

Using Decision Trees for Baseball Run Prediction

Lance Kevin
lkevin@email.sc.edu

Ian McDowell
mcdoweli@email.sc.edu

Michael Templeton
mat10@email.sc.edu

Abstract

Plate discipline is a crucial way to evaluate a baseball team and player. Plate discipline is how well the batter is seeing pitches. A batter, for the most part, should swing at strikes and not swing at balls. We will look at plate discipline and how it can help the South Carolina Baseball team improve their team's batting discipline to have a higher chance at winning games. Our approach uses decision tree and random forest models as explainable models, with a basic neural network to compare. The highest test accuracy we achieved with the explainable models was 65.95% and the maximum test accuracy of the neural network was 68.28%.

1. Introduction

South Carolina baseball is looking to gain a better understanding of their team's batting approach and the effectiveness of their offense. All this in hopes of building a more successful team. This is where our project of modeling plate discipline comes in. Plate discipline is a hitter's ability to discern between pitches they should swing at versus ones they should take (not swing). When thinking about plate discipline, we first need to take into account the strike zone. Plate discipline tells whether a batter is consistently swinging at pitches that are in the strike zone and taking pitches that are not.

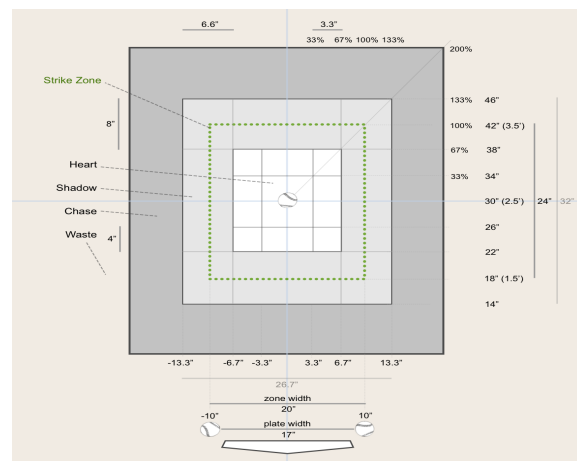


Figure 1: [Image of dimensions of different types of zones in baseball. This helps determine run value for batter by location. Image courtesy of Tangotiger.]

Evaluating plate discipline and learning how to improve it is important because a player with good plate discipline will generally generate more points for their team than someone with worse plate discipline. This metric can be used with Run-Value (the number of runs produced from a pitch based on the current situation) to evaluate and improve player performance, make lineup adjustments, further scouting on opponents, and more. We aim to use pitching data with a decision tree model to predict the run-value of a given pitch and gain an understanding of the model so the important factors to help players improve plate discipline can be discerned and improved.

2. Related Work

Due to the wealth of data available from both collegiate and professional baseball there has been a lot of public work done on analyzing this data,

so much in fact that there is a name for the field, “sabermetrics.”

A key related work comes from The Book: Playing the Percentages in Baseball by Tango Lichtman Dolphin and their blog. This blog article on swing/take, and run expectancy matrix has created a baseline for baseball run prediction. Their method is used by Major League Baseball (MLB) for the run value of players.

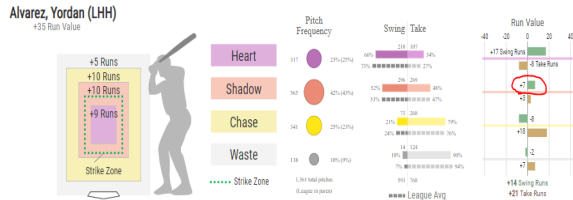


Figure 2: [Visual shows the ability of the project.

The last image of run value is what we will be working towards. *Image courtesy of MLB Statcast website.]*

An example of run value is: a batter is at the plate, there is 1 out, and a runner on second. The batter swings at a pitch that is clearly in the dirt. Because of this, the expected run would be -.4 runs. By swinging at this pitch he cost his team an expected -.4 runs.

	0-2	K:	1-2	J:	0-1	I:	2-2	H:	1-1	G:	0-0	F:	1-0	E:	2-1	D:	3-2	C:	2-0	B:	3-1	A:	3-0	Base Runners		
	0.42	0.44	0.47	0.48	0.50	0.51	0.55	0.55	0.59	0.61	0.67	0.74	0.76	0.80	0.84	0.87	0.89	0.90	0.96	0.98	1.03	1.07	1.15	1.25	1B ---	
	0.99	1.03	1.10	1.09	1.13	1.15	1.20	1.20	1.19	1.25	1.30	1.38	1.39	1.31	1.34	1.41	1.47	1.50	1.58	1.57	1.64	1.74	1.83	1.98	1B 2B ---	
	1.23	1.26	1.33	1.34	1.37	1.38	1.44	1.41	1.43	1.48	1.52	1.60	1.59	1.56	1.60	1.68	1.74	1.78	1.85	1.82	1.86	1.91	1.98	2.08	1B --- 3B	
	1.56	1.60	1.68	1.67	1.74	1.78	1.85	1.82	1.86	1.91	1.98	2.08	2.09	2.09	2.21	2.26	2.27	2.32	2.40	2.46	2.53	2.62	2.69	2.78	1B 2B 3B	
	1.77	1.81	1.91	1.85	1.95	1.98	2.03	1.99	1.97	2.09	2.11	2.19	2.18	2.08	2.09	2.21	2.26	2.27	2.32	2.40	2.46	2.53	2.62	2.69	1B 2B 3B	
	2.08	2.09	2.21	2.26	2.27	2.32	2.40	2.46	2.53	2.62	2.69	2.78	2.83	2.78	2.83	2.92	2.99	3.06	3.14	3.22	3.30	3.38	3.46	3.54	1B 2B 3B	
	0.21	0.22	0.24	0.25	0.26	0.27	0.30	0.30	0.32	0.35	0.38	0.43	0.43	0.41	0.43	0.48	0.49	0.51	0.53	0.57	0.57	0.60	0.64	0.69	0.77	1B ---
	0.55	0.58	0.63	0.63	0.67	0.69	0.73	0.72	0.71	0.79	0.79	0.85	0.85	0.76	0.80	0.86	0.86	0.91	0.93	1.00	1.00	1.06	1.10	1.21	1.38	1B 2B ---
	0.77	0.80	0.90	0.86	0.93	0.97	1.00	0.99	0.95	1.05	1.06	1.13	1.13	0.98	1.03	1.13	1.09	1.18	1.21	1.24	1.25	1.24	1.33	1.36	1.44	1B --- 3B
	1.10	1.17	1.28	1.22	1.32	1.37	1.42	1.39	1.33	1.48	1.48	1.54	1.54	1.27	1.34	1.43	1.46	1.53	1.57	1.68	1.68	1.78	1.82	1.91	2.08	1B 2B 3B
	1.27	1.34	1.43	1.46	1.53	1.57	1.68	1.68	1.78	1.82	1.91	2.08	2.09	2.09	2.21	2.26	2.27	2.32	2.40	2.46	2.53	2.62	2.69	2.78	2.83	1B 2B 3B
	0.06	0.07	0.09	0.09	0.10	0.10	0.12	0.12	0.12	0.14	0.16	0.18	0.18	0.14	0.16	0.19	0.19	0.20	0.21	0.22	0.23	0.24	0.25	0.26	0.27	1B ---
	0.14	0.16	0.19	0.18	0.21	0.23	0.26	0.24	0.24	0.25	0.30	0.32	0.37	0.31	0.34	0.38	0.38	0.41	0.45	0.45	0.49	0.49	0.57	0.61	0.68	1B 2B ---
	0.19	0.22	0.27	0.26	0.31	0.32	0.35	0.34	0.32	0.38	0.39	0.41	0.41	0.26	0.31	0.38	0.38	0.41	0.45	0.45	0.49	0.49	0.57	0.61	0.68	1B 2B ---
	0.23	0.26	0.32	0.31	0.34	0.36	0.39	0.38	0.37	0.43	0.45	0.49	0.49	0.31	0.37	0.42	0.44	0.47	0.50	0.55	0.54	0.63	0.67	0.73	1B --- 3B	
	0.35	0.39	0.48	0.44	0.53	0.57	0.62	0.57	0.54	0.68	0.69	0.75	0.75	0.46	0.54	0.62	0.67	0.74	0.76	0.80	0.89	0.95	1.05	1.18	1.38	1B 2B 3B
	0.46	0.50	0.60	0.56	0.67	0.70	0.76	0.73	0.70	0.84	0.85	0.93	0.93	0.55	0.65	0.74	0.79	0.86	0.91	0.98	1.05	1.14	1.24	1.38	1.60	1B 2B 3B

Figure 3: [Background of run value from Baseball Tango shows the expected run value of each situation of average batter. *Image courtesy of Tangotiger.]*

L:	0-2	K:	1-2	J:	0-1	I:	2-2	H:	1-1	G:	0-0	F:	1-0	E:	2-1	D:	3-2	C:	2-0	B:	3-1	A:	3-0	Base Runners		
	-0.09	-0.07	-0.04	-0.02	-0.01	0.00	0.04	0.04	0.08	0.10	0.16	0.23	0.23	-0.14	-0.10	-0.06	-0.03	-0.02	0.00	0.05	0.05	0.03	0.10	0.14	0.23	(0 out)
	-0.14	-0.10	-0.06	-0.03	-0.01	0.00	0.06	0.08	0.13	0.17	0.25	0.35	0.35	-0.18	-0.14	-0.09	-0.06	-0.03	-0.01	0.00	0.06	0.06	0.04	0.12	0.16	1B ---
	-0.16	-0.12	-0.05	-0.06	-0.02	0.00	0.05	0.05	0.03	0.10	0.14	0.23	0.23	-0.16	-0.12	-0.05	-0.06	-0.02	0.00	0.05	0.05	0.03	0.10	0.14	0.23	1B --- 2B
	-0.20	-0.16	-0.09	-0.09	-0.03	0.00	0.07	0.06	0.14	0.24	0.33	0.43	0.43	-0.15	-0.12	-0.05	-0.04	-0.01	0.00	0.06	0.03	0.05	0.09	0.14	0.21	1B 2B ---
	-0.15	-0.12	-0.05	-0.04	-0.01	0.00	0.06	0.03	0.05	0.09	0.14	0.24	0.24	-0.15	-0.12	-0.05	-0.04	-0.01	0.00	0.06	0.03	0.05	0.09	0.14	0.21	1B --- 3B
	-0.22	-0.18	-0.10	-0.11	-0.04	0.00	0.07	0.03	0.08	0.13	0.20	0.28	0.28	-0.19	-0.17	-0.06	-0.11	-0.04	0.00	0.03	0.02	-0.02	0.08	0.09	0.17	1B --- 3B
	-0.20	-0.17	-0.06	-0.12	-0.02	0.00	0.05	0.02	-0.01	0.11	0.14	0.21	0.21	-0.24	-0.23	-0.12	-0.06	-0.06	0.00	0.08	0.14	0.27	0.20	0.31	0.37	1B 2B 3B
	-0.24	-0.23	-0.12	-0.06	-0.06	0.00	0.08	0.14	0.27	0.20	0.31	0.37	0.37	-0.24	-0.23	-0.12	-0.06	-0.06	0.00	0.08	0.14	0.27	0.20	0.31	0.37	(1 out)
	-0.06	-0.05	-0.03	-0.02	-0.01	0.00	0.03	0.03	0.05	0.08	0.11	0.16	0.16	-0.12	-0.10	-0.05	-0.04	-0.02	0.00	0.04	0.04	0.07	0.11	0.16	0.24	1B ---
	-0.13	-0.11	-0.05	-0.06	-0.02	0.00	0.04	0.04	0.02	0.10	0.11	0.16	0.16	-0.13	-0.11	-0.05	-0.06	-0.02	0.00	0.04	0.03	0.02	0.10	0.11	0.16	1B ---
	-0.17	-0.13	-0.07	-0.07	-0.02	0.00	0.07	0.07	0.13	0.17	0.28	0.40	0.40	-0.17	-0.13	-0.07	-0.07	-0.02	0.00	0.07	0.07	0.13	0.17	0.28	0.40	1B 2B ---
	-0.19	-0.17	-0.06	-0.11	-0.04	0.00	0.03	0.02	-0.02	0.08	0.09	0.17	0.17	-0.19	-0.17	-0.06	-0.11	-0.04	0.00	0.03	0.02	-0.02	0.08	0.09	0.17	1B --- 3B
	-0.23	-0.18	-0.08	-0.12	-0.03	0.00	0.03	0.03	0.12	0.15	0.23	0.23	0.23	-0.23	-0.18	-0.08	-0.12	-0.03	0.00	0.03	0.03	0.12	0.15	0.23	0.23	1B --- 3B
	-0.27	-0.21	-0.09	-0.16	-0.05	0.00	0.05	0.02	-0.05	0.10	0.11	0.16	0.16	-0.27	-0.21	-0.09	-0.16	-0.05	0.00	0.05	0.02	-0.05	0.10	0.11	0.16	1B 2B 3B
	-0.30	-0.23	-0.14	-0.11	-0.04	0.00	0.11	0.12	0.21	0.25	0.34	0.38	0.38	-0.30	-0.23	-0.14	-0.11	-0.04	0.00	0.11	0.12	0.21	0.25	0.34	0.38	1B 2B 3B
	-0.04	-0.03	-0.02	-0.02	-0.01	0.00	0.00	0.02	0.01	0.02	0.04	0.05	0.08	-0.09	-0.07	-0.04	-0.04	-0.01	0.00	0.03	0.02	0.02	0.07	0.09	0.14	1B ---
	-0.09	-0.07	-0.04	-0.04	-0.01	0.00	0.03	0.02	0.02	0.06	0.07	0.09	0.09	-0.09	-0.07	-0.04	-0.04	-0.01	0.00	0.03	0.02	0.02	0.07	0.09	0.14	1B ---
	-0.13	-0.10	-0.05	-0.06	-0.02	0.00	0.05	0.05	0.05	0.05	0.13	0.17	0.28	-0.13	-0.10	-0.05	-0.06	-0.02	0.00	0.05	0.05	0.05	0.13	0.17	0.28	1B 2B ---
	-0.13	-0.10	-0.05	-0.06	-0.02	0.00	0.03	0.02	0.00	0.06	0.09	0.12	0.12	-0.13	-0.10	-0.05	-0.06	-0.02	0.00	0.03	0.02	0.00	0.06	0.09	0.12	1B --- 3B
	-0.19	-0.13	-0.07	-0.06	-0.03	0.00	0.06	0.04	0.05	0.13	0.17	0.23	0.23	-0.19	-0.13	-0.07	-0.06	-0.03	0.00	0.06	0.04	0.05	0.13	0.17	0.23	1B --- 3B
	-0.21	-0.18	-0.09	-0.13	-0.04	0.00	0.06	0.01	-0.03	0.12	0.12	0.19	0.19	-0.21	-0.18	-0.09	-0.13	-0.04	0.00	0.06	0.01	-0.03	0.12	0.12	0.19	1B 2B 3B
	-0.30	-0.22	-0.14	-0.09	-0.02	0.00	0.14	0.13	0.16	0.29	0.42	0.62	0.62	-0.30	-0.22	-0.14	-0.09	-0.02	0.00	0.14	0.13	0.16	0.29	0.42	0.62	1B 2B 3B

Figure 4: [Background of run value from Baseball

Tango shows the expected run value differences based on the start of an at-bat 0-0 from each situation of average batter. *Image courtesy of Tangotiger.]*

One of the most closely related works to our project is the development of QOP™ (quality of pitch) and the Griner Index. The Griner Index (GI) was originally developed by Jarvis Griner to develop a standard quantitative way to measure curveballs. The GI was developed using a multiple regression model using pitches labeled from 0-100 in terms of difficulty for a batter to hit. The formula to calculate the Greiner Index is:

$$GI = -2.51 * rise + 1.88 * breakpoint - 0.47 * knee distance + 0.51 * total break$$

Speed and horizontal break are added to the GI equation to calculate the QOP™. One benefit of QOP™ and the GI is that they rely only on measurements directly from the pitch and do not depend on the batter, like Earned Run Average, however, QOP™ and GI depended on the subjective opinion of one coach when generating their formulas.

3. Data

The data we used was provided by the University of South Carolina Baseball Team. The data the team provided is their Trackman data set. Trackman data tracks every pitch thrown with the statistics and outcomes attached to it. Data is from every pitch thrown in the University of South Carolina baseball stadium, Founders Park, as well as every other college team's stadium that has the trackman system set up. January 2018 all the way to May 2021 with a total of 1,040,373 data points. Examples of statistics found are release speed, spin rate, horizontal break for pitches and exit speed, launch angle, and direction for batting. Outcome

examples include the names of the pitcher and hitter, the type of pitch, and the outcome of the play. More information can be found about the Trackman system here {3}. Glossary for the columns of statistics and outcomes{4}

4. Methods

For our approach, we used the SciKit Learn Python library. Our first step, before giving the data to the model to train, was preprocessing the data. Preprocessing involved generating some new labels based on features of the data, plate zone, and pitch count, dropping all of the features that we would not use for training. Additionally, in preprocessing we used the data to generate the ground truth labels. To do this we looked at whether the player swung and what the pitch call was to generate four labels, any pitch labeled a strike was labeled 2, if the pitch was a ball and the player made a good decision not to swing it was labeled 3, but if they made a bad decision to swing it was labeled a 0. The last label made up very little of the data and consisted of intentional walks and players hit by pitches (all labeled 1) as the batter would not have the opportunity to swing at these pitches.

After preprocessing the data, we were left with 1,029,479 data points. With these, we trained three types of models: a decision tree model, a random forest model, and a neural network (multi-layer perceptron) model. Our goal was to use the decision tree and random forest models to give us a model with high explainability and interpretability. Meanwhile, the neural network gives a proper context for the explainability vs. accuracy tradeoff using the same data. For each model, we separated the data into training and testing sets. The training set consisted of 720,635 samples and the testing set consisted of 308,844 samples for a 70:30 ratio. For our decision tree model, we explored the hyperparameters of depth and min samples leaf. For our random forest model,

we explored the hyperparameters of the number of trees, max depth, and min samples leaf. For our neural network model, we explored the hyperparameters of hidden layer configurations (list of sizes for each hidden layer of that model), batch sizes, and learning rates.

5. Experiments

The first type of model we looked at were random decision tree classifiers. We initially tested a wide range of depths and the minimum number of samples at a leaf node to see what hyperparameters seemed to have the most effect on accuracy, and it seemed increasing depth was the largest driver to improve accuracy, and to a lesser extent a smaller minimum number of samples at a leaf node tended to improve accuracy as well. Despite running a multitude of tests the highest accuracy we were able to achieve was a mere 63.60% with a depth of 14 and a minimum of 1000 samples at a leaf node. The low accuracies achieved with the decision tree models were probably most influenced by our overall lack of experience with machine learning and the best way to go about tuning hyperparameters, and that decision trees are prone to overfitting to their data. An example of one of the decision trees can be seen in Figure 5.

The next model we looked at, Random Forest Classifiers, works to reduce the overfitting that is common in Random Decision Trees, and we were able to see a general increase in accuracy. When testing different hyperparameters for the random forest models we started out by randomly setting different values for the hyperparameters and it seems that increasing the depth was the most reliable way to increase the model's accuracy. The maximum score from Table 1 comes from a model with a depth of 30 which is the deepest model we trained throughout our experiments.

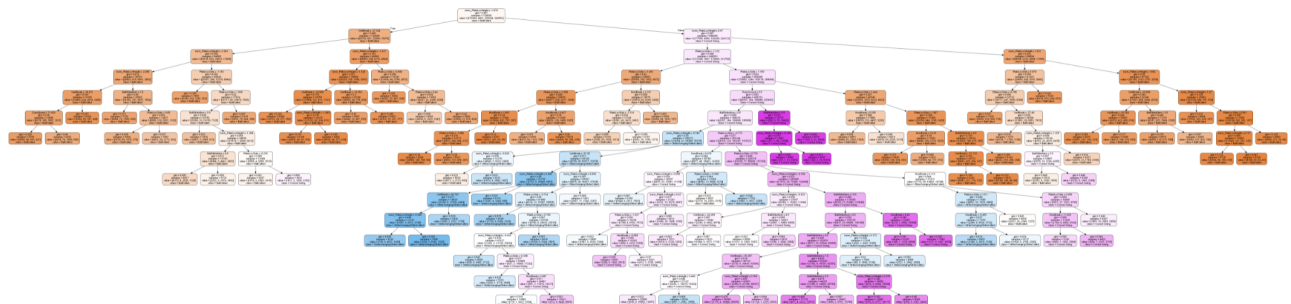


Figure 5: [The top decision tree model for accuracy and node count. Color Meanings: Orange: Ball Called, Pink: Correct Swing, Blue: Strike Swinging/Strike Called]

Table 1: [Accuracies for random forest model. The parameters not shown in the table were set to the default values of scikit-learn’s RandomForestClassifier.]

idx	number of trees	max depth	min samples leaf	score	idx	number of trees	max depth	min samples leaf	score	idx	number of trees	max depth	min samples leaf	score
0	10	5	1000	63.98%	46	10	5	1000	63.93%	96	10	14	1000	65.58%
1	80	5	1000	63.20%	49	100	5	1000	63.00%	97	100	14	1000	65.23%
2	110	5	1000	62.08%	50	200	5	1000	63.05%	98	200	14	1000	65.23%
3	380	5	1000	62.17%	51	300	5	1000	63.07%	100	300	14	1000	65.08%
4	10	5	1000	61.87%	52	10	5	1000	63.25%	100	10	14	1000	64.55%
5	80	5	1000	61.70%	53	40	5	1000	63.25%	101	100	14	1000	64.40%
6	110	5	1000	62.09%	54	110	5	1000	63.51%	101	200	14	1000	64.62%
7	380	5	1000	61.90%	55	100	5	1000	63.28%	102	300	14	1000	64.59%
8	10	5	1000	61.83%	56	10	5	1000	63.22%	104	10	14	1000	64.23%
9	80	5	1000	61.79%	57	40	5	1000	63.48%	105	100	14	1000	64.24%
10	110	5	1000	61.92%	58	110	5	1000	63.18%	106	200	14	1000	64.19%
11	380	5	1000	61.87%	59	100	5	1000	63.44%	107	300	14	1000	64.27%
12	10	5	1000	61.87%	60	10	5	1000	63.28%	108	10	14	1000	65.09%
13	80	5	1000	61.82%	61	40	5	1000	63.37%	109	100	14	1000	65.89%
14	110	5	1000	62.01%	62	110	5	1000	63.13%	110	200	14	1000	65.80%
15	380	5	1000	61.94%	63	100	5	1000	63.14%	111	300	14	1000	65.93%
16	10	5	1000	61.80%	64	10	5	1000	63.00%	112	10	14	1000	64.77%
17	80	5	1000	62.01%	65	40	5	1000	63.29%	113	100	14	1000	64.72%
18	110	5	1000	61.84%	66	110	5	1000	63.23%	114	200	14	1000	64.83%
19	380	5	1000	61.84%	67	100	5	1000	63.08%	115	300	14	1000	64.78%
20	10	5	1000	61.80%	68	10	5	1000	63.51%	116	10	14	1000	64.73%
21	80	5	1000	63.02%	69	40	5	1000	63.57%	117	100	14	1000	64.26%
22	110	5	1000	63.00%	70	110	5	1000	63.47%	118	200	14	1000	64.24%
23	380	5	1000	63.06%	71	100	5	1000	63.44%	119	300	14	1000	64.51%
24	10	5	1000	62.49%	72	10	5	1000	63.70%	120	10	14	1000	65.81%
25	80	5	1000	63.14%	73	100	10	1000	63.68%	121	100	14	1000	65.85%
26	110	5	1000	63.04%	74	200	10	1000	63.69%	122	200	14	1000	65.81%
27	380	5	1000	63.04%	75	100	10	1000	63.60%	123	300	14	1000	65.89%
28	10	5	1000	62.76%	76	10	10	1000	63.63%	124	10	14	1000	64.76%
29	80	5	1000	63.17%	77	100	10	1000	63.69%	125	100	14	1000	64.77%
30	110	5	1000	63.95%	78	200	10	1000	63.75%	126	200	14	1000	64.76%
31	380	5	1000	63.10%	79	100	10	1000	63.64%	127	300	14	1000	64.83%
32	10	5	1000	62.18%	80	10	10	1000	63.44%	128	10	14	1000	64.42%
33	80	5	1000	62.09%	81	100	10	1000	63.57%	129	100	14	1000	64.10%
34	110	5	1000	63.15%	82	200	10	1000	63.48%	130	200	14	1000	64.43%
35	380	5	1000	63.00%	83	100	10	1000	63.64%	131	300	14	1000	64.54%
36	10	5	1000	62.79%	84	10	12	1000	64.40%	132	10	14	1000	65.92%
37	80	5	1000	63.07%	85	100	12	1000	64.44%	133	100	14	1000	65.86%
38	110	5	1000	63.07%	86	200	12	1000	64.71%	134	200	14	1000	65.94%
39	380	5	1000	63.00%	87	100	12	1000	64.64%	135	300	14	1000	65.86%
40	10	5	1000	63.09%	88	10	12	1000	64.33%	136	10	14	1000	64.82%
41	80	5	1000	63.09%	89	100	12	1000	64.44%	137	100	14	1000	64.82%
42	110	5	1000	63.11%	90	200	12	1000	64.17%	138	200	14	1000	64.81%
43	380	5	1000	63.10%	91	100	12	1000	64.28%	139	300	14	1000	64.76%
44	10	5	1000	63.21%	92	10	12	1000	64.18%	140	10	14	1000	65.72%
45	80	5	1000	63.39%	93	100	12	1000	64.05%	141	100	14	1000	64.84%
46	110	5	1000	63.44%	94	200	12	1000	64.04%	142	200	14	1000	64.81%
47	380	5	1000	63.09%	95	100	12	1000	64.11%	143	300	14	1000	64.54%

In tests where we trained models on only data labeled with a specific pitch type, we were able to achieve higher accuracy for fastballs than the model with all the data and relatively high accuracy for curveballs.

Table 2: [Accuracies for random forest model on pitch-type separated data.]

Fastball	number of trees	max depth	min samples split	min samples leaf	max features	max leaf nodes	score
	100	None	100	100	9	100	65.70%
	100	None	100	100	9	500	66.19%
	100	None	100	100	9	None	62.27%
	100	None	1000	100	9	100	65.73%
	100	None	1000	100	9	500	66.03%
	100	None	1000	100	9	None	66.05%
Curveball	number of trees	max depth	min samples split	min samples leaf	max features	max leaf nodes	score
	100	None	100	100	9	100	65.61%
	100	None	100	100	9	500	65.90%
	100	None	100	100	9	None	62.72%
	100	None	1000	100	9	100	65.73%
	100	None	1000	100	9	500	66.03%
	100	None	1000	100	9	None	65.06%
ChangeUp	number of trees	max depth	min samples split	min samples leaf	max features	max leaf nodes	score
	100	None	100	100	9	100	63.01%
	100	None	100	100	9	500	63.10%
	100	None	100	100	9	None	63.03%
	100	None	1000	100	9	100	62.51%
	100	None	1000	100	9	500	62.51%
	100	None	1000	100	9	None	62.44%
Slider	number of trees	max depth	min samples split	min samples leaf	max features	max leaf nodes	score
	100	None	100	100	9	100	63.25%
	100	None	100	100	9	500	63.43%
	100	None	100	100	9	None	63.33%
	100	None	1000	100	9	100	63.01%
	100	None	1000	100	9	500	63.01%
	100	None	1000	100	9	None	63.03%

We only tested a small number of hyperparameters and used the same ones for all of the pitch types. One potential reason for the high accuracy of the fastball model may be due to the heterogeneous distribution of pitches in the dataset; there are 580,335 fastballs, 85,542 curveballs, 109,613 changeups, and 176,522 sliders. With a majority of the data being fastballs, this high accuracy is understandable. Surprisingly, the

curveball model was able to achieve the second-highest accuracy of the separated models despite curveballs comprising only 8.3% of our data. One potential cause for the low accuracy of the Slider and ChangeUp models is that they take more skill to throw than a Fastball, but a Curveball is considered a more advanced pitch. This may mean that Sliders and ChangeUps are less consistent than fastballs, which nearly everyone can throw, and curveballs, which take more skill and are attempted less frequently. Another major reason for the low accuracies of the Slider and ChangeUp models is that those pitches are used to set up a certain situation to try to entice the batter to swing at less favorable pitches. The Slider is used for this in particular as it is a slower, more curvy version of a fastball. Being used as setup pitches means that with each pitch the pitcher has a different goal and this will lead to a lot of variability in the total break, speed, and angle of these pitches.

For the neural network model, we started with a larger hyperparameter search space, but we quickly found out that a learning rate of 0.001 and batch sizes of 640 and 1280 seemed to produce the highest test accuracy so the majority of experiments were run with those parameters. Table 3 contains the hyperparameters and their respective test accuracies.

Table 3: [Accuracies for neural network model. Hidden layer configs are labeled in appendix]

Hidden Layer Config	Batch Size	Learning Rate	Test Accuracy	Hidden Layer Config	Batch Size	Learning Rate	Test Accuracy
A	640	0.001	67.62%	I	1280	0.001	68.16%
A	1280	0.001	66.99%	I	1280	0.003	68.28%
B	640	0.001	66.98%	I	1280	0.005	68.16%
B	1280	0.001	66.80%	I	2560	0.001	68.19%
C	640	0.001	67.88%	I	2560	0.003	68.08%
C	1280	0.001	67.91%	I	2560	0.005	68.16%
D	640	0.001	66.98%	I	5120	0.003	68.17%
D	1280	0.001	67.63%	I	5120	0.005	68.12%
E	640	0.001	67.61%	I	10240	0.003	67.95%
E	1280	0.001	67.49%	I	10240	0.005	68.00%
F	640	0.001	67.65%	I	20480	0.003	67.87%
F	1280	0.001	67.47%	I	20480	0.005	67.98%
G	640	0.001	67.02%	J	640	0.001	68.26%
H	640	0.001	68.18%	J	5120	0.003	68.06%
H	1280	0.001	68.08%	J	5120	0.005	67.94%
I	320	0.001	68.13%	J	10240	0.003	67.98%
I	320	0.003	68.06%	J	10240	0.005	67.41%
I	320	0.005	67.23%	J	20480	0.003	66.17%
I	640	0.001	68.16%	J	20480	0.005	68.04%
I	640	0.003	68.12%	K	640	0.001	67.67%
I	640	0.005	67.58%	K	1280	0.001	67.52%

As you can see, the highest test accuracy we were able to achieve with a standard neural network model was 68.28% with an average of 67.76%. While the highest is about 2% better than the best decision tree and random forest models we created, it is not as high as we were hoping to achieve (>70%). This shortcoming likely has multiple factors including how we preprocessed the data and our relative inexperience in designing neural networks. Despite this, with how similar the accuracy was across all neural network architectures (standard deviation of 0.48%), it is clear to see that there is an inherent improvement in accuracy when explainability is of lower significance.

6. Future Experiments and Improvements

Despite the wealth of data collected and the amount of statistical inference that is done about Baseball, it is still a complicated sport with many variables that will affect the outcome of a plate appearance and game overall. One of these complexities is that not every batter takes the same approach to batting. For example, one batter may be more conservative and be more willing to take a pitch in search of something they are more comfortable with, while another batter may be more nervous about taking pitches and will be more likely to swing at any pitch in the strike zone. Our experimental setup now would say that the first batter has worse plate discipline than the second, as they are taking pitches we consider “good,” despite the fact that the first batter may be more likely to make better contact and generate more runs than the second.

In our current model, we preprocessed the location of the pitch into 4 zones, (heart, shadow, chase, waste) based on the pitch's location relative to the plate. One potential area for improvement would be to further split these zones, or at least the heart and shadow zones, into inside, middle, and outside. In baseball, a pitch that is “inside” is closer to the batter in the strike zone while a pitch that is “outside” is on the far side of the plate from the batter. Whether a pitch is inside or outside has an effect on how the player must swing to make contact and therefore batters will perform differently with these different pitch types. One of the most obvious examples of this can be seen when looking at the performance of

batters against same-handed pitchers vs opposite-handed pitchers. It has been shown that batters will have higher batting averages when facing pitchers who throw from the opposite hand (Loftus). There are a few reasons for this, but it is mainly caused by the different angle a pitch will be approaching the batter from. For a pitcher of the same hand it will initially look like the pitch is coming directly at the batter, while a pitch from the other side may not look as intimidating. Additionally, when the pitcher and batter are of the same dominant hand a curveball will move away from the batter, while if they are opposite-handed a curveball will move inside.

Further splitting up the location zones for training is simply one suggestion, and if the discussion above is any indication, there is a lot of complexity and thought that can go into improving the model further. We took a little time to look at how separating the data by pitch type and training individual models for each pitch type would affect the accuracy and we observed relatively high accuracy for Fastballs and Curveballs and somewhat low accuracy for ChangeUps and Sliders, so further experiments could be done to determine why the accuracy did not increase for all of the pitch types and to improve. A future idea would be to take the decision trees and apply what we have into live scrimmages and see how the decisions impact the game.

7. Conclusion

Of the three types of models we tested, we were able to achieve the highest accuracy, 68.28%, using the Neural Network, while the highest accuracy achieved on the next best model type, the random forest classifier was 65.95% when both models were trained using all of the available data. In tests run with data separated by pitch type, we were able to achieve 66.19% accuracy for fastballs. One of the goals of our project was to have our model be explainable and to use the model to identify the most important characteristics of a pitch to determine its hit ability. We initially believed that a random decision tree would provide the explainability we desired, but it was quickly apparent that we would not be able to achieve adequate accuracy. We next looked at random forest classifiers to try to boost

accuracy while retaining explainability, however, in order to achieve the highest accuracies, the trees tended to have 2000 - 3000 nodes. While the most important decisions are in the top levels of the tree, trying to interpret that number of nodes will be difficult and time-consuming if even possible or useful. With the higher accuracies provided by the Neural Network and no real potential for explainability of the random forest models, it is clear that the sheer number of potential variables and situations to consider lends itself to a larger more complex model with potentially less explainability.

References


1. Tango Lichtman Dolphin.. (2019 September 23). *Statcast Lab: Swing/Take and a Primer on Run Value*. tangotiger.
<http://tangotiger.com/index.php/site/article/statcast-lab-swing-take-and-a-primer-on-run-value>
2. Tango Lichtman Dolphin. *Run Expectancy Matrix, 1950-2015 Actual Runs Scored, following each base/out state to end of inning*. tangotiger
<http://tangotiger.net/re24.html>
3. TrackMan Baseball.
<https://trackmanbaseball.com>
4. Woods, James. "Radar Measurement Glossary of Terms – Trackman." *TrackMan*, TrackMan, Mar. 2021,
<https://trackman.zendesk.com/hc/en-us/articles/115002776647-Radar-Measurement-Glossary-of-Terms>.
5. *Quality of Pitch*.
https://en.wikipedia.org/wiki/Quality_of_Pitch
6. Jason Wilson & Jarvis Greiner (2014) A Curveball Index: *Quantification of Breaking Balls for Pitchers*, CHANCE, 27:3, 34-40, DOI: [10.1080/09332480.2014.965629](https://doi.org/10.1080/09332480.2014.965629)
<https://www.tandfonline.com/doi/full/10.1080/09332480.2014.965629?scroll=top&needAccess=true>
7. Loftus, Matthew, "Significance of Handedness in Baseball" (2020). *Symposium on Undergraduate Research and Creative Expression (SOURCE)*. 882.
<https://scholar.valpo.edu/cus/882>
8. <http://tangotiger.net/strikezone/zone%20chart.png>
9. Tango Lichtman Dolphin.. (2018 November 23). *RE288: Run Expectancy by the 24 base-out states x 12 plate-count states, recursively*. tangotiger.
<http://tangotiger.com/index.php/site/comments/re288-run-expectancy-by-the-24-base-out-states-x-12-plate-count-states-recu>

Appendix

Hidden layer configuration labels for Table 3

A=500, **B**=1000, **C**=500;250, **D**=1000;500, **E**=500;250;125,
F=500;250;125;75, **G**=500;500;500;500, **H**=100;100;50;50;25;25,
I=100;100;100;50;50;50;25;25;25, **J**=100;90;80;70;60;50;40;30;20;10,
K=200;200;200;100;100;100;50;50;50;25;25;25

Full Experiment Tables

 validation_data

Trackman Data Labels

<https://trackman.zendesk.com/hc/en-us/articles/115002776647-Radar-Measurement-Glossary-of-Terms>

Python Notebooks

<https://github.com/ianm24/CSCE585-Project>

Data Used (Shared only with those given permission by UofSC Baseball Team)

[data.csv](#)

Presentation Video:

<https://youtu.be/eQ0i3PCtKxs>

Presentation Slides:

https://github.com/ianm24/CSCE585-Project/blob/main/supplemental_documents/presentation_slides.pdf