

**Institute of Engineering & Technology**

MINI PROJECT REPORT

**On**

**Cricket World Cup**

**PREDICTION**

**Submitted by**

**PRAFULLIT SHUKLA**

**Roll No: 171500222**

**SEC-A**

**NITIN KUMAR SRIVASTAVA**

**ROLL NO:171500208**

**SEC-A**

**ROHAN YADAV**

**ROLL NO:171500266**

**SEC-C**

Department of Computer Engineering & Applications

**Institute of Engineering & Technology**



**Department of computer Engineering and Applications**

**GLA University, Mathura**

**17 km. Stone NH#2, Mathura-Delhi Road, P.O. – Chaumuha,**

**Mathura – 281406**

**Declaration**

We hereby declare that the work which is being presented in the S Mini Project **“ICC Cricket World Cup Prediction”,** in partial fulfillment of the requirements for Mini project, is an authentic record of our own work carried under the supervision of **“Department of Computer Engineering & Applications, GLA University” .**

**Signature of Candidates:**

**Name of Candidates: Prafullit Shukla,Nitin Kumar Srivastava,Rohan Yadav**

**Course: B.TECH CSE**

**Year: 3rd Year**

**Semester: Vth Semester**

**SYNOPSIS**

MINI PROJECT – I (2019-20)

**PROJECT TITLE -** *Cricket World Cup Prediction*



GLA UNIVERSITY, MATHURA

Institute of Engineering & Technology

Team Members

Student 1

PRAFULLIT SHUKLA

(University Roll No)

171500222

Student 2

NITIN KUMAR SRIVASTAVA

(University Roll No)

171500208

Student 3

Rohan Yadav

(University Roll No)

171500266

**Supervised BY**

Vivek Kumar

Department of Computer Engineering & Applications

**About the Project**

We are predicting the result of a ODI WorldCup cricket matches using machine learning concepts such as supervised learning to predict the champs of the matches. We utilize career statistics and also the team performances such as batting and bowling performances in order to train the models We are using supervised learning algorithms to predict the outcome of the game and it will help the coaches of the team to learn and analyze where actually the team is going wrong and the area of the improvement.

**Motivation**

Cricket is one of the most popular and most watched team games in the world.  We can use some machine learning algorithms such as linear regression to predict the final score of the team batting first or logistic regression to predict the probability of winning when chasing starts.

**Requirements:**

1. **Hardware:**

* **Processors:i3 processor or above**
* **Disk space: 2 GB**
* **Operating systems: Windows\*10**
* **GPU: NVidia GTX**

1. **Software:**

* Anaconda Enterprise
* SPYDER(FOR ML ALGORITHM)
* PYTHON(FOR CODE WRITING)
* MS-OFFICE(FOR DATA STORAGE)

# 

# **c) Environment and tools:**

* Jupyter Notebook
* Numpy
* Pandas
* Seaborn
* Matplotlib
* Scikit-learn

**Future Prospects**

Machine learning in future can predict better result , more accurate percentage of winning a game, making more interesting and interactive for cricket fan. Machine learning can be used to check performance of a player. From previous data different types of prediction can be used using machine learning.

* Dataset - to improve dataset you could use 2018 and 2019 years into account by scraping them from the ESPN website and also possibly use the players data to assess the quality of each team player.
* A confusion matrix would be great to analyse which games the model got wrong.
* We could ensemble that is we could try stacking more models together to improve the accuracy.

**ACKNOWLEDGMENT**

We found this golden chance to acknowledge all those people who had blessed, encouraged and supported us technically and morally through all the phases of our project. We take this opportunity to express our profound sense of gratitude. We thank all mighty GOD for giving us this opportunity to express gratitude to all those who helped me in successful completion of this project.

We pay our immense gratitude to **Mr Vivek Kumar, Dept. of Computer Science And Engineering** for providing help and giving us a chance for showing our skills through continued support and co-operation during the concerned project. We deeply indebted to staff of Computer Dept., for their sincere co-operation and sparing time to answer questionnaires with their selfless efforts and co-operation because of which we are able to complete this project.

We are deeply indebted to our parents who have always been a perennial source of information, encouragement and inspiration for entire education required.

ABSTRACT

Prediction about the result of cricket matches astonishing as a research problem, especially due to its complexity, unpredictable assumption (weather and pitch conditions). Because the ultimate outcome of a cricket match is based on many aspect and unaccepted bearings therefore it is difficult responsibility to predict the exact and partial truth-based outcomes of cricket matches such a research expects a multi criteria decision making approach. The main objective of the project is to present and analyze thoroughly some interesting historical stats about the World Cup, by making an extensive flashback to its history.

**Table of Contents**

|  |  |
| --- | --- |
| Acknowledgments | i |
| Abstract | ii |
| 1. **Introduction** | **1** |
| 1.1 Motivation and Overview ………………………………........ | 1 |
| 1.3 Objective .………………………………………………….... | 1 |
| 1. **Software Requirement Analysis** | **2** |
| 2.1 Define the problem ………………………………………………….. | 2 |
| 2.2 Define the modules and their functionalities (SRS) ……………..………………………………… | 2 |
| 1. **Software Design** | **3** |
| 3.1 ……………………….………………………… | 3 |
| 3.2 ……………………..…………………. | 4 |
| 1. **Testing** | **5** |
| 4.1 ………….……………………………………… | 5 |
| 1. **Implementation and User Interface** | **6** |
| 5.1 Data Cleaning And Formatting | 7 |
| 5.2 Exploratory Data Analysis | 12 |
| **6. Conclusion And References** | **14** |
| **7. Appendices** | **15** |

**INTRODUCTION**

**1.1 Motivation and Overview**

Prediction about the result of cricket matches astonishing as a research problem, especially due to its complexity, unpredictable assumption (weather and pitch conditions). Because the ultimate outcome of a cricket match is based on many aspect and unaccepted bearings therefore it is difficult responsibility to predict the exact and partial truth-based outcomes of cricket matches such a research expects a multi criteria decision making approach.

**1.2 Objective**

The main objective of the project is to present and analyze thoroughly some interesting historical stats about the World Cup, by making an extensive flashback to its history. The span of the flashback starts in 1975, the year that the first World Cup took place. The last part of the research presents some useful information about the current World Cup of 2019.

The first part is consisted of a thoroughly analysis about the best teams in the history of the Cup, simultaneously with a briefly introductory information about the pitches history. Additionally, the second part, contains some more complicated stats and analyses, such as some useful graphs about the World Cup players and the number of wins.

**Software Requirement Analysis**

**2.1 Define the problem**

The most common set of requirements defined by any operating system or software application is the physical computer resources, also known as hardware. A hardware requirements list is often accompanied by a hardware compatibility list (HCL), especially in case of operating systems. An HCL lists tested, compatibility and sometimes incompatible hardware devices for a particular operating system or application. The following sub-sections discuss the various aspects of hardware requirements.

**2.2 Define the modules and their functionalities (SRS)**

It is a technique that uses an array of static and interactive visuals within a specific context to help people understand and make sense of large amount of data. The data is often displayed in a format that visualizes patterns, trends and correlations. The images may include interactive capabilities, enabling users to manipulate them or drill into the data for querying and analysis.

It describes the effort to help people understand the significance of data by placing it in a visual context. There may be several languages on which we can perform the Data Visualization, but the ones that are much widely used in the field of Data Science are Python. Python is a general purpose programming language as well as for Data Visualization and Machine Learning.

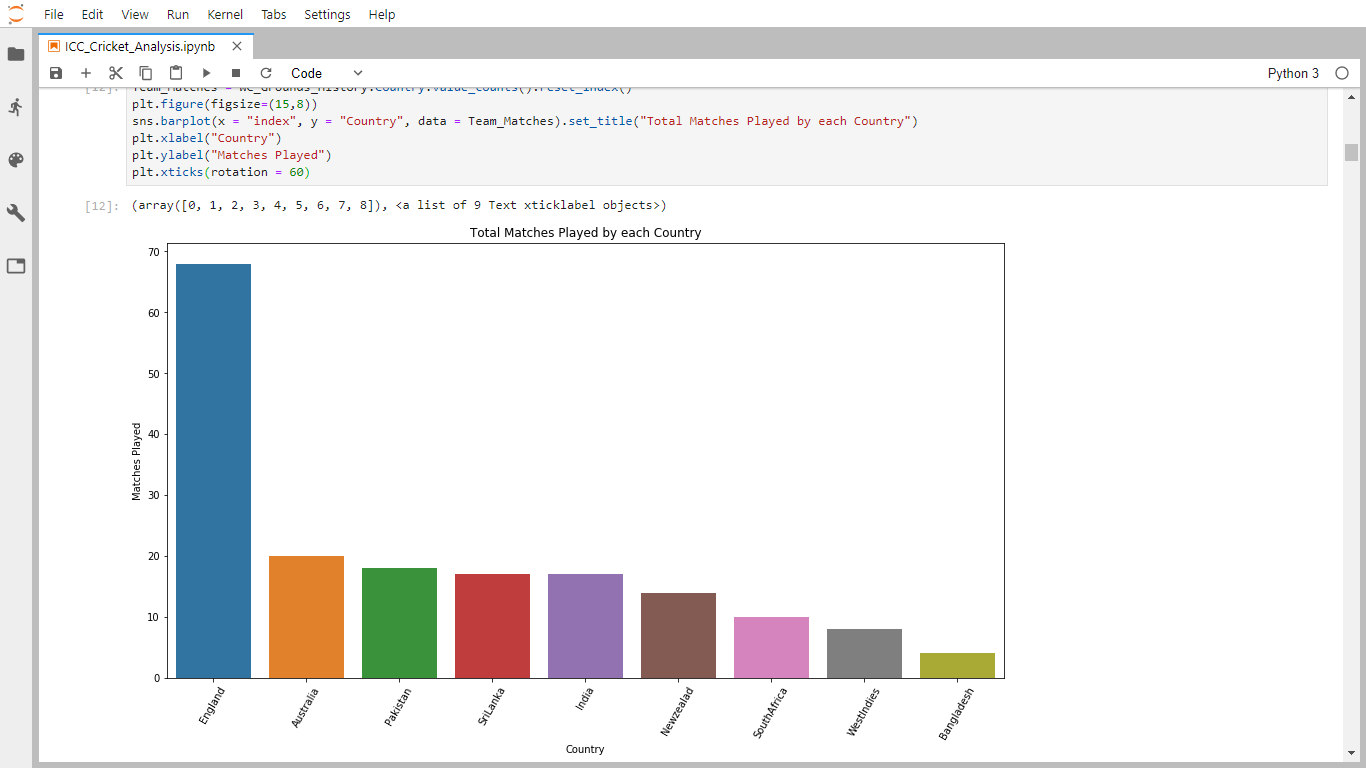
We havee compared support verctor machines, logistic regression , random forest and k nearest neighbours model.

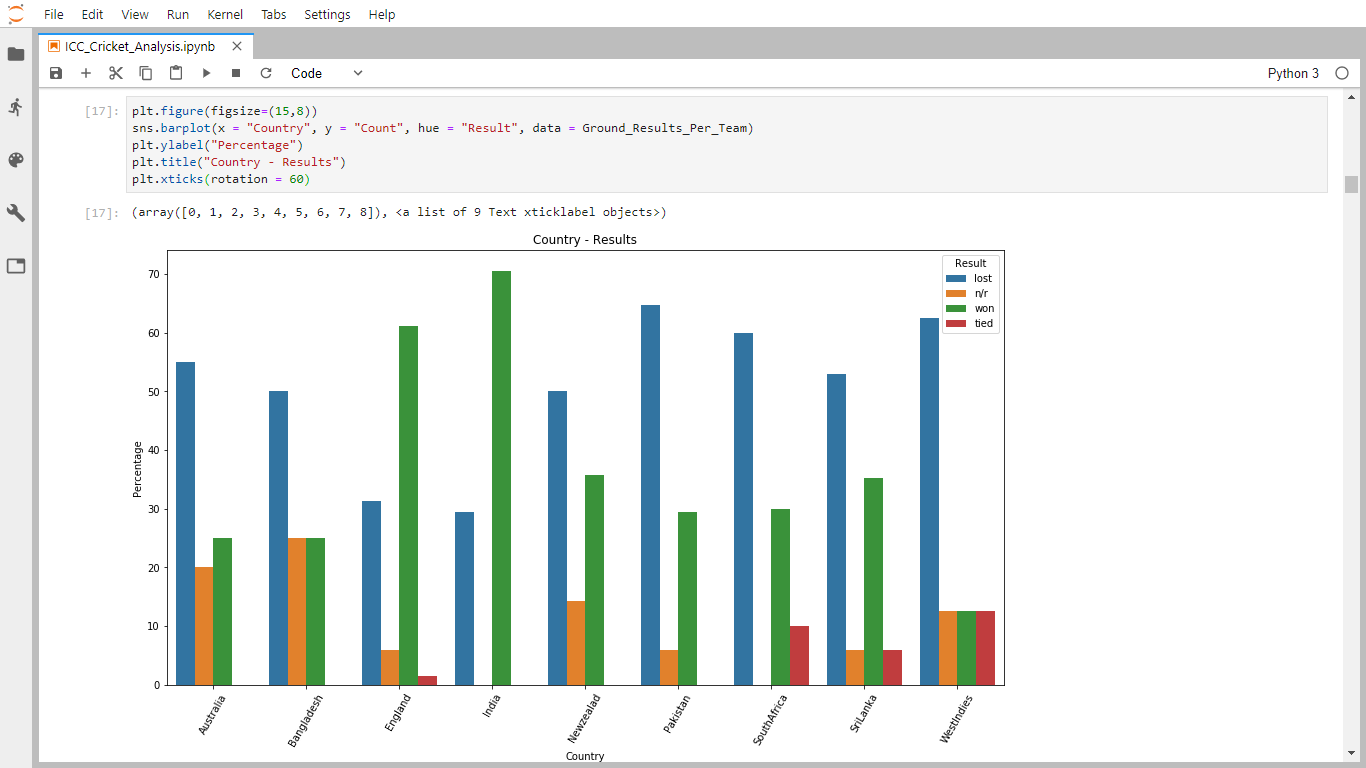
Random forest was the winner with training accuracy of 70% and test accuracy of 67.5%.

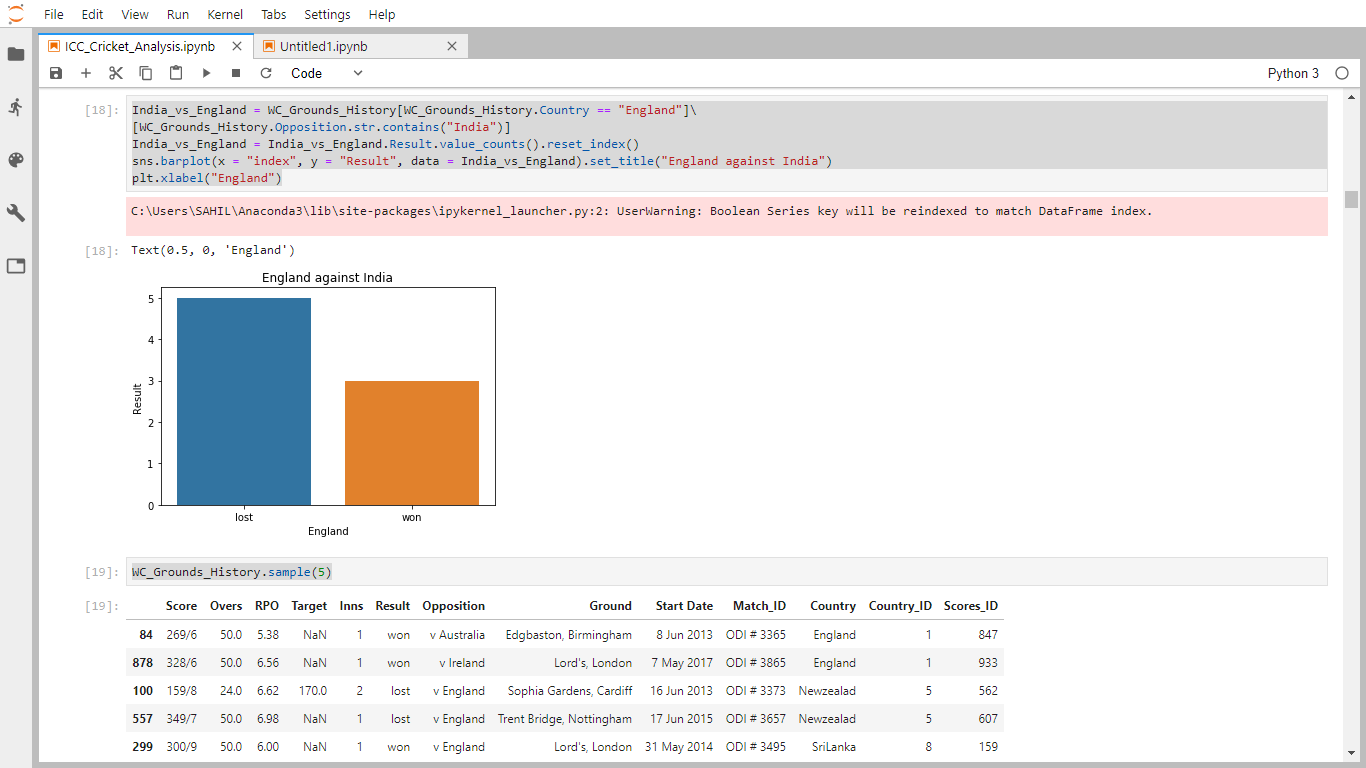
**Software Design**

System desing is the most creative phase of system development the term derscribes a final system and process by which it is developed.

3.1 Data Visualization And Analysis

****

****



**Testing**

As testing is the last phase before the final software is delivered, it has the enormous responsibility of detecting any type of error that may in the software. A software typically undergoes changes even after it has been delivered. And to validate that a change has not affected some old functionality of software regression testing is performed.

**4.1 Levels of Testing:**

The basic levels of testing are unit testing, integration testing and system and acceptance testing. These different levels of testing attempt to detect different types of faults.

Code testing and implementation is a critical process that can even consume more than sixty percent of the development time.

**Unit Testing:**

The system development life cycle involves the phases of testing and debugging after the requirement analysis, designing and coding. The project in question was tested , debugged and implemented successfully.

**Integration Testing:**

During the phase of unit testing different constituent modules were testing against the specifications produced during the design for the modules. Unit testing is essentially for the verification of the code produced during the coding the phase, and goal is to test the internal logic of the modules. The modules once tested were then considered for integration and use by others.

**System testing :**

It examines the specification stating about what program should do and how it performs under various conditions. This testing strategy is better strategy since it focuses on the way the software is expected to work

**Implementation and User Interface**

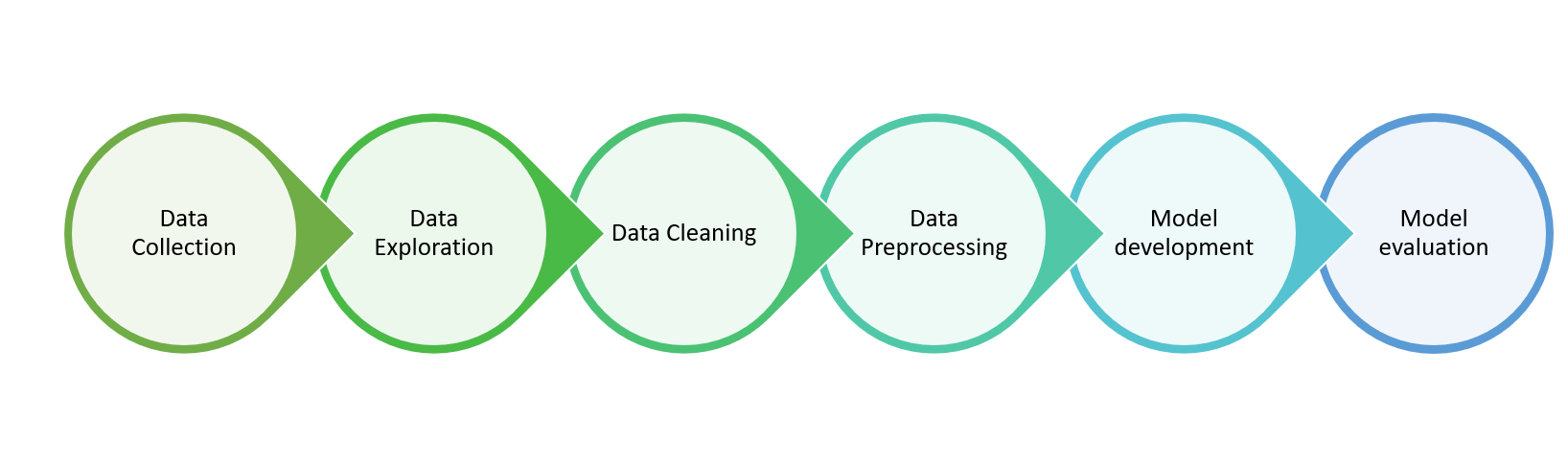
Implementation is the stage in the project where the theoretical design is turned into the working system and is giving confidence to the new system for the users i.e. will work efficiently and effectively.

Real world data is dirty. We can’t expect a nicely formatted and clean data as provided by Kaggle. Therefore, data pre-processing is so crucial that I can’t stress enough how important it is. It is the most important stage as it could occupy 40%-70% of the whole workflow, just to clean the data to be fed to your models.

We scraped three scripts from Crickbuzz website comprising of rankings of teams as of May 2019, details of the fixtures of 2019 world cup and details of each team’s history in previous world cups. I stored the above piece of data in three separate csv files. For the fourth file, I grabbed odi data-set for matches played between 1975 and 2017 from Kaggle in another csv file. In this file, I removed all the data from 1975 to 2010. This was done as the results of the last few years should only matter for our predictions. Since I didn’t get the data for 2018 and 2019 so this model might not be that accurate but still I believe this gives a fairly good idea. Then I did manual cleaning of the data as per my needs to make a machine learning model out of it.

We followed the general machine learning workflow step-by-step:

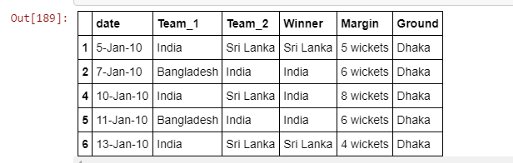
1. Data cleaning and formatting.
2. Exploratory data analysis.
3. Feature engineering and selection.
4. Compare several machine learning models on a performance metric.
5. Perform hyper-parameter tuning on the best model.
6. Evaluate the best model on the testing set.
7. Interpret the model results.
8. Draw conclusions and document work.



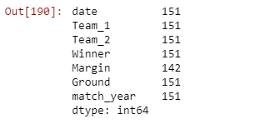
**Machine learning workflow**

# **5.1 Data cleaning and formatting:**

Next, let’s display the details of matches played by India.

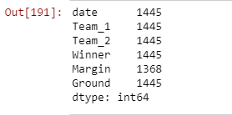


We continued by creating a column to display the details of matches played in 2010 and taking it as a reference for future work.



# **5.2 Exploratory data analysis:**

After that, we merged the details of the teams participating this year with their past results.



We deleted the columns like date of the match, margin of victory, and the ground on which the match was played. These features doesn’t look important for our prediction.

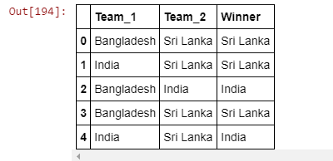
# **5.3 Feature engineering and selection**

This is probably the most important part in the machine learning workflow. Since the algorithm is totally dependent on how we feed data into it, feature engineering should be given topmost priority for every machine learning project.

**Reduces Overfitting**: Less redundant data means less opportunity to make decisions based on noise.

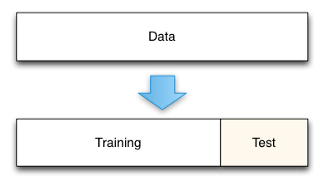
**Improves Accuracy**: Less misleading data means modeling accuracy improves.

Reduces Training Time: fewer data points reduce algorithm complexity and algorithms train faster.



Then we converted team-1 and team-2 from categorical variables to continuous inputs using pandas function *pd.get\_dummies.*This variable has only two answer choices: team 1 and team 2. It creates a new dataframe which consists of zeros and ones. The dataframe will have a one depending on the team of a particular game in this case.

Also, we separated training and test sets with 70% and 30% in training and validation sets respectively.



# **5.4 Compare several machine learning models on a performance metric**

I used Logistic Regression, Support Vector Machines, Random Forests and K Nearest Neighbours for training the model.

Random Forest outperformed all other algorithms with 70% training accuracy and 67.5% test accuracy.

The random forest combines hundreds or thousands of decision trees, trains each one on a slightly different set of the observations, splitting nodes in each tree considering a limited number of the features. The final predictions of the random forest are made by averaging the predictions of each individual tree.

# **5.5 Perform hyperparameter tuning on the best model**

Training set accuracy: 0.700  
Test set accuracy: 0.675

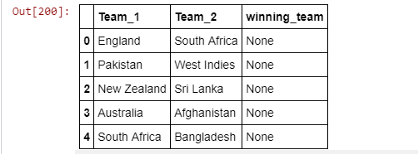
# **5.6 Evaluate the best model on the testing set:**

Let’s continue. We added ICC rankings of teams giving priority to higher ranked team to win this year.

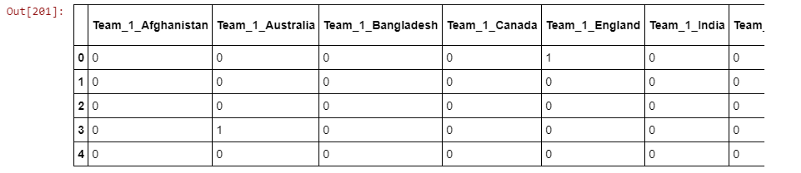
Next, We added new columns with ranking position for each team and slicing the dataset for first 45 games since there are 45 league stage games in total.



Then We added teams to new prediction dataset based on ranking position of each team.



After that, We added scripts for getting dummy variables and added missing columns compared to model training dataset.



# **5.7 Interpret the model results:**

So the four teams to march to the semi finals are New Zealand, India, England and South Africa.

And then we created a function to repeat the above work. This is the final function to predict the winner of ICC Cricket World Cup

New Zealand and India  
Winner: India

South Africa and England  
Winner: England

**5.8 Draw conclusions and document work**

# India and England

# Winner: England

**Conclusion**

Predictiveanalytics is a category of data analytics aimed at making predictions about future outcomes based on historical data and analytics techniques such as statistical modeling and machine learning. The science of predictiveanalytics can generate future insights with a significant degree of precision.Predictiveanalytics are used to determine customer responses or purchases, as well as promote cross-sell opportunities. Predictive models help businesses attract, retain and grow their most profitable customers. Improving operations. Many companies use predictive models to forecast inventory and manage resources. The benefits of video analysisinsports.is one of the obvious benefits of using video analysisinsports is having the ability to hone in and work on the points where players need correcting or need to improve.

# **REFRENCES**

* + - 1. <https://www.kaggle.com/>
      2. <https://www.researchgate.net/publication/331399453_Prediction_of_Cricket_World_Cup_2019_by_TOPSIS_Technique_of_MCDM-A_Mathematical_Analysis>
      3. <https://towardsdatascience.com/data-cleaning-with-python-and-pandas-detecting-missing-values-3e9c6ebcf78b>
      4. <https://www.invensis.net/blog/data-processing/5-steps-data-cleansing-customer-data/>
      5. <https://www.samford.edu/sports-analytics/fans/2018/Why-is-Data-Analytics-So-Important-in-Sports>

**Appendix**

#importing the packages

import os

import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

import matplotlib.ticker as ticker

import matplotlib.ticker as plticker

import warnings

warnings.filterwarnings('ignore')

from sklearn.model\_selection import train\_test\_split

from sklearn.ensemble import RandomForestClassifier

from sklearn.datasets import make\_classification

#loading the dataset

Batsman\_Data = pd.read\_csv('desktop/dataset/Batsman\_Data.csv')

Ground\_Data = pd.read\_csv('desktop/dataset/Ground\_Averages.csv')

ODI\_Scores\_Data = pd.read\_csv('desktop/dataset/ODI\_Match\_Totals.csv')

ODI\_Results\_Data = pd.read\_csv('desktop/dataset/ODI\_Match\_Results.csv')

WC\_Players\_Data = pd.read\_csv('desktop/dataset/WC\_players.csv')

Bowler\_Data = pd.read\_csv('desktop/dataset/Bowler\_data.csv')

world\_cup = pd.read\_csv('desktop/dataset/World Cup 2019 Dataset.csv')

results = pd.read\_csv('desktop/dataset/results.csv')

#seeing sample data of ground\_dataset since matches was occured in England

Ground\_Data.sample(6)

#seeing dataset

ODI\_Results\_Data.head()

ODI\_Scores\_Data.head()

#replacing the unamed:0 with Scores\_ID in ODI\_Scores\_Data

#Since we can understand that ODI\_Scores\_Data gives much information compared to other file

ODI\_Scores\_Data["Scores\_ID"] = ODI\_Scores\_Data["Unnamed: 0"]

ODI\_Scores\_Data.drop(columns="Unnamed: 0",inplace=True)

ODI\_Scores\_Data.head()

#Let's analyze the England pitches and conditions and results over their grounds.

WC\_venue\_pitches = ["The Oval, London","Trent Bridge, Nottingham",

"Sophia Gardens, Cardiff","County Ground, Bristol",

"Rose Bowl, Southampton","County Ground, Taunton",

"Old Trafford, Manchester","Edgbaston, Birmingham",

"Headingley, Leeds","Lord's, London","Riverside Ground","Chester-le-Street"]

#Total Grounds

WC\_Ground\_Stats = []

ODI\_Grounds = ODI\_Scores\_Data.Ground

for i in ODI\_Grounds:

for j in WC\_venue\_pitches:

if i in j:

#print("i ; ",i,"--j : ",j)

WC\_Ground\_Stats.append((i,j))

Ground\_names = dict(set(WC\_Ground\_Stats))

def Full\_Ground\_names(value):

return Ground\_names[value]

Ground\_names

#Seems like there's no Data regardign ODI Matches in this Taunton Ground.

#So we can't analyze the stats of 6. County Ground, Taunton .

#Let's gather the data of all ODI's in these WC Venues

WC\_Grounds\_History = ODI\_Scores\_Data[ODI\_Scores\_Data.Ground.isin([Ground[0] for Ground in WC\_Ground\_Stats])]

WC\_Grounds\_History["Ground"] = WC\_Grounds\_History.Ground.apply(Full\_Ground\_names)

WC\_Grounds\_History.head()

#So, now we have the data of matches that were played in WC venues.

#How many WC teams have played in these venues before and what are they?

Team\_Matches = WC\_Grounds\_History.Country.value\_counts().reset\_index()

plt.figure(figsize=(15,8))

sns.barplot(x = "index", y = "Country", data = Team\_Matches).set\_title("Total Matches Played by each Country")

plt.xlabel("Country")

plt.ylabel("Matches Played")

plt.xticks(rotation = 60)

#Team wise Winning Percentage in England Pitches

WC\_Grounds\_History.sample(5)

WC\_Grounds\_History.Result.value\_counts()

#seems like some data is missed, So I will remove Data with Result "-"

WC\_Grounds\_History = WC\_Grounds\_History[~WC\_Grounds\_History.Result.isin(["-"])]

WC\_Grounds\_History.Result.value\_counts()

winnings = WC\_Grounds\_History[["Country","Result"]]

winnings["count"] = 1

Ground\_Results\_Per\_Team = winnings.groupby(["Country","Result"]).aggregate(["sum"])

Ground\_Results\_Per\_Team = Ground\_Results\_Per\_Team.groupby(level=0).apply(lambda x:100 \* x / float(x.sum())).reset\_index()

Ground\_Results\_Per\_Team.columns = ["Country","Result","Count"]

Ground\_Results\_Per\_Team.head()

plt.figure(figsize=(15,8))

sns.barplot(x = "Country", y = "Count", hue = "Result", data = Ground\_Results\_Per\_Team)

plt.ylabel("Percentage")

plt.title("Country - Results")

plt.xticks(rotation = 60)

#Let's see what happens when the Top Two Teams face?

India\_vs\_England = WC\_Grounds\_History[WC\_Grounds\_History.Country == "England"]\

[WC\_Grounds\_History.Opposition.str.contains("India")]

India\_vs\_England = India\_vs\_England.Result.value\_counts().reset\_index()

sns.barplot(x = "index", y = "Result", data = India\_vs\_England).set\_title("England against India")

plt.xlabel("England")

#Let's analyze the Win and Lose Percentage based on Innings wrt Stadium

WC\_Grounds\_History.sample(5)

Inning\_Wins = WC\_Grounds\_History[WC\_Grounds\_History.Result == "won"].Inns.value\_counts(normalize = True).reset\_index()

sns.barplot(x = "index", y = "Inns", data = Inning\_Wins).set\_title("Winnings by Innigs")

plt.xlabel("Innings")

plt.ylabel("Winning Percentage")#Those Teams which Bowl First has won around 55% of the matches

#Let's Divide them by Pitches and see the history

Pitch\_Innings = WC\_Grounds\_History[WC\_Grounds\_History.Result == "won"][["Inns","Ground"]]

Pitch\_Innings["Count"] = 1

Pitch\_Innings = Pitch\_Innings.groupby(["Ground","Inns"]).sum()

Pitch\_Innings = Pitch\_Innings.groupby(level=0).apply(lambda x:100 \* x / float(x.sum())).reset\_index()

Pitch\_Innings.columns = ["Ground", "Inns","Wins"]

Pitch\_Innings.head( 5 )

plt.figure(figsize=(15,8))

sns.barplot(x = "Ground", y = "Wins", hue = "Inns", data = Pitch\_Innings).set\_title("Innings vs Winnings")

plt.xticks(rotation = 60)#Teams that Bat second will Win most of the times.

#HERE COME THE PREDICTION PART

world\_cup.head()

results.head()

df = results[(results['Team\_1'] == 'India') | (results['Team\_2'] == 'India')]

india = df.iloc[:]

india.head()

#creating a column for matches played in 2010

year = []

for row in india['date']:

year.append(int(row[7:]))

india ['match\_year']= year

india\_2010 = india[india.match\_year >= 10]

india\_2010.count()

#narrowing to team patcipating in the world cup

worldcup\_teams = ['England', ' South Africa', '', 'West Indies',

'Pakistan', 'New Zealand', 'Sri Lanka', 'Afghanistan',

'Australia', 'Bangladesh', 'India']

df\_teams\_1 = results[results['Team\_1'].isin(worldcup\_teams)]

df\_teams\_2 = results[results['Team\_2'].isin(worldcup\_teams)]

df\_teams = pd.concat((df\_teams\_1, df\_teams\_2))

df\_teams.drop\_duplicates()

df\_teams.count()

df\_teams.head()

#dropping columns that wll not affect match outcomes

df\_teams\_2010 = df\_teams.drop(['date','Margin', 'Ground'], axis=1)

df\_teams\_2010.head()

#Building the model

#the prediction label: The winning\_team column will show "1" Team 1 has won and "2" if the away team has won.

df\_teams\_2010 = df\_teams\_2010.reset\_index(drop=True)

df\_teams\_2010.loc[df\_teams\_2010.Winner == df\_teams\_2010.Team\_1,'winning\_team']=1

df\_teams\_2010.loc[df\_teams\_2010.Winner == df\_teams\_2010.Team\_2, 'winning\_team']=2

df\_teams\_2010 = df\_teams\_2010.drop(['winning\_team'], axis=1)

df\_teams\_2010.head()

#convert team-1 and team-2 from categorical variables to continous inputs

# Get dummy variables

final = pd.get\_dummies(df\_teams\_2010, prefix=['Team\_1', 'Team\_2'], columns=['Team\_1', 'Team\_2'])

# Separate X and y sets

X = final.drop(['Winner'], axis=1)

y = final["Winner"]

# Separate train and test sets

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.30, random\_state=42)

final.head()

rf = RandomForestClassifier(n\_estimators=100, max\_depth=20,random\_state=0)

rf.fit(X\_train, y\_train)

score = rf.score(X\_train, y\_train)

score2 = rf.score(X\_test, y\_test)

print("Training set accuracy: ", '%.3f'%(score))

print("Test set accuracy: ", '%.3f'%(score2))

#adding ICC rankings

#the team which is positioned higher on the ICC Ranking will be considered "favourite" for the match

#and therefore, will be positioned under the "Team\_1" column

# Loading new datasets

ranking = pd.read\_csv('desktop/dataset/icc\_rankings.csv')

fixtures = pd.read\_csv('desktop/dataset/fixtures.csv')

# List for storing the group stage games

pred\_set = []

# Create new columns with ranking position of each team

fixtures.insert(1, 'first\_position', fixtures['Team\_1'].map(ranking.set\_index('Team')['Position']))

fixtures.insert(2, 'second\_position', fixtures['Team\_2'].map(ranking.set\_index('Team')['Position']))

# We only need the group stage games, so we have to slice the dataset

fixtures = fixtures.iloc[:45, :]

fixtures.tail()

for index, row in fixtures.iterrows():

if row['first\_position'] < row['second\_position']:

pred\_set.append({'Team\_1': row['Team\_1'], 'Team\_2': row['Team\_2'], 'winning\_team': None})

else:

pred\_set.append({'Team\_1': row['Team\_2'], 'Team\_2': row['Team\_1'], 'winning\_team': None})

pred\_set = pd.DataFrame(pred\_set)

backup\_pred\_set = pred\_set

pred\_set.head()

# Get dummy variables and drop winning\_team column

pred\_set = pd.get\_dummies(pred\_set, prefix=['Team\_1', 'Team\_2'], columns=['Team\_1', 'Team\_2'])

# Add missing columns compared to the model's training dataset

missing\_cols = set(final.columns) - set(pred\_set.columns)

for c in missing\_cols:

pred\_set[c] = 0

pred\_set = pred\_set[final.columns]

pred\_set = pred\_set.drop(['Winner'], axis=1)

pred\_set.head()

#group matches

predictions = rf.predict(pred\_set)

for i in range(fixtures.shape[0]):

print(backup\_pred\_set.iloc[i, 1] + " and " + backup\_pred\_set.iloc[i, 0])

if predictions[i] == 1:

print("Winner: " + backup\_pred\_set.iloc[i, 1])

else:

print("Winner: " + backup\_pred\_set.iloc[i, 0])

print("")

# List of tuples before

semi = [('New Zealand', 'India'),

('England', 'South Africa')]

def clean\_and\_predict(matches, ranking, final, logreg):

# Initialization of auxiliary list for data cleaning

positions = []

# Loop to retrieve each team's position according to ICC ranking

for match in matches:

positions.append(ranking.loc[ranking['Team'] == match[0],'Position'].iloc[0])

positions.append(ranking.loc[ranking['Team'] == match[1],'Position'].iloc[0])

# Creating the DataFrame for prediction

pred\_set = []

# Initializing iterators for while loop

i = 0

j = 0

# 'i' will be the iterator for the 'positions' list, and 'j' for the list of matches (list of tuples)

while i < len(positions):

dict1 = {}

# If position of first team is better then this team will be the 'Team\_1' team, and vice-versa

if positions[i] < positions[i + 1]:

dict1.update({'Team\_1': matches[j][0], 'Team\_2': matches[j][1]})

else:

dict1.update({'Team\_1': matches[j][1], 'Team\_2': matches[j][0]})

# Append updated dictionary to the list, that will later be converted into a DataFrame

pred\_set.append(dict1)

i += 2

j += 1

# Convert list into DataFrame

pred\_set = pd.DataFrame(pred\_set)

backup\_pred\_set = pred\_set

# Get dummy variables and drop winning\_team column

pred\_set = pd.get\_dummies(pred\_set, prefix=['Team\_1', 'Team\_2'], columns=['Team\_1', 'Team\_2'])

# Add missing columns compared to the model's training dataset

missing\_cols2 = set(final.columns) - set(pred\_set.columns)

for c in missing\_cols2:

pred\_set[c] = 0

pred\_set = pred\_set[final.columns]

pred\_set = pred\_set.drop(['Winner'], axis=1)\

# Predict!

predictions = logreg.predict(pred\_set)

for i in range(len(pred\_set)):

print(backup\_pred\_set.iloc[i, 1] + " and " + backup\_pred\_set.iloc[i, 0])

if predictions[i] == 1:

print("Winner: " + backup\_pred\_set.iloc[i, 1])

else:

print("Winner: " + backup\_pred\_set.iloc[i, 0])

print("")

clean\_and\_predict(semi, ranking, final, rf)

# Finals

finals = [('India', 'England')]

clean\_and\_predict(finals, ranking, final, rf)