Predicting Base Hits in MLB using Naïve Bayes and Logistic Regression

Christopher Webb

MSc Data Science
City, University of London

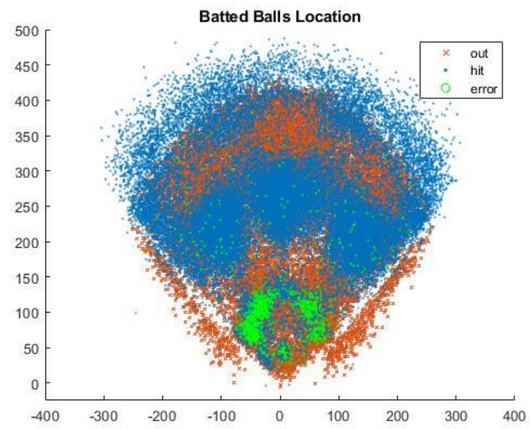
Motivation and Description of Problem

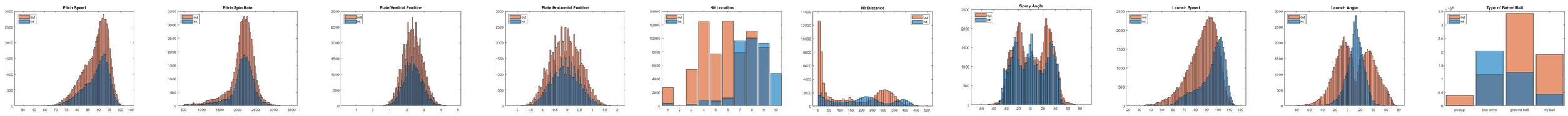
A hitter's best chance to help his team win in Major League Baseball (MLB) is by not getting an out when it is his turn at bat. This is most often achieved by getting a hit which has the added benefit of potentially advancing any runners on base as well, increasing the team's chances of scoring a run. There is a recent movement in baseball analytics to consider all batted balls in play as equivalent [1] when evaluating pitcher performance to remove ballpark size and team defense variables. Other studies have attempted to apply values to different types of batted balls [2]. This exploration of the data is a first step at attempting to better quantify batted balls in play by removing the outcome (hit, out, error), but focusing on some of the predictors that lead to a successful hit or a failed out. It's the hope this can be linked back to pitchers and pitch types that minimize the chances for a hit and maximize the chances for an out. Two machine learning approaches, Naïve Bayes (NB) and Logistic Regression (LR), were used to tackle this problem. This allowed a comparison of machine learning methodologies and results to take place on this dataset.

Initial Analysis of the Dataset

- Game logs from regular season MLB games from 2015 were mined for at bat information, including but not limited to: game state, pitch descriptors, and batted ball information.
- This is created by MLB Advance Media using their PITCHf/x system [3], that uses multiple high-speed cameras to capture pitch and batted ball data. This is uploaded nightly to MLB's api using json formatting.
- 130,472 at bat records with 64 predictors and one response variable were initially chosen.
- Dataset was simplified by removing 10 columns related to game information, 8 containing fielders present, 9 were included in other variables, and 3 further descriptors of the play, leaving 34 predictors.
- Mean and standard deviation were calculated for all numeric predictors as seen in table on the right.
- Error response was converted to out because MLB scorekeeper determined it should have been out if not for a fielder mistake and limited perceivable differences in the data from an out.
- A further 23,890 records were removed due to missing pitch or batted ball information.
- Despite these adjustments, the proportions of hits vs outs + errors remained consistent with all 2015 plate appearances and 2015 was the median season over the past ten years [4].
- Data was normalized.

	Mean			Standard Deviation			Skew		
	Hit	Out	Error	Hit	Out	Error	Hit	Out	Error
Pitch Speed	88.2	88.8	88.6	6.0	5.8	5.9	-0.7	-0.8	-0.8
Pitch Spin Rate	2102.8	2114.7	2101.8	358.0	367.0	384.7	-1.4	-1.3	-1.1
Plate Location X	-0.1	-0.1	-0.1	0.6	0.6	0.6	0.0	0.0	-0.1
Plate Location Y	2.3	2.4	2.3	0.5	0.6	0.5	0.1	0.6	0.1
Hit Distance	53.1	202.7	150.7	76.1	127.0	134.3	2.3	0.1	0.3
Spray Angle	0.9	2.9	-5.0	28.8	26.7	28.1	0.0	0.0	0.4
Launch Speed	93.4	85.8	85.6	12.5	13.1	14.7	-1.2	-0.8	-0.7
Launch Angle	-5.0	0.9	2.9	28.1	26.8	26.7	0.4	0.0	0.0
Hit Location X	6.5	11.2	-6.0	126.1	81.7	61.1	0.0	0.1	0.4
Hit Location Y	251.3	166.9	108.1	80.4	96.9	55.3	0.2	0.7	2.1





Logistic Regression

- LR is a supervised learning task that generally is used for predicting binary classifications, although it can be used for more than two classes. This is often called Multinomial Logistic Regression [5].
- Like Linear Regression, LR fits data on a curve but is an S shaped from 0 to 1 that illustrates the probability a given set of predictors is a specific class.
- LR uses maximum likelihood to determine probabilities between classes.

Pros

- LR is easy to implement and there are few parameters to optimize.
- Can be used with both continuous and discrete predictors.

Cons

- Due to its simplicity, LR can be not as accurate as some other Machine Learning techniques.
- LR does not solve non-linear problems.
- LR is susceptible to overfitting. And for successful models, it's important to identify important features to avoid irrelevant or highly correlated features.

Naïve Bayes

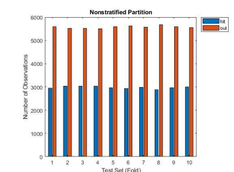
- NB is a supervised learning task that calculates conditional probabilities for each predictor, given its class. Mathematically, this can be articulated with the formula: $P(y|X) = \frac{P(X|y) * P(y)}{P(X)}$
- NB also assumes all predictors are independent given their class.
- Classification is determined by taking the class with the highest probability calculated from the product of each predictor for each class.

Pros

- NB is rather simple and easily understood but despite this has reasonable classification success.
- No prior knowledge is needed; NB can calculate prior probabilities from the dataset.
- Works just as well with multiple classes as it does with Boolean.

Cons

- There can be a loss of accuracy due to the assumption of conditional independence due to the existence of dependencies among predictors.



Hypothesis

- The literature shows some analysis on this topic has been performed [6], but it is unknown if LR and NB have been utilized. The expectation is they will perform slightly similar, but with LR being slightly more accurate based on other classifying problems [7].
- NB will probably have less decrease in accuracy when moving from training data to test data than LR due to its tendency to not be overfit [8].

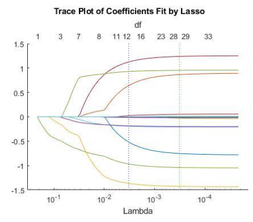
Training and Evaluation Methods

- 20% of dataset was held out for final model evaluation.
- The remaining 80% was used for training, utilizing a 10-fold cross-validation methodology.
- Hyperparameters were adjusted during each model run and an average accuracy score against validation data was calculated over the ten folds to gauge the model's success or failure. $\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$
- Accuracy was chosen over other similar methods, like F1 Score, because it weighed positive ('hit') and negative ('out') errors equally, giving no preference between the two [9].

Parameter Choices and Experimental Results

Logistic Regression

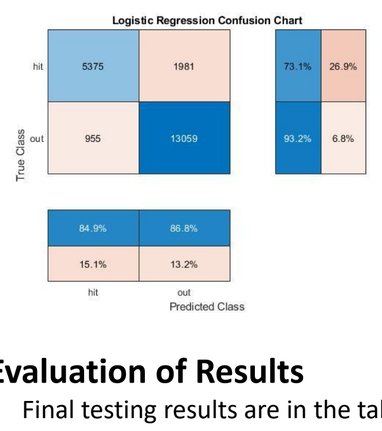
- An initial LR model was created with a binomial distribution as a baseline and the logit cost function. This had an average accuracy score of 86.3%.
- Lasso Regularization was used to help with feature selection to aid in removing unnecessary predictors. This was done with cross validation in place to hopefully avoid any key variables being removed by mistake. This led to 33 initial predictors being reduced to 12.
- A new model with same settings as before was run on the 12 predictors and accuracy decreased slightly to an average of 86.2% with 3 of the 10 folds scoring slightly better on the validation set with fewer predictors.
- Further exploration was done using Gradient Descent to find the best fit using log loss as our cost function when finding our theta values. This had the same accuracy, 86.2%, as before which was expected since the cost functions were the same despite the code syntax being different.
- Lambda regularization rate was introduced into the cost function to further explore the impact on the accuracy of our model, attempting to strike a balance between a low lambda value with the increased potential for overfitting and a high lamda value and underfitting. Results are in the table to the far right.

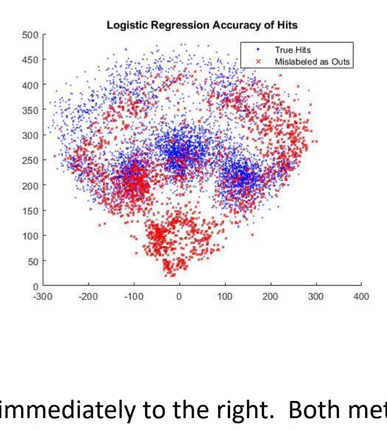


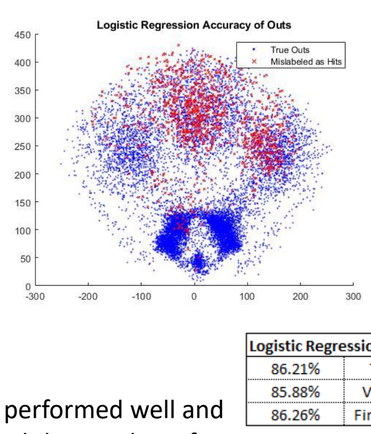
Naïve Bayes

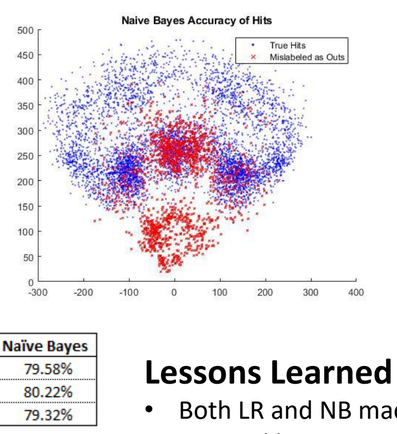
- The initial NB models had an average accuracy score of 72.3% and utilizing the feature selection from Lasso Regularization, an average accuracy score of 66.5% was found with the reduced predictors.
- For a fair comparison to LR, the reduced predictors were used in future models.
- Prior distribution was set matching the historical average, this further reduced accuracy to 66.0%.
- Prior hyperparameter was kept, and Kernel distributions were specified for all variables. Accuracy was greatly improved to 78.8%.
- Lastly, kernel was kept for all continuous variables, but categorical predictors were modeled with multivariate multinomial distribution instead. The prior constraint was also removed, leading to an accuracy score of 79.6%. This model was chosen as the best and was re-trained using the entire training dataset for further analysis and evaluation.

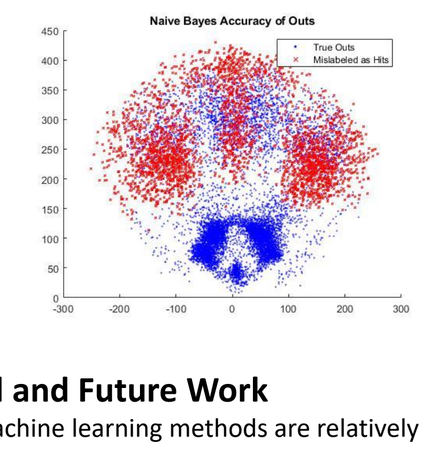
Logistic Regression			Naïve Bayes
Settings	Accuracy	Model	Accuracy	Settings
Binomial	86.3%	1	72.3%	Initial
After Feature Selection	86.2%	2	66.5%	After Feature Selection
Lambda = 1	83.0%	3	66.0%	Prior
Lambda = 10	83.2%	4	78.8%	Prior, Kernel
Lambda = 100	82.4%	5	79.6%	Kernel / MVMN
Lambda = 25	83.3%	6		

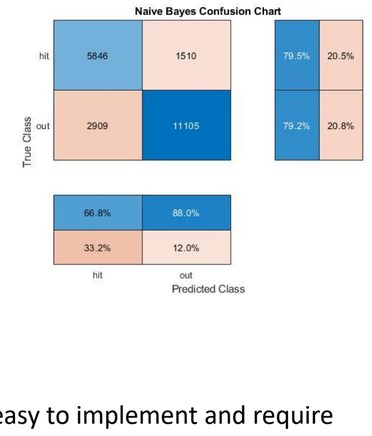








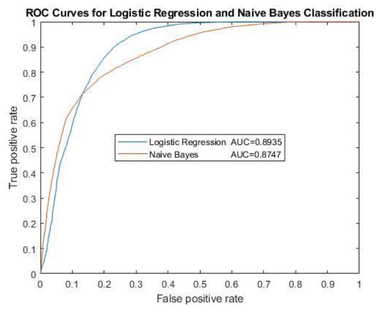




Evaluation of Results

Logistic Regression		Naïve Bayes
86.21%	Training	79.58%
85.88%	Validation	80.22%
86.26%	Final Testing	79.32%

- Final testing results are in the table immediately to the right. Both methods performed well and were consistent between training and final testing. The size of the dataset and the number of predictors probably helped contribute to not overfitting the data.
- The biggest gain in parameter tuning was using the Kernel density function in NB, but both methods had decent accuracy to start and did not require major tuning.
- The LR models were all created in under a couple of seconds and the longest NB model about 2 minutes. Despite the large dataset, this allowed many parameters to be tested in a short amount of time.
- The hypothesis was correct about LR being slightly more accurate than NB, but incorrect on how they would fare relative to training data.
- As illustrated in the ROC curve to the right, NB had a higher percentage of probabilities close to 0 or 1 than LR which might contribute to the similar AUC despite the differences to accuracy at the 0.5 threshold.
- The difference between the two models mainly came down to the false negative rate in the NB model. You can see in the classification error charts that both LR and NB do well at classifying batted balls as outs in the infield, but NB struggles to classify outfield balls correctly.
- NB assumes all features are independent which they rarely are. In this dataset, we have a series of events (pitch, batter, fielder) which are known to not be independent which could help explain the lower accuracy.



Lessons Learned and Future Work

- Both LR and NB machine learning methods are relatively easy to implement and require minimal hyperparameter tuning to get accurate results.
- The dataset studied required a considerable amount of pre-processing prior to reviewing these machine learning techniques. In the future, it might be wise to narrow the list of features before model creation to help minimize noise.
- Although not articulated in the analysis above, a brief look at predicting 'error' in addition to 'out' and 'hit' was done with NB. However, the initial model had very few predicted for this class. It could be worth exploring further if that was an anomaly or 'error' and 'out' classes are indistinguishable in the data as suspected.
- Binning continuous variables was not explored nor was varying the training and validation set size to explore how this impacts accuracy. These could be rewarding exercises in the future.
- Similarly, this dataset lends itself to other machine learning algorithms, including Random Forests, which might prove interesting to explore further to see if accuracy could be improved by methods with higher sophistication. Could also try to break hits into further classes single, double, triple, or home run instead.
- For simplicity, accuracy was chosen to evaluate these models. Since the proportion of 'hit' versus 'out' are not the same (~35% / 65%), balanced accuracy may have been a better metric to use if it was determined the weight of these two classes should be equal regardless of frequency.

[1] Swartz, Philippa, et al. "The Quality of Pitches in Major League Baseball." The American Statistician, vol. 71, no. 2, 2017, pp. 148–54. Crossref, doi:10.1080/00031305.2016.1264313.
[2] Alceo, Pedro, and Roberto Henriques. "Sports Analytics: Maximizing Precision in Predicting MLB Base Hits." Proceedings of the 11th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management, 2019. Crossref, doi:10.5220/0008362201900201.
[3] "The Hardball Times Baseball Annual 2010." What the Heck Is PITCHf/x?, written by Mike Fast, ACTA Publications, 2009, pp. 1–6.
[4] "Major League Baseball Batting Year-by-Year Averages." Baseball-Reference.com, 2020, www.baseball-reference.com/leagues/MLB/bat.shtml.
[5] Bishop, Christopher. "Linear Models for Classification." Pattern Recognition and Machine Learning (Information Science and Statistics), Springer, 2006, pp. 179–224.
[6] Healey, Glenn. "Learning, Visualizing, and Assessing a Model for the Intrinsic Value of a Batted Ball." IEEE Access, vol. 5, 2017, pp. 13811–22. Crossref, doi:10.1109/access.2017.2728663.
[7] Pranckevičius, Tomas, and Virginijus Marcinkevičius. "Comparison of Naive Bayes, Random Forest, Decision Tree, Support Vector Machines, and Logistic Regression Classifiers for Text Reviews Classification." Baltic Journal of Modern Computing, vol. 5, no. 2, 2017. Crossref, doi:10.22364/bjmc.2017.5.2.05.
[8] Tsangaratos, Paraskevas, and Ioanna Ilia. "Comparison of a Logistic Regression and Naïve Bayes Classifier in Landslide Susceptibility Assessments: The Influence of Models Complexity and Training Dataset Size." CATENA, vol. 145, 2016, pp. 164–79. Crossref, doi:10.1016/j.catena.2016.06.004.
[9] Zheng, Alice. Evaluation Metrics. Culemborg-Netherlands, Netherlands, Van Duuren Media, 2015.