Using Machine Learning to predict MLB success Based on MILB performance

Alexander Gow

Programming Humanity

Prof Chun & Prof Elkins

Overview

In 2017 alone Major League Baseball organizations spent a combined 492 million dollars on acquiring new, young, talented players through the draft or signing international free agents. The effectiveness of an MLB organization's scouting department is pertinent to ensuring future success. Due to the highly volatile nature of professional sports, identifying a predictive and measurable statistic(s) associated with success among players would be valuable. Furthermore, a team's ability to quickly and correctly identify players on their Minor League affiliates that will have a positive impact at the Major League level can give them an advantage over other teams. For this project I wanted to build a Neural Network trained on Minor League batting statistics from the AA level that predicts success in the MLB.

Brief explanation of ML

In order to understand how I did my analysis, I must give a brief explanation of what a neural network is and how it works. The intention behind a neural network is to artificially recreate an environment that mimics interconnected brain cells. The structure can be reduced down to an input followed by a series of fully connected layers each containing varying numbers of artificial neurons each holding a different weight, leading to an output layer. As data is fed through the network, the neurons in each individual layer passes information to the next layer, eventually reaching the output layer where the network can evaluate how well the weights produced the desired outcome. Information is fed back through the network from output layer so the neurons can adjust their weights thus increasing the network's accuracy. The data is run through the network many times so the weights can be optimized.

Data Used

My dataset consisted of Minor League (AA) batting statistics by season collected from 2007 through 2016, resulting in a total of 1438 individual player statistics. I chose to only focus on the AA level because it is the most common place for MLB teams to place their future stars. I hand picked what statistics I would collect on each player by researching what stats are considered to be most representative of player performance. The seasonal stats used as the input included: games played, at bats, plate appearances, hits, home runs, runs batted in, batting average, BB/K (walks per strikeout), strikeout percentage, slugging percentage (total bases divided by at bats), OPS (slugging percentage plus on-base percentage), stolen bases, spd (evaluates the player's speed), wRC+ (quantifies and normalizes how well the player creates runs), BABIP (batting average on balls hit in play), line drive percentage, and ground-balls / fly balls. The statistic used as the output was a value from 0 - 4 placed on how successful a career the player had or is having. A 0would mean that the player either never made the MLB or had less than 400 MLB at bats. A 4 is assigned to players who have established themselves as one of the players in the MLB.

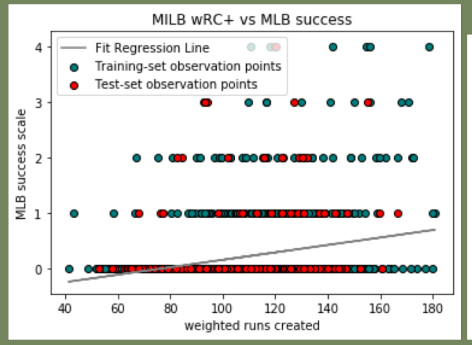

Figure 1

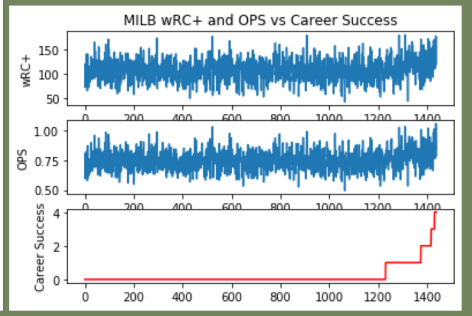

Figure 2

Methods

In order to properly understand if there was any predictive or significant correlation between MILB performance and MLB success I ran three different tests. The first test was a simple linear regression using the scikit learn package in python to examine the relationship between Minor League wRC+ and MLB success. I chose wRC+ as my variable because it is viewed as the most comprehensive batting statistic available for Minor League data. The second test was a multivariable linear regression using xgboost which employs a gradient boosted decision tree algorithm. In the process of using xgboost I found the direct correlation of each statistic with the level of success. I wanted to use xgboost because of how effectively it can run a multivariable regression, thus providing me with how and to what extent the MILB statistics can predict MLB success. The final test I ran was the creation of the neural network using keras, a tensorflow library that is used for building deep learning algorithms. The neural network was intended to show that MLB success can be predicted with a variety of Minor League statistics.

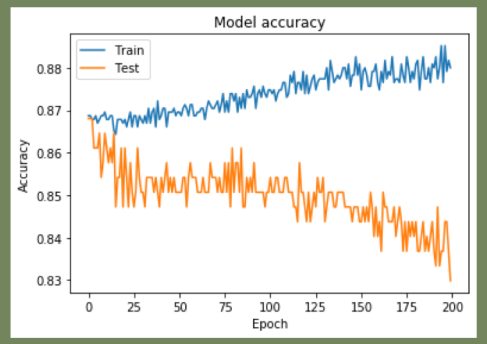

Figure 3

Figure 4

Results & Analysis

Using the Scikit learn package to run a linear regression on the relationship between wRC+ in the MILB and a successful MLB career (Figure 1). The result was a value of 0.05164, meaning that wRC+ only explains about 5 percent of the outcome. With the best possible value being 1, a simple linear regression based upon MILB wRC+ statistics is not a good predictor of future MLB success.

In the implementation of xgboost I wanted to find the correlations of each variable with the level of MLB success. OPS, and wRC+ were the most positively correlated values with success level(Figure 2). Without using xgboost's advantages I ran a regression of all the variables against the success level of the players and received 0.11752. When taking all variables into account the model is able to predict what players have been successful much better than only using one. However, this means that the model's prediction is right about 12% of the time. The RMSE(root means squared error) is the standard deviation of the prediction error made by the model. In this case, a value of 0.46 means that the model's predictions are not effective.

Using xgboost to create a system of decision trees in order to predict success resulted in an explained variance score of -1.656815232554322. Ideally this value would be close to one. However, a negative value implies that my decision tree has been overfit to my training data. Overfitting the model to your training data is one of the dangers when using xgboost and decision trees. As a result, using xgboost to build a model that predicts MLB success based on Minor League statistics is not an effective tool.

I did not have high expectations for the results from both the linear regression run using scikit learn and the predictor model using xgboost. However, I had hoped that building a neural network using the keras library in tensorflow would produce a more accurate model. After building and training the neural network, the training accuracy was 0.8800 and the training loss was 0.3040. Furthermore, my testing accuracy was 0.8299 and testing loss was 0.5138 (Figure 3, Figure 4). Clearly neural network predictions are far more accurate than those made by the scikit learn linear and xgboost models. Although, the loss values for the neural network are higher than what I would like. Figure 4 shows the model loss as the network trains, training loss decreases as it trains and testing loss increases which might imply that either the career outcome is too random for the input statistics or that the network has been overfit to the training data. The accuracy assessment speaks to how much better a deep neural network can wrangle and understand many variables that lead to an outcome.

Future work

Before running any of my tests, I had an expectation that the results from the models would not be very significant because I was trying to model a highly volatile career path simply using a selection of in game statistics. Trying to accurately predict and model a baseball players career without taking into account the countless and almost immeasurable external variables, such as the individual's work ethic or social tendencies (ie. partying, drug use) appears to be almost impossible. If I were to continue my research and try to assign a value to a player's character, I would use a twitter API to scrape old tweets of players and using an NLP algorithm try to correlate the types of social media activity associated with "good character" and "detrimental characters". Furthermore, one of the most predictive variables when trying to model a players career arc is their age. For example, a 22 year old with the same statistics as a 32 year old in AA will likely have a much more successful career. In addition, I would be able to build a far more accurate and interesting neural network if I were to be able to obtain the actual career earnings of each player in my dataset, the weights of each neuron would be trained on the monetary outcome of players career based upon a variety of in-game statistics and character evaluation.

My inspiration for this project stemmed from a San Francisco based company that assists semi-professional (Minor league and independent) baseball players in mitigating the risk associated with pursuing professional sports. They group players into salary pools in which a portion of a player's earnings is placed into the pool and divided amongst the others, allowing players to live on a salary greater than the poverty line. The company uses Machine Learning algorithms in order project careers arcs. Which, in turn, inspired my attempt at modeling career success based on MILB statistics.