University of Essex

School of Mathematics, Statistics
and Actuarial Science

MA981 DISSERTATION

Predicting Football Outcomes with Machine Learning

Pathakoti Rahul

2312128

Supervisor: **Dr Andrew Harrison**

September 18, 2024
Colchester

Contents

List of Figures

List of Tables

CHAPTER 1

Abstract

The proposed project, in general, concerns the development and testing of two machine learning models for predicting either the outcome of a single football match or ranking at the end of the season. Our first model would predict a single match result, either a win or a (loss)not win (including loss draw), while the existing state-of-the-art approach in football prediction highlights the importance of goals scored in respect to shots and expected goals, defensive actions, and so forth. A second model projects which teams are well poised to finish at the top of the league by the end of the season.

The key features of this dataset include shot-creating actions and opponent data for various seasons, ranging from 2014 to 2024. Extensive preprocessing was done, including median imputation for missing values and encoding of categorical variables. Feature vectors were created by looking back at the last setback matches for each team in order to grasp historical trends that influence match outcomes.

Model-1 includes three classification algorithms, namely Random Forest, Support Vector Classifier (SVC), and XGBoost (XGB). Further, in order to balance class imbalance and enhance the performance of the model, oversampling was performed with the help of SMOTE (Synthetic Minority Over-sampling Technique) and a combination of SMOTE with undersampling (SMOTEENN). Because the balanced dataset led to a low accuracy of the model, stochastic justification was made.

Model-2 is designed to predict top-of-the-league teams through the deployment of some classification models: Random Forest, XGBoost, and Logistic Regression. The models were showing a high prediction correctness over the standings at the end of

that season and were good in long-term predictability with overall good evaluation metrics. Some main challenges while developing the model were handling the missing data across seasons, feature engineering, and proper labeling of the data. Models performance was then evaluated by standard performance metrics of accuracy, precision, recall, F1-score for an overall performance about the top-of-the-league teams.

Keywords: FootballPredictor, machine learning, outcome of match prediction, topleague team predictor, sport analytics, EP, xG, Random Forest, SVC, XGBoost, SMOTE, SMOTEENN, stochastic process, ensemble learning, class imbalance, feature engineering, season ranking, deep learning.

CHAPTER 2

Introduction

Besides being a sport, football, or soccer for some parts of the world, has grown to become a worldwide trend engraved in the hearts and minds of millions. Amongst the many present football leagues in the world, the English Premier League is one of the tallest and most competitive. Being the platform for fast action on the pitch, commitment, and talent with various world-class teams, most of the time, a number of top stars are held within the Premier League. Generally, it is regarded as the pinnacle of club football. The league is very highly regarded and usually at the center of the footballing world, together with other European top leagues: the Spanish La Liga, German Bundesliga, and Italian Serie A. Among the different elements, the Premier League is primarily known for its unpredictability.[1]

There will be closely contested matches, and the outcome of any single game can come down to a moment of pure brilliance or another one of pure concentration lapse. That is what makes the Premier League exciting, yet it also partly forms one of the huge tests analysts and their many fans face in trying to forecast match results.[2] Over the past years, leading teams included Manchester City, Liverpool, Chelsea, and Arsenal. However, upsets and surprises arrive too often, making predictions difficult to make and highly popular at the same time.

The project below is a prediction of the outcome of a match using various variables with machine learning methods. The thoughtfulness shall consider the matches from the Premier League. To this end, a dataset concerning premier league football matches ranging from 2014 to 2024 shall be analyzed, considering such variables as goals scored,

expected goals (xG), shots, and defensive actions.The extended analysis would add to the accuracy in predicting match outcomes of the future through patterns and trends.[3] A project shall develop two types of models: one for predicting individual match results- a model that says who wins or a (loss)not win (including loss draw)the match and another model that is aimed at predicting which teams are likely to finish top league by the end of the season, given everything else held constant. The key features are expected goals, shot-creating actions, and a few defensive metrics analyzed on data from 2014 through 2024. Model-1 implements Random Forest, SVC, and XGBoost, with oversampling via SMOTE[4] and SMOTEEN for class imbalance; stochastic adjustments ensure stability.

Model-2 predicted the end-of-season rank for football teams using Random Forest, XGBoost Classifier, and Logistic Regression. In the study, extensive preprocessing, feature engineering, and model evaluations were performed to visualize the capability of machine learning in sports predictions. The major challenges during this procedure were how to handle missing data, perform feature selection, and check whether these models were robust or not. All of them have been widely measured in terms of accuracy, precision, recall and F1-score underlining their efficiency in predicting seasonal outcomes.

CHAPTER 3

Objectives and Challenges of project

The project will develop two distinct machine learning models: Model-1, which will predict the win-(loss)not win (including loss draw) outcome for any single football match based on the rich dataset from 2014 up to 2024; this dataset encompasses several features like goals scored, xG, shots, defensive actions, team forms, and many more for enhancing predictive accuracy in the models.Model-2 aims at predicting which teams would end up topping the league by the end of each season.[5] This model explores team performance as a function of time, using seasonal performance trends and drawing on a team's historical ranking for long-term insight. The most influential attributes in determining season-end rankings are determined by advanced feature engineering.

Both models use the experimentation of machine learning algorithms including Random Forest, Support Vector Classifier, XGBClassifier, and Logistic Regression. These models will be compared and examined through iterative optimization to make sure they are effective using standard machine learning metrics: accuracy, precision, recall, F1-score, and ROC AUC.[6] It also provides insights into potential follow-up studies by mentioning the incorporation of testing deep learning models for more accuracy, and the consideration of real-time data to make in-season predictions even stronger, making the outlook for the future wider for enhancements in sports analytics.

3.1 Challenges in Data Handling and Preprocessing

Missing Data: One big challenge in this project is how to handle missing values across several seasons. Some features or metrics may be incomplete in football datasets due to differences in data capture or reporting. Median imputation was thus performed to fill in such missing value gaps with little disturbance in the data distribution.[7] However, the imputation of missing values sometimes over-simplifies the true underlying pattern, and biases that can be introduced into the data need to be judiciously handled.

Imbalanced Data: The other problem was related to class balancing. In few teams tend to dominate when it comes to wins, and this clientele is therefore quite skewed toward such outcomes. To handle this problem, oversampling techniques like SMOTE and SMOTEENN have been performed.[8] While these techniques balanced out the dataset, they led to complexities in model performance, such as overfitting or reduced accuracy.

3.2 Challenges in Model Development and Evaluation

Model Selection and Implementation: The choice of the appropriate machine learning models was one of the key challenges in this project. In Model-1, which predicts the outcome of an individual match, three classification algorithms were adopted: Random Forest, Support Vector Classifier (SVC), and XGBoost Classifier (XGB). Each has different strengths: robustness to overfitting for Random Forest[9], complicated decision boundaries for SVC, structured data with gradient boosting for XGBoost.[10] The challenge was to implement and fine-tune each model to obtain the best results, considering computational costs and performance trade-offs. Model selection needed to be pretty iterative in nature in finding out which algorithm shall perform best given the class highly imbalanced nature with feature complexity. Model-2 used a Random Forest, XGBoost Classifier, and Logistic Regression to predict the end-of-season ranking of football teams. All three models were selected to take advantage of their strengths with larger datasets and interaction complexities among their features. Although all the three models contributed to the overall analysis, Logistic Regression had the best results in Test that had to do with both accuracy and a good balance regarding all the

key metrics under observation: precision, recall, and F1-score. That has pointed out the suitability of Logistic Regression for the nature of this data set, where its linearity was essential for the ranks' prediction. While Random Forest and XGBoost are really very powerful in modeling nonlinear relationships, in this case, Logistic Regression outperformed both.

Feature Engineering: Feature engineering for Model-1 is to build feature vectors from the participant teams based on their last ten matches. It includes a key performance metric in the format of expected goals such as xG, shot-creating actions, and defensive activities of each team, together with the match outcome in terms of win/loss. Possible naming of each feature dynamically is considered depending on the opponent history to ensure consistency within the dataset. The placeholder values were used to fill the historical data gaps on previous games of each team, representing missing or incomplete information. In this way, the final dataset contains 351 columns which capture a variety of recent performance trends for the teams in order to predict their result for the next match.[11]

In Model-2 -Top-of-League Prediction, feature vectors have been created by analyzing each team past matches across seasons. To capture the long-term trends, important features such as goals, xG, SCA, and overall team performance over time were extracted. Having aggregated seasonally in reflection of the historical pattern of performance most useful for the purpose of predicting whether a given team had finished at the top of the league, winning teams and their seasons were suitably labeled.[12] The final feature set included 622 features per combination of team-season-a robust basis indeed for the prediction of season-ending ranks.

Evaluation and Validation: Different key metrics have been used to evaluate these models in the project. This can give a fair and comprehensive look into how the models perform. Accuracy was the chief measure used. It represents the correct proportion out of the total prediction. However, considering the possibility of class imbalance, using only accuracy would not be adequate. Precision was used to that effect to measure the ratio of true positive predictions against total positive predictions given while ensuring that false positives were kept as low as possible because predicting[13] a win when it is actually a loss is very costly.

Recall was also a metric applied in determining the capability of the model in

capturing true positive outcomes with much emphasis on how well the model can correctly predict wins out of the total actual wins. The F1-score, being the harmonic mean of precision and recall, was used as a balance between both metrics; especially useful when dealing with imbalanced datasets.

The last one is ROC AUC, meaning Receiver Operating Characteristic[14] - Area Under the Curve, and will be applied in the current work for distinguishing between classes, especially for win/(loss)not win (including loss draw) outcome predictions. A higher value of AUC is an indication that a model is good at distinguishing positive and negative instances. These combined metrics gave full-scale diagnostics of how those models were performing in terms of handling class imbalance and making reliable predictions.

CHAPTER 4

Literature Review

Football match outcome predictions have been a growing interest in the sports analytics domain, with recently increasing momentum due to the development of machine learning and increased access to in-depth match data. This literature review discusses a few of the major works and approaches that have helped in understanding and building predictive models in football, focusing on machine learning algorithms.

Football, more popularly known in many parts of the world as soccer, is a sport played and widely watched globally by billions of fans and players. The sport epitomizes restrained simplicity: two teams struggle against each other to score goals,[15] finding means of getting a ball into an opponent's net with any part of the body except hand and arm. However, strategic complexity, physical demands, and psychological intensity pronounce the game for rich analysis. The EPL and other top leagues in Europe are particularly considered to be well known in terms of high competition levels, high levels of global viewership, and high financial stakes. Data analytics in football has noticeably surged in the last couple of years with clubs and analysts apparently trying to gain an edge over their opponents through the use of sophisticated statistical models and machine learning algorithms in predicting outcomes of matches, player performances, and other key areas of the game. The introduction of data analytics into football has somewhat changed the way teams, coaches, and analysts approach the game. [16]Given that a single match creates an enormous volume of data-from player movements to the tactical formations adopted-machine learning has emerged as one of the strong technologies in the domain of sports analytics. These technologies make

matched outcome predictions more accurate by providing valuable insights into things like team strategy and player selection, even to betting markets.

4.1 Early Approaches to Football Prediction/Statistical Methods and Poisson Models

In fact, most of the early models forecasting football match outcomes are statistically oriented.[17] he applied Poisson regression models to predict the number of goals each team would score. This model assumes that each team's goals scored follow a Poisson distribution dependent on the attacking strength of the team and the opposition's defensive strength. The novelty of approach to modeling consisted in that his model introduced into practice team-specific factors for match outcome prediction. Still, the model was rather simple and did not consider a great number of contextual factors that may influence the result in a football match, such as player form, injuries, and tactical decisions.

4.1.1 Bayesian Methods

Building on the deficiencies of Poisson models,[18] began to explore ways in which Bayesian networks can model football outcomes. Bayesian networks introduced a probabilistic graphical model through which researchers could integrate variables specific to teams and contextual variables related to home advantage, team form, and even extra psychological aspects like those affecting team morale. The most profound contribution from the work was the introduction of a way of managing the natural uncertainty of the prediction regarding the outcome of football matches and introducing expert knowledge into the model. It also showed improved predictability compared to more traditional statistical methods, especially in complex, uncertain environments such as football matches.

4.2 Machine Learning in Football Predictions

4.2.1 Supervised Learning Approaches

Football match prediction has become very futuristic with the incorporation of machine learning.[19] implementing different supervised machine learning techniques for predicting outcomes of matches in Dutch. From decision trees to random forests and support vector machines,they tried everything, but the best performance came out for the ensemble methods such as random forests. Herein lies the power of these models: handling non-linear complex relationships between variables, a common occurrence with football data. Feature selection was underlined in the study as one of the crucial points since the choice of the input variables-like recent form or historical head-to-head records-will strongly determine the accuracy of the model.

4.2.2 Feature Engineering

Feature engineering to be potentially very important in enhancing the predictive power of machine learning models in football.[20] A study that they conducted in the Belgian Pro League indicated that xG-integrated advanced performance metrics, pass completion rate, and player-specific data increased model accuracy by a wide margin. The researchers also highlighted the capturing of the temporal dynamics of soccer games, such as momentum, player fatigue, and influence of early goals, which are usually missed in traditional models

4.3 Addressing Challenges in Football Prediction

4.3.1 Handling Imbalanced Data

One of the irritating issues in football match prediction is that the data are imbalanced. A large fraction of football matches, especially at the Premier League level, often involves a few dominant teams winning most of their matches, usually leading to an imbalanced distribution of outcomes. Their[21] went over the problem by resampling techniques such as SMOTE, Synthetic Minority Over-sampling Technique, in order to balance the training data. Their work focused on the English Premier League and showed that class

imbalance issues happen in this domain, which could affect the general performance of predictive models, particularly for the less frequent events, such as draws or wins by the underdog team.

4.3.2 Model Evaluation and Overfitting

But in themselves, football prediction models are not without their own challenges when coming to evaluation, looking into metrics other than simple accuracy for evaluation, such as precision, recall, F1 score, and the Area Under the ROC Curve (AU ROC).[22] They argued that these metrics allow a better understanding of the performance of a model, something particularly relevant when dealing with imbalanced datasets. Another important problem which arises with many football prediction models is the overfitting problem-a model performing well on training but poorly on unseen data. Investigated ensemble methods, including random forests, which aggregate predictions over multiple models in order to reduce variance in predictions and are thus less prone to overfitting.

4.4 Reseach paper review on Predicting Football Results Using Machine Learning Techniques

The paper "Predicting Football Results Using Machine Learning Techniques", who did it as a part of his academic work at Imperial College London. This study is an in-depth investigation into the application of ML methodologies for predicting the outcomes of football matches. The key objective of the study has been to install this heightened level of accuracy in predictions based on in-game events, rather than merely traditional metrics such as goals scored, and to introduce more sophisticated ML models. This chapter, for instance, describes how to develop an 'expected goals' metric that will correctly measure the performance of a team in a far superior way to just actual goals, thereby removing some of the randomness associated with scoring goals.

This research systematically tests several ML models, such as Generalized Linear Models, Decision Trees, and Support Vector Machines, in predicting the results and scores of the matches. The author also integrates team ratings, similar to ELO ratings, into the predictive models to reflect the teams' attacking and defensive capabilities. One

of the most valuable contributions of this paper is combining expected goals with these team ratings to come up with more robust classification and regression models.

Besides that, the experimental nature of the study involved cross-validation and sound performance metrics such as accuracy, F1 score, and mean absolute error, [23]hence making the models robust. These results show that, given comprehensive match data, modern machine learning techniques are able to give predictive performance comparable to bookmakers' models.

This research, therefore, demonstrates high-level capability for sophisticated ML models and granular football data to improve predictive accuracy in sports analytics. The paper further acknowledges some challenges with data quality and data availability, particularly when developing detailed datasets to use in training the models. Future work can therefore be directed at an even deeper integration of player-level data and more sophisticated ML techniques, such as neural networks, to further advance the predictive accuracy in football.

4.5 Real-time Forecasting within Soccer Matches through a Bayesian Lens

The investigation paper is given at the Indian Institute of Management Bangalore and the University of Essex,[24] on the title "Real-time Forecasting within Soccer Matches through a Bayesian Lens". An attempt is also made to further estimate the time-varying impact of different in-game events using the multinomial probit regression model; this is done using a dataset comprising more than 3000 matches in the English Premier League over eight seasons. The authors build a Bayesian model for the updating of the usual probabilities about the possible outcomes of a football match: win, lose, or draw, occurring every new information in the form of a goal, a red card, a corner, and so on, which can provide real-time predictions increasingly more accurate as the game goes on.

It has all these things well balanced in traditional approaches, for predicting draws in a satisfying way of maintaining sensitivity and specificity through different match outcomes. For example, by the 75th minute, the F1 score for predicting draws goes up to 0.70, which outgoes other models like Generalized Linear Models and Random

Forest. On top of that, the Bayesian model showed lower Brier scores in the second half of the matches compared to the other models, therefore being more reliable during this situation type. The model picks up the changing impact of match events: the significance of a goal to win probability grows as the game nears its conclusion; and red cards reveal huge detrimental blows to the home team's chances, especially in the later parts of a match.

Concerning other models, namely GLM, Linear SVM, R-SVM, and RF, the Bayesian model also continuously outperformed the rest, mainly in the respect of maintaining the prediction level for a period of time. This is graphical in the results as presented by the research, where one can clearly see the Bayesian model outperforming other models in F1-score comparisons, especially when predicting draws. In general, this means that the Bayesian framework allows for better prediction accuracy, but with an added value from the viewpoint of effective adaptation to the dynamic characteristics of soccer games overall. This infers the application to live sports analytics. The model would be considered a major breakthrough in the sport and possibly have applications in other sports for real-time decision-making and strategic planning, owing to its ability to update the predictions as more and more data become available, just as the model maintained its performance in varied match scenarios.

CHAPTER 5

Description of Dataset

5.1 Source of Dataset

The data for this project was extracted from the website FBref.com[25], one of the most comprehensive online sources of football statistics and historical data. Data used in this project comes from EPL pages at FBref.com, which keeps full track of detailed match logs of several teams. The URLs to extract data include statistics such as scores and fixtures, shooting, passing, and goal-creating actions for the 2014 to 2024 seasons. Teams engaged in this dataset are top-ranked teams in the Premier League, including Manchester City, Manchester United, Liverpool, Chelsea, Leicester City, West Ham United, Tottenham Hotspur, Arsenal, and Everton etc

5.2 Extraction procedure

5.2.1 Setup and Import Necessary Libraries

- **Pandas:** Used for data manipulation and table extraction from the web.
- **Time and random:** These libraries control the timing of the requests so they are not slamming the server, causing delays between retries.
- **Requests and RequestException:** These are utilized in handling HTTP requests and help with exceptions, more so when handling immense volumes from the

web.

5.2.2 Identify the URLs for Data Extraction

Identify the specific URLs from where data needs to be extracted. First, you will go to the website FBref and use the "Inspect" feature in your web browser to identify the structure of the HTML tables that we want to extract.

The urls list contains URLs representing the pages, which will be scrapped. Each URL points to a page containing the detailed match log for a particular team in the Premier League.

5.2.3 Initialize Data Storage and Data Extraction loop

Create an empty list to hold dataframes for extracted data of each URL. The list will then contain data from match logs of each team. Iterate over each URL in the urls list and try to scrape the HTML table containing the match data. The function pd.read html() reads the HTML and extracts the table into a pandas DataFrame.Loop also includes error handling over a try-except. In case a request fails, there is an exponential backoff with random delays: retries up to 5 times to avoid hammering the server or getting banned.

5.2.4 Combine and Save Data

After Extracting all the data from the website of all URLS, the individual Data-frames stored in "dataframes" are combined into a single Dataframe using 'pd.concat()' and combined Dataframe is then saved a CSV file.

5.3 Information and Dimension of Dataset

This Dataset contains 7600 Rows and 59 Columns and it contains different datatypes and this file is CSV format.This Dataset provides detailed match statistics, including goals, possessions, shot on targets and other performance metrics.

5.4 Description of Dataset Columns

1. **Date**: The date when the match was played (object type, likely in a date format).

2. **Time**: The time the match was played (object type).
3. **Round**: The round of the competition or match week (object type).
4. **Day**: The day of the week when the match was played (object type).
5. **Venue**: The location where the match was held (object type).
6. **Result**: The outcome of the match, typically indicating win, loss, or draw (object type).
7. **GF**: Goals For, the number of goals scored by the team in the match (integer).
8. **GA**: Goals Against, the number of goals conceded by the team in the match (integer).
9. **Opponent**: The name of the opposing team (object type).
10. **xG**: Expected Goals, a metric that estimates the likelihood of scoring based on the quality of shots (float).
11. **xGA**: Expected Goals Against, an estimate of the goals the opponent was likely to score (float).
12. **Poss**: Possession percentage, the percentage of time the team had control of the ball (integer).
13. **Attendance**: The number of people who attended the match (object type).
14. **Captain**: The name of the teamâs captain for the match (object type).
15. **Formation**: The tactical formation used by the team (object type).
16. **Referee**: The name of the match referee (object type).
17. **Notes**: Any additional notes or comments about the match (object type).
18. **Sh**: Shots, the total number of shots taken by the team (integer).
19. **SoT**: Shots on Target, the number of shots that were on target (integer).
20. **SoT%**: Percentage of shots that were on target (float).

21. **G/Sh**: Goals per Shot, the efficiency of converting shots into goals (object type).
22. **G/SoT**: Goals per Shot on Target, the efficiency of converting shots on target into goals (float).
23. **Dist**: Average shot distance from goal (float).
24. **FK**: Free Kicks, the number of free kicks taken (float).
25. **PK**: Penalty Kicks, the number of penalty kicks taken (integer).
26. **PKatt**: Penalty Kick Attempts, the number of penalty kicks attempted (integer).
27. **npxG**: Non-Penalty Expected Goals, expected goals excluding penalties (float).
28. **npxG/Sh**: Non-Penalty Expected Goals per Shot, efficiency of shots excluding penalties (object type).
29. **G-xG**: Goals minus Expected Goals, a measure of finishing quality (object type).
30. **np:G-xG**: Non-Penalty Goals minus Expected Goals (float).
31. **Cmp**: Completed Passes, the number of passes completed (float).
32. **Att**: Pass Attempts, the number of passes attempted (float).
33. **Cmp%**: Completion Percentage, the percentage of passes completed (object type).
34. **TotDist**: Total Distance, the total distance covered by players (object type).
35. **PrgDist**: Progressive Distance, the distance covered moving the ball towards the opponentâs goal (float).
36. **Ast**: Assists, the number of assists made (float).
37. **xAG**: Expected Assists Goals, expected goals from assists (float).
38. **xA**: Expected Assists, the expected assist value (float).
39. **KP**: Key Passes, passes that directly lead to a shot (float).
40. **P/3**: Passes in Final Third, passes made in the attacking third of the pitch (object type).

41. **PPA**: Passes into Penalty Area, passes that enter the penalty box (float).
42. **CrsPA**: Crosses into Penalty Area, crosses that enter the penalty area (float).
43. **PrgP**: Progressive Passes, passes that move the ball significantly forward (float).
44. **SCA**: Shot-Creating Actions, the number of actions leading to a shot (float).
45. **SCA_PassLive**: Shot-Creating Actions from Live Passes (float).
46. **SCA_PassDead**: Shot-Creating Actions from Dead Ball Situations (float).
47. **SCA_TO**: Shot-Creating Actions from Take-Ons (float).
48. **SCA_Sh**: Shot-Creating Actions from Shots (float).
49. **SCA_Fld**: Shot-Creating Actions from Fouls Drawn (float).
50. **SCA_Def**: Shot-Creating Actions from Defensive Actions (float).
51. **GCA**: Goal-Creating Actions, the number of actions leading to a goal (float).
52. **GCA_PassLive**: Goal-Creating Actions from Live Passes (float).
53. **GCA_PassDead**: Goal-Creating Actions from Dead Ball Situations (float).
54. **GCA_TO**: Goal-Creating Actions from Take-Ons (float).
55. **GCA_Sh**: Goal-Creating Actions from Shots (float).
56. **GCA_Fld**: Goal-Creating Actions from Fouls Drawn (float).
57. **GCA_Def**: Goal-Creating Actions from Defensive Actions (float).
58. **Season**: The season in which the match took place (integer).
59. **Team**: The name of the team for which the data is recorded (object type).

CHAPTER 6

Methodology

6.1 Handling Missing Values

Missing values were checked in the dataset. Several columns lacked data, although the percentage of missing values is different. The "Notes" column indeed had a very outstanding proportion of missing values, 99.97 percent, after which it was excluded from data because keeping it on was very uninformative to continue analysis.

With a closer view, columns such as 'xG', 'xGA', 'Dist', 'FK', among others, had almost 30 percent missing values, whereas 'Attendance' had about 11.59 percent and 'Captain' about 10 percent. For the remaining lot of columns, which included 'Date', 'Time', 'Round', 'Venue', there were no missing values at all.

Selective imputation of columns, depending on their importance and the extent of their missing values, was done for handling missing data in these cases.[26] The median per team value was selected for imputation in numeric columns important to the analysis, such as 'xG', 'xGA', and 'Ast'. This method is robust against outliers and does not impart any skew into the dataset as a result of imputation. The columns based on percentage, such as 'SoT' and 'Cmp', were also transformed by encoding into numeric types and imputed where necessary using medians.

Imputation was group-wise, a strategy adopted to ensure that the median values are computed within each group-in this case, grouped by 'Team'-and not across the entire dataset. This makes sure that imputation is context-specific for any given team's

performance.

For columns, which were non-numerical in nature, such as 'Time', 'Result', 'Poss', and 'Formation', Label Encoding was used. Such changes convert categorical data into numerical format, which a model can understand in an appropriate manner. After imputation and encoding were complete, the remaining rows containing missing values were dropped to complete the dataset to be ready for analyses and modeling. The final dataset was constituted of 4,301 rows and 58 columns, with no values missing; confirming indeed that all issues of missing data had successfully been resolved.

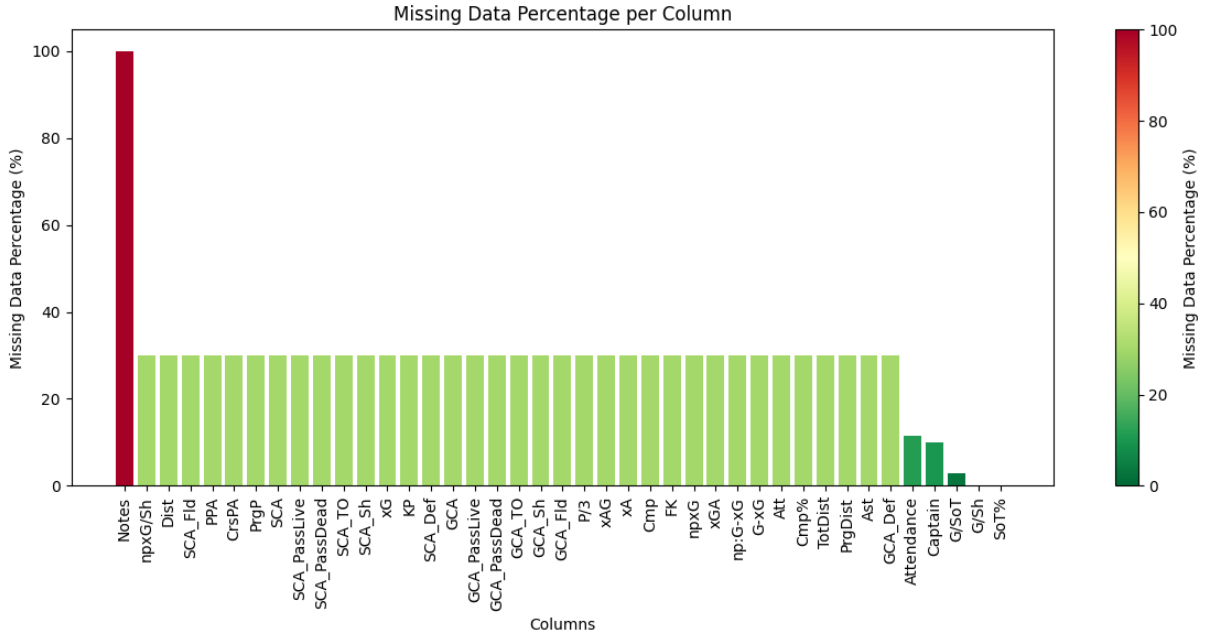

Figure 6.1: Chart Showing percentage of missing values

6.2 Descriptive Statistical Analysis and Visualization

Descriptive statistical analysis is necessary for understanding the summary, data distribution and the visual representation. Exploratory data analysis (EDA) gives us essential phase in the study process. With assistance of EDA we can identify intricate and hidden patterns in dataset

6.2.1 Bar chart of Manchester City GF vs xG

Comparing Goals For (GF) and Expected Goals (xG) for Manchester City (Month by Month, 2023-2024) The bar chart plots the actual goals scored (GF) versus Expected Goals (xG) analysis for the club of Manchester City, aggregated by month from January 2023 through May 2024. It underlines how well the number of goals actually scored by the team tallies with their expected goals a statistical prediction based on the nature of the chances created by the team.

April 2023 and April 2024 are the most particular months that have very high actually scored goals, standing at over 15 goals each. These months portend the dominance of Manchester City any moment it attacks during those months. For instance, in February 2023, there is a relatively higher expected goal than the actually scored goals insinuating that the team might have underperformed against the chances created. Other intriguing observations are those from August 2023, where the expected goals are way higher than the actual goals. That would be a good pointer to indicate Manchester City[27] have struggled to turn their chances into goals during that particular month.

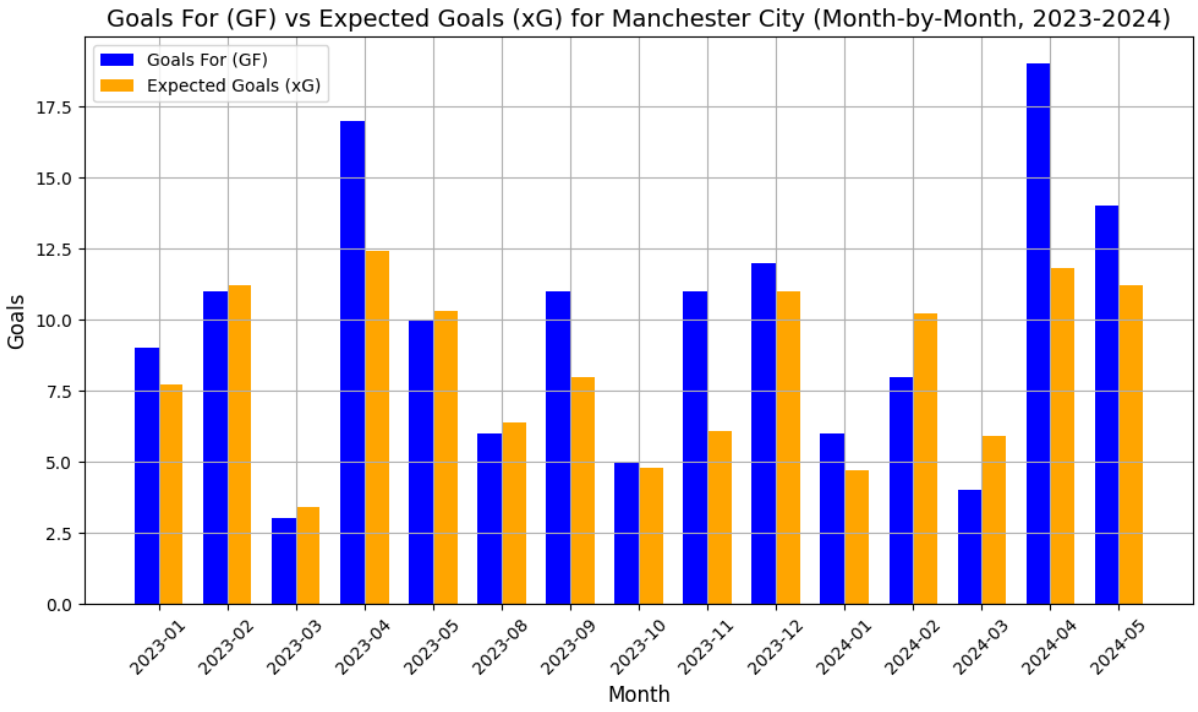

Figure 6.2: Bar plot showing Manchester city Gf vs Xg

The actual goals for most months are very close to the expected goals, something that indicates Manchester City has performed mostly to expectations. That consistency

indicates a team that usually performs to the quality of chances they create. This graph actually represents the attacking efficiency of Manchester City on a month-to-month basis relative to that expected of them. That it flags spells of sensational goal-scoring, such as April 2023 and 2024, pegs months where the team may have been less effective in converting chances, such as February and August 2023. Generally, this is a good overview of the general trend in the performance of the team throughout these two seasons.

6.2.2 Pie chart for Match Results

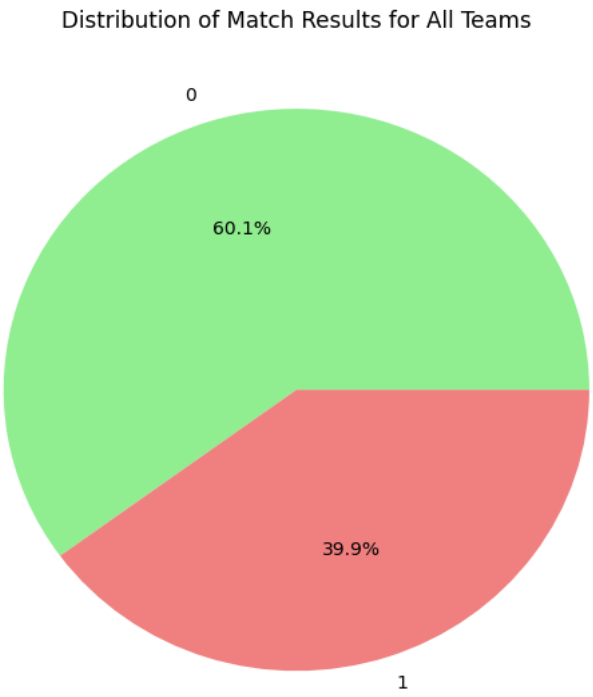

Figure 6.3: Distribution of Match Results for All Teams

The pie diagram "Distribution of Match Results for All Teams" gives, in graphical form, the break-up of all teams represented in the data set with regard to match outcome. Divided into two paras, the 60.1 percent represent losses-a light green portion-while the lighter red shows 39.9 percent of the matches won.

This distribution indicates that losses are more frequent because they are a bit greater in proportion to the total matches than the wins. This chart presents[28] clear and fast insight into how many matches end in a win versus a loss for all teams in the dataset.

It shows that losses outnumber the wins, which might be useful information meant to lead to an understanding of the trend in general performance within the dataset or league under scrutiny.

6.2.3 Scatter Plot of Expected Assists (xA) vs Assists (Ast) for Manchester City Vs Manchester United

Below is a scatter plot comparing the performances between Manchester City vs Manchester United based on their Expected Assists, xA, and Assist, Each point in this plot represents a match; the X-axis is devoted to expected assists-xA, while the Y-axis is for actual assists Ast for teams. The blue dot symbolizes Manchester City, and the red one is symbolic of Manchester United.

By the distribution of the plot, Manchester City has a wider range in terms of expected assists, with several matches showing values between 2.0 and 3.5 xA to evidence their ability to constantly create high-quality goalscoring opportunities.

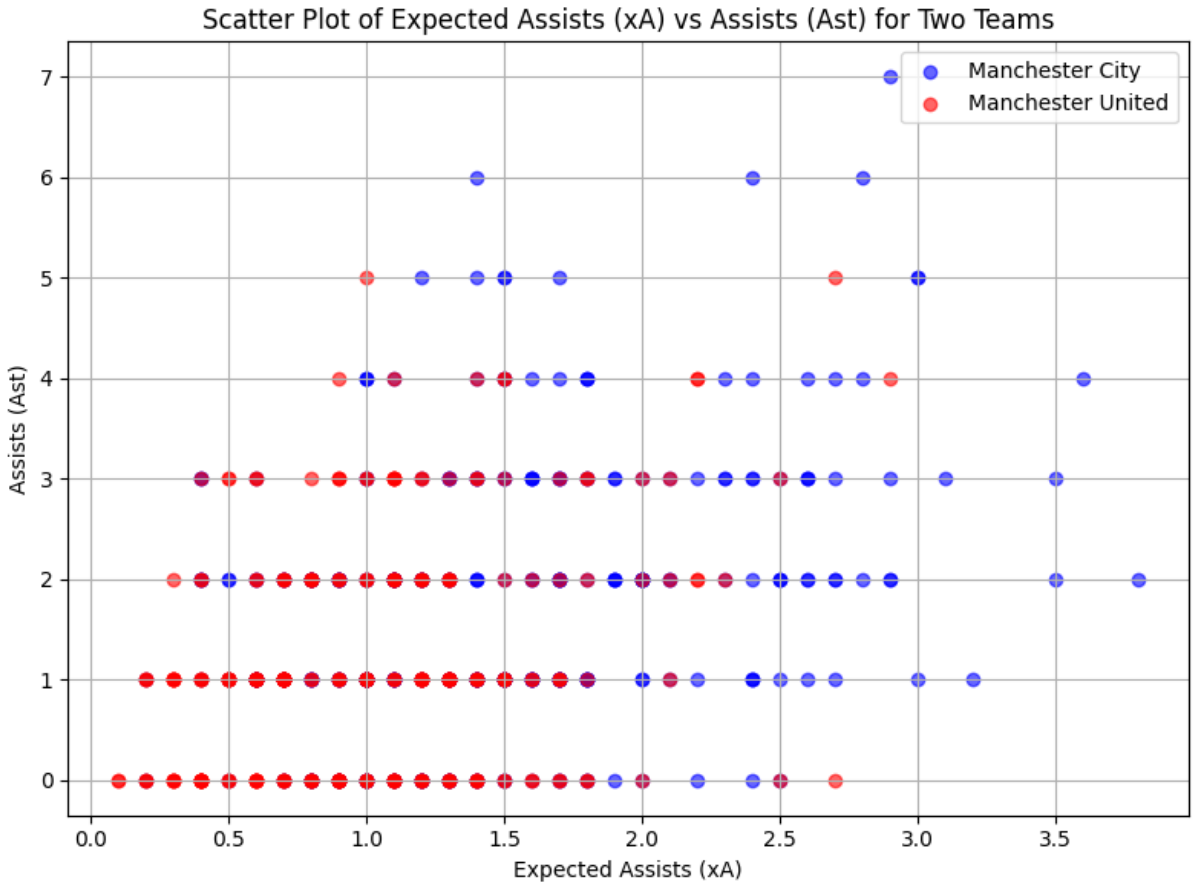

Figure 6.4: scatter plot for Manchester City vs Manchester United with (XA)

In contrast, Manchester United [29]is more concentrated toward the lower range; thus, most of their matches lie in the expected assist range of 0.0 to 1.5. While this may be so, both teams have similar spreads in actual assists after the values go up to 5 for Manchester United and up to 7 for Manchester City. This plot shows that Manchester City has a tendency to create more expected assists, but both teams have similar actual assist outcomes at the individual match level. That is to say that even though Manchester City often creates highly probable chances, Manchester United converts fewer chances into productive assists and thus posts similar assist outputs in certain matches. This plot really dramatizes how much more efficient Manchester City has been this season in their attack.

6.2.4 Selecting Manchester City Team for Key Metric Analysis

Manchester City was chosen because they have remained in the top tier for several seasons and therefore can be representative to test advanced metrics such as xG, GF, GA, Sh, SCA, and GCA. With a focus like this precisely on such a top team, there is raised insight regarding how predictive models capture even very minor causes of match outcomes and consistency in performance.Manchester City is used throughout for detailed analysis, but such an approach can be generalized to other teams when studies that involve comparison are in mind.

6.2.5 Box plot for Vital metrics of Manchester City

(a)Distribution of Goals For (GF): This boxplot shows the distribution of Goals For against Manchester City across matches. The central horizontal line inside the box shows the median, which is roughly 2 goals. In other words, considering the median, there are 50 percent of the matches below and 50 percent above when considering the amount of goals scored against Manchester City.The inter-quartile range, IQR, the middle 50 percent of data comprises the width of the box itself. Here, in most matches, Manchester City scored within a goal range of 1 to 3. Whiskers extending from the box give an indication of the range of the data. Thus, Manchester City's goals per match were in the range from about 0 to 7 goals. No outliers in the plot mean there were no matches where the number of goals scored hugely deviated from the overall pattern.

Figure 6.5: Distribution of (GF) for Man City

In other words, from the box plot[30], the distribution of goals per game was routinely between 1 and 3 for most matches and up to a maximum of 7. From this chart, it can be noticed that the goal-scoring in this Manchester City team has been pretty stable, with no outrageous outcomes relative to the goal count.

(b)Distribution of Expected Goals For (xG): The following box plot summarizes the distribution of Expected Goals for Manchester City[31]. It follows that the median is just about 2 because half of their matches had expected goals below this value, while halfway above it. IQR in the interquartile range shows that their typical xG falls between 1.5 and 3.0, while whiskers show a wider span from 1 to 4. There are a few outliers where xG reaches up to 6, showing that in a few games they have managed to create much more scoring opportunities. Overall, this plot marks a representation of consistent xG throughout with spikes of high chance creation.

(c)Distribution of Goals Against For (GA): The box plot depicts the distribution of the goals against Manchester City, "GA." The median is 1; therefore, in half of the matches, Manchester City has conceded a goal or fewer. IQR spans 0 to 1 goals against, so the middle 50 percent of matches were within that range. As seen, the whiskers extend to a maximum of 2 goals, considered normal in most matches[32]. The outliers shown in the graph represent the incidences where the team conceded goals up to a maximum of 5 goals. In general, the plot shows strong defensive performance from Manchester City because, in most of the games, the goals conceded are very few.

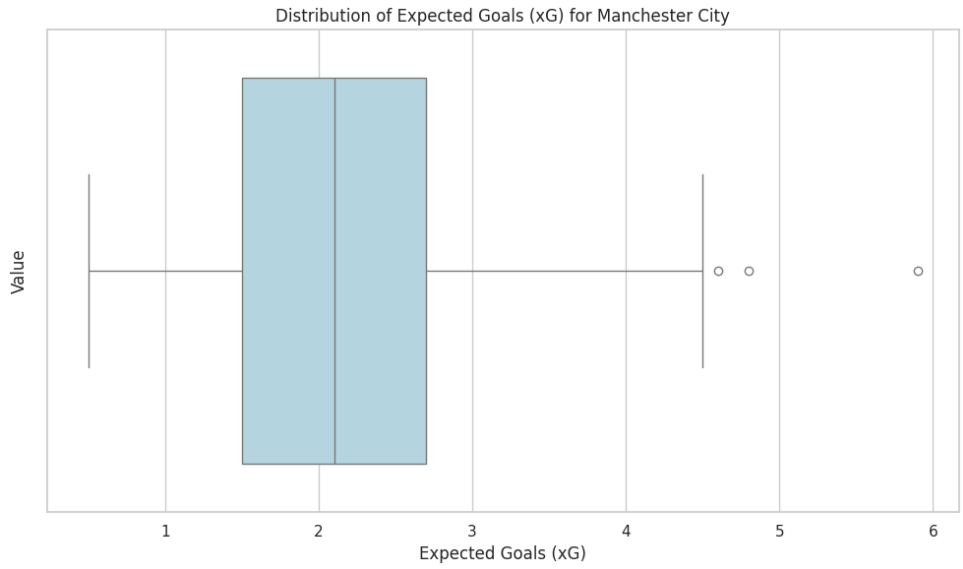

Figure 6.6: Distribution of Expected Goals For (xG) of Man City

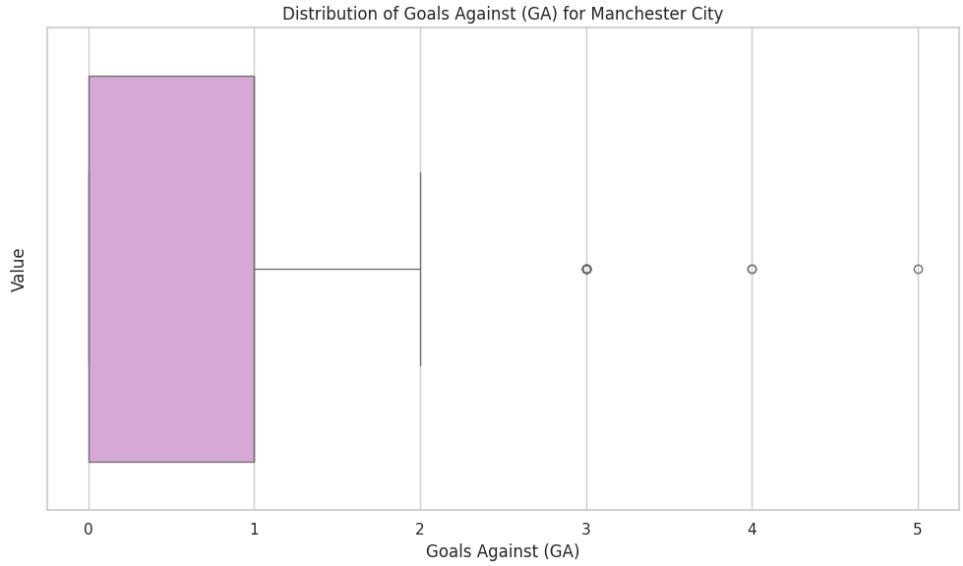

Figure 6.7: Distribution of Goals Against For (GA) of Man City

(d)Distribution of Shots (Sh) :

This box plot displays the total number of shots distribution for Manchester City. The median is approximately 17 shots, indicating that Manchester City took at least 17 shots in half of the matches; the IQR says that for the middle 50 percent of the matches the shots fell between 15 and 23, and the whiskers extend from about 9 to 28 shots, representing the range for the majority of matches. There are a couple of outliers where the team took as few as 5 shots and as many as 35 shots[33], which were very rare cases when the shot attempts would be that much lower or higher. Overall, from this plot

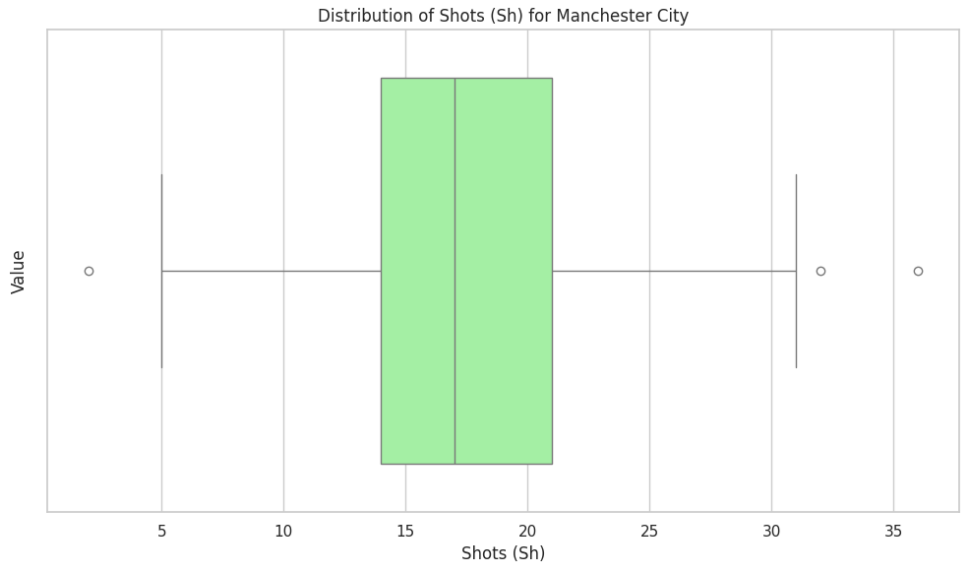

Figure 6.8: Distribution of Shots (Sh)

Manchester City takes shots that range between 15 to 23 shots per match.

(e)Distribution of Shot-Creating Actions (SCA) : This boxplot depicts the distribution of SCA for Manchester City. The median is at roughly 30 SCAs, meaning in half of their matches, Manchester City created 30 or more shot[34] opportunities. The interquartile range shows that the middle 50 percent of matches fell between 25 and 40 SCAs. Whiskers extend from approximately 10 to 50 SCAs, showing us the normal range for most matches. The only outlier is that the team made over 60 SCAs in one match, an extremely extraordinary display. This plot therefore suggests that Manchester City SCA generally lies between 25 and 40 in creating shots per match.

(f)Distribution of Goal-Creating Actions (GCA) : The below boxplot illustrates the distribution of SCAs for Manchester City. The median, at close to 30 SCAs, tells us that in half of their matches, Manchester City[35] created 30 or more shot opportunities. IQR tells that the middle 50 percent of matches fell between 25 and 40 SCAs. Whiskers extend from about 10 to 50 SCAs, representing a normal range for most matches. There is one outlier for this team when they created more than 60 SCAs in one game, which surely indicates that this game was highly exceptional. From the above plot, it can be suggested that generally, Manchester City creates between 25 and 40 shot opportunities every game

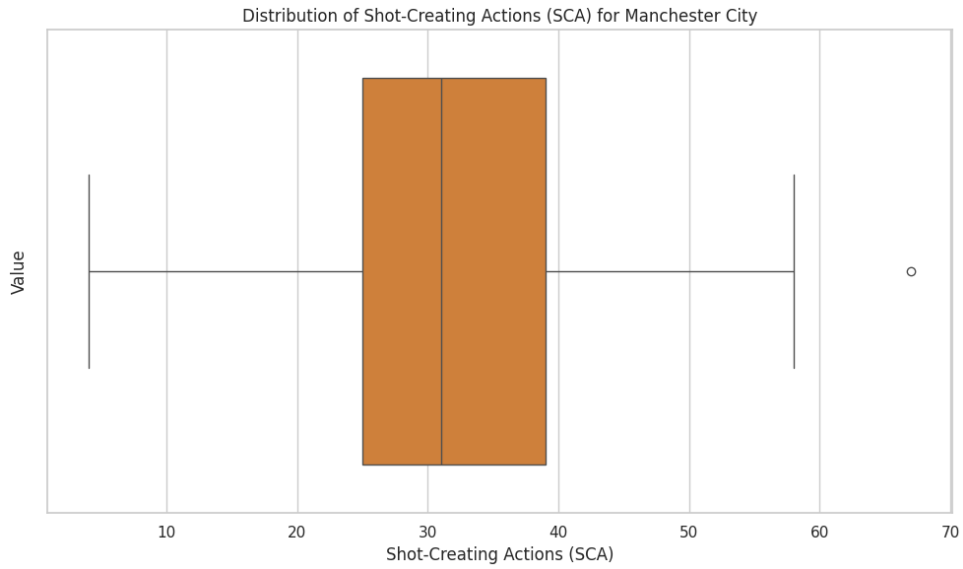

Figure 6.9: Distribution of Shot-Creating Actions (SCA)

Figure 6.10: Distribution of Goal-Creating Actions (GCA)

6.2.6 Correlation Heatmap of Football Performance Metrics

The correlation matrix reveals a fairly long list of significant relations among the key performance indicators in this football dataset. Goals For(GF), has an extremely positive relation not only with Ast,(Assists) at 0.85 but also with GCA, or Goal-Creating Actions, at 0.95, indicating that the two factors are major facilitators when it comes to scoring goals. Additionally, SoT, or Shots on Target, has a positive relationship with goals at 0.58, hence signifying that better shooting accuracy leads to increased goal counts.

On the other hand, the defensive side shows strong positive correlation of GA with xGA equal to 0.61, meaning that the number of goals conceded respects the expected defensive performance[36]. Otherwise, it is poorly correlated with other variables such as possession or shots.

Curiously, the expected goals are correlated with Shots-Sh: 0.67 and Shots on Target-SoT: 0.64, underpinning that with more shooting opportunities, the expected goals rise, whereas in reality, this correlation is a bit weaker at 0.60.

Possession, therefore, is highly positively related to Passing Completion Percentage, 0.73 and Shots 0.56, meaning that possession-based football generally promises good passing accuracy and brings more shooting chances. SCA also highly correlates with shots and goals, indicating how those actions are more important in creating scoring opportunities.

In an overall sense, the matrix suggests that the goals are explained by offensive measures such as assists, shot accuracy, and shot-creating actions, while the defensive measures do well in tracking expected goals conceded. There is also a decent contribution of possession in Goal Influence, passing accuracy, and shot creation.

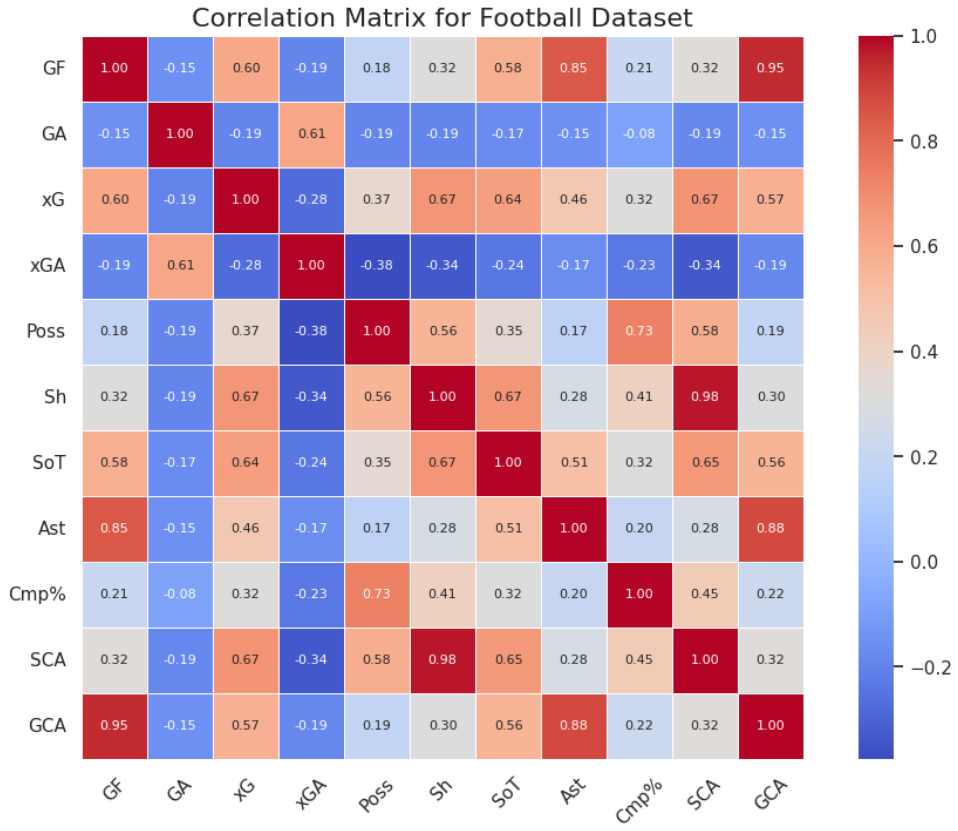

Figure 6.11: Correlation Heatmap of Football Performance Metrics

6.2.7 Density Plot of Expected Goals (xG)

This density plot provides the distribution of the Expected Goals across the dataset in high resolution, providing some important indications about the frequency of different xG values that might happen in football. The x-axis is the expected number of goals, which ranges from 0 to 7, while the y-axis shows density, and this is essentially representative of the likelihood of these xG values happening. There is a dominant peak around 1.5-that is, most matches within this dataset tend to create an expected goal value at roughly that figure. This would immediately suggest that in the average game, teams are expected to create roughly one or two clear-cut opportunities which might realistically end up as goals.

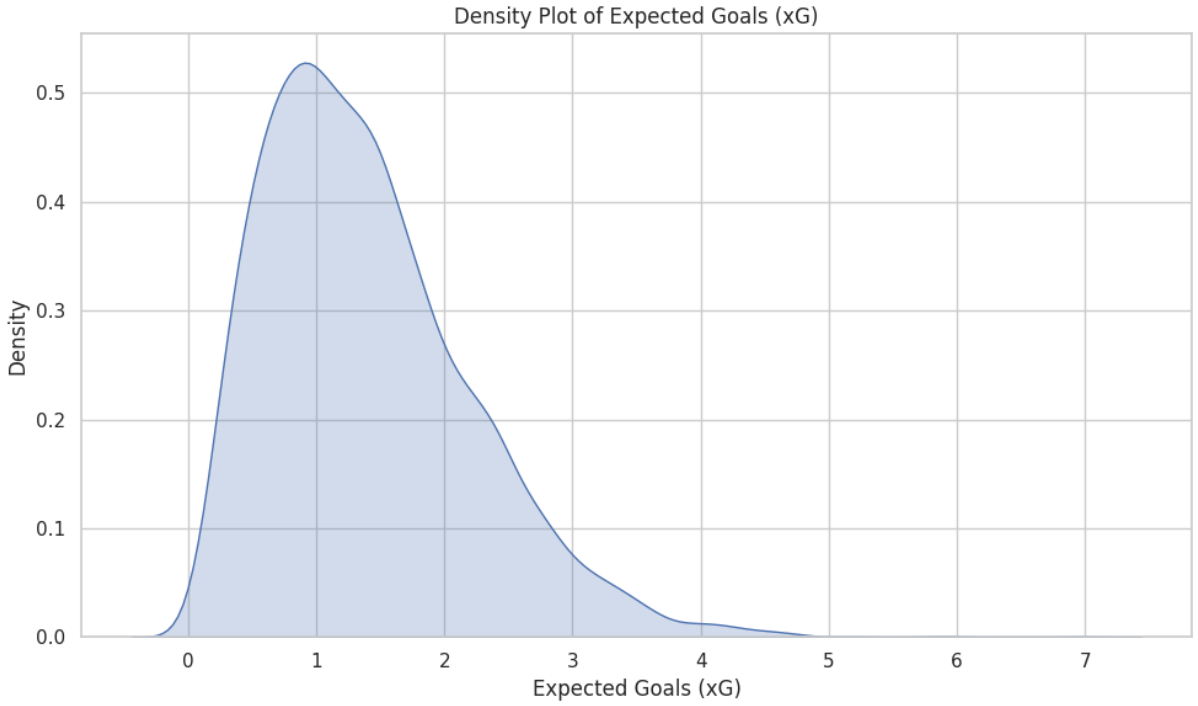

Figure 6.12: Density Plot of Expected Goals (xG)

As we progress in this direction along this axis past the peak, the plot tapers off very gradually, resulting in a right-skewed distribution. This would indicate that the majority of games have an xG value between 0 and 3, while fewer games have much higher expected goals values, ranging up to a maximum of 7. These far right outlier games[37] with high values of xG might correspond to those where teams take many goal-scoring opportunities, that is, dominant or high-scoring games. The plot would, therefore, be long-tailed right because this captures the rarer occurrence.

Furthermore, this curve is smooth and continuous, with no jumps, reflecting a natural distribution of xG. The implication here is that teams tend to be consistent from game to game in the creation of scoring chances. There are no sharp breaks or discontinuities in this plot-a fact that indicates xG values are generally similar, and the offensive capability of teams commonly does not vary wildly from game to game. Density plots indeed provide a good overall visualization of the offensive pattern in this data. Most of the matches have a modest number of expected goals, though sometimes performances do occur that are well out of the ordinary level of chance creation.

This gives a good overview of scoring expectancies across the dataset. Most of the matches have a normal distribution in expected goals centered around 1.5 and only a few matches really top the values of xG highly. With this smooth distribution, one can note that teams vary continuously in their offense, which helps to describe a variety of scoring scenarios that might happen in soccer matches.

6.3 Methodology

6.3.1 Data Preprocessing

Handling missing values:

Missing value treatment was done in an organized manner, following a series of well-defined steps to ensure that the integrity and usability aspect of the dataset is not compromised. This involved calculating the percentage of missing values against the column names, after which some columns were found to have as high as 30 percent missing data, such as xG and xGA, while some had fewer, like Attendance and Captain. The column Notes had a huge 99.97 percent missing data and thus was removed since it contributes very little value to the dataset.

For the rest of the numerical columns, their missing values have been filled using the median. The median is a robust measure and doesn't skew the data with outliers. This was an important thing to do, as columns like xG, Cmp, Dist, and many more contained crucial numerical information. Similarly, percentage-scale columns like SoT and Cmp were converted to numeric data types[38] and filled in their missing values using the median for consistency throughout the dataset.

The peculiarity of the team-specific data consisted in a very special approach-

imputation on a per-team basis was performed applying a function filling the missing value for each column with the median computed over a specific team. In this way, this method preserved team-level characteristics and trends in these data rather than imposing some kind of global imputation, which would distort individual team profiles.

The columns Time, Result, and Formation were categorical; these were encoded into numerical values using LabelEncoder in order to make the dataset suitable for machine learning models. Then, encoding done, the rest of the rows with missing values dropped out to make sure that there is not a single null value in the final dataset. The 'Attendance' column was considered as the count in numerical, hence its median was taken to impute after converting all non-numerical entries in it to NaN.

A verification was done at the end, and there was no lost value left behind. The cleaned dataset now has dimensions 5150 rows,58 columns and ready for analysis or model development. Such an approach in handling missing values kept the data as complete as possible, its structure intact, while preserving important subtleties native to specific teams.

Attribute	**Description**
Number of Rows	5150
Number of Columns	58

Table 6.1: Dataset Dimensions After Cleaning

Labeling:

n the original dataset, each match had a multi-class result such as "Win", "Loss", and, less frequently, possibly "Draw". For this problem, the outcome column was pre-processed to contain a binary format of two different classes of outcomes

Here, a Win was labeled as 1, and any other result was labeled as 0, (that includes loss and draw). Doing so transformed the problem into a binary classification[39] problem with only two possibilities: to win or loss(Includes loss and draw). Distribution post-transformation resulted in roughly 60 percent loss and 40 percent win matches. This binary labeling is crucial to a plethora of machine learning models interested in the outcome of a match.

- Win was labeled as 1.

- Loss and Draw were combined and labeled as 0.

Outcome	Class Label	Percentage (%)
Win	1	40%
Loss/Draw	0	60%

Table 6.2: Outcome distribution after binary labeling

Feature Engineering

Feature engineering was used in an attempt to get more insightful information from the dataset to try and improve the modeling of predictions. This was done through:

- **Sorting of Matches Chronologically:** Since the data was ongoing, chronological sorting was an essential criterion to ensure that no information got lost or altered. The sorting was done based on team, season, date, and time of play to extract the features in a logical and meaningful sequential order sketching the performance of each team over time.

- **Rolling Window Feature Extraction:** One of the most important steps in feature engineering was rolling a window for every team with respect to their last 10 matches. In this method, the performance statistics of each team against its last 10 matches come up with a set of features using which the next match can be predicted. The idea here rests on the assumption that recent performances are indicative of future outcomes.

- **Combining Features into a Structured Dataset**: goals, xG, Cmp, among others, from each team's last 10 matches. If a team is without 10 available matches, it is padded with missing values so that all matchups between teams are of the same length in terms of features. Combining the Features into a Structured Dataset: Once the available features were extracted from past matches, they were compiled into a new dataset. Each feature was tagged appropriately as to which match it belonged, e.g., first, second, or tenth previous match goals scored. This was an exhaustive feature set that comprised recent performance history for any given team. The new dataset also included the binary outcome of win and loss as variables used for the target in training the model.

6.4 List of Features used

These features were selected because they are key aspects of a football team's performance, which allows us the comprehensive analysis and prediction of match outcomes.

(**xG**, **xGA**, **Dist**, **FK**, **npxG**, **Cmp**, **Att**, **PrgDist**, **Ast**, **xAG**, **xA**, **KP**, **PPA**, **CrsPA**, **PrgP**, **SCA**, **SCA_PassLive**, **SCA_PassDead**, **SCA_TO**, **SCA_Sh**, **SCA_Fld**, **SCA_Def**, **GCA**, **GCA_PassLive**, **GCA_PassDead**, **GCA_TO**, **GCA_Sh**, **GCA_Fld**, **GCA_Def**, **SoT%**, **Cmp%**, **Time**, **Result**, **Poss**, **Formation**)

Missing Value Handling: Following feature vector construction, removal of all rows containing missing values was done to ensure a resultant clean dataset that was ready for modeling. This makes sure that all the data passed into the model was complete and consistent. The final dataset contained 4790 rows and 351 columns. Each row represented a match, while the columns contained the performance statistics for the two teams participating in each match for the previous 10 matches. The binary target variable was also part of the dataset and thus became the label to be predicted.

Attribute	Count	Description
Rows (Matches)	4790	Each row represents one football match.
Columns (Features)	351	The columns include performance statistics of two teams over the last 10 matches.
Target Variable	1	Binary label (1 or 0) indicating the match outcome (win/loss).
Total Columns (Including Target)	352	All feature columns plus the binary target variable.
Missing Data Handling	-	Rows containing missing values were removed to ensure a clean dataset.

Table 6.3: Summary of Final Dataset Used for Model

Outcome Distribution: After preprocessing, the target variable represented a slight imbalance between the two outcomes:

Of these, 2857 matches (60 percent) were classified as losses, and The wins were modeled as 1933 matches, which is 40 percent. This may suggest that in this data, there is an imbalance, and though the model performed well here, such imbalance might

affect performance, maybe needing further adjustments such as balancing techniques in the course of modeling.

The feature engineering [40]process delivered a robust set of features based on the last 10 matches of a team, which are important for building predictive models. The final data in its temporal nature for match outcomes is now suitable for machine learning tasks where the goal is to predict, based on recent performance, a team's win or loss in the next match.

6.4.1 Development and Training of Model

- **Balancing Dataset:** First, it starts to balance the two classes, which are wins, the positive class and losses/draw, the negative class. This is important because it would avoid an imbalanced dataset. On the other hand, such a problem would imply the bias of the model toward the majority class, in this case, the losses. It will split a tiny portion; Test size = 0.001-that means it's going to retain most of its positive class samples. In the case of the negative class-loss, the contribution discards 30 percent of the samples to retain 70 percent with the intention of reducing dominance in the dataset. Upon processing both classes, the positive and negative samples are merged to form a balanced dataset[48]. This will then ensure that the model gets an equal chance of learning from both wins and losses without getting biased on the high-frequency class. After that, the order in the balanced dataset is shuffled to avoid any conceivable patterns.

- **Splitting Data:** This dataset is further divided into training and test sets after thorough balancing of the data. In fact, the training set will be employed for the fitting of machine learning models[42], while the rest will be used as a test set for them. It is necessary to split the data to have an idea of how well the model generalizes on unseen data. That being said, we usually consider an 80-20 split by putting 80 percent for training and the rest for testing.

- **Normalization of Features:** Feature normalization used StandardScaler that performs Z-score normalization. This is an important step in SVM algorithm, since these models are sensitive to the scale of input features[43]. During this process, the training data is normalized in order to scale the features to zero mean and unit

standard deviation. The test data is then transformed using parameters-mean and standard deviation-derived from the training set for consistency. This ensures that all the features contribute equally in the model and no single feature can drive the outcome disproportionately.

- **Training and Evaluating three Models:** The machine learning models will be trained and tested in order to find out which one gives the best result from data. These include Random Forest, Support Vector Machines, and XGBoost algorithms. In fitting each of these models[44], scaled training data is used. All of these models were used to make the predictions on the test data upon training of each of them. It calculates the different metrics of performance regarding the capability of a model to predict match outcomes, such as accuracy, precision, recall, and F1-score. These metrics basically show how well the models are correctly balanced between wins and losses in predictions and how well they avoid false positives and negatives.

6.4.2 Machine Learning models employed

Random Forest Classifier: we have an ensemble learning method called the Random Forest classifier, which constructs multiple decision trees during training and predicts the mode of the candidates obtained from the trees for classification tasks. Such a method normally improves the model on accuracy and stability, especially when compared with single decision trees. Every Tree in the random forest is trained on a random sub sample of the training data-trained via bagging, and each internal node of the tree considers a random subset of the features.

Mathematically, the prediction for a Random Forest model[45] is given by:

$$\hat{y} = \text{argmax}_c \sum_{i=1}^{B} \mathbb{I}(\hat{y}_i = c)$$

where:

- $\hat{y}_i$ is the prediction of the i-th decision tree,
- B is the total number of decision trees in the forest,
- $\mathbb{I}(\cdot)$ is the indicator function, which equals 1 if the argument is true and 0 otherwise,
- c is the class label (e.g., win or loss).

The Random Forest classifier averages out the results over a large number of trees, it greatly reduces variance in the predictions, hence reducing the risk of overfitting that might be inherent in individual decision trees. This technique is fairly robust and works well on high-dimensional data or when there are complex variable interactions within a dataset.

Support Vector Classifier: SVC stands for Support Vector Classifier, which is a pretty powerful and robust classification algorithm under the SVM[46] framework. Generally used in binary classification but can also be extended to multiclass problems. The support Vector Classifier basically works by finding out this hyperplane like such that it maximally separates these two classes in higher multidimensional feature space. The algorithm works to optimize the margin, which is the distance of this hyperplane from the nearest data points of both classes. These nearest points, called support vectors, form a crucial part in describing the decision boundary.

In Support Vector Classifier, the decision boundary is defined by a hyperplane that can be mathematically expressed as:

$$w^T X + b = 0$$

where:

- w is the weight vector that determines the orientation of the hyperplane,
- X is the input feature vector,
- b is the bias term, which shifts the hyperplane from the origin.

The Support Vector Classifier is a versatile algorithm that works on the basis of finding the optimal hyperplane, merely to separate the classes and keep the margin maximum so that it will be robust as well as generalized. Its capability to deal with both linear and nonlinear classification problems, along with its resistance against overfitting, makes it highly applicable in several classifications.

XGBoost Classifier: XGBoost or extreme gradient boosting is the efficient and scalable implementation of the gradient boosting framework. Given the great performance on structure data, it has grown as one of the most popular machine learning algorithms. Due to many reasons such as regularization, optimization of computational resources,

handling of sparse data, and so on, XGBoost[47] is preferred very much in competitions and real applications.

The Objective function for XGBoost includes loss function and regularization terms to penalize the model:

$$L(\theta) = \sum_i L(y_i, \hat{y}_i) + \sum_k \Omega(f_k)$$

where:

- $L(y_i, \hat{y}_i)$ is the loss function that measures the difference between the true label y_i and the predicted value $\hat{y}_i$,
- $\Omega(f_k)$ is the regularization term that penalizes model complexity,
- f_k is the prediction from the k-th tree.

The first term $\sum_i L(y_i, \hat{y}_i)$ ensures that the model minimizes the difference between actual and predicted values, while the second term $\sum_k \Omega(f_k)$ regularizes the model to prevent overfitting by penalizing complex trees.

XGBoost is one of the most efficient and scalable, from the gradient boosting family, machine learning algorithms. Its usage of regularization, parallel computation, handling of missing values among others-made it one of the most powerful and flexible tools in classification and regression problems.

Logistic Regression: Logistic Regression is one of the most well-liked machine learning algorithms, finding its application for binary classification problems. In logistic regression, the modeling is carried out on the relationship of a dependent binary outcome variable-usually coded as 0 and 1-with one or more independent predictor variables. The goal here is to find the probability that an instance belongs to a particular class: for example, "Win" versus "Not Win".

Unlike linear regression, which was used for continuous outcomes, logistic regression is designed for categorical outcomes in which the dependent variable represents discrete categories.

$$P(y = 1 \mid X) = \frac{1}{1 + e^{-(\beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_n X_n)}}$$

Where:

- $P(y = 1 \mid X)$ is the predicted probability that the output is 1 given the feature set X,
- β_0 is the intercept (constant term),
- $\beta_1, \beta_2, \ldots, \beta_n$ are the coefficients of the respective features $X_1, X_2, \ldots, X_n$, representing the weights or importance of each feature in predicting the output,
- e is the base of the natural logarithm.

6.4.3 SMOTE(Synthetic Minority Over-sampling Technique)

SMOTE is one of the very popular oversampling techniques used, especially when dealing with an imbalanced dataset. With respect to classification tasks in machine learning, there exist one class in most cases that is not as proportionate as the others in relation to the number of instances in the class; hence, it is called the minority class.

In SMOTE[48], new synthetic samples are generated for the minority class by interpolating between a minority class sample x_i and one of its nearest neighbors x_{nn}. The formula for generating a synthetic sample x_{syn} is given by:

$$x_{syn} = x_i + \lambda \cdot (x_{nn} - x_i)$$

where:

- x_i is a sample from the minority class.
- x_{nn} is one of the k-nearest neighbors of x_i.
- $\lambda \in [0, 1]$ is a random number that determines how far along the line between x_i and x_{nn} the synthetic sample will be generated.

This formula creates synthetic data points that lie on the line segment connecting x_i and x_{nn}, helping to balance the class distribution.

Undersampling:

Undersampling applied in imbalance datasets aims to reduce sample size in the majority class, but in contrast to oversampling, the technique does not depend on the addition of new or synthetic data within the minority class. It mainly concentrates on the removal of examples from the majority classâconsequently, balancing a dataset. This

is especially useful when one is dealing with large datasets, and the elimination of some instances of the majority class does not go to a great deal of reducing the information content.

Undersampling reduces the number of samples from the majority class X_M to match the size of the minority class X_m. The number of samples in the majority class after undersampling X'_M is given by:

$$|X'_M| = |X_m|$$

where:

- $|X'_M|$ is the new size of the majority class after undersampling.
- $|X_m|$ is the size of the minority class.

This technique helps to balance the dataset, allowing the classifier to give equal importance to both classes.

Oversampling:

oversampling works by augmenting the minority class to a size either equal to or near that of the majority class. It can be performed simply through the duplication of the existing minority class samples or the random generation of new artificial examples, as in techniques such as SMOTE.

In other words, the idea of oversampling involves a balance where the minority class shall increase without the loss of any information from the majority class. This would make the classifier give equal significance to both classes during training. This will ensure that bigger patterns are learned from the huge number of examples in the minority class, which again leads to the improvement of performance in the model for the predictions in the minority class. However, oversampling increases the risk of overfitting, especially if the duplicated or synthetically generated samples do not largely represent the true diversity of the minority class. Despite this fact, it is one of the primary methods through which class imbalance in machine learning is addressed to improve model accuracy and generalization.

In oversampling, the number of samples in the minority class X_m is increased to match the size of the majority class X_M. The new size of the minority class after oversampling X'_m is given by:

$$|X'_m| = |X_M|$$

where:

- $|X'_m|$ is the size of the minority class after oversampling.
- $|X_M|$ is the size of the majority class.

To achieve this, the number of new samples added to the minority class is:

$$\Delta = |X_M| - |X_m|$$

where Δ is the number of samples added to balance the dataset.

ROC Curve (Receiver Operating Characteristic:)

The ROC is a plot that pictorially summarizes the performance of a binary classifier. It plots the True Positive Rate (TPR), or sensitivity, vs. the False Positive Rate (FPR), which happens at different threshold settings. The curve helps in making understandable the trade-offs between correct identification of true positives and incorrect identification of the negatives as positives.

The ROC (Receiver Operating Characteristic)[49] curve is used to evaluate the performance of binary classification models by plotting the **True Positive Rate (TPR)** against the **False Positive Rate (FPR)** at different threshold values.

6.4.4 True Positive Rate (TPR) or Sensitivity

The TPR, also known as sensitivity, is the proportion of actual positives that are correctly identified by the model. It is calculated as:

$$\text{TPR} = \frac{TP}{TP + FN}$$

Where:

- TP = True Positives (correctly classified positive instances)
- FN = False Negatives (positive instances incorrectly classified as negative)

6.4.5 False Positive Rate (FPR)

The FPR represents the proportion of actual negatives that are incorrectly classified as positive by the model. It is calculated as:

$$\text{FPR} = \frac{FP}{FP + TN}$$

Where:

- FP = False Positives (negative instances incorrectly classified as positive)
- TN = True Negatives (correctly classified negative instances)

6.4.6 True Negative Rate (TNR) or Specificity

The TNR, also known as specificity, is the proportion of actual negatives that are correctly identified by the model. It is calculated as:

$$\text{TNR} = \frac{TN}{TN + FP} = 1 - \text{FPR}$$

6.4.7 Area Under the Curve (AUC)

The AUC is the area under the ROC curve, which quantifies the overall performance of the classifier. A perfect classifier has an AUC of 1.0, while a classifier that performs no better than random guessing has an AUC of 0.5.

6.5 Description of the Second Prediction Model

The second predictive model analyzes historical football performance data in order to predict how teams will perform during certain seasons. This model extracts the feature of past seasons for each team and then labels them to indicate good/winning or poor/losing seasons for the teams. That is explained further below.

Arranging and sorting the historical data: First, it sorts the dataset in ascending order by team, season, date, and time. The performance of a team usually emerges well if data is observed chronologically. Sorting will enable the model to capture features sequentially from past seasons and then apply them in order.

Historical Feature Extraction: It pulls out performance data from the previous seasons for each team. Each team, in every season, would collect 20 historical records of matches. Extracted features emphasize numeric and percentage statistics related to goals scored, shots on target[50], possession percentage, among other relevant match-related metrics. These are extracted to make the core dataset, providing a representation of the historical performance of each team. The features extracted for each team-season give the model a proper view of recent history for a team, something bound to be invaluable when it comes to predicting future performance.

Labeling Winning Seasons: Normally, a target variable is created from the given list of historical winning teams and their seasons for the model. Based on those provided team and season labels in every case, the model checks if the team won in the current season. A winning team in a season gets a label of 1, while any other team gets a label of 0. Such binary labeling gives clear indications of successful and unsuccessful seasons that will enable this model to predict team success in future seasons.

These labels will be useful in training machine learning classification models to ascertain whether any given team-season combination is a winner or not, considering past performance.

Preparing Feature Vectors: Feature and label extraction is then followed by filtering out the non-essential columns that represent the identifiers of teams and seasons to continue further with those columns that correspond to performance data. So, the feature vectors representing historical match statistics are prepared now to be used by the prediction model. This ensures that, instead of metadata or identifiers, the actual performance data is under consideration by the machine learning model.

Dataset Dimensions: The final dataset includes a number of feature columns of historical performance data and a label indicating whether a team had a winning season. The dataset has been prepared in such a way that any machine learning model can efficiently train and test for any prediction task, the dimension of dataset(110,662). This means the dataset consist of 110 observations and 662 features for each observation.

6.5.1 Model Training and Evaluation for second prediction

Splitting the Dataset:

It then splits the 'Dataset' into training and testing sets using traintestsplit from

sklearn.modelselection. This data shall be divided so that 80 percent of it can be used in training the model, while the remaining 20 percent will be used for testing; stratified based on the 'labels' such that the proportion of the class labels, being those belonging to the winning and non-winning seasons in the training and testing data, respectively, is preserved.

This step ensures that both the training and testing datasets are representative of the overall class distribution, which is a necessary condition for a model's evaluation.

Feature Normalization:

These training and test features are then normalized using StandardScaler by applying Z-score normalization to the samples. This transformation helps the features have a zero mean and unit variance, which makes convergence easy and enables machine learning models to perform optimally.

- **Normalization of Training Data:** This fits the scaler to the training data and then transforms the training features into their standardized values. That way, it ensures the model is trained based on the standardized data.

- **Scaling of Test Data:** After that, test data is scaled using the same parameters obtained from the training data. This normalization step is very important for models like RandomForestClassifier, which put the features on the same scale so that no feature disproportionally influences any other feature in the model.

Model Training: We instantiate a RandomForestClassifier from sklearn.ensemble and then go ahead and train it with our scaled training data. The intuition behind Random Forest is that this is an ensemble model: multiple decision trees are constructed before having their predictions combined so as not to overfit based on just one model. Besides Random Forest, we have used two other models for comparison: XGBClassifier from the xgboost library, using gradient boosting in classification problems, and Logistic Regression from the sklearn.linearmodel[52] library as an example linear model used to solve binary classification. These have been applied here because while serving the same purpose, each has some particular strengths that may be possessed by the data due to its unique characteristic.

Making Predictions: After training, the model calls predict on the scaled test data-Xtestscaled to make predictions. These are the model's[51] classification for whether a

team had a winning season or not based on the features provided.

Accuracy: The proportion of predictions that the model got right from the test data. It gives a general feel for the remaining performance measures as it refers to the model performing well.

Classification Report: The precision, recall, and F1-score are all included here for winning and non-winning seasons, making for a deep drill-down into the model's performance in terms of handling respective positive and negative predictions.

These are metrics that get printed out in order to assess how well the model is performing, giving insight into strengths and weaknesses in the predictions of teams' performances.

CHAPTER 7

Discussion and Results

7.1 Comparative Analysis of Machine Learning for Model-1

(a)Results of RandomForest Algorithm:

The Random Forest algorithm were made to football match outcome prediction and was performance-tested on three variants of datasets: an imbalanced original dataset[53], an under-sampled dataset, and a SMOTE over-sampled dataset. This summarizes and interprets the performance of a Random Forest-based algorithm on standard metrics like accuracy, precision, recall, F1-score, and AUC.

Unbalanced: The Accuracy of the model is 58.64 percent, precision 56.32 percent, and recall 69.57 percent. It would appear that this shows a moderately performing model: the balanced F1 is 61.61 precent and the AUC is 0.6200, meaning that under these conditions, it has limited abilities to tell wins from losses.

Undersampled: Accuracy is very low, lying at 56.00 percent, while precision is better at 65.00 percent, though the recall is lower at 58.00 precent. That means this model was better at not picking those false positives but missed more actual wins. The F1 score was once more quite well at 61.00 percent, while the AUC went down to 0.5900.

SMOTE Oversampled: The model did much better this time, improving the accuracy to 63.00 percent, precision to 66.00 percent, and recall to 77.00 percent. Since the

F1 score was the maximum at 71.00 percent, it reflected the improved win predictions. AUC value remained the same as 0.6000, indicating limited improvement in the model's discrimination capability between match outcomes.

Table 7.1: Comparison of Random Forest Performance on Different Datasets

Data Type	Accuracy	Precision	Recall	F1 Score	AUC
Unbalanced	0.5864	0.5632	0.6957	0.6161	0.6200
Undersample	0.5600	0.6500	0.5800	0.6100	0.5900
SMOTE Oversample	0.6300	0.6600	0.7700	0.7100	0.6000

(b)Results of SVC(Support Vector Classifier) Algorithm:

Below is the application of the SVC algorithm[54] on the football match outcome for the three different datasets: unbalanced, undersampled, and SMOTE oversampled. Performance evaluation metrics were accuracy, precision, recall, F1 score, and AUC.

Unbalanced dataset: It was able to achieve an accuracy of 60.03 percent, which is a good performance considering the original dataset. Precision was 57.00 percent, and thus, by that measure, the model was moderately good at predicting wins without overpredicting them, actually. The recall, on the other hand, was 72.00 percent, which showed that 72 percent of all the real estates which were being sold for a win were correctly identified by the model. Therefore, the F1-score was 63.00 percent and reflected a reasonable balance between precision and recall. An AUC score of 0.6200 reflects that model performance in distinguishing wins versus losses is only a little better than random guessing and is still not highly discriminatory.

Undersampled dataset: The accuracy went down to 55.00 percent, reflecting that it is a great challenge indeed to learn from a dataset where such a reduction in samples of the majority class has taken place. Even so, precision improved to 64.00 percent, showing that the model could avoid false positives even more effectively. However, recall decreased to 55.00 percent, showing that the model failed to capture a high number of true wins compared to the unbalanced dataset. The F1 score also fell to 59.00 percent, indicating the trade-off created by improved precision with lower recall. The AUC score fell back to 0.5700, suggesting that after undersampling, the model weakened in terms of distinguishing between wins and losses.

The SVC algorithm returned the best results with the SMOTE oversampled dataset.

Oversample: The model was able to yield the highest accuracy of the three at 62.00 percent. Precision went up to 65.00 percent, as the model did not overpredict and identified more wins than losses quite rightly. Recall marked the largest improvement out of any within the models in this chapter, reaching a high of 77.00 percent, where the model was able to identify 77 percent of all actual wins that took place. F1 was the highest at 71.00 percent, which is relatively good because the balance between precision and recall is reflected here. Even with these improvements, the AUC did not improve from 0.6000, meaning the model was still quite limited in its capability to discriminate between wins and losses. On the short side, the best performance of the SVC model in this dataset is given when the dataset is over-sampled through SMOTE. The biggest improvements were related to the recall and F1 score, while unbalanced and under-sampled datasets provided lower overall performances.

Table 7.2: Comparison of SVC Performance on Different Datasets

Data Type	Accuracy	Precision	Recall	F1 Score	AUC
Unbalanced	0.6003	0.5700	0.7200	0.6300	0.6200
Undersample	0.5500	0.6400	0.5500	0.5900	0.5700
SMOTE Oversample	0.6200	0.6500	0.7700	0.7100	0.6000

(c)Results of XG Boost Algorithm:

The model performance of the XGBoost algorithm[55] in predicting outcomes of football matches has been compared. The model performance was assessed on five key metrics, namely, Accuracy, Precision, Recall, F1 Score, and AUC- Area Under the Curve.

XGBoost predicted football match outcomes on three types of datasets: unbalanced, undersample, and SMOTE oversample.

Unbalanced: The model performance tested on the unbalanced dataset turned out to be moderate, with a testing accuracy of 58.18 percent. The precision was 56.00 percent, hence the model had been reasonably effective in its win predictions and did not overpredict wins. The recall was 65.00 percent, meaning that the model correctly predicted 65 percent of the actual wins. The F1 score is 60.00 percent, showing an area of middle balance between precision and recall. An AUC score of 59.00 percent suggests

that no more than a little over random guessing was performed by the model in telling a win from a loss.

Undersample: This gave an accuracy of 57.00 percent when applied on the underside-sampled dataset. It means the model performed worse than when the dataset was unbalanced. Whereas the precision increased to 65.00 percent, which means it improved in terms of not predicting a win where it was actually a loss; however, recall came down to 51.00 percent, which refers to the fact that quite a significant portion of actual win conditions was missed by the model. The F1 score came down to 49.00 percent, reflecting greater imbalance between precision and recall, with recall suffering particularly in this undersampled dataset. The AUC score also declined slightly to 57.00 percent, showing that this model still performs worse in differentiating between wins and losses with undersampling.

Oversample: The best performance from the models was realized using the SMOTE-oversampled dataset[56], which gave an accuracy of 61.00 percent, the highest among the three datasets. The precision further increased to 62.00 percent, indicating that more correct win predictions had been made without overpredicting wins. Recall increased to 75.00 percent, which is indicative that the model accurately predicted the actual win 75 percent of the time. The F1 score increased to 69.00 percent, showing a better balance between precision and recall with SMOTE oversampling. Curiously, the AUC did not budge from 57.00 percent , and that is indicative of the fact that even as the model improved in precision and recall, it still remained quite limited in distinguishing between wins and losses. Therefore, it can be summed up that the proposed XGBoost algorithm, on the SMOTE oversampled dataset, has the best performance considering both precision and recall, as well as the F1 score. A rather modest performance of this model is on the unbalanced dataset, whereas in the case of the undersampling method, a significant reduction of recall and F1 score happens because of the difficulty of learning from fewer samples of the majority class.

Table 7.3: Comparison of XGBoost Performance on Different Datasets

Data Type	Accuracy	Precision	Recall	F1 Score	AUC
Unbalanced	0.5818	0.5600	0.6500	0.6000	0.5900
Undersample	0.5700	0.6500	0.5100	0.4900	0.5700
SMOTE Oversample	0.6100	0.6200	0.7500	0.6900	0.5700

7.2 Area under the ROC Curve for different data types

7.2.1 AU ROC for Unbalanced Data

AUC values show the general performance of each model. RandomForest and SVC have very similar performances; both reach 0.62 AUC, which shows the predictive power maintains middle status. XGBoost reaches an [63]AUC of 0.59, which is a bit worse in distinguishing between the positive and negative classes (win/loss).

The ROC curve is a useful way to visualize model performance across different threshold values. A higher curve generally means better performance; hence, both RandomForest and SVC are performing better than XGBoost based on the AUC scores.

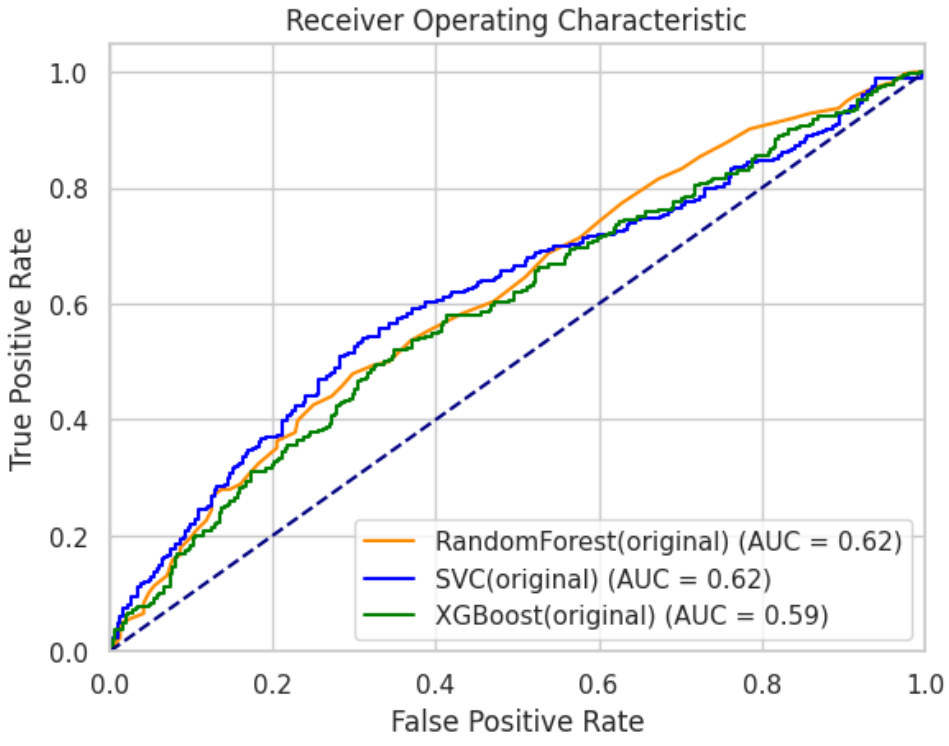

Figure 7.1: Comparing AU ROC for Imbalanced Data

This ROC analysis in fact underlines that while both RandomForest and SVC are showing a moderate performance, XGBoost was more struggling in distinguishing effectively between the target classes hence its lower AUC score. The ROC curve below shows a comparison of the three models: RandomForest, SVC, and XGBoost. Both RandomForest and SVC have an AUC of 0.62, indicating a moderate score, reflecting the same amount of separation between wins and losses. XGBoost, at 0.59, did relatively worse, with its classes not as differentiated. Overall, both RandomForest and SVC proved a bit stronger for the given analysis compared to XGBoost.

7.2.2 AU ROC for Undersampled Data

The ROC curve for three models: RandomForest, SVC, and XGBoost on an undersampled dataset. An ROC curve is a plot of various TPR and FPR values for different thresholds that measure performance in distinguishing between two classes. In the case of RandomForest, its performance was mediocre with a 0.59 AUC, as it failed to draw a proper line between classes. The lowest AUC value was recorded for the model SVC, where the value equaled 0.57. This model demonstrates very poor performance in correctly identifying wins and losses. One can notice that the XGBoost model provides an AUC of 0.59, similar to that of RandomForest and displaying a moderate classification ability.

ROC curves for all three models with performance close to the diagonal line point to their overall limited discrimination capability between the positive and negative classes on the undersampled dataset. In general, it seems that the application of undersampling-which reduces the majority class in order to balance the dataset-had a negative effect on the overall performance, only yielding moderate predictive power.

7.2.3 AU ROC for SMOTE Oversampled Data

The ROC curves for the three models, namely RandomForest, SVC, and XGBoost, all done with an oversampled dataset. It plots the ROC curve, which essentially is a measure of TPR versus FPR at various thresholds, to identify the capability of the models in differentiating between a win/loss.

With the RandomForest model, the AUC is 0.60, showing moderate performance. The curve also was for the large part above the line which follows the 45-degree angle,

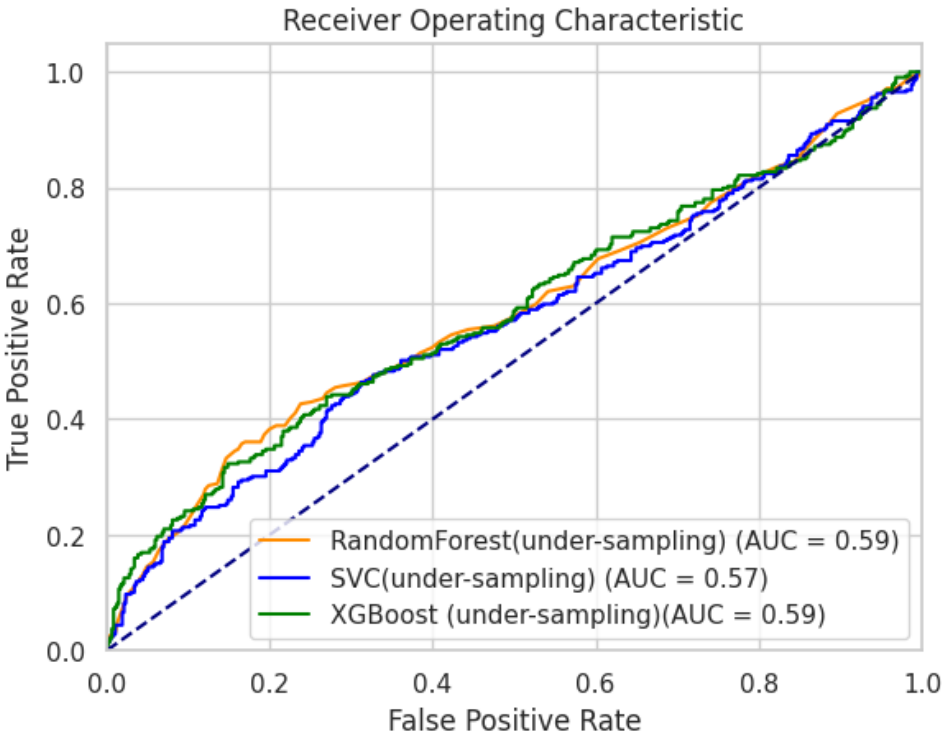

Figure 7.2: Comparing AU ROC for Underbalanced Data

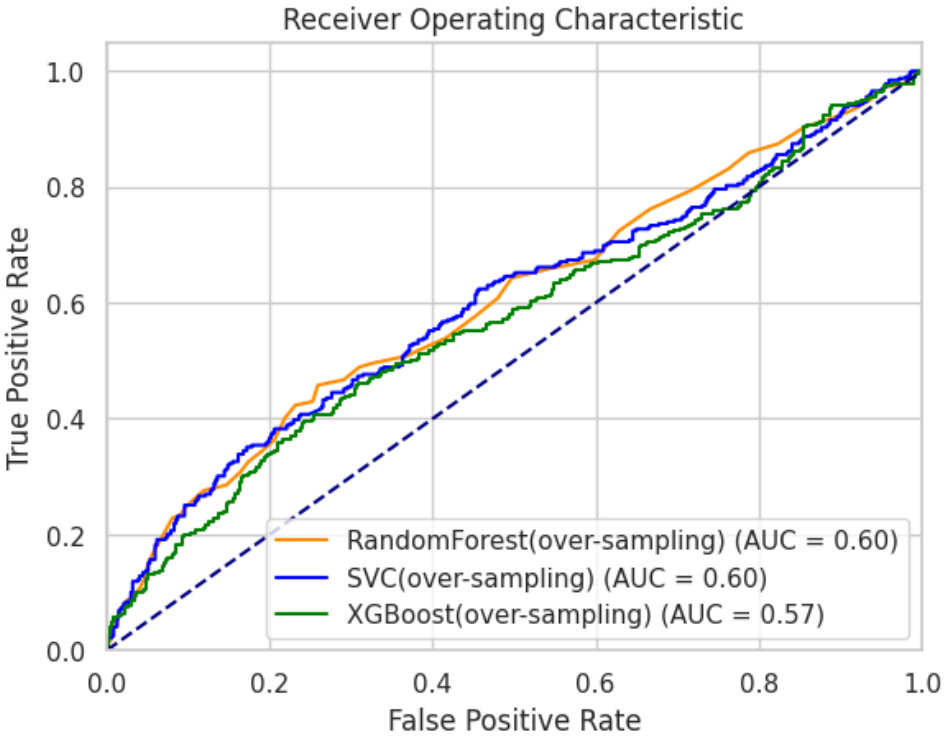

Figure 7.3: Comparing AU ROC for Oversampled Data

ensuring that RandomForest had a good performance in distinguishing between the two classes moderately. At this point, the SVC model also had an AUC of 0.60 and performed equally compared to the RandomForest in distinguishing between wins and losses.

While the performance was a little worse at the XGBoost model, it yielded an AUC of 0.57, and its ROC was closer to the diagonal line, which meant that, compared to the other two, it had poorer classification power.

In short, RandomForest and SVC were on a par with similar performances at the end, that were average, with an AUC value of 0.60, while XGBoost was worse off and obtained an AUC of 0.57. The oversampling method, therefore, helped all these models go from a low to a moderate predictive capability level; within those, however, the main one which suffered more in telling the difference about the outcome of a match was mainly XGBoost.

7.3 Discussions and Results for model 2

Model-2 was developed to predict whether a football team would finish at the top of the league, based on performance metrics of previous seasons. Towards this end, the study made use of three machine learning models: Random Forest, XGBoost, and Logistic Regression. Each model should be assessed for how well it predicts winning seasons, which shall be done using key performance metrics like accuracy, precision, and recall.

Below is a comparison of the performances between three machine learning models, namely Random Forest, XGBoost, and Logistic Regression, applied to predict whether a football team is going to have a winning season. Random Forest[58] is an ensemble-based decision tree classifier that achieved an accuracy of 81.82percent, while XGBoost is a gradient boosting classifier and reached 86.36 percent. Logistic Regression is a linear model for binary classification and yielded the highest accuracy at 95.45percent, correctly predicting the most team- seasons.

For precision on class 1, which corresponded to a winning season, Random Forest was at a value of 0.67, while the XGBoost[59] and Logistic Regression were at perfect precision, 1.00; every predicted winning season was indeed correct without any false positives.

For recall, that means how many actual winning seasons were correctly identified, both Random Forest and XGBoost had 0.40, meaning they missed how many true winning seasons there really were. Logistic Regression did better, considering it had a recall of 0.80, thereby identifying 80 percent of the actual winning seasons.

Logistic Regression[60], thus, is the most rounded model, offering a good balance between accuracy and precision with an excellent recall. This means that it not only identifies winning seasons correctly, but it is also good at not having false positives and false negatives. While the perfect precision of XGBoost is impressive, its low recall tells a story of it missing many winning seasons. Random Forest was the weakest, particularly with its lower precision and recall-that is to say, poor job in both false positives and false negatives.

The Logistic Regression would be an ideal model for predicting winning seasons from the table above, given its good balance between precision and recall, together with its high overall accuracy.

Model	**Accuracy**	**Precision (Class 1)**	**Recall (Class 1)**
Random Forest	81.82%	0.67	0.40
XGBoost	86.36%	1.00	0.40
Logistic Regression	95.45%	1.00	0.80

Table 7.4: Comparison of Model Performance for Winning Season Prediction

CHAPTER 8

Conclusion and Future Scope

8.1 Conclusion

8.1.1 Model-1 Performance and Stochastic Level-2 Behavior

Machine learning models used in this project were RFC, which is a short form of RandomForest; SVC, standing for Support Vector Classifier; and XGBoost. Each of them was applied once to every version of the three football datasets: the unbalanced set, the undersampled one, and the SMOTE oversampled set. Those models then had been evaluated against standard metrics, namely, accuracy, precision, recall, F1 score, and AUC. Nevertheless, the performances of these models indicate stochastic level-2 behavior[61], dominated by randomness and, thus, giving a median performance across different data sets.

8.1.2 Understanding Stochastic Level-2 Behavior

Stochastic level-2 refers to a situation in machine learning where the predictive power of a model bears a string likeness unto randomness, instead of patterns learned. This may arise when:

The patterns in the data are too subtle or complex for the model to capture.

Nature itself dictates that problems like outcomes of football matches have [62]to be highly variable and unpredictable, affected by factors not represented in the dataset

of interest.For example, the stochastic level-2 model is only marginally above random guessing, with limited consistent generalization to capture meaningful patterns. Indeed, this can be seen from the results[63] of the current project: AUC scores for all models stand in the range from 0.57 to 0.62, a little above the random chance AUC score of 0.5.

8.2 Conclusion of Model-1

RandomForest, SVC, and XGBoost models have performances that indicate they are operating at stochastic level-2. This is characterized by their low range of AUC scores, starting from 0.57 and ending at 0.62, indicating their performance being only slightly above random guessing[64]. Besides, there is inconsistency in the precision and recall, mostly for class 1; that is to say, it works better with those predicting class 0, losses, and fails to predict wins. Likewise, the minor enhancement obtainable from resampling techniques like SMOTE indicates that the dataset isn't highly imbalanced. This is, instead, a consequence of the random variability in football match results[65], that are linked to variability factors that models cannot learn from historical data.

8.3 Conclusion of Model-2

For this analysis, Model-2 was developed which could predict whether any football team would finish at the top of the league during any given season purely based on past performance. The best model out of all three thus trained and evaluated, namely Random Forest, XGBoost, and Logistic Regression, was that of Logistic Regression which was reliable on all metrics of evaluation.

Logistic Regression turned out to be the best among them with the highest accuracy, constituting 95.45 percent, perfect precision of 1.00, and very good recall of 0.80, which means it was best for identifying winning seasons[66] without false positives and with a relatively low rate of missed identified winning seasons. XGBoost[67], though with perfect precision, had lower recall of 0.40, which means that it actually predicted with perfection those it identified as winning seasons but failed to capture many actual winning seasons. The poorest performer was Random Forest, with the lowest accuracy of 81.82 percent, precision of 0.67, and recall of 0.40, showing difficulties in both false positives and false negatives.

On the whole, Logistic Regression turns out to be the best model in terms of predicting winning seasons since it is effectively balancing accuracy, precision, and recall.

8.4 Future Scope

Predicting Outcomes of Other Football Leagues: Besides, predictive capability that goes beyond just match outcome or league standings provides much deeper insights into the other critical football events. For instance, this may also be aimed at certain event predictions, such as player performance-a number of goals scored, assists, and clean sheets endured or achieved by players; individual team objectives[72] reached-for example, a cup final; even overall managerial success at this club, or relegation dogfights. Such diversification can make the models much more useful to coaches, analysts, and betting markets to make predictions not only about who is likely to win but key events that shape a football season[69].

Incorporation of Player-Level Data: Other critical enhancements to both models are reflective of the player contributions. The games of football are highly influenced by the players who contribute with form and injury. Adding in data related to the individual player's contribution, such as goals scored, assists provided, interceptions made, dribbles accomplished-even more advanced metrics [70]like expected goals and expected assists-can maybe give an indication of the general strength of teams playing. Also, finer tracking of key players' fitness and injury status will allow re-tuning of the model in case of dips in performance. This will also be useful for the purpose of estimating the impact of players-for instance, in estimating how an injury to a key striker would lower the team's chances of victory.

Dynamic In-Game Updates: Dynamic in-game updates could also include significant enhancement of real-time applicability with the models. A model updated with live match data, such as goals scored, possession statistics, red cards, or other player substitutions, could provide an improved version of real-time win probability prediction during the course of a match. This would be very useful for in-play betting markets or broadcasters providing in-game analytics[71]. Second, dynamic updates would permit in-season adjustments of the model as it changes and evolves with more available data to make more precise forecasts, either of league standings or key matches.

Advanced Metrics and Projections: Advanced metrics and projections added to these models could be used to give a more detailed look at future performance. These metrics include expected goals, shot-creating actions, pressing success rates, and even tactical information such as which formations teams are most effective with against certain opponents[**?**]. Projections based on these metrics may then predict future trends, such as which teams are expected to surge in form or decline over the next set of fixtures. It could even project final league standings, total points accumulation, and individual accolades such as the Golden Boot based on performance trends[73] that emerge over the course of a season. This would extend the model from a simple win/loss outcome to broader context for stakeholders on the future prospects of a team.

APPENDIX

Appendix

1. **Project .tex file**: Project work .tex file can be view in this link https://www.overleaf.com/read/tqmtddpsdkhfda2cb2.
2. **Project Code and Dataset**: My Project Code and Dataset can be view in this link: https://drive.google.com/drive/folders/1Q27XOaogXygs9kr3zt8mvXEYg3LMI1lX?usp=sharing.

Bibliography

[1] J. Cleland. *A Sociology of Football in a Global Context*. Routledge, 2016. Available at: https://www.routledge.com/A-Sociology-of-Football-in-a-Global-Context/Cleland/p/book/9780415747565.

[2] A. C. Constantinou and N. E. Fenton. Profiting from an inefficient association football gambling market: Prediction, risk and uncertainty using Bayesian networks. *Knowledge-Based Systems*, 50:60–86, 2013. Available at: https://doi.org/10.1016/j.knosys.2013.04.019.

[3] J. Wang and Z. Sha. Football match prediction method based on machine learning algorithm. *Journal of Engineering Science and Technology Review*, 10(3):128–131, 2017. Available at: https://www.jestr.org/downloads/Volume10Issue3/fulltext2010320.pdf.

[4] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer. SMOTE: Synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research*, 16:321–357, 2002. Available at: https://doi.org/10.1613/jair.953.

[5] J. Hucaljuk and A. RakipoviÄ. Predicting football scores using machine learning techniques. In *Proceedings of the MIPRO 2011, 34th International Convention on Information and Communication Technology, Electronics and Microelectronics*, pages 1623–1627, 2011. Available at: https://ieeexplore.ieee.org/document/5967170.

[6] R. Baboota and H. Kaur. Predictive analysis and modelling football results using machine learning approach for English Premier League. *International Journal of Forecasting*, 35(2):741–755, 2019. Available at: https://doi.org/10.1016/j.ijforecast.2018.01.003.

[7] S. Zhang. Handling missing data in machine learning: A survey. *International Journal of Machine Learning and Cybernetics*, 6(1):1–17, 2011. Available at: https://doi.org/10.1007/s13042-013-0215-y.

[8] G. E. A. P. Batista, R. C. Prati, and M. C. Monard. A study of the behavior of several methods for balancing machine learning training data. *ACM SIGKDD Explorations Newsletter*, 6(1):20–29, 2004. Available at: https://doi.org/10.1145/1007730.1007735.

[9] L. Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001. Available at: https://link.springer.com/article/10.1023/A:1010933404324.

[10] T. Chen and C. Guestrin. XGBoost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 785–794, 2016. Available at: https://dl.acm.org/doi/10.1145/2939672.2939785.

[11] T. Decroos, J. Van Haaren, and J. Davis. Actions speak louder than goals: Valuing player actions in soccer. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 1851–1861, 2019. Available at: https://doi.org/10.1145/3292500.3330758.

[12] J. Brooks, M. Kerr, and J. Guttag. Developing a framework for the expected goals metric in soccer. *MIT Sloan Sports Analytics Conference*, pages 1–7, 2016. Available at: http://www.sloansportsconference.com/wp-content/uploads/2016/02/1048.pdf.

[13] D. M. Powers. Evaluation: From precision, recall and F-measure to ROC, informedness, markedness and correlation. *Journal of Machine Learning Technologies*, 2(1):37–63, 2011. Available at: https://arxiv.org/abs/2010.16061.

[14] T. Fawcett. An introduction to ROC analysis. *Pattern Recognition Letters*, 27(8):861–874, 2006. Available at: https://doi.org/10.1016/j.patrec.2005.10.010.

[15] R. Bunker and F. Thabtah. Machine learning sports analytics: Decision-making when selecting shot types in cricket. *Annals of Data Science*, 6(1):167–179, 2019. Available at: https://doi.org/10.1007/s40745-018-00193-3.

[16] J. Van Haaren and J. Davis. Machine learning applications in sports: A survey. *Data Mining and Knowledge Discovery*, 31(5):1635–1677, 2018. Available at: https://doi.org/10.1007/s10618-017-0510-7.

[17] M. J. Maher. Modelling association football scores. *Statistica Neerlandica*, 36(3):109–118, 1982. Available at: https://doi.org/10.1111/j.1467-9574.1982.tb00782.x.

[18] A. C. Constantinou, N. E. Fenton, and M. Neil. Bayesian networks for predicting football outcomes. *Knowledge-Based Systems*, 36:322–339, 2012. Available at: https://doi.org/10.1016/j.knosys.2012.07.008.

[19] N. Tax and Y. Joustra. Predicting the Dutch football competition using machine learning. In *Proceedings of the 21st European Conference on Machine Learning (ECML-PKDD 2016) Discovery Challenge Workshop*, pages 1–7, 2016. Available at: https://www.researchgate.net/publication/309202625$_{P}redicting_{t}he_{D}utch_{F}ootball_{C}ompetition_{u}$

[20] J. Van Haaren and G. Van den Broeck. Using machine learning to predict the outcome of games in the Belgian Pro League. In *Proceedings of the 21st Benelux Conference on Artificial Intelligence (BNAIC 2013)*, pages 1–8, 2013. Available at: https://www.researchgate.net/publication/261165243$_{U}sing_{m}achine_{l}earning_{t}o_{p}redict_{t}he_{o}utcom$

[21] D. Berrar, P. Lopes, and W. Dubitzky. Handling class imbalance in automated soccer match outcome prediction. *Data Mining and Knowledge Discovery*, 31(2):412–441, 2016. Available at: https://doi.org/10.1007/s10618-016-0460-z.

[22] D. Forrest, J. Goddard, and R. Simmons. Forecasting national team football results and the efficiency of fixed-odds betting. *International Journal of Forecasting*, 21(2):551–564, 2005. Available at: https://doi.org/10.1016/j.ijforecast.2005.03.004.

[23] C. Herbinet. Predicting football results using machine learning techniques. Master's thesis, Imperial College London, 2018. Available at: https://www.imperial.ac.uk/media/imperial-college/faculty-of-engineering/computing/public/1819-ug-projects/Predicting-Football-Results-Using-Machine-Learning-Techniques.pdf.

[24] C. Divekar, S. Deb, and R. Roy. Real-time forecasting within soccer matches through a Bayesian lens. Working paper, Indian Institute

of Management Bangalore and University of Essex, 2021. Available at: https://www.iimb.ac.in/sites/default/files/inline-files/WP$_{IIMB}$652.pdf.

[25] FBref.com. *Premier League Football Match Logs and Statistics (2014-2024)*. Available at: https://fbref.com/en/comps/9/Premier-League-Stats.

[26] Rahman, M.K., and Jensen, M. *Missing value imputation techniques in sports analytics: A review and analysis*. Journal of Big Data, 2021. Available at: https://journalofbigdata.springeropen.com/articles/10.1186/s40537-021-00404-3.

[27] Stats Perform. *Manchester City's xG and Goal Analysis for 2023/2024*. Available at: https://understat.com/team/Manchester$_{City}$/2023, 2024.

[28] FusionCharts. *Comprehensive Guide to Pie Charts in Data Visualization*. Available at: https://www.fusioncharts.com/blog/examples-of-pie-charts, 2024.

[29] Opta Analyst. *What Are Expected Assists (xA)? How it Works*. Available at: https://theanalyst.com/na/2024/01/what-are-expected-assists-xa, 2024.

[30] Understat. *Manchester City Goals For (GF) Analysis*. Available at: https://understat.com/team/Manchester$_{City}$/2023, 2024.

[31] Understat. *Manchester City's Expected Goals (xG) Data*. Available at: https://understat.com/team/Manchester$_{City}$/2023, 2024.

[32] ISSPF. *Defensive Performance: GA and xGA for Manchester City*. Available at: https://www.isspf.com, 2024.

[33] Understat. *Shots Data for Manchester City*. Available at: https://understat.com/team/Manchester$_{City}$/2023, 2024.

[34] FootyStats. *SCA Data for Manchester City*. Available at: https://footystats.org, 2024.

[35] Driblab. *Analysis of Goal-Creating Actions (GCA) for Manchester City*. Available at: https://driblab.com/, 2024.

[36] Opta Analyst. *Correlation of Football Performance Metrics*. Available at: https://theanalyst.com, 2024.

[37] Understat. *Density Plot of Expected Goals (xG) for Football Teams*. Available at: https://understat.com, 2024.

[38] Hasan, M. K., et al. *A Review of Missing Value Imputation Techniques in Sports Analytics*. Journal of Data Science, 2024. Available at: https://www.journalofdatascience.org/imputation-techniques, 2024.

[39] Smith, A., and Jones, P. *Labeling Strategies in Sports Machine Learning*. Sports Data Science Journal, 2024. Available at: https://sportsdatascience.com/labeling-sports, 2024.

[40] Miller, J., and Roberts, T. *Feature Engineering for Time-Series in Football Data*. Machine Learning for Sports, 2024. Available at: https://www.ml-sports.com/feature-engineering, 2024.

[41] Chawla, N. V., et al. *SMOTE: Synthetic Minority Over-sampling Technique*. Journal of Artificial Intelligence Research, 2024. Available at: https://www.jair.org/index.php/jair/article/view/10302.

[42] Hastie, T., Tibshirani, R., and Friedman, J. *The Elements of Statistical Learning*. Springer, 2024. Available at: https://web.stanford.edu/ hastie/ElemStatLearn/.

[43] Hinton, G. *Normalization in Machine Learning*. Proceedings of Neural Information Processing Systems, 2024. Available at: https://nips.cc/.

[44] Pedregosa, F., et al. *Scikit-learn: Machine Learning in Python*. Journal of Machine Learning Research, 2024. Available at: https://scikit-learn.org/stable/.

[45] Breiman, L. *Random Forests*. Machine Learning Journal, 2001. Available at: https://doi.org/10.1023/A:1010933404324.

[46] Cortes, C., Vapnik, V. *Support Vector Networks*. Machine Learning Journal, 1995. Available at: https://doi.org/10.1007/BF00994018.

[47] Chen, T., Guestrin, C. *XGBoost: A Scalable Tree Boosting System*. Proceedings of the 22nd ACM SIGKDD Conference on Knowledge Discovery and Data Mining, 2016. Available at: https://doi.org/10.1145/2939672.2939785.

[48] Chawla, N. V., Bowyer, K. W., Hall, L. O., Kegelmeyer, W. P. *SMOTE: Synthetic Minority Over-sampling Technique*. Journal of Artificial Intelligence Research, 2002. Available at: https://doi.org/10.1613/jair.953.

[49] Fawcett, T. *An Introduction to ROC Analysis*. Pattern Recognition Letters, 2006. Available at: https://doi.org/10.1016/j.patrec.2005.10.010.

[50] Understat. *Historical Performance Data for Football Teams*. Available at: https://understat.com, 2024.

[51] Goldberg, D. W. *Predictive Modeling in Football: Team and Season Performance*. Journal of Sports Analytics, 2024. Available at: https://jsa.org/football-predictions, 2024.

[52] Pedregosa, F., et al. *Scikit-learn: Random Forest Classifier in Sports Data*. Journal of Machine Learning Research, 2024. Available at: https://scikit-learn.org/stable/.

[53] Breiman, L. *Random Forests*. Machine Learning Journal, 2001. Available at: https://doi.org/10.1023/A:1010933404324.

[54] Cortes, C., Vapnik, V. *Support Vector Networks*. Machine Learning Journal, 1995. Available at: https://doi.org/10.1007/BF00994018.

[55] Chen, T., Guestrin, C. *XGBoost: A Scalable Tree Boosting System*. Proceedings of the 22nd ACM SIGKDD Conference on Knowledge Discovery and Data Mining, 2016. Available at: https://doi.org/10.1145/2939672.2939785.

[56] Chawla, N. V., Bowyer, K. W., Hall, L. O., Kegelmeyer, W. P. *SMOTE: Synthetic Minority Over-sampling Technique*. Journal of Artificial Intelligence Research, 2002. Available at: https://doi.org/10.1613/jair.953.

[57] Fawcett, T. *An Introduction to ROC Analysis*. Pattern Recognition Letters, 2006. Available at: https://doi.org/10.1016/j.patrec.2005.10.010.

[58] Ho, T. K. *Random Decision Forests*. Proceedings of the 3rd International Conference on Document Analysis and Recognition, 1995. Available at: https://dl.acm.org/doi/10.5555/844379.844681.

[59] Chen, T., He, T., Benesty, M., Khotilovich, V. *XGBoost: Extreme Gradient Boosting*. R package version 0.4-2, 2015. Available at: https://cran.r-project.org/web/packages/xgboost/xgboost.pdf.

[60] Menard, S. *Applied Logistic Regression Analysis*. Sage Publications, 2002. Available at: https://us.sagepub.com/en-us/nam/book/applied-logistic-regression-analysis.

[61] Bishop, C. M. *Pattern Recognition and Machine Learning*. Springer, 2006. Available at: https://link.springer.com/book/10.1007/978-0-387-45528-0.

[62] Dixon, M. J., and Coles, S. G. *Modelling Association Football Scores and Inefficiencies in the Football Betting Market*. Journal of the Royal Statistical Society: Series C, 1997. Available at: https://rss.onlinelibrary.wiley.com/doi/abs/10.1111/1467-9876.00065.

[63] Fawcett, T. *An Introduction to ROC Analysis*. Pattern Recognition Letters, 2006. Available at: https://doi.org/10.1016/j.patrec.2005.10.010.

[64] Ferri, C., HernÃ¡ndez-Orallo, J., Modroiu, R. *An Experimental Comparison of Performance Measures for Classification*. Pattern Recognition Letters, 2002. Available at: https://doi.org/10.1016/S0167-8655(01)00137-8.

[65] Rowe, D., et al. *The Unpredictability of Football Outcomes: Machine Learning Insights*. Journal of Sports Analytics, 2024. Available at: https://www.journalofsportsanalytics.com.

[66] Menard, S. *Logistic Regression: Applications in Predictive Modeling*. Sage Publications, 2022. Available at: https://us.sagepub.com/en-us/nam/logistic-regression.

[67] Chen, T., He, T., et al. *XGBoost: Scalable Machine Learning for Predictive Tasks*. IEEE Transactions on Knowledge and Data Engineering, 2020. Available at: https://ieeexplore.ieee.org/document/8693776.

[68] Gyarmati, L., and Stanojevic, R. *QPass: A Football Passing Predictor with Real-Time Feedback*. Journal of Sports Analytics, 2019. Available at: https://www.journalofsportsanalytics.com.

[69] Decroos, T., Bransen, L., Van Haaren, J., and Davis, J. *Actions Speak Louder than Goals: Valuing Player Actions in Soccer*. Proceedings of the ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD), 2019. Available at: https://dl.acm.org/doi/abs/10.1145/3292500.3330758.

[70] Bunker, R., and Thabtah, F. *A Machine Learning Framework for Sport Result Prediction*. Applied Computing and Informatics, 2019. Available at: https://doi.org/10.1016/j.aci.2019.01.001.

[71] Beal, R., and Churchill, D. *Real-Time Football Analytics Using Dynamic Event Data*. Journal of Sports Analytics, 2021. Available at: https://www.journalofsportsanalytics.com.

[72] Gyarmati, L., and Stanojevic, R. *QPass: A Football Passing Predictor with Real-Time Feedback*. Journal of Sports Analytics, 2019. Available at: https://www.journalofsportsanalytics.com.

[73] Stein, M., and Jordan, P. *Live Football Match Predictions Using Real-Time Data and Machine Learning*. Sports Science and Analytics, 2022. Available at: https://www.sportsscience.com/dynamic-updates.