STATS - 101C Project Report

Predicting NBA Game Outcomes: An Analysis of Key Metrics and Performance Indicators

Xianya Fu (UID: 406178735)
Ngan Poon (UID: 306284963)
Yuer Tang (UID: 005959967)
Mingye Wang (UID: 705766569)
Edith Chan (UID: 806284772)

December 13, 2024

Contents

1 Introduction

This paper focuses on predicting the outcomes of NBA games during the 2023-2024 season [3] by analyzing data from previous games within the same season. The goal is to leverage various key metrics and performance indicators to train and evaluate statistical models, ultimately identifying the most accurate models to predict game results.

In addition to the features provided in the dataset, we introduced several engineered features to enhance model performance: Home Advantage, Stability, Previous Competitions, Weighting Statistics, and Upset Factor, all derived from the given data. These features were designed to capture the performance dynamics of a team as a home and guest team, with some indicators being more predictive than others.

Following feature engineering and selection, we applied various statistical methods to train and test the models. We also used k-fold cross-validation to ensure the robustness of our experiments. Most models achieved an accuracy of around 70%, with XGBoost slightly improving the performance to 71%. Among all models evaluated, Random Forest and AdaBoost demonstrated the best performance, achieving an accuracy of 72%, representing an improvement of 8% over the baseline model.

2 Data Preprocessing

2.1 Transformed features

According to the original dataset, each row represented the performance of a single team in a specific game. The `Matchup` details were denoted as either `"Team A vs. Team B"` or `"Team A @ Team B"`, where `"vs."` indicated that Team A was the home team, and `"@"` indicated that Team A was the visiting team. To simplify the analysis, the W/L column in the original dataset was converted to a binary format: `1` indicated that Team A won, and `0` indicated that Team A lost (and Team B won).

Since our goal is to use the model to predict the outcome of a game using only the difference in the two teams' data available prior to the game day, we then transformed the raw dataset into a comparative format to highlight the relative performance of the two teams in each matchup. For each game, previous matches were grouped based on the two teams involved (e.g., Team A and Team B), and a new single row was created to represent the game. This row contained the differences in the average statistics between the two teams, calculated based on their performance before the match date. Specifically, the historical averages for all relevant statistics were computed separately for each team, considering only games that occurred before the current match date. The differences were calculated as follows:

$$\text{Difference} = \text{Average Team A Statistics} - \text{Average Team B Statistics} \tag{1}$$

We repeatedly apply (1) to all selected features, ensuring that the transformed dataset provides a clear comparison of the two teams' past performances. The features selected for this transformation included all game statistics such as `PTS`, `FGM`, `FGA`, `FG%`, `3PM`, `3PA`, `3P%`, `FTM`, `FTA`, `FT%`, `OREB`, `DREB`, `REB`, `AST`, `STL`, `BLK`, `TOV`, `PF`, and `+/-` [3]. After transformation, the new dataset contained columns representing the transformed features, the binary target label (1 for a Team A win, 0 for a Team A loss), team identifiers, matchup details, game date, and total minutes played.

To ensure the integrity of the calculated features, rows, where sufficient historical data was unavailable (e.g., the first 300 rows with limited prior game information) were excluded from the dataset. This preprocessing step ensured that all rows in the dataset had meaningful and accurate feature values for modeling and prediction.

2.2 New engineering features

To enhance the dataset's predictive power, we added additional features that were not in the original dataset. These features are designed to provide additional insight into background factors and historical trends that may have influenced the game's outcome. They are listed and explained below:

`Home Advantage`: This is a binary variable derived from the `Matchup` column in the original dataset. If the column entry contains the string `"vs."`, it indicates that Team A is the home team, and the `Home Advantage` variable is assigned a value of `1`. In contrast, if the entry contains `"@"`, it means that Team A is the guest team, and the Home Advantage variable is assigned a value of `0`. Figure 1 illustrates the outcomes of 1,230 matches and shows the distribution of home team wins (label = `1`) and losses (label = `0`). It is evident that the distribution is somewhat imbalanced, which suggests that the home advantage indeed plays a role. Therefore, we decided to include this variable in our analysis.

`Stability`: We aim to take into account how consistent a team's performance is across all features. We do this by taking the mean of each feature and dividing it by its variance, then finally adding across all features. We do not divide by zero. More consistent teams will have a higher stability rating.

`Weighting Statistics`: It would make sense that games played closer to the predicted game date would impact team performance more. We implemented an exponential weight function that utilizes a parameter α and takes into account the difference in dates between previous games and the current game by the time difference in days. Every feature mentioned above that isn't an additional feature had its means calculated using this weight function.

`Previous Competitions`: This feature aims to take into account the history of any two teams. We named this variable the prior win rate (`PWR`). Suppose we have Team A as our POV team playing against Team B; then the `PWR` of Team A against Team B is calculated as the proportion of games Team A has won against Team B across all games they have played each other in the past.

`Win/Loss Rate`: Given a row R, the win rate of all games for Team A is calculated prior to the date of R. We also calculate the win rate of all games for Team B. The difference in win rate between Team A and Team B is then subtracted.

`Upset Factor`: This is not a new variable but rather a weight designed to emphasize past wins given home team advantage (the Home Advantage variable only considers the current game, whereas this variable addresses past games). When calculating Team A's win rate, if Team A wins a game as the guest team, that win is counted as 1.5 wins as opposed to 1 win. This is meant to emphasize winning when not on home advantage.

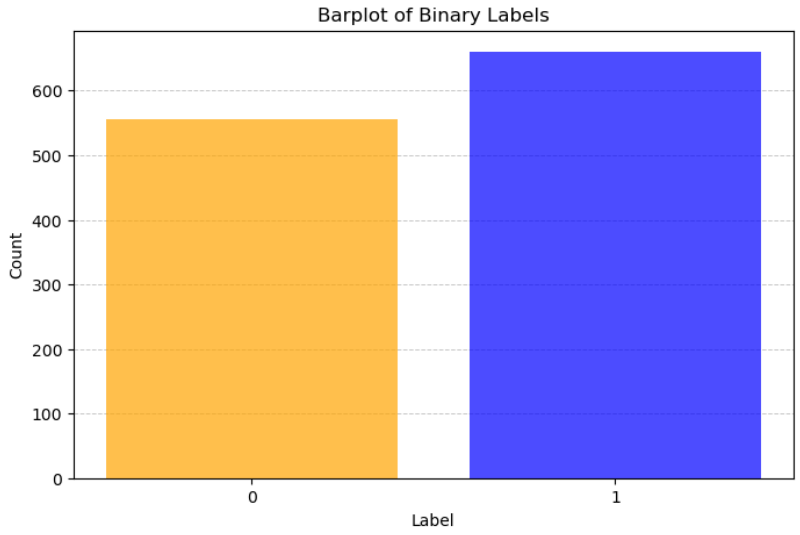

Figure 1: Distribution of Home Teams' Matches Result

2.3 Z-Score Normalization

To ensure that all features contribute equally to the model and to improve model performance, we apply z-score normalization [5] on the dataset. This preprocessing step is crucial because the dataset contains features on different scales, such as percentages (e.g., `FG%` between 0 and 1) and scores (e.g., `PTS` in the hundreds). Without normalization, a larger range of features may dominate the model training process, leading to biased predictions. Z-score normalization is implemented using the formula:

$$Z = \frac{X - \text{mean}}{\text{standard deviation}} \tag{2}$$

This process transforms all numerical features in the newly transformed dataset into intervals with a mean of 0 and a standard deviation of 1. Compared to normalization, this method is less sensitive to outliers, ensures comparability between features, eliminates the effect of scale differences, and facilitates the convergence of algorithms sensitive to feature scales, such as logistic regression and support vector machines.

2.4 Table of features

Table 1: Summary of feature engineering steps

Original	Preprocessing Action	Transformed
Team	Unchanged	Team
Matchup	Unchanged	Matchup
Game Date	Unchanged	Game Date
MIN	Unchanged	MIN
W/L	Converted into binary 0, 1	Label
Label	Calculated weighted difference and standardized	W/L Difference
PTS	Eliminated during feature selection	
FGM	Calculated weighted difference and standardized	FGM Difference
FGA	Calculated weighted difference and standardized	FGA Difference
FG%	Calculated weighted difference and standardized	FG% Difference
3PM	Calculated weighted difference and standardized	3PM Difference
3PA	Calculated weighted difference and standardized	3PA Difference
3P%	Calculated weighted difference and standardized	3P% Difference
FTM	Eliminated during feature selection	
FTA	Eliminated during feature selection	
FT%	Calculated weighted difference and standardized	FT% Difference
OREB	Calculated weighted difference and standardized	OREB Difference
DREB	Calculated weighted difference and standardized	DREB Difference
REB	Calculated weighted difference and standardized	REB Difference
AST	Calculated weighted difference and standardized	AST Difference
STL	Calculated weighted difference and standardized	STL Difference
BLK	Calculated weighted difference and standardized	BLK Difference
TOV	Calculated weighted difference and standardized	TOV Difference
PF	Calculated weighted difference and standardized	PF Difference
+/-	Calculated weighted difference and standardized	+/- Difference
NA	Counted as 1 if Team A is the home team, 0 otherwise	Home Advantage
NA	Calculated weighted mean of all features, got difference and standardized	Stability Difference
NA	Added (Subtracted) 1 if A won (lost) B once in the previous matches	Previous Competition
Upset Factor	Eliminated during feature selection	

3 Experiment Setup

3.1 Correlation matrix

Before we added all the new features, we checked the correlation matrix [1] in which each entry ranges from -1 to 1, indicating the relationship between the two variables (1 indicates a perfect positive correlation, -1 indicates a perfect negative correlation, and 0 indicates no linear correlation between the variables). The correlation matrix is visualized using a heatmap, where the colors indicate the strength of the correlation:

- Red or dark blue typically indicates a high positive or negative correlation, respectively.
- White or light colors indicate weak correlations.

Although most of the variables are not highly correlated, we should note some high correlation ships:

- FGM vs. PTS: Their relationship is significant and can often be substantial. FGM directly contributes to PTS, but the contribution depends on whether the field goal was a two-pointer or a three-pointer

- FG% vs. OREB: reflects how teams that are good at getting offensive rebounds tend to generate higher-quality shots (especially near the basket), extending their possessions and giving them more opportunities for a successful field goal. This leads to a positive relationship between offensive rebounding and field goal efficiency, as teams with more offensive boards can create more scoring chances and typically shoot at a higher percentage.

- FTM vs. FTA: The correlation between FTA and FTM is high because the number of attempts directly influences the number of successful conversions.

- 3PM vs. 3PA: Similar to FTM vs. FTA, they are highly correlated.

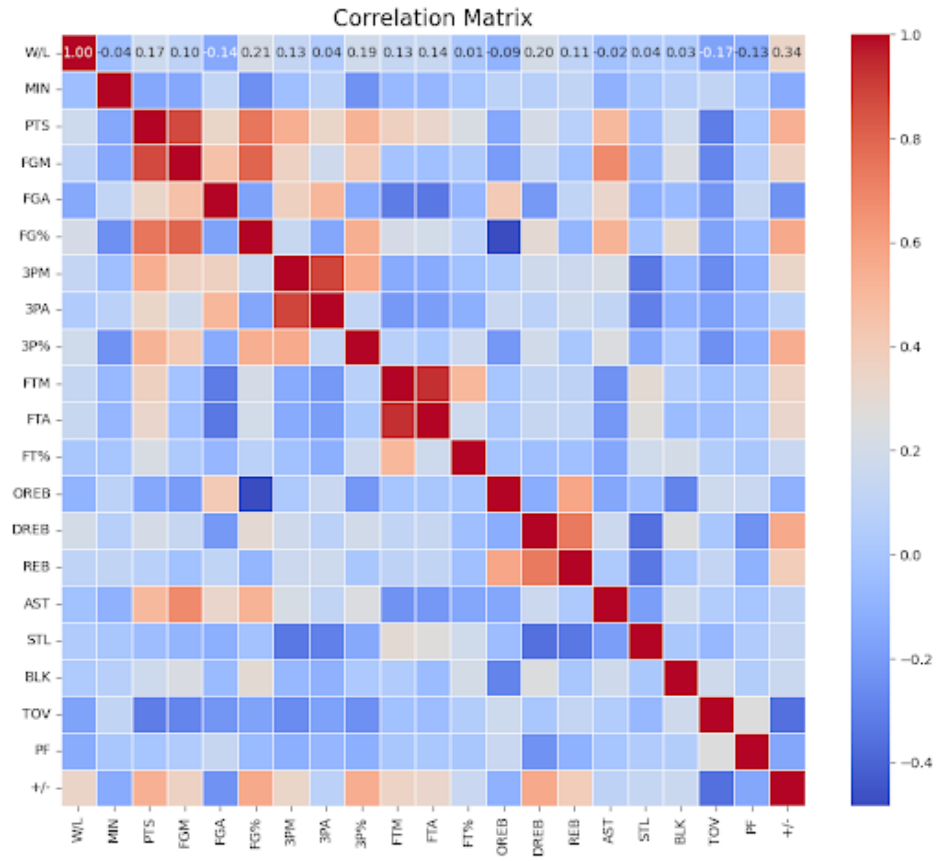

Figure 2: Mean Accuracy and Accuracy Variance Trend

3.2 Feature selection

The feature selection process was conducted using L1-Regularized Logistic Regression (LASSO) [2] to identify the most significant predictors of game outcomes. LASSO is a technique that identifies the most significant predictors by shrinking the less important coefficients to zero. L1 regularization adds a penalty proportional to the absolute values of the model's coefficients, reducing feature complexity. This allows LASSO to select a subset of the most relevant features and ignore those with minimal predictive power. Before applying LASSO, features were standardized using z-score normalization to ensure all features had zero mean and unit variance. This will eliminate bias due to scaling during the regularization process. We then trained the model on standardized features. The following feature indexes were selected since they resulted in non-zero coefficients: `[0, 2, 3, 4, 5, 6, 7, 10, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22]`, which corresponds to the following game metrics: `[W/L, FGM, FGA, FG%, 3PM, 3PA, 3P%, FT%, DREB, REB, AST, STL, BLK, TOV, PF, +/-, Home Advantage, Stability, Previous Competitions]`. These features capture a team's scoring efficiency, rebounding, playmaking, and defensive aspects, strongly influencing game outcomes. The exclusion of other features suggests that their predictive power is minimal in the model. Incorporating feature selection in our model ensures we are focusing on the most impactful variables, improving the interpretability and maintaining predictive accuracy.

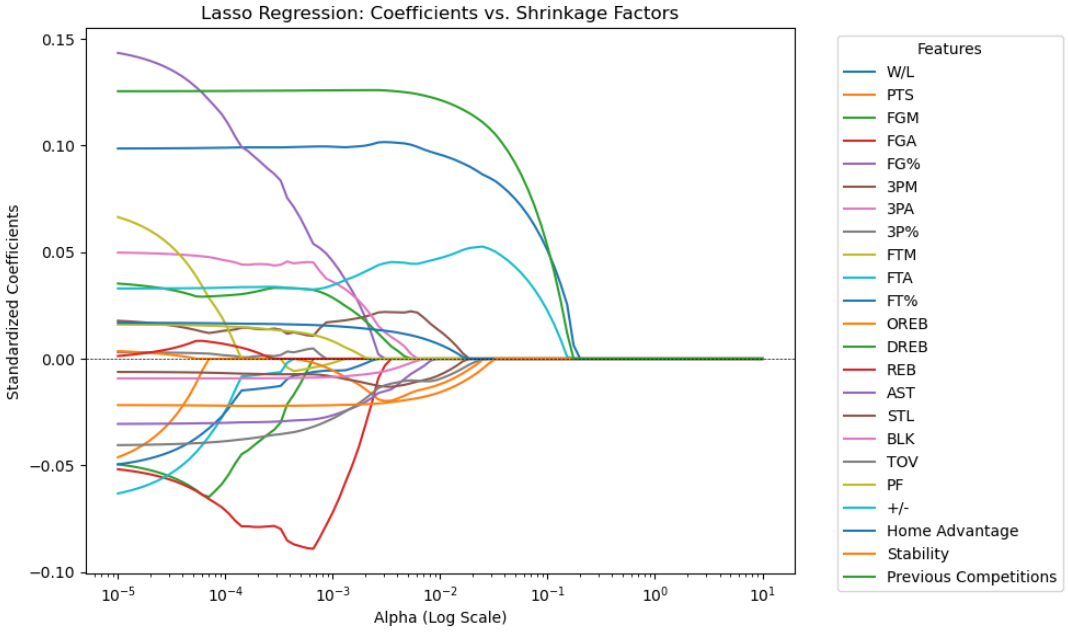

Figure 3: Lasso Regression

As a result of feature selection, some of the provided features were removed from our models. The relationship between `FGM` (Field Goals Made) and `PTS` (Points) is significant and can often be quite substantial. `FGM` directly contributes to `PTS`, but the contribution depends on whether the field goal was a two-pointer or a three-pointer. In feature selection, if a model uses `PTS` as a feature, `FGM` may be eliminated because `PTS` indirectly captures it. However, if the model is interested in distinguishing the type of scoring, `FGM` could still be useful if used alongside other features. Another feature that was eliminated is `3P%`. `3P%` is essentially a ratio or a performance indicator derived from the two features `3PM` and `3PA`. When these two features (attempts and makes) are already included in the model, `3P%` may become redundant because it can be reconstructed from the others. `FTM` and `FTA` can be removed without losing much predictive power when `FT%` is included in the model, as `FT%` encapsulates both the efficiency (success rate) and the underlying attempt numbers. By eliminating `FTA` and `FTM`, the model becomes more straightforward and less prone to overfitting, as there are fewer features to track, especially when those features are highly correlated with `FT%`. The last feature removed is the Upset Factor, an additional engineering feature. Apparently, it did not show much affection toward the model or improve the training accuracy. As shown in the correlation matrix, we could predict that these features were potentially eliminated during feature selection.

3.3 Statistical methods applied

To predict game outcomes, a variety of statistical models were employed, including Random Forest, Logistic Regression, Decision Tree, AdaBoost, XGBoost, Gaussian SVM, and Linear SVM [2]. Each of these models was selected for their unique strengths in handling classification tasks and providing diverse perspectives. Random Forest constructs multiple decision trees and averages their outputs to improve accuracy. Its randomized feature selection and bootstrapping also reduce the likelihood of overfitting. Logistic regression is a linear model that estimates the probability of an outcome using a logistic function. It provides a simple and efficient approach. The Decision Tree model builds a tree by splitting the data iteratively based on feature values and forming decision rules. AdaBoost reweights misclassified observations in each iteration and, in doing so, enhances weak classifiers. This results in a stronger predictive model through sequential learning. XGBoost is an extension of gradient boosting, constructing decision trees sequentially while minimizing a loss function. Support Vector Machine (SVM) [2] models were also utilized. For linearly separable data, linear SVM finds the straight-line separation. In cases where the data is not linearly separable, Kernel SVM applies a transformation in terms of kernel functions to project the data into higher-dimensional space to establish more complex decision boundaries.

3.4 K-fold cross-validation

K-Fold cross-validation [2] is a statistical technique used to evaluate the performance of machine learning models. The dataset is split into K subsets, and each subset is used as a testing set exactly once, while the remaining K-1 folds form the training set. Selecting an appropriate K value is crucial, as a poorly chosen K can lead to misleading evaluations: a high variance score (indicating instability across folds) or a high bias score (overestimating the model's performance). We used the Random Forest model to address this and experimented with different K values from 2 to 15. We compared the mean accuracy and accuracy in variance across these K values to identify the optimal choice.

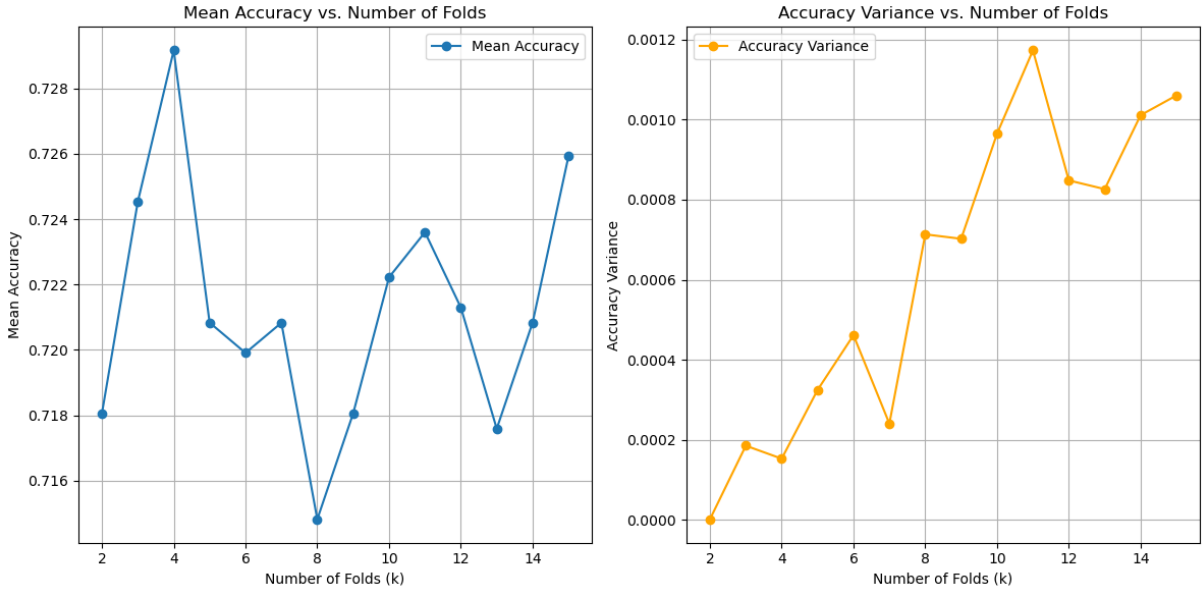

Figure 4: Mean Accuracy and Accuracy Variance Trend

As shown in Figure 3, the larger K is, the more significant the variance of accuracy we will have. As a result, we determined that K = 5 provides the best balance between the mean testing accuracy and the variance of testing accuracy, making it the most suitable choice for this dataset and model.

4 Results and Analysis

In order to assess the impact of our additional engineered features and more sophisticated data representations, we conducted a two-phase evaluation: a baseline scenario and a final scenario. The baseline models were trained using only the original transformed dataset, which consisted of the difference in average statistics between the two competing NBA teams prior to each game. The final models integrated the newly engineered features—exponential weighting of historical statistics, stability metrics, previous competition results, and Lasso regularization for feature selection before putting them into the model.

Table 2: Baseline and Final Model Performance

Model	**Baseline Mean Accuracy**	**Final Mean Accuracy**
Random Forest	0.64	0.72
Logistic Regression	0.68	0.70
Decision Tree	0.58	0.64
AdaBoost	0.64	0.72
XGBoost	0.62	0.71
Kernel SVM	0.64	0.70
Linear SVM	0.67	0.70

Table 2 presents the mean accuracy of each model when trained on the basic transformed dataset. In the initial scenario, the models rely only on the raw differences in averaged performance metrics before that date. As a result, we can see that logistic regression and linear SVM gas have the highest mean accuracy, `0.68` and `0.67`. Random Forest, AdaBoost, and Kernel SVM have moderately good mean accuracy of `0.64`. As we can see, all baseline models generally struggled to exceed 70% accuracy, suggesting that while basic performance metrics and raw historical averages contained some predictive signal, there was substantial room for improvement.

Table 3: Classification Report of AdaBoost for Testing Data

	Precision	Recall	F1 Score	Support
0	0.7102	0.7319	0.7209	1078
1	0.7245	0.7024	0.7133	1082
Accuracy		0.72		
Macro Avg	0.7173	0.7172	0.7170	2160
Weighted Avg	0.7173	0.7171	0.7170	2160

Table 4: Classification Report of Random Forest for Testing Data

	Precision	Recall	F1 Score	Support
0	0.7161	0.7300	0.7230	1078
1	0.7251	0.7116	0.7186	1082
Accuracy		0.72		
Macro Avg	0.7209	0.7209	0.7208	2160
Weighted Avg	0.7209	0.7208	0.7208	2160

The performance profile shifts notably after adding some features and having selection and normalization procedures. Our final model performance shows that these enriched features yield marked improvements for several models. Specifically, Random Forest and AdaBoost both reach `0.72` accuracy, an increase of 8%. This notable jump in accuracy for these ensemble methods suggests they are particularly adept at leveraging the newly engineered features. In conclusion, our models successfully predicted the outcome of each game with more than 70% accuracy during the NBA 2023-2024 season.

5 Further discussion

The main task of this analysis is to focus on engineering features, created based on common assumptions, that would improve our accuracy in predicting each game's result. The first assumption is that the home team tends to play better than the guest team. Using only the Home Advantage variable, we achieved 54% accuracy. According to an NBA article, this season's home advantage was not ideal since the home advantage should be "All-time, NBA teams win home games about 62% of the time" [4]. The second assumption we implemented in this study is that we weighted the results of recent games higher; this meant that if the team was performing well during these recent games, then it likely had a higher chance of winning. Last but not least, we assumed that Team A would lose to B if Team A lost more games than Team B before their game. These assumptions helped us understand our data better.

However, there were some restrictions while we were building our models. Our data includes all games of one season only, so we could not make any predictions on early games of the season. Hence, we skipped the first 300 games. If our data included games from the previous season, we could have predicted those 300 games. Doing so might also increase our accuracy. Another issue was most of our dataset statistics were recorded and measured using different scales. Without performing z-score normalization, our data is highly imbalanced, and accuracy could not get up to 70%. Despite these restrictions, we were able to build our models to increase accuracy more than we expected.

References

[1] *Correlation Matrix*. Accessed: 2024-12-13. 2024. URL: https://www.w3schools.com/datascience/ds_stat_correlation_matrix.asp.

[2] Gareth James, Daniela Witten, Trevor Hastie, Robert Tibshirani, and Jonathan Taylor. *An Introduction to Statistical Learning with Applications in Python*. 2nd. New York: Springer, 2021.

[3] *NBA Team Stats: Traditional - 2023-24 Season*. Accessed: 2024-12-13. 2023. URL: https://www.nba.com/stats/teams/traditional?Season=2023-24.

[4] Tim Reynolds. *NBA Home-Court Win Percentage This Season Will Be Worst Ever*. Accessed: 2024-12-13. 2024. URL: https://www.nba.com/news/nba-home-court-win-percentage-this-season-will-be-worst-ever.

[5] *Z-Score Normalization: Definition and Examples*. Accessed: 2024-12-13. 2024. URL: https://www.geeksforgeeks.org/z-score-normalization-definition-and-examples/.