

**Experiment No. 5**

**Title: Matplotlib for Data Visualization**

### Batch:A3 Roll No:16010421075 Experiment No.:5



Aim: To use Pandas in built visualization and Matplotlib visualization to perform exploratory data analysis

**Resources needed:** Python IDE

### Theory:

**Pandas Built-in Data Visualization**

Pandas have got built-in capabilities for data visualization. It's built-off of matplotlib, but it baked into pandas for easier usage!

import numpy as np import pandas as pd import matplotlib as mp

%matplotlib inline

There are some fake data csv files you can read in as dataframes:

df1 **=** pd.read\_csv('df1',index\_col**=**0) df1.head().

#Bar plot for df1 can be plot using df2.plot.bar(stacked=True)

#ploting histogram of only one column with 50 bins are setting bar width less than 1. df1['A'].plot.hist(bins=50, rwidth=0.8)

#line plot in pandas

df1.plot.line()

#scatter plot with color and colormaps df1.plot.scatter()

#boxplot of data frame will helps us to spot the outliers(mild and extream both) df2.plot.box()

#density plots- to explore symmetric or assymetric nature of your dataset. df2.plot.density()

### Mathplotlib for Data Visualization

Matplotlib is the "grandfather" library of data visualization with Python. It was created by John Hunter. He created it to try to replicate MatLab's (another programming language) plotting

capabilities in Python. So if you happen to be familiar with matlab, matplotlib will feel natural to you.



It is an excellent 2D and 3D graphics library for generating scientific figures. Some of the major Pros of Matplotlib are:

* Generally easy to get started for simple plots
* Support for custom labels and texts
* Great control of every element in a figure
* High-quality output in many formats
* Very customizable in general

Matplotlib allows you to create reproducible figures programmatically

### Installation

You'll need to install matplotlib first with either: conda install matplotlib

or pip install matplotlib

### Importing

Import the matplotlib.pyplot

### Basic Matplotlib Commands

We can create a very simple line plot using the following plt.plot(x, y, 'r') # 'r' is the color red

#setting x and y axis labels, title of plot plt.xlabel('X Axis Title Here') plt.ylabel('Y Axis Title Here') plt.title('StringTitlehere')

### Using subplot a grid of plots can be created as shown below. Also we can set marker and linestyle along with color of plot.

plt.subplot(1,2,1) plt.plot(x, y, 'r.--') # plt.subplot(1,2,2) plt.plot(y, x, 'g\*-.');

### Matplotlib’s object oriented api:

The main idea in using the more formal Object Oriented method is to create figure objects and then just call methods or attributes off of that object. This approach is nicer when dealing with a canvas that has multiple plots on it.

# Create Figure object to represent an empty canvas fig = plt.figure()

# Add set of axes to figure(manually)

axes = fig.add\_axes([0.1, 0.1, 0.8, 0.8]) # left, bottom, width, height (range 0 to 1)

# Plot on that set of axes axes.plot(x, y, 'b')

axes.set\_xlabel('Set X Label') # Notice the use of set\_ to begin methods axes.set\_ylabel('Set y Label')

axes.set\_title('Set Title') axes.set\_legend(loc=1)



### Figure size, aspect ratio and DPI

Matplotlib allows the aspect ratio, DPI and figure size to be specified when the Figure object is created. You can use the figsize and dpi keyword arguments.

* figsize is a tuple of the width and height of the figure in inches
* dpi is the dots-per-inch (pixel per inch).

For example:

fig **=** plt.figure(figsize**=**(8,4), dpi**=**100)

### Activities:

(use pandas and matplotlib for following activities)

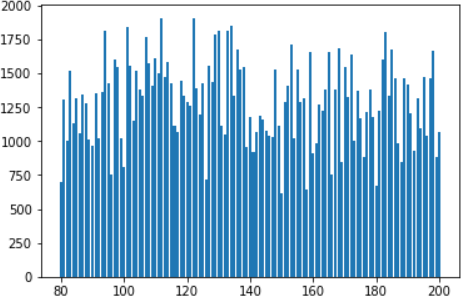
1. Download data set with atleast 1500 rows and 10-20 columns(numeric and non numeric) from valid data sources
2. Visualization to summarize your data set(density, frequency plot)
3. Measures of central tenancy of data set (mean, median etc)
4. Determining presence of outliers in your dataset(boxplot)
5. Correlation of attributes in your dataset( scatter plot and line plot on 2-3 pairs which are correlated)
6. Comparison of data ploted on same scale using barplot( 3 plots for 3 different columns pairs)
7. Use different, colors, styles, markers,marker with different size, legends, labels, colormaps dpi, figsize etc in the plot
8. Save these plots
9. Write down your comment on each of these plots
10. place legends at appropriate location on the plot
11. Write down observation for your dataset for each of above listed task of analysis.

**Result: (script and output)**

# BARGRAPH

from matplotlib import pyplot as plt x=df['mobile\_wt'].tolist() y=df['px\_height'].tolist() plt.bar(x,y)

plt.show()

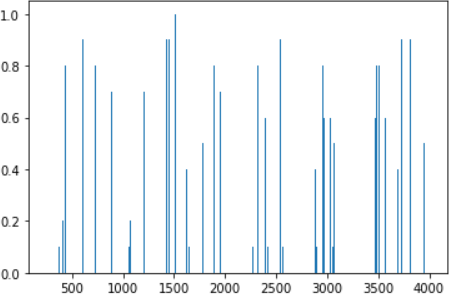


x=df['ram'].tolist()

y=df['m\_dep'].tolist()

z=df['blue'].tolist()

plt.bar(x,y,z) plt.show()



* **HISTOGRAM** y=df['talk\_time'].tolist() plt.hist(y)

plt.show()

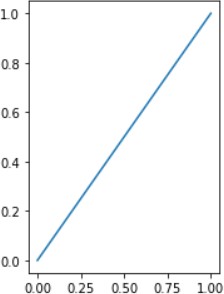
## LINE GRAPH

import io

import pandas as pd import numpy as np

from matplotlib import pyplot as plt x=df['dual\_sim']

y=df['dual\_sim'] plt.subplot(1,2,1) plt.plot(x,y) plt.show()

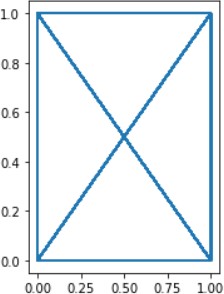


import io

import pandas as pd import numpy as np

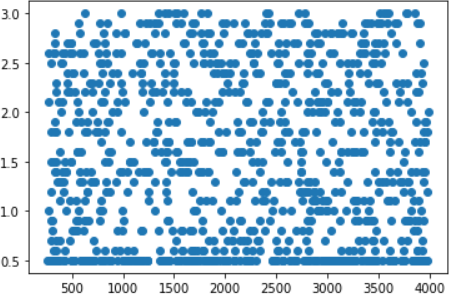
from matplotlib import pyplot as plt x=df['dual\_sim']

y=df['blue'] plt.subplot(1,2,1) plt.plot(x,y) plt.show()

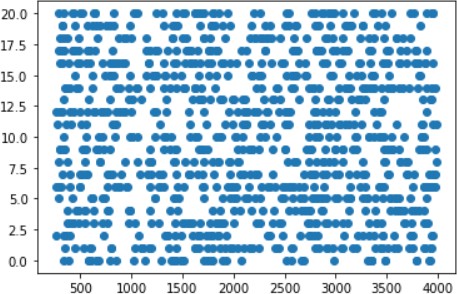


# SCATTER PLOT

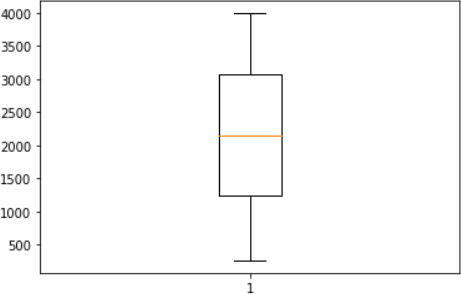
x=df['ram'].tolist() y=df['clock\_speed'].tolist() plt.scatter(x,y)

plt.show()

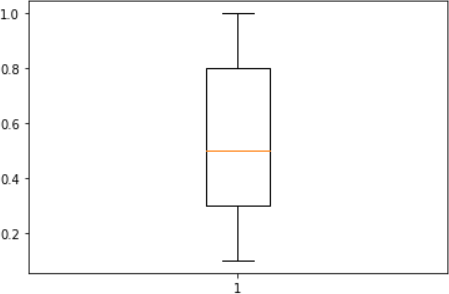
x=df['ram'].tolist()

y=df['pc'].tolist() plt.scatter(x,y) plt.show()

# BOXPLOT

x=df['ram'].tolist() plt.boxplot(x) plt.show()

y=df['m\_dep'].tolist() plt.boxplot(y) plt.show()



# Measuring Central Tendency

import io

import pandas as pd import numpy as np df=pd.read\_csv("test.csv") x=df["ram"]

x.mean()



import io

import pandas as pd import numpy as np df=pd.read\_csv("test.csv") x=df["blue"]

x.median()

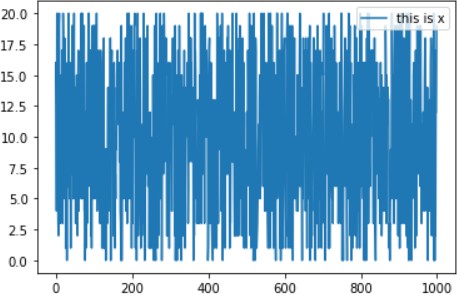


import io

import pandas as pd import numpy as np df=pd.read\_csv("test.csv") x=df["int\_memory"] x.mode()



* **LEGEND** y=df['pc'].tolist() plt.plot(y)

plt.legend(["this is x","this is y"],loc='upper right') plt.show()

# USING DIFF COLOURS

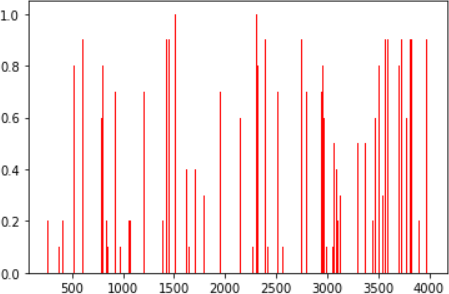
import io

import pandas as pd import numpy as np

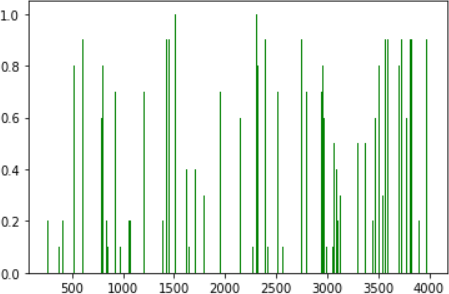
from matplotlib import pyplot as plt x=df['ram'].tolist()

y=df['m\_dep'].tolist()

plt.bar(x,y,’r’) plt.show()



x=df['ram'].tolist()

y=df['m\_dep'].tolist() plt.bar(x,y,color='green') plt.show()

## Labeling the graph

import io

import pandas as pd import numpy as np

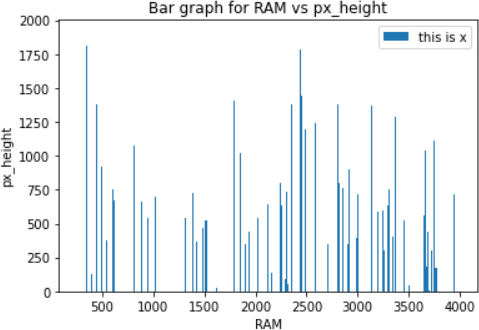
from matplotlib import pyplot as plt x=df['ram'].tolist() y=df['px\_height'].tolist() plt.bar(x,y)

plt.xlabel('RAM') plt.ylabel('px\_height')

plt.title('Bar graph for RAM vs px\_height') plt.legend(["this is x","this is y"],loc='upper right') plt.subplot

plt.show()





# Figure size, style and DPI

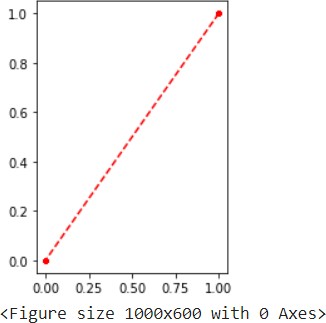
import io

import pandas as pd import numpy as np

from matplotlib import pyplot as plt x=df['dual\_sim']

plt.subplot(1,2,1) plt.plot(x, y, 'r.--')

fig = plt.figure(figsize=(10,6), dpi=100) plt.show()



### Outcomes:

CO2. Inculcate the knowledge of python libraries like numpy, pandas, matplotlib for scientific- computing and data visualization.

**Conclusion:** (Conclusion to be based on the objectives and outcomes achieved)

Through this experiment we successfully used Matplotlib visualization to perform exploratory

(A Constituent ar University)

College of Somaiya Vidyavi

h

data analysis.

### References:

* 1. <https://pandas.pydata.org/pandas-docs/stable/user_guide/visualization.html>
  2. Daniel Arbuckle, Learning Python Testing, Packt Publishing, 1st Edition, 2014
  3. Wesly J Chun, Core Python Applications Programming, O’Reilly, 3rd Edition, 2015
  4. Wes McKinney, Python for Data Analysis, O’Reilly, 1st Edition, 2017
  5. Albert Lukaszewsk, MySQL for Python, Packt Publishing, 1st Edition, 2010
  6. Eric Chou, Mastering Python Networking, Packt Publishing, 2nd Edition, 2017

(A Constituent College of Somaiya Vidyavihar University)