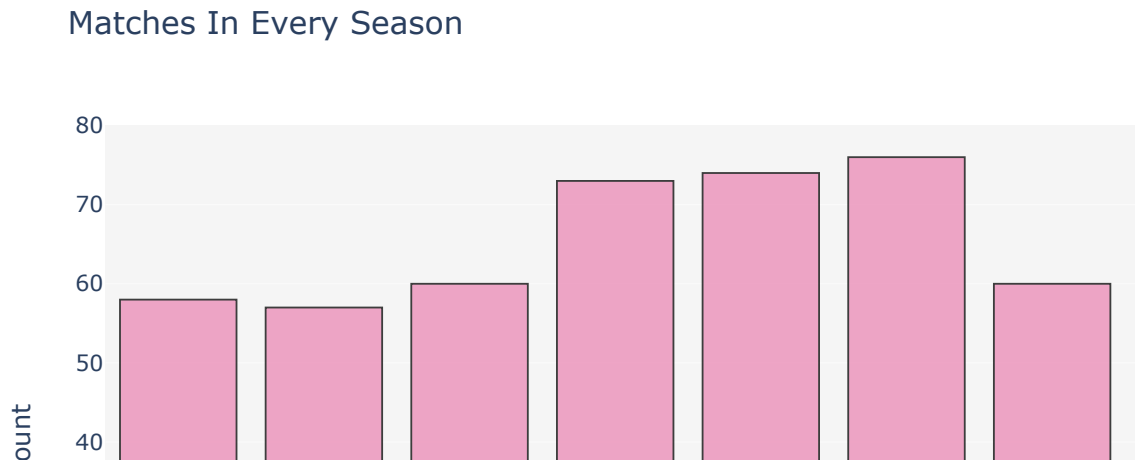
In [4]:
```
# Now let's start the task of IPL analysis with Python by importing the necessary Python lib
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import os

# Plotly to create interactive graph
import chart_studio.plotly as py
from plotly import tools
from plotly.offline import init_notebook_mode,iplot
init_notebook_mode(connected=False)
import plotly.figure_factory as ff
import plotly.graph_objs as go

%matplotlib inline
sns.set_style("whitegrid")
plt.style.use("fivethirtyeight")

# To remove un-necessary warnings
import warnings
warnings.filterwarnings("ignore")

deliveries = pd.read_csv("C:\\Users\\Anwar Alam\\Downloads\\archive\\IPL Ball-by-Ball 2008-2
matches = pd.read_csv("C:\\Users\\Anwar Alam\\Downloads\\archive\\IPL Matches 2008-2020.csv.
```

In [5]:
```
x=['Sunrisers Hyderabad', 'Mumbai Indians', 'Gujarat Lions',
    'Rising Pune Supergiant', 'Royal Challengers Bangalore',
    'Kolkata Knight Riders', 'Delhi Daredevils', 'Kings XI Punjab',
    'Chennai Super Kings', 'Rajasthan Royals', 'Deccan Chargers',
    'Kochi Tuskers Kerala', 'Pune Warriors', 'Rising Pune Supergiants', 'Delhi Capitals']

y = ['SRH','MI','GL','RPS','RCB','KKR','DC','KXIP','CSK','RR','SRH','KTK','PW','RPS','DC']

matches.replace(x,y,inplace = True)
deliveries.replace(x,y,inplace = True)
```

In [6]:
```
# Let's start with looking at the number of matches played in every season of the IPL:
matches['season'] = matches['date'].str[:4].astype(int)
data = [go.Histogram(x=matches['season'], marker=dict(color='#EB89B5', line=dict(color='#000
layout = go.Layout(title='Matches In Every Season ',xaxis=dict(title='Season',tickmode='line
                   yaxis=dict(title='Count'),bargap=0.2, plot_bgcolor='rgb(245,245,245)')

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

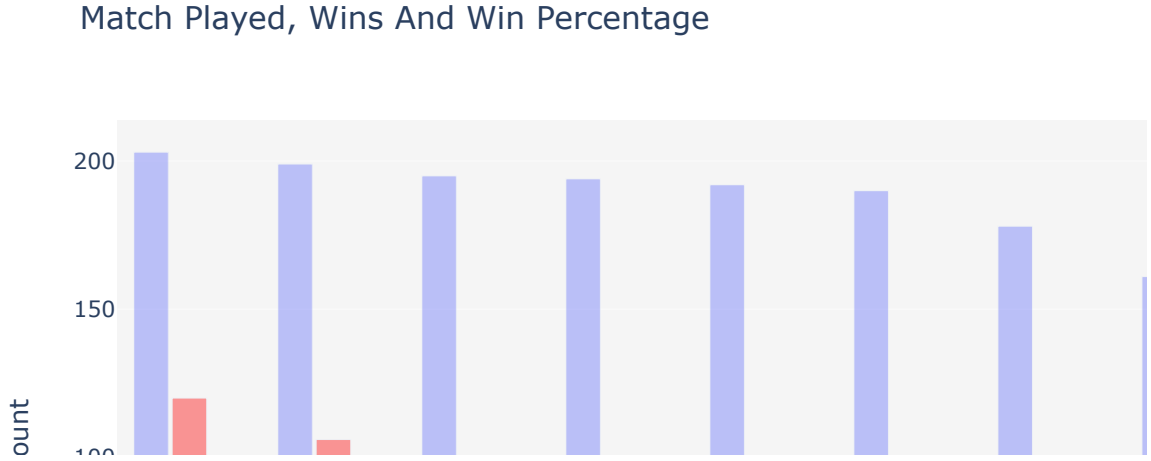


In [7]:
```
# Matches Played Vs Wins
matches_played=pd.concat([matches['team1'],matches['team2']])
matches_played=matches_played.value_counts().reset_index()
matches_played.columns=['Team','Total Matches']
matches_played['wins']=matches['winner'].value_counts().reset_index()['winner']

matches_played.set_index('Team',inplace=True)
totm = matches_played.reset_index().head(8)

trace = go.Table(
    header=dict(values=["Team","Total Matches","Wins"],
                fill = dict(color='#ff96ea'),
                font = dict(color=['rgb(45, 45, 45)'] * 5, size=14),
                align = ['center'],
               height = 30),
    cells=dict(values=[totm['Team'], totm['Total Matches'], totm['wins']],
               fill = dict(color=['rgb(235, 193, 238)', 'rgba(228, 222, 249, 0.65)']),
               align = ['center'], font_size=13, height=25))

layout = dict(
    width=750,
    height=420,
    autosize=False,
    title='Total Matches vs Wins per team',
    margin = dict(t=100),
    showlegend=False,
)

fig1 = dict(data=[trace], layout=layout)
iplot(fig1)
```

Total Matches vs Wins per team

Team	Total Matches	Wins
MI	203	120
SRH	199	106
RCB	195	99
DC	194	95
KKR	192	91
KXIP	190	88
CSK	178	86
RR	161	81

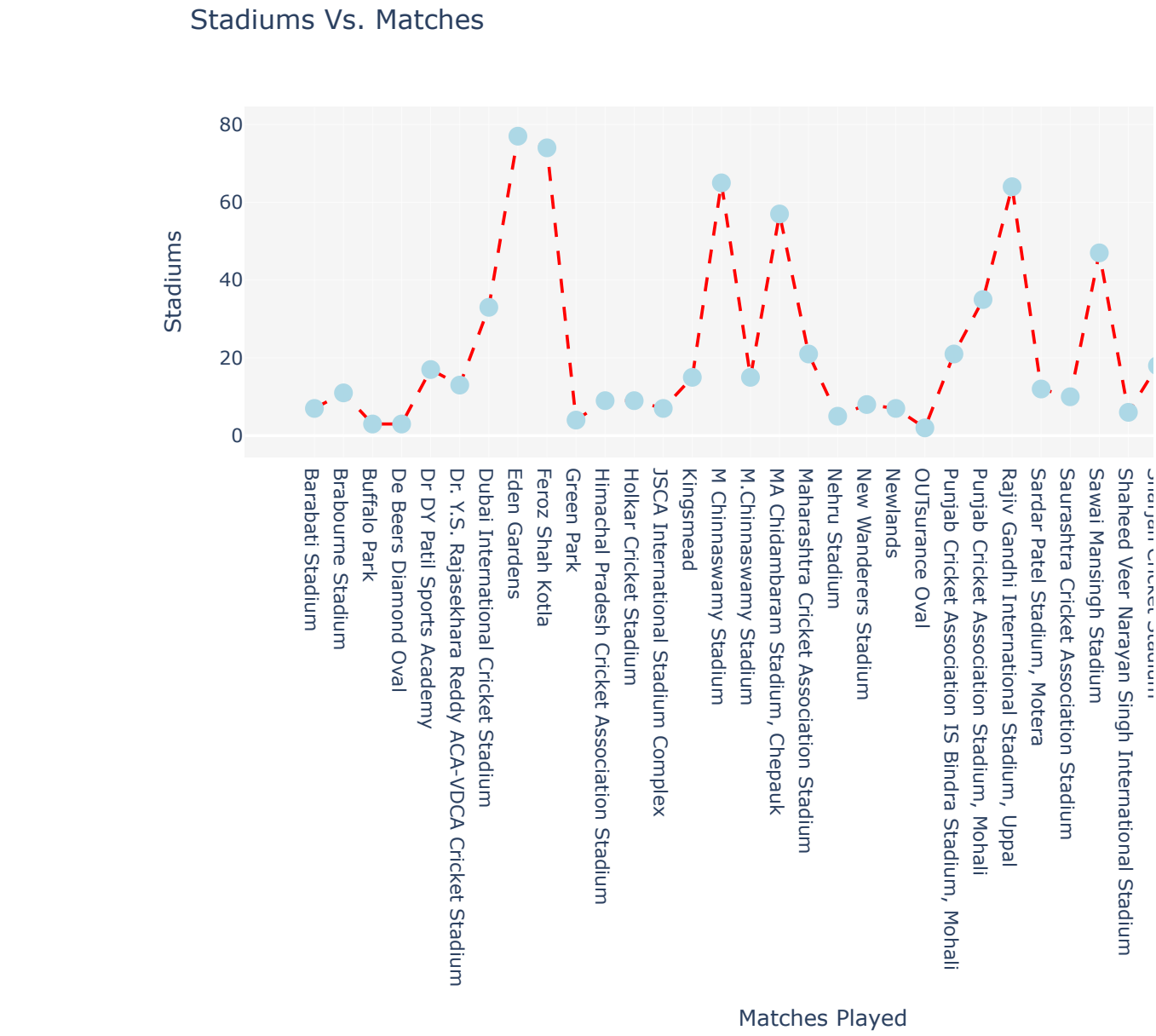
In [8]:
```
# Now let's analyze the winning percentage of all IPL teams:
trace1 = go.Bar(x=matches_played.index,y=matches_played['Total Matches'],
                name='Total Matches',opacity=0.4)

trace2 = go.Bar(x=matches_played.index,y=matches_played['wins'],
                name='Matches Won',marker=dict(color='red'),opacity=0.4)

trace3 = go.Bar(x=matches_played.index,
               y=(round(matches_played['wins']/matches_played['Total Matches'],3)*100),
               name='Win Percentage',opacity=0.6,marker=dict(color='gold'))

data = [trace1, trace2, trace3]

layout = go.Layout(title='Match Played, Wins And Win Percentage',xaxis=dict(title='Team'),
                   yaxis=dict(title='Count'),bargap=0.2,bargroupgap=0.1, plot_bgcolor='rgb(2

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



In [9]:
```
# So MI, SRH and RCB are the top three teams with the highest winning percentage. Let's
# look at the winning percentage of these three teams:
win_percentage = round(matches_played['wins']/matches_played['Total Matches'],3)*100
win_percentage.head(3)
```

Out[9]:
```
Team
MI     59.1
SRH    53.3
RCB    50.8
dtype: float64
```

In [10]:
```
# The next step in IPL analysis is to have a look at the venues where the most number of mat
venue_matches=matches.groupby('venue').count()[['id']].sort_values(by='id',ascending=False).
ser = pd.Series(venue_matches['id'])
venue_matches=matches.groupby('venue').count()[['id']].reset_index()

data = [{"y": venue_matches['id'],"x": venue_matches['venue'],
          "marker": {"color": "lightblue", "size": 12},
         "line": {"color": "red","width" : 2,"dash" : 'dash'},
          "mode": "markers+lines", "name": "Women", "type": "scatter"}]

layout = {"title": "Stadiums Vs. Matches",
          "xaxis": {"title": "Matches Played", },
          "yaxis": {"title": "Stadiums"},
          "autosize":False,"width":900,"height":700,"plot_bgcolor":"rgb(245,245,245)"}

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

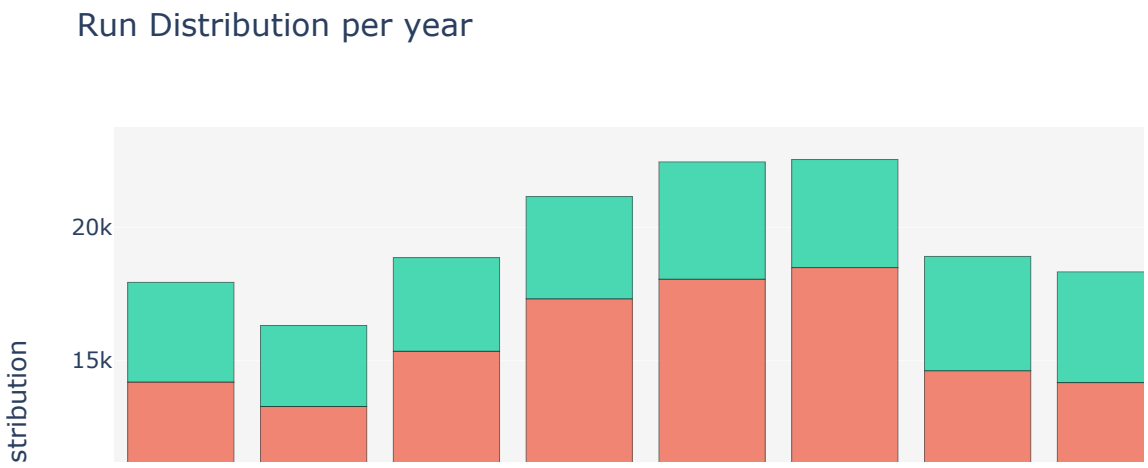


In [11]:
```
# So Eden Gardens, M Chinnaswamy, Wankhede and Feroz Shah Kotla are stadiums with most match
#eliminators, playoffs and finals were there. Now let's have a look the most prefered decisi
data = [go.Bar(
    x = matches["toss_decision"].value_counts().index,
    y = matches["toss_decision"].value_counts().values,
    marker = dict(line=dict(color='#000000', width=1))
)]

layout = go.Layout(
   {
      "title":"Most Likely Decision After Winning Toss",
       "xaxis":dict(title='Decision'),
       "yaxis":dict(title='Number of Matches'),
       "plot_bgcolor":'rgb(245,245,245)'
   }
)
fig = go.Figure(data=data,layout = layout)
iplot(fig)
```

In [13]:
```
batsmen = matches[['id','season']].merge(deliveries, left_on = 'id', right_on = 'id', how =
season=batsmen.groupby(['season'])['total_runs'].sum().reset_index()

avgruns_each_season=matches.groupby(['season']).count().id.reset_index()
avgruns_each_season.rename(columns={'id':'matches'},inplace=1)
avgruns_each_season['total_runs']=season['total_runs']
avgruns_each_season['average_runs_per_match']=avgruns_each_season['total_runs']/avgruns_each
```

In [14]:
```
 # Now let's have a look the distributions of runs over the years which will be distributed
Season_boundaries=batsmen.groupby("season")["batsman_runs"].agg(lambda x: (x==6).sum()).rese
fours=batsmen.groupby("season")["batsman_runs"].agg(lambda x: (x==4).sum()).reset_index()
Season_boundaries=Season_boundaries.merge(fours,left_on='season',right_on='season',how='left
Season_boundaries=Season_boundaries.rename(columns={'batsman_runs_x':'6"s','batsman_runs_y':

Season_boundaries['6"s'] = Season_boundaries['6"s']*6
Season_boundaries['4"s'] = Season_boundaries['4"s']*4
Season_boundaries['total_runs'] = season['total_runs']

trace1 = go.Bar(
    x=Season_boundaries['season'],
    y=Season_boundaries['total_runs']-(Season_boundaries['6"s']+Season_boundaries['4"s']),
    marker = dict(line=dict(color='#000000', width=1)),
    name='Remaining runs',opacity=0.6)

trace2 = go.Bar(
    x=Season_boundaries['season'],
    y=Season_boundaries['4"s'],
    marker = dict(line=dict(color='#000000', width=1)),
    name='Run by 4"s',opacity=0.7)

trace3 = go.Bar(
    x=Season_boundaries['season'],
    y=Season_boundaries['6"s'],
    marker = dict(line=dict(color='#000000', width=1)),
    name='Run by 6"s',opacity=0.7)
```

```
data = [trace1, trace2, trace3]
layout = go.Layout(title="Run Distribution per year",barmode='stack',xaxis = dict(tickmode='
                                    yaxis = dict(title= "Run Distribution"), plot_bgcolor='r

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



In [15]:
```
# We can see just a slight increase in runs by boundaries over the years. At last, we will l
high_scores=deliveries.groupby(['id', 'inning','batting_team','bowling_team'])['total_runs']
high_scores=high_scores[high_scores['total_runs']>=200]
hss = high_scores.nlargest(10,'total_runs')

trace = go.Table(
    header=dict(values=["Inning","Batting Team","Bowling Team", "Total Runs"],
                fill = dict(color = 'red'),
                font = dict(color = 'white', size = 14),
                align = ['center'],
               height = 30),
    cells=dict(values=[hss['inning'], hss['batting_team'], hss['bowling_team'], hss['total_r
               fill = dict(color = ['lightsalmon', 'rgb(245, 245, 249)']),
               align = ['center'], font_size=13))

layout = dict(
    width=830,
    height=410,
    autosize=False,
    title='Highest scores of IPL',
    showlegend=False,
)

fig1 = dict(data=[trace], layout=layout)
iplot(fig1)
```

Highest scores of IPL

Inning	Batting Team	Bowling Team	Total
1	RCB	PW	2
1	RCB	GL	2
1	CSK	RR	2
1	KKR	KXIP	2
1	CSK	KXIP	2
1	RCB	MI	2
1	KXIP	RCB	2
1	KKR	MI	2
1	DC	KXIP	2
1	KXIP	CSK	2

In [ ]: