The Effect of Lead Distance on the Expected Probability of an Out on the Basepaths

By Jake Sauberman

**Summary:**

Since 2015, runners on first base attempting a steal averaged a 10.95-foot lead, while runners staying put averaged 9.78 feet. Runners adjust their typical lead distances when they know they are going to try to steal second base in order to give themselves the best possible success rate. Too far and they risk a pickoff; too close and they increase their chances of getting caught stealing.

By leveraging two logistic regression models based on in-game data, the “optimal” lead distance is calculated for a runner attempting a steal of second base based on characteristics of the runner himself, the pitcher, the catcher, and the batter. The optimal lead is defined as the lead distance which minimizes the predicted pickoff probability plus the predicted caught stealing probability.

**Background:**

When a baserunner prepares to steal second base, he takes a primary lead off the base. That is, as the opposing pitcher stands on the rubber prior to the delivery of the pitch, the runner positions himself as close to the next base as he feels comfortable. That level of comfort is determined by the runner’s ability to get back to first base should the pitcher step off the rubber without delivering a pitch, and instead throw to the first baseman in attempt to tag the runner out between the bases. Such a move is referred to as a pickoff attempt.

With each incremental inch he distances himself from the base, the probability of making an out on the basepaths changes in two major ways. The further away from first base the runner is when he begins his steal attempt, the less distance he needs to cover to reach second base, and therefore a higher chance of a successful steal. However, the further away from first base the runner gets, the danger of potentially getting picked off increases. An increased lead means a larger distance to get back to first base on a pickoff attempt, and therefore a higher chance of a pickoff out. Thus, the lead distance has an inherent risk-reward relationship in two opposite directions.

With these two relationships in mind, we infer that a larger lead with the intention of a stolen base attempt simultaneously means a greater chance of getting thrown out on a pickoff attempt while a lesser chance of getting thrown out on the stolen base attempt. Where does the optimal lead distance lie for a specific runner on first base to maximize his stolen base success and minimize his pickoff probability? What are the situational factors that hold the most influence over this equation? These are the questions that this project seeks to answer.

To do so, we look at the characteristics of four different actors that have varying degrees of impact on the outcome. The runner, of course, has some inherent stolen base ability and speed attributes. Different pitchers have qualities that can limit the effects of the running game in similar ways to different catchers – by limiting attempts and preventing success. Even the batter may have an impact, by causing more or less concentration from the pitcher and catcher, and thereby more or less concentration on the runner. All information that we feed into the model will be information that would theoretically be available to a baserunner in real-time in a game situation.

The context we are evaluating is assuming the runner intends to attempt a steal on the next pitch. Whether or not the runner *should* attempt a steal is a different question altogether and will not be included in the scope of this project.

*Goal: We want to be able to predict the optimal lead distance for a runner on first base with the intent to steal second, based off the contextual information available in real-time. These predictions will feed into an app that calculates the optimal lead distance when given the identities of the runner, pitcher, catcher, and batter.*

**Methodology:**

In order to find the point of optimization in terms of lead distance, we must first create two separate models for expected pickoff probability and expected caught stealing probability. With these two models estimating the likelihood of an out on the basepaths in hand, we can then back-solve for the lead distance that minimizes the sum of the two out probabilities. The methodologies for the two models are nearly the same, save for some context-specific tweaking.

The dataset for the following analysis comes primarily from Statcast, which dates back to the 2015 season. Career stolen base success data comes from FanGraphs.

For the pickoff model, since pickoff attempts could not be identified from the data, the data was queried to include all MLB regular season plays from 2015 through 2019 with a runner on first, with second and third base open. Thus, each row has pitch-level data on pitches that either ended a plate appearance (hit, strikeout, field out, walk, etc.) or had a baserunner event (stolen base, picked off, etc.). Additionally, the game score at the time of the event had to be within a five-run lead/deficit. This should help avoid situations where defenses feel unobligated to defend against stolen bases. This data had 106,364 such events.

For the caught stealing model, the data was queried to include all MLB regular season stolen base attempts from 2015 through 2019 with a runner on first, with second and third base open. Again, this only includes data where the game score was within five runs. Thus, each row has pitch-level data on all steal attempts of second base, both successful and otherwise. This data had 5,295 such events for the model.

*Step 1: Identify Features*

The first important aspect of feature selection is considering what the result of these models will look like. We want to be able to predict the optimal lead distance for a runner on first base with the intent to steal second, based off the contextual information available in real-time. Therefore, any information that the model takes in has to be information that we could have known about before the play takes place, with two key exceptions – the outcome of the play (safe or out) and the variable in question, the lead distance. We need the outcome for the model’s algorithm to learn predictions, and we need the lead distance as the independent variable for the back-solving.

When considering the potentially influential variables for the pickoff model and the caught stealing model, one must look at the scenario from the perspective of four actors: the runner, the pitcher, the catcher, and the batter. The following features were tested in the initial versions of the model to check for significance:

Runner:

* Lead distance
  + Actual lead distance that the runner on first base took on the pitch
  + Measured in feet, from Statcast
* Average sprint speed
  + Measured in feet per second
  + Calculated by averaging the players’ sprint speeds from home to first on ground balls hit with an exit velocity under 90 miles per hour
  + Estimates how fast the runner travels at top speed between first and second base
* Average 10-foot acceleration
  + Measured in feet per second squared
  + Calculated by taking Statcast’s 90-foot sprint splits from the 0-10 foot interval, solving for acceleration given the distance (10 feet), time (Statcast value), and initial velocity (0 ft/s)
  + Estimates how quickly the runner can reach top sprint speed
* Stolen base ability
  + Calculated using player stolen base data from Fangraphs over a player’s career
  + Uses *The Book*’s method of finding true platoon splits by regressing each player’s career stolen base percentage to the mean
    - Increases the sample size by 30 attempts at the league-average stolen base success rate (72.43%) to estimate a player’s true stolen base ability
    - Formula: *(0.7243 \* 30 + stolen base percentage \* stolen base attempts) / (stolen base attempts + 30)*
  + Divided by the league-average stolen base success rate to normalize the metric
  + Estimates how much better than average a player is at stealing bases
* Stolen base attempts per 600 plate appearances
  + Calculated using player stolen base data from Fangraphs over a player’s career
  + Converts stolen base attempts to a “per 600 plate appearance” basis to account for differing sample sizes among players
  + Estimates stolen base attempt frequency

Pitcher:

* Average lead distance allowed
  + Calculated by averaging all opposing runner leads from first base when second and third base were empty
  + Estimates how well a pitcher “holds runners on”
* Stolen base prevention ability
  + Same methodology as “stolen base ability” but using opposing runners when each pitcher was on the mound
* Stolen base attempts allowed per 600 plate appearances
  + Same methodology as “stolen base attempts per 600 plate appearances” but using opposing runners when each pitcher was on the mound
* Average zone time
  + Calculated by averaging the time between the pitch leaving the pitcher’s hand and crossing the plate
  + Estimates any marginal time the runner gains from a low velocity/high off-speed pitch mixes versus a high velocity/high fastball pitch mix
* Percentage of off-speed pitches
  + Calculated by taking the sum of the frequency of sliders, curveballs, changeups, splitters, and knuckleballs in a pitcher’s mix
  + Estimates any marginal time the runner gains from a low velocity/high off-speed pitch mixes versus a high velocity/high fastball pitch mix
* Pitcher handedness
  + Coded as 1 for a left-handed pitcher, 0 for a right-handed pitcher

Catcher:

* Average pop time
  + Calculated by averaging catcher pop times from 2015-19, as provided by Statcast
* Average throwing accuracy
  + Calculated by averaging catcher throwing accuracies on stolen base attempts of second base from 2015-19, as provided by Statcast
* Stolen base prevention ability
  + Same methodology as “stolen base ability” but using opposing runners when each catcher was behind the plate
* Stolen base frequency allowed
  + Same methodology as “stolen base attempts per 600 plate appearances” but using opposing runners when each catcher was behind the plate

Batter:

* wOBA
  + Calculated each batter’s wOBA from 2015-19, using 2019 linear weights for simplicity
  + Estimates the level of skill of the batter at the plate, which may draw focus away from the runner as skill increases

Of all these potential features, the following were selected in each model, due to a significance level (p-value) under 0.01. Thus, we can be at least 99% confident that these features are significant factors in predicting a pickoff or caught stealing event. By taking the absolute value of the t-statistic for each feature, we can rank each model’s features in terms of importance.

*Pickoff Model features*

|  |  |  |
| --- | --- | --- |
| **Feature Name** | **p-value** | **Importance** |
| Lead distance | 2.25 x 10-11 | 48.443 |
| Average lead against | < 2 x 10-16 | 34.756 |
| Pitcher handedness | < 2 x 10-16 | 33.550 |
| Stolen base attempts per 600 - Runner | < 2 x 10-16 | 23.038 |
| Average sprint speed | < 2 x 10-16 | 15.248 |
| Acceleration | 1.22 x 10-6 | 4.852 |

Unsurprisingly, the relevant features to the pickoff model only pertain to the runner and the pitcher. However, the runner’s speed and acceleration abilities do not appear to be as important to the model as the qualities of the pitcher.

*Caught Stealing Model features*

|  |  |  |
| --- | --- | --- |
| **Feature Name** | **p-value** | **Importance** |
| Stolen base prevention ability - Pitcher | < 2 x 10-16 | 26.827 |
| Stolen base ability - Runner | < 2 x 10-16 | 20.621 |
| Stolen base prevention ability - Catcher | < 2 x 10-16 | 19.468 |
| Lead distance | < 2 x 10-16 | 10.681 |
| wOBA | < 2 x 10-16 | 8.744 |
| Stolen base attempts per 600 - Runner | 1.03 x 10-9 | 6.105 |
| Average sprint speed | 3.26 x 10-8 | 5.527 |
| Stolen base attempts per 600 - Catcher | 6.21 x 10-4 | 3.422 |

The top three features in the caught stealing model pertain to the stolen base ability of the three main actors in the stolen base event. Perhaps contrary to expectations, the model values the pitcher’s ability to prevent stolen bases over the runner’s ability to steal or the catcher’s ability to prevent steals.

A screenshot of a cell phone

Description automatically generatedA screenshot of a cell phone

Description automatically generated

*The correlation plot for the pickoff model is on the left; the correlation plot for the caught stealing model is on the right.*

Looking at the correlation plots for the selected features in both models, the relationships between singular features are generally not strong. However, we know that the p-values are highly significant for these features, meaning that in the context of the totality of the features, their relationships are more meaningful.

*Step 2: Build and Run Model*

In both the pickoff and caught stealing models, the algorithm will read in the training data and predict a probabilistic outcome between 0 (safe) and 1 (pickoff/caught stealing). To achieve that outcome, we choose logistic regression as the model type to build.

The methodology uses k-fold cross validation where k = 3, where two-thirds of the dataset was split into the training set, while the remaining one-third was set aside as the testing data. This is run three times, so that every data entry ends up in one of the three testing sets. Such a method will help to reduce bias in the results by increasing the sample size through re-sampling.

However, there is an existing issue in the datasets that needs to be addressed before proceeding with the model building. In the pickoff dataset, the minority class, or the rows with pickoff = 1, makes up just 0.65% of the total data. In the caught stealing dataset, the minority class, or the rows with caught stealing = 1, makes up just 24.05% of the total data. There is a severe imbalance between the majority and minority classes, and since we do not have millions of rows in the dataset to begin with, there is a relatively small sample size of the minority class. Such a sample size can lead to biased outcomes, or the machine learning algorithm may largely ignore the minority class. To address this issue, we use the Synthetic Minority Oversampling Technique, or SMOTE, in the DMwR package in R. This creates new examples of the minority class in the training set by selecting examples that are nearby in the feature space, drawing a line between the examples, and creating a new example at a randomly selected point along the line.

For the pickoff dataset, remember that the data comes from all plays with a runner on first and second and third empty. However, the question this project seeks to answer is within the confines of knowing that the runner is going to try and steal second base. When a runner is about to attempt a steal, there will generally be greater attention paid by the pitcher, and thus the odds of a pickoff attempt should increase. We see that relationship reflected in the data – in the pickoff dataset, a pickoff occurs 0.65% of the time. But since 2015, a pickoff occurs roughly every 10 stolen base attempts on average. Thus, to try and replicate that environment within the pickoff dataset, we can use SMOTE to set the ratio between the majority and minority class to 10:1. At the same time, the most fundamental reasoning for using SMOTE was to synthetically oversample on the minority class. By setting the oversampling percent of the minority class to 2000% and the ratio of majority-to-minority to 10:1, we achieve both goals.



For the caught stealing dataset, both issues of minority sample size and class imbalance are much less pronounced than in the pickoff dataset. Nonetheless, the original minority sample of 1389 caught stealing events can be synthetically boosted for more accurate readings. By setting the oversampling percent of the minority class to 500% and the ratio of majority-to-minority to reflect the league-wide 28% caught stealing rate from 2015-19, the data is transformed as shown below.



With the data properly transformed to include acceptable minority class sample sizes and mirror real-world relationships between the majority and minority classes, the logistic regression models can be built. The ‘glm’ function in R with the aforementioned features to predict pickoff/caught stealing built three logistic regression models each (3-fold cross validation) for the pickoff and caught stealing models based off the training sets. Predictions from the three models were then put into the three testing sets, which were combined to form the holistic original data set with predictions.

*Step 3: Evaluate Model*

Due to the severe class imbalance of the dataset (the ratio of pickoff or caught stealing events to safe events), typical logistic regression accuracy scores are not helpful for model evaluation. In what is referred to as the “accuracy paradox”, the confusion matrix is overwhelmed by the number of true negatives (non-pickoff or non-caught stealing events that are accurately predicted as such) and the accuracy metric reported is arbitrarily high.

The summary of the evaluation of both models can be found below:

|  |  |  |
| --- | --- | --- |
|  | **Pickoff Model** | **Caught Stealing Model** |
| McFadden’s Psuedo-R2 | 0.1293 | 0.1111 |
| Cox and Snell Psuedo-R2 | 0.0757 | 0.1221 |
| Nagelkerke Psuedo-R2 | 0.1660 | 0.1769 |
| Likelihood Ratio p-value | 0 | 0 |
| ROC AUC | 0.755 | 0.711 |

We use three different pseudo-R2 metrics to evaluate the logistic regression models. Pseudo-R2 metrics do not explain what percentage of the relationship is explained by the model like R2 does in linear regression; rather, they are used to compare different models trained on the same dataset. However, a common rule of thumb for interpreting McFadden’s pseudo-R2 is that a value between 0.2 and 0.4 is considered an excellent fit.

With a p-value of zero, we know that the amount of predictive power that does exist is not by coincidence, and that the features selected as predictors for the model are highly significant. The ROC AUC values of 0.753 and 0.711, respectively, indicate a fair performance in separating a 0 outcome from a 1 outcome. As shown below, there is a distinctive difference between the model’s ROC curve (in black) and the null ROC curve based on a random guess (in gray). We can be very confident that, for all its potential imperfections, the model does a better job in predicting caught stealing and pickoff events than a random guess.

A close up of a map

Description automatically generatedA close up of a map

Description automatically generated

*The ROC curve for the pickoff model is on the left; the ROC curve for the caught stealing model is on the right.*

Ultimately, outside of the p-value, these evaluation metrics are not as high as we would like them to be but considering the mission of predicting the outcome only equipped with knowledge that the baserunner would have *before* the attempt, a fair-not-excellent performance is to be expected. Nonetheless, it is something to consider when making predictions based off these models.

*Step 4: Find Optimal Lead Distance*

The two models’ outputs of expected pickoff and caught stealing probabilities are part of the equation we need to find a given situation’s optimal lead distance. The function needs to take in the current runner, pitcher, catcher and batter, and return the optimal lead distance and the expected out risk (expected pickoff probability plus expected caught stealing probability). The optimal lead distance will be that which minimizes the expected out risk.

To do so, a data frame is created, with all possible lead distances ranging from six feet to 15 feet in increments of 0.01. The other columns are constants, representing each feature of the models based on the player inputs. Both models are run on this data frame, creating predictions for each row of data. The two model outputs are summed up in a subsequent column, and the function extracts the lead distance with the minimum sum.

This algorithm is then run on actual stolen base data, for all regular season attempted steals of second base since 2015 with no runners on second or third base. Thus, we can compare actual lead distances with what the algorithm says should have been the runner’s lead distance to minimize their chances of making an out on the basepaths.

**Results:**

Looking at the distributions between the actual lead distances prior to attempted steals of second base verses the algorithm’s optimal lead distances, one aspect immediately jumps out.

A picture containing screenshot

Description automatically generated

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | Min | 1st Qu. | Median | Mean | 3rd Qu. | Max |
| Actual | 0.127 | 10.214 | 10.952 | 10.955 | 11.735 | 23.681 |
| Optimal | 6.000 | 10.260 | 11.480 | 11.465 | 12.700 | 15.000 |

The interquartile range in the actual lead distance distribution is just 1.521 feet, as opposed to 2.440 feet in the optimal distribution. With all the different combinations of possible runners, pitchers, catchers, and batters, how is it that 50 percent of lead distances are within 18 inches? Perhaps there is a need for experimentation in this area after all.

The other major difference is that the optimal lead distribution is slightly higher, ignoring outliers. The optimal first quartile lead distance is 0.046 feet higher; the mean lead distance is 0.51 feet higher; and the third quartile lead distance is 0.965 feet higher. If these optimal lead distances are true, then there is an opportunity for runners on average to take slightly higher leads than they currently do. Let’s break it down by individual runners (minimum 10 stolen base attempts of second base since 2015).



Just from the names on these lists, the correlation between running speed and the difference between optimal and actual lead differences appears to be strong. The top 10 largest positive differences belong to slower-footed runners, while the top 10 largest negative differences belong to some of the game’s elite base-stealers. Digging into what caused this trend, the reason becomes apparent within the pickoff model.

By analyzing the original pickoff dataset, we can find the underlying trend that causes the model to penalize faster runners. Players were grouped into average sprint speed buckets, and pickoff percentage was calculated for each bucket. Pickoff percentage takes all successful pickoffs and divides them by the number of plays where the runner was on first base with second and third base empty.

Faster runners get picked off at a higher rate – roughly twice the frequency – than slower runners. At first glance, this may seem counterintuitive. Theoretically, faster runners should be able to get back to the base quicker and should therefore be picked off less frequently. However, the data above shows an underlying trend, that faster runners will draw the attention of the pitcher, culminating in a higher likelihood of a pickoff attempt. An opposing pitcher is much more likely to be wary of a Byron Buxton over a Francisco Cervelli, after all. One is a legitimate threat to attempt a steal, the other is lucky to steal a couple of bases per year. The coefficient of the runner’s average sprint speed in the pickoff model is 0.145359. It is positive, meaning the higher the average sprint speed, the higher the chances of getting picked off first base. Pickoff attempt frequency appears to be a function of the opposing teams’ perception that the runner will attempt a stolen base. It is more a tool to suppress lead distances and mess up runner jump timing than it is a tool to actually pick off runners. That theory is tested below.

Below is a scatterplot of players’ average sprint speed versus their average *actual* lead. Notice the weak positive linear trend between the variables, with a correlation of 0.22. There is not conclusive evidence to suggest that faster players take bigger leads than the average player, or that slower players take smaller leads.

A picture containing photo, different, table, white

Description automatically generated

If the actual lead distances are more or less the same for faster players versus slower players, but the pickoff percentages are noticeably higher, then the logical conclusion is that pickoff attempts have little to do with the actual lead distance, and more to do with the perception of the runner. Pitchers attempt pickoffs much more frequently for faster runners than for slower runners because they are a greater threat. Real in-game pickoff attempt data would give greater certainty on the matter, but that conclusion appears to hold up given the dataset we have.

Now compare the above scatterplot to the one below, representing the players’ average sprint speed versus their average *optimal* lead. Notice the stronger negative linear trend between the variables, arriving at a correlation of -0.61.

A close up of a map

Description automatically generated

For slower runners, the risk-reward payoff tilts in the favor of larger leads, where they stand to gain more in the way of a successful steal than they do to lose in a pickoff. On the flip side, faster runners with greater stolen base ability will rely less on their lead distance. Pitchers watch them closer, so any incremental gain in a lead is more likely to be noticed and acted upon. In their case, the algorithm deems the risk-reward payoff in favor of smaller leads, where avoiding a pickoff is paramount to give them a chance to steal.

The algorithm was fed into a Shiny app using R, where the user can set the identity of the runner, pitcher, catcher, and batter from dropdown menus to mimic real life situations. The optimal lead distance is calculated in real-time with the predicted out risk, along with a user-given lead distance and its predicted out risk. There is also an accompanying graph showing the predicted pickoff and caught stealing curves by potential lead distances, as well as vertical lines showing the optimal lead and the given lead. A screenshot of the user interface can be found below.

A screenshot of a social media post

Description automatically generated

What does this mean in the context of game planning? Based on these findings, there appears to be a fairly sizable opportunity for baserunners to fine-tune their lead distances based on the situation to minimize their chances of making an out on an attempted steal. Slower runners should take advantage of opposing pitchers’ lack of attention and take every incremental inch they can get away with if they are planning a steal attempt. Faster runners, under the constant suspicion of a theft, can afford to be more conservative with their leads and rely on their natural ability to make up for it.

From the pitchers’ perspective, however, there is another takeaway from this information. Since the pickoff model is influenced by the data-driven bias that pitchers will tend to ignore the leads of slower runners and pay closer attention to those of faster runners, there may be an opportunity to adjust pickoff attempt frequencies. If slower runners’ lead distances are overinflated due to lack of attention, pitchers can capitalize by increasing the frequency of pickoff attempts.

**Conclusion:**

This algorithm will help deliver insights into what factors should be considered when designing the perfect lead distance for a runner on first base who will attempt a steal. The data used to build the model is not perfect, as discussed in the model evaluation, but it is designed to simulate what the runner (or coach) would know in real-time as he takes his lead. Data that would likely improve the accuracy of the model include the runner’s average first-step efficiency (jump) on steal attempts, the tagging ability of the middle infielders, and information about pickoff attempts.

Overall, runners have generally been clustering their lead distances around the same 18-inch range, when the impact of the different actors at play provide more flexibility for adjustment. The characteristics deemed important by the models tend to vary from player to player, providing grounds for more tailored leads. There may be an opportunity for slower runners to take advantage of pitchers’ pickoff attempt frequency by increasing their leads on steal attempts. Each incremental inch of lead distance increases predicted pickoff probability much more for faster runners than slower runners.