



Lokmanya Tilak Jankalyan Shikshan Sanstha's
PRIYADARSHINI COLLEGE OF ENGINEERING, NAGPUR
(An Autonomous Institute affiliated to Rashtrasant Tukadoji Maharaj Nagpur University)

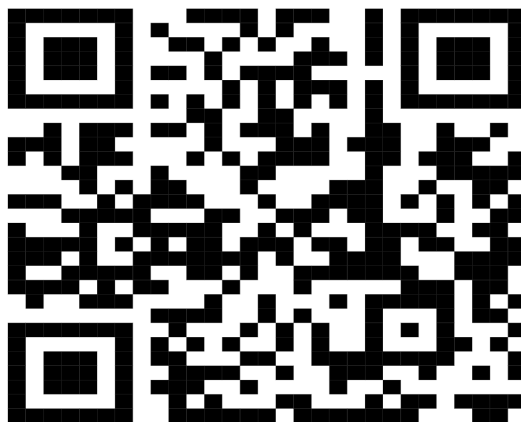


DEPARTMENT OF INFORMATION TECHNOLOGY

DATA EXPLORATION AND VISUALIZATION - LAB MANUAL

Subject code: : **25UAI507T**
Academic Year : 2025-2026
Semester: V

KNOW ABOUT YOUR LAB





Lokmanya Tilak Jankalyan Shikshan Sanstha's
PRIYADARSHINI COLLEGE OF ENGINEERING, NAGPUR
(An Autonomous Institute affiliated to Rashtrasant Tukadoji Maharaj Nagpur University)



DEPARTMENT OF INFORMATION TECHNOLOGY

INDEX


Sr No.	Contents
1	Vision & Mission of the Institute
2	Vision & Mission of the Department
3	Program Educational Objectives
4	Program Outcomes
5	Program Specific Outcomes
6	Laboratory Infrastructure
7	Course Outcomes & CO-PO & PSO Mapping
8	List of Experiments & Content Beyond Syllabus



Lokmanya Tilak Jankalyan Shikshan Sanstha's
PRIYADARSHINI COLLEGE OF ENGINEERING, NAGPUR
(An Autonomous Institute affiliated to Rashtrasant Tukadoji Maharaj Nagpur University)




DEPARTMENT OF INFORMATION TECHNOLOGY



LOKMANYA TILAK JANKALYAN SHIKSHAN SANSTHA'S

PRIYADARSHINI COLLEGE OF ENGINEERING

An Autonomous Institute Affiliated to R.T.M. Nagpur University, Nagpur
Accredited with Grade 'A+' by NAAC
Near C.R.P.F. Campus, Hingna Road, Nagpur - 440 019 (Maharashtra) India
Phone : 07104 - 299648, Fax : 07104-299681
E-mail : principal.pce.ngp@gmail.com • Website: www.pcenagpur.edu.in
AICTE ID No. 1-5435581; DTE CODE No. 4123,
UNIVERSITY CODE No. : 278




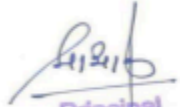
Vision

To become one of the India's leading Engineering Institutes in both education and research. We are committed to provide quality and state-of-the-art technical education to our students so that they become Technologically competent and in turn contribute for creating a great society.

Mission

1. Fostering a dynamic learning environment that equips students with Technical expertise, problem-solving skills and a deep commitment to ethical practices.
2. To cultivate a culture of innovation, incubation, research and entrepreneurship that drives technological advancements.
3. To uphold the spirit of mutual excellence while interacting with stake holders of our Institutional ecosystem.
4. Promoting lifelong learning, professional growth and ensuring holistic development of students and the well being of society.





Principal
Priyadarshini College of Engg.
Nagpur.



Lokmanya Tilak Jankalyan Shikshan Sanstha's
PRIYADARSHINI COLLEGE OF ENGINEERING, NAGPUR
(An Autonomous Institute affiliated to Rashtrasant Tukadoji Maharaj Nagpur University)



DEPARTMENT OF INFORMATION TECHNOLOGY

PROGRAM EDUCATIONAL OBJECTIVES:

PEOs	Program Educational Objectives Statements
PEO1	Demonstrate strong technical expertise and uphold ethical values to excel in their professional careers.
PEO2	Apply advanced skills and knowledge in Information Technology to address real-world challenges and contribute to industry success.
PEO3	Pursue lifelong learning and engage in research and innovation to address societal needs and advance the field of Information Technology.



Lokmanya Tilak Jankalyan Shikshan Sanstha's
PRIYADARSHINI COLLEGE OF ENGINEERING, NAGPUR
(An Autonomous Institute affiliated to Rashtrasant Tukadoji Maharaj Nagpur University)



DEPARTMENT OF INFORMATION TECHNOLOGY

PROGRAM OUTCOMES:

Engineering Graduates will Able to:

POs	Program Outcomes
PO1	Engineering knowledge: Apply the knowledge of mathematics, science, engineering fundamentals and an engineering specialization to the solution of complex engineering problems.
PO2	Problem analysis: Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.
PO3	Design/development of solutions: Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.
PO4	Conduct investigations of complex problems: Use research-based knowledge and research methods Including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.



Lokmanya Tilak Jankalyan Shikshan Sanstha's
PRIYADARSHINI COLLEGE OF ENGINEERING, NAGPUR
(An Autonomous Institute affiliated to Rashtrasant Tukadoji Maharaj Nagpur University)



DEPARTMENT OF INFORMATION TECHNOLOGY

PO5	The engineer and society: Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.
PO6	The engineer and society: Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.
PO7	Environment and sustainability: Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.
PO8	Ethics: Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.
PO9	Individual and team work: Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.

PO10	Communication: Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.
PO11	Project management and finance: Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.
PO12	Life-long learning: Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.



Lokmanya Tilak Jankalyan Shikshan Sanstha's
PRIYADARSHINI COLLEGE OF ENGINEERING, NAGPUR
(An Autonomous Institute affiliated to Rashtrasant Tukadoji Maharaj Nagpur University)



DEPARTMENT OF INFORMATION TECHNOLOGY

PROGRAM SPECIFIC OUTCOMES :

PSOs	Program Specific Outcomes
PSO1	An ability to apply mathematical foundations, algorithmic principles and computer science theory in the modeling and design of software systems of varying complexity.
PSO2	An ability to work with Open Source Software and use off the shelf utilities for program integration.



Lokmanya Tilak Jankalyan Shikshan Sanstha's
PRIYADARSHINI COLLEGE OF ENGINEERING, NAGPUR
(An Autonomous Institute affiliated to Rashtrasant Tukadoji Maharaj Nagpur University)



DEPARTMENT OF INFORMATION TECHNOLOGY

COURSE OUTCOMES :

Course Outcomes	Statement
CO1	Understand the fundamental concepts of data exploration, preprocessing techniques, and visualization tools used in data analytics.
CO2	Apply appropriate data preprocessing, cleaning, and transformation techniques to prepare datasets for effective analysis and visualization.
CO3	Develop and interpret various static and interactive data visualizations using tools like Matplotlib, Seaborn, Plotly, Tableau, or Power BI to derive insights from datasets.

CO-PO & PSO MAPPING:

	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12	PSO 1	PSO2
CO1	2	1	1	1	1	1	2	1	1	1	1	1	2	2
CO2	2	2	1	1	1	1	2	1	1	1	1	1	2	3
CO3	2	2	3	2	1	1	3	1	1	1	1	1	3	3



Lokmanya Tilak Jankalyan Shikshan Sanstha's
PRIYADARSHINI COLLEGE OF ENGINEERING, NAGPUR
(An Autonomous Institute affiliated to Rashtrasant Tukadoji Maharaj Nagpur University)



DEPARTMENT OF INFORMATION TECHNOLOGY

List of Experiments:

Sr. No.	Name of Practical
1	To install and explore Matplotlib for creating basic plots.
2	To use Seaborn for generating aesthetically pleasing statistical visualizations
3	To compare different Python visualization libraries (Matplotlib, Seaborn, Plotly) for various data types.
4	To load a dataset using Pandas and perform basic data cleaning (handling missing values and duplicates).
5	To preprocess categorical and numerical data for visualization using Pandas and NumPy.
6	To apply feature scaling and normalization techniques for effective visualization
7	To create bar charts, line graphs, and scatter plots using Matplotlib and Seaborn.
8	To visualize distributions using histograms, box plots, and violin plots in Seaborn.
9	To create advanced visualizations like heatmaps and pair plots to analyze relationships in data.
10	To create an interactive dashboard using Plotly and Dash
Content Beyond Syllabus	
11	To integrate multiple visualizations into a single dashboard.
12	To build a weather data visualization app for analyzing temperature trends over time.



DEPARTMENT OF INFORMATION TECHNOLOGY

PRACTICAL NO.1

Aim:

To install the Matplotlib library in Python and use it to create basic plots such as line plot, bar chart, scatter plot, and histogram.

Theory:

Matplotlib is a widely used plotting library in Python, primarily used for 2D visualizations. It enables users to generate plots, histograms, power spectra, bar charts, error charts, and more with just a few lines of code.

- The **matplotlib.pyplot** module provides a MATLAB-like interface.
- Plots are highly customizable (titles, labels, legends, colors, etc.)
- Matplotlib supports exporting graphics to different file formats (PNG, PDF, SVG, etc.)



DEPARTMENT OF INFORMATION TECHNOLOGY

Key Features:

- Easy to use for beginners.
- Works well with NumPy and pandas.
- Produces publication-quality figures.

Code:

✓ Step 1: Install Matplotlib

```
pip install matplotlib
```

✓ Step 2: Basic Plotting Examples

```
import matplotlib.pyplot as plt
```

1. Line Plot

```
x = [1, 2, 3, 4, 5]
```

```
y = [2, 3, 5, 7, 11]
```

```
plt.plot(x, y, marker='o')
```

```
plt.title("Line Plot")
```

```
plt.xlabel("X-axis")
```

```
plt.ylabel("Y-axis")
```

```
plt.grid(True)
```

```
plt.show()
```



DEPARTMENT OF INFORMATION TECHNOLOGY

2. Bar Chart

```
categories = ['A', 'B', 'C', 'D']  
values = [4, 7, 1, 8]  
plt.bar(categories, values, color='orange')  
plt.title("Bar Chart")  
plt.xlabel("Category")  
plt.ylabel("Value")  
plt.show()
```

3. Scatter Plot

```
x = [1, 2, 3, 4, 5]  
y = [5, 7, 4, 6, 5]  
plt.scatter(x, y, color='red')  
plt.title("Scatter Plot")  
plt.xlabel("X-axis")  
plt.ylabel("Y-axis")  
plt.show()
```

4. Histogram

```
import numpy as np  
data = np.random.randn(1000)  
plt.hist(data, bins=20, color='green')  
plt.title("Histogram")  
plt.xlabel("Value")  
plt.ylabel("Frequency")  
plt.show()
```

Expected Output: A circular pie chart showing percentage distribution.



DEPARTMENT OF INFORMATION TECHNOLOGY

Viva Questions:

1. What is Matplotlib and what is it used for?
 - Matplotlib is a Python library used to create static, animated, and interactive visualizations such as plots and charts.
2. What is the difference between a line plot and a scatter plot?
 - A line plot connects data points with lines, showing trends, while a scatter plot displays data as individual points to show relationships or patterns.
3. How do you label axes and add a title in Matplotlib?
 - Using `plt.xlabel()`, `plt.ylabel()`, and `plt.title()` functions.
4. Which module of Matplotlib is most commonly used for plotting?
 - The `pyplot` module (`import matplotlib.pyplot as plt`) is most commonly used.



DEPARTMENT OF INFORMATION TECHNOLOGY

PRACTICAL NO: 2

Aim:

To install and use the Seaborn library for creating visually appealing and informative statistical graphics in Python.

Theory:

Seaborn is a Python data visualization library based on Matplotlib. It provides a high-level interface for drawing attractive and informative statistical graphics.

It works well with pandas DataFrames and integrates closely with NumPy and SciPy.

Key Features:

- Built-in themes and color palettes.
- Easy integration with pandas DataFrames.
- Functions for visualizing univariate and bivariate distributions.
- Automatic estimation and plotting of linear regression models.



Lokmanya Tilak Jankalyan Shikshan Sanstha's
PRIYADARSHINI COLLEGE OF ENGINEERING, NAGPUR
(An Autonomous Institute affiliated to Rashtrasant Tukadoji Maharaj Nagpur University)



DEPARTMENT OF INFORMATION TECHNOLOGY

☀ **Common Seaborn Plots:**

- **`sns.histplot()`** – histogram with optional KDE.
- **`sns.scatterplot()`** – scatter plot.
- **`sns.boxplot()`** – box and whisker plot.
- **`sns.barplot()`** – bar chart with confidence intervals.
- **`sns.pairplot()`** – multiple pairwise relationships.

Code:

✅ **Step 1: Install Seaborn**

```
pip install seaborn
```

✅ **Step 2: Python Program Using Seaborn**

```
import seaborn as sns
```

```
import matplotlib.pyplot as plt
```

```
import pandas as pd
```



DEPARTMENT OF INFORMATION TECHNOLOGY

Load built-in dataset

```
tips = sns.load_dataset("tips")
```

1. Histogram

```
sns.histplot(data=tips, x="total_bill", kde=True)
```

```
plt.title("Histogram of Total Bill")
```

```
plt.show()
```

2. Scatter Plot

```
sns.scatterplot(data=tips, x="total_bill", y="tip", hue="sex")
```

```
plt.title("Total Bill vs Tip by Gender")
```

```
plt.show()
```

3. Box Plot

```
sns.boxplot(data=tips, x="day", y="total_bill", hue="sex")
```

```
plt.title("Boxplot of Total Bill by Day and Gender")
```

```
plt.show()
```

4. Pair Plot

```
sns.pairplot(tips, hue="sex")
```

```
plt.suptitle("Pairplot of Tips Dataset", y=1.02)
```

```
plt.show()
```




Lokmanya Tilak Jankalyan Shikshan Sanstha's
PRIYADARSHINI COLLEGE OF ENGINEERING, NAGPUR
(An Autonomous Institute affiliated to Rashtrasant Tukadoji Maharaj Nagpur University)



DEPARTMENT OF INFORMATION TECHNOLOGY

Expected Output:

1. A histogram showing the distribution of total bill values with a smooth KDE line.
2. A scatter plot showing the relationship between total bill and tip, color-coded by gender.
3. A box plot comparing total bill amounts across different days, with male/female differentiation.
4. A multi-panel pairplot showing pairwise relationships among numerical variables (e.g., total_bill, tip, size), color-coded by gender.



DEPARTMENT OF INFORMATION TECHNOLOGY

Viva Questions:

1. What is Seaborn and how is it different from Matplotlib?
 - Seaborn is a statistical plotting library built on top of Matplotlib that provides prettier and more informative visualizations with less code.
2. What type of data format does Seaborn work best with?
 - Seaborn works best with pandas DataFrames.
3. How do you display a distribution using Seaborn?
 - Using `sns.histplot()` or `sns.kdeplot()` functions.
4. What does the `hue` parameter do in Seaborn plots?
 - It adds a third dimension by grouping the data based on the value of another categorical variable (e.g., gender).



DEPARTMENT OF INFORMATION TECHNOLOGY

PRACTICAL NO. 3

Aim:

To compare Matplotlib, Seaborn, and Plotly libraries in Python for creating visualizations and to understand their strengths, limitations, and best use cases based on different data types and visualization needs.

Theory:

Data visualization is a crucial step in data analysis, and Python offers several libraries for this. Among the most popular are Matplotlib, Seaborn, and Plotly. Each serves different purposes and has unique features:

Feature	Matplotlib	Seaborn	Plotly
Type	Low-level	High-level (built on Matplotlib)	High-level & interactive
Ease of use	Requires more code	Easier with built-in themes	Very intuitive with interactivity
Interactivity	Static	Static	Interactive (zoom, hover, etc.)
Customization	Highly customizable	Limited to Matplotlib backend	Very customizable
Best use case	Custom static plots	Statistical data visualization	Web-based, interactive dashboards

Code:

We'll use the same dataset (tips) and create the same type of plot — a scatter plot of total_bill vs tip — using all three libraries to compare their output.

✅ Step 1: Install Required Libraries

pip install matplotlib seaborn plotly



DEPARTMENT OF INFORMATION TECHNOLOGY

✓ Step 2: Comparison Code

```
# Import libraries
import matplotlib.pyplot as plt
import seaborn as sns
import plotly.express as px
import pandas as pd

# Load dataset
tips = sns.load_dataset("tips")

# ----- Matplotlib -----
plt.figure(figsize=(6, 4))
plt.scatter(tips['total_bill'], tips['tip'], c='blue')
plt.title("Matplotlib: Total Bill vs Tip")
plt.xlabel("Total Bill")
plt.ylabel("Tip")
plt.grid(True)

plt.show()

# ----- Seaborn -----
sns.scatterplot(data=tips, x='total_bill', y='tip', hue='sex')
plt.title("Seaborn: Total Bill vs Tip by Gender")
plt.show()

# ----- Plotly -----
fig = px.scatter(tips, x='total_bill', y='tip', color='sex',
                title='Plotly: Total Bill vs Tip by Gender',
                labels={'total_bill': 'Total Bill', 'tip': 'Tip'})
fig.show()
```

Expected Output:

- Matplotlib Plot: A basic, static scatter plot with blue dots, no interactivity.
- Seaborn Plot: A more polished static plot with gender-based coloring, legend, and better aesthetics.



DEPARTMENT OF INFORMATION TECHNOLOGY

-
- Plotly Plot: A fully interactive scatter plot (hover, zoom, pan), also with gender-based color.

Viva Questions:

1. What are the major differences between Matplotlib, Seaborn, and Plotly?
 - Matplotlib is low-level and static, Seaborn is high-level and better for statistical plots, and Plotly is interactive and ideal for dashboards.
2. Which library is best for creating interactive visualizations?
 - Plotly is best for creating interactive visualizations.
3. Why is Seaborn considered easier than Matplotlib for statistical plots?
 - Seaborn has built-in functions and themes for complex statistical plots with fewer lines of code.
4. Can you use Seaborn and Plotly together? Why or why not?
 - No, not directly. Seaborn is based on Matplotlib and produces static plots, while Plotly is a separate interactive library.

PRACTICAL NO. 4



DEPARTMENT OF INFORMATION TECHNOLOGY

Aim:

To load a dataset using the Pandas library in Python and perform basic data cleaning tasks such as handling missing values and duplicate entries.

Theory:

Pandas is a powerful Python library used for data manipulation and analysis. It provides two primary data structures:

- Series – 1D labeled array.
- DataFrame – 2D table of data (rows and columns), similar to an Excel spreadsheet.

Data Cleaning is a crucial step before analysis or modeling. Common tasks include:

- Handling Missing Values:
 - `df.isnull()` – identifies missing values.
 - `df.dropna()` – removes rows with missing values.
 - `df.fillna(value)` – fills missing values with a given value.
- Handling Duplicates:
 - `df.duplicated()` – returns boolean Series for duplicate rows.
 - `df.drop_duplicates()` – removes duplicate rows.

Code:



DEPARTMENT OF INFORMATION TECHNOLOGY

We'll use a small sample dataset to demonstrate the process.

```
import pandas as pd
```

```
# Sample dataset with missing values and duplicates
```

```
data = {  
    'Name': ['Alice', 'Bob', 'Charlie', 'Alice', None],  
    'Age': [25, 30, None, 25, 22],  
    'City': ['New York', 'Los Angeles', 'Chicago', 'New York', 'Miami']  
}
```

```
# Load data into a DataFrame
```

```
df = pd.DataFrame(data)
```

```
print("Original Dataset:")
```

```
print(df)
```

```
# -----
```

```
# Handling Missing Values
```

```
# -----
```

```
print("\nMissing Values in Dataset:")
```

```
print(df.isnull().sum())
```

```
# Fill missing 'Age' with mean and drop missing 'Name'
```

```
df['Age'].fillna(df['Age'].mean(), inplace=True)
```

```
df.dropna(subset=['Name'], inplace=True)
```

```
# -----
```

```
# Handling Duplicates
```

```
# -----
```

```
print("\nCheck for Duplicate Rows:")
```

```
print(df.duplicated())
```

```
# Drop duplicates
```

```
df.drop_duplicates(inplace=True)
```

```
# Final cleaned dataset
```

```
print("\nCleaned Dataset:")
```

```
print(df)
```

Expected Output:



Lokmanya Tilak Jankalyan Shikshan Sanstha's
PRIYADARSHINI COLLEGE OF ENGINEERING, NAGPUR
(An Autonomous Institute affiliated to Rashtrasant Tukadoji Maharaj Nagpur University)



DEPARTMENT OF INFORMATION TECHNOLOGY

Original Dataset:

	Name	Age	City
0	Alice	25.0	New York
1	Bob	30.0	Los Angeles
2	Charlie	NaN	Chicago
3	Alice	25.0	New York
4	None	22.0	Miami

Missing Values in Dataset:

Name 1
Age 1
City 0
dtype: int64

Check for Duplicate Rows:

0 False
1 False
2 False
3 True
Name: duplicated, dtype: bool

Cleaned Dataset:

	Name	Age	City
0	Alice	25.000000	New York
1	Bob	30.000000	Los Angeles
2	Charlie	26.666667	Chicago

Note: The missing Age is filled with the average (≈ 26.67), and duplicate rows and missing names are removed.



DEPARTMENT OF INFORMATION TECHNOLOGY

Viva Questions:

1. What is the purpose of data cleaning in data analysis?
 - Data cleaning ensures the dataset is free from errors, missing values, and duplicates, making it ready for analysis.
2. How can you identify missing values in a Pandas DataFrame?
 - By using `df.isnull()` or `df.isnull().sum()` to check which columns have missing values.
3. What is the difference between `dropna()` and `fillna()` in Pandas?
 - `dropna()` removes missing value rows; `fillna()` replaces them with specified values (mean, median, etc.).
4. How do you check and remove duplicate rows in a DataFrame?
 - Use `df.duplicated()` to check and `df.drop_duplicates()` to remove them.



DEPARTMENT OF INFORMATION TECHNOLOGY

Aim:

To preprocess categorical and numerical data using Pandas and NumPy for effective visualization and analysis.

Theory:

In real-world datasets, raw data often needs preprocessing to be suitable for visualization or machine learning.

Preprocessing includes:

- Handling missing values
- Converting categorical data to numerical form
- Scaling or transforming numerical data

♦ Categorical Data:

- These are data with limited, fixed values (e.g., gender, city).
- Often stored as text and need encoding for numeric analysis.
- Techniques:
 - **Label Encoding** – assigns each unique category a number.
 - **One-Hot Encoding** – creates binary columns for each category.

♦ Numerical Data:



DEPARTMENT OF INFORMATION TECHNOLOGY

-
- Continuous or discrete numbers (e.g., age, income).
 - May need normalization or scaling.
 - Can be processed using NumPy for calculations (mean, std, etc.)

Code:

```
import pandas as pd
```

```
import numpy as np
```

```
# Sample dataset with categorical and numerical data
```

```
data = {
```

```
    'Name': ['Alice', 'Bob', 'Charlie', 'David', 'Eva'],
```

```
    'Gender': ['Female', 'Male', 'Male', 'Male', 'Female'],
```

```
    'Age': [25, 30, np.nan, 22, 28],
```

```
    'City': ['New York', 'Los Angeles', 'Chicago', 'Chicago', 'Miami']
```

```
}
```

```
# Create DataFrame
```

```
df = pd.DataFrame(data)
```

```
print("Original Data:")
```

```
print(df)
```



Lokmanya Tilak Jankalyan Shikshan Sanstha's
PRIYADARSHINI COLLEGE OF ENGINEERING, NAGPUR
(An Autonomous Institute affiliated to Rashtrasant Tukadoji Maharaj Nagpur University)



DEPARTMENT OF INFORMATION TECHNOLOGY

Handle missing numerical values

df['Age'].fillna(df['Age'].mean(), inplace=True)

Encode categorical columns

Label Encoding for Gender

df['Gender_encoded'] = df['Gender'].map({'Female': 0, 'Male': 1})

One-Hot Encoding for City

city_dummies = pd.get_dummies(df['City'], prefix='City')

Merge one-hot encoded columns

df = pd.concat([df, city_dummies], axis=1)

Drop original categorical columns

df.drop(['Gender', 'City'], axis=1, inplace=True)



Lokmanya Tilak Jankalyan Shikshan Sanstha's
PRIYADARSHINI COLLEGE OF ENGINEERING, NAGPUR
(An Autonomous Institute affiliated to Rashtrasant Tukadoji Maharaj Nagpur University)



DEPARTMENT OF INFORMATION TECHNOLOGY

Final Preprocessed Data

```
print("\nPreprocessed Data:")
```

```
print(df)
```

Expected Output:

Original Data:

	Name	Gender	Age	City
0	Alice	Female	25.0	New York
1	Bob	Male	30.0	Los Angeles
2	Charlie	Male	NaN	Chicago
3	David	Male	22.0	Chicago
4	Eva	Female	28.0	Miami

Preprocessed Data:



DEPARTMENT OF INFORMATION TECHNOLOGY

	Name	Age	Gender_encoded	City_Chicago	City_Los Angeles	City_Miami	City_New York
0	Alice	25.0	0	0	0	1	
1	Bob	30.0	1	0	1	0	
2	Charlie	26.25	1	1	0	0	
3	David	22.0	1	1	0	0	
4	Eva	28.0	0	0	0	1	

The missing **Age** was filled with the mean value (26.25), **Gender** was label encoded, and **City** was one-hot encoded for easy visualization and further analysis.

Viva Questions:

1. Why do we preprocess categorical data before visualization or modeling?
 - Because most algorithms and some visualizations require numerical input, not text labels.
2. What is the difference between Label Encoding and One-Hot Encoding?
 - Label Encoding assigns numbers to categories, One-Hot Encoding creates separate binary columns for each category.
3. How can you handle missing numerical data in Pandas?
 - By using methods like **fillna()** to fill missing values with mean, median, or a constant.



DEPARTMENT OF INFORMATION TECHNOLOGY

Aim:

To apply feature scaling and normalization techniques using Python to preprocess data for improved and meaningful visualization.

Theory:

♦ What is Feature Scaling?

Feature scaling is the process of bringing all features (columns) to a similar scale so that no feature dominates the others due to its magnitude.

♦ Why is it important for visualization?

- Ensures fair comparison across features.
- Prevents skewed graphs due to differing units (e.g., salary vs. age).
- Highlights patterns more clearly.

Common Techniques:

Method	Description
Min-Max Scaling	Scales values between 0 and 1 using: $\frac{(X - \min)}{(\max - \min)}$
Z-score Normalization (Standardization)	Centers around mean (0) with unit variance using: $\frac{(X - \mu)}{\sigma}$
MaxAbs Scaling	Scales each feature by its maximum absolute value.

These techniques are especially useful for visualization like heatmaps, PCA plots, or any multi-dimensional graph.

Code:



DEPARTMENT OF INFORMATION TECHNOLOGY

We'll demonstrate Min-Max Scaling and Z-score Normalization using a sample dataset.

```
import pandas as pd
import numpy as np
from sklearn.preprocessing import MinMaxScaler, StandardScaler
import matplotlib.pyplot as plt

# Sample dataset
data = {
    'Age': [22, 25, 47, 52, 46],
    'Salary': [25000, 32000, 56000, 60000, 58000]
}
df = pd.DataFrame(data)
print("Original Data:\n", df)

# -----
# Min-Max Scaling (0 to 1)
# -----
minmax_scaler = MinMaxScaler()
df_minmax = pd.DataFrame(minmax_scaler.fit_transform(df), columns=['Age_Scaled',
'Salary_Scaled'])
print("\nMin-Max Scaled Data:\n", df_minmax)

# -----
# Standardization (Z-score)
# -----
std_scaler = StandardScaler()
df_standardized = pd.DataFrame(std_scaler.fit_transform(df), columns=['Age_Std',
'Salary_Std'])
print("\nZ-score Normalized Data:\n", df_standardized)

# -----
# Visualization: Compare Original vs Scaled
# -----
plt.figure(figsize=(10, 5))

# Original
plt.subplot(1, 3, 1)
plt.plot(df['Age'], label='Age')

plt.plot(df['Salary'], label='Salary')
plt.title("Original Data")
plt.legend()
```




DEPARTMENT OF INFORMATION TECHNOLOGY

```
# Min-Max Scaled
plt.subplot(1, 3, 2)
plt.plot(df_minmax['Age_Scaled'], label='Age_Scaled')
plt.plot(df_minmax['Salary_Scaled'], label='Salary_Scaled')
plt.title("Min-Max Scaled")
plt.legend()

# Standardized
plt.subplot(1, 3, 3)
plt.plot(df_standardized['Age_Std'], label='Age_Std')
plt.plot(df_standardized['Salary_Std'], label='Salary_Std')
plt.title("Z-score Normalized")
plt.legend()

plt.tight_layout()
plt.show()
```

Expected Output:

- **Original Data:** Displays raw Age and Salary values.
- **Min-Max Scaled Data:** Values between 0 and 1.
- **Z-score Normalized Data:** Mean = 0, Std Dev = 1.

Graphs:

- First plot shows original values on very different scales.
- Second (Min-Max) and third (Z-score) plots show features on similar scales, making it easier to compare trends visually.

Viva Questions:



DEPARTMENT OF INFORMATION TECHNOLOGY

1. What is the purpose of feature scaling in data visualization?
 - To bring all numerical features to a similar scale so that comparisons and visualizations are meaningful.
2. How does Min-Max scaling differ from Z-score normalization?
 - Min-Max scales between 0–1, while Z-score normalization centers the data around 0 with unit variance.
3. Which Python libraries are used for feature scaling?
 - **scikit-learn** provides **MinMaxScaler**, **StandardScaler**, and others.
4. When should feature scaling be applied before visualization?
 - When visualizing features with different units/magnitudes (e.g., age vs. income), especially in scatter plots, heatmaps, or PCA.



DEPARTMENT OF INFORMATION TECHNOLOGY

Aim:

To create bar charts, line graphs, and scatter plots using Matplotlib and Seaborn libraries in Python for effective data visualization.

Theory:

Visualization is an essential part of data analysis that helps in understanding trends, relationships, and patterns within data. Python offers libraries like Matplotlib and Seaborn to create high-quality plots.

♦ **Matplotlib:**

- A low-level, highly customizable plotting library.
- Suitable for static, basic plots.

♦ **Seaborn:**

- Built on top of Matplotlib.
- Provides a high-level interface for statistical graphics.
- Prettier and easier for certain types of plots, especially with pandas DataFrames.

♦ **Types of Plots Covered:**



DEPARTMENT OF INFORMATION TECHNOLOGY

Plot Type	Purpose
Bar Chart	Show comparisons between categories.
Line Graph	Show trends over a continuous variable like time.
Scatter Plot	Show relationship between two numerical variables.

Code:

```
import matplotlib.pyplot as plt
import seaborn as sns
import pandas as pd
```

```
# Sample data
```

```
data = {
    'Month': ['Jan', 'Feb', 'Mar', 'Apr', 'May'],
    'Sales': [200, 250, 300, 280, 320],
    'Profit': [20, 35, 50, 40, 60]
}
df = pd.DataFrame(data)
```

```
# -----
```

```
# Bar Chart (Matplotlib)
```

```
# -----
```

```
plt.figure(figsize=(6, 4))
plt.bar(df['Month'], df['Sales'], color='skyblue')
plt.title("Monthly Sales (Bar Chart - Matplotlib)")
plt.xlabel("Month")
plt.ylabel("Sales")
plt.grid(True, linestyle='--', alpha=0.5)
plt.show()
```

```
# -----
```

```
# Line Graph (Seaborn)
```

```
# -----
```

```
plt.figure(figsize=(6, 4))
sns.lineplot(x='Month', y='Sales', data=df, marker='o', label='Sales')
sns.lineplot(x='Month', y='Profit', data=df, marker='s', label='Profit')
```



DEPARTMENT OF INFORMATION TECHNOLOGY

```
plt.title("Sales and Profit Over Months (Line Graph - Seaborn)")
plt.ylabel("Value")
plt.legend()
plt.grid(True, linestyle='--', alpha=0.5)
plt.show()
```

```
# -----
# Scatter Plot (Matplotlib)
# -----
plt.figure(figsize=(6, 4))
plt.scatter(df['Sales'], df['Profit'], color='red')
plt.title("Sales vs Profit (Scatter Plot - Matplotlib)")
plt.xlabel("Sales")
plt.ylabel("Profit")
plt.grid(True, linestyle='--', alpha=0.5)
plt.show()
```

Expected Output:

1. Bar Chart (Matplotlib) – Displays sales values per month as vertical bars.
2. Line Graph (Seaborn) – Two lines showing trends of sales and profit across months.
3. Scatter Plot (Matplotlib) – Shows the relationship between sales and profit; each point is one month.

These visualizations help:

- Compare monthly performance.
- See trends in growth or decline.
- Observe correlation between sales and profit.



Lokmanya Tilak Jankalyan Shikshan Sanstha's
PRIYADARSHINI COLLEGE OF ENGINEERING, NAGPUR
(An Autonomous Institute affiliated to Rashtrasant Tukadoji Maharaj Nagpur University)



DEPARTMENT OF INFORMATION TECHNOLOGY

Viva Questions:

1. What is the difference between a bar chart and a line graph?
 - Bar charts show categorical comparisons; line graphs show trends over a continuous interval (like time).
2. Why use Seaborn when Matplotlib can also plot graphs?
 - Seaborn provides a cleaner syntax, built-in themes, and better default aesthetics for statistical plots.
3. What kind of data is best suited for a scatter plot?
 - Numerical data where you want to show relationships or correlations between two variables.
4. Which function is used in Seaborn to create a line plot?
 - **sns.lineplot()** is used to create line plots in Seaborn.



Lokmanya Tilak Jankalyan Shikshan Sanstha's
PRIYADARSHINI COLLEGE OF ENGINEERING, NAGPUR
(An Autonomous Institute affiliated to Rashtrasant Tukadoji Maharaj Nagpur University)



DEPARTMENT OF INFORMATION TECHNOLOGY

Aim:

To visualize the **distribution of data** using **histograms**, **box plots**, and **violin plots** with the **Seaborn** library in Python.

Theory:

Understanding the distribution of data is essential for statistical analysis and decision-making. Seaborn provides multiple tools for this purpose:

Plot Type	Description
Histogram	Shows the frequency of data within intervals. Helps understand the shape of the distribution.
Box Plot	Summarizes data with median, quartiles, and outliers using a simple box-and-whisker diagram.
Violin Plot	Combines box plot with a kernel density estimate to show the distribution's shape.

These plots help identify **skewness**, **outliers**, and **spread** of the data.

Code:



Lokmanya Tilak Jankalyan Shikshan Sanstha's
PRIYADARSHINI COLLEGE OF ENGINEERING, NAGPUR
(An Autonomous Institute affiliated to Rashtrasant Tukadoji Maharaj Nagpur University)



DEPARTMENT OF INFORMATION TECHNOLOGY

```
import seaborn as sns
```

```
import matplotlib.pyplot as plt
```

```
# Load example dataset
```

```
tips = sns.load_dataset("tips")
```

```
# -----
```

```
# Histogram of Total Bill
```

```
# -----
```

```
plt.figure(figsize=(6, 4))
```

```
sns.histplot(data=tips, x="total_bill", kde=True, color="skyblue")
```

```
plt.title("Histogram of Total Bill")
```

```
plt.xlabel("Total Bill")
```

```
plt.ylabel("Frequency")
```

```
plt.grid(True, linestyle='--', alpha=0.5)
```

```
plt.show()
```

```
# -----
```

```
# Box Plot of Total Bill by Day
```

```
# -----
```

```
plt.figure(figsize=(6, 4))
```

```
sns.boxplot(data=tips, x="day", y="total_bill", palette="Set2")
```

```
plt.title("Box Plot of Total Bill by Day")
```




Lokmanya Tilak Jankalyan Shikshan Sanstha's
PRIYADARSHINI COLLEGE OF ENGINEERING, NAGPUR
(An Autonomous Institute affiliated to Rashtrasant Tukadoji Maharaj Nagpur University)



DEPARTMENT OF INFORMATION TECHNOLOGY

```
plt.xlabel("Day")
plt.ylabel("Total Bill")
plt.grid(True, linestyle='--', alpha=0.5)
plt.show()

# -----
# Violin Plot of Total Bill by Sex
# -----

plt.figure(figsize=(6, 4))
sns.violinplot(data=tips, x="sex", y="total_bill", palette="muted")
plt.title("Violin Plot of Total Bill by Gender")
plt.xlabel("Gender")
plt.ylabel("Total Bill")
plt.grid(True, linestyle='--', alpha=0.5)
plt.show()
```



DEPARTMENT OF INFORMATION TECHNOLOGY

Expected Output:

1. **Histogram** – Displays the frequency distribution of the `total_bill` column with a smooth KDE line overlay.
2. **Box Plot** – Shows median, quartiles, and outliers of `total_bill` grouped by `day` (Thu, Fri, Sat, Sun).
3. **Violin Plot** – Visualizes both the box plot and the distribution (KDE) of `total_bill` grouped by `sex`.

These plots help analyze central tendency, spread, and variation between groups in a dataset.

Viva Questions:

1. **What is the main difference between a box plot and a violin plot?**
 - A box plot shows summary statistics (median, quartiles), while a violin plot shows the full distribution with KDE.
2. **Why do we use a histogram?**
 - To understand the frequency distribution and shape (e.g., normal, skewed) of a numerical variable.
3. **What is KDE in a histogram?**
 - KDE stands for Kernel Density Estimate, which is a smooth curve representing the data distribution.
4. **How do you detect outliers using a box plot?**
 - Outliers are points that lie outside the whiskers (typically $1.5 \times \text{IQR}$ from the quartiles).



DEPARTMENT OF INFORMATION TECHNOLOGY

Aim:

To create advanced data visualizations such as heatmaps and pair plots using the Seaborn library to explore and analyze relationships between multiple variables in a dataset.

Theory:

Visualizing relationships between multiple features in a dataset helps in identifying correlations, clusters, and outliers. Seaborn offers two powerful tools for this:

◆ Heatmap:

- A color-coded matrix that displays the correlation coefficients between numerical variables.
- Values close to **+1** or **-1** indicate strong relationships.

◆ Pair Plot:

- Shows pairwise relationships between all numeric variables in the dataset using scatter plots and histograms.
- Useful for detecting patterns, correlations, and clusters.

Use Cases:

- Identify strongly correlated variables.
- Explore how variables relate to each other.



DEPARTMENT OF INFORMATION TECHNOLOGY

- Detect multicollinearity before applying machine learning models.

Code:

```
import seaborn as sns
```

```
import matplotlib.pyplot as plt
```

```
# Load example dataset
```

```
iris = sns.load_dataset("iris")
```

```
# -----
```

```
# Heatmap of Correlation Matrix
```

```
# -----
```

```
plt.figure(figsize=(8, 6))
```

```
correlation_matrix = iris.corr()
```

```
sns.heatmap(correlation_matrix, annot=True, cmap="coolwarm", linewidths=0.5)
```

```
plt.title("Heatmap - Correlation Matrix (Iris Dataset)")
```

```
plt.show()
```

```
# -----
```

```
# Pair Plot
```



DEPARTMENT OF INFORMATION TECHNOLOGY

```
# -----  
  
sns.pairplot(iris, hue="species", palette="Set1")  
  
plt.suptitle("Pair Plot - Iris Dataset", y=1.02)  
  
plt.show()
```

Expected Output:

1. Heatmap – A grid showing correlation values between features like **sepal_length**, **petal_length**, etc. The color intensity indicates the strength and direction of the relationship.
2. Pair Plot – A matrix of scatter plots showing how each feature correlates with others, with color coding by species (**setosa**, **versicolor**, **virginica**).

These plots help you:

- Detect linear relationships (e.g., **petal_length** vs **petal_width**).
- Visually separate classes (e.g., how species differ based on measurements).



DEPARTMENT OF INFORMATION TECHNOLOGY

Viva Questions:

1. What does a heatmap represent in data visualization?
 - A heatmap displays the correlation coefficients between numerical variables using color intensity.
2. What is the use of a pair plot in data analysis?
 - It visualizes pairwise relationships between multiple numerical variables and can reveal clusters and correlations.
3. What does a correlation value of **1**, **-1**, and **0** signify?
 - **1** = perfect positive correlation, **-1** = perfect negative correlation, **0** = no correlation.
4. Why is the **hue** parameter used in Seaborn's **pairplot()**?
 - To color-code the plots based on a categorical variable (e.g., species), helping to identify group-wise trends.



DEPARTMENT OF INFORMATION TECHNOLOGY

Aim:

To design and implement an interactive dashboard using Plotly and Dash in Python for dynamic data visualization and real-time interaction.

Theory:

♦ What is Dash?

- Dash is a Python framework built by Plotly for building web-based interactive dashboards.
- Combines Flask (for backend), Plotly.js (for visualizations), and React.js (for front-end interactivity).
- Allows data scientists to build interactive web apps with pure Python.

♦ What is Plotly?

- Plotly is a graphing library that enables the creation of interactive and web-ready visualizations like line charts, bar graphs, scatter plots, etc.

Advantages of Dash:



DEPARTMENT OF INFORMATION TECHNOLOGY

-
- No need to learn HTML, CSS, or JavaScript.
 - Fully interactive and responsive dashboards.
 - Good for displaying real-time data or user-driven insights.

Code:

This simple dashboard allows the user to select a species from the Iris dataset and view a scatter plot of **sepal_length** vs **sepal_width**.

✓ Step 1: Install Dash
pip install dash plotly

✓ Step 2: Full Python Code

```
# app.py
import dash
from dash import dcc, html
from dash.dependencies import Input, Output
import plotly.express as px
import seaborn as sns
import pandas as pd

# Load dataset
df = sns.load_dataset('iris')

# Initialize Dash app
app = dash.Dash(__name__)

# Layout of the dashboard
app.layout = html.Div([
    html.H1("Iris Species Dashboard", style={'textAlign': 'center'}),

    html.Label("Select Species:"),

    dcc.Dropdown(
        id='species-dropdown',
        options=[{'label': species, 'value': species} for species in df['species'].unique()],
        value='setosa'
```




DEPARTMENT OF INFORMATION TECHNOLOGY

```
),  
  
    dcc.Graph(id='scatter-plot')  
)  
  
# Callback to update graph  
@app.callback(  
    Output('scatter-plot', 'figure'),  
    Input('species-dropdown', 'value')  
)  
def update_graph(selected_species):  
    filtered_df = df[df['species'] == selected_species]  
    fig = px.scatter(filtered_df,  
                     x='sepal_length',  
                     y='sepal_width',  
                     color='species',  
                     title=f"Sepal Length vs Sepal Width for {selected_species.capitalize()}",  
                     labels={'sepal_length': 'Sepal Length', 'sepal_width': 'Sepal Width'})  
    return fig  
  
# Run the app  
if __name__ == '__main__':  
    app.run_server(debug=True)
```

Expected Output:

- A web-based dashboard hosted at <http://127.0.0.1:8050/>
- Dropdown menu to select a species (**setosa**, **versicolor**, **virginica**)
- A live interactive scatter plot that updates based on the selected species
- Hovering over points shows exact values
- Users can zoom, pan, and export the plot as an image



DEPARTMENT OF INFORMATION TECHNOLOGY

Viva Questions:

1. What is the role of Plotly in the Dash framework?
➤ Plotly is used to create interactive charts inside the Dash dashboard.
2. How does Dash handle interactivity in a dashboard?
➤ Dash uses callbacks to update components (like graphs) based on user input (like dropdowns or sliders).
3. What is a callback function in Dash?
➤ A callback function updates dashboard components dynamically based on inputs using decorators (**@app.callback**).
4. Can Dash apps be deployed to the web? How?
➤ Yes, Dash apps can be deployed using platforms like Heroku, Render, or Dash Enterprise.

Optional Enhancements:

- Add more dropdowns or sliders for different variables
- Include multiple charts on the same page
- Link the dashboard with a live API or real-time data



Lokmanya Tilak Jankalyan Shikshan Sanstha's
PRIYADARSHINI COLLEGE OF ENGINEERING, NAGPUR
(An Autonomous Institute affiliated to Rashtrasant Tukadoji Maharaj Nagpur University)



DEPARTMENT OF INFORMATION TECHNOLOGY
