Predicting Whether an NBA Player Will Have a Career Lasting at Least 5 Years Based on Their Rookie Year Statistics

Gavin Murdock
Department of Computer Science
Utah State University
Logan, UT, United States
a02277093@aggies.usu.edu

***Abstract*—This project was created with the goal of predicting whether a player in the National Basketball Association (NBA) will have a career lasting at least 5 years based solely on per game statistics from their rookie year. It will explain the datasets used and how they were processed in order to get the best performance out of the chosen classification models. This project utilized four popular machine learning classification models: logistic regression, decision tree, support vector machine (SVM), and random forest. This paper will detail each of these models, how they were optimized, and evaluate how good they were at predicting whether an NBA rookie will have a career lasting at least 5 years. It will also explain the features present in the data and evaluate which features were the most important in predicting career length. Possible applications of this work will also be discussed.**

***Index Terms*—classification, machine learning, logistic regression, decision tree, support vector machine, random forest**

I. Introduction

Basketball is the second most popular sport in America and the popularity of the National Basketball Association, or NBA, in the United States is growing an estimated 4% every year [1]. There are currently 30 NBA teams that are each allowed to have a maximum of 15 players on their roster. This means that just 450 players are allowed in the NBA at any one time. Because such a small number of players are allowed in the NBA, teams make frequent adjustments to their rosters with a substantial number of players joining and leaving the league each year. Players are also frequently traded from one team to another as each team tries their best to be competitive and win as many games as possible. This makes it an important goal of teams to predict which players will perform the best and for the longest.

There have been thousands of players over the NBA's lifetime with an average career length of 4.5 years [2]. This project aims to build a model that can effectively predict which players will have an above average career length of at least 5 years. Knowing which players are likely to have long careers can help teams try to acquire these players early. Acquiring players early, especially before their prime, can help teams by giving up less if they are trading for them and by giving them more time to build their players around them in hopes of making them a franchise player. Knowing which players are not likely to have long careers can also be beneficial for teams. This can let teams know which players probably aren't worth investing too much in and could be good candidates to trade away.

In order to build an effective model for this task, four different popular types of machine learning classification models were used. Logistic regression, decision tree, support vector machine (SVM), and random forest models were constructed, optimized, and tested for performance. These models were evaluated on five core metrics: accuracy, precision, recall, F1-score, and area under the receiver operating characteristic curve, or ROC curve. Before these models were fitted and tested, relevant data was also gathered and processed to optimize model performance. Analysis of the data features was also performed to find which features hold the most weight or are the most important to predicting career length of NBA players.

II. Method

A. Data Collection

To collect data for this project, I utilized two main datasets that when put together would provide the information I needed. The first dataset I used was acquired through data.world where it was uploaded by the user Gabe Salzar after he collected the data from NBA.com [3]. It included information on every NBA rookie from the year 1980 up to 2016 for a total of 1,538 players. This dataset contained the following information for each rookie: player name, year drafted, games played, and per game averages for points, field goals made, field goals attempted, field goal percentage, 3 points field goals made, 3 point field goals attempted, 3 point field goal percentage, free throws made, free throws attempted, free throw percentage, offensive rebounds, defensive rebounds, rebounds, assists, steals, blocks, turnovers, and efficiency.

Now that I had data detailing rookie year statistics for a substantial number of players, I needed data that would provide the career lengths of those players. The second dataset I used was also retrieved from data.world and uploaded by

TABLE I
THREE SAMPLE DATA POINTS AFTER PRE-PROCESSING

Draft Year	**GP**	**MIN**	**PTS**	**FGM**	**FGA**	**FG%**	**3PM**	**3PA**	**3P%**	**FTM**
-0.849298	0.107273	-1.076585	-0.954533	-0.954100	-1.089941	1.211006	-0.664860	-0.758268	-1.232828	-0.702844
-0.183914	-0.646852	-1.183498	-1.138319	-1.131601	-1.117398	-1.389432	-0.664860	-0.569436	0.177821	-0.806789
-0.469079	0.977416	2.142689	1.986053	1.885925	1.847943	0.399404	1.667824	1.696551	1.111899	1.480006

FTA	**FT%**	**OREB**	**DREB**	**REB**	**AST**	**STL**	**BLK**	**TOV**	**EFF**	**Career $\geq$ 5yrs**
-0.611638	-0.511850	-0.116544	-0.373715	-0.289393	-0.998863	-0.781721	1.117691	-0.952998	-0.638194	1
-0.844801	0.429138	-0.500132	-0.744758	-0.679360	-0.860495	-1.028303	-0.394465	-1.230883	-1.077238	0
1.486832	0.391498	0.011319	0.368373	0.246812	1.560951	2.423839	0.865665	1.964790	1.622882	1

Gabe Salzar who initially retrieved the data from basketball-reference.com [4]. It contained a variety of information on 3,407 NBA players through the 2016 draft class. This second dataset detailed the following information: player name, career totals for a number of statistics, player birth day, player birth location, and, most importantly, career length in number of years.

B. Data Pre-processing

After collecting the two datasets and transforming them to pandas dataframes, I needed to extract the relevant information for my task. For my second dataset that included career totals for a number of categories, I removed all features except player name and career length in years. I removed the other features such as career total points because they included information beyond a player's rookie year and would've gone against my goal of basing predictions solely on rookie year statistics. This second dataset also had many extra rows where every feature was a null value, so I also removed these extra rows. I kept all the features from my first dataset and then merged the two dataframes based on player name, keeping only the players that appeared in both datasets. This left me with data for 1,285 rookies, a number I felt was substantial enough to work with but ideally would've been higher. After merging the dataframes, I also removed the player name column because it could cause certain models like a decision tree to classify players based on their names in order to achieve a high training accuracy and player names are irrelevant to my task.

After combining my datasets and removing the irrelevant features, I looked for missing values in the data. I found there were a significant number of players who had missing values for three point field goals made, three point field goals attempted, or three point field goal percentage. Almost all of the players who had missing three point statistics seemed to be from the 80s and early 90s. The three point field goal wasn't implemented in the NBA until the 1979-80 season and there wasn't a league average of more than ten attempted three point field goals per game until the 1994-95 season [5]. Because of this, I assumed these three point statistic values were missing for some players because they never attempted or made any three point field goals, so I replaced these missing values with zeros. I didn't find any other missing values in the data. The only other feature that needed adjusting was the target feature showing career length in years. I modified the values in this column to reflect my target classes of either having a career shorter than five years or having a career lasting at least five years. I set the value of career length to zero for all players who had a career shorter than five years and I set the value of career length to one for all players who had a career lasting at least five years.

Next, I separated the target column, career length, from the input features and created training and testing sets from the data. I used scikit-learn's train_test_split function to create a testing set containing 20% of the total data, or 257 data points, and a training set consisting of the other 80% of the data, or 1,028 data points. I then standardized the input data using the StandardScaler class, again from scikit-learn. I fit the StandardScaler to the input training data and then used it to standardize both the training input data and the testing input data. Table 1 shows a sample of three training data points after all of these pre-processing steps.

C. Model Construction and Parameter Tuning

I tested a number of different model types in order to determine which one would work best for my task of predicting whether an NBA player's career will last at least 5 years based solely on statistics from their rookie year. I tested the following four different model types with varying parameter values: logistic regression, decision tree, support vector machine, and random forest. This section will explain how I constructed these models, what parameters I tested for each model, and each model's performance in terms of mean F1-score after performing 10-fold cross validation on my training data. I kept the best performing model of each type to use for testing.

1) Logistic Regression: I first created a logistic regression model using a class from scikit-learn. Then, using scikit-learn's cross validation grid search function, I tested combinations of different values for the following parameters: C, class weight, solver, and max iterations. For the C parameter that specifies the strength of regularization I tested the following values: 1.0, 0.75, 0.5, 0.1, 0.01, and 0.001. For the class weight parameter that defines if and how the different target classes are weighted, I tested both none, meaning no class weights were applied, and balanced, which adjusts the weight of each class based on its frequency. For the solver parameter that specifies which algorithm is used in the optimization problem, I tested both lbfgs and liblinear. Lastly, I tested the following values for the number of max iterations taken for the solver to converge: 100, 150, 200, 250, 300, and 500. A 10-fold cross validation test was performed on only the training data

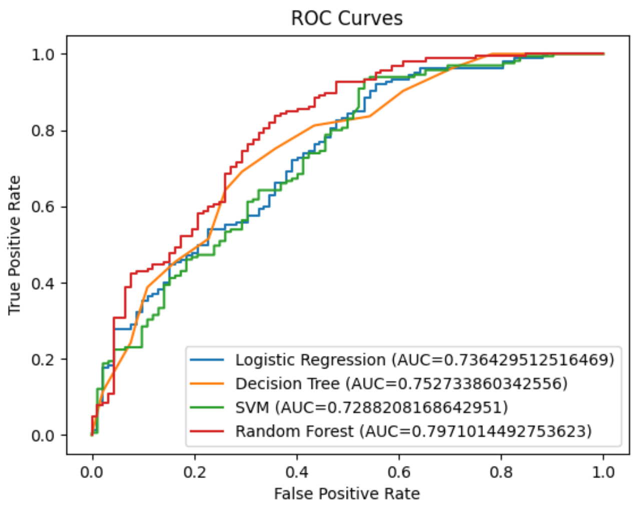

Fig. 1. ROC curves for best performing model of each type.

for every combination of the above parameter values and then scored by its mean F1-score. The mean F1-scores ranged from about 0.69 to 0.78. A C value of 0.1, no class weights, an lbfgs solver, and a max iterations value of 100 led to the highest mean F1-score of 0.7758.

2) Decision Tree: I then constructed a decision tree model using another class from scikit-learn. I then used the same cross validation grid search function as before to test different combinations of the following parameters: criterion, max depth, min samples split, and min samples leaf. For the criterion parameter that specifies what function is used to measure the quality of a tree split, I tested both gini and entropy. I tested the following values for the maximum depth of the decision tree: 10, 15, 20, 30, 40, and none, meaning the tree could be as deep as needed. For min samples split, which specifies the minimum number of samples needed to split a tree node, I tested the following values: 2, 10, 50, and 100. For min samples leaf, or the minimum number of samples required to be at a leaf node, I tested the following values: 1, 10, 20, 50, and 100. After testing every combination of the above parameters using 10-fold cross validation on only the training data, I got a range of mean F1-scores from around 0.7 to 0.78. Using entropy as the criterion, a max depth of 10, a min samples split value of 2, and a min samples leaf value of 50 led to the highest mean F1-score of 0.7759.

3) Support Vector Machine: Next, I created a support vector machine, or SVM, model using a class from scikit-learn. Using the same cross validation grid search function as before, I tested combinations of values for the C parameter and the kernel parameter. For the C parameter that specifies the strength of regularization, I tested the following values: 1.0, 0.75, 0.5, 0.1, 0.01, and 0.001. I also tested the following four kernel types to be used in the SVM algorithm: linear, poly, rbf, and sigmoid. After again performing 10-fold cross validation tests on only the training data for each combination

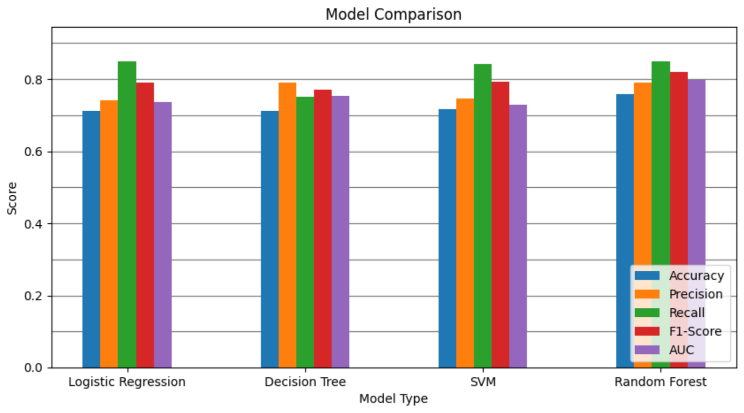

Fig. 2. Evaluation metrics for best performing model of each type.

of the above parameters, I got mean F1-scores from around 0.69 to 0.77. A C value of 0.01 and a linear kernel led to the highest mean F1-score of 0.772.

4) Random Forest: The last model I created was a random forest model that was constructed from another scikit-learn class. I again used scikit-learn's cross validation grid search function to test different values of the following parameters: n estimators, criterion, max depth, min samples split, and min samples leaf. For n estimators, which specifies the number of decision trees in the random forest, I tested the following values: 75, 100, 135, 165, and 200. For criterion, or the function used to measure how good a split is, I tested both gini and entropy. For max depth, which specifies the maximum depth of decision trees in the random forest, I tested the following values: 10, 15, 20, 30, 40, and none. For min samples split, or the minimum number of samples needed in a tree node to split, I tested the following values: 2, 10, 50, and 100. For the last parameter, min samples leaf, that determines the minimum number of samples required to be at a leaf node, I tested the following values: 1, 10, 20, 50, and 100. A 10-fold cross validation test was performed on only the training data for every combination of the above parameter values, which led to mean F1-scores from around 0.75 to 0.80. Using entropy as the criterion, a max depth of 30, a min samples split value of 10, a min samples leaf value of 1, and having 135 estimators led to the highest mean F1-score of 0.7983.

III. Experimental Results

After testing several combinations of different parameters for logistic regression, decision tree, support vector machine, and random forest models, I kept the best performing one of each type in terms of mean F1-score after performing a 10-fold cross validation test on only the training data. I then used these four models to predict the target class of my testing data before evaluating them based on the following five metrics: accuracy, precision, recall, F1-score, and area under ROC curve or AUC.

The logistic regression model had a testing accuracy of 0.712, a precision of 0.7407, a recall of 0.8485, an AUC of 0.7364, and an F1-score of 0.791. The decision tree model had a testing accuracy of 0.7121, a precision of 0.7898, a recall of 0.7515, an AUC of 0.7527, and an F1-score of 0.7702. The support vector machine model had a testing accuracy of 0.7160, a precision of 0.7473, a recall of 0.8424, an AUC of 0.7288, and an F1-score of 0.792. Lastly, the random forest model had a testing accuracy of 0.7588, a precision of 0.7910, a recall of 0.8485, an AUC of 0.7971, and an F1-score of 0.8187. All four model's ROC curves can be seen in Fig. 1 and a bar chart visualizing the four model's tested metrics can be seen in Fig. 2.

Unsurprisingly, the random forest model performed the best. I thought this would be likely because it was the only ensemble classifier I used and random forests are known for being very robust. The random forest model performed better than all the other models in every tested metric except recall, where it tied with logistic regression for the best recall. What I found a little more surprising was how similarly the logistic regression, decision tree, and support vector machine models performed. They were all just a slight step down from the random forest model.

In addition to building and testing the four models above, I did some feature analysis to understand the data a bit better. I first looked at my best performing model, the random forest model. Using some features of the scikit-learn class I used to build the model, I was able to look at how important each feature of the input data was in being able to classify a player as having a career of at least five years or less than 5 years. These feature importances are calculated as the mean and standard deviation of accumulation of the entropy decrease within each tree in the random forest. This is also known as feature importance based on the mean decrease in impurity, and impurity in this case is entropy. Using this data, I constructed a feature importance plot that can be seen in Fig. 3. I found draft year, games played, efficiency, and field goal percentage to be the four most important features (in that order) for my random forest model when predicting the career length of an NBA rookie player. I also performed Pearson correlation tests between each input data feature and the target class. I found the strongest correlation to be games played with a correlation coefficient of 0.391. The correlation with games played also had a p value of 3.018e-48, making the correlation very statistically significant. Efficiency, a basketball statistic that takes points, rebounds, assists, steals, blocks, missed shots, and games played into account, had the second highest correlation with a coefficient of 0.347 and a p value of 1.390e-37. These two correlations suggest these features would be important predictors of career length, and that is shown in the feature importance plot for the random forest model as well. It makes sense that a rookie who plays more often and is efficient on the court would have a long career.

When performing Pearson correlation tests between the input data features and the target variable, I also found some features that didn't have much correlation. Three point field goals made, three point field goals attempted, and three point field goal percentage all had low correlation values of 0.042, 0.024, and 0.004 respectively. On top of that, these were the only three features that had p values above 0.05, the threshold for statistical significance. These three features also had pretty low feature importance values for the random forest model. Because of this, I decided to remove these features from the data and build and test another random forest model. After removing these three features from the data, I built another random forest model in the same manner as before. This new model had an F1-score of 0.7949, which is slightly lower than my other random forest model that operated on the entire dataset. These low correlation values and slightly lower F1-score might be because, as I stated before, there were a significant amount of players in this dataset from the 1980s and 90s without three point field goal data because shooting three point field goals wasn't as popular back then. In the 1980-81 NBA season there was an average of 2.8 three point field goals attempted per game, and in the 2022-23 NBA season there was an average of 34.2 three point field goals attempted per game [5]. I would speculate that three point field goal statistics would be a lot more important for predicting the career length of newer rookies than those of the past.

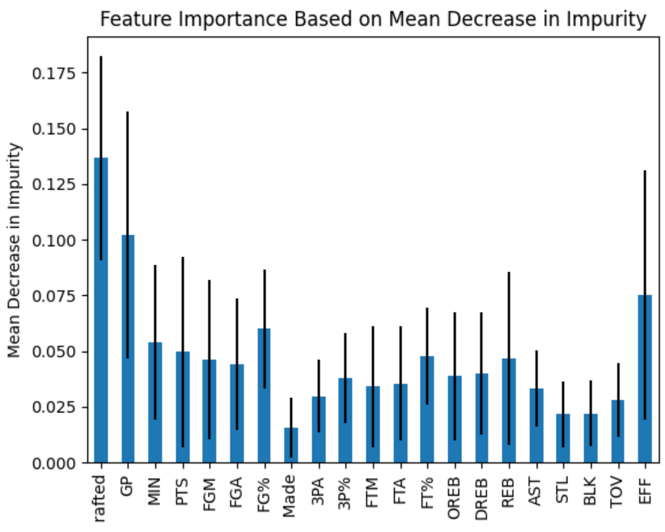

Fig. 3. Feature importance plot for random forest model.

IV. Conclusion

The goal of this project was to build a machine learning model that could effectively predict whether an NBA player's career would last at least five years based on statistics solely from their rookie year. The closest I came to doing this was with a random forest model using 165 decision trees. This model had an F1-score of 0.8187 and an accuracy just above 75%, meaning it could correctly predict whether a rookie player would have a career lasting at least 5 years about three out of four times. I obviously would've wanted these numbers to be a little higher, but predicting the career length of an NBA player based on only their first year in the league isn't an easy task. There are many factors that go into a player's career beyond their first year like injuries and team chemistry which are hard to predict.

In this project, I also looked at which features were the most important in predicting whether a rookie's career would last at least five years. I found year drafted, games played, efficiency, and field goal percentage to be the most important factors. For NBA teams and managers, this information could also be beneficial by providing them a set of statistics they should be most concerned with when deciding which players are worth investing in. I also found three point field goal statistics to be the least important, but I am also using older data and I believe these statistics could be important for newer players as three point field goals are becoming a bigger part of the NBA.

In the future, I would like to see what effect changing the input data would have on my model's effectiveness. I think using newer data would be a lot more beneficial. The NBA has changed in many ways since the 80s and 90s and I think using newer input data would be more relevant to today's players and lead to better predictions for new players. It would also be interesting to look at player's college basketball statistics as well as their NBA rookie year which I think could improve the effectiveness of my models. Lastly, I would like to see the effect of adding player's physical features like height and weight to the input data on my models in hopes of making better predictions. Better predictions would in turn make these models more useful for teams deciding which players to go after or trade away.

References

[1] R. Gurung, "Top 12 most popular sports in America," Players Bio, 27-Apr-2023. [Online]. Available: https://playersbio.com/most-popular-sports-in-america/. [Accessed: 01-May-2023].

[2] "Life after NBA comes sooner than many players think," NBA.com, 10-Jun-2010. [Online]. Available: https://www.nba.com/nuggets/features/junior_bridgeman_20100610.html. [Accessed: 01-May-2023].

[3] G. Salzer, "NBA rookies by min (1980-2016)," data.world, 05-Jan-2017. [Online]. Available: https://data.world/gmoney/nba-rookies-by-min-1980-2016. [Accessed: 02-May-2023].

[4] G. Salzer, "NBA players' birthplaces," data.world, 07-Nov-2017. [Online]. Available: https://data.world/gmoney/nba-players-birthplaces. [Accessed: 02-May-2023].

[5] "NBA league averages - per game," Basketball reference, 2023. [Online]. Available: https://www.basketball-reference.com/leagues/NBA_stats_per_game.html. [Accessed: 02-May-2023].