**The MVP Money Model**

By Trevor Klingensmith

# Abstract

The sports world is no stranger to data analysis. In the past 30 years data collected about professional sports has increased exponentially, and the amount of money around it has grown just as much. From negotiating contracts, to coaches’ gameplans, to fantasy sports, data is more prevalent in the world of sports than ever before. One interesting way that technology has changed the world of sports is through online sports betting. Millions of dollars are won and lost every day due to sports betting, and unfortunately most people end up on the losing end of that over a long enough sample size. Through this project I aim to answer the question: Is it possible to gain an edge over the bookies by analyzing large amounts of NBA data and creating a model to predict who the next MVP will? This topic is interesting to me because I love basketball and money, so the two go hand in hand. The resulting model could be used by myself, and perhaps for the right price by some of my friends to get some inside knowledge on the most likely picks.

For everything one can bet on, the bookkeeper sets odds for that bet, a number describing how much the bet would pay out if correctly bet on. Odds of +110 mean that if one places a bet for 100 dollars on the Bulls, and the Bulls win, the total payout they will receive is 210 dollars ($100 original bet + $110 profit). These odds are set by the bookkeeper and change all the time based on how likely they predict that outcome is. The less likely an outcome, the more favorable the odds will be for the bettor. The reason that I am creating a model to predict who the MVP will be is because the odds on bets like these typically have very favorable odds. There are endless possibilities for any individual player’s circumstances to change. especially early in or before a season. Players go from dominant to sub-par, and unknown to unstoppable all the time. Even more impossible to predict are injuries that can instantaneously take a player out of contention. Knowing this, bookkeepers typically give all MVP bets very good odds. I plan to build a model that will take in inputs such as points, assists, rebounds, and various other data points and will output its prediction as to whether or not the player’s given statistics will be an MVP.

# Introduction

The dataset that was used for this model was taken from basketball-reference.com, a well-known repository of pretty much all official statistics published in NBA history. Various statistics began being collected at different times. For example, points have been recorded since the 50s, however more advanced statistics like stats on every possession rather than just the whole game have only been recorded since 1974. There are many ways that NBA statistics can be evaluated, and it is always a goal to get players to be on an even playing field to compare them accurately. Some players may have gotten injured in their prime, some may have gotten drafted straight from high school versus other players being in college for four years. With that in mind, I elected to use a measure of player stats that is on a scale that is commonly used to compare players in an unbiased way. “Per 100 Possessions” stats take all of a players statistics from every possession of a season and average simply divide each category by 100 to get a digestible look at their productivity. After cleaning this dataset, the methods of which will be explained in the next section, I appended another column that is simply a boolean value for whether a player was the MVP during that season or not. That being the case, nearly every row has a value of false in that field except for the one MVP of that season. Following that, I used standard measures of player performance: points, rebounds, assists, blocks, steals, minutes, and age as predictors of the category MVP, and ran models to predict if a given player would be the MVP of that season.

# Methodology

The first step in making this happen was uploading and cleaning the data. To begin, I loaded in the “Player Award Shares” csv file, a dataframe with all of the player awards from each season (Defensive Player of the Year, Sixth Man of the Year, Rookie of the Year, Most Improved Player, and Most Valuable Player). This dataframe included the top five players for that award (it uses a point system based on votes that is irrelevant to this paper) as well as a boolean value column indicating if the player won that award that season. For this purpose, I only cared about the MVP category, so I dropped every column except the player’s name, player\_id, the season, and whether they won. Because the only column with meaningful data is a boolean that is stating whether a player was the MVP of the season for that row, there is no cleaning that should be done. I definitely want unadulterated values in this field. With the MVP field settled, I could move on to the predictors and come back to this later.

Graphical user interface

Description automatically generated with medium confidence

After opening the “Per 100 Poss” file in Jupyter and doing a .head function to look at the columns, I found that it was a very wide dataset. Before getting to the predictors I wanted to use, I started by getting rid of the most glaring issues, which was the data from the current season that is incomplete and therefore has no MVP. Graphical user interface, text, application, email

Description automatically generated

Now that we were left with only complete seasons I moved on to really cleaning up the data. Any statistic of a player’s performance in a season is in some way going to be a predictor of them being the MVP that year. However, it’s humans that make that choice, and humans tend to act on emotion and just look at the major categories when evaluating a player. Because of this, I did not want to overly complicate my model such that it had high variance sensitive to the input data and was not predictive for new data. Humans are not computers, so using a large amount of parameters to find patterns that no human would see would not make sense for this purpose. As a result, I dropped 24 columns, leaving us with the main stat categories of a players performance: points, rebound, assists, steals, blocks, minutes, and age.

One of the choices I made for the data and my model is that I did not discretize any values. While some algorithms can benefit from discretized values, in this case I felt that it would take away from the true story the data is telling. For example, for a category like steals or blocks the numbers are very low for the vast majority of players. Putting players in buckets would over generalize in this case, with very few players in the high number bucket and the overwhelming majority of players falling in the low number bucket. With a response variable that is true in so few instances relative to the total number of entries, I felt that any sort of generalization would lead to my model having very little ability to accurately extrapolate.

The next step was to find and address missing data, which this dataset had very little of. There were exactly five missing values on five columns: points, rebounds, assists, steals, and blocks. Upon investigating to see if they were the same five rows, I found that they were. They were all players I’ve never heard of with zero minutes, and given that this dataset had over 15,000 rows I knew that just dropping these rows would not introduce any biases in the data. Now that there were no missing values, I analyzed the data for outliers beginning with age. Again, given the nature of this data being stats recorded from actual games and our purpose being to predict the most valuable player, the outliers are the values we’re most interested in. Besides age, outliers are the key to this model being functional, therefore I did very little to alter this data. Relative to the amount of data, there were very few outliers for this parameter. Given that these were players ages and verifying that they were legitimate ages, there was nothing to do with these. The “minutes played” category had no outliers. Next was rebounds, which had quite a few, all on the high end. Upon looking into these outliers I found that there were a few numbers that were extremely high. After looking at the data in descending order, all of the highest numbers were from players that had only played a small handful of games, so their numbers were artificially high. As a result, I reduced my sample size of entries only to players that have played at least 41 games, which is half of an NBA season. After doing this, there were still outliers left, but these were legitimate numbers from players known for rebounding. Next was assists, which also had a considerable amount of outliers. When looking at these values, they were all high numbers but nothing unreasonable, and they were by players known for getting assists. Another thing to note about this data is that the nature of these legitimate player stats really does not have numbers that would dominate others. As a result, I did not need to scale any of my predictors. Steals and blocks both had very similar results, with several outliers on the high side that were all reasonable – for reference as to what I mean by reasonable, the top seven spots for blocks were by a player that was seven foot five, and several of the top spots for points are Michael Jordan and Kobe Bryant. With that said, points was the only predictor with outliers on the low end. After checking these, they also seemed to be legitimate values that wouldn’t get dominated by others.

Next I looked for skews in the data. I found that all of these predictors had a very surprisingly even distribution, which does make some sense considering the nature of athletics. As a result, I felt that none of the predictors needed to be transformed or scaled for the reasons mentioned earlier. Keeping these numbers in their natural form with the distribution that they have seems to me to be the best option for this context.

Chart, line chart

Description automatically generatedChart, line chart

Description automatically generated

Chart, histogram

Description automatically generatedChart

Description automatically generated

Chart, histogram

Description automatically generatedChart, histogram

Description automatically generated

Now that the two independent dataframes are cleaned and ready to be passed to my model, all I need to do is merge them. In order to keep all of the players from the season stats dataframe and simply add a new column that is false for everyone except the MVP of each season, I did an outer join on ‘player\_id” and ‘season’. After this, the last step was replacing all null values in the ‘winner’ column to false.

The model I selected for this purpose was the Naïve Bayes Classifier. Our response variable is categorical and while there is a substantial amount of rows of data, there are only 48 instances where the MVP is true due to our records starting in 1974. As far as the assumption of independence, there is a reasonable amount of independence in the predictors in the sense that any one of the predictors has no causal relationship on any of the others. I defined my predictors and response variables, trained my NB model, and no

When going to do test the model with current players in order for our models prediction on where I should place my bet, it came to my attention that it would not make sense to use minutes as a predictor since players this season would not have a full season worth of minutes.

# Results

I evaluated my model both a Naïve Bayes and a K Nearest Neighbors algorithm with three players, the probable front runners for MVP as the season stands so far: Ja Morant, Stephen Curry, and Luka Doncic.

Ja Morant Prediction: True

Based on Ja Morant’s numbers so far this season, our model predicts that Ja Morant will be the MVP this year.

Text

Description automatically generated

Stephen Curry Prediction: False

Based on Stephen Curry’s numbers our model does not predict that he will be the MVP. He has very similar number to Ja Morant, however his age is much older. This may seem like an oversight by our model, however in the NBA older players that have been great for a long time tend to not win MVP, as if we are all used to their production.

Text

Description automatically generated

Luka Doncic Prediction: True

Our model predicts that Luka will also be the MVP this season based on his numbers.

Text

Description automatically generated

The K-Nearest Neighbor algorithm produced the same results and accuracy scores as my Naïve Bayes model.

Graphical user interface, text, application, email

Description automatically generated

I set aside $10 to be used to place a bet based on my model’s predictions. Given my results, I will place $5 on Luka Doncic, and $5 on Ja Morant, reducing my risk and maximizing my chances of winning.

# Discussion

Both my KNN and Naïve Bayes models had very high accuracy but AUC scores of 0.5. Clearly my model does not do a great job of predicting who will be an MVP between two players. This of course raises the question of why my accuracy is so high, and this I believe is due to the nature of this data. There are over 15000 rows, and only 48 MVPs, so in reality my model could have just guessed false every time and had over 99% accuracy. Before running my models I had not considered the issue with having so many rows with so few true values in our response variable, however it is now clear that the issue may have been with the approach of using every player from every season and just the MVPs as hits. I also tried an SVM model and got the same results.

# References

“Basketball Statistics & History of Every Team & NBA and WNBA Players.” *Basketball*, https://www.basketball-reference.com/.

“Learn R, Python & Data Science Online.” *DataCamp*, https://www.datacamp.com/.

**Data Story**

The features that I am using as predictors in my model are taken from the overarching branch of “per 100 possessions” statistics, a measure that is commonly used to analyze player production on a more even scale, since some players don’t get as many minutes their numbers will naturally be lower despite them potentially being more productive when they’re on the court. Per 100 possession statistics are the counterpart to per 36 minutes statistics, which simply takes a player’s game averages and extrapolates it by multiplying all of their statistics by however many times more 36 minutes is than the amount they played. Per 100 possessions on the other hand does not boost players’ production like that. It simply averages out their actual production over every possession in a season to 100 possessions. The variables I am as predictors are:

* Minutes played (this one is a total for the season)
  + No outliers, very even distribution
* Total Rebounds
  + Several outliers all on the high end, all reasonable valid values. Relatively even distribution
* Assists
  + Many outliers all on the high end, all values were legitimate. Slightly right-skewed data as a result.
* Steals
  + Many outliers all on the high end, all values were legitimate. Even distribution of data.
* Blocks
  + Many outliers on the high end, all values were legitimate. Right skewed distribution, but the skewed values shouldn’t dominate smaller ones.
* Points
  + Some outliers on low end, many on high end. All legitimate recordings, no values should dominate others. Most even distribution I’ve seen on a real dataset.
* Age
  + 7 outliers, all legitimate players that played

I am using minutes played because that is generally a good measure of how good a player is. Pretty much across the board, better players play more minutes. This will also help weed out players that have good numbers but from a small sample size. Total rebounds (as opposed to offensive or defensive), assists, steals, blocks, and points are the main measures used to evaluate any player. These statistics are the most direct way track what a player contributed to a game, and the MVPs always have high numbers in all or most of these categories. The final measure from this category that I am using is age. The reason I am using age as a predictor is because there tends to be a bias against great players that have been doing it for a while, almost as if we are used to their greatness. Many people believe that Lebron James has been robbed of several MVP awards because we are so used to his greatness that it doesn’t stand out anymore, and the MVP award tends to go to players that stand out. For example, anytime a player averages a triple-double or very close over a season, that is three of the aforementioned categories with double digit averages, they pretty much always win the MVP award. Their numbers are so extraordinary that people feel obligated to give them the award (or so NBA fans say). All of this is to say that older players with great numbers tend to be less likely to win the MVP award as their production is unsurprising to people.

Our response variable is a binary column that is either going to be True or False, False for every single player except the one MVP every season.

One interesting thing to note about our data is how little had to be done in order to clean it. The main things that needed done were to remove the very few rows that had missing data. This data was so insignificant that it couldn’t have possibly made any impact on our overall data. The other main thing that had to be done was removing all entries of players that had not played at least half of the season. This was for a couple reasons. First, some of these players had artificially high numbers that revealed themselves when checking for outliers and seeing that they had only played one or two games. It was also obvious from looking at the top performers and they were all names that I was unfamiliar with. The other reason that we got rid of data from players that played less than 41 games is that no MVP has ever played that few games (the lest ever was 49). It is safe to say that a player that hadn’t played at least half of a season is ineligible to become the MVP, and therefore their data would only serve as noise to our model. One thing to note is that I did not scale or transform any of the data. For one, most of the data had pretty even distributions, but even where there was slight skews I opted not to scale the data. I think the context is important for understanding this. I am trying to predict the MVP, so I don't necessarily want all of our values to be clos together. In fact, the outliers are the values that I am going to get the most value out of. The higher values are not so high that they will dominate the other's, so I think leaving them as is the way to go here for best results.