Brian McCabe
Data Science 4449 – Capstone
Denver University
Final
11/15/2022
Professor: Sada Narayanappa

Baseball Hall of Fame: Classification

Baseball is Americas past time, while it is mainly played in the United States baseball has many things to offer as far as statistics go. Sports analyses are constantly creating new formulas to measure players' performance. The goal of this project is to use the more well-known statistics of baseball to see if we can train an algorithm to classify a player's Hall of Fame status with respect to the player's basic batting, fielding, pitching, and accolade stats. Of the tens of thousands of baseball players that have been throughout its history, 340 of them are in the Baseball Hall of Fame, and of that only 268 were players.

Coaches, scouts, and even players are always tracking their stats and seeing where they can improve both on and off the field. This project can be used by baseball players to see how their stats compare to the players in the Baseball Hall of Fame. As well as by coaches and scouts to find the next talented player. The big questions I look to answer with this project are what statistics make a Hall of Fame player and how this information can be useful.

My data set used in this analysis is the Lahman Baseball Database, which was officially released on January 24, 2015, and is updated at the end of every baseball season. The data includes multiple files for many distinct aspects of baseball. The data ranges from 1871 through 2021. My analysis will be done on a combined dataset from 7 distinct data sets. Including batting, fielding, pitching, all-star stats, awards, basic information, and Hall of Fame status. With the goal of determining key aspects that make a Hall of Fame player. All the combined statistics make up a data set with the possible key features for this analysis.

The batting data set is a CSV file with over 110k records. Players had records for multiple years, so I condensed the data set into unique players by combining records on playerID. Features included games played, at-bats, runs, hits, walks, steals, strikeouts, and so on. All the batting stats are numerical integers or floating-point numbers.

The fielding data set is a CSV file with over 147k records. Same as with the batting data set, I condensed the data on playerID and removed some fielding statistics that I had deemed unnecessary to my analysis. Features included, games started, Inn Outs, Put Outs, Assists, Errors, and so on. All the fielding stats are numerical integers or floating-point numbers.

The pitching data set is a CSV file with over 49k records. Did the same as with the above data set. Features included, games played, Strike Outs, Walks, Earned Runs, Inning Pitched, and so on. All the pitching states are numerical integers or floating-point numbers.

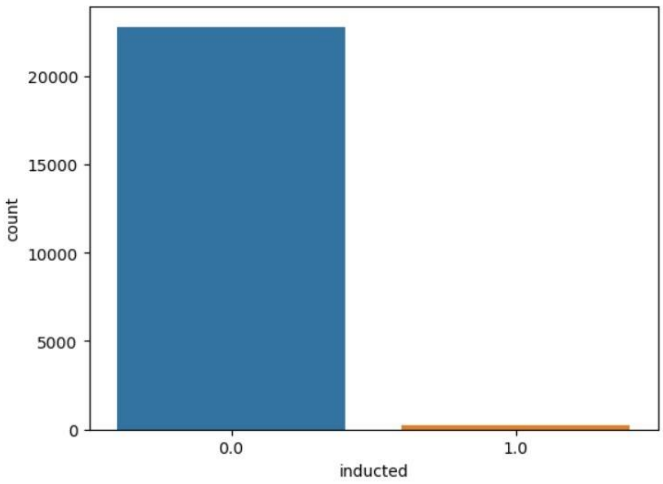
The last 4 data sets were a bit easier. For the awards, I condensed the records and added a count for the number of awards won and merged that into the main data set. Similarly for the all-star data set. I counted the number of all-star appearances and merged that into the main data set. Lastly was the basic info set, so I could easily search for players by their first and last name, and the Hall of Fame data set, the target variable. Both of these were easily merged on the player id after I took out anyone that was not a player. Some Awards and Hall of Fame statues are given to umpires and coaches. The Hall of Fame status was imputed to be binary 1 and 0 instead of Y and N.

To create one data frame with all the information that I needed I combined the 7 data sets that I had cleaned on playerID and added 0 for all the NAN values as a value of zero represents the data correctly. With all the data merged and cleaning done I was left with 23005 records and 50 features including my target feature, player position, and player first and last name.

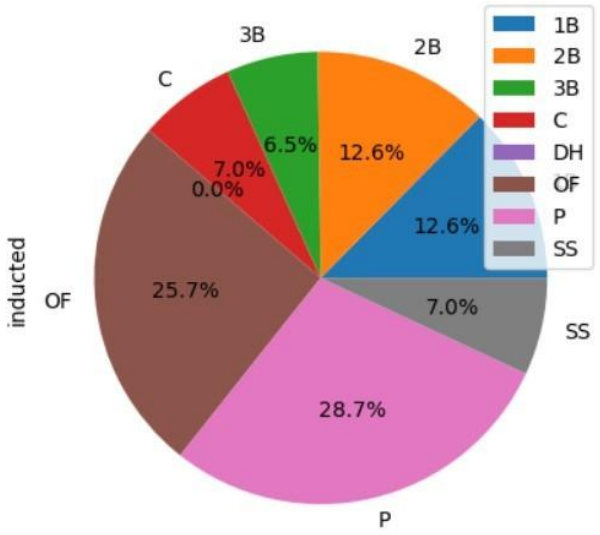
Exploratory data analysis included looking at the distributions and outliers for each of my features. I used a library called Dtale to help me visualize all my data. Dtale is an interactive data frame that can produce histograms, count plots, and summary statistics for each variable. After looking through all the features in Dtale it gives me a better idea of what I want to be looking at more specifically when doing my exploratory analysis.

	Awards	inducted	AS_ap
count	23005.000000	23005.000000	23005.000000
mean	1.083112	0.009998	0.709194
std	3.453841	0.099490	2.132709
min	0.000000	0.000000	0.000000
25%	0.000000	0.000000	0.000000
50%	0.000000	0.000000	0.000000
75%	0.000000	0.000000	0.000000
max	47.000000	1.000000	26.000000

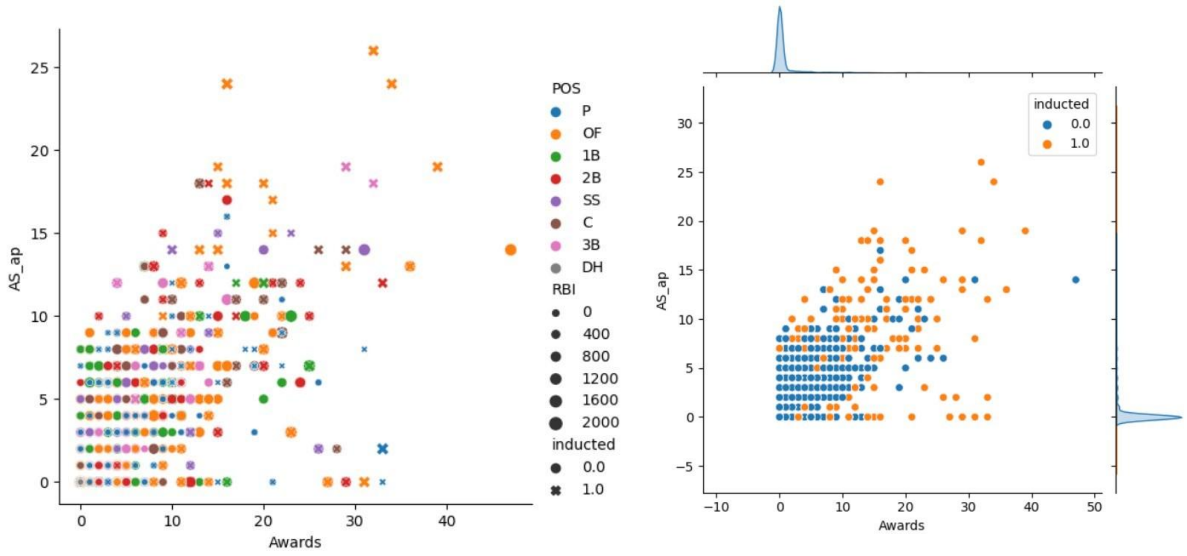
The figure above shows the description of the awards, Hall of Fame, and all-star appearances stats. As you can see it is exceedingly rare to be in the Baseball Hall of Fame. It is far less rare to have at least one all-star appearance, but it is still only for the best players. As for the Awards, the reason the mean is high is because of the number of awards that are given out and the number of players that have won a considerable number of awards. The main takeaway is that the distribution of inducted must be looked at.



Based on the chart above we can see that our target variable is imbalanced. I wanted to see what will happen before and after we address the imbalance of the data. So, I would have more proof of the importance of the implementation of imbalance techniques. Therefore, I did the analysis both before and after using an up-sampling technique.

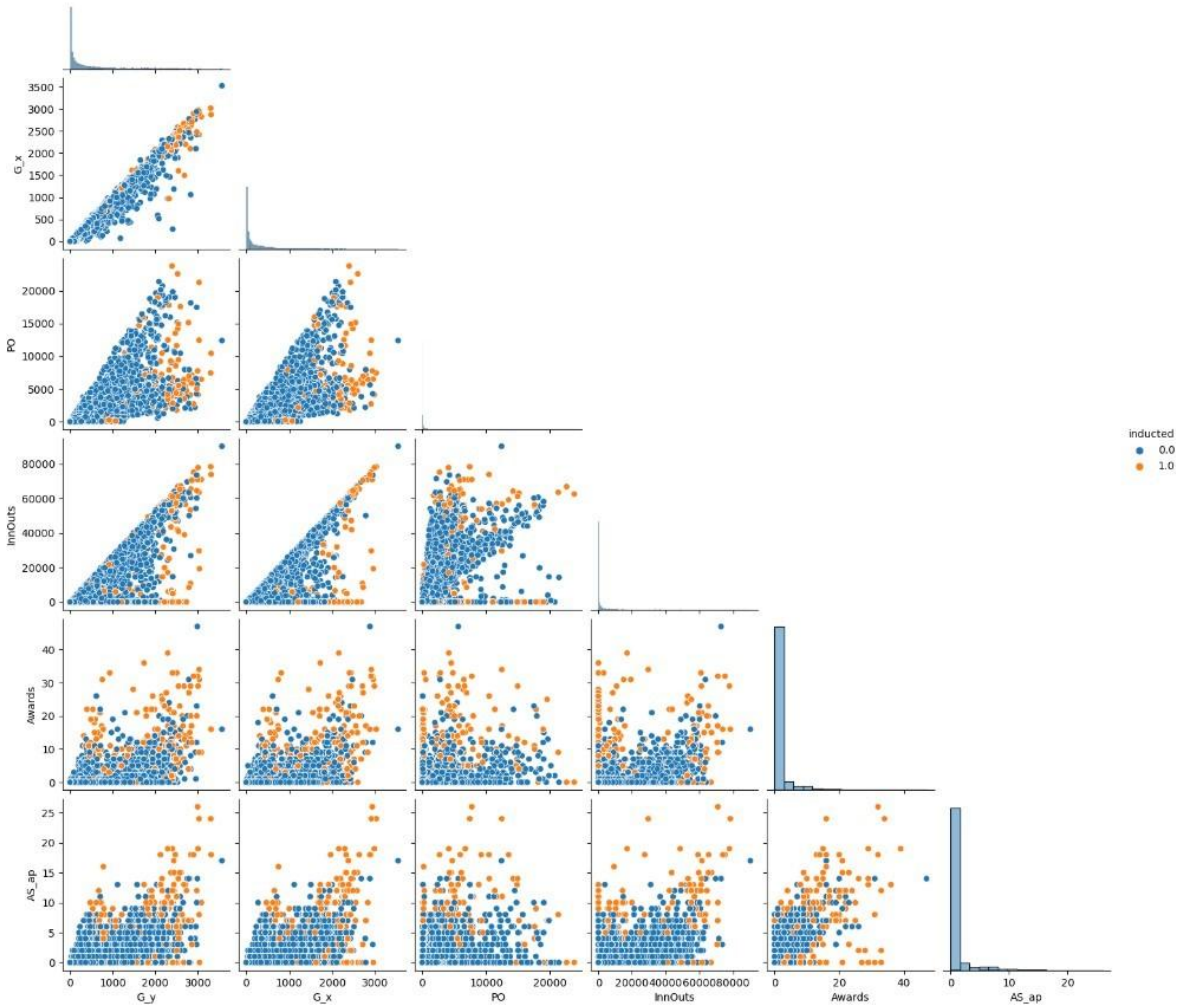


This pie chart displays the distribution of players in the Hall of Fame based on the position they played. I wanted to see if there would be an imbalance like there was during my midterm project. This was not the case, while most players are pitchers, pitchers are not unproportionally overrepresented in the Hall of Fame.



The chart on the left displays awards won vs all-star appearances, the color hue represents the player's position, the shape indicates if they are in the hall of fame or not and the size represents how many runs batted in the player has. The chart on the right similarly displays awards won vs all-star appearances with a hue where orange indicates Hall of Fame players and distribution on the top and right. The distributions show that the means are close to zero. These two plots help to see how the players in the Hall of Fame often have a high number of all-star appearances and awards. But there is the line of orange at the bottom that shows there are many players that are in the Hall of Fame and have not played in an all-star game. This can be explained by the fact that professional baseball started in 1871 and the first all-star game did not occur until 1933. We can also easily identify an outlier that has about 45+ awards, 15 all-star appearances, and is not in the Hall of Fame.

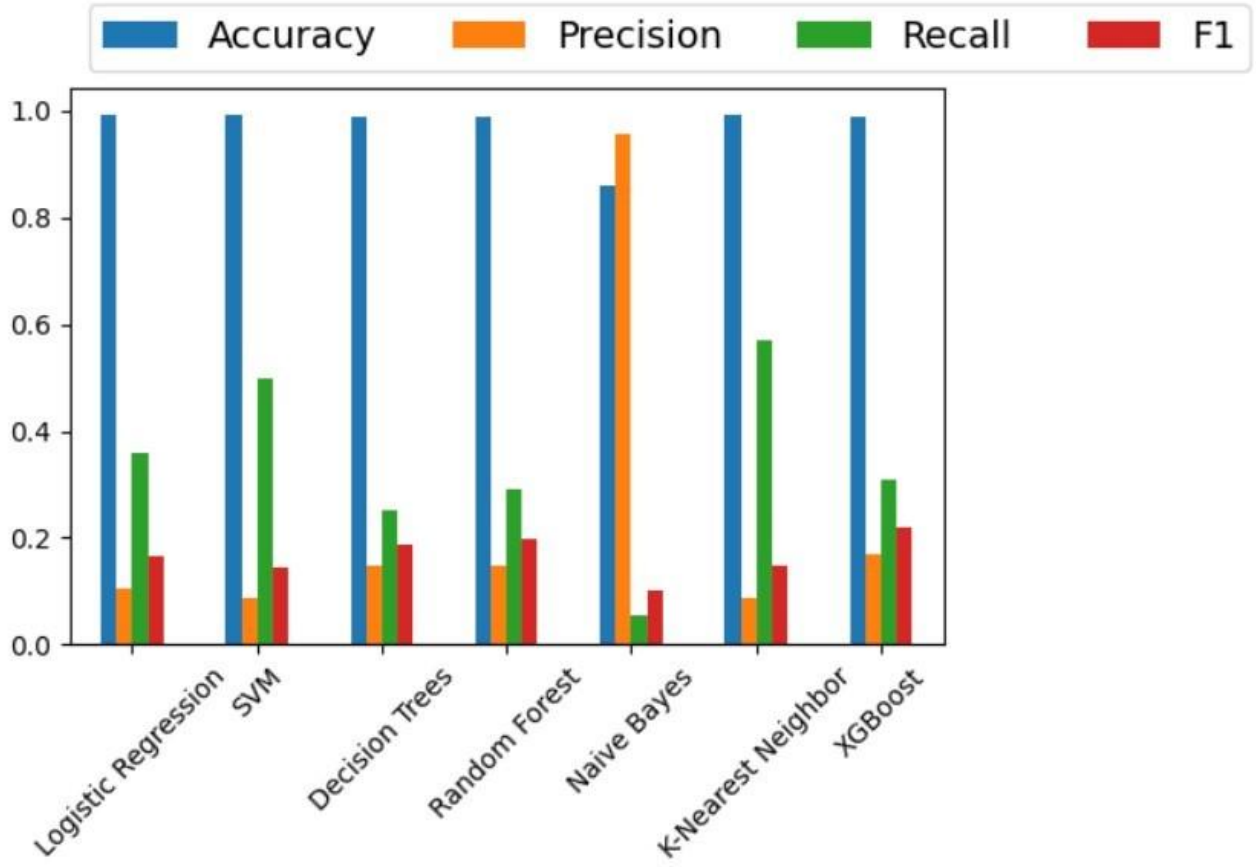
Other exploratory analyses included looking at feature selection with chi^2 and mutual info classification techniques. This helped me to identify which features would be most important in my analysis. These features included InnOuts, Games played, and one that I expected, awards. Based on the importance of awards and how players are voted on to win awards like how they are voted into the Hall of Fame made me think there would be a link between them.



So, I took the top features from my feature selection and looked at them in a pair plot with a hue overlay defining the Hall of Fame Status. This resulted in a better view of how some of the statistics were dispersed between the Hall of Fame and not Hall of Fame players. When I saw this, I had a good feeling that my classification would be accurate.

Before starting my initial analysis, the merged data frame was split into a 75% training set and a 25% test set. I chose this split as I had a large dataset and wanted more data to train on. The data was also scaled using a standard scaler transformation. This scales the numbers down making it more efficient for the algorithm to train on. I ran these training and test sets through seven different algorithms. I used Gaussian Naïve Bayes, Support Vector Classifier, K-Nearest Neighbors, Logistic Regression, XGBoost, Decision Trees, and Random Forest Classifiers to see which would give me the best results. I wanted to test multiple algorithms because it is simple to

create a loop to test each algorithm. You never know which algorithm will perform the best. All the accuracies were high on my initial analysis; however, upon closer inspection, the precision, recall, and F1 scores were poor on my initial analysis as seen below.



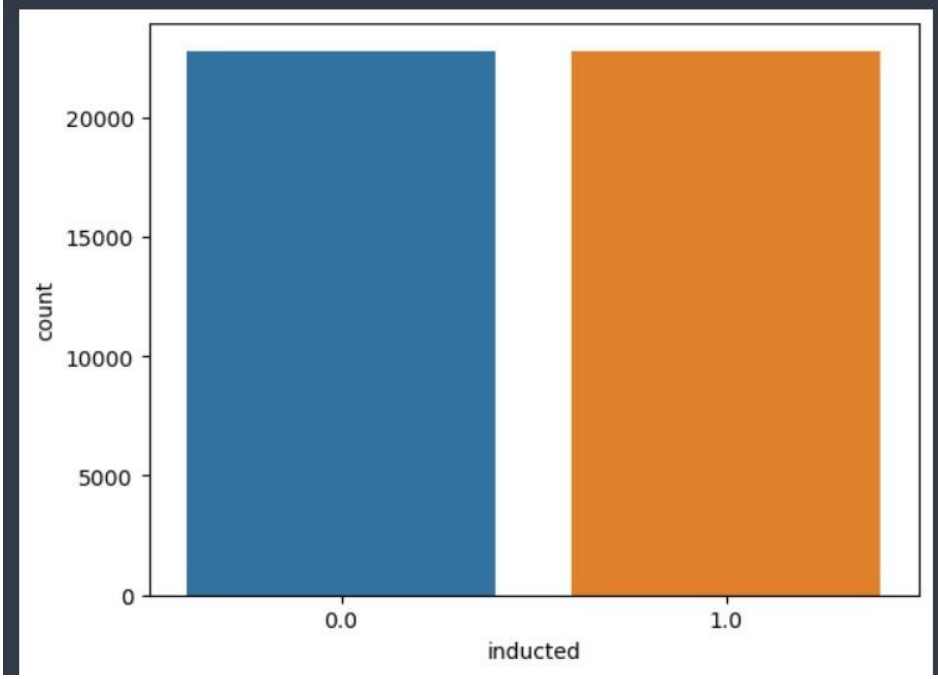
Thus, I knew that something had to be done to yield better results. There were two techniques I tried to address the data set imbalance. There were 22,775 players not in the Hall of Fame and 230 players in the Hall of Fame. The data set was vastly imbalanced, my first technique was upscaling the target variable. So, using the resample method from the sklearn.utils package. I was able to balance out the distribution of the target variable. As seen in the image below, now my data set was balanced using replicated rows from the minority result.

```
df_majority = merged_df[(merged_df['inducted']==0)]
df_minority = merged_df[(merged_df['inducted']==1)]
df_minority_upsampled = resample(df_minority, replace=True,n_samples= 22775,random_state=42)
df_upsampled = pd.concat([df_minority_upsampled, df_majority])
```

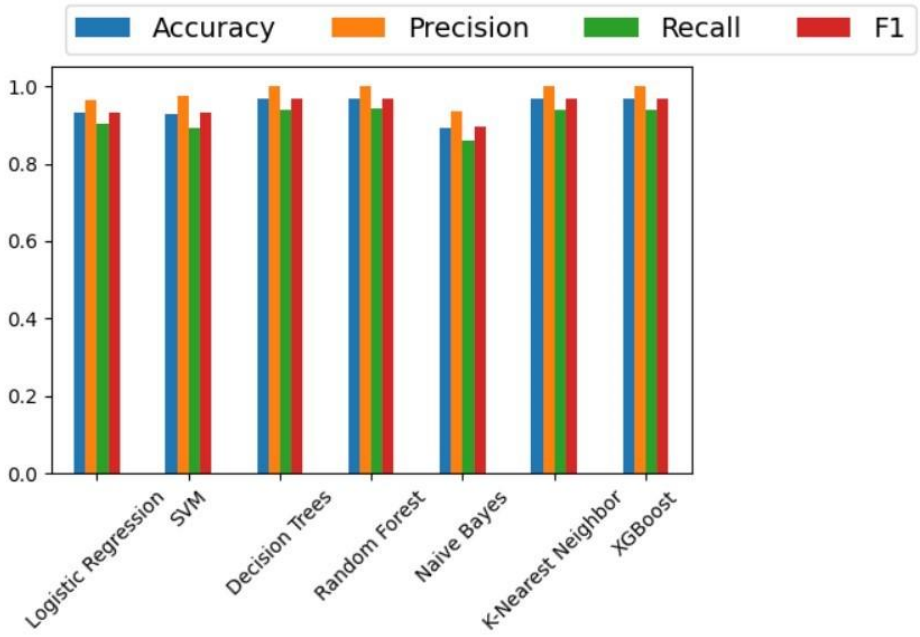
```
#Data is now evenly distributed
print(df_upsampled['inducted'].value_counts())
sns.countplot(df_upsampled['inducted'])
```

```
1.0    22775
0.0    22775
Name: inducted, dtype: int64

<AxesSubplot:xlabel='inducted', ylabel='count'>
```



My second analysis used this up-sampled data. While the accuracy of the models went down slightly the tradeoff was well worth it. The precision, recall, and F1 scores increased significantly. This showed me the importance of looking at metrics other than accuracy scores. Increasing the accuracy would not have created a better model. I needed to create a model that considered increases in recall and F1 scores.



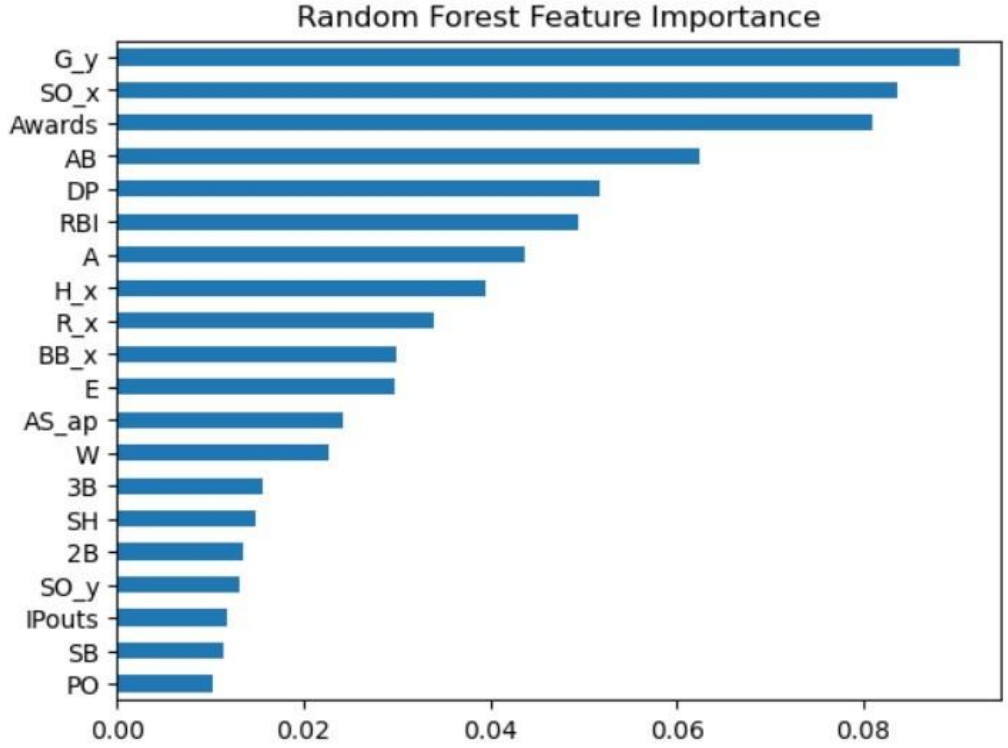
The chart above shows the significant increase in all the scoring metrics across all the algorithms trained on. Considering the importance of F1 and recall scores, for this analysis, XGBoost and Random Forest were my two best-performing models. With a slight edge to the Random Forest Model. Seeing how much of a difference the up-sampling made for my models, shows the significance of metrics other than accuracy when dealing with imbalanced data. It also showed me how simple techniques can make a substantial difference.

But I wanted to take it a step further. I was happy with the results above and wanted to try another technique for handling imbalanced data. The other technique I used was finding an optimal threshold that maximizes the area under the ROC curve. I created a loop with a step to walk through different thresholds, stopping at the threshold that maximizes the area under the ROC curve. Resulting in a threshold of 0.7. With the threshold applied to the Random Forest model, there was an increase in accuracy, recall, and F1 scores.

My best model was when the data was up-sampled and transformed with a standard scaler. Then trained on a Random Forest model with a .7 threshold. The accuracy was .969, the precision was 1, the recall was .942, and the F1 score was .97. In the table below this is the model labeled “RF.7”. The table below shows all the top-performing models from each of the different algorithms used.

	Accuracy	Precision	Recall	F1
Logistic Regression	0.930453	0.965529	0.902070	0.932722
SVM	0.928082	0.973444	0.892310	0.931113
Decision Trees	0.967597	1.000000	0.939059	0.968572
Random Forest	0.968739	1.000000	0.941079	0.969645
Naive Bayes	0.891113	0.936159	0.858548	0.895676
K-Nearest Neighbor	0.967422	1.000000	0.938749	0.968407
XGBoost	0.968476	1.000000	0.940612	0.969397
RF.7	0.969178	1.000000	0.941859	0.970059

I was incredibly happy with these results and moved on to looking at feature importance for my best model, Random Forest. Below we can see that games played, strikeouts as a batter, awards earned, at-bats, and double plays involved in as a fielder are the five most prominent features when predicting Hall of Fame baseball players. This was expected to me, people in the Hall of Fame played a lot throughout their careers and built-up lots of games awards, and at-bats. All three of these statistics come with longevity and consistency throughout a player's career.



This helps me answer one of my project questions. What statistics make a Hall of Fame player? Based on my analysis, playing a long, healthy career while minimizing the number of strikeouts, maximizing participation on the field, and being voted for awards are the things that make up Hall of Fame baseball players.

Now that I have identified these features. I can answer, who can use this knowledge and how can they use it? Coaches, teams, and staff can see how their players compare to Hall of Fame players and identify areas of improvement for players to grow. For scouts, people that look for the next budding talent, can use this model and upscale the data of the player they are scouting to see, if they have a lasting career, and what are the chances they will be a Hall of Fame player. That is what every team wants, talented players that win games and sell tickets/merchandise.

By using this model teams can scout potential next Hall of Fame players, and they can try to sign them to their team. In most sports scouting revolves around finding the best players and getting them on your team as soon as possible. With this algorithm, we can identify with certain probabilities how likely up-and-coming players are to be the next "great."

Once we have identified these players, we must keep them healthy and on track to succeed. As we saw earlier a long and consistent career is most important to make it into the Hall of Fame. So just because a player has high statistics if they have a short career; it means nothing as far as the Hall of Fame is conserved. Longevity and consistency are key.

A good amount of this project went as I expected. I knew the algorithms would learn to predict 0 and perform at 99% accuracy until I handled the imbalance. I assumed awards and all-star appearances would be the highest among feature importance. So, I was surprised when the all-star appearances feature importance was so low. What was shocking to me was how important precision, recall, and F1 scores can be. As well as not relying on the Accuracy score all the time as it can be misleading.

Some challenges I faced during this project was handling my imbalanced data. Which lead me to discover and use new techniques. Determining which scalar to use to transform the data. I always get stuck up on which scalars are best to use and when. The thing I had the most trouble with was getting the data into a usable format. Professors have been telling me that data science is 80% data cleaning and the more I work with my data sets the more a believe that

statement to be true. I learned how to effectively use aggregate functions to combine records as well as making sure I used the correct merge types, so I did not lose or gain any unwanted data.

Overall, this project was a success. I was able to train an algorithm that was able to predict a player's Hall of Fame status with about 97% accuracy, 100% precision, 94% recall, and 97% F1 score. I was able to pull out and interpret the most prominent features in this model. Which helped me to answer one of my research questions. Lastly, I was able to construct a valuable use case for baseball teams.

I learned a lot from this project. About baseball and how much thought goes into player statistics. About all the fun and interesting libraries that python has to offer to make data analysis visually appealing and more accessible. As well as all the intricacies of machine learning and hyperparameter tuning.