Introduction :

In this second session, we will delve into the world of data visualization and the key libraries that empower us to transform raw data into insightful visual representations. By the end of this session, participants will have a comprehensive understanding of the pivotal role data visualization plays in enhancing data exploration, analysis, and ultimately model performance. We will focus on essential Python libraries such as Pandas, Matplotlib, and Seaborn to achieve this goal. Data Visualization: Unveiling the Power Data visualization is the art and science of representing data graphically. It is a crucial tool in the data scientist's arsenal, serving several critical purposes.

General IT informations:

1- A compiled language is converted into machine code so that the processor can execute it, as c,c++,pascal
2- An interpreted language is a language in which the implementations execute instructions directly without earlier compiling a program into machine language. like python ,java,javascool

kernel for python and OS kernel :

Kernels are processes that run independently and interact with JupyterLab. ipykernel provides the IPython kernel for Jupyter, which provides an interactive Python development environment. Kernels in JupyterLab allow the use of different Python versions and virtual environments. kernel(os) :
kernel connects application software to the hardware of a computer A full kernel controls all hardware resources

1. Enhancing Understanding

Data visualization helps us bridge the gap between raw, complex datasets and human comprehension. Through visual forms, we can represent data in a way that's intuitive and easier to grasp, making it more accessible to both technical and non-technical audiences. This visual translation aids in the identification of patterns, trends, and anomalies within the data.

2. Facilitating Data Exploration

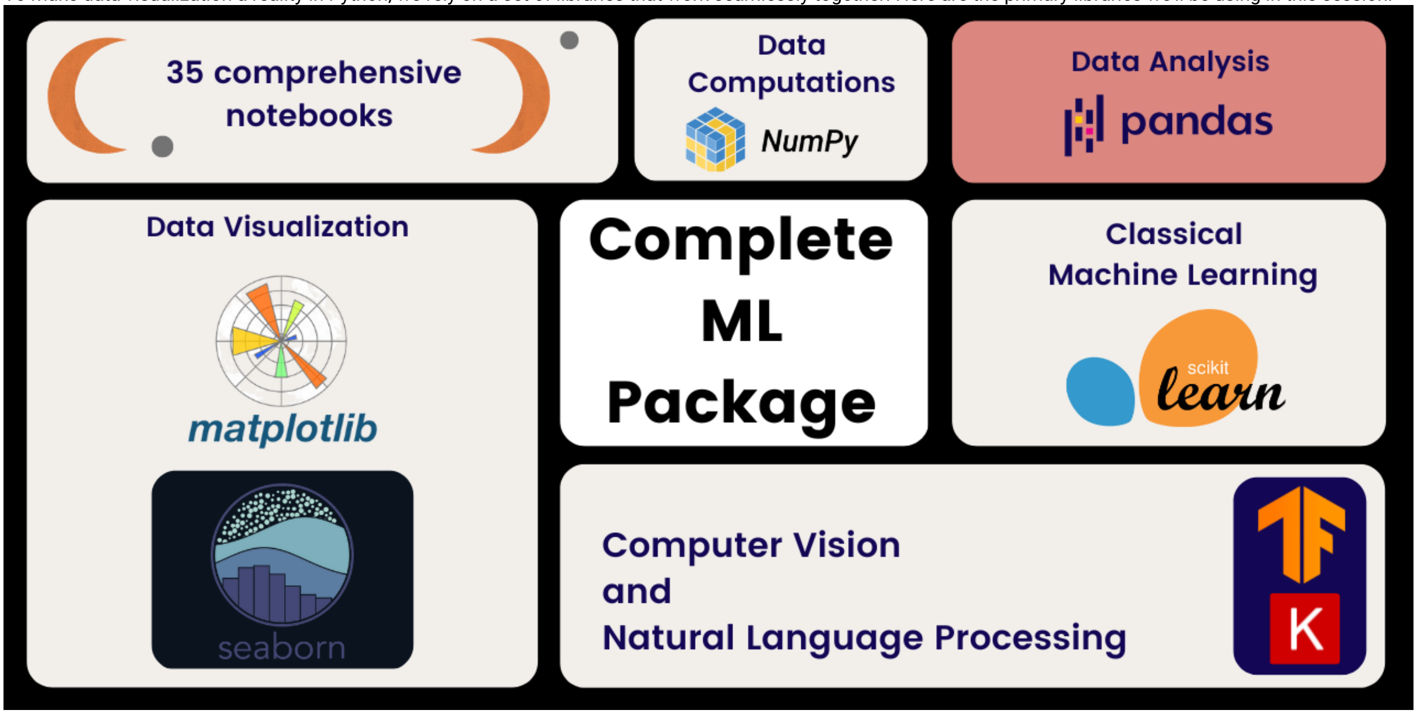
One of the initial steps in any data analysis journey is to get an overview of the dataset. Visualization provides this essential insight. It allows us to quickly grasp the dataset's size, distribution, and structure. Moreover, it can highlight potential issues like missing values or outliers that require attention.

3. Feature Selection

Data visualization empowers us to identify the most relevant features for our model. By visually exploring the relationships between various features and the target variable, we can make informed decisions about which variables to include in our modeling process.

Tools and Libraries

To make data visualization a reality in Python, we rely on a set of libraries that work seamlessly together. Here are the primary libraries we'll be using in this session:



1. Pandas

Pandas is a powerful library for data manipulation and analysis. It provides data structures like DataFrames and Series, which are pivotal for data organization. Participants will learn how to leverage Pandas for data preparation and cleaning before visualization.

2. Matplotlib

Matplotlib is a versatile plotting library in Python. It allows us to create a wide array of static, animated, or interactive visualizations. We'll explore the basics of Matplotlib, including line plots, scatter plots, bar charts, and more, to visualize data effectively.

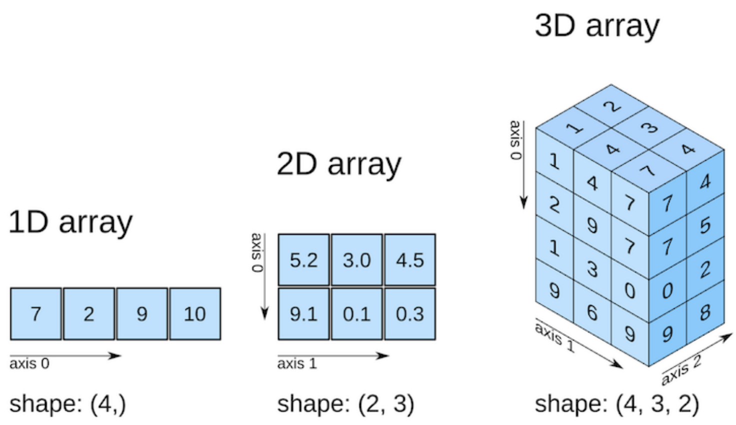
3. Seaborn

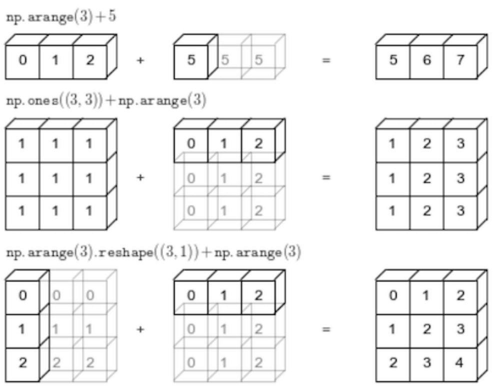
Seaborn is built on top of Matplotlib and provides a high-level interface for creating informative and attractive statistical graphics. It simplifies the creation of complex visualizations and is particularly useful when dealing with statistical data analysis.

General overview :

1 - Data Computation With NumPy

Creating a NumPy Array Selecting Data: Indexing and Slicing An Array Performing Mathematical and other Basic Operations Performing Basic Statistics Manipulating Data




np.arange(3)+5
np.ones((3,3))+np.arange(3)
np.arange(3).reshape((3,1))+np.arange(3)

2 - Data Manipulation with Pandas

Basics of Pandas Series and DataFrames Data Indexing and Selection Dealing with Missing data Basic operations and Functions Aggregation Methods Groupby Merging, Joining and Concatenate Beyond Dataframes: Working with CSV, and Excel Real World Exploratory Data Analysis (EDA)

# Data Visualization : ### Data Visualizations with Matplotlib : To gain more insights or understand the problem you're solving, it is very important to visualize the dataset that you are working with. Matplotlib is powerful visualization tool in Python. It can be used to create 2D and 3D figures. Seaborn that we will see later is built on Matplotlib. This is what you will learn: 1. Basics of Matplotlib 2. Types of Plots 3. Image

Plotting 4. Further Resources ### Data Visualization with Seaborn : Seaborn is a fantastic and easy to use Python Visualization which is built on Matplotlib. For a quick look, check out the gallery page. To be covered: 1. Relational Plots: Scatter plots Line plots 2. Distribution Plots: Plotting Histograms with displot() and histplot() Plotting Bivariate Data with Jointplot() Plotting Many Distribution with pairplot() Plotting Distributions with rugplot() Kernel Density Estimation (KDE) Plot with kdeplot() and displot() Cumulative Distributions 3. Categorical Plots: Categorical estimate plots Categorical distribution plots Categorical scatter plots Plotting multiple categorical plots 4. Regression Plots 5. Multiplots: Facet grids Pair grids Pair plots 6. Matrix Plots: Heat and Cluster Maps Heat maps Cluster maps 7. Style and Color ### Pandas for Data Visualization : Pandas that we used for data analysis and manipulation can also be used to visualize data. And it is so simple. To step back a bit, Matplotlib is the primary visualization library in Python. Both Seaborn and Pandas visualization are built on top of Matplotlib. Contents: 1. Imports and loading datasets 2. Basic Plots 3. More Plots A. Bar plots B. Histogram C. Box plots D. Area plots E. Kernel Density Estimation (KDE) plots F. Scatter plots G. Hexagonal plots H. Pie plots 4. Further learnings Styles and themes Colors # How choose visualization : Choosing the right visualization for your data is crucial for effectively communicating your insights and making data-driven decisions. Here are some steps to help you decide which visualization is most suitable for your data:

Understand Your Data: Before selecting a visualization, you need a deep understanding of your data. Consider the type of data you have, whether it's categorical, numerical, time-series, or something else. Understand the relationships within the data, including trends, patterns, and outliers.

Define Your Objective: Clearly define the goal of your visualization. Are you trying to compare data, show trends over time, identify patterns, or display proportions? Your objective will guide your choice of visualization.

Select the Right Chart Type: Different data types and objectives call for specific chart types. Here are some common types of visualizations and when to use them:

Bar Chart: Use for comparing categories or showing discrete data.
Line Chart: Ideal for displaying trends over time or ordered categories.
Pie Chart: Use to show parts of a whole (proportions).
Scatter Plot: Suitable for visualizing relationships between two numerical variables.
Heatmap: Shows the intensity of relationships between two categorical variables.
Histogram: Great for visualizing the distribution of a single numerical variable.
for more details check this book : Resources/How-to-Choose.pdfhttps://github.com/wissemkarous/Machine_Learning-Documentation-GDSC/blob/5faa0ff20e0b18427b93c050b82749977b0b1dc3/Resources/How-to-Choose.pdf
what's correlation :

$$\rho_{xy} = \frac{\mathrm{Cov}(x, y)}{\sigma_x \sigma_y}$$

Tabular Data:

For Machine Learning we commonly use tabular data and structured data.

This how the tabular data going to look like:

https://github.com/wissemkarous/Machine_Learning-Documentation-GDSC/tree/63819e0ed5b39e5ece7cc4f3925aa6b14edb2a5b/Resources/P-bascis The dataset is separated to features (n columns) and target (1 column).

The features are the characteristics or attributes of the data that the model uses to make predictions about the target variable.

This was for the supervised learning. For the unsupervised learning the data does have a target.

After all this the cycle of our data:

After extracting the target and the features, we going to extract the training and testing data (data split).

Missing Values:

- How do we get missing values: **Missing values can occur for various reasons, including data entry errors, incomplete data collection.**
- How to see missing values with code: **These values are usually denoted as "NaN" (Not-a-Number) in numerical datasets or as "null" in dataset.**
- Impact on Machine Learning Models:

1. **It may lead to incorrect prediction.**
2. **Reduced Model Performance.**
3. **Many ML models cannot handle missing values, so they will return errors.**

Missing values imputation: Missing values can be imputed with a provided constant value, or using the statistics (mean, median or most frequent) of each column in which the missing values are located.

What Outliers are

- An outlier is a data point in a dataset that significantly differs from the majority of the other data points.
- Outliers can be caused by various factors, including errors in data collection, natural variation, or extreme events.
- Boxplots and identifying outliers

1. First quartile (***Q1): 25% of the data values are lower than this point.***
2. Median (***Q2): the middle value of the dataset, so that 50% of the data values are lower than this point.***
3. Third quartile (***Q3): 75% of the data values are lower than this point.***
4. IQR: Interquartile range

Correlation Explanation:

Correlation Formula:

ρ (X, Y) = cov(X, Y) / (σ(X) * σ(Y))

- **ρ (X, Y):** This is the Pearson correlation coefficient between the variables X and Y. It quantifies the strength and direction of the linear relationship between these two variables. The value of ρ can range from -1 to 1.
- **cov(X, Y):** The covariance between X and Y. Covariance is a measure of how two variables change together. A positive covariance indicates that when one variable increases, the other tends to increase as well, and vice versa for negative covariance. However, the magnitude of covariance is not standardized, so it may be challenging to compare across different datasets.
- **σ(X) and σ(Y):** The standard deviations of the variables X and Y, respectively. The standard deviation measures the degree of dispersion or variability in the values of a variable. A high standard deviation means that the values are spread out, while a low standard deviation indicates that the values are close to the mean.

Correlation Helps Selecting Important Features:

Analyzing the correlation between features and the target variable is a critical step in feature selection and model building in machine learning. Features that have a strong positive or negative correlation with the target variable are typically **considered more important for predicting the target** and are more likely **to be included in the model**. features with little to no correlation with the target may be less informative and can potentially be excluded from the model to simplify it.

Normal Distribution:

A normal distribution, is a common way that numbers naturally spread out in many real-world situations. It looks like a symmetrical curve with a peak in the middle and tails on both sides

Why it's useful in Machine Learning:

Predictive Power: In many cases, data that follows a normal distribution is easier to work with. Machine learning models often make predictions based on patterns they find in data. Normal distributions help these models find those patterns more easily.

Assumptions: Some machine learning methods assume that the data they work with is normally distributed. It's like giving the model a hint about how the data behaves. For example, linear regression assumes that the relationship between variables is linear, and the errors follow a normal distribution. This assumption simplifies the math and can make the model work better.