Cavan Ingram

Statistical Modeling

26 April 2020

Predicting Basketball Wins and Losses Statistical Modeling Project

Introduction

The goal of this project is to create a model with the dataset found that can accurately predict whether a given NBA basketball team will win or lose a game. The purpose of this is not necessarily to predict wins and losses for games that have not already been played, but instead figure out the important statistics and variables that go into a basketball game. This problem will be answered using three models: logistic regression, K-Nearest Neighbors, and random forests. These models made in Jupyter Notebook will help visualize what is going on in the dataset and see what variables are most important to the outcome of a basketball game.

Dataset and Features

Overview

The dataset that I used included a wide range of information and statistics. The data covers four years of NBA games from 2014 through 2018 which results in nearly 10,000 games to study. With this particular dataset, each game has multiple different categories and statistics such as the teams playing, who was home, the date the game was played, as well as all the typical statistics you would see in a box score for a given NBA game (such as rebounds, points scored, steals, and fouls among others). Fifteen input variables are used for this project to help predict whether a given team will win a game.

Inclusion Criteria

To enhance the models used in the project, some variables were omitted as input variables because they were not necessary or enhancing the model in any way. The reason for this is that some of their p-values were too high to be used in the models. Because of this, the fifteen variables used all have zero p-values when being used to predict whether a given team will win a game or not. However, some variables were omitted simply because they were too good at predicting whether a team would win a game or not. These variables included the points scored and three-pointers made for a particular team and their opponent. The reason points scored was too good at predicting is that if the score of the game is already known, then no other variables are needed to predict whether a team would win the game or not. The reason three-pointers made was excluded follows along the same idea because if a team made more three-point baskets, then they would have a higher score.

Data Engineering

Little to no data engineering was needed for this dataset. The reason for this is that basketball has straight-forward and well recognized statistics that do not need to be manipulated. The only manipulation for this dataset was to make categorical variables into integer numbers and to delete unnecessary information.

Outliers

There were no real outliers in this data set. The main reason for this is that nearly all of the numbers used for the model in the dataset were less than one hundred and there are not huge discrepancies from game to game due to the nature of basketball.

Features

The model includes sixteen numerical input variables to predict one categorical variable. The statistics that will be used are a given team's and their opponent's field goals made (FG), field goals attempted (FGA), three-point field goals attempted (FGA3), free throws made (FT), offensive rebounds (OREB), assists (AST), blocks (BLK), and turnovers (TO). These features did not need to be normalized due to the fact that all of the numbers used in the project are within a reasonable range of one another. These statistics will be used to predict whether or not a NBA team will win a game.

Models

Model 1 – Logistic Regression

The problem that we modeled was a classification problem to determine whether or not a given NBA team will win or lose their game versus their opponent. We trained about eighty percent of the data and used basic basketball statistics to attempt to classify the other twenty percent of games correctly. Since this was a classification problem, the first model used was a logistic regression model since it is better at calculating classification problems then a linear regression model. First, we simply evaluated the model with different features to get the best model possible. We did this using backward selection. We started with a full model of variables and eliminated variables one by one to get a better model. Values with too high of a p-value were eliminated until we were left with a total of sixteen variables. Feature scaling was not used for the variables because the model performed more accurately without it. When splitting the data into the training and testing data, we needed to check to make sure that the error rates (that help show how much error is in the model) were similar to one another. The error rate before splitting was .9586 and the error rate after splitting was .9577. We reran the error rates three times for the training data and got three values around .96. The training data error rates and the original error rates were close enough together to show that there is no overfitting happening in the model.

Further, the rates are high and show that there is not much error in the model, so we can continue using this model and testing other features on it to determine its validity.

After using and executing the logistic regression model, we created a confusion matrix to continue to figure out how the model is performing. The confusion matrix helps create a visual representation of how well the model did at predicting correctly whether the given NBA team won the game. From seeing Figure 1, one can tell the model did a good job at predicting wins and losses. The top left dark blue box shows the percentage of games that were true negatives. That means that the model predicted the team would lose the game and they did. The darker bottom right box shows the percentage of games that were true positives, meaning the model predicted the team would win and they did. However, the top right and bottom left boxes represent false positive and false negative percentages respectively. These are areas where the model failed to accurately predict whether a team won or lost. Both of these percentages are small, so the model still seems to be performing well.

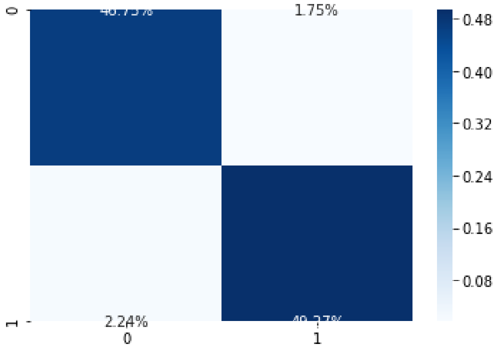


Figure 1

Next, we plotted a Receiver Operating Characteristic curve (ROC curve). This curve is made by plotting the true positive rate against the false negative rate and helps show how well the model was at picking between win and loss. As seen in Figure 2, the Area Under the Curve (AUC) is very high. This high AUC shows that the model is doing a good job at predicting wins and losses.

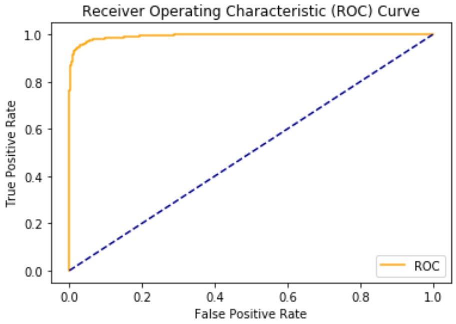


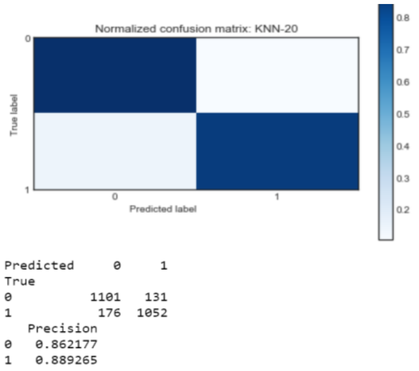
Figure 2

We also fit LDA (Linear Discriminant Analysis) and QDA (Quadratic Discriminant Analysis) models to the data. The LDA predicted that the team would win 1258 times while the QDA model predicted the team would win 1266 times. The original logistic regression model predicted the team would win 1267 times, but all the models are accurate in their predictions. However, the original logistic regression model still has the best prediction averages based on the classification reports. It has an average of .960 while the LDA and QDA models score .956 and .943 respectively.

Ultimately, the logistic regression model used performed well and was accurate in predicting wins and losses. The high output from the ROC curve and the numbers in the confusion matrix help show the accuracy of the model. Not only that, but the variables used in the model (training data) all have zero or near zero p-values and the model has a .8519 R-squared value. This shows that the model accounts for eighty-five percent of the data. The logistic regression model used for this classification problem is more than adequate at predicting wins and losses given certain common basketball statistics.

Model 2 – K-Nearest Neighbors

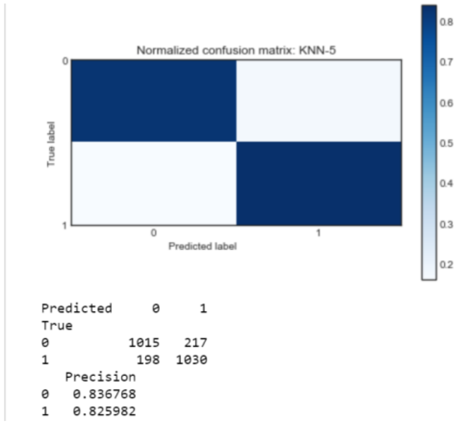
We are continuing to model the problem of determining whether a given NBA team would win on a given day, but instead of using logistic regression, we will use a non-linear model of K-Nearest Neighbors. We are still using training data to train our model to then perform on our testing data. This model tries to estimate the conditional probability for a class by looking at and using the nearest "K" data points. For this model, we used four different numbers for "K" to get a better understanding of which number worked the best and when it started to perform poorly. We chose to use this method, because it is assumed that the teams who win will be clumped closer together and the team who lose will be clumped together. The reason for this is that the teams who win will usually score lots of points and the teams who lose will score lower amounts of points (which will affect the variables used). However, this will not work perfectly, because there are some games that can be very high scoring and close, which will put the losing team in the midst of the winner data points (and vice versa). This model is fairly easy to use because there is no real fine-tuning that needs to be done (other than adjusting the number of neighbors one wants to use).



```
Predicted     0     1
True
0          1101   131
1           176  1052
   Precision
0   0.862177
1   0.889265
```

Figure 3

For this model, we used four different numbers of K. We used 5, 10, 15, and 20 neighbors try and get the best model possible. We used these numbers and then plotted a confusion matrix to see how accurate our model performed using the data. In Figure 3, one can see the confusion matrix that was created when using 20 neighbors. Overall, this was the highest performing model of the four as it was the most accurate overall. This makes sense because the more datapoints the model can group (in this case 20) the more accurate it can be at correctly guessing whether the given team will win or lose the game. Specifically, this model had a precision of .852 when correctly predicting a loss and a precision of .889 when correctly predicting a win. The other neighbors performed well too. The model with 5 neighbors (shown in Figure 4) performed the worst. It has a precision of .837 when predicting a loss and a precision of .826 when predicting a win.



```
Predicted     0     1
True
0          1015   217
1           198  1030
   Precision
0   0.836768
1   0.825982
```

Figure 4

The results from this model were promising. We were able to predict whether a team would win or lose with over .8 precision (depending on the number of neighbors used). Our data that was used is doing a good job at being able to predict whether a team would win or lose. We could get a higher level of precision if we used more neighbors, but it would not be a significant increase (and in some cases, the precision might even decrease).

\

Model 3 – Random Forests

The last model we will use for determining whether a given NBA team will win their game is the Random Forests Model (still using the training data to train the model and the testing data to test the model). This model deals with decision trees, but enhances the model to be more reliable and accurate. The reason for this is that basic decision tree models are often overfitted and cause the model to not perform well when seeing new data for the first time. Random Forests allows to use a "bootstrapped" dataset and use a random subset of features for each decision tree. This makes a wide variety of uncorrelated trees to use and better classify the problem.

For this model, we created scatter plots for our data and we also used the Random Forest Regressor with different numbers for the max features to see what would work the best. For the first model, we used a max of 15 features for the model. After applying this, we received a low mean squared error of .0831. Next, we tested the same model, but with a smaller number of features (6). This model had a higher mean squared error of .101. This model did not perform as well because the mean squared error was higher. This means that the predictions were farther away from the mean than the model that had a maximum of 15 features. Knowing this, we were able to print out a confusion matrix to better determine how the model performed. In Figure 5, one can see that this model (again with 15 features) performed well. The top left dark blue box shows where the model accurately predicted where the given team would lose the game. The dark bottom right box shows where the model accurately predicted where the given team would win the game. However, the top left and bottom right boxes show where the model failed to accurately predict whether a team won or lost. The model performed well and was more than 90 percent accurate in predicting wins and losses.

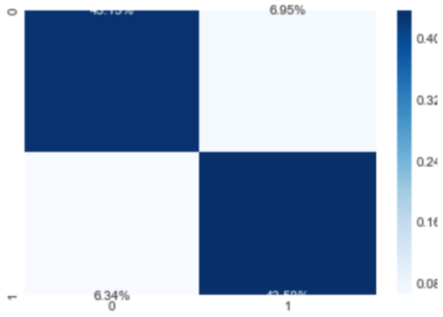


Figure 5

Along with these things, we were able to determine the importance of each feature in our model (Figure 7). This bar graph shows us the most and least important features in the graph. The graph shows that the most important features are the ones that involve a team scoring or giving up points. This makes sense because the team with more points wins the games. This obviously makes the features not directly associated with points less important.

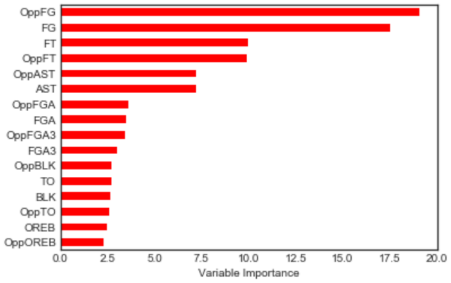


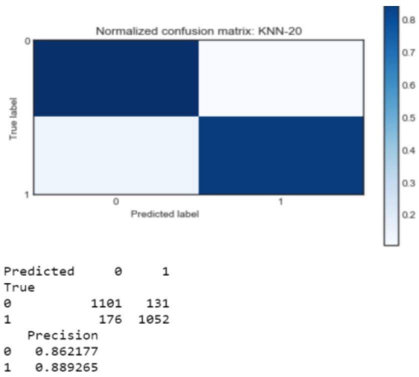
Figure 6

The results we obtained from this model were positive. We were able to get a low mean squared error along with pretty good precision on the model. The more features we used, the better the model performed. Hence, we decided to use all fifteen of the features that are used for the model.

Model Comparison

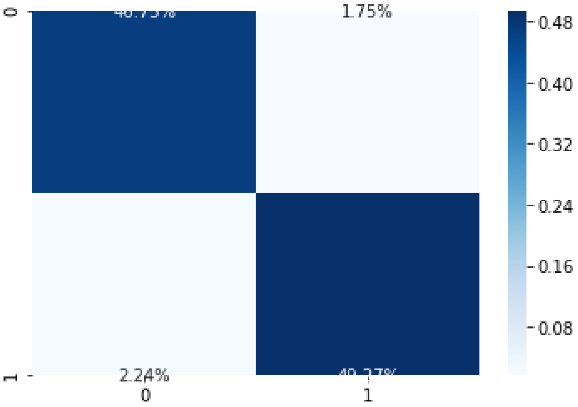
There was a total of three models that we used to evaluate the performance of our data to determine a classification problem of whether an NBA basketball team would win a given day. There was a total of sixteen variables that we used (including field goals, rebounds, assists, etc.) to help determine the answer to this question. These variables were determined because they help explain how a team performed in the game. For the models, we trained about eighty percent of our total data and set aside the other twenty percent for testing (more about this and the process of selecting the variables in the Models section above). The three models that we used to determine whether a team would win or lose the game were logistic regression, K-Nearest Neighbors, and random forests.

The reasons we selected to use each model differ in slightly different ways. The first model we used was a logistic regression model (which is linear). We chose to use this model with our data because we were trying to solve a classification problem (win or loss) and logistic regression is able to take multiple variables and explain the relationship between them to help figure out whether a team won or lost. The next model we used with the data was a non-linear model in the K-Nearest Neighbors model. This model was chosen because it is assumed that the winners would have data points clustered together and the losers would have a different cluster of data points together. Lastly, we used another non-linear model in random forests. We chose to do this second non-linear model because we wanted to use decision trees in a more reliable way (to prevent overfitting). Ultimately, we need to look at how each model performed to determine what model works the best and why.



```
Predicted     0     1
True
0          1101   131
1           176  1052
   Precision
0   0.862177
1   0.889265
```

Figure 8



One of the best ways we can easily compare each of the three models is to make confusion matrices for each model and see which model most accurately classified the data points. First, let us look at the confusion matrix for the linear logistic regression model. As seen in Figure 7, one can see that this model performed extremely well and was better than 95 percent accurate in predicting whether a team would win or lose a given basketball game. Next, we can look at the confusion matrix created for the K-Nearest Neighbors model and see how that model performed. In Figure 8, one can observe that this model performed well, but not as well as our logistic regression model. The reason for this is that the K-Nearest Neighbors model only accurately predicted a little over 85 percent (in the best case) of the data points as a win or loss. The reason this model did not perform as well as the first model is that the number of neighbors (in this case twenty) were not as good as predicting the wins and losses due to the fact that losers of the game can still have good statistics, but lose (which would put them in a similar cluster to the winners). Lastly, let's compare the random forests model to the other two models used for the project. The confusion matrix for this model, shown in figure 9, is similar to the K-Nearest Neighbors model (as it accurately precited a little over 85 percent of the data) but was still behind the logistic regression model. The reason this model was so similar to the K-Nearest Neighbors model is because the decision tree will work in a similar way to the number of neighbors, because losers might still have good stats which will put them in similar situations/trees as the winners (which is why this model was still not as good as the logistic regression model).

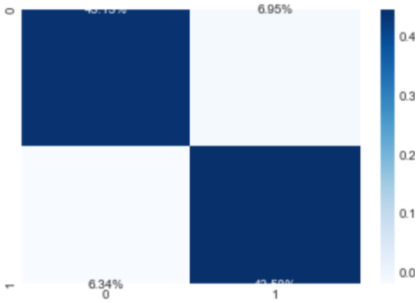


Figure 9

Now that we know that the logistic regression model was the best (by a good margin) at accurately predicting whether a team would win or lose their game, we can take a closer look as to why that is. By plotting a Receiver Operating Characteristic curve (ROC curve), we can see that the area underneath the curve is very high (shown in Figure 10). This is a good sign because it means the model did a great job at predicting whether the team would win or lose the game. Overall, we can see that the logistic regression model.

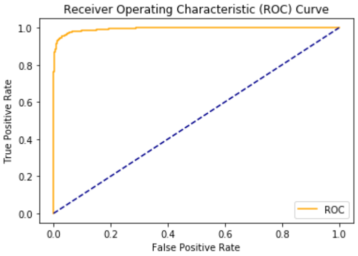


Figure 10

<u>Conclusion</u>

Through the models created during this project, it is safe to say that the models (especially the logistic regression model) were accurately able to predict whether a given NBA basketball team would win or lose their game. This question was not too difficult to answer given the data used from the dataset because the games have already been played and that is why we have the statistics to help predict the outcome of a given game. In the future, it might be beneficial to use this project as a jumping off point to help predict outcome of games that have not already been played. An example of this might be taking a given team's past five games and using those statistics to determine what the outcome of the sixth game might be. However, basketball, like all sports, is truly unpredictable due to things like injuries, the stadium the game is played in, and coaching. This project was beneficial in observing that the most statistics in the outcome of a basketball game are those directly correlated with scoring points. Not only that, but this project shows that defense might be more important than offense when figuring out what team will win and what team will lose. More research can be done with this dataset to improve the results and discover new findings about the game of basketball.