BSD2333 DATA WRANGLING

'BE A DATA WRANGLER'

MATRIC NUMBERS	NAMES
SD20025	EIFFEL SYAFHAMKA NADIMEN A/L SYAMSUL
SD20033	HAFIZ ASYRAAF BIN GHANI
SD20039	MUHAMMAD 'ARIF BIN MOHD ANUWA
SD20052	NURUL ASHYIBAH BINTI MAZLAN
SD20059	NUR AIDA IZZATI BINTI KAMARULAZUA

2021/2022

TABLE OF CONTENTS

1.0 SYNOPSIS

1.1 Description of the assignment

The title of our assignment is The Wrang-Maker that discuss about the football player's performance in the previous season. In this assignment, the dataset used is "English Premier League (EPL) Player in 2021 statistics" in the form of CSV file referred from Kaggel. In short brief, the Premier League is an English professional league for association football clubs. The dataset has common attributes i.e. Name, Position, Appearances, and the statistics of the player's performance throughout the season. Various data preprocessing steps were performed like omitting columns with too many null values, exchanging the null values into zeros and renaming rows.

1.2 Problem to be solved

Using individual performance in the previous season to predict the rating of players, some examples of variables used to determine the extent of the teams' weaknesses such as yellow and red cards served, fouls against the team, shots on target and offsides. For a list of example variables that were used to determine the severity of the teams' strengths are goals, shooting accuracy, accurate long balls and tackle success. With the given variables, it can be used to track and predict sport performance of the athletes which can provide many advantages. Examples of the advantages are it can help coaches find a rising star in sports, it can help coaches and athletes to develop effective training plans or it can help coaches and athletes master the opponent's habits and specialties in the game to make value judgement in the game. For example by analyzing each athlete's recent game performance, the coach can make the right decisions in selecting players for the game.

1.3 Question to be answered

The questions that will be answered using the English Premier League(EPL) Player statistics data set:

- Who has the most fouls?
- Who is the top scorer?
- Who is the greatest goalkeeper?
- Who is the best midfielder?
- Who is the solid defender?
- Who is the top forward?

1.4 Objectives

With the given individual performance in the previous season and their rating in the previous version of FIFA20 game, we were attempting to forecast the rating of players in the FIFA21 game. The project is aimed at studying the kaggle football dataset, to analyse, extract information from it and make predictions based on the data.

1.5 Data Description

The dataset of "English Premier League (EPL) Player 2021 statistics" contains 46548 data. The data contains 54 columns. The dataset has common attributes. Name, Position, Appearances, and the statistics of the player's performance throughout the season.

2.0 PACKAGES REQUIRED

The packages used are :-

Numpy- used for working with arrays

pandas- used for working with data sets to analyse data

matplotlib.pyplot- used for data visualisation using plotting

Seaborn- used with matplotlib to visualise random distributions

3.0 DATA PREPARATION

3.1 Data Import

Import the dataset from the csv file and view the output

```
data = pd.read_csv("epl21.csv")
data
```

	Unnamed: 0	Name	Position	Appearances	Clean sheets	Goals Conceded	Tackles	Tackle success %	Last man tackles	Blocked shots	...	Shooting accuracy %	Big chances missed	Saves	Penalties Saved	Punches	High Claims
0	0	Tammy Abraham	Forward	22	NaN	NaN	7.0	NaN	NaN	5.0	...	41%	4.0	NaN	NaN	NaN	NaN
1	1	Che Adams	Forward	36	NaN	NaN	25.0	NaN	NaN	10.0	...	56%	17.0	NaN	NaN	NaN	NaN
2	2	Tosin Adarabioyo	Defender	33	9.0	41.0	37.0	51%	0.0	2.0	...	NaN	NaN	NaN	NaN	NaN	NaN
3	3	Dennis Adeniran	Midfielder	0	NaN	NaN	0.0	0%	NaN	0.0	...	0%	0.0	NaN	NaN	NaN	NaN
4	4	Adrián	Goalkeeper	3	1.0	9.0	NaN	NaN	NaN	NaN	...	NaN	NaN	10.0	0.0	3.0	0.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
857	857	Hakim Ziyech	Midfielder	23	NaN	NaN	14.0	71%	NaN	16.0	...	37%	1.0	NaN	NaN	NaN	NaN
858	858	Kenneth Zohore	Forward	0	NaN	NaN	0.0	NaN	NaN	0.0	...	0%	0.0	NaN	NaN	NaN	NaN
859	859	Kurt Zouma	Defender	24	9.0	25.0	16.0	63%	0.0	6.0	...	NaN	NaN	NaN	NaN	NaN	NaN

3.2 Data Cleaning

For cleaning we need to know the data type for each column.

```
data.dtypes
```

```
Unnamed: 0                 int64
Name                      object
Position                  object
Appearances                int64
Clean sheets             float64
Goals Conceded           float64
Tackles                  float64
Tackle success %          object
Last man tackles         float64
Blocked shots            float64
Interceptions            float64
Clearances               float64
Headed Clearance         float64
Clearances off line      float64
Recoveries               float64
Duels won                float64
Duels lost               float64
Successful 50/50s        float64
Aerial battles won       float64
Aerial battles lost      float64
Own goals                float64
Errors leading to goal   float64
Assists                    int64
Passes                    object
Passes per match         float64
Big Chances Created      float64
Crosses                  float64
```

Then check for null values using isnull()

```
#tmpt true ade null value
data.isnull()
```

	Unnamed: 0	Name	Position	Appearances	Clean sheets	Goals Conceded	Tackles	Tackle success %	Last man tackles	Blocked shots	...	Shooting accuracy %	Big chances missed	Saves	Penalties Saved	Punches	High Claims	Catches
0	False	False	False	False	True	True	False	True	True	False	...	False	False	True	True	True	True	True
1	False	False	False	False	True	True	False	True	True	False	...	False	False	True	True	True	True	True
2	False	False	False	False	False	False	False	False	False	False	...	True	True	True	True	True	True	True
3	False	False	False	False	True	True	False	False	True	False	...	False	False	True	True	True	True	True
4	False	False	False	False	False	False	True	True	True	True	...	True	True	False	False	False	False	False
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
857	False	False	False	False	True	True	False	False	True	False	...	False	False	True	True	True	True	True
858	False	False	False	False	True	True	False	True	True	False	...	False	False	True	True	True	True	True
859	False	False	False	False	False	False	False	False	False	False	...	True	True	True	True	True	True	True
860	False	False	False	False	False	False	True	True	True	True	...	True	True	False	False	False	False	False
861	False	False	False	False	True	True	False	False	True	False	...	False	False	True	True	True	True	True

862 rows × 54 columns

Since there is null value, check which column has the highest number of null

```
[ ] check_null_value = pd.isnull(data["Penalties Saved"])
    data[check_null_value]
```

	Unnamed: 0	Name	Position	Appearances	Clean sheets	Goals Conceded	Tackles	Tackle success %	Last man tackles	Blocked shots	...	Shooting accuracy %	Big chances missed	Saves	Penalties Saved	Punches	High Claims	Ca
0	0	Tammy Abraham	Forward	22	NaN	NaN	7.0	NaN	NaN	5.0	...	41%	4.0	NaN	NaN	NaN	NaN	
1	1	Che Adams	Forward	36	NaN	NaN	25.0	NaN	NaN	10.0	...	56%	17.0	NaN	NaN	NaN	NaN	
2	2	Tosin Adarabioyo	Defender	33	9.0	41.0	37.0	51%	0.0	2.0	...	NaN	NaN	NaN	NaN	NaN	NaN	
3	3	Dennis Adeniran	Midfielder	0	NaN	NaN	0.0	0%	NaN	0.0	...	0%	0.0	NaN	NaN	NaN	NaN	
5	5	Adrien Silva	Midfielder	0	NaN	NaN	0.0	0%	NaN	0.0	...	0%	0.0	NaN	NaN	NaN	NaN	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
856	856	Richairo Zivkovic	Forward	0	NaN	NaN	0.0	NaN	NaN	0.0	...	0%	0.0	NaN	NaN	NaN	NaN	
857	857	Hakim Ziyech	Midfielder	23	NaN	NaN	14.0	71%	NaN	16.0	...	37%	1.0	NaN	NaN	NaN	NaN	

```
[ ] check_null_value = pd.isnull(data["Punches"])
    data[check_null_value]
```

	Unnamed: 0	Name	Position	Appearances	Clean sheets	Goals Conceded	Tackles	Tackle success %	Last man tackles	Blocked shots	...	Shooting accuracy %	Big chances missed	Saves	Penalties Saved	Punches	High Claims	Ca
0	0	Tammy Abraham	Forward	22	NaN	NaN	7.0	NaN	NaN	5.0	...	41%	4.0	NaN	NaN	NaN	NaN	
1	1	Che Adams	Forward	36	NaN	NaN	25.0	NaN	NaN	10.0	...	56%	17.0	NaN	NaN	NaN	NaN	
2	2	Tosin Adarabioyo	Defender	33	9.0	41.0	37.0	51%	0.0	2.0	...	NaN	NaN	NaN	NaN	NaN	NaN	
3	3	Dennis Adeniran	Midfielder	0	NaN	NaN	0.0	0%	NaN	0.0	...	0%	0.0	NaN	NaN	NaN	NaN	
5	5	Adrien Silva	Midfielder	0	NaN	NaN	0.0	0%	NaN	0.0	...	0%	0.0	NaN	NaN	NaN	NaN	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
856	856	Richairo Zivkovic	Forward	0	NaN	NaN	0.0	NaN	NaN	0.0	...	0%	0.0	NaN	NaN	NaN	NaN	
857	857	Hakim Ziyech	Midfielder	23	NaN	NaN	14.0	71%	NaN	16.0	...	37%	1.0	NaN	NaN	NaN	NaN	

```
[ ] check_null_value = pd.isnull(data["High Claims"])
    data[check_null_value]
```

	Unnamed: 0	Name	Position	Appearances	Clean sheets	Goals Conceded	Tackles	Tackle success %	Last man tackles	Blocked shots	...	Shooting accuracy %	Big chances missed	Saves	Penalties Saved	Punches	High Claims	Ca
0	0	Tammy Abraham	Forward	22	NaN	NaN	7.0	NaN	NaN	5.0	...	41%	4.0	NaN	NaN	NaN	NaN	
1	1	Che Adams	Forward	36	NaN	NaN	25.0	NaN	NaN	10.0	...	56%	17.0	NaN	NaN	NaN	NaN	
2	2	Tosin Adarabioyo	Defender	33	9.0	41.0	37.0	51%	0.0	2.0	...	NaN	NaN	NaN	NaN	NaN	NaN	
3	3	Dennis Adeniran	Midfielder	0	NaN	NaN	0.0	0%	NaN	0.0	...	0%	0.0	NaN	NaN	NaN	NaN	
5	5	Adrien Silva	Midfielder	0	NaN	NaN	0.0	0%	NaN	0.0	...	0%	0.0	NaN	NaN	NaN	NaN	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
856	856	Richairo Zivkovic	Forward	0	NaN	NaN	0.0	NaN	NaN	0.0	...	0%	0.0	NaN	NaN	NaN	NaN	
857	857	Hakim Ziyech	Midfielder	23	NaN	NaN	14.0	71%	NaN	16.0	...	37%	1.0	NaN	NaN	NaN	NaN	

```
[ ] check_null_value = pd.isnull(data["Catches"])
    data[check_null_value]
```

	Unnamed: 0	Name	Position	Appearances	Clean sheets	Goals Conceded	Tackles	Tackle success %	Last man tackles	Blocked shots	...	Shooting accuracy %	Big chances missed	Saves	Penalties Saved	Punches	High Claims	Ca
0	0	Tammy Abraham	Forward	22	NaN	NaN	7.0	NaN	NaN	5.0	...	41%	4.0	NaN	NaN	NaN	NaN	
1	1	Che Adams	Forward	36	NaN	NaN	25.0	NaN	NaN	10.0	...	56%	17.0	NaN	NaN	NaN	NaN	
2	2	Tosin Adarabioyo	Defender	33	9.0	41.0	37.0	51%	0.0	2.0	...	NaN	NaN	NaN	NaN	NaN	NaN	
3	3	Dennis Adeniran	Midfielder	0	NaN	NaN	0.0	0%	NaN	0.0	...	0%	0.0	NaN	NaN	NaN	NaN	
5	5	Adrien Silva	Midfielder	0	NaN	NaN	0.0	0%	NaN	0.0	...	0%	0.0	NaN	NaN	NaN	NaN	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
856	856	Richairo Zivkovic	Forward	0	NaN	NaN	0.0	NaN	NaN	0.0	...	0%	0.0	NaN	NaN	NaN	NaN	
857	857	Hakim Ziyech	Midfielder	23	NaN	NaN	14.0	71%	NaN	16.0	...	37%	1.0	NaN	NaN	NaN	NaN	

```
[ ] check_null_value = pd.isnull(data["Throw outs"])
    data[check_null_value]
```

	Unnamed: 0	Name	Position	Appearances	Clean sheets	Goals Conceded	Tackles	Tackle success %	Last man tackles	Blocked shots	...	Shooting accuracy %	Big chances missed	Saves	Penalties Saved	Punches	High Claims	Ca
0	0	Tammy Abraham	Forward	22	NaN	NaN	7.0	NaN	NaN	5.0	...	41%	4.0	NaN	NaN	NaN	NaN	
1	1	Che Adams	Forward	36	NaN	NaN	25.0	NaN	NaN	10.0	...	56%	17.0	NaN	NaN	NaN	NaN	
2	2	Tosin Adarabioyo	Defender	33	9.0	41.0	37.0	51%	0.0	2.0	...	NaN	NaN	NaN	NaN	NaN	NaN	
3	3	Dennis Adeniran	Midfielder	0	NaN	NaN	0.0	0%	NaN	0.0	...	0%	0.0	NaN	NaN	NaN	NaN	
5	5	Adrien Silva	Midfielder	0	NaN	NaN	0.0	0%	NaN	0.0	...	0%	0.0	NaN	NaN	NaN	NaN	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
856	856	Richairo Zivkovic	Forward	0	NaN	NaN	0.0	NaN	NaN	0.0	...	0%	0.0	NaN	NaN	NaN	NaN	
857	857	Hakim Ziyech	Midfielder	23	NaN	NaN	14.0	71%	NaN	16.0	...	37%	1.0	NaN	NaN	NaN	NaN	

```
[ ] data['Through balls'].isna().sum()

281

[ ] data['Accurate long balls'].isna().sum()

176

[ ] data['Own goals'].isna().sum()

482

[ ] data['Big Chances Created'].isna().sum()

105

[ ] data['Cross accuracy %'].isna().sum()

281
```

The total data that has null value was

```
[ ] ##TOTAL NUMBER OF NULL VALUES
    data.isna().sum().sum()

15477
```

Then we drop some columns that has null values to tidy up the data

```
[ ] data.drop(('Last man tackles'), axis = 1, inplace = True)
```

```
[ ] data.drop(('Punches'), axis = 1, inplace = True)
```

```
[ ] data.drop(('High Claims'), axis = 1, inplace = True)
```

```
[ ] data.drop(('Clearances off line'), axis = 1, inplace = True)
```

```
[ ] data.drop(('Throw outs'), axis = 1, inplace = True)
```

VIEW ALL DATA AFTER DROP

```
[ ] data
```

The data will decreases after we drop the columns

```
[ ] data.isna().sum().sum()

12032
```

Then we will replace the NaN value with 0

```
[ ] data_fillna = data.fillna(0)
```

3.3 Data Preview

View the data that has been clean. Then, declare it as data_new

```
[ ] data_new = data_fillna[data_fillna["Position"].isin(['Forward','Midfielder','Defender','Goalkeeper'])]
    data_new
```

	Unnamed: 0	Name	Position	Appearances	Clean sheets	Goals Conceded	Tackles	Tackle success %	Blocked shots	Interceptions	...	Freekicks scored	Shots	Shots on target	Shooting accuracy %	Big chances missed	Sav
0	0	Tammy Abraham	Forward	22	0.0	0.0	7.0	0	5.0	5.0	...	0.0	32.0	13.0	41%	4.0	[illegible]
1	1	Che Adams	Forward	36	0.0	0.0	25.0	0	10.0	4.0	...	0.0	55.0	31.0	56%	17.0	[illegible]
2	2	Tosin Adarabioyo	Defender	33	9.0	41.0	37.0	51%	2.0	42.0	...	0.0	0.0	0.0	0	0.0	[illegible]
3	3	Dennis Adeniran	Midfielder	0	0.0	0.0	0.0	0%	0.0	0.0	...	0.0	0.0	0.0	0%	0.0	[illegible]
4	4	Adrián	Goalkeeper	3	1.0	9.0	0.0	0	0.0	0.0	...	0.0	0.0	0.0	0	0.0	[illegible]
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
857	857	Hakim Ziyech	Midfielder	23	0.0	0.0	14.0	71%	16.0	10.0	...	0.0	35.0	13.0	37%	1.0	[illegible]
858	858	Kenneth Zohore	Forward	0	0.0	0.0	0.0	0	0.0	0.0	...	0.0	0.0	0.0	0%	0.0	[illegible]
859	859	Kurt Zouma	Defender	24	9.0	25.0	16.0	63%	6.0	28.0	...	0.0	0.0	0.0	0	0.0	[illegible]

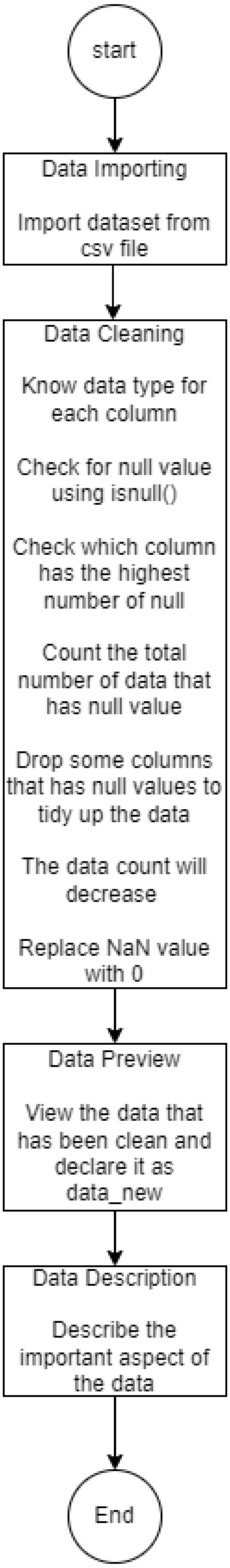
3.4 Data Description,

Describe the important aspect of the data.

Data Variable	Data description
Name	The player’s name
Appearances	The number of time the player played for the team

Position	The player role in the game
Yellow Cards	The warning given by the referee
Red Cards	The dismissal given given by the referee
Fouls	The unfair act committed by a player
Clearances	This is a defensive action where a player kicks the ball away from his own goal with no intended recipient.
Tackles	When a player stop the possession and ball movement from the opponent without committing a foul
Goals	The score when a team kick the ball into the goal post
Sweeper clearances	Given anytime a goalkeeper anticipates danger and rushes off their line to try to either cut out an attacking pass (in a race with the opposition player) or to close-down an opposition player.
Penalties Saved	A goalkeeper preventing the ball from entering the goal with any part of his body when facing an intentional attempt from an opposition player during penalties.
Clean sheets	A player or team who does not concede a goal for the full match.
Saves	Awarded to goalkeeper when the shot is block from goal
Blocked shots	Attempt to score that is blocked by other player
Interceptions	The act of getting possession of the ball following an attempted pass or shot from a player on the opposing team
Goals per match	Number of goal score in a match
Shooting accuracy%	A calculation of Shots on target divided by all shots
Big chances missed	A big chance opportunity when the player does not get a shot away, typically given for big chance attempts where the player shooting completely misses the ball (air shot) but can also be given when the player has a big chance opportunity to shoot and decides not to, resulting in no attempt occurring in that attack.

3.5 Flow chart


start
Data Importing
Import dataset from csv file
Data Cleaning
Know data type for each column
Check for null value using isnull()
Check which column has the highest number of null
Count the total number of data that has null value
Drop some columns that has null values to tidy up the data
The data count will decrease
Replace NaN value with 0
Data Preview
View the data that has been clean and declare it as data_new
Data Description
Describe the important aspect of the data
End

4.0 EXPLORATORY DATA ANALYSIS

VISUALISE USING MATPLOTLIB

Analyse the position who has the most fouls.

```
df_fouls = data_new[['Name', 'Position', 'Yellow cards','Red cards', 'Fouls']]
df_fouls.head()
```

	Name	Position	Yellow cards	Red cards	Fouls
0	Tammy Abraham	Forward	0	0	22
1	Che Adams	Forward	1	0	30
2	Tosin Adarabioyo	Defender	1	0	16
3	Dennis Adeniran	Midfielder	0	0	0
4	Adrián	Goalkeeper	0	0	0

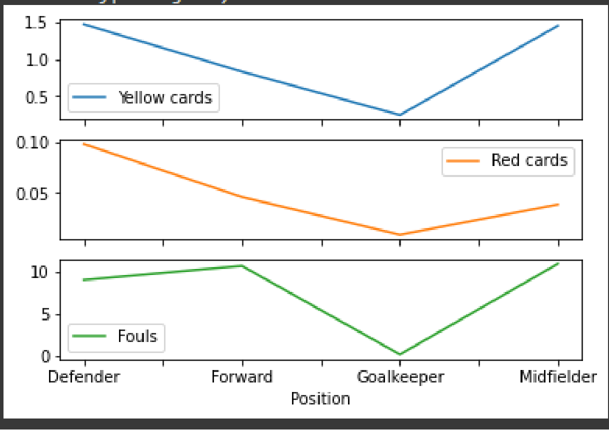
```
df_fouls.dtypes
```

```
Name            object
Position        object
Yellow cards     int64
Red cards        int64
Fouls            int64
dtype: object
```

```
viz_fouls = pd.DataFrame(df_fouls)
```

```
viz_fouls.groupby('Position').mean().plot(subplots = True)
```

```
array([<matplotlib.axes._subplots.AxesSubplot object at 0x7f8d8bc5d210>,
       <matplotlib.axes._subplots.AxesSubplot object at 0x7f8d8bc8d9d0>,
       <matplotlib.axes._subplots.AxesSubplot object at 0x7f8d8bcc1250>],
      dtype=object)
```



Based on the graph, the midfielder has the highest fouls recorded. But, defender got the highest in yellow card and red card.

Analyse of the player who recorded the most fouls.

df_mostfouls = data_new[['Name', 'Position', 'Fouls']]
df_mostfouls

```
[134] #Summary to most fouls appeared
      df_mostfouls = data_new[['Name', 'Position', 'Fouls']]
      df_mostfouls
```

	Name	Position	Fouls
0	Tammy Abraham	Forward	22
1	Che Adams	Forward	30
2	Tosin Adarabioyo	Defender	16
3	Dennis Adeniran	Midfielder	0
4	Adrián	Goalkeeper	0
...	...	...	...
857	Hakim Ziyech	Midfielder	19
858	Kenneth Zohore	Forward	0
859	Kurt Zouma	Defender	17
860	Oliwer Zych	Goalkeeper	0
861	Martin Ødegaard	Midfielder	1

862 rows × 3 columns

#Summary most fouls in each position

viz_mostfouls=df_mostfouls.sort_values(by=['Fouls'], ascending=False) viz_mostfouls.head(5)

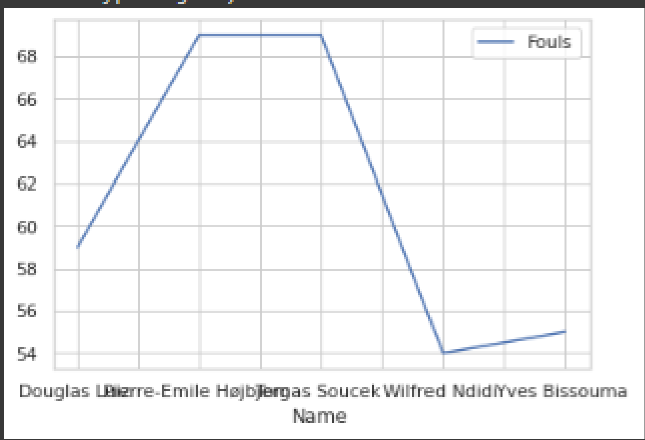
```
[135] #Summary most fouls in each position
      viz_mostfouls=df_mostfouls.sort_values(by=['Fouls'], ascending=False)
      viz_mostfouls.head(5)
```

	Name	Position	Fouls
364	Pierre-Emile Højbjerg	Midfielder	69
740	Tomas Soucek	Midfielder	69
223	Douglas Luiz	Midfielder	59
101	Yves Bissouma	Midfielder	55
572	Wilfred Ndidi	Midfielder	54

viz_mostfouls.head().groupby('Name').mean().plot(subplots = True)

```
[136] viz_mostfouls.head().groupby('Name').mean().plot(subplots = True)

array([<matplotlib.axes._subplots.AxesSubplot object at 0x7fdfec237890>],
      dtype=object)
```



To be specific, Pierre-Emile Højbjerg and Tomas Soucek shared the most fouls in EPL 2021. They had 69 recorded fouls during that season.

Analysis of the player who recorded the top scorer.

df_topscorer = data_new[['Name', 'Position', 'Goals', 'Appearances']] df_topscorer

	Name	Position	Goals	Appearances
0	Tammy Abraham	Forward	6	22
1	Che Adams	Forward	9	36
2	Tosin Adarabioyo	Defender	0	33
3	Dennis Adeniran	Midfielder	0	0
4	Adrián	Goalkeeper	0	3
...	...	...	...	...
857	Hakim Ziyech	Midfielder	2	23
858	Kenneth Zohore	Forward	0	0
859	Kurt Zouma	Defender	5	24
860	Oliwer Zych	Goalkeeper	0	0
861	Martin Ødegaard	Midfielder	1	14

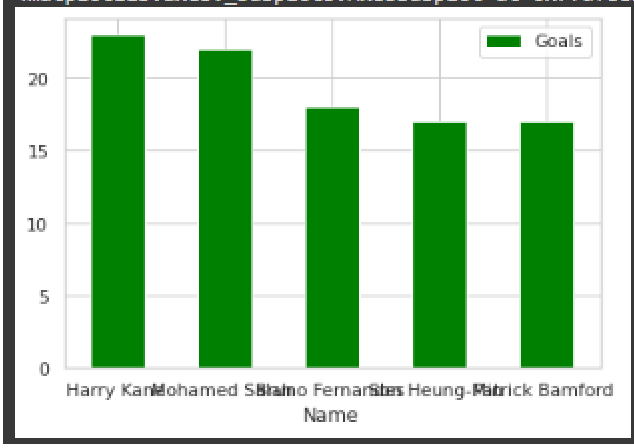
862 rows × 4 columns

```
viz_topscorer = df_topscorer.sort_values(by=['Goals'], ascending=False)
head_scorer=viz_topscorer.head()
head_scorer
```

	Name	Position	Goals	Appearances
405	Harry Kane	Forward	23	35
550	Mohamed Salah	Forward	22	37
122	Bruno Fernandes	Midfielder	18	37
737	Son Heung-Min	Forward	17	37
67	Patrick Bamford	Forward	17	38

```
head_scorer.plot.bar(x='Name', y='Goals', rot=0, color='green')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fdfee1c2710>
```



The graph shown, the top scorer is Harry Kane which is 23 goals followed by Mohamed Salah which is 22 goals.

VISUALISE USING SEABORN

Analyse the best goalkeeper

```
df_gk = data_new.query("Position=='Goalkeeper'")
df_gk
```

	Unnamed: 0	Name	Position	Appearances	Clean sheets	Goals Conceded	Tackles	Tackle success %	Blocked shots	Interceptions	...	Freekicks scored	Shots	Shots on target	Shooting accuracy %	Big chances missed	Saves	Penalties Saved	Catc
4	4	Adrián	Goalkeeper	3	1.0	9.0	0.0	0	0.0	0.0	...	0.0	0.0	0.0	0	0.0	10.0	0.0	
22	22	Alisson	Goalkeeper	33	10.0	32.0	0.0	0	0.0	0.0	...	0.0	0.0	0.0	0	0.0	84.0	1.0	
32	32	Joseph Anang	Goalkeeper	0	0.0	0.0	0.0	0	0.0	0.0	...	0.0	0.0	0.0	0	0.0	0.0	0.0	
41	41	Alphonse Areola	Goalkeeper	36	9.0	48.0	0.0	0	0.0	0.0	...	0.0	0.0	0.0	0	0.0	116.0	0.0	
43	43	Kepa Arrizabalaga	Goalkeeper	7	2.0	8.0	0.0	0	0.0	0.0	...	0.0	0.0	0.0	0	0.0	16.0	0.0	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
814	814	Christian Walton	Goalkeeper	0	0.0	0.0	0.0	0	0.0	0.0	...	0.0	0.0	0.0	0	0.0	0.0	0.0	
828	828	Alfie Whiteman	Goalkeeper	0	0.0	0.0	0.0	0	0.0	0.0	...	0.0	0.0	0.0	0	0.0	0.0	0.0	
838	838	Ben Winterbottom	Goalkeeper	0	0.0	0.0	0.0	0	0.0	0.0	...	0.0	0.0	0.0	0	0.0	0.0	0.0	
854	854	Karlo Ziger	Goalkeeper	0	0.0	0.0	0.0	0	0.0	0.0	...	0.0	0.0	0.0	0	0.0	0.0	0.0	
860	860	Oliwer Zych	Goalkeeper	0	0.0	0.0	0.0	0	0.0	0.0	...	0.0	0.0	0.0	0	0.0	0.0	0.0	

105 rows × 49 columns

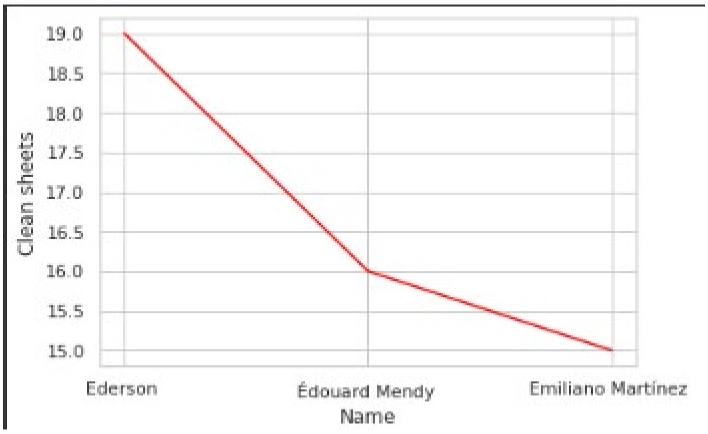
```
gk_data = df_gk[['Name','Saves','Sweeper clearances','Penalties Saved','Clean sheets']].nlargest(3,
['Clean sheets'])
gk_data
```

	Name	Saves	Sweeper clearances	Penalties Saved	Clean sheets
235	Ederson	66.0	16.0	1.0	19.0
533	Édouard Mendy	57.0	9.0	1.0	16.0
244	Emiliano Martínez	142.0	12.0	1.0	15.0

```
gk_data.dtypes
```

```
Name                  object
Saves                float64
Sweeper clearances   float64
Penalties Saved      float64
Clean sheets         float64
dtype: object
```

```
sns.lineplot(gk_data["Name"], gk_data["Clean sheets"], color='red')
```



The data shown, Ederson is the best goalkeeper due to the most clean sheets he has which is 19 clean sheets in 38 games.

Analyse the solid defender.

```
df_def = data_new.query("Position=='Defender'")
df_def
```

	Unnamed: 0	Name	Position	Appearances	Clean sheets	Goals Conceded	Tackles	Tackle success %	Blocked shots	Interceptions	...	Freekicks scored	Shots	Shots on target	Shooting accuracy %	Big chances missed	Saves	Penalties Saved	Catches	Sweeper clearances	Goal Kicks
2	2	Tosin Adarabioyo	Defender	33	9.0	41.0	37.0	51%	2.0	42.0	...	0.0	0.0	0.0	0	0.0	0.0	0.0	0.0	0.0	0.0
8	8	Ahmed El Mohamady	Defender	14	3.0	9.0	17.0	76%	1.0	17.0	...	0.0	0.0	0.0	0	0.0	0.0	0.0	0.0	0.0	0.0
9	9	Ahmed Hegazi	Defender	1	1.0	0.0	0.0	0%	0.0	1.0	...	0.0	0.0	0.0	0	0.0	0.0	0.0	0.0	0.0	0.0
10	10	Ola Aina	Defender	31	7.0	36.0	37.0	54%	3.0	45.0	...	0.0	0.0	0.0	0	0.0	0.0	0.0	0.0	0.0	0.0
11	11	Rayan Aït-Nouri	Defender	21	3.0	23.0	29.0	55%	5.0	15.0	...	0.0	0.0	0.0	0	0.0	0.0	0.0	0.0	0.0	0.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
846	846	DeAndre Yedlin	Defender	6	1.0	6.0	9.0	44%	1.0	7.0	...	0.0	0.0	0.0	0	0.0	0.0	0.0	0.0	0.0	0.0
848	848	Maya Yoshida	Defender	0	0.0	0.0	0.0	0%	0.0	0.0	...	0.0	0.0	0.0	0	0.0	0.0	0.0	0.0	0.0	0.0
852	852	Davide Zappacosta	Defender	0	0.0	0.0	0.0	0%	0.0	0.0	...	0.0	0.0	0.0	0	0.0	0.0	0.0	0.0	0.0	0.0
855	855	Oleksandr Zinchenko	Defender	20	0.0	0.0	25.0	60%	7.0	23.0	...	0.0	16.0	4.0	25%	0.0	0.0	0.0	0.0	0.0	0.0
859	859	Kurt Zouma	Defender	24	9.0	25.0	16.0	63%	6.0	28.0	...	0.0	0.0	0.0	0	0.0	0.0	0.0	0.0	0.0	0.0

276 rows × 49 columns

```
def_data = df_def[['Name','Tackles','Blocked shots','Clearances','Appearances']].nlargest(3,
['Clearances'])
def_data
```

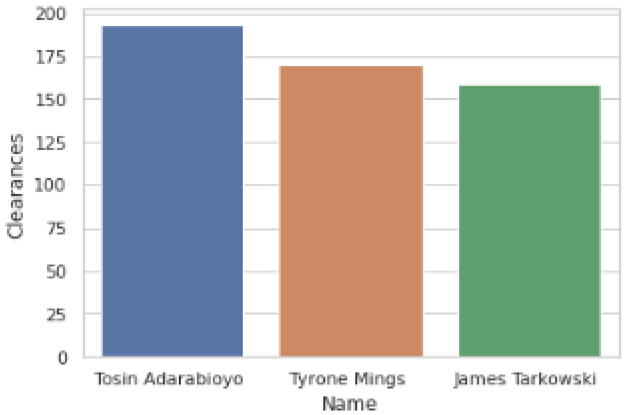
	Name	Tackles	Blocked shots	Clearances	Appearances
2	Tosin Adarabioyo	37.0	2.0	193.0	33
543	Tyrone Mings	32.0	4.0	170.0	36
763	James Tarkowski	66.0	1.0	159.0	36

```
def_data.dtypes
```

```
Name             object
Tackles         float64
Blocked shots   float64
Clearances      float64
Appearances       int64
dtype: object
```

```
sns.set(style='whitegrid')

sns.barplot(data=def_data, x='Name', y= 'Clearances')
plt.show()
```



Based on the data, there are 3 top players who got the most clearances. Tosin Adarabioyo recorded the most clearances , 193 clearances.

Analyse the top midfielder.

```
df_mid = data_new.query("Position=='Midfielder'")
df_mid
```

	Unnamed: 0	Name	Position	Appearances	Clean sheets	Goals Conceded	Tackles	Tackle success %	Blocked shots	Interceptions	...	Freekicks scored	Shots	Shots on target	Shooting accuracy %	B chances missed	Saves	Saved	Catches
3	3	Dennis Adeniran	Midfielder	0	0.0	0.0	0.0	0%	0.0	0.0	...	0.0	0.0	0.0	0%	0.0	0.0	0.0	0.0
5	5	Adrien Silva	Midfielder	0	0.0	0.0	0.0	0%	0.0	0.0	...	0.0	0.0	0.0	0%	0.0	0.0	0.0	0.0
15	15	Marc Albrighton	Midfielder	31	0.0	0.0	33.0	58%	7.0	17.0	...	0.0	21.0	10.0	48%	1.0	0.0	0.0	0.0
21	21	Ezgjan Alioski	Midfielder	36	0.0	0.0	74.0	42%	7.0	23.0	...	0.0	25.0	8.0	32%	1.0	0.0	0.0	0.0
23	23	Allan	Midfielder	24	0.0	0.0	80.0	56%	1.0	19.0	...	0.0	7.0	2.0	29%	0.0	0.0	0.0	0.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
847	847	Okay Yokuslu	Midfielder	16	0.0	0.0	38.0	66%	3.0	35.0	...	0.0	12.0	0.0	0%	1.0	0.0	0.0	0.0
849	849	Brad Young	Midfielder	0	0.0	0.0	0.0	0%	0.0	0.0	...	0.0	0.0	0.0	0%	0.0	0.0	0.0	0.0
851	851	André-Frank Zambo Anguissa	Midfielder	36	0.0	0.0	78.0	47%	10.0	56.0	...	0.0	34.0	9.0	26%	0.0	0.0	0.0	0.0
857	857	Hakim Ziyech	Midfielder	23	0.0	0.0	14.0	71%	16.0	10.0	...	0.0	35.0	13.0	37%	1.0	0.0	0.0	0.0
861	861	Martin Ødegaard	Midfielder	14	0.0	0.0	9.0	33%	9.0	2.0	...	0.0	15.0	3.0	20%	0.0	0.0	0.0	0.0

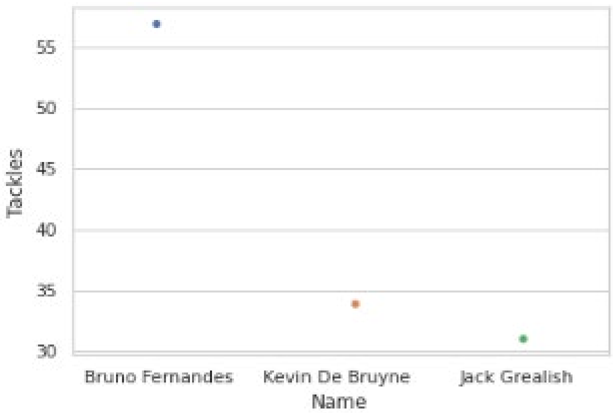
```
mid_data = df_mid[['Name','Interceptions','Assists','Tackles','Appearances']].nlargest(5, ['Assists'])
mid_data
```

	Name	Interceptions	Assists	Tackles	Appearances
122	Bruno Fernandes	26.0	12	57.0	37
197	Kevin De Bruyne	9.0	12	34.0	25
315	Jack Grealish	12.0	10	31.0	26

```
mid_data.dtypes
```

```
Name             object
Interceptions   float64
Assists           int64
Tackles         float64
Appearances       int64
dtype: object
```

```
sns.stripplot(x='Name',y='Tackles',data=mid_data)
plt.show()
```



Based on the graph shown, Bruno Fernandes has recorded the most assists which is 12 assists for midfielder. It shown that he is the best midfielder during that season.

Analyse the best forward.

```
df_fwd = data_new.query("Position=='Forward'")
df_fwd
```

	Unnamed: 0	Name	Position	Appearances	Clean sheets	Goals Conceded	Tackles	Tackle success %	Blocked shots	Interceptions	...	Freekicks scored	Shots	Shots on target	Shooting accuracy %	Big chances missed	Saves	Penalties Saved	Catches	Sweeper clearances	Goal Kicks
0	0	Tammy Abraham	Forward	22	0.0	0.0	7.0	0	5.0	5.0	...	0.0	32.0	13.0	41%	4.0	0.0	0.0	0.0	0.0	0.0
1	1	Che Adams	Forward	36	0.0	0.0	25.0	0	10.0	4.0	...	0.0	55.0	31.0	56%	17.0	0.0	0.0	0.0	0.0	0.0
6	6	Oladapo Afolayan	Forward	0	0.0	0.0	0.0	0	0.0	0.0	...	0.0	0.0	0.0	0%	0.0	0.0	0.0	0.0	0.0	0.0
7	7	Sergio Agüero	Forward	12	0.0	0.0	4.0	0	2.0	2.0	...	0.0	19.0	12.0	63%	2.0	0.0	0.0	0.0	0.0	0.0
13	13	Albian Ajeti	Forward	0	0.0	0.0	0.0	0	0.0	0.0	...	0.0	0.0	0.0	0%	0.0	0.0	0.0	0.0	0.0	0.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
845	845	Andriy Yarmolenko	Forward	15	0.0	0.0	5.0	0	3.0	1.0	...	0.0	7.0	2.0	29%	0.0	0.0	0.0	0.0	0.0	0.0
850	850	Wilfried Zaha	Forward	30	0.0	0.0	26.0	0	23.0	9.0	...	0.0	60.0	20.0	33%	5.0	0.0	0.0	0.0	0.0	0.0
853	853	Andi Zeqiri	Forward	9	0.0	0.0	3.0	0	3.0	0.0	...	0.0	7.0	2.0	29%	1.0	0.0	0.0	0.0	0.0	0.0
856	856	Richairo Zivkovic	Forward	0	0.0	0.0	0.0	0	0.0	0.0	...	0.0	0.0	0.0	0%	0.0	0.0	0.0	0.0	0.0	0.0
858	858	Kenneth Zohore	Forward	0	0.0	0.0	0.0	0	0.0	0.0	...	0.0	0.0	0.0	0%	0.0	0.0	0.0	0.0	0.0	0.0

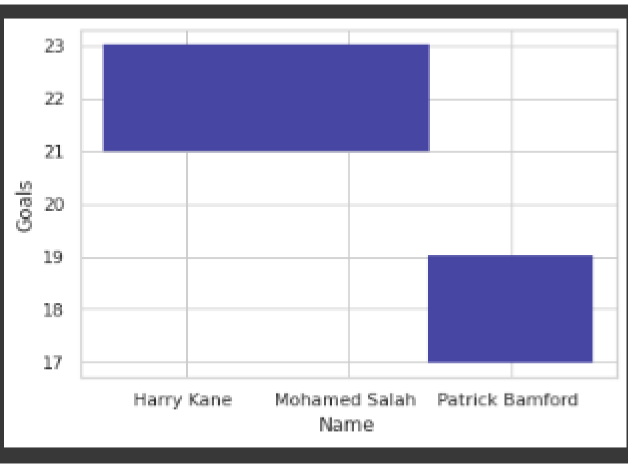
```
fwd_data = df_fwd[['Name','Goals','Goals per match','Shooting accuracy %','Big chances missed']].nlargest(3, ['Goals'])
fwd_data
```

	Name	Goals	Goals per match	Shooting accuracy %	Big chances missed
405	Harry Kane	23	0.66	39%	13.0
550	Mohamed Salah	22	0.59	41%	19.0
67	Patrick Bamford	17	0.45	45%	21.0

```
fwd_data.dtypes
```

```
Name                   object
Goals                   int64
Goals per match       float64
Shooting accuracy %    object
Big chances missed    float64
dtype: object
```

```
sns.histplot(x="Name", y="Goals", data=fwd_data, color="Blue")
plt.show()
```



The graph shows the top 3 best forwards during EPL 2020-2021. It shows that Harry Kane is the best forward because he got 23 goals and brought the golden boot for his team.

5.0 SUMMARY

5.1 Overall Summary

```
##Overall Summary
overall_data = data_new.drop("Unnamed: 0", axis=1)
overall_data.describe()
```

	Appearances	Clean sheets	Goals Conceded	Tackles	Blocked shots	Interceptions	Clearances	Headed Clearance	Recoveries	Duels won	...	Penalties scored	Freekicks scored	Sh
count	862.000000	862.000000	862.000000	862.000000	862.000000	862.000000	862.000000	862.000000	862.000000	862.000000	...	862.000000	862.000000	862.000
mean	11.190255	1.100928	5.339907	12.761021	2.697216	8.502320	14.074246	7.661253	35.207657	31.718097	...	0.112529	0.015081	8.270
std	13.293908	2.939621	12.752236	20.428248	5.246090	14.410584	28.693649	16.225011	62.238235	56.235882	...	0.660908	0.169730	18.755
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	...	0.000000	0.000000	0.000
25%	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	...	0.000000	0.000000	0.000
50%	2.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	...	0.000000	0.000000	0.000
75%	24.000000	0.000000	0.000000	20.000000	3.000000	12.000000	14.000000	7.000000	49.750000	42.750000	...	0.000000	0.000000	5.000
max	38.000000	19.000000	74.000000	108.000000	37.000000	84.000000	193.000000	102.000000	296.000000	364.000000	...	9.000000	4.000000	137.000

Based on the summary above, only the attributes for Appearances has the value for mean and median. Both the mean of 11.1903 and median of 2.0000 indicate where the centre of the data is located, and the player's participation in a match. Thus, the player's participation in a match is about 11 times.

5.2 Summary by Attributes

Summary of attribute Appearances

```
##Summary for Appearances
data_new['Appearances'].describe()
```

```
count    862.000000
mean      11.190255
std       13.293908
min        0.000000
25%        0.000000
50%        2.000000
75%       24.000000
max       38.000000
Name: Appearances, dtype: float64
```

```
data_new['Appearances'].skew()
```

```
0.7009163630869755
```

For column Appearances, the skewness is negatively skewed and the average is 11.1903

Summary of attribute Clean Sheets

```
##Summary for Clean sheets
data_new['Clean sheets'].describe()
```

```
count    862.000000
mean       1.100928
std        2.939621
min        0.000000
25%        0.000000
50%        0.000000
75%        0.000000
max       19.000000
Name: Clean sheets, dtype: float64
```

```
data_new['Clean sheets'].skew()
```

```
2.934708865207929
```

For column Clean sheets, the skewness is negatively skewed and the average is 1.1009

Summary of attribute Tackles

```
##Summary for Tackles
data_new['Tackles'].describe()
```

```
count    862.000000
mean      12.761021
std       20.428248
min        0.000000
25%        0.000000
50%        0.000000
75%       20.000000
max      108.000000
Name: Tackles, dtype: float64
```

```
data_new['Tackles'].skew()
```

```
1.8922032557199115
```

For column Tackles, the skewness is negatively skewed and the average is 12.7610

Summary of attribute Blocked shots

```
##Summary for Blocked shots
data_new['Blocked shots'].describe()
```

```
count    862.000000
mean       2.697216
std        5.246090
min        0.000000
25%        0.000000
50%        0.000000
75%        3.000000
max       37.000000
Name: Blocked shots, dtype: float64
```

```
data_new['Blocked shots'].skew()
```

```
2.762438058267655
```

For column Blocked shots, the skewness is negatively skewed and the average is 2.6972

Summary for attribute Interceptions

```
##Summary for Interceptions
data_new['Interceptions'].describe()
```

```
count    862.000000
mean       8.502320
std       14.410584
min        0.000000
25%        0.000000
50%        0.000000
75%       12.000000
max       84.000000
Name: Interceptions, dtype: float64
```

```
data_new['Interceptions'].skew()
```

```
1.9902623814443803
```

For column Interceptions, the skewness is negatively skewed and the average is 8.5023

Summary for attribute Clearances

```
##Summary for Clearances
data_new['Clearances'].describe()
```

```
count    862.000000
mean      14.074246
std       28.693649
min        0.000000
25%        0.000000
50%        0.000000
75%       14.000000
max      193.000000
Name: Clearances, dtype: float64
```

```
data_new['Clearances'].skew()
```

```
2.9194077037559683
```

For column Clearances, the skewness is negatively skewed and the average is 14.0742

Summary for attribute Yellow cards

```
##Summary for Yellow cards
data_new['Yellow cards'].describe()
```

```
count    862.000000
mean       1.186775
std        1.985116
min        0.000000
25%        0.000000
50%        0.000000
75%        2.000000
max       12.000000
Name: Yellow cards, dtype: float64
```

```
data_new['Yellow cards'].skew()
```

```
2.022438609564947
```

For column Yellow cards, the skewness is negatively skewed and the average is 1.1868

Summary for attribute Red cards

```
##Summary for Red cards
data_new['Red cards'].describe()
```

```
count    862.000000
mean       0.055684
std        0.234452
min        0.000000
25%        0.000000
50%        0.000000
75%        0.000000
max        2.000000
Name: Red cards, dtype: float64
```

```
data_new['Red cards'].skew()
```

```
4.150335582394723
```

For column Red cards, the skewness is negatively skewed and the average is 0.0557

Summary for attribute Fouls

```
##Summary for Fouls
data_new['Fouls'].describe()
```

```
count    862.000000
mean       8.953596
std       13.121435
min        0.000000
25%        0.000000
50%        0.000000
75%       15.000000
max       69.000000
Name: Fouls, dtype: float64
```

```
data_new['Fouls'].skew()
```

```
1.513366509115005
```

For column Fouls, the skewness is negatively skewed and the average is 8.9536

Summary for attribute Goals

```
##Summary for Goals
data_new['Goals'].describe()
```

```
count    862.000000
mean       1.073086
std        2.720934
min        0.000000
25%        0.000000
50%        0.000000
75%        1.000000
max       23.000000
Name: Goals, dtype: float64
```

```
data_new['Goals'].skew()
```

```
3.88851005433
```

For column Goals, the skewness is negatively skewed and the average is 1.0731

Summary for attribute Goals per match

```
##Summary for Goals per match
data_new['Goals per match'].describe()
```

```
count    862.000000
mean       0.034803
std        0.092739
min        0.000000
25%        0.000000
50%        0.000000
75%        0.000000
max        0.660000
Name: Goals per match, dtype: float64
```

```
data_new['Goals per match'].skew()
```

```
3.41435501843l0434
```

For column Goals per match, the skewness is negatively skewed and the average is 0.0348

Summary for attribute Big chances missed

```
##Summary for Big chances missed
data_new['Big chances missed'].describe()
```

```
count    862.000000
mean       0.807425
std        2.531574
min        0.000000
25%        0.000000
50%        0.000000
75%        0.000000
max       21.000000
Name: Big chances missed, dtype: float64
```

```
data_new['Big chances missed'].skew()
```

```
4.678235296162453
```

For column Big chances missed, the skewness is negatively skewed and the average is 0.8074

Summary for attribute Saves

```
##Summary for Saves
data_new['Saves'].describe()
```

```
count    862.000000
mean       2.462877
std       15.717464
min        0.000000
25%        0.000000
50%        0.000000
75%        0.000000
max      166.000000
Name: Saves, dtype: float64
```

```
data_new['Saves'].skew()
```

```
7.197968396137915
```

For column Saves, the skewness is negatively skewed and the average is 2.4629

Summary for attribute Penalties Saved

```
##Summary for Penalties Saved
data_new['Penalties Saved'].describe()
```

```
count    862.000000
mean       0.015081
std        0.121947
min        0.000000
25%        0.000000
50%        0.000000
75%        0.000000
max        1.000000
Name: Penalties Saved, dtype: float64
```

```
data_new['Penalties Saved'].skew()
```

```
7.971453435664094
```

For column Penalties Saves, the skewness is negatively skewed and the average is 0.0151

Summary for attribute Sweeper clearances

```
##Summary for Sweeper clearances
data_new['Sweeper clearances'].describe()
```

```
count    862.000000
mean       0.256381
std        1.851747
min        0.000000
25%        0.000000
50%        0.000000
75%        0.000000
max       28.000000
Name: Sweeper clearances, dtype: float64
```

```
data_new['Sweeper clearances'].skew()
```

```
9.798084264764446
```

For column Sweeper clearances, the skewness is negatively skewed and the average is 0.2564

6.0 REFERENCES

Ojeabulu, G. (2021, August 16). *Exploratory data analysis expounded with FIFA 2021(part 1)*. Medium. Retrieved May 26, 2022, from https://pub.towardsai.net/exploratory-data-analysis-expounded-with-fifa-2021-part-1-f20c465d483e

2020/21 Premier League Player Stats & Season Archives. (n.d.). Retrieved May 26, 2021, from https://www.premierleague.com/stats/top/players/