MLB Pitch Prediction

Team members:

1. **Jonathan Castellanes**
2. **Hubert Castellanes**
3. **Gaurav Batra**

SCS 3253 - *024*

Team 6 Baseball

MACHINE LEARNING | UNIVERSITY OF TORONTO

Table of Contents

Objective 2

Data Sources 2

Data Preparation 2

Data Exploration 3

Data Visualization 4

Approach and Analysis 6

Applying the Results to the problem 7

Conclusion 9

References 9

Choosing the Correct Pitch Sequences: Data-Driven Decisions 9

Teaching Effective Pitching Sequences 9

Appendix/Project Notebook 10

Objective

Utilize machine learning to predict a pitch based Major League Baseball (MLB) in game situation

The task of hitting a baseball at the major league level is extremely difficult. If the batter knew what pitch the pitcher was likely to throw next, his chances of success could improve significantly.

This project aims to discover and analyze the pitch data with aims to predict pitches using machine learning techniques. When predicting pitches, it is best to select a pitcher find a model that best fits him instead of modelling on all pitchers. In this case, Max Scherzer was chosen because of his high pitch count. Of those techniques the team used Decision tree classifier, KNN classifier, MultiLayerPerceptron Classifier, Random forest classifier and XGBoost. Then evaluated models and used the best one for prediction.

The XGBoost model was chosen by fine tuning the model that achieved roughly 59% accuracy and precision.

Data Sources

***Major League Baseball Pitch Data* from Kaggle**

- Pitch-level data for every pitch thrown during the 2015-2018 MLB regular seasons from MLB Pitch Data 2015-2018

- Data scraped from http://gd2.mlb.com/components/game/mlb/. Each row represents a single pitch.

Data Preparation

Data Pre-Processing:

- Dropped Unnecessary attributes/columns
- Removed all rows contain an NaN.
- This project focused on one pitcher.
- Extracted the pitcher’s Player ID using their name.
- This Player ID is used to find all At Bat IDs to pertaining to the selected Player ID.
- Each At Bat IDs can have multiple pitches associated to it
- The goal is to get all pitches related to a Player ID from 2015 to 2018 season.
- These values were hot encoded (type_B, type_S , type_X, standL)
- Pitches were removed ['Foul Out', 'Pitch Out', 'Intentional ball', 'UNknown', 'FA', 'AB']
- Pitch types were grouped to FB= fastball, BR = breaking ball, OS = offspeed
- The grouped pitch types were label encoded

Data Splitting:

- 80% for training and 20% for testing and validation

Data Exploration

These are the final features used in the modelling and analysis.

Independent variables :

- B_score - batter’s team score
- B_count - number of balls in the count
- S_count - number of strikes in the count
- Outs - number of outs in the inning
- On_1b - is someone on 1st base
- On_2b - is someone on 2nd base
- On_3b - is someone on 3rd base
- Inning - what inning is it in the game
- P_score - pitcher’s team score
- type_B - ball result
- type_S - strike result
- type_X - other result
- stand_L - whether or not the batter stands left or right

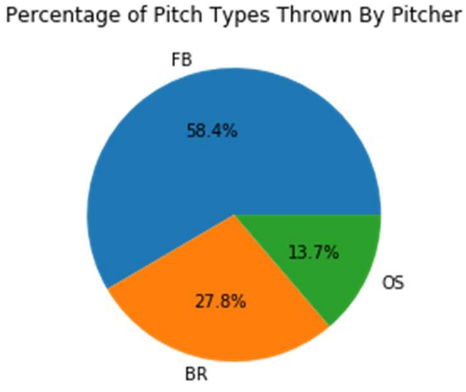
Dependent Variable :

- Pitch Type

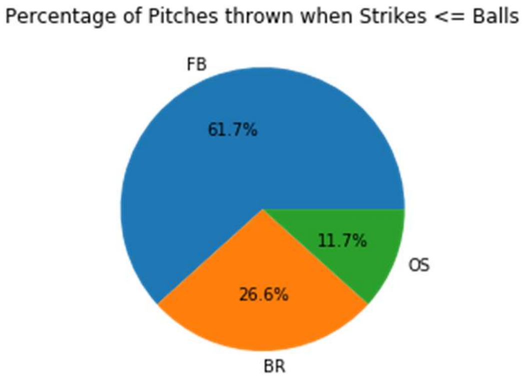
Data Visualization

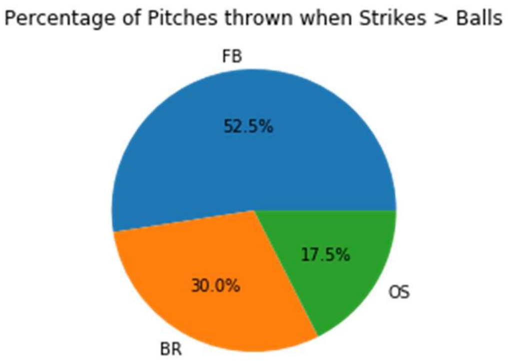
As stated in the object, this project and modelling will be focusing on a single pitcher Max Scherzer.

The pie chart below details a breakdown of all of Max's pitches from 2015 to 2018. Each pitch is grouped into one of three types of pitches: Fastball (FB), Breaking ball (BR), and Off speed (OS). As it stands, Max has a tendency to throw fastballs in the three years sampled. This is the baseline of all his pitches for those years.



The pie charts below details Max distributes his pitches when he is behind in the count and ahead in the count (strikes > balls).

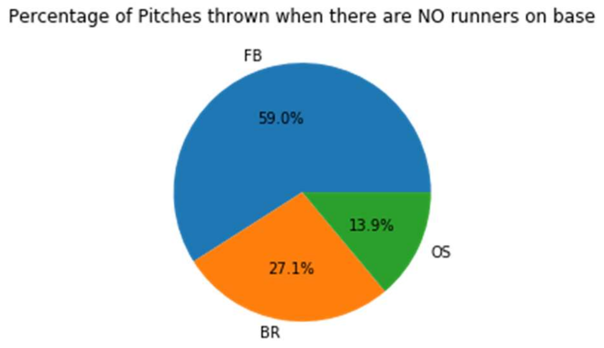


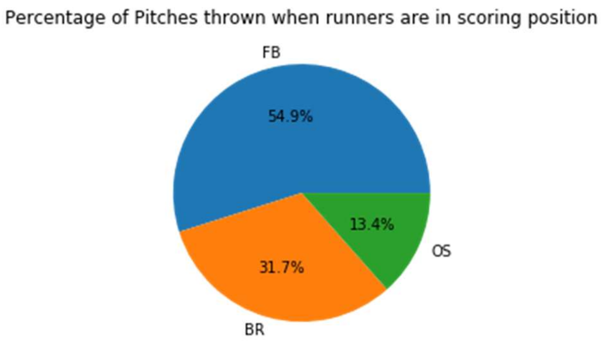


The pie chart below details Max distributes his pitches when he is behind on the count against a batter. Compared to being ahead of the count Max will throw almost 10% more fastballs.

The reason for the increased FB frequency when a pitcher is behind in the count is due to the fact that fastballs are the easiest to throw for strikes to get a batter out. When a pitcher is ahead in the count, they are more likely to throw breaking balls and offspeed pitches to get batters to chase pitches outside of the strike zone.

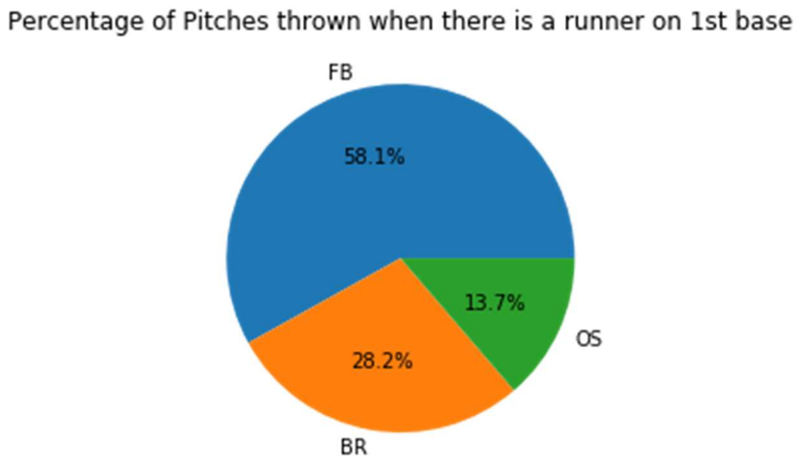
In the pie chart below, when there are no runners on base, Max’s pitch selection is close to his baseline 2015 to 2018 data source.

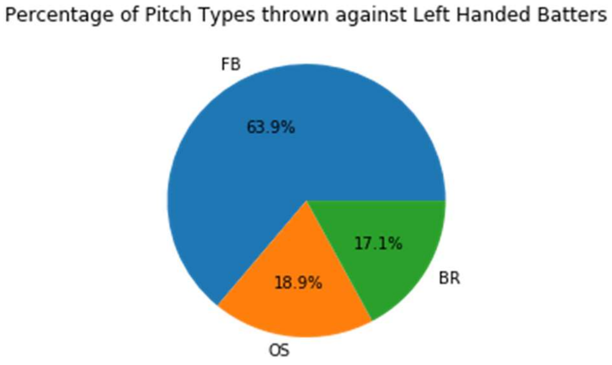

Percentage of Pitches thrown when there are NO runners on base

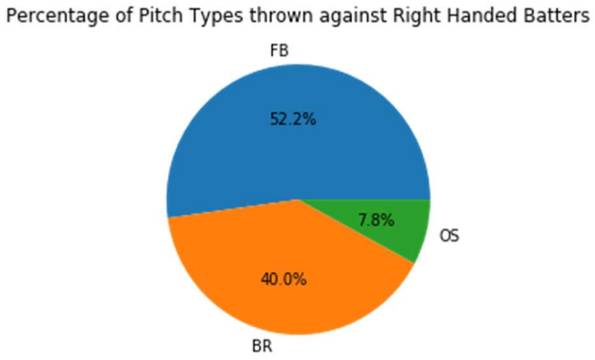

Percentage of Pitches thrown when runners are in scoring position

When runners are in scoring position, Max will likely throw more breaking balls and offspeed pitches to induce ground balls plays.

When there is a runner on first base, his pitch selection is similar to the NO runners on base. There is a slight increase in breaking balls to slow runner progress.


Percentage of Pitches thrown when there is a runner on 1st base


Percentage of Pitch Types thrown against Left Handed Batters


Percentage of Pitch Types thrown against Right Handed Batters

Approach and Analysis

To establish a baseline, a random guess model was used and it had an accuracy of approximately 44%. kNeighbors Classifier model was used next, the model high accuracy against training data (67%) and a low accuracy against test data (53%).

Decision Tree Classifier had an accuracy of 90% on training data. The validation test data accuracy fared much worse at 49%. This suggests Decision tree model was overtrained..

The Random Forest Classifier (RFC) model was chosen because of the decision tree classifier. The RFC had an 88% training data accuracy, with test data the accuracy was 50%. This model suggested overtraining less so that the Decision tree model.

Default MLP was then used to model because it is a neural network. Default MLP Classifier had a 62% accuracy against training data. But had 57% accuracy with test data.

The model XGBoost classifier was the final model. It was chosen over other boosters because it is considered one of the best gradient boosters. With grid search cross validation hyper tuning (grid search CV), the model produced a 59% accuracy with training data and 59% with validation data.

As XGBoost had the highest accuracy with validation data, XGBoost was chosen for predicting.

The XGBoost classification report for the training data.

	Precision	**Recall**	**F1-Score**	**Support**
BR	0.49	0.12	0.20	2987
FB	0.6	0.96	0.74	6241
OS	0	0	0	1476
micro avg			0.59	10704
macro avg	0.34	0.36	0.31	10704
weight avg	0.48	0.59	0.49	10704

Here is the XGBoost classification report for the test data.

	Precision	Recall	F1-Score	Support
BR	0.43	0.12	0.18	736
FB	0.61	0.95	0.74	1579
OS	0	0	0	361
micro avg			0.59	2676
macro avg	0.34	0.36	0.31	2676
weight avg	0.48	0.59	0.49	2676

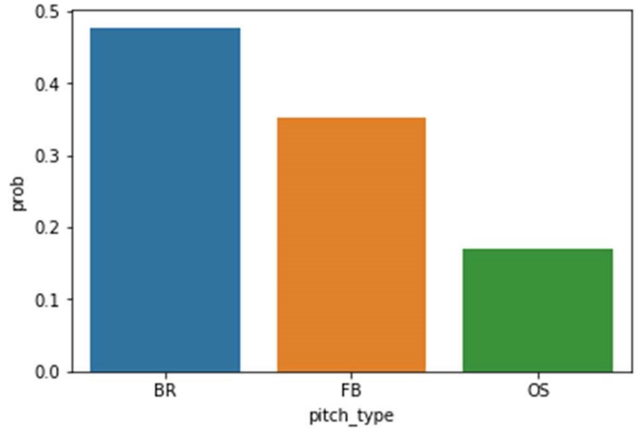
After testing the model, the accuracy was slightly lower than 60%. So the approach now is to use the pitch probabilities. The application utilizing pitch probabilities can be seen in the following section.

Applying the Results to the problem

Here is a test using 3 random sample data. The random samples input used in the model's prediction matched the actual result and it concludes the model is efficient.

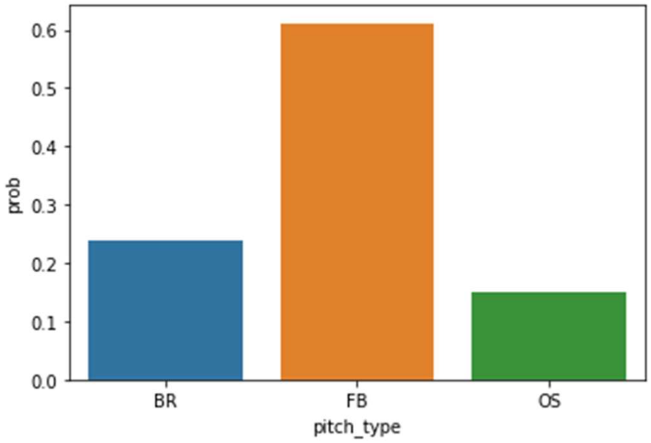
Random Sample #1

Index: 1931
Actual: BR (breaking ball pitch)
Predicted: BR (breaking ball pitch)



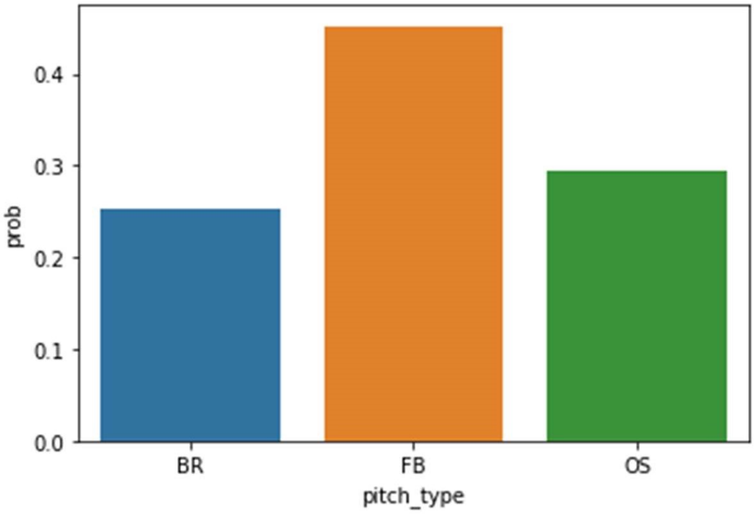
Random Sample #2

Index: 1641
Actual: BR
Predicted: FB



Random Sample #3

Index: 1206
Actual: FB
Predicted: OS



Conclusion

The Baseball pitch predictions has endless possibilities of predictions. As a preliminary attempt, we finalized XGBoost model for prediction with 59% accuracy for one specific pitcher. As there are well over 2000 pitchers, an individual prediction model would be required for each pitcher.

From the perspective of a baseball game, this project gives insight into the factors that predicts pitch probability. Pitchers typically use a pitching sequence to determine the best possible order of pitches to use on a batter (taking into account how a batter is currently hitting, what areas in the strike zone is the batter likely to hit) assuming that a pitcher wants to get a batter to strike out.

However, this project attempts to do what is known as “game calling” prediction, wherein the catcher and pitcher decide on what pitch to throw depending on the current circumstances (such as if there are runners on base, the strike and ball count, and how much of a lead a pitcher’s team has). As well, pitchers have to take into account what kind of result they want out of the at bat given the situation (do they want a strikeout, infield groundouts, fly ball outs, etc.).

One method to explore is to use a time series based neural network to aid in pitch prediction. This will simulate a model in which catchers and pitchers take previous pitches to a given batter into account. This can be done with tensorflow. As this project is a preliminary exploration into pitch prediction, and due to time constraints, not much was done with tensorflow at this time.

References

Choosing the Correct Pitch Sequences: Data-Driven Decisions

https://www.drivelinebaseball.com/2012/05/choosing-the-correct-pitch-sequences-data-driven-decisions/

Teaching Effective Pitching Sequences

http://myyouthbaseball.com/teaching-effective-pitching-sequences.html

Appendix/Project Notebook