T20 Match Predictor

A Machine Learning Approach to Match Outcome Prediction

Tasmia Azrine

BSc. in CSE

Brac University

Email: tasmiaazrin@gmail.com

POC	🔗 GitHub Repository
Dataset	🔗 Kaggle

Abstract

This project is about trying to predict the results of the T20 International cricket matches with the help of machine learning. This dataset consists of the matches supported by 2005-2023, with plenty of the necessary attributes like the team name, venue, toss details, and match results. Following the preprocessing and exploratory data analysis, several classification models were fit, such as Logistic Regression, Decision Tree, Random Forest, K-Nearest Neighbours, and Gradient Boosting. Confusion matrix and accuracy are used as it determines model performance. Logistic Regression performed the best, and it proved that it is effective in analysing complicated patterns when dealing with sports data. The project emphasises the possibility of data-assisted methods in sports analytics.

Table of Contents

1. Introduction

The Twenty20 (T20) cricket has became most powerful game in recent years because of its unpredictable and fast- paced nature. Machine learning (ML) is a huge tool in the analysis and prediction of match outcomes given the availability of match-historical data in an increasing extent. Better forecasts can be advantageous to fans, broadcasters, analysts, and participants of fantasy leagues, as proper prediction will allow them to gain some insights about the dynamics of the corresponding matches based on some facts.

The goal of this project is to develop a machine learning model that could forecast the outcome of an international T20 game based on the past statistics. This is by the analysis of factors as match venue, station of toss, team compositions, with an aim of discovering trends which affect the probability of a team winning.

2. Dataset Description

This project uses the publicly available dataset titled **"All T20 Internationals Dataset (2005–2023)"**, which was sourced from Kaggle:
https://www.kaggle.com/datasets/bhuvaneshprasad/all-t20-internationals-dataset-2005-to-2023

It contains comprehensive storted data on **T20 International (T20I) cricket matches** from 2005 to 2023, including both men's and women's games. **Duplicate or incomplete records were dropped** to ensure the reliability of training and evaluation.

2.1 Dataset Overview

- **File Used: "t20i_Matches_Data.csv"**
- **Total Columns (Initially):** 33
- **Total Rows (Initially):** 2592
- **Match Types:** International T20 Matches
- **Time:** From February 2008 to July 2024
- **Data Format:** Tabular. Each row represents a single T20 match

2.2 Attributes Available in the Dataset

The dataset includes the following types of features:

- **Match details:** Match Date, Match Id, Match Format
- **Team details:** Team1 Name, Team2 Name, Match Winner
- **Toss information:** Toss Winner, Toss Winner Choice
- **Score details:** Team1 Runs Scored, Team2 Runs Scored, Team1 Wickets, Team2 Wickets
- **Venue details:** Match Venue(Stadium), Match Venue (City), Match Venue (Country)
- **Additional metadata:** Umpires, Series Name, Man of the Match, Match Result, etc.

3. Data Cleaning

The raw data were subjected to a set of cleaning processes to make them consistent, complete, and relevant before the performance of machine learning processes. As the data had a long range (2005-2023) and different types of matches, missing data, irrelevant data, and noisy data were to be filled in and dropped.

3.1 Steps Performed:

3.1.1 Dropped Unnecessary Columns

Multiple columns in the initial data, like names of players, names of umpires, form of the match, and the city where it was played, did not help predict the outcome, and hence, these were dropped to make the data less complex. These included:

- Umpire1, Umpire2, Player of the Match
- Match Venue (City), Match Venue(Stadium), Match Number, Match Format
- Series Name, Team1 Captain, Team2 Captain, Team Wicket and others

The focus remained on team-level statistics and match-level decisions only.

3.1.2. Removed Matches with Missing Data

Rows with **missing or null values** in columns were dropped. These included:

- Missing Match Winner, Toss Winner, and Toss Winner Choice

This step helped ensure that the machine learning model only learned from **complete and valid** data.

3.2 Final Cleaned Dataset:

- **Total after Cleaning:** 2493 rows x 9 columns
- **Final Columns Used for Modelling:** Match Date, Team1 Name, Team2 Name, Toss Winner, Toss Winner Choice, Match Venue (Country), Match Winner, Team1 Runs Scored, Team2 Runs Scored
- **Target Variable:** Match Winner

The dataset after cleaning was further used in the exploration of data, feature engineering, and training of models.

4. Exploratory Data Analysis (EDA)

EDA intended to interpret the structure and patterns in the participating in the T20I match dataset, discover the influential variables and provide a way forward to generate meaningful features to train the models. The visual analysis and statistical analysis were conducted with libraries: **pandas, matplotlib, and seaborn**.

4.1 Wins and Matches Played by Each Team

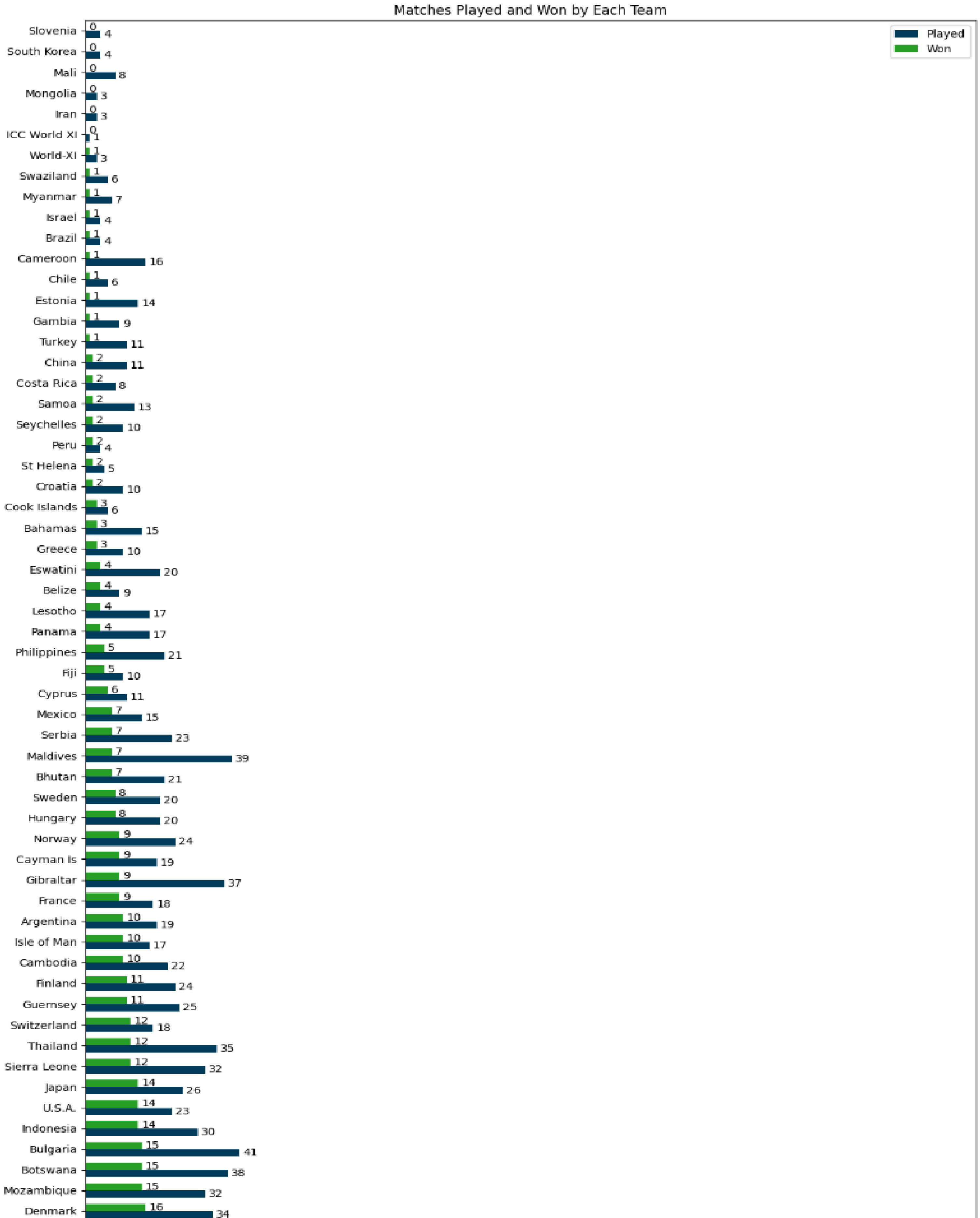
Using bar charts, I explored how many matches each team played and how many matches they won.

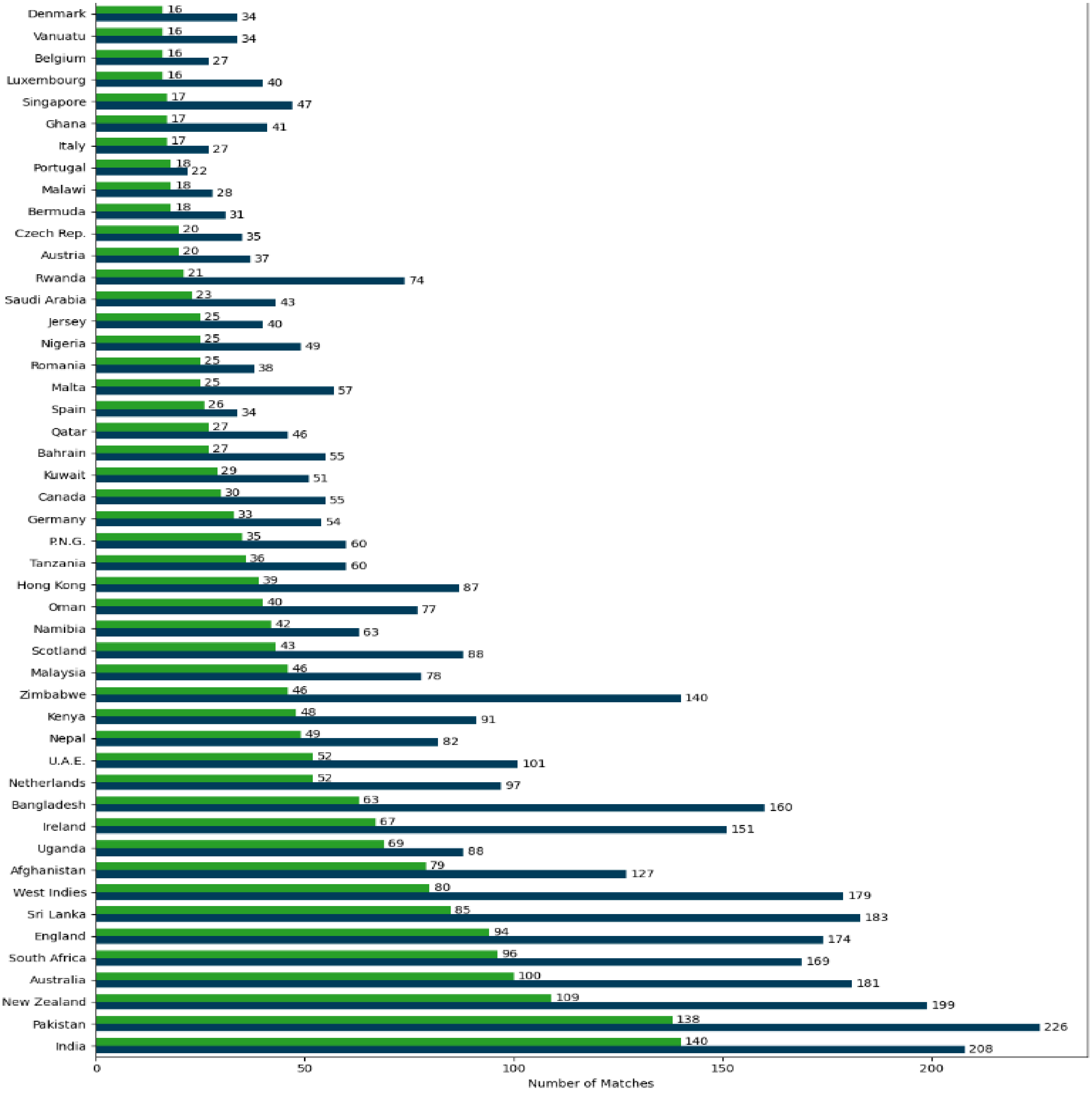
- **Top 5 teams by matches played.**
  - Pakistan (226 matches)
  - India (208 matches)
  - New Zealand (119 matches)
  - Sri Lanka (183 matches)
  - Australia (181 matches)

- **Top 5 teams by match wins.**
  - India (140 wins)
  - Pakistan (138 wins)
  - New Zealand (109 wins)
  - Australia (112 wins)
  - South Africa (100 wins)

Historically, these teams are dominants in T20I cricket and create the basis of match activity within the set of data. Most of the low ranked teams such as Nepal, Uganda and China played less than 20 matches.

These figures were used in the later analysis and also in reducing the confusion matrices to the best teams alone.


Matches Played and Won by Each Team
Played
Won
Slovenia 0 4
South Korea 0 4
Mali 0 8
Mongolia 0 3
Iran 0 3
ICC World XI 0 1
World-XI 1 3
Swaziland 1 6
Myanmar 1 7
Israel 1 4
Brazil 1 4
Cameroon 1 16
Chile 1 6
Estonia 1 14
Gambia 1 9
Turkey 1 11
China 2 11
Costa Rica 2 8
Samoa 2 13
Seychelles 2 10
Peru 2 4
St Helena 2 5
Croatia 2 10
Cook Islands 3 6
Bahamas 3 15
Greece 3 10
Eswatini 4 20
Belize 4 9
Lesotho 4 17
Panama 4 17
Philippines 5 21
Fiji 5 10
Cyprus 6 11
Mexico 7 15
Serbia 7 23
Maldives 7 39
Bhutan 7 21
Sweden 8 20
Hungary 8 20
Norway 9 24
Cayman Is 9 19
Gibraltar 9 37
France 9 18
Argentina 10 19
Isle of Man 10 17
Cambodia 10 22
Finland 11 24
Guernsey 11 25
Switzerland 12 18
Thailand 12 35
Sierra Leone 12 32
Japan 14 26
U.S.A. 14 23
Indonesia 14 30
Bulgaria 15 41
Botswana 15 38
Mozambique 15 32
Denmark 16 34


Denmark 16 34
Vanuatu 16 34
Belgium 16 27
Luxembourg 16 40
Singapore 17 47
Ghana 17 41
Italy 17 27
Portugal 18 22
Malawi 18 28
Bermuda 18 31
Czech Rep. 20 35
Austria 20 37
Rwanda 21 74
Saudi Arabia 23 43
Jersey 25 40
Nigeria 25 49
Romania 25 38
Malta 25 57
Spain 26 34
Qatar 27 46
Bahrain 27 55
Kuwait 29 51
Canada 30 55
Germany 33 54
P.N.G. 35 60
Tanzania 36 60
Hong Kong 39 87
Oman 40 77
Namibia 42 63
Scotland 43 88
Malaysia 46 78
Zimbabwe 46 140
Kenya 48 91
Nepal 49 82
U.A.E. 52 101
Netherlands 52 97
Bangladesh 63 160
Ireland 67 151
Uganda 69 88
Afghanistan 79 127
West Indies 80 179
Sri Lanka 85 183
England 94 174
South Africa 96 169
Australia 100 181
New Zealand 109 199
Pakistan 138 226
India 140 208
0
50
100
150
200
Number of Matches

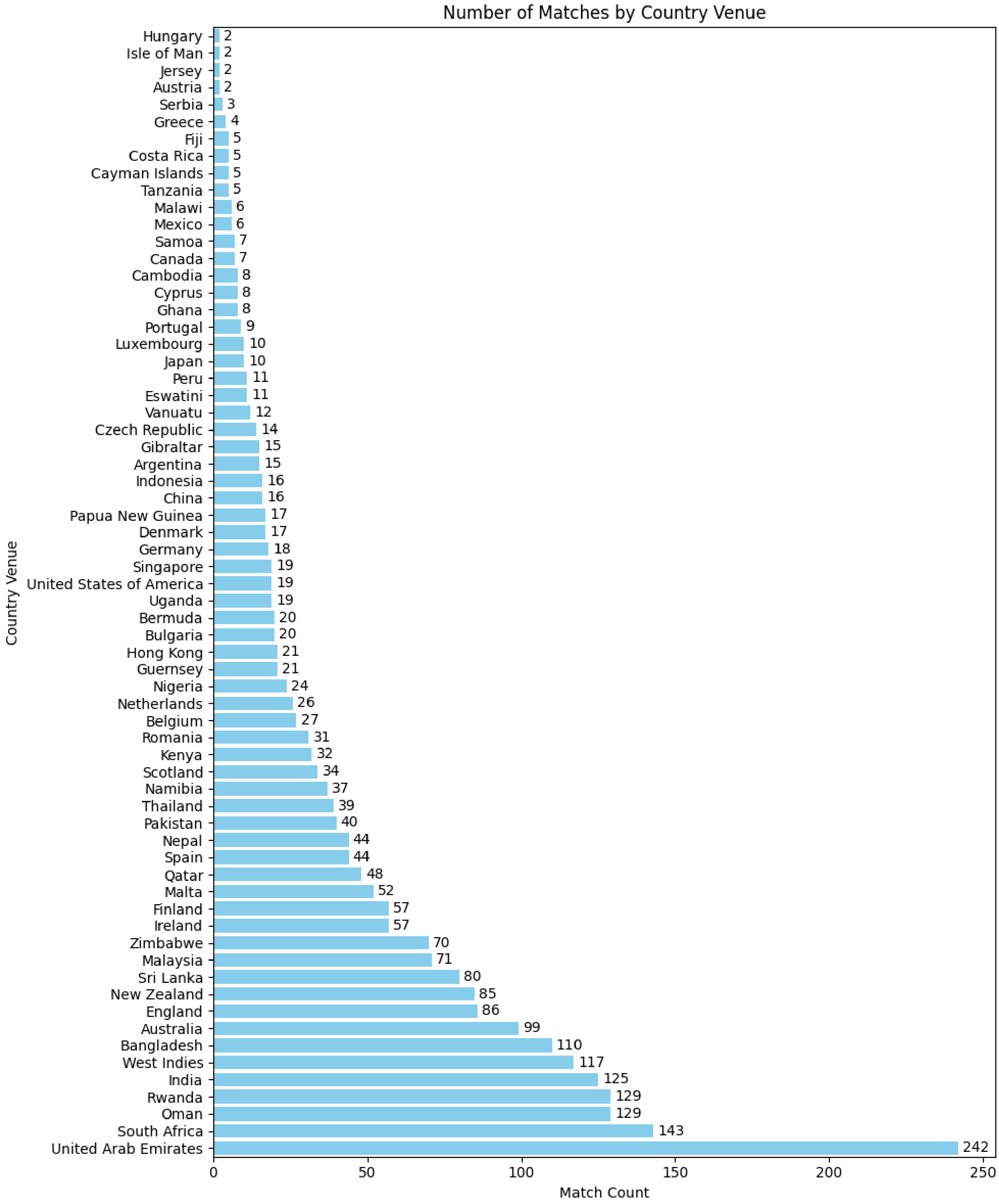
4.2 Number of Matches Played by Match Venue (Country)

To identify home and neutral match patterns, I examined where matches were hosted.

- **Top 5 hosting countries**:
-
    - United Arab Emirates (UAE): 242 matches
    - South Africa: 143 matches
    - Oman and Rwanda: 129 matches
    - India: 125 matches

Some nations such as UAE held a lot of neutral venues games, whereas others were mainly playing at home.

This was deployed in the process of making the feature of Home Advantage.


Number of Matches by Country Venue
Country Venue
Match Count
Hungary 2
Isle of Man 2
Jersey 2
Austria 2
Serbia 3
Greece 4
Fiji 5
Costa Rica 5
Cayman Islands 5
Tanzania 5
Malawi 6
Mexico 6
Samoa 7
Canada 7
Cambodia 8
Cyprus 8
Ghana 8
Portugal 9
Luxembourg 10
Japan 10
Peru 11
Eswatini 11
Vanuatu 12
Czech Republic 14
Gibraltar 15
Argentina 15
Indonesia 16
China 16
Papua New Guinea 17
Denmark 17
Germany 18
Singapore 19
United States of America 19
Uganda 19
Bermuda 20
Bulgaria 20
Hong Kong 21
Guernsey 21
Nigeria 24
Netherlands 26
Belgium 27
Romania 31
Kenya 32
Scotland 34
Namibia 37
Thailand 39
Pakistan 40
Nepal 44
Spain 44
Qatar 48
Malta 52
Finland 57
Ireland 57
Zimbabwe 70
Malaysia 71
Sri Lanka 80
New Zealand 85
England 86
Australia 99
Bangladesh 110
West Indies 117
India 125
Rwanda 129
Oman 129
South Africa 143
United Arab Emirates 242
0
50
100
150
200
250

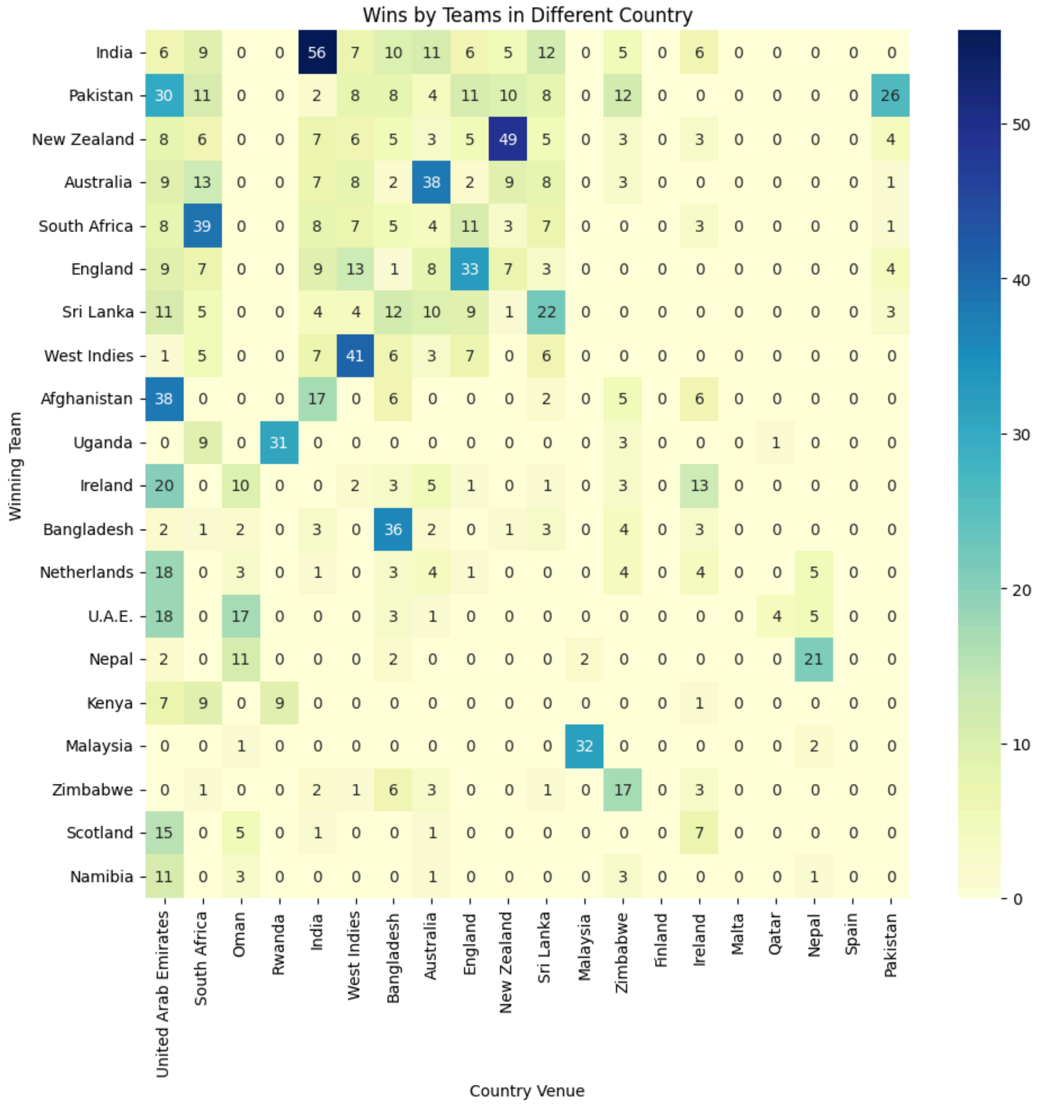
4.3. Team Wins by Venue Country – Heatmap

I built a matrix (team × country venue) showing how many matches each team won in each country.

Example:

- **India** won 56 matches in India and 53 in other countries
- **Pakistan** won 30 matches in the UA, more than in Pakistan **and** won most of their matches at home and in neutral countries like the UAE and South Africa

This clearly showed home ground effects and neutral venue strengths, validating the need for a based feature.

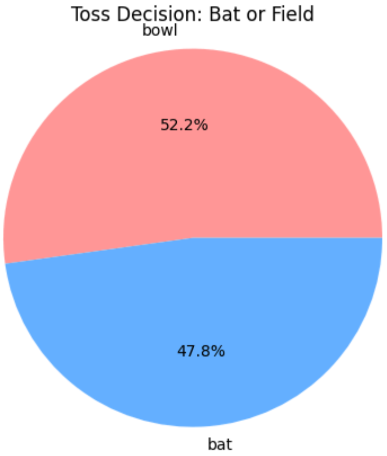


	United Arab Emirates	South Africa	Oman	Rwanda	India	West Indies	Bangladesh	Australia	England	New Zealand	Sri Lanka	Malaysia	Zimbabwe	Finland	Ireland	Malta	Qatar	Nepal	Spain	Pakistan
India	6	9	0	0	56	7	10	11	6	5	12	0	5	0	6	0	0	0	0	0
Pakistan	30	11	0	0	2	8	8	4	11	10	8	0	12	0	0	0	0	0	0	26
New Zealand	8	6	0	0	7	6	5	3	5	49	5	0	3	0	3	0	0	0	0	4
Australia	9	13	0	0	7	8	2	38	2	9	8	0	3	0	0	0	0	0	0	1
South Africa	8	39	0	0	8	7	5	4	11	3	7	0	0	0	3	0	0	0	0	1
England	9	7	0	0	9	13	1	8	33	7	3	0	0	0	0	0	0	0	0	4
Sri Lanka	11	5	0	0	4	4	12	10	9	1	22	0	0	0	0	0	0	0	0	3
West Indies	1	5	0	0	7	41	6	3	7	0	6	0	0	0	0	0	0	0	0	0
Afghanistan	38	0	0	0	17	0	6	0	0	0	2	0	5	0	6	0	0	0	0	0
Uganda	0	9	0	31	0	0	0	0	0	0	0	0	3	0	0	0	1	0	0	0
Ireland	20	0	10	0	0	2	3	5	1	0	1	0	3	0	13	0	0	0	0	0
Bangladesh	2	1	2	0	3	0	36	2	0	1	3	0	4	0	3	0	0	0	0	0
Netherlands	18	0	3	0	1	0	3	4	1	0	0	0	4	0	4	0	0	5	0	0
U.A.E.	18	0	17	0	0	0	3	1	0	0	0	0	0	0	0	0	4	5	0	0
Nepal	2	0	11	0	0	0	2	0	0	0	0	2	0	0	0	0	0	21	0	0
Kenya	7	9	0	9	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
Malaysia	0	0	1	0	0	0	0	0	0	0	0	32	0	0	0	0	0	2	0	0
Zimbabwe	0	1	0	0	2	1	6	3	0	0	1	0	17	0	3	0	0	0	0	0
Scotland	15	0	5	0	1	0	0	1	0	0	0	0	0	0	7	0	0	0	0	0
Namibia	11	0	3	0	0	0	0	1	0	0	0	0	3	0	0	0	0	1	0	0

4.4 Toss Choice Distribution

A pie chart was generated to analyse what teams usually chose after winning the toss.

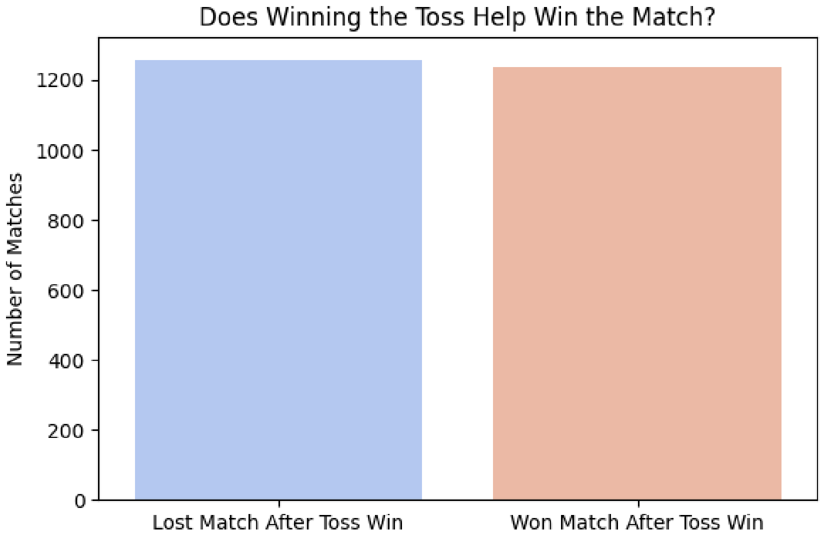
- **52.4%** of the time, teams chose to **bowl first**
- **47.6%** chose to **bat first**



Teams slightly prefer chasing, which aligns with modern T20 strategy trends. Toss choice was considered a potentially useful predictive feature.

4.5 Does Winning the Toss Affect Match Result?

To explore whether winning the toss influences the match outcome, a bar chart was compared.



There is a small advantage for toss winners, but it's not overwhelming. Still, it justified the inclusion of a **"Toss Advantage"** feature.

4.6 Key Observations Summary from EDA

Feature Explored	**Key Finding**
Team Wins	India & Pakistan dominate
Team Wins by Venue	Teams show strong home/neutral patterns
Toss Choice	Bowling slightly preferred
Low Participation Teams	Some teams appeared in very few matches

5. Feature Engineering and Pre-processing

In machine learning, feature engineering is a significant aspect since it helps convert raw data to meaningful inputs that can improve the performance of a model. In this case of predicting this T20 game, a few measures have been undertaken towards building and developing features in effective manners that encompass a historical significance and the elimination of data leakage.

5.1 Handling Categorical Variables

The dataset contained multiple categorical columns such as team names, venue country, and toss winner, toss winner choice. These were handled in two steps:

5.1.1 Label Encoding

- Converted categorical strings (team names, match winner and toss winner choice) into numeric codes for processing in alphabetical order. For example :

Afghanistan→0, Argentina→1, Australia→2 -------------- Zimbabwe → 109

bat → 0 bowl → 1

5.1.2 One-Hot Encoding

- Applied to categorical features
  - Team1 Name
  - Team2 Name
  - Toss Winner Choice

One-hot encoding makes sure that models will not assume the ordinal relationships among teams or countries. Random Forest and Gradient Boosting have tree-based models that perform well when dealing with non-ordinal categories using one-hot encoding.

5.2 Home Advantage Feature

Created a custom feature to capture whether a team was playing on its home ground based on Team1 Name, Team2 Name and Match Venue(Country) column.

Team 1 is playing in its home country	+1
Team 2 is playing in its home country	-1
Neutral venue (not home for either)	0

Teams often perform better in home conditions due to crowd support, familiarity with the pitch, and climate.

5.3 Toss Advantage Feature

Another engineered feature was based on which team won the toss:

Toss Winner	**Value**
Team1 Name	+1
Team2 Name	-1

Toss can influence strategy (batting/bowling first), and the EDA showed a slight advantage to toss winners.

5.4 Historical Team Performance: Average Team Score

To represent batting strength, two features were computed

- **Team1_Avg_Score**
- **Team2_Avg_Score**

For Training Data, an expanding mean was used for each team's past match scores. It prevented data leakage by separating current and future match data. For Testing Data, average scores calculated from the training set were used. In testing the score filled with an overall average score. Although technically correct, models like Logistic Regression and Gradient Boosting **performed better without** these features. It is probably due to increased variance or confusion with unseen teams. So, in the final training, these two features were **excluded** later on.

5.5 Final Feature Selection

- **Dropped Columns Before Training**:
  - Irrelevant Columns: Match Date, Team1 Runs Scored, Team2 Runs Scored, Toss Winner, Match Venue (Country)
- **Final feature sets:**
  - One-hot encoded team names and toss choice info
  - Home Advantage
  - Toss Advantage
- **Target Variable**: Match Winner

5.6 Train-Test Split

Used 80% for training, 20% for testing

6. Model Selection Criteria

The choice of the models of machine learning to be used in this project was dictated by the problem at hand (multi-class classification) and the feature types categorical variables (team names, toss winner choise) and engineered features (home advantage, toss advantage)). They used the following criteria:

Interpretability: A model such as Logistic Regression was selected due to its interpretability when it comes to figuring out the impact of features.

Categorical Data Processing: Due to the good handling of the categorical data that is encoded as one-hot and non-linear patterns, tree based models such as Decision Tree, Random Forest and Gradient Boosting were taken into account.

Comparative Accuracy: Comparative accuracy measured models assessed on metrics such as accuracy, precision, and recall, as well as F1-score.

Strength in Imbalanced Data: The models were resistant to class imbalance, particularly caused by teams that have fewer matches. Fair evaluation was provided with the help of weighted and macro metrics.

7. Model Architecture

Logistic Regression is a linear algorithm of classification that draws a dependence between the features and the probability of a specific category label. In the binary scenario, it applies the sigmoid activity to apply the linear combination of input features into a probability of the arrangement of 0 to 1. When the problem being solved is multi-class, as in the problem where we have to predict which one of the several cricket teams wins the match, it outputs a probability of a class using the softmax function. The model comprises a set of weights and a bias unit, which are being learnt through training using a loss function, which most time is the cross-entropy loss. Logistic Regression presupposes a linear decision boundary, and it performs the best when the relationship between the features and the target is not quite linear.

Decision Tree Classifier is a non-linear model that divides the data into partitions about feature values and creates the kind of tree. The internal nodes denote tests on features (e.g. "Did Team1 win the toss?"), The branches to these nodes denote the results of the test, and the leaf nodes denote the predicted classes (e.g. the winning team). On each step, the model applies a greedy algorithm in choosing the feature and the threshold that offers the maximum information gain or minimum Gini impurity. Growth of the tree continues until one of the stopping criteria has been reached, which could be an upper limit of a depth or a lower limit of the number of instances in a node. Decision trees can be used on both categorical and numerical datasets and do not need feature scaling. They can, however, be vulnerable to overfitting when the tree happens to be too deep or fancy.

Random Forest Classifier is an extension of Decision Tree in that it is developed by training hundreds of Decision Trees, each separately, on random subsets of the data and features. That is accomplished using the technique known as bagging (bootstrap aggregation), which adds randomness and diversity to the trees. When it is time to make predictions, the outcome of a tree is a vote, and the output is the class to which the majority of trees vote. Such a combination allows decreasing variance and overfitting over a single tree, yet it preserves high levels of accuracy. They further give feature importance scores, which aid in the interpretation of the feature that plays a greater role in the predictions.

Gradient Boosting is another common approach to ensemble learning, which trains models in series and trains each new model to minimise the error of all previous models. In contrast with Random Forests, where each tree is trained randomly and ensembles trees sequentially, Gradient Boosting grows trees sequentially, and each tree tries to maximise the performance of the model by minimising a given loss. At each iteration, the model calculates the residuals (the differences between actual and predicted values) and fits a new weak learner—typically a shallow decision tree—to these residuals. The outputs are added in an additive way. The issue of overfitting is also averted by a learning rate on the amount by which each new tree is added to the model as a whole. Gradient Boosting is very strong and able to model non-linear and complicated dependencies in the data, but sometimes it is computationally heavy and prone to parameter tuning.

K-Nearest Neighbours (KNN) algorithm is an instance-based non-parametric model that fails to construct an explicit model during training. Rather, it captures all training information and directly relies on it to make predictions. Upon input of a new observation, the algorithm computes a distance between this and every single point in the training set, usually a Euclidean one. It then identifies the k nearest neighbours and votes for the neighbours to classify the new instance by majority. Since KNN is based on distance computations, it is also sensitive to feature scaling, and it can suffer in high-dimensional domains due to the curse of dimensionality. As much as it may seem simple, KNN may represent complex patterns given that the data is properly arranged and that the appropriate number of k is selected.

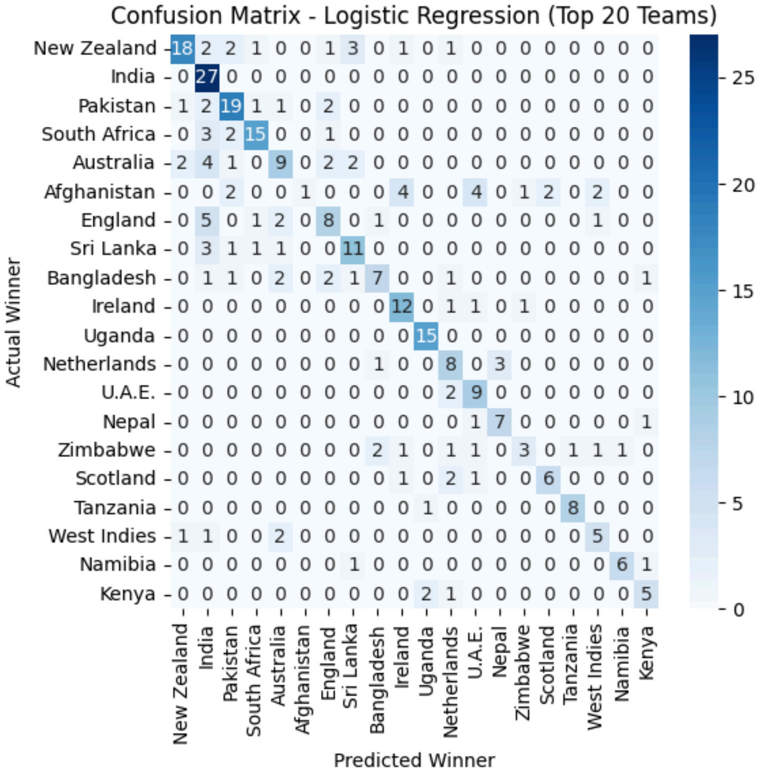
8. Evaluation

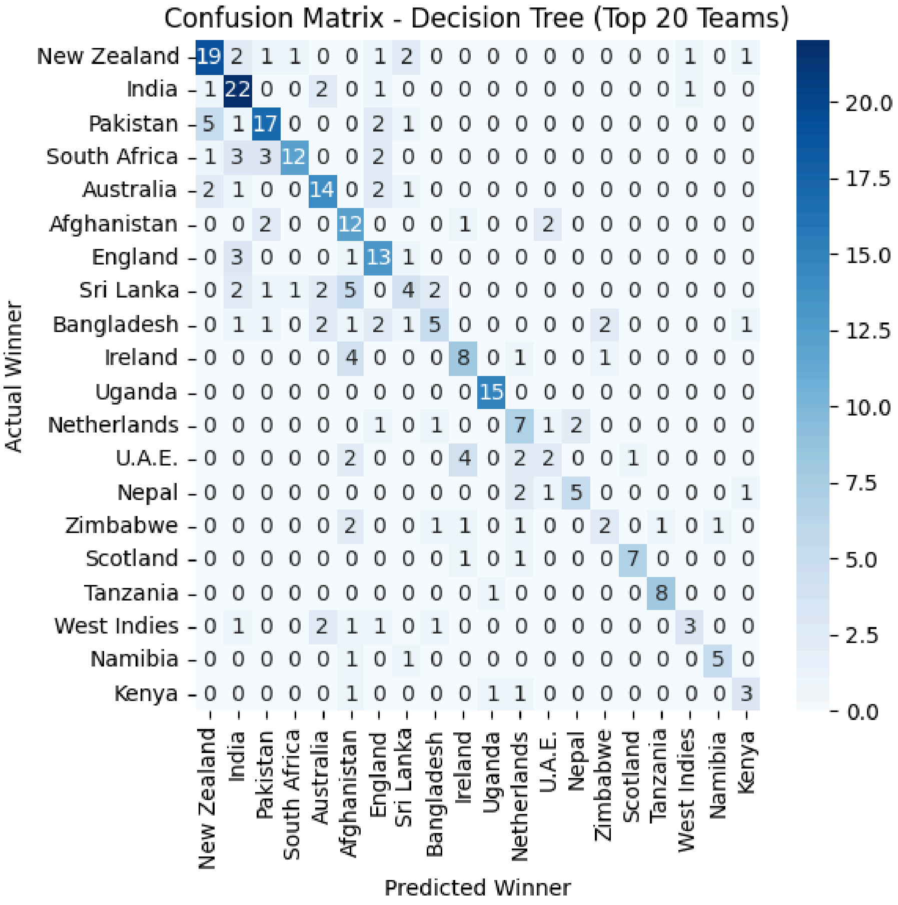
I employed some measures to assess the performance of the models such as accuracy, precision, recall, and F1-score. These measures assist in what number of predictions were precise yet in addition how great the model was on the spatial information (precision), should also be of specific interest in a multi-classification exercise such as this match victor in T20 cricket.

Accuracy = (Correct Predictions / Total Predictions) × 100

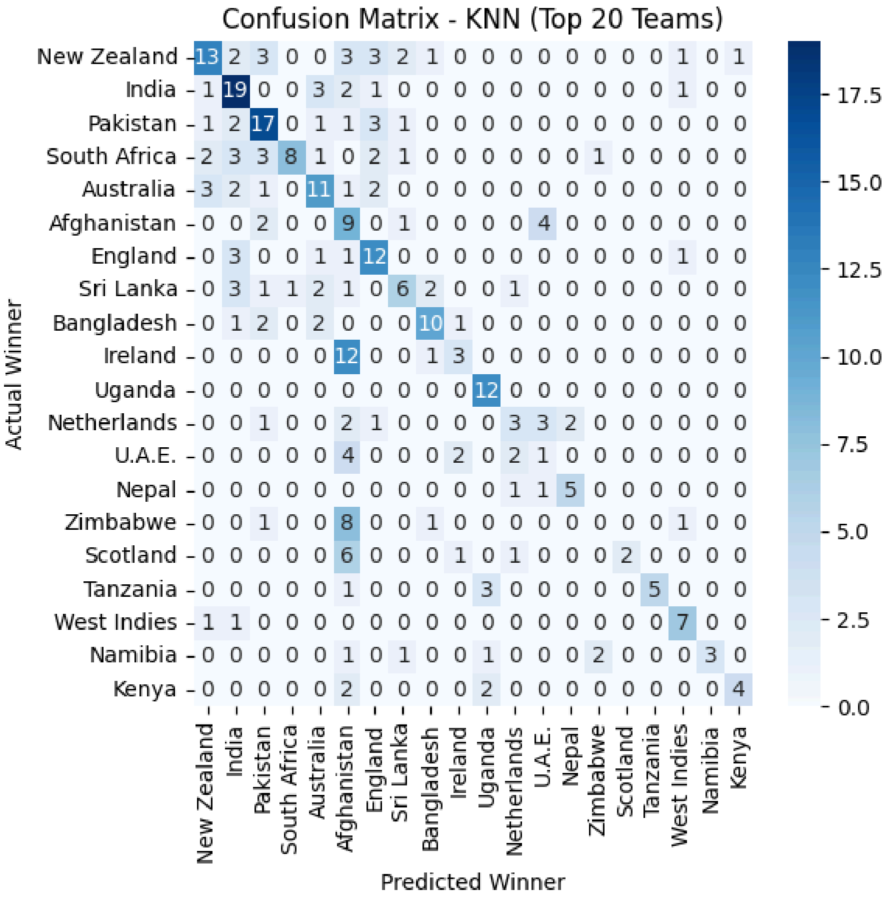
Metric	**Logistic Regression**	**Decision Tree**	**KNN**	**Random Forest**	**Gradient Boosting**
Accuracy	**62%**	**58%**	**44%**	**59%**	**60%**
Macro Precision	0.54	0.51	0.35	0.55	0.55
Macro Recall	0.53	0.51	0.30	0.55	0.55
Macro F1-score	0.51	0.48	0.29	0.51	0.52
Weighted Precision	0.63	0.60	0.50	0.63	0.64
Weighted Recall	0.62	0.58	0.44	0.59	0.60
Weighted F1-score	0.59	0.57	0.44	0.58	0.60

Logistic Regression and Gradient Boosting provided a good balance and an accuracy of **62% and 60%,** respectively. The degree of accuracy was moderate in all the models. However, this differs with the teams as there is an imbalance of classes since some teams will have more matches. The confusion matrix is below:

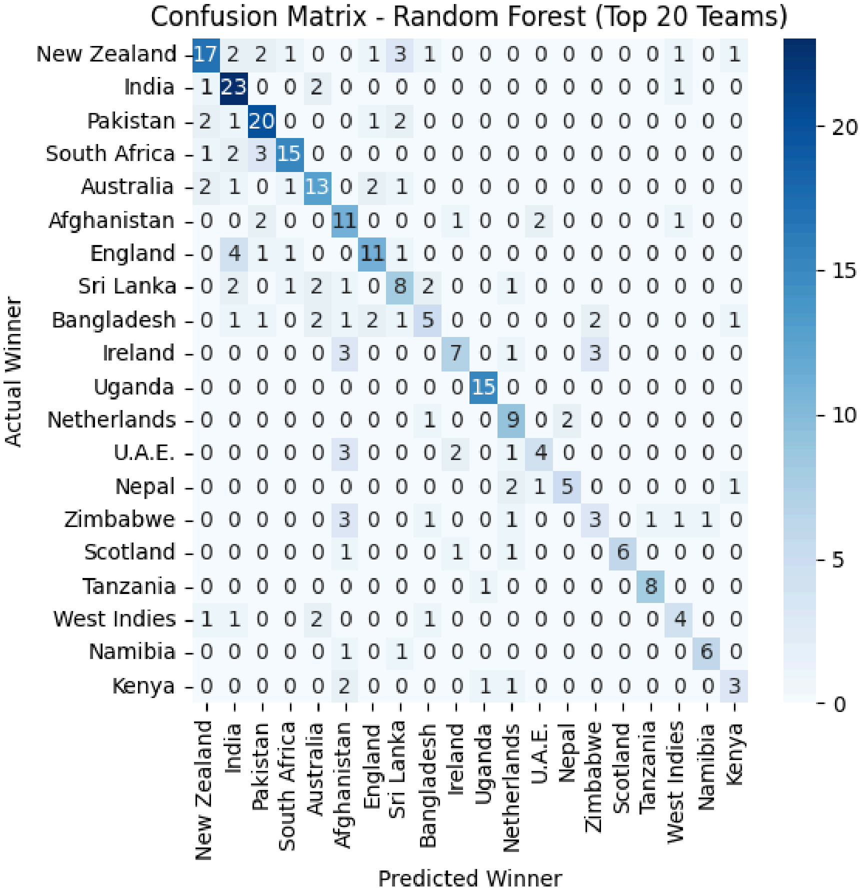


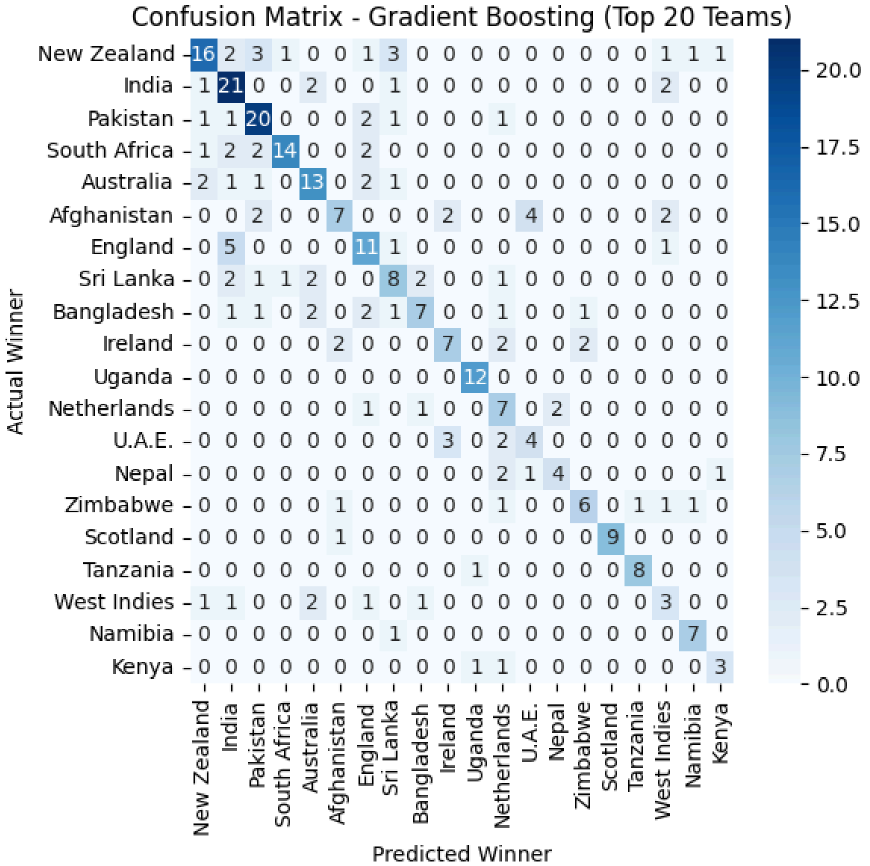


	New Zealand	India	Pakistan	South Africa	Australia	Afghanistan	England	Sri Lanka	Bangladesh	Ireland	Uganda	Netherlands	U.A.E.	Nepal	Zimbabwe	Scotland	Tanzania	West Indies	Namibia	Kenya
New Zealand	19	2	1	1	0	0	1	2	0	0	0	0	0	0	0	0	0	1	0	1
India	1	22	0	0	2	0	1	0	0	0	0	0	0	0	0	0	0	1	0	0
Pakistan	5	1	17	0	0	0	2	1	0	0	0	0	0	0	0	0	0	0	0	0
South Africa	1	3	3	12	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0
Australia	2	1	0	0	14	0	2	1	0	0	0	0	0	0	0	0	0	0	0	0
Afghanistan	0	0	2	0	0	12	0	0	0	1	0	0	2	0	0	0	0	0	0	0
England	0	3	0	0	0	1	13	1	0	0	0	0	0	0	0	0	0	0	0	0
Sri Lanka	0	2	1	1	2	5	0	4	2	0	0	0	0	0	0	0	0	0	0	0
Bangladesh	0	1	1	0	2	1	2	1	5	0	0	0	0	0	2	0	0	0	0	1
Ireland	0	0	0	0	0	4	0	0	0	8	0	1	0	0	1	0	0	0	0	0
Uganda	0	0	0	0	0	0	0	0	0	0	15	0	0	0	0	0	0	0	0	0
Netherlands	0	0	0	0	0	0	1	0	1	0	0	7	1	2	0	0	0	0	0	0
U.A.E.	0	0	0	0	0	2	0	0	0	4	0	2	2	0	0	1	0	0	0	0
Nepal	0	0	0	0	0	0	0	0	0	0	0	2	1	5	0	0	0	0	0	1
Zimbabwe	0	0	0	0	0	2	0	0	1	1	0	1	0	0	2	0	1	0	1	0
Scotland	0	0	0	0	0	0	0	0	0	1	0	1	0	0	0	7	0	0	0	0
Tanzania	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	8	0	0	0
West Indies	0	1	0	0	2	1	1	0	1	0	0	0	0	0	0	0	0	3	0	0
Namibia	0	0	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	5	0
Kenya	0	0	0	0	0	1	0	0	0	0	1	1	0	0	0	0	0	0	0	3



	New Zealand	India	Pakistan	South Africa	Australia	Afghanistan	England	Sri Lanka	Bangladesh	Ireland	Uganda	Netherlands	U.A.E.	Nepal	Zimbabwe	Scotland	Tanzania	West Indies	Namibia	Kenya
New Zealand	13	2	3	0	0	3	3	2	1	0	0	0	0	0	0	0	0	1	0	1
India	1	19	0	0	3	2	1	0	0	0	0	0	0	0	0	0	0	1	0	0
Pakistan	1	2	17	0	1	1	3	1	0	0	0	0	0	0	0	0	0	0	0	0
South Africa	2	3	3	8	1	0	2	1	0	0	0	0	0	0	1	0	0	0	0	0
Australia	3	2	1	0	11	1	2	0	0	0	0	0	0	0	0	0	0	0	0	0
Afghanistan	0	0	2	0	0	9	0	1	0	0	0	0	4	0	0	0	0	0	0	0
England	0	3	0	0	1	1	12	0	0	0	0	0	0	0	0	0	0	1	0	0
Sri Lanka	0	3	1	1	2	1	0	6	2	0	0	1	0	0	0	0	0	0	0	0
Bangladesh	0	1	2	0	2	0	0	0	10	1	0	0	0	0	0	0	0	0	0	0
Ireland	0	0	0	0	0	12	0	0	1	3	0	0	0	0	0	0	0	0	0	0
Uganda	0	0	0	0	0	0	0	0	0	0	12	0	0	0	0	0	0	0	0	0
Netherlands	0	0	1	0	0	2	1	0	0	0	0	3	3	2	0	0	0	0	0	0
U.A.E.	0	0	0	0	0	4	0	0	0	2	0	2	1	0	0	0	0	0	0	0
Nepal	0	0	0	0	0	0	0	0	0	0	0	1	1	5	0	0	0	0	0	0
Zimbabwe	0	0	1	0	0	8	0	0	1	0	0	0	0	0	0	0	0	1	0	0
Scotland	0	0	0	0	0	6	0	0	0	1	0	1	0	0	0	2	0	0	0	0
Tanzania	0	0	0	0	0	1	0	0	0	0	3	0	0	0	0	0	5	0	0	0
West Indies	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	7	0	0
Namibia	0	0	0	0	0	1	0	1	0	0	1	0	0	0	2	0	0	0	3	0
Kenya	0	0	0	0	0	2	0	0	0	0	2	0	0	0	0	0	0	0	0	4


Confusion Matrix - Random Forest (Top 20 Teams)
Actual Winner
Predicted Winner
New Zealand 17 2 2 1 0 0 1 3 1 0 0 0 0 0 0 0 0 1 0 1
India 1 23 0 0 2 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0
Pakistan 2 1 20 0 0 0 1 2 0 0 0 0 0 0 0 0 0 0 0 0
South Africa 1 2 3 15 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
Australia 2 1 0 1 13 0 2 1 0 0 0 0 0 0 0 0 0 0 0 0
Afghanistan 0 0 2 0 0 11 0 0 0 1 0 0 2 0 0 0 0 1 0 0
England 0 4 1 1 0 0 11 1 0 0 0 0 0 0 0 0 0 0 0 0
Sri Lanka 0 2 0 1 2 1 0 8 2 0 0 1 0 0 0 0 0 0 0 0
Bangladesh 0 1 1 0 2 1 2 1 5 0 0 0 0 0 2 0 0 0 0 1
Ireland 0 0 0 0 0 3 0 0 0 7 0 1 0 0 3 0 0 0 0 0
Uganda 0 0 0 0 0 0 0 0 0 0 15 0 0 0 0 0 0 0 0 0
Netherlands 0 0 0 0 0 0 0 0 1 0 0 9 0 2 0 0 0 0 0 0
U.A.E. 0 0 0 0 0 3 0 0 0 2 0 1 4 0 0 0 0 0 0 0
Nepal 0 0 0 0 0 0 0 0 0 0 0 2 1 5 0 0 0 0 0 1
Zimbabwe 0 0 0 0 0 3 0 0 1 0 0 1 0 0 3 0 1 1 1 0
Scotland 0 0 0 0 0 1 0 0 0 1 0 1 0 0 0 6 0 0 0 0
Tanzania 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 8 0 0 0
West Indies 1 1 0 0 2 0 0 0 1 0 0 0 0 0 0 0 0 4 0 0
Namibia 0 0 0 0 0 1 0 1 0 0 0 0 0 0 0 0 0 0 6 0
Kenya 0 0 0 0 0 2 0 0 0 0 1 1 0 0 0 0 0 0 0 3
New Zealand India Pakistan South Africa Australia Afghanistan England Sri Lanka Bangladesh Ireland Uganda Netherlands U.A.E. Nepal Zimbabwe Scotland Tanzania West Indies Namibia Kenya
20
15
10
5
0


Confusion Matrix - Gradient Boosting (Top 20 Teams)
Actual Winner
Predicted Winner
New Zealand 16 2 3 1 0 0 1 3 0 0 0 0 0 0 0 0 0 1 1 1
India 1 21 0 0 2 0 0 1 0 0 0 0 0 0 0 0 0 2 0 0
Pakistan 1 1 20 0 0 0 2 1 0 0 0 1 0 0 0 0 0 0 0 0
South Africa 1 2 2 14 0 0 2 0 0 0 0 0 0 0 0 0 0 0 0 0
Australia 2 1 1 0 13 0 2 1 0 0 0 0 0 0 0 0 0 0 0 0
Afghanistan 0 0 2 0 0 7 0 0 0 2 0 0 4 0 0 0 0 2 0 0
England 0 5 0 0 0 0 11 1 0 0 0 0 0 0 0 0 0 1 0 0
Sri Lanka 0 2 1 1 2 0 0 8 2 0 0 1 0 0 0 0 0 0 0 0
Bangladesh 0 1 1 0 2 0 2 1 7 0 0 1 0 0 1 0 0 0 0 0
Ireland 0 0 0 0 0 2 0 0 0 7 0 2 0 0 2 0 0 0 0 0
Uganda 0 0 0 0 0 0 0 0 0 0 12 0 0 0 0 0 0 0 0 0
Netherlands 0 0 0 0 0 0 1 0 1 0 0 7 0 2 0 0 0 0 0 0
U.A.E. 0 0 0 0 0 0 0 0 0 3 0 2 4 0 0 0 0 0 0 0
Nepal 0 0 0 0 0 0 0 0 0 0 0 2 1 4 0 0 0 0 0 1
Zimbabwe 0 0 0 0 0 1 0 0 0 0 0 1 0 0 6 0 1 1 1 0
Scotland 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 9 0 0 0 0
Tanzania 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 8 0 0 0
West Indies 1 1 0 0 2 0 1 0 1 0 0 0 0 0 0 0 0 3 0 0
Namibia 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 7 0
Kenya 0 0 0 0 0 0 0 0 0 0 1 1 0 0 0 0 0 0 0 3
New Zealand India Pakistan South Africa Australia Afghanistan England Sri Lanka Bangladesh Ireland Uganda Netherlands U.A.E. Nepal Zimbabwe Scotland Tanzania West Indies Namibia Kenya
20.0
17.5
15.0
12.5
10.0
7.5
5.0
2.5
0.0

9. Hyperparameter Tuning

The estimation of parameters constituting an enhanced machine learning model performance is referred to as hyperparameter tuning. To give each one of the models a hyperparameter tuning in this project, the GridSearchCV was applied. GridSearchCV exhaustively searches over a specified parameter grid and evaluates model performance using cross-validation(cv=3).

Models	Best Parameters	Result After Tuning	Result Before Tuning
Logistic Regression	'C': 1, 'max_iter': 500, 'solver': 'liblinear'	61%	62%
Decision Tree	'criterion': 'gini', 'max_depth': None, 'min_samples_leaf': 1, 'min_samples_split': 10	55%	58%
Random Forest	'bootstrap': True, 'max_depth': None, 'max_features': 'sqrt', 'min_samples_split': 4, 'n_estimators': 100	59%	59%
Gradient Boosting	'learning_rate': 0.01, 'max_depth': 3, 'n_estimators': 100, 'subsample': 1.0	59%	60%

10. Proof of Concept (POC)

The complete implementation, dataset, code, and results are available on GitHub.
🔗 GitHub Repository

11. Conclusion

The project proves that machine learning is a viable way of predicting the outcome of T20 cricketing matches based on match based variables like which team plays, result of the toss, the advantage of playing on home, past performance of the teams. One model that performs the best in terms of predictive ability is **Logistic Regression with 62% accuracy**.

The next best step will be to integrate the player-level statistics, weather in the game, ball-by-ball engineering and even more detailed match events to predict more accurately. As well, more advanced models of capturing complex dependencies, such as deep learning, might be investigated.