Predicting good pitches

DS-GA 1003: Machine Learning

Github

Atakan Okan	Juan Zamudio	Kuixian Zhu
ao1512@nyu.edu	jjz289@nyu.edu	kz1041@nyu.edu

May 18, 2018

1 Problem Definition and Data

The goal of this project is to predict the probability of a baseball pitch being "good" based on Pitch F/X data. We classify a pitch as good if it induces a strike, a foul, or if it generates 'weak' contact. We define weak contact as: flyouts, groundouts, popouts, forceouts, double plays, fielders choice out and bunt out.

The data we use is the Pitch F/X data from 2012 to 2017. These datasets contain information of each pitch thrown (including, but not limited to, velocity, movement and location) as well as the outcome of the at bat. The total number of pitches amounts to 3.2 million pitches for the training set and approximately 500 thousand pitches for the test set which were divided temporally.

Using Random Forests, XGBoost and neural networks algorithms, we obtain a prediction for each pitch given velocity (mph), location (normalized where 0 represents the center of the plate), horizontal and vertical movement, spin (rpm), number of pitches thrown by the pitcher in that game, batters' or pitchers' count, number of balls and strikes, pitch type, and the the difference of a long term rolling mean and short term rolling mean for each pitcher.

To train the model we use the data for the 2012 to 2016 seasons as a training set and use the 2017 data as a test set. Hyperparameters are obtained through time series cross-validation.

Our baseline is a Random Forest Classifier using as only input the vertical and horizontal locations of the pitches. Since it is relevant to capture differences in predicted probabilities and not just predicted classes, our primary evaluation metric will be mean log-loss.

Our main findings are that location (vertical and horizontal) are extremely important in predicting if a pitch is good or not. Additionally, velocity and movement also provide some predictive value. From our analysis, it seems that for borderline pitches it is hard to predict if a pitch is good or not, given our dataset. We would need additional features such as handedness of the batters and pitchers and some statistic that measures the batters' quality.

2 Exploratory Data Analysis, Baseline & Feature engineering

Every feature of the data was analyzed to see whether it contained illogical values or NaNs and corrective actions were taken to keep the data as clean and good as possible. In general, the data needed little preprocessing. For numerical features, the features' histograms were plotted to see distribution of the values and to determine whether some kind of transformation is needed. The data also contains some features that do not have predictive value, which were dropped before model training.

In order to measure how "locked in" the pitcher is we obtained two rolling means of fastball velocity for each pitcher, one that included the last 250 fastballs and the other

one that included the last 10 fastballs. After this, we substracted the second one from the first one to obtain the "speed diff" feature that we used in our analysis. It turned out this feature did provide any significant predictive value whether the pitch is good or bad.

By doing some exploratory data analysis we can see that, as we would expect, velocity and location play a very important role in defining the success of a pitch. As we can see in Figure 1 (a), location of the pitch seems very relevant, pitches in the strike zone are much more likely to be good pitches (red) according to our definition, as we would expect. From Figure 2 (b) we can infer that velocity also plays a role, especially in faster pitches.

Due to the importance of pitch location, our baseline to improve upon is a Random Forest Classifier using only as inputs horizontal and vertical location of the pitch (px and pz). The hyperparameters were trees of at most 7 levels of depth and we used 50 estimators (trees). The results can be seen on Table 1, it achieved a mean log-loss of 0.52 and an accuracy of 76%, a significant improvement over the majority class. The predicted probabilities on the test set can also be seen in figure 3 (a), the algorithm seems to be classifying as a good pitch everything within certain squares.

3 Results

As there are numerous potential interactions between different variables, we chose to use models that could learn features (interactions between different variables) by themselves. In specific, we chose Random Forest, XGBoost (both are ensemble tree methods, and thus able to learn features) and Fully Connected Feedforward Neural Networks (which could learn features from multiple layers of transformations).

We use two different tree-based methods that work in different manners. In Random Forest, we would build numerous decision trees in parallel, and then bag them together to reduce variance. Also, in order to reduce the dependency between trees, we would restrict choice of splitting variable to a randomly chosen subset of features. On the other hand, XGBoost is a gradient boosted decision tree method that is implemented for speed and performance. As a gradient boosting method, in XGBoost, we would build the trees that are in the locally best step direction to enhance the performance. Thus, we build the trees step by step, rather than build all the trees in parallel.

The first iterations of our model to improve upon the baseline is a Random Forest Classifier and an XGBoost classifier, with the following features: spin, location, velocity, horizontal movement, vertical movement, pitch count, number of strikes, speed diff (as described before). Our results are summarized in Table 1.

	Baseline (RF)	**Random Forest**	**XGBoost**	**Neural Net**
Log loss	0.522	0.488	0.478	0.471
Accuracy	76.0%	78.2%	78.3%	78.8 %
Majority class	58.4%	58.4%	58.4%	58.4%

Table 1: **Results**

We can see that that both classifiers improve significantly over the Logistic Regression baseline. Analyzing the feature importances from Random Forest and XGBoost (figure 2) we can see that location (horizontal and vertical) are the most important features. After that, XGBoost tends to rate higher other features such as speed and spin. However, both models have very similar results, the models are still learning something very similar to the strike zone, although they are a lot less confident about the edges of the strike zone (Figure 3 (b)).

For this reason, and since tree-based models only partition the space perpendicular to the axes, we implemented a feedforward neural network with three fully connected hidden layers to try to better capture nonlinear relationships. The results of this model can be seen in Table 1 and in Figure 3 (c). The neural network improved slightly over the XGBoost model, we also can see in Figure 3 (c) that its predictions are less confident around the strike zone and its boundaries are less 'linear' than the other models.

4 Error analysis

In this subsection we analyze the errors of the neural network, since it was our best performing model. As it can be seen in Figure 4 it is clear that the models are mispredicting the pitches in the edge of the strike zone. We analyzed the distribution of several features for correctly and incorrectly predicted pitches. As it can be seen in Figure 5, there seems to be at least a slight difference in velocity, spin and movement for these groups of pitches. To try to improve our models, we ran a Random Forest Classifier for each type of pitch, the results can be seen in Table 2.

This strategy did not seem to work, in most cases it worsened the performance, presumably because by dividing the dataset the models lost valuable information about other pitches.

	Log Loss	**Accuracy**	**Majority Class**
Baseline (RF)	0.522	0.760	0.584
Random Forest - CH	0.505	0.765	0.536
Random Forest - CU	0.482	0.781	0.525
Random Forest - FC	0.533	0.757	0.548
Random Forest - FF	0.447	0.809	0.575
Random Forest - FT	0.485	0.785	0.538
Random Forest - KC	0.550	0.733	0.513
Random Forest - SI	0.516	0.767	0.555
Random Forest - SL	0.513	0.758	0.565

Table 2: **Random Forest for Every Pitch Type**

From this analysis, we conclude that to improve significantly our model we need more features such as handedness of batters and pitchers, a talent measure of hitters and maybe even control for umpire.

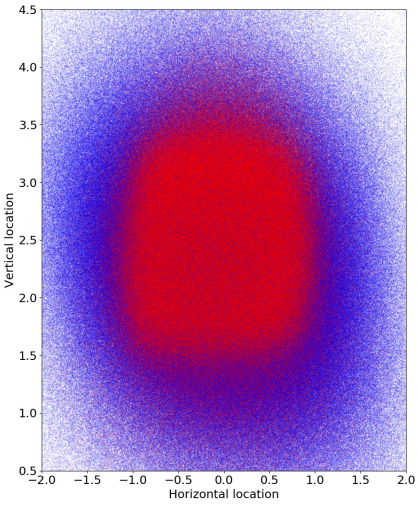
Finally, it is worth mentioning that we tried several other approaches that got significantly worse performance, such as: using a feature that measures the euclidean distance from the center of the strike zone, quadratic and interaction terms for the location features and running a Decision Tree classifier with inputs location, pitch type and velocity and using the regions obtained by this model as features to other models.

5 Conclusions and extensions

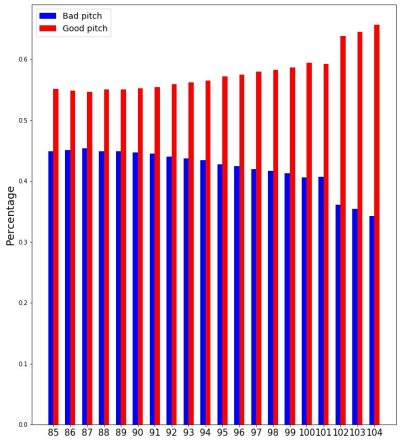
Our main finding is that for our binary definition of what constitutes a good pitch, location is extremely relevant, after that, there are other factors such as velocity and movement that also have predictive value. This can be seen by the significant improvement that we achieved in log-loss and accuracy by adding velocity, spin, among others to our baseline model. To further improve our model, it would be interesting to add more data about the players, such as handedness and a measure of talent.

Finally, the results obtained in this project could be used as a measure of pitchers performance in each game. It would also be very interesting to analyze the predictive power of our statistic by trying to predict pitcher statistics (ERA or FIP) with our measure.

6 Figures



(a) Locations (red: good, blue: bad)



(b) Percentage of good and bad pitches by velocity

Figure 1: Pitch locations and velocities

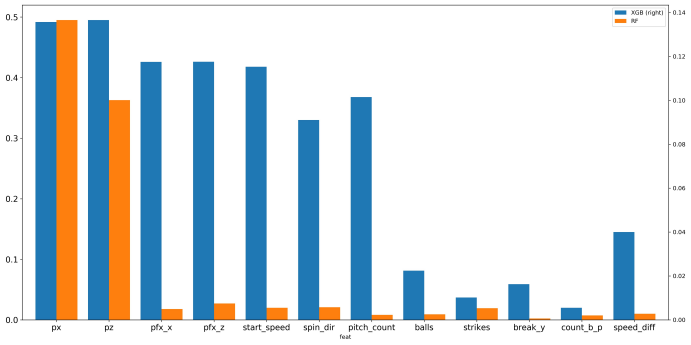
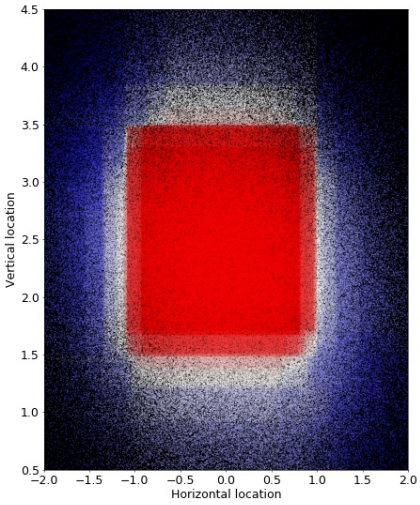
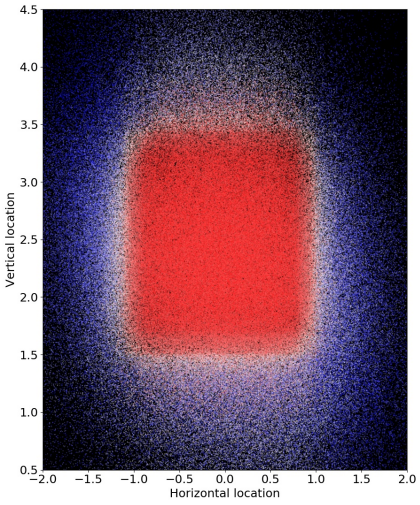


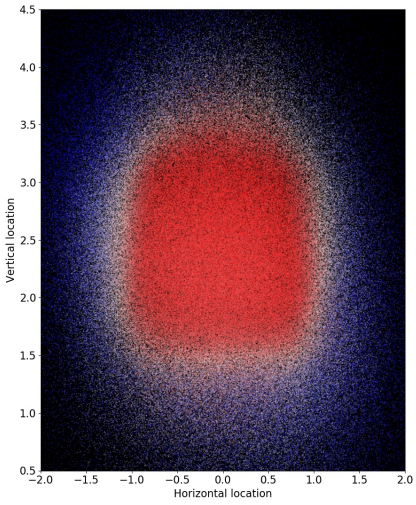
Figure 2: Feature importances for Random Forest and XGBoost (right axis)



(a) Random Forest Baseline

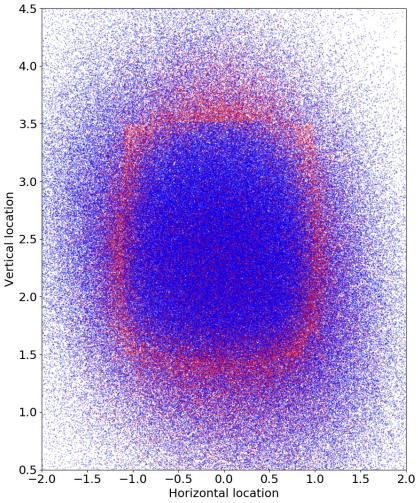


(b) Random Forest

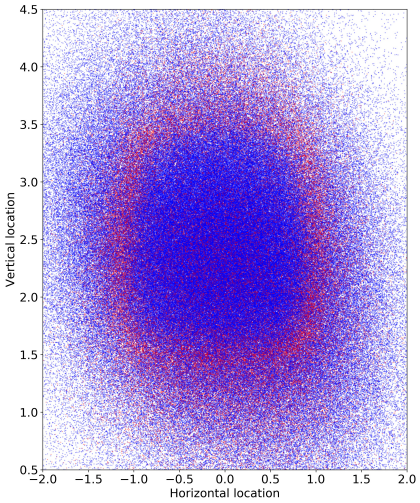


(c) Neural Network

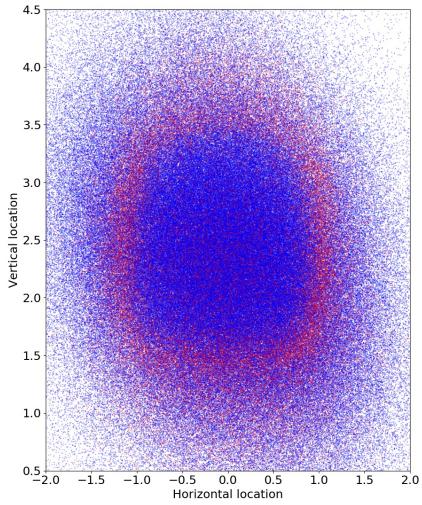
Figure 3: Predicted probabilities on test set (red: good, white: uncertain, blue: bad)



(a) Random Forest Baseline

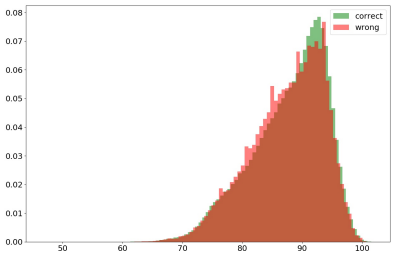


(b) Random Forest

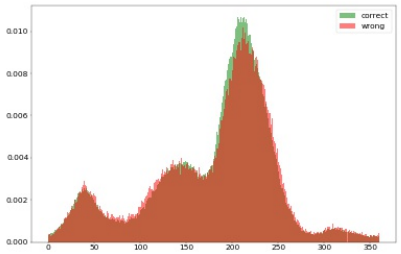


(c) Neural net

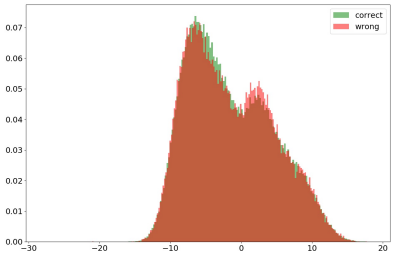
Figure 4: Errors in test set (red: error, blue: correctly classified)



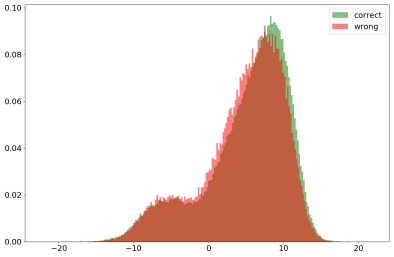
(a) Velocity



(b) Spin



(c) Horizontal movement



(d) Vertical movement

Figure 5: Distributions of features for errors and correctly classified pitches

References

[1] Eli Ben-Porat, *SANTA: A Binary Approach to Pitcher Evaluation*, `https://www.fangraphs.com/tht/tht-annual-2018/santa-a-binary-approach-to-pitcher-evaluation/`

[2] Rob Arthur and Greg Matthews, *Baseball's 'Hot Hand' Is Real*, `https://fivethirtyeight.com/features/baseballs-hot-hand-is-real/`.

[3] *What is PITCHf/x*, `https://www.fangraphs.com/library/misc/pitch-fx/`.