STATS 101C Final Report
Prediction of NBA Game Outcomes

Contents

1 Introduction

Predicting game outcomes has become an important part of sports analytics, providing valuable insights for teams, coaches, and fans. This project uses historical game data from the 2023-2024 NBA season to predict whether a team will win or lose.

Feature engineering is essential for transforming the raw game statistics into meaningful inputs. Thus, key features like home-court advantage, team performance consistency, previous matchup results between teams and average scores before the game are created to improve the accuracy of predictions. To further refine the model, variable selection techniques like lasso regression and random forest permutation are implemented to select the most significant variables.

A variety of modeling approaches, including logistic regression, QDA, decision tree, ridge and Support Vector Machine will be tested to determine the best approach for predicting the game outcomes. Train-test split will also be applied to ensure the models perform well on unseen data.

By combining feature engineering, variable selection, and a range of prediction models, we hope to achieve better accuracy than the baseline of 65% and provide deeper insights into the factors driving NBA game outcomes.

2 Descriptive Analysis

The data set consists of 2460 matches in total, with 82 games for each of the 30 teams. The summary of each team's winning rate is presented below.

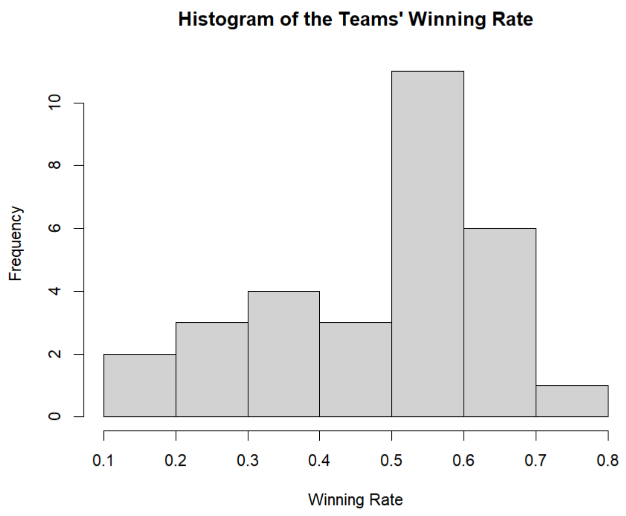

Figure 1: The Distribution of Winning Rate

The distribution of the team's average winning rate is slightly right-skewed, with the most frequent winning rate centered around 0.5-0.6.

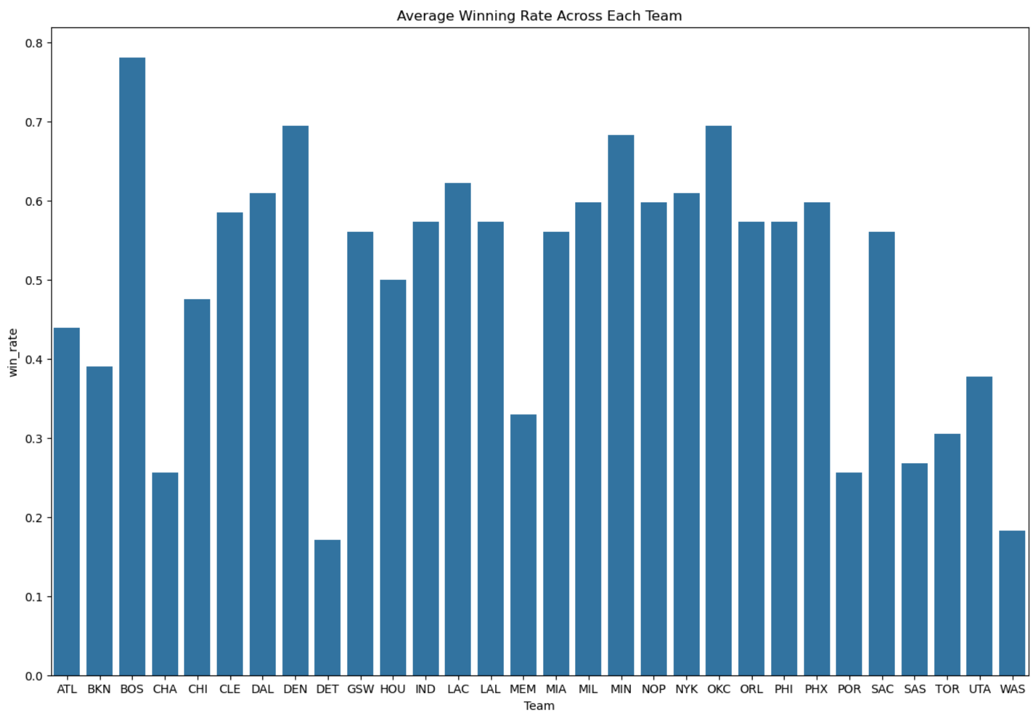

Figure 2: Winning Rate of Each Team

Taking a closer inspection, Team BOS has the highest winning rate of 0.7804878, while DET has the lowest winning rate of 0.1707317. Our task is to predict the win/loss outcome of each game, given the historical data.

3 Feature Engineering

Considering that there exist variables that may affect the outcomes but weren't included in the dataset, we started data processing by adding the following variables.

3.1 Home Advantage

As teams with home advantage are likely to perform better, we took this factor into consideration by creating a dummy variable that equals 1 if the team is the home team in the competition, and equals 0 otherwise. This variable will be denoted as "Home advantage."

3.2 Historical Variance of the Team's PTS

Historical variance of team's PTS captures the variability of team's scoring performance over time. Specifically, it calculates the variance of all previous matches that the team participated in before the current game in the dataset, and this feature is only available for games occurring on and after November 1st. Historical variance of team's points might have potential connection or relation to winning or losing the next game. This variable will be denoted as "variance."

3.3 Previous Competition Record Between the Two Teams

We want to take the total number of wins that each team obtained before the match date into account. To keep track of this, we start by setting the value to zero for all teams. Every time a team plays a game, we update the record: we add 1 to the previous record of the team if the team wins, and doesn't change the value if the team loses. This way, we get a simple and clear view of how well each team has performed over time. It makes comparing the competition between the two teams more straightforward. This variable will be denoted as "previous competition record." Please note that in the data pre-processing, we are only taking the matches that happened before the match date into account. Therefore, the value of this feature depends entirely on the previous data.

3.4 Historical Average Score in Matches Between the Two Teams

Considering that the competition outcome cannot tell us about the difference in points score between the winning team and the losing team, the record of previous competition outcomes doesn't recap all information about the difference in ability between these two teams. Therefore, we construct this feature. For each match between two teams (denoted as A and B), we computed the historical average PTS in every match between team A and team B prior to the current date for team A and team B respectively. For example, before a match between GSW and PHX, if there was only one historical match between GSW and PHX in which GSW scored 110 and PHX scored 100, then the historical average score is 110 for GSW and 100 for PHX. If there were two matches between GSW and PHX, in the first one GSW scored 110 and PHX scored 100 while in the second one GSW scored 100 and PHX scored 110, then the historical average score is 105 for both teams. If the current match is the earliest match between A and B in the dataset, then the historical average score for team A will be approximated by taking the average of all previous matches that A participated in, and similarly, the historical average score for team B will be calculated by taking the average of all previous matches that B participated in. For example, if GSW never competed with PHX before and GSW's average score in other previous matches is 100, then the value will be 100. This variable will be denoted as "average score."

4 Data Pre-processing

In the prediction task, we are only given the historical record of the two teams. Therefore, we constructed a new data set by computing the average difference between the two teams in each variable before each competition and adding the features we constructed in section 3. Then, we made variable selection and predictions based on the new dataset. In this new dataset, all features are based only on the previous data, except for home advantage, which refers to the home advantage in the current match–since we are supposed to know who will be the home team before the match begins. As early matches have limited previous records, we included only the matches on or after November 1st in our model, which gave us a dataset of 2352 observations. Besides, we cleaned the data by removing a row with a missing value, converting the outcome variable into binary values (1 for win, 0 for loss), and eliminating irrelevant categorical columns like identifiers and dates.

5 Variable Selection

In variable selection, we used two methods: lasso classification and random forest variable importance ranking.

5.1 Lasso Classification

We took the competition result as the response variable, and the other numerical variables as the predictors. From the result of lasso, we concluded that MIN (The total time of the game, measured in minutes), FGA (Field Goals Attempted), FG% (Field Goal Percentage), 3PA (Three-Point Attempts), 3P% (Three-Point Percentage), FTA (Free Throws Attempted), FT% (Free Throw Percentage), AST (Assists), STL (Steals), BLK (Blocks), TOV (Turnovers), PF (Personal Fouls), +/- (Plus/Minus), home advantage of the match (home advantage), variance (historical variance of the team's PTS) and average score (historical average score) should be included in the model. With these variables, the testing accuracy of the lasso is 0.66997.

5.2 Random Forest Variable Importance Ranking

We used random forest to rank the importance of each variable, and the result presented in Table 1 shows that the importance of each variable is roughly equal, except for the home advantage. This is aligned with the lasso result, since this algorithm also indicates that home advantage is among the most important variables. Therefore, we decided to use the variables selected in the lasso classification.

Table 1: Ranking of the Variable Importance

Feature	Score	Feature	Score
Previous Record	0.075819	Average Score	0.037870
+/-	0.074976	FT%	0.037617
3P%	0.046854	REB	0.037462
TOV	0.045735	Home Advantage	0.036415
DREB	0.045485	BLK	0.036290
FGA	0.042301	PF	0.036110
Variance	0.041122	STL	0.036019
FG%	0.040602	3PM	0.035731
PTS	0.039400	FGM	0.035421
AST	0.038930	OREB	0.033845
FTA	0.038318	MIN	0.031601
FTM	0.038175		
3PA	0.037903		

6 Model

To predict the outcome of each competition, we used logistic regression, QDA, decision tree, ridge, and Support Vector Machine (SVM).

6.1 Logistic Regression

Logistic Regression is a statistical method used for binary classification problems. Here, a logistic regression model is employed to predict whether games resulted in a "Win" or a "Loss" based on the selected features.

The logistic regression model was trained using Leave-One-Out Cross-Validation (LOOCV) to ensure robust evaluation. The model achieved an accuracy of 65.306%, with a Kappa value of 0.3061, indicating moderate agreement between the predicted and actual outcomes.

To demonstrate the model's performance, a decision boundary was plotted using the features PTS (Points) and previous records with other features taking their average values. The upper region is where the model predicts a "Win" (blue) and the bottom region predicts a "Loss" (red). The decision boundary illustrates how the model separates the two outcomes.

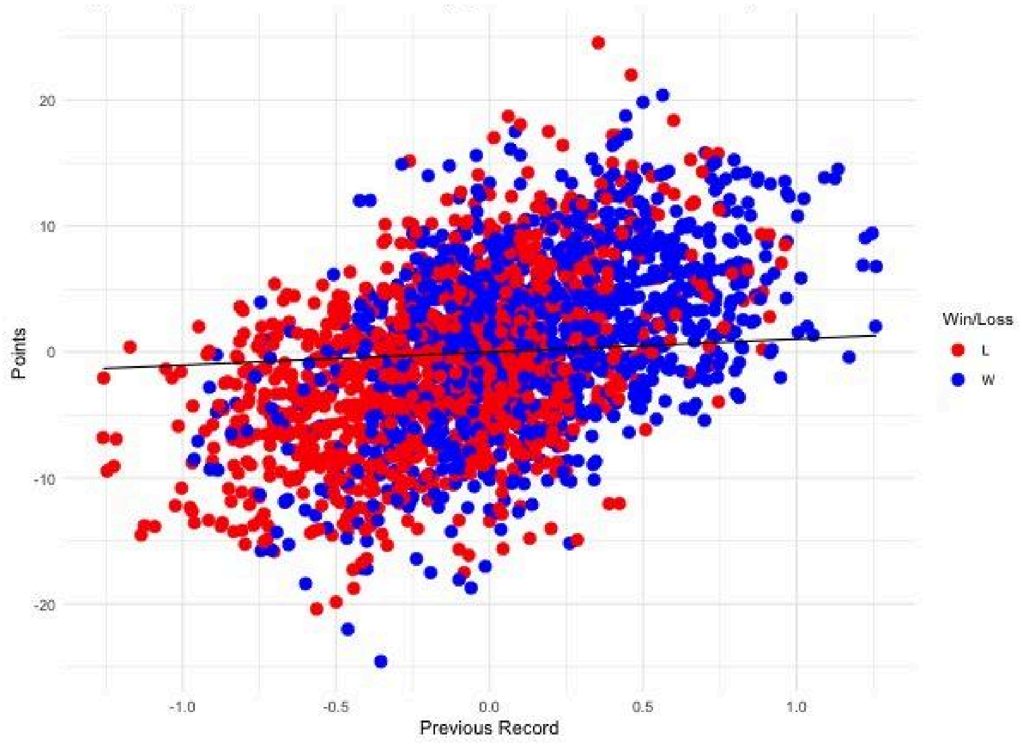

Figure 1: Illustration of Logistic Regression Decision Boundary

6.2 QDA

Quadratic Discriminant Analysis (QDA) is a classification method that assumes each class in the data follows its own Gaussian distribution. Unlike simpler models, QDA allows each class to have unique covariance structures, which helps it handle non-linear relationships more effectively. By creating quadratic decision boundaries, QDA can classify data points based on the probabilities of belonging to different classes.

For this project, a QDA model was built to predict whether basketball games resulted in a "Win" or a "Loss." The data was split into training and testing sets, with 70% used for training the model and 30% for testing its performance. After training, the QDA model was evaluated on the test set, achieving an accuracy of 0.6671. This accuracy was calculated by comparing the model's predictions with the actual game outcomes.

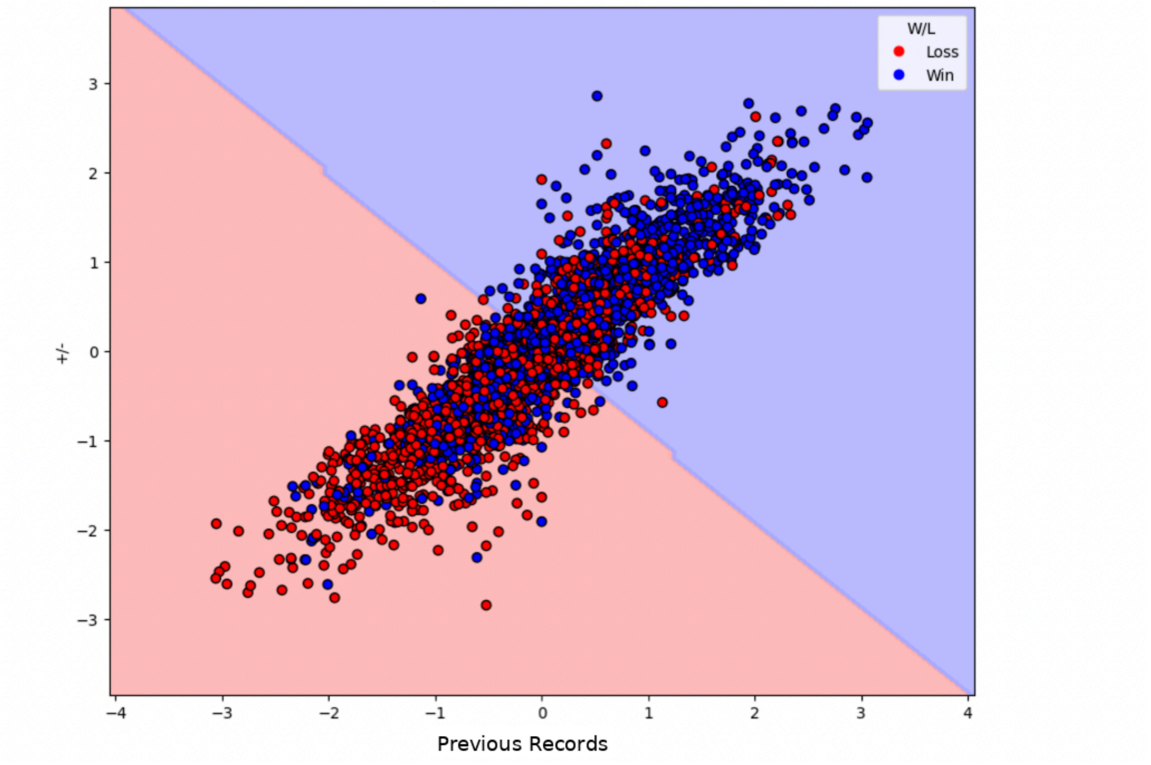

Figure 2: Illustration of QDA Decision Boundary

To better understand how the QDA model works, a decision boundary plot was created using the features +/- and previous records as an example. The plot visually shows the regions where the model predicts a "Win" (blue) or a "Loss" (red). Data points were overlaid on this background, with blue and red points representing actual "Win" and "Loss" outcomes, respectively. The decision boundary highlights how the model separates the two outcomes.

6.3 Decision Tree

A decision tree model with XGBoost was used to predict the outcomes of games, specifically focusing on whether a team would "win" or "lose" based on some selected features ("previous record" and "+/-").

The decision tree classifier was trained and tested using a split of the dataset (70% for training and 30% for testing) to effectively measure the model's performance. The model reached an accuracy of 65.16%, which means it could reasonably distinguish between winning and losing outcomes based on past performance and current metrics. Interestingly, this accuracy was the highest we could achieve; adding any other features actually made the accuracy drop, which might be because of the overfitting caused by increased complexity.

To better understand how the model was making its predictions, we generated a decision tree plot to show the sequence of splits and the importance of each feature. The decision nodes clearly show how crucial "previous record" and "+/-" are for predicting whether a team would win or lose. By simplifying the decision tree to focus only on these key features, the model became more interpretable, making it easier to see the impact of previous games and score differences on game outcomes.

6.4 Ridge

Though the algorithm is usually designed for regression, we can use it in the classification task with Python, which will return the sign of the prediction as our classification result (scikit-learn developers, n.d.). This model aimed to predict game outcomes (win or loss) using Ridge Regression. Categorical features were one-hot encoded, and all predictors were standardized to ensure consistent scaling. The dataset was then split into 70% training and 30% testing, and Ridge regression was applied with cross-validation over a range of alpha values (0.1, 1.0, 10.0, 100.0, 1000.0, 10000, 100000) and we selected the alpha with which the model has the best performance.

Table 2: Table of Coefficients in Ridge Classifier

Feature	Estimated Coefficient
Previous Record	0.063
Average Score	-0.057
AST	- 0.053
FG%	0.057
+/-	0.044
Variance	0.037
STL	0.023
FGA	-0.016
3PA	0.029
PTS	0.026
FT%	-0.014
TOV	-0.013
BLK	0.011
PF	-0.009
MIN	-0.007
OREB	0.004
REB	0.009
3P%	0.009
FTM	-0.012

The model achieved 65.58% accuracy on the test set, a moderate result suggesting some patterns were captured but with room for improvement. The estimated coefficients are presented in Table 2. Key insights from the feature coefficients revealed that previous records and field goal percentage (FG%) had the strongest positive impact on predicting a win. Interestingly, some features, like the average score before a match and assists, had negative coefficients, possibly pointing to unexpected influences. Other predictors, such as offensive rebounds (OREB) and three-point percentage (3P%), showed little to no impact on game outcomes.

While Ridge Regression provided clear insights and helped avoid overfitting, the model's accuracy suggests it may miss deeper patterns or nonlinear relationships in the data. The unexpected behavior of some features highlights the need for better key factors.

6.5 Support Vector Machine

With the cleaned data, we trained the Support Vector Machine, an algorithm that makes classification decisions based on the separating hyperplane. We tested several choices of C (0.1, 0.3, 0.5, 0.7, 1.0) and selected the model with the best performance on the testing dataset. Overall, the model performed moderately with a testing accuracy of 69.55%.

7 Conclusion

Above all, we converted the original data into a dataset containing the historical records before each match and trained 5 models on the new dataset. Since the dataset is balanced, we use testing accuracy as the metric to evaluate the performance of each model, which is presented in Table 3.

Table 3: Summary of Model Accuracy

Model	Testing Accuracy
Logistic Regression	66.67%
QDA	66.71%
Decision Tree	65.16%
Ridge	65.58%
SVM	67.14%

As shown in the table, the model with the best performance on the testing dataset is SVM, with an accuracy of 67.14%, outperforming our baseline while still having room for improvement.

To achieve better model performance, we may enlarge our dataset. Currently, we are training on a dataset with only 2352 observations, which limits our model choice and provides insufficient training for each model. Furthermore, we can consider more features, such as collecting data on the performance of each player.

8 References

scikit-learn developers. (n.d.). 1.1. Linear Models. Scikit-Learn. Retrieved December 7, 2024, from https://scikit-learn/stable/modules/linear_model.html