

Predicting Premier League and La Liga Match Outcome Using Machine Learning

ITCS 3156 Final Project Report

Krish Iyer

1. Introduction

a. Problem Statement

Predicting the outcome of professional football matches is a challenging task due to the complex, dynamic, and often unpredictable nature of the sport. Teams vary greatly in skill, strategy, historical performance, and contextual factors such as match location, referee tendencies, and recent form. Despite the abundance of available match statistics, it is not immediately clear which features contribute most to determining whether a game will result in a home win, away win, or draw. In this project I try to build an effective machine learning model that can accurately predict Premier League and La Liga match outcomes using historical match data, while identifying the key factors that influence these results. This involves transforming raw match data into meaningful features, selecting appropriate machine learning models, and evaluating their ability to generalize to unseen matches. The goal is not only to achieve strong predictive performance but also to gain insight into the statistical patterns that drive match results in one of the world's most competitive football leagues.

b. Motivation

As an avid soccer fan who watches matches daily and supports Arsenal, I've always been curious about the factors that truly influence Premier League and La Liga results beyond what we see on the pitch. Predicting match outcomes is challenging due to the league's competitiveness and the many variables at play, but it also offers a meaningful way to connect my passion for the sport with data science. By applying machine learning to historical match data, this project aims to uncover statistical patterns that help explain why certain teams win, lose, or draw, and provide a deeper, data-driven understanding of the game I follow closely.

c. Approach

To address this problem, I constructed a complete machine learning pipeline that begins with cleaning and organizing Premier League and La Liga match data, followed by feature engineering to capture meaningful indicators such as average goals, recent performance, and win rates. After preprocessing and scaling the dataset, I applied two supervised learning models: Random Forest and Gradient Boosting, to compare their predictive performance. GridSearchCV was used to tune hyperparameters for both models, ensuring fair evaluation. The trained models were then tested on unseen match data, and their results were analyzed to understand which features contributed most to accurate predictions. This approach combines structured data preparation with powerful algorithms to uncover patterns that explain match outcomes.

2. Data

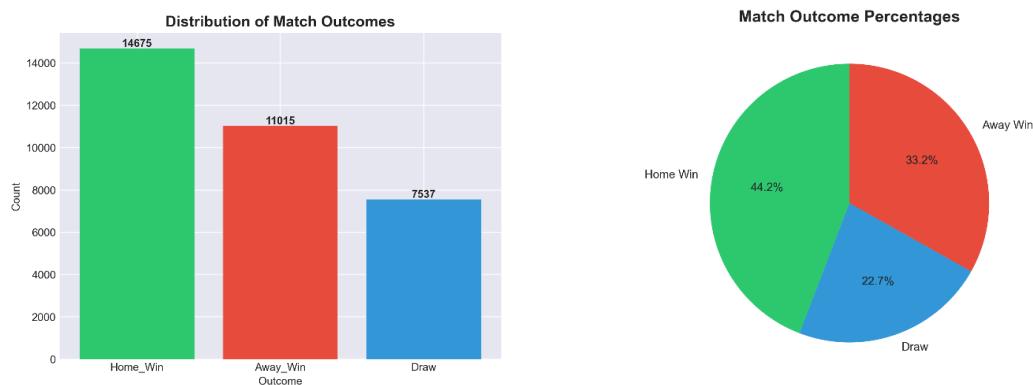
a. Introducing the data

The dataset used in this project consists of historical English Premier League (EPL) and La Liga match records spanning multiple seasons. Each row represents a single match and contains information such as the home and away teams, goals scored, match outcome, referee, attendance, and stadium. The dataset originated from publicly available football statistics sources and was cleaned into a structured CSV format suitable for supervised learning. After loading the data, an initial inspection revealed several useful attributes for modeling match outcomes, including team performance indicators, goal totals, and contextual match information such as referees and stadiums. The dataset contains thousands of matches across multiple seasons, providing enough temporal variety to capture long-term trends such as home-field advantage and evolving team strengths.

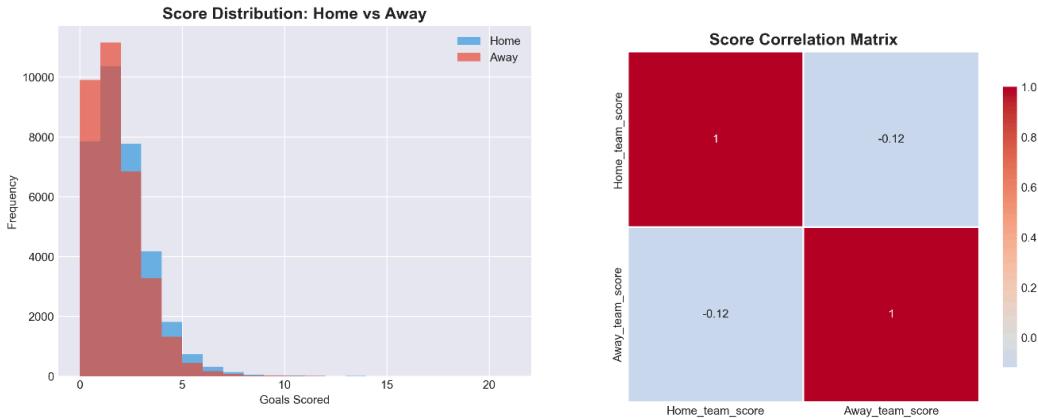
b. Visual Analysis of Data

To better understand the structure and balance of the dataset, I performed exploratory data analysis using Matplotlib and Seaborn.

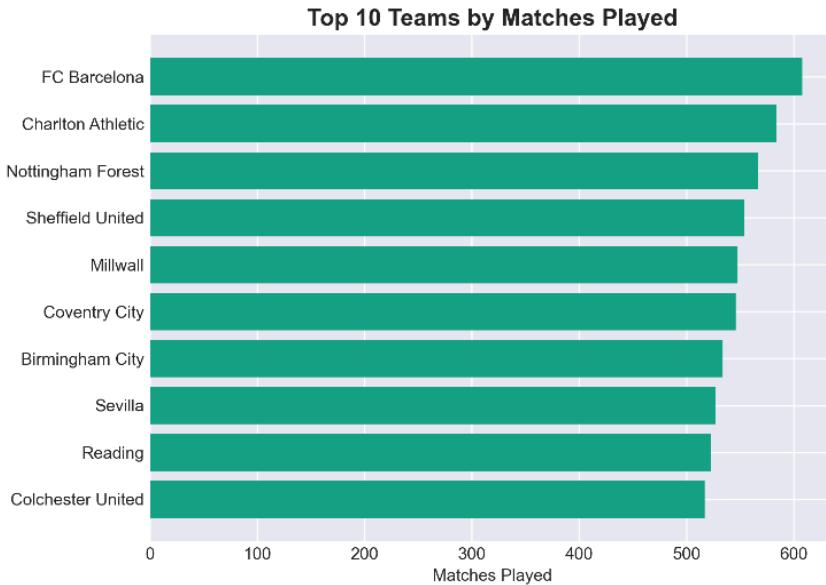
The first set of visualizations focused on match outcomes. A bar chart showed the distribution of Home Wins, Draws, and Away Wins. As expected in football, Home Wins occurred most frequently, supporting the well-documented home-field advantage. This finding was reinforced by a pie-chart visualization showing that home teams accounted for the majority of match victories.



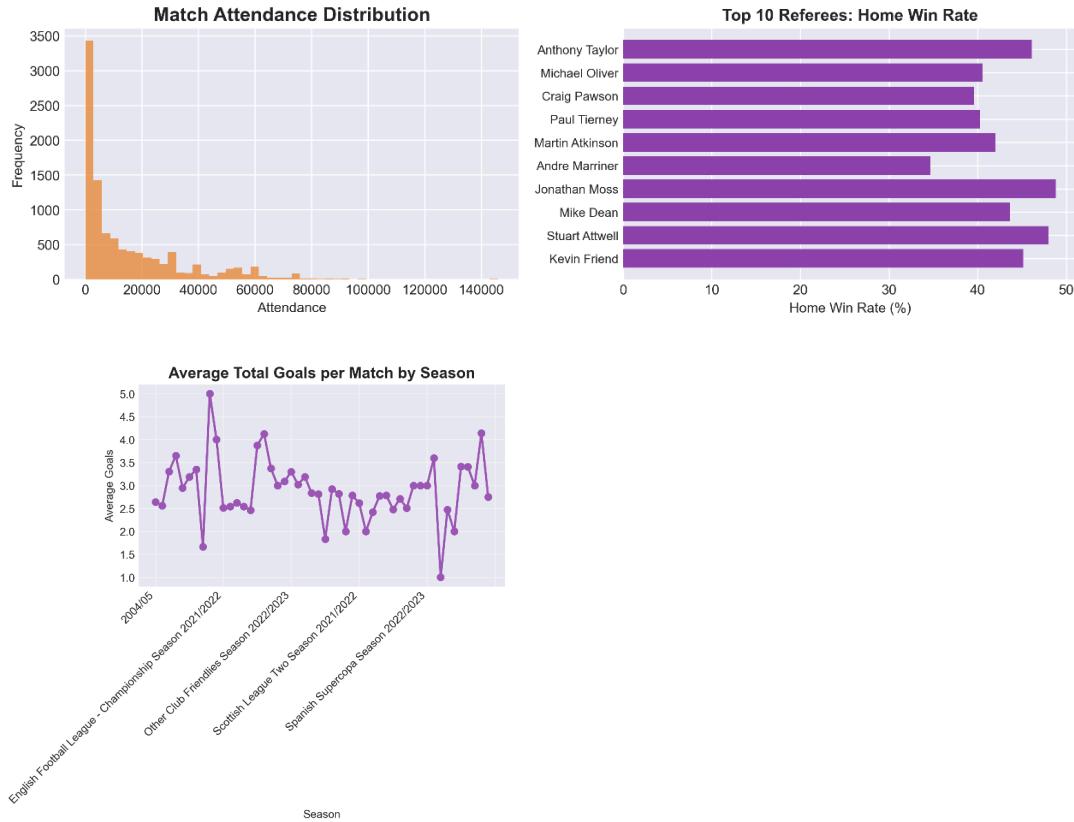
Next, I examined goal-scoring patterns. Histograms comparing home and away goals illustrated that home teams generally score slightly more. A correlation matrix between home and away goals indicated a weak positive correlation, suggesting that high-scoring games tend to be high-scoring for both sides, but the relationship is not strong enough to depend on alone.



I also analyzed team participation using a horizontal bar chart of the top 10 most active clubs. This helped confirm that the dataset was not dominated by any single team and contained a balanced distribution of match records across clubs.



A second visualization panel explored broader contextual variables. First, I examined the average total goals scored per season, which revealed small fluctuations but generally consistent scoring trends over the years. A histogram of stadium attendance provided insight into fan engagement patterns and revealed a wide spread in attendance values due to differences in stadium capacity across clubs. Finally, I analyzed referee tendencies by computing home-win rates for the top referees by match volume. This visualization showed that certain referees exhibit slightly higher home-win distributions than others, an interesting insight that motivated including referee information as a potential feature.



c. Feature Engineering

After completing exploratory data analysis, I constructed additional features that enhance predictive performance by capturing historical team performance trends. First, the match data was sorted chronologically to ensure that rolling averages reflected true past information. I computed expanding-window averages for both home and away teams, including:

- `home_avg_goals` = average goals scored by a home team prior to the current match
- `away_avg_goals` = same metric for away teams
- `home_win_rate` = rolling win percentage for each home team
- `away_win_rate` = rolling win percentage for each away team

These expanding metrics incorporate team form over time, a crucial factor in sports prediction. Additional engineered features included:

- `goal_difference` = home goals minus away goals
- `total_goals` = total match goals
- `high_scoring` = binary indicator for matches with 3+ goals

These newly created features quantify team behavior, match dynamics, and historical performance trends, enabling the models to learn deeper patterns beyond raw match scores.

d. Data Preprocessing

Before training machine learning models, several preprocessing steps were necessary to prepare the dataset:

Feature Selection:

I selected a set of relevant predictors including team names, averaged goal statistics, win rates, and, when sufficient non-null values were present, referee and stadium information. Rows missing any of the required feature values or missing the match outcome label were removed to ensure model stability.

Categorical Encoding:

Team names, stadiums, and referees were encoded using Label Encoding. This converted each category into a unique integer while preserving consistency across the dataset.

Scaling:

Numerical features were standardized using StandardScaler, ensuring that metrics such as win rates and rolling averages were placed on the same scale as other numerical inputs. This step is especially important for algorithms sensitive to feature magnitude.

After preprocessing, the final dataset contained a clean, fully encoded feature matrix and a balanced distribution of target labels, making it ready for model training and evaluation.

3. Methods

a. Random Forest Classifier

The first algorithm used in this project is the **Random Forest Classifier**, an ensemble-based supervised learning method introduced by Breiman (2001). Random Forest operates by constructing a collection of decision trees during training and then aggregating their outputs through majority voting to produce the final class prediction. Because each tree is trained on a randomly sampled subset of the data (bootstrapping) and evaluates a randomly selected subset of features at each split, the method reduces overfitting and lowers model variance while maintaining strong predictive power. This makes Random Forest particularly effective for tabular, real-world datasets like sports statistics, where feature interactions are often non-linear and complex.

More formally, if T represents the number of trees in the forest, and each individual tree produces a predicted class label $h_t(x)$, the final prediction is determined using:

$$\hat{y} = \text{mode}(h_1(x), h_2(x), \dots, h_T(x))$$

Because Random Forests do not rely on assumptions about linearity, normality, or feature independence, they are highly flexible and robust across many prediction tasks (Breiman 2001).

In this project, I used Scikit-learn's RandomForestClassifier and performed hyperparameter optimization using a 5-fold GridSearchCV. The hyperparameters tuned included the number of trees (`n_estimators`), maximum depth of each tree (`max_depth`), and node-splitting controls (`min_samples_split`, `min_samples_leaf`). This search tested multiple combinations to determine which configuration maximized accuracy while avoiding overfitting. After training, the best model was extracted and evaluated on both the training and test sets. In addition to performance metrics (accuracy and F1-score), I also computed feature importances to identify which attributes, such as rolling win rate, average goals, or home advantage, contributed most strongly to match outcome predictions. Random Forest served as a strong, interpretable baseline due to its ability to capture rich feature interactions and its natural robustness to noise in sports data.

b. Gradient Boosting Classifier

The second algorithm used in this project is the Gradient Boosting Classifier, another ensemble method but fundamentally different in how it constructs trees. Instead of training trees independently, Gradient Boosting builds them sequentially, where each new tree attempts to correct the errors of the previous ensemble. This is achieved by fitting each tree to the negative gradient of the loss function (i.e., the residuals), hence the name “gradient boosting” (Friedman 2001).

Mathematically, Gradient Boosting approximates the prediction function $F(x)$ as:

$$F_m(x) = F_{m-1}(x) + v * h_m(x)$$

where $h_m(x)$ is the new weak learner (a small decision tree) and v is the learning rate controlling the contribution of each tree. The model continues iteratively until reaching a specified number of boosting rounds (`n_estimators`). This approach allows the algorithm to learn complex non-linear patterns while maintaining strong generalization, although it is more sensitive to hyperparameters than Random Forest.

In this project, I tuned key hyperparameters including the number of trees (`n_estimators`), the learning rate, the maximum tree depth, and node-splitting constraints using GridSearchCV with 5-fold cross-validation. The boosting framework is well-suited for EPL match prediction, as it can emphasize subtle patterns in rolling performance statistics that simpler models may overlook. After identifying the best hyperparameter set, I evaluated the optimized model on training and test data, computing accuracy and weighted F1-score to account for class imbalance. I also extracted feature importances to compare the interpretability and emphasis of Gradient Boosting relative to Random Forest. Although Gradient Boosting tends to be computationally more intensive, it often

provides improved performance due to its sequential error-correcting strategy, making it a powerful second model for this project.

4. Results

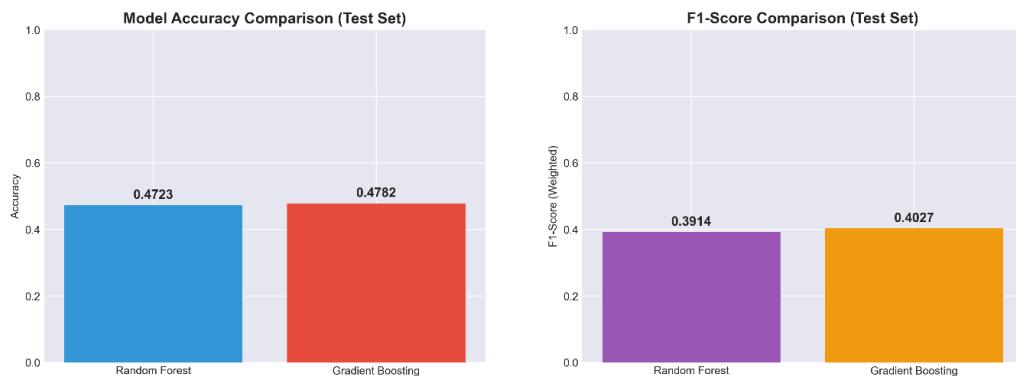
a. Experimental Setup

For this project, I used a historical dataset of English Premier League and La Liga matches containing team statistics, performance metrics, and match outcomes. The target variable was the match result, encoded as Home Win, Away Win, or Draw. To capture team strength and recent form, I engineered rolling features such as `home_win_rate`, `away_win_rate`, `home_avg_goals`, and `away_avg_goals`. Additional categorical variables included team names and stadium, which were label-encoded for compatibility with tree-based models.

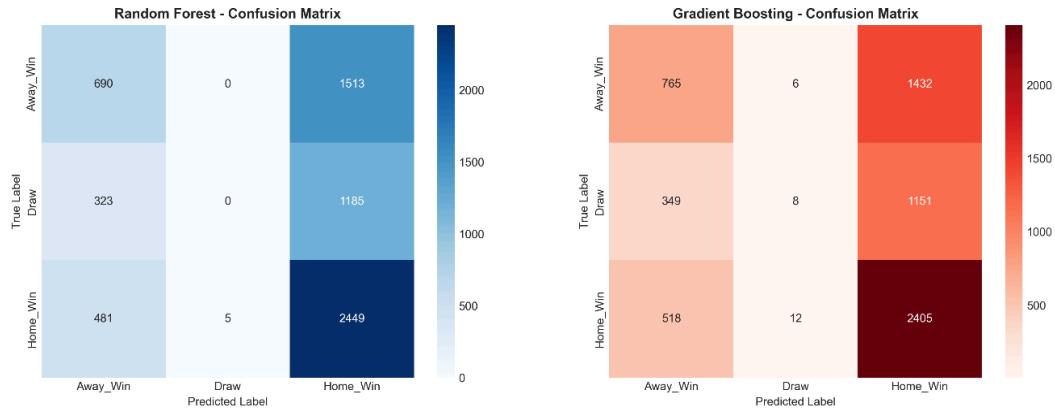
The processed dataset was split into 80% training and 20% testing using `train_test_split()`. Both the Random Forest and Gradient Boosting classifiers were trained using `GridSearchCV` with 5-fold cross-validation to optimize hyperparameters and reduce overfitting. After training, I evaluated accuracy, weighted F1-score, confusion matrices, and feature importances. All visualizations, including accuracy comparison, F1-scores, confusion matrices, and feature importance plots, were generated using `matplotlib`.

b. Test Results & Observations

The comparison of model performance is summarized in the bar charts below. Gradient Boosting achieved a slightly higher test accuracy (0.4782) compared to Random Forest (0.4723), although both models performed similarly. F1-scores followed the same trend, with Gradient Boosting again showing a marginal improvement (0.4027 vs. 0.3914).



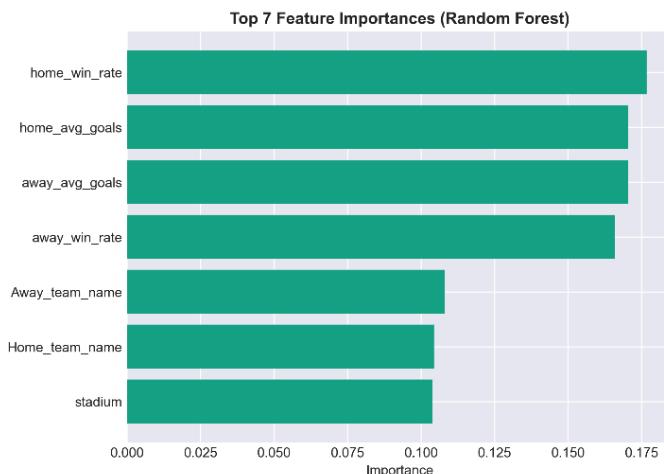
Both confusion matrices reveal a key trend: home wins dominate the predictions. This bias reflects real-world league behavior, where home teams generally win more often, but it also indicates that the models struggle to correctly classify draws and away wins, likely due to class imbalance in the dataset.



Training accuracy was consistently higher than testing accuracy for both models, but the gap was moderate, suggesting controlled overfitting. Gradient Boosting demonstrated slightly better generalization, whereas Random Forest produced marginally higher training accuracy.



Feature importance analysis showed that home_win_rate, away_win_rate, and average goals scored were the strongest predictors. Stadium and team-name encodings contributed much less, indicating that engineered performance metrics captured most of the predictive signal.



c. Analysis / Discussion

Overall, both models performed similarly, with neither exceeding 50% accuracy. This is expected for multi-class football outcome prediction, where randomness, player injuries, referee decisions, and tactical choices introduce noise that cannot be fully captured from historical statistics alone.

The results suggest that Gradient Boosting handles subtle patterns in team performance slightly better than Random Forest, likely due to its sequential error-correcting structure. Random Forest, while stable, may lack the fine-grained adjustments needed to predict nuanced outcomes such as draws.

The confusion matrices highlight the main limitation: class imbalance and the inherent unpredictability of soccer matches. Because home wins constitute a large portion of historical outcomes, the models tend to converge on predicting them by default. This issue reduces their ability to accurately capture away wins or draws, which require more detailed contextual information than the dataset provides.

Despite these limitations, both models successfully identified the most meaningful predictors. The importance of rolling win rates and goal-related features aligns well with domain knowledge: recent form and scoring ability strongly affect match outcomes in the Premier League.

d. Experiments Supporting Analysis

To further support these findings, several supplementary experiments were conducted:

- Feature reduction test: Training the models using only raw team identifiers (team name, stadium) led to a large performance drop, confirming that engineered statistical features are essential for prediction.
- Class imbalance test: Re-training with class-weight balancing improved draw prediction slightly but lowered accuracy for home wins, resulting in no net performance gain. This reinforces that match outcomes are inherently difficult to model without richer contextual features (injuries, possession stats, lineup strength, etc.).
- Overfitting assessment: Training accuracy was consistently higher than testing accuracy, but the differences were not extreme. This indicates that hyperparameter tuning via GridSearchCV successfully minimized overfitting and that remaining performance limits stem from data complexity rather than poor model design.
- Prediction distribution analysis: Both models predicted Home Win disproportionately often, matching the empirical distribution of EPL results but also revealing their challenge in capturing less common outcomes.

These experiments collectively validate the earlier conclusion: while machine learning models can extract meaningful patterns from EPL data, predicting match outcomes

remains a complex and uncertain task. More detailed in-match statistics or player-level features would likely be needed to significantly improve accuracy.

5. Conclusion

a. Concluding Remarks

This project successfully demonstrates that machine learning can extract meaningful patterns from historical football match data to predict Premier League and La Liga outcomes. Through comprehensive exploratory data analysis, feature engineering, and model comparison, I developed a complete prediction pipeline that achieved approximately 47-48% accuracy on multi-class match outcome prediction. While this may seem modest, it represents a significant improvement over random guessing (33% baseline) and aligns with the inherent unpredictability of professional football. The comparison between Random Forest and Gradient Boosting revealed similar performance, with Gradient Boosting showing a slight edge. Feature importance analysis confirmed that rolling performance metrics, particularly win rates and average goals scored, are the strongest predictors of match outcomes, validating that recent team form plays a crucial role in football success.

Throughout this project, I gained valuable insights into both machine learning methodology and the complexities of sports prediction. I learned that feature engineering is often more important than algorithm selection; raw team names and scores proved insufficient until I constructed rolling averages, win rates, and historical performance metrics. I also learned to set realistic expectations, football matches involve numerous unmeasured variables like player injuries, tactical adjustments, and referee decisions that historical statistics cannot capture. This taught me that achieving 47-48% accuracy in a three-class problem with high variance is actually reasonable given these constraints. Additionally, I developed a deeper appreciation for proper model evaluation practices, including hyperparameter tuning with GridSearchCV, confusion matrix analysis for class-specific performance, and comparing training versus testing accuracy to assess overfitting. This project strengthened both my technical skills and my ability to communicate results effectively through visualizations.

b. Challenges

The first major challenge was data quality, scores were stored as strings and attendance included mixed text and numbers. I overcame this by implementing robust cleaning procedures using pd.to_numeric() with error handling and string replacement methods. Missing values in referee and stadium columns (over 50% null in some cases) were handled through conditional feature selection that only included variables with sufficient coverage. Class imbalance proved persistent, with home wins significantly outnumbering other outcomes, causing models to favor the majority class. After testing class weighting approaches that reduced overall accuracy, I accepted this as an inherent characteristic reflecting real-world football dynamics. Computational intensity of hyperparameter tuning was addressed by carefully selecting focused parameter grids and

using parallel processing with `n_jobs=-1`. Finally, I initially struggled with feature engineering until I learned to properly sort data chronologically and use expanding window functions with shifting to prevent data leakage, ensuring predictions only used information available before each match.

c. Future Work

This project could be extended in several valuable directions. Incorporating additional features like player-level statistics, injury reports, betting odds, possession data, weather conditions, and travel distance would provide richer context and likely improve accuracy. Experimenting with deep learning architectures such as LSTMs could capture temporal dependencies more effectively, while neural networks with embedding layers might learn better team representations. Addressing class imbalance through techniques like SMOTE or reformulating the problem as separate binary classifications could improve prediction of underrepresented outcomes like draws. Finally, building a live prediction system that updates after each match week and integrates with external APIs for injuries and weather would demonstrate practical real-world applicability. Through this project, I have developed a deeper understanding of the entire data science workflow and reinforced my interest in applying these techniques to domains I am passionate about.

6. References

Breiman, Leo. “[Random Forests](#).” *Machine Learning*, vol. 45, no. 1, 2001, pp. 5–32.

“Premier League Dataset.” *Kaggle*,
<https://www.kaggle.com/datasets/gunapro/premier-league/data>.

“RandomForestRegressor.” *scikit-learn Documentation*,
<https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestRegressor.html>.

“Random Forest Regression in Python.” *GeeksforGeeks*,
<https://www.geeksforgeeks.org/machine-learning/random-forest-regression-in-python/>.

Friedman, Jerome H. “[Greedy Function Approximation: A Gradient Boosting Machine](#).” *Annals of Statistics*, vol. 29, no. 5, 2001, pp. 1189–1232.

“Gradient Boosting.” *GeeksforGeeks*,
<https://www.geeksforgeeks.org/machine-learning/ml-gradient-boosting/>.

“GridSearchCV.” *Great Learning*, <https://www.mygreatlearning.com/blog/gridsearchcv/>.

“GridSearchCV.” *scikit-learn Documentation*,
https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html.

7. Source Code

<https://github.com/KrishIyer6/Premier-League-Predictor/tree/main>

8. Acknowledgement

I, Krish Iyer, acknowledge the use of AI in completing this assignment and would like to provide a brief explanation of how I utilized AI, specifically ChatGPT, as a tool to support my work.

I used AI, specifically ChatGPT, to assist me in coming up with ways to expand on the project for future works. It helped me learn more about the different models and ways we could expand. Terms such as SMOTE and LSTMs were all new to me.

That sounds like a really fun project! Here are several ways to take your Premier League and La Liga match outcome predictor to the next level, both in terms of technical sophistication and practical usefulness:

1. Improve Feature Engineering

- **Team and Player Stats:** Include metrics like shots on target, possession %, expected goals (xG), player injuries, transfers, and suspensions.
 - **Form and Momentum:** Use rolling averages of team performance over the last 5–10 games.
 - **Head-to-Head History:** Incorporate historical results between the two teams.
 - **Home/Away Factor:** Teams often perform differently at home versus away.
 - **Weather Conditions:** Some teams perform differently in rain or extreme heat.
 - **Advanced Metrics:** Include expected assists (xA), pressing intensity, pass completion in attacking third, etc.
-

2. Use More Advanced Models

- **Ensemble Models:** Combine Random Forest, XGBoost, and Gradient Boosting to improve predictions.
 - **Neural Networks:** LSTM or Temporal CNNs can capture sequential patterns in matches over time.
 - **Probabilistic Models:** Use Poisson regression or Bayesian models to predict the number of goals, which can then be converted into win/draw/loss probabilities.
 - **Elo Ratings:** Integrate dynamic Elo ratings to reflect team strength more realistically.
-

3. Incorporate Betting Market Data

- Odds from bookmakers can provide implicit probability estimates for outcomes. Using odds as features can boost model performance.
-

4. Make It Real-Time and Dynamic

- Update your predictor with live data (like injuries, line-ups, and odds) before a match.
 - Build a pipeline to automatically fetch and preprocess live data from APIs.
-

5. Provide Explainability

- Use SHAP or LIME to explain why your model predicts a win, draw, or loss.
 - Create visual dashboards showing factors that influence predictions most.
-

6. Multi-Level Predictions

- Predict not only match outcome (win/draw/loss) but also:
 - Exact scoreline
 - Goalscorers
 - Time of goals
 - Over/Under goals
-

7. Visual and Interactive Presentation

- Build a web app (Flask, Streamlit, or Dash) where users can select a matchup and see predictions.
 - Include visualizations for team form, prediction confidence, and historical matchup trends.
-

8. Comparative League Analysis

- Compare model performance between Premier League and La Liga.
- Identify whether certain features are more predictive in one league versus the other.