ISyE 6740 – Spring 2021
Final Report

Team Member Names: Ryan Cooper

Project Title: Analyzing and Improving an MLB Pitcher's Decision Making and Execution With Machine Learning

Problem Statement

A pitcher in Major League Baseball relies on a combination of strategy, deception, skill, and execution to be successful. Their sole job is to get an out for each batter they face.

To do so, they analyze the situation for each pitch and ask themselves questions like, "How many balls and strikes are there? Are any runners on base? What's the score?" They also consider decisions like "What pitch should I throw and where? How fast should I throw it?"

Based on the answers to these questions, the pitcher will carefully decide which approach he thinks has the best chance of sending each batter back to the dugout. The release point of the ball, spin rate, and breaking amount all contribute towards physically executing each pitch to the best of his ability.

What if a pitcher could make these decisions based on a seemingly endless library of historical data? While such a library exists in the form of Baseball Savant at MLB.com, a pitcher cannot realistically take this library with him to the pitcher's mound. Most pitchers are likely aware of their biggest strengths, but do they have any hidden strengths that aren't being used to their full potential? How does a pitcher's actual success and potential success stack up with others?

Data Source

Extensive data for each pitch of Major League Baseball seasons dating back to 2015 are available from Baseball Savant at MLB.com (https://baseballsavant.mlb.com).

Every pitch contains variables such as the specific batter and pitcher, inning, number of outs, the count (number of "balls" and "strikes" in the at-bat), runners on base, the score, the pitch count of the pitcher, the hand he throws with, the types of pitches he is known to throw (fastball, curveball, change-up, etc.), the "zone" of the pitch (if the strike zone and areas around it are divided into defined rectangular areas), the release point coordinates of the pitch, the speed of the pitch, and whether the batter is right handed or left handed. A comprehensive list of variables and their descriptions is located at https://baseballsavant.mlb.com/csv-docs.

Summarized statistics, like batting average, on-base percentage, number of hits (also split into doubles, triples, and home runs), win-loss record, ERA (earned run average), walks, and strikeouts are also available.

There are many methods for obtaining the data, however one of the more straightforward methods is by using the “baseballr” package for the R programming language (http://billpetti.github.io/baseballr). This package was developed by Bill Petti and conveniently extracts and pre-processes most of the Baseball Savant data.

Methodology

Background Research

The Baseball Savant website contains fantastic visualizations of the data by player (pitcher or batter) (https://baseballsavant.mlb.com/illustrator). Different variables can be adjusted to visualize their effect on other variables. While this is convenient for data exploration, comprehensive analysis using multiple players in multiple situations is not possible.

The authors of the MLB Technology blog (https://technology.mlblogs.com) take this data to another level by conducting analyses that are more along the lines of machine learning with articles like “Using Clustering Algorithms to Identify Distinct Pitcher Release Points” and “MLB Pitch Classification”. Additionally, the FanGraphs Baseball website (https://www.fangraphs.com) contains all sorts of baseball fan-contributed analyses using Baseball Savant data.

Predicting a pitcher’s next pitch and its location from the perspective of a batter appears to be a well-researched area, but approaching this prediction from the mind of a pitcher is not as widely available.

Data Access

The data used for this analysis comes from the 2018 Major League Baseball season. It was obtained using R programming by accessing the “baseballr” package. Each pitch is a data point with all variables previously described. Since each query only allows a certain number of pitches to be pulled, a loop allows for pulling short ranges of dates and then concatenating the data.

Overall Approach

Each pitch can be defined as a “success” or a “non-success”. A description of the outcome is provided for each pitch as part of the raw data, however it is not initially classified in this way. After some data manipulation, descriptions such as “ball”, “blocked ball”, “hit by pitch”, “hit into play no out”, “hit into play score”, and “pitchout” are classified as a “non-success” since they all have negative outcomes for the pitcher. Descriptions like “called strike”, “foul”, “hit into play”, “missed bunt”, and “swinging strike” are positive outcomes for the pitcher and are assigned as a “success”.

The machine learning model will attempt to predict successful pitches for each pitcher based on 3 groupings of variables:

1) **Execution/decision variables** – related to a pitcher’s raw talent and decision making (ie: what pitch to throw, where to throw the pitch in or out of the strike zone, how fast, where the pitch is released form the pitchers hand and how much spin is on it)

    - pitch_type, release_speed, release_pos_x, release_pos_z, pfx_x, pfx_z, plate_x, plate_z, vx0, vy0, vz0, ax, ay, az, effective_speed, release_spin_rate, release_extension, release_pos_y

2) **Situational variables** – related to the situation a pitcher is in, not controllable by the pitcher on the current pitch (ie: whether the batter is left/right handed, the count of balls and strikes, runners on base, number of outs, the inning, the top/bottom of the strike zone based on the height of the batter and his stance, the number of batters he has faced and the number of pitches he has thrown so far, whether or not his team is winning and by how much, how the infield/outfield are positioned and if they are in a shifted alignment)
    - stand, count, on_base, outs_when_up, inning, sz_top, sz_bot, at_bat_number, pitch_number, score_difference, if_fielding_alignment, of_fielding_alignment

3) **Combined variables** – all of the above, combined

Data Preparation

Three of the situational variables are not initially in the form needed for the model and have to be created from the raw data.

The "count" variable is derived from "balls" hyphenated with "strikes" (ie: 2-1 for 2 balls, 1 strike). Otherwise, the interaction of balls and strikes would not be considered.

The "score_difference" variable is initially separated into a "fld_score" and "bat_score" as part of the raw data. Subtracting the difference gives a positive (pitcher's team is winning) or negative (pitcher's team is losing) number and also indicates by how much the pitcher's team is winning or losing.

The "on_base" variable was initially three separate variables ("on_1b", "on_2b", "on_3b", and included the player ID of who was on each base). The new variable removes the player ID indicator, replaces it with whether or not the base is occupied, and concatenates the three (ie: 0-1-0, means there is a runner on 2nd base, and 1-1-0 means there are runners on 1st and 2nd base). Similar to the "count" variable, the combination of runners on base would otherwise not be considered.

Several column types must also be converted to numeric/continuous.

Model Selection

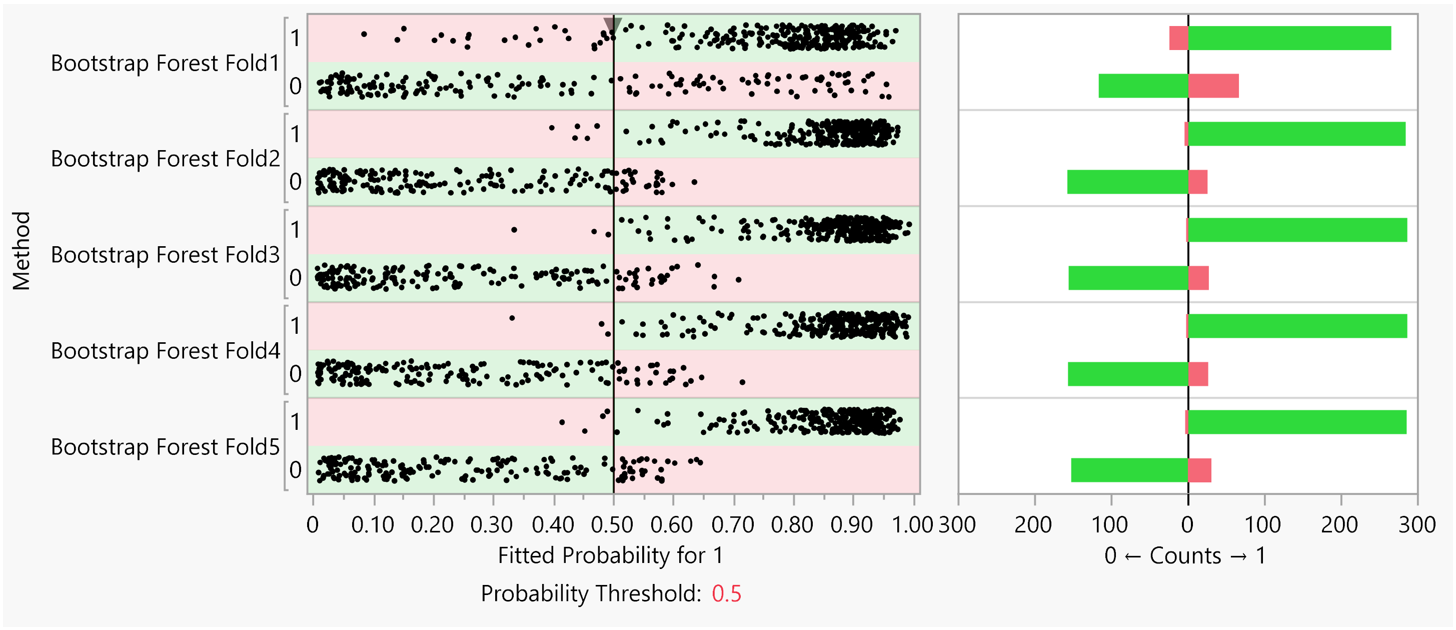
Misclassification rate is used to determine which type of model works best for each pitcher. The best model is where misclassification is the lowest.

JMP 16 Pro's "Model Screening" feature allows for training, validation, and basic tuning of multiple machine learning models at the same time. "Model Screening" was run on several different pitchers with k=5 for cross-validation.

An example of some of the “Model Screening” output using combined variables (execution/decision and situational) for Clayton Kershaw of the Los Angeles Dodgers is shown below:

Summary Across the Folds

	Validation Set Folds						
Method	**N Trials Folds**	**Sum Freq**	**RSquare**	**Mean RASE**	**StdDev RASE**	**Mean AUC**	**Mean MR**
Bootstrap Forest	5	472.80	0.3315	0.37311	0.00907	0.8468	0.18740
Decision Tree	5	472.80	0.2846	0.38868	0.00894	0.8079	0.19667
Boosted Tree	5	472.80	0.2491	0.40059	0.00698	0.8154	0.20855
Support Vector Machines	5	471.00	0.2209	0.40258	0.01598	0.8184	0.22593
Neural Boosted	5	471.00	0.2155	0.41340	0.00847	0.8050	0.23613
K Nearest Neighbors	5	472.80	0.0762	.	.	.	0.30629
Fit Stepwise	5	471.00	0.0738	0.45935	0.01316	0.6721	0.29517
Nominal Logistic	5	471.00	0.0723	0.45269	0.01340	0.7007	0.29977
Generalized Regression Lasso	5	471.00	0.0632	0.46312	0.00921	0.6607	0.29600



		Predicted		Rates	
Method	**Actual**	**1**	**0**	**1**	**0**
Bootstrap Forest Fold1	1	265	24	0.917	0.083
	0	66	117	0.361	0.639
Bootstrap Forest Fold2	1	284	5	0.983	0.017
	0	25	158	0.137	0.863
Bootstrap Forest Fold3	1	286	3	0.990	0.010
	0	27	156	0.148	0.852
Bootstrap Forest Fold4	1	286	3	0.990	0.010
	0	26	157	0.142	0.858
Bootstrap Forest Fold5	1	285	4	0.986	0.014
	0	30	153	0.164	0.836

Elapsed Time

Method	Details	Milliseconds	Elapsed Time
Decision Tree		639	0:00:00.639
K Nearest Neighbors		1190	0:00:01.190
Nominal Logistic		1279	0:00:01.279
Boosted Tree		3033	0:00:03.033
Generalized Regression Lasso		7405	0:00:07.405
Support Vector Machines		13515	0:00:13.515
Fit Stepwise		23359	0:00:23.359
Bootstrap Forest		26739	0:00:26.739
Neural Boosted		51819	0:00:51.819

Some observations:

- The "Mean MR" (misclassification rate) for the Bootstrap (Random) Forest is the lowest. In running "Model Screening" for several other pitchers, it became clear that a Random Forest model is generally best for modeling a successful pitch, regardless of pitcher.

- Predicting successes is generally easier to predict than non-successes, however errantly predicting a success when it is actually a non-success is far more common than predicting a non-success that turns out to be a success.

- While the Random Forest model is best with respect to misclassification rate, it is much more computationally intensive. A Decision Tree model may be a better tradeoff when it comes to accuracy versus speed.

For simplicity, considering there were 799 different pitchers in the 2018 MLB season, three different Random Forest models with 30 trees each will be used for each pitcher who threw at least 100 pitches (reduces the number of pitchers to 688). Since each player will have an execution/decision, situational, and combined variables model, this results in a total of 2064 different Random Forest classifiers. The data will be split into a training set (80%) and validation set (20%) for evaluation

An initial test of this approach using the "sklearn" package in Python with Clayton Kershaw results in the following accuracy rates for the validation set:

Player	Combined Variables	Execution/ Decision	Situational
Clayton Kershaw	0.8259	0.7962	0.5987

The misclassification rate for the JMP model using all variables combined was 0.1874. This one shows 0.1741 (1 minus accuracy of 0.8259), so the choice of using a Random Forest model with 30 trees seems reasonable for each pitcher.

Evaluation and Final Results

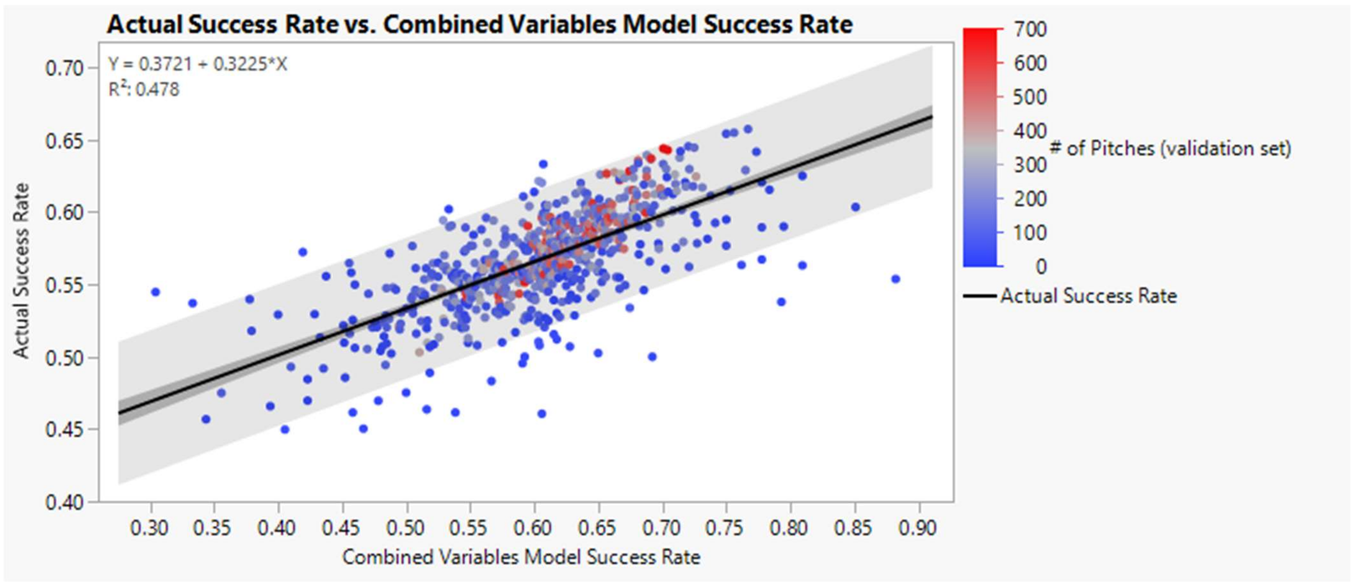
The resulting 3 models can be used to evaluate actual versus predicted success rates for each pitcher and determine which pitchers might benefit from the models the most. Then, a deeper dive into the variables that result in the highest probability of a successful pitch can be performed.

Actual vs. Predicted Success Rate

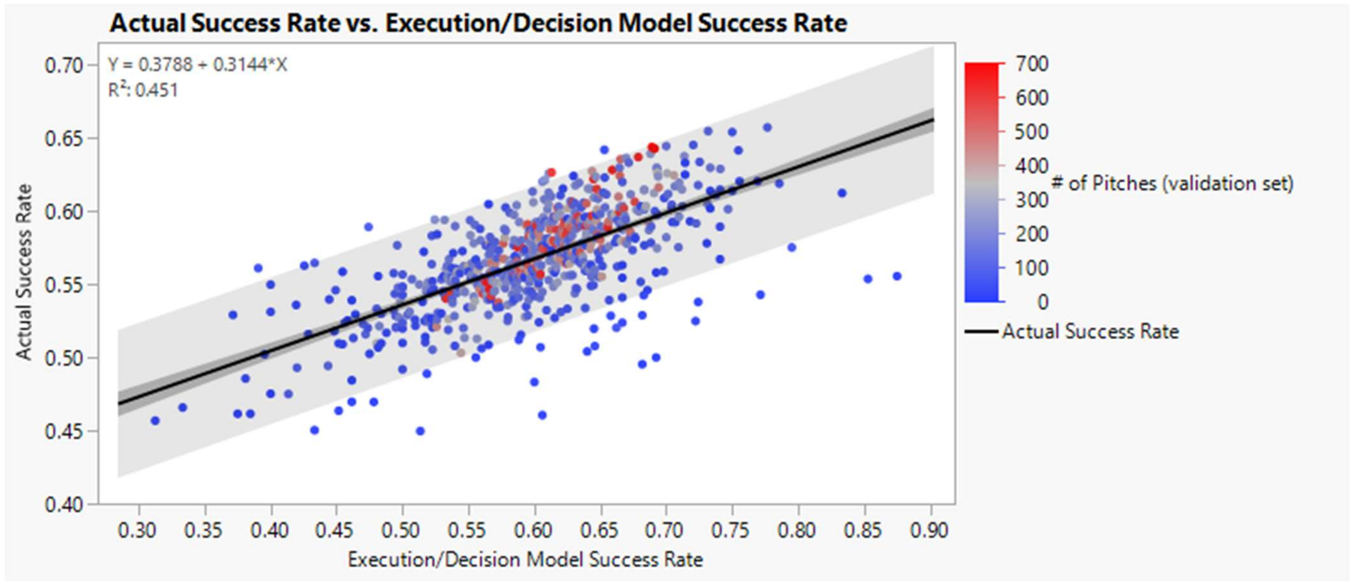
A plot of actual versus predicted success rate is provided for each model, comparing all pitchers. Success rates are the accuracy of only successes. Non-successes are omitted.

Each data point is colored by number of pitches in the validation set to demonstrate the impact that having less data has on success rates.

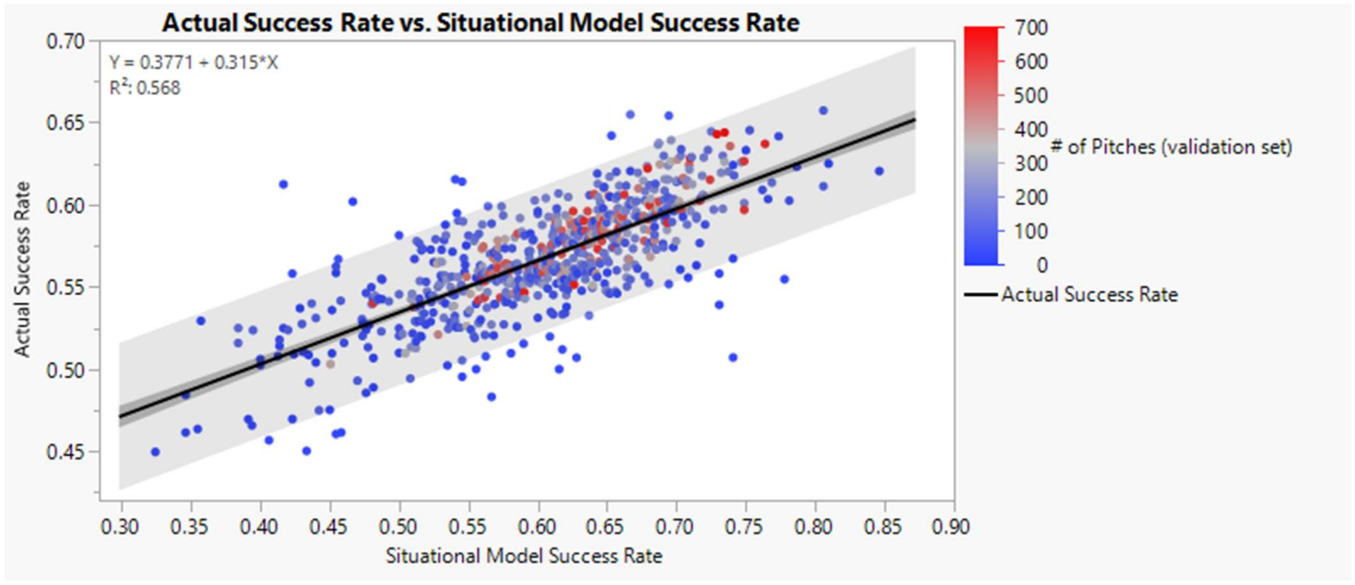
Combined Variables



Execution/Decision

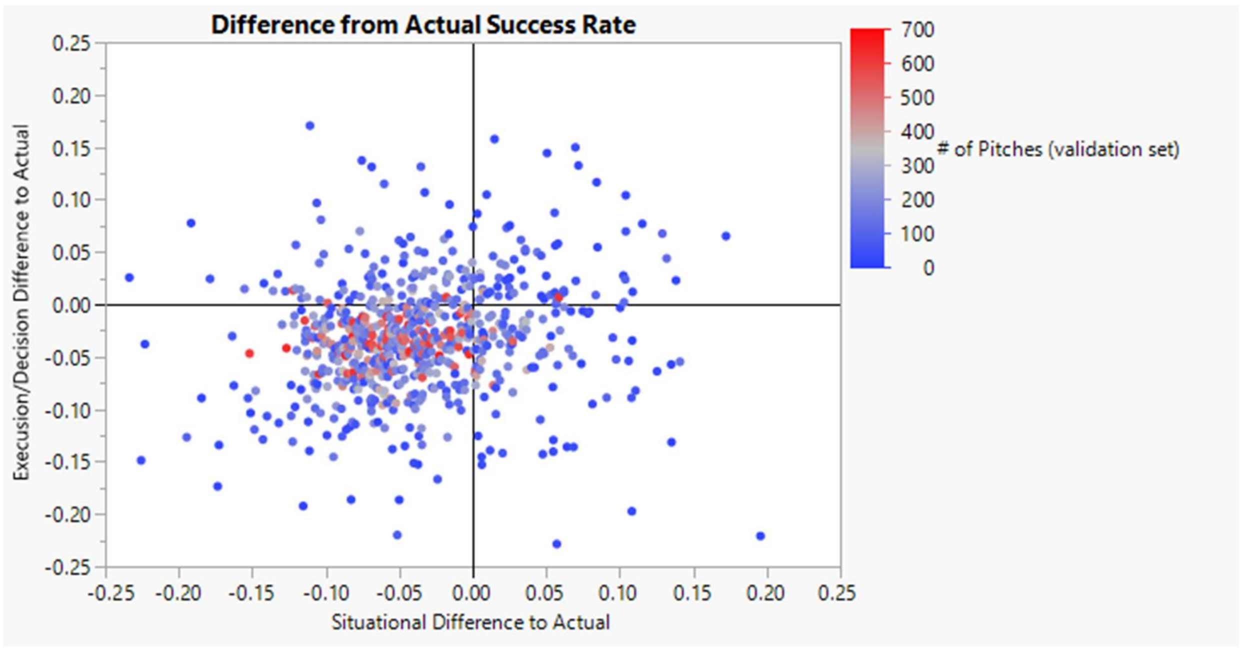


Situational



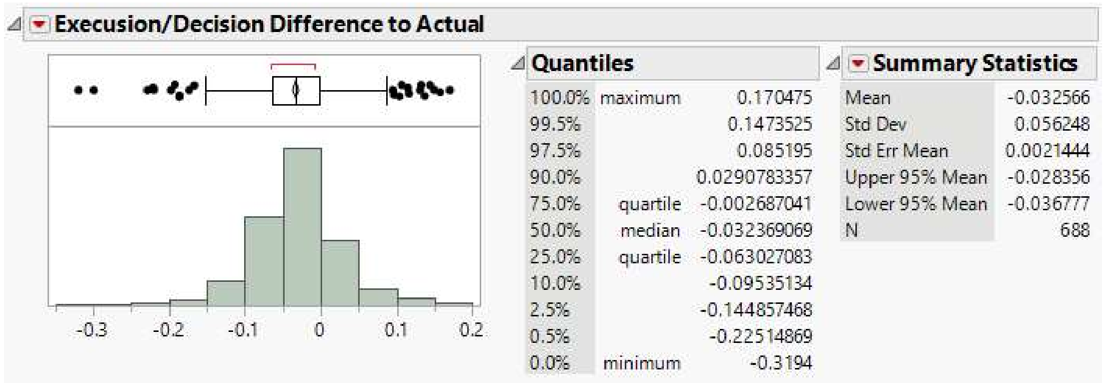
Generally, each model under-predicts lower success rates and over-predicts higher success rates. However, with more data for each pitcher, the models become much more accurate.

A plot of the difference between predicted and actual success rates for each of the two models using separate variables (execution/decision and situational) is shown below.



This plot reveals that the models over-predict success rates for most pitchers that have a substantial amount of data, but that difference is <u>much</u> closer than it is for pitchers without as much data.

The spread for pitchers with more data is also much less for the execution/decision model (up to just over 0.05 over-predicted) compared to that of the situational model (up to just over 0.10 over-predicted.



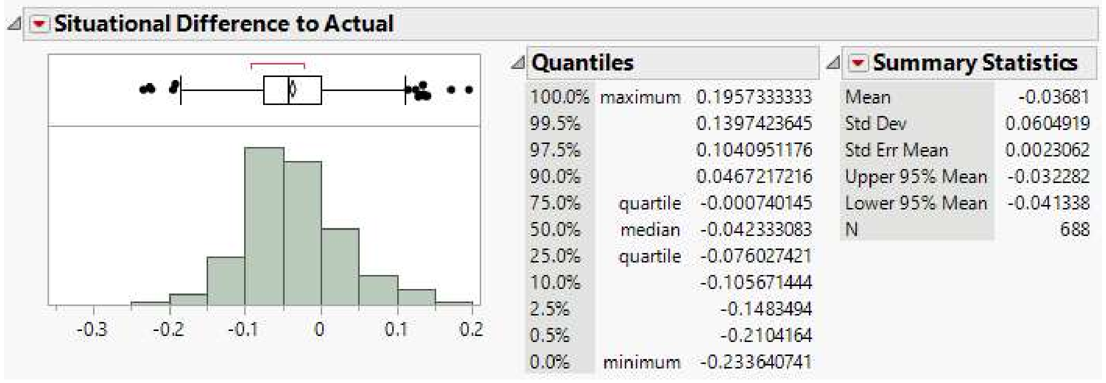
Execusion/Decision Difference to Actual

Quantiles

100.0%	maximum	0.170475
99.5%		0.1473525
97.5%		0.085195
90.0%		0.0290783357
75.0%	quartile	-0.002687041
50.0%	median	-0.032369069
25.0%	quartile	-0.063027083
10.0%		-0.09535134
2.5%		-0.144857468
0.5%		-0.22514869
0.0%	minimum	-0.3194

Summary Statistics

Mean	-0.032566
Std Dev	0.056248
Std Err Mean	0.0021444
Upper 95% Mean	-0.028356
Lower 95% Mean	-0.036777
N	688



Situational Difference to Actual

Quantiles

100.0%	maximum	0.1957333333
99.5%		0.1397423645
97.5%		0.1040951176
90.0%		0.0467217216
75.0%	quartile	-0.000740145
50.0%	median	-0.042333083
25.0%	quartile	-0.076027421
10.0%		-0.105671444
2.5%		-0.1483494
0.5%		-0.2104164
0.0%	minimum	-0.233640741

Summary Statistics

Mean	-0.03681
Std Dev	0.0604919
Std Err Mean	0.0023062
Upper 95% Mean	-0.032282
Lower 95% Mean	-0.041338
N	688

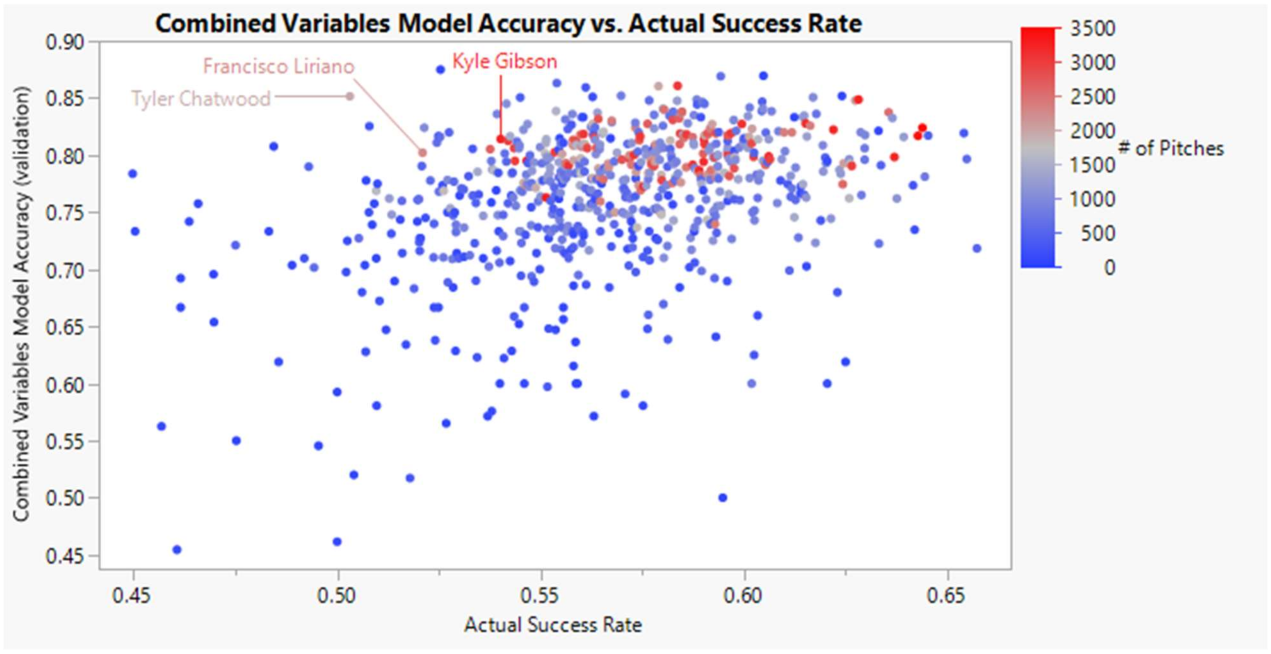
This suggests execution/decision variables have a more substantial impact on success rate, since the predicted success rates of that model are generally closer to the actual success rate, as long as there is enough data.

Model Accuracy vs. Actual Success Rate

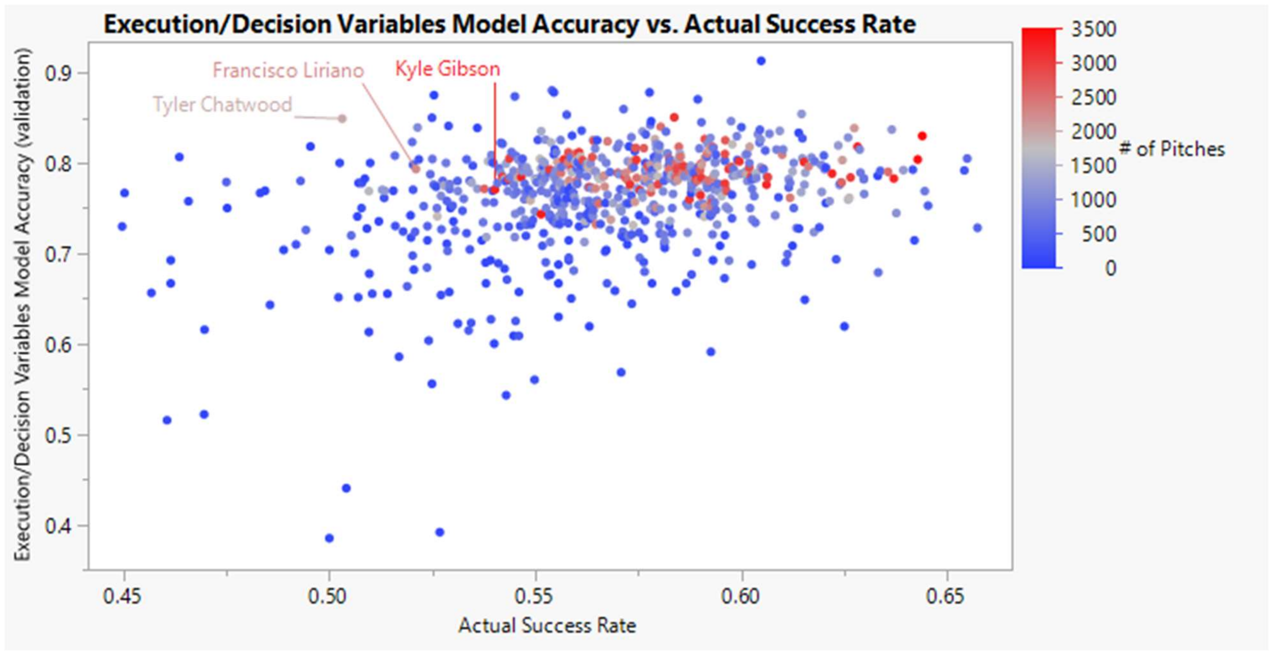
To determine which pitchers might benefit the most from the models, it can be assumed that a high model accuracy greater than the actual success rate means there is potential for improvement. As long as there is enough data, a pitcher could choose and execute a pitch with a high probability of success based on information from the model.

It is important to note that the model accuracy is the rate of correctly classified successes and non-successes, whereas the actual success rate is just based off of successes.

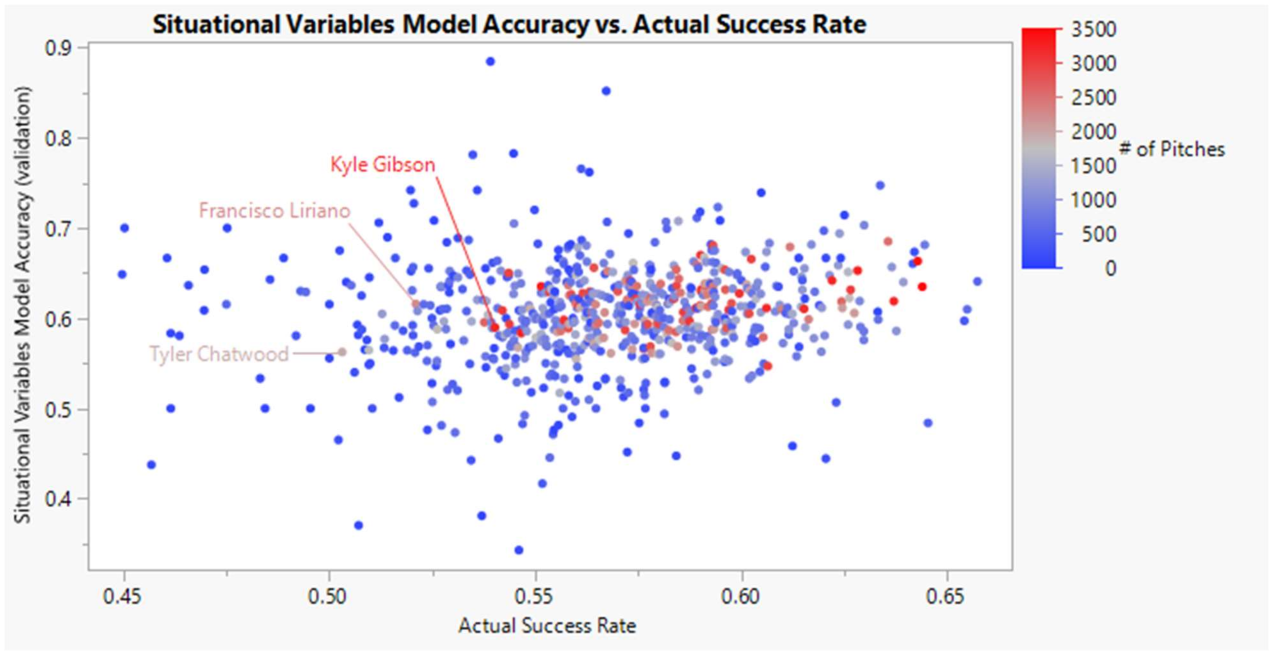
Combined Variables



Execution/Decision



Situational



From earlier analysis, ensuring a pitcher has enough data for evaluation is important when using these models for prediction. Given this important finding, Tyler Chatwood, Francisco Liriano, and Kyle Gibson appear to be good candidates who could benefit the most from learning more about their execution and decision making tendencies under different situations.

Their actual success rate is low, but the predictably for their models is high. There is also a substantial amount of data available, limiting the possibility for skewed results.

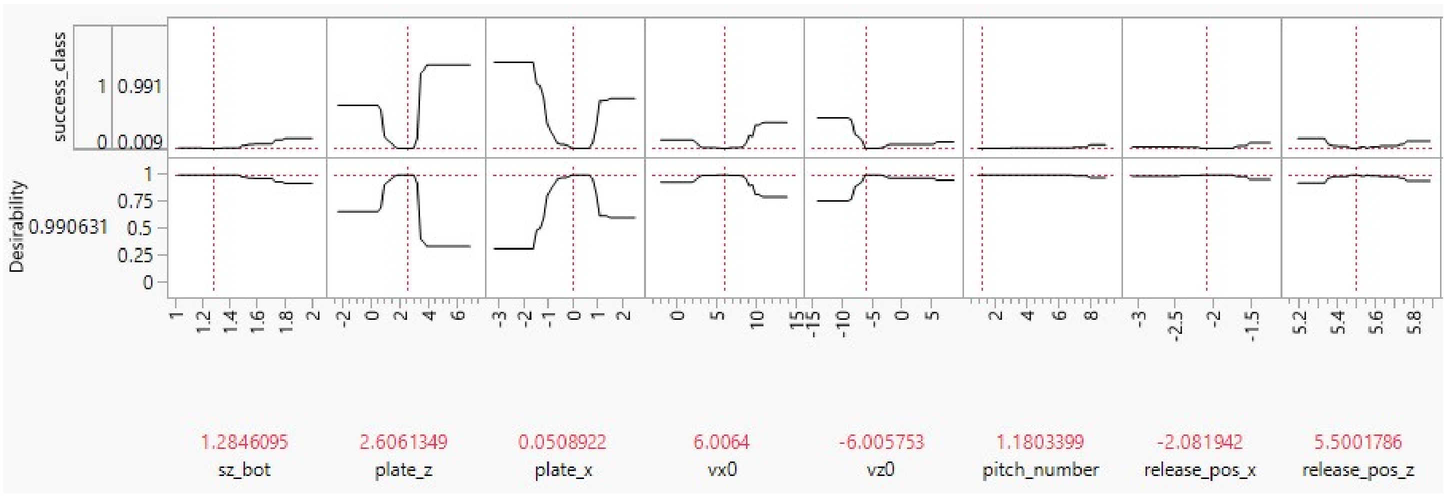
<u>Finding the Highest Probability of a Successful Pitch</u>

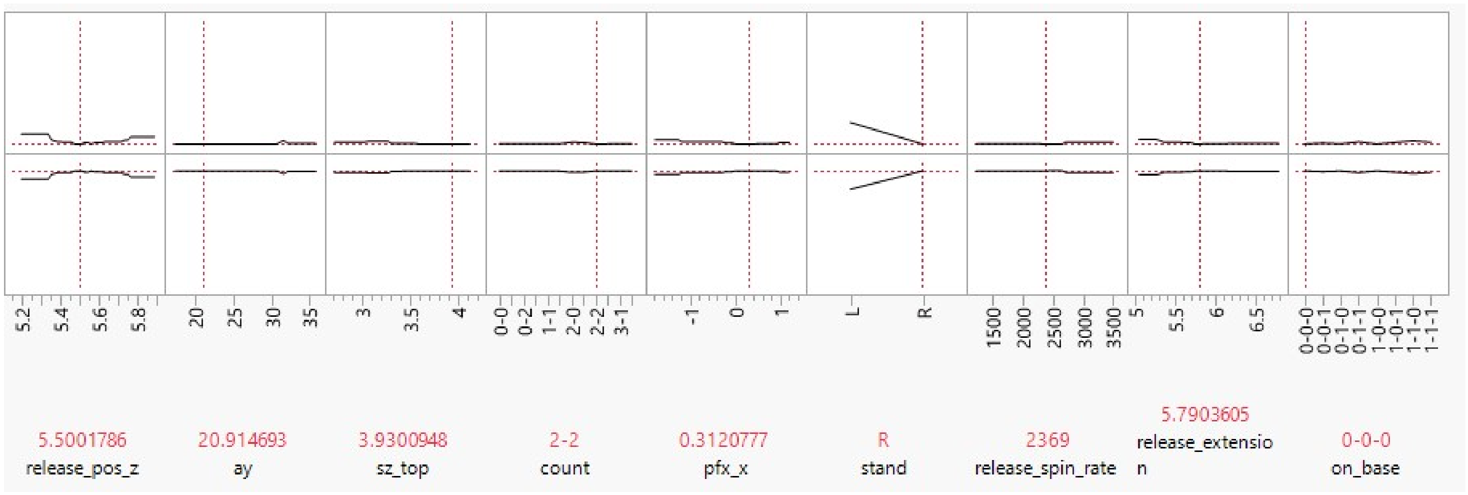
Tyler Chatwood has the highest combined variables model accuracy and the lowest actual success rate of the three pitchers previously mentioned. Taking a deeper dive into the variables that result in the highest probability of a successful pitch can reveal more about his tendencies.

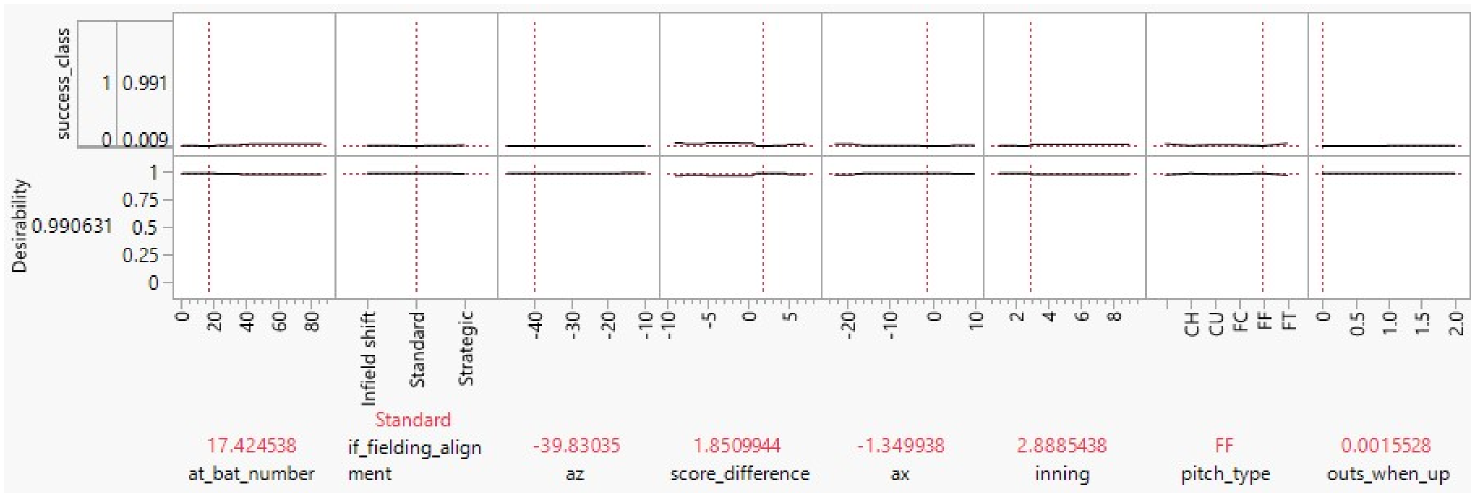
To do this, JMP's profiler tool can be used. It allows for determining the "maximum desirability" of a model. In this case, the desirability is having a probability of success as close to 1 as possible.

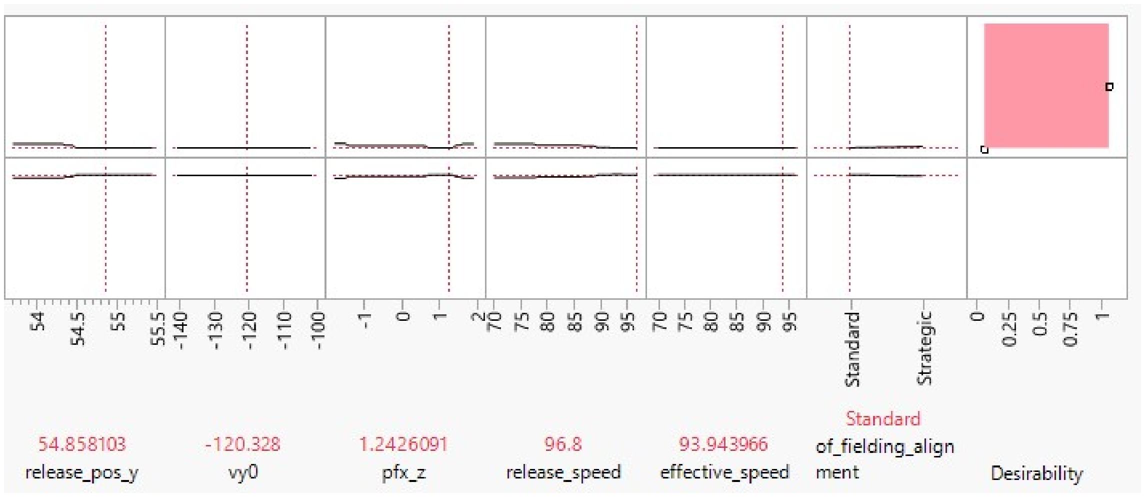
Each variable is assessed by importance in prediction and ordered by that importance.

The output of the maximum desirability tool in the profiler for Tyler Chatwood's combined variables model is shown below:









The basic idea of the output is that each variable has a desirability curve where the ideal value is "1". Each variable is optimized to maximize that desirability. Some variables do not matter as much and are more of a straight line, but others have different ranges where the predicted success is heavily affected by a difference in the variable.

In Tyler Chatwood's maximum desirability model, 'plate_z', and 'plate_x' are very influential because they are all related to the ball's location as it crosses the plate, and 'sz_bot' is the bottom position of the strike zone (which it appears Chatwood relies on for success). The parabolic nature of the success curve shows that a certain range results in more successes (the strike zone).

'vx0' and 'vz0' are related to the break on the ball in the 'x' and 'z' directions as it approaches home plate, so this also clearly has an influence. If there is little break, the pitch is less successful. Higher break is also not good and may indicate a wild pitch. There is clearly a "sweet spot" for break.

It is important to note that this maximum desirability model was run without holding any of the categorical variables constant, or without restricting any of the situational variables, resulting in a very high probability of success (0.990631). These variables can be altered interactively to explore the model in more depth. For example, changing the 'count' variable to 2-0 (2 balls, 0 strikes), allowing the bases to be loaded ('on_base' variable to 1-1-1), and switching the handedness of the batter from 'R' (right) to 'L' (left), results in a 0.762 success probability holding all other conditions constant. To maximize desirability of success under these conditions, Tyler Chatwood might need to change his approach on other variables.

Altering variables based on the situation and execution/decision of the pitcher results in different success probabilities and can help the pitcher understand what changes are more or less likely to be successful. They can discover hidden strengths that they were not aware of.

<u>Limitations and Possible Extensions</u>

As there are with any model, limitations exist. Several of these limitations are potential extensions that could be explored further.

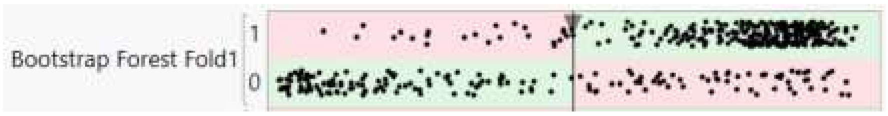
Some of them include the following:

- Since the model is built by pitch, it is very heavily influenced by the strike zone (as seen in the JMP profiler while finding the highest probability of a successful pitch). It is fairly easy to classify balls out of the strike zone as non-successes based on the positioning, and these positioning factors are the most predictive. Instead, it might be interesting to break the model down by balls that are hit into play.

- Each individual batter's strengths are not consider as factors. In fact, the batter is not considered at all in the model. This would require extensive modification to each pitcher's model. Something along these lines might be considered.

    1) **Predicted accuracy** relative to the <u>pitcher's overall strengths</u>
    2) **Predicted accuracy** relative to the <u>individual batter's weaknesses</u>
    3) Pitcher's **success rate** relative to whether the pitch was geared more towards the <u>pitcher's overall strengths</u> or the <u>individual batter's weaknesses</u>

- Rounding the probabilities of a successful pitch into a classification model can create a false sense of confidence that the result of the pitch is strictly a "success" or a "non-success". For example, the 1st fold of the Clayton Kershaw model shows that pitches with a probability of just over 0.50 are classified as successes, and just under 0.50 for non-successes. Transforming the model into more of a logistic regression might be something for further exploration.



- For some pitchers, there are not enough pitches to conduct a thorough analysis. A threshold of at least 100 pitches was selected for this analysis to try and avoid this issue, but investigating sample size might be good to consider in the future.

- Some continuous variables found to obtain the maximum desirability may not be easy to attain, or perhaps even unrealistic. For example, a high spin rate may contribute to more success, but obtaining the highest spin rate is incredibly difficult and potentially unreasonable for a pitcher to repeat consistently. The JMP profiler (particularly JMP Pro 16) has a new feature that allows for restricting the range of a variable. Modeling the variables as more of a distribution, as opposed to a straight range of numbers, may also be worth looking into.