Capstone Milestone Report

Table of Contents

[Introduction 2](#_Toc492997932)

[Data Story 2](#_Toc492997933)

[Preliminary Findings 6](#_Toc492997934)

[Future Approach 7](#_Toc492997935)

# Introduction

Sports analytics is a common and well-known application of data science. The prevalence of data analysis and statistics in baseball stands out. The main goal of sports is winning, and the front offices of many teams employ analytics to increase their team’s chances of doing so. For instance, analytics can help project prospects in the minors and identify trends in player performance.

A common complaint of casual baseball fans is that the games are boring. Often in baseball, there are long periods of little action and excitement. A good measure of excitement for casual fans is run scoring; batting is typically considered more interesting than fielding (offense vs. defense). Perhaps if these casual fans could determine which innings are more likely to feature scoring, they would get more interested in baseball.

In this project, the goal is to create a model to predict whether a half-inning will feature run scoring based only on information available at the beginning of the inning. Potential clients include fans who have limited time to watch games, casual fans, and advertising companies interested in seeing when the most exciting parts of the game occur for maximum viewership. In addition, baseball teams and managers may be interested in predicting the likelihood of scoring, and thus making changes to improve their chances of scoring and thus increase their chances of winning.

# Data Story

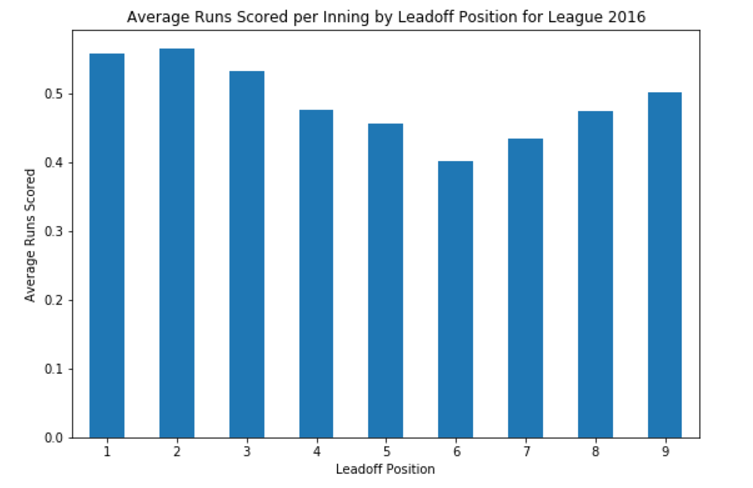
Play-by-play data for every game and every season is available through Retrosheet.org. The available data includes type of hit, out, every player on the field, positions, etc.

The play-by-play data is available in event file format from Retrosheet.org. The downloads may be parsed into .csv files using programs they provide. An alternative is to use R and a set of files known as Chadwick to parse a season's play-by-play data into a single .csv file, as detailed at http://isaacmiller.co/how-to-create-a-single-season-csv-of-retrosheet-play-by-play-data-pc/.

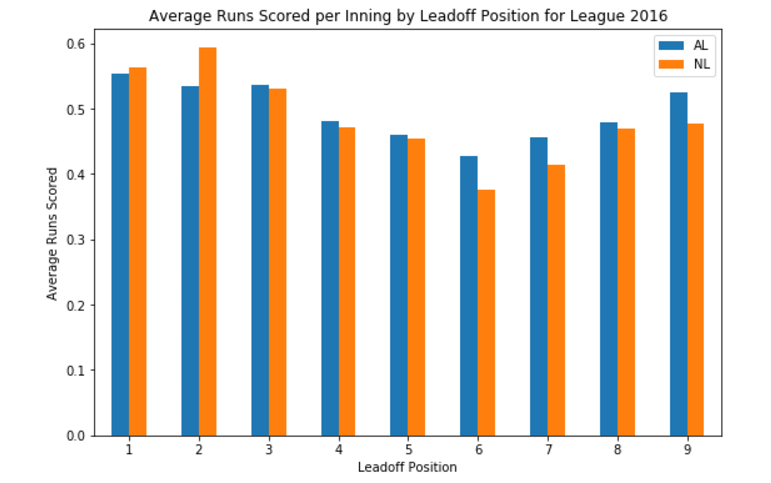
Since there are over 90 fields available, a subset of features are selected to make the data easier to work with.

One field that could be of interest in predicting whether there will be scoring in an inning is the batting order position of the batter leading off. Managers often place their best hitters in the 3-5 spots, with the worst hitters in the 8th position in the lineup. However, the order players come up to bat is not the same for every inning; only the 1st inning is guaranteed to have batters come up in the original 1-9 order. Do innings where the listed leadoff hitter actually leads off contribute to more runs and a better chance at winning? Is it possible to predict the number of runs scored in an inning based on features such as batting position leading off, inning, etc.? A prospective viewer may, depending on what kind of hitter is leading off, pay more attention in innings with higher scoring-chance.

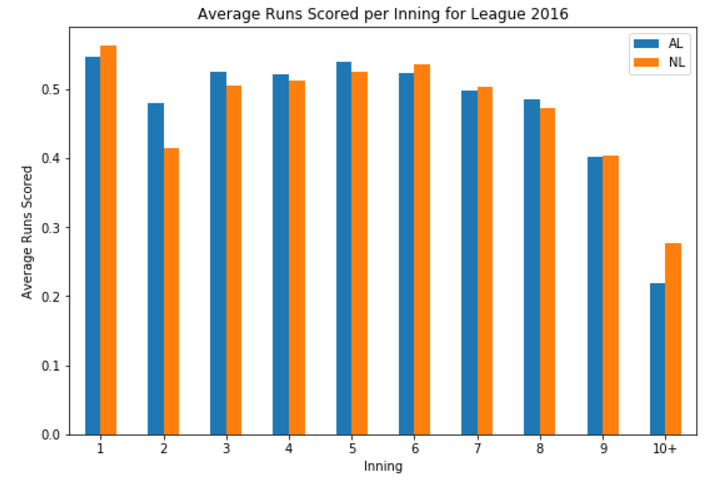
The following plot shows the average runs scored per inning depending on which order position is leading off for the 2016 MLB season:



Since the AL uses the designated hitter in place of the poor-hitting pitchers, it may be beneficial to consider the league of the team batting:



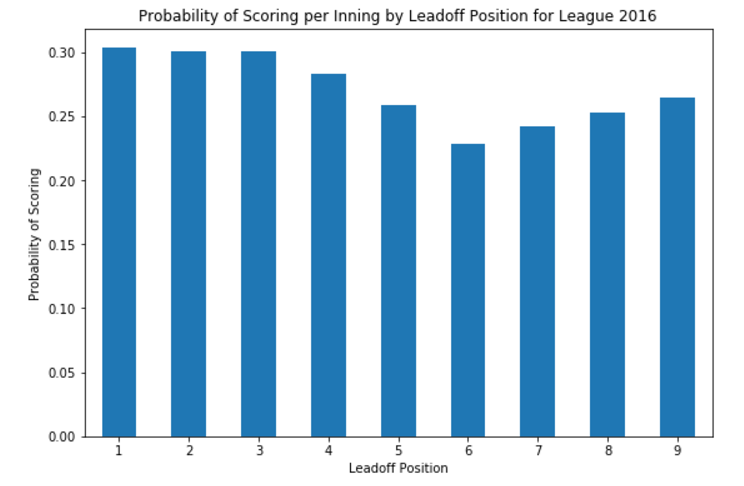
Another potential interesting feature is the inning number itself. For the purposes of the following plot, extra innings are grouped into the same bin in order to make a clearer plot:

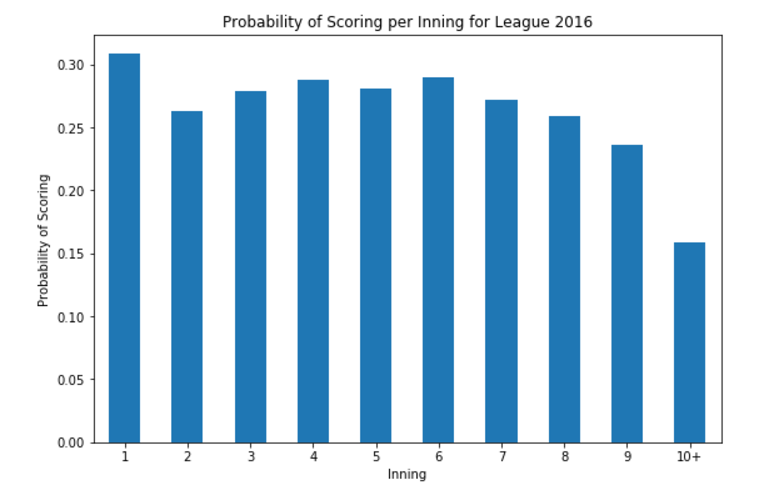


From the visualizations, for hits per inning, there is a slight drop for the second inning before going up a little into the mid-innings, then trending down from there. The same could be said for runs per inning. Unsurprisingly, the presence of the DH allowed the AL teams to have slightly better offensive numbers.

When considering leadoff positions, there was a similar trend in that starting with the top of the order led to more runs, and leading off with the 6th spot led to fewer runs. Number of hits also followed this trend, although the effect is less obvious. Surprisingly, NL teams did not average the fewest amount of runs for when the 9 hitter, usually the pitcher, leads off. Even if the first batter is almost a free out, the top of the lineup is still able to produce runs better than if the bottom of the order was due up.

Another value to plot is the probability of actually scoring runs dependent on the leadoff position or inning. This value will later be used as the Y value for the models.





From these plots, it is fairly clear that the trends are similar to the average runs scored per inning plots from before.

The dataset does not provide detailed player data. For instance, there is no data available that references player batting average, which could easily help predict the likelihood of scoring depending on who will be coming up to bat in a particular inning. There are also game logs available on Retrosheet. This data can provide additional information on games that is typically fixed for each game, such as whether the game was a day or night game, who the umpires were, etc. The game log data could be used to obtain several additional features that can be considered for use in the modeling.

# Preliminary Findings

An initial attempt at a model requires first cleaning the data such that the rows are all innings instead of plays. The rows of the play-by-play data for the whole 2016 season is iterated through to count how many runs and hits that occurred in each inning. The counts are taken and then a True/False flag is generated for each inning to specify whether any runs are scored. Each inning thus has various features that are known at the beginning of the inning, along with the dependent variables of number of runs/hits that occurred and whether any runs/hits occurred.

The first model considered for use is Logistic Regression with the Y value of the T/F of there being runs scored for a particular inning, as that is a binary variable to predict. The initial features considered are inning, leadoff position, and score differential, which is how far ahead or behind the team batting is when their half of the inning to bat starts.

Since the inning number and the leadoff position are integers yet categorical variables, the model is first tested while encoding the variables using the One Hot Encoder. These two features are encoded and combined with the score differential to generate a model to predict whether or not runs are scored in a particular inning. This generates an accuracy of around 0.72. Then, the inning and leadoff position are allowed to remain as numerical variables, and the model is generated again. The accuracy is shown to not change much.

The accuracy of 0.72 seems decent, but when examining precision and recall, the scores are both 0. Taking a look at the confusion matrix, reproduced below, shows the problem:

|  |  |  |
| --- | --- | --- |
| **Runs scored?** | **False** | **True** |
| **False** | 7904 | 2961 |
| **True** | 0 | 0 |

The model as is currently predicts every inning to be scoreless when it is fit to the test data portion of the train-test split.

Sklearn’s GridSearchCV method is used to determine the best value for the parameter C in an attempt to improve the model; the best C is found to be 0.001. However, the resulting accuracy, precision, and recall remain the same as before.

Due to the problems with using Logistic Regression, we decide to test an alternative modeling algorithm, the Random Forest Classifier. The algorithm is run on the same training and test data split as before, with the same features. The results are compared to Logistic Regression:

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Accuracy** | **Precison** | **Recall** |
| **Logistic Regression** | 0.727473538886 | 0.0 | 0.0 |
| **Random Forest** | 0.716244822826 | 0.031746031746 | 0.303225806452 |

Also, here is the confusion matrix for Random Forest:

|  |  |  |
| --- | --- | --- |
| **Runs scored?** | **False** | **True** |
| **False** | 7700 | 2888 |
| **True** | 204 | 73 |

The Random Forest model shows that there is now predictions for True in the model, an improvement over Logistic Regression. However, the precision and accuracy scores are still low, so more tuning is needed.

# Future Approach

It is clear that there needs to be improvements in the modeling. More models can be considered, and more parameter tuning can be completed. More features can be added to the model for consideration as well. These features can include which team is batting, whether it is the home or away team, etc. Some of these features will need to be encoded.

One factor to consider is the fact that the data is slightly imbalanced, as most of the innings were scoreless. Therefore, there were many more “False” flags than “True” flags. Some methods to rectify this include stratification, oversampling, and undersampling. Hopefully, these methods will be able to improve the precision and recall scores to decent, acceptable levels while deciding on which modeling algorithm is the best.