Project Midway Report

Lee Richardson (lrichard), Daren Wang (darenw)
Chi Zhang (chiz2), Xiaofeng Yu (xiaofen1)

November 4, 2014

1 Question

Our goal for this project is to predict the outcomes of NBA basketball games as accurately as possible. To do this, we have collected data from various sources in order to find the best predictive features. The metric for success we're using is a 0-1 loss function. We will predict each game for an entire season, and then check the outcomes to see the percentage of games we predicted correctly.

2 Dataset

2.1 Data Sources

We did not have a processed dataset for this project, so we created our own database. The three main sources we used were ESPN's NBA website [7], basketball reference [6], and a new website from Jeremias Engleman [5]. We used the ESPN data to get information about all NBA games from 2009-2014. Specifically, this includes the game score, the home and away teams, the players involved and their statistics. Also from ESPN, we have a player database, which has 50 individual statistics for each player in each season. There are 7139 games in this dataset.

The next data source we used was basketball reference [6]. The main reason we used this site is because they have a larger individual player database, with information dating back to the 1950's and more advanced statistics, such as the widely used Player Efficiency Rating (PER), compared with the box score stats provided by ESPN.

The final source we used was from a website put together by Jeremias Engleman [5]. This site has Regularized Adjusted Plus Minus (RAPM) statistics dating back to the 1980's. This statistic has been widely adopted in the nba statistics community, and it's one of the few trustworthy stats which provides an individual assessment of defense. As we see below, RAPM is a very useful feature in predicting game outcomes.

2.2 Web Crawlers

To obtain all of these datasets, we used webcrawlers to pull them off their websites. All of these scripts can be found in our Github repository [8]. For the ESPN data, we used the BeautifulSoup package in the Python language. For the other two datasets, we used the XML package in R.

2.3 Merging Sources into One Database

One of the major obstacles in our project has been combining these three data sources into one single database. The ESPN dataset had both match and playerID's for each game, so merging the game statistics with the ESPN player database wasn't very difficult. However, the basketball reference and RAPM datset didn't have these identifiers, so it was more challenging to put them together. We ended up using the player's name, team, and season the join both of these datasets together with ESPN. Some common probelms we have were inconsistent spelling of names in different datsets, inconsistent team names, teams have changed cities, etc.. In the end, we were able to sync most of the idiosyncracies between the three datasets, which is important because we will have more interesting features than just box score statistics. That being said, we don't doubt that there will still be further cleaning to do, and we will deal with these situations as they arise.

We are using an SQLite database to store all of the tidied data. The design of this database follows the third normal form to ensure there's no redundancy, and the indexes were built on frequently used keys to ensure the queries are fast.

3 Literature Review

We looked into the literature to see if anyone had worked on the same or similar problems. We found a couple papers, [1] and [2], which also attempted to predict the results of NBA games. These papers used linear regression, logistic regression, naive bayes, and SVM's to predict the outcome of single games. They also used the same loss function that we are proposing, which gives us a prediction rate to shoot for when implementing our algorithms. Specifically, [1] achieved the highest single season classification rate of 73% in the 1996 season using linear regression. All of the other seasons/algorithm combinations had error rates from the mid-high 60's to low 70's.

One advantage we believe we have compared with these groups is that we have a more robust feature set, most notably we have RAPM. RAPM has been anointed by many as the next big thing [4] in basketball statistics, and we hope that using it as a feature can help differentiate our attempts at game classification. There is a full explanation of the statistic here [3], but the basic idea is to split each game into miniature games, each one occuring in time periods when there's no substitutions. Then these ten players play a certain amount of possessions on offense and defense, and we can estimate their overall effect on both ends of the floor. RAPM adjusts flaws in original plus minus, by coerrecting for the fact that each player's totals are heavily influenced by the play of his on court teammates, and pooling information from previous seasons to reduce the error margin.

4 Current Status

4.1 Training and test datasets

We have put a substantial amount of time into constructing our training and test datasets. So far we have created two, one which is using defensive and offensive RAPM, and the other uses all of ESPN's player statistics. To create these datsets, we went through each match to find the players on each team, and merged these players statistics from the previous season with the match results in the current season. To form the RAPM matrix, we used each players average minutes pergame from the season before as weights to compute weighted offensive and defensive RAPM statistic for each team. Say there's n players on each team, each playing m minutes per game with and RAPM score r in the previous season. Then the formula for the players weights and team offensive RAPM is (the computation is the same for defensive RAPM):

$$w_i = \frac{m_i}{\frac{1}{n}\sum_{i=1}^{n} m_i}$$

$$\text{Team Offensive RAPM} = \frac{1}{n}\sum_{i=1}^{n} w_i \times r_i$$

ORPM_home	DRPM_home	ORPM_away	DRPM_away	homeWin
-0.28	0.89	0.65	0.18	1
-0.28	0.89	1.15	1.05	1
-0.29	0.99	-1.44	0.12	1
0.03	-0.66	0.04	1.09	1
0.28	1.26	-0.81	-0.20	0
-0.75	-0.46	0.51	1.02	1

Table 1: A look at what our training and test datasets look like. The first four columns are features and the 5th column is our labels

Table 1 shows a look at our final tidied dataset.

4.2 Classification

After constructing these two matrices, we were able to run classification algorithms on them to test their prediction accuracy. Given the amount of time effort that went into creating the datasets, we didn't have a lot of time to experiment with different algorithms and features. However, we were able to fit a Naive Bayes classifier, trained on the 2009 season and tested on the 2010 season, which gave us 67% accuracy. We think this is a good sign, and with more work on the specific algorithms and features we think we can achieve rates similar if not higher than those achieved before us.

4.3 RPI

5 Realistic Goals

As mentioned above, a substantial amount of time was put into collecting and tidying the data, and creating testable datasets. Now that this has been completed, we have more time to focus on different classification algorithms and combinations of features, to see if we can improve our accuracy. Specifically, we hope to implement:

- Linear Regression
- Logistic Regression
- Support Vector Machine

We believe it is a realistic goal to achieve a greater 70% classification rate on one of our seasons before the end of the course.

6 Stretch Goals

One part of our classification that could be improved is that we are only using the previous seasons data to predict the current season. There are some obvious issues with this strategy. For instance, some rookies, even if they're very productive players, don't have statistics from last season, so we are just assigning them league average rates. Also, some players, for example former MVP Derrick Rose, was injured for two entire seasons, so using just the last year to predict a game and discounting his MVP season would misrepresent how good the Chicago Bulls are. To combat these issues, we think it may be helpful to use more than just last season data, perhaps using the average of the last three years, or set up a projection system. The first seems like a more tractable goal than actually maing projections, which could probably spawn an entirely new project.

Another thing we could try to to is predict the outcome of the numer of wins in a season, as opposed to just single game results. To do this, consider fitting a Naive Bayes model. The outcome is two probabilities of each team winning th game. We could compress these probabilities to sum up to one, and then use a Uniform $\sim (0,1)$ random variable to choose the winner for each game. We could do this for each game in the season and add up the wins and losses for each team. Then we could repeat this process a couple thousand times to see a distribution of total wins for each team.

References

[1] Matthew Beckler, Hongfei Wang, Michael Papamichael *NBA Oracle* 2009.

[2] Dragan Miljkovic, Ljubia Gajic, Aleksandar Kovacevic, Zora Konjovic *The Use of Data Mining for Basketball Matches Outcomes Prediction* 2010: SISY 2010

[3] Paul Fearnhead, Benjamin M. Taylor *On Estimating the Ability of NBA Players*. 2010: http://arxiv.org/pdf/1008.0705.pdf.

[4] Steve Illardi. *The next big thing: real plus minus*. 2014. ESPN.com

[5] Jeremias Engleman. http://stats-for-the-nba.appspot.com/

[6] Basketball Reference. http://www.basketball-reference.com/

[7] ESPN. http://espn.go.com/nba/.

[8] Repository for Game Simulation. https://github.com/leerichardson/game_simulation.