Building Machine Learning Models to Predict the Outcome of a Pitch in an MLB Game

Authors: Zachary Becker, Zachary Armand, Zhaofeng Li

Northeastern University

DS 5500 Capstone: Applications in Data Science

Spring 2025

March 10, 2025

Contents

Chapter 1

Introduction and Purpose of the Methodology

The goal of our project is to take a machine learning approach to predicting the most likely outcomes for batted balls in a Major League Baseball game. We focus on predicting the launch angle, which is the trajectory that the baseball takes when leaving the bat, and the exit velocity, which is the velocity at which the ball leaves the bat. We build two main models for this task, one that uses XGBoost, and one that uses a Recurrent Neural Network.

The problem of predicting batted ball outcomes based on pitch characteristics and hitter-specific historical data requires a combination of different machine learning methods to capture various relationships smong input features and expected performance. Our approach uses K-Means clustering, K-Nearest Neighbors, XGBoost, and a Recurrent Neural Network to try and build a process of solving our problem effectively.

The XGBoost model and RNN together help us gain a better understanding of the strengths and limitations of the two approaches. The XGBoost will have strong predictive performance with how it models feature interactions implicitly, and the RNN allows us to analyze pitch sequencing, which allows for clubs to understand how various pitch patterns influence hitter success. Comparing the two will overall allow for a holistic approach to the modeling.

Chapter 2

Problem Statement

2.1 Defining the Problem

Our project is a Machine Learning problem, where we are trying to predict the most likely batted ball outcome of a specific type of pitch in a Major League Baseball game. Specifically, it is a regression problem where we want to predict the exit velocity and launch angle of a batted ball, and then use those metrics in an expected wOBA (weighted on-base average) calculation. The prediction is based on the characteristics of a pitch profile, such as velocity, spin rate, release extension, spin axis, location, etc., as well as contextual information about the batter's historical performance by pitch type and location.

2.2 Significance

Predicting batted ball outcomes in Major League Baseball games is crucial for the clubs' in-game strategy as well as player evaluation. We are entrenched in the era of data-driven baseball, where teams no longer are relying solely on traditional scouting reports or simple counting stats for their player evaluation, projection, and strategic decision making. Every club is now using various machine learning models in tandem with their traditional scouting techniques to decide how they want to approach a season, a game, or even a specific pitch. In this case, knowing which type of pitch profile is most likely to provide a positive outcome for the defense is an obvious opportunity for a competitive advantage. Pitching staffs are able to play into their pitcher's strengths as well as try and expose weaknesses of the hitters, and at the same time, the hitters are able to gain a better sense of how the pitcher is likely to approach them in an at-bat.

Beyond individual matchups, this data can influence defensive positioning, player development, and even front-office decisions on trades and contracts. If a club feels that they can identify some pitch profiles that have more success than other common pitch profiles, they can attempt to acquire pitchers who have an arsenal similar to what the player development staff desires.

Chapter 3

Data Collection and Preparation

3.1 Data Sources

For this project, we are using Statcast data from the 2024 MLB season that is publicly available for download from Baseball Savant. To access the data, we use the python package *pybaseball* to directly access the Baseball Savant API via python. On the GitHub repository, we include a link to the package's GitHub page, which includes more details about the package.

3.2 Data Description

The dataset that we use has a row for each pitch thrown in the 2024 MLB season, and contains a wide selection of variables that describe player and game contexts, pitch data, batted ball data, and more. There were around 750,000 pitches thrown in the 2024 MLB season, but we only looked at the pitches that were put into play. Some of the key features used to do the pitch profile segmentation include the pitch type, the velocity, spin rate, spin axis, vertical break, horizontal break, release extension, and release height. For the hitter-specific data, we look at the most common launch angles and exit velocities by pitch location (divided into 16 zones) as well as the by pitch type. Some hitters will fare better against specific types of pitches and specific locations of the strike zone, so we of course need to take that into account. We aim to predict the exit velocity and launch angle of a batted ball, which are then used in calculating the expected weighted on-base average (wOBA) of the play.

3.3 Preprocessing Steps

We started by downloading all of the 2024 Statcast data available using pybaseball. We filter out games that did not take place in the regular season, games that had a differential of 6 or more runs, pitches from pitchers that had less than 3 appearances, and rows that appear to have Statcast misreads. These misreads can come in the form of not picking up the pitch type, incorrectly reading the velocity or spin rate, incorrectly reading the batted ball results, etc. We encode the categorical data, normalize the numerical data, and establish that the target values are the exit velocity and launch angle batted ball metrics which are used to get the expected wOBA.

Chapter 4

Model Selection

4.1 Model Consideration

For this project, we considered a handful of machine learning models before deciding on an XGBoost and a Recurrent Neural Network (RNN). We had thought about Random Forests and SVMs, but decided that Random Forests likely would not have the predictive power of an XGBoost and SVMs may have had a tough time handling our big dataset with many continuous features. When considering what type of neural networks would be effective, we knew an RNN would do a great job at capturing the effects of pitch sequencing, which is crucial to the outcome of an at-bat. Simpler neural regressors like Multi-Layer Perceptrons (MLPs) wouldn't be able to as effectively capture the sequence dependencies and patterns in our data. The dynamics of pitcher-hitter matchups change with every pitch thrown, and we wanted to ensure that we had a model that could handle this.

4.2 Final Model Selection

For our final model selection, we chose XGBoost and an RNN to tackle different aspects of predicting batted ball outcomes. Both models predict expected wOBA, which is calculated using launch angle and exit velocity, using static pitch characteristics and historical hitter data. The RNN, as stated above, incorporated pitch sequences along with the profiles of the pitches. The batted ball predictions were measured using Root Mean Squared Error and Mean Absolute Error. For the work going forward, we may make a prediction for a range of expected wOBA and make it a classification problem. If the actual batted ball falls in the range that we decide, we will consider it a correct prediction, and incorrect otherwise.

We compared the RNN model against a simple MLP regressor as a baseline comparison. The two models were tasked with predicting the same variable, and their outcomes were measured and compared using Root Mean Squared Error and Mean Absolute Error. After training both models for the same number of epochs, the RNN outperformed the MLP. As stated previously, this could be due to the RNN capturing long-term pitch sequence dependencies.

Chapter 5

Model Development and Training

5.1 Architecture, Configuration, and Training Process

The first step in the approach was to take the available pitch characteristic features that are described in the following section in order to create distinct clusters for each pitch type. We want to have a designation for various types of fastballs, sliders, changeups, etc. For example, one pitcher may have a fastball with elite velocity but not a super high spin rate or spin efficiency, where as a separate pitcher may have average velocity on his fastball, but with a different spin axis and higher spin rate that can cause it to have more ride when thrown up in the zone. Doing this step allowed us to assign a cluster label by pitch type to pitches with similar profiles.

The full dataset was split into training and testing datasets, with 80% of the data used for training and 20% of the data testing. The data was randomly shuffled, and in the case of the RNN, pitch sequences were kept intact. For the RNN, the training dataset was split further into 80% training and 20% validation, for final splits of 64% training, 16% validation, and 20% testing. To reduce overfitting, the XGBoost model employed L1 and L2 regularization, and the RNN employed layer normalization.

5.2 Hyperparameter Tuning

Both the XGBoost and RNN were fine-tuned using a grid search. The grid search for the XGBoost tried different values for max depth, learning rate, number of estimators, sub sampling ratio, subsample ratio of columns when constructing each tree, L1 regularization term weight, and L2 regularization term weight. The RNN tried different values for learning rate, RNN model (vanilla RNN, GRU, LTSM), number of RNN layers, and RNN hidden layer size.

Chapter 6

Evaluation and Comparison

As mentioned above, we reference the RMSE and MAE of the models to get a sense of performance, and for the XGBoost, we generate a feature importance plot to help identify which pitch characteristics and batter metrics contribute the most to the predictions. For the training speed, the XGBoost trains moderately quick, depending on the hyperparameter tuning with Grid Search. The RNN is more intensive with its sequential modeling capabilities, leading to slower training time, but benefits from the wealth of data that we have available.