CS4731: Predicting NBA Game Outcomes using Linear Regression and Naive Bayes Classification

Merritt Hancock, Chuck Millsap

December 1, 2019

1 Problem Definition

Since sports matches have been around, individuals have tried to predict the outcome of said matches. Whether knowingly or unknowingly, every prediction takes into account the team's data and prior performance. The data is then passed into our self-formed algorithms of sort, and then a winner and loser are predicted. These self-formed algorithms vary in their complexity which makes, when bet upon, certain people wealthy or just gives them bragging rights among friends. Through our time learning about computing and machine learning, we have come to the realization that computers are just better at analyzing data than us. This left us with a question. Just how much better are computers at predicting the outcome of sports matches?

For this project, we tried to answer this question by building a Naive Bayes Probabilistic Classifier to predict the outcome of any given NBA match-up. Additionally, we used Linear Regression to get a better idea of what statistics best translate to better team performance. When selecting what sport we wanted to analyze, we considered the NFL, NBA, and Premier League. The main issue that presented itself was the turnover each team had every year in personnel. This can cause a team who was a championship contender one year to be a rebuilding team the next year. We chose the NBA because there are 1230 total games a year in the regular season giving us more than enough data points for our classifier.

We decided to train our model on the 2018 regular season and then compare the results from games already played in 2019 as well as the 2018, 2017, and 2016 regular seasons. This gave us a better idea of how well the model performs over the course of different seasons. Additionally, the accuracy of the classifier should help demonstrate how turnover in personnel affects team performance year over year.

2 Data Sources

Once of the benefits of doing analysis on the NBA is the plethora of data available. When researching we found an awesome python API named Sportsreference. Sportsreference is a

free API that pulls statistics from www.sportsreference.com for all the major sports leagues and allows them to be easily imported into python applications. For the NBA package of the API, there are Boxscore, Player, Roster, Schedule, and Teams subcategories. Each of these categories contains valuable NBA information and statistics. For this project we used each of these to gather needed data.

3 Additional Resources

For this project, we did use a number of different python libraries.

```
1 from sportsreference.nba.teams import Teams
2 from sportsreference.nba.schedule import Schedule
3 import sportsreference.nba as nba
4 import numpy as np
5 import matplotlib.pyplot as plt
```

We used sportsreference to import the data needed for the assignment. Additionally, we used Numpy for array creation as well as the other functionality it provides. Lastly, we used MatPlotLib for plotting our results from the experiment. The use of theses libraries gives us the additional needed functionality beyond that of what comes with just python.

4 Methods

4.1 Method 1

The first method we chose as a solution for our problem was the use of Linear Regression. Linear Regression gave us the information needed to make an informed decision on what statistics best translated to overall team success which, when building our classifier upon, should give us the best results. We began by pulling the data that was needed from the API. For this method, we decided to compare Point Differential, Effective Field Goal Percentage(eFG), and True Shooting Percentage(TSP) against win totals.

Point Differential is defined as the number of points allowed subtracted from the number of points the scored. This has been a reliable statistic as it doesn't just take into account whether a team won or lost, but it also factors in how much a team won or lost by. Effective Field Goal Percentage is a bit more complex than point differential. It factors in a different weight for a made 3 point shot.

$$PD = PTS - PTSallowed$$

$$eFG = (FG + 0.5 * 3P)/FGA$$

$$TSP = PTS/(2 * TSA)$$

True Shooting Percentage is also useful as it weighs free throw attempts differently than field goals. Many teams use this metric as an overview of how their offense if performing. Once we had the data, we began performing operations to get the desired metrics. We then graphs the different teams values against their win totals. All of these metrics had relatively linear results, but point differential was the most linear and had the least noisy data points.

4.2 Method 2

When it finally came to the task of predicting NBA games, we decided to use a Naive Bayes Probabilistic Classifier. Through our research, we found a number of models that trained their classifiers on different metrics. We were able to decide on the infrastructure of our classifier once we discovered Dean Oliver's "Four Factors of Basketball Success." Oliver was one of the pioneers for basketball analytics and wrote a book on the topic. His "Four Factors of Basketball Success" revolved around shooting, turnovers, rebounding, and free throws. He created a model where he weighted each of these statistics differently. He weighted his values as 40 percent shooting, 25 percent turnovers, 20 percent rebounding, and 15 percent free throws. We decided to give our classifier similar weighted values.

```
team_val = np.float(0.4*TSP[name] + 0.25*TOV[name] + 0.20*REB[name] ...
                                    + 0.15*FreeThrow[name]+ ...
                                    0.2*team_differential[name])
```

As seen, we built a very similar model. Additionally, we did include, from our linear regression research, point differential with a weight of 20 percent. After we had the classifier built, we did four different implementations for the 2019, 2018, 2017, and 2016 seasons. This allowed us to see how the model, being trained on 2018 data, performed on other seasons as well. Once we had the results we put them into a bar graph to evaluate how well the classifier performed.

5 Evaluation

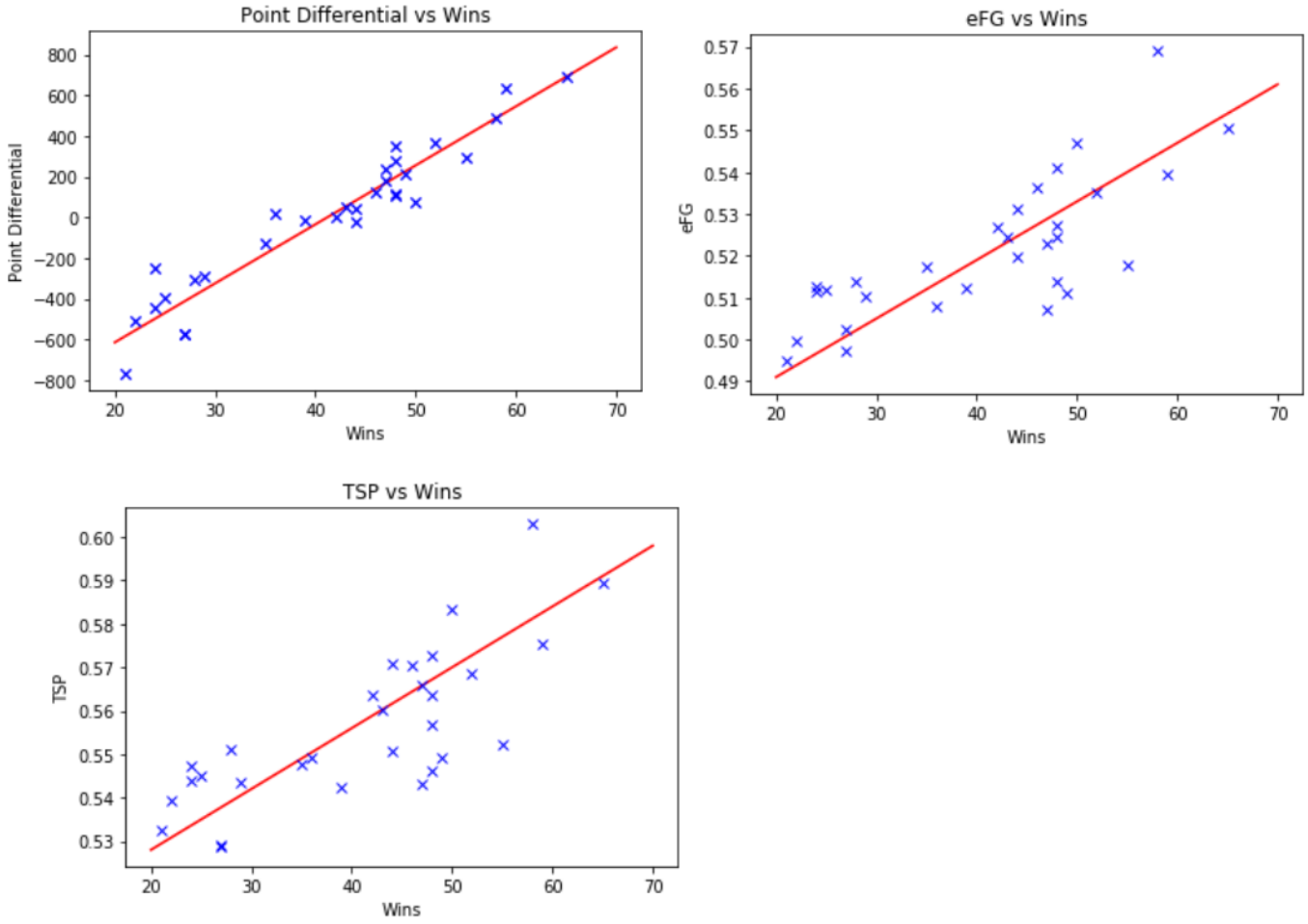
To evaluate the first method we started out by graphing the different metrics against win totals. Each metric is based on statistics from the 2018 regular season. This means that the stats are gathered from 82 games for 30 teams totaling 2,460 individual data points. Then, depending on the nature of each statistic, either the average or total of each statistic is calculated. This makes it so that each team has one value for each metric that can then be compared to win totals. Once graphed, we fit a line going through the upward trajectory of each group of points. We optimized the fit of each line using a loss value in order to achieve the best possible fit. We were then able to visually see which statistic had the best correlation with win totals.

$$L(B_0, B_1, Y, X) = \frac{1}{N}\sum_{i=1}^{N}(\hat{y}_i - (B_0 + \hat{x}_i B_1))^2$$

The evaluation for the second method took a little more consideration. Our classifier is was trained on data from the 2018 NBA regular season which, mentioned above, contains 2,460 individual data points. Typically a classifier is evaluated by its accuracy. This led to us evaluating ours using misclassification rate. This rate is a measure of the number of wrong predictions divided by the total number of classifications. This will give us the desired accuracy of each implementation of the classifier, and we will have the ability to numerically measure how the classifier performs over time.

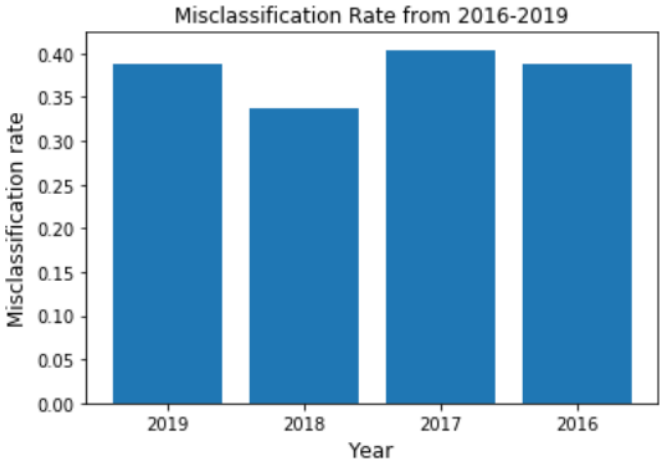
$$MCR = totalwrong/total$$

6 Results



When analyzing the graphs from the first method, each of the metrics seems to be relatively linear. Although this is the case, point differential seems to be the most reliable of the metrics. It's graph has the least noisy points and also is the most linear. Effective field goal percentage and true shooting percentage both seem pretty equal in how linear and scattered their points are. This left me with the conclusion that point differential is the

most reliable when predicting team performance. Because of this, I included it as a weighted value for the classifier in addition to Dean Oliver's "Four Factors."



When measuring the accuracy of the classifier, it hovered pretty steadily around 60-65 percent. We experimented with the weights of the values from the model of the classifier, but we were unable to raise the accuracy noticeably above that of the original weights. When we first started the project, we had hoped to have a higher accuracy for the classifier to create a more reliable prediction for each game. The accuracy may not be where we had hoped, but the model is still more reliable than a coin flip which, in many cases, is what predictions can come down to. The other factor to consider is how well the model performs across multiple seasons. As expected, the model performs best on the season it was trained on, 2018. Also, as expected, the model performed worse in 2017. There was about a 7 percent rise in misclassification from 2018 to 2017. This mainly due to the shift in player personal that happens every off season. Interestingly enough, the rate goes back down when going from 2017 to 2016 by 1 percent. This is likely caused by a team that had drastically different personnel from 2018 to 2017. The accuracy for 2019 thus far has actually been fairly decent in its small sample size.

7 Discussion

The main factor that is hard to account for when predicting sports matches is that the players are in fact human beings. Any person can have an off night, and any athlete can have the game of their life. This is what makes sports exciting and fun to watch. The whole point of watching is not knowing what the outcome will be. When sports betting is concerned, the casinos also understand this. If a team is likely to win, the casinos will adjust the odds accordingly. This makes getting rich off of sports betting nearly impossible. A team can be the under dog and win, but if the casino adjust their odds wisely then they won't even have to pay for the loss.

Our model isn't sophisticated enough to predict how much a team will win by, but when just predicting the winner it is still fairly accurate. In a perfect world the model would be able to account for things like players missing games, rough stretches, and hot streaks.

For this reason, my opinion is that theses metrics are best utilized to help predict team performance over a larger stretch of games than a singular match-up. Every NBA team now has their own analytics department to help teams make roster decisions as well as aid in determining play styles. While it is difficult to predict a singular match-up, these metrics are still highly valuable in predicting team success.

8 Conclusion

Predicting sports matches is still far from a perfect science. There are factors that many models just aren't able to take into consideration. However, models can still be built that have relatively good accuracy. While these models do give valuable information when trying to predict games, they are far from reliable. The best use for this data is for long term bets concerning overall team performance. It can also be utilized by teams to find areas to improve as well as what areas, when improved upon, can increase their win totals. While our classifier isn't a "sure thing" when predicting the outcome of games, we still view it as an achievement when concerning the overall performance of a team.