|| Jai Sri Gurudev |

Sri AdichunchanagiriShikshana Trust (R)

SJB INSTITUTE OF TECHNOLOGY

SJBIT

Study Material

Subject Name: Exploratory Data Analytics

Subject Code: 23CSE422

Semester: IV

By

Faculties Name:

Dr. Prakruthi M K,
Asst. Professor,
Dept. of CSE,
SJBIT.

Mrs. Shilpashree S,
Asst. Professor,
Dept. of CSE,
SJBIT.

Mrs. Vinutha K,
Asst. Professor,
Dept. of CSE,
SJBIT.

Department of Computer Science & Engineering

Aca. Year: Even Sem / 2024-25

Module 1
Introduction to EDA
Exploratory Data Analysis Fundamentals

Key Phases of Data Analysis:

1. **Data Requirements:**
    - Data requirements refer to understanding the types of data needed for analysis. This could include numerical or categorical data, or specific data like sensor data (e.g., heart rate, electro-dermal activity).
    - Example: For an application tracking sleep patterns of dementia patients, data on sleep, heart rate, and user activity are crucial.
2. **Data Collection:**
    - Data is collected from various sources using different methods such as sensors, surveys, databases, or web scraping.
    - Data needs to be stored in the correct format and transferred to the appropriate personnel.
3. **Data Processing:**
    - Preprocessing the collected data to make it ready for analysis. This may involve structuring the data, exporting it into correct tables, and ensuring it's in the right format.
4. **Data Cleaning:**
    - The data cleaning phase involves identifying and correcting errors in the dataset. This includes checking for duplicates, handling missing values, and identifying any inconsistencies.
    - Techniques like outlier detection can be applied to clean quantitative data.
5. **Exploratory Data Analysis (EDA):**
    - EDA is the process of exploring the data to uncover patterns, anomalies, relationships, and insights. It involves using techniques like descriptive statistics (mean, median, standard deviation) and visualizations (histograms, boxplots).
    - EDA helps to understand the data distribution, identify potential issues, and guide the modeling process.
6. **Modeling and Algorithms:**
    - In this phase, mathematical models or algorithms are used to identify relationships between variables. Models like regression, classification, or clustering help explain the underlying structure of the data.
    - Example: A regression model might predict sales based on the price of products and quantity sold.
7. **Data Product:**
    - The output of data analysis often leads to a data product, such as a recommendation system or predictive model, which can be used for decision-making in business or other applications.

    - Example: A recommendation system that suggests products based on user purchase history.
8. **Communication:**
    - Communicating the results of the data analysis to stakeholders is crucial. This often involves data visualization tools (charts, graphs, tables) to convey the findings effectively.
    - Example: Business stakeholders might receive a dashboard showing trends in customer behavior.

The Significance of EDA

- **Exploratory Data Analysis (EDA):**
  EDA is the process of analyzing data sets to summarize their main characteristics, often with visual methods. It is a critical initial step in any data mining project and plays a key role in making sense of large datasets.
- **Data Mining:**
  Data mining is a process used to analyze large datasets and extract useful information or patterns. EDA is the first step in this process, allowing data scientists to visualize and understand the data before further analysis.
- **Ground Truth:**
  EDA helps to uncover the "ground truth" or the real insights that the data holds, without making any underlying assumptions. This process allows data scientists to form hypotheses and decide on appropriate models for further analysis.
- **Role of EDA in Data Mining:**
    - Visualizes data to understand patterns and relationships.
    - Helps generate hypotheses for further analysis.
    - Provides insights that guide the next steps in a data mining project.
- **Components of EDA:**
    - **Summarizing Data:**
      Using tools like Pandas to create a summary of data, which may include measures such as mean, median, and standard deviation.
    - **Statistical Analysis:**
      Conducted using tools like SciPy to compute statistical measures and test hypotheses.
    - **Data Visualization:**
      Tools like Matplotlib and Plotly are used to create visual representations of data, such as histograms, scatter plots, and box plots, which help in identifying trends and outliers.

Examples:

- EDA is applied in various fields such as economics, engineering, and marketing. For example, in marketing, EDA can be used to visualize customer purchasing behavior and segment the customer base.

Tools Used for EDA in Python:

- **Pandas:** Used for summarizing data.
- **SciPy:** Provides statistical tools for data analysis.
- **Matplotlib and Plotly:** These libraries are used to create plots and visualizations to better understand the data.

Steps in EDA:

The process of Exploratory Data Analysis (EDA) involves four main steps that guide the data analysis from problem definition to presenting results. These steps are critical for structuring the analysis, preparing the data, and interpreting the results effectively.

1. Problem Definition:

- **Importance:** The first step in any data analysis is defining the business problem to be solved. The problem definition serves as the foundation and guides the entire analysis process.
- **Key Tasks:**
    - Defining the Main Objective: Clarifying the problem to be solved with the data.
    - Defining Deliverables: Identifying what outputs or results will be expected from the analysis.
    - Outlining Roles and Responsibilities: Assigning tasks to different team members or stakeholders.
    - Obtaining Current Data Status: Assessing the current state of the data before analysis begins.
    - Defining the Timetable: Setting a timeline for completing the analysis.
    - Cost/Benefit Analysis: Evaluating the financial and resource implications of conducting the analysis.

2. Data Preparation:

- **Importance:** Data preparation is critical to ensure that the dataset is clean, well-structured, and ready for analysis.
- **Key Tasks:**
    - Defining Data Sources: Identifying where the data will come from.
    - Defining Data Schemas and Tables: Structuring the data and setting up the necessary tables.
    - Understanding Data Characteristics: Analyzing the data for its key features.
    - Data Cleaning: Handling missing or erroneous data points.

    - Removing Non-Relevant Datasets: Eliminating data that is not useful for the analysis.
    - Data Transformation: Converting the data into a format that can be used for analysis.
    - Dividing Data into Chunks: Breaking the data into smaller, manageable parts.

3. Data Analysis:

- **Importance:** This step involves applying statistical techniques to the dataset to derive meaningful insights.
- **Key Tasks:**
    - Summarizing Data: Using summary statistics (mean, median, mode, etc.) to describe the data.
    - Finding Correlations and Relationships: Identifying any connections between variables.
    - Developing Predictive Models: Creating models to predict future outcomes or trends.
    - Evaluating Models: Assessing the performance and accuracy of predictive models.
    - Calculating Accuracies: Using various metrics (e.g., precision, recall) to evaluate model accuracy.
- **Techniques Used:**
    - Summary Tables, Graphs
    - Descriptive and Inferential Statistics
    - Correlation Statistics
    - Mathematical Models

4. Development and Representation of Results:

- **Importance:** After analysis, it is crucial to represent the findings in a clear and interpretable format for stakeholders.
- **Key Tasks:**
    - Graphical Representation: Presenting results using charts and plots.
    - Summary Tables and Maps: Using tables to summarize key findings.
    - Communicating Results Effectively: Ensuring that business stakeholders can easily understand the results of the analysis.
- **Graphical Analysis Techniques:**
    - Scatter Plots
    - Character Plots
    - Histograms
    - Box Plots
    - Residual Plots
    - Mean Plots
    - Others

Making Sense of Data

Types of Data in Data Analysis

1. **Data:** A dataset is a collection of observations about a particular object or entity. Each dataset contains variables that describe characteristics of the object.
2. **Example Dataset:** A dataset about patients in a hospital can include variables like:
    - Patient ID: A unique identifier for each patient.
    - Name, Address, Date of Birth (DOB), Email, Weight, Gender: Features that describe the patient.
3. **Example:**
    - PATIENT_ID = 1001
    - Name = Yoshmi Mukhiya
    - Address = Mannsverk 61, 5094, Bergen, Norway
    - Date of Birth = 10th July 2018
    - Email = yoshmimukhiya@gmail.com
    - Weight = 10
    - Gender = Female
4. **Data Storage:** Most datasets are stored in tables in databases (schema).
   Example Table: A patient table might look like this:

PATIENT_ID	NAME	ADDRESS	DOB	EMAIL	GENDER	WEIGHT
001	Suresh Kumar	Mannsverk, 61	30.12.1989	skmu@hvl.no	Male	68
002	Yoshmi Mukhiya	Mannsverk, 61, 5094	10.07.2018	yoshmimukhiya@gmail.com	Female	10
003	Anju Mukhiya	Mannsverk, 61, 5094	10.12.1997	anjumukhiya@gmail.com	Female	24
004	Asha Gaire	Butwal, Nepal	30.11.1990	aasha.gaire@gmail.com	Female	23

005	Ola Nordmann	Danmark, Sweden	12.12.1789	ola@gmail.com	Male	75

Types of Data

1. **Numerical Data:**
    - **Definition:** Data involving measurements such as age, weight, or temperature. Often referred to as quantitative data.
    - **Subtypes:**
        - **Discrete Data:** Countable data with a fixed number of distinct values.
            - **Example:** The number of students in a class, the number of heads in 200 coin flips.
        - **Continuous Data:** Data that can take any value within a range.
            - **Example:** Weight, temperature, and height.
2. **Discrete Data Example:**
    - The "Country" variable can have values like Nepal, India, Norway, etc., and it is a fixed, countable set of values.
3. **Continuous Data Example:**
    - A temperature reading is continuous because it can have an infinite number of possible values between two points.

2. **Categorical Data:**
    - **Definition:** Data that describes characteristics of an object, such as gender, marital status, or movie genres. Often referred to as qualitative data.
    - **Types of Categorical Data:**
        - **Binary (Dichotomous):** Only two possible values (e.g., success or failure).
        - **Polytomous:** More than two values (e.g., marital status can be married, divorced, widowed, etc.).
3. **Examples of Categorical Data:**
    - **Gender:** Male, Female, Other, Unknown.
    - **Marital Status:** Annulled, Divorced, Legally Separated, Married, etc.
    - **Movie Genres:** Action, Comedy, Drama, Fantasy, etc.
4. **Categories can be:**
    - **Nominal:** No specific order (e.g., gender, blood type).
    - **Ordinal:** Has a specific order (e.g., rank, education level).

Measurement Scale

Nominal Scale

- **Definition:** The nominal scale is used for labeling variables without any quantitative value. It is considered a qualitative scale where the labels are mutually exclusive and do not carry numerical significance.
- **Characteristics:**
    - Labels variables with no inherent order or ranking.
    - No arithmetic calculations can be made.
    - The primary use is for categorization and identification.
- **Examples:**
    - Gender: Male, Female, Third gender, Other.
    - Languages spoken in a country.
    - Biological species: Human, Dog, Cat.
    - Taxonomic ranks: Archea, Bacteria, Eukarya.
    - Parts of speech in grammar: Noun, Verb, Adjective.
- **Applications:**
    - Frequency: Rate at which a label occurs in the dataset.
    - Proportion: Frequency divided by the total number of events.
    - Visualization: Can be visualized using pie charts or bar charts.

2. Ordinal Scale

- **Definition:** The ordinal scale is similar to the nominal scale but with an additional feature: the order or ranking of the values is significant.
- **Characteristics:**
    - The values can be ordered or ranked.
    - The exact difference between the values is not defined.
    - Central tendency can be measured using the median; however, mean is not applicable.
- **Examples:**
    - Likert scale (commonly used in surveys): Strongly Agree, Agree, Neutral, Disagree, Strongly Disagree.
- **Applications:**
    - Used to understand the order or rank of categories, but without knowing the exact difference between them.
    - Visualization: Bar charts can represent ordinal data.

3. Interval Scale

- **Definition:** Interval scales not only have an order but also define the exact difference between values. However, there is no true zero in the interval scale.

- **Characteristics:**
  - The order and exact difference between values are significant.
  - Arithmetic operations like mean, median, mode, and standard deviation are applicable.
- **Examples:**
  - Temperature measured in Celsius or Fahrenheit (e.g., 20°C to 30°C).
  - Dates on a calendar.
- **Applications:**
  - The interval scale allows for a variety of mathematical operations.
  - Visualization: Histograms and line charts can effectively represent interval data.

4. Ratio Scale

- **Definition:** The ratio scale includes order, exact values, and an absolute zero point, which enables meaningful ratios to be computed.
- **Characteristics:**
  - It has all the features of an interval scale, plus an absolute zero.
  - Can perform all arithmetic operations.
  - The measure of central tendencies (mean, median, mode), dispersion, and coefficients of variation can be computed.
- **Examples:**
  - Energy, mass, length, duration, volume.
  - Speed, weight, temperature (Kelvin scale).
- **Applications:**
  - Widely used in descriptive and inferential statistics.
  - Visualization: Histograms and scatter plots are often used.

Summary of Measurement Scales

Scale Type	Order	Exact Differences	True Zero	Examples
Nominal	No	No	No	Gender, Languages
Ordinal	Yes	No	No	Likert scale
Interval	Yes	Yes	No	Temperature (Celsius)
Ratio	Yes	Yes	Yes	Weight, Length, Energy

Comparing EDA with classical and Bayesian analysis

Classical Data Analysis Approach:

- **Steps:**
    1. **Problem definition.**
    2. **Data collection.**
    3. **Model development.**
    4. **Data analysis.**
    5. **Result communication.**

- Focuses on building and validating models as a key part of the process.

Exploratory Data Analysis (EDA) Approach:

- **Steps:**
    1. **Problem definition.**
    2. **Data collection.**
    3. **Data analysis.**
    4. **Model imposition.**
    5. **Result communication.**
- Focuses on the data's structure, outliers, patterns, and visualizations before imposing models.
- Does not assume deterministic or probabilistic models in the early stages.

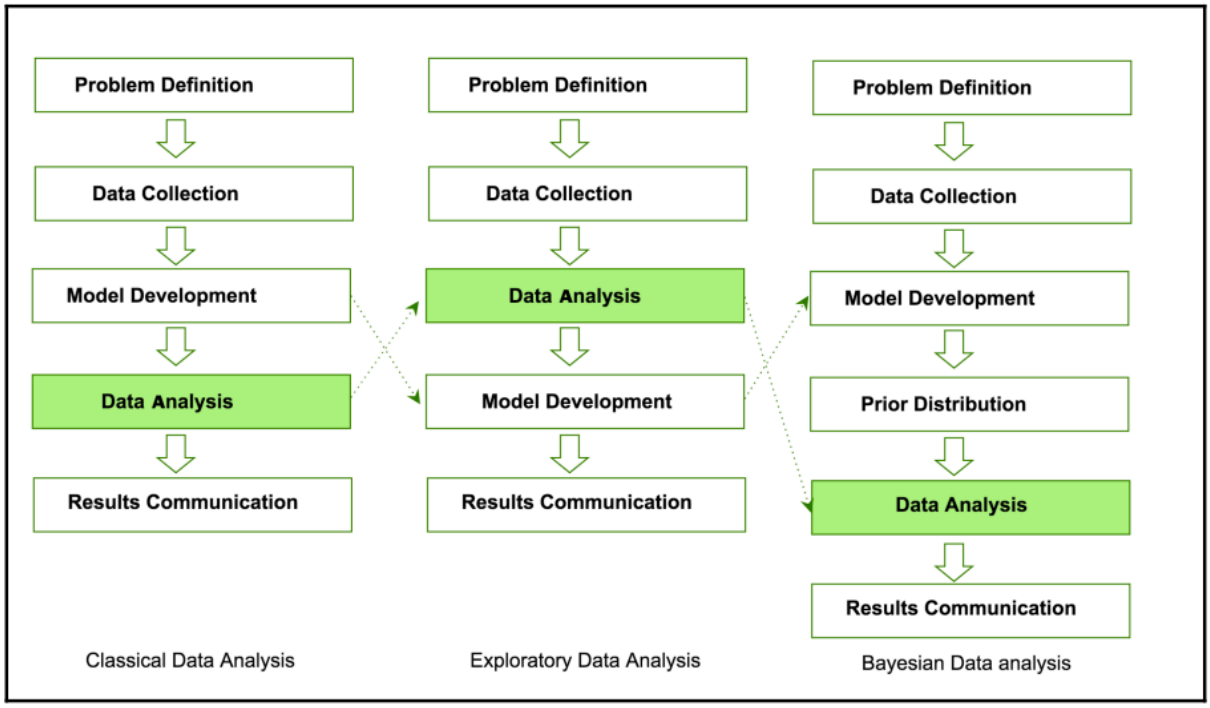
Bayesian Data Analysis Approach:

- **Steps:**
    1. **Problem definition.**
    2. **Data collection.**
    3. **Incorporation of prior probability distributions.**
    4. **Data analysis.**
    5. **Result communication.**
- Includes prior probability distributions to express beliefs about quantities before considering new evidence.
- Combines prior information and observed data for analysis.

Comparison:

- **Model Handling:**
    - Classical: Prioritizes model development early.
    - EDA: Delays model imposition; focuses on understanding data first.
    - Bayesian: Integrates prior knowledge into model development.
- **Flexibility:**

    - EDA is more flexible, allowing exploration of unexpected patterns.
- **Use Cases:**
    - Classical: Useful for hypothesis testing and predictive modeling.
    - EDA: Ideal for gaining initial insights and detecting anomalies.
    - Bayesian: Effective when prior knowledge significantly influences outcomes.



Python tools and packages

Python programming	Fundamental concepts of variables, string, and data types Conditionals and functions Sequences, collections, and iterations Working with files Object-oriented programming

NumPy	Create arrays with NumPy, copy arrays, and divide arrays Perform different operations on NumPy arrays Understand array selections, advanced indexing, and expanding Working with multi-dimensional arrays Linear algebraic functions and built-in NumPy functions
pandas	Understand and create `DataFrame` objects Subsetting data and indexing data Arithmetic functions, and mapping with pandas Managing index Building style for visual analysis
Matplotlib	Loading linear datasets Adjusting axes, grids, labels, titles, and legends Saving plots
SciPy	Importing the package Using statistical packages from SciPy Performing descriptive statistics Inference and data analysis

Prerequisites:

Setting up a virtual environment	`> pip install virtualenv` `> virtualenv Local_Version_Directory -p Python_System_Directory`

Reading/writing to files	`filename = "datamining.txt"`
	`file = open(filename, mode="r", encoding='utf-8')`
	`for line in file:`
	`  lines = file.readlines()`
	`print(lines)`
	`file.close()`

Error handling	`try:`
	`  Value = int(input("Type a number between 47 and 100:"))`
	`except ValueError:`
	`  print("You must type a number between 47 and 100!")`
	`else:`
	`  if (Value > 47) and (Value <= 100):`
	`  print("You typed: ", Value)`
	`  else:`
	`  print("The value you typed is incorrect!")`

Object-oriented concept	`class Disease:`
	`  def __init__(self, disease = 'Depression'):`
	`  self.type = disease`
	`  def getName(self):`
	`  print("Mental Health Diseases: {0}".format(self.type))`
	
	`d1 = Disease('Social Anxiety Disorder')`
	`d1.getName()`

Basic NumPy Operations

1. **Creating Arrays**
    - **1D:** `np.array([1, 2, 3])`
    - **2D:** `np.array([[1, 2], [3, 4]])`
    - **3D:** `np.array([[[1, 2], [3, 4]], [[5, 6], [7, 8]]])`
2. **Array Properties**
    - **Data type:** `array.dtype`
    - **Shape:** `array.shape`
    - **Strides:** `array.strides`
3. **Special Arrays**
    - **Ones:** `np.ones((3,4))`
    - **Zeros:** `np.zeros((2,3))`
    - **Random:** `np.random.random((2,2))`
    - **Evenly spaced:** `np.linspace(0, 2, 9)`
4. **File Operations**
    - **Save:** `np.savetxt('file.out', array, delimiter=',')`
    - **Load:** `np.loadtxt('file.out')`
5. **Broadcasting**
    - **Arrays with matching dimensions or one dimension as 1 can interact. Example:**

```
x = np.ones((3,4))
y = np.arange(4)
result = x - y
```

6. **Mathematics**
    - **Operations:** `np.add`, `np.subtract`, `np.multiply`, `np.divide`.

Pandas

Wes McKinney open-sourced the pandas library ([GitHub Link](https://github.com/wesm)) which is widely used in data science. It enables users to derive meaningful insights from data. This document covers fundamental techniques in pandas to provide a strong foundation.

1. Setting Default Parameters

```
import numpy as np
import pandas as pd
print("Pandas Version:", pd.__version__)
```

```
pd.set_option('display.max_columns', 500)
pd.set_option('display.max_rows', 500)
```

2. Creating Data Structures
DataFrame from Series

```
series = pd.Series([2, 3, 7, 11, 13, 17, 19, 23])
print(series)

series_df = pd.DataFrame({
 'A': range(1, 5),
 'B': pd.Timestamp('20190526'),
 'C': pd.Series(5, index=list(range(4)), dtype='float64'),
 'D': np.array([3] * 4, dtype='int64'),
 'E': pd.Categorical(["Depression", "Social Anxiety", "Bipolar Disorder", "Eating
Disorder"]),
 'F': 'Mental health',
 'G': 'is challenging'
})
print(series_df)
```

DataFrame from Dictionary

```
dict_df = [{'A': 'Apple', 'B': 'Ball'}, {'A': 'Aeroplane', 'B': 'Bat', 'C': 'Cat'}]
dict_df = pd.DataFrame(dict_df)
print(dict_df)
```

DataFrame from n-dimensional Arrays

```
sdf = {
 'County': ['Østfold', 'Hordaland', 'Oslo', 'Hedmark', 'Oppland', 'Buskerud'],
 'ISO-Code': [1, 2, 3, 4, 5, 6],
 'Area': [4180.69, 4917.94, 454.07, 27397.76, 25192.10, 14910.94],
 'Administrative centre': ["Sarpsborg", "Oslo", "City of Oslo", "Hamar", "Lillehammer",
"Drammen"]
}
sdf = pd.DataFrame(sdf)
```

```
print(sdf)
```

3. Loading External Datasets

```
columns = ['age', 'workclass', 'fnlwgt', 'education', 'education_num',
 'marital_status', 'occupation', 'relationship', 'ethnicity',
 'gender', 'capital_gain', 'capital_loss', 'hours_per_week', 'country_of_origin', 'income']

df = pd.read_csv('http://archive.ics.uci.edu/ml/machine-learning-
databases/adult/adult.data', names=columns)
df.head(10)
```

4. Dataframe Info

```
df.info()
```

Output:

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 32561 entries, 0 to 32560
Data columns (total 15 columns):
 age 32561 non-null int64
 ...
memory usage: 3.7+ MB
```

5. Selecting Rows and Columns

```
# Select a specific row
df.iloc[10]

# Select first 10 rows
df.iloc[0:10]

# Select a range of rows
df.iloc[10:15]

# Select the last 2 rows
df.iloc[-2:]
```

```
# Select every other row in columns 3-5
df.iloc[::2, 3:5].head()
```

6. Combining Pandas and NumPy

```
np.random.seed(24)
dFrame = pd.DataFrame({'F': np.linspace(1, 10, 10)})
dFrame = pd.concat([dFrame, pd.DataFrame(np.random.randn(10, 5),
columns=list('EDCBA'))], axis=1)
dFrame.iloc[0, 2] = np.nan
print(dFrame)
```

7. Styling DataFrames
Apply Conditional Styling

```
def colorNegativeValueToRed(value):
 if value < 0:
 color = 'red'
 elif value > 0:
 color = 'black'
 else:
 color = 'green'
 return 'color: %s' % color

s = df.style.applymap(colorNegativeValueToRed, subset=['A', 'B', 'C', 'D', 'E'])
s
```

Highlight Min/Max Values

```
def highlightMax(s):
 isMax = s == s.max()
 return ['background-color: orange' if v else '' for v in isMax]

def highlightMin(s):
 isMin = s == s.min()
 return ['background-color: green' if v else '' for v in isMin]

df.style.apply(highlightMax).apply(highlightMin).highlight_null(null_color='red')
```

Gradient Styling with Seaborn

```
import seaborn as sns
colorMap = sns.light_palette("pink", as_cmap=True)
styled = df.style.background_gradient(cmap=colorMap)
styled
```