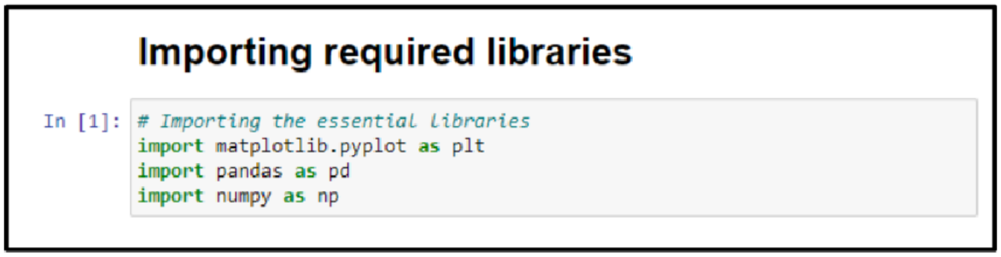
Applied Data Science Assignment-1

Name: Vasanth Kumar Duggimpudi
Student No: 22018255
Course: MSc Data Science

Repository: https://github.com/Vasanthkumar09/visualisation

Task 1:

Displaying line plot with multiple lines(Specific labels, as well as legend, are also included)



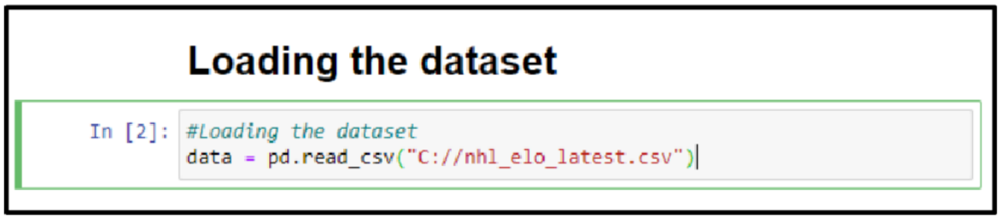
Importing required libraries

```
In [1]: # Importing the essential libraries
        import matplotlib.pyplot as plt
        import pandas as pd
        import numpy as np
```

Figure 1: Importing required libraries

(Source: Obtained from the Jupyter notebook environment)

The figure that is shown above is displaying the entire process of importing the necessary libraries in the jupyter notebook platform. In this specific stage, three essential libraries such as pyplot, pandas, and NumPy have been imported in order to perform the data visualization. Hence, these three specific libraries can be able in order to provide a very powerful toolkit for data visualization and various sorts of scientific computing in Python programming language. Hence, this particular code snippet visualises the overall process of setting up the entire environment in order to utilise the libraries in the concerned code.



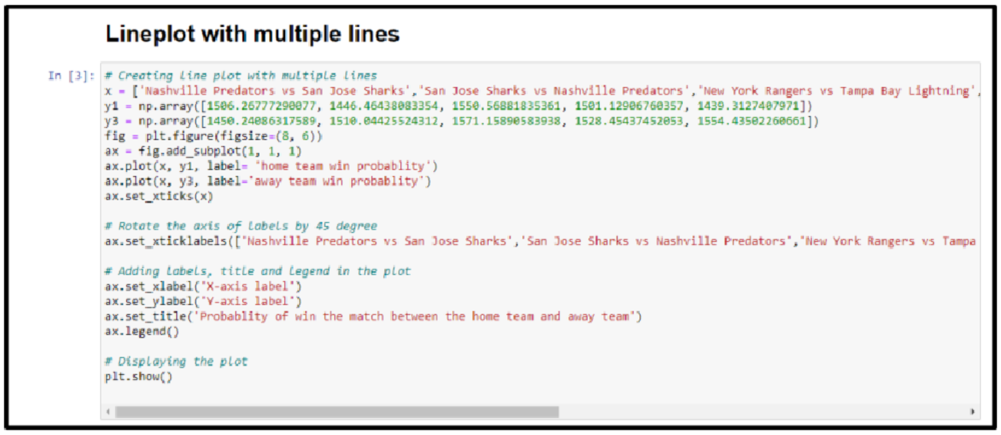
Loading the dataset

```
In [2]: #Loading the dataset
        data = pd.read_csv("C://nhl_elo_latest.csv")
```

Figure 2: Loading the entire set of input data

(Source: Obtained from the Jupyter notebook environment)

After importing the necessary libraries now the researcher will have to load the entire dataset in the jupyter notebook platform by using the function “pd. read_csv”.


Lineplot with multiple lines

```
In [3]: # Creating line plot with multiple lines
        x = ['Nashville Predators vs San Jose Sharks','San Jose Sharks vs Nashville Predators','New York Rangers vs Tampa Bay Lightning',
        y1 = np.array([1506.26777290077, 1446.46438083354, 1550.56881835361, 1501.12906760357, 1439.3127407971])
        y3 = np.array([1450.24086317589, 1510.04425524312, 1571.15890583938, 1528.45437452053, 1554.43502260661])
        fig = plt.figure(figsize=(8, 6))
        ax = fig.add_subplot(1, 1, 1)
        ax.plot(x, y1, label= 'home team win probablity')
        ax.plot(x, y3, label='away team win probablity')
        ax.set_xticks(x)

        # Rotate the axis of labels by 45 degree
        ax.set_xticklabels(["Nashville Predators vs San Jose Sharks','San Jose Sharks vs Nashville Predators","New York Rangers vs Tampa

        # Adding labels, title and legend in the plot
        ax.set_xlabel("X-axis label")
        ax.set_ylabel("Y-axis label")
        ax.set_title('Probablity of win the match between the home team and away team")
        ax.legend()

        # Displaying the plot
        plt.show()
```

Figure 3: Code for creating a line plot with multiple lines

(Source: Obtained from the Jupyter notebook environment)

The code snippet that is shown above is displaying the entire code for developing a line plot with multiple lines. In order to create this particular plot, various values regarding the chance of winning the home team and the chance of winning the away team have been chosen. In addition to that, in order to create the labels of the multiple lines plot, the researcher has chosen the name of the players the between two teams.

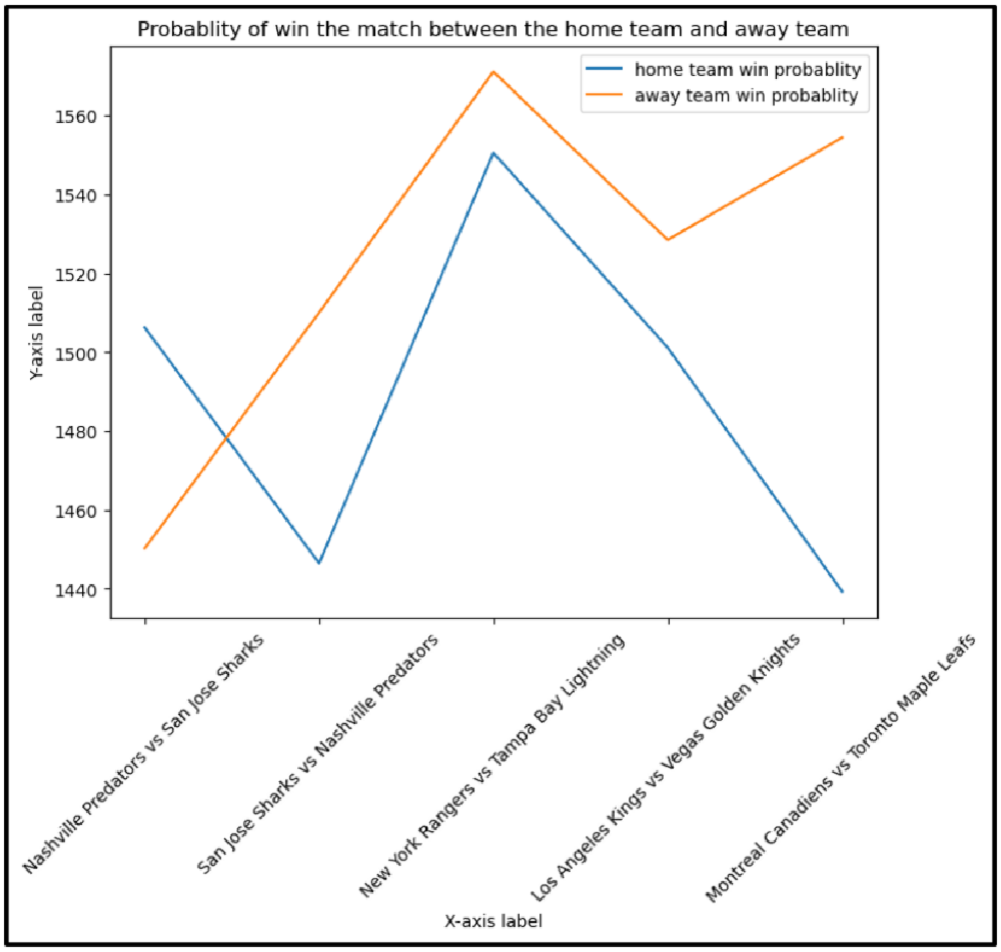


Figure 4: Probability of winning the match between the two teams

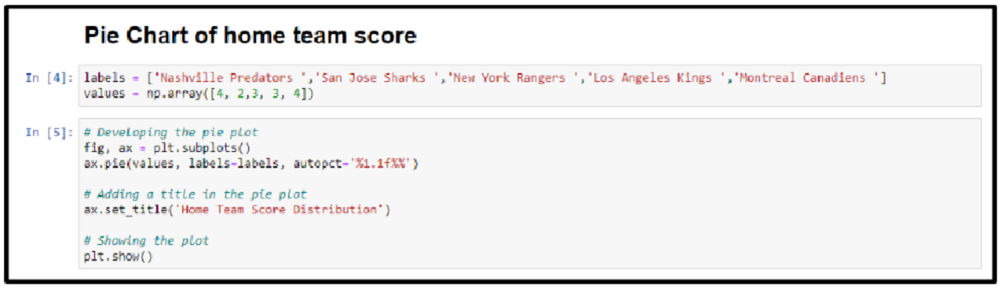
(Source: Obtained from the Jupyter notebook environment)

The plot that is shown above is displaying the chance of winning the match between the following team and away team in the form of a line plot. Hence from the above plot, it can easily be inferred that the probability of winning is greater for the Away team as compared to the home team.

Task 2:

Displaying graphs with another two data visualisation methods(Pie chart and scatter plot)

In this specific stage, the other two data visualisation such as pie chart as well as scatter plot has been displayed.



Pie Chart of home team score

```
In [4]: labels = ['Nashville Predators ','San Jose Sharks ','New York Rangers ','Los Angeles Kings ','Montreal Canadiens ']
        values = np.array([4, 2,3, 3, 4])

In [5]: # Developing the pie plot
        fig, ax = plt.subplots()
        ax.pie(values, labels=labels, autopct='%1.1f%%')

        # Adding a title in the pie plot
        ax.set_title('Home Team Score Distribution')

        # Showing the plot
        plt.show()
```

Figure 5: Pie chart based on the score of the home team

(Source: Obtained from the Jupyter notebook environment)

The code that is mentioned above is required in order to generate a pie chart by utilising the "matplotlib" library in the Python programming language in order to visualize the overall distribution of the actual scores for five different players of the home team. Hence a list named "labels" has been created that eventually contains various sorts of names of the team members. In addition to that, an array named 'values' is also created by utilising "numpy" function which eventually contains the respected scores for every team.

Along with that, a pie chart is also developed by using the "ax.pie()" technique. On the other hand, this particular method basically considers in the "values" as well as "labels" as input variables, and the function named "autopct='%1.1f%%'" eventually formats the mentioned values to one decimal point along with a percentage mark. After that, A specific title has been added to this particular pie chart by utilising the ax.set_title() technique. Finally, the "plt.show() method" is utilised in order to visualize the pie chart.

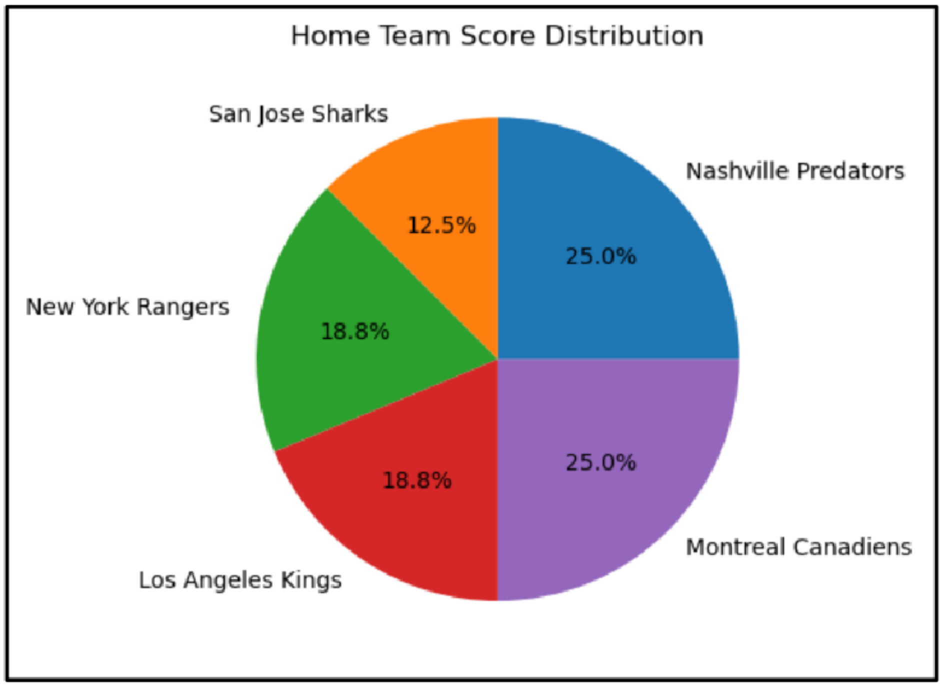
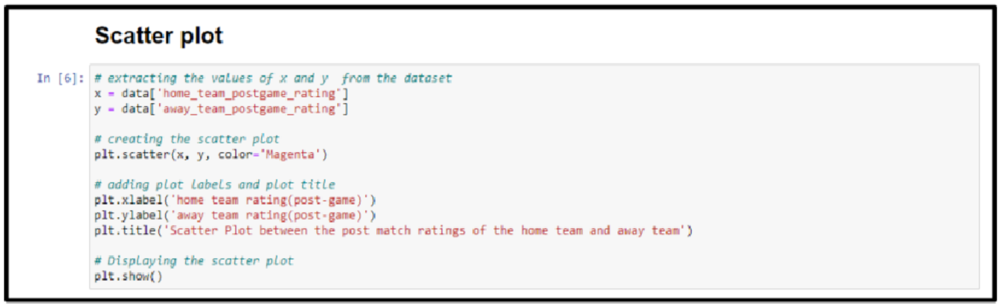


Figure 6: Home Team Score Distribution in a form of a pie chart

(Source: Obtained from the Jupyter notebook environment)

The figure that is displayed above is showing the Score Distribution of the Home Team in a form of a pie chart. In addition to that, from the above plot, it can easily be inferred that the maximum percentage of scores is achieved by two players and the highest percentage is 25 %. On the other hand, the lowest percentage is 12.5 % and it is denoted by the orange colour.



Scatter plot

```
In [6]: # extracting the values of x and y  from the dataset
        x = data['home_team_postgame_rating']
        y = data['away_team_postgame_rating']

        # creating the scatter plot
        plt.scatter(x, y, color='Magenta')

        # adding plot labels and plot title
        plt.xlabel('home team rating(post-game)')
        plt.ylabel('away team rating(post-game)')
        plt.title('Scatter Plot between the post match ratings of the home team and away team')

        # Displaying the scatter plot
        plt.show()
```

Figure 7: Scatter plot based on the post-match ratings of the following two teams

(Source: Obtained from the Jupyter notebook environment)

The code that is mentioned above is required in order to generate a scatter chart based on the post-match ratings of the home team as well as away team. First of all, the values of the variables named x and y are extracted from the entire set of input data by using the data object.

In this case, the “x” variable eventually represents the post-match rating of the home team, and the “y” variable denotes the post-match rating of the away team. After that, a scatter plot has been developed by using the “plt. scatter()” technique. In addition to that, this method eventually takes in the values of x and y as input, as well as the colour parameter, which is eventually set to 'Magenta' in order to change the colour of the scatter plot. Along with that, various sorts of functions such as “plt. xlabel ()”, “plt. ylabel ()”, and “plt. title()” has been utilised in order to include labels to the x-axis, and y-axis, as well as the title of the scatter plot, respectively. Ultimately, the plt. show() function has been utilised in order to visualize the scatter plot.

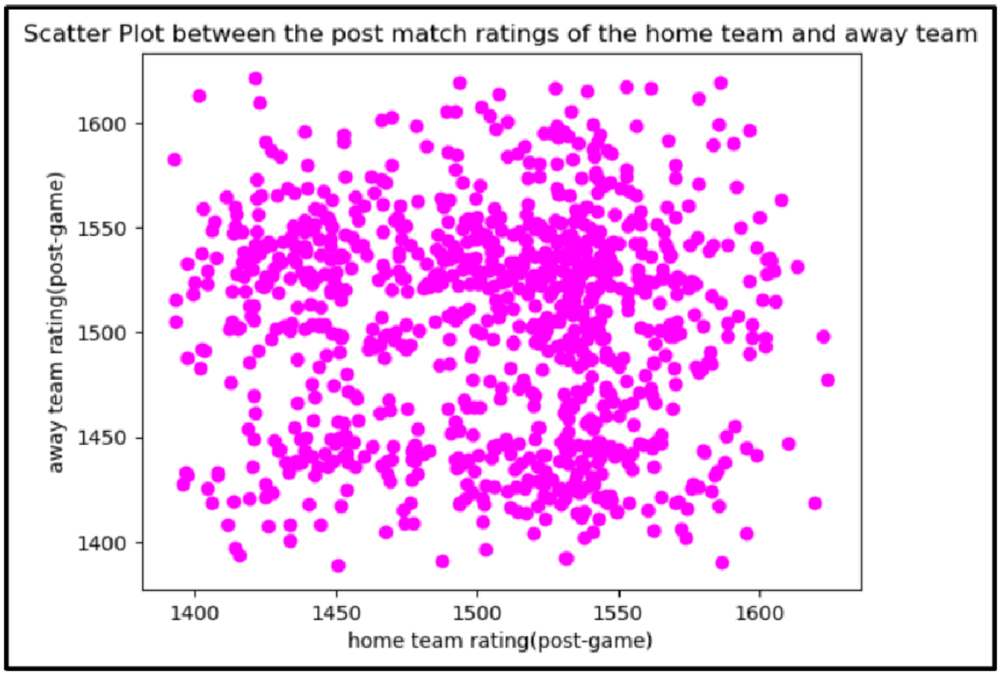


Figure: home team rating Vs away team rating in a form of a Scatter plot

(Source: Obtained from the Jupyter notebook environment)

The figure that is displayed above is showing the rating of both the following teams in a form of a Scatter plot. The final scatter plot displays the actual relationship between the post-match ratings of the home team as well as the away team. In addition to that, the x-axis denotes the post-match rating of the home team, as well as the y-axis, denotes the post-match rating of the away team. Along with that, each of the data points denotes a single match.