Matthew Spitulnik

IST 707: Applied Machine Learning

3/30/23

Final Report

MLB MVP Predictor

Contents

[Introduction 2](#_Toc192535667)

[Data Gathering: 2](#_Toc192535668)

[Machine Learning Models and Methods 10](#_Toc192535669)

[Technical Results 10](#_Toc192535670)

## Introduction

For this project, statistics from Major League Baseball (MLB) were used to build a model that could predict which MLB players would receive votes for Most Valuable Player (MVP). . I started off with a list of every season for every MLB player to ever play the game, provided by the Lahman’s Baseball Database. I then built two different data mining scripts to collect data from Baseballreference.com; one script to compile a list of all players, each year, that received any votes for league MVP, and one script to collect the Wins Above Replacement (WAR) statistic for every single play to have ever played. Multiple cleaning and preparation steps needed to be performed to merge all of the required statistics together into one cohesive dataset. I then used the final dataset to try to build Decision Tree, Naïve Bayes, and Support Vector Machine models that could correctly predict whether or not a player would receive MVP votes in a given season. It became apparent that some bias existed in the MVP voting system, since certain statistics seemed to lead to more MVP votes, even if a player may have been thriving in numerous other statistics instead. I was surprised to see however, that there was not as much biased tied to name recognition—meaning players did not seem to get nods over other players just because they were a household name that had maybe received a lot of MVP votes in the past.

This project took place in the middle of an MLB season, so the player stats from that current year were not included in the dataset that was used to build the models. Instead, all of the players stats from that current season were collected separately and then used to test my final predictive models. Since MVP voting had not taken place yet, I judged the accuracy of my models by comparing the results to the players listed as having the highest odds to win MVP, according to various Las Vegas based sources. I determined my models were about 80% accurate during these tests.

## Data Gathering:

The baseball data came from Lahman’s Baseball Database and Baseball Reference. Lahman’s Baseball Database contains baseball statistics from 1871 to the present day. The data is available to download in spreadsheet format. Baseball Reference is one of the preeminent baseball statistics websites. Baseball Reference provides advanced statistics or sabermetrics such as OPS+ and wins above replacement (WAR) to further evaluate player performance.

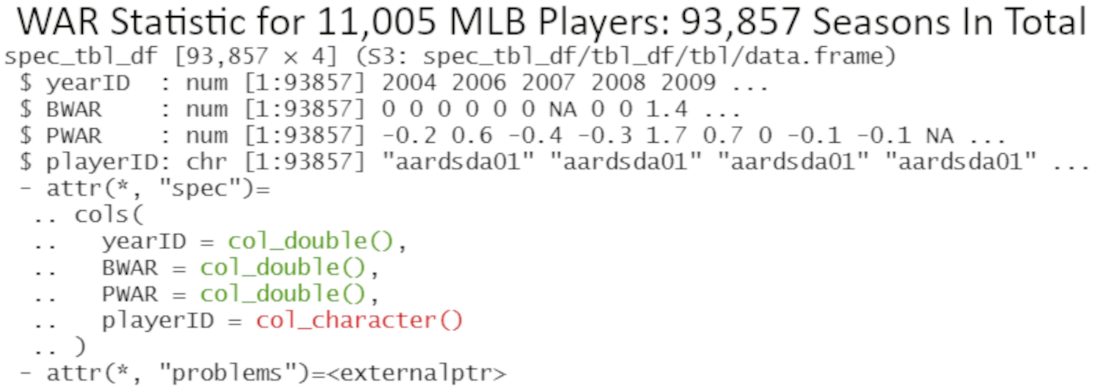
To perform the MLB MVP analysis, some additional files needed to be collected on top of the ones provided in the Lahman’s data base. If only the 204 players who had won the MVP award over the course of the MLB’s history were used, there was a good chance it would not be enough data to support an accurate model. Instead, the 40-50 players that all received votes each year could be used, which raised the collection of test subjects from 204 to 5,491. Lahman’s provided a list of all the players that had won the MVP awards each year, but not a list of all the players that had received MVP votes. Baseballreference.com had this information but it was divided into individual pages by year. To avoid having to manually go through all 100+ pages and download the information one page at a time, a “for loop” was created in R studio that was able to go through each page, download the required details, and then save it into a data frame. Going forward, any reference to an MVP player implies that they were MVP “vote-getters”, but they did not necessarily win the award.

The process to collect the MVP’s was also needed to collect the MLB player statistic known as Wins Above Replacement (WAR) for each player. WAR measures how many more wins a player is worth than replacement level players at the same position and is a very highly regarded statistic when considering MVP voting. The data set of MVPs already contained the WAR statistic, but the overall list of players that the MVPs were going to be compared to did not, and WAR is a complicated calculation that would not be easy to replicate. After cutting out player seasons that would not be used, 11,000 individual players still comprised the players list, so manually collecting the information was also not an option. Instead, another “for loop” was created that went out to a player’s page on Baseballreference.com, downloaded the information for their batting WAR, then the information for their pitching WAR, then put them into a data frame that slowly compiled the information for all the players. Once this was completed—with a run time of about 6 hours—all the information needed for the MVP analysis had been collected. Figures 1 and 2 below show the data structure of the MVP list and the collection of players WAR list:

Text

Description automatically generated with medium confidence

*Figure 1*



*Figure 2*

Although the required data had been collected, it was still located across various data frames, so sorting and combining was required. Figure 3 below can be referenced for a visual representation of how the data all tied together:

Diagram

Description automatically generated

*Figure 3*

 First, the “for loop” that collected all the MVPs produced two data frames, one for the American League (AL) players, and one for the National League (NL) players, so that was combined into one overall data set which subsequently was cleaned up to change character rows to numbers and replace NA values with zeros. Next, the batters and pitchers data sets were each aggregated to get an idea of how many unique batters and pitchers were being worked with (20,166 and 10,199), and to also account for players who moved teams mid-season so that they were not counted as multiple player and year entries. The batters and pitchers data sets were then merged together, resulting in 102,198 individual player’s seasons.

The MVP award only started being handed out consistently in 1931, but was intermittently awarded starting in 1911, so players seasons taking place during those non-MVP years were all removed. Additionally, only players that played for a certain amount of time should qualify for the award, so using the list of MVPs, a custom “MVPCutoOff” figure was calculated, which equaled the percent of innings pitched plus the percent of at-bats taken and then divided by two. The lowest MVPCutOff value amongst the 5,000+ MVPs, 0.04545401, was then applied as a benchmark to the Batters/Pitchers data set, with all seasons that had MVPCutOffs under .045 removed from the analysis data. This dropped the number of players seasons to analyze down to 58,308.

The “For Loop” that collected the WAR information produced players batting WAR (BWAR) and pitching WAR (PWAR). A players total WAR is equal to PWAR + BWAR, and this overall stat is what’s generally used during evaluations, so this was calculated and added as a variable. To collect the data, a baseball reference ID (bbrefID) was used to identify which pages needed to be visited. The batter/pitcher data set used something slightly different, a plane “playerID”, so using the overall Player data set (which contains both numbers), the bbrefID and playerID were matched up so that the WAR data could be merged with the batter/pitcher data.

After the batter and pitcher data sets were merged, columns with the same names, like the number of hits a batter had (H) and the number of hits a pitcher gave up (also H), had either “.x” or “.y” added to the end to differentiate them. To make further analysis of these variables easier, all of the “.x” were removed from the hitter stats, and the pitcher stats had an “\_A” added, to represent “against”, as in Hits Against, Runs Against, etc. The updated column names made it easier to now perform the calculations for the advance stats that needed to be added: Batting Average (BA), On Base Percentage (OBP), Singles (1XB, to use for other calculations), Slugging Percentage (SLG), On-Base Plus Slugging (OPS), and Walks And Hits Per Inning (WHIP). Figure 4 below shows the updated data frame:

A screenshot of a computer

AI-generated content may be incorrect.

*Figure 4*

In order to run the MVP analysis, the list of players in the MVP table needed to somehow be linked to the overall combined list of pitchers and batters. The MVP table had a player’s first and last name, but no playerID, while the batter/pitcher list had the playerID’s, but did not have the player’s name. The overall Players table had player’s first and last names, as well as the playerIDs, but the full names used did not always match up perfectly with the full name listed in the MVP table. This all meant multiple steps would be needed to obtain all the of the correct playerIDs for the associated names in the MVP table so that the MVP table could be properly merged with the batter/pitcher list. The steps were as follows:

1. Many of the playerID values appeared to be comprised of the first five letters of the players last name, plus the first two letters of their first name, plus “01”. The full name listed in the “Name” column of the MVP table was first broken apart into separate first and last name columns, then letters from each were subscripted out to match the pattern seen in the playerIDs, then they were combined with “01” to create an estimated playerID for each player in the MVP table. The process can be seen in figure 5 below. This technique successfully matched 4, 864 of the MVPs, but 627 remained.

Graphical user interface, application

Description automatically generated

*Figure 5*

1. Join commands were used to test the difference in rows between the MVP and Pitcher/Batter data sets. The difference was exported to a CSV file which was then manually updated, imported back in, and then used again to check how many MVPs still did not have the correct playerIDs, which was now down to down to 170.
2. For some players, the baseball reference ID listed in the Player data set was being used instead of the playerID, so a for loop was created to check all the remaining playerIDs, see if they were actually bbrefIDs, and if they were, change them to the playerID instead. This brought the difference down to 148 words.
3. If a player was not the first of their name to play in the MLB, instead of having an “01” at the end of their player ID, they had an “02”, or another number based on how many other people with the same name played before them. The “01”s in the playerIDs of the remaining players were changed to “02” and then checked again. Only 9 players remained, which meant a more manual process could now be used.
4. 6 of the remaining MVP seasons were for Jackie Robinson, whose playerID did not match the common template. His correct playerID was found and then manually updated, leaving only 3 players. 2 of the remaining 3 players also just had to have their names correctly updated, so once that was accomplished, only 1 player remained.
5. The last player remaining whose playerID could not be found was Joe Cronin. When the “MVPCutOff level was set, his number was the minimum amongst the MVPs, and thus used as the benchmark. Something changed between the value calculated in the MVP table and the value calculated in the Pitcher/Batter table because Joe Cronin had been filtered out of the data set being used for the analysis. This was corrected and Joe was returned to the analysis data set, eliminating any remaining differences, and enabling all players from the MVP table to be merged with the Player/Batter table.

From here, the testing and modeling process could now proceed. Figure 6 below shows a snapshot of the final data frame used for the analysis:

Table

Description automatically generated

*Figure 6*

## Machine Learning Models and Methods

A decision tree (DT) is a modelling technique that is especially useful for making decisions or classifying new data against already established categories in a sample population. In a decision tree, a root node is established which represents the primary question the modeling is being used to answer. The root node then splits into decisions nodes, representing variables in the data that have an effect on what the final decision outcome may end up being. The decision nodes continue splitting until a leaf node is reached, representing a final outcome or classification.

Naïve Bayes (NB) is a machine learning algorithm that uses the Bayes theorem to calculate the probably of one event occurring given a certain other piece of information. In the case of MVP voting, NB would find the probability that an MLB player would receive an MVP vote based on those players statistics in a season, compared to the statistics of all other players that ever received MVP votes in previous season. The “naïve” in NB is due to the formula’s assumption that all variables presented to it are completely independent and not correlated to each other.

A support vector machine (SVM) is an advanced machine learning algorithm used for the classification of data. It is a supervised learning technique, meaning the classifications it needs to predict are provided to it as part of the input process. SVMs distinguish between the different classes it is identifying using a hyperplane. The goal is to find the optimal hyperplane; the hyperplane with the maximum margin, or maximum distance between the points of the various classes. The hyperplanes are not limited to a 2-dimensional shape either. For each class that is added to the calculation, another dimension is added to the hyperplane, resulting in 3D or greater models (anything above 3D cannot be visually represented). This idea of multi-dimensional models is possible because of the use of *kernels*. Kernels are mathematical functions that take in data and converts it into the required form/dimension based on the type of kernel selected and type of data used. When the correct type of kernel is used with the correct type of data set, extremely accurate models can be calculated and built without having to use complex operations that take up time and resources.

## Technical Results

Three different models were used for this analysis in an attempt to get the most accurate results possible: Decision Tree (DT), Naïve Bayes(NB), and Support Vector Machines (SVM). The DT models worked right away without any major changes needing to be implemented. The NB models were failing at first, but once the “nonparametric” NB model was applied they began succeeding. No matter how many changes were made to the SVM’s, they unfortunately were not able to build a model, with failures every time. Luckily the other two techniques were able to produce good models with high accuracy.

Overall, changing the different parameter options on the DT and NB models (besides using nonparametric with the NB) did not have too much of an effect on the accuracies. Making changes to the data itself, however, did raise the accuracy a bit. On some models, certain attributes that were thought to be irrelevant were removed, some models had variables that were all normalized, and some models had a combination of both. Figures 7 through 10 below show some of the different variations of data that were used in the model building:

A diagram of a diagram

AI-generated content may be incorrect.

*Figure 7*

A screenshot of a computer screen

AI-generated content may be incorrect.

*Figure 8*

A diagram of a number of squares

AI-generated content may be incorrect.

*Figure 9*

A screenshot of a graph

AI-generated content may be incorrect.

*Figure 10*

To show the importance of having as many as labeled data points as possible, the first model demonstrated used only the actual 204 MVP award winners. It claimed to have produced a model with 99.61% accuracy, but it only accurately predicted 4 of the 16 MVPs used for the training, which fortified the fact that additional data was needed. Figure 11 below shows an output of this model:

A screenshot of a computer

AI-generated content may be incorrect.

*Figure 11*

When using the data set that included the 5491 MVPs, more balanced models were created. The best DT model produced had an accuracy of 93.51%, and it was built using a data set that had removed some variables that appeared, on the surface, to not be contributing anything to the predictions: Games, At-Bats, Games Started, and Innings Pitched. Figure 12 below shows the resulting DT. An important take away from this model was that the first node in the tree was WAR, which helps validate the need for including the WAR in the analysis.

A diagram of a tree model

AI-generated content may be incorrect.

*Figure 12*

 The best accuracy obtained from an NB model was 92.79%, using a data set that normalized all the standard, tallied variables, such as runs scored and HRs hit. An important take from this model is once again the role WAR plays in the predictions, which can be seen in figure 13 below:

A graph of a function

AI-generated content may be incorrect.

*Figure 13*

The majority of data points lie to the right of the zero, showing that a positive, strong WAR was an important factor in predicting who would be MVPs. Figure 14 below shows this model’s info:

A screenshot of a computer

AI-generated content may be incorrect.

*Figure 14*

 To further test these models, the stats for the current 2022 players were all collected, with the intention of running the data through the strongest models to see how their predictions compared to the odds that Las Vegas sites have predicted for who would win MVP. First, the data set with just the 204 actual MVP winners was used to further validate the necessity for including more rows in the analysis. The resulting output did not predict any of the analyzed players would be MVP, showing the weakness of this method. The outcome for this test can be seen in figure 15 below:

A screenshot of a computer

AI-generated content may be incorrect.

*Figure 15*

When the highest accuracy DT model was used to predict the data, it successfully classified 23 players that would be MVP candidates. Per Covers.com, 16 of the 23 players the model predicted were within the top 20 for highest odds of winning the award. These results can be seen in figure 16 below:

A screenshot of a sports game

AI-generated content may be incorrect.

*Figure 16*

The NB model was not as successful however, predicting 177 potential MVP candidates, but only 3 of them matching the names in the Covers.com list. The NB model with the second highest accuracy was also not very successful, predicting 60 MVPs but only one of them matching the Covers.com list. Figure 17 below compares the results of the different 2022 model predictions:

A graph of different colored squares

AI-generated content may be incorrect.

*Figure 17*