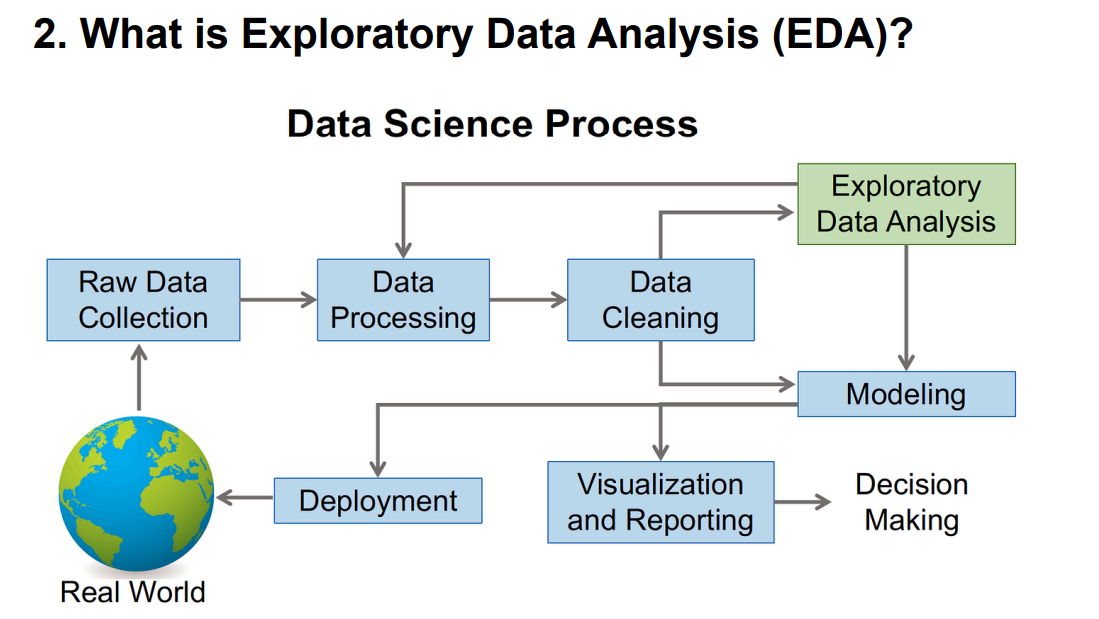
**Exploring Data with NumPy and Pandas:**

This session marks the final module on Python, focusing on Exploratory Data Analysis (EDA). The aim is to introduce new concepts, including NumPy and Pandas, to provide foundational understanding for future detailed modules on EDA. The goal is for participants to understand how things work so they have a basic understanding when EDA is taught in depth later.

**Introduction**

**Focus**: This session is dedicated to Python, specifically focusing on exploratory data analysis using NumPy and Pandas.



Definition: **EDA** stands for Exploratory Data Analysis. In simple terms, **EDA** is about understanding a dataset in detail.

**Purpose of EDA**:

**Data Exploration**: Understanding the dataset before diving into formal analysis or modelling.

**Visualisation**: Creating charts, graphs, and dashboards to summarize and present data.

**Identifying Patterns and Anomalies**: Spotting interesting trends, patterns, and any unusual data points (outliers).

**Testing Initial Hypotheses:** Checking assumptions about the dataset through visualisation and basic statistics.

**Dataset Definition**: A dataset refers to a collection of data points, typically in forms like Excel, CSV, or JSON files, coming from various domains such as sales, HR data, or employee records.

**Real-World Problems in EDA:** Example: Netflix/YouTube Recommendations: These platforms suggest content based on user patterns and watch history, leveraging a large dataset of user actions, which is analysed using machine learning models.

**EDA vs. Data Science**:

**EDA**: A preliminary step focusing on the high-level analysis of data, using statistical methods and visual tools without employing machine learning models.

**Data Science**: A broader discipline that involves building machine learning models and applying algorithms to draw conclusions from data.

**Basic Statistics in EDA:**

**Count**: The number of data points in a dataset.

**Mean**: The average value of a specific numerical column.

**Median**: The middle value in a sorted list.

**Standard Deviation**: A measure of the spread or dispersion of the data.

**Minimum and Maximum**: The smallest and largest values, respectively.

**Core Idea of EDA**: EDA is essentially about summarizing and understanding the data at a high level to gain insights before applying more sophisticated methods like modelling.

A chart with green and black text

AI-generated content may be incorrect.

A diagram of data analysis

AI-generated content may be incorrect.

**Pandas**: The backbone of data analysis in Python. It is an essential library for data manipulation and analysis.

**Functions:**

**Data Loading:** Use pd.read\_csv('file.csv') or pd.read\_excel('file.xlsx') to load data. Data Cleaning: Handles missing values, converts data types, and processes data.

**Data Manipulation:** Slicing, filtering, and aggregating data.

**Descriptive Statistics:** Functions like df.mean(), df.median(), df.describe() to generate summary statistics.

**Matplotlib**: A core plotting library in Python that helps in creating various charts such as line charts, bar charts, pie charts, and histograms.

**Seaborn**:

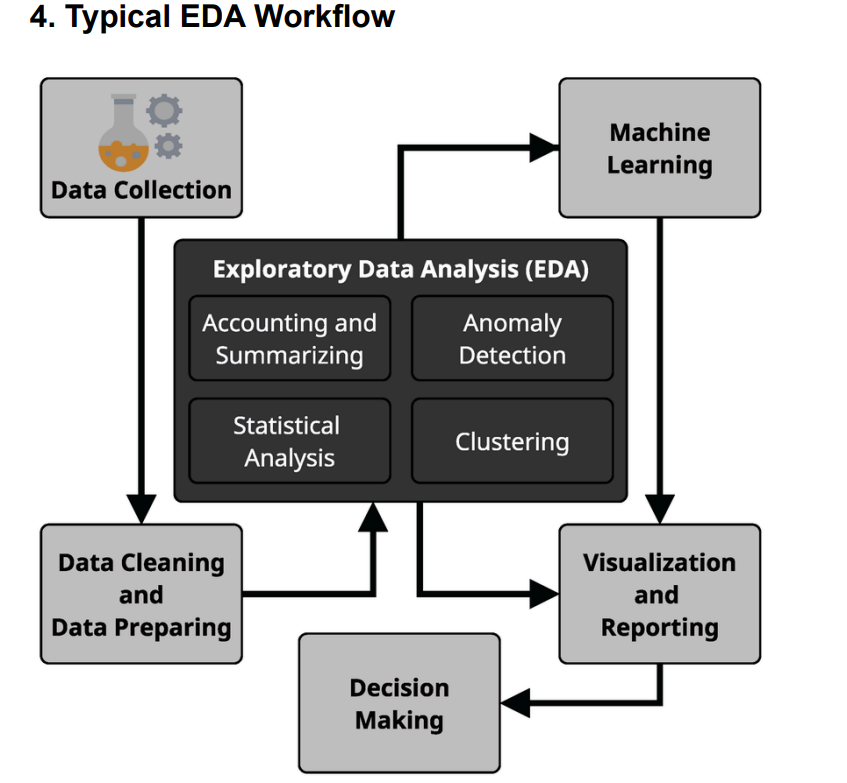
A data visualization library built on top of Matplotlib that simplifies the creation of aesthetically pleasing plots. It also integrates well with Pandas for quick data visualization.

**NumPy**:

Although primarily used for numerical computing and scientific applications, NumPy is useful for data analysis as it handles arrays, matrix operations, and more. While not used directly in the current EDA session, it's often required for data manipulation.

A diagram of a diagram

AI-generated content may be incorrect.



**Starting With A New Dataset**

1. **Preview data** Use df.head() in Pandas to look at a few rows.
2. **Understand Data Types** Check for numbers, dates, text columns.
3. **Check for Missing or Odd Values**

**📀 Basic Statistics**

* **Count** – How many data points?
* **Mean** – Average value (e.g., average age of users)
* **Median** – Middle value when sorted (better for skewed data)
* **Standard Deviation** – Spread of values from the mean
* **Min/Max** – Range of values (spot outliers)

Use df.describe() to get all of this at once.

**Basic Statistics Example**

data = [1, 2, 2, 3, 14]

mean = sum(data)/len(data) # 4.4

* **Mean** = 4.4 seconds (pulled up by the outlier 14)
* **Median** = 2 seconds (better "typical" value)
* **Standard Deviation** = Larger due to outlier

💡 **Use median for skewed data** like income, web traffic, etc.

**🧮 Understanding Distributions**

* Is the data **normal** (bell-shaped)?
* Is it **skewed** (left or right)?
* Are there **multiple peaks** (bi-modal)?

Use visual tools:

* **Histograms**
* **Box plots**

**Histogram** groups values into bins and shows how frequent each bin is. E.g., most website sessions are under 200 seconds, but there’s a long tail of longer ones → right-skewed.

**Key Python Libraries for EDA**

**1. Pandas**

* Load data: pd.read\_csv('data.csv')
* Clean data: handle missing values, convert types
* Compute stats: df.mean(), df['column'].median()
* Slice and dice: filter and aggregate

Pandas is the backbone of data analysis in Python!

**2. Matplotlib**

* Core plotting library in Python
* Create line plots, bar charts, pie charts, histograms
* Import as: import matplotlib.pyplot as plt

It may not be pretty by default, but it's powerful and flexible!

**3. Seaborn**

* Built on top of Matplotlib
* Easier to make **beautiful** plots
* Integrates well with Pandas

Examples:

* Correlation heatmaps
* Box plots
* Violin plots

Seaborn provides visually appealing defaults and statistical overlays (like confidence intervals).

**Typical EDA Workflow**

1. **Load Data**

df = pd.read\_csv('data.csv')

1. **Understand Structure**

df.shape

df.columns

df.dtypes

1. **Clean Data**

df.dropna()

# or

df.fillna(value)

df['date'] = pd.to\_datetime(df['date'])

1. **Compute Statistics**

df.describe()

df['sales'].mean()

1. **Visualize Distributions**

import seaborn as sns

sns.histplot(df['sales'])

sns.boxplot(y=df['sales'])

1. **Explore Relationships**

sns.scatterplot(x='ad\_spend', y='sales', data=df)

1. **Iterate** Ask new questions, explore deeper, create new features.

EDA is a **playful, iterative** process. It's like detective work with data!