ITM (SLS) Baroda University
School of Computer Science Engineering & Technology
DATA ANALYSIS & VISUALIZATION – C2610D4
B. Tech Semester VI CSE
Batch 2020-24

MINI – PROJECT

PROJECT TITLE : EXPLORATORY DATA ANALYSIS (EDA) ON INDIAN PREMIER LEAGUE (IPL)

PROJECT CREATOR : SAHAJ PATEL [20C21051]

INTRODUCTION

The Indian Premier League (IPL) is a professional Twenty20 cricket league in India. The league was founded by the Board of Control for Cricket in India (BCCI) in 2007, with the first season taking place in 2008.

Exploratory Data Analysis (EDA) is an approach for summarizing, visualizing, and understanding the essential characteristics of a data set. In this project, we will be analyzing IPL dataset using EDA techniques.

In this project, we perform exploratory data analysis on IPL data using Python and various data visualization libraries such as Plotly. The aim of this project is to gain insights into different aspects of IPL matches, including the most popular venues, top players, teams with the most toss wins, and more.

DATA SETS

The IPL dataset is read into the code using the read_csv() function from pandas. The Matches and Records variables are assigned to the respective datasets read from CSV files. The index_col parameter is set to 'id' for the Matches dataset.

We use two data sets for this project:

- IPL_Dataset.csv: This data set contains information about IPL matches, including the teams, venue, winner, and more.

- Stats.csv: This data set contains information about IPL players, including the number of runs, centuries, strike rate, and more.

LIBRARIES USED

The following libraries have been imported and used in this project:

Numpy: For linear algebra
Pandas: For data processing and CSV file I/O
Plotly.express: For interactive data visualization

DATA PRE-PROCESSING

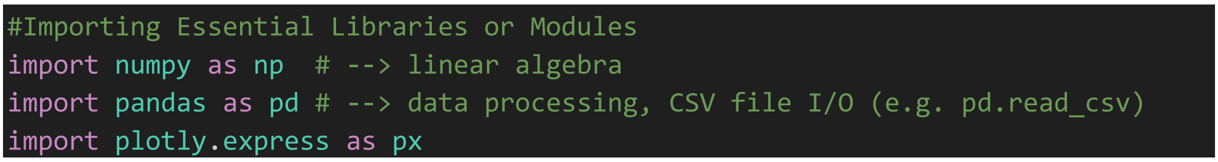
The Matches dataset's method column is found to be not useful and is, therefore, dropped using the drop() function. The axis parameter is set to 1, and the inplace parameter is set to True.

VISUALIZATIONS

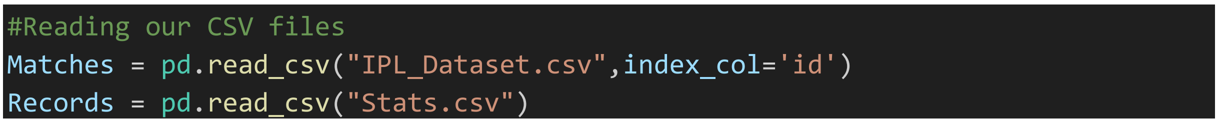
In this project, we have used various visualization techniques to understand the IPL data set better.

Pie Graph - Most Wins in IPL
Bar Plot - Most Wins in Eliminator
Scatter Plot - Most Runs Scored by Individual in IPL
Bar Plot - Most No of Centuries in IPL
Sunburst Chart - Player Stats
Scatter Plot - Most Sixes
Bar Chart - Top Famous Venues
Scatter Plot - Most Player of the Match Awards
Pie Chart - Most no of Toss Wins!
Bar Chart - Elected To Bat or Field after Winning Toss
Bar Chart - Top Umpires
Bar Chart & Scatter Plot – Rivalry between Strongest Teams in IPL

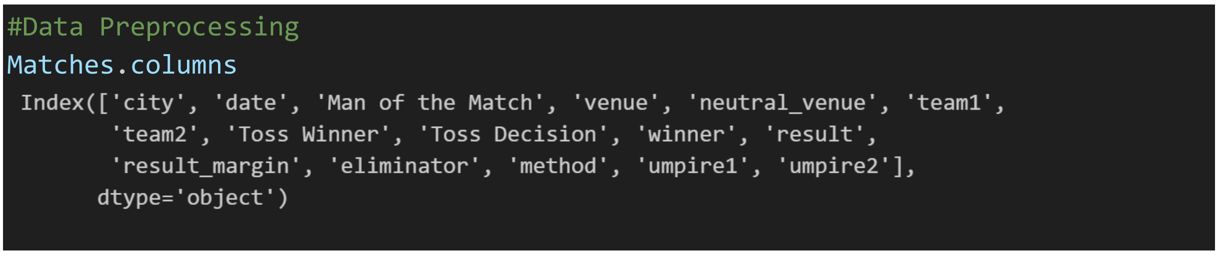
CODE & OUTPUT



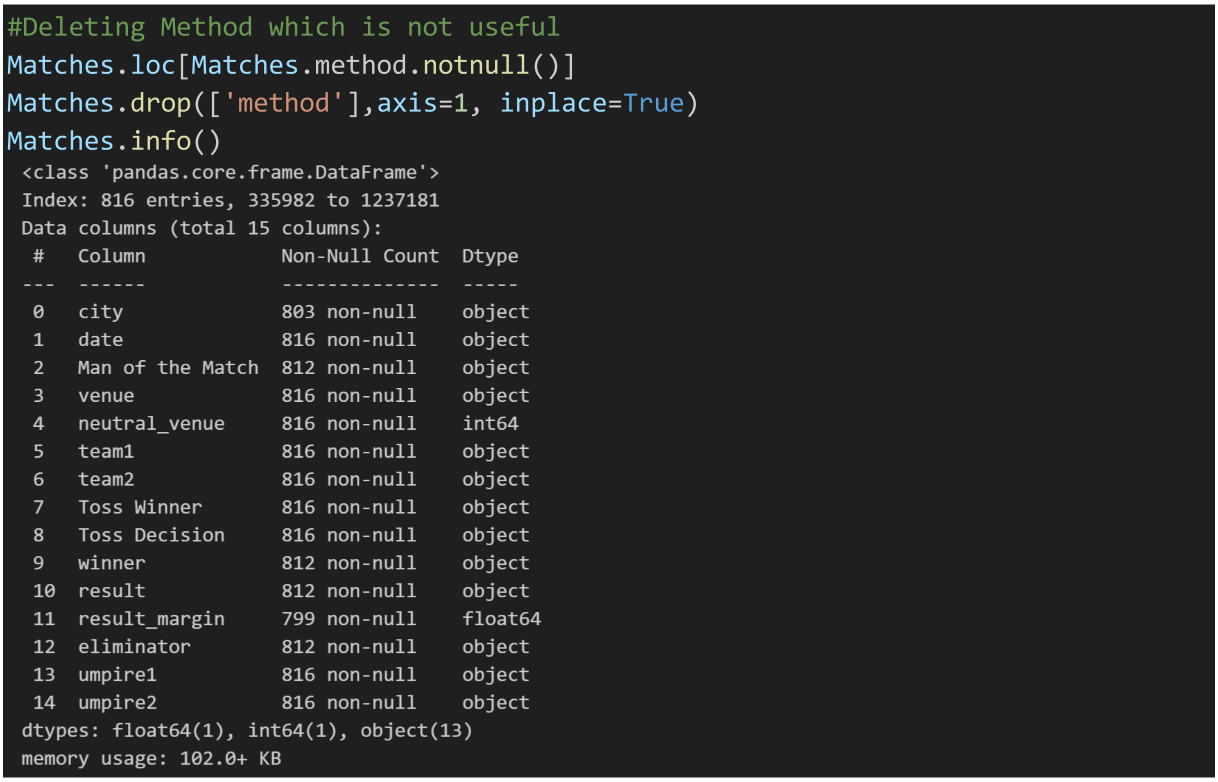
```
#Importing Essential Libraries or Modules
import numpy as np  # --> linear algebra
import pandas as pd # --> data processing, CSV file I/O (e.g. pd.read_csv)
import plotly.express as px
```



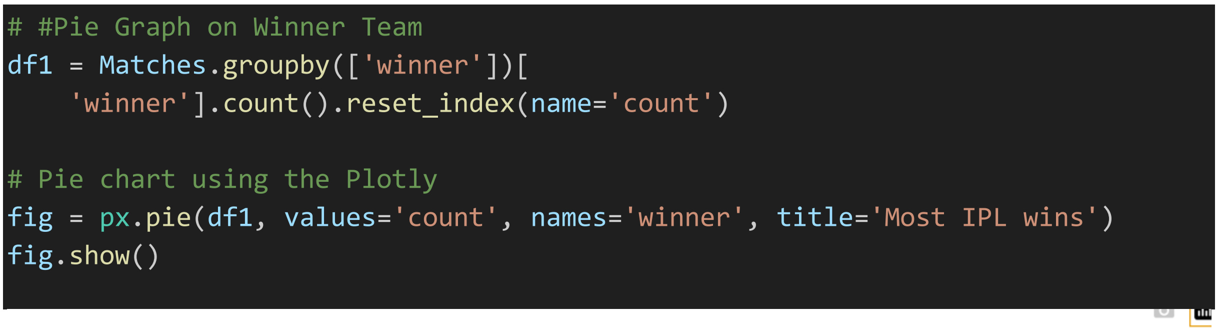
```
#Reading our CSV files
Matches = pd.read_csv("IPL_Dataset.csv",index_col='id')
Records = pd.read_csv("Stats.csv")
```



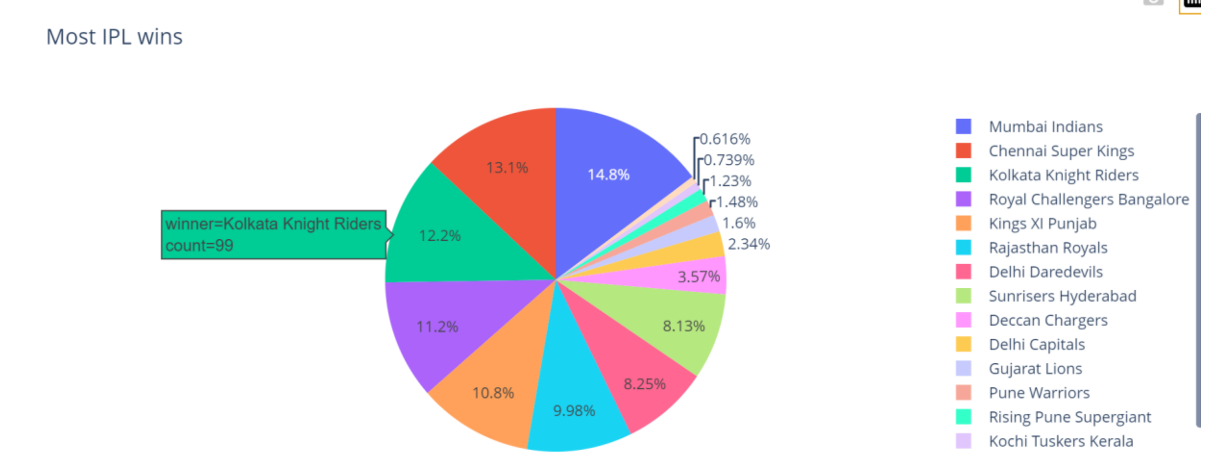
```
#Data Preprocessing
Matches.columns
 Index(['city', 'date', 'Man of the Match', 'venue', 'neutral_venue', 'team1',
        'team2', 'Toss Winner', 'Toss Decision', 'winner', 'result',
        'result_margin', 'eliminator', 'method', 'umpire1', 'umpire2'],
       dtype='object')
```

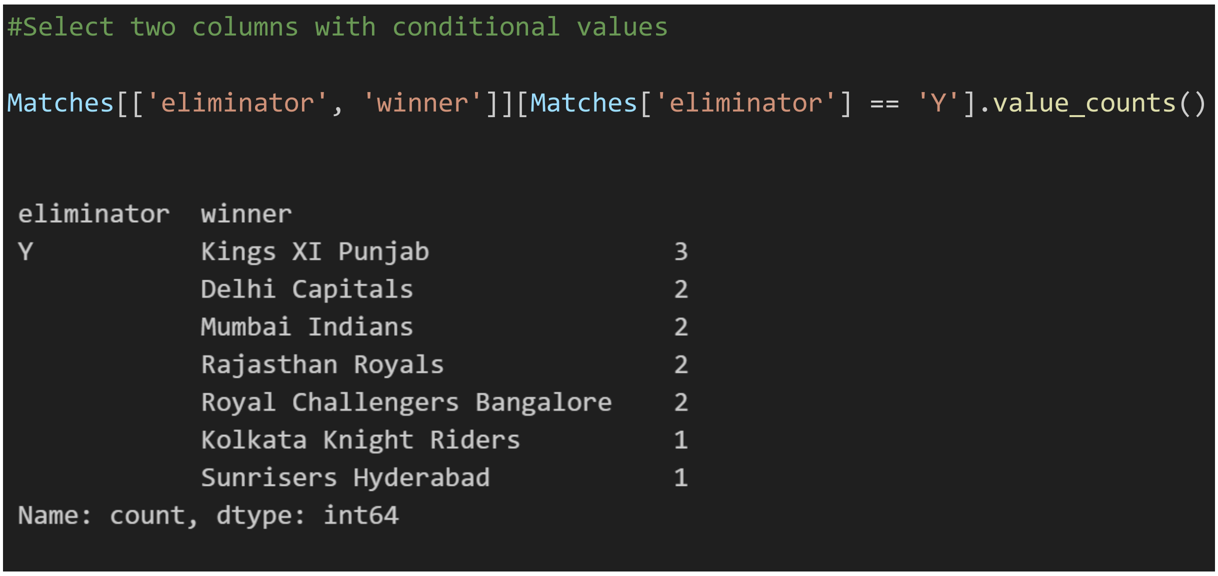


```
#Deleting Method which is not useful
Matches.loc[Matches.method.notnull()]
Matches.drop(['method'],axis=1, inplace=True)
Matches.info()
 <class 'pandas.core.frame.DataFrame'>
 Index: 816 entries, 335982 to 1237181
 Data columns (total 15 columns):
  #   Column            Non-Null Count  Dtype  
 ---  ------            --------------  -----  
  0   city              803 non-null    object 
  1   date              816 non-null    object 
  2   Man of the Match  812 non-null    object 
  3   venue             816 non-null    object 
  4   neutral_venue     816 non-null    int64  
  5   team1             816 non-null    object 
  6   team2             816 non-null    object 
  7   Toss Winner       816 non-null    object 
  8   Toss Decision     816 non-null    object 
  9   winner            812 non-null    object 
  10  result            812 non-null    object 
  11  result_margin     799 non-null    float64
  12  eliminator        812 non-null    object 
  13  umpire1           816 non-null    object 
  14  umpire2           816 non-null    object 
 dtypes: float64(1), int64(1), object(13)
 memory usage: 102.0+ KB
```



```
# #Pie Graph on Winner Team
df1 = Matches.groupby(['winner'])[
    'winner'].count().reset_index(name='count')

# Pie chart using the Plotly
fig = px.pie(df1, values='count', names='winner', title='Most IPL wins')
fig.show()
```

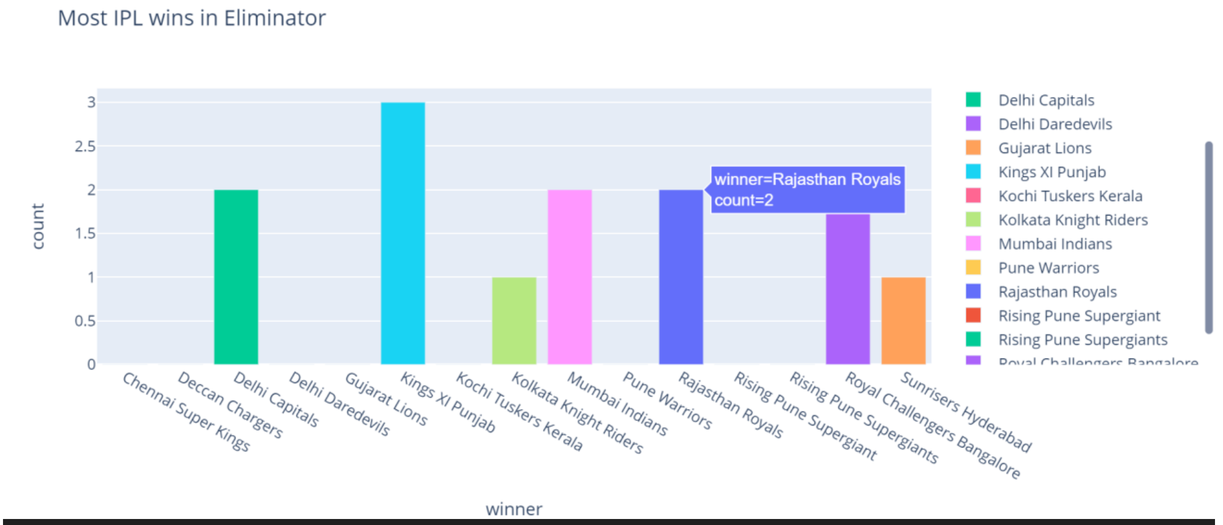




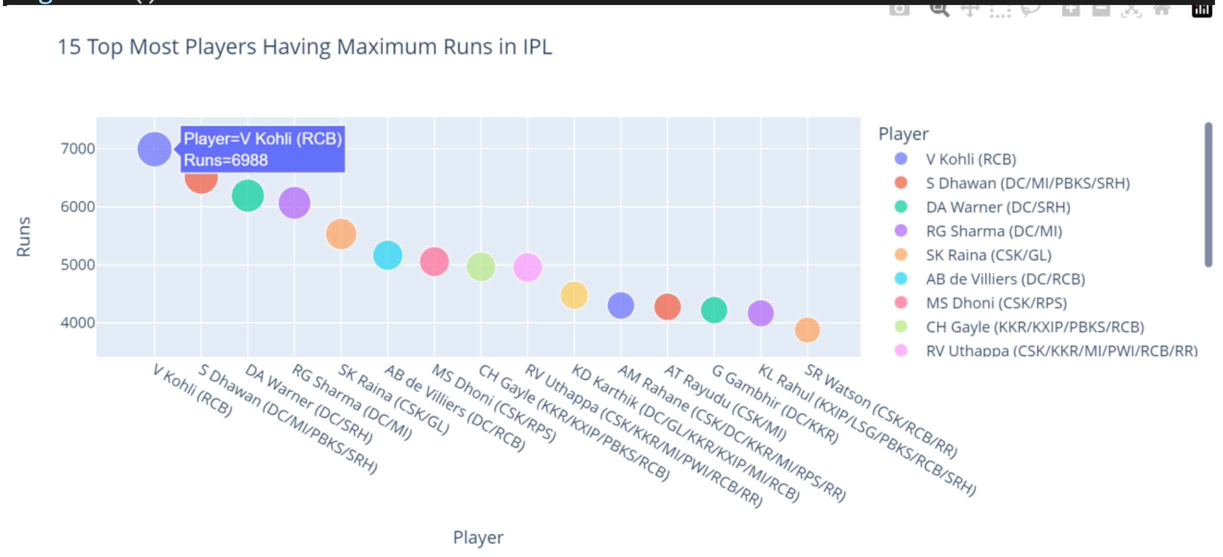
```
#Select two columns with conditional values

Matches[['eliminator', 'winner']][Matches['eliminator'] == 'Y'].value_counts()
```

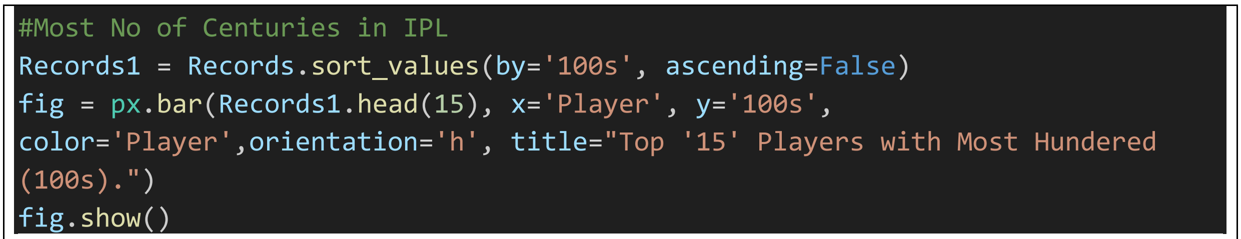
```
eliminator  winner
Y           Kings XI Punjab                3
            Delhi Capitals                 2
            Mumbai Indians                 2
            Rajasthan Royals               2
            Royal Challengers Bangalore    2
            Kolkata Knight Riders          1
            Sunrisers Hyderabad            1
Name: count, dtype: int64
```

```
#Bar Plot - Most Wins in Eliminator
df2 = Matches.groupby('winner')['eliminator'].apply(lambda x: (x ==
'Y').sum()).reset_index(name='count')
fig = px.bar(df2, x='winner', y='count', color="winner", title='Most IPL wins
in Eliminator')
fig.show()
```

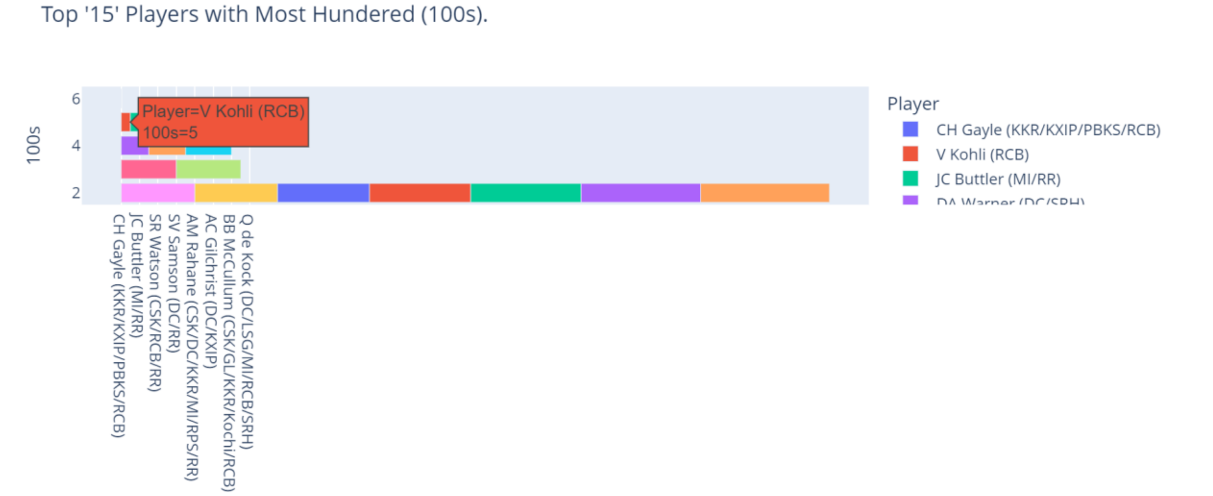


```
#Most Runs in IPL
fig=px.scatter(Records.head(15), x='Player', y='Runs', color='Player',
size='Runs', title='15 Top Most Players Having Maximum Runs in IPL')
fig.show()
```

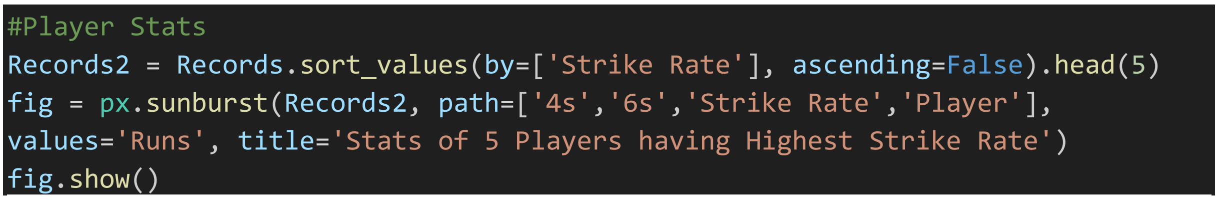




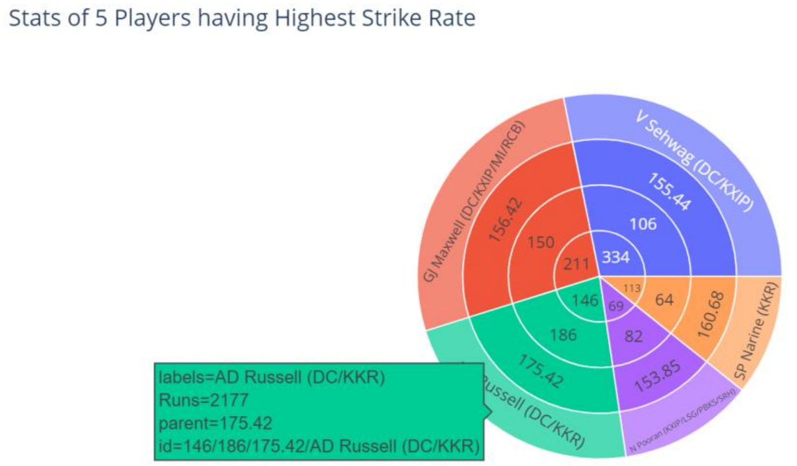
```
#Most No of Centuries in IPL
Records1 = Records.sort_values(by='100s', ascending=False)
fig = px.bar(Records1.head(15), x='Player', y='100s',
color='Player',orientation='h', title="Top '15' Players with Most Hundered
(100s).")
fig.show()
```



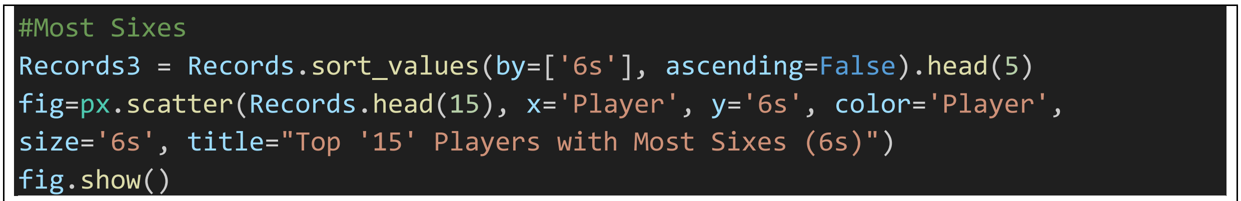
Top '15' Players with Most Hundered (100s).



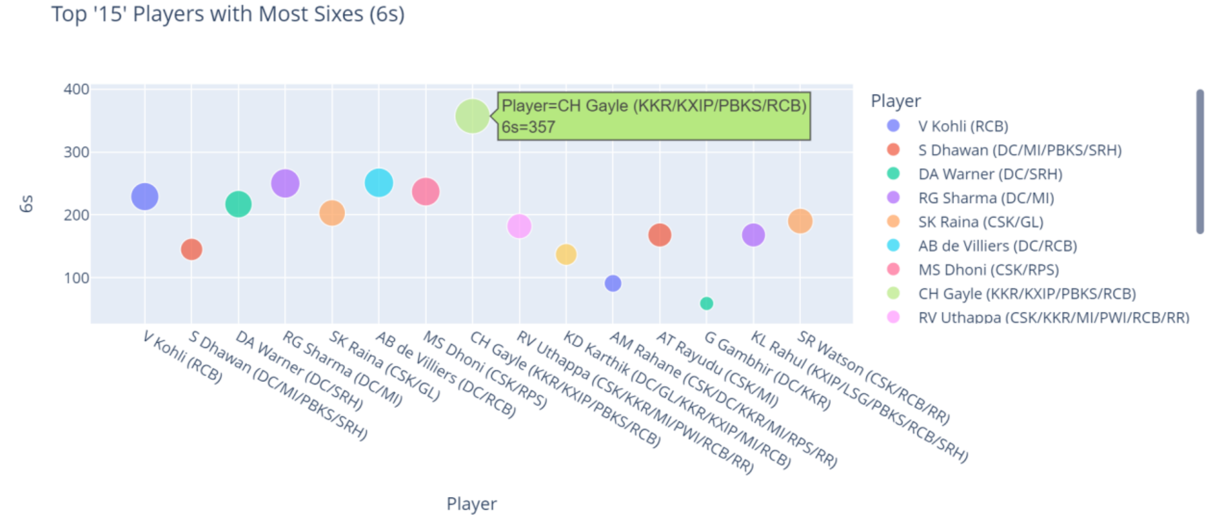
```
#Player Stats
Records2 = Records.sort_values(by=['Strike Rate'], ascending=False).head(5)
fig = px.sunburst(Records2, path=['4s','6s','Strike Rate','Player'],
values='Runs', title='Stats of 5 Players having Highest Strike Rate')
fig.show()
```

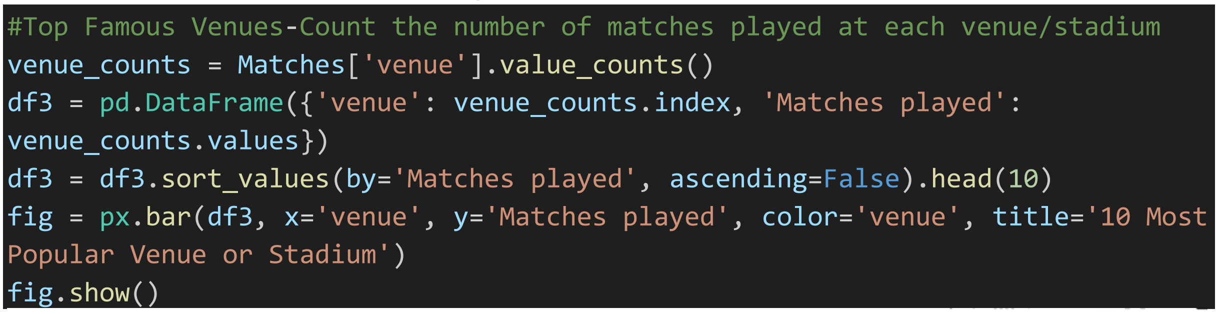


Stats of 5 Players having Highest Strike Rate

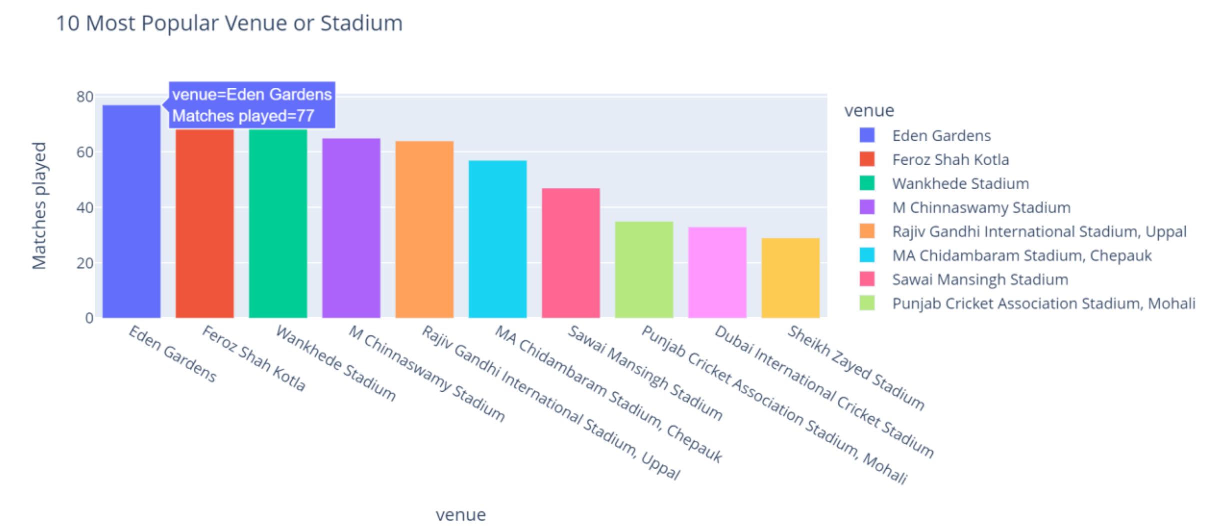


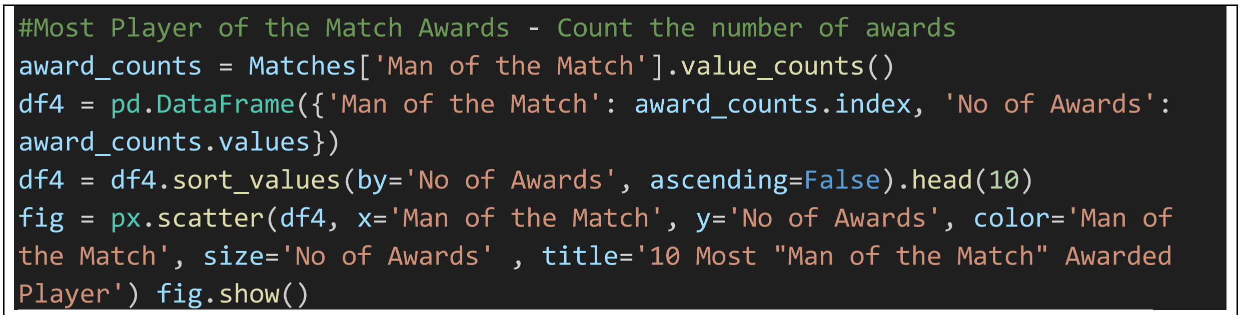
```
#Most Sixes
Records3 = Records.sort_values(by=['6s'], ascending=False).head(5)
fig=px.scatter(Records.head(15), x='Player', y='6s', color='Player',
size='6s', title="Top '15' Players with Most Sixes (6s)")
fig.show()
```



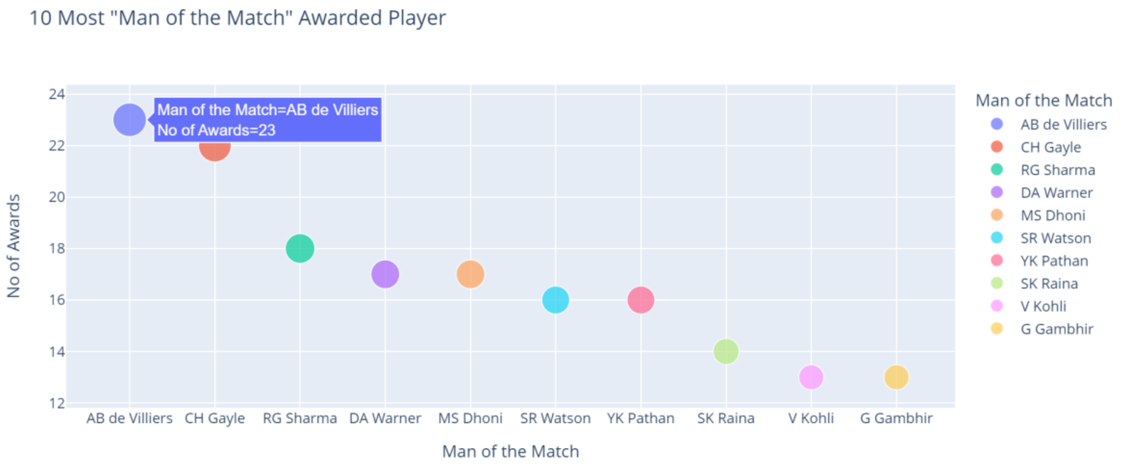


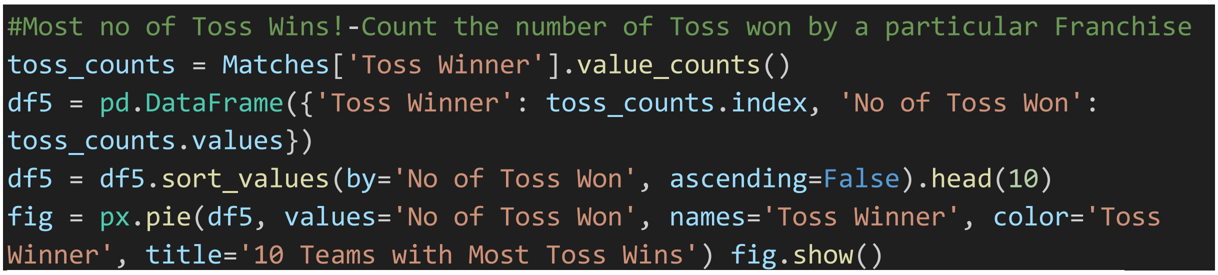
```
#Top Famous Venues-Count the number of matches played at each venue/stadium
venue_counts = Matches['venue'].value_counts()
df3 = pd.DataFrame({'venue': venue_counts.index, 'Matches played':
venue_counts.values})
df3 = df3.sort_values(by='Matches played', ascending=False).head(10)
fig = px.bar(df3, x='venue', y='Matches played', color='venue', title='10 Most
Popular Venue or Stadium')
fig.show()
```



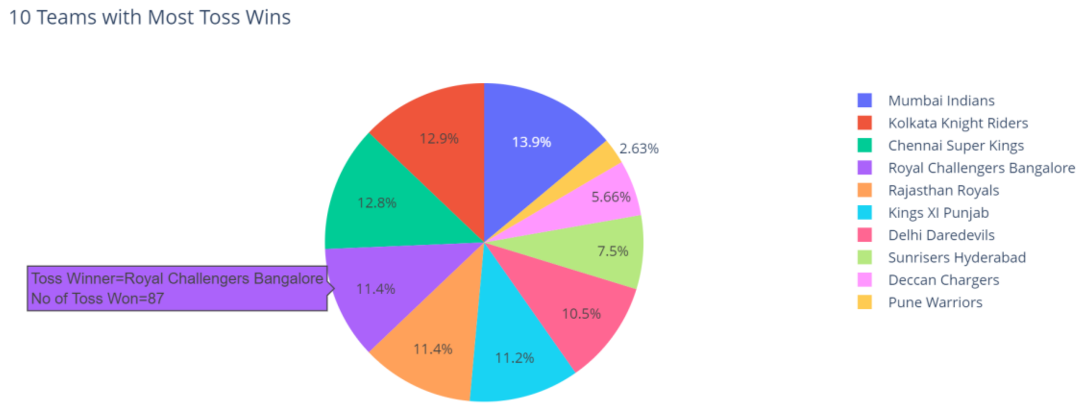


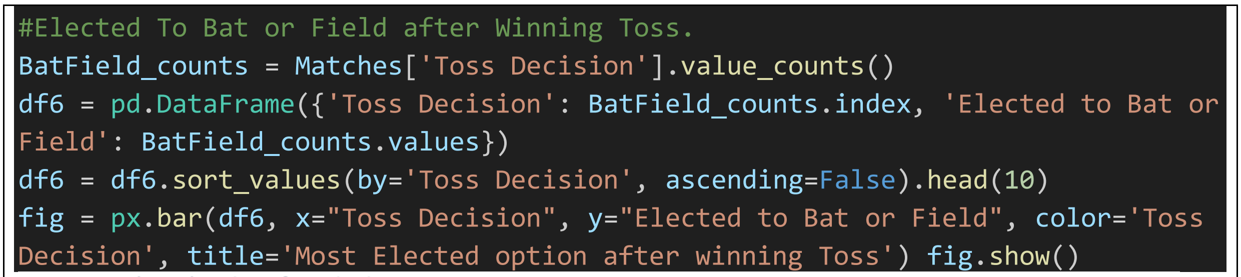
```
#Most Player of the Match Awards - Count the number of awards
award_counts = Matches['Man of the Match'].value_counts()
df4 = pd.DataFrame({'Man of the Match': award_counts.index, 'No of Awards':
award_counts.values})
df4 = df4.sort_values(by='No of Awards', ascending=False).head(10)
fig = px.scatter(df4, x='Man of the Match', y='No of Awards', color='Man of
the Match', size='No of Awards' , title='10 Most "Man of the Match" Awarded
Player') fig.show()
```



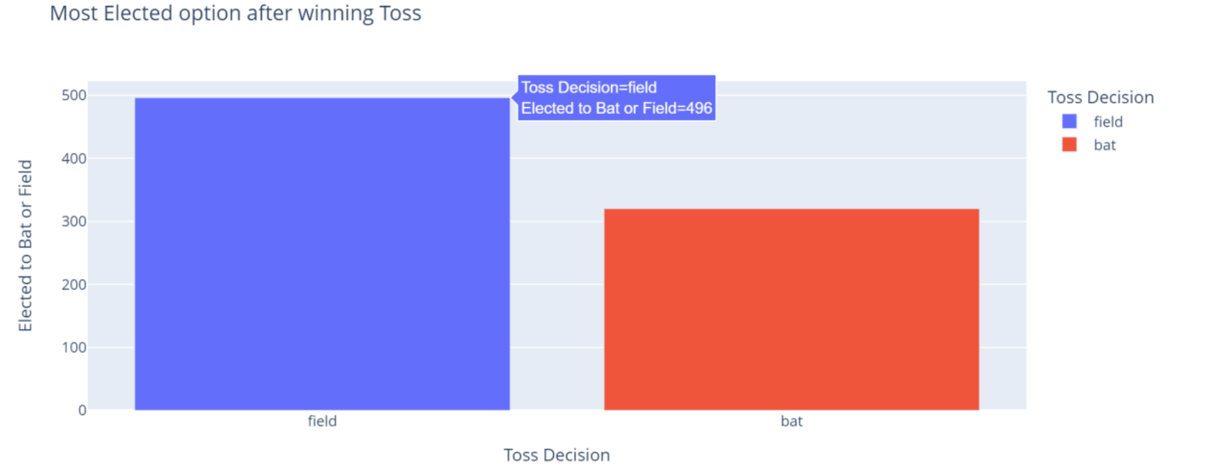


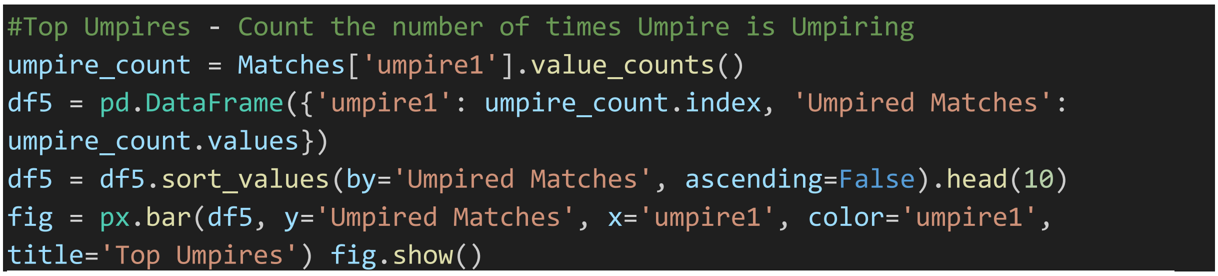
```
#Most no of Toss Wins!-Count the number of Toss won by a particular Franchise
toss_counts = Matches['Toss Winner'].value_counts()
df5 = pd.DataFrame({'Toss Winner': toss_counts.index, 'No of Toss Won':
toss_counts.values})
df5 = df5.sort_values(by='No of Toss Won', ascending=False).head(10)
fig = px.pie(df5, values='No of Toss Won', names='Toss Winner', color='Toss
Winner', title='10 Teams with Most Toss Wins') fig.show()
```



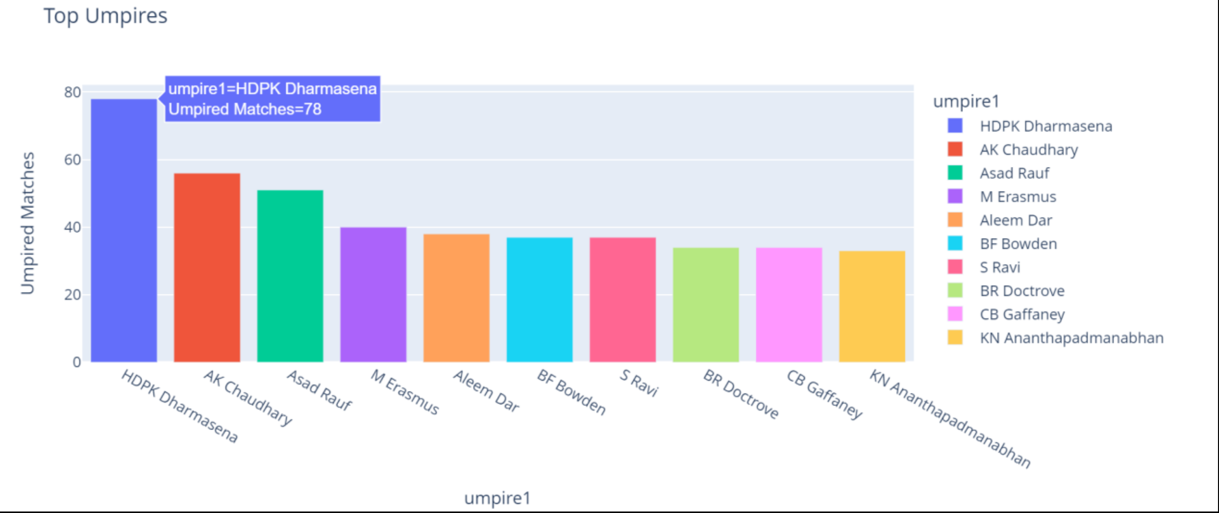


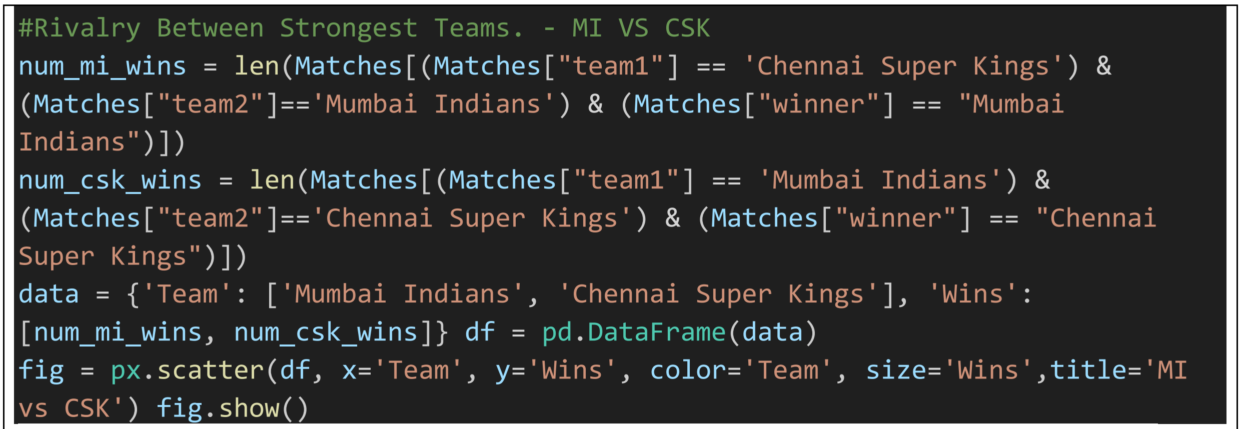
```
#Elected To Bat or Field after Winning Toss.
BatField_counts = Matches['Toss Decision'].value_counts()
df6 = pd.DataFrame({'Toss Decision': BatField_counts.index, 'Elected to Bat or Field': BatField_counts.values})
df6 = df6.sort_values(by='Toss Decision', ascending=False).head(10)
fig = px.bar(df6, x="Toss Decision", y="Elected to Bat or Field", color='Toss Decision', title='Most Elected option after winning Toss') fig.show()
```



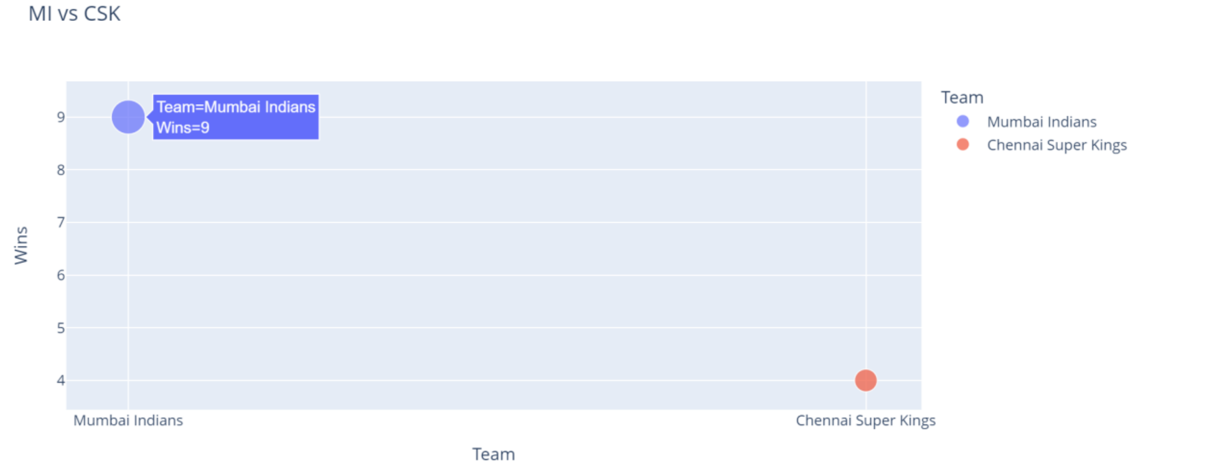


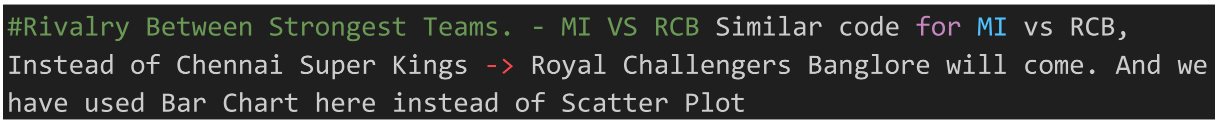
```
#Top Umpires - Count the number of times Umpire is Umpiring
umpire_count = Matches['umpire1'].value_counts()
df5 = pd.DataFrame({'umpire1': umpire_count.index, 'Umpired Matches': umpire_count.values})
df5 = df5.sort_values(by='Umpired Matches', ascending=False).head(10)
fig = px.bar(df5, y='Umpired Matches', x='umpire1', color='umpire1', title='Top Umpires') fig.show()
```



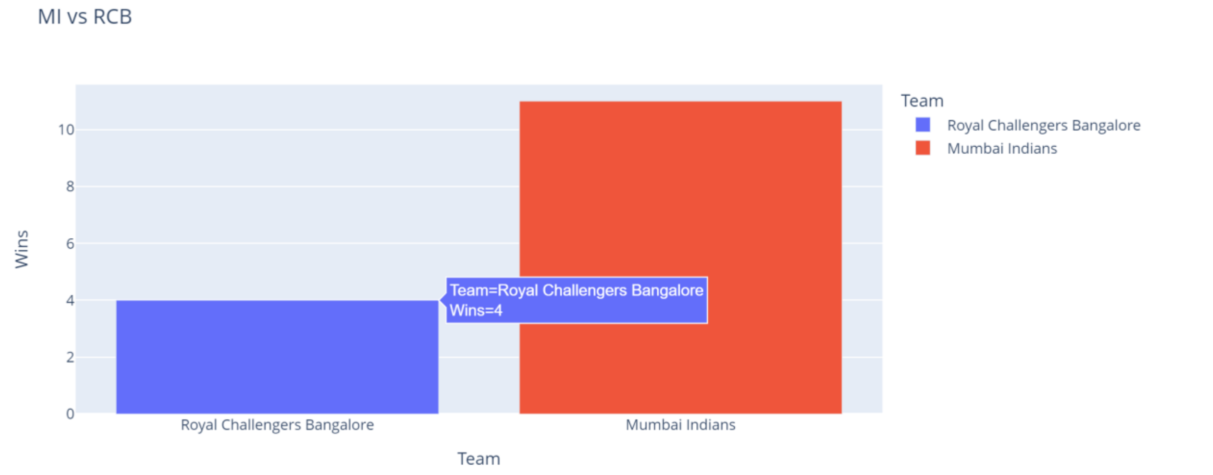


```
#Rivalry Between Strongest Teams. - MI VS CSK
num_mi_wins = len(Matches[(Matches["team1"] == 'Chennai Super Kings') &
(Matches["team2"]=='Mumbai Indians') & (Matches["winner"] == "Mumbai
Indians")])
num_csk_wins = len(Matches[(Matches["team1"] == 'Mumbai Indians') &
(Matches["team2"]=='Chennai Super Kings') & (Matches["winner"] == "Chennai
Super Kings")])
data = {'Team': ['Mumbai Indians', 'Chennai Super Kings'], 'Wins':
[num_mi_wins, num_csk_wins]} df = pd.DataFrame(data)
fig = px.scatter(df, x='Team', y='Wins', color='Team', size='Wins',title='MI
vs CSK') fig.show()
```





```
#Rivalry Between Strongest Teams. - MI VS RCB Similar code for MI vs RCB,
Instead of Chennai Super Kings -> Royal Challengers Banglore will come. And we
have used Bar Chart here instead of Scatter Plot
```



CONCLUSION

EDA is a critical process to understand the data set better. In this project, we have used various visualization and plotting techniques to understand the IPL data set better. We have gained insights into the data and answered different questions like the most number of toss wins, the most player of the match awards, and many more. The visualizations give insights into the data set and help make data-driven decisions also these visualizations will help stakeholders make better decisions in the future.

In conclusion, the IPL dataset analysis and visualization project provided insights into various aspects of the Indian Premier League.
From the data analysis and visualization, we can draw the following conclusions:

Mumbai Indians is the **most successful team** with over **120 wins**.
Kings XI Punjab has the most IPL wins in Eliminator **3 wins.**
Virat Kohli has the **highest runs** in IPL with over **6980 runs**.
Chris Gayle has the most **number of centuries (6)** and the **most number of sixes (357)** in IPL.
AD Russell has the highest **strike rate (175.42)** in IPL.
Eden Gardens is the most **popular venue**.
AB de Villiers has the most number of "**Man of the Match**" awards **(23)** in IPL.
Mumbai Indians has the **most toss wins** record **(106).**

Overall, this EDA project provides valuable insights and interesting trends into the IPL dataset. The data can be further analyzed and utilized for strategic planning by the teams, players, and management to improve their performance and increase their chances of winning the IPL championship.