Andrew Fitzsimmons
11/17/2022
DSC478

Final Project - Executive Summary
Major League Baseball - All-Star Predictions

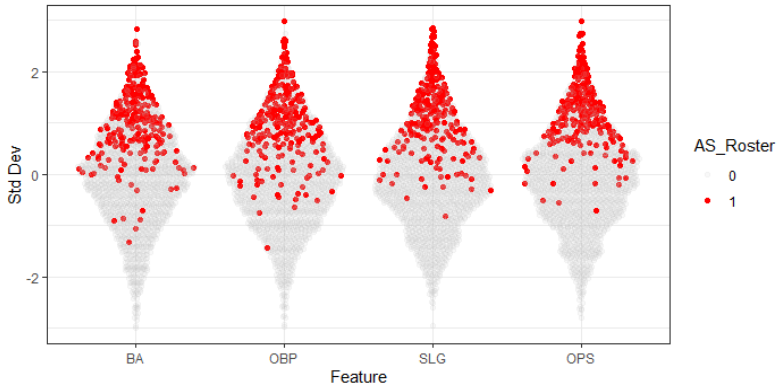
Team: Andrew Fitzsimmons - Graduate Student (Data Science)

Type of Project: Data Analysis

Project Goal
This project is a data analysis project that serves to analyze Major League Baseball offensive statistics by player. My goal was to build a machine learning algorithm to predict/classify whether a player will be selected to make the all-star team based on similarity to offensive statistics of players from previous years, and whether or not they were selected for the all-star team. I also will be performing qualitative cluster analysis using Kmeans clustering.

Data Cleaning/Preprocessing
I initially imported the data into R using the 'baseballr' package. The data was pulled using the 'daily_batter_bref' function, which pulls standard batting statistics for all players within a specified date range from baseballreference.com. The batting statistics needed to be joined with my manually collected dataset on which players made the all-star team for each year represented in the dataset.

Exploratory Visualization
I performed exploratory visualization in R. I performed a histogram analysis of my attributes on only the records of 'all-stars' in my dataset. I also created beeswarm plots to visualize the distribution of z-scores of all attributes for all players. I coded all-star players as red on the plots to better identify any patterns in attributes that separate all-stars from the rest of the players in the league. The image below shows that in all 4 of the primary offensive metrics (BA, OBP, SLG, and OPS), All-Stars tend to cluster to the higher extreme of the distribution.



Classification and Clustering

I utilized a KNN algorithm for my classification. I generated accuracy scores for various values of ‘k’ and plotted them in order to visually determine an optimal ‘k’ to use for my algorithm. I also tried different distance metrics (cosine vs euclidean). K value of 3, and euclidean distance metric were selected for the final model.

Since my dataset is biased towards non all-stars (12% of observations in the training data set are classified as all-stars), I did not use KNN to classify based solely on the majority class of ‘k’ nearest neighbors. Instead, I used each instance of an all-star in the neighborhood as a ‘vote’ towards that player’s all-star selection. I then calculated a ‘total distance’ metric for each player that received votes by summing the distances of all all-stars in the neighborhood. Finally, players were ranked from most votes to least, and sorted within their ‘# of votes’ groups from lowest to highest based on the ‘total distance’ metric.

I also performed an exploratory Kmeans cluster analysis on the dataset to identify and analyze interesting groupings of players by centroid attributes by returning the top ‘n’ nearest players to each cluster centroid. I ran kmeans multiple times for different values of ‘k’. I ultimately selected k=4 since it seemed to give the most interesting segmentation of player groups.

Conclusion

From the final sort of my KNN results (sorted by number of votes and total distance of voters), I selected the top 60 players as my all-star predictions. Each all-star team (American League and National League) has 25 offensive players each. I chose to select 60 total players for my prediction set because I feel my algorithm would be most useful not in directly picking all-stars, but rather to suggest a pool from which likely all-star candidates will be drawn. For this reason, my all-star selection pool was 10% greater than the total roster capacities.

When running the KNN algorithm on the test set of 2022 MLB players, with k=3, I was able to correctly predict 66% of the 2022 MLB All-Stars. I am happy with the performance of the KNN algorithm in this application, especially considering that All-Star targets only make up 12% of the overall training data population.

My Kmeans cluster analysis returned some interesting groupings of players. The top 5 players nearest to each centroid seemed to be fairly well defined by their grouping. The kmeans clustering did a good job to separate ‘power hitters’ from ‘base hitters’. Centroids 1 and 2 appeared to be the most useful since they provided unique separation between ‘good’ players. Whereas Centroids 0 and 3 appeared to separate the more massive part of the data set which is average to below average players, and players with minimal Plate Appearances.