Introduction to Data Preprocessing

Introduction to Data Preprocessing

X

Heuristical motivations for data preprocessing: presence of data in daily context, examples of usage of data. Reasons to preprocess data. Definition of Data Preprocessing. Examples of improper data preprocessing.

X

1. Data is everywhere

Data is everywhere, as it is always present in our daily context. To name a few examples, we have mobile applications, sensors, loyalty programs, credit cards, online shops. With this data, we can do a lot of stuff! For example, we can create *reccomendation systems* or even *guess pregnancies* by looking at consumer patterns. To name other specific examples, we have:

- **Tesla's autonomous driving**, as its technology is deeply integrated with AI, which is powered by deep neural networks that anallyze data from the cameras and sensors.
- **Moneyball** is a term associated with the strategy of using *data science* to build *competitive sport teams*.
- **Banks** approve loans by using a series of criterias, which are insights gained from *data of previous customers*. In this way, they can have a pattern of characteristics which can determine good or bad clients.

2. Reasons for Data Preprocessing

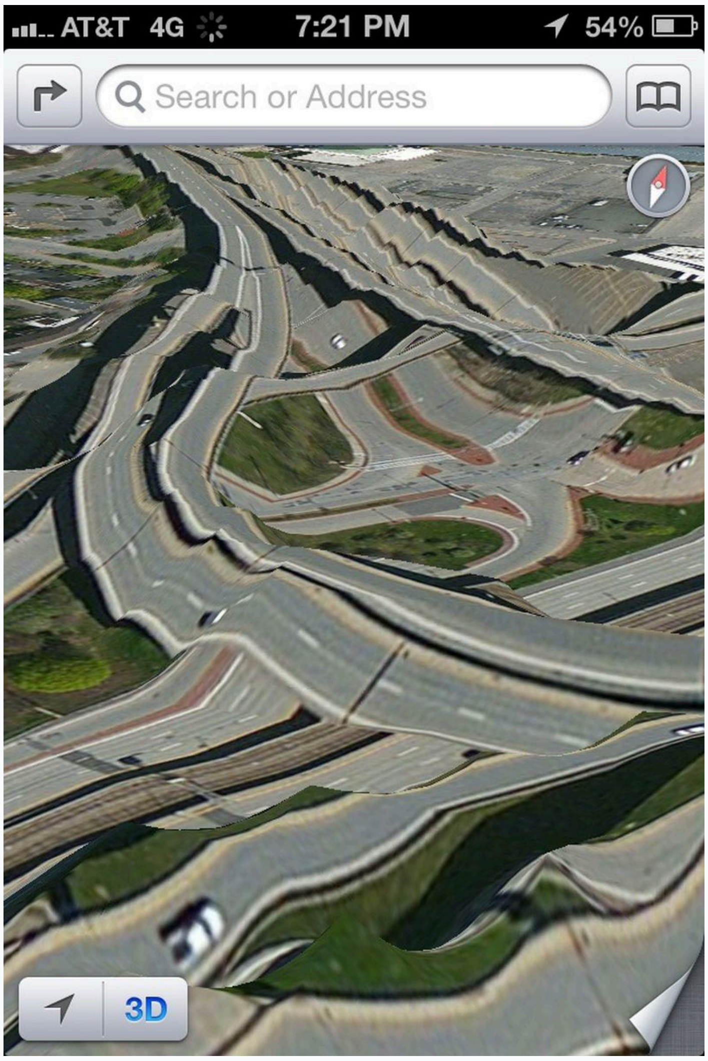
So, we have data. To briefly define *data preprocessing*, it's the act of treating data so it is provided in the amount, structure and format that suit each task perfectly.

Q. *Why preprocess data?*
A. *For a lot of reasons, such as to filter data and improve decisions. However, the main reason is that we have a lot of messy datasets, which lead to low-quality data analytics data performance! Let us see a few results of improper data preprocessing.*

EXAMPLE. (*Windrush immigrants*)
The *1971 immigration Act* gave *Commonwealth citizens* living in the UK indefinite leave to remain - so the permanent right to live and work in the UK. Then, in 2012 the UK home office implemented a *"hostile environment"* policy aimed at cracking down on illegal immigrations. This policy required individuals to show documentation proving their right to live and work in the UK. However, as the UK government had not maintained proper records of *Windrush immigrants*, many of these immigrants have been deported. So, this is a result of *improper data preprocessing*, as:

1. There has been *missing historical data*
2. They *failed to account for exceptions* (1971 Immigration Act)
3. *Badly mantained dataset*

EXAMPLE. (*Apple maps*)
In 2012 Apple has launched its own mapping service, which was intended to compete with Google maps. However, users quickly noticed that Apple Maps was riddled with inaccuracies and missing data. Graphical illustration below:

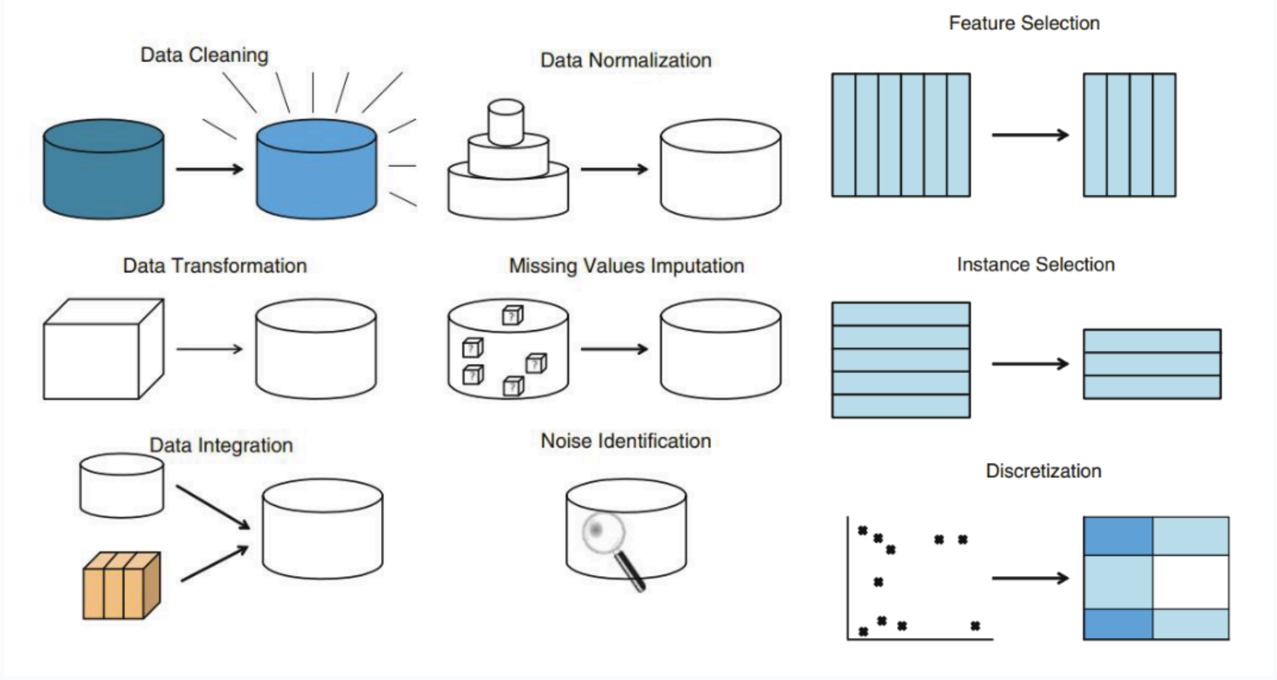


This was a result of *bad data preprocessing*, as Apple failed to cleanse and validate mapping data properly. In fact, there were a lot of errors and inaccuracies: for example, Apple Maps directed people to rivers instead of airports.

3. Focus of Data Preprocessing

Data preprocessing will focus on *data quality* and *sustainability*, as these are one of the most important pre-requisites for any type successful data analysis project. In particular, we will learn:

- *Data integration*: integrate data from multiple sources into one common format
- *Data cleaning*: resolve missing values, errors, outliers and other types of inconsistencies in the data (the most important part; about the 80% of the work!)
- *Data transformation and reduction*: normalize data, reduce dimensions of the dataset, ...



X

Methodology in Data Science

Methodology in Data Science

X

Definition of methodology in the Data Science context. Common methodologies: CRISP-DM, SEMMA and OSEMN.

X

0. Voci correlate

- Introduction to Data Preprocessing
- Introduction to Data Mining
- Models in Machine Learning

1. Definition of Methodology

DEFINITION. (*Methodology*)
A *methodology*, in the context of *Data Science*, is a *framework for recording experience*, allowing projects to be replicated. In particular it describes a series of processes which have the goal of turning *data* into *information*.

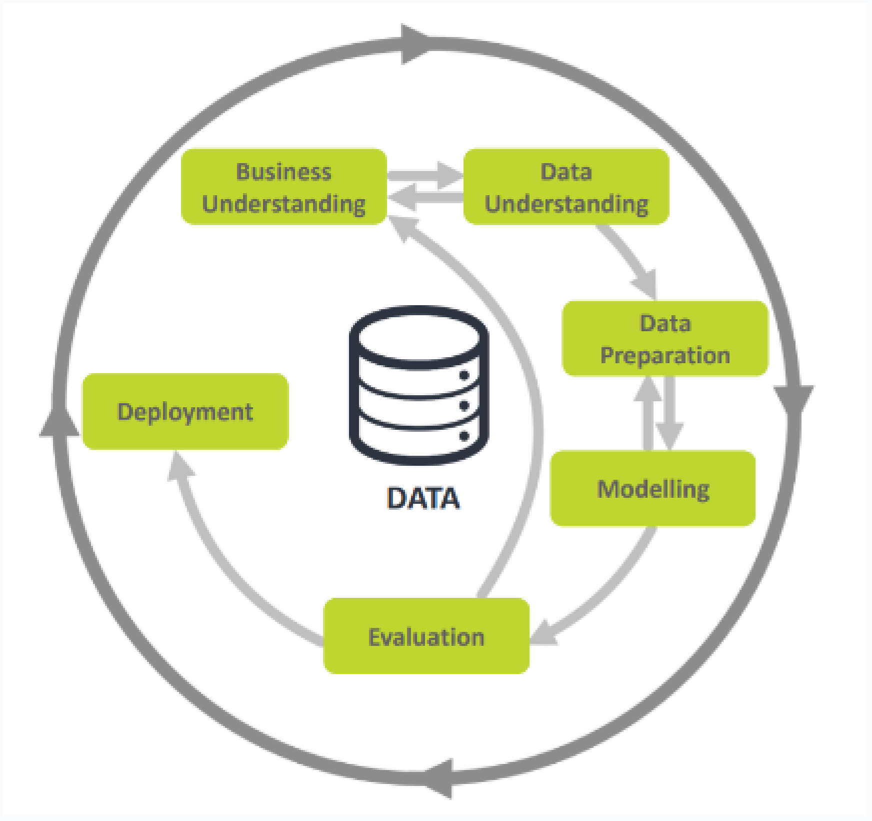
Today we have a lot of methodologies, but the commonly used are the following:

- **CRISP-DM** (1996)
- **SEMMA** (≈ 2000)
- **OSEMN** (2010)

X

2. CRISP-DM

CRISP-DM (*Cross-Industry Standard Process for Data Mining*). Designed in 1996, is considered to be the most *"complete"* methodology. In fact, this comprises *business understanding*.



Moreover, this is not a *linear process*, as in some steps we have *loops*. Let us look at the main steps.

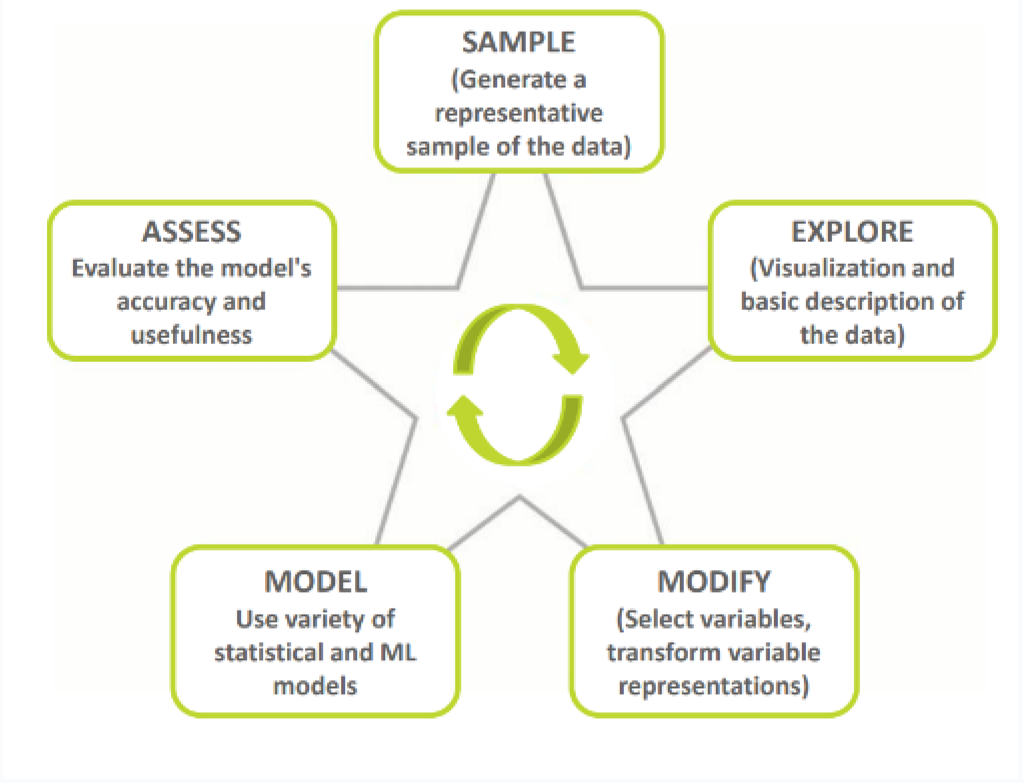
1. **BUSINESS UNDERSTANDING.** Define objectives and requirements of the project.
2. **DATA UNDERSTANDING AND PREPARATION.** Mainly, we have to *prepare* the *data*; in particular first we have to *look* at the data to identify problems, and then *prepare* it to our needs.
3. **MODELLING.** Choose models, calibrate parameters and optimize the chosen model.
4. **EVALUATION.** Evaluate the model according to the criterions defined in the *business understanding* (1) step.
5. **DEPLOYMENT.** Deploy the model with a repeatable implementation; might not be necessary, it depends on the requirements.

In particular, the course *"Data Preprocessing"* will only look at the first two steps.

3. SEMMA

SEMMA (*Sample, Explore, Modify, Model and Assess*). Designed around the year 2000 by the SAS company analytics. It is a more *"simple"* version of CRISP-DM, where business undersanding is no longer a requirmeent. It is a cyclical process, with the following steps:

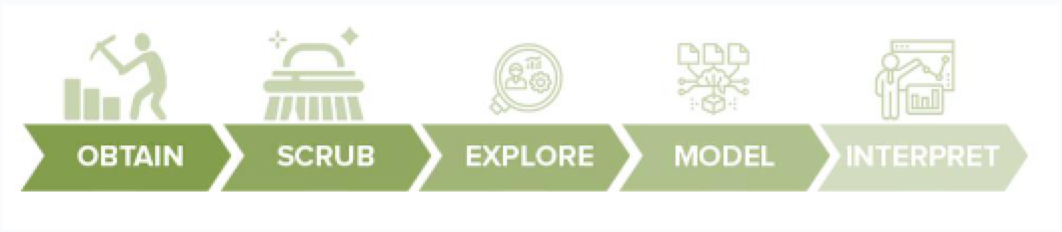
- *Generate a sample of the data*
- *Explore and visualize data*
- *Modify the data*
- *Make a model*
- *Assess the model*


SAMPLE
(Generate a representative sample of the data)
EXPLORE
(Visualization and basic description of the data)
MODIFY
(Select variables, transform variable representations)
MODEL
Use variety of statistical and ML models
ASSESS
Evaluate the model's accuracy and usefulness

In the context of *Data Preprocessing*, we look at only the first three steps: Sample, Explore and Modify.

4. OSEMN

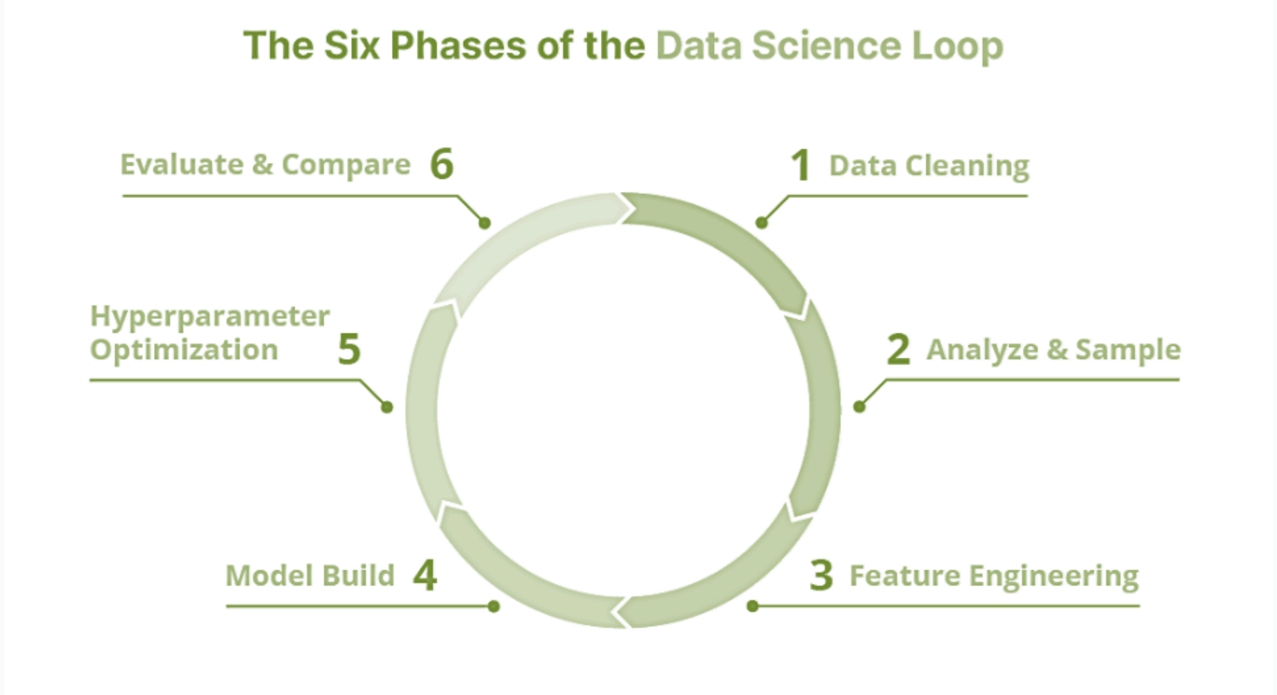
OSEMN (*Obtain, Scrub, Explore, Model, Interpret*). The most recent framework, is mostly the same as the other models. This particularly enhancizes on the *web sources*, and this framework is better documented.



5. Generalized Data Science Process

We can generalize the aformentioned frameworks with the following:

1. **DATA CLEANING.** Make sure the data is *"clean"* enough for the *algorithms*; so *no missing values*, *is correct*, has *acceptable values*.
2. **ANALYZE AND SAMPLE DATA.** Make sure that our quantity of data makes sense, explore data to see any *patterns* or how it is *distributed*.
3. **IMPROVE DATA.** If needed, apply transformations, create new features, or select some features and discard others.
4. **APPLY MODELS.** Choose and build the appropriare modeling technique
5. **IMPROVE MODEL.** Do hypeparameter tuning
6. **EVALUATE AND COMPARE.** Evaluate models by comparing their predictions with real data; pick the model based on performance and other metrics, such as *scalability, interpretability, etc...*



X

Introduction to Data Mining

Introduction to Data Mining

X

Introduction to Data Mining: definition, motivations and techniques.

X

0. Voci correlate

- Introduction to Machine Learning

1. Definition of Data Mining

Data Mining is the *essential process* where the methods are used to extract *valid data patterns*. This includes the choice of the most suitable task, algorithm, fine tuning and deployment. We are essentially using the tools from *Machine learning* to gain insight from data (Introduction to Machine Learning). To sumarize the definition, we can quote the following:
"Data mining is a business process for exploring large amounts of data to discover meaningful patterns and rules."

MOTIVATIONS. The tools for data mining gave beeb developed decades ago, already; the reason for which data mining's popularity has been rising since the 1990s, is the fact that *we have more data* and *it is everywhere*. Moreover, it is even *critical* as the the interest in customer relationship management is rising: in fact, *every business is a service business*.

METHODOLOGY. For data mining, we will use the *same methodologies* in any *Data Science project* (Methodology in Data Science).

2. Tools of Data Mining

Mainly, in *data mining* we have two techniques: *predictive* and *descriptive modelling*. In terms of *machine learning*, they respecively correspond to *linear* and *logistic regression* (Linear Regression, Logistic Regression).

However, the role of *data preprocessing* is to make sure that the *data* is effectively prepared to be used in *data mining*.

X

Assumptions for Data Mining Methods

Assumptions for Data Mining Methods

X

Assumptions in data mining to check for during the phase of data preprocessing.

X

0. Voci correlate

- Introduction to Data Mining

1. Introduction

We can do a lot of interesting, useful and insightful things with *data mining*. However, before applying any of the *data mining techniques*, we first have to assure that our data is *compliant* with the *requirments* of each *method* used.

X

2. Factors for Assumptions

Generally, the assumptions we're talking about are as follows.

- **Variable Type**: Quantitative or qualitative data
- **Missing Values**: Whether some values are missing or not...
- **Outliers**: In layman's words, outliers are *"improper"* or *"extraordinary"* values. For example, in measuring the worker's monthly wage, Bill Gates would be an outlier.
- **Homo/hetero-cedasticity**: Whether we have similar or different variance between variables.
- **Multicollinearity**: Whether we have a correlation between the explanatory variables. Measured in the interval $[-1, 1]$, and is considered to be high (bad enough!) if it's in the interval $[-1, -.7] \cup [.7, 1]$.
- **Variables Distribution**: We have to consider *how* the variables are distributed; for example if they follow a normal, Poisson, exponential and similar distributions.
- **Duplicates**: Self-explanatory term.

- **Inconsistencies**: Whether certain pieces of data make sense or not.
- **Standardization**: In some cases, standardizing numerical variables are useful. The typical formula to do that, with μ, σ (mean and standard deviation) given is the following.

$$x \mapsto \frac{x-\mu}{\sigma}$$

3. Methods' Assumptions

The following table provides a quick overview of assumptions needed.

REQUIREMENT	Decision Tree	Regression	KNN	Neural Network	Clustering
Quantitative Variables	NO	DESIRABLE	YES	DESIRABLE	DESIRABLE
No Missing Values	NO	YES	YES	YES	YES
No Outliers	NO	YES	YES	YES	YES
Homocedascity	NO	YES	ANY	NO	ANY
Low Multicollinearity	NO	YES	ANY	YES	YES
Normal Distribution	NO	YES	NO	NO	ANY
Standardized Variables	NO	NO	YES	NO	YES

REMARK. Note that *decision tree* is the *only* method that doesn't have any requirements!

X

4. How to detect and resolve problems

If we had any problems in meeting these requirement, we have to *transform* or *modify* the data accordingly. We have a lot of tools for that, such as *Excel*, *SAS Miner*, *Python* and so on...

DUPLICATES.

- To detect duplicates in Excel, it is just necessary to use the pre-built feature in the *"Data"* section.
- Usually, duplicates are deleted.

OUTLIERS.

- To detect *outliers* in Excel, it is necessary to *visualize data*. Usually, *boxplots* or *histograms* work well. In particular, *boxplot* work extraordinarily with normally distributed data.
- If we have any outliers, we usually *delete them* (or do *analysis* separately)

MISSING VALUES.

- To detect missing values, it is simply sufficient to *count* the terms (Example: Excel has the prebuilt function -, or Pandas has the prebuilt function `df.info()`).
- In case of *missing values*, we have two approaches:
  - In certain cases, *deleting* is simply fine. In particular, if the rows are few. However, this reduces data, which can affect our model performance.

- Predict the missing value with certain algorithms, such as *KNN*. In excel, we have the following procedure:
    - Create a new sheet -> Click on *"Import Excel sheet"* -> Click on *"Transform data (power query)"* -> Find variables that are "important" for predicting -> Sort the table by these variables -> Go to *"Transform"* tab and click *"Fill* -> Close and load, then copy and paste new data

X

Data Integration

Data Integration

X

Data Integration for Data Science projects: theoretical background, integration with Pandas and with PROC SQL.

X

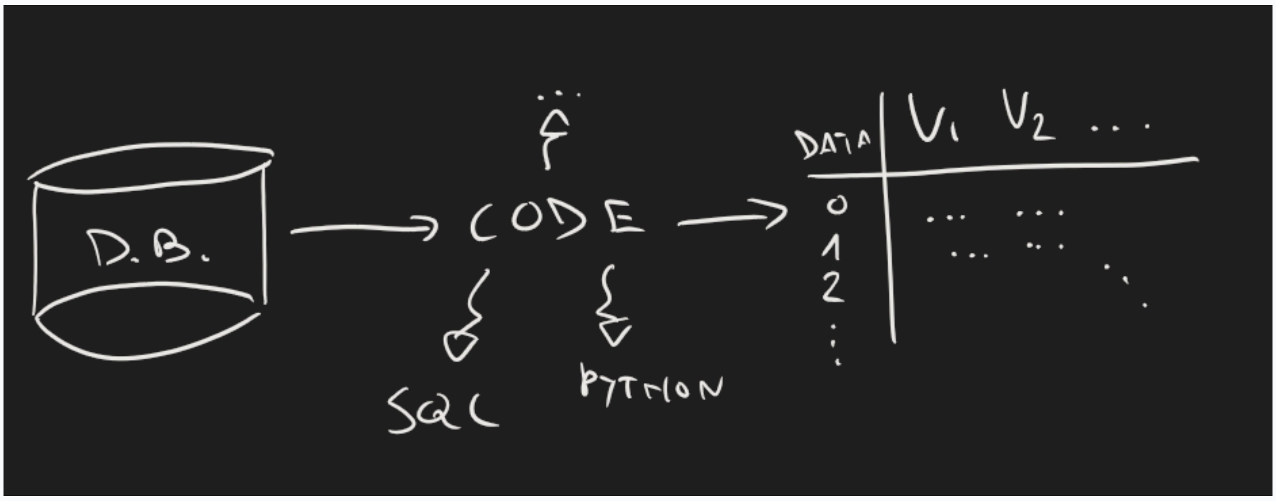
0. Voci correlate

- Data Science Process
- Methodology in Data Science

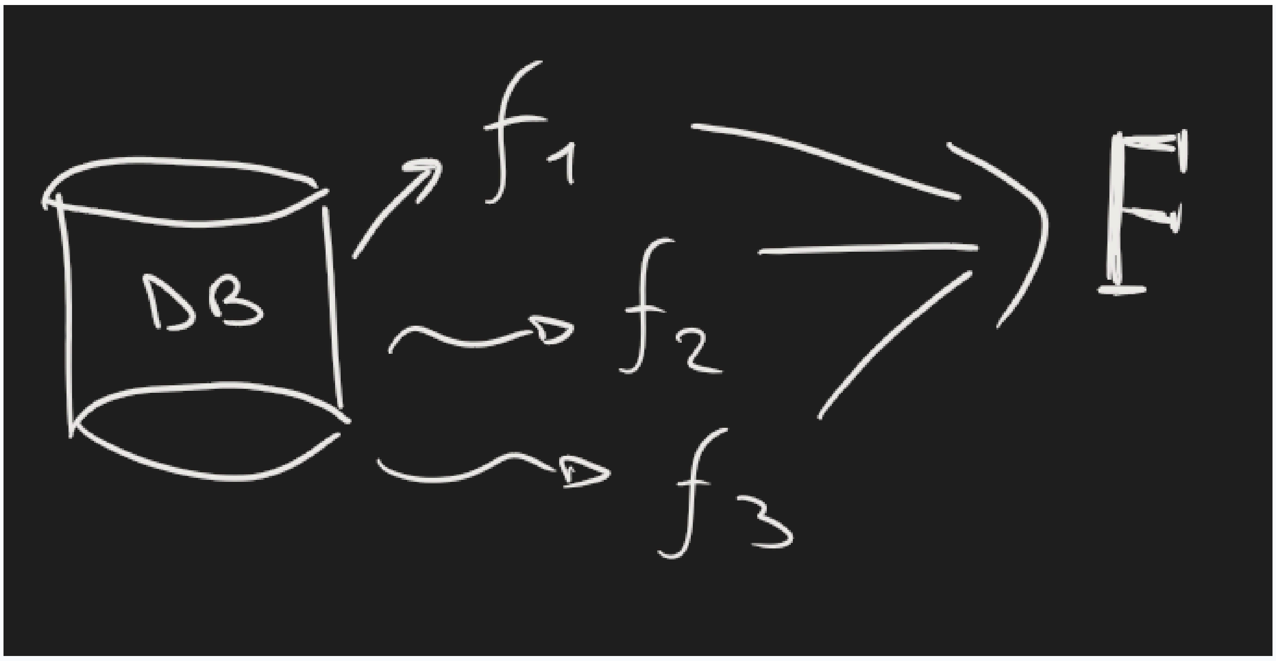
1. Introduction and Theoretical Background

Introduction

Before we start our *data science project*, we have first learn how to *import (export)* from (to) external data sources. Graphically, we have this process:



Moreover, we also have to deal with *multiple data formats*, such as Excel spreadsheets, `.csv` files or `.json` data structures. Therefore, we might also have to *integrate* multiple data formats into one



Usually, the data sources are typically the following types:

- Physical (HDDs, servers, ...)
- Databases (SQL, ...)
- Network (APIs, Web Scraping, ...)
- *Data warehouses*: places where structured data are stored in a common format, designed to enable business intelligence activities
    - Usually they have *summary data* for business decision making
    - We have *metadata* to explain the data itself (formats, descriptions, dictionaries...)
    - The data itself

So, the process of *merging data from multiple sources* is called *data integration*.

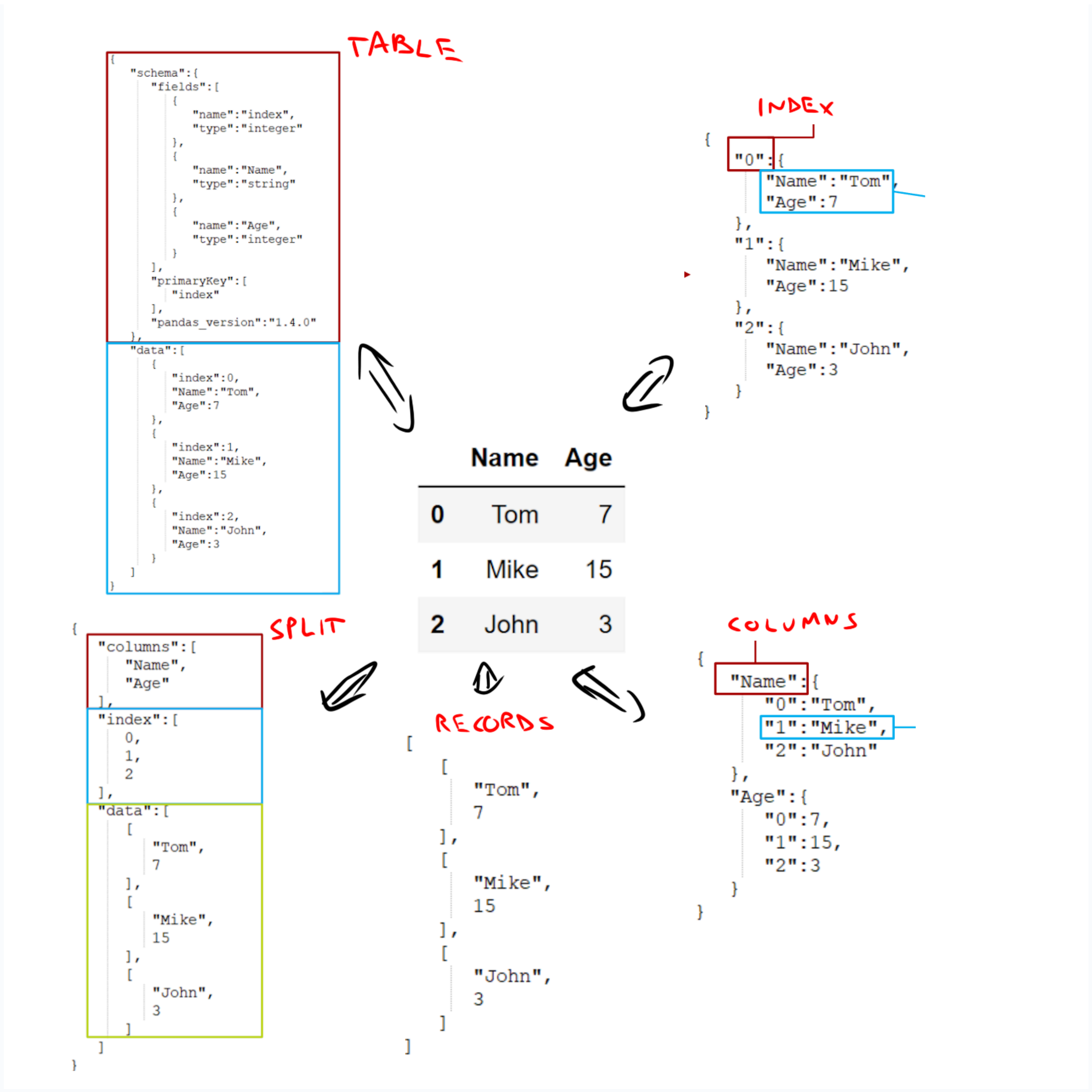
Data Formats

Before we see how *data importing* works, first we define the data formats.

DEFINITION. A plain-text file is in `.csv` (*Comma-Separated Values) format if it has the following characteristics:

1. Each line represents a datapoint
2. In each line the values are separated by a single-character delimiter (such as the comma `,`)
3. Optionally, have a header in the first line

DEFINITION. *Json* is a non-tabular data format, where it is organized into a *collection of objects*. There are lot of *formal structures* for JSON files that can be automatically parsed by Pandas, which are as follows: Split, records, index, columns, values.



```
{
  "schema":{
    "fields":[
      {
        "name":"index",
        "type":"integer"
      },
      {
        "name":"Name",
        "type":"string"
      },
      {
        "name":"Age",
        "type":"integer"
      }
    ],
    "primaryKey":[
      "index"
    ],
    "pandas_version":"1.4.0"
  },
  "data":[
    {
      "index":0,
      "Name":"Tom",
      "Age":7
    },
    {
      "index":1,
      "Name":"Mike",
      "Age":15
    },
    {
      "index":2,
      "Name":"John",
      "Age":3
    }
  ]
}
```

```
{
  "0":{
    "Name":"Tom",
    "Age":7
  },
  "1":{
    "Name":"Mike",
    "Age":15
  },
  "2":{
    "Name":"John",
    "Age":3
  }
}
```

	Name	Age
0	Tom	7
1	Mike	15
2	John	3

```
{
  "columns":[
    "Name",
    "Age"
  ],
  "index":[
    0,
    1,
    2
  ],
  "data":[
    [
      "Tom",
      7
    ],
    [
      "Mike",
      15
    ],
    [
      "John",
      3
    ]
  ]
}
```

```
[
  [
    "Tom",
    7
  ],
  [
    "Mike",
    15
  ],
  [
    "John",
    3
  ]
]
```

```
{
  "Name":{
    "0":"Tom",
    "1":"Mike",
    "2":"John"
  },
  "Age":{
    "0":7,
    "1":15,
    "2":3
  }
}
```

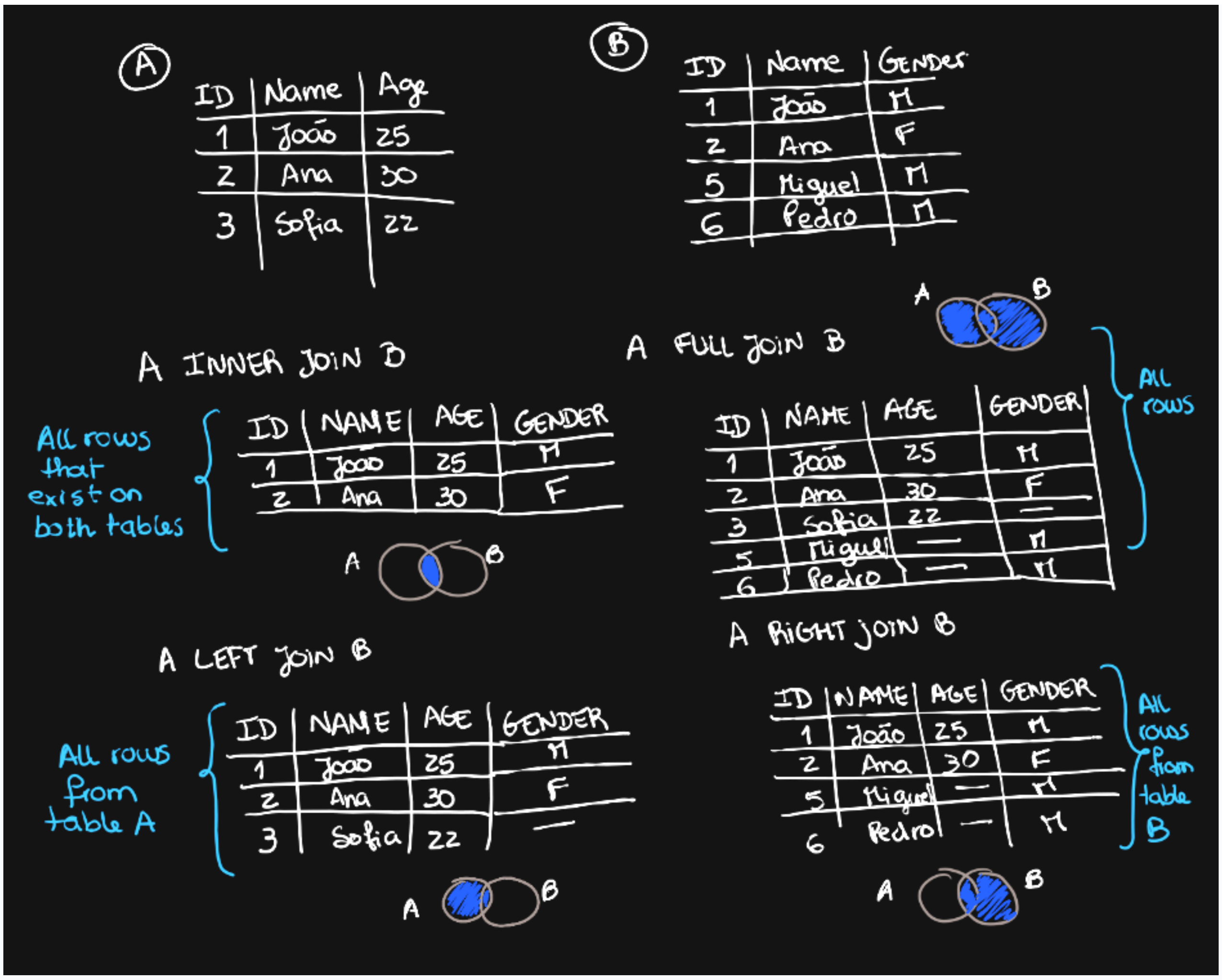
Theoretical Definitions

As introduced previously, we also should be able to combine multiple DataFrames into one only. We have multiple ways to do it; mainly, we can either combine them by joining the *columns* (horizontal) or by joining the *rows* (vertical).

In particular, if we want to *join columns*, we have more ways to do that. In particular, we have to determine which *rows* remain or not, or if we have common columns at all.

Regarding the first point, we have the *"how"* option:

- *Left*: We keep all of the rows of data present on the left side of the operator: $A \overset{\rightarrow}{+} B$
- *Right*: Analogous to *left*, but keep the rows on the right side: $A \overset{\leftarrow}{+} B$
- *Outer*: Keep every row, $A \overset{\cup}{+} B$
- *Inner*: Keep only rows which has no missing values on any variable of both datasets: $A \overset{\cap}{+} B$
- *Cross*: Make a cartesian product: $A \times B$



ID	Name	Age
1	João	25
2	Ana	30
3	Sofia	22

ID	Name	Gender
1	João	M
2	Ana	F
5	Miguel	M
6	Pedro	M

ID	NAME	AGE	GENDER
1	Joao	25	M
2	Ana	30	F

ID	NAME	AGE	GENDER
1	João	25	M
2	Ana	30	F
3	Sofia	22	—
5	Miguel	—	M
6	Pedro	—	M

ID	NAME	AGE	GENDER
1	Joao	25	M
2	Ana	30	F
3	Sofia	22	—

ID	NAME	AGE	GENDER
1	João	25	M
2	Ana	30	F
5	Miguel	—	M
6	Pedro	—	M

X

2. Data Integration with Pandas

Data Import and Export

The syntax to import data as DataFrames or do the inverse is very intuitive. We have, assuming `pd` as alias and `df` any DataFrame, the following:

DATA IMPORT

- `pd.read_csv(fpath, sep)`
- `pd.read_excel(fpath, sheetname)`
- `pd.read_json(fpath, orient)`

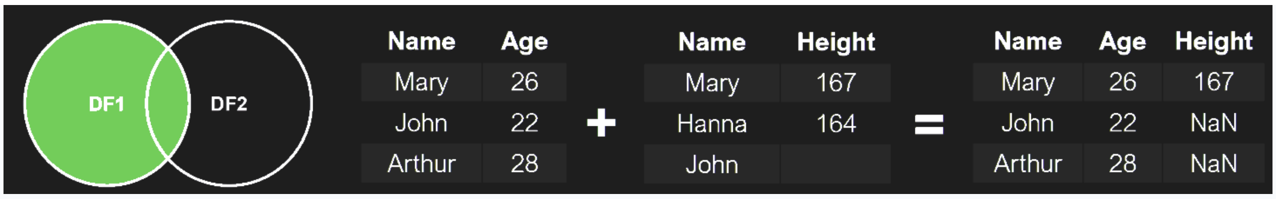
  DATA EXPORT
- `df.to_csv(fpath, sep)`
- `df.to_excel(fpath, sheetname)`
- `df.to_json(fpath, orient)`

Data Integration

Now let us see the detailed operators present in *Pandas*.

MERGE. If we want to combine data *horizontally* with a *common column*, we can use the `df1.merge(df2, on, how)` method. It selects all the rows by the *"direction"*, and then combines them with a common key: $A \underset{X}{\uplus} B$

EXAMPLE. (*Left merge on name*)



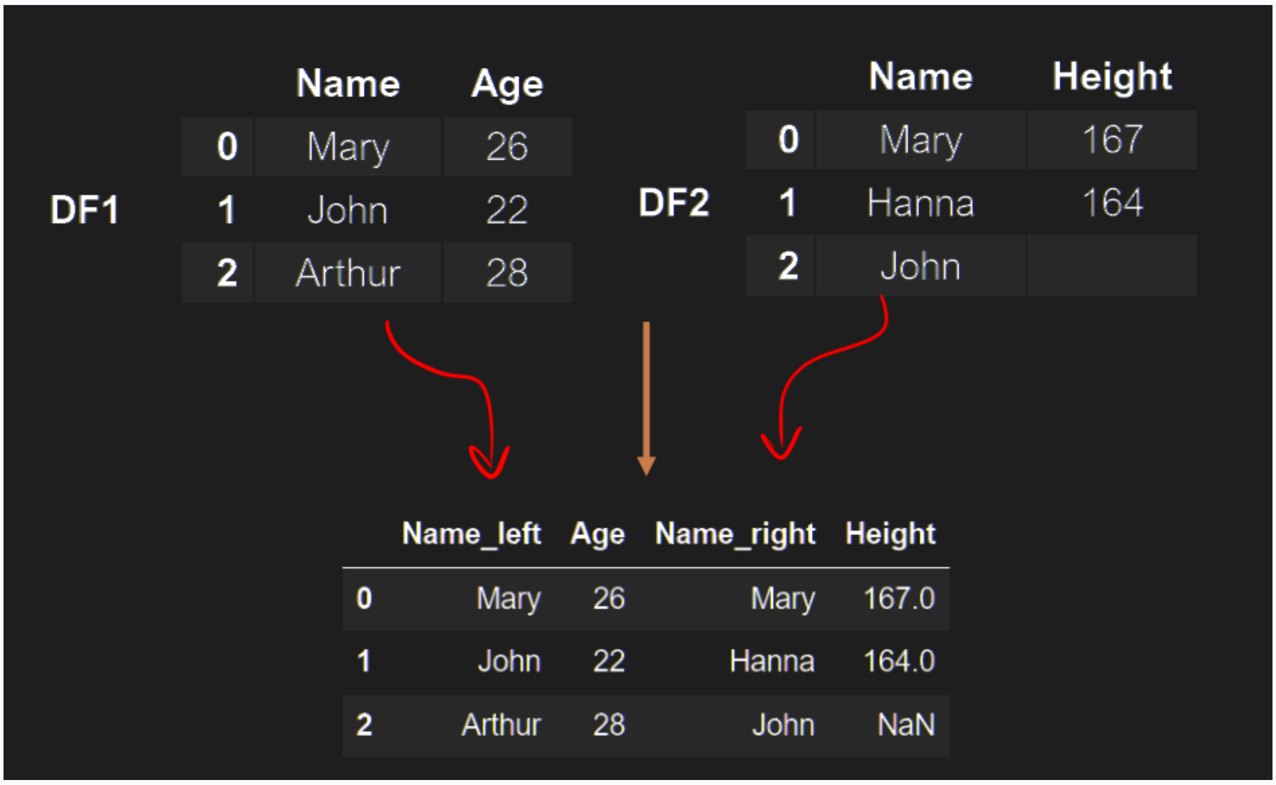
Name	Age
Mary	26
John	22
Arthur	28

Name	Height
Mary	167
Hanna	164
John	

Name	Age	Height
Mary	26	167
John	22	NaN
Arthur	28	NaN

JOIN. Same as the `merge` option, but basing on the *index positions*. Syntax: `df1.join(df2, lsuffix, rsuffix)`

REMARK. If we use `join` with two DataFrames s.t. they have columns in common, we have to at least specify either a *"right suffix"* or a *"left suffix"* for the new columns.



	Name	Age
0	Mary	26
1	John	22
2	Arthur	28

	Name	Height
0	Mary	167
1	Hanna	164
2	John	

	Name_left	Age	Name_right	Height
0	Mary	26	Mary	167.0
1	John	22	Hanna	164.0
2	Arthur	28	John	NaN

CONCATENATE. If we want to generalize the previous methods, we can use the `pd.concat([df1, ..., dfn], axis, ignore_index)`, where:

- `axis` allows us to specify if we want to merge rows or columns
- `ignore_index` allows us to perform joins ignoring the indexes, creating a table with a new index.

APPEND. Deprecated in newer version of pandas, omitted.

Advanced Options for Data Integration

Moreover, we can have advanced options for data combinations. In particular, if we want to fill missing values.

- `merge_ordered` allows us to merge and sort the entries on the key values. Optionally, it can fill or interpolate missing values.
- `merge_asof` allows us to match entries on the *"closest"* key

X

3. Data Integration with PROC SQL or SAS

Alternatively, we can go for proprietary software with *SAS* and their implementation of relational databases with *PROC SQL*. In particular, the former one allows us to:

- Create reports, descriptive statistics, tables, views and indexes
- Query tables
- Combine data
- Update values

1. **Concatenate by rows**

```
DATA S;
        SET x y;
RUN;
```

with PROC SQL

```SQL
PROC SQL;
CREATE TABLE S as
        (SELECT * FROM x)
OUTER UNION CORR
        (SELECT * FROM y)
RUN;
```

2. **Merge by columns**

```SQL
PROC SORT DATA=x OUT=x; BY <key>; RUN;
PROC SORT DATA=y OUT=y; BY <key>; RUN;

DATA S <options>;
        MERGE x y <options>;
        BY <key>
        <...>
RUN;
```

Note that with *PROC SQL* there's no need to sort original datasets.

```SQL
PROC SQL;
CREATE TABLE S as
        SELECT x.* y.*
        FROM x <LEFT/RIGHT/OUTER/INNER/SELF> JOIN y ON (x.key1=y.key2);
WHERE <conditions>;
RUN;
```

3. **Advanced Options**
   With PROC SQL we can also group our query and filter these groups:

```SQL
PROC SQL;
SELECT x1, ..., xn
FROM X
        WHERE <conditionX>
GROUP BY xi
        HAVING <conditionxj>;
RUN;
```

Also, the `WHERE`, `SELECT`, `FROM` and `HAVING` clauses can be used with the predicates `EXISTS`, `IN`, `ANY`, `ALL`.

Data Visualization Techniques for EDA

Data Visualization Techniques for EDA

Data visualization techniques. Introduction: motivation, Tufte's rules of thumbs. Main types of plots: bar charts, histograms, pie charts, box plots, line plots and scatter plots.

---

0. Voci correlate

- Methodology in Data Science

1. Introduction and Principles for Data Visualization

Q. We have imported and aggregated pieces of data into one. What now?
A. Before doing anything to the data itself, such as cleaning, we have to first explore it; so we can gain insights and detect any problems with it. In particular, we're going to use visualization techniques for this.

So one of the most important parts in the *Exploratory Data Analysis* phase in a *Data Science* project is *data visualization*. As mentioned before, it allows us to:

- Explore data
- Analyze data and check for certain hypothesis
- Present data to the public
  In order to achieve the goals efficiently, the professor Tufte has designed the following *"rules of thumb"*:

1. **DATA INTEGRITY** - Avoid presenting data in an intentional misleading manner, e.g. scaling or cutting axes in a very specific way.

2. **MAXIMIZE DATA-INK RATIO** - Be as essential as possible, and avoid *"useless decorations"* for your visualization (i.e. using 3D graphs for a simple 2D scatterplot).
3. **AVOID CHART JUNK** - Be as minimalist as possible and make the visualizations as readable as possible.

X

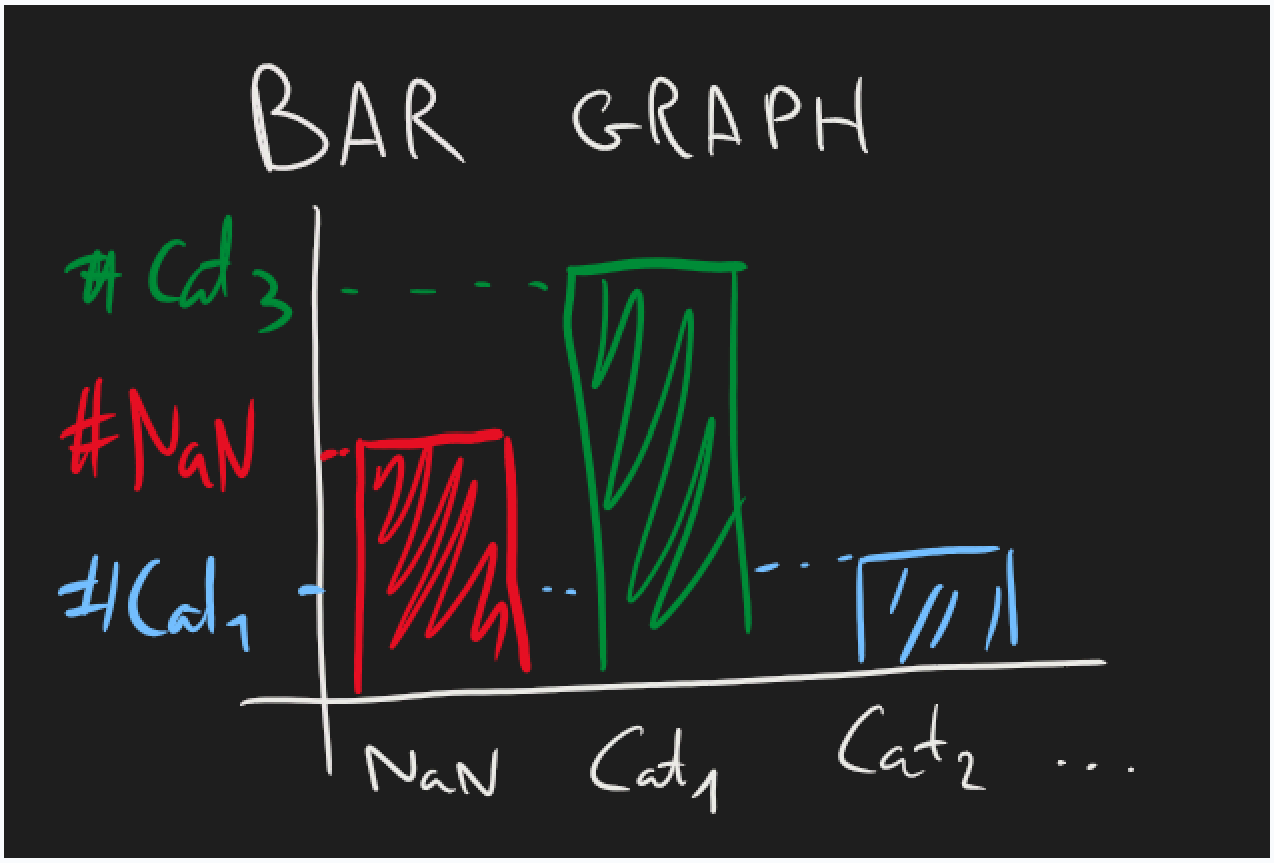
2. Main Visualization Techniques

Now we're going to see the main visualization graphs. Some charts might be favourable, depending on type of data and goals.

2.1. Bar Graphs

BAR GRAPHS. Graphs where each categorical information are represented by their absolute frequency. Use cases:

- Figuring out the distribution of a variable
- Detecting outliers (to be defined later)
- Detecting missing values, represented in a separated box
  Moreover, they can also take the form of a *stacked bar chart*, where each *"part"* of a bar chart is divided by a certain discriminant (e.g. common category across variables)



2.2. Histogram

HISTOGRAMS. They are *bar graphs* where a *quantitative variables is aggregated into bins* and represented as a *bar graph*, where each category is a bin.

2.2. Box & Whisker

BOX & WHISKER PLOTS. Supposing that our variable X is qualitative, with our samples $(x_n)_{n\leq N}$, we define the following:

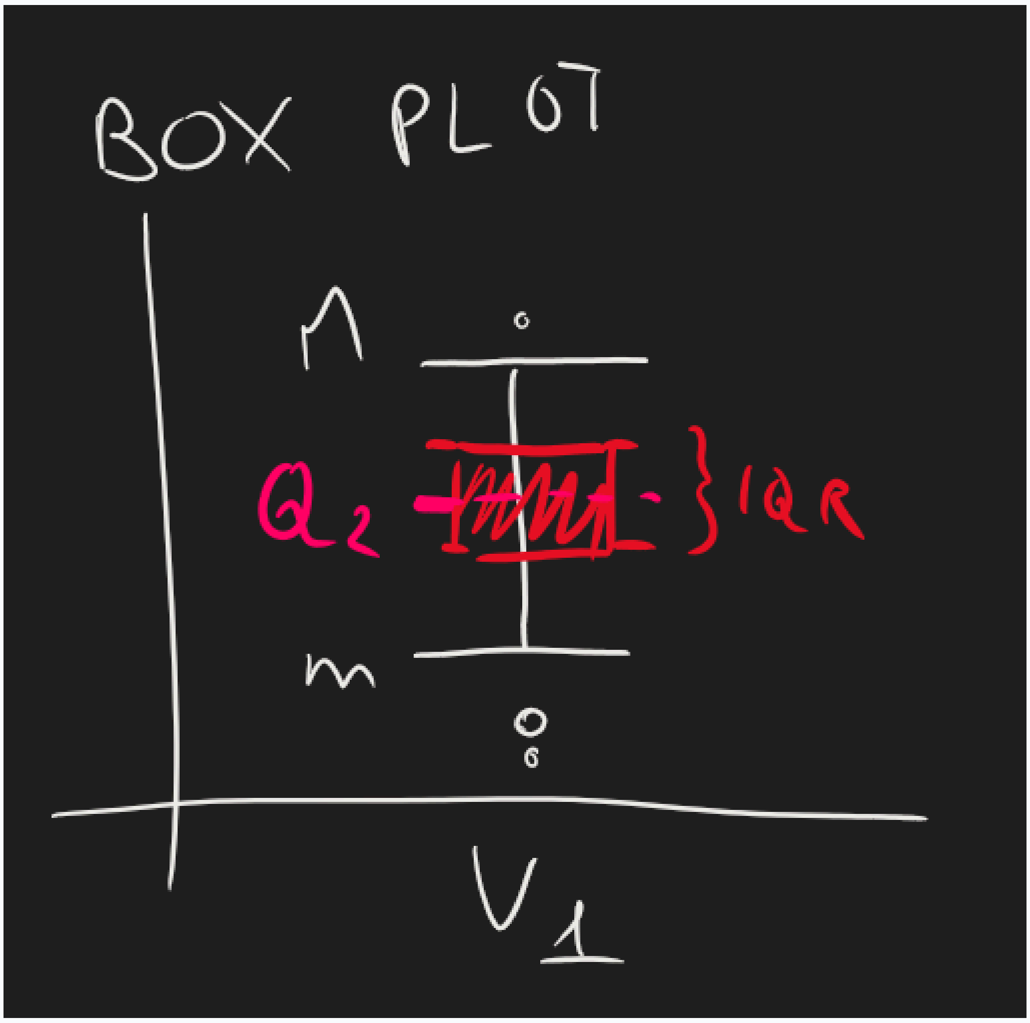
- $\forall\beta \in [0,1]$, the $100\beta\%$-quantile is a value x_n such that we have

$$\frac{\sum_{Y\leq x} 1}{N} = \beta$$

  the value is denoted as $Q(\beta)$.
- $Q_1 := Q(.25), Q_2 := Q(.50), Q_3 := Q(.75)$
- Let the *interquartile-range* be defined as $\text{IQR} := Q_3 - Q_2$
- Let the *boundaries* m, M be defined as $m := Q_1 - 1.5 \cdot \text{IQR}$, $M := Q_3 + 1.5 \cdot \text{IQR}$
- $Q_0 = m, Q_4 = M$

Having defined those statistical measures, we can condense them into a single plot. Use cases:

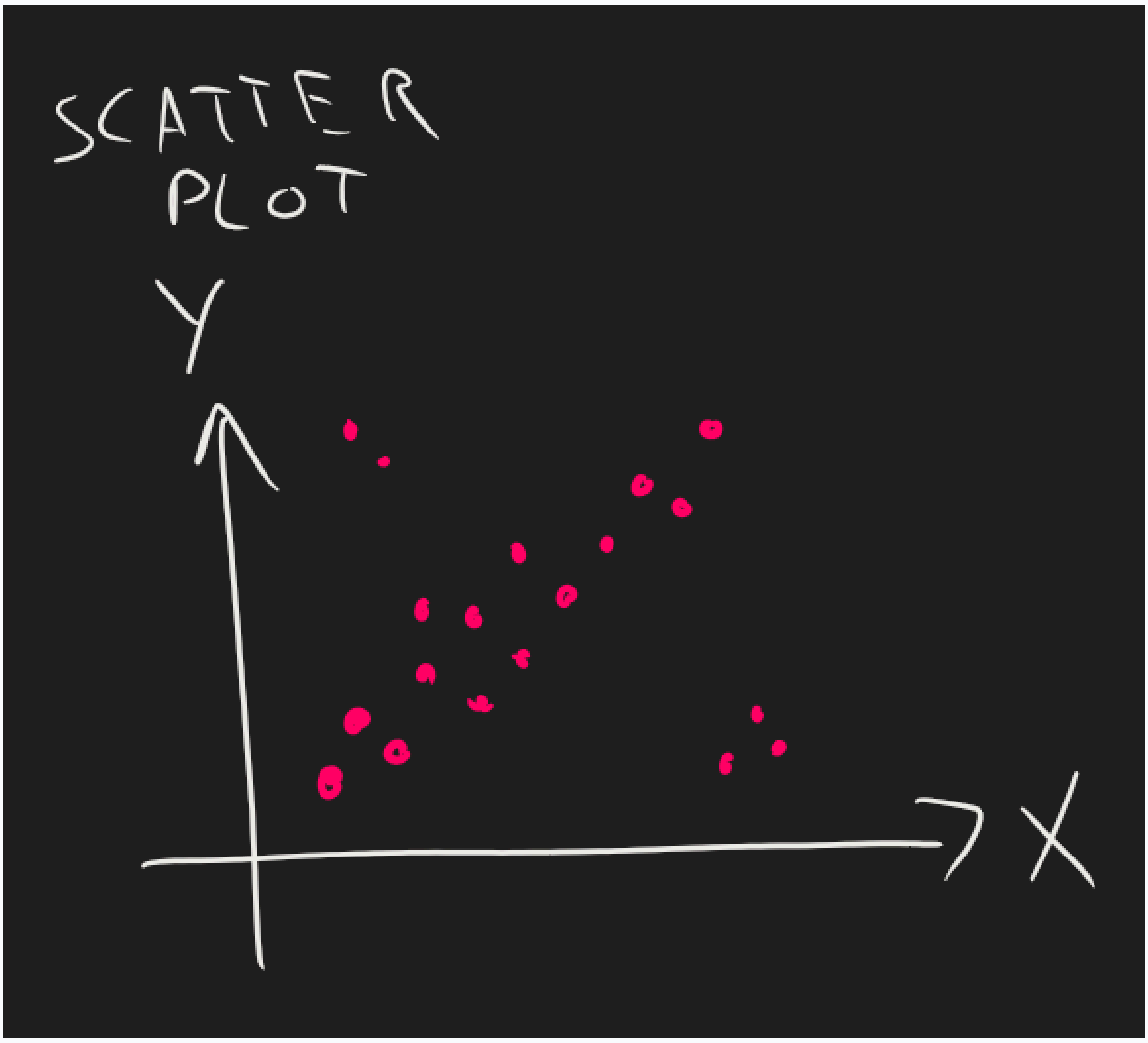
- See data dispersion
- See quartiles
- See outliers (values out of the range $[Q_0, Q_4]$)



REMARK. Note that this method works *best* for normally distributed variables, especially when used to detect outliers.

2.3. Scatterplot

SCATTERPLOT. Supposing two quantitative variables X, Y, we denote each couple $v_n = (x_n, y_n)$. Plotting each point v_n in the $\mathbb{R}^2$ space gives us the *scatterplot*.



This reveals to be useful for the following:

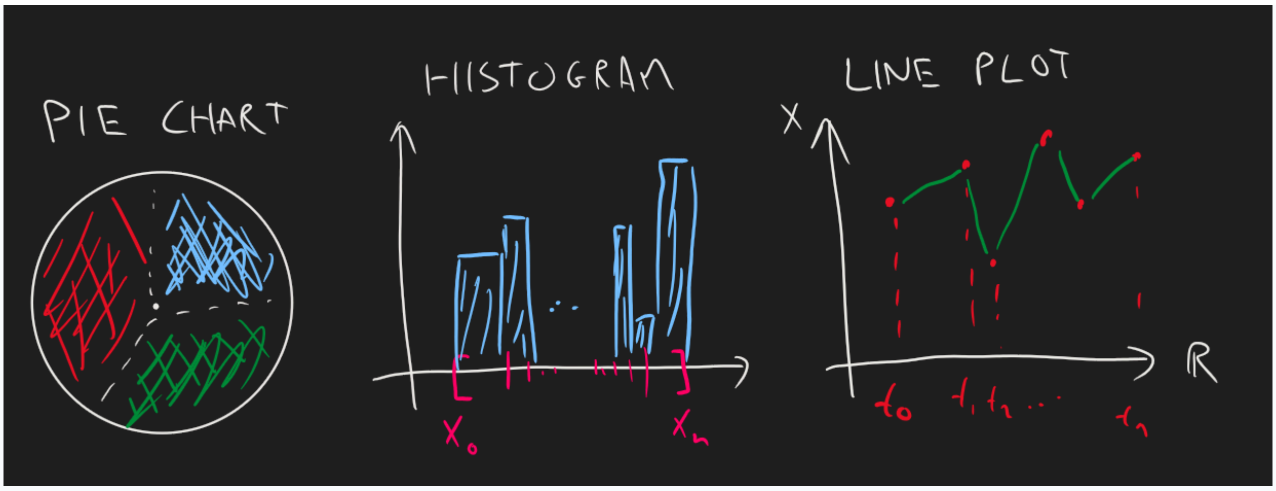
- Determining correlation between the variables
- Seeking potential *multidimensional outliers*

2.4. Other charts

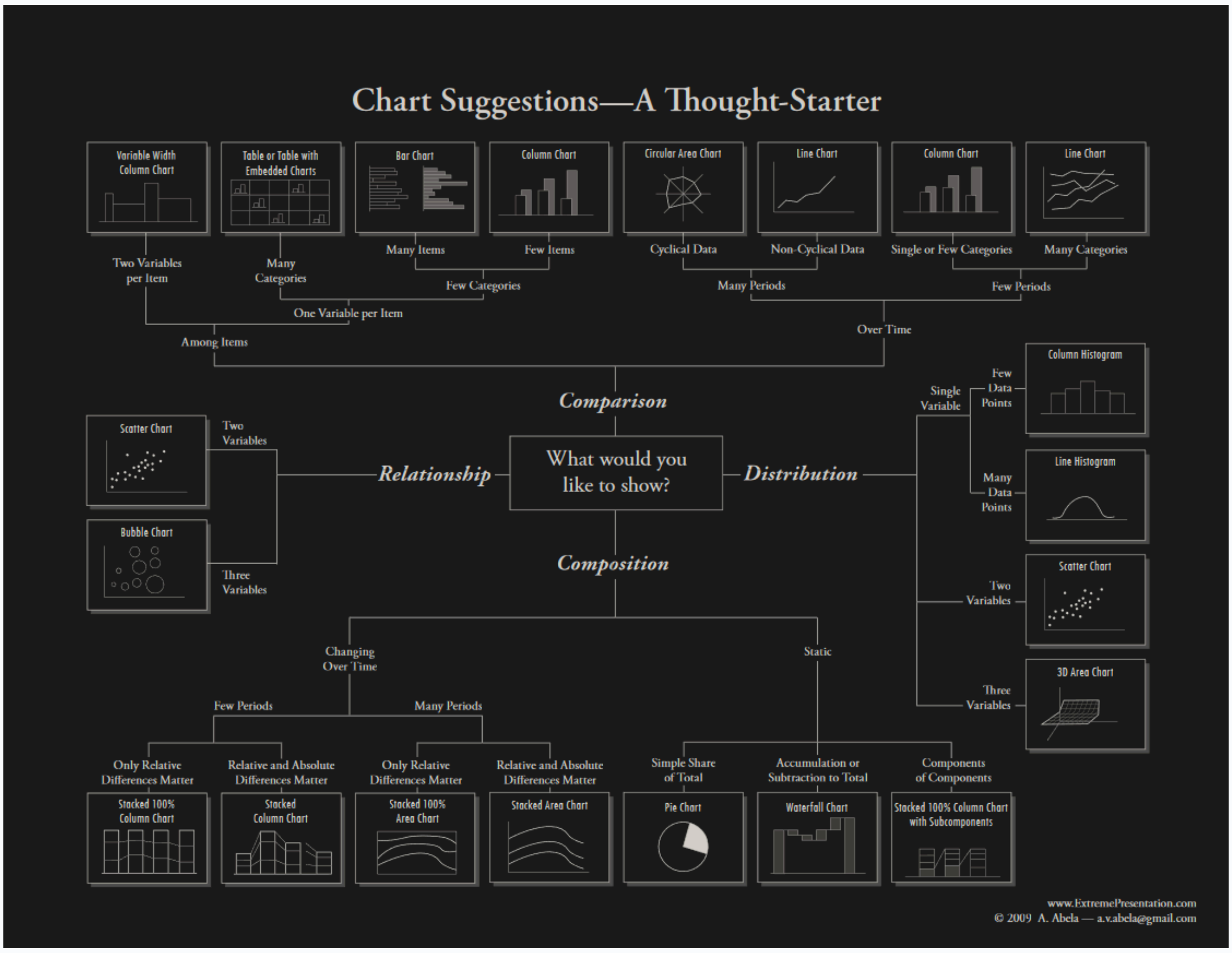
PIE CHART. For comparing the relative frequencies of categorical data. Suppose we have a *"pie"*, each category has a proportion equal to its relative frequency $f_r(A)$.

LINEPLOT. Ideal for visualizing the trend of one (or more) numerical values over a continuous domain, such as time. Similar to scatterplot, except points are connected with a line.

HISTOGRAM. Given a numerical variable, we can divide it into partitions (*"bins"*) and represent them with a box plot.



2.5. General Scheme



X

Determining and Solving Outliers

Determining and Solving Outliers

X

Everything about outliers: general definition, various methods to define outliers. Multidimensional outliers, way of determining dimensional outliers (k-means).

X

0. Voci correlate

- Data Science Process
- Methodology in Data Science
- Introduction to Data Preprocessing

1. Introduction to Outliers

DEFINITION. An *outlier* is an *observation* of a quantitative variable that is distant from the other observations.

Q. Where do outliers come from?
A. Usually outliers are due to the *extreme variability* of a phenomenon, or due to experimentation errors.

PROPOSITION. The *Bill Gates effect* is when an *outlier* has a significant impact on the interpretation of a dataset (e.g. estimators or statistics). For example, the sample mean $\hat{\mu}$ can be significantly increased (or decreased) with an outlier!

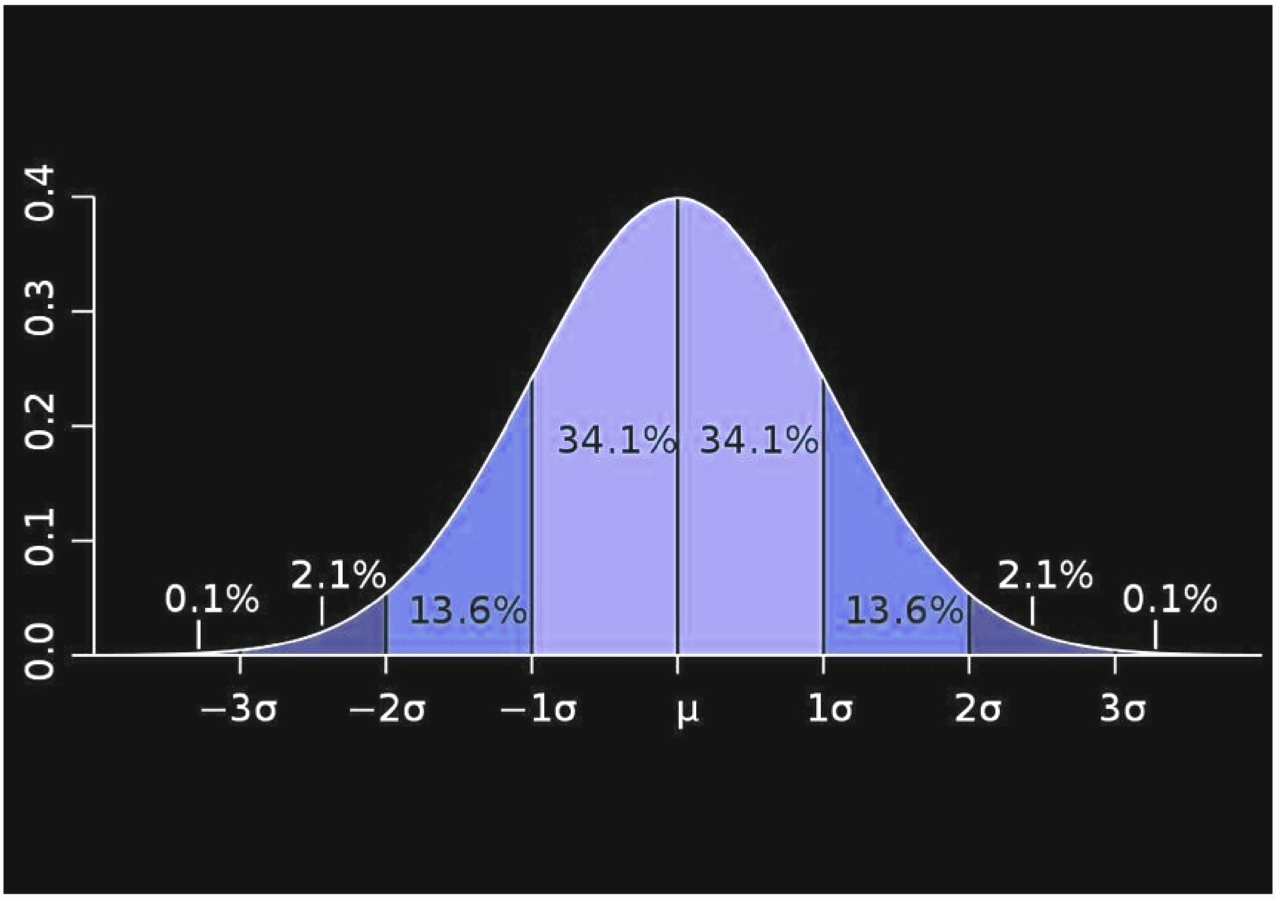
REMARK. If an *outlier* is due to the *variability of the phenomenon*, it is best to do separate analysis for them, instead of just discarding them; they might lead to more insights...

X

2. Statistical Methods for Detecting Outliers

A part from graphical methods (Data Visualization Techniques for EDA; histograms or box plots), we can also have statistical criterions to define outliers.

1. A first and trivial approach is to arbitrarily define a *"domain"* range for our variable (useful if we have metadata)
2. If $X \sim \mathcal{N}(\mu, \sigma^2)$, then the *"domain"* range can be defined as $\mu \pm 3\sigma := [\mu + 3\sigma]$



X

3. Multidimensional Outliers

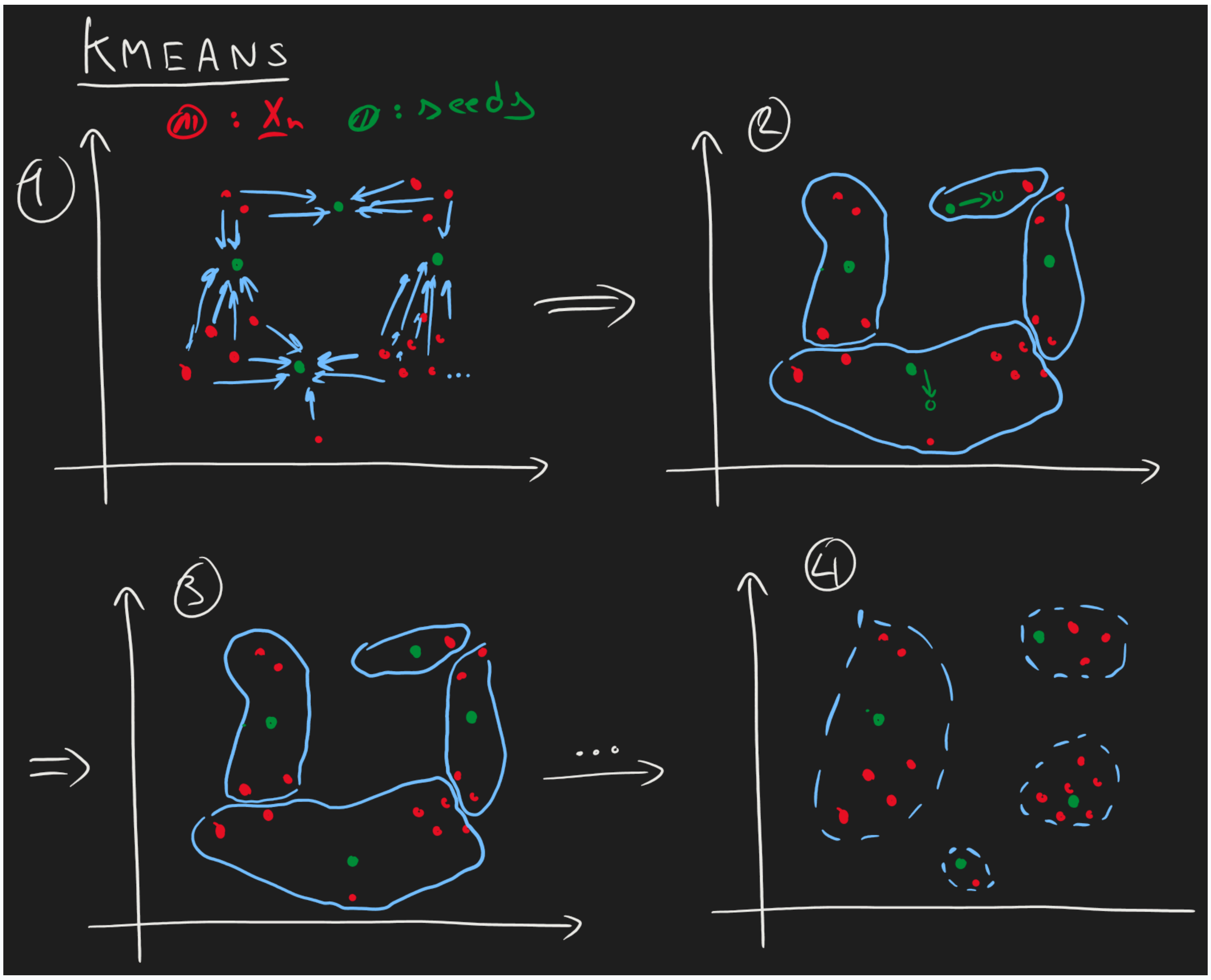
Let us generalize the concept of outliers in more dimension.

DEFINITION. Let $\mathbf{D} := \bigtimes_n D_n$ the multidimensional variable. An observation $\underline{x} \in \mathbf{D}$ is said to be a *multidimensional outlier* if *at least two or more components of the observation* $\pi_n(\underline{x})$ are distant from the other observations. In other words, a multidimensional outlier might have all of their singular components as non-outliers, but when considered globally might be an outlier for the vector-points in $\mathbf{D}$.

These are the hardest type of outliers to detect, and the most interesting as well. A part from graphical visualizations, we can use a method to seek potential outliers.

ALGORITHM. *k-means* is a descriptive technique, where $k \in \mathbb{N}$ is defined a-priori and proceeds with the following iteration:

- Randomly insert k datapoints in the domain $\mathbf{D}$, marked as *"seeds"*.
- Begin iteration of type `do-while`:
  - Calculate every distance between observation points $\underline{x}_n \in \mathbf{D}$ and the seeds
  - Aggregate datapoints, according to their closest seed (so k clusters!)
  - Calculate the midpoint of each cluster (*centroid*) and set them as *"new seeds"*
- If the centroids are *"stable"*, stop iteration. Otherwise, repeat iteration



PROPOSITION. The k-means method is sensitive to multidimensional outliers. Moreover, it tends to generate clusters with similar dimensions. In particular, it tends to group outliers in small clusters.

PROOF (Idea). Think about it in terms of datapoints *"pulling"* the seeds; the non-outliers pull the seeds to each other, and as distances are small between them, we have groups of similar sized. Whereas the outliers are far from the non-outliers, and form their own group.

X

Types of Missing Values

Types of Missing Values

X

Generalities about missing values in source data: missing values in categorical data, missing values in numeric data. Looking for missing values: masked and non-masked missing values. Unknown and non-existent missing values. Types of missing values: missing completely at rando, at random and not at random.

X

0. Voci correlate

- Introduction to Data Preprocessing

1. General Definition

Missing data may be defined as the case where *valid values* on certain *variables* are not available for our *analysis*. Usually, they represent the *hardest part* in preprocessing data.

They are hard as those can take many forms:

- Blank spaces
- Specific values or categories; e.g. NaN, 99, -1, et cetera...
    - In this case, we need *metadata* to be able to distinguish actual values from missing values
- Default values
    - Due to the existence of *input forms* or *drop-down lists* with default values in information systems.
    - To detect them, we need to notice *unreasonable patterns*; e.g. we have cases where the city is *Lisbon* or *Toronto* but the country is *United States*.

X

2. Unknown or Non-Existent Data

To distinguish between types of *missing data*, we proceed by questioning.

Q. For what reason is a value missing?
A. We have two cases:

- We say that a *missing value* is *unknown* if the value exists but is unknown as it has not been captured. For example, some people might not be comfortable with sharing their age.
- We say that a *missing value* is *not applicable* if the variable is not applicable to a certain observation. For example, children should not have incomes.

X

3. Patterns in Missing Values

Q. Why do missing values happen? Is there any correlation with other variables?
A. We have three cases.

Missing Completely at Random (MCAR). When the missing data is not *related in any way* with other variables; in other words, there's only randomness in missing data.

- Usually caused by errors in measurements

Missing at Random (MAR). When the missing values *depend* on other *existing values*; however, the same variable does not depend on itself (i.e. the existing values are not related with missing values)

- A typical example is when in a survey the *women* tend to *hide their age* (this example might be sexist) To sum it up, let Y, X be the variables, if Y is MAR then we know the probability

$$p\{Y = \emptyset | X \in E\}$$

Missing Not at Random (MNAR). In this case the *missing values* depends on the same *variable itself*.

- A typical example is when people are not sharing their own incomes, whether it be really high or low
  To sum it up, we have a situation where Y is MNAR (and $\tilde{Y}$ are the actual values) iff we know the probability

$$p\{Y = \emptyset | \tilde{Y} \in E\}$$

X

Handling Missing Values

Handling Missing Values

X

Handling missing values. Generally bad ideas: suppressing variables, replacing with special values, replacing with statistics. Generally good ideas: doing nothing, considering multiple models, considering imputation.

X

0. Voci correlate

- Assumptions for Data Mining Methods
- Types of Missing Values

1. What Not to Do

As we have detected some *missing values*, now we might have to *treat them*, in order to clean our data.
First we present the following ideas which might be *simple*, but are potentially *bad ideas*.

1.1. Suppressing Data

One of the first temptations is to simply just *remove data*, either the *singular observations with missing values* or the *entire variable* if they turn out to be a problem.
However, this is not usually a reasonable idea, since that our *missing values* could potentially be of type *MAR* or *MNAR* (Types of Missing Values); by removing data, we would be throwing *important information*, which could be even characteristic of our data!
The cases where this idea is acceptable is when it is provable that the missing values are *MCAR* (Types of Missing Values).

1.2. Replacing with Special Values in Numeric Cases

If we're dealing with *numerical variables*, replacing them with a special value (such as -1 or 999) is a bad idea: it is *ineffective* and leads to *biased results*, as models use *numerical metrics* for their predictions (such as KNN with metrics in $\mathbb{R}^n$). In fact, by replacing them with a certain value we're assuming that it is always *smaller* or *bigger* than any known value (or even in between known values).

1.3. Imputing with Statistics

Another common approach is to *impute missing values* with *statistics/estimators*, such as the mean μ, median M or mode m.
In one hand, this might sound a decent idea as it helps to *maintain the distribution* of our variable. So, for

data mining techniques this might not be so bad...
On the other hand, we will also *lose true information* thus leading to *biased analysis* (example: clustering).
In particular, this changes our interpretation by a lot.
An example which would explain that this is a bad idea is as follows:

- Suppose we have a dataset with occupation, age and income
- Suppose that a CEO's and a barista's income are the missing
- Suppose that our average income is $\mu = 2500€$
- Then, does this mean that their incomes are 2500 euros? No! Clearly a *bad interpretation result*.

X

2. What to Do

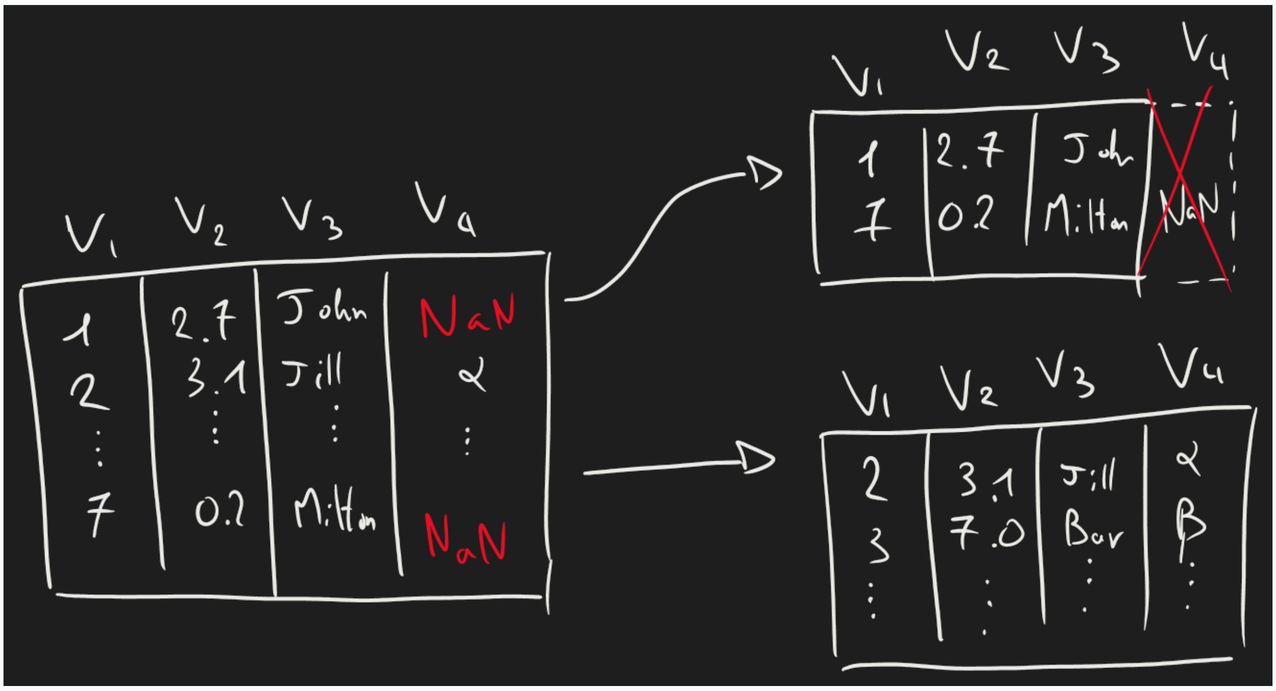
We have several alternatives, which are better for handling *missing values*. In particular, when we choose to *impute values*, we will do it in a way so the *"damage"* to the data is *minimized*.

2.1. Doing Nothing

We can do *nothing*, as certain models allow *missing values* by treating them as *separate categories* (such as decision trees of Naive Bayes). In certain cases, they might turn out to be the *best predictor* of the target variable!

2.2. Consider Multiple Models

We can split our *dataset* into more datasets, where one contains the *original dataset with no missing values* and another one contains the *dataset with no variables with missing values*. In other words, we're dividing our problem into two sub-problems.



2.3. Imputing with Predictors

We can use the *predictive methods of Machine Learning* to *impute records*. In particular, the following algorithms can be used:

- Decision trees

- k-nearest neighbours
- Linear or logistic regression
- Neural networks
- et cetera...

X

Data Transformation

Data Transformation

X

Motivations for data transformation. Techniques for transforming data. Single-variable derivations: mean-centering, standardizing, dummying, binning. Variable combinations: linear combinator, Box-Cox transformer, power and exponential (and inverse) transformers.

X

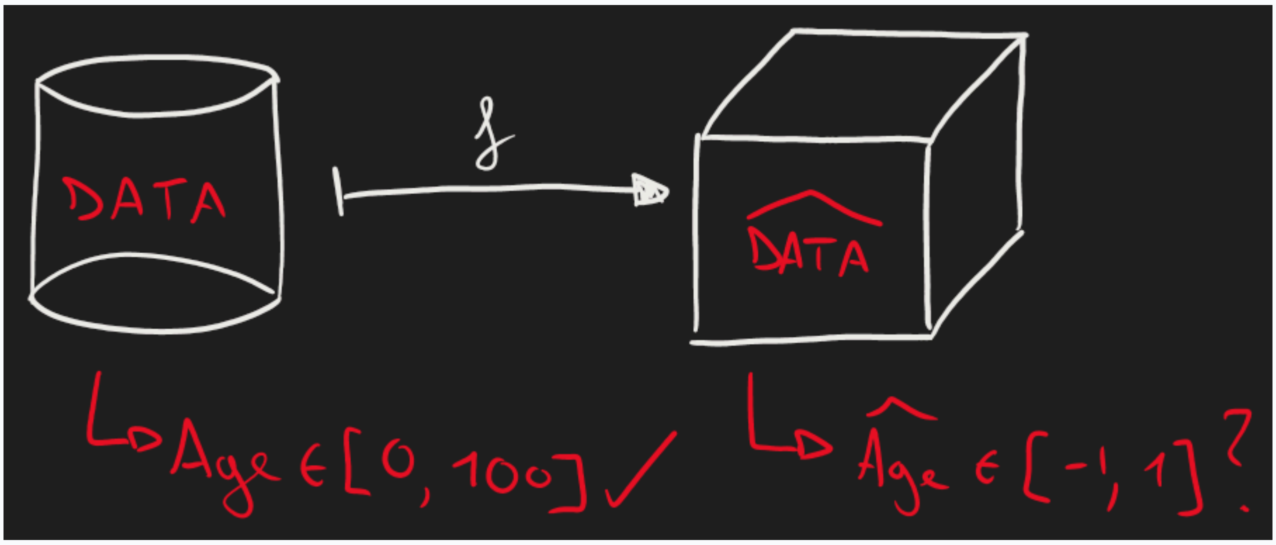
0. Voci correlate

- Assumptions for Data Mining Methods

1. Motivations for Data Transformation

MOTIVATION. After cleaning our data from outliers or missing values, our job might still not be done yet. In particular, certain data mining methods have further requirements for our data, such as *removing correlation between data* or *standardizing distributions*. So, we might still want to transform our data in a way or another, like converting from numerical to categorical (or vice versa).

REMARK. Data transformation may affect our interpretation of results, especially when scaling numerical data. So, when we are interpreting the results of our data mining methods, it is usually better to use the *original variables*.


DATA
f
$\widehat{\text{DATA}}$
$\hookrightarrow \text{Age} \in [0, 100]$ ✓
$\hookrightarrow \widehat{\text{Age}} \in [-1, 1]$?

X

2. Single-Variable Derivation

OBJECTIVE. Transform a data column into another (or more) data column(s). Mostly done with statistical methods

$$f : \mathbb{D}^1 \longrightarrow \mathbb{D}^{n \geq 1}$$

2.1. Mean corrector and standardizer

#Definizione

Definizione (mean-corrector and standardizer).

Let $(x_n)_n$ be the instances of a numerical variable X in a dataset. Defining $\bar{x} := \frac{1}{n}\sum_n x_n$ as the sample mean and $\bar{s}^2 := \frac{1}{n-1}\sum_n (x_n - \bar{x})^2$ as the sample variance, we have the following transformations:

- *Mean corrector*

$$M(x_i) = x_i - \bar{x}$$

- *Standardizer*

$$N(x_i) = \frac{M(x_i)}{\sqrt{\bar{s}^2}}$$

#Osservazione

Osservazione (use cases of mean corrector and standardizer).

M, N are useful for dealing with *incompatible units* between data columns. Used in models where the metric distances are used, such as k-*nearest neighbours*.

2.2. Dummy transformer

#Definizione

Definizione (dummy transformer).

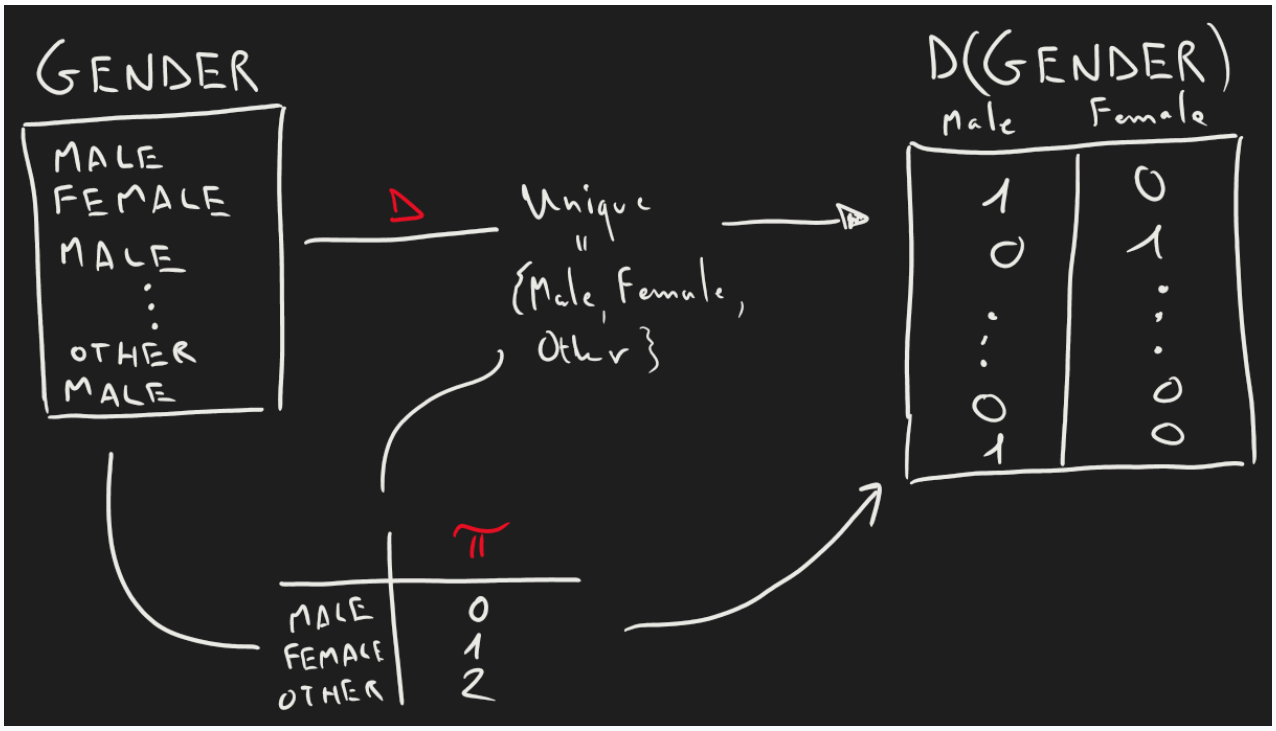
Let $(A_n)_n$ be the observations of a categorical variable X in the dataset. Assuming there is a bijective map $\pi : \text{Unique}(A_n) \longrightarrow \mathbb{N}$ (i.e. we can *"count"* every category), the *dummy transformer* is defined as the function D

$$\begin{aligned} D : X &\longrightarrow \{0,1\}^{|\text{Unique}(X)|-1} \\ A_i &\mapsto \underline{b} \end{aligned}$$

where the boolean vector $\underline{b}$ is defined as

$$\underline{b} = (b_1, \ldots, b_n) \Longleftarrow b_i = \begin{cases} 0, \pi(A_i) \neq i \\ 1, \pi(A_i) = i \end{cases}$$

In other words, we're transforming single categories into boolean vectors with sum 1.



#Osservazione

Osservazione (dropping one column).

Why do we create $|\text{Unique}(X) - 1|$ columns instead of $|\text{Unique}(X)|$? Because if we acted like that, the tuple $\underline{0} = (0, \ldots, 0)$ would have no meaning, creating wasted space and interpretability problems. With defining $|\text{Unique}(X) - 1|$, we have that $\underline{0}$ is defined as the last element in the bijection π.

#Osservazione

Osservazione (use case).

The dummer D is useful for transforming qualitative into quantitative data.

#Osservazione

Osservazione (why do we not do ordinal encoding).

Why do we not just simply use the map π (i.e. use ordinal encoding)? Because this would lead to creating some sort of *ordinality* between the variables, which might not make sense in our context.

#Osservazione

Osservazione (what if we have too many elements).

Q. What if $|\text{Unique}(X)|$ is too large? We would have too many variables in our transformed data
A. We have mainly two approaches:

- Aggregate them further. Example: we can aggregate nations into continents, reducing significantly the number of categories

- Replace them with *proxy variables*, which are ought to be *highly correlated* (if not perfectly) with the original variable. Example: GDP can be a proxy variable for a nation.

2.3. Binner

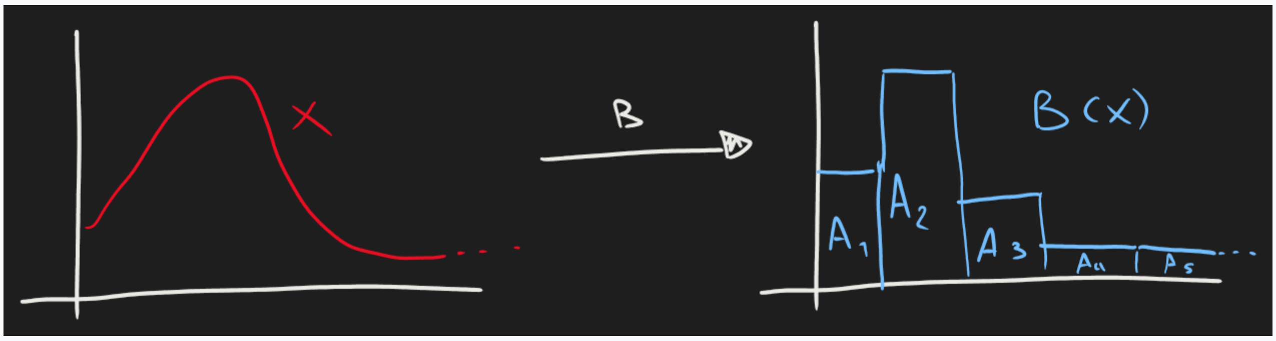
#Definizione

Definizione (binner).

Let $(x_n)_n$ be the observations of a numerical variable X in a dataset. Let $(A_k)_k$ be sets (can have an order!) or categories which at least form a partition for the domain of X.

Then $B(x_i) := \chi_{A_j}(x_i) \cdot \sup A_j$ is defined as the *binner*.

In other words, we're dividing our continuous range of numbers into a discrete range of numbers and treating them as histograms.



#Osservazione

Osservazione (use cases).

Why would we ever want to transform *numerical variables* into *categorical variables*? We're losing information, after all...

- Some models work better with quantitative variables (Decision trees)
- In some cases it's a requisite (χ^2, Naive-Bayes)
- It is effective for classifying outliers or skewed data.

#Osservazione

Osservazione (types of binner).

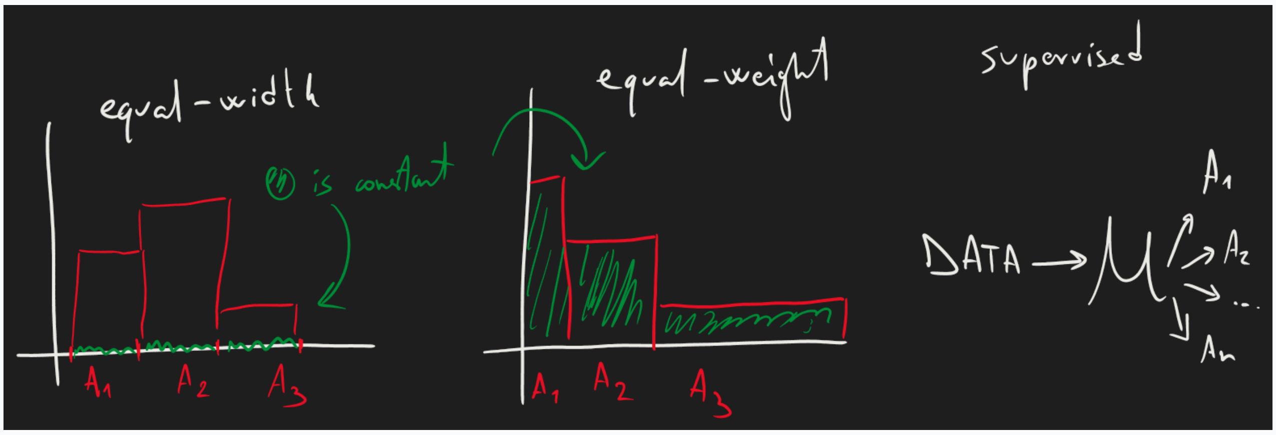
It seems that the *bins* $(A_i)_i$ have to be *arbitrarily defined*. In fact, we have the following conventions for defining the bins.

#Definizione

Definizione (equal-width, equal-weight and supervised binners).

Let B be a binner from X to the categories $(A_i)_i$.

- If $\forall i, \|A_i\| = k \in \mathbb{R}$, then B is an *equal-width binner*
- If $\forall i, \int_{A_i} 1 = k \in \mathbb{R}$, then B is an *equal-weigth binner*
- If $(A_i)_i$ is determined with a *supervised model*, then B is a *supervised binner*



X

3. Variables Combination

OBJECTIVE. Transform one or more variables into one variable. So, something like

$$f : \mathbb{D}^{n \geq 1} \longrightarrow \mathbb{D}^1$$

3.1. Linear Combinator

#Definizione

Definizione (linear combinator).

Let $B_1, \ldots, B_n$ be *numerical variables* of a dataset. Then, given $\underline{w} \in \mathbb{R}^n$ and $\underline{x}_i = (b_{1i}, \ldots, b_{ni})$ we have the *linear combinator* L defined as the following:

$$L(\underline{x_i}) = \langle \underline{w}, \underline{x_i} \rangle = \sum_n w_n x_{ni}$$

3.2. Box-Cox Transformer

#Definizione

Definizione (box-cox transformer).

Let x_i be an instance of a numerical variable X.

If $\forall x_i \in X, x_i > 0$, then we have the *Box-Cox transformer* defined as

$$\text{BX}(x_i) = \begin{cases} \frac{x_i^{\lambda - 1}}{\lambda}, \lambda \neq 0 \\ \ln(x_i), \lambda = 0 \end{cases}$$

Otherwise it is defined as:

$$\mathrm{BX}_C(x_i) = \begin{cases} \dfrac{(x_i + c)^{\lambda - 1}}{g\lambda}, \lambda \neq 0 \\ \dfrac{\ln(x+g)}{g}, \lambda = 0 \end{cases}$$

Note that usually $\lambda \in [-3, 3]$ and $g = \mu_X$.

#Osservazione

Osservazione (use case of Box-Cox).

Usually Box-Cox is used to transform any type of variable into a variable that *"looks similar to a normal variable"*.

3.3. Other Consideration

A lot of other *"creative methods"* can be used, such as:

- Using $\sqrt{x}, \ln(x), \log_{10}(x)$ for correcting *right-skewed data*.
- Similarly using $x^{|k|}$ for *left-skewed data*.
- Define arbitrary functions for specific cases
  - Example: BMI from weight and height.

X

Data Reduction

Data Reduction

X

Data reduction. Motivations for data reduction: risk of multicollinearity, of overfitting, of data sparseness. Curse of dimensionality. Options for handling sparse data: feature selection, Principal Component Analysis, how to interpret a dendogram, variable clustering.

X

0. Voci correlate

- Feature Selection
- Methodology in Data Science

1. Motivations for Data Reduction

Q. Is data always good?
A. As a *"corollary"* of the *Central Limit Theorem*, one usually thinks that *more data* is always *better*. However, let us look at this problem in a critical manner.

- *Instances of data*: usually there's no problem with having more *instances of data*, computational expense aside. Some models might require *instance selection* to optimize calculations, such as *KNN*

(Instance-Based Learning)
- *Features*: here the story changes completely. When we have too many variables, we might have the risk of having *data sparseness* (which is when we have too many missing values), of having *highly correlated variables* and of having *overfitting models* (as features' number is associated to model complexity).
  As we can see, our focus is on *problems with having too many features*. In the next sections, we will se how to deal with them.

X

2. Dealing with Problems

PROBLEM. (*Multicollinearity*) Some variables become *"more important"*, especially when we have lots of them; in fact, information would be represented n-fold times.
This is significant for:

- Regression problems as we cannot evaluate individual effects
  - This creates *unstable coefficients* and makes the model *less interpretable*
- Statistical analysis as some assumptions are not valid

OBSERVATION. In some models, this is not a problem (like *decision trees*)

SOLUTION. We can either:

- Drop the variable that is *least correlated with the target variable* (or pick one if we're dealing with cluster problems)
- Create a compound variable

PROBLEM. (*Overfitting*) Model might *"memorize"* the patterns, which is not learning. Becomes a problem especially when the model starts to use *non-discriminant variables*.

PROBLEM. (*Sparseness*) *Sparse data* means that in customer signatures the *most of values are either zero or missing*. The more variables we add, the sparser data gets

EXAMPLE. Suppose that we want to collect information about customers. If we want to collect their age and gender, we may except no missing values at all. However, if we start to add other variables such as *income* or *first home bought*, we might get *missing values* as they might not be applicable to our population.

DEFINITION. (*Curse of dimensionality*) To summarize every problem above, we can say that they're a part of the so-called *curse of dimensionality*, which affects majorly models based on *distance metrics* (as it is harder to track groups). Other problems than the ones mentioned above are:

- Increased computation
- Distances lose meaning
- Visualization challenge

X

3. Feature Selection

Go see Feature Selection for specific *feature selection methods* which involve *predictive models*.

X

4. Principal Component Analysis

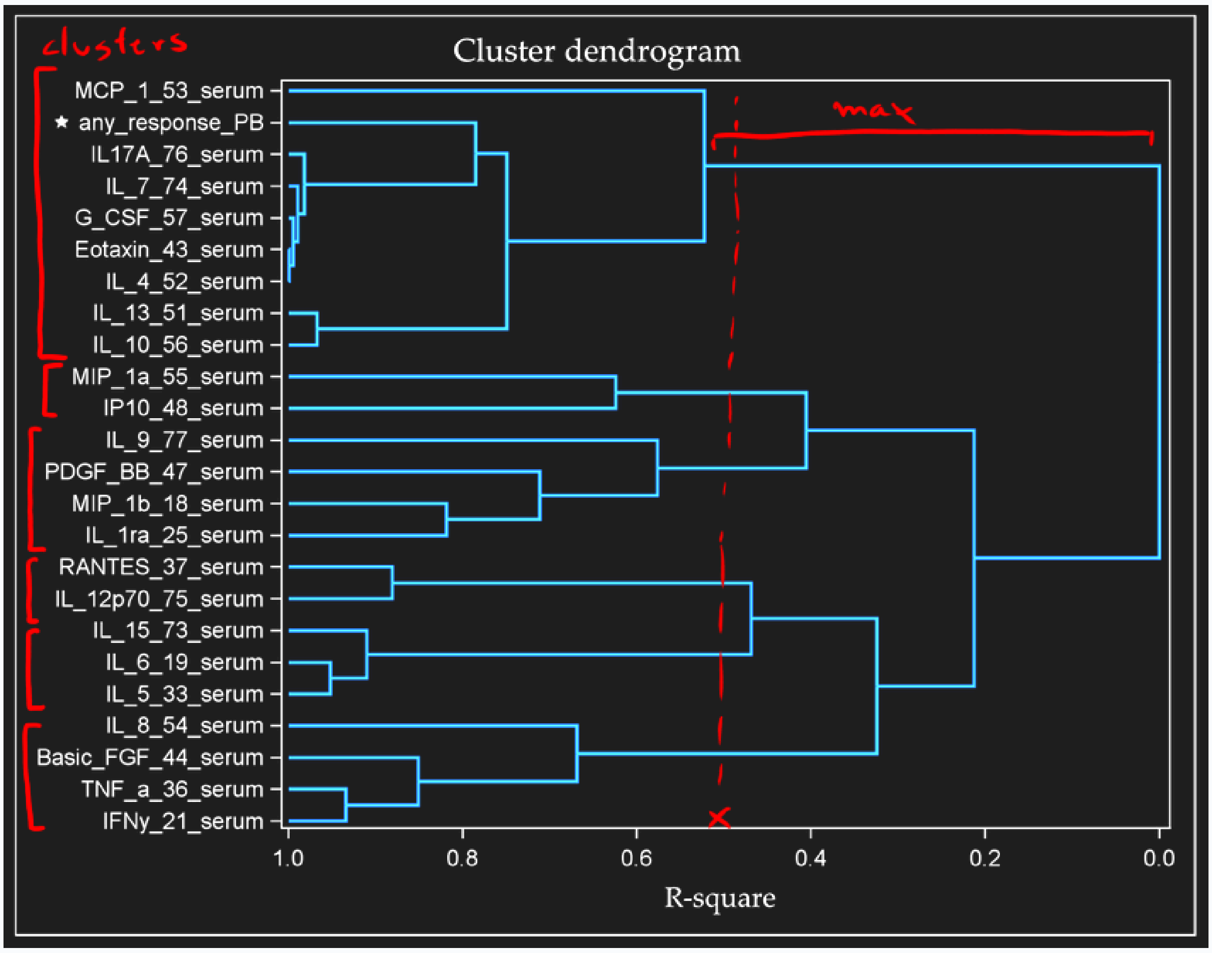
Another approach for *data reduction* is *principal component analysis*.

DEFINITION. A *principal component* is a slight variation of the best-fit line which examines the correlation between variables. It minimizes the sum of the squares of the distances from the line to the data points (so not only the vertical distance!).

PRINCIPAL COMPONENT ANALYSIS is an *unsupervised learning algorithm* which examines the *correlation between two variables* through a *principal component*.

We will use *PCA* to do *variable clustering*. We can obtain variable clusters in two forms:

- **Dendogram** represents groups the variable together in a *hierarchical way*, and the *"distance"* between the cluster level represents the *difference in proportion of variance*.
  - Usually we decide the clusters by *"cutting"* the vertical line at the *"biggest step"*.



- Alternatively we can have a simple *graph* where each cluster is a *node* connected to its member variables.

Q. I have the clusters, containing multiple variables. How do I combine them?
A. I have three approaches:

- Manually choose the variable which is the most *"descriptive"* in each cluster. However, this requires *human interpretation* so this is cannot be automatized.
- Use the *first principal component of the cluster*; this can be automatized but is not interpretable
- Find the variable that has the most *predictive power* in the cluster by using statistical criterions

Analytical Base Tables and Transactional Tables

Analytical Base Tables and Transactional Tables

X

Definition of signature, of analytical base table (ABT). Motivations for having ABTs. Definition of customer. Definition of transactional table. Types of transactions: anonymous, linked to a card, to a cookie, to an account and to a customer.

X

1. Definition of Signature or ABT

DEFINITION. A *signature* or *analytic-base table* is a *source of data* such that each *"customer"* is represented only one time (by the so-called primary key). In other words, it is the data in right format to be used for *predictive models*.

As we can see, definition of customer can vary.

DEFINITION. A customer is basically the *"unit of study"* for our data science project. So, the definition of *customer* depends on the business goal. For example, there might be types of customers:

- *Payer*: the one who executed the transaction
- *Decider*: the one who decided the transaction
- *User*: the one using the product from the transaction
  Note that the three of them can be exclusively disjunct from each other. For example, in a transaction of buying a toy, the father might be the payer, the mother the decider and the kid the user.

X

2. Transactional Table

Q. Do we *always* have *ABTs* ready for our analysis?
A. Sadly that is rarely the case: usually, from the information systems we can get *transactional tables*.

DEFINITION. A *transaction* is an instance where the rows are uniquely identified solely by the *transaction act*, not by the *customer*.

OBSERVATION. Just because we have a transactional table does not mean we cannot get any insights from it. To the contrary, we can still apply some predictive methods for it: in particular, we can use *clustering methods* to gain *association rules* for the purchases. As an example, take the canonic case of Target's pregnancy prediction.

Note that as we have *different* types of transactions, we also have *different transactional tables*. Let us do a few examples:

1. **ANONYMOUS TRANSACTIONS.** Transactions are said to be *anonymous* if it is not possible to discern the customer's identity from a transaction.
2. **TRANSACTIONS LINKED TO PAYMENTS.** Transactions paid with *debit or credit cards* may store the card number, which can eventually be used to represent a *"customer"*. However, in this case we might not fully accurately represent a customer, as it is possible to have a *many-to-many relationship*

between *credit cards and customers*. In fact, it might be common for a card to be used by ≥ 2 people or singular people having ≥ 2 cards.
3. **TRANSACTIONS LINKED TO ONLINE IDENTIFIERS.** Similar as above, but has many limitations. Due to the fact that the online identifiers (such as cookies) can be easily removed, there's no guarantee of a bijection between cookie and user.
4. **TRANSACTION LINKED TO ACCOUNTS.** In some systems, every transaction is captured and linked to an *account*. It is usually a good measure, as *accounts* usually represent the *singular customer* (for example with insurance cards).
5. **TRANSACTION LINKED TO CUSTOMERS.** The most preferable case, where every transaction is linked to a *specific customer*.

X

Building Analytical Base Tables

Building Analytical Base Tables

X

Building Analytical Base Tables: conversion from transactional tables to ABTs. Possible steps: copying, pivoting, aggregating, table lookup, summarizing and deriving variables.

X

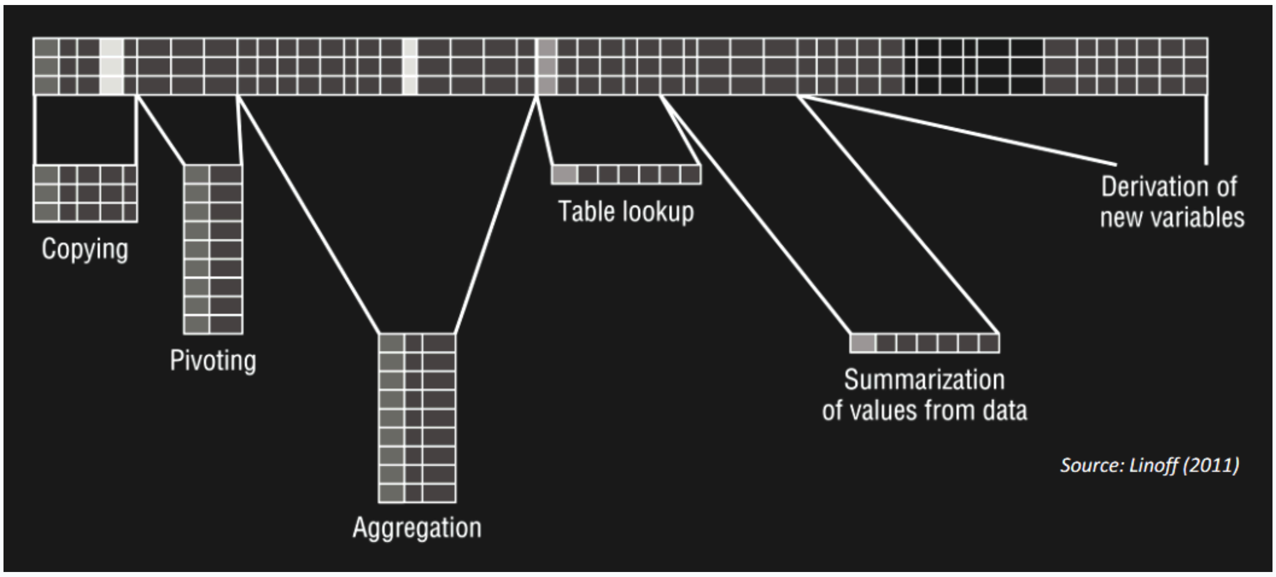
0. Voci correlate

- Analytical Base Tables and Transactional Tables

1. Main Techniques for building ABTs

Q. Suppose I have a transactional table. How do I built an analytical base table from it?
A. There isn't a unique way to do it, but here are the main techniques.

1. **COPYING.** Sometimes data is at a *correct granularity*, which means they can be directly be copied from the transactional table. For example, it might be demographical data.
2. **PIVOTING.** We can *pivot* multiple transactions into one row, across *multiple columns*. It works for cases where the transactions are done on a *regular schedule* (example: phone companies).
3. **AGGREGATING.** We can aggregate by the *customer identifier* (if possible) and get a summary for the transactions made over the time. Note that if we're dealing with time-stamped transactions, we have to decide a *time period*, which can significantly affect our final table.
   Possible summarizers are as follows:
   > - Time since first transaction
   > - Time since most recent transaction
   > - Largest, smallest transaction
   > - Average or median of transactions
4. **DERIVING VARIABLES.** If summarizing our data reduces *too much* information, we can alternatively *derive* new variables for our final table. For example, they can represent the *proportions of purchase types*.


Copying
Pivoting
Aggregation
Table lookup
Summarization of values from data
Derivation of new variables
Source: Linoff (2011)

Let us give some main guidelines for creating *new variables*.

- Usually it is important to have variables which summarize the *recent history of the client*
- It is also important to have summarized the *lifetime transaction history of a customer*
- We can also attempt to create some *"behavior fields"*
- It is usually good to use *all variables of the transactional table*!

X

Data Visualization for Presentations

Data Visualization for Presentations

X

Principles of data visualization for presentation. Motivation for visualizing data. The tradeoff between data interpretability and details. Interactivity in data visualization. Main visualization techniques (new ones). Common mistakes in data visualization.

X

0. Voci correlate

- Data Visualization Techniques for EDA

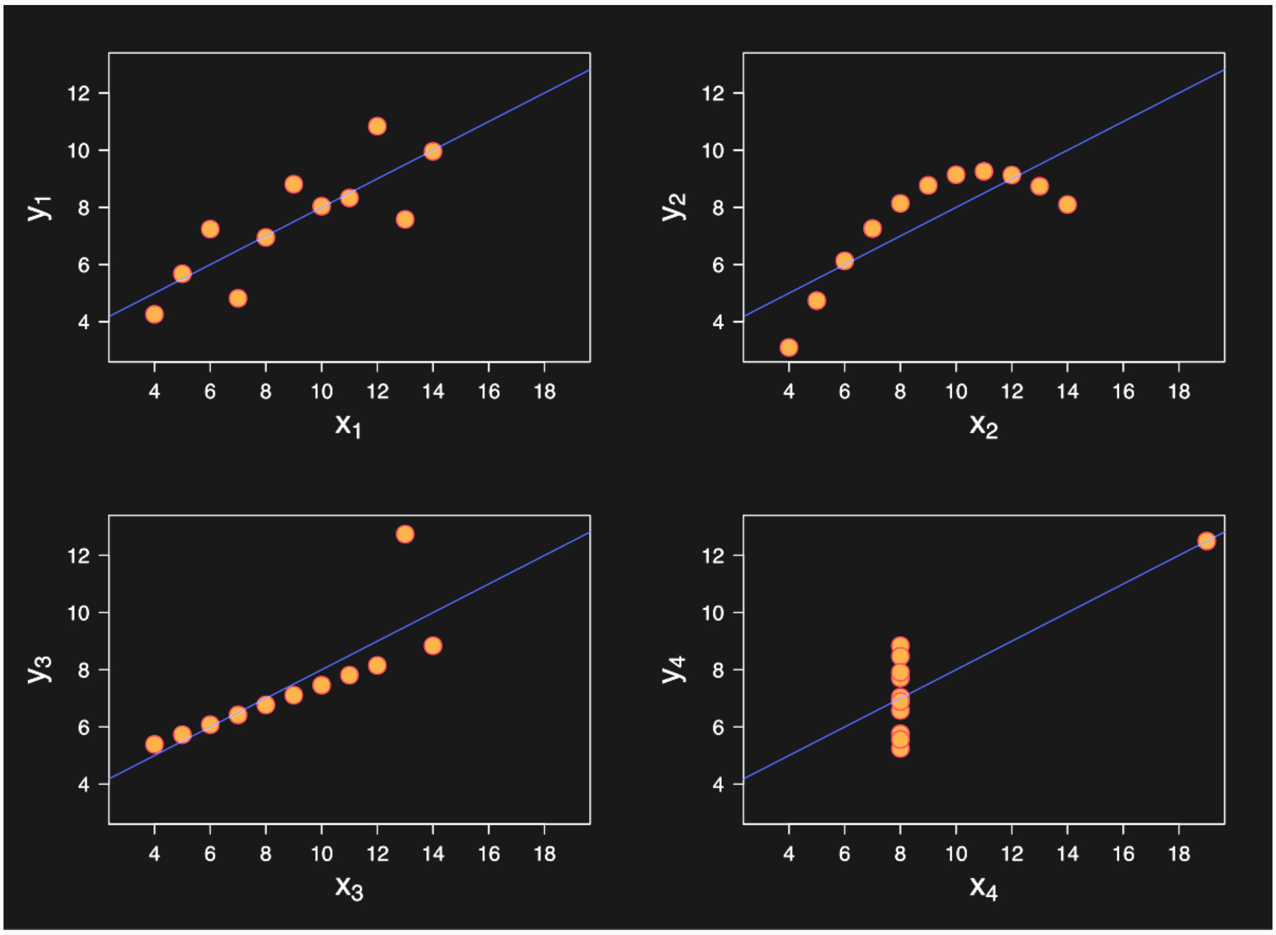
1. Motivations for Visualizing Data

Visualization is the process of transforming *data* into *visual representations* for the purpose of *data exploration*, *hypothesis confirmation* and *presentation*. We will focus on the last purpose, as we already have seen the first two with *EDA* (Data Visualization Techniques for EDA).

MOTIVATIONS. We have a long series of *motivations* for visualizing data for presenting:

- Human beings process *60.000 times faster* than plain text
    - It is estimated that 65% of human beings in the worlds are *visual learners*
- Data visualization is an *universally understood*
- We can do story telling with data

- Sometimes they provide more information that *simple summary statistics*. Anscombe's quarter shows us this.



PRINCIPLE. There is always a trade-off between *data interpretability* and its *details* when we visualize and summarize data.

INTERACTIVITY. One way to "avoid" this *trade off* is to add *interactivity* to our data visualizations. In this way, we can handle the complexity of the dataset; also users will be able to only see one *aspect of the dataset* at their will, making the presentation process even easier.

- Tools to do this are either *Plotly* or *PoweBI*

X

2. Main Visualization Techniques

PRINCIPLE. In any way, there *does not exist a picture that can communicate "the truth, the whole truth and nothing but the truth"*; we always have to make choices in abstractions and which aspects of data to emphasize, such as volume, dimension, color, context, et cetera...

2.1. Quantitative and Qualitative Variables

To know about techniques for visualizing *quantitative and qualitative variables*, see the page Data Visualization Techniques for EDA.

2.2. Spatial Axis Orientation

To understand more about the *correlation between variables*, we can use the following techniques.

PARALLEL COORDINATES. One way to represent *relationship between variables* is to use *"parallel coordinates"*, where we represent *variables* as *parallel axes* and each *line* represents the *relationship* between the variables in an *observation*.

- If we have *parallel lines*, there is *perfect positive correlation*
- If all *lines cross over each other at a single spot*, there is *perfect negative correlation*
- Otherwise we have potentially *uncorrelated data*

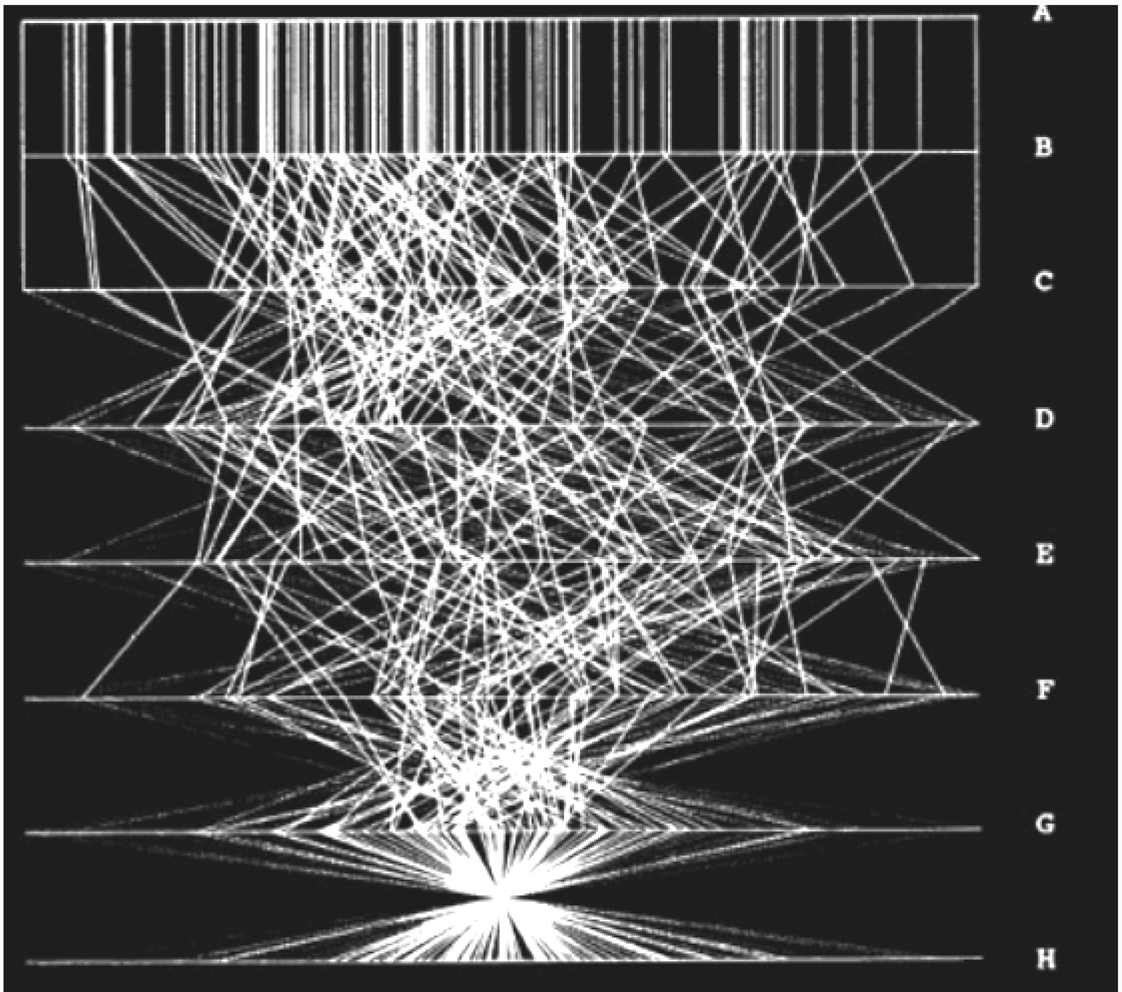


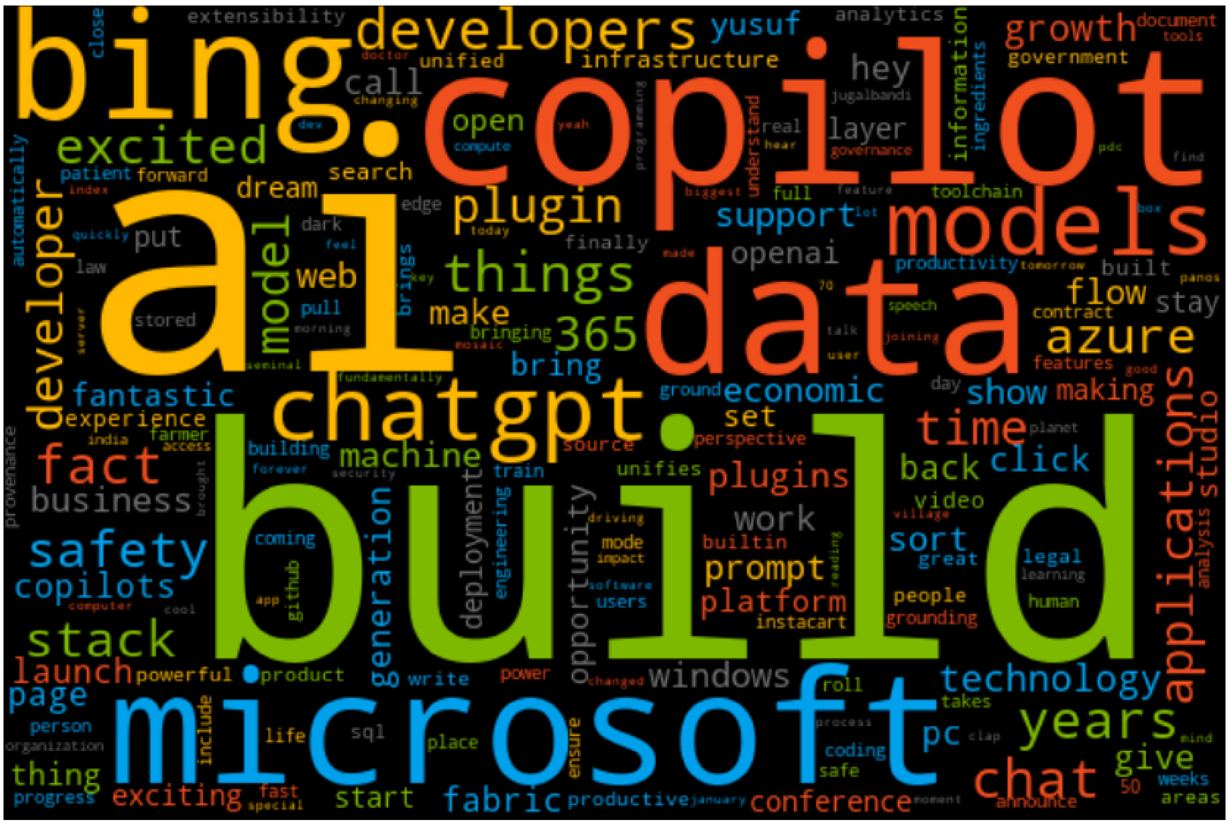
Figure 3. Parallel Coordinate Plot of Six-Dimensional Data Illustrating Correlations of ρ = 1, .8, .2, 0, −.2, −.8, and −1.

.7

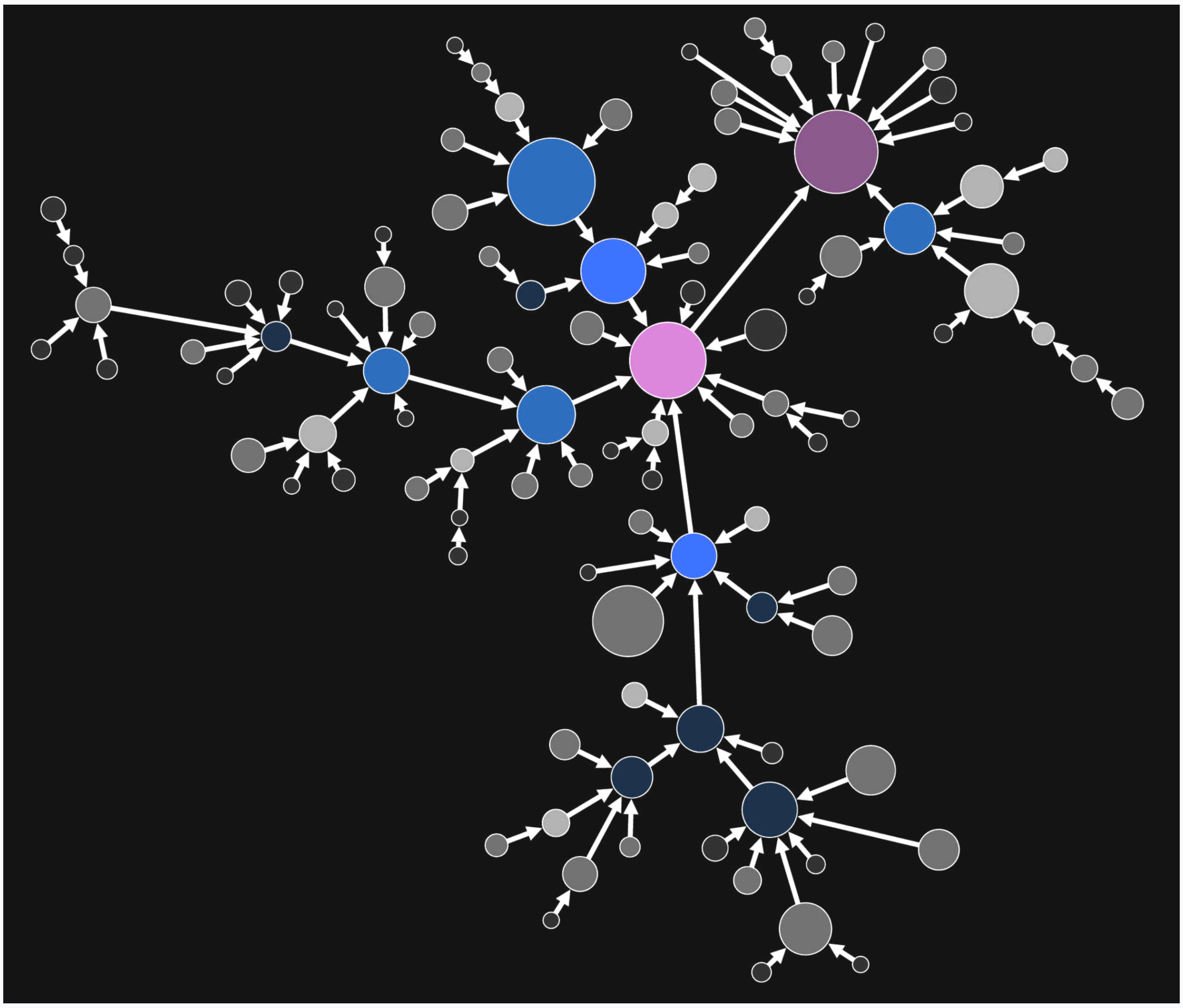
2.3. Network Data Visualization

Datasets connect deeply with other *datasets*. With *network data visualization techniques*, we can see how they relate one to another, as a *"network"*. In other words, we are demonstrating relationships between the instances of the dataset.

WORD CLOUDS represent the *most common words* within a given text dataset, where a bigger dimension is given to more common words.



NODE-LINK DIAGRAMS is the most common technique, where *nodes* are drawn as *point marks* and the *links connecting them* are *line marks*. With this, we can represent relationships between *items*.



X

3. Common Data Visualization Techniques

We end with a few pointers on how *to avoid bad data visualization*:

- Avoid redundant graphs
- Avoid using graphs that do not make sense
- Represent data scale in a faithful manner
- Avoid representing same information
- Choose colors wisely
- Do not exclude context from your visualization
- Avoid diagonal labels; if you cannot fit labels on the horizontal axis, swap with the vertical axis.
- Make proper legends by sorting codes
- Avoid cramming too much information in a single graph

X