UNIVERSITY OF ELECTRONIC SCIENCE AND TECHNOLOGY OF CHINA
SCHOOL OF COMPUTER SCIENCE TECHNOLOGY

Final Report: World Cup Prediction with Logistic Regression

NUMERICAL ANALYSIS

SUBMITTED TO

PROF. ZHAO XILE

BY

YEBOAH-DUAKO ELVIS

yeboahduako770@gmail.com

(ID: 202124080120)

JUNE, 2022

ABSTRACT

This is a detailed report on how logistic regression, a numerical analysis and machine learning technique, was used to predict the winners of simulated world cup matches and, ultimately, the tournament champion. I generated a logit model that analyzes key features of paired teams to estimate the chance of a team winning against a particular opponent by examining three separate datasets: a monthly FIFA Ranking dataset, a history of past matches between the national teams and the current world cup dataset. With an AUC score of 0.75, the trained model exhibited a 68 percent accuracy which is excellent for the goal of this project.

Also, I provided the mathematical background that supports my trained logit model for your review.

TABLE OF CONTENTS

Introduction

The FIFA World Cup is an international association football tournament played by senior men's national teams from members of FIFA, the sport's global governing organization. Since the first tournament in 1930, the championship has been awarded every four years, with the exception of 1942 and 1946, when it was not held due to World War II. France is the current champion, having won the 2018 event in Russia for the second time. The current format includes a qualification round that lasts three years and determines which teams will compete in the tournament phase. During the tournament phase, 32 teams, including the automatically qualifying host nation(s), fight for the championship over the course of around a month at venues throughout the host nation(s). [1]

Twenty-one final tournaments have been organized as of the 2018 FIFA World Cup, with a total of 79 national teams competing. Eight national teams have won the award. Brazil has won five tournaments and is the only team to have competed in all of them. Germany and Italy have four World Cup titles each; Argentina, France, and the first winner Uruguay have two titles each; and England and Spain have one title each. [2]

Fig. 1: France, the defending champions, with their trophy from the 2018 world cup

Goal of the Project

The ultimate goal of this project is to investigate logistic regression and data analysis techniques in order to create a predictive model that can be used to simulate and forecast the winning team at the upcoming FIFA World Cup in Qatar.

Mathematical Background

The underlying branch of mathematics used for this project is Numerical Analysis, which deals with the design, analysis and implementation of numerical methods that yield exact or approximate solutions to mathematical problems. Of the myriad techniques numerical analysis provide, Regression is the ideal one I chose to work with. Regression in general helps to establish relationships between dependent variables (outcome/response variable or label) and one or more independent variables (predictors/ features/ covariates). And that is exactly what is needed for every good prediction model: A strong relationship or correlation between interested variables. And further under regression, I used Logistic Regression which is basically a classification technique. By definition, Logistic Regression is a regression model used to model the Probability of one event (out of two alternatives) taking place [3]. It is basically a classification algorithm used to predict a binary outcome given a set if independent variables.

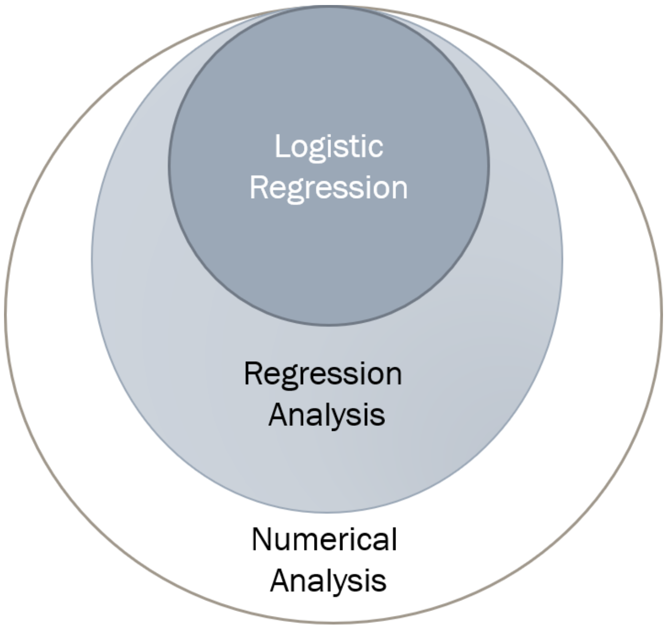


Fig. 2: Mathematical Background of the Project

Logistic Regression

As stated earlier, Logistic Regression is mainly used in determining the probability of the occurrence of a categorical dependent variable based on other correlated independent variables. [4] To illustrate this concept well, let's consider the simple data in table 1, where certain students passed or failed an exam depending on the number of hours they studied. With logistic regression, our aim is to generate a model or a sigmoid function that provides a mathematical relationship between hours of studies and passing an exam so as to use the number of hours the student studies as an input to output the probability of that student passing. Based on the value of the probability and a set threshold, we can confidently tell or predict that the student who studied for t hours will either pass or fail the exams. In figure 3, the generated sigmoid graph can be used to predict that a student who studied for 2 hours will pass the exam a probability of 0.25 and another student who studied for 4 hours is predicted to pass with a probability of 0.91.

Table 1: Simple data on hours studied and pass results

Hours	0.50	1.00	1.50	1.75	2.50	3.50	4.00	4.75	5.00	5.50
Pass	0	0	0	1	0	0	1	1	1	1

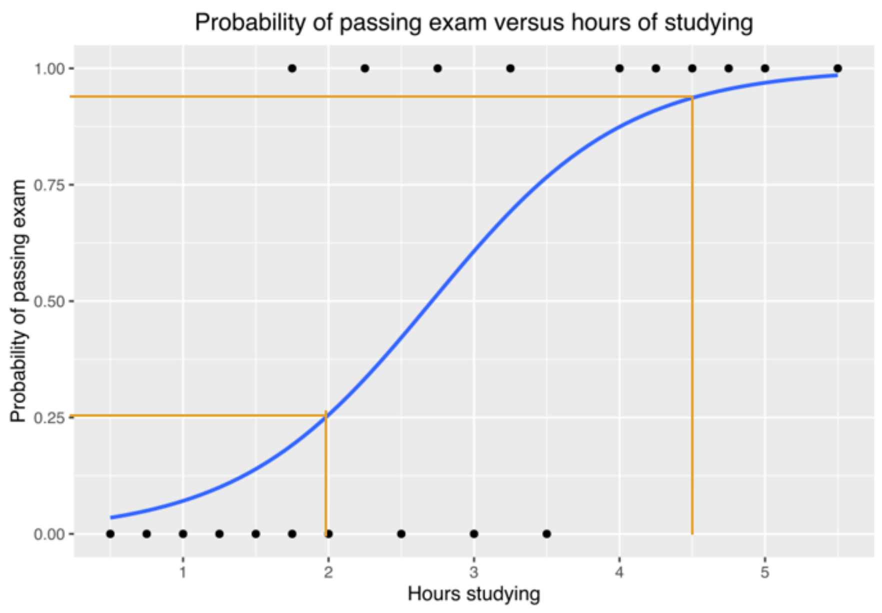


Fig3.: Sigmoid function used to predict the probability of passing for a given number of hours

Mathematically, we present the dependent variable as a linear combination of a constant term and the independent variables using the linear predictor function as shown below;

$$y(x) = \beta_o + \beta_1(x_1) + \beta_2(x_2) + \cdots + \beta_i(x_i) + \varepsilon_i \qquad (1)$$

Where:

$$y(x) = independent\ variable$$

$$x_i = dependent\ variables\ (predictors)$$

$$\beta_o = constant\ term\ (y - intercept)$$

$$\beta_i = rate\ parameters\ of\ x_i$$

$$\varepsilon_i = error\ term$$

Equation (1) represents the general linear regression model, and unlike the logistic regression we will be deriving shortly, is used to predict continuous values for the dependent variable. The output of a Logistic regression model, however, must be categorical. That is, its range must respect the boundaries of probabilistic values ([0,1]). As such, equation (1) needs to be modified into a more befitting model or function.

Derivation of Logit Model Equations

From equation (1), since our interest in Logistic regression is to predict probabilities, we substitute $p(x)$ for $y(x)$ on the assumption that, the probability is also a linear combination of the independent predictor variables. [4]

$$p(x) = \beta_o + \beta_1(x_1) + \beta_2(x_2) + \cdots + \beta_i(x_i) + \varepsilon_i \qquad (2)$$

As stated earlier, the range of values that $p(x)$ can take is $[0,1]$ with a midpoint value of 0.5. AS we know, values of x are independent and can take any value in the real number space ($x \in \mathbb{R}$) and hence, the linear combination of these values of x can cause the value of $p(x)$ to go out of its defined limits.

To solve this phenomenon, we resort to the concept of odds which is simply a ratio of the probability of one event occurring to the probability of the event not occurring.

$$odds\ (\theta(x)) = \frac{p(x)}{1 - p(x)} \qquad (3)$$

The goal here is to find the odds as the linear combination of the independent variables and then after, using equation (3), we can find the probability in terms of the given odds value. Hence, we substitute $\theta(x)$ for $p(x)$ in equation (2).

$$\theta(x) = \beta_o + \beta_1(x_1) + \beta_2(x_2) + \cdots + \beta_i(x_i) + \varepsilon_i \qquad (4)$$

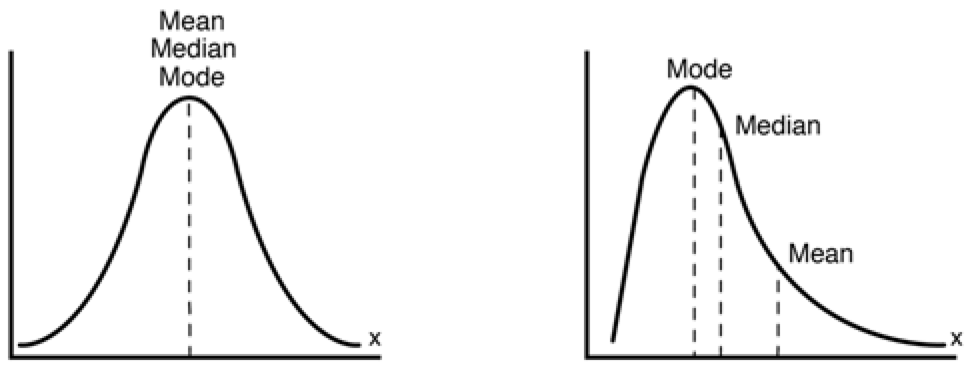
Which can further be written in terms of probability as:

$$\frac{p(x)}{1 - p(x)} = \beta_o + \beta_1(x_1) + \beta_2(x_2) + \cdots + \beta_i(x_i) + \varepsilon_i \qquad (4^*)$$

Now, let’s examine the range of odds to see if the aforementioned problem is solved. Focusing on the Left-Hand Side (LHS) of equation (4^*), the minimum value $\theta(x)$ can take is zero (0) that is when $p(x)$ is minimum(at $p(x) = 0$). And $\theta(x)$ approaches ∞ as $p(x)$ approaches 1. Hence, the range of values for odds is $[0, \infty]$ and a midpoint of 1 at $p(x) = 0.5$

While the problem of range seems to be solved, it is not entirely perfect yet for two reasons:

1. There is still a hard lower boundary limit of zero (0) on the range.
2. The distribution is positively skewed given that the midpoint is 1 within the $[0, \infty]$ range as shown below. Mostly, it is preferred to have a normal symmetric distribution.



(a) Normal (No Skew) (b) Positively Skewed

Fig. 4: Skewness of a distribution

Owing to the above reasons, a slight modification was necessary. So the concept of log-odds was introduced where we express the dependent variable as the natural log of the odds $(ln(\theta(x))$.

$$\ln\,(\theta(x)) = \beta_o + \beta_1(x_1) + \beta_2(x_2) + \cdots + \beta_i(x_i) + \varepsilon_i \qquad (5)$$

which is equivalent to:

$$\ln\left(\frac{p(x)}{1 - p(x)}\right) = \beta_o + \beta_1(x_1) + \beta_2(x_2) + \cdots + \beta_i(x_i) + \varepsilon_i \qquad (5^*)$$

By analyzing the log-odds above, we find the range to be $[-\infty, \infty]$ with a midpoint of 0 at $p(x) = 0.5$ thereby providing a feasible solution to both problems raised in the ordinary odds model equation: the lower hard limit of 0 is extended to $-\infty$ and the midpoint taken to 0 thereby producing a symmetric or normal distribution as shown in fig. 4(a).

Equation (5^*) represents the general Binomial Logistic Regression model equation. Remember however, that our interest was to determine a model for predicting the occurrence of an even based on the probability and not log-odds. Hence, we obtain our logistic or sigmoid function as follows:

Let ℓ represent the log-odds:

$$\ell = \ln\left(\frac{p(x)}{1-p(x)}\right) \qquad (6)$$

$$\Rightarrow \quad e^{\ell} = \frac{p(x)}{1-p(x)}$$

$$\Rightarrow \quad e^{\ell}\big(1-p(x)\big) = p(x)$$

$$\Rightarrow \quad e^{\ell} - e^{\ell}p(x) = p(x)$$

$$\Rightarrow \quad p(x) + e^{\ell}p(x) = e^{\ell}$$

$$\Rightarrow \quad p(x)\big[1+e^{\ell}\big] = e^{\ell}$$

$$\Rightarrow \quad p(x) = \frac{e^{\ell}}{1+e^{\ell}}$$

$$\therefore \quad p(x) = \frac{1}{1+e^{-\ell}} \qquad (7)$$

Equation (7) therefore is our **logit model equation**.

Now, all along, our focus has been on the Left-Hand Side of the original equation (that is equation (1)). let's take a look at the right-hand side of equation (1). How do we obtain the rate parameter values (β_i) and the constant term (β_0)?

First, the rate parameter values serve as the weights of the various individual independent variables. They tell the degree of change that is be expected for a unit increase in one independent variable when we keep all other independent variables constant. These values of the rate parameters are estimated from a given sample data or observations using either the *Iteratively Reweighted Least Squares (IRLS) method* or the *Maximum Likelihood Estimation (MLE) method.*

Maximum Likelihood Estimation (MLE)

The Maximum Likelihood Estimation is a frequentist probabilistic framework for estimating the parameters of a model (the logistic model in this case). It maximizes the conditional probability of observing the independent variable data (*x*) given a specific probability distribution and its parameters. Unlike linear regression, it is not possible to obtain a closed-form formula for the coefficient values that maximizes the likelihood function in logistic regression with normally distributed residuals, hence an iterative procedure, such as Newton's method, must be employed instead. This approach starts with a rough solution, adjusts it slightly to see if it can be greatly improved, and repeats the process until no further improvements can be made, at which time it is said to have converged.

The model may not attain convergence in some cases. Because the iterative process was unable to identify adequate solutions; non-convergence of a model shows that the coefficients are meaningless. A significant ratio of predictors to cases, multicollinearity, sparseness, or perfect separation can all cause the model to fail to converge [5]. The Maximum likelihood estimation function or formular is shown below.

$$L = \prod_{y=1} p(x) \prod_{y=0} (1 - p(x)) \qquad (8)$$

Iteratively reweighted least squares (IRLS)

Binary logistic regression can be calculated, for example, using iteratively reweighted least squares (IRLS), which is comparable to utilizing Newton's method to maximize the log-likelihood of a Bernoulli distributed process. If the problem is written in vector matrix form, with parameters, $w^T = [\beta_0, \beta_1, \beta_2, \dots]$, explanatory variables $x(i) = [1, x_1(i), x_2(i), \dots]^T$ and expected value of the Bernoulli distribution $\mu(i) = \frac{1}{1+e^{-w^T x(i)}}$, the parameters w can be found using the following iterative algorithm:

$$w_{(k+1)} = (X^T S_k X)^{-1} X^T (S_k X w_k + y - \mu_k) \qquad (9)$$

Where $S = diag\left(\mu(i)\big(1 - \mu(i)\big)\right)$ is a diagonal weighing matrix, $\mu = [\mu(1), \mu(2), \dots]$ is the vector of expected values, and

$$X = \begin{bmatrix} 1 & x_1(1) & x_2(1) & \cdots \\ 1 & x_1(2) & x_2(2) & \cdots \\ \vdots & \vdots & \vdots & \end{bmatrix}$$

Is the regressor matrix, and $y(i) = [y(1), y(2), \dots]^T$ is the vector of response variables [6].

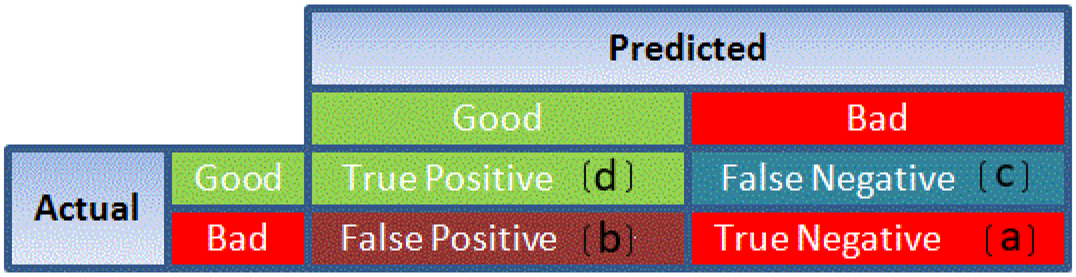
The Rule of ten

The "one in ten rule" argues that logistic regression models with a minimum of roughly 10 events per explanatory variable (EPV), where event represents occurrences belonging to the less frequent category in the dependent variable, yield consistent values for the explanatory variables. However, there is significant disagreement concerning the validity of this rule, which is based on simulation research and lacks a solid theoretical foundation. The rule, according to some authors, is overly conservative in some situations, with the authors stating, "If we consider confidence interval coverage less than 93 percent, type I error greater than 7%, or relative bias greater than 15% to be problematic, our findings show that problems are common with 2–4 EPV, uncommon with 5–9 EPV, and still present with 10–16 EPV. With 5–9 EPV, the worst cases of each condition were not as severe as those with 10–16 EPV, and were usually equivalent to those with 10–16 EPV " [7].

Performance Analysis of Logit Models

The following metrics are used to evaluate the performance of a logistic regression model.

1. Confusion Matrix: It's a tabular representation of actual vs predicted values. It helps to determine the accuracy of the model and to also avoid overfitting.



		Predicted	
		Good	Bad
Actual	Good	True Positive (d)	False Negative (c)
	Bad	False Positive (b)	True Negative (a)

Fig. 5: Confusion Matrix

$$Model\ Accuracy = \frac{a+d}{a+b+c+d} \qquad (9)$$

Where, as shown in fig. 5:

$$a = True\ negative$$
$$b = False\ Positive$$
$$c = False\ Negative$$
$$d = True\ Positive$$

Other useful metrics such as Sensitivity (the True Positive Rate (TRP)) and Specificity (the True Negative Rate (TNR)) which play a critical role in the receiver operating characteristic (ROC) curve can be generated from the Confusion matrix as follows:

$$Sensitivity\ (TPR) = \frac{d}{c+d} \qquad (10)$$

$$Specificity\ (TNR) = \frac{a}{a+b} \qquad (11)$$

$$False\ Positive\ rate\ (FPR)\ or\ \ (1 - Specificity) = \frac{b}{a+b} \qquad (12)$$

$$False\ Negative\ Rate\ (FNR) = \frac{c}{c+d} \qquad (13)$$

2. Receiver Operating Characteristic (ROC) Curve: summarizes the model's performance by evaluating the tradeoffs between true positive rate (sensitivity) and false positive rate(1-specificity). For plotting ROC, it is advisable to assume a threshold of $p > 0.5$ since we are more concerned about success rate. ROC summarizes the predictive power for all possible values of $p > 0.5$.

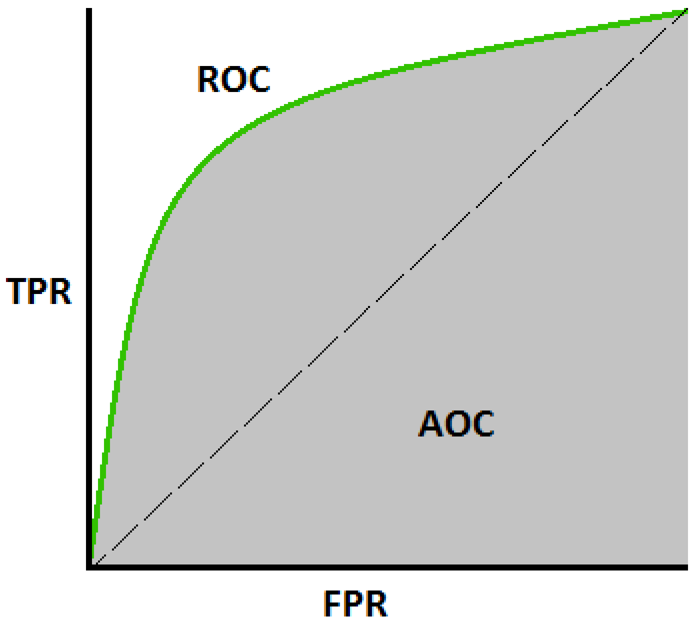


Fig. 6: ROC and AUC

The area under the curve (AOC), also known as the index of accuracy (A) or concordance index, is a perfect ROC curve performance metric. The greater the area under the curve, the higher the model's prediction power. A sample ROC curve is shown below. A perfect prediction model's ROC has TP equal to 1 and FP equal to 0. The upper left corner of the graph will be touched by this curve.

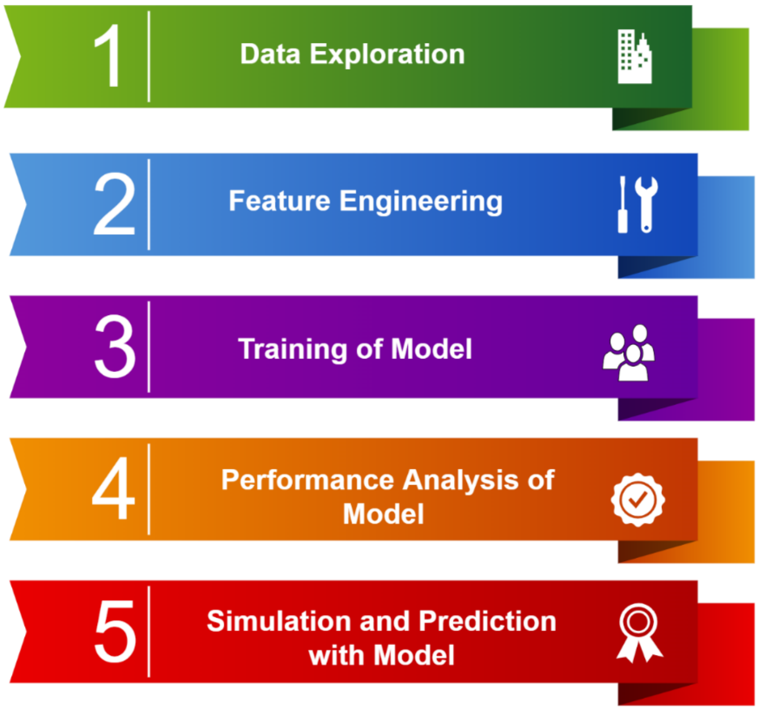
3. Akaike Information Criteria (AIC): This is the analogous metric of the adjusted R^2 in logistic regression. It is the measure of fit which penalizes model for the number of model coefficients. A minimum value for AIC is mostly preferred for a good model.

4. Null Deviance and Residual Deviance: Null Deviance indicates the response predicted by a model with nothing but an intercept. Lower the value, better the model. Residual deviance indicates the response predicted by a model on adding independent variables. Lower the value, better the model.

Simulation and Prediction

Methodology

The Project is carried out using data analysis modules from the python programming language. For a seamless and interactive experience, I used Jupiter Notebook as my programming environment for the project. The figure below summarizes the steps taken to simulate and predict the winner of the FIFA world cup.



1 Data Exploration
2 Feature Engineering
3 Training of Model
4 Performance Analysis of Model
5 Simulation and Prediction with Model

Fig. 7: Project Methodology

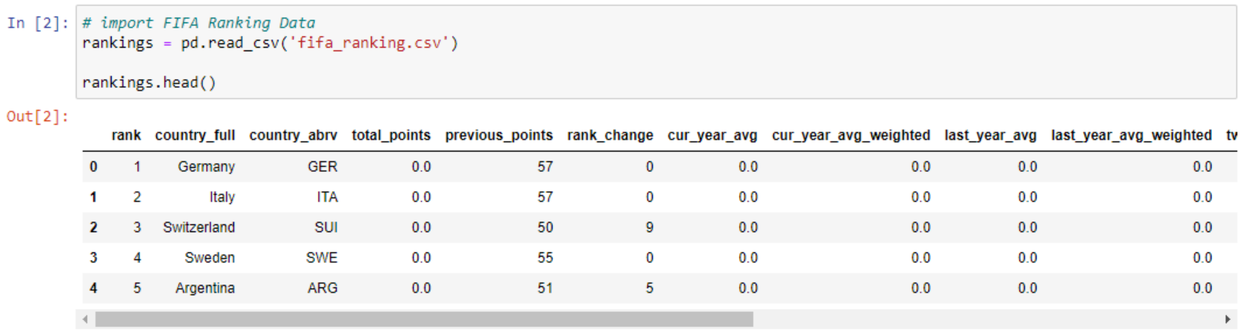
Note:

To make the report simplified as possible, I have only provided relevant screenshots of the codes and results in this report. However, I have attached the Jupiter notebook file containing the full codes (together with the datasets) to this report for your perusal if need be.

Data Exploration

I performed this simulation with three different sets of data as described below.

1. FIFA Soccer Rankings
   This dataset contains the official monthly FIFA rankings of the national teams from August 1993 to April, 2018. This ranking data serves as a basis for the general performance of the teams over the past years. I obtained this dataset from GitHub and it is accessible via this link.

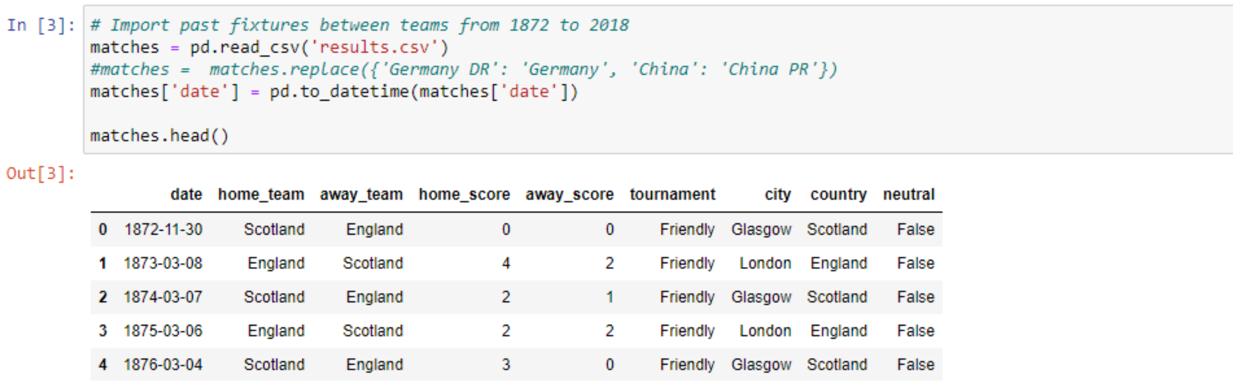


```
In [2]: # import FIFA Ranking Data
        rankings = pd.read_csv('fifa_ranking.csv')

        rankings.head()
```

Out[2]:

	rank	country_full	country_abrv	total_points	previous_points	rank_change	cur_year_avg	cur_year_avg_weighted	last_year_avg	last_year_avg_weighted	tv
0	1	Germany	GER	0.0	57	0	0.0	0.0	0.0	0.0	
1	2	Italy	ITA	0.0	57	0	0.0	0.0	0.0	0.0	
2	3	Switzerland	SUI	0.0	50	9	0.0	0.0	0.0	0.0	
3	4	Sweden	SWE	0.0	55	0	0.0	0.0	0.0	0.0	
4	5	Argentina	ARG	0.0	51	5	0.0	0.0	0.0	0.0	

2. Past International Soccer Matches
   This dataset contains match results between the various national teams from 1872 to 2018. This helps to ascertain the strengths of the teams against one another. I obtained this dataset from Kaggle. It is accessible via this link .



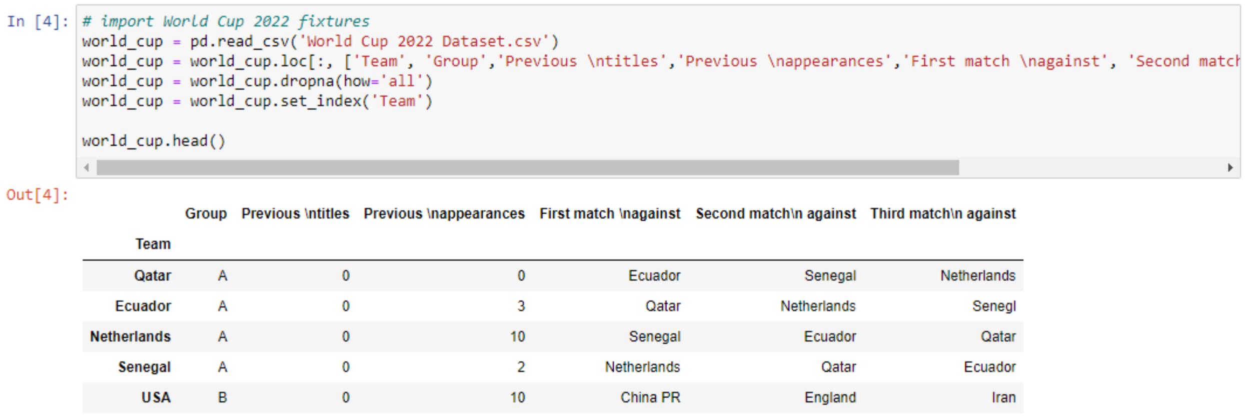
```
In [3]: # Import past fixtures between teams from 1872 to 2018
        matches = pd.read_csv('results.csv')
        #matches =  matches.replace({'Germany DR': 'Germany', 'China': 'China PR'})
        matches['date'] = pd.to_datetime(matches['date'])

        matches.head()
```

Out[3]:

	date	home_team	away_team	home_score	away_score	tournament	city	country	neutral
0	1872-11-30	Scotland	England	0	0	Friendly	Glasgow	Scotland	False
1	1873-03-08	England	Scotland	4	2	Friendly	London	England	False
2	1874-03-07	Scotland	England	2	1	Friendly	Glasgow	Scotland	False
3	1875-03-06	England	Scotland	2	2	Friendly	London	England	False
4	1876-03-04	Scotland	England	3	0	Friendly	Glasgow	Scotland	False

3. FIFA World Cup, 2022 dataset
   This dataset contains the match pairings for the impending FIFA world cup. I obtained the 2018 dataset from Kaggle via this link and I personally modified the contents with the updated information as available from FIFA's official website. At the time I was carrying this project, three (3) slots were yet to be populated as a result of delayed fixtures owing

to the Russia-Ukraine conflict so I filled those slots with 3 teams (China, Nigeria and Italy) to make up for those slots.



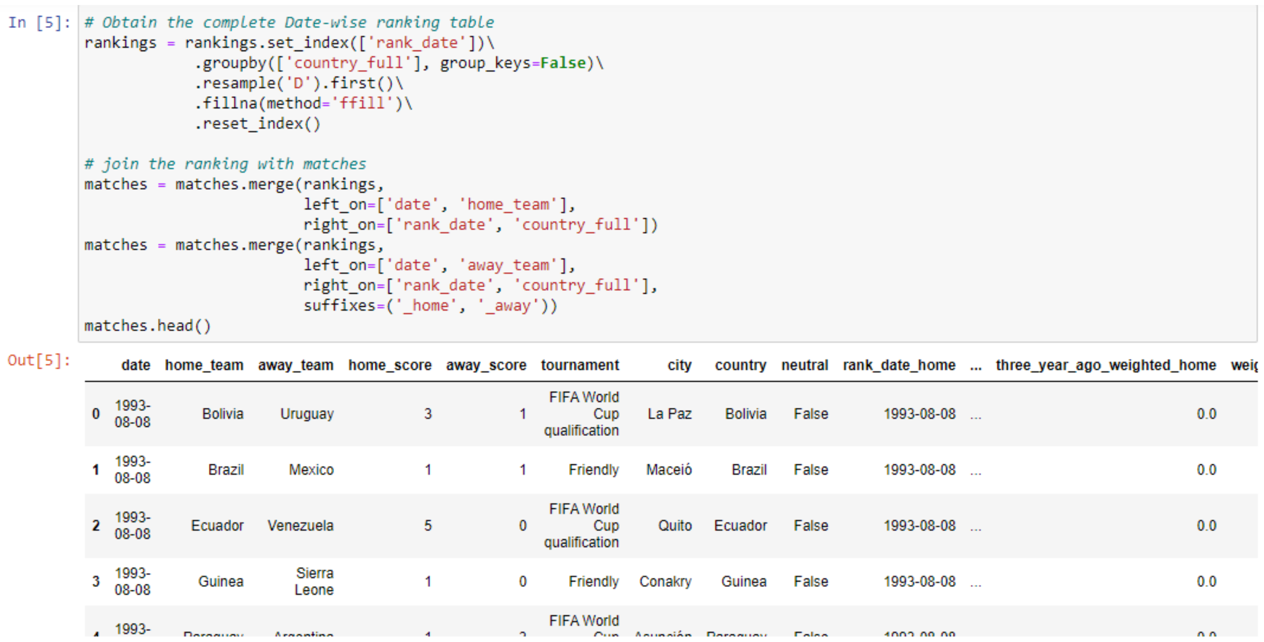
```
In [4]: # import World Cup 2022 fixtures
        world_cup = pd.read_csv('World Cup 2022 Dataset.csv')
        world_cup = world_cup.loc[:, ['Team', 'Group','Previous \ntitles','Previous \nappearances','First match \nagainst', 'Second match
        world_cup = world_cup.dropna(how='all')
        world_cup = world_cup.set_index('Team')

        world_cup.head()
```

Out[4]:

	Group	Previous \ntitles	Previous \nappearances	First match \nagainst	Second match\n against	Third match\n against
Team						
Qatar	A	0	0	Ecuador	Senegal	Netherlands
Ecuador	A	0	3	Qatar	Netherlands	Senegl
Netherlands	A	0	10	Senegal	Ecuador	Qatar
Senegal	A	0	2	Netherlands	Qatar	Ecuador
USA	B	0	10	China PR	England	Iran

Feature Engineering

In this section, I carefully selected and generated features that I believe were important to help in the prediction of the model. Features such as a team's average rank, the rank-difference between any two teams, the point difference between the teams and others were all generated from the existing datasets after I had joined relevant features from the ranking with that of the matches data frame.

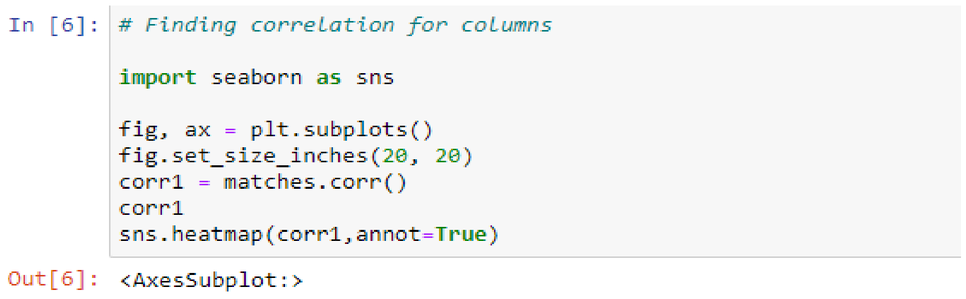


```
In [5]: # Obtain the complete Date-wise ranking table
        rankings = rankings.set_index(['rank_date'])\
                    .groupby(['country_full'], group_keys=False)\
                    .resample('D').first()\
                    .fillna(method='ffill')\
                    .reset_index()

        # join the ranking with matches
        matches = matches.merge(rankings,
                                left_on=['date', 'home_team'],
                                right_on=['rank_date', 'country_full'])
        matches = matches.merge(rankings,
                                left_on=['date', 'away_team'],
                                right_on=['rank_date', 'country_full'],
                                suffixes=('_home', '_away'))
        matches.head()
```

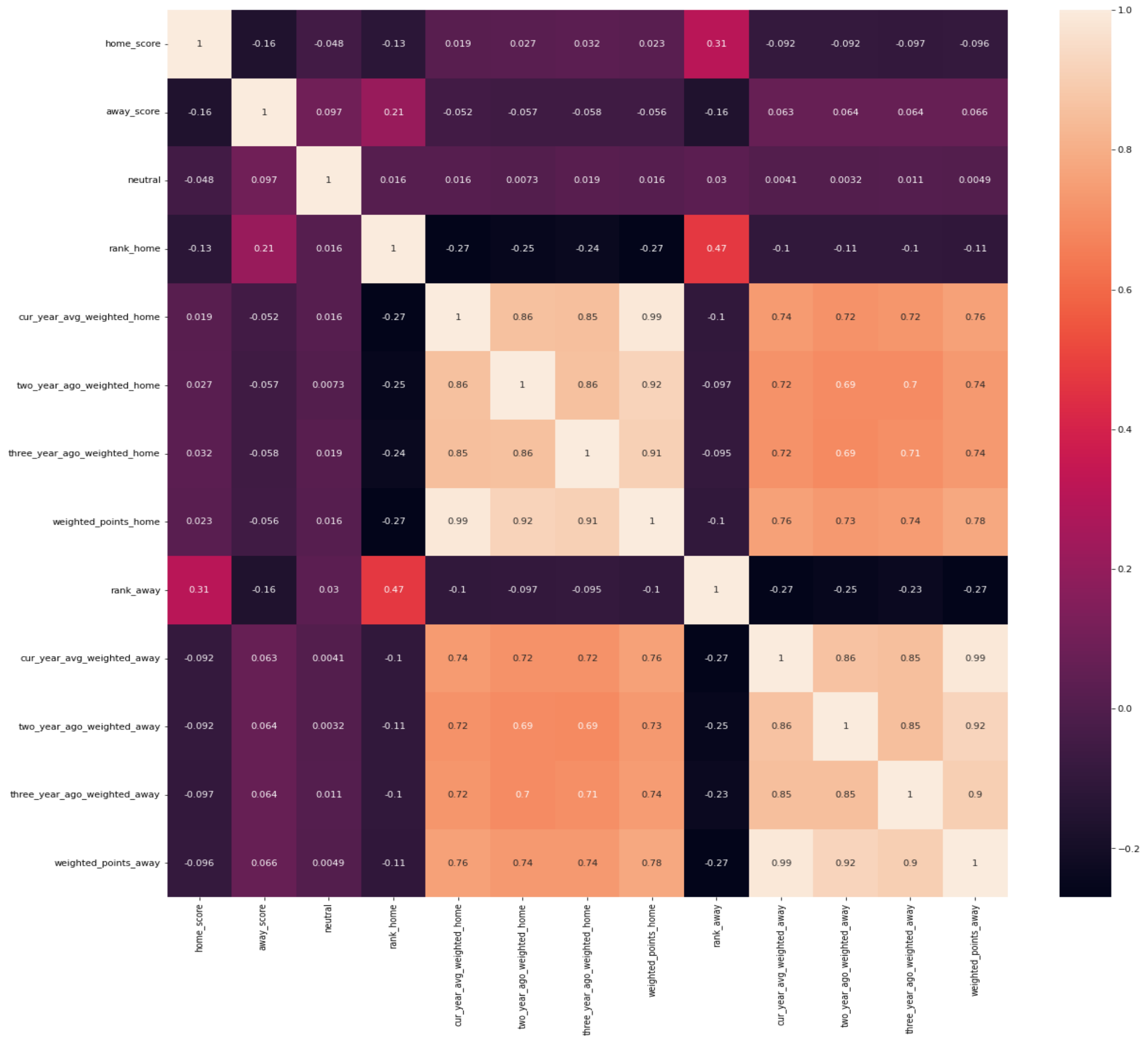
Out[5]:

	date	home_team	away_team	home_score	away_score	tournament	city	country	neutral	rank_date_home	...	three_year_ago_weighted_home	weig
0	1993-08-08	Bolivia	Uruguay	3	1	FIFA World Cup qualification	La Paz	Bolivia	False	1993-08-08	...	0.0	
1	1993-08-08	Brazil	Mexico	1	1	Friendly	Maceió	Brazil	False	1993-08-08	...	0.0	
2	1993-08-08	Ecuador	Venezuela	5	0	FIFA World Cup qualification	Quito	Ecuador	False	1993-08-08	...	0.0	
3	1993-08-08	Guinea	Sierra Leone	1	0	Friendly	Conakry	Guinea	False	1993-08-08	...	0.0	
4	1993-	Paraguay	Argentina	1	2	FIFA World Cup	Asunción	Paraguay	False	1993-08-08		0.0	

Also, the correlation between the various features were examined and it provided some good results and a great motivation to continue with the project.

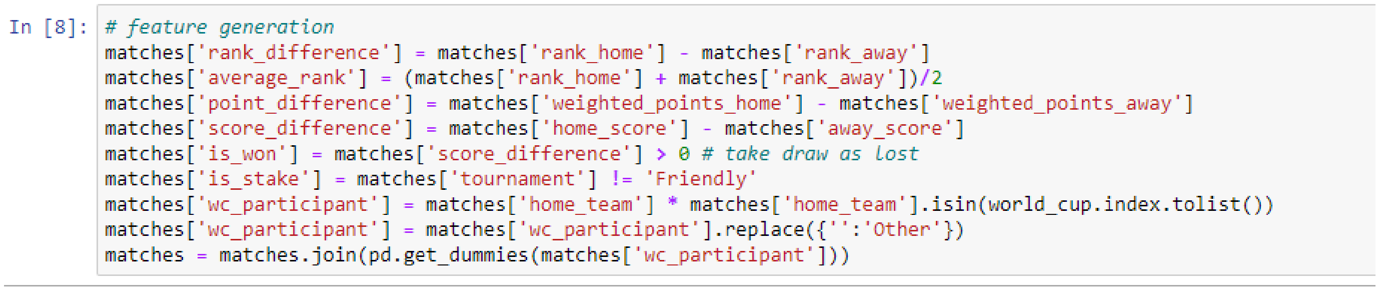


```
In [6]: # Finding correlation for columns

        import seaborn as sns

        fig, ax = plt.subplots()
        fig.set_size_inches(20, 20)
        corr1 = matches.corr()
        corr1
        sns.heatmap(corr1,annot=True)

Out[6]: <AxesSubplot:>
```



	home_score	away_score	neutral	rank_home	cur_year_avg_weighted_home	two_year_ago_weighted_home	three_year_ago_weighted_home	weighted_points_home	rank_away	cur_year_avg_weighted_away	two_year_ago_weighted_away	three_year_ago_weighted_away	weighted_points_away
home_score	1	-0.16	-0.048	-0.13	0.019	0.027	0.032	0.023	0.31	-0.092	-0.092	-0.097	-0.096
away_score	-0.16	1	0.097	0.21	-0.052	-0.057	-0.058	-0.056	-0.16	0.063	0.064	0.064	0.066
neutral	-0.048	0.097	1	0.016	0.016	0.0073	0.019	0.016	0.03	0.0041	0.0032	0.011	0.0049
rank_home	-0.13	0.21	0.016	1	-0.27	-0.25	-0.24	-0.27	0.47	-0.1	-0.11	-0.1	-0.11
cur_year_avg_weighted_home	0.019	-0.052	0.016	-0.27	1	0.86	0.85	0.99	-0.1	0.74	0.72	0.72	0.76
two_year_ago_weighted_home	0.027	-0.057	0.0073	-0.25	0.86	1	0.86	0.92	-0.097	0.72	0.69	0.7	0.74
three_year_ago_weighted_home	0.032	-0.058	0.019	-0.24	0.85	0.86	1	0.91	-0.095	0.72	0.69	0.71	0.74
weighted_points_home	0.023	-0.056	0.016	-0.27	0.99	0.92	0.91	1	-0.1	0.76	0.73	0.74	0.78
rank_away	0.31	-0.16	0.03	0.47	-0.1	-0.097	-0.095	-0.1	1	-0.27	-0.25	-0.23	-0.27
cur_year_avg_weighted_away	-0.092	0.063	0.0041	-0.1	0.74	0.72	0.72	0.76	-0.27	1	0.86	0.85	0.99
two_year_ago_weighted_away	-0.092	0.064	0.0032	-0.11	0.72	0.69	0.69	0.73	-0.25	0.86	1	0.85	0.92
three_year_ago_weighted_away	-0.097	0.064	0.011	-0.1	0.72	0.7	0.71	0.74	-0.23	0.85	0.85	1	0.9
weighted_points_away	-0.096	0.066	0.0049	-0.11	0.76	0.74	0.74	0.78	-0.27	0.99	0.92	0.9	1

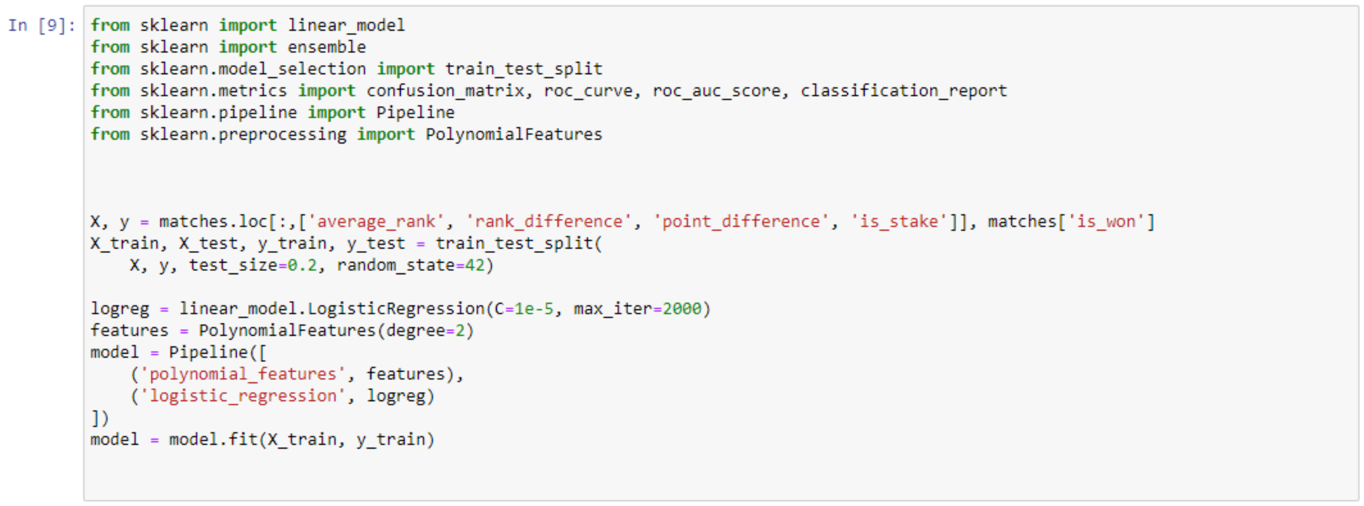
The features discussed previously were then generated into new columns with the following codes.



```
In [8]: # feature generation
        matches['rank_difference'] = matches['rank_home'] - matches['rank_away']
        matches['average_rank'] = (matches['rank_home'] + matches['rank_away'])/2
        matches['point_difference'] = matches['weighted_points_home'] - matches['weighted_points_away']
        matches['score_difference'] = matches['home_score'] - matches['away_score']
        matches['is_won'] = matches['score_difference'] > 0 # take draw as lost
        matches['is_stake'] = matches['tournament'] != 'Friendly'
        matches['wc_participant'] = matches['home_team'] * matches['home_team'].isin(world_cup.index.tolist())
        matches['wc_participant'] = matches['wc_participant'].replace({'':'Other'})
        matches = matches.join(pd.get_dummies(matches['wc_participant']))
```

Data Modelling

In this section, I fit the engineered feature data to a logistic regression model. Before, I do so, I divided my data into training data and testing data in a 8:2 ratio respectively. The coefficients are automatically computed by the logistic regression module and the test data.



```
In [9]: from sklearn import linear_model
        from sklearn import ensemble
        from sklearn.model_selection import train_test_split
        from sklearn.metrics import confusion_matrix, roc_curve, roc_auc_score, classification_report
        from sklearn.pipeline import Pipeline
        from sklearn.preprocessing import PolynomialFeatures



        X, y = matches.loc[:,['average_rank', 'rank_difference', 'point_difference', 'is_stake']], matches['is_won']
        X_train, X_test, y_train, y_test = train_test_split(
            X, y, test_size=0.2, random_state=42)

        logreg = linear_model.LogisticRegression(C=1e-5, max_iter=2000)
        features = PolynomialFeatures(degree=2)
        model = Pipeline([
            ('polynomial_features', features),
            ('logistic_regression', logreg)
        ])
        model = model.fit(X_train, y_train)
```

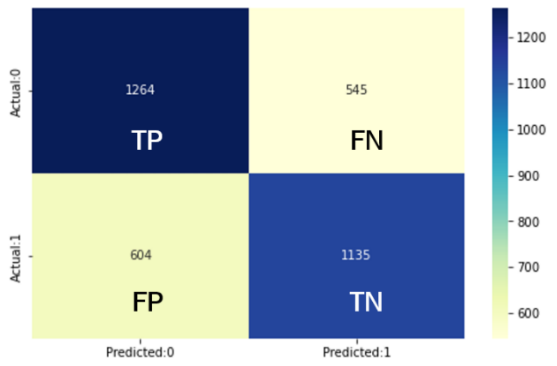
Performance Analysis of Trained Model

With the test data and the generated logit model, I access the performance of the model by analyzing the model accuracy from the confusion matrix and also by observing the ROC curve and its subsequent AUC. The trained model produced an accuracy of approximately 68% which is quite good, although it can be improved with further intensive feature engineering. Also, the AUC score was 0.75 which is really good for the model.

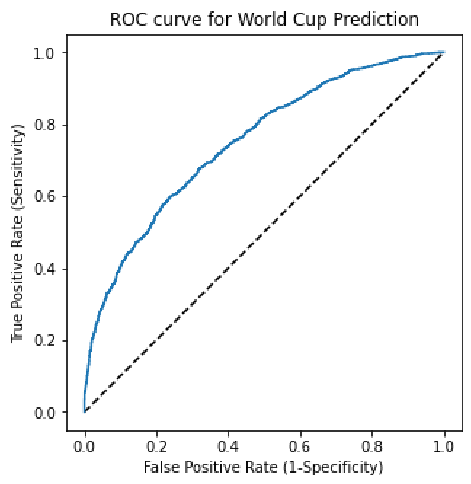
```
In [10]: # Performance Evaluation: Accuracy and Confusion Matrix

mod_accuracy = model.score(X_test,y_test)
print("Logistic Regression accuracy is :",mod_accuracy)

lr_pred= model.predict(X_test)
report = classification_report(y_test,lr_pred)
print(report)

cm = confusion_matrix(y_test, model.predict(X_test))
conf_matrix=pd.DataFrame(data=cm,columns=['Predicted:0','Predicted:1'],index=['Actual:0','Actual:1'])
plt.figure(figsize = (8,5))
sns.heatmap(conf_matrix, annot=True,fmt='d',cmap="YlGnBu")
```

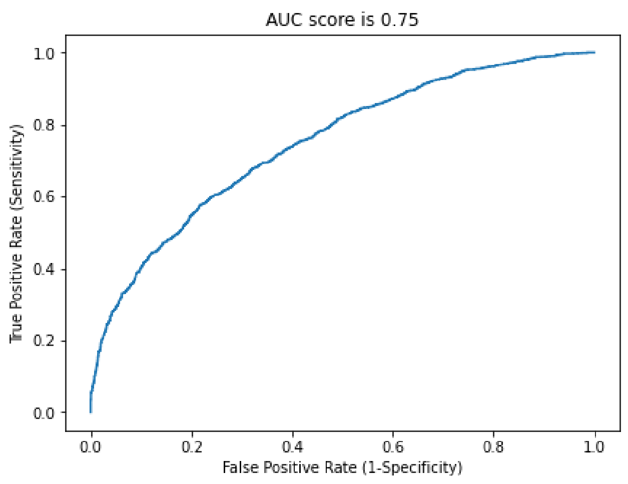
```
Logistic Regression accuracy is : 0.6761555806087937
              precision    recall  f1-score   support

       False       0.68      0.70      0.69      1809
        True       0.68      0.65      0.66      1739

    accuracy                           0.68      3548
   macro avg       0.68      0.68      0.68      3548
weighted avg       0.68      0.68      0.68      3548
```

```
Out[10]: <AxesSubplot:>
```



```
In [11]: # Performance evaluation: ROC and AUC


fpr, tpr, _ = roc_curve(y_test, model.predict_proba(X_test)[:,1])
plt.figure(figsize=(15,5))
ax = plt.subplot(1,2,1)
ax.plot([0, 1], [0, 1], 'k--')
ax.plot(fpr, tpr)
ax.set_title('ROC curve for World Cup Prediction')
ax.set_xlabel('False Positive Rate (1-Specificity)')
ax.set_ylabel('True Positive Rate (Sensitivity)')
ax.set_aspect(1)


ax = plt.subplot(1,2,2)
ax.plot(fpr, tpr)
ax.set_title('AUC score is {0:0.2}'.format(roc_auc_score(y_test, model.predict_proba(X_test)[:,1])))
ax.set_xlabel('False Positive Rate (1-Specificity)')
ax.set_ylabel('True Positive Rate (Sensitivity)')

pass
```



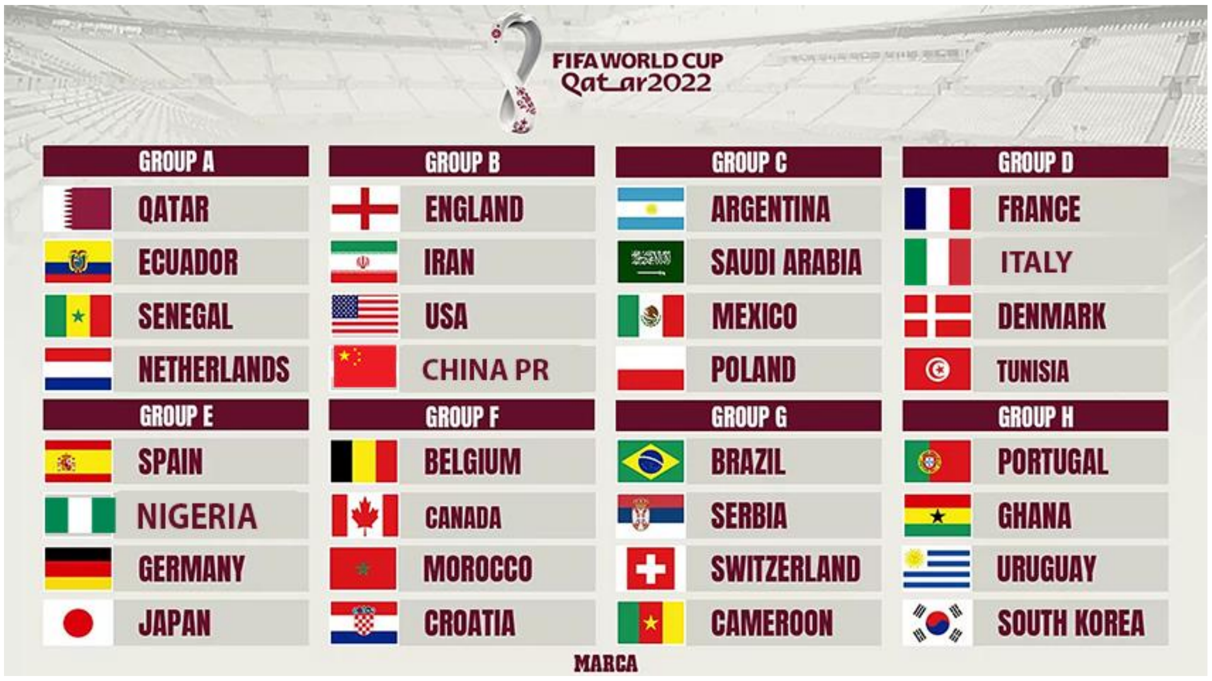


World Cup Simulation

With a good fit model, I proceeded to simulate the world cup matches and to predict the winners of individual match right from the group stages to the final match.

Group Stages

The FIFA world cup dataset contained the fixed matches as shown above in the data exploration section. Below is a graphical representation of the groups.



FIFA WORLD CUP Qatar2022

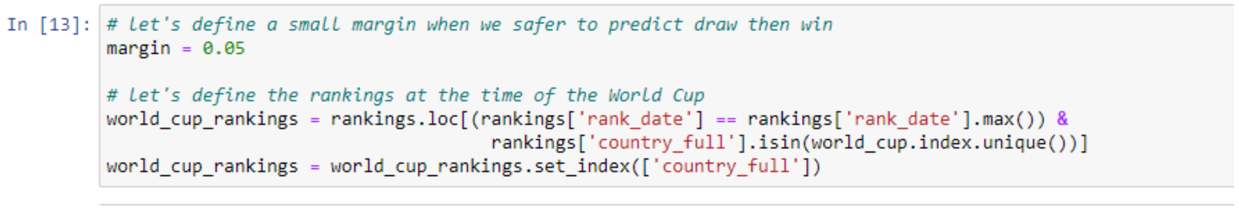
GROUP A	GROUP B	GROUP C	GROUP D
QATAR	ENGLAND	ARGENTINA	FRANCE
ECUADOR	IRAN	SAUDI ARABIA	ITALY
SENEGAL	USA	MEXICO	DENMARK
NETHERLANDS	CHINA PR	POLAND	TUNISIA

GROUP E	GROUP F	GROUP G	GROUP H
SPAIN	BELGIUM	BRAZIL	PORTUGAL
NIGERIA	CANADA	SERBIA	GHANA
GERMANY	MOROCCO	SWITZERLAND	URUGUAY
JAPAN	CROATIA	CAMEROON	SOUTH KOREA

MARCA

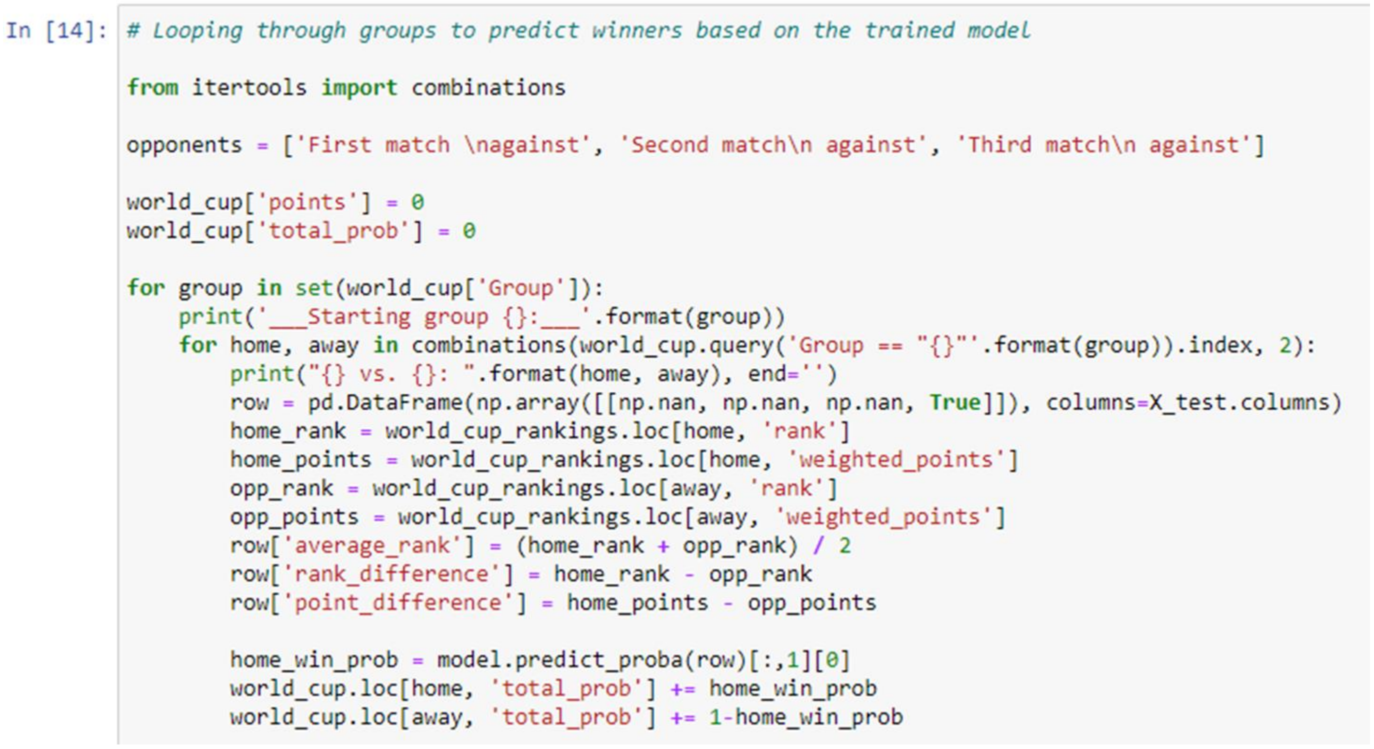
Fig. 8: Visualized group stage data for the Qatar 2022 FIFA World Cup

First, a small safe margin of 0.05 was defined to help predict draws as well. Also, the current ranking of each team was obtained as a direct feature.



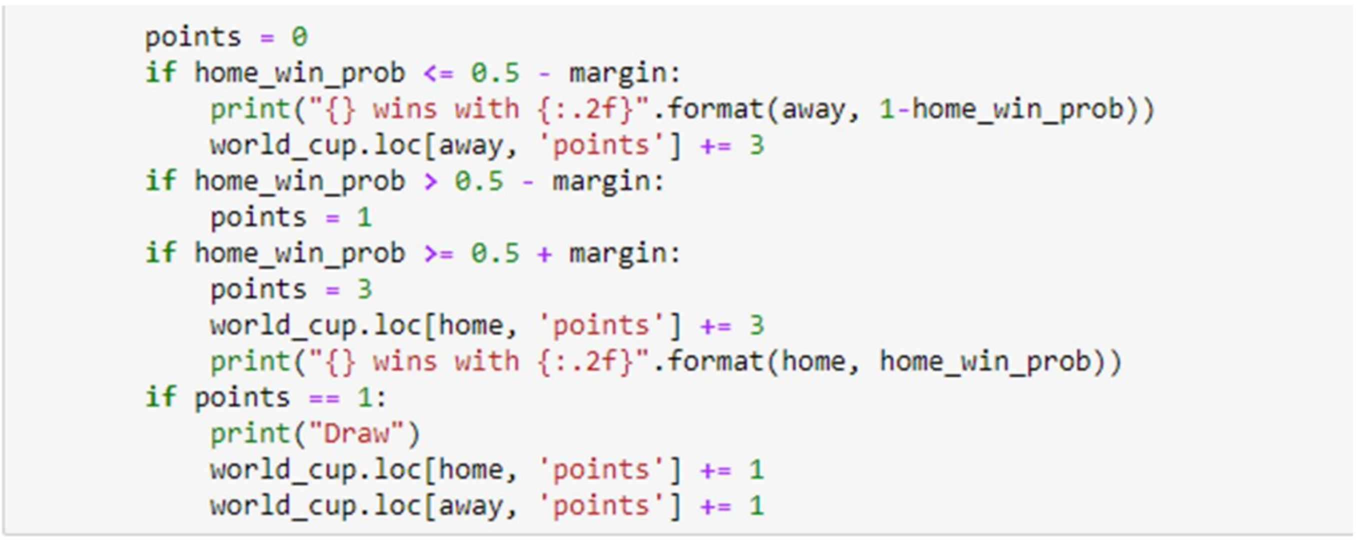
```
In [13]: # let's define a small margin when we safer to predict draw then win
         margin = 0.05

         # let's define the rankings at the time of the World Cup
         world_cup_rankings = rankings.loc[(rankings['rank_date'] == rankings['rank_date'].max()) &
                                     rankings['country_full'].isin(world_cup.index.unique())]
         world_cup_rankings = world_cup_rankings.set_index(['country_full'])
```

Then, I iterated through each group and each match and predicted the probability of the home team winning. When the probability of win for the home team is greater than the sum of threshold value of 0.5 and the safe margin ($0.5 + margin$), the home team wins and is credited with the full $3\ points$. And when it is lower than the difference between the threshold and margin ($0.5 - margin$), the way team is credited with the win and the corresponding $3\ points$. And finally, when the probability of win for the home team is greater than the difference between the threshold and the margin, then the match is drawn and each team is credited with $a\ point$.



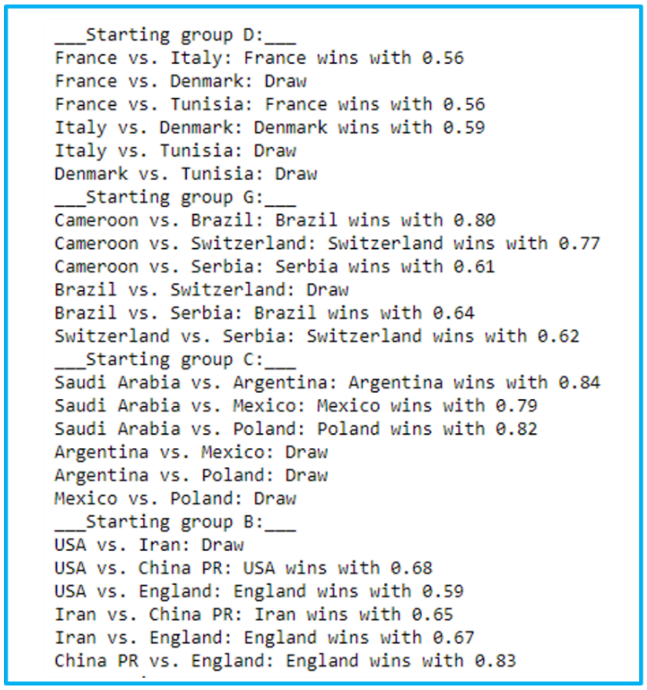
```
In [14]: # Looping through groups to predict winners based on the trained model

         from itertools import combinations

         opponents = ['First match \nagainst', 'Second match\n against', 'Third match\n against']

         world_cup['points'] = 0
         world_cup['total_prob'] = 0

         for group in set(world_cup['Group']):
             print('___Starting group {}:___'.format(group))
             for home, away in combinations(world_cup.query('Group == "{}"'.format(group)).index, 2):
                 print("{} vs. {}: ".format(home, away), end='')
                 row = pd.DataFrame(np.array([[np.nan, np.nan, np.nan, True]]), columns=X_test.columns)
                 home_rank = world_cup_rankings.loc[home, 'rank']
                 home_points = world_cup_rankings.loc[home, 'weighted_points']
                 opp_rank = world_cup_rankings.loc[away, 'rank']
                 opp_points = world_cup_rankings.loc[away, 'weighted_points']
                 row['average_rank'] = (home_rank + opp_rank) / 2
                 row['rank_difference'] = home_rank - opp_rank
                 row['point_difference'] = home_points - opp_points

                 home_win_prob = model.predict_proba(row)[:,1][0]
                 world_cup.loc[home, 'total_prob'] += home_win_prob
                 world_cup.loc[away, 'total_prob'] += 1-home_win_prob
```

Code continued on next page:

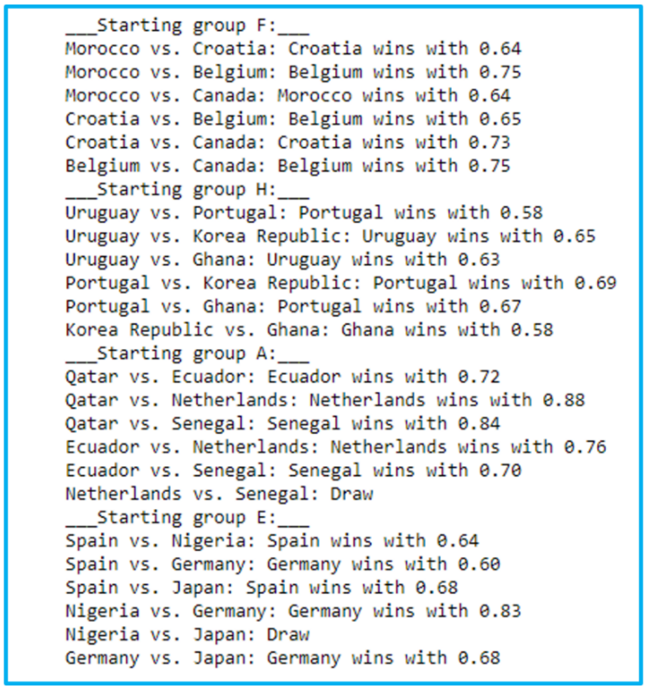


```
        points = 0
        if home_win_prob <= 0.5 - margin:
            print("{} wins with {:.2f}".format(away, 1-home_win_prob))
            world_cup.loc[away, 'points'] += 3
        if home_win_prob > 0.5 - margin:
            points = 1
        if home_win_prob >= 0.5 + margin:
            points = 3
            world_cup.loc[home, 'points'] += 3
            print("{} wins with {:.2f}".format(home, home_win_prob))
        if points == 1:
            print("Draw")
            world_cup.loc[home, 'points'] += 1
            world_cup.loc[away, 'points'] += 1
```

After running the simulation, the results were as follows:

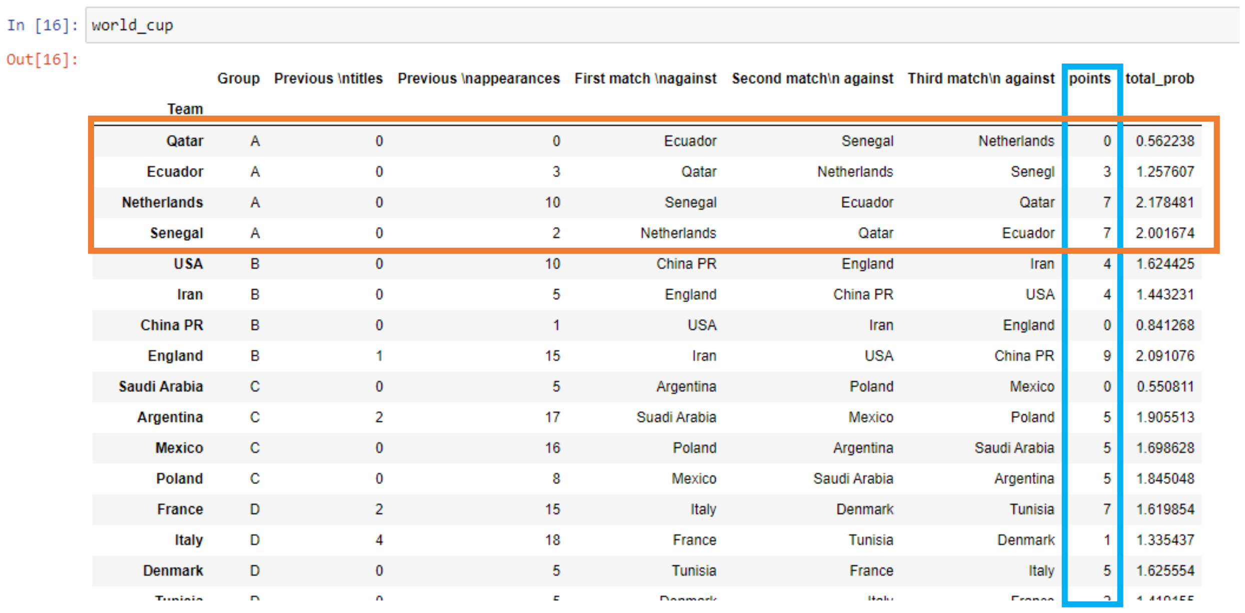


```
___Starting group D:___
France vs. Italy: France wins with 0.56
France vs. Denmark: Draw
France vs. Tunisia: France wins with 0.56
Italy vs. Denmark: Denmark wins with 0.59
Italy vs. Tunisia: Draw
Denmark vs. Tunisia: Draw
___Starting group G:___
Cameroon vs. Brazil: Brazil wins with 0.80
Cameroon vs. Switzerland: Switzerland wins with 0.77
Cameroon vs. Serbia: Serbia wins with 0.61
Brazil vs. Switzerland: Draw
Brazil vs. Serbia: Brazil wins with 0.64
Switzerland vs. Serbia: Switzerland wins with 0.62
___Starting group C:___
Saudi Arabia vs. Argentina: Argentina wins with 0.84
Saudi Arabia vs. Mexico: Mexico wins with 0.79
Saudi Arabia vs. Poland: Poland wins with 0.82
Argentina vs. Mexico: Draw
Argentina vs. Poland: Draw
Mexico vs. Poland: Draw
___Starting group B:___
USA vs. Iran: Draw
USA vs. China PR: USA wins with 0.68
USA vs. England: England wins with 0.59
Iran vs. China PR: Iran wins with 0.65
Iran vs. England: England wins with 0.67
China PR vs. England: England wins with 0.83
```



```
___Starting group F:___
Morocco vs. Croatia: Croatia wins with 0.64
Morocco vs. Belgium: Belgium wins with 0.75
Morocco vs. Canada: Morocco wins with 0.64
Croatia vs. Belgium: Belgium wins with 0.65
Croatia vs. Canada: Croatia wins with 0.73
Belgium vs. Canada: Belgium wins with 0.75
___Starting group H:___
Uruguay vs. Portugal: Portugal wins with 0.58
Uruguay vs. Korea Republic: Uruguay wins with 0.65
Uruguay vs. Ghana: Uruguay wins with 0.63
Portugal vs. Korea Republic: Portugal wins with 0.69
Portugal vs. Ghana: Portugal wins with 0.67
Korea Republic vs. Ghana: Ghana wins with 0.58
___Starting group A:___
Qatar vs. Ecuador: Ecuador wins with 0.72
Qatar vs. Netherlands: Netherlands wins with 0.88
Qatar vs. Senegal: Senegal wins with 0.84
Ecuador vs. Netherlands: Netherlands wins with 0.76
Ecuador vs. Senegal: Senegal wins with 0.70
Netherlands vs. Senegal: Draw
___Starting group E:___
Spain vs. Nigeria: Spain wins with 0.64
Spain vs. Germany: Germany wins with 0.60
Spain vs. Japan: Spain wins with 0.68
Nigeria vs. Germany: Germany wins with 0.83
Nigeria vs. Japan: Draw
Germany vs. Japan: Germany wins with 0.68
```

A look at the world cup data frame shows the updated points of the various teams in their groups. As shown below, Senegal and Netherlands of group A both made 7 points and being the top two teams of the group, they qualify for the next round of the tournament, which is the elimination stages. Similar analysis can be made for other groups to ascertain the teams that qualify from the other groups.

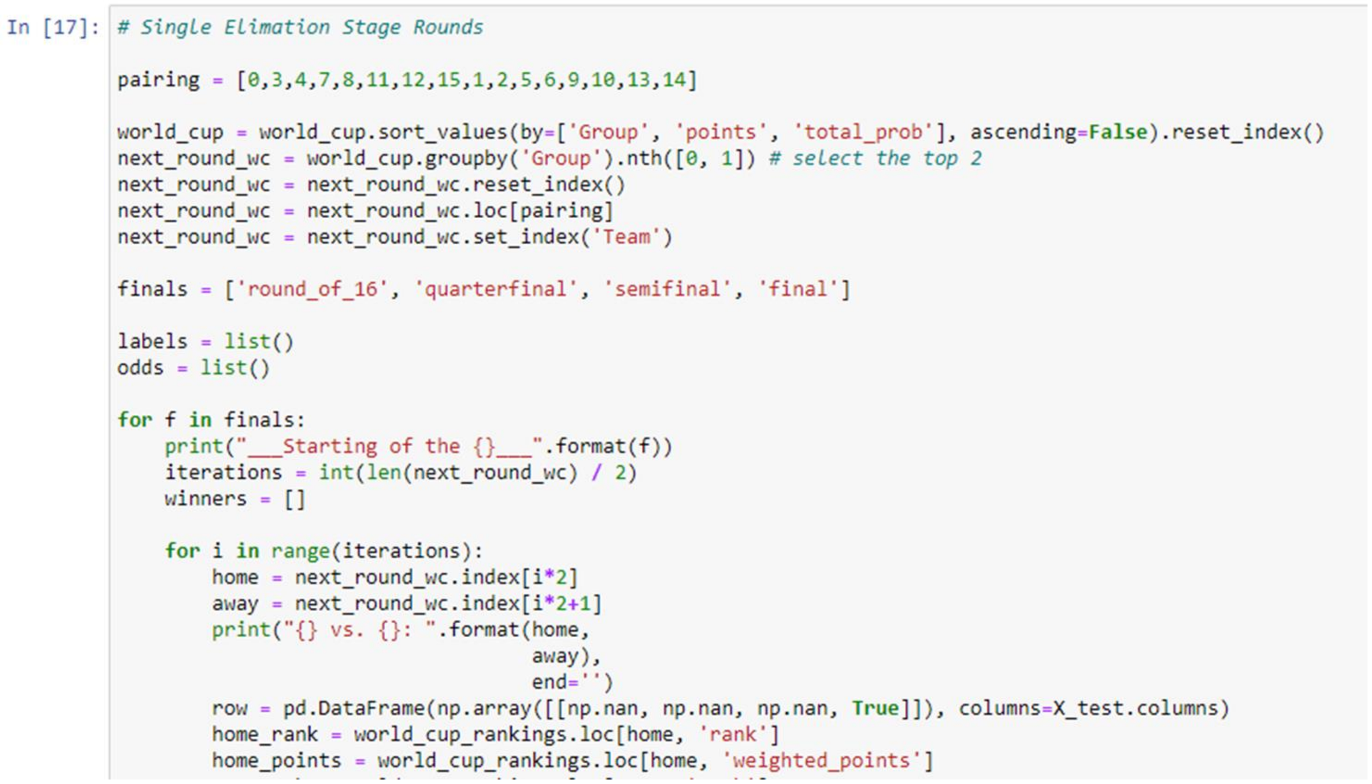


```
In [16]: world_cup
Out[16]:
```

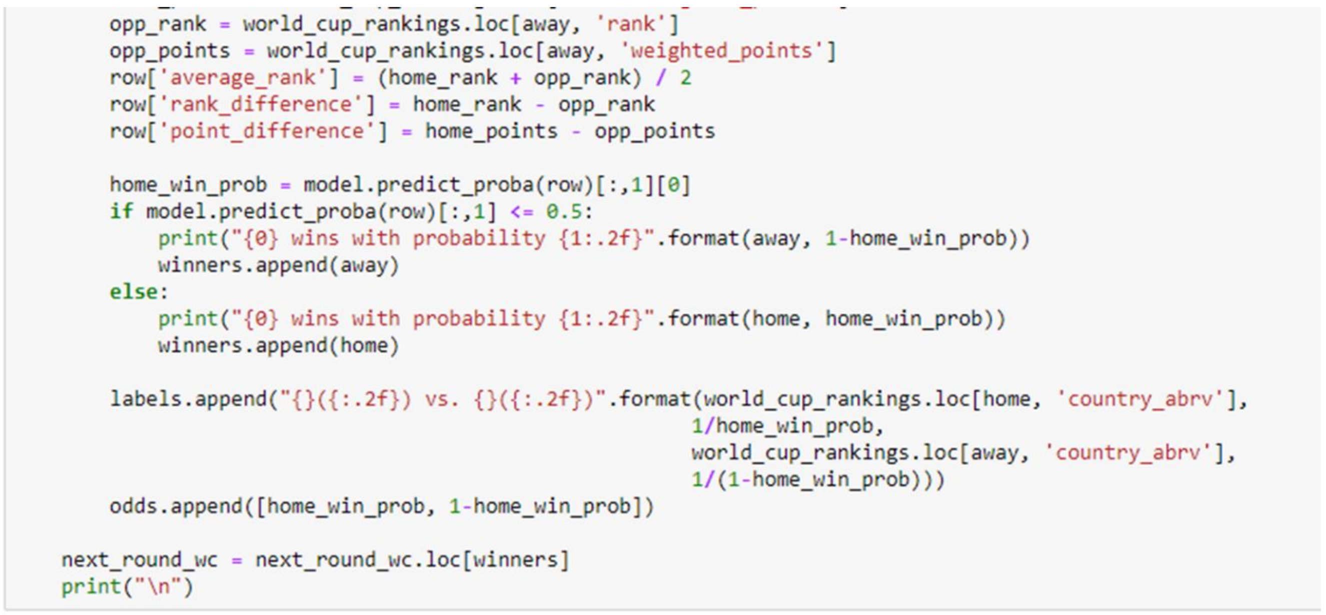
Team	Group	Previous \ntitles	Previous \nappearances	First match \nagainst	Second match\n against	Third match\n against	points	total_prob
Qatar	A	0	0	Ecuador	Senegal	Netherlands	0	0.562238
Ecuador	A	0	3	Qatar	Netherlands	Senegl	3	1.257607
Netherlands	A	0	10	Senegal	Ecuador	Qatar	7	2.178481
Senegal	A	0	2	Netherlands	Qatar	Ecuador	7	2.001674
USA	B	0	10	China PR	England	Iran	4	1.624425
Iran	B	0	5	England	China PR	USA	4	1.443231
China PR	B	0	1	USA	Iran	England	0	0.841268
England	B	1	15	Iran	USA	China PR	9	2.091076
Saudi Arabia	C	0	5	Argentina	Poland	Mexico	0	0.550811
Argentina	C	2	17	Suadi Arabia	Mexico	Poland	5	1.905513
Mexico	C	0	16	Poland	Argentina	Saudi Arabia	5	1.698628
Poland	C	0	8	Mexico	Saudi Arabia	Argentina	5	1.845048
France	D	2	15	Italy	Denmark	Tunisia	7	1.619854
Italy	D	4	18	France	Tunisia	Denmark	1	1.335437
Denmark	D	0	5	Tunisia	France	Italy	5	1.625554
Tunisia	D	0	5	Denmark	Italy	France	[illegible]	[illegible]

Elimination Stage

At this stage, I simulated the round 16, quarter-finals, semi-finals and final games of the world cup. A fixed paring is made based on the FIFA standard where the team that topped a chosen group, say group H, plays the runner-up of another, say group D and so on.

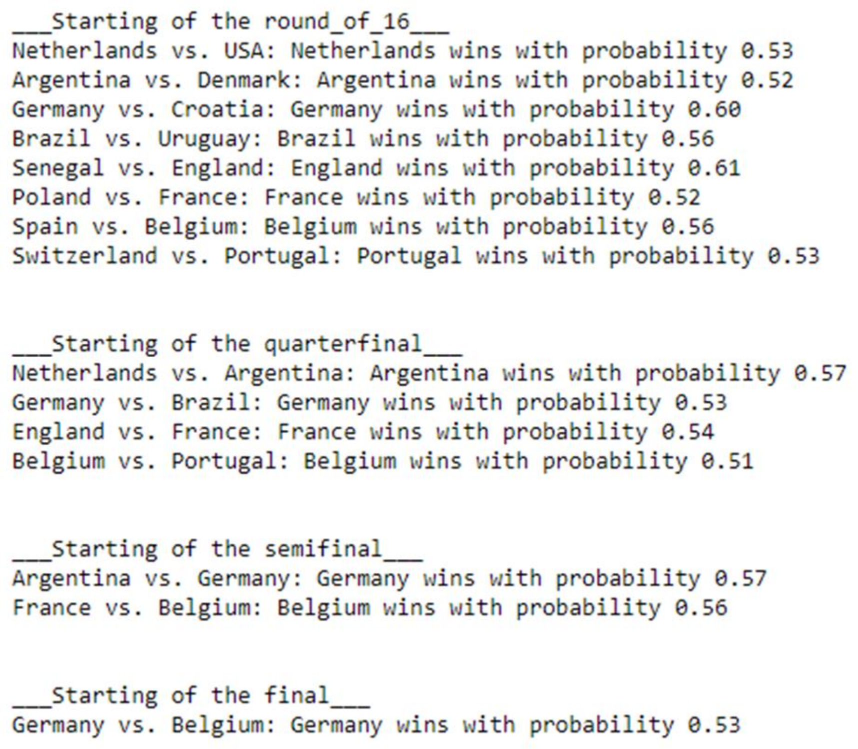


```
In [17]: # Single Elimation Stage Rounds

         pairing = [0,3,4,7,8,11,12,15,1,2,5,6,9,10,13,14]

         world_cup = world_cup.sort_values(by=['Group', 'points', 'total_prob'], ascending=False).reset_index()
         next_round_wc = world_cup.groupby('Group').nth([0, 1]) # select the top 2
         next_round_wc = next_round_wc.reset_index()
         next_round_wc = next_round_wc.loc[pairing]
         next_round_wc = next_round_wc.set_index('Team')

         finals = ['round_of_16', 'quarterfinal', 'semifinal', 'final']

         labels = list()
         odds = list()

         for f in finals:
             print("___Starting of the {}___".format(f))
             iterations = int(len(next_round_wc) / 2)
             winners = []

             for i in range(iterations):
                 home = next_round_wc.index[i*2]
                 away = next_round_wc.index[i*2+1]
                 print("{} vs. {}: ".format(home,
                                            away),
                                            end='')
                 row = pd.DataFrame(np.array([[np.nan, np.nan, np.nan, True]]), columns=X_test.columns)
                 home_rank = world_cup_rankings.loc[home, 'rank']
                 home_points = world_cup_rankings.loc[home, 'weighted_points']
```



```
        opp_rank = world_cup_rankings.loc[away, 'rank']
        opp_points = world_cup_rankings.loc[away, 'weighted_points']
        row['average_rank'] = (home_rank + opp_rank) / 2
        row['rank_difference'] = home_rank - opp_rank
        row['point_difference'] = home_points - opp_points

        home_win_prob = model.predict_proba(row)[:,1][0]
        if model.predict_proba(row)[:,1] <= 0.5:
            print("{0} wins with probability {1:.2f}".format(away, 1-home_win_prob))
            winners.append(away)
        else:
            print("{0} wins with probability {1:.2f}".format(home, home_win_prob))
            winners.append(home)

        labels.append("{}({:.2f}) vs. {}({:.2f})".format(world_cup_rankings.loc[home, 'country_abrv'],
                                                        1/home_win_prob,
                                                        world_cup_rankings.loc[away, 'country_abrv'],
                                                        1/(1-home_win_prob)))
        odds.append([home_win_prob, 1-home_win_prob])

    next_round_wc = next_round_wc.loc[winners]
    print("\n")
```

After running the above simulated elimination stage games, the results were as follows:



```
___Starting of the round_of_16___
Netherlands vs. USA: Netherlands wins with probability 0.53
Argentina vs. Denmark: Argentina wins with probability 0.52
Germany vs. Croatia: Germany wins with probability 0.60
Brazil vs. Uruguay: Brazil wins with probability 0.56
Senegal vs. England: England wins with probability 0.61
Poland vs. France: France wins with probability 0.52
Spain vs. Belgium: Belgium wins with probability 0.56
Switzerland vs. Portugal: Portugal wins with probability 0.53


___Starting of the quarterfinal___
Netherlands vs. Argentina: Argentina wins with probability 0.57
Germany vs. Brazil: Germany wins with probability 0.53
England vs. France: France wins with probability 0.54
Belgium vs. Portugal: Belgium wins with probability 0.51


___Starting of the semifinal___
Argentina vs. Germany: Germany wins with probability 0.57
France vs. Belgium: Belgium wins with probability 0.56


___Starting of the final___
Germany vs. Belgium: Germany wins with probability 0.53
```

Which shows that Germany wins the world cup over Belgium in the final with a probability of 0.53.

Fig. 9: Germany with their world cup trophy at the 2014 FIFA world cup

Conclusion

In this Report:

- I demonstrated the use of logistic regression in predicting the winner of the world cup by simulating each individual match with a model accuracy of 68%. Germany won this simulated world cup.
- With further extensive Feature engineering, the accuracy and performance of the model can be enhanced.
- Also, I gave a general explanation on the underlying mathematical model behind Logistic Regression

Reference

[1] FIFA world Cup, Wikipedia content. Accessible via this link.

[2] Men's FIFA World Cup Statistics from FIFA's official site. Accessible via this link.

[3] Hosmer, David W and Lemeshow, Stanley (2000). Applied Logistic Regression (2nd ed.) Wiley (ISBN 978-0-471-35632-5)

[4] Tolles, Juliana; Meurer, William J (2016). "Logistic Regression Relating Patient Characteristics to Outcomes".

[5] Menard, Scott W. (2002). *Applied Logistic Regression* (2nd ed.). SAGE. ISBN 978-0-7619-2208-7.

[6] Murphy, Kevin P. (2012). *Machine Learning – A Probabilistic Perspective*. The MIT Press. pp. 245pp. ISBN 978-0-262-01802-9.

[7] Vittinghoff, E.; McCulloch, C. E. (12 January 2007). "Relaxing the Rule of Ten Events per Variable in Logistic and Cox Regression". *American Journal of Epidemiology*.

[8] FIFA Ranking Dataset. Accessible via this link.

[9] Past International Matches Dataset. Available via this link.

[10] World cup 2018 Dataset. Available via this link. Modified version attached to this report.