MLB Stolen Base Predictions

Stat 451

Waleed Almousa & Colin Macy

I. Introduction

An important yet often overlooked statistic in baseball is a team's stolen base attempt success rate. The rate of stolen bases in the MLB rose this year as a result of two major rule changes. Those changes were increasing the size of the bases and adding a pitch clock. The purpose of this report is to answer the following questions: Can we predict the outcome of a stolen base attempt, and which factors influence the success of a stolen base attempt the most?

II. Data Description

To begin, we created a custom dataset that consists of data pulled from MLB's statcast database. It consists of data from their catcher throwing, pitch tempo, sprint speed, and pitcher running game leaderboards. Additionally we accessed the data for each stolen base attempt using an API call to baseballsavant.mlb.com which hosts their statcast data.

The data set consisted of 2798 rows and 15 columns. Columns game_pk, at_bat_number, runner_id, pitcher_id, and catcher_id were used for merging the various data sets together, but were not used for any analysis. The column successful_sb is our target variable. The columns runner_sprint_speed, pitch_tempo, sb_att_rate, lead_dist_gained_opp, lead_dist_gained_att, pop_time, exchange_time, arm_strength, and cs_aa_per_throw are the input variables.

successful_sb	runner_sprint_speed	pitch_tempo	sb_att_rate	lead_dist_gained_opp	pop_time	exchange_time	arm_strength	cs_aa_per_throw
1	28.0	17.433	0.023847	1.847397	1.988417	0.644583	78.32061	0.175065
0	27.8	17.433	0.023847	1.847397	1.988417	0.644583	78.32061	0.175065
1	29.2	17.433	0.023847	1.847397	1.988417	0.644583	78.32061	0.175065
1	30.4	15.792	0.014737	1.715638	1.988417	0.644583	78.32061	0.175065
1	27.1	17.923	0.012097	3.617790	1.988417	0.644583	78.32061	0.175065

Figure 1. First 5 rows of dataset with relevant columns

After feature selection (discussed in the next part) we landed on four variables: runner_sprint_speed (average sprint speed for runner in ft/sec.), lead_dist_gained_opp (distance a runner has advanced from the start of delivery to pitch release (in feet), for all stolen base opportunities), lead_dist_gained_att (same as lead_dist_gained_opp, but only for stolen base attempts), and cs_aa_per_throw (The CS Above Avg. gained per throw compared to the expectation of an average catcher. Calculated by CS Above Avg. / Throw Opportunities.)

III. Feature Selection

Out of our nine variables, we wanted to optimize the input values by choosing the four most impactful variables. We ran permutation importance to do so and selected the four variables to use for our models. This resulted in cs_aa_per_throw, lead_dist_gained_att, lead_dist_gained_opp, and runner_sprint_speed being selected.

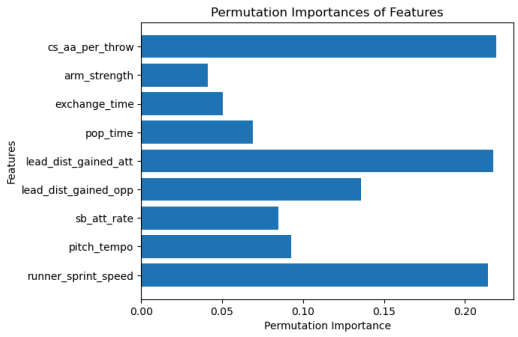

Figure 2. Barplot of Feature Importance

IV. Building A Model

Upon the first attempts at running our models, we noticed they failed to perform any better than guessing a successful stolen base for every observation. This was due to the lack of data since we had only a year's worth of stolen base attempts and over 80% of attempts were successful stolen bases. To combat this, we oversampled our data so that we would have more unsuccessful observations and a better distribution of data to work with. The histograms below illustrate our data before and after resampling.

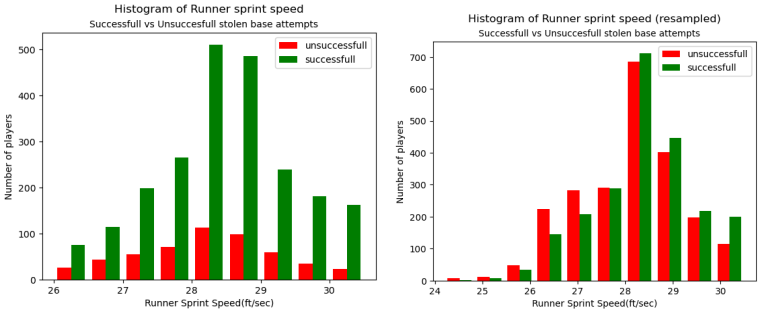

Figure 3. Histogram of runner_sprint_speed before and after resampling

We then used GridSearchCV to find our hyperparameters and test on 5 different models - SVC, LogisticRegression, DecisionTreeClassifier, KNeighborsClassifier, and RandomForestClassifier. Our best model was RandomForestClassifier, with max_depth set to 100 and n_estimators set to 100. This model had an accuracy of 0.896 on the validation data, which is roughly 0.25 more accurate than most of the other models. Figure 4 shows the accuracy of each of our models and represents what models work best for the sake of our analysis.

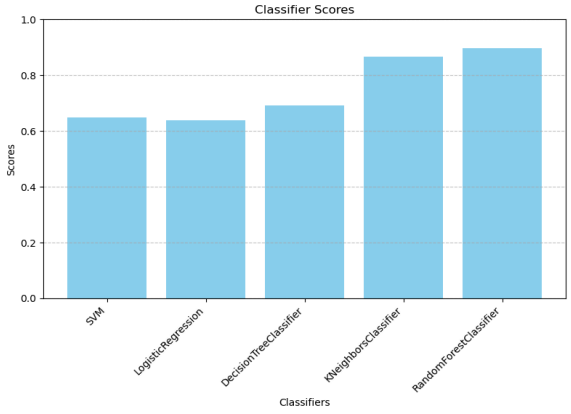

Figure 4. Barplot of model accuracies

V. Results

Upon using our test data on the random forest model we created, we received an accuracy score of 0.88 for our model. Even more important though was our precision score, as it was 0.97. When predicting stolen bases, precision score is the most important metric as it represents how often our model correctly predicts a successful attempt. For a coach, this would go a long way to minimize stolen base risk and ensure that when he decides to attempt a steal, the baserunner has a high probability of making it successfully.

VI. Conclusion

Due to the rule changes in the MLB and the increased number of stolen bases, we sought to find a model that could predict the outcome of a stolen base attempt at a high rate. We implemented the statistical methods of feature selection and oversampling to strengthen our model. Model accuracy was important, but precision is the most meaningful measurement of model success. In this context, precision represents the coach telling the player to steal and the steal being successful. With a precision of 97%, MLB teams could gain a competitive advantage over their opponent by utilizing our model.