DATA NARRATIVE

#1 Vipul Sunil Patil (22110189)
Btech First Year Student (CSE),
IIT Gandhinagar,
Gandhinagar, Gujrat, India.
vipul.patil@iitgn.ac.in

I. Overview of the Dataset

The dataset consists of 2 files.

Strings, integers, and floats are all present in this data. However, although being large, this data is insufficient in certain ways. Certain things are not adequately described. Several missing information had needed to be filled in but, Data was usable after all, being easily transformed, calculated according to our need; this data was ready to create questions.

In this dataset we have 8 file which contains common elements/ columns which are Player1: The name of the first player, Player2: The name of the second player, Round: The round of the tournament in which the match was played, Result: The final result of the match, indicating which player won (usually denoted by "W") or if it was a draw (denoted by "D"), FNL1: The number of games won by Player1 in the match, FNL2: The number of games won by Player2 in the match, FSP.1: The percentage of first serves made by Player1, FSW.1: The percentage of first serve points won by Player1, SSP 0.10: The percentage of second serves made by Player1, SSW.1: The percentage of second serve points won by Player1, ACE.1: The number of aces served by Player1, DBF.1: The number of double faults committed by Player1, WNR.1: The percentage of return games won by Player1, UFE.1: The number of unforced errors committed by Player1, BPC.1: The number of break points created by Player1, BPW.1: The number of break points won by Player1, NPA.1: The number of net points attempted by Player1, NPW.1: The number of net points won by Player1, TPW.1: The total number of points won by Player1 in the match, ST1.1, ST2.1, ST3.1, ST4.1, ST5.1: The score in each set won by Player1 (if the match was played in best-of-five format), FSP.2: The percentage of first serves made by Player2, FSW.2: The percentage of first serve points won by Player2, SSP 0.20: The percentage of second serves made by Player2, SSW.2: The percentage of second serve points won by Player2, ACE.2: The number of aces served by Player2, DBF.2: The number of double faults committed by Player2, WNR.2: The percentage of return games won by Player2, UFE.2: The number of unforced errors committed by Player2 ,BPC.2: The number of breakpoints created by Player2, BPW.2: The number of break points won by Player2, NPA.2: The number of net points attempted by Player2, NPW.2: The number of net points won by Player2, TPW.2: The total number of points won by Player2 in the match, ST1.2, ST2.2, ST3.2, ST4.2, ST5.2: The score in each set won by Player2 (if the match was played in best-of-five format)

In my situation, I make use of data visualization and statistics to make sense of the data. It also helps us understand the plot and what data tried to convey us. On the other hand, exploring this data and seeing patterns and trends explains to me how the games like tennis are carried out.

Statistical analysis and charts came to provide numerous conclusions in front of me, but this data did not support some of them. With this information, we were able to provide several answers. Below questions and their answers will explain the data much closer and will make it easy to understand the data.

II. Scientific Questions/Hypotheses

1. Suppose there is a match between 2 players, can we predict who is going to win.

2. Form a strategy for a player to play against a specific player.

3. How can we make recommendations to a viewer based on preferable matches of that specific viewer.

4. How can we choose players for a match so that the match becomes interesting and competitive.

5. Can we group tennis players based on their playing style, and then use this information to predict which group is most likely to succeed against another group in a match?

6. Plot an 6d plot for common entries?

7. Can we calculate the number of tough matches and easiest matches?

8.Is there a significant difference in the number of games won by the first player versus second player in a match?

III. Details of Libraries and Functions

Libraries Used in the Code are:

1. Pandas can be defined as a Python package that offers user-friendly data structures and tools for data analysis. It is constructed on top of NumPy, a scientific computing library for Python, and presents an effective and simple environment for data manipulation and analysis to Python developers. Pandas includes two primary data structures, namely the Series and DataFrame objects. The Series is a one-dimensional array-like object that can contain any data type, whereas the DataFrame is a two-dimensional table-like data structure with rows and columns capable of holding various data types.[1]

2. Matplotlib is a versatile and easy-to-use visualization library in Python that offers powerful tools for creating different types of plots such as line, bar, scatter, histogram, and many more. It is based on NumPy arrays and has seamless integration with the broader SciPy ecosystem.[2]

3. plotly.graph objs is a Python module that allows you to create interactive and dynamic data visualizations. It's part of the Plotly ecosystem, which also contains libraries like Plotly Express and Dash.

4. Numpy is a key Python module for scientific computing. It supports massive, multi-dimensional arrays and matrices, as well as a wide library of mathematical functions for working with these arrays.

5. Sklearn.decomposition

6. Sklearn.preprocessing

7. Sklearn.cluster

Functions used in code.

1. For making dataframe we use Dataframe().
2. To sum the values we use sum().
3. To group data as we need to access we use groupby().
4. reset_index() method to reset the index of the dataframe.
5. using the plot() we can create a plot of graphs and specify kind using kind.
6. We show the chart using the show().
7. sort_values() is used to sort values ascending or descending accordingly.
8. scatter() helps us in plotting scatter plots.
9. For the bar plot we use barh() where h means horizontal.
10. We set the title for the chart using the title() and x and y labels using xlabel() and ylabel() respectively.
11. go.scatter() helps us in plotting 3d scatter plots.

IV. Answers to the Questions (with appropriate illustrations)

1. For this question we need the gameplay and strategy of both the players. For this we need to take the average of points that a particular player received in previous matches. we had taken the best players in terms of maximum number of matches played so

```
player1_df = df.groupby('Player1').size().reset_index(name='Matches')
player2_df = df.groupby('Player2').size().reset_index(name='Matches')
```

	Player	Matches
0	Adrian Mannarino	2
1	Albert Montanes	1
2	Albert Ramos	1
3	Alejandro Falla	2
4	Alejandro Gonzalez	1

We had taken Rafael Nadal and Roger Federer as they had played maximum number of games

98	Rafael Nadal	7
104	Roger Federer	6

Now we will make new dataframe for both players and add matches

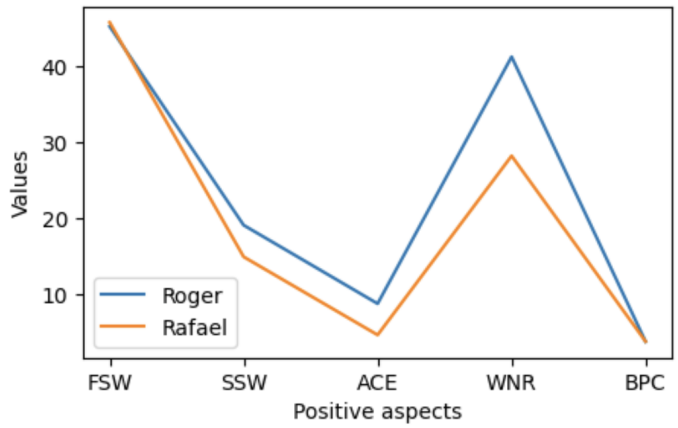
```
Rogerdf = pd.DataFrame()
Rafaeldf = pd.DataFrame()
```

```
if row['Player1'] == 'Roger Federer':
    Rogerdf = Rogerdf.append({'FSP': row['FS
```

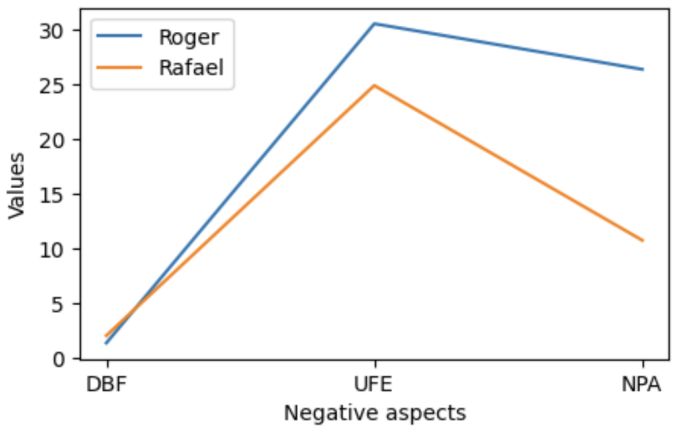
```
elif row['Player2'] == 'Roger Federer':
    Rogerdf = Rogerdf.append({'FSP': row['FS
```

Plotting line graph for positive points and negative points for both players will show that which player will win the game

```
fig, ax = plt.subplots(figsize=(5,3))
x1 = ['FSW', 'SSW', 'ACE', 'WNR', 'BPC']
y1_1 = Rogerdf.iloc[6, [1, 3, 4, 6, 8]]
y1_2 = Rafaeldf.iloc[7, [1, 3, 4, 6, 8]]
```




```
fig, ax = plt.subplots(figsize=(5,3))
x2 = ['DBF', 'UFE', 'NPA']
y2_1 = Rogerdf.iloc[6, [5, 7, 10]]
y2_2 = Rafaeldf.iloc[7, [5, 7, 10]]
```



through the plot it is not clear as Roger has made more positive points and more negative points therefore we need to make formula and one with more score will be the best one and will have more probability of winning (this is based on my idea)

```
def formula(FSW,SSW,ACE,BPC,TPW,WNR,DBF,UFE
  return (1*FSW)+(2*SSW)+(3*ACE)+(2*BPC)+(2
```

```
E,NPA):
2*TPW)+(1*WNR)-((3*DBF)+(3*UFE)+(2*NPA))
```

```
Score of Roger is 221.00000000000003
Score of Rafael is 223.1428571428571
```

```
Score of Roger is 2.45227033057059
Score of Rafael is 2.790741199759231
```

Here we can see that Rafael has scored maximum points in the formula therefore the probability of winning a match of Rafael is more.

2. For this question, we need to give the player information about his opponent's strengths and weaknesses and his own stats based on which he can see where he is lagging and where he needs to improve his game.

Therefore we need to take into account all the games played by the opponent player so we will make a new data frame where we will upload all matches and points the opponent player has scored in each match he played. Similarly for his own stats.

```
|  32 | Dominika Cibulkova          |          7 |
```

```
|  83 | Na Li                       |          7 |
```

Let Dominika Cibulkova be the player and Na Li as the opponent

```
Player = pd.DataFrame()
Opponent = pd.DataFrame()
```

```
for index, row in df2.iterrows():
    if row['Player1'] == 'Dominika Cibulkova'
        Player = Player.append({'FSP': row['FS
```

```
   elif row['Player2'] == 'Dominika Cibulkova
       Player = Player.append({'FSP': row['FS
```

```
for index, row in df2.iterrows():
    if row['Player1'] == 'Na Li':
        Opponent = Opponent.append({'FSP': ro
```

```
    elif row['Player2'] == 'Na Li':
        Opponent = Opponent.append({'FSP': ro
```

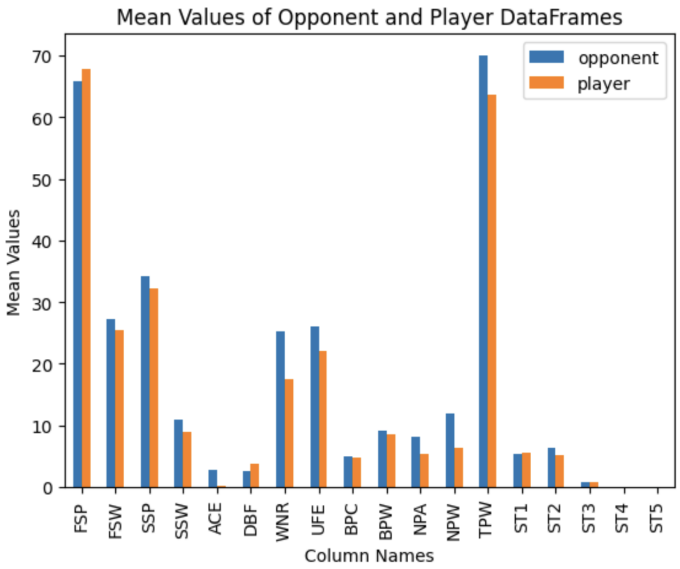
And take the mean of all points of both players.

```
mean_opponent = Opponent.mean()
mean_player = Player.mean()
```

```
df_mean = pd.DataFrame({'opponent': mean_opponent, 'player': mean_player})
```

Now we will plot bar to show out the player

```
df_mean.plot(kind='bar')
plt.xlabel('Column Names')
plt.ylabel('Mean Values')
```

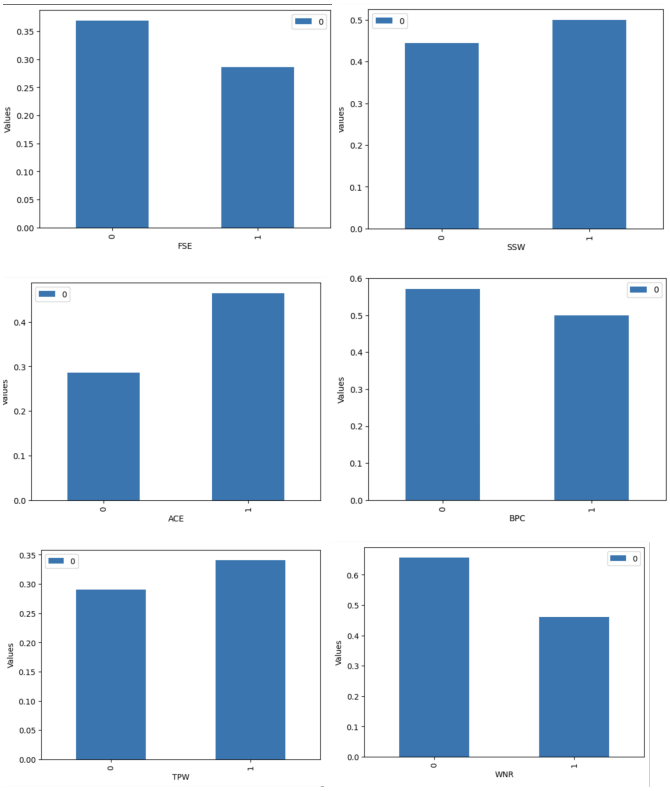


We will now check where the strong points of this player are like how many points he has scored in which directions. For example there are pulse points of FSW, SSW, ACE, BPC, BPW, TPW and negative points are DBF and UFE.that is, in terms of pulse points our player needs more than opponent and in terms of negative points our player needs less than opponent. Before that we need to first scale the data.

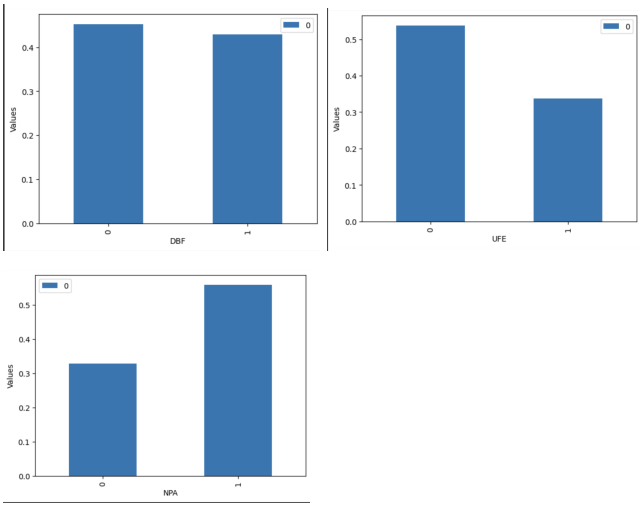
```
scale = MinMaxScaler()
scale.fit(Opponent)
Opponent = scale.transform(Opponent)

scale.fit(Player)
Player = scale.transform(Player)
```

```
indices = [1, 3, 4, 8, 12, 6]
names = ['FSE', 'SSW', 'ACE', 'BPC', 'TPW', 'WNR']
count = 0
for i in indices:
    a = np.mean(Player[:, i])
    b = np.mean(Opponent[:, i])
```




```
indices = [5, 7, 10]
names = ['DBF', 'UFE', 'NPA']
count = 0
for i in indices:
    a = np.mean(Player[:, i])
    b = np.mean(Opponent[:, i])
```



This will give a clear overview of the opponent to the player. This will show which are the fields at which which player is good.

So, the data shows in positive points SSW ACE and TPW are the points that the player should look at, and in negative points DBF and UFE are the points that the player should look at.

3. For this we first need to check the matches which the viewer had seen in the past. As there is no viewer, we will assume it.

```
'Rafael Nadal', 'Fabio Fognini',
'Gilles Simon', 'Roger Federer',
```

```
'Tobias Kamke', 'Paolo Lorenzi'
'Gilles Simon', 'Lleyton Hewitt
```

```
'Viktor Troicki', 'James Blake'
```

```
Benjamin Becker', 'Jeremy Chardy
```

assume these are the matches seen by our viewer. Now to give recommendations to viewers we need to check similar matches and to find this we can use k means that will provide the clusters. to apply k means we need to first eliminate the first 2 columns.

```
new_df = df3.iloc[:, 2:].copy()
```

Now k means, we are assuming 5 clusters.

```
num_clusters = 5
km = KMeans(num_clusters)
predicted = km.fit_predict(new_df)
```

here we need to append all the matches according to their clusters like lis1 consists of cluster 1, lis2 consists of cluster 2, and so on.

```
count=0
for i in predicted:
  listx1[count].append(i)
  count = count + 1

lis1=[]
lis2=[]
lis3=[]
lis4=[]
lis5=[]
j=0
i=0

for i in listx1:
  if i[3]==0:
    lis1.append(listx1[j])
  elif i[3]==1:
    lis2.append(listx1[j])
  elif i[3]==2:
    lis3.append(listx1[j])
```

Now in this all lists we need to find a match which the viewer had seen in the past (assumed) and then form a table of matches of similar clusters. Here we get to now out of 6 matches the viewer had 4 are from list 2. Therefore we can recommend matches from list2 to our viewer.

	0	1
0	Tobias Kamke	Paolo Lorenzi
1	Gilles Simon	Lleyton Hewitt
2	Radek Stepanek	Nick Kyrgios
3	Steve Johnson	Albert Montanes
4	Marcel Granollers	Feliciano Lopez
5	Jan-Lennard Struff	Evgeny Donskoy
6	Steve Darcis	Michael Llodra
7	Michal Przysiezny	Rhyne Williams
8	Kenny De Schepper	Robin Haase
9	Vasek Pospisil	Horacio Zeballos
10	Stanislas Wawrinka	Thiemo De Bakker
11	Rafael Nadal	Daniel Brands
12	Mikhail Youzhny	Pablo Andujar
13	Jiri Vesely	Philipp Kohlschreiber
14	Alexandr Dolgopolov	Dmitry Tursunov
15	Feliciano Lopez	Joao Sousa
16	Evgeny Donskoy	Kevin Anderson
17	Milos Raonic	Michael Llodra
18	Igor Sijsling	Tommy Robredo
19	Gael Monfils	Ernests Gulbis
20	Robin Haase	Jerzy Janowicz
21	Lukas Rosol	Fabio Fognini
22	Mikhail Youzhny	Federico Delbonis
23	Gilles Simon	Sam Querrey
24	Viktor Troicki	Marin Cilic
25	Stanislas Wawrinka	Jerzy Janowicz
26	Benoit Paire	Kei Nishikori
27	Rafael Nadal	Fabio Fognini
28	Gilles Simon	Roger Federer
29	Novak Djokovic	Philipp Kohlschreiber

This will preferably provide us best and recommend matches for the viewer.

4. At this stage we know that if a match needs to be interesting then it should be played with players of equal capacity and playing style. This will ensure whether the match will be engaging or a regular match. So the steps are to First make a new dataframe with players name and with their average playing styles.

```
player1_df = df4.groupby('Player1').agg({
    'BPW.1': 'mean',
    'NPA.1': 'mean',
    'NPW.1': 'mean',
```

```
}).reset_index().rename(columns={'Player1': 'Player'})

player2_df = df4.groupby('Player2').agg({
    'BPW.1': 'mean',
```

```
    'ST5.2': 'mean'
}).reset_index().rename(columns={'Player2': 'Player'}

player_matches_df = pd.concat([player1_df, player2_df
```

Now apply k means,

```
new_df = player_matches_df.iloc[:, 2:].copy()
```

```
num_clusters = 5
km = KMeans(num_clusters)
predicted = km.fit_predict(reduceddf)
listx1 = player_matches_df.values.tolist()
```

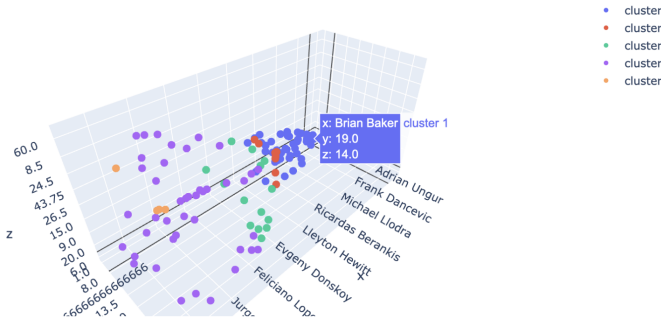
Here, we will plot 3D which will contain player names. For which we will put all clusters in different lists,

```
lis1=[]
lis2=[]
lis3=[]
lis4=[]
lis5=[]
j=0
i=0

for i in listx1:
  if i[28]==0:
    lis1.append(listx1[j])
  elif i[28]==1:
    lis2.append(listx1[j])
  elif i[28]==2:
    lis3.append(listx1[j])
  elif i[28]==3:
    lis4.append(listx1[j])
  elif i[28]==4:
    lis5.append(listx1[j])
  j= j + 1
```

and then plot a 3d graph using a go plot.

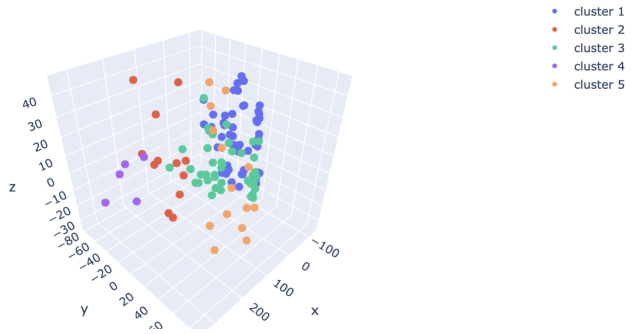
```
fig = go.Figure()
```



and also to plot we need to convert data points into 3d or 2d so we need to reduce it.

```
from sklearn.decomposition import PCA

pc = PCA(n_components = 3)
reduceddf = pc.fit_transform(new_df)
```



Here now we will get clusters and the players in the same cluster have most probably similar gameplay and if the match is played with the players of a similar cluster then it has the high probability that the match will be an interesting one.

5. First we need to make a new dataframe with players name and with their average playing styles (sum / total match). Styles here meant by FSW, SSW, ACE, BPC, BPW, TPW, DBF and UFE.

```
player1 = df5.groupby('Player1').agg({
    'BPW.1': 'mean',
    'NPA.1': 'mean',
    'NPW.1': 'mean',
```

```
}).reset_index().rename(columns={'Player1': 'Player'})

player2 = df5.groupby('Player2').agg({
    'BPW.1': 'mean',
```

```
}).reset_index().rename(columns={'Player2': 'Player'})

player_matches = pd.concat([player1, player2], ignore_i
# print(player_matches.to_markdown())
```

For this we need to apply k means on data and form 5 numbers of clusters.

```
new_df = player_matches.iloc[:, 2:].copy()

new_clusters = 5
km = KMeans(new_clusters)
predicted3d = km.fit_predict(new_df)
```

We need to add these clusters to players so we could get cluster number of bestplayer

```
player_matches['predicted'] = predicted
player_matches
```

Then inside these clusters we need to find the best players based on their TPW (total points win) so we need to make new column of TPW1 +TPW2

```
player_matches['TPW'] = player_matches['TPW.1'] + player_matches['TPW.2']
player_matches
```

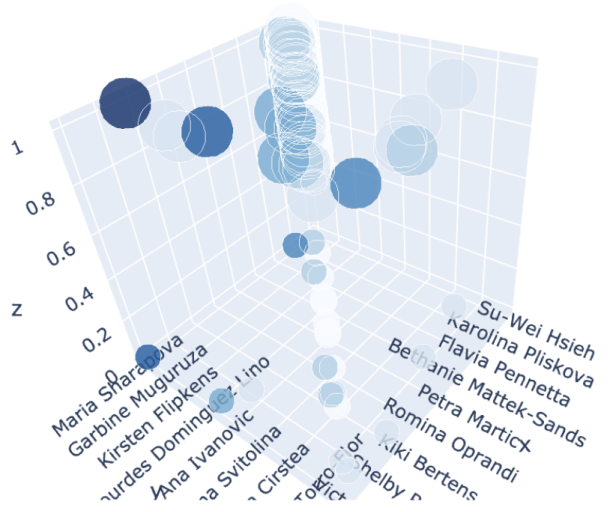
and give out rankings to these different groups according to best players. So we will sort it on the basis of TPW and see

which top 50 players are in which group. From the table of f20 we see that 8 are from cluster number 4, 7 are from cluster number 0 and 5 are from cluster number 3. Therefore, we can say cluster number 4 is at the top, next cluster number 0, then cluster number 3 and so on.

	Player	BPW.1	NPA.1	NPW.1	TPW.1	ST1.1	ST2
5	Alex Bogomolov Jr.	37.000000	60.000000	82.000000	364.000000	11.000000	12.0000
119	Tim Smyczek	30.500000	55.500000	78.500000	313.000000	9.500000	12.0000
15	Bernard Tomic	26.000000	44.000000	63.000000	282.000000	9.000000	9.0000
29	Dudi Sela	31.000000	0.000000	0.000000	257.000000	10.000000	7.0000
75	Marcel Granollers	24.000000	44.500000	69.500000	276.000000	5.500000	10.0000

ST1.2	ST2.2	ST3.2	ST4.2	ST5.2	predicted	TPW
10.0	9.000000	8.00	11.000000	13.0	3	714.000000
9.0	10.000000	4.00	9.500000	9.5	3	612.500000
7.0	9.000000	11.00	13.000000	6.0	3	580.000000
13.0	12.000000	12.00	5.000000	6.0	0	546.000000
12.0	9.500000	2.50	9.000000	11.5	3	542.500000
8.0	13.000000	9.50	10.000000	3.0	3	540.000000
9.0	12.000000	8.00	7.000000	6.0	0	515.000000

6.

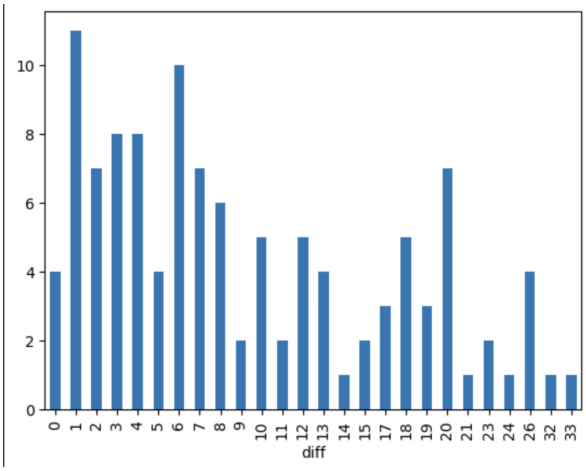


This plot is in 3d hence shows 3d dimensions and the other 2d is from the size and colour of the plot.

7. For this que we will take difference and find how many players are won by how many points. We also need to take modulus as player2 - player1 could come in negative.

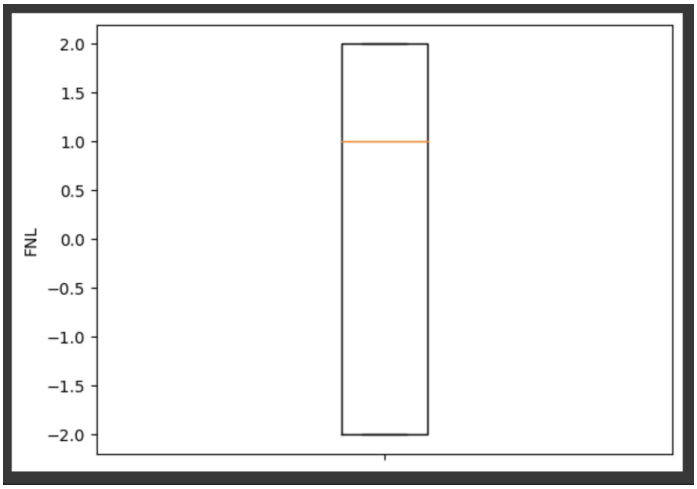
```
df8["diff"]=(df8["NPW.1"] - df8["NPW.2"]).abs()
```

This will also ensure that which was easy to win match (match with high difference) and which was very hard to win match (match with very low difference)



8.

```
df8['FNL'] = df8['FNL.1'] - df8['FNL.2']
fig, ax = plt.subplots()
ax.boxplot(df8['FNL'])
ax.set_ylabel('FNL')
ax.set_xticklabels([''])
plt.show()
```



VI. Unanswerable Questions, if any

Summary of the Observation that could be created by Unanswerable Questions

VIII. References

[1]The pandas development team. 2023. *Pandas-Dev/Pandas: Pandas*. Zenodo.

[2]"Matplotlib Tutorial." 2021. GeeksforGeeks. February 8, 2021. https://www.geeksforgeeks.org/matplotlib-tutorial/.

This are some web sources that I used in my code:

1. "Python • Sort Grouped Pandas Dataframe by Group Size." n.d. Tutorialspoint.com. Accessed March 30, 2023.

2. "How to Sort Data by Column in a CSV File in Python ?" 2021. GeeksforGeeks. March 30, 2021. https://www.geeksforgeeks.org/how-to-sort-data-by-column-in-a-csv-file-in-python/.

3. "Difference between '&' and 'and' in Pandas." n.d. Stack Overflow. Accessed March 30, 2023. https://stackoverflow.com/questions/54315627/difference between-and-and-in-pandas.

4. "Range Filtering with BETWEEN." n.d. Peachpit.com. Accessed March 30, 2023. https://www.peachpit.com/articles/article.aspx?p=1276352&seqNum=8.

5. "Pandas DataFrame Count() Method." n.d. W3schools.com. Accessed March 30, 2023. https://www.w3schools.com/python/pandas/ref_df_count. asp.

6. "Map Configuration and Styling in Python." n.d. Plotly.com. Accessed March 30, 2023. https://plotly.com/python/map-configuration/.

7. Ostwal, Prasad. 2019. "Multidimensional Plots in Python — From 3D to 6D. - Prasad Ostwal." Medium. May 28, 2019. https://medium.com/@prasadostwal/multi-dimension-plots-in-python-from-2d-to-6d-9a2bf7b8cc74.

IX. Acknowledgements

My sincere gratitude goes out to Sir Shanmuga R, our course lecturer for ES 114 Probability, Statistics, and Data Visualization, as well as to all of the TAs who helped me out whenever I needed assistance with this assignment.

First and foremost, I want to express my gratitude to all the online resources that provided me with the necessary resources to finish this task. Also, I want to thank all of the organizations that have helped me by lending me their knowledge and insight in a variety of sectors through a variety of additional channels. Finally, I want to express my gratitude for the chance you offered me to ask questions, learn how to interpret facts, and acquire this incredible skill.