# Predicting 2023-2024 NBA Game Results with Statistical Modeling

STATS 101C, Section 1A

Thomas Peeler (505918210), Reeshad Mohammed (505918615), Jessica Nguyen (906056500), Olivia Nguyen (006165344), Francis Chan (306281422), Stephanie Lei (906030355)

2024-12-13

# Contents

# 1 Introduction

Can use a table for summarizing thos efeatures.

In this project, we aim to predict the results of NBA games throughout 2023 and 2024 via the results of past games. In prediction, for a given game, all that is known is the two teams that are playing and their home and away status; our goal is to use past game statistics for both teams to predict the outcome of the given game before it has happened. From each past game, we can gather a number of statistics for each team that may be useful in prediction, including: home/away status, win/loss, minutes played, points earned, field goals made/attempted/percentage accuracy, three-pointers made/attempted/percentage accuracy, free throws made/attempted/percentage accuracy, offensive/defensive/total rebounds, assists, steals, blocks, turnovers, personal fouls, and the overall plus/minus statistic for the team during the game.

The biggest challenge for this project was in the process of getting the data in a form suitable for prediction. To this end, the features for prediction of a given game outcome were constructed via a weighted average of past game statistics for each team playing in the game. Additionally, feature selection via $L_1$ regularized Logistic Regression was performed in order to select suitable transformations for our features and greatly reduce the size of our feature space, reducing variability in our model. Finally, we chose to try a number of different models for prediction after preprocessing, including Logistic Regression, Linear/Radial SVM, Linear/Quadratic Discriminant Analysis, K-Nearest-Neighbors, and Random Forest . We found that Linear SVM could most accurately predict the outcome of a game with a mean validation accuracy of 0.687.

# 2 Data Preprocessing

Even prior to creating the features for the model, there is a considerable amount of preprocessing to be done on the data as it is given. In its initial structure, the data contains information on 1230 games throughout 2023 and 2024, where each game has two entries in the data; each entry corresponds to the game statistics for *one* team in a given game. However, with this, each entry can successfully be matched up to the other entry corresponding to the same game, about which we will go into detail later.

## 2.1 Data Cleaning

Contrary to what one would expect for real-world data, there was a very small amount of cleaning to be done. In an away game for the Boston Celtics versus the Milwaukee Bucks on April 9, 2024, the Celtics did not perform any free throws. Thus, their free throw percentage for that game would be computed as $\frac{0}{0}$, so that statistic in the dataset was replaced with a null value. In order to retain this sample, we elected to fill in this value with the mean free throw accuracy for the Celtics across the entire dataset.

## 2.2 Data Reshaping

As mentioned, the data was initially in a form where each entry corresponds to only half of the available statistics for the game. So, part of our preprocessing is to create a representation of the data where each entry corresponds to all of the statistics for the game, such that the number of entries is equal to the number of games. In this representation, there is no need for a variable indicating the home team; instead, each

entry is divided into the name and statistics for the home team and the guest team, and the W/L for each entry indicates whether the home team won. Now, for modeling purposes, our model is really just predicting the W/L of the game for the home team, as the result for the guest team is simply its opposite.

The matching of entries to create this representation of the data was done by matching the date and teams for each entry; we found that each entry only had one match in this regard, so there was no chance of accidentally mismatching entries. With this, we find that the home team wins 53.4% of the games in the dataset, which aligns with the home-team advantage that we would expect.

# 3 Feature Engineering

## 3.1 Weighted Mean of Past Statistics

In order to use past game statistics for prediction, we need a way to collate the statistics for a variable number of past games into a fixed number of features. To achieve this, we take a weighted average of game statistics for all of the past games for each team in the game that we are trying to predict. We chose to weight this average by:

- Time between games: Games that happened long before the current game that we are trying to predict likely do not reflect the ability of the team as much as games that happened more recently.
- Home/away status: For predicting the outcome where a given team is playing at home, past games where that team was also playing at home will likely better reflect the ability of the team in the game that we are trying to predict (and the same goes for away games).
- The opposing team: If we want to predict the outcome of team A versus team B, past games where team A played against team B will likely be strong predictors for how the current match will go.

Now, the weighting scheme: Suppose we have a given game for which we want to predict the outcome; we wish to construct a weighted average of past statistics for both teams. Let $t_1$ denote the name of the team that we first wish to construct features for, and let $o_1$ denote the opposing team. Now, let $d_1$ denote the number of days between October 1, 2023 and the day in which the game took place, and let $h_1$ denote the home/away status for $t_1$. Now, consider some past game statistics for which we must assign a weight. For this set of statistics, let $t_2$ denote the team that the statistics belong to, and let $o_2$ denote the team that $t_2$ was playing against to get those statistics. Let $d_2$ denote the number of days between October 1, 2023 and the day in which this other game was played, and let $h_2$ denote the home/away status of $t_2$ for this game.

With this, the weight of this given game's statistics are as follows:

$$w(t_1, d_1, h_1, o_1, t_2, d_2, h_2, o_2, \alpha, \beta, \gamma) = I(t_1 = t_2) I(d_2 < d_1) \frac{(1 + (\beta - 1) I(h_1 = h_2))(1 + (\gamma - 1) I(o_1 = o_2))}{2^{\frac{(d2 - d1)}{\alpha}}}$$

where $\alpha, \beta, \gamma$ are tunable parameters. After the weights for $t_1$ are computed across all (previous) games in the dataset, they are then divided by their sum to ensure that they sum to 1, and then used to compute the

looks like a time-varying weighting scheme that incorporating previous match up into consideration.

should beta and gamma larger than or equal to 1? If yes, why don't you just use beta and gamma?

weighted mean of the past game statistics for $t_1$. This process is repeated for the other team in the game that we wish to predict the outcome of. Once we have these averaged statistics for both teams, we now take the difference in the averages for each statistic to get our final set of features. This works well to greatly reduce our number of features while retaining all of the information that will be relevant to prediction; since we are looking to predict the outcome of a given game, we are most concerned with whether team A does better or worse in a certain aspect than team B, which we estimate as the difference in averaged game statistics.

It is worth noting here that, for any prediction going forward, we will only be creating features for and predicting the outcome of games that occur on or after December 1, 2023; this means that about 22% of the games in the dataset will never be tested for prediction. This is because there is simply not much information before these games to use for creating features. By excluding these early games, we remove much variability from the results of our model and get a better idea of its true performance in the case that we were to use it for predicting game outcomes in the future.

## 3.2 Creation of New Features

### 3.2.1 Block Percentage and Assist Percentage

There are some simple features we can make from the field goals attempted, field goals made, assists, and blocks variables. In a basketball game, a block can only occur as a means to stop a field goal; so, for a given game, if we take a given team's number of blocks and divide it by the number of attempted field goals from their opponent, we get the percentage of attempted field goals by the opponent that were blocked by the team. Further, an assist can only occur as the result of a successful field goal; so, if we take a given team's number of assists in a given game and divide it by their number of successful field goals, we get the percentage of field goals by the team in that game that came as the result of an assist. These additional variables provide useful relative measures for these statistics, and may be useful for prediction.

### 3.2.2 Team Instability

It may be useful to use past games to extract other information about a team aside from their mean performance statistics. For instance, we can get a measure of a team's inconsistency in performance by looking at the variance of some of their past statistics. A team with greater variance in performance statistics may be more inconsistent during play, leading to fewer wins than a more organized team that can play with more consistency. So, we will be creating a new variable for prediction that takes the mean of the variances of some specific statistics over all of the past games for a given team. However, it is likely that not all variables will really measure "consistency" in a way that is useful for predicting a game's outcome. So, we will be doing a backward stepwise selection with one-way ANOVA in order to select the statistics that create an instability measure that is most associated with the response. Considering that our class sizes are equal (there are an equal number of instability measures associated with wins and losses) and our very large sample size of 1914 (only considering games on or after December 1, 2023), we find that the distribution assumptions for ANOVA can be relaxed, and the conclusions from the analysis will be sound.

After doing the selection, we find that the best statistics for creating an instability measure are field goal percentage, number of free throws made, defensive rebounds, assists, steals, and block percentage; the p-value

The mean of variances could be less informative. Because some features like 3-point percentage have low variance in nature. Therefore the mean may decrease it effects.

for the ANOVA is about $10^{-11}$.

#### 3.2.3 Relative Measures of Game Statistics

As far as other transformations for the model, we need not use the raw game statistics from each game to construct our weighted means. Instead, for a given game, some function of the statistics between the teams may give us a more relative measure of the team's performance in that game, and the weighted mean of those relative statistics can be used for our features instead; for example, instead of looking at the raw number of points scored, we may consider the difference of points scored between the two teams, and use that value for our weighted mean. This kind of measure may be more informative than the raw statistics; team A scoring 50 points in a game may seem like poor performance, but with the context that team B, their opponents, scored 10 points, the 40 point lead actually indicates quite good performance (these example values are unrealistic but they are used to illustrate the idea).

For a given game between team A and team B, let $s_A, s_B$ be a given statistic for each team. If we aim to create features for team A, our relative statistic measures that we will test are:

- Identity: $s_A$
- Raw difference: $s_A - s_B$
- Square difference: $(\text{sign}(s_A - s_B))(s_A - s_B)^2$
- Square root difference: $(\text{sign}(s_A - s_B))(\sqrt{|s_A - s_B|})$

These functions give us a good range of relative statistics with different kinds of accentuated differences; something like the square difference will greatly emphasize large differences in statistics while diminishing smaller ones, while the square root difference will do the opposite. So, if there is a kind of measure of statistics that is best for estimating future performance of a team, we hope to capture it in these functions.

### 3.3 Standardization

It should be mentioned here that each of the game statistics will be standardized to have mean 0 and standard deviation 1 prior to taking any means, including those relative statistics mentioned above. This will ensure that we do not run into issues that arise from variables having varying ranges.

### 3.4 Feature Selection

As it stands, there are many possible features we can use for prediction. If we consider all of the relative measures of statistics, we get 108 possible features for prediction; though, many of these are clearly correlated or extraneous, only adding variability to our model. To start, we will be removing the "minutes played" variable from our usable features. 95.2% of the games in the dataset have the same value of 240 minutes played, so the increase in variance from the added variable is likely not worth any small amount of predictive power it may have.

Further, the plus/minus statistic in this dataset is simply the difference in points achieved between the two teams; as we already have a "points" variable, we will be dropping the plus/minus statistic (and all of its relative measures), as well. However, for the other variables, we will need alternative methods to reduce our number of features.

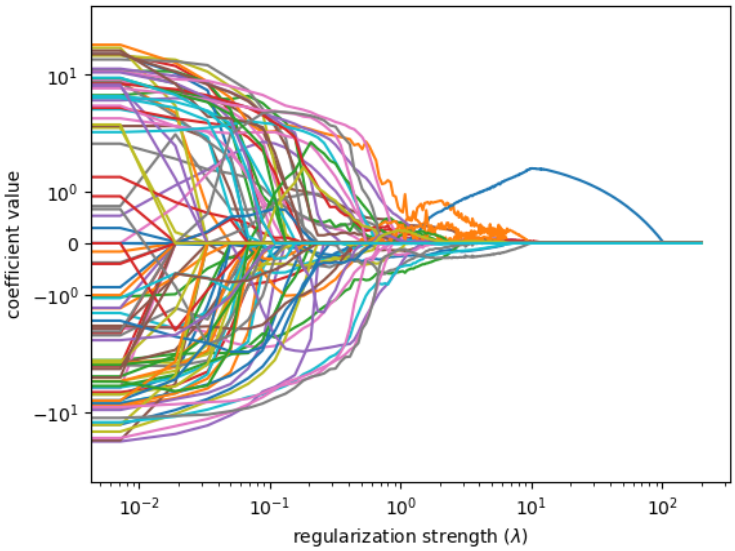
### 3.4.1 $L_1$ Regularized Logistic Regression

In order to get a measure of how informative a given variable is for prediction, we will be using Logistic Regression with $L_1$ regularization for feature selection. If we have many variables that we wish to evaluate, we can put them in a Logistic Regression model and slowly increase the $\lambda$ parameter for the $L_1$ regularization, which should eventually send each variable parameter to 0. We will use this with 5-fold cross-validation to increase $\lambda$ until we max out our validation accuracy, at which point we will drop any variables that are a zero-slope in this "best" model.

(Implementation Note: this portion was performed using the Logistic Regression implementation in Scikit-Learn. In this case, we use `penalty="l1"` for `sklearn.linear_model.LogisticRegression`, where the `C` parameter is the inverse of the regularization strength, that is, $\frac{1}{\lambda}$. Further, for any modeling from here on out, `class_weights="balanced"` was passed to any model that supports it.)

We will use this regularized Logistic Regression approach on all of our possible features, including all of the relative game statistics, in order to pick out those that will be best for prediction. In order to keep the approach as simple as possible at this step, we will be forgoing any weighting for creating features; for the moment, in order to create features for the prediction of a given game, we will simply take the mean of the game statistics, relative and otherwise, for all of the prior games for each team, and then take the difference of those means between the teams.

Here is a plot of the coefficients of all of our possible features fit on a regularized Logistic Regression as $\lambda$ increases:




Of course, this doesn't tell us much, but it gives us a vague idea of how the coefficients will behave when we

start increasing $\lambda$ for our cross validation.

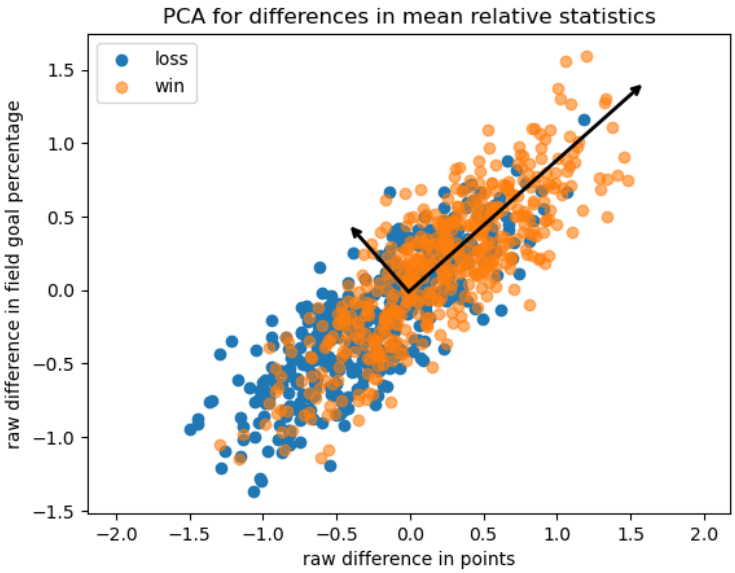
After running this feature selection process with multiple different random states for the cross validation, we find a subset of 10 features with coefficients that are typically nonzero when the validation accuracy is maximized. However, there is still more selection to be done.

### 3.4.2 Further Feature Selection

With these features, we perform forward selection with unregularized Logistic Regression, using the 5-fold cross-validation accuracy as our scoring metric. From this, we get three variables that, together, give us the best performance:

- Raw difference in points earned
- Raw difference in field goal percentage
- Three-point shots achieved (raw value)

where the maximum validation accuracy is about 67.1%. However, we actually find quite a bit of correlation between the features constructed from the raw difference in points earned and the raw differences in field goal percentage, with a correlation coefficient of 0.852. Using PCA, we find that the explained variance ratio from just one component from those two variables is 0.926, indicating that almost all of their variance can be retained with a single linear combination of them. This can also be seen visually:




where the vectors represent the two principal components that can be constructed from the two variables. This new feature and the feature constructed from the three-point shots statistics only have a correlation coefficient of 0.234, so there is little worry of any remaining colinearity. Further, we find that our unregularized Logistic Regression validation accuracy slightly increases when using this principal component for prediction

as opposed to the two variables separately; it would seem that the decrease in variance from decreasing our number of features outweighs any additional information provided by the features separately. So, we will be using the principal component in place of its two constituents for prediction purposes, meaning that our modeling will be done with just two variables.

# 4 Modeling

## 4.1 Hyperparameter Tuning

Now, we come to the issue of hyperparameters. Along with those for specific models, our weighting function for creating weighted means of past statistics has three of its own hyperparameters:

- $\alpha$, which controls how quickly weights should diminish as the time between games becomes larger.
- $\beta$, which is the multiplier for team statistics which have the same home/away status as the game that we aim to predict.
- $\gamma$, which is the multiplier for team statistics that were earned against the same opponent as in the game we aim to predict.

To find optimal values for these, we use a grid search with 5-fold cross validation using Logistic Regression, aiming to maximize our validation accuracy. From our search, we find our optimal parameters as:

- $\alpha \to \infty$ (i.e. forgoing any weighting based on time)
- $\beta = 1.85$
- $\gamma = 1$ (i.e. forgoing any weighting based on opponent)

where our validation accuracy is maximised at about 68.3%. This is a somewhat surprising result; it would seem that the increase in feature variance from lending greater weights to games based on time or opponent outweighs any benefit that it may provide to estimating the ability of a team. However, we do find that weighting by home/away status considerably improves accuracy, so it would seem that a team's performance in past home/away games is a strong predictor for their performance in future home/away games, respectively.

The models that we will be using for prediction have their own hyperparameters, such as minimum impurity decrease for a Random Forest or the $C$ parameter of Logistic Regression. These will be tuned with a similar grid search as was used previously.

# 5 Results

Below are the 5-fold cross validation accuracies for all of the tested models, as well as any hyperparameters that were found to maximise this accuracy:

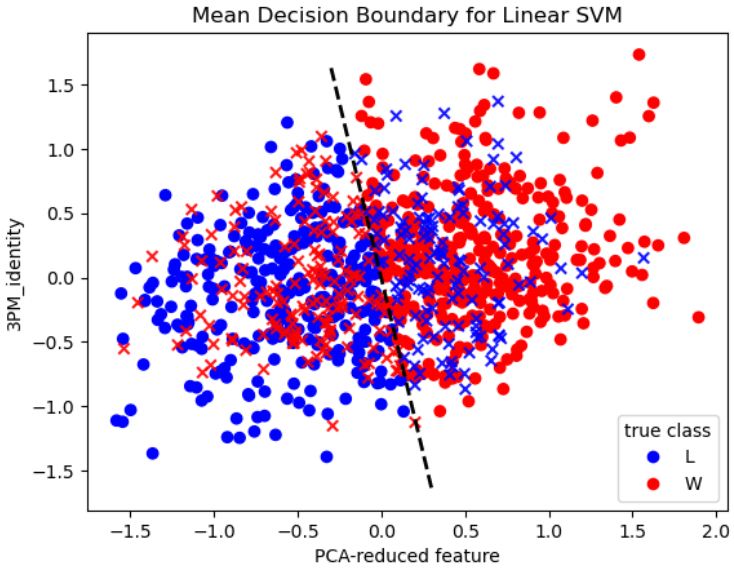
| Model | Hyperparameters | Training Accuracy | Validation Accuracy |
|---|---|---|---|
| Linear SVM | $C \to \infty$ (no regularization) | 0.688 | 0.687 |
| Logistic Regression | $C \to \infty$ (no regularization) | 0.685 | 0.683 |
| Radial SVM | $C = \frac{1}{50}$ | 0.680 | 0.677 |
| K-Nearest-Neighbors | $K = 241$ | 0.680 | 0.676 |
| QDA | $\lambda = 0$ (no regularization) | 0.681 | 0.676 |
| LDA | $\lambda = 0$ (no regularization) | 0.676 | 0.675 |
| Random Forest | Minimum Impurity Decrease $= \frac{1}{30}$ | 0.679 | 0.673 |

And the Linear SVM cross-validation classification report:

| | Precision | Recall | F1 Score |
|---|---|---|---|
| L | 0.65 | 0.69 | 0.67 |
| W | 0.72 | 0.68 | 0.70 |
| Accuracy | | | 0.69 |
| Macro Average | 0.69 | 0.69 | 0.69 |
| Weighted Average | 0.69 | 0.69 | 0.69 |

So, we find that we get the best accuracy with Logistic Regression and Linear SVM; as these two methods are very similar, this is unsurprising. The poor performance of the Random Forest is unexpected, though this is likely due in part to how much we have reduced the data; a Random Forest generally fares better with many features rather than fewer, as its fitting process makes it resilient to the variance that extraneous features would normally cause.

Since we are only modeling with two variables, we can actually make a visualization for our Linear SVM; here, we show the mean decision boundary as determined by many 5-fold cross-validation iterations (Xs represent points misclassified by the mean decision boundary):



# 6 Limitations & Conclusion

As stated, the most significant part of this project was the process of constructing and selecting features. While we feel that we managed to do this in an efficient yet effective way, we concede that this process did have some limitations. A majority of our feature selection was done using regularized Logistic Regression; this is effective for selecting features that do well in a Logistic Regression model, but may miss features that, while useful for classification, don't lend themselves to linear separability of classes. This is likely the reason for the comparatively low performance of the Random Forest and K-Nearest-Neighbors using our final set of features. We may have been able to improve on this front had we instead used another, less selective method of feature selection such as RFE with the Random Forest; however, the severe computational demands of the Random Forest (and most other high-complexity models) would have made this difficult, which is why we opted to use the more efficient regularized Logistic Regression. From what we can tell, there is considerable noise in the data to be sifted through to create an effective set of features, and our method of feature selection, while efficient, was likely heavy-handed in its selection.

We also found that our method of weighting past game statistics was largely ineffective, only hurting our model performance, aside from the $\beta$ parameter. There are certainly improvements to be made here; we only considered a scheme that was inversely exponential in the time between games. A scheme that is instead linear or even logarithmic in this aspect may have better estimated future team performance, as the weights attributed to games longer in the past would diminish less dramatically than with our current exponential scheme, reducing the variance in our weighted means. Additionally, the data only contains games from October 2023 to April 2024; it is unlikely that any team dramatically changed in their performance in this small amount of time, so weighting by time may simply not be an effective strategy. Further, the $\gamma$ parameter of the model is a rather primitive method of considering past matchups for weights. We had an idea of creating a method of measuring the similarity between teams by considering the $L_2$ norm of the difference of their mean standardized statistics and then using this measure for weighting, though we did not get the opportunity to implement this.

To improve model performance, more specific information about each match, such as specific player statistics and the roster of each game, may help. This way, we can account for events, such as a star-player's injury, that would certainly affect the outcome of a match. However, there are also other improvements that could be made with our approach of the current data. With more domain knowledge, it may have been useful to create measures of teams' playstyles; if we know that both team A and team B typically play defensively, it may lead to us considering different features more heavily for the prediction as opposed to if both teams typically play aggressively. Though, without deep knowledge of professional basketball, it is difficult to pick out these statistics and the method of measuring playstyles in this way. Additionally, we did not consider past match results as deeply as we could have; say we wish to predict the result of team A versus team B. If we find that team A beat team C in the past but team B lost to team C, that should lead us to believe that team A will win against team B. However, we did not have the opportunity to implement this sort of matchup statistic.

While there may have been limitations in our process, I believe our results are still quite interesting. Considering the popularity of professional sports and the massive size of the sports betting industry, it is notable that the results of these matches can be predicted with nearly 70% accuracy with so few predictors, using quite simple models. It may be interesting to find new sets of NBA data or to scrape data to append to this set, then try to modify our approach and see if we can improve our accuracy further.