# Group 24 Final Report: Premier League Predictor

**Omar Abdelhamid, Khalid Farag, Omar El-Aref**
{abdelo8,faragk1,elarefo}@mcmaster.ca

## 1 Introduction

Predicting the outcomes of football competitions and generating full-season league standings is a challenging problem that lies at the intersection of statistics, machine learning, and sports analytics. The English Premier League (EPL), with its twenty teams, dense schedule, and high competitive parity, provides an especially difficult testbed: small changes in form, injuries, or tactics can produce large shifts in results over a season. In this project, we focus on building a data-driven model that uses historical EPL match data and pre-match betting odds to predict both individual match outcomes and the resulting final league table for the 2024–25 season.

Machine learning has been widely applied to sports prediction problems, where structured historical data and clearly defined outcomes make the domain suitable for classification and forecasting tasks. Prior work has proposed general frameworks for sport result prediction that evaluate different families of models and feature sets across multiple sports (Bunker and Thabtah, 2019), and deep learning has been successfully used to model complex patterns in game results, such as predicting Major League Baseball outcomes from rich statistical inputs (Huang and Li, 2021). These efforts build on broader advances in representation learning, regularized optimization, and large-scale model training in machine learning (Ando and Zhang, 2005; Andrew and Gao, 2007; Rasooli and Tetreault, 2015), as well as foundational algorithmic and formal-methods ideas that underlie modern data processing pipelines (Gusfield, 1997; Chandra et al., 1981).

To address this prediction task, we design a model that integrates several forms of pre-match information like team metadata, betting odds, head-to-head history, and each team's recent form; through a set of specialized encoders whose outputs are combined to forecast the goals scored by each team. This architecture allows the model to learn meaningful patterns in how past performance, momentum, and historical interactions influence future outcomes. Our primary objective is to build a system that can accurately predict match results and produce reliable season-long forecasts, with full implementation details presented later in the report.

## 2 Dataset

Our dataset consists of historical English Premier League match data covering seasons 2010–11 through 2024–25, obtained from the public repository (Football-Data.co.uk, 2025). The English Premier League is the top tier of English football, featuring 20 teams that compete in a double round-robin format: each team plays every other team twice (once at home, once away), resulting in 380 matches per season, which is 38 matches per team per season. The dataset spans 15 complete seasons, providing 5,700 total matches for this project.

### 2.1 Dataset Properties

The dataset contains comprehensive match-level information organized into several categories. Each match record includes the following types of data:

| Category | Example Columns |
|---|---|
| Match Info | Date, HomeTeam, AwayTeam, Season, Div |
| Full-time Outcome | FTHG, FTAG, FTR |
| Half-time Stats | HTHG, HTAG, HTR |
| Offensive Metrics | HS, AS, HST, AST, HC, AC |
| Discipline | HF, AF, HY, AY, HR, AR |
| Betting Odds | B365H, B365D, B365A |

Table 1: Example columns included in the Premier League dataset.

The dataset contains many important features for predictive modeling. First, it contains the temporal

continuity over the 15 seasons, allowing our model to learn the long-term patterns and adapt to team performance and form during each season. Second, it contains the pre-match betting odds, which provide a strong feature expectation for the match outcome. Third, the chronological ordering enables proper temporal modeling where predictions for future matches can only use information from past matches. Finally, the dataset is complete, with no or minimal missing data, with most matches having complete records for goals, results, and betting odds.

### 2.2 Preprocessing Operations

The preprocessing pipeline, implemented in `build_dataset.py` and `importing_files.py`, transforms raw match data into a format suitable for sequential modeling. The pipeline consists of several stages: data cleaning, feature extraction, history construction, and data splitting.

#### 2.2.1 Data Cleaning

The first stage involves cleaning and standardizing the raw match data:

- **Removal of incomplete records**: Matches with missing full-time goals (FTHG, FTAG) or results (FTR) are excluded from the dataset using `dropna` to ensure that the dataset is complete and can be used for training and evaluation.
- **Betting odds processing**: Betting odds columns (B365H, B365D, B365A) are converted to numeric format and filled with 0.0 to avoid any invalid or NaN values.
- **Date sorting**: Match dates are parsed to datetime format and the entire dataset is sorted chronologically by season and date, which is essential for temporal modeling.
- **Team identification**: Create mapping of the team names to unique IDs to be used for the model.

#### 2.2.2 Feature Extraction

For each match, we extract features from the perspective of each team using the `_game_features_from_perspective` function. This creates a 7-dimensional feature vector per match that captures the essential match outcomes. The features are:

- **Goals for (gf)**: Number of goals scored by the team in the match
- **Goals against (ga)**: Number of goals conceded by the team
- **Goal difference (gd)**: Computed as gf - ga, providing a single metric for match performance
- **Home indicator (is_home)**: Binary value (1.0 if playing at home, 0.0 if away), capturing stadium effects
- **Win indicator**: Binary value (1.0 if the team won, 0.0 otherwise)
- **Draw indicator**: Binary value (1.0 if the match was drawn, 0.0 otherwise)
- **Loss indicator**: Binary value (1.0 if the team lost, 0.0 otherwise)

#### 2.2.3 History Construction

The preprocessing builds two types of match history sequences that capture different aspects of team performance:

**Team Form History:** For each team, we maintain a window of the last $k_{form} = 5$ matches, capturing recent form of the team. This form history is updated incrementally as matches are processed chronologically, ensuring that only information available before each match is used for prediction. The choice of $k_{form} = 5$ balances the need for sufficient context to capture momentum and trends while avoiding excessive noise from older matches that may be less relevant to current form.

**Head-to-Head History:** For each team pair, we maintain the last $k_{h2h} = 4$ encounters between the two teams, using the same 7-dimensional feature representation. Head-to-head history captures matchup specific information, such as tactical advantages, psychological factors, and historical dominance patterns. This is particularly valuable for teams with strong historical records against specific opponents, as it allows the model to learn these matchup-specific patterns.

Both history types are constructed incrementally as matches are processed in chronological order. When a team has fewer than $k$ previous matches (e.g., at the start of a season or for newly promoted teams), the history is padded with zero vectors, ensuring consistent input dimensions for the model.

#### 2.2.4 Data Split

We employ a chronological train-validation split to simulate realistic prediction scenarios:

- **Training set**: All matches from seasons 2010–11 through 2023–24, comprising approximately 5,320 matches (14 seasons $\times$ 380 matches per season)
- **Validation set**: All matches from the 2024–25 season, comprising 380 matches

This split strategy ensures that the model is evaluated on future matches relative to its training data, reflecting how the model would be used in practice to predict upcoming matches. The validation set represents a complete, unseen season, providing a robust evaluation that tests the model's ability to generalize to new seasons with potentially different team compositions, managerial changes, and league dynamics.

The chronological split is critical because football data exhibits strong temporal dependencies: team performance evolves over time, and using future information to predict past matches would create unrealistic performance estimates. By strictly maintaining temporal order, we ensure that our evaluation metrics reflect the model's true predictive capability.

### 2.3 Dataset Changes Since Progress Report

Since the progress report, we made significant changes to address some of the concerns raised the group had in regards of the prediction process. The primary modification was the removal of post-match statistics (such as shots, corners, fouls, and cards) from the feature set, as these are only known after a match concludes.

**Initial Approach:** Our first implementation used season-level aggregated statistics (average goals, shots, cards, etc.) combined with post-match statistics as features in a feedforward network. While this approach showed promising results, it suffered from data leakage, as many statistics are only available after matches are played.

**Current Approach:** The final implementation uses only pre-match betting odds as explicit input features, with match history sequences constructed from past match outcomes. This ensures that all features used for prediction are available before kickoff, making the model suitable for real-world forecasting applications. The betting odds serve as a strong pre-match signal, while the match history sequences capture temporal patterns in team performance.

This change required restructuring the preprocessing pipeline to focus on outcome-based features rather than in-game statistics. The resulting model is more realistic, as it can make predictions using only information that would be available to a someone before a match begins.

## 3 Features and Inputs

Our model uses a carefully designed set of features that capture both pre-match expectations and historical team performance patterns. All inputs are available before a match begins, ensuring realistic predictions.

### 3.1 Feature Components

**Pre-Match Features:** The primary pre-match features are Bet365 betting odds (`B365H`, `B365D`, `B365A`), which represent betting odds of match outcome probabilities. These odds aggregate expert knowledge and statistical analysis. We also use learned embeddings for team identities (16-dimensional) and stadium effects (4-dimensional), capturing team characteristics and home advantage.

**Historical Sequence Features:** We construct two types of match history sequences:

- **Team form history**: Last $k_{form} = 5$ matches per team, each represented by a 7-dimensional vector (goals for/against/difference, home/away indicator, win/draw/loss indicators). Processed by a GRU encoder to produce a 32-dimensional representation of recent performance trends.
- **Head-to-head history**: Last $k_{h2h} = 4$ encounters between each team pair, using the same 7-dimensional representation. Also encoded by a GRU to capture matchup-specific dynamics and historical dominance patterns.

The choice of $k = 5$ balances sufficient context with computational efficiency, avoiding noise from older matches while capturing recent momentum.

### 3.2 Feature Engineering Rationale

Following feedback from the progress review and advice from the professor, we restricted features to those available before match time. Betting odds are the only explicit pre-match statistics, while all other features are derived from historical match

outcomes. The sequential nature of team form and head-to-head histories allows the model to capture temporal dependencies, preserving match order rather than using aggregated statistics. The model employs learned embeddings and GRU encoders to discover latent team characteristics and extract relevant patterns from match sequences.

### 3.3 Input Representation

The final input concatenates: team ID embeddings for both teams (16 dimensions each), stadium embedding (4 dimensions), betting odds vector (3 dimensions), and encoded histories for team 1 form, team 2 form, and head-to-head (32 dimensions each from GRU encoders).

## 4 Implementation

Our final model is a multi-encoder neural architecture designed to predict the number of goals scored by each team in a match. Unlike our earlier progress-report model, which relied solely on pre-match betting odds and a single RNN over match sequences, our updated system incorporates multiple sources of football-specific information, enabling more expressive representations and stronger predictive performance.

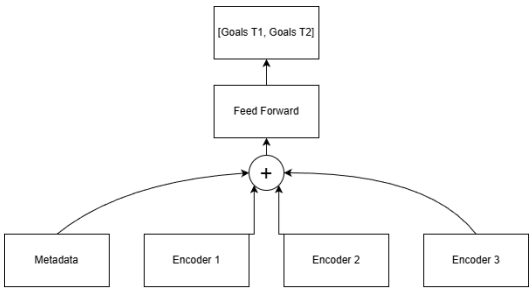


Figure 1: High-level architecture of the final multi-encoder goal prediction model.

### 4.1 Model Architecture

The model (Figure 1) consists of four separate encoders whose outputs are combined and passed into a feed-forward prediction module:

**Metadata Encoder:** This component embeds team identities using learned team-ID embeddings and encodes contextual numeric inputs such as Bet365 odds. It also incorporates information about which team is home and which is away. These representations capture all information known before entering a match.

**Head-to-Head (H2H) Encoder (Encoder 1):** A GRU-based sequence encoder processes the last 5 matches played between the two teams, capturing rivalry patterns, historical dominance, and recurring matchup behaviors. The number of past matches used is a tunable hyperparameter.

**Team-Form Encoders for Team 1 and Team 2 (Encoder 2 and Encoder 3):** Two GRU-based encoders summarize each team's recent form using their past 5 matches, including goals scored, goals conceded, match results, goal difference, and home/away indicators. When `share_team_encoders=True`, both teams use the same GRU weights, allowing the model to learn a general form representation applicable across all clubs.

**Prediction Head.** The outputs of the metadata encoder, H2H encoder, and both team-form encoders are concatenated and passed through a feed-forward network that outputs

$$[\hat{g}_1, \hat{g}_2]$$

representing the predicted goals for Team 1 and Team 2. A softplus activation ensures all predictions are non-negative.

### 4.2 Training Procedure

We train the model chronologically. All Premier League seasons from 2010–11 through 2023–24 are used for training, and the 2024–25 season serves as the held-out validation set. Batch construction and feature engineering are handled by our dataset builder (`build_dataset.py`), which generates team IDs, historical match sequences, and ground-truth goal labels.

Because the task is goal regression rather than multi-class classification, we use the Poisson Negative Log-Likelihood Loss.

which is appropriate for football scores that typically follow Poisson-like distributions in statistical literature. This choice is also supported by our experimentation with multiple loss functions (detailed later), where PoissonNLLLoss provided superior validation performance.

We optimize the model using Adam with a learning rate of $3 \times 10^{-4}$ and weight decay of $1 \times 10^{-4}$.

### 4.3 Baselines

We compare our model against two baselines:

- **Majority baseline:** predict the most common outcome.
- **Progress-report RNN model:** a single-sequence RNN using only betting odds and evolving hidden states.

Our updated model outperforms both baselines across exact-score accuracy and Win–Draw–Loss (WDL) accuracy. Because the new architecture incorporates richer contextual information—team identity, matchup history, and recent form—it produces predictions that are both more stable and more reflective of real football dynamics. This demonstrates that adding structured encoders meaningfully improves predictive power over both naive and earlier single-encoder approaches.

### 4.4 Model Variants and Ablations

Throughout development, we experimented with several architectural and training variations to understand how design choices affected predictive performance. Instead of removing entire encoders, our primary ablations focused on two dimensions: loss function selection and layer size/depth tuning.

**1. Loss Function Experiments.** Our early experiments used Mean Squared Error (MSE) for goal regression, but this produced unstable gradients and disproportionately penalized high-scoring matches. We then tested alternatives, including:

- **L1 Loss**, which improved robustness to outliers but resulted in overly conservative goal predictions.
- **Poisson Negative Log-Likelihood (PoissonNLLLoss)**, which assumes a Poisson-like distribution typical of football scoring and produced the most stable and realistic score predictions.

After comparison, PoissonNLLLoss consistently yielded higher WDL accuracy and smoother training dynamics, and therefore became our final loss function.

**2. Layer Size and Hidden Dimension Tuning.** We also varied the size of the GRU hidden layers, team embeddings, and the depth of the feed-forward prediction head to balance model capacity and overfitting. Key variations included:

- Increasing team ID embedding sizes from $16 \rightarrow 32 \rightarrow 64$
- Increasing GRU hidden sizes for form and H2H encoders ($32 \rightarrow 64 \rightarrow 128$)
- Adjusting the feed-forward block from a shallow 1-layer MLP to a deeper 2–3 layer network
- Testing dropout rates between 0.1 and 0.4

These experiments showed that hidden dimensions of 64 and a two-layer feed-forward head provided the best performance without overfitting. Larger networks (e.g., 128–256 hidden units) offered marginal gains but increased training time and tended to overfit high-scoring outliers.

Overall, these ablations refined the architecture and confirmed that performance is most sensitive to the choice of loss function and the expressive capacity of the encoders.

## 5 Results and Evaluation

### 5.1 Evaluation Strategy

Our evaluation strategy employs a chronological train-validation split that respects the temporal nature of football match data. This approach simulates realistic prediction scenarios where historical data is used to forecast future outcomes.

**Train-Validation Split:** We use a strict temporal split where all matches from seasons 2010–11 through 2023–24 serve as the training set, comprising approximately 5,320 matches (14 complete seasons). The validation set consists of all 380 matches from the 2024–25 season, representing a complete, unseen season. This split strategy ensures that the model is evaluated on future matches relative to its training data.

We do not employ a separate test set, as the validation set already represents a complete season that is temporally distinct from the training data. The validation set provides a robust evaluation that tests the model's ability to generalize to new seasons with potentially different team compositions, managerial changes, and evolving league dynamics.

**Cross-Validation:** We do not use cross-validation for several reasons. First, the temporal nature of football data requires maintaining strict chronological order; random or k-fold splits would violate this temporal structure and create unrealistic evaluation scenarios. Second, season-level integrity is important: teams change composition between seasons, and splitting within seasons could create issues. Third, our validation set represents

a complete, unseen season, which provides a more realistic assessment of model performance than cross-validation on historical data.

**Label Distributions:** The validation set demonstates a natural class imbalance typical of football match outcomes (labels). The three outcome classes are: Home win, Draw, and Away win. The distribution heavily favors Home with an average of around 40% of the matches being Home wins. The Draw class is the least frequent with an average of around 25% of the matches being Draws. The Away class is the most frequent with an average of around 35% of the matches being Away wins. This imbalance reflects the difficulty of predicting draws, which are the least frequent outcome in Premier League matches. The training set exhibits similar distributions across the 14 seasons, ensuring that the model learns from representative class proportions.

### 5.2 Evaluation Metrics

We evaluate our model using a comprehensive set of classification metrics appropriate for multi-class prediction of match outcomes. These metrics provide complementary perspectives on model performance:

- **Accuracy**: The overall proportion of correctly predicted match outcomes
- **Precision**: The proportion of predicted positive cases that are actually positive, computed per class
- **Recall**: The proportion of actual positive cases that are correctly identified, computed per class
- **F1-score**: The harmonic mean of precision and recall, providing a balanced metric

These metrics are computed for each of the three outcome classes, as well as macro-averaged and weighted-averaged across all classes.

### 5.3 Results

Table 2 presents the classification performance metrics for our model on the 2024–25 validation season. The model achieves an overall accuracy of 51.58% (196 out of 380 matches correctly predicted). The per-class metrics reveal significant performance variation across outcome types.

The model demonstrates its strongest performance on Home wins, achieving a precision of 0.57, recall of 0.72, and F1-score of 0.64. This reflects the model's ability to leverage home advantage signals from betting odds and historical sequences. Performance on Away wins is also strong, with precision of 0.65 and recall of 0.41, resulting in an F1-score of 0.50. The model struggles most with Draw predictions, achieving the lowest metrics across all classes (precision 0.29, recall 0.32, F1-score 0.31), which is consistent with the difficulty of predicting draws in football.

| Class | Precision | Recall | F1-score | Support |
|---|---|---|---|---|
| Home | 0.57 | 0.72 | 0.64 | 155 |
| Draw | 0.29 | 0.32 | 0.31 | 93 |
| Away | 0.65 | 0.41 | 0.50 | 132 |
| **Accuracy** | | 0.5158 (196/380) | | |
| **Macro Avg** | 0.51 | 0.48 | 0.48 | 380 |
| **Weighted Avg** | 0.53 | 0.52 | 0.51 | 380 |

Table 2: Classification metrics for match outcome prediction on the 2024–25 validation season.

### 5.4 Adequacy of Metrics

The metrics selected for evaluation have proven adequate for assessing model performance on this multi-class classification task. The combination of accuracy, precision, recall, and F1-score provides a comprehensive view of model performance, while the confusion matrix offers detailed insights into misclassification patterns. These metrics effectively capture both overall performance and class-specific challenges, particularly the difficulty of predicting draws.

Compared to the progress report stage, our current metrics remain consistent, but the results reflect improvements in model architecture and feature engineering. The progress report metrics revealed a severe bias toward Home win predictions, with zero Draw predictions. While the current model still struggles with Draws, it now makes some correct Draw predictions (28 out of 93), indicating progress in addressing class imbalance. The metrics highlight areas for improvement, particularly the need for better Draw prediction capability, which could be addressed through techniques such as more sophisticated sequence modeling approaches.

## 6 Progress

In the progress report we promised to strip post match stats, keep only prematch features, and use

a chronological split. We followed through: the dataset now uses only pre kickoff odds plus outcome derived histories, built in time order to avoid leakage, and we train on past seasons with the final season held out. We did pivot on modeling: instead of the simple odd only RNN, we built a multi encoder GRU (team form, head-to-head, team/home embeddings, odds MLP) and predict scorelines, deriving W/D/L for metrics. We also added a validation league table as planned. What slipped: we didn't run a full hyperparameter sweep or calibration, and draw handling is still weak, however it has improved over previous draw odds.

## 7 Error Analysis

We examine errors on the held-out 2024–25 season using the same chronological split described in Section 5.3. Accuracy is 51.58% (196/380), but class-wise behavior varies (Table 2). The confusion matrix (Fig. 2) is our primary diagnostic.

**What the model gets right.** Home wins are the strongest class (P/R/F1 = 0.57/0.72/0.64). Away wins are middling (0.65/0.41/0.50). Clear home-favored fixtures and lopsided matches tend to be predicted correctly.

**What the model gets wrong.** Draws remain the weakest (0.29/0.32/0.31). In the confusion matrix, 47/93 true draws are pushed to Home and 18/93 to Away. Away wins are also overcalled as Home (42/132) or Draw (42/132). This shows a slight home-lean and difficulty with balanced fixtures.

**Patterns and bias.** Most misclassifications cluster around close games: draws are converted to wins, and away wins are nudged toward home. The model captures home-advantage signals but is overconfident in tight matches.

**How to address it.** To improve draws and reduce home bias, we would (1) rebalance the loss (class weights or focal loss), (2) add a draw-aware head (joint goal and outcome prediction), and (3) calibrate outputs to temper overconfident home predictions. Additional diagnostics we would include are residual histograms for goal errors and a short list of worst-missed fixtures (high goal error or wrong outcome) to target data or loss tweaks.

Figure 2 visualizes the confusion matrix, revealing detailed patterns in prediction errors. The matrix shows that the model correctly predicts 112 Home wins but misclassifies 32 Home wins as Draws and 11 as Away wins. For Draws, the model correctly identifies only 30 out of 93 cases, with 45 misclassified as Home wins and 18 as Away wins. For Away wins, the model correctly predicts 54 cases, with 38 misclassified as Home wins and 40 as Draws. This pattern suggests that the model has a slight bias toward predicting Home wins, which reflects the home advantage effect captured in the training data.

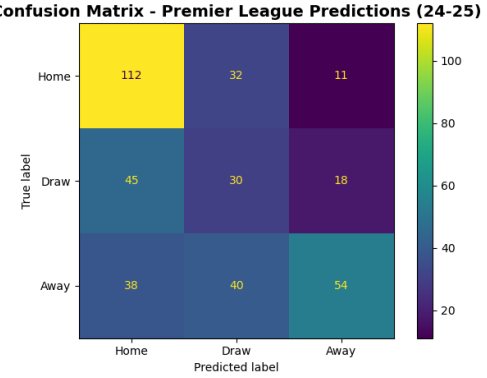


Figure 2: Confusion matrix for match outcome predictions on the 2024–25 season. Rows are true labels, columns are predicted labels.

## Team Contributions

The project was an equal collaboration between the three of us, with each member contributing roughly one-third of the overall work. We worked closely through frequent calls and shared debugging sessions, iterating on architectural choices and evaluation methods together. Omar El-Aref focused primarily on building and organizing the end-to-end training pipeline, including the chronological data split, evaluation metrics, Poisson-based loss, and draw-margin logic. Omar Abdelhamid contributed heavily to model tuning and stability, refining hyperparameters, experimenting with hidden dimensions, and improving validation behaviour through standardized preprocessing and optimized training settings. Khalid concentrated on data preprocessing and feature construction, ensuring the Premier League tables were consistently formatted across seasons and implementing the match-history tensors used by the GRU encoders. Overall, the workflow was highly collaborative, with all three members contributing to model design decisions, testing, and iterative improvements.

## References

Rie Kubota Ando and Tong Zhang. 2005. A framework for learning predictive structures from multiple tasks

and unlabeled data. *Journal of Machine Learning Research*, 6:1817–1853.

Galen Andrew and Jianfeng Gao. 2007. Scalable training of L1-regularized log-linear models. In *Proceedings of the 24th International Conference on Machine Learning*, pages 33–40.

Rory P. Bunker and Fadi Thabtah. 2019. A machine learning framework for sport result prediction.

Ashok K. Chandra, Dexter C. Kozen, and Larry J. Stockmeyer. 1981. Alternation. *Journal of the Association for Computing Machinery*, 28(1):114–133.

Football-Data.co.uk. 2025. Football-data.co.uk: Historical football results and betting odds data. Accessed: November 9, 2025.

Dan Gusfield. 1997. *Algorithms on Strings, Trees and Sequences*. Cambridge University Press, Cambridge, UK.

Mei-Ling Huang and Yun-Zhi Li. 2021. Use of machine learning and deep learning to predict the outcomes of major league baseball matches.

Mohammad Sadegh Rasooli and Joel R. Tetreault. 2015. Yara parser: A fast and accurate dependency parser. *Computing Research Repository*, arXiv:1503.06733. Version 2.