Experiment No. 4

Title: Matplotlib for Data Visualization

Batch: B-1 **Roll No: 16010422234** **Name:Chandana Ramesh Galgali**

Experiment No.: 4

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()
```

Matplotlib 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)

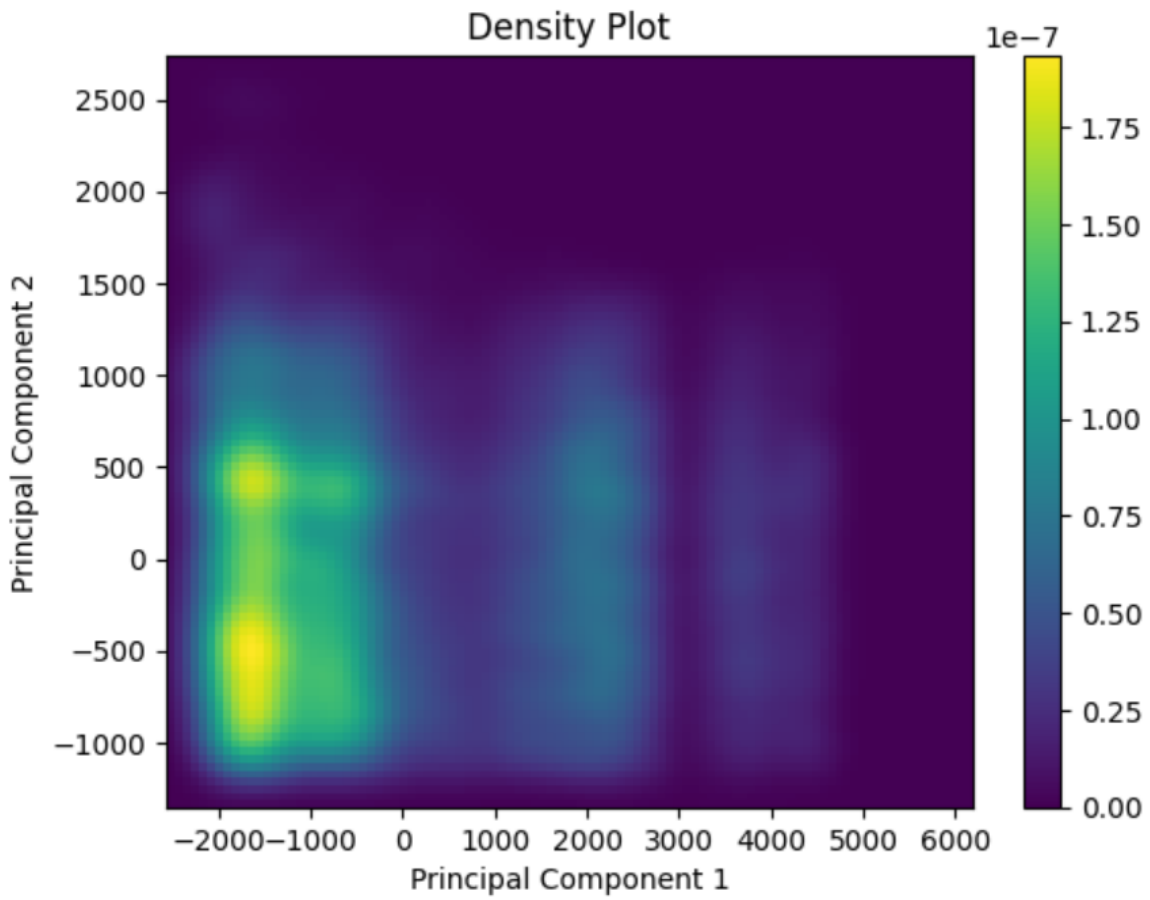
1. Download data set with at least 1500 rows and 10-20 columns(numeric and non numeric) from valid data sources
2. Visualization to summarize your data set(density, frequency plot)
3. Measures of central tendency of data set (mean, median etc)
4. Determining presence of outliers in your dataset(boxplot)
5. Correlation of attributes in your dataset(scatter plot and line plot on 2-3 pairs which are correlated)
6. Comparison of data plotted 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

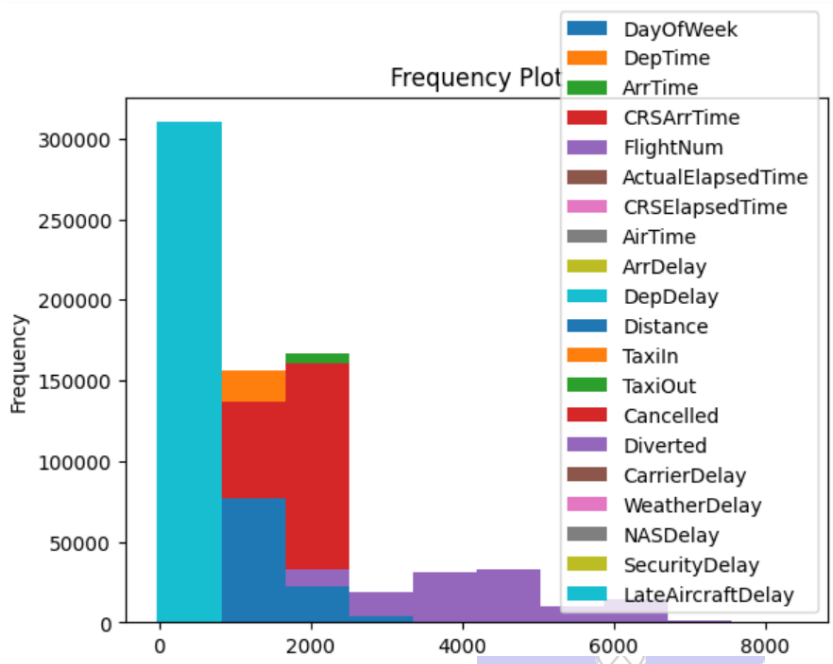
11. Write down observations for your dataset for each of the above listed tasks of analysis.

Result: (script and output)

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.decomposition import PCA
from sklearn.impute import SimpleImputer
from scipy.stats import gaussian_kde

data = pd.read_csv('/content/Flight_delay (1).csv')
numeric_data = data.select_dtypes(include=[np.number])

# Handle missing values using SimpleImputer
imputer = SimpleImputer(strategy='mean')
numeric_data_imputed = imputer.fit_transform(numeric_data)

pca = PCA(n_components=2)
reduced_data = pca.fit_transform(numeric_data_imputed)
kde = gaussian_kde(reduced_data.T)

x, y = reduced_data[:, 0], reduced_data[:, 1]
x_grid, y_grid = np.mgrid[x.min():x.max():100j, y.min():y.max():100j]
positions = np.vstack([x_grid.ravel(), y_grid.ravel()])
density = kde(positions).reshape(x_grid.shape)

plt.imshow(density.T, origin='lower', aspect='auto', extent=[x.min(), x.max(), y.min(), y.max()])
plt.colorbar()
plt.title('Density Plot')
plt.xlabel('Principal Component 1')
plt.ylabel('Principal Component 2')
plt.show()

data.plot(kind='hist')
plt.title('Frequency Plot')
```

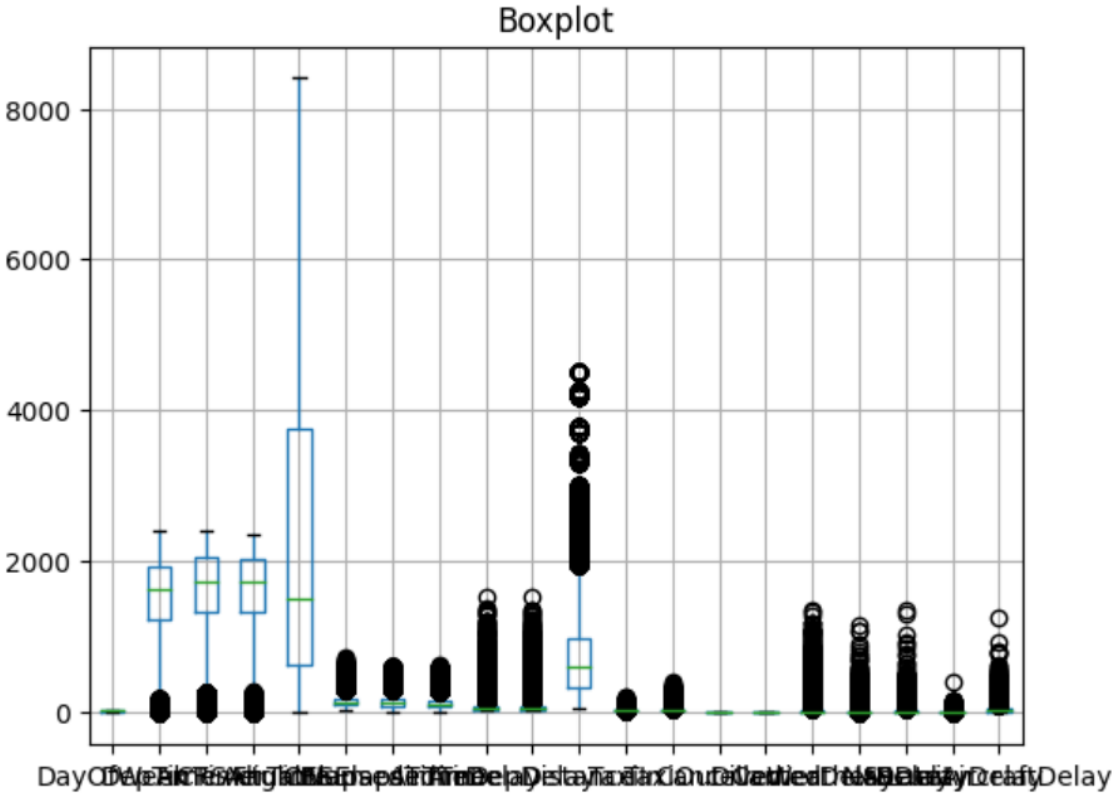
```
plt.show()

mean = data.mean()
median = data.median()
print('Mean:', mean)
print('Median:', median)

data.boxplot()
plt.title('Boxplot')
plt.show()

correlation_data = data[['AirTime', 'ActualElapsedTime', 'LateAircraftDelay']]
scatter_matrix = pd.plotting.scatter_matrix(correlation_data)
plt.title('Scatter Plot Matrix')
plt.show()
```

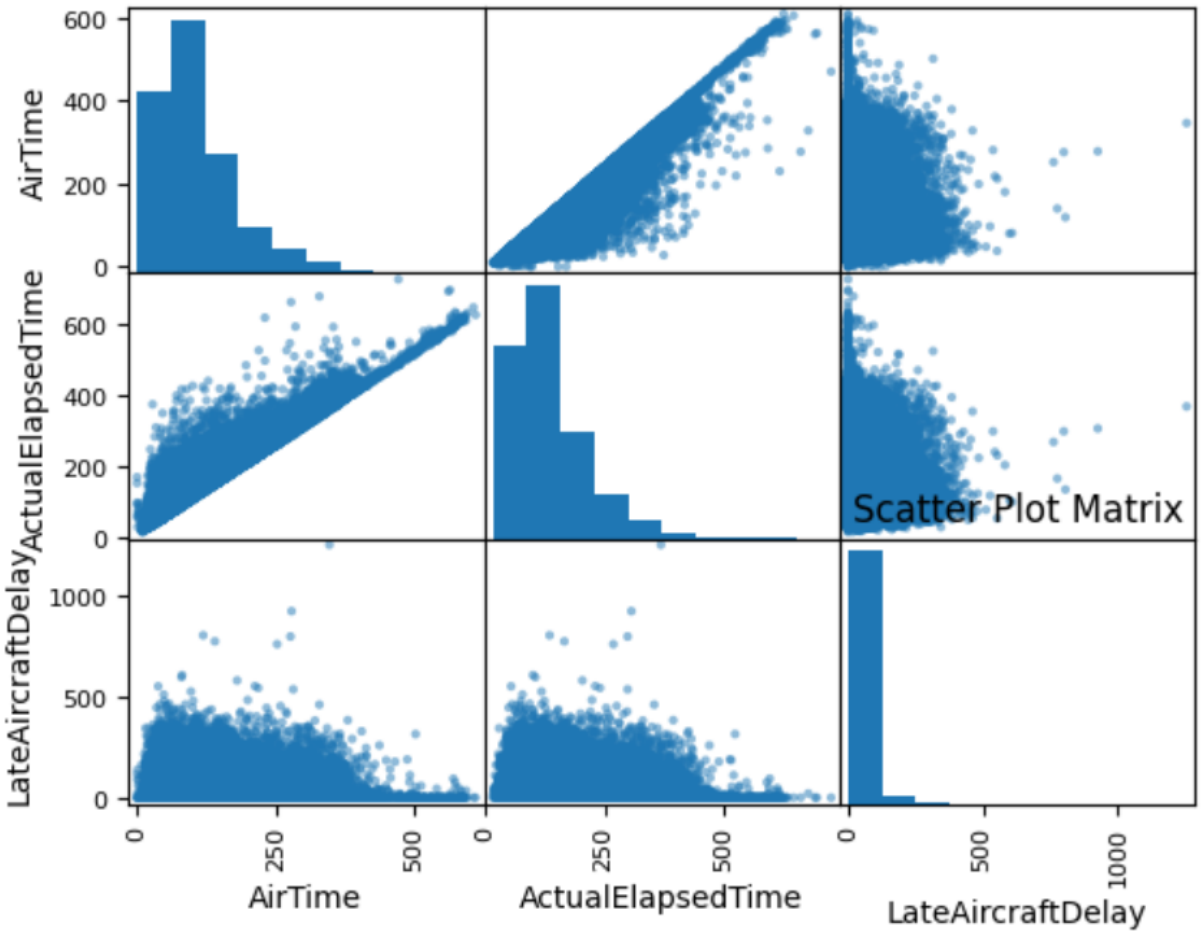





```
Mean: DayOfWeek                3.966447
DepTime                     1556.399912
ArrTime                     1615.361277
CRSArrTime                  1642.839211
FlightNum                   2154.994276
ActualElapsedTime            133.288993
CRSElapsedTime               130.110936
AirTime                      107.547177
ArrDelay                      61.000210
DepDelay                      57.822126
Distance                     738.470976
TaxiIn                         6.796386
TaxiOut                       18.945448
Cancelled                      0.000000
Diverted                       0.000000
CarrierDelay                  17.645500
WeatherDelay                   3.485797
NASDelay                      13.126560
SecurityDelay                  0.092861
LateAircraftDelay             26.649118
dtype: float64
```

```
Median: DayOfWeek                4.0
DepTime                       1614.0
ArrTime                       1731.0
CRSArrTime                    1715.0
FlightNum                     1506.0
ActualElapsedTime              115.0
CRSElapsedTime                 111.0
AirTime                         88.0
ArrDelay                        42.0
DepDelay                        40.0
Distance                       590.0
TaxiIn                           5.0
TaxiOut                         15.0
Cancelled                        0.0
Diverted                         0.0
CarrierDelay                     3.0
WeatherDelay                     0.0
NASDelay                         0.0
SecurityDelay                    0.0
LateAircraftDelay               13.0
dtype: float64
```





Outcomes: 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)
The utilization of Pandas built-in visualization and Matplotlib visualization for exploratory data analysis proved to be highly effective in achieving the aim of analyzing and understanding the dataset. These tools provided us with the necessary means to visualize and interpret the data, enabling us to make informed decisions and derive valuable insights.

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