

Predicting Baseball Pitch Outcomes with Multiclass Classification

Mark Rubin

Adviser: Dr. Xiaoyan Li

Abstract

This paper explores the idea of how model interpretability of multiclass classification models can be leveraged as a means of gaining insight into how pitch and game context features factor into individual pitch outcomes for specific batters. Isolating areas of improvement for professional baseball players is a notoriously hard and manual task as there are many features that contribute to every pitch. The research begins by building various multiclass classification models including SVM, Random Forest and a Neural Network to classify pitch outcomes as a ball, called strike, swinging strike or hit, and then uses permutation importance to evaluate how particular features affect the batter's outcome. Ultimately, the project concludes with a case study in how the models may be used for hypothesizing a player's weaknesses, suggesting a novel application of machine learning to Major League Baseball training.

1. Introduction

1.1. Problem Background & Context

Hitting a baseball is widely renowned as one of the hardest tasks in all of sports. As a batter, there is a small ball hurtling towards you at over 90 miles per hour from only 60 feet and 6 inches away. This leaves the batter less than half a second to decide whether to swing while still needing the time to initiate the swing and hopefully angle that swing to make contact with the ball. This does not even account for the fact the pitcher can place many different types of spin on the ball to deceive the batter making the task all the more difficult.

Every pitch, defined as "the delivery of the ball to the batter by the pitcher" [6], has four possible outcomes. The first is referred to as a *ball*; this is a pitch for which the batter does not swing and

that is outside the strike zone, a term used to classify "the area over home plate from the midpoint between a batter's shoulders and the top of the uniform pants – when the batter is in his stance and prepared to swing at a pitched ball – and a point just below the kneecap" [12] also denoted by Zones 1-9 in Appendix Figure 7, but may be adjusted by an umpire's discretion. The next possible outcome is a *called strike*, this is a pitch the umpire deems to be in the zone, but the batter did not swing at. Similarly, a *swinging strike* is any pitch where a batter attempts to swing but misses the ball regardless of the pitch's location. Finally, if the batter attempts to swing and their bat touches the ball this is called *hit into play*, combining both foul and fair balls for the sake of simplicity in this project, this is the best possible outcome for the batter.

1.2. Motivation & Gap

From the batter's perspective, there are many different factors that would go into their decision of whether to swing and even more factors that impact their ability to put the ball into play if they decide to swing. For instance, the location of the pitch, the number of strikes and balls in the at-bat (commonly referred to as the count), or some deceptive characteristics of the pitch may all contribute to the decision of whether to swing, and the ultimate effectiveness of these deceptive characteristics typically determine whether the swing will be successful or not.

While it is clear that these general attributes impact what classification (ball, called strike, swinging strike or hit into play) every pitch ends up in, it is not well understood *how* each feature plays a role in the batter's mind. Moreover, this idea is even less understood for individual players who may have good or bad tendencies that lead to different results. For instance, consider Yankees' Right-Fielder Aaron Judge and Dodgers' First-Baseman Freddie Freeman seeing an identical high and outside curveball with two balls and one strike. One player may have a more aggressive approach and be more likely to swing whereas the other may not. Furthermore, even if they both swing, certain features of this pitch may be less favorable to one of these players making them less likely to hit the ball than the other. Imagine a perfect world in which both of these players had a complete understanding of their strengths and weaknesses, they would then be able to improve

upon these weaknesses, leading to better performance, more championships and larger contracts. Additionally, consider the problem from a pitcher's point of view. If they had a perfect understanding of their opponent's weaknesses, they would be able to target these weaknesses, again leading to better performance. The issue for many players is identifying what causes them to swing at bad pitches, watch good pitches and not swing, or swing and miss on a perfect pitch.

1.3. Goal & Innovation

The goal of this project is to develop multiclass classification models for individual players using Major League Baseball (MLB) pitch data to predict the outcomes of pitches as one of the four classes discussed above. Then, by interpreting the importance of features in these models, we can obtain greater insight into the mind of a batter, which could be useful for that batter in their training by quantifying the impact of a pitch's feature on their performance.

2. Related Work

2.1. Related Work - Whiteside et al.

To begin an analysis of what pitch features individual batters may struggle with, it is important to first look at the problem through the opposite lens, what features contribute to a pitcher's success and how these features may be quantified. This idea is explored in Whiteside et al.'s paper "Ball Speed and Release Consistency Predict Pitching Success in Major League Baseball" which proposes a multiple regression model for identifying the characteristics of pitchers that contribute to their effectiveness on the mound [16]. By analyzing 76,000 pitches, they ventured to examine how pitch selection, speed, movement and location as well as variability in speed, movement and location impact a pitcher's *fielding independent pitching (FIP)* metric, a statistic designed to isolate a pitcher's success from the rest of their team. Ultimately, the research found that three factors explain 22% of the variance in pitcher's FIP: maximizing pitch speed, having a consistent release location and varying speed across pitches.

While this work is valuable in attempting to explain pitcher success, it lacks two pieces of

important information for answering the fundamental question of this project. First, it assumes a general batter; by assuming homogeneity across batters, we lose valuable information that can be used for analyzing pitcher-batter match-ups, an idea that will be explored in Doo and Kim's paper below. Furthermore, while this project explains the success of pitchers at large, two of the three important variables pertain to the variance of speed and release location which are not usable characteristics in predicting individual pitches since you cannot calculate the variability of these metrics on an individual pitch level. Therefore, this paper seeks to expand upon Whiteside et al.'s findings by evaluating the features that most strongly impact a pitch's outcome on an individual batter and individual pitch level.

2.2. Related Work - Doo and Kim

Expanding upon the previously discussed paper, Doo and Kim's research article "Modeling the probability of a batter/pitcher matchup event: A Bayesian approach" utilizes data from 2008 to 2017 from the Korea Baseball Organization (KBO) to model the potential outcomes of a pitcher and batter matchup, training models based on individual pitchers' and batters' histories [17]. The paper proposes a four-variable Bayesian hierarchical log5 model to develop a probability distribution over the likelihood of a batter reaching base as the outcome of an at bat. The four variables in the model are summary statistics to describe the batter, pitcher, league averages and the pitching team's defensive abilities.

This research develops the foundation for the ability to train models on individual players, removing the batter homogeneity assumption from Whiteside et al.'s paper [16]. However, this study solely focuses on the outcome of an at bat which is typically a collection of pitches rather than each individual pitch itself. Furthermore, the state space of outcomes considered are treated as a Bernoulli random variable in which a batter either reaches base or does not, and the inputs to the model are all based on summary statistics rather than individual pitch feature metrics. This removes much of the granularity that would be useful for a batter in attempting to improve upon their weaknesses and focuses more on developing a predictive signal for coaches to make better

decisions.

2.3. Related Work - Mould

Transitioning to a stronger focus on the batter's point of view, Joshua Mould's "Quantifying Hitter Plate Discipline in Major League Baseball" develops a framework for evaluating a hitter's *plate discipline*, a term used to define "the ability of a hitter to select the correct pitches to swing [at]" [18]. In his research, Mould aggregated data from the 2019-20 MLB season on every pitch as well as summary statistics information on both pitchers and hitters. He theorized that with this data, for every pitch you can determine the game theoretically optimal decision for a batter of whether to swing or not using an xgboost classifier and then use that to develop a metric that evaluates how closely a batter's plate discipline aligns with the optimal decision. Ultimately, this led to the creation of a new metric which Mould coined EAGLE, Expected Additional runs Gained by Looking/Swinging Estimate.

This metric contributes a structure for analyzing player's swing decisions as a quantifiable skill. Despite this, the paper acknowledges "EAGLE does not measure hitting ability, simply the ability to swing at the best pitches" [18]. This motivates a new line of research into how we can best predict the full outcome of a pitch (ie. ball, called strike, swinging strike, hit into play), not just the batter's decision to swing or not. With that type of classifier, we can have more granular information as to our expectation for the outcome of a pitch.

2.4. Gap in Existing Literature & Novel Idea

In this section, we recognize that a significant amount of research has been done to study the attributes that lead to both batter and pitcher success. For example, Mould's EAGLE metric attempts to explain batter success as a function of plate discipline assuming that swinging at the best pitches implies that a batter will achieve greater success [18]. On the other hand, Whiteside et al.'s paper focuses on explaining the variance in pitcher success by determining what characteristics of a pitcher yields the highest success against a general batter. Nonetheless, no existing literature attempts to leverage machine learning to codify a batter's decision making logic as a means of explaining

the variance in that batter's performance. This paper hypothesizes that by developing multiclass classification algorithms to predict the outcomes of pitches, we can obtain greater insight into the mind of a batter which can serve as an aid to the batter in achieving their fullest potential.

3. Approach

3.1. Theory Behind Approach

Now that we have established the gap in the existing literature, it is important to clearly define the approach this project will be taking and why this approach is reasonable in theory. To mathematically define the problem, there is a set of four possible outcomes for every pitch:

$$O = \{\text{Ball, Called Strike, Swinging Strike, Hit into Play}\}$$

where each of these terms were explained in the introduction. Moreover, each of these outcomes are only possible based on the batter's initial decision of whether to swing or not.

$$O_{\text{No Swing}} = \{\text{Ball, Called Strike}\} \text{ and } O_{\text{Swing}} = \{\text{Swinging Strike, Hit into Play}\}$$

Thus, to formulate the batter's problem like a machine learning task, they must first classify whether this is a good pitch to swing at based on all their available information on the pitch's features and game context, and then if they choose to take the action of swinging they will attempt to hit the ball into play. If they swing and hit the ball that would be a successful swing, and if they do not swing and it is called a ball that would also be a successful outcome; the other outcomes are generally unsuccessful or negative for the batter.

As a batter, there are no concrete rules that you perfectly follow on every pitch. Players are typically taught how they should generally behave and, through years of training, learn patterns in pitches and games that allow them to generalize how to behave as a more efficient agent. Ultimately, since there is a small, discrete set of potential outcomes, and the same inputs to the batter's decision

(pitch features and game context) can be quantified, we posit that *multiclass classification* is the appropriate machine learning task for this problem as it most closely imitates the problem’s natural form. Moreover, we propose training models for each player individually as we suppose that each player has slight differences in the patterns they recognize which may improve that model’s predictive accuracy. This will allow us to answer several questions that address the literary gaps regarding the development of a greater understanding of what features contribute to determining the outcome of a pitch and analyzing weaknesses in individual players. These questions, along with their significance will be discussed in the succeeding section.

3.2. Questions to Answer

3.2.1. Model Performance To derive any valuable information from the models, we first must establish how well each model performs on the classification task of predicting the outcome of a pitch. As seen in Appendix Table 2, the most frequent outcome is a ball, occurring roughly 37.9% of the time. This serves as the base accuracy for an extremely naive model which only predicts the most common class. Additionally, a decision tree classifier with minimal depth can be used as a secondary baseline model since it encodes many of the basic relationships across variables. As a foundation to the other questions, it is important that we identify at least one model with considerable improvement in *accuracy, precision and recall* over the baseline models.

3.2.2. Feature Importance Once we establish that the models make generally accurate predictions, we will look to answer the question of what features are most important for performing this classification task. This will address the shortcomings of Mould’s paper which identifies differences in batter’s swing decisions but lacks an explanation of the features that explain these differences [18].

3.2.3. Consistency in Feature Importance If we find through the previous question that feature importance can be identified, we will then look to see how these vary by player. In Whiteside et al.’s paper, they seeks to explain variation in pitcher success versus a general batter [16]. We suggest that this analysis can potentially be improved by relaxing the assumption of all batter’s valuing the same

features. This raises the question of whether feature importance is consistent across all players. If the answer to this question is that feature importance varies by player, this will be significant in two ways; it will motivate a potential further analysis of the problem discussed in Whiteside et al.'s research attempting to explain variance in pitcher success by now accounting for specific pitcher-batter match-ups, and second, it will enable us to potentially explain systematic weaknesses in a player's game that affect the probability distribution over possible outcomes on any given pitch.

3.2.4. Hypothesizing Batter Weaknesses All of the previous questions ultimately build up to the final question of can we hypothesize a batter's weaknesses based on an analysis of that player's models. As discussed in the Introduction, understanding your weaknesses as a baseball player is a very difficult task and would yield great value to identify and fix. If we assume that the model serves as a method of encoding the patterns in a player's performance, then we hypothesize that by interpreting the model, we can learn something about the player's pattern recognition skills. Thus, by examining a player's model closely in comparison to other player's models, we can potentially identify weaknesses that can attempt to be verified in future work.

3.3. Model Selection

To perform this classification task, we attempted to fit several models which each bear some particular advantage that we hypothesize can add value to the research.

3.3.1. Decision Tree Classifier The decision tree classifier is a machine learning model that effectively pairs binary classifiers into a tree format to split data and eventually classify each data point as some label or class based on a set of rules [4]. For the purpose of this project, the decision tree is intended to serve as a secondary baseline model since it encodes many of the most basic relationships that can classify pitches, and will allow us to better evaluate how the other classifiers perform over a very basic model. For instance, most pitches located outside of the strike zone will tend to be labeled as balls and this is not something of particular interest to us. The more interesting aspect is understanding if the other models can outperform the decision tree, if so, how much they will improve upon the model, and how they value features differently to achieve greater

performance. This model was trained using the `sklearn.tree DecisionTreeClassifier` package [2].

3.3.2. Random Forest A random forest classifier is quite similar to a decision tree except it is more complex in the sense that it trains a series of decision trees and ultimately predicts the majority vote among all the trees [8]. This model is worth investigating because it contains a very similar architecture to the decision tree, but is slightly more advanced. This can potentially provide insight as to whether more complex models can improve our predictive abilities over the baseline. This model was trained using the `sklearn.ensemble RandomForestClassifier` package [9].

3.3.3. Support Vector Machine (SVM) Transitioning to a different model architecture, SVM differs from the decision tree classifiers in that its primary objective is to find the optimal hyperplane to serve as the decision boundary in the classification task [13]. This allows the model to potentially generalize better than the decision trees which are both prone to potential overfitting and sparse feature importance. Even if the SVM model does not yield significantly greater performance than the previously discussed models, it still may be useful for obtaining a better understanding of feature importance which will elucidate potential player weaknesses. The `svm` package from `sklearn` was used for training this model [14].

3.3.4. Neural Network Moreover, an additional model that may hold predictive value is a neural network which best encapsulates complex relationships between variables by leveraging an architecture that is meant to most closely mimic the human brain [1]. This poses an advantage over the other models since the problem in its most natural form is being solved by a human learning pitch patterns, so the neural network's job here is to identify and replicate the patterns in the player's decisions. This model was developed using the `torch.nn` package from PyTorch with a ReLU activation function and 128 hidden layers in a feed forward network [15].

3.3.5. Hierarchical Model Finally, the hierarchical model is a custom model that chains three SVM models together to first classify whether a batter will swing at the pitch, and then conditional on that classification, if it predicts a swing it will predict if the batter makes contact or not, and if it predicts the batter will not swing, it predicts whether the pitch will be a ball or called strike. This was attempted as a potential optimization on the other models by forcing the classifier to "think"

like the batter which must decide whether to swing or not based on some set of features and then branches off into a new set of potential outcome states based on that decision.

3.4. Evaluation of Feature Importance

Once the models are trained, in addition to being evaluated for correctness, to achieve the goal of understanding how different features contribute to a batter's decision, *permutation importance* is used to evaluate the importance of a feature to the classifier. It is important to distinguish that permutation importance is not the weights of the feature in the model, but rather is a metric describing how important that feature is to the model forming its classification decision [5]. This is useful to the approach of the project because it provides insight into the mind of the model, thereby providing insight into the mind of the player who the model is trained on.

4. Implementation

4.1. Data Collection

To begin the project, the data was collected from the Python pybaseball package [7]. This package provides connectivity to data aggregated from popular baseball data sources such as Baseball Reference, Baseball Savant, FanGraphs and MLB Statcast. In the Statcast database, there is a table with information on every pitch from every game including information on the context of the game when that pitch was thrown, metrics on the pitch such as its release point, spin and velocity, and the outcome of that pitch such as whether the batter swung, if they made contact, and how hard the ball was hit. In a typical game there are over 250 pitches thrown and there are almost 2,500 games played per season; this means we expect to have over 600,000 data points per season indexed by pitch. Since this is a large sample size, this project only collects data from 2020 to 2024. Thus, the initial scraping of this data yielded 3.3 million rows (ie. pitches) with 120 columns (ie. features relevant to that pitch) [11].

4.2. Exploratory Data Analysis

To get an understanding of the dataset, the first element to analyze is what factors contribute to a pitch's outcome. Based on our previous discussion of how batters determine when to swing or not, and what contributes to a batter's successful or unsuccessful decision, we suggest that the features contributing to this outcome can be categorized into two styles of features: contextual features and pitch features. A breakdown of all of the features used in the analysis can be found in Appendix Table 3. Beginning with contextual features, these include information such as how many strikes or balls are in the count against the batter or how many outs there are. These would have an impact on the batter's approach, impacting how aggressive the batter might be in that moment. Figure 1 clearly demonstrates, for example, how the number of strikes in the batter's count impacts their decision to swing or not. Assuming all other factors are distributed the same regardless of the count, a batter is almost two times as likely to swing at a pitch with two strikes on them than if they have no strikes on them. This is an important relationship that should be expressed in the models' logic.

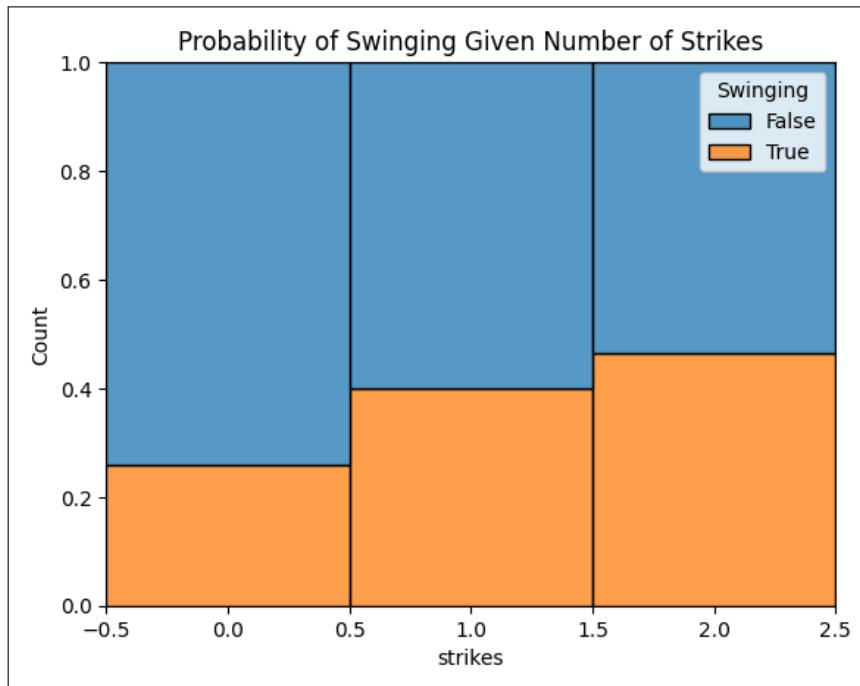


Figure 1: Probability of swinging given the number of strikes in the batter's count.

Transitioning into a focus on the pitch specific features, this would include information on the

type of pitch being thrown (whether that be a fastball, curveball, slider, etc.), the release position of the pitch, the pitch's ultimate location, speed, and a variety of other factors. In Figure 2, we attempt to develop an understanding of the distribution of pitch outcomes as a function of the pitch's location (referencing the strike zone chart in Appendix Figure 7). Intuitively, this is a very useful feature for classifying balls versus the other three classes, but even within the strike zone, we find that pitches in different zones bear quite different distributions over the pitch outcomes.

One other important finding to note in the exploratory data analysis was the vastly different distributions of features as a function of the pitch type. For instance, in the data set, we find that the average release speed of a 4-seam fastball is 93.9 MPH versus a curveball which has an average speed of 78.8 MPH. From a baseball standpoint, these pitches differ in features so vastly because they are attempting to deceive the batters in different ways. This means it is important to scale the features relative to the distribution of their pitch type since the speed of a "fast" curveball may be confused with that of a "slow" fastball without it being explicitly explained to the model.

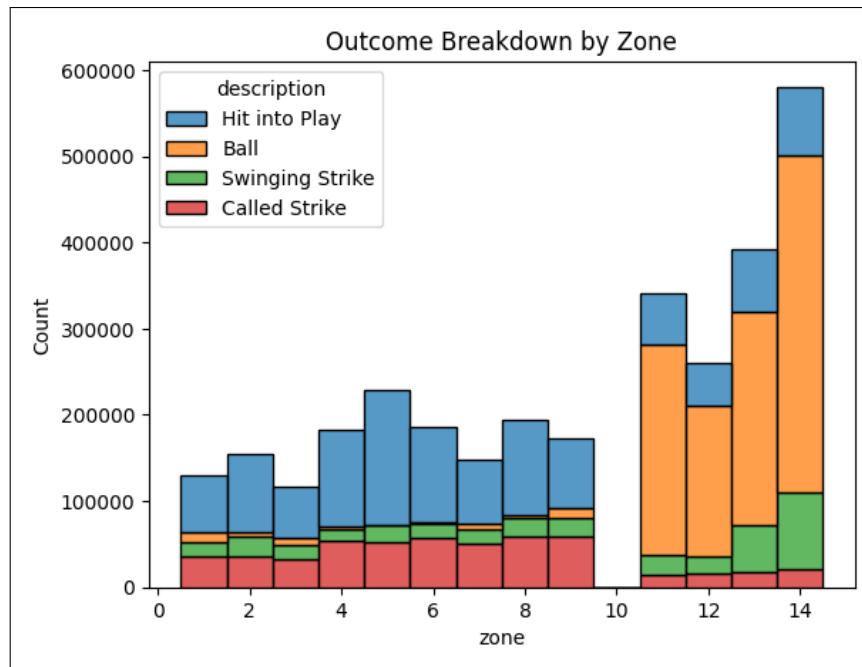


Figure 2: Pitch outcomes broken down by the pitch's zone (ie. location) as defined in Figure 6.

4.3. Data Preparation

In the data preparation phase there were a variety of steps necessary to make the data as clean as possible for training the models. The first step was narrowing the state space of outcomes. In the collected data, the 'description' feature provides 14 unique outcomes of every pitch. Since many of these could be categorized into the four outcome model described in this paper, a new outcome feature was developed mapping each of these fourteen outcomes to one of ball, called strike, swinging strike or hit into play. In addition to reducing the number of outcome states, many of the 3.3 million rows of the dataset were unneeded or could potentially lead to bias in the dataset so they were removed for a variety of reasons. The most notable such reason was player reduction. Since each model would be trained on an individual player, for the scope of this project, only about 50 players were needed. As a result, only the pitches to the fifty players with the most pitches seen between 2020 and 2024 remained in the dataset, and in the training stage, each player's model is only trained on a dataset with their pitches seen (about 10,000 pitches per player). Some additional steps for limiting the number of rows in the dataset included only keeping regular season games to remove the possibility of players acting more or less aggressive in the postseason influencing the models, and only keeping pitches with the 9 most common pitch types since there are very few instances of any other pitch type which could cause overfitting in the models.

Furthermore, for preparing the columns of the dataset, any column that related to the outcome of the pitch beyond the fours states described above was removed. This includes factors such as the exit velocity (a metric describing how hard the batter hit the ball) or the batter's launch angle which would contain look-ahead bias by providing information on the batter's choice to swing. In addition to removing columns, many columns required cleaning and encoding to be best prepared for model training. This included taking the absolute values of columns that depended on the pitcher's handedness (ie. the release point of the pitch is initially negative or positive depending on if the pitcher is left or right handed), and one-hot encoding various columns such as pitch-type, pitcher handedness, and pitch location. This ultimately reduced the dataset down to 530k rows and 69 columns with 21 unique features since many columns are encoded making them span several

columns.

4.4. Model Training and Evaluation

In the training and evaluation phase, the first step was preparing the data for each model to be trained. Since a model is trained for each player, the process for each individual player began with filtering the data-frame to only pitches that batter had seen. The data was then split 80:20 with a random split irrespective of time which relies on an assumption that a player's skill and tendencies remain constant over time. After splitting the data into a training and testing set, the continuous features, as noted in Appendix Table 3, were transformed into Z-scores relative to the mean and standard deviation of the training set to avoid any bias from the testing set, and to ensure all features were standardized for the model. Upon preparation of the dataset, one of the five classifiers would be trained on this dataset and evaluated through the testing set. This process was done separately for each player and model, meaning even for an individual player, their training and testing sets were different for each model.

After a model was trained, it was evaluated through accuracy, precision and recall metrics as well as analysis of the model's confusion matrix, from sklearn's metrics library, for the training and testing sets to ensure the model was not overfitting to the training data [10]. This information was then used for hyperparameter tuning which will be discussed in the next section, Model Revision and Refinement. Once a model was completely hyperparameter tuned, the interpretability of the model was evaluated through permutation importance, a package from sklearn which evaluates how important a feature is in classifying the data point to a class.

4.5. Model Revision and Refinement

Once the models were initially trained, the primary issue consistent among all the models was that no model was classifying pitches as swinging strikes which necessitated refinement to the models. There were two approaches to improving the models and fixing this issue: hyperparameter tuning and the addition of the hierarchical model. In the hyperparameter tuning stage, we utilized grid search to identify the best hyper-parameters for each model. For instance, in this phase we found

that the Random Forest model generalized best with a max depth of 10, and 4 as the minimum samples per leaf which prevented overfitting and allowed the model to improve its accuracy on the swinging strike class and accuracy on the testing set as a whole. All hyper-parameters for the models can be found in Appendix Table 4.

Beyond the hyperparameter tuning, one hypothesized potential improvement was the addition of the hierarchical model. The initial training phase consisted of only four models, but the hierarchical model was added as an attempt to potentially improve the model's classification by breaking the process into three steps of binary classifiers by predicting first whether the batter will swing at the pitch and then if the pitch will be a ball or called strike if the batter does not swing, or if it will be a swinging strike or hit if the batter does swing. Ultimately, this did not yield significant results over the other models in terms of the model's general performance, but did improve the model's classification of swinging strikes.

All code related to the implementation of this project can be found at <https://github.com/markruber920/TW-Final>.

5. Evaluation

5.1. Question 1: Model Performance

Before attempting to identify player weaknesses through the models, it is vital to first establish that the models are effective at the classification task of predicting a pitch's outcome. As discussed in the Approach section, the most basic model which only predicts the most frequent class, ball, would have an accuracy of almost 38%. This is far surpassed by all of the models which obtain accuracies of around 60% or higher as shown in Table 1. Furthermore, the decision tree was also discussed as a secondary baseline model since it could encode many of the basic relationships in the classification task that were evident in the exploratory data analysis. We still find that all of the other models have better accuracy and precision than this model, suggesting there are relationships that the decision tree cannot identify as strongly as the other models, implying the other models bear some advantage

over the baseline. We find that the Neural Network had the best accuracy and precision among the models, around 67.8% and 54.7% respectively, and the SVM model had the best recall around 56%.

Model	Avg Accuracy	Avg Precision	Avg Recall
Decision Tree	59.2% ($\pm 4.8\%$)	51.1% ($\pm 2.9\%$)	52.3% ($\pm 2.5\%$)
Random Forest	65.2% ($\pm 3\%$)	54.1% ($\pm 2\%$)	55% ($\pm 1.6\%$)
SVM	60.4% ($\pm 2.1\%$)	54.6% ($\pm 1.7\%$)	56% ($\pm 2\%$)
Neural Network	67.8% ($\pm 2.6\%$)	54.7% ($\pm 4.6\%$)	52.3% ($\pm 2.2\%$)
Hierarchical	60.4% ($\pm 2.7\%$)	51.3% ($\pm 1.8\%$)	51.2% ($\pm 1.9\%$)

Table 1: Average Performance on Classification Task by Model

This data suggests that the models are effective at this classification task, and are fairly accurate in predicting the outcomes of pitches in the test set based on the relationships identified in the training data.

5.2. Question 2: Feature Importance

Now that the performance of the models have been established, the next goal is to formulate an understanding of how the models make their decisions to ultimately identify player weaknesses. As expected, among all the models, the most important features for this classification task were the pitch location (particularly pitches outside the strike zone), and number of strikes in the count. This is useful as it serves as a sanity check in the process to confirm that the most important features in the classification task make intuitive sense. In addition to these features, vx0 and vz0, the velocity of the pitch in the x and z directions, were the next most commonly important features suggesting speed and movement of the pitch are quite important in predicting the pitch's outcome, another intuitive result that suggests the models accurately model player performance.

Furthermore, when analyzing feature importance across models, we find that the more accurate models placed greater importance on more features. For example, the decision tree classifier which was the worst performing model only had 8 features of non-negligible importance values whereas the best performing model, the neural network, considered almost every feature to have some level of importance. This can serve as an explanation as to why some models performed better than others since they could identify some value in every feature.

5.3. Question 3: Consistency in Feature Importance

After establishing that we can identify feature importance through the models, the next step towards our goal of identifying player weaknesses is establishing that feature importance varies by player. This is crucial because it invalidates the assumption in Whiteside et al. which assumes all batters can be treated homogeneously, and the same factors that affect one batter negatively will affect all batters equally negatively [16]. While permutation importance cannot directly tell us how the feature impacts a batter, we can develop an understanding of how important that feature is in classifying pitches for that batter. In Figure 3, we analyze the importance of the eight most commonly important features in five players' SVM models. In the graph, it is clear that the value of these features varies widely by player. For example, the model reflects former Diamondbacks and Cardinals First Baseman, Paul Goldschmidt, values the Zone 14 feature almost two times as much as Pirates Outfielder Bryan Reynolds in his classification decision. Combining these results with an understanding of this feature's role in baseball, we may hypothesize that Bryan Reynolds would have a high rate of swinging at pitches outside of the strike zone since the model identifies that this feature is not as important to him as it perhaps should be. Ultimately, this data suggests that variance exists in how important pitch and context features are to players, which opens the door for hypothesizing weaknesses based off of model interpretability.

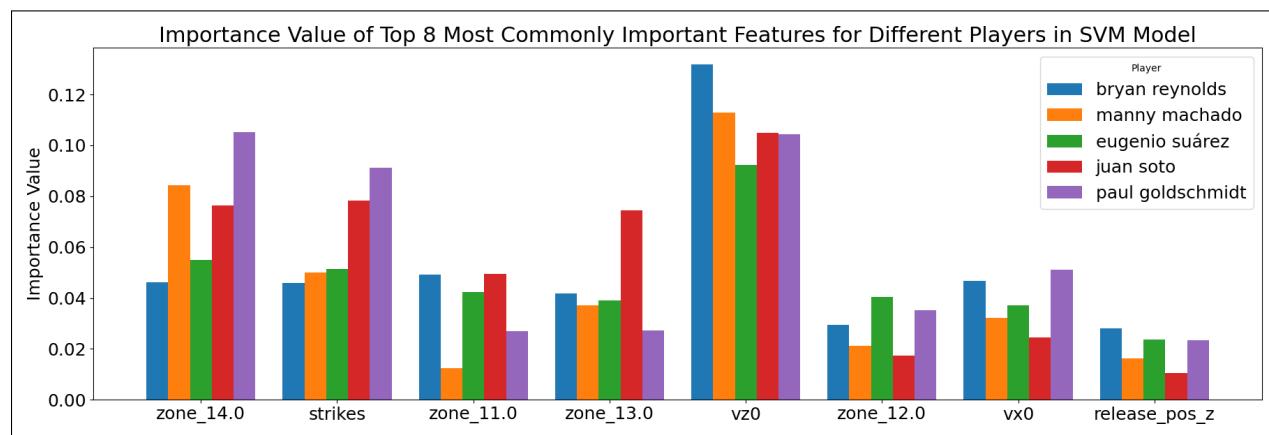


Figure 3: Importance value of top 8 most commonly important features for different players in their SVM models as computed through permutation importance.

5.4. Question 4: Hypothesizing Batter Weaknesses

Finally, to address the motivation of the project of identifying batter weaknesses through model evaluation, we will examine a case study of two players, Aaron Judge and Trea Turner, to understand how these models may be used for achieving this goal. In the first step of comparing these two players, we would look at the difference in how these two players value particular features. One striking distinction is how these two players value the zone 14 feature. As established in Appendix Figure 7, a pitch in location zone 14 is outside the strike zone and would not be considered a good pitch for a batter to swing at. However, it seems like this is a far more important factor in Aaron Judge's decision making than Trea Turner's, exemplified in Figure 4 where Judge values the feature almost two times as much as Turner does.

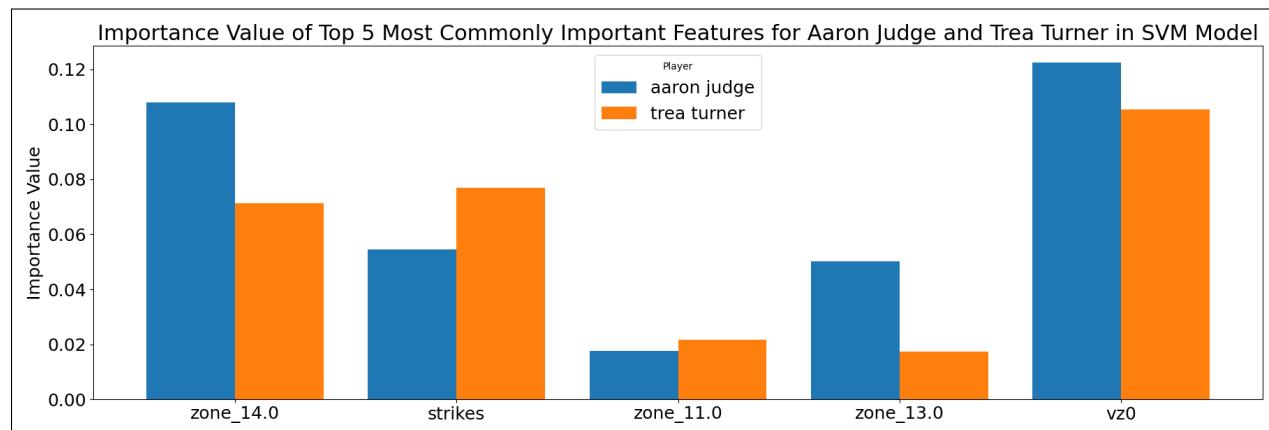


Figure 4: Comparison of importance values of several features for Aaron Judge and Trea Turner in their SVM models as computed through permutation importance.

After establishing this as a potential feature of interest for comparing these two players, the next step is examining the confusion matrices of these players, particularly when the pitch is in zone 14. In Figures 5 and 6, we compare the confusion matrices of Judge and Turner, and find that Turner's model predicts him to swing at far more pitches in zone 14 which is low and inside. This would suggest that he can benefit from training that focuses on improving his plate vision and discipline, particularly on low and inside pitches.

From a broader perspective, a player's weaknesses can be analyzed by comparing their importance of a feature relative to that of an average player. In the case of a categorical variable such as a

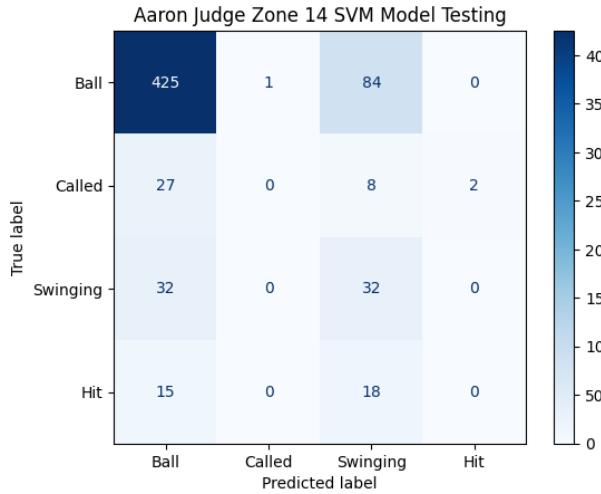


Figure 5: Aaron Judge’s confusion matrix for SVM model on pitches in Zone 14.

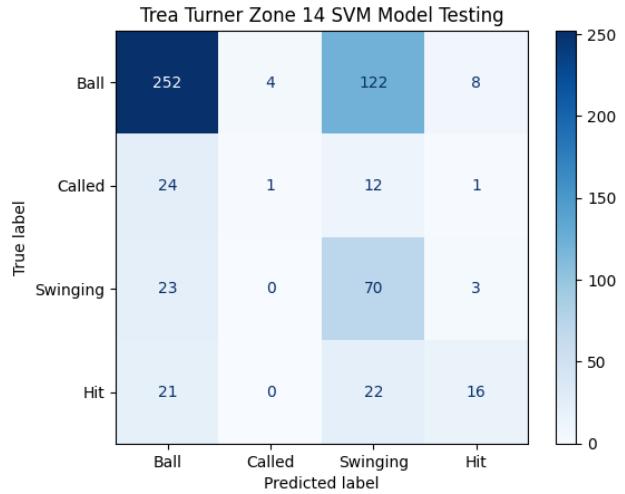


Figure 6: Trea Turner’s confusion matrix for SVM model on pitches in Zone 14.

specific part of the strike zone, the same process as described for Judge and Turner can be applied. In the case of a continuous variable (ie. $vz0$), the independent variable would need to be split into bins and confusion matrices would need to be analyzed for each bin to find potential player weaknesses.

6. Conclusion and Future Work

To conclude, this project poses a novel method of identifying player weaknesses through leveraging feature interpretability of multiclass classification models as a means of improving insights for baseball players’ training. In the beginning, the success of this project in achieving its goal was determined to be measured by the ability to identify at least one potential weakness in a player, proving that this is a function of the models. We attempted to achieve this through a four-step approach, first proving that the models were effective in this classification task, then finding that feature importance can be derived from these models, and then demonstrating that feature importance varies by player, ultimately finishing by attempting a case study in how the models can be used to hypothesize player weaknesses. In this process, we found that the best models were the neural network for accuracy (67.8%) and precision (54.7%), and SVM for recall (56%), significant improvements over the baseline decision tree model which achieved 59.2%, 51.1% and 52.3% accuracy, precision and recall respectively. Additionally, we found that feature

importance varies greatly across players, suggesting that we cannot assume a generalized batter which is a key assumption in much research in this area. Finally, we combined this knowledge to propose a weakness in Trea Turner's game discussed in greater detail in the *Evaluation, Question 4: Hypothesizing Batter Weaknesses* section, achieving the goal of the project. This provides a useful improvement over the status quo of identifying player weaknesses since most current methods are through manually watching a player over time, or using certain advanced metrics that struggle to isolate individual features as the root cause of a player's struggles.

While the project achieved great success, there are a myriad of future steps that can be done to build upon this research. The most important step would be to work on weakness verifiability. While the models can hypothesize a batter's weaknesses which is a useful step, it can only make suggestions based on the model's findings. In future work, by combining this evaluation with the current human-driven evaluation approaches, we can verify the correctness and accuracy of the model's suggestions. Similarly, I would work on building a more systematic approach to suggesting player weaknesses since there is still a human in the loop needing to look at the importance of features in these models and analyze the confusion matrices. Additionally, I would explore adding an additional set of features related to pitch sequencing which is a feature known to impact batter performance. Right now, the models evaluate each pitch independent from the other pitches in that at bat, but future work may explore how the other pitches in the at bat may correlate with a batter's performance on a given pitch. Ultimately, as this future work suggests, this work builds upon the existing field of literature by opening the door to explore many new methods for utilizing machine learning and feature interpretability to improve player performance.

References

- [1] "The complete guide to neural networks multinomial classification." [Online]. Available: <https://towardsdatascience.com/the-complete-guide-to-neural-networks-multinomial-classification-4fe88bde7839/>
- [2] "Decisiontreeclassifier." [Online]. Available: <https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeClassifier.html>
- [3] "Game-day-zones." [Online]. Available: https://www.researchgate.net/publication/358572353/figure/fig1/AS:1151690594107400@1651595843768/Game-Day-Zones-as-defined-by-Statcast-and-Baseball-Savant-The-strike-zone-is-presented_Q640.jpg
- [4] "Implementing decision tree classifiers with scikit-learn." [Online]. Available: <https://www.geeksforgeeks.org/machine-learning/building-and-implementing-decision-tree-classifiers-with-scikit-learn-a-comprehensive-guide/>

- [5] “Permutation importance.” [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.inspection.permutation_importance.html
- [6] “Pitch baseball dictionary.” [Online]. Available: <https://www.baseball-almanac.com/dictionary-term.php?term=pitch>
- [7] “pybaseball pypi.” [Online]. Available: <https://pypi.org/project/pybaseball/>
- [8] “Random forest classifier using scikit-learn.” [Online]. Available: <https://www.geeksforgeeks.org/dsa/random-forest-classifier-using-scikit-learn/>
- [9] “Randomforestclassifier.” [Online]. Available: <https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html>
- [10] “sklearn.metrics.” [Online]. Available: <https://scikit-learn.org/stable/api/sklearn.metrics.html>
- [11] “Statcast search csv documentation.” [Online]. Available: <https://baseballsavant.mlb.com/csv-docs>
- [12] “Strike zone | glossary | mlb.com.” [Online]. Available: <https://www.mlb.com/glossary/rules/strike-zone?msockid=266fd43e95ce6a8a1119c293943e6bea>
- [13] “Support vector machine (svm) algorithm.” [Online]. Available: <https://www.geeksforgeeks.org/machine-learning/support-vector-machine-algorithm/>
- [14] “Support vector machines.” [Online]. Available: <https://scikit-learn.org/stable/modules/svm.html>
- [15] “torch.nn.” [Online]. Available: <https://docs.pytorch.org/docs/stable/torch.html>
- [16] R. Z. David Whiteside, Douglas Martini and G. Goulet, “Ball speed and release consistency predict pitching success in major league baseball.” Journal of Strength and Conditioning, 2016, pp. 1787–1795. Available: https://journals.lww.com/nsca-jscr/fulltext/2016/07000/ball_speed_and_release_consistency_predict.1.aspx
- [17] W. Doo and H. Kim, “Modeling the probability of a batter/pitcher matchup event: A bayesian approach.” PLOS One, 2018. Available: <https://journals.plos.org/plosone/article?id=10.1371%2Fjournal.pone.0204874>
- [18] J. Mould, “Quantifying hitter plate discipline in major league baseball,” 2021. Available: <https://stat.cmu.edu/cmsac/conference/2021/assets/pdf/JoshuaMould.pdf>

7. Honor Code and Acknowledgments

I pledge my honor that this report represents my own work in accordance with University Honor Code. - *Mark Rubin*

I would also like to acknowledge my advisor Professor Xiaoyan Li, TA Andre Rubungo and my classmates for help in advising this project as well as the use of AI as an aid in the coding of this project as cited in the repository files in accordance with course, department and university policy.

8. Appendix

Outcome	Relative Frequency
Ball	37.9%
Hit into Play	35.6%
Called Strike	16.1%
Swinging Strike	10.4%

Table 2: Distribution of outcomes for MLB pitches 2020-2024 regular season games only containing players with 50 most pitches seen in that time frame.

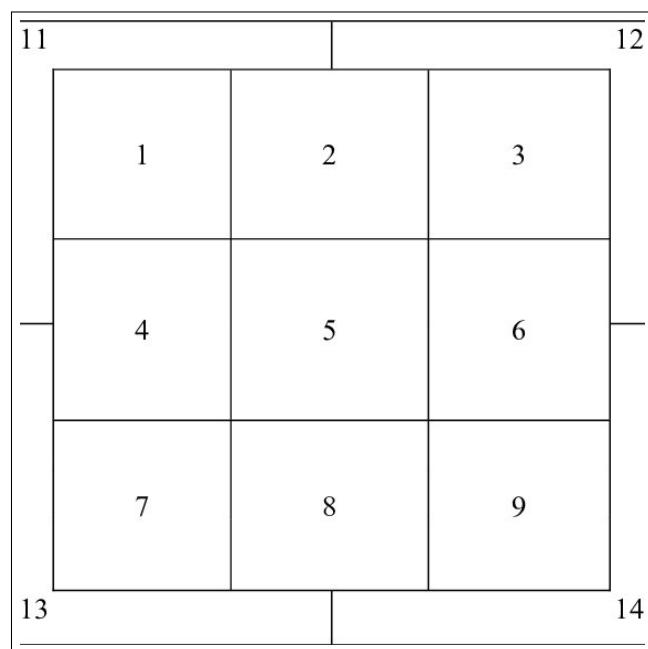


Figure 7: Layout of the Strike Zone. Zones 1-9 are in the zone and should be called a strike; Zones 11-14 are out of the zone and should be called a ball. [3]

Feature	Meaning	Category	Type
Zone	Location of the ball when it crosses the plate	Pitch	Discrete
Pitch Name	Name of the pitch type (ie. fastball, curveball)	Pitch	Discrete
Pitcher Side	Hand pitcher throws with	Pitch	Discrete
Balls	Pre-pitch number of balls in count	Context	Continuous
Strikes	Pre-pitch number of strikes in count	Context	Continuous
Outs When Up	Pre-pitch number of outs in inning	Context	Continuous
Release Speed	Out-of-hand pitch velocity	Pitch	Continuous
Release Spin Rate	Spin rate of pitch tracked by Statcast	Pitch	Continuous
Release Extension	Release extension of pitch in feet	Pitch	Continuous
Spin Axis	Spin Axis in the 2D X-Z plane in degrees. 180 represents a pure backspin fastball, 0 represents a pure topspin (12-6) curveball	Pitch	Continuous
Api break x arm	Horizontal break to pitcher's arm side in inches	Pitch	Continuous
Api break x batter in	Horizontal break to batter in inches	Pitch	Continuous
Api break z with gravity	Vertical break including gravity in inches	Pitch	Continuous
Release Pos x	Horizontal release position of the ball in feet	Pitch	Continuous
Release Pos z	Vertical release position of the ball in feet	Pitch	Continuous
Release Pos y	Release position of pitch measured in feet	Pitch	Continuous
vx0	X-dimension velocity of pitch (ft/s)	Pitch	Continuous
vy0	Y-dimension velocity of pitch (ft/s)	Pitch	Continuous
vz0	Z-dimension velocity of pitch (ft/s)	Pitch	Continuous
ax0	X-dimension acceleration of pitch (ft/s ²)	Pitch	Continuous
ay0	Y-dimension acceleration of pitch (ft/s ²)	Pitch	Continuous
az0	Z-dimension acceleration of pitch (ft/s ²)	Pitch	Continuous

Table 3: Feature dictionary explaining the meaning of each feature and categorizing each feature as a context or pitch feature. Meanings adapted from Statcast Search CSV Documentation [11]

Model	Hyperparameters
Decision Tree [2]	class weight: 'balanced', criterion: 'gini', max depth: 5, max features: None min samples leaf: 1, min samples split: 10, splitter: 'random'
Random Forest [9]	class weight: 'balanced', max depth: 10, max features: 'sqrt' min samples leaf: 4, min samples split: 10, n estimators: 500
SVM [14]	C: 10, class weight: 'balanced', degree: 2, gamma: 'auto', kernel: 'poly'
Neural Network [15]	Activation Function: ReLU, Hidden Layers: 128

Table 4: Model Hyperparameters