# Predicting Player Performance in the NBA

By Johnny McGregor

## Project Goal

The NBA has embraced the use of data to evaluate players over the last decade, and the practice has become ever more popular. New advanced metrics beyond traditional statistics are used to evaluate players. While these advanced metrics have proven to be useful in predicting how valuable a player can be to a team over the course of a season, or a multi-year contract, I want to see if I can use the traditional and advanced statistics together to see how well a player might perform in a given game. Furthermore, I want to see if combining those advanced player statistics in a machine learning model with the advanced statistics available on his opponent will improve the predictions.

To quantify a player's performance, I am using a fantasy score determined by the daily fantasy sports site, Draft Kings. This score is compiled as follows:

- Each point scored is worth 1 fantasy point
- Each three-point basket made is worth an additional .5 fantasy points
- Each rebound is worth 1.25 fantasy points
- Each assist is worth 1.5 fantasy points
- Each steal is worth 2 fantasy points
- Each block is worth 2 fantasy points
- Each turnover deducts .5 fantasy points
- If a player has a double-digit amount of any two of the above stats (i.e. 10 or more points and 10 or more rebounds) it is known as a double-double and worth an extra 1.5 fantasy points
- If a player has a double-digit amount of any three above stats it is known as a triple-double and worth an extra 3 fantasy points.

There is a wrinkle in tackling this problem. When putting together a roster on Draft Kings, you have to pick at least one player at each of the five different positions, plus three additional players that play a position of your choosing. A player is assigned a dollar value between $3,000 and $10,500. You have $50,000 to spend on 8 players (an average of $6,250 a player). The task at hand isn't just predicting the fantasy output for a particular NBA player, but also finding which players are a good value for a given game.

Root mean squared error will be the metric by which we evaluate the predictions of all regression models. If the model predicts relatively well, this could be a way to "game the system" of sorts to win money on Draft Kings.

## Gathering Data

Basketball-Reference.com is a website that provides statistics on every player that has participated in at least one game. I used the python library Beautiful Soup to web scrape data from the current season as well as the five previous seasons dating back to the fall of 2014. This initial set of data totaled over 117,000 rows and had statistics on over 879 unique players. I also performed web scraping to gather the advanced statistics on every NBA team going back to the beginning of the 2014-2015 season. There are 30 teams in the NBA and each plays 82 games in a season, so each full season is a dataframe containing 2460 rows. The player and/or team features available to train a model include the fantasy statistics I mentioned above as well as the following:

- Field goals attempted (how many shots a player or team takes)
- Field goals (how many shots a player or team makes)
- Field goal percentage (% of field goals attempted that are made)
- 3-point field goals attempted
- 3-point field goal percentage
- Free throws attempted/made/percentage
- Minutes played
- Whether or not a player started a game (marked as 1 or 0)
- Whether a player or team is playing at home or away
- Plus-minus (the difference in score while a player is on the court)
- True shooting percentage (a measure of shooting efficiency that takes into account 2-point field goals, 3-point field goals, and free throws)
- Effective field goal percentage (adjusts for the fact that a 3-point field goal is worth 1 more point than a 2-point field goal)

- Total rebound percentage (an estimate of the percentage of available rebounds grabbed by a player or team)
- Total assist percentage
- Total steal percentage
- Total block percentage
- Total turnover percentage
- Usage percentage (an estimate of the percentage of a team's plays that a player was used while he was on the floor)
- Offensive rating (points produced per 100 possessions
- Defensive rating (points allowed per 100 possessions)

## Exploratory Data Analysis / Preprocessing Data

The advantage of web scraping from such a well-organized site is the absence of missing values. I wrote a line in the web scrape code not to include a game in the data if a player did not play. Therefore, there weren't any null values to search nor were there any outliers due to incorrectly entered data.

There was a fair amount of cleaning to be done in other aspects, however. Each value from the web scrape was returned as an object and needed to be converted to either an integer or a float. There were a handful of columns that had duplicate labels and others that had "/" or "%" signs in them. I also needed to create and add columns for double doubles and triple doubles, as well as the eventual target variable (fantasy points). In order to essentially automate the process of collecting the data from hundreds of individual players' web pages I had to employ some rather lengthy functions to accomplish the following:

- Gather player data from the regular and advanced statistics pages and concatenate them together.
- Clean the concatenated player data, transform some column names, and add the necessary columns mentioned above.
- Do the same thing for each opposing team's regular and advanced statistics
- Index all the data by date so that the row of data for a player's particular game can be joined with the row of stats for the opposing team.

To use this model for future games, the features will be averages of all of the relevant statistical categories up to the current date.  I wrote a function to make each row of data the average value of all the previous rows up to but not including that day's game.  This is little hard to explain, so I will just use one statistic as an example to further illustrate what I mean.

For instance, the value in the ninth row of the points column of a player's dataframe is his average points scored through the first eight games.  I also did this with a rolling five game average using the pandas rolling function, since each season is long and how a player performed earlier in the season doesn't necessarily have an impact on how well he is playing now.  To use the previous example, the value for points in a given row would be the average points scored in the previous five games.  This forced me to drop the first five rows for each player since the rolling five game average cannot be calculated until they have played at least five games.

## Modeling

During the course of modeling I fit Linear Regression, Random Forest Regressor, Ada Boost Regressor, and Gradient Boosting Regressor models, as well as Sequential neural networks with Keras.  The baseline root mean squared error (from just predicting the average fantasy points for every player in every game) was 14.62.  The three best performing models – Random Forest, Gradient Boosting, and Linear Regression - achieved root mean squared errors of 9.67, 9.60, and 9.58, respectively.  When taking the mean of all three models' predictions, I got a root mean squared error of 9.56.  The $R^2$ score on the training data with the Random Forest model was .939 while on the test data it was .563 (very overfit to the training data).  The Gradient Boosting had the smallest difference between $R^2$ scores on training and test data at .57 and .569, respectively.

At this point, I decided to see if classification models would provide more insightful results.  Instead of trying to predict a specific score, I assigned four levels to the possible fantasy score values.  Guided somewhat by running summary statistics on the target variable (with a pandas describe() method), I decided to make any fantasy point total of 50 or more tier 1. Tier 2 is between 30 and 49 points. Tier 3 is 20-29 points and tier 4 is under 20.

The baseline accuracy score to test the models against was .475 (if the model predicted all tier 4, it would be 47.5% accurate). I used a Random Forest Classifier, Gradient Boosting Classifier, and Logistic Regression – the Random Forest predicted the training data with 100% accuracy and 60.7% on the test data, while Gradient Boosting predicted with 64.8% accuracy on training data and 60.9% on test, and finally Logistic Regression was 63.2% accurate on training predictions and 60.9% on test. For what it's worth, the Random Forest and Gradient Boosted models classified almost 100 more tier 1 performances correctly compared to logistic regression, but they also had a higher amount of falsely predicted tier 1 games.

It makes sense that the tier 1 games would be the most difficult to predict since they require a player outperforming his average output. I was hoping that including statistics on the opposing team would help provide the model with some insight as to when these outsized performances are more likely to occur.

## Recommendations

For one to seriously attempt to use data to make money playing daily fantasy sports, they will have to do so using a combination of data analysis and human intuition. Incorporating the daily player salary information over time would be helpful in order to identify players who outperform their value more often. Natural language processing can help identify when a bench player is starting due to an injury to one of their teammates. In conclusion, it may not be possible for a model to predict what a player might do on a given night with high accuracy, but it can still provide someone with subject matter expertise a slight advantage over the rest of the crowd.