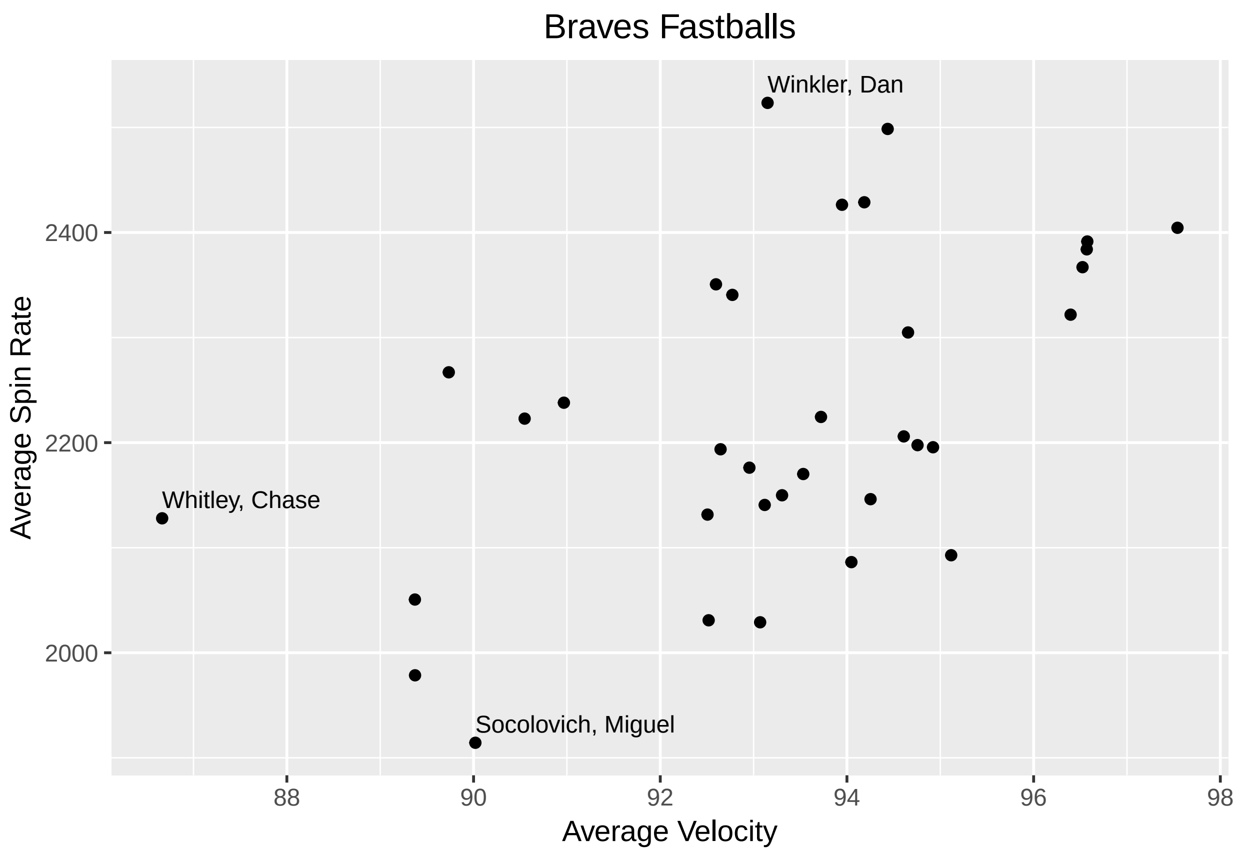
Braves Questionnaire

Nolan McCafferty

11/12/2018

1. Prepare a scatterplot that shows each player's average fastball velocity and spin rate from the 2018 season, as if you were presenting it to a baseball executive. Plot velocity on the x-axis and spin rate on the y-axis.

Below is the plot of each player's average fastball velocity and spin rate from the 2018 season, with a few interesting players labeled.



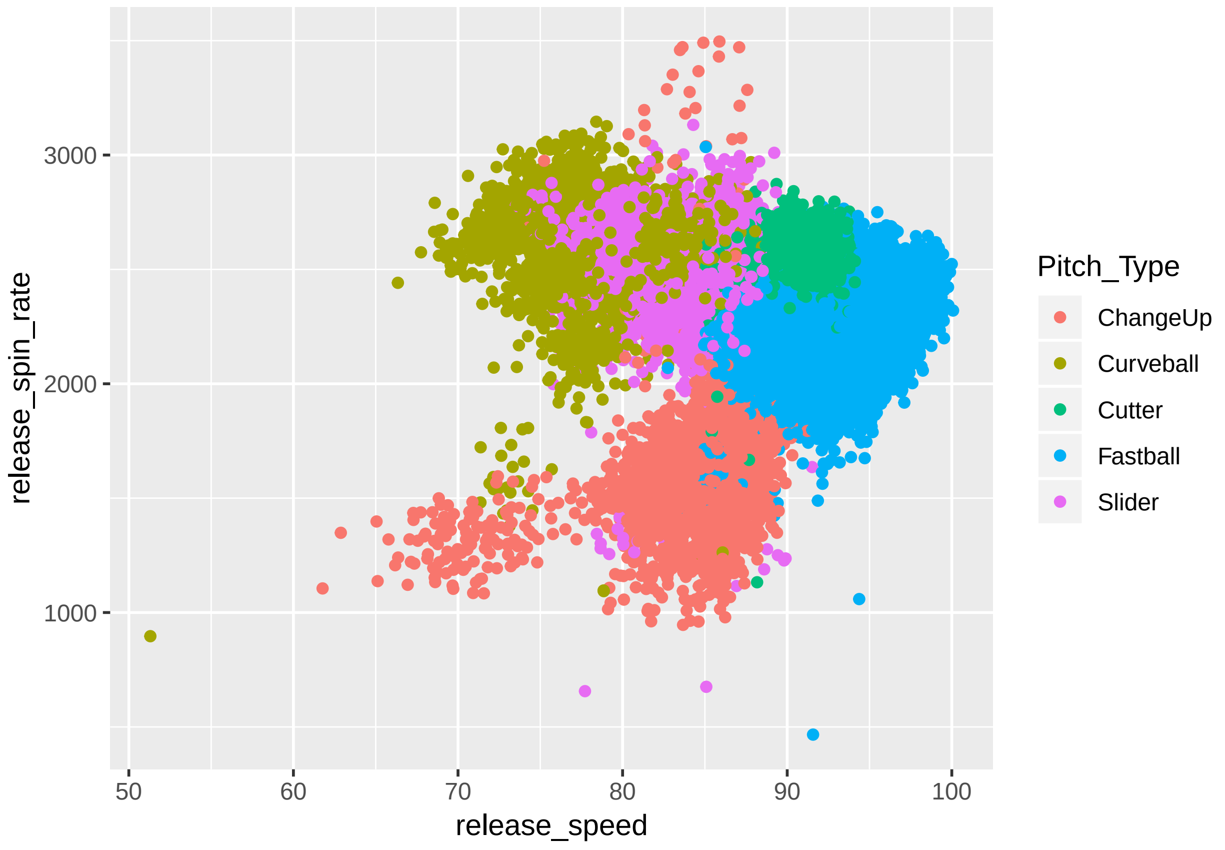
2. Find all pitchers in the dataset with a slider usage rate of at least 20%. Arrange those pitchers in a data frame from the highest slider usage to the lowest. Print a data frame of the top 5 pitchers in slider usage, including how many sliders and total pitches they threw.

```
## Selecting by Slider.Perc
```

Pitcher	Total.Pitches	Num.Sliders	Slider.Perc
Ravin, Josh	52	29	55.8
Jackson, Luke	703	311	44.2
Wisler, Matt	402	171	42.5
Moylan, Peter	505	180	35.6
Gohara, Luiz	266	87	32.7

3. Using Trackman data from before July 1st of the season, build TWO different pitch classification models

to classify all pitches in time period from July 1 to the end of the season: Fastball, Changeup, Curveball, Slider and Cutter. Evaluate and compare the performances of your models using any method(s) you'd prefer. Explain your results in 500 words or less.



After some initial data exploration, we can see that for the most part each pitch forms a cluster when velocity and spin rate are plotted against each other, with the cutter being the tighest of these clusters. There are some interesting things to note. First, there are a couple sliders and fastballs with spin rates below 1000 rpm (and a curveball with velocity below 55). These pitches are obvious outliers and thus will be removed from the data. Also, there is an interesting group of change ups with spin rate over 3000 rpm. It is clear that this group of 21 pitches is not indicative of a true change up spin rate, so these will be discarded as well.

Now we split the data into training and test splits. Our first model will be a multinomial logistic regression approach:

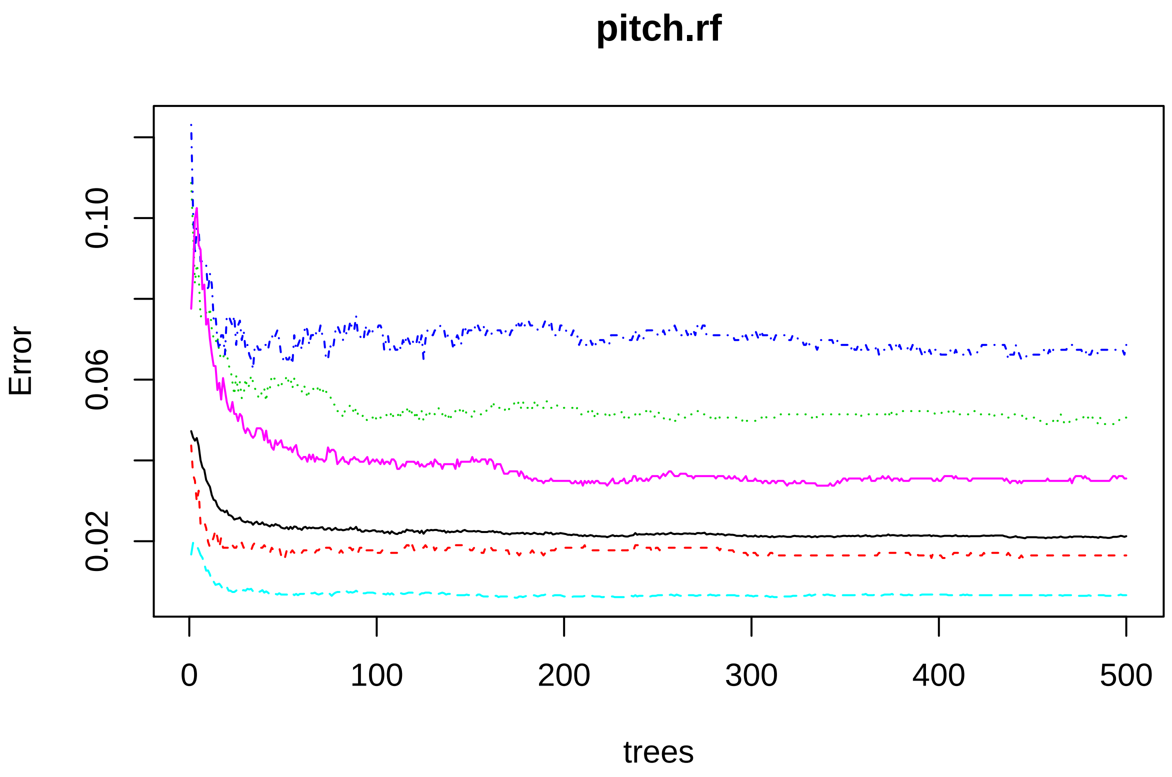
Now predicting on our test data:

```
predictions <- predict(pitch.ml, test)
mean(as.character(predictions) == as.character(test$Pitch_Type))
```

```
## [1] 0.8997028
```

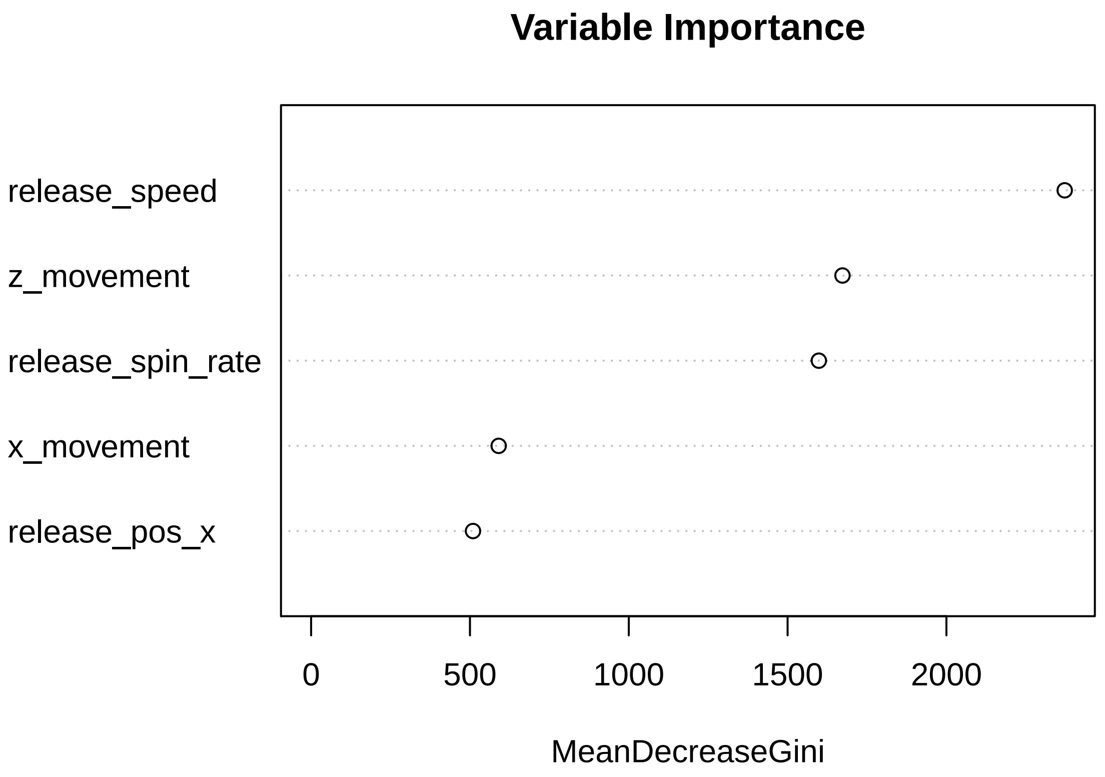
We can see that the accuracy of our model is about 90%, which is not bad but I think we can do better. For the second model, we will use a random forest model.

```
# build model
pitch.rf <- randomForest(factor(Pitch_Type) ~ release_speed + x_movement +
                           z_movement + release_spin_rate + release_pos_x +
                           release_pos_z + spin_dir, data=train, mtry=3, important=T)

plot(pitch.rf)
```



The error of our model seems to stabalize around 100 trees.

```r
varImpPlot(pitch.rf, sort=T, main="Variable Importance", n.var=5)
```



```r
var.imp <- data.frame(importance(pitch.rf, type=2))
```

Training this model, we get a 100% accuracy on the training data. Also, we see the most important variables in the model are release speed, z movement, release spin rate, x movement, and release position x. The variables plate_x, plate_y, and release extension were found to be unimportant, the model accuracy increased when these variables were removed. Now for the next step we will predict the pitch type for the second half of the season, examine our results using a confusion matrix, and evaluate our model accuracy.

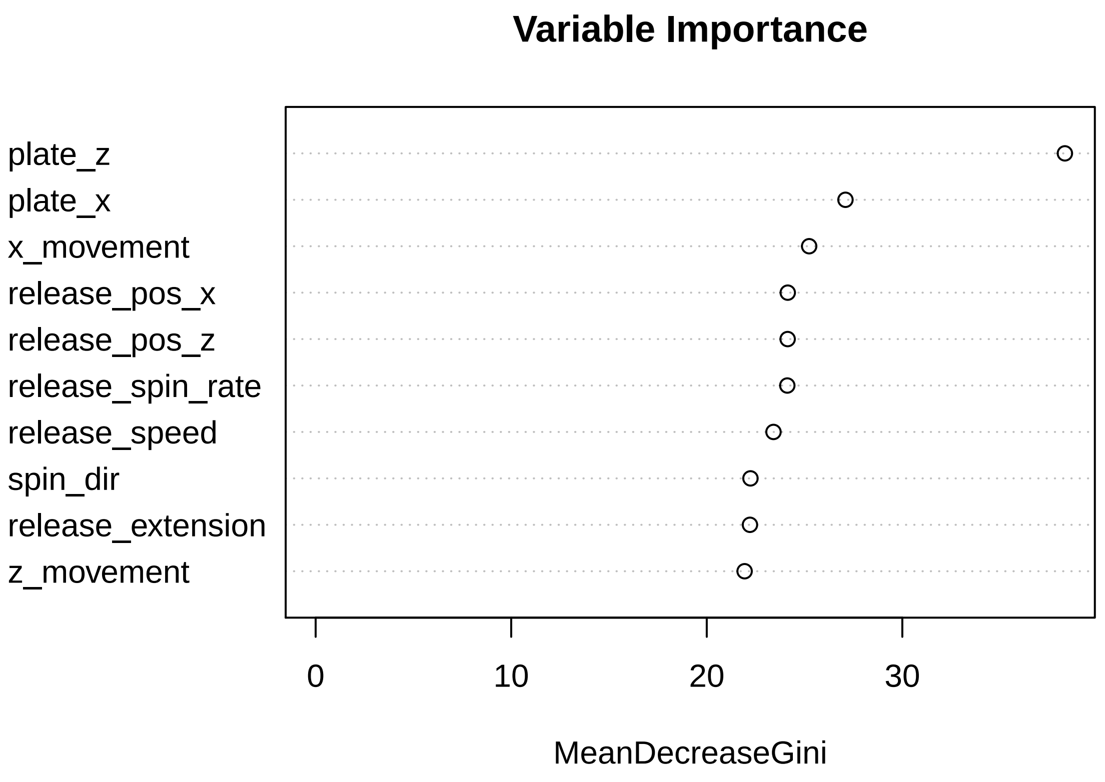
```
##             Reference
## Prediction   ChangeUp Curveball Cutter Fastball Other Slider Undefined
##   ChangeUp       1076         0      3       46     0     18         0
##   Curveball         0      1078      0        0     0     99         0
##   Cutter            1         1    557       52     0     29         0
##   Fastball        107         0    307     6701     0     28         0
##   Other             0         0      0        0     0      0         0
##   Slider            4       121     38        3     0   1506         0
##   Undefined         0         0      0        0     0      0         0
```

```
##       Accuracy          Kappa  AccuracyLower  AccuracyUpper   AccuracyNull
##      0.9272187      0.8797317      0.9223776      0.9318472      0.5776645
## AccuracyPValue  McnemarPValue
##      0.0000000            NaN
```

We see that our prediction accuracy is about 92.7%. The change up seems to be the pitch that is easiest to classify, which makes sense given that is the only offspeed pitch that breaks to the arm-side, unlike the slider, curveball, and cutter. The slider and curveball seem to be mistaken for eachother a decent amount, which makes sense because watching a game live it is sometimes difficult to tell if a pitch is a slider, curveball, or slurve. Finally, the fastball and cutter get mistaken for each other the most because they have can definitely have similar behavior (the cutter can also be called the cut-fastball). By using the random forest model, we were able to increase our accuracy and learn more about the importance of our variables compared to the multinomial logistic regression approach.

4. Create a model to predict the likelihood of a swing and miss based on the characteristics of a curveball. Create a visualization to display the most important characteristics of a curveball in recording a swing-and-miss. Explain the results of your model and visualization in 500 words or less.

```r
curveballs <- pitch.data %>%
  filter(Pitch_Type=="Curveball" & release_spin_rate > 1300)

# change the vector of outcomes to only have two values, swinging strike
# and not swinging strike, which we will call InPlay
curveballs$Pitch_Outcome <- replace(curveballs$Pitch_Outcome, curveballs$Pitch_Outcome %in%
                                      c("FoulBall","BallCalled", "StrikeCalled",
                                        "HitByPitch"), "InPlay")

# split into test and training
curveballs$Game_Date <- parse_date_time(curveballs$Game_Date, orders = c("dmy", "mdy", "ymd"))

splits <- split(curveballs, curveballs$Game_Date < as.Date("2018-07-01"))
train <- splits[1]$`FALSE` %>%
  select(-c(Game_Date))
test <- splits[2]$`TRUE` %>%
  select(-c(Game_Date))

# build model
curveballs.rf <- randomForest(factor(Pitch_Outcome) ~ release_speed + x_movement +
                           z_movement + release_spin_rate + release_pos_x +
                           release_pos_z + plate_x + plate_z + spin_dir +
                             release_extension, data=train, mtry=3, important=T)

varImpPlot(curveballs.rf, sort=T, main="Variable Importance", n.var=10)
```



Looking at the plot above of variable importance in our model for predicting the likelihood of a swing and miss on a curveball, we see that the two most important varibales are plate z and plate x. This makes sense because curveballs that are down and out of the zone are much more likely to get a swing and miss than hangers. The rest of the variables all have almost the exact same level of importance in our model. Now we predict on the test data:

```
##                   Reference
## Prediction         BallCalled FoulBall HitByPitch InPlay StrikeCalled
##   BallCalled                0        0          0      0            0
##   FoulBall                  0        0          0      0            0
##   HitByPitch                0        0          0      0            0
##   InPlay                    0        0          0   1057            0
##   StrikeCalled              0        0          0      0            0
##   StrikeSwinging            0        0          0      6            0
##   Undefined                 0        0          0      0            0
##                   Reference
## Prediction         StrikeSwinging Undefined
##   BallCalled                    0         0
##   FoulBall                      0         0
##   HitByPitch                    0         0
##   InPlay                      133         0
##   StrikeCalled                  0         0
##   StrikeSwinging                3         0
##   Undefined                     0         0
##       Accuracy          Kappa  AccuracyLower  AccuracyUpper   AccuracyNull
##     0.88407006     0.02768848     0.86458346     0.90164886     0.88657214
## AccuracyPValue  McnemarPValue
##     0.62901480            NaN
```

Our model has a prediction accuracy of 88.5% for the second half of the 2018 season. That is, we correctly classified 88.5% of the curveballs as either swing and misses or not.