DATA VISUALIZATION ASSIGNMENT-1

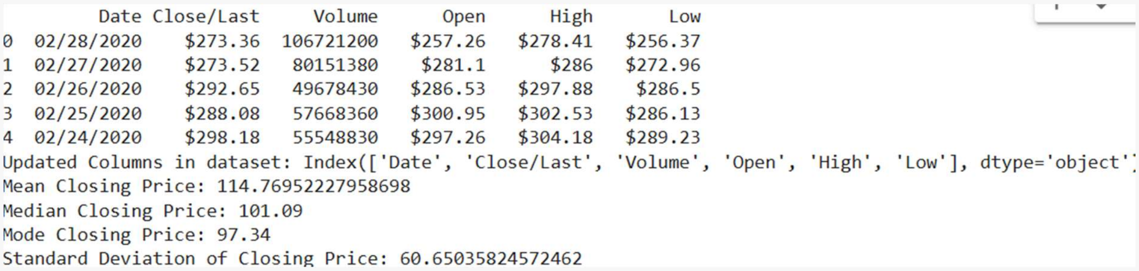
1. Develop a code to demonstrate mean, medium, mode, standard

deviation using Numpy and Pandas using a real time data set of

Apple stock data available on kaggle

Code:

```
import pandas as pd
import numpy as np
df = pd.read_csv('/content/HistoricalQuotes.csv')
df.columns = df.columns.str.strip()
print(df.head())
print("Updated Columns in dataset:", df.columns)
if 'Close/Last' in df.columns:
    df['Close/Last'] = df['Close/Last'].replace('[\$,]', '',
regex=True).astype(float)
    mean_price = np.mean(df['Close/Last'])
    median_price = np.median(df['Close/Last'])
    mode_price = df['Close/Last'].mode()[0]
    std_dev_price = np.std(df['Close/Last'])
    print("Mean Closing Price:", mean_price)
    print("Median Closing Price:", median_price)
    print("Mode Closing Price:", mode_price)
    print("Standard Deviation of Closing Price:", std_dev_price)
else:
    print("The 'Close/Last' column was not found. Please verify the
dataset's
column names.")
```



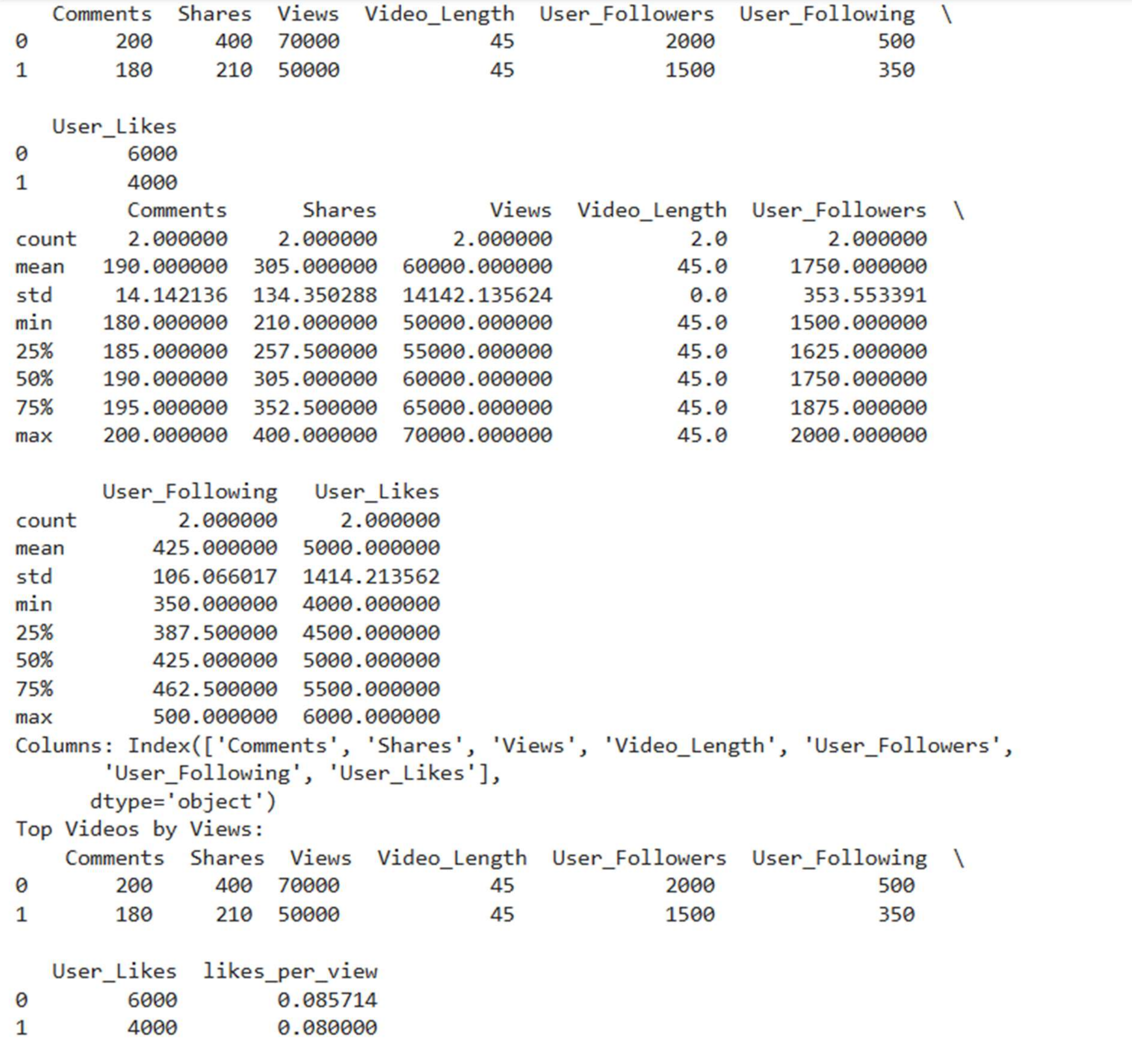
```
         Date Close/Last     Volume     Open     High      Low
0  02/28/2020    $273.36  106721200  $257.26  $278.41  $256.37
1  02/27/2020    $273.52   80151380   $281.1     $286  $272.96
2  02/26/2020    $292.65   49678430  $286.53  $297.88   $286.5
3  02/25/2020    $288.08   57668360  $300.95  $302.53  $286.13
4  02/24/2020    $298.18   55548830  $297.26  $304.18  $289.23
Updated Columns in dataset: Index(['Date', 'Close/Last', 'Volume', 'Open', 'High', 'Low'], dtype='object'
Mean Closing Price: 114.76952227958698
Median Closing Price: 101.09
Mode Closing Price: 97.34
Standard Deviation of Closing Price: 60.65035824572462
```

2. Develop a code to perform basic to advanced operation using both

Numpy and Pandas using TikTok video performance dataset

Code:

```
import numpy as np
import pandas as pd
df = pd.read_csv('test_features.csv')
print(df.head())
print(df.describe())
print("Columns:", df.columns)
if 'User_Likes' in df.columns and 'Views' in df.columns:
    df['likes_per_view'] = df['User_Likes'] / df['Views']
    top_videos = df.nlargest(5, 'Views')
    print("Top Videos by Views:\n", top_videos)
else:
    print("The columns 'User_Likes' and 'Views' are not found in the
dataset.")
```



```
   Comments  Shares  Views  Video_Length  User_Followers  User_Following  \
0       200     400  70000            45            2000             500
1       180     210  50000            45            1500             350

   User_Likes
0        6000
1        4000
         Comments      Shares         Views  Video_Length  User_Followers  \
count    2.000000    2.000000      2.000000           2.0        2.000000
mean   190.000000  305.000000  60000.000000          45.0     1750.000000
std     14.142136  134.350288  14142.135624           0.0      353.553391
min    180.000000  210.000000  50000.000000          45.0     1500.000000
25%    185.000000  257.500000  55000.000000          45.0     1625.000000
50%    190.000000  305.000000  60000.000000          45.0     1750.000000
75%    195.000000  352.500000  65000.000000          45.0     1875.000000
max    200.000000  400.000000  70000.000000          45.0     2000.000000

       User_Following   User_Likes
count        2.000000     2.000000
mean       425.000000  5000.000000
std        106.066017  1414.213562
min        350.000000  4000.000000
25%        387.500000  4500.000000
50%        425.000000  5000.000000
75%        462.500000  5500.000000
max        500.000000  6000.000000
Columns: Index(['Comments', 'Shares', 'Views', 'Video_Length', 'User_Followers',
       'User_Following', 'User_Likes'],
      dtype='object')
Top Videos by Views:
     Comments  Shares  Views  Video_Length  User_Followers  User_Following  \
0       200     400  70000            45            2000             500
1       180     210  50000            45            1500             350

   User_Likes  likes_per_view
0        6000        0.085714
1        4000        0.080000
```

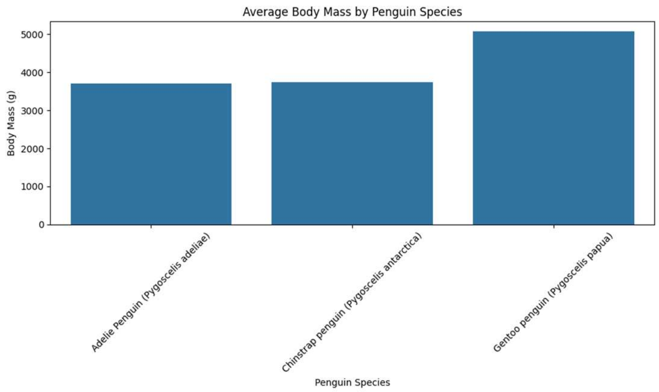
3. Develop a code to plot different comparison plots and composition

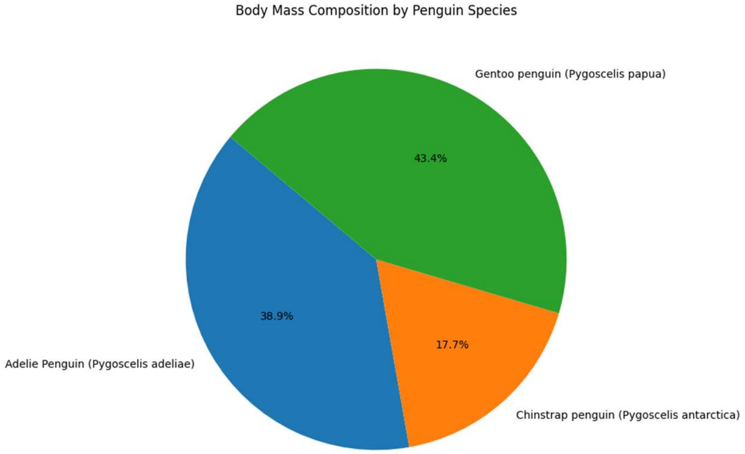
plots considering any suitable dataset.

Code:

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

```
df = pd.read_csv('/content/penguins_lter.csv')
plt.figure(figsize=(10, 6))
sns.barplot(x='Species', y='Body Mass (g)', data=df, errorbar=None)
plt.title('Average Body Mass by Penguin Species')
plt.xlabel('Penguin Species')
plt.ylabel('Body Mass (g)')
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
species_mass = df.groupby('Species')['Body Mass (g)'].sum()
plt.figure(figsize=(8, 8))
species_mass.plot.pie(autopct='%1.1f%%', startangle=140)
plt.title('Body Mass Composition by Penguin Species')
plt.ylabel('')
plt.show()
```

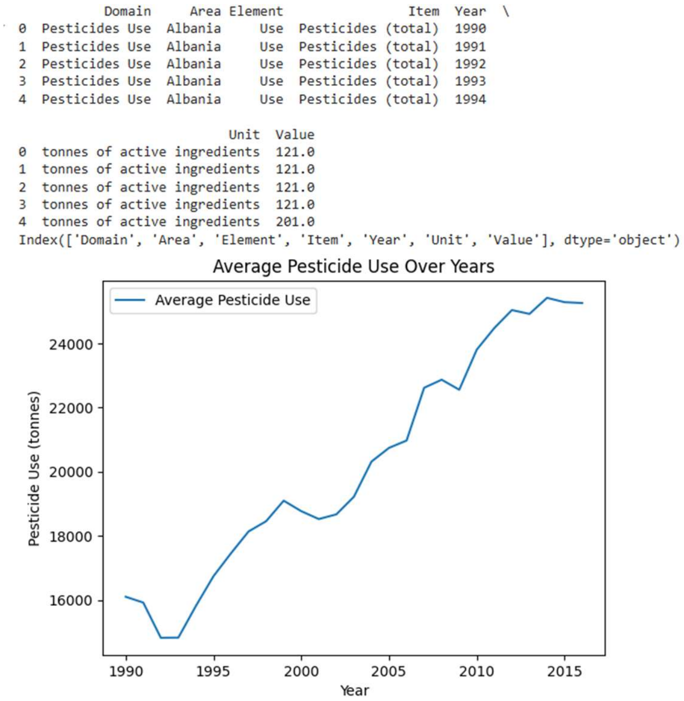




4. Develop a code using Matplotlib performing all Pyplot basics operation basic text and legend using Agriculture crop yield data set.

Code:

```
import pandas as pd
import matplotlib.pyplot as plt
df = pd.read_csv('/content/pesticides.csv')
print(df.head())
print(df.columns)
yearly_yield = df.groupby('Year')['Value'].mean().reset_index()
plt.plot(yearly_yield['Year'], yearly_yield['Value'], label='Average
Pesticide Use')
plt.xlabel('Year')
plt.ylabel('Pesticide Use (tonnes)')
plt.title('Average Pesticide Use Over Years')
plt.legend()
plt.show()
```

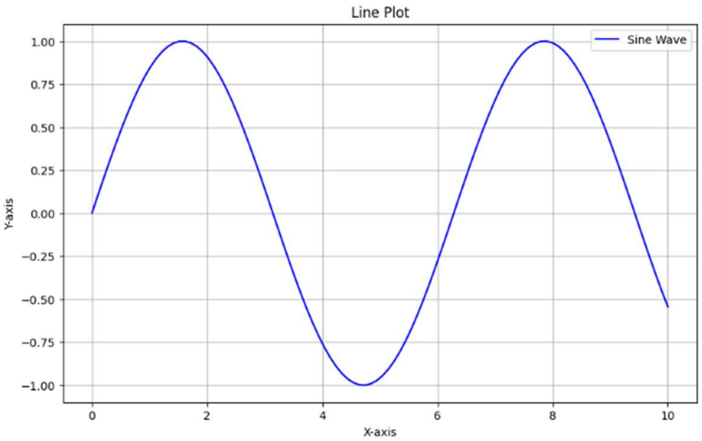


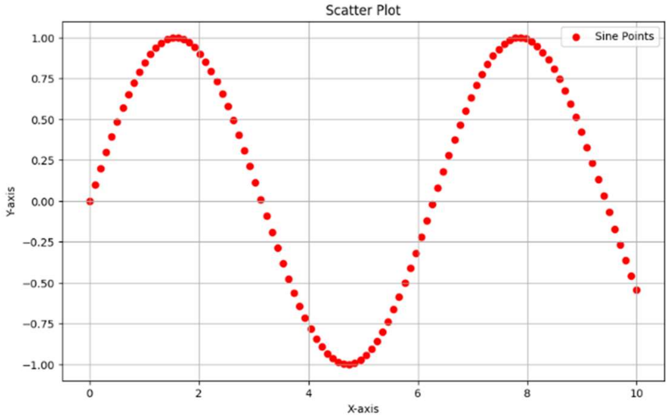
```
          Domain     Area Element                Item  Year  \
0  Pesticides Use  Albania     Use  Pesticides (total)  1990
1  Pesticides Use  Albania     Use  Pesticides (total)  1991
2  Pesticides Use  Albania     Use  Pesticides (total)  1992
3  Pesticides Use  Albania     Use  Pesticides (total)  1993
4  Pesticides Use  Albania     Use  Pesticides (total)  1994

                        Unit  Value
0  tonnes of active ingredients  121.0
1  tonnes of active ingredients  121.0
2  tonnes of active ingredients  121.0
3  tonnes of active ingredients  121.0
4  tonnes of active ingredients  201.0
Index(['Domain', 'Area', 'Element', 'Item', 'Year', 'Unit', 'Value'], dtype='object')
```

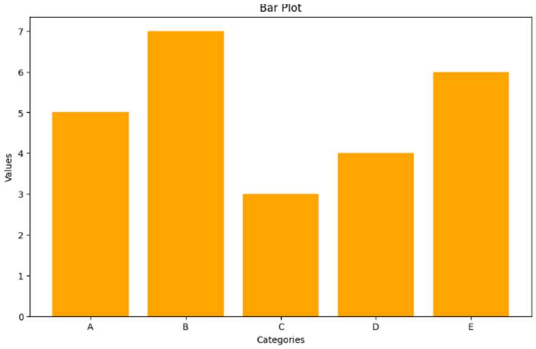
5. Develop a code to perform Matplotlib functions to display all the basic plots.

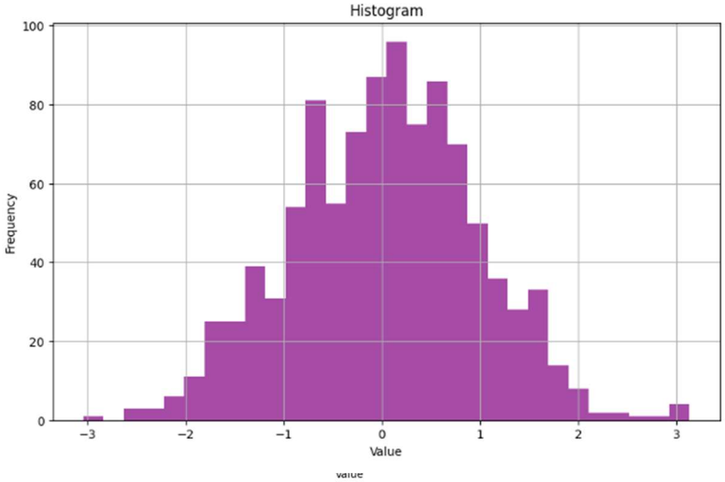
Code:

```
import matplotlib.pyplot as plt
import numpy as np
x = np.linspace(0, 10, 100)
y = np.sin(x)

# 1. Line Plot
plt.figure(figsize=(10, 6))
plt.plot(x, y, label='Sine Wave', color='b')
plt.title('Line Plot')
plt.xlabel('X-axis')
plt.ylabel('Y-axis')
plt.legend()
plt.grid()
plt.show()

# 2. Scatter Plot
plt.figure(figsize=(10, 6))
plt.scatter(x, y, color='r', label='Sine Points')
plt.title('Scatter Plot')
plt.xlabel('X-axis')
plt.ylabel('Y-axis')
plt.legend()
plt.grid()
plt.show()

# 3. Bar Plot
categories = ['A', 'B', 'C', 'D', 'E']
values = [5, 7, 3, 4, 6]
plt.figure(figsize=(10, 6))
plt.bar(categories, values, color='orange')
plt.title('Bar Plot')
plt.xlabel('Categories')
plt.ylabel('Values')
plt.show()

# 4. Histogram
data = np.random.randn(1000)
plt.figure(figsize=(10, 6))
plt.hist(data, bins=30, color='purple', alpha=0.7)
plt.title('Histogram')
plt.xlabel('Value')
plt.ylabel('Frequency')
plt.grid()
plt.show()

# 5. Pie Chart
sizes = [15, 30, 45, 10]
labels = ['A', 'B', 'C', 'D']
plt.figure(figsize=(8, 8))
plt.pie(sizes, labels=labels, autopct='%1.1f%%', startangle=140)
plt.title('Pie Chart')
plt.axis('equal')
plt.show()
```

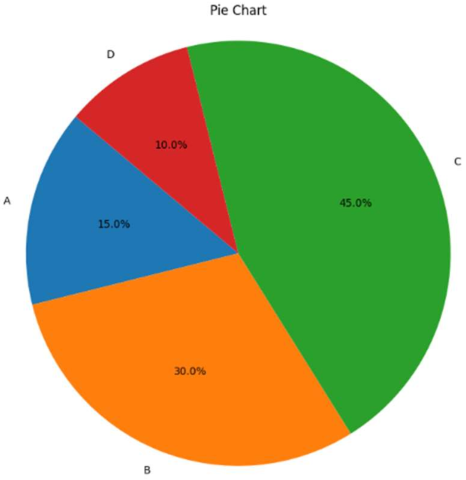

Line Plot
Sine Wave
Y-axis
X-axis
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
0
2
4
6
8
10


Scatter Plot
Sine Points
Y-axis
X-axis
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
0
2
4
6
8
10

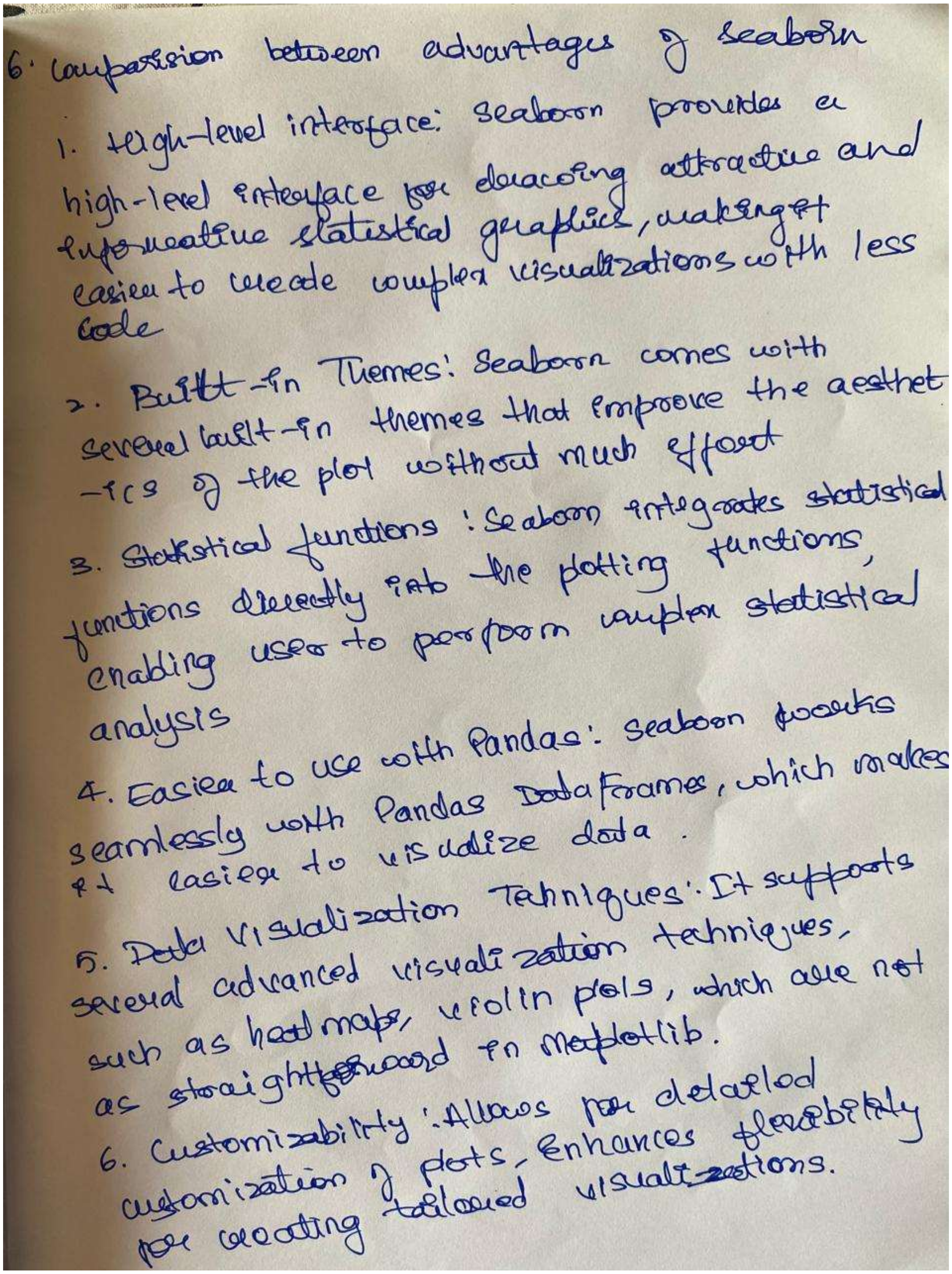

Bar Plot
Values
Categories
7
6
5
4
3
2
1
0
A
B
C
D
E


Histogram
100
80
60
40
20
0
Frequency
−3
−2
−1
0
1
2
3
Value
Value

6,


Pie Chart
D
10.0%
C
45.0%
A
15.0%
30.0%
B

6. Compare between advantages of seaborn and illustrate the role of controlling figure aesthetics using seaborn with a code snippet



6. comparision between advantages of seaborn

1. High-level interface: seaborn provides a high-level interface for drawing attractive and informative statistical graphics, making it easier to create complex visualizations with less code

2. Built-in Themes: Seaborn comes with several built-in themes that improve the aesthet-ics of the plot without much effort

3. Statistical functions: Seaborn integrates statistical functions directly into the plotting functions, enabling users to perform complex statistical analysis

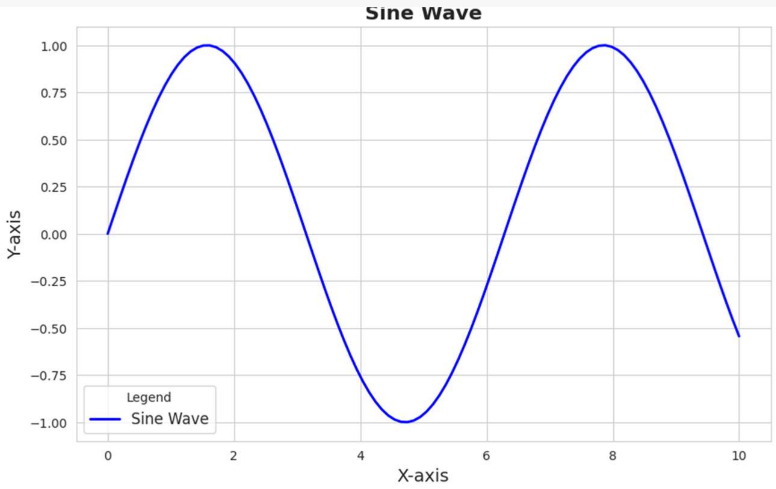
4. Easier to use with Pandas: seaborn works seamlessly with Pandas DataFrames, which makes it easier to visualize data.

5. Data Visualization Techniques: It supports several advanced visualization techniques, such as heatmaps, violin plots, which are not as straightforward in Matplotlib.

6. Customizability: Allows for detailed customization of plots, Enhances flexibility for creating tailored visualizations.

Code:

```
import seaborn as sns
import matplotlib.pyplot as plt
import numpy as np

sns.set_style("whitegrid")
x = np.linspace(0, 10, 100)
y = np.sin(x)
plt.figure(figsize=(10, 6))
sns.lineplot(x=x, y=y, label='Sine Wave', color='blue', linewidth=2)
plt.title('Sine Wave', fontsize=16, fontweight='bold')
plt.xlabel('X-axis', fontsize=14)
plt.ylabel('Y-axis', fontsize=14)
plt.legend(title='Legend', fontsize=12)
plt.grid(True)
plt.show()
```



Link of google colab notebook:

https://colab.research.google.com/drive/1kbPz0LwPbyYAd-rghxLXT-GGWqE1ySbI?usp=sharing