```
import pandas as pd
import numpy as np
```

```
df = pd.read_csv("matches.csv")
```

```
df.head()
```

result	dl_applied	winner	win_by_runs	win_by_wickets	player_of_match	venue	umpire1	umpire2
normal	0	Sunrisers Hyderabad	35	0	Yuvraj Singh	Rajiv Gandhi International Stadium, Uppal	AY Dandekar	NJ Llong
normal	0	Rising Pune Supergiant	0	7	SPD Smith	Maharashtra Cricket Association Stadium	A Nand Kishore	S Ravi
normal	0	Kolkata Knight Riders	0	10	CA Lynn	Saurashtra Cricket Association Stadium	Nitin Menon	CK Nandan
normal	0	Kings XI Punjab	0	6	GJ Maxwell	Holkar Cricket Stadium	AK Chaudhary	C Shamshuddin
normal	0	Royal Challengers Bangalore	15	0	KM Jadhav	M Chinnaswamy Stadium	NaN	NaN

```
df.shape
```

```
(756, 18)
```

```
df.drop('umpire3' , axis = 1)
```

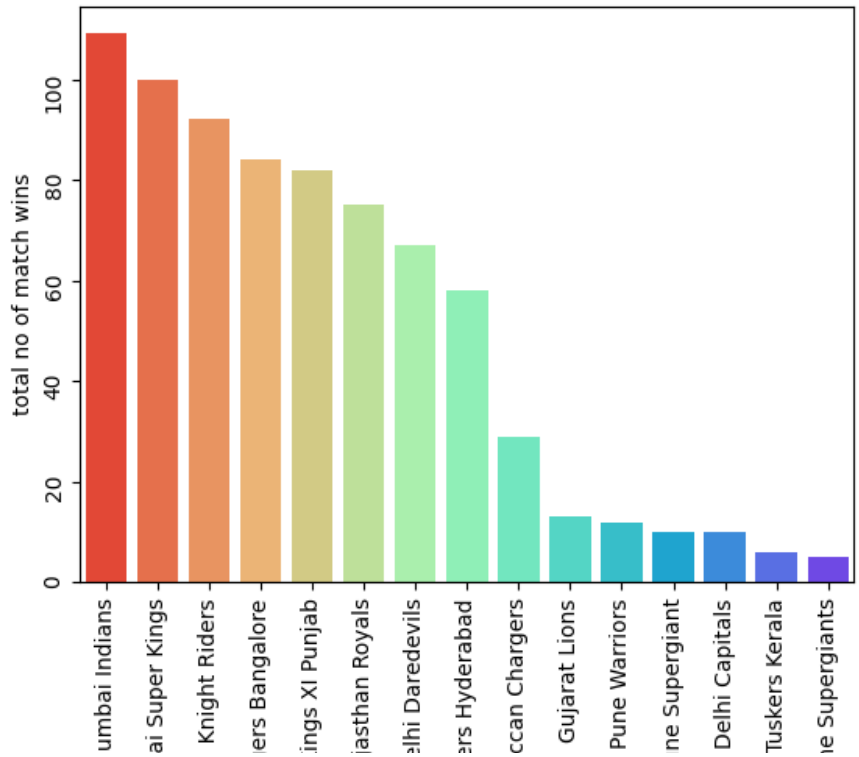
	id	Season	city	date	team1	team2	toss_winner	toss_decisi
0	1	IPL-2017	Hyderabad	05-04-2017	Sunrisers Hyderabad	Royal Challengers Bangalore	Royal Challengers Bangalore	fi
1	2	IPL-2017	Pune	06-04-2017	Mumbai Indians	Rising Pune Supergiant	Rising Pune Supergiant	fi
2	3	IPL-2017	Rajkot	07-04-2017	Gujarat Lions	Kolkata Knight Riders	Kolkata Knight Riders	fi
3	4	IPL-2017	Indore	08-04-2017	Rising Pune Supergiant	Kings XI Punjab	Kings XI Punjab	fi
4	5	IPL-2017	Bangalore	08-04-2017	Royal Challengers Bangalore	Delhi Daredevils	Royal Challengers Bangalore	
...	...	...	...	...	...	...	...	
				05-	Kolkata			

```
#most number of wins by team
df.winner.value_counts()
```

```
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
```

```
import matplotlib.pyplot as plt
import seaborn as sns


cntplot = sns.countplot( x = "winner" , data = df , order  = df.winner.value_counts().index , palette = "rainbow_r")

for c in cntplot.patches:
  print(c)


plt.tick_params(rotation = 90)
plt.ylabel("total no of match wins")
plt.show()
```

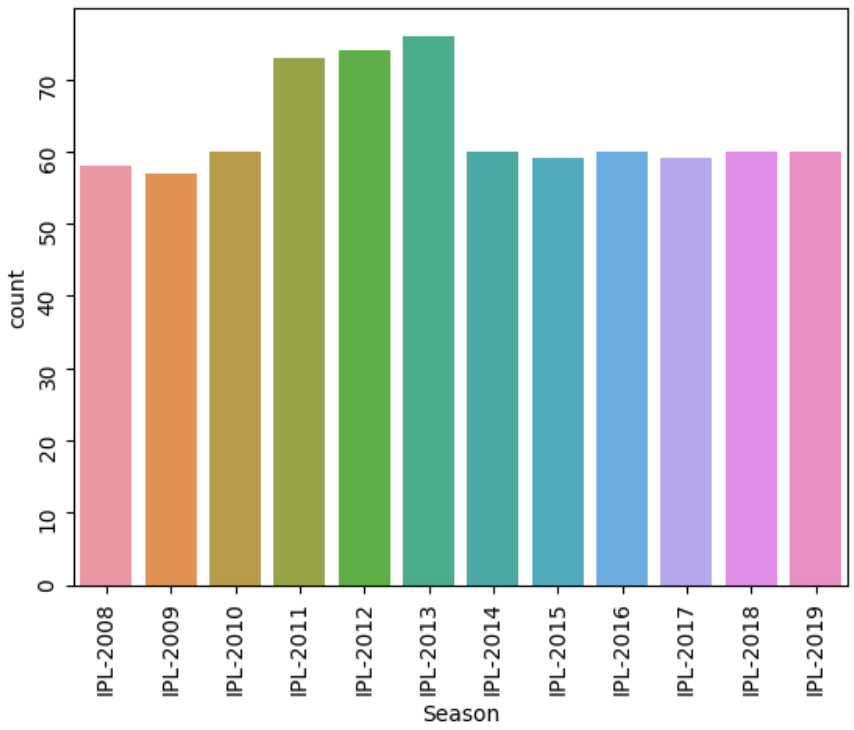
```
Rectangle(xy=(-0.4, 0), width=0.8, height=109, angle=0)
Rectangle(xy=(0.6, 0), width=0.8, height=100, angle=0)
Rectangle(xy=(1.6, 0), width=0.8, height=92, angle=0)
Rectangle(xy=(2.6, 0), width=0.8, height=84, angle=0)
Rectangle(xy=(3.6, 0), width=0.8, height=82, angle=0)
Rectangle(xy=(4.6, 0), width=0.8, height=75, angle=0)
Rectangle(xy=(5.6, 0), width=0.8, height=67, angle=0)
Rectangle(xy=(6.6, 0), width=0.8, height=58, angle=0)
Rectangle(xy=(7.6, 0), width=0.8, height=29, angle=0)
Rectangle(xy=(8.6, 0), width=0.8, height=13, angle=0)
Rectangle(xy=(9.6, 0), width=0.8, height=12, angle=0)
Rectangle(xy=(10.6, 0), width=0.8, height=10, angle=0)
Rectangle(xy=(11.6, 0), width=0.8, height=10, angle=0)
Rectangle(xy=(12.6, 0), width=0.8, height=6, angle=0)
Rectangle(xy=(13.6, 0), width=0.8, height=5, angle=0)
```



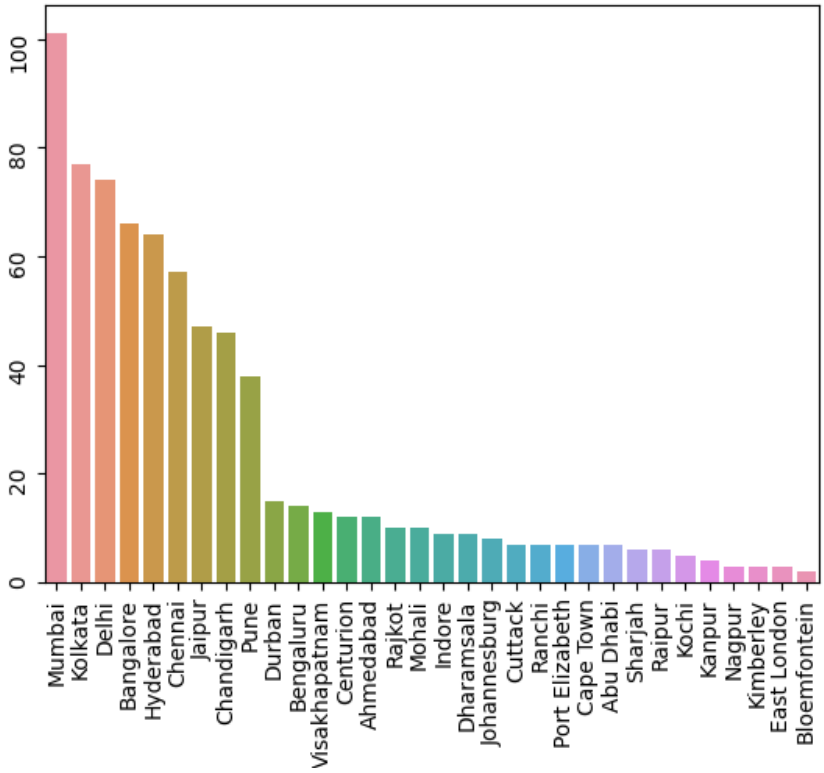

```
df1 = df.sort_values(by ="Season")
df1
```

	id	Season	city	date	team1	team2	toss_winner	toss_decision	result	dl_applied	winne
116	117	IPL-2008	Mumbai	01-06-2008	Chennai Super Kings	Rajasthan Royals	Rajasthan Royals	field	normal	0	Rajastha Royal
82	83	IPL-2008	Jaipur	04-05-2008	Chennai Super Kings	Rajasthan Royals	Chennai Super Kings	bat	normal	0	Rajastha Royal
81	82	IPL-2008	Mumbai	04-05-2008	Mumbai Indians	Delhi Daredevils	Delhi Daredevils	field	normal	0	Mumba Indian
80	81	IPL-2008	Chandigarh	03-05-2008	Kings XI Punjab	Kolkata Knight Riders	Kings XI Punjab	bat	normal	0	Kings X Punja
79	80	IPL-2008	Hyderabad	25-05-2008	Deccan Chargers	Royal Challengers Bangalore	Deccan Chargers	bat	normal	0	Roya Challenger Bangalor
...	...	...	...	...	...	...	...	...	...	...	.
721	11317	IPL-	Kolkata	12-04-	Kolkata Knight	Delhi	Delhi	field	normal	0	Dell

```
sns.countplot(x = 'Season'  , data = df1 )
plt.tick_params(rotation = 90
               )
plt.show()
```




```
x = df.city.value_counts().index
y =df.city.value_counts().values
sns.barplot(x = x , y = y , data = df )
plt.tick_params(rotation = 90)
plt.show()
```




```
# filtering out the un necassary teams

df['team1'].unique()
```

```
array(['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'], dtype=object)
```

```
teams = [

    'Sunrisers Hyderabad', 'Mumbai Indians', 'Royal Challengers Bangalore',
       'Kolkata Knight Riders', 'Kings XI Punjab',
       'Chennai Super Kings', 'Rajasthan Royals',
        'Rising Pune Supergiants','Delhi Capitals'

]


df['team1'] = df['team1'].str.replace('Delhi Daredevils' , 'Delhi Capitals')
df['team2'] = df['team1'].str.replace('Delhi Daredevils' , 'Delhi Capitals')


df['team1'] = df['team1'].str.replace('Deccan Chargers' , 'Sunrisers Hyderabad')
df['team2'] = df['team1'].str.replace('Deccan Chargers' , 'Sunrisers Hyderabad')




df = df[df['team1'].isin(teams)]
df = df[df['team2'].isin(teams)]


df.head()
```

	id	Season	city	date	team1	team2	toss_winner	toss_decision	resu
0	1	IPL-2017	Hyderabad	05-04-2017	Sunrisers Hyderabad	Sunrisers Hyderabad	Royal Challengers Bangalore	field	norm
1	2	IPL-2017	Pune	06-04-2017	Mumbai Indians	Mumbai Indians	Rising Pune Supergiant	field	norm
4	5	IPL-2017	Bangalore	08-04-2017	Royal Challengers Bangalore	Royal Challengers Bangalore	Royal Challengers Bangalore	bat	norm
6	7	IPL-2017	Mumbai	09-04-2017	Kolkata Knight Riders	Kolkata Knight Riders	Mumbai Indians	field	norm
7	8	IPL-2017	Indore	10-04-2017	Royal Challengers Bangalore	Royal Challengers Bangalore	Royal Challengers Bangalore	bat	norm

```
df.shape
```

```
(707, 18)
```

```
df['dl_applied'].value_counts()
```

```
0    688
1     19
Name: dl_applied, dtype: int64
```

```
df = df[df['dl_applied'] == 0 ]
```

```
df.head()
```

inner	toss_decision	result	dl_applied	winner	win_by_runs	win_by_wickets	player_of_match	venue	un
Royal engers galore	field	normal	0	Sunrisers Hyderabad	35	0	Yuvraj Singh	Rajiv Gandhi International Stadium, Uppal	Da
Pune ergiant	field	normal	0	Rising Pune Supergiant	0	7	SPD Smith	Maharashtra Cricket Association Stadium	A K
Royal engers galore	bat	normal	0	Royal Challengers Bangalore	15	0	KM Jadhav	M Chinnaswamy Stadium	
umbai ndians	field	normal	0	Mumbai Indians	0	4	N Rana	Wankhede Stadium	
Royal engers galore	bat	normal	0	Kings XI Punjab	0	8	AR Patel	Holkar Cricket Stadium	Cha

```
match_df = df[['Season', 'team1' , 'team2' ,  'city'  , 'player_of_match' , 'venue' , 'winner']]
match_df.head()
```

	Season	team1	team2	city	player_of_match	venue	winner
0	IPL-2017	Sunrisers Hyderabad	Sunrisers Hyderabad	Hyderabad	Yuvraj Singh	Rajiv Gandhi International Stadium, Uppal	Sunrisers Hyderabad
1	IPL-2017	Mumbai Indians	Mumbai Indians	Pune	SPD Smith	Maharashtra Cricket Association Stadium	Rising Pune Supergiant
4	IPL-2017	Royal Challengers Bangalore	Royal Challengers Bangalore	Bangalore	KM Jadhav	M Chinnaswamy Stadium	Royal Challengers Bangalore
6	IPL-2017	Kolkata Knight Riders	Kolkata Knight Riders	Mumbai	N Rana	Wankhede Stadium	Mumbai Indians
7	IPL-2017	Royal Challengers Bangalore	Royal Challengers Bangalore	Indore	AR Patel	Holkar Cricket Stadium	Kings XI Punjab

```
def res(row):
  return 1 if row['team1'] == row['winner'] else 0


match_df['result'] = match_df.apply(res , axis = 1  )
match_df
```

```
<ipython-input-67-f6f46508895f>:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#return
  match_df['result'] = match_df.apply(res , axis = 1  )
```

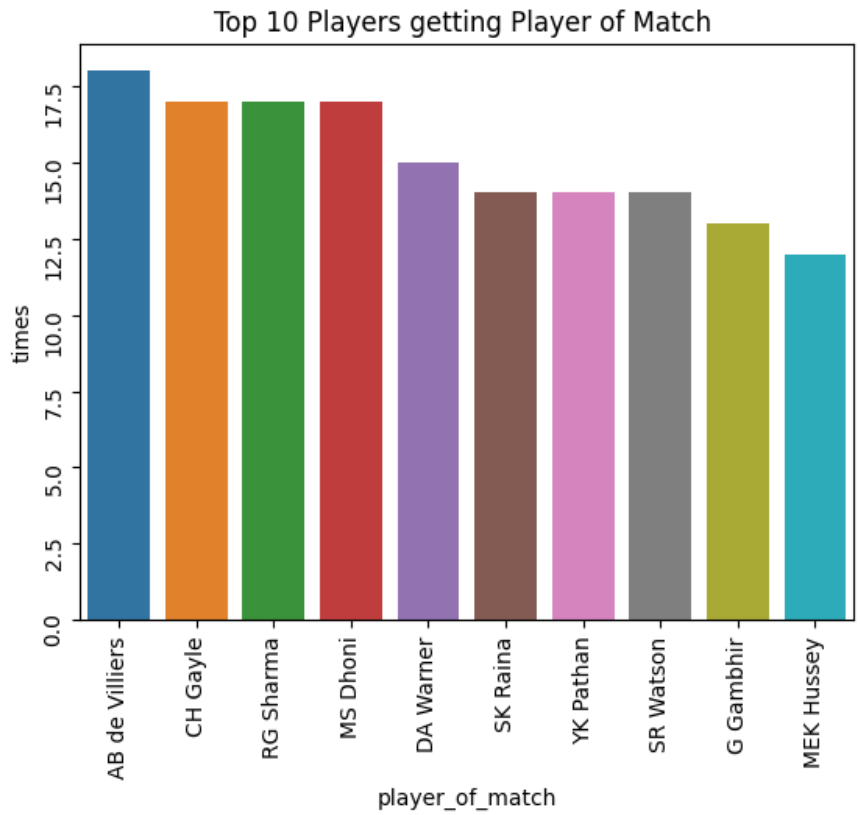
```
match_df.shape
```

```
(688, 8)
```

```
match_df = match_df.sample(match_df.shape[0]
```

```
#players of match most

# top 10 players

player = match_df.player_of_match.value_counts().reset_index()

player = player.head(10)
player.columns = ["player_of_match" , "times"]
player.head()
```

```
sns.barplot(x = "player_of_match" , y = "times" ,  data = player)
plt.tick_params(rotation = 90)
plt.title("Top 10 Players getting Player of Match ")
plt.show()
```



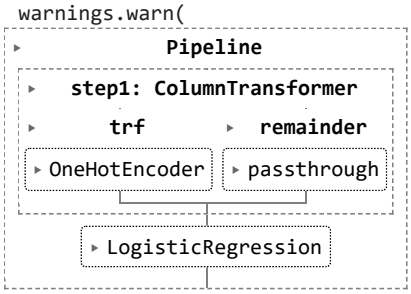

```
match_df = match_df.drop("player_of_match" , axis = 1)
```

```
match_df.sample()
```

	Season	team1	team2	city	venue	winner	result
457	IPL-2014	Kolkata Knight Riders	Kolkata Knight Riders	Abu Dhabi	Sheikh Zayed Stadium	Kolkata Knight Riders	1

```
X = match_df.iloc[: , :-1]
Y = match_df.iloc[: , -1]

from sklearn.model_selection import train_test_split
x_tr , x_ts , y_tr , y_ts = train_test_split(X , Y , test_size = 0.2 , random_state = 1)


# preprocessing the data values
from sklearn.compose import ColumnTransformer
from sklearn.preprocessing import OneHotEncoder
trf = ColumnTransformer ([
    ('trf' , OneHotEncoder(sparse = False , drop="first") , ['Season' , 'team1' , 'team2' , 'city' , 'venue' , 'winner']
  ]
, remainder= 'passthrough')


from  sklearn.linear_model import LogisticRegression
from sklearn.pipeline import Pipeline




pipe = Pipeline(steps = [
    ('step1' , trf),
    ('step2' , LogisticRegression(solver ='liblinear'))

])


pipe.fit(x_tr, y_tr)
```

```
/usr/local/lib/python3.10/dist-packages/sklearn/preprocessing/_encoders.py:868: FutureWarning: `sparse` was rename
  warnings.warn(
```




```
y_pred = pipe.predict(x_ts)
y_pred
```

```
array([0, 1, 1, 0, 0, 1, 0, 0, 1, 0, 0, 0, 1, 1, 1, 0, 0, 0, 1, 0, 1, 1,
       0, 1, 1, 0, 0, 0, 1, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0,
       1, 1, 0, 0, 0, 0, 0, 0, 1, 0, 1, 1, 1, 0, 0, 1, 0, 0, 0, 1, 1, 0,
       0, 1, 0, 1, 0, 0, 0, 1, 1, 0, 0, 1, 0, 1, 0, 0, 0, 1, 0, 0, 0, 1,
       1, 1, 0, 0, 1, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       1, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0,
       1, 0, 0, 1, 0, 0])
```

```
from sklearn.metrics import accuracy_score

accuracy_score(y_ts , y_pred)
```

```
0.6231884057971014
```

```
pipe.predict_proba(x_ts)[9]
```

```
array([0.70790385, 0.29209615])
```