MACHINE LEARNING ZOOMCAMP: 2024 COHORT

Table of Contents

[**1.** **Brief Introduction to DataTalks** 3](#_Toc178609998)

[**2.** **Introduction to Machine Learning** 3](#_Toc178609999)

[2.1 Introduction to Machine Learning 3](#_Toc178610000)

[2.2 ML vs Rule-Based Systems 7](#_Toc178610001)

[Overview: 7](#_Toc178610002)

[Rule-based systems 7](#_Toc178610003)

[Machine Learning 7](#_Toc178610004)

[Collect the data 7](#_Toc178610005)

[Define and calculate (extract) the features 7](#_Toc178610006)

[Training 8](#_Toc178610007)

[Apply the model 8](#_Toc178610008)

[2.3 Supervised Machine Learning 8](#_Toc178610009)

[Overview: 8](#_Toc178610010)

[What is Supervised Machine Learning? 9](#_Toc178610011)

[Types of Supervised Machine Learning 11](#_Toc178610012)

[2.4 CRISP-DM 11](#_Toc178610013)

[Overview: 11](#_Toc178610014)

[CRISP-DM Machine Learning Process 12](#_Toc178610015)

[CRISP-DM is an Iterative process with 6 steps 12](#_Toc178610016)

[Business Understanding 12](#_Toc178610017)

[Data Understanding 13](#_Toc178610018)

[Data Preparation (= Feature Engineering) 13](#_Toc178610019)

[Modelling 14](#_Toc178610020)

[Evaluation 14](#_Toc178610021)

[Evaluation + Deployment (Often happens together) 14](#_Toc178610022)

[Deployment (=engineering practices) 14](#_Toc178610023)

[Iterate! 15](#_Toc178610024)

[General note 15](#_Toc178610025)

[2.5 Model Selection Process 15](#_Toc178610026)

[2.6 Setting up the Environment 15](#_Toc178610027)

[2.7 Introduction to NumPy 15](#_Toc178610028)

[2.8 Linear Algebra Refresher 15](#_Toc178610029)

[2.9 Introduction to Pandas 15](#_Toc178610030)

[2.10 Summary 15](#_Toc178610031)

[2.11 Homework 15](#_Toc178610032)

[**3.** **Machine Learning for Regression** 15](#_Toc178610033)

[**4.** **Machine Learning for Classification** 15](#_Toc178610034)

[**5.** **Evaluation Metrics for Classification** 15](#_Toc178610035)

[**6.** **Deploying Machine Learning Models** 15](#_Toc178610036)

[**7.** **Decision Trees and Ensemble Learning** 15](#_Toc178610037)

[**8.** **Neural Networks and Deep Learning** 15](#_Toc178610038)

[**9.** **Serverless Deep Learning** 15](#_Toc178610039)

[**10.** **Kubernetes and TensorFlow Serving** 15](#_Toc178610040)

# **Brief Introduction to** [**DataTalks**](https://datatalks.club/)

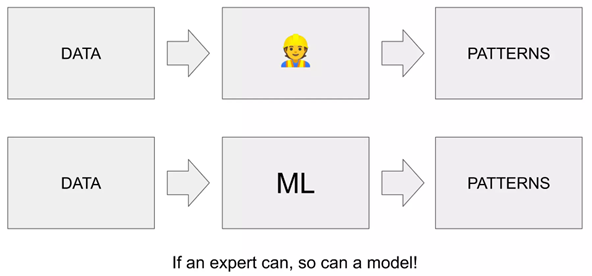
# **Introduction to Machine Learning**

## Introduction to Machine Learning

**Overview:**

1. Train a model
2. Use a model

This is a summary of what I’ve learned from the great ML course (<https://github.com/DataTalksClub/machine-learning-zoomcamp>) by Alexey Grigorev. All images from this post are from the course material. Images in other posts can also be copies of that material.

The introduction starts with an explanation about what ML really is. You can imagine a task that is normally done by an expert, like getting a good price for selling a car. The expert takes the data about the car and combines all the characteristics to get his opinion about the fair price. What he does is, he extracts patterns from the data.If a human is able to do this, so a model can do the same.

Following is an explanation of what ML is.

**Machine Learning** (ML) is about using ***features***and the **target**information to train a **model**and use this model to predict unknown object targets. In other words it is a process of extracting patterns from data (features + target).

To understand this, you have to distinguish between the terms feature, target and model.

**Features**means what we know about an object. In this example what we know about the characteristics of a car. A feature is a characteristic of an object in form of a number, string, or other more complex form (e.g. location information, …)

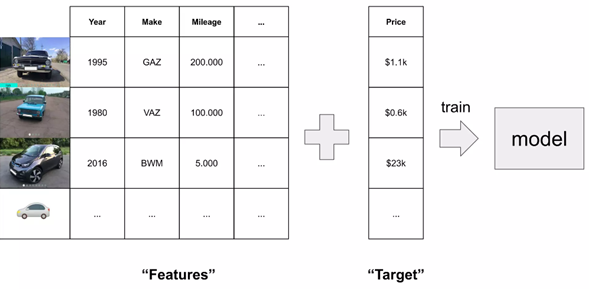
**Target**is what we want to predict. Other courses / sources also use the term label for this purpose.That means, in training, you talk about a labeled data set because you know the target. In this example, many labeled data sets of cars with prices are used to predict a label for an unknown data set of another car.

A **model**is the output artefact of a **training**that contains all the patterns learned from the trained examples. This output can be used later to make a **prediction**(try to output the target variable) based on features of an unknown object and the model itself.

**Train a Model**

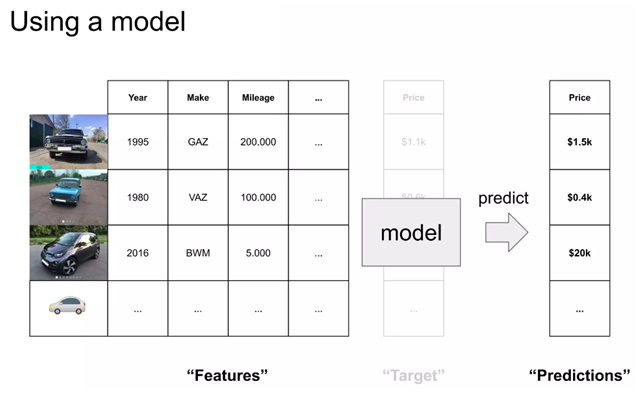
Model training is the process where the machine extracts the patterns from the given training data. In easy words the features are combined with the target – this leads to the model.

**Machine Learning**



**Use a Model**

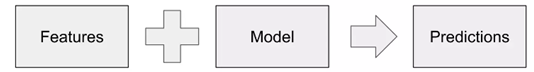
Mere training does not make a model useful. Only the application brings the benefit. Applying the trained model to an unknown data set (without target), you obtain a prediction for the missing information (here: the price).



To summarize the difference between model training



And prediction



is that in training process you use features and the target to get the model. And in the prediction process you only use the features and apply the trained model to get a prediction for the target variable.

## ML vs Rule-Based Systems

### Overview:

1. Rule-based systems
2. Machine Learning
3. Collect the data
4. Define & calculate (extract) the features
5. Training
6. Apply the model

The second part is about the distinction between **Machine Learning** (ML) and **rule-based systems**. The example of a spam filter is used to explain how the implementation would look like without ML.

### Rule-based systems

What you need to do is to define some rules to distinguish between ham and spam. So you start defining the rules and for a while everything works fine. However, at some point you have to adjust the rule set and you end up on the hamster wheel because you can’t handle the constant reconfiguration of the rules. Also, this system gets harder and harder to maintain.

### Machine Learning

The second way to implement this Spam filter is to use ML instead of using hard-coded rules. That means you need to collect the data, define & calculate (extract) the features, and then train and use the model to classify messages into spam and not spam.

### Collect the data

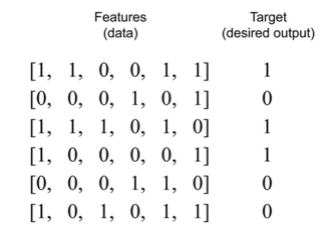
Collecting the data while using the “SPAM” button of your mail system.

### Define and calculate (extract) the features

Features:

* Length of title > 10? True/False
* Length of body > 10? True/False
* Sender [promotions@online.com](mailto:promotions@online.com)? True/False
* Sender [hpYOSKmL@test.com](mailto:hpYOSKmL@test.com)? True/False
* Sender domain “test.com”? True/False
* Description contain “deposit”? True/False

All of the six features here are binary features, so you can encode each mail as binary code like [1,1,0,0,1,1]. Besides this every email has a label1/target (spam = 1, no-spam = 0), which is the desired output.



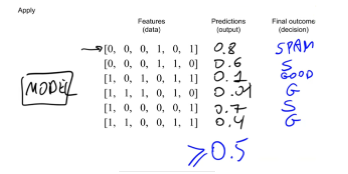
### Training

This data is used to train the model. This process if often called a fitting a model.

In training, something happens that is similar to solving a very complex system of equations with many parameters. Here, the features are offset against each other in such a way that the correct classification is obtained at the end. Correct in this example means 1 for spam 0 for no spam. More precisely, we get a probability for the correct label. The trained model contains exactly the information that best solves the equation, namely the weights with which the individual features must be offset to get the correct result.

### Apply the model

If the model is now applied to unknown data sets, 5 the result is a probability. This probability indicates whether this is a spam mail or not. To finally decide how to categorize the mail, a threshold is used (e.g., 0.5). Thus, everything greater than or equal to 0.5 is declared as spam.



## Supervised Machine Learning

### Overview:

1. What is Supervised Machine Learning
2. Types of Supervised Machine Learning

As I mentioned before (in ML vs rule-based systems) there are several approaches to get a software solution for a problem. To give an overview there is the classical approach where everything is hard-coded. In contrast to this there are AI-approaches.

On the one hand there are knowledge-based systems, that are divided into rule-based systems and case-based Reasoning. Rule-based systems were mentioned before this section.

On the other hand, there is machine learning as another AI approach. “*Machine Learning […] provides systems with the ability to learn from experience without being programmed explicitly. Machine Learning is concerned with the development of […] applications that can access data and learn from it on themselves*.”

There are different kind of problems ML is trying to solve.

* **Regression** (predict continuous values, e.g., prices)
* **Classification** (predict labels to distinguish between different classes)
* **Clustering** (predict groups or the data without having any group labels or group characteristics)

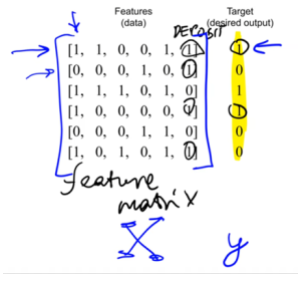
Depending on these problem types you can use several different **learning strategies** to solve the problem.

* **Supervised Learning**
* **Unsupervised Learning**
* **Semi-supervised Learning**
* **Reinforcement Learning**
* **Active Learning**

This should give you a brief overview about where Supervised Learning fits in. For more information on all other approaches, please refer to the literature.

### What is Supervised Machine Learning?

Supervising the model means that the act as teacher for the model while showing different examples with its target value (e.g., price of the car). The machines is able to learn from this examples, while it’s extracting the patterns and generalize to new examples.



From the figure above, you can extract a lot of important information:

* Rows are the observations or objects for which we want to predict something
* Columns are features of the dedicated observation/object
* X is defined as the whole set of features that is called feature matrix (two-dimensional array, array of arrays)
* y is defined as vector with the target variable (one-dimensional array)

From this you can derive the formal definition of supervised machine learning: **g(X) ~ y** where:

* X: feature matrix
* y: target variable
* g: model that takes X and produces something that is approximately close to y

The aim of a training is to get the function g. Mostly this model (g) won’t be able to predict always the correct target variable, but we try to be as close as possible.

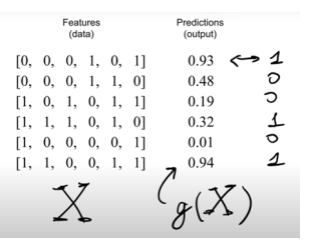


Figure above shows sample predictions for a few objects. You see the output of the function g(X) as a likelihood. Depending on the threshold this value is evaluated to 0 or 1.

### Types of Supervised Machine Learning

#### Regression:

* e.g., price of a car/house/…
* g predicts a number between -inf…+inf

#### Classification:

* e.g., identify a picture as a car, identify mail as spam
* g predicts a category
* Input is a picture (or a characteristics of an object/observation) and the output is the label/class

##### Subtypes of Classification:

* Multiclass classification problem (distinguish between several classes (e.g., cat, dog, car))
* Binary classification problem (distinguishes between two classes (e.g., spam vs. not spam))

#### Ranking:

* Usually used when you want to rank something (e.g., recommender system) -> giving a score to each item in an e-commerce shop and show the top values, because the algorithm calculates that these items have the highest potential to be bought by this customer.
* Google search engine works in a similar way.

## CRISP-DM

### Overview:

1. CRISP-DM Machine Learning Process
2. CRISP-DM is an iterative process with 6 steps
3. Business Understanding
4. Data Understanding
5. Data Preparation (=Feature Engineering)
6. Modelling
7. Evaluation
8. Evaluation + Deployment (Often happens together)
9. Deployment (=engineering practices)
10. Iterate!
11. General note

### CRISP-DM Machine Learning Process

This is about the CRISP-DM Machine Learning Process (Cross-industry standard process for data mining). Methodologies like CRISP-DM help us to organize the ML project in a way that is manageable (what needs to happen in which order).

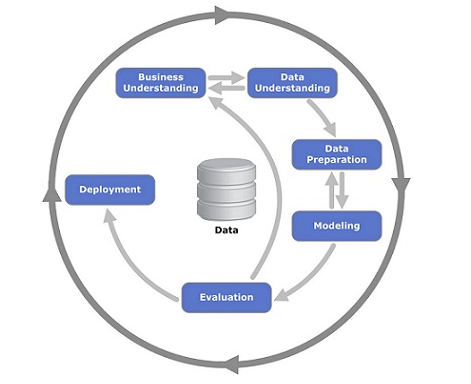


Figure above is from [Wikipedia](https://en.wikipedia.org/wiki/Cross-industry_standard_process_for_data_mining). You can find more information on that topic especially in the reference section there is a link to “[CRISP-DM 1.0 Step-by-step data mining guide](https://www.semanticscholar.org/paper/CRISP-DM-1.0%3A-Step-by-step-data-mining-guide-Chapman/54bad20bbc7938991bf34f86dde0babfbd2d5a72)” if you need more details on that.

### CRISP-DM is an Iterative process with 6 steps

1. Business Understanding (try to understand the problem)
2. Data Understanding
3. Data Preparation (often called as Feature Engineering)
4. Modelling (train the model)
5. Evaluation
6. Deployment (using the model)

In the following more detailed descriptions of the steps there are some *italic* lines that are not from the course videos but from a book.

### Business Understanding

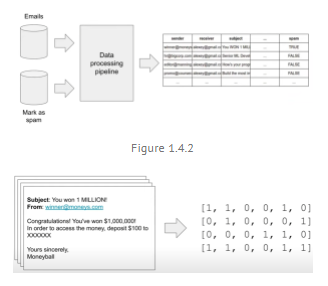
* Identify the business problem
* *Detect available data sources*
* *Specify requirements, premises, and conditions*
* *Clarify risks and uncertainties*
* Understand whether the problem is important
* Understand how we can solve it
* **Understand how we measure the success of our project (Cost-Benefit-Analysis)**
* Do we actually need ML here?

### Data Understanding

* Analyse available data sources
* *Collect and analyse data*
* Analyse if something is missing and what is missing
* Decide if this data is good/reliable/large enough
* Decide if we need to get more data

### Data Preparation (= Feature Engineering)

* Transform the data so it can be put into a ML algorithm
* Usually this means extracting different features
* Clean the data / remove all the noise
* Build the pipelines (that transform raw data into clean data)
* Convert data into tabular form (needed to put in machine learning model)



Feature Engineering is a key element of every ML project. There is a quote of Andrew Ng, Professor of the Standford University, about Feature Engineering: “**Coming up with features is difficult, time-consuming, requires expert knowledge. ‘Applied Machine Learning’ is basically feature engineering.**” I found this quote in a very good german book. This contains a chapter about the CRISP-DM model and Feature Engineering.

In addition, I found on [towardsdatascience.com](https://towardsdatascience.com/machine-learning-isnt-models-it-s-features-be87b386db39) an old but still interesting article on this subject. There, the importance of feature engineering is also highlighted.

### Modelling

* Train the model (the actual ML happens here)
* Try different models (e.g., Logistic regression, Decision tree, Neural network, others)
* *Select model parameters*
* *Try to improve model quality*
* Select the best one
* Sometimes, we may go back to data preparation
* Add new features
* Fix data issues
* General aspect that I’ve learned from practice: **model quality significantly depends on data quality -> keep in mind: Garbage in, Garbage out!**

### Evaluation

* Measure how well the model solves the business problem
* Is the model good enough?
* Have we reached the goal?
* Do our metrics improve?
* Goal: Reduce the amount of spam by 50%
* Have we reduced it? By how much?
* (Evaluate on the test group)
* Do a retrospective:
* Was the goal achievable?
* Did we solve/measure the right thing?
* After that, we may decide to:
* Go back and adjust the goal
* Roll out the model to more users/all users
* Stop working on the project

### Evaluation + Deployment (Often happens together)

* Online evaluation: evaluation of live users
* It means: deploy the model, evaluate it

### Deployment (=engineering practices)

* After online evaluation of some users -> deploy the model to production (all remaining users)
* Roll out the model to all users
* Proper monitoring
* Ensuring the quality and maintainability
* -> when we deploy model it has to work, it has to be reliable
* After that we care about scalability and other things
* Like in project management this includes creating the final report

### Iterate!

* ML projects require many iterations!
* After deployment we come back to business understanding to check how can we improve the model or decide that it needs to be improved or not.

### General note

* Start simple (e.g., with a simple model)
* Learn from feedback
* Improve (e.g., come back to business understanding and make this model a bit more complex)

## 2.5 Model Selection Process

## 2.6 Setting up the Environment

## 2.7 Introduction to NumPy

## 2.8 Linear Algebra Refresher

## 2.9 Introduction to Pandas

## 2.10 Summary

## 2.11 Homework

# **Machine Learning for Regression**

# **Machine Learning for Classification**

# **Evaluation Metrics for Classification**

# **Deploying Machine Learning Models**

# **Decision Trees and Ensemble Learning**

# **Neural Networks and Deep Learning**

# **Serverless Deep Learning**

# **Kubernetes and TensorFlow Serving**