Final Report for Fall 2018 Baseball Project

for

Dr. Alan Dabney
Associate Professor
Texas A&M University
College Station, TX

by

Taylor Rao
Ian Scarff
STAT 485-500
December 7, 2018

Abstract

Two models were created using data from the A&M baseball team. One based off the pitches of various A&M games and one based off the batting of A&M players. For the pitcher model, data sets from 37 games were used to develop a user friendly algorithm that allowed for the importation data, setting of parameters, construction of a multinomial logistic model, and the ability to make predictions on the type of pitch a pitcher might throw next. Data sets of scrimmage games were used to test the model. An accuracy measurement across the whole test data would be calculated from these predictions.

For the batter model, the goal was to create a program that could take a few user inputted variables concerning a hypothetical opposing player's pitch, and output the probability of each result for each player. The data was split into training and testing data and various machine learning algorithms were trained on the training data and their accuracy was evaluated using the testing data. The random forest algorithm was decided on and tagged pitch type, relative vertical angle, plate height, and plate side were chosen as predictor variables. A program was then created to achieve the goal of predicted the likelihood of each outcome for each player given those predictor variable and the random forest model.

Introduction

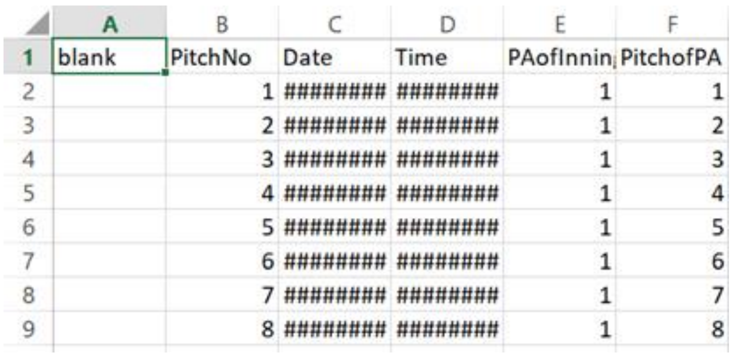
In the world of sports, it is vital to have as much information and insight as possible to improve one's odds in a game. In baseball, it is important for coaches to know how their players and their opponent's players will perform. For this reason we have developed two tools to aid coaches, a pitcher model and a batter model. The purpose of the first tool is to aid coaches and players during a game by predicting what a pitcher will throw next. The coach would communicate this information to the current batter so that they could anticipate how the ball will travel to them and adjust their swing accordingly. The purpose of the second tool is to assist coaches before a game to determine how they should order the batters based on the opponent pitchers. This would allow for greater insight on not only to gameplay, but also on the batters themselves. This tool would help identify where batters need improvement and how successful they are against different pitch types.

Methods and Procedures

Pitcher Model

For the pitcher model, the R package *nnet* was used to create a multinomial logistic model via neural networks to predict what type of pitch a pitcher would throw next. The predictor variables for the model were the pitch number of the game, the pitch number of the current inning, the number of pitches thrown for one batter, the pitcher's throwing arm, the batter's swinging arm, and the number of innings, outs, balls, and strikes. Lag variables for each pitch type were also included as predictors. The response variable for the model was the type of pitch thrown. Specific batter and pitchers from specific teams were not used in the development of this model so that the results could be generalized to the circumstances of any game.

37 separate games were used to create a training data set. To load each game into R, they had to be of a .csv type and the spreadsheet and to have the format shown below.



	A	B	C	D	E	F
1	blank	PitchNo	Date	Time	PAofInnin	PitchofPA
2		1	########	########	1	1
3		2	########	########	1	2
4		3	########	########	1	3
5		4	########	########	1	4
6		5	########	########	1	5
7		6	########	########	1	6
8		7	########	########	1	7
9		8	########	########	1	8

If the files were not of this format when loaded into R, every column entry would be shifted to the column to the left with the first column's data being overwritten. The addition of the blank column prevented this. In addition, some files had data entries separated by a blank row. These blank rows cause loading confusion in R if not removed.

For each game, the lag variables for pitch type were calculated by starting with a current pitch *m*. From *m*, the amount of occurrences for each pitch type would be calculated for the past *n* pitches. *m* and *n* would be decided upon before the creation of a model. After the lag variables were calculated, the individual games were combined into one large data frame. Undefined levels for pitch type, batter arm, and pitcher arm were removed from the data frame. The amount of data entries removed were insignificant when compared to the data frame as a whole. The functionality of the code used to generate the training data set from the individual 37 games allows the user to choose different *m* and *n* values to create multiple models with different training data sets without overriding the original data sets.

With a training data set now created, the *multinom* function from the *nnet* package was used to build the model. The function iterates over many possible models and chooses the best one. From there, the model was used to predict pitch types on different test data sets. The test data sets used were three individual Texas A&M scrimmage games. Lag variables using the same *m* and *n* values were calculated for the test data set to make predictions. From this prediction, a measurement of accuracy was calculate for the model. Accuracy was measured by comparing the predicted pitch with the true pitch value. If the pitched was correctly predicted, a value of 1 was assigned. Otherwise, a value of 0 was assigned. These values were then averaged to get an accuracy percentage. However, this measure of accuracy is not the most proper. Different initial value of *m* and *n* can change the accuracy measurement for the same game.

Batter Model

The first goal in creating the batter model was cleaning the data. The data was subset to include only rows where the batter was on A&M's team. Next, columns with a high number of null values were removed and then all rows which contained null values were removed. Variables that provided information that would only be available after the pitch call and variables that seemed irrelevant (such as date) were removed.

The second goal in creating the batter model was figuring out which algorithm to use. To do this, the column denoting the pitch call was made into the response variable. The cleaned dataset was then split into testing and training data using sklearn's *train_test_split* module and scaled using sklearn's *minmaxscaler* module. Then the following models were fitted to the data and their accuracy on the test dataset was measured:

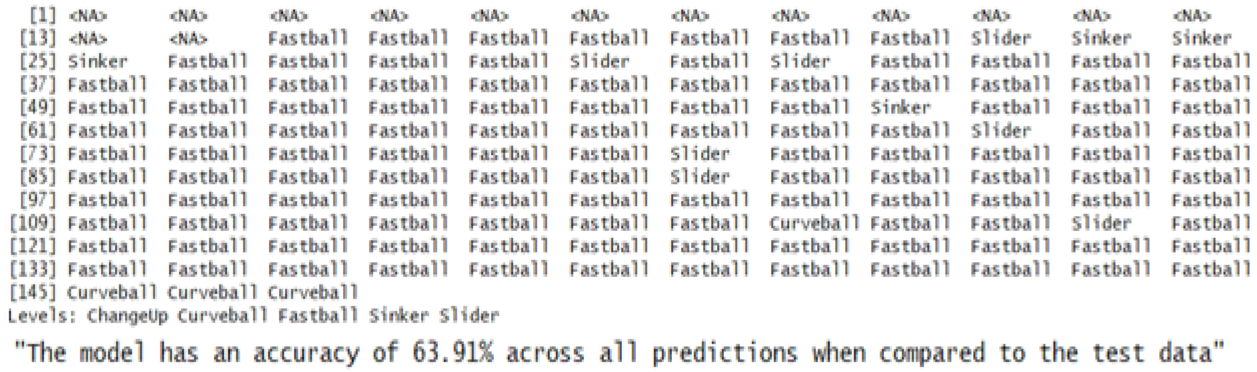
The third goal was to select a small subset of features to use in the model. This was done using forward selection which is where the feature that produces the highest accuracy is added one at a time. A function was written that takes the number of variables you wish to include and the algorithm you with to use and outputs a bar graph showing the accuracy and variable added at each step.

Even though a single decision tree seemed to be more accurate than a random forest model for small subsets of variables, the random forest model was chosen as it allows us to predict the probabilities of each outcome. The variables selected were Plate height, plate side, tagged pitch type and relative vertical angle.

The last task was to use our selected algorithm and variables to create a program where the user enters values for the selected variables and the program outputs the odds of each outcome for each player. To do this a random forest model was used with all the rows in the cleaned data set for player and our selected variables. The dataset was not split into testing and training rows. The program then predicts values for each outcome of pitch type using the predict_proba function in sklearn. It does this 16 times, once for each of the 16 players on the team paired with the same values of the predictor variables that the user entered. A test run of this program can be found in the appendix.

Results

For the pitcher model. An example prediction and accuracy measurement with *m* set to 15 and *n* set to 10 using the first scrimmage game is shown below.



```
  [1] <NA>      <NA>      <NA>      <NA>      <NA>      <NA>      <NA>      <NA>      <NA>      <NA>      <NA>      <NA>     
 [13] <NA>      <NA>      Fastball  Fastball  Fastball  Fastball  Fastball  Fastball  Fastball  Slider    Sinker    Sinker   
 [25] Sinker    Fastball  Fastball  Fastball  Fastball  Slider    Fastball  Slider    Fastball  Fastball  Fastball  Fastball 
 [37] Fastball  Fastball  Fastball  Fastball  Fastball  Fastball  Fastball  Fastball  Fastball  Fastball  Fastball  Fastball 
 [49] Fastball  Fastball  Fastball  Fastball  Fastball  Fastball  Fastball  Fastball  Sinker    Fastball  Fastball  Fastball 
 [61] Fastball  Fastball  Fastball  Fastball  Fastball  Fastball  Fastball  Fastball  Fastball  Slider    Fastball  Fastball 
 [73] Fastball  Fastball  Fastball  Fastball  Fastball  Fastball  Slider    Fastball  Fastball  Fastball  Fastball  Fastball 
 [85] Fastball  Fastball  Fastball  Fastball  Fastball  Fastball  Slider    Fastball  Fastball  Fastball  Fastball  Fastball 
 [97] Fastball  Fastball  Fastball  Fastball  Fastball  Fastball  Fastball  Fastball  Fastball  Fastball  Fastball  Fastball 
[109] Fastball  Fastball  Fastball  Fastball  Fastball  Fastball  Fastball  Curveball Fastball  Fastball  Slider    Fastball 
[121] Fastball  Fastball  Fastball  Fastball  Fastball  Fastball  Fastball  Fastball  Fastball  Fastball  Fastball  Fastball 
[133] Fastball  Fastball  Fastball  Fastball  Fastball  Fastball  Fastball  Fastball  Fastball  Fastball  Fastball  Fastball 
[145] Curveball Curveball Curveball
Levels: ChangeUp Curveball Fastball Sinker Slider
 "The model has an accuracy of 63.91% across all predictions when compared to the test data"
```

Examples using the other scrimmage games at the same *m* and *n* values can be found in the appendix.

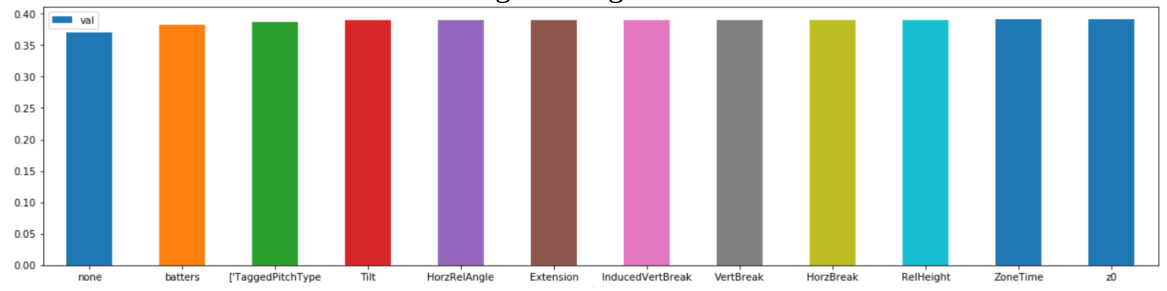
On data from the batter model, when the entire cleaned dataset where the batter was an a&m player. The accuracy on the testing data for the models implemented are as follows:

- Logistic regression: 38% accuracy
- Decision tree classifier: 50% accuracy
- K- nearest neighbor: 34% accuracy
- Linear discriminant analysis: 37% accuracy
- Naive bayes: 6% accuracy

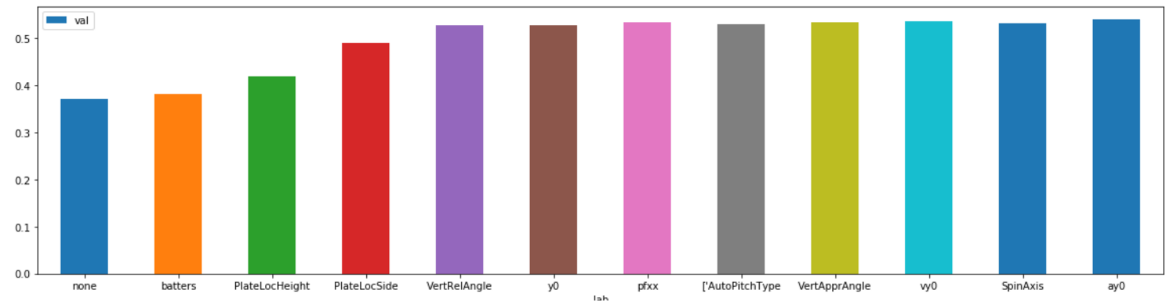
- Support vector machines: 39% accuracy
- Random forest: 59% accuracy

It can be noted that the the tree based algorithms performed significantly better than the regression based algorithms. When forward selection was performed using the following algorithms, the variables and accuracy at each step is shown below;

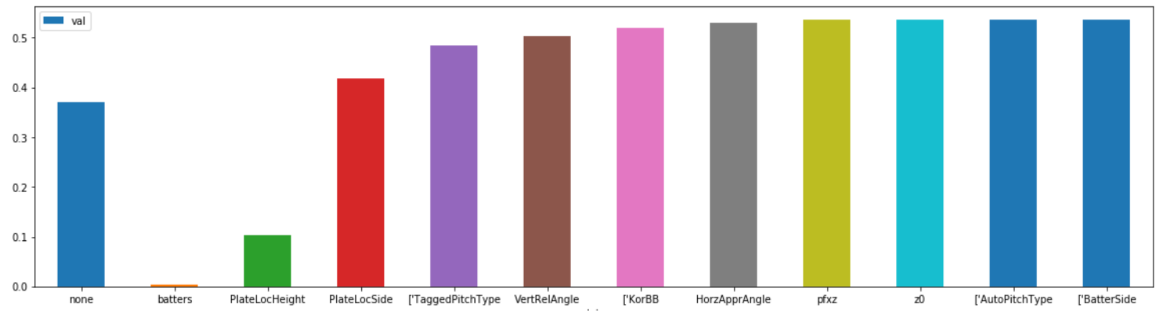
Logistic Regression



Decision tree



Random forest (300 trees)



Of note here is that the function makes the first bar the accuracy of having no predictor variables (i.e choosing the most common response each time) the second bar represents the accuracy if hitter was the only predictor variable, since for our purposes we have to include batter. It is not until after hitter is added to the model that forward selection is done.

Discussion

Future work for the pitcher model would include improving the current model and making a better accuracy measurement. Other classification models such as LDA, QDA, and Random Forests should be developed and compared to the current model. The final goal for this project will be the integration of these tools into the shiny application.

Future work for the batter model would be to build a shiny application version of the program that predicts the odds of each outcome for each player. Since shiny applications run on top of R some of the Python code will have to be rewritten in R. The good new is that most of the work in creating the batter model was in choosing which algorithm and which variables to use. To write a program that does the same thing in R, all someone would have to do is;

1. Subset the uncleaned dataset to only include rows where the batter is an a&m player.
2. Select only the columns 'Batter', 'TaggedPitchType', 'PlateLocHeight', 'PlateLocSide', 'VertRelAngle', and 'PitchCall.'
3. Convert the 'PitchCall' column to numeric values.
4. One-hot-encode the remaining variables.
5. Build a random forest model with 'PitchCall' as the response and the remaining variables as predictors.
6. Write a program that gets predictor variable values from the user and outputs the predicted probabilities of each outcome for each player.

It is of interest that while the accuracy of the random forest model built on randomly selected training data was discovered, we don't know the accuracy of the final model built on the entire dataset. If one were curious about the accuracy of the model trained on the full dataset this could be found using the ‘out of bag error rate’ which is where the data not included in random forest’s bootstrapping process is used as testing data.

Acknowledgements

We would like to thank Dr. Alan Dabney, Darpit Dave, and Guannan Gao for their assistance and guidance throughout this project.

Bibliography

McKinney, Wes and PyData Development Team. "pandas: powerful Python data analysis." 06 August 2018. <https://pandas.pydata.org/pandas-docs/stable/pandas.pdf>.

Ripley, Brian and William Venables. "Package ‘nnet’." *Feed-Forward Neural Networks and Multinomial Log-Linear Models*. 2 February 2016. <https://cran.r-project.org/web/packages/nnet/nnet.pdf>.

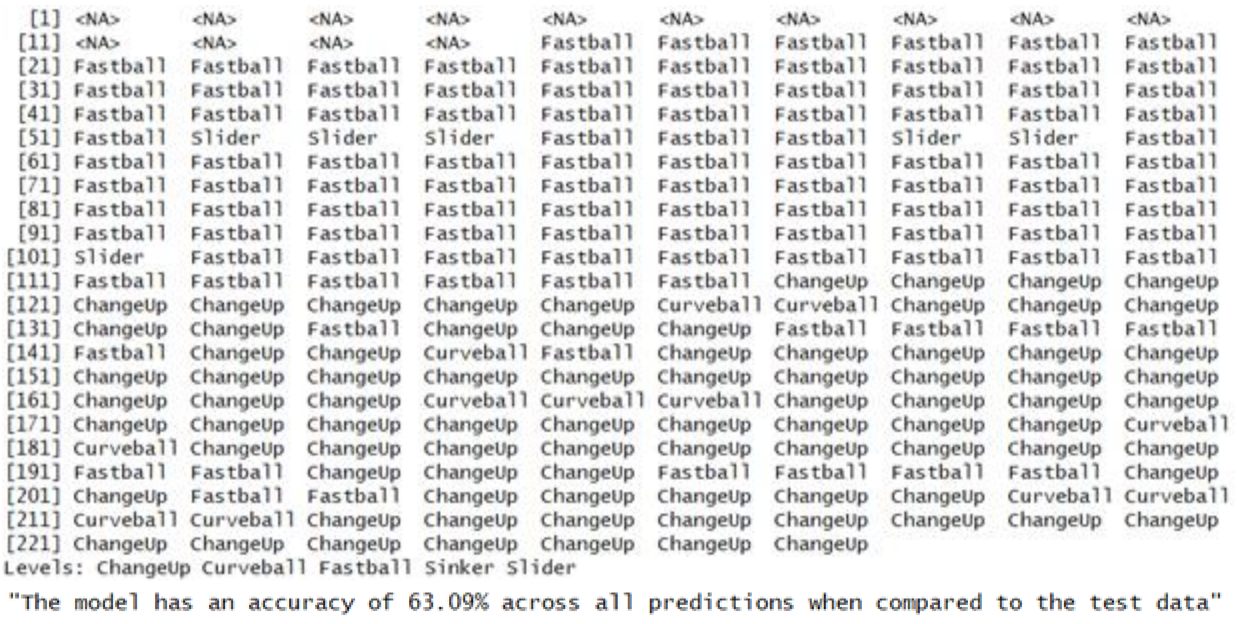
scikit-learn developers. *Documentation of scikit-learn*. 2018. <https://scikit-learn.org/stable/documentation.html>.

The SciPy community. *NumPy Reference*. 04 November 2018. <https://docs.scipy.org/doc/numpy/reference/>.

Appendix

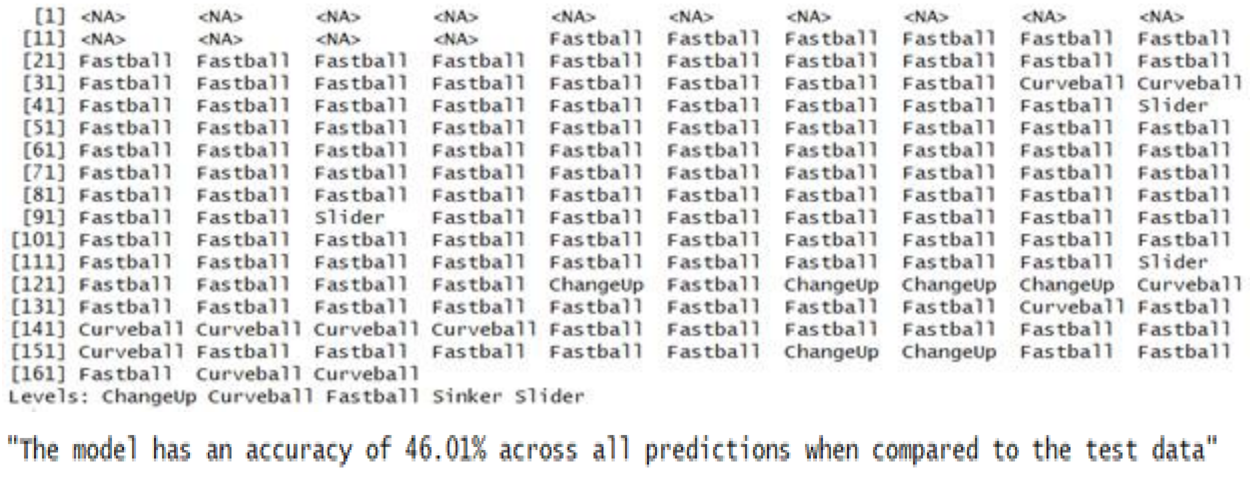
Pitcher Model

Prediction for the second scrimmage game:



```
  [1] <NA>      <NA>      <NA>      <NA>      <NA>      <NA>      <NA>      <NA>      <NA>      <NA>     
 [11] <NA>      <NA>      <NA>      <NA>      Fastball  Fastball  Fastball  Fastball  Fastball  Fastball 
 [21] Fastball  Fastball  Fastball  Fastball  Fastball  Fastball  Fastball  Fastball  Fastball  Fastball 
 [31] Fastball  Fastball  Fastball  Fastball  Fastball  Fastball  Fastball  Fastball  Fastball  Fastball 
 [41] Fastball  Fastball  Fastball  Fastball  Fastball  Fastball  Fastball  Fastball  Fastball  Fastball 
 [51] Fastball  Slider    Slider    Slider    Fastball  Fastball  Fastball  Slider    Slider    Fastball 
 [61] Fastball  Fastball  Fastball  Fastball  Fastball  Fastball  Fastball  Fastball  Fastball  Fastball 
 [71] Fastball  Fastball  Fastball  Fastball  Fastball  Fastball  Fastball  Fastball  Fastball  Fastball 
 [81] Fastball  Fastball  Fastball  Fastball  Fastball  Fastball  Fastball  Fastball  Fastball  Fastball 
 [91] Fastball  Fastball  Fastball  Fastball  Fastball  Fastball  Fastball  Fastball  Fastball  Fastball 
[101] Slider    Fastball  Fastball  Fastball  Fastball  Fastball  Fastball  Fastball  Fastball  Fastball 
[111] Fastball  Fastball  Fastball  Fastball  Fastball  Fastball  ChangeUp  ChangeUp  ChangeUp  ChangeUp 
[121] ChangeUp  ChangeUp  ChangeUp  ChangeUp  ChangeUp  Curveball Curveball ChangeUp  ChangeUp  ChangeUp 
[131] ChangeUp  ChangeUp  Fastball  ChangeUp  ChangeUp  ChangeUp  Fastball  Fastball  Fastball  Fastball 
[141] Fastball  ChangeUp  ChangeUp  Curveball Fastball  ChangeUp  ChangeUp  ChangeUp  ChangeUp  ChangeUp 
[151] ChangeUp  ChangeUp  ChangeUp  ChangeUp  ChangeUp  ChangeUp  ChangeUp  ChangeUp  ChangeUp  ChangeUp 
[161] ChangeUp  ChangeUp  ChangeUp  Curveball Curveball Curveball ChangeUp  ChangeUp  ChangeUp  ChangeUp 
[171] ChangeUp  ChangeUp  ChangeUp  ChangeUp  ChangeUp  ChangeUp  ChangeUp  ChangeUp  ChangeUp  Curveball
[181] Curveball ChangeUp  ChangeUp  ChangeUp  ChangeUp  ChangeUp  ChangeUp  ChangeUp  ChangeUp  ChangeUp 
[191] Fastball  Fastball  ChangeUp  ChangeUp  ChangeUp  Fastball  Fastball  Fastball  Fastball  ChangeUp 
[201] ChangeUp  Fastball  Fastball  ChangeUp  ChangeUp  ChangeUp  ChangeUp  ChangeUp  Curveball Curveball
[211] Curveball Curveball ChangeUp  ChangeUp  ChangeUp  ChangeUp  ChangeUp  ChangeUp  ChangeUp  ChangeUp 
[221] ChangeUp  ChangeUp  ChangeUp  ChangeUp  ChangeUp  ChangeUp  ChangeUp 
Levels: ChangeUp Curveball Fastball Sinker Slider
"The model has an accuracy of 63.09% across all predictions when compared to the test data"
```

Prediction for the third scrimmage game:



```
  [1] <NA>      <NA>      <NA>      <NA>      <NA>      <NA>      <NA>      <NA>      <NA>      <NA>     
 [11] <NA>      <NA>      <NA>      <NA>      Fastball  Fastball  Fastball  Fastball  Fastball  Fastball 
 [21] Fastball  Fastball  Fastball  Fastball  Fastball  Fastball  Fastball  Fastball  Fastball  Fastball 
 [31] Fastball  Fastball  Fastball  Fastball  Fastball  Fastball  Fastball  Fastball  Curveball Curveball
 [41] Fastball  Fastball  Fastball  Fastball  Fastball  Fastball  Fastball  Fastball  Fastball  Slider   
 [51] Fastball  Fastball  Fastball  Fastball  Fastball  Fastball  Fastball  Fastball  Fastball  Fastball 
 [61] Fastball  Fastball  Fastball  Fastball  Fastball  Fastball  Fastball  Fastball  Fastball  Fastball 
 [71] Fastball  Fastball  Fastball  Fastball  Fastball  Fastball  Fastball  Fastball  Fastball  Fastball 
 [81] Fastball  Fastball  Fastball  Fastball  Fastball  Fastball  Fastball  Fastball  Fastball  Fastball 
 [91] Fastball  Fastball  Slider    Fastball  Fastball  Fastball  Fastball  Fastball  Fastball  Fastball 
[101] Fastball  Fastball  Fastball  Fastball  Fastball  Fastball  Fastball  Fastball  Fastball  Fastball 
[111] Fastball  Fastball  Fastball  Fastball  Fastball  Fastball  Fastball  Fastball  Fastball  Slider   
[121] Fastball  Fastball  Fastball  Fastball  ChangeUp  Fastball  ChangeUp  ChangeUp  ChangeUp  Curveball
[131] Fastball  Fastball  Fastball  Fastball  Fastball  Fastball  Fastball  Fastball  Curveball Fastball 
[141] Curveball Curveball Curveball Curveball Fastball  Fastball  Fastball  Fastball  Fastball  Fastball 
[151] Curveball Fastball  Fastball  Fastball  Fastball  Fastball  ChangeUp  ChangeUp  Fastball  Fastball 
[161] Fastball  Curveball Curveball
Levels: ChangeUp Curveball Fastball Sinker Slider

"The model has an accuracy of 46.01% across all predictions when compared to the test data"
```

Batter Model

```
enter PlateLocHeight: 1.4
enter PlateLocSide: -5.34
enter VertRelAng: -2.1
enter TaggedPitchType: Fastball


DeLoach, Zach :
 BallCalled:  0.848     StrikeCalled:  0.012     Inplay:  0.014
```

```
FoulBall:  0.052     HitbyPitch:  0.005     Undefined:  0.038
BallIntentional:  0.0

Helman, Michael :
 BallCalled:  0.862     StrikeCalled:  0.02     Inplay:  0.046
FoulBall:  0.007     HitbyPitch:  0.018     Undefined:  0.011
BallIntentional:  0.0

Shewmake, Braden :
 BallCalled:  0.791     StrikeCalled:  0.123     Inplay:  0.007
FoulBall:  0.019     HitbyPitch:  0.018     Undefined:  0.009
BallIntentional:  0.005

Andritsos, Chris :
 BallCalled:  0.914     StrikeCalled:  0.018     Inplay:  0.015
FoulBall:  0.007     HitbyPitch:  0.004     Undefined:  0.012
BallIntentional:  0.001

Bedford, Cole :
 BallCalled:  0.846     StrikeCalled:  0.081     Inplay:  0.002
FoulBall:  0.018     HitbyPitch:  0.016     Undefined:  0.012
BallIntentional:  0.0

Janca, George :
 BallCalled:  0.86     StrikeCalled:  0.017     Inplay:  0.009
FoulBall:  0.072     HitbyPitch:  0.006     Undefined:  0.01
BallIntentional:  0.001

Frizzell, Will :
 BallCalled:  0.866     StrikeCalled:  0.056     Inplay:  0.009
FoulBall:  0.019     HitbyPitch:  0.005     Undefined:  0.012
BallIntentional:  0.001

Foster, Logan :
 BallCalled:  0.928     StrikeCalled:  0.01     Inplay:  0.005
FoulBall:  0.002     HitbyPitch:  0.003     Undefined:  0.01
BallIntentional:  0.0

Schoenvogel, Baine :
 BallCalled:  0.802     StrikeCalled:  0.038     Inplay:  0.101
FoulBall:  0.006     HitbyPitch:  0.007     Undefined:  0.011
BallIntentional:  0.001

Walters, Aaron :
 BallCalled:  0.876     StrikeCalled:  0.015     Inplay:  0.004
FoulBall:  0.023     HitbyPitch:  0.011     Undefined:  0.039
BallIntentional:  0.001

Coleman, Hunter  :
 BallCalled:  0.879     StrikeCalled:  0.034     Inplay:  0.013
FoulBall:  0.01     HitbyPitch:  0.016     Undefined:  0.015
BallIntentional:  0.001

Wingate, Allonte :
 BallCalled:  0.907     StrikeCalled:  0.017     Inplay:  0.008
FoulBall:  0.018     HitbyPitch:  0.008     Undefined:  0.009
BallIntentional:  0.001

Coleman, Hunter :
 BallCalled:  0.878     StrikeCalled:  0.012     Inplay:  0.016
FoulBall:  0.014     HitbyPitch:  0.04     Undefined:  0.01
BallIntentional:  0.0

Taylor, Cole :
 BallCalled:  0.884     StrikeCalled:  0.03     Inplay:  0.013
FoulBall:  0.021     HitbyPitch:  0.008     Undefined:  0.013
BallIntentional:  0.001

Morris, Chandler :
 BallCalled:  0.872     StrikeCalled:  0.066     Inplay:  0.013
FoulBall:  0.003     HitbyPitch:  0.005     Undefined:  0.01
BallIntentional:  0.001

Blake, Cam :
 BallCalled:  0.886     StrikeCalled:  0.026     Inplay:  0.008
FoulBall:  0.022     HitbyPitch:  0.015     Undefined:  0.012
BallIntentional:  0.001
```