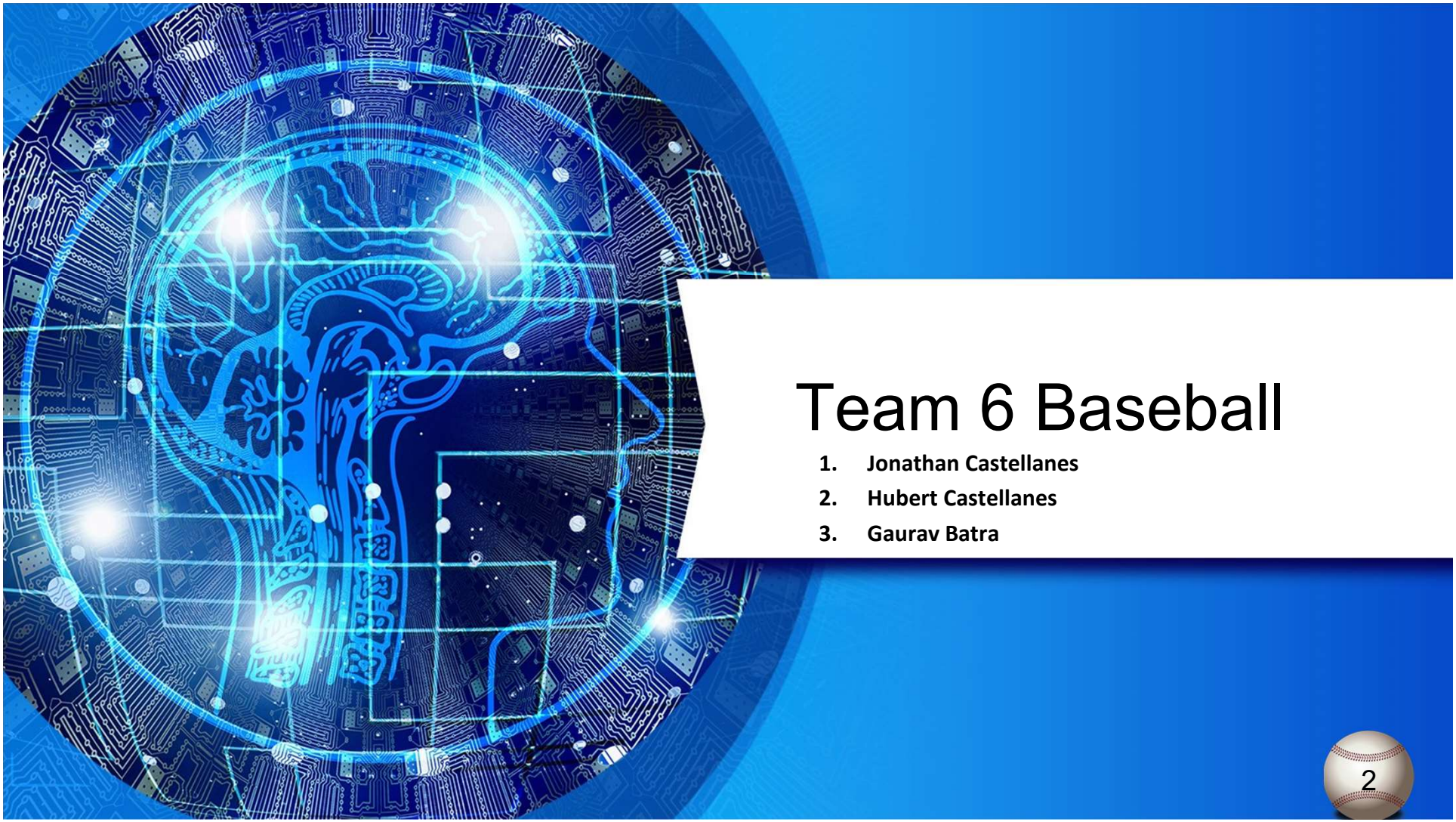
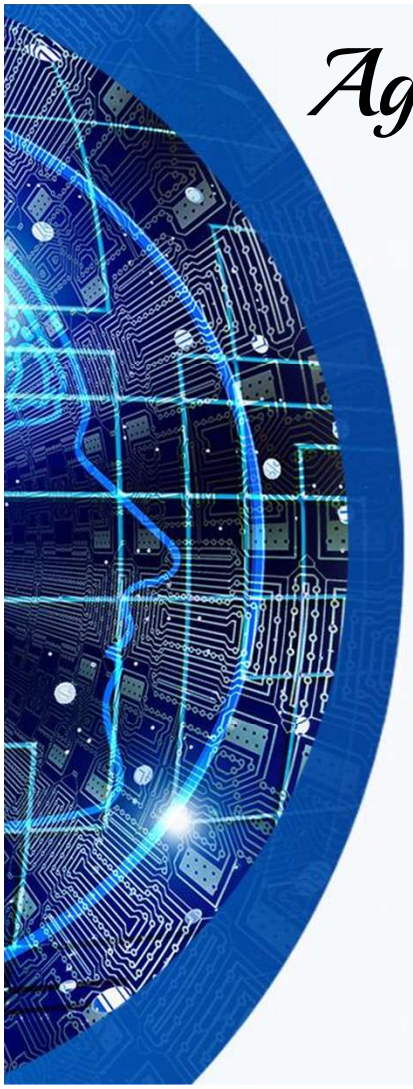

MLB Pitch Prediction



Team 6 Baseball

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



Agenda

- Objective
- Data Preparation
- Data Exploration
- Data Visualization
- Approach & Analysis
- Test Data
- Model & Tuning
- Results
- Conclusion

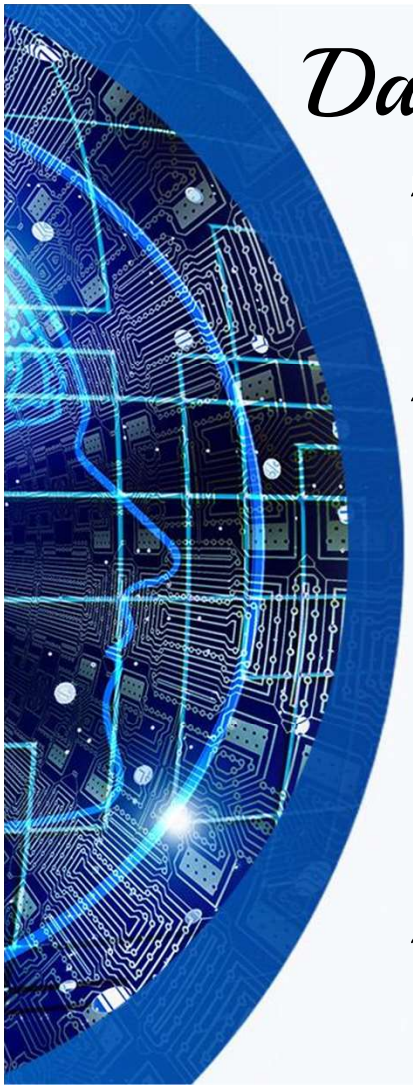
Objective

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 Preparation

Data Source:

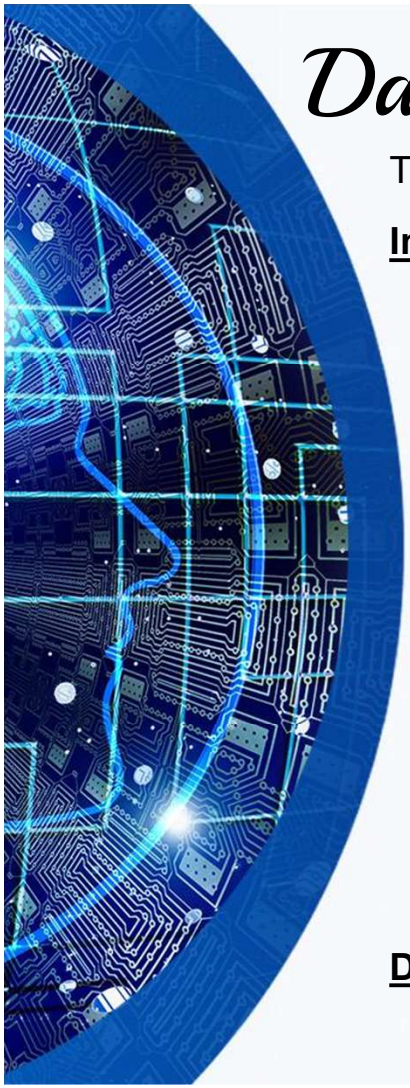
- Pitch-level data for every pitch thrown during the 2015-2018 MLB regular seasons is collected from Kaggle - MLB Pitch Data 2015-2018

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

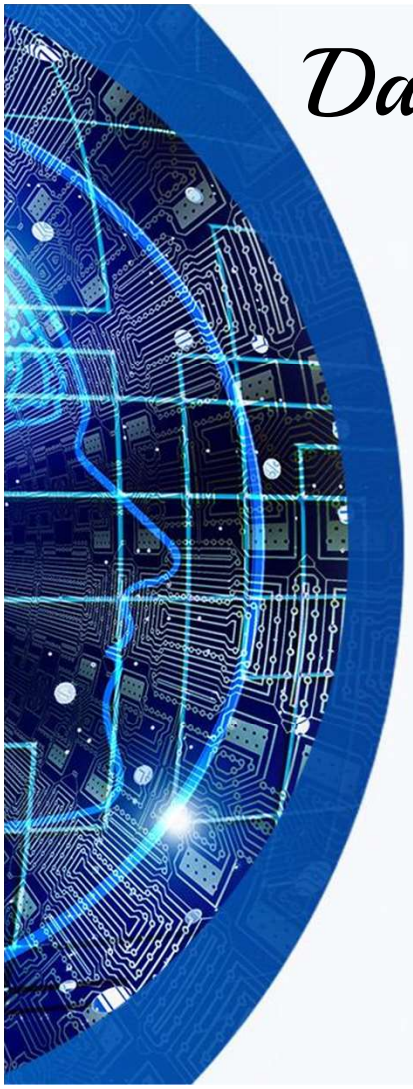
These are the final features used in the modelling and analysis.

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

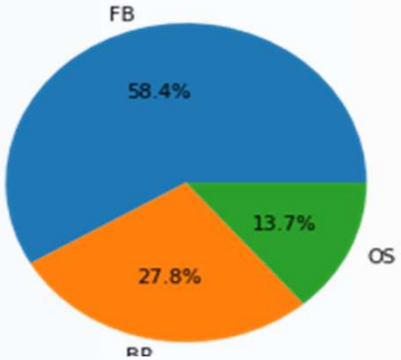
Dependent Variable :

- Pitch Type

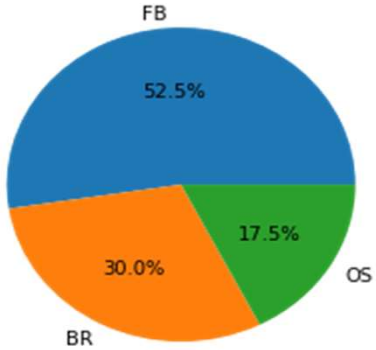


Data Visualization

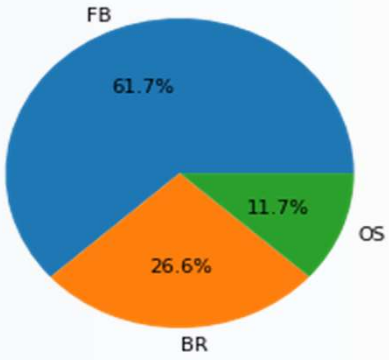
Percentage of Pitch Types Thrown By Pitcher



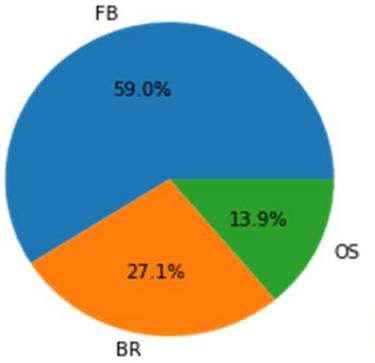
Percentage of Pitches thrown when Strikes > Balls



Percentage of Pitches thrown when Strikes <= Balls

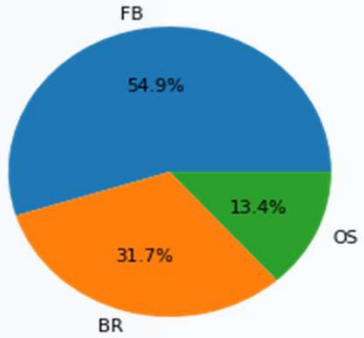


Percentage of Pitches thrown when there are NO runners on base

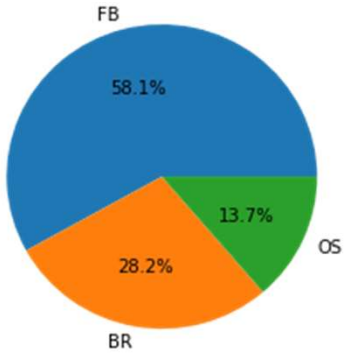


Data Visualization

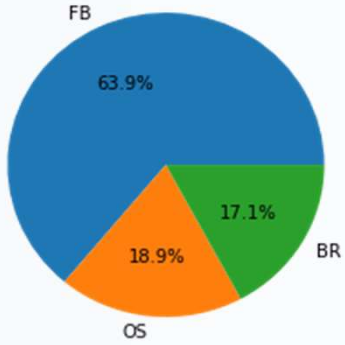
Percentage of Pitches thrown when runners are in scoring position



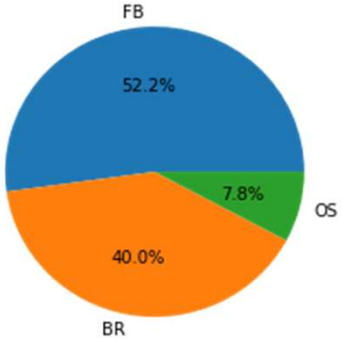
Percentage of Pitches thrown when there is a runner on 1st base

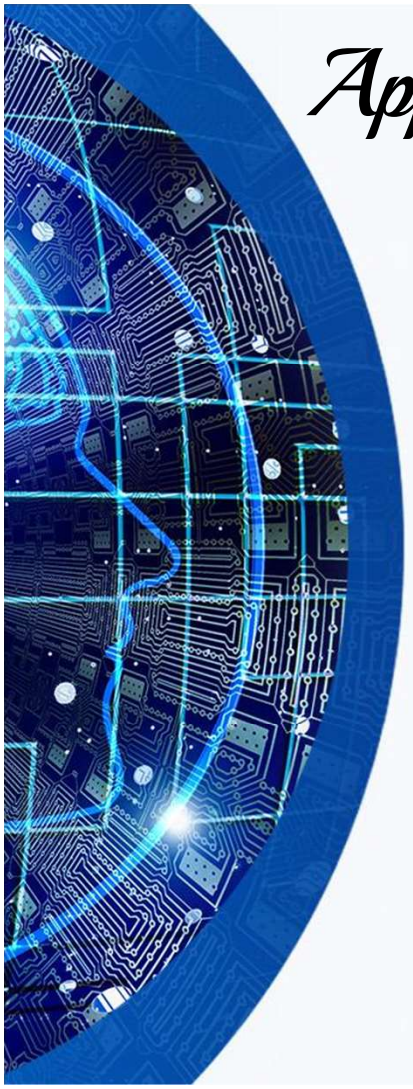


Percentage of Pitch Types thrown against Left Handed Batters



Percentage of Pitch Types thrown against Right Handed Batters





Approach & 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%).

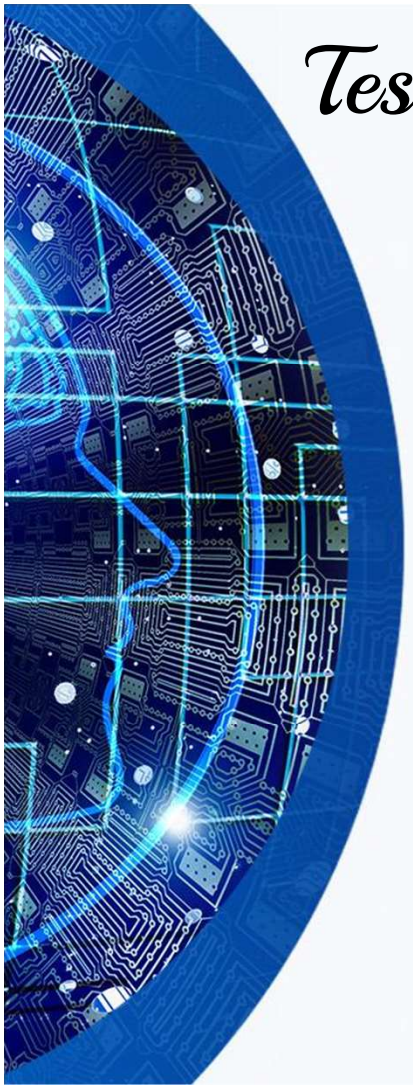
Decision Tree Classifier had an accuracy of 90% on training data. The validation test data accuracy fared much worse at 49%. This suggests Decision tree model was overtrained..

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.

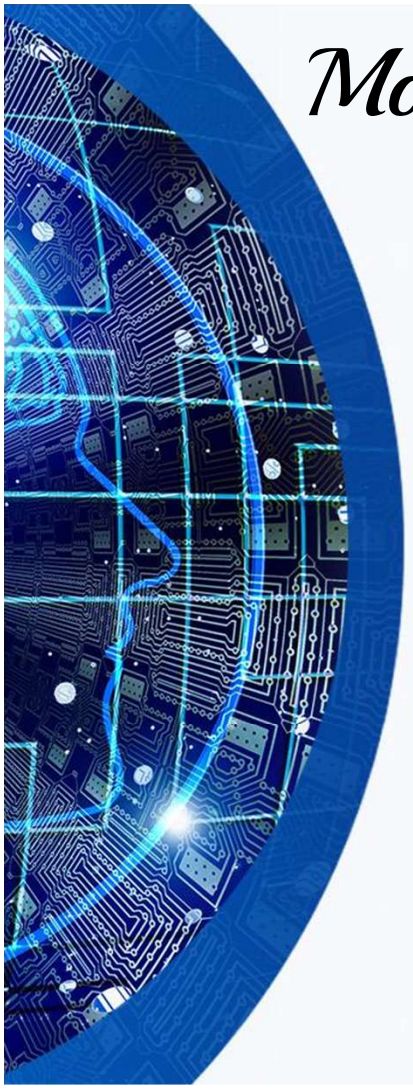
As XGBoost had the highest accuracy with validation data, XGBoost was chosen for predicting.



Test Data

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



Model & Tuning

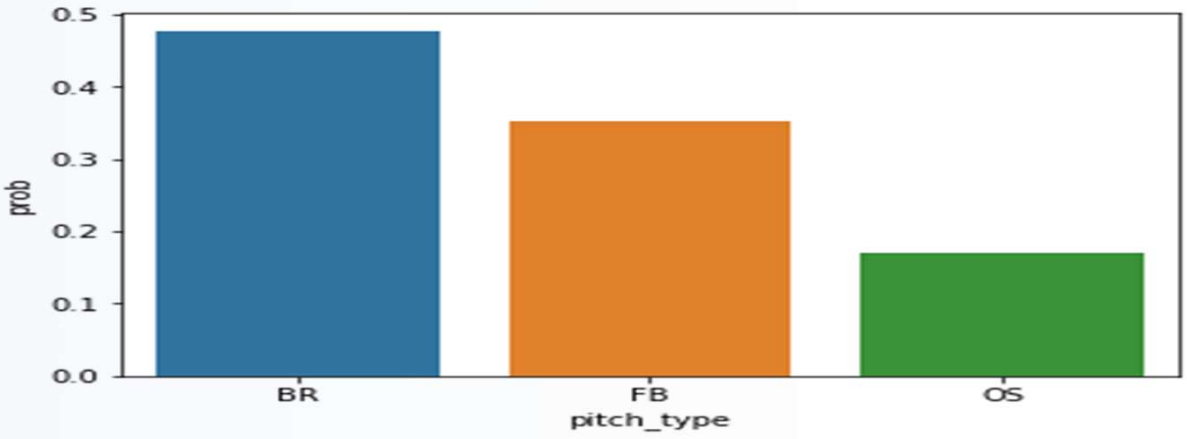
We performed Hyperparameter tuning for XGBoost using a grid search cross validated estimator and generated a classification report for the best estimator for the training data.

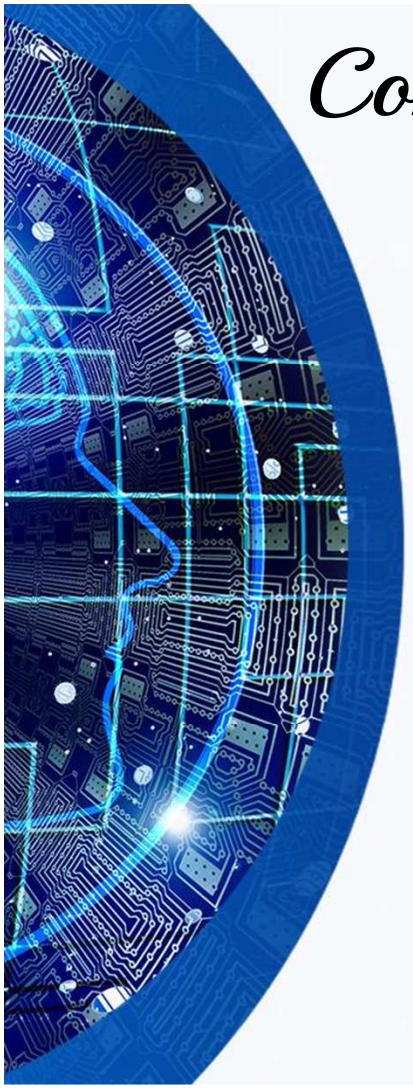
	Precision	***Recall***	***F1-Score***	***Support***
BR	*0.49*	*0.12*	*0.2*	*2987*
FB	*0.6*	*0.96*	*0.74*	*6241*
OS	*0*	*0*	*0*	*1476*
micro avg			*0.59*	*10704*
macro avg	*0.36*	*0.36*	*0.1*	*10704*
weight avg	*0.49*	*0.59*	*0.49*	*10704*

Results

The model is not accurate which indicates we would have to rely on probabilities to predict the pitch. The graph is designed to aid in helping a PERSON to predict the likelihoods of pitch types.

Index: 1641
Actual: BR (breaking ball pitch)
Predicted: BR (breaking ball pitch)





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.


Thank You
Adobe
Driving the future of Digital Media & Digital Marketing