IPL Data Analysis from visualisation

My target for this notebook is to learn interactive data visualisation with plotly,matplotlib,cufflinks. I will try out various plots using plotly,matplotlib,cufflinks and finally try to put altogether in a dashboard.

Importing the libraries

In [93]:
```
import numpy as np
import pandas as pd
from matplotlib import pyplot as plt
import seaborn as sns
import plotly.offline as pyo
from plotly.offline import download_plotlyjs, init_notebook_mode, plot, iplot
import plotly.graph_objs as go
import random
import cufflinks as cf
cf.go_offline()
cf.set_config_file(offline=False, world_readable=True)
def random_colors(number_of_colors):
    color = ["#"+''.join([random.choice('0123456789ABCDEF') for j in range(6)])
                 for i in range(number_of_colors)]
    return color
```

Loading the datasets

In [94]:
```
matches = pd.read_csv('IPL Matches 2008-2020.csv')
balls = pd.read_csv('IPL Ball-by-Ball 2008-2020.csv')
```

First look at the ball by ball data

In [95]:
```
balls.head()
```

Out[95]:

	id	inning	over	ball	batsman	non_striker	bowler	batsman_runs	extra_runs	total_runs	non_boundary	is_wicket	dismissal_kind	player_dismisse
0	335982	1	6	5	RT Ponting	BB McCullum	AA Noffke	1	0	1	0	0	NaN	Nal
1	335982	1	6	6	BB McCullum	RT Ponting	AA Noffke	1	0	1	0	0	NaN	Nal
2	335982	1	7	1	BB McCullum	RT Ponting	Z Khan	0	0	0	0	0	NaN	Nal
3	335982	1	7	2	BB McCullum	RT Ponting	Z Khan	1	0	1	0	0	NaN	Nal
4	335982	1	7	3	RT Ponting	BB McCullum	Z Khan	1	0	1	0	0	NaN	Nal

First look at the matches data

In [96]:
```
matches.head()
```

Out[96]:

	id	city	date	player_of_match	venue	neutral_venue	team1	team2	toss_winner	toss_decision	winner	result	result_
0	335982	Bangalore	2008-04-18	BB McCullum	M Chinnaswamy Stadium	0	Royal Challengers Bangalore	Kolkata Knight Riders	Royal Challengers Bangalore	field	Kolkata Knight Riders	runs	
1	335983	Chandigarh	2008-04-19	MEK Hussey	Punjab Cricket Association Stadium, Mohali	0	Kings XI Punjab	Chennai Super Kings	Chennai Super Kings	bat	Chennai Super Kings	runs	
2	335984	Delhi	2008-04-19	MF Maharoof	Feroz Shah Kotla	0	Delhi Daredevils	Rajasthan Royals	Rajasthan Royals	bat	Delhi Daredevils	wickets	
3	335985	Mumbai	2008-04-20	MV Boucher	Wankhede Stadium	0	Mumbai Indians	Royal Challengers Bangalore	Mumbai Indians	bat	Royal Challengers Bangalore	wickets	
4	335986	Kolkata	2008-04-20	DJ Hussey	Eden Gardens	0	Kolkata Knight Riders	Deccan Chargers	Deccan Chargers	bat	Kolkata Knight Riders	wickets	

In [97]:
```
matches.columns
```

Out[97]:
```
Index(['id', 'city', 'date', 'player_of_match', 'venue', 'neutral_venue',
       'team1', 'team2', 'toss_winner', 'toss_decision', 'winner', 'result',
       'result_margin', 'eliminator', 'method', 'umpire1', 'umpire2'],
      dtype='object')
```

In [98]:
```
balls.columns
```

Out[98]:
```
Index(['id', 'inning', 'over', 'ball', 'batsman', 'non_striker', 'bowler',
       'batsman_runs', 'extra_runs', 'total_runs', 'non_boundary', 'is_wicket',
       'dismissal_kind', 'player_dismissed', 'fielder', 'extras_type',
       'batting_team', 'bowling_team'],
      dtype='object')
```

In [99]:
```
print(matches['winner'].unique())
print(matches['city'].unique())
```

```
['Kolkata Knight Riders' 'Chennai Super Kings' 'Delhi Daredevils'
 'Royal Challengers Bangalore' 'Rajasthan Royals' 'Kings XI Punjab'
 'Deccan Chargers' 'Mumbai Indians' 'Pune Warriors' 'Kochi Tuskers Kerala'
 nan 'Sunrisers Hyderabad' 'Rising Pune Supergiants' 'Gujarat Lions'
 'Rising Pune Supergiant' 'Delhi Capitals']
['Bangalore' 'Chandigarh' 'Delhi' 'Mumbai' 'Kolkata' 'Jaipur' 'Hyderabad'
 'Chennai' 'Cape Town' 'Port Elizabeth' 'Durban' 'Centurion' 'East London'
 'Johannesburg' 'Kimberley' 'Bloemfontein' 'Ahmedabad' 'Cuttack' 'Nagpur'
 'Dharamsala' 'Kochi' 'Indore' 'Visakhapatnam' 'Pune' 'Raipur' 'Ranchi'
 'Abu Dhabi' nan 'Rajkot' 'Kanpur' 'Bengaluru' 'Dubai' 'Sharjah']
```

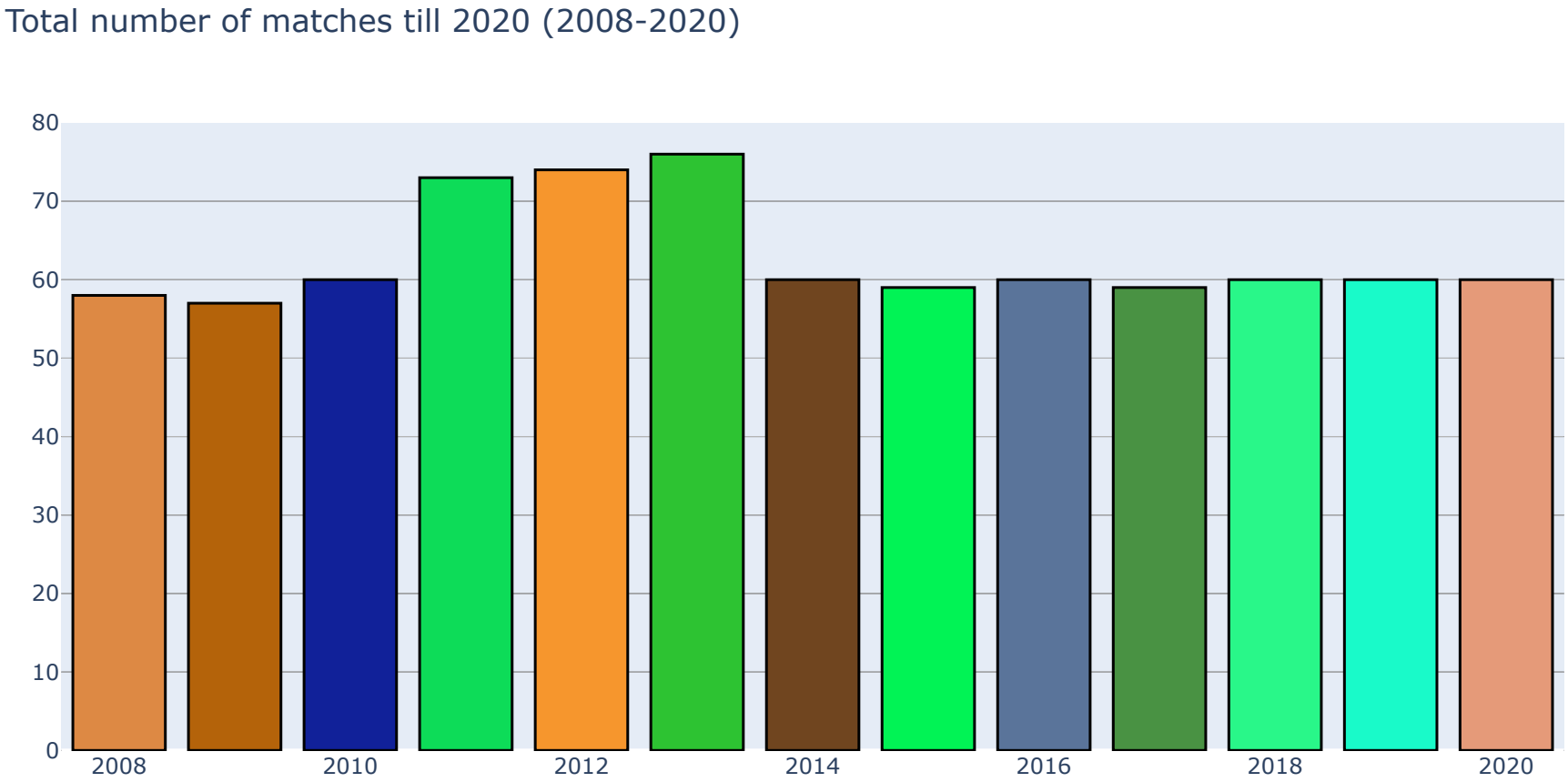
In [100]:
```
matches.team1.replace({'Rising Pune Supergiants' : 'Rising Pune Supergiant'},regex=True,inplace=True)
matches.team2.replace({'Rising Pune Supergiants' : 'Rising Pune Supergiant'},regex=True,inplace=True)
matches.winner.replace({'Rising Pune Supergiants' : 'Rising Pune Supergiant'},regex=True,inplace=True)
matches.venue.replace({'Feroz Shah Kotla Ground':'Feroz Shah Kotla',
                      'M Chinnaswamy Stadium':'M. Chinnaswamy Stadium',
                      'MA Chidambaram Stadium, Chepauk':'M.A. Chidambaram Stadium',
                       'M. A. Chidambaram Stadium':'M.A. Chidambaram Stadium',
                       'Punjab Cricket Association IS Bindra Stadium, Mohali':'Punjab Cricket Association Stadium',
                       'Punjab Cricket Association Stadium, Mohali':'Punjab Cricket Association Stadium',
                       'IS Bindra Stadium':'Punjab Cricket Association Stadium',
                      'Rajiv Gandhi International Stadium, Uppal':'Rajiv Gandhi International Stadium',
                      'Rajiv Gandhi Intl. Cricket Stadium':'Rajiv Gandhi International Stadium'},regex=True,inplace=True)
```

Total number of matches each season

In [101]:
```
matches["season"] = matches["date"].apply(lambda x:x.split("-")[0])
```

In [102]:

```
seasons_data = matches["season"].value_counts()
total_matches = matches.groupby('season')['id'].count()
data = [go.Bar(
    x = seasons_data.index,
    y = seasons_data.values,
    marker = dict(color = random_colors(len(seasons_data.index)),line=dict(color='#000000', width=1.5))
)]

layout = go.Layout(
   {
      "title":"Total number of matches till 2020 (2008-2020)",
   }
)

fig = go.Figure(data=data,layout = layout)
iplot(fig)
```



Number of Player of the match

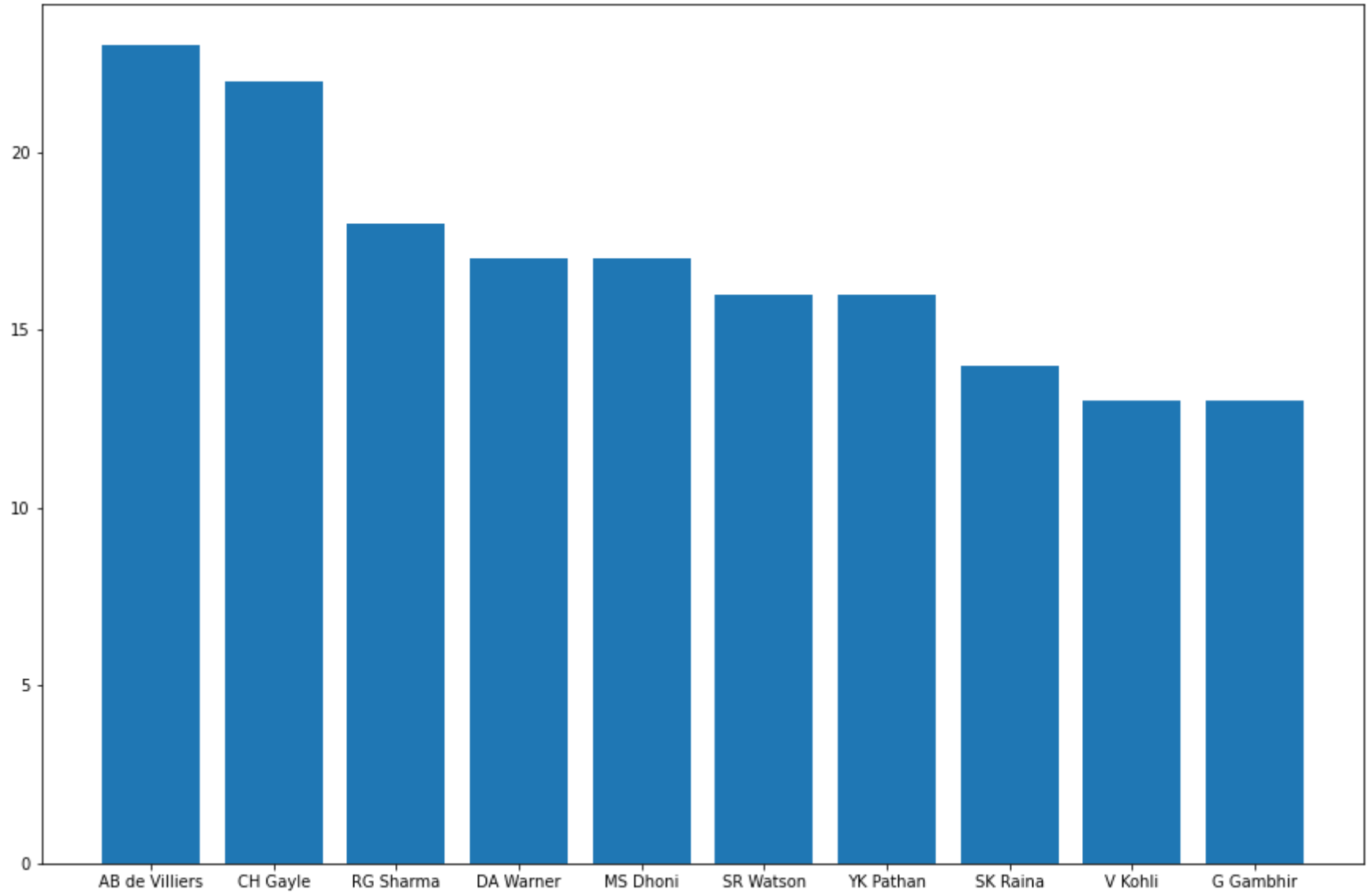
In [103]:
```
matches['player_of_match'].value_counts()[0:10]
```

Out[103]:
```
AB de Villiers    23
CH Gayle          22
RG Sharma         18
DA Warner         17
MS Dhoni          17
SR Watson         16
YK Pathan         16
SK Raina          14
V Kohli           13
G Gambhir         13
Name: player_of_match, dtype: int64
```

In [104]:
```
list(matches['player_of_match'].value_counts()[0:10].keys())
```

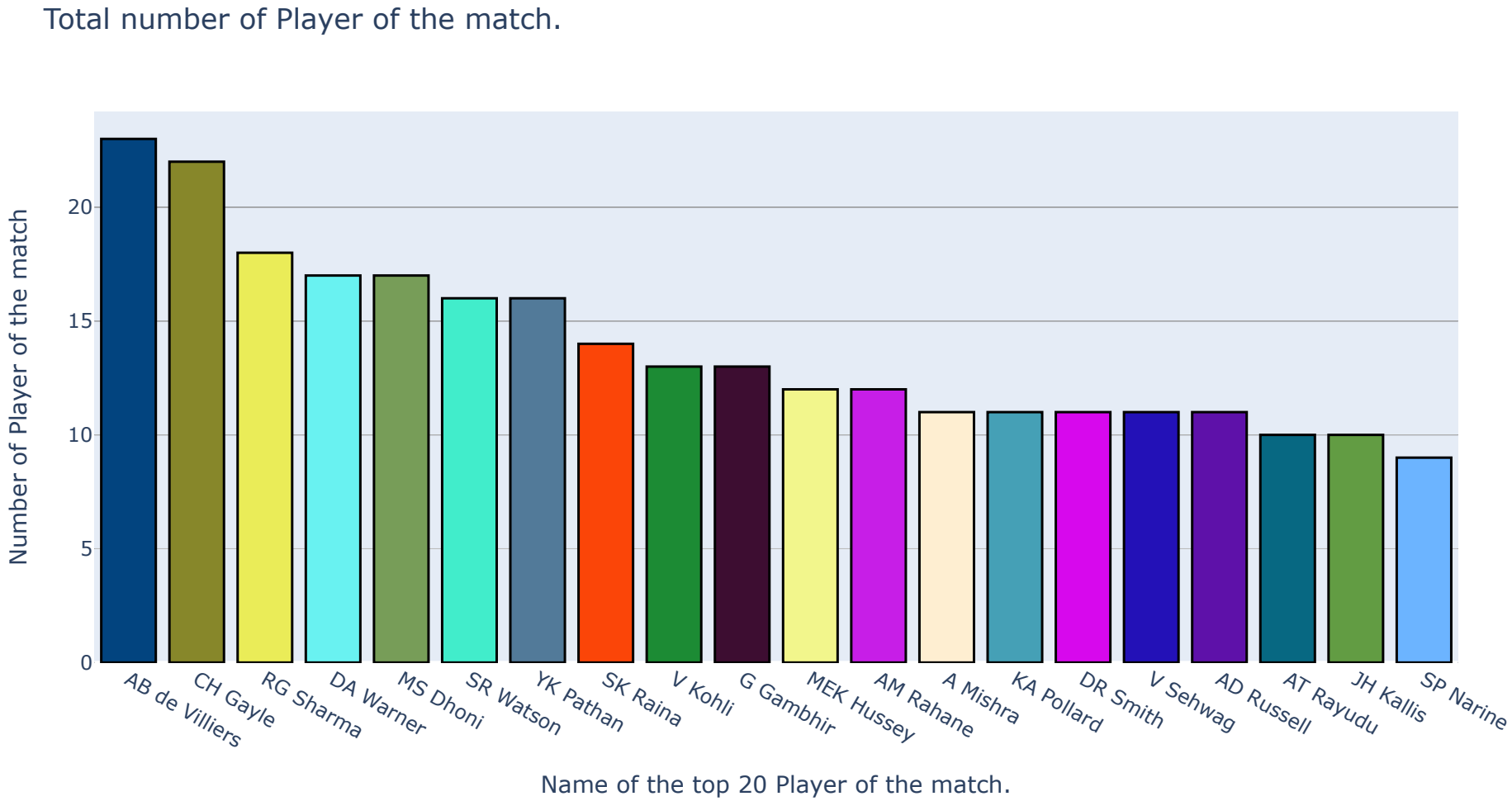
Out[104]:
```
['AB de Villiers',
 'CH Gayle',
 'RG Sharma',
 'DA Warner',
 'MS Dhoni',
 'SR Watson',
 'YK Pathan',
 'SK Raina',
 'V Kohli',
 'G Gambhir']
```

In [105]:
```
plt.figure(figsize=(15,10))
plt.bar(list(matches['player_of_match'].value_counts()[0:10].keys()),list(matches['player_of_match'].value_counts()[0:10]))
plt.show()
```



In [106]:

```
data = [go.Bar(
    x = matches['player_of_match'].value_counts()[:20].index,
    y = matches['player_of_match'].value_counts()[:20].values,
    marker = dict(color = random_colors(20),line=dict(color='#000000', width=1.5))
)]


layout = go.Layout(title="Total number of Player of the match. ",
                   xaxis=dict(title="Name of the top 20 Player of the match."),
                   yaxis=dict(title="Number of Player of the match"))

# Create figure with all prepared data for plot
fig = go.Figure(data=data, layout=layout)
# Create a plot in your Python script directory with name "bar-chart.html"
iplot(fig)
```

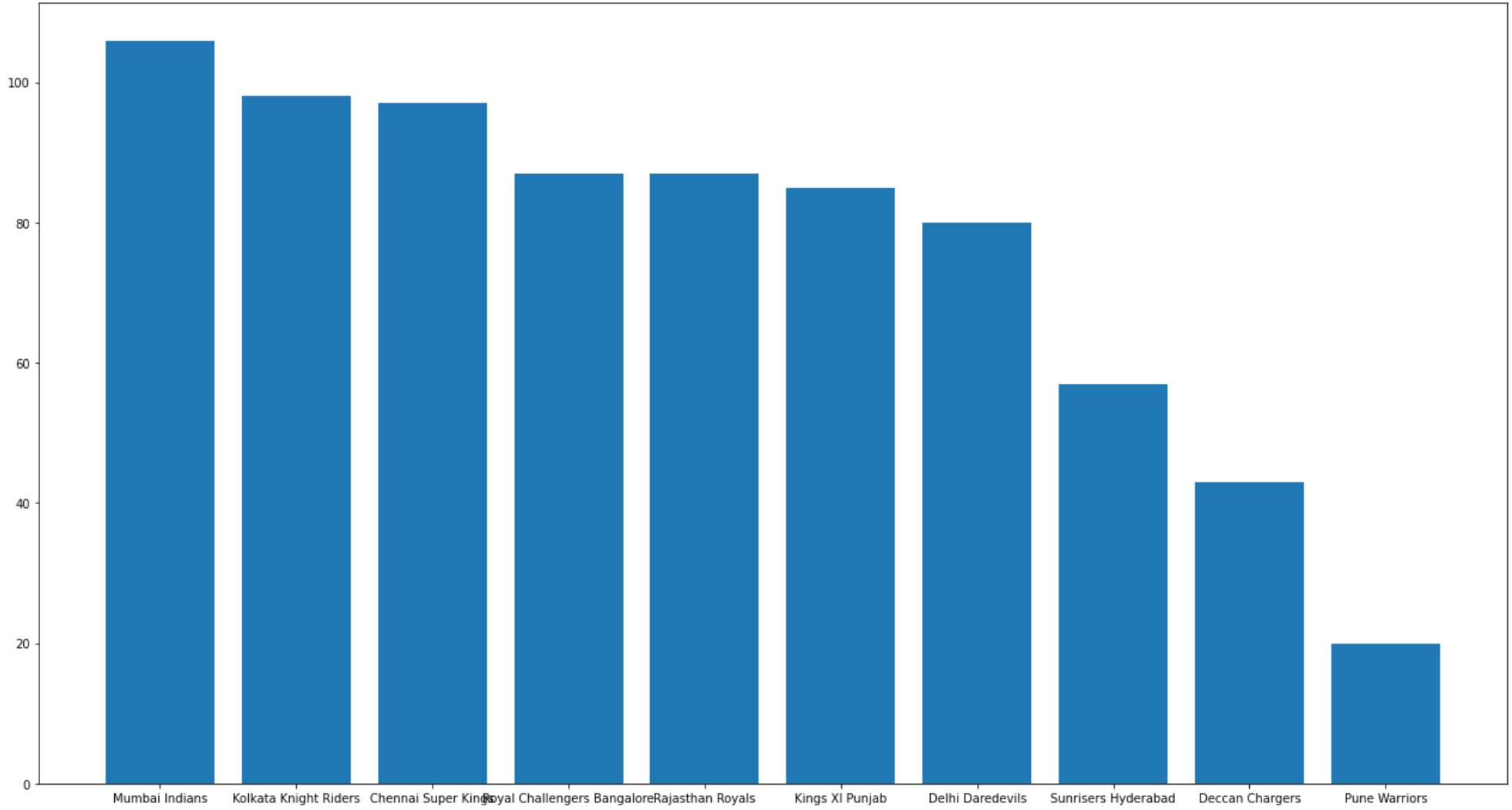


Total number of toss Win by Each Teams

In [107]:
```
matches['toss_winner'].value_counts().keys()
```

Out[107]:
```
Index(['Mumbai Indians', 'Kolkata Knight Riders', 'Chennai Super Kings',
       'Royal Challengers Bangalore', 'Rajasthan Royals', 'Kings XI Punjab',
       'Delhi Daredevils', 'Sunrisers Hyderabad', 'Deccan Chargers',
       'Pune Warriors', 'Delhi Capitals', 'Gujarat Lions',
       'Kochi Tuskers Kerala', 'Rising Pune Supergiants',
       'Rising Pune Supergiant'],
      dtype='object')
```

In [108]:
```
plt.figure(figsize=(22,12))
plt.bar(list(matches['toss_winner'].value_counts()[0:10].keys()),list(matches['toss_winner'].value_counts()[0:10]))
plt.show()
```

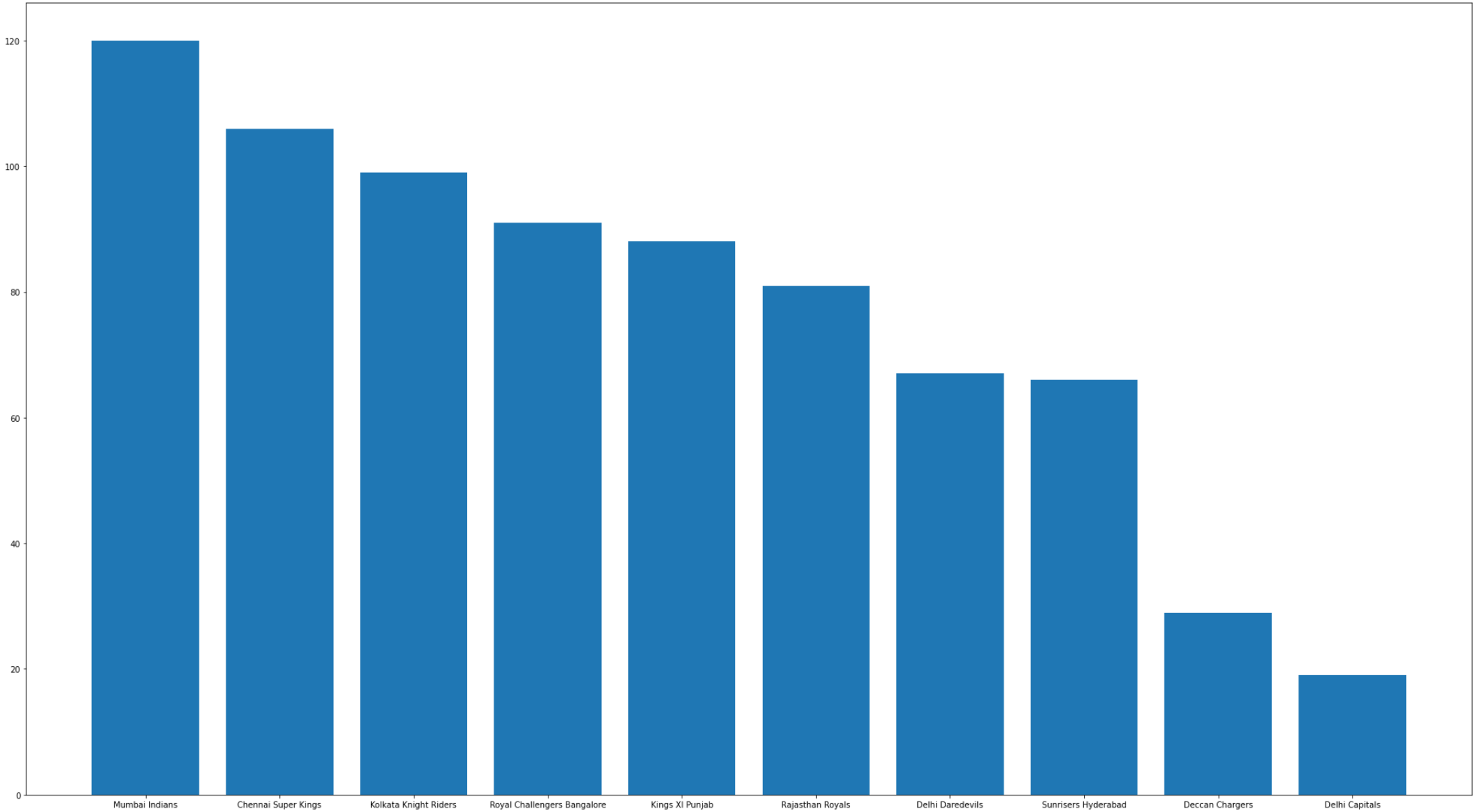


Total number of matches Win by Each Teams

In [109]:
```
matches['winner'].value_counts()
```

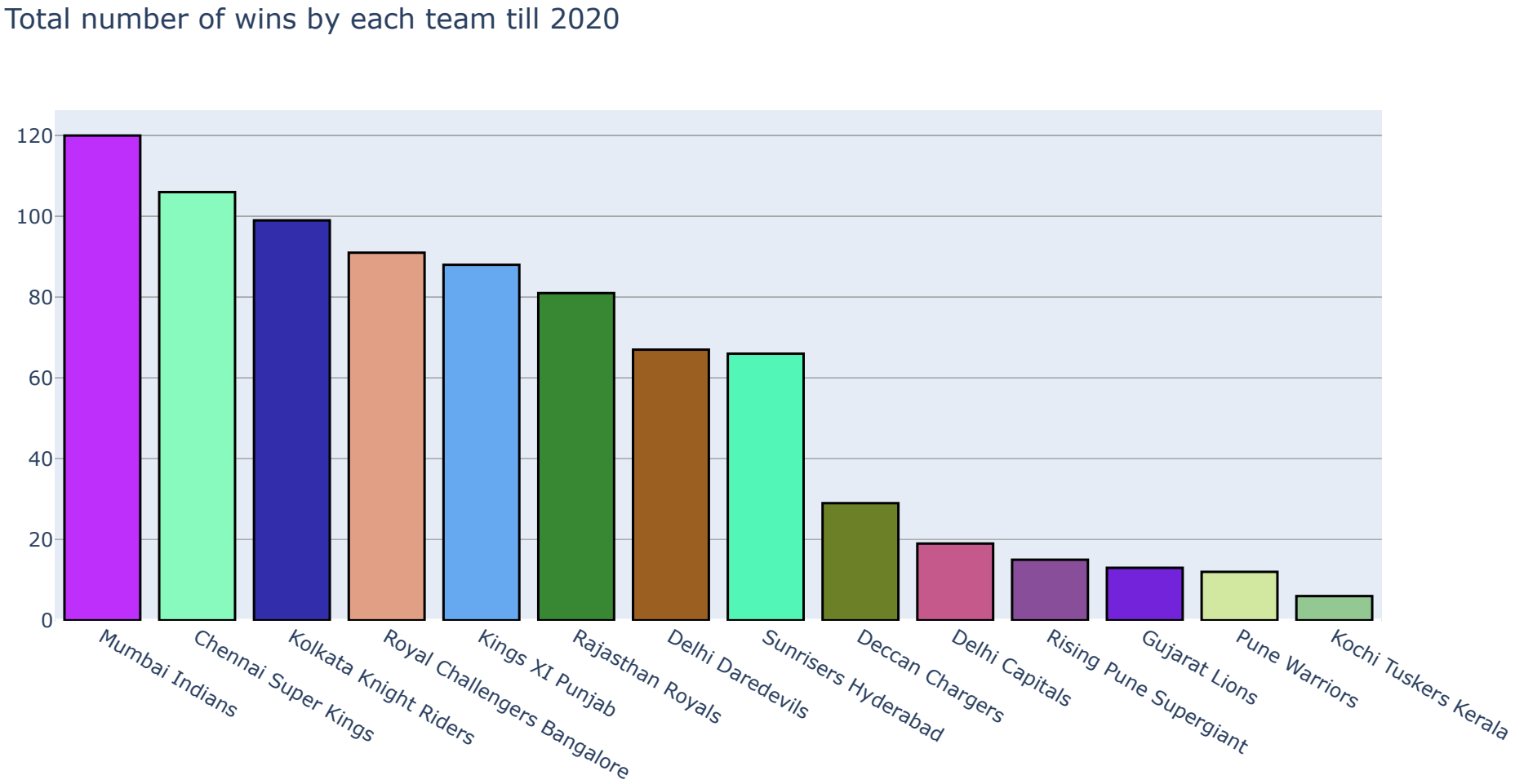
Out[109]:
```
Mumbai Indians                 120
Chennai Super Kings            106
Kolkata Knight Riders           99
Royal Challengers Bangalore     91
Kings XI Punjab                 88
Rajasthan Royals                81
Delhi Daredevils                67
Sunrisers Hyderabad             66
Deccan Chargers                 29
Delhi Capitals                  19
Rising Pune Supergiant          15
Gujarat Lions                   13
Pune Warriors                   12
Kochi Tuskers Kerala             6
Name: winner, dtype: int64
```

In [110]:
```
plt.figure(figsize=(32,18))
plt.bar(list(matches['winner'].value_counts()[0:10].keys()),list(matches['winner'].value_counts()[0:10]))
plt.show()
```



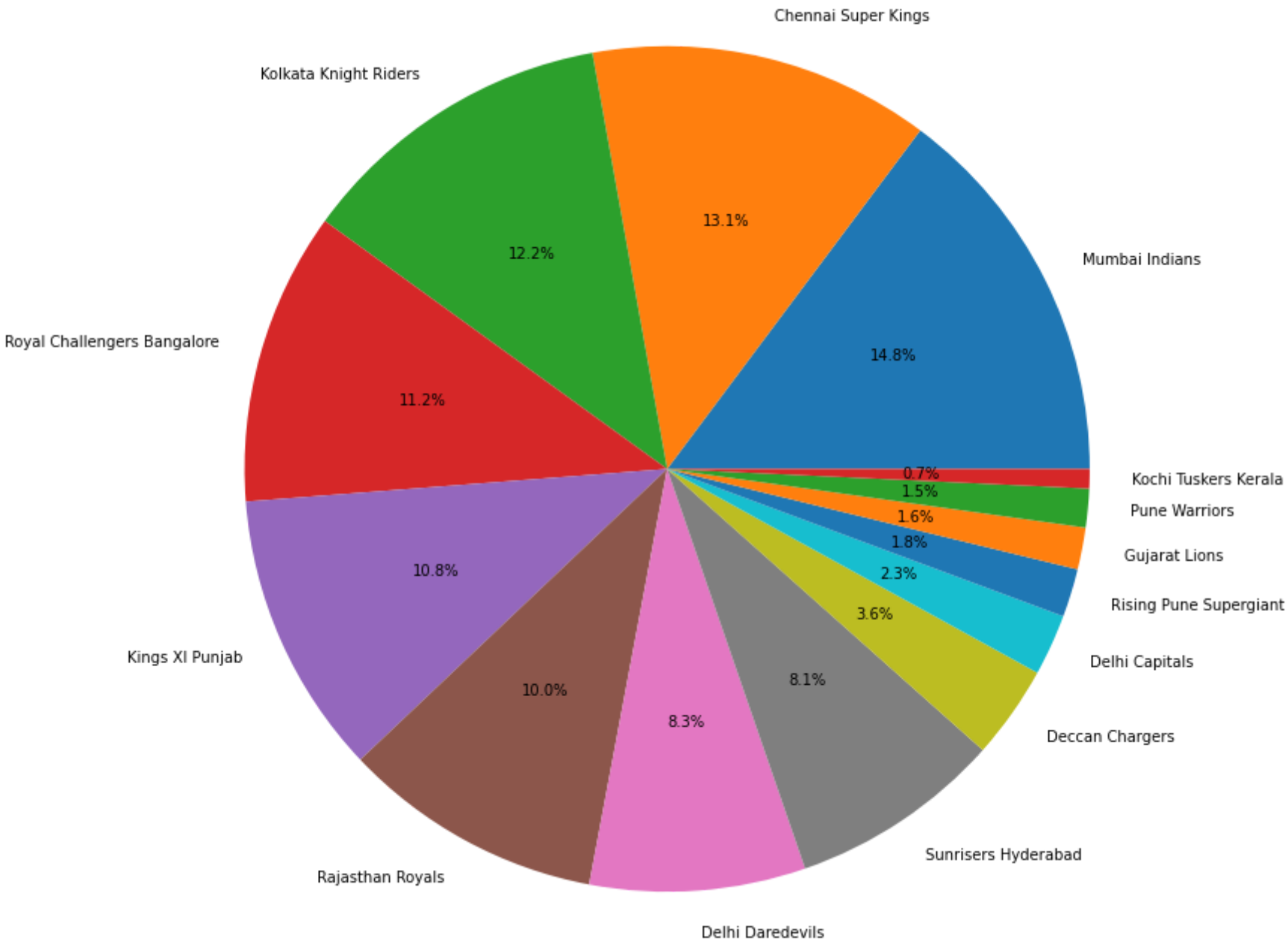
In [111]:

```
data = [go.Bar(
    x = matches['winner'].value_counts().index,
    y = matches['winner'].value_counts().values,
    marker = dict(color = random_colors(len(matches['winner'].value_counts().index)),line=dict(color='#000000', width=1.5))
)]

layout = go.Layout(
   {
      "title":"Total number of wins by each team till 2020",
   }
)

fig = go.Figure(data=data,layout = layout)
iplot(fig)
```



pie plot Total number of matches Win by Each Teams

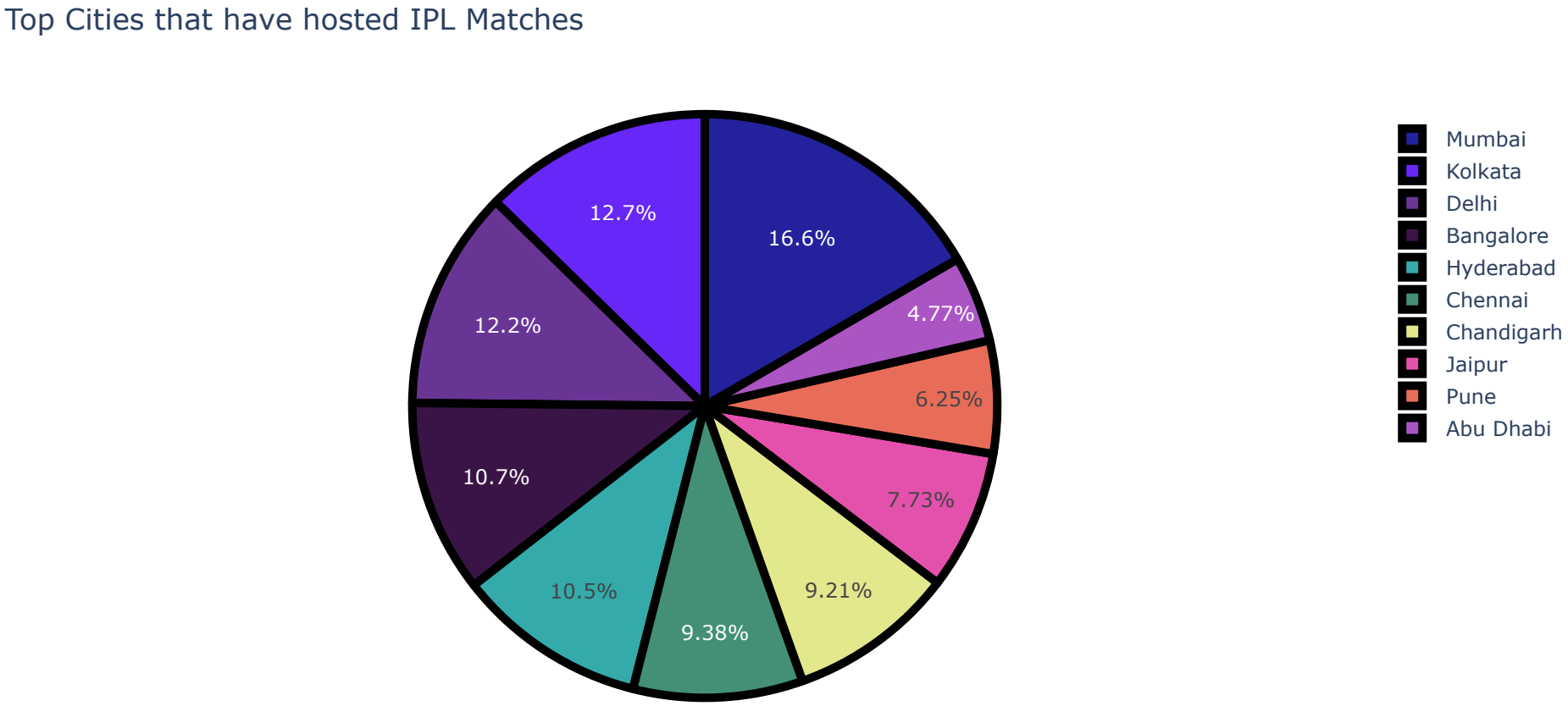
In [112]:
```
plt.figure(figsize=(13,13))
plt.pie(list(matches['winner'].value_counts()),labels=list(matches['winner'].value_counts().keys()),autopct='%0.1f%%')
plt.show()
```


Chennai Super Kings
13.1%
Mumbai Indians
14.8%
Kolkata Knight Riders
12.2%
Royal Challengers Bangalore
11.2%
Kings XI Punjab
10.8%
Rajasthan Royals
10.0%
Delhi Daredevils
8.3%
Sunrisers Hyderabad
8.1%
Deccan Chargers
3.6%
Delhi Capitals
2.3%
Rising Pune Supergiant
1.8%
Gujarat Lions
1.6%
Pune Warriors
1.5%
Kochi Tuskers Kerala
0.7%

Top Cities that have hosted IPL Matches

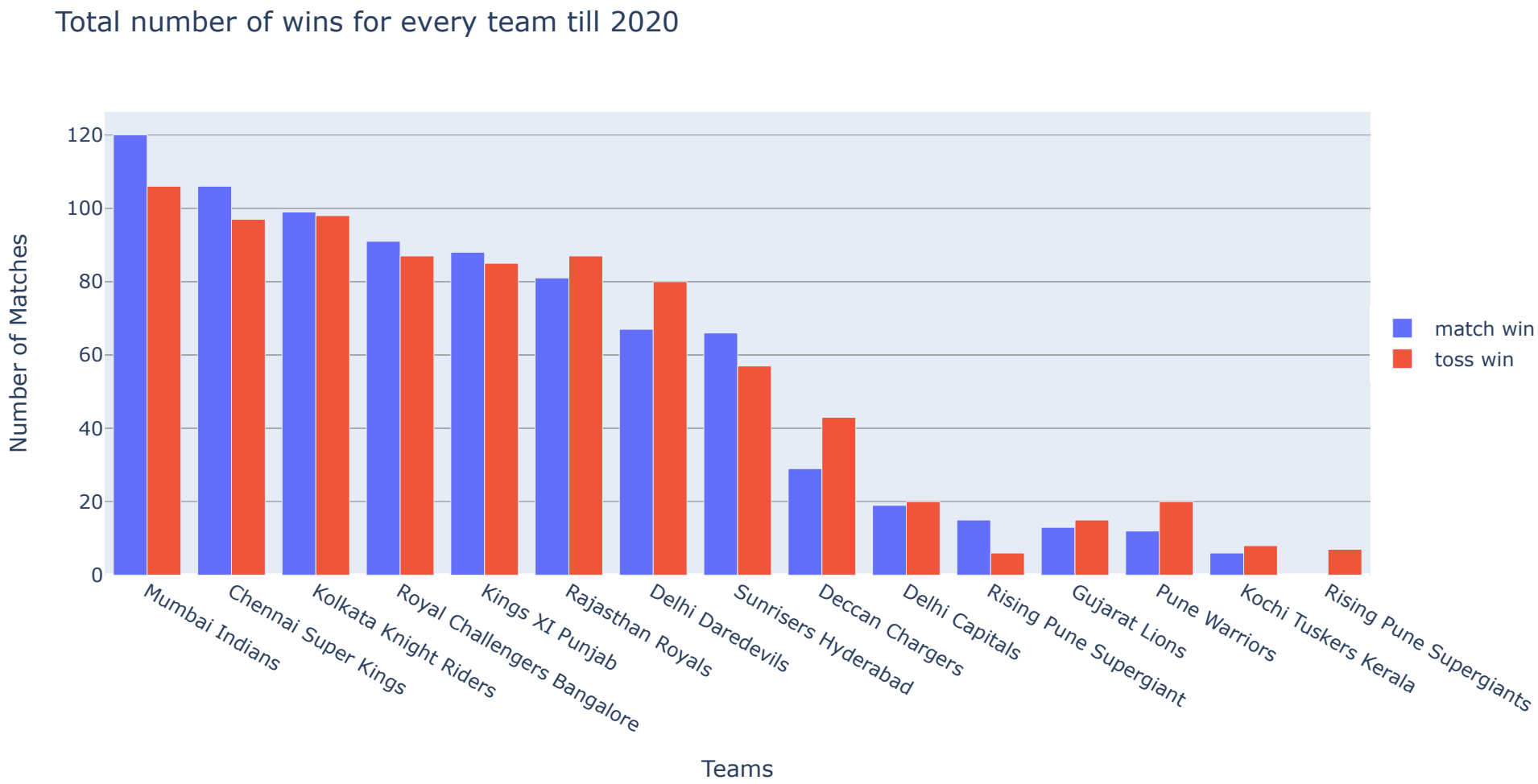
In [113]:

```
city_counts= matches.groupby('city').apply(lambda x:x['city'].count()).reset_index(name='Match Counts')
top_cities_order=city_counts.sort_values(by='Match Counts',ascending=False)
top_cities=top_cities_order[:10]

trace = go.Pie(labels = top_cities.city.values, values =top_cities["Match Counts"].values,
               marker=dict(colors = random_colors(10),
                           line=dict(color='#000000', width=5)
                           ))
data = [trace]
layout = go.Layout(
   {
      "title":"Top Cities that have hosted IPL Matches",
   }
)

fig = go.Figure(data=data,layout = layout)
iplot(fig)
```



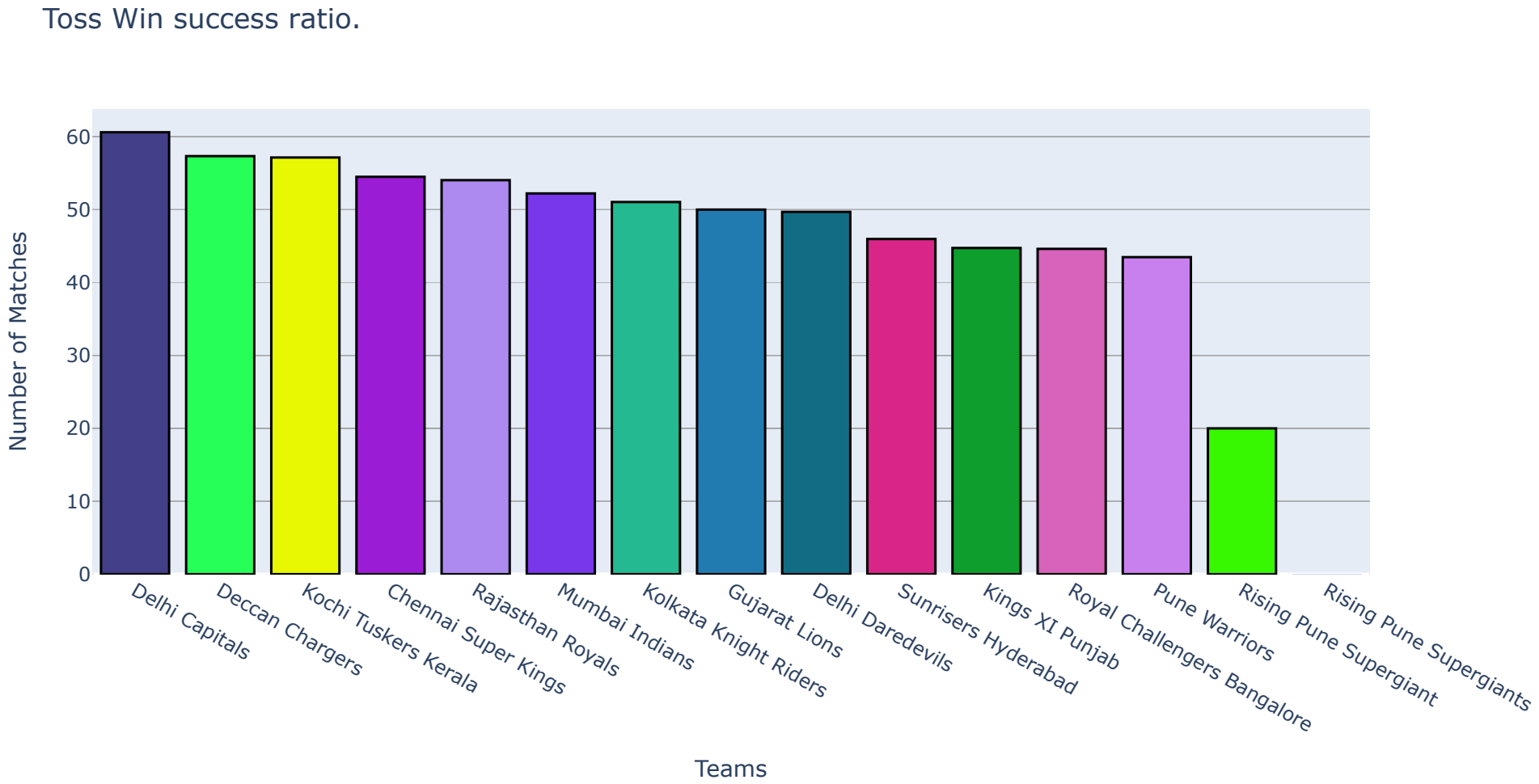
Total number of toss and match wins for every team till 2020

In [114]:
```
trace1 = go.Bar(x=matches["winner"].value_counts().index, y=matches["winner"].value_counts().values,name="match win")
trace2 = go.Bar(x=matches["toss_winner"].value_counts().index, y=matches["toss_winner"].value_counts().values,name="toss win")

# Fill out  data with our traces
data = [trace1, trace2]
# Create layout and specify title, legend and so on

layout = go.Layout(title="Total number of wins for every team till 2020",
                   xaxis=dict(title="Teams"),
                   yaxis=dict(title="Number of Matches"),
                   legend=dict(x=1.0, y=0.5)
                   ,barmode="group")


# Create figure with all prepared data for plot
fig = go.Figure(data=data, layout=layout)
# Create a plot in your Python script directory with name "bar-chart.html"
iplot(fig)
```



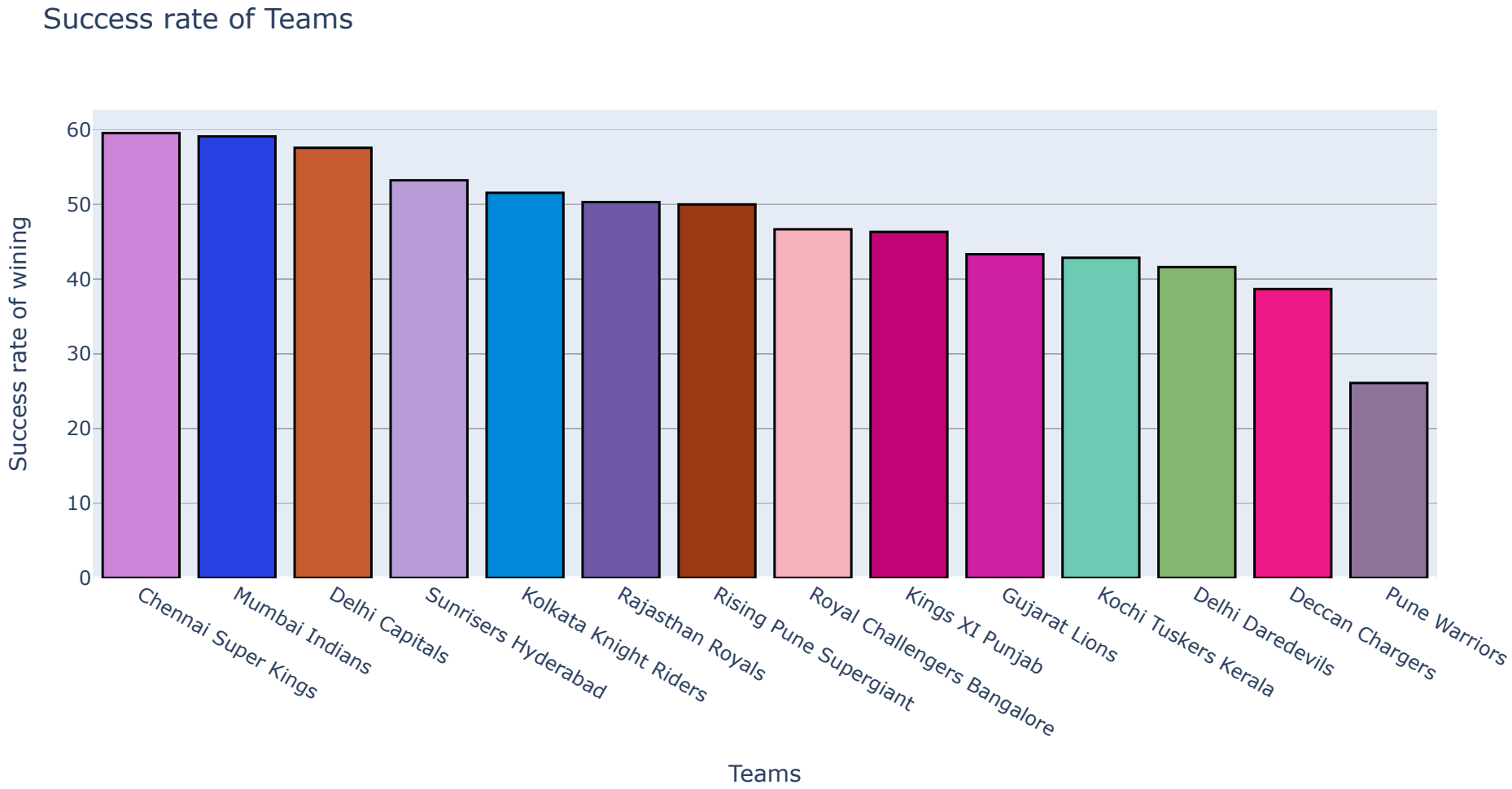
Toss Win success ratio for every team

In [115]:
```
Total_matches_played = matches['team1'].value_counts() + matches['team2'].value_counts()

toss_won = matches['toss_winner'].value_counts()
toss_win_success_rate = (toss_won/Total_matches_played)*100
toss_win_success_rate_sort = toss_win_success_rate.sort_values(ascending = False)
toss_win_success_rate_sort

data = [go.Bar(
    x = toss_win_success_rate.sort_values(ascending=False).index,
    y = toss_win_success_rate.sort_values(ascending=False).values,
    marker = dict(color = random_colors(len(toss_win_success_rate.sort_values(ascending=False).index)),line=dict(color='#00000
0', width=1.5))
)]

layout = go.Layout(title="Toss Win success ratio.",
                   xaxis=dict(title="Teams"),
                   yaxis=dict(title="Number of Matches"))

# Create figure with all prepared data for plot
fig = go.Figure(data=data, layout=layout)
# Create a plot in your Python script directory with name "bar-chart.html"
iplot(fig)
```


Toss Win success ratio.
Number of Matches
0
10
20
30
40
50
60
Delhi Capitals
Deccan Chargers
Kochi Tuskers Kerala
Chennai Super Kings
Rajasthan Royals
Mumbai Indians
Kolkata Knight Riders
Gujarat Lions
Delhi Daredevils
Sunrisers Hyderabad
Kings XI Punjab
Royal Challengers Bangalore
Pune Warriors
Rising Pune Supergiant
Rising Pune Supergiants
Teams

In [116]:

```
matches_won = matches.groupby('winner').count()
total_matches = matches['team1'].value_counts() + matches['team2'].value_counts()

matches_won['Total matches'] = total_matches
win_df = matches_won[["Total matches","result"]]
success_ratio = round((matches_won['id']/total_matches),4)*100
success_ratio_sort = success_ratio.sort_values(ascending = False)

data = [go.Bar(
    x = success_ratio_sort.index,
    y = success_ratio_sort.values,
    marker = dict(color = random_colors(len(success_ratio_sort.index)),line=dict(color='#000000', width=1.5))
)]

layout = go.Layout(title="Success rate of Teams",
                   xaxis=dict(title="Teams"),
                   yaxis=dict(title="Success rate of wining"))

# Create figure with all prepared data for plot
fig = go.Figure(data=data, layout=layout)
# Create a plot in your Python script directory with name "bar-chart.html"
iplot(fig)
```

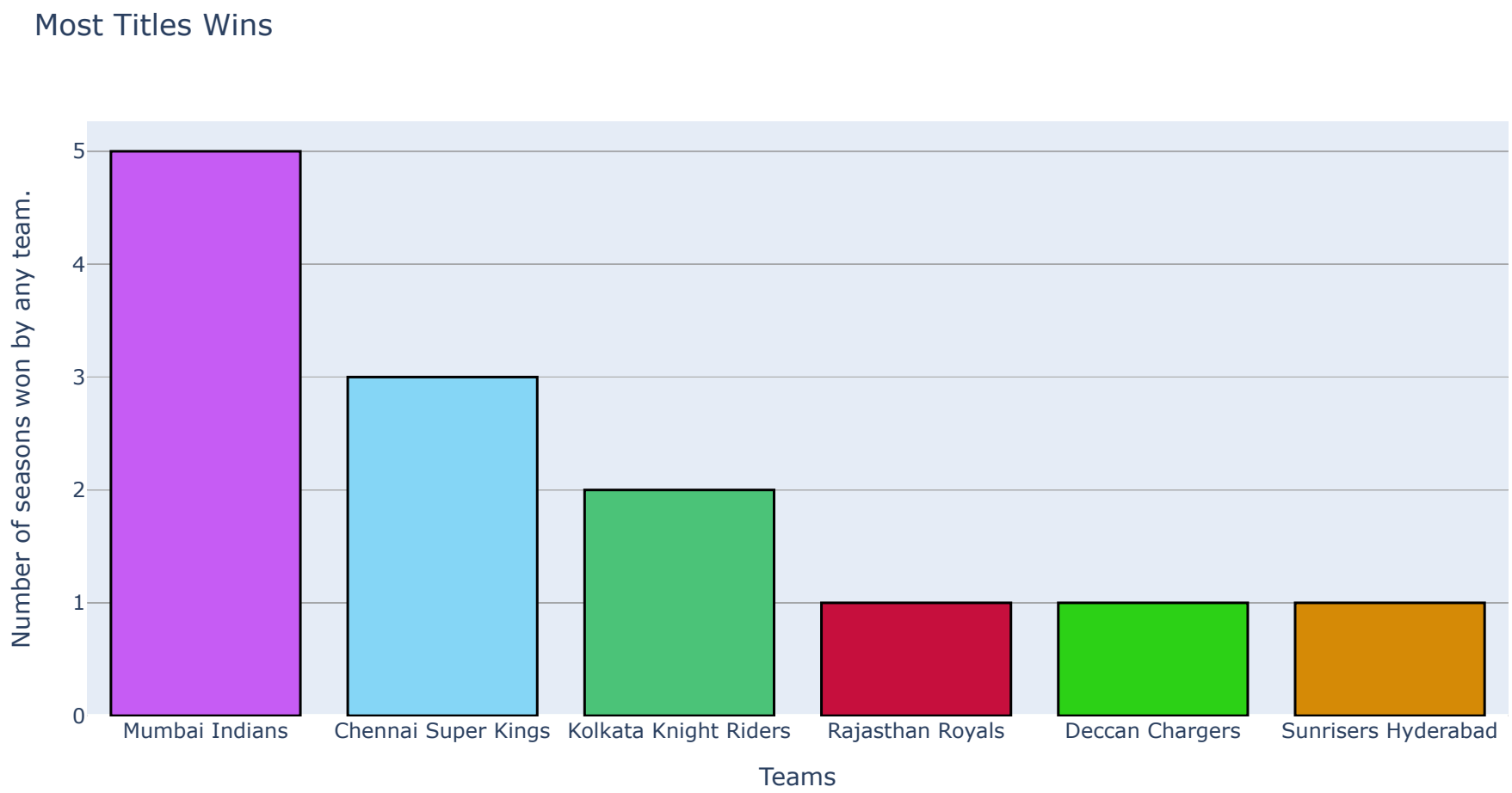


Number of seasons won by any team

In [117]:

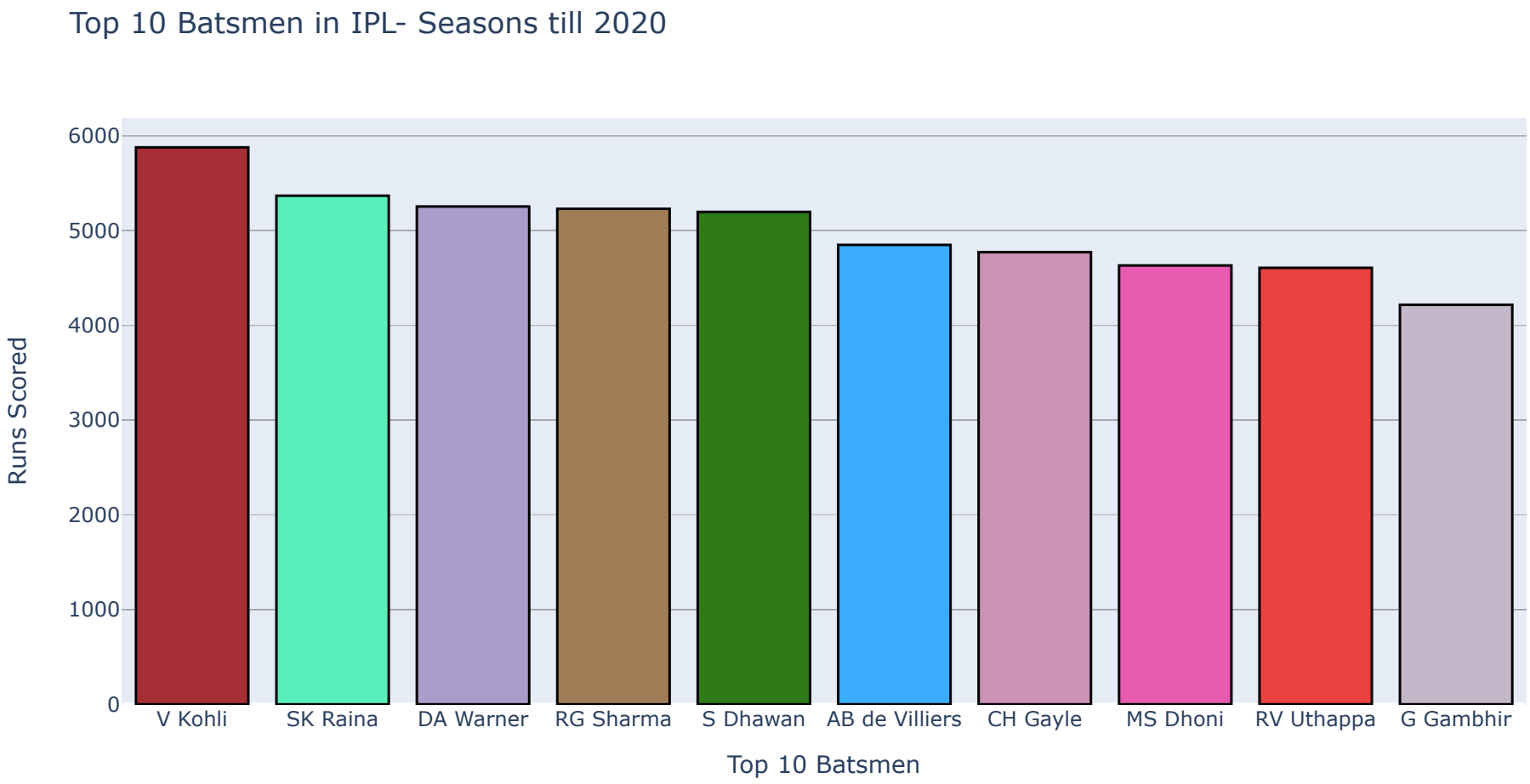
```
each_season_winner = matches.groupby('season')['season','winner'].tail(1)
each_season_winner_sort = each_season_winner.sort_values('season',ascending = True)

data = [go.Bar(
    x = each_season_winner_sort["winner"].value_counts().index,
    y = each_season_winner_sort["winner"].value_counts().values,
    marker = dict(color = random_colors(len(each_season_winner_sort["winner"].value_counts().index)),line=dict(color='#000000',
width=1.5))
)]

layout = go.Layout(title="Most Titles Wins",
                   xaxis=dict(title="Teams"),
                   yaxis=dict(title="Number of seasons won by any team."))

# Create figure with all prepared data for plot
fig = go.Figure(data=data, layout=layout)
# Create a plot in your Python script directory with name "bar-chart.html"
iplot(fig)
```

```
/home/dharmveer/venv/lib/python3.6/site-packages/ipykernel_launcher.py:1: FutureWarning:

Indexing with multiple keys (implicitly converted to a tuple of keys) will be deprecated, use a list instead.
```



Top 10 Batsman in IPL- Seasons till 2020

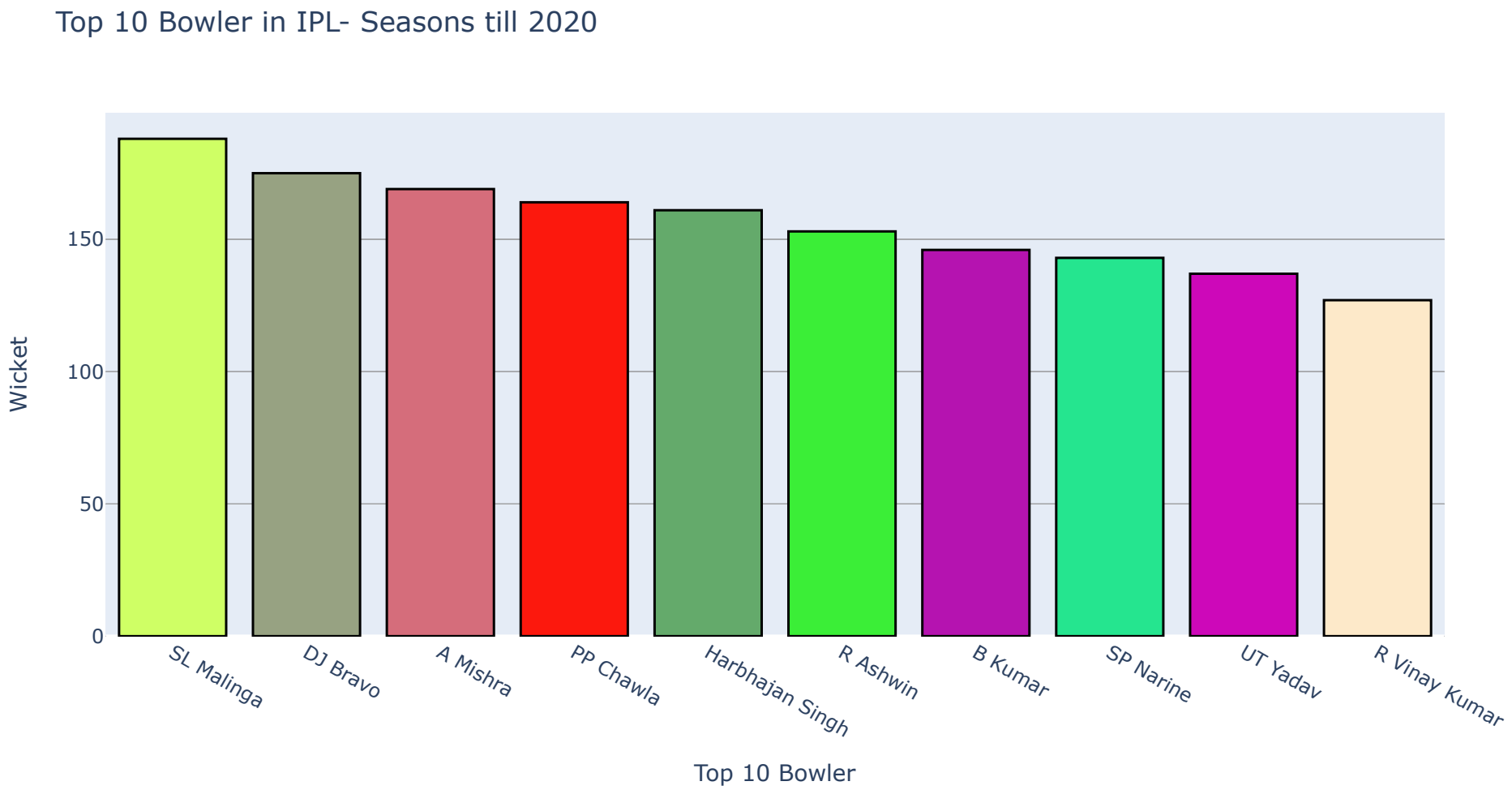
In [118]:
```
batting_tot=balls.groupby('batsman').apply(lambda x:np.sum(x['batsman_runs'])).reset_index(name='Runs')
batting_sorted=batting_tot.sort_values(by='Runs',ascending=False)
top_batsmen=batting_sorted[:10]

data = [go.Bar(
    x = top_batsmen.batsman,
    y = top_batsmen.Runs,
    marker = dict(color = random_colors(10),line=dict(color='#000000', width=1.5))
)]

layout = go.Layout(title="Top 10 Batsmen in IPL- Seasons till 2020",
                   xaxis=dict(title="Top 10 Batsmen"),
                   yaxis=dict(title="Runs Scored"))

# Create figure with all prepared data for plot
fig = go.Figure(data=data, layout=layout)
# Create a plot in your Python script directory with name "bar-chart.html"
iplot(fig)
```



Top 10 Bowler in IPL- Seasons till 2020

In [119]:

```
bowling_tot=balls.groupby('bowler').apply(lambda x:np.sum(x['is_wicket'])).reset_index(name='wicket')
bowling_sorted=bowling_tot.sort_values(by='wicket',ascending=False)
top_bowler=bowling_sorted[:10]

data = [go.Bar(
    x = top_bowler.bowler,
    y = top_bowler.wicket,
    marker = dict(color = random_colors(10),line=dict(color='#000000', width=1.5))
)]

layout = go.Layout(title="Top 10 Bowler in IPL- Seasons till 2020",
                   xaxis=dict(title="Top 10 Bowler"),
                   yaxis=dict(title="Wicket"))

# Create figure with all prepared data for plot
fig = go.Figure(data=data, layout=layout)
# Create a plot in your Python script directory with name "bar-chart.html"
iplot(fig)
```



Merging the two datasets

In [120]:
```
data = pd.merge(left=matches, right=balls, on='id', how='right')
data.head()
```

Out[120]:

	id	city	date	player_of_match	venue	neutral_venue	team1	team2	toss_winner	toss_decision	...	extra_runs	total_runs	non_b
0	335982	Bangalore	2008-04-18	BB McCullum	M. Chinnaswamy Stadium	0	Royal Challengers Bangalore	Kolkata Knight Riders	Royal Challengers Bangalore	field	...	0	1	
1	335982	Bangalore	2008-04-18	BB McCullum	M. Chinnaswamy Stadium	0	Royal Challengers Bangalore	Kolkata Knight Riders	Royal Challengers Bangalore	field	...	0	1	
2	335982	Bangalore	2008-04-18	BB McCullum	M. Chinnaswamy Stadium	0	Royal Challengers Bangalore	Kolkata Knight Riders	Royal Challengers Bangalore	field	...	0	0	
3	335982	Bangalore	2008-04-18	BB McCullum	M. Chinnaswamy Stadium	0	Royal Challengers Bangalore	Kolkata Knight Riders	Royal Challengers Bangalore	field	...	0	1	
4	335982	Bangalore	2008-04-18	BB McCullum	M. Chinnaswamy Stadium	0	Royal Challengers Bangalore	Kolkata Knight Riders	Royal Challengers Bangalore	field	...	0	1	

5 rows × 35 columns

In [121]:
```
print(matches.shape)
print(balls.shape)
print(data.shape)
```

```
(816, 18)
(193468, 18)
(193468, 35)
```

In [122]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 193468 entries, 0 to 193467
Data columns (total 35 columns):
 #   Column            Non-Null Count   Dtype  
---  ------            --------------   -----  
 0   id                193468 non-null  int64  
 1   city              190329 non-null  object 
 2   date              193468 non-null  object 
 3   player_of_match   193096 non-null  object 
 4   venue             193468 non-null  object 
 5   neutral_venue     193468 non-null  int64  
 6   team1             193468 non-null  object 
 7   team2             193468 non-null  object 
 8   toss_winner       193468 non-null  object 
 9   toss_decision     193468 non-null  object 
 10  winner            193096 non-null  object 
 11  result            193096 non-null  object 
 12  result_margin     189871 non-null  float64
 13  eliminator        193096 non-null  object 
 14  method            3208 non-null    object 
 15  umpire1           193468 non-null  object 
 16  umpire2           193468 non-null  object 
 17  season            193468 non-null  object 
 18  inning            193468 non-null  int64  
 19  over              193468 non-null  int64  
 20  ball              193468 non-null  int64  
 21  batsman           193468 non-null  object 
 22  non_striker       193468 non-null  object 
 23  bowler            193468 non-null  object 
 24  batsman_runs      193468 non-null  int64  
 25  extra_runs        193468 non-null  int64  
 26  total_runs        193468 non-null  int64  
 27  non_boundary      193468 non-null  int64  
 28  is_wicket         193468 non-null  int64  
 29  dismissal_kind    9495 non-null    object 
 30  player_dismissed  9495 non-null    object 
 31  fielder           6784 non-null    object 
 32  extras_type       10233 non-null   object 
 33  batting_team      193468 non-null  object 
 34  bowling_team      193277 non-null  object 
dtypes: float64(1), int64(10), object(24)
memory usage: 53.1+ MB
```

In [123]:
```
data.describe()
```

Out[123]:

	id	neutral_venue	result_margin	inning	over	ball	batsman_runs	extra_runs	total_runs	non_boundary	is_wicket
count	193,468	193,468	189,871	193,468	193,468	193,468	193,468	193,468	193,468	193,468	193,468
mean	756,769	0	17	1	9	4	1	0	1	0	0
std	306,097	0	22	0	6	2	2	0	2	0	0
min	335,982	0	1	1	0	1	0	0	0	0	0
25%	501,227	0	6	1	4	2	0	0	0	0	0
50%	729,297	0	8	1	9	4	1	0	1	0	0
75%	1,082,628	0	20	2	14	5	1	0	1	0	0
max	1,237,181	1	146	2	19	9	6	7	7	1	1

In [124]:
```
data['season'].unique()
```

Out[124]:
```
array(['2008', '2009', '2010', '2011', '2012', '2013', '2014', '2015',
       '2016', '2017', '2018', '2019', '2020'], dtype=object)
```

Extracting year from the date

In [125]:
```
data['date'] = pd.to_datetime(data['date'])
data['year'] = pd.DatetimeIndex(data['date']).year
```

In [126]:
```
data.head()
```

Out[126]:

	id	city	date	player_of_match	venue	neutral_venue	team1	team2	toss_winner	toss_decision	...	total_runs	non_boundary	is_
0	335982	Bangalore	2008-04-18	BB McCullum	M. Chinnaswamy Stadium	0	Royal Challengers Bangalore	Kolkata Knight Riders	Royal Challengers Bangalore	field	...	1	0	
1	335982	Bangalore	2008-04-18	BB McCullum	M. Chinnaswamy Stadium	0	Royal Challengers Bangalore	Kolkata Knight Riders	Royal Challengers Bangalore	field	...	1	0	
2	335982	Bangalore	2008-04-18	BB McCullum	M. Chinnaswamy Stadium	0	Royal Challengers Bangalore	Kolkata Knight Riders	Royal Challengers Bangalore	field	...	0	0	
3	335982	Bangalore	2008-04-18	BB McCullum	M. Chinnaswamy Stadium	0	Royal Challengers Bangalore	Kolkata Knight Riders	Royal Challengers Bangalore	field	...	1	0	
4	335982	Bangalore	2008-04-18	BB McCullum	M. Chinnaswamy Stadium	0	Royal Challengers Bangalore	Kolkata Knight Riders	Royal Challengers Bangalore	field	...	1	0	

5 rows × 36 columns

In [127]:
```
runs_by_years = data.groupby(by='year').sum()['total_runs']
runs_by_years = pd.DataFrame(runs_by_years)
runs_by_years.reset_index(inplace=True)
runs_by_years
```

Out[127]:

	year	total_runs
0	2008	17937
1	2009	16320
2	2010	18864
3	2011	21154
4	2012	22453
5	2013	22541
6	2014	18909
7	2015	18332
8	2016	18862
9	2017	18769
10	2018	19901
11	2019	19400
12	2020	19352

wicket taken over the year

In [128]:
```
wicket_by_years = data.groupby(by='year').sum()['is_wicket']
wicket_by_years = pd.DataFrame(wicket_by_years)
wicket_by_years.reset_index(inplace=True)
```

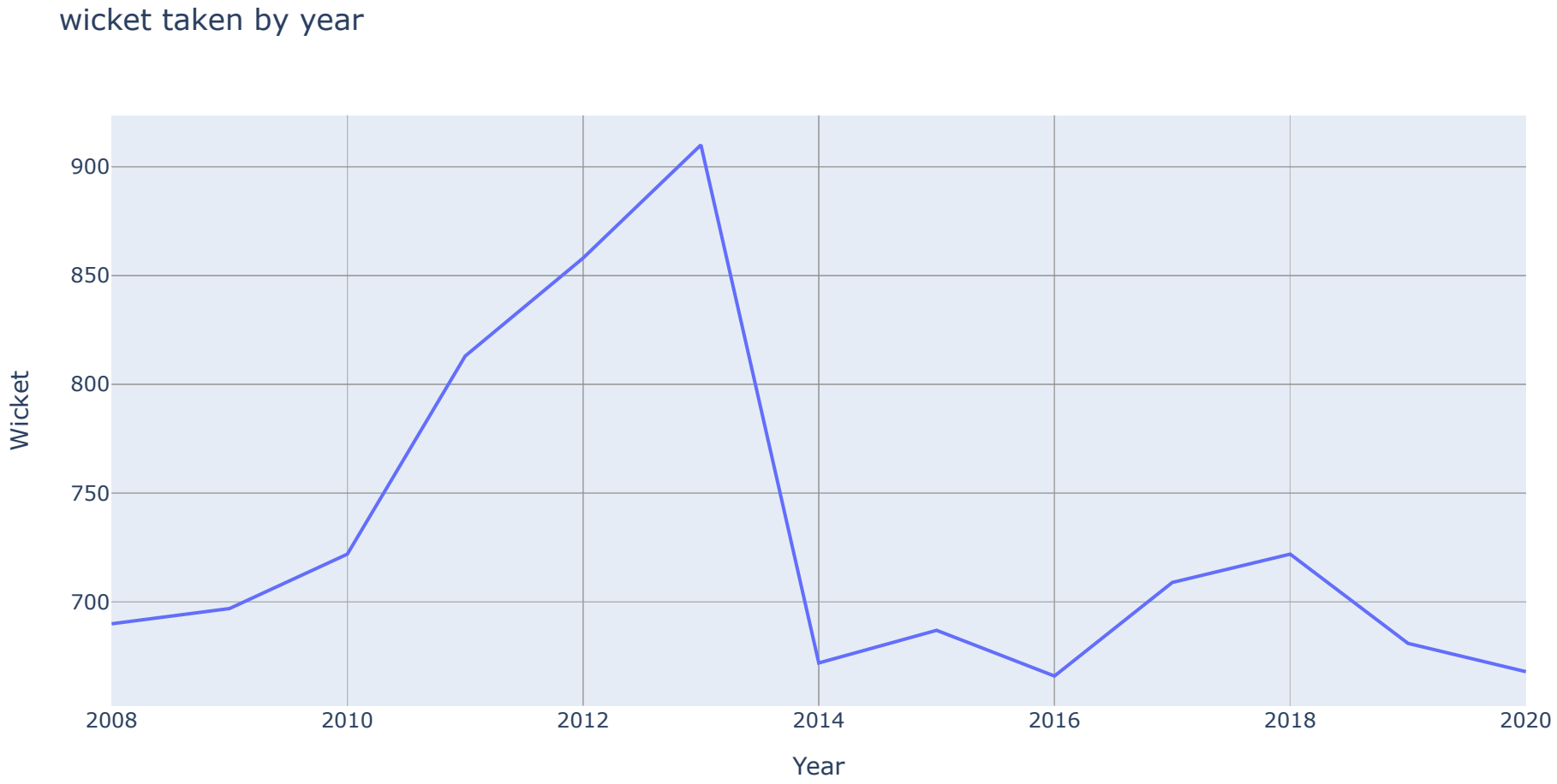
In [129]:
```
wicket_by_years
```

Out[129]:

	year	is_wicket
0	2008	690
1	2009	697
2	2010	722
3	2011	813
4	2012	858
5	2013	910
6	2014	672
7	2015	687
8	2016	666
9	2017	709
10	2018	722
11	2019	681
12	2020	668

In [130]:

```
total_wicket = go.Scatter(
                    x=wicket_by_years['year'],
                    y=wicket_by_years['is_wicket'],
                    mode='lines',
                    name='wickets')

data = [total_wicket]

layout = go.Layout(title='wicket taken by year',
                  xaxis = dict(title='Year'),
                  yaxis = dict(title='Wicket'))

fig = go.Figure(data=data, layout=layout)

pyo.iplot(fig)
```



Runs scored over the years

In [131]:
```
runs_by_years
```

Out[131]:

	year	total_runs
0	2008	17937
1	2009	16320
2	2010	18864
3	2011	21154
4	2012	22453
5	2013	22541
6	2014	18909
7	2015	18332
8	2016	18862
9	2017	18769
10	2018	19901
11	2019	19400
12	2020	19352

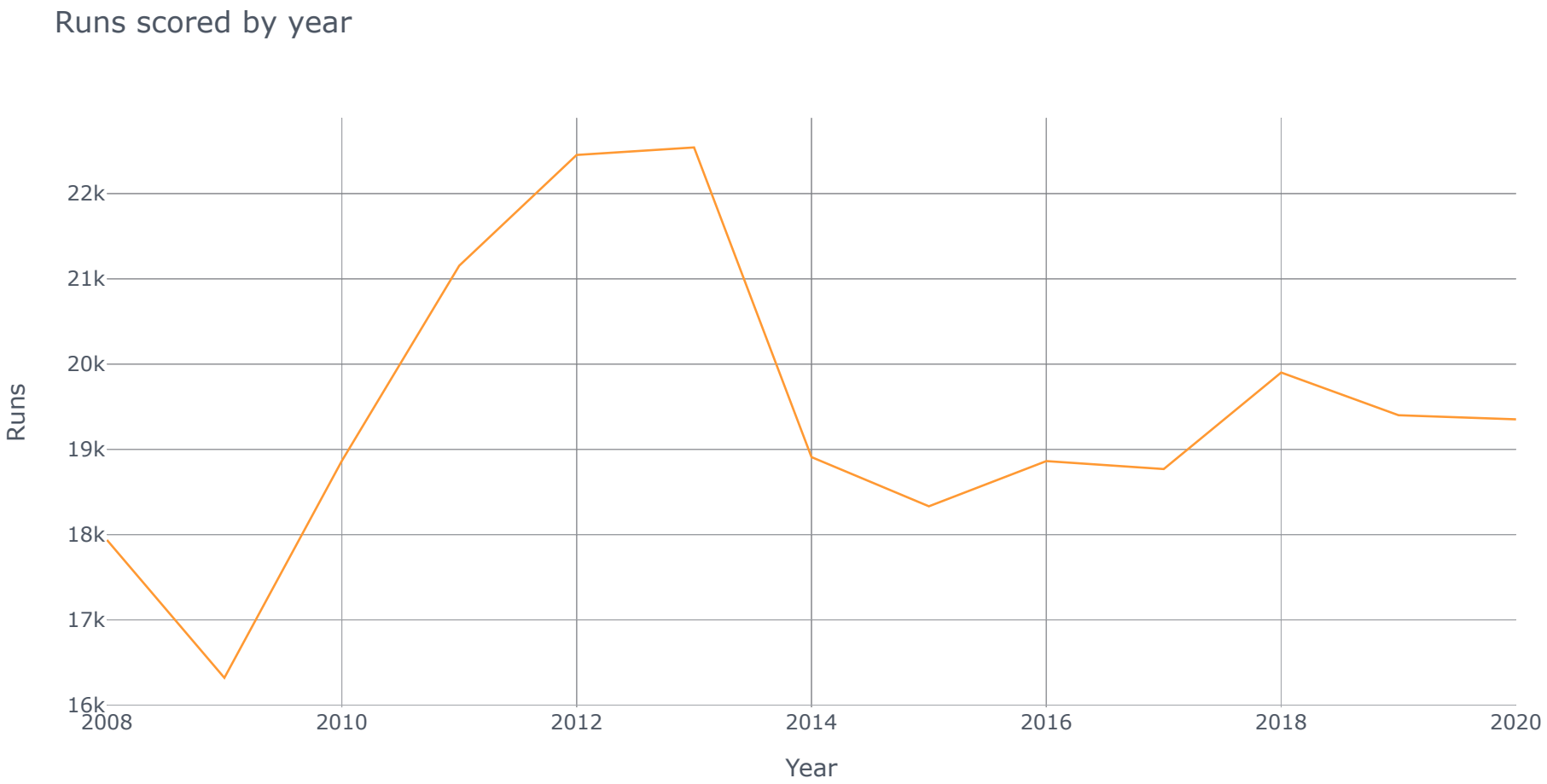
Using plotly

In [132]:

```
total_runs = go.Scatter(
                    x=runs_by_years['year'],
                    y=runs_by_years['total_runs'],
                    mode='lines',
                    name='runs')

data = [total_runs]

layout = go.Layout(title='Runs scored by year',
                  xaxis = dict(title='Year'),
                  yaxis = dict(title='Runs'))

fig = go.Figure(data=data, layout=layout)

pyo.iplot(fig)
```

Using cufflinks

In [133]:
```
runs_by_years.iplot(kind='scatter', x='year', y='total_runs', title='Runs scored by year', xTitle='Year', yTitle='Runs')
```



Export to plot.ly »

Preferred toss decision

In [134]:
```
toss_decisions = matches.groupby(by='toss_decision').count()
toss_decisions = pd.DataFrame(toss_decisions['id'])
toss_decisions.reset_index(inplace=True)
toss_decisions
```

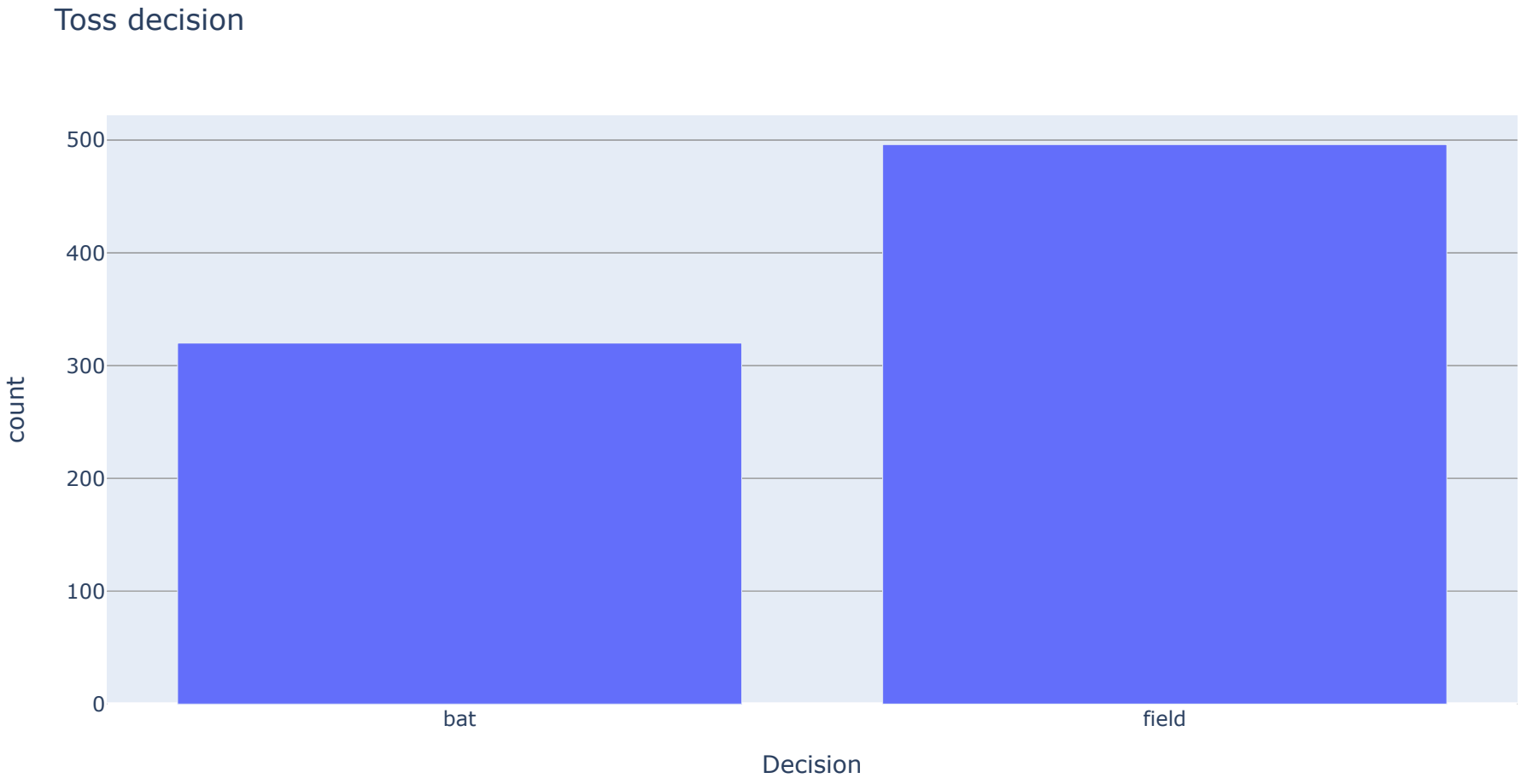
Out[134]:

	toss_decision	id
0	bat	320
1	field	496

Using plotly

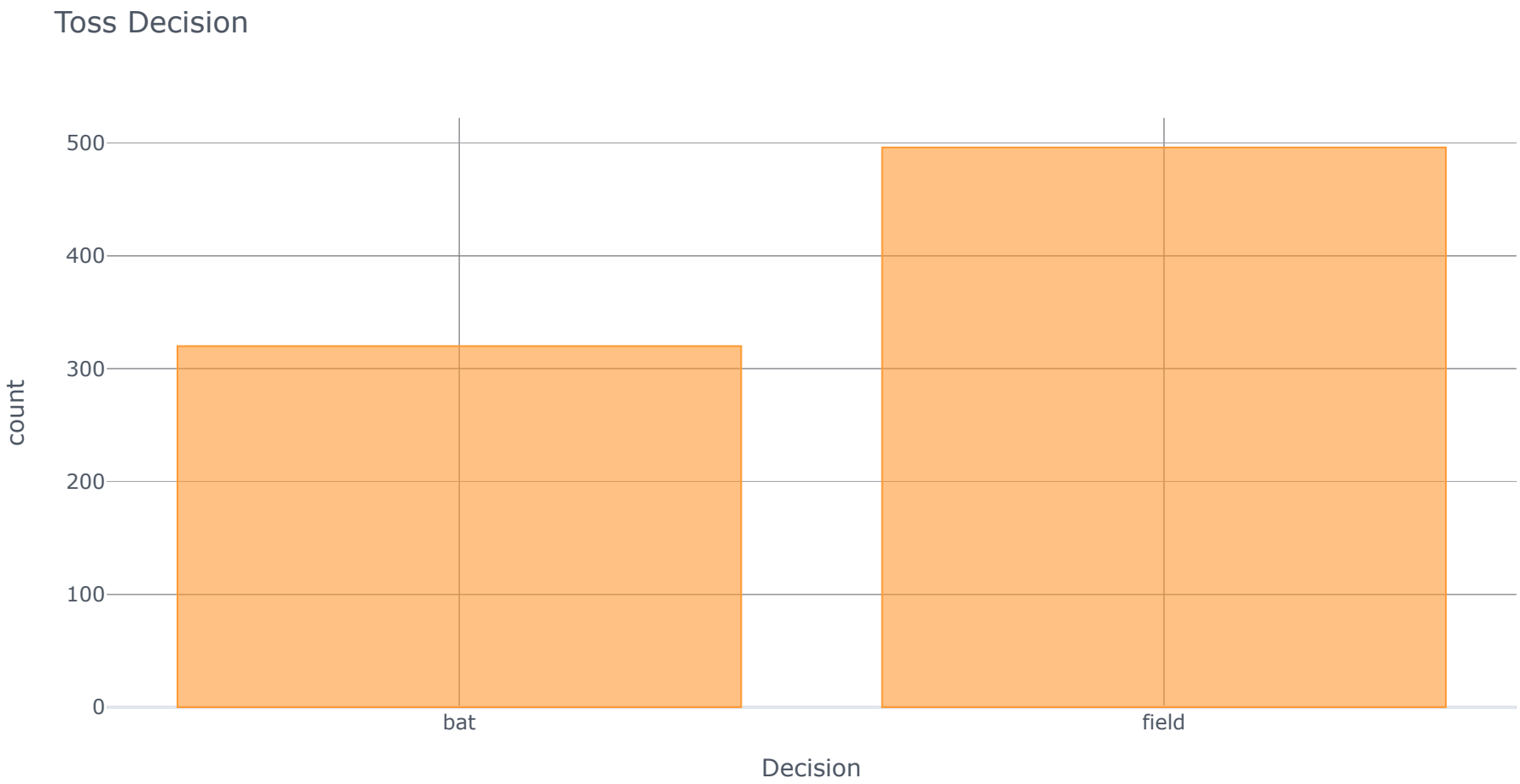
In [135]:

```
toss_decision = go.Bar(
                    x=toss_decisions['toss_decision'],
                    y=toss_decisions['id']
                )

data = [toss_decision]

layout = go.Layout(title='Toss decision',
                  xaxis = dict(title='Decision'),
                  yaxis = dict(title='count'))

fig = go.Figure(data=data, layout=layout)

pyo.iplot(fig)
```



Using cufflinks

In [136]:
```
toss_decisions.iplot(kind='bar', x='toss_decision', y='id', title='Toss Decision', xTitle='Decision', yTitle='count')
```



Export to plot.ly »

Totals runs and wickets by over

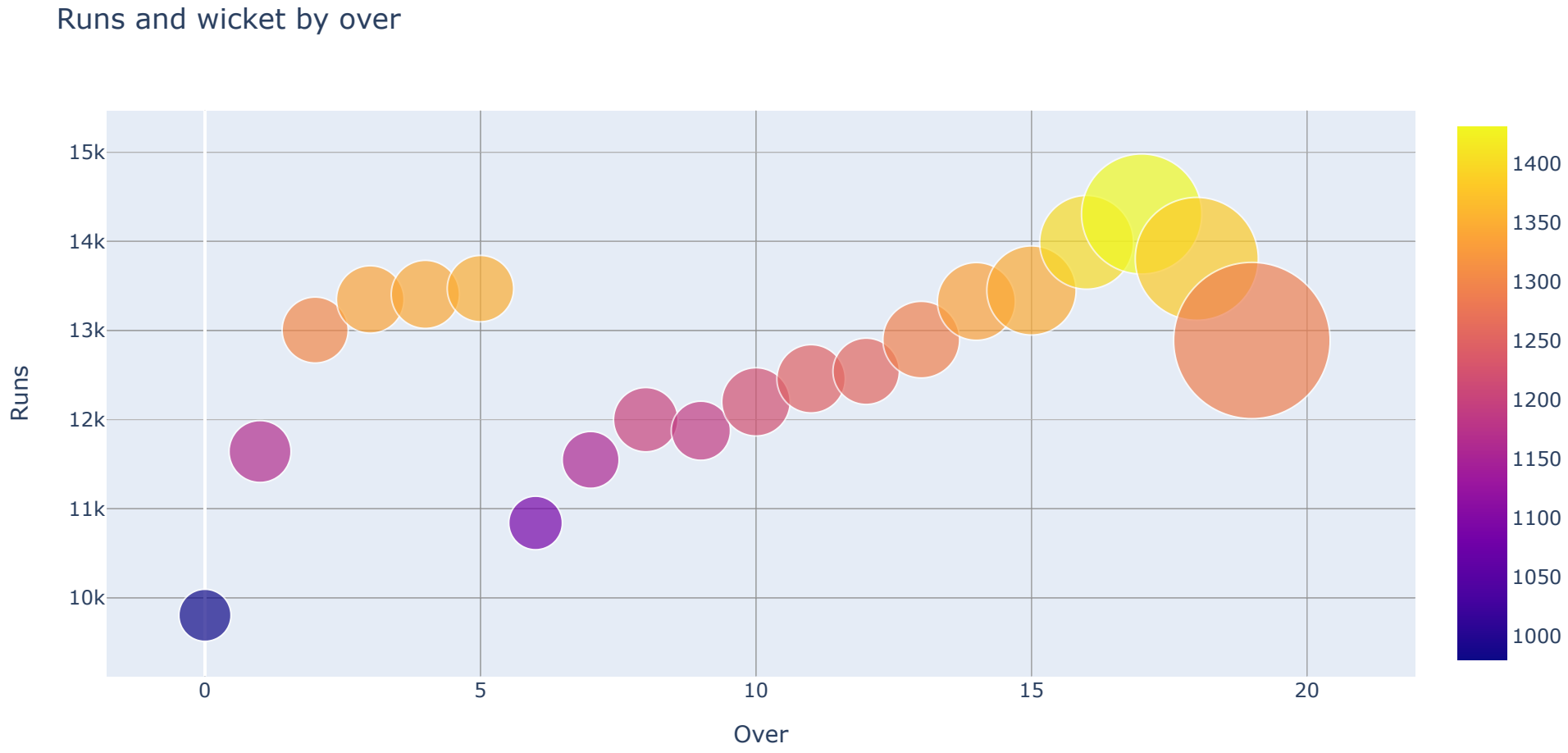
In [137]:
```
runs_and_wickets_by_over = balls.groupby(by='over').sum()
runs_and_wickets_by_over = pd.DataFrame(runs_and_wickets_by_over[['total_runs', 'is_wicket']])
runs_and_wickets_by_over.reset_index(inplace=True)
runs_and_wickets_by_over
```

Out[137]:

	over	total_runs	is_wicket
0	0	9804	318
1	1	11642	378
2	2	13005	403
3	3	13348	412
4	4	13405	416
5	5	13470	405
6	6	10840	327
7	7	11548	347
8	8	11999	393
9	9	11875	361
10	10	12199	417
11	11	12458	417
12	12	12542	405
13	13	12897	468
14	14	13326	475
15	15	13449	544
16	16	13989	572
17	17	14307	733
18	18	13804	750
19	19	12887	954

Using plotly

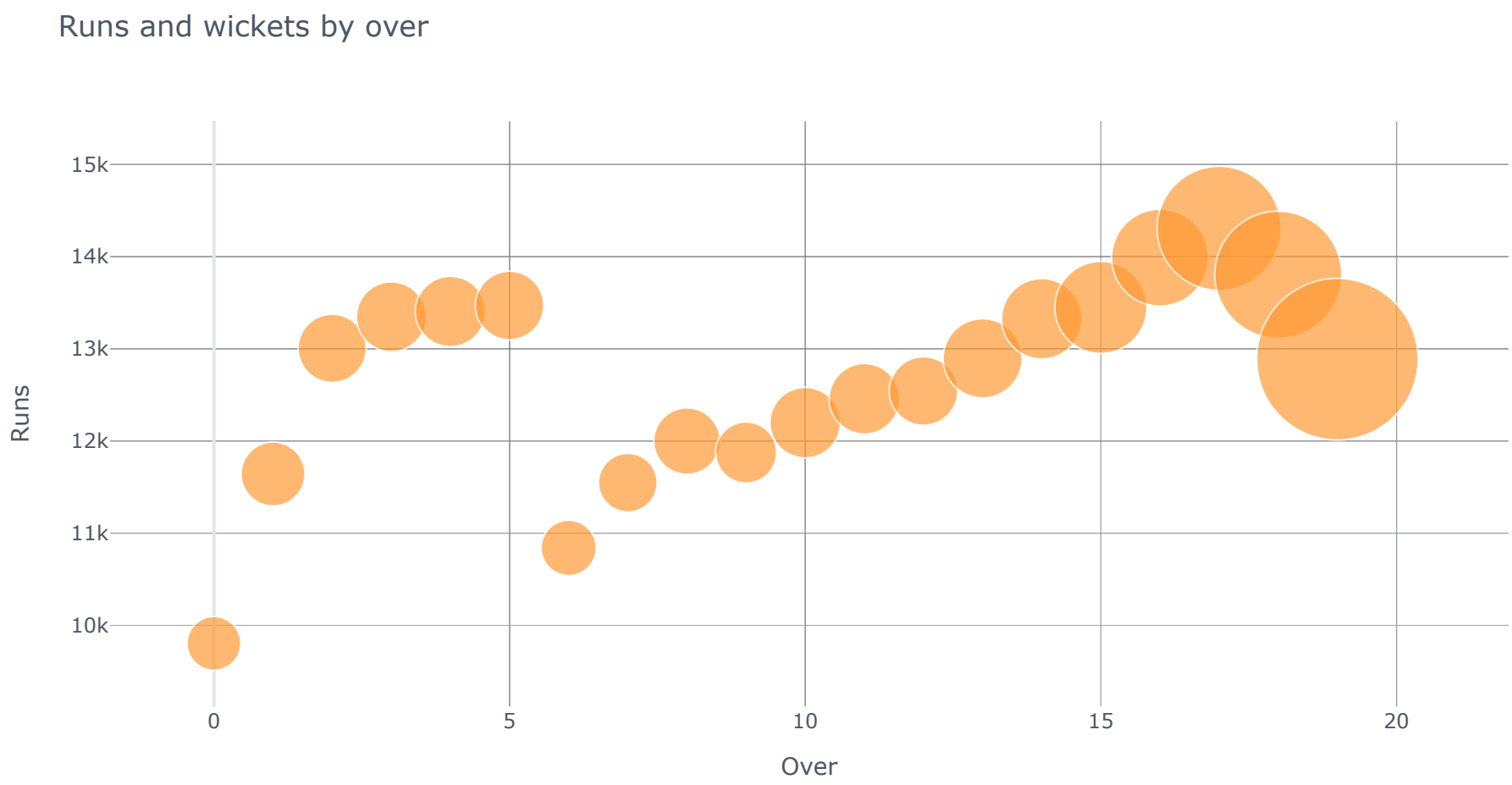
In [138]:
```
runs_and_wickets_by_overs = go.Scatter(
                    x=runs_and_wickets_by_over['over'],
                    y=runs_and_wickets_by_over['total_runs'],
                    text=runs_and_wickets_by_over['is_wicket'],
                    mode='markers',
                    marker=dict(size=runs_and_wickets_by_over['is_wicket']/10,
                                color=runs_and_wickets_by_over['total_runs']/10,
                                showscale=True)
                )

data = [runs_and_wickets_by_overs]

layout = go.Layout(title='Runs and wicket by over',
                  xaxis = dict(title='Over'),
                  yaxis = dict(title='Runs'))

fig = go.Figure(data=data, layout=layout)

pyo.iplot(fig)
```



Using cufflinks

Failed to get colorscale in cufflinks

In [139]:
```
runs_and_wickets_by_over.iplot(kind='scatter', x='over', y='total_runs', mode='markers',
                               title='Runs and wickets by over',
                               xTitle='Over', yTitle='Runs', size=runs_and_wickets_by_over['is_wicket']/10)
```



Export to plot.ly »

Runs distribution over wise

In [140]:
```
balls = pd.read_csv('IPL Ball-by-Ball 2008-2020.csv')
runs_overs = balls[['total_runs', 'over']]
runs_overs
```

Out[140]:

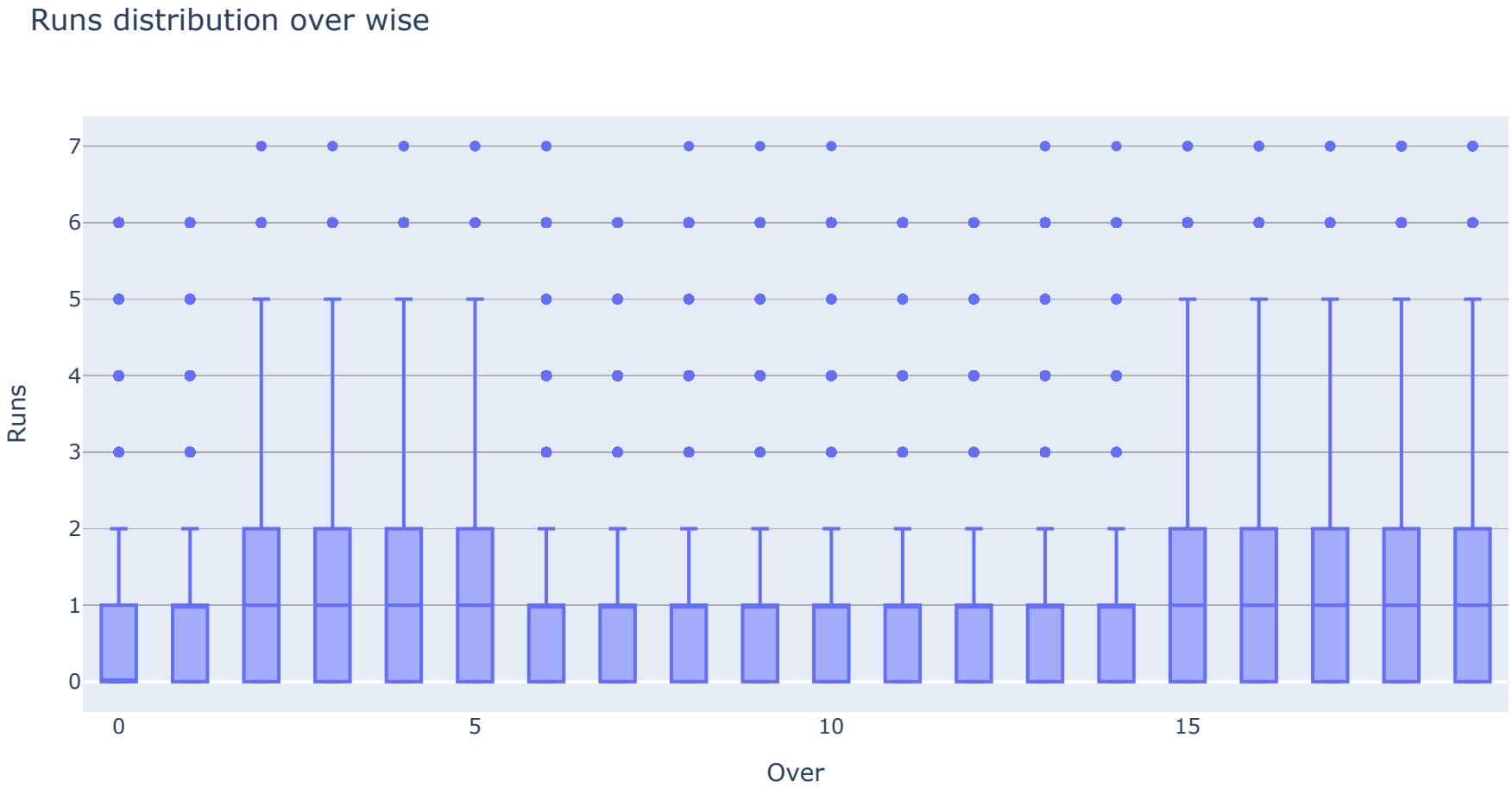
	total_runs	over
0	1	6
1	1	6
2	0	7
3	1	7
4	1	7
...	...	...
193463	0	12
193464	1	12
193465	1	13
193466	1	13
193467	1	13

193468 rows × 2 columns

Using plotly

In [141]:

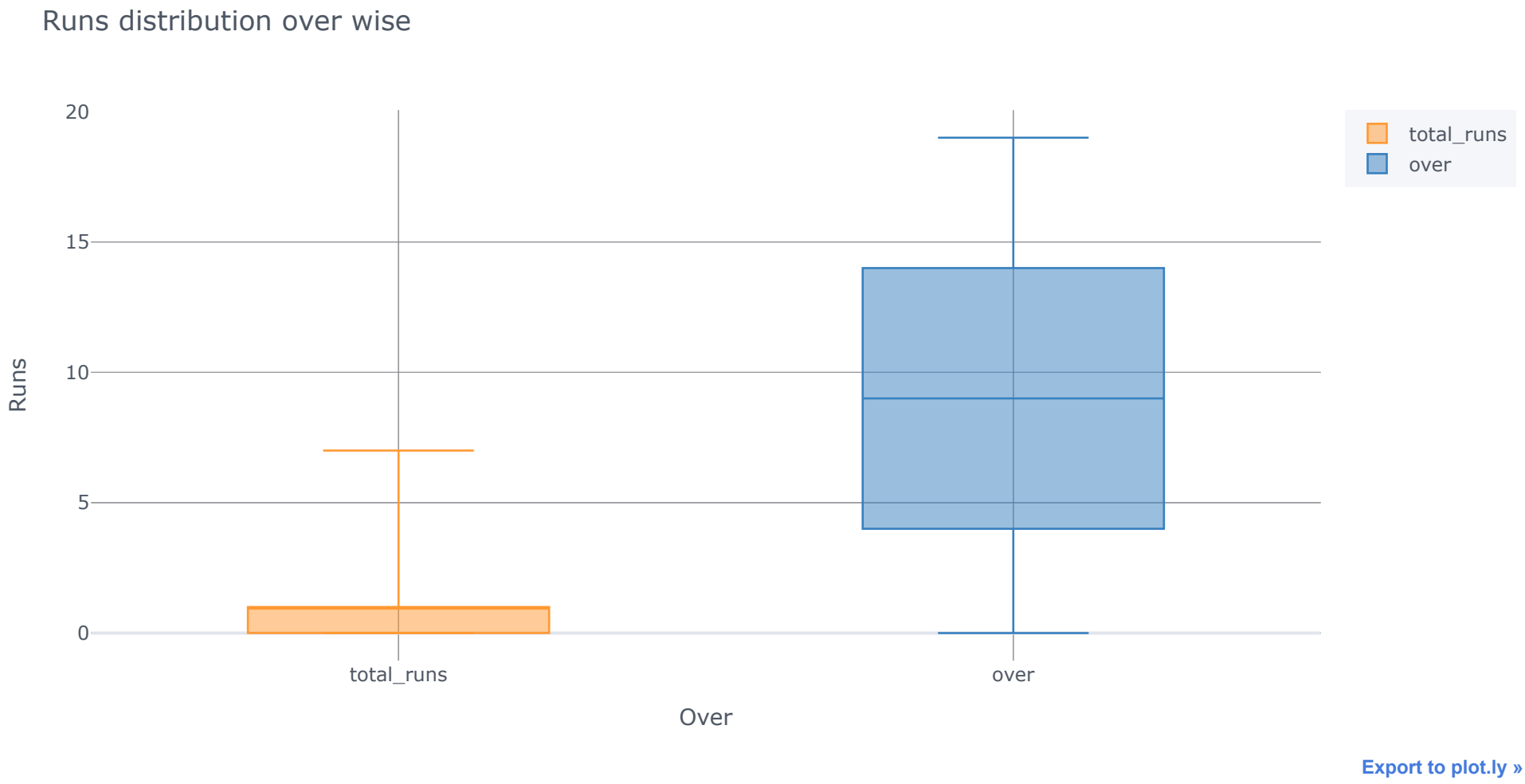
```
runs_over = go.Box(
                    x=runs_overs['over'],
                    y=runs_overs['total_runs']
                )

data = [runs_over]

layout = go.Layout(title='Runs distribution over wise',
                  xaxis = dict(title='Over'),
                  yaxis = dict(title='Runs'))

fig = go.Figure(data=data, layout=layout)

pyo.iplot(fig)
```



Using cufflinks

A simple one-liner doesn't seem to work here. See the last plot which is correct.

In [142]:
```
runs_overs.iplot(kind='box', y='over',title='Runs distribution over wise', xTitle='Over', yTitle='Runs')
```



Runs distribution match wise

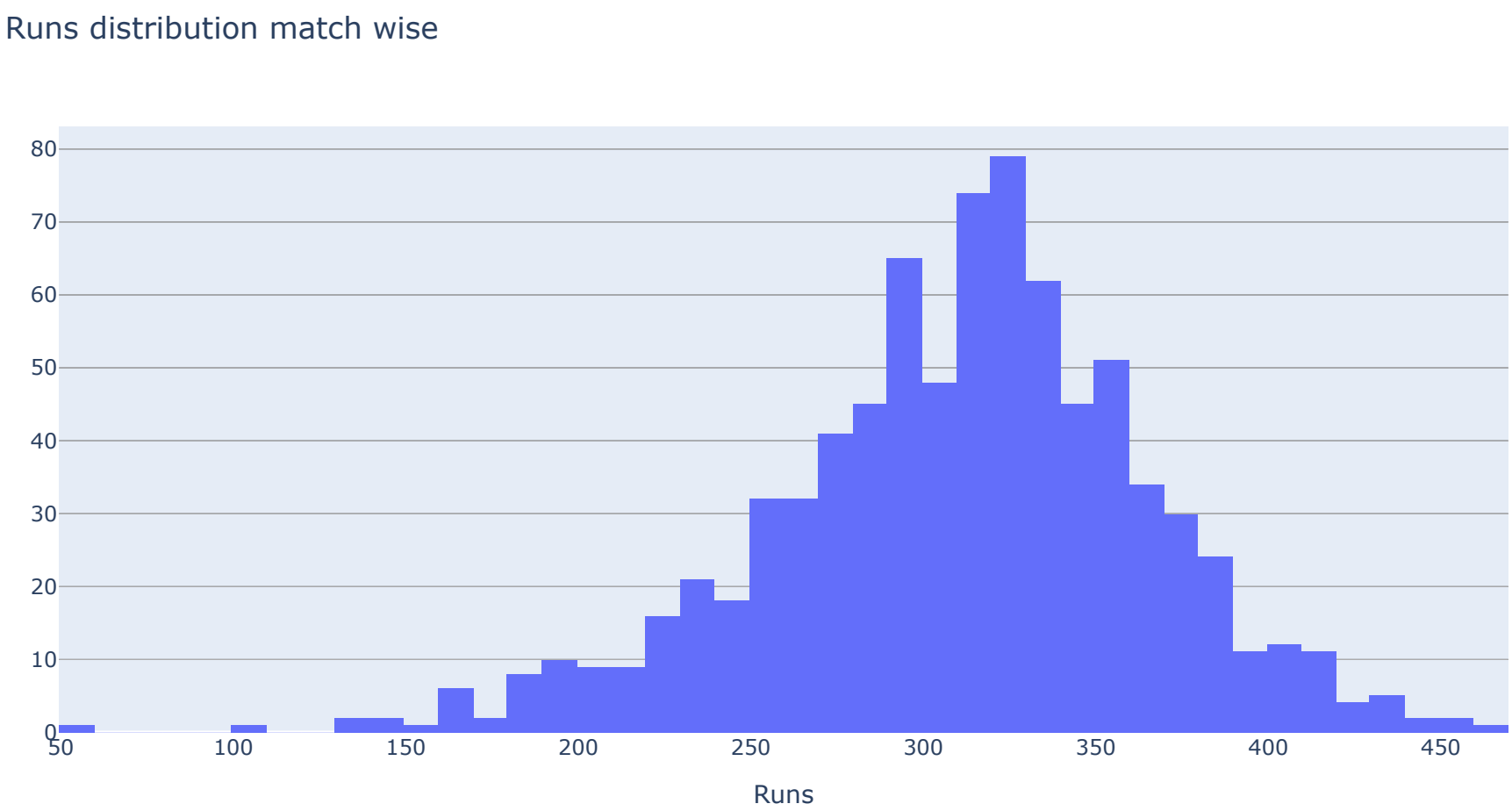
In [143]:
```
runs_by_match = balls.groupby(by='id').sum()
runs_by_match = pd.DataFrame(runs_by_match[['total_runs']])
runs_by_match.reset_index(inplace=True)
runs_by_match
```

Out[143]:

	id	total_runs
0	335982	304
1	335983	447
2	335984	261
3	335985	331
4	335986	222
...	...	...
811	1216547	402
812	1237177	343
813	1237178	263
814	1237180	361
815	1237181	313

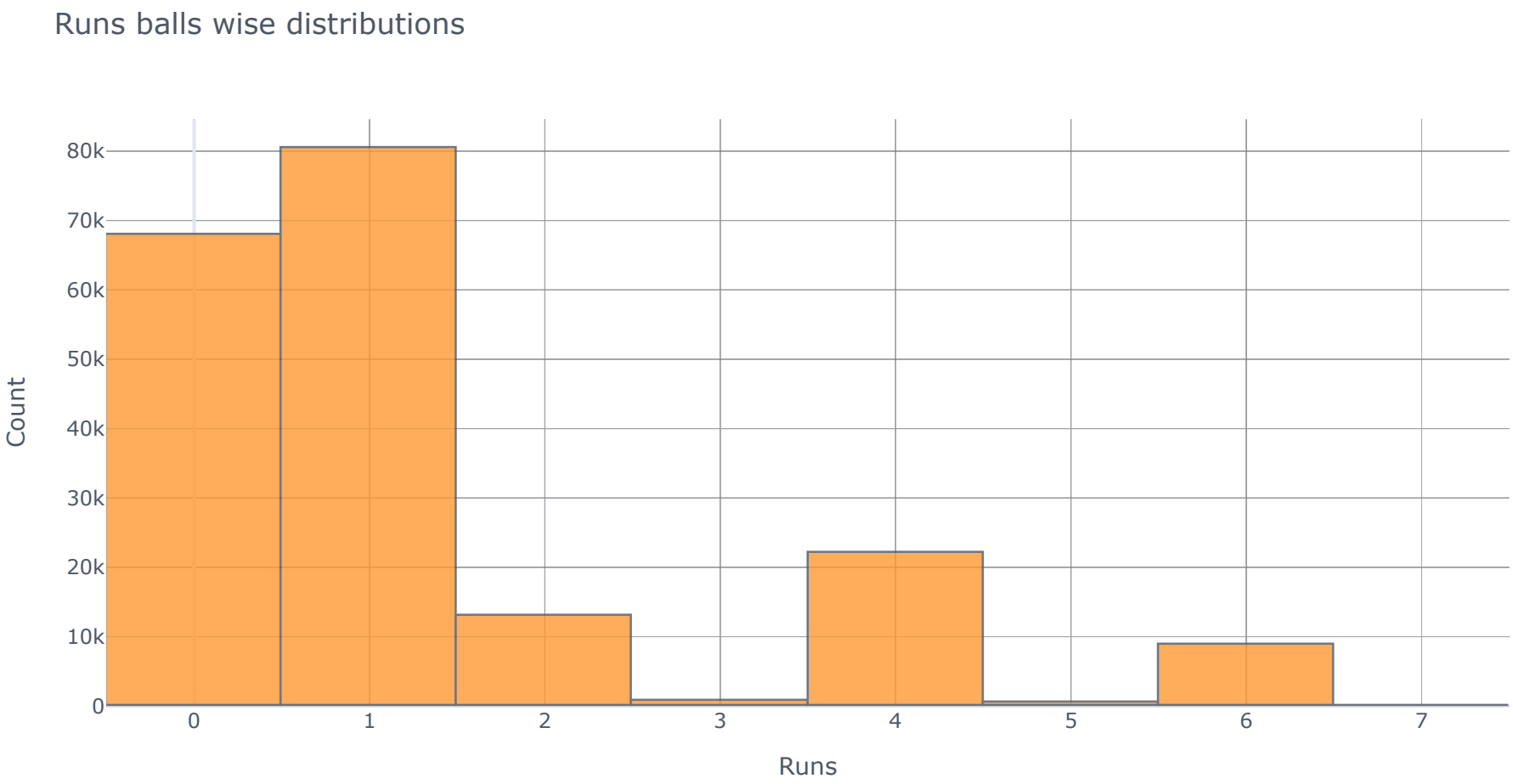
816 rows × 2 columns

In [144]:
```
runs_by_match = go.Histogram(
                    x=runs_by_match['total_runs']
                )

data = [runs_by_match]

layout = go.Layout(title='Runs distribution match wise',
                  xaxis = dict(title='Runs'))

fig = go.Figure(data=data, layout=layout)

pyo.iplot(fig)
```



Runs balls wise distributions

In [145]:
```
balls['total_runs'].iplot(kind='hist', title='Runs balls wise distributions', xTitle='Runs', yTitle='Count')
```



Export to plot.ly »

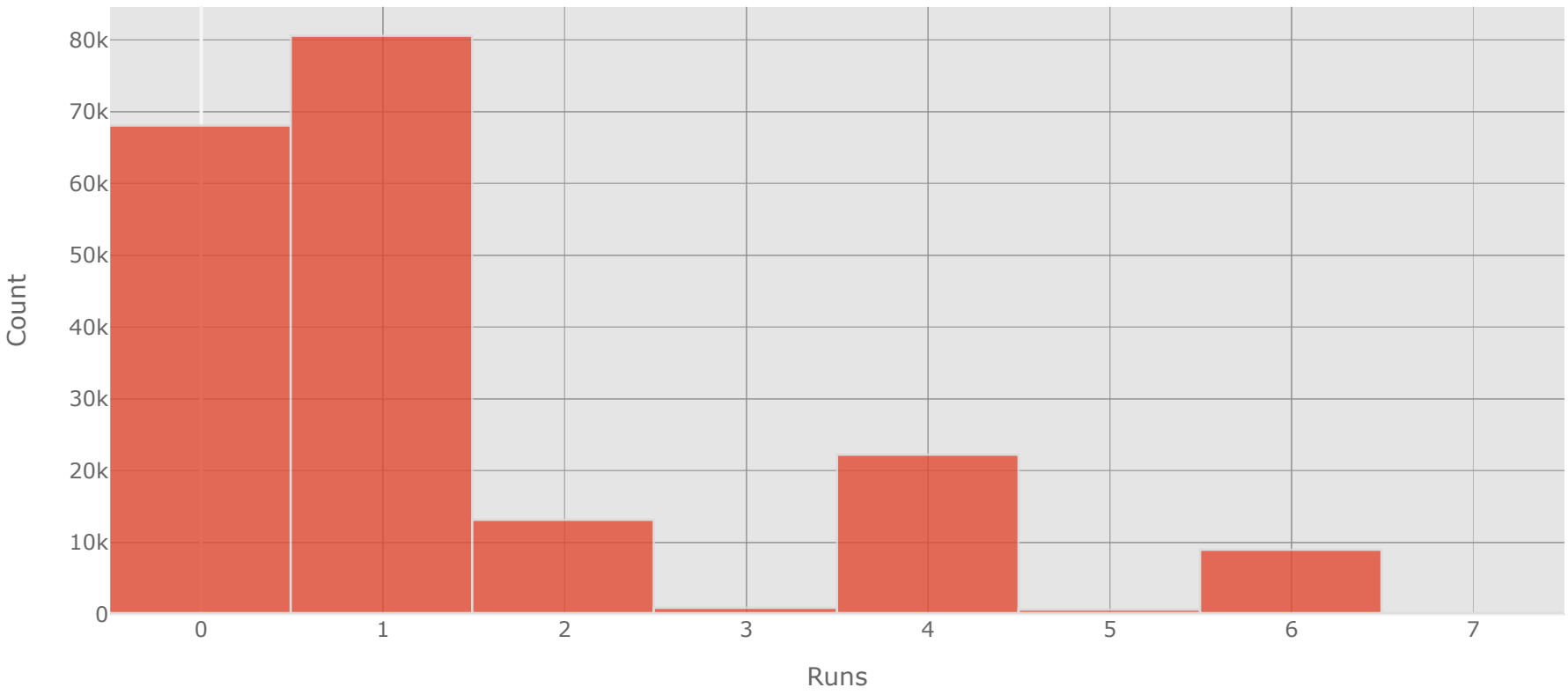
Testing the different themes

In [146]:
```
themes = cf.getThemes()
themes
```

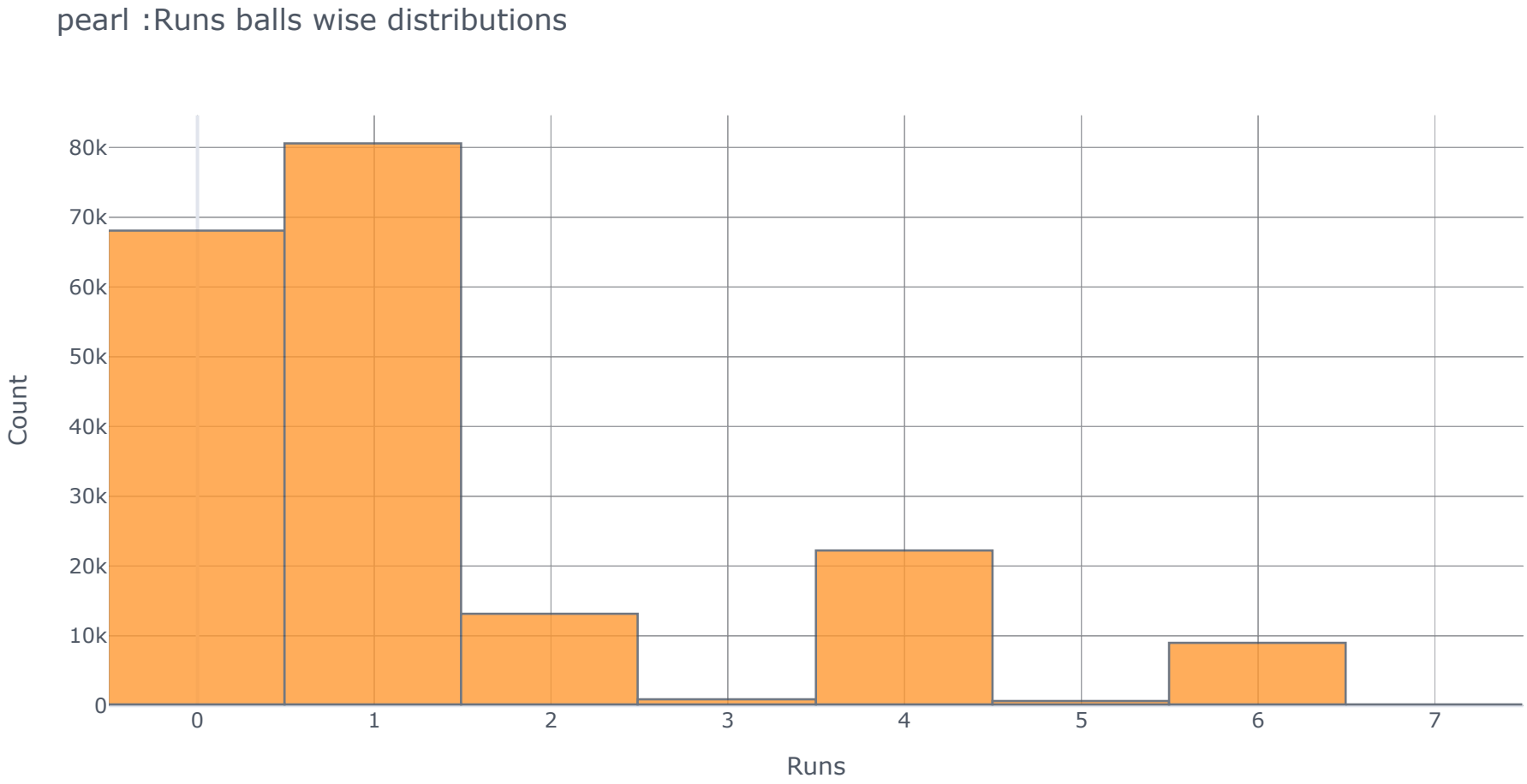
Out[146]:
```
['ggplot', 'pearl', 'solar', 'space', 'white', 'polar', 'henanigans']
```

In [147]:
```
for theme in themes:
    balls['total_runs'].iplot(kind='hist', theme=theme, title=theme+' :Runs balls wise distributions ', xTitle='Runs', yTitle=
'Count')
```

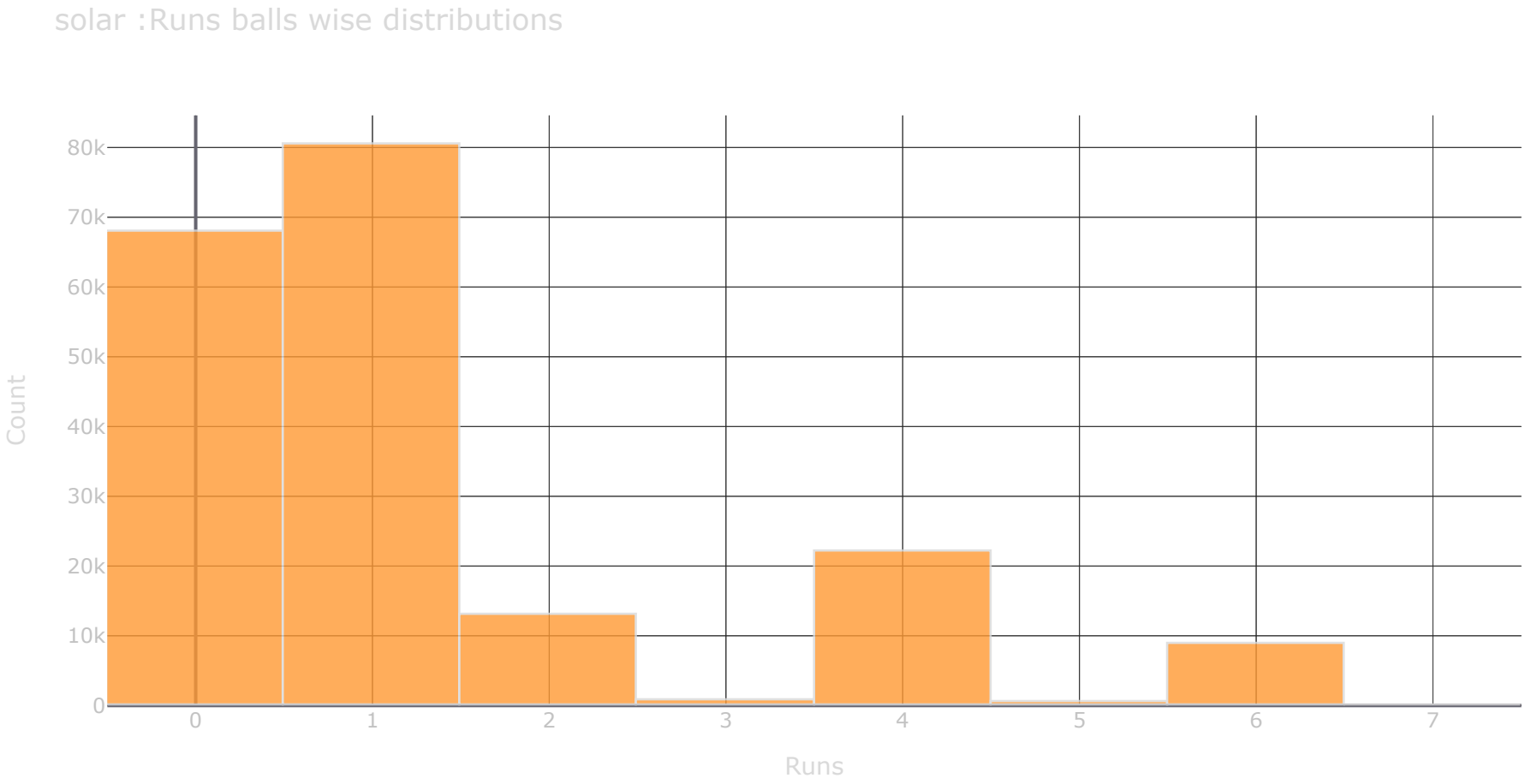
ggplot :Runs balls wise distributions



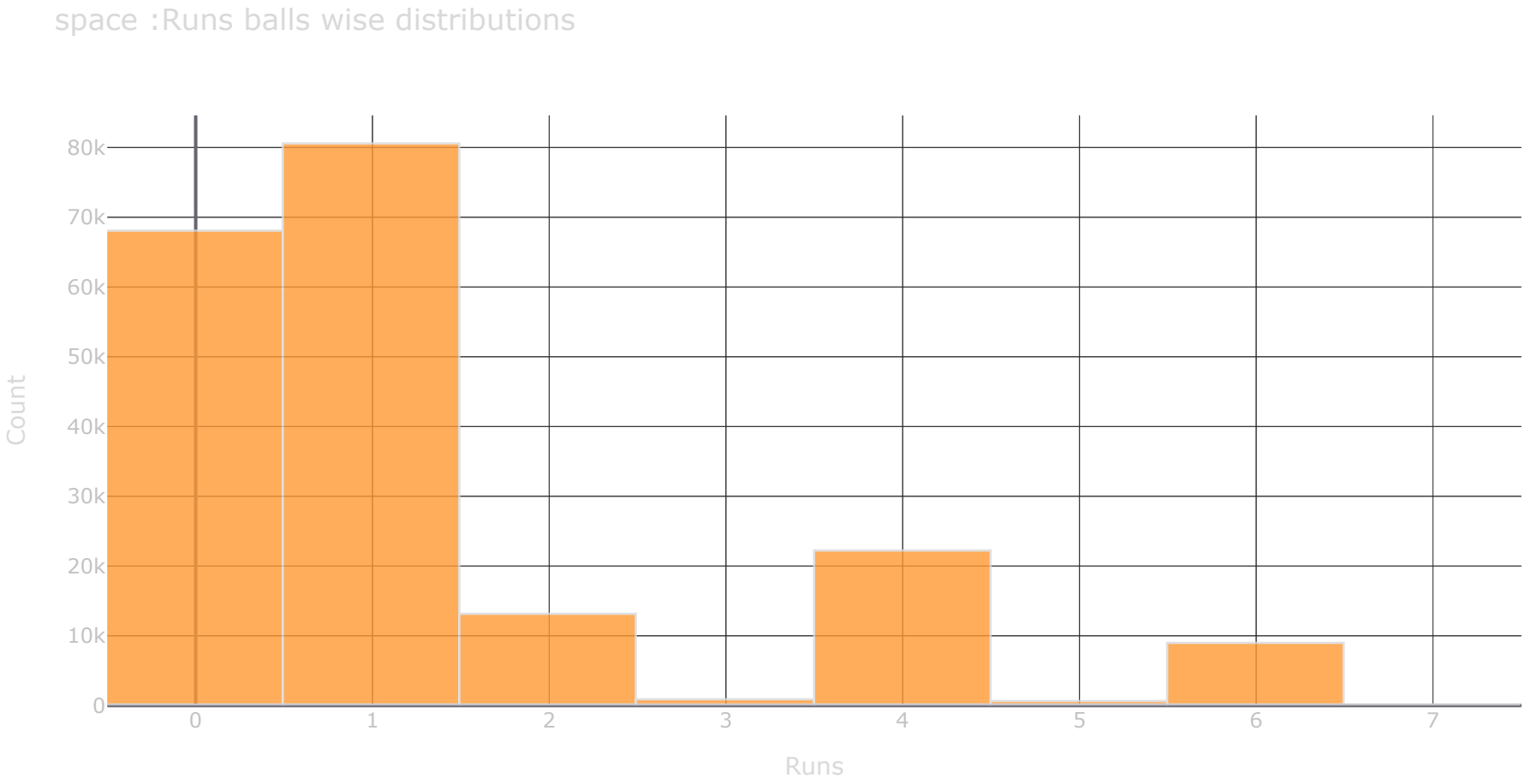
Export to plot.ly »



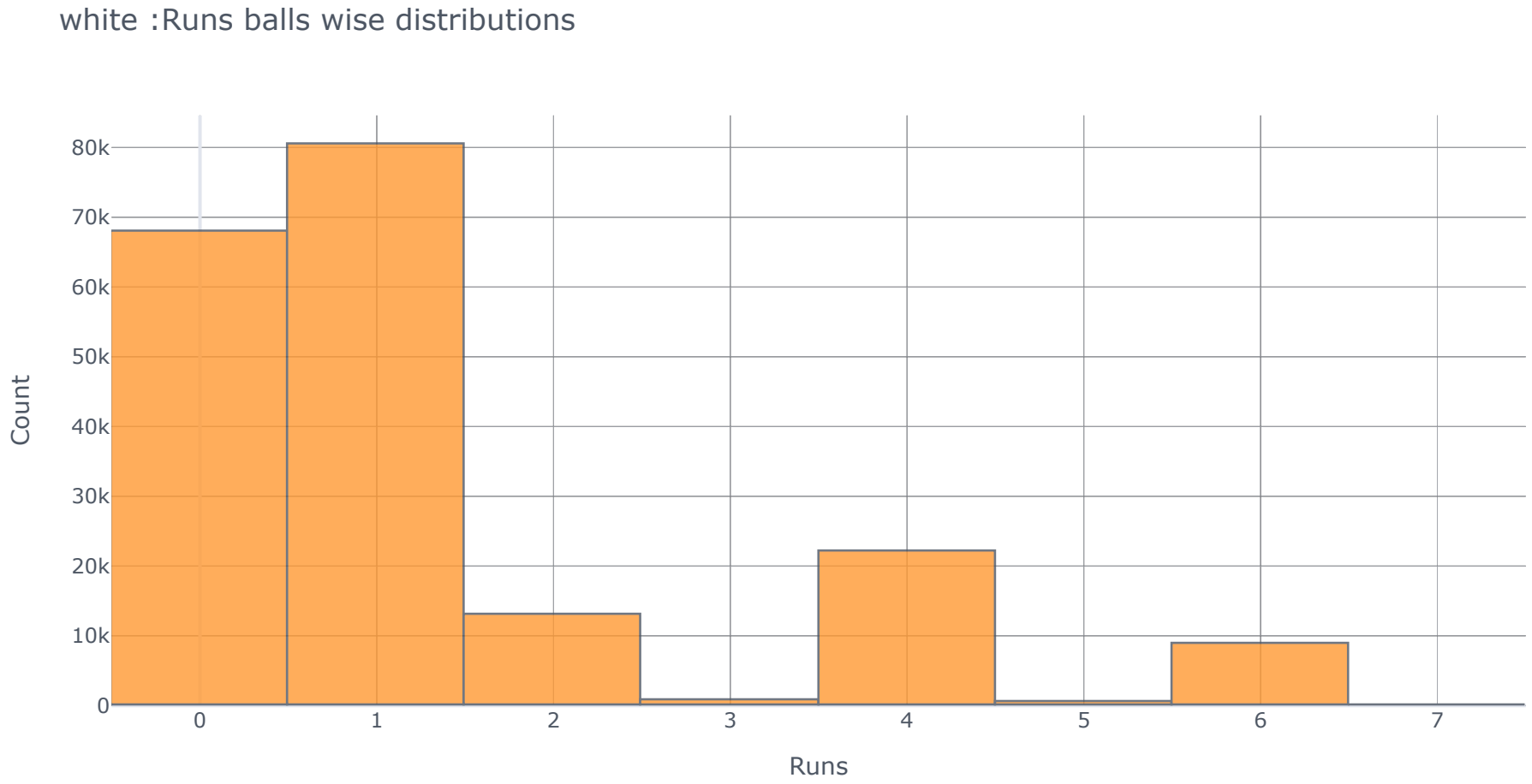
Export to plot.ly »



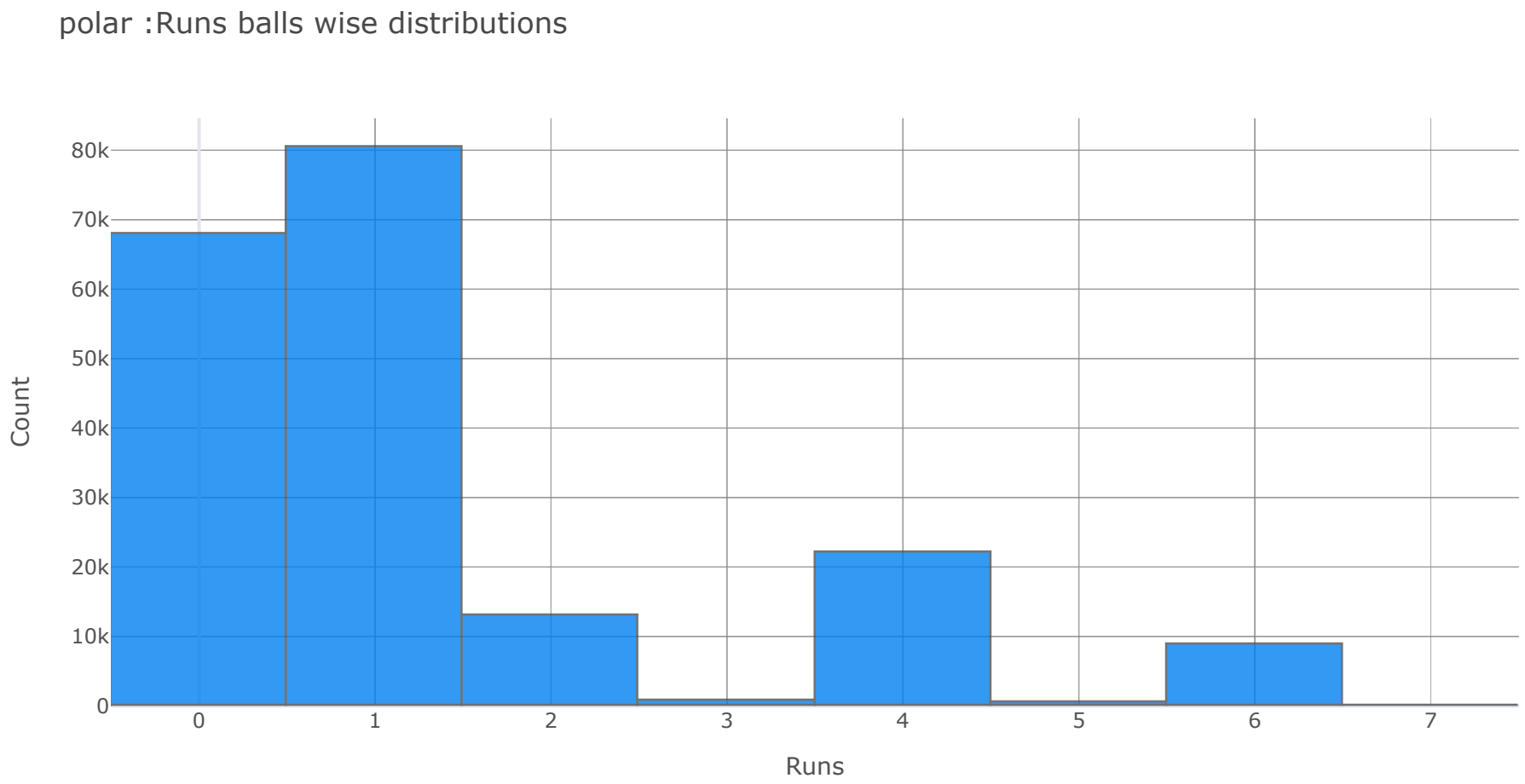
Export to plot.ly »



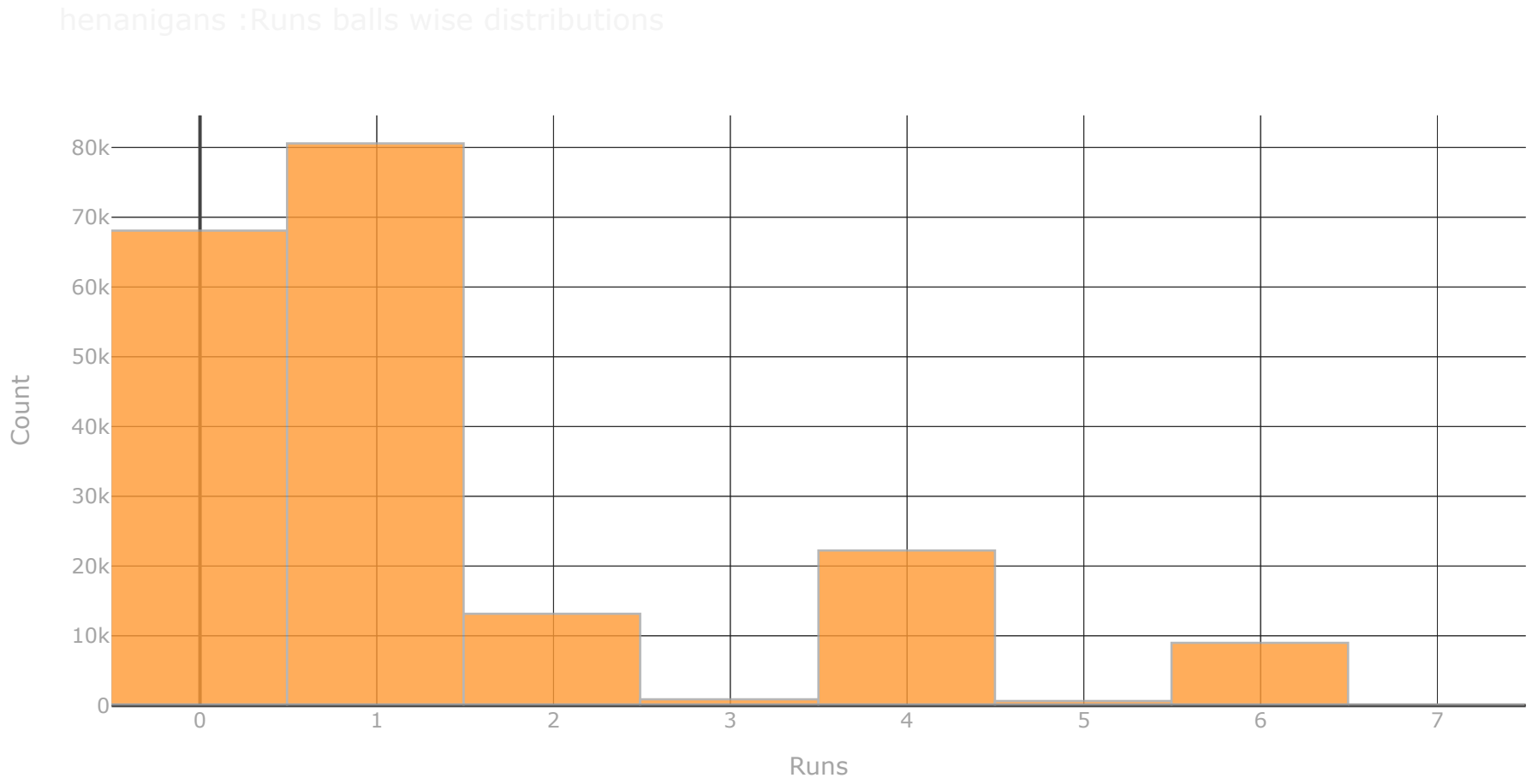
Export to plot.ly »



Export to plot.ly »



Export to plot.ly »

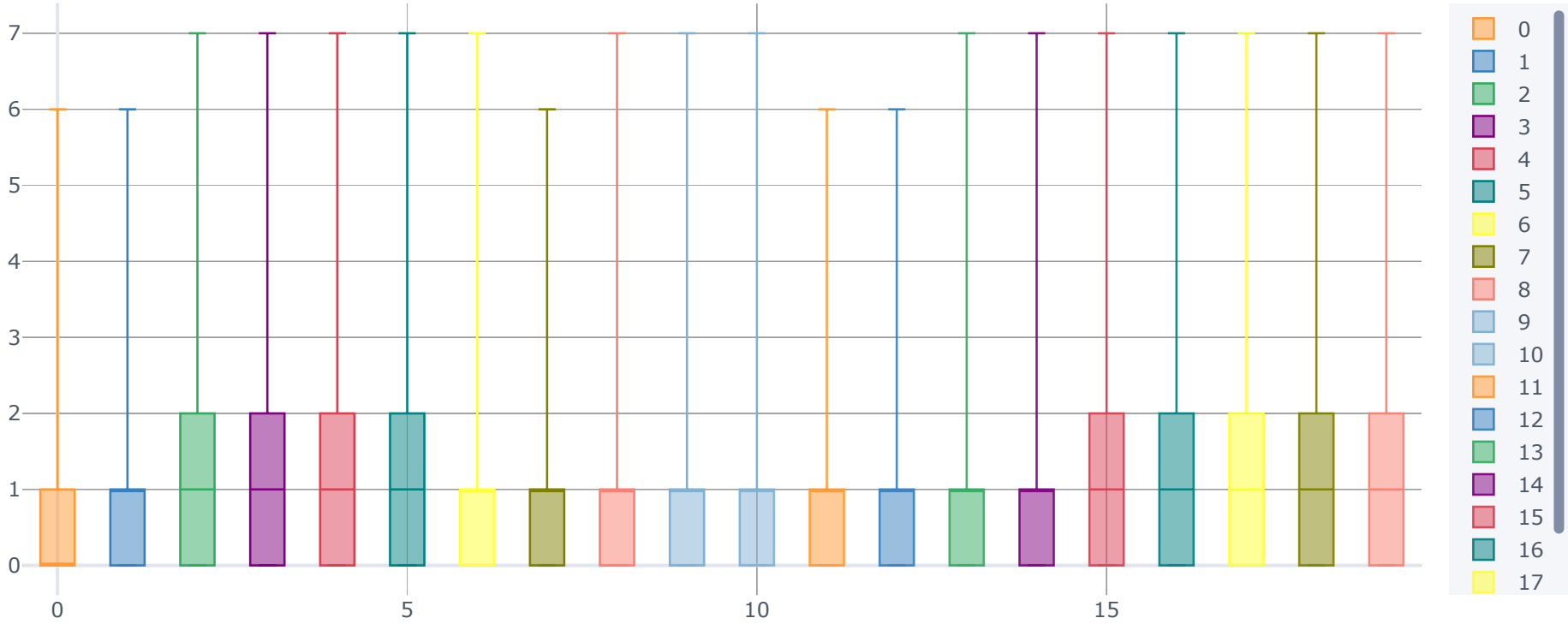


Export to plot.ly »

Runs distribution over wise (using cufflinks)

I had to put this here as it was causing a time-out and preventing other plots from loading.

In [148]:
```
runs_overs = balls[['total_runs', 'over']]
runs_overs.pivot(columns='over', values='total_runs').iplot(kind='box')
```



Export to plot.ly »

Preprocessing and normalization of the data

In [149]:
```
data = pd.merge(left=matches, right=balls, on='id', how='right')
data.head()
```

Out[149]:

	id	city	date	player_of_match	venue	neutral_venue	team1	team2	toss_winner	toss_decision	...	extra_runs	total_runs	non_b
0	335982	Bangalore	2008-04-18	BB McCullum	M. Chinnaswamy Stadium	0	Royal Challengers Bangalore	Kolkata Knight Riders	Royal Challengers Bangalore	field	...	0	1	
1	335982	Bangalore	2008-04-18	BB McCullum	M. Chinnaswamy Stadium	0	Royal Challengers Bangalore	Kolkata Knight Riders	Royal Challengers Bangalore	field	...	0	1	
2	335982	Bangalore	2008-04-18	BB McCullum	M. Chinnaswamy Stadium	0	Royal Challengers Bangalore	Kolkata Knight Riders	Royal Challengers Bangalore	field	...	0	0	
3	335982	Bangalore	2008-04-18	BB McCullum	M. Chinnaswamy Stadium	0	Royal Challengers Bangalore	Kolkata Knight Riders	Royal Challengers Bangalore	field	...	0	1	
4	335982	Bangalore	2008-04-18	BB McCullum	M. Chinnaswamy Stadium	0	Royal Challengers Bangalore	Kolkata Knight Riders	Royal Challengers Bangalore	field	...	0	1	

5 rows × 35 columns

In [150]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 193468 entries, 0 to 193467
Data columns (total 35 columns):
 #   Column            Non-Null Count   Dtype  
---  ------            --------------   -----  
 0   id                193468 non-null  int64  
 1   city              190329 non-null  object 
 2   date              193468 non-null  object 
 3   player_of_match   193096 non-null  object 
 4   venue             193468 non-null  object 
 5   neutral_venue     193468 non-null  int64  
 6   team1             193468 non-null  object 
 7   team2             193468 non-null  object 
 8   toss_winner       193468 non-null  object 
 9   toss_decision     193468 non-null  object 
 10  winner            193096 non-null  object 
 11  result            193096 non-null  object 
 12  result_margin     189871 non-null  float64
 13  eliminator        193096 non-null  object 
 14  method            3208 non-null    object 
 15  umpire1           193468 non-null  object 
 16  umpire2           193468 non-null  object 
 17  season            193468 non-null  object 
 18  inning            193468 non-null  int64  
 19  over              193468 non-null  int64  
 20  ball              193468 non-null  int64  
 21  batsman           193468 non-null  object 
 22  non_striker       193468 non-null  object 
 23  bowler            193468 non-null  object 
 24  batsman_runs      193468 non-null  int64  
 25  extra_runs        193468 non-null  int64  
 26  total_runs        193468 non-null  int64  
 27  non_boundary      193468 non-null  int64  
 28  is_wicket         193468 non-null  int64  
 29  dismissal_kind    9495 non-null    object 
 30  player_dismissed  9495 non-null    object 
 31  fielder           6784 non-null    object 
 32  extras_type       10233 non-null   object 
 33  batting_team      193468 non-null  object 
 34  bowling_team      193277 non-null  object 
dtypes: float64(1), int64(10), object(24)
memory usage: 53.1+ MB
```

In [151]:
```
matches.head()
```

Out[151]:

	id	city	date	player_of_match	venue	neutral_venue	team1	team2	toss_winner	toss_decision	winner	result	result_
0	335982	Bangalore	2008-04-18	BB McCullum	M. Chinnaswamy Stadium	0	Royal Challengers Bangalore	Kolkata Knight Riders	Royal Challengers Bangalore	field	Kolkata Knight Riders	runs	
1	335983	Chandigarh	2008-04-19	MEK Hussey	Punjab Cricket Association Stadium	0	Kings XI Punjab	Chennai Super Kings	Chennai Super Kings	bat	Chennai Super Kings	runs	
2	335984	Delhi	2008-04-19	MF Maharoof	Feroz Shah Kotla	0	Delhi Daredevils	Rajasthan Royals	Rajasthan Royals	bat	Delhi Daredevils	wickets	
3	335985	Mumbai	2008-04-20	MV Boucher	Wankhede Stadium	0	Mumbai Indians	Royal Challengers Bangalore	Mumbai Indians	bat	Royal Challengers Bangalore	wickets	
4	335986	Kolkata	2008-04-20	DJ Hussey	Eden Gardens	0	Kolkata Knight Riders	Deccan Chargers	Deccan Chargers	bat	Kolkata Knight Riders	wickets	

In [152]:
```
matches[pd.isnull(matches['winner'])]
```

Out[152]:

	id	city	date	player_of_match	venue	neutral_venue	team1	team2	toss_winner	toss_decision	winner	result	result_ma
241	501265	Delhi	2011-05-21	NaN	Feroz Shah Kotla	0	Delhi Daredevils	Pune Warriors	Delhi Daredevils	bat	NaN	NaN	
486	829763	Bangalore	2015-04-29	NaN	M. Chinnaswamy Stadium	0	Royal Challengers Bangalore	Rajasthan Royals	Rajasthan Royals	field	NaN	NaN	
511	829813	Bangalore	2015-05-17	NaN	M. Chinnaswamy Stadium	0	Royal Challengers Bangalore	Delhi Daredevils	Royal Challengers Bangalore	field	NaN	NaN	
744	1178424	Bengaluru	2019-04-30	NaN	M.Chinnaswamy Stadium	0	Royal Challengers Bangalore	Rajasthan Royals	Rajasthan Royals	field	NaN	NaN	

In [153]:
```
matches['winner'].fillna('Draw', inplace=True)
```

In [154]:
```
matches.loc[241,'winner']
```

Out[154]:
```
'Draw'
```

In [155]:
```
matches['winner'].value_counts()
```

Out[155]:
```
Mumbai Indians                 120
Chennai Super Kings            106
Kolkata Knight Riders           99
Royal Challengers Bangalore     91
Kings XI Punjab                 88
Rajasthan Royals                81
Delhi Daredevils                67
Sunrisers Hyderabad             66
Deccan Chargers                 29
Delhi Capitals                  19
Rising Pune Supergiant          15
Gujarat Lions                   13
Pune Warriors                   12
Kochi Tuskers Kerala             6
Draw                             4
Name: winner, dtype: int64
```

In [156]:
```
matches[pd.isnull(matches['city'])]
```

Out[156]:

	id	city	date	player_of_match	venue	neutral_venue	team1	team2	toss_winner	toss_decision	winner	result	result_margi
399	729281	NaN	2014-04-17	YS Chahal	Sharjah Cricket Stadium	1	Delhi Daredevils	Royal Challengers Bangalore	Royal Challengers Bangalore	field	Royal Challengers Bangalore	wickets	8
402	729287	NaN	2014-04-19	PA Patel	Dubai International Cricket Stadium	1	Royal Challengers Bangalore	Mumbai Indians	Royal Challengers Bangalore	field	Royal Challengers Bangalore	wickets	1
403	729289	NaN	2014-04-19	JP Duminy	Dubai International Cricket Stadium	1	Kolkata Knight Riders	Delhi Daredevils	Kolkata Knight Riders	bat	Delhi Daredevils	wickets	4
404	729291	NaN	2014-04-20	GJ Maxwell	Sharjah Cricket Stadium	1	Rajasthan Royals	Kings XI Punjab	Kings XI Punjab	field	Kings XI Punjab	wickets	1
406	729295	NaN	2014-04-22	GJ Maxwell	Sharjah Cricket Stadium	1	Kings XI Punjab	Sunrisers Hyderabad	Sunrisers Hyderabad	field	Kings XI Punjab	runs	72
407	729297	NaN	2014-04-23	RA Jadeja	Dubai International Cricket Stadium	1	Rajasthan Royals	Chennai Super Kings	Rajasthan Royals	field	Chennai Super Kings	runs	1
408	729299	NaN	2014-04-24	CA Lynn	Sharjah Cricket Stadium	1	Royal Challengers Bangalore	Kolkata Knight Riders	Royal Challengers Bangalore	field	Kolkata Knight Riders	runs	2
409	729301	NaN	2014-04-25	AJ Finch	Dubai International Cricket Stadium	1	Sunrisers Hyderabad	Delhi Daredevils	Sunrisers Hyderabad	bat	Sunrisers Hyderabad	runs	4
410	729303	NaN	2014-04-25	MM Sharma	Dubai International Cricket Stadium	1	Chennai Super Kings	Mumbai Indians	Mumbai Indians	bat	Chennai Super Kings	wickets	1
413	729309	NaN	2014-04-27	M Vijay	Sharjah Cricket Stadium	1	Delhi Daredevils	Mumbai Indians	Mumbai Indians	bat	Delhi Daredevils	wickets	6
414	729311	NaN	2014-04-27	DR Smith	Sharjah Cricket Stadium	1	Sunrisers Hyderabad	Chennai Super Kings	Sunrisers Hyderabad	bat	Chennai Super Kings	wickets	5
415	729313	NaN	2014-04-28	Sandeep Sharma	Dubai International Cricket Stadium	1	Kings XI Punjab	Royal Challengers Bangalore	Kings XI Punjab	field	Kings XI Punjab	wickets	5

	id	city	date	player_of_match	venue	neutral_venue	team1	team2	toss_winner	toss_decision	winner	result	result_margi
417	729317	NaN	2014-04-30	B Kumar	Dubai International Cricket Stadium	1	Mumbai Indians	Sunrisers Hyderabad	Mumbai Indians	field	Sunrisers Hyderabad	runs	1

In [157]:
```
matches['city'].fillna('UAE', inplace=True)
```

In [158]:
```
matches.loc[414,'city']
```

Out[158]:
```
'UAE'
```

In [159]:
```
matches['toss_decision'].value_counts()
```

Out[159]:
```
field    496
bat      320
Name: toss_decision, dtype: int64
```

In [160]:
```
encode = {'city':{'Mumbai':1 ,'Kolkata':2,'Delhi':3,'Bangalore':4,'Hyderabad':5,'Chennai':6,'Chandigarh':7,'Jaipur':8,'Pune':9,
'Abu Dhabi':10,'Dubai':11,'Bengaluru':12,'Durban':13,'Visakhapatnam':14,'Ahmedabad':15,'Centurion':16,'Sharjah':17,'Rajkot':18,
'Dharamsala':19,'Indore':20,'Johannesburg':21,'Port Elizabeth':22,'Ranchi':23,'Cape Town':24,'Cuttack':25,'Raipur':26,'Kochi':2
7,'Kanpur':28,'Nagpur':29,'Kimberley':30,'East London':31,'Bloemfontein':32,'UAE':33},
          'team1': {'Mumbai Indians':1,'Kolkata Knight Riders':2,'Royal Challengers Bangalore':3,'Delhi Capitals':4,'Chennai Su
per Kings':5,'Rajasthan Royals':6,'Delhi Daredevils':7,'Gujarat Lions':8,'Kings XI Punjab':9,'Sunrisers Hyderabad':10,'Rising P
une Supergiant':11,'Kochi Tuskers Kerala':12,'Pune Warriors':13,'Deccan Chargers':14},
          'team2': {'Mumbai Indians':1,'Kolkata Knight Riders':2,'Royal Challengers Bangalore':3,'Delhi Capitals':4,'Chennai Su
per Kings':5,'Rajasthan Royals':6,'Delhi Daredevils':7,'Gujarat Lions':8,'Kings XI Punjab':9,'Sunrisers Hyderabad':10,'Rising P
une Supergiant':11,'Kochi Tuskers Kerala':12,'Pune Warriors':13,'Deccan Chargers':14},
          'toss_winner': {'Mumbai Indians':1,'Kolkata Knight Riders':2,'Royal Challengers Bangalore':3,'Delhi Capitals':4,'Chen
nai Super Kings':5,'Rajasthan Royals':6,'Delhi Daredevils':7,'Gujarat Lions':8,'Kings XI Punjab':9,'Sunrisers Hyderabad':10,'Ri
sing Pune Supergiant':11,'Kochi Tuskers Kerala':12,'Pune Warriors':13,'Deccan Chargers':14,'Rising Pune Supergiants':11},
          'winner': {'Mumbai Indians':1,'Kolkata Knight Riders':2,'Royal Challengers Bangalore':3,'Delhi Capitals':4,'Chennai S
uper Kings':5,'Rajasthan Royals':6,'Delhi Daredevils':7,'Gujarat Lions':8,'Kings XI Punjab':9,'Sunrisers Hyderabad':10,'Rising
 Pune Supergiant':11,'Kochi Tuskers Kerala':12,'Pune Warriors':13,'Deccan Chargers':14,'Draw':15}}
matches.replace(encode, inplace=True)
matches.head()
```

Out[160]:

	id	city	date	player_of_match	venue	neutral_venue	team1	team2	toss_winner	toss_decision	winner	result	result_margin	eliminator	m
0	335982	4	2008-04-18	BB McCullum	M. Chinnaswamy Stadium	0	3	2	3	field	2	runs	140	N	
1	335983	7	2008-04-19	MEK Hussey	Punjab Cricket Association Stadium	0	9	5	5	bat	5	runs	33	N	
2	335984	3	2008-04-19	MF Maharoof	Feroz Shah Kotla	0	7	6	6	bat	7	wickets	9	N	
3	335985	1	2008-04-20	MV Boucher	Wankhede Stadium	0	1	3	1	bat	3	wickets	5	N	
4	335986	2	2008-04-20	DJ Hussey	Eden Gardens	0	2	14	14	bat	2	wickets	5	N	

In [161]:
```
matches['result'].value_counts()
```

Out[161]:
```
wickets    435
runs       364
tie         13
Name: result, dtype: int64
```

In [162]:
```
matches['eliminator'].value_counts()
```

Out[162]:
```
N    799
Y     13
Name: eliminator, dtype: int64
```

In [163]:
```
matches['method'].value_counts()
```

Out[163]:
```
D/L    19
Name: method, dtype: int64
```

In [164]:
```
encode = {'toss_decision': {'field':1,'bat':2},
          'result': {'wicket':1,'runs':2,'tie':3},
          'eliminator': {'N':1,'Y':2}}
matches.replace(encode, inplace=True)
matches.head()
```

Out[164]:

	id	city	date	player_of_match	venue	neutral_venue	team1	team2	toss_winner	toss_decision	winner	result	result_margin	eliminator	m
0	335982	4	2008-04-18	BB McCullum	M. Chinnaswamy Stadium	0	3	2	3	1	2	2	140	1	
1	335983	7	2008-04-19	MEK Hussey	Punjab Cricket Association Stadium	0	9	5	5	2	5	2	33	1	
2	335984	3	2008-04-19	MF Maharoof	Feroz Shah Kotla	0	7	6	6	2	7	wickets	9	1	
3	335985	1	2008-04-20	MV Boucher	Wankhede Stadium	0	1	3	1	2	3	wickets	5	1	
4	335986	2	2008-04-20	DJ Hussey	Eden Gardens	0	2	14	14	2	2	wickets	5	1	

In [165]:

```
encode = {'M. Chinnaswamy Stadium':1, 'Punjab Cricket Association Stadium':2,
       'Feroz Shah Kotla':3, 'Wankhede Stadium':4, 'Eden Gardens':5,
       'Sawai Mansingh Stadium':6, 'Rajiv Gandhi International Stadium':7,
       'M.A. Chidambaram Stadium':8, 'Dr DY Patil Sports Academy':9, 'Newlands':10,
       "St George's Park":11, 'Kingsmead':12, 'SuperSport Park':13, 'Buffalo Park':14,
       'New Wanderers Stadium':15, 'De Beers Diamond Oval':16, 'OUTsurance Oval':17,
       'Brabourne Stadium':18, 'Sardar Patel Stadium, Motera':19, 'Barabati Stadium':20,
       'Vidarbha Cricket Association Stadium, Jamtha':21,
       'Himachal Pradesh Cricket Association Stadium':22, 'Nehru Stadium':23,
       'Holkar Cricket Stadium':24,
       'Dr. Y.S. Rajasekhara Reddy ACA-VDCA Cricket Stadium':25,
       'Subrata Roy Sahara Stadium':26,
       'Shaheed Veer Narayan Singh International Stadium':27,
       'JSCA International Stadium Complex':28, 'Sheikh Zayed Stadium':29,
       'Sharjah Cricket Stadium':30, 'Dubai International Cricket Stadium':31,
       'Maharashtra Cricket Association Stadium':32,
       'Saurashtra Cricket Association Stadium':33, 'Green Park':34,
       'M.Chinnaswamy Stadium':35}
matches.replace(encode, inplace=True)
matches.head()
```

Out[165]:

	id	city	date	player_of_match	venue	neutral_venue	team1	team2	toss_winner	toss_decision	winner	result	result_margin	eliminator	method
0	335982	4	2008-04-18	BB McCullum	1	0	3	2	3	1	2	2	140	1	NaN
1	335983	7	2008-04-19	MEK Hussey	2	0	9	5	5	2	5	2	33	1	NaN
2	335984	3	2008-04-19	MF Maharoof	3	0	7	6	6	2	7	wickets	9	1	NaN
3	335985	1	2008-04-20	MV Boucher	4	0	1	3	1	2	3	wickets	5	1	NaN
4	335986	2	2008-04-20	DJ Hussey	5	0	2	14	14	2	2	wickets	5	1	NaN

In [166]:
```
matches.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 816 entries, 0 to 815
Data columns (total 18 columns):
 #   Column           Non-Null Count  Dtype
---  ------           --------------  -----
 0   id               816 non-null    int64
 1   city             816 non-null    int64
 2   date             816 non-null    object
 3   player_of_match  812 non-null    object
 4   venue            816 non-null    int64
 5   neutral_venue    816 non-null    int64
 6   team1            816 non-null    int64
 7   team2            816 non-null    int64
 8   toss_winner      816 non-null    int64
 9   toss_decision    816 non-null    int64
 10  winner           816 non-null    int64
 11  result           812 non-null    object
 12  result_margin    799 non-null    float64
 13  eliminator       812 non-null    float64
 14  method           19 non-null     object
 15  umpire1          816 non-null    object
 16  umpire2          816 non-null    object
 17  season           816 non-null    object
dtypes: float64(2), int64(9), object(7)
memory usage: 114.9+ KB
```

Normalization of data

In [167]:
```
from sklearn.preprocessing import MinMaxScaler
scaling = MinMaxScaler()
X = scaling.fit_transform(data[['id','neutral_venue','inning','over','ball','batsman_runs','extra_runs','total_runs','non_bound
ary','is_wicket']])
```

In [168]:
```
X_meaned = X - np.mean(X , axis = 0)

cov_mat = np.cov(X_meaned , rowvar = False)
```

In [169]:
```
mean_vec = np.mean(X, axis=0)
cov_mat = (X - mean_vec).T.dot((X - mean_vec)) / (X.shape[0]-1)
print('Covariance matrix \n%s' %cov_mat)
```

```
Covariance matrix
[[ 1.15365779e-01 -2.91746954e-02  5.09370508e-05  7.99041556e-04
  -1.50071061e-04  2.59402183e-03 -3.02410076e-04  1.92103720e-03
   7.67595784e-07 -2.07032233e-04]
 [-2.91746954e-02  8.58353255e-02  1.64457592e-04 -1.12281269e-04
   5.00474710e-05 -1.66029768e-03  5.87528883e-05 -1.36435941e-03
  -2.67349379e-06  1.68592814e-04]
 [ 5.09370508e-05  1.64457592e-04  2.49682004e-01 -6.94406981e-03
  -4.37800993e-04 -9.79513846e-04 -1.32222710e-05 -8.52805568e-04
   6.64659818e-06 -1.07707031e-04]
 [ 7.99041556e-04 -1.12281269e-04 -6.94406981e-03  8.92703657e-02
  -4.64116648e-04  6.97604489e-03 -1.89115799e-05  5.96055547e-03
  -3.21894338e-06  4.76271394e-03]
 [-1.50071061e-04  5.00474710e-05 -4.37800993e-04 -4.64116648e-04
   5.10267217e-02  4.34686382e-04 -2.72809896e-05  3.45307338e-04
  -4.42936003e-06  2.23157310e-04]
 [ 2.59402183e-03 -1.66029768e-03 -9.79513846e-04  6.97604489e-03
   4.34686382e-04  7.20803069e-02 -1.83711992e-03  5.99460003e-02
   4.83771607e-05 -9.86214083e-03]
 [-3.02410076e-04  5.87528883e-05 -1.32222710e-05 -1.89115799e-05
  -2.72809896e-05 -1.83711992e-03  2.35906344e-03  7.84389222e-04
  -7.84648651e-07 -4.33888487e-04]
 [ 1.92103720e-03 -1.36435941e-03 -8.52805568e-04  5.96055547e-03
   3.45307338e-04  5.99460003e-02  7.84389222e-04  5.21666752e-02
   4.06814891e-05 -8.88715206e-03]
 [ 7.67595784e-07 -2.67349379e-06  6.64659818e-06 -3.21894338e-06
  -4.42936003e-06  4.83771607e-05 -7.84648651e-07  4.06814891e-05
   8.26946031e-05 -4.05881178e-06]
 [-2.07032233e-04  1.68592814e-04 -1.07707031e-04  4.76271394e-03
   2.23157310e-04 -9.86214083e-03 -4.33888487e-04 -8.88715206e-03
  -4.05881178e-06  4.66694862e-02]]
```

In [170]:
```
matches = matches[['team1','team2','city','toss_decision','toss_winner','venue','winner']]
matches.head()
```

Out[170]:

	team1	team2	city	toss_decision	toss_winner	venue	winner
0	3	2	4	1	3	1	2
1	9	5	7	2	5	2	5
2	7	6	3	2	6	3	7
3	1	3	1	2	1	4	3
4	2	14	2	2	14	5	2

In [171]:
```
matches.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 816 entries, 0 to 815
Data columns (total 7 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   team1          816 non-null    int64
 1   team2          816 non-null    int64
 2   city           816 non-null    int64
 3   toss_decision  816 non-null    int64
 4   toss_winner    816 non-null    int64
 5   venue          816 non-null    int64
 6   winner         816 non-null    int64
dtypes: int64(7)
memory usage: 44.8 KB
```

In [172]:
```
df = pd.DataFrame(matches)
df.describe()
```

Out[172]:

	team1	team2	city	toss_decision	toss_winner	venue	winner
count	816	816	816	816	816	816	816
mean	6	6	8	1	6	12	6
std	4	4	7	0	4	11	4
min	1	1	1	1	1	1	1
25%	3	3	3	1	3	4	2
50%	5	6	6	1	6	7	5
75%	9	9	11	2	9	20	9
max	14	14	33	2	14	35	15

In [173]:
```
pd.options.display.float_format = '{:,.0f}'.format
df
```

Out[173]:

	team1	team2	city	toss_decision	toss_winner	venue	winner
0	3	2	4	1	3	1	2
1	9	5	7	2	5	2	5
2	7	6	3	2	6	3	7
3	1	3	1	2	1	4	3
4	2	14	2	2	14	5	2
...	...	...	...	...	...	...	...
811	3	1	11	1	1	31	3
812	1	4	11	1	4	31	1
813	3	10	10	1	10	29	10
814	4	10	10	2	4	29	4
815	4	1	11	2	4	31	1

816 rows × 7 columns

In [174]:
```
df['city'].value_counts()
```

Out[174]:
```
1     101
2      77
3      74
4      65
5      64
6      57
7      56
8      47
9      38
10     29
11     26
12     15
13     15
14     13
33     13
17     12
15     12
16     12
18     10
19      9
20      9
21      8
22      7
23      7
25      7
24      7
26      6
27      5
28      4
29      3
30      3
31      3
32      2
Name: city, dtype: int64
```

In [175]:
```
df.apply(lambda x: sum(x.isnull()),axis=0)
    #find the null values in every column
```

Out[175]:
```
team1            0
team2            0
city             0
toss_decision    0
toss_winner      0
venue            0
winner           0
dtype: int64
```

In [176]:
```
df.head()
```

Out[176]:

	team1	team2	city	toss_decision	toss_winner	venue	winner
0	3	2	4	1	3	1	2
1	9	5	7	2	5	2	5
2	7	6	3	2	6	3	7
3	1	3	1	2	1	4	3
4	2	14	2	2	14	5	2

Implement different algorithm to report the accuracy

In [177]:
```
#Import models from scikit learn module:
from sklearn.linear_model import LogisticRegression
#from sklearn.cross_validation import KFold   #For K-fold cross validation
from sklearn.ensemble import RandomForestClassifier
from sklearn.tree import DecisionTreeClassifier, export_graphviz
from sklearn import metrics

#Generic function for making a classification model and accessing performance:
def classification_model(model, data, predictors, outcome):
      model.fit(data[predictors],data[outcome])
      predictions = model.predict(data[predictors])
      print(predictions)
      accuracy = metrics.accuracy_score(predictions,data[outcome])
      print('Accuracy : %s' % '{0:.3%}'.format(accuracy))
```

logistic Regression

In [178]:
```
outcome_var=['winner']
predictor_var = ['team1', 'team2', 'venue', 'toss_winner','city','toss_decision']
model =LogisticRegression()
classification_model(model, df,predictor_var,outcome_var)
```

```
[ 1  5  5  1  5  5 10  1 10  9  3  1 11  7  1  1  2 10  1  5  5  5  1  5
  3 14  1  9  1  9  1  9  7  9  5  2  1  9 14  1  5  2 14  1  3  1  2  1
  9  1  5 14  5  1  7 10  5  2  3  5 14  9  3  9 14  5  9  5  7  5  7 14
  3  5  1  5  9  9  3  5 14  9  7  3  9  9  5  7  3  5  7 14  7  3  3 14
  3  5 14  3  3 14  5 14 14  7  5  3  3  5  3  9 14  3  9 10  1  9  1  5
  9  5  1  1  2  5 14  5  1  9  9  1  1  9  1  6  5  2  5 10  1 11  1  5
  5  5  5  5  5  1  9  1  5  2 14  2  9  1  5  1  9  1  1  5  5 14  1  1
  9  9  1  1 10  5  1  1 10  7  1 10  5  5  1  9 10 10  2  3  1 10 10  1
 14  9  3  9  9  1  1  9 14  9 14  2  5 14  1  1 14  9  2  9  9  3 14 14
  5 10  9 10  3  1  1  5  9  9 10  9  2  5  9  1  3  9  9 11  5  5 10  1
  9  9  1  1  1  1  1  1  1  1  9  9  3 10  2 10  5  1  9  1  1  9  1  9
 10  2  1  1  9  5  5 10 14  9  5  9  2  9  3 10  5 10  2  5  1  5  9  1
 14  5  3  2  9  5  5  1  9  1  5 14  1  9 10  1  9  9  9  1  1  5  1  9
  1 10  9  2 10  1  1  1  9  1  2  1 14  5  1 10  5  2  1  9  1 10  5  5
  1  5  9  5  9  2 10  1  5 10  1  2  2  9  5 14  9  1  5  5  5  1  1 14
  3  2 10  2  5  5  5  1  5  3  1  6  5  5  1  9  9  2  1  5  9  3  9  9
  3  9  5  1  7  5  9 10  1  2  1  5  1  1  1  9  5 10  3  3  9  5 14  9
  3  9  3  1  2  5  7  9  1  9  3  1  9  9  5  9  1  1  9  9  9  9  1  9
  5  1  5  3  9  3  9  3  5  5  9  5  1  9  1  9  1  9  1  9  9  1  1  1
  9  9  1  9 10  5  1  9  2  9  5  1  5 10  1  5  2  5  1  2  9 10  1  2
  5  2  5  2  9  2  3  1  9  1  1  9  2 11  1  3  1  3 11  1  5  9  1  5
 10  5  9  1  5  5  1  2  1  1  1  3  1  5  2  9  9  1  5  9  5  3  9  3
  1  9  1 10 10  2  9  5 10  7 10  9  1 10  2  9 10  2  1 10  3  9  9  9
  9  5  3  7 10  1  9  5  3  5  9  3  1 10  3  3  9 10  3  3  9  9  9  9
  5  9  1  5  9  5  9  1  5 10  2  2  1 10  3  5  2  9  1  9  3  5  9  3
  1 11 10  1  9 10  3  2  9 10  5  9  9  1 11  9  3  9  9 10  1  1  9  2
  9  9  2  5  9  1 11  2  9  9  1  5  1  9  1  9  1  9  1  4  1  3  1  9
  1  1  2  9  1  3  4  9  1  9  2  1  9  2  1  5  1  3  1  9  1  9  1  5
  1  9  9  5  1  5  5  3  2 11  1  9  1  3  3  9  5  5  1 11 10  1  9  9
  1  9  1  9  1  3  1  5  7  1  9  5  9  3  1  9  1  5  1  1  2  9  1  2
  9  1  1  9  1  9  1  9  5  1  1  1  9  5  1  9  9  3  1  1  9  1  1  9
  3  2  2  9  5  4  9  1  1  9  3  1  1  3  1  3  3  1 10  1  3  1  3  3
  1  1 10  5  1  5  2  1  3  5  1  1  5  5 10  1  3  1  1  5 10  1  1 10
  5  1  1  5 10  5  3  1  1 10  5  1  1  1  5  5  1  5  5  1  1  3  5  1]
Accuracy : 25.980%
```

```
/home/dharmveer/venv/lib/python3.6/site-packages/sklearn/utils/validation.py:72: DataConversionWarning:

A column-vector y was passed when a 1d array was expected. Please change the shape of y to (n_samples, ), for example using rav
el().

/home/dharmveer/venv/lib/python3.6/site-packages/sklearn/linear_model/_logistic.py:764: ConvergenceWarning:

lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
```

NAive bayes algorithm

In [179]:
```
from sklearn.naive_bayes import GaussianNB
outcome_var=['winner']
predictor_var = ['team1', 'team2', 'venue', 'toss_winner','city','toss_decision']
model = GaussianNB()
classification_model(model, df,predictor_var,outcome_var)
```

```
[12  6  5  1 13  6 14 12 12 12 12  1 12  6  5  1 12 12  5  5 14  6 12  5
 12 12 12 12  1 12  1 12  2 12  2  1 12 12 12  1  5 12 12  1 12 12 12  1
 12 12  5 12  5 12 14 12  5 12 12  5 12  7  5 12 14  5 12  5 14  5  7 12
  3  5  1  6 12 12  3  5 14 12  7  3 14 12  5 14 12  5  7 12  7  3 12 14
 12  7 12  5  5 12  5 12 12  7  5  5 12  5 12  5 12 12 12 12  1 12 12 14
 12  6  1 12 12  5 12  5  1 14 12  1 12 12 12 14  1 12  5 12  5 12  1  5
  2  6  5  2  1  5 14  1  6  1 12  1 14 12  5 12 12  1  1  5  5 12 12  1
 12 13  1  1  5 14  1  5 12 14  5 10  2  6  1 12 13 12 12 12  5 12 12 12
 12 13 13 12 12 12 12 12 12 12 12 12 14 12 12 12 12  7 12 12  5 12 12 12
  5 12 12 12 12  1 12 13 10 12 13 12 12  5 12 12 13 12 12 12  7  5 12  1
 12 10 12  1 12  1 12  5 12 12 12 12 12 12 12 13 14 12 12 12  1 12  1 13
 10 12  5 12 14 13  6 14 12 12  5 12  1 12 12 13  6 13 12  6  1  5 12  5
 14  5 12  2  5  2  6 12 12  5 10 12 12 12 12  1 12  6 13  1 12  6  1 12
 12 10 12  2 12  5  5 12 12  1 12  1 12  5  1 13  2 12  1 12 12 10  5  1
 12  1 12 13 12 12 12  5  5 10  1 12  1 12  5 12 12  1  6  5  6  1 12 12
 12  1 10  1  5  5  2  1  6 12  1 13  6  6  1 12 12  2 12 14 12 12 12 12
  1 12  5 12 12 10 12 13 12  1  1  6  1  1  1 12  5 12 12  7 12  5 12 12
 12  7  7 12 12  7  7 12  5 12  5  1 12 12 12 12 12  5 12 12 12 12 12 12
  1  6  5 12 12 12 12  5  6  1 12  2 12 12 12 12 12 12 12 12 12 12 12 12
 12 12  1 12 12  5 12 12 12 12  1 12 10 12  1  6 12  5 12 12 12 12 12 12
 12  1  5 12 12  1 12 12 12 12 12 12  1 12  5 12  1 12 12  1  5  7  1  5
 10  7 12  1  6  6  5 12  6  1  4 12 12  1 12 12 12  1 10 12  6 12 10 12
 12 12 12 12 12 12 12  5 12 12 12 12  1 12 12 12 12 12 12 12 12 12 12 12
 12 12 12 12 10  1 12  3 12  3 10 12 12 12 12 12 12 10 12 12 12 12 12 12
  6 12 12 12 12  4 12  1  5 12 12 12  1 12 12  5 12 12  5 12 12 10 12 12
 12 12 12 12 12 12 12 12 12 12  5 12 12  1 12 12 12 12 12 12 12 12 12  1
 12 12 12  6 12  1 12  1 12 12  1  1 12 12 12 12 12 12 12 12 12 12 12 12
 12 12 12 10 12 12 12 12  1 12 12 12 12 12 12  6 12 12 12 12 12 12 12  6
  1 12 12  5  1  5  5 12 12 12 12 12 12 12 12 12  5  6  5 12 12 12 12 12
 12 12 12 12  5 12 12  6 12 12 12 12 12 12 12 12 12  5 12 12 12 12 12 12
 12 12 12 12 12 12 12 12  6  1 12 12 12 12 12 12 12 12 12 12 12  1  1 12
 12 12  1 12  5 12 12  1  1 12 12  1 12 12  1 12 12 12 12 12 12  3 12 12
 12 12 12 10 12  5 12  1 12 10  5 12 10  3 12 12 12 12  5  3 12  4  1 12
  5  5 12  5 12  5 12 12 12 12  3 12  3  1 10  4  4 10  5 12 12 12  5  5]
Accuracy : 16.054%

/home/dharmveer/venv/lib/python3.6/site-packages/sklearn/utils/validation.py:72: DataConversionWarning:

A column-vector y was passed when a 1d array was expected. Please change the shape of y to (n_samples, ), for example using rav
el().
```

KNN algorithm

In [180]:

```
#applying knn algorithm
from sklearn.neighbors import KNeighborsClassifier
model = KNeighborsClassifier(n_neighbors=3)
classification_model(model, df,predictor_var,outcome_var)
```

```
[ 3  5  6  1  1  6  7  1  6  9  6  5  2  9  3  1  3  9  6  7  2  2  1  6
  9 14  2  5  2  6  1  5 14  6  9  2  1  9 14  1  9  2  1  5  3  6  9  5
  9  1  5  3  2  6  3  6  5  1  1  3  3  5  3  2 14  3  3  3 14  3  6 14
  1  5  3  1  6  6  1  3  6  3  5  1  5  6  2 14  3  5  1 14  5  1  7 14
  3  1 14  2  6  5  1 14  6  5  2  3  2  5  1  3 14  3  3  1  1  9  2 14
  6  9  5  1  3  7 14  6  1 14  5  1  3  9  1 14  9  3  6  1  2  1  5  7
  2  2  5  1  9  7  6  1  6  2  3  1 14  3  9  6 14  1  5  3  7 14  1  2
  5 14  1  1  5  2  1  5  6  3  1 13  2  6  1  9 13 14  2  1  5  9  7  2
  3 14  1  2  9  5  3  9  1  6  5  3  1 14  1  6  3  6  2  6  5  1 10  1
  5  1  2  6  9  5  1  3  9  5  9  9  2  7  9  3  1  9  9 13  9  5  2  1
 14  7  3  1  5  1  1  5  1  2  1  6  3  1  2  9  1  3  5  6  5  9  2 14
 13  2  6  1  6  3  2  7  5  3  5  7  1  2  6  6  1 10  7  5  2  7  1  5
 14  6  9  2  5  2  6  1  3  2  6  7  1  5  1  1  7  6  9  1  5  9  1  9
  3  6  9  2 14  6  1  1  5  1  2  1 11  6  1  9 10  2  1  5  3 13 10  1
  5  2  6  5  9  3  9  6  5 10  5  3  7  9  5  3  9  1  5  2  6  1  5 10
  6  1 13 10  7  5  2  2 10  9  1  6 10  6  1  5  9  2  3  1 10  2  5  9
  1  1  7  6  3 10  1  6  3 10  4  6  2  1  1  3  4  6  2  2  9  5  9  3
  2  5  3  3  9  3  5  5  2  2  5  1  6  3  6  5  1  2  2  6  9 10  5  2
  3  1  5  3 10  2  6  5  2  6  9 10  2  9  3 10  1  9  3  2  9  1  2  1
  4  9  1  5  5  5  3  6  9  3  6  2  6  6  1  6  2  6  1  7  6  2  3  7
  6  1  5  3 10  2  6  5  7  6  3 10  1  6  5  2  1  9  1  1  2 10  3  5
 10  3  3  1  2  9  6  3  1  1  3  5  5 11  2  8  3  1  8  7  2  1  9  3
 10  9  1  8  3  7 10  2  2  1 11  7  1  6  7 10  9  2  2  9  2 10 10  3
  9  4  8  3 11  1 10  4  3  2  9  1  3 11  9  8 10  7  8  2  7  2  2 10
 10 10  2  1  9  3 10  1  1 11  1  2  1  9  2  9  1  3  2 10  1 10  1  8
  1 11  9  3 11  2  3  2 10  2  1  9 10  3 11  7  2  7  9 11  1  2  8 10
  9  7  9 10  7  1 11  3 11  2  1  2  5  9  2 10  5  6 10  3  1  2  3  5
  2  1  2  9  5  2  3  5  2  7  1  3 10  7  1  6  2  5  2  6  2  1  5 10
  1  9 10  6  1 10  6  9  3  5  6  1  2  1  3  5  3  2  1  5  5  6  2  4
  5  2  1  6  4  2  2 10  1  4 10  5  9  6  5 10  2  5 10  3  2 10  5  9
  5  3  6  3  5  5  1  9 10  1  2  6  9  2  3  3  5  3  6  1  6  4  1 10
  3  5  1  9  4  3  9  1  1  4  1  5  2  3  1 10  4  2 10  1  4  1 10  1
  2  2  6  5  1  4  2  1  1  5  3  1  5  9  4  1 10  1  3  9 10  3  1  6
  5  1  2  1 10  4  3  1  2  6  1  4  1  1  9  4  3  2  4  1  1  2  9  1]
Accuracy : 64.093%

/home/dharmveer/venv/lib/python3.6/site-packages/ipykernel_launcher.py:10: DataConversionWarning:

A column-vector y was passed when a 1d array was expected. Please change the shape of y to (n_samples, ), for example using rav
el().
```

Linear regression

In [181]:
```
X = df.iloc[:, :-1].values
y = df.iloc[:, 1].values
```

In [182]:
```
X.shape
```

Out[182]:
```
(816, 6)
```

In [183]:
```
y.shape
```

Out[183]:
```
(816,)
```

In [184]:
```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.25, random_state = 45)
from sklearn.linear_model import LinearRegression
regressor = LinearRegression()
regressor.fit(X_train, y_train)
# Predicting the Test set results
y_pred = regressor.predict(X_test)
print('Coefficients: \n', regressor.coef_)
# The mean squared error
print("Mean squared error: %.2f" % np.mean((regressor.predict(X_test) -y_test)**2))
# Explained variance score: 1 is perfect prediction
print('Variance score: %.2f' % regressor.score(X_test, y_test))
```

```
Coefficients:
 [ 1.17226834e-15  1.00000000e+00 -5.55111512e-16  2.88657986e-15
  1.55431223e-15  3.88578059e-16]
Mean squared error: 0.00
Variance score: 1.00
```

In [ ]: