Introduction to Data Science: Football Worldcup Data Analysis

Ishika Maheshwari - 20070122058 - CS Siddharth Khatri - 20070124032 - IT Shikhar Srivastava - 20070124032 - IT Saurabh Jain - 20070124050 - IT Akash Singh - 20070124055 - IT

In [1]:
```
# Importing all the modules required
import numpy as np                    # numpy for linear algebra operations
import pandas as pd                   # pandas for datafile processing
import matplotlib.pyplot as plt       # matplotlib for data visualization
import seaborn as sns                 # also for data visualation (graphs e
%matplotlib inline
import plotly as py                   # for data visualisation (it supports
import cufflinks as cf                # it connects plotly with pandas so t
```

In [2]:
```
# Importing the datasets
players = pd.read_csv("WorldCupPlayers.csv")
matches = pd.read_csv("WorldCupMatches.csv")
world_cup = pd.read_csv("WorldCups.csv")
```

In [3]:
```
#First 5 rows of the players dataset
players.head()

# If we want, we can specify a number to get the first 'n' rows as well.
# Example, players.head(20)
```

Out[3]:

	RoundID	MatchID	Team Initials	Coach Name	Line-up	Shirt Number	Player Name	Position	Event
0	201	1096	FRA	CAUDRON Raoul (FRA)	S	0	Alex THEPOT	GK	NaN
1	201	1096	MEX	LUQUE Juan (MEX)	S	0	Oscar BONFIGLIO	GK	NaN
2	201	1096	FRA	CAUDRON Raoul (FRA)	S	0	Marcel LANGILLER	NaN	G40'
3	201	1096	MEX	LUQUE Juan (MEX)	S	0	Juan CARRENO	NaN	G70'
4	201	1096	FRA	CAUDRON Raoul (FRA)	S	0	Ernest LIBERATI	NaN	NaN

In [4]:
```
# First 5 rows of matches dataset
matches.head()
```

Out[4]:

	Year	Datetime	Stage	Stadium	City	Home Team Name	Home Team Goals	Away Team Goals	Away Team Name	con
0	1930.0	13 Jul 1930 - 15:00	Group 1	Pocitos	Montevideo	France	4.0	1.0	Mexico	
1	1930.0	13 Jul 1930 - 15:00	Group 4	Parque Central	Montevideo	USA	3.0	0.0	Belgium	
2	1930.0	14 Jul 1930 - 12:45	Group 2	Parque Central	Montevideo	Yugoslavia	2.0	1.0	Brazil	
3	1930.0	14 Jul 1930 - 14:50	Group 3	Pocitos	Montevideo	Romania	3.0	1.0	Peru	
4	1930.0	15 Jul 1930 - 16:00	Group 1	Parque Central	Montevideo	Argentina	1.0	0.0	France	

In [5]:
```
# The last 5 entries in the matches dataset

matches.tail()
```

Out[5]:

	Year	Datetime	Stage	Stadium	City	Home Team Name	Home Team Goals	Away Team Goals	Away Team Name	Win conditions	At
4567	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
4568	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
4569	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
4570	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
4571	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	

In [6]:
```
# First 5 rows of worldcup dataset

world_cup.head()
```

Out[6]:

	Year	Country	Winner	Runners-Up	Third	Fourth	GoalsScored	Qualifie
0	1930	Uruguay	Uruguay	Argentina	USA	Yugoslavia	70	
1	1934	Italy	Italy	Czechoslovakia	Germany	Austria	70	
2	1938	France	Italy	Hungary	Brazil	Sweden	84	
3	1950	Brazil	Uruguay	Brazil	Sweden	Spain	88	
4	1954	Switzerland	Germany FR	Hungary	Austria	Uruguay	140	

Data Cleaning

Our datasets contain various entries in which there is no data. So, we will remove (clean) those rows which contain NULL values.

In [7]:
```python
matches.dropna(subset=['Year'], inplace=True)

# It removes all entries where the 'Year' column had NULL values
```

In [8]:
```python
matches.tail()

# You can see the dataset after data cleaning
```

Out[8]:

	Year	Datetime	Stage	Stadium	City	Home Team Name	Home Team Goals	Away Team Goals	Away Te Na
847	2014.0	05 Jul 2014 - 17:00	Quarter-finals	Arena Fonte Nova	Salvador	Netherlands	0.0	0.0	Costa I
848	2014.0	08 Jul 2014 - 17:00	Semi-finals	Estadio Mineirao	Belo Horizonte	Brazil	1.0	7.0	Germ
849	2014.0	09 Jul 2014 - 17:00	Semi-finals	Arena de Sao Paulo	Sao Paulo	Netherlands	0.0	0.0	Argen
850	2014.0	12 Jul 2014 - 17:00	Play-off for third place	Estadio Nacional	Brasilia	Brazil	0.0	3.0	Netherla
851	2014.0	13 Jul 2014 - 16:00	Final	Estadio do Maracana	Rio De Janeiro	Germany	1.0	0.0	Argen

In [9]:
```python
# Getting the value count in a dataset (meaning how many time a specific
matches['Home Team Name'].value_counts()
```

```
Out[9]: Brazil                          82
        Italy                           57
        Argentina                       54
        Germany FR                      43
        England                         35
                                        ..
        Wales                            1
        Norway                           1
        rn">United Arab Emirates         1
        Haiti                            1
        rn">Bosnia and Herzegovina       1
        Name: Home Team Name, Length: 78, dtype: int64
```

```
In [10]: matches['Home Team Goals'].value_counts()
```

```
Out[10]: 1.0     246
         2.0     205
         0.0     177
         3.0     116
         4.0      59
         6.0      17
         5.0      17
         7.0       8
         8.0       4
         9.0       2
         10.0      1
         Name: Home Team Goals, dtype: int64
```

```
In [11]: # Some Home Teams include unrequired characters.

         names = matches[matches['Home Team Name'].str.contains('rn">')]['Home Tea

         names
```

```
Out[11]: rn">Republic of Ireland       5
         rn">United Arab Emirates      1
         rn">Trinidad and Tobago       1
         rn">Serbia and Montenegro     1
         rn">Bosnia and Herzegovina    1
         Name: Home Team Name, dtype: int64
```

```
In [12]: # Creating a new list with wrong names of the Home Teams (they include th

         wrong = list(names.index)

         wrong
```

```
Out[12]: ['rn">Republic of Ireland',
          'rn">United Arab Emirates',
          'rn">Trinidad and Tobago',
          'rn">Serbia and Montenegro',
          'rn">Bosnia and Herzegovina']
```

```
In [13]: # Removing the 'rn">' from the names. We do not require it.

         correct = [name.split('>')[1] for name in wrong]

         correct

Out[13]: ['Republic of Ireland',
          'United Arab Emirates',
          'Trinidad and Tobago',
          'Serbia and Montenegro',
          'Bosnia and Herzegovina']

In [14]: old_name = ['Germany FR', 'Maracan� - Est�dio Jornalista M�rio Filho', 'E
         new_name = ['Germany', 'Maracan Stadium', 'Maracan Stadium'] # These are

In [15]: wrong = wrong + old_name #Adding oldname to the wrong names list
         correct = correct + new_name #Adding newname to the correct names list

In [16]: wrong # Displaying the updated wrong team names list

Out[16]: ['rn">Republic of Ireland',
          'rn">United Arab Emirates',
          'rn">Trinidad and Tobago',
          'rn">Serbia and Montenegro',
          'rn">Bosnia and Herzegovina',
          'Germany FR',
          'Maracan� - Est�dio Jornalista M�rio Filho',
          'Estadio do Maracana']

In [17]: correct # Displaying the updated correct team names list

Out[17]: ['Republic of Ireland',
          'United Arab Emirates',
          'Trinidad and Tobago',
          'Serbia and Montenegro',
          'Bosnia and Herzegovina',
          'Germany',
          'Maracan Stadium',
          'Maracan Stadium']

In [18]: # Replacing wrong values with corrected ones in players, matches and worl

         for index, wr in enumerate(wrong):
             world_cup = world_cup.replace(wrong[index], correct[index])

         for index, wr in enumerate(wrong):
             matches = matches.replace(wrong[index], correct[index])

         for index, wr in enumerate(wrong):
             players = players.replace(wrong[index], correct[index])
```

In [19]:
```
# Checking if there are still any wrong entries listed in the dataset

names = matches[matches['Home Team Name'].str.contains('rn">')]['Home Tea
names
```

Out[19]:
```
Series([], Name: Home Team Name, dtype: int64)
```

Most Number of World Cup Winning Title

In [20]:
```
# Counting the number of times each country has appeared as the 'Winner'

winner = world_cup['Winner'].value_counts()

winner
```

Out[20]:
```
Brazil       5
Italy        4
Germany      4
Uruguay      2
Argentina    2
England      1
France       1
Spain        1
Name: Winner, dtype: int64
```

In [21]:
```
# Counting the number of times each country has appeared as the 'Runner U

runnerup = world_cup['Runners-Up'].value_counts()

runnerup
```

Out[21]:
```
Germany           4
Argentina         3
Netherlands       3
Czechoslovakia    2
Hungary           2
Brazil            2
Italy             2
Sweden            1
France            1
Name: Runners-Up, dtype: int64
```

In [22]:
```
# Counting the number of times each country has appeared as the 'Third' i

third = world_cup['Third'].value_counts()

third
```

Out[22]:
```
Germany        4
Brazil         2
Sweden         2
France         2
Poland         2
USA            1
Austria        1
Chile          1
Portugal       1
Italy          1
Croatia        1
Turkey         1
Netherlands    1
Name: Third, dtype: int64
```

In [23]:
```
# Creating a new table listing the all the above values

teams = pd.concat([winner, runnerup, third], axis=1)     # Concating the
teams.fillna(0, inplace=True)   # Replaces the NULL values with a 0
teams = teams.astype(int)       # Converts all dtypes to int

teams
```

```
Out[23]:
```

	Winner	Runners-Up	Third
Brazil	5	2	2
Italy	4	2	1
Germany	4	4	4
Uruguay	2	0	0
Argentina	2	3	0
England	1	0	0
France	1	1	2
Spain	1	0	0
Netherlands	0	3	1
Czechoslovakia	0	2	0
Hungary	0	2	0
Sweden	0	1	2
Poland	0	0	2
USA	0	0	1
Austria	0	0	1
Chile	0	0	1
Portugal	0	0	1
Croatia	0	0	1
Turkey	0	0	1

In [24]:
```
# Importing iplot from plotly and connecting it to offline mode

from plotly.offline import iplot
py.offline.init_notebook_mode(connected=True)
cf.go_offline()
```

In [25]:
```
teams.iplot(kind = 'bar', xTitle='Teams', yTitle='World Cup Winning Count
```

Number of Goal Per Countary

In [26]:
```
matches.head(2)
```

Out[26]:

	Year	Datetime	Stage	Stadium	City	Home Team Name	Home Team Goals	Away Team Goals	Away Team Name	conditi
0	1930.0	13 Jul 1930 - 15:00	Group 1	Pocitos	Montevideo	France	4.0	1.0	Mexico	
1	1930.0	13 Jul 1930 - 15:00	Group 4	Parque Central	Montevideo	USA	3.0	0.0	Belgium	

In [27]:
```
# Cleaning the NULL fields from Teams and Goals and storing output to new

home = matches[['Home Team Name', 'Home Team Goals']].dropna()
away = matches[['Away Team Name', 'Away Team Goals']].dropna()
```

In [28]:
```
# Setting the columns as 'Countries' and 'Goals' for home and away teams

home.columns = ['Countries', 'Goals']
away.columns = home.columns
```

In [29]:
```
# Creating a new list combining both values from home and away

goals = home.append(away, ignore_index = True)
```

```
/var/folders/ly/dlqmsw2s4f7gpzxx893rqbm40000gn/T/ipykernel_99227/1555894
462.py:3: FutureWarning:

The frame.append method is deprecated and will be removed from pandas in
a future version. Use pandas.concat instead.
```

In [30]:
```
# Grouping the countries by total goals they have scored (as home teams +

goals = goals.groupby('Countries').sum()
goals
```

Out[30]:

	Goals
Countries	
Algeria	14.0
Angola	1.0
Argentina	133.0
Australia	11.0
Austria	43.0
...	...
United Arab Emirates	2.0
Uruguay	80.0
Wales	4.0
Yugoslavia	60.0
Zaire	0.0

82 rows × 1 columns

In [31]:
```
# Sorting the list in descending order (most to less goals)

goals = goals.sort_values(by = 'Goals', ascending=False)

goals
```

Out[31]:

Countries	**Goals**
Germany	235.0
Brazil	225.0
Argentina	133.0
Italy	128.0
France	108.0
...	...
Trinidad and Tobago	0.0
Canada	0.0
China PR	0.0
Dutch East Indies	0.0
Zaire	0.0

82 rows × 1 columns

In [32]:
```
# Plotting graph for the most number of goals by countries
# Note that we are only displaying the top 30 countries from the list

goals[:30].iplot(kind='bar', xTitle = 'Country', yTitle = 'Number of Goal
```

Attendance, Number of Teams, Goals, and Matches per Cup

In [33]:
```
# Removing the decimals from the attendance column

world_cup['Attendance'] = world_cup['Attendance'].str.replace(".", "")
```

```
/var/folders/ly/dlqmsw2s4f7gpzxx893rqbm40000gn/T/ipykernel_99227/2667348
274.py:3: FutureWarning:

The default value of regex will change from True to False in a future ve
rsion. In addition, single character regular expressions will *not* be t
reated as literal strings when regex=True.
```

In [34]:
```
world_cup.head(10)
```

Out[34]:

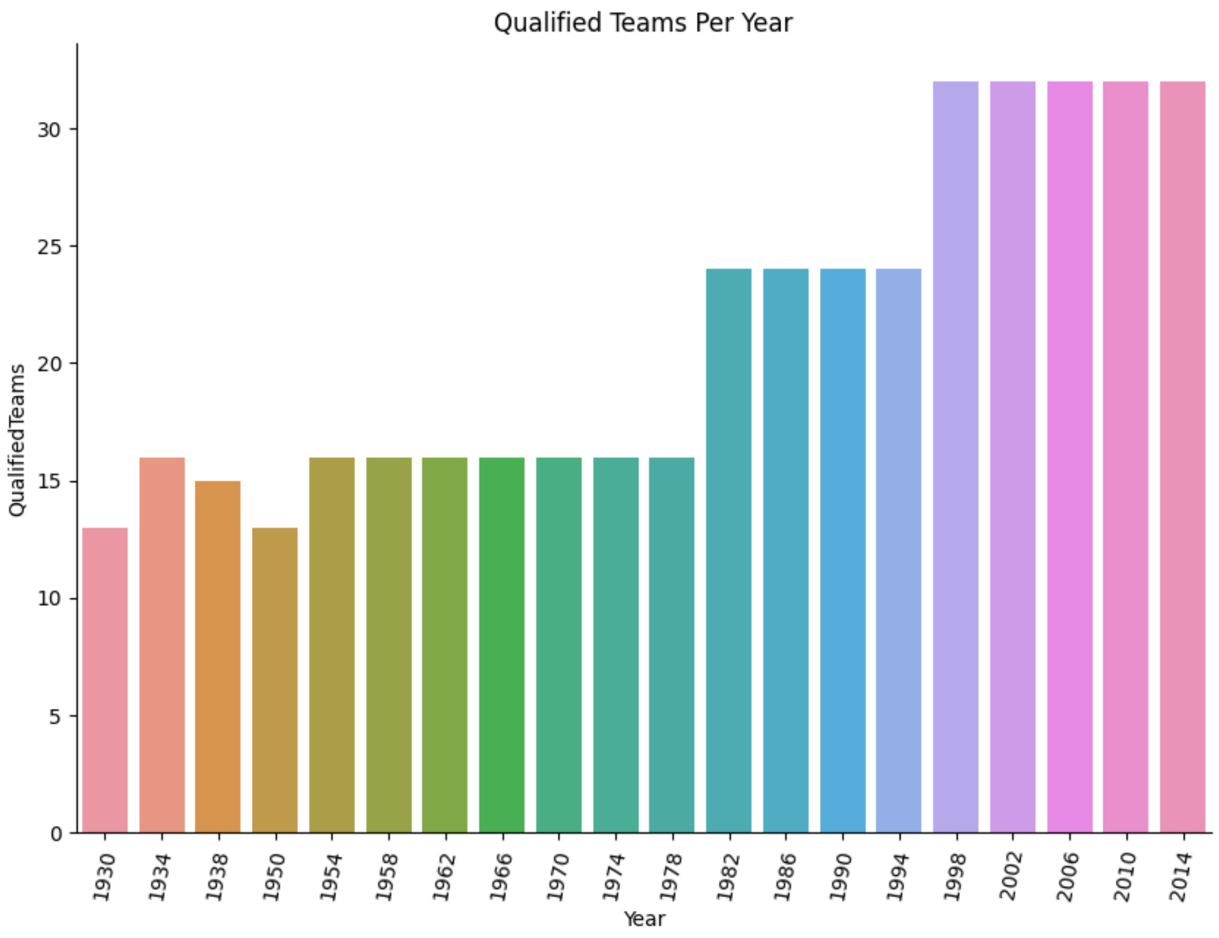
	Year	Country	Winner	Runners-Up	Third	Fourth	GoalsScored	Qualifie
0	1930	Uruguay	Uruguay	Argentina	USA	Yugoslavia	70	
1	1934	Italy	Italy	Czechoslovakia	Germany	Austria	70	
2	1938	France	Italy	Hungary	Brazil	Sweden	84	
3	1950	Brazil	Uruguay	Brazil	Sweden	Spain	88	
4	1954	Switzerland	Germany	Hungary	Austria	Uruguay	140	
5	1958	Sweden	Brazil	Sweden	France	Germany	126	
6	1962	Chile	Brazil	Czechoslovakia	Chile	Yugoslavia	89	
7	1966	England	England	Germany	Portugal	Soviet Union	89	
8	1970	Mexico	Brazil	Italy	Germany	Uruguay	95	
9	1974	Germany	Germany	Netherlands	Poland	Brazil	97	

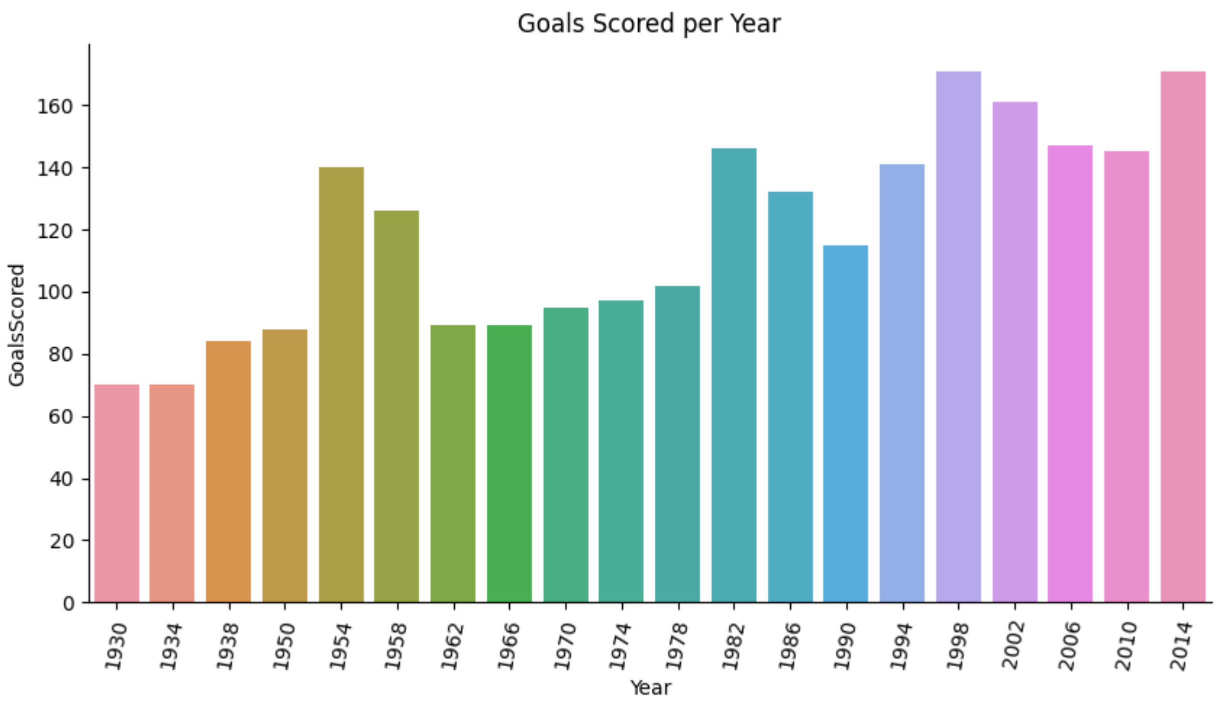
In [41]:

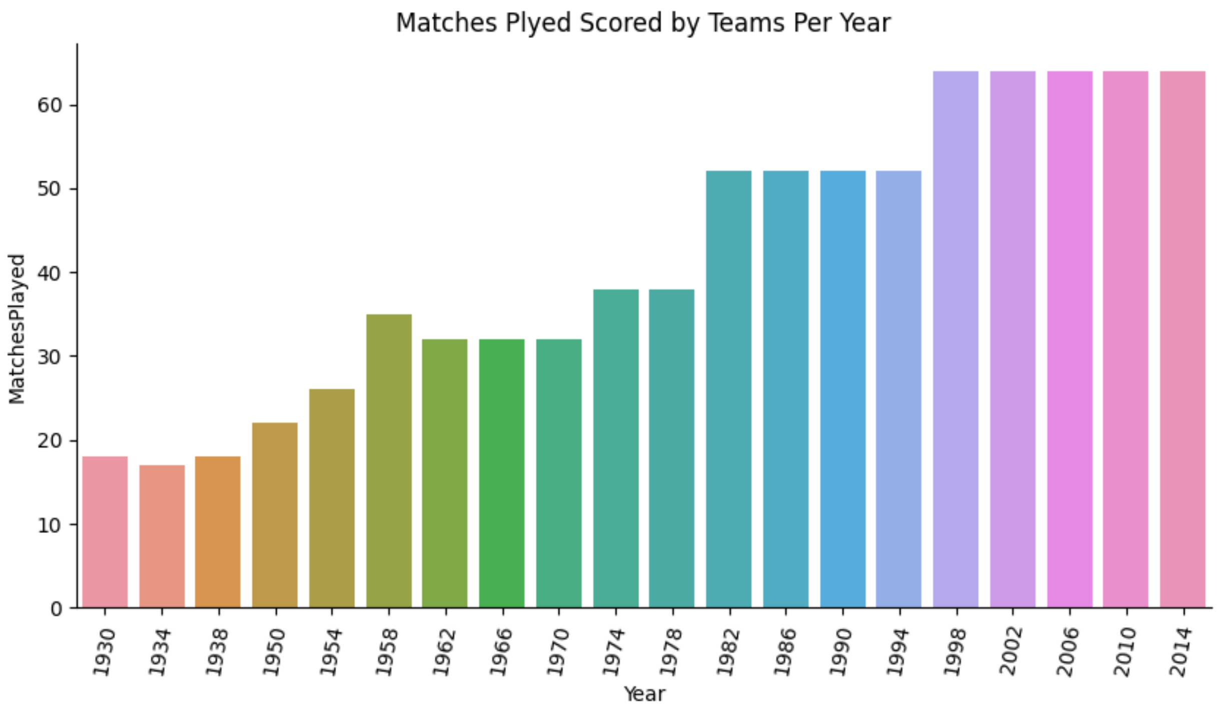
```
fig, ax = plt.subplots(figsize = (10, 7))
sns.despine(right = True)
g = sns.barplot(x = 'Year', y = 'QualifiedTeams', data = world_cup)
g.set_xticklabels(g.get_xticklabels(), rotation = 80)
g.set_title('Qualified Teams Per Year')

#======================

fig, ax = plt.subplots(figsize = (10,5))
sns.despine(right = True)
g = sns.barplot(x = 'Year', y = 'GoalsScored', data = world_cup)
g.set_xticklabels(g.get_xticklabels(), rotation = 80)
g.set_title('Goals Scored per Year')

#======================

fig, ax = plt.subplots(figsize = (10,5))
sns.despine(right = True)
g = sns.barplot(x = 'Year', y = 'MatchesPlayed', data = world_cup)
g.set_xticklabels(g.get_xticklabels(), rotation = 80)
g.set_title('Matches Plyed Scored by Teams Per Year')
```

Out[41]:

```
Text(0.5, 1.0, 'Matches Plyed Scored by Teams Per Year')
```


Qualified Teams Per Year
QualifiedTeams
0
5
10
15
20
25
30
1930
1934
1938
1950
1954
1958
1962
1966
1970
1974
1978
1982
1986
1990
1994
1998
2002
2006
2010
2014
Year


Goals Scored per Year
GoalsScored
0
20
40
60
80
100
120
140
160
1930
1934
1938
1950
1954
1958
1962
1966
1970
1974
1978
1982
1986
1990
1994
1998
2002
2006
2010
2014
Year



Goals Per Team Per World Cup

In [43]:
```
# Creating a new list which stores the grouped Year and Home teams
home = matches.groupby(['Year', 'Home Team Name'])['Home Team Goals'].sum

home
```

Out[43]:
```
Year     Home Team Name
1930.0   Argentina          16.0
         Brazil              4.0
         Chile               4.0
         France              4.0
         Paraguay            1.0
                            ...
2014.0   Russia              1.0
         Spain               1.0
         Switzerland         4.0
         USA                 2.0
         Uruguay             3.0
Name: Home Team Goals, Length: 366, dtype: float64
```

In [44]:
```
# Similarly, creating a list for away teams (which also stores goals scor
away = matches.groupby(['Year', 'Away Team Name'])['Away Team Goals'].sum
away
```

```
Out[44]: Year    Away Team Name
         1930.0  Argentina         2.0
                 Belgium           0.0
                 Bolivia           0.0
                 Brazil            1.0
                 Chile             1.0
                                  ...
         2014.0  Russia            1.0
                 Spain             3.0
                 Switzerland       3.0
                 USA               4.0
                 Uruguay           1.0
         Name: Away Team Goals, Length: 411, dtype: float64
```

```
In [61]: # New list 'goals' which concats 'home' and 'away' teams
         goals = pd.concat([home, away], axis=1)
         # Replace all NULL values with 0
         goals.fillna(0, inplace=True)
         # Adding the home and away team goals
         goals['Goals'] = goals['Home Team Goals'] + goals['Away Team Goals']
         # Removing the Home Team Goals and Away Team Goals titles from list 'goal
         goals = goals.drop(labels = ['Home Team Goals', 'Away Team Goals'], axis


         goals
```

Out[61]:

Year		Goals
1930.0	Argentina	18.0
	Brazil	5.0
	Chile	5.0
	France	4.0
	Paraguay	1.0
...	...	...
1998.0	Iran	2.0
	Mexico	8.0
	Norway	5.0
	Tunisia	1.0
2006.0	IR Iran	0.0

427 rows × 1 columns

```
In [62]: # Reset the list index to default(default starts from 0)
         goals = goals.reset_index()
```

In [63]:
```python
# Adding column headers
goals.columns = ['Year', 'Country', 'Goals']

# Sort list to ascending (by year) and in descending (by goal)
goals = goals.sort_values(by = ['Year', 'Goals'], ascending = [True, Fals

goals
```

Out[63]:

	Year	Country	Goals
0	1930.0	Argentina	18.0
7	1930.0	Uruguay	15.0
6	1930.0	USA	7.0
8	1930.0	Yugoslavia	7.0
1	1930.0	Brazil	5.0
...	...	...	...
355	2014.0	Japan	2.0
361	2014.0	Russia	2.0
340	2014.0	Cameroon	1.0
352	2014.0	Honduras	1.0
353	2014.0	IR Iran	1.0

427 rows × 3 columns

In [65]:
```python
# Top 5 goals by each year
top5 = goals.groupby('Year').head()

top5.head(10)
```

Out[65]:

	Year	Country	Goals
0	1930.0	Argentina	18.0
7	1930.0	Uruguay	15.0
6	1930.0	USA	7.0
8	1930.0	Yugoslavia	7.0
1	1930.0	Brazil	5.0
13	1934.0	Italy	12.0
11	1934.0	Germany	11.0
10	1934.0	Czechoslovakia	9.0
9	1934.0	Austria	7.0
12	1934.0	Hungary	5.0

In [66]:
```
import plotly.graph_objects as go
```

In [67]:
```
x, y = goals['Year'].values, goals['Goals'].values
```

In [74]:
```
data = []

# Looping over the top5 list (with removed duplicates)
for team in top5['Country'].drop_duplicates().values:
    year = top5[top5['Country'] == team]['Year']
    goal = top5[top5['Country'] == team]['Goals']

#     Appending the Bargraphs to the end of data list
    data.append(go.Bar(x = year, y = goal, name = team))
# Cerating a layout with stack as bargraph
layout = go.Layout(barmode = 'stack', title = 'Top 5 Teams with Most Goal
# Creating the bargraph figure (diagram) with 'data' as data list and 'la
fig = go.Figure(data = data, layout = layout)
fig.show()
```

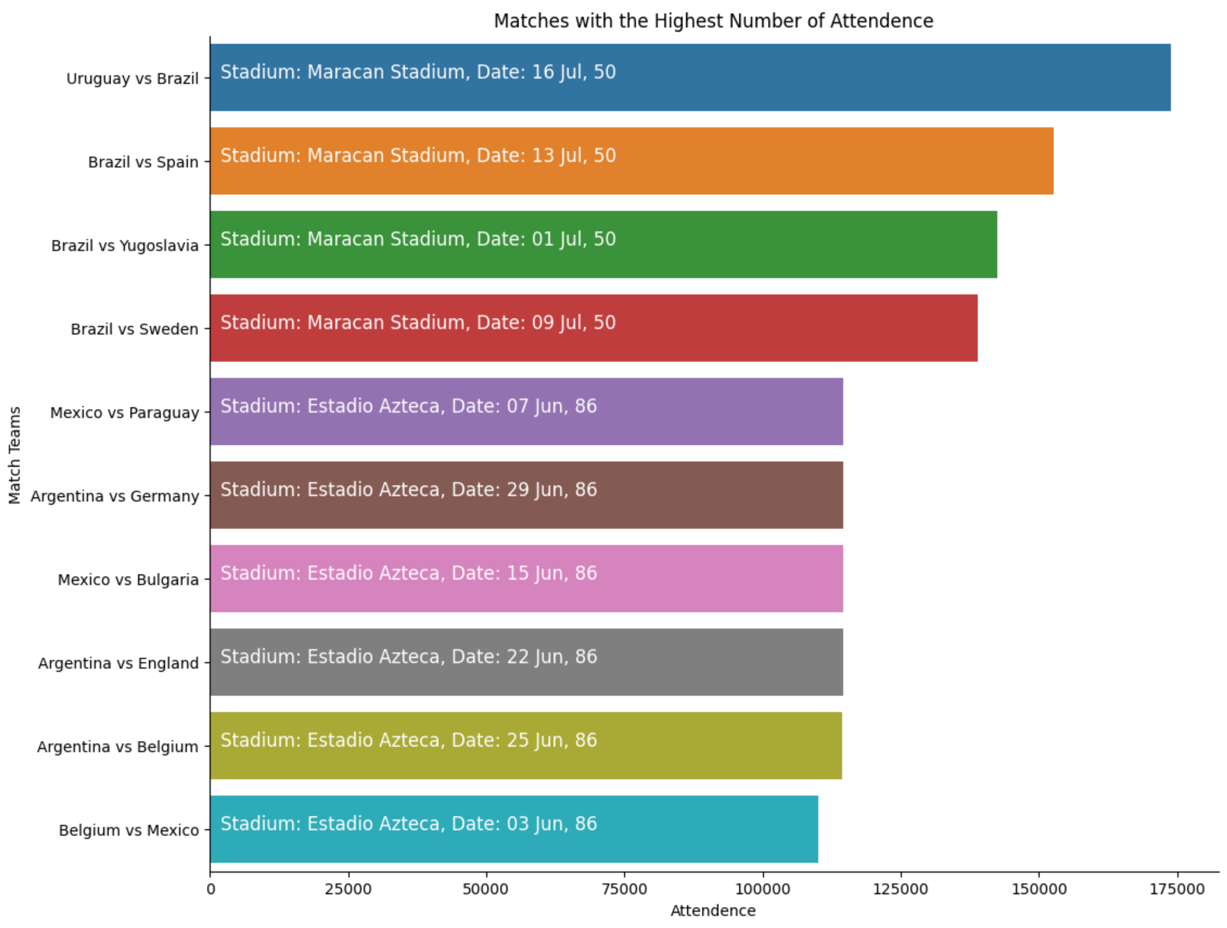
Matches With Highest Number Of Attendance

In [89]:
```
# Converting the date to datetime dtype
matches['Datetime'] = pd.to_datetime(matches['Datetime'])
```

In [90]:
```
# Stringifying the date format
# %d - date
# %b - abbrevated month name
# %y - year (YY format)

matches['Datetime'] = matches['Datetime'].apply(lambda x: x.strftime('%d
matches['Datetime']
```

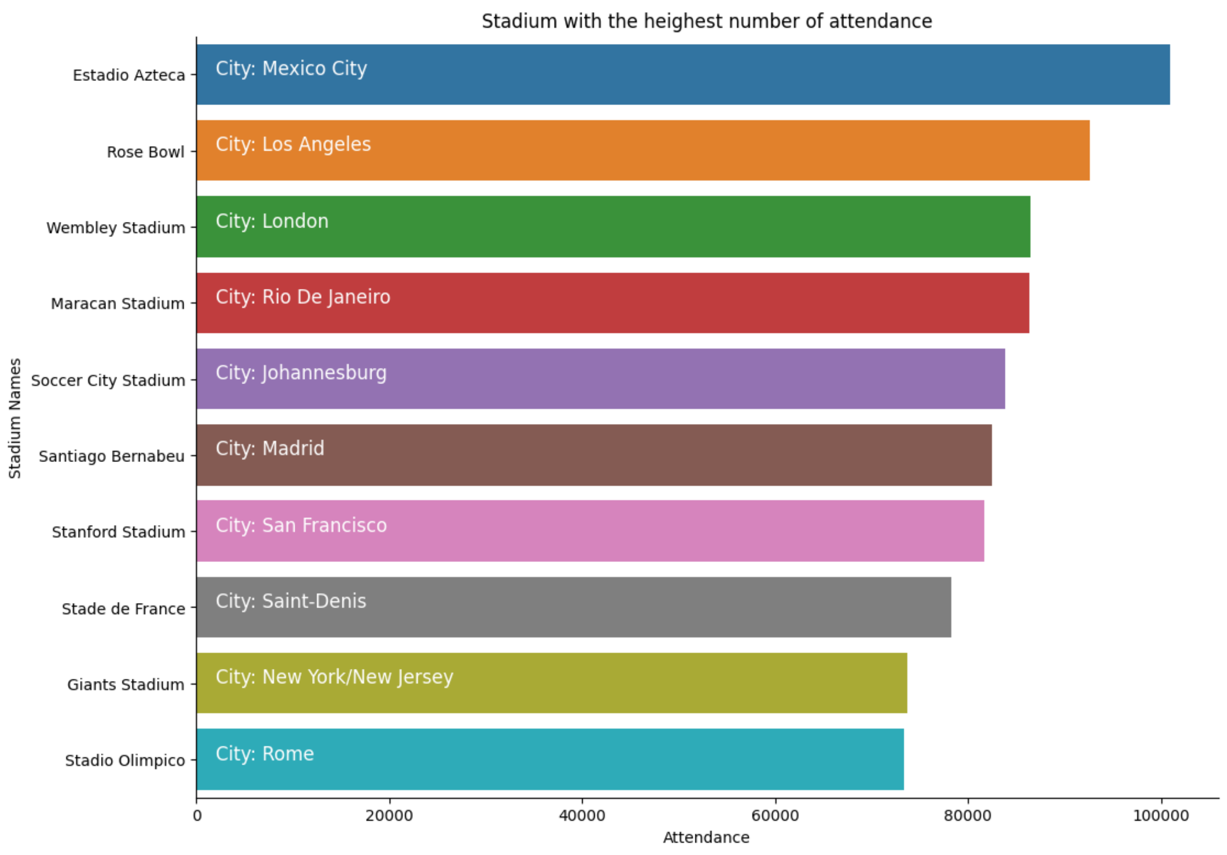
```
Out[90]: 0      13 Jul, 30
         1      13 Jul, 30
         2      14 Jul, 30
         3      14 Jul, 30
         4      15 Jul, 30
                   ...
         847    05 Jul, 14
         848    08 Jul, 14
         849    09 Jul, 14
         850    12 Jul, 14
         851    13 Jul, 14
         Name: Datetime, Length: 852, dtype: object
```

```
In [94]: # Sorting the top 10 matches
top10 = matches.sort_values(by = 'Attendance', ascending = False)[:10]
top10['vs'] = top10['Home Team Name'] + " vs " + top10['Away Team Name']

# Creating the plot figure with size (12, 10)
plt.figure(figsize = (12,10))

ax = sns.barplot(y = top10['vs'], x = top10['Attendance'])
sns.despine(right = True)

plt.ylabel('Match Teams')
plt.xlabel('Attendence')
plt.title('Matches with the Highest Number of Attendence')

for i, s in enumerate("Stadium: " + top10['Stadium'] +", Date: " + top10[
    ax.text(2000, i, s, fontsize = 12, color = 'white')
plt.show()
```



Stadium with Highest Average Attendance

In [96]:
```
matches['Year'] = matches['Year'].astype(int)

std = matches.groupby(['Stadium', 'City'])['Attendance'].mean().reset_ind

top10 = std[:10]

plt.figure(figsize = (12,9))
ax = sns.barplot(y = top10['Stadium'], x = top10['Attendance'])
sns.despine(right = True)

plt.ylabel('Stadium Names')
plt.xlabel('Attendance')
plt.title('Stadium with the heighest number of attendance')
for i, s in enumerate("City: " + top10['City']):
        ax.text(2000, i, s, fontsize = 12, color = 'white')

plt.show()
```

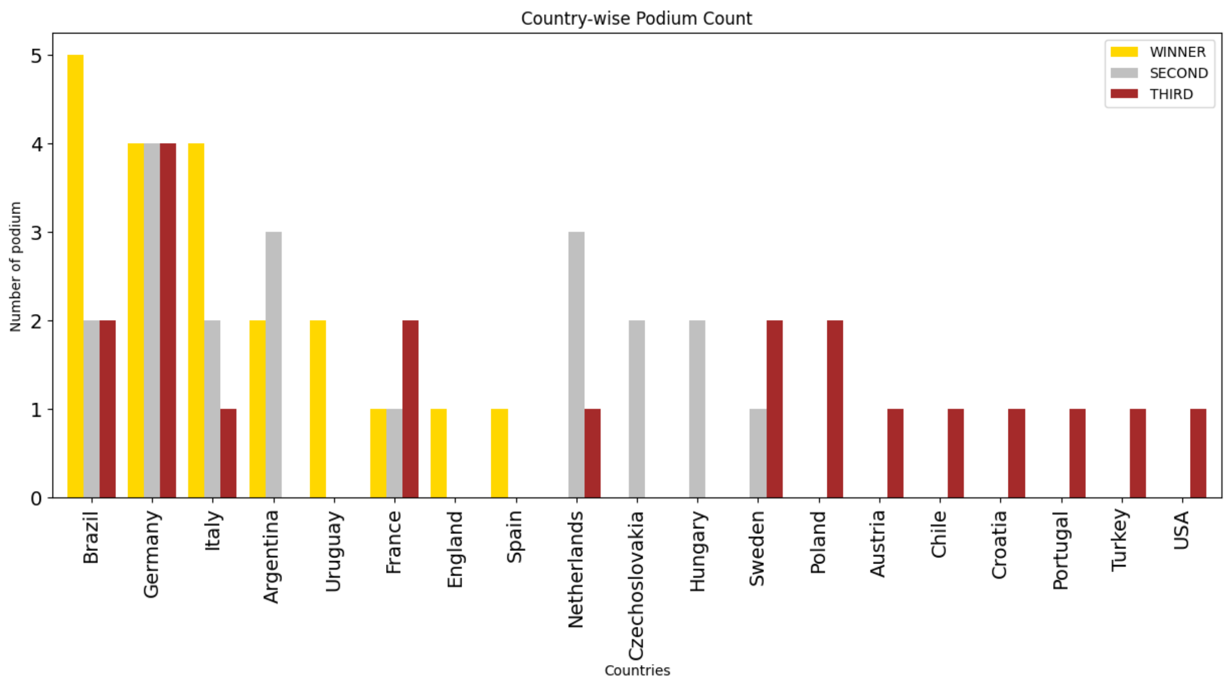


In [99]:
```
# Most matches were played in which cities
matches['City'].value_counts()[:20].iplot(kind = 'bar')
```

Which countries had won the cup ?

```
In [104…  # Creating 3 lists categorising Winner, Runners-Up and Third positions
          gold = world_cup["Winner"]
          silver = world_cup["Runners-Up"]
          bronze = world_cup["Third"]

          # Counting values from the lists (we have used from_dict() method as we a
          gold_count = pd.DataFrame.from_dict(gold.value_counts())
          silver_count = pd.DataFrame.from_dict(silver.value_counts())
          bronze_count = pd.DataFrame.from_dict(bronze.value_counts())

          # Performing Outer join
          podium_count = gold_count.join(silver_count, how='outer').join(bronze_cou
          # Filling NULL values with 0
          podium_count = podium_count.fillna(0)
          # Adding column titles
          podium_count.columns = ['WINNER', 'SECOND', 'THIRD']
          # Converting the dtype of list to int64
          podium_count = podium_count.astype('int64')
          # Sorting values in descending order
          podium_count = podium_count.sort_values(by=['WINNER', 'SECOND', 'THIRD'],

          # Plotting the bargraph
          podium_count.plot(y=['WINNER', 'SECOND', 'THIRD'], kind="bar",
                            color =['gold','silver','brown'], figsize=(15, 6), font
                           width=0.8, align='center')
          plt.xlabel('Countries')
          plt.ylabel('Number of podium')
          plt.title('Country-wise Podium Count')
```

```
Out[104]:  Text(0.5, 1.0, 'Country-wise Podium Count')
```



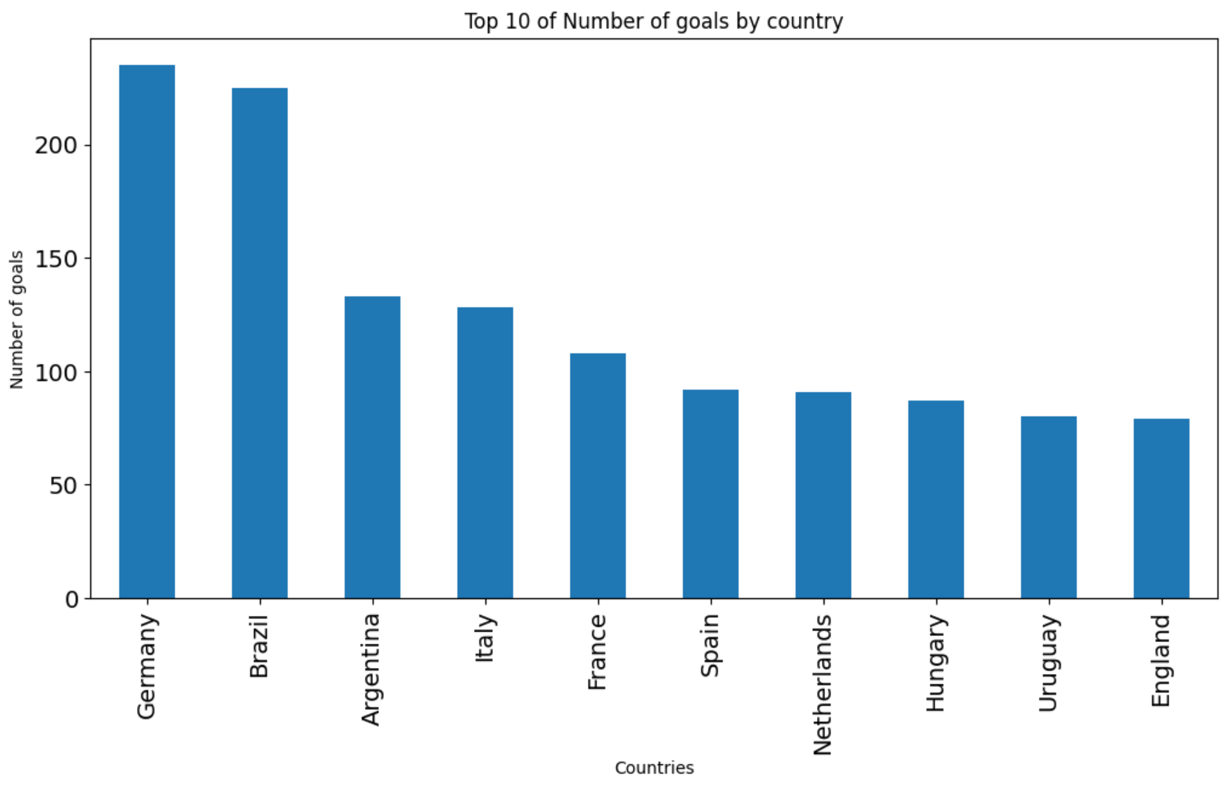
In [116…

```
# Removing NULL values
home = matches[['Home Team Name', 'Home Team Goals']].dropna()
# Removing NULL values
away = matches[['Away Team Name', 'Away Team Goals']].dropna()

# Creating a new DataFrame with countires and goals
goal_per_country = pd.DataFrame(columns=['countries', 'goals'])
# Appending away to home and then assigning the resultant to goal_per_cou
goal_per_country = goal_per_country.append(home.rename(index=str, columns
goal_per_country = goal_per_country.append(away.rename(index=str, columns

# Converting the dtype of goals
goal_per_country['goals'] = goal_per_country['goals'].astype('int64')

# Grouping the countries by country name, adding the goal values and then
goal_per_country = goal_per_country.groupby(['countries'])['goals'].sum()

# Plotting the graph
goal_per_country[:10].plot(x=goal_per_country.index, y=goal_per_country.v
plt.xlabel('Countries')
plt.ylabel('Number of goals')
plt.title('Top 10 of Number of goals by country')
```

```
/var/folders/ly/dlqmsw2s4f7gpzxx893rqbm40000gn/T/ipykernel_99227/2782247
911.py:9: FutureWarning:

The frame.append method is deprecated and will be removed from pandas in
a future version. Use pandas.concat instead.

/var/folders/ly/dlqmsw2s4f7gpzxx893rqbm40000gn/T/ipykernel_99227/2782247
911.py:10: FutureWarning:

The frame.append method is deprecated and will be removed from pandas in
a future version. Use pandas.concat instead.
```

Out[116]:

```
Text(0.5, 1.0, 'Top 10 of Number of goals by country')
```



Match outcome by home and away teams

In [117…
```
# Creating a function get_labels() which takes in a match as argument and
# If both teams scored equally, it returns 'DRAW'
def get_labels(matches):
    if matches['Home Team Goals'] > matches['Away Team Goals']:
        return 'Home Team Win'
    if matches['Home Team Goals'] < matches['Away Team Goals']:
        return 'Away Team Win'
    return 'DRAW'
```

In [118…
```
# Creating a new column 'outcome' in matches dataframe and performing the
matches['outcome'] = matches.apply(lambda x: get_labels(x), axis=1)
```

In [120…
```
# Matches dataframe after executing get_label() function
matches.head()
```

Out[120]:

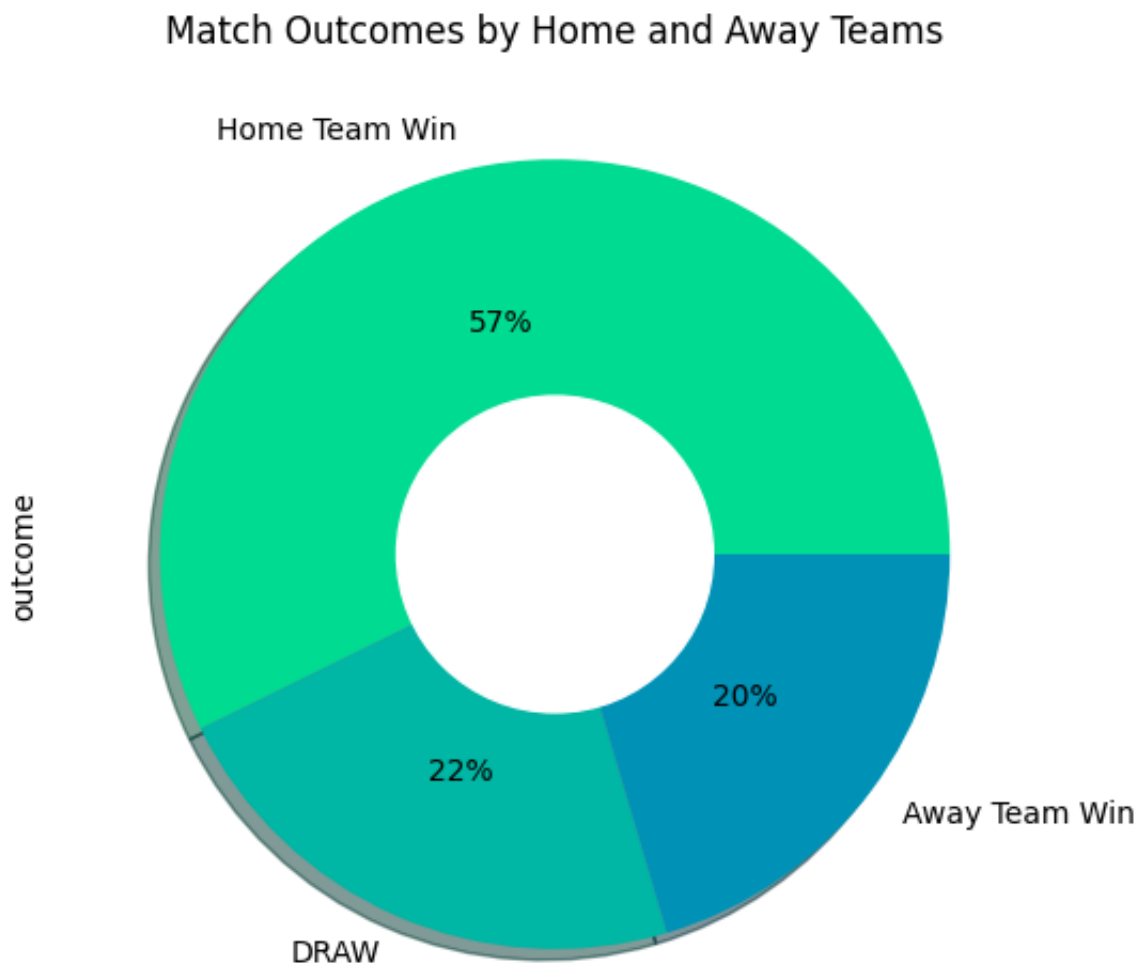
	Year	Datetime	Stage	Stadium	City	Home Team Name	Home Team Goals	Away Team Goals	Away Team Name	conc
0	1930	13 Jul, 30	Group 1	Pocitos	Montevideo	France	4.0	1.0	Mexico	
1	1930	13 Jul, 30	Group 4	Parque Central	Montevideo	USA	3.0	0.0	Belgium	
2	1930	14 Jul, 30	Group 2	Parque Central	Montevideo	Yugoslavia	2.0	1.0	Brazil	
3	1930	14 Jul, 30	Group 3	Pocitos	Montevideo	Romania	3.0	1.0	Peru	
4	1930	15 Jul, 30	Group 1	Parque Central	Montevideo	Argentina	1.0	0.0	France	

5 rows × 21 columns

In [121…
```
# Counting the total result of matchs played
match_outcomes = matches['outcome'].value_counts()
match_outcomes
```

Out[121]:
```
Home Team Win    488
DRAW             190
Away Team Win    174
Name: outcome, dtype: int64
```

In [122…
```
# Plotting graph on match_outcomes
plt.figure(figsize = (6,6))

match_outcomes.plot.pie(autopct = "%1.0f%%", colors = sns.color_palette('

c = plt.Circle((0,0), 0.4, color =  'white')
plt.gca().add_artist(c)
plt.title('Match Outcomes by Home and Away Teams')
plt.show()
```


Match Outcomes by Home and Away Teams
Home Team Win
57%
outcome
20%
22%
Away Team Win
DRAW

```
In [ ]:
```