Keep your Eye on the Ball: Understanding the Pitches and Payoffs of the MLB

Ally Dalman
Princeton University
adalman@princeton.edu

Uri Schwartz
Princeton University
uris@princeton.edu

David Major
Princeton University
djmajor@princeton.edu

Ami Berman
Princeton University
azberman@princeton.edu

Abstract

The Major League Baseball Pitch Data 2015-2018 dataset on Kaggle gathers comprehensive data about all pitches thrown over the past four seasons. MLB teams have a clear interest in applying advanced machine learning algorithms to identify predictive features for fastballs, outs and ejections. Teams can use this information to prepare pitchers, batters and players in light of these results. We analyze the data set of 2.8M samples with Logistic Regression to predict these outcomes. We then use unsupervised learning methods to identify notable trends for future research. We find that the most predictive features for fastballs, outs and ejections are pitcher identity, location/previous at-bat outcome, and umpire identity, respectively. Our unsupervised learning results suggest that handedness of pitcher and batter, and game score may prove important in prediction of pitch type and outcome.

1 Introduction

In Major League Baseball (MLB) today, teams increasingly rely on sophisticated statistical analysis of the game for strategic decision-making. Machine learning and other statistical techniques allow teams to understand player and team trends better than human scouts. In this study, we attempt to identify several trends that may be useful to the manager of a professional baseball team.

This study analyzes a comprehensive data set of pitches, at-bats and ejections from MLB over the past four seasons. We use supervised learning techniques to answer specific questions that may be useful to the manager, such as predictions of pitch type, the outcome of a given at-bat, and player ejections. These problems correspond to the binary classification problem in machine learning. Understanding these trends will assist the manager to answer strategic questions including: Should the pitcher attack the strike-zone with a fastball, or should he deceive the batter with an off-speed pitch? Is the current batter likely to get out, or should a reliever be called in? We discuss important preprocessing steps and observations later in this paper. We also use unsupervised learning techniques to reveal the latent structure of the data, and identify opportunities for future research.

2 Related Work

Sabermetrics, the application of data analysis to baseball statistics, has emerged as a prominent field of study in recent years. Beneventano et al. (2012) found that baseball statistics were partially predictive of baseball runs production and prevention [5]. Other researchers used empirical data to find that stealing bases generally *did* improve team scores [6].

Researchers have applied machine learning techniques to baseball with some success. Hamilton et al. (2014) used machine learning tools on pitch data and found the resulting predictions to improve previous results moderately [8]. MIT researchers used linear regression on pitcher-specific data to accurately predict a pitcher's future performance [9]. We build on both these studies by testing whether machine learning can be used to yield valuable insights into fastball prediction, out prediction and ejection prediction.

3 Nature of Data

For our analysis, we use the MLB Pitch Data 2015-2018 dataset from Kaggle [10], which contains information about every pitch thrown in the past four seasons of baseball in the following files:

1. **pitches.csv:** Contains information about speed, location, and break angle of the pitch, the ball/strike count, number of outs, presence of runners on bases, and identity of the at-bat.
2. **atbats.csv:** Contains information about the score, the inning, pitcher/batter identities, and handedness.
3. **player_names.csv:** Links players to unique player IDs.
4. **ejections.csv:** Contains information about every umpire, score, inning during an ejection..
5. **games.csv:** Contains information about the weather, final score, teams, attendance, and stadium of the game.

4 Methods

4.1 Fastball Prediction

For pre-processing in this section, we merge pitches.csv and atbats.csv, such that each pitch has all information contained in atbats.csv and pitches.csv. In the interest of reducing computation time, we only focus on pitches from the past two seasons and randomly select a subset of this data. This leaves us with about 108,700 pitch samples. We then one-hot encode all categorical variables. We then scale all continuous variables appropriately. In terms of features, we test the classifiers on several different feature sets. When the number of features is large, we use the χ^2 statistic to select the 250 most predictive features (empirically determined to produce optimal results). We use the following different feature sets:

1. **Baseline Data:** Includes standard data on the game situation such as the score, the inning, the number of balls and strikes, and pitch type.
2. **Batters Data:** Includes baseline data, along with identity of the batter for each pitch.
3. **Pitchers Data:** Includes baseline data, along with identity of the pitcher for each pitch.
4. **Only Pitchers Data:** Includes only the identity of the pitcher for each pitch.
5. **All Data:** Includes the baseline data, along with identities of pitcher and batter.

Before running classification, we use K-fold cross validation to create train and testing sets. For prediction, we used Logistic Regression with ℓ_2 penalty (LR) using stochastic gradient descent.

4.2 Out Prediction

For pre-processing in this section, we merge pitches.csv and atbats.csv, such that each pitch contains all relevant information from atbats.csv and pitches.csv. In the interest of reducing computation time, we only focus on pitches from the past two seasons and randomly select a subset of this data. This leaves us with about 368,895 pitch samples, where each pitch results in an out or a batter on base. We then one-hot encode all categorical variables. We then scale all continuous variables appropriately.

In terms of features, we test the classifiers on several different feature sets. When the number of features is large, we use the χ^2 statistic to select the 100 most predictive features (empirically determined to produce optimal results). We use the following different feature sets:

1. **Baseline Data:** Includes standard data on the game situation such as the score, the inning, the number of balls and strikes, and the type of pitch thrown.
2. **Location Data:** Includes baseline data, along with information about speed, location, spin rate, break angle of pitch.
3. **Previous AB Data:** Includes baseline data, along with a constructed variable representing whether the previous at-bat resulted in an out.
4. **Only Batters Data:** Includes only the identity of the batter.
5. **Only Pitchers Data:** Includes only the identity of the pitcher.
6. **All Data:** Includes all features found in the previous input sets.

Before running classification, we use K-fold cross validation to create train and testing sets. For predicting the outcome of an at-bat, we used Logistic Regression with ℓ_2 penalty (LR) using stochastic gradient descent.

4.3 Ejection Prediction

For pre-processing in this section, we construct a variable for each game to represent the event of an ejection in the game. We randomly selected a subset of games (such that ejections. This leaves us with about 1,500 samples, where each game results in an ejection or not. We do not differentiate between the number of players ejected. For weather, we constructed a temperature variable (ex. 52°) and a sky variable (ex. "partly cloudy", "rainy") from an existing weather variable. We then one-hot encode all categorical variables, and scale all continuous variables appropriately.

We use the following different input feature sets:

1. **Umpire-Only Data:** Includes only identities of the umpires (406 features).
2. **Game Data:** Includes data about teams, attendance, weather, score, time, date, weather, and delay (76 features).
3. **Umpire and Game Data:** Includes data from the previous sets (482 features).

Given that ejections do not occur in the vast majority of games, the underlying class distribution was deeply imbalanced. We address this fact by under-sampling from the majority class to even out the data set. This reduced the number of samples to about 1,000, and improved performance of the model.

Before running classification, we use K-fold cross validation to create train and testing setsn [2]. For predicting the outcome of an ejection, we used Logistic Regression with ℓ_2 penalty (LR) using stochastic gradient descent [3].

4.4 Unsupervised Learning Methods

All methods are drawn from Sci-Kit Learn libraries. All parametrizations are the default unless specified otherwise.

1. *Principal Component Analysis* (PCA): using the solver selected by the default policy and 30 components, determined by a scree plot
2. *Factor Analysis* (FA): using expectation maximization and 3o components, determined by the same scree plot as PCA
3. *K-Means Clustering* (KM): using k-means++ initialization method and three clusters, determined by silhouette score

When running these methods on a merged version of pitches.csv and atbats.csv, we drop all columns related to pitch speed and location, and we one hot encoding various categorical variables, such as pitch-type, outcome, etc.

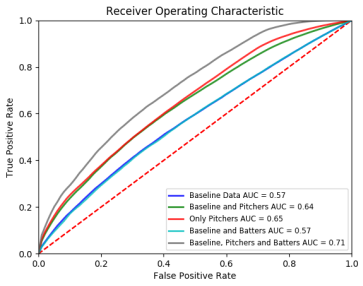

Figure 1: ROC Curve of the different data inputs for fastball prediction

Data	Accuracy	Precision	Recall	F-1	AUC
Baseline	0.55	0.56	0.74	0.64	0.57
Batters	0.55	0.57	0.73	0.64	0.57
Pitchers	0.65	0.63	0.84	0.72	0.65
Only Pitchers	0.65	0.61	0.95	0.74	0.65
All	0.65	0.63	0.83	0.72	0.71

Table 1: Evaluation Metrics for Fastball Prediction

4.5 Evaluation Metrics

We evaluate our methods using the following metrics. Accuracy is the proportion of correctly classified test inputs, precision is the proportion of true positives out of possitively classified samples, recall is the preportion of true positives out of true positives and false negatives, and f-1 is a weighted mean of precision and recall. Receiver Operating Characteristic (ROC) graphs the predictiioin threshold across the full range of possibilities, comparing the false positive rate (x-axis) againist the true positive rate (y-axis). Area Under Curve (AUC) measures the quality of the classifier and performs better as the value approaches 1. Silhouette score was used for K-Means clustering to measure the distinctness of each cluster and performs better as the value approaches 1.

5 Supervised Learning Results

5.1 Fastball Prediction

In this section, we attempt to predict whether a given pitch will be a fastball or not. We label all pitches as either positive (fastball, 1) or negative (not fastball, 0). Fastballs account for approximately 55% of all pitches thrown, yielding a relatively balanced class distribution. Using various combinations of input features, we run a Logistic Regression binary classifier along with K-fold cross-validation. This classifier outputs average accuracy, precision, recall, F-1 and AUC for each set of data inputs. The ROC curves for each input feature set appear in Figure 1. We also present the evaluation metrics in Table 1.

There are several observations to be made from this data. First, all evaluation metrics improve with the addition of pitcher identity to the baseline feature set. Accuracy, F-1 scores and AUC all improve about 8-10% with this modification. This suggests that pitcher identity is a strong indicator of the pitch type being a fastball. We also observe that recall is consistently higher than precision, which implies that false positives outnumber false negatives. We should note that the model may be biased toward predicting fastballs.

We test the importance of pitcher identity further by evaluating the classifier on the set of pitcher identities alone. Unexpectedly, this feature set outperforms the baseline and pitcher feature set in terms of F-1 score. The baseline, batter, and pitcher (comprehensive) feature set performs better in terms of AUC, but the Only Pitcher feature set shows higher recall and F-1. This shows that the performance of the Only Pitcher feature set is highly similar to that of the comprehensive feature

Likely		Unlikely
Santiago Casilla	Chase Anderson	Casey Fien
Richard Clayton	Sam Freeman	David Hale
Justin Wilson	Daniel Stumpf	Damien Magnifico
Matt Hall		

Table 2: Most and Least Predictive Pitchers for Fastballs

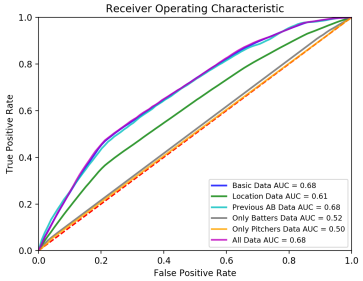

Figure 2: ROC Curve of the different data inputs for out prediction

set. We conclude from this that pitcher identity is a highly significant indicator of the pitch being a fastball. Alternatively, batter identity and the baseline features are relatively insignificant.

While this conclusion is reasonable, we caution against strongly concluding that pitcher identities are sufficient for fastball prediction. The comprehensive feature set may perform worse due to the vulnerability of Logistic Regression to multi-collinearity. This may be the case because certain pitchers identities may be collinear with other features, such as teams that score higher, handed-ness features, or inning.

Given that pitcher identity is a significant indicator of the likelihood of throwing a fastball, we follow up with a list of the most and least predictive pitchers of throwing a fastball. This list in Table 2 represents the ten pitchers most heavily associated with either category, as defined by the χ^2 statistic.

5.2 Out Prediction

In this section, we attempt to predict whether an at-bat will result in an out or not. We label all at-bats resulting in an out as positive (1), and all other outcomes as negative (0). Outs account for approximately 68% of all at-bats, yielding an imbalanced class distribution. We thus focus more heavily on precision, recall and F-1 scores.

Using various input feature sets, we run a Logistic Regression binary classifier along with K-fold cross-validation. This classifier outputs an average accuracy, precision, recall, F-1 and AUC for each set of data inputs. The ROC curves of each of the input datasets can be seen in Figure 2. Table 3 presents the evaluation metrics for each input feature set.

Data	Accuracy	Precision	Recall	F-1	AUC
Basic	0.71	0.71	0.97	0.82	0.68
Location	0.72	0.72	0.96	0.82	0.61
Previous AB	0.72	0.72	0.96	0.82	0.68
Only Batters	0.68	0.69	0.99	0.82	0.50
Only Pitchers	0.67	0.67	0.98	0.80	0.52
All	0.71	0.72	0.96	0.82	0.68

Table 3: Evaluation Metrics for Out Prediction

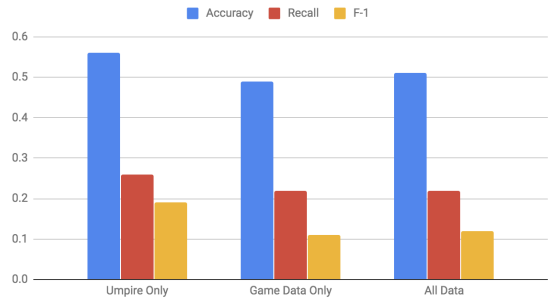

Figure 3: Accuracy, Recall, and F-1 for predicting ejections with the following data sets

There are several observations to be made from this data. First, we see that the addition of Location and Previous AB data slightly improves precision of the classifier. This implies that the location of the pitch and the previous at-bat data is moderately valuable in predicting the outcome of the current at-bat. Managers might consider what spots on the strike zone are more likely to yield an out when facing off against a particular batter, thereby giving his pitcher an edge in the match-up. Unlike fastball prediction, pitcher and batter identity are not as predictive for the outcome of an at-bat. In this case, the pitch type and game situation are the best indicators of the outcome of an at-bat. Managers should focus more on pitch type, and less on the identities of the players. Finally, we also observe that recall is consistently higher than precision, which implies that false positives outnumber false negatives. This implies that the model may be biased toward the prediction of an out for a given at-bat. This may be due to the imbalances in the class distribution toward positive samples. Nonetheless, our model is not blindly predicting outs as recall is not 100%. Thus, we are comfortable with the high number of positively classified samples.

5.3 Ejections Predictions

In this section, we attempt to predict whether a game will result in an ejection or not. We label all games resulting in ejections as positive (1), and all other outcomes as negative (0). Through undersampling, we balance out the class distribution. Using various combinations of input features, we run a Logistic Regression binary classifier along with K-fold cross-validation. This classifier outputs average accuracy, recall and F-1 for each set of data inputs as represented in Figure 3.

Overall, the model does not yield highly predictive results for any of the three data sets. While both the umpire-only set and the comprehensive data set yield accuracy scores above 50% (55.9 %and 50.6%, respectively), recall and F-1 scores are consistently poor.

It is likely that there is insufficient data to obtain conclusive results. By evening out the data through undersampling, the set was reduced to 1,500 game samples. We experimented with balancing the class distribution, as we found that the model generally defaults to prediction of no ejection. Empirically, we settled on an ejection/non-ejection outcome ratio of 2:3 (shown in Figure 3), which produced more varied predictions.

Model performance, however, does produce one interesting observation. Umpire-only data was far more predictive than game-only data. This suggests that the identity of the umpire in a game is one of the more significant indicators of an ejection during a game. General game data (such as the duration, time of day, weather, and length of game) does not appear to be significant for prediction. In Table 4, we also present a list of the umpires with the most and least rejections (based on the number of games they umpire total).

6 Extension: Unsupervised Learning

6.1 PCA Results

To determine the optimal number of components for PCA, we construct a scree plot to identify the number of components that preserves almost 100% of the total variance. As the figure below reveals a number of 20 components, we use 20 components for PCA. Table 5 displays the three highest and

Most Ejections	Least Ejections
Jeremie Rehak	Mike Estabrook
Bob Davidson	Shane Livensparger
John Hirschbeck	Paul Schrieber
Bill Welke	Ed Hickox

Table 4: Umpires with most and least ejections.

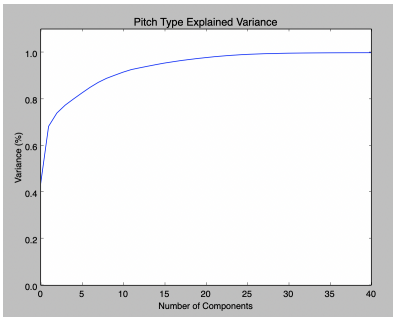

Figure 4: Scree Plot for PCA

three lowest feature coefficients of the first three principal components to identify which features are heavily weighted positively and negatively.

Component 1		Component 2		Component 3	
Coef.	Feature	Coef.	Feature	Coef.	Feature
0.743	p_score	0.742	b_score	0.832	b_count
0.669	b_score	0.023	outs	0.544	s_count
0.01	outs	0.010	on_2b	0.057	event_Walk
-0.004	p_throws_L	-0.006	s_count	-0.017	on_1b
-0.006	pitch_type_FT	-0.010	top_False	-0.020	event_Groundout
-0.007	top_true	-0.669	p_score	-0.047	type_B

Table 5: Coefficients of the PCA component variables

The first principal component corresponds to the direction of maximum variance in the data set, and the following components account for as much of the remaining variance as possible. Based on this, the first component most heavily emphasizes the scores of the pitcher and batter's team, which makes sense as these values vary greatly from game to game. The second component weights b_score very positively, but weights p_score very negatively, focusing on a potential negative correlation between batter score and pitcher score. The third component most heavily emphasizes b_count and s_count, the number of strikes and balls at the at-bat, which could suggest that these features tend to be positively correlated, where if there are more strikes in an at bat, then it is likely that there have also been some balls thrown and vice versa. These general trends are useful for understanding the directions of maximum variance in the data set, and how those variables are correlated. Future research should focus on correlations between the score of the two teams, and the number of balls and strikes.

6.2 Factor Analysis Results

For Factor Analysis, we also use 20 components as this number of components accounts for almost the total variance of the data set. Table 6 displays the three highest loadings for the first three factors so that we can better understand the characteristics of the groupings the model created.

The factor analysis yields interesting insights about features grouped together in the first three factors. The first factor highly weights right-handedness of the batter against a left-handed pitcher, while the second factor highly weights left-handedness of the batter against a left-handed pitcher.

Factor 1		Factor 2		Factor 3	
Loading	Feature	Loading	Feature	Loading	Feature
0.448	stand_R	0.364	p_throws_L	0.429	type_B
0.252	p_throws_L	0.207	stand_L	0.060	event_Walk
0.023	type_S	0.033	type_S	0.040	event_X

Table 6: Highest Loadings of Factors

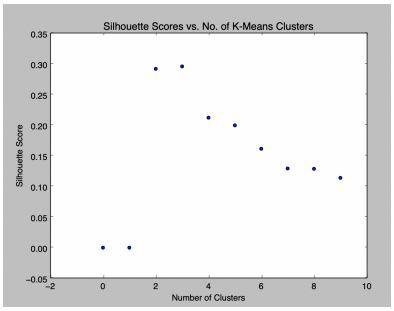

Figure 5: Silhouette Scores of K-Means Clustering

This reveals that the latent structure of the data may differ among left and right-handed batters against left-handed pitchers. Thus, game situations when the pitcher or batter is left or right-handed often perform very differently, and should be analyzed by the team manager differently. The third factor was also interesting, in that it highly weights type_B, the throw of a ball, and walks, thus giving the batter a walk.

We recommend that the observations from the second factor be explored further through supervised learning. This will allow managers to take advantage of the handed-ness of pitcher and batter.

6.3 K-Means Clustering Results

For K-Means clustering, we find the optimal number of clusters from 2 to 9 as determined by the silhouette score. Before clustering, we drop extra columns inning and pitch_num to prevent arbitrary clustering. In Figure 5, the silhouette score peaks with four clusters around 0.3, but this silhouette score is consistently poor. Analyzing the cluster centers using four clusters, the first cluster showed an average value of 6.196 for b_score and 2.041 for p_score, and the second cluster had an average value of 4.070 for p_score and 1.654 for b_score. This means the first two clusters emphasized situations when the pitcher's team was leading, and situation when the batter's team was leading. Even though silhouette score is average, the cluster centers seem to be based on more arbitrary variables, such as the score of the pitcher and batter's team, rather than specific situations. Since clustering,

7 Conclusion

This paper analyzes MLB pitch, at-bat, game and ejection data and uses machine learning classification libraries to predict useful outcomes, such as pitch type, pitch outcome, and player ejections. Our results demonstrate that the pitch data, filtered by certain features such as location, speed, and previous at-bat outcomes, was predictive of future outcomes. The ejection data was less predictive overall, although we did conclude that certain features, particularly umpire information, were more predictive than general game data. Overall, our results demonstrate that even with common machine learning libraries, managers can make accurate predictions to improve team effectiveness. For example, our results suggest that a player attempting to predict pitch type should emphasize the history of the given pitcher. Our results also suggest that a manager concerned about a potential ejection should only challenge the umpire based on the umpire's history.

Future studies may build on this analysis by turning to our unsupervised learning results for more detailed and accurate predictions. This may include inspecting clusters based on handedness and the performance of the pitcher's team. Researchers may also extend our analysis to additional features used in related work, such as stealing bases and player salaries.

References

[1] https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.get_dummies.html

[2] https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.KFold.html

[3] https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html

[4] Baumer, Benjamin, and Andrew Zimbalist. 2014. The Sabermetric Revolution: Assessing the Growth of Analytics in Baseball. University of Pennsylvania Press.

[5] Philip Beneventano, Paul D. Berger, and Bruce D. Weinberg. n.d. "Predicting Run Production and Run Prevention in Baseball: The Impact of Sabermetrics." International Journal of Business, Humanities and Technology.

[6] Demmink, Herman. 2010. "Value of Stealing Bases in Major League Baseball." Public Choice 142 (3): 497–505.

[7] Lewis, Michael. 2004. Moneyball: The Art of Winning an Unfair Game. 1st edition. New York, NY: W. W. Norton Company.

[8] Michael Hamilton, Phuong Hoang, Lori Layne, Joseph Murray, David Padget, Corey Stafford, and Hien Tran. Applying machine learning techniques to baseball pitch prediction. In Proceedings of the 3rd International Conference on Pattern Recognition Applications and Methods, pages 520–527. SCITEPRESS-Science and Technology Publications, Lda, 2014.

[9] Gartheeban, Ganeshapillai, and John Guttag. 2013. "A Data-Driven Method for in-Game Decision Making in MLB: When to Pull a Starting Pitcher." In Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining - KDD '13, 973. Chicago, Illinois, USA: ACM Press. https://doi.org/10.1145/2487575.2487660.

[10] "MLB Pitch Data 2015-2018." Accessed May 14, 2019. https://kaggle.com/pschale/mlb-pitch-data-20152018.