Capstone Project

Machine Learning Engineer Nanodegree

David Quinn

April 24, 2017

Definition

Project Overview

In the US, the National Basketball Association (NBA) is one of the four major sports. There are many statistics recorded in a basketball game, both offensive and defensive. Further, many other metrics can be computed from the recorded stats. For instance, points are recorded as well as minutes played, so we can create points per minute.

Additionally, fantasy sports, and more specifically, daily fantasy sports have become very popular as of late. In daily fantasy sports, a user selects about 8 basketball players that are playing that night. As many as hundreds of thousands of users pick a team on any given night. The entry fee to this daily tournament can range from $1 to $100. Each player is rewarded a fantasy point value for statistics recorded and the user that achieves the most fantasy points wins. In some tournaments, the users with the top 10% highest scores win some cash back.

Therefore, if I can predict how many fantasy points a player will score on a given night. Combining this prediction with a method to optimize lineups for daily fantasy contests could be lucrative. However, this prediction needs to be better than simple strategies such as the average fantasy points for each player.

The website basketball-reference.com has historical data for every NBA player going back for decades. For this project, I have downloaded the last 3 years of gamelog data for each player. I downloaded full seasons 2013-14, 2014-15 and 2015-16, which comprise of 82 potential games for each player (sometimes there are injuries) and about 480 active players in the season. Therefore, there are about 82 x 480 = 39,360 potential gamelogs per season. However, given that there are injuries the number of actual gamelogs will be lower. Further, many players don't actually play in a given night or play very infrequently. These players are likely not important as they won't be selected in the daily fantasy contests. Therefore, for 3 seasons, my guess is that there are about 10,000 gamelogs per season that would be interesting to investigate.

Problem Statement

In this project, I attempt to predict the fantasy points for each player in the 2016-17 NBA season by building regression models based on statistics recorded by each player in previous seasons. I am going to investigate different regression models in scikit-learn to determine for each player what the ideal model is. Subsequently, I will test these models out of sample in the 2016-17 NBA season to determine if the model is better than simpler prediction methods. The current season (2016-17) is only halfway complete to date. I have downloaded data through 26-Jan-2017, about 40 games. Therefore, I will train my models based on previous seasons and test the model on the current season.

Sports, and fantasy sports in particular, are very popular in the US. Therefore, for my regression models, I am going to consider data that is common as well as less frequently used, including data that I compute myself. The rationale for this is that regression prediction has likely taken place already on these data sets. Hence, different data may lend curious insights.

I anticipate that there is not a single model that can predict each player's fantasy score each night perfectly. Since each player is very different – some more consistent, some with a hot hand, some who play better at home and some who play better against certain opponents – I anticipate that each player will have a different ideal model. Therefore, I will consider various models taking all players into account and player dependent models if the computation time is not too onerous.

Metrics

To evaluate the efficacy of the various regression models, I will be using Mean Squared Error (MSE) to evaluate how well the model is predicting the out of sample data. MSE was used as it easily compares all models against one another and is interpretable. The square root of MSE gives the predicted vs. actual value error on average. For instance, if the MSE is 100, then the average prediction error is around 10 fantasy points.

One of the issues with MSE is that it can increase significantly when outliers are present. Given that the error term is squared, large errors create large MSE values. However, these errors would be concerning to me. Besides the occasional mid-game injury, our predictions shouldn't lead to huge errors. A player may have a great game and we predicted an average game, this would be a bad outcome as the goal is to predict the great performances! Therefore, penalizing large errors by squaring them is a good approach.

Analysis

Data Exploration

To make predictions for fantasy points scored in a given evening, it's important to understand how fantasy scores are computed. Various websites have different computations for their contests; however, DraftKings is very popular and uses the following rubric:

- Point = +1 PT
- Made 3pt Shot = +0.5 PTs
- Rebound = +1.25 PTs
- Assist = +1.5 PTs
- Steal = +2 PTs
- Block = +2 PTs
- Turnover = -0.5 PTs
- Double-Double = +1.5 PTs
- Triple-Double = +3 PTs

Therefore, all these statistics are relevant when predicting fantasy scores. In addition, other qualitative variables may be relevant, such as location of game (home or away), opponent quality, player rest before game and minutes played. In fact, minutes played seem to be very significant in prediction fantasy output: http://www.quantifan.com/post/106972747368/projecting-nba-fantasy-points-per-game

Obviously, we can't use the statistics recorded from the game in question to predict fantasy scores as we don't know this information ahead of time. Therefore, we need to create some estimate of what these values may be. In order to do this, I am going to use a rolling average of the past 3 and 10 games. Some features do not require rolling averages, such as location and rest between games.

The NBA season runs from October to April, so it spans two years. For the training data, we have gamelogs from three NBA seasons, 2013-14, 2014-15, 2015-16. I estimated in the 'Project Overview' that there would be about 10,000 gamelogs per season that would be interesting to investigate. Let's look at the data to validate.

Summary of training data 2013-2016:

```
number of training gamelogs: 55162
max fantasy in a game: 93.5
min fantasy in a game: 0
average fantasy in a game: 22.491
std dev fantasy in a game: 13.726
highest fantasy average over last 10 games: 69.0
number of games with fantasy points = 0: 958
games above fantasy average: 25347
```

The number of "clean" gamelogs is 55,162 (more info on the cleaning process in the 'Preprocessing' section below), which is almost double my estimated guess of 10,000 per season (30,000 total). The mean fantasy points scored is around 22 per game with a standard deviation of roughly 14. Furthermore, almost 1,000 of these samples or roughly 2% had a fantasy value of 0. This sample only includes gamelogs for players that played on that evening.

Next, let's investigate what an average level is for the various stats: points, rebounds, assists, steals and blocks. For the training set, we can take a simple average across all players and then determine what the average composition each stat has to fantasy points.

Average Statistics Across All Players, 2013-2016:

```
Pts/game average for all players: 10.513
Reb/game average for all players: 4.374
Ast/game average for all players: 2.259
Stl/game average for all players: 0.782
Blk/game average for all players: 0.489
```

If we only consider these statistics, then the average fantasy score should be close to the averages above multiplied by the aforementioned DraftKings rubric. Therefore, the average fantasy score based on this method is:
10.513*1 + 4.374*1.25 + 2.259*1.5 + 0.782*2 + 0.489*2 = 21.911

This figure is very close to the actual average from the dataset, which suggests that the added value of a 3-point shot made is negated by the detraction of turnovers. As a student of the game, this makes sense. Those that make more 3 point shots, likely have more turnovers as they tend to play the position of Guard, whom handle the ball more often and are likely to turn it over (yes, even Steph Curry).

Contribution:
Points: 10.513*1 / 21.911 = 47.98%
Rebounds: 4.374*1.25 / 21.911 = 24.95%
Assists: 2.259*1.5 / 21.911 = 15.46%
Steals: 0.782*2 / 21.911= 7.13%
Blocks: 0.489*2 / 21.911= 4.46%

Therefore, the average fantasy score is mostly derived from how many points a player scores in a given night, followed by how many rebounds they get and then how many assists, steals and blocks. It is therefore, most important to have an accurate estimation for points and rebounds in a given night.

As a rough estimate, I can assume that tonight's statistics are similar to the rolling averages of the previous 3 and 10-day statistics. Further, this similarity should be born out when looking at correlations of the rolling average statistics and fantasy points scored.

Let's take a closer look at the rolling average of features and see how they correlate to fantasy points scored:

Correlation with Fantasy Points:

location	0.028328	avg10pts	0.655792
avg10fantasy	0.707328	avg10plus_minus	0.168233
avg10ast	0.474218	avg10trb	0.458727
avg10stl	0.429628	avg10blk	0.255566
avg10fga	0.640972	avg10p3p	0.112987
avg10mp	0.618161	avg3pts	0.613882
avg3fantasy	0.672831	avg3plus_minus	0.113682
avg3ast	0.451748	avg3trb	0.432383
avg3stl	0.340002	avg3blk	0.218268
avg3fga	0.619287	avg3p3p	0.096356
avg3mp	0.602903	rest	-0.087989
avg 10opp_sum_fantasy	-0.013210		

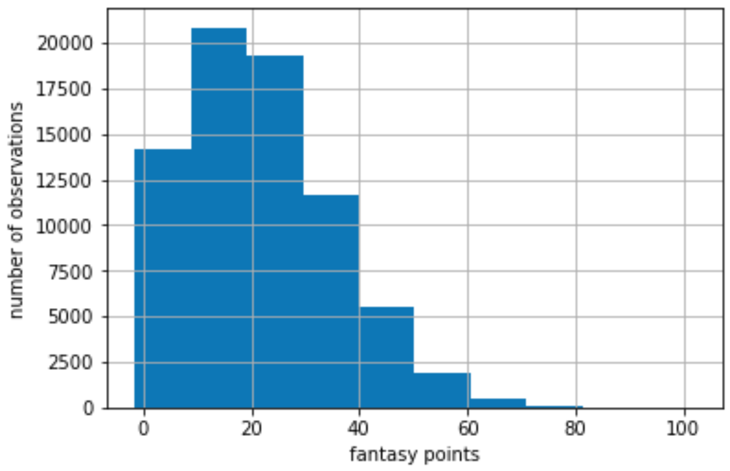
The first aspect to notice about these correlations is that the eight strongest correlations to fantasy points are all related to the rolling averages of fantasy points, points, field goals attempted and minutes played. This is somewhat expected after reading the aforementioned article on minutes played having an impact on fantasy production. Further, players tend to be consistent so fantasy points appear to have some autocorrelation.

Additionally, points correlate strongly with fantasy points as they make up a large contribution to the fantasy points, as demonstrated above. Given this, it appears that there is some autocorrelation in points as well.

After some research, I found the following Stanford CS 229 research paper conducting similar analysis to mine. One of the findings is that field goals attempted were very informative in predicting fantasy scores. This result is consistent with the analysis conducted on minutes played. Just being on the court (high minutes) correlates well with higher fantasy scores. Similarly it seems with shooting, it doesn't matter how *well* you shoot, it matters how *much* you shoot. Given that the fantasy score doesn't penalize missed shots, this makes sense. Although, this lesson isn't necessarily great for winning real basketball games!

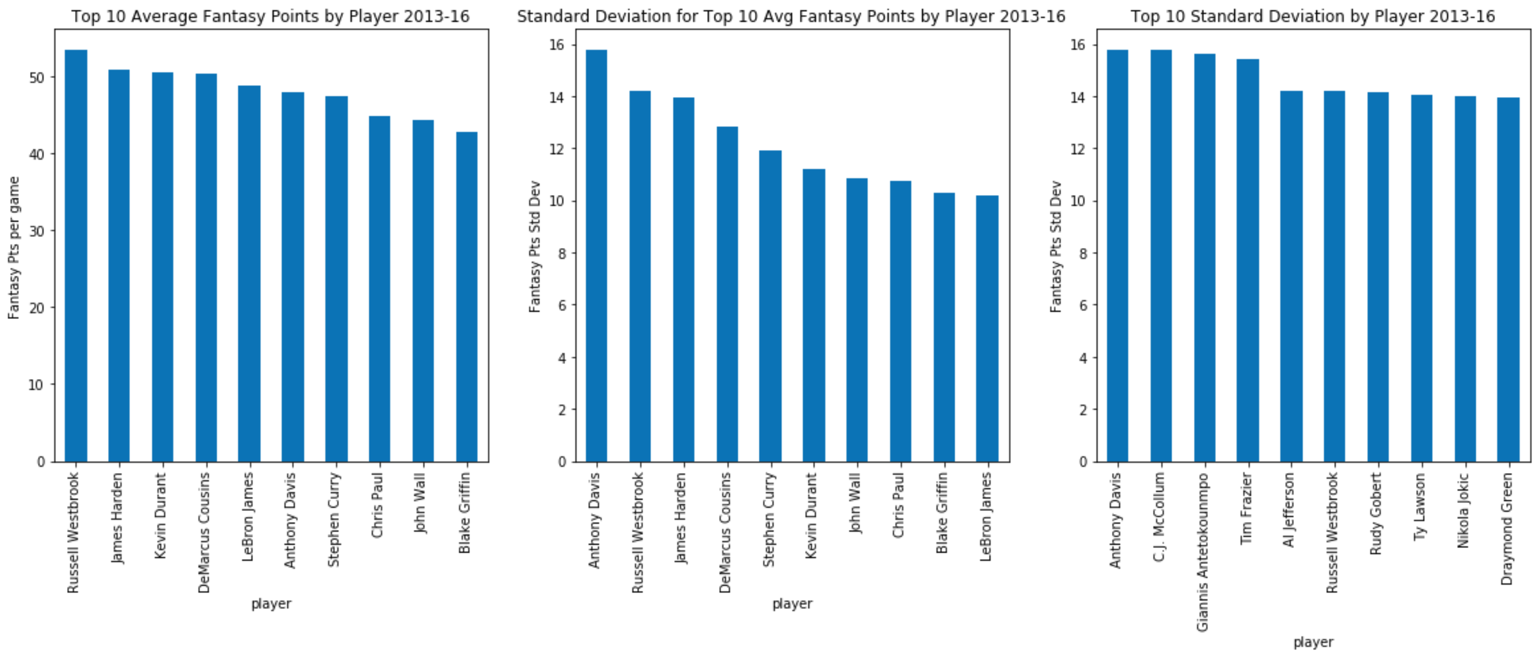
Exploratory Visualization

To better understand the dataset, let's take a look at some visualizations of certain statistics. First, the distribution of fantasy scores per game for the training set:



Clearly from the histogram above, the distribution of fantasy points per game is right skewed (mean is farther right than centered). There is a longer tail to the right than there is towards the left. This makes sense given there are a few very great players in the NBA and a large number of average players (much, much better than I, but average in a relative sense).

Since I want to ultimately win daily fantasy competitions, I'd like to see how the elite basketball players perform. First, let's see who the best players are when it comes to fantasy points and how their fantasy points change:



Over the training set, Russell Westbrook averaged the most fantasy points per game, followed by other well known players: James Harden, Kevin Durant, DeMarcus Cousins, LeBron James, Anthony Davis, Stephen Curry, Chris Paul, John Wall and Blake Griffin. As a side note, only two of these players have won championships

(James and Curry), suggesting that fantasy points may not be the path towards victory on the court. The average fantasy points scored for the top 10 players is around 45 to 50 points per game. However, there is significant variation around each of their means. For instance, Anthony Davis had the largest standard deviation of the group (and of the entire NBA, far right chart above) of around 16. The coefficient of variation for Davis (std dev / mean) was 33% where it was 21% for LeBron James.

Algorithms & Techniques

There are four algorithms that I intend to use for the project. Given that it is a regression problem, I am going to focus on regression algorithms. The first two are well suited for large amounts of data as the processing time is quick: Decision Trees and Stochastic Gradient Descent. The next two algorithms are a bit more computationally expensive, but they are comprehensive additions for this exercise, Gradient Boosting Regression and MLP Regression.

Also, I am going to fit those same models using only training data with fantasy scores above the average fantasy scores to determine if different features are relevant when looking at the top NBA players. I will then test all eight models on the out of sample data from the 2016-17 NBA season.

Benchmark

In order to understand if our models are even worthwhile, we need to compare them to a benchmark. There are two benchmarks I am going to use for prediction of fantasy points, the rolling 3-day average of fantasy points and the rolling 10-day average of fantasy points. A simple model would suggest that tonight's fantasy points would be what we have observed in the past. This is computationally easy and intuitive. I will compare my models to this benchmark using the MSE metric as previously mentioned.

Methodology

Data Preprocessing

I need to create the rolling 3 and 10-day averages for the relevant statistics. Given the necessity to do this a number of times, I created a function to do so:

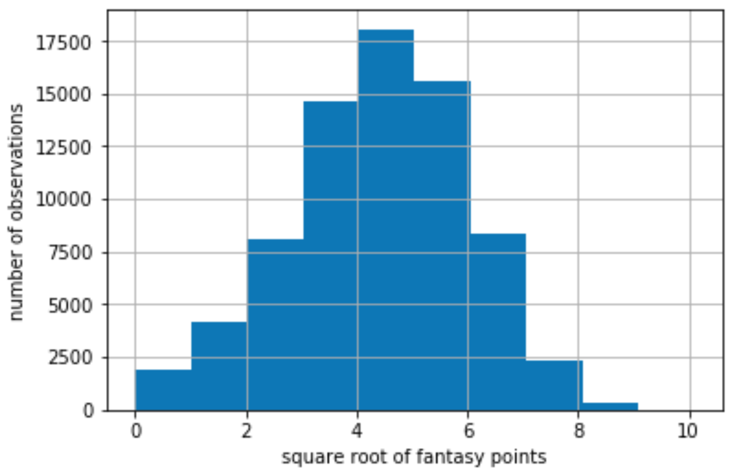
```
def add_avg(dataset, stat, roll):
    dataset['avg'+str(roll)+stat]=dataset.groupby('player')[stat].rolling(window=roll).mean().shift(1).values
```

Next, I wanted to create a measure of how good the opposing team is. Therefore, I created a variable to represent the 10-day average of how many fantasy points each team has given up. If an opposing team yields many fantasy points, then they are not very good.

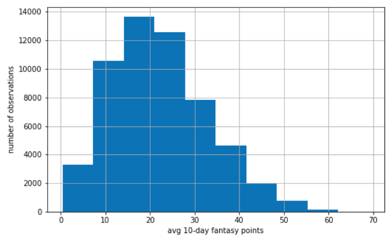
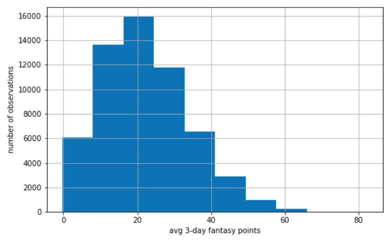
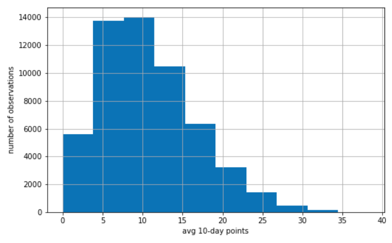
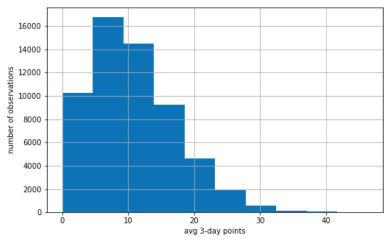
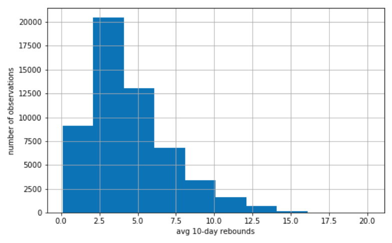
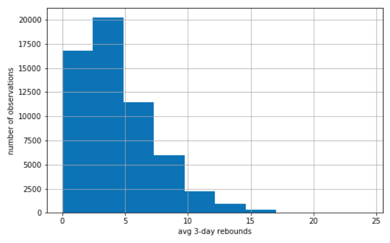
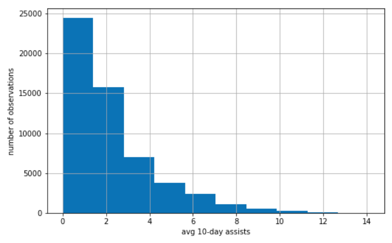
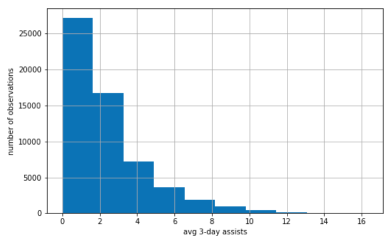
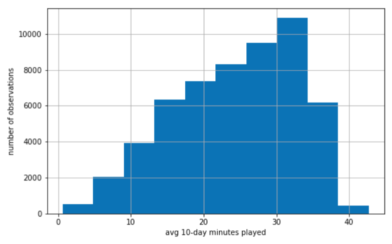
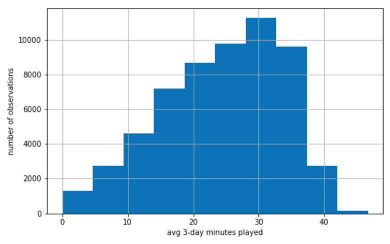
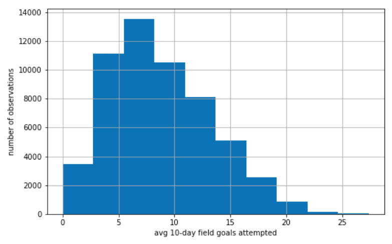
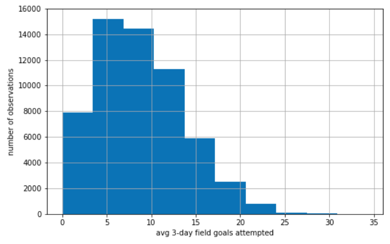
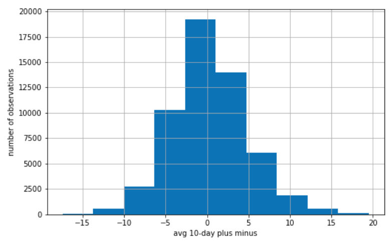
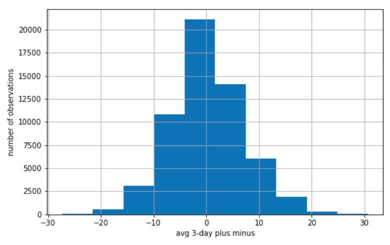
Lastly, I created a location dummy variable for home and away to determine if players score more fantasy points based on venue, as some research has suggested.

As noted in the 'Exploratory Visualization' section, the distribution of fantasy points is right skewed. Therefore, in processing the data it makes sense to transform it to a more normal distribution before running any models. When it comes to prediction, we can transform it back again.

I will use a power law transformation to convert the fantasy points to a more normal distribution. By raising the fantasy points scored to the 0.5 (analogous to taking the square root), the distribution becomes more normal:

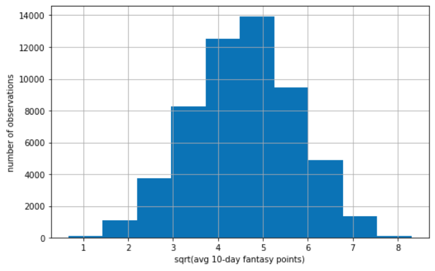


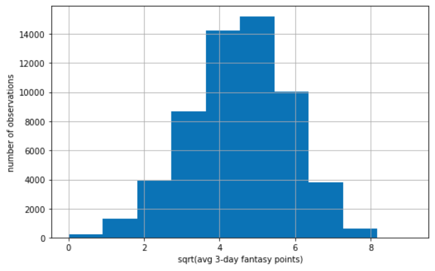
Now I can run the models against the transformed fantasy points variable. Given that the dependent variable was not normally distributed, I will examine the distributions of the features as well:

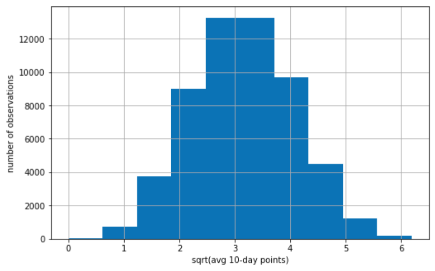

number of observations
avg 10-day fantasy points

number of observations
avg 3-day fantasy points

number of observations
avg 10-day points

number of observations
avg 3-day points

number of observations
avg 10-day rebounds

number of observations
avg 3-day rebounds

number of observations
avg 10-day assists

number of observations
avg 3-day assists

number of observations
avg 10-day minutes played

number of observations
avg 3-day minutes played

number of observations
avg 10-day field goals attempted

number of observations
avg 3-day field goals attempted

number of observations
avg 10-day plus minus

number of observations
avg 3-day plus minus

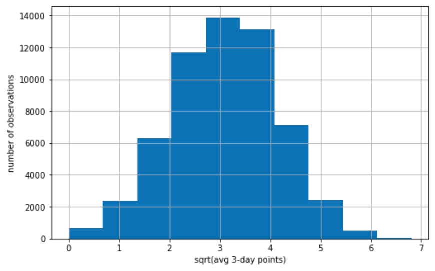
A number of the features also display right skewness, therefore, I will transform the following variables: fantasy points 3 and 10-day, points 3 and 10-day, rebounds 3 and 10-day and assists 3 and 10-day. While it is not an issue to run regressions with skewed features, I wanted to explore the impact it makes.

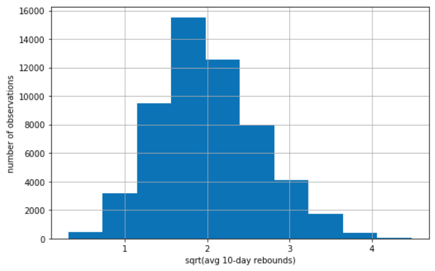
Here are the resulting distributions of the features after transformation:

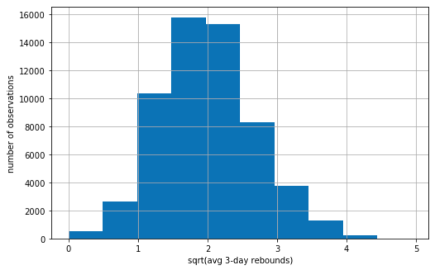


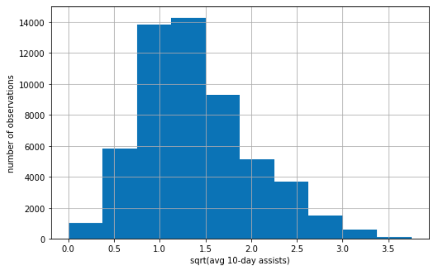


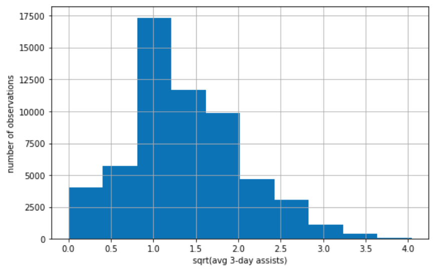


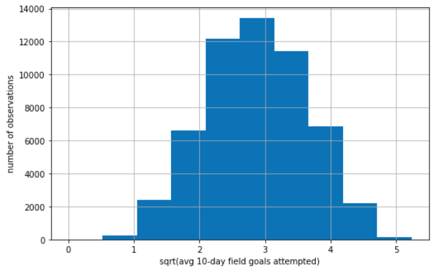


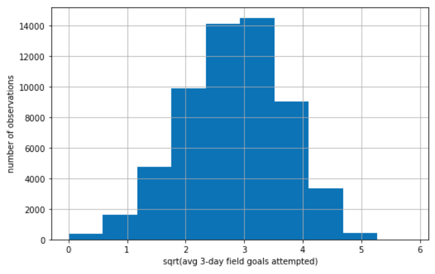








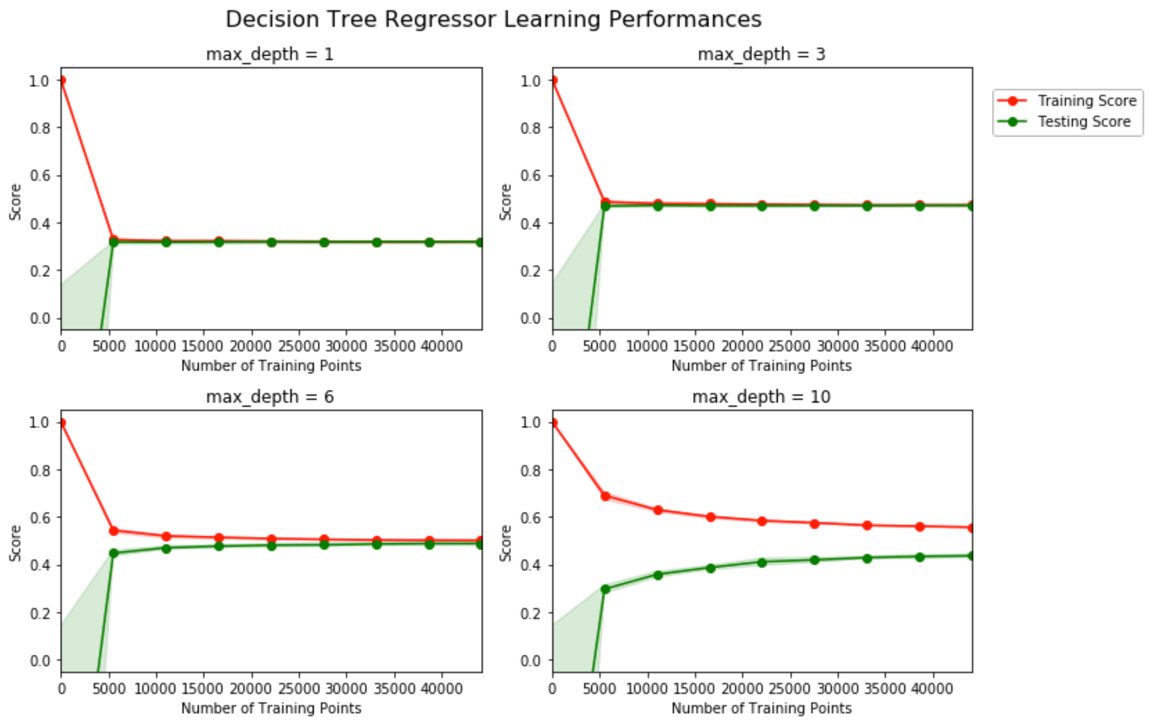


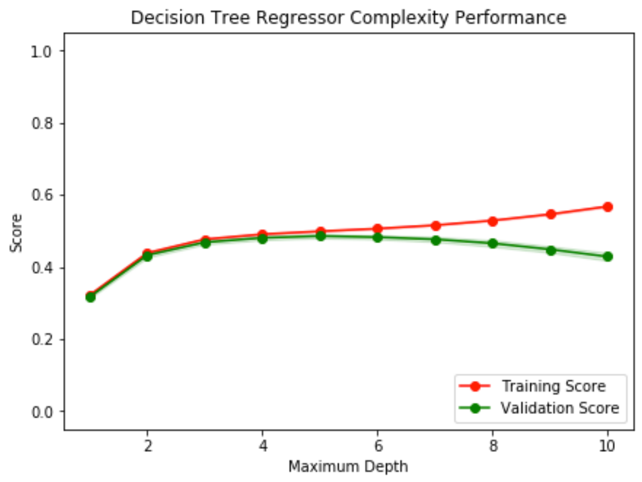


Implementation

Now that the data is processed and ready to be examined, I can begin testing models. First, I ran the training data through cross validation to split it into training and testing sets.

Once the data is split into training and testing sets (within the in-sample data, 2013-2016), I examined the appropriate depth for the Decision Tree model by using the visuals.py code from Project 1. I am using the R^2 metric to evaluate the score for the training and testing data.





Visually, depths of around 4-6 seem to do best for the Decision Tree model. The rationale for this is because the training and testing scores seem to converge at these depths as we add more training points. This observation indicates that we are not over-fitting the training data. Furthermore, when looking at complexity depths around 4-6 have similar training and testing performance, again, suggesting that are not over-fitting.

To check my estimate on ideal depth, we can use grid search when fitting the models. I fit all four aforementioned models on the training data for all players using grid search to determine the best parameters for each model. For the Decision Tree model, I searched across depths from 1 to 10. For the SGD model I searched across loss functions of squared loss, Huber and epsilon insensitive. For the Boosting model I searched across depths from 1 to 5. Lastly, for the MLP model I searched across activation functions of logistic (sigmoid) and rectified linear unit (relu) and learning rates of constant and inverse scaling. Again, when evaluating the efficacy of each model, I am using the R^2 metric for comparison.

The results from grid search for the decision tree model match my visual guess. The optimal parameter for max depth was 5, while my guess was 4-6. The remaining parameters are below:

Results from Grid Search

```
Parameter 'max_depth' is 5 for the optimal Decision Tree model.
Parameter 'loss' is huber for the optimal SGD model.
Parameter 'max_depth' is 3 for the optimal Boosting model.
Parameter 'activation' is logistic for the optimal MLP model.
Parameter 'learning rate' is constant for the optimal MLP model.
```

Refinement

After finding the ideal parameters for each model and input data, I examined the R^2 score from a train and predict process on the cross validation data set. The R^2 scores for the models are below:

Results from training and predicting four models on cross validation set:

```
Training a SGDRegressor using a training set size of 44124
R2 score for training set: 0.1981.
R2 score for test set: 0.1872.

Training a GradientBoostingRegressor using a training set size of 44
124
R2 score for training set: 0.5138.
R2 score for test set: 0.4905.

Training a DecisionTreeRegressor using a training set size of 44124
R2 score for training set: 0.4970.
R2 score for test set: 0.4785.
```

```
Training a MLPRegressor using a training set size of 44124
R2 score for training set: 0.5000.
R2 score for test set: 0.4849.
```

The 3 models: Boosting, Decision Tree and MLP all performed similarly with an R^2 in training and testing around 0.5. The SGD regressor was very quick, however, the R^2 on training and testing was weak.

The first refinement I made was to explore the models not transforming the feature set. The fantasy score will still be transformed. The results from training and predicting using the untransformed feature set are:

Results from training and predicting four models on cross validation set:

```
Training a SGDRegressor using a training set size of 44129
R2 score for training set: 0.1774.
R2 score for test set: 0.1875.

Training a GradientBoostingRegressor using a training set size of 44
129
R2 score for training set: 0.5108
R2 score for test set: 0.5057.

Training a DecisionTreeRegressor using a training set size of 44129
R2 score for training set: 0.4936.
R2 score for test set: 0.4932.

Training a MLPRegressor using a training set size of 44129
R2 score for training set: 0.4936.
R2 score for test set: 0.4986.
```

The out of sample results for the refined model, where features are not transformed but the fantasy score is, perform better than the first model. The R^2 are similar, but the testing performance is slightly better when the features are not transformed. These results are interesting as I thought transforming the data to a more normal distribution would improve performance.

The second refinement was to consider the validity of transforming the dependent variable, fantasy points. Given that the results above are a bit better without transformation, I should consider not transforming the dependent variable. The results from training and predicting using the untransformed feature set and fantasy variable are:

Results from training and predicting four models on cross validation set:

```
Training a SGDRegressor using a training set size of 44129
R2 score for training set: -0.0730.
R2 score for test set: -0.0509.

Training a GradientBoostingRegressor using a training set size of 44
129
R2 score for training set: 0.5230.
```

```
R2 score for test set: 0.5177.

Training a DecisionTreeRegressor using a training set size of 44129
R2 score for training set: 0.5059.
R2 score for test set: 0.5084.

Training a MLPRegressor using a training set size of 44129
R2 score for training set: 0.5032.
R2 score for test set: 0.5114.
```

Results

Model Evaluation & Valuation

The best model from the cross validation testing was a Boosting regression with no transformations on dependent variable or the features. This model was selected based on the best cross validation R^2 score for the testing data 2013-2016. The feature importances from this model are:

```
location: 0.0212
avg10pts: 0.0302                    avg3pts: 0.0065
avg10fantasy: 0.1685                avg3fantasy: 0.0956
avg10plus_minus: 0.0300             avg3plus_minus: 0.0465
avg10ast: 0.0393                    avg3ast: 0.0168
avg10trb: 0.0383                    avg3trb: 0.0297
avg10stl: 0.0238                    avg3stl: 0.0100
avg10blk: 0.0261                    avg3blk: 0.0072
avg10fga: 0.0430                    avg3fga: 0.0716
avg10p3p: 0.0689                    avg3p3p: 0.0201
avg10mp: 0.0509                     avg3mp: 0.0684
avg 10opp_sum_fantasy: 0.0593       rest: 0.02815
```

The final parameters of the model above make a lot of sense. As expected, the trailing 3-day fantasy average and trailing 10-day fantasy average are the most important features. Since these features also displayed the highest correlation to futures fantasy scores, this makes sense. Subsequently, the trailing 3-day field goals attempted average and trailing 3-day minutes played average are the second most important. These also make sense from correlations and analysis above. The longer a player is on the court and the more shots he heaves, the more fantasy points he will score.

To test the validity of the model further, I will examine how well the model generalizes to 2017 data, which the model was not tested on. This out of sample consists of player data for the first 40 games of the 2016-17 season. If the model does well, as defined as outperforming the average 10-day trailing fantasy score, then I can assume that the model generalizes.

First, the model explains a good amount of variation in fantasy scores in the out of sample data. As evidenced by the R^2 score, the Boosting regression model has an R^2 well over 0.50 which outperforms the results from the training R^2 score:

```
The R2 for the best model regression: 0.5776
```

Further, the model does quite well on an MSE and MAE basis out of sample. As compared to the benchmark, the model outperforms:

```
Here is what I am trying to beat:
The MSE for all players, using past 10 game fantasy average: 85.428
The MSE for all players, using past 3 game fantasy average: 99.903

The MAE for all players, using past 10 game fantasy average: 7.228
The MAE for all players, using past 3 game fantasy average: 7.784

Here is how our model performed:
The MSE for the best model regression: 84.040
The MAE for the best model regression: 7.249
```

Next, I wanted to see if Boosting regression models trained on each player would outperform the single model trained on many players. If so, then each player may have unique experiences and these models would not benefit from how other players fantasy scores attribute.

Further, I wanted to check this on the top 50 and 100 players. Getting these predictions right is most important, as they are the highly sought after players and can really impact your total fantasy score on any given night. The results from this test are interesting; the average MSE from the models trained on each player was quite poor relative to the single model trained on all players:

```
MSE from Boosting regressions by player for top 50 players: 200.102
MSE from rolling 3-day average fantasy for top 50 players: 150.490
MSE from rolling 10-day average fantasy for top 50 players: 125.595

MSE from single Boosting regression for top 50 players: 125.298
MSE from rolling 3-day average fantasy for top 50 players: 150.490
MSE from rolling 10-day average fantasy for top 50 players: 125.595

MSE from single Boosting regression for top 100 players: 113.192
MSE from rolling 3-day average fantasy for top 100 players: 136.505
MSE from rolling 10-day average fantasy for top 100 players: 114.706
```

The single boosting regression trained on all players does the best for the top 50 and top 100 players. Additionally, the model trained on each player underperforms the single Boosting regression, the rolling 10-game average and the rolling 3-game average, which is quite a poor result.

Justification

The model performs essentially the same as using the 10-game rolling average for fantasy points. It far outperforms the 3-game rolling average, which suggests that performance over the short-term is quite variable. Players tend to gravitate towards the longer-run mean over time.

The model didn't blow the 10-game rolling average out of the water but I think it is still justified. On an out-of-sample dataset, the Boosting model continued to match or outperform the benchmark. Therefore, the model was not overfit on the training data. Further, the feature importances make sense given my understanding of the game of basketball and fantasy sports.

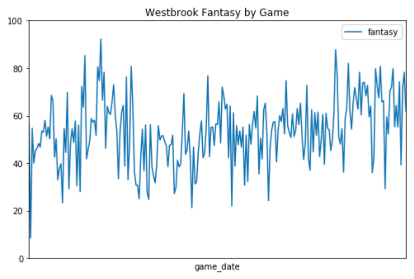
An important insight from the section above is that the model works best when trained using data from all players, and then that model applied to *each* player. When investigating training many different models using data exclusively for each player, the model did not work as well. This makes sense in that some players may encounter situations that other players have encountered previously. Potentially, the model trained using all players is more robust due to this fact that players have similar fantasy experiences but at different times.

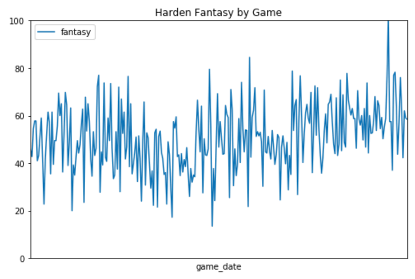
Conclusion

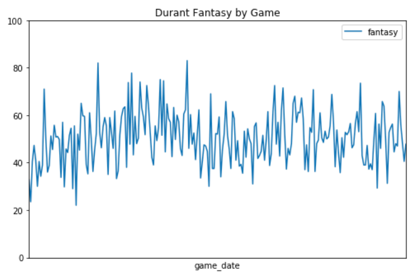
Free-Form Visualization

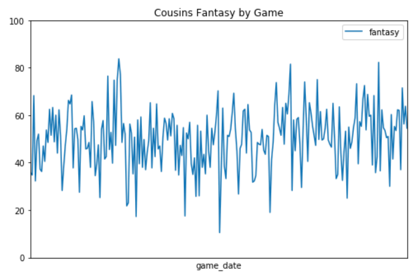
Consistently, the results from the 3-game rolling average of fantasy points underperformed the results from the 10-game rolling average of fantasy points. This is a curious result as much of the fantasy literature suggests that players 'get hot' and you should spend extra 'fantasy money' on players that have played well. However, from the MSE results above, players are more likely to perform at the 10-game average than their 3-game average.

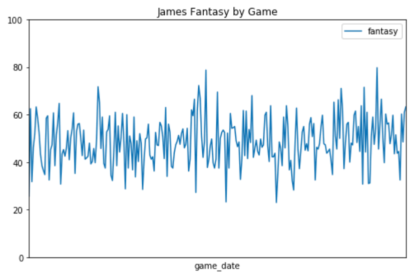
To investigate this, I created charts for the top players documenting their fantasy score by game:

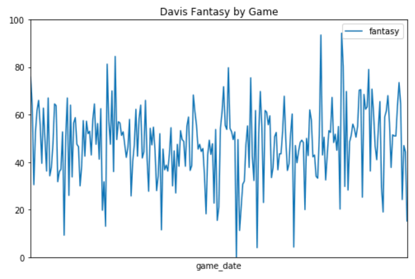












From the charts above, we can see that the variation for certain players is much larger than for others. Players like Anthony Davis and Russell Westbrook have more variation than players like Kevin Durant and LeBron James. While my best model outperformed a simple model of the 10-game rolling average, it didn't blow it out of the water. Therefore, understanding more about how to predict the spikes and valleys in the daily fantasy score would be very beneficial.

Reflection

I started this project with a goal of creating a machine-learning model that does a good job in predicting the daily fantasy scores of top NBA basketball players. With the intention that if the results were strong, this model could be used to compete in daily fantasy sports competitions. My process started by collecting stats for every player over the last 3.5 years. I transformed this data such that the model did not have a look-ahead bias. I then tested 4 different models on the data collected over the last 3 years. The model and model parameters that did best over this training period I used to investigate the out-of-sample data from the last half year. This model was a Boosting regression and it outperformed the benchmark I had set (the rolling 10-game fantasy average) on an MSE basis for all players, the top 50 players and the top 100 players.

Two important insights that I learned in this process are that the longer term (10 games) matters much more than the shorter term (3 games) and that use-based stats matter much more than efficiency-based stats. First, being able to predict when spikes occur is difficult and the 3-game rolling average does not help do this. Players don't appear to get hot and continue playing at a high level. Players most often play at their long-term level. Therefore, you may be able to extract some value if you discount a hot player and another person in your league is interested in them.

Secondly, for fantasy leagues, it is much better to have players that are in the game, get minutes and take shots than it is to have players that are 'smart' with the ball. Consistently I found that players that are just on the court a lot (high minutes played) and take a lot of shots (field goals attempted) is much more predictive of higher fantasy scores than players that have a higher field goal percentage. However, this does not lead to a good basketball player when it comes to winning games, efficiency and care with the ball is important. Hence why it may be likely that only two of the top 10 fantasy players have won actual championships.

Improvement

While the model outperformed the benchmark I set, there is still progress I can make. I don't think the model can necessarily be improved, but I think the feature set can be more robust. In order to predict fantasy scores tonight, you need to predict how many stats a player will put up tonight. To do this, I thought the 10-game rolling average and the 3-game rolling average for a number of stats would be helpful. From here, I ended up with a Boosting regression using each one of these averages to predict how well a player will perform. Given that the use-based stats had stronger correlations to fantasy scores, I think the model could be improved by including more features that relate to how much a player is used.

Stats that incorporate how good the opponent is – if a opposing team is very bad the best players may get rested. Stats that incorporate the time of the season – at the end of the season the best players may be on a minute restriction. Stats that incorporate how much time the player has the ball – some players play a lot of minutes but don't touch the ball much.

Further, I discovered that the spikes are very hard to predict. Therefore, it is worth exploring other features that may aid in the time-varying nature of fantasy scores. Do individual matchups matter a lot? What about time of day games? What about how much travel each team had to make? There are a number of features that I can explore to attempt to model the volatility better.

Now that I feel comfortable about using a Boosting regression model, I can focus my efforts on the best feature set to use. Collecting and cleaning these features is time-consuming but I don't have to run this data through a number of parameter selections, which takes a lot of time. I can continue to improve the model over the next months and see how it performs during the 2017-18 NBA season.