PSTAT 131 Final Project

Joe Kinderman (4129896) Atziry Madrigal (5900386) Austin Miles (4472031)

March 12, 2020

Introduction

The NFL draft is an annual event that gives professional football teams the opportunity to improve their roster by recruiting the college football players whom they consider the most talented. The NFL draft is a large event with great media attention, including people whose fame arose from being able to project the draft. Being able to know whether you will be drafted or not allows a player to make an educated decision on potentially declaring for the draft. Additionally, agents want to target players that are likely to be drafted, as they will receive a percentage of their NFL salary. Whereas, undrafted players may not produce any value for the agent. If an NFL team knows which players are going to be drafted, they have more information to develop a strategy to maximize their value from the draft. Being able to know if an athlete will be drafted is vital information for players, agents, teams, and the media.

In this project, we apply supervised machine learning techniques to predict the probability collegiate quarterbacks are drafted into the NFL. These techniques are based on classification methods such as decision trees, random forest, bootstrap aggregating, K-Nearest Neighbors, and logistic regression models. Although our focus is on the quarterback position, our methods can ultimately be applied to any other positions as well. The goal of our project is to simply develop a model that can determine the attributes that largely define a player's draft status Overall, the importance behind our project lies in the decision-making since it would help narrow down the players that will have the biggest impact in the NFL upon leaving college.

Data

The data was sourced from sports-reference.com, using a quarterback's passing statistics in their final year of Division 1 college football and cross referencing with their draft status. This data is available to use as per the website's sharing policy. The accumulated data consists of 1036 observations with no missing values. In the data, there are a total of 12 predictors variables: 1 categorical and 11 numerical. Not all quarterbacks play the same level of competition and each conference has a varying degree of difficulty. As a proxy for level of difficulty we created an additional predictor: power5. This is a binary variable stating whether a quarterback played in an esteemed conference which typically face more difficult opponents. Over the timespan of our data conferences have realigned so we have assigned the quarterbacks power5 for playing in these conferences: "Big 12","Pac-10","Pac-12" "Big Ten", "ACC", "SEC", and "Big East". Additionally, Notre Dame quarterbacks are determined power5 for playing in particularly esteemed and difficult matches regardless of the school's conference status. Finally, we converted interception and touchdown statistics to a percentage by dividing by the number of passing attempts. This better demonstrates a quarterback's talent than the raw statistic. The data also includes the players' names and schools which serve as identifiers rather than predictors. Our targeted response variable is a binary value of draft status.

```
## [1] 1036   15
```

```
## Player School   Conf      G    Cmp    Att    Pct    Yds     YA    AYA     TD
##      0      0      0      0      0      0      0      0      0      0      0
##    Int   Rate   Year  Draft
##      0      0      0      0
```

```
## # A tibble: 2 x 6
##   Draft Yards `Completion %`    TD `Yards per attempt` `Passer Rating`
##   <dbl> <dbl>          <dbl> <dbl>               <dbl>           <dbl>
## 1     0 2298.           57.9  15.8                7.08            127.
## 2     1 3213.           62.1  25.0                8.01            145.
```

Thw table above shows that there is a significant difference in the statistics between undrafted and drafted quarterbacks. This indicates that it may be possible to differentiate which quarterbacks will be drafted.

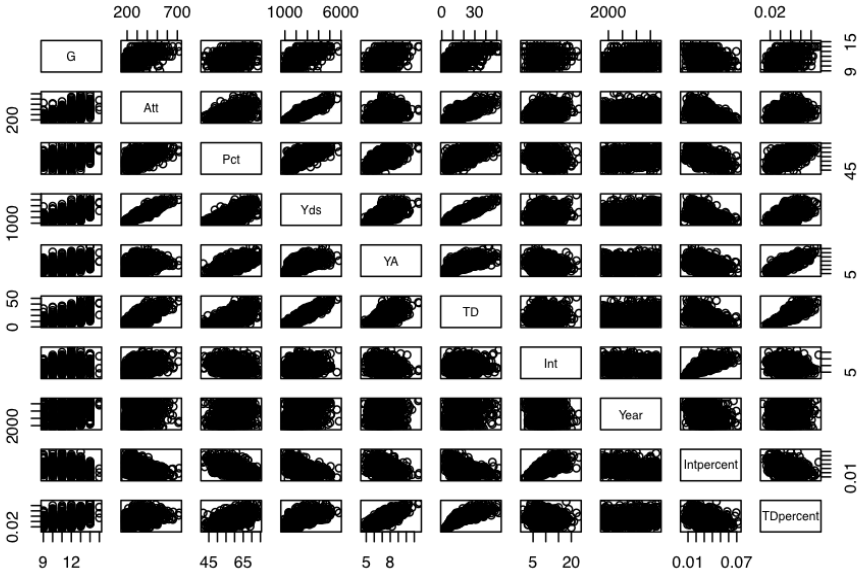
```
## # A tibble: 15 x 6
##    Conf     Yards `Completion %`    TD `Yards per attempt` `% Drafted`
##    <chr>    <dbl>          <dbl> <dbl>               <dbl>       <dbl>
##  1 ACC      2482.           58.9  16.7                7.29        22.5
##  2 American 2878            60.9  20.6                7.65        17.4
##  3 Big 12   2709.           59.0  19.9                7.45        25.7
##  4 Big East 2274.           58.5  16.6                7.38        27.5
##  5 Big Ten  2441.           59.0  17.7                7.33        30
##  6 Big West 2289.           53.1  19.5                7.12         0
##  7 CUSA     2549.           58.9  18.6                7.23        13.3
##  8 Ind      2440.           56.5  17                  6.93        17.9
##  9 MAC      2430.           58.1  17.3                6.90        12.0
## 10 MWC      2429.           59.4  17.2                7.29        15.8
## 11 Pac-10   2470.           57.0  17.7                7.31        41.8
## 12 Pac-12   3004.           62.7  22.5                7.66        37.5
## 13 SEC      2432.           58.8  17.7                7.55        31.9
## 14 Sun Belt 2333.           59.4  14.6                7.07         3.03
## 15 WAC      2456.           58.7  17.3                7.09        15
```

This table above shows a significant differnce in percent drafted by conference despite similar passing averages. Showing that the difficulty and prestige of a conference strongly affect a quarterbacks chances of being drafted.

```
##                  G     Cmp     Att     Pct     Yds      YA     AYA      TD
## G           1.0000  0.5368  0.5311  0.3553  0.5896  0.3868  0.4223  0.5345
## Cmp         0.5368  1.0000  0.9661  0.6679  0.9395  0.3693  0.4650  0.7993
## Att         0.5311  0.9661  1.0000  0.4674  0.9091  0.2435  0.3415  0.7469
## Pct         0.3553  0.6679  0.4674  1.0000  0.6271  0.5957  0.6491  0.5978
## Yds         0.5896  0.9395  0.9091  0.6271  1.0000  0.6110  0.6643  0.8917
## YA          0.3868  0.3693  0.2435  0.5957  0.6110  1.0000  0.9393  0.6526
## AYA         0.4223  0.4650  0.3415  0.6491  0.6643  0.9393  1.0000  0.7434
## TD          0.5345  0.7993  0.7469  0.5978  0.8917  0.6526  0.7434  1.0000
## Int         0.1193  0.2515  0.3507 -0.1255  0.2119 -0.1632 -0.3544  0.1174
## Rate        0.4335  0.5430  0.3869  0.7910  0.6981  0.9266  0.9662  0.7898
## Year        0.2880  0.2562  0.2148  0.2847  0.2247  0.1163  0.1875  0.1889
## Intpercent -0.2750 -0.4335 -0.3711 -0.4592 -0.4370 -0.3466 -0.6177 -0.4091
## TDpercent   0.3640  0.3761  0.2824  0.4992  0.5511  0.7679  0.8301  0.8257
##                 Int    Rate    Year Intpercent TDpercent
## G           0.1193  0.4335  0.2880    -0.2750    0.3640
## Cmp         0.2515  0.5430  0.2562    -0.4335    0.3761
## Att         0.3507  0.3869  0.2148    -0.3711    0.2824
## Pct        -0.1255  0.7910  0.2847    -0.4592    0.4992
## Yds         0.2119  0.6981  0.2247    -0.4370    0.5511
## YA         -0.1632  0.9266  0.1163    -0.3466    0.7679
## AYA        -0.3544  0.9662  0.1875    -0.6177    0.8301
## TD          0.1174  0.7898  0.1889    -0.4091    0.8257
## Int         1.0000 -0.2323 -0.1202     0.7004   -0.1094
```

```
## Rate       -0.2323  1.0000  0.2076    -0.5203    0.8572
## Year       -0.1202  0.2076  1.0000    -0.2720    0.1116
## Intpercent  0.7004 -0.5203 -0.2720     1.0000   -0.3216
## TDpercent  -0.1094  0.8572  0.1116    -0.3216    1.0000
```

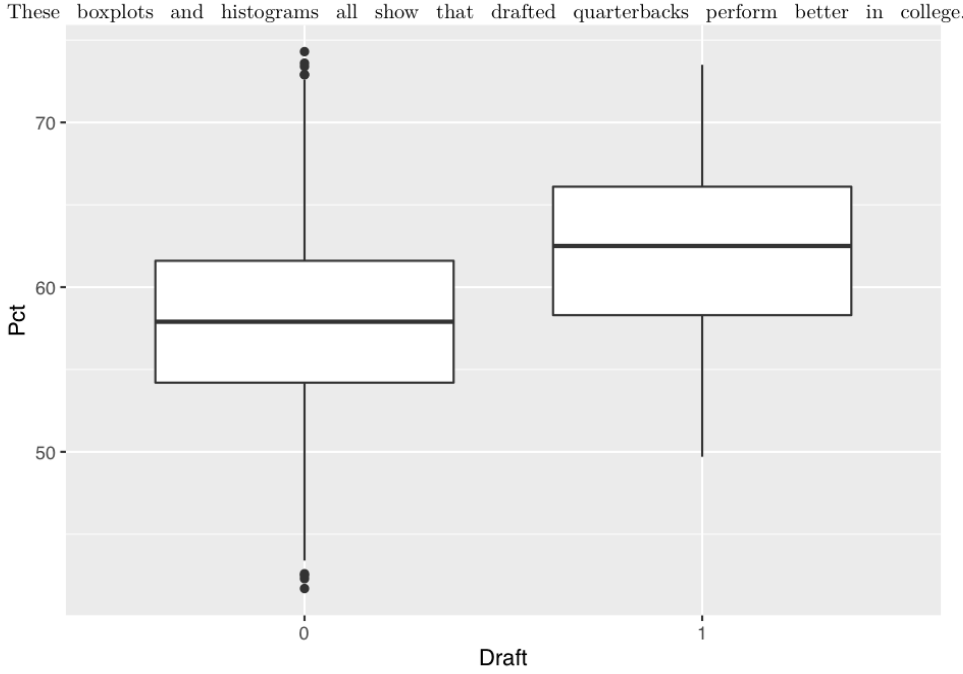
Utilizing the correlation function in R, we found that there were issues with multicollinearity. As a result, we excluded Cmp (number of completions), AYA (adjusted yards per attempts), and Rate (passer rating). It is also important to note that although yards has a high correlation rate with attempts and touchdowns, this is due to the nature of the game where more attempts give you the opportunity for more yards, in turn bringing you closer to the endzone. So although these variables are coordinated, they each display a distinct ability representing a quarterback's talent, giving them predictive power. From here on out we simply focus on a total of 10 predictor variables.

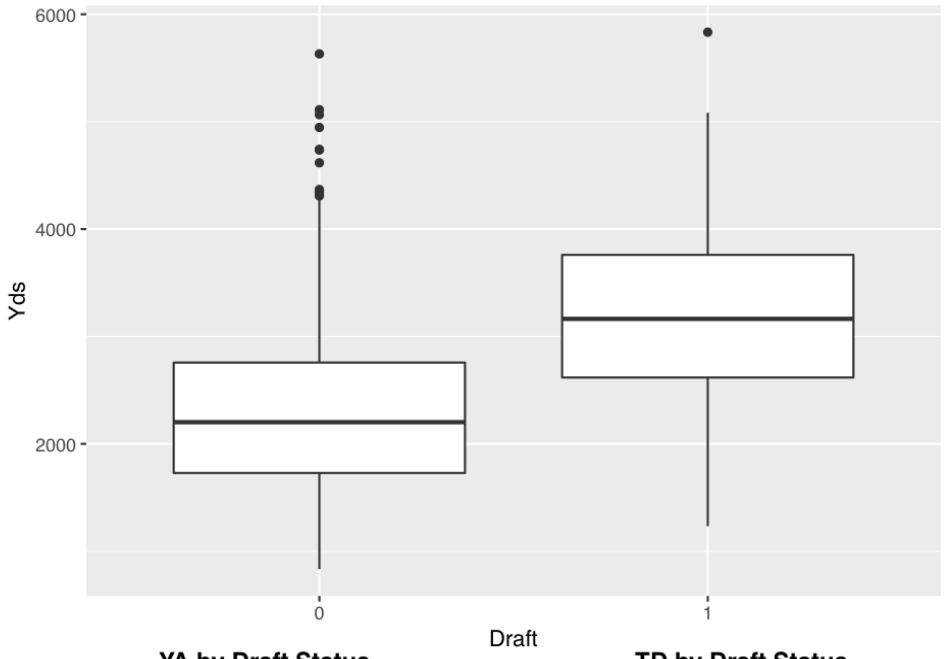


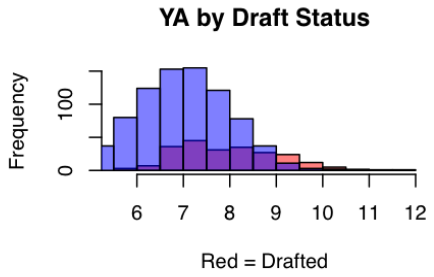

```
##                  G     Att     Pct     Yds      YA      TD     Int    Year
## G           1.0000  0.5311  0.3553  0.5896  0.3868  0.5345  0.1193  0.2880
## Att         0.5311  1.0000  0.4674  0.9091  0.2435  0.7469  0.3507  0.2148
## Pct         0.3553  0.4674  1.0000  0.6271  0.5957  0.5978 -0.1255  0.2847
## Yds         0.5896  0.9091  0.6271  1.0000  0.6110  0.8917  0.2119  0.2247
## YA          0.3868  0.2435  0.5957  0.6110  1.0000  0.6526 -0.1632  0.1163
## TD          0.5345  0.7469  0.5978  0.8917  0.6526  1.0000  0.1174  0.1889
## Int         0.1193  0.3507 -0.1255  0.2119 -0.1632  0.1174  1.0000 -0.1202
## Year        0.2880  0.2148  0.2847  0.2247  0.1163  0.1889 -0.1202  1.0000
## Intpercent -0.2750 -0.3711 -0.4592 -0.4370 -0.3466 -0.4091  0.7004 -0.2720
## TDpercent   0.3640  0.2824  0.4992  0.5511  0.7679  0.8257 -0.1094  0.1116
##            Intpercent TDpercent
## G             -0.2750    0.3640
## Att           -0.3711    0.2824
## Pct           -0.4592    0.4992
## Yds           -0.4370    0.5511
## YA            -0.3466    0.7679
## TD            -0.4091    0.8257
```

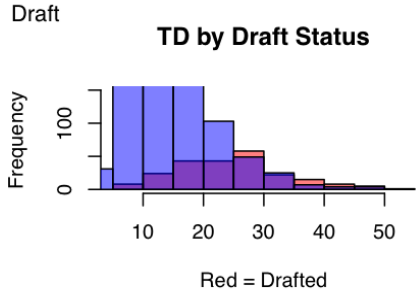
```
## Int            0.7004   -0.1094
## Year          -0.2720    0.1116
## Intpercent     1.0000   -0.3216
## TDpercent     -0.3216    1.0000
```

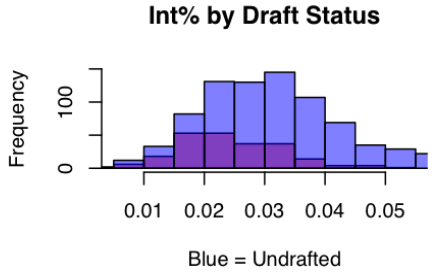
These boxplots and histograms all show that drafted quarterbacks perform better in college.

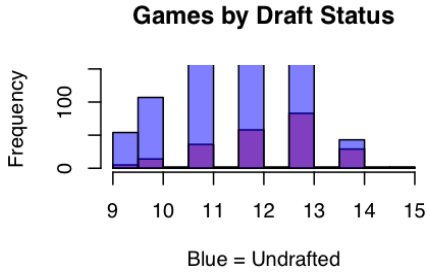



6000
4000
2000
Yds
0
1
Draft


YA by Draft Status
Frequency
0
100
6
7
8
9
10
11
12
Red = Drafted


TD by Draft Status
Frequency
0
100
10
20
30
40
50
Red = Drafted


Int% by Draft Status
Frequency
0
100
0.01
0.02
0.03
0.04
0.05
Blue = Undrafted


Games by Draft Status
Frequency
0
100
9
10
11
12
13
14
15
Blue = Undrafted

Methods

In order to answer our research question, we must build an appropriate model that best determines which players should be drafted based on their college statistics. In order to do so, we use cross validation to compare a variation of both non-ensemble and ensemble methods. For non-ensemble methods, we employ decision trees, K-Nearest Neighbors, and logistic regression. Whereas, our ensemble methods consist of random forest, ridge and lasso, and bootstrap aggregating. We carry-out model selection by assessing which method's model results in the lowest test error when applied to our test set which contains 25% of our original data.

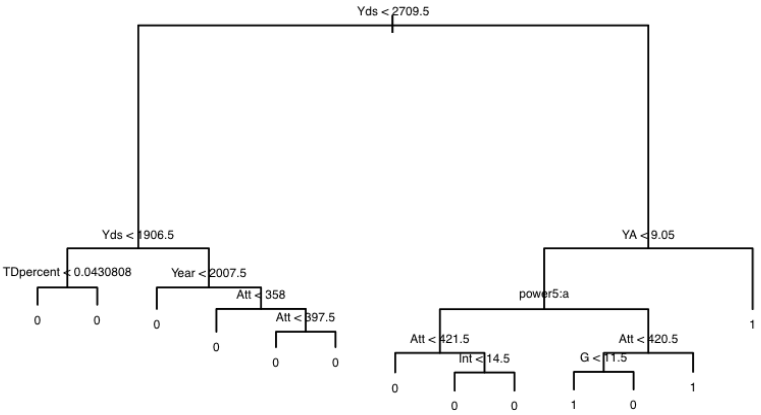
Model Building

Decision Tree

We first used a decision tree model which is a non-parametric classification method. A decision tree uses recursive partitioning to split the dataset into subsets which label each observation into a targeted class. In our case, the decision tree splits into nodes that distinguishes the regions in which a player is likely to be drafted or not drafted.

```
##
## Classification tree:
## tree(formula = Draft ~ ., data = QBdraft, subset = train)
## Variables actually used in tree construction:
## [1] "Yds"       "TDpercent" "Year"      "Att"       "YA"        "power5"
## [7] "Int"       "G"
## Number of terminal nodes:  13
## Residual mean deviance:  0.665 = 508 / 764
## Misclassification error rate: 0.143 = 111 / 777
```

```
## [1] 0.2008
```



We can see from this summary that the variables used in tree construction are: Yards, Touchdown Percentage, Year, power5, Intercept Percentage, Yards Per Attempt, Intercepts, and Games Played. Additionally, there is a misclassification error rate of 0.2008.

More importantly, we used cross validation as a way to find the optimal size for the tree as a way to prevent overfitting. Using 10-fold cross validation, we consider whether pruning the tree might lead to a lower test error.

```
# 10-fold CV for selecting best tree size
tree.cv = cv.tree(drafttree, FUN=prune.misclass, K=10)
```

```r
# Best size
best.cv = min(tree.cv$size[tree.cv$dev==min(tree.cv$dev)])
best.cv
```

```
## [1] 5
```

```r
# Prune the tree to the optimal size
tree.prune = prune.misclass(drafttree, best=best.cv)
summary(tree.prune)
```

```
##
## Classification tree:
## snip.tree(tree = drafttree, nodes = c(2L, 12L, 26L))
## Variables actually used in tree construction:
## [1] "Yds"    "YA"     "power5" "Att"
## Number of terminal nodes:  5
## Residual mean deviance:  0.78 = 602 / 772
## Misclassification error rate: 0.151 = 117 / 777
```

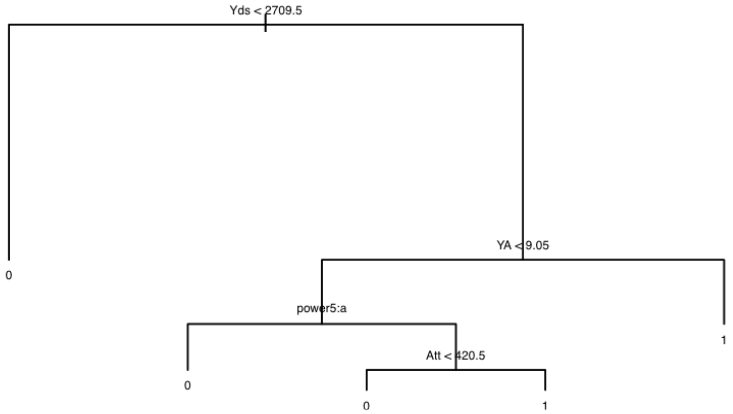
```r
# Test error for tree.prune
treePred=predict(tree.prune, newdata=testing, type="class")
table(Pred=treePred,truth=testing$Draft)
```

```
##     truth
## Pred   0   1
##    0 182  41
##    1  11  25
```

```r
prune.err <- 1 - mean(treePred==testing$Draft)
prune.err
```

```
## [1] 0.2008
```

```r
plot(tree.prune)
text(tree.prune, cex=.5)
```



The pruned tree reduced the number of variables for the model to include: Yards, Yards Per Attempt, Power5, and Attempts. We can see that there is no change on the test error since it is 0.2008.

Bagging

Since decision trees tend to have a higher variance, we decided to use bagging as an alternative tree-based method to improve accuracy over the prediction. For bagging, all 11 predictors are considered for each split of the tree.

```
## [1] 0.2046
```

```
##                 0      1 MeanDecreaseAccuracy MeanDecreaseGini
## G           8.419 -1.394                6.696            8.356
## Att        13.604 -2.488               13.412           20.607
## Pct        13.350 -1.261               12.262           26.864
## Yds        35.593 11.969               43.401           61.433
## YA         17.844 10.790               22.813           24.656
## TD         19.607 -4.423               19.106           22.634
## Int         7.123 -2.188                5.554            9.036
## Year        8.954  3.632                9.110           23.119
## power5      7.959 12.469               13.318           12.461
## Intpercent  7.014  2.627                8.273           22.879
## TDpercent  13.186 -2.374               11.752           22.084
```

The test set error rate associated with the bagged classification tree is 0.2046, lower than that obtained using an optimally-pruned single tree.

Random Forrest

Growing a random forest proceeds in exactly the same way as bagging, except that a smaller number of predictors are considered for each split. Random forest for classification uses the square root of number of predictors, hence for our model only 3 predictors to be considered for each split. It is important to note that a smaller number of predictors helps when predictors are highly correlated.

```
## [1] 0.1969
```

```
##                 0       1 MeanDecreaseAccuracy MeanDecreaseGini
## G           5.139 -0.9272                3.882            9.053
## Att        18.713 -0.9873               19.109           29.567
## Pct         5.962  1.6455                6.404           24.375
## Yds        23.639  9.3465               27.667           42.959
## YA         15.364 12.3001               20.660           27.215
## TD         18.816  0.1690               19.052           28.351
## Int         8.565 -2.8297                7.207           12.291
## Year        5.218  1.5910                5.375           20.206
## power5      6.361 13.6564               12.476           10.309
## Intpercent  7.859  3.4765                9.655           23.362
## TDpercent  13.065  1.8695               14.466           26.221
```

The resulted test error was 0.1969, which is a slight improvement form the previous methods used.

KNN

We then applied K-Nearest Neighbors (KNN) as an alternative non-parametric, hard classification method. For each observation in the test set, KNN will assign it a label in accordance with the majority class of the "k nearest neighbor" of the training data.

```
##  [1] 0.0000 0.1338 0.1145 0.1493 0.1493 0.1583 0.1609 0.1570 0.1660 0.1725
## [11] 0.1699 0.1763 0.1712 0.1750 0.1737 0.1776 0.1712 0.1737 0.1763 0.1737
## [21] 0.1828 0.1763 0.1802 0.1815 0.1815 0.1815 0.1802 0.1866 0.1802 0.1840
```

```
##  [1] 0.2510 0.2394 0.2355 0.2278 0.2239 0.2201 0.2239 0.2317 0.2239 0.2162
## [11] 0.2162 0.2124 0.2162 0.2162 0.2124 0.2201 0.2162 0.2124 0.2239 0.2124
## [21] 0.2162 0.2008 0.2085 0.2008 0.2046 0.2008 0.2085 0.1969 0.2046 0.2008
```

Within our project, we tested a different k's from 1 to 30 and determined that k = 29 produced the smallest test error of 0.1969. We can see that this model performed the same as the random forest model.

Logistic Regression

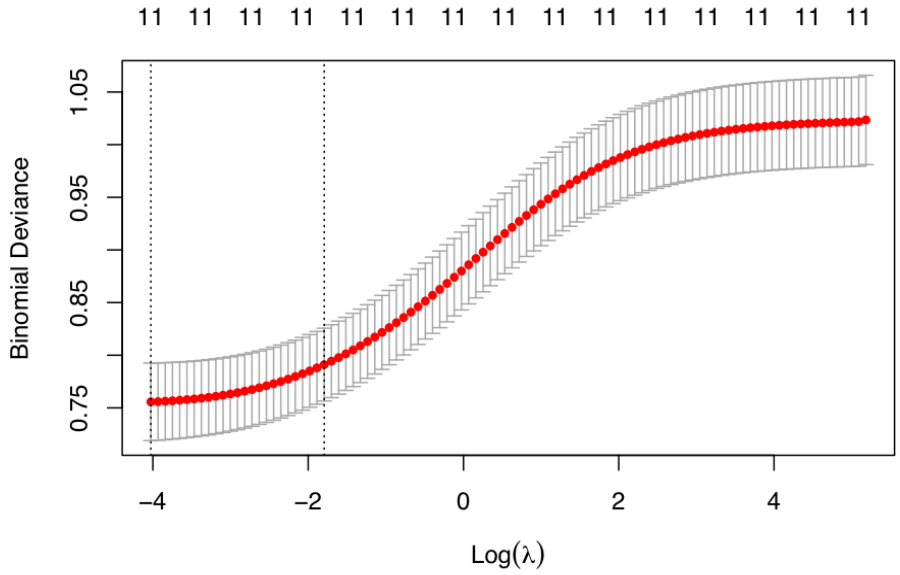
We also used logistic which is considered as soft classification since it explicitly estimates the probabilities rather than class labels unlike the previous models.

```
## [1] 0.1544
```

```
##      true
## pred   0   1
##    0 582  86
##    1  34  75
```

Using logistic regression, we get an error rate of 0.1544. There were 582 observations where the model correctly predicted the player would not get drafted and 75 observations where the model correctly predicted the player would get drafted. However, there were misclassified observations, 86 players who got drafted but the model predicted they didn't and 34 players who didn't get drafted but the model predicted they would.

Ridge




```
## [1] 0.01784
```

```
## 12 x 1 sparse Matrix of class "dgCMatrix"
##                       s0
## (Intercept)  1.165e+02
## G           -5.583e-02
## Att          3.964e-03
```

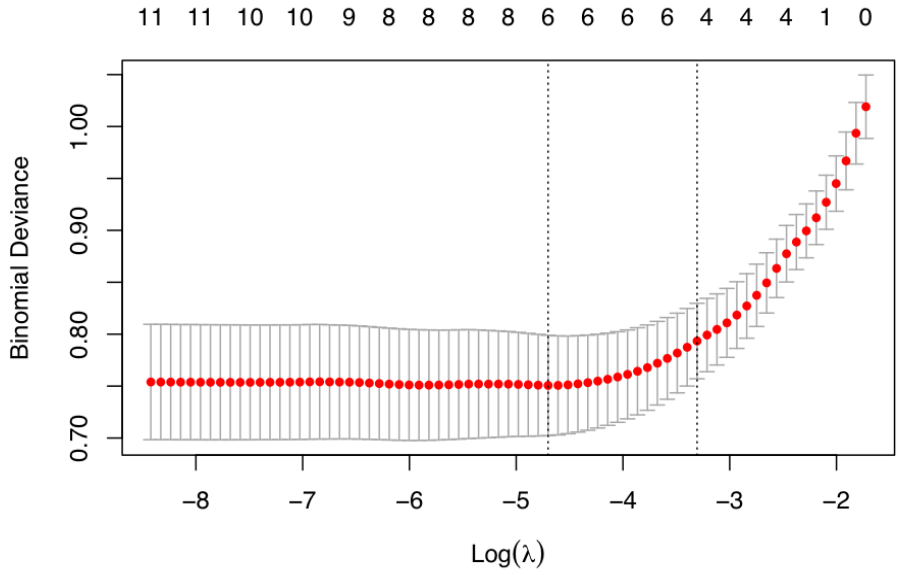
```
## Pct           1.048e-02
## Yds           5.008e-04
## YA            4.099e-01
## TD            6.172e-03
## Int           2.553e-02
## Year         -6.194e-02
## power51       1.187e+00
## Intpercent   -3.007e+01
## TDpercent     7.874e+00
```

```
## [1] 0.2162
```

```
##
## predicted.classes   0   1
##                 0 189  52
##                 1   4  14
```

The plot displays the cross-validation error depending on the log of lambda. The dashed vertical line indicates that the log of the optimal value of lambda is around -2, which minimizes the prediction error. The exact value is approximately 0.2162. Using this value, we can find the regression coefficients.

LASSO




```
## [1] 0.009087
```

```
## 12 x 1 sparse Matrix of class "dgCMatrix"
##                      s0
## (Intercept)  71.490495
## G             .
## Att           .
## Pct           .
## Yds           0.001041
## YA            0.224945
```

```
## TD            .
## Int           .
## Year         -0.038779
## power51       1.004891
## Intpercent  -11.020402
## TDpercent     4.014524
```

```
## [1] 0.2085
```

```
##     true
## pred   0   1
##    0 184  45
##    1   9  21
```

The plot displays cross-validation error according to the log of lambda. The left vertical line indicates that the optimal value of log lambda is approximately -6.3 which minimizes the prediction error. The exact value is approximately 0.2085. Using this value, we can find the regression coefficients.

Conclusions

Our final model can be framed from the logistic regression model since it has the lowest test error at approximately 0.1544. Because only 22% of the quarterbacks in our dataset were drafted, a model predicting no one to get drafted would have a test error of .22. In order to have an effective model we must have a test error lower than this. Some models were unable to beat the baseline, but our final model was able to and is potentially useful in predicting a quarterback's draft status. There are limitations of this study because NFL quarterback scouts look at variables not included in this study. A player's height, weight, speed, intelligence, throwing motion, leadership, etc. are all considered in real life, but many factors are intangible and difficult to incorporate into a model. In the future, we could gather more predictors on the players we study to see if these factors increase the accuracy of the model.

References

https://www.sports-reference.com/cfb/years/2017-passing.html Data can be found on this website and is organized by college year.