How to do you project career trajectories for college WRs: What mattters

Andrew Moss

December 12, 2018; Updated Summer 2019

##Abstract: The goal of this project is to create a model for scouting receivers that is more accurate than the currently accepting methods of scouting. Right now, entering the NFL, receivers are expected to produce in accordance with their draft position. I hoped to create a model that would beat draft position to exploit a market inefficiency and find receivers that are more likely to succeed than the NFL would initially think. To model the success of draft position in predicting WR success, I used a linear model with draft round and pick. In order to create a model that works better than this one, I wanted to use athleticism and production statistics to create a random forest. From there, I would be able to put a receiver into the program and it would better model the career trajectory than the original linear model. However, because not every receiver that is drafted is invited to the combine, where the athleticism numbers are recorded, I needed to impute the data to create estimates so I would have the same inputs for every player I attempt to model. My final model was a random forest that includes the following combine drills in addition to height and weight: 40 yard dash, 10 yard dash, vertical jump, broad jump, and short shuttle. The final model also included the percentage of the production of their college offense the individual made up of in terms of yards, receptions, and touchdowns. My final random forest has 6.5 percent more predictive power than the linear model based on draft order, and also created thresholds can be further tested to eliminate receivers that may seem like they have a chance at NFL success, but based on my model likely will not

##Motication and Background My motivation for doing this project is because scouting is my passion. I have always thought it interesting that two people could watch the same college film on a player and walk away with completely disparate conclusions. I wanted to find an objective way to make a prediction for college receivers' NFL success. I also want to exploit the market because there currently isn't that much analytics in scouting beyond linear regression or comparing market share decimals. I want to create an innovation in scouting and create value by improving the process that teams use when projecting the value of college players to the NFL. Some of the major challenges associated with collecting data points for football analytics is that because the game is so continuous, the only data I can collect are snapshots of a given game or season. Also, like I highlighted in the abstract, I am not going to have the same measurements on every player. The first article I studied is Kent Lee Platte's work with relative athletic scores. He standardizes combine scores by position and assigns them a ranking from 1 to 10. With positions other than Quarterback (so Wide Receiver falls into this group) there is distinct cutoff in chance of NFL success with players that scored above 5.0 (the mean), and players that scored below 5.0. Furthermore, there was an even further assurance of success of players that scored an RAS above 8.0 (80th percentile for their position). Another article I reviewed, from Peter Howard of Dynasty league Football, showed that there was a relationship between college market share scores (What percentage of the player's college offense's production he made up) for receptions, touchdowns, and yards, and NFL success. However, in both of these studies, only a linear relationship was created for this data. I hoped to combine the work of what they were both doing and create a comprehensive metric to evaluate receivers that manifested itself in a prediction for how many yards a college receiver would accrue in his NFL career.

#Loading the Data

```
Combine_data = read.csv(file = "C:/Users/Andrew Moss/Documents/Input data to impute.csv", row.names = 1)
WRData = read.csv(file = "C:/Users/Andrew Moss/Documents/final WR project sheet.csv", row.names = 1)
```

###Exploratory Data Analysis To get my data, I used a python web scrape where I grabbed their college statistics and market share numbers. Because I am using total yards as my response variable, I wanted to collect a sample of college players where almost all of them have completed their NFL careers. Also I was limited because I could not collect college data before 2000. This led me to collecting college data on receivers

from the 2000 to 2009 seasons. I then did another scrape for NFL receiver production and matched the college data to the NFL data by name. Finally, I used Kent Lee Platte's combine database to again match the RAS scores and combine times to the receivers. After I collected the data, I needed to clean it. Because I filtered my data collection by college receiving yards, the first thing I needed to do was eliminate all of the running backs, because they had the potential to throw off the model. Then I had to clean out the receivers with too much missing combine data. Even though I used a random forest to project the missing combine scores, I knew the accuracy could be compromised by leaving them in the data set. Also those scores were missing not at random because they were either too hurt to perform or weren't deemed good enough to be invited to the combine. The receivers with some, but not all drills missing could be classified as missing at random, but not completely at random. Because the 40 yard dash and then 10 yard dash are taken in the same measurement, every receiver who was missing the 40 was also missing the 10. My ending dataset had 400 observations of 27 columns, but some were alternative response variables, such as seasons played, seasons spent as a starter, or yards, touchdowns, or receptions per season spent in the NFL. What's left are the variables I am using as predictors, minus RAS because there is very strong multicollinearity between the individual combine score and RAS, because RAS is a function of the combine scores, along with height and weight.

```
cor(WRData$Msrecs,WRData$NFLYds)
```

```
## [1] 0.1492817
```

```
cor(WRData$Msyards,WRData$NFLYds)
```

```
## [1] 0.1667546
```

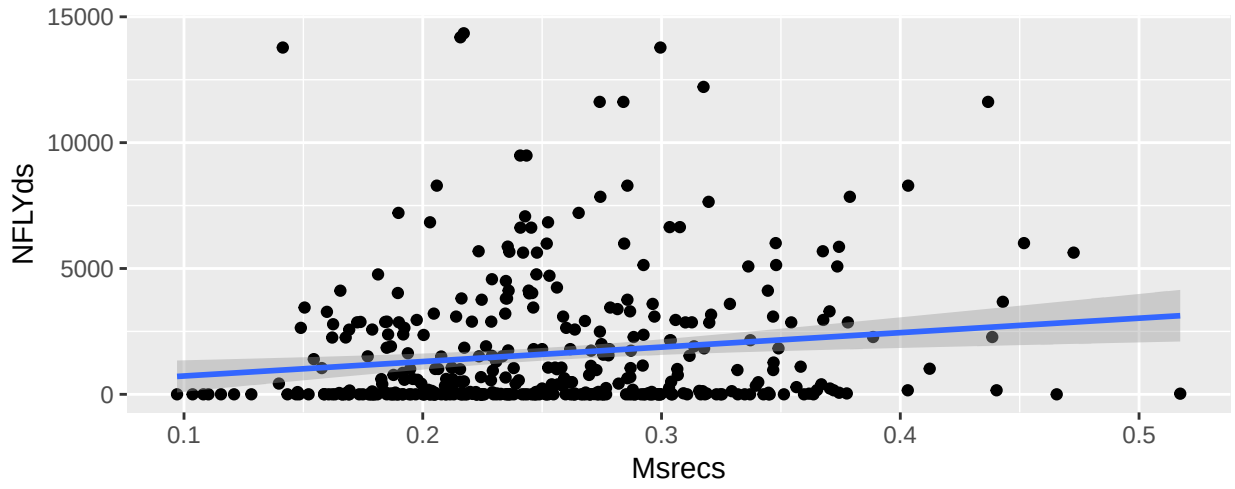
```
cor(WRData$MStds,WRData$NFLYds)
```

```
## [1] 0.162107
```

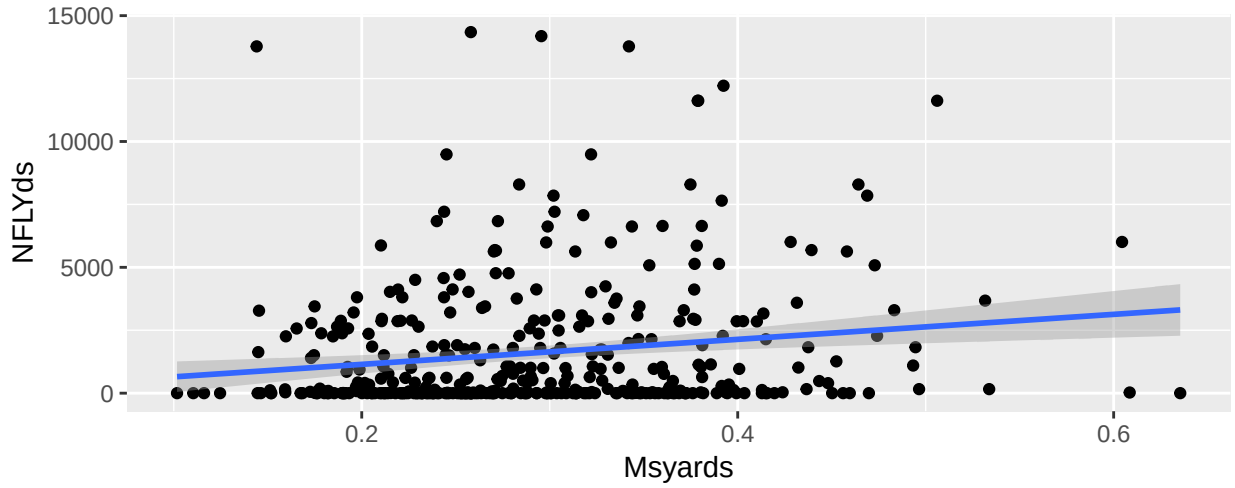
```
cor(WRData$RAS,WRData$NFLYds)
```

```
## [1] 0.237506
```

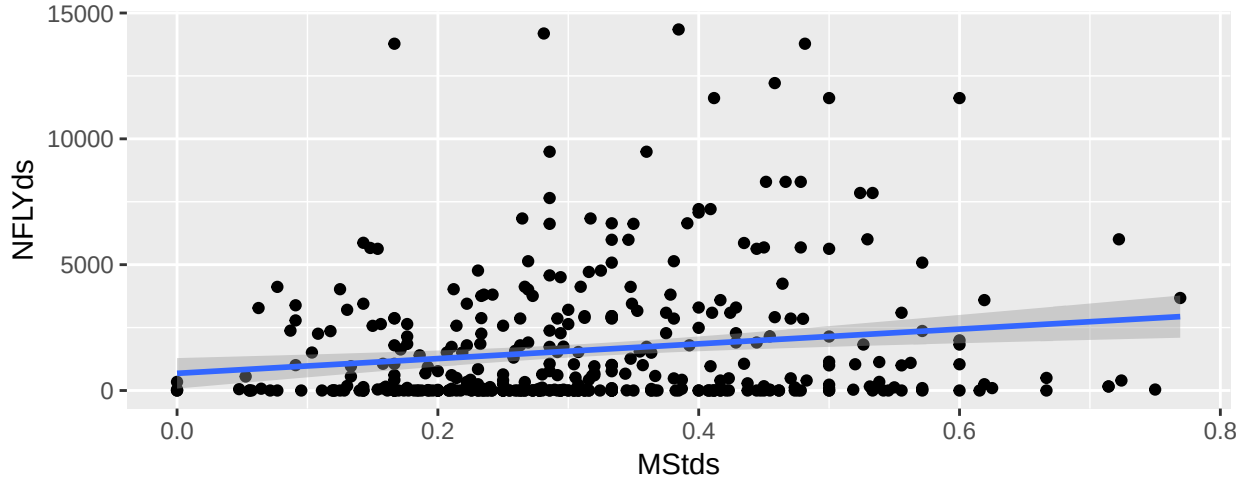
```
ggplot(WRData, aes(Msrecs,NFLYds))+geom_point()+geom_smooth(method = "lm")
```



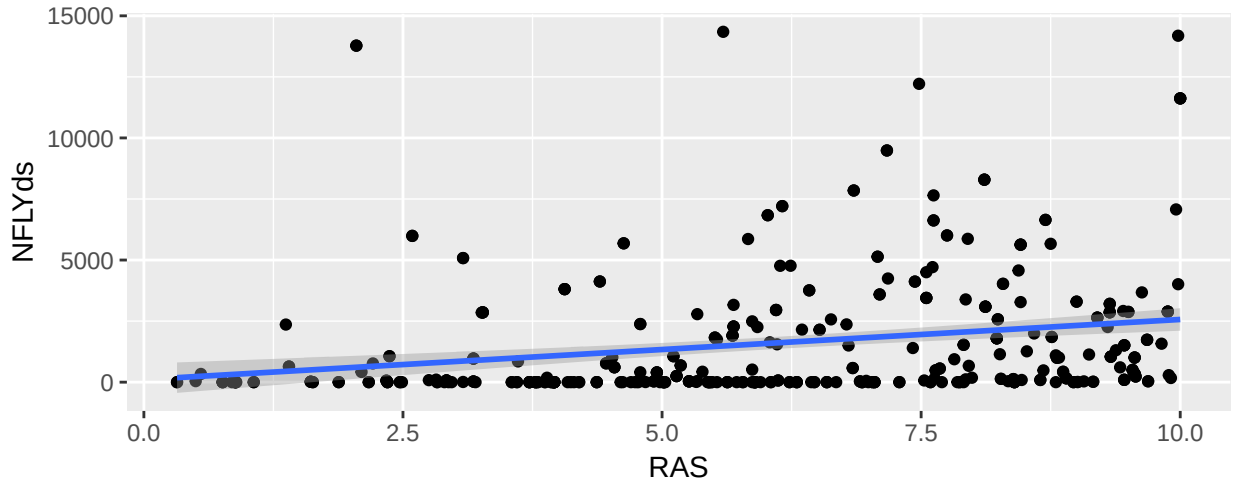

```
ggplot(WRData, aes(Msyards,NFLYds))+geom_point()+geom_smooth(method = "lm")
```




```
ggplot(WRData, aes(MStds,NFLYds))+geom_point()+geom_smooth(method = "lm")
```




```
ggplot(WRData, aes(RAS,NFLYds))+geom_point()+geom_smooth(method = "lm")
```




```
cor(log(WRData$Msrecs),WRData$NFLYds)
```

```
## [1] 0.1472444
```

```
cor(log(WRData$Msyards),WRData$NFLYds)
```

```
## [1] 0.1641728
```

```
cor(log(WRData$MStds),WRData$NFLYds)
```

```
## [1] NaN
```

```
cor(log(WRData$RAS),WRData$NFLYds)
```

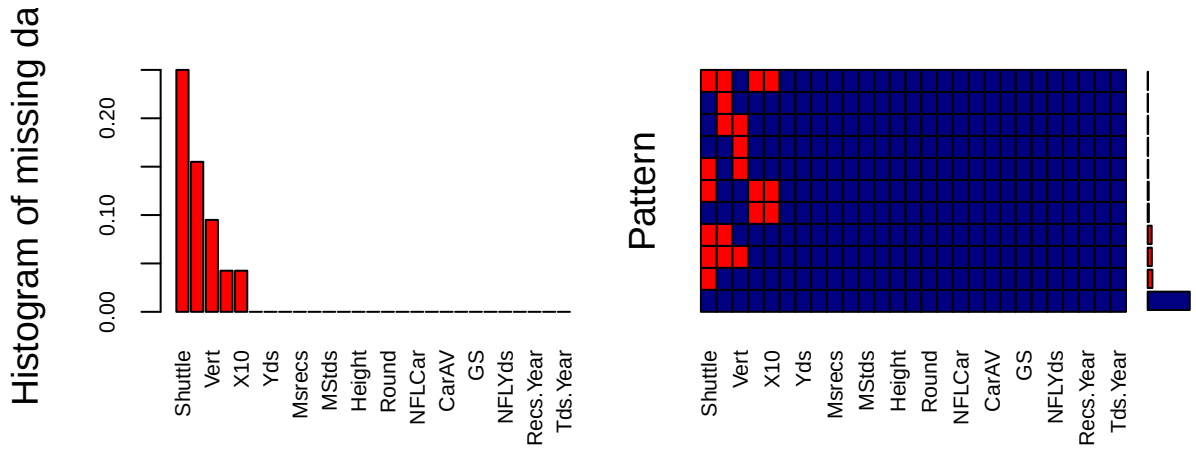
```
## [1] 0.2130368
```

```
ScoutingModel = lm(NFLYds~Round+Pick, data = WRData)
summary(ScoutingModel)
```

```
##
## Call:
## lm(formula = NFLYds ~ Round + Pick, data = WRData)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4062.0 -1082.4  -215.5   365.5 10365.0
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 4987.131    284.554  17.526  < 2e-16 ***
## Round       -931.174    108.815  -8.557 2.55e-16 ***
## Pick           5.356      1.499   3.572 0.000398 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2122 on 397 degrees of freedom
## Multiple R-squared:  0.3336, Adjusted R-squared:  0.3302
## F-statistic: 99.37 on 2 and 397 DF,  p-value: < 2.2e-16
```

```
aggr_plot <- aggr(WRData, col=c('navyblue','red'),
numbers=TRUE,
sortVars=TRUE,labels=names(WRData),
cex.axis=.7, gap=3,
ylab=c("Histogram of missing data",
"Pattern"))
```

```
## Warning in plot.aggr(res, ...): not enough vertical space to display
## frequencies (too many combinations)
```




```
##
##  Variables sorted by number of missings:
##   Variable  Count
##    Shuttle 0.2500
##      Broad 0.1550
##       Vert 0.0950
##        X40 0.0425
##        X10 0.0425
##       Recs 0.0000
##        Yds 0.0000
##        TDs 0.0000
##     Msrecs 0.0000
##    Msyards 0.0000
##      MStds 0.0000
##        RAS 0.0000
##     Height 0.0000
##     Weight 0.0000
##      Round 0.0000
##       Pick 0.0000
##     NFLCar 0.0000
##    Starter 0.0000
##      CarAV 0.0000
##         GP 0.0000
##         GS 0.0000
##    NFLRecs 0.0000
##     NFLYds 0.0000
##     NFLTDs 0.0000
##  Recs.Year 0.0000
##   Yds.Year 0.0000
##   Tds.Year 0.0000
```

I recognize that I have an uphill climb in terms of making my market share and athleticism scores explain as much variance as the scouting model. The R squared for the scouting model is .33, but none of the correlations for any of the market share metrics, or RAS are even above .25. This shows there may be a nonlinear relationship between these variables, but none of the obvious transformations make the correlations much better. I even get a NAn score for the log of MStds because there are 2 data
points of eligible college seasons where the receiver scored 0 touchdowns that year. As we can see from the log transformations, the correlations seem pretty robust to transformations and the plots show no clear

relationship. However, because of the previous research I still think I can find a relationship, I am just more confident it is nonlinear. One of the reasons that the correlations are so low is because, for both RAS and the market share numbers, there are unsuccessful NFL players on the high and low ends of those metrics. I hypthesize that the high market share guys with no NFL production may not have been athletic enough, and the high athleticism guys with no NFL production may have not had as much production in college, but were still expected to succeed based on their atheltic profile alone.

###Methods The first method I used was to get my imputations for the missing combine scores. To do this, I used a missing Forest. It uses a random Forest to project the missing data. The random Forest works by growing several trees with a number of predictors equal to a square root of the number of total predictors. Each new tree grown is made with the knowledge of the predictions of the previous trees After growing all of the trees, it feeds the other data points into the model and makes a final prediction for what the missing data point is. As an iterative process, it can do this up to maxit times, which I did not set, but it ran five of, and continues until the OOB error for the next forest is not an improvement over the OOB error for the last forest.

```
set.seed(4)
ImpModel = missForest(Combine_data,variablewise = T, ntree=100)
```

```
##   missForest iteration 1 in progress...done!
##   missForest iteration 2 in progress...done!
##   missForest iteration 3 in progress...done!
##   missForest iteration 4 in progress...done!
##   missForest iteration 5 in progress...done!
```

```
Combine_data = ImpModel$ximp
WRData[7:14] = NA
WRData[7:14]= Combine_data
```

Like I said in the EDA, I knew that a non-linear approach would be need to mold the data I have into a model that has more predictive power than the linear scouting model that I used as a baseline. I chose two non-parametric models, bagging and random forest to model the data. Bagging works by creating a bootstrap sample and fitting an unpruned tree for each sample. It then averages the trees into a single tree to make a prediction. Random Forests work in the same way but instead of using all of the predictors in each tree like bagging does, it grows trees with a square root of the number of predictors used in bagging. This creates more variety in between the trees and eliminates some of the multicollinearity between the individual trees.

```
myardsmodel.bag = randomForest(NFLYds~ Height+Weight+X40+X10+Shuttle+Vert+Broad+Msyards+MStds+Msrecs,
data = WRData, mtry = 10, importance = TRUE, ntree = 100)

myardsmodel.rf = randomForest(NFLYds~ Height+Weight+X40+X10+Shuttle+Vert+Broad+Msyards+MStds+Msrecs,
data = WRData, mtry = sqrt(10), importance = TRUE, ntree = 100)
```

###Results

```
predict.OOB <- myardsmodel.bag$predicted
OOB.bag = mean( (WRData$NFLYds - predict.OOB)^2)
OOB.bag
```

```
## [1] 4435575
```

```
predict.OOB <- myardsmodel.rf$predicted
OOB.rf = mean((WRData$NFLYds - predict.OOB)^2)
OOB.rf
```

```
## [1] 4363370
```

```
(4350432-4250863)/4250863
```

```
## [1] 0.02342324
```

Since the random Forest's OOB error estimate is 2.34 percent better than the bags, I will ultimately compare the accuracy of the random forest to the linear scouting model. I now will compare the OOB error estimate to the LOOCV estimate of the test MSE for the linear scouting model

```
mse.store = c()
n = nrow(WRData)
for(k in 1:n){
cv.test <- k
fitScoutCV <- lm(NFLYds~Pick+Round, data= WRData[-k,])
preds <- predict(fitScoutCV, newdata = WRData[k,], se = TRUE)

mse.here <- mean( (WRData[cv.test,"NFLYds"] -preds$fit)^2)
mse.store <- c(mse.store,mse.here)
}
mean(mse.store)
```

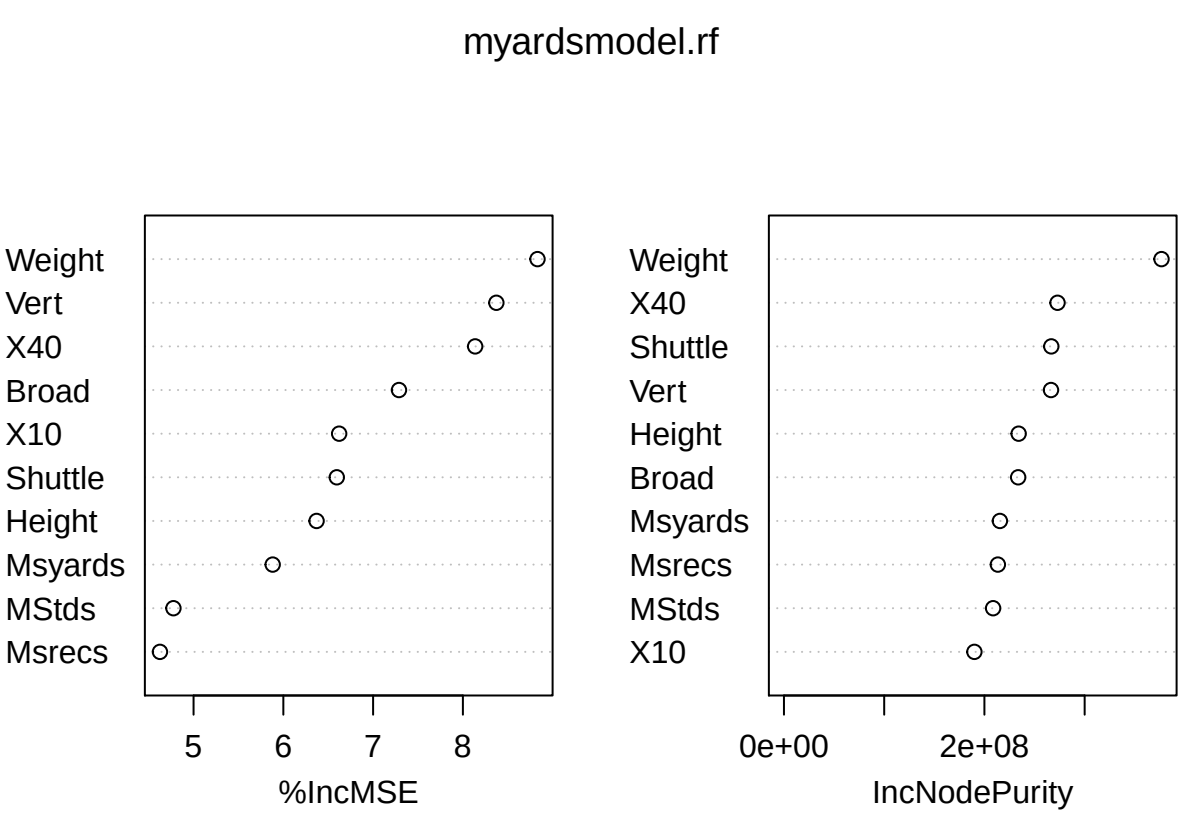
```
## [1] 4530469
```

```
(4530469-4250863)/4250863
```

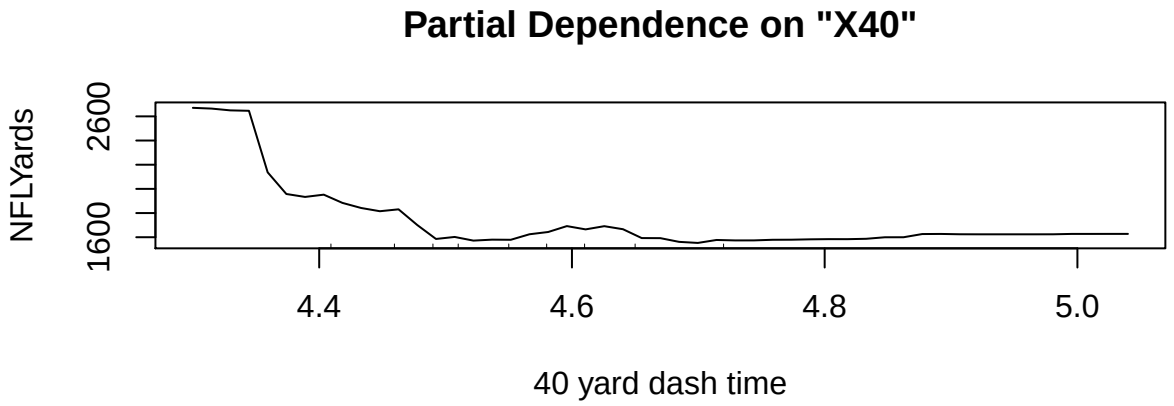
```
## [1] 0.06577629
```

The estimate of the test MSE is 4530469 or 6.5 percent worse than my random forest with the imputed combine scores. Now that I have established that my random forest model has more predictive power than the linear scouting model, I want to see what the most important predictors were within the random forest.

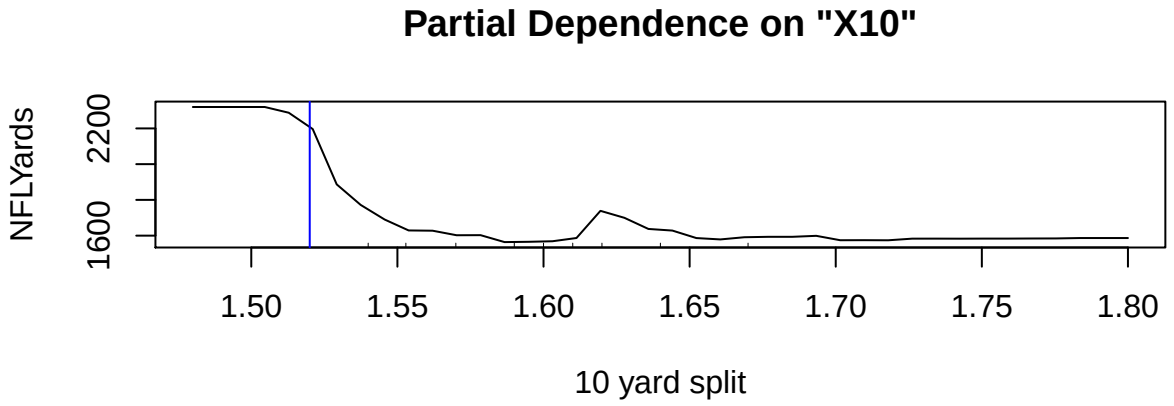
```
varImpPlot(myardsmodel.rf)
```



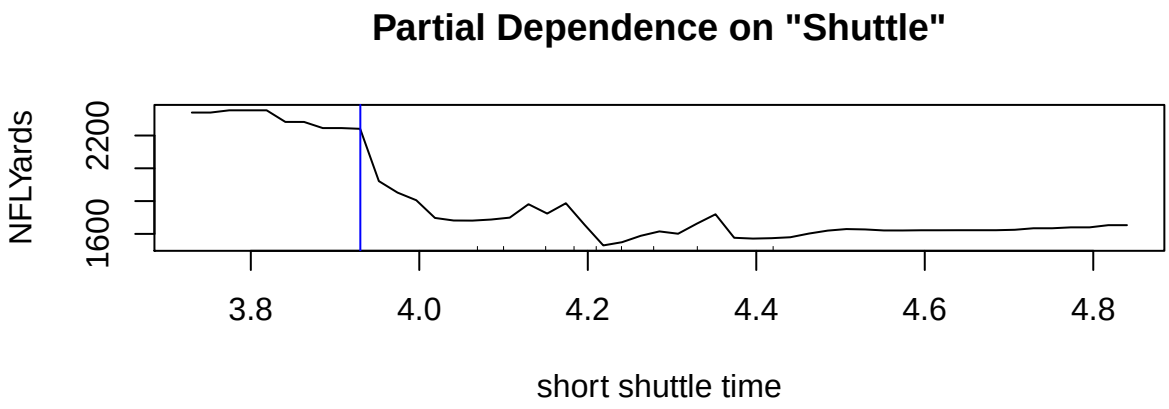
```
partialPlot(myardsmodel.rf, x.var = "X40", WRData, plot= T, xlab = "40 yard dash time", ylab = "NFLYards
```



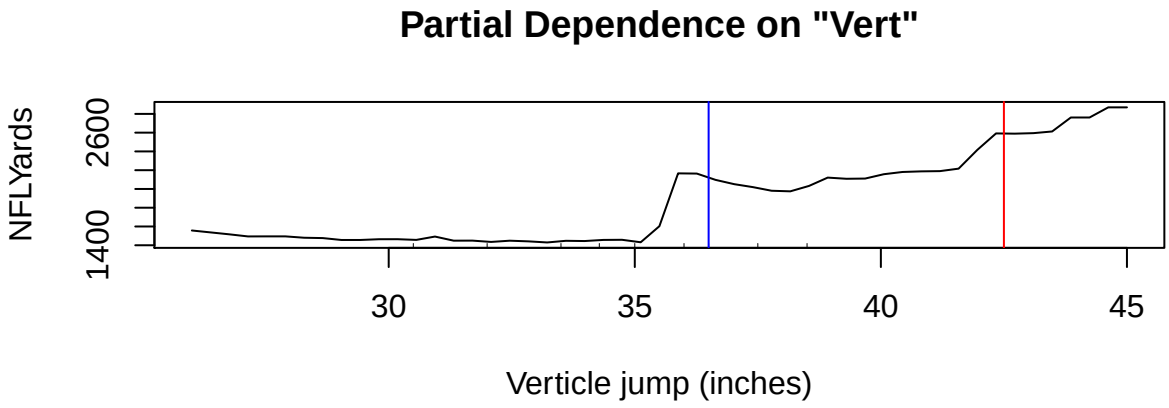

```
partialPlot(myardsmodel.rf, x.var = "X10", WRData, plot= T, xlab = "10 yard split", ylab = "NFLYards")
abline(v=1.52, col = "blue")
```



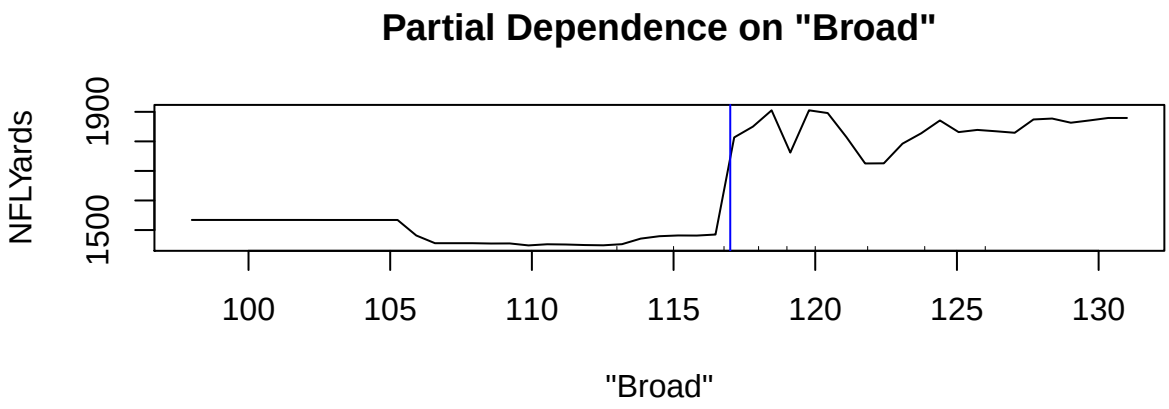

```
partialPlot(myardsmodel.rf, x.var = "Shuttle", WRData, plot= T, xlab = "short shuttle time", ylab = "NFL
abline(v=3.93, col = "blue")
```



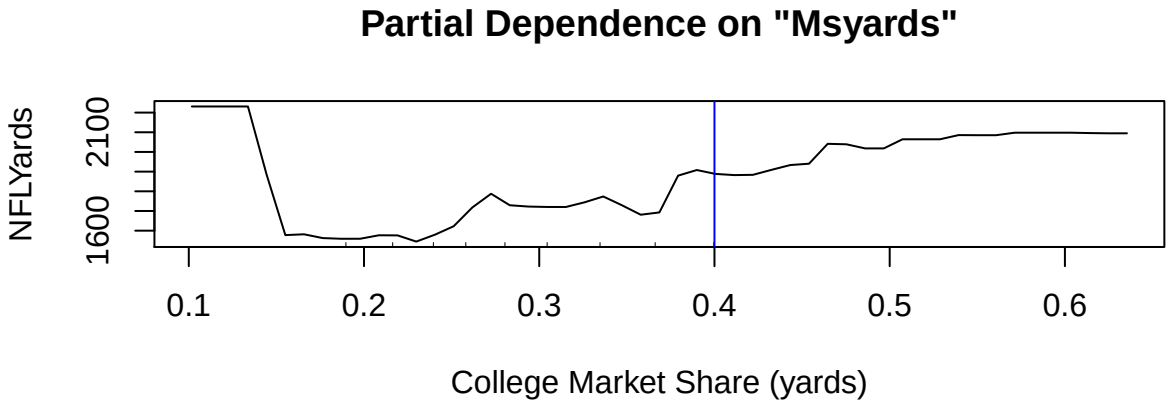

```
partialPlot(myardsmodel.rf, x.var = "Vert", WRData, plot= T, xlab = "Verticle jump (inches)", ylab = "NF
abline(v=36.5, col = "blue")
abline(v=42.5, col = "red")
```



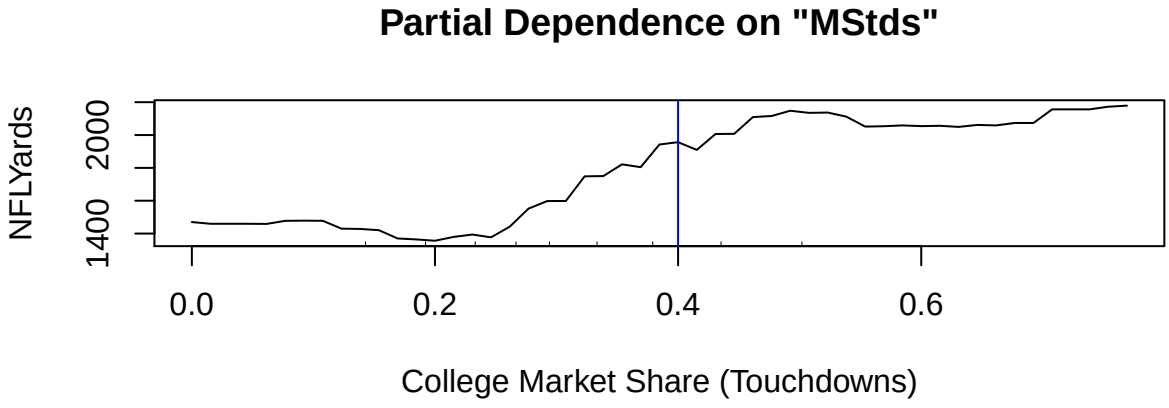

```
partialPlot(myardsmodel.rf, x.var = "Broad", WRData, plot= T, "Broad Jump (inches)", ylab = "NFLYards")
abline(v=117, col = "blue")
```



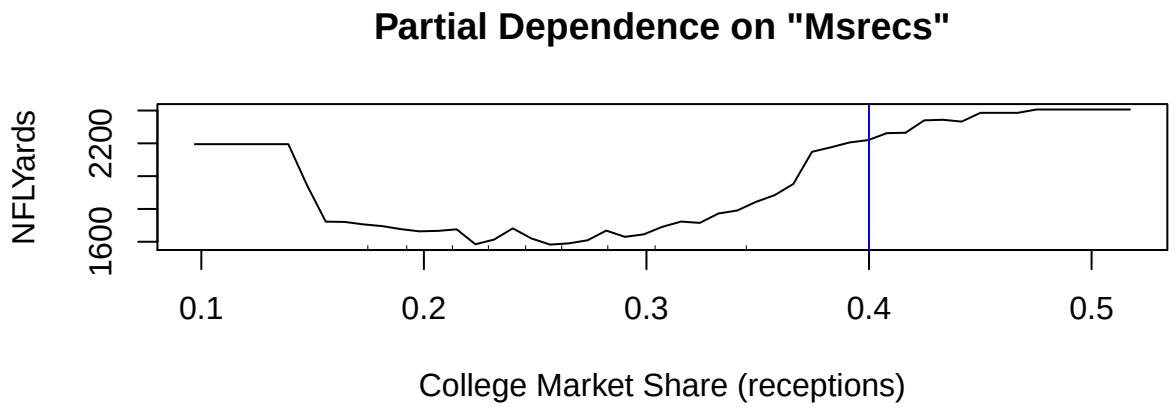

```
partialPlot(myardsmodel.rf, x.var = "Msyards", WRData, plot= T, xlab = "College Market Share (yards)", y
abline(v=.4, col = "blue")
```



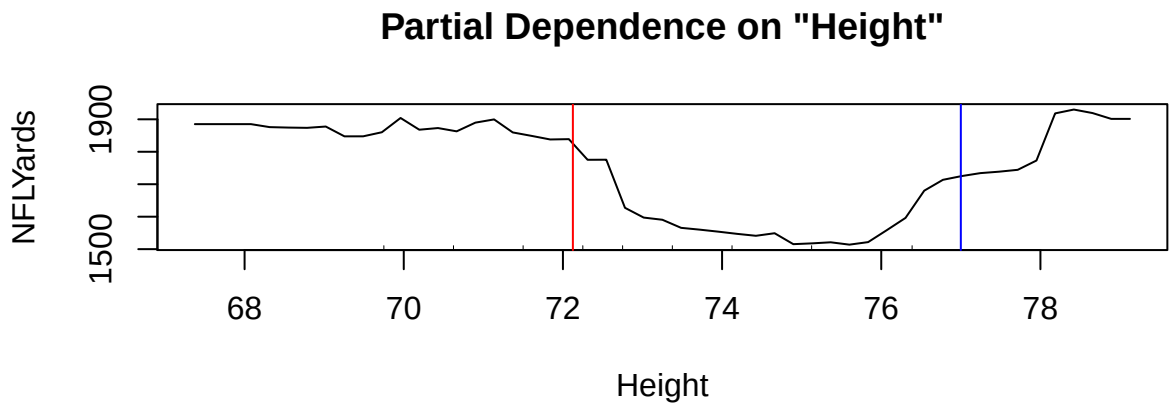

```
partialPlot(myardsmodel.rf, x.var = "MStds", WRData, plot= T, xlab = "College Market Share (Touchdowns)
abline(v=.4, col = "blue")
```



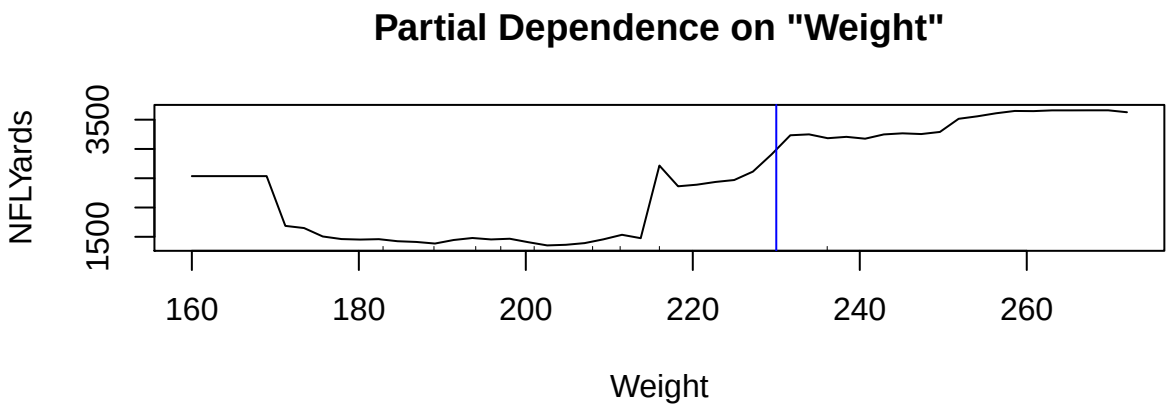
```
partialPlot(myardsmodel.rf, x.var = "Msrecs", WRData, plot= T, xlab = "College Market Share (receptions)
abline(v=.4, col = "blue")
```




```
partialPlot(myardsmodel.rf, x.var = "Height", WRData, plot= T, xlab = "Height", ylab = "NFLYards")
abline(v=72.125, col = "red")
abline(v= 77, col = "blue")
```




```
partialPlot(myardsmodel.rf, x.var = "Weight", WRData, plot= T, xlab = "Weight", ylab = "NFLYards")
abline(v = 230, col = "blue")
```



After getting the result, I ran a Variable importance plot to determine which metrics explained the most variance in the data. Since 40 yard dash time, broad jump, and weight were the only metrics in the top of both the MSE and node purity plots, I conclude that they are the most important in predicting receiver success, according to the model. For the MSE plot, there were clear discrepancies in terms of how important individual metrics were in predicting success. However, for the node purity plot, after weight, shuttle, and 40, most of the other metrics seem to score about the same.

The Partial Plots can help create scouting thresholds where, if a player is below the threshold for a given test, he is unlikely to have NFL success. The 40 yard dash actually appears to have the least of trend in terms of the partial dependence plot for the random forest that I used. However, there were sharp upticks in NFL production with those who had better than a 1.52 10 yard split. The same was true with those who had short shuttles better than 3.93 seconds, broad jumps better than 117 inches or a market share score, for any statistic that was better than .4. Vertical jump was the only plot that appeared to have 2 thresholds, as there were sharp jumps up in NFL production at 36.5 inches, and then again at 42.5 inches. In the future, it would be interesting to run a linear regression with dummy variables for each of these thresholds and see how they did in terms of predictive power, or even to eliminate receivers with a negligible chance at NFL success because they don't hit enough of the thresholds.

###Conclusion and Future Work

The goal of this project was to use statistical modeling to create a model, based on athleticism and college production, that was drastically better than traditional scouting methods or draft order at evaluating receiver prospects prior to the NFL draft. While I only found something that was 6.5 percent better at predicting the number of yards a receiver would have in his career, I gained some valuable insights in terms of what I would want to do next. It would be interesting to create a metric, based on weights from the importance plots and the thresholds I identified in the partial dependence plots, that ranked receivers based on their overall likelihood to succeed, or even outproduce their expected draft position. A limitation that this project had, that could improve the model in the future, is that I did not have age data available to me. In scouting analytics, there is a theory of "breakout age," or the age that they first eclipsed 20% in the market share for yards. A younger breakout age has been correlated to a higher likelihood of NFL success. To improve this model, it would be beneficial to include age at a certain date within the college season that the player was in. I wonder how that could change the importance plots, which would change the weights for the categorical variables that are represented by the thresholds.

###The actual future Work

```
WRcats = read.csv(file = "C:/Users/Andrew Moss/Documents/WRcats.csv", row.names = 1)

n = nrow(WRcats)
```

```r
mgeneral = lm(NFLYds~Score, data = WRcats)
summary(mgeneral)
```

```
##
## Call:
## lm(formula = NFLYds ~ Score, data = WRcats)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3658.3 -1612.9 -1032.3   827.1 13225.8
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)    553.2      230.0   2.405   0.0166 *
## Score          565.4      104.0   5.435 9.58e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2504 on 398 degrees of freedom
## Multiple R-squared:  0.06909,    Adjusted R-squared:  0.06675
## F-statistic: 29.54 on 1 and 398 DF,  p-value: 9.577e-08
```

```r
cor(mgeneral$fitted.values, WRcats$NFLYds)
```

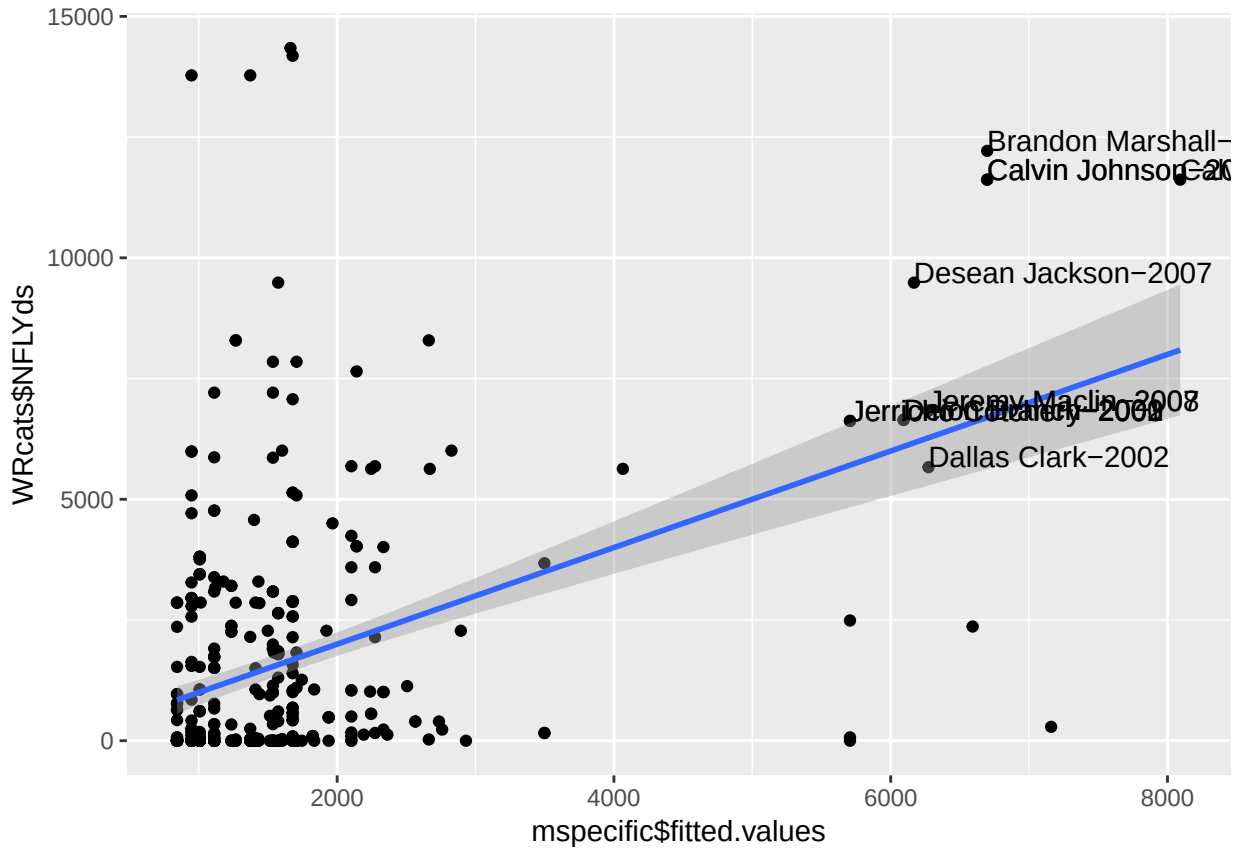
```
## [1] 0.2628466
```

```r
mspecific = lm(NFLYds ~ height.6001 + height.6050+ X10.1.52+shuttle.3.93+ vert.42..36.5+broad.117+MSRecs
summary(mspecific)
```

```
##
## Call:
## lm(formula = NFLYds ~ height.6001 + height.6050 + X10.1.52 +
##     shuttle.3.93 + vert.42..36.5 + broad.117 + MSRecs..4 + MSYardss..4 +
##     MSTDs..4, data = WRcats)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -6872.3 -1224.1  -915.3   777.9 12831.2
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)      947.8      249.7   3.796  0.00017 ***
## height.6001     -104.7      254.5  -0.412  0.68092
## height.6050      287.8      586.9   0.490  0.62415
## X10.1.52         656.0      607.4   1.080  0.28085
## shuttle.3.93    4594.3      596.8   7.699 1.14e-13 ***
## vert.42..36.5    566.9      231.9   2.445  0.01492 *
## broad.117        163.9      260.7   0.629  0.52997
## MSRecs..4       1224.2      842.9   1.452  0.14720
## MSYardss..4      170.9      493.7   0.346  0.72938
## MSTDs..4         424.2      299.6   1.416  0.15758
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2364 on 390 degrees of freedom
```

```
## Multiple R-squared:  0.1873, Adjusted R-squared:  0.1685
## F-statistic: 9.986 on 9 and 390 DF,  p-value: 7.812e-14
```

```
mspecific_vals = data.frame(mspecific$fitted.values, WRcats$NFLYds)

catplot = ggplot(mspecific_vals, aes(x=mspecific$fitted.values, y= WRcats$NFLYds))+geom_point()+geom_sm

plot(catplot)
```




```
cor(mspecific$fitted.values, WRcats$NFLYds)
```

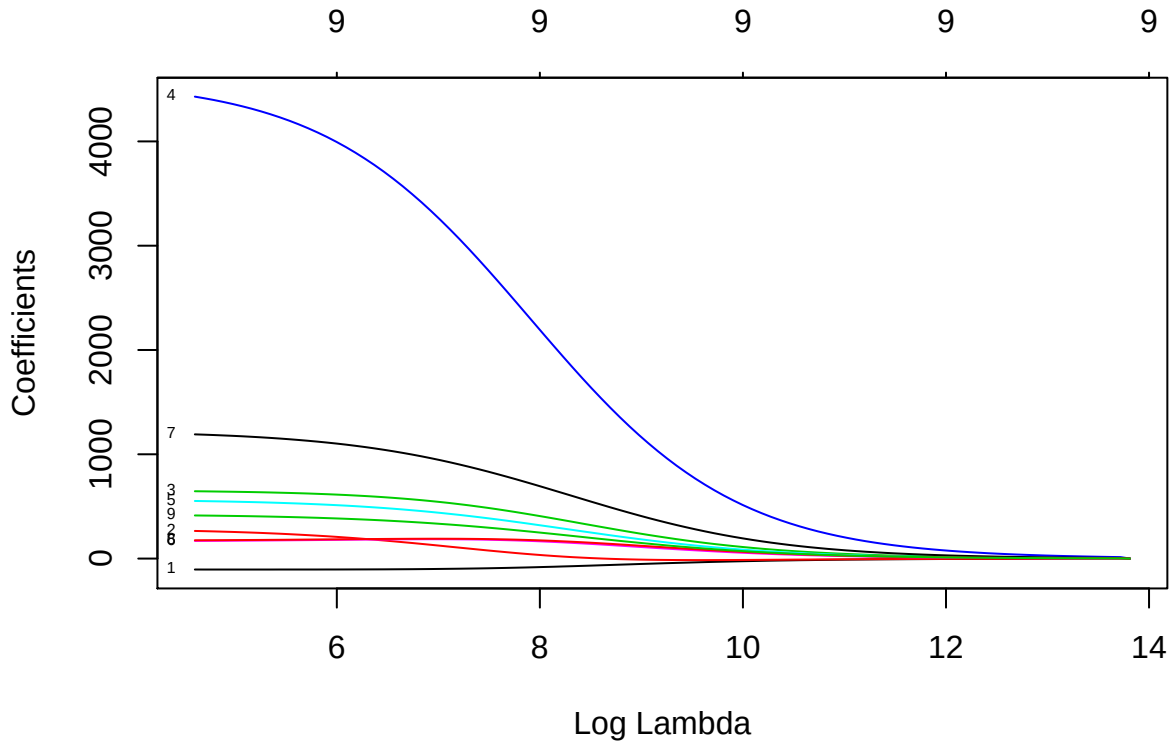
```
## [1] 0.4327646
```

```
specific.fits = c(mspecific$fitted.values)
actuals = c(WRcats$NFLYds)
specific.preds = cbind(specific.fits,actuals)
write.csv(specific.preds, file = "specific_preds.csv")


mse.store = c()
n = nrow(WRData)
for(k in 1:n){
cv.test <- k
fitspecificCV <- lm(NFLYds~ height.6001 + height.6050+ X10.1.52+shuttle.3.93+ vert.42..36.5+broad.117+M
preds <- predict(fitspecificCV, newdata = WRcats[k,], se = TRUE)

mse.here <- mean( (WRcats[cv.test,"NFLYds"] -preds$fit)^2)
mse.store <- c(mse.store,mse.here)
```

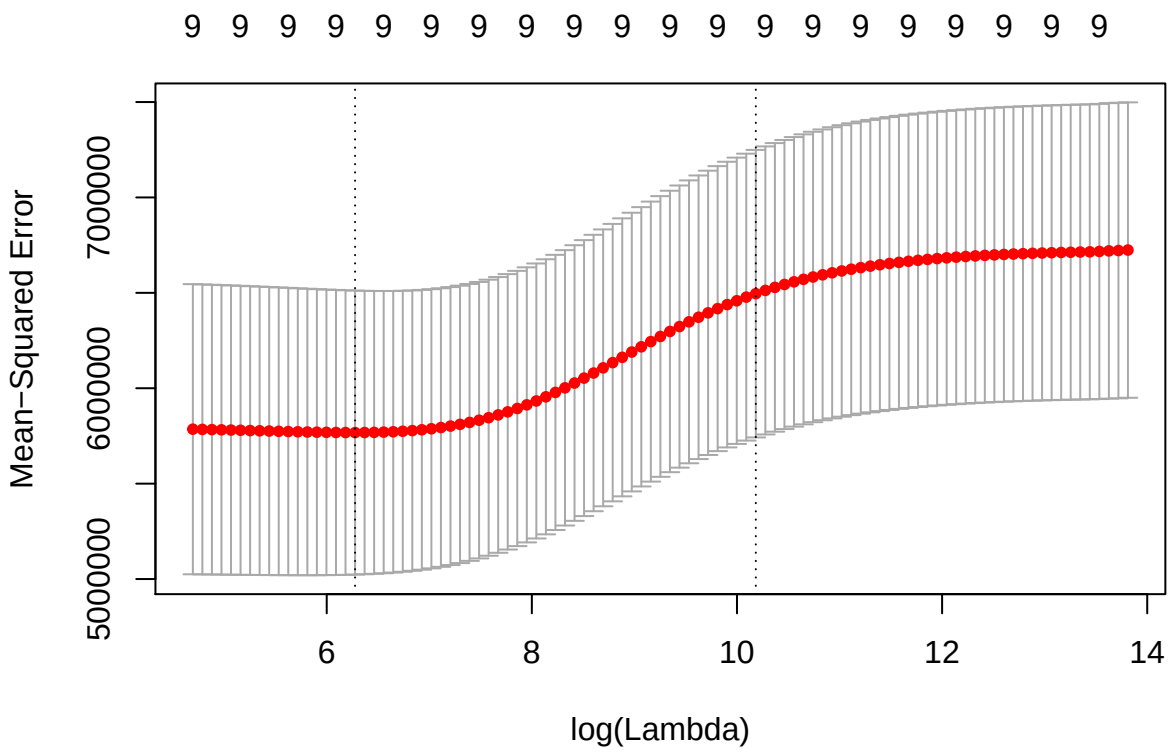
```
}
mean(mse.store)
```

```
## [1] 5847660
```

```
xinfo <- model.matrix(NFLYds~height.6001 + height.6050+ X10.1.52+shuttle.3.93+ vert.42..36.5+broad.117+l


ridge.mod <- glmnet(xinfo, WRcats$NFLYds, alpha =0 , standardize = TRUE)
plot(ridge.mod, xvar="lambda", label =TRUE)
```




```
cv.out <- cv.glmnet(xinfo, WRcats$NFLYds, alpha = 0 )
plot(cv.out)
```




```
cv.out$lambda.min
```

```
## [1] 531.7889
```

```
mse.min <- cv.out$cvm[cv.out$lambda == cv.out$lambda.min]
mse.min
```

```
## [1] 5767602
```

```
best.ridge.mod = ridge.mod <- glmnet(xinfo, WRcats$NFLYds, alpha =0 , lambda =531.7889,  standardize = 
coef(best.ridge.mod)
```

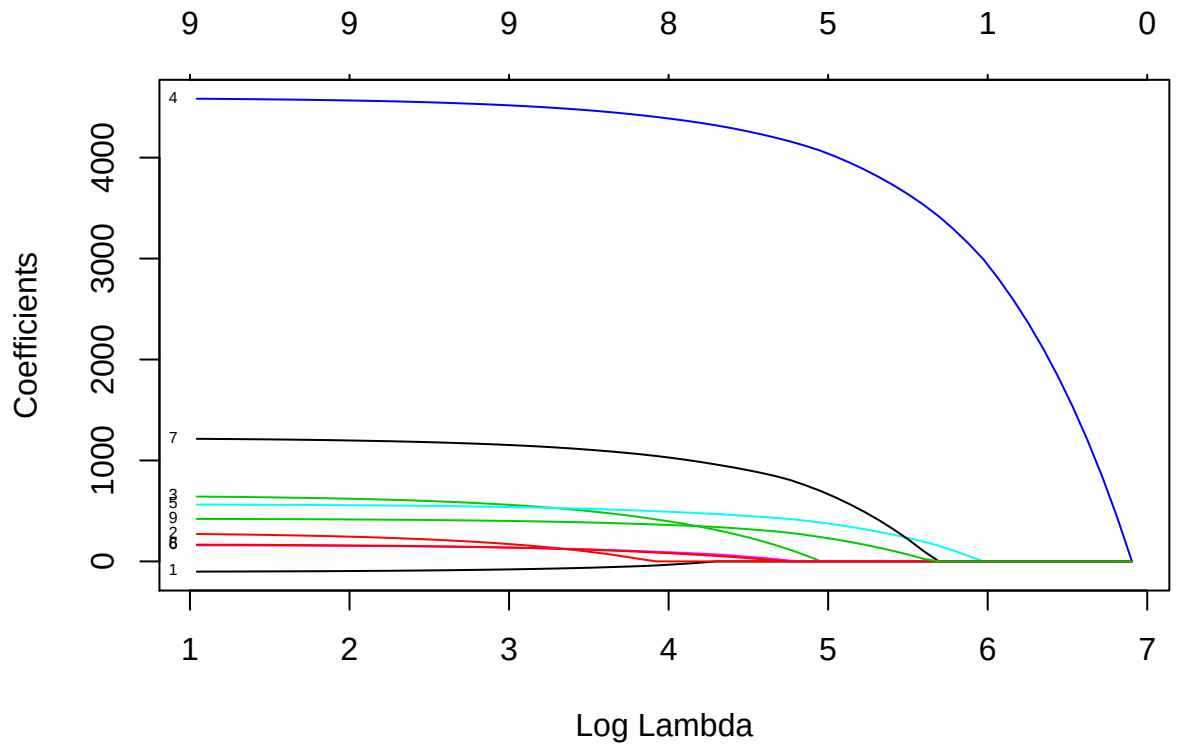
```
## 10 x 1 sparse Matrix of class "dgCMatrix"
##                       s0
## (Intercept)   1014.5364
## height.6001   -103.9071
## height.6050    188.3336
## X10.1.52       599.7854
## shuttle.3.93  3834.9485
## vert.42..36.5  496.8074
## broad.117      182.0898
## MSRecs..4     1069.5268
## MSYardss..4    186.6237
## MSTDs..4       374.0080
```

```
intercept = 1014

height.6001.slope = -103.9071
height.6050.slope = 188.3336
```

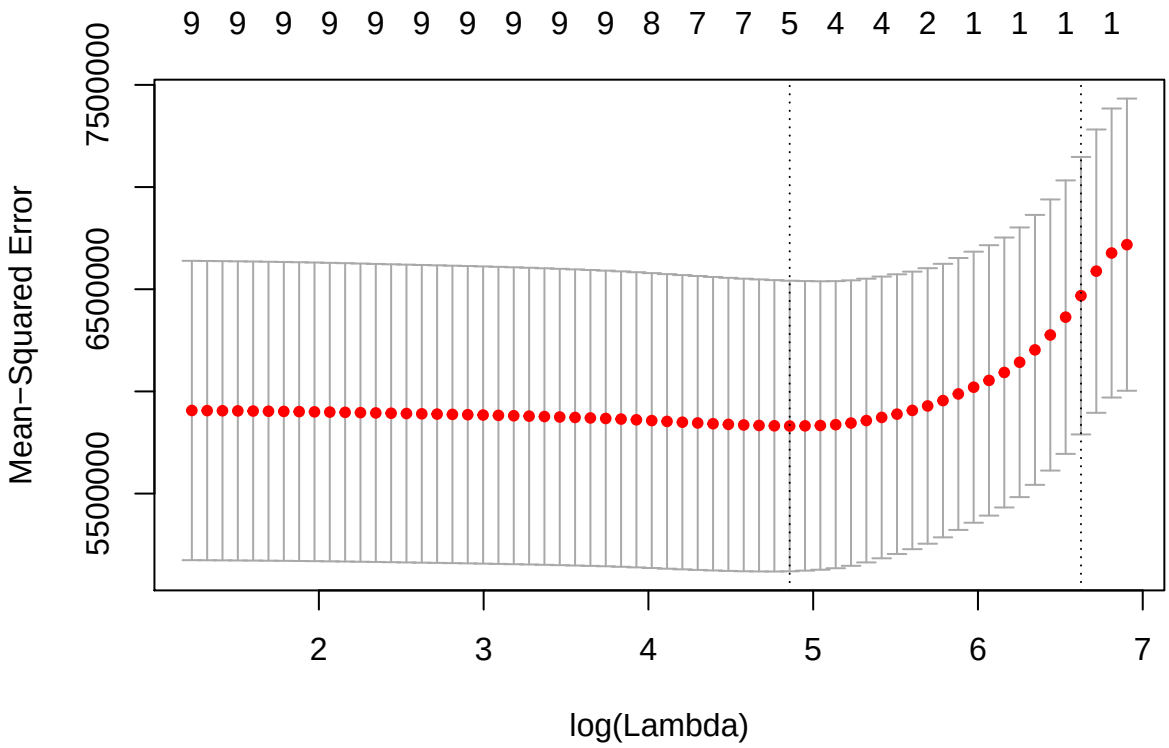
```
X10.1.52.slope     =  599.7854
shuttle.3.93.slope =  3834.9485
vert.42..36.5.slope = 496.8074
broad.117.slope     = 182.0898
MSRecs..4.slope     = 1069.5268
MSYardss..4.slope   = 186.6237
MSTDs..4.slope      = 374.0080

ridge.preds = c()
i = 1

for(i in 1:400){
  temp = WRcats$height.6001[i]*height.6001.slope + WRcats$height.6050[i]*height.6050.slope+ WRcats$X10.
  ridge.preds[i]=temp
}

WRcats = cbind(WRcats, ridge.preds)
cor(WRcats$ridge.preds, WRcats$NFLYds)
```

```
## [1] 0.4324244
```

```
xinfo <- model.matrix(NFLYds~height.6001 + height.6050+ X10.1.52+shuttle.3.93+ vert.42..36.5+broad.117+


lasso.mod <- glmnet(xinfo, WRcats$NFLYds, alpha =1 , standardize = TRUE)
plot(lasso.mod, xvar="lambda", label =TRUE)
```



```
cv.out <- cv.glmnet(xinfo, WRcats$NFLYds, alpha = 1 )
plot(cv.out)
```




```
cv.out$lambda.min
```

```
## [1] 128.6997
```

```
mse.min <- cv.out$cvm[cv.out$lambda == cv.out$lambda.min]
mse.min
```

```
## [1] 5830602
```

```
best.lasso.mod = ridge.mod <- glmnet(xinfo, WRcats$NFLYds, alpha =1 , lambda =88.70774,  standardize = 
coef(best.lasso.mod)
```

```
## 10 x 1 sparse Matrix of class "dgCMatrix"
##                         s0
## (Intercept)   1115.50488
## height.6001      .
## height.6050      .
## X10.1.52       242.98912
## shuttle.3.93  4263.87850
## vert.42..36.5  451.87408
## broad.117       47.35919
## MSRecs..4      906.45259
## MSYardss..4     34.61438
## MSTDs..4       320.69831
```

```
intercept = 1115.5048


X10.1.52.slope     =  599.7854
shuttle.3.93.slope =  4263.87850
vert.42..36.5.slope = 451.87408
broad.117.slope     = 47.35919
MSRecs..4.slope     = 906.45259
MSYardss..4.slope   = 34.61438
MSTDs..4.slope      = 320.69831

lasso.preds = c()
i = 1

for(i in 1:400){
  temp =  WRcats$X10.1.52[i]*X10.1.52.slope  + WRcats$shuttle.3.93[i]*shuttle.3.93.slope+ WRcats$vert.4
  lasso.preds[i]=temp
}
WRcats = cbind(WRcats, lasso.preds)
cor(WRcats$lasso.preds, WRcats$NFLYds)
```

```
## [1] 0.429444
```

##Preds for 2019 Class

```
Draft2019 = read.csv(file ="2019 WR class.csv", row.names = 1)

set.seed(4)
ImpModel = missForest(Draft2019,variablewise = T, ntree=100)
```

```
##   removed variable(s) 11 due to the missingness of all entries
##   missForest iteration 1 in progress...done!
##   missForest iteration 2 in progress...done!
##   missForest iteration 3 in progress...done!
```

```
Full2019 = ImpModel$ximp

Draft2019[1:10]=Full2019

Preds2019 = predict(myardsmodel.rf,Draft2019, type ="response")
```

###Works Cited Howard, Peter. “Adjusting Draft Capital with College Production.” Dynasty League Football, 25 Aug. 2018, dynastyleaguefootball.com/2018/08/26/adjusting-draft-capital-with-college-production/.

Platte, Kent Lee. “Does Athleticism Correlate to NFL Success for Running Backs?” RAS, 21 May 2017, relativeathleticscores.com/2017/05/21/does-athleticism-correlate-to-nfl-success-for-running-backs/.

Underworld, Roto. “NFL Player Profile Advanced Fantasy Football Stats-Metrics Terms.” PlayerProfiler, www.playerprofiler.com/terms-glossary/.