Data Profiling

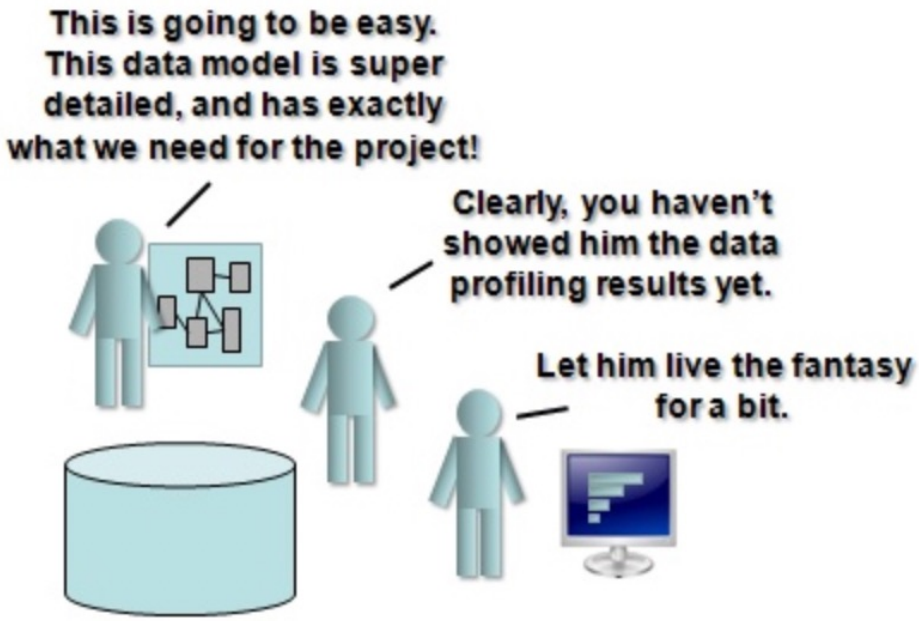
A first step before you dive into a pool of data

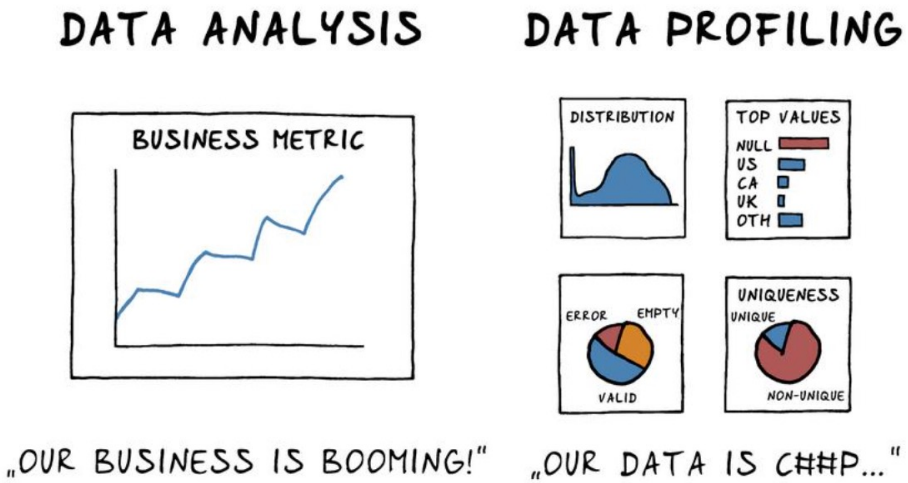
Lizda.iswari@uii.ac.id

Pesantren Sains Data CDS UII

12 April 2023 / 21 Ramadhan 1444H

Why do we need data profiling?





Data Profiling

- ”**first look at the data**”
- A critical part of the machine learning workflow.
- Start to understand the data we are working with and what it contains.
- Allows us to make sense of the the data before applying advanced analytics and machine learning

Data Profiling

- Also known as data understanding, Exploratory Data Analysis (EDA).
- Allows us to familiar with data from multiple angles, through statistics, data visualisations, and data summaries.
- Profiling data means we try to:
  - Identify patterns within the data.
  - Understand relationships between features.
  - Identify possible outliers.
  - Identify if we have missing values.

Data Profiling

- We can profile our data using the descriptive statistic-based or the visualization-based approach.
- Common library in Python: Pandas and Matplotlib.
  - Pandas has a number of functions including df.describe() and df.info() which help summarise the statistics of the dataset.
  - Matplotlib has a number of plots such as barplots, scatter plots and histograms to allow us to visualise our data.

Let’s Practise

- Earthquake data set: https://www.kaggle.com/datasets/warcoder/earthquake-dataset
- Datasets contain records of 782 earthquakes from 1/1/2001 to 1/1/2023 and has 19 attributes.
- Some attributes:
  - title: title name given to the earthquake
  - magnitude: The magnitude of the earthquake
  - date_time: date and time
  - mmi: The maximum estimated instrumental intensity for the event
  - alert: The alert level - “green”, “yellow”, “orange”, and “red”
  - depth: The depth where the earthquake begins to rupture
  - …

Let’s Practise

- Profiling based on descriptive statistics:
  - df.dtypes : to know the data type of each column.
  - df.describe(): to get a statistical summary of each column.
  - df.info(): provides a concise summary of your DataFrame
  - df.corr(): to calculate the correlation between numeric attributes.

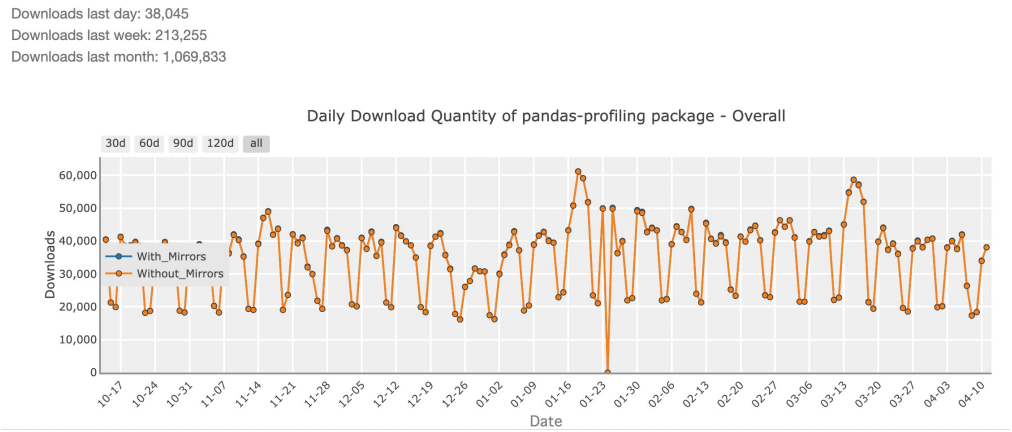
- Profiling based on visualization:
  - Regresi plot
  - Boxplot

Library for Auto Data Profiling

- When working with machine learning or data science training datasets the above methods may be satisfactory as much of the data has already been cleaned and engineered to make it easier to work with.
- In real world datasets, data is often dirty and requires cleaning.
- This can be a time consuming task to check using the methods above.
- Here, we need an auto profiling to help us speed up this part of the workflow without compromising on quality.

What is the Pandas Profiling Python Library?

- Pandas Profiling is a Python library that allows users to generate a very detailed report on pandas dataframe without much input from the user.
- According to PyPi Stats, the library has over 1,000,000 downloads each month, which proves its a very popular library within data science.



How to use Pandas Profiling in Python?

- Install the pandas-profiling

```
pip install pandas-profiling
```

- Import libraries

```
import pandas as pd
from pandas_profiling import ProfileReport
```

- Run pandas-profiling

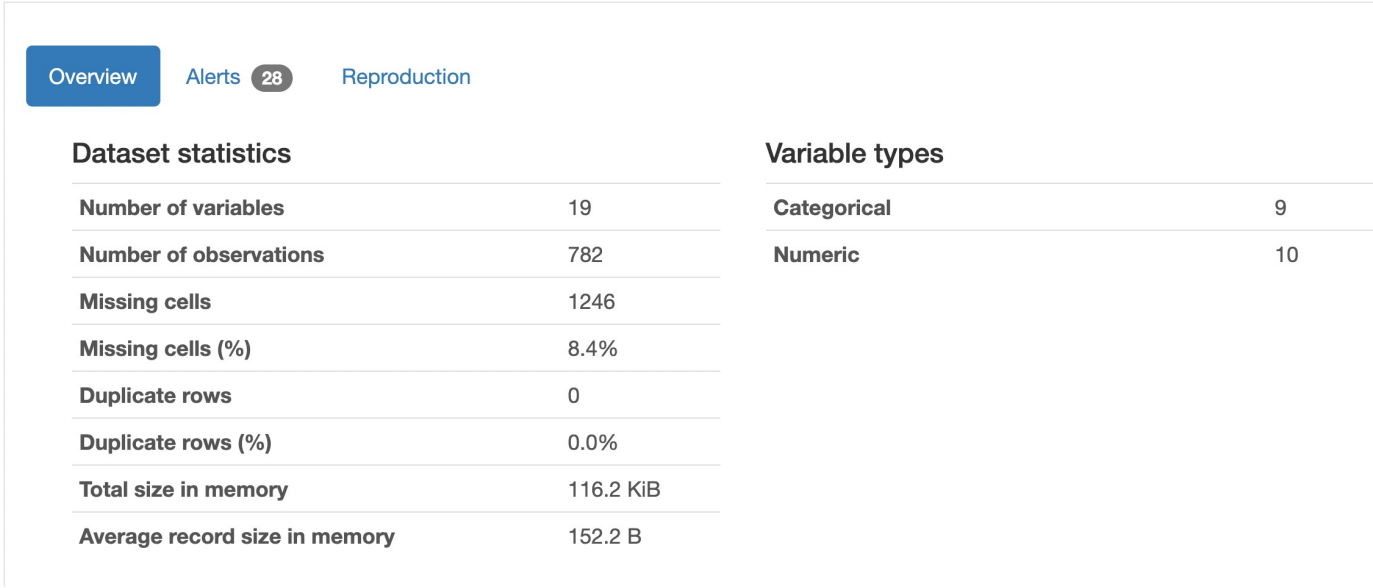
```
report = ProfileReport(df)
report
```

Structure of pandas-profiling

- **Overview**: general recap containing high-level information both concerning the dataset (number of variables, number of rows etc.) and reproducibility (configuration, package version etc.).
- **Variables**: this is the **main section**. It contains a sub-section for each variable (column) describing basic statistics (Distinct, Median, Std etc.) and plots.
- **Interactions**: it contains heatmaps for all combinations of numeric variables.
- **Correlations**: This is the **second most important section**. It contains various correlation matrices using different metrics.
- **Missing values**: it contains a barplot describing the number of missing values for each variable.
- **Sample**: it contains a sample of the dataset.

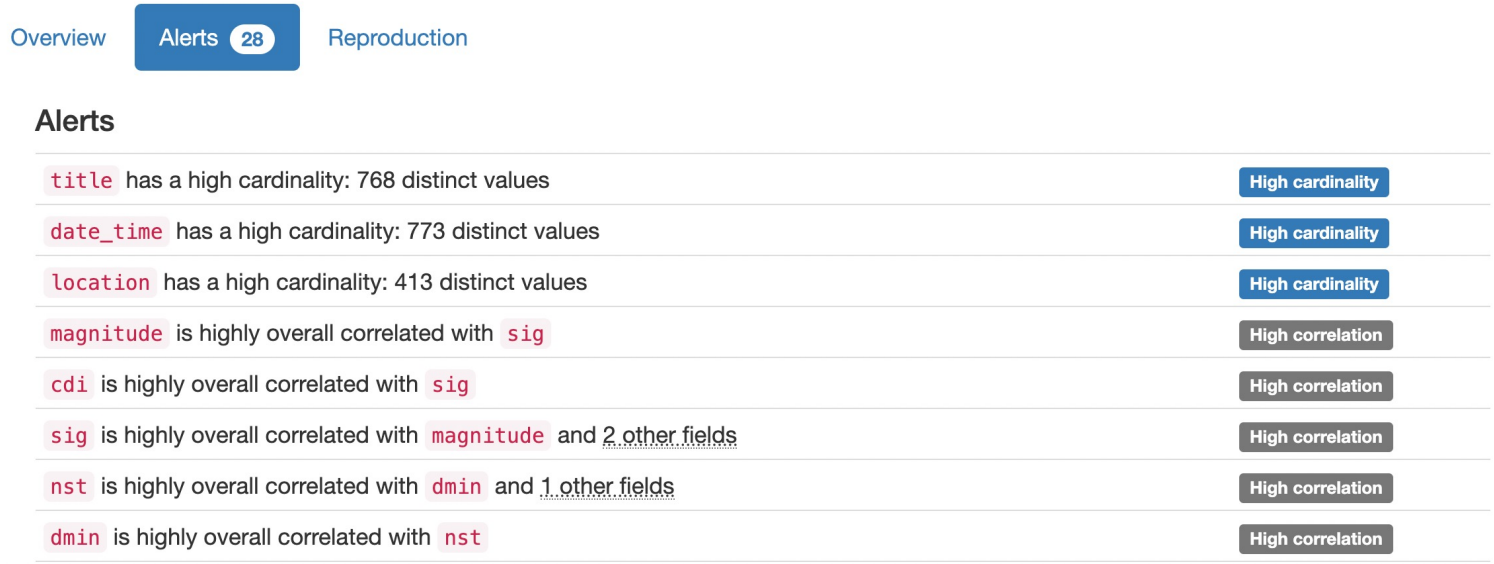
Overview

- The overview section contains three tabs: **Overview, Alerts, and Reproduction.**
- The **Overview tab** provides statistical information about your dataset including the number of variables (columns in the dataframe), number of observations (total number of rows), how many values are missing along with the percentage, how many duplicates there are, and the file size.



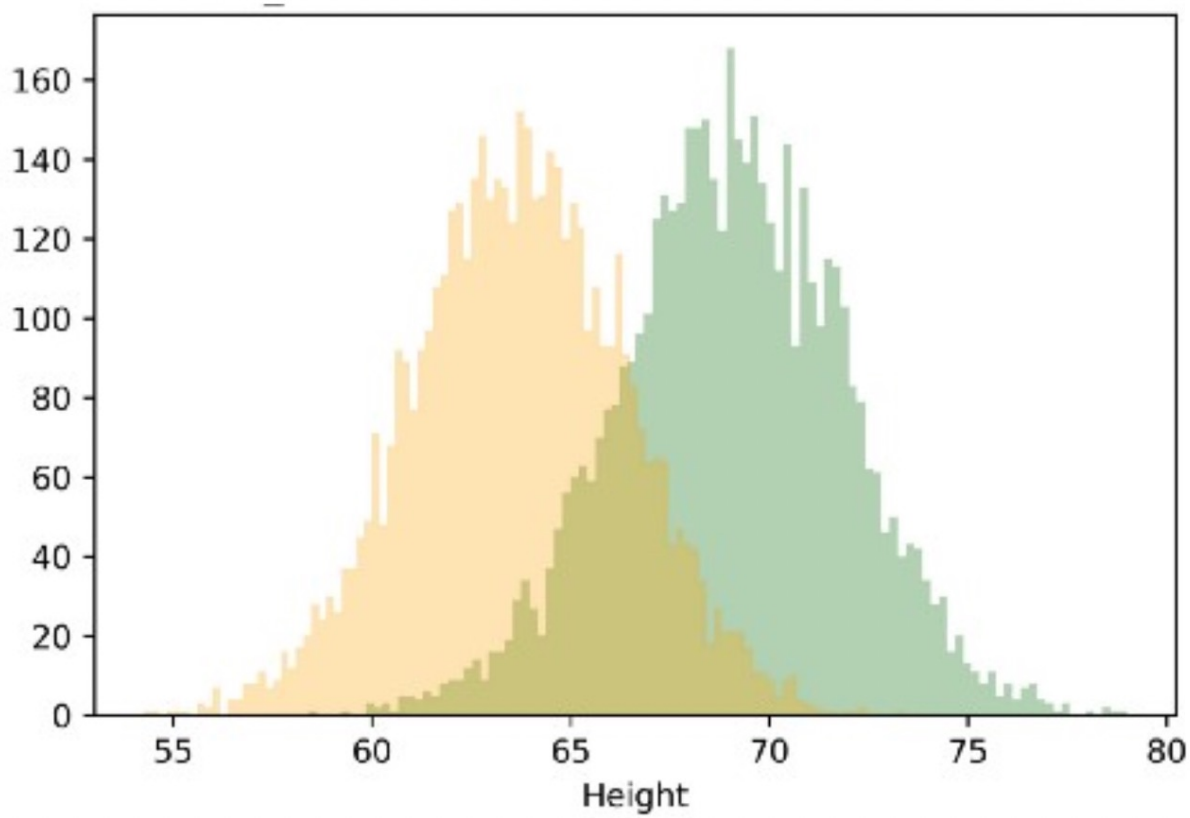
Alerts

- The Alerts tab is used to inform about any issues with each of the columns within data, such as correlation between variables, data skewness, data distribution.

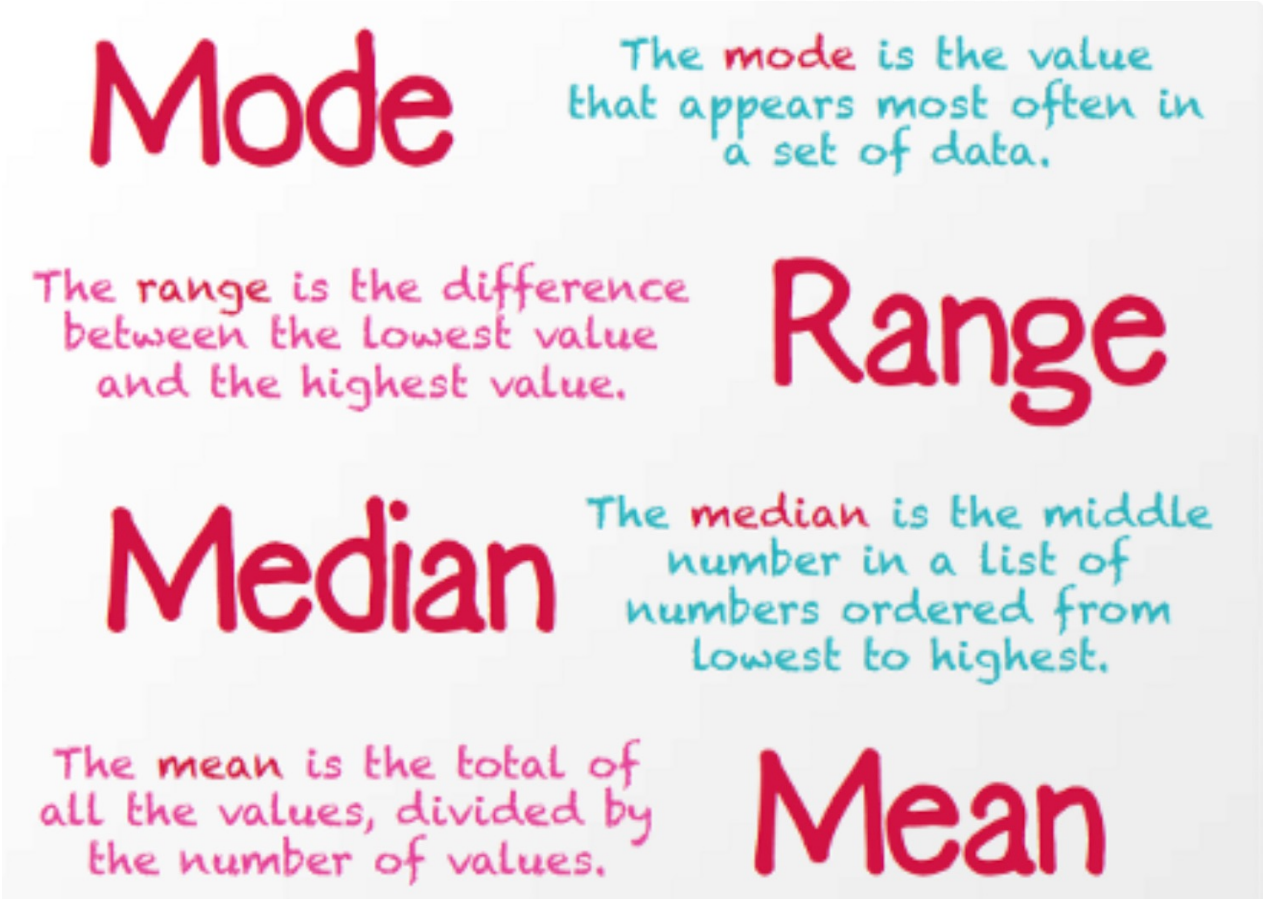


What is skewed data?

- Skewed data is data that creates an uneven curve distribution on a graph.
- We know data is skewed when the statistical distribution's curve appears distorted to the left or right.
- Example: In this graph, green indicates males and yellow indicates females.

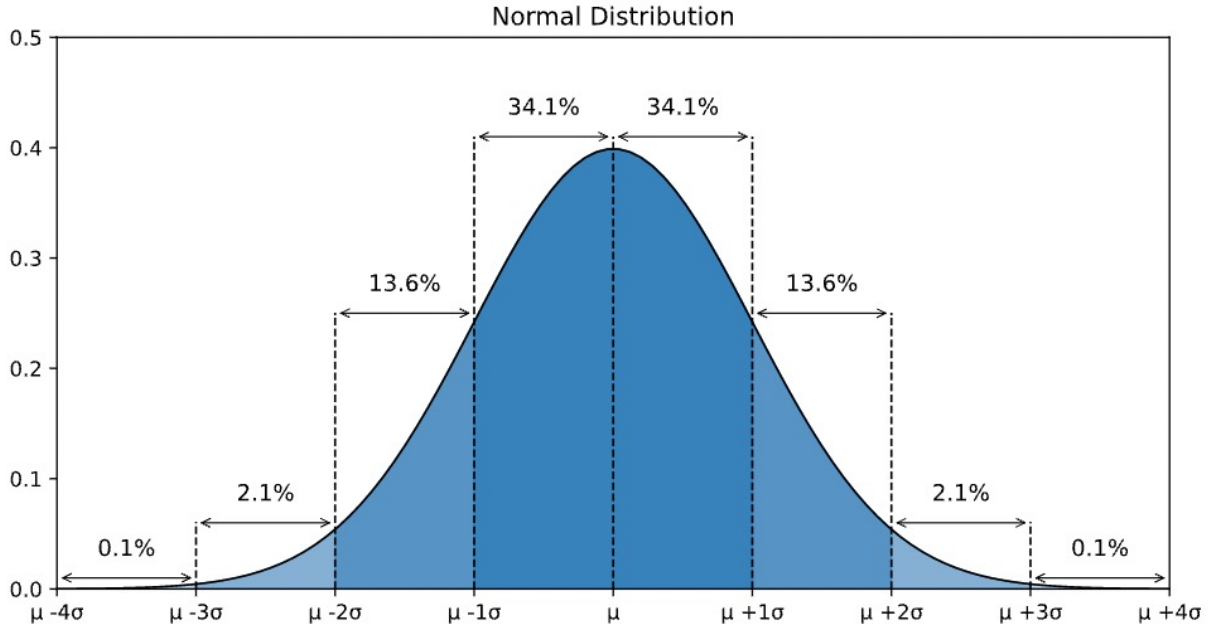


- In the case of normal distribution, the mean, median and mode are close together.
- These three are all measures of the center of data.
- We can determine the skewness of the data by how these quantities relate to one another.



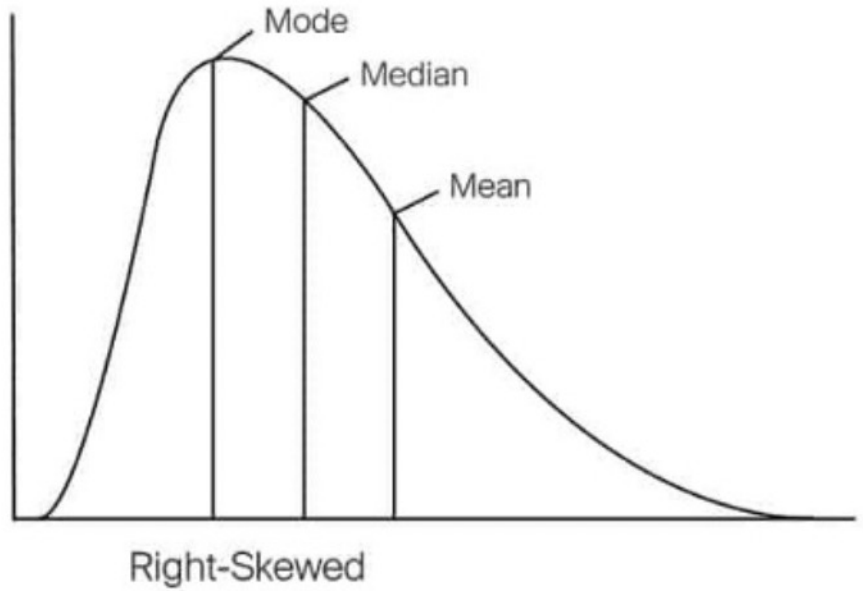
Normal Distribution

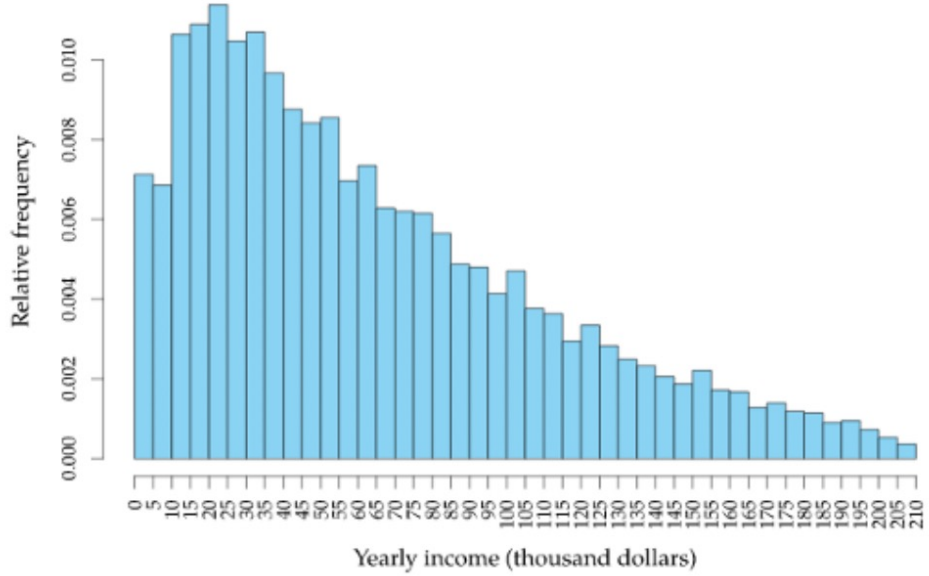
- In a normal distribution, data is **symmetrically distributed with no skew**. When plotted on a graph, the data follows a bell shape, with most values clustering around a central region and tapering off as they go further away from the center.



Right (or Positively) Skewed Data

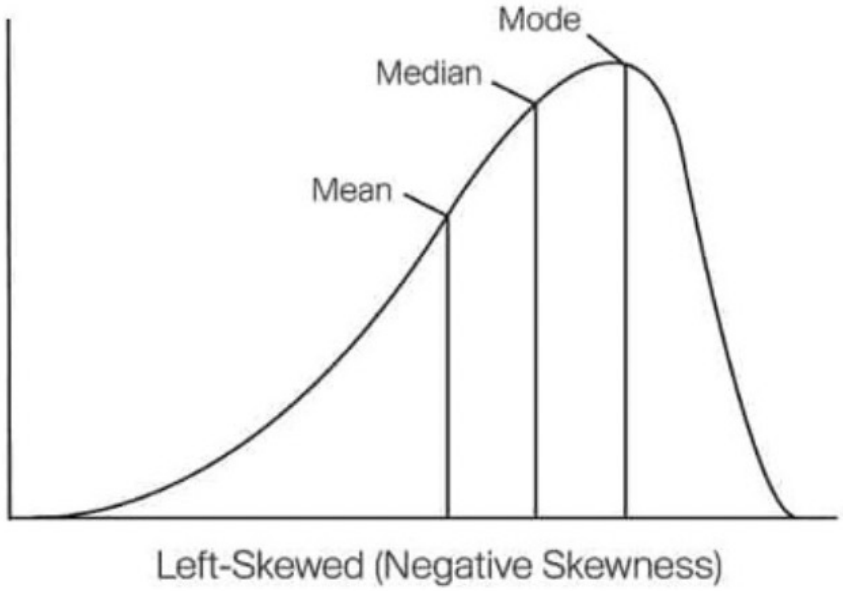
- A right-skewed distribution has a long tail that extends to the right or positive side of the x-axis.

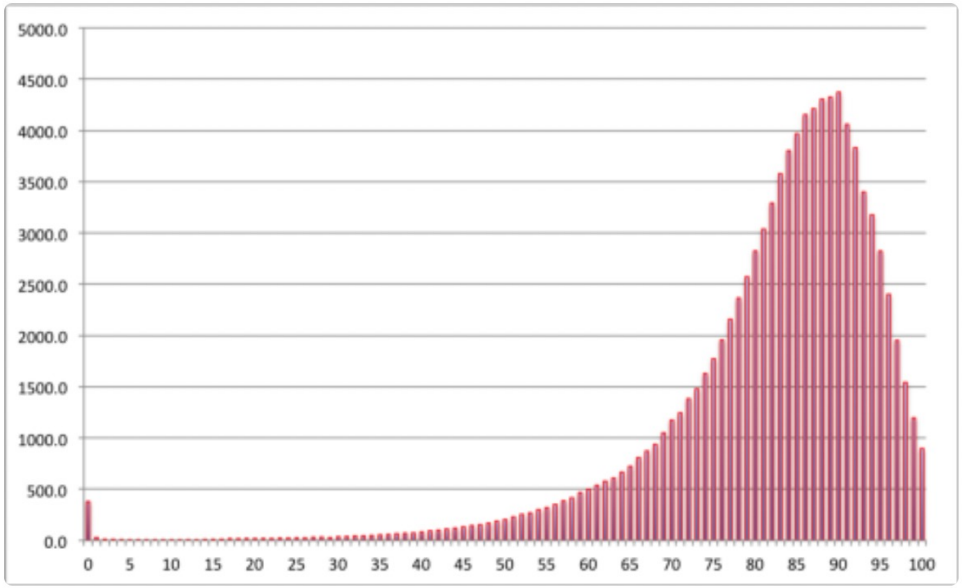




Left (or Negatively) Skewed Data

- A left-skewed distribution has a long tail that extends to the left (or negative) side of the x-axis.





My data is skewed. So what?

If there's too much skewness in the data, then many statistical models don't work effectively.

In skewed data, the tail region may act as an outlier for the statistical model, and outliers surely affect a model's performance, especially regression-based models.

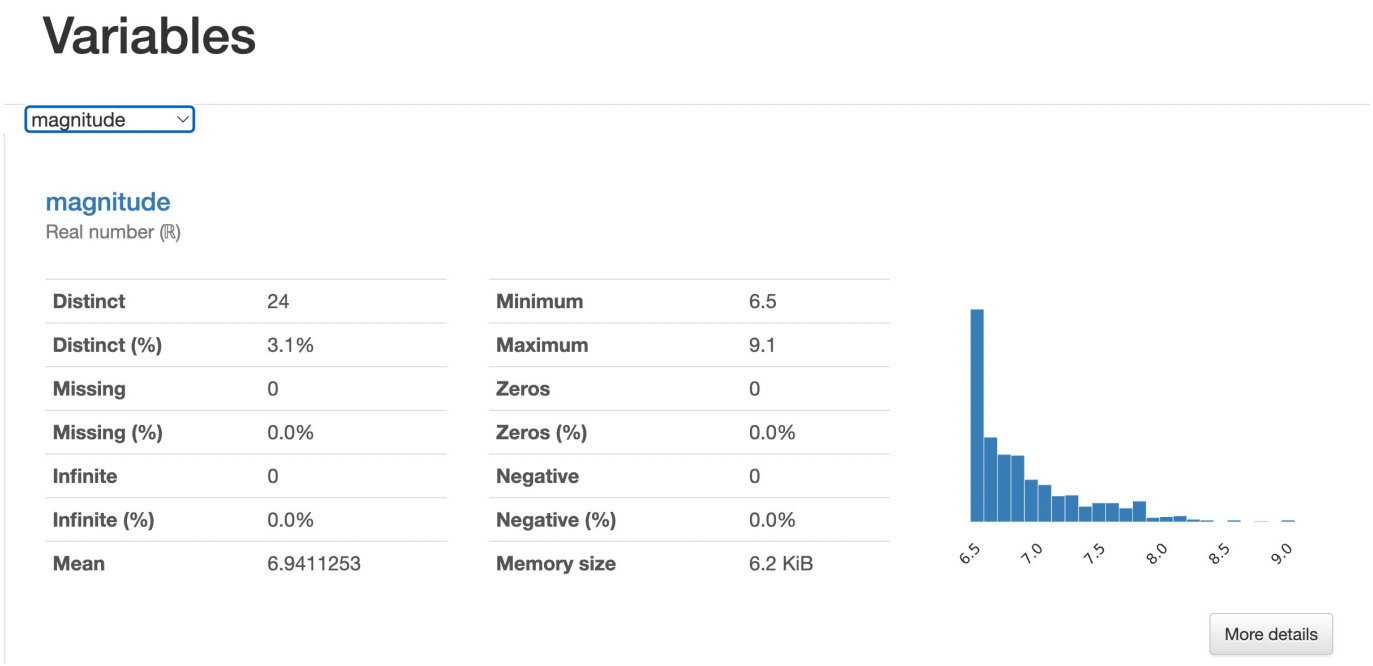
- There are statistical models that are robust enough to handle outliers like tree-based models, but how about your chosen model unable to do that?

So what do you do? You'll need to transform the skewed data so that it becomes a Gaussian (or normal) distribution.

- Removing outliers and normalizing our data will allow us to experiment with more statistical models.

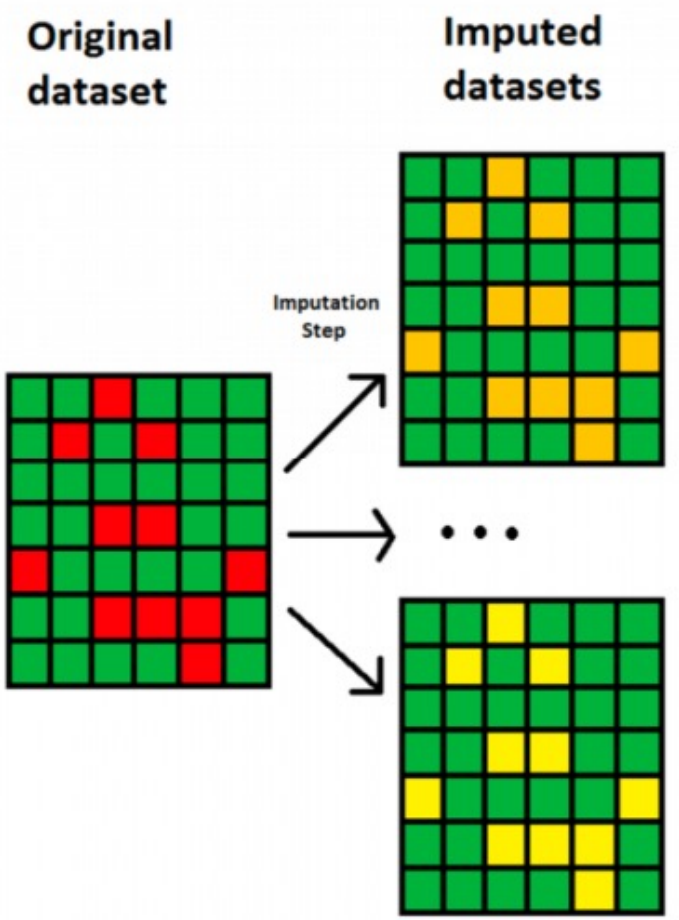
Variable

- Within the variables section of the report we can view the detailed statistics of each of the columns contained within the dataframe. This includes how many missing values there are, the statistics of the data (mean, minimum and maximum), and more.
- On the right hand side of each section, we can see a histogram of the data distribution. This gives us an indication of the skewness of the data, as well as it's spread.



Missing Values

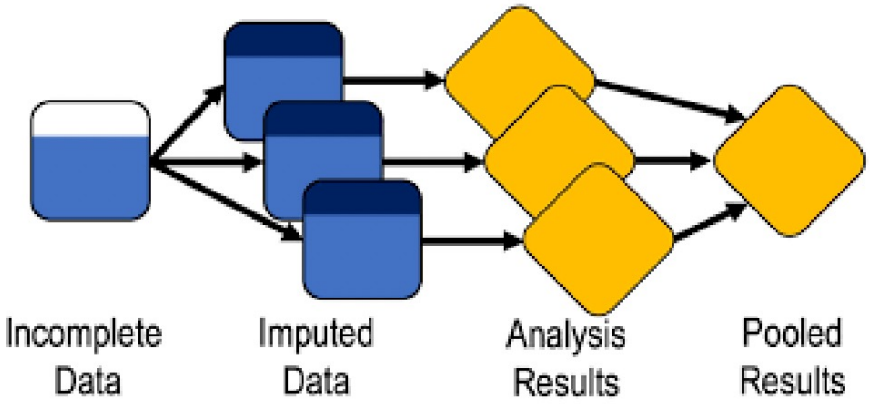
- Many aspects of data will need to be “handled” (cleaned or removed or “fixed”), including: NULL values, missing values, errors, noise or unexpected data artifacts.
- How to handle missing values? → Imputation
  - Replace the missing values (NaN; blank) with a replacement value.



Imputation Stages and Techniques

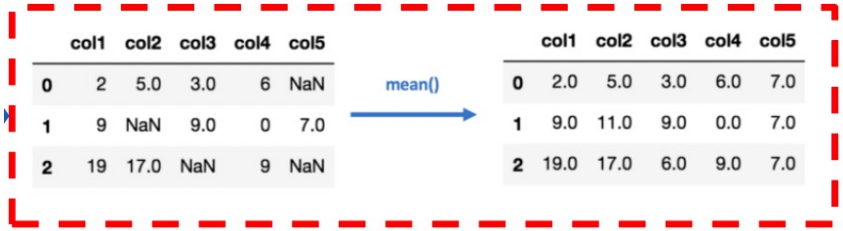
- If the data type is Numeric Variable
  - Imputation of mean or median.
  - Imputation of arbitrary values.
  - Imputation of end of tail value.

- If the data type is Categorical Variable
  - Imputation of frequently occurring categories.
  - Add missing categories



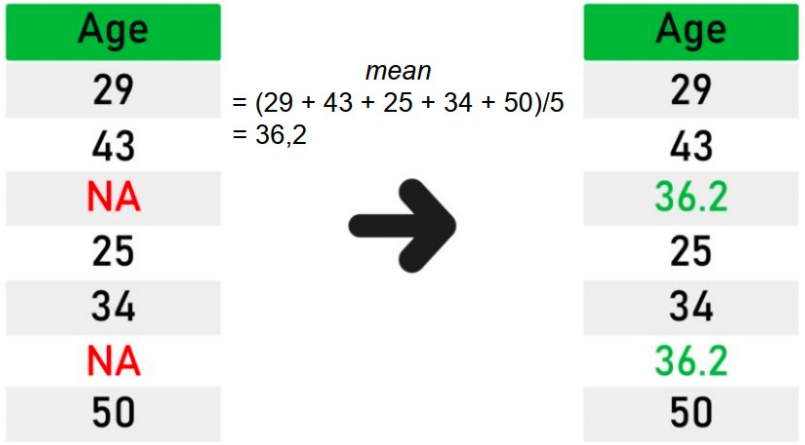
Imputation of Mean or Median

- Pro:
  - Easy and fast.
  - Works well for small numeric datasets.
  - Suitable for numeric variables.
  - Suitable for data missing completely at random (MCAR)
  - Can be used in production (eg in deployment models).
- Cons:
  - Less accurate.
  - Does not take into account probability/uncertainty.
  - Not suitable for >5% missing data.



	col1	col2	col3	col4	col5
0	2	5.0	3.0	6	NaN
1	9	NaN	9.0	0	7.0
2	19	17.0	NaN	9	NaN

	col1	col2	col3	col4	col5
0	2.0	5.0	3.0	6.0	7.0
1	9.0	11.0	9.0	0.0	7.0
2	19.0	17.0	6.0	9.0	7.0



Imputation of Arbitrary Values

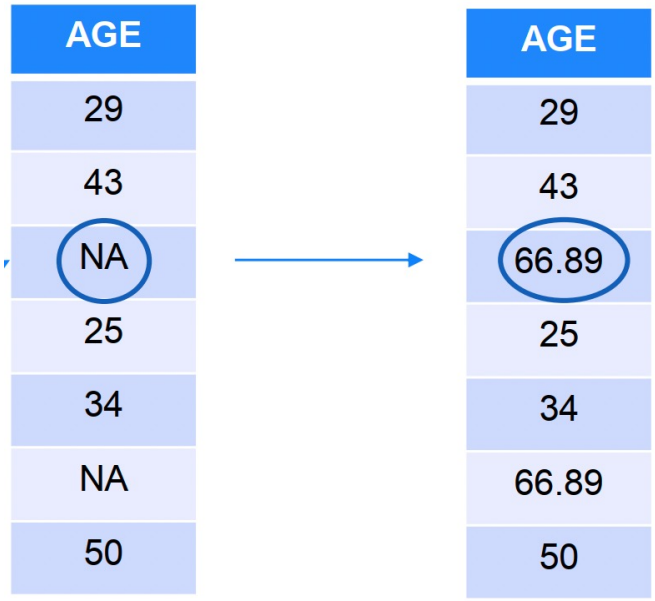
- Pro:
  - Assuming data is not missing at random.
  - Easy and fast to apply to complete datasets.
- Cons:
  - Distort the variance and distribution of the original variables.
  - Forming outliers (if the arbitrary value is at the end of the distribution).
  - The greater the NA, the greater the distortion.
  - Avoid choosing arbitrary values that are close to the mean or median.

Age
29
43
NA
25
34
NA
50

Age
29
43
99
25
34
99
50

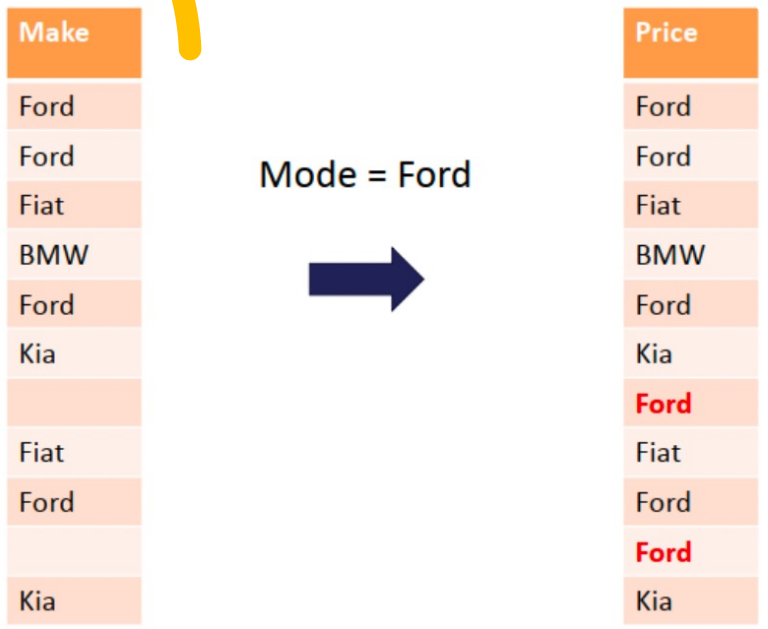
Imputation of End of Tail Value

- Pro:
  - Similar to voluntary imputation.
  - Suitable for numeric variables.
- Special conditions in selecting arbitrary values:
  - If the variables are normally distributed, then arbitrary value = mean + 3 * std.
  - If the variable is skew, then use the IQR proximity rule
    - IQR = 75th Quantile – 25th Quantile.
    - = 75th Quantile + IQR ×3 (Upper Limit).
    - = 25th Quantile - IQR ×3 (Lower Limit).
  - Only used on training data (training set)



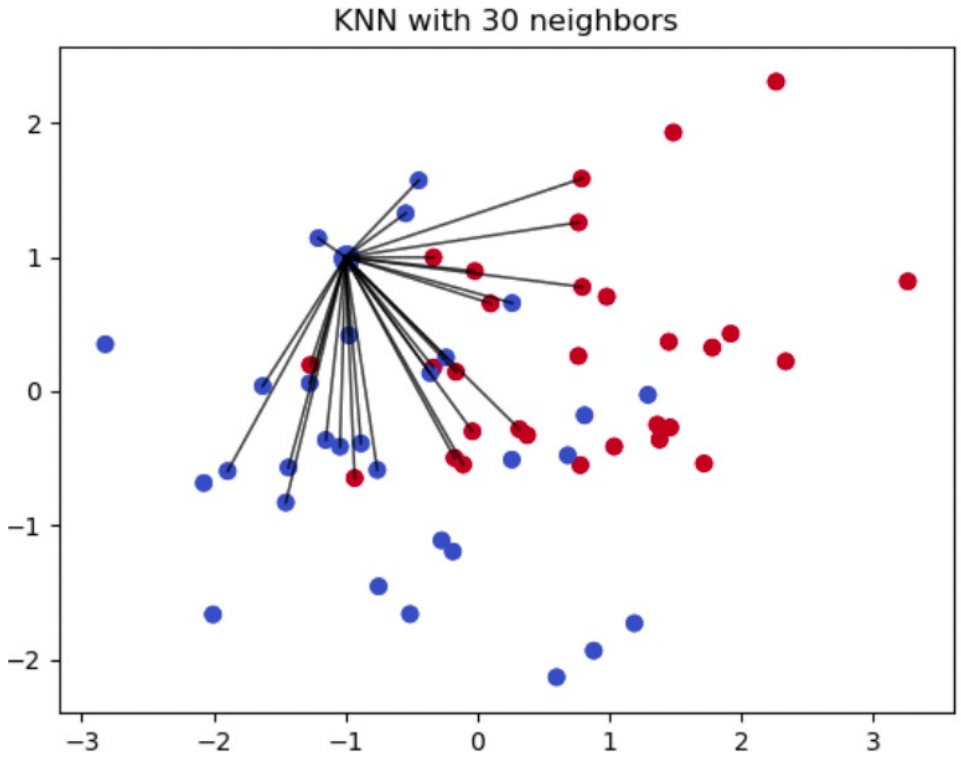
Imputation of Frequent Category (Modus)

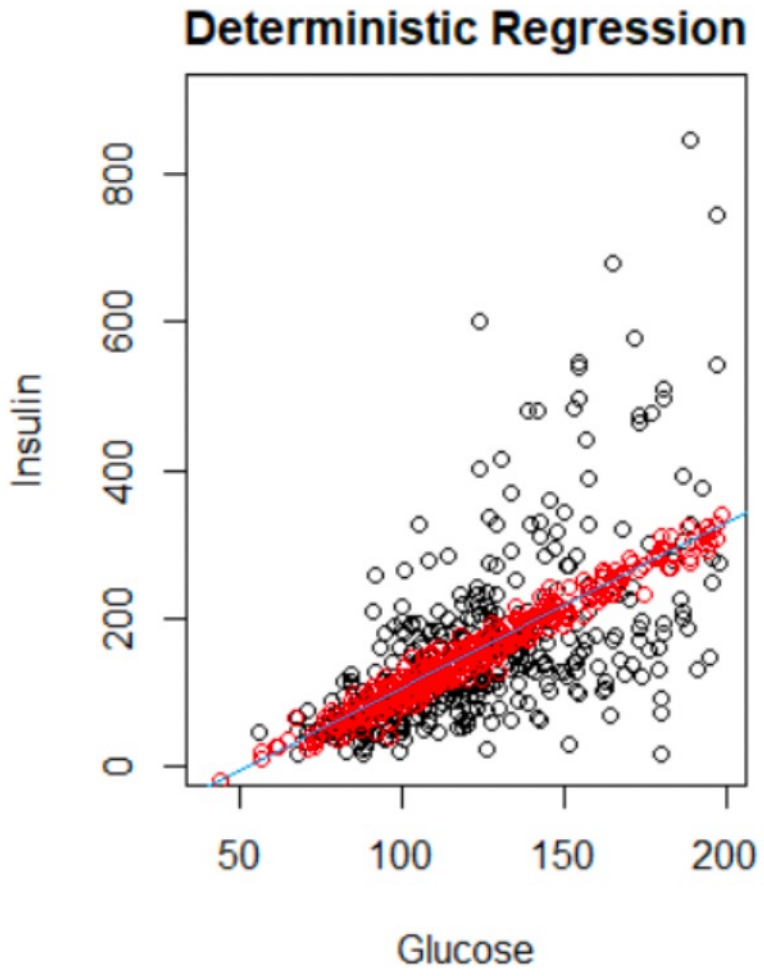
- Pro:
  - Suitable for data with missing at random.
  - Easy and fast to apply.
  - Suitable for skew data.
  - Can be used in production (eg in a deployment model).
- Cons:
  - Distorts the relation of the label with the highest frequency vs other variables.
  - Generates over-representation if a lot of data is missing



Imputation with k-NN

- Pro:
  - More accurate vs mean/median/most frequent.
- Cons:
  - High computational cost (because KNN works by storing the entire training dataset in memory).
  - Sensitive to outliers in the data (unlike SVM).



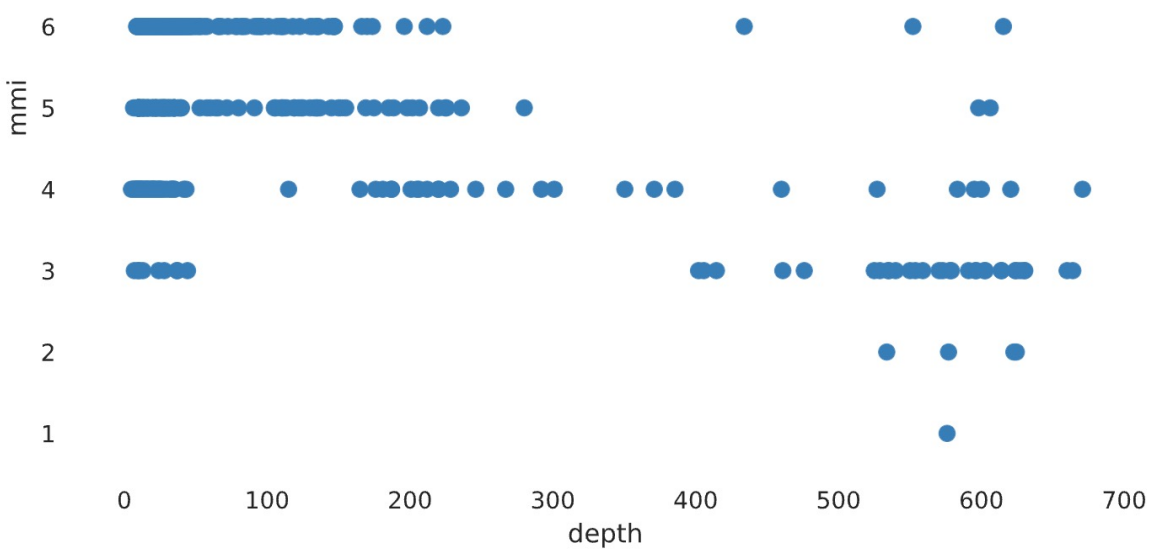


Regression Imputation: Deterministic

- Deterministic
  - Replace the missing values with the correct predictions from the regression model.
  - Does not consider random variations around the slope (regression slope).
  - Imputed values are often too precise and overestimate the X-Y correlation

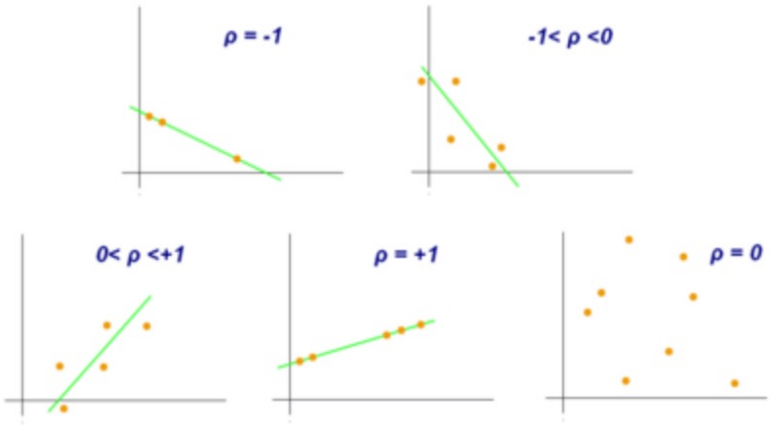
Interactions

- The interactions section of the report allows you to plot one variable against another in order to understand how they relate to each other.



Correlation

- Correlation coefficients are used to measure the strength of the relationship between two variables.
- Pearson correlation is the one most commonly used in statistics. This measures the strength and direction of a linear relationship between two variables.
- Values always range between -1 (strong negative relationship) and +1 (strong positive relationship). Values at or close to zero imply weak or no relationship.
- Correlation coefficient values less than +0.8 or greater than -0.8 are not considered significant.



correlation coefficient

Thank you…

- https://medium.com/@anandsubbu7/data-profiling-2b222a025b5d
- https://medium.com/codex/data-profiling-having-that-first-date-with-your-data-2e05de50fca7
- https://www.projectpro.io/article/8-feature-engineering-techniques-for-machine-learning/423
- https://www.kaggle.com/datasets/warcoder/earthquake-dataset
- https://towardsdatascience.com/pandas-profiling-easy-exploratory-data-analysis-in-python-65d6d0e23650
- https://builtin.com/data-science/skewed-data