Releasing Speed Prediction Based on Pitch Types

Xiaoqing Yao
Data Science Institute, Brown University
Github

1. **Introduction:**

   Purpose:

   Data science is one of the hottest topics today, especially when combined with other fields, it has shown strong application potential and commercial value. As an important field with a large audience in the United States, the combination of sports and data science has become a major trend. Driven by this trend, I chose baseball as my project topic and explored key issues in baseball through data analysis. This is not only due to my love for sports, but also because baseball data analysis has high research value and practical significance. The goal of this project is to use machine learning methods to predict the release speed of baseball pitchers based on a series of features before pitching. This is a typical regression problem. Through the analysis and prediction of this study, we can deeply understand the specific impact of different variables on pitching performance. It can help pitchers adjust their pitching strategies and improve their actual performance in the game. Provide scientific basis for coaches and data analysts to develop more effective game tactics. Prediction models can also be used for player selection and training, identify potential players, and optimize player training plans.

   Dataset resource:

   The dataset for this project comes from Baseball Savant [1] and is collected and organized by the Statcast system [2]: Statcast is an automated tool developed to analyze player movements and athletic abilities in Major League Baseball (MLB). [3] This project focuses on Clayton Kershaw, a famous pitcher for the Los Angeles Dodgers of Major League Baseball (MLB). He was selected to the All-Star Game ten times and widely regarded as one of the greatest pitchers in baseball history. Kershaw's pitching technique is of great research value. The dataset covers all game records from 2017 to 2023 (excluding the 2020 season). Because of the pandemic, there are not so many games played during 2020 season. I removed the data points in that year.

2. **EDA:**

   Target variable:

   The target variable in this project is release speed. I used histogram to show the distribution of the target variable as figure 1. The histogram has two peaks and does not distributed normally.

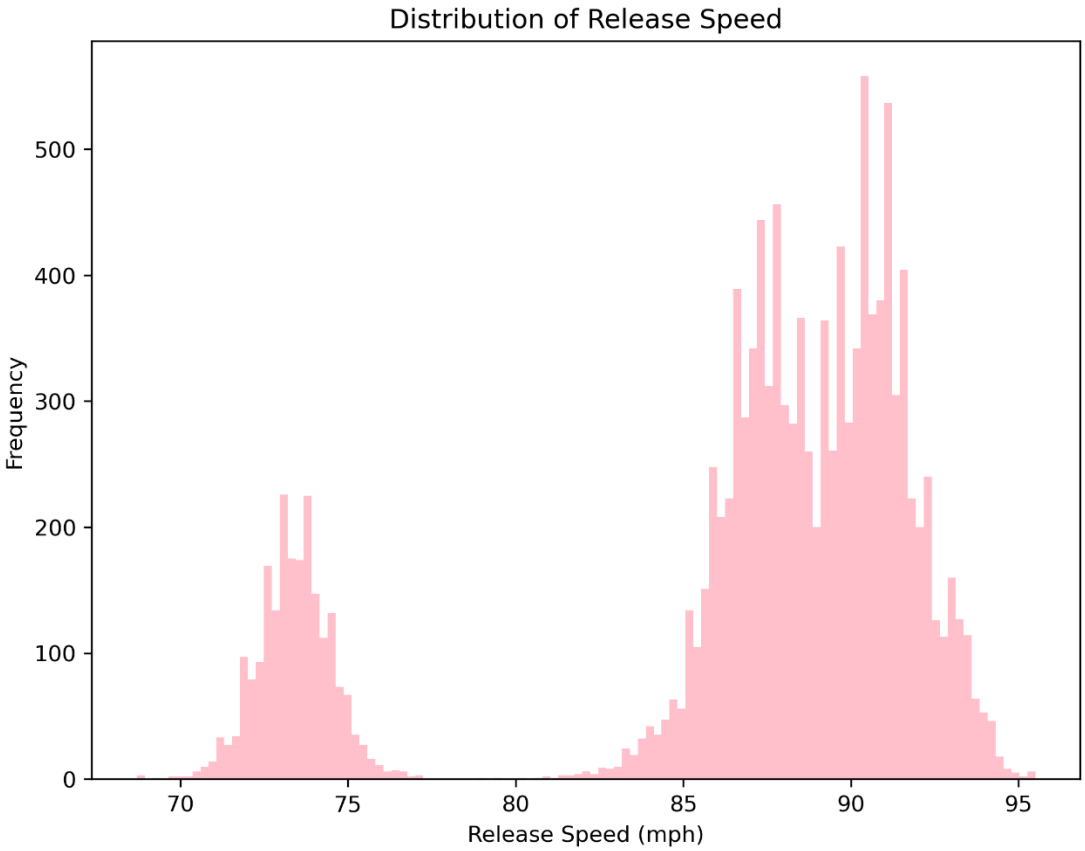

Figure 1: The distribution of target variable (Release Speed)

Feature analysis:

For all other features, I selected the most related features to do the further steps. The features I selected are: pitch_type, release_speed, release_pos_x, release_pos_y, release_pos_z, balls, strikes, outs_when_up, stand, p_throws, ax, ay, az,release_extension. Also, I showed all distributions of these features as figure 2.

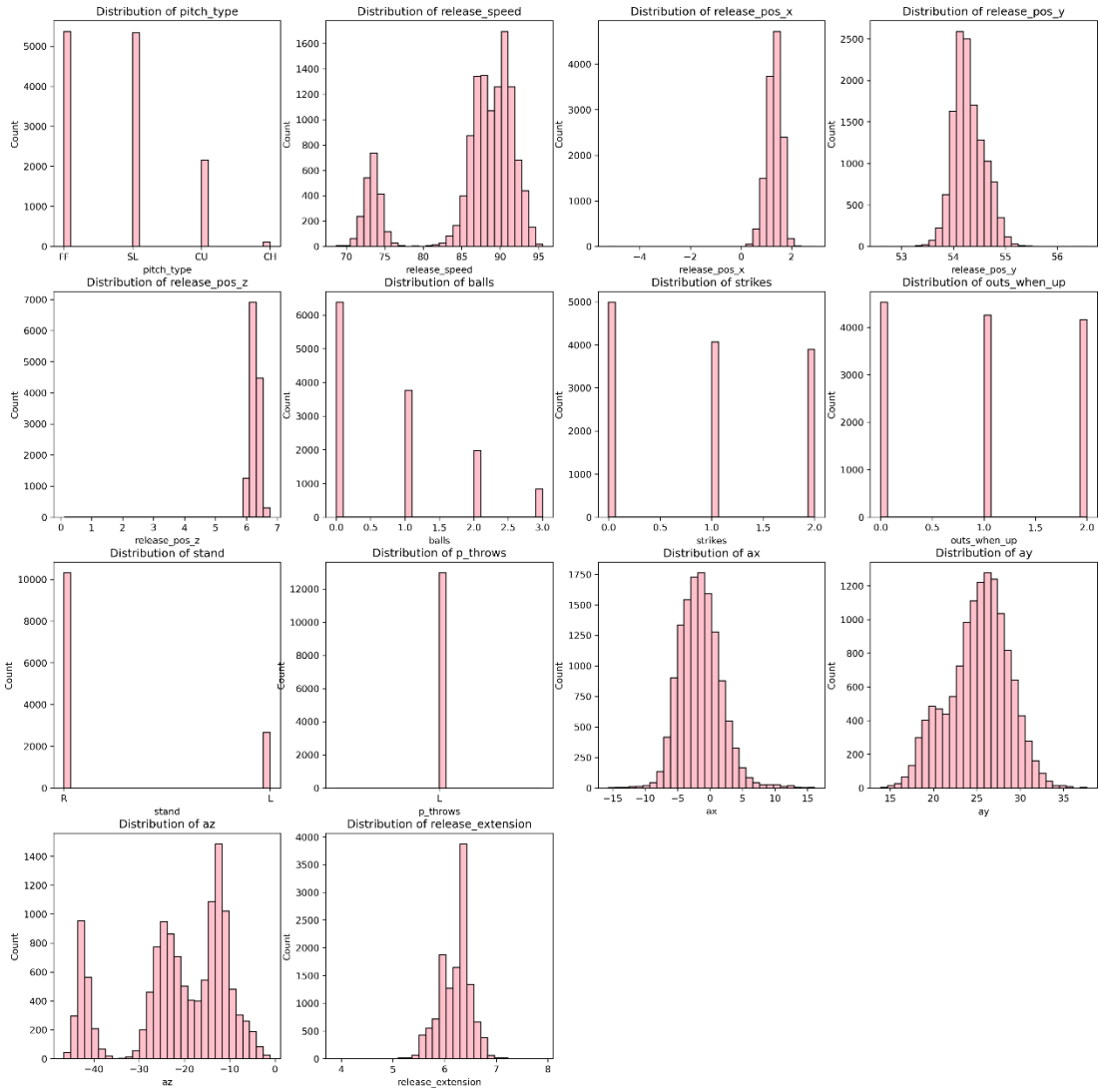

Figure 2: Distributions of all selected features

Next, my dataset is based on different pitch types. Thus, the dataset is not iid. I analyzed and plotted the percentage of each pitch type among all as figure 3.

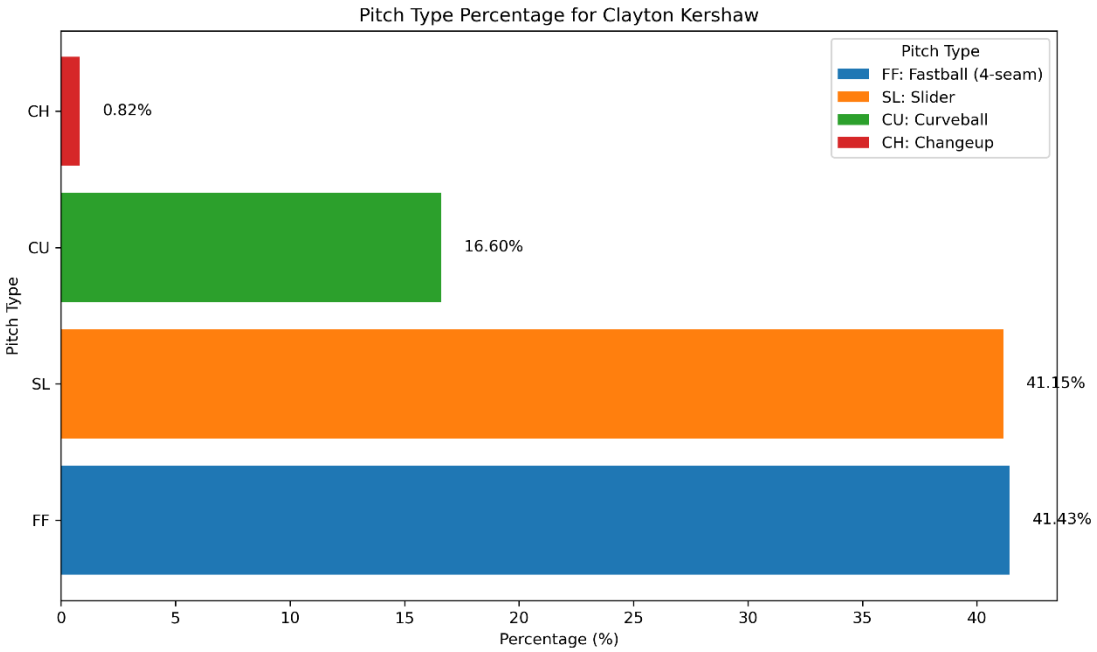

Figure 3: Pitch type percentage for Clayton Kershaw

The most interesting thing I did in EDA is to explore whether Kershaw was affected by COVID 19. Thus, I plotted a box plot to figure out as figure 4.

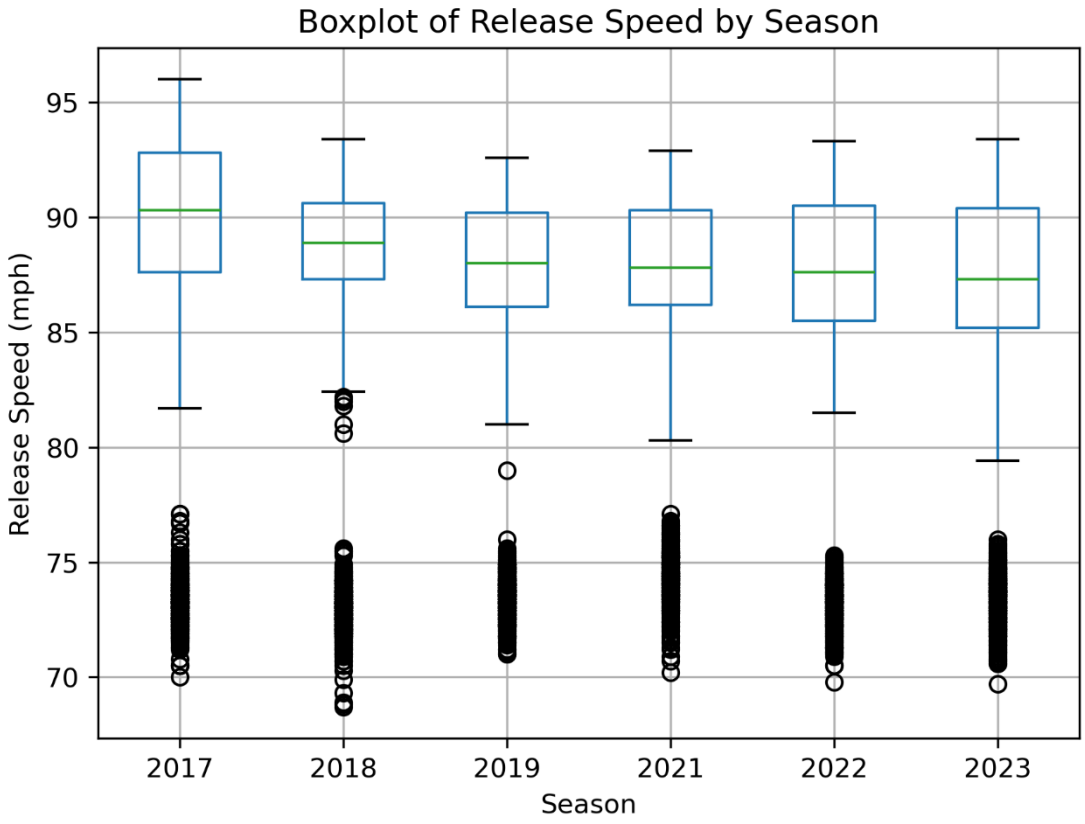

Figure 4: Boxplot of release speed by season

After plotting this, I found that the trend and distribution of release speed were almost the same. Thus, he was not affected by COVID 19.

3. **Method:**

Splitting strategy:

In the project, the dataset is based on group ID which is pitch type. I used GroupShuffleSplit and selected 20% of the data as the test set. This grouping method avoids the same pitch type appearing in both the training and test sets. Then, I used GroupKFold and set 3-fold cross-validation on the training set. For each time divide the data into training and validation sets according to pitch type. This method can ensure that the characteristics of different pitch types are covered in the training and validation sets each time, improving the stability of model performance. By cycling through 5 different random seeds, I have evaluated the performance of the model under different data multiple times, reducing the result fluctuations. This is better than only splitting data just once.

Preprocessing:

There are three categorial features: pitch_type, stand, and p_throws. For these categorial features, I used OneHotEncoder to convert them into numerical formats. For other continuous features, I used StandardScaler for them to ensure that all continuous features have the same scale range. And there is no ordinal features that I selected.

ML pipeline:

In ML pipeline, I put the splitting and preprocessing parts into the pipeline function. In the first step of the pipeline, it is the splitting part using GroupShuffleSplit and GroupKFold. In order to find the best model parameter combination, I used GridSearchCV in the function. By providing a parameter grid (param_grid), a search process is performed on different parameter combinations of the model. Perform 3-fold cross validation on the training set and calculate the negative root mean square error (neg_root_mean_squared_error) of the model on the validation set. After the search is completed, the model with the smallest error on the validation set is selected as the best model (best_model). In this project, I chose to use the root mean square error (RMSE) as the evaluation metric. RMSE has intuitive interpretation, and the unit is consistent with the target variable (pitch release speed). At the same time, RMSE is a commonly used evaluation metric in regression problems, which can reflect the changes in model performance in hyperparameter tuning and cross-validation.

In this project, six algorithms are used and for each algorithm, several parameters are tuned as shown in Table 1.

Supervised ML algorithms	Parameter
Linear regression Lasso regularization	alpha = logspace(-7, 0 ,29)
Linear regression Ridge regularization	alpha = logspace(-10,0,51)

Linear regression Elastic net	alpha = logspace(-3,1,10) l1_ratio = [0.1, 0.3, 0.5, 0.7, 1.0]
Random Forest regression	max_depth = [1,2,3,10,30] max_features = [‘sqrt’, ‘log2’]
SVR	gamma = [1e-3, 1e-1, 1e1, 1e3] C = [1e-2, 1e-1, 1e0, 1e1]
XGBoost	learning_rate = [0.03] n_estimators = [100] seed = [0] reg_alpha:[0e0, 1e-2, 1e-1, 1e0, 1e1, 1e2] reg_lambda = [0e0, 1e-2, 1e-1, 1e0, 1e1, 1e2] max_depth = [1,3,10,30,100] colsample_bytree = [0.9] subsample = [0.06]

Table 1: Tuned parameters for each machine learning algorithm

For the end of the pipeline, I stored the best model, test score, result and the elements we used in future into lists for each random state.

4. **Results:**

The baseline is the mean of the target variable (release speed) in the whole dataset. And I computed the baseline RMSE which is about 6.273.

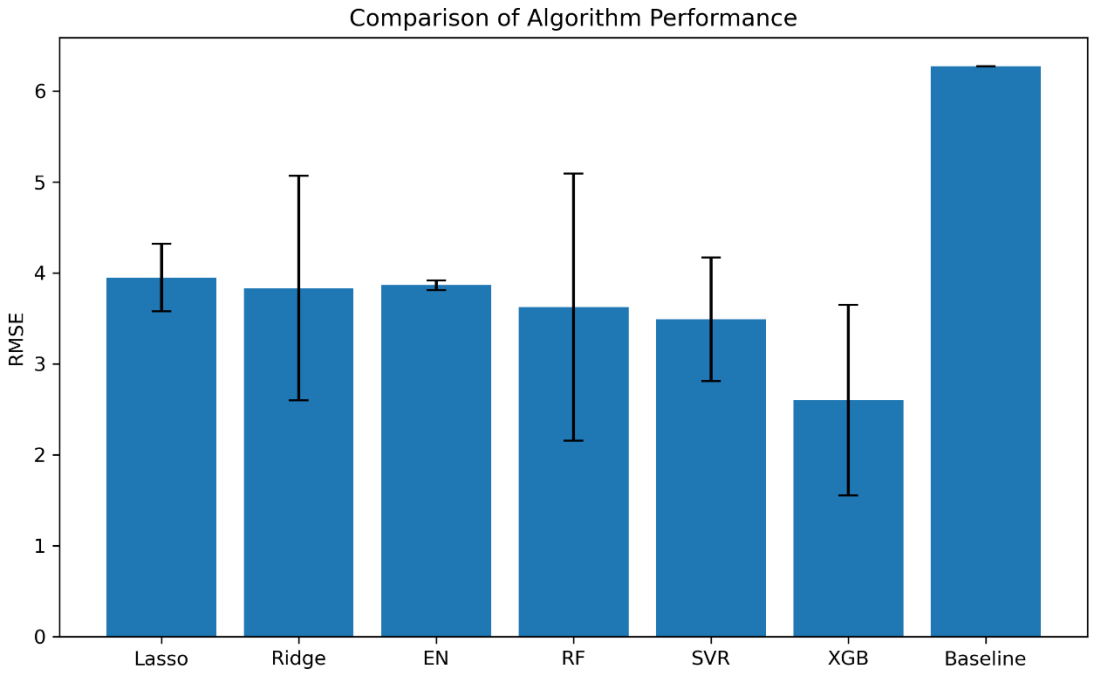

Figure 5: Comparison of algorithm performance with baseline

Figure 5 shows that RMSE of all six algorithms compare with baseline and error bar of each algorithm shows the standard deviation. And the error bars show the uncertainties of my evaluation metric due to splitting. From figure 5, XGBoost has

the lowest RMSE (2.600) which means this is the best algorithm among all and the most predictive. I chose XGBoost algorithm to do next steps.

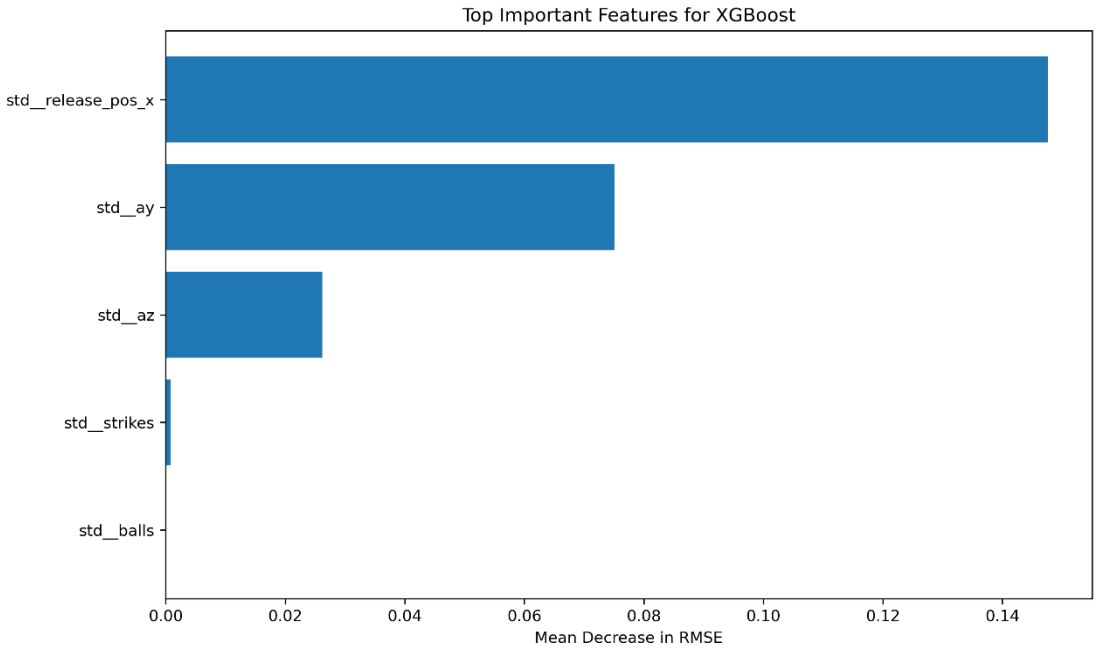

Figure 6: Top permutation importance feature using XGBoost

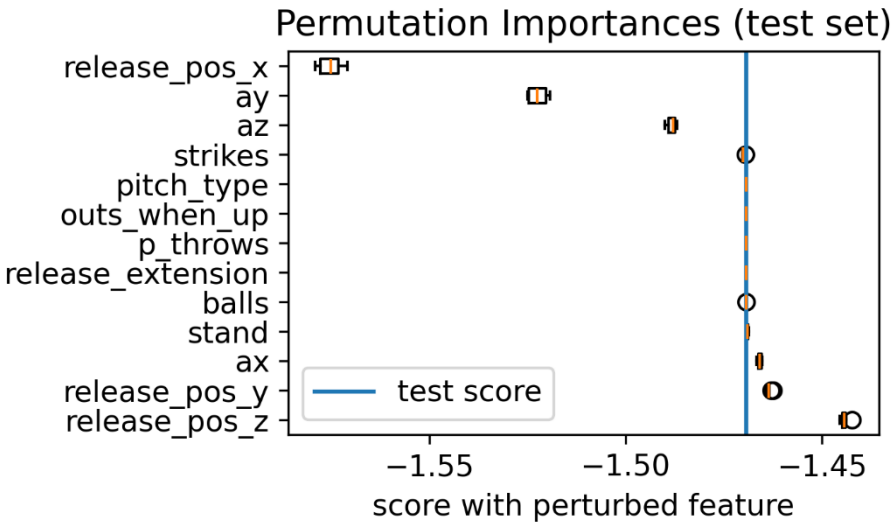

Figure 7: Top permutation importance feature using another method

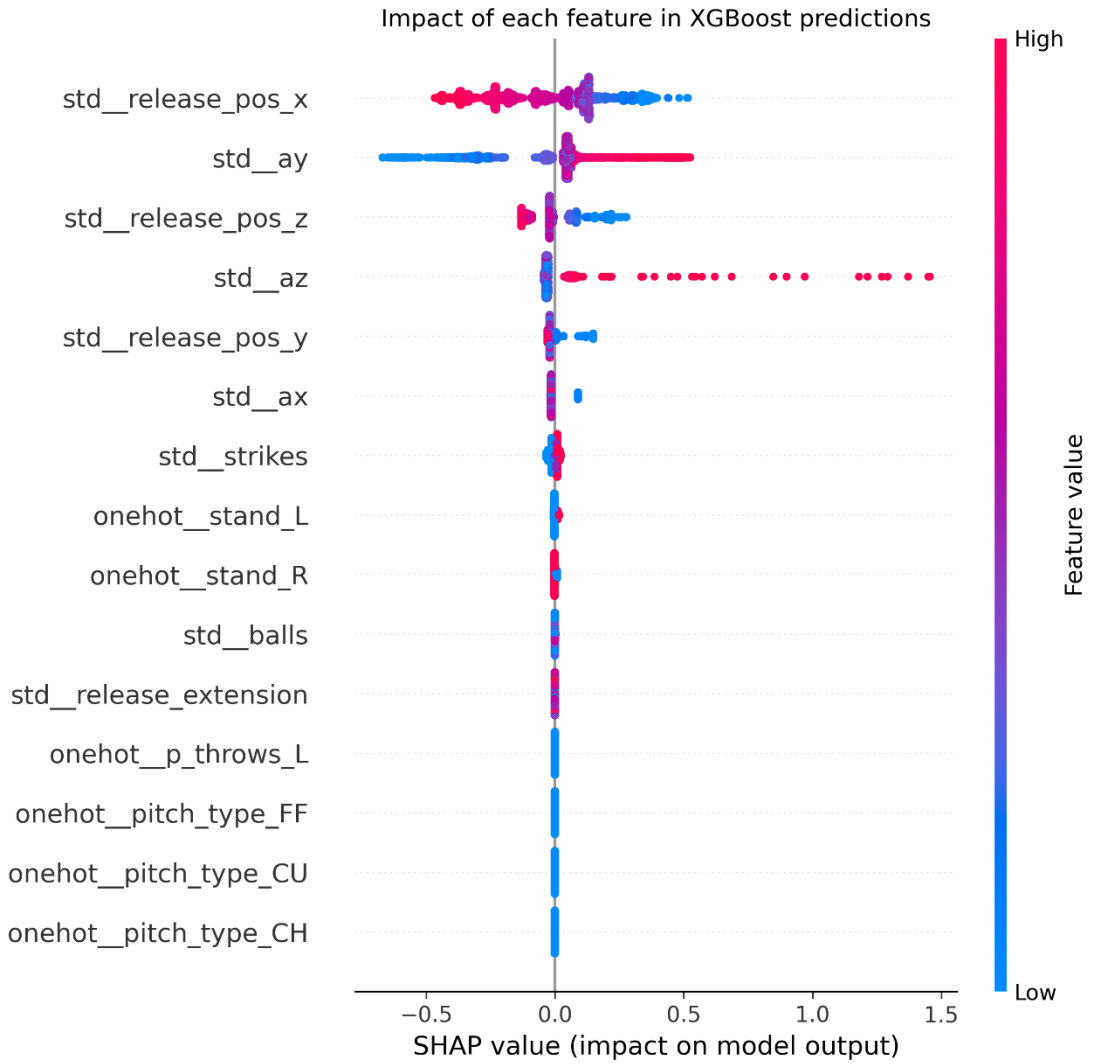

Figure 8: Impact of each feature in XGBoost predictions

In global feature importance using permutation importance function as figure 6 and the method from lecture note as figure 7, permutation importance results of XGBoost model show that release_pos_x has the greatest impact on the prediction of the target variable. Meanwhile, balls is evaluated as less important. In global feature importance using SHAP value as figure 8, the plot shows the same result as the permutation importance. The release_pos_x is the most important. There are some features with 0 SHAP value which means these features are all the least important.

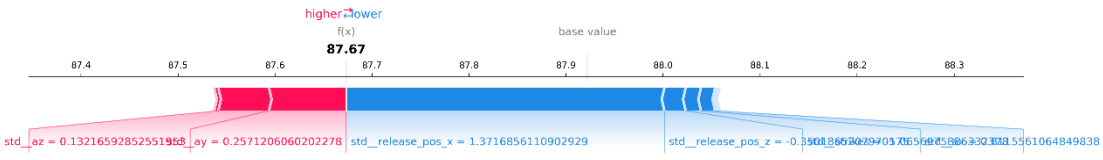

Figure 9: Local SHAP value for index = 0

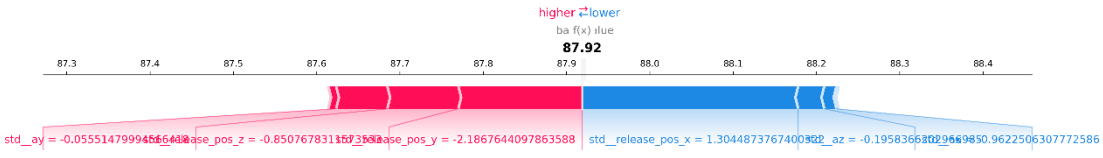

Figure 10: Local SHAP value for index = 50

In local SHAP value analysis as figure 9 of the point at the index = 0 and figure 10 of the point at the index = 50, the features in the red area like ay, release_pos_z have the positive influences on the predictions and the features in the blue area like release_pos_x, az have the negative influences on the predictions. I found that in global importance analysis, release_pos_x is the most important feature, but in the local importance analysis, release_pos_x has the negative influence. Thus, it is necessary to do both analyses. They may have the opposite results.

5. **Outlook:**

Adding interaction terms between features is an important improvement direction, which can capture the potential nonlinear relationship between features. In addition, introducing weather features related to the game (such as temperature and humidity on the day of the game) can help to more comprehensively describe the factors affecting the target variable, thereby improving the performance of the model.

Further optimization of the model's hyperparameters (such as learning rate, regularization coefficient, etc.) is another important direction to improve model performance.

Because of the imbalanced dataset, it is recommended to try other more appropriate methods, such as StratificationGroupKFold. This method can perform stratified sampling based on grouped data, thereby ensuring the consistency of data distribution in the training set and test set.

6. **References:**

[1] Baseball Savant: Statcast, Trending MLB Players and Visualizations | baseballsavant.com

[2] Statcast - Wikipedia

[3] Casella, Paul (April 24, 2015). "Statcast primer: Baseball will never be the same". *MLB.com*. Retrieved September 30, 2015.