DATA MINING, ANALYSIS AND VISUALIZATION OF INDIAN PREMIER LEAGUE (IPL)

Shubh Deo

2019IMG-057

ABSTRACT

Data driven decision-making leads to maximization of goals. Various organizations are leveraging the actionable data generated through the analytical process and field of sports is no different. Sports analytics has been playing a major role in shaping success for many teams in various sports. Indian Premier League, IPL provides the most successful form of cricket. Sports analytics and data visualization can play an important role in ensuring that the major objective of this platform holds good. Considering the various aspects of this game, seamless integration of technology into the process to enhance the quality of the game is the need of the hour.

In this project also we will analyze the data of various teams, ball by ball analysis of two teams playing a particular match and visualizing various other things. We will make use of python language and its various data visualizing and analysing libraries like pandas, numpy, matplotlib, etc and implement this project on Jupyter notebook.

The uses of this project are :

- Good for strategy building and achieving good results.
- Analysis of a specific player, team and exploring strengths and weaknesses.
- It can be used for score forecasting and win percentage.

.

KEYWORDS

- **Data visualization**
- **Sports analytics**
- **Player Performance**
- **Python tools**

INTRODUCTION

Sports analytics and Data Visualization has provided a greater platform for Player selectors, managers and also the players to increase on field performance. Decision makers and analysis, the next piece of the framework, is the process of applying statistical tools and algorithms to data to gain insight into what is likely to happen in the future. Each movement of the ball, the player strike rate, run rate, everything is captured using special camera systems and other recording mechanisms. This data is run through various statistical algorithms, tools and visualization techniques to provide deeper insight and pave the way for recommendations to the player or team. With the ease of obtaining and storing data, advanced analytics and machine learning techniques are applied to engineer a predictive model for cricket.

The T20 format gave birth to Indian Premier League (IPL) a professional league contested during April and May of every year. It was initiated by the BCCI (Board of Control for Cricket in India) in 2008.

This shorter version of cricket is one of the most successful one in terms of fan engagement and business. Everyone enjoys this shorter version of cricket. The main objective of this league is to provide a platform for young and talented players. IPL works on the franchise system of hiring players. There are eight teams in IPL. Each team is a group of eleven players consisting of batsmen, bowler, and all-rounders. This tournament is being played in different cities, because of this, there is a huge fan following with a lot of media interest and business involvement.

Analytics can help in all these tough situations. Analytics bridges the gap for team selectors, coaches, and managers. Analytics gives us a clearer idea about player consistency, fast scoring and finishing ability. To manage the risk in a better way and to get the probable winners, analytics play a crucial role in the field and out of the field. Data Visualization is one of the major outcomes in sports analytics. The visual form of data is more easily understandable over numbers and text.

BACKGROUND OF THE PROPOSED WORK

Analytics can help in crucial situations. It bridges the gap for team selectors, coaches, and managers. Analytics gives them a clearer idea about player consistency, fast scoring and finishing ability. To manage the risk in a better way and to get the probable winners, analytics play a crucial role in the field and out of the field. Most of the studies related to IPL analysis are focused on a particular season or on a specific player. But in this report we will focus on the overall analysis of IPL from 2008 to 2019.

For example, Cluster analysis has been applied on the datasets of players of IPL season 2010. The study reveals that players of England had performed well as a group and New Zealand players were the lowest performers. The factor analysis used with various statistical techniques which shows that batting capability dominates over bowling.

Analysis is also done on various other key factors like the type of pitches –

Flat pitches, pitches that favor fast bowling, spin bowling and swing bowling and whether they are beneficial for batsmen, non-striker batsman, and bowlers for holding a good partnership.

This report shows the overall analysis of IPL from 2008-2019 and gives useful information in a simple manner. It is not only beneficial for the team management of various IPL franchises but it is also helpful to the BCCI and other organisations from a business perspective.

OBJECTIVES

We will make use of python language and its various data visualizing and analysing libraries like pandas, numpy, matplotlib, etc and implement this project on Jupyter notebook.

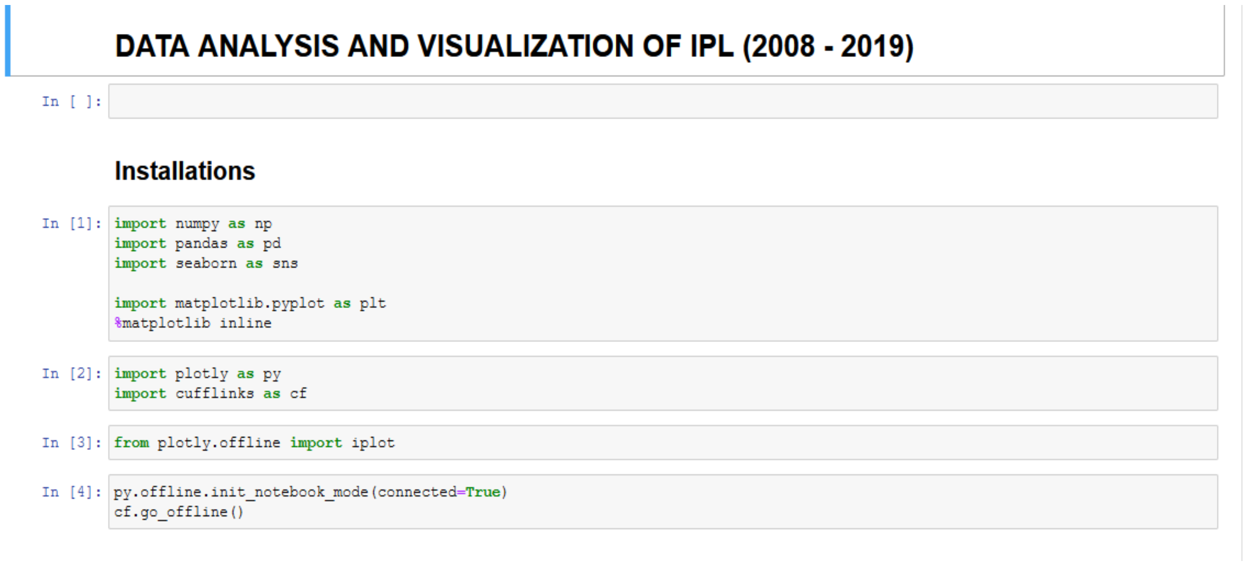
The report also covers the following :

- Data Reading
- Data Cleaning
- Wins and Loss Analysis
- Overall Team Performances
- Individual Player Analysis
- Ball by Ball Analysis
- Head to Head Analysis

Based on the observations of each analysis various outcomes will be discussed and how this result would help the management team to take calculated decisions.

METHODOLOGY

- ### INSTALLATIONS



DATA ANALYSIS AND VISUALIZATION OF IPL (2008 - 2019)

```
In [ ]:
```

Installations

```
In [1]: import numpy as np
        import pandas as pd
        import seaborn as sns

        import matplotlib.pyplot as plt
        %matplotlib inline

In [2]: import plotly as py
        import cufflinks as cf

In [3]: from plotly.offline import iplot

In [4]: py.offline.init_notebook_mode(connected=True)
        cf.go_offline()
```

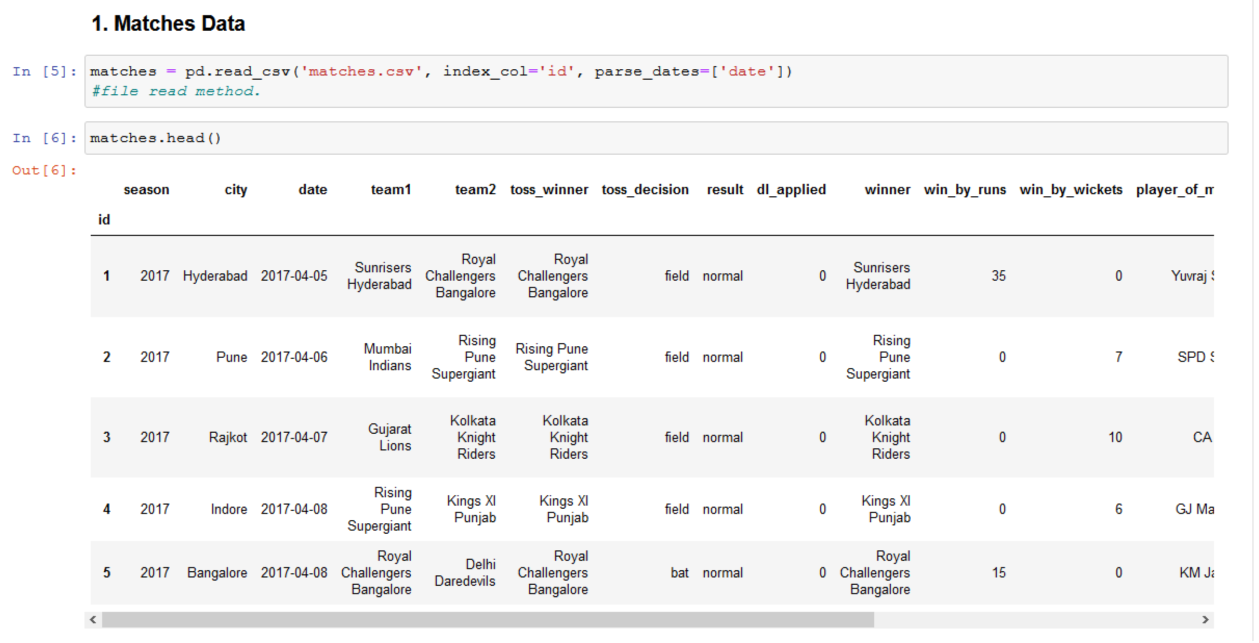
Libraries Installed:

1. Numpy
2. Pandas

3. Seaborn
4. Matplotlib
5. Plotly and Cufflinks

- ## DATA READING

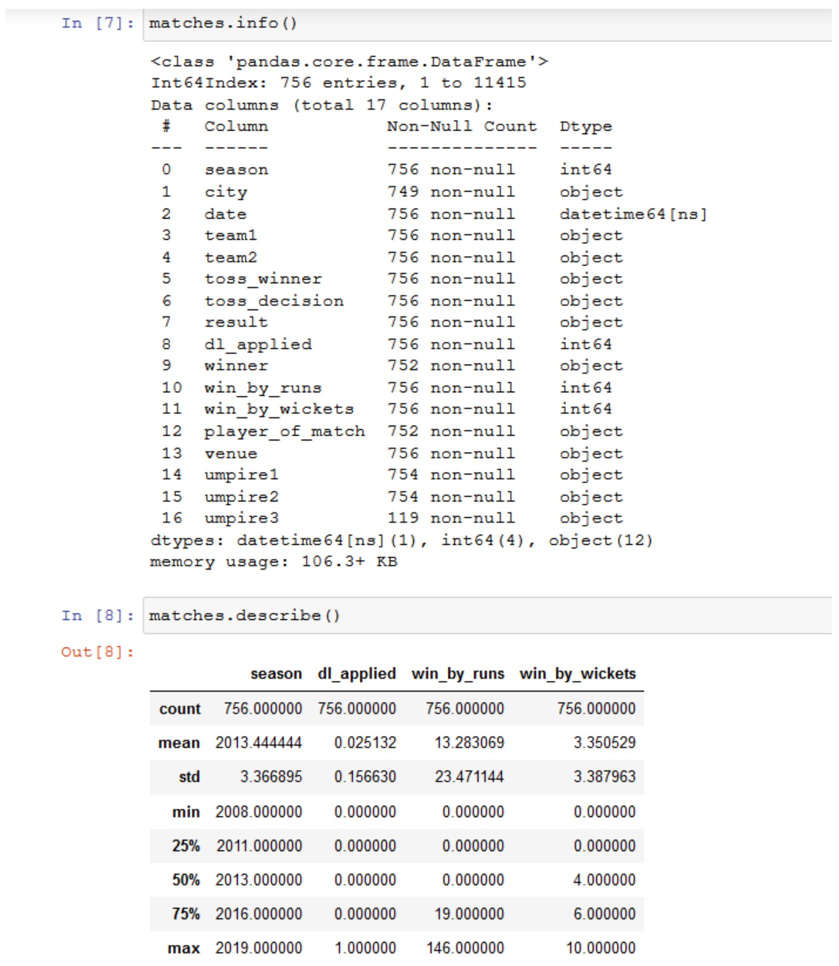
Matches Data : It provides the basic information about every single IPL match that has taken place from 2008 to 2019. It contains several attributes like season, city, venue, team1, team2, winner, etc. There are a total of **756 entries** in this dataset.



1. Matches Data

```
In [5]: matches = pd.read_csv('matches.csv', index_col='id', parse_dates=['date'])
        #file read method.

In [6]: matches.head()

Out[6]:
```

id	season	city	date	team1	team2	toss_winner	toss_decision	result	dl_applied	winner	win_by_runs	win_by_wickets	player_of_m
1	2017	Hyderabad	2017-04-05	Sunrisers Hyderabad	Royal Challengers Bangalore	Royal Challengers Bangalore	field	normal	0	Sunrisers Hyderabad	35	0	Yuvraj S
2	2017	Pune	2017-04-06	Mumbai Indians	Rising Pune Supergiant	Rising Pune Supergiant	field	normal	0	Rising Pune Supergiant	0	7	SPD S
3	2017	Rajkot	2017-04-07	Gujarat Lions	Kolkata Knight Riders	Kolkata Knight Riders	field	normal	0	Kolkata Knight Riders	0	10	CA
4	2017	Indore	2017-04-08	Rising Pune Supergiant	Kings XI Punjab	Kings XI Punjab	field	normal	0	Kings XI Punjab	0	6	GJ Ma
5	2017	Bangalore	2017-04-08	Royal Challengers Bangalore	Delhi Daredevils	Royal Challengers Bangalore	bat	normal	0	Royal Challengers Bangalore	15	0	KM Ja

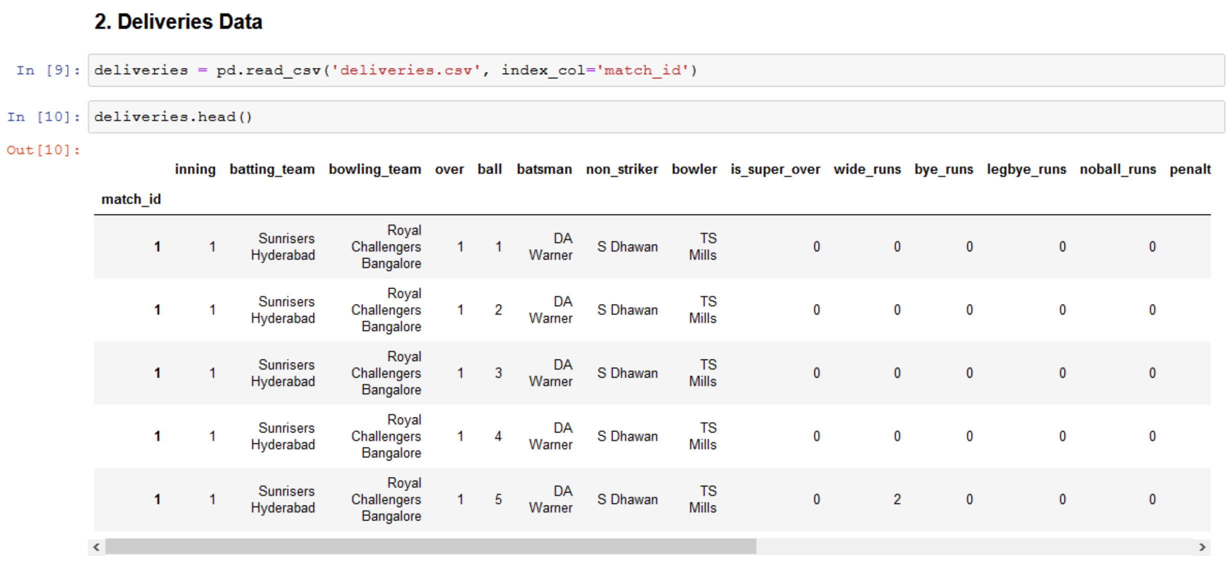


```
In [7]: matches.info()

<class 'pandas.core.frame.DataFrame'>
Int64Index: 756 entries, 1 to 11415
Data columns (total 17 columns):
 #   Column           Non-Null Count  Dtype
---  ------           --------------  -----
 0   season           756 non-null    int64
 1   city             749 non-null    object
 2   date             756 non-null    datetime64[ns]
 3   team1            756 non-null    object
 4   team2            756 non-null    object
 5   toss_winner      756 non-null    object
 6   toss_decision    756 non-null    object
 7   result           756 non-null    object
 8   dl_applied       756 non-null    int64
 9   winner           752 non-null    object
 10  win_by_runs      756 non-null    int64
 11  win_by_wickets   756 non-null    int64
 12  player_of_match  752 non-null    object
 13  venue            756 non-null    object
 14  umpire1          754 non-null    object
 15  umpire2          754 non-null    object
 16  umpire3          119 non-null    object
dtypes: datetime64[ns](1), int64(4), object(12)
memory usage: 106.3+ KB
```

```
In [8]: matches.describe()

Out[8]:
```

	season	dl_applied	win_by_runs	win_by_wickets
count	756.000000	756.000000	756.000000	756.000000
mean	2013.444444	0.025132	13.283069	3.350529
std	3.366895	0.156630	23.471144	3.387963
min	2008.000000	0.000000	0.000000	0.000000
25%	2011.000000	0.000000	0.000000	0.000000
50%	2013.000000	0.000000	0.000000	4.000000
75%	2016.000000	0.000000	19.000000	6.000000
max	2019.000000	1.000000	146.000000	10.000000

Deliveries Data : It provides detailed ball by ball information about every single IPL match that has taken place from 2008 to 2019. It contains several attributes like inning , bowling team, over , ball, total_runs, extra_runs, etc. There are a total of **179078 entries** in this dataset.



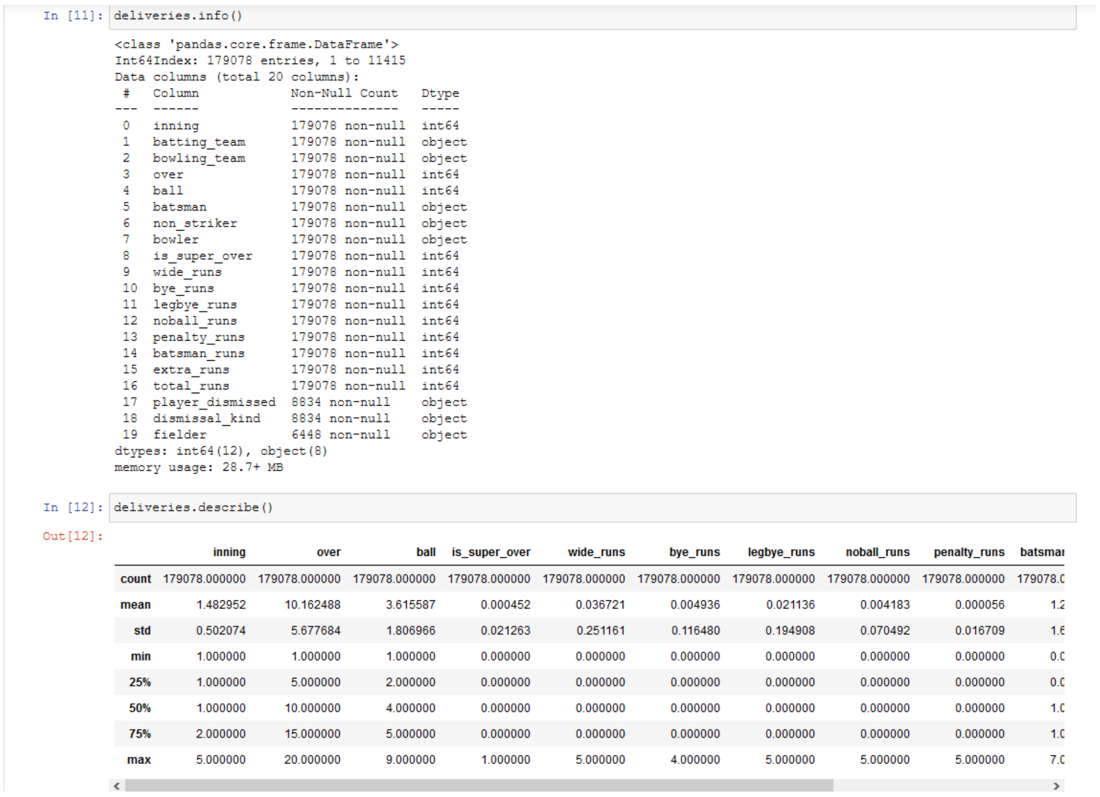
2. Deliveries Data

```
In [9]: deliveries = pd.read_csv('deliveries.csv', index_col='match_id')
```

```
In [10]: deliveries.head()
```

Out[10]:

	inning	batting_team	bowling_team	over	ball	batsman	non_striker	bowler	is_super_over	wide_runs	bye_runs	legbye_runs	noball_runs	penalt
match_id														
1	1	Sunrisers Hyderabad	Royal Challengers Bangalore	1	1	DA Warner	S Dhawan	TS Mills	0	0	0	0	0	
1	1	Sunrisers Hyderabad	Royal Challengers Bangalore	1	2	DA Warner	S Dhawan	TS Mills	0	0	0	0	0	
1	1	Sunrisers Hyderabad	Royal Challengers Bangalore	1	3	DA Warner	S Dhawan	TS Mills	0	0	0	0	0	
1	1	Sunrisers Hyderabad	Royal Challengers Bangalore	1	4	DA Warner	S Dhawan	TS Mills	0	0	0	0	0	
1	1	Sunrisers Hyderabad	Royal Challengers Bangalore	1	5	DA Warner	S Dhawan	TS Mills	0	2	0	0	0	



```
In [11]: deliveries.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 179078 entries, 1 to 11415
Data columns (total 20 columns):
 #   Column            Non-Null Count   Dtype
---  ------            --------------   -----
 0   inning            179078 non-null  int64
 1   batting_team      179078 non-null  object
 2   bowling_team      179078 non-null  object
 3   over              179078 non-null  int64
 4   ball              179078 non-null  int64
 5   batsman           179078 non-null  object
 6   non_striker       179078 non-null  object
 7   bowler            179078 non-null  object
 8   is_super_over     179078 non-null  int64
 9   wide_runs         179078 non-null  int64
 10  bye_runs          179078 non-null  int64
 11  legbye_runs       179078 non-null  int64
 12  noball_runs       179078 non-null  int64
 13  penalty_runs      179078 non-null  int64
 14  batsman_runs      179078 non-null  int64
 15  extra_runs        179078 non-null  int64
 16  total_runs        179078 non-null  int64
 17  player_dismissed  8834 non-null    object
 18  dismissal_kind    8834 non-null    object
 19  fielder           6448 non-null    object
dtypes: int64(12), object(8)
memory usage: 28.7+ MB
```

```
In [12]: deliveries.describe()
```

Out[12]:

	inning	over	ball	is_super_over	wide_runs	bye_runs	legbye_runs	noball_runs	penalty_runs	batsmai
count	179078.000000	179078.000000	179078.000000	179078.000000	179078.000000	179078.000000	179078.000000	179078.000000	179078.000000	179078.0
mean	1.482952	10.162488	3.615587	0.000452	0.036721	0.004936	0.021136	0.004183	0.000056	1.2
std	0.502074	5.677684	1.806966	0.021263	0.251161	0.116480	0.194908	0.070492	0.016709	1.6
min	1.000000	1.000000	1.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.0
25%	1.000000	5.000000	2.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.0
50%	1.000000	10.000000	4.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	1.0
75%	2.000000	15.000000	5.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	1.0
max	5.000000	20.000000	9.000000	1.000000	5.000000	4.000000	5.000000	5.000000	5.000000	7.0

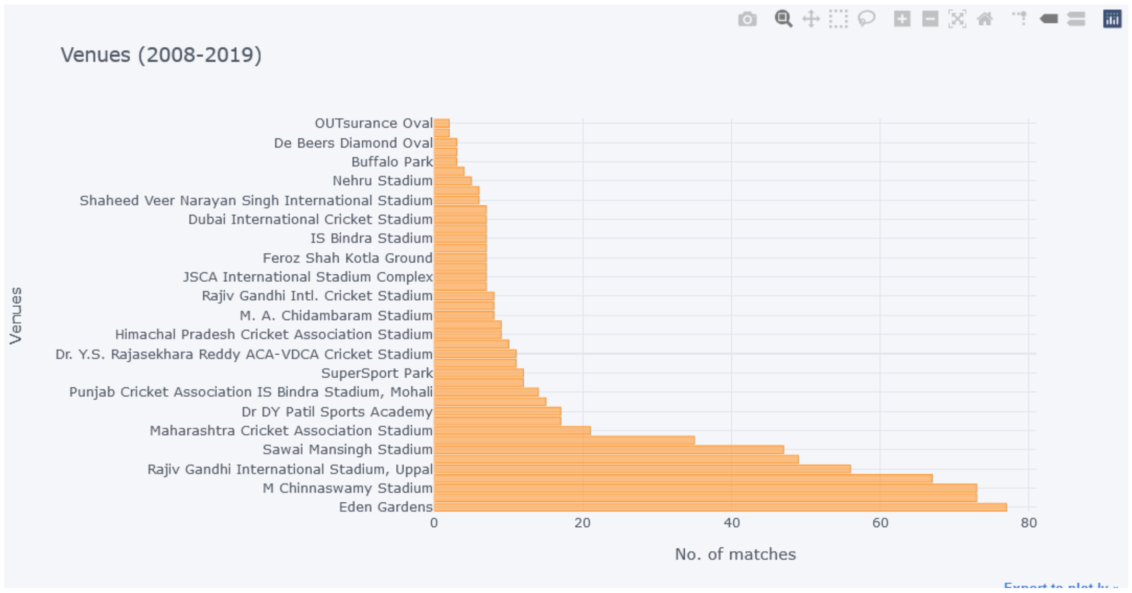
• WINS AND LOSS AND VENUES ANALYSIS

Venues Analysis : It is shown both in a tabular form and graphical form.

```
In [13]: venues = matches['venue'].value_counts()
         venues
```

```
Out[13]: Eden Gardens                                            77
         Wankhede Stadium                                        73
         M Chinnaswamy Stadium                                   73
         Feroz Shah Kotla                                        67
         Rajiv Gandhi International Stadium, Uppal               56
         MA Chidambaram Stadium, Chepauk                         49
         Sawai Mansingh Stadium                                  47
         Punjab Cricket Association Stadium, Mohali              35
         Maharashtra Cricket Association Stadium                 21
         Subrata Roy Sahara Stadium                              17
         Dr DY Patil Sports Academy                              17
         Kingsmead                                               15
         Punjab Cricket Association IS Bindra Stadium, Mohali    14
         Sardar Patel Stadium, Motera                            12
         SuperSport Park                                         12
         Brabourne Stadium                                       11
         Dr. Y.S. Rajasekhara Reddy ACA-VDCA Cricket Stadium     11
         Saurashtra Cricket Association Stadium                  10
         Himachal Pradesh Cricket Association Stadium             9
         Holkar Cricket Stadium                                   9
         M. A. Chidambaram Stadium                                8
         New Wanderers Stadium                                    8
         Rajiv Gandhi Intl. Cricket Stadium                       8
         Barabati Stadium                                         7
         JSCA International Stadium Complex                       7
         St George's Park                                         7
         Feroz Shah Kotla Ground                                  7
         Sheikh Zayed Stadium                                     7
         IS Bindra Stadium                                        7
         Newlands                                                 7
         Dubai International Cricket Stadium                      7
         M. Chinnaswamy Stadium                                   7
         Shaheed Veer Narayan Singh International Stadium         6
         Sharjah Cricket Stadium                                  6
         Nehru Stadium                                            5
         Green Park                                               4
         Buffalo Park                                             3
         Vidarbha Cricket Association Stadium, Jamtha             3
         De Beers Diamond Oval                                    3
         ACA-VDCA Stadium                                         2
         OUTsurance Oval                                          2
         Name: venue, dtype: int64
```

```
In [14]: venues.iplot(kind='bar', xTitle='No. of matches', yTitle='Venues',title='Venues (2008-2019)', orientation='h')
```

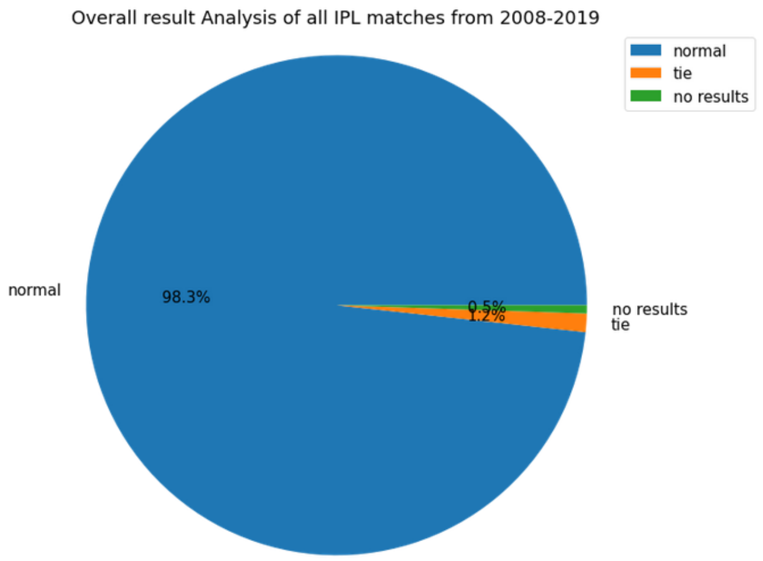


Overall result Analysis of all IPL matches from 2008-2019 :

```
In [15]: winlost = matches['result'].value_counts()
         values=[]
         for key in winlost:
             values.append(key)
         winlost
```

```
Out[15]: normal       743
         tie            9
         no result      4
         Name: result, dtype: int64
```

```
In [16]: plt.rcParams['font.size'] = 15.0
         plt.figure(figsize=(15,10))
         plt.pie(values, labels=['normal','tie','no results'],  autopct='%2.1f%%')
         plt.title('Overall result Analysis of all IPL matches from 2008-2019')
         plt.axis('equal')
         plt.legend()
         plt.show()
```

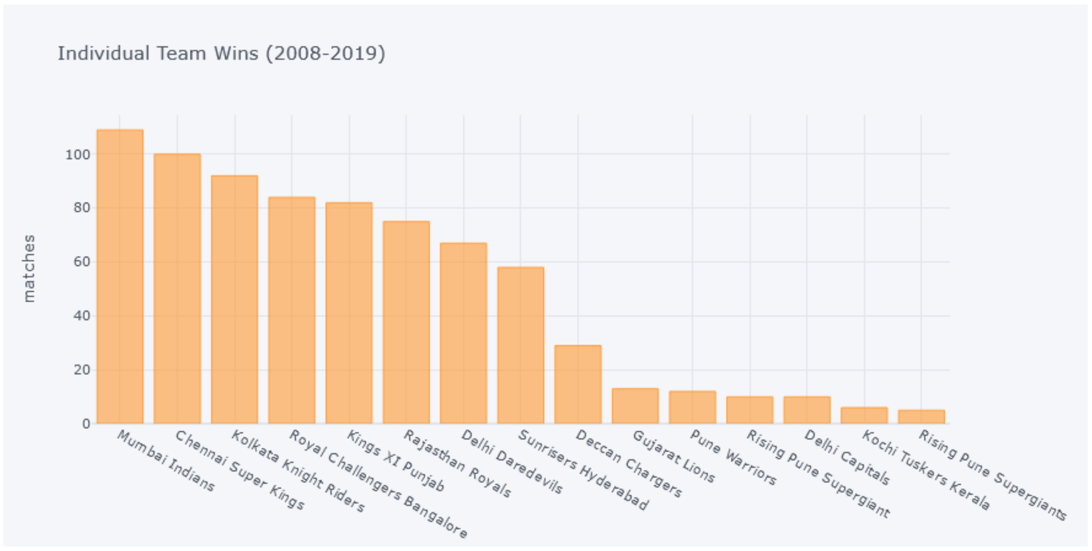


Overall Individual Team Wins from 2008-2019 :

3. Individual Team Wins

```
In [17]: wins = matches['winner'].value_counts()
         wins

Out[17]: Mumbai Indians                 109
         Chennai Super Kings            100
         Kolkata Knight Riders           92
         Royal Challengers Bangalore     84
         Kings XI Punjab                 82
         Rajasthan Royals                75
         Delhi Daredevils                67
         Sunrisers Hyderabad             58
         Deccan Chargers                 29
         Gujarat Lions                   13
         Pune Warriors                   12
         Rising Pune Supergiant          10
         Delhi Capitals                  10
         Kochi Tuskers Kerala             6
         Rising Pune Supergiants          5
         Name: winner, dtype: int64
```

```
In [18]: wins.iplot(kind='bar', xTitle='Team', yTitle='matches',title='Individual Team Wins (2008-2019)')
```

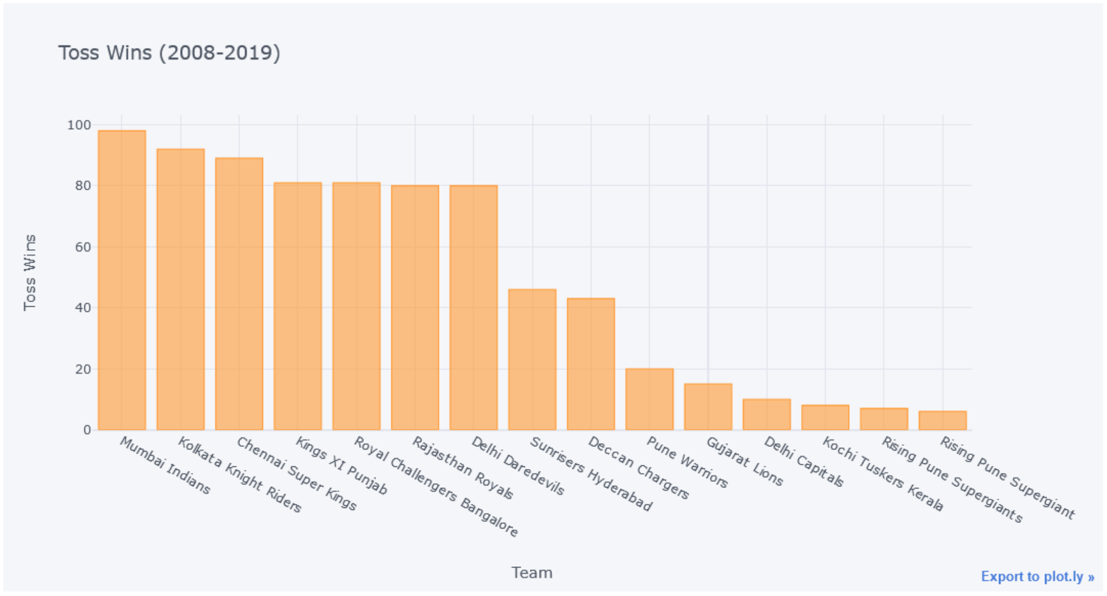


Overall Toss Wins :

4. Toss Wins

```
In [19]: toss = matches['toss_winner'].value_counts()
         toss

Out[19]: Mumbai Indians                 98
         Kolkata Knight Riders          92
         Chennai Super Kings            89
         Kings XI Punjab                81
         Royal Challengers Bangalore    81
         Delhi Daredevils               80
         Rajasthan Royals               80
         Sunrisers Hyderabad            46
         Deccan Chargers                43
         Pune Warriors                  20
         Gujarat Lions                  15
         Delhi Capitals                 10
         Kochi Tuskers Kerala            8
         Rising Pune Supergiants         7
         Rising Pune Supergiant          6
         Name: toss_winner, dtype: int64
```

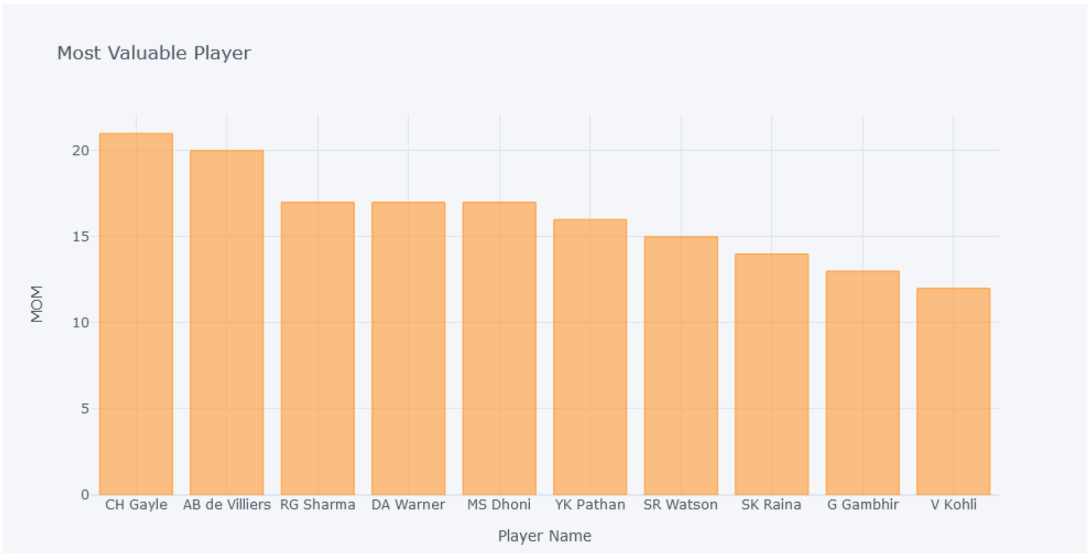
```
In [20]: toss.iplot(kind='bar', xTitle='Team', yTitle='Toss Wins',title='Toss Wins (2008-2019)')
```



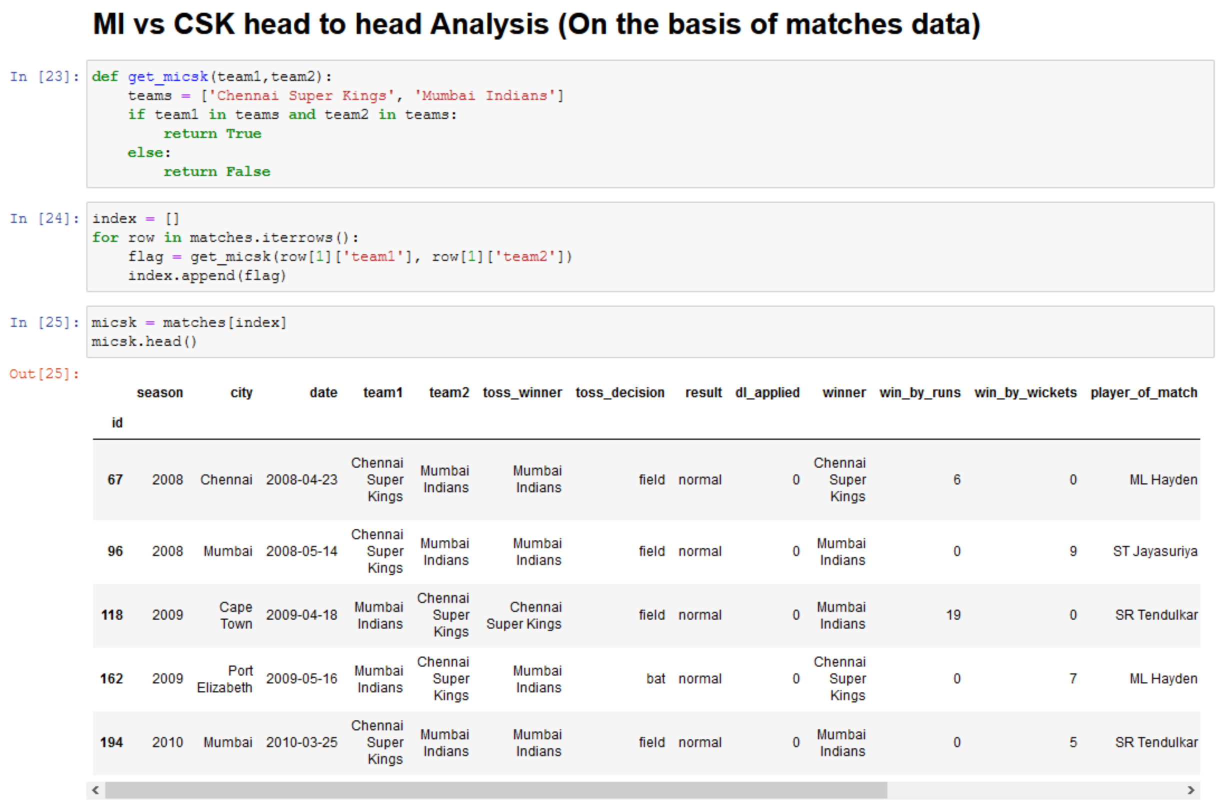
● TOP 10 MOST VALUABLE PLAYERS :

```
In [21]: mom = matches['player_of_match'].value_counts()
         mom[:10]

Out[21]: CH Gayle          21
         AB de Villiers    20
         RG Sharma         17
         DA Warner         17
         MS Dhoni          17
         YK Pathan         16
         SR Watson         15
         SK Raina          14
         G Gambhir         13
         V Kohli           12
         Name: player_of_match, dtype: int64
```

```
In [22]: mom[:10].iplot(kind='bar', xTitle='Player Name', yTitle='MOM', title='Most Valuable Player')
```



• MI vs CSK head to head (On the basis of matches data)



MI vs CSK head to head Analysis (On the basis of matches data)

```
In [23]: def get_micsk(team1,team2):
             teams = ['Chennai Super Kings', 'Mumbai Indians']
             if team1 in teams and team2 in teams:
                 return True
             else:
                 return False
```

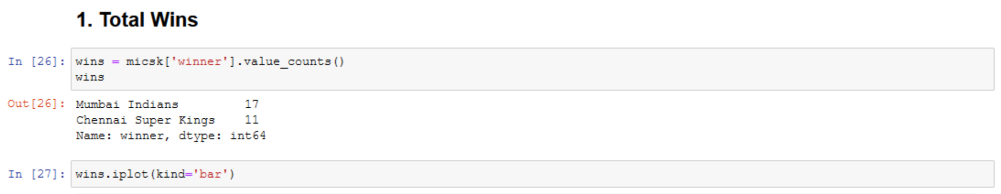
```
In [24]: index = []
         for row in matches.iterrows():
             flag = get_micsk(row[1]['team1'], row[1]['team2'])
             index.append(flag)
```

```
In [25]: micsk = matches[index]
         micsk.head()
```

Out[25]:

	season	city	date	team1	team2	toss_winner	toss_decision	result	dl_applied	winner	win_by_runs	win_by_wickets	player_of_match
id													
67	2008	Chennai	2008-04-23	Chennai Super Kings	Mumbai Indians	Mumbai Indians	field	normal	0	Chennai Super Kings	6	0	ML Hayden
96	2008	Mumbai	2008-05-14	Chennai Super Kings	Mumbai Indians	Mumbai Indians	field	normal	0	Mumbai Indians	0	9	ST Jayasuriya
118	2009	Cape Town	2009-04-18	Mumbai Indians	Chennai Super Kings	Chennai Super Kings	field	normal	0	Mumbai Indians	19	0	SR Tendulkar
162	2009	Port Elizabeth	2009-05-16	Mumbai Indians	Chennai Super Kings	Mumbai Indians	bat	normal	0	Chennai Super Kings	0	7	ML Hayden
194	2010	Mumbai	2010-03-25	Chennai Super Kings	Mumbai Indians	Mumbai Indians	field	normal	0	Mumbai Indians	0	5	SR Tendulkar

Total Wins head to head :

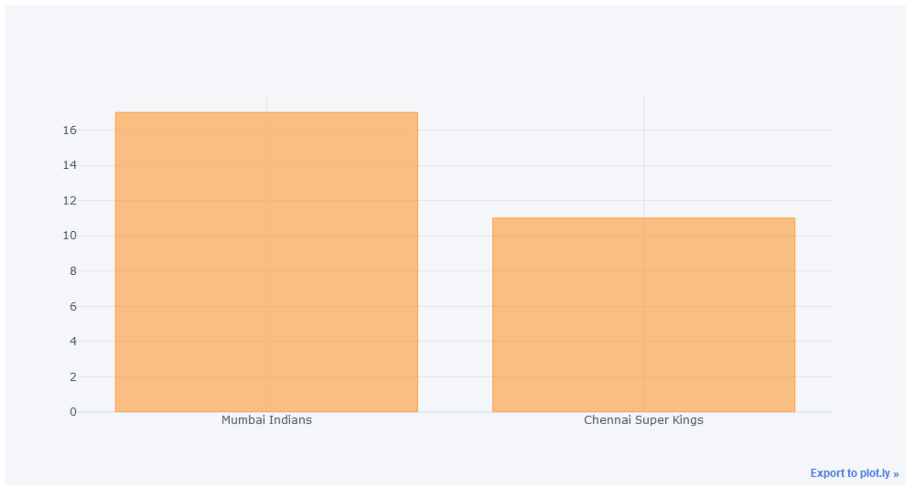


1. Total Wins

```
In [26]: wins = micsk['winner'].value_counts()
         wins
```

```
Out[26]: Mumbai Indians         17
         Chennai Super Kings    11
         Name: winner, dtype: int64
```

```
In [27]: wins.iplot(kind='bar')
```



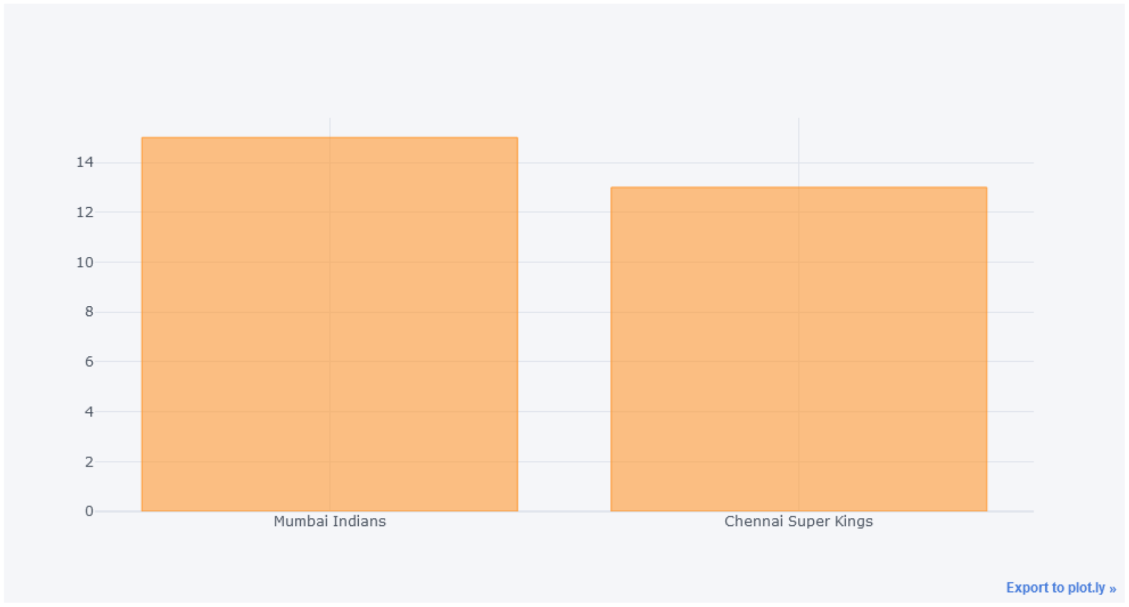
Toss Wins :

Toss Wins

```
In [28]: toss = micsk['toss_winner'].value_counts()
         toss
```

```
Out[28]: Mumbai Indians         15
         Chennai Super Kings    13
         Name: toss_winner, dtype: int64
```

```
In [29]: toss.iplot(kind='bar')
```

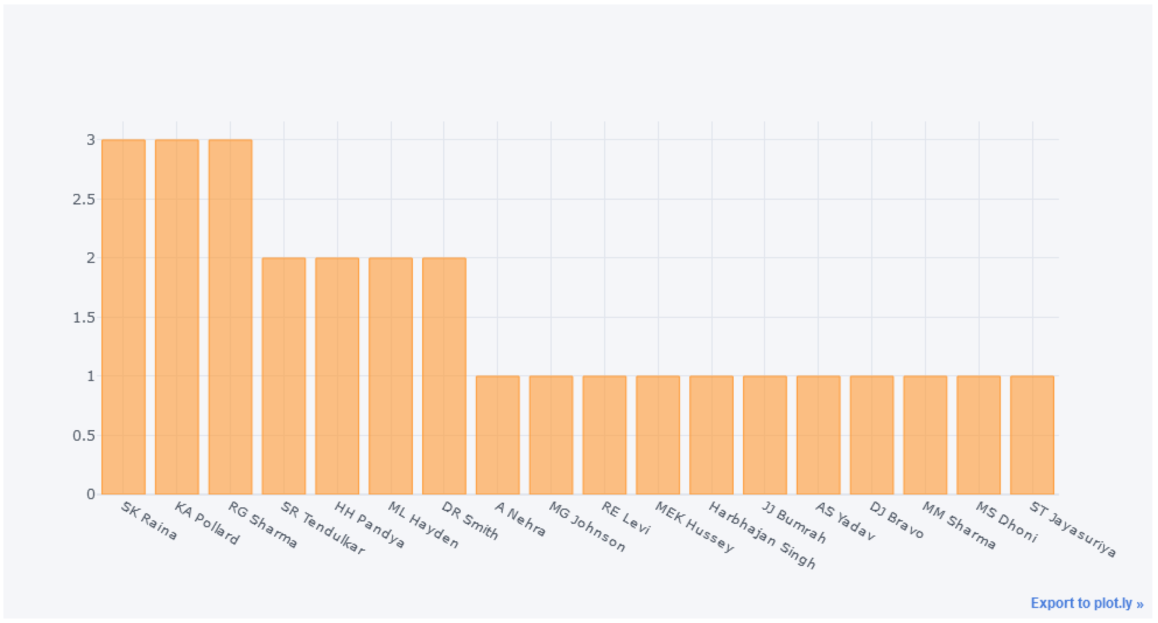


Player of the Match :

```
In [30]: mom = micsk['player_of_match'].value_counts()
         mom
```

```
Out[30]: SK Raina           3
         KA Pollard         3
         RG Sharma          3
         SR Tendulkar       2
         HH Pandya          2
         ML Hayden          2
         DR Smith           2
         A Nehra            1
         MG Johnson         1
         RE Levi            1
         MEK Hussey         1
         Harbhajan Singh    1
         JJ Bumrah          1
         AS Yadav           1
         DJ Bravo           1
         MM Sharma          1
         MS Dhoni           1
         ST Jayasuriya      1
         Name: player_of_match, dtype: int64
```

```
In [31]: mom.iplot(kind='bar')
```

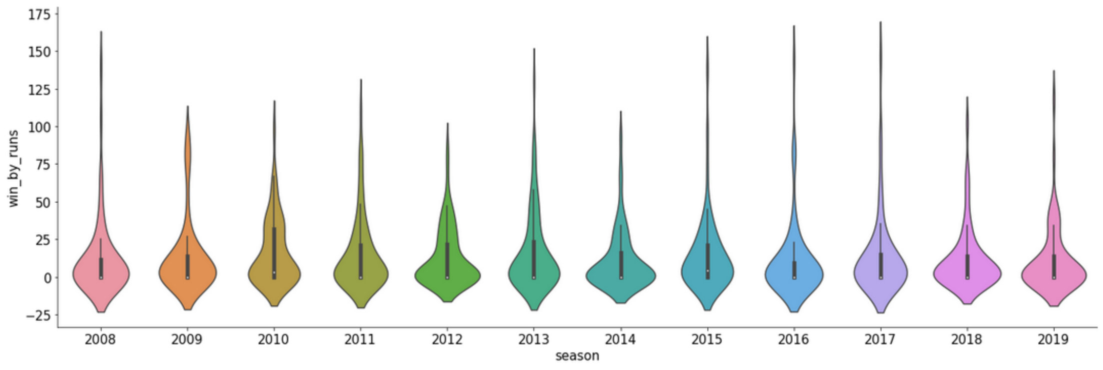


● SEASON WISE MATCH SUMMARY

Season wise wins by runs :

1. Season wise win by runs

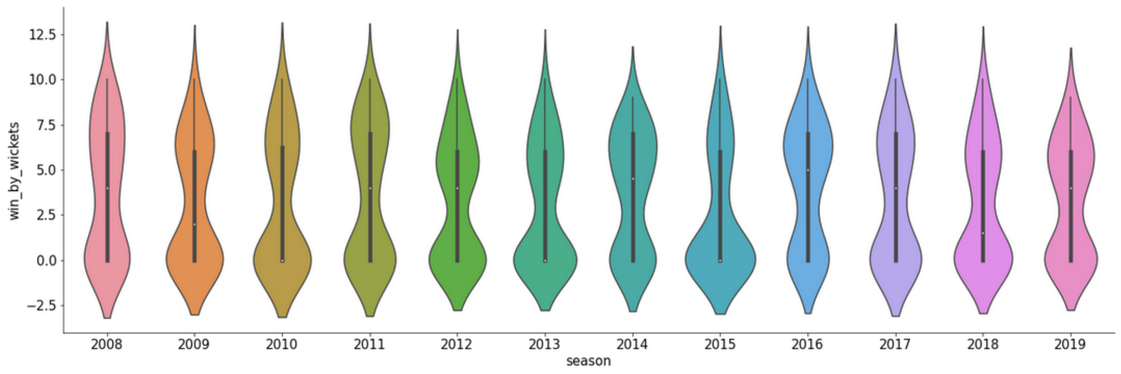
```
In [32]: sns.catplot(x='season', y='win_by_runs', data=matches, kind='violin', height=6, aspect=3)
Out[32]: <seaborn.axisgrid.FacetGrid at 0x1b2920cc5e0>
```



Season wise wins by wickets :

2. Season wise win by wickets

```
In [33]: sns.catplot(x='season', y='win_by_wickets', data=matches, kind='violin', height=6, aspect=3)

Out[33]: <seaborn.axisgrid.FacetGrid at 0x1b2920cc940>
```



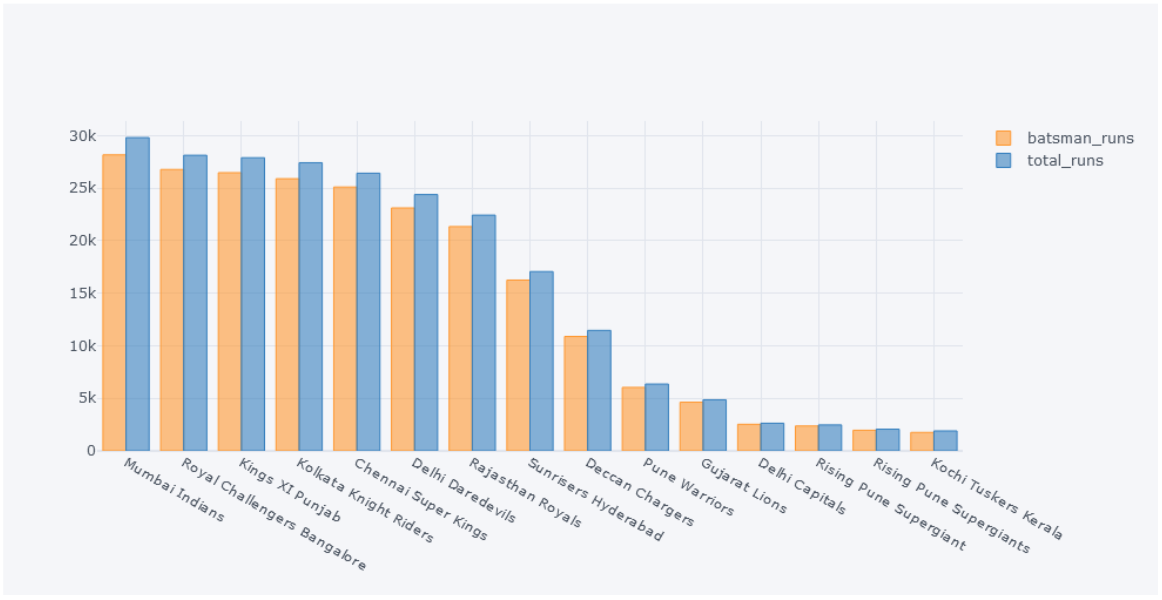
- ## BALL AND BALL ANALYSIS (On the basis of deliveries data)

Teams with their total runs and batsman runs :

```
In [65]: runs = deliveries.groupby('batting_team').sum()[['batsman_runs','total_runs']].sort_values('batsman_runs', ascending=False)
         runs

Out[65]:
```

	batsman_runs	total_runs
batting_team		
Mumbai Indians	28164	29809
Royal Challengers Bangalore	26775	28126
Kings XI Punjab	26468	27893
Kolkata Knight Riders	25895	27419
Chennai Super Kings	25104	26418
Delhi Daredevils	23115	24388
Rajasthan Royals	21341	22431
Sunrisers Hyderabad	16250	17059
Deccan Chargers	10885	11463
Pune Warriors	6040	6358
Gujarat Lions	4629	4862
Delhi Capitals	2530	2630
Rising Pune Supergiant	2370	2470
Rising Pune Supergiants	1962	2063
Kochi Tuskers Kerala	1758	1901

```
In [66]: runs.iplot(kind='bar')
```



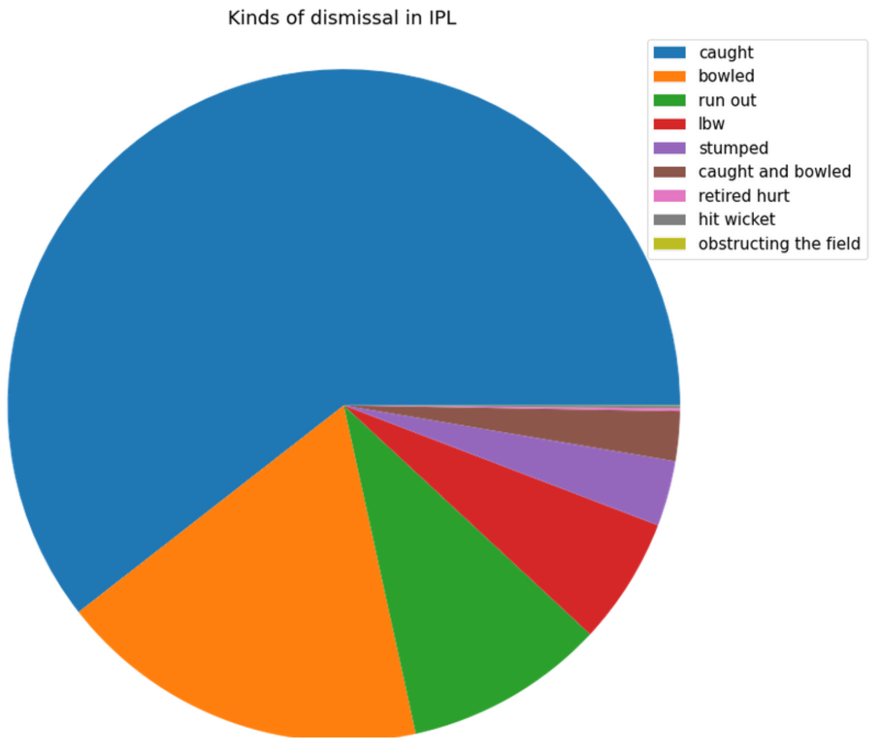
Kinds of Dismissal :

2. Dismissal Kind

```
In [37]: dis = deliveries['dismissal_kind'].value_counts().rename_axis('dismissal_kind').reset_index(name='number')
         dis
Out[37]:
```

	dismissal_kind	number
0	caught	5348
1	bowled	1581
2	run out	852
3	lbw	540
4	stumped	278
5	caught and bowled	211
6	retired hurt	12
7	hit wicket	10
8	obstructing the field	2

```
In [38]: values = list(dis['number'])
         plt.rcParams['font.size'] = 15
         plt.figure(figsize=(18,13))
         plt.pie(values, labels=None)
         plt.title('Kinds of dismissal in IPL')
         plt.axis('equal')
         plt.legend(labels=dis['dismissal_kind'].unique())
         plt.show()
```

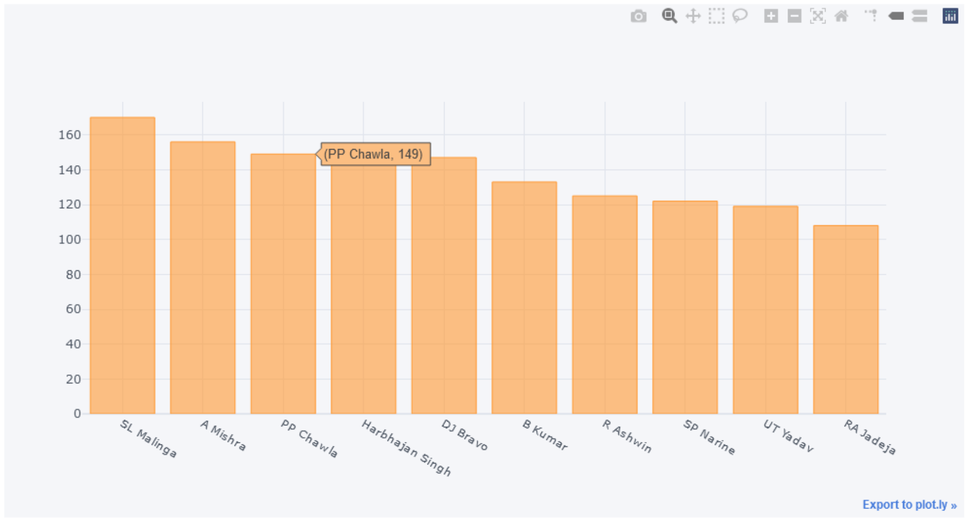


Top 10 highest wicket taker in IPL (2008 - 2019) :

3. Most Dismissals by a bowler in IPL (2008-2019)

```
In [39]: df = deliveries.query('dismissal_kind != ["run out","retired hurt", "obstructing the field","hit wicket"]')
         dff = df[['dismissal_kind', 'bowler']].dropna()
         bow = dff['bowler'].value_counts()[:10]
```

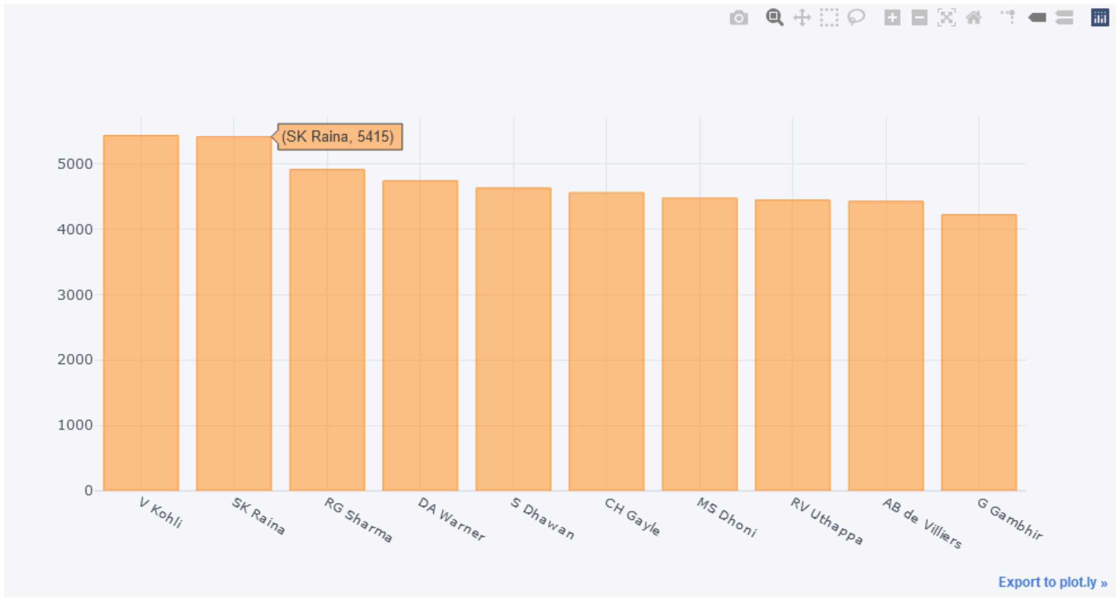
```
In [40]: bow.iplot(kind='bar')
```



Top 10 highest run scorer in IPL (2008 - 2019) :

```
In [41]: #Top 10 highest scorers of IPL
         x = deliveries[['batsman','batsman_runs']]
         bat = x.groupby('batsman').sum()['batsman_runs'].sort_values(ascending=False)[:10]
```

```
In [42]: bat.iplot(kind='bar')
```



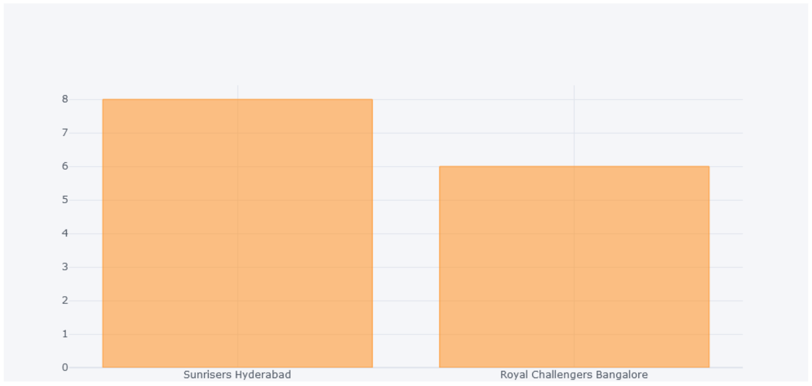
- ## RCB vs SRH head to head analysis (On the basis of matches and deliveries dataset)

Wins and Loss Analysis :

```
In [43]: def get_srhrcb(team1,team2):
             teams = ['Sunrisers Hyderabad', 'Royal Challengers Bangalore']
             if team1 in teams and team2 in teams:
                 return True
             else:
                 return False
```

```
In [44]: index = []
         for row in matches.iterrows():
             index.append(get_srhrcb(row[1]['team1'], row[1]['team2']))
         srhrcb = matches[index]
         print("Total matches between RCB AND SRH: ", len(srhrcb))

         Total matches between RCB AND SRH:  14
```

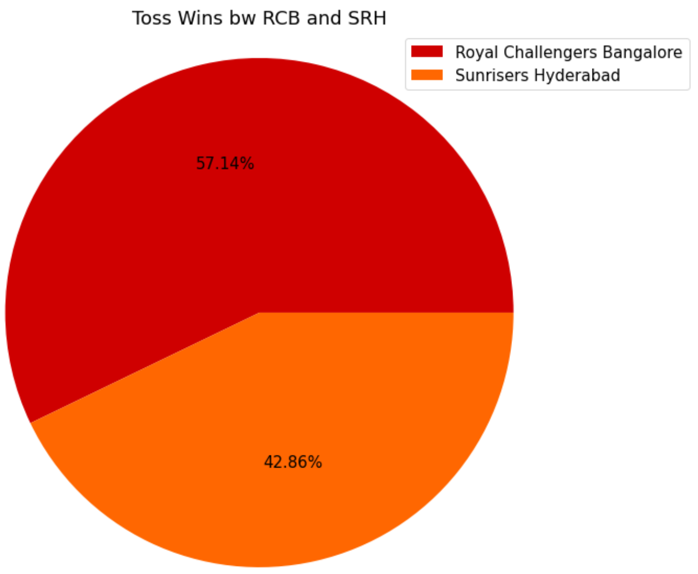
```
In [45]: srhrcb['winner'].value_counts().iplot(kind='bar')
```



Toss Wins :

In [46]:
```python
toss = srhrcb['toss_winner'].value_counts().rename_axis('team').reset_index(name='toss_wins')
values = list(toss['toss_wins'])
```

In [47]:
```python
plt.rcParams['font.size'] = 15
plt.figure(figsize=(15,10))
plt.pie(values, labels=None, autopct='%.2f%%', colors=['#cf0000','#ff6701'])
plt.title('Toss Wins bw RCB and SRH',loc='center')
plt.axis('equal')
plt.legend(labels=toss['team'].unique())
plt.show()
```



Total Extra runs :

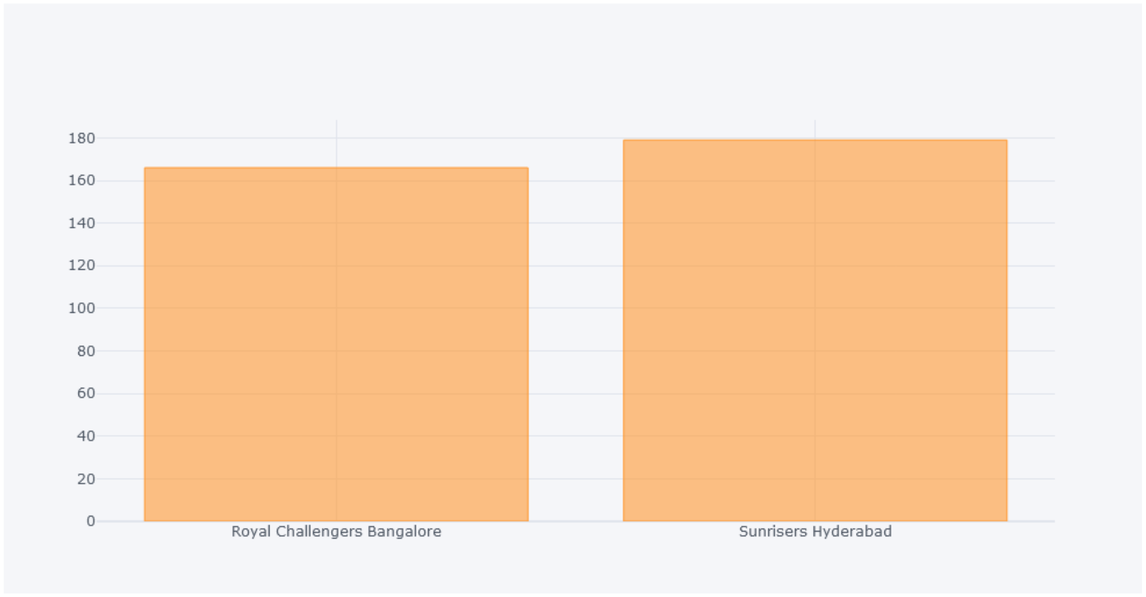
In [60]:
```python
srhrcb_ball = deliveries.query('batting_team == ["Sunrisers Hyderabad", "Royal Challengers Bangalore"] and bowling_team == ["S
extras = srhrcb_ball.groupby('match_id').sum()[['wide_runs','bye_runs','legbye_runs','noball_runs','penalty_runs', 'extra_runs
extras.insert(0,'match', np.arange(1,len(extras)+1, dtype='int'))
extras
```

Out[60]:

match_id	match	wide_runs	bye_runs	legbye_runs	noball_runs	penalty_runs	extra_runs
1	1	9	0	2	2	0	13
388	2	13	1	10	1	0	25
432	3	8	0	1	1	0	10
481	4	10	0	4	1	0	15
503	5	7	3	4	0	0	14
525	6	21	0	9	2	0	32
568	7	4	0	2	2	0	8
580	8	13	1	3	1	0	18
603	9	5	1	1	1	0	8
636	10	15	1	7	0	0	23
7932	11	8	0	5	1	0	14
7944	12	5	1	3	0	0	9
11147	13	4	4	1	0	0	9
11345	14	9	0	3	1	0	13

Average target score :

```
In [51]: target = srhrcb_ball.groupby('batting_team').sum()['total_runs']
         target[0] =  round(target[0]/len(srhrcb),0)
         target[1] =  round(target[1]/len(srhrcb),0)
         target.iplot('bar')
```

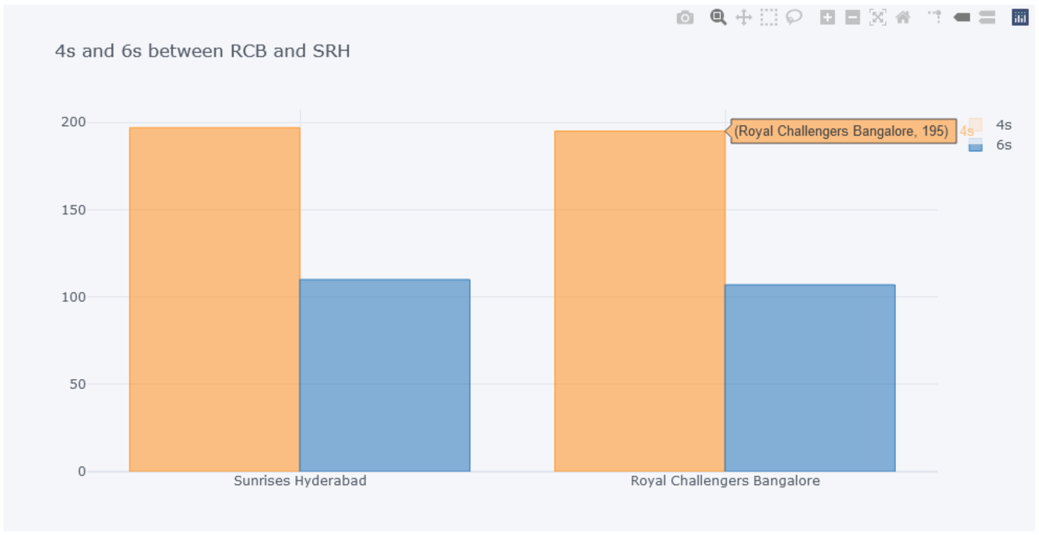


4s and 6s between SRH and RCB :

```
In [52]: runs = srhrcb_ball.query('batsman_runs == [4,6]')
```

```
In [53]: srh_runs = runs.query('batting_team == "Sunrisers Hyderabad"')['batsman_runs'].value_counts()
         rcb_runs = runs.query('batting_team == "Royal Challengers Bangalore"')['batsman_runs'].value_counts()
```

```
In [54]: data = {
             'team': ["Sunrises Hyderabad", "Royal Challengers Bangalore"],
             '4s': [srh_runs[4], rcb_runs[4]],
             '6s': [srh_runs[6], rcb_runs[6]]
         }
         table = pd.DataFrame(data,columns=['team','4s','6s'])
         tab = table.set_index('team')
         tab.iplot(kind='bar', title='4s and 6s between RCB and SRH')
```



Average batting score and wickets taken in powerplay :

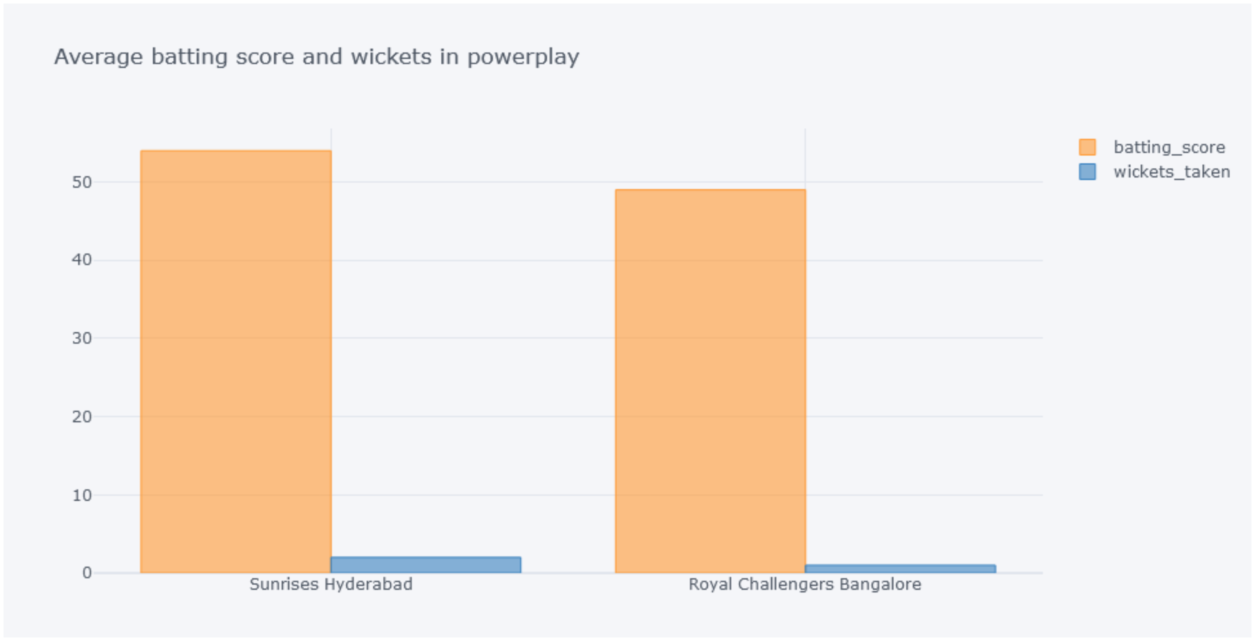
```
In [55]: srh_pow = srhrcb_ball.query('over<=6 and batting_team == "Sunrisers Hyderabad"').sum()['total_runs']
         rcb_pow = srhrcb_ball.query('over<=6 and batting_team == "Royal Challengers Bangalore"').sum()['total_runs']

         srh_pow_avg = srh_pow//len(srhrcb)
         rcb_pow_avg = rcb_pow//len(srhrcb)
         print("Total Powerplay score of RCB against SRH in",len(srhrcb),"matches =", rcb_pow)
         print("Total Powerplay score of SRH against RCB in",len(srhrcb),"matches =", srh_pow)
```

```
Total Powerplay score of RCB against SRH in 14 matches = 687
Total Powerplay score of SRH against RCB in 14 matches = 760
```

```
In [56]: srh_wicks = len(srhrcb_ball.query('over<=6 and bowling_team == "Sunrisers Hyderabad"')['dismissal_kind'].dropna())
         rcb_wicks = len(srhrcb_ball.query('over<=6 and bowling_team == "Royal Challengers Bangalore"')['dismissal_kind'].dropna())

         srh_wicks_avg = round(srh_wicks/len(srhrcb),0)
         rcb_wicks_avg = round(rcb_wicks/len(srhrcb),0)
         print("Total Powerplay wickets of RCB against SRH in",len(srhrcb),"matches =", rcb_wicks)
         print("Total Powerplay wickets of SRH against RCB in",len(srhrcb),"matches =", srh_wicks)
```

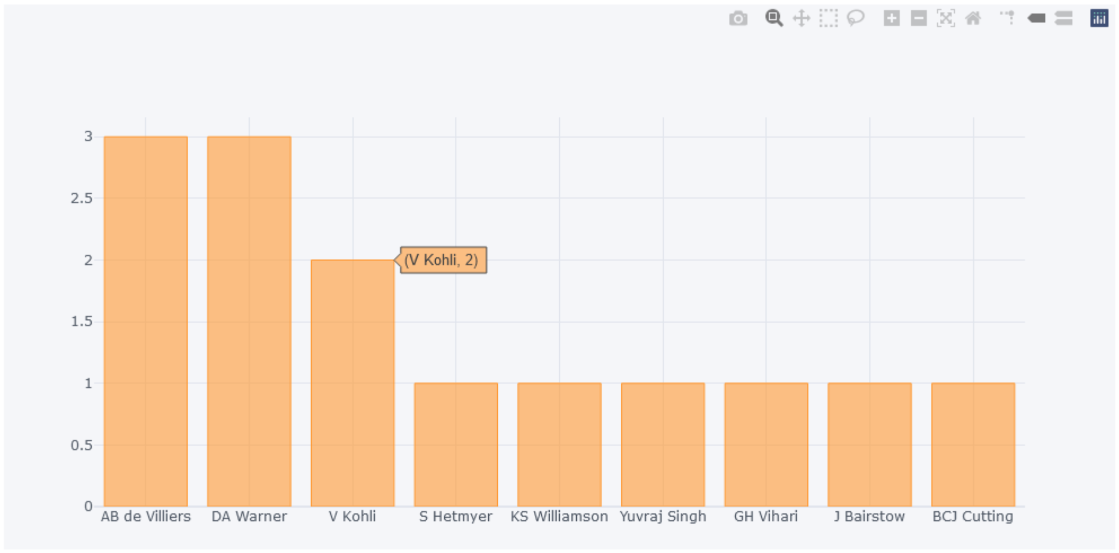
```
Total Powerplay wickets of RCB against SRH in 14 matches = 13
Total Powerplay wickets of SRH against RCB in 14 matches = 24
```

```
In [57]: pow_data = {
             'team': ["Sunrises Hyderabad", "Royal Challengers Bangalore"],
             'batting_score': [srh_pow_avg, rcb_pow_avg],
             'wickets_taken': [srh_wicks_avg, rcb_wicks_avg]
         }
         pow_table = pd.DataFrame(pow_data,columns=['team','batting_score','wickets_taken'])
         pow_tab = pow_table.set_index('team')
         pow_tab.iplot(kind='bar', title='Average batting score and wickets in powerplay')
```



Most Valuable Player :

```
In [58]: mvp = srhrcb['player_of_match'].value_counts()
         mvp

Out[58]: AB de Villiers    3
         DA Warner         3
         V Kohli           2
         S Hetmyer         1
         KS Williamson     1
         Yuvraj Singh      1
         GH Vihari         1
         J Bairstow        1
         BCJ Cutting       1
         Name: player_of_match, dtype: int64

In [59]: mvp.iplot('bar')
```



RESULTS AND DISCUSSION

In this section we will make use of the tabular and graphical data to find an outcome and reach a conclusion.

- Venue Analysis : According to the data, if we look at the top 5 IPL venues that are Eden Garden, Wankhede Stadium, Chinnaswamy Stadium, Feroz Shah Kotla and Chepauk Stadium. They are situated in metropolitan areas of the country which attracts a lot of people. Also from this data IPL has introduced the concept of fan parks in those regions which are very far apart from metropolitan areas where people gather in a lot of numbers to enjoy the match on big screens and play games. These are some of the factors that tell us why IPL has been successful for many years.

- Overall result Analysis of all IPL matches from 2008-2019 : From the chart we observed that 98.3% are normal, 0.5% with no results and 1.2% end up with a tie. From this we can conclude that there is a very low possibility that a match is abandoned, which also makes IPL a very interesting sporting league in the country. The overall team wins and the toss wins reveals that the team who wins the toss has a higher chance of winning the game, which is justified because a lot goes into planning and strategy making before the game and winning the toss makes it easier for the team to execute their plans and end up on the winning side.

- Top 10 most valuable players : The data shows that Chris Gayle from the West Indies has won most of the man of the matches in the IPL followed by batsmen like AB de Villiers, MS Dhoni and all rounders like Yusuf Pathan and Shane Watson. But what it also shows is that most man of the matches go towards the batsman side, which shows that IPL is very much dominated by the batsman rather than the bowlers. This data is also used during the auction time for bidding purposes. These players have a tendency to get a high bid who are released by their franchise, though most of them are retained because of their high performances. This data can also be used by the sponsors for advertisement purposes to engage fans and generate revenues.

- MI vs CSK head to head (On the basis of matches dataset) : The data reveals that the Mumbai Indians have won 17 matches and CSK have won only 11 matches. The toss data also reveals that MI have 15 toss wins and CSK have only 13. So as discussed earlier the trend shows that toss plays a huge role in the game. Keiron Pollard emerged as most valuable player against CSK followed by Suresh Raina, Rohit Sharma, DR smith, etc. This shows that the Indians and the West Indians players contribute most towards the winning side as Indians are familiar with the conditions and West Indians are big hitters . The top valuable players data can be used to study the players who emerged as the match winners and the management can make new strategies against them to win the match.

- Season wise Analysis : From both the season wise win by runs and win by wickets it shows a majority of the IPL games are close encounters, which makes them interesting and attracts a lot of people to watch their favourite team win the game. This also shows that each and every team is balanced and no team is superior in this league.

- Ball by Ball Analysis (On the basis of deliveries dataset) : The first data which is the teams with their batsman runs and total runs shows that MI is at the top position because they have a core set if Indian batsmen and some big hitting West Indian all rounders. Then the next set of data shows various dismissal kinds in IPL, as catches are the most common type of dismissal the teams can focus more on their fielding. As they in cricket 'Catches win matches' it is also important to have a strong fielding unit.
  The next data shows that Lasith Malinga of Sri Lanka is the most successful bowler in IPL followed by Indian spinners like Amit Mishra, Harbhajan Singh, Piyush Chawla and some fast bowlers like DJ Bravo of West Indies and Bhuvneshwar Kumar. This data reveals that spinners dominate than the fast bowlers and emerge as match winners and also fast bowlers like Malinga and Bravo bring variations to their bowling which is very effective in taking wickets. The subcontinent conditions suites spin bowling and fast bowling variations which is quite evident from the above data.
  The last set of data shows Virat Kohli is the batsman with most runs, followed by Suresh Raina and Rohit Sharma which again shows that Indian players dominate in this league because of familiarity with the home conditions. Some foreign players like David Warner, CH Gayle and AB de Villiers are also in the list.
  From all the above observations, we see that now teams are investing more on young Indian talent rather than foreign players because they can get high future returns from them.

- RCB vs SRH head to head (On the basis of deliveries and matches dataset) : First of all

looking at the head to head wins and toss wins, it clearly shows that SRH is a stronger side than RCB. Though from the previous data, it shows that RCB has more runs than SRH, but from the wins we can see SRH is a more balanced side than RCB.
Though RCB has more runs but in SRH vs RCB games the average target score of SRH is more than RCB i.e. 179 of SRH and 166 of RCB which shows the batsmen of SRH has contributed pretty well but also the bowling side of SRH has always been successful to restrict RCB to a smaller total.
According to the power play analysis the average batting score and wicket taken by SRH is 54 and 2 and RCB has 49 and 1 respectively. Powerplay plays a huge role in the T20s and in this data it clearly shows that SRH has dominated over RCB in the powerplay as well.

CONCLUSION

In this paper, the performance of cricket players(batsmen) and toss related analysis in IPL from season 2008-2019 has been visualized. Finding out the hidden parameters, patterns and attributes that lead to the outcome of a cricket match helps the team owners and selectors to recognize better players. The salary of IPL cricket players is decided through the auction process. Thus, it is a part of the franchise and a matter of decision making about which player to be bidded for and at what cost by the past performance of players in IPL. Every Selector needs young and dynamic players who can handle the pressure calmly, and go towards the winning line.
This report also includes head to head match analysis that can be used by team managements, coaches and players to plan against the opponent team and increase their win outcome. Sponsors also can make use of this data to invest in young players and other popular players in their advertisements to get more engagements and business.