Ex.No.2 Perform data manipulation using NumPy and Pandas and, data visualization
DATE : using matplotlib.

AIM:
To perform data manipulation using numpy and pandas and data visualization using matplotlib.

Program:

A. Data Manipulation using Numpy

NumPy stands for Numerical Python. NumPy is a python library used for working with arrays.

array(): Used to create numpy ndarray
copy(): It is a new array and it owns the data and any changes made to the copy will not affect original array
view():View is just a view of the original array.The view does not own the data and any changes made to the view will affect the original array reshape() : Used to change the dimensions of the array concatenate(): Used to join two or more arrays into single array split(): Splitting is reverse operation of Joining. Array and number of splits are passed as argument.
where(): search an array for a certain value, and return the indexes that get a match
sort(): Sorting means putting elements in an ordered sequence. random: Module to work with random numbers

1. Program to create array and perform reshape:

```
import numpy as np
arr = np.array([12, 24, 32, 41, 40, 45])
reshaped_arr = arr.reshape(2, 3)
print("Original Array:")
print(arr)
print("Reshaped Array:")
print(reshaped_arr)
```

Output:

```
Original Array:
[12 24 32 41 40 45]
Reshaped Array:
[[12 24 32]
 [41 40 45]]
```

2. Program to demonstrate copy and view

```
import numpy as np
arr = np.array([11, 24, 32, 41, 40, 45])
copy_arr = arr.copy()
```

```
view_arr = arr.view()
arr[0] = 100
print("Original Array:", arr)
print("Copied Array:", copy_arr)
print("View Array:", view_arr)
```

Output:

```
Original Array: [100 24 32 41 40 45]
Copied Array: [11 24 32 41 30 45]
View Array: [100 24 32 41 30 45]
```

3. Program to demonstrate concatenate and split

```
import numpy as np
arr1 = np.array([12, 21, 30])
arr2 = np.array([45, 40, 62])
concat_arr = np.concatenate((arr1, arr2))
print("Concatenated Array:")
print(concat_arr)
split_arr = np.split(concat_arr, 2)
print("Split Arrays:")
print(split_arr)
```

Output:

```
Concatenated Array:
[12 21 30 45 40 62]
Split Arrays:
[array([12, 21, 30]), array([45, 40, 62])]
```

4. Program to demonstrate sorting and searching

```
import numpy as np
arr = np.array([45, 22, 67, 11, 89])
sorted_arr = np.sort(arr)
print("Original Array:", arr)
print("Sorted Array:", sorted_arr)
arr2 = np.array([5, 15, 25, 35, 45, 55])
index = np.where(arr2 == 45)
print("Array:", arr2)
print("Index of 45:", index)
```

Output:

```
Original Array: [45 22 67 11 89]
Sorted Array: [11 22 45 67 89]
Array: [ 5 15 25 35 45 55]
Index of 45: (array([4]),)
```

B. Data Manipulation using Pandas

Pandas is a python library that allow us to analyze big data and make conclusions based on statistical theories.

read_csv(dataset name): Used to load dataset dropna(): Used to remove rows or columns with missing values fillna(): Used to replace the NaN values with other values mean(): Used to find the mean of the values for the requested axis sum(): Function return the sum of the values for the requested axis count(): is used to count the no. of non-NA/null observations across the given axis.

1. Program for handling missing data:

```
import numpy as np
import pandas as pd

dframe = pd.read_csv('Data.csv')
print(dframe)

dframe.dropna(inplace = True)
print(dframe)

dframe.fillna(value = dframe.mean(), inplace = True)
print(dframe)
```

Output:

	a	b	c
0	34	50.0	0.0
1	25	12.0	NaN
2	23	NaN	NaN

	a	b	c
0	34	50.0	0.0

	a	b	c
0	34	50.0	0.0
1	25	12.0	0.0
2	23	31.0	0.0

2. Program for merge and join of data frame using panda:

```
import numpy as np
import pandas as pd

left = pd.DataFrame({
   'Key': ['J0', 'J1', 'J2', 'J3'],'A': ['X1', 'X2', 'X3', 'X4'],'B': ['Y1', 'Y2', 'Y3', 'Y4']})
right = pd.DataFrame({
   'Key': ['K0', 'K1', 'K2', 'K3'], 'C': ['C0', 'C1', 'C2', 'C3'],'D': ['D0', 'D1', 'D2', 'D3']})

# Merging the dataframes

merged_df = pd.merge(left, right, on='Key')

# Joining the dataframes

left_indexed = left.set_index('Key')

right_indexed = right.set_index('Key')

joined_df = left_indexed.join(right_indexed)
```

Output:

	Key	A	B
0	J0	X1	Y1
1	J1	X2	Y2
2	J2	X3	Y3
3	J3	X4	Y4

	Key	C	D
0	K0	C0	D0
1	K1	C1	D1
2	K2	C2	D2
3	K3	C3	D3

	Key	A	B	C	D
0	K0	A0	B0	C0	D0
1	K1	A1	B1	C1	D1
2	K2	A2	B2	C2	D2
3	K3	A3	B3	C3	D3

3. Program for aggregation in panda:

```
import numpy as np
import pandas as pd
df = pd.read_csv('data.csv')
print(df)
print(df.mean())
print(df.count())
print(df.sum())
```

Output:

Mean:

a 27.3333

b 31.00000

c 0.000000

dtype: float64

Count:

a 3

b 2

c 1

dtype: int64

Sum:

a 82.0

b 62.0

c 0.0

dtype: float64

C. Data Visualization using Matplotlib

Matplotlib is a multi-platform data visualization library built on NumPy arrays. Matplotlib consists of several plots like line, bar, scatter, histogram etc.

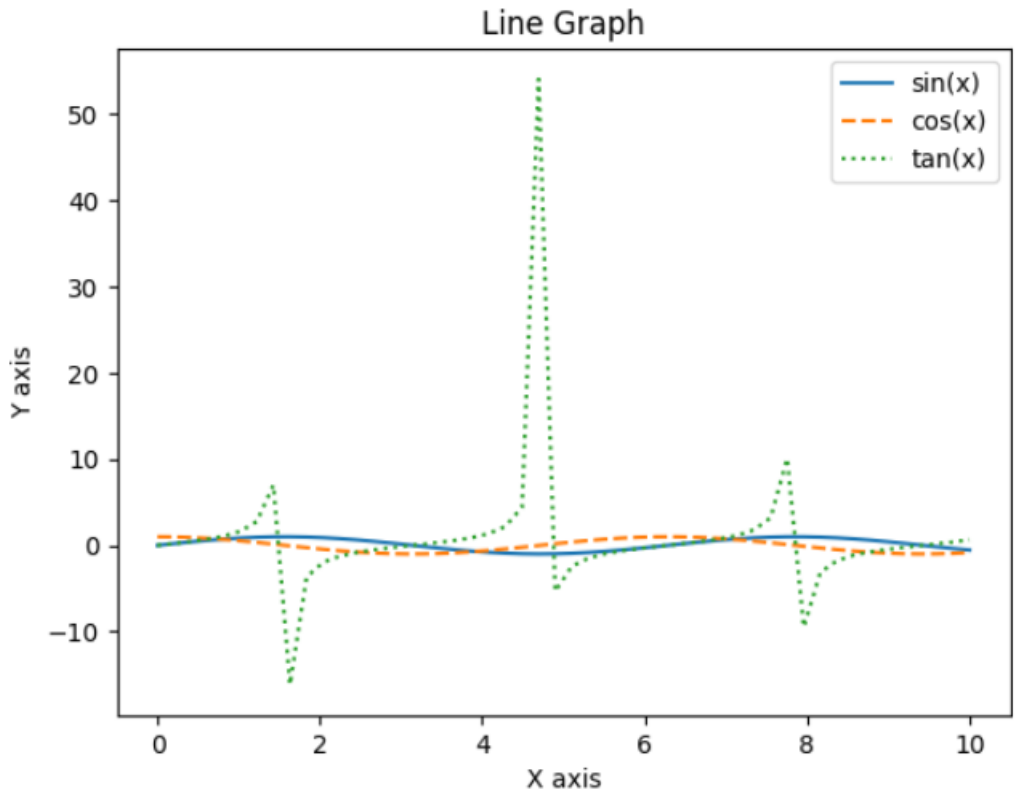
figure(): Thought of as a single container that contains all the objects representing axes, graphics, text, and labels axes(): Bounding box with ticks and labels, which will eventually contain the plot elements that make up our visualization. xlim(), ylim(): Used to set x and y limit for axis
axis[xmin,xmax,ymin,ymax]: Used to set x and y limit for axis
xlabel(), ylabel(): Used to set label for axis legend():
Used to set label for multiple plot lines in graph
plt.axis('equal'): Equal splitting of x and y limit values
plt.axis('tight'): Default spacing left between values of x and y limits
plt.savefig(): Used to save the output graph to specified format

1. Line Drawing

Program:

```
import matplotlib.pyplot as plt
import numpy as np
x = np.linspace(0, 10, 50)
y1 = np.sin(x)
y2 = np.cos(x)
y3 = np.tan(x)
plt.plot(x, y1, linestyle='solid',
label='sin(x)')
plt.plot(x, y2, linestyle='dashed',
label='cos(x)')
plt.plot(x, y3, linestyle='dotted',
label='tan(x)')
plt.xlabel("X axis")
plt.ylabel("Y axis")
plt.title("Line Graph")
plt.legend()
plt.show()
```

Output:

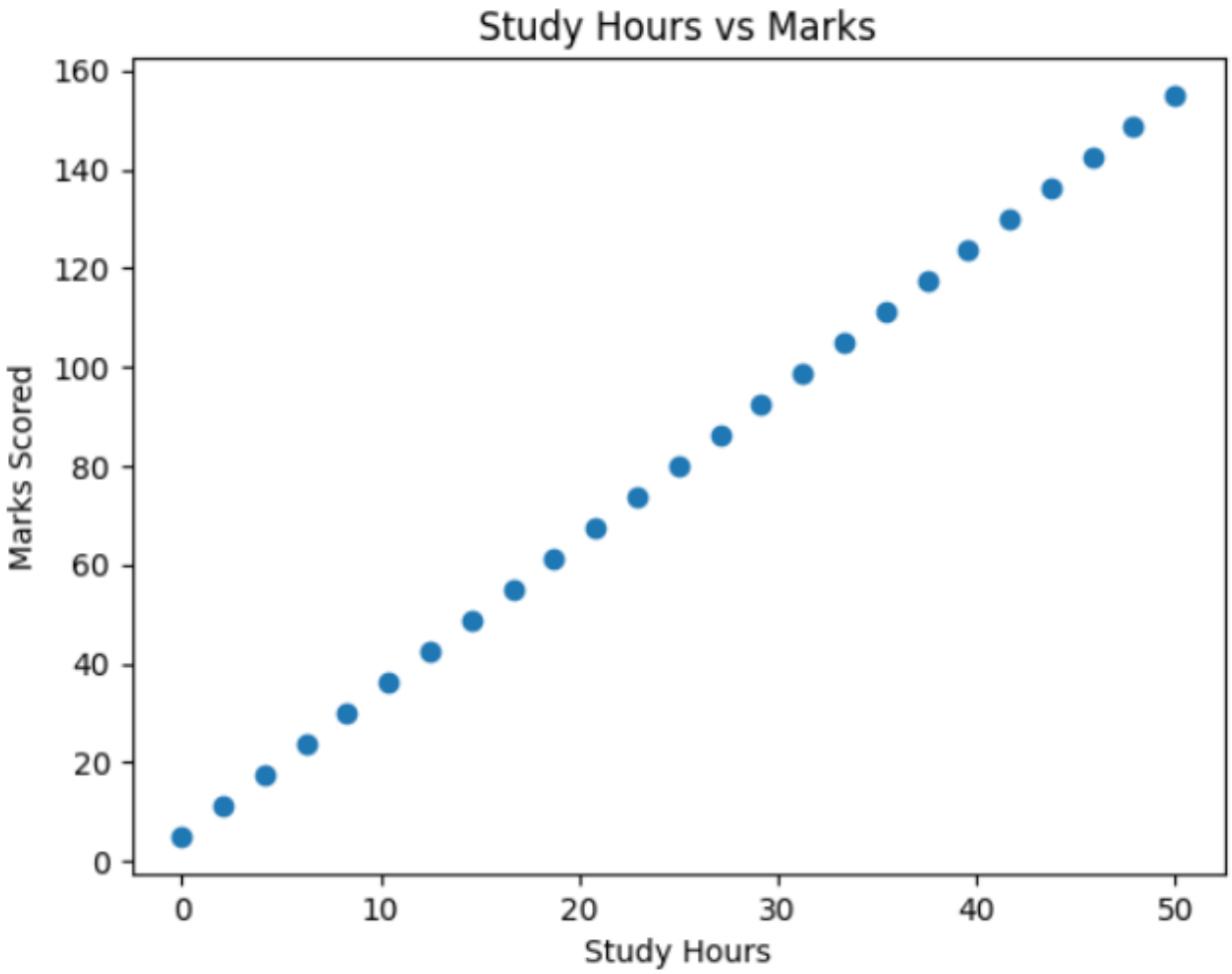


2. Scatter Plot

Program:

```
import matplotlib.pyplot as plt
import numpy as np
x = np.linspace(0, 50, 25)
y = x * 3 + 5
plt.scatter(x, y)
plt.xlabel("Study Hours")
plt.ylabel("Marks Scored")
plt.title("Study Hours vs Marks")
plt.show()
```

Output:

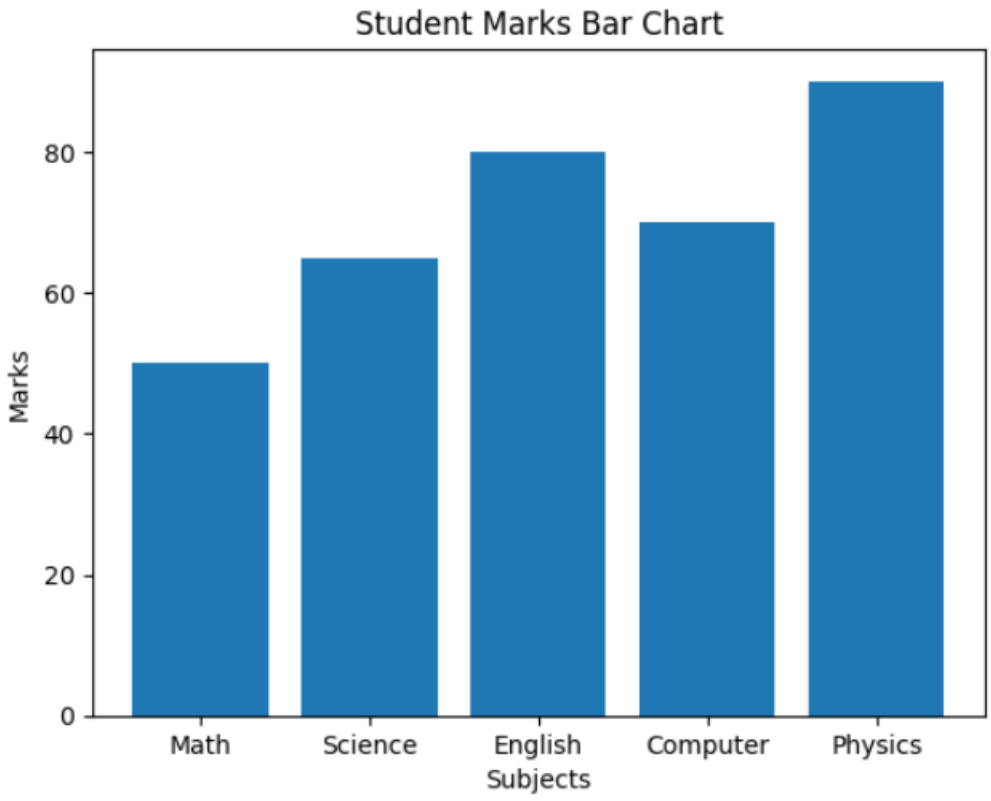


3. Bar chart

Program:

```
import matplotlib.pyplot as plt
left = [1, 2, 3, 4, 5]
height = [50, 65, 80, 70, 90]
labels = ['Math', 'Science', 'English', 'Computer', 'Physics']
plt.bar(left, height, tick_label=labels)
plt.xlabel("Subjects")
plt.ylabel("Marks")
plt.title("Student Marks Bar Chart")
plt.show()
```

Output:



MARK ALLOCATION	
Conduct of Experiment(30)	
Record Observation (20)	
Viva (10)	
Total (60)	
Signature of the Faculty with Date	

Result:

Data manipulation using numpy , pandas and data visualization using matplotlib are successfully performed.