Analysis of NBA Team Statistics

Jovan Krcadinac, Danny Annese

May 5, 2019

Introduction and Literature Review:

Our group used the following dataset for our analysis: - https://www.kaggle.com/dzchen/nba-advanced-statistics-20172018 (https://www.kaggle.com/dzchen/nba-advanced-statistics-20172018) This dataset gives NBA Advanced Statistics for every active player within the NBA during the 2017-2018 Regular Season. This dataset has 532 observations (for each of the players) and 57 variables (that either give basic player information or season statistics).

The scientific goal for our analysis is to utilize the player statistics within the previously mentioned dataset to predict an NBA team's win percentage. Namely, we are trying to answer questions like: - Which NBA player statistics best predict the highest winning percentage? - Out of Linear Regression, Logistic Regression, Bayesian Linear Regression, Ridge Regression, and Lasso Regression; which model is most appropriate / will most accurately predict a team's win percentage?

In terms of our analysis, we will face significant challenges throughout each step. In order to systematically tackle these challenges, we'll generally stick to the following steps: - Identify relevant NBA player statistics - Isolate the relevant statistics - Model the relevant statistics using the 5 aforementioned regression methods - Identify the method that yields the best model - Within the best model, identify the statistics that most significantly influenced a team's win percentage

After reviewing literature and relevant sources, we found the following analysis: - https://arxiv.org/pdf/1604.03186.pdf (https://arxiv.org/pdf/1604.03186.pdf) This piece of literature has the same scientific goals as us, but utilizes data from a different season and took a rather different approach by focusing on the statistics of a few high-impact players (e.g. Lebron James, Dirk Nowitzki, etc.).

Summary statistics and data visualization:

Challenges

A challenge that our group may face when doing this analysis is that we are using two different datasets and will need to find a way to integrate both sets of data with each other to properly do our analysis. It will be difficult to matchup all of the players from one team to the team statistics from the other dataset for each year since there are 30 teams with different players each year. Our analysis is relying heavily on the combination of these two datasets since we are predicting team winning percentage based on players' individual statistics. This process will take up a lot of time and will need to be done before we perform any regression for our analysis.

Team Variables Analysis

- The first six variables are relevant to our analysis goal because they not only help us to identify and index the current dataset, but also helps us match players with their respective teams within 'final_data_sort'. Having these common indices will allow us to integrate data from multiple sources within our final regression model (that models team win percentage).

    1. 'PLAYER_ID' Index
    2. 'PLAYER_NAME' Name
    3. 'Age' Age
    4. 'TEAM_ABBREVIATION' Team
    5. 'TEAM_ID' Team ID
    6. 'CFID' ID
    7. 'CFPARAMS' CF Parameters
- The next three variables are relevant to our analysis goal because they determine the weight of each players contribution in relation with the teams win percentage. Naturally, players that play less have a smaller impact on the team's win percentage and vice versa.

    8. 'GP' Games Played
    9. 'W' Wins
    10. 'L' Losses
    11. 'W_PCT' Win Percentage
    12. 'MIN' Minutes
    13. 'OFF_RATING' Offensive Rating
    14. 'DEF_RATING' Defensive Rating
    15. 'NET_RATING' Net Rating
- The next variables are relevant because it helps us determine how each player's individual shooting contributions affect the team's win percentage. The remaining three variables are relevant because they will be used to see how each player's shooting efficiency (within three different categories) affects the team's win percentage.

    10. 'FGM' Field Goals Made
    11. 'FGA' Field Goals Attempted
    12. 'FGM_PG' Field Goals Made Per Game
    13. 'FGA_PG' Field Goals Attempted Per Game
    14. 'FG_PCT' Field Goal Percentage
    15. 'EFG_PCT' Effective Field Goal Percentage
    16. 'TS_PCT' True Shooting Percentage
- Although scoring is one of the primary facets of a player's game it is definetly not the whole picture. The following percentage data reflects other primary statistics for players; more specifically: Rebound Percentage, Assist Percentage, Steal Percentage, Block Percentage, Turnover Percentage, and Usage Percentage. A player's ability to influence the game through these categories will significantly contribute to whether or not a team will be able to win.

    17. 'AST_PCT' Assist Percentage
    18. 'AST_TO' Turnovers
    19. 'AST_RATIO' Turnovers to Assits Ratio
    20. 'OREB_PCT' Offensive Rebound Percentage
    21. 'DREB_PCT' Deffensive Rebound Percentage

22. ‘TM_TOV_PCT’ Team Turnover Percentage
23. ‘USG_PCT’ Usage Percent

- The following set of statistics represent the major statistical categories for each player. They will play a majority role in determining how each player’s contribution on each team will influence the team’s overall win percentage for the season.

24. ‘PACE’ Pace Factor
25. ‘PIE’ PIE Rating

- The final category of variables comes from the statistical ranks of each individual players.

26. “GP_RANK”
27. “W_RANK”
28. “L_RANK”
29. “W_PCT_RANK”
30. “MIN_RANK”
31. “OFF_RATING_RANK”
32. “DEF_RATING_RANK”
33. “NET_RATING_RANK”
34. “AST_PCT_RANK”
35. “AST_TO_RANK”
36. “AST_RATIO_RANK”
37. “OREB_PCT_RANK”
38. “DREB_PCT_RANK”
39. “REB_PCT_RANK”
40. “TM_TOV_PCT_RANK”
41. “EFG_PCT_RANK”
42. “TS_PCT_RANK”
43. “USG_PCT_RANK”
44. “PACE_RANK”
45. “PIE_RANK”
46. “FGM_RANK”
47. “FGA_RANK”
48. “FGM_PG_RANK”
49. “FGA_PG_RANK”
50. “FG_PCT_RANK”

Our final variable is Team Win Percentage, the variable we are trying to determine.

```
51. "TEAM_WIN_PCT"
```

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
data = read.csv('NBAadvancedstats17_18.csv', header = T)

data_sort = data[order(data$TEAM_ABBREVIATION), ]

TEAM_WIN_PCT = c(.293, .341, .671, .439, .329, .610, .293, .561, .476, .707, .793, .5
85, .512, .427, .268, .537, .537, .573, .585, .354, .585, .305, .634, .256, .598, .32
9, .573, .720, .585, .524)

TEAM_ABBREVIATION = c('ATL', 'BKN', 'BOS', 'CHA', 'CHI', 'CLE', 'DAL', 'DEN', 'DET',
'GSW', 'HOU', 'IND', 'LAC', 'LAL', 'MEM', 'MIA', 'MIL', 'MIN', 'NOP', 'NYK', 'OKC', '
ORL', 'PHI', 'PHX', 'POR', 'SAC', 'SAS', 'TOR', 'UTA', 'WAS')

w = data.frame(TEAM_WIN_PCT = TEAM_WIN_PCT, TEAM_ABBREVIATION = TEAM_ABBREVIATION)

final_data = merge(data_sort, w, by  = 'TEAM_ABBREVIATION')

head(final_data)
```

	TEAM_ABBREVIATI...	PLAYER...	PLAYER_NAME	TEAM_ID	...	...	W	L	W_...
	<fctr>	<int>	<fctr>	<int>	<dbl>	<int>	<int>	<int>	<dbl>
1	ATL	1628510	Andrew White III	1610612737	24	14	1	13	0.071
2	ATL	1628499	Antonius Cleveland	1610612737	24	13	5	8	0.385
3	ATL	1627814	Damion Lee	1610612737	25	10	2	8	0.200
4	ATL	1627761	DeAndre' Bembry	1610612737	23	22	6	16	0.273
5	ATL	203471	Dennis Schroder	1610612737	24	67	21	46	0.313
6	ATL	203473	Dewayne Dedmon	1610612737	28	58	16	42	0.276

6 rows | 1-10 of 59 columns

```
nrow(final_data)
```

```
## [1] 531
```

```
ncol(final_data)
```

```
## [1] 58
```

Since our data set consisted of advanced player statistics, we knew we were going to have highly correlated variables. For example, FG_PCT and FG_PCT_RANK are both based on the player's field goal accuracy, so we knew we had to eliminate some of the variables right away. Another issue our team had facing the data was that it was filled with factor variables. These include TEAM_ABBREVIATION, TEAM_ID, PLAYER_ID, PLAYER_NAME, AGE, CFID, and CFPARAMS. These variables provided little to no value when we perform our analysis.

```
final_data_sort = select(.data = final_data, -c(TEAM_ABBREVIATION, TEAM_ID, PLAYER_ID
, PLAYER_NAME, AGE, CFID, CFPARAMS))

ncol(final_data_sort)
```

```
## [1] 51
```

After removing the factor variables, it's time to see which variables are highly correlated.

```
tmp <- cor(final_data_sort)
tmp[!lower.tri(tmp)] <- 0
final_data_sort <- final_data_sort[,!apply(tmp,2,function(x) any(x > 0.95))]

ncol(final_data_sort)
```

```
## [1] 44
```

We can see here that variables such as L, EFG_PCT, FGM, FGM_PG, DREB_PCT_RANK, FGM_RANK, and FGM_PG_RANK were all removed based on a .95 threshold. After removing the unnecessary and highly correlated variables, we are left with a data set of 531 rows and 44 variables. Now it's time to remove the outliers. We will remove the players that played less than a minute.

```
tm = which((final_data_sort$MIN < 1) == T)

final_data_sort = final_data_sort[-c(tm), ]

head(final_data_sort)
```

	GP	W	W_P...	MIN	OFF_RATING	DEF_RATING	NET_RATING	AST_PCT	AST_TO
	<int>	<int>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>
1	14	1	0.071	14.2	94.3	110.6	-16.3	0.044	0.71
2	13	5	0.385	6.2	88.3	99.1	-10.8	0.038	1.00

3	10	2	0.200	26.0	99.0	106.3	-7.2	0.107	2.38
4	22	6	0.273	17.2	101.2	107.8	-6.6	0.140	0.87
5	67	21	0.313	31.0	103.6	110.2	-6.6	0.352	2.28
6	58	16	0.276	24.8	102.0	107.9	-5.8	0.089	1.00

6 rows | 1-10 of 45 columns

After cleaning the data, our final data set consists of 529 observations and 44 columns.

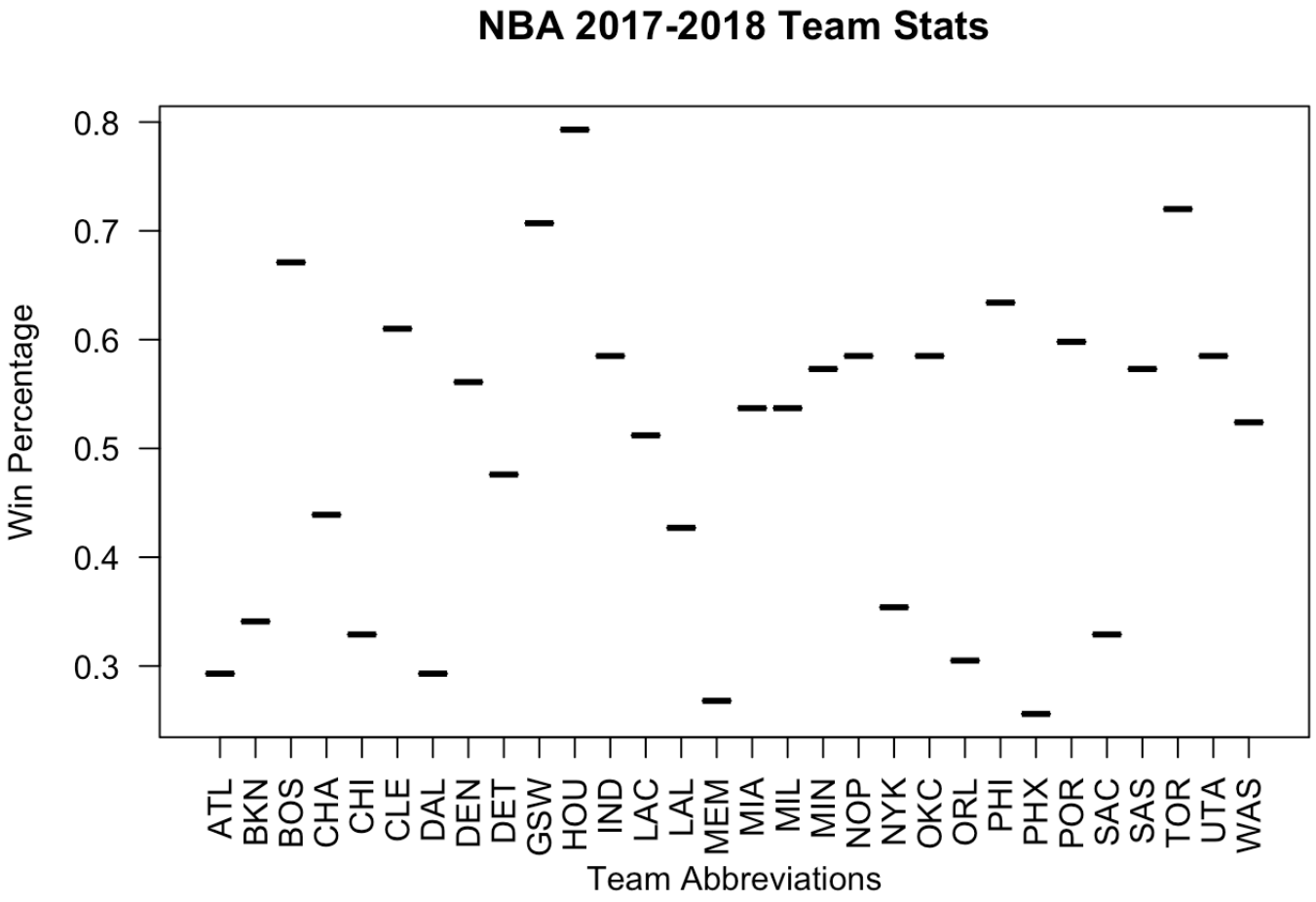
```
summary(final_data_sort)
```

```
##        GP               W              W_PCT             MIN       
##  Min.   : 1.00   Min.   : 0.00   Min.   :0.0000   Min.   : 1.20  
##  1st Qu.:20.00   1st Qu.: 9.00   1st Qu.:0.3330   1st Qu.:12.00  
##  Median :54.00   Median :22.00   Median :0.5070   Median :19.00  
##  Mean   :46.32   Mean   :23.13   Mean   :0.4826   Mean   :19.08  
##  3rd Qu.:69.00   3rd Qu.:37.00   3rd Qu.:0.6070   3rd Qu.:26.70  
##  Max.   :78.00   Max.   :62.00   Max.   :1.0000   Max.   :37.10  
##    OFF_RATING      DEF_RATING      NET_RATING          AST_PCT       
##  Min.   :  0.0   Min.   :  0.0   Min.   :-120.800   Min.   :0.0000  
##  1st Qu.:101.1   1st Qu.:103.3   1st Qu.:  -6.500   1st Qu.:0.0680  
##  Median :105.1   Median :106.3   Median :  -0.900   Median :0.1010  
##  Mean   :103.7   Mean   :106.0   Mean   :  -2.281   Mean   :0.1317  
##  3rd Qu.:108.0   3rd Qu.:109.1   3rd Qu.:   3.300   3rd Qu.:0.1770  
##  Max.   :138.9   Max.   :225.0   Max.   :  75.000   Max.   :1.0000  
##      AST_TO        AST_RATIO        OREB_PCT          DREB_PCT      
##  Min.   : 0.00   Min.   :  0.0   Min.   :0.00000   Min.   :0.0000  
##  1st Qu.: 1.00   1st Qu.: 10.6   1st Qu.:0.01700   1st Qu.:0.1010  
##  Median : 1.43   Median : 14.4   Median :0.03100   Median :0.1400  
##  Mean   : 1.61   Mean   : 16.5   Mean   :0.04538   Mean   :0.1521  
##  3rd Qu.: 2.03   3rd Qu.: 20.8   3rd Qu.:0.06600   3rd Qu.:0.1930  
##  Max.   :11.00   Max.   :100.0   Max.   :0.25000   Max.   :1.0000  
##     REB_PCT         TM_TOV_PCT        TS_PCT          USG_PCT       
##  Min.   :0.0000   Min.   : 0.00   Min.   :0.0000   Min.   :0.0000  
##  1st Qu.:0.0620   1st Qu.: 8.30   1st Qu.:0.5000   1st Qu.:0.1460  
##  Median :0.0850   Median :10.10   Median :0.5420   Median :0.1750  
##  Mean   :0.0984   Mean   :10.59   Mean   :0.5258   Mean   :0.1842  
##  3rd Qu.:0.1300   3rd Qu.:12.60   3rd Qu.:0.5790   3rd Qu.:0.2200  
##  Max.   :0.5000   Max.   :66.70   Max.   :1.5000   Max.   :0.5000  
##       PACE             PIE                FGA             FGA_PG      
##  Min.   : 70.72   Min.   :-4.00000   Min.   :   0.0   Min.   : 0.000  
##  1st Qu.: 97.75   1st Qu.: 0.06100   1st Qu.:  68.0   1st Qu.: 3.100  
##  Median : 99.24   Median : 0.08500   Median : 302.0   Median : 5.400  
##  Mean   : 99.49   Mean   : 0.08496   Mean   : 375.3   Mean   : 6.678  
##  3rd Qu.:101.06   3rd Qu.: 0.11000   3rd Qu.: 599.0   3rd Qu.: 9.700  
##  Max.   :119.62   Max.   : 4.00000   Max.   :1603.0   Max.   :21.100  
##      FG_PCT          GP_RANK          W_RANK          L_RANK      
##  Min.   :0.0000   Min.   :  1.0   Min.   :  1.0   Min.   :  1.0  
```

```
##  1st Qu.:0.3950   1st Qu.:124.0   1st Qu.:129.0   1st Qu.:121.0
##  Median :0.4400   Median :260.0   Median :261.0   Median :260.0
##  Mean   :0.4372   Mean   :260.4   Mean   :259.8   Mean   :261.6
##  3rd Qu.:0.4940   3rd Qu.:397.0   3rd Qu.:391.0   3rd Qu.:399.0
##  Max.   :1.0000   Max.   :519.0   Max.   :511.0   Max.   :531.0
##    W_PCT_RANK       MIN_RANK   OFF_RATING_RANK DEF_RATING_RANK
##  Min.   :  1.0   Min.   :  1   Min.   :  2     Min.   :  1
##  1st Qu.:134.0   1st Qu.:133   1st Qu.:134     1st Qu.:135
##  Median :266.0   Median :265   Median :266     Median :267
##  Mean   :264.5   Mean   :265   Mean   :266     Mean   :267
##  3rd Qu.:392.0   3rd Qu.:397   3rd Qu.:398     3rd Qu.:399
##  Max.   :511.0   Max.   :529   Max.   :529     Max.   :531
##  NET_RATING_RANK  AST_PCT_RANK    AST_TO_RANK  AST_RATIO_RANK
##  Min.   :  2.0   Min.   :  1.0   Min.   :  1   Min.   :  1.0
##  1st Qu.:134.0   1st Qu.:133.0   1st Qu.:133   1st Qu.:133.0
##  Median :267.0   Median :265.0   Median :265   Median :265.0
##  Mean   :266.5   Mean   :264.3   Mean   :263   Mean   :264.3
##  3rd Qu.:399.0   3rd Qu.:397.0   3rd Qu.:383   3rd Qu.:397.0
##  Max.   :531.0   Max.   :503.0   Max.   :494   Max.   :503.0
##  OREB_PCT_RANK    REB_PCT_RANK   TM_TOV_PCT_RANK  EFG_PCT_RANK
##  Min.   :  1.0   Min.   :  1.0   Min.   :  1.0   Min.   :  1.0
##  1st Qu.:133.0   1st Qu.:133.0   1st Qu.:135.0   1st Qu.:134.0
##  Median :265.0   Median :265.0   Median :267.0   Median :266.0
##  Mean   :263.1   Mean   :264.9   Mean   :266.5   Mean   :265.5
##  3rd Qu.:397.0   3rd Qu.:397.0   3rd Qu.:399.0   3rd Qu.:398.0
##  Max.   :485.0   Max.   :518.0   Max.   :531.0   Max.   :518.0
##   TS_PCT_RANK     USG_PCT_RANK   PACE_RANK      PIE_RANK      FGA_RANK
##  Min.   :  1.0   Min.   :  2   Min.   :  3   Min.   :  1   Min.   :  1.0
##  1st Qu.:134.0   1st Qu.:134   1st Qu.:135   1st Qu.:134   1st Qu.:133.0
##  Median :266.0   Median :266   Median :267   Median :266   Median :265.0
##  Mean   :265.8   Mean   :266   Mean   :267   Mean   :266   Mean   :264.5
##  3rd Qu.:393.0   3rd Qu.:398   3rd Qu.:399   3rd Qu.:398   3rd Qu.:397.0
##  Max.   :519.0   Max.   :527   Max.   :531   Max.   :531   Max.   :527.0
##   FGA_PG_RANK     FG_PCT_RANK     TEAM_WIN_PCT
##  Min.   :  1.0   Min.   :  1.0   Min.   :0.2560
##  1st Qu.:133.0   1st Qu.:134.0   1st Qu.:0.3410
##  Median :265.0   Median :266.0   Median :0.5370
##  Mean   :264.7   Mean   :265.4   Mean   :0.4947
##  3rd Qu.:397.0   3rd Qu.:398.0   3rd Qu.:0.5850
##  Max.   :527.0   Max.   :518.0   Max.   :0.7930
```

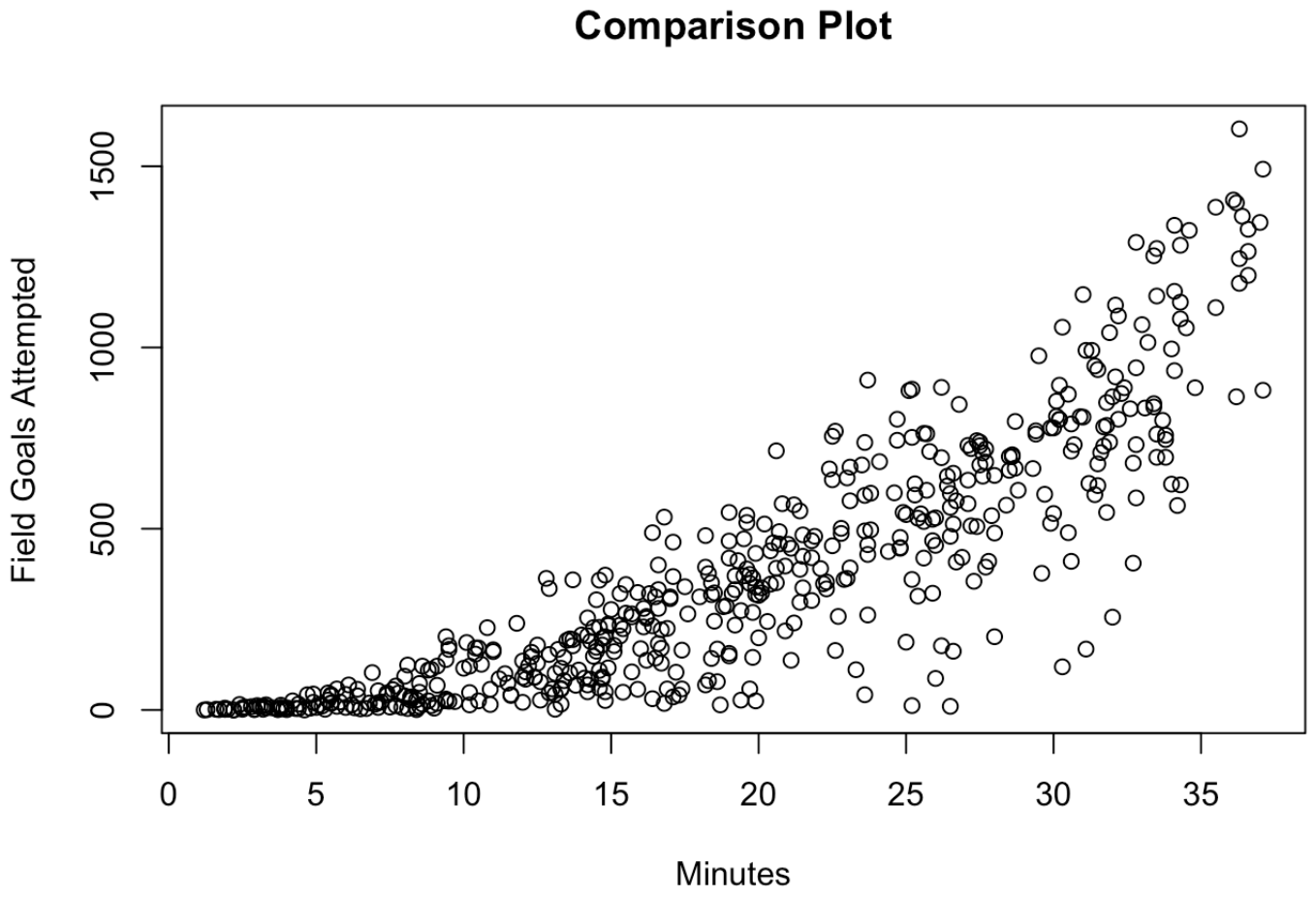
Teams and their win percentages.

```
w$TEAM_ABBREVIATION = as.factor(w$TEAM_ABBREVIATION)

plot(w$TEAM_ABBREVIATION, w$TEAM_WIN_PCT, las = 2, xlab = 'Team Abbreviations', ylab
= 'Win Percentage', main = 'NBA 2017-2018 Team Stats')
```



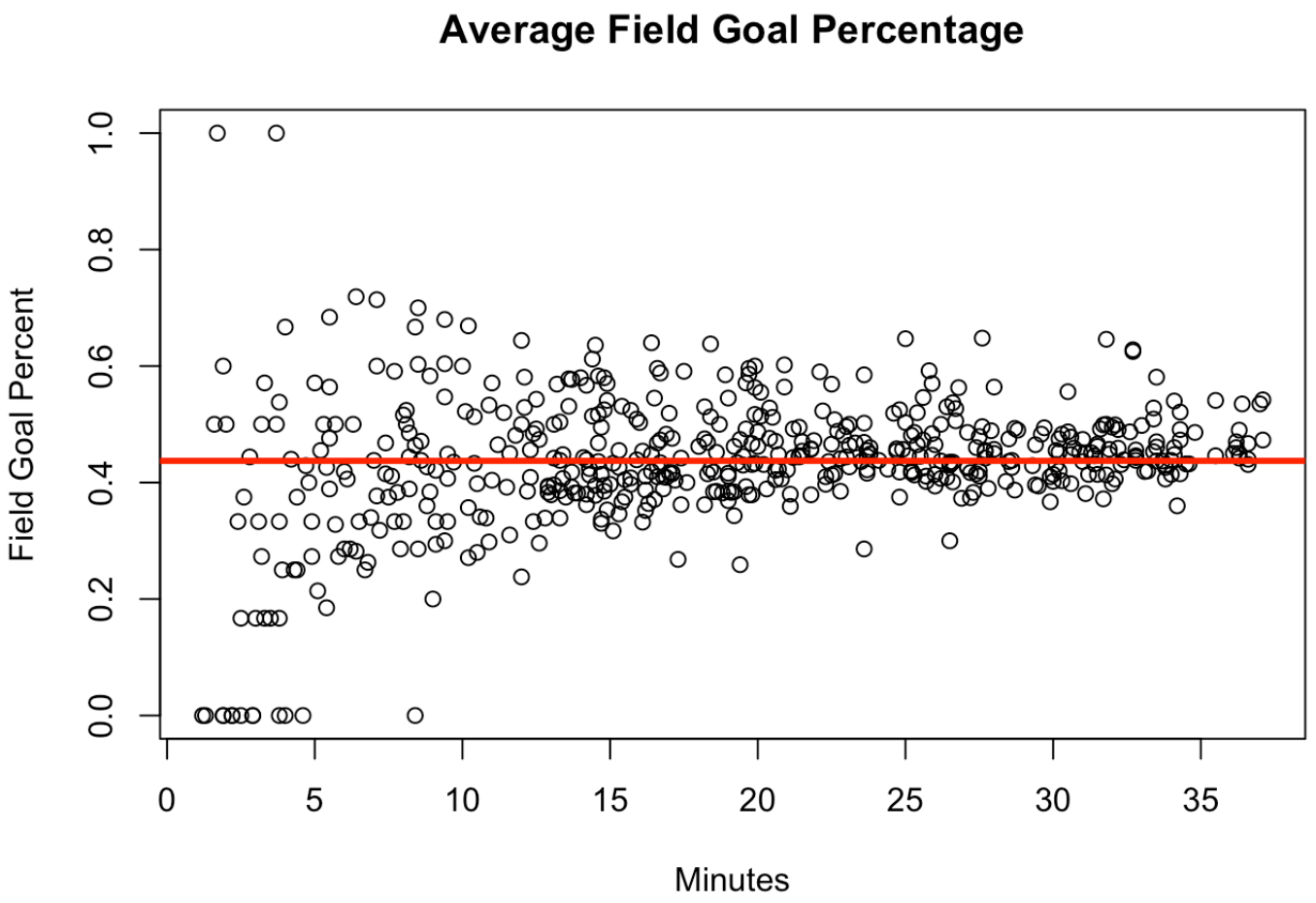
We can see that some variables will still have a relationship, especially the minutes played variable. [see plot below]

```
plot(final_data_sort$MIN, final_data_sort$FGA, xlab = 'Minutes', ylab = 'Field Goals Attempted', main = 'Comparison Plot')
```



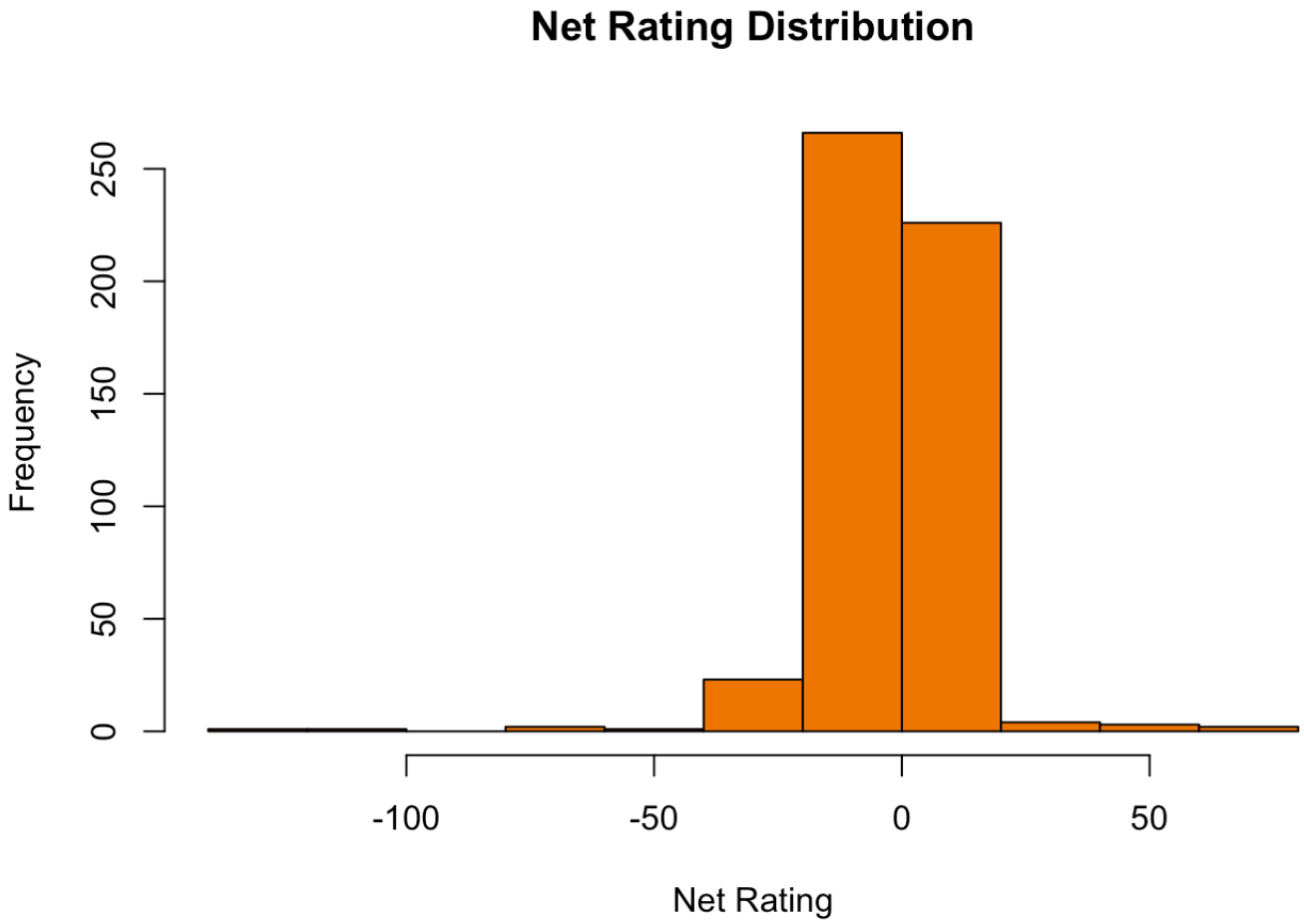
Other visualizations can show more about the player statistics themselves.

```
ave = mean(final_data_sort$FG_PCT)

plot(final_data_sort$MIN, final_data_sort$FG_PCT, xlab = 'Minutes', ylab = 'Field Goa
l Percent', main = 'Average Field Goal Percentage')
abline(h = ave, lwd = 3, col = 'red')
```



We can see here that the average percentage is 0.4372174.

```
hist(final_data_sort$NET_RATING, main = 'Net Rating Distribution', xlab = 'Net Rating
', col="darkorange2")
```



Most players are well balanced between offense and defense while a few are only good at one or the other.

These visualizations will help us in picking out the best parameters for our models and ultimately give us the most accurate results.

Analysis:

Statistical Learning Task

Our team plans on identifying which player statistics best predict the highest winning percentage. We will fit the data with multiple different types of regression models including: linear, lasso, ridge, logistic, and bayesian. Furthermore, our analysis will find the most significant variables that provide the strongest relationship between the variables and the winning percentage.

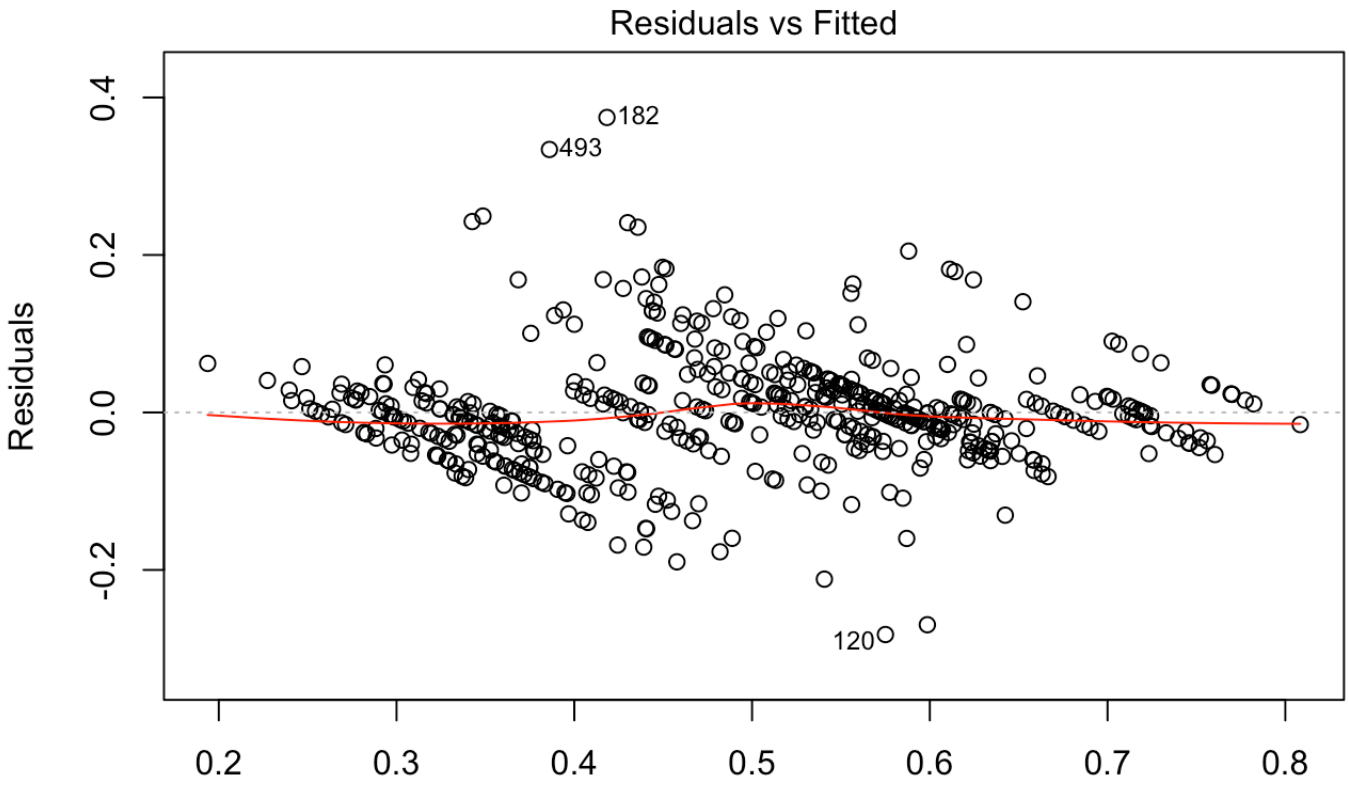
Results:

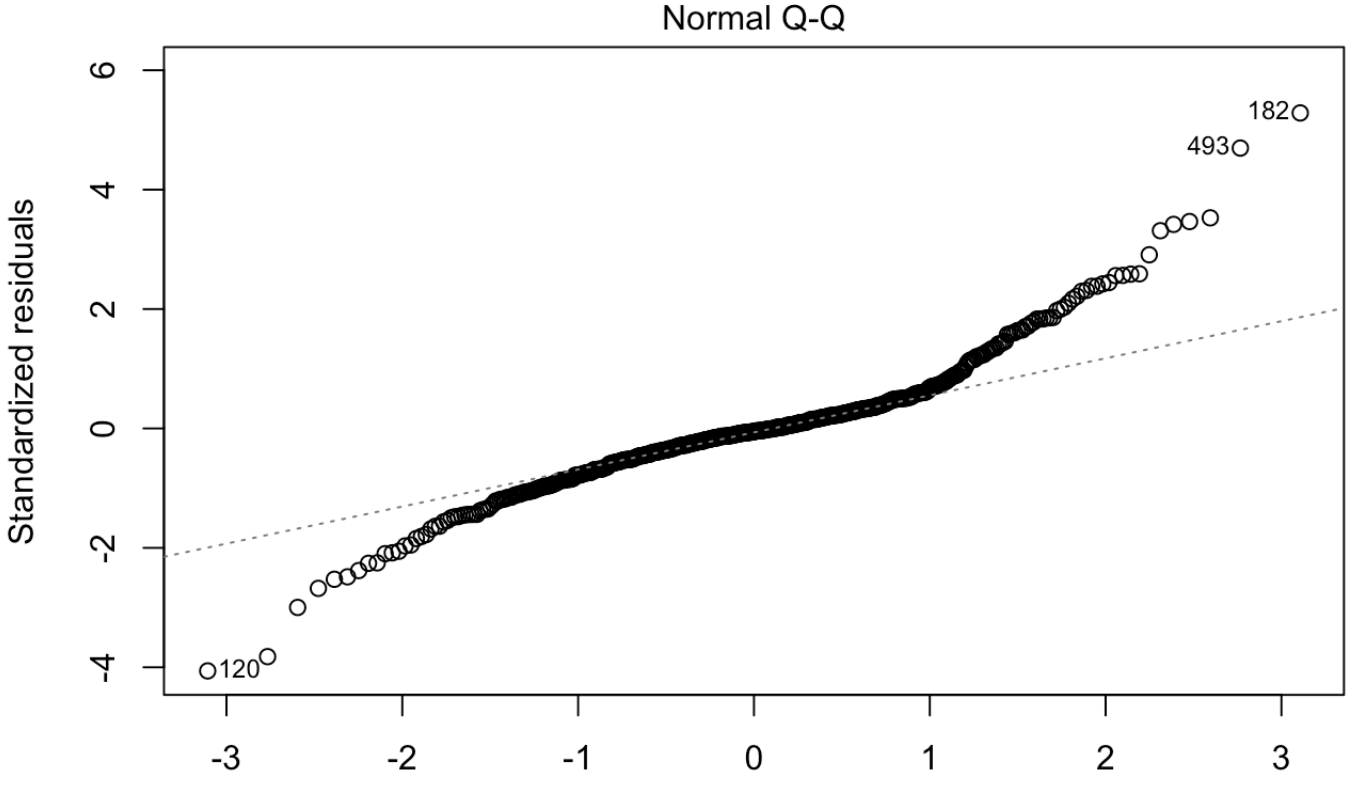
The first procedure we performed was multiple linear regression. In order to find the best subset of explanatory variables, we used stepwise selection using AIC parameters in both directions to find the best significant factors to include in the final model shown below.

```
linear = lm(TEAM_WIN_PCT~ ., data = final_data_sort)
linear_final = step(linear, direction = 'both', trace = F)
summary(linear_final)
```

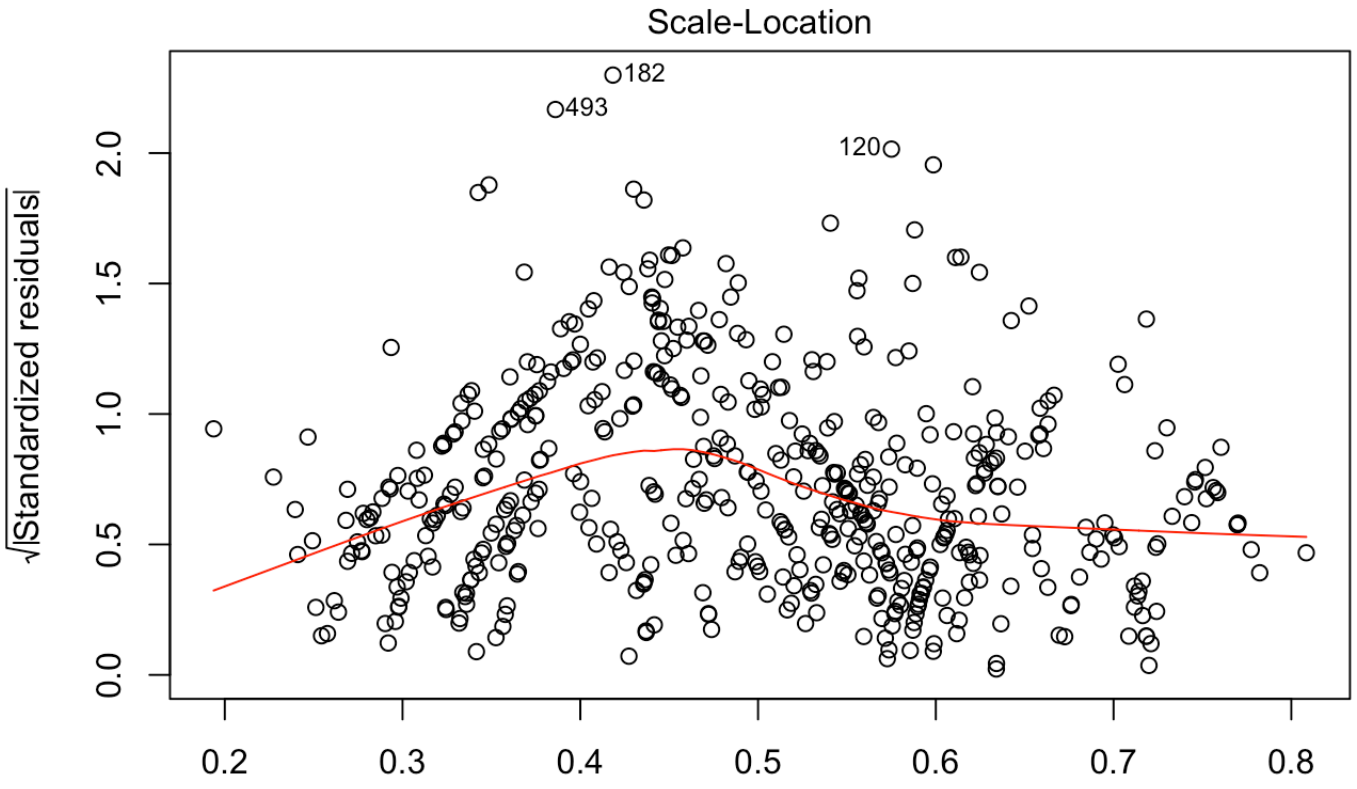
```
##
## Call:
## lm(formula = TEAM_WIN_PCT ~ GP + W + MIN + DEF_RATING + AST_RATIO +
##     USG_PCT + PIE + FGA + FGA_PG + GP_RANK + W_RANK + W_PCT_RANK +
##     AST_PCT_RANK + TS_PCT_RANK + PIE_RANK + FGA_PG_RANK + FG_PCT_RANK,
##     data = final_data_sort)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.28205 -0.03398 -0.00380  0.02487  0.37463
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)   9.453e-01  1.789e-01   5.284 1.87e-07 ***
## GP           -4.387e-03  7.614e-04  -5.761 1.44e-08 ***
## W             4.319e-03  2.517e-03   1.716 0.086812 .
## MIN          -4.647e-03  1.486e-03  -3.127 0.001867 **
## DEF_RATING   -4.843e-04  3.022e-04  -1.603 0.109625
## AST_RATIO     2.426e-03  6.516e-04   3.723 0.000219 ***
## USG_PCT       2.066e-01  1.074e-01   1.924 0.054882 .
## PIE           5.815e-02  1.483e-02   3.921 0.000100 ***
## FGA           1.867e-04  4.493e-05   4.156 3.80e-05 ***
## FGA_PG       -1.485e-02  5.040e-03  -2.947 0.003354 **
## GP_RANK       2.238e-04  1.025e-04   2.182 0.029548 *
## W_RANK       -5.973e-04  3.008e-04  -1.986 0.047595 *
## W_PCT_RANK   -3.436e-04  4.362e-05  -7.878 2.01e-14 ***
## AST_PCT_RANK  1.100e-04  4.729e-05   2.325 0.020438 *
## TS_PCT_RANK  -9.300e-05  3.748e-05  -2.481 0.013414 *
## PIE_RANK      1.964e-04  3.894e-05   5.045 6.32e-07 ***
## FGA_PG_RANK  -4.030e-04  1.202e-04  -3.353 0.000858 ***
## FG_PCT_RANK  -8.805e-05  3.859e-05  -2.282 0.022929 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.07163 on 511 degrees of freedom
## Multiple R-squared:  0.7743, Adjusted R-squared:  0.7668
## F-statistic: 103.1 on 17 and 511 DF,  p-value: < 2.2e-16
```

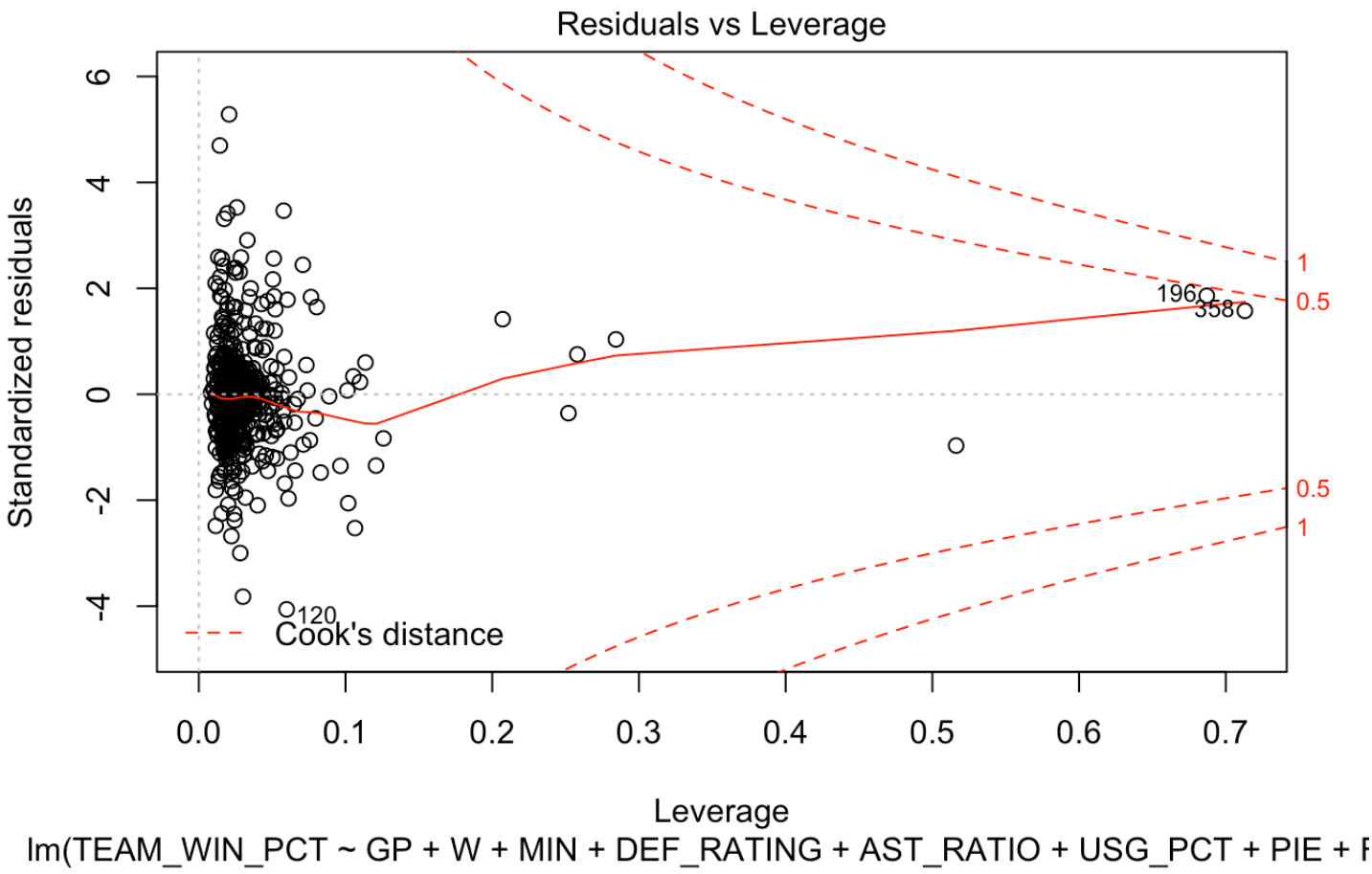
```
plot(linear_final)
```


Residuals vs Fitted
Residuals
Fitted values
lm(TEAM_WIN_PCT ~ GP + W + MIN + DEF_RATING + AST_RATIO + USG_PCT + PIE + F
182
493
120
0.4
0.2
0.0
-0.2
0.2
0.3
0.4
0.5
0.6
0.7
0.8


Normal Q-Q
Standardized residuals
182
493
120
6
4
2
0
-2
-4
-3
-2
-1
0
1
2
3

Theoretical Quantiles
lm(TEAM_WIN_PCT ~ GP + W + MIN + DEF_RATING + AST_RATIO + USG_PCT + PIE + F





The final model had an r-squared value of 0.7743 with a p-value close to zero [2.2e-16]. Looking at the selected variables, it makes sense to see general statistics such as wins, minutes played, and pie stats to make the cut. Other individual statistics include defensive rating, field goal statistics, rebound percentage rank, and usage (a crucial factor because it shows how much a player is used during the season) are also included. However, one thing to note is that many of the factors included are heavily correlated and this is depicted when only six variables have a VIF less than 5 (multicollinearity) hinting linear regression isn't the optimal method.

```
library(car)
```

```
## Loading required package: carData
```

```
##
## Attaching package: 'car'
```

```
## The following object is masked from 'package:dplyr':
##
##     recode
```

```
vif = vif(linear_final)
vif
```

```
##           GP            W          MIN   DEF_RATING    AST_RATIO
##    39.592414   160.430306    20.511259     1.216012     4.662697
##      USG_PCT          PIE          FGA       FGA_PG      GP_RANK
##     4.426313     1.417094    25.128294    55.454552    25.922724
##       W_RANK   W_PCT_RANK AST_PCT_RANK  TS_PCT_RANK     PIE_RANK
##   214.135951     4.549325     5.298110     3.366608     3.643322
##  FGA_PG_RANK  FG_PCT_RANK
##    34.567931     3.567831
```

Another regression conducted was Lasso regression. This is a form of penalized regression logistic regression in which it minimizes the sum of squares, but with constraints. In other words, they take into weight on significant variables and basically removes variables that really don't contribute to the model. This method is called shrinkage in which it uses lambda values aka tuning parameter to determine which coefficients become eliminated (turned to zeroes). Typically the higher the lambda values are, more and more coefficients will be reduced to zero and bias increases at the cost of variance. In this case, we use two methods to determine the best lambda value and coefficients to consider in the final method. The first way is using the value that minimizes the root mean square and this is shown in the graph as the log of the optimal value is around -9 (actual value around 0.000194).

```
library(glmnet)
```

```
## Loading required package: Matrix
```

```
## Loading required package: foreach
```

```
## Loaded glmnet 2.0-16
```

```
final_data2=na.omit(final_data_sort)
with(final_data_sort,sum(is.na(TEAM_WIN_PCT)))
```

```
## [1] 0
```

```
dim(final_data2)
```

```
## [1] 529  44
```

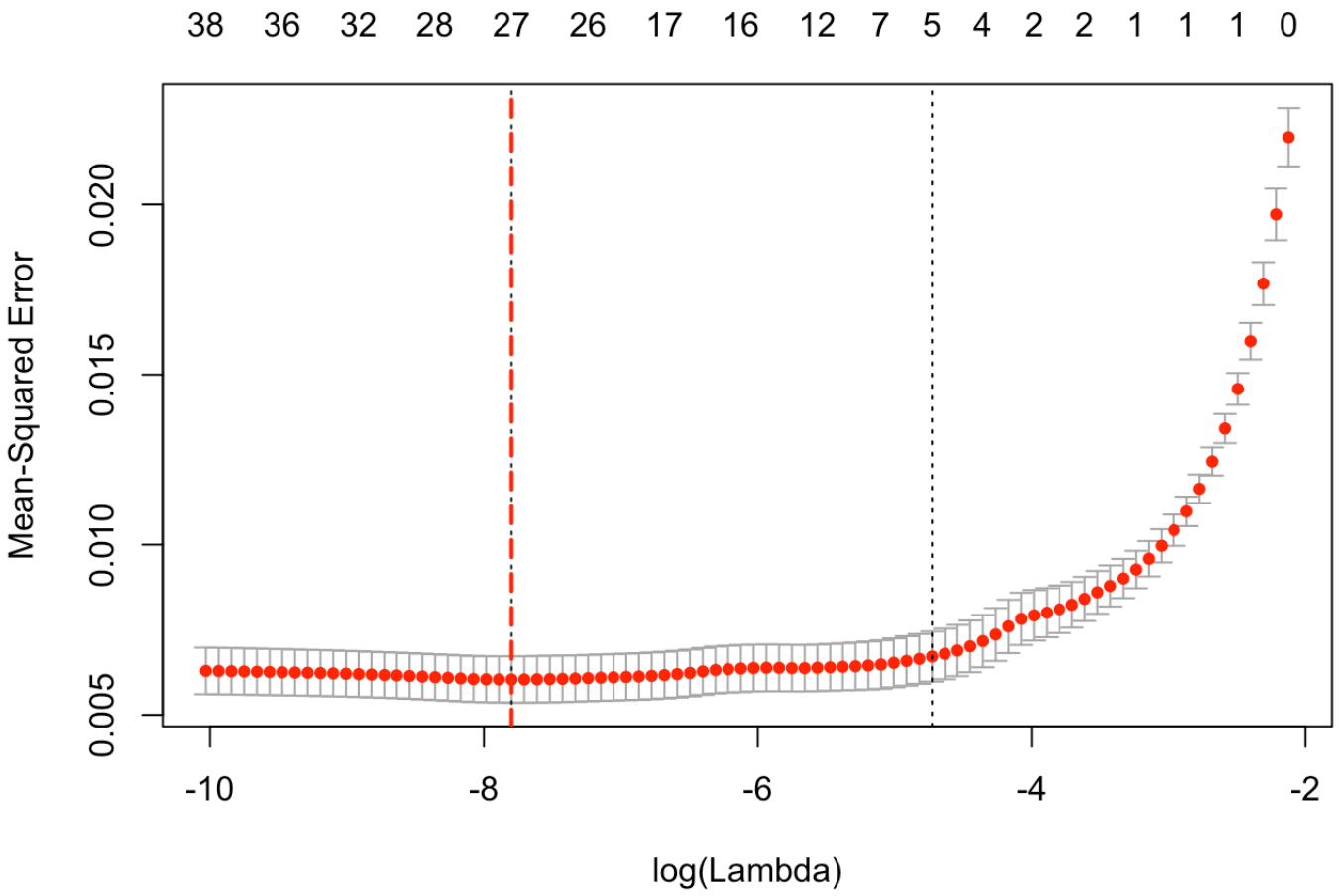
```
set.seed(1000)
lasso_fit1 = cv.glmnet(data.matrix(final_data2[, 6:43]), final_data2$TEAM_WIN_PCT, nf
olds = 10)

coef(lasso_fit1,s='lambda.min')
```

```
## 39 x 1 sparse Matrix of class "dgCMatrix"
##                              1
## (Intercept)      8.503891e-01
## DEF_RATING      -4.104532e-04
## NET_RATING       9.771160e-05
## AST_PCT          6.811820e-02
## AST_TO           .
## AST_RATIO        1.665728e-03
## OREB_PCT         .
## DREB_PCT         3.737321e-03
## REB_PCT          .
## TM_TOV_PCT       .
## TS_PCT           .
## USG_PCT          8.092458e-02
## PACE            -1.200128e-04
## PIE              4.830075e-02
## FGA              5.993350e-05
## FGA_PG          -5.563933e-03
## FG_PCT          -9.454083e-03
## GP_RANK          1.370754e-04
## W_RANK          -5.774548e-04
## L_RANK          -4.284982e-04
## W_PCT_RANK      -3.392524e-04
## MIN_RANK         1.043348e-04
## OFF_RATING_RANK -4.870125e-06
## DEF_RATING_RANK  6.732688e-06
## NET_RATING_RANK  .
## AST_PCT_RANK     4.048873e-05
## AST_TO_RANK      4.679456e-05
## AST_RATIO_RANK   .
## OREB_PCT_RANK   -2.530830e-05
## REB_PCT_RANK     .
## TM_TOV_PCT_RANK -5.971662e-05
## EFG_PCT_RANK     .
## TS_PCT_RANK     -9.566007e-05
## USG_PCT_RANK    -7.822694e-05
## PACE_RANK        .
## PIE_RANK         2.133935e-04
## FGA_RANK        -1.389565e-04
## FGA_PG_RANK      .
## FG_PCT_RANK     -8.844791e-05
```

```
coef(lasso_fit1, s = "lambda.1se")
```

```
## 39 x 1 sparse Matrix of class "dgCMatrix"
##                               1
## (Intercept)      7.657964e-01
## DEF_RATING       .
## NET_RATING       .
## AST_PCT          .
## AST_TO           .
## AST_RATIO        .
## OREB_PCT         .
## DREB_PCT         .
## REB_PCT          .
## TM_TOV_PCT       .
## TS_PCT           .
## USG_PCT          .
## PACE             .
## PIE              1.157642e-02
## FGA              .
## FGA_PG           .
## FG_PCT           .
## GP_RANK          .
## W_RANK          -2.611772e-04
## L_RANK          -2.377727e-04
## W_PCT_RANK      -5.002357e-04
## MIN_RANK         .
## OFF_RATING_RANK  .
## DEF_RATING_RANK  .
## NET_RATING_RANK -3.649688e-05
## AST_PCT_RANK     .
## AST_TO_RANK      .
## AST_RATIO_RANK   .
## OREB_PCT_RANK    .
## REB_PCT_RANK     .
## TM_TOV_PCT_RANK  .
## EFG_PCT_RANK     .
## TS_PCT_RANK      .
## USG_PCT_RANK     .
## PACE_RANK        .
## PIE_RANK         .
## FGA_RANK         .
## FGA_PG_RANK      .
## FG_PCT_RANK      .
```

```
plot(lasso_fit1)
abline(v=log(lasso_fit1$lambda.min),col='red',lwd=2,lty=2)
```



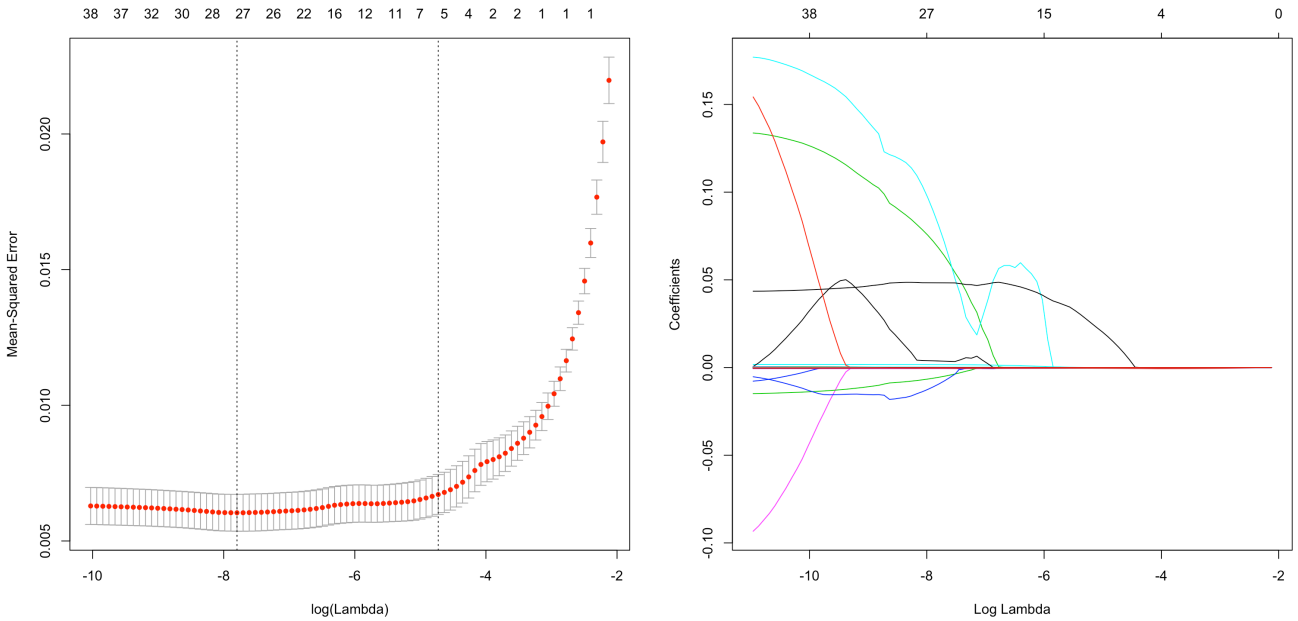

```
lasso_fit1$lambda.min
```

```
## [1] 0.0004106259
```

```
lasso_fit1$lambda.1se
```

```
## [1] 0.008846667
```

```
par(mfrow = c(1, 2))
plot(lasso_fit1)
plot(lasso_fit1$glmnet.fit, "lambda")
```



As one can see, games played, offensive rating, rebound percentage, field goals made, and several others were considered significant. However, for a more sparser model we used the second method using the one standard error threshold for optimal lambda value (log of lambda is .008).

We can see here only wins, losses, minutes, net rating, pie, win percentage rank, net rating rank and effective field goal pct rank made the final cut. This is understandable as net rating (difference between offensive rating and defensive rating), pie (better version of EFF), and efg pct rank are basically ratings that summarize various statistics together (for ex: efg is where they count three points twice as much than two points made) so therefore, including other variables would be redundant. This method seemed to be the optimal final model using Lasso regression.

We also attempted to used logistic regression, which is typically used for categorical variables. Win percentage is a quantitative variable, so it became obvious that log regression is not appropriate for this data set. The only variable that ended up being significant was win percentage rank. This project showed us how important it is to use an appropriate regression method; log regression was useless because the data was not categorical.

```
q= as.matrix(colnames(final_data_sort))

input <- final_data_sort[,c(q)]
a= glm(formula = TEAM_WIN_PCT ~ ., data = input, family = binomial)
```

```
## Warning in eval(family$initialize): non-integer #successes in a binomial
## glm!
```

```
summary(a)
```

```
##
## Call:
## glm(formula = TEAM_WIN_PCT ~ ., family = binomial, data = input)
```

```
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -0.61257  -0.07542  -0.00409   0.05713   0.74752
##
## Coefficients:
##                   Estimate Std. Error z value Pr(>|z|)
## (Intercept)      0.7327420 11.3750646   0.064    0.949
## GP              -0.0202883  0.0770202  -0.263    0.792
## W                0.0209499  0.1088509   0.192    0.847
## W_PCT            0.6442794  2.1157366   0.305    0.761
## MIN             -0.0234272  0.2058489  -0.114    0.909
## OFF_RATING       0.2566935  1.9377164   0.132    0.895
## DEF_RATING      -0.2562971  1.9368375  -0.132    0.895
## NET_RATING      -0.2541257  1.9383739  -0.131    0.896
## AST_PCT          0.7245588  3.0783517   0.235    0.814
## AST_TO          -0.0020713  0.1690470  -0.012    0.990
## AST_RATIO        0.0087123  0.0285849   0.305    0.761
## OREB_PCT        -1.3460043 15.0897166  -0.089    0.929
## DREB_PCT        -0.8906598 13.2034034  -0.067    0.946
## REB_PCT          2.6537050 27.2195062   0.097    0.922
## TM_TOV_PCT       0.0018586  0.0339279   0.055    0.956
## TS_PCT          -0.1639272  3.0588944  -0.054    0.957
## USG_PCT          0.7983956  6.8158966   0.117    0.907
## PACE             0.0069028  0.0583260   0.118    0.906
## PIE              0.1581770  0.6521198   0.243    0.808
## FGA              0.0005999  0.0019656   0.305    0.760
## FGA_PG          -0.0655990  0.1746674  -0.376    0.707
## FG_PCT           0.2106341  4.3830370   0.048    0.962
## GP_RANK          0.0008712  0.0033402   0.261    0.794
## W_RANK          -0.0024545  0.0096878  -0.253    0.800
## L_RANK          -0.0004287  0.0068948  -0.062    0.950
## W_PCT_RANK      -0.0003665  0.0032977  -0.111    0.912
## MIN_RANK        -0.0004544  0.0124082  -0.037    0.971
## OFF_RATING_RANK  0.0002285  0.0015996   0.143    0.886
## DEF_RATING_RANK  0.0001079  0.0014280   0.076    0.940
## NET_RATING_RANK -0.0001057  0.0020563  -0.051    0.959
## AST_PCT_RANK     0.0005977  0.0030253   0.198    0.843
## AST_TO_RANK      0.0001882  0.0016737   0.112    0.910
## AST_RATIO_RANK  -0.0001603  0.0027627  -0.058    0.954
## OREB_PCT_RANK   -0.0002767  0.0018929  -0.146    0.884
## REB_PCT_RANK     0.0002428  0.0022518   0.108    0.914
## TM_TOV_PCT_RANK -0.0003437  0.0014439  -0.238    0.812
## EFG_PCT_RANK     0.0003260  0.0023505   0.139    0.890
## TS_PCT_RANK     -0.0008228  0.0024914  -0.330    0.741
## USG_PCT_RANK    -0.0001258  0.0030189  -0.042    0.967
## PACE_RANK        0.0001459  0.0012821   0.114    0.909
## PIE_RANK         0.0011205  0.0017642   0.635    0.525
## FGA_RANK        -0.0011706  0.0063817  -0.183    0.854
## FGA_PG_RANK     -0.0011386  0.0058207  -0.196    0.845
```

```
## FG_PCT_RANK     -0.0003978  0.0023766  -0.167    0.867
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 47.958  on 528  degrees of freedom
## Residual deviance: 10.692  on 485  degrees of freedom
## AIC: 642.94
##
## Number of Fisher Scoring iterations: 4
```

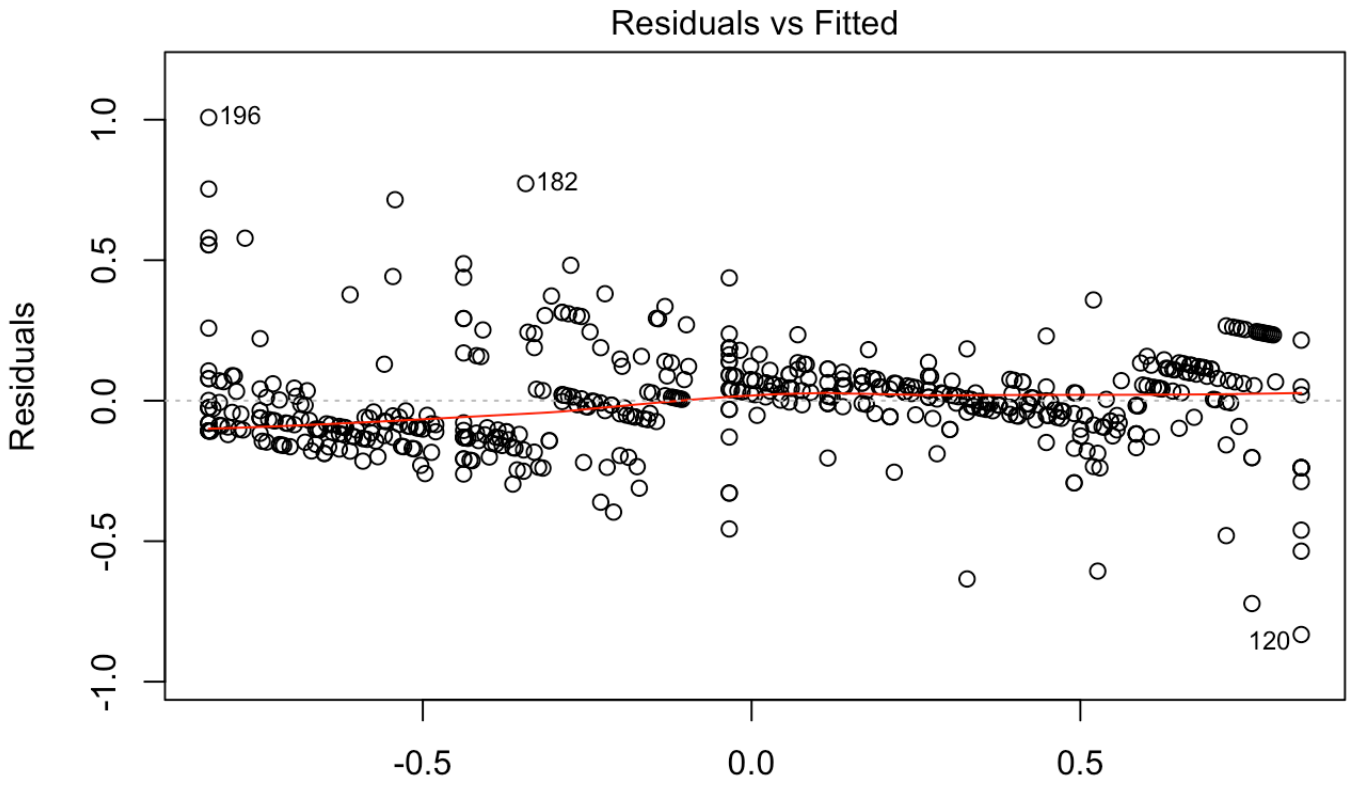
```
a= glm(formula = TEAM_WIN_PCT ~ W_PCT_RANK, data = final_data_sort, family = binomial
)
```

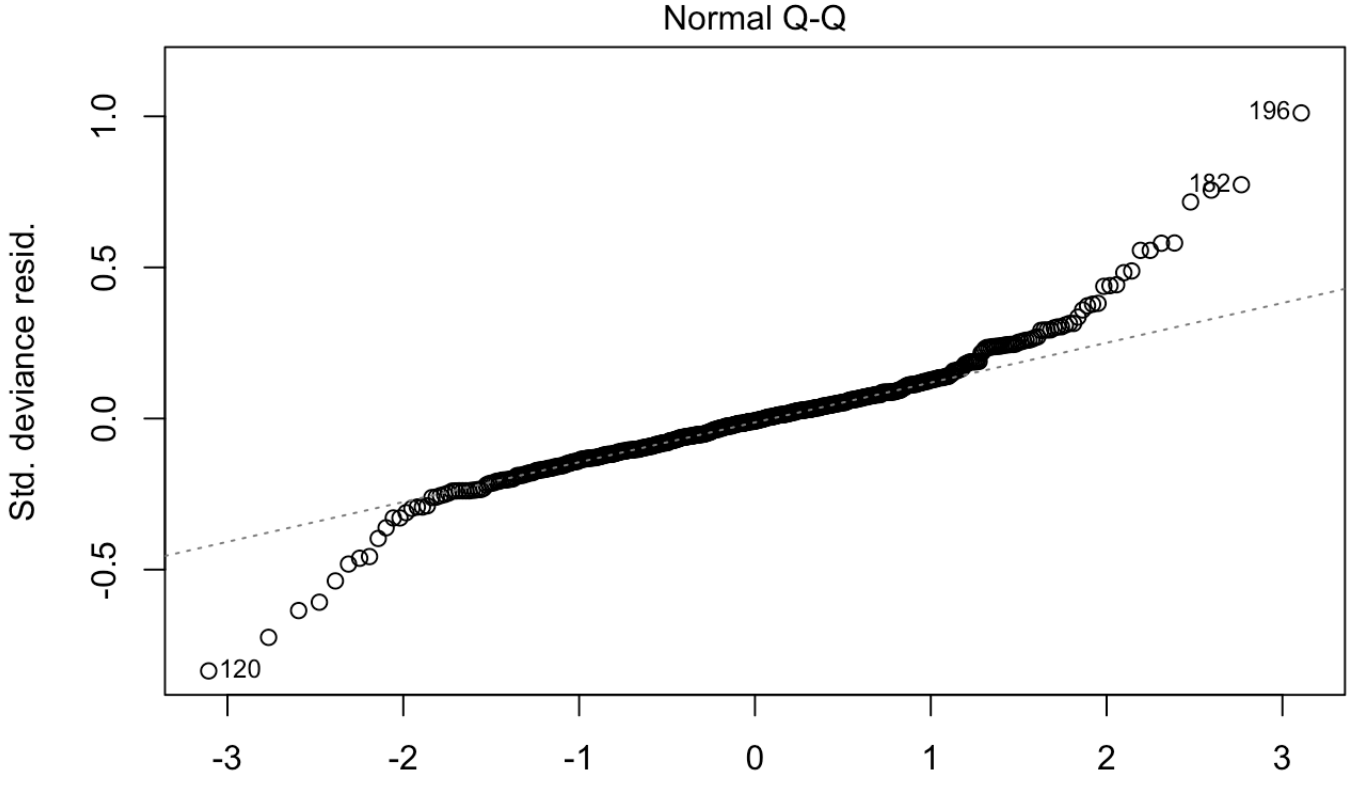
```
## Warning in eval(family$initialize): non-integer #successes in a binomial
## glm!
```

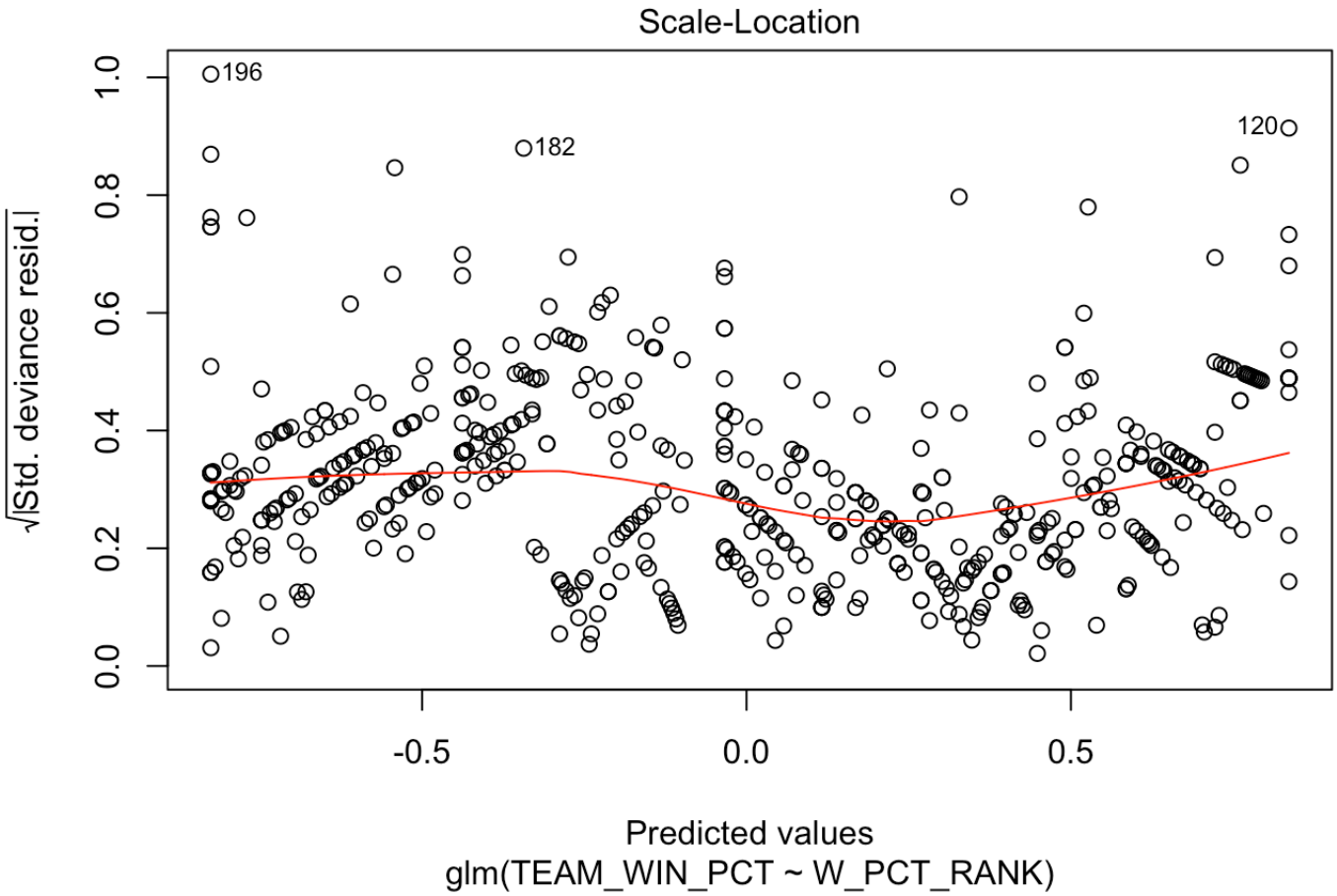
```
summary(a)
```

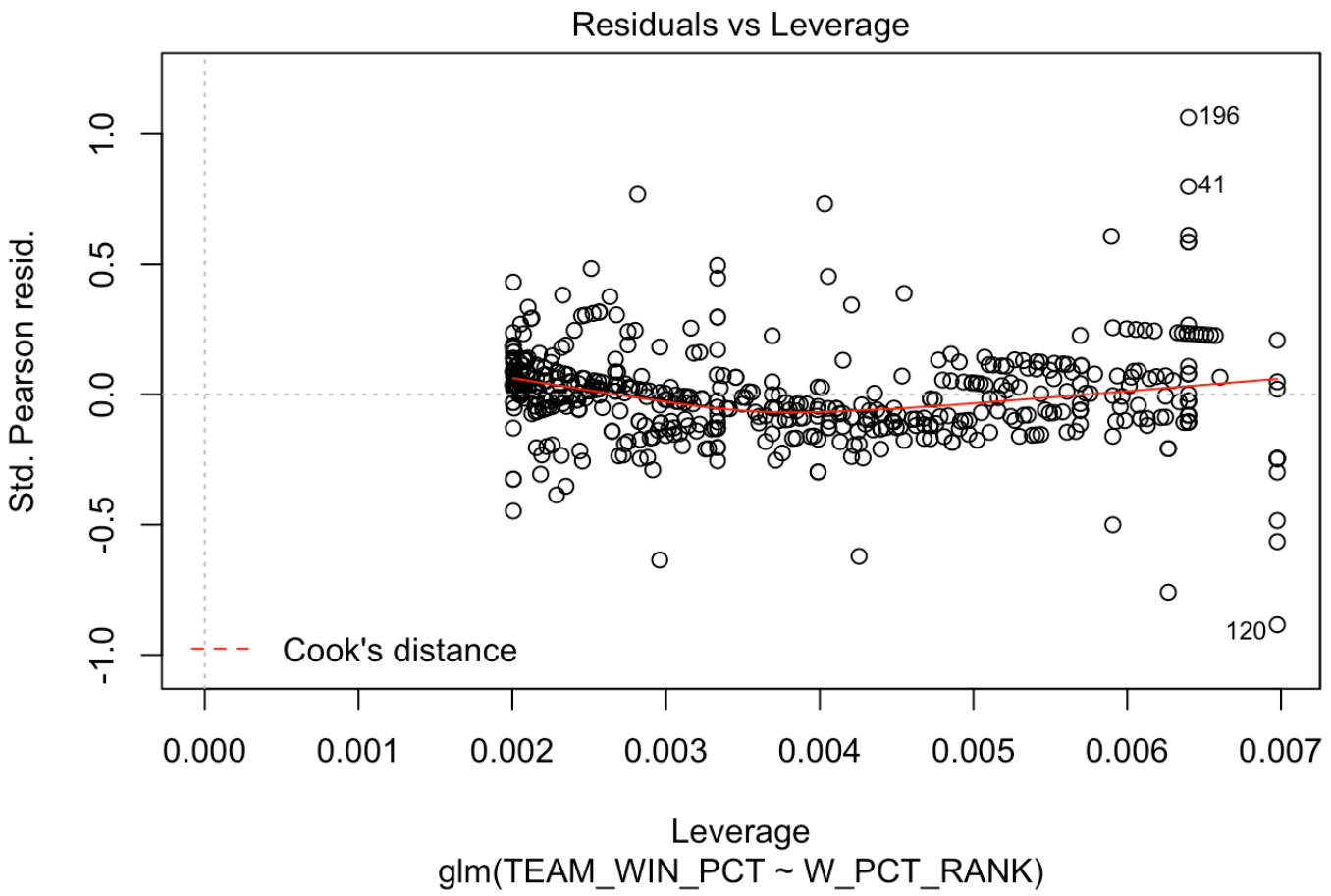
```
##
## Call:
## glm(formula = TEAM_WIN_PCT ~ W_PCT_RANK, family = binomial, data = final_data_sort
)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -0.83222  -0.10144  -0.00783   0.07591   1.00798
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept)  0.8392186  0.1823710   4.602 4.19e-06 ***
## W_PCT_RANK  -0.0032581  0.0006024  -5.409 6.34e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 47.958  on 528  degrees of freedom
## Residual deviance: 17.098  on 527  degrees of freedom
## AIC: 585.59
##
## Number of Fisher Scoring iterations: 3
```

```
plot(a)
```


Residuals vs Fitted
Residuals
196
182
120
1.0
0.5
0.0
-0.5
-1.0
-0.5
0.0
0.5
Predicted values
glm(TEAM_WIN_PCT ~ W_PCT_RANK)


Normal Q-Q
Std. deviance resid.
196
182
120
1.0
0.5
0.0
-0.5
-3
-2
-1
0
1
2
3


Theoretical Quantiles
glm(TEAM_WIN_PCT ~ W_PCT_RANK)
Scale-Location
196
182
120
√|Std. deviance resid.|
0.0
0.2
0.4
0.6
0.8
1.0
-0.5
0.0
0.5
Predicted values
glm(TEAM_WIN_PCT ~ W_PCT_RANK)



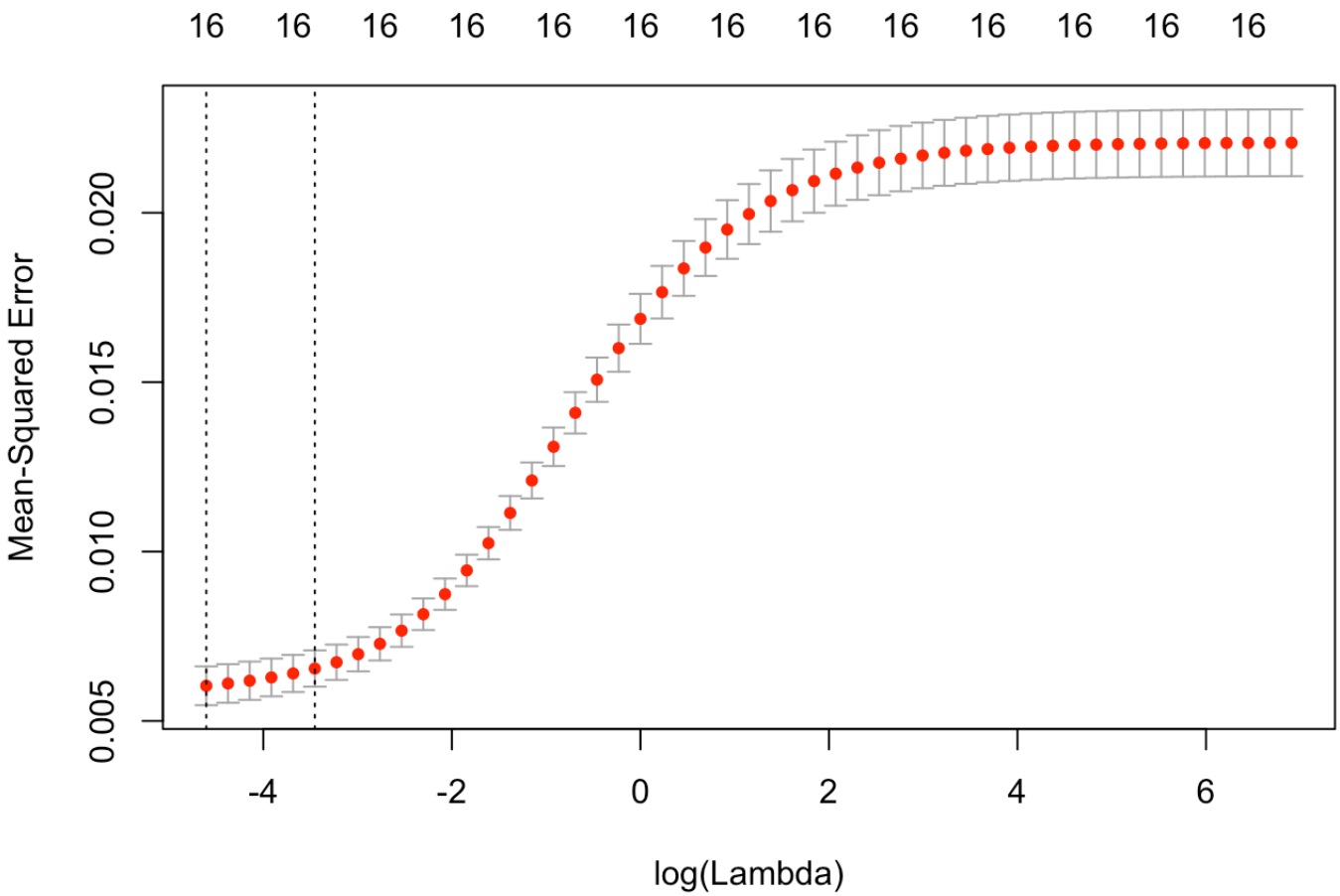
We learned about Bayesian regression in STAT 432; it concerns making statements about unknown quantities in terms of probabilities given the observed data and our prior knowledge. We performed a Bayesian regularized linear regression using conjugate priors. To do this, we used Bayesian model averaging (BMA) via the bas.lm() from the “BAS” package. Multiple models were averaged to obtain posteriors of coefficients and predictions from new data. We were also able to use this data to provide 95% confidence intervals for each variable.

Ridge regression is typically used to deal with multicollinearity in variables. This data set has many variables that are intrinsically related to one another, so we expected there to be multicollinearity. After running the ridge regression, we had a R-squared of .755743, which means that over 75% of the variation is explained by the model. We also discovered that the optimal lambda value is .01 after using the cv.glmnet function. This small lambda value meant that ridge regression was one of our most accurate models.

```
y = final_data_sort$TEAM_WIN_PCT
x = final_data_sort %>% select(GP, W, MIN, DEF_RATING, AST_RATIO, PIE, GP_RANK, W_RAN
K, W_PCT_RANK, AST_PCT_RANK, REB_PCT_RANK, TS_PCT_RANK, PIE_RANK, FGA_RANK, FGA_PG_RA
NK, USG_PCT) %>% data.matrix()
lambdas = 10^seq(3, -2, by = -.1)

fit = glmnet(x, y, alpha = 0, lambda = lambdas)
summary(fit)
```

```
##            Length Class      Mode
## a0          51    -none-     numeric
## beta       816    dgCMatrix  S4
## df          51    -none-     numeric
## dim          2    -none-     numeric
## lambda      51    -none-     numeric
## dev.ratio   51    -none-     numeric
## nulldev      1    -none-     numeric
## npasses      1    -none-     numeric
## jerr         1    -none-     numeric
## offset       1    -none-     logical
## call         5    -none-     call
## nobs         1    -none-     numeric
```

```
cv_fit = cv.glmnet(x, y, alpha = 0, lambda = lambdas)
plot(cv_fit)
```




```
opt_lambda = cv_fit$lambda.min
opt_lambda
```

```
## [1] 0.01
```

```
fit2 = cv_fit$glmnet.fit
summary(fit2)
```

```
##           Length Class     Mode
## a0         51    -none-    numeric
## beta      816    dgCMatrix S4
## df         51    -none-    numeric
## dim         2    -none-    numeric
## lambda     51    -none-    numeric
## dev.ratio  51    -none-    numeric
## nulldev     1    -none-    numeric
## npasses     1    -none-    numeric
## jerr        1    -none-    numeric
## offset      1    -none-    logical
## call        5    -none-    call
## nobs        1    -none-    numeric
```

```
y_predicted = predict(fit2, s = opt_lambda, newx = x)

# Sum of Squares Total and Error
sst = sum((y - mean(y))^2)
sse = sum((y_predicted - y)^2)

# R squared
rsq = 1 - sse / sst
rsq
```

```
## [1] 0.751191
```

Conclusion

Through all of our models and data analysis that we have conducted with this dataset, we have concluded that a team's assist ratio and defensive rating are the most important factors in determining their win percentage for the season. We also determined that our ridge regression model that had an optimal lambda of 0.01 and that our lasso regression model that had an optimal lambda of 0.0002 gave us the best fitted model of the five different models that we tested. When creating our models, we had high hopes for the ridge regression model because of how useful those types of models can be when dealing with a dataset that has high multicollinearity such as the dataset that we used for this project. Lasso regression can do very similar things as ridge regression, but can also remove variables automatically that it deems unnecessary to the model. So, considering the structure of our dataset and the types of variables that were included in it, it made

sense that these two types of regression models, ridge and lasso, ended up yielding the best models for our data. We feel that there were not really any pitfalls during our process of analyzing the data and there is not really anything that we would change going forward if we were to conduct a similar type of analysis.

Our analysis of a team's assist ratio and defensive rating being the most important factor of having a successful team can be supported by this year's NBA regular season standings and NBA Playoffs. There were eight teams that finished in the top 10 of both assist ratio and defensive rating. All eight of those teams went on to qualify for the NBA playoffs at the conclusion of the regular season. Furthermore, of these eight teams, five of them currently remain in the playoffs. This indicates that success in both of these two statistical categories not only will lead to regular season success, but also lead to great success in the playoffs and the ultimate goal of winning an NBA Championship.