NBA Prediction Model

Ted Conklin
Computer Science
CU Boulder
rico6210@colorado.edu

Introduction

For my project I will be creating an NBA game prediction model. Sports have seen a leap in popularity over the past few decades, partially due to increases in technology making it a more consumable product. For many years now data scientists have been fascinated by the way data science can be applied to various sports, especially in the sports betting scene. Although I don't gamble myself, the rise of online sports gambling has triggered my intrigue in the matter. With basketball being among the most popular sports in the world, I thought this would be a great opportunity to apply what I learn in this class, in addition to the data science skills I have learned through the years, to create a model for myself that can predict the outcomes of professional basketball games.

2 Related Work

Using machine learning to predict he ouncomes of sporting events is an area of study with an abundance of research. One study I have found that exemplifies the process structure of sports prediction model development is the paper in reference[1]. The study follows various sports prediction models such as regression, neural networks, SVM's and ensamble methods on a single sporting event. The models were created using data from the individual players, the match, and bookmaker data. At the conclusion of the sporting event results, the study found that the Random Forest ensamble method provided the most accurate predictions, with an accuracy score of 70.9%, which was about 4.5% higher than that of the bookmaker predicitons.

While the study in reference[1] is a great reference for the structure of building prediction models, it doesn't address some key variables that are prevelent in the NBA betting field, most notibly the fact that it was only used to predict the winner of a game without accounting for the margin of victory. The study in reference[2] attempts to cover this aspect by predicting the total points each team will score based on various data attributes. To interesting attributes included in this study that I will pursue are the number of rest days each team has going into a game, and the impact of unavailable players. It is very rare that both teams have all of their players available for a regular season NBA, making the measurement of player absence of utmost importance for a useful model. One way the study represented this measurment is by weighting player importance as a function of their salary.

In addition the statistics above which I will be calculating myself, there is an abundace of statistics for the NBA teams that can be used to build prediction models. The study in reference[3] explores various combinations of these statistics to create a regression model for point spread predictions.

1 PROPOSED WORK

The plan for this project is to split the work into four sections of development. The first section will involve the collection, cleaning and formatting of data.

1.1.1 Data Collection. As far as sports data is concerned, there are a lot of datasets and methods to choose from. One method that I have developed is a Beautiful Soup web scraper that collects a vast

amount of data from https://www.basketball-reference.com/, which is a great site that I have used in the past for looking up both broad, and specific sports statistics. One of the biggest benefits of this web scraper is that the data is always up to date with the current NBA season and is run in python which makes it easy to store, format and manipulate the data as needed. From my python scraper I have built several data frames storing game results, team statistics, and opposing team statistics for every NBA season going back to 1990. I have also exported the dataframes onto my PC in the form of .csv files as a backup of my dataset.

1.1.2 Data Cleaning and Formatting. The next portion of the data collection component is to clean out data that isn't necessary for our model. Garbage data includes attributes such as 'arena name' and games that result in ties since they don't help predict a winner. Additionally, the game of basketball has changed substantially over the past decade or so due to increased 3-point attempts and rule changes causing higher free throw attempt numbers. For this reason, I will have to consider how far in the past I can use data for it to be relevant in predicting NBA games of today. Once I have all the relevant data, I plan to format it using Python, specifically pandas data frames.

1.2 Model Fitting. The second development phase of this project will involve taking the dataset from part 1.1 and using supervised, and semi-supervised learning to first create a binary prediction model. The learning model will be supervised with parameters being mined/calculated statistics, and the labels being + or -, representing a win or a loss. I will test multiple algorithms including KNN, AdaBoost and Bagging via decision tree classifiers, perceptron via linear classifiers and others.

1.3 Model Testing. After training the data and building various classifiers, the next development step is to test the data. To do this, I plan to use a stratified k-fold on the full dataset, and use each of my classifiers to predict the outcome of NBA games in a given unlabeled test set. With K-fold, each of my prediction methods will output 'k' number of predictions on randomly sampled testing sets, and an accuracy score for each fold which will be evaluated in the final development step.

1.4 Accuracy Evaluation. The fourth and final development step for this project is to evaluate the models I built. The ultimate goal is to find the classification method, and the combination of data attributes/statistics that can fit and predict the outcome of unlabeled NBA games with the highest level of accuracy. The level of accuracy in this project will be determined by the probability that the predicted output/label from the fitted classifier matches the true label of that data point in the dataset. In other words, prob(y_pred == y_true) for a given classifier. For more complex models, however, I will also calculate log-loss. To find the best inputs for my model, I will start by testing various classification methods on a constant combination of statistics. This will allow me to determine which method will fit data most accurately. Once I determine the most optimal classifier to use, I will experiment with various combinations of statistics and fit, predict, and evaluate them using the chosen model. Among these results I aim to discover which statistics play the most significant roles in predicting the outcome of an NBA game.

2.1 EVALUATION & MILESTONES

Goal:

> The goal of this project is to learn more about finding patterns in data and using them to predict the winner of NBA games based on a classifier model.

2.1.1 Evaluation of Success. The goal stated above will be a guideline for evaluating the success of this project. When it comes to evaluating machine learning models, the first bar we must surpass in order to claim any success at all is the bar of chance. If we flip a coin to determine which team will cover the spread (points given to underdog to create ~50/50

chance of each team winning) then we would expect to predict the winner correctly about 50% of the time. Therefore, our model must predict the outcome of NBA games with a mean accuracy over 50% in order to be considered successful, because otherwise you're better off flipping a coin, meaning the model is useless. Of course, getting slightly above 50% isn't the goal of this project, though. Based on the studies I have referenced, a model predicting the winner of an NBA game should have an accuracy score of around 70%. By working with various classifiers and parameters, I aim to find the best model possible given a set scope of models to implement. As more data becomes available throughout this NBA season I will continue to test my data and hopefully find an accuracy score that is consistently and substantially over 70%.

3 Completed Work (Checkpoint)

3.1 Data. Thus far I have completed all proposed data collection, cleaning and formatting for my model. I have created a python script that utilizes a BeautifulSoup web scraper on all NBA game results and season statistics going back to 1990, and exported the data into backup csv files stored locally on my machine. Data cleaning methods I have implemented include: removing unnecessary table attributes (such as arena name and attendance), removing playoff games to regularize data consistency, deleting empty and repeated rows, and discarding data columns for stats that were not tracked during all seasons in my data. For data formatting, I have created functions to change the date-time of games into a python-mutable format, calculate the number of days rest for each team in each game, and store team statistics in numpy array format to be used as inputs for my classification models.

3.2 Model Fitting and Testing. When it comes to actually building the prediction models, I have fully implemented a decision tree classifier, as well as Bagging ensemble and AdaBoost ensemble methods that use the decision tree as a weak classifier. In order to evaluate the performance of my models, I created a model evaluator class which takes a classification model as a parameter and partitions the data into to training and validation sets, fits the data with the model, predicts game results and evaluates the accuracy and elapsed time. As originally proposed, I have implemented stratified k-fold as a method of partitioning training and testing sets, which can take 'k' as an input.

3.3 Model Evaluation. The first model I decided to test was the simple decision tree classifier. To do this I started with simple team and opponent statistics for the home and away teams in each game and concatenated them into a numpy array. This array was then used as the 'X' input for training and testing my models. The input data contained 78 features including stats such as: points per game, field goal percentage, assists per game, etc. for the team and their opponents. The 'y' output for the data, or label to classify, was the winner or the game (1 if the home team wins, 0 if the visitor wins). After running my evaluator on the decision tree classifier, on all data from 2000 to 2021, the model yielded an accuracy score of 62.85% on the simple team stats input with a 10-fold partition. This score is solid, but was improved when the same data and partition was evaluated using the AdaBoost ensemble method, yielding an accuracy score of 65.89%.

4 Final Results

While an accuracy score of 65.89% was somewhat impressive given my research, there were many important features I implemented in order to discover more impactful findings. One of those features I implemented was a Random Forest Model. According to the study in reference 1, of all the models used to predict sporting events, Random Forest has the best accuracy results for predicted the winner of a sporting event. Given the success of some of my other models, I was curious to see if my model scores would be consistent with those I had researched. If so, the random forest algorithm would be the strongest

model. In order to test this, the first task was to discover which combinations of input statistics and model parameters yield the best results for all of my models. I decided to use the same features and dataset when evaluating models in order to establish consistency.

4.1 impactful statistics. In order to find the ideal number and combination of statistics I used to different approach algorithms. The first approach was a randomized brute force method in which I looped through combinations of 10 to 129 input features. The features were selected at random without replacement for each set of 'n' features. I then determined the information gained by fitting a decision tree model to each set of 'n' randomly selected features, and saved the accuracy scores in a numpy array, as well as the selected set of features that yielded the highest score. Finally, I repeated this process 200 times, saving the number of times that each feature was present in a highest scoring prediction. After running the algorithm, it was found that the two most frequent features by far were the team field goal percentages for both the home and away team. Another interesting finding was the fact that all other features had a frequency of at least 10 out of 200.

The second feature selection approached I experimented with was a genetic algorithm approach. The genetic algorithm generates an initial population of 100 sets of ten randomly selected features, calculates the decision tree accuracy of each population, and reproduces the features that yield the best results. This process was repeated 100 times and saves the 10 most impactful features as a function of accuracy score from a fitted model. Unfortunately due to large computational complexity due to my data and fitness algorithm, I was not able to efficiently find a consistent set of ideal features.

4.2 Data selection. In my original models, I had used just simple team and opponent season statistics from 2000 to 2021 as my model input. The next task I performed was experimenting with data set size and feature selection size. After experimenting with the feature selection methods described above and using them to generate input features, I determined that only my adaBoost model yielded better results with these smaller feature sets. For the other models, however, using all original features, in addition to all team advanced statistics, and the days rest statistic I calculated provided significant improvements to the prediction accuracy. Furthermore, I also found that adding the NBA data from 1990-2000 improved prediction accuracy for all of my models. To put these improvements into perspective, the accuracy score of my decision tree weak learner improved from 62.9% to 68.5% after adding the aforementioned data and features. Bagging also improved by 1.4%.

5 Conclusion

After experimenting with all of combinations of features and data points, I discovered that using all data from 1990 to present, and all season statistics and advanced statistics as input features yielded the best results for my models. The final best results for all my models were:

- 69.32% for Random Forest
- 69.2% for Bagging with 10 predictors
- 67.75% for AdaBoost

The most intriguing discovery was the fact that adding data from before 2000 improved my predictions, proving that the game of basketball hasn't change as much as I had originally thought. During the planning stage of this project, I was concerned that including data prior to the year 2000 would skew my findings due to the fact that the game of basketball has changed so drastically over the past decade or so. However, my project findings proved otherwise. As a result of these findings I have concluded that the statistical aspects of basketball that have changed the most over the years, such as increased 3 point attempts, decreased 2 point attempts, and faster pace, are relatively insignificant when compared to other statistics that have remained constant during that time. The results from my feature selection models show that field goal %, assists, days rest, and rebound percentage have consistently remained the most

impactful stats in determining the winner of an NBA game.

Another key conclusion from this project is that the complex team aspect of basketball doesn't make the game outcomes less predictable. The study from reference 1 performed various model predictions on the game of tennis which is a one player sport. I initially suspected that tennis would be far more predictable than a team sport like basketball, but yet again I was proven wrong. In the study, the majority of models predicted the outright winner of a tennis match with an accuracy between 69 and 71 percent. Meanwhile, my models were able to predict the winner of an NBA game with an accuracy around 69% proving that team sports are more predictable than I originally believed.

2.1.2 Milestones.

1. Collect NBA data and store in python program
2. 10/28/22 (COMPLETE)
3. Calculate hidden statistics
   10/28/22 (PARTIALLY COMPLETE)
4. Build simple decision tree classifier
   By 11/6/22 (COMPLETE)
5. Implement k-fold and First Ensemble method
   By 11/6/22 (COMPLETE)
6. Build the rest of the ensemble methods
   11/20/22 (COMPLETE)
7. Plot, evaluate models
   11/27/22 (COMPLETE)
8. Modify model parameters, store accuracy
9. 11/27/22 (COMPLETE)
10. Choose optimal parameters and test more
    By project due date (COMPLETE)

References

1. Wilkens, Sascha. "Sports Prediction and Betting Models in the Machine Learning Age: The Case of Tennis." *Journal of Sports Analytics*, IOS Press, 1 Jan. 2021, https://content.iospress.com/articles/journal-of-sports-analytics/jsa200463.
2. University, Yasi Zhang Fudan, et al. "Modeling and Predicting the Outcomes of NBA Basketball Games: 2021 2nd European Symposium on Software Engineering." *ACM Other Conferences*, 1 Nov. 2021, https://dl.acm.org/doi/10.1145/3501774.3501788.
3. Jones, Eric S. "Predicting Outcomes of NBA Basketball Games." | *NDSU Libraries*, Apr. 2016, https://library.ndsu.edu/.