CUNY DATA 621: HW #1

Andrew Carson

February 6, 2018

Overview

In this homework assignment, you will explore, analyze and model a data set containing approximately 2200 records. Each record represents a professional baseball team from the years 1871 to 2006 inclusive. Each record has the performance of the team for the given year, with all of the statistics adjusted to match the performance of a 162 game season.

Your objective is to build a multiple linear regression model on the training data to predict the number of wins for the team. You can only use the variables given to you (or variables that you derive from the variables provided).

Deliverables

- A write-up submitted in PDF format. Your write-up should have four sections. Each one is described below. You may assume you are addressing me as a fellow data scientist, so do not need to shy away from technical details.
- Assigned predictions (the number of wins for the team) for the evaluation data set.
- Include your R statistical programming code in an Appendix.

1. Data Exploration

Describe the size and the variables in the moneyball training data set. Consider that too much detail will cause a manager to lose interest while too little detail will make the manager consider that you aren't doing your job. Some suggestions are given below. Please do NOT treat this as a check list of things to do to complete the assignment. You should have your own thoughts on what to tell the boss. These are just ideas.

a. Mean / Standard Deviation / Median
b. Bar Chart or Box Plot of the data
c. Is the data correlated to the target variable (or to other variables?)
d. Are any of the variables missing and need to be imputed "fixed"?

Answer:

There are 2276 observations and 17 variables in the data set. The data can roughly be described as containing the number of wins, hitting numbers (e.g., hits, walks), fielding numbers(e.g., errors), and pitching numbers (e.g., strikeouts by pitchers). While some numbers are considered to impact wins positively (e.g., home runs), others are considered to impact wins negatively (e.g., strikeouts by batters)

Below is a summary of the data (quartiles, median, mean). Some observations:

- There are some "missing" variables that could be of interest that are derivable from the data. Examples include team batting singles and on base percentage.
- There are particular cases where there are obvious outliers, even just from looking at the summary (e.g., TEAM_BATTING_3B has a max of 223 when the median is 47; TEAM_PITCHING_SO has a max of 19,278, which is not possible, as a pitcher would face at most about 5,000 batters in a single season).

- There are lots of missing values (NA's), so those will have to be treated in some way. TEAM_BATTING_HBP is particularly bad (2085 missing). More on that later.
- The mean and the median are typically close, so that is important for having distributions that are relatively normal. Where this is not the case are in TEAM_BASERUN_SB, TEAM_PITCHING_H, and TEAM_FIELDING_E. A closer visual look will be able to give us more information about the distribution.

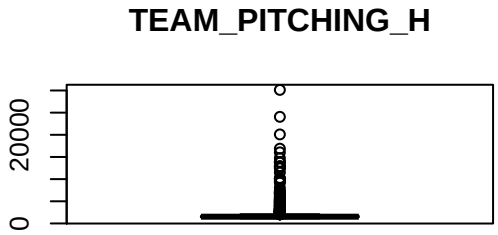
```
## [1] "Summary of train"
```

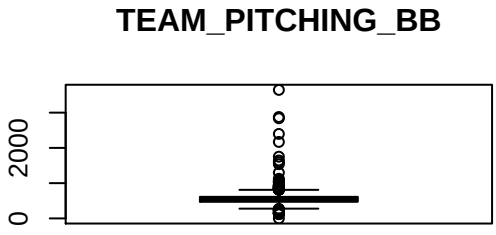
```
##   TARGET_WINS     TEAM_BATTING_H TEAM_BATTING_2B TEAM_BATTING_3B
##  Min.   :  0.00   Min.   : 891   Min.   : 69.0   Min.   :  0.00
##  1st Qu.: 71.00   1st Qu.:1383   1st Qu.:208.0   1st Qu.: 34.00
##  Median : 82.00   Median :1454   Median :238.0   Median : 47.00
##  Mean   : 80.79   Mean   :1469   Mean   :241.2   Mean   : 55.25
##  3rd Qu.: 92.00   3rd Qu.:1537   3rd Qu.:273.0   3rd Qu.: 72.00
##  Max.   :146.00   Max.   :2554   Max.   :458.0   Max.   :223.00
##
##  TEAM_BATTING_HR  TEAM_BATTING_BB TEAM_BATTING_SO  TEAM_BASERUN_SB
##  Min.   :  0.00   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
##  Median :102.00   Median :512.0   Median : 750.0   Median :101.0
##  Mean   : 99.61   Mean   :501.6   Mean   : 735.6   Mean   :124.8
##  3rd Qu.:147.00   3rd Qu.:580.0   3rd Qu.: 930.0   3rd Qu.:156.0
##  Max.   :264.00   Max.   :878.0   Max.   :1399.0   Max.   :697.0
##                                   NA's   :102      NA's   :131
##  TEAM_BASERUN_CS TEAM_BATTING_HBP TEAM_PITCHING_H TEAM_PITCHING_HR
##  Min.   :  0.0   Min.   :29.00    Min.   : 1137   Min.   :  0.0
##  1st Qu.: 38.0   1st Qu.:50.50    1st Qu.: 1419   1st Qu.: 50.0
##  Median : 49.0   Median :58.00    Median : 1518   Median :107.0
##  Mean   : 52.8   Mean   :59.36    Mean   : 1779   Mean   :105.7
##  3rd Qu.: 62.0   3rd Qu.:67.00    3rd Qu.: 1682   3rd Qu.:150.0
##  Max.   :201.0   Max.   :95.00    Max.   :30132   Max.   :343.0
##  NA's   :772     NA's   :2085
##  TEAM_PITCHING_BB TEAM_PITCHING_SO  TEAM_FIELDING_E  TEAM_FIELDING_DP
##  Min.   :   0.0   Min.   :    0.0   Min.   :  65.0   Min.   : 52.0
##  1st Qu.: 476.0   1st Qu.:  615.0   1st Qu.: 127.0   1st Qu.:131.0
##  Median : 536.5   Median :  813.5   Median : 159.0   Median :149.0
##  Mean   : 553.0   Mean   :  817.7   Mean   : 246.5   Mean   :146.4
##  3rd Qu.: 611.0   3rd Qu.:  968.0   3rd Qu.: 249.2   3rd Qu.:164.0
##  Max.   :3645.0   Max.   :19278.0   Max.   :1898.0   Max.   :228.0
##                   NA's   :102                        NA's   :286
```

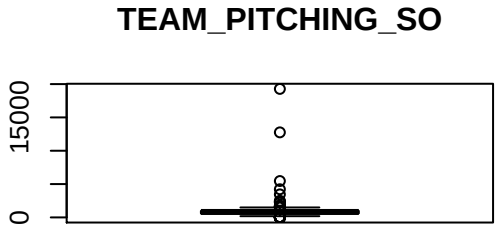
Looking at some charts to see if that can help us make any observations about the distributions of the variables, we see that due to the lower bound of 0, the distributions tend to skew right and have outliers above the mean/median, although this is not always the case. Particular variables of concern are:

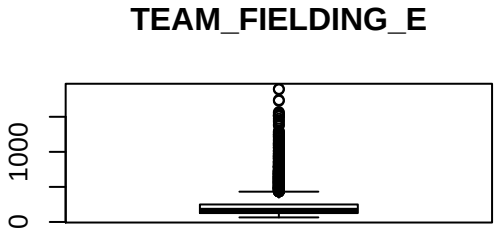
- TEAM_BASERUN_SB
- TEAM_BASERUN_CS
- TEAM_PITCHING_H
- TEAM_PITCHING_BB
- TEAM_PITCHING_SO
- TEAM_FIELDING_E
- TEAM_BATTING_3B
- Team_BATTING_HR: this looks like two combined distributions. Was there a rule change that make home runs more likely afterwards?
- TEAM_BATTING_SO: this also looks like two combined distributions...

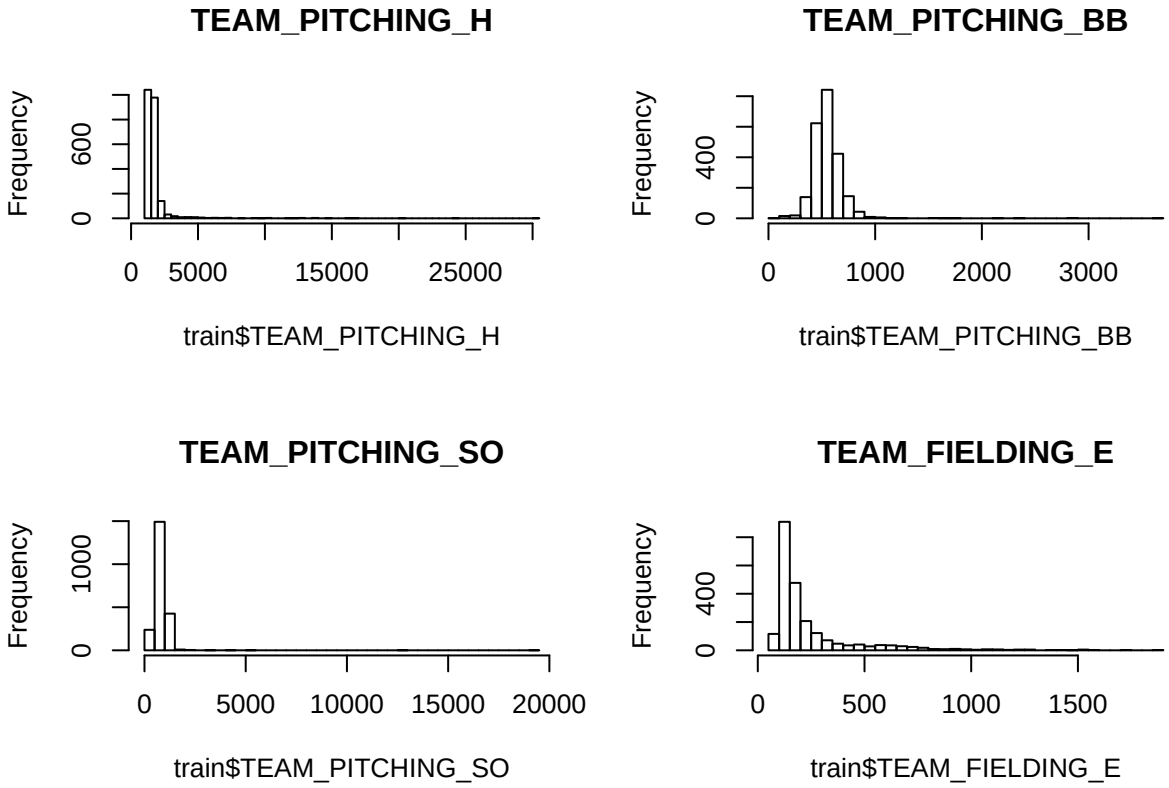
These will need to be transformed into more normal looking distributions (e.g., box-cox transformation). I include charts of the most skewed variables. It is interesting that these are the pitching variables. The batting variables are much more normally distributed.











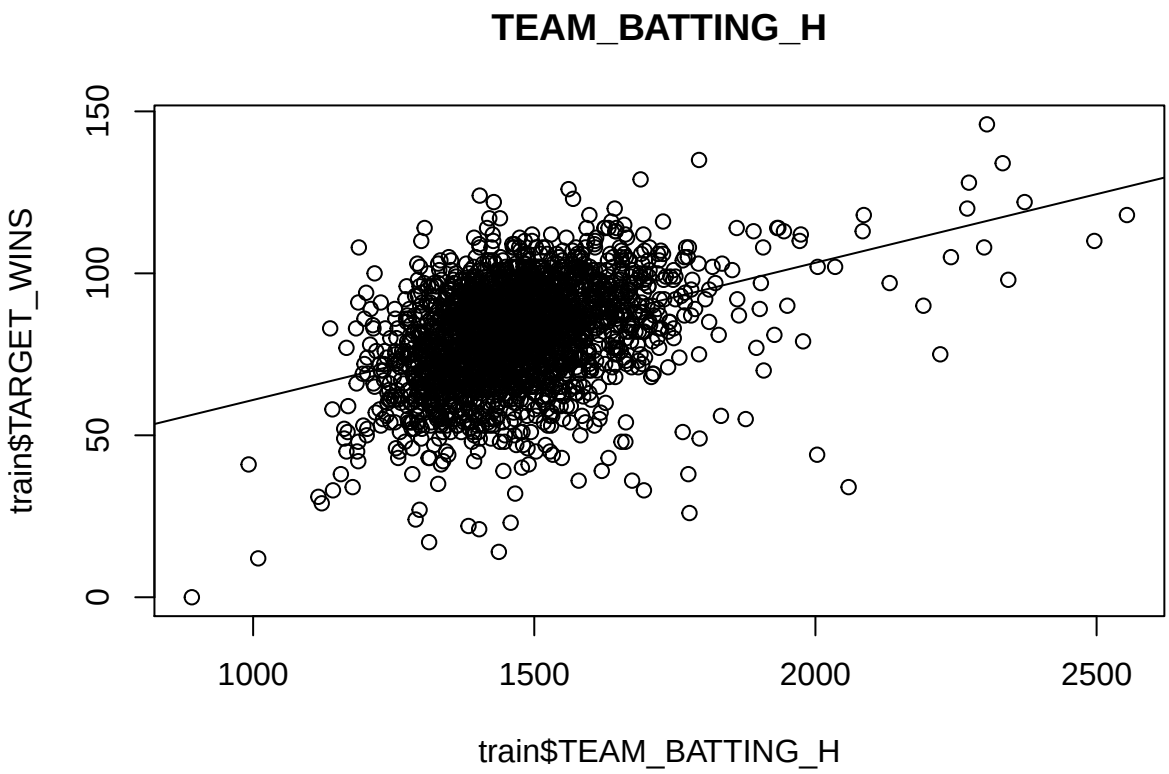
Ultimately, we are trying to predict "TARGET_WINS", that is, the number of wins in a particular season. Do any of the variables correlate well with this? Do they correlate with each other? First, focusing on which variables correlate with TARGET_WINS, we see the following top 5 variables with high correlation:

- TEAM_BATTING_H: 0.38. Positively correlated.
- TEAM_BATTING_2B: 0.28. Positively correlated.
- TEAM_BATTING_BB: 0.23. Positively correlated.
- TEAM_PITCHING_HR: 0.18. Positively correlated, although we would expect this to be negatively correlated. This will require more exploration.
- TEAM_FIELDING_E: -0.17. Negatively correlated.

In short, it appears that more getting on base (hits, walks) means more wins, while more errors and more hits allowed means fewer wins. Unfortunately, the relationship isn't very clear. For example, TEAM_BATTING_H, which has the highest correlation, has a pretty messy scatterplot.

```
## [1] "Correlation of Target Wins vs. Variables"
```

```
##      TARGET_WINS   TEAM_BATTING_H  TEAM_BATTING_2B  TEAM_BATTING_BB
##       1.00000000       0.38876752       0.28910365       0.23255986
## TEAM_PITCHING_HR  TEAM_BATTING_HR  TEAM_BATTING_3B  TEAM_BASERUN_SB
##       0.18901373       0.17615320       0.14260841       0.13513892
## TEAM_PITCHING_BB TEAM_BATTING_HBP  TEAM_BASERUN_CS  TEAM_BATTING_SO
##       0.12417454       0.07350424       0.02240407      -0.03175071
## TEAM_FIELDING_DP TEAM_PITCHING_SO  TEAM_PITCHING_H  TEAM_FIELDING_E
##      -0.03485058      -0.07843609      -0.10993705      -0.17648476
```



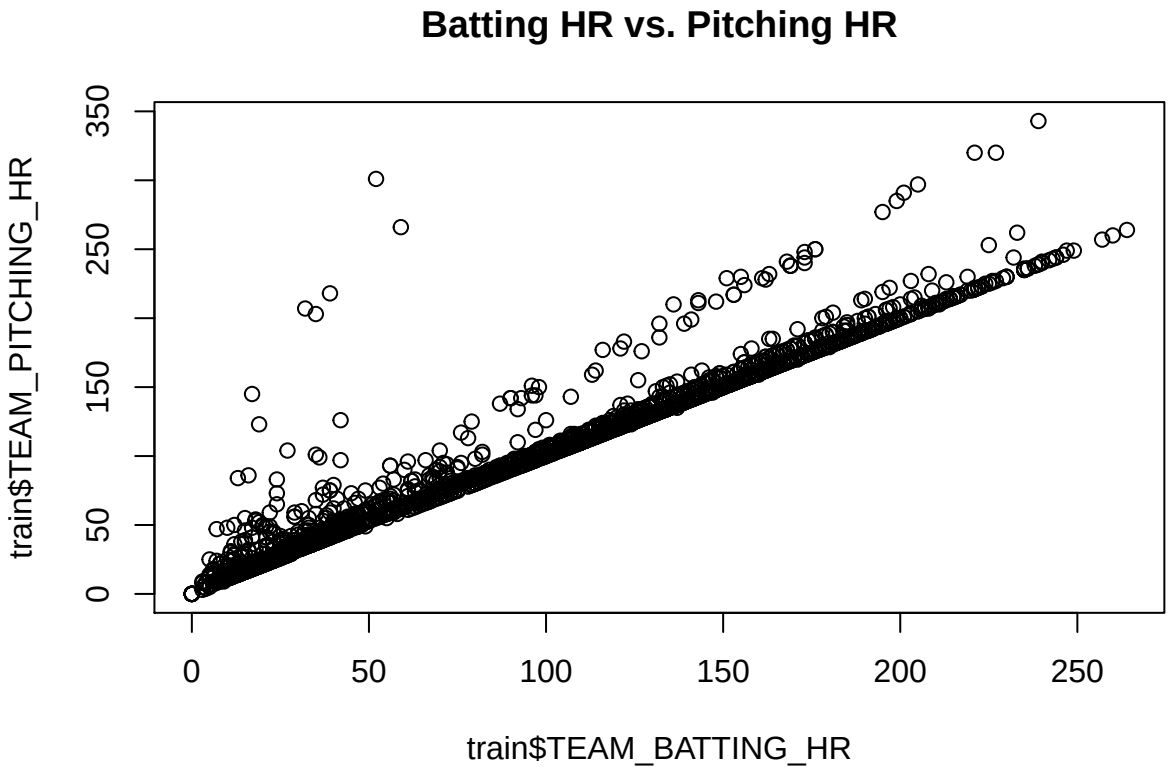
Are any variables highly correlated with each other, meaning roughly that they contain the same information? Defining highly correlated with each other to mean that a variable has a correlation of > 0.5 or < -0.5 with another variable, we see that:

- TEAM_BATTING_H highly correlates with TEAM_BATTING_2B (0.56). This relationship is obvious since 2B is a subset of H.
- TEAM_BATTING_3B highly correlates with TEAM_BATTING_HR (-0.63), TEAM_BATTING_SO (-0.66), TEAM_BASERUN_SB (0.53), TEAM_PITCHING_HR (-0.56), TEAM_FIELDING_E (0.50). I would think that better 3B would mean better HR, but it is the opposite, so that is strange. Better batting means fewer strikeouts and a better team (so stolen bases, fewer pitching HRs), but maybe it also means choosing batting and pitchers over field performance (hence more errors).
- TEAM_BATTING_HR highly correlates with TEAM_BATTING_BB (0.51), TEAM_BATTING_SO (0.72), TEAM_PITCHING_HR (0.96), TEAM_FIELDING_E (-0.58). It makes sense that a person that hits more home runs would be walked more often, hence, the positive correlation between HR and BB. But home run hitters are also known to strike out more, hence the positive correlation with SO. I do not know why batting HR should correlate with pitching HR, and so highly as that. Something is not right here. More batting home runs could mean a better team and so fewer errors.
- TEAM_BATTING_BB highly correlates with TEAM_FIELDING_E (-0.65). More walks means fewer errors? Not sure why this would be true.
- TEAM_BATTING_SO highly correlates with TEAM_PITCHING_HR (0.66), TEAM_FIELDING_E (-0.58). More strikeouts means a worse team which means worse pitching? Perhaps, but why would errors be negatively correlated, meaning that more batting strikeouts is related to fewer errors?
- TEAM_BASERUN_SB highly correlates with TEAM_BASERUN_CS (0.65), TEAM_FIELDING_E (0.50). More stolen bases is related to more times caught stealing, which makes sense. Not sure why more stolen bases would relate to more field errors.

- TEAM_PITCHING_H highly correlates with TEAM_FIELDING_E (0.66). This makes sense, since

if a pitcher allows more hits, then there are more chances for fielding errors.

There are some obvious explanations for the correlations here, but others are mysterious, at least to me, and these might just be noise. The most highly suspect is the relationship between TEAM_BATTING_HR and TEAM_PITCHING_HR. Of the 2276 observations, 35% are exactly the same number in both variables, 48% have a difference of 1 or less and 55% have a difference of 2 or less. I am inclined to believe that this is a data error and would consequently not use TEAM_PITCHING_HR.



2. Data Preparation

Describe how you have transformed the data by changing the original variables or creating new variables. If you did transform the data or create new variables, discuss why you did this. Here are some possible transformations.

a. Fix missing values (maybe with a Mean or Median value)
b. Create flags to suggest if a variable was missing
c. Transform data by putting it into buckets
d. Mathematical transforms such as log or square root (or use Box-Cox)
e. Combine variables (such as ratios or adding or multiplying) to create new variables
f. Additional: fix outliers

Answer:

First, before filling in the missing values, let's create flag variables for each variable that has missing values so as to mark the presence of a missing value in the original data. I also create a quick linear model to establish

a baseline. While it has an R^2 of 0.5116, I notice that most of the rows (2085) have been ignored due to incompleteness, so this isn't a really good starting point. Let's fill in the missing values and check again.

```
##
## Call:
## lm(formula = train$TARGET_WINS ~ ., data = train)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -19.8708  -5.6564  -0.0599   5.2545  22.9274
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)      60.28826   19.67842   3.064  0.00253 **
## TEAM_BATTING_H    1.91348    2.76139   0.693  0.48927
## TEAM_BATTING_2B   0.02639    0.03029   0.871  0.38484
## TEAM_BATTING_3B  -0.10118    0.07751  -1.305  0.19348
## TEAM_BATTING_HR  -4.84371   10.50851  -0.461  0.64542
## TEAM_BATTING_BB  -4.45969    3.63624  -1.226  0.22167
## TEAM_BATTING_SO   0.34196    2.59876   0.132  0.89546
## TEAM_BASERUN_SB   0.03304    0.02867   1.152  0.25071
## TEAM_BASERUN_CS  -0.01104    0.07143  -0.155  0.87730
## TEAM_BATTING_HBP  0.08247    0.04960   1.663  0.09815 .
## TEAM_PITCHING_H  -1.89096    2.76095  -0.685  0.49432
## TEAM_PITCHING_HR  4.93043   10.50664   0.469  0.63946
## TEAM_PITCHING_BB  4.51089    3.63372   1.241  0.21612
## TEAM_PITCHING_SO -0.37364    2.59705  -0.144  0.88577
## TEAM_FIELDING_E  -0.17204    0.04140  -4.155 5.08e-05 ***
## TEAM_FIELDING_DP -0.10819    0.03654  -2.961  0.00349 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 8.467 on 175 degrees of freedom
##   (2085 observations deleted due to missingness)
## Multiple R-squared:  0.5501, Adjusted R-squared:  0.5116
## F-statistic: 14.27 on 15 and 175 DF,  p-value: < 2.2e-16
```

Second, lets fill in the missing values.

- TEAM_BATTING_SO: there are 102 missing values. The mean (735) and median (750) are pretty close. This looks like two normal distributions merged together, and maybe there was a rule change the impacted strike outs. On a hunch that the missing values in TEAM_BASERUN_CS indicated earlier years, I split TEAM_BATTING_SO into two groups: those with TEAM_BASERUN_CS missing and those with it present. Having done so, the two normal distributions separated. When TERAM_BASERUN_CS is missing, the mean is 517 and the median is 533. When it is not missing, the mean is 832 and the median is 868. So these are very different. Consequently, I used the appropriate median to fill in the missing value based on the status of TEAM_BASERUN_CS. I also observed that there were several values of 0, which didn't make sense. I treated these as though they were missing values.
- TEAM_BASERUN_SB: with 131 missing values, the mean (124) and median (101) are a bit different given the right skew. Accordingly, I will use the median.
- TEAM_BASERUN_CS: there are 772 missing values, probably because this statistic was not recorded in earlier years. There is a little bit of a right-skew, but not too bad, and the mean (52.8) is pretty close to the median (49). Using the median.
- TEAM_BATTING_HBP: there are 2085 missing values, meaning there are only 191 actual values.

With so many missing values, I don't think we should even use this variable. There is only a 0.073 correlation with TARGET_WINS, so it doesn't appear to be worth it.

- TEAM_PITCHING_SO: there are 102 missing values. The mean (817) and median (813) are fairly close. Using the median.
- TEAM_FIELDING_DP: there are 286 missing values. The mean (146) and the median (149) are really close. Using the median.

After filling in all of the missing values, we have a baseline model with an R^2 of 0.4075. So I will take this as my starting place.

```
##
## Call:
## lm(formula = train$TARGET_WINS ~ ., data = train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -61.154  -8.204   0.271   7.976  49.558
##
## Coefficients: (1 not defined because of singularities)
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)         14.0687518  5.7050752   2.466 0.013737 *
## TEAM_BATTING_H       0.0487845  0.0034904  13.977  < 2e-16 ***
## TEAM_BATTING_2B     -0.0287120  0.0090409  -3.176 0.001515 **
## TEAM_BATTING_3B      0.0548046  0.0159384   3.439 0.000596 ***
## TEAM_BATTING_HR      0.0730361  0.0260758   2.801 0.005139 **
## TEAM_BATTING_BB      0.0246362  0.0055737   4.420 1.03e-05 ***
## TEAM_BATTING_SO     -0.0068916  0.0025241  -2.730 0.006376 **
## TEAM_BASERUN_SB      0.0510773  0.0049747  10.267  < 2e-16 ***
## TEAM_BASERUN_CS     -0.0257371  0.0163156  -1.577 0.114832
## TEAM_PITCHING_H      0.0022346  0.0003824   5.843 5.88e-09 ***
## TEAM_PITCHING_HR    -0.0068167  0.0227636  -0.299 0.764619
## TEAM_PITCHING_BB    -0.0005298  0.0039127  -0.135 0.892309
## TEAM_PITCHING_SO    -0.0015734  0.0008519  -1.847 0.064890 .
## TEAM_FIELDING_E     -0.0569276  0.0033996 -16.745  < 2e-16 ***
## TEAM_FIELDING_DP    -0.1000689  0.0136797  -7.315 3.56e-13 ***
## TEAM_BATTING_SO_NA   6.8641779  1.4862084   4.619 4.08e-06 ***
## TEAM_BASERUN_SB_NA  35.2259173  1.8923348  18.615  < 2e-16 ***
## TEAM_BASERUN_CS_NA   0.0005119  0.9340836   0.001 0.999563
## TEAM_BATTING_HBP_NA  3.8292052  1.1083459   3.455 0.000561 ***
## TEAM_PITCHING_SO_NA         NA         NA      NA       NA
## TEAM_FIELDING_DP_NA  4.5496022  1.5109604   3.011 0.002632 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 12.12 on 2256 degrees of freedom
## Multiple R-squared:  0.4125, Adjusted R-squared:  0.4075
## F-statistic: 83.35 on 19 and 2256 DF,  p-value: < 2.2e-16
```

Is there a need to transform data by putting it into buckets? For those variables that have extreme outliers that appear to fall outside the general patterns, I bucketed these into new categorical variables. In particular, I have bucketed:

- TEAM_PITCHING_H: with a mean of 1779 and a sd of 1406, a standard outlier of the mean + 3 * sd is about 6,000 and the 99th percentile is about 7000. Looking at the sorted data, there are 21 values above 8,000, with the highest being 30,132. I will create a bucket variable using quantiles.
- TEAM_PITCHING_BB: with a mean of 553 and a sd of 166, a traditional outlier is anything above

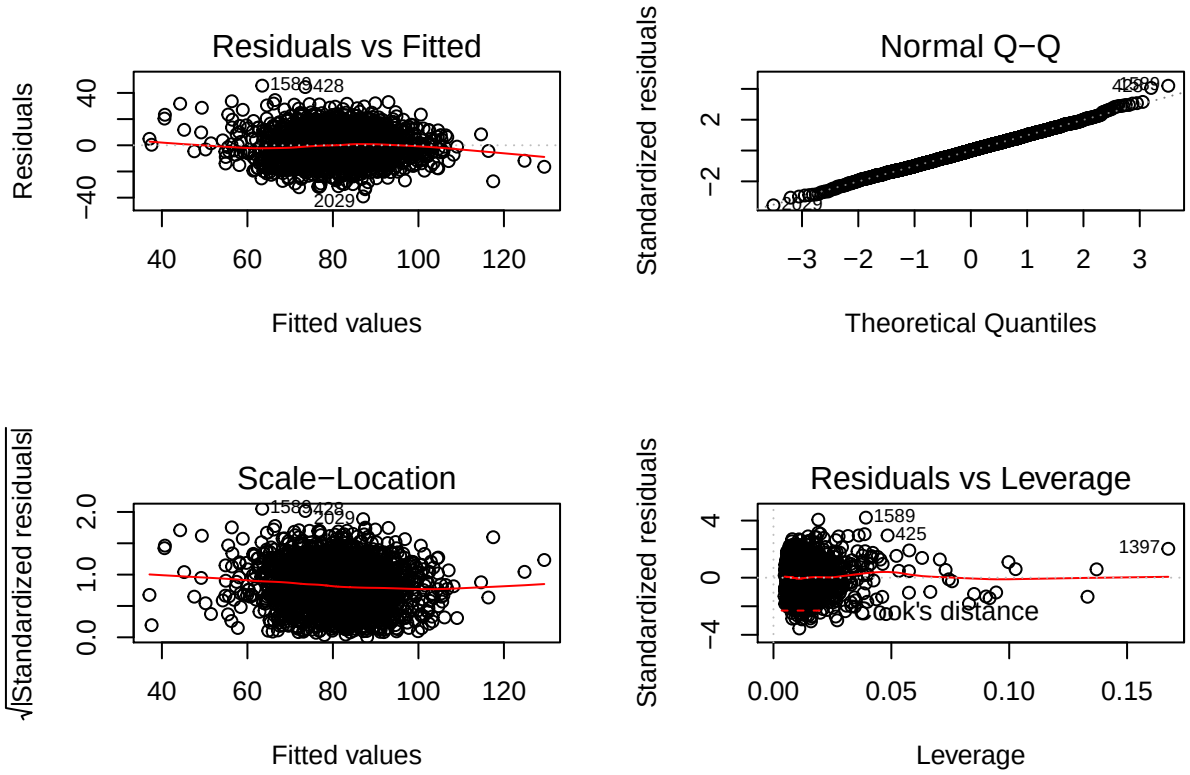
1052, and the 99th percentile is 921. Apart from the 20 or so values above 900, the distribution looks pretty normal. I will create a bucket variable using quantiles.

- TEAM_PITCHING_SO: with a mean of 817 and a sd of 553, a traditional outlier is anything above 2476 and the 99th percentile is 1466. There were 6 values above 2476. When these are removed, the distribution looks pretty normal. I will create a bucket variable using quantiles.

What else can I do about outliers, influential points, or leverage points? I went through the original variables and examined them for outliers and influential/leverage points and removed any rows that severely negatively impacted the relationship between the variable and TARGET_WINS. After doing so, the adjusted R^2 from using all variables went down from the starting value to 0.3891. After removing additional rows that were influential outliers based on the above model, the adjusted R^2 went up to 0.4119.

```
##
## Call:
## lm(formula = train$TARGET_WINS ~ ., data = train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -39.075  -7.498   0.110   7.305  45.535
##
## Coefficients: (1 not defined because of singularities)
##                                       Estimate Std. Error t value
## (Intercept)                         52.6129779  7.0272770   7.487
## TEAM_BATTING_H                      -0.0002725  0.0067076  -0.041
## TEAM_BATTING_2B                     -0.0133770  0.0089091  -1.502
## TEAM_BATTING_3B                      0.1423454  0.0166657   8.541
## TEAM_BATTING_HR                      0.0864444  0.0091617   9.435
## TEAM_BATTING_BB                      0.1048122  0.0165008   6.352
## TEAM_BATTING_SO                     -0.0134150  0.0095473  -1.405
## TEAM_BASERUN_SB                      0.0719504  0.0051530  13.963
## TEAM_BASERUN_CS                     -0.0242967  0.0156121  -1.556
## TEAM_PITCHING_H                      0.0149650  0.0027268   5.488
## TEAM_PITCHING_BB                    -0.0626756  0.0135465  -4.627
## TEAM_PITCHING_SO                    -0.0006366  0.0079750  -0.080
## TEAM_FIELDING_E                     -0.0971193  0.0047849 -20.297
## TEAM_FIELDING_DP                    -0.0903749  0.0128429  -7.037
## TEAM_BATTING_SO_NA                   5.1003492  2.4421605   2.088
## TEAM_BASERUN_SB_NA                  45.5315580  2.7340401  16.654
## TEAM_BASERUN_CS_NA                   3.9266924  0.9191444   4.272
## TEAM_BATTING_HBP_NA                  3.0163376  1.0501648   2.872
## TEAM_PITCHING_SO_NA                         NA         NA      NA
## TEAM_FIELDING_DP_NA                  8.1711050  1.7107960   4.776
## TEAM_PITCHING_H_Bin(1.42e+03,1.52e+03]  2.1191253  0.7749659   2.734
## TEAM_PITCHING_H_Bin(1.52e+03,1.68e+03]  2.7619841  1.0409498   2.653
## TEAM_PITCHING_H_Bin(1.68e+03,3.01e+04]  6.4744787  1.6502183   3.923
## TEAM_PITCHING_BB_Bin(476,536]       -1.0229795  0.8403263  -1.217
## TEAM_PITCHING_BB_Bin(536,611]       -0.5053825  1.1156818  -0.453
## TEAM_PITCHING_BB_Bin(611,3.64e+03]  -1.6516370  1.6769537  -0.985
## TEAM_PITCHING_SO_Bin(626,814]        1.2232185  1.0029628   1.220
## TEAM_PITCHING_SO_Bin(814,957]        2.1588700  1.5362444   1.405
## TEAM_PITCHING_SO_Bin(957,1.93e+04]  -0.2277172  2.0981185  -0.109
##                                     Pr(>|t|)
## (Intercept)                         1.02e-13 ***
## TEAM_BATTING_H                       0.96759
## TEAM_BATTING_2B                      0.13337
```

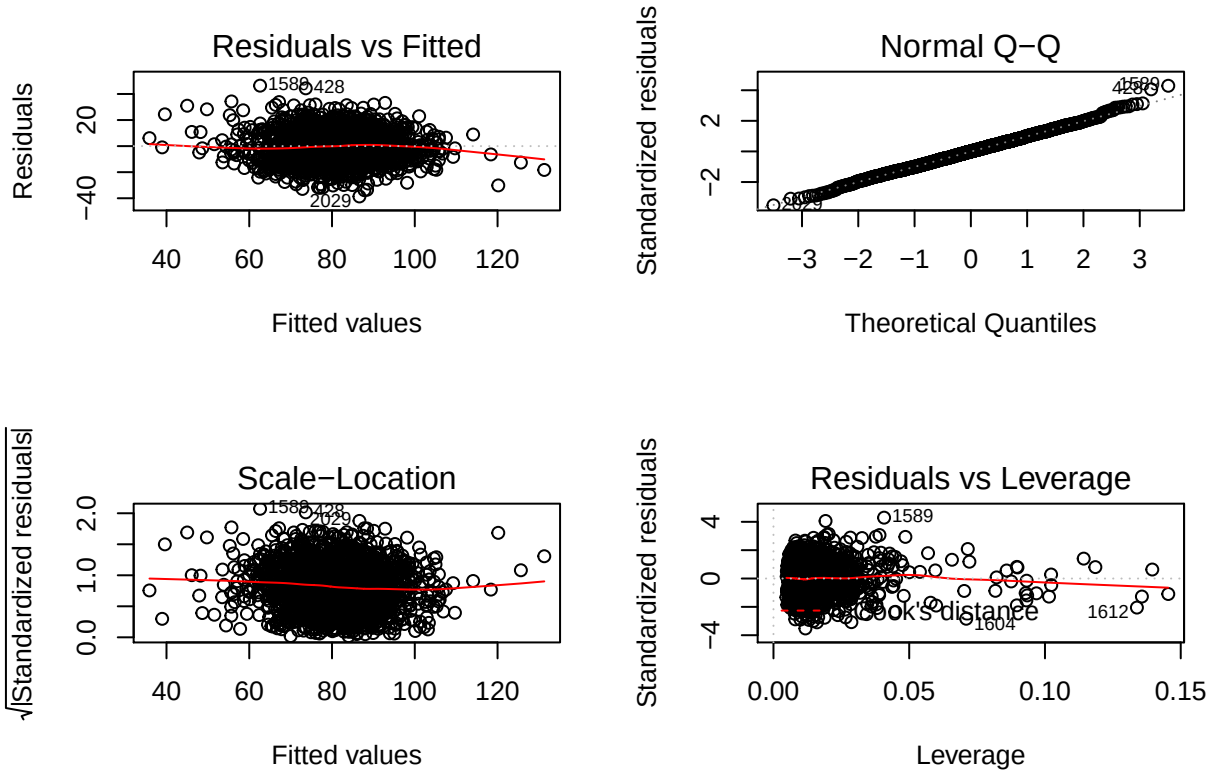
```
## TEAM_BATTING_3B                          < 2e-16 ***
## TEAM_BATTING_HR                          < 2e-16 ***
## TEAM_BATTING_BB                         2.58e-10 ***
## TEAM_BATTING_SO                          0.16013
## TEAM_BASERUN_SB                          < 2e-16 ***
## TEAM_BASERUN_CS                          0.11979
## TEAM_PITCHING_H                         4.53e-08 ***
## TEAM_PITCHING_BB                        3.93e-06 ***
## TEAM_PITCHING_SO                         0.93638
## TEAM_FIELDING_E                          < 2e-16 ***
## TEAM_FIELDING_DP                        2.62e-12 ***
## TEAM_BATTING_SO_NA                       0.03687 *
## TEAM_BASERUN_SB_NA                       < 2e-16 ***
## TEAM_BASERUN_CS_NA                      2.02e-05 ***
## TEAM_BATTING_HBP_NA                      0.00411 **
## TEAM_PITCHING_SO_NA                           NA
## TEAM_FIELDING_DP_NA                     1.91e-06 ***
## TEAM_PITCHING_H_Bin(1.42e+03,1.52e+03]  0.00630 **
## TEAM_PITCHING_H_Bin(1.52e+03,1.68e+03]  0.00803 **
## TEAM_PITCHING_H_Bin(1.68e+03,3.01e+04] 9.00e-05 ***
## TEAM_PITCHING_BB_Bin(476,536]            0.22360
## TEAM_PITCHING_BB_Bin(536,611]            0.65061
## TEAM_PITCHING_BB_Bin(611,3.64e+03]       0.32478
## TEAM_PITCHING_SO_Bin(626,814]            0.22275
## TEAM_PITCHING_SO_Bin(814,957]            0.16008
## TEAM_PITCHING_SO_Bin(957,1.93e+04]       0.91358
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 11.05 on 2179 degrees of freedom
##   (12 observations deleted due to missingness)
## Multiple R-squared:  0.4191, Adjusted R-squared:  0.4119
## F-statistic: 58.22 on 27 and 2179 DF,  p-value: < 2.2e-16
```



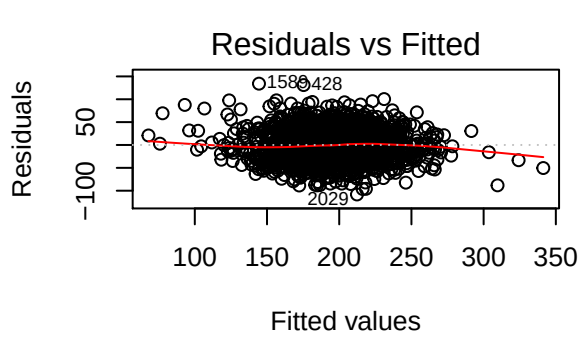
Before doing any transformations, I combined and created new variables. I have seen **Moneyball** and I've worked on this problem before, so I recalled that "getting on base" is a very important variable, as is slugging. Here are my additional variables:

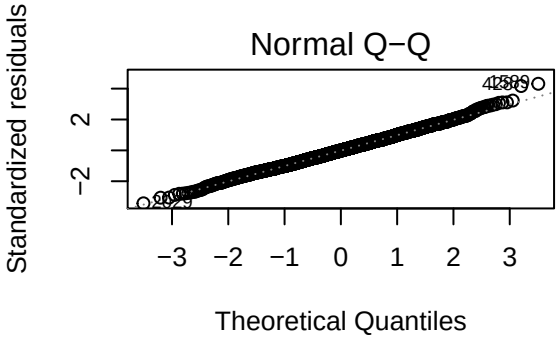
- Single base hits: Hits - 2B - 3B - HR
- On base: hits + walks + HBP (which has too many NAs, so ignoring this)
- At bats: on base + strikeouts
- on base percentage: on base / at bats
- pitching on base: hits + BB
- pitching on base percentage: pitching on base / (pitching on base + pitching strikeout)
- slugging: 1B + 2*2B + 3*3B + 4*HR
- slugging percentage: slugging / at bats
- on base + slugging: on base + slugging

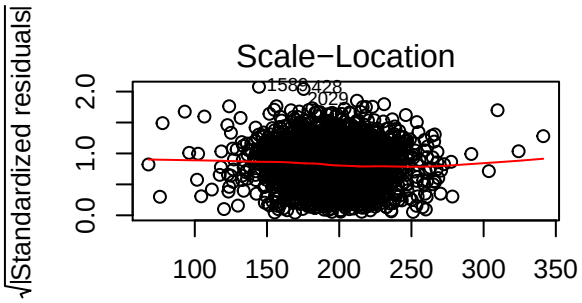
After adding these variables, removing one outlier, and rerunning the baseline model, I got an adjusted R^2 of 0.4181.

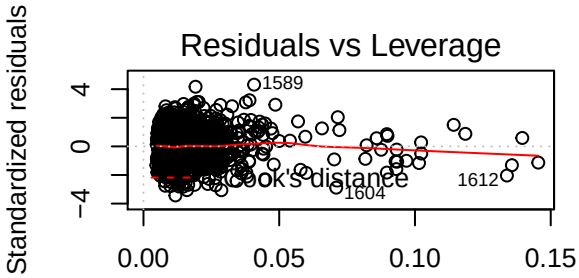


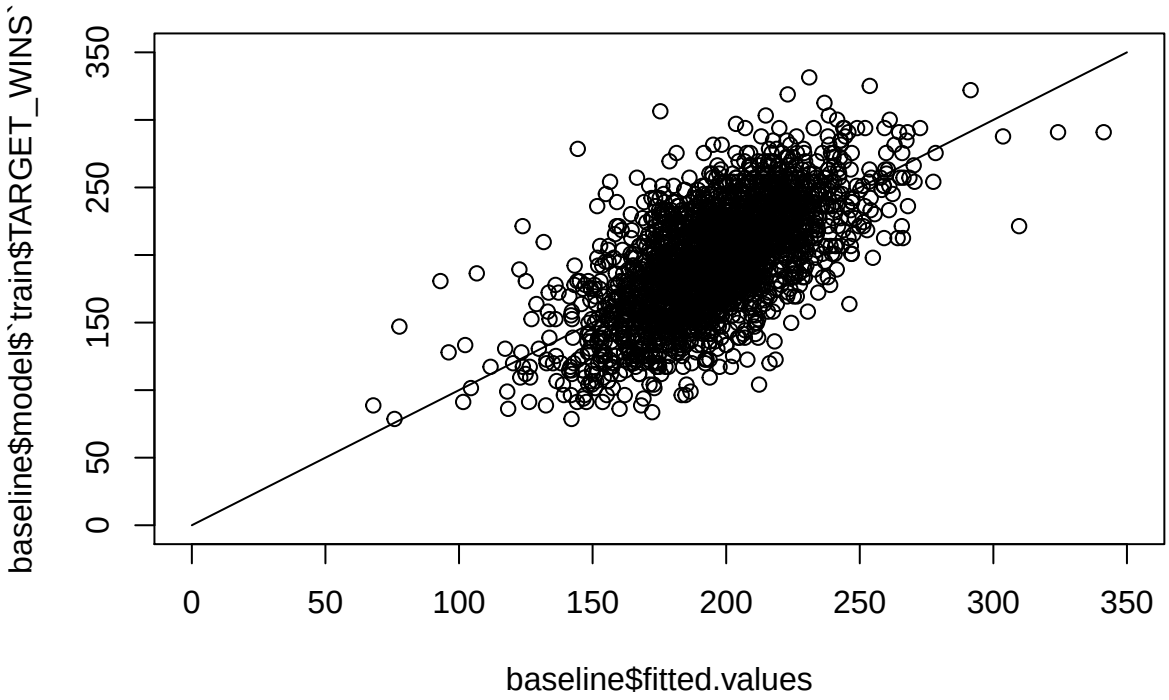
Finally, in order to make the variables more normal, I used various boxcox transformations. A boxcox analysis on TARGET_WINS using the baseline model suggested that raising TARGET_WINS to the power of 1.2 made it more normal. The adjusted R^2 went down a little to 0.4133. I included a plot of fitted values vs. actual TARGET_WINS below for reference.


Residuals vs Fitted
Residuals
Fitted values
1589
428
2029


Normal Q−Q
Standardized residuals
Theoretical Quantiles
1589

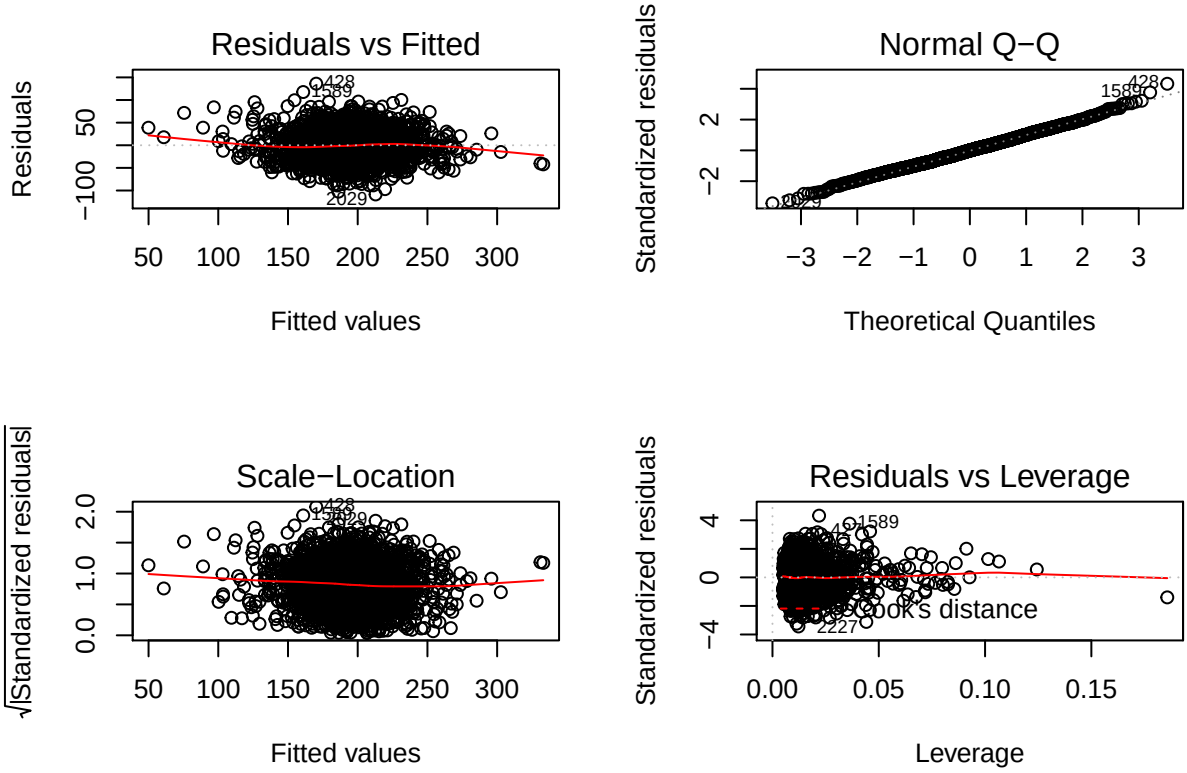

Scale−Location
√|Standardized residuals|
Fitted values
1589
428
2029

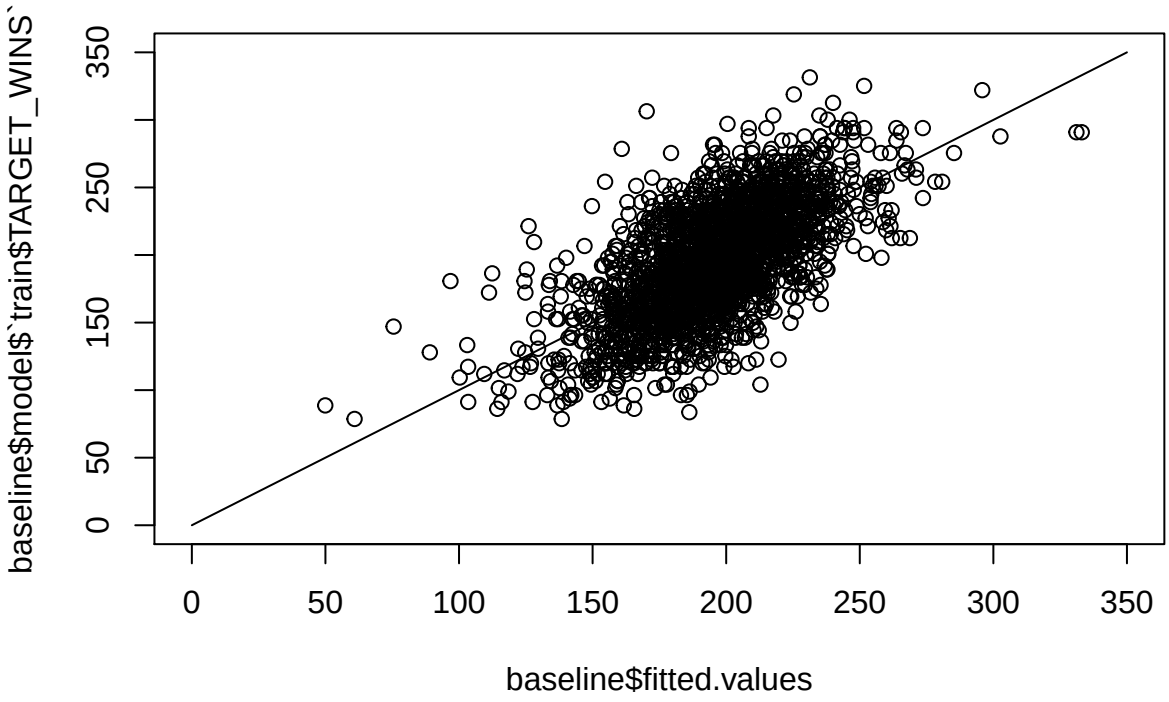

Residuals vs Leverage
Standardized residuals
Leverage
Cook's distance
1589
1612
1604



For transforming the predictor values, I decided to use invTranPlot and invTranEstimate from the alr3 package to find the optimal power transformation. Despite going through each variable and manually adjusting for the best fit, the adjusted R^2 didn't change much, and ended up at 0.41. The F-statistic also dropped from 52.78 to 48.87.

Importantly, the attempt to normalize TEAM_FIELDING_E dropped the R^2 to 0.3174. After some attempts to rectify this to no success, I decided to leave the variable as is.


Residuals vs Fitted
Residuals
Fitted values
Normal Q−Q
Standardized residuals
Theoretical Quantiles
Scale−Location
√|Standardized residuals|
Fitted values
Residuals vs Leverage
Standardized residuals
Cook's distance
Leverage



3. Build Models

Using the training data set, build at least three different multiple linear regression models, using different variables (or the same variables with different transformations). Since we have not yet covered automated variable selection methods, you should select the variables manually (unless you previously learned Forward or Stepwise selection, etc.). Since you manually selected a variable for inclusion into the model or exclusion into the model,indicate why this was done.

Discuss the coefficients in the models, do they make sense? For example, if a team hits a lot of Home Runs, it would be reasonably expected that such a team would win more games. However, if the coefficient is negative (suggesting that the team would lose more games), then that needs to be discussed. Are you keeping the model even though it is counter intuitive? Why? The boss needs to know.

Answer

The first model uses all variables up to this point. Remember that both:

- TEAM_BATTING_HBP (2085 missing values)
- TEAM_PITCHING_HR

were excluded, the first due to missing values, and the second due to collinearity with TEAM_BATTING_HR.

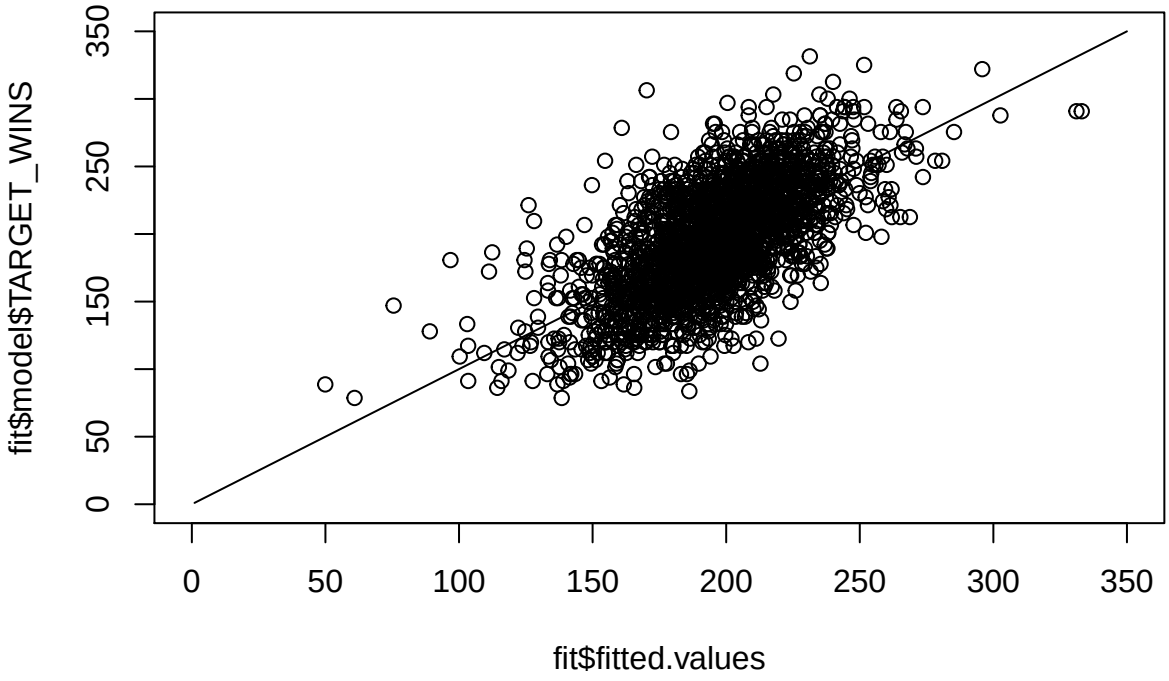
With an adjusted R^2 of 0.41 , an F-statistic of 48.87, and RSME of 31.83, there were lots of insignificant variables (p-value$>$ 0.1) and many variables that were not defined because they were linear combinations of other variables. The residuals looked pretty normally distributed, and appeared to be independent, but there

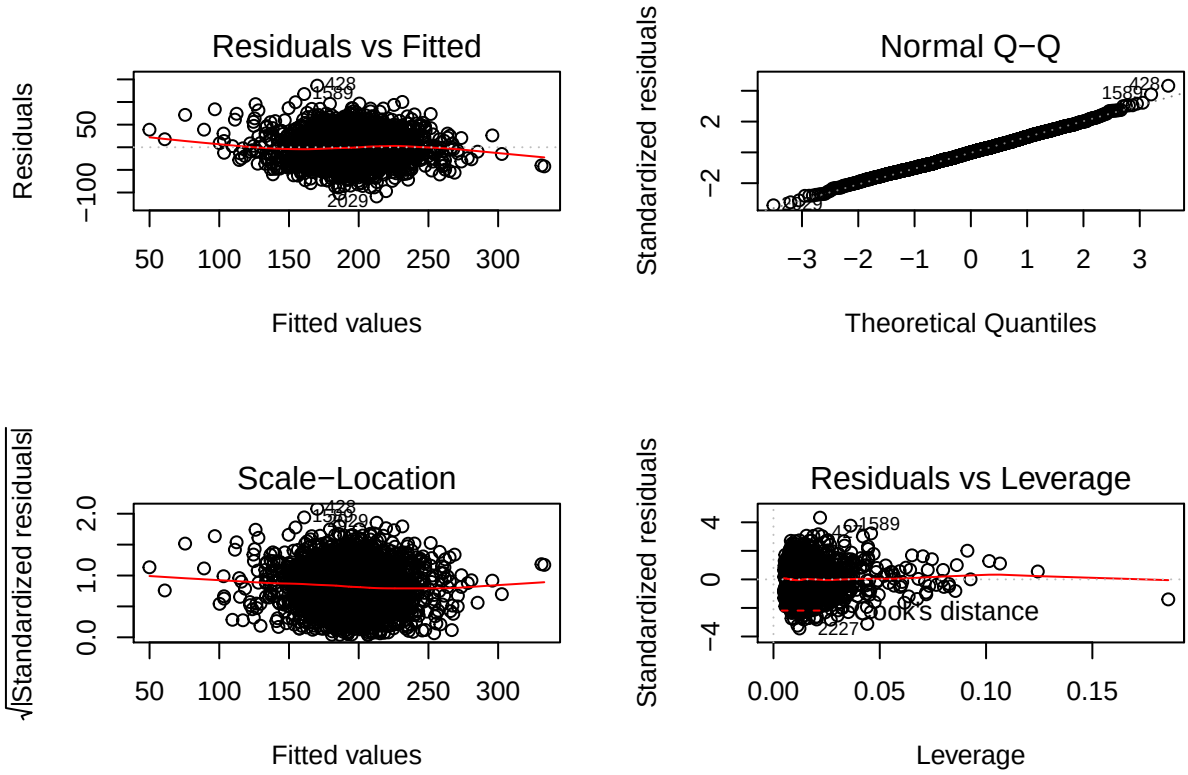
was not constant variance. The residuals were small in both the lower and upper values, while they expanded in the middle values. I also wonder if the linear model is appropriate here.

The model starts with an intercept of about 6639000 and then proceeds to add and subtract values based on coefficients. This is why many variables that we would think should positively correlate with wins (e.g., TEAM_BATTING_H) do not. This is not an easily interpretable model and I would not use it.

```
##
## Call:
## lm(formula = TARGET_WINS ~ ., data = train)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -108.714  -22.646   -0.091   20.771  136.128
##
## Coefficients: (5 not defined because of singularities)
##                                          Estimate Std. Error t value
## (Intercept)                             6.639e+06  3.242e+06   2.048
## TEAM_BATTING_H                         -9.783e+06  4.909e+06  -1.993
## TEAM_BATTING_2B                        -2.546e+05  1.477e+05  -1.723
## TEAM_BATTING_3B                         1.069e-04  3.145e-05   3.400
## TEAM_BATTING_HR                         3.442e-06  5.365e-06   0.642
## TEAM_BATTING_BB                         1.240e-09  2.225e-10   5.572
## TEAM_BATTING_SO                                NA         NA      NA
## TEAM_BASERUN_SB                         5.828e-02  4.106e-03  14.194
## TEAM_BASERUN_CS                        -3.734e-02  4.451e-02  -0.839
## TEAM_PITCHING_H                                NA         NA      NA
## TEAM_PITCHING_BB                       -1.343e-02  6.354e-03  -2.114
## TEAM_PITCHING_SO                        3.509e-01  3.529e-01   0.994
## TEAM_FIELDING_E                        -2.581e-01  1.351e-02 -19.102
## TEAM_FIELDING_DP                       -2.900e-01  3.691e-02  -7.856
## TEAM_BATTING_SO_NA                      1.612e+01  8.681e+00   1.857
## TEAM_BASERUN_SB_NA                      1.309e+02  8.458e+00  15.473
## TEAM_BASERUN_CS_NA                      1.173e+01  2.654e+00   4.419
## TEAM_BATTING_HBP_NA                     8.224e+00  3.037e+00   2.708
## TEAM_PITCHING_SO_NA                            NA         NA      NA
## TEAM_FIELDING_DP_NA                     2.032e+01  4.827e+00   4.210
## TEAM_PITCHING_H_Bin(1.42e+03,1.52e+03]  3.388e+00  2.468e+00   1.373
## TEAM_PITCHING_H_Bin(1.52e+03,1.68e+03]  3.967e+00  3.247e+00   1.221
## TEAM_PITCHING_H_Bin(1.68e+03,3.01e+04]  1.180e+01  4.644e+00   2.541
## TEAM_PITCHING_BB_Bin(476,536]           7.418e-01  2.742e+00   0.271
## TEAM_PITCHING_BB_Bin(536,611]           3.945e+00  3.676e+00   1.073
## TEAM_PITCHING_BB_Bin(611,3.64e+03]      1.900e+00  5.072e+00   0.375
## TEAM_PITCHING_SO_Bin(626,814]           1.679e+00  3.258e+00   0.515
## TEAM_PITCHING_SO_Bin(814,957]           1.918e+00  4.867e+00   0.394
## TEAM_PITCHING_SO_Bin(957,1.93e+04]     -4.483e+00  6.161e+00  -0.728
## TEAM_BATTING_1B                         7.065e-05  4.258e-05   1.659
## TEAM_BATTING_OnBase                     1.330e+02  2.755e+01   4.829
## TEAM_BATTING_AtBat                             NA         NA      NA
## TEAM_BATTING_OnBase_PERC                4.292e+02  1.863e+02   2.304
## TEAM_PITCHING_OnBase                           NA         NA      NA
## TEAM_PITCHING_OnBase_PERC               2.620e+00  9.362e-01   2.799
## TEAM_BATTING_SLUGGING                   3.903e+01  7.736e+00   5.045
## TEAM_BATTING_SLUGGING_PERC             -1.666e+02  1.127e+02  -1.478
## TEAM_BATTING_OnBaseSlugging            -1.174e+01  2.228e+00  -5.270
```

```
##                                          Pr(>|t|)
## (Intercept)                              0.040719 *
## TEAM_BATTING_H                           0.046399 *
## TEAM_BATTING_2B                          0.084964 .
## TEAM_BATTING_3B                          0.000686 ***
## TEAM_BATTING_HR                          0.521225
## TEAM_BATTING_BB                          2.83e-08 ***
## TEAM_BATTING_SO                                NA
## TEAM_BASERUN_SB                           < 2e-16 ***
## TEAM_BASERUN_CS                          0.401578
## TEAM_PITCHING_H                                NA
## TEAM_PITCHING_BB                         0.034621 *
## TEAM_PITCHING_SO                         0.320122
## TEAM_FIELDING_E                           < 2e-16 ***
## TEAM_FIELDING_DP                         6.19e-15 ***
## TEAM_BATTING_SO_NA                       0.063378 .
## TEAM_BASERUN_SB_NA                        < 2e-16 ***
## TEAM_BASERUN_CS_NA                       1.04e-05 ***
## TEAM_BATTING_HBP_NA                      0.006826 **
## TEAM_PITCHING_SO_NA                            NA
## TEAM_FIELDING_DP_NA                      2.65e-05 ***
## TEAM_PITCHING_H_Bin(1.42e+03,1.52e+03] 0.169938
## TEAM_PITCHING_H_Bin(1.52e+03,1.68e+03] 0.222055
## TEAM_PITCHING_H_Bin(1.68e+03,3.01e+04] 0.011121 *
## TEAM_PITCHING_BB_Bin(476,536]          0.786788
## TEAM_PITCHING_BB_Bin(536,611]          0.283299
## TEAM_PITCHING_BB_Bin(611,3.64e+03]     0.708048
## TEAM_PITCHING_SO_Bin(626,814]          0.606318
## TEAM_PITCHING_SO_Bin(814,957]          0.693628
## TEAM_PITCHING_SO_Bin(957,1.93e+04]     0.466938
## TEAM_BATTING_1B                          0.097213 .
## TEAM_BATTING_OnBase                      1.47e-06 ***
## TEAM_BATTING_AtBat                             NA
## TEAM_BATTING_OnBase_PERC                 0.021340 *
## TEAM_PITCHING_OnBase                           NA
## TEAM_PITCHING_OnBase_PERC                0.005171 **
## TEAM_BATTING_SLUGGING                    4.92e-07 ***
## TEAM_BATTING_SLUGGING_PERC               0.139470
## TEAM_BATTING_OnBaseSlugging              1.50e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 31.83 on 2172 degrees of freedom
##   (12 observations deleted due to missingness)
## Multiple R-squared:  0.4186, Adjusted R-squared:   0.41
## F-statistic: 48.87 on 32 and 2172 DF,  p-value: < 2.2e-16
```


fit$model$TARGET_WINS
0
50
150
250
350
0
50
100
150
200
250
300
350
fit$fitted.values

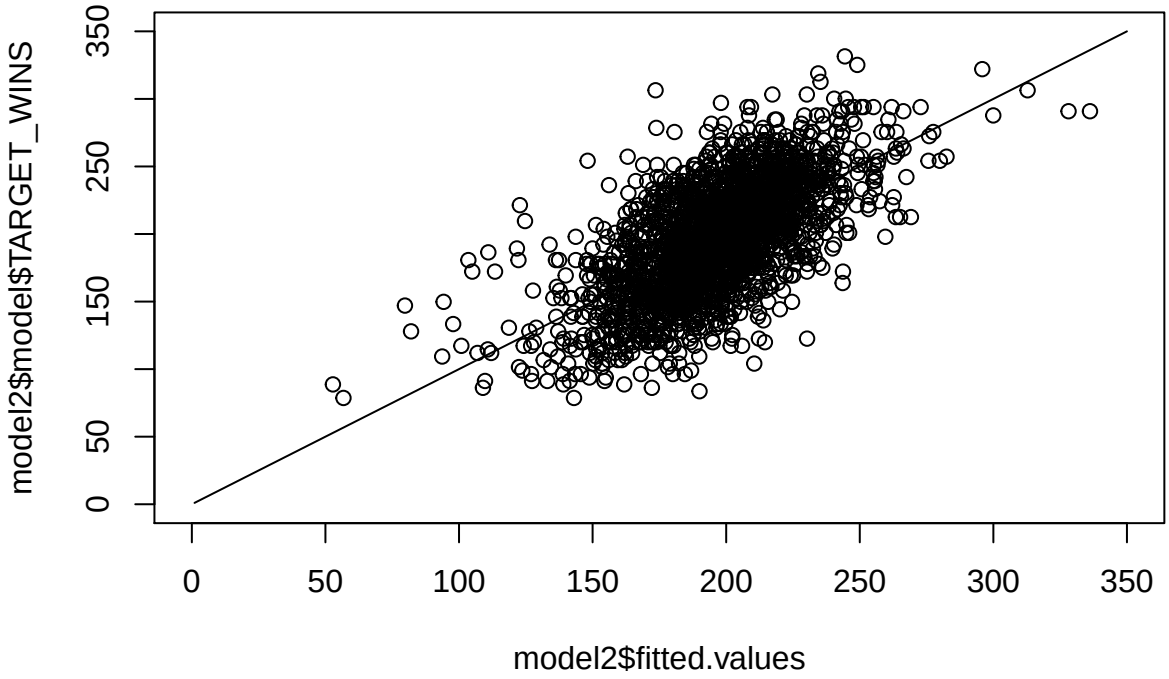


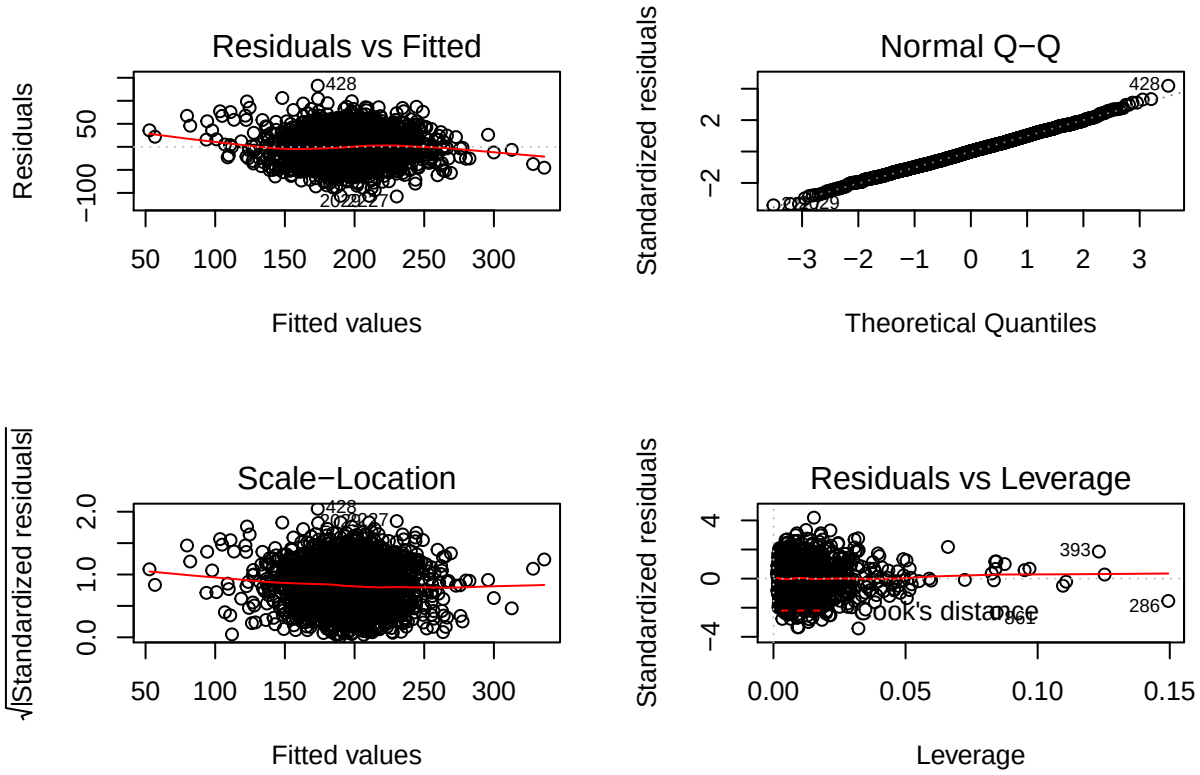
My second model used stepAIC to do backward search, starting with the first model, in order to limit the variables. It had an adjusted R^2 of 0.4066 . Almost every variable was now significant, and the F-statistic went up to 76.93. The RSME didn't change much at 31.96. However, the model suffered the same issues as the first. The residuals looked pretty normally distributed, and appeared to be independent, but there was not constant variance.

The same issues also apply to the coefficients. With an intercept of 188500, second base hits have a negative coefficient, suggesting that fewer of these is actually better. Similarly, OnBaseSlugging is is negatively associated. This is also not easily interpretable.

```
##
## Call:
## lm(formula = TARGET_WINS ~ TEAM_BATTING_2B + TEAM_BATTING_3B +
##     TEAM_BATTING_BB + TEAM_BASERUN_SB + TEAM_PITCHING_BB + TEAM_PITCHING_SO +
##     TEAM_FIELDING_E + TEAM_FIELDING_DP + TEAM_BATTING_SO_NA +
##     TEAM_BASERUN_SB_NA + TEAM_BASERUN_CS_NA + TEAM_BATTING_HBP_NA +
##     TEAM_FIELDING_DP_NA + TEAM_BATTING_1B + TEAM_BATTING_OnBase +
##     TEAM_BATTING_OnBase_PERC + TEAM_PITCHING_OnBase_PERC + TEAM_BATTING_SLUGGING +
##     TEAM_BATTING_SLUGGING_PERC + TEAM_BATTING_OnBaseSlugging,
##     data = train)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -107.687  -22.142    0.339   21.074  132.813
##
## Coefficients:
##                                Estimate Std. Error t value Pr(>|t|)
```

```
## (Intercept)                 1.885e+05  6.393e+04   2.949 0.003223 **
## TEAM_BATTING_2B            -3.472e+05  1.163e+05  -2.986 0.002858 **
## TEAM_BATTING_3B             8.509e-05  2.801e-05   3.038 0.002410 **
## TEAM_BATTING_BB             1.076e-09  1.935e-10   5.564 2.96e-08 ***
## TEAM_BASERUN_SB             5.689e-02  3.853e-03  14.764  < 2e-16 ***
## TEAM_PITCHING_BB            7.867e-03  3.122e-03   2.520 0.011810 *
## TEAM_PITCHING_SO           -6.919e-01  1.016e-01  -6.810 1.26e-11 ***
## TEAM_FIELDING_E            -2.327e-01  1.196e-02 -19.459  < 2e-16 ***
## TEAM_FIELDING_DP           -2.884e-01  3.653e-02  -7.897 4.47e-15 ***
## TEAM_BATTING_SO_NA          3.706e+01  4.532e+00   8.177 4.86e-16 ***
## TEAM_BASERUN_SB_NA          1.391e+02  7.539e+00  18.458  < 2e-16 ***
## TEAM_BASERUN_CS_NA          1.210e+01  2.449e+00   4.939 8.44e-07 ***
## TEAM_BATTING_HBP_NA         8.724e+00  2.887e+00   3.022 0.002538 **
## TEAM_FIELDING_DP_NA         1.478e+01  4.395e+00   3.363 0.000783 ***
## TEAM_BATTING_1B             7.229e-05  2.343e-05   3.085 0.002064 **
## TEAM_BATTING_OnBase         1.034e+02  2.146e+01   4.817 1.56e-06 ***
## TEAM_BATTING_OnBase_PERC    2.306e+02  1.585e+02   1.455 0.145785
## TEAM_PITCHING_OnBase_PERC   3.860e+00  7.819e-01   4.937 8.51e-07 ***
## TEAM_BATTING_SLUGGING       4.065e+01  7.148e+00   5.687 1.47e-08 ***
## TEAM_BATTING_SLUGGING_PERC -2.491e+02  9.649e+01  -2.582 0.009886 **
## TEAM_BATTING_OnBaseSlugging -1.143e+01  2.052e+00  -5.571 2.84e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 31.96 on 2196 degrees of freedom
## Multiple R-squared:  0.412,  Adjusted R-squared:  0.4066
## F-statistic: 76.93 on 20 and 2196 DF,  p-value: < 2.2e-16
```

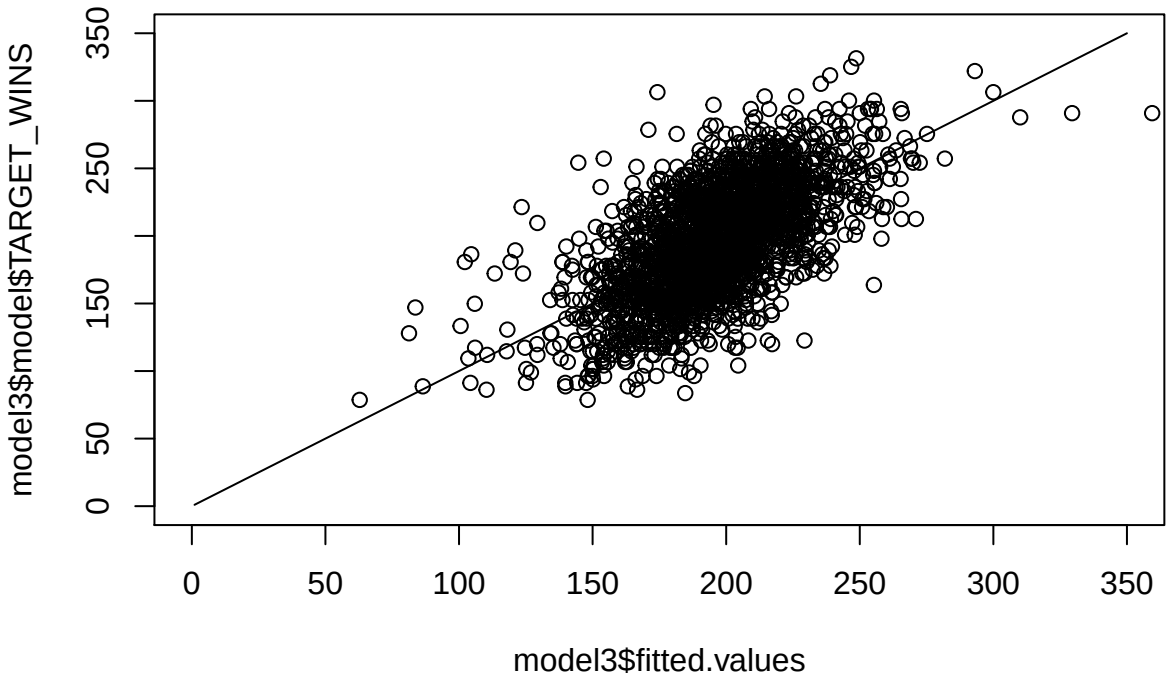

model2$model$TARGET_WINS
0
50
150
250
350
0
50
100
150
200
250
300
350
model2$fitted.values

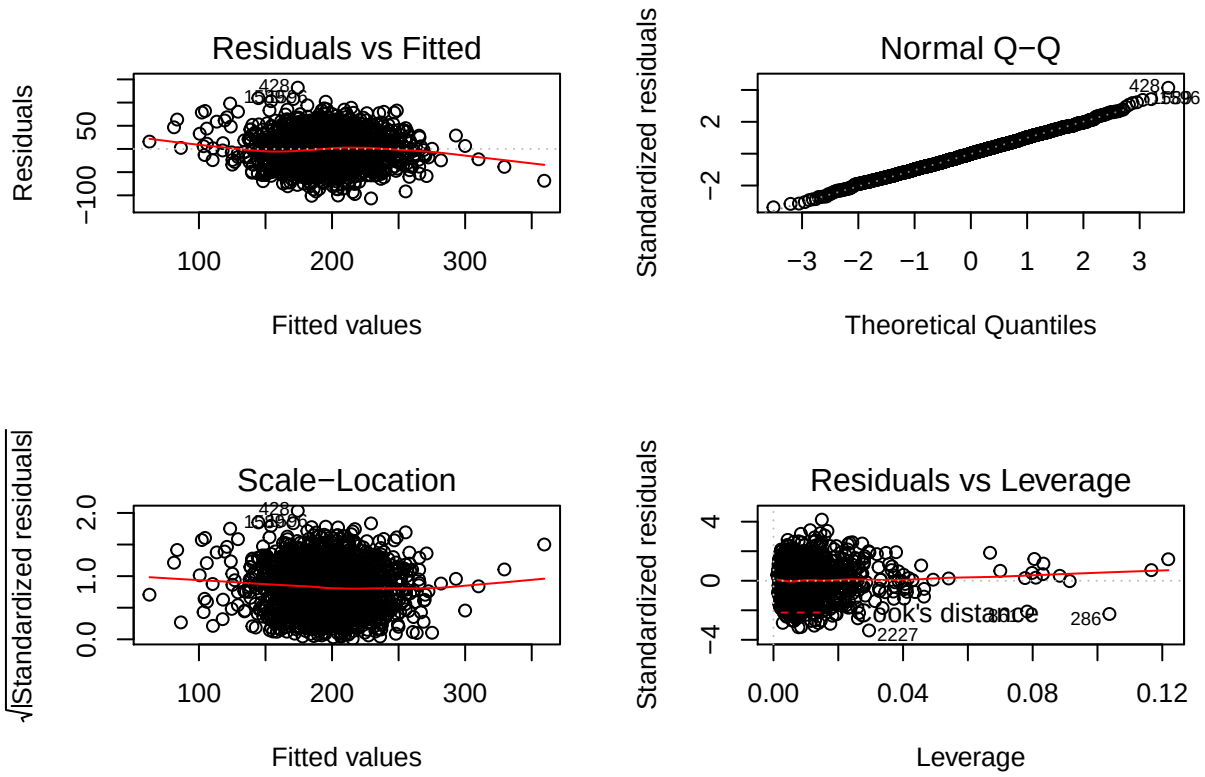


My third model does a forward stepwise search. It has an adjusted R^2 went down a little to 0.3985, and not all variables are significant, but the F-statistic is higher at 82.56. The RSME is 32.18. Interestingly, the intercept is now negative at -1568000. Hits is now positively correlated, but 2B is negatively correlated. So still not easily interpretable.

```
##
## Call:
## lm(formula = TARGET_WINS ~ TEAM_BATTING_H + TEAM_BATTING_BB +
##     TEAM_FIELDING_DP + TEAM_BATTING_SLUGGING + TEAM_BASERUN_SB +
##     TEAM_PITCHING_OnBase_PERC + TEAM_BASERUN_SB_NA + TEAM_FIELDING_E +
##     TEAM_PITCHING_SO + TEAM_BATTING_SO_NA + TEAM_BATTING_2B +
##     TEAM_BASERUN_CS_NA + TEAM_BATTING_HBP_NA + TEAM_FIELDING_DP_NA +
##     TEAM_BATTING_OnBase_PERC + TEAM_BATTING_3B + TEAM_PITCHING_BB +
##     TEAM_BASERUN_CS, data = train)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -106.720  -22.475   -0.066   21.326  132.126
##
## Coefficients:
##                             Estimate Std. Error t value Pr(>|t|)
## (Intercept)               -1.568e+06  9.984e+05  -1.570 0.116516
## TEAM_BATTING_H             2.712e+06  1.515e+06   1.790 0.073665 .
## TEAM_BATTING_BB            5.177e-10  1.456e-10   3.555 0.000386 ***
## TEAM_FIELDING_DP          -2.867e-01  3.676e-02  -7.798 9.65e-15 ***
## TEAM_BATTING_SLUGGING      2.839e+00  3.411e-01   8.323  < 2e-16 ***
## TEAM_BASERUN_SB            5.857e-02  4.023e-03  14.558  < 2e-16 ***
```

```
## TEAM_PITCHING_OnBase_PERC  2.540e+00  5.957e-01   4.264 2.10e-05 ***
## TEAM_BASERUN_SB_NA         1.482e+02  7.141e+00  20.749  < 2e-16 ***
## TEAM_FIELDING_E           -2.294e-01  1.143e-02 -20.063  < 2e-16 ***
## TEAM_PITCHING_SO          -7.642e-01  1.004e-01  -7.608 4.09e-14 ***
## TEAM_BATTING_SO_NA         3.011e+01  4.369e+00   6.891 7.19e-12 ***
## TEAM_BATTING_2B           -4.261e+05  1.060e+05  -4.018 6.06e-05 ***
## TEAM_BASERUN_CS_NA         1.011e+01  2.600e+00   3.889 0.000104 ***
## TEAM_BATTING_HBP_NA        1.170e+01  2.883e+00   4.057 5.15e-05 ***
## TEAM_FIELDING_DP_NA        1.088e+01  4.368e+00   2.490 0.012860 *
## TEAM_BATTING_OnBase_PERC  -1.832e+02  4.067e+01  -4.504 7.01e-06 ***
## TEAM_BATTING_3B            8.811e-05  2.817e-05   3.128 0.001782 **
## TEAM_PITCHING_BB           6.420e-03  2.751e-03   2.333 0.019715 *
## TEAM_BASERUN_CS           -6.402e-02  4.368e-02  -1.466 0.142922
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 32.18 on 2198 degrees of freedom
## Multiple R-squared:  0.4034, Adjusted R-squared:  0.3985
## F-statistic: 82.56 on 18 and 2198 DF,  p-value: < 2.2e-16
```



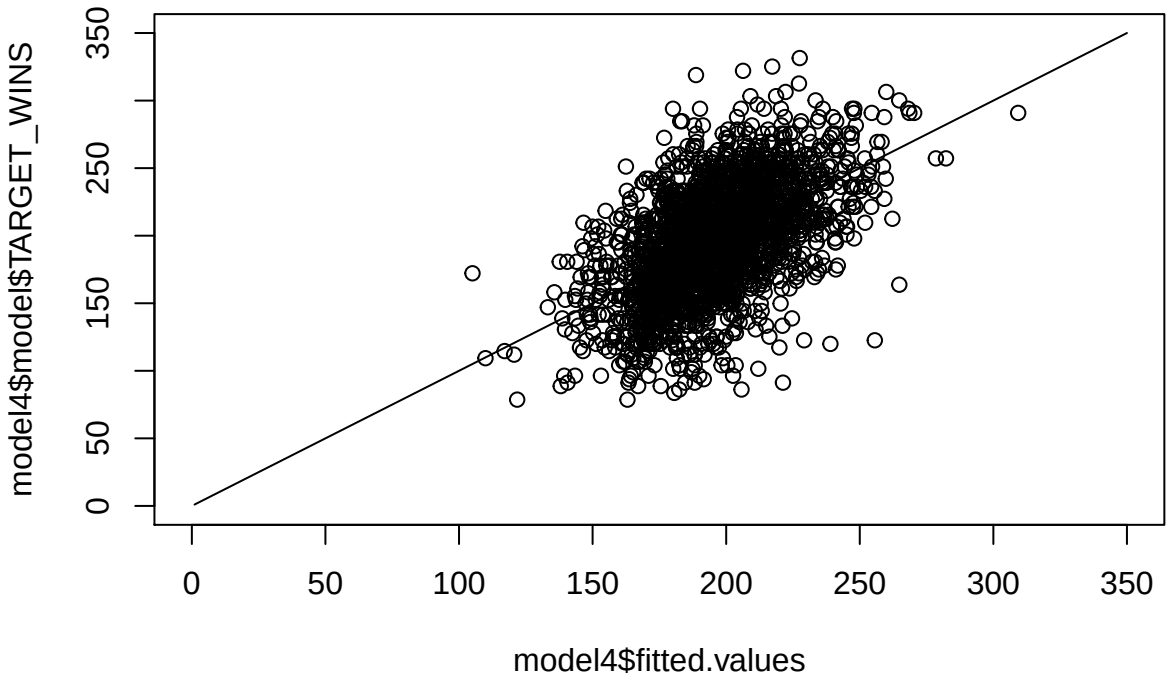


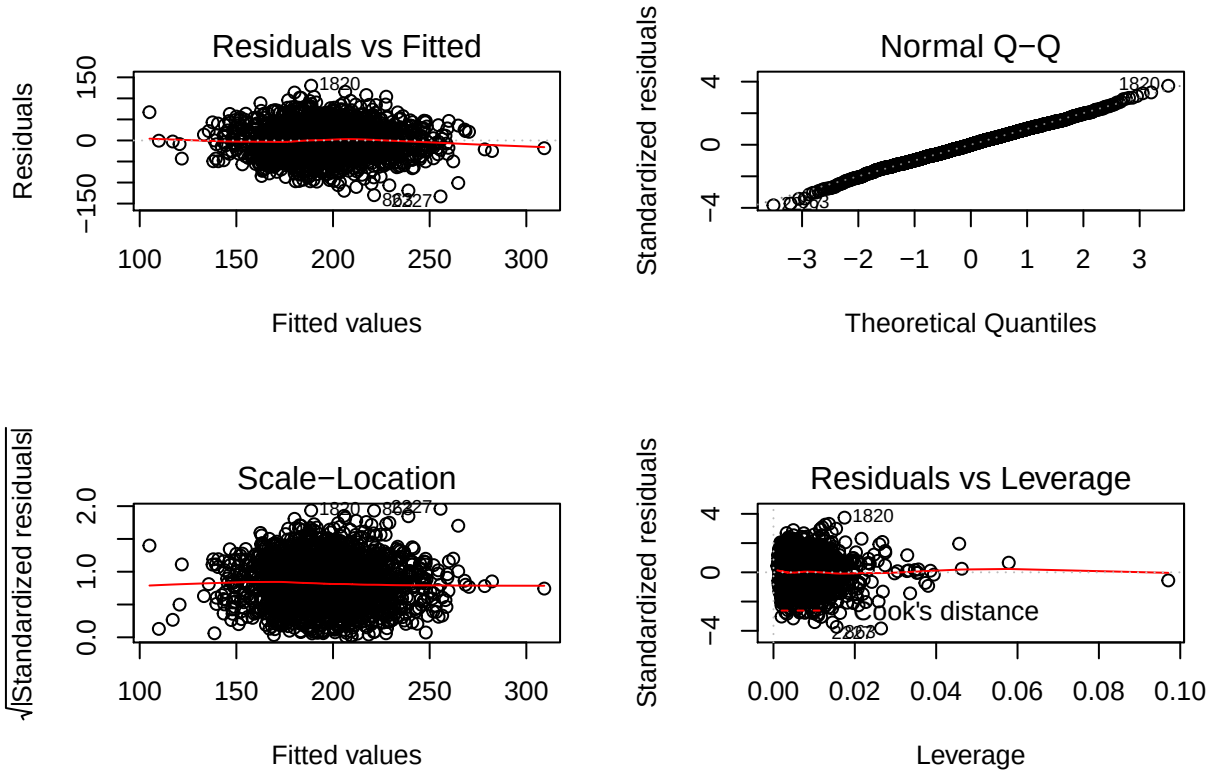
My last model (model 4) starts with all variables, but removes highly correlated variables to avoid multi-collinearity. High correlation is defined here as 0.6 and above or -0.6 and below. So as to avoid removing the most important variables, I created a loop that would start with the highest two correlated variables, and then remove the variable that had the lower p-value.

The adjusted R^2 went down to 0.2833 and the F-statistic dropped a little to 68.39. The RSME went up to 35.13. The same comment on residuals the are above still apply. The coefficients are not as confusing though. The intercept of -193.6 is more reasonable, and variables that we would expect to be positively correlated are: 3B, BB, SB, Slugging. Variables we would expect to be negatively correlated are generally also negative, like CS. So while this is more interpretable, it does not predict as well.

```
##
## Call:
## lm(formula = TARGET_WINS ~ TEAM_BATTING_3B + TEAM_BATTING_BB +
##     TEAM_BATTING_SO + TEAM_BASERUN_SB + TEAM_BASERUN_CS + TEAM_FIELDING_DP +
##     TEAM_BATTING_SO_NA + TEAM_BASERUN_SB_NA + TEAM_BASERUN_CS_NA +
##     TEAM_BATTING_HBP_NA + TEAM_PITCHING_SO_NA + TEAM_FIELDING_DP_NA +
##     TEAM_PITCHING_OnBase_PERC + TEAM_BATTING_SLUGGING + TEAM_BATTING_SLUGGING_PERC,
##     data = train)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -133.044  -23.430    0.141   23.785  130.186
##
## Coefficients: (2 not defined because of singularities)
##                               Estimate Std. Error t value Pr(>|t|)
## (Intercept)                 -1.936e+02  3.330e+01  -5.815 6.94e-09 ***
```

```
## TEAM_BATTING_3B             5.784e-05  2.951e-05   1.960 0.050082 .
## TEAM_BATTING_BB             5.004e-10  1.101e-10   4.545 5.80e-06 ***
## TEAM_BATTING_SO                    NA         NA      NA       NA
## TEAM_BASERUN_SB             2.952e-02  4.087e-03   7.222 7.01e-13 ***
## TEAM_BASERUN_CS            -3.012e-02  4.659e-02  -0.646 0.518083
## TEAM_FIELDING_DP           -1.998e-01  3.981e-02  -5.018 5.63e-07 ***
## TEAM_BATTING_SO_NA          2.175e+01  4.378e+00   4.968 7.30e-07 ***
## TEAM_BASERUN_SB_NA          3.643e+01  4.546e+00   8.012 1.81e-15 ***
## TEAM_BASERUN_CS_NA          1.007e-01  2.718e+00   0.037 0.970447
## TEAM_BATTING_HBP_NA         1.055e+01  3.111e+00   3.391 0.000708 ***
## TEAM_PITCHING_SO_NA                NA         NA      NA       NA
## TEAM_FIELDING_DP_NA        -2.037e+01  4.083e+00  -4.988 6.56e-07 ***
## TEAM_PITCHING_OnBase_PERC   3.355e+00  5.736e-01   5.850 5.65e-09 ***
## TEAM_BATTING_SLUGGING       3.850e+00  2.939e-01  13.100  < 2e-16 ***
## TEAM_BATTING_SLUGGING_PERC -5.734e+01  1.887e+01  -3.039 0.002405 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 35.13 on 2203 degrees of freedom
## Multiple R-squared:  0.2875, Adjusted R-squared:  0.2833
## F-statistic: 68.39 on 13 and 2203 DF,  p-value: < 2.2e-16
```





4. Select Models

Decide on the criteria for selecting the best multiple linear regression model. Will you select a model with slightly worse performance if it makes more sense or is more parsimonious? Discuss why you selected your model.

For the multiple linear regression model, will you use a metric such as Adjusted R^2, RMSE, etc.? Be sure to explain how you can make inferences from the model, discuss multi-collinearity issues (if any), and discuss other relevant model output. Using the training data set, evaluate the multiple linear regression model based on:

a. mean squared error
b. R^2
c. F-statistic
d. residual plots

Make predictions using the evaluation data set.

Answer:

of the four models above, the "best" model in my opinion is model 2. It had the second highest adjusted R^2 at 0.4066, the second highest F-statistic at 76.93, the second lowest RSME, and nearly every variable used was significant. In short, it does the best job of balancing all of these important factors. Multi-collinearity does not seem to be much of an issue as my attempt to remove it made the model worse (model 4). Although the coefficients were a little easier to interpret, there wasn't much improvement here and I don't think it is worth the trade off.

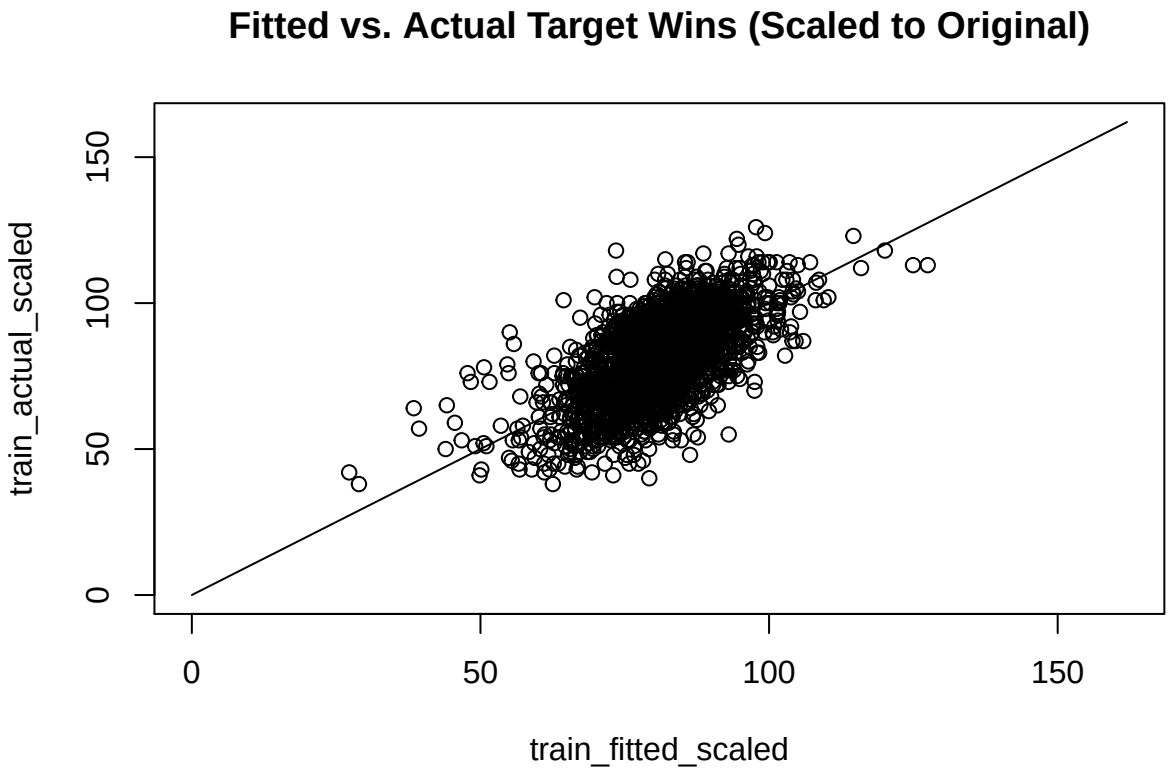
In terms of residuals, each model had the same issues: the residuals looked pretty normally distributed, and appeared to be independent (no pattern was discernible), but there was not constant variance. I think this is likely due to a limitation in the data and or the use of a linear model to begin with. In any case, I couldn't determine how to remove this issue from the model.

Since model2 is the "best" model, I will use it to make predictions in the evaluation data set. After doing so, the unscaled predictions have two values that are negative, which is impossible. Scaling back using the same transformation, these values become NAs (and are basically excluded as outliers).

Comparing the training actuals vs. training fitted vs. test predictions (no test actuals are available), the distributions are fairly similar. Medians are all around 82, the 1st quartile is mid 70s and 3rd quartile is upper 80s. The maximum values are pretty close too. The training fitted and actual minimums are at 27 and 38 respectively, but the prediction minimum after transformation back is at 17. This should actually be 0 considering the negative values that could not be transformed back. In short, the prediction distribution has a larger standard deviation and is slightly shifted toward the lower values.

The predictions for the test set are output:

- test_index (the original index for the test set row)
- predictions_scale (the prediction for that row, scaled appropriately. The NA values have been set to 0)




```
## [1] "train_actual_scaled"
```

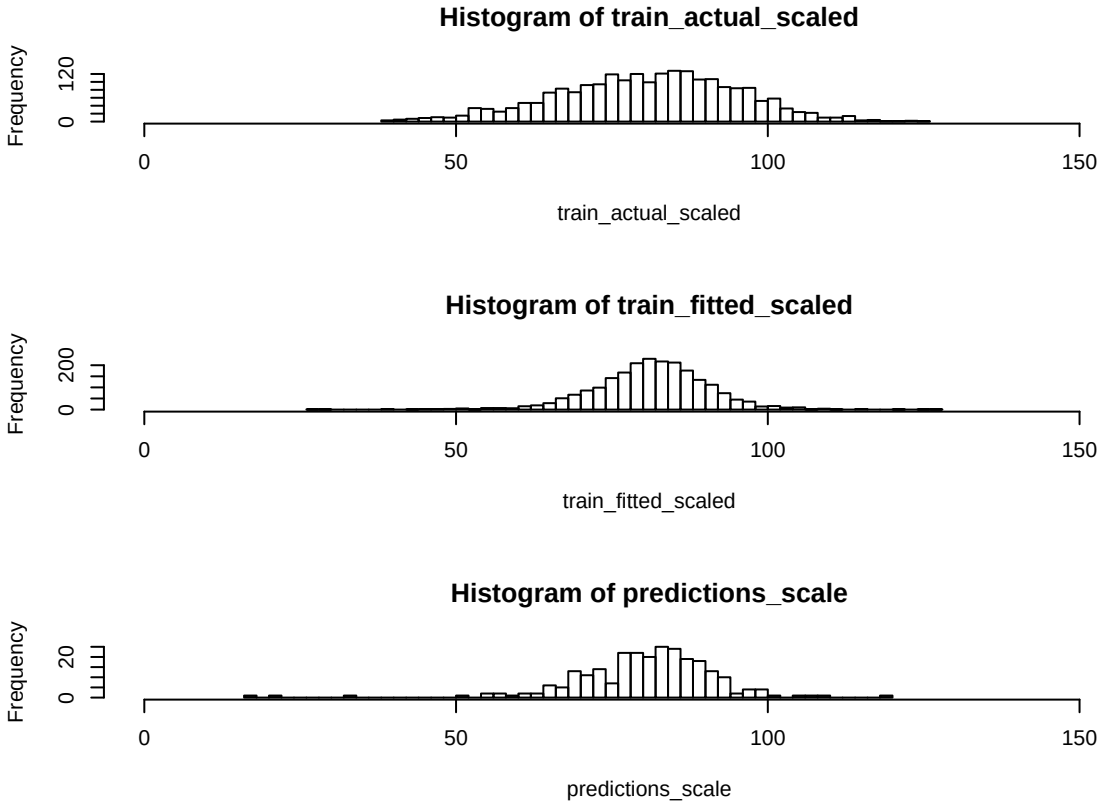
```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   38.00   72.00   82.00   81.15   91.00  126.00
```

```
## [1] "train_fitted_scaled"
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   27.26   76.17   81.57   81.30   86.94  127.50
```

```
## [1] "predictions_scale"
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##   17.38   74.24   81.46   80.35   87.17  118.70       2
```



Appendix

Conclusion

Despite all of my efforts and the great amount of time spent on this assignment (30+ hours), the model did not really improve. I might be missing something that would radically improve the model, but I suspect I am not. Instead, the real issue I believe is a lack of good data. There are lots of outliers, missing values, and missing variables that I know would be important but are not available (e.g., Year). With better data and a better understanding of how the raw data was transformed and changed into the current data set, I would hope that I could come up with a better model.

The use of a linear model may also be inappropriate, but not having much experience with other kinds of models, I am not really sure. As the course progresses, I hope I will know how to evaluate this, and perhaps will be able to judge that a different kind of model is more appropriate.

I am very interested to see other solutions to this assignment to understand what I may have missed and how my work could be improved.

Code

```
##code for Question 1 - Data Explorations
#part a
train <- read.csv("C:/Users/Andy/Desktop/Personal/Learning/CUNY/DATA621/moneyball-training-data.csv", s
test <- read.csv("C:/Users/Andy/Desktop/Personal/Learning/CUNY/DATA621/moneyball-evaluation-data.csv", s

#remove index
train$INDEX<-NULL
test_index <-test$INDEX
test$INDEX<-NULL

#summary
print("Summary of train")
summary(train)


#part b

# qqnorm(train$TARGET_WINS)

#boxplots
#boxplot(train$TEAM_BASERUN_SB)
#boxplot(train$ TEAM_BASERUN_CS)
par(mfrow=c(2,2))
boxplot(train$ TEAM_PITCHING_H, main="TEAM_PITCHING_H")
boxplot(train$ TEAM_PITCHING_BB, main="TEAM_PITCHING_BB")
boxplot(train$ TEAM_PITCHING_SO, main="TEAM_PITCHING_SO")
boxplot(train$ TEAM_FIELDING_E, main="TEAM_FIELDING_E")
#boxplot(train$ TEAM_BATTING_3B)

#histograms
par(mfrow=c(2,2))
hist(train$ TEAM_PITCHING_H, main="TEAM_PITCHING_H", breaks=50)
hist(train$ TEAM_PITCHING_BB, main="TEAM_PITCHING_BB", breaks=50)
hist(train$ TEAM_PITCHING_SO, main="TEAM_PITCHING_SO", breaks=50)
hist(train$ TEAM_FIELDING_E, main="TEAM_FIELDING_E", breaks=50)

# pairs(train)
correlationMatrix <- cor(train, use = "pairwise.complete.obs")

#TARGET_WINS vs. other variables
print("Correlation of Target Wins vs. Variables")
sort(correlationMatrix[1,], decreasing = TRUE)

# for (i in 2:length(train)){
#   plot(train[,i],train$TARGET_WINS, main=names(train)[i])
#   abline(lm(train$TARGET_WINS~train[,i]))
# }

plot(train$TEAM_BATTING_H,train$TARGET_WINS, main="TEAM_BATTING_H")
abline(lm(train$TARGET_WINS~train$TEAM_BATTING_H))
```

```r
#part c

#other variables
#correlationMatrix > 0.5 | correlationMatrix < -0.5
correlationMatrix_High<-correlationMatrix
correlationMatrix_High[correlationMatrix_High < 0.5 & correlationMatrix_High > -0.5]<- NA

#plots
plot(train$TEAM_BATTING_HR, train$TEAM_PITCHING_HR, main="Batting HR vs. Pitching HR")

#same variable?
# sum(train$TEAM_BATTING_HR == train$TEAM_PITCHING_HR)/length(train$TEAM_BATTING_HR)
# sum(abs(train$TEAM_BATTING_HR - train$TEAM_PITCHING_HR) <= 2)/length(train$TEAM_BATTING_HR)

##code for Question 2 - Data Preparation
#part b

# #baseline lm - 0.5116
baseline<-lm(train$TARGET_WINS~., data=train)
summary(baseline)

#create missing value flags variables-train
train$ TEAM_BATTING_SO_NA<-ifelse(is.na(train$ TEAM_BATTING_SO),1,0)
train$ TEAM_BASERUN_SB_NA<-ifelse(is.na(train$ TEAM_BASERUN_SB),1,0)
train$ TEAM_BASERUN_CS_NA<-ifelse(is.na(train$ TEAM_BASERUN_CS),1,0)
train$ TEAM_BATTING_HBP_NA<-ifelse(is.na(train$ TEAM_BATTING_HBP),1,0)
train$ TEAM_PITCHING_SO_NA<-ifelse(is.na(train$ TEAM_PITCHING_SO),1,0)
train$ TEAM_FIELDING_DP_NA<-ifelse(is.na(train$ TEAM_FIELDING_DP),1,0)

#create missing value flags variables-test
test$ TEAM_BATTING_SO_NA<-ifelse(is.na(test$ TEAM_BATTING_SO),1,0)
test$ TEAM_BASERUN_SB_NA<-ifelse(is.na(test$ TEAM_BASERUN_SB),1,0)
test$ TEAM_BASERUN_CS_NA<-ifelse(is.na(test$ TEAM_BASERUN_CS),1,0)
test$ TEAM_BATTING_HBP_NA<-ifelse(is.na(test$ TEAM_BATTING_HBP),1,0)
test$ TEAM_PITCHING_SO_NA<-ifelse(is.na(test$ TEAM_PITCHING_SO),1,0)
test$ TEAM_FIELDING_DP_NA<-ifelse(is.na(test$ TEAM_FIELDING_DP),1,0)

#part a
#summary(train)

#missing values
train$TEAM_BATTING_SO[train$TEAM_BATTING_SO==0]<-NA
test$TEAM_BATTING_SO[test$TEAM_BATTING_SO==0]<-NA

median1<-median(train[is.na(train$TEAM_BASERUN_CS),]$TEAM_BATTING_SO, na.rm=TRUE)
median2<-median(train[!is.na(train$TEAM_BASERUN_CS),]$TEAM_BATTING_SO, na.rm=TRUE)

train$TEAM_BATTING_SO[is.na(train$ TEAM_BATTING_SO)& is.na(train$TEAM_BASERUN_CS)] <- median1
train$TEAM_BATTING_SO[is.na(train$ TEAM_BATTING_SO) & !is.na(train$TEAM_BASERUN_CS)] <- median2

test$TEAM_BATTING_SO[is.na(test$ TEAM_BATTING_SO)& is.na(test$TEAM_BASERUN_CS)] <- median1
test$TEAM_BATTING_SO[is.na(test$ TEAM_BATTING_SO) & !is.na(test$TEAM_BASERUN_CS)] <- median2
```

```
# plot(train$TEAM_BATTING_SO, train$TARGET_WINS)
# abline(lm(train$TARGET_WINS~train$TEAM_BATTING_SO))
# sort(train$TEAM_BATTING_SO)

# hist(train[is.na(train$TEAM_BASERUN_CS),]$TEAM_BATTING_SO, breaks = 50)
# hist(train[!is.na(train$TEAM_BASERUN_CS),]$TEAM_BATTING_SO, breaks = 50)
# abline(v=735)
# abline(v=750)
# summary(train[is.na(train$TEAM_BASERUN_CS),]$TEAM_BATTING_SO)
# summary(train[!is.na(train$TEAM_BASERUN_CS),]$TEAM_BATTING_SO)

median1 <- median(train$TEAM_BASERUN_SB, na.rm=TRUE)
train$TEAM_BASERUN_SB[is.na(train$ TEAM_BASERUN_SB)] <- median1
test$TEAM_BASERUN_SB[is.na(test$ TEAM_BASERUN_SB)] <- median1

# hist(train$TEAM_BASERUN_SB, breaks = 50)
# abline(v=101)
# abline(v=124.8)

median1 <- median(train$TEAM_BASERUN_CS, na.rm=TRUE)
train$TEAM_BASERUN_CS[is.na(train$ TEAM_BASERUN_CS)] <- median1
test$TEAM_BASERUN_CS[is.na(test$ TEAM_BASERUN_CS)] <- median1

# hist(train$TEAM_BASERUN_CS, breaks = 50)
# abline(v=49.0)
# abline(v=52.8)

#train$TEAM_BATTING_HBP[is.na(train$ TEAM_BATTING_HBP)] <- median(train$TEAM_BATTING_HBP, na.rm=TRUE)
# hist(train$TEAM_BATTING_HBP, breaks = 50)

median1 <-median(train$TEAM_PITCHING_SO, na.rm=TRUE)
train$TEAM_PITCHING_SO[is.na(train$ TEAM_PITCHING_SO)] <- median1
test$TEAM_PITCHING_SO[is.na(test$ TEAM_PITCHING_SO)] <- median1
# hist(train$TEAM_PITCHING_SO, breaks = 50)

median1 <- median(train$TEAM_FIELDING_DP, na.rm=TRUE)
train$TEAM_FIELDING_DP[is.na(train$ TEAM_FIELDING_DP)] <- median1
test$TEAM_FIELDING_DP[is.na(test$ TEAM_FIELDING_DP)] <- median1
# hist(train$TEAM_FIELDING_DP, breaks = 50)

#removing HBP
train$TEAM_BATTING_HBP <- NULL
test$TEAM_BATTING_HBP <- NULL

# #baseline lm -  0.4075
baseline<-lm(train$TARGET_WINS~., data=train)
summary(baseline)

#part c
#summary(train)

#bucket-train
train$TEAM_PITCHING_H_Bin <- cut(train$TEAM_PITCHING_H, breaks = quantile(train$TEAM_PITCHING_H))
```

```r
train$TEAM_PITCHING_BB_Bin <- cut(train$TEAM_PITCHING_BB, breaks = quantile(train$TEAM_PITCHING_BB))
train$TEAM_PITCHING_SO_Bin <- cut(train$TEAM_PITCHING_SO, breaks = quantile(train$TEAM_PITCHING_SO))

#bucket-test
test$TEAM_PITCHING_H_Bin <- cut(test$TEAM_PITCHING_H, breaks = quantile(train$TEAM_PITCHING_H))
test$TEAM_PITCHING_BB_Bin <- cut(test$TEAM_PITCHING_BB, breaks = quantile(train$TEAM_PITCHING_BB))
test$TEAM_PITCHING_SO_Bin <- cut(test$TEAM_PITCHING_SO, breaks = quantile(train$TEAM_PITCHING_SO))

#part f

#train_saved <-train
# train <-train_saved

#remove outliers
#names(train[,2:16])

#"TEAM_BATTING_H"
  # temp_model<-lm(TARGET_WINS~TEAM_BATTING_H, data=train)

  # plot(temp_model)
  # summary(temp_model)

  #1825, 1828, 1210, 1811, 417
  train<-
    train[!(rownames(train) %in% c(1825,1828,1210,1811,417)),]


#"TEAM_BATTING_2B"
  # temp_model<-lm(TARGET_WINS~TEAM_BATTING_2B, data=train)

  # plot(temp_model)
  # summary(temp_model)

  #2233, 1211, 299
  train<-
     train[!(rownames(train) %in% c(1211, 2233, 299)),]

#"TEAM_BATTING_3B"
  # temp_model<-lm(TARGET_WINS~TEAM_BATTING_3B, data=train)

  # plot(temp_model)
  # summary(temp_model)

  #416,
  train<-
     train[!(rownames(train) %in% c(416)),]

#"TEAM_BATTING_HR" -looks fine

#"TEAM_BATTING_BB"
  # temp_model<-lm(TARGET_WINS~TEAM_BATTING_BB, data=train)

  # plot(temp_model)
```

```
  # summary(temp_model)

  #296, 422, 2012,
  train<-
     train[!(rownames(train) %in% c(296, 422, 2012)),]

#"TEAM_BATTING_SO"
  # temp_model<-lm(TARGET_WINS~TEAM_BATTING_SO, data=train)

  # plot(temp_model)
  # summary(temp_model)

  #862, 859, 982
  train<-
     train[!(rownames(train) %in% c(862, 859, 982)),]


#"TEAM_BASERUN_SB"
  # temp_model<-lm(TARGET_WINS~TEAM_BASERUN_SB, data=train)

  # plot(temp_model)
  # summary(temp_model)

  #415, 1708, 2022
  train<-
     train[!(rownames(train) %in% c(415, 1708, 2022)),]

#"TEAM_BASERUN_CS"
  # temp_model<-lm(TARGET_WINS~TEAM_BASERUN_CS, data=train)

  # plot(temp_model)
  # summary(temp_model)

  #1345, 2239, 2276, 1503
  train<-
     train[!(rownames(train) %in% c(1345, 2239, 2276, 1503)),]


#"TEAM_PITCHING_H"
  # temp_model<-lm(TARGET_WINS~TEAM_PITCHING_H, data=train)

  # plot(train$TEAM_PITCHING_H, train$TARGET_WINS)
  # abline(temp_model)

  # plot(temp_model)
  # summary(temp_model)

  #1584, 2136, 2015, 1083, 2232, 1
  train<-
     train[!(rownames(train) %in% c(1584, 2136, 2015, 1083, 2232, 1, 391, 1346, 2219, 1342, 2220, 53)),]


#"TEAM_PITCHING_HR"
```

```
  train$TEAM_PITCHING_HR<-NULL
  test$TEAM_PITCHING_HR<-NULL

#"TEAM_PITCHING_BB"
  # temp_model<-lm(TARGET_WINS~TEAM_PITCHING_BB, data=train)

  # plot(temp_model)
  # summary(temp_model)

  #282, 418, 1082, 1340, 1810
  train<-
     train[!(rownames(train) %in% c(282, 418, 1082, 1340, 1810)),]

#"TEAM_PITCHING_SO"
  # temp_model<-lm(TARGET_WINS~TEAM_PITCHING_SO, data=train)

  # plot(temp_model)
  # summary(temp_model)

  #1826, 1698
  train<-
     train[!(rownames(train) %in% c(1826, 1698)),]

#"TEAM_FIELDING_E"
  # temp_model<-lm(TARGET_WINS~TEAM_FIELDING_E, data=train)

  # plot(temp_model)
  # summary(temp_model)

  #998, 297, 298
  train<-
     train[!(rownames(train) %in% c(998, 297, 298)),]

#"TEAM_FIELDING_DP"
  # temp_model<-lm(TARGET_WINS~TEAM_FIELDING_DP, data=train)

  # plot(temp_model)
  # summary(temp_model)

  #996, 1199,2242,2031,
  train<-
     train[!(rownames(train) %in% c(996, 1199,2242,2031)),]

## check baseline - 0.3891
# baseline<-lm(train$TARGET_WINS~., data=train)
# summary(baseline)
#
# plot(baseline)

# 295, 1822, 394, 392, 394, 294, 1347, 1348, 272, 273
train<-
     train[!(rownames(train) %in% c(295, 1822, 394, 392, 394, 294, 1347, 1348, 272, 273)),]
```

```r
#try again - 0.4119
baseline<-lm(train$TARGET_WINS~., data=train)
summary(baseline)

par(mfrow=c(2,2))
plot(baseline)



#single base hits
train$TEAM_BATTING_1B <- train$TEAM_BATTING_H - train$TEAM_BATTING_2B - train$TEAM_BATTING_3B - train$TE
test$TEAM_BATTING_1B <- test$TEAM_BATTING_H - test$TEAM_BATTING_2B - test$TEAM_BATTING_3B - test$TEAM_BA

#total on base (hits + walks + HBP).  Can't use HBP since so many NAs
train$TEAM_BATTING_OnBase <- train$TEAM_BATTING_H + train$TEAM_BATTING_BB #+ train$TEAM_BATTING_HBP
test$TEAM_BATTING_OnBase <- test$TEAM_BATTING_H + test$TEAM_BATTING_BB #+ test$TEAM_BATTING_HBP

#at bats - hits, walks, HBP, strikeouts (don't know outs, but there would be three).  Can't use HBP.
train$TEAM_BATTING_AtBat <- train$TEAM_BATTING_H + train$TEAM_BATTING_BB + train$TEAM_BATTING_SO #+ tra
test$TEAM_BATTING_AtBat <- test$TEAM_BATTING_H + test$TEAM_BATTING_BB + test$TEAM_BATTING_SO #+ test$TE

#onbase-%.  onbase/atbats
train$TEAM_BATTING_OnBase_PERC <- train$TEAM_BATTING_OnBase/train$TEAM_BATTING_AtBat
test$TEAM_BATTING_OnBase_PERC <- test$TEAM_BATTING_OnBase/test$TEAM_BATTING_AtBat

#pitching-onbase.  Hits + BB
train$TEAM_PITCHING_OnBase <- train$TEAM_PITCHING_H + train$TEAM_PITCHING_BB
test$TEAM_PITCHING_OnBase <- test$TEAM_PITCHING_H + test$TEAM_PITCHING_BB

#pitching-onbase %.  Onbase / (onbase + SO)
train$TEAM_PITCHING_OnBase_PERC <- train$TEAM_PITCHING_OnBase / (train$TEAM_PITCHING_OnBase + train$TEA
test$TEAM_PITCHING_OnBase_PERC <- test$TEAM_PITCHING_OnBase / (test$TEAM_PITCHING_OnBase + test$TEAM_PI

#slugging
train$TEAM_BATTING_SLUGGING <- train$TEAM_BATTING_1B + 2*train$TEAM_BATTING_2B + 3* train$TEAM_BATTING_
test$TEAM_BATTING_SLUGGING <- test$TEAM_BATTING_1B + 2*test$TEAM_BATTING_2B + 3* test$TEAM_BATTING_3B +

#slugging_perc
train$TEAM_BATTING_SLUGGING_PERC <- train$TEAM_BATTING_SLUGGING / train$TEAM_BATTING_AtBat
test$TEAM_BATTING_SLUGGING_PERC <- test$TEAM_BATTING_SLUGGING / test$TEAM_BATTING_AtBat

#on base + slugging
train$TEAM_BATTING_OnBaseSlugging <- train$TEAM_BATTING_OnBase + train$TEAM_BATTING_SLUGGING
test$TEAM_BATTING_OnBaseSlugging <- test$TEAM_BATTING_OnBase + test$TEAM_BATTING_SLUGGING

# 1397
train<-
     train[!(rownames(train) %in% c(1397)),]

#try again - 0.4142
baseline<-lm(train$TARGET_WINS~., data=train)
# summary(baseline)
```

```r
#residuals
par(mfrow=c(2,2))
plot(baseline)


#train_saved <- train
#train <- train_saved

library(MASS)
library(dplyr)

#TARGET_WINS value- boxcox
bc <- boxcox(baseline, plotit = FALSE)
bestLambda_Y<-bc$x[which.max(bc$y)]

# qqnorm((train$TARGET_WINS)^(1.2))
# qqline((train$TARGET_WINS)^(1.2))

train$TARGET_WINS <- (train$TARGET_WINS)^(bestLambda_Y)
#test$TARGET_WINS <- (test$TARGET_WINS)^(bestLambda_Y)

#try again - adjr2 = 0.4133
baseline<-lm(train$TARGET_WINS~., data=train)
# summary(baseline) #f-stat = 52.78

par(mfrow=c(2,2))
plot(baseline)

par(mfrow=c(1,1))
#fit vs. actual
plot(baseline$fitted.values, baseline$model$`train$TARGET_WINS`, xlim=c(0,350), ylim=c(0,350))
lines(0:350,0:350)

#train_saved <- train
#train <- train_saved

# X values
library(alr3)

invTranCalculate<-function(variable, lambda){
  if(lambda!=0){
    return((variable^lambda-1)/lambda)
  } else if(lambda==0){
    log(variable)
  }

}


#"TEAM_BATTING_H" - -1.504504
#invTranPlot(train$TEAM_BATTING_H, train$TARGET_WINS)
lambda <-invTranEstimate(train$TEAM_BATTING_H, train$TARGET_WINS)$lambda
train$TEAM_BATTING_H<-
```

```
  invTranCalculate(train$TEAM_BATTING_H,lambda)
test$TEAM_BATTING_H<-
  invTranCalculate(test$TEAM_BATTING_H,lambda)
#plot(train$TEAM_BATTING_H, train$TARGET_WINS)

#"TEAM_BATTING_2B": -1.82, 0.4162
#invTranPlot(train$TEAM_BATTING_2B, train$TARGET_WINS)
lambda <- invTranEstimate(train$TEAM_BATTING_2B, train$TARGET_WINS)$lambda
train$TEAM_BATTING_2B<-
  invTranCalculate(train$TEAM_BATTING_2B,lambda)
test$TEAM_BATTING_2B<-
  invTranCalculate(test$TEAM_BATTING_2B,lambda)
# plot(train$TEAM_BATTING_2B, train$TARGET_WINS)

#"TEAM_BATTING_3B": 2.68, 0.4179
#invTranPlot(train$TEAM_BATTING_3B, train$TARGET_WINS)
lambda<-invTranEstimate(train$TEAM_BATTING_3B, train$TARGET_WINS)$lambda
train$TEAM_BATTING_3B<-
  invTranCalculate(train$TEAM_BATTING_3B,lambda)
test$TEAM_BATTING_3B<-
  invTranCalculate(test$TEAM_BATTING_3B,lambda)
#plot(train$TEAM_BATTING_3B, train$TARGET_WINS)

#"TEAM_BATTING_HR": 2.8, 0.4093
train$TEAM_BATTING_HR<- train$TEAM_BATTING_HR + 1 #add 1 to make variable all positive
test$TEAM_BATTING_HR<- test$TEAM_BATTING_HR + 1 #add 1 to make variable all positive
#invTranPlot(train$TEAM_BATTING_HR, train$TARGET_WINS)
lambda<-invTranEstimate(train$TEAM_BATTING_HR, train$TARGET_WINS)$lambda
train$TEAM_BATTING_HR<-
  invTranCalculate(train$TEAM_BATTING_HR,lambda)
test$TEAM_BATTING_HR<-
  invTranCalculate(test$TEAM_BATTING_HR,lambda)
#plot(train$TEAM_BATTING_HR, train$TARGET_WINS) #note: not constant variance.  Need to address

#"TEAM_BATTING_BB" : 3.93, 0.4056
#invTranPlot(train$TEAM_BATTING_BB, train$TARGET_WINS)
lambda<-invTranEstimate(train$TEAM_BATTING_BB, train$TARGET_WINS)$lambda
train$TEAM_BATTING_BB<-
  invTranCalculate(train$TEAM_BATTING_BB,lambda)
test$TEAM_BATTING_BB<-
  invTranCalculate(test$TEAM_BATTING_BB,lambda)
#plot(train$TEAM_BATTING_BB, train$TARGET_WINS)

#"TEAM_BATTING_SO": -2.54, 0.4056
#invTranPlot(train$TEAM_BATTING_SO, train$TARGET_WINS)
lambda <- invTranEstimate(train$TEAM_BATTING_SO, train$TARGET_WINS)$lambda
train$TEAM_BATTING_SO<-
  invTranCalculate(train$TEAM_BATTING_SO,lambda)
test$TEAM_BATTING_SO<-
  invTranCalculate(test$TEAM_BATTING_SO,lambda)
#plot(train$TEAM_BATTING_SO, train$TARGET_WINS)

#"TEAM_BASERUN_SB"  -0.4053
```

```
#invTranPlot(train$TEAM_BASERUN_SB, train$TARGET_WINS)
lambda <- invTranEstimate(train$TEAM_BASERUN_SB, train$TARGET_WINS)$lambda
train$TEAM_BASERUN_SB<-
  invTranCalculate(train$TEAM_BASERUN_SB,lambda)
test$TEAM_BASERUN_SB<-
  invTranCalculate(test$TEAM_BASERUN_SB,lambda)
#plot(train$TEAM_BASERUN_SB, train$TARGET_WINS)

#"TEAM_BASERUN_CS": no need to adjust

#"TEAM_PITCHING_H" : -2.02, 0.4053
#invTranPlot(train$TEAM_PITCHING_H, train$TARGET_WINS)
lambda <- invTranEstimate(train$TEAM_PITCHING_H, train$TARGET_WINS)$lambda
train$TEAM_PITCHING_H<-
  invTranCalculate(train$TEAM_PITCHING_H,lambda)
test$TEAM_PITCHING_H<-
  invTranCalculate(test$TEAM_PITCHING_H,lambda)
#plot(train$TEAM_PITCHING_H, train$TARGET_WINS)

#"TEAM_PITCHING_BB": 1.28, 0.4057
#invTranPlot(train$TEAM_PITCHING_BB, train$TARGET_WINS)
lambda <- invTranEstimate(train$TEAM_PITCHING_BB, train$TARGET_WINS)$lambda
train$TEAM_PITCHING_BB<-
  invTranCalculate(train$TEAM_PITCHING_BB,lambda)
test$TEAM_PITCHING_BB<-
  invTranCalculate(test$TEAM_PITCHING_BB,lambda)
#plot(train$TEAM_PITCHING_BB, train$TARGET_WINS)

#"TEAM_PITCHING_SO": 0.62, 0.4046
train$TEAM_PITCHING_SO <- train$TEAM_PITCHING_SO + 1 #make strictly positive
test$TEAM_PITCHING_SO <- test$TEAM_PITCHING_SO + 1 #make strictly positive
#invTranPlot(train$TEAM_PITCHING_SO, train$TARGET_WINS)
lambda <- invTranEstimate(train$TEAM_PITCHING_SO, train$TARGET_WINS)$lambda
train$TEAM_PITCHING_SO<-
  invTranCalculate(train$TEAM_PITCHING_SO,lambda)
test$TEAM_PITCHING_SO<-
  invTranCalculate(test$TEAM_PITCHING_SO,lambda)
#plot(train$TEAM_PITCHING_SO, train$TARGET_WINS)

#"TEAM_FIELDING_E": -2.99, 0.3174 - investigate this.  Greatly lowers R^2.  don't transform.
#invTranPlot(train$TEAM_FIELDING_E, train$TARGET_WINS)
# train$TEAM_FIELDING_E<-
#   invTranCalculate(train$TEAM_FIELDING_E,invTranEstimate(train$TEAM_FIELDING_E, train$TARGET_WINS)$la
# #plot(train$TEAM_FIELDING_E, train$TARGET_WINS) #not constant variance.  Need to do something about t

#"TEAM_FIELDING_DP": no need to transform

#"TEAM_BATTING_1B": 1.99, 0.4053
#invTranPlot(train$TEAM_BATTING_1B, train$TARGET_WINS)
lambda <- invTranEstimate(train$TEAM_BATTING_1B, train$TARGET_WINS)$lambda
train$TEAM_BATTING_1B<-
  invTranCalculate(train$TEAM_BATTING_1B,lambda)
test$TEAM_BATTING_1B<-
```

```
  invTranCalculate(test$TEAM_BATTING_1B,lambda)
#plot(train$TEAM_BATTING_1B, train$TARGET_WINS)

#"TEAM_BATTING_OnBase", 0.416
#invTranPlot(train$TEAM_BATTING_OnBase, train$TARGET_WINS)
lambda <- invTranEstimate(train$TEAM_BATTING_OnBase, train$TARGET_WINS)$lambda
train$TEAM_BATTING_OnBase<-
  invTranCalculate(train$TEAM_BATTING_OnBase,lambda)
test$TEAM_BATTING_OnBase<-
  invTranCalculate(test$TEAM_BATTING_OnBase,lambda)
#plot(train$TEAM_BATTING_OnBase, train$TARGET_WINS)

#"TEAM_BATTING_AtBat": -1.84, 0.4097
#invTranPlot(train$TEAM_BATTING_AtBat, train$TARGET_WINS)
lambda <- invTranEstimate(train$TEAM_BATTING_AtBat, train$TARGET_WINS)$lambda
train$TEAM_BATTING_AtBat<-
  invTranCalculate(train$TEAM_BATTING_AtBat,lambda)
test$TEAM_BATTING_AtBat<-
  invTranCalculate(test$TEAM_BATTING_AtBat,lambda)
#plot(train$TEAM_BATTING_AtBat, train$TARGET_WINS)

#"TEAM_BATTING_OnBase_PERC": no transformation needed

#"TEAM_PITCHING_OnBase" 0.409
#invTranPlot(train$TEAM_PITCHING_OnBase, train$TARGET_WINS)
lambda <- invTranEstimate(train$TEAM_PITCHING_OnBase, train$TARGET_WINS)$lambda
train$TEAM_PITCHING_OnBase<-
  invTranCalculate(train$TEAM_PITCHING_OnBase,lambda)
test$TEAM_PITCHING_OnBase<-
  invTranCalculate(test$TEAM_PITCHING_OnBase,lambda)
#plot(train$TEAM_PITCHING_OnBase, train$TARGET_WINS)

#"TEAM_PITCHING_OnBase_PERC": -9.68, 0.4098
#invTranPlot(train$TEAM_PITCHING_OnBase_PERC, train$TARGET_WINS)
lambda <- invTranEstimate(train$TEAM_PITCHING_OnBase_PERC, train$TARGET_WINS)$lambda
train$TEAM_PITCHING_OnBase_PERC<-
  invTranCalculate(train$TEAM_PITCHING_OnBase_PERC,lambda)
test$TEAM_PITCHING_OnBase_PERC<-
  invTranCalculate(test$TEAM_PITCHING_OnBase_PERC,lambda)
#plot(train$TEAM_PITCHING_OnBase_PERC, train$TARGET_WINS) #note: not constant variance.  need to do som

#"TEAM_BATTING_SLUGGING", 0.4093
#invTranPlot(train$TEAM_BATTING_SLUGGING, train$TARGET_WINS)
lambda <- invTranEstimate(train$TEAM_BATTING_SLUGGING, train$TARGET_WINS)$lambda
train$TEAM_BATTING_SLUGGING<-
  invTranCalculate(train$TEAM_BATTING_SLUGGING,lambda)
test$TEAM_BATTING_SLUGGING<-
  invTranCalculate(test$TEAM_BATTING_SLUGGING,lambda)
#plot(train$TEAM_BATTING_SLUGGING, train$TARGET_WINS)

#"TEAM_BATTING_SLUGGING_PERC", 0.4087
#invTranPlot(train$TEAM_BATTING_SLUGGING_PERC, train$TARGET_WINS)
lambda <-invTranEstimate(train$TEAM_BATTING_SLUGGING_PERC, train$TARGET_WINS)$lambda
```

```
train$TEAM_BATTING_SLUGGING_PERC<-
  invTranCalculate(train$TEAM_BATTING_SLUGGING_PERC,lambda)
test$TEAM_BATTING_SLUGGING_PERC<-
  invTranCalculate(test$TEAM_BATTING_SLUGGING_PERC,lambda)
#plot(train$TEAM_BATTING_SLUGGING_PERC, train$TARGET_WINS)

#"TEAM_BATTING_OnBaseSlugging": 0.68, 0.4092
#invTranPlot(train$TEAM_BATTING_OnBaseSlugging, train$TARGET_WINS)
lambda <- invTranEstimate(train$TEAM_BATTING_OnBaseSlugging, train$TARGET_WINS)$lambda
train$TEAM_BATTING_OnBaseSlugging<-
  invTranCalculate(train$TEAM_BATTING_OnBaseSlugging,lambda)
test$TEAM_BATTING_OnBaseSlugging<-
  invTranCalculate(test$TEAM_BATTING_OnBaseSlugging,lambda)
#plot(train$TEAM_BATTING_OnBaseSlugging, train$TARGET_WINS)

# 1604
train<-
     train[!(rownames(train) %in% c(1604)),]

#try again - 0.41
baseline<-lm(train$TARGET_WINS~., data=train)
# summary(baseline)

#residuals
par(mfrow=c(2,2))
plot(baseline)

#plot
par(mfrow=c(1,1))
plot(baseline$fitted.values, baseline$model$`train$TARGET_WINS`, xlim=c(0,350), ylim=c(0,350))
lines(0:350,0:350)

# summary(train)
# model 1
fit<-lm(TARGET_WINS~., data=train)
summary(fit) #adjR2-0.41

#plots
plot(fit$fitted.values,fit$model$TARGET_WINS, xlim=c(0,350), ylim=c(0,350))
lines(1:350,1:350)

#residuals
par(mfrow=c(2,2))
plot(fit)


#train_saved<-train
#train<-train_saved

#model 2
model2 <-lm(formula = TARGET_WINS ~ ., data = train)
model2 <- update(model2, as.formula(paste0(". ~ . -","
                                    TEAM_PITCHING_H_Bin -
```

```
                                    TEAM_PITCHING_BB_Bin -
                                    TEAM_PITCHING_SO_Bin
                                    ")))

model2<-stepAIC(model2, direction = "backward", trace=FALSE)

#summary
summary(model2) #0.4066

#plots
plot(model2$fitted.values, model2$model$TARGET_WINS, xlim=c(0,350), ylim=c(0,350))
lines(1:350,1:350)

#residuals
par(mfrow=c(2,2))
plot(model2)

#model 3
model3 <-lm(formula = TARGET_WINS ~ TEAM_BATTING_H , data = train)
model3<-stepAIC(model3, direction = "forward", scope=list(lower=~1, upper=~TEAM_BATTING_H + TEAM_BATTIN

#as.formula(paste(names(train)[c(2:20,24:32)], collapse = " + "))

summary(model3) #0.3985

#plots
plot(model3$fitted.values,model3$model$TARGET_WINS, xlim=c(0,350), ylim=c(0,350))
lines(1:350,1:350)

#residuals
par(mfrow=c(2,2))
plot(model3)

#model4 - remove multicollinearity
model4 <-lm(formula = TARGET_WINS ~ ., data = train)
model4 <- update(model4, as.formula(paste0(". ~ . -","
                                    TEAM_PITCHING_H_Bin -
                                    TEAM_PITCHING_BB_Bin -
                                    TEAM_PITCHING_SO_Bin
                                    ")))
notNA<-36

while(notNA > 2){
  train2 <- train[,names(train) %in% names(model4$coefficients)]
  correlationMatrix2_High<-cor(train2)
  correlationMatrix2_High[correlationMatrix2_High < 0.6 & correlationMatrix2_High > -0.6]<- NA
  correlationMatrix2_High[correlationMatrix2_High == 1]<- NA
  notNA <-sum(!is.na(correlationMatrix2_High))

  variables<-dimnames(which(correlationMatrix2_High==max(correlationMatrix2_High, na.rm = TRUE), arr.in
  var1pvalue<-summary(model4)$coefficients[which(rownames(summary(model4)$coefficients)==variables[1]),
  var2pvalue<-summary(model4)$coefficients[which(rownames(summary(model4)$coefficients)==variables[2]),
```

```
  if(length(var1pvalue)==0){
    model4 <- update(model4, as.formula(paste0(". ~ . - ", variables[1])))
  } else if (length(var2pvalue)==0){
     model4 <- update(model4, as.formula(paste0(". ~ . - ", variables[2])))
  } else if (var1pvalue > var2pvalue){
     model4 <- update(model4, as.formula(paste0(". ~ . - ", variables[1])))
  } else{
    model4 <- update(model4, as.formula(paste0(". ~ . - ", variables[2])))
  }


} #while

#summary: 0.2833
summary(model4)

#plots
plot(model4$fitted.values, model4$model$TARGET_WINS, xlim=c(0,350), ylim=c(0,350))
lines(1:350,1:350)

#residuals
par(mfrow=c(2,2))
plot(model4)


#summary
# summary(model2)

#predict
predictions<-predict(model2, newdata=test)
# summary(predictions)

#adjust predictions to proper scale -
predictions_scale <-predictions^(1/bestLambda_Y)

#scale training data
train_fitted_scaled<-(model2$fitted.values)^(1/bestLambda_Y)
train_actual_scaled <-(model2$model$TARGET_WINS)^(1/bestLambda_Y)

#plots-adjusted to proper scale
plot(train_fitted_scaled,train_actual_scaled, xlim=c(0,162), ylim=c(0,162), main="Fitted vs. Actual Tar
lines(0:162,0:162)

#compare summaries
print ("train_actual_scaled")
summary(train_actual_scaled )
print ("train_fitted_scaled")
summary(train_fitted_scaled)
print ("predictions_scale")
summary(predictions_scale)

#histogram comparisons
par(mfrow=c(3,1))
```

```r
hist(train_actual_scaled, breaks=50, xlim=c(0,162))
hist(train_fitted_scaled, breaks=50, xlim=c(0,162))
hist(predictions_scale, breaks=50, xlim=c(0,162))

#output predictions
predictions_scale[is.na(predictions_scale)] <- 0
predictions_output<-cbind(test_index, predictions_scale)
# write.csv(predictions_output,"C:/Users/Andy/Desktop/Personal/Learning/CUNY/DATA621/moneyball-predicti
```