Predicting Run Values by Location with Machine Learning

Brian T. Cook

January 15, 2019

A pitch's location is often one of the most important attributes in determining whether or not it is a success or failure according to the pitcher. The goal of this project is to create a map of the zone by expected wOBA using exit speed/angle with a k-nearest neighbors algorithm.

Inputs

In order to control for the pitch location only we choose to provide the platoon, count, and pitch type. For example, we could take all of the off-speed pitches (changeup, curveball, and so on) thrown by RH pitchers to RH batters in a 1-2 count.

Description of the algorithm

Nearest neighbors

Each pitch location is written as an ordered pair (x, z) and all of the relevant pitches being considered are then stored in a 2D array. With each pitch is the expected wOBA using exit speed/angle as a target for the nearest neighbors algorithm to emulate. The program then asks how many nearby pitches you would like to consider when computing the map value associated with a particular region in the zone. Fewer leads to a faster computation time but more leads to better accuracy.

Pitches are weighted in the calculation according to the square of their distance, i.e. a pitch that is three times further away from the map location being considered will be 1/9 as important. The `sklearn` library chooses which nearest neighbors algorithm is best for the given data, but this could in principle be adjusted.

Included are the map's mean and standard deviation, which indicates whether or not a region in the zone is exceptionally good/bad for the pitcher.

Possible Improvements

For now I am only considering pitches that were hit into play. Outcomes such as an intentional walk, strikeout, or hit by pitch are saved in the `.csv` files as `ewOBA = 'null'`. Once I have some

time to take a look I can select out results that are irrelevant to this analysis (intentional walk, fielding error) and then assign values to other types of walks and strikeouts (`ewOBA` = 0.7, 0 respectively).

I could also combine the years to increase the number of data points being considered, which would provide a more accurate picture regardless of the pitch attributes.

Sample Result

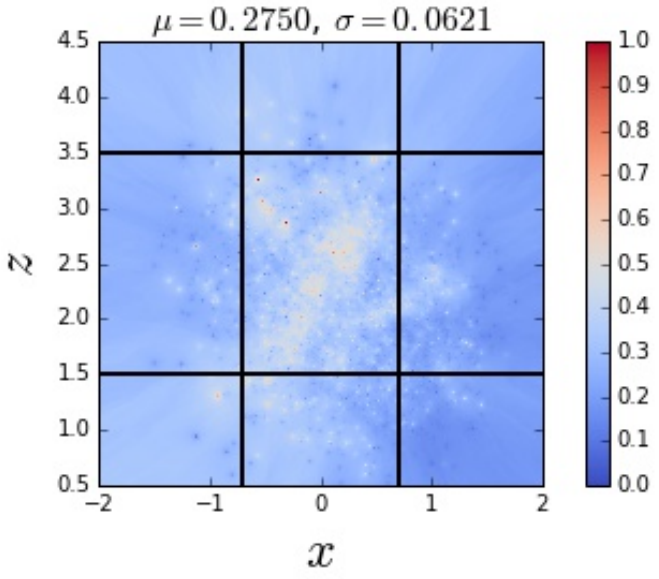

Figure 1: This is the computed result for an off-speed pitch thrown in a 1-2 count with a R v. R platoon. I only included pitches from the 2018 pitches ($n = 3151$).

Conclusions

Given the richness of data associated with the game of baseball and the possibilities machine learning techniques provide it seemed like a fun idea to put together a tool for pitchers to use. If I'm facing a RH hitter in a 2-1 count is it better if I throw a fastball or something off-speed, and where should I throw it?