NBA ELO

Addison Wood

CMSE381 - Final Project

Introduction

Every month, over $500 million is bet on NBA basketball games [1]. This is a ludicrous amount of money, and as such, there is a ludicrous amount of money to be made, if you just have a bit of an edge. In addition, there is a plethora of data that can be used to try to extract insights. Thus, the goal of this project is to take that data, and begin to maximize game result prediction accuracy.

Dataset

This data comes from FiveThirtyEight's *The Complete History of the NBA* [2], which compiles team ELO ratings on a game-by-game basis, from 1947 through 2015. In the article *How We Calculate NBA Elo Ratings* [3], again on FiveThirtyEight, by Nate Silver and Rueben Fischer-Baum. Elo is a zero-sum rating system, in which teams "gain Elo points after winning games and lose ground after losing them. They gain more points for upset wins and for winning by wider margins." The dataset, amongst other descriptive, but not really factorial or numeric variables, contains the Elo for each team and their opponent heading into the game, as well as if the game was home, away, or neutral. These three seemingly simple values will constitute the bulk of the modeling in this project.

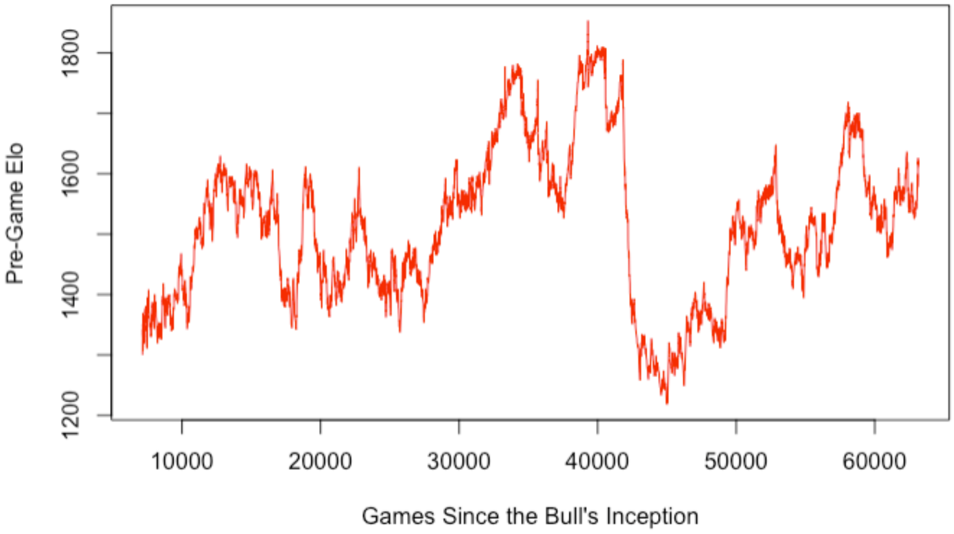

Figure 1: Historic Elo of the Chicago Bulls.

Methods

In order to predict the outcome of NBA games using this data, we will be implementing a variety of classification methods. These methods will be, in order of increasing complexity, k-nearest neighbors, logistic regression, linear discriminant analysis, and quadratic discriminant analysis. In each of these methods, a probability is generated, and depending on if it is greater than or less than 0.5, a prediction of a 'win' or a 'loss' is made. This is then compared to the prediction made from the `forecast` variable in the dataset, which is generated from the Elo ratings as well. The goal for this part of the project is to surpass the accuracy of this `forecast` variable – or at least come as close as possible.

K-Nearest Neighbors

K-Nearest neighbors is a method of classification, in which the classes of the neighboring training samples are averaged to create a prediction for the testing sample. In this situation, I only used two predictors: `elo_i` and $d = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$ `opp_elo_i`, which are the pre-game Elo ratings for both the team and their opponent. In the other methods I will use a variable that is a factor: `game_location`, but KNN only works with numeric variables, so this method will have two predictors, instead of three. For KNN, the k closest other points are determined by their Euclidian distance to the test point. In this case, *x* is *elo_i*, and *y* is *opp_elo_i*.
Here is the equation for Euclidian distance:
In order to do cross-validation, I found that – most likely – the best value of *k* was very high. I tested values between 11 and 201, and found that 201 was the best, although it was plateauing.

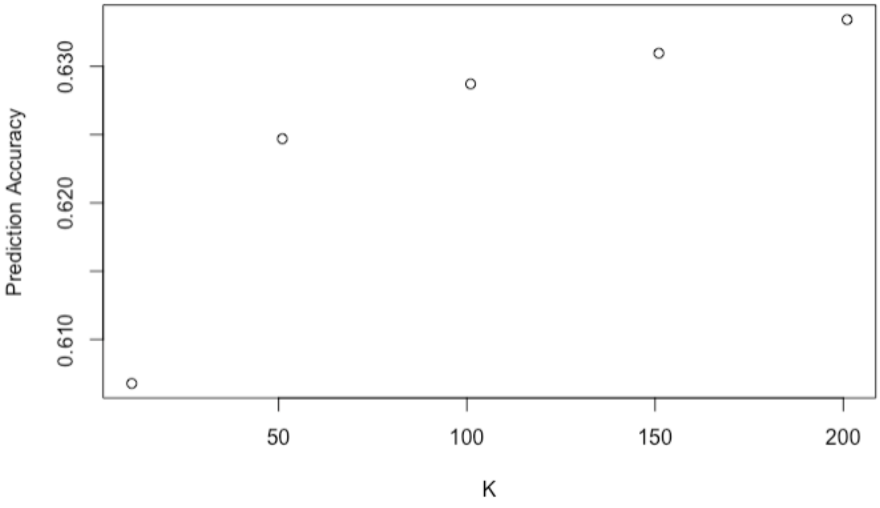

Figure 2: Prediction accuracy for various *k* values in the KNN model.

As we get larger in k-values, the computational cost increases exponentially, so I made the decision to cut my losses, save a significant amount of computation time, and run the KNN model with k=201.

In addition, to cross-validate the model itself, I created a train-test split in which a single year is a testing set, while the rest of the data is the training set. As the data goes from 1947-2015, there are 69 unique testing sets, and each one creates a train/test split of approximately 1.5/98.5. This seems like an appropriate compromise, as the dataset is likely too big to do LOOCV for each method, and splitting on the year means we can generate interesting data for each year without too much more computational expense.

Logistic Regression

This regression method, seeks to us log likelihood estimations in order to minimize the prediction error by maximizing the likelihood. To do this, we find the most likely model, give it a log transform. Now, there is a function from 0-1 in which there is a strict maximum. That point is our likelihood. In this situation, if the max loglikelihood is less than 0.5, the prediction is 0, or a loss. Conversely, if the max loglikelihood is greater than 0.5, the prediction is a 1, or a win. This algorithm boils down to this equation:

$$p(X) = \frac{e^{\beta_0+\beta_1 X_1+\beta_2 X_2+\ldots+\beta_m X_n}}{1+e^{\beta_0+\beta_1 X_1+\beta_2 X_2+\ldots+\beta_m X_n}}$$

Linear Discriminant Analysis (LDA)

This is a classification model, like the other, but is unique in that is uses Bayes Theorem, which is shown here:

$$\Pr(Y=k|X=x) = \frac{\Pr(X=x|Y=k)\cdot\Pr(Y=k)}{\Pr(X=x)}$$

In this case, P(X=x|Y=k) is the density for X in some class k, while Pr(Y=k) is the prior probability for that class k. The classification of the point depends on the which class has the highest density, using this density function:

$$p(x) = \frac{1}{\sigma\sqrt{2\pi}}e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

Once the density is generated, it must then be transformed to a probability, so it can be classified based on if it is greater than or less than 0.5, as in the previous classification methods. The equation we use to do this is:

$$\widehat{\Pr}(Y = k|X = x) = \frac{e^{\hat{\delta}_k(x)}}{\sum_{l=1}^{K} e^{\hat{\delta}_l(x)}}.$$

The cross-validation for this method is the same as the others, where each year is its own test set, with the rest of the samples constituting the training set.

Quadratic Discriminant Analysis (QDA)

This is similar to the method for LDA, as the distribution is again Gaussian. For this method, however, there is a different covariance matrix for each class. This leads to nonlinear decision boundaries, and generally a more flexible model than LDA.

Results and Discussion

In this project, a cross-validation method was used to find a more accurate test accuracy than a single split (or no split at all). In this case, each year was a test split. So, for instance, all games from 1950 were excluded from the model creation, and constituted the test set, and the model was generated on all games that were not played in 1950. Then, the model generated predictions for those games in 1950, and they were compared against the real-world results from that year. That was repeated for each year from 1947-2015, resulting in 69 different iterations, in which the train/test split was usually around 98.5% / 1.5%. The test prediction accuracy was averaged for all of these iterations to generate the methods official prediction accuracy.

The equation to generate this prediction accuracy was simple, just

$$prediction\ accuracy = \frac{correct\ predictions}{total\ predictions}$$

We generated the prediction accuracy for KNN when *k*=201, logistic regression, LDA, and QDA, and compared all of those to the prediction accuracy for the *forecast* variable already present in the dataset, which was the result of FiveThirtyEight's model, which also utilized Elo to some extent.

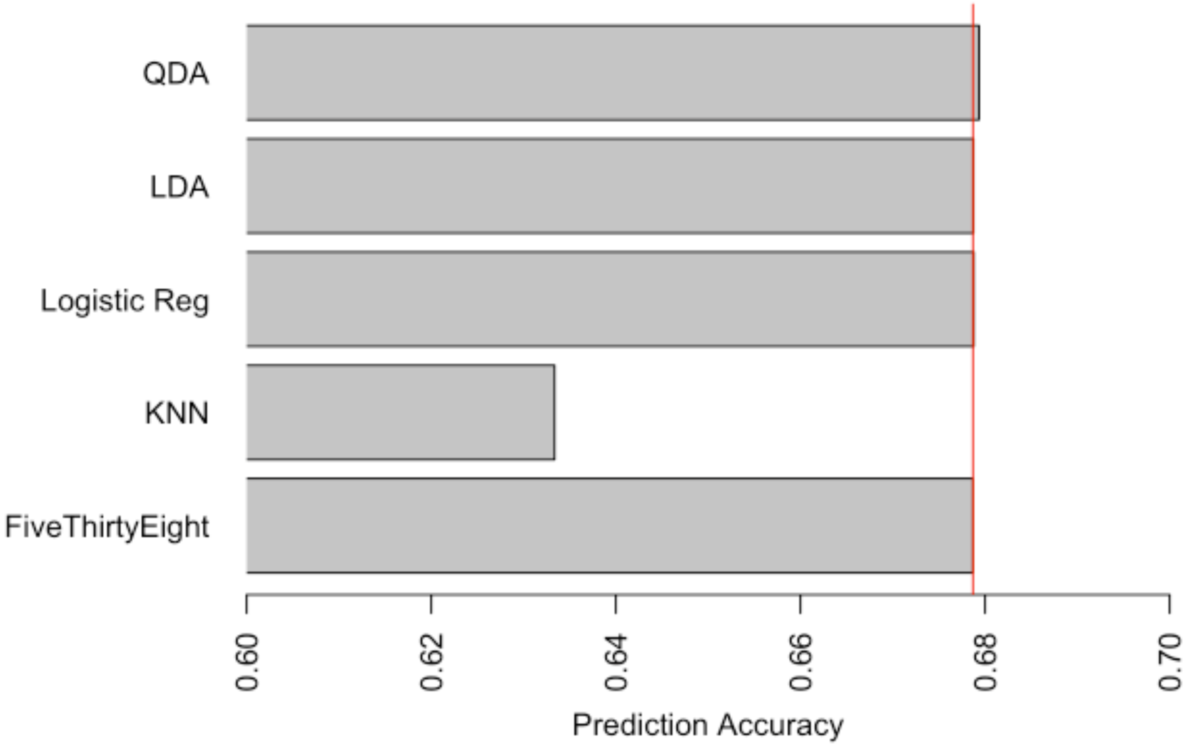

Figure 3: Prediction accuracy of the various models

Here, we can see that KNN is the obvious loser amongst the models. Although the reduced x-axis can be deceiving, and the difference is only around 0.045, in relative terms, KNN lags significantly. This could likely be due to the fact that the model was not completely optimized for the *k* value, as the prediction accuracy was still increasing when we reached our limit for computational time and power.

In addition, we see that logistic regression and LDA perform almost exactly as well as the FiveThirtyEight model, and QDA performs just a hair better, The difference between QDA and FiveThirtyEight's is likely not significant, however, and could fathomably change if more games – for instance the games since 2015 – are added to the dataset. The incredible similarity between logistic regression, LDA, and FiveThirtyEight's model, however, leads me to believe that they may be using one of the two models in the predictions they generate, as well.

As we subset by year to navigate our cross-validation, we can also explore how well each model did by year.

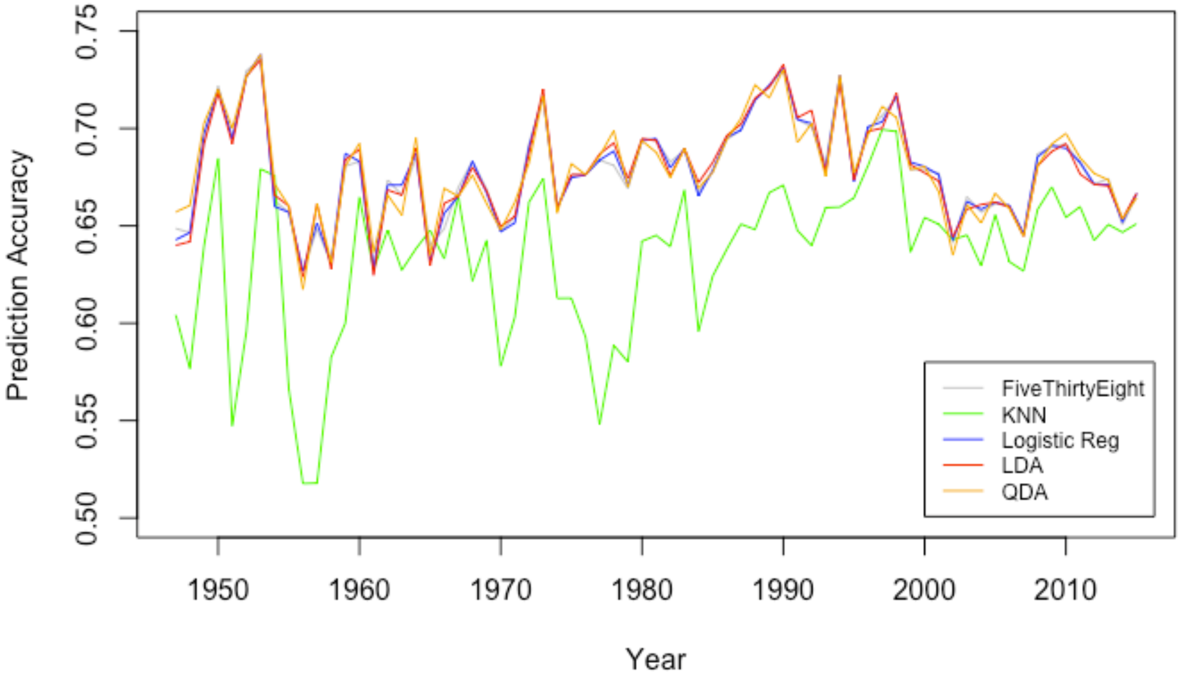

Figure 4: Prediction accuracy for each model, for each year (1947-2015)

Here, we can see again that KNN lags behind, while FiveThirtyEight, logistic regression, LDA, and QDA all perform similarly. Logistic regression performs so similarly – even year-to-year – to FiveThirtyEight, that I am led to believe this is the model they used to generate their predictions.

References

[1] R. Scott, "How much money is wagered on NBA basketball?," *American Gambler*, May 10, 2019. https://www.americangambler.com/how-much-money-is-wagered-on-nba-basketball/

[2] N. Silver, R. Fischer-Baum, "The Complete History Of The NBA," *FiveThirtyEight*, updated April 19, 2021. https://projects.fivethirtyeight.com/complete-history-of-the-nba/

[3] N. Silver, R. Fischer-Baum, "How We Calculate NBA Elo Ratings," *FiveThirtyEight*, May 21, 2015. https://fivethirtyeight.com/features/how-we-calculate-nba-elo-ratings/