PRATICAL-7

AIM: (i). Implementation of Python Libraries for ML application such as Pandas and Matplotlib.

◉ SOURCE CODE :

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt

# (a) Create a Series using pandas and display it
data = [10, 20, 30, 40, 50]
series = pd.Series(data)
print(series)
```

OUTPUT :-

```
[Running] python -u "c:\Users\Ankit raj\Downloads\try.py"
0    10
1    20
2    30
3    40
4    50
dtype: int64
```

```
# (b) Access the index and values of the our Series
data = [10, 20, 30, 40, 50]
series = pd.Series(data)
print("Index:", series.index)
print("Values:", series.values)
```

OUTPUT :-

```
[Running] python -u "c:\Users\Ankit raj\Downloads\try.py"
Index: RangeIndex(start=0, stop=5, step=1)
Values: [10 20 30 40 50]
```

```
# (c) Compare an array using NumPy with a Series using pandas
array = np.array([10, 20, 30, 40, 50])
series = pd.Series([10, 20, 30, 40, 50])


print("NumPy Array:", array)
print("\nPandas Series:\n", series)
```

OUTPUT :-

```
[Running] python -u "c:\Users\Ankit raj\Downloads\try.py"
NumPy Array: [10 20 30 40 50]

Pandas Series:
 0    10
1    20
2    30
3    40
4    50
dtype: int64
```

```
# (d) Define Series objects with individual indices
data = [100, 200, 300, 400, 500]
index_labels = ['a', 'b', 'c', 'd', 'e']  # Custom indices
series = pd.Series(data, index=index_labels)
print(series)
```

OUTPUT :-

```
[Running] python -u "c:\Users\Ankit raj\Downloads\try.py"
a    100
b    200
c    300
d    400
e    500
dtype: int64
```

```
# (e) Access single value of a Series
data = [100, 200, 300, 400, 500]
index_labels = ['a', 'b', 'c', 'd', 'e']
series = pd.Series(data, index=index_labels)
print("Value at index 'c':", series.loc['c'])
print("Value at position 2:", series.iloc[2])
```

OUTPUT :-

```
[Running] python -u "c:\Users\Ankit raj\Downloads\try.py"
Value at index 'c': 300
Value at position 2: 300
```

```
# (f) Load datasets in a DataFrame variable using pandas
df = pd.read_excel(r"C:\Users\saifa\Downloads\Saif Data.xlsx")
print("Iris Dataset (First 5 rows):")
print(df.head())
print("Has Been Printed")
```

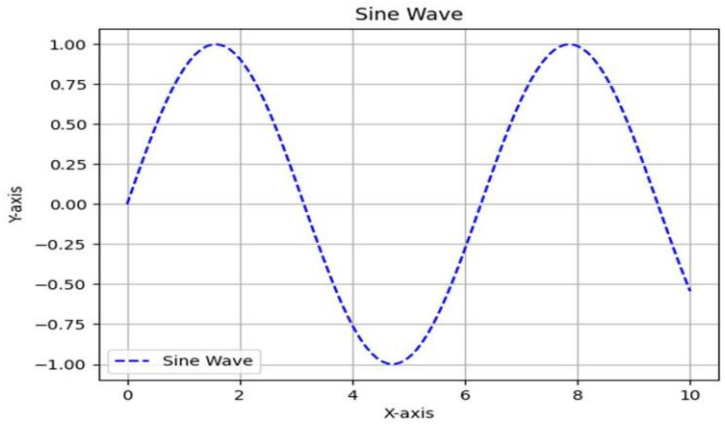
OUTPUT :-

```
Iris Dataset (First 5 rows):
   Sr. No.  Admission No            Name
0        1      11232503           AJIT SAH
1        2      11232506  AASHISH KUMAR MEHTA
2        3      11232507  AASHUTOSH KUMAR SAH
3        4      11232508          ABHAY SINGH
4        5      11232509         ABHINAV GARG
Has Been Printed
```

```
# (g) Usage of different methods in Matplotlib

x = np.linspace(0, 10, 100)
y = np.sin(x)
plt.plot(x, y, label="Sine Wave", color='b', linestyle='--')
plt.title("Sine Wave")
plt.xlabel("X-axis")
plt.ylabel("Y-axis")
plt.grid(True)
plt.legend()
plt.show()
```

OUTPUT :-



(ii) a) Creation and Loading different types of datasets in Python using the required libraries.

```
import pandas as pd
import numpy as np
from scipy import stats
from sklearn.preprocessing import MinMaxScaler
import os

# Step 1: Create and Save CSV File
csv_filename = "sample_dataset.csv"

# Creating sample data
data = {
    'Name': ['Alice', 'Bob', 'Charlie', 'David', 'Eve'],
    'Age': [25, 30, 35, 40, 28],
    'Salary': [50000, 60000, 70000, 80000, 55000]
}

# Creating DataFrame
df = pd.DataFrame(data)

# Save dataset as CSV if not exists
if not os.path.exists(csv_filename):
    df.to_csv(csv_filename,  index=False)
    print(f"CSV file '{csv_filename}' has been created successfully.")
else:
    print(f"CSV file '{csv_filename}' already exists.")

# Step 2: Load CSV Dataset
df_csv = pd.read_csv(csv_filename)
print("\nLoaded CSV Dataset:\n", df_csv)

# Step 3: Compute Mean, Median, Mode, Variance, Standard Deviation
mean_value = df_csv['Salary'].mean()
median_value = df_csv['Salary'].median()
```

```
# Fix mode issue for newer SciPy versions
mode_result = stats.mode(df_csv['Salary'], keepdims=True)
mode_value = mode_result.mode[0] if mode_result.mode.size > 0 else None

variance_value = df_csv['Salary'].var()  # Sample variance (pandas default)
std_dev_value = df_csv['Salary'].std()   # Sample standard deviation (pandas default)

print(f"\nMean Salary: {mean_value}")
print(f"Median Salary: {median_value}")
print(f"Mode Salary: {mode_value}")
print(f"Variance of Salary: {variance_value}")
print(f"Standard Deviation of Salary: {std_dev_value}")

# Step 4: Data Preprocessing Techniques

# 1. Reshaping the Data (For demonstration, creating a NumPy array)
data_array = np.array([[1, 2, 3], [4, 5, 6]])
reshaped_data = data_array.reshape(3, 2)
print("\nReshaped Data:\n", reshaped_data)

# 2. Filtering the Data (Filtering Age > 30)
filtered_df = df_csv[df_csv['Age'] > 30]
print("\nFiltered Data (Age > 30):\n", filtered_df)

# 3. Merging Data
data2 = {'Name': ['Alice', 'Bob', 'Charlie', 'David', 'Eve'],
        'Department': ['HR', 'IT', 'Finance', 'Marketing', 'Sales']}
df2 = pd.DataFrame(data2)
merged_df = pd.merge(df_csv, df2, on='Name')
print("\nMerged Dataset:\n", merged_df)

# 4. Handling Missing Values (Introduce and Fix Missing Values)
df_csv.loc[2, 'Salary'] = np.nan  # Introduce NaN in Salary column
```

```
df_csv['Salary'] = df_csv['Salary'].fillna(df_csv['Salary'].mean())  # Fix missing values
print("\nDataset after Handling Missing Values:\n", df_csv)

# 5. Feature Normalization (Min-Max Normalization)
scaler = MinMaxScaler()
df_csv['Salary_Normalized'] = scaler.fit_transform(df_csv[['Salary']])
print("\nMin-Max Normalized Salary:\n", df_csv):
```

OUTPUT :

CSV file 'sample_dataset.csv' already exists.

Loaded CSV Dataset:

	Name	Age	Salary
0	Alice	25	50000
1	Bob	30	60000
2	Charlie	35	70000
3	David	40	80000
4	Eve	28	55000

Loaded Iris Dataset from sklearn:

	sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)
0	5.1	3.5	1.4	0.2
1	4.9	3.0	1.4	0.2
2	4.7	3.2	1.3	0.2
3	4.6	3.1	1.5	0.2
4	5.0	3.6	1.4	0.2

Mean Salary: 63000.0

Median Salary: 60000.0

Mode Salary: 50000

Variance of Salary: 145000000.0

Standard Deviation of Salary: 12041.594578792296

Reshaped Data:

```
[[1 2]
 [3 4]
 [5 6]]
```

Filtered Data (Age > 30):

	Name	Age	Salary
2	Charlie	35	70000
3	David	40	80000

Merged Dataset:

	Name	Age	Salary	Department
0	Alice	25	50000	HR
1	Bob	30	60000	IT
2	Charlie	35	70000	Finance
3	David	40	80000	Marketing
4	Eve	28	55000	Sales

Dataset after Handling Missing Values:

	Name	Age	Salary
0	Alice	25	50000.0
1	Bob	30	60000.0
2	Charlie	35	61250.0
3	David	40	80000.0
4	Eve	28	55000.0

Min-Max Normalized Salary:

	Name	Age	Salary	Salary_Normalized
0	Alice	25	50000.0	0.000000
1	Bob	30	60000.0	0.333333
2	Charlie	35	61250.0	0.375000
3	David	40	80000.0	1.000000
4	Eve	28	55000.0	0.166667