🔍 Exploratory Data Analysis (EDA)

Data Science and Machine Learning PART 1

🎯 Learning Objectives

By the end of this lecture, you'll understand:

1. What **EDA** is and **why it's essential** before building models.
2. The **main steps** in performing EDA.
3. How to **summarize**, **visualize**, and **interpret** data using Python.
4. How to identify **patterns, trends, and relationships** in data.
5. How to make data-driven decisions from your findings.

🌟 1. What Is Exploratory Data Analysis?

🧠 Simple Definition:

Exploratory Data Analysis (EDA) is the process of **examining your data before doing anything else** — before modeling, before predictions, before drawing conclusions.

It helps you understand:

- What the data looks like
- What's missing
- What's unusual (outliers)
- What patterns or relationships exist

🧩 **Analogy:**

Think of EDA as being a **detective investigating a mystery**.
Your dataset is the **crime scene**.
Before jumping to conclusions, you first:

- Look for **clues** (patterns)
- Identify **inconsistencies** (missing data, errors)
- Observe **connections** between variables

Only then can you solve the mystery, or in our case, **build a good machine learning model**.

🧮 2. Why EDA Is Important

Let's say you're working on a project to predict **student exam scores** based on their **study hours**, **sleep**, and **attendance**.

Before trusting the data, you need to check:

- Are there missing or wrong entries (e.g., "sleep = -5 hours")?
- Are there relationships (do more study hours = higher scores)?
- Are there any patterns worth noting?

If you skip this step, your model might "learn" from **bad data**, leading to **poor or misleading predictions**.

In short:

EDA helps you clean, understand, and visualize your data, it's the foundation of every good data project.

🧱 3. Steps in EDA

Here are the main steps we'll explore:

1. Understanding your dataset
2. Summarizing data (numerical and categorical)
3. Handling missing or wrong values
4. Finding patterns and correlations
5. Visualizing your findings

⚙️ 4. Step 1: Understanding Your Dataset

Let's start with a sample dataset, for example, a **Student Performance Dataset**.

Name	Study Hours	Sleep Hours	Attendance (%)	Score
Alice	5	8	90	85
Bob	3	6	70	60
Charlie	8	7	95	92
Diana	2	5	65	50
Eve	4	7	80	75

🧰 Import the Libraries

```
import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns

data = pd.read_csv("students.csv")  # Example CSV file
```

🕵️ Peek at the Data

```
print(data.head())
```

Shows the first 5 rows, like looking at the first page of a spreadsheet.

📏 Check Dimensions and Structure

```
print(data.shape)  # (rows, columns)

print(data.info()) # Data types and missing values
```

Output:

```
(100, 5)

<class 'pandas.core.frame.DataFrame'>

Columns: 5 entries
```

🧠 **Analogy:**

Think of this step like **checking your pantry** before cooking, you're figuring out how many ingredients (columns) you have and if any are missing.

🧾 5. Step 2: Descriptive Statistics

Let's explore the **basic statistics** that summarize your data.

```
print(data.describe())
```

Output:

	Study Hours	**Sleep Hours**	**Attendance**	**Score**
count	100	100	100	100
mean	5.2	7.0	85.4	78.5
std	2.1	1.2	10.5	12.4
min	1.0	4.5	60.0	45.0
max	9.0	9.0	100.0	98.0

🧠 What Does This Tell Us?

- **Mean (average)**: Students study about 5.2 hours and sleep 7 hours on average.
- **Standard deviation**: There's some variation, not everyone studies or sleeps the same amount.
- **Minimum/Maximum**: One student studied only 1 hour (ouch!) and another 9 hours.

This gives you a quick overview of your data's **range and spread**.

🔹 For Categorical Data:

If you had columns like *Gender* or *Grade*, you'd use:

```
print(data['Gender'].value_counts())
```

Output:

Male: 60

Female: 40

🧠 **Analogy:**
Descriptive statistics are like a **quick health check** for your data, you're taking its pulse before doing deeper analysis.

🧩 6. Step 3: Handling Missing and Incorrect Data

Even in EDA, you'll encounter incomplete or suspicious values.

```
print(data.isnull().sum())
```

If you see:

Study Hours   2

Sleep Hours   1

Score   0

You know where the problems are.

You can fix them:

```
data['Study Hours'].fillna(data['Study Hours'].mean(), inplace=True)
```

Or remove them:

```
data.dropna(inplace=True)
```

🚨 Check for Outliers

Outliers are extreme values that don't fit the pattern (e.g., someone studied 30 hours a day, impossible!).

We can spot them visually:

```
sns.boxplot(data['Study Hours'])
plt.show()
```

🧠 **Analogy:**
If everyone brings normal-sized apples to a fruit market and one person brings a watermelon, that's your **outlier**.

📊 7. Step 4: Finding Patterns and Relationships

EDA is all about **asking smart questions** and using the data to find answers.

🤔 Example Questions

- Do more study hours lead to higher scores?

- Does attendance affect performance?
- Is there an optimal amount of sleep?

Let’s find out on our next lecture.