Experiment No. 5

Title: Matplotlib for Data Visualization

Batch:A2 **Roll No:16010421063** **Experiment No.:5**

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

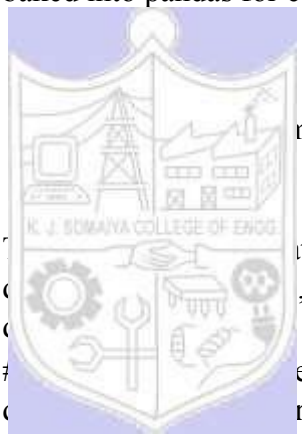
---

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!



np

ta csv files you can read in as dataframes:
,index_col=0)

e plot using
rue)

#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

atplotlib first with either:

lotlib

yplot

mands

mple 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')
')
```

atio and DPI

spect ratio, DPI and figure size to be specified when the Figure object is
e figsize and dpi keyword arguments.

e of the width and height of the figure in inches
per-inch (pixel per inch).

```
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)

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

[9] ✓ 0.2s

```
df=pd.read_csv("data.csv")
df
```

[10] ✓ 0.4s

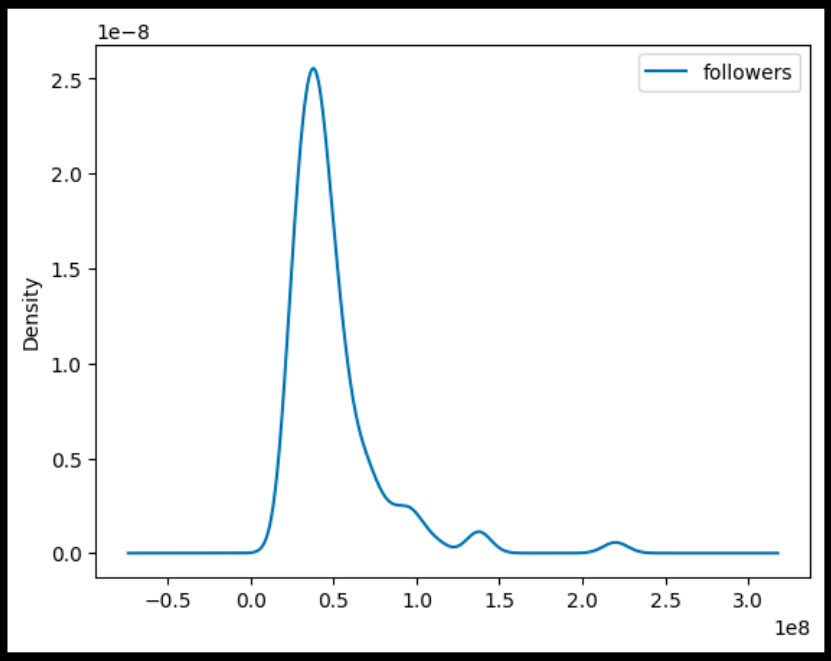
	Country	Channel Name	Category	Main Video Category	username	followers	Main topic	More t
0	IN	T-Series	Gaming & Apps	Music	T-Series	220000000	Music of Asia	Entertainment,Mu Asia,Music,M
1	US	ABCkidTV - Nursery Rhymes	Gaming & Apps	Education	ABCkidTV - Nursery Rhymes	138000000	Movies	Entertainment,Music,M
2	IN	SET India	Gaming & Apps	Shows	SET India	137000000	Movies	Entertainme shows,Music,M
3	US	PewDiePie	Gaming & Apps	Gaming	PewDiePie	111000000	Lifestyle	Gaming,A game,Lifestyle,Action-advent
4	US	MrBeast	Gaming & Apps	Entertainment	MrBeast	98100000	Lifestyle	Entertainment,Lifestyle,Techn
...	...	...	...	...	...	...	...	
852	NaN	1theK (원더케이)	Music	Music	1theK (원더케이)	24100000	Pop music	Music of Asia,Music,Pop
853	US	Post Malone	Gaming & Apps	Music	Post Malone	24000000	Music	Music,Hip hop music,Pop
854	IN	Amit Bhadana	NaN	Entertainment	Amit Bhadana	24000000	Entertainment	Music,Movies,Entertain
855	US	James Charles	Gaming & Apps	Entertainment	James Charles	24000000	Lifestyle	Lif
856	US	Netflix	None	Entertainment	Netflix	24000000	TV shows	Entertainment,TV shows,M

857 rows × 22 columns

```
data=df['followers'].astype('int64')
followers=pd.DataFrame(data)
followers.plot(kind='density')
```

```
data=df['followers'].astype('int64')
followers=pd.DataFrame(data)
followers.plot(kind='density')
plt.savefig('xyz.jpg',dpi=300)
plt.show()
```

[11] ✓ 0.2s




```
followers_mean=np.mean(df['followers'])
followers_mean
```

[12] ✓ 0.3s

```
49529754.95915986
```

```
followers_median=np.median(df['followers'])
followers_median
```

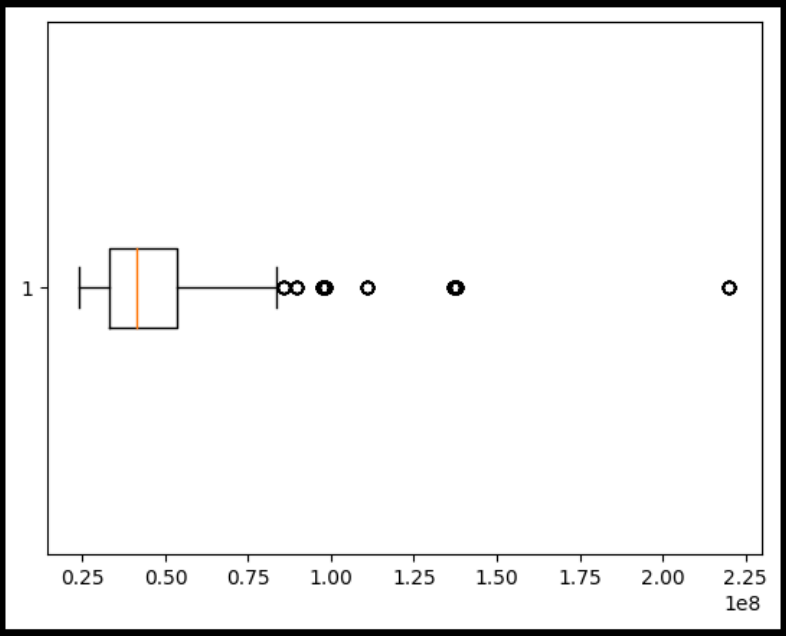
```
followers_median=np.median(df['followers'])
followers_median
```

[14] ✓ 0.2s

```
41300000.0
```

```
data=df['followers']
plt.boxplot(data,vert=0)
plt.show()
```

[17] ✓ 0.1s

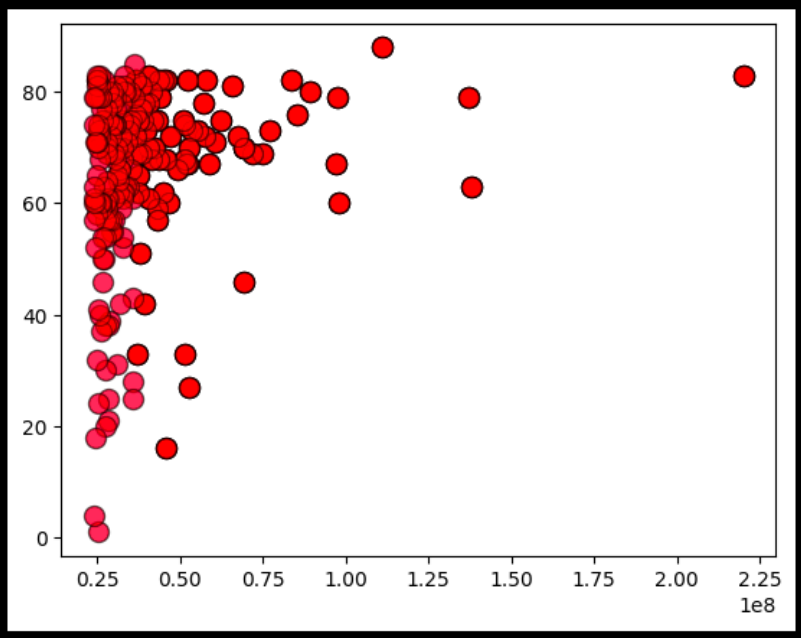



```
followers_1=df['followers']
boost_index=df['Boost Index']

plt.scatter(followers_1,boost_index,s=100,c='red',edgecolors='black',linewidths=1,alpha=0.4)
plt.show()
```

[18] ✓ 0.5s

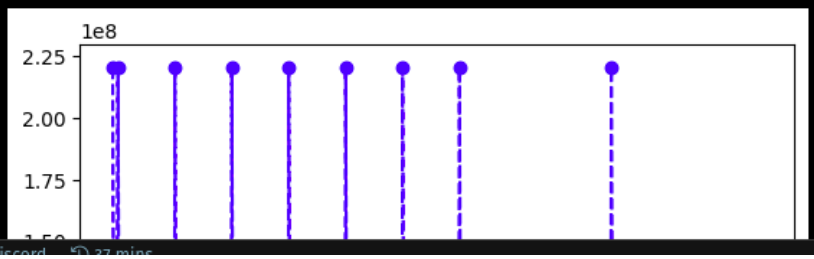
```
followers_1=df['followers']
boost_index=df['Boost Index']

plt.scatter(followers_1,boost_index,s=100,c='red',edgecolors='black',linewidths=1,alpha=0.4)
plt.show()
```

[18] ✓ 0.5s



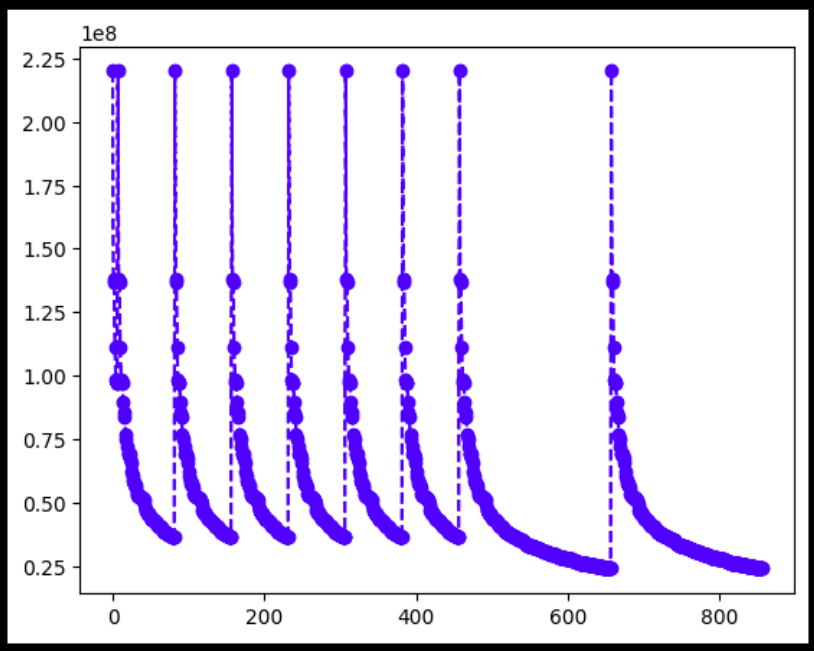

```
plt.plot(followers_1,'bo--')
plt.show()
```

[20] ✓ 0.1s



```
plt.plot(followers_1,'bo--')
plt.show()
```

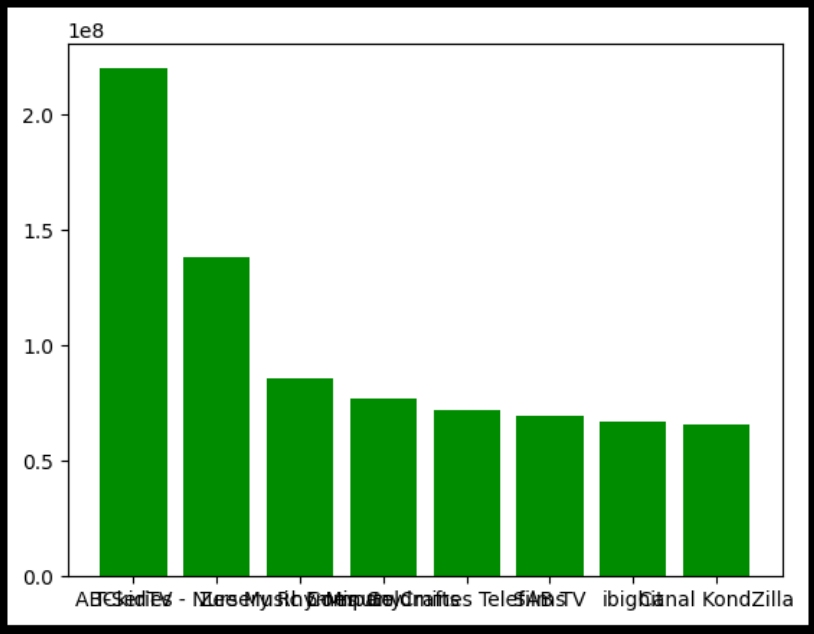
[20] ✓ 0.1s



```
channel_name=df['Channel Name'][:10]
plt.bar(channel_name,df['followers'][:10],color='green')
plt.show()
```

[49] ✓ 0.1s

```
/tmp/ipykernel_16204/1284212586.py:1: FutureWarning: The behavior of `series[i:j]` with an integer-
*label-based* indexing, consistent with e.g. `series[i]` lookups. To retain the old behavior, use `
  channel_name=df['Channel Name'][:10]
/tmp/ipykernel_16204/1284212586.py:2: FutureWarning: The behavior of `series[i:j]` with an integer-
*label-based* indexing, consistent with e.g. `series[i]` lookups. To retain the old behavior, use `
  plt.bar(channel_name,df['followers'][:10],color='green')
```



[ ]

Outcomes:

Inculcate the knowledge of python library like numpy, pandas, matplotlib for scientific-computing and data visualisation.

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

We used matplotlib to create

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