

Introduction to Data Science: Football Worldcup Data Analysis

```
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 visualisation (graphs etc.)
import matplotlib inline
import plotly as py # for data visualisation (it supports a huge variety of chart types)
import cufflinks as cf # it connects plotly with pandas so that we can create charts

C:\Users\conta\anaconda3\lib\site-packages\scipy\__init__.py:138: UserWarning: A NumPy version >=1.16.5 and <1.23.0 is required for this version of SciPy (detected version 1.24.4)
warnings.warn(f"{'A NumPy version'>='{np_minversion}' and '<np_maxversion}' is required for this version of "

In [4]: # Importing the datasets
worldcup = pd.read_csv("C:/Users/conta/OneDrive/Desktop/Projects/Data Analysis (World Cup)/worldcup.csv")
matches = pd.read_csv("C:/Users/conta/OneDrive/Desktop/Projects/Data Analysis (World Cup)/matches.csv")
world_cup = pd.read_csv("C:/Users/conta/OneDrive/Desktop/Projects/Data Analysis (World Cup)/world_cup.csv")

In [5]: # 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[5]:
```

	Year	Datetime	Stage	Stadium	City	Home Team Name	Home Team Goals	Away Team Goals	Away Team Name	Win conditions	Attendance
0	1930.0	13 Jul 1930 - 15:00	Group 1	Pocitos	Montevideo	France	4.0	1.0	Mexico		4444.0
1	1930.0	13 Jul 1930 - 15:00	Group 4	Parque Central	Montevideo	USA	3.0	0.0	Belgium		18346.0
2	1930.0	14 Jul 1930 - 12:45	Group 2	Parque Central	Montevideo	Yugoslavia	2.0	1.0	Brazil		24059.0
3	1930.0	14 Jul 1930 - 14:50	Group 3	Pocitos	Montevideo	Romania	3.0	1.0	Peru		2549.0
4	1930.0	15 Jul 1930 - 16:00	Group 1	Parque Central	Montevideo	Argentina	1.0	0.0	France		23409.0

```
In [6]: # First 5 rows of matches dataset
matches.head()

Out[6]:
```

	Year	Datetime	Stage	Stadium	City	Home Team Name	Home Team Goals	Away Team Goals	Away Team Name	Win conditions	Attendance
0	1930.0	13 Jul 1930 - 15:00	Group 1	Pocitos	Montevideo	France	4.0	1.0	Mexico		4444.0
1	1930.0	13 Jul 1930 - 15:00	Group 4	Parque Central	Montevideo	USA	3.0	0.0	Belgium		18346.0
2	1930.0	14 Jul 1930 - 12:45	Group 2	Parque Central	Montevideo	Yugoslavia	2.0	1.0	Brazil		24059.0
3	1930.0	14 Jul 1930 - 14:50	Group 3	Pocitos	Montevideo	Romania	3.0	1.0	Peru		2549.0
4	1930.0	15 Jul 1930 - 16:00	Group 1	Parque Central	Montevideo	Argentina	1.0	0.0	France		23409.0

```
In [7]: # The last 5 entries in the matches dataset
matches.tail()
```

```
Out[7]:
```

	Year	Datetime	Stage	Stadium	City	Home Team Name	Home Team Goals	Away Team Goals	Away Team Name	Win conditions	Half-time Goals	Half-time Average Goals
4567	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
4568	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
4569	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
4570	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
4571	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN

```
In [8]: # First 5 rows of worldcup dataset
world_cup.head()
```

```
Out[8]:
```

	Year	Country	Winner	Runners-Up	Third	Fourth	GoalsScored	QualifiedTeams	MatchesPlayed
0	1930	Uruguay	Uruguay	Argentina	USA	Yugoslavia	70	13	1
1	1934	Italy	Italy	Czechoslovakia	Germany	Austria	70	16	1
2	1938	France	Italy	Hungary	Brazil	Sweden	84	15	1
3	1950	Brazil	Uruguay	Brazil	Sweden	Spain	88	13	2
4	1954	Switzerland	Germany FR	Hungary	Austria	Uruguay	140	16	2

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 [9]: matches.dropna(subset=['Year'], inplace=True)
# It removes all entries where the 'Year' column had NULL values

In [10]: matches.tail()
# You can see the dataset after data cleaning

Out[10]:
```

	Year	Datetime	Stage	Stadium	City	Home Team Name	Home Team Goals	Away Team Goals	Away Team Name	Win conditions	Attendance
847	2014.0	05 Jul 2014 - 17:00	Quarter-finals	Arena Fonte Nova	Salvador	Netherlands	0.0	0.0	Costa Rica	Netherlands win on penalties (4 - 3)	
848	2014.0	08 Jul 2014 - 17:00	Semi-finals	Estadio Mineirao	Belo Horizonte	Brazil	1.0	7.0	Germany		
849	2014.0	09 Jul 2014 - 17:00	Semi-finals	Arena do Sao Paulo	Sao Paulo	Netherlands	0.0	0.0	Argentina	Argentina win on penalties (2 - 4)	
850	2014.0	12 Jul 2014 - 17:00	Play-off for third place	Estadio Nacional	Brasilia	Brazil	0.0	3.0	Netherlands		
851	2014.0	13 Jul 2014 - 16:00	Final	Estadio do Maracana	Rio De Janeiro	Germany	1.0	0.0	Argentina	Germany win after extra time	

```
In [11]: # Getting the value count in a dataset (meaning how many time a specific value has been used)
matches['Home Team Name'].value_counts()

Out[11]:
Brazil      82
Italy       57
Argentina   54
Germany FR  43
England    35
..         ..
Jamaica     1
Bolivia     1
New Zealand 1
Iran         1
rn>'United Arab Emirates' 1
rn>'United Arab Emirates' 1
Name: Home Team Name, Length: 78, dtype: int64

In [12]: matches['Home Team Goals'].value_counts()

Out[12]:
1.0    246
2.0    205
0.0    177
3.0    116
4.0     59
5.0     17
6.0     17
7.0      8
8.0      4
9.0      2
10.0     1
Name: Home Team Goals, dtype: int64

In [13]: # Some Home Teams include unrequired characters.
names = matches[matches['Home Team Name'].str.contains('rn>')]['Home Team Name'].values
names

Out[13]:
rn>'Republic of Ireland'      5
rn>'United Arab Emirates'     1
rn>'Bosnia and Herzegovina'    1
rn>'Serbia and Montenegro'     1
rn>'Trinidad and Tobago'      1
Name: Home Team Name, dtype: int64

In [14]: # Creating a new list with wrong names of the Home Teams (they include the 'rn>' in the name)
wrong = list(names.index)

Out[14]:
['rn>'Republic of Ireland',
 'rn>'United Arab Emirates',
 'rn>'Bosnia and Herzegovina',
 'rn>'Serbia and Montenegro',
 'rn>'Trinidad and Tobago']

In [15]: # Removing the 'rn>' from the names. We do not require it.
correct = [name.split('>')[1] for name in wrong]
correct

Out[15]:
['Republic of Ireland',
 'United Arab Emirates',
 'Bosnia and Herzegovina',
 'Serbia and Montenegro',
 'Trinidad and Tobago',
 'Germany FR',
 'Maracan Stadium',
 'Maracan Stadium']

In [16]: old_name = ['Germany FR', 'Maracan Stadium', 'Maracan Stadium']
new_name = ['Germany', 'Maracan Stadium', 'Maracan Stadium'] # These are the correct names

In [17]: wrong = wrong + old_name #Adding oldname to the wrong names list
correct = correct + new_name #Adding newname to the correct names list

In [18]: wrong # Displaying the updated wrong team names list

Out[18]:
['rn>'Republic of Ireland',
 'rn>'United Arab Emirates',
 'rn>'Bosnia and Herzegovina',
 'rn>'Serbia and Montenegro',
 'rn>'Trinidad and Tobago',
 'Germany FR',
 'Maracan Stadium',
 'Maracan Stadium']

In [19]: correct # Displaying the updated correct team names list

Out[19]:
['Republic of Ireland',
 'United Arab Emirates',
 'Bosnia and Herzegovina',
 'Serbia and Montenegro',
 'Trinidad and Tobago',
 'Germany',
 'Maracan Stadium',
 'Maracan Stadium']

In [20]: # Replacing wrong values with corrected ones in players, matches and worldcup dataset.
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 [21]: # Checking if there are still any wrong entries listed in the dataset
names = matches[matches['Home Team Name'].str.contains('rn>')]['Home Team Name'].values
names

Out[21]:
Series([], Name: Home Team Name, dtype: int64)
```

Most Number of World Cup Winning Title

```
In [22]: # Counting the number of times each country has appeared as the 'Winner' in the worldcup dataset
winner = matches[matches['Winner'] == 'Winner'].value_counts()

winner

Out[22]:
Brazil      5
Germany     4
Italy       4
Argentina   2
Uruguay     2
England     1
France      1
Spain       1
Name: Winner, dtype: int64

In [23]: # Counting the number of times each country has appeared as the 'Runner Up' in the worldcup dataset
runnerup = matches[matches['Runners-Up'] == 'Runner Up'].value_counts()

runnerup

Out[23]:
Germany     4
Netherlands 3
Argentina   3
Italy       2
Czechoslovakia 2
Hungary     2
Brazil      2
France      1
Sweden      1
Name: Runners-Up, dtype: int64

In [24]: # Counting the number of times each country has appeared as the 'Third' in the worldcup dataset
third = matches[matches['Third'] == 'Third'].value_counts()

third

Out[24]:
Germany     4
France      2
Poland      2
Sweden      2
Brazil      2
Austria     1
Chile       1
Croatia     1
Portugal    1
USA         1
Turkey     1
Italy       1
Netherlands 1
Name: Third, dtype: int64

In [25]: # Creating a new table listing the all the above values
teams = pd.concat([winner, runnerup, third], axis=1) # Concatating the three values
teams.fillna(0, inplace=True) # Replaces the NULL values with a 0
teams = teams.astype(int) # Converts all dtypes to int
teams

Out[25]:
```

	Winner	Runners-Up	Third
Brazil	5	2	2
Germany	4	4	4
Italy	4	2	1
Argentina	2	3	0
Uruguay	2	0	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
Poland	0	1	2
Austria	0	0	1
Chile	0	0	1
Portugal	0	0	1
USA	0	0	1
Turkey	0	0	1

```
In [26]: # Importing iplot from plotly and connecting it to offline mode
from plotly.offline import plot
py.offline.init_notebook_mode(connected=True)
cf.go_offline()
```

```
In [27]: teams.iplot(kind = 'bar', xTitle='Teams', yTitle='World Cup Winning Count', title='FIFA World Cup Winning Count')

FIFA World Cup Winning Count
```

Team	Winner	Runners-Up	Third
Brazil	5	2	2
Germany	4	4	4
Italy	4	2	1
Argentina	2	3	0
Uruguay	2	0	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
Poland	0	1	2
Austria	0	0	1
Chile	0	0	1
Portugal	0	0	1
USA	0	0	1
Turkey	0	0	1

Export to plot.ly >

Number of Goal per Country

```
In [28]: matches.head(2)

Out[28]:
```

	Year	Datetime	Stage	Stadium	City	Home Team Name	Home Team Goals	Away Team Goals	Away Team Name	Win conditions	Attendance	Home Goals	Away Goals
0	1930.0	13 Jul 1930 - 15:00	Group 1	Pocitos	Montevideo	France	4.0	1.0	Mexico		4444.0	4	1
1	1930.0	13 Jul 1930 - 15:00	Group 4	Parque Central	Montevideo	USA	3.0	0.0	Belgium		18346.0	3	0

```
In [29]: # Cleaning the NULL fields from Teams and Goals and storing output to new lists
home = matches[['Home Team Name', 'Home Team Goals']].dropna()
away = matches[['Away Team Name', 'Away Team Goals']].dropna()

In [30]: # Setting the columns as 'Countries' and 'Goals' for home and away teams
home.columns = ['Countries', 'Goals']
away.columns = home.columns

In [31]: # Creating a new list combining both values from home and away
goals = home.append(away, ignore_index = True)

In [32]: # Grouping the countries by total goals they have scored (as home teams + as away teams)
goals = goals.groupby('Countries').sum()

goals

Out[32]:
```

	Goals
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 x 1 columns

```
In [33]: # Sorting the list in descending order (most to less goals)
goals = goals.sort_values(by = 'Goals', ascending=False)

goals

Out[33]:
```

	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 Indies	0.0
Zaire	0.0

82 rows x 1 columns

```
In [34]: # Plotting graph for the most number of goals by countries
# Note that we're only displaying the top 30 countries from the list
goals[30].iplot(kind='bar', xTitle = 'Country', yTitle = 'Number of Goals', title = 'Countries Hits Number of Goals')

Countries Hits Number of Goals
```

Country	Goals
Brazil	225
Germany	235
Argentina	133
Italy	128
France	108
Hungary	84
Uruguay	80
Netherlands	63
Sweden	60
Yugoslavia	60
Mexico	57
Switzerland	57
Czechoslovakia	57
Poland	57
Austria	57
Chile	57
Romania	57
Paraguay	57
Scotland	57
Croatia	57
Bulgaria	57

Export to plot.ly >

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

```
In [35]: # Removing the decimals from the attendance column
world_cup['Attendance'] = world_cup['Attendance'].str.replace(".", "")

<ipython-input-35-0d5c66ba50cf>: FutureWarning:
The default value of regex will change from True to False in a future version. In addition, single character regular expressions will not be treated as literal strings when regex=True.

In [36]: world_cup.head(10)
```

	Year	Country	Winner	Runners-Up	Third	Fourth	GoalsScored	QualifiedTeams	MatchesPlayed
0	1930	Uruguay	Uruguay	Argentina	USA	Yugoslavia	70	13	1
1	1934	Italy	Italy	Czechoslovakia	Germany	Austria	70	16	1
2	1938	France	Italy	Hungary	Brazil	Sweden	84	15	1
3	1950	Brazil	Uruguay	Brazil	Sweden	Spain	88	13	2
4	1954	Switzerland	Germany	Hungary	Austria	Uruguay	140	16	2
5	1958	Sweden	Brazil	Sweden	France	Germany	126	16	3
6	1962	Chile	Brazil	Czechoslovakia	Chile	Yugoslavia	89	16	3
7	1966	England	England	Germany	Portugal	Soviet Union	89	16	3
8	1970	Mexico	Brazil	Italy	Germany	Uruguay	95	16	3
9	1974	Germany	Germany	Netherlands	Poland	Brazil	97	16	3

```
In [37]: 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 Played Scored by Teams Per Year')
```

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

Qualified Teams Per Year

Year	Qualified Teams
1930	13
1934	16
1938	15
1950	13
1954	16
1958	16
1962	16
1966	16
1970	16
1974	16
1978	16
1982	24
1986	24
1990	24
1994	24
1998	32
2002	32
2006	32
2010	32
2014	32

Goals Scored per Year

Year	Goals Scored
1930	70
1934	70
1938	84
1950	88
1954	140
1958	126
1962	89
1966	89
1970	95
1974	97
1978	108
1982	146
1986	132
1990	140
1994	146
1998	161
2002	155
2006	146
2010	146
2014	161

Matches Played Scored by Teams Per Year

Year	Matches Played
1930	18
1934	18
1938	18
1950	22
1954	26
1958	34
1962	32
1966	32
1970	32
1974	32
1978	38
1982	38
1986	52
1990	52
1994	52
1998	52
2002	62
2006	62
2010	62
2014	62

Goals Per Team Per World Cup

```
In [38]: # 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[38]:
```

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

```
In [39]: # Similarly, creating a list for away teams (which also stores goals scored by groups)
away = matches.groupby(['Year', 'Away Team Name'])['Away Team Goals'].sum()
away

Out[39]:
```

Year	Away Team Name	Goals
1930.0	Argentina	2.0
	Belgium	0.0
	Bolivia	0.0
	Brazil	1.0
	Chile	1.0

2014.0	Russia	1.0
	Spain	1.0
	Switzerland	3.0
	USA	4.0
	Uruguay	1.0
Name: Away Team Goals, Length: 411, dtype: float64		

```
In [40]: # 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 'goals'
goals = goals.drop(labels = ['Home Team Goals', 'Away Team Goals'], axis = 1)
goals

Out[40]:
```

	Year	Country	Goals
0	1930.0	Argentina	18.0
		Belgium	0.0
		Bolivia	0.0
		Brazil	5.0
		Chile	5.0
...
2014.0		Russia	2.0
		Spain	4.0
		Switzerland	7.0
		USA	6.0
		Uruguay	4.0
427 rows x 1 columns			

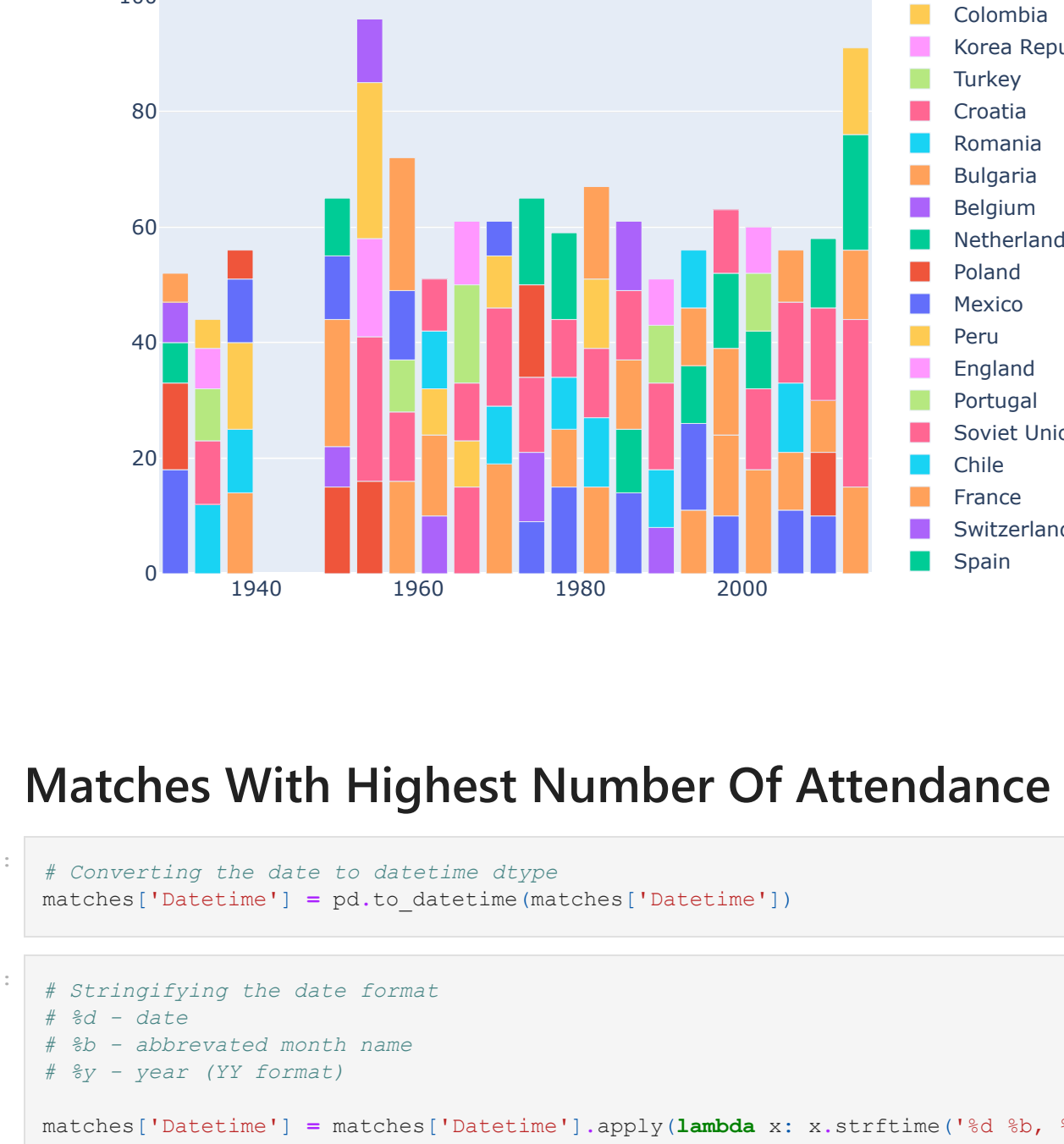
```
In [41]: # Reset the list index to default (default starts from 0)
goals = goals.reset_index()

In [42]: # Adding column headers
goals.columns = ['Year', 'Country', 'Goals']

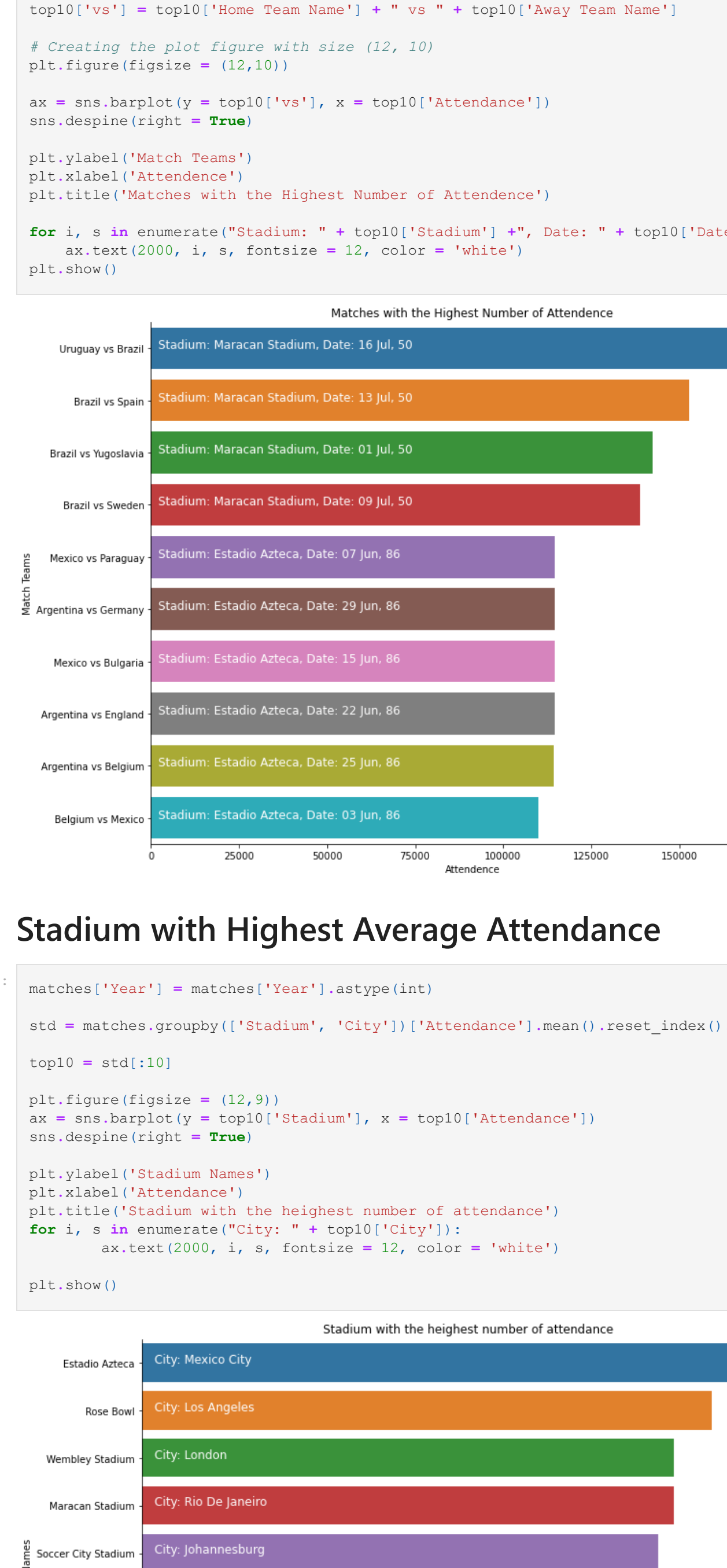
# Sort list to ascending (by year) and in descending (by goal)
goals = goals.sort_values(by = ['Year', 'Goals'], ascending = [True, False])
goals

Out[42]:
```

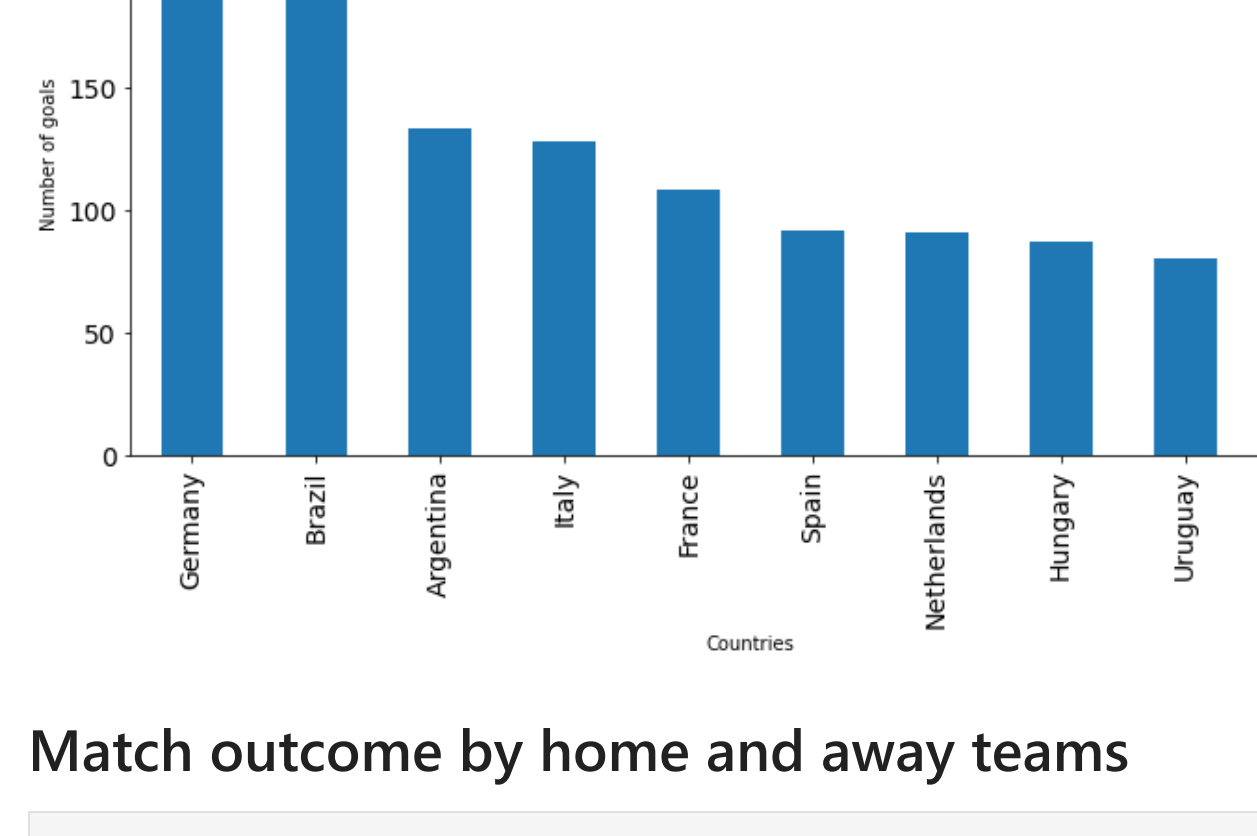
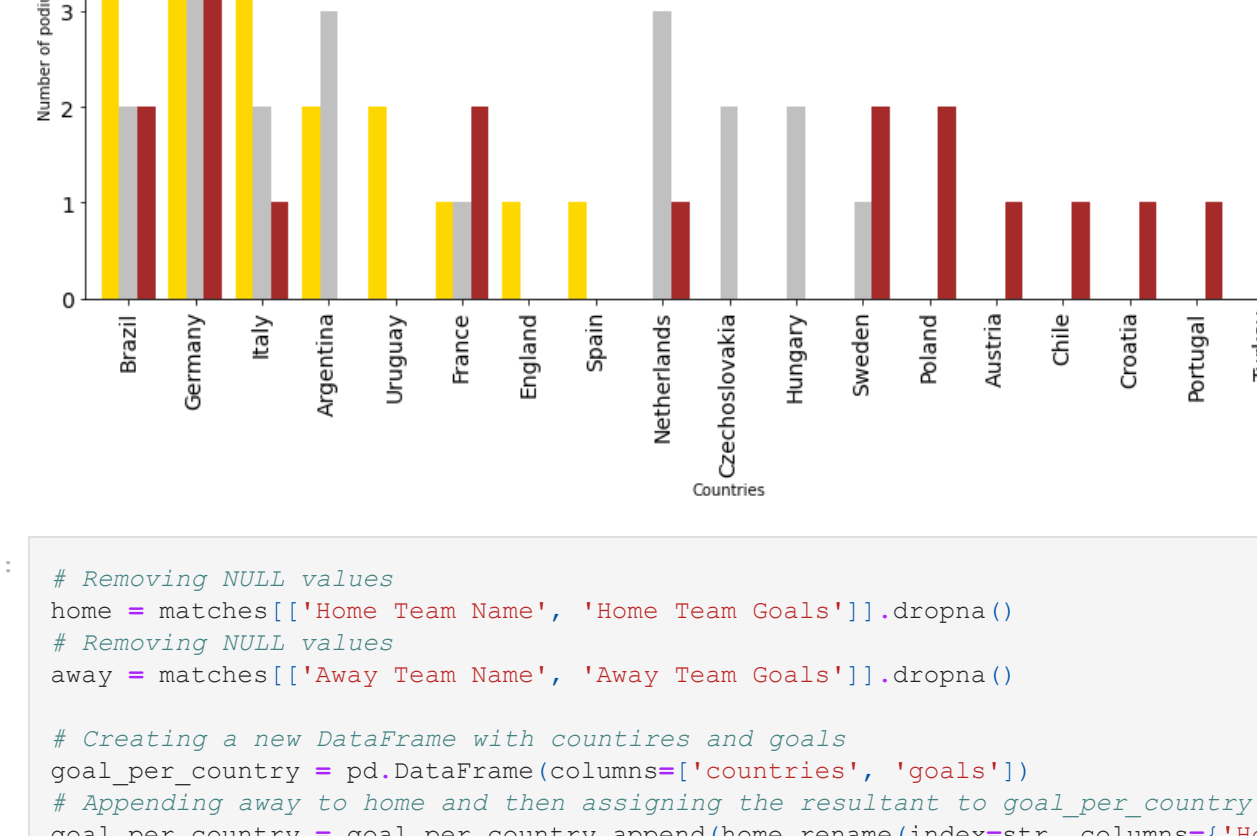
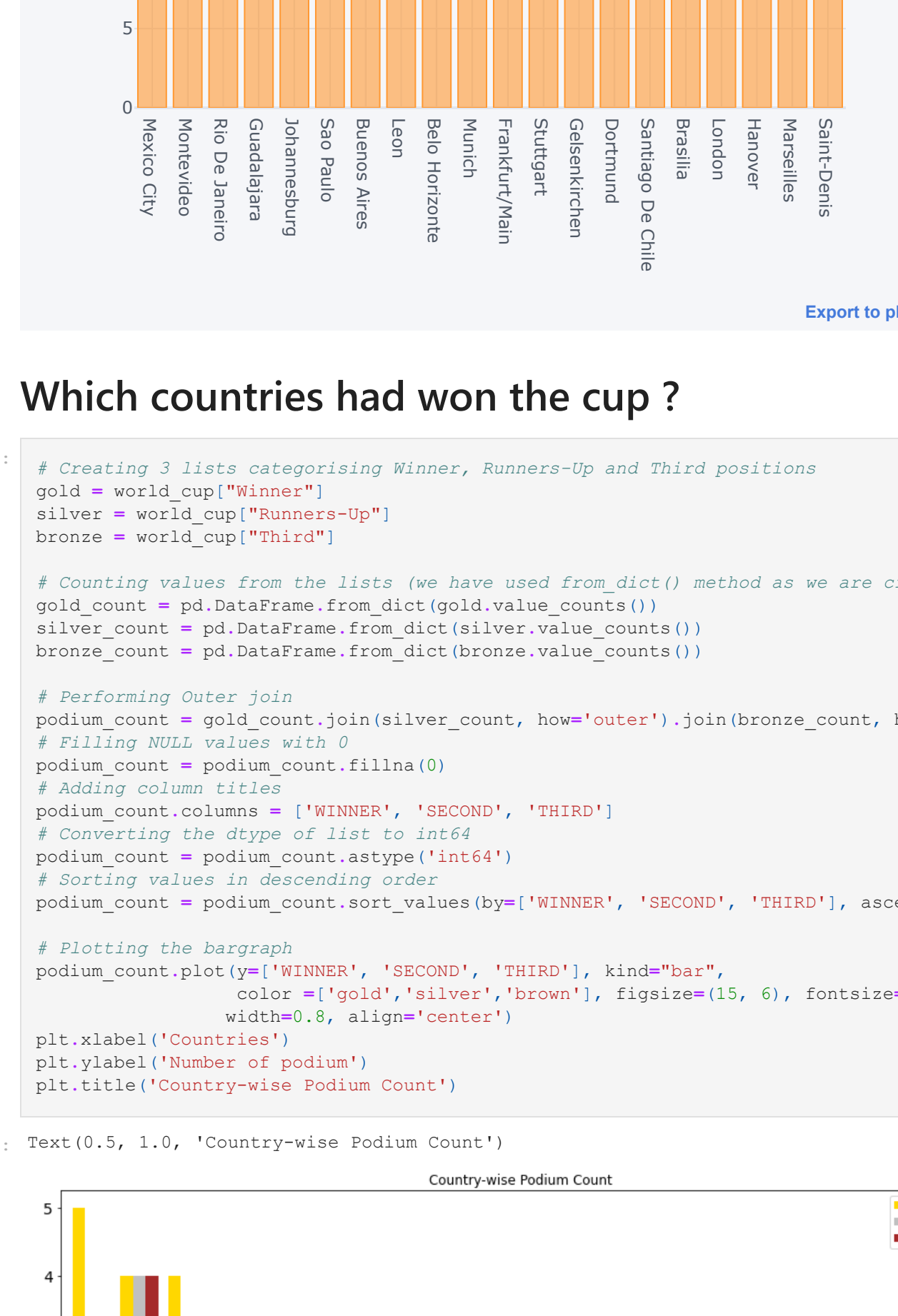
	Year	Country	Goals
0	1930.0	Argentina	18.0
11	1930.		



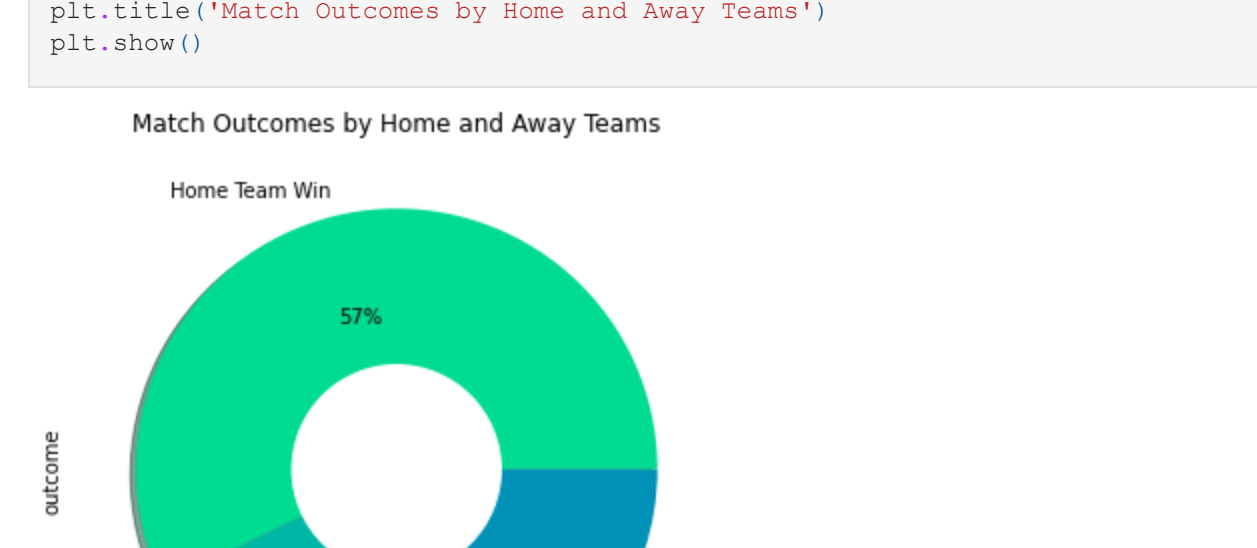
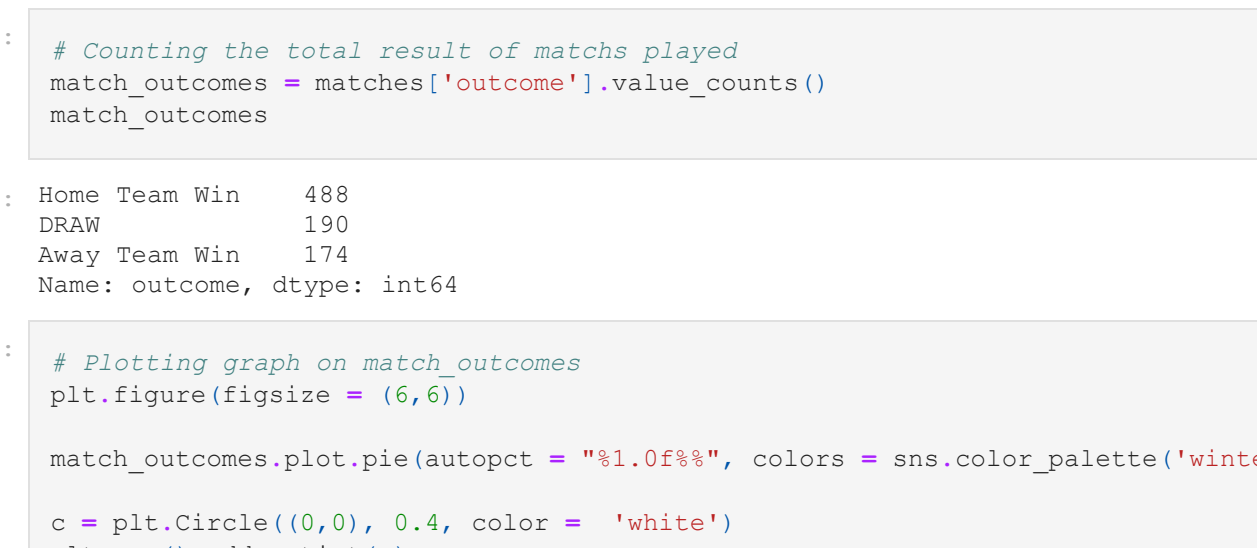
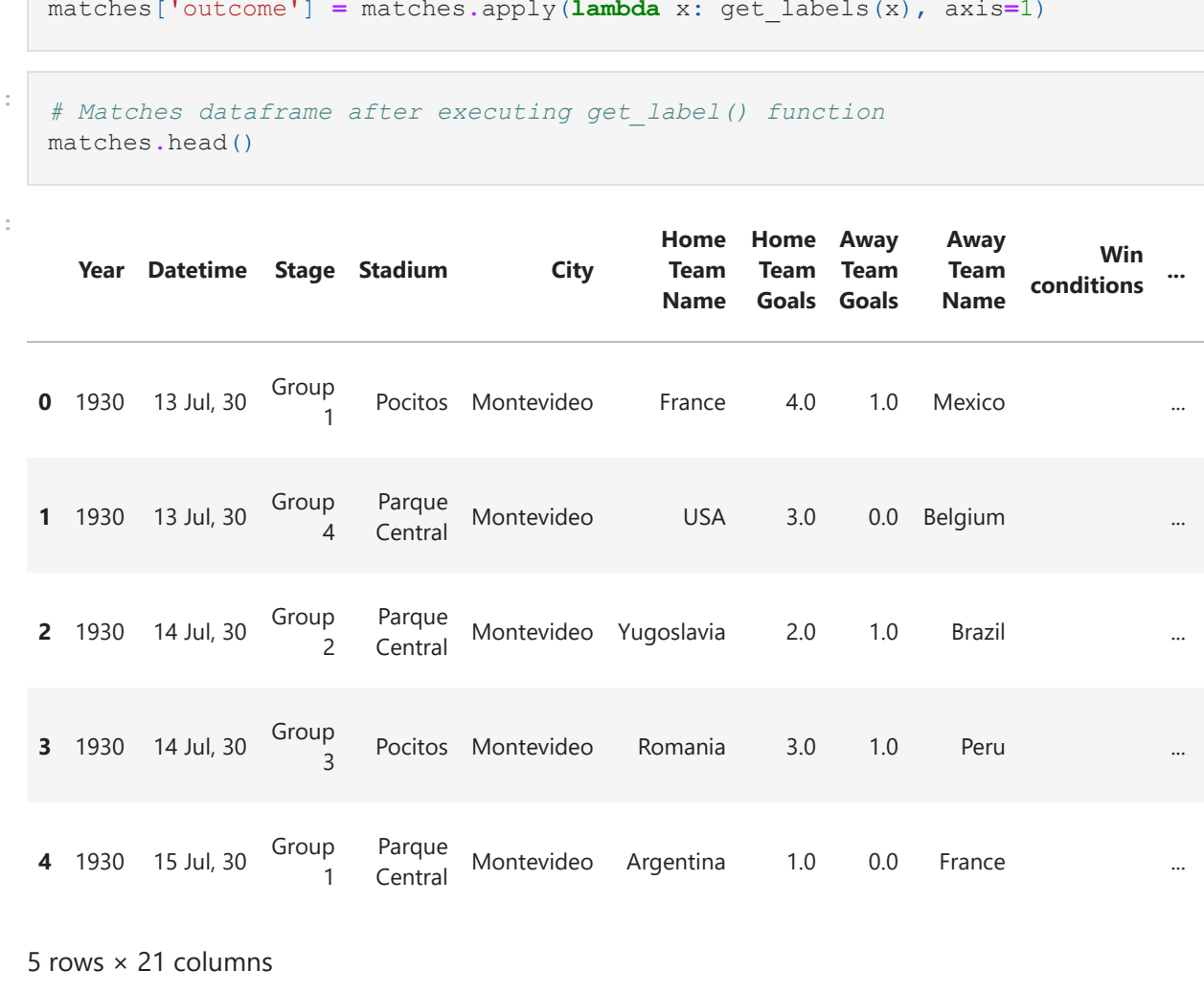
Matches With Highest Number Of Attendance



Stadium with Highest Average Attendance



Which countries had won the cup ?



Match outcome by home and away teams

