

Antoine PALISSON

Machine Learning

Basics



TABLE OF CONTENTS

01

Introduction

**Problem Framing &
Data Collection**

02

03

Data Exploration

Data Preprocessing

04

05

Training a model

**Performances &
Metrics**

06

TABLE OF CONTENTS

07

**Overfitting &
Underfitting**

**Fine-Tuning
a Model**

08

09

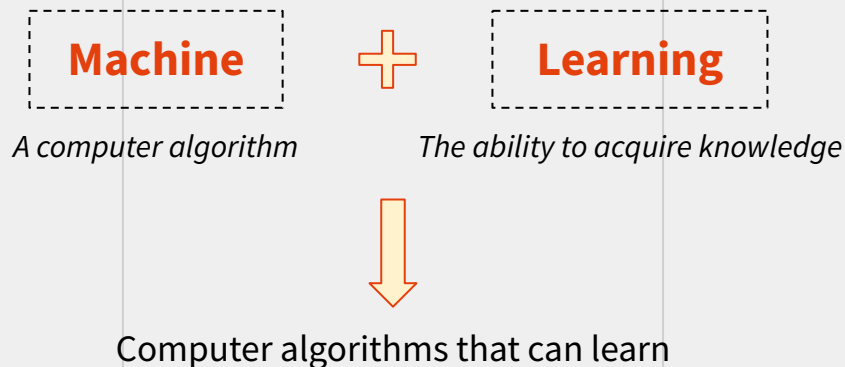
**Introduction to
Unsupervised
methods**



Introduction

Definition

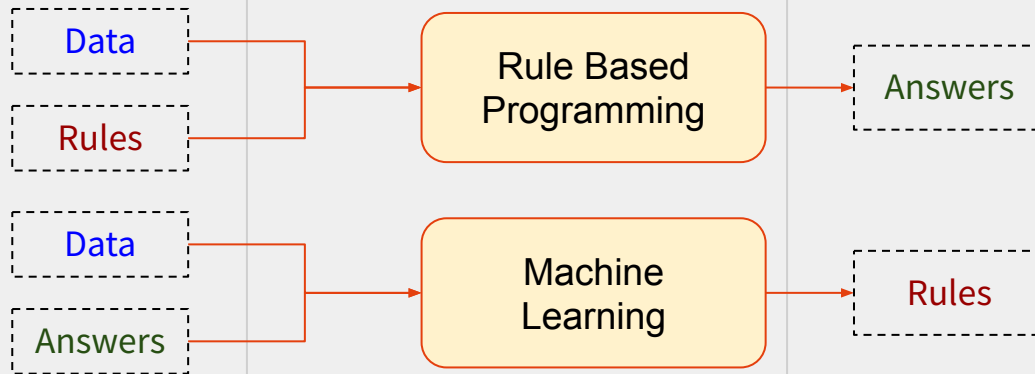
Machine learning is a field of artificial intelligence (AI) that focuses on creating computer algorithms that can **learn from data** and **make predictions** or decisions based on that learning.



Definition (2)

Machine learning algorithms are the opposite of the rule based programming algorithms and are designed to **recognize patterns in data**.

The core idea is to build computer programs that can improve their performance on a task by analyzing data, rather than being explicitly programmed to do so.



Why using it ?

Machine learning offers a range of potential benefits for businesses and organizations, making it an increasingly popular and valuable tool in a variety of industries:

- **Improved Performances** - *it can often achieve higher accuracy rates than traditional methods*
- **Time savings** - *it can automate many tasks that would otherwise require human intervention*
- **Personalization** - *it can be used to personalize products, services, or content to individual users*
- **Cost savings** - *it can save businesses money by automating tasks and improving efficiency*

Competitive Advantage

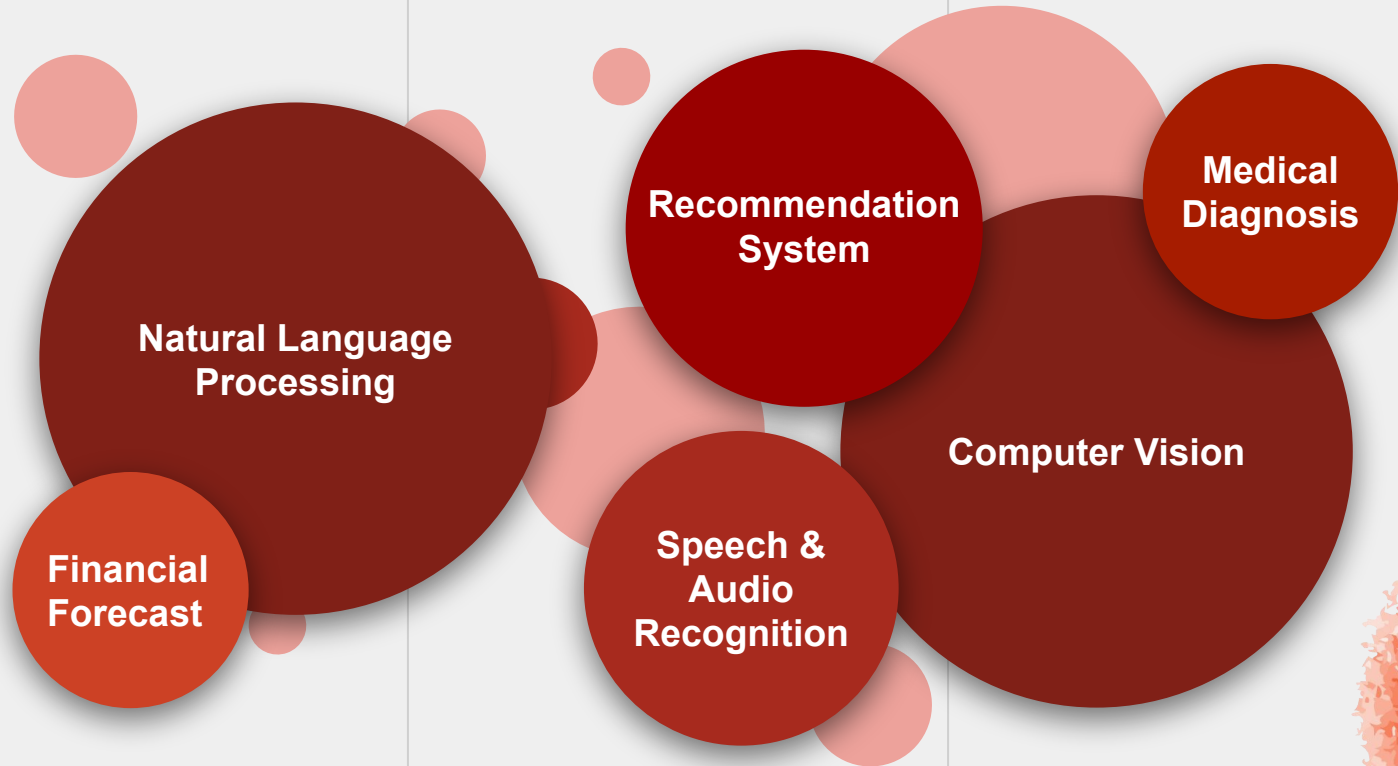
One of the key benefits of machine learning is that it can **give businesses a competitive advantage**. By using machine learning to analyze data and automate tasks, businesses can **gain insights and make decisions more quickly and effectively than their competitors**.

Amazon uses machine learning algorithms to personalize its product recommendations for individual users based on their past browsing and purchasing behavior.

This personalization has given Amazon a competitive edge in the online retail market, as it allows the company to offer a highly tailored and personalized shopping experience to each individual customer.

30% of its sales are estimated to be generated by its recommendation system.

Applications



Data

Every machine learning algorithm need data. **Data generally refers to the input variables, known as features, used to train a algorithm.**

Features typically comes with their corresponding output variable that the algorithm is trying to predict, known as **label** or **target**.

Features

The input variables that are used to describe each instance in the dataset.

Labels/Targets

The output variable that the machine learning algorithm is trying to predict

For example, in a dataset of customer records, the features might include age, income, and location while the label might be the type of customer (high-value, low-value, frequent buyer ...).

Machine Learning Types

Machine Learning can be divided into 4 categories.

01

**Supervised
Learning**

- Regression
- Classification

02

**Unsupervised
Learning**

- Clustering
- Dimensionality Reduction
- Anomaly Detection

03

**Semi-Supervised
Learning**

Supervised & Unsupervised

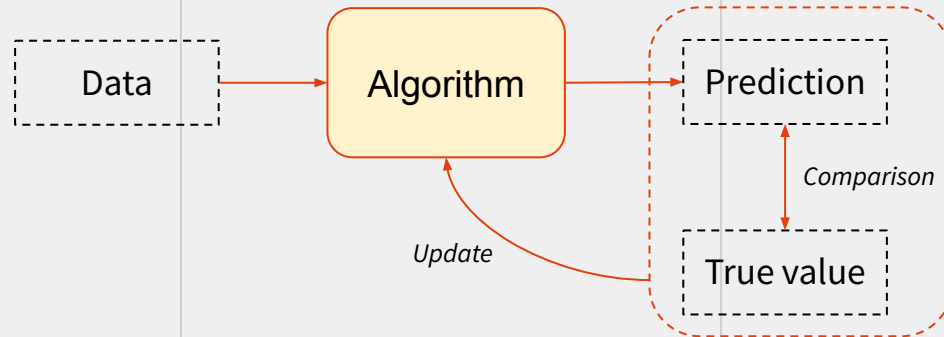
04

**Reinforcement
Learning**

- State-action-reward-state-action
- Q-learning

Supervised Learning

Supervised learning is a type of machine learning where an algorithm learns to map inputs to outputs based on **feature/label** pairs provided in a dataset. In other words, a supervised machine learning algorithm is “supervised by a teacher” who provides it with examples of **what the correct output should be for a given input**.

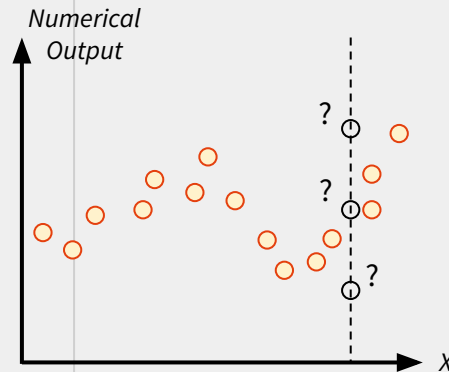
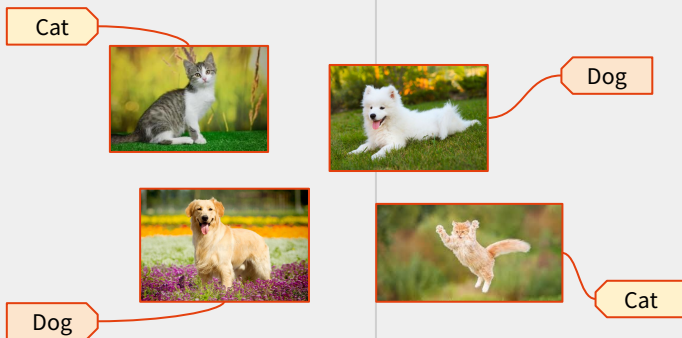


Supervised Learning (2)

Tasks

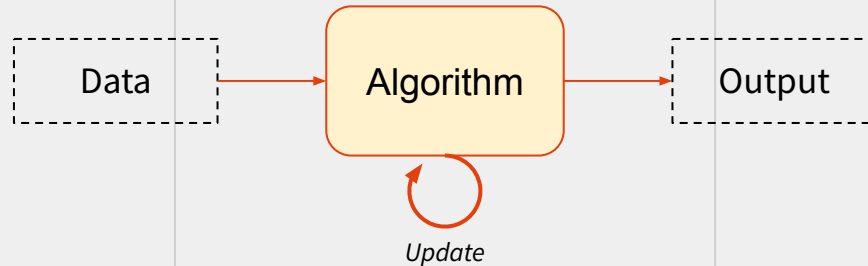
There are 2 main supervised learning tasks :

- **Regression** - *an algorithm learns to predict a continuous numerical output based on a set of input features.*
- **Classification** - *an algorithm learns to predict a categorical label or class based on a set of input features.*



Unsupervised Learning

Unsupervised learning is a type of machine learning where an algorithm learns to **identify patterns and relationships in a dataset without being explicitly told what the correct output should be**. Unlike supervised learning, there is no labeled data provided, so the algorithm must find its own structure and organization in the data.



Unsupervised Learning (2)

There are 4 main unsupervised learning tasks :

- **Clustering** - *an algorithm groups together similar data points based on their features.*
- **Dimensionality Reduction** - *an algorithm reduces the number of features in a dataset while retaining as much relevant information as possible.*
- **Anomaly Detection** - *an algorithm identifies data points that are significantly different from the rest of the data.*
- **Association Rules** - *an algorithm discovers patterns and correlations between items that occur together frequently.*



Semi-Supervised Learning

Semi-supervised learning is a type of machine learning in which an algorithm learns to make predictions based on a **combination of labeled and unlabeled data**. The goal is to leverage the information contained in both the labeled and unlabeled data to improve the performance of the algorithm.

The labeled data is used to guide the learning process, while the unlabeled data is used to identify patterns and relationships that may not be immediately apparent from the labeled data alone. By combining both types of data, the algorithm can learn to make better predictions on new, unseen data.



Reinforcement Learning

Reinforcement learning is a type of machine learning in which an algorithm learns to make decisions in an environment/simulation by **trial and error**. The algorithm receives feedback in the form of **rewards or penalties** for its actions, and uses this feedback to learn how to maximize the total reward it receives over time.

This type of learning is often used in games, robotics, and other complex environments where the optimal strategy is not known in advance.



02

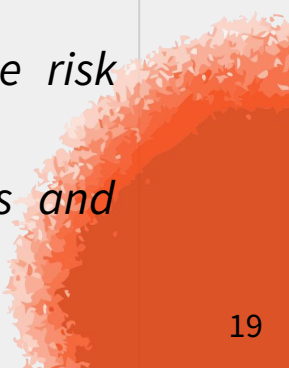
**Problem
Framing & Data
Collection**



Objectives

Every single machine learning project starts with the **definition of the problem and the objectives**.

First, you should always define the objectives in business terms and validate them. Some examples of well-defined objectives are:

- *Predicting customer behavior to target marketing efforts and to retain customers more effectively.*
 - *Identifying fraud to reduce losses and to better manage the risk exposure.*
 - *Improving healthcare outcomes to improve patient outcomes and reduce costs.*
- 



Existing solutions

Most machine learning projects are trying to improve existing solutions. It is always useful to have a good understanding of these solutions:

- It generally shows **how your machine learning algorithm will be used**
- It **inspires ideas and provide insight** into how others have approached similar problems.
- Existing solutions can be used as **good starting point** for new projects, especially if the data or problem is similar.
- Existing solutions can be used as **benchmarks**.



Problem Framing

The problem framing is the part where you need to set the technical landscape of your machine learning problem by answering two questions:

- **What type of machine learning should you use ?** Supervised or Unsupervised ? Both of them ?
- **Is there any performance requirements ?** If so, do you think it is doable given the business objectives ?

These questions are critical as they will help you to choose the best methods to solve the problem.

Data Collection

Data collection is a crucial part of any machine learning project, and involves gathering and preparing the data that will be used to train and validate the machine learning model.

The quality and quantity of the data collected can have a significant impact on the performance of the model, so it's important to approach data collection with care.



Source of data

The sources of data for a machine learning project can vary depending on the specific problem being solved and the type of data required. However, you mostly get your data from:

- **Databases** - *Offline databases, Clouds ...*
- **APIs** - *Many web services and applications provide APIs such as social media platforms, weather services, or financial markets.*
- **Web Scrapping** - *Data is extracted from web pages by parsing the HTML code (news articles, product listings, user reviews ...)*
- **Public Datasets**
- **Measuring devices**

Data Type

Understanding the file types used in a machine learning project is important for working with the data effectively and efficiently. Different file types may require different processing or import techniques, and some file types may be better suited for certain types of data than others. The most common ones include:

- **CSV** - *A simple file format used to represent tabular data*
- **JSON** - *A file format used to store semi-structured or unstructured data in a hierarchical structure.*
- **Image and video files** (JPEG, PNG, MP4, AVI ...)
- **Text file formats** (TXT, PDF, DOC ...)
- **Binary file formats** - *These file formats are used to store non-textual data, such as audio or sensor data.*

Labelling

Sometimes, data is not labelled or contain even more unlabelled data.

Data labeling is the process of **adding annotations or tags to data to provide additional information about its content**. In machine learning, data labeling is often used in supervised learning, where the algorithm is trained on labeled data to make predictions on unlabeled data.

There are several methods for labeling data, depending on the type of data and the complexity of the labeling task. Methods include **manual labeling**, where human experts label the data, and **automatic labeling**, where labeling is performed by algorithms.

03

**Data
Exploration**

Definition

Data exploration is the process of examining and analyzing data in order to **gain a deeper understanding of its structure, patterns, and relationships**. It involves using various statistical and visualization techniques to uncover insights and trends that may not be immediately apparent from the data.

Some common techniques used in data exploration include:

- Dataset analysis
- Summary statistics
- Data visualization
- Unsupervised analysis (clustering, dimensionality reduction ...)



Dataset analysis

The first step is to **analyze the dataset**, especially the features and the labels. Here is a checklist of the questions you should answer:

- How many features ? How many labels ?
- What is the data type of the features ? The labels ?
- Is it a supervised or an unsupervised task ? If supervised, is it a regression or a classification ?
- Is there any unlabelled data ? Are they exploitable ?
- Is there any missing values in the features ?

Features/Labels

A variable can be a feature or a label: it depends on the objective.

	Features		Label
	BMI	Weight	Size
Person 1	21.54	63	171
Person 2	20.94	55	162
Person 3	26.59	94	188

The task is to **predict the size** based on the BMI and the weight of the person. Thus, the features are the BMI and the weight and the label is the size.

	Label	Features	
	BMI	Weight	Size
Person 1	21.54	63	171
Person 2	20.94	55	162
Person 3	26.59	94	188

The task is to **predict the BMI** based on the size and the weight of the person. Thus, the features are the size and the weight and the label is the BMI.

Understanding

It is crucial that you understand the meaning of each feature because:

- Misinterpreting the meaning of a feature can **lead to incorrect analysis and flawed results**.
- **Some features may be irrelevant** to the task and should be discarded.
- Understanding the meaning of each feature can **help in identifying and cleaning erroneous or missing data**.
- When **communicating the results** of an analysis, it is important to be able to explain the meaning of each feature and how it relates to the results.

Understanding (2)

If you don't understand a feature in a dataset, there are several steps you can take to try and solve the issue:

1. Consult the dataset dictionary or **metadata**
2. **Research** the feature **online**
3. **Consult an expert** in that field

If all these steps failed, you can try **analyzing the feature in more depth** to see if you can get any information about its meaning. This could involve looking at summary statistics, creating visualizations, or conducting further data exploration to understand how the feature relates to other variables in the dataset.

Data Types

Every feature (and label) can be classified in one of the two following categories:

- A **numerical** variable
- A **categorical** variable

A dataset generally contains both features types.

	Job	BMI	Weight	Size	Children
Person 1	Engineer	21.54	63	171	0
Person 2	No job	20.94	55	162	2
Person 3	Fireman	26.59	94	188	1

Data Types (2)

Numerical

A **numerical variable** is a type of variable that represents a **numerical quantity or measurement**. Numerical variables can take on a range of numerical values, including whole numbers (integers) or decimal numbers (floats).

Numerical variables can be either **discrete or continuous**. Discrete numerical variables take on only certain values within a finite or countable set, while continuous numerical variables can take on any value within a given range.

Examples of numerical variables include:

Age, Height, Weight, Income, Number of children in a household, Temperature or Time.

Data Types (3)

Categorical

A **categorical variable** is a type of variable that represents a **set of possible values or categories**. Categorical variables can take on a limited number of values or levels, and the values are usually descriptive in nature.

There are several subtypes of categorical variables:

- **Nominal** variables
- **Ordinal** variables
- **Cyclical** variables

Examples of categorical variables include:

Gender, Eye color, Education level, Income, Marital Status or Month of the year.

Data Types (4)

Categorical - Nominal

Nominal variables represent categories that have **no natural ordering or hierarchy**. For example, gender, eye color, or country of origin are all nominal variables.

	Job	Children
Person 1	Engineer	0
Person 2	No job	2
Person 3	Fireman	1

*It is possible to rank the **Children** feature: 2 children is higher than 1 that is also higher than 0.*

*However, it is not possible to rank the **Job** feature because Engineer is neither higher or lower than the Fireman and No job categories.*

Data Types (5)

Categorical - Ordinal

Ordinal variables represent categories that **have a natural ordering or hierarchy**. For example, education level, income level, or customer satisfaction levels are all ordinal variables. The categories can be ranked or ordered based on some underlying property, but the differences between the categories may not be equal.

	Rating /5	Color	Temperature
Instance 1	3	Blue	Hot
Instance 2	2	Green	Cold
Instance 3	5	Red	Freezing

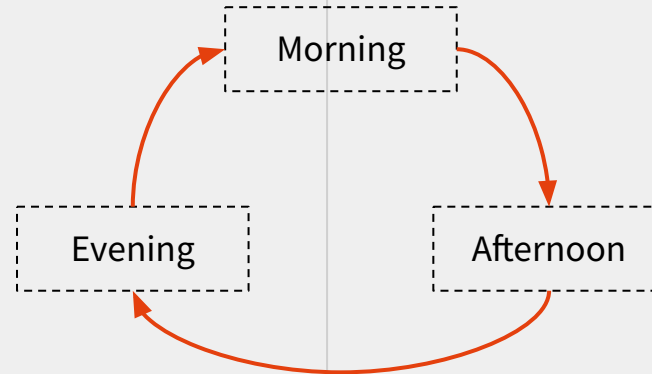
The **Rating** feature can be ranked: 0 is lower than 1 which lower than 2 ...
The **Color** feature cannot be ranked.
The **Temperature** can be ranked: Hot is “higher” than Cold which is “higher” than Freezing.

Data Types (6)

Categorical - Cyclical

Cyclical variables represent categories that **repeat in a cyclical pattern**, such as days of the week or months of the year. The categories can be ordered, but there is no inherent numerical value associated with the categories.

	Month	Time of day
Instance 1	April	Morning
Instance 2	May	Afternoon
Instance 3	June	Evening



Task

The task of a machine learning project depends on the label:

- If there is no label, it is an **unsupervised** task
- If there is one or more label(s), it is a **supervised** task
 - If the label is **categorical**, the task is a **classification**
 - If the label is **numerical**, the task is a **regression**

It is possible to have multiple labels and thus different type of tasks.



Missing data

Missing data refers to the **absence of values for certain labels or features** in a dataset. Missing data can arise for a variety of reasons, such as data entry errors, faulty sensors or equipment, non-response by participants in a survey, or data loss during transmission or storage.

Most of the time, the missing values are either discarded or imputed as we will see in the Data Preprocessing part of this course.

Missing data (2)

Missing Features

In some cases, **missing values in a feature can have a meaning**. In these cases, it may be more appropriate to retain the missing values and **treat them as a separate category** or level within the feature, rather than imputing or deleting them.

Examples :

In survey data, participants may be allowed to skip certain questions if they are not applicable or if they prefer not to answer. In this case, missing values may indicate that the participant did not have a response to that question or chose to skip it intentionally.

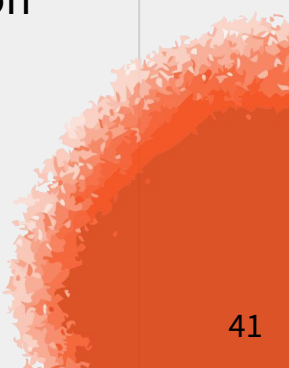
In medical data, missing values for certain symptoms or lab results may indicate that the symptom was not present or that the lab test was not performed for a particular patient.



Summary Statistics

Summary statistics are descriptive statistics that provide a quick overview of a dataset. They are commonly used in data exploration to help analysts get a better understanding of the **structure**, **patterns**, and **distribution** of the data.

Some of the most common summary statistics:

- **Measures of central tendency** - Mean, median and mode
 - **Measures of variability** - Range, variance and standard deviation
 - **Percentiles**
 - **Correlation**
 - **Skewness & Kurtosis**
- 



Summary Statistics (2)

Summary statistics can be used to detect a wide range of characteristics and patterns in a dataset:

- Identifying **outliers**
- Assessing the **spread** of the data
- Providing insights into the **symmetry** or **asymmetry** of the data
- Measuring the strength and direction of linear **relationships** between two variables

Outliers

Outliers are instances that are **significantly different from the other values in the dataset**. Outliers can arise due to measurement errors, data entry errors, or unusual or rare events, and can have a significant impact on the analysis and modeling of the data.

Generally, outliers are detected by **visual inspection** (histograms, scatter plots ...) or **statistical methods** (z-scores, interquartile range, Mahalanobis distance).

The outliers are either discarded or replaced as we will see in the Data Preprocessing part of this course.

Correlation

Correlation is a statistical technique that is used to **measure the strength and direction of the linear relationship between two variables**. Correlation is a useful tool for data exploration and analysis, as it can provide insights into how variables are related to each other and can guide further investigation and modeling.

The most common type of correlation is **Pearson correlation**, which measures the degree of linear association between two variables. Pearson correlation ranges from -1 to 1:

- -1 indicates a perfect negative relationship
- 1 indicates a perfect positive relationship
- 0 indicates no relationship between the variables

Correlation (2)

Correlation can be used to :

- **identify features that are strongly correlated with the labels** - *such features should be analyzed further and carefully preprocessed as they might be extremely important for the task.*
- **identify features that are highly correlated with each other** - *highly correlated features may not provide much additional information and can be removed from the analysis to reduce redundancy.*

Correlation only measures linear relationships between variables. Nonlinear relationships, interactions, and other types of relationships may not be captured by correlation.

Visualization

Data visualization is the process of **representing data in a visual form to facilitate understanding and analysis**. It is an important part of data exploration, as it can help to **reveal patterns, trends, and insights** in the data that may not be immediately apparent from summary statistics or raw data.

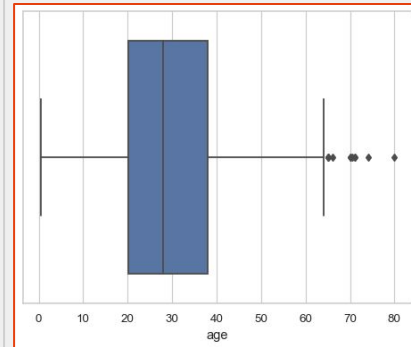
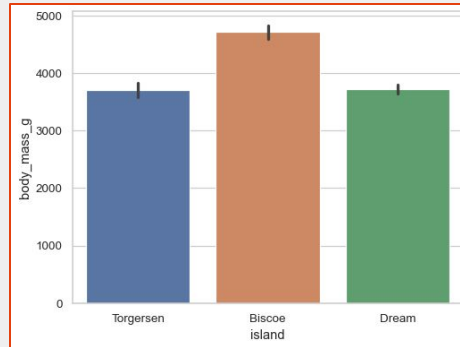
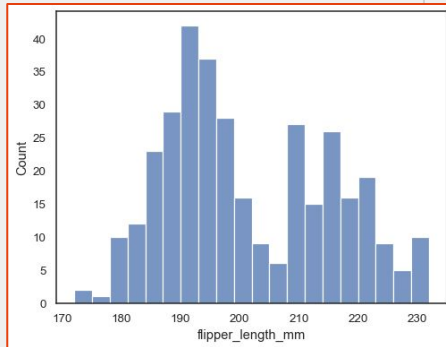
It is generally done in multiple steps:

1. **One**-dimensional analysis of the features
2. **Two**-dimensional analysis
 - a. *Feature-Feature relationships*
 - b. *Feature-Label relationships*
3. **Multi**-dimensional analysis such as *feature-feature-label relationships*

Visualization (2)

One-dimensional analysis is typically used to **explore and understand the distribution of a single variable**. It can be helpful in identifying outliers or the spread of the data.

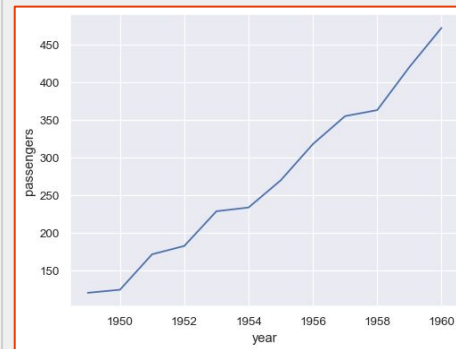
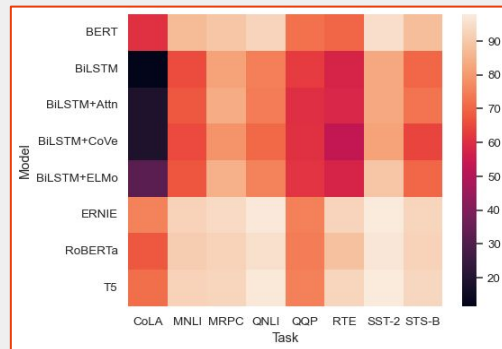
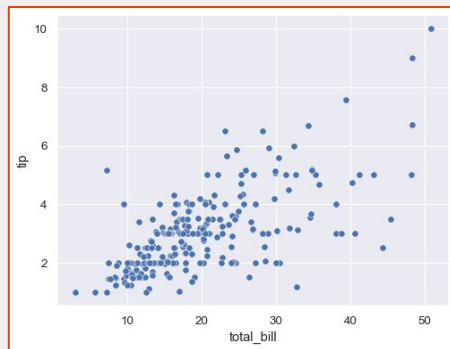
Some of the most common graphs for one-dimensional analysis include histograms, bar graphs, box plots ...



Visualization (3)

Multidimensional analysis, on the other hand, is used to **explore and understand the relationships such as patterns or correlations between multiple variables** in the dataset.

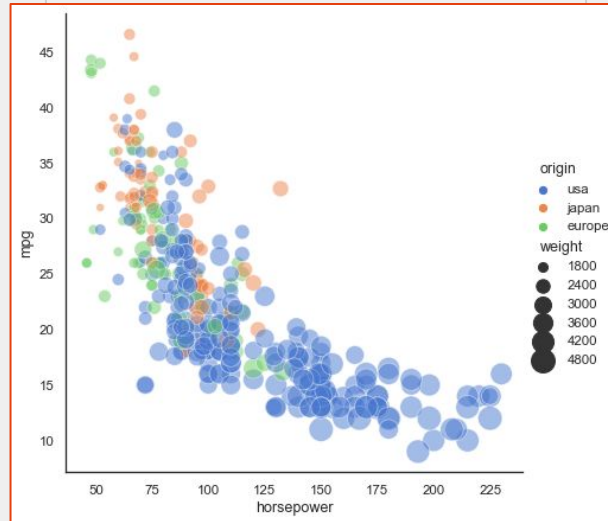
Some of the most common graphs for multi-dimensional analysis include the one-dimensional graph as well as line graph, scatterplots, heatmaps ...



Visualization (4)

Scatter Plot

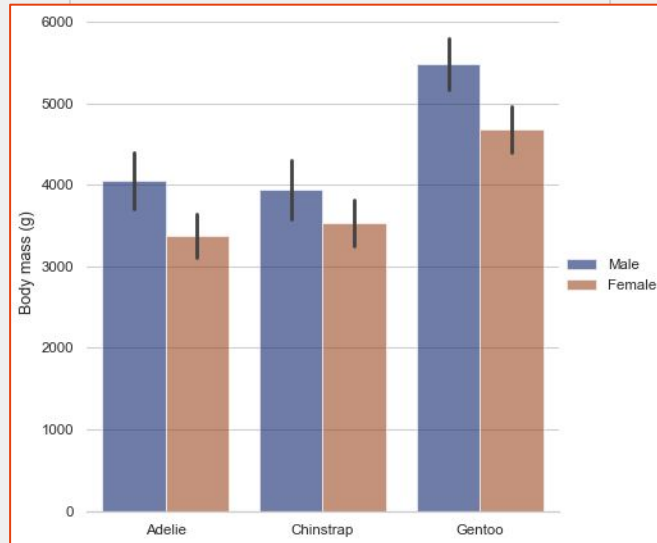
A **scatter plot** is a two-dimensional plot that uses dots to represent the values of two variables. They can be used to identify patterns or relationships between variables, such as **positive** or **negative correlation**, **clusters**, or **outliers**.



Visualization (4)

Bar Charts

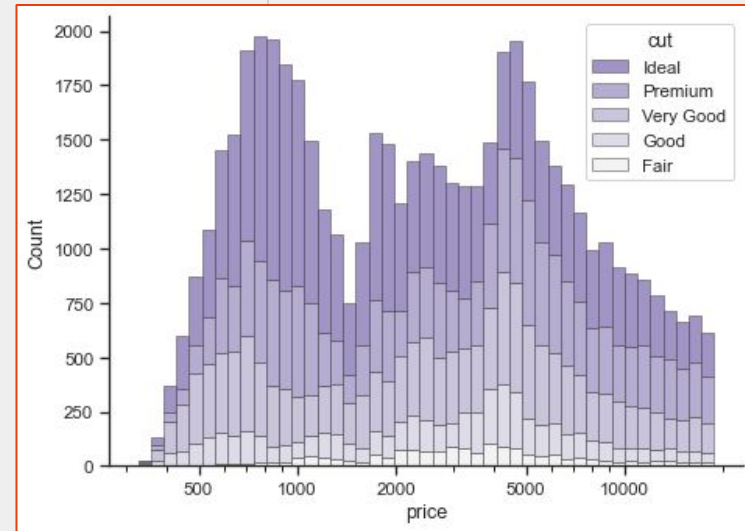
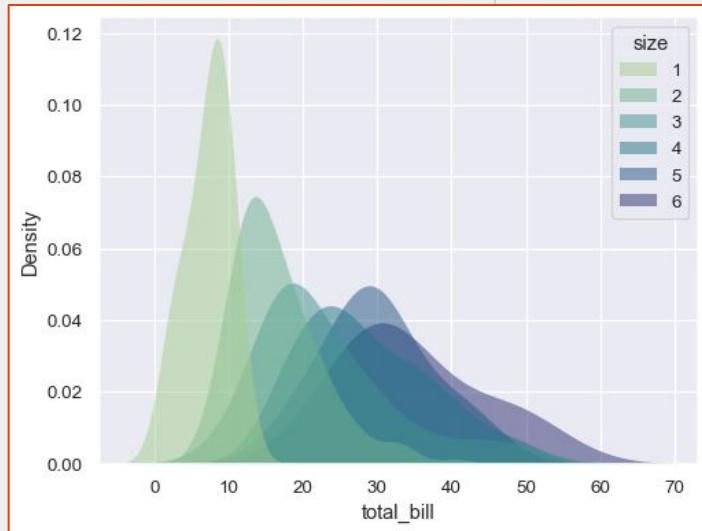
A **bar chart** is a graph that uses rectangular bars to represent the values of different categories. They can be used to **compare the values of different categories or to show changes over time.**



Visualization (5)

Histograms

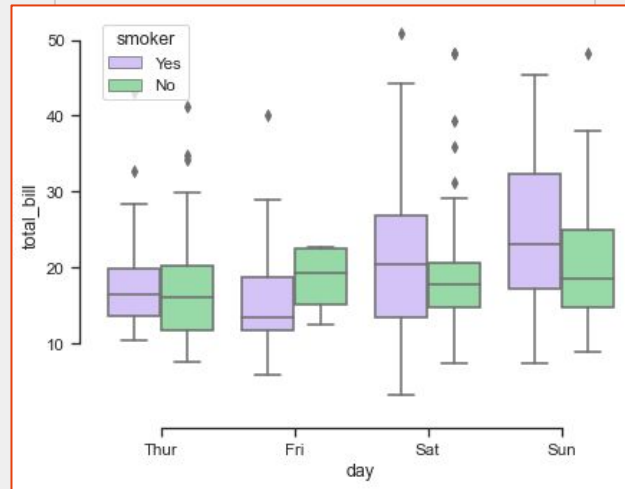
A **histogram** is a graph that uses bars to represent the distribution of a numerical variable. They can be used to **show the shape of the distribution**, such as whether it is skewed, bimodal, or normal.



Visualization (6)

Box Plots

A **box plot** is a graph that shows the distribution of a numerical variable and highlights important summary statistics such as the median, quartiles, and outliers. They can be used to **compare the distributions of different groups** or **to identify outliers or extreme values**.



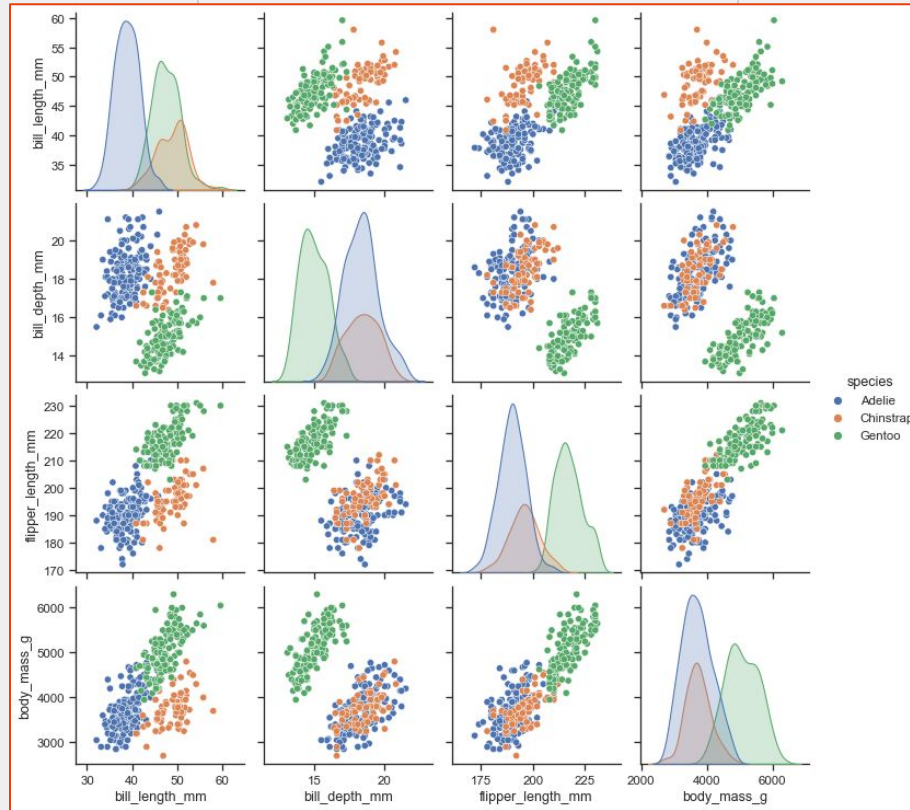
Visualization (7)

Heatmaps

A **heatmap** is a graphical representation of data where the values are represented by colors in a two-dimensional grid. They can be used to **identify patterns or clusters in the data**, particularly in large datasets or when **visualizing correlations**.



Visualization (8)



04

**Data
Preprocessing**

Definition

Data preprocessing is the process of **cleaning**, **transforming**, and **preparing** data to produce a high-quality dataset that can be used for data analysis or machine learning tasks.

It involves various techniques such as handling **missing data**, dealing with **outliers**, correcting data **errors**, **transforming** data into a more structured format, **reducing the size** of the dataset, and **scaling** or **normalizing** the data.

The quality of the dataset has a significant impact on the performance machine learning models, and data preprocessing is essential for ensuring the performance and reliability of the results obtained from the data.

Splitting

Data splitting is **one of the first thing to do in a Machine Learning project**. It is the process of dividing a dataset into two or more subsets for the purpose of training, testing, and validating machine learning models :

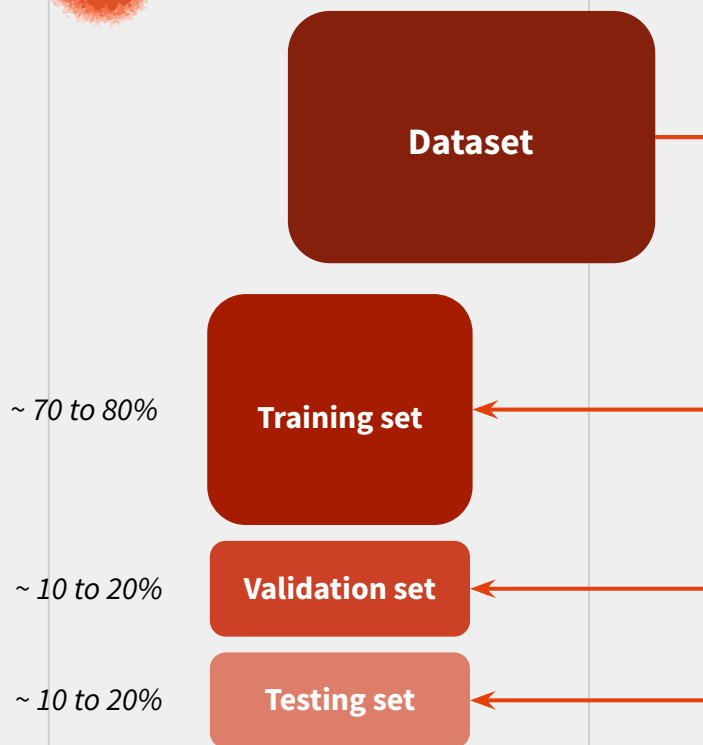
- **Training** subset - *used to train the model*
- **Validation** subset - *used to prevent the overfitting problem*, to select the best models and to tune their hyperparameters***
- **Testing** subset - *used to evaluate the overall performance*** of the machine learning models → it simulates the future data*

* see chapter 8 in this course

** see chapter 9 in this course

*** see chapter 7 in this course

Splitting (2)



The most common way to split a dataset is to **randomly divide** it into two or more subsets. It is important to ensure that the data splitting process is **representative of the original dataset**.

Biased data splitting can result in incorrect or misleading results, which can negatively impact the performance of the machine learning model.

Splitting (3)

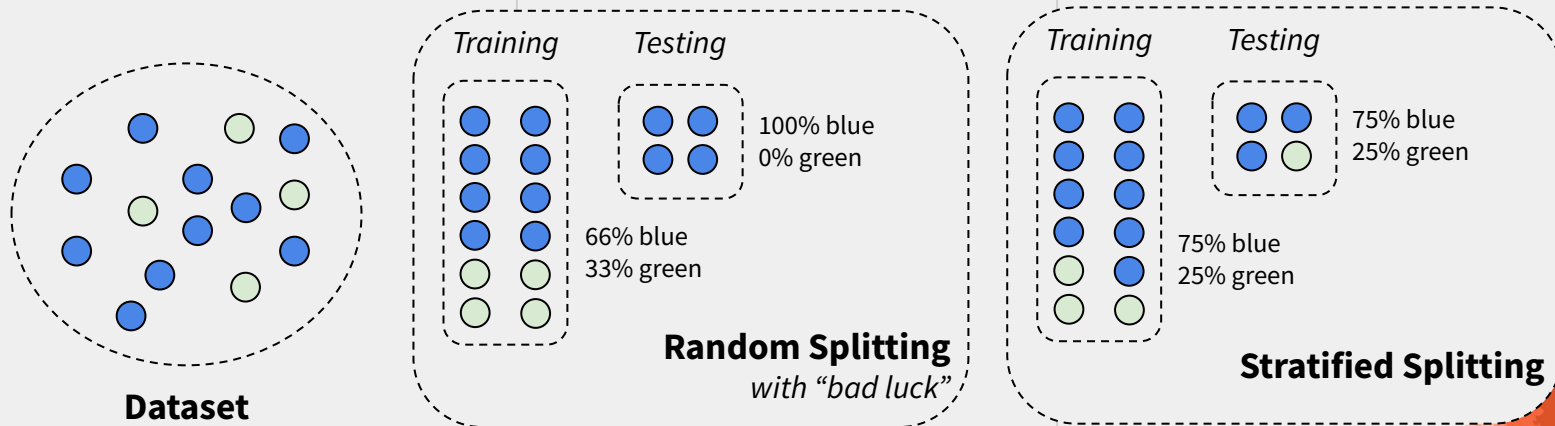
There are other ways to split data including:

- **Stratified Splitting** - the dataset is divided into subsets based on the values of a categorical variable, generally the label.
- **Time-Based Splitting** - the dataset is divided into subsets based on time, such as the year, month, or day.
- **K-Fold Splitting** - the training set is subdivided into multiple sets that are used as validation independently
- **Leave-One-Out Splitting** - An extreme K-fold variant where each instance is used as a validation set independently
- **Group-based Splitting** - A k-fold variant where the training set is subdivided into multiple sets based on a common characteristic

Splitting (4)

Stratified Splitting

Stratified splitting is commonly used when the dataset is **imbalanced** or when you want to ensure that the distribution of the target variable is similar in the subsets.



Cleaning

Data cleaning is the process of **correcting errors**, **inconsistencies**, and **inaccuracies** in data before it is used for further analysis. The main goal is to ensure that the data is accurate, consistent, and complete, which is essential for obtaining reliable and meaningful insights from the data.

It involves many techniques including:

- Handling **missing data**
- Handling **outliers**
- Correcting **data errors** & **inconsistencies**
- Removing **duplicates**

Missing Data

There are two options to deal with missing values:

- **Deleting** - it removes the rows or columns with missing values.
- **Imputing** - it replaces the missing values with a new value.

	Job	BMI	Weight	Size	Children
Person 1	Engineer	21.54	63	NaN	NaN
Person 2	No job	NaN	55	162	2
Person 3	Fireman	26.59	94	188	NaN

Missing Data (2)

Deleting

Deleting missing values should be considered:

- If the missing values are only a **small proportion of the dataset** (e.g. *less than 10/20%*) AND if the missing values are **completely random** (i.e. *missing values are unrelated to a variable*).
 - In that case, the **rows/instances** that contain missing values should be deleted.
- If a **variable has a large proportion of missing values** (e.g. *more than 70/80%*) AND if **this variable is NOT related to other variables**.
 - In that case, the **columns/features** that contain missing values should be deleted.

Missing Data (3)

Imputing

Imputing missing data can be considered as an alternative to deleting missing data when the **missing data is not missing at random**, when the **missing data is a key variable**, when the **analysis requires complete cases** or simply when the **dataset is small**.

There are several methods for imputing missing values:

- **Statistical** imputation
- **Supervised Machine Learning** imputation
- **Unsupervised Machine Learning** imputation

Missing Data (4)

Imputing

The **statistical** imputation is done using by setting a fixed value that can be :

- A **constant** value (e.g. 0)
- The **mean** of the feature - *for numerical variables - it assumes that the data is normally distributed and the missing values are randomly distributed.*
- The **median** of the feature - *for numerical variables - it is robust to outliers and does not assume normal distribution.*
- The **mode** (most common value) of the feature - *it is suitable for categorical variables.*

Additionally, **missing values can be treated as a new category for categorical variables.**

Missing Data (5)

Imputing

The supervised machine learning imputation can be done using any supervised machine learning model to predict the missing values.

The dataset is divided into:

- a **training set** that contains the cases with **complete data**
- a **test set** that contains the cases with the **missing values**

A machine learning model is then trained on the training set using the other variables as input features and the **variable with missing values as the target variable**.

Missing Data (6)

Imputing

Training Set

	X_1	X_2	X_3
Instance 1	3	Blue	Hot
Instance 2	2	Green	Cold
Instance 3	5	Red	Freezing
...	...		
Instance n	4	Blue	Warm

Testing Set

	X_1	X_2	X_3
Instance 1	2	Green	NaN
Instance 2	0	Red	NaN
Instance 3	4	Blue	NaN
...	...		
Instance n	1	Green	NaN

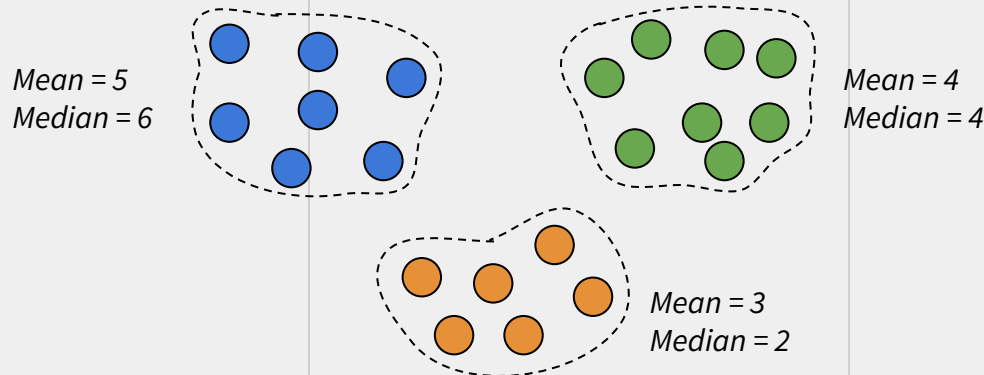
X_3 is the column that contains the missing values (NaN).
Thus, it is used as the target of the machine learning model
in order to predict the missing values.

Missing Data (7)

Imputing

The **unsupervised machine learning** imputation can be done using any unsupervised machine learning model to **identify relationships between the features** and use them to impute the missing values.

It is generally done by grouping the instances based on the features and then imputing the missing values based on statistics computed on these groups.



Missing Data (8)

Pros & Cons

	Pros	Cons
Statistical	<ul style="list-style-type: none">• Simple and easy to implement• Works with any dataset size	<ul style="list-style-type: none">• May not be suitable for missing values that are not random• May underestimate the variability of the data and produce imputed values that are too similar to each other
Supervised ML	<ul style="list-style-type: none">• Generally, more accurate than statistical imputation	<ul style="list-style-type: none">• Requires a large amount of data• May not be suitable for datasets with a large proportion of missing values• May not be suitable for missing values that are not related to other variables in the dataset.
Unsupervised ML	<ul style="list-style-type: none">• Generally, more accurate than statistical imputation• Effective when the missing values are random	<ul style="list-style-type: none">• May not be suitable for missing values that are not random.• May not be suitable for missing values that are not related to other variables in the dataset.

Outliers

Outliers can be either :

- **Deleted** - it removes the rows or columns with outliers.
- **Imputed** - it replaces the outliers with a new value.
- **Kept** - it keeps the outliers in the data

	Job	BMI	Weight	Size	Children
Person 1	Engineer	21.54	63	15	18
Person 2	No job	256.24	55	162	2
Person 3	Fireman	26.59	94	188	0

Outliers (2)

Outliers can be detected and removed using multiple methods including:

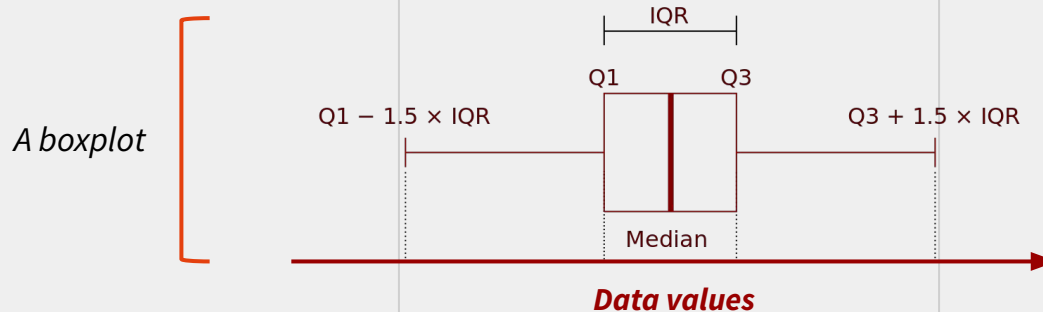
- **Percentile trimming** - *it removes the outliers based on the lower and upper percentiles of the dataset (e.g. removing the 5% lowest and highest values).*
- **Interquartile range** - *it removes the outliers based on the range between the 1st and the 3rd quartiles of the dataset.*
- **Z-score** - *it removes the outliers that are a number of standard deviations away from the mean of the data.*

Outliers (3)

Interquartile range

The interquartile range (IQR) works as follows:

1. Sort the data in ascending order.
2. Calculate the 1st quartile (Q1) and the 3rd quartile (Q3) of the data.
3. Calculate the interquartile range: **$IQR = Q3 - Q1$** .
4. Removes the values that are more than 1.5 times the IQR away from the 1st quartile (Q1) and the 3rd quartile (Q3).



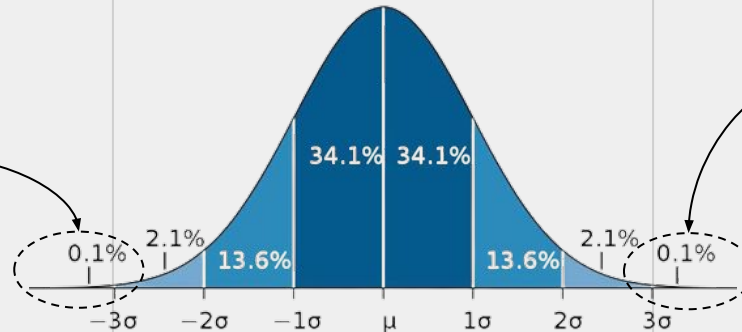
Outliers (4)

Z-score

The Z-score works as follows:

1. Calculate the mean (μ) and the standard deviation (σ) of the data.
2. For each instance x , calculate the Z-score: $z = (x - \mu) / \sigma$
3. Remove the values that have a z-score greater than a chosen thresholds (*generally, 3 or 4*).

0.1% of the values of a unit normal distribution have a z-score **lower** than 3



0.1% of the values of a unit normal distribution have a z-score **higher** than 3

Outliers (5)

Imputation & passthrough

Outliers can also be **imputed** using the **winsorization** method which replaces the extreme values in a dataset with a less extreme value, such as the 5th or 95th percentile value.

In some cases, outliers should not be removed from the dataset. In general, **they should be kept when they represent important and informative instances** that are consistent with the underlying process being studied i.e. when they are not generated by an error or a mistake.

When outliers are kept in the data, robust statistical methods can be used to reduce the effect of the outliers on the preprocessing steps and the models.

Outliers (6)

Categorical outliers

In categorical data, **outliers may occur when there is a rare category** that appears much less frequently than the other categories, when there are data entry errors or when a category is incorrectly assigned.

Here are some common methods for handling outliers in categorical data:

- **Merge** categories - *A category outlier in a categorical variable could be merge within another category which is not an outlier.*
- **Create a new “rare” category** - *Several outliers in a categorical variable could be merged together to create a new category.*

Data Errors & duplicates

Data errors should be **handled as outliers if the value makes sense or as missing values if it doesn't.**

Duplicates should be **removed from the dataset.**

Numerical data

Numerical features can be transformed using various techniques to improve the performance of models. There are three main ways to transform a numerical feature:

- **Scaling** transformation
- **Math** transformation
- **Binning** transformation

It is very important to evaluate the impact of any transformation on the models, as some transformations may introduce bias or distort the relationships between variables.

Numerical data (2)

Scaling

Scaling is a technique for transforming numerical data to a **common range**. It involves applying a mathematical transformation to the data that changes its range without changing its shape or distribution.

Having the same range is particularly important in statistical analysis and machine learning models, where **differences in the magnitude of the data can lead to biased results or inaccurate predictions**.

Some of the most common scaling techniques include:

- **Max absolute** Scaling
- **Min-Max** scaling
- **Standardization**
- **Robust** Scaling

Numerical data (3)

Max-Absolute scaling

Max absolute scaling scales the data to a range between -1 and 1 by dividing each data point by the maximum absolute value of the data:

$$X = \frac{X}{|X_{max}|}$$

Max absolute scaling can be useful **when the magnitude of the data is not important, but the direction of the data is important.**

Numerical data (4)

Min-Max scaling

Min-Max scaling scales the data to a range between 0 and 1 by using the minimum and the maximum values of the data:

$$X = \frac{X - X_{min}}{X_{max} - X_{min}}$$

Min-max scaling can be useful **when the magnitude of the data is important, but the direction of the data is not important.**

Numerical data (5)

Standardization

Standardization scales the data to have a mean of 0 and a standard deviation of 1:

$$X = \frac{X - X_{mean}}{X_{std}}$$

Standardization can be useful **when the magnitude and direction of the data are both important**. For example, in some machine learning algorithms, standardization is necessary to ensure that all variables contribute equally to the model.

Numerical data (6)

Robust Scaling

Robust scaling is a technique for scaling numerical data using the median and interquartile range (IQR):

$$X = \frac{X - X_{median}}{IQR}$$

Robust scaling can be useful **when the data contains outliers or is not normally distributed**. It preserves the direction of the data, but not necessarily the magnitude because the relative position of each instance with respect to the median is not preserved.

Numerical data (7)

Math transformation

A lot of possible **math transformation** can be applied to numerical data to improve the distribution or reduce skewness, and ultimately improve the performance of machine learning models. The three main ones are:

- **Power** transformation - *it is useful when the data has a skewed distribution, with a long tail on one side.*
- **Log** transformation - *it is useful when the data has a skewed distribution, especially with a long right tail.*
- **Box-Cox** transformation - *it is useful when the data has a non-normal distribution, and it can be more effective than other transformation methods.*

Numerical data (8)

Boxcox

The **Box-Cox** transformation is a method for transforming non-normal data to a normal distribution:

$$B(X, \lambda) = \begin{cases} \frac{X^\lambda - 1}{\lambda} & \text{if } \lambda \neq 0 \\ \ln(X) & \text{if } \lambda = 0 \end{cases}$$

The optimal lambda parameter for the Box-Cox transformation can be found by maximizing the log-likelihood function which measures the goodness of fit of the transformed data to a normal distribution.

Numerical data (9)

Binning

Binning is a technique for **transforming continuous numerical data into categorical data** by grouping the data into intervals, or "bins". There are two main types of binning:

- **Equal-width binning** - *it divides the range of the data into equal-width intervals (e.g. 0-20, 20-40, 40-60, 60-80, and 80-100). It may not be appropriate for non-uniform distributions.*
- **Equal-frequency binning** - *it divides the data into intervals with equal numbers of instances (e.g. 10 instances in the 0-5 bin, 10 instances in the 5-20 bin and 10 instances in the 20-100 bin). It is more appropriate non-uniform distributions, as it can ensure that each bin contains a representative sample of the data.*

Numerical data (10)

Binning

Numerical Values
1
2
5
4
2
9
3
7
6



Binned <i>Equal-width</i>
1-2-3
1-2-3
4-5-6
4-5-6
1-2-3
7-8-9
1-2-3
7-8-9
4-5-6

4 x "1-2-3"
3 x "4-5-6"
2 x "7-8-9"

Binned <i>Equal-frequency</i>
1-2
1-2
3-4-5
3-4-5
1-2
6-7-8-9
3-4-5
6-7-8-9
6-7-8-9

3 x "1-2"
3 x "3-4-5"
3 x "6-7-8-9"

Categorical data

Machine learning models require **categorical data to be transformed into a numerical format** in order to use them because they cannot be directly compared or used in mathematical operations.

However, the transformation methods for categorical data are generally different from those used for numerical data because they do not have the same mathematical properties.

For example, it is not appropriate to calculate the mean or standard deviation of nominal categorical data as its categories do not have a numerical value.

There are several methods to transform categorical data, including **One-hot encoding**, **Ordinal encoding** or **Target encoding**.

Categorical data (2)

One Hot Encoding

One-Hot encoding creates a **binary column for each unique category** in the categorical variable.

*One column for each category
in the Temperature feature*

	Temperature
Instance 1	Hot
Instance 2	Cold
Instance 3	Freezing
Instance 4	Hot
Instance 5	Freezing
Instance 6	Lava Hot

→
One Hot Encoding

	Lava Hot	Hot	Cold	Freezing
Instance 1	0	1	0	0
Instance 2	0	0	1	0
Instance 3	0	0	0	1
Instance 4	0	1	0	0
Instance 5	0	0	0	1
Instance 6	1	0	0	0

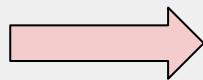
Only one 1 per row

Categorical data (3)

Ordinal Encoding

Ordinal encoding assigns a **numerical value to each category** in the categorical variable.

	Temperature
Instance 1	Hot
Instance 2	Cold
Instance 3	Freezing
Instance 4	Hot
Instance 5	Freezing
Instance 6	Lava Hot



Ordinal Encoding

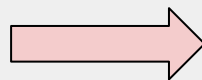
	Temperature
Instance 1	2
Instance 2	1
Instance 3	0
Instance 4	2
Instance 5	1
Instance 6	3

Categorical data (4)

Target Encoding

Target encoding, also known as mean encoding, is a method for **encoding categorical variables as numerical values based on the mean of the target variable** for each category.

	Temperature	Target
Instance 1	Hot	0
Instance 2	Cold	2
Instance 3	Cold	1
Instance 4	Hot	0
Instance 5	Hot	1
Instance 6	Cold	3



Target Encoding

	Temperature	Target
Instance 1	0.33	0
Instance 2	2	2
Instance 3	2	1
Instance 4	0.33	0
Instance 5	0.33	1
Instance 6	2	3

Mean of Hot $\rightarrow (0+0+1) / 3 = 0.33$

Mean of Cold $\rightarrow (1+2+3) / 3 = 2$

Categorical data (5)

Target Encoding

Target encoding can be useful because it **captures the relationship between the categorical variable and the target variable**, and it can improve the performance of some machine learning algorithms.

However, **target encoding can also lead to overfitting***, especially if there are many categories or if some categories have very few observations. In addition, target encoding can be sensitive to outliers or imbalanced classes. To mitigate the risk of overfitting, **cross-validation** should be applied**.

* see chapter 8 in this course

** see chapter 6 in this course

Categorical data (6)

Ordinal & Cyclical categorical variables

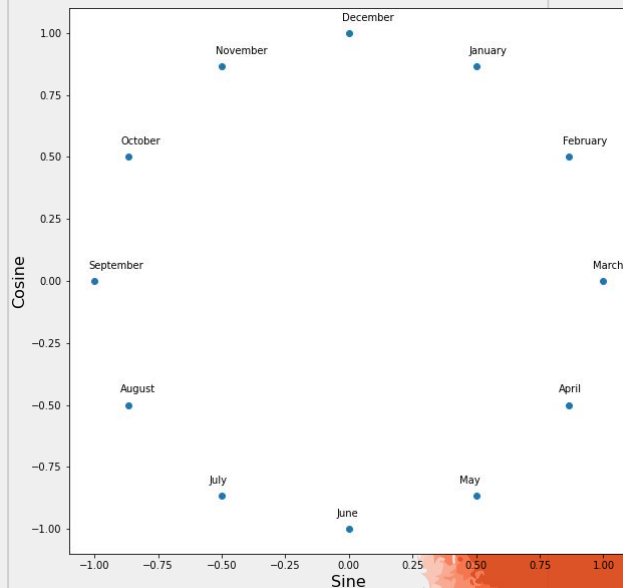
Ordinal categories can be transformed using numerical (scaling ...) or categorical (one hot encoding, ordinal encoding ...) methods.

Cyclical categories can be transformed using the categorical methods but it is generally preferred to use a cyclical encoding method. Each **cyclical variable** is represented by two numerical variables, **sine** and **cosine**, that capture the periodic nature of the variable

$$X_{sin} = \sin\left(\frac{2\pi X}{n}\right)$$

$$X_{cos} = \cos\left(\frac{2\pi X}{n}\right)$$

The period (e.g. 7 for the days of week, 12 for the months, 24 for the hours ...)



Training, Validation, Testing

It is crucial to compute the preprocessing steps, such as scaling or encoding, on the training set and to apply, not compute, the same transformations to the validation and testing sets.

This is very important in order to:

- **Avoid data leakage** - By computing the preprocessing on the training set and applying the same steps to the validation and testing sets, we ensure that the model is not using any information from the validation or testing sets during training.
- **Ensure consistency** - By applying the same preprocessing steps to all sets, we ensure that the data is consistent and the model is making fair comparisons across all sets.
- **Simulate real-world performance** - Preprocessing on the training set and applying the same transformations to the validation and testing sets simulates real-world scenarios where the model must make predictions on new, unseen data that it has not been trained on.

Training, Validation, Testing (2)

Missing data and outliers

Missing Data

*If missing data is present in the validation or testing sets, **the missing values should be imputed using the same method used on the training set**. This ensures that the imputed values are consistent across all datasets and reduces the risk of introducing bias into the analysis.*

Outliers

*If outliers are identified in the validation or testing sets, they should not be removed or imputed separately from the training set. Instead, **the same outlier detection and handling methods used on the training set should be applied to the validation and testing sets** to ensure consistency and reduce the risk of overfitting.*

Training, Validation, Testing (3)

Encodings

If **new categories are present in the validation or testing sets** that were not present in the training set, it is important to handle them carefully to avoid introducing bias or errors in the model. Here are some possible strategies:

- **Ignore** the new categories - *If the new categories are rare or unimportant, it may be appropriate to ignore them and treat them as missing values.*
- **Assign** a default value - *If the new categories are important, but there is no corresponding encoding value in the training set, it may be appropriate to assume that the new categories are similar and therefore to assign a default value.*
- **Retrain** the model - *If the new categories are numerous or significant, it may be necessary to retrain the model with the new categories included in the training set.*

05

Training a model

Definition

In machine learning, a **model** typically consists of **a set of parameters** that are **learned** from a **training dataset** using **an optimization algorithm**, and then used to **make predictions on new data**.

The process of training a model differs depending on whether it is a supervised or an unsupervised learning problem. However, it generally comes to **optimize a loss function** by adjusting the model's parameters using an optimization algorithm.

Models

There are several main classes of supervised machine learning models:

- **Linear models** - *it makes predictions by fitting a linear equation to the input features*
- **Decision trees** - *it makes predictions by following a sequence of binary decisions based on the input features*
- **Support Vector Machines** - *it makes predictions by finding the best boundary that separates the different classes in the data*
- **Similarity models** - *it makes predictions based on the similarity between the instances.*
- **Bayesian Networks** - *it is a probabilistic model that represents a set of variables and their conditional dependencies through a graph network*




Parameters

The **parameters** or **weights** are the values that the model learns during training. The specific parameters or weights depend on the type of model being used, for example:

- In **linear models** and **Support Vector Machines** the weights correspond to the coefficients of the input features and the bias term
- In **decision trees** the parameters are the split points for each feature

Example for a Multiple Linear Regression

$$y = \theta_1 X_1 + \theta_2 X_2 + \theta_3 X_3$$

	<i>parameters</i> θ_1	θ_2	θ_3
			
	Feature n°1	Feature n°2	Feature n°3
Instances

Loss Function

The goal of training most machine learning models is to **find the optimal set of parameters that minimize the difference between the predicted values and the actual values.**

The choice of loss function depends on the type of machine learning task being performed, the model being used and its associated optimization algorithm. The two most common loss functions are:

- **Mean Squared Error** - *used for regression tasks*
- **Cross-Entropy** - *used for classification tasks*

However, not all machine learning models use an explicit loss function such as the similarity models and the bayesian networks. Others such as Decision trees and Support Vector Machine models use very specific losses.

Optimization Algorithm

There are several optimization algorithms used in machine learning, and the choice of algorithm depends on the type of machine learning task being performed and the specific requirements of the algorithm.

Here are some of the most common ones:

- **Gradient Descent** - *it is a first-order optimization algorithm works by iteratively updating the model parameters in the direction of the negative gradient of the loss function.*
- **Quasi-Newton method** (such as BFGS) - *it is a second-order optimization algorithm that works by approximating the inverse Hessian matrix (a matrix of second-order partial derivatives of the loss function that provides information about the curvature of the loss function)*

Selecting Models

Choosing an appropriate algorithm for a specific task involves:

- **Understanding the characteristics** of different models - *some algorithms may be better suited for the task (classification, regression ...) and/or the data (small or large number of features and instances, noisiness, sparsity ...)*
- **Comparing** their respective **performances** - *models learn differently from the data and thus perform differently*

The models should be **trained on the trained set** while their respective **performances should be measured on the validation set**. The testing set is kept for the last step: to assess the generalization ability of the final model on new/unseen data.

Selecting Models (2)

Cross-Validation

Cross-validation is a technique used in machine learning to evaluate the performance of a model. It is generally used on limited datasets.

It involves **splitting the dataset into multiple subsets**, also called folds, and **using each fold as a validation set while training the model on the remaining folds**.

This process is repeated several times, with each fold used once as the validation set and the remainder used as the training set. The results are then averaged across the different folds to obtain an estimate of the model's performance. It helps to reduce the variability of the estimate and increase its reliability.

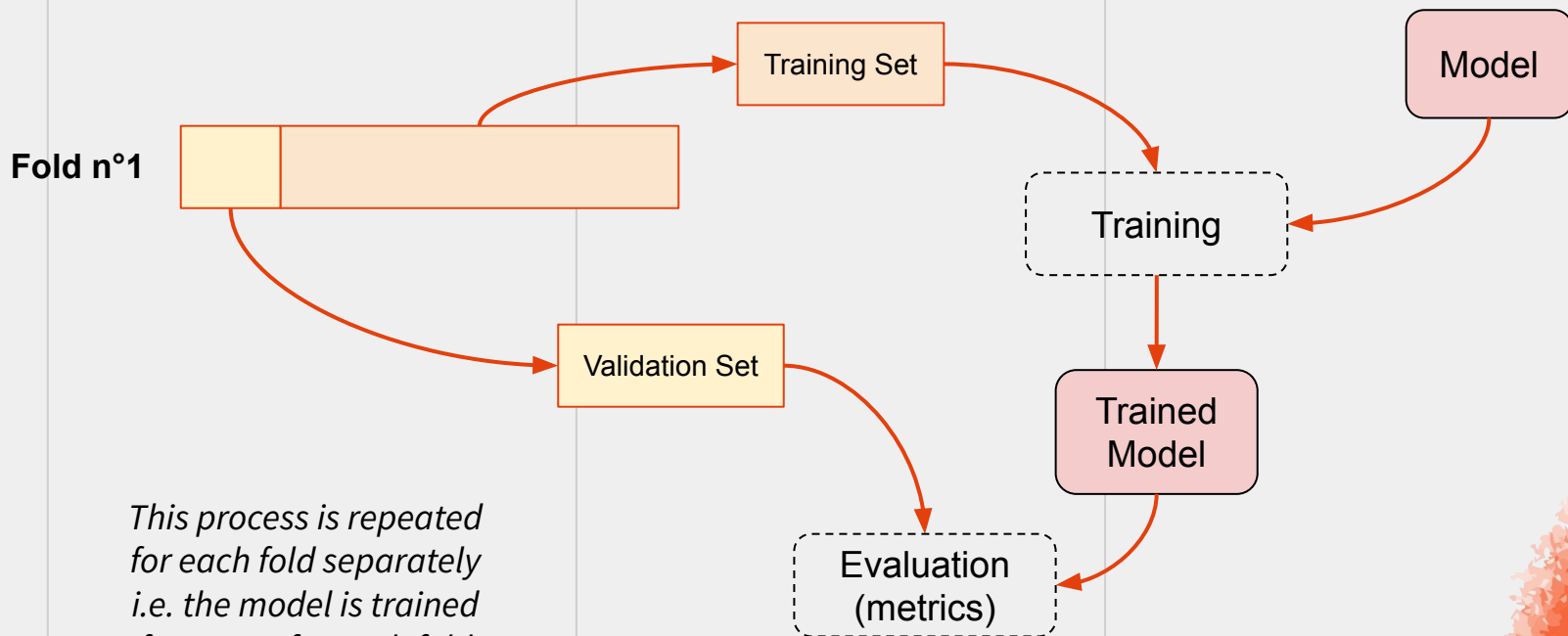
Selecting Models (3)

k-fold cross-Validation

	Training Set				Testing Set
Fold n°1	Validation Set	Training Set	Training Set	Training Set	Training Set
Fold n°2	Training Set	Validation Set	Training Set	Training Set	Training Set
Fold n°3	Training Set	Training Set	Validation Set	Training Set	Training Set
Fold n°4	Training Set	Training Set	Training Set	Validation Set	Training Set
Fold n°5	Training Set	Training Set	Training Set	Training Set	Validation Set

Selecting Models (4)

k-fold Cross-Validation





06

Performances & Metrics

Definition

The **performance** of a machine learning model is typically measured by its **ability to accurately predict or classify new instances that the model has not seen before**. This can be quantified using various metrics, depending on the type of problem and the specific goals of the model.

A **metric** is a **quantitative measure that is used to evaluate the performance of a model**. Metrics are typically used to assess how well a model is able to generalize to new, unseen data and how well it is able to achieve its intended objective.

Types

There are five main category of metrics:

- **Classification** metrics - *used where the goal is to predict a categorical label for each instance.*
- **Regression** metrics - *used where the goal is to predict a continuous numerical value.*
- **Clustering** metrics - *used to evaluate the performance of clustering algorithms by assessing the “quality” of the groups.*
- **Ranking** metrics - *used where the goal is to rank items based on their relevance or importance.*

Several domain-specific metrics also exist such as the Natural Language Processing specific or the Computer Vision specific metrics.

Classification

True/False Positive/Negative

In binary classification problems, there are typically two classes:

- **Positive** class - *it is the class of interest, the one that the model is trying to predict.*
- **Negative** class - *it is the complement of the positive class, meaning it is the class that the model is not interested in predicting.*

The performance of the classification of these two classes can be measured by 4 complementary metrics : the **True Positive** (TP), the **True Negative** (TN), the **False Positive** (FP), and the **False Negative** (FN).

These metrics are the base of many other metrics such as the accuracy, the recall, the precision or the f1-score.

Classification (2)

True/False Positive/Negative

True Positive

measures when the model correctly predicts the positive class as positive.

*Prediction $\rightarrow 1$
Truth $\rightarrow 1$*

True Negative

measures when the model correctly predicts the negative class as negative.

*Prediction $\rightarrow 0$
Truth $\rightarrow 0$*

True Positive

True Negative

False Positive

False Negative

False Positive

measures when the model predicts the positive class but it is actually negative.

*Prediction $\rightarrow 1$
Truth $\rightarrow 0$*

False Negative

measures when the model predicts the negative class but it is actually positive.

*Prediction $\rightarrow 0$
Truth $\rightarrow 1$*

Classification (3)

Confusion Matrix

A **confusion matrix** is a table that is often used to evaluate the performance of a binary classification model. The matrix **displays the number of true positives, false positives, true negatives, and false negatives** that the model has predicted.

Generally, the rows of the confusion matrix correspond to the actual classes, while the columns correspond to the predicted classes.

	Predicted Positive	Predicted Negative
Actual Positive	<i>True Positive</i>	<i>False Negative</i>
Actual Negative	<i>False Positive</i>	<i>True Negative</i>

Classification (4)

Confusion Matrix

	Predicted Positive	Predicted Negative
Actual Positive	150	0
Actual Negative	0	200

An **excellent** confusion matrix

	Predicted Positive	Predicted Negative
Actual Positive	75	75
Actual Negative	0	200

An **mediocre** confusion matrix - the model struggles with the **positive** classe

	Predicted Positive	Predicted Negative
Actual Positive	150	0
Actual Negative	100	100

An **mediocre** confusion matrix - the model struggles with the **negative** classe

	Predicted Positive	Predicted Negative
Actual Positive	75	75
Actual Negative	100	100

An **poor** confusion matrix - the model struggles with the both classes

Classification (5)

Accuracy

The most famous classification metrics is **Accuracy**. It is a classification metric that **measures the proportion of correctly classified instances among all samples in the dataset**.

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

Correctly classified instances (pointing to $TP + TN$)

All the instances (pointing to $TP + TN + FN + FP$)

It ranges from 0 to 1, where 1 indicates a perfect classifier, and 0 indicates a completely random classifier.

Classification (6)

Accuracy

Unfortunately, **accuracy can be misleading in imbalanced datasets** because it does not take into account the class distribution of the dataset. In other words, **if one class is much larger than the other, the classifier can achieve a high accuracy by simply predicting the majority class all the time.**

For example, the classification of a dataset with 99% negative instances and 1% positive instances should not be evaluated with accuracy because the model can always predict the negative class. The accuracy of this model would then be 99%, which seems high, but it is not useful for the task of identifying positive instances.

Therefore, the **accuracy should only be used with balanced datasets** i.e. datasets with a similar proportion of positive and negative classes.

Classification (7)

Precision & Recall

Other metrics such as the **Precision** and the **Recall** can be used with imbalance datasets.

Precision measures the **fraction of true positive predictions among all instances that the model predicted as positive**:

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

Recall measures the **fraction of true positive predictions among all positive instances in the dataset**:

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

Classification (8)

Precision & Recall

Precision is useful when the **false positives should be as low as possible** whereas **recall** is useful when **false negatives should be as low as possible**.

When both precision and recall are used simultaneously, they take into account the class distribution and can provide a more informative evaluation of the model's performance on the minority class (*i.e. the less frequent class*).

	Predicted Positive	Predicted Negative
Actual Positive	20	30
Actual Negative	10	200

$$\text{Accuracy} = (20+200) / (20+200+30+10) = \mathbf{84.6\%}$$

$$\text{Precision} = 20 / (20 + 10) = \mathbf{66\%}$$

$$\text{Recall} = 20 / (20 + 30) = \mathbf{40\%}$$

Classification (9)

F1-Score

F1-score is the harmonic mean of precision and recall and **provides a way to balance the two metrics**:

$$f1 = \frac{2 * \text{precision} * \text{recall}}{\text{precision} + \text{recall}}$$

The F1-score ranges from 0 to 1, where a score of 1 indicates a perfect precision and recall, and a score of 0 indicates that either precision or recall is zero. The **F1-score is high when both precision and recall are high**, which means the model has a low false positive rate and a low false negative rate.

Classification (10)

Receiver Operating Characteristic curve

The **ROC curve** is a plot of the true positive rate (TPR) against the false positive rate (FPR) at different classification thresholds:

- The **true positive rate** is also known as the **recall** or the **sensitivity**.
- The **false positive rate** is the proportion of negative instances that are incorrectly classified as positive

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

In binary classification, the **threshold** is the value used to decide whether a predicted probability or score corresponds to a positive or negative classification.

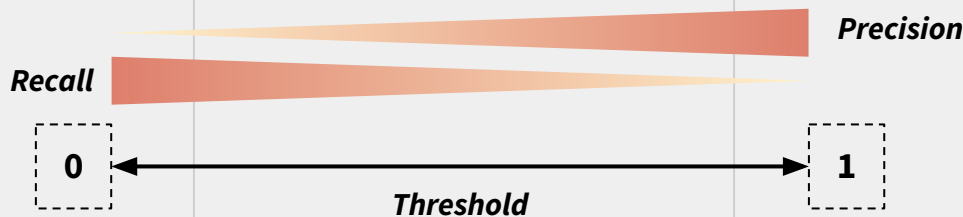
Classification (11)

Receiver Operating Characteristic curve

By default, that threshold is set to 0.5:

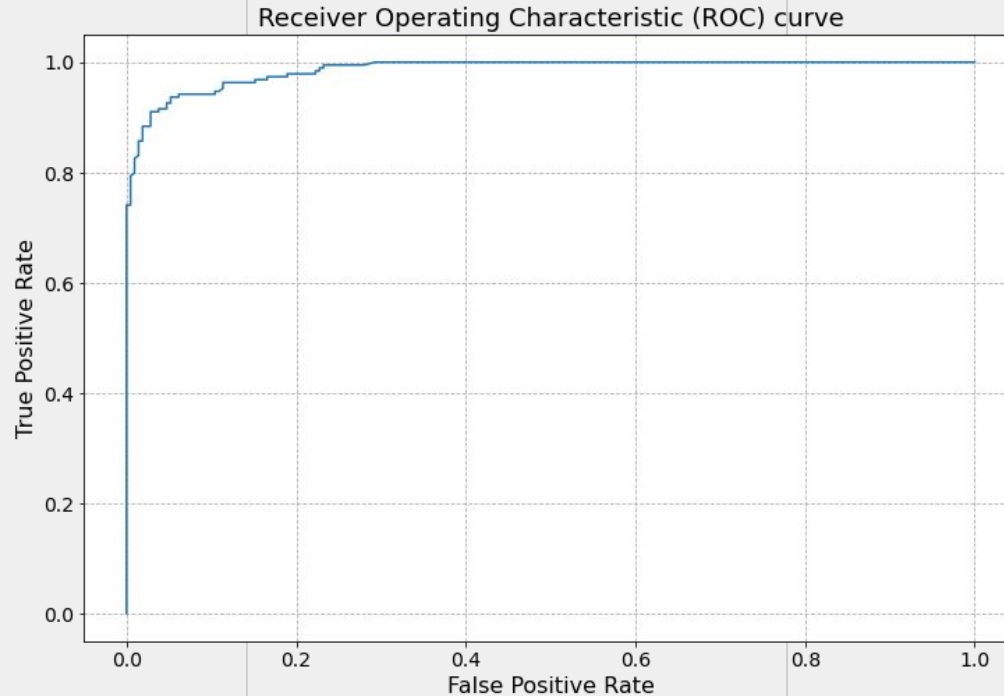
- A predicted probability **higher than 0.5 is a predicted positive**
- A predicted probability **lower than 0.5 is a predicted negative**

However, it may be interesting to choose another threshold. The choice of threshold depends on the problem and the goals of the classification task. A higher threshold leads to a higher precision but lower recall, while a lower threshold leads to a higher recall but lower precision.



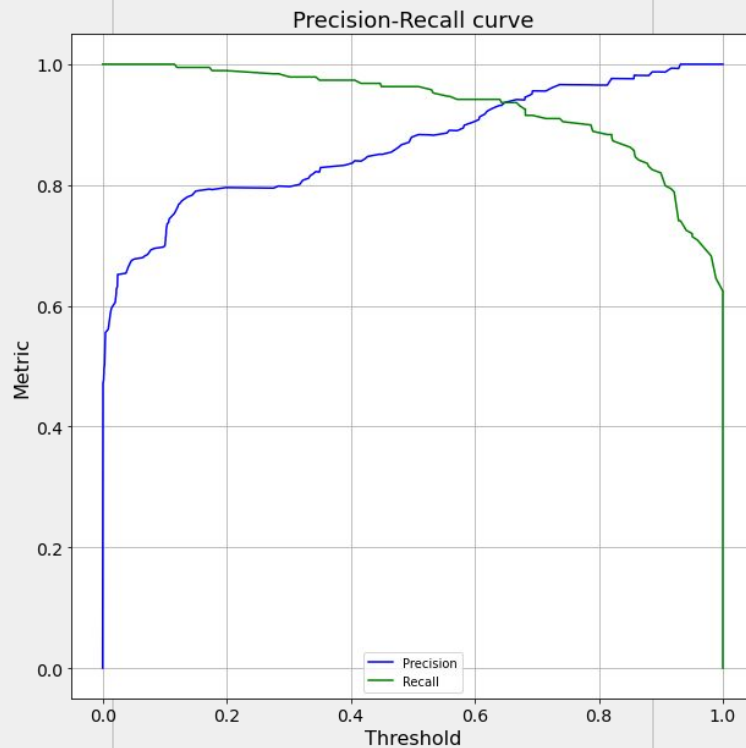
Classification (12)

Receiver Operating Characteristic curve



Classification (13)

Precision-Recall curve

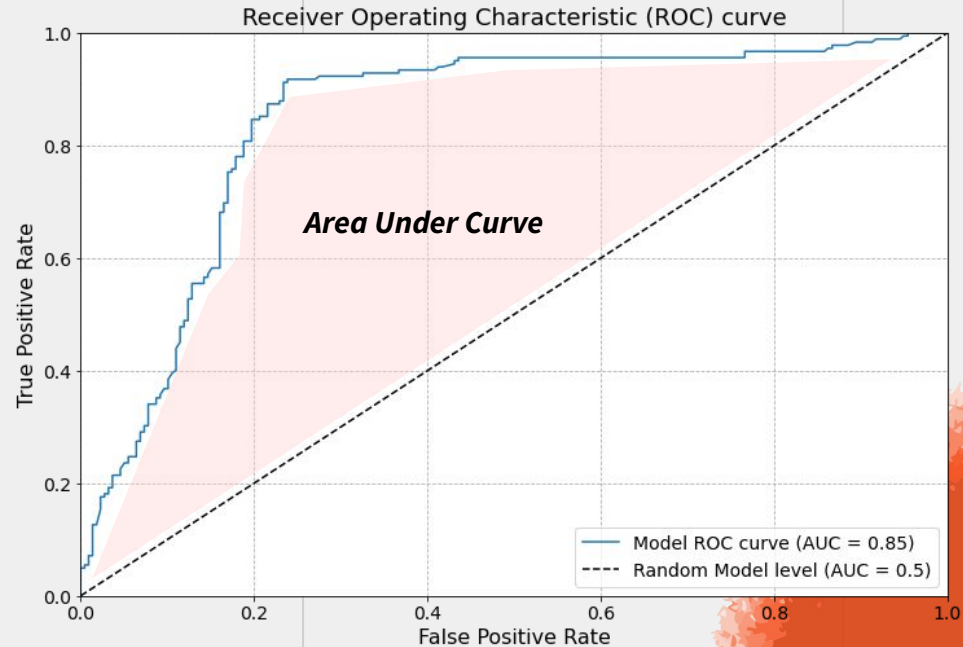


Classification (14)

Area Under the Curve

The AUC score is the **area under the ROC curve**, which ranges from 0 to 1. A score of 0.5 indicates that the model is randomly predicting, while a score of 1 indicates a perfect model.

The AUC score provides a way to compare different models based on their overall performance, regardless of the threshold value and is **useful in imbalanced datasets** because it is insensitive to changes in class distribution.



Classification (15)

Multi-class classification

Classification can be binary or multi-class i.e. there are more than 2 classes in the label. Accuracy, precision, recall, or even F1-score can be extended to multi-class classification problems in different ways. Here are some common approaches:

- **Macro-averaging** - Calculate the metric for each class separately and then average the results.
- **Weighted averaging** - Calculate the metric for each class separately and then average the results, weighted by the number of instances in each class.
- **Micro-averaging** - Aggregate the true positives, true negatives, false positives, and false negatives over all classes and then compute the metric.

Classification (16)

Multi-class classification

Macro-averaging should be used when all classes are equally important regardless of the class distribution. It is useful when the goal is to identify classes that the classifier is struggling with.

Micro-averaging should be used when the class distribution is imbalanced and the majority class should be emphasize.

Weighting-averaging should be used when the class distribution is imbalanced and the minority class should be emphasize.

Classification (17)

Multi-label classification

Multi-label classification is a type of classification problem where each instance can be associated with multiple labels or categories simultaneously. One way to extend the metrics to multi-label classification is to **treat each label as a separate binary classification problem and then average the results over all labels.**

This average can be macro (*calculate the metric for each label separately*) or micro (*aggregate the true positives, true negatives, false positives, and false negatives over all labels and then compute the metric*).

Ranking metrics such as the **Mean Average Precision at K** can also be used for multi-label classification.

Regression

Mean Squared Error

The most commonly used regression metric is the **Mean Squared Error** (MSE). It is the average of the squared differences between the predicted and actual values.

$$MSE = \frac{1}{n} \sum (y_{pred} - y_{true})^2$$

It is **useful when the outliers have a significant impact on the performance of the model**. In such cases, the squared errors can provide a more robust measure of the error than the absolute errors.

However, one limitation is that the **MSE penalizes large errors more heavily than small errors** because of the squared term.

Regression (2)

Root Mean Squared Error

The **Root Mean Squared Error** (RMSE) provides a measure of the error in the same units as the target variable and is **more interpretable than the MSE**.

$$RMSE = \sqrt{\frac{1}{n} \sum (y_{pred} - y_{true})^2}$$

However, it is still a bit sensitive to outliers and extreme values and may also penalize large errors more heavily than small errors. Additionally, like the MSE, it **does not provide any information about the direction of the error**, whether the model is overestimating or underestimating the actual values.

Regression (3)

Mean Absolute Error

The **Mean Absolute Error** (MAE) measures the average of the absolute differences between the predicted and actual values.

$$MAE = \frac{1}{n} \sum |y_{pred} - y_{true}|$$

It is **less sensitive to outliers and extreme values than MSE and RMSE** because it does not use squared errors. Additionally, MAE provides a **more robust measure of the error when the distribution of the errors is non-normal**.

However, it does not differentiate between overestimation and underestimation, which can be important in some applications.

Regression (4)

R-Squared

R-squared, also known as the coefficient of determination, measures the proportion of the variance in the target variable that is explained by the model.

$$R^2 = 1 - \frac{\sum (y_{pred} - y_{true})^2}{\sum (y_{pred} - y_{mean})^2}$$

A value of 1 indicates that the model perfectly fits the data, while a value of 0 indicates that the model does not explain any of the variance in the target variable.

However, it measures how well the model fits the training data, but **it does not necessarily reflect how well the model will perform on unseen data**. and that the relationship between the target variable and the features is linear. Additionally, **it can be misleading when used to compare models with different numbers of features**.

Regression (5)

Adjusted R-Squared

Adjusted R-squared is a modified version of the R-squared metric that accounts for the number of features in the model.

$$AR^2 = 1 - \frac{(1 - R^2)(n - 1)}{(n - k - 1)}$$

*n is the number of instances
k is the number of features*

It **penalizes the addition of irrelevant features to the model** and provides a more accurate measure of the goodness of fit than R-squared, especially when the number of features is large. Adjusted R-squared can also be used to determine the optimal number of features for a given model.

However, **it still does not necessarily reflect how well the model will perform on unseen data.**

Regression (6)

Other regression metrics

Other regression metrics can be useful such as:

- **Mean Absolute Percentage Error** (MAPE) - *it measures the average percentage difference between the predicted and actual values and can be useful in situations where the magnitude of the error is important.*
- **Mean Squared Log Error** (MSLE) - *it measures the squared difference between the logarithm of the predicted and actual values and can be useful when the label has a wide range of values.*

07

Overfitting & Underfitting

Underfitting

Underfitting is a phenomenon in machine learning where a **model fails to capture the complexity of the underlying data**.

As a result, the model performs poorly on the training data and unseen data such as the validation and the testing data.

It generally occurs for one or both of the following reasons:

- The model is too simple
- The model is not trained for long enough

Approaches to address underfitting involve to **increase the model capacity** by tuning its hyperparameters, to **increase the amount of data** or simply to **change the model** to a more complex one.

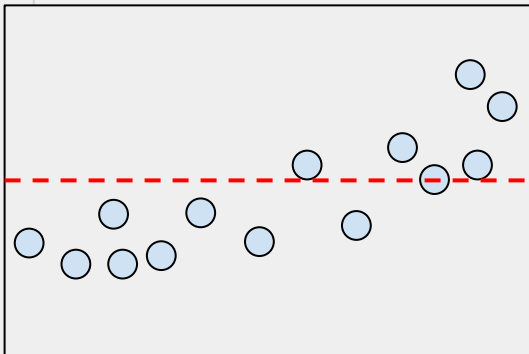
Overfitting

Overfitting is the reverse phenomenon of underfitting. It happens when a model becomes too complex and **starts to memorize the training data** instead of learning the underlying patterns i.e. it has learned the noise in the training data instead of the signal.

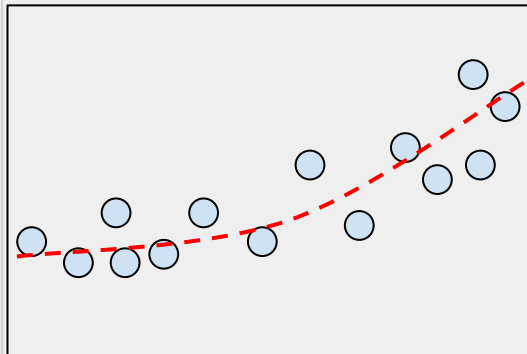
Generally, overfitting **leads to great performances on the training data but on poor generalization performance** i.e. the model does not perform well on the validation and the testing data.

Overfitting (2)

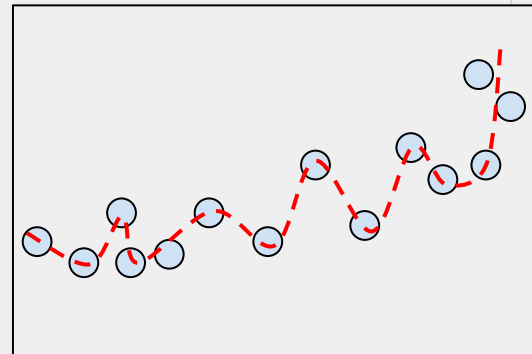
The model is **underfitting**.
It does not understand the
patterns in the data.



The model is well-fitted.
It understands the patterns in
the data.



The model is **overfitting**.
It understands the training data
too well and misses the
underlying patterns.



Overfitting (3)

Several strategies can be employed to address overfitting:

- **Reducing the model's complexity** by tuning its hyperparameters
- Using **regularization techniques** such as L1 or L2 regularization which penalize the model.
- **Reducing the training time**
- **Increasing the training data**

To ensure that the model performs well on both the training data and the unseen data, techniques such as the cross-validation method can be used.

Bias-Variance Tradeoff

The **bias-variance tradeoff** describes the relationship between a model's ability to fit the training data (bias) and its ability to generalize to unseen data (variance).

Models with **high bias are often too simple and fail to capture the underlying patterns** in the data, resulting in poor training performance. It leads to underfitting.

Models with **high variance are often too complex and fail to generalize well to unseen data**, resulting in poor testing performance. It leads to overfitting.

The goal is to find a model with a perfect balance i.e. a **model that has low bias and low variance**.

Learning Curves

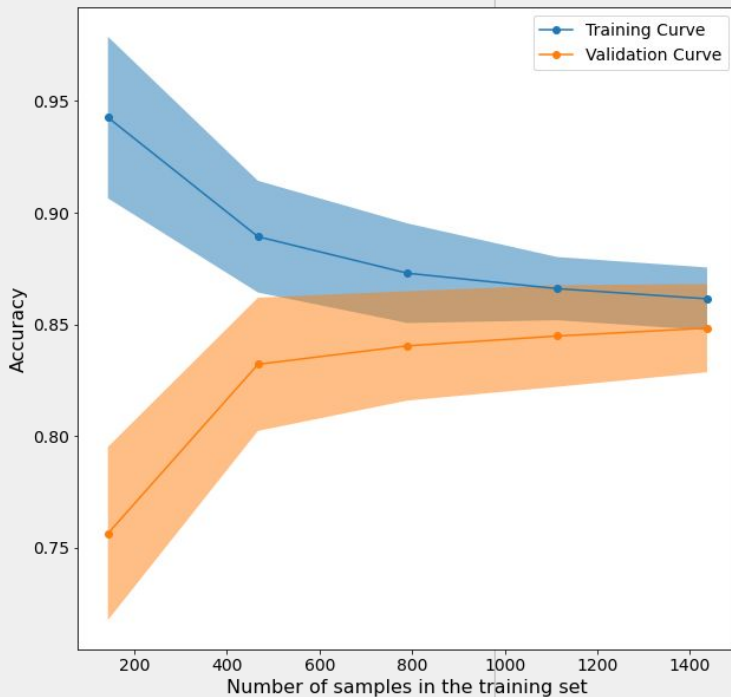
Learning curves can help to detect overfitting and underfitting. They plot the performance of a model on both the training data and the validation relative to the number of iteration (*or training examples used*).

By analyzing the shape and trajectory of the learning curves, we can determine whether the model is underfitting, overfitting, or is well-fitted to the data:

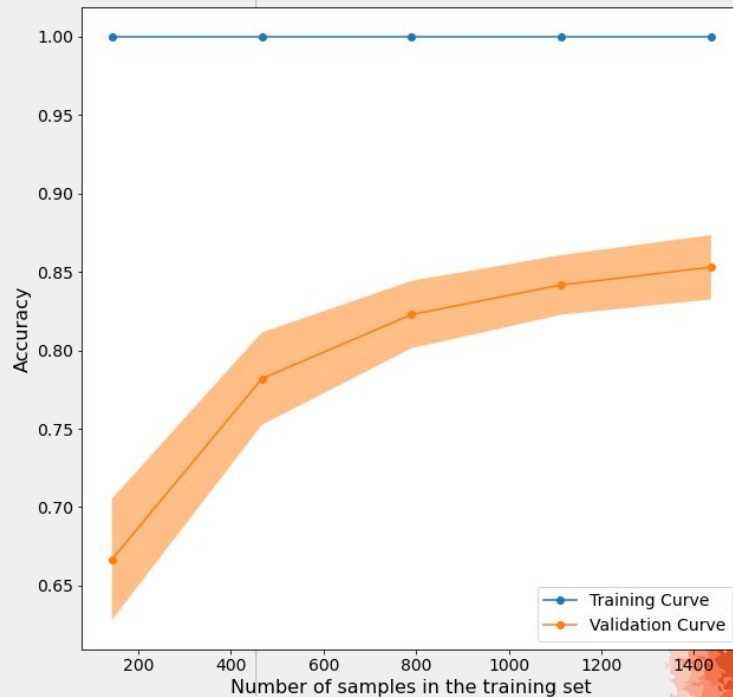
- In the case of **underfitting**, we would expect to see **high error rates for both the training and validation data**
- In the case of **overfitting**, we would expect to see **low error rates for the training data, but high error rates for the validation data**

Learning Curves (2)

Good fit - the training and the validation curve are close at the end of the training.



Overfitting - the training curve is much higher than the validation curve.



Validation

Validation and especially cross-validation are techniques that can be used to **prevent overfitting**.

Validation allows to estimate how well the model performs on unseen data. Cross-validation helps to prevent overfitting by providing a **more accurate estimate of the model's performance than a single validation set**.