Report (Note, these are screen shots);

Introduction:

The purpose of this assignment is to predict the wins of an MLB team based on specific stats. The data is provided to us at the start of the assignment and is considered not clean data. I will go through several sections to discuss the data, data preparation, model building, model selection and conclusions of findings.

Data Exploration

The data provided was the Moneyball data set. It included all integers with a variety of values. I have printed the summary data of the original information provided below.

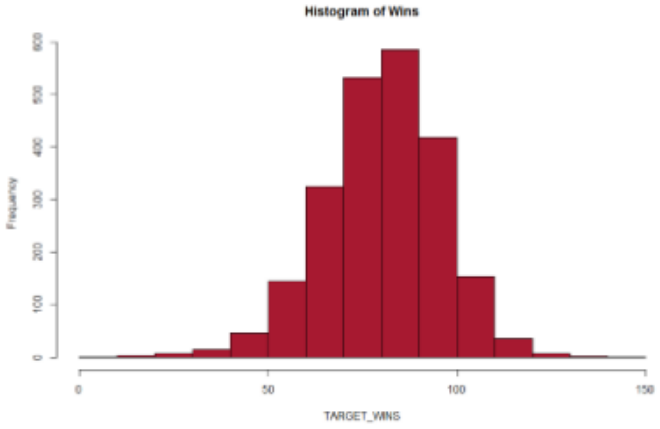
INDEX	TARGET_WINS	TEAM_BATTING_H	TEAM_BATTING_2B	TEAM_BATTING_3B
Min. : 1.0	Min. : 0.00	Min. : 891	Min. : 69.0	Min. : 0.00
1st Qu.: 630.8	1st Qu.: 71.00	1st Qu.:1383	1st Qu.:208.0	1st Qu.: 34.00
Median :1270.5	Median : 82.00	Median :1454	Median :238.0	Median : 47.00
Mean :1268.5	Mean : 80.79	Mean :1469	Mean :241.2	Mean : 55.25
3rd Qu.:1915.5	3rd Qu.: 92.00	3rd Qu.:1537	3rd Qu.:273.0	3rd Qu.: 72.00
Max. :2535.0	Max. :146.00	Max. :2554	Max. :458.0	Max. :223.00

TEAM_BATTING_HR	TEAM_BATTING_BB	TEAM_BATTING_SO	TEAM_BASERUN_SB	TEAM_BASERUN_CS
Min. : 0.00	Min. : 0.0	Min. : 0.0	Min. : 0.0	Min. : 0.0
1st Qu.: 42.00	1st Qu.:451.0	1st Qu.: 548.0	1st Qu.: 66.0	1st Qu.: 38.0
Median :102.00	Median :512.0	Median : 750.0	Median :101.0	Median : 49.0
Mean : 99.61	Mean :501.6	Mean : 735.6	Mean :124.8	Mean : 52.8
3rd Qu.:147.00	3rd Qu.:580.0	3rd Qu.: 930.0	3rd Qu.:156.0	3rd Qu.: 62.0
Max. :264.00	Max. :878.0	Max. :1399.0	Max. :697.0	Max. :201.0
		NA's :102	NA's :131	NA's :772

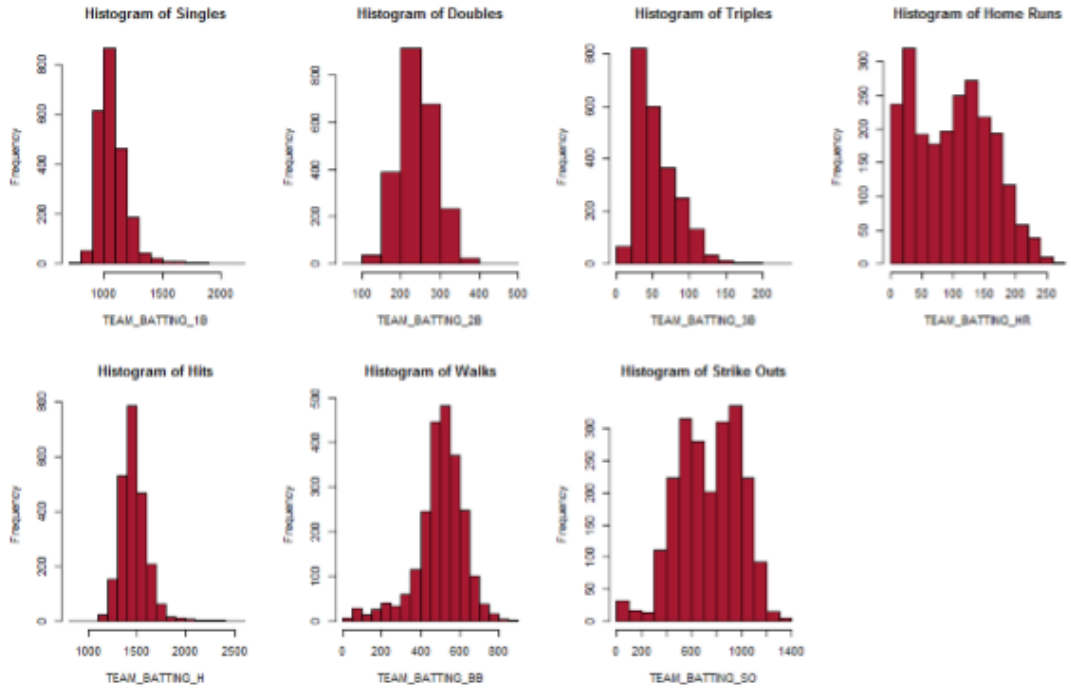
TEAM_BATTING_HBP	TEAM_PITCHING_H	TEAM_PITCHING_HR	TEAM_PITCHING_BB	TEAM_PITCHING_SO
Min. :29.00	Min. : 1137	Min. : 0.0	Min. : 0.0	Min. : 0.0
1st Qu.:50.50	1st Qu.: 1419	1st Qu.: 50.0	1st Qu.: 476.0	1st Qu.: 615.0
Median :58.00	Median : 1518	Median :107.0	Median : 536.5	Median : 813.5
Mean :59.36	Mean : 1779	Mean :105.7	Mean : 553.0	Mean : 817.7
3rd Qu.:67.00	3rd Qu.: 1682	3rd Qu.:150.0	3rd Qu.: 611.0	3rd Qu.: 968.0
Max. :95.00	Max. :30132	Max. :343.0	Max. :3645.0	Max. :19278.0
NA's :2085				NA's :102

```
TEAM_FIELDING_E  TEAM_FIELDING_DP
 Min.   :  65.0   Min.   : 52.0
 1st Qu.: 127.0   1st Qu.:131.0
 Median : 159.0   Median :149.0
 Mean   : 246.5   Mean   :146.4
 3rd Qu.: 249.2   3rd Qu.:164.0
 Max.   :1898.0   Max.   :228.0
                  NA's   :286
```

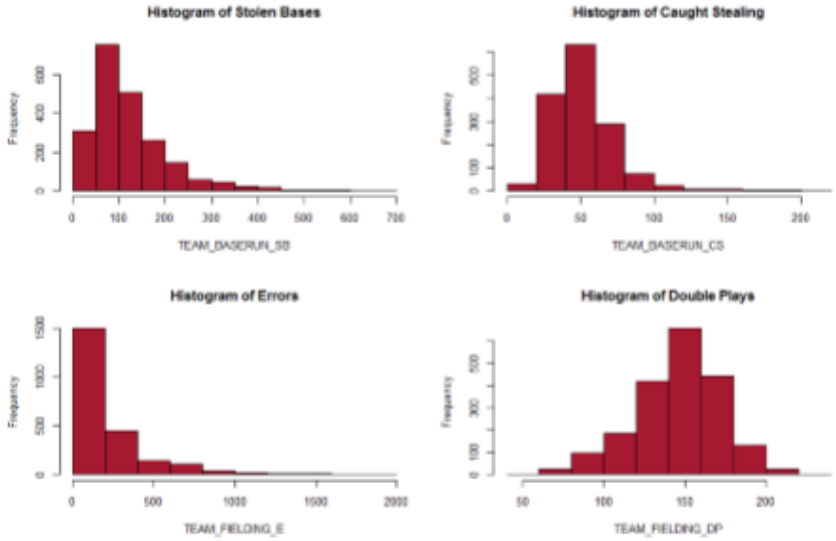
Above I have noted a print out of the various variables available in the data set. There are a total of 2,276 rows in the data set. In looking at the above summary print out I notice several issues with the data. The variables fielding double plays, pitching strike outs, batting hit by pitch, stolen bases, caught stealing, batting strike outs, and batting walks all have a lot of NAs, or missing values. The batting hit by pitch had the most with over 2085 missing values. There also seems to be some incorrect data in multiple variables target wins, hits, doubles, triples, home runs, walks, strike outs, stolen bases, caught stealing, pitcher home runs, walks and strikeouts. All of these are out of characteristic for a 162 game season. It would be near impossible for a team to not have a strikeout, walk, or home run for 162 games. Below are the distribution graphs for each of the variables.



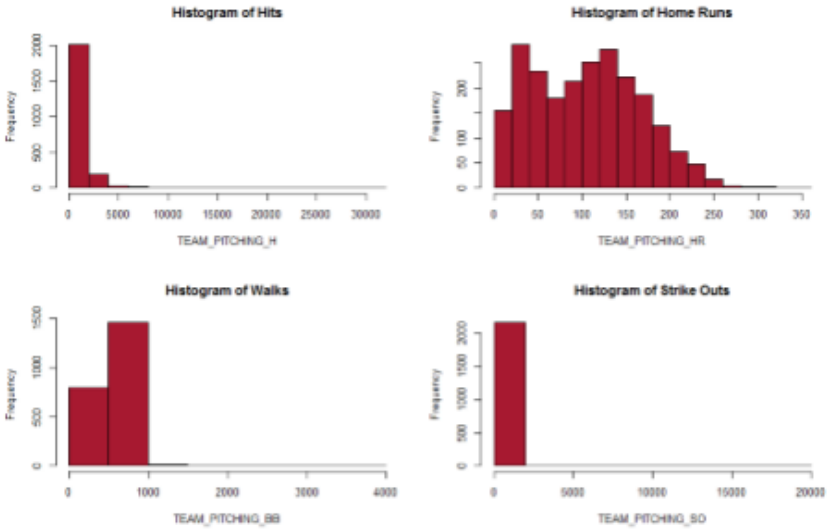
Above is the variable target wins. This variable is the goal to predict properly in the assignment. There appears to be a pretty normal distribution, with some significant outliers impacting the result. This was fairly apparent from the summary above, but the histogram shows a good representation of that data. Next, I will show the batting statistics and how they are distributed.



The hitting statistics have a unique distribution. Singles and triples have strong positive skews. The hits statistic has a weaker positive skew. Then walks has a negative skew. Home runs and strikeouts are platykurtic, and lastly doubles appears to be more normally distributed.



The next grouping of variables are fielding and base running statistics. Errors, stolen bases, and caught stealing all have a pretty significant positive skew. While double plays have a slight negative skew. These variables are more volatile due to missing values, as pointed out in the summary statistics. I am unsure if these values are true stats from baseball or not, but these stats have not always been tracked throughout the years. The last statistics being examined are pitching statistics.

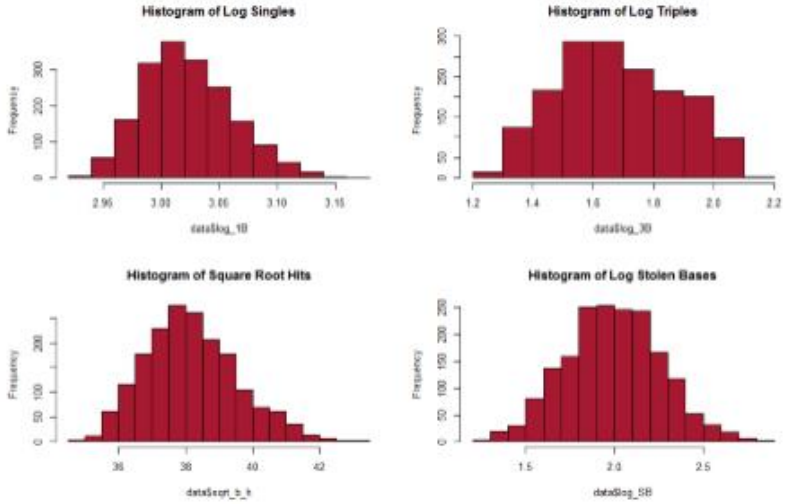


These graphs above are the pitching statistics as they were originally delivered. The summary statistics showed some extreme outliers and they are apparent in the graphs above. I will be trimming the data to get rid of these extreme outliers. This is due to how extreme these data points are and the impossibility of these statistics occurring in baseball.

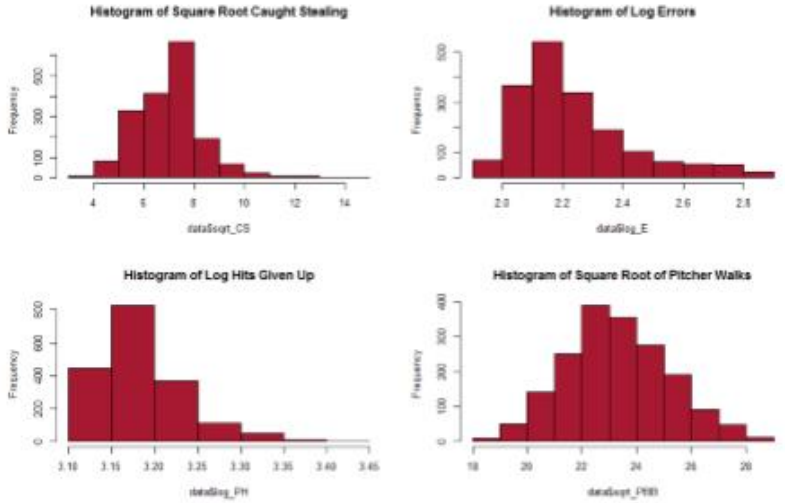
Data Preparation:

The first step I did to prepare for the model was to trim the data by .5% on both sides. There are apparent outliers in the data set and appears that information had been misreported. Since there was a significant sample to work form, it allowed for me to feel comfortable to do this. I then started to normalize the distribution of the variables by applying log and sqrt transformations on each variable according to the skew. I also created several new variables: slugging, pitching hits and walks, singles, and hitting walks and hits.

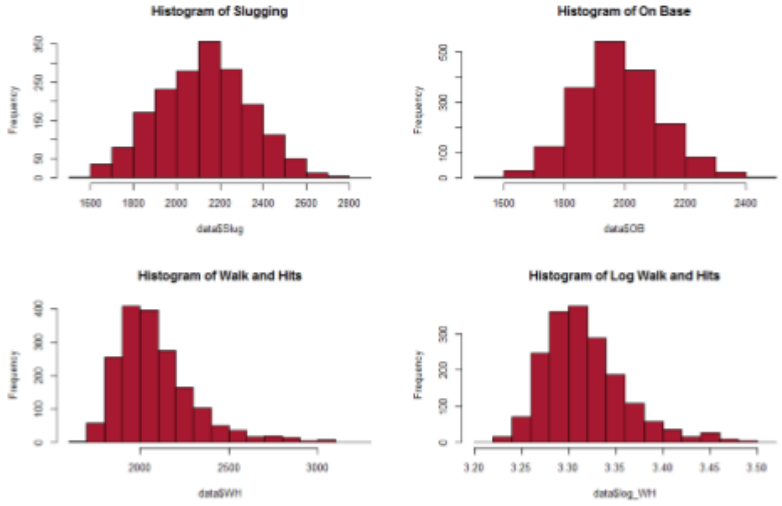
After adding these variables, I decided to not utilize the raw variable for hit by pitches, due to most of the sample did not have values for this. I did create a flag variable to be utilized, to allow for the model to understand where there was a variable and one that did not have this variable. The ability to impute requires to have a good size sample and with the statistics in baseball changing so much throughout generations, I did not feel comfortable with imputing this variable. The variables batting and pitching strikeouts, fielding double plays, stolen bases and base runners caught stealing I imputed using regression trees. There was enough data for these variables to feel comfortable regressing those variables. Below I will show how the different changes done look graphically, to illustrate how the data was impacted.



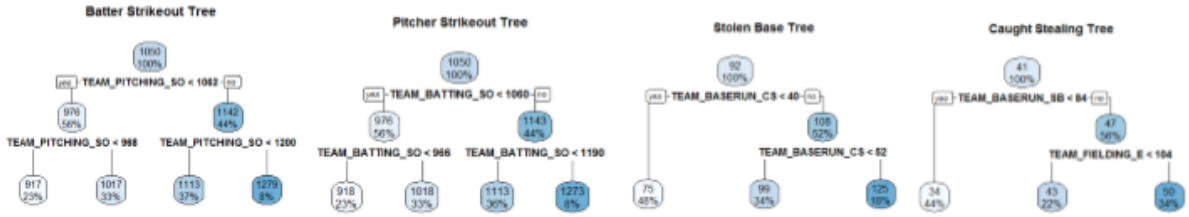
Above are what the distribution now looks like for Log Singles, Log Triples, Log Stolen Bases and Square Root Hits. I decided to do these transformations due to the large skew of the data on the original data. Above you can see that they have become more normalized, though there is still a skew.

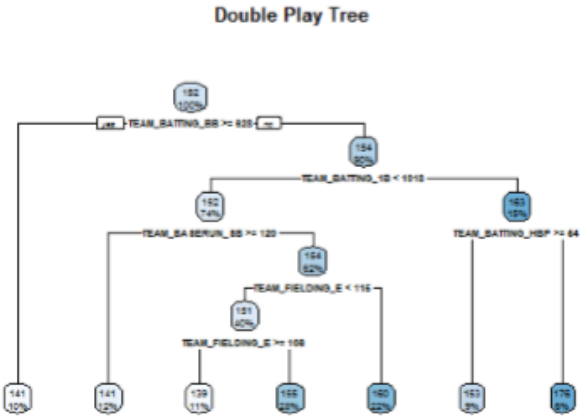


The above are the transformed variables for Pitcher Walks, Pitcher Hits Given Up, Fielder Errors and Runners Caught Stealing. The transformation helped normalize these histograms, but there is still a large positive skew on the data for Fielder Errors and Pitcher Hits Given Up. Due to how many distinct values available in the data, I decided to accept the skew on these variables. Now I will show the new variables that were added to the data set: Slugging, Walk and Hits, and On Base.



The above are the added variables. Slugging and On Base both have a pretty normalized curve. Pitcher Walk and Hits has a strong positive skew. I transformed this variable and the results are shown on the right. The final touches for the variables from this exercise was imputing the values for all the missing variables except hit by pitch, which I excluded. I utilized rpart tree to impute the variables, with pruning at .05. Below are the tree diagrams to show how the imputation occurred.





The trees above show that strikeouts seem to be correlated among pitching and hitting. Stolen bases and caught stealing are also correlated. Double plays did not have a direct linkage to another stat so it need to go through several other stats to come to a decision. The final step was the creation of flag variables for all imputed variables. This will allow the model to understand when it was done and to determine if there was any correlation in the data being missing and a time frame of it occurring. If the stats mirror baseball purely, then this would be very valuable, since different decades played the game differently.

Model Build:

I built six different models and tested different techniques. The first thing I switched up was the way to remove outliers. At first, I tried to use the baseball almanac to eliminate bad data, since there are record lows and highs in each of these stats. I then built a linear model with just the basic stats given. Then I started to build as many variables as I possibly could. I did every transformation listed in the book and added them to data set. Then utilized stepwise AIC model to decide on the model. This produced a really high adjusted r-squared score, but when I utilized it against the test data, that I had sectioned off, it did not perform as well. Then I did a more diligent approach, and utilized a procedure to get a high adjusted r-squared score, while choosing the variables that were correlated the most with the residuals. This yielded a better model in prediction, but still was not the best, because it relied too much on as many variables as possible. Then I did a trim of data to eliminate the extreme outliers in the data set, and found that the output was much closer to what I would like to see. The mean target wins and mean square errors were more in line what I was looking for. I have pasted several of the outputs below to review.

```
lm(formula = TARGET_WINS ~ TEAM_BATTING_2B + TEAM_BATTING_3B +
    TEAM_BATTING_HR + TEAM_BATTING_BB + TEAM_PITCHING_H + TEAM_PITCHING_HR +
    TEAM_PITCHING_BB + TEAM_FIELDING_E + IMP_BR_SB + M_BR_SB +
    IMP_BA_SO + M_BA_SO + IMP_PI_SO + M_DP + M_CS + log_2b +
    log_bb + log_so + log_cs + log_sb + log_e + log_dp, data = train)

Residuals:
    Min      1Q  Median      3Q     Max
-36.616  -7.007   0.091   6.820  27.037

Coefficients:
                  Estimate Std. Error t value Pr(>|t|)
(Intercept)       35.563172 118.029327   0.301 0.763226
TEAM_BATTING_2B   -0.251452   0.071212  -3.531 0.000428 ***
TEAM_BATTING_3B    0.169818   0.022627   7.505 1.11e-13 ***
TEAM_BATTING_HR    0.614596   0.129281   4.754 2.21e-06 ***
TEAM_BATTING_BB    0.280957   0.044098   6.371 2.57e-10 ***
TEAM_PITCHING_H    0.037152   0.004533   8.195 5.79e-16 ***
TEAM_PITCHING_HR  -0.524148   0.123960  -4.228 2.51e-05 ***
TEAM_PITCHING_BB  -0.177331   0.032957  -5.381 8.75e-08 ***
TEAM_FIELDING_E   -0.076453   0.016078  -4.755 2.20e-06 ***
IMP_BR_SB          0.093391   0.012027   7.765 1.61e-14 ***
M_BR_SB          -11.707285   5.340090  -2.192 0.028526 *
IMP_BA_SO         -0.140334   0.022985  -6.105 1.34e-09 ***
M_BA_SO          -17.320768   2.808719  -6.167 9.21e-10 ***
IMP_PI_SO          0.095233   0.022926   4.154 3.47e-05 ***
M_DP             -13.633283   2.642991  -5.158 2.87e-07 ***
M_CS              -5.437946   1.157268  -4.699 2.88e-06 ***
log_2b           115.683128  39.424499   2.934 0.003400 **
log_bb           -71.951782  39.886236  -1.804 0.071467 .
log_so            46.584990  16.203921   2.875 0.004105 **
```

```
log_cs             4.541250   2.908232   1.562 0.118638
log_sb            -9.197037   3.522482  -2.611 0.009130 **
log_e            -22.598849   8.512363  -2.655 0.008029 **
log_dp           -41.065882   5.133688  -7.999 2.68e-15 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 10.5 on 1342 degrees of freedom
Multiple R-squared:  0.3998,   Adjusted R-squared:   0.39
F-statistic: 40.63 on 22 and 1342 DF,  p-value: < 2.2e-16
```

```
    Test MSE        Train MSE       Train AIC
 Min.   :111.7   Min.   :102.0   Min.   :10237
 1st Qu.:114.9   1st Qu.:105.6   1st Qu.:10283
 Median :117.6   Median :106.4   Median :10297
 Mean   :118.9   Mean   :106.3   Mean   :10294
 3rd Qu.:122.0   3rd Qu.:107.7   3rd Qu.:10313
 Max.   :130.9   Max.   :108.5   Max.   :10319
```

The above is the printout of linear model 1. The adjusted r-squared is .39 which for this data set was pretty good. It showed well on the training data set with AIC and MSE. The Test MSE though was not really as good as the trained data set. The variables of the linear model that stick out to me are doubles, and log double and walks. The doubles variable has a negative value associated, though it is one of the best kinds of hits in the game, and are usually related with runs in baseball. This is counteracted by the large positive number for log doubles, but it has some concern because this appears to be trying to fit the data too tightly and may be at risk of overfit. The next model took in all of the variables possible.

```
lm(formula = TARGET_WINS ~ TEAM_BATTING_3B + TEAM_BATTING_HR +
    TEAM_BATTING_BB + TEAM_PITCHING_H + TEAM_PITCHING_HR + TEAM_PITCHING_BB +

    TEAM_FIELDING_E + IMP_BR_SB + IMP_BA_SO + M_BA_SO + IMP_PI_SO +
    IMP_DP + M_DP + IMP_CS + M_CS + log_slug + log_ob + log_wh +
    log_2b + log_so + log_cs + log_pso + log_e + log_dp + sqrt_slug +
    sqrt_wh + sqrt_1b + sqrt_3b + sqrt_bb + sqrt_so + sqrt_ph +
    sqrt_pso + sqrt_e + sqrt_dp + plog_slug + plog_h + plog_2b +
    plog_3b + plog_so + plog_cs + plog_dp + psqrt_slug + psqrt_h +
    psqrt_1b + psqrt_2b + psqrt_so + psqrt_cs + psqrt_dp, data = train)

Residuals:
    Min      1Q     Median     3Q     Max
  -34.058  -6.582   0.012   6.527  31.078

Coefficients:
                   Estimate Std. Error t value Pr(>|t|)
(Intercept)       1.633e+04  9.534e+03   1.713 0.086932 .
TEAM_BATTING_3B   4.560e+00  7.965e-01   5.725 1.28e-08 ***
TEAM_BATTING_HR   6.190e+00  1.137e+00   5.442 6.30e-08 ***
TEAM_BATTING_BB   3.151e-01  7.863e-02   4.007 6.49e-05 ***
TEAM_PITCHING_H  -2.772e+00  8.581e-01  -3.231 0.001265 **
TEAM_PITCHING_HR -3.795e-01  1.269e-01  -2.990 0.002845 **
TEAM_PITCHING_BB -2.499e+00  8.391e-01  -2.978 0.002954 **
TEAM_FIELDING_E   2.113e-01  1.360e-01   1.554 0.120342
IMP_BR_SB         7.368e-02  7.252e-03  10.160  < 2e-16 ***
IMP_BA_SO         3.892e+00  1.053e+00   3.696 0.000228 ***
M_BA_SO          -1.953e+01  4.678e+00  -4.176 3.17e-05 ***
```

```
IMP_PI_SO        -8.401e-01  2.517e-01  -3.337 0.000869 ***
IMP_DP           -3.441e+01  1.238e+01  -2.781 0.005504 **
M_DP             -1.565e+01  2.899e+00  -5.398 8.00e-08 ***
IMP_CS           -1.263e+00  4.407e-01  -2.867 0.004215 **
M_CS             -5.619e+00  1.149e+00  -4.891 1.13e-06 ***
log_slug          3.742e+03  1.218e+03   3.072 0.002170 **
log_ob            2.934e+03  1.079e+03   2.718 0.006660 **
log_wh           -1.454e+04  4.417e+03  -3.292 0.001021 **
log_2b            6.056e+02  1.119e+02   5.413 7.38e-08 ***
log_so            3.910e+03  9.103e+02   4.295 1.88e-05 ***
log_cs            2.271e+01  1.079e+01   2.105 0.035508 *
log_pso          -1.747e+03  4.807e+02  -3.634 0.000290 ***
log_e             1.467e+02  7.748e+01   1.894 0.058472 .
log_dp           -5.651e+03  2.343e+03  -2.413 0.015981 *
sqrt_slug        -2.066e+02  5.029e+01  -4.108 4.24e-05 ***
sqrt_wh           4.370e+02  1.584e+02   2.759 0.005882 **
sqrt_1b           2.763e+01  1.409e+01   1.961 0.050070 .
sqrt_3b          -4.014e+00  2.521e+00  -1.592 0.111537
sqrt_bb          -6.437e+00  4.050e+00  -1.589 0.112241
sqrt_so          -2.907e+02  7.023e+01  -4.139 3.71e-05 ***
sqrt_ph           9.863e+01  4.555e+01   2.165 0.030549 *
sqrt_pso          1.070e+02  2.877e+01   3.718 0.000209 ***
sqrt_e           -1.918e+01  8.773e+00  -2.186 0.028964 *
sqrt_dp           1.036e+03  4.112e+02   2.518 0.011910 *
plog_slug        -9.247e+01  3.805e+01  -2.430 0.015229 *
plog_h            9.697e+01  4.825e+01   2.010 0.044661 *
plog_2b           6.843e+01  1.834e+01   3.732 0.000198 ***
plog_3b           1.858e+01  6.789e+00   2.736 0.006296 **
plog_so          -5.053e+02  1.701e+02  -2.971 0.003021 **
plog_cs           6.320e+01  2.990e+01   2.114 0.034710 *
plog_dp           5.819e+02  1.633e+02   3.564 0.000379 ***
psqrt_slug        1.669e+01  4.912e+00   3.397 0.000701 ***
psqrt_h          -1.932e+01  8.338e+00  -2.318 0.020630 *
psqrt_1b         -2.166e+00  1.432e+00  -1.513 0.130426
psqrt_2b         -2.715e+01  6.222e+00  -4.363 1.38e-05 ***
psqrt_so          6.650e+01  2.142e+01   3.105 0.001942 **
psqrt_cs         -2.910e+01  1.067e+01  -2.727 0.006487 **
psqrt_dp         -2.037e+02  5.909e+01  -3.447 0.000585 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 10.05 on 1298 degrees of freedom
Multiple R-squared:  0.4438,   Adjusted R-squared:  0.4232
F-statistic: 21.57 on 48 and 1298 DF,  p-value: < 2.2e-16
```

TEST MSE	TRAIN MSE	TRAIN AIC
139.03076	97.37248	10089.91868

The above model was extremely overfit. You can see that the training stats are great with a below 100 MSE for the training set. When I tested it against data withheld it did not do nearly as well. You can also see signs of being overfit by the negative numbers for home runs, walks and hits which are traditionally the strongest correlation with runs scoring. There is a balance occurring between each variable adding and subtracting through the regression equation. The last model that I will illustrate in this section is the model I will discuss in my selection section.

```
lm(formula = TARGET_WINS ~ TEAM_BATTING_H + TEAM_BATTING_2B +
    TEAM_BATTING_3B + TEAM_BATTING_HR + TEAM_BATTING_BB + TEAM_PITCHING_H +
    TEAM_PITCHING_HR + TEAM_FIELDING_E + IMP_BR_SB + M_BR_SB +
    IMP_BA_SO + M_BA_SO + IMP_PI_SO + IMP_DP + M_DP + IMP_CS +
    M_CS + log_1B + sqrt_CS + log_E + sqrt_PBB, data = train)

Residuals:
    Min      1Q  Median      3Q     Max
-36.769  -7.234   0.006   7.107  29.558

Coefficients:
                  Estimate Std. Error t value Pr(>|t|)
(Intercept)     -7.396e+02  3.811e+02  -1.941 0.052507 .
TEAM_BATTING_H  -1.799e-01  6.318e-02  -2.847 0.004478 **
TEAM_BATTING_2B  1.226e-01  5.989e-02   2.048 0.040749 *
TEAM_BATTING_3B  3.301e-01  6.506e-02   5.074 4.45e-07 ***
TEAM_BATTING_HR  6.841e-01  1.480e-01   4.623 4.16e-06 ***
TEAM_BATTING_BB  2.465e-01  4.451e-02   5.538 3.68e-08 ***
TEAM_PITCHING_H  4.967e-02  1.357e-02   3.660 0.000262 ***
TEAM_PITCHING_HR -4.214e-01  1.320e-01  -3.193 0.001442 **
TEAM_FIELDING_E -8.816e-02  1.816e-02  -4.854 1.35e-06 ***
IMP_BR_SB        7.683e-02  7.402e-03  10.380  < 2e-16 ***
M_BR_SB         -1.825e+01  6.195e+00  -2.947 0.003267 **
IMP_BA_SO       -1.043e-01  2.456e-02  -4.246 2.33e-05 ***
M_BA_SO         -2.089e+01  2.722e+00  -7.673 3.23e-14 ***
IMP_PI_SO        8.191e-02  2.315e-02   3.539 0.000416 ***
IMP_DP          -1.010e-01  1.581e-02  -6.391 2.27e-10 ***
M_DP            -1.259e+01  2.845e+00  -4.426 1.04e-05 ***
IMP_CS           2.196e-01  8.676e-02   2.531 0.011495 *
M_CS            -3.518e+00  1.170e+00  -3.007 0.002684 **
log_1B           3.865e+02  1.452e+02   2.662 0.007869 **
sqrt_CS         -3.483e+00  1.405e+00  -2.480 0.013277 *
log_E           -1.785e+01  9.123e+00  -1.956 0.050655 .
sqrt_PBB        -9.474e+00  1.962e+00  -4.828 1.54e-06 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 10.71 on 1339 degrees of freedom
Multiple R-squared:  0.3885,   Adjusted R-squared:  0.3789
F-statistic:  40.5 on 21 and 1339 DF,  p-value: < 2.2e-16
       Test MSE        Train MSE       Train AIC
     Min.   :105.5   Min.   :105.5   Min.   :10249
     1st Qu.:111.6   1st Qu.:109.7   1st Qu.:10300
     Median :113.3   Median :110.7   Median :10315
     Mean   :113.8   Mean   :110.5   Mean   :10311
     3rd Qu.:116.6   3rd Qu.:111.1   3rd Qu.:10323
     Max.   :129.0   Max.   :113.0   Max.   :10342
```

Above is the last model. Though it's adjusted r-squared is not as high as the model with the most variables, it follows a good logic when looking at the variables. The singles, doubles, triples and home runs all have a positive impact on target wins. Pitching hits and home runs are negative. Also, this model's test scores show to be pretty strong.

Model Selection and Conclusion:

I selected the final model for several reasons. It did well on the withheld test data comparatively. When running the model, I did not have to impute or create a floor or ceiling for target wins for the model. This means it probably won't overfit by a lot. The variables themselves are easy to understand and the impact on linear regression makes sense. The mean reported target wins ended up being just above 80, which is in line with 81 being the mean ground for wins in baseball. I think this model would be usable and would help make some judgements on teams and where they will be.

Code:

```
#Program written by Student

library(rpart)

library(rJava)

library(readr)

library(pbkrtest)

library(car)

library(leaps)

library(MASS)

library(xlsxjars)

library(xlsx)

library(moments)


data=read.csv("C:/Users/NAME/Desktop/School/411/Module 1/moneyball.csv",header=T)

summary(data)


mse <- function(sm)

  mean(sm$residuals^2)


par(mfrow=c(1,1))

hist(data$TARGET_WINS, col = "#A71930", xlab = "TARGET_WINS", main = "Histogram of Wins")

boxplot(data$TARGET_WINS, col = "#A71930", main = "Boxplot of Wins")

par(mfrow = c(1,1))

data$TEAM_BATTING_1B=data$TEAM_BATTING_H-data$TEAM_BATTING_2B-
data$TEAM_BATTING_3B-data$TEAM_BATTING_HR

par(mfrow=c(2,4))

hist(data$TEAM_BATTING_1B, col = "#A71930", xlab = "TEAM_BATTING_1B", main = "Histogram of
Singles")

#boxplot(data$TEAM_BATTING_1B, col = "#A71930", main = "Boxplot of TEAM_BATTING_1B")
```

```
hist(data$TEAM_BATTING_2B, col = "#A71930", xlab = "TEAM_BATTING_2B", main = "Histogram of Doubles")

#boxplot(data$TEAM_BATTING_2B, col = "#A71930", main = "Boxplot of TEAM_BATTING_2B")

hist(data$TEAM_BATTING_3B, col = "#A71930", xlab = "TEAM_BATTING_3B", main = "Histogram of Triples")

#boxplot(data$TEAM_BATTING_3B, col = "#A71930", main = "Boxplot of TEAM_BATTING_3B")

hist(data$TEAM_BATTING_HR, col = "#A71930", xlab = "TEAM_BATTING_HR", main = "Histogram of Home Runs")

skewness(data$TEAM_BATTING_HR)

#boxplot(data$TEAM_BATTING_HR, col = "#A71930", main = "Boxplot of TEAM_BATTING_HR")

hist(data$TEAM_BATTING_H, col = "#A71930", xlab = "TEAM_BATTING_H", main = "Histogram of Hits")

#boxplot(data$TEAM_BATTING_H, col = "#A71930", main = "Boxplot of TEAM_BATTING_H")

skewness(data$TEAM_BATTING_H)

hist(data$TEAM_BATTING_BB, col = "#A71930", xlab = "TEAM_BATTING_BB", main = "Histogram of Walks")

#boxplot(data$TEAM_BATTING_HR, col = "#A71930", main = "Boxplot of TEAM_BATTING_BB")

hist(data$TEAM_BATTING_SO, col = "#A71930", xlab = "TEAM_BATTING_SO", main = "Histogram of Strike Outs")

#boxplot(data$TEAM_BATTING_SO, col = "#A71930", main = "Boxplot of TEAM_BATTING_SO")


par(mfrow=c(2,2))

hist(data$TEAM_BASERUN_SB, col = "#A71930", xlab = "TEAM_BASERUN_SB", main = "Histogram of Stolen Bases")

#boxplot(data$TEAM_BASERUN_SB, col = "#A71930", main = "Boxplot of TEAM_BASERUN_SB")

hist(data$TEAM_BASERUN_CS, col = "#A71930", xlab = "TEAM_BASERUN_CS", main = "Histogram of Caught Stealing")

#boxplot(data$TEAM_BASERUN_CS, col = "#A71930", main = "Boxplot of TEAM_BASERUN_CS")

hist(data$TEAM_FIELDING_E, col = "#A71930", xlab = "TEAM_FIELDING_E", main = "Histogram of Errors")

#boxplot(data$TEAM_FIELDING_E, col = "#A71930", main = "Boxplot of TEAM_FIELDING_E")

hist(data$TEAM_FIELDING_DP, col = "#A71930", xlab = "TEAM_FIELDING_DP", main = "Histogram of Double Plays")
```

```
#boxplot(data$TEAM_FIELDING_DP, col = "#A71930", main = "Boxplot of TEAM_FIELDING_DP")


par(mfrow=c(2,2))

hist(data$TEAM_PITCHING_H, col = "#A71930", xlab = "TEAM_PITCHING_H", main = "Histogram of
Hits")

#boxplot(data$TEAM_PITCHING_H, col = "#A71930", main = "Boxplot of TEAM_PITCHING_H")

hist(data$TEAM_PITCHING_HR, col = "#A71930", xlab = "TEAM_PITCHING_HR", main = "Histogram of
Home Runs")

#boxplot(data$TEAM_PITCHING_HR, col = "#A71930", main = "Boxplot of TEAM_PITCHING_HR")

hist(data$TEAM_PITCHING_BB, col = "#A71930", xlab = "TEAM_PITCHING_BB", main = "Histogram of
Walks")

#boxplot(data$TEAM_PITCHING_BB, col = "#A71930", main = "Boxplot of TEAM_PITCHING_BB")

hist(data$TEAM_PITCHING_SO, col = "#A71930", xlab = "TEAM_PITCHING_SO", main = "Histogram of
Strike Outs")

#boxplot(data$TEAM_PITCHING_SO, col = "#A71930", main = "Boxplot of TEAM_PITCHING_SO")


data=data[(data$TARGET_WINS>quantile(data$TARGET_WINS,.005))&(data$TARGET_WINS<quantile(da
ta$TARGET_WINS,.9955)),]

data=data[(data$TEAM_BATTING_H>quantile(data$TEAM_BATTING_H,.005))&(data$TEAM_BATTING_H
<quantile(data$TEAM_BATTING_H,.995)),]

data=data[(data$TEAM_BATTING_2B>quantile(data$TEAM_BATTING_2B,.005))&(data$TEAM_BATTING
_2B<quantile(data$TEAM_BATTING_2B,.995)),]

data=data[(data$TEAM_BATTING_3B>quantile(data$TEAM_BATTING_3B,.005))&(data$TEAM_BATTING
_3B<quantile(data$TEAM_BATTING_3B,.995)),]

data=data[(data$TEAM_BATTING_HR>quantile(data$TEAM_BATTING_HR,.005))&(data$TEAM_BATTING
_HR<quantile(data$TEAM_BATTING_HR,.995)),]

data=data[(data$TEAM_BATTING_BB>quantile(data$TEAM_BATTING_BB,.005))&(data$TEAM_BATTING
_BB<quantile(data$TEAM_BATTING_BB,.995)),]

#data=data[(data$TEAM_BATTING_SO>quantile(data$TEAM_BATTING_SO,.005))&(data$TEAM_BATTIN
G_SO<quantile(data$TEAM_BATTING_SO,.995)),]

#data=data[(data$TEAM_BASERUN_SB>quantile(data$TEAM_BASERUN_SB,.005))&(data$TEAM_BASER
UN_SB<quantile(data$TEAM_BASERUN_SB,.995)),]
```

```
#data=data[(data$TEAM_BASERUN_CS>quantile(data$TEAM_BASERUN_CS,.005))&(data$TEAM_BASERUN_CS<quantile(data$TEAM_BASERUN_CS,.995)),]

#data=data[(data$TEAM_BATTING_HBP>quantile(data$TEAM_BATTING_HBP,.005))&(data$TEAM_BATTING_HBP<quantile(data$TEAM_BATTING_HBP,.995)),]

data=data[(data$TEAM_PITCHING_H>quantile(data$TEAM_PITCHING_H,.005))&(data$TEAM_PITCHING_H<quantile(data$TEAM_PITCHING_H,.995)),]

data=data[(data$TEAM_PITCHING_HR>quantile(data$TEAM_PITCHING_HR,.005))&(data$TEAM_PITCHING_HR<quantile(data$TEAM_PITCHING_HR,.995)),]

data=data[(data$TEAM_PITCHING_BB>quantile(data$TEAM_PITCHING_BB,.005))&(data$TEAM_PITCHING_BB<quantile(data$TEAM_PITCHING_BB,.995)),]

#data=data[(data$TEAM_PITCHING_SO>quantile(data$TEAM_PITCHING_SO,.005))&(data$TEAM_PITCHING_SO<quantile(data$TEAM_PITCHING_SO,.995)),]

data=data[(data$TEAM_FIELDING_E>quantile(data$TEAM_FIELDING_E,.005))&(data$TEAM_FIELDING_E<quantile(data$TEAM_FIELDING_E,.995)),]

#data=data[(data$TEAM_FIELDING_DP>quantile(data$TEAM_FIELDING_DP,.005))&(data$TEAM_FIELDING_DP<quantile(data$TEAM_FIELDING_DP,.995)),]


#removal based on actual baseball data via baseball almanac

print (summary(data))

data=data[(data$TARGET_WINS>=32)&(data$TARGET_WINS<120),]

print (summary(data))

data=data[(data$TEAM_BATTING_3B>=11),]

print(quantile(data$TEAM_BATTING_3B,.99))

print(data[data$TEAM_BATTING_3B>132,]$TEAM_BATTING_3B)

print (summary(data))

data=data[(data$TEAM_BATTING_HR>=3),]

print(quantile(data$TEAM_BATTING_HR,.99))

print(data[data$TEAM_BATTING_HR>235,]$TEAM_BATTING_HR)

print(quantile(data$TEAM_PITCHING_BB,.99))

print(data[data$TEAM_PITCHING_BB>885,]$TEAM_PITCHING_BB)

data=data[(data$TEAM_PITCHING_BB<2000),]

print (summary(data))
```

```
print(quantile(data$TEAM_PITCHING_H,.99))
print(data[data$TEAM_PITCHING_H>3149,]$TEAM_PITCHING_H)
data=data[(data$TEAM_PITCHING_BB<5000),]
print(quantile(data$TEAM_FIELDING_E,.99))
print(data[data$TEAM_FIELDING_E>1076,]$TEAM_FIELDING_E)

#rpart tree imputation for TEAM_BASERUN_SB
data$IMP_BR_SB=data$TEAM_BASERUN_SB
data$M_BR_SB=data$TEAM_BASERUN_SB
data$M_BR_SB[!is.na(data$M_BR_SB)]=1
data$M_BR_SB[is.na(data$M_BR_SB)]=0
drops <- c("TEAM_BASERUN_SB","INDEX")
sb=data[ , !(names(data) %in% drops)]

SBANOVA <- rpart(IMP_BR_SB ~ .,
            data=sb[!is.na(data$IMP_BR_SB),-2]
            , method="anova", na.action=na.omit,control = rpart.control(cp = 0.05))
SB_PRED <- predict(SBANOVA, data[is.na(data$IMP_BR_SB), ])

data$IMP_BR_SB[is.na(data$IMP_BR_SB)]=SB_PRED
summary(data)
par(mfrow = c(1,1), xpd = NA) # otherwise on some devices the text is clipped
rpart.plot(SBANOVA)

#rpart tree imputation for TEAM_BATTING_SO
data$IMP_BA_SO=data$TEAM_BATTING_SO
data$M_BA_SO=data$TEAM_BATTING_SO
data$M_BA_SO[!is.na(data$M_BA_SO)]=1
data$M_BA_SO[is.na(data$M_BA_SO)]=0
```

```
drops <- c("TEAM_BATTING_SO","INDEX")
bs=data[ , !(names(data) %in% drops)]

BA_SO_ANOVA <- rpart(IMP_BA_SO ~ .,
                     data=bs[!is.na(data$IMP_BA_SO),-2]
                     , method="anova", na.action=na.omit,control = rpart.control(cp = 0.05))
BA_SO_PRED <- predict(BA_SO_ANOVA, data[is.na(data$IMP_BA_SO), ])

data$IMP_BA_SO[is.na(data$IMP_BA_SO)]=BA_SO_PRED
summary(data)
par(mfrow = c(1,1), xpd = NA) # otherwise on some devices the text is clipped
rpart.plot(BA_SO_ANOVA,main="Batter Strikeout Tree")

#rpart tree imputation for TEAM_PITCHING_SO
data$IMP_PI_SO=data$TEAM_PITCHING_SO
data$M_PI_SO=data$TEAM_PITCHING_SO
data$M_PI_SO[!is.na(data$M_PI_SO)]=1
data$M_PI_SO[is.na(data$M_PI_SO)]=0

drops <- c("TEAM_PITCHING_SO","INDEX")
ps=data[ , !(names(data) %in% drops)]



PI_SO_ANOVA <- rpart(IMP_PI_SO ~ .,
                     data=ps[!is.na(data$IMP_PI_SO),-2]
                     , method="anova", na.action=na.omit,control = rpart.control(cp = 0.05))
PI_SO_PRED <- predict(PI_SO_ANOVA, data[is.na(data$IMP_PI_SO), ])

data$IMP_PI_SO[is.na(data$IMP_PI_SO)]=PI_SO_PRED
```

```
summary(data)
par(mfrow = c(1,1), xpd = NA) # otherwise on some devices the text is clipped
rpart.plot(PI_SO_ANOVA)



#rpart tree imputation for TEAM_FIELDING_DP
data$IMP_DP=data$TEAM_FIELDING_DP
data$M_DP=data$TEAM_FIELDING_DP
data$M_DP[!is.na(data$M_DP)]=1
data$M_DP[is.na(data$M_DP)]=0
drops <- c("TEAM_FIELDING_DP")
data=data[ , !(names(data) %in% drops)]
dp=data[,!(names(data) %in% "INDEX")]
DP_ANOVA <- rpart(IMP_DP ~ .,
                  data=dp[!is.na(dp$IMP_DP),-2]
                  , method="anova", na.action=na.omit,control = rpart.control(cp = 0.05))
DP_PRED <- predict(DP_ANOVA, dp[is.na(dp$IMP_DP), ])

data$IMP_DP[is.na(data$IMP_DP)]=DP_PRED
summary(data)
par(mfrow = c(1,1), xpd = NA) # otherwise on some devices the text is clipped
rpart.plot(DP_ANOVA)



#rpart tree imputation for TEAM_BASERUN_CS
data$IMP_CS=data$TEAM_BASERUN_CS
data$M_CS=data$TEAM_BASERUN_CS
data$M_CS[!is.na(data$M_CS)]=1
```

```
data$M_CS[is.na(data$M_CS)]=0
drops <- c("TEAM_BASERUN_CS","INDEX")
cs=data[ , !(names(data) %in% drops)]
CS_ANOVA <- rpart(IMP_CS ~ .,
        data=cs[!is.na(data$IMP_CS),]
        , method="anova", na.action=na.omit,control = rpart.control(cp = 0.05))
CS_PRED <- predict(CS_ANOVA, data[is.na(data$IMP_CS), ])

data$IMP_CS[is.na(data$IMP_CS)]=CS_PRED
summary(data)
par(mfrow = c(1,1), xpd = NA) # otherwise on some devices the text is clipped
rpart.plot(CS_ANOVA)


par(mfrow = c(1,1),xpd= NA) # otherwise on some devices the text is clipped
rpart.plot(BA_SO_ANOVA,main="Batter Strikeout Tree")
rpart.plot(PI_SO_ANOVA, main="Pitcher Strikeout Tree")
rpart.plot(DP_ANOVA, main="Double Play Tree")
rpart.plot(SBANOVA, main="Stolen Base Tree")
rpart.plot(CS_ANOVA, main="Caught Stealing Tree")

#fixing data from the imputed variables
data=data[(data$IMP_BA_SO>308),]
data=data[(data$IMP_PI_SO>308)&(data$IMP_PI_SO<2000),]
print (summary(data))
print(quantile(data$IMP_PI_SO,.99))
print(data[data$IMP_PI_SO>1430,]$IMP_PI_SO)

#Adding additional Stats
```

```
data$Slug=data$TEAM_BATTING_2B*2+data$TEAM_BATTING_3B*3+data$TEAM_BATTING_1B*1+data
$TEAM_BATTING_HR*4

data$OB=data$TEAM_BATTING_H+data$TEAM_BATTING_BB

data$WH=data$TEAM_PITCHING_BB+data$TEAM_PITCHING_H



par(mfrow=c(2,2))

hist(data$TEAM_PITCHING_H, col = "#A71930", xlab = "TEAM_PITCHING_H", main = "Histogram of
Hits")

#boxplot(data$TEAM_PITCHING_H, col = "#A71930", main = "Boxplot of TEAM_PITCHING_H")

hist(data$TEAM_PITCHING_HR, col = "#A71930", xlab = "TEAM_PITCHING_HR", main = "Histogram of
Home Runs")

#boxplot(data$TEAM_PITCHING_HR, col = "#A71930", main = "Boxplot of TEAM_PITCHING_HR")

hist(data$TEAM_PITCHING_BB, col = "#A71930", xlab = "TEAM_PITCHING_BB", main = "Histogram of
Walks")

#boxplot(data$TEAM_PITCHING_BB, col = "#A71930", main = "Boxplot of TEAM_PITCHING_BB")

hist(data$IMP_PI_SO, col = "#A71930", xlab = "TEAM_PITCHING_SO", main = "Histogram of Strike Outs")

#boxplot(data$TEAM_PITCHING_SO, col = "#A71930", main = "Boxplot of TEAM_PITCHING_SO")

par(mfrow=c(2,2))

hist(data$TEAM_BASERUN_SB, col = "#A71930", xlab = "TEAM_BASERUN_SB", main = "Histogram of
Stolen Bases")

#boxplot(data$TEAM_BASERUN_SB, col = "#A71930", main = "Boxplot of TEAM_BASERUN_SB")

hist(data$TEAM_BASERUN_CS, col = "#A71930", xlab = "TEAM_BASERUN_CS", main = "Histogram of
Caught Stealing")

#boxplot(data$TEAM_BASERUN_CS, col = "#A71930", main = "Boxplot of TEAM_BASERUN_CS")

hist(data$TEAM_FIELDING_E, col = "#A71930", xlab = "TEAM_FIELDING_E", main = "Histogram of
Errors")

#boxplot(data$TEAM_FIELDING_E, col = "#A71930", main = "Boxplot of TEAM_FIELDING_E")

hist(data$TEAM_FIELDING_DP, col = "#A71930", xlab = "TEAM_FIELDING_DP", main = "Histogram of
Double Plays")

#boxplot(data$TEAM_FIELDING_DP, col = "#A71930", main = "Boxplot of TEAM_FIELDING_DP")

#Adding transformation variables
```

```
data$log_1B=log10(data$TEAM_BATTING_1B)
data$log_3B=log10(data$TEAM_BATTING_3B)
data$sqrt_b_h=sqrt(data$TEAM_BATTING_H)
data$log_SB=log10(data$IMP_BR_SB)
data$sqrt_CS=sqrt(data$IMP_CS)
data$log_E=log10(data$TEAM_FIELDING_E)
data$log_PH=log10(data$TEAM_PITCHING_H)
data$sqrt_PBB=sqrt(data$TEAM_PITCHING_BB)
data$log_WH=log10(data$WH)

par(mfrow=c(2,2))
#transformation for
hist(data$log_1B, col = "#A71930", main="Histogram of Log Singles")
hist(data$log_3B, col = "#A71930", main="Histogram of Log Triples")
hist(data$sqrt_b_h, col = "#A71930", main="Histogram of Square Root Hits")
hist(data$log_SB, col = "#A71930", main="Histogram of Log Stolen Bases")

hist(data$sqrt_CS, col = "#A71930", main="Histogram of Square Root Caught Stealing")
hist(data$log_E, col = "#A71930",main="Histogram of Log Errors")
hist(data$log_PH, col = "#A71930",main="Histogram of Log Hits Given Up")
hist(data$sqrt_PBB, col = "#A71930",main="Histogram of Square Root of Pitcher Walks")

hist(data$Slug, col = "#A71930", main="Histogram of Slugging")
hist(data$OB, col = "#A71930",main="Histogram of On Base")
hist(data$WH, col = "#A71930",main="Histogram of Walk and Hits")
hist(data$log_WH, col = "#A71930",main="Histogram of Log Walk and Hits")
start=data.frame()
for (x in 1:25)
{
```

```
smp=floor(.75*nrow(data))

set.seed(x)

train_ind <- sample(seq_len(nrow(data)), size = smp)

train=data[train_ind,]

test=data[-train_ind,]


drops <-
c("TEAM_BASERUN_SB","TEAM_BASERUN_CS","TEAM_BATTING_SO","TEAM_PITCHING_SO","TEAM_B
ATTING_HBP","TEAM_FIELDING_DP")

train=train[ , !(names(train) %in% drops)]


summary(train)

stepwisemodel <- lm(formula = TARGET_WINS ~ ., data = train)

summary(stepwisemodel)

stepwise <- stepAIC(stepwisemodel, direction = "both")

summary(stepwise)


test$predicted=predict(stepwise,test)

mean((test$TARGET_WINS-test$predicted)**2)

compare=data.frame()

compare=as.data.frame(test$TARGET_WINS)

compare$Predict=test$predicted


mse(stepwise)

AIC(stepwise)

scores=c(mean((test$TARGET_WINS-test$predicted)**2),mse(stepwise),AIC(stepwise))

start=rbind(start,scores)

}

colnames(start)=c("Test MSE","Train MSE","Train AIC")
```

```
print(summary(start))


print(summary(start))
print(summary(test))
print(summary(stepwise))
print(mean(test$predicted))

TWANOVA <- rpart(TARGET_WINS ~ .,  #testing tree regression
                 data=train
                 , method="anova")
TWANOVA <- prune(TWANOVA, cp = 0.01)
test$tPredict <- predict(TWANOVA, test)
mean(test$tPredict)
mean((test$TARGET_WINS-test$tPredict)**2)




outcome=read.csv("C:/Users/NAME/Desktop/School/411/Module 1/moneyball_test.csv",header=T)
Max_h=max(data$TEAM_BATTING_H)
Min_h=min(data$TEAM_BATTING_H)
summary(data)


#rpart tree imputation for TEAM_BASERUN_SB
outcome$IMP_BR_SB=outcome$TEAM_BASERUN_SB
outcome$M_BR_SB=outcome$TEAM_BASERUN_SB
outcome$M_BR_SB[!is.na(outcome$M_BR_SB)]=1
outcome$M_BR_SB[is.na(outcome$M_BR_SB)]=0
SB_PRED <- predict(SBANOVA, outcome[is.na(outcome$IMP_BR_SB), ])
```

```
outcome$IMP_BR_SB[is.na(outcome$IMP_BR_SB)]=SB_PRED
summary(outcome)
par(mfrow = c(1,1), xpd = NA) # otherwise on some devices the text is clipped
rpart.plot(SBANOVA)

#rpart tree imputation for TEAM_BATTING_SO
outcome$IMP_BA_SO=outcome$TEAM_BATTING_SO
outcome$M_BA_SO=outcome$TEAM_BATTING_SO
outcome$M_BA_SO[!is.na(outcome$M_BA_SO)]=1
outcome$M_BA_SO[is.na(outcome$M_BA_SO)]=0

BA_SO_PRED <- predict(BA_SO_ANOVA, outcome[is.na(outcome$IMP_BA_SO), ])

outcome$IMP_BA_SO[is.na(outcome$IMP_BA_SO)]=BA_SO_PRED
summary(outcome)
par(mfrow = c(1,1), xpd = NA) # otherwise on some devices the text is clipped
rpart.plot(BA_SO_ANOVA,main="Batter Strikeout Tree")

#rpart tree imputation for TEAM_PITCHING_SO
outcome$IMP_PI_SO=outcome$TEAM_PITCHING_SO
outcome$M_PI_SO=outcome$TEAM_PITCHING_SO
outcome$M_PI_SO[!is.na(outcome$M_PI_SO)]=1
outcome$M_PI_SO[is.na(outcome$M_PI_SO)]=0

PI_SO_PRED <- predict(PI_SO_ANOVA, outcome[is.na(outcome$IMP_PI_SO), ])

outcome$IMP_PI_SO[is.na(outcome$IMP_PI_SO)]=PI_SO_PRED
summary(outcome)
```

```
par(mfrow = c(1,1), xpd = NA) # otherwise on some devices the text is clipped
rpart.plot(PI_SO_ANOVA)


#rpart tree imputation for TEAM_FIELDING_DP
outcome$IMP_DP=outcome$TEAM_FIELDING_DP
outcome$M_DP=outcome$TEAM_FIELDING_DP
outcome$M_DP[!is.na(outcome$M_DP)]=1
outcome$M_DP[is.na(outcome$M_DP)]=0

DP_PRED <- predict(DP_ANOVA, dp[is.na(dp$IMP_DP), ])

outcome$IMP_DP[is.na(outcome$IMP_DP)]=DP_PRED
summary(outcome)
par(mfrow = c(1,1), xpd = NA) # otherwise on some devices the text is clipped
rpart.plot(DP_ANOVA)



#rpart tree imputation for TEAM_BASERUN_CS
outcome$IMP_CS=outcome$TEAM_BASERUN_CS
outcome$M_CS=outcome$TEAM_BASERUN_CS
outcome$M_CS[!is.na(outcome$M_CS)]=1
outcome$M_CS[is.na(outcome$M_CS)]=0

CS_PRED <- predict(CS_ANOVA, outcome[is.na(outcome$IMP_CS), ])
summary(outcome)
summary(data)
```

```
outcome$TEAM_BATTING_H[(outcome$TEAM_BATTING_H>max(data$TEAM_BATTING_H))]=max(data$TEAM_BATTING_H)

outcome$TEAM_BATTING_H[(outcome$TEAM_BATTING_H<min(data$TEAM_BATTING_H))]=min(data$TEAM_BATTING_H)

outcome$TEAM_BATTING_2B[(outcome$TEAM_BATTING_2B>max(data$TEAM_BATTING_2B))]=max(data$TEAM_BATTING_2B)

outcome$TEAM_BATTING_2B[(outcome$TEAM_BATTING_2B<min(data$TEAM_BATTING_2B))]=min(data$TEAM_BATTING_2B)

outcome$TEAM_BATTING_3B[(outcome$TEAM_BATTING_3B>max(data$TEAM_BATTING_3B))]=max(data$TEAM_BATTING_3B)

outcome$TEAM_BATTING_3B[(outcome$TEAM_BATTING_3B<min(data$TEAM_BATTING_3B))]=min(data$TEAM_BATTING_3B)

outcome$TEAM_BATTING_HR[(outcome$TEAM_BATTING_HR>max(data$TEAM_BATTING_HR))]=max(data$TEAM_BATTING_HR)

outcome$TEAM_BATTING_HR[(outcome$TEAM_BATTING_HR<min(data$TEAM_BATTING_HR))]=min(data$TEAM_BATTING_HR)

outcome$TEAM_BATTING_BB[(outcome$TEAM_BATTING_BB>max(data$TEAM_BATTING_BB))]=max(data$TEAM_BATTING_BB)

outcome$TEAM_BATTING_BB[(outcome$TEAM_BATTING_BB<min(data$TEAM_BATTING_BB))]=min(data$TEAM_BATTING_BB)

outcome$TEAM_PITCHING_H[(outcome$TEAM_PITCHING_H>max(data$TEAM_PITCHING_H))]=max(data$TEAM_PITCHING_H)

outcome$TEAM_PITCHING_H[(outcome$TEAM_PITCHING_H<min(data$TEAM_PITCHING_H))]=min(data$TEAM_PITCHING_H)

outcome$TEAM_PITCHING_HR[(outcome$TEAM_PITCHING_HR>max(data$TEAM_PITCHING_HR))]=max(data$TEAM_PITCHING_HR)

outcome$TEAM_PITCHING_HR[(outcome$TEAM_PITCHING_HR<min(data$TEAM_PITCHING_HR))]=min(data$TEAM_PITCHING_HR)

outcome$TEAM_FIELDING_E[(outcome$TEAM_FIELDING_E>max(data$TEAM_FIELDING_E))]=max(data$TEAM_FIELDING_E)

outcome$TEAM_FIELDING_E[(outcome$TEAM_FIELDING_E<min(data$TEAM_FIELDING_E))]=min(data$TEAM_FIELDING_E)

outcome$IMP_BR_SB[(outcome$IMP_BR_SB>max(data$IMP_BR_SB))]=max(data$IMP_BR_SB)

outcome$IMP_BR_SB[(outcome$IMP_BR_SB<min(data$IMP_BR_SB))]=min(data$IMP_BR_SB)
```

```
outcome$IMP_BA_SO[(outcome$IMP_BA_SO>max(data$IMP_BA_SO))]=max(data$IMP_BA_SO)

outcome$IMP_BA_SO[(outcome$IMP_BA_SO<min(data$IMP_BA_SO))]=min(data$IMP_BA_SO)

outcome$IMP_PI_SO[(outcome$IMP_PI_SO>max(data$IMP_PI_SO))]=max(data$IMP_PI_SO)

outcome$IMP_PI_SO[(outcome$IMP_PI_SO<min(data$IMP_PI_SO))]=min(data$IMP_PI_SO)

outcome$IMP_DP[(outcome$IMP_DP>max(data$IMP_DP))]=max(data$IMP_DP)

outcome$IMP_DP[(outcome$IMP_DP<min(data$IMP_DP))]=min(data$IMP_DP)

outcome$IMP_CS[(outcome$IMP_CS>max(data$IMP_CS))]=max(data$IMP_CS)

outcome$IMP_CS[(outcome$IMP_CS<min(data$IMP_CS))]=min(data$IMP_CS)
```

```
outcome$TEAM_BATTING_1B=outcome$TEAM_BATTING_H-outcome$TEAM_BATTING_2B-
outcome$TEAM_BATTING_3B-outcome$TEAM_BATTING_HR

outcome$Slug=outcome$TEAM_BATTING_2B*2+outcome$TEAM_BATTING_3B*3+outcome$TEAM_BAT
TING_1B*1+outcome$TEAM_BATTING_HR*4

outcome$OB=outcome$TEAM_BATTING_H+outcome$TEAM_BATTING_BB

outcome$WH=outcome$TEAM_PITCHING_BB+outcome$TEAM_PITCHING_H

outcome$log_1B=log10(outcome$TEAM_BATTING_1B)

outcome$log_3B=log10(outcome$TEAM_BATTING_3B)

outcome$sqrt_b_h=sqrt(outcome$TEAM_BATTING_H)

outcome$log_SB=log10(outcome$IMP_BR_SB)

outcome$sqrt_CS=sqrt(outcome$IMP_CS)

outcome$log_E=log10(outcome$TEAM_FIELDING_E)

outcome$log_PH=log10(outcome$TEAM_PITCHING_H)

outcome$sqrt_PBB=sqrt(outcome$TEAM_PITCHING_BB)

outcome$log_WH=log10(outcome$WH)
```

```
outcome$log_1B[(outcome$log_1B>max(data$log_1B))]=max(data$log_1B)

outcome$log_1B[(outcome$log_1B<min(data$log_1B))]=min(data$log_1B)

outcome$log_E[(outcome$log_E>max(data$log_E))]=max(data$log_E)

outcome$log_E[(outcome$log_E<min(data$log_E))]=min(data$log_E)

outcome$sqrt_CS[(outcome$sqrt_CS>max(data$sqrt_CS))]=max(data$sqrt_CS)

outcome$sqrt_CS[(outcome$sqrt_CS<min(data$sqrt_CS))]=min(data$sqrt_CS)

outcome$sqrt_PBB[(outcome$sqrt_PBB>max(data$sqrt_PBB))]=max(data$sqrt_PBB)

outcome$sqrt_PBB[(outcome$sqrt_PBB<min(data$sqrt_PBB))]=min(data$sqrt_PBB)


outcome$predicted=round(predict(stepwise,outcome),0)

summary(outcome$predicted)

submission=data.frame

submission=outcome$INDEX

submission=as.data.frame(submission)

submission$TARGET_WINS=outcome$predicted

colnames(submission)=c("INDEX","TARGET_WINS")

submission

write.csv(submission,"C:/Users/NAME/Desktop/School/411/Module 1/
Moneyball_Scored.csv",row.names=FALSE)
```