Christopher Boatto

Fastball Command

Fastball command is a major factor that is analyzed to evaluate a pitcher. The pitch that is meant to be thrown in the strike zone the most is the fastball, including both two seam and four seam. Pitchers that can command their fastball within the strike zone will have their secondary pitches become much more effective. Being able to know your fastball's trajectory and placement consistently would allow a pitcher to determine where to start their off-speed pitches to create the illusion of a fastball for every pitch. The vast majority of pitchers pitch off their fastball and need to establish their fastball early in the game to have success.

With the fastballcommand.xlsx data set, I used the Random Forest multivariate model to develop a predictive score for each observation. I used 'Strikes' as a predictive attribute because I am trying to predict which pitcher is able to most frequently throw his fastball in the strike zone. Random Forest is a multivariate mathematical model that creates multiple decision trees simultaneously to predict an outcome. It produces many trees to avoid error bias. For this reason, I decided to use the Random Forest model as opposed to another model like Gradient Boosting which grows each tree one by one and then takes the average result of all trees. Growing each tree simultaneously would give a better understanding of the predicted score than the average of each tree individually. I also did not want to create only one tree using a Decision Tree model because it can be biased whereas using multiple trees, or an entire forest, would remove the bias issue.

All footnotes at the bottom of the document reference the fastballcommand_code.txt file.

Question 1

Answer: The pitcher that commanded his fastball the best was PitcherID 2779 by the slimmest of margins over PitcherID 857.

```
##   PitcherID        FC
## 1       857 0.8195394
## 2      1594 0.7827070
## 3      2696 0.7984499
## 4      2779 0.8298668
## 5    114013 0.7803167
```

Question 2

Answer: The metric I would apply to any pitcher would be the Weighted Fastball Command + (wFC+) metric I created below. Using the same formula of Weighted Runs Created + (wRC+), I applied it to the fastball command metric created above. It shows on a larger scale which pitcher is best at commanding his fastball where 100 is league average and each point higher is one percentage point better than the average. This can apply across the board throughout the league much like wRC+ does.

The wRC+ statistic is a commonly used statistic in baseball to quantify how well a hitter does at creating runs based on the average. It attempts to credit each player's value for each outcome's run expectancy as opposed to treating each outcome equally like 'Batting Average' does. Much like wRC+, wFC+ would credit pitchers for commanding their fastball superior to the rest of the MLB. It would normalize their results over a season to allow front office executives to make an informed decision when evaluating pitchers. [i]

```
##   PitcherID        FC      wFC+
## 1       857 0.8195394 102.16455
## 2      1594 0.7827070  97.57297
## 3      2696 0.7984499  99.53551
## 4      2779 0.8298668 103.45197
## 5    114013 0.7803167  97.27500
```

The first thing I did was check the structure and summary of the data set. The structure and summary of the fastballcommand.xlsx data set shows there are four-character attributes while the rest are numeric. Also, there are only seven NA values within the data set. Given that the NA values constitute 0.05% of the data set, I chose to remove them completely. If the NA values were a major part of the data set, it would be prudent to handle these values by setting them to the average of their respective columns. Setting the NA values to the average of their columns would allow a data scientist to keep as many observations as possible, while not manipulating the inter quartile range of each attribute and thus, would not change the performance of the model. [ii]

Structure

```
## tibble [12,481 x 17] (S3: tbl_df/tbl/data.frame)
##  $ YearID           : num [1:12481] 2018 2018 2018 2018 2018 ...
##  $ GameID           : chr [1:12481] "DFCBC8CF-E0F5-45EF-B07E-0122EE55B257
" "DFCBC8CF-E0F5-45EF-B07E-0122EE55B257" "DFCBC8CF-E0F5-45EF-B07E-0122EE55B25
7" "DFCBC8CF-E0F5-45EF-B07E-0122EE55B257" ...
##  $ GamePitchSequence : num [1:12481] 1 2 3 4 5 6 8 9 10 11 ...
##  $ PAofInning       : num [1:12481] 1 1 1 2 2 2 3 3 3 3 ...
```

```
##  $ AtBatPitchSequence: num [1:12481] 1 2 3 1 2 3 1 2 3 4 ...
##  $ Inning            : num [1:12481] 1 1 1 1 1 1 1 1 1 ...
##  $ Balls             : num [1:12481] 0 1 1 0 0 0 0 0 0 0 ...
##  $ Strikes           : num [1:12481] 0 0 1 0 1 2 0 1 2 2 ...
##  $ PitcherID         : num [1:12481] 857 857 857 857 857 857 857 857 857 8
57 ...
##  $ PitcherHand       : chr [1:12481] "L" "L" "L" "L" ...
##  $ BatSide           : chr [1:12481] "R" "R" "R" "L" ...
##  $ PitchType         : chr [1:12481] "FA" "FA" "SI" "FA" ...
##  $ Velocity          : num [1:12481] 90.8 91.3 89.6 92.1 90.5 ...
##  $ VerticalBreak     : num [1:12481] 17.64 17.82 7.51 19.4 12.72 ...
##  $ HorizontalBreak   : num [1:12481] -6.05 -4.2 -14.93 -9.25 -15.63 ...
##  $ PlateLocHeight    : num [1:12481] 3.42 1.83 1.7 2.89 1.7 ...
##  $ PlateLocSide      : num [1:12481] -0.574 0.484 -0.502 0.209 -0.588 ...
```

Summary

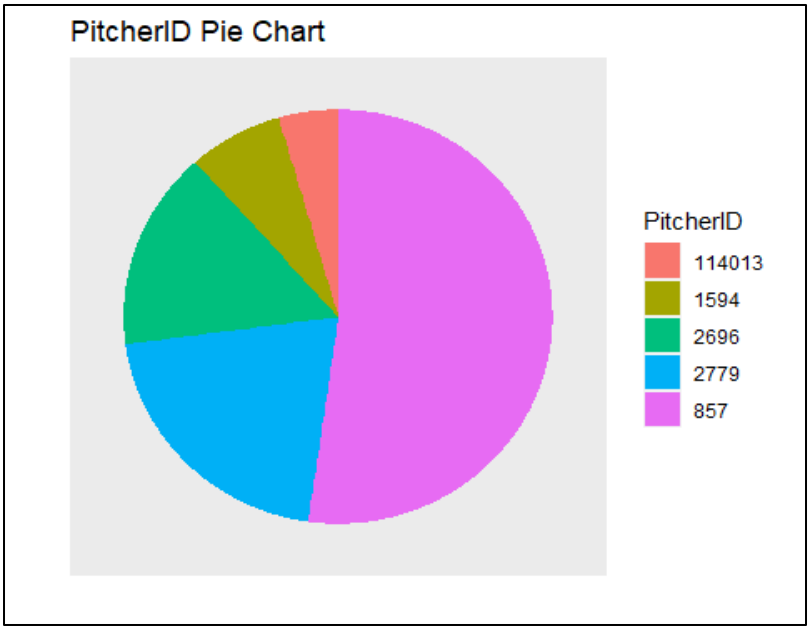
```
##      YearID        GameID          GamePitchSequence   PAofInning
##  Min.   :2018   Length:12481       Min.   :  1.0     Min.   : 1.00
##  1st Qu.:2018   Class :character   1st Qu.: 44.0     1st Qu.: 1.00
##  Median :2018   Mode  :character   Median : 95.0     Median : 2.00
##  Mean   :2019                      Mean   :102.2     Mean   : 2.65
##  3rd Qu.:2019                      3rd Qu.:151.0     3rd Qu.: 3.00
##  Max.   :2020                      Max.   :480.0     Max.   :11.00
##
##  AtBatPitchSequence     Inning           Balls           Strikes
##  Min.   : 1.00      Min.   : 1.000   Min.   :0.0000   Min.   :0.0000
##  1st Qu.: 1.00      1st Qu.: 2.000   1st Qu.:0.0000   1st Qu.:0.0000
##  Median : 2.00      Median : 3.000   Median :1.0000   Median :1.0000
##  Mean   : 2.82      Mean   : 3.553   Mean   :0.8922   Mean   :0.8116
##  3rd Qu.: 4.00      3rd Qu.: 5.000   3rd Qu.:2.0000   3rd Qu.:2.0000
##  Max.   :13.00      Max.   :15.000   Max.   :3.0000   Max.   :2.0000
##
##    PitcherID      PitcherHand          BatSide           PitchType
##  Min.   :   857   Length:12481       Length:12481       Length:12481
##  1st Qu.:   857   Class :character   Class :character   Class :character
##  Median :  2696   Mode  :character   Mode  :character   Mode  :character
##  Mean   : 19274
##  3rd Qu.:  2779
##  Max.   :114013
##
##     Velocity     VerticalBreak    HorizontalBreak    PlateLocHeight
##  Min.   :85.19   Min.   : 1.258   Min.   :-21.2033   Min.   :-1.816
##  1st Qu.:90.95   1st Qu.:11.653   1st Qu.: -9.4091   1st Qu.: 2.004
##  Median :92.30   Median :15.356   Median : -4.2730   Median : 2.588
##  Mean   :92.40   Mean   :14.397   Mean   : -0.4446   Mean   : 2.614
##  3rd Qu.:93.85   3rd Qu.:17.587   3rd Qu.:  8.9039   3rd Qu.: 3.214
##  Max.   :99.24   Max.   :23.751   Max.   : 23.5143   Max.   : 7.161
##  NA's   :7       NA's   :7        NA's   :7          NA's   :7
##   PlateLocSide
```

```
##  Min.   :-3.69103
##  1st Qu.:-0.60808
##  Median :-0.07387
##  Mean   :-0.06747
##  3rd Qu.: 0.47759
##  Max.   : 3.18969
##  NA's   :7
```

Next, I decided to determine how many pitchers were in the data set. Knowing how many pitchers there are in the fastballcommand.xlsx data set will provide an understanding of unique values in the PitcherID attribute. This would allow for a cross reference in the off chance my final answer has a different quantity of PitcherID's present. Below shows five unique PitcherID's. It is a little interesting that there are only 5 pitchers in this data set which is unusual for a team. They must have stayed fairly healthy throughout the seasons. [iii]

```
## [1]    857 114013   2696   1594   2779
```

Then I created a Pie chart based off the 'PitcherID' to show how often each pitcher pitches. In order to do so, I changed the 'PitcherID' attribute briefly into a character attribute to allow for the generation of the pie chart. I would immediately then change it back to a numeric value later for the purpose of aggregation. The aggregate function is used to group the data together based off a specific parameter and then form a summary based off the min, max, sum, or mean of all other attributes. Thus, the total amount of pitches per pitcher would determine how I would define my summary parameter. As you can see below, PitcherID 857 threw more than 50% of the pitches in this data set. [iv]

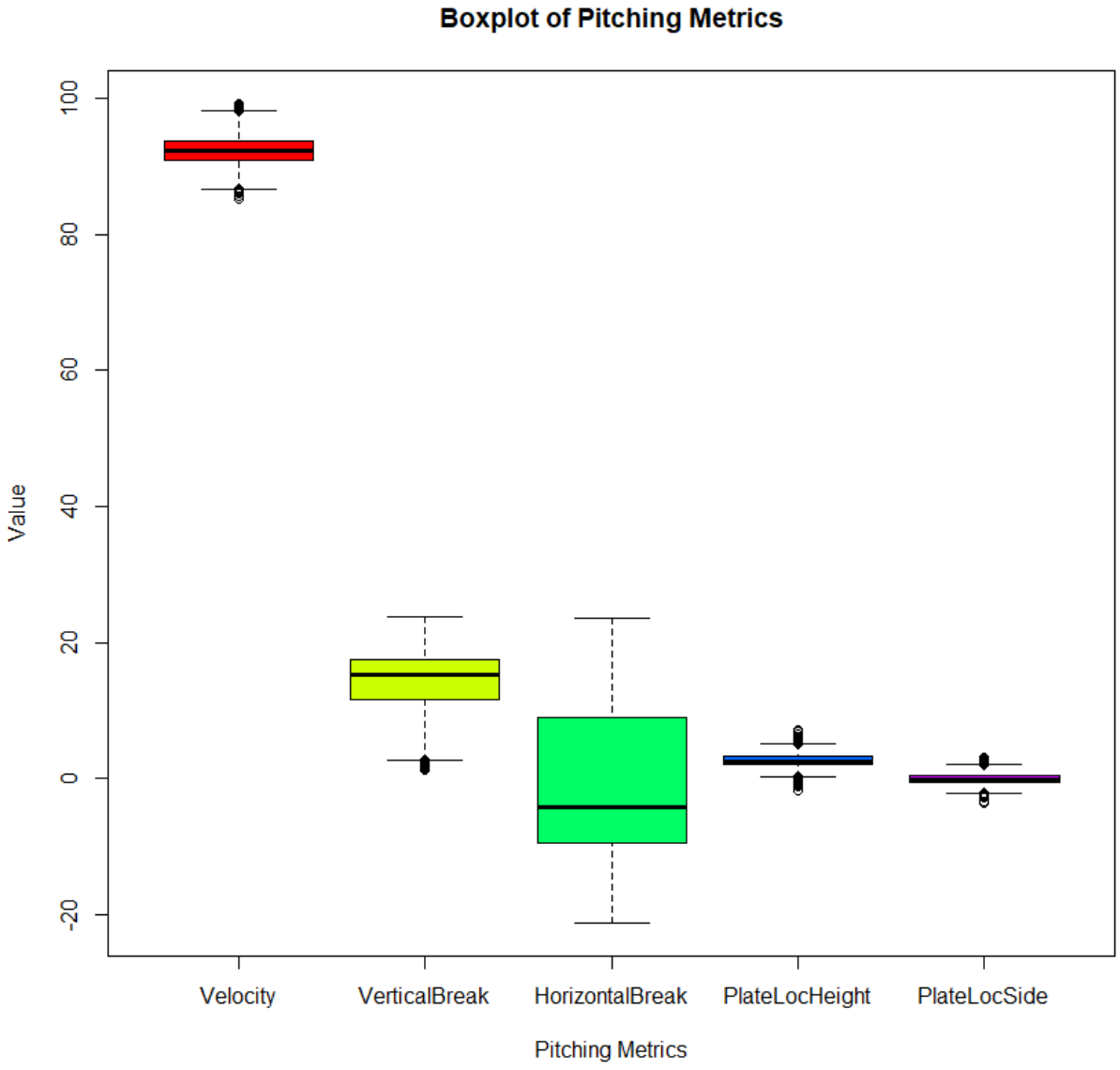


I decided to recheck the summary after some exploration to see if my changes took effect. There is a difference between this summary and the previous one. There are no NA values and after the previous line was ran, 'PitcherID' went back to being numeric. [v]

```
#      YearID        GameID          GamePitchSequence   PAofInning
##  Min.   :2018   Length:12474       Min.   :  1.0     Min.   : 1.000
##  1st Qu.:2018   Class :character   1st Qu.: 44.0     1st Qu.: 1.000
##  Median :2018   Mode  :character   Median : 94.0     Median : 2.000
##  Mean   :2019                      Mean   :102.2     Mean   : 2.649
##  3rd Qu.:2019                      3rd Qu.:150.0     3rd Qu.: 3.000
##  Max.   :2020                      Max.   :480.0     Max.   :11.000
##  AtBatPitchSequence     Inning          Balls          Strikes
##  Min.   : 1.00      Min.   : 1.000   Min.   :0.0000   Min.   :0.0000
##  1st Qu.: 1.00      1st Qu.: 2.000   1st Qu.:0.0000   1st Qu.:0.0000
##  Median : 2.00      Median : 3.000   Median :1.0000   Median :1.0000
##  Mean   : 2.82      Mean   : 3.553   Mean   :0.8923   Mean   :0.8116
##  3rd Qu.: 4.00      3rd Qu.: 5.000   3rd Qu.:2.0000   3rd Qu.:2.0000
##  Max.   :13.00      Max.   :15.000   Max.   :3.0000   Max.   :2.0000
##    PitcherID      PitcherHand          BatSide           PitchType
##  Min.   :   857   Length:12474       Length:12474       Length:12474
##  1st Qu.:   857   Class :character   Class :character   Class :character
##  Median :  2696   Mode  :character   Mode  :character   Mode  :character
##  Mean   : 19275
##  3rd Qu.:  2779
##  Max.   :114013
##     Velocity     VerticalBreak    HorizontalBreak    PlateLocHeight
##  Min.   :85.19   Min.   : 1.258   Min.   :-21.2033   Min.   :-1.816
```

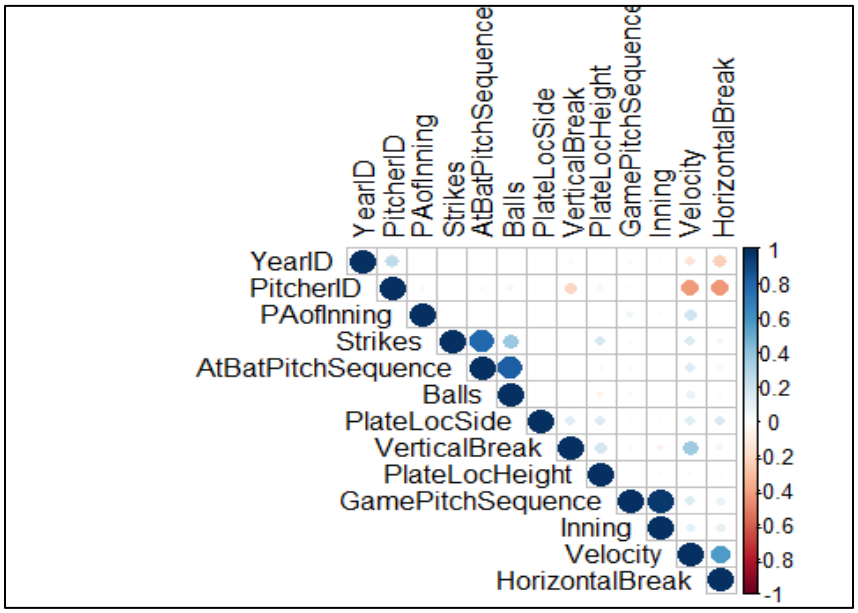
```
##  1st Qu.:90.95   1st Qu.:11.653   1st Qu.: -9.4091   1st Qu.: 2.004
##  Median :92.30   Median :15.356   Median : -4.2730   Median : 2.588
##  Mean   :92.40   Mean   :14.397   Mean   : -0.4446   Mean   : 2.614
##  3rd Qu.:93.85   3rd Qu.:17.587   3rd Qu.:  8.9039   3rd Qu.: 3.214
##  Max.   :99.24   Max.   :23.751   Max.   : 23.5143   Max.   : 7.161
##   PlateLocSide
##  Min.   :-3.69103
##  1st Qu.:-0.60808
##  Median :-0.07387
##  Mean   :-0.06747
##  3rd Qu.: 0.47759
##  Max.   : 3.18969
```

I then created a boxplot to show the inter quartile ranges along with the extremities of the pitch parameters; ‘Velocity’, ‘VerticalBreak’, ’HorizontalBreak’, ‘PlateLocHeight’, and ‘PlateLocSide’. There are no massive outliers within the data set, which indicates that the data is fairly clean. Usually there is more cleaning needed to be done with data sets, but this one was exceptional with only seven NA values. [vi]



Next, I created the correlation matrix below. It shows how well certain attributes correlate to each other. The thicker and bluer a circle became, the more positive the attributes are correlated. Red represents negatively correlated attributes. Some of the attributes correlating to each other make sense such as ‘Ball’ and ‘Strike ’to ’AtBatPitchSequence.’ The majority of At Bats in the MLB have a count to them as not many batters consistently swing at the first pitch and put it into play. ‘GamePitchSequence’ and ‘Inning’ also are positively correlated which makes sense as the deeper a pitcher goes in the game the more pitches he would throw.

One of the aspects notable from the matrix is that ‘PitcherID’ to ‘Velocity’ and ‘HorizontalBreak’ are more negatively correlated. This would make sense as every pitcher is different. Some pitchers throw two seams causing more horizontal break than those who throw four seams. One aspect to note is that ‘VerticalBreak’ has almost no correlation. This would make sense since fastballs are not supposed to break up and down. This is more an attribute of off-speed pitches. [vii]

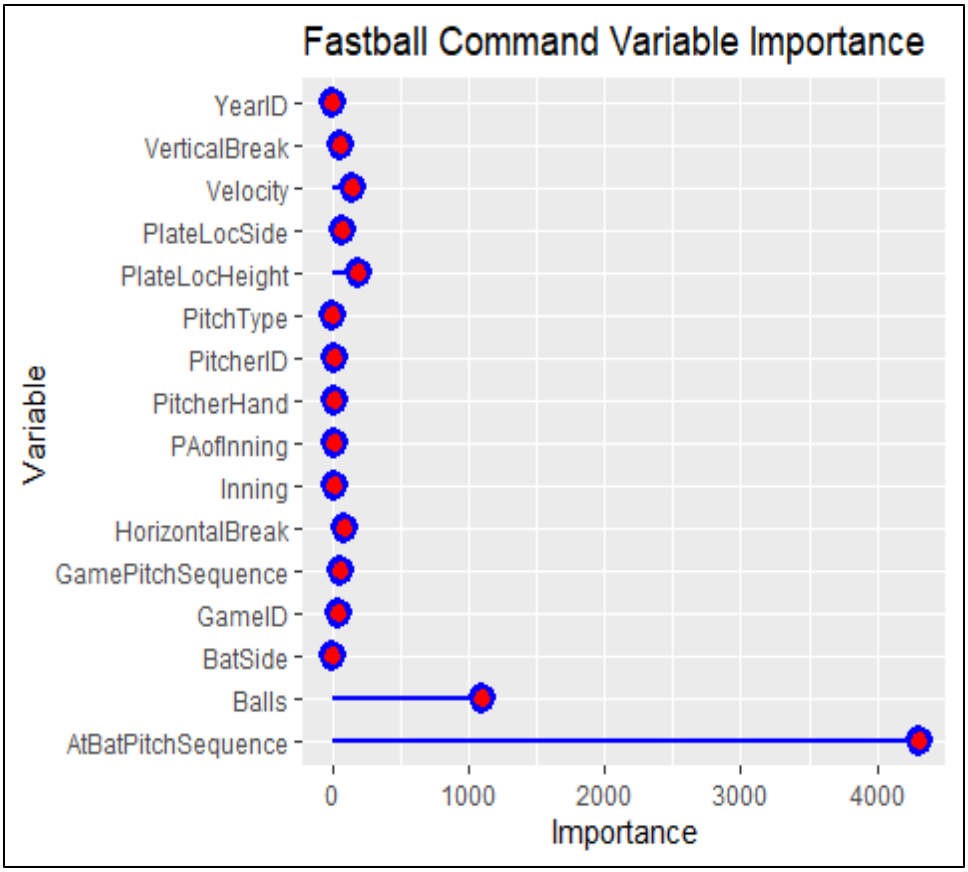


In order to properly perform the analysis, I split my data set into the train and test sets. This would allow me to see how well the model would do at predicting for both sets.[viii]

I created a base random forest model to get an understanding of the importance of each attribute. The Importance function in the 'randomForest' package shows how much each attribute will contribute to the model. With a high level of importance, an attribute would take over the model when determining a prediction, rendering all others obsolete. To determine this, I created my model using the 'Strikes' attribute as my dependent variable and all other attributes as independent variables. The 'Strikes' attribute is described as 'Strikes prior to the pitch' which is exactly what would need to be determined. As previously mentioned, fastballs are the only pitch that are meant to be thrown mostly in the strike zone so using the 'Balls' outcome would be counterintuitive. Using other attributes would be impractical due to them not being an outcome trait.[ix]

The below two charts show what attributes dominated this model. As you can see the 'AtBatPitchSequence' and 'Balls' attributes were much further ahead of every other attribute. This makes sense as a strike is an outcome of a pitch within the At Bat. Also, the attribute 'Balls' is of high importance because of it being the only other outcome to a strike when a pitch is thrown. Even when a pitch is put in play or fouled off, it is tabulated as a strike because the batter swung.[x]

```
##                     IncNodePurity
## YearID                  10.298087
## GameID                  42.441320
## GamePitchSequence       56.862229
## PAofInning              24.402360
## AtBatPitchSequence    4304.677346
## Inning                  18.675308
## Balls                 1098.444318
## PitcherID               25.692612
## PitcherHand             13.188189
## BatSide                  8.812651
## PitchType                5.605226
## Velocity               154.746361
## VerticalBreak           65.751044
## HorizontalBreak         93.140218
## PlateLocHeight         187.928469
## PlateLocSide            73.577104
```



The newest model below was created based off the importance of the last model and personal experience of the game. There are attributes of the new model that were included and some that were omitted to obtain a better understanding of what pitcher is able to command his fastball the best. The following are the reasons as to why each attribute was included or omitted.

Attribute	**Meaning**	**Reason for Including the Attribute**
'Inning'	Inning in which the event occurred	This was included in the model as speaking from experience, strikes in the later innings are always harder to achieve, especially in a high-pressure situation. Therefore, 'Inning' was left in because of the effect it could have on a pitcher.
'GamePitchSequence' (GPS)	Pitch number within the given game; taken together, 'GameID' and 'GamePitchSequence' should uniquely identify each event	Knowing the pitch number in a given game and season would show if a pitcher is tiring when pitches begin to accumulate throughout the year or improves with more repetition. It should be noted if the amount of repetition causes the pitcher to improve his command or tire and lose command.
'GameID'	Identifier for the game in which the event occurred	Combined with 'GPS,' 'GameID' gives you a unique ID of each pitch in the data set. It also assists in determining the tiring vs improvement theory for the model.

‘Velocity’	Velocity at release	To throw a ball hard, there is a point in which your body needs to move at a high speed. The faster your body moves, the harder it is to control it, thus making it difficult to control the ball.
‘VerticalBreak’	Vertical break in inches	This attribute will have a similar reasoning to ‘HorizontalBreak’ as the more movement a pitch has the harder it is to control.
‘HorizontalBreak’	Horizontal break in inches (positive = toward third base)	Similar to ‘VerticalBreak’, the more movement, the tougher it is to control the pitch.
‘PlateLocHeight’	Vertical pitch location in feet	This attribute was included as it is the result of the vertical pitch placement when the ball crosses the plate. The combination with ‘HorizontalBreak’ gives a complete understanding of where the ball crossed.

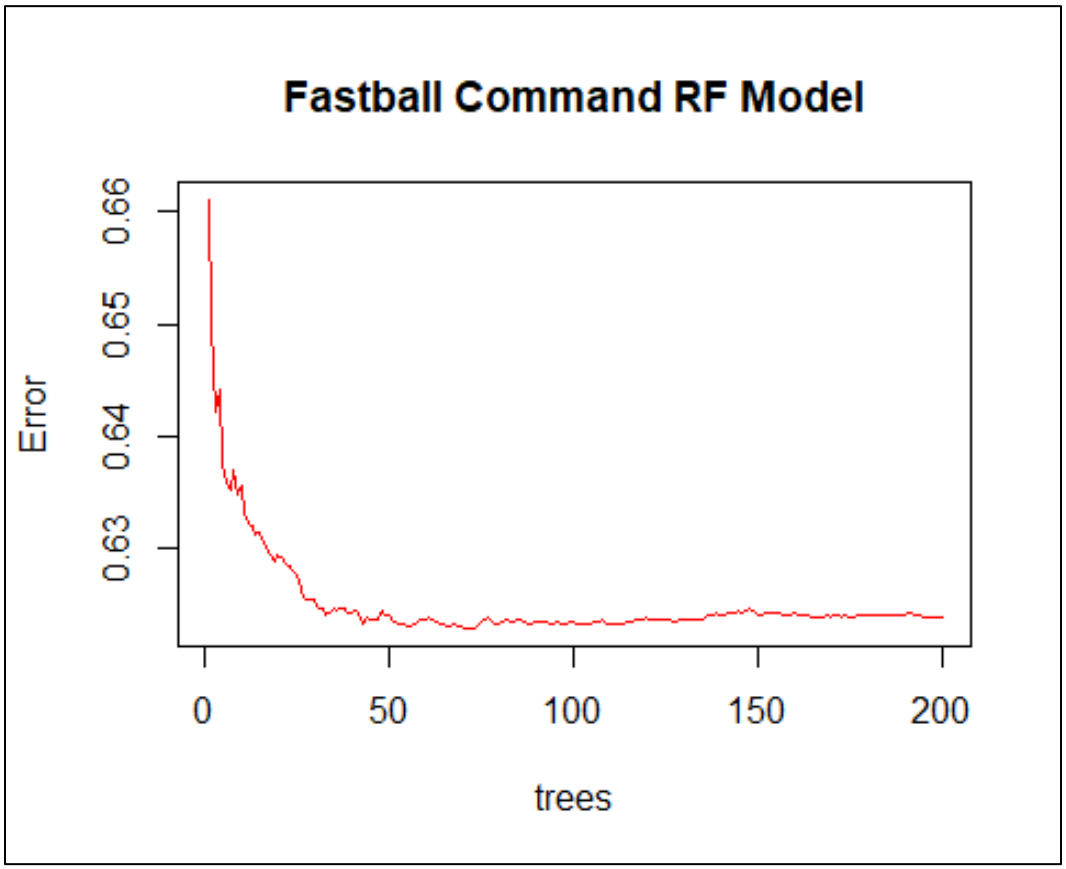
Attribute	**Meaning**	**Reason for Omitting the Attribute**
‘Balls’	Balls prior to the pitch	There were two factors why this attribute was removed. These are its high importance score and it being the opposite outcome of the desired result. Fastballs are meant to be thrown for strikes, therefore keeping the opposite outcome in the model would be counterintuitive.
'AtBatPitchSequence’(ABPS)	Pitch number within a given plate appearance	This attribute would cause a redundancy with the combination of ‘GameID’ and ‘GPS.’ The combination of ‘GameID’ and ‘GPS’ would uniquely identify each event. Therefore, each pitch tabulated in the ‘ABPS’ would already be accounted for within the ‘GameID’ and ‘GPS’ combination. Another reason why this attribute was left out was due to its high importance score. Leaving it in would dominate the model thus resulting in a redundant predictive outcome.

‘PAofInning’	Plate appearance number within a given inning	The combination of ‘PAofInning,’ ‘GPS’ and ‘GameID’ would give a sufficient understanding of the number of pitches thrown within the inning thus leaving this attribute redundant.
‘YearID’	Season in which the event occurred	Each ‘GameId’ is different in itself. Therefore, there is no need to include the ‘YearID’ as it becomes redundant.
‘PitcherID’	Pitcher identity	The ‘PitcherID’ is the person we are attempting to evaluate thus no need to include him in the model.
‘BatSide’	Batter’s handedness	The side in which the batter hits from has no effect on if the pitch is a strike or ball.
‘PitchType’	Pitch classification	‘FA’ refers to four seam fastball, while ‘SI’ refers to two seam fastball: This is redundant as the pitch properties within the model will also determine if a pitch is a two seam or four seam pitch.

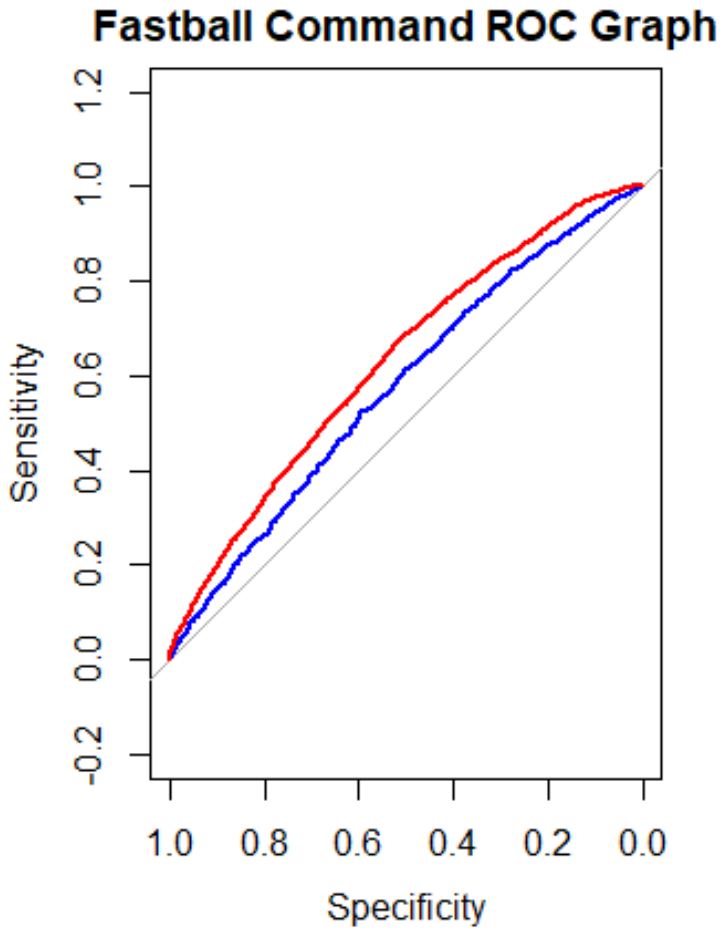
I set the maxnodes to eight because I wanted my decision tree depiction later to be as concise as possible. A decision tree depiction can become cluttered if you do not specify the parameters.[xi]

With the "response" method, I predicted the scores using the model above on the observations from the train and test data sets. I used "response" instead of "class" because I wanted a numerical result for each observation rather than a value being assigned. I then added the predicted scores to their respective data sets as an additional attribute and renamed them for consistency. I bound both the train and test sets back together to create a full data set with predictions to create the command_full data set.[xii]

The graph below shows how well the model ran with the error bias. At around 150 trees, the error bias begins to flatline and level off, thus showing a decrease in error bias. There was a brief increase in error however, right before 150, but it then leveled off after. The model would have worked well after 150 trees, but 200 was chosen to ensure accuracy.[xiii]

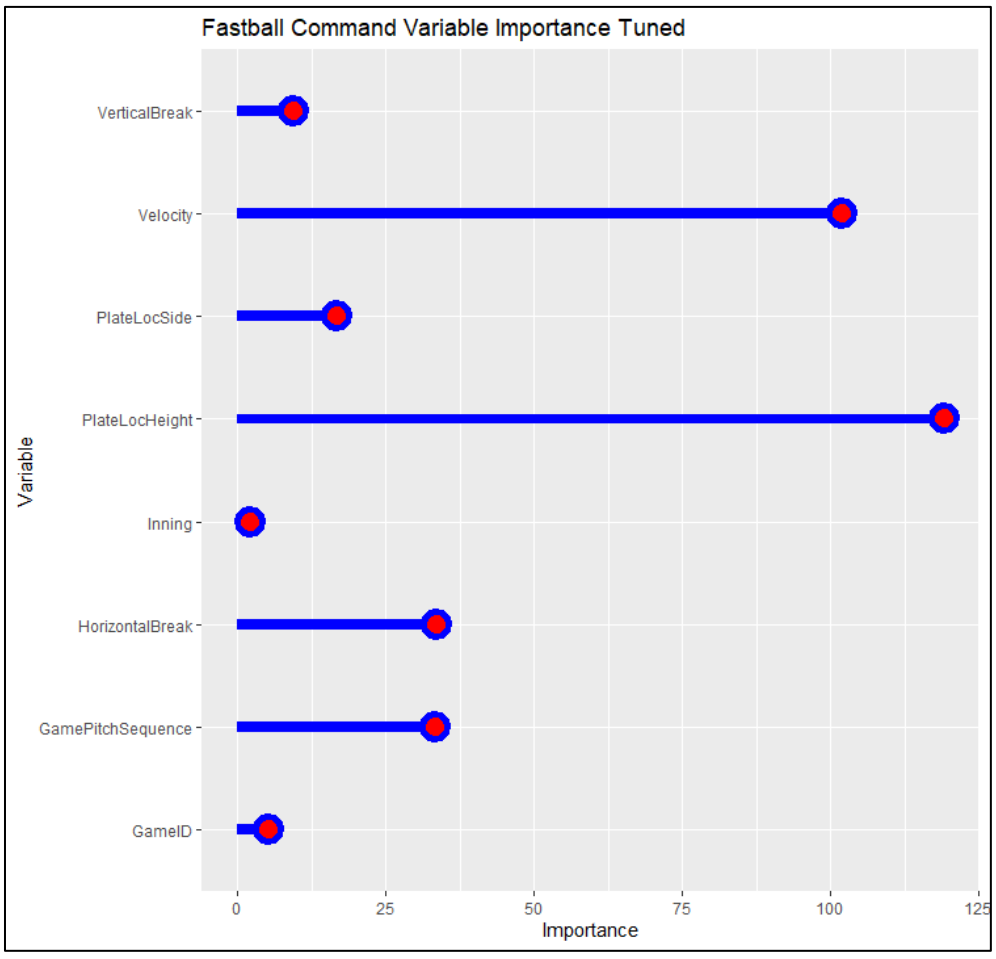


The Receiver Operating Characteristic (ROC) Curve below shows how well the model performed at all the thresholds. The graph below shows that both the train (red line) and test (blue line) performed well moving to the top left corner on a curve thus showing that the model did well in predicting between the train and test sets.[xiv]



I decided to recreate the importance chart to see the differences between the attributes in the model. It may look like a large discrepancy but basing this off the first Variable Importance chart above it is much more balanced, thus no one attribute took over the model.[xv]

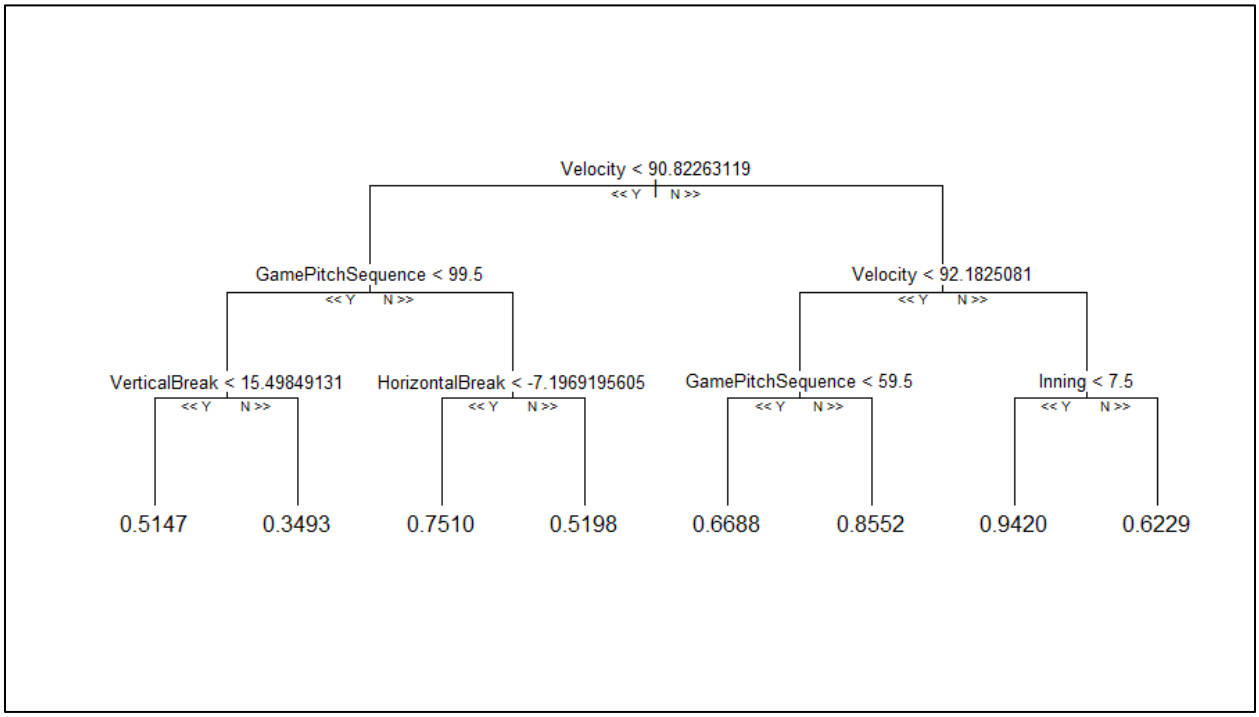
```
##                    IncNodePurity
## Velocity              81.768822
## VerticalBreak         11.048630
## HorizontalBreak       41.697258
## PlateLocHeight       105.329225
## PlateLocSide          14.475988
## GamePitchSequence     34.401014
## GameID                 4.645625
## Inning                 1.622984
```



The decision tree below is based off the ‘reprtree’ package created by Github profile skinner927. Reprtree allows a data analyst to create a decision tree from the random forest model they create.

The tree below is a depiction of the 100th tree in the “command_tuned_model”. It shows the flow in which the model made its decision to determine the predictive score of a strike. The model makes a binary decision at each node or parameter, “Yes” or “No”, which then leads to another and another until you reach a final predictive score grouping at the bottom.

I had previously commented on the maxnodes parameter being important due to clarity in this tree. Setting the maxnodes parameter to eight will create a depiction of eight result groupings. This allows for the tree to be much easier to read and understand. Without doing so, this tree would have been extremely cluttered, with every outcome as a result. You can choose any tree from your model to depict. Each tree will be slightly different, thus choosing a different tree will result in different depictions.[xvi]



I selected the two attributes from command_full that I would need to solve the problem of which pitcher commanded his fastball the best from command_full, 'PitcherID' and 'FC_pred'.

Using the 'aggregate function,' I grouped the data together based off the 'PitcherID' and then formed a summary based off the average of the predicted scores for the pitches each respective pitcher threw. The pie chart was useful here because 'PitcherID' 857 threw the vast majority of pitches in the data set. Using 'sum' in the FUN parameter, it would have automatically given PitcherID 857 the highest score. Using 'mean' made it fair amongst all pitchers to gain a better understanding of which pitcher commanded his fastball the best.[xvii]

[i] Creation of wFC+
[ii] Checking Structure, Summary and Omitting NA values
[iii] Checking Unique PitcherID
[iv] PitcherID Pie Chart and conversions of PitcherID
[v] Recheck of Summary
[vi] Creation of Boxplot
[vii] Correlation Matrix Creation
[viii] Train and Test Data Creation
[ix] Base Random Forest Model Creation
[x] Variable Importance of Base Model and Graph
[xi] Random Forest Tuned
[xii] Predictions, Bindings and Renaming After Tuned Model
[xiii] RF Model Plot
[xiv] ROC Chart Creation
[xv] Variable Importance Tuned and Graph
[xvi] Creation of Decision Tree 100
[xvii] Selection and Aggregation of Final Data Set and Renaming to FC