FIFA World Cup Analysis and Prediction

University of Toronto - SCS 3250 - Group Assignment 2018 (Group 2)

Jevonne E. Peters (Jevi)
School of Continuing Studies
University of Toronto
Toronto, Canada
hello@jevi.me

Muhammad Rizwan Kalim (Rizwan)
School of Continuing Studies
University of Toronto
Toronto, Canada
rizwanfastian@yahoo.com

Xiaming Gu (Caroline)
School of Continuing Studies
University of Toronto
Toronto, Canada
caroline3599@gmail.com

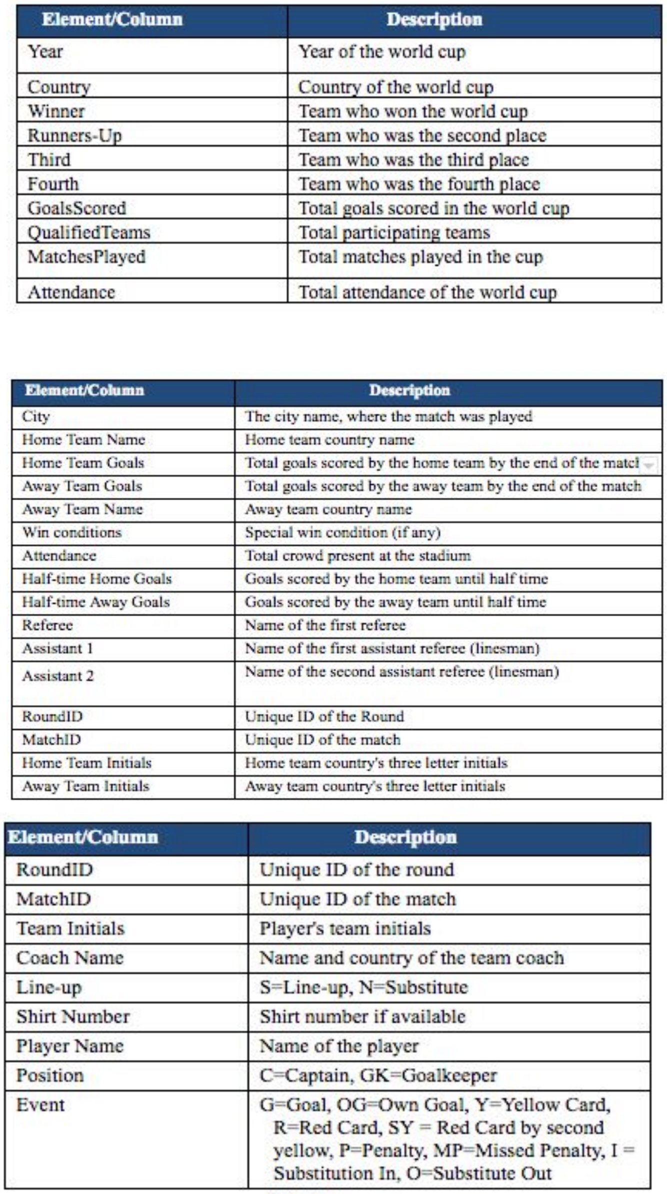
***Abstract*—An analysis of the FIFA World Cup dataset from 1930 to 2014, starting with an examination of proper data preparation techniques. This report includes a statistical and graphical study of the match attendance over the years, and correlations of note, if any. An analysis of the match performance of the teams participating in the FIFA World Cup games was also done, and used as a basis to predict qualifying teams, and the outcomes of FIFA 2018 match-ups.**

Keywords—FIFA, world cup, python, analysis

I. Data Context and Collection

The FIFA World Cup is a global football competition contested by the senior men's national teams from the 208 Member Associations of FIFA. The competition is held every four years since the inaugural tournament in 1930, with two exceptions. It is the most prestigious and important trophy in the sport of football.

Kaggle 'FIFA World Cup Data' [1] was used as the data source, with the origin being courtesy of the FIFA World Cup Archive website. It belongs to the 'Event' category, and comprised of three .csv files: World Cup, World Matches, World Cup Players. An additional, well-prepared data file featuring World Cup Hosts was compiled using the FIFA World Cup website, to supplement this dataset. (Fig. 1) shows the data dictionary of the raw data.



Element/Column	Description
Year	Year of the world cup
Country	Country of the world cup
Winner	Team who won the world cup
Runners-Up	Team who was the second place
Third	Team who was the third place
Fourth	Team who was the fourth place
GoalsScored	Total goals scored in the world cup
QualifiedTeams	Total participating teams
MatchesPlayed	Total matches played in the cup
Attendance	Total attendance of the world cup

Element/Column	Description
City	The city name, where the match was played
Home Team Name	Home team country name
Home Team Goals	Total goals scored by the home team by the end of the matcl
Away Team Goals	Total goals scored by the away team by the end of the match
Away Team Name	Away team country name
Win conditions	Special win condition (if any)
Attendance	Total crowd present at the stadium
Half-time Home Goals	Goals scored by the home team until half time
Half-time Away Goals	Goals scored by the away team until half time
Referee	Name of the first referee
Assistant 1	Name of the first assistant referee (linesman)
Assistant 2	Name of the second assistant referee (linesman)
RoundID	Unique ID of the Round
MatchID	Unique ID of the match
Home Team Initials	Home team country's three letter initials
Away Team Initials	Away team country's three letter initials

Element/Column	Description
RoundID	Unique ID of the round
MatchID	Unique ID of the match
Team Initials	Player's team initials
Coach Name	Name and country of the team coach
Line-up	S=Line-up, N=Substitute
Shirt Number	Shirt number if available
Player Name	Name of the player
Position	C=Captain, GK=Goalkeeper
Event	G=Goal, OG=Own Goal, Y=Yellow Card, R=Red Card, SY = Red Card by second yellow, P=Penalty, MP=Missed Penalty, I = Substitution In, O=Substitute Out

Fig. 1

II. Data Preparation

It is said that 80% of data analysis is spent on the process of cleaning and preparing the data [2]. With this in consideration, the techniques outlined in (Fig. 2) were employed as part of the data preparation process.

A. Completeness of the Data

Comparison of Gaps with Real-World Knowledge: It was determined that the dataset contained all the dates for the World Cup. Any gaps were checked for historical accuracy.□

Example: In the World Cup table, there are no records as would be expected for 1942 and 1946. This was checked against real-world data, which verified that those years were cancelled due to World War II.

Treatment of Null Rows & Cells: Every table was checked for null rows & cells, and the rows were discarded if they were completely blank.

Treatment of Duplicate Rows: Every table was checked for duplicate rows, which were deleted from the dataset.

'Missing' vs 'Not Applicable': For cases where rows were blank for a column, it was verified whether the data was 'missing' or 'not applicable'.

Example: World Cup Players table – Column 'Position'. This column shows whether a player was the captain of the team or the goalkeeper in a match. As there can be only one captain and one goalkeeper in a team, the empty rows under this column are not the missing, but rather 'not applicable'.

B. Data Anomalies

The following anomalies were found as part of the data preparation processing and were corrected.

Anomaly # 1 – Incorrect Data Types: All the tables were checked, using .info() method, to ensure that their columns had a suitable data type.□

Example: World Cup table – 'Attendance' column. The imported datatype was 'object' or 'string', which was converted to 'integer'.

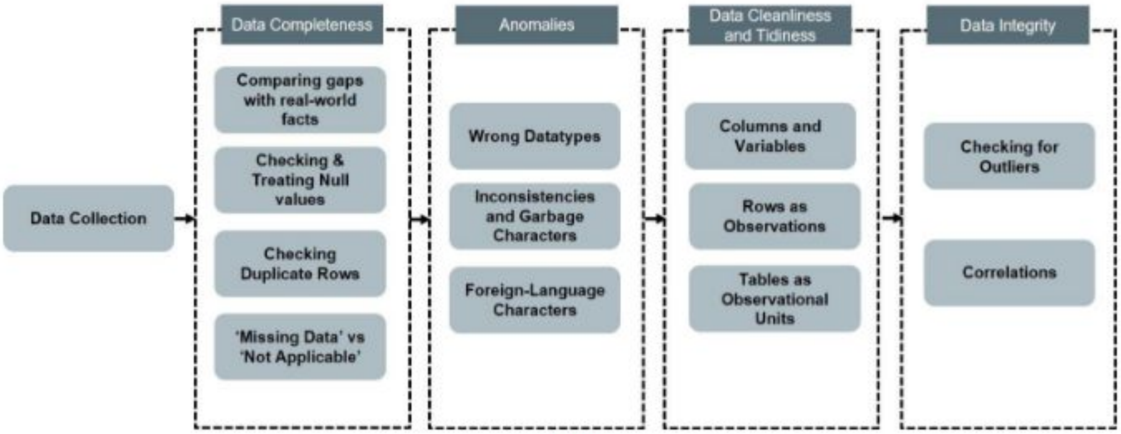

Data Collection
Data Completeness
Comparing gaps with real-world facts
Checking & Treating Null values
Checking Duplicate Rows
'Missing Data' vs 'Not Applicable'
Anomalies
Wrong Datatypes
Inconsistencies and Garbage Characters
Foreign-Language Characters
Data Cleanliness and Tidiness
Columns and Variables
Rows as Observations
Tables as Observational Units
Data Integrity
Checking for Outliers
Correlations

Fig. 2

Anomaly # 2 – Inconsistency in Names and Spelling, and Mojibake: Tables were parsed and reviewed for inconsistency in names and spellings or mojibake (garbled text). In most cases the correction of these anomalies required the consultation of external sources. □

Example: World Cup table – 'Winner' column. Throughout history, different names were used to refer to Germany: 'Germany FR', 'German DR' and 'Germany'. This was also the case for 'Iran'/'IR Iran' and 'Soviet Union'/'Russia'. Errors of this kind were also amended in the 'World Cup Matches' table to aid with better performance on statistical analysis.

Example: World Cup Matches table – 'Home Team' and 'Away Team' columns have mojibake due to encoded and file import errors. The records showed additional characters such as in this case: 'rn">Bosnia and Herzegovina'. These were found by scanning, and corrected.

Anomaly # 3 – Foreign Language Characters: Tables were reviewed for foreign language characters, and corrected using the correct Latin character based on the researched information.

Example: Based on supporting information, 'Stade V�lodrome' should be 'Stade Vélodrome

Example: Based on supporting information, 'Malm�' should be 'Malmö'

C. Organise & Clean Data

Data Tidiness: The following objectives were adhered to –

- Each column is a variable□
- Each row is an observation□
- Each type of observational unit forms a table

Example: World Cup table - A new column with World Cup Number was created to provide clarity. The column order was also changed.□

Example: World Cup Players table – Columns were rearranged for a more logical flow while following the objective of 'each row is an observation'.

D. Data Integrity

Outliers: The data was plotted to aid in the identifications of unexpected spikes and lows.

Example: World Cup table – 'Matches Played' vs 'Goals Scored'. An outlier was found between 20 and 30 on the x-axis (Fig. 3) which corresponds with FIFA World Cup # 5 in Switzerland. The entry had an unusually high number of goals compared to matches played during that Cup. This outlier was found to be accurate when compared to research.

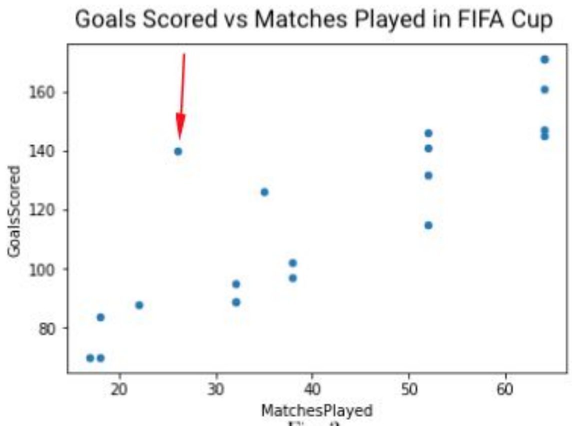
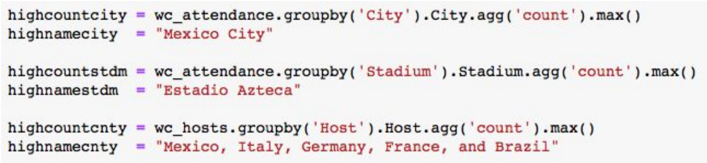


Fig. 3

III. Attendance Analysis & Correlations

A. Statistics

A statistical analysis of the data showed that there were a total of 836 matches played between 1930 and 2014. Mexico, Italy, Germany, France, and Brazil have equally held the most FIFA World Cup tournaments, and Mexico City has hosted the highest number of matches, 23. Mexico's stadium, Estadio Azteca has held 19 of these matches, making it the stadium with the highest count of FIFA matches. These calculations were performed using group-by and aggregation as demonstrated in (Fig. 4) below.



```
highcountcity = wc_attendance.groupby('City').City.agg('count').max()
highnamecity  = "Mexico City"

highcountstdm = wc_attendance.groupby('Stadium').Stadium.agg('count').max()
highnamestdm  = "Estadio Azteca"

highcountcnty = wc_hosts.groupby('Host').Host.agg('count').max()
highnamecnty  = "Mexico, Italy, Germany, France, and Brazil"
```

Fig 4.

B. Graphs

(Fig. 5), drawn using matplotlib, shows the attendance over the years. It demonstrates an upward trend between 1930 and 2014. Most matches were held on Sundays with the lowest count being on Mondays, followed by Friday (Fig. 6 and 7). However, despite the low number of matches being held on Fridays historically, the matches on Fridays are very well attended as the second highest. Saturday matches have the highest overall attendance (Fig. 8). Finals are also highly attended matches, and over the years, matches with Brazil or Germany playing are well attended.

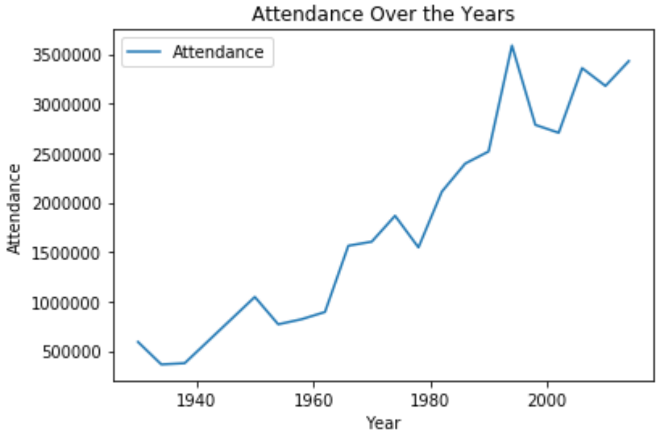


Fig 5.

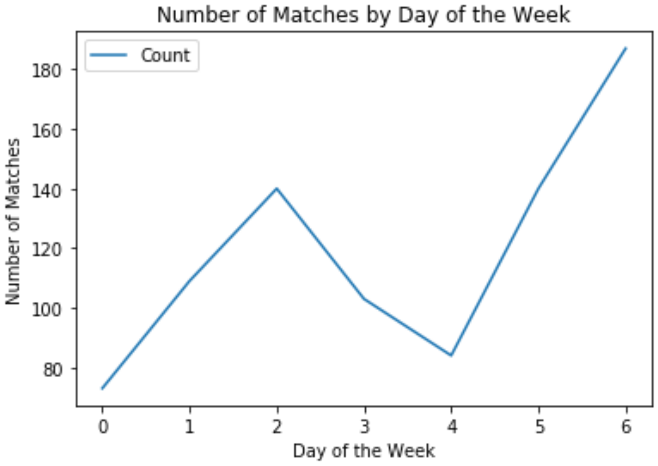


Fig 6.

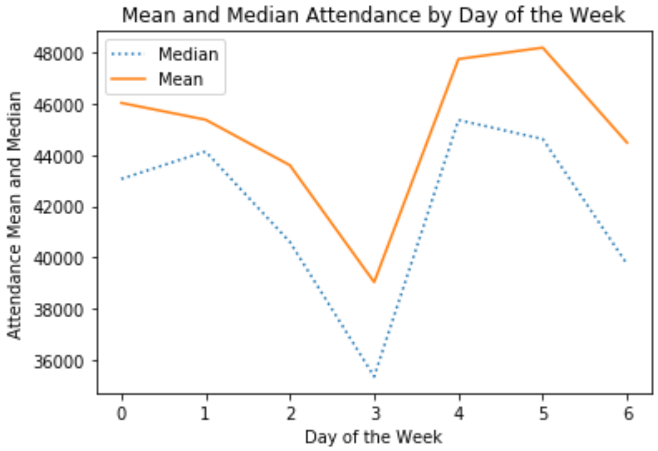


Fig 7.

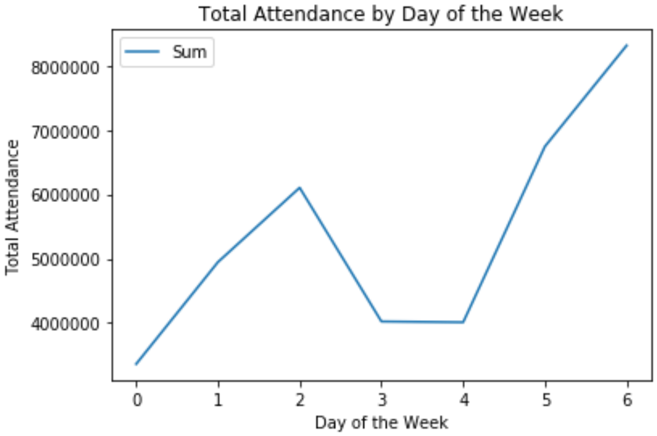


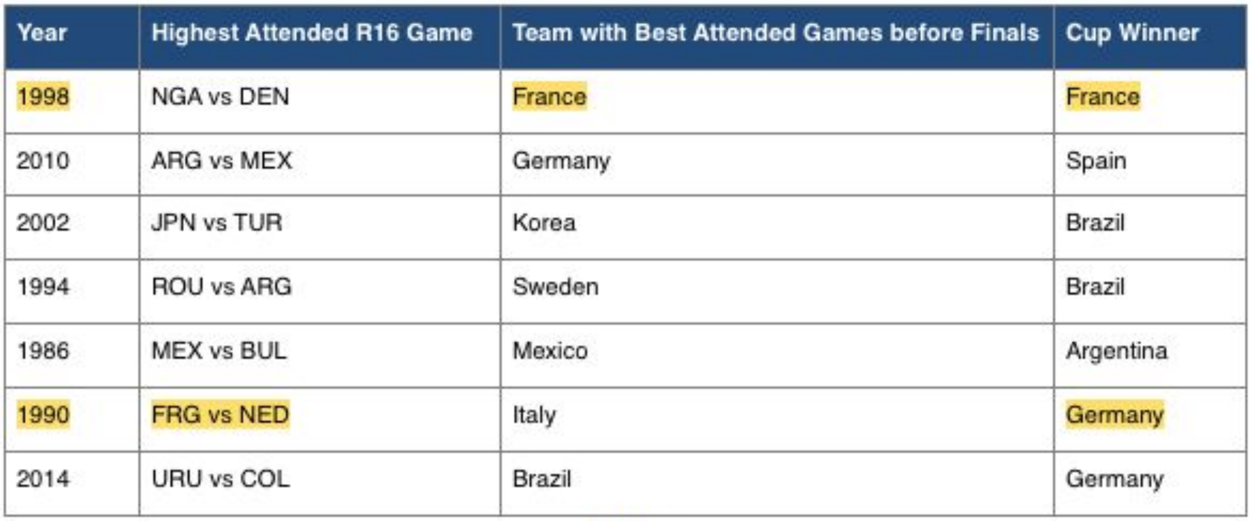
Fig 8.

C. Correlations / Prediction Ability

An analysis of interest was to determine whether attendance in earlier matches can predict a winner in later games. That is, do fans have an intuition, which translates into attending the matches of teams that will win. This analysis was done for seven games over the years, and was determined to have no significant prediction relation with the overall winner of the FIFA cup for that year (Table 1).

For attendance during the Round of 16 stage, only one of the seven randomly chosen dates (1990) being somewhat promising as a predictor.

The total attendance of the FIFA Cup Attendance that year leading up to the Final was also considered. Of the sample years taken into consideration, only 1998 seems promising, however, this match was held in France. There seemed to a higher correlation between host country and attendance of matches for the domestic team, that attendance of matches and FIFA winner.



Year	Highest Attended R16 Game	Team with Best Attended Games before Finals	Cup Winner
1998	NGA vs DEN	France	France
2010	ARG vs MEX	Germany	Spain
2002	JPN vs TUR	Korea	Brazil
1994	ROU vs ARG	Sweden	Brazil
1986	MEX vs BUL	Mexico	Argentina
1990	FRG vs NED	Italy	Germany
2014	URU vs COL	Brazil	Germany

Table 1

IV. Match Performance Analysis

A. Statistics and Graphs

An analysis of the matches, and performance of the teams was also done using the dataset. Brazil, Germany, Italy, and Argentina were found to have played in FIFA World Cup matches the highest number of times as shown in (Fig. 9).

To determine which team could be considered the best, three different approaches were attempted, and combined:

The first two approaches were based on the team's rank placement in the FIFA World cup tournament. Brazil, Germany, Italy, and Argentina were found to place highly most frequently, as shown in (Fig. 10). In 1998 the number of qualifying teams allowed was changed (it was increased to 32). If the ranks post-change are considered, Germany tops the list as shown in (Fig 11).

The third method employed in this report, examined the teams with the most goals, overall, and per match. The goals scored were combined, and the finding was that Germany, Brazil, Argentina and Italy, were amongst the top in terms of overall goals (Fig. 12). Hungry, Brazil, Germany and Turkey lead in terms of highest average goals per match (Fig. 13).

B. Analysis

Another analysis, (Fig. 14), shows teams with the most potential to turn a game around (called reversers in this report). These are teams who may start off seeming as the losing team, but ultimately are victors. There are 38 matches of this type in the history of FIFA that was analysed. Amongst them, Brazil was noted to appear 6 times on the list.

C. Prediction: Qualifying Teams

Three methods of determining the best FIFA teams were demonstrated above: (1) the top three teams in terms of placement in historical FIFA World Cups (1930 - 2014), (2) the top three teams in terms of placement in historical FIFA World Cups (1998 - 2014), (3) the top 10 teams that scored the most goals per game in FIFA World Cup matched. A combination of these can be used to determine the best teams, and would-be qualifiers. A predictive list of 14 of the 32 could be: Argentina, Brazil, Cuba, Denmark, England, France, Germany, Hungary, Italy, Netherlands, Portugal, Spain, Turkey, Uruguay. (Table 2) shows the 2018 qualifiers, with highlighted teams being the predictions. The full-list prediction was 56.25% on target (18 of 32 correct).

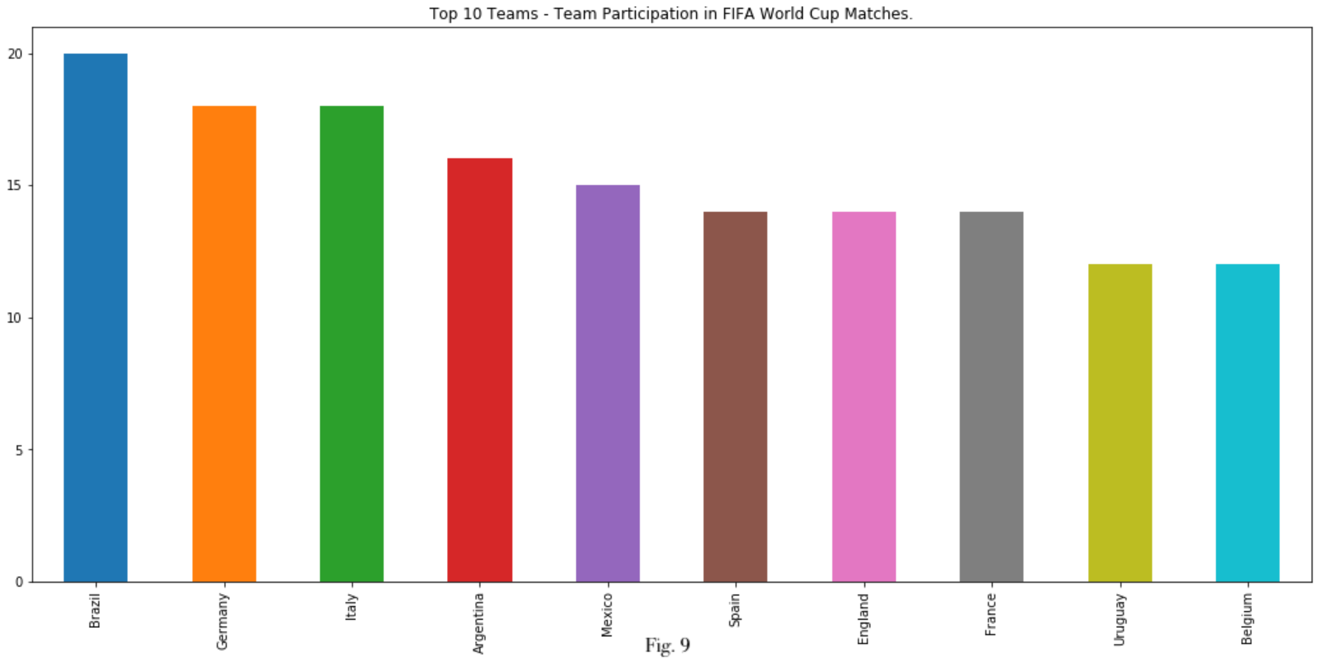


Fig. 9

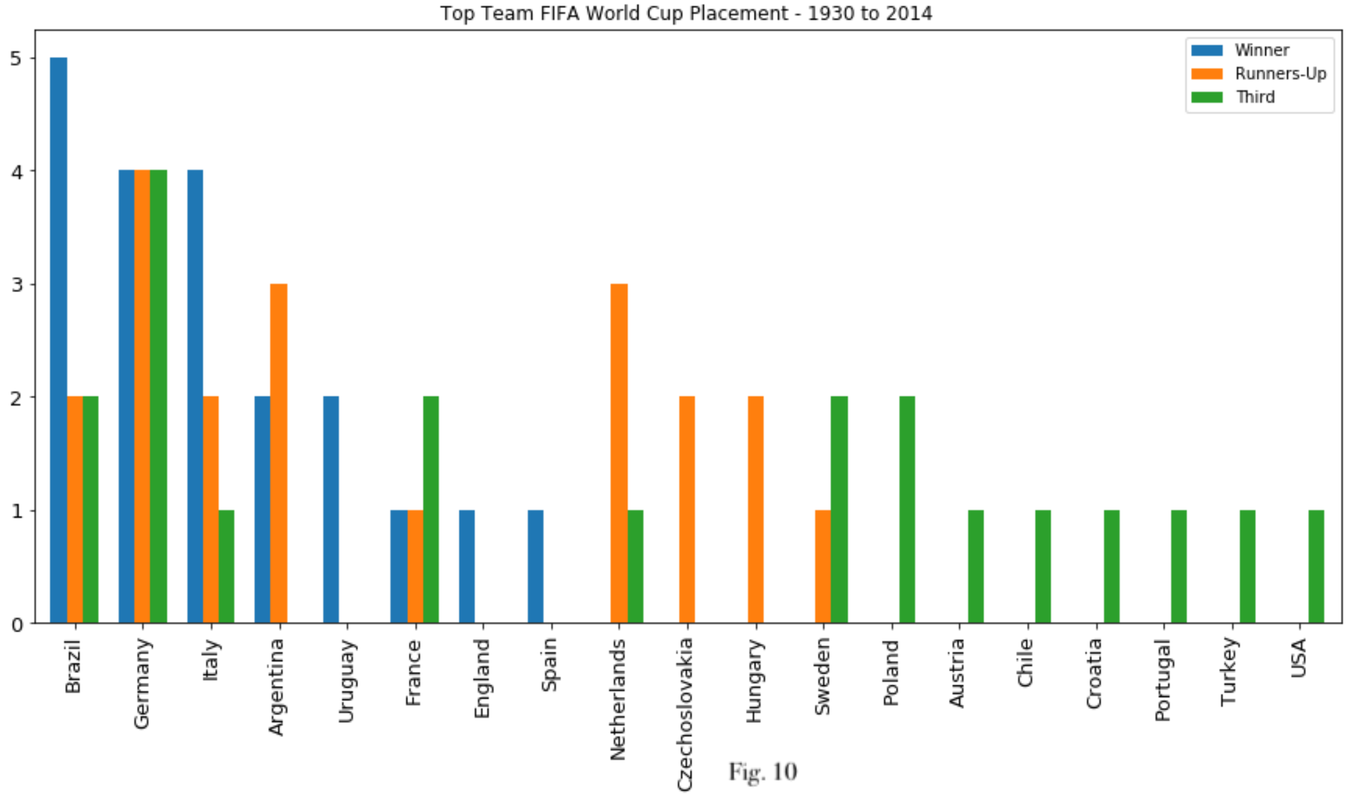

Top Team FIFA World Cup Placement - 1930 to 2014
Winner
Runners-Up
Third
0
1
2
3
4
5
Brazil
Germany
Italy
Argentina
Uruguay
France
England
Spain
Netherlands
Czechoslovakia
Hungary
Sweden
Poland
Austria
Chile
Croatia
Portugal
Turkey
USA

Fig. 10

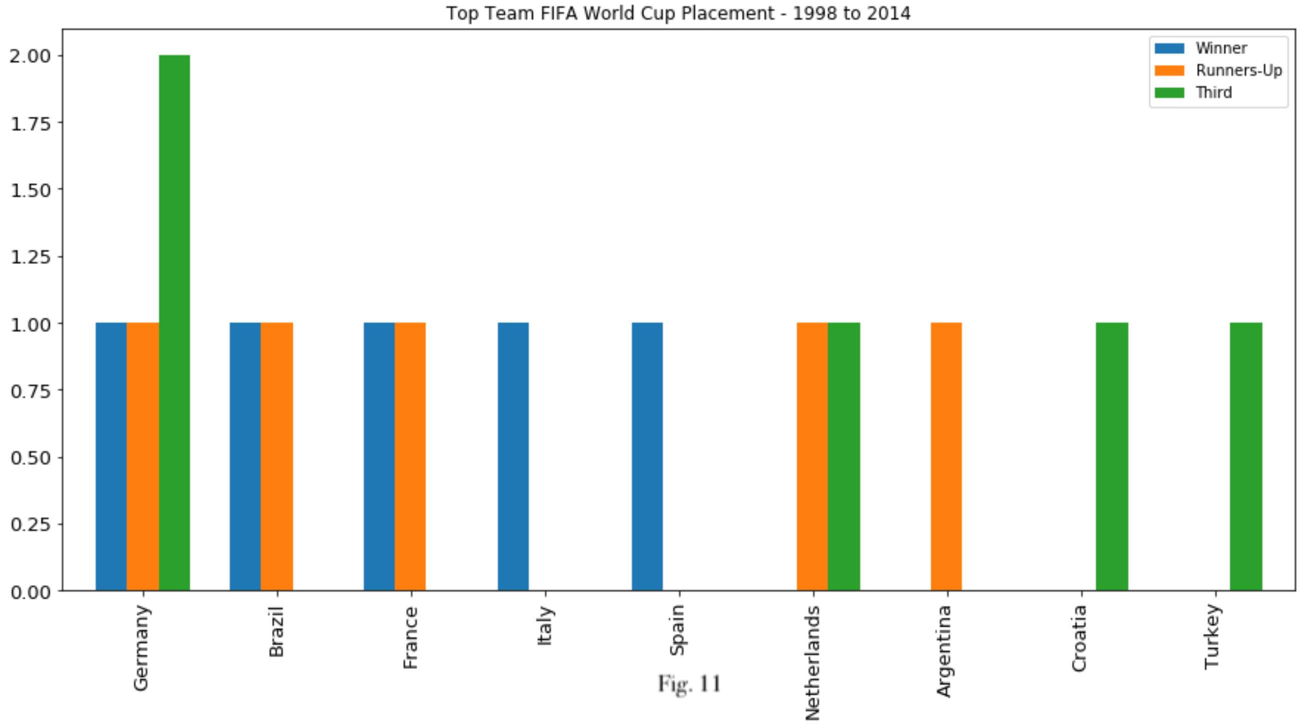

Top Team FIFA World Cup Placement - 1998 to 2014
Winner
Runners-Up
Third
0.00
0.25
0.50
0.75
1.00
1.25
1.50
1.75
2.00
Germany
Brazil
France
Italy
Spain
Netherlands
Argentina
Croatia
Turkey

Fig. 11

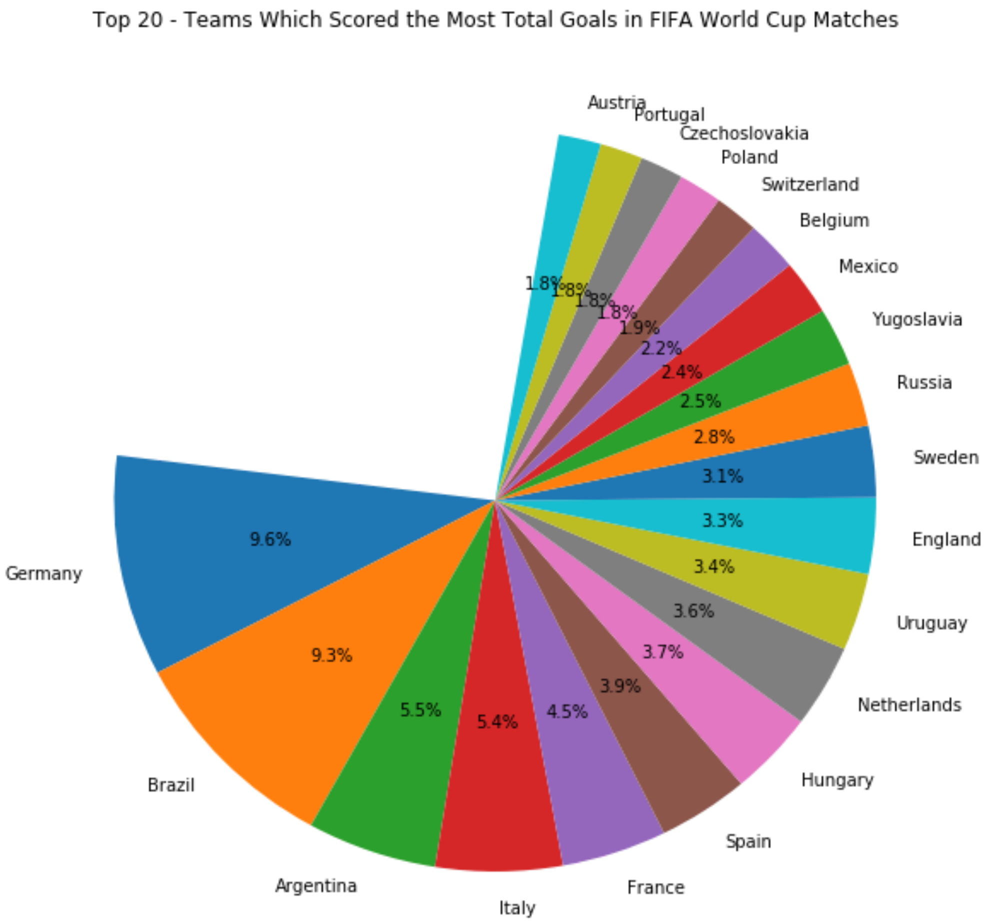

Top 20 - Teams Which Scored the Most Total Goals in FIFA World Cup Matches
Austria
Portugal
Czechoslovakia
Poland
Switzerland
Belgium
Mexico
Yugoslavia
Russia
Sweden
England
Uruguay
Netherlands
Hungary
Spain
France
Italy
Argentina
Brazil
Germany
1.8%
1.8%
1.8%
1.8%
1.9%
2.2%
2.4%
2.5%
2.8%
3.1%
3.3%
3.4%
3.6%
3.7%
3.9%
4.5%
5.4%
5.5%
9.3%
9.6%

Fig. 12

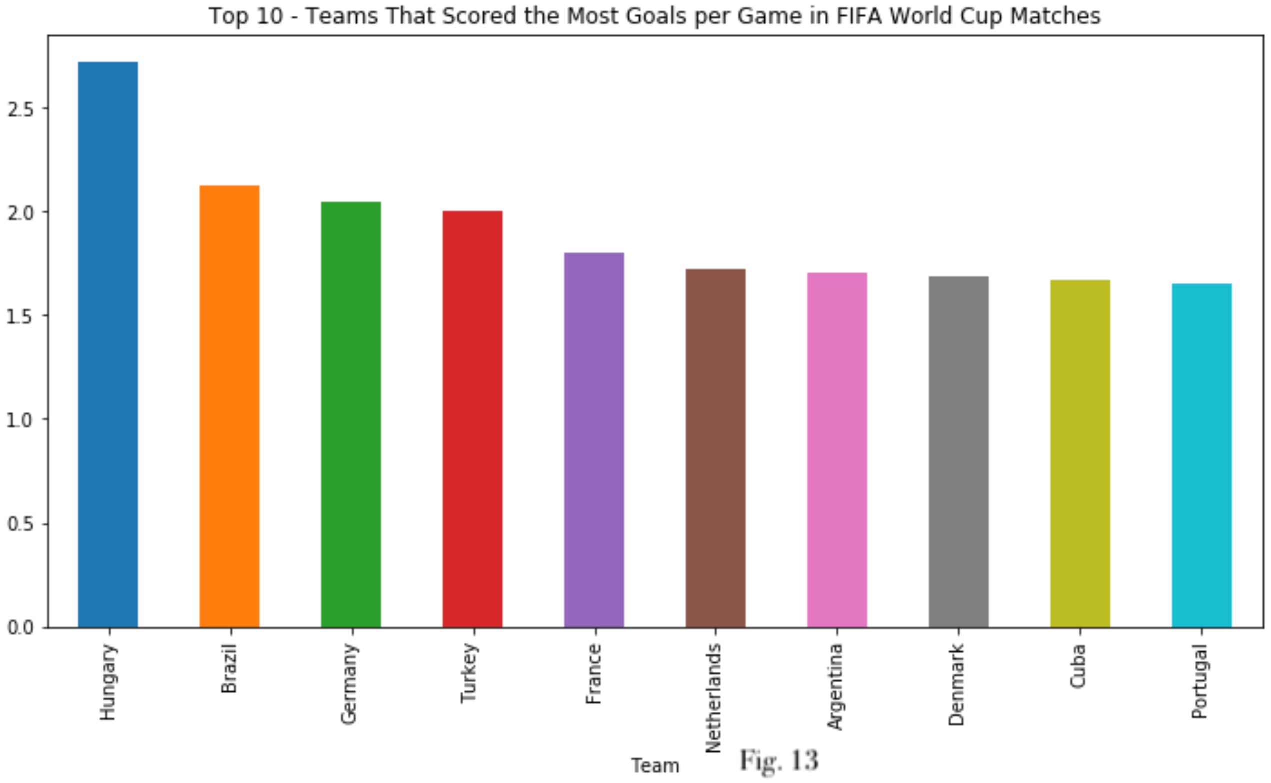

Top 10 - Teams That Scored the Most Goals per Game in FIFA World Cup Matches
2.5
2.0
1.5
1.0
0.5
0.0
Hungary
Brazil
Germany
Turkey
France
Netherlands
Argentina
Denmark
Cuba
Portugal
Team

Fig. 13

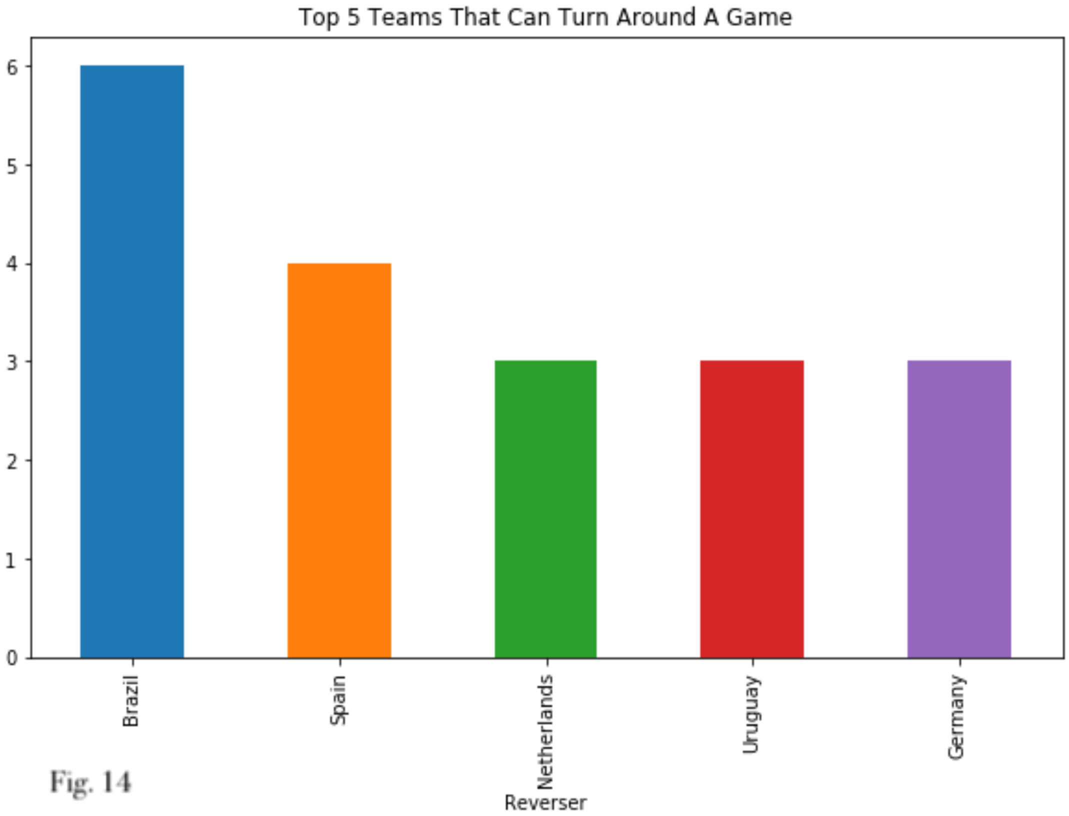
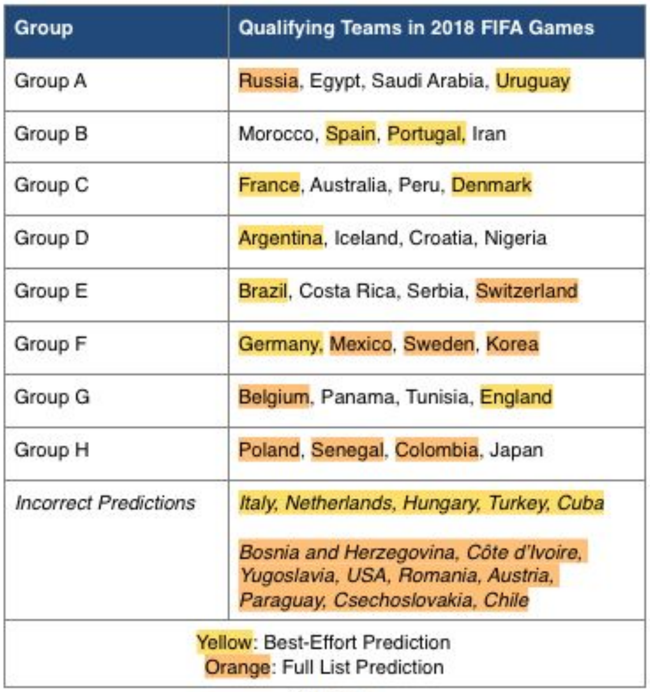


Fig. 14



Group	Qualifying Teams in 2018 FIFA Games
Group A	Russia, Egypt, Saudi Arabia, Uruguay
Group B	Morocco, Spain, Portugal, Iran
Group C	France, Australia, Peru, Denmark
Group D	Argentina, Iceland, Croatia, Nigeria
Group E	Brazil, Costa Rica, Serbia, Switzerland
Group F	Germany, Mexico, Sweden, Korea
Group G	Belgium, Panama, Tunisia, England
Group H	Poland, Senegal, Colombia, Japan
Incorrect Predictions	*Italy, Netherlands, Hungary, Turkey, Cuba* *Bosnia and Herzegovina, Côte d'Ivoire, Yugoslavia, USA, Romania, Austria, Paraguay, Csechoslovakia, Chile*
Yellow: Best-Effort Prediction Orange: Full List Prediction	

Table 2

V. Cup Analysis & Prediction

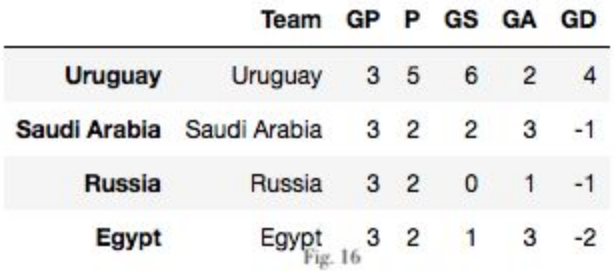
A. Statistics: 2018 Participants Quick Facts

Of the teams participating in the 2018 FIFA World Cup, four of them have never won a match, namely, Egypt, Iran, Iceland and Panama. Of these, Egypt and Iran have played in the FIFA World Cups before. Iceland and Panama are newcomers to the FIFA World Cup tournament. The top twenty match winners include Germany, Brazil, and Argentina who are historically known for high performance in the FIFA World Cup.

B. Prediction: Predicting The Outcomes Of Matches

The remainder of this report will be based on the flow shown in (Fig. 15). The aim here is to demonstrate the likelihood of arriving at the correct winner, now that the results at each stage of the 2018 FIFA World Cup are known. The predictions made will compared with the actual outcome at each round.

In the previous section, the historical match performance was used to predict which teams may qualify. Now, given the full list of 32 teams that did qualify, the next step is to determine which of these are likely to move onto the next round (Round of 16) from each Group A - H. This was done by first calculating the ratio of the goals scored vs goals against for each of the teams. This was then used to be create a method for simulating the outcome of a match. (Fig. 16) shows an example of the qualification rates for Group A.

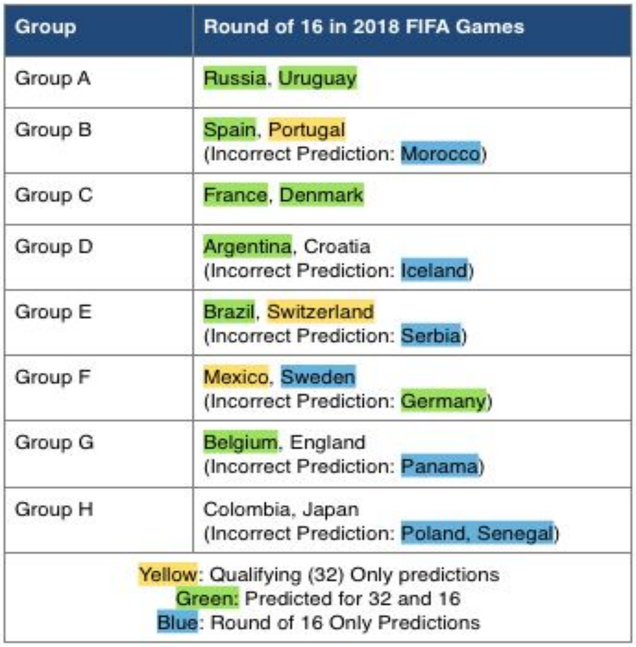


	Team	GP	P	GS	GA	GD
Uruguay	Uruguay	3	5	6	2	4
Saudi Arabia	Saudi Arabia	3	2	2	3	-1
Russia	Russia	3	2	0	1	-1
Egypt	Egypt	3	2	1	3	-2

Fig. 16

This method was repeated for Groups B - H to select the top two performers who would most likely move on to the quarter-finals. (Table 3) shows the proposed round of 16 compared to the actual 2018 FIFA World Cup round of 16. The highlighted teams are the predictions.

If the round of 16 predictions were supplemented with the predictions for the qualifying teams, meaning it was predicted in both rounds, as shown in green, the predictions would be at best 50% on target [3]. The prediction accuracy increases to 56.25% if the old results are discarded, and the correct 2018 qualifying teams feed into the prediction module.



Group	Round of 16 in 2018 FIFA Games
Group A	Russia, Uruguay
Group B	Spain, Portugal (Incorrect Prediction: Morocco)
Group C	France, Denmark
Group D	Argentina, Croatia (Incorrect Prediction: Iceland)
Group E	Brazil, Switzerland (Incorrect Prediction: Serbia)
Group F	Mexico, Sweden (Incorrect Prediction: Germany)
Group G	Belgium, England (Incorrect Prediction: Panama)
Group H	Colombia, Japan (Incorrect Prediction: Poland, Senegal)
Yellow: Qualifying (32) Only predictions Green: Predicted for 32 and 16 Blue: Round of 16 Only Predictions	

Table 3

The Semi-Finals, Quarter-Finals, Finals and ultimate winner were predicted in a similar manner, each time discarding the old predictions, and using the known participants of the rounds in the calculations. (Table 4) shows the breakdown for the remaining stages of the FIFA World Cup and predicted winners of the match-ups. (Ref 1) shows the results of the 2018 FIFA World Cup as extracted from the Official FIFA website.

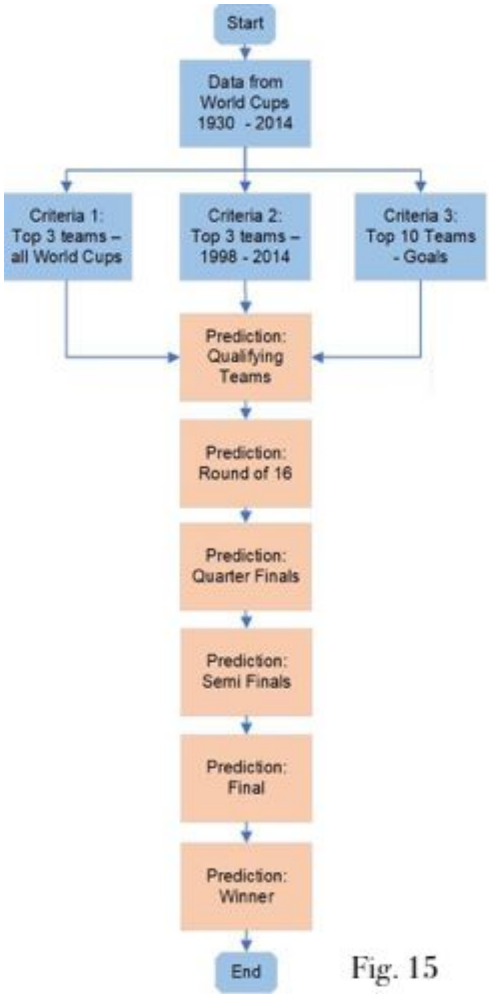
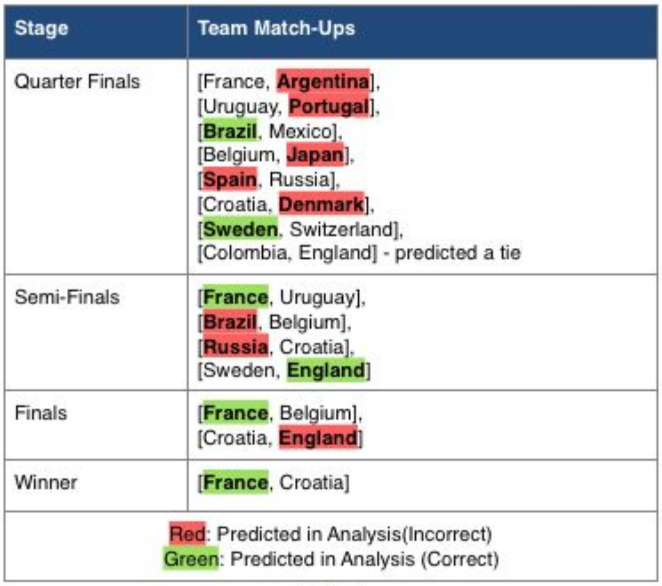
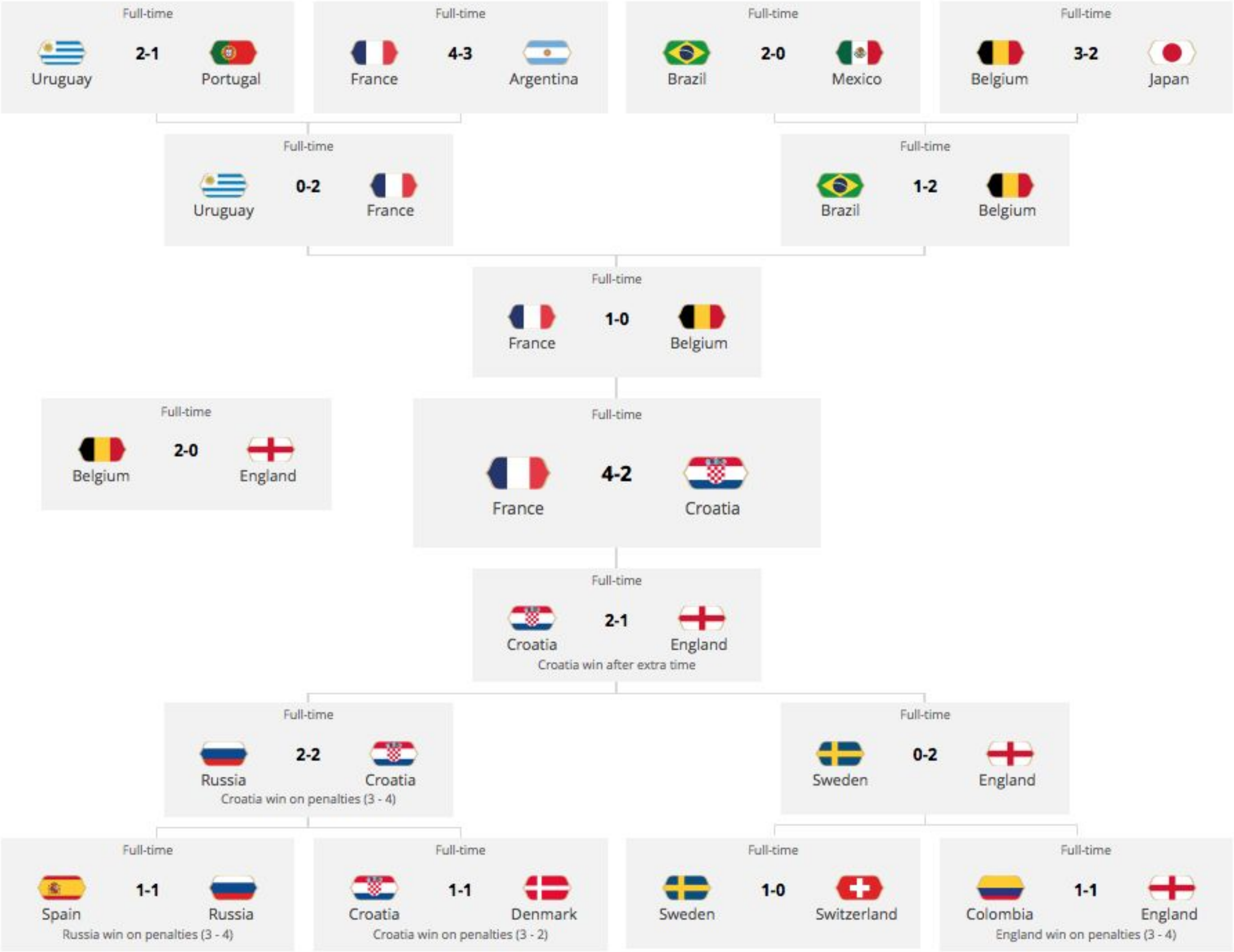

Start
Data from World Cups 1930 - 2014
Criteria 1: Top 3 teams – all World Cups
Criteria 2: Top 3 teams – 1998 - 2014
Criteria 3: Top 10 Teams - Goals
Prediction: Qualifying Teams
Prediction: Round of 16
Prediction: Quarter Finals
Prediction: Semi Finals
Prediction: Final
Prediction: Winner
End

Fig. 15



Stage	Team Match-Ups
Quarter Finals	[France, **Argentina**], [Uruguay, **Portugal**], [**Brazil**, Mexico], [Belgium, **Japan**], [**Spain**, Russia], [Croatia, **Denmark**], [**Sweden**, Switzerland], [Colombia, England] - predicted a tie
Semi-Finals	[**France**, Uruguay], [**Brazil**, Belgium], [**Russia**, Croatia], [Sweden, **England**]
Finals	[**France**, Belgium], [Croatia, **England**]
Winner	[**France**, Croatia]
Red: Predicted in Analysis(Incorrect) Green: Predicted in Analysis (Correct)	

Table 4


Full-time
Uruguay 2-1 Portugal
Full-time
France 4-3 Argentina
Full-time
Brazil 2-0 Mexico
Full-time
Belgium 3-2 Japan
Full-time
Uruguay 0-2 France
Full-time
Brazil 1-2 Belgium
Full-time
France 1-0 Belgium
Full-time
Belgium 2-0 England
Full-time
France 4-2 Croatia
Full-time
Croatia 2-1 England
Croatia win after extra time
Full-time
Russia 2-2 Croatia
Croatia win on penalties (3 - 4)
Full-time
Sweden 0-2 England
Full-time
Spain 1-1 Russia
Russia win on penalties (3 - 4)
Full-time
Croatia 1-1 Denmark
Croatia win on penalties (3 - 2)
Full-time
Sweden 1-0 Switzerland
Full-time
Colombia 1-1 England
England win on penalties (3 - 4)

Ref. 1

Conclusion

In our analysis, the match attendance before the Final was not a sufficient indicator of the winner for FIFA World Cup. However, there appears to be a relationship with the attendance of games played by countries the games are hosted in (a geographical-bias). Finals are well-attended matches, and matches with Brazil or Germany playing are also well-attended. Fridays are historically not a common day to hold matches, but draw the high crowds in comparison.

The 2018 FIFA World Cup welcomed top participants, Brazil and Germany, to this year's cup. This fact was used, in part, to determine which teams could be considered the best, thereby setting the groundwork for picking teams that would qualify for the games. Other methods used to aid in the assessment were, goals scored overall in the FIFA World Cup tournaments and, goals scored per match which participating in FIFA World Cup tournaments.

In the analysis, the features and weights that were used to determine what makes a strong team, are hand-picked by intuition, rather than learned through the data. Similar hardcoded methods were used to determine the likelihood of predicting the outcome of the matches of the World Cup tournament.

'Reversers' and games featuring this behaviour were identified in this report. The term, coined for this write-up, refers to teams who may be seemingly losing by half-time, but are ultimately the winner of the match. Matches of this type are crowd-pleasers, and 38 matching this pattern were found. Brazil and Spain were the most found to have matches of this type most frequently.

The 2018 FIFA World Cup also welcomed newcomers, Iceland, and Panama to the games. The games also gave the teams Egypt and Iran a chance to win a FIFA World Cup match post-qualification.

Simulated matches were used to predict winners in each of the rounds, with inconsequential accuracy. The ability to predict purely on the historical match performance was proven to be a poor method in practice, as shown in this report.

While researching, papers that speak on methods for predicting the outcome of organised football games were consulted. A paper by Constantinou, Fenton, and Neil published in 2012, specifically looked into forecasting the outcome of the English Premier League matching during 2010/11. [4] The method of prediction used in this paper considered both objective and subjective information, unlike our efforts, with future efforts being on revising the methods used for calculating the strength of the teams.

Many other methods have been used in the past to better rank the FIFA teams. The current FIFA 2018 ranking system is closely modelled after the Elo rating system, which is often used to determine the relative strength of players in zero-sum games. The official FIFA system has been revised three times in the past, and criticised a few times for inaccuracy, however, the methods use richer data and more features than the crude methods used in this report. [5]

It is hypothesised that given a better model of team strength, combined with a feature-identification based, or otherwise advanced method of learning what variables are important in identifying a winner, more accurate predictions could be performed. ▮

Endnotes

[1] "FIFA World Cup | Kaggle", Kaggle.com, 2018. [Online]. Available: https://www.kaggle.com/abecklas/fifa-world-cup. [Accessed: 08- Jul- 2018].

[2] T. Dasu and T. Johnson, Exploratory data mining and data cleaning. New York, NY: John Wiley & Sons Inc., 2003.

[3] This assumes that the outcomes of Round of 16 predictions are the same for the actual set of qualifying teams, and a set of predicted qualifying teams. As this may not be the case, the 50% quoted is an approximate likelihood of accuracy.

[4] Constantinou, Fenton, and Neil, "pi-football: A Bayesian network model for forecasting Association Football match outcomes," QMRO Home, 01-Dec-2012. [Online]. Available: http://qmro.qmul.ac.uk/xmlui/handle/123456789/10780. [Accessed: 01-Aug-2018].

[5] S. Price, "How FIFA's New Ranking System Will Change International Soccer," Forbes, 11-Jun-2018. [Online]. Available: https://www.forbes.com/sites/steveprice/2018/06/11/how-fifas-new-ranking-system-will-change-international-soccer/#18864e86c412. [Accessed: 02-Aug-2018].

Contributor Biographies and Contribution Summary

Xiaming Gu (Caroline)

Contribution: Match Analysis, Python Notebook Compilation.

Caroline has a B.CS in Computer Science and Technology, and several years of experience in software developing (hands-on program and mobile phone system developing). Most recently, she was a part of a team developing robotics and participated in the development of Natural Language Processing in Chinese. She has a passion for the NLP and machine learning, and is now working towards a Data Science graduate certification at the University of Toronto.

Muhammad Rizwan Kalim (Rizwan)

Contribution: Data Preparation, Cup Analysis.

Rizwan is an accomplished Information Security and IT Risk Management professional currently working for TD Bank. He holds a Masters degree in Computer Science coupled with certifications of CISA (Certified Information Security Auditor), CISSP (Certified Information Systems Security Professional), and CCSP (Certified Cloud Security Professional). His interests include the application of Artificial Intelligence and Data Sciences in the field of Information Security – a secure digital world for all – and he intends to continue upgrading his knowledge and skills in this domain after building a platform in this course

Jevonne E. Peters (Jevi)

Contribution: Attendance Analysis, Report Write-up, Presentation Design

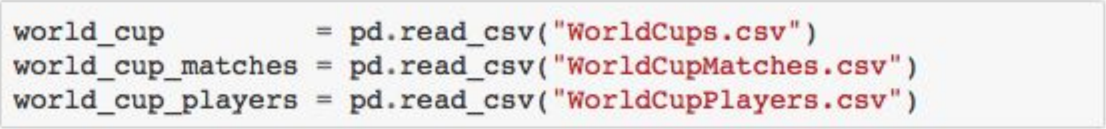
Jevi is a Business Informatics B.ASc. McMaster University graduate, and studied Machine Learning and Artificial Intelligence (Prof. Certification) at MIT. She has worked in the IT industry over the years as a Business Intelligence

Analyst, an Ethical Hacker (IT Security Consultant) and a Software Programmer, and currently works at the TRU research institute at University of Toronto. Hailed from a family of great writers, she enjoys photography, poetry and prose, and pursued the study of Art & Design at the GBC School of Design, graduating with honours. In her spare time, she works towards a Data Science graduate certification at the University of Toronto, and for a double-concentration certification in Digital Media and Social Innovation Design at OCAD U.

Appendix

Appendix - Python Code Samples

Data Collection



```
world_cup         = pd.read_csv("WorldCups.csv")
world_cup_matches = pd.read_csv("WorldCupMatches.csv")
world_cup_players = pd.read_csv("WorldCupPlayers.csv")
```

Data Preparation

Description

A. Completeness of the Data 1. Treatment of Null Rows / Cells
2. Treatment of Duplicate Rows
1. 'Missing Data' vs 'Not Applicable'

B. Anomalies

1. Wrong Data Type
2. Inconsistency in Names/Spellings and Mojibake
3. Foreign Language Characters

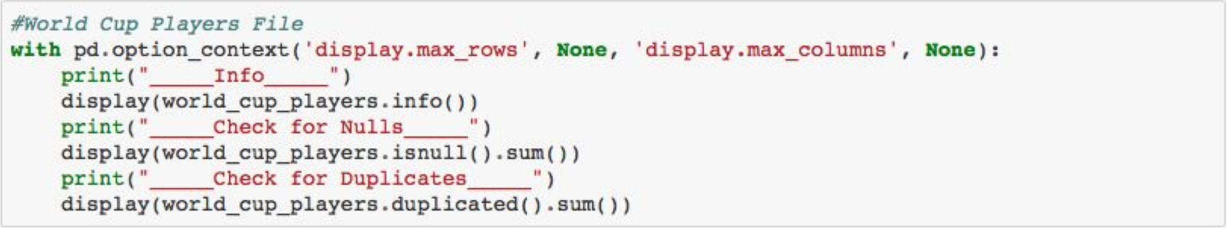
C. Data Tidiness

D. Data Integrity

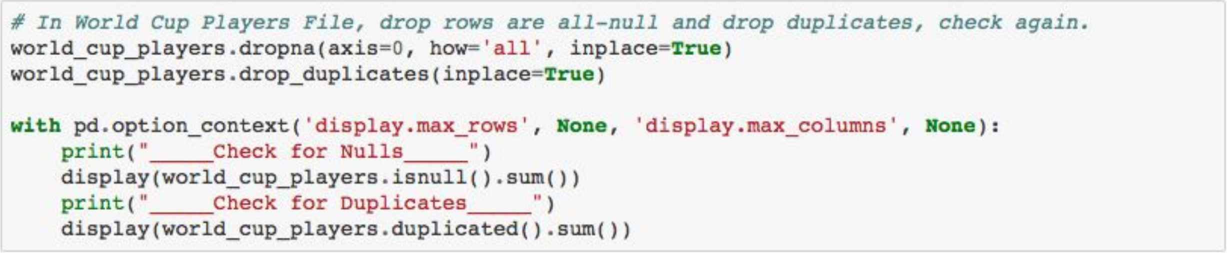
1. Basic Trends and Correlation
2. Outlier Identification

A. Completeness of the Data

1/2. Treatment of Null and Duplicate Rows



```
#World Cup Players File
with pd.option_context('display.max_rows', None, 'display.max_columns', None):
    print("_____Info_____")
    display(world_cup_players.info())
    print("_____Check for Nulls_____")
    display(world_cup_players.isnull().sum())
    print("_____Check for Duplicates_____")
    display(world_cup_players.duplicated().sum())
```

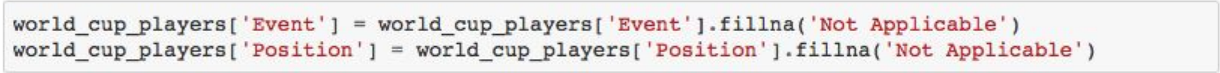


```
# In World Cup Players File, drop rows are all-null and drop duplicates, check again.
world_cup_players.dropna(axis=0, how='all', inplace=True)
world_cup_players.drop_duplicates(inplace=True)

with pd.option_context('display.max_rows', None, 'display.max_columns', None):
    print("_____Check for Nulls_____")
    display(world_cup_players.isnull().sum())
    print("_____Check for Duplicates_____")
    display(world_cup_players.duplicated().sum())
```

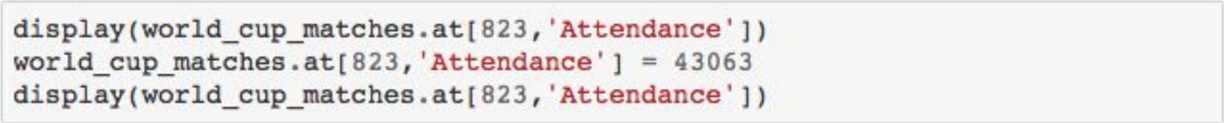
…code omitted...

A3

3. 'Missing Data' vs 'Not Applicable'?



```
world_cup_players['Event'] = world_cup_players['Event'].fillna('Not Applicable')
world_cup_players['Position'] = world_cup_players['Position'].fillna('Not Applicable')
```

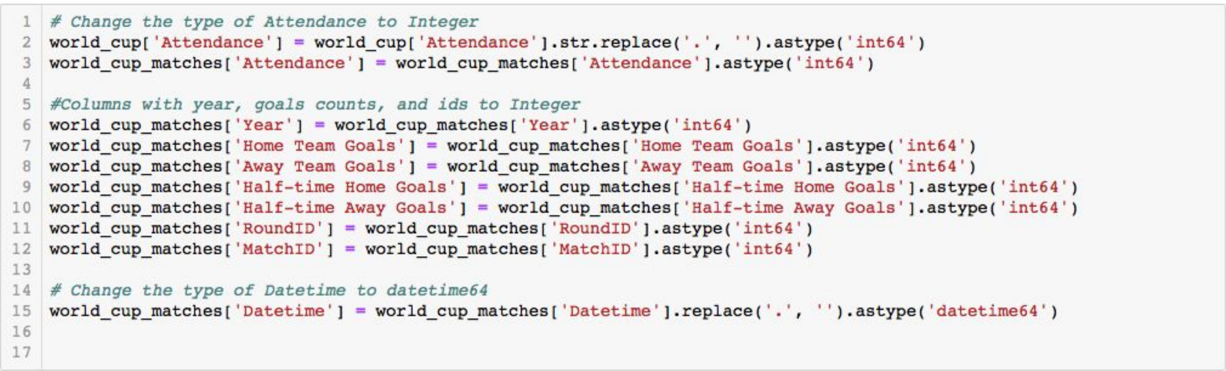


```
display(world_cup_matches.at[823,'Attendance'])
world_cup_matches.at[823,'Attendance'] = 43063
display(world_cup_matches.at[823,'Attendance'])
```

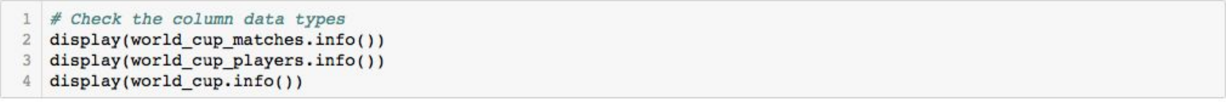
B. Data Anomalies

Anomaly # 1 - Wrong Data Type

'Attendance' columns were imported as 'object', but is better suited as an 'integer'.
Columns with year, goals counts, and IDs are also better suited as 'integers'
'Datetime' columns were imported 'object', but is better suited as an 'datetime'.



```
# Change the type of Attendance to Integer
world_cup['Attendance'] = world_cup['Attendance'].str.replace('.', '').astype('int64')
world_cup_matches['Attendance'] = world_cup_matches['Attendance'].astype('int64')

#Columns with year, goals counts, and ids to Integer
world_cup_matches['Year'] = world_cup_matches['Year'].astype('int64')
world_cup_matches['Home Team Goals'] = world_cup_matches['Home Team Goals'].astype('int64')
world_cup_matches['Away Team Goals'] = world_cup_matches['Away Team Goals'].astype('int64')
world_cup_matches['Half-time Home Goals'] = world_cup_matches['Half-time Home Goals'].astype('int64')
world_cup_matches['Half-time Away Goals'] = world_cup_matches['Half-time Away Goals'].astype('int64')
world_cup_matches['RoundID'] = world_cup_matches['RoundID'].astype('int64')
world_cup_matches['MatchID'] = world_cup_matches['MatchID'].astype('int64')

# Change the type of Datetime to datetime64
world_cup_matches['Datetime'] = world_cup_matches['Datetime'].replace('.', '').astype('datetime64')


```

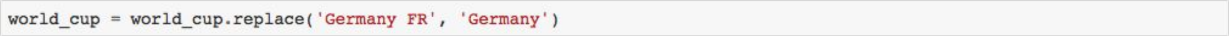


```
# Check the column data types
display(world_cup_matches.info())
display(world_cup_players.info())
display(world_cup.info())
```

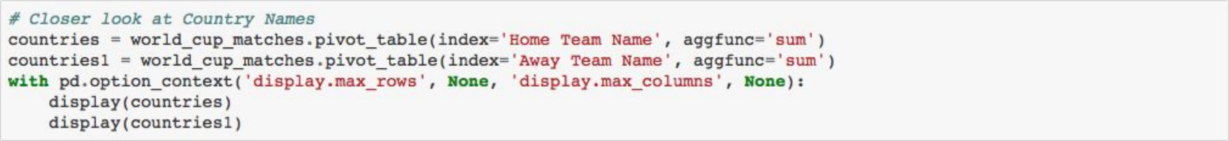
Anomaly # 2 - Inconsistency in Names/Spellings and Mojibake

Tables were parsed and reviewed for inconsistency in names and spellings or mojibake (garbled text) e.g. Under the 'Winner' column, there are two different names for Germany: 'Germany FR' and 'Germany'.

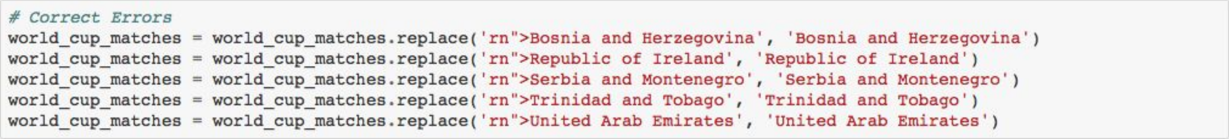
…code omitted...



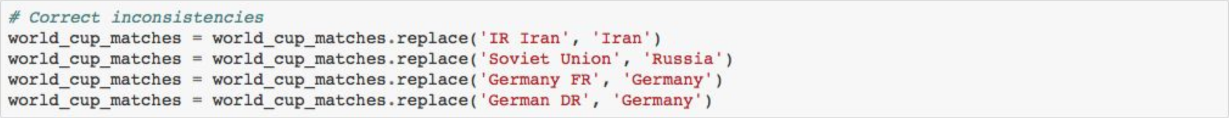
```
world_cup = world_cup.replace('Germany FR', 'Germany')
```



```
# Closer look at Country Names
countries = world_cup_matches.pivot_table(index='Home Team Name', aggfunc='sum')
countries1 = world_cup_matches.pivot_table(index='Away Team Name', aggfunc='sum')
with pd.option_context('display.max_rows', None, 'display.max_columns', None):
    display(countries)
    display(countries1)
```



```
# Correct Errors
world_cup_matches = world_cup_matches.replace('rn">Bosnia and Herzegovina', 'Bosnia and Herzegovina')
world_cup_matches = world_cup_matches.replace('rn">Republic of Ireland', 'Republic of Ireland')
world_cup_matches = world_cup_matches.replace('rn">Serbia and Montenegro', 'Serbia and Montenegro')
world_cup_matches = world_cup_matches.replace('rn">Trinidad and Tobago', 'Trinidad and Tobago')
world_cup_matches = world_cup_matches.replace('rn">United Arab Emirates', 'United Arab Emirates')
```



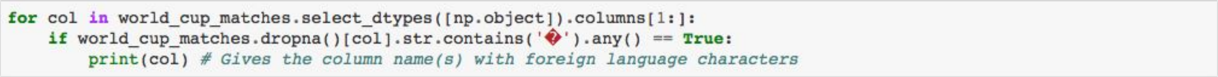
```
# Correct inconsistencies
world_cup_matches = world_cup_matches.replace('IR Iran', 'Iran')
world_cup_matches = world_cup_matches.replace('Soviet Union', 'Russia')
world_cup_matches = world_cup_matches.replace('Germany FR', 'Germany')
world_cup_matches = world_cup_matches.replace('German DR', 'Germany')
```

…code omitted...

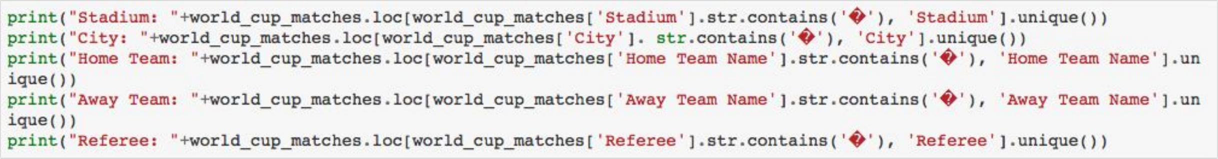
Anomaly # 3 - Foreign Language Characters

Foreign language characters we improperly encoded, e.g.: 'Côte d'Ivoire' which appeared as 'C�te d'Ivoire'. These were corrected using a closely related Latin character based on research information.

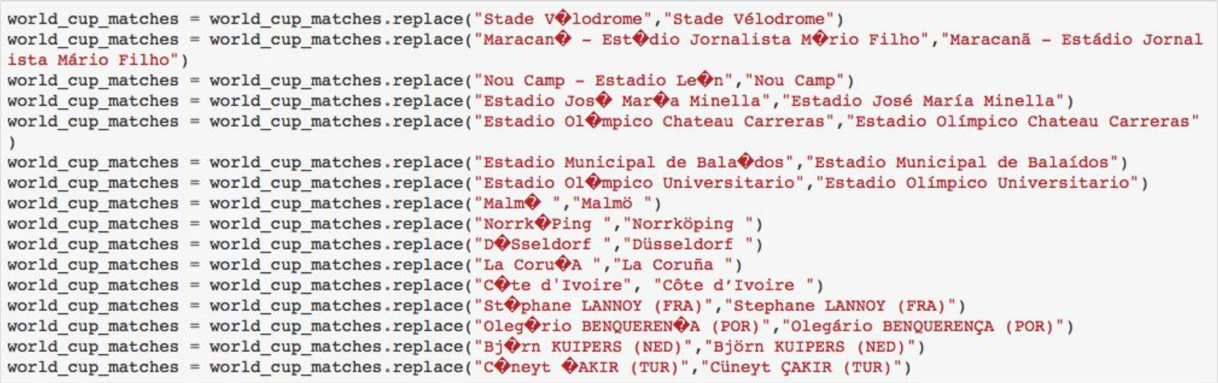
…code omitted...



```
for col in world_cup_matches.select_dtypes([np.object]).columns[1:]:
    if world_cup_matches.dropna()[col].str.contains('�').any() == True:
        print(col) # Gives the column name(s) with foreign language characters
```



```
print("Stadium: "+world_cup_matches.loc[world_cup_matches['Stadium'].str.contains('�'), 'Stadium'].unique())
print("City: "+world_cup_matches.loc[world_cup_matches['City']. str.contains('�'), 'City'].unique())
print("Home Team: "+world_cup_matches.loc[world_cup_matches['Home Team Name'].str.contains('�'), 'Home Team Name'].un
ique())
print("Away Team: "+world_cup_matches.loc[world_cup_matches['Away Team Name'].str.contains('�'), 'Away Team Name'].un
ique())
print("Referee: "+world_cup_matches.loc[world_cup_matches['Referee'].str.contains('�'), 'Referee'].unique())
```

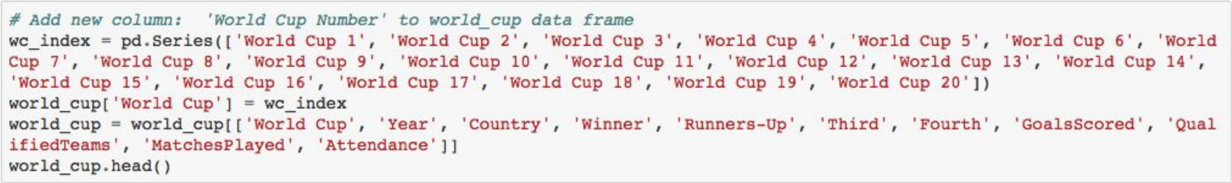


```
world_cup_matches = world_cup_matches.replace("Stade V�lodrome","Stade Vélodrome")
world_cup_matches = world_cup_matches.replace("Maracan� - Est�dio Jornalista M�rio Filho","Maracanã - Estádio Jornal
ista Mário Filho")
world_cup_matches = world_cup_matches.replace("Nou Camp - Estadio Le�n","Nou Camp")
world_cup_matches = world_cup_matches.replace("Estadio Jos� Mar�a Minella","Estadio José María Minella")
world_cup_matches = world_cup_matches.replace("Estadio Ol�mpico Chateau Carreras","Estadio Olímpico Chateau Carreras"
)
world_cup_matches = world_cup_matches.replace("Estadio Municipal de Bala�dos","Estadio Municipal de Balaídos")
world_cup_matches = world_cup_matches.replace("Estadio Ol�mpico Universitario","Estadio Olímpico Universitario")
world_cup_matches = world_cup_matches.replace("Malm� ","Malmö ")
world_cup_matches = world_cup_matches.replace("Norrk�Ping ","Norrköping ")
world_cup_matches = world_cup_matches.replace("D�Sseldorf ","Düsseldorf ")
world_cup_matches = world_cup_matches.replace("La Coru�A ","La Coruña ")
world_cup_matches = world_cup_matches.replace("C�te d'Ivoire", "Côte d’Ivoire ")
world_cup_matches = world_cup_matches.replace("St�phane LANNOY (FRA)","Stephane LANNOY (FRA)")
world_cup_matches = world_cup_matches.replace("Oleg�rio BENQUEREN�A (POR)","Olegário BENQUERENÇA (POR)")
world_cup_matches = world_cup_matches.replace("Bj�rn KUIPERS (NED)","Björn KUIPERS (NED)")
world_cup_matches = world_cup_matches.replace("C�neyt �AKIR (TUR)","Cüneyt ÇAKIR (TUR)")
```

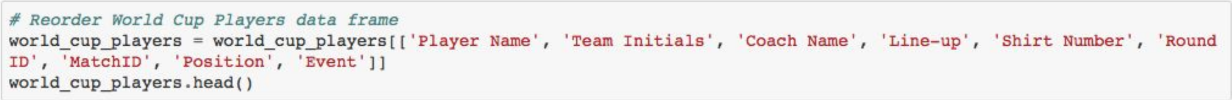
C. Data Tidiness

The following objectives were adhered to:

- Each column is a variable
- Each row is an observation
- Each type of observational unit forms a table



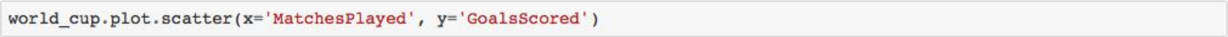
```
# Add new column:  'World Cup Number' to world_cup data frame
wc_index = pd.Series(['World Cup 1', 'World Cup 2', 'World Cup 3', 'World Cup 4', 'World Cup 5', 'World Cup 6', 'World
Cup 7', 'World Cup 8', 'World Cup 9', 'World Cup 10', 'World Cup 11', 'World Cup 12', 'World Cup 13', 'World Cup 14',
'World Cup 15', 'World Cup 16', 'World Cup 17', 'World Cup 18', 'World Cup 19', 'World Cup 20'])
world_cup['World Cup'] = wc_index
world_cup = world_cup[['World Cup', 'Year', 'Country', 'Winner', 'Runners-Up', 'Third', 'Fourth', 'GoalsScored', 'Qual
ifiedTeams', 'MatchesPlayed', 'Attendance']]
world_cup.head()
```



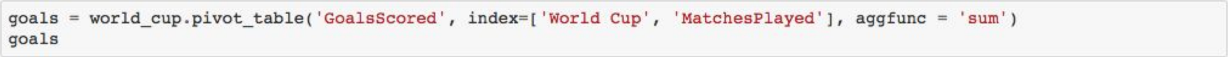
```
# Reorder World Cup Players data frame
world_cup_players = world_cup_players[['Player Name', 'Team Initials', 'Coach Name', 'Line-up', 'Shirt Number', 'Round
ID', 'MatchID', 'Position', 'Event']]
world_cup_players.head()
```

D. Data Integrity

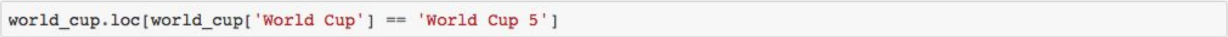
- Basic Trends and Correlations
- Outlier Identification



```
world_cup.plot.scatter(x='MatchesPlayed', y='GoalsScored')
```



```
goals = world_cup.pivot_table('GoalsScored', index=['World Cup', 'MatchesPlayed'], aggfunc = 'sum')
goals
```



```
world_cup.loc[world_cup['World Cup'] == 'World Cup 5']
```

END OF DATA PREPARATION

Data Analysis

Analysis: Attendance

Description

A. Statistics

1. How many matches have been played in all? Each year?
2. What is the city, stadium, and country with the most matches?

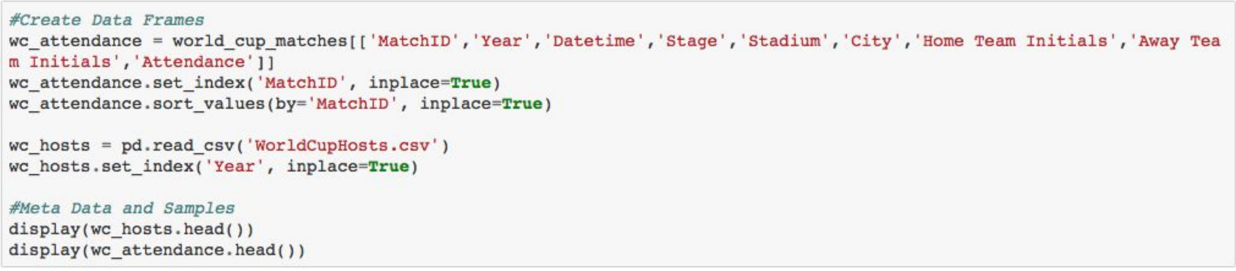
B. Graphs

1. Graph of all attendance over the years in chron order
2. Graph the attendance by day over the years

C. Correlations

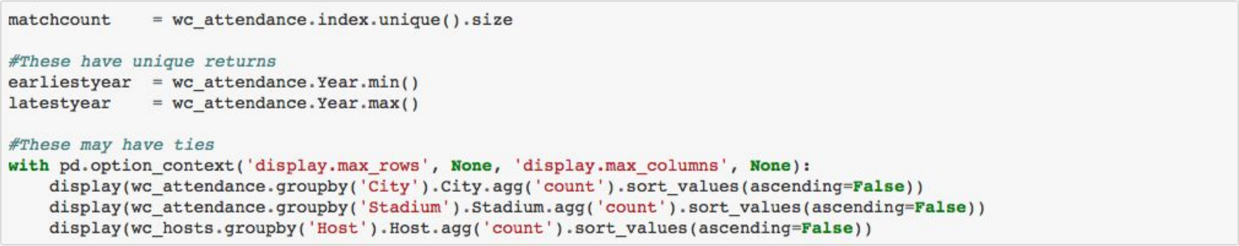
1. Determine the three highest attended matches for each cup and make simple observations.
2. Can attendance alone be used to predict a winner?

Quick Data Preparation

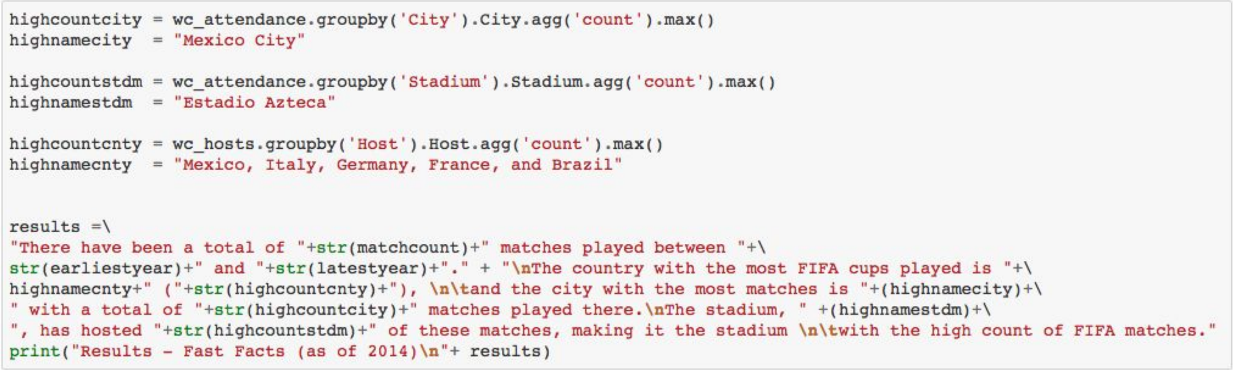


```
#Create Data Frames
wc_attendance = world_cup_matches[['MatchID','Year','Datetime','Stage','Stadium','City','Home Team Initials','Away Tea
m Initials','Attendance']]
wc_attendance.set_index('MatchID', inplace=True)
wc_attendance.sort_values(by='MatchID', inplace=True)

wc_hosts = pd.read_csv('WorldCupHosts.csv')
wc_hosts.set_index('Year', inplace=True)

#Meta Data and Samples
display(wc_hosts.head())
display(wc_attendance.head())
```

A. Statistics



```
matchcount    = wc_attendance.index.unique().size

#These have unique returns
earliestyear  = wc_attendance.Year.min()
latestyear    = wc_attendance.Year.max()

#These may have ties
with pd.option_context('display.max_rows', None, 'display.max_columns', None):
    display(wc_attendance.groupby('City').City.agg('count').sort_values(ascending=False))
    display(wc_attendance.groupby('Stadium').Stadium.agg('count').sort_values(ascending=False))
    display(wc_hosts.groupby('Host').Host.agg('count').sort_values(ascending=False))
```



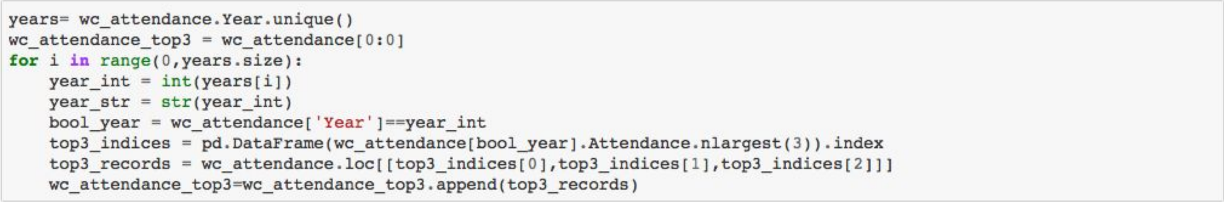
```
highcountcity = wc_attendance.groupby('City').City.agg('count').max()
highnamecity  = "Mexico City"

highcountstdm = wc_attendance.groupby('Stadium').Stadium.agg('count').max()
highnamestdm  = "Estadio Azteca"

highcountcnty = wc_hosts.groupby('Host').Host.agg('count').max()
highnamecnty  = "Mexico, Italy, Germany, France, and Brazil"


results =\
"There have been a total of "+str(matchcount)+" matches played between "+\
str(earliestyear)+" and "+str(latestyear)+"." + "\nThe country with the most FIFA cups played is "+\
highnamecnty+" ("+str(highcountcnty)+"), \n\tand the city with the most matches is "+(highnamecity)+\
" with a total of "+str(highcountcity)+" matches played there.\nThe stadium, " +(highnamestdm)+\
", has hosted "+str(highcountstdm)+" of these matches, making it the stadium \n\twith the high count of FIFA matches."
print("Results - Fast Facts (as of 2014)\n"+ results)
```

B. Graphs

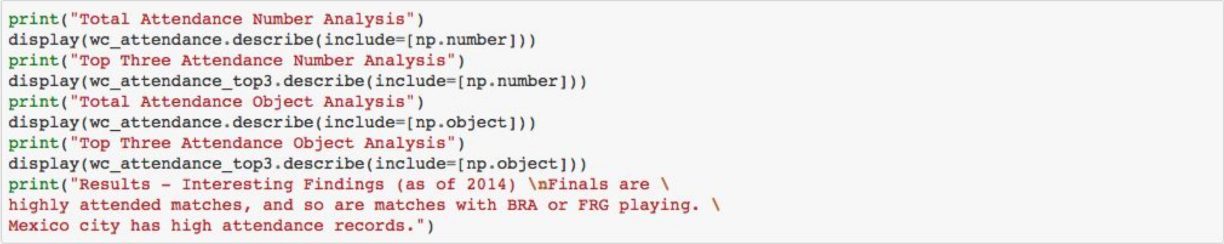
…code omitted…

C. Correlations

1. Observations of the Highest Three Matches



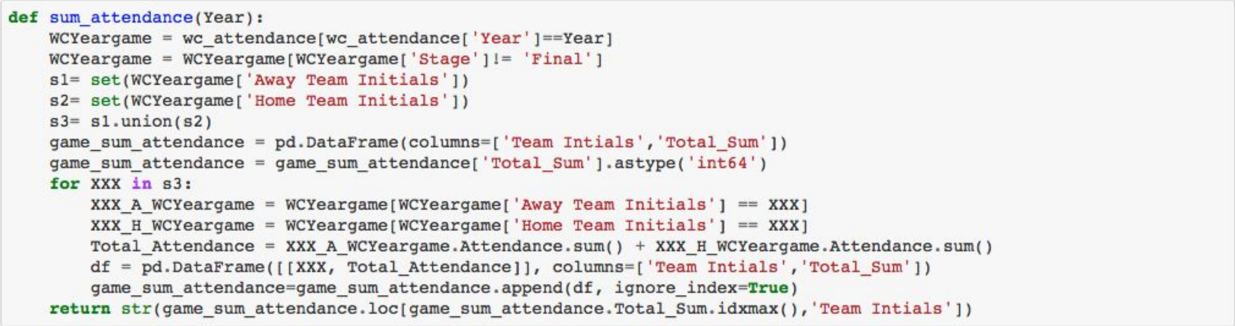
```
years= wc_attendance.Year.unique()
wc_attendance_top3 = wc_attendance[0:0]
for i in range(0,years.size):
    year_int = int(years[i])
    year_str = str(year_int)
    bool_year = wc_attendance['Year']==year_int
    top3_indices = pd.DataFrame(wc_attendance[bool_year].Attendance.nlargest(3)).index
    top3_records = wc_attendance.loc[[top3_indices[0],top3_indices[1],top3_indices[2]]]
    wc_attendance_top3=wc_attendance_top3.append(top3_records)
```



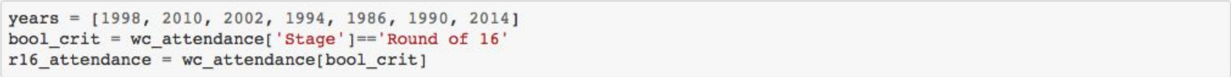
```
print("Total Attendance Number Analysis")
display(wc_attendance.describe(include=[np.number]))
print("Top Three Attendance Number Analysis")
display(wc_attendance_top3.describe(include=[np.number]))
print("Total Attendance Object Analysis")
display(wc_attendance.describe(include=[np.object]))
print("Top Three Attendance Object Analysis")
display(wc_attendance_top3.describe(include=[np.object]))
print("Results - Interesting Findings (as of 2014) \nFinals are \
highly attended matches, and so are matches with BRA or FRG playing. \
Mexico city has high attendance records.")
```

2. Can attendance in early stages of the cup alone be used to predict a winner? i.e. does the audience have a preference for the games of would be winners?

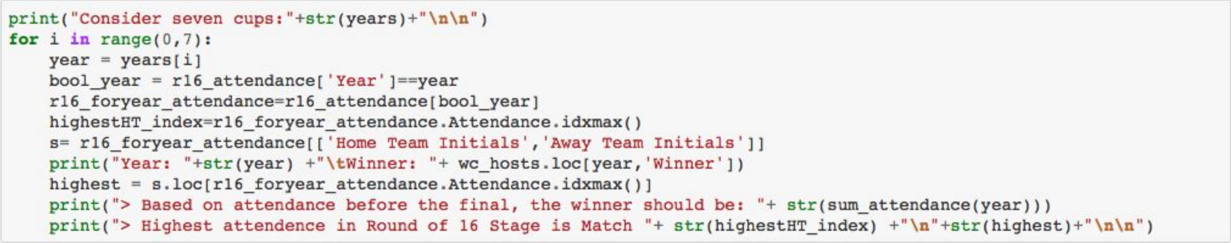
Answer: No



```
def sum_attendance(Year):
    WCYeargame = wc_attendance[wc_attendance['Year']==Year]
    WCYeargame = WCYeargame[WCYeargame['Stage']!= 'Final']
    s1= set(WCYeargame['Away Team Initials'])
    s2= set(WCYeargame['Home Team Initials'])
    s3= s1.union(s2)
    game_sum_attendance = pd.DataFrame(columns=['Team Intials','Total_Sum'])
    game_sum_attendance = game_sum_attendance['Total_Sum'].astype('int64')
    for XXX in s3:
        XXX_A_WCYeargame = WCYeargame[WCYeargame['Away Team Initials'] == XXX]
        XXX_H_WCYeargame = WCYeargame[WCYeargame['Home Team Initials'] == XXX]
        Total_Attendance = XXX_A_WCYeargame.Attendance.sum() + XXX_H_WCYeargame.Attendance.sum()
        df = pd.DataFrame([[XXX, Total_Attendance]], columns=['Team Intials','Total_Sum'])
        game_sum_attendance=game_sum_attendance.append(df, ignore_index=True)
    return str(game_sum_attendance.loc[game_sum_attendance.Total_Sum.idxmax(),'Team Intials'])
```



```
years = [1998, 2010, 2002, 1994, 1986, 1990, 2014]
bool_crit = wc_attendance['Stage']=='Round of 16'
r16_attendance = wc_attendance[bool_crit]
```



```
print("Consider seven cups:"+str(years)+"\n\n")
for i in range(0,7):
    year = years[i]
    bool_year = r16_attendance['Year']==year
    r16_foryear_attendance=r16_attendance[bool_year]
    highestHT_index=r16_foryear_attendance.Attendance.idxmax()
    s= r16_foryear_attendance[['Home Team Initials','Away Team Initials']]
    print("Year: "+str(year) +"\tWinner: "+ wc_hosts.loc[year,'Winner'])
    highest = s.loc[r16_foryear_attendance.Attendance.idxmax()]
    print("> Based on attendance before the final, the winner should be: "+ str(sum_attendance(year)))
    print("> Highest attendence in Round of 16 Stage is Match "+ str(highestHT_index) +"\n"+str(highest)+"\n\n")
```

END OF ATTENDANCE ANALYSIS

Analysis: Matches

Description

A. Statistics and Graphs

1. Which team has played the most world cup matches?
2.
3. Which team could be considered to be the best team?
4. a. Top 3 teams that have rank highly (1, 2, 3) in the the FIFA World Cup Tournament
5. b. Top 3 teams that have rank highly (1, 2, 3) in the the FIFA World Cup Tournament 1998 to 2014 (number of qualifying teams increased to 32 starting from 1998 World Cup)
6. c. Top 10 teams have achieved the most goals overall, and per match

B. Analysis

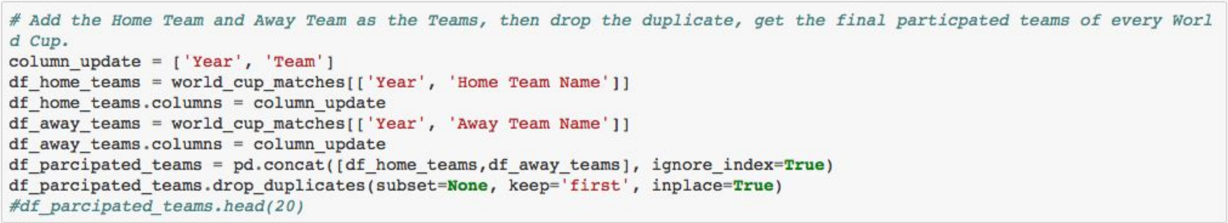
1. Which team can best turnaround the game i.e. which teams seem to be losing at half-time, but pull it together in the second half to become the winner of that match?
2. a. How many games can be marked as turnaround game
3. b. Which team get the most times to achieve turnaround
4.

C. Predicting Qualifying Teams

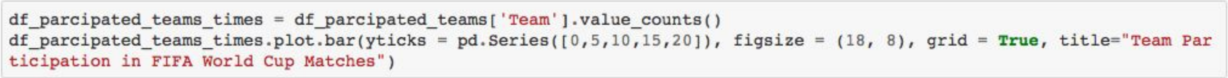
Statement on results

A. Statistics and Graphs

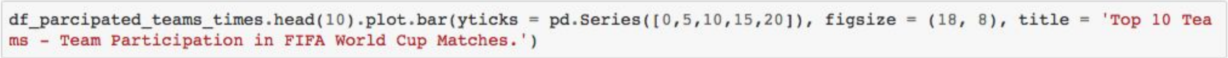
1. Team Participation in the World Cup



```
# Add the Home Team and Away Team as the Teams, then drop the duplicate, get the final particpated teams of every Worl
d Cup.
column_update = ['Year', 'Team']
df_home_teams = world_cup_matches[['Year', 'Home Team Name']]
df_home_teams.columns = column_update
df_away_teams = world_cup_matches[['Year', 'Away Team Name']]
df_away_teams.columns = column_update
df_parcipated_teams = pd.concat([df_home_teams,df_away_teams], ignore_index=True)
df_parcipated_teams.drop_duplicates(subset=None, keep='first', inplace=True)
#df_parcipated_teams.head(20)
```



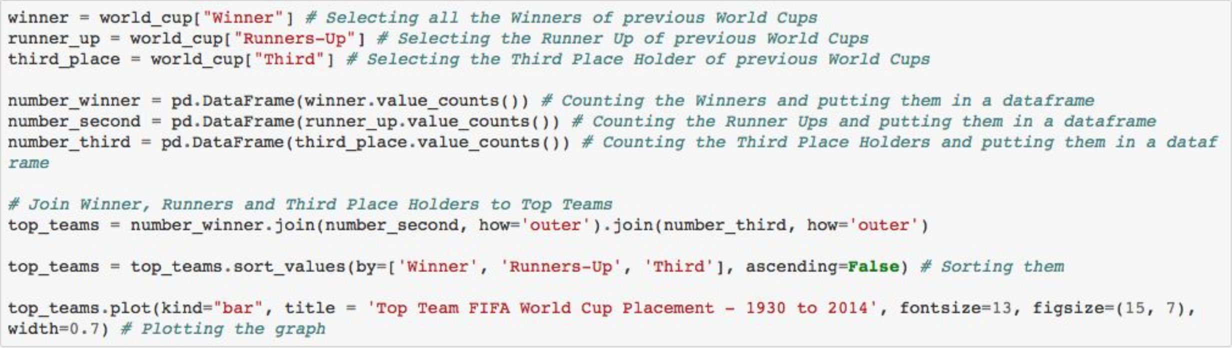
```
df_parcipated_teams_times = df_parcipated_teams['Team'].value_counts()
df_parcipated_teams_times.plot.bar(yticks = pd.Series([0,5,10,15,20]), figsize = (18, 8), grid = True, title="Team Par
ticipation in FIFA World Cup Matches")
```



```
df_parcipated_teams_times.head(10).plot.bar(yticks = pd.Series([0,5,10,15,20]), figsize = (18, 8), title = 'Top 10 Tea
ms - Team Participation in FIFA World Cup Matches.')
```

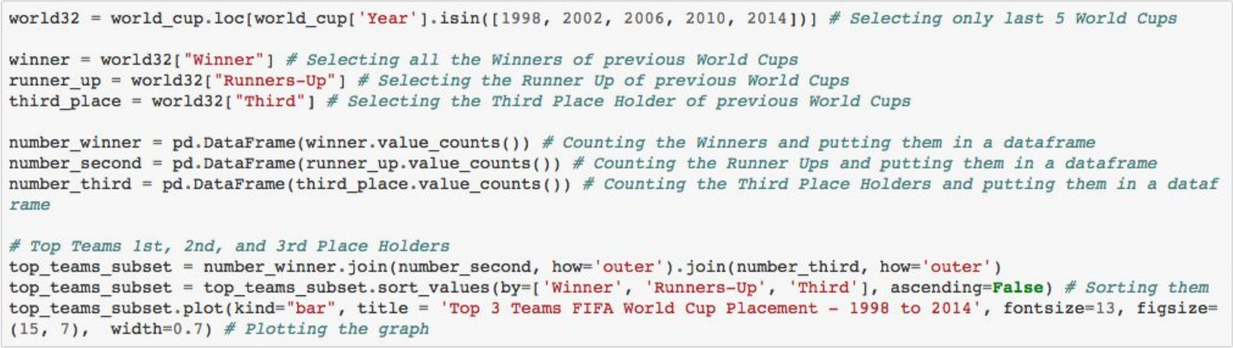
2. Which Team Could Be Considered The Best?

2a. Team Placement in the FIFA World Cups - Overall



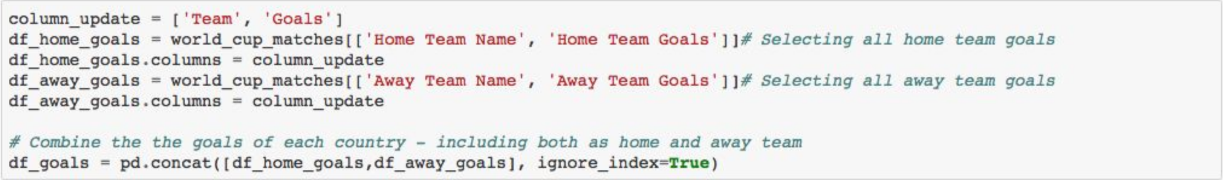
```
winner = world_cup["Winner"] # Selecting all the Winners of previous World Cups
runner_up = world_cup["Runners-Up"] # Selecting the Runner Up of previous World Cups
third_place = world_cup["Third"] # Selecting the Third Place Holder of previous World Cups

number_winner = pd.DataFrame(winner.value_counts()) # Counting the Winners and putting them in a dataframe
number_second = pd.DataFrame(runner_up.value_counts()) # Counting the Runner Ups and putting them in a dataframe
number_third = pd.DataFrame(third_place.value_counts()) # Counting the Third Place Holders and putting them in a dataf
rame

# Join Winner, Runners and Third Place Holders to Top Teams
top_teams = number_winner.join(number_second, how='outer').join(number_third, how='outer')

top_teams = top_teams.sort_values(by=['Winner', 'Runners-Up', 'Third'], ascending=False) # Sorting them

top_teams.plot(kind="bar", title = 'Top Team FIFA World Cup Placement - 1930 to 2014', fontsize=13, figsize=(15, 7),
width=0.7) # Plotting the graph
```

2b. Team Placement for World Cups - 1998 to 2014 (number of qualifying teams increased to 32 starting from 1998 World Cup)

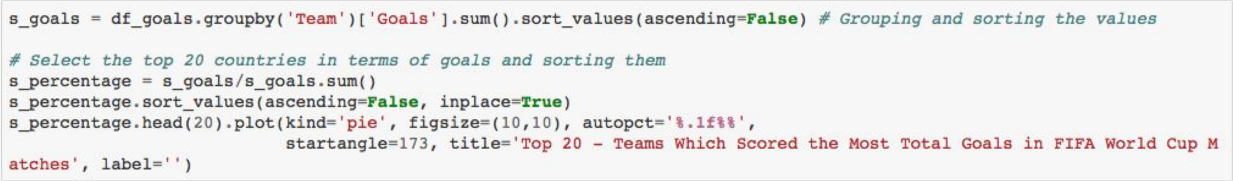


```
world32 = world_cup.loc[world_cup['Year'].isin([1998, 2002, 2006, 2010, 2014])] # Selecting only last 5 World Cups

winner = world32["Winner"] # Selecting all the Winners of previous World Cups
runner_up = world32["Runners-Up"] # Selecting the Runner Up of previous World Cups
third_place = world32["Third"] # Selecting the Third Place Holder of previous World Cups

number_winner = pd.DataFrame(winner.value_counts()) # Counting the Winners and putting them in a dataframe
number_second = pd.DataFrame(runner_up.value_counts()) # Counting the Runner Ups and putting them in a dataframe
number_third = pd.DataFrame(third_place.value_counts()) # Counting the Third Place Holders and putting them in a dataf
rame

# Top Teams 1st, 2nd, and 3rd Place Holders
top_teams_subset = number_winner.join(number_second, how='outer').join(number_third, how='outer')
top_teams_subset = top_teams_subset.sort_values(by=['Winner', 'Runners-Up', 'Third'], ascending=False) # Sorting them
top_teams_subset.plot(kind="bar", title = 'Top 3 Teams FIFA World Cup Placement - 1998 to 2014', fontsize=13, figsize=
(15, 7),  width=0.7) # Plotting the graph
```

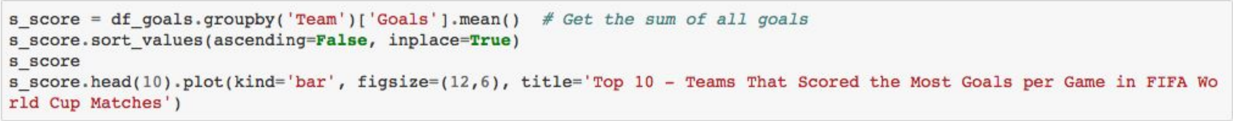
2c. Which Teams Achieved The Most Goals, Overall and per Match?



```
column_update = ['Team', 'Goals']
df_home_goals = world_cup_matches[['Home Team Name', 'Home Team Goals']]# Selecting all home team goals
df_home_goals.columns = column_update
df_away_goals = world_cup_matches[['Away Team Name', 'Away Team Goals']]# Selecting all away team goals
df_away_goals.columns = column_update

# Combine the the goals of each country - including both as home and away team
df_goals = pd.concat([df_home_goals,df_away_goals], ignore_index=True)
```



```
s_goals = df_goals.groupby('Team')['Goals'].sum().sort_values(ascending=False) # Grouping and sorting the values

# Select the top 20 countries in terms of goals and sorting them
s_percentage = s_goals/s_goals.sum()
s_percentage.sort_values(ascending=False, inplace=True)
s_percentage.head(20).plot(kind='pie', figsize=(10,10), autopct='%.1f%%',
                           startangle=173, title='Top 20 - Teams Which Scored the Most Total Goals in FIFA World Cup M
atches', label='')
```

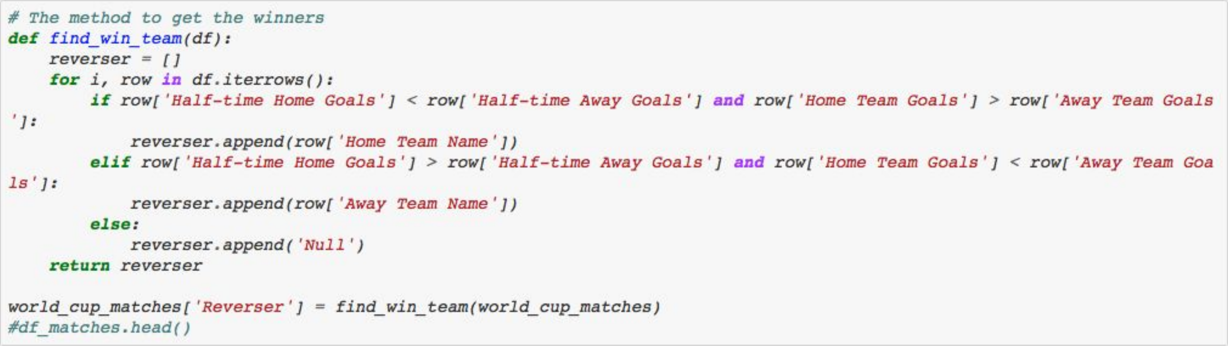


```
s_score = df_goals.groupby('Team')['Goals'].mean()  # Get the sum of all goals
s_score.sort_values(ascending=False, inplace=True)
s_score
s_score.head(10).plot(kind='bar', figsize=(12,6), title='Top 10 - Teams That Scored the Most Goals per Game in FIFA Wo
rld Cup Matches')
```

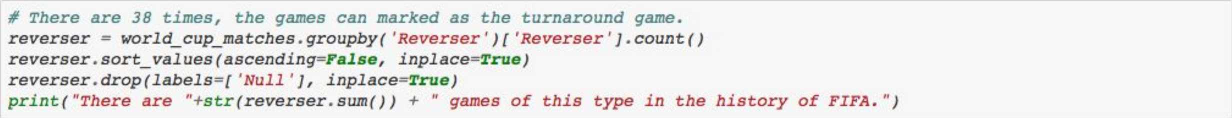
B. Analysis

1. Teams That Can Turn Around A Game

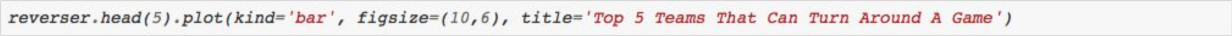
Which teams seem to be losing at half-time, but pull it together in the second half to become the winner of that match?



```
# The method to get the winners
def find_win_team(df):
    reverser = []
    for i, row in df.iterrows():
        if row['Half-time Home Goals'] < row['Half-time Away Goals'] and row['Home Team Goals'] > row['Away Team Goals
']:
            reverser.append(row['Home Team Name'])
        elif row['Half-time Home Goals'] > row['Half-time Away Goals'] and row['Home Team Goals'] < row['Away Team Goa
ls']:
            reverser.append(row['Away Team Name'])
        else:
            reverser.append('Null')
    return reverser

world_cup_matches['Reverser'] = find_win_team(world_cup_matches)
#df_matches.head()
```



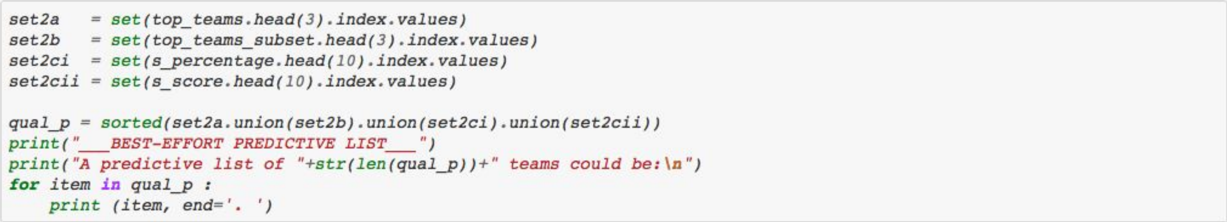
```
# There are 38 times, the games can marked as the turnaround game.
reverser = world_cup_matches.groupby('Reverser')['Reverser'].count()
reverser.sort_values(ascending=False, inplace=True)
reverser.drop(labels=['Null'], inplace=True)
print("There are "+str(reverser.sum()) + " games of this type in the history of FIFA.")
```



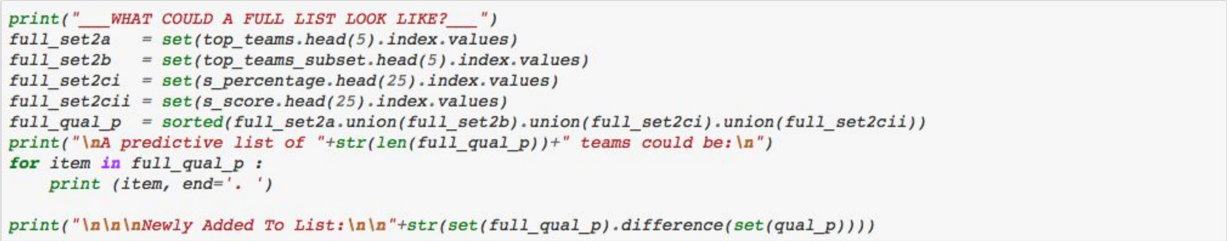
```
reverser.head(5).plot(kind='bar', figsize=(10,6), title='Top 5 Teams That Can Turn Around A Game')
```

C. Predict Qualifying Teams

The results of 2a, 2b, and 2c can be used to predict the what teams can qualify for the 2018 FIFA World Cup.



```
set2a   = set(top_teams.head(3).index.values)
set2b   = set(top_teams_subset.head(3).index.values)
set2ci  = set(s_percentage.head(10).index.values)
set2cii = set(s_score.head(10).index.values)

qual_p = sorted(set2a.union(set2b).union(set2ci).union(set2cii))
print("___BEST-EFFORT PREDICTIVE LIST___")
print("A predictive list of "+str(len(qual_p))+" teams could be:\n")
for item in qual_p :
    print (item, end='. ')
```



```
print("___WHAT COULD A FULL LIST LOOK LIKE?___")
full_set2a   = set(top_teams.head(5).index.values)
full_set2b   = set(top_teams_subset.head(5).index.values)
full_set2ci  = set(s_percentage.head(25).index.values)
full_set2cii = set(s_score.head(25).index.values)
full_qual_p  = sorted(full_set2a.union(full_set2b).union(full_set2ci).union(full_set2cii))
print("\nA predictive list of "+str(len(full_qual_p))+" teams could be:\n")
for item in full_qual_p :
    print (item, end='. ')

print("\n\n\nNewly Added To List:\n\n"+str(set(full_qual_p).difference(set(qual_p))))
```

END OF MATCH ANALYSIS

Analysis: Predicting the Winner of World Cup

Description

A. Statistics: 2018 Participants Quick Facts

1. Qualifying Teams Prediction Performance
2. Teams Participating in the 2018 FIFA World Cup That are First-time Participants
3. Teams Participating in the 2018 FIFA World Cup That Have Never Won A Match
4. Teams Participating in the 2018 FIFA World Cup That Are Not First-time Participants, and Have Never Won a Match
5. Top 20 Match Winners Amongst the Teams Participating in the 2018 FIFA World Cup

B. Prediction: Predicting the Outcome of Matches

1. Set Up the Framework for Prediction Analysis
   a. Calculate the ratio of the goals scored (GS) vs goals against (GA)
   b. Use Poisson distribution to simulate the result of a match

2. Predict Qualification Teams
   a. Set Up Framework
   b. Calculate Qualification Rate

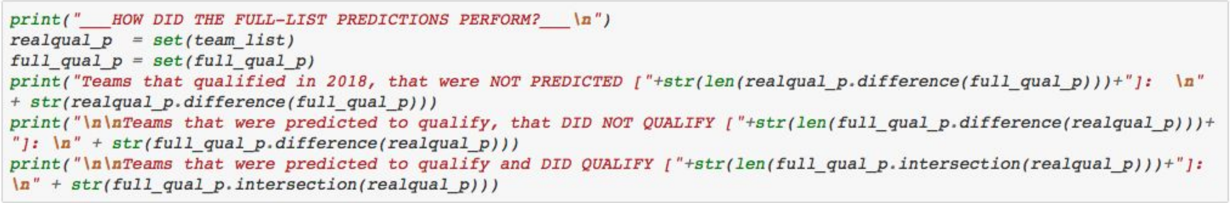
3. Determine Outcome of the matches at each elimination stage
   a. Qualification Rate for Group A
   b. Qualification Rate for Group B - Group H
   c. Predict Quarter, Semis, Finals, and Cup Winner

A. Statistics: 2018 Participants Quick Facts

For the 2018 FIFA Cup, there are 32 teams

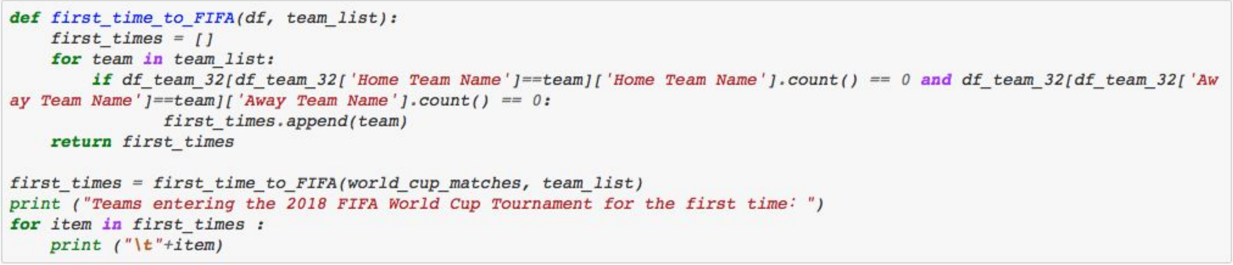
Group A : Russia, Egypt, Saudi Arabia,Uruguay
Group B: Morocco, Spain, Portugal, Iran
Group C: France, Australia, Peru, Denmark
Group D: Argentina, Iceland, Croatia, Nigeria
Group E: Brazil, Costa Rica, Serbia, Switzerland
Group F: Germany, Mexico, Sweden, Korea Republic
Group G: Belgium, Panama, Tunisia, England
Group H: Poland, Senegal, Colombia, Japan

1. Qualifying Teams Prediction Performance



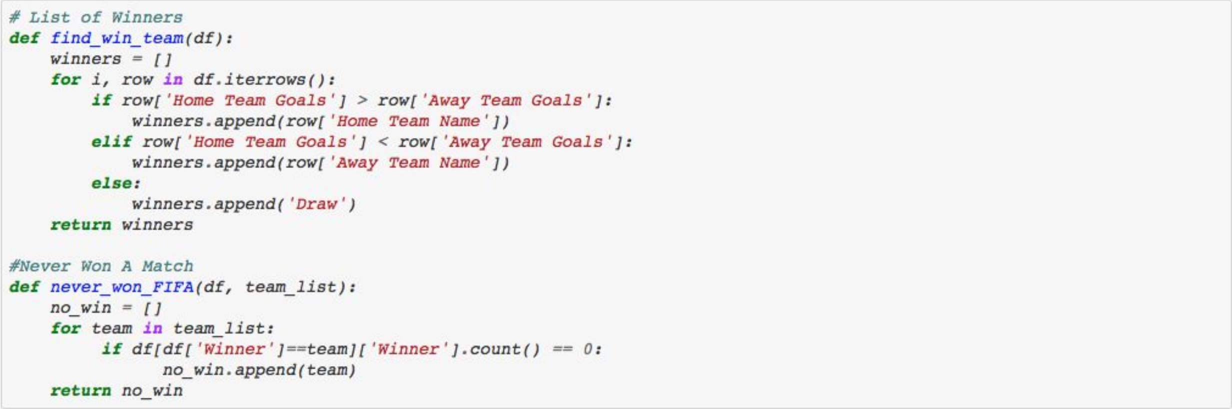
```
print("___HOW DID THE FULL-LIST PREDICTIONS PERFORM?___\n")
realqual_p  = set(team_list)
full_qual_p = set(full_qual_p)
print("Teams that qualified in 2018, that were NOT PREDICTED ["+str(len(realqual_p.difference(full_qual_p)))+"]:  \n"
+ str(realqual_p.difference(full_qual_p)))
print("\n\nTeams that were predicted to qualify, that DID NOT QUALIFY ["+str(len(full_qual_p.difference(realqual_p)))+
"]: \n" + str(full_qual_p.difference(realqual_p)))
print("\n\nTeams that were predicted to qualify and DID QUALIFY ["+str(len(full_qual_p.intersection(realqual_p)))+"]:
\n" + str(full_qual_p.intersection(realqual_p)))
```

2. Teams Participating in the 2018 FIFA World Cup That are First-time Participants

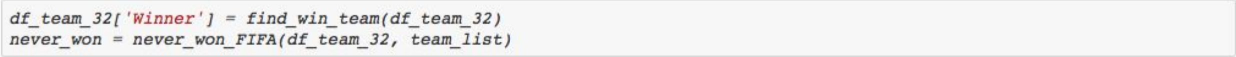


```
def first_time_to_FIFA(df, team_list):
    first_times = []
    for team in team_list:
        if df_team_32[df_team_32['Home Team Name']==team]['Home Team Name'].count() == 0 and df_team_32[df_team_32['Aw
ay Team Name']==team]['Away Team Name'].count() == 0:
            first_times.append(team)
    return first_times

first_times = first_time_to_FIFA(world_cup_matches, team_list)
print ("Teams entering the 2018 FIFA World Cup Tournament for the first time: ")
for item in first_times :
    print ("\t"+item)
```

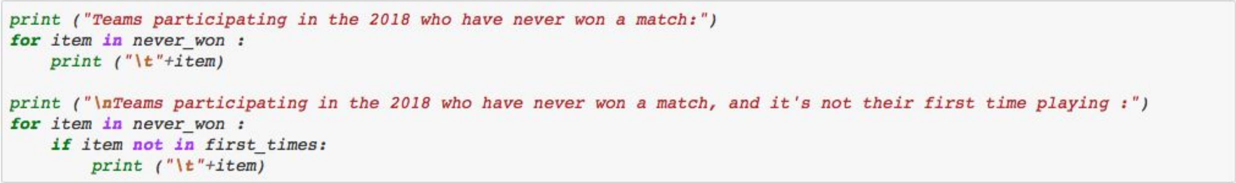
3/4. Teams Participating in the 2018 FIFA World Cup That Have Never Won A Match & First Time Playing



```
# List of Winners
def find_win_team(df):
    winners = []
    for i, row in df.iterrows():
        if row['Home Team Goals'] > row['Away Team Goals']:
            winners.append(row['Home Team Name'])
        elif row['Home Team Goals'] < row['Away Team Goals']:
            winners.append(row['Away Team Name'])
        else:
            winners.append('Draw')
    return winners

#Never Won A Match
def never_won_FIFA(df, team_list):
    no_win = []
    for team in team_list:
         if df[df['Winner']==team]['Winner'].count() == 0:
               no_win.append(team)
    return no_win
```

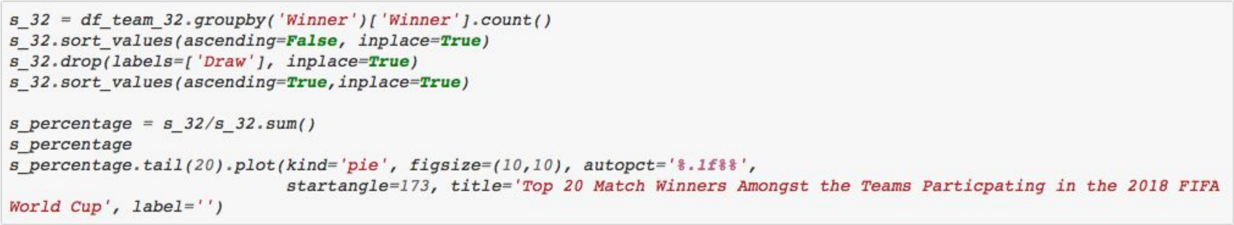


```
df_team_32['Winner'] = find_win_team(df_team_32)
never_won = never_won_FIFA(df_team_32, team_list)
```



```
print ("Teams participating in the 2018 who have never won a match:")
for item in never_won :
    print ("\t"+item)

print ("\nTeams participating in the 2018 who have never won a match, and it's not their first time playing :")
for item in never_won :
    if item not in first_times:
        print ("\t"+item)
```

5. Top 20 Match Winners Amongst the Teams Participating in the 2018 FIFA World Cup

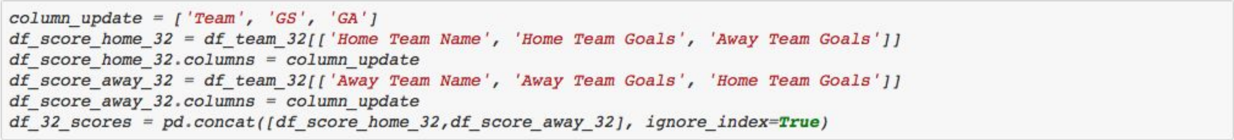


```
s_32 = df_team_32.groupby('Winner')['Winner'].count()
s_32.sort_values(ascending=False, inplace=True)
s_32.drop(labels=['Draw'], inplace=True)
s_32.sort_values(ascending=True,inplace=True)

s_percentage = s_32/s_32.sum()
s_percentage
s_percentage.tail(20).plot(kind='pie', figsize=(10,10), autopct='%.1f%%',
                           startangle=173, title='Top 20 Match Winners Amongst the Teams Particpating in the 2018 FIFA
World Cup', label='')
```

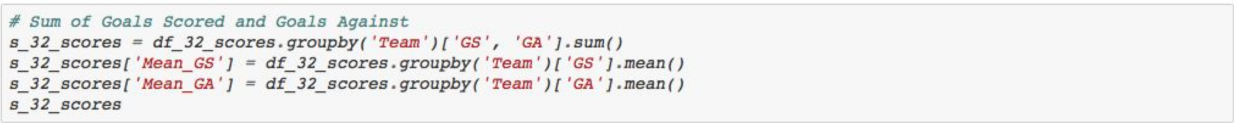
B. Prediction: Predicting the Outcomes of Matches

1. Set Up the Framework for Prediction Analysis

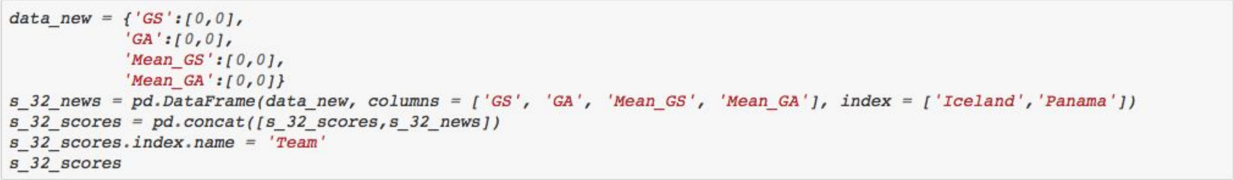
1a. Calculate the ratio of the goals scored (GS) vs goals against (GA)



```
column_update = ['Team', 'GS', 'GA']
df_score_home_32 = df_team_32[['Home Team Name', 'Home Team Goals', 'Away Team Goals']]
df_score_home_32.columns = column_update
df_score_away_32 = df_team_32[['Away Team Name', 'Away Team Goals', 'Home Team Goals']]
df_score_away_32.columns = column_update
df_32_scores = pd.concat([df_score_home_32,df_score_away_32], ignore_index=True)
```

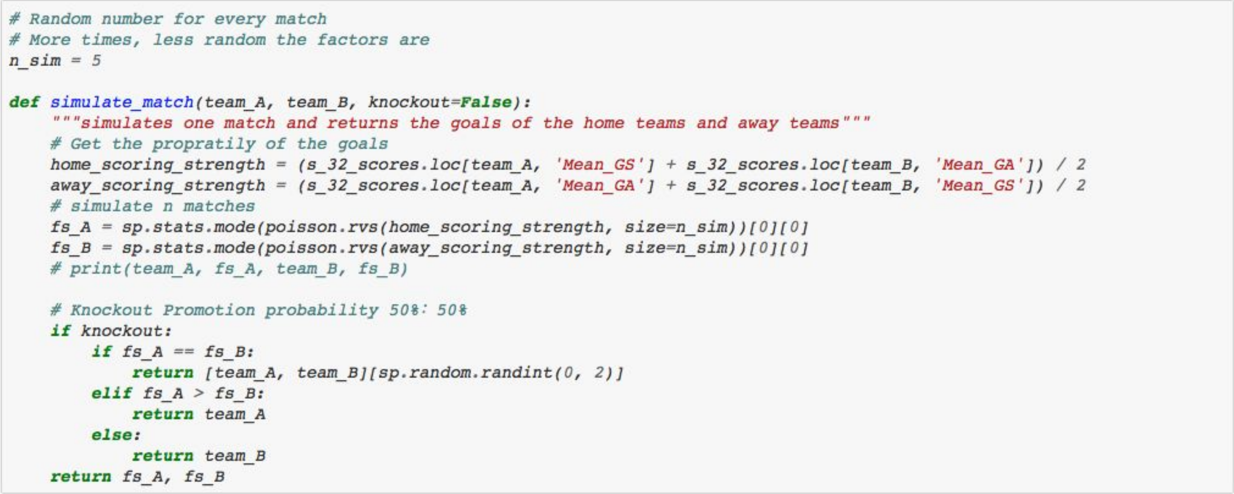


```
# Sum of Goals Scored and Goals Against
s_32_scores = df_32_scores.groupby('Team')['GS', 'GA'].sum()
s_32_scores['Mean_GS'] = df_32_scores.groupby('Team')['GS'].mean()
s_32_scores['Mean_GA'] = df_32_scores.groupby('Team')['GA'].mean()
s_32_scores
```



```
data_new = {'GS':[0,0],
            'GA':[0,0],
            'Mean_GS':[0,0],
            'Mean_GA':[0,0]}
s_32_news = pd.DataFrame(data_new, columns = ['GS', 'GA', 'Mean_GS', 'Mean_GA'], index = ['Iceland','Panama'])
s_32_scores = pd.concat([s_32_scores,s_32_news])
s_32_scores.index.name = 'Team'
s_32_scores
```

1b. Use Poisson distribution to simulate the result of a match

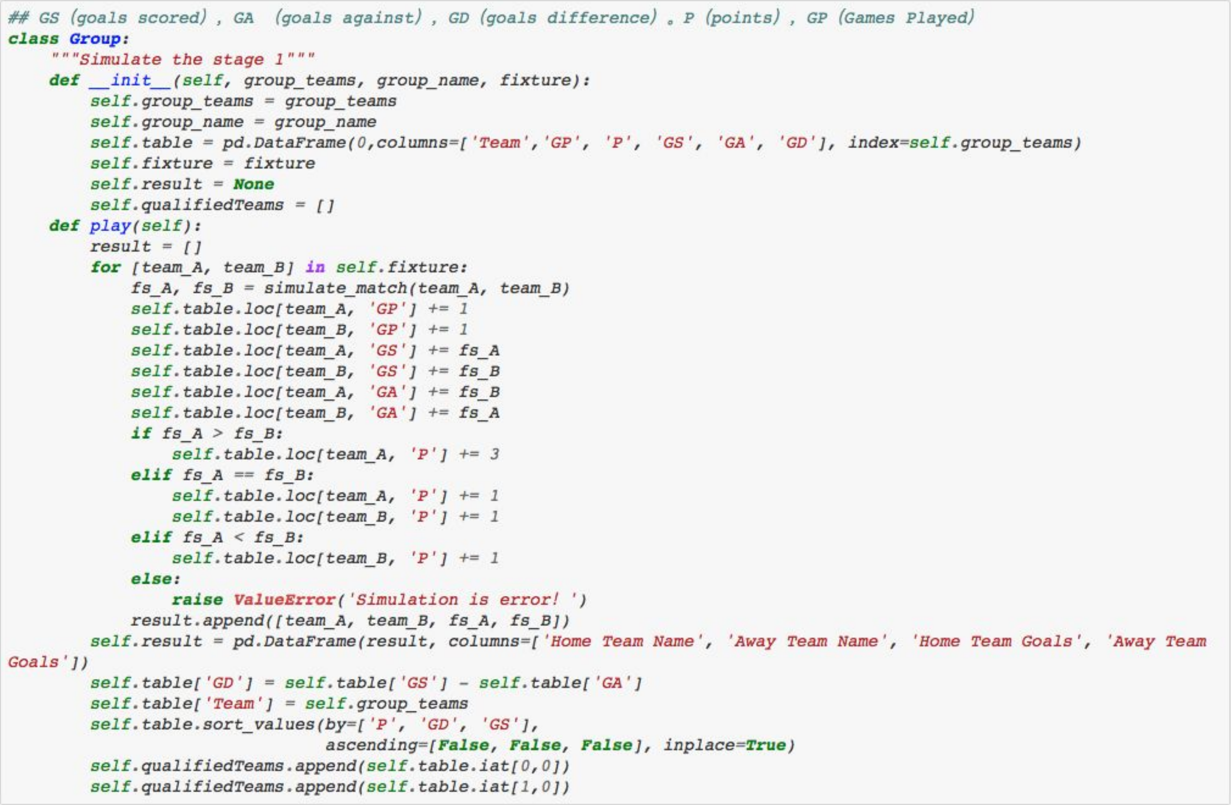


```
# Random number for every match
# More times, less random the factors are
n_sim = 5

def simulate_match(team_A, team_B, knockout=False):
    """simulates one match and returns the goals of the home teams and away teams"""
    # Get the propratily of the goals
    home_scoring_strength = (s_32_scores.loc[team_A, 'Mean_GS'] + s_32_scores.loc[team_B, 'Mean_GA']) / 2
    away_scoring_strength = (s_32_scores.loc[team_A, 'Mean_GA'] + s_32_scores.loc[team_B, 'Mean_GS']) / 2
    # simulate n matches
    fs_A = sp.stats.mode(poisson.rvs(home_scoring_strength, size=n_sim))[0][0]
    fs_B = sp.stats.mode(poisson.rvs(away_scoring_strength, size=n_sim))[0][0]
    # print(team_A, fs_A, team_B, fs_B)

    # Knockout Promotion probability 50%： 50%
    if knockout:
        if fs_A == fs_B:
            return [team_A, team_B][sp.random.randint(0, 2)]
        elif fs_A > fs_B:
            return team_A
        else:
            return team_B
    return fs_A, fs_B
```

...code omitted….

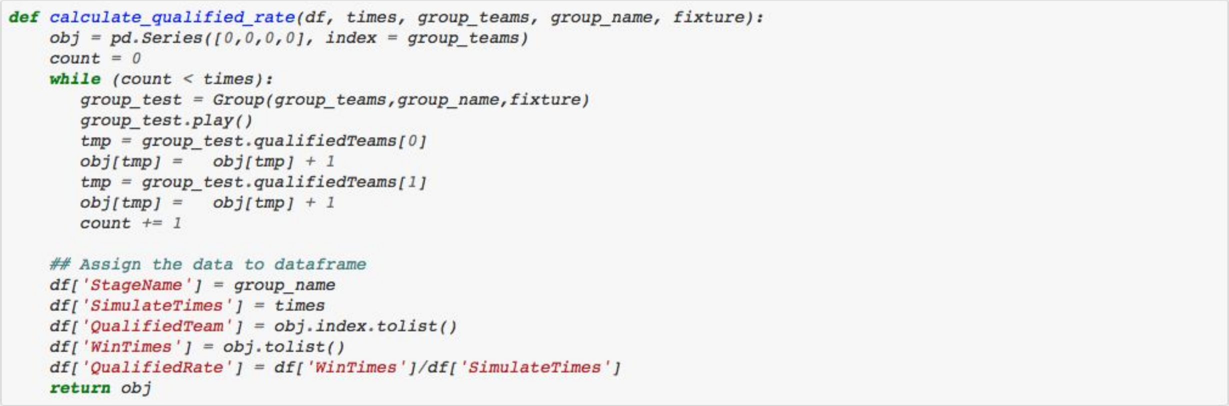
2. Simulate X times, at the Group Stage to Predict Qualified Teams

2a. Set Up Framework



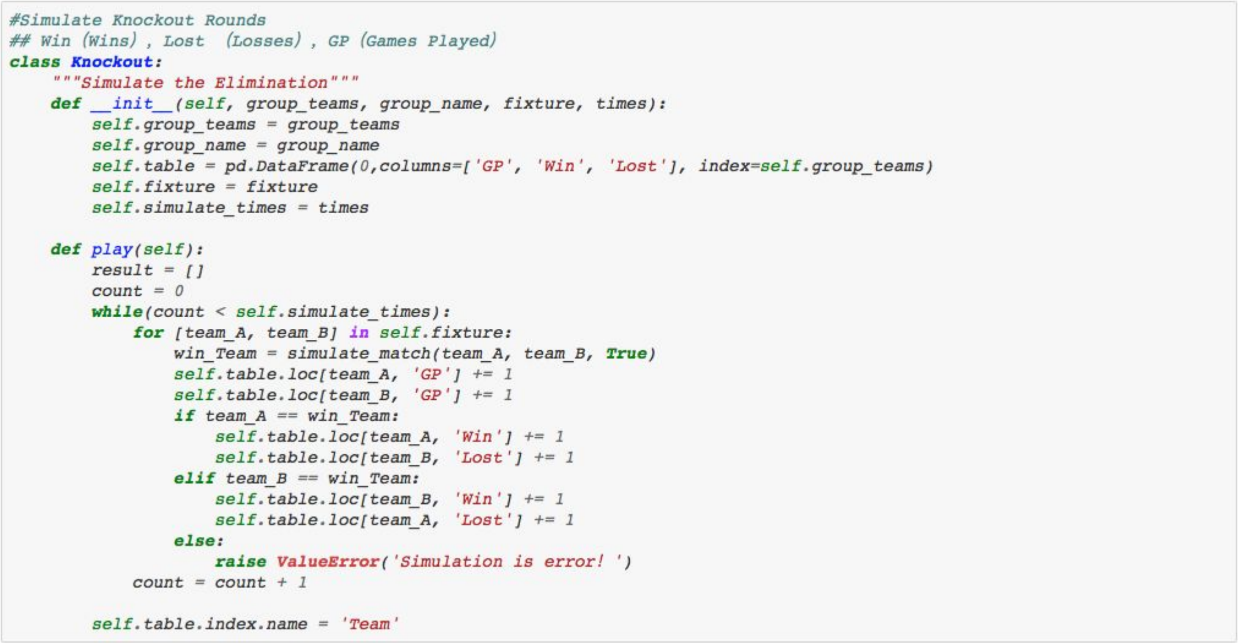
```
## GS (goals scored) , GA  (goals against) , GD (goals difference) 。P (points) , GP (Games Played)
class Group:
    """Simulate the stage 1"""
    def __init__(self, group_teams, group_name, fixture):
        self.group_teams = group_teams
        self.group_name = group_name
        self.table = pd.DataFrame(0,columns=['Team','GP', 'P', 'GS', 'GA', 'GD'], index=self.group_teams)
        self.fixture = fixture
        self.result = None
        self.qualifiedTeams = []
    def play(self):
        result = []
        for [team_A, team_B] in self.fixture:
            fs_A, fs_B = simulate_match(team_A, team_B)
            self.table.loc[team_A, 'GP'] += 1
            self.table.loc[team_B, 'GP'] += 1
            self.table.loc[team_A, 'GS'] += fs_A
            self.table.loc[team_B, 'GS'] += fs_B
            self.table.loc[team_A, 'GA'] += fs_B
            self.table.loc[team_B, 'GA'] += fs_A
            if fs_A > fs_B:
                self.table.loc[team_A, 'P'] += 3
            elif fs_A == fs_B:
                self.table.loc[team_A, 'P'] += 1
                self.table.loc[team_B, 'P'] += 1
            elif fs_A < fs_B:
                self.table.loc[team_B, 'P'] += 1
            else:
                raise ValueError('Simulation is error! ')
            result.append([team_A, team_B, fs_A, fs_B])
        self.result = pd.DataFrame(result, columns=['Home Team Name', 'Away Team Name', 'Home Team Goals', 'Away Team
Goals'])
        self.table['GD'] = self.table['GS'] - self.table['GA']
        self.table['Team'] = self.group_teams
        self.table.sort_values(by=['P', 'GD', 'GS'],
                               ascending=[False, False, False], inplace=True)
        self.qualifiedTeams.append(self.table.iat[0,0])
        self.qualifiedTeams.append(self.table.iat[1,0])
```

2b. Calculate Qualification Rate



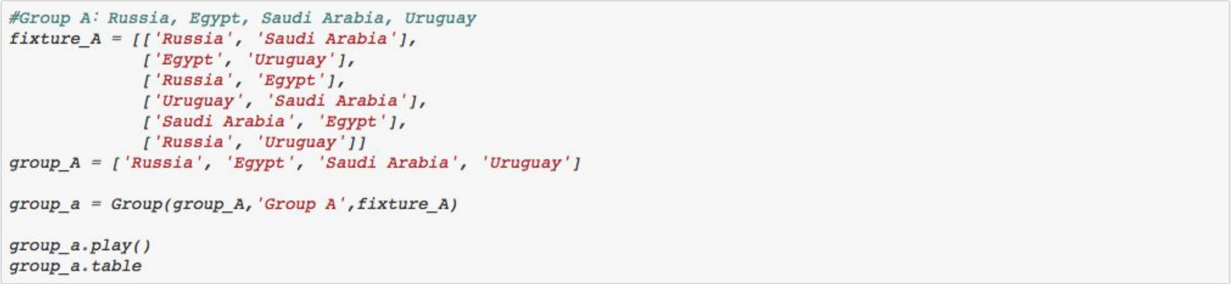
```
def calculate_qualified_rate(df, times, group_teams, group_name, fixture):
    obj = pd.Series([0,0,0,0], index = group_teams)
    count = 0
    while (count < times):
       group_test = Group(group_teams,group_name,fixture)
       group_test.play()
       tmp = group_test.qualifiedTeams[0]
       obj[tmp] =   obj[tmp] + 1
       tmp = group_test.qualifiedTeams[1]
       obj[tmp] =   obj[tmp] + 1
       count += 1

    ## Assign the data to dataframe
    df['StageName'] = group_name
    df['SimulateTimes'] = times
    df['QualifiedTeam'] = obj.index.tolist()
    df['WinTimes'] = obj.tolist()
    df['QualifiedRate'] = df['WinTimes']/df['SimulateTimes']
    return obj
```

...code omitted…

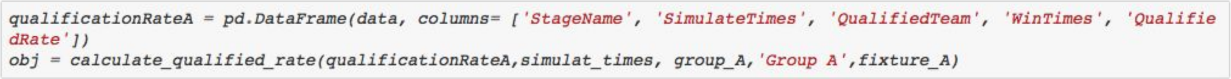


```
#Simulate Knockout Rounds
## Win (Wins) , Lost  (Losses) , GP (Games Played)
class Knockout:
    """Simulate the Elimination"""
    def __init__(self, group_teams, group_name, fixture, times):
        self.group_teams = group_teams
        self.group_name = group_name
        self.table = pd.DataFrame(0,columns=['GP', 'Win', 'Lost'], index=self.group_teams)
        self.fixture = fixture
        self.simulate_times = times

    def play(self):
        result = []
        count = 0
        while(count < self.simulate_times):
            for [team_A, team_B] in self.fixture:
                win_Team = simulate_match(team_A, team_B, True)
                self.table.loc[team_A, 'GP'] += 1
                self.table.loc[team_B, 'GP'] += 1
                if team_A == win_Team:
                    self.table.loc[team_A, 'Win'] += 1
                    self.table.loc[team_B, 'Lost'] += 1
                elif team_B == win_Team:
                    self.table.loc[team_B, 'Win'] += 1
                    self.table.loc[team_A, 'Lost'] += 1
                else:
                    raise ValueError('Simulation is error! ')
            count = count + 1

        self.table.index.name = 'Team'
```

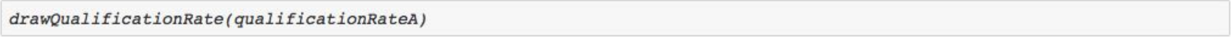
3a. Qualification Rate for Group A



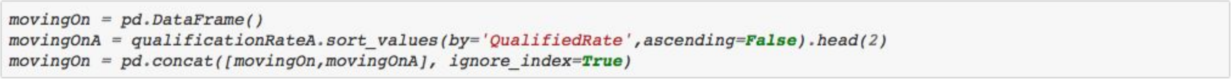
```
#Group A: Russia, Egypt, Saudi Arabia, Uruguay
fixture_A = [['Russia', 'Saudi Arabia'],
             ['Egypt', 'Uruguay'],
             ['Russia', 'Egypt'],
             ['Uruguay', 'Saudi Arabia'],
             ['Saudi Arabia', 'Egypt'],
             ['Russia', 'Uruguay']]
group_A = ['Russia', 'Egypt', 'Saudi Arabia', 'Uruguay']

group_a = Group(group_A,'Group A',fixture_A)

group_a.play()
group_a.table
```



```
qualificationRateA = pd.DataFrame(data, columns= ['StageName', 'SimulateTimes', 'QualifiedTeam', 'WinTimes', 'Qualifie
dRate'])
obj = calculate_qualified_rate(qualificationRateA,simulat_times, group_A,'Group A',fixture_A)
```

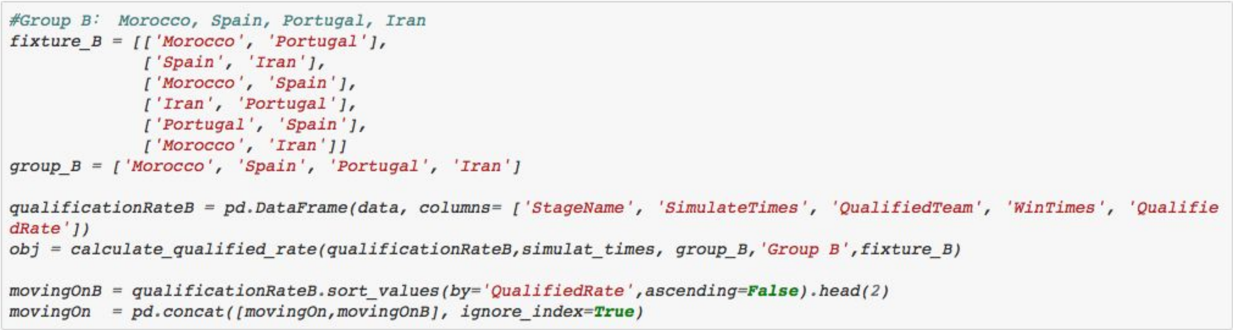


```
drawQualificationRate(qualificationRateA)
```



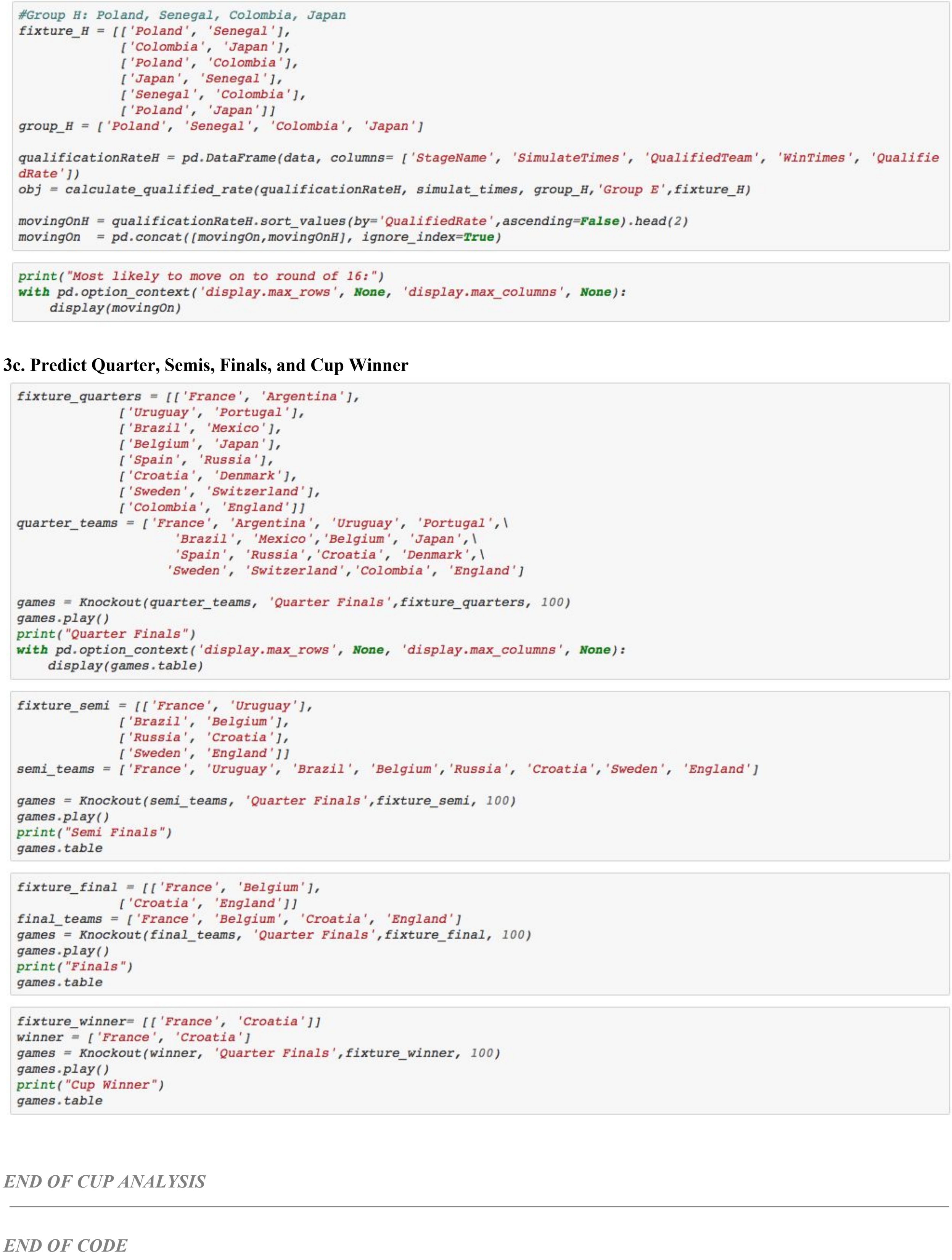
```
movingOn = pd.DataFrame()
movingOnA = qualificationRateA.sort_values(by='QualifiedRate',ascending=False).head(2)
movingOn = pd.concat([movingOn,movingOnA], ignore_index=True)
```

3b. Qualification Rate for Group B -- Group H



```
#Group B:  Morocco, Spain, Portugal, Iran
fixture_B = [['Morocco', 'Portugal'],
             ['Spain', 'Iran'],
             ['Morocco', 'Spain'],
             ['Iran', 'Portugal'],
             ['Portugal', 'Spain'],
             ['Morocco', 'Iran']]
group_B = ['Morocco', 'Spain', 'Portugal', 'Iran']

qualificationRateB = pd.DataFrame(data, columns= ['StageName', 'SimulateTimes', 'QualifiedTeam', 'WinTimes', 'Qualifie
dRate'])
obj = calculate_qualified_rate(qualificationRateB,simulat_times, group_B,'Group B',fixture_B)

movingOnB = qualificationRateB.sort_values(by='QualifiedRate',ascending=False).head(2)
movingOn  = pd.concat([movingOn,movingOnB], ignore_index=True)
```

...code omitted…



```
#Group H: Poland, Senegal, Colombia, Japan
fixture_H = [['Poland', 'Senegal'],
             ['Colombia', 'Japan'],
             ['Poland', 'Colombia'],
             ['Japan', 'Senegal'],
             ['Senegal', 'Colombia'],
             ['Poland', 'Japan']]
group_H = ['Poland', 'Senegal', 'Colombia', 'Japan']

qualificationRateH = pd.DataFrame(data, columns= ['StageName', 'SimulateTimes', 'QualifiedTeam', 'WinTimes', 'QualifiedRate'])
obj = calculate_qualified_rate(qualificationRateH, simulat_times, group_H,'Group E',fixture_H)

movingOnH = qualificationRateH.sort_values(by='QualifiedRate',ascending=False).head(2)
movingOn  = pd.concat([movingOn,movingOnH], ignore_index=True)
```

```
print("Most likely to move on to round of 16:")
with pd.option_context('display.max_rows', None, 'display.max_columns', None):
    display(movingOn)
```

3c. Predict Quarter, Semis, Finals, and Cup Winner

```
fixture_quarters = [['France', 'Argentina'],
             ['Uruguay', 'Portugal'],
             ['Brazil', 'Mexico'],
             ['Belgium', 'Japan'],
             ['Spain', 'Russia'],
             ['Croatia', 'Denmark'],
             ['Sweden', 'Switzerland'],
             ['Colombia', 'England']]
quarter_teams = ['France', 'Argentina', 'Uruguay', 'Portugal',\
                  'Brazil', 'Mexico','Belgium', 'Japan',\
                  'Spain', 'Russia','Croatia', 'Denmark',\
                 'Sweden', 'Switzerland','Colombia', 'England']

games = Knockout(quarter_teams, 'Quarter Finals',fixture_quarters, 100)
games.play()
print("Quarter Finals")
with pd.option_context('display.max_rows', None, 'display.max_columns', None):
    display(games.table)
```

```
fixture_semi = [['France', 'Uruguay'],
             ['Brazil', 'Belgium'],
             ['Russia', 'Croatia'],
             ['Sweden', 'England']]
semi_teams = ['France', 'Uruguay', 'Brazil', 'Belgium','Russia', 'Croatia','Sweden', 'England']

games = Knockout(semi_teams, 'Quarter Finals',fixture_semi, 100)
games.play()
print("Semi Finals")
games.table
```

```
fixture_final = [['France', 'Belgium'],
             ['Croatia', 'England']]
final_teams = ['France', 'Belgium', 'Croatia', 'England']
games = Knockout(final_teams, 'Quarter Finals',fixture_final, 100)
games.play()
print("Finals")
games.table
```

```
fixture_winner= [['France', 'Croatia']]
winner = ['France', 'Croatia']
games = Knockout(winner, 'Quarter Finals',fixture_winner, 100)
games.play()
print("Cup Winner")
games.table
```

END OF CUP ANALYSIS

END OF CODE