# Predicting NWSL 2023 Match Outcomes from Halftime Data

## 1 Data Exploration and Preprocessing

### 1.1 Dataset Acquisition and Overview

The dataset consists of manually collected halftime statistics and final game results for all available National Women's Soccer League (NWSL) games from the 2023 season, including playoff matches. Each game is represented as two observations, one for each team, recording halftime statistics and their corresponding match outcomes (win, loss, or tie). The data was manually entered from `https://www.fotmob.com/leagues/9134/overview/nwsl` and saved as a CSV file for analysis. Observations with missing halftime statistics were excluded. The dataset also included a `full_time_goals` column, which was dropped as it would not be available at halftime for prediction purposes.

### 1.2 Initial Data Cleaning

Duplicate columns (e.g., variables ending in `.1`) and those with high proportions of missing values (e.g., `touches_in_opp_box`) were removed to ensure dataset quality. The target variable, `result`, was separated from the input features, and features were standardized using `StandardScaler` to normalize the data. Standardization helped ensure that no single feature dominated due to differences in scale, improving the performance of machine learning models. These preprocessing steps provided a clean and well-structured dataset for further analysis.

### 1.3 Exploratory Data Analysis

Several statistical methods were employed to understand the dataset and determine the most important features:

### 1.3.1 Mutual Information (MI)

MI analysis identified `total_shots` (MI = 0.136), `first_half_goals` (MI = 0.115), and `corners` (MI = 0.098) as the most informative features. Features like `big_chances_missed` and `successful_dribbles` had an MI score of 0.000, suggesting no dependency with the target variable.

### 1.3.2 ANOVA (F-Statistic and p-values)

The ANOVA test confirmed statistical significance for features such as `first_half_goals` (F = 28.37, p ¡ 0.0001) and `shots_on_target` (F = 22.75, p ¡ 0.0001). Conversely, features like `ball_possession` and `throws` exhibited very high p-values, indicating negligible impact on the target variable and warranting exclusion.

### 1.3.3 Random Forest Feature Importances

A Random Forest classifier revealed `total_shots` (importance = 0.046) and `first_half_goals` (importance = 0.045) as the most impactful predictors. Features like `duels_won` and `successful_dribbles_percent` also showed moderate importance, while `red_cards` and `hit_woodwork` had minimal influence.

## 1.4 Feature Selection

To refine the dataset, a composite scoring system was used, combining normalized MI scores, F-statistics, and Random Forest importances. The top five features identified were:

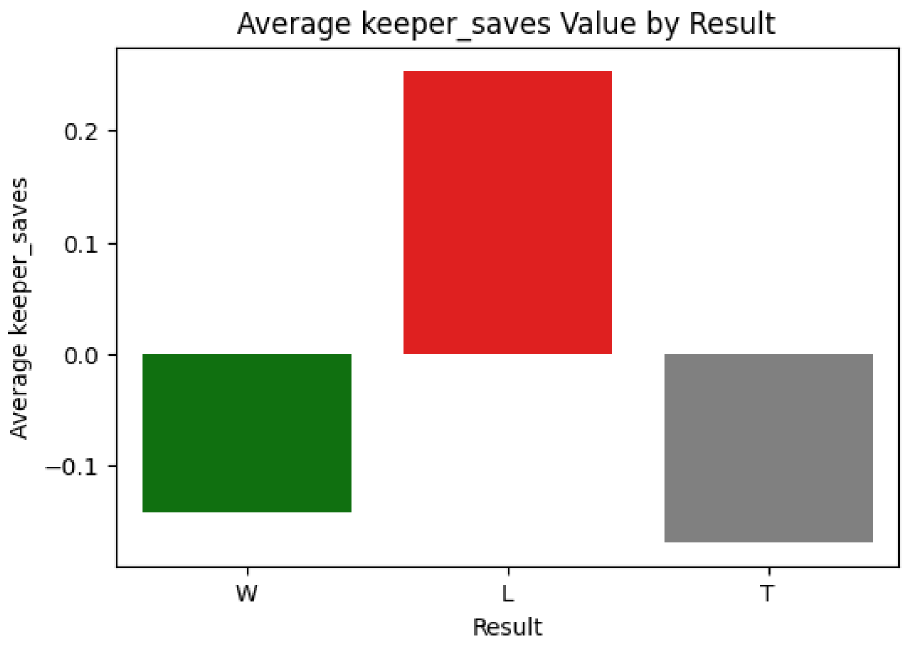
- Total Shots
- Shots on Target
- First Half Goals
- Shots Inside Box
- Keeper Saves

These features not only ranked highly across multiple analyses but also aligned with domain knowledge—for instance, factors such as `total shots` or `first-half goals` are well-recognized indicators of success in soccer, as teams that take more shots or establish an early lead are generally more likely to win. This alignment supports their inclusion for predictive modeling, as it ensures the model captures meaningful and contextually relevant patterns, rather than relying solely on statistical correlations.

## 1.5 Visualizations and Insights

Visualizations reinforced the relationships between features and outcomes. For example, higher `keeper_saves` correlated with losses, likely indicating defensive pressure due to frequent opposition attacks. These insights helped contextualize statistical findings and guided feature prioritization.

The following bar chart (`ks_bc.png`) illustrates the distribution of `keeper_saves` across different match outcomes (win, loss, tie). The chart shows a clear trend where teams with higher numbers of saves were more likely to lose, which supports the hypothesis that frequent defensive action correlates with negative outcomes.



# 2 Model Selection and Training

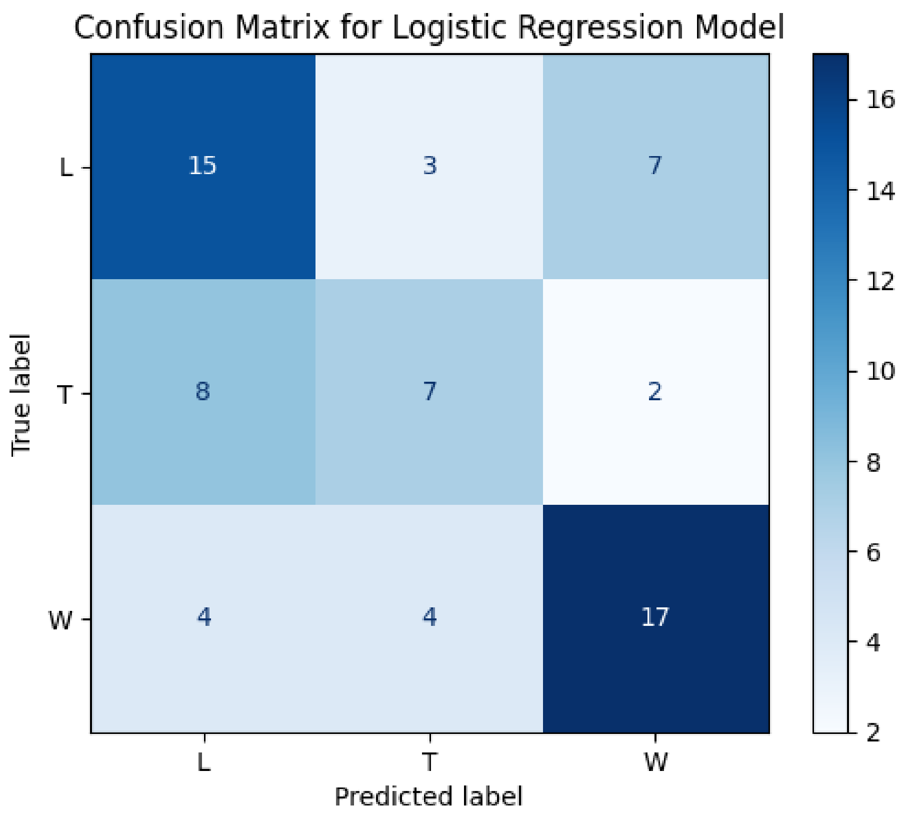
## 2.1 Data Split and Initial Model Testing

To begin the model development process, the dataset was split into training and testing sets using stratification. This approach was chosen because of the class imbalance present in the data, where certain outcomes (such as ties) are less frequent than others. Stratification ensures that both the training and testing sets maintain a similar distribution of classes. We performed this split for both the full feature set and the reduced top 5 feature set.

## 2.2 Logistic Regression Model

The initial classification model chosen for evaluation was logistic regression. The model was first trained using all available features and then using only the five most informative features identified during the exploratory data analysis. The performance of these two models was compared to assess the impact of reducing feature noise and focusing on the most important variables.

The logistic regression model using all features achieved an overall accuracy of approximately 45%, whereas the model using only the top 5 features demonstrated a marked improvement, achieving 55% accuracy. This suggested that simplifying the model and limiting it to the most informative features helped reduce noise and improved performance. Importantly, there was little difference between the accuracy of the training set (52%) and the test set (55%), indicating that the model was not overfitting.



## 2.3 Neural Network Model

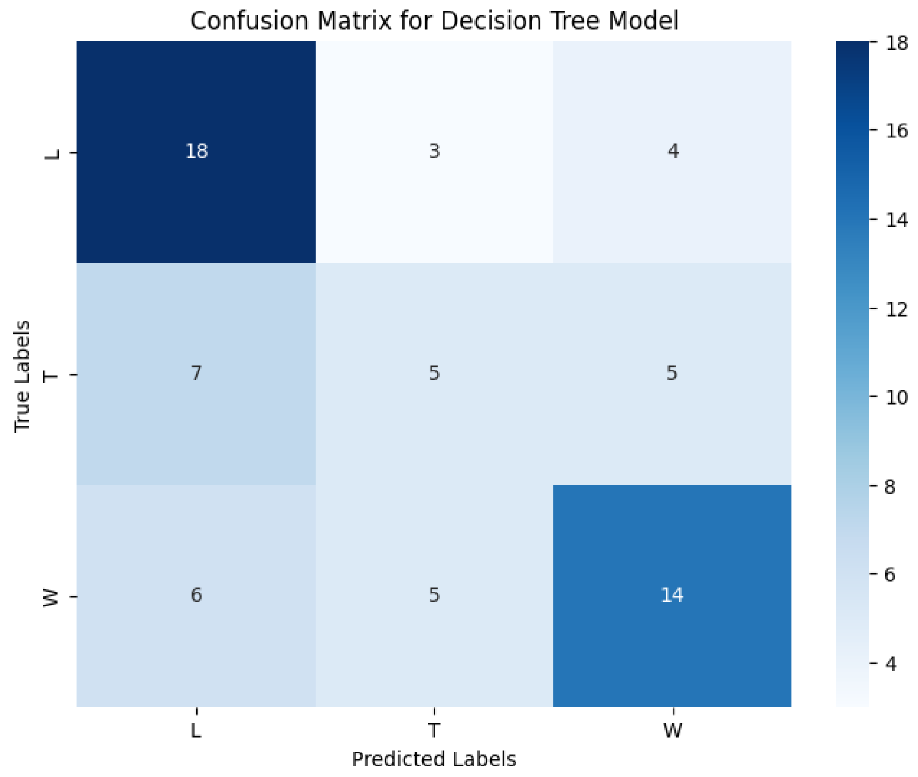
Next, we explored the performance of a neural network for classification. Neural networks require numerical outputs, so the game results were encoded as 0 for Loss, 1 for Tie, and 2 for Win. The initial neural network model achieved a test accuracy of 53.7% with a training accuracy of 57%. These results suggested that while the model was able to generalize reasonably well, there were noticeable signs of overfitting, as the training accuracy was higher than the test accuracy, indicating that the model might not perform as well on unseen data. The model's loss function (0.97 for the test set, 0.89 for the training set) reinforced the idea that the model's predictions were generally accurate, though it still had room for improvement. Additionally, there were relatively few ties predicted, which is a common challenge in neural network training as the model may favor more dominant classes.

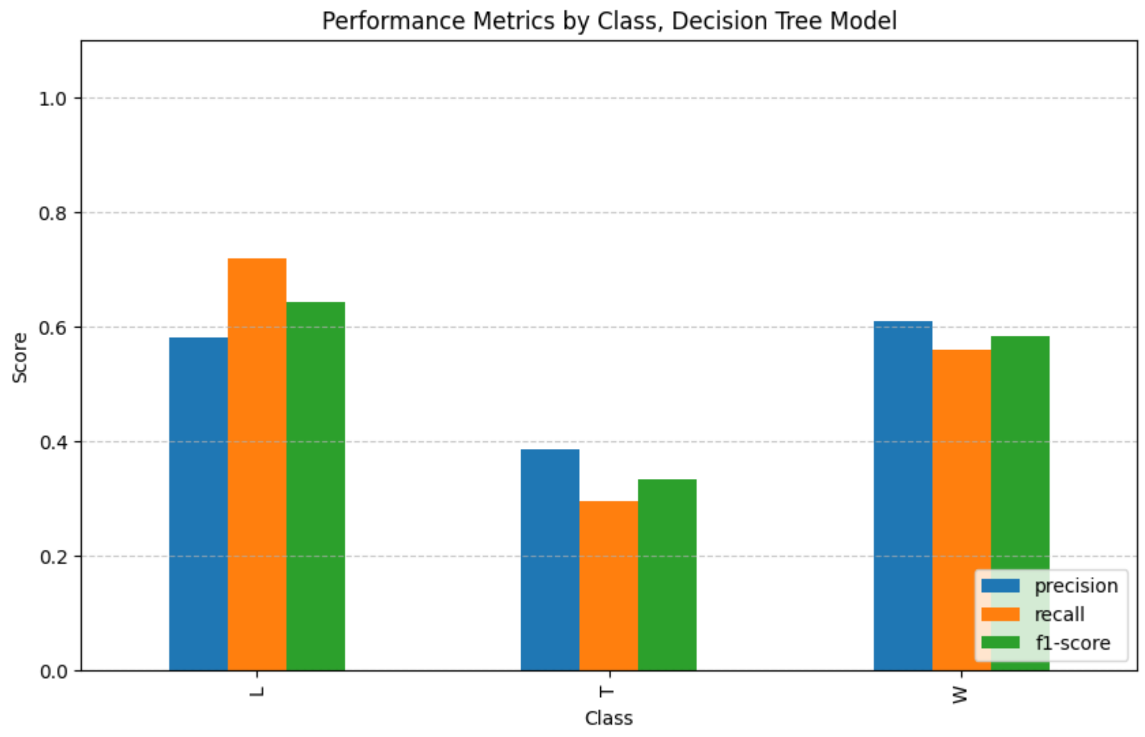
An updated version of the neural network was trained with additional layers, more neurons, and dropout layers to prevent overfitting. However, the test accuracy dropped to 50.7%, while the training accuracy increased to 62.3%. This indicated that overfitting was still present, and the tie class remained a challenging prediction for the network. Given these results, the neural network did not outperform other models, and further adjustments did not improve the model sufficiently. Ultimately, we decided to exclude the neural network model from further consideration.

## 2.4 Decision Tree Model

A decision tree model was created next, with no limitations on tree depth. The initial model showed high training accuracy (92.9%) but relatively poor test accuracy (55%), signaling overfitting. To address this, the hyperparameters were adjusted using `GridSearchCV` to find the optimal values for parameters such as `max_depth`, `min_samples_split`, and `min_samples_leaf`. Additionally, pruning was applied using `ccp_alpha` to prevent the tree from becoming too complex.

After tuning the decision tree, the model showed a much more balanced performance. The training accuracy decreased to 53.27%, and the test accuracy improved to 62.69%, indicating better generalization. The confusion matrix revealed that the decision tree was particularly effective at predicting wins (with a high recall of 0.84) but had some difficulty distinguishing between losses and ties. Despite some misclassifications, the model showed solid performance overall.





This bar chart visualizes the performance metrics—precision, recall, and F1-score—for three classes (`L`, `T`, `W`). Here's a brief interpretation:

- **Class** `L`: This class has relatively balanced performance, with recall being slightly higher than precision. The F1-score, as expected, falls between the precision and recall values.
- **Class** `T`: Performance for this class is notably lower compared to the others. All metrics (precision, recall, and F1-score) are considerably reduced, indicating challenges in predicting this class accurately.
- **Class** `W`: This class demonstrates strong and consistent performance across all three metrics, with precision, recall, and F1-score all being high and closely aligned.

Overall, the model struggles the most with class `T` predictions, likely due to class imbalance or overlapping features, and performs most consistenyly on class `W`. Future efforts to improve performance might focus on handling class `T` specifically, such as rebalancing the dataset or enhancing feature representation.

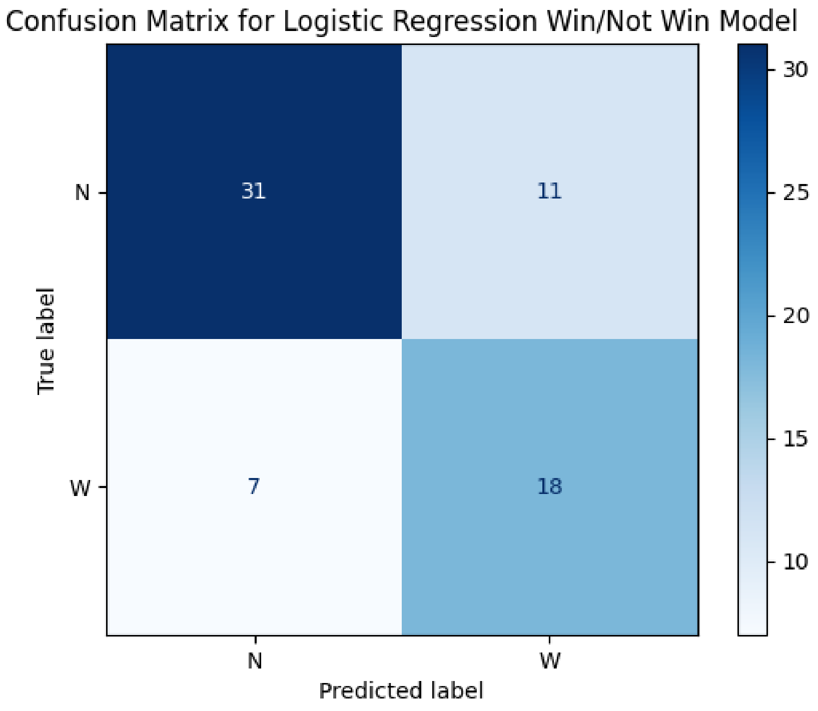
## 2.5 Binary Classification: Win vs. Non-Win

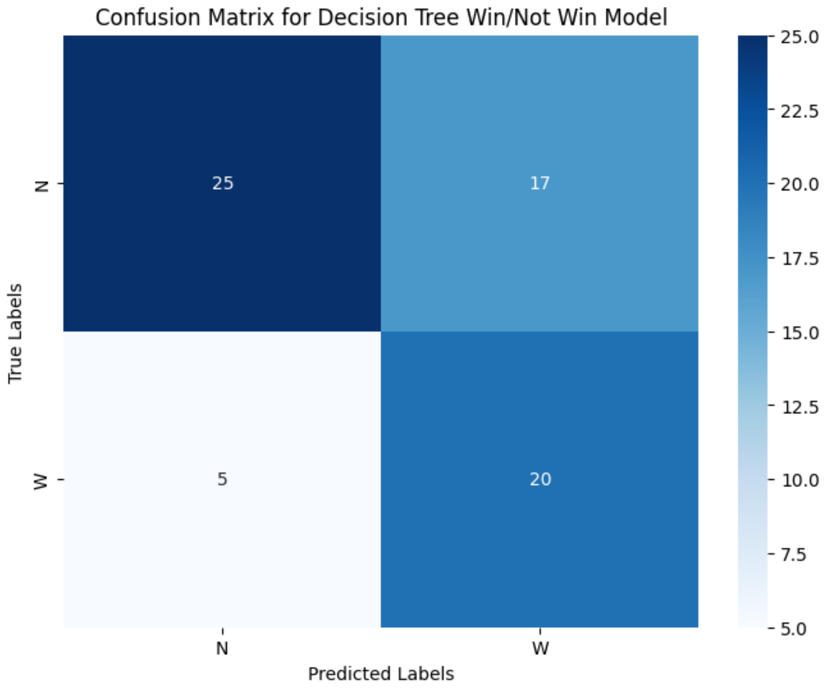
At this stage, we focused on implementing binary classification models for predicting a `win` versus a `not win` outcome. This is particularly useful in practical applications, as ties are less significant in competitions such as playoffs, where ties are not allowed, and winning holds more importance. The binary classification scenario also reflects real-world strategic decisions, where teams often focus on securing a win rather than a tie.

The logistic regression model performed well with a test accuracy of 73.13%. It was particularly effective at predicting non-wins (with a precision of 0.80 and recall of 0.76), though its ability to predict wins (with a precision of 0.63 and recall of 0.68) was somewhat limited. The model's accuracy was consistent with the training set (67.84%), suggesting that it generalized well to new data without overfitting.

In contrast, the decision tree model achieved a test accuracy of 71.64%, with a training accuracy of 71.86%. The model's confusion matrix indicated that it was more successful in predicting wins, with a high recall of 0.84, but struggled with predicting non-wins, as evidenced by the lower recall of 0.64 for the `not win` class. The precision for the `not win` class was high (0.87), but the precision for wins was lower (0.58), which suggests that the model is more likely to misclassify wins as non-wins.

Both models showed solid performance with no signs of overfitting, and each model had its strengths. The logistic regression model excelled at predicting non-wins, while the decision tree model was better at predicting wins. Depending on the specific application, either model could be more suitable.


Confusion Matrix for Logistic Regression Win/Not Win Model
True label
N
W
31
11
7
18
N
W
Predicted label
30
25
20
15
10


Confusion Matrix for Decision Tree Win/Not Win Model
True Labels
N
W
25
17
5
20
N
W
Predicted Labels
25.0
22.5
20.0
17.5
15.0
12.5
10.0
7.5
5.0

## 2.6 Practical Applications of the Models

In practical applications, these models can be valuable tools for analyzing games in real-time. By using statistics from the first half—such as total shots, shots on target, first half goals, shots inside the box, and keeper saves—commentators and coaching staff can gain immediate insights into the dynamics of the game. For example, the logistic regression model, which excels in predicting non-wins, can help assess a team's likelihood of failing to secure a win. This insight allows analysts to discuss the team's potential for a comeback and can inform coaching staff on tactical adjustments, such as focusing on increasing shot accuracy or defensive strategy, depending on the model's predictions.

Meanwhile, the decision tree model, with its strong performance in predicting wins, could be particularly useful for teams in high-stakes situations where securing a win is crucial. This model's ability to predict wins with a high degree of accuracy can help coaches understand their team's potential for victory, suggesting strategies to enhance their chances—whether it's by increasing shots inside the box or focusing on key defensive plays to limit the opponent's scoring opportunities. The decision tree's balanced precision and recall also provide actionable insights for teams looking to fine-tune their approach based on their current position in the game.

Both models can be applied in various ways, such as guiding in-game decisions, supporting commentators in their analysis, or enhancing the fan experience by offering real-time predictions. The insights from these models make it easier for coaching staff to react and adjust strategies during halftime, improving their ability to plan for the second half. Additionally, the models are adaptable for future seasons, as they can be updated with new data, ensuring they remain relevant and effective tools for analyzing and predicting game outcomes.

## 2.7 Conclusion

After testing multiple models, both logistic regression and decision tree classifiers showed promising results. The logistic regression model excelled in predicting non-wins, while the decision tree model was better suited for predicting wins. These findings highlight the trade-offs involved in model selection, as the best choice depends on the specific goals of the analysis. The logistic regression model's strong precision for non-wins makes it ideal for analyzing general game outcomes, while the decision tree's ability to prioritize wins is particularly useful for playoff predictions and competitive scenarios. Both models demonstrate the potential for valuable, actionable insights in sports prediction, making them viable tools for real-time analysis and strategic decision-making.

# 3 Challenges to Overcome and Future Work

## 3.1 Challenges

One of the biggest challenges we faced was data acquisition. The National Women's Soccer League (NWSL) does not have comprehensive public datasets available, and despite reaching out to the league directly, we never received a response. This meant we had to manually enter the data for the first-half statistics and game outcomes for every match in the 2023 season, including playoff games. Each game counted as two observations: one for Team A's first-half stats and result (win, loss, or tie) and one for Team B's stats and result.

This process was time-consuming and repetitive, and there was always the risk of errors from manual data entry. To address this, we dedicated time to carefully reviewing and double-checking all entries for accuracy. Initially, we had hoped to collect data from multiple seasons to provide a more robust analysis, but we adjusted our expectations and focused on the 2023 season. While we ended up with over 250 observations, which was sufficient for the project, the scope was narrower than we had originally envisioned.

Another challenge was cleaning the dataset. Some statistics were duplicated or unavailable for certain observations, requiring careful review to ensure the data was usable and consistent.

The next hurdle was selecting the right features for the predictive models. To address this, we used several techniques: **Mutual Information Scores**, **F-statistics with p-values**, and a **Random Forest model with feature importance**. These methods provided insight into which features were most predictive of game outcomes. However, narrowing down the features and comparing their effectiveness was initially confusing. We wanted to identify statistically significant features based on p-values and then standardize all three metrics to score and rank the features. We weren't sure how to implement this process in code, so we utilized ChatGPT to help write and refine the necessary scripts.

Class imbalance in the dataset was another obstacle. The data included 100 wins, 100 losses, and only 66 ties, which posed challenges for model accuracy, especially in predicting ties. To resolve this, we used stratification during the train-test split and applied `class_weight = 'balanced'` in our logistic regression model. This ensured the model did not overlook ties, which it initially ignored due to their smaller representation.

The neural network classification model presented significant difficulties and, ultimately, was not included in our final results. Despite various attempts to improve its performance—such as adding more layers, introducing dropout to prevent overfitting, and adjusting hyperparameters—it rarely predicted ties correctly. Overfitting remained a persistent issue, with the training set significantly outperforming the testing set, so we decided to focus on other models.

Overfitting was also a challenge with our Decision Tree model initially. To address this, we used `GridSearchCV`, which systematically tested different parameter combinations to find the optimal settings for the model. This significantly improved performance and allowed us to adapt the models for playoff scenarios, where we redefined the target variable as `win` or `not win` (loss or tie). ChatGPT was instrumental in helping us write and implement the GridSearchCV code.

## 3.2 Future Work

Moving forward, the model could be expanded to include additional features, such as `team` composition, player statistics, and previous match history. Incorporating a `team` categorical variable would be particularly interesting, as different teams often have distinct playing styles that influence how the features predict outcomes. This could enable the creation of "mini" models tailored to each team, identifying which features are most important for specific teams and providing more targeted feedback for coaches or analysts. However, this approach might be less useful for modeling an entire season, as it could lead to predictions heavily biased by a team's historical performance, rather than accounting for changes from season to season.

Furthermore, the binary classification approach ("win vs. not win") could be extended to predict the exact scoreline of the game, introducing a regression aspect to the model. Exploring advanced ensemble methods like XGBoost or deep learning models could also improve performance. Additionally, experimenting with cross-validation techniques to further validate model performance and avoid potential overfitting would be beneficial for future iterations of this project.

By incorporating these improvements, future versions of the model could provide even more accurate and actionable insights for predicting game outcomes.

# AI Disclaimer

Generative AI was used to assist with writing code for comparing feature selection methods by scaling different performance metrics, improving visualizations of feature comparisons, troubleshooting errors in the neural network model, brainstorming ideas for model improvements, and creating the code for the grid search method to select hyperparameters for the decision tree. All other code, interpretations, explanations, and this report are my own.