National Basketball Association Most Valuable Player Prediction Model

Robert Shea, Department of Systems Engineering and Engineering Management, Stevens Institute of Technology, Castle Point on Hudson, Hoboken, NJ, USA

The National Basketball Association MVP award is an annual award given to the best performing player of the regular season. The voters of the award each have their own methods of determining the value of a player and whether they are deserving of MVP. In general, the merit for the decision for the recipient of the MVP award is highly subjective and there is a significant amount of open-ended definition of a player’s value. The most prevalent arguments are that the player is the best player on the best team or that the player is putting up historical statistics. These two arguments were used to create a model that would evaluate the value of top MVP candidates according to their contribution to team wins and overall individual output. It is arguably impossible to put an objective truth on value on such a subjective award however the goal is to look at consistent statistics that demonstrate player success and value for the top NBA candidates over the past ten seasons. A numerical value assignment of a player’s value will hopefully make the decision more obvious as to who is more deserving of the MVP award. Current models that measure value have issues such as unjustified weightings of variables, the overvaluing of “big men”, or the undervaluing of a player’s defensive ability. The newer model includes more metrics to measure an individual’s adeptness and success such as Player Efficiency Rating and Game Score. These newer metrics aim to offset the overvaluing of big men and balance the player’s contribution to team wins with the player’s overall individual input without using arbitrary scalar values. In addition, the weighting for Net Rating was increased to investigate whether this would have a positive effect for better defensive players. However, the metric in its current state does not consider certain intangibles such as a player’s true defensive ability or player narrative. The introduction of newer defensive advanced metrics in the future could be modified into the metric to add more value to better defensive players.

* 1. Introduction
  2. Problem Statement

The National Basketball Association Most Valuable Player Award is an annual award given since 1955-1956 to the best performing player of the regular season. Until the 1979-1980 season, the recipient of the reward was selected by a vote from the NBA players. Since the 1980-1981 season, the award is selected by a panel of sportswriters and broadcasters from the United States and Canada [10]. The voters each have their own methods of determining the value of a player and whether they are deserving of MVP. In 2015-2016, Stephen Curry won the award unanimously however there are seasons in which the MVP of the league is much less obvious because multiple players performed at a high level and led their team to a great number of wins. This metric would be used to compare many MVP candidates by assigning a value to them based on box-score statistics and advanced statistics.

* 1. Problem Motivation

The Most Valuable Player (MVP) award is the highest individual accolade that a player may receive in the National Basketball Association (NBA). The recipient of the MVP award is decided upon by media members and the merit for this decision is highly subjective. The open-ended definition of value for the award incites intense conversation on how to justify value for a player. Each season, the award is given to a single player and there is no real consolation to players that received the second or third most MVP votes. All-NBA Honors is given to the top 15 performing players for a single season and the Defensive Player of the Year is given to the best defensive player for a season. Receiving an MVP is important to a player’s legacy and helps them gain the respect of their fellow players however some players may also have a bonus in their contract in place that would award them a significant amount of money if they were to win the award. This gives the player a significant amount of incentive to win the award. Team owners and general managers also have the incentive for their players to win the award to increase fan support and increase revenue. Currently, the panel of MVP voters are encouraged to take the voting process seriously however they are not required to release their votes so their reasoning for voting for a player may be biased. The most prevalent arguments are that the player is the best player on the best team or that the player is putting up historical statistics [3][21]. These two arguments were used to create a model that would evaluate the value of top MVP candidates according to their contribution to team wins and overall individual output. It is arguably impossible to put an objective truth on value on such a subjective award however the goal is to look at consistent statistics that demonstrate player success and value for the top NBA candidates over the past ten seasons. A numerical value assignment of a player’s value will hopefully make the decision more obvious as to who is more deserving of the MVP award. However, the metric in its current state does not consider certain intangibles. For example, a player may receive more votes because they are more likable, or a player may receive less votes because of “voter fatigue” or they are unlikable [8][20]. However, it is important to consider that the MVP award means a lot to players whether it be for pride or for money and the media members have an obligation to assign the award to the most deserving player. The goal is to make the most objective evaluation of each top player so that the choosing of an MVP is done with the least amount of consideration of intangible factors. The 2019-2020 NBA season is currently suspended due to COVID- 19 however current statistics may be evaluated to determine which player is the front runner MVP candidate for this season.

* 1. Problem Background

In general, the merit for the decision for the recipient of the MVP award is highly subjective and there is a significant amount of open-ended definition of a player’s value that incites conversation. Perhaps one of the reasons that there is no clear-cut definition of who deserves to be the league MVP is because the media stands to benefit greatly from constant discussion of who deserves the MVP award. “Hot takes” are known as unpopular opinions as to who deserves the MVP and “hot takes” allow sports writers to generate more clicks and in turn generate more advertisement revenue. Peter Li is a data scientist and self-proclaimed “NBA nerd” that created an MVP prediction model based on team and individual success [11]. The model was well made however it came with disadvantages such as the inclusion of arbitrary scalar values and the undervaluing or overvaluing of certain players. Ultimately, the model was enhanced to try and address some of these issues and leave room for modularity in the future.

* 1. Technical Introduction

This model includes box score statistics and advanced statistics that were taken and analyzed from basketball-reference.com. The Player Comparison Finder tool was useful for comparing the box-score and advanced statistics of the top players for each season [15]. Box score statistics are any individual statistics that can be recorded for a player after a single game played. This includes but is not limited to: minutes played, points scored, assists on field goals made, rebounds, steals, blocks, turnovers, or personal fouls. Advanced statistics are metrics that define a player worth and are typically based on an equation that includes variables that are box score statistics. Advanced statistics include Value Over Replacement Player (VORP), Player Efficiency Rating, Usage Rate, and Win Shares.

1.4.1 Assumptions and Uncertainties:

There is a significant amount of uncertainties when creating a model to accurately predict the MVP of the NBA. The panel of voters do not typically release their justification for why they voted for a certain player and so it is uncertain how each voter assigns value to a player. Some voters may weigh individual statistics more than team wins because they show how well a player performed independent of team performance [21][22]. However, some voters may also value the best player on the best team much more than a player who put up better individual statistics on a lesser team [3][14]. Therefore, individual statistics and win contributions for a player were weighted equally. Also, certain liberties were taken in evaluating which advanced statistics should have higher weighting than others. Advanced statistics such as Player Efficiency Rating and VORP both aim to evaluate a player’s value however one statistic is not more widely accepted over the other. Furthermore, defense is valued in Defensive Rating or Defensive Box Plus Minus however these metrics are not catch all in evaluating a player’s defensive prowess and so the following metric is not perfect in accounting for a player’s defensive abilities. Some MVP voters may value a player’s defensive ability in their vote however defensive ability may be represented by the “eye test” of how a player looks to perform defensively when watching them in game. Defense may also be influenced by reputation. Also, defensive metrics may rely on blocks and defensive rebounds to represent how well a player plays defense however both those box score metrics are biased towards players with a height advantage. Therefore, the assumption was made that the available box score and advanced statistics would be sufficient in evaluating a players value and the addition of newer statistics that address “intangibles” may be included in the metric in the future to further refine it.

* 1. Background

Relevant research towards predicting the NBA MVP was done by Peter Li in April 2019 and his model uses a combination of box score and advanced statistics to assign a numerical value to each player [11]. The newer model that was created was a modification of Peter Li’s model to try and address some of the glaring issues with his model. According to Peter Li, value is an even average of a players Win Contribution and a players Total Stats.

Value= .5(Win Contribution) + .5(Total Stats)

Win Contribution = Level of Impact\*Quality of Impact

Level of Impact = ((Team Wins)\*(Games Played/82)\*(Minutes/48)\*(Usage Rate/100)

Quality of Impact=.4(VORP + Win Share) + .2(Net Rating)

Total Stats= (Points\*(True Shooting Percentage) + 1.5\*(Assists) + 1.2\*(Rebounds) + 3\*(Blocks) + 3\*(Steals) – Fouls – Turnovers)/25

In his model, the calculation of Total Stats was done with each player’s total box score statistics for the season. In other words, the total points that the player scored for a season or the total assists on field goals for a season. The solution from this calculation was too high compared to each player win contribution and so to make the two numbers closer the Total Stats was divided by a scalar value of 25 [11]. However, this number has no real significance and there is no justification to divide the Total Stats by 25. Dividing the Total Stats by the amount of games the player played would give Per Game Total Statistics and dividing the Total Stats by 36 would give Per 36 Minutes Total Statistics. The arbitrary selection of 25 appeared to be the first room for improvement in Peter Li’s model and was addressed in the newer model.

The results of Peter Li’s value calculation also tended to overvalue or undervalue certain groups of players. In 2015, the pace and playstyle of the league had shifted considerably with the success of the Golden State Warriors as a three-point shooting team. Analytics has shown that it is much more valuable for teams to be taking more three pointers and so other teams began developing talent around the three-point shot [2]. Stephen Curry and Klay Thompson of the Golden State Warriors were notable exceptional three-point shooters that helped cause this shift towards teams taking more three-point shots. The success of the Golden State Warriors in winning 4 championships since 2015 has shifted the league wide perception of analytics. Therefore, although players such as Kobe Bryant and Carmelo Anthony are widely accepted as great players, Peter Li’s model significantly undervalued them due to their poor advanced statistics. The Golden State Warriors pride themselves on unselfish team basketball however players such as Kobe Bryant and Carmelo Anthony found their success playing “hero ball” and isolation basketball. These players were high usage, low efficiency scorers and although they are great basketball players, the model would not typically reflect this in their value. This is a disadvantage of the model that was not fixed with the newer model. Both models rely on statistics and analytics to assign value to a player and without proper analytics to justify Kobe Bryant’s hero ball playstyle, he would be consistently undervalued by the model. Future iterations of the model would aim to address this undervaluation if there is ever newer analytics.

Peter Li also claimed that the model overvalued “big men” such as Kevin Love, Karl-Anthony Towns, or Blake Griffin [11]. These players would have great value according to this model by demonstrating a combination of great box score statistics and favorable advanced statistics. These players may not be ball dominant compared to their point guard teammates that facilitate the team’s offense however it can be argued that these “big men” are more valuable to the team because of their contribution despite not being a ball dominant player. Therefore, it is more likely that these players are undervalued by the media and are more deserving of consideration in the MVP conversation. One reason the media may give more attention to the smaller ball dominant players is because they have a “flashier” playstyle that is more enjoyable to watch. The value of these big men is accurately reflected in the model however “shorter” players that rely on shooting may be undervalued because they are getting less rebounds. This is addressed in the newer model with the modification of the Total Stats calculation.

Defense was another player characteristic that was undervalued in Peter Li’s model. As addressed previously, currently there is not a lot of available advanced statistics that accurately evaluate a player’s defensive capability. Peter Li discusses how the addition of better defensive statistics may not improve the model significantly because the general fan bias is towards offense and the league MVP can be heavily influenced by public sentiment/narrative. Defense does not typically make headlines or attract sponsorships so he believed that the model would not significantly change with the addition of newer analytics [11]. However, when evaluating the newer model there were three obvious instances where defensive statistics may have greatly benefited the players. In 2015-2016, Kawhi Leonard finished second in MVP voting due to his success on defense and his winning of the Defensive Player of the Year award. In 2018-2019, Giannis Antetokounmpo was known as one of the better defensive players in the league and his team’s success was attributed to his success on defense. In 2013-2014, Joakim Noah finished fourth in MVP voting due to his success on defense and his winning of the Defensive Player of the Year award. These players deserved better value because of their success on defense however it was not properly represented in Peter Li’s model. The new model attempts to slightly address this undervaluation (despite lack of defensive analytics) by adjusting the weighting of each players Net Rating from .2 to .3 in the Quality of Impact calculation. This is a small change that would offset some of the undervaluation.

3. Body of Work

3.1 Discussion of Proposed Methods

3.1.1 Value

The goal of this metric is to determine value based on the two main narratives that drive media members to choose an MVP. The weighting of Total Stats and Win Contribution are even because there is no true evidence that one should be valued more than the other. To make the model as accurate as possible across the last 10 years, both variables were weighted the same. The formula for Value is shown in equation (1).

Value = .5 (Total Stats) + .5(Win Contribution) (1)

3.1.2 Total Stats

Total Stats is the first component that is determined when calculating an individual’s value. The calculation for Total Stats is performed with three components: Fantasy Basketball Stats Average, Game Score, and Player Efficiency Rating. The formula for Total Stats is shown in equation (2).

Total Stats = ((.3\*(Fantasy Basketball Stats Average)) + .3\*(Game Score) +

.4\*(Player Efficiency Rating) (2)

3.1.2.1 Fantasy Basketball Stats Average

The first component for calculating Total Stats for a player is Fantasy Basketball Stats Average (FBS Average). Fantasy Basketball Stats (FBS) is based on the weighting and statistics that are a part of fantasy basketball. The weighting of the statistics was done by several different departments of the NBA and was influenced by the fans as well via a survey [5]. However, fantasy basketball does not include a multiplier for points that adjusts for scoring efficiency. True Shooting Percentage is a statistic that measures a player’s shooting efficiency that considers field goals, 3-point field goals, and free throws. True shooting percentage may be calculated using equations (3) and (4) but for the purpose of simplifying the model each players True Shooting Percentage was taken from the already calculated value provided by basketball-reference.com.

TS% = Points/(2\*True Shooting Attempts) (3)

True Shooting Attempts = TSA = (Field Goal Attempts) + .44(Free Throw Attempts) (4)

Where Field Goal Attempts contains both 3-point field goal attempts and 2-point field goal attempts [7]. FBS includes True Shooting Percentage so that statistics such as field goal makes and attempts, free throw makes and attempts, and 3-point makes, and attempts does not need to be included.

Peter Li calculated Total Stats by just using Fantasy Basketball Stats. He took the totals of the players box scores and input those variables into equation (5). Then he divided the solution by 25. The value of 25 was chosen because it would make the value of Total Stats closer to that of Win Contribution however there was no further justification for the use of that number. With the newer model, Fantasy Basketball Stats is calculated as shown in equation (6). Points, assists, rebounds, etc. are all individual accolades for a player that are recorded on a game by game basis. At the end of the season, each player will have total for each individual accolade. The totals for each individual accolade were used in the Fantasy Basketball Stats Totals Equation as shown in equation (7). Evaluating totals is critical because it demonstrates a player’s longevity and willingness to play as many games in a season. The league MVP should be the player that is most important to his team and is leading his team to the most wins. This would ideally be the player that is putting up the most points, assists, rebounds, etc. across the league. However, the reason that this component of FBS Average is weighted the least is because injuries and load management are commonplace in basketball. The NBA season is 82 games and some players will make a “business decision” to sit out less important games against weaker opponents so that they can rest up and mitigate risk of injury in a game that may be won by the rest of the team. In a perfect world without injuries, the league MVP should be playing every single game however there are small injuries that if exacerbated may become season ending or career ending. The average amount of games played by the top 20 Value Over Replacement Player (VORP) players in the 2018-2019 season was 72.65 games. A player that plays all 82 games will be playing about 10 more games than the average and at similar skill levels he will surely be totaling more points, assists, and rebounds. Therefore, the weighting of FBS Totals in equation (6) was to be less than the other two components as to not excessively benefit players that play more games than average. The weightings of the three components were tweaked for optimization purposes and this will be better explained in section 4. The other two components are Fantasy Basketball Stats Per 100 Possessions and Fantasy Basketball Stats Per 36 Minutes and their calculations may be seen in equations (8) and (9) respectively. Each player has his total points, rebounds, and assists for a season and the points per game, rebounds per game, and assists per game are all determined by dividing the totals by the amount of games that the player played. “Per 100 Possessions” and “Per 36 Minutes” are two methods of comparing a players performance to each other because not every player plays the same amount of games and not every player plays the same number of minutes per game. The “Per 36 Minutes” statistics is like per game statistics however some players may play more or less than 36 minutes per game and this adjusts their statistics to 36 minutes a game [16]. This will inflate a player’s statistics if they typically play much less than 36 minutes per game however the top 20 VORP players of the 2018-2019 season played on average 34 minutes a game so this was not a concern. It is a method of comparing the potential MVP candidates using the same minutes per game played. The “Per 100 Possessions” is a little more complicated in that it adjusts for pace of play. The style of play and the pace of the game can change year after year and players on faster teams will put up higher per game and per 36 statistics than slower teams [16]. This is another way of balancing and comparing the performance of players across a season. These two components had the same weighting in the FBS average equation because either component may be argued as a better method of comparing the performance or value of two players. The results of the FBS average equation turned out to be greater than that of Total Stats calculated by Peter Li which means that if Total Stats was still based on the calculation using Fantasy Basketball weightings then it would still significantly influence the results of the players Value compared to Win Contribution. The weighting of a player’s performance based on Fantasy Basketball weightings is still relatively new therefore the average of the three calculations of Fantasy Basketball Stats would help achieve a more balanced evaluation of a player’s individual performance. The other two new components of Total Stats equation would serve the same purpose of evaluating a player’s individual performance for the season while also stabilizing Total Stats value to be on the same magnitude as Win Contribution.

Fantasy Basketball Stats = Points\*True Shooting Percentage + 1.5(Assists) + 1.2(Rebounds) + 3(Blocks) + 3(Steals) – Personal Fouls – Turnovers (5)

Fantasy Basketball Stats Average = (.275\*(Fantasy Basketball Stats Totals) + .3625\*(Fantasy Basketball Stats Per 100 Possessions) + .3625\*(Fantasy Basketball Stats Per 36 Mins) (6)

Fantasy Basketball Stats Totals = (Total Points) \* True Shooting Percentage + 1.5(Total Assists) + 1.2(Total Rebounds) + 3(Total Blocks) + 3(Total Steals) – Total Personal Fouls – Total Turnovers (7)

Fantasy Basketball Stats Per 100 Possessions= (Points Per 100 Possessions)\*True Shooting Percentage + 1.5(Assists Per 100 Possessions) + 1.2(Rebounds Per 100 Possessions) + 3(Blocks Per 100 Possessions) + 3(Steals Per 100 Possessions) – Personal Fouls Per 100 Possessions – Turnovers Per 100 Possessions (8)

Fantasy Basketball Stats Per 36 Minutes= (Points Per 36 Minutes)\*True Shooting Percentage + 1.5(Assists Per 36 Minutes) + 1.2(Rebounds Per 36 Minutes) + 3(Blocks Per 36 Minutes) + 3(Steals Per 36 Minutes) – Personal Fouls Per 36 Minutes – Turnovers Per 36 Minutes (9)

3.1.2.2 Game Score

The second component of the Total Stats Calculation is Game Score. This is a similar metric to FBS however it has different weighting for statistics. Game Score was created by John Hollinger to give a rough measure of a player’s productivity for a single game [6]. Equation (10) shows the calculation for Game Score. Using a players per game statistics, the Game Score for each possible MVP candidate was determined. The purpose of Game Score is to evaluate the offensive and defensive performance of a player and it takes every statistic that is typical for a box-score. A Game Score of 40 would indicate that the player had a fantastic game while a Game Score of 10 would reflect an average performance by a player [6]. Although Game Score is typically only used on a single game basis, in this case the “per game” statistics were used in the equation to give a Game Score for the season.

Game Score= (Points + .4\*(Field Goals) - .7\*(Field Goal Attempts) - .4\* (Free Throw Attempts – Free Throws) + .7\*(Offensive Rebounds) + .3\*(Defensive Rebounds) + Steals + .7\*(Assists) + .7\*(Blocks) - .4\*(Personal Fouls) - Turnovers. (10)

3.1.2.3 Player Efficiency Rating

The third component of the Total Stats Calculation is Player Efficiency Rating (PER). Player Efficiency Rating is another metric that was developed by ESPN columnist John Hollinger. It is meant to sum up all the players positive accomplishments, subtract the negative accomplishments, and return a per minute rating of a player’s performance [7]. A breakdown of the calculation of PER is shown below [9]. However, for the sake of reducing the complexity of the model, the calculated PER for each MVP candidate is taken from the advanced statistics available from basketball-reference.com. The main drawback of PER is that top tier defensive players may not be well represented. Player Efficiency Rating is based on the personal actions of a player and does not accurately include a player’s defensive capability or their ability to spread the court to give their teammates better scoring options. Also, PER assumes that all teams are equal which is indicted each time a “league” statistic pops up. Playing the team with the best record is not the same as playing the team with the worst record [9].

Player Efficiency Rating Calculation:

**Unadjusted PER = uPER** = (1/(Minutes Played))\*[3 Pointers

+ 2/3\*Assists

+ (2-factor\*(team assists)/(team field goals))\*field goals

+ (Free throws\*.5\*(1+(1-(team assists)/(team field goals))+2/3\*(team assists)/(team field goals))

- VOP\*Turnovers

- (VOP\*(Defensive Rebound Percentage)\*(Field Goal Attempts-Field Goals))

- (VOP\*.44\*(.44+(.56\*(Defensive Rebound Percentage))\*(Free Throw Attempts-Free Throws))

+ (VOP\*(1-(Defensive Rebound Percentage))\*(Total Rebounds- Offensive Rebounds)

+ (VOP\*(Defensive Rebound Percentage)\*Offensive Rebounds

+ (VOP\*Steals)

+ (VOP\*(Defensive Rebound Percentage)\*Blocks)

- (Personal Fouls\*(league free throws)/(league personal fouls)-.44\*((league free throw attempts)/(league personal fouls)\*VOP)]

**factor** = (2/3) – (.5\*(league assists/league field goals)/(2\*(league field goals/league free throws))

**VOP** = league points/ (league field goal attempts – league offensive rebounds+ league turnovers + .44\*league free throw attempts)

**Defensive Rebound Percentage** = (league total rebounds- league offensive rebounds)/(league total rebounds)

After uPER is calculated, an adjustment is made for the teams pace.

**Pace adjustment**= league pace/ team pace

**Adjusted PER** = aPER= (pace adjustment)\*uPER

Standardizing the PER is done by calculating the league average aPER (league aPER) using player minutes played as the weights.

**PER** = aPER\*(15/league aPER)

John Hollinger set up PER so that the league average for each season is 15.00. Table 1 shows a reference guide for the significance of a PER [12]. It shows that the implementation of PER is vital to demonstrate a player’s value and determine if they had an MVP season.

Table 1: Significance of PER

|  |  |
| --- | --- |
| All- time great season | 35.0 + |
| Runaway MVP Candidate | 30.0 – 35.0 |
| Strong MVP Candidate | 27.5 – 30.0 |
| Weak MVP Candidate | 25.0 – 27.5 |
| Definite All-Star | 22.5 – 25.0 |
| Borderline All-Star | 20.0 – 22.5 |
| Second Offensive Option | 18.0 – 20.0 |
| Third Offensive Option | 16.5 – 18.0 |
| Slightly Above Average Player | 15.0 – 16.5 |
| Rotation Player | 13.0 – 15.0 |
| Non-Rotation Player | 11.0 – 13.0 |
| Fringe Rotation Player | 9.0 – 11.0 |
| Player Who Will Not Stick In The League | 0 – 9.0 |

The three components of the total stats calculation are three different metrics that measure a player’s individual value and success. Player Efficiency Rating is a widely accepted metric of evaluating an individual’s value and a high PER for a season typically correlates with the MVP winner. FBS Average considers the players total individual performance for the season which means that it is an important metric to account for when evaluating the MVP for an entire season. The similarity of measurement between FBS Average and Game Score meant that they were to receive the same weightings of .3. The importance of PER means that it was given a weighting of .4. FBS Per 100 Possessions, FBS Per 36 Minutes, PER, and Game Score are all meant to equalize the individual performance of multiple players regardless of games played, minutes played, or “height advantage”. One of the disadvantages of Peter Li’s metric is the “overvaluing” of big men and the incorporation of all these metrics in the calculation of Total Stats is to lessen this disadvantage as much as possible. The overall individual output is a great way of understanding which players are having better seasons however some players on worse teams may be putting up exceptional statistics while not exactly leading their team to wins. Analyzing win contribution allows the evaluation of a player’s success and how it contributes to the team success.

3.1.3 Win Contribution

The Win Contribution is the second component that is calculated when determining a player’s value for a season. The calculation of Win Contribution is performed with two components: Quality of Impact and Level of Impact. The formula for Win Contribution is shown in equation (11).

Win Contribution = Quality of Impact \* Level of Impact (11)

3.1.3.1 Quality of Impact

The purpose of Quality of Impact is to measure a player’s quality of impact when they are in the game. In baseball, Wins Above Replacement is a widely accepted measure of a player’s quality of impact that summarizes a player’s total contributions to their team in one statistic [18]. In basketball, there is no comparable widely accepted metric of a player’s quality of impact and so weighted values of multiple advanced statistics were used to create this measurement. The calculation for Quality of Impact is shown in equation (12).

Quality of Impact = .35(Win Share) + .35(VORP) + .3(Net Rating) (12)

Win Share measures a player’s contribution to a team’s win total based on the players production on offense and defense. This metric is adjusted for league average production, and league and team pace to make it a consistent metric across the league. Win Shares estimate how many wins a player is responsible for on his team. A team’s win should be near the sum of each of its individual player’s win share totals [5]. Value over Replacement Player (VORP) is a measurement of a player’s relative impact (based on Box Plus/Minus), is adjusted for a player’s minutes played and compared to a league average. It is a box score estimate of the points per 100 team possessions that a player contributed above a replacement level player. The estimate is translated to an average team and prorated to an 82-game season [5]. Net Rating is a measurement of the point difference between how much a player produces on offense per 100 possessions and how much a player gives up on defense per 100 possessions. The measure of how much a player produces on offense per 100 possessions is known as their Offensive Rating and the measure of much a player gives up on defense per 100 possessions is known as their Defensive Rating [19]. Win Share is a value metric that may be used to distinguish a player’s contribution to a win relative to his teammates. VORP is a value metric that may be used to distinguish a player’s contribution relative to the rest of the league. A disadvantage of these two metrics is that they penalize a player that plays lower minutes. Some instances a super star will play significantly less minutes compared to the league average because his team is leading greatly, and any further play is just a risk of injury. Net Rating is a value metric that will distinguish a player’s overall output when they are playing and affect their Quality of Impact in favor of players that play less minutes. Net rating has a lower impact on the calculation of quality of impact because historically the metric will favor “big-men” that perform better on defense due to their height advantage. Furthermore, the calculation of Net Rating is based on Offensive Rating and Defensive Rating however Defensive Rating is also used to calculate Defensive Win Shares, which is a part of Win Share [5]. Peter Li assigned a lesser weighting to Net Rating to mitigate the overvaluing of “big men” with his model however it inadvertently hurt players such as Kawhi Leonard who has a high defensive rating because he is a better defensive player. Therefore, the weighting of Net Rating was increased from .2 to .3 so that the results of the new model may further benefit better defensive players.

3.1.3.2

The purpose of Level of Impact is to assess a player’s level of impact on his team’s success based on team wins during the games and minutes played, and usage rate. The part of this metric that demonstrates a player’s high value is if the team is winning games. The other components of Level of Impact aid in proving that the player was key in winning those games. The team wins for each player was taken from season summaries available from basketball-reference.com The calculation for Level of Impact is shown in equation (13).

Level of Impact = Team Wins\*(Games Played/82)\*(Minutes Played/48)\*(Usage Rate/100) (13)

The usage rate metric is used to measure how much a team relies on a player in possession in comparison with his teammates. For example, during the 2015-2016 NBA season the Golden State Warriors won a record breaking 73 games during the regular season. The three players that were key to that team’s success were Steph Curry, Klay Thompson, and Draymond Green and they each played about 80 games each and averaged 34 minutes a game. These players did not all have the same level of impact on the team’s success and usage rate is the differentiator that will demonstrate the player that more so impacted team wins [11]. The main issue with the usage rate statistic is that although it is generally biased towards guards and ball-handlers that control the flow of the team’s offense, elite big men such as Joel Embiid and Anthony Davis consistently rank in the top ten for usage rate due to their first-class skill. Usage rate can be heavily skewed towards big-men and to balance out this bias it is included in the win contribution formula.

3.2 Proposed Methods

The goal of this project is to create a prediction model using Python and Excel that will calculate the value of an individual based on Team and Individual success. The statistics of each individual MVP candidate will be collected and stored in a dictionary. Each statistic is critical towards proving the value of a player and this model is meant to be as accurate as possible. The top MVP candidates of the last 10 years will be tested with this model to prove its accuracy. There are three groups of players that were evaluated with this new model. The first group are the players that received MVP votes. The panel of MVP voters each cast a vote for first to fifth place selections. The first-place vote is worth ten points, the second-place vote is worth seven points, the third-place vote is worth five points the fourth-place vote is worth three points and the fifth-place vote is worth one point. Every player that received a single vote from the panel was to be evaluated for their value. The players that received MVP votes each season was found from NBA Awards Voting that is available for each season on basketball-reference.com The second group are players that received All-NBA honors. There are 3 All-NBA teams per year and 5 players are selected for each team based on their overall performance for that season. The MVP winner for that year will be on All-NBA first team and typically any other player that received an MVP vote would also be on an All-NBA Team. Therefore, it was appropriate to test the value of players that were the best all-around players for a season. The players on All-NBA teams was determined from the season summary available through basketball-reference.com. The last group of players are the top 20 players in Value Over Replacement Player (VORP) for a season. As previously described, VORP is based on Box Plus Minus (BPM). Box Plus Minus estimates a player’s contribution to his team’s point differential, relative to an average player whose BPM is zero. VORP is a minutes-weighted counterpart to BPM. Essentially, it shows the value of a superstar basketball player in comparison to an average player or a bench player [13]. The reason that this statistic was singled out was because since 2006, except for 5 instances the league leader in VORP has won MVP every year [11]. A graph was also made for each season to display how each player performed based on team success and individual success for the season. The validity of the metric was determined by evaluating the top players each season for the last ten seasons. This would show whether the metric is accurate at predicting the MVP, and it would also show if the metric is labelling value like how the media members were determining which players deserve All-NBA honors. The model would also be used to predict the MVP winner of the 2019-2020 season, although it has been cut short. Lastly, the value for each player each season was combined into a comprehensive table to determine the MVP of the decade based on the new value model.

The players that were to be evaluated with the new value model were saved in 11 Excel files that were named for the corresponding season of the pool of players. The first season that was evaluated with an established MVP winner was the 2009-2010 season and the last season that was evaluated with an established MVP winner was the 2018-2019 season. The 11th season is the 2019-2020 season and is incomplete, so the model is to be a predictor in this instance. Since there are three groups of players that are being evaluated with the model, the number of players per season that are evaluated with the model ranges anywhere from 20 to 29 players. Every player from the 2018-2019 season that received an MVP vote or All-NBA honor was in the top 20 VORP players whereas the top 20 VORP players for the 2011-2012 season included many players that did not receive either an MVP vote or All-NBA honor. Each Excel file has 36 sheets that have specific information that the Python file can read. The first sheet is the “Per Game” sheet that includes each player’s “per game” box score statistics including minutes per game, assists per game, rebounds per game, and points per game. The second sheet is the “Totals” sheet that includes each player’s “total” statistics including total minutes played, total games played, total points scored, total rebounds, and total assists. The third sheet is the “Advanced” sheet which includes PER, Usage Rate (USG%), Win Shares, and VORP. The fourth sheet is the “Per 100 Poss” sheet which includes points per 100 possessions, assists per 100 possessions, and rebounds per 100 possessions. The fifth sheet is the “Per 36 Min” sheet which includes points per 36 minutes, assists per 36 minutes, and rebounds per 36 minutes. The sixth sheet is “MVP Voting” which includes the results of the MVP voting for each season and shows how many votes each player received. The rest of the 30 sheets is for the 30 NBA teams. It includes the roster and the number of wins that the team earned that season.

First, the Python program would read a file name using a file\_reader() function and determine if the file can be found and opened. Any problems would raise an error. Next, the first sheet of the file would be read by evaluating the name of each column to find the “Player” column that has the name of each NBA player that will be evaluated for that season. The players name plus the name of the excel file (the season that is being evaluated) was to be stored as a new instance and key in the self.players dictionary that is a part of the Excel\_Data\_Reader class. The name and player index for each player were then stored as values for their corresponding keys. Then, the individual box score and advanced statistics were parsed from the excel sheets and stored as values for the respective player keys. Each value, except for the season, was stored as a float so that it could be used in calculations. Each statistic could be found by the name of the column. The row that corresponds to each player was deciphered as the players index number as stored in the excel sheet. The players minutes played was labelled as “MP” in the sheet and was needed for the Level of Impact calculation. The minutes played was found in multiple sheets, so it was vital that only the player’s minutes per game was parsed. Similarly, field goals made, and field goals attempted were needed for Game Score but only the per game stats were needed so the parser needed to make sure that the sheet name was “Per Game” first before it parsed the data. The player’s points, assists, steals, blocks, turnovers, and personal fouls needed to be parsed four different times from four different sheets to be used in multiple calculations. Therefore, the Python data parser needed to make sure that it was always working with the right sheet and the right column. The players that received MVP votes were determined by checking if that players name was in the “Player” column in the “MVP Voting” sheet. The players ranking in the MVP vote for that season was then stored as a value by reading the “Rank” column and cross referencing the row that has the same name as the player. The dictionary value for a player’s wins was determined by reading the roster for each team, checking if the players name was under the “Player” column, and then getting the team wins from the first row of “Wins” column. This is not the most efficient method of getting player statistics for each player, but it was a simple method that was relatively easy to understand and implement.

After all the statistics were stored as values in the dictionary, they were used when applicable to calculate FBS Totals, FBS Per 36 Minutes, FBS Per 100 Possessions, FBS Average, Game Score, Total Stats, Net Rating, Quality of Impact, Level of Impact, Win Contribution, and Value. The calculations for each equation were done automatically by Python for each player and the equations for each calculation are the same as defined in section 3.1. Next, the table() equation would create three different tables to be outputted to an Excel file into three different sheets. The name of the excel file would correspond to the season and would be in the format “season\_Results.xlsx”. Each table included the player name, the season, the Fantasy Basketball Stats Average results, the Game Score results, the Player Efficiency Rating, the Total Stats results, the player’s VORP, the Quality of Impact results, the Level of Impact results, the Win Contribution results, the Value results, and the players MVP Voting Standing if applicable. The first table is the table that is organized by Value in descending order. This shows us the players that scored the best according to the metric and each players MVP Voting Standing is also displayed in the table to see how the results of the model compare to the results of the MVP votes. The next table shows the same columns as the first table except the table is organized by MVP Voting Standing. This is another way of visualizing the value of the players who received MVP votes. The third table is organized by each players VORP in descending order to visualize the players value compared to their VORP. This would help display if there was any correlation between a player’s VORP and a player’s calculated Value. The last output of the program would be a scatter chart that would display a player’s Total Stats vs Win Contribution. The chart would include a linear regression line that is a representation of the average for Total Stats and Win Contribution for the players. Each circle in the scatter chart is represented by a player.

There are also two other iterations of the program that have different functions. The first is for predicting the MVP instead of validating the model. The original program would analyze the sixth sheet of the excel file which is the results of the MVP voting. The first iteration analyzes the sixth sheet for the 2019-2020.xlsx file which is “MVP Tracker”. The “MVP Tracker” sheet displays the top MVP candidates according to basketball-reference.com and the Python model will output the value for all the top MVP candidates to predict the MVP winner for the 2019-2020 season. The second iteration of the program will output the total Value results across the last ten years. The program will iterate through all the excel files and parse the statistics for each player each season and then output the Value and other relevant calculation results into the same tables as the original program. This will show the MVP of the last decade based on value. This second iteration would output into a Decade\_Results.xlsx file the comprehensive tables and a comprehensive scatter chart that includes all players evaluated over the last 10 seasons.

1. Results

4.1 2009-2010 Results

The results of Peter Li’s model are shown in table 2. The results of the newer model are shown in table 3.

Table 2: Peter Li 2009-2010 Results

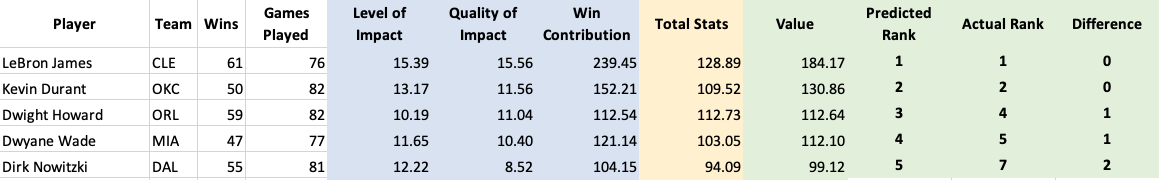


Table 3: 2009-2010 New Model Results



The newer model has significantly higher values due to higher total stats results and slightly higher win contribution results. The correct MVP was still predicted and the players with top 5 Value remained the same. According to John Hollinger, with a PER of 31.1 Lebron James is a Runaway MVP Candidate [12]. Table 4 shows the results organized by MVP Voting Standing.

Table 4: 2009-2010 MVP Voting Standing



Kobe Bryant (3rd in voting and 9th in value) and Carmelo Anthony (6th in voting and 20th in value) were both undervalued with this model, but the results were expected as they had unspectacular advanced statistics. They were below the average VORP (4.436) for the players analyzed, below average in Win Share (10.48) for the players analyzed and well below average in Net Rating (9.96). Peter Li’s model also undervalued these two players for this season because they did not have great advanced statistics. This means there is not exactly an issue with the new model, but it shows the beginning of a trend that advanced statistics were not as favored during this time. Table 5 shows the relation between VORP and Value for 2009-2010 season.

Table 5: 2009-2010 VORP and Value



The top 3 VORP players were also in the top 4 of Value and the player with the worst VORP also had the worst Value. Despite Stephen Jackson’s low value and low VORP, he received a single MVP vote which means that advanced statistics are not the only consideration of the voters. Chart 1 shows Total Stats vs Win Contribution for the 2009-2010 season and that Lebron James was the clear MVP winner.

Chart 1: 2009-2010 Total Stats vs Win Contribution

* 1. 2010-2011 Results

The results of Peter Li’s model are shown in table 6. The results of the newer model are shown in table 7.

Table 6: Peter Li 2010-2011 Results

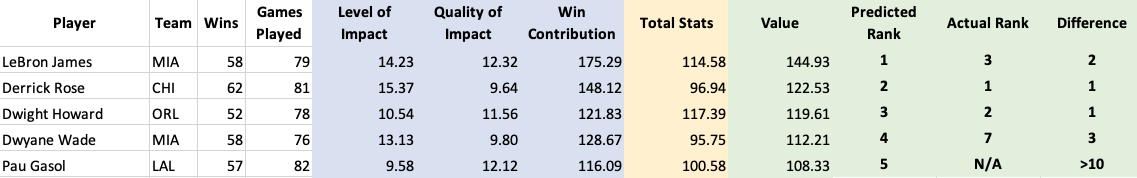


Table 7: 2010-2011 New Model Results



The difference between the two results is that Dwight Howard now has the second highest value compared to Derrick Rose. This is the first instance where the model had an error. This season, Lebron James switched teams from the Cleveland Cavaliers to the Miami Heat and became one of the biggest sports villains in the country [8]. The narrative surrounding Derrick Rose was that he was a humble, young, hometown guard of Chicago that helped lead the team to 62 wins. This shows that the media will vote with intangibles that are not easy to depict in a model. According to John Hollinger, with a PER of 23.5 Derrick Rose is a Definite All-Star while Dwight Howard and Lebron James are both weak MVP candidates [12]. Dwight Howard received the second most MVP votes that season because he had a spectacular season averaging 22.9 points per game, 14.1 total rebounds per game, and 2.4 blocks per game. He was a great offensive and defensive player, and this is demonstrated by his high value in the new model. Lebron James had league high VORP, PER, and Win Shares and helped lead the Miami Heat to 58 regular season wins. Derrick Rose led his team to more wins and had a higher usage rate, meaning that his team used him more. Therefore, his higher Level of Impact could have also had a correlation with voters voting for him. Pau Gasol is notable in the top 5 because he did not receive any MVP votes. This is most likely because he was teammates with Kobe Bryant, the more notable star on the 2010-2011 Lakers team that won 57 games. Pau Gasol’s high net rating (20) and above average advanced statistics most likely resulted in his high value. Table 8 shows results organized by MVP standing.

Table 8: 2010-2011 MVP Voting Standing



Once again, this shows that Kobe Bryant is undervalued by the metric because of unimpressive advanced statistics. Players like Rajon Rondo and Tony Parker had lower values despite receiving MVP votes because they have unselfish playstyles which causes less impressive advanced statistics. They played on teams that had great team success which garnered them MVP points. Table 9 shows the relationship between VORP and value for the 2010-2011 season.

Table 9: 2010-2011 VORP and Value



This shows that the players with a higher VORP tend to have a higher value and the players with a lower VORP have a lower value. It’s also the first instance where the league leader in VORP did not win the league MVP. Chart 2 shows that Lebron James had better Total Stats and Win Contribution than the other evaluated players. It also shows that Derrick Rose had a well above average Win Contribution which may have contributed to his MVP win.

Chart 2: 2010-2011 Total Stats vs Win Contribution

* 1. 2011-2012 Results

The results of Peter Li’s model are shown in table 10. The results of the newer model are shown in table 11.

Table 10: Peter Li 2011-2012 Results

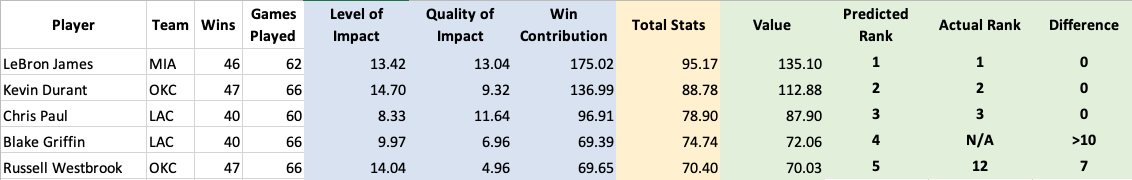


Table 11: 2011-2012 New Model Results



Lebron James has the highest value of this season and won MVP. According to John Hollinger, with a PER of 30.7 Lebron James is a Runaway MVP Candidate [12]. The top 3 players according to value remain the same with both models but the newer model has Josh Smith and Dwight Howard at 4 and 5 instead of Blake Griffin and Russell Westbrook. The results of this season are unique, and the values are lower than usual because this season was a lock-out shortened season and only 66 games were played instead of 82. Therefore, Josh Smith, who played all 66 games, appears to have an inflated value because he has higher total stats than average. Table 12 shows results organized by MVP standing.

Table 12: 2011-2012 MVP Voting Standing



This table shows how the lockout shortened season influenced the value. Kobe Bryant and Tony Parker were both on successful basketball teams, but their playstyles led to lower advanced statistics and therefore a lower value. Kobe Bryant is a high volume and low efficiency player which affected his value. Kobe Bryant also had a negative net rating which affected his Quality of Impact. Table 13 shows the relationship between value and VORP.

Table 13: 2011-2012 VORP and Value



The overvaluing of total statistics this season due to the lockout and players having less wins shows that there is less of a correlation between VORP and Value. However, Lebron James won the MVP this season, had the highest value, and the highest VORP which proves that despite the model’s lesser precision during lockout seasons, it will still highlight players who had exceptionally better seasons. Chart 3 demonstrates that Lebron deserved MVP because of his well above average Win Contribution and Total Stats.

Chart 3: 2011-2012 Total Stats vs Win Contribution

* 1. 2012-2013 Results

The results of Peter Li’s model are shown in table 14. The results of the newer model are shown in table 15.

Table 14: Peter Li 2012-2013 Results

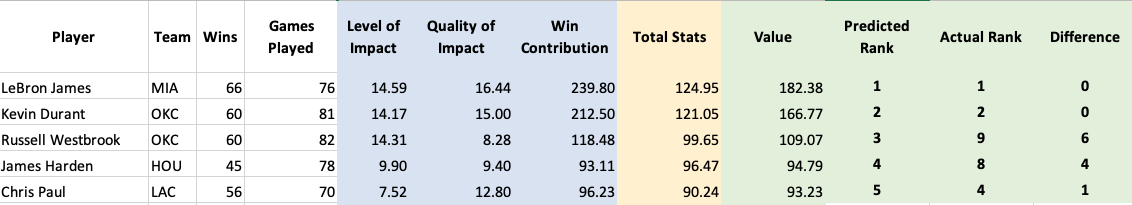


Table 15: 2012-2013 New Model Results



The results of the new model are like Peter Li’s except Chris Paul is valued higher with the new model. Chris Paul has a higher Player Efficiency Rating (26.4 vs 23) and Net Rating (25 vs 10) compared to James Harden. Russell Westbrook is undervalued by voters because he was on the same team as Kevin Durant, who had the better season. According to John Hollinger, with a PER of 31.6 Lebron James is a Runaway MVP Candidate [12]. Table 16 shows the results organized by MVP standing.

Table 16: 2012-2013 MVP Voting Standing



Players like Carmelo Anthony and Tony Parker are undervalued because of their playstyle. Tony Parker has an unselfish playstyle that helps his team win more games but does not lead to spectacular advanced statistics. Carmelo Anthony is a high volume and low efficiency player that had subpar advanced metrics (33rd in VORP and 14th in Win Shares). Kobe Bryant was fifth in MVP voting and 6th in value, showing that he had a much better season in terms of advanced metrics. Table 17 shows the relationship between value and VORP.

Table 17: 2012-2013 Value and VORP



Lebron James won the MVP, had the highest value, and had the highest VORP. This once again shows a general tendency of players with higher VORP having a higher value and players with a lower VORP having a lower value. Chart 4 shows that Lebron had the highest above average Total Stats vs Win Contribution compared to the average, although Kevin Durant was a close second.

Chart 4: 2012-2013 Total Stats vs Win Contribution

* 1. 2013-2014 Results

The results of Peter Li’s model are shown in table 18. The results of the newer model are shown in table 19.

Table 18: Peter Li 2013-2014 Results

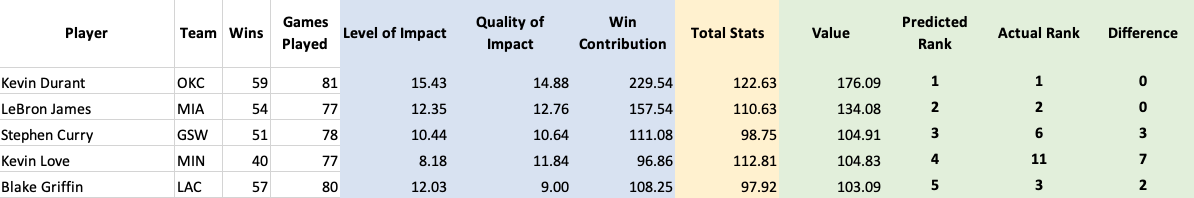


Table 19: 2013-2014 New Model Results



The top 5 players according to value are the same but the order is different. Kevin Durant had by far the best individual statistics of the season which was attributed to his increase in playmaking ability [22] Kevin Love had a great individual season and was 3rd in the league in VORP behind Lebron James and Kevin Durant. His team may not have had much success that season but most of their success can be attributed to the great season of Kevin Love. He had the fourth most total points scored and third most total rebounds during the 2013-2014 season. He was also 33rd in assists that season which was 6th most for forwards. If anything, this shows that the MVP voters highly undervalued the great season that Kevin Love had because Minnesota as a team had more losses than wins and missed the playoffs. Stephen Curry and Blake Griffin were the best players on their respective teams that season and were close in value because they had similar Total Stats and Win Contribution results. According to John Hollinger, with a PER of 29.8 Kevin Durant is a Strong MVP Candidate [12]. Table 20 shows the results organized by MVP Standing.

Table 20: 2013-2014 MVP Voting Standing



Stephen Curry’s higher individual statistics helped give him a higher value than James Harden. Joakim Noah put up great Total Stats and was top ten in the league for rebounds and blocks. However, he is the first instance of a player being undervalued by this metric because they are a great defensive player and there is currently a lack of defensive advanced metrics. He had a lowest usage (18.7%) amongst players who received MVP votes and averaged 12.6 points per game which meant he was not a focal point of the team’s offense. However, he was crucial to the team’s defense which was the reason he finished fourth in MVP voting. Table 21 shows the relationship between Value and VORP.

Table 21: 2013-2014 Value and VORP



The players with the highest VORP had the highest value, except for Blake Griffin. The continuing trend shows that there is a relationship between a player having higher VORP and having higher value. Chart 5 shows that Kevin Durant had the best season according to individual and team performance and was well deserving of the MVP.

Chart 5: 2013-2014 Total Stats vs Win Contribution

* 1. 2014-2015 Results

The results of Peter Li’s model are shown in table 22. The results of the newer model are shown in table 23.

Table 22: Peter Li 2014-2015 Results

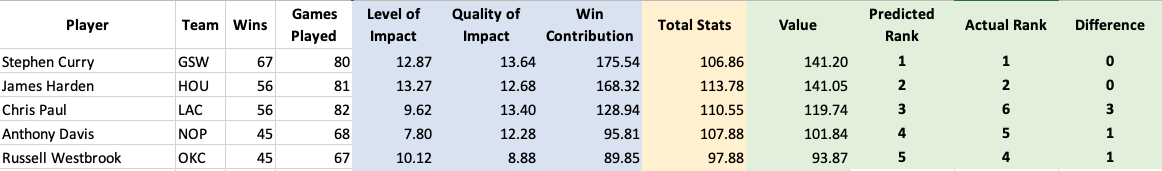


Table 23: 2014-2015 New Model Results



This was the first of the close margin seasons between the first and second place winners. James Harden had better Total Stats than Steph Curry however Steph Curry had better Win Contribution which led to his slightly higher value. In terms of points won in the actual MVP voting, Steph Curry had 1198 points and James Harden had 936 points meaning that the actual MVP race was close too. James Harden had better total box score stats, per 100 possessions statistics and better per 36 minutes statistics but Steph Curry had a higher player efficiency rating (28 vs 26.7) and a higher net rating (21 vs 15). The rest of the order of the table for the new metric is like Peter Li’s results except DeAndre Jordan has the fifth most value. He has a slightly higher value than Russell Westbrook which can be attributed to having a higher quality of impact due to a Net Rating of 28. DeAndre Jordan also led the league in rebounds and was third in the league in blocks which means that in this instance the metric was able to reward him for his defensive success. However, with Kevin Durant injured and Russell Westbrook leading the Oklahoma City Thunder to 45 wins, in this instance it appears that Peter Li’s metric was more accurate at predicting value. Although the margin of value is slim between Russell Westbrook and DeAndre Jordan, a perfect metric would display that Russell Westbrook had the higher value for that season. According to John Hollinger, with a PER of 28 Steph Curry is a Strong MVP Candidate [12]. Table 24 shows the results organized by MVP standing.

Table 24: 2014-2015 MVP Voting Standing



Lebron James and Russell Westbrook are the clear omissions from the top 5 in value even though they were in the top 4 in MVP Voting. Lebron had subpar advanced metrics for his standard (6th in VORP and 10th in Win Shares) which can be attributed to the sow start of the Cleveland Cavaliers due to team chemistry issues [17]. Table 25 shows the relationship between value and VORP for the season.

Table 25: 2014-2015 Value and VORP



The team success of the Los Angeles Clippers and DeAndre Jordan’s high Net Rating allowed him to have a high Value despite having a low VORP. Stephen Curry was the MVP and yet did not have the highest VORP which is the second instance so far of the player with the highest VORP not winning the MVP. In general, this still shows that players with a higher VORP have a higher Value. Chart 6 is interesting to analyze because the Value results were so close. Both Steph Curry and James Harden were well above the average total stats vs win contribution, but Steph Curry’s greater win contribution edged him out to have a higher value and win the MVP award.

Chart 6: 2014-2015 Total Stats vs Win Contribution

* 1. 2015-2016 Results

The results of Peter Li’s model are shown in table 26. The results of the newer model are shown in table 27.

Table 26: Peter Li 2015-2016 Results

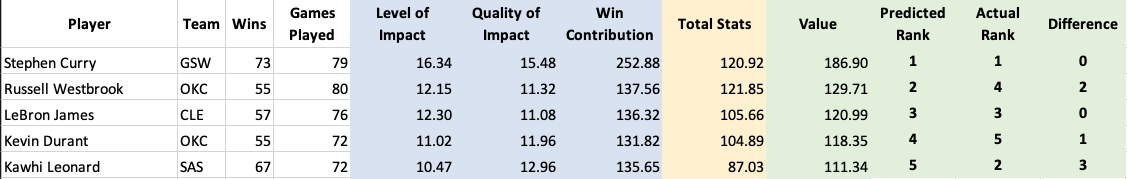


Table 27: 2015-2016 New Model Results



Stephen Curry is the clear MVP winner due to having a much higher value than the rest of the players. He helped lead the Golden State Warriors to an NBA record 73 wins. According to John Hollinger, with a PER of 31.5 Steph Curry is a Runaway MVP Candidate [12]. He also had the highest VORP of 9.5 and highest Win Shares of 17.9. With his historically great season and exceptional advanced stats, Steph Curry was a clear lock for MVP [3]. He would become the league’s first unanimous MVP. The top 5 Value players for the new model is slightly different than Peter Li’s model as the newer model values Kawhi Leonard and Lebron James less. This is the second instance of the model being imperfect because it underrepresents defensive players. Kawhi Leonard won the defensive player of the year so with better defensive metrics available his value may have been higher, and he may have been in the top 5 for Value. Kawhi Leonard had per game averages of 21.6 points per game, 6.8 rebounds per game, and 2.6 assists per game. These statistics are relatively unimpressive, but they are due to being on the San Antonio Spurs which is a team known for playing unselfish, team basketball that leads to relatively unimpressive advanced statistics for its players (Tony Parker in 2011-2012 and 2012-2013). Kawhi Leonard had a much better win contribution than James Harden however James Harden had much higher Total Stats which made him barely edge out Kawhi Leonard for fifth most Value. Kawhi Leonard helped lead the San Antonio Spurs to 67 wins which was the second-best record in the league. Ideally, an improved metric would include better advanced defensive metrics which would enhance his value. The newer model also has Kevin Durant edging out Lebron James in Value despite Lebron being the definitive best player on his team while Kevin Durant and Russell Westbrook were both great players sharing the spotlight for the Oklahoma City Thunder. This can be attributed to Kevin Durant’s higher quality of impact due to a higher VORP, Win Share, and Net Rating. Ultimately, Kevin Durant had a slightly higher Value than Lebron James and this is not a glaring issue with the new model. Table 28 shows the results organized by MVP Standing.

Table 28: 2015-2016 MVP Voting Standing



As discussed before, the metric undervalued the success of Kawhi Leonard, but the voters valued him appropriately according to his defensive success. Despite putting up impressive individual statistics, James Harden led the Houston Rockets to a record of 41 wins and 41 losses and so the MVP panel took that into account when voting. Table 29 shows the relationship between Value and VORP.

Table 29: 2015-2016 Value and VORP



As expected, with Steph Curry’s historic season he had the highest Value and the highest VORP. This once again shows that the players with a higher VORP had a higher Value. James Hard had the fourth highest VORP and so despite his teams’ mediocre season which lowered his level of impact, he had a high value. Chart 7 shows that Steph Curry had a significantly higher value than the rest of the players for the 2015-2016 season.

Chart 7: 2015-2016 Total Stats vs Win Contribution

* 1. 2016-2017 Results

The results of Peter Li’s model are shown in table 30. The results of the newer model are shown in table 31.

Table 30: Peter Li 2016-2017 Results

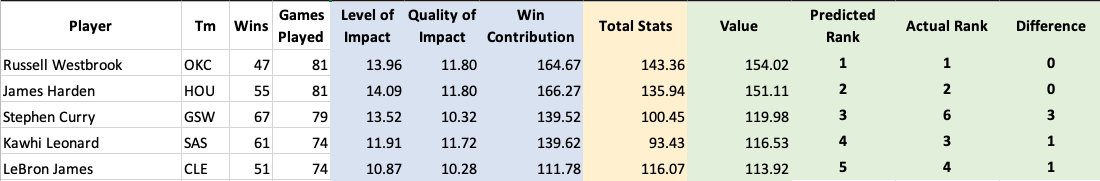


Table 31: 2016-2017 New Model Results



This was the second of the close margin seasons between the first and second place winners. In this instance however, Russell Westbrook had a significantly higher Total Stats and a lower Win Contribution, but he barely edged out James Harden to have the highest value. This season and the 2014-2015 season were the seasons that impacted the weightings of the Fantasy Basketball Stats Average equation. In the 2014-2015 season, Steph Curry had a higher value than James Harden because he had a higher win contribution. Harden had much higher total stats that season. In this seasons case, Russell Westbrook had a historically great individual season putting up 31.6 points per game, 10.4 assists per game, and 10.7 rebounds per game. Averaging a season long triple-double (double digits in three statistical categories; in this case points, rebounds, and assists) is a historic accomplishment and is arguably more difficult today compared to in the past [21]. This is a rare achievement, so Russell Westbrook won MVP. James Harden and the Houston Rockets won 55 games and finished third in the western conference while Russell Westbrook and the Oklahoma City Thunder only won 47 games and finished 6th in the Western Conference. It was difficult to adjust the weightings of the metric to justify Russell Westbrook winning MVP in 2016-2017 due to better individual statistics while James Harden lost the MVP in 2014-2015 despite having better individual statistics than Steph Curry. This was a tight MVP race and had a lot of analysts reevaluating what it means to be the most valuable player. James Harden succeeded and was important to his team because the Rockets were a team that was built around his talents. Russell Westbrook on the other hand insisted on doing everything to will his team to win and the Thunder built the team around that notion [4]. According to John Hollinger, with a PER of 30.6 Russell Westbrook is a Runaway MVP Candidate [12]. Kawhi Leonard was once again undervalued by the metric because even though he excelled defensively and had high value to his team’s success, this was not shown with him placing fifth in Value. Lebron James and Steph Curry had slightly more Value so with better defensive statistics available he may demonstrate a higher Value. The top 5 value players for this season in the new model are the same as in Peter Li’s model and its possible that a future iteration of the model will adjust the order of the Values to be the same as the MVP voting results. Table 32 shows the results organized by MVP Standing.

Table 32: 2016-2017 MVP Voting Standing



Isaiah Thomas had an impressive offensive season averaging 28.9 points per game however due to his size (5’9”) he was less impressive on defense which lowered his net rating and total rebounds for the season. The Boston Celtics were also the best team in the Eastern Conference due to their great overall team which meant that Isaiah Thomas had a lower VORP. Steph Curry may have been undervalued by voters although he put up great individual statistics and became the first unanimous MVP in the previous season. The Golden State Warriors added Kevin Durant to their roster in 2016-2017 and he has shown to have consistently high value over the last 10 years. As a result, Steph Curry’s value was underrated by voters. Table 33 shows the relationship between value and VORP.

Table 33: 2016-2017 Value and VORP



This is yet another season that displays the correlation between a high VORP and high Value. It is worth noting that Demar DeRozan received an MVP vote for being the best player on the Toronto Raptors that had 51 wins and finished 3rd in the eastern conference. However, his team’s overall talent made his VORP much lower. Chart 8 shows how well James Harden and Kawhi Leonard performed in terms of Win Contribution, but it is not as strong of an indicator as previous seasons at justifying the MVP.

Chart 8: 2016-2017 Total Stats vs Win Contribution

* 1. 2017-2018 Results

The results of Peter Li’s model are shown in table 34. The results of the newer model are shown in table 35.

Table 34: Peter Li 2017-2018 Results

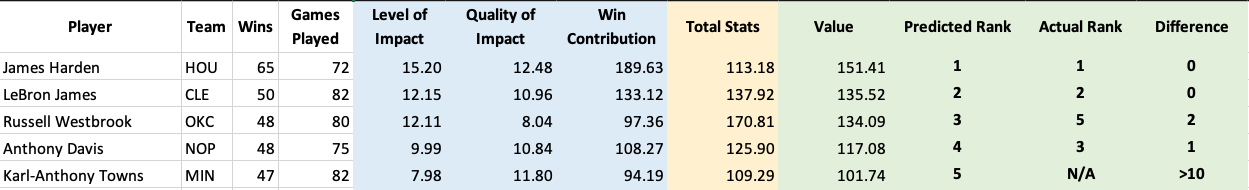


Table 35: 2017-2018 New Model Results



The new model has predicted the MVP correctly again and James Harden has finally won his first MVP while having the highest overall Value. James Harden had the highest win shares, the highest usage rate, the best PER, and led the Houston Rockets to 65 wins which was the most amount of wins that season. He was the best player on the best team which put him at the top of the MVP debate [14]. According to John Hollinger, with a PER of 29.8 James Harden is a Strong MVP Candidate. The model correctly predicted the players that received the top 3 MVP votes however Damian Lillard (4th in MVP votes) is a notable omission to the top 5 Value. Table 36 shows the results organized by MVP Standing.

Table 36: 2017-2018 MVP Voting Standing



Damian Lillard’s lower total stats is the reason he had the eighth highest value. He was a great scorer because he had the 6th most points in the league however, he was not as proficient at rebounding or assists compared to other notable guards such as James Harden or Russell Westbrook. The newer model benefits better all-around players that help the team in more ways than just scoring so Giannis Antetokounmpo, Kevin Durant, and Russell Westbrook all had higher values. Damian Lillard had the third highest VORP in the league and led his team to the third best record in the western conference so his value may have been higher with better box score statistics than high points scored. Table 37 shows the relationship between VORP and Value.

Table 37: 2017-2018 Value and VORP



This is another instance where the player with the best VORP (Lebron James) did not win the MVP. This does continue to demonstrate that players with a high VORP will have a high Value. Chart 9 shows that James Harden’s win contribution is far greater than the other players and this combined with his exceptional individual statistics is the reason that he won MVP in the 2017-2018 season.

Chart 9: 2017-2018 Total Stats vs Win Contribution

* 1. 2018-2019 Results

The results of Peter Li’s model are shown in table 38. The results of the newer model are shown in table 39.

Table 38: Peter Li 2018-2019 Results

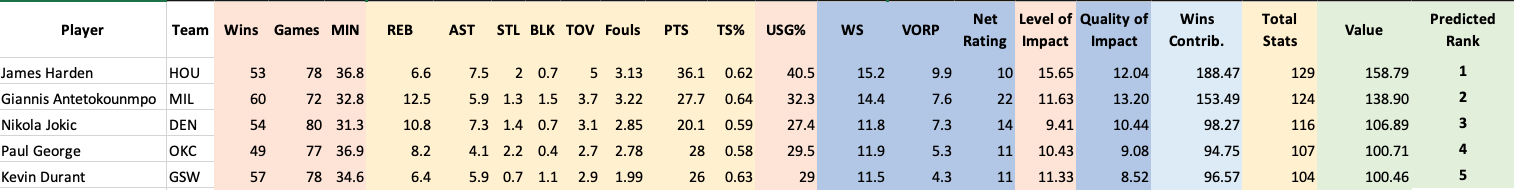


Table 39: 2018-2019 New Model Results



Both models did not correctly predict the results of the MVP vote. James Harden had better total stats and a better win contribution. He had the highest VORP, the second highest PER, the highest usage, and the highest win shares. The reason he did not win MVP may be explained in a few reasons. The first reason is voter fatigue. Lebron James is widely considered the best player in the NBA (7 finals appearances in the last decade) but the reason he does not win the MVP every single year is arguably because it would be less interesting to award the MVP to the same player each year. The results so far have shown two instances of Lebron James having the highest VORP and not winning MVP. James Harden won the MVP in the 2017-2018 season, so he was possibly a less desirable winner. The second reason is his “unfavorable” playstyle. James Harden is an exceptional scorer and in the 2018-2019 season he scored 2,818 points which is 659 more points (30.5% greater) than the second-best scorer (Paul George). However, his playstyle relies on drawing personal fouls on the defender and getting to the free throw line. He had the most free throw attempts in the league with 858 attempts which was 172 more attempts than Giannis Antetokounmpo who had the second most attempts. This playstyle was “bad” to some people because it was boring to watch the game stop so frequently so that James Harden could shoot free throws [20]. The third reason is that this metric undervalues defensive ability. This is the third instance in which a defensive player would greatly benefit from more advanced defensive metrics to show their value. Giannis had the twelfth most blocks in the league, the sixth most rebounds, the tenth best defensive rating, the best defensive box-plus minus, and the third highest win shares. These statistics may favor “big men” at times whose height allow them to have more blocks and make shots more difficult. Giannis is no exception to this however he is a prolific athlete and a great scorer (7th most total points). Giannis also had a lower VORP than James Harden because his team was better all around. Table 40 shows the results organized by MVP Standing.

Table 40: MVP Voting Standing



Steph Curry was the best player on the best team in the western conference however Kevin Durant is another exceptional player on that team who has a lot of Value. Therefore, Steph Curry’s Total Stats were lower than players who were by far the most valuable players on their team and as a result his Value results were lower. Paul George is another player on this list who is underappreciated by his Value results because he was a great defensive player (5th in defensive win shares) during this season. He also had the second most points scored that season however his team was not as successful being the 6th best team in the western conference. Nikola Jokic was on the second-best team in the western conference, had the second highest VORP, and the fourth highest Player Efficiency Rating which is why he had the third highest Value. Table 41 shows the relationship between VORP and Value.

Table 41: 2018-2019 Value and VORP



The players with higher VORP are the best players who typically may not have the strongest supporting cast on their team. Giannis Antetokounmpo has the second highest VORP despite having a strong team (1st in eastern conference with 60 wins). The trend over the last decade clearly demonstrates a relationship between players having a high VORP and having a high value, and players with a high VORP winning the MVP. According to Chart 10, James Harden and Giannis have much higher values than the rest of the top players in the league.

Chart 10: 2018-2019 Total Stats vs Win Contribution

* 1. Predicting the 2019-2020 NBA MVP

The prediction of the 2019-2020 MVP was done using the new model. Currently, the season is suspended due to the outbreak of COVID-19 however the current MVP Odds and player statistics may be used to predict the MVP winner at this point [1]. Based on table 42, it can be predicted that Giannis Antetokounmpo will win the MVP for the second year in a row. He currently has the highest Value, the highest Win Contribution, and the highest Total Stats.

Table 42: 2019-2020 Predicted MVP Voting Standing



* 1. Decade Results

Table 43 shows the players with the top 20 Value over the last ten years. The only two players that won an MVP and are not in this table are Derrick Rose in 2010-2011 and Lebron James in 2011-2012. 2011-2012 was the lockout season and every player had a lower value because of having lower total stats. Derrick Rose was arguably the weakest MVP winner of the last decade and his winning of MVP was heavily influenced by narrative. Russell Westbrook had the highest Total Stats of the decade during his 2016-2017 MVP season and Steph Curry had the highest Win Contribution during his 2015-2016 MVP Season when he led the Golden State Warriors to an NBA record 73 wins and 9 losses. Lebron James accounts for 25% of the highest Value seasons over the last decade. Stephen Curry is the “MVP of the decade” with a Value of 272.13472. Chart 12 shows Total Stats vs Win Contribution for all the best players over the last decade. Each individual player may be identified by hovering over the data point.

Table 43: Decade Value Results for New Model



Chart 12: Decade Total Stats vs Win Contribution

1. Conclusion and Future Research

5.1 Conclusions

The new model correctly predicted the MVP in 8 out of the last 10 seasons so the prediction of Giannis winning the MVP in 2020 is the best prediction to make at this time. This is the same amount of times as Peter Li’s model however the newer model is slightly more accurate because it considers more player statistics and eliminates arbitrary scalar values. The new model accurately predicted the top 5 players that received MVP votes 35/50 times. The reason for this “low” success rate is because there are intangible factors and biases that contribute to which players the MVP committee votes. Also, there are not enough defensive metrics available to add more value to better defensive players. Sometimes a player will receive more MVP votes due an impressive defensive season and the metric in its current form does not do justice to these players. The newer model adjusted the weighting of the Net Rating to benefit players who had great defensive ratings, but this did not appear to have a major impact on results. It helped increase the value of “big men” that had great defensive seasons such as Dwight Howard in 201-2011, Kevin Love in 2013-2014, DeAndre Jordan in 2014-2015, and Rudy Gobert in 2018-2019. The model was made by parsing data from Excel files and creating new tables of results in new Excel files. This took a significant amount of time to process a lot of data and so there is room for improvement in modifying the way that the model reads and processes data.

* 1. Future Research

The model was evaluated for accuracy using the players who received MVP votes but also by using players who received All-NBA honors. The model could also be tested in the future in predicting the accuracy of players that received All-NBA votes. This may be more difficult because the All-NBA first team must have a player for each of the five positions: two guards, two forwards, and a center. Some players get All-NBA honors by identifying under a position that they do not normally play. For example, in the 2017-2018 season Anthony Davis made the All-NBA first team as a center although he typically plays the power forward position.

Another change for the model is to refactor it to make it more maintainable or easier to add new functionality. The model should be reengineered to scrape the data directly from basketball-reference.com. Also, the model in its current form takes advanced metrics such as Player Efficiency Rating and Win Shares directly from basketball-reference.com. If the model could calculate these values independently then the players Values could be updated in real time to account for any time a player scores in game, gets a rebound, or earns an assist. As previously discussed, with newer defensive advanced metrics created in the future the model could be further modified to add more value to better defensive players. Also, if possible, the model could add numbers or multipliers that correspond to certain intangibles that cause bias in voting. This includes narrative from fan appeal or possible voter fatigue. This will be much more difficult to master and will not be consistent from season to season. It is more useful to take the results of a player’s value and understand that just because a player is having an incredible season individually that the media members may be weighting value different. If a significant amount of media members also release how they measure a player’s value when voting, then the model may be adjusted accordingly. This may be possible with a poll of the media members on the MVP voting panel however media members may be hesitant to release their methods and justifications due to fear of negative public backlash. In the 2012-2013 season, Lebron James was one first place vote away from being the unanimous MVP as one voter submitted a first-place vote for Carmelo Anthony. It is highly likely that this voter would like to remain anonymous. The method of calculating the model in Python could surely be improved to make the performance better however the model could be adjusted in the future to follow trends in the criticality of certain advanced statistics. Currently. Player Efficiency Rating and Value Over Replacement Player are widely accepted metrics however newer metrics may become more widely accepted in the future and the model may be adjusted accordingly.

1. References

1. “2019-20 NBA MVP Award Tracker.” *Basketball Reference*, [www.basketball-reference.com/friv/mvp.html](http://www.basketball-reference.com/friv/mvp.html).

2. Beardsley, Bove. “Winning with Data Science, Golden State Warriors Style.” *Dataconomy*, 3 July 2017, dataconomy.com/2017/07/golden-state-warriors-data-science/.

3. Buckley, Zach. “Would Stephen Curry Be the NBA's MVP If He Didn't Play Another Game This Season?” *Bleacher Report*, Bleacher Report, 9 Sept. 2017, bleacherreport.com/articles/2622624-would-stephen-curry-be-the-nbas-mvp-if-he-didnt-play-another-game-this-season.

4. Chau, Danny. “Where You Stand in the Westbrook-Harden MVP Debate Says a Lot About You.” *The Ringer*, The Ringer, 31 Mar. 2017, www.theringer.com/2017/3/31/16045480/russell-westbrook-james-harden-nba-mvp-a166d91ff6cf.

5. Darcy, Kieran. “NBA Goes Fantasy-Friendly with New Official Scoring System.” *ESPN*, ESPN Internet Ventures, 4 Oct. 2017, www.espn.com/fantasy/basketball/story/\_/id/20916781/fantasy-basketball-nba-embraces-fantasy-basketball-introduces-official-scoring-system.

5. “Expanded Glossary of Basketball Stats.” *Alone In The Green Room*, 10 Oct. 2015, aloneinthegreenroom.wordpress.com/statsketball/glossary-of-basketball-stats/##unique-identifier.

6. “Game Score.” *Stat Lab*, www.nba.com/resources/static/team/v2/thunder/statlab-gamescore-191201.pdf.

7. “Glossary.” *Basketball Reference*, [www.basketball-reference.com/about/glossary.html](http://www.basketball-reference.com/about/glossary.html).

8. Jackson, Scoop. “All Hail Lebron James, the Villain.” *ESPN*, 13 Jan. 2011, www.espn.com/espn/commentary/news/story?page=jackson/110113.

9. Jacobs, Justin. “Breaking Down Player Efficiency Rating.” *Squared Statistics: Understanding Basketball Analytics*, 2 Sept. 2017, squared2020.com/2017/09/01/breaking-down-player-efficiency-rating/.

10. “Land Of Basketball.com.” *NBA Season MVP Awards - Summary*, www.landofbasketball.com/awards/nba\_season\_mvps.htm.

11. Li, Peter. “NBA MVP Prediction Model.” *Medium*, Towards Data Science, 20 Apr. 2019, towardsdatascience.com/nba-mvp-predictor-c700e50e0917.

12. Maroun, Eric. “Understanding Advanced Statistics: Player Efficiency Rating.” *Hardwood Paroxysm*, 7 Mar. 2012, web.archive.org/web/20170910105350/hardwoodparoxysm.com/2012/03/07/understanding-advanced-statistics-player-efficiency-rating/.

13. Myers, Daniel. “About Box Plus/Minus (BPM).” *Basketball Reference*, Feb. 2020, [www.basketball-reference.com/about/bpm2.html](http://www.basketball-reference.com/about/bpm2.html).

14. O'Connor, Kevin. “The MVP Race Is Over, and It's James Harden in a Landslide.” *The Ringer*, The Ringer, 9 Apr. 2018, www.theringer.com/nba/2018/4/9/17213702/james-harden-mvp-all-nba.

15. “Player Comparison Finder.” *Basketball Reference*, www.basketball-reference.com/play-index/pcm\_finder.fcgi.

16. Price, Marc. “Advanced Stats in Basketball: An Explainer Series.” *Medium*, Medium, 29 Mar. 2016, [medium.com/@mrprice33/advanced-stats-in-basketball-an-explainer-series-15cc1668da78](mailto:medium.com/@mrprice33/advanced-stats-in-basketball-an-explainer-series-15cc1668da78).

17. Rose, Bryan. “LeBron James Frustrated With Cleveland Cavaliers Poor Play.” *FanSided*, FanSided, 22 Nov. 2014, fansided.com/2014/11/22/lebron-james-frustrated-cleveland-cavaliers-poor-play/.

18. Slowinski, Steve. “What Is WAR?” *What Is WAR? | Sabermetrics Library*, 15 Feb. 2010, library.fangraphs.com/misc/war/.

19. “Understanding Basketball Analytics - Net Rating.” *PivotAnalysis*, PivotAnalysis, 1 May 2019, www.pivotanalysis.com/post/net-rating.

20. Vale, Bryan. “James Harden Plays Morally Deficient Basketball.” *Medium*, Medium, 11 Nov. 2018, medium.com/@bryanvale\_4924/james-harden-plays-morally-deficient-basketball-d742819fcdbd.

21. Wagner, Kyle. “The Case For Russell Westbrook, MVP.” *FiveThirtyEight*, FiveThirtyEight, 14 Apr. 2017, fivethirtyeight.com/features/the-case-for-russell-westbrook-mvp/.

22. Ziller, Tom. “How Kevin Durant Took the MVP from LeBron James.” *SBNation.com*, SBNation.com, 14 Apr. 2014, [www.sbnation.com/2014/4/14/5612662/kevin-durant-nba-mvp-2014-lebron-james](http://www.sbnation.com/2014/4/14/5612662/kevin-durant-nba-mvp-2014-lebron-james).

1. Appendix

Python Program:

'''Author: Robert Shea

Code Purpose: The purpose of this code is to predict the Most Valuable Player for a season in the National Basketball Association. The program will parse statistics from

various Excel files that are organized by season. The sheets in the Excel files are organized according to different types of statistics to be parsed. The goal is to create

the most accurate model at predicting a players value using statistics that are available from basketball-reference.com'''

import pandas as pd #Libraries that are imported to read and organize data

from pandas import ExcelWriter

from pandas import ExcelFile

'''import seaborn as sns

sns.set(color\_codes=True)''' #There were issues in getting this library to work properly and output a scatter chart

class Excel\_Data\_Reader:

        '''This class will read an excel file, parse each individual player to be evaluated, then parse the respective statistics for each player. Then the statistics/data will be

        used as variables in various equations to understand a players Value when determining whether an NBA player should win the MVP award. Lastly, various tables and graphs will

        be created to display the results of calculations'''

        def \_\_init\_\_(self, path, sheet\_name):

                self.path = path #Excel file

                self.sheet\_name = sheet\_name

                self.players = dict() #Stores each instance of a player to be evaluated using the model

                self.season = []

                self.analyze\_excel\_file()

                self.table() #This function is for parsing Excel files by a single season

                #self.decade\_table() #This function will parse multiple Excel files and output a comprehensive table of Value over the last decade

                #self.chart()

        def file\_reader(self, path, sheet\_name):

                '''This function will try to read an excel file and return an error if the Excel file can not be found or if it does not exist.'''

                try:

                        df = pd.read\_excel(path, sheet\_name= sheet\_name)

                        return df

                except FileNotFoundError:

                        raise FileNotFoundError(f"Can not open file {path}!")

        def analyze\_excel\_file(self):

                '''This function starts the analysis of an Excel file'''

                player = ''

                for files in self.path: #Useful for when there is more than one file to be analyzed

                        df = self.file\_reader(files, 'Per Game')

                        self.season.append(files.strip('.xslx')) #The current season will be appended and added to a list. The excel files are conveniently named according to season

                        for n in df.columns: #Reads each of the first columns in the excel sheet to find the column labeled 'Player'

                                if n == 'Player':

                                        for i in df.index: #Goes through the values of the column that is labelled 'Player' to collect the player names of each player to be analyzed and create a new dictionary instance

                                                player = df[n][i] + ' ' + files.strip('.xslx')

                                                self.players[player] = Players() #A new dictionary key is created for each player

                                                self.players[player].player\_name = (df[n][i]).strip('\*') #Removes an asterisk for instances where the player is a Hall of Famer

                                                self.players[player].player\_index = i

                                                for sheet in self.sheet\_name: #Parses the stats by going from sheet to sheet in the excel file and determining which stats are needed from each sheet

                                                        self.parse\_stats(files, sheet, player)

                                        '''These are the different functions for calculating different results using the parsed statistics as variables'''

                                                self.fantasy\_basketball\_stats\_totals(player)

                                                self.fantasy\_basketball\_stats\_per\_100\_poss(player)

                                                self.fantasy\_basketball\_stats\_per\_36\_min(player)

                                                self.fantasy\_basketball\_stats\_average(player)

                                                self.game\_score(player)

                                                self.total\_stats(player)

                                                self.net\_rating(player)

                                                self.quality\_of\_impact(player)

                                                self.level\_of\_impact(player)

                                                self.win\_contribution(player)

                                                self.value(player)

        def parse\_stats(self, files, sheet, player):

                '''The function for parsing statistics for each player. Each instance of player is evaluated using this function. Also, this takes into account

                that more than one sheet should be analyzed and sometimes more than one file may be analyzed. Each statistic, except the Season, will be stored as a float

                so that it may be used as a variable in an equation.'''

                df = self.file\_reader(files, sheet)

                if sheet in ['Per Game', 'Totals', 'Advanced','Per 100 Poss', 'Per 36 Min']: #These sheets contain specific individual statistics that are to be stored as values for each instance in the players dictionary

                        for n in df.columns:

                                if n == 'Season': #Stores the season for the player

                                        self.players[player].season = str(df[n][self.players[player].player\_index])

                                elif n == 'G': #Stores the total games played by the player

                                        self.players[player].games\_played = float(df[n][self.players[player].player\_index])

                                elif n == 'MP' and sheet == 'Per Game': #Stores the minutes played per game by the player

                                        self.players[player].minutes\_per\_game = float(df[n][self.players[player].player\_index])

                                elif n == 'PER': #Stores each players Player Efficiency Rating as found from the 'Advanced' sheet

                                        self.players[player].player\_efficiency\_rating = float(df[n][self.players[player].player\_index])

                                elif n == 'TS%': #Stores each players True Shooting Percentage as found from the 'Advanced' sheet

                                        self.players[player].true\_shooting\_percentage = float(df[n][self.players[player].player\_index])

                                elif n == 'USG%': #Stores each players Usage Rate as found from the 'Advanced' sheet

                                        self.players[player].usage = float(df[n][self.players[player].player\_index])

                                elif n == 'WS': #Stores each players Win Shares as found from the 'Advanced' sheet

                                        self.players[player].Win\_Share = float(df[n][self.players[player].player\_index])

                                elif n == 'VORP': #Stores each players Value Over Replacement Player as found from the 'Advanced' sheet

                                        self.players[player].VORP = float(df[n][self.players[player].player\_index])

                                elif n == 'ORtg': #Stores each players Offensive Rating as found from the 'Per 100 Poss' sheet

                                        self.players[player].offensive\_rating = float(df[n][self.players[player].player\_index])

                                elif n == 'DRtg': #Stores each players Defensive Rating as found from the 'Per 100 Poss' sheet

                                        self.players[player].defensive\_rating = float(df[n][self.players[player].player\_index])

                                elif n == 'FG' and sheet == 'Per Game': #The field goals made are taken from the 'Per Game' sheet for the Game Score calculation

                                        self.players[player].FGM\_per\_game = float(df[n][self.players[player].player\_index])

                                elif n == 'FGA' and sheet == 'Per Game': #The field goals attempted are taken from the 'Per Game' sheet for the Game Score calculation

                                        self.players[player].FGA\_per\_game = float(df[n][self.players[player].player\_index])

                                elif n == 'FT' and sheet == 'Per Game': #The free throws made are taken from the 'Per Game' sheet for the Game Score calculation

                                        self.players[player].FTM\_per\_game = float(df[n][self.players[player].player\_index])

                                elif n == 'FTA' and sheet == 'Per Game': #The free throws attempted are taken from the 'Per Game' sheet for the Game Score calculation

                                        self.players[player].FTA\_per\_game = float(df[n][self.players[player].player\_index])

                                elif n == 'ORB' and sheet == 'Per Game': #The offensive rebounds are taken from the 'Per Game' sheet for the Game Score calculation

                                        self.players[player].ORB\_per\_game = float(df[n][self.players[player].player\_index])

                                elif n == 'DRB' and sheet == 'Per Game': #The defensive rebounds are taken from the 'Per Game' sheet for the Game Score calculation

                                        self.players[player].DRB\_per\_game = float(df[n][self.players[player].player\_index])

                                elif n == 'TRB' and sheet == 'Totals': #The total rebounds (offensive rebounds + defensive rebounds) are taken from three different sheets for the Fantasy Basketball Stats Calculation

                                        self.players[player].total\_rebounds = float(df[n][self.players[player].player\_index])

                                elif n == 'TRB' and sheet == 'Per 100 Poss':

                                        self.players[player].rebounds\_per\_100\_poss = float(df[n][self.players[player].player\_index])

                                elif n == 'TRB' and sheet == 'Per 36 Min':

                                        self.players[player].rebounds\_per\_36\_min = float(df[n][self.players[player].player\_index])

                                elif n == 'AST' and sheet == 'Per Game': #The assists are taken from four different sheets for the Fantasy Basketball Stats Calculation and Game Score calculation

                                        self.players[player].assists\_per\_game = float(df[n][self.players[player].player\_index])

                                elif n == 'AST' and sheet == 'Totals':

                                        self.players[player].total\_assists = float(df[n][self.players[player].player\_index])

                                elif n == 'AST' and sheet == 'Per 100 Poss':

                                        self.players[player].assists\_per\_100\_poss = float(df[n][self.players[player].player\_index])

                                elif n == 'AST' and sheet == 'Per 36 Min':

                                        self.players[player].assists\_per\_36\_min = float(df[n][self.players[player].player\_index])

                                elif n == 'STL' and sheet == 'Per Game': #The steals are taken from four different sheets for the Fantasy Basketball Stats Calculation and Game Score calculation

                                        self.players[player].steals\_per\_game = float(df[n][self.players[player].player\_index])

                                elif n == 'STL' and sheet == 'Totals':

                                        self.players[player].total\_steals = float(df[n][self.players[player].player\_index])

                                elif n == 'STL' and sheet == 'Per 100 Poss':

                                        self.players[player].steals\_per\_100\_poss = float(df[n][self.players[player].player\_index])

                                elif n == 'STL' and sheet == 'Per 36 Min':

                                        self.players[player].steals\_per\_36\_min = float(df[n][self.players[player].player\_index])

                                elif n == 'BLK' and sheet == 'Per Game': #The blocks are taken from four different sheets for the Fantasy Basketball Stats Calculation and Game Score calculation

                                        self.players[player].blocks\_per\_game = float(df[n][self.players[player].player\_index])

                                elif n == 'BLK' and sheet == 'Totals':

                                        self.players[player].total\_blocks = float(df[n][self.players[player].player\_index])

                                elif n == 'BLK' and sheet == 'Per 100 Poss':

                                        self.players[player].blocks\_per\_100\_poss = float(df[n][self.players[player].player\_index])

                                elif n == 'BLK' and sheet == 'Per 36 Min':

                                        self.players[player].blocks\_per\_36\_min = float(df[n][self.players[player].player\_index])

                                elif n == 'TOV' and sheet == 'Per Game': #The turnovers are taken from four different sheets for the Fantasy Basketball Stats Calculation and Game Score calculation

                                        self.players[player].turnovers\_per\_game = float(df[n][self.players[player].player\_index])

                                elif n == 'TOV' and sheet == 'Totals':

                                        self.players[player].total\_turnovers = float(df[n][self.players[player].player\_index])

                                elif n == 'TOV' and sheet == 'Per 100 Poss':

                                        self.players[player].turnovers\_per\_100\_poss = float(df[n][self.players[player].player\_index])

                                elif n == 'TOV' and sheet == 'Per 36 Min':

                                        self.players[player].turnovers\_per\_36\_min = float(df[n][self.players[player].player\_index])

                                elif n == 'PF' and sheet == 'Per Game': #The personal fouls are taken from four different sheets for the Fantasy Basketball Stats Calculation and Game Score calculation

                                        self.players[player].fouls\_per\_game = float(df[n][self.players[player].player\_index])

                                elif n == 'PF' and sheet == 'Totals':

                                        self.players[player].total\_fouls = float(df[n][self.players[player].player\_index])

                                elif n == 'PF' and sheet == 'Per 100 Poss':

                                        self.players[player].fouls\_per\_100\_poss = float(df[n][self.players[player].player\_index])

                                elif n == 'PF' and sheet == 'Per 36 Min':

                                        self.players[player].fouls\_per\_36\_min = float(df[n][self.players[player].player\_index])

                                elif n == 'PTS' and sheet == 'Per Game': #The points scored are taken from four different sheets for the Fantasy Basketball Stats Calculation and Game Score calculation

                                        self.players[player].points\_per\_game= float(df[n][self.players[player].player\_index])

                                elif n == 'PTS' and sheet == 'Totals':

                                        self.players[player].total\_points= float(df[n][self.players[player].player\_index])

                                elif n == 'PTS' and sheet == 'Per 100 Poss':

                                        self.players[player].points\_per\_100\_poss = float(df[n][self.players[player].player\_index])

                                elif n == 'PTS' and sheet == 'Per 36 Min':

                                        self.players[player].points\_per\_36\_min = float(df[n][self.players[player].player\_index])

                elif sheet == 'MVP Voting':

                        '''The sheet named 'MVP Voting' contains the results of the MVP voting for the season and each player that received a vote will have their ranking (according to votes received)

                        stored as a value for each instance of player'''

                        for n in df.columns:

                                for i in df.index:

                                        if n == 'Player' and self.players[player].player\_name in df[n][i]: #The 'Player' column in the sheet is first found and the value for players name is cross checked with the list of players to detect if that player received MVP votes

                                                if str(df['Rank'][i]).endswith('T'): #If the Rank ends in T it means the player tied and this will remove the T

                                                        self.players[player].MVP\_vote\_standing = int(df['Rank'][i].strip('T'))

                                                else:

                                                        self.players[player].MVP\_vote\_standing = int(df['Rank'][i])

                elif sheet == 'MVP Tracker':

                        '''The sheet named 'MVP Tracker' contains the current odds for players that are in the MVP conversation.

                The players are ranked on probability of winning MVP and the ranking is parsed to detect if the player with the highest Value calculated from the model is the same

                player that is the frontrunner for MVP.'''

                        for n in df.columns:

                                for i in df.index:

                                        if n == 'Player' and self.players[player].player\_name in df[n][i]:

                                                if str(df['Rk'][i]).endswith('T'):

                                                        self.players[player].MVP\_vote\_standing = int(df['Rk'][i].strip('T'))

                                                else:

                                                        self.players[player].MVP\_vote\_standing = int(df['Rk'][i])

                elif sheet in ['ATL', 'BKN', 'BOS', 'CHA', 'CHI', 'CLE', 'DAL', 'DEN', 'DET','GS', 'HOU', 'IND', 'LAC', 'LAL', 'MEM', 'MIA', 'MIL', 'MIN', 'NO', 'NY', 'OKC', 'ORL', 'PHI', 'PHX', 'POR', 'SA', 'SAC', 'TOR', 'UTAH', 'WSH']:

                        '''Used to figure out how many wins a player had in a season. The number of wins is used for the Level of Impact calculation.

                        This is not the most efficienct way of determining how many wins a player has in a season but it is easy to understand and implement.'''

                        for n in df.columns:

                                for i in df.index:

                                        if n == 'Player' and self.players[player].player\_name in df[n][i]: '''Each team sheet has a roster of the players on that team for the season. If the player is on that roster then the wins associated with that team will be added as a value for that instance of player'''

                                                self.players[player].wins = int(df['Wins'][0])

        def fantasy\_basketball\_stats\_totals(self, player):

                '''Function to calculate Fantasy Basketball Stats Totals based on a players total box score statistics'''

                player = self.players[player]

                player.fantasy\_basketball\_stats\_totals = ((player.total\_points)\*player.true\_shooting\_percentage) + 1.5\*(player.total\_assists) + 1.2\*(player.total\_rebounds) + 3\*(player.total\_blocks) + 3\*(player.total\_steals) - player.total\_fouls - player.total\_turnovers

                return player.fantasy\_basketball\_stats\_totals

        def fantasy\_basketball\_stats\_per\_36\_min(self, player):

                '''Function to calculate Fantasy Basketball Stats Per 36 Minutes based on a players per 36 minutes box score statistics'''

                player = self.players[player]

                player.fantasy\_basketball\_stats\_per\_36\_min = ((player.points\_per\_36\_min)\*player.true\_shooting\_percentage) + 1.5\*(player.assists\_per\_36\_min) + 1.2\*(player.rebounds\_per\_36\_min) + 3\*(player.blocks\_per\_36\_min) + 3\*(player.steals\_per\_36\_min) - player.fouls\_per\_36\_min - player.turnovers\_per\_36\_min

                return player.fantasy\_basketball\_stats\_per\_36\_min

        def fantasy\_basketball\_stats\_per\_100\_poss(self, player):

                '''Function to calculate Fantasy Basketball Stats Per 100 Possessions based on a players per 100 possessions box score statistics'''

                player = self.players[player]

                player.fantasy\_basketball\_stats\_per\_100\_poss = ((player.points\_per\_100\_poss)\*player.true\_shooting\_percentage) + 1.5\*(player.assists\_per\_100\_poss) + 1.2\*(player.rebounds\_per\_100\_poss) + 3\*(player.blocks\_per\_100\_poss) + 3\*(player.steals\_per\_100\_poss) - player.fouls\_per\_100\_poss - player.turnovers\_per\_100\_poss

                return player.fantasy\_basketball\_stats\_per\_100\_poss

        def fantasy\_basketball\_stats\_average(self, player):

                '''Takes an average for the three different calculations of Fantasy Basketball Statistics. The calculations have different weightings because

                the total box score statistics would heavily skew the results if it was weighted significantly.'''

                player = self.players[player]

                player.fantasy\_basketball\_stats\_average = .275\*(player.fantasy\_basketball\_stats\_totals)+ .3625\*(player.fantasy\_basketball\_stats\_per\_100\_poss) + .3625\*(player.fantasy\_basketball\_stats\_per\_36\_min)

                return player.fantasy\_basketball\_stats\_average

        def game\_score(self, player):

                '''Calculation  of Game Score, a statistic that was created by John Hollinger to give an idea fo a player's productivity for a single game.

                The weightings are widely accepted so the equation was used with per game box score statistics'''

                player = self.players[player]

                player.game\_score = (player.points\_per\_game + .4\*(player.FGM\_per\_game) - .7\*(player.FGA\_per\_game) - .4\*(player.FTA\_per\_game - player.FTM\_per\_game) + .7\*(player.ORB\_per\_game) + .3\*(player.DRB\_per\_game) + player.steals\_per\_game + .7\*(player.assists\_per\_game) + .7\*(player.blocks\_per\_game) - .4\*(player.fouls\_per\_game) - player.turnovers\_per\_game)

                return player.game\_score

        def total\_stats(self, player):

                '''Calculation of Total Stats: the first component of determining a players Value'''

                player = self.players[player]

                player.total\_stats = (.3\*(player.fantasy\_basketball\_stats\_average) + .3\*(player.game\_score) + .4\*(player.player\_efficiency\_rating))

                return player.total\_stats

        def net\_rating(self, player):

                '''Calculation of Net Rating by subtracting a players offensive rating from their defensive rating. Used in the calculation of Quality of Impact'''

                player = self.players[player]

                player.net\_rating = player.offensive\_rating - player.defensive\_rating

                return player.net\_rating

        def quality\_of\_impact(self, player):

                '''Calculation of Quality of Impact: the first component of Win Contribution'''

                player = self.players[player]

                player.quality\_of\_impact = .35\*(player.Win\_Share) + .35\*(player.VORP) + .3\*(player.net\_rating)

                return player.quality\_of\_impact

        def level\_of\_impact(self, player):

                '''Calculation of Level of Impact: the second component of Win Contribution'''

                player = self.players[player]

                player.level\_of\_impact = player.wins\*(player.games\_played/82)\*(player.minutes\_per\_game/48)\*(player.usage/100)

                return player.level\_of\_impact

        def win\_contribution(self, player):

                '''Calculation of Win Contribution: The second component of Value'''

                player = self.players[player]

                player.win\_contribution = player.quality\_of\_impact \* player.level\_of\_impact

                return player.win\_contribution

        def value(self, player):

                '''Calculation of a players Value'''

                player = self.players[player]

                player.value = .5\*(player.total\_stats) + .5\*(player.win\_contribution)

                return player.value

        def table(self):

                '''This function will create tables and a scatter chart for whatever season is being evaluated. The method of creating the tables is not the most efficient but

                it is simple and works properly.'''

                for season in self.season:

                        '''These lists will store values according to each instance of player in the order that they are evaluated. The values are appended to the end

                        of the list so that when the lists need to be evaluated, the players statistics are in the same index spot in each list'''

                        total\_player\_name = []

                        total\_season = []

                        total\_fantasy\_basketball\_stats\_average = []

                        total\_game\_score = []

                        total\_player\_efficiency\_rating = []

                        total\_total\_stats = []

                        total\_VORP = []

                        total\_quality\_of\_impact = []

                        total\_level\_of\_impact = []

                        total\_win\_contribution = []

                        total\_value = []

                        total\_MVP\_vote\_standing = []

                        for player in self.players:

                                candidate = self.players[player]

                                total\_player\_name.append(candidate.player\_name)

                                total\_season.append(candidate.season)

                                total\_fantasy\_basketball\_stats\_average.append(candidate.fantasy\_basketball\_stats\_average)

                                total\_game\_score.append(candidate.game\_score)

                                total\_player\_efficiency\_rating.append(candidate.player\_efficiency\_rating)

                                total\_total\_stats.append(candidate.total\_stats)

                                total\_VORP.append(candidate.VORP)

                                total\_quality\_of\_impact.append(candidate.quality\_of\_impact)

                                total\_level\_of\_impact.append(candidate.level\_of\_impact)

                                total\_win\_contribution.append(candidate.win\_contribution)

                                total\_value.append(candidate.value)

                                total\_MVP\_vote\_standing.append(candidate.MVP\_vote\_standing)

                        MVP\_candidates = {'Player': total\_player\_name, 'Season': total\_season,'Fantasy Basketball Stats Average': total\_fantasy\_basketball\_stats\_average, 'Game Score': total\_game\_score,'Player Efficiency Rating': total\_player\_efficiency\_rating, 'Total Stats': total\_total\_stats, 'VORP': total\_VORP, 'Quality of Impact': total\_quality\_of\_impact, 'Level of Impact': total\_level\_of\_impact, 'Win Contribution': total\_win\_contribution, 'Value': total\_value, 'MVP Voting Standing': total\_MVP\_vote\_standing}

                        table = pd.DataFrame(MVP\_candidates) #Creates a table using the Pandas library that contains columns that are the same as listed in line 316

                        graph = pd.DataFrame({'Player': total\_player\_name, 'Season': total\_season, 'Total Stats': total\_total\_stats, 'Win Contribution': total\_win\_contribution}) #Less data is needed when creating a scatter chart

                        with pd.ExcelWriter(str(season) + '\_Results.xlsx') as writer: #Writes the chart into a new excel file that is named for the season and \_Results

                                table.sort\_values('Value',ascending=False).to\_excel(writer, sheet\_name='Value Descending') #A table is created in the first sheet named 'Value Descending' that is organized by each players value in descending order

                                table.sort\_values('Predicted MVP Voting Standing',ascending=True).to\_excel(writer, sheet\_name='MVP Vote Standing') #A table is created in the second sheet named 'MVP Vote Standing' that organizes the table by each players MVP rank in ascending order

                                table.sort\_values('VORP',ascending=False).to\_excel(writer, sheet\_name='VORP') #A table is created in the third sheet named 'VORP' that is organized by VORP of the players in descending order

                                graph.to\_excel(writer, sheet\_name='Total Stats vs Win Contribution') #A blank graph is created in the fourth sheet named 'Total Stats vs Win Contribution'. The graph will be filled with series which correspond to players values

                                workbook = writer.book

                                worksheet=writer.sheets['Total Stats vs Win Contribution']

                                chart = workbook.add\_chart({'type': 'scatter'}) #Adds a scatter chart

                                max\_row = len(total\_total\_stats) #Figures out the length of the total\_total\_stats list to determine how many players, or series, are to be evaluated

                                for i in range(len(total\_player\_name)):

                                        col = i + 1

                                        chart.add\_series({'name': ['Total Stats vs Win Contribution', col, 1],

                                        'categories': ['Total Stats vs Win Contribution', col, 3, col, 3], #The x values in the scatter chart are the Total Stats

                                        'values': ['Total Stats vs Win Contribution', col, 4, col, 4], #The y values in the scatter chart are the Win Contribution

                                        'marker': {'type': 'circle', 'size': 7},})

                                chart.add\_series({'categories': ['Total Stats vs Win Contribution', 1, 3, max\_row, 3], #Adds a regression line or trendline to the scatter chart

                                        'values': ['Total Stats vs Win Contribution', 1, 4, max\_row, 4],

                                        'marker': {'type': 'none'},

                                        'trendline': {'type': 'linear'}})

                                chart.set\_title({'name': 'Total Stats vs Win Contribution'})

                                chart.set\_x\_axis({'name': 'Total Stats', 'min': 80})

                                chart.set\_y\_axis({'name': 'Win Contribution',

                                'major\_gridlines': {'visible': False}})

                                worksheet.insert\_chart('K2', chart, {'x\_offset': 25, 'y\_offset': 10})

                                writer.save()

        '''def decade\_table(self): #Similar funtion to the table() function except this is used for when more than one Excel file, or season, is to be evaluated to create a comprehensive table or scatter chart

                total\_player\_name = []

                total\_season = []

                total\_fantasy\_basketball\_stats\_average = []

                total\_game\_score = []

                total\_player\_efficiency\_rating = []

                total\_total\_stats = []

                total\_VORP = []

                total\_quality\_of\_impact = []

                total\_level\_of\_impact = []

                total\_win\_contribution = []

                total\_value = []

                total\_MVP\_vote\_standing = []

                for player in self.players:

                        candidate = self.players[player]

                        total\_player\_name.append(candidate.player\_name)

                        total\_season.append(candidate.season)

                        total\_fantasy\_basketball\_stats\_average.append(candidate.fantasy\_basketball\_stats\_average)

                        total\_game\_score.append(candidate.game\_score)

                        total\_player\_efficiency\_rating.append(candidate.player\_efficiency\_rating)

                        total\_total\_stats.append(candidate.total\_stats)

                        total\_VORP.append(candidate.VORP)

                        total\_quality\_of\_impact.append(candidate.quality\_of\_impact)

                        total\_level\_of\_impact.append(candidate.level\_of\_impact)

                        total\_win\_contribution.append(candidate.win\_contribution)

                        total\_value.append(candidate.value)

                        total\_MVP\_vote\_standing.append(candidate.MVP\_vote\_standing)

                MVP\_candidates = {'Player': total\_player\_name, 'Season': total\_season,'Fantasy Basketball Stats Average': total\_fantasy\_basketball\_stats\_average, 'Game Score': total\_game\_score,'Player Efficiency Rating': total\_player\_efficiency\_rating, 'Total Stats': total\_total\_stats, 'VORP': total\_VORP, 'Quality of Impact': total\_quality\_of\_impact, 'Level of Impact': total\_level\_of\_impact, 'Win Contribution': total\_win\_contribution, 'Value': total\_value, 'MVP Voting Standing': total\_MVP\_vote\_standing}

                table = pd.DataFrame(MVP\_candidates)

                graph = pd.DataFrame({'Player': total\_player\_name, 'Season': total\_season, 'Total Stats': total\_total\_stats, 'Win Contribution': total\_win\_contribution})

                with pd.ExcelWriter('Decade\_Results.xlsx') as writer:

                        table.sort\_values('Value',ascending=False).to\_excel(writer, sheet\_name='Value Descending')

                        table.sort\_values('MVP Voting Standing',ascending=True).to\_excel(writer, sheet\_name='MVP Vote Standing')

                        table.sort\_values('VORP',ascending=False).to\_excel(writer, sheet\_name='VORP')

                        graph.to\_excel(writer, sheet\_name='Total Stats vs Win Contribution')

                        workbook = writer.book

                        worksheet=writer.sheets['Total Stats vs Win Contribution']

                        chart = workbook.add\_chart({'type': 'scatter'})

                        max\_row = len(total\_total\_stats)

                        for i in range(len(total\_player\_name)):

                                col = i + 1

                                chart.add\_series({'name': ['Total Stats vs Win Contribution', col, 1],

                                'categories': ['Total Stats vs Win Contribution', col, 3, col, 3],

                                'values': ['Total Stats vs Win Contribution', col, 4, col, 4],

                                'marker': {'type': 'circle', 'size': 7},})

                        chart.add\_series({'categories': ['Total Stats vs Win Contribution', 1, 3, max\_row, 3],

                                'values': ['Total Stats vs Win Contribution', 1, 4, max\_row, 4],

                                'marker': {'type': 'none'},

                                'trendline': {'type': 'linear'}})

                        chart.set\_title({'name': 'Total Stats vs Win Contribution'})

                        chart.set\_x\_axis({'name': 'Total Stats', 'min': 80})

                        chart.set\_y\_axis({'name': 'Win Contribution',

                        'major\_gridlines': {'visible': False}})

                        worksheet.insert\_chart('K2', chart, {'x\_offset': 25, 'y\_offset': 10})

                        writer.save()'''

        '''def graph(self): #Attempted to create a scatter chart using seaborn library and MatPlotLib library but was unsuccessful. May look into it when adjusting model in future

                total\_total\_stats = []

                total\_win\_contribution = []

                for player in self.players:

                        candidate = self.players[player]

                        total\_total\_stats.append(candidate.total\_stats)

                        total\_win\_contribution.append(candidate.win\_contribution)

                MVP\_candidates = {'Total Stats': total\_total\_stats, 'Win Contribution': total\_win\_contribution}

                table = pd.DataFrame(MVP\_candidates)

                graph = sns.load\_dataset(table)

                ax = sns.regplot(x='Total Stats', y='Win Contribution', data=graph)'''

class Players:

        '''This class stores the values for each instance of player in the self.player dictionary. Each value corresponds to a statistic or result of a calculation'''

        def \_\_init\_\_(self, player\_index=None, player\_name=None, season=None, games\_played=None, minutes\_per\_game=None, player\_efficiency\_rating= None, true\_shooting\_percentage=None, usage= None, Win\_Share=None, VORP=None, offensive\_rating=None, defensive\_rating= None, FGA\_per\_game=None, FGM\_per\_game=None, FTM\_per\_game=None, FTA\_per\_game= None, ORB\_per\_game=None, DRB\_per\_game=None, total\_rebounds=None, rebounds\_per\_100\_poss=None, rebounds\_per\_36\_min=None,assists\_per\_game=None, total\_assists=None, assists\_per\_100\_poss=None, assists\_per\_36\_min=None, steals\_per\_game=None, total\_steals=None, steals\_per\_100\_poss=None, steals\_per\_36\_min=None, blocks\_per\_game=None, total\_blocks=None, blocks\_per\_100\_poss=None, blocks\_per\_36\_min=None, turnovers\_per\_game=None, total\_turnovers=None, turnovers\_per\_100\_poss=None, turnovers\_per\_36\_min=None, fouls\_per\_game=None, total\_fouls=None, fouls\_per\_100\_poss=None, fouls\_per\_36\_min=None, points\_per\_game=None, total\_points=None, points\_per\_100\_poss=None, points\_per\_36\_min=None, wins=None, fantasy\_basketball\_stats\_totals=None, fantasy\_basketball\_stats\_per\_100\_poss=None, fantasy\_basketball\_stats\_per\_36\_min=None, fantasy\_basketball\_stats\_average=None, game\_score=None, total\_stats=None, net\_rating=None, quality\_of\_impact=None, level\_of\_impact=None, win\_contribution=None, value=None, MVP\_vote\_standing=None):

                self.player\_name = player\_name

                self.player\_index = player\_index

                self.season = season

                self.games\_played= games\_played

                self.minutes\_per\_game= minutes\_per\_game

                self.player\_efficiency\_rating= player\_efficiency\_rating

                self.true\_shooting\_percentage= true\_shooting\_percentage

                self.usage= usage

                self.Win\_Share= Win\_Share

                self.VORP= VORP

                self.offensive\_rating= offensive\_rating

                self.defensive\_rating= defensive\_rating

                self.FGM\_per\_game= FGM\_per\_game

                self.FGA\_per\_game= FGA\_per\_game

                self.FTM\_per\_game= FTM\_per\_game

                self.FTA\_per\_game= FTA\_per\_game

                self.ORB\_per\_game= ORB\_per\_game

                self.DRB\_per\_game= DRB\_per\_game

                self.total\_rebounds= total\_rebounds

                self.rebounds\_per\_100\_poss= rebounds\_per\_100\_poss

                self.rebounds\_per\_36\_min= rebounds\_per\_36\_min

                self.assists\_per\_game= assists\_per\_game

                self.total\_assists = total\_assists

                self.assists\_per\_100\_poss = assists\_per\_100\_poss

                self.assists\_per\_36\_min = assists\_per\_36\_min

                self.steals\_per\_game= steals\_per\_game

                self.total\_steals= total\_steals

                self.steals\_per\_100\_poss= steals\_per\_100\_poss

                self.steals\_per\_36\_min= steals\_per\_36\_min

                self.blocks\_per\_game= blocks\_per\_game

                self.total\_blocks=total\_blocks

                self.blocks\_per\_100\_poss= blocks\_per\_100\_poss

                self.blocks\_per\_36\_min= blocks\_per\_36\_min

                self.turnovers\_per\_game= turnovers\_per\_game

                self.total\_turnovers = total\_turnovers

                self.turnovers\_per\_100\_poss=turnovers\_per\_100\_poss

                self.turnovers\_per\_36\_min=turnovers\_per\_36\_min

                self.fouls\_per\_game= fouls\_per\_game

                self.total\_fouls=total\_fouls

                self.fouls\_per\_100\_poss=fouls\_per\_100\_poss

                self.fouls\_per\_36\_min=fouls\_per\_36\_min

                self.points\_per\_game = points\_per\_game

                self.total\_points= total\_points

                self.points\_per\_100\_poss= points\_per\_100\_poss

                self.points\_per\_36\_min= points\_per\_36\_min

                self.wins= wins

                self.fantasy\_basketball\_stats\_totals = fantasy\_basketball\_stats\_totals

                self.fantasy\_basketball\_stats\_per\_100\_poss = fantasy\_basketball\_stats\_per\_100\_poss

                self.fantasy\_basketball\_stats\_per\_36\_min = fantasy\_basketball\_stats\_per\_36\_min

                self.fantasy\_basketball\_stats\_average = fantasy\_basketball\_stats\_average

                self.game\_score = game\_score

                self.total\_stats = total\_stats

                self.net\_rating = net\_rating

                self.quality\_of\_impact = quality\_of\_impact

                self.level\_of\_impact = level\_of\_impact

                self.win\_contribution = win\_contribution

                self.value = value

                self.MVP\_vote\_standing = MVP\_vote\_standing

def main():

    '''This runs the program.'''

    path = ["2019-2020.xlsx"] #The list of files to be evaluated. Some iterations may evaluate a single file or more than one file

    sheet\_name = ['Per Game', 'Totals', 'Advanced', 'Per 100 Poss', 'Per 36 Min', 'MVP Tracker', 'ATL', 'BKN', 'BOS', 'CHA', 'CHI', 'CLE', 'DAL', 'DEN', 'DET','GS', 'HOU', 'IND', 'LAC', 'LAL', 'MEM', 'MIA', 'MIL', 'MIN', 'NO', 'NY', 'OKC', 'ORL', 'PHI', 'PHX', 'POR', 'SA', 'SAC', 'TOR', 'UTAH', 'WSH']

    Excel\_Data\_Reader(path, sheet\_name)

if \_\_name\_\_ == '\_\_main\_\_':

    main()