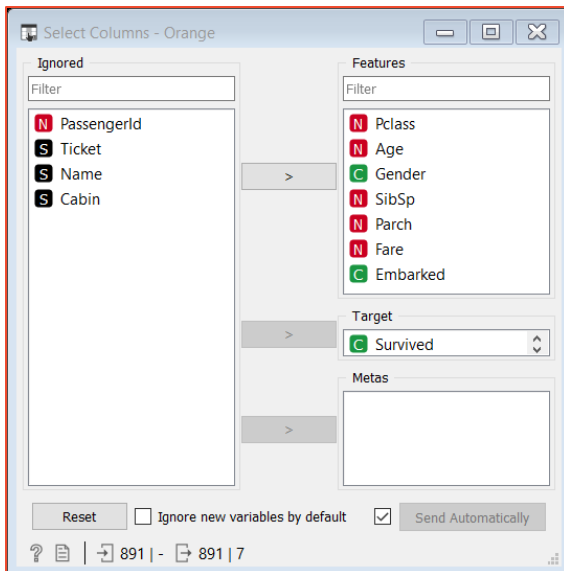
Once the dataset is clean, the data is first sifted under different aspects. Here, the "Feature Statistics" node stands out. Here you can formulate several conditional probabilities, which is a good way to practice the Bayes theorem.

The "Neural Network" node is not seriously needed for a prediction, because here one shoots with "cannons on sparrows" (German proverb). But the node is needed to calculate the "Feature Importance" node. Here, meaningful interpretations can be made as to which feature is particularly important and which features are rather unimportant.

Part III: An exemplary complete orange setup

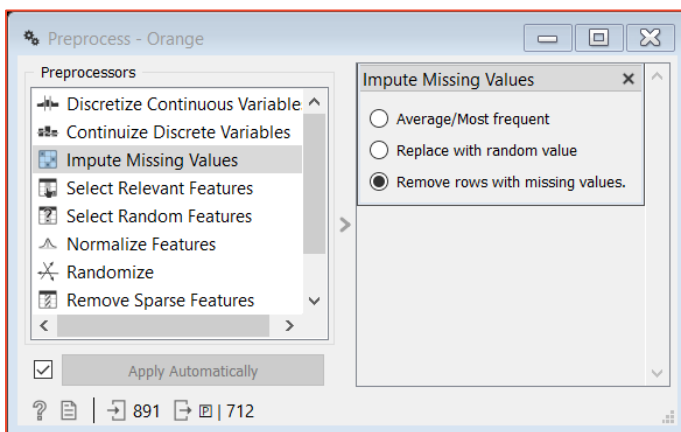


Part IV: Short explanation of the nodes



Select Columns

The irrelevant features are removed from the dataset, "Survived" is defined as target feature.



Preprocess

There are some values missing. This can be fixed in several ways. Here we just remove these rows with missing values.

	Survived	Pclass	Age	Gender	SibSp	Parch	Fare	Embarked
21	0	2	35.00	male	0	0	26.0000	S
22	1	2	34.00	male	0	0	13.0000	S
23	1	3	15.00	female	0	0	8.0292	Q
24	1	1	28.00	male	0	0	35.5000	S
25	0	3	8.00	female	3	1	21.0750	S
26	1	3	38.00	female	1	5	31.3875	S
27	0	3	?	male	0	0	7.2250	C
28	0	1	19.00	male	3	2	263.0000	S
29	1	3	?	female	0	0	7.8792	Q
30	0	3	?	male	0	0	7.8958	S
31	0	1	40.00	male	0	0	27.7208	C



Feature Statistics

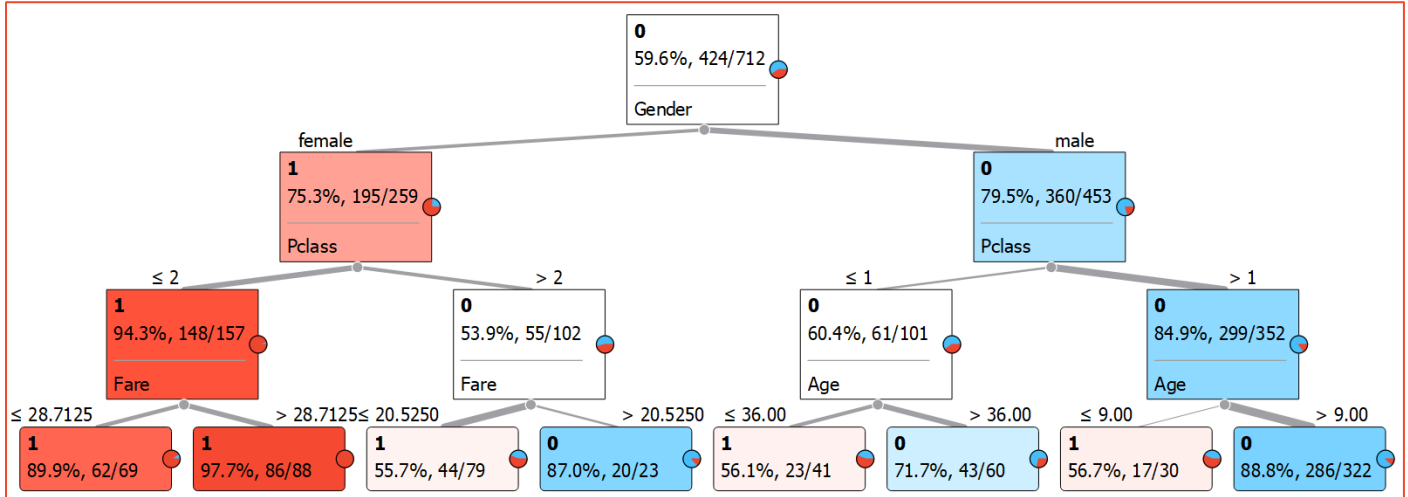
Here you can practice estimating in conditional probabilities, for example:

What was the probability of surviving under the condition of being female / being male?

What was the probability of surviving under the condition of having been enrolled in the first class?

Attention: Sensitive language is required here. On the one hand, because we are dealing with the death of human beings and, on the other hand, because we are dealing with gender-sensitive considerations.

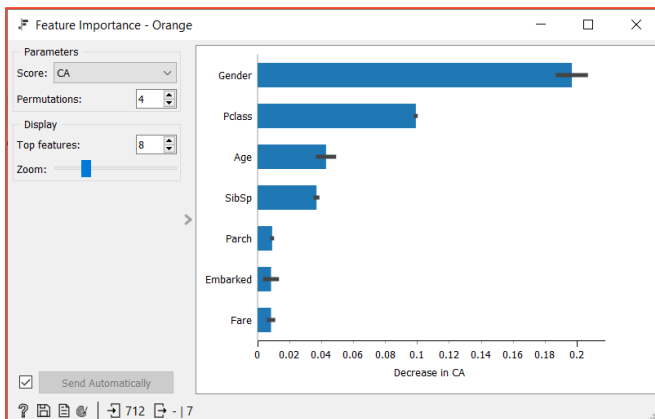
Decision tree & Tree Viewer



According to the Test and Score- Node the Decision Tree achieves an accuracy-score of 0.78 which is quite good. Additionally, the Decision Tree gives us a way to interpret the relationships:

- Accordingly, the ladies in the first class have the highest probability of survival.
- If you were a male passenger in third class, you had a halfway reasonable probability of survival if you were younger than 9. Children were therefore more likely to survive than adults.

The famous saying "women and children first" probably applied.



Feature importance

Together with the NN Node, which serves here as a kind of analytical tool, the previous statements can be confirmed and strengthened once again.

In order to survive, gender, passenger class and age were most important. As expected, the port of entry, for example, hardly played a role.,

Test and Score

Model	AUC	CA	F1	Precision	Recall
kNN	0.770	0.727	0.726	0.726	0.727
Tree	0.844	0.798	0.792	0.801	0.798
Neural Network	0.856	0.808	0.806	0.808	0.808
Naive Bayes	0.818	0.752	0.752	0.753	0.752

With a Train-Test-Split of 66% and 50 Epochs, Tree and Neural Network seem to be the best models for predicting the survival probability.

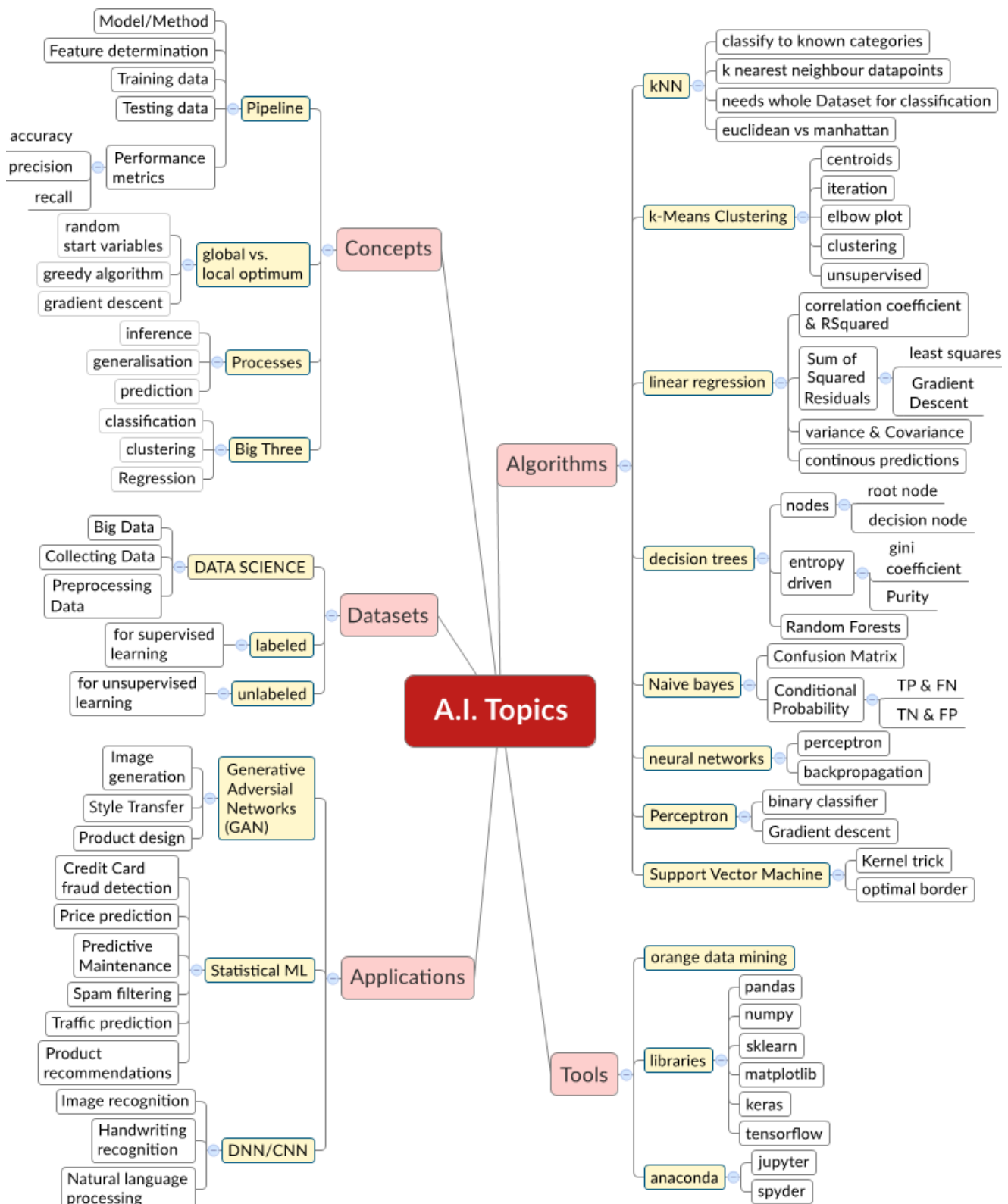
Lesson 18: Summary of the complete lesson

What students should learn?

Students should use an overall review to visualize the lesson content they have learned. Here there is the possibility to cross-link the learning contents. A general overview provides a better understanding of the interrelationships.

Possible students' activities

Students summarize all the content of the lesson by getting together in groups and discussing it. The work result could be, for example, a mind map or a short presentation. This can be presented by a short talk or a poster.



Didactical considerations at the end

What is the essence of the many different ML algorithms? What should students have implicitly understood about them when the lesson is over?

Many different algorithms have been studied: Linear Regression, Naive Bayes, Support Vector Machine, Decision Tree, kNN, and k-Means. Later, neurons and neural networks will be added. Is there a common principle behind all these mosaic stones?

Yes, it is the ability of all algorithms to adapt to unpredictable data. This is done by these algorithms trying to adapt to the data as well as possible in a similar way. Dynamic adaptability thus becomes the heart, the substance of all the different methods.

Strictly speaking, these are optimization algorithms. In the case of the decision tree, the driving force is entropy reduction, which is carried by a greedy algorithm. A k-means algorithm optimizes the location of the centroids in distinct iterative steps, and a support vector machine optimizes the width of the corridor between points.

The linear regression algorithm differs from the one used in statistical science by its implementation: by iteratively changing the slope, the influence of the residual squares is minimized. It was not explicitly taught because it is higher mathematics, but here the principle of gradient descent works.

The central element of gradient descent is found again in neural networks: A neuron also learns by optimizing its weights. These are mediated by the backpropagation algorithm. And it works, similar to linear regression, according to the principle of gradient descent.

This is the common thread that students should take away: Putting the picture together from the individual pieces of the mosaic, we end up with the realization that the essence of machine learning lies in optimizing to unknown data.

