Statistics 101C Final Project

Predicting Game Outcome from Past NBA Game Data

Elena Flauto (505697812)

Sofia Jain (205533049)

Taylor Prince (105682412)

Jasmine Yung (505699156)

Contents

1 Introduction

This paper examines the NBA 2023-2024 dataset and proposes a method by which this past data can be used to predict the outcome of future NBA matchups.

Each entry of the dataset represents the home team, opponent, date, win or loss indicator (W/L), minutes played (MIN), points scored (PTS), field goal percentage (FG%) in relation to field goals made (FGM) and attempted (FGA), three-point field goal percentage (3P%) in relation to three-point field goals made (3PM) and attempted (3PA), free throw percentage (FT%) in relation to free throws made (FTM) and attempted (FTA), total rebounds (REB) in relation to offensive (OREB) and defensive rebounds (DREB), assists (AST), steals (STL), blocks (BLK), turnovers (TOV), personal fouls (PF), and plus/minus statistic (+/-) for a game in the 2023-24 season. Additionally, we took into consideration a team's home advantage, stability, previous matchups against the current opponent, and weighted statistics based on the recency of the matchup.

Using preprocessed data, we trained logistic regression, QDA, random forest, and KNN models to predict whether a match-up would result in a win or loss. Our trained logistic regression model performed the best, with a test accuracy of approximately 88%.

2 Data Preprocessing

We cleaned the NBA 2023-2024 dataset by replacing non-numeric entries for FT%, removing missing cases, and converting the target variable W/L to a numeric format such that 0 indicates loss and 1 indicates win. Then, we conducted feature engineering by constructing four more informative features to optimize the prediction performance of our models. We applied LASSO to figure out which predictors should not be included and removed certain numeric variables based on the results.

2.1 Data Cleaning

Firstly, since some entries in the column FT% were non-numeric (i.e., "-"), we replaced these entries with NaN values and then dropped all rows in the dataset containing missing values (e.g., NA, NaNs). To convert the binary target variable W/L from categorical to numeric, we encoded the win or loss indicator by setting a win as the value 1 and a loss as the value 0.

2.2 Feature Engineering

2.2.1 Creating the Home Advantage Variable

Through string splitting, we extracted the home team and the visiting team for every matchup. To account for the expectation that a team performs better as the home team compared to the visiting team, we added a column named "Home Advantage" to the dataset with the values 1 if the team is the home team and 0 if the team is the visiting team. We then transformed the column named "Match Up" to solely specify the abbreviation of the team's opponent in each matchup.

2.2.2 Creating the Stability Variable

This variable accounts for the home team's shooting consistency. The stability variable returns the value of the standard deviation of the home team's field goal percentage (FG%) in their last four games, regardless of their opponent. We extracted the home team from the current game, identified the past four games they played, and calculated the standard deviation of their field goal percentage for these games. The larger this standard deviation, the more unstable a team is in regards to their scoring consistency.

2.2.3 Creating the Weights Variable

This variable assigns weights evenly, from 10 to 100 percent, to games according to their date. The oldest dates are weighted starting at 10 percent, and this increases up to 100 percent as we approach the most recent date. This accounts for the recency of past matchups when predicting future matchups. Games played on the same day have been assigned the same weights. We found adding this variable useful, as it becomes relevant when creating classifiers for the random forest model, for example, which contains the "sample weights" parameter.

2.2.4 Creating the Previous Match-up Variable

To account for past games of the same match-up, we introduced the previous match-up variable, which shows the average point differentials from the previous games between the two teams. If there have not been any games of the same match-up prior to the game in question, this value will be assigned as 0.

2.2.5 Removing the MIN, PTS, FGM, 3PA, FTA, FT%, OREB, DREB, and AST Columns

After running a LASSO regression analysis, we found that the PTS, FGM, 3PA, FTA, FT%, OREB, DREB, and AST columns had LASSO coefficients corresponding to exactly or relatively close to 0; therefore, we removed these predictors. We also removed the MIN variable as it was identical for most of the data entries. REB, STL and TOV had 0 or close to 0 coefficients, but we found our models had better accuracy when we kept these in. Additionally, these statistics are unique and capture important characteristics that aren't featured in the other included statistics.

Since there were other statistics similar to the ones we removed (such as FG% for PTS), we felt comfortable using the LASSO results and removing the specified variables. Therefore, in a match-up, the minutes played, total points scored, field goals made, three-point field goals attempted, free throw percentage, offensive rebounds, defensive rebounds, and number of assists were not predictors of the game outcome for our models.

Table 1: Summary of feature engineering steps

Original Column	Preprocessing Action	Final Column
Team	Transformed to home team	Home Team
Match Up	Transformed to only include visiting team	Opponent
W/L	Converted to binary (1 = win, 0 = loss)	W/L
MIN	Eliminated during feature selection	--
PTS	Eliminated during feature selection	--
FGA	Kept unchanged	FGA
FGM	Eliminated during feature selection	--
FG%	Kept unchanged	FG%
3PA	Eliminated during feature selection	--
3PM	Kept unchanged	3PM
3P%	Kept unchanged	3P%
FTA	Eliminated during feature selection	--
FTM	Kept unchanged	FTM
FT%	Eliminated during feature selection	--
OREB	Eliminated during feature selection	--
DREB	Eliminated during feature selection	--
REB	Kept unchanged	REB
AST	Eliminated during feature selection	--
STL	Kept unchanged	STL
BLK	Kept unchanged	BLK
TOV	Kept unchanged	TOV
PF	Kept unchanged	PF
+/-	Kept unchanged	+/-
Home Team	Used to calculate home advantage	Home Advantage
FG%	Used to calculate SD of FG% for home team past games	Stability
Game Date	Used to weight games dependent on recency	Weights
+/-	Used to calculate point differential in previous games	Previous Match Up

The final columns shown above are the columns we used in our analysis.

3 Experimental Setup

After splitting our data into 70% training data and 30% testing data, we compared Logistic Regression, QDA, Random Forest, and KNN models. The processes for fitting each of these models are provided below.

1. **Logistic Regression.** In logistic regression, we can implement predictions for new samples based on estimates of β_0 and $\boldsymbol{\beta}$. Assuming $log(\frac{P(Y=1|X)}{P(Y=0|X)}) = \beta_0 + \boldsymbol{\beta}^T x$, we can assign a prediction value of 1 to values of $\hat{\beta}_0 + \hat{\boldsymbol{\beta}}^T x$ greater than 0 and a prediction value of 0 to values of $\hat{\beta}_0 + \hat{\boldsymbol{\beta}}^T x$ less than 0. We applied best subset selection with cross validation (CV) error as the criterion to determine the combination of predictors utilized for fitting the best-performing model. 13 predictors produced the best accuracy in our logistic regression model.

k	CV Accuracy Score
1	0.7083
2	0.7722
3	0.7873
4	0.7995
5	0.8350
6	0.8402
7	0.8413
8	0.8489
9	0.8542
10	0.8576
11	0.8582
12	0.8588
13	0.8617
14	0.8553

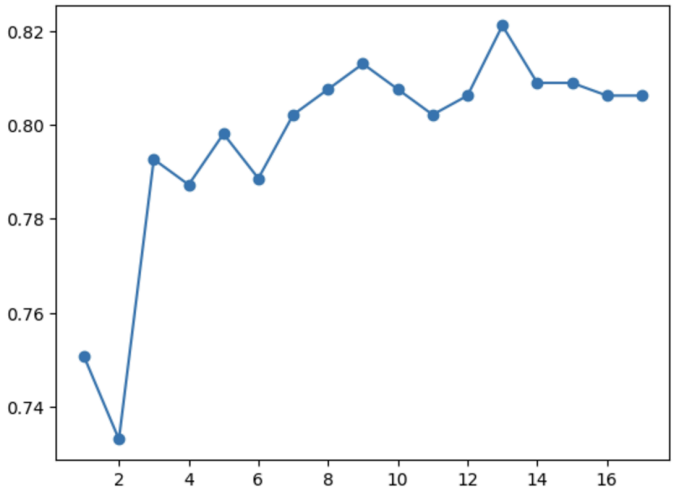
2. **QDA.** QDA, or quadratic discriminant analysis, uses the conditional probability $\hat{P}(Y = 1 \mid \mathbf{X})$ for classification, assigning probability greater than ½ to the value 1 and those less than one half to the value 0. Deriving the decision boundaries as $\hat{P}(Y = 1 \mid \mathbf{X}) = ½$ yields quadratic functions which can be ellipsoids or hyperbolas. We find this decision boundary to separate different classes. This model involves the assumption that the distribution of $P(Y = 1 \mid \mathbf{X})$ is multivariate normal with different mean vectors and covariance matrices.

3. **Random Forest.** Random forest involves drawing bootstrap samples and fitting classification trees for each by randomly selecting a subset of the p variables, picking the best split among the chosen subset, and splitting the node. The random forest estimate then classifies based on majority vote. This method allows specification of various hyperparameters, including sample

weights. Through averaging de-correlated trees, the random forest model is less susceptible to overfitting.

4. **KNN.** KNN, or K-nearest neighbor, is a non-parametric approach that does not involve assumptions about the shape of the decision boundary. The classification involves specifying an odd integer value for K to avoid ties, finding K points in the training dataset that are closest to $\mathbf{x}_0$ for $\mathbf{X} = \mathbf{x}_0$, and denoting this subset of points as N_0. Classify the value of 1 for values of $\hat{P}(Y = 1 | \mathbf{X} = \mathbf{x}_0)$ greater than ½ and the value of -1 to values of $\hat{P}(Y = 1 | \mathbf{X} = \mathbf{x}_0)$ less than ½ for $\hat{P}(Y = 1 | \mathbf{X} = \mathbf{x}_0) = \frac{1}{K}\sum_{i \in N0} I(y_i = 1)$. To find the optimal k, we plotted accuracies from KNN models with different k values and chose the best one. K = 13 produced the best accuracy in our KNN model so we chose k = 13 for our final implementation of the KNN model.

Figure 1: KNN Testing Accuracy for Different K



4 Results and Analysis

The results below exhibit the training and testing accuracy for each of the four models we trained. The testing data made up 30% of the dataset after preprocessing.

Table 2: Training and testing performance across classical models

Model	Training Accuracy	Testing Accuracy
Logistic Regression	0.8582	0.8808
QDA	0.8669	0.8659
KNN	0.8211	0.8379
Random Forest	0.9064	0.8347

With a testing accuracy of approximately 0.8808, our best-performing model was the Logistic Regression model. Additionally, there is not a large gap between training and testing accuracy, meaning that the data does not indicate overfitting. The classification report for the Logistic Regression model evaluated on the testing data is given below.

Table 3: Classification report of logistic regression for testing data

	Precision	Recall	F1 Score	Support
0	0.88	0.87	0.88	366
1	0.88	0.89	0.88	372
Accuracy			0.88	738
Macro Avg	0.88	0.88	0.88	738
Weighted Avg	0.88	0.88	0.88	738

This model produces a precision of 0.88 for both classes, and the model's recall only slightly differs among classes, at 0.87 for class 0 and 0.89 for class 1. Additionally, the F1 scores for both classes indicate the model contains a good balance between precision and recall, as is expected by our previous results. The macro and weighted averages are the same in this instance because the dataset is close to completely balanced, with a support of 366 for class 0 and 372 for class 1. This high precision and recall signal low levels of false positives and false negatives. The model predicts wins or losses with similar accuracy and is consistent. Overall, these results show that the model performs well and does so essentially equally across both classes.

We, additionally, performed 10-fold cross validation on each of our models to test for overfitting and the overall estimated performance of each model on unseen data across multiple folds.

Table 4: Average 10-fold cross validation scores

Model	Average CV Score
Logistic Regression	0.8658
QDA	0.8585
Random Forest	0.8211
KNN	0.8077

These cross validation scores indicate that the Logistic Regression is expected to perform the best on unseen data, with an average cross-validation score of approximately 0.8658 across all folds. This corroborates the results we found on the model's performance above, with the Logistic Regression model overall producing the most accurate predictions and doing so consistently on the testing data set. Each model's CV scores were close to their respective test accuracies, indicating low levels of overfitting.

5 Conclusion and Limitations

Overall, we are satisfied with our results in predicting NBA match-ups using the Logistic Regression model we trained. We found that the addition of measures for home advantage, stability, previous match up, and weighted statistics based on the recency of the match-up greatly improved the model accuracy.

While we added four additional features to the data, there may still be relevant variables that could have been added during the feature selection process to improve the model accuracy even further. Additionally, there are likely external factors that are not easily measurable that impact a team's performance and likelihood of winning a match-up on a game-to-game basis and cannot be easily accounted for in this model. For example, unexpected changes to a team's roster or injuries to main players have potential to impact the accuracy of the model or outcome of a game. If we had access to more extensive data beyond the 2023-2024 NBA season, we would explore past team roster changes in addition to any other changes that occurred over the years to improve our model's prediction accuracy and generalizability beyond the 2023-24 season.

Following our feature engineering procedure in which we used LASSO regression to remove variables with correlations of 0 or near 0, we found that the remaining variables were not correlated. However, running an analysis of correlation among predictors as well as between predictors and the target variable could be implemented in the future to further improve our model.

Lastly, logistic regression assumes a linear relationship between the predictors and the outcome. This linear relationship may not account for the true nature of NBA games. Random forest and QDA models performed well in addition to Logistic Regression, and it is possible that these non-linear models could provide even better prediction results if combined with even more extensive feature engineering.

References

[1] NBA Dataset, 2023-2024