NBA MVP Predictions for the 2021 Season

Anuj Amin
aamin6@wisc.edu

Sam Murray
spmurray3@wisc.edu

Brendan Zimmer
btzimmer@wisc.edu

Abstract

Building accurate prediction models using NBA analytics is becoming standard in the modern NBA. Every team in the NBA uses analytics to make better informed decisions. With a wider talent pool and increased demands on players, analytics becomes crucial in not only selecting players, but making sure that they perform their best. Reduced injury rates and better team plays are some of the benefits of analytics. In this project, we exploit machine learning techniques to predict NBA MVPs. By using statistics, we eliminate potential voter biases. We use 40 NBA seasons to predict the latest NBA season. Using linear regression as a baseline model, we compare results between regression trees, random forests, bagging, AdaBoost and support vector machines. As for metric evaluations, mean squared error (MSE), R^2 and mean absolute error (MAE) are used. The 2020-21 NBA season was exluded from the training dataset as it is used to evaluate our model. Feature selection was used to reduce the complexity of the model and to eliminate collinearity. All models utilized 14 predictors. Amongst all our models, based off of MSE, Adaboost performed the worst as it had an MSE of 0.087. In contrast, Regression tree had the best performance with an MSE of 0.056. While the results were initially surprising, we need to keep in mind that the actual MVP voting results have human biases, which can change the true outcome.

1. Introduction

NBA analysts conduct extensive data collecting and analysis in order to make better decisions for NBA teams. However, predicting the NBA MVP is a bit more challenging, since everyone has a different opinion on who is the MVP. Some voters pick the best player on the best team, others pick a player with a feel-good story behind them. In this project, we are motivated to eliminate the voters' biases and strictly use analytics to predict the most valuable player. We use a data set of roughly 400 players that utilizes basketball references' MVP voting results to uncover a relationship between the proportion of votes to a players statistics. From this analysis. we've trained and adapted many machine learning models to predict which player will win the MVP award for the 2020-2021 NBA season.

Additionally, as a side-effect of this study, we can learn what makes a good star player for a team. While there can be one MVP in a season, that doesn't guarantee a player will lead a team to a championship. This model can help front offices make better decisions on who to trade for or pursue in free agency. More practically, this model can help keep fans more in the loop about MVP voting and with fantasy sports. This is due to fact that our data-set is using available data and statistics that are commonly referenced when talking about NBA players.

If our model is not overly complex and interpretable, we could tell which statistics are important for choosing an MVP candidate or superstar. This would greatly help in forming some 'rule of thumb' in creating a competitive and winning team.

Using a variety of machine learning techniques including Linear Regression, Regression Trees, Regression Forests, Bagging and Boosting, we can now address the question of which player truly is the MVP of the league.

Our analysis will examine the features related to a players offensive and defensive skills along with winning. Features such as games won, points per game, assists per game etc. are some of the parameters of interest; as well as the connection they have to the proportion of votes received for the MVP reward. If there is a connection between an explanatory variable and the target variable, it will be included in the final model in order to predict which player is an MVP.

2. Related Work

Because our dataset was initially from Kaggle, plenty of people have explored the question of predicting NBA MVPs. The first piece of related work comes from Duarte Freire[1]. Their study focused on a similar dataset as ours, however, they used different machine learning algorithms and different feature selection methods. That being said, they also predicted MVP voting based off of proportion of votes received instead of making it a binary classification task. For feature selection and importance they used random forest feature scoring, while we used more common

Model	MSE	R-squared
DNN	0.027	0.681
KNN	0.023	0.726
RF	0.034	0.605

Figure 1. Results from Duarte Freire [1]

sense and trial and error for feature selection. They kept the algorithm choices simple by using neural networks, random forests and k-nearest neighbors, while our group decided to use more regression algorithms ranging from simple linear regression to ensemble methods. Our group also tried using methods of hyper-parameter tuning such as grid search to improve performance of a single regression tree. Our groups final model was also less accurate because we used boosting, while Duarte's most accurate model was the kNN. It's not uncertain if we are able to get a better result than that, but other methods and tuning should be tried.

Another study by Peter Li used a similar dataset and predicted MVPs based on a weighted score of team and individual success. This is because MVP voting is more subjective and based off of either team success or posting historical numbers [3]. The algorithm is not a statistically tested model, but the author of the analysis took into account each players value and win contribution. This leads to more subjectivity and one can argue that this model creates biases towards certain players. The author even mentions that their model overvalues big men who had great advanced statistics, but not so exceptional winning statistics. Another cause for concern raised by the author is the lack of defensive statistics because he simply did not include them. An additional concern with this data analysis is the size of the dataset is roughly 25% of our current dataset size.

3. Proposed Method

3.1. Linear Regression

Linear Regression is one of the most basic machine learning models. The goal is to fit the best linear line between the independent variable(s) and the dependent variable, assuming there is a linear relationship between the predictor variables and the target variable. Because we have multiple predictor variables we are using multiple linear regression. The formula for multiple linear regression is

$$y = b_0 + b_1 x_1 + ... + b_n x_n$$

where b_0 is the intercept, b_1,...,b_n are the coefficients of the independent variables x_1,...,x_n and y is the target variable.

The goal of linear regression is to find a linear equation such that our error is minimized, where the error is the difference in the actual value and the predicted value.

3.2. Regression Tree

A regression tree is a type of decision tree that at a high level uses a bunch cases to predict the outcome for a continuous variable. A regression tree predicts the label of a test sample by recursively searching for a leaf node where the conditions match. In order to get optimal results we used grid search as a method for hyper-parameter tuning. In our project we used a binary, squared error decision tree where via grid-search we experimented with varying combinations of max depth and samples to split. We have these parameters in order to control overfitting. Regression trees use recursive binary splitting [2] via sum squared error

$$SSE = \sum_{i=1}^{n} (Y_i - \hat{Y})^2$$

as a cost function in order to split points to create decision boundaries. The algorithm stops splitting once we have a certain number of instances assigned to a leaf node.

3.3. Random Forest

Random Forest is an ensemble method created from decision trees based on bagging and using random feature subsets. For each iteration, we train a regression tree based off a bootstrap sample and at each node of the tree, a random set of features is chosen for the purpose of splitting. In order to save on computation costs, we do not prune or do any sort of hyper-parameter tuning. In our project we have a maximum depth of 2 to reduce runtime, our criterion is squared error and we used 100 trees.

3.4. Adaptive Boosting

Boosting is another ensemble method that is similar to gradient boosting. The main differences are how weights are updated and how each classifier/regressor is combined. The basic idea is to combine multiple weak classifiers into a strong classifier where each weak learner is a decision tree with a single split. Adaboost puts more weight on difficult to classify instances and less on the easy to classify instances. The algorithm will do this for a k number of rounds, then it will be used to predict our continuous label. In our project we used 500 regression trees with a learning rate of 1 with a linear loss function.

3.5. Bagging

Bagging, also called bootstrap aggregation, is an ensemble method used to reduce variation in the dataset by taking bootstrap samples. For each classifer used in bagging, a random sample from the training set is selected with replacement (bootstrapping) and a classifier is trained on this bootstrap sample. Then the average of each classifier is taken and that becomes the final prediction. It is essentially

a simpler version of the Random Forest that was described previously.

3.6. Support Vector Machine

Support Vector Machines are a type of linear model used for both classification and regression. In our case we are using it for regression. The basic idea is to create a hyperplane that separates the data into classes. Assuming two points are separable by a line, if we have an n-dimensional feature space, we want a (n-1) dimensional feature space that separates the data points of two different classes. For example, assuming that the equation for the hyperplane is

$$w^T x + b = 0$$

and that distance 'c' is the decision boundary of two classes, the equation for the decision boundary is

$$w^T x + b = \pm c$$

Therefore, we must decide a decision boundary 'c' from the hyperplane such that data points closest to the hyperplane are within that boundary line. Our goal then becomes to finding points within the decision boundary with the smallest error rate. While the example above is a simple linear kernel, for this dataset, a polynomial kernel was used to create separating decision boundaries

$$\kappa(x, y) = (x^T y + c)^d$$

4. Experiments

Linear Regression, Regression Tree, Regression Forest, Bagging, Support Vector Machines and Adaptive Boosting models were all tested in a similar manner in order to evaluate their abilities to predicted proportion of MVP votes received per player.

4.1. Dataset

The dataset, originally obtained from Basketball Reference, was published on Kaggle after minimal preprocessing. However, because the dataset was relatively old, slight changes had to be made. Basketball Reference now only includes the top 10 players in the MVP race, while originally the website showed all the votes. Additionally, there were many more advanced statistics available to use, however the number of advanced statistics have drastically been reduced. Therefore, we had to omit and add some predictor variables in the dataset to account for new NBA seasons and the new Basketball Reference format. The main change was reducing the number of MVP votes to the top 10 per each NBA season instead of every player. Overall, a small amount of data engineering had to be done in order to make analysis much easier to do. The current dataset provides over 400 NBA players spanning over 40 NBA seasons. The dataset has 19 variables.

4.2. Feature Selection

Advanced Statistic Descriptions from basketball-reference.com.

Statistic	Description	Measures
Win Shares (WS)	An estimate of the number of wins contributed by a player	estimates how much a player is contributing to winning
Box Plus-Minus (BPM)	A box score estimate of the points per 100 possessions that a player contributed above a league-average player, translated to an average team	estimates a player's value to the team when that player is on the court
Value Over Replacement Player (VORP)	A box score estimate of the points per 100 TEAM possessions that a player contributed above a replacement-level (-2.0) player, translated to an average team and prorated to an 82-game season	estimates a players value to the team in contrast to teammates, considering how much they are or are not on the court

Initially, no feature selection was done for each model. However, this led to our results fitting the data a little too well to the validation set. For example, our linear regression model had a R^2 value of over 98% and virtually zero for the MSE. Clearly something was wrong. Upon manual inspection of the dataset using a correlation matrix (refer to figure 2), we saw collinearity was the issue. Specifically the variables 'Pts Won' and 'Pts Max' ans they are used in the calculations for MVP voting shares. Additionally, we removed 'Team Wins' as it had collinearity to the 'Overall Seed' variable. Finally we removed 'WS/48' because it is a similar stat to 'WS'. Therefore, our final model only included 14 predictors, reducing the initial 18 predictors.

4.3. Train, Validation and Test Splits

In order to accurately assess our model. Split up our "test" dataset into two parts. We split it up using sklearn's train_test_split using a validation size of .25 and train size of .75. Our test dataset was the 2020-2021 NBA season's top 10 MVP candidates. We also attempted to do leave one out cross validation, but we decided to to modify it slightly by attempting to do a "leave one NBA season out cross validation" i.e. a special type of group k-fold cross validation. This is because over the span of 40 years the NBA has changed so much, so we argued doing LOOCV would throw off our results because different players in different eras would mess up MVP voting. However, we were unable to implement this due to some of the MVP voting results

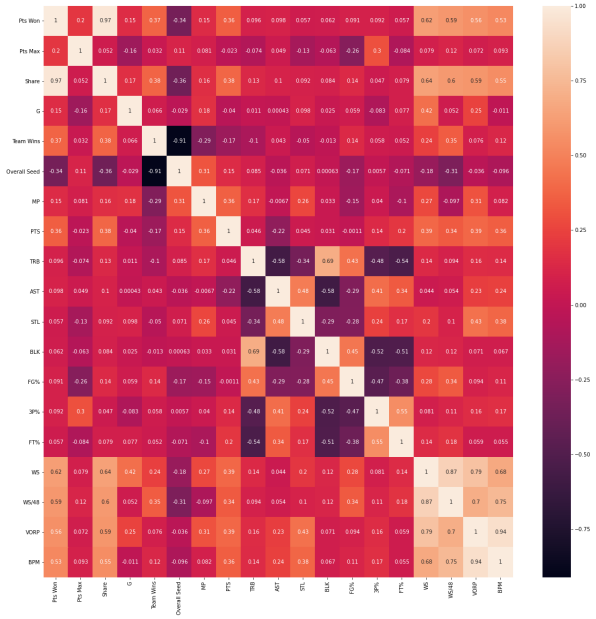

Figure 2. Correlation Matrix of Data

not having 10 players exactly due to ties in voting. Unfortunately, we had to scrap this idea due to time constraints.

4.4. Software

All data analysis was conducted in JupyterLab using Python 3.9. Packages used to explore data were pandas and operator. Matplotlib and seaborn were used for data visualizations. Finally, scikit-learn was used for a wide variety of machine learning models. While additional packages used were lightgm, xgboost, catboost, these were left out of the final report and analysis due to redundancies and time restraints.

4.5. Hardware

All data analysis was conducted on Windows 10 machines.

5. Results and Discussion

5.1. Results

As a baseline we fitted a basic linear regression with no penalties what so ever and this model achieved a MSE of 0.066, a MAE of 0.226 and a R^2 score of 0.297 on the test test. This is due to Linear Regression being such a simple model, which meant it should, in theory, be the least

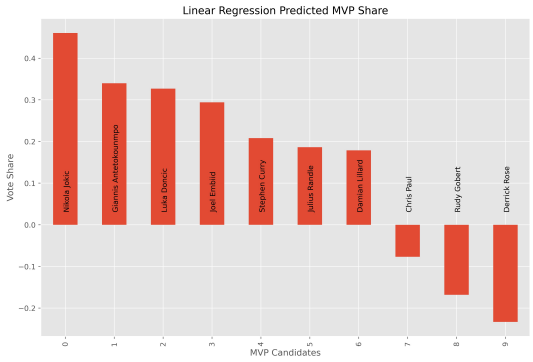

Figure 3. Projected Vote Shares via Linear Regression

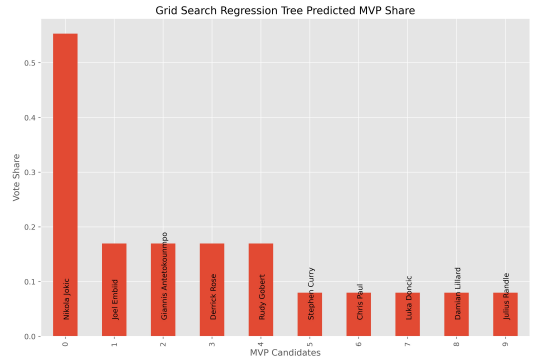

Figure 4. Projected Vote Shares via Regression Tree

accurate. While the linear regression model predicted the correct MVP (Nikola Jokic), it actually predicted the bottom 3 candidates to have a negative proportion of votes (see figure 3). This may be due to the fact that they didn't have "eye-popping" statistics like the other players.

Second, we fitted a Regression Tree and used grid search in order to find the optimal tree with varying max depths and minimum sample split values. Our final model for the regression tree had a max depth of 3 and used a minimum of 2 samples to split. This model achieved a MSE of 0.056, a MAE of 0.188 and a R^2 score of 0.396 on the test test. Clearly, while this was a relatively short tree, we did see some improvement compared to linear regression along with interpretability. Another benefit was that we did predict that the MVP was Jokic (see figure 4),

Third, we fitted a Regression Forest using 500 trees. This model achieved a MSE of 0.064, a MAE of 0.193 and a R^2 score of 0.315 on the test test. Clearly we did not see further improvement as the model did worse than the single tree. This surprised us because generally speaking, using numer-

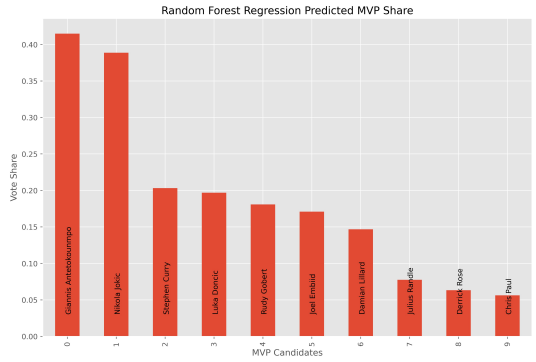

Figure 5. Projected Vote Shares via Regression Forest

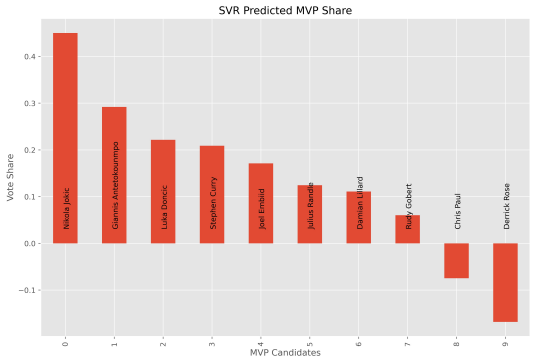

Figure 7. Projected Vote Shares via SVR

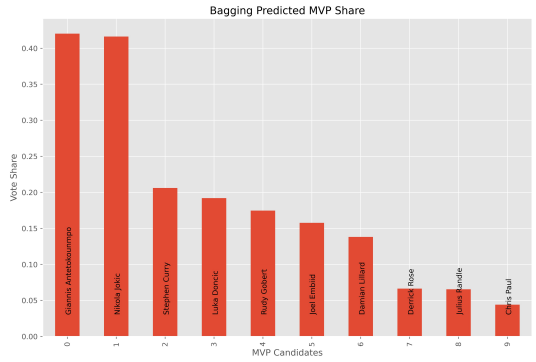

Figure 6. Projected Vote Shares via Bagging

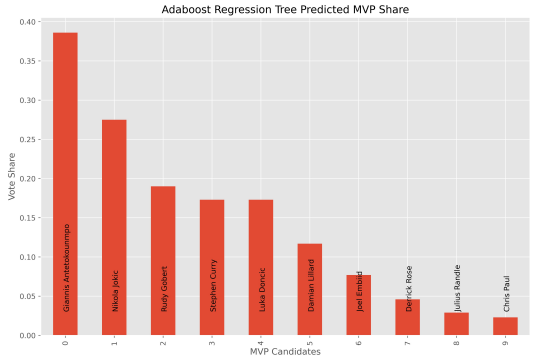

Figure 8. Projected Vote Shares via Adaboost

ous unpruned trees further improves predictability. Additionally, the predicted MVP was Antetokounmpo and not Jokic (see figure 5),

Fourth, we fitted a bagging regressor using 500 trees. This model achieved a MSE of 0.061, a MAE of 0.190 and a R^2 score of 0.341 on the test test. Clearly we did see some further improvement compared to the regression forest. This is likely due to using bootstrap samples in the model which can increase accuracy. However, we still predicted that Antetokounmpo and not Jokic was the MVP (see figure 6), even though the MVP race was much closer.

Fifth, we fitted a SVM regressor using a polynomial kernel. This model achieved a MSE of 0.062, a MAE of 0.201 and a R^2 score of 0.339 on the test test. We did not see further improvement compared to bagging since we only have one regressor compared to 500 regression trees. However, we were able to predict that the MVP was Jokic (see figure 7), That being said, similar to Linear Regression we got two MVP candidates who had negative vote shares. This may be due to SVM putting more priority on a candidate's points per game.

Finally, we fitted an Adaboost regressor. This model achieved the worst results with a MSE of 0.087, a MAE of 0.206 and a R^2 score of 0.064 on the test test. In addition to the abysmal results, the predicted MVP was Antetokounmpo and not Jokic (see figure 7),

5.2. Discussion

To assess the accuracy of the prediction methods used, we can compare the mean squared error, mean absolute error as well as the R^2 of the predictions. These values give us an estimate of how well each model makes MVP vote share predictions based on player stats compared to the actual vote share players received.

$$MAE = \sum_{i=1}^{n} |Y_i - \hat{Y}|$$

$$MSE = \sum_{i=1}^{n} (Y_i - \hat{Y})^2$$

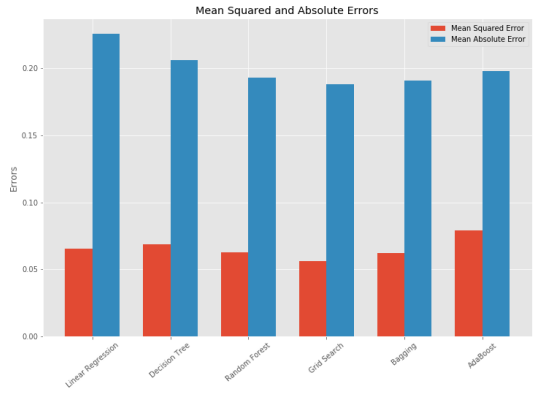

Figure 9. Evaluate Prediction Accuracy by Comparing Errors

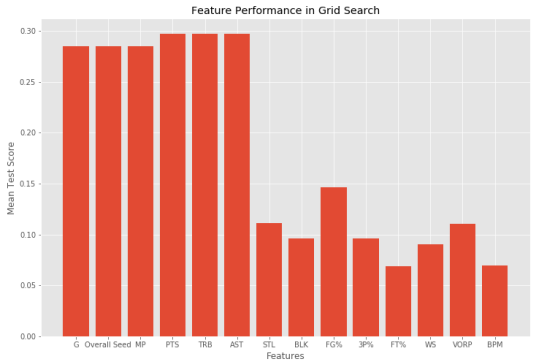

Figure 11. Mean Test Score for Features in Grid Search

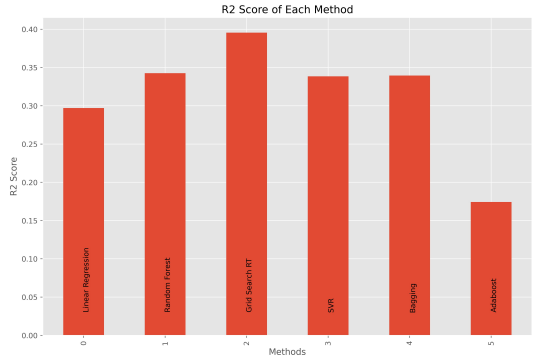

Figure 10. R2 Score of Each Method Used

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(Y_i - \hat{Y})^2}{\sum_{i=1}^{n}(Y_i - \bar{Y})^2}$$

Figure 9 shows the mean squared and absolute errors for all six prediction methods. Comparing mean squared error, AdaBoost resulted in the highest error while the regression tree was the most accurate. In contrast, mean absolute error is highest when linear regression and decision trees are used. However, the model with the least absolute error is also the grid search regression tree. Based on this we can conclude that predicting MVP vote shares based on NBA players' stats is most accurate when using a regression tree, while least accurate is a toss up between adaboost and linear regression.

Figure 10 shows the R^2 score of each method that was used for our predictions. Looking at the graph we have the same methods mentioned above. Grid Search through regression tree ended up having the best score. Usually you want the R^2 score to be higher which means the prediction will be better. But due to the varying data, as each NBA player has his respective position meaning different scoring, it causes lower R^2 scores. The lowest score was Adaboosting, which is most likely due to the large amount of variables that we used, being 27. Obviously the grid search with regression tree is the best prediction model and will give the best results based on the R^2 score. Including mean squared and absolute errors then linear regression would be optimal prediction model.

The results were surprising to us, because many of our models predicted a completely different MVP race than the actual outcome. The ensemble methods favored Giannis as the MVP while the other methods were in favor of Jokic as MVP. This begs the question of why our models performed so poorly. The first reason is there isn't much data to work with. While we have data over 40 years, 400 NBA players is a very small sample size. Therefore, our model will have higher error and variance as shown above. Additionally, basketball is a constantly evolving game. Just 10 years ago, the game relied heavily on big men and inside scoring. Now the game revolves more around shooting and quick point guards. Another reason for such an extreme difference is voter fatigue. Superstars like Lebron and Jordan were consistently at the top of the NBA for over a decade, yet they didn't win every year. Despite having amazing regular season and playoff statistics and records they often lost to other players. This is due to voters getting tired of voting for the same person and so they start looking for other players who have breakout seasons. This is potentially what happened to Giannis, as he was the back to back MVP coming into the 2020-21 season. He even won finals MVP for the 2020-21 season! Statistically, Jokic was great, but you could argue that Giannis more valuable based off of his success. This is why MVP voting is so difficult to do because many players make great cases.

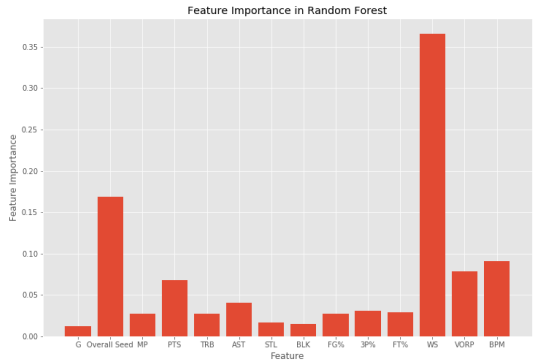

Figure 12. Feature Importance in Random Forest Predictions

5.3. Feature Importance

Figure 10 shows each feature's average test accuracy in the grid search prediction method. From this we can gather that points, assists, and rebounds per game are strong contributors to MVP voting, as well as the amount a player plays, both in number of games and time in games is very influential. Looking at the more advanced stats, value over replacement is one of the most influential in prediction MVP vote share. Least influential is box plus minus.

The difference between box plus minus and value over replacement is that while they are both statistics to measure a players value to a team while they are playing, box plus minus is only a rate statistic, so volume is not incorporated just a players value per 100 possessions. Value over replacement, on the other hand, shows a players value given the minutes and statistical involvement in games. Because grid search makes its predictions with more weight on individual stats, value over replacement has more value to the model than box plus minus because it considers volume. If a player has an efficient and effective 5 minutes in a game, they can have a high BPM, but would not have a VORP. When individual statistics are the valued the highest, knowing how a player performs over longer stretches of game time is more important than efficiency in MVP vote share predictions.

Alternatively, feature weight in Random Forests relies much more on team statistics to make its predictions. Figure 11 shows the importance of each feature in making predictions in the Random Forest model, with the two significantly most important input factors being win shares and overall seed. Conversely to grid search, this model first values that a player is on a successful team as opposed to valuing a players individual statistical impact. As a result, box plus minus has more importance than value over replacement because good teams often will have good bench players.

Value over replacement incorporates a player's value considering volume over the backup player at their position. Because the Random Forest model predicts strongly based on team success, candidates with higher predicted vote share will typically have less value over their replacements because their teams have better reserves. Box plus minus is valuable to this model because win shares is the feature with the highest importance. Players with high predicted MVP vote shares will typically be responsible for a high number of wins and therefore have high win shares. Because win shares is based on the volume of games where are you contribute to wins, pairing this with a rate statistic such as box plus minus to see how valuable a player is per 100 possessions given that they contribute to a high number of wins gives strong predictions for MVP vote shares.

6. Conclusions

The goal of this paper was to use a machine learning model to predict an NBA player's MVP vote shares based on their statistics. The prediction accuracy of several models was compared on a test data set after being fit on the top ten MVP vote share finishers over the previous several years. Despite the small dataset and relatively poor accuracy, we were able to draw insight on how potential MVP's are likely to be chosen.

Considering the mean squared errors, mean absolute errors, and R^2 scores of the different models, we can conclude that the regression tree model performed the best when making predictions on MVP vote shares. This model emphasized individual statistics such as points and rebounds, putting weight on individual success. Ensemble methods on the other hand value advanced statistics such as box plus minus which relate more to team success.

From this we can say that it is likely that the committee that votes on the NBA's MVP highly considers a mix of individual and team success along with something that makes a good media story.

A future direction this project could go in is comparing feature weight amongst different models and the likely prediction methods we can draw from them. For example, grid search was a well performing model and had very different emphasis on features than Random Forests, focusing on individual stats paired with advanced statistics like value over replacement. We suspect a pattern of confirming a players success either through winning or individual stats, then considering their specific value using an advanced statistic. These models use different methods for predictions but make very similar ones, so it would be interesting to investigate more to see what is similar in the models that makes both of them accurate.

7. Acknowledgements

Considering we knew very little about machine learning in Python, we'd like to thank and acknowledge Professor Raschka's teachings to explore this field. Additionally, we'd like to thank Freire Duarte for providing guidance and inspiration on how to update the original Kaggle dataset because it was fairly old. Finally, we'd like to thank Basketball Reference for being a source of clean data to work with.

8. Contributions

Each group member contributed to the modelling and writing process.

Anuj was responsible for fitting models, tuning models and updating the dataset. He also wrote the introduction, related work, proposed method and experiments.

Sam was responsible for visualizations and tables in the project along with writing experiments, results, discussion and conclusion.

Brendan was responsible for visualizations along with writing results, discussions, acknowledgements, and conclusion.

Every single one of us worked on each section of the write up as well.

References

[1] Duarte Freire. Predicting 2020–21 nba's most valuable player using machine learning, 2021. https://towardsdatascience.com/predicting-2020-21-nbas-most-valuable-player-using-machine-learning-24aaa869a740, Accessed: 2021-11-27.

[2] Jason Brownlee. Classification and regression trees for machine learning, 2020. https://machinelearningmastery.com/classification-and-regression-trees-for-machine-learning/, Accessed: 2021-11-28.

[3] Peter Li. Nba mvp prediction model, 2019. https://https://towardsdatascience.com/nba-mvp-predictor-c700e50e0917, Accessed: 2021-11-27.