Max Scherzer Pitch Recommender

Aja Klevs

October, 2019

Github Link

Overview

The goal of this project was to create a model that would recommend which pitch to throw given various characteristics of the game: what inning it was, who was at bat, the count on the batter, ect. To complete this project I used Kaggle pitch data from 2015-2018, and to limit the scope I decided to train on data from one pitcher in particular. I chose Max Scherzer because I have always been a fan of his, and he had ample data from the past three years.

My approach was to develop a basic binary model which would predict whether or not a pitch resulted in a "success" for the batter. I would then run the model on different pitches and return which pitch had the lowest probability of batter success.

Model Decisions

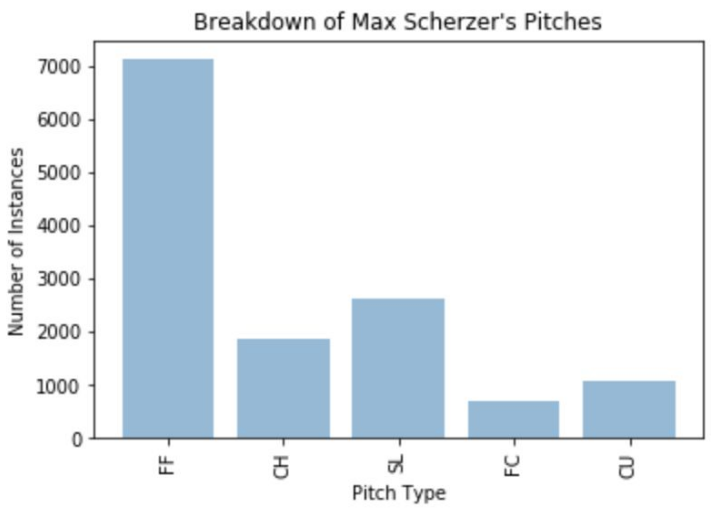
There were a few decisions I had to make before diving into this project. First, was what constituted a batter's 'success'. I considered several ways of defining this variable, including considering the pitch a success for the batter if it was a ball or resulted in the batter getting on base. I also tried only considering base hits or home runs as a success. In the end, I settled on setting the target variable to one when the hitter made it on base without an out. This system seemed to have the least ambiguities and resulted in about 6% of the data points in my training set to be considered successes for the batter. I called this target feature "code". I got rid of any instances of intentional walks, as to not consider these events in my model.

I also had to decide how to represent the batter in my data set. To do this I gave each batter a score for each pitch, based on his historical record against that type of pitch. I wrote a function look_up_hitter whose inputs are the hitter's id in the dataset, the type of pitch, and the zone the pitch was in. The function then finds the fraction of times that pitch has resulted in that batter getting on base without an out. If the batter has seen the pitch less than a hundred times, the function returns Max Scherzer's score for when he has used the pitch. For pitch types with low representation in the dataset, I set the batter score to -5, which was slightly lower than all known scores.

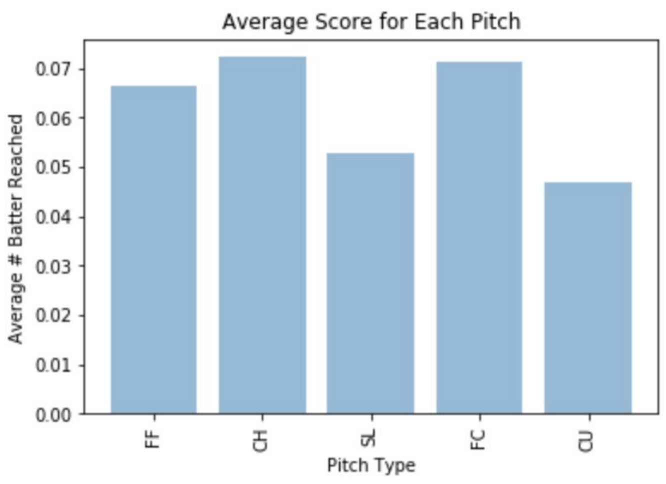
Lastly I had to figure out how to represent the different types of pitches in my dataset. I limited it to Scherzer's five most common pitch types, which were Four-Seam Fastball (FF), Slider (SL), Changeup (CH), Curveball (CU) and Cutter (FC). I then recorded the different height and width of the pitch as additional features, based on whether the batter was left handed or right handed. Ultimately, this left me with ten features, two for each pitch type. These features were zero when the instance was not of the same pitch type, and contained the height and width information of the pitch when they were of the same pitch type.

<u>Baselines</u>

After preprocessing most of the data, I wanted to see some trends so that I could later evaluate the performance of my model. First, I tried to see how often Scherzer throws each pitch in the dataset:

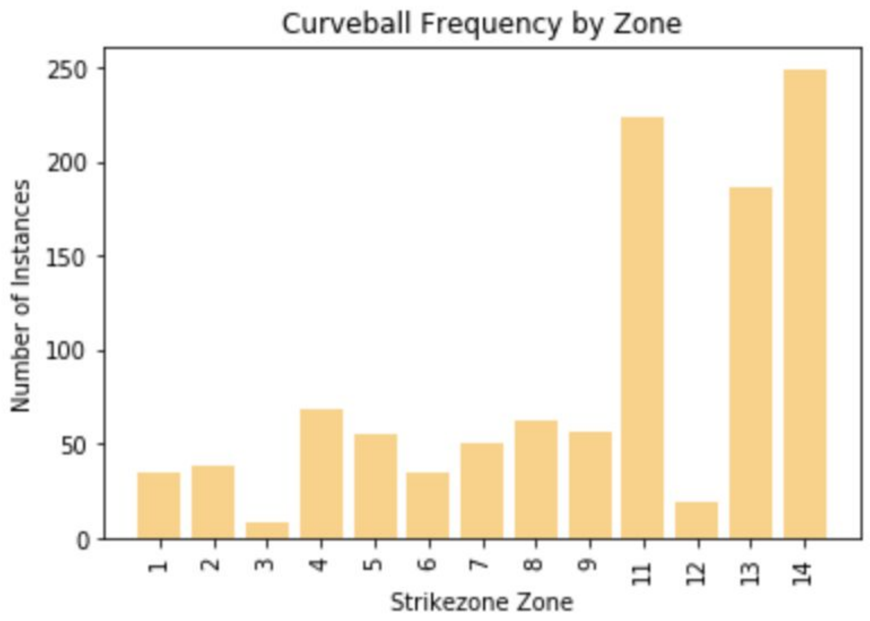


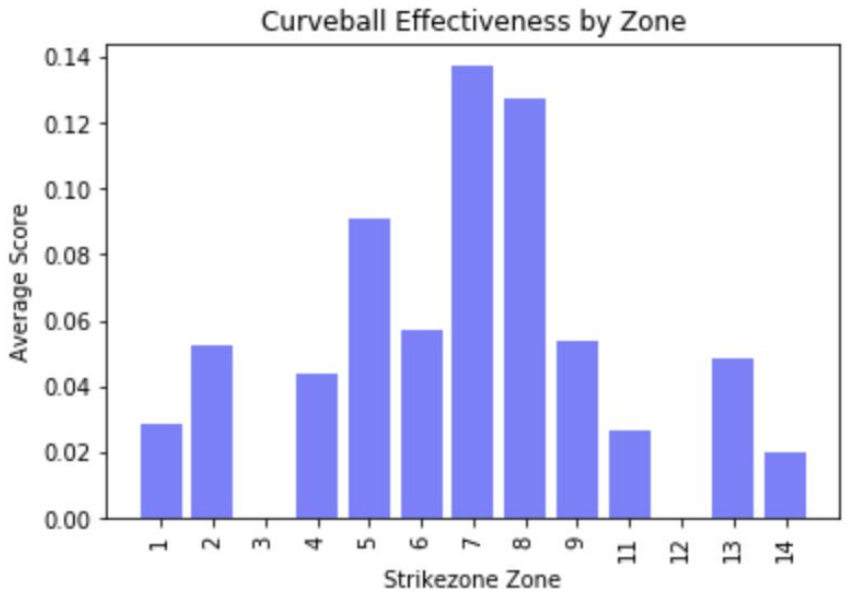
Unsurprisingly, over half of Scherzer's pitches are four-seam fastballs. However, I wanted to know which of his pitches results in the most batters reaching base. Therefore I found the number of times a batter reached on each pitch and divided by the number of instances of that pitch in the dataset.



By this metric, Scherzer's curveball is his best pitch, with runners reaching on it less than 5% of the time. There are likely more nuanced reasons for this; i.e. Scherzer's curveball could be often taken for a ball, although not always ball 4. In any case, I should expect my model to suggest curveballs more often than Scherzer throws them.

Lastly, when looking at curveballs I wanted to see which zones are the most effective. Similarly to above, I calculated the average score for curveballs in each zone. I found both the frequency of curveballs and the effectiveness (average score) by zone.

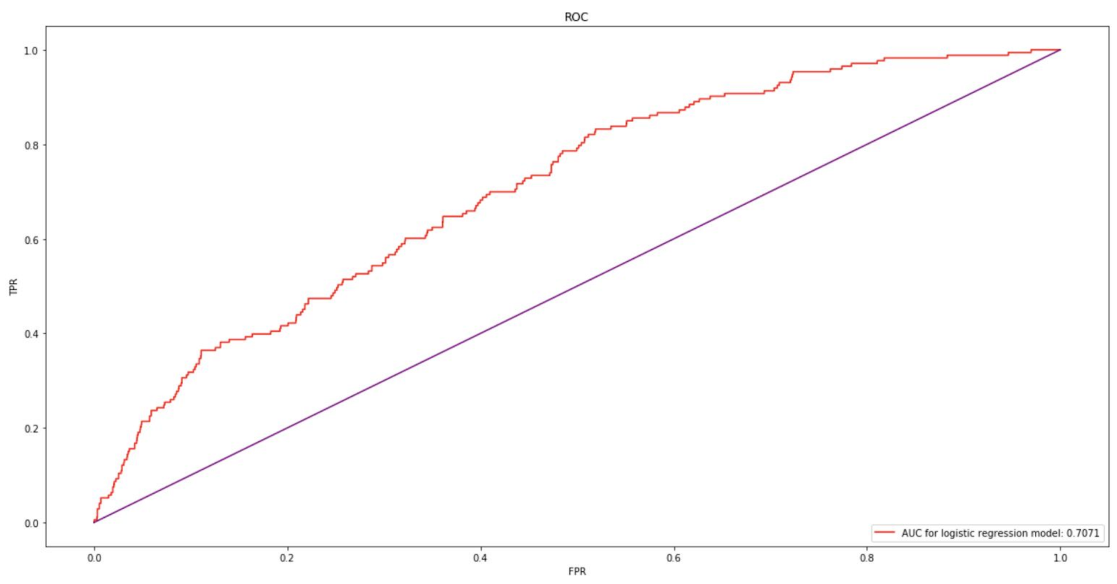




These results are interesting, indicating that the vast majority of Scherzer's curveballs are off the plate, and pitches off the plate are far more effective than those in the strike-zone. Also, Scherzer's curveballs tend to be low, although his most effective curveballs are in the upper right hand part of the strike-zone. Therefore I expect my model to predict many out of the strike-zone curveballs.

Results

I used the processed data to perform a logistic regression with a regularization parameter of 10^{-5}. I used 80% of my data to train, and 20% to test. In order to initially gauge the accuracy my pitch recommendations, I decided to evaluate the performance of my model. Due to imbalanced classes, I graphed the Precision Recall ROC Curve and found the AUC of the model, which was 0.707.



From the ROC curve, it was easy to see that there were no thresholds where the precision recall tradeoff would be particularly unbalanced. The model appears to be a relatively predictive of whether the hitter would reach base without an out.

For the first twenty instances in the test randomly shuffled test set, I replaced the pitch type and location data for every type of pitch in every zone (65 total combinations) and returned the pitch with the lowest probability of resulting in the hitter reaching base.

For most of the instances, my model would suggest a curve-ball in the upper right corner, regardless of whether the batter was a righty or a left. However, it sometimes suggested a Four Seam Fastball on the upper left corner of the plate.

Conclusions

Ultimately, my recommender predicted Scherzer's most effective pitch by the standards of the problem for nearly every batter in every situation. There are several reasons this could be happening.

First, there was likely insufficient training data. Some of the pitch-type location combinations were poorly represented in the data, and some of the batters hadn't faced enough pitching to give them accurate scores. In the future some combination of upsampling pitches that were poorly represented in the data, or getting data from other pitchers could have improved the recommenders performance.

Secondly, linear regression may have not been the best choice of model. In fact, our results indicate that a naive model would have given similar if not identical results. It is likely that the relationship of the features was not linear, or that variables such as start time, if anyone is on first base, ect are significantly less informative than features which were not perfectly captured by the model, ie who is batting and what pitches have already been thrown.

In the future, I would like to experiment with how to define the target variable (how to evaluate the 'success' of a pitch) and also how to account for who is at bat and which pitches have already been thrown.