Moneyball

Jorge Fernandes

MSDS 411

24 April 2018

```
library(e1071) # to understand skewness
library(dplyr)
library(stringr) # Used to rename the columns by removing the word team
from the column header
library(VIM) # To understand NAs
library(caret)

## Warning in as.POSIXlt.POSIXct(Sys.time()): unknown timezone
'zone/tz/2018c.
## 1.0/zoneinfo/America/New_York'

library(mice) # Imputation
library(missForest) # Imputation
library(MASS) # to use for robust Linear Regression.

# browse to the data
moneyball = read.csv('/Users/legs_jorge/Documents/Data Science
Projects/MSDS_Northwestern/MSDS 411/Unit 01 Moneyball Baseball
Problem/Data/moneyball.csv', header = T)
colnames(moneyball) <- str_replace_all(colnames(moneyball),"TEAM_","")
%>%
  tolower() # Fixing column names
```

Introduction

The moneyball dataset has sparked many companies, teams, and organizations to understand and utilize the data they generate/gather. This project highlights many pitfalls that those same individuals fall into simply because they forgot to do the due diligence and prepare the data before modeling.
This paper will focus on;
1. Data Exploration
2. Data Transformation
3. Model Building
4. How to select the best model

Data Exploration

Step 1: Are there lots of NAs in the data?

R gives us a lot of ways to understand the distribution of `Nulls` within the data. Let's first try to calculate the percentage of Null values to the total number of observation.

```
NAPerc <-
  sapply(moneyball, function(x)
    (sum(is.na(x)) / length(x)) * 100) %>%
  data.frame()
NAPerc$Column <- rownames(NAPerc)
colnames(NAPerc) <- c("NA_Perc", "Col_Name")

# Trying to understand the percentage of NAs per Column
NA_col <- subset(NAPerc, NA_Perc > 0) %>% arrange(desc(NA_Perc))
NA_col
```

```
##     NA_Perc    Col_Name
## 1 91.608084 batting_hbp
## 2 33.919156  baserun_cs
## 3 12.565905 fielding_dp
## 4  5.755712  baserun_sb
## 5  4.481547  batting_so
## 6  4.481547 pitching_so
```

Let's look at the pattern of missing data to try to get more insights. It's clear that batting_hbp is going to be a problematic column with 92% of the data missing. Before we start the imputation or deleting variables, let's try to understand why we have missing data.

Let's use the `mice` package to help us understant how all the NAs behave in the data. `mice` provides a handy function called `md.pattern` that allows one to understand the pattern of missing data. Hopefully by looking at the pattern, we can have an idea on why the data could be missing.

```
md.pattern(moneyball) %>% data.frame()
```

The **first column** of the output shows the number of unique missing data patterns. There are 191 observations with nonmissing values, and there are 1295 observations with nonmissing values except for the variable batting_hbp. The **rightmost column** shows the number of *missing variables* in a particular missing pattern. For example, the first row has no missing value and it is "0" in the row. The **last row** counts the number of missing values for each variable. For example, the variable pitching_bb contains no missing values and the variable batting_so contains 102 missing values. This table can be helpful when you decide to drop some observations with missing variables exceeding a preset threshold.

After careful analysis, the decision is to keep `batting_hbp`. Because I want to transform it into a binary variable, and will keep it out until all the imputation is done.

```
batting_hbp_bi <- if_else(is.na(moneyball$batting_hbp),0,1)
batting_hbp <- moneyball$batting_hbp
moneyball_trans <- subset(moneyball, select = -c(batting_hbp))
```

Let's impute and treat the data for missing values before testing it for multicollinearity.

The `missForest` package will be the package used to help us with this task. `missForest` is an implementation of random forest algorithm. It's a non parametric imputation method applicable to various variable types. A great resource to understand this techinique is found here.

Let's add `batting_hbp` back into the data.

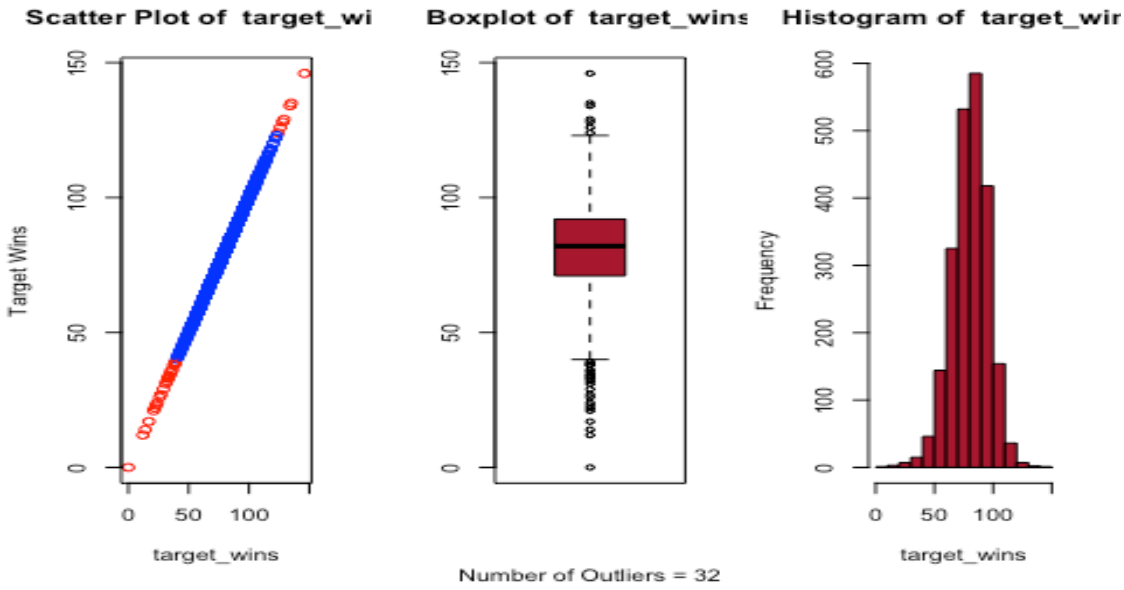
```
moneyball_MF$batting_hbp <-
if_else(is.na(batting_hbp),0,as.numeric(batting_hbp))
moneyball_MF$batting_hbp_bi <- batting_hbp_bi
```

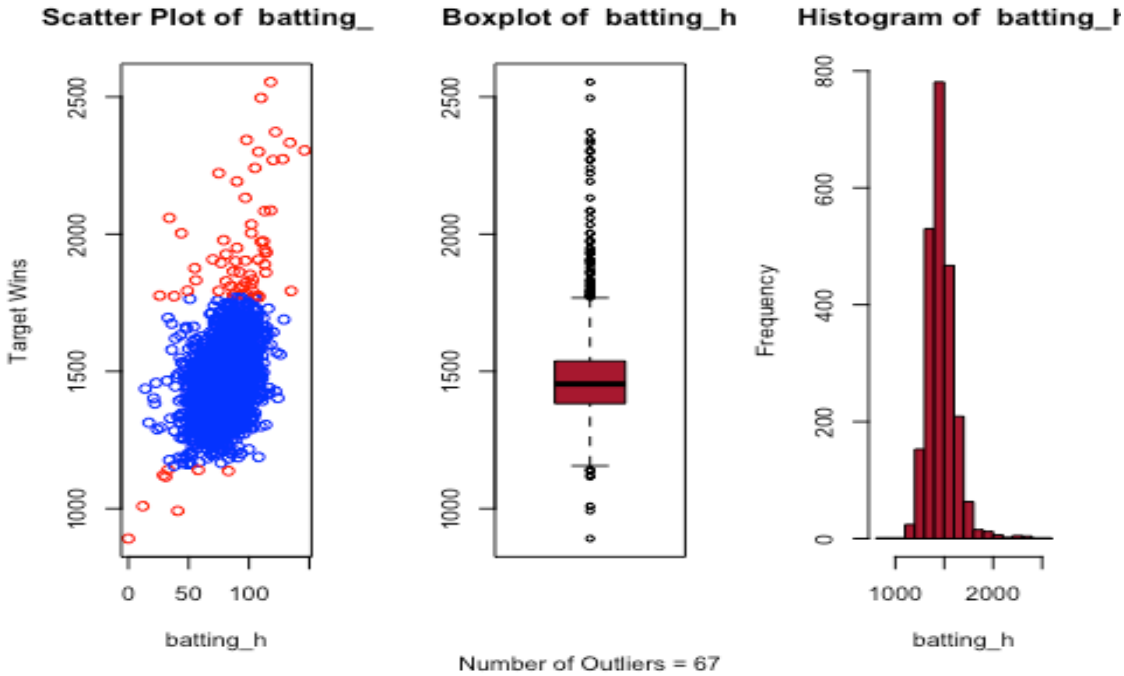
Step 2: Can we find outliers in our Independent and Dependent variables?

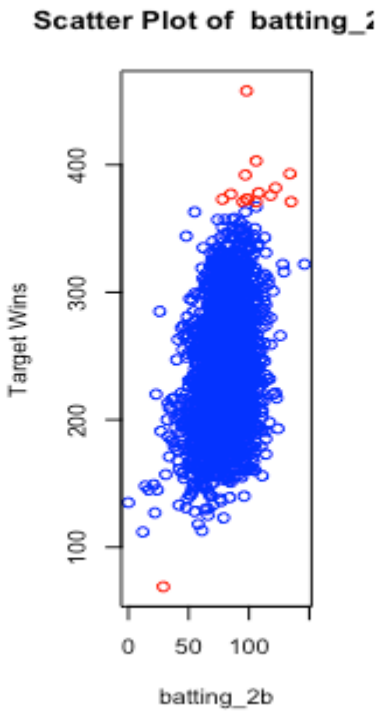
Outliers can cause our model to produce the wrong output by influencing its fit. Creating boxplots will aid in identifying those outliers. We can also use the cleveland dotplot to understand the outliers better. This technique uses the row number against actual value to quickly point out any patterns of outliers. This plot will easilly allow us to check the raw data for errors such as typos during the data collection phase. Points on the far right side, or on the far left side, are observed values that are considerably larger, or smaller, than the majority of the observations, and require further investigation. When we use this chart, together with the box plot and histogram, we can easily identify patterns at to where in the data we're seeing outliers.

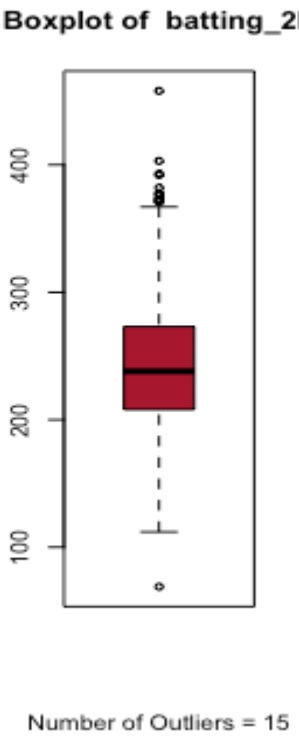
```
par(mfrow = c(1, 3))
i = 2
while (i %in% c(2:17)) {
out.lier <- boxplot.stats(moneyball_MF[,i])$out
plot(moneyball_MF$target_wins,
moneyball_MF[,i],col=ifelse(moneyball_MF[,i] %in% out.lier, "red",
"blue"),  xlab = colnames(moneyball_MF)[i] , ylab = "Target Wins", main
= paste("Scatter Plot of ",colnames(moneyball_MF)[i]))

boxplot(moneyball_MF[,i], col = "#A71930", main = paste("Boxplot of
",colnames(moneyball_MF)[i]))

title(sub = paste0("Number of Outliers = ",
length(boxplot.stats(moneyball_MF[,i])$out)))
```

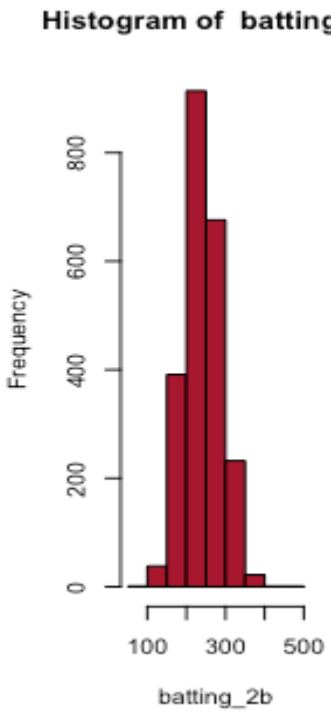
```
hist(
  moneyball_MF[,i],
  col = "#A71930",
  xlab = colnames(moneyball_MF)[i],
  main = paste("Histogram of ",colnames(moneyball_MF)[i])
)
  i = i + 1
}
```

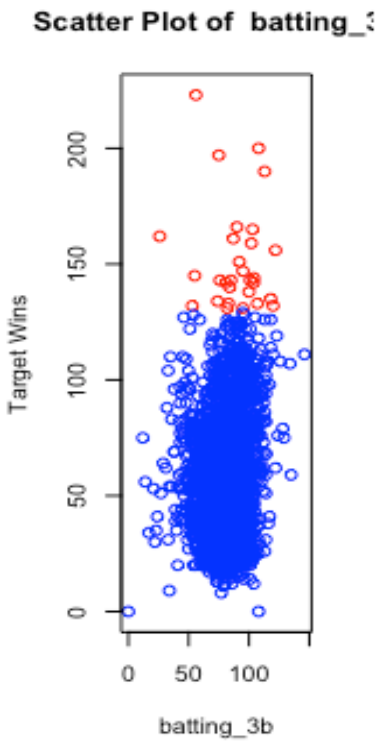


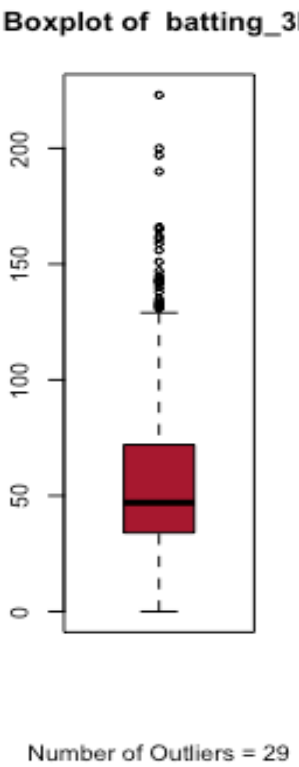


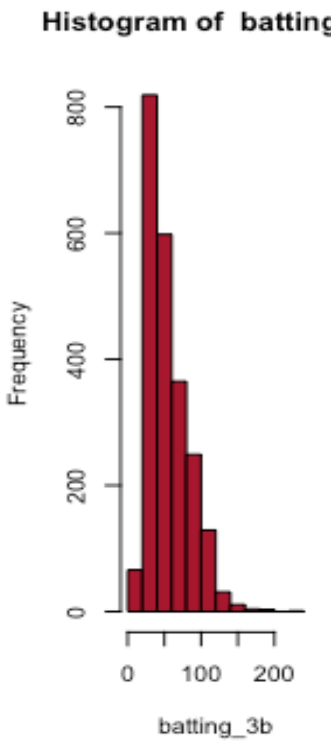

Scatter Plot of batting_2
Target Wins
100
200
300
400
0
50
100
batting_2b

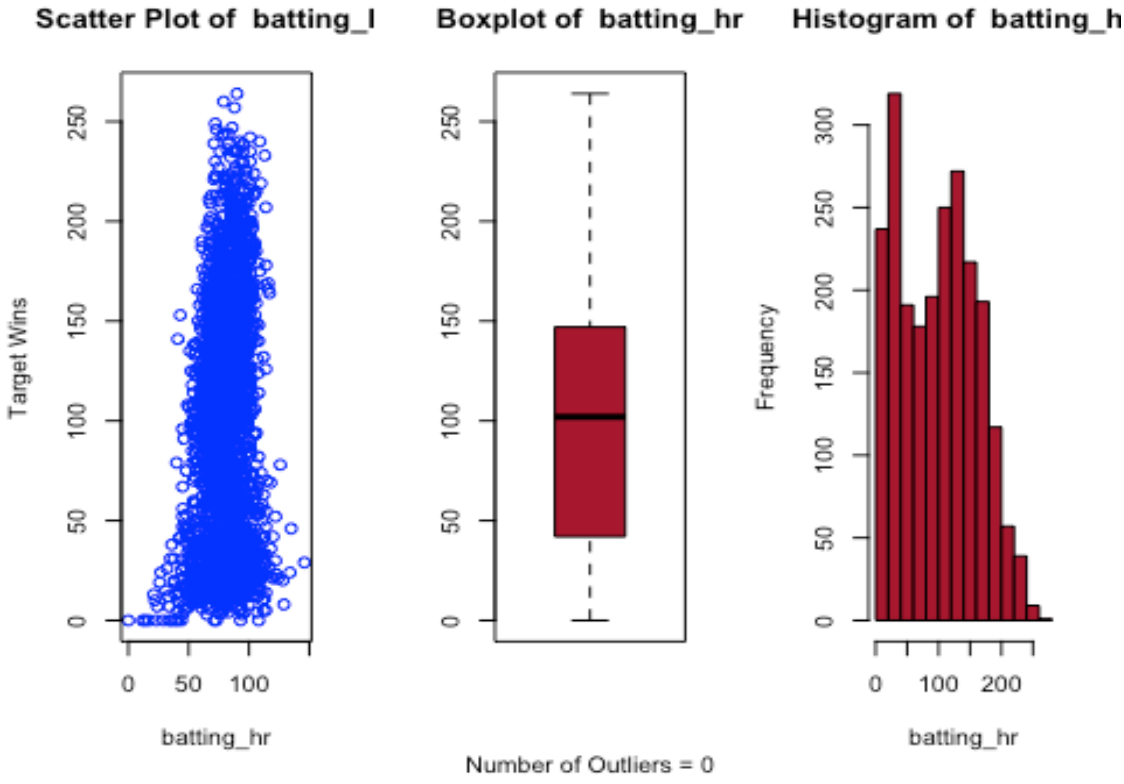

Boxplot of batting_2b
100
200
300
400
Number of Outliers = 15

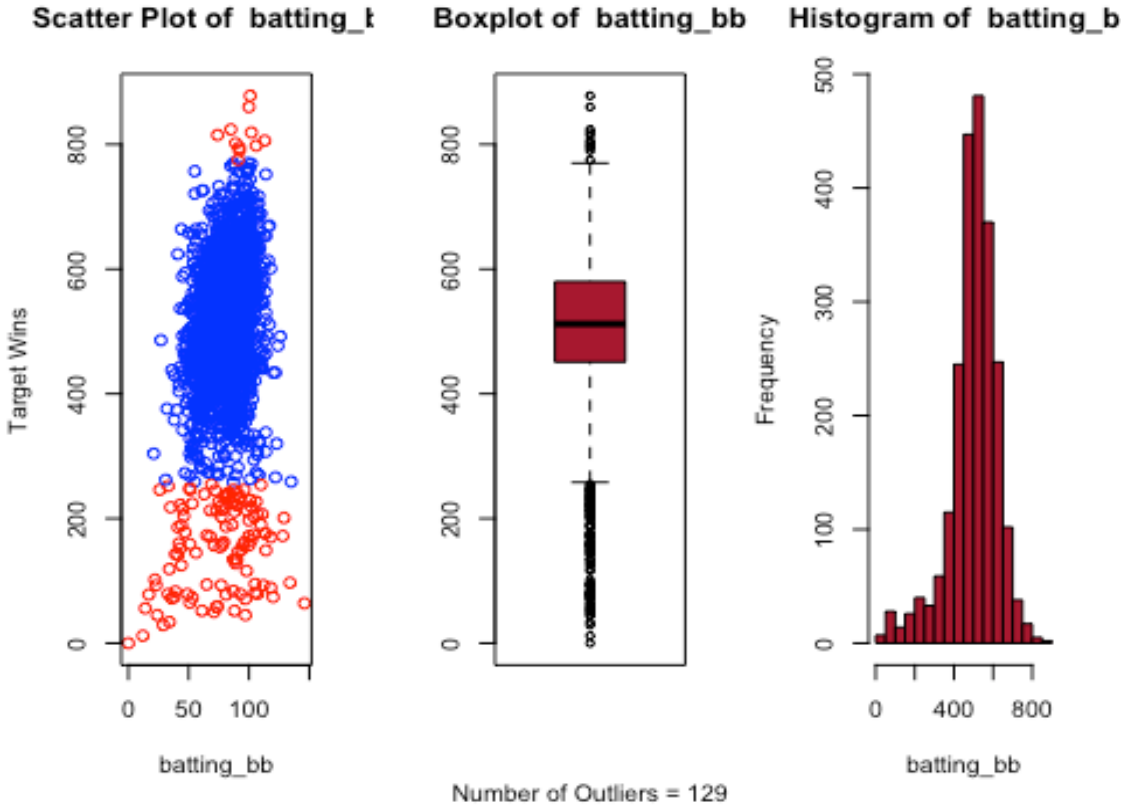

Histogram of batting_2
Frequency
0
200
400
600
800
100
300
500
batting_2b

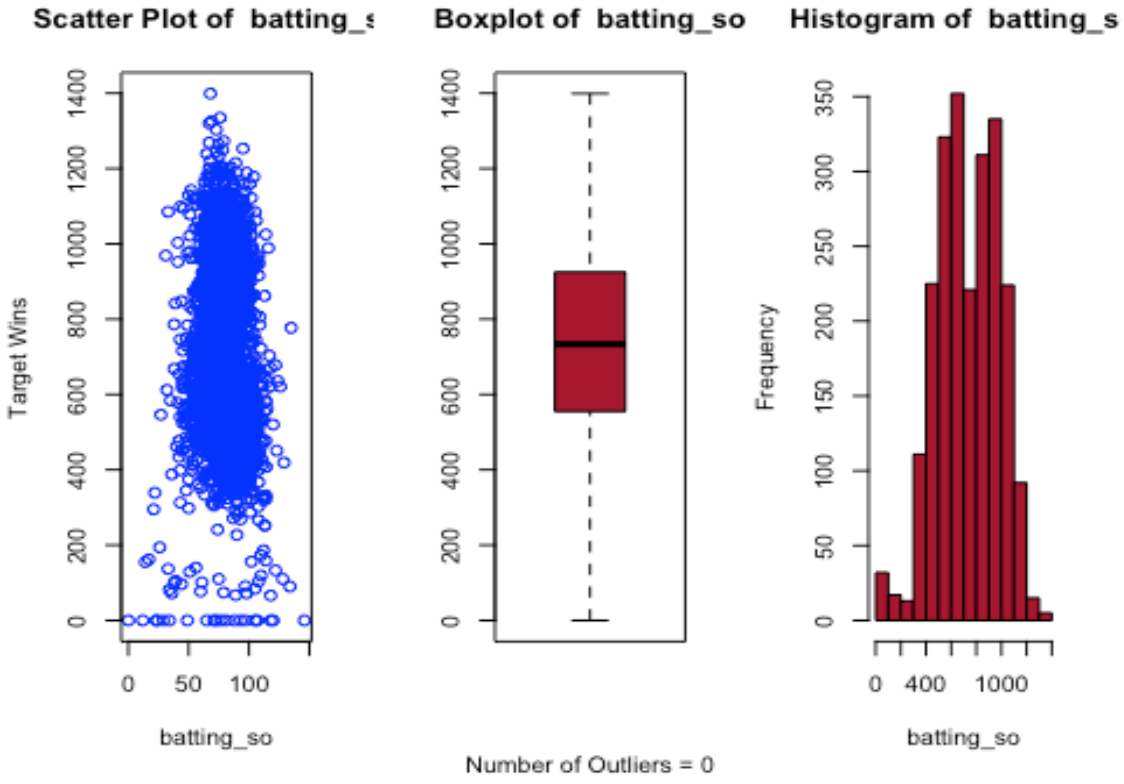

Scatter Plot of batting_3
Target Wins
0
50
100
150
200
0
50
100
batting_3b

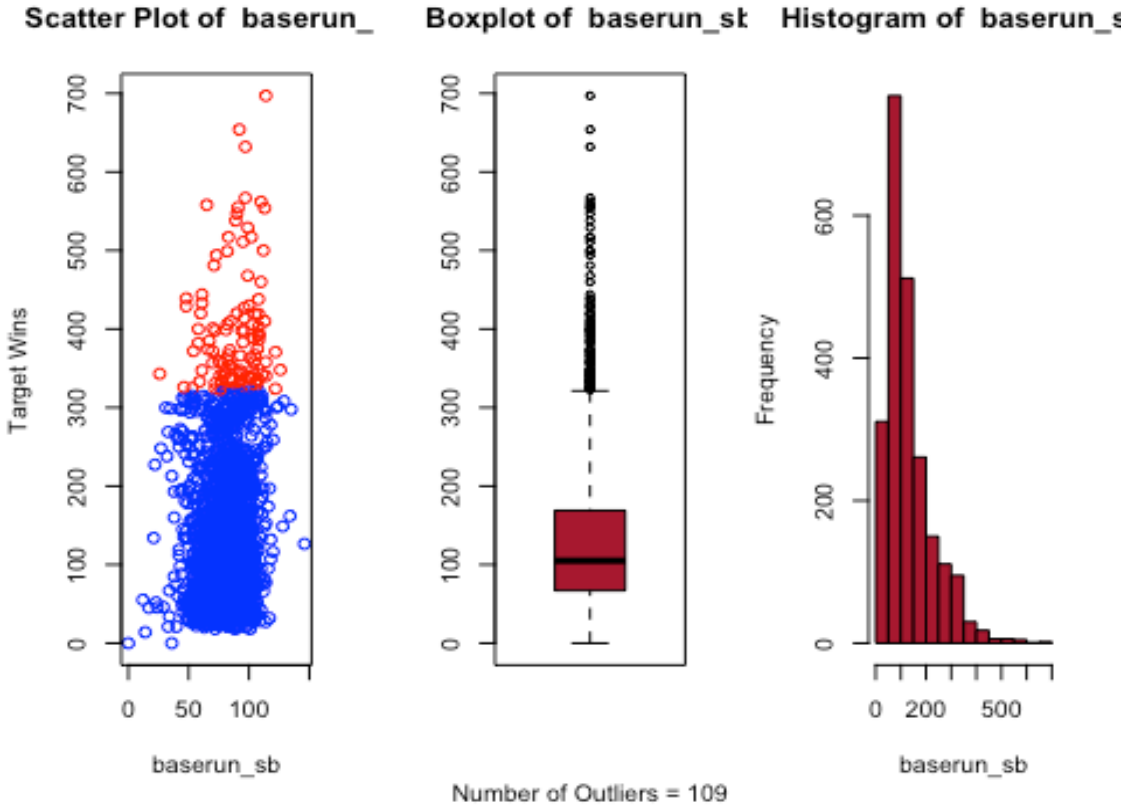

Boxplot of batting_3b
0
50
100
150
200
Number of Outliers = 29

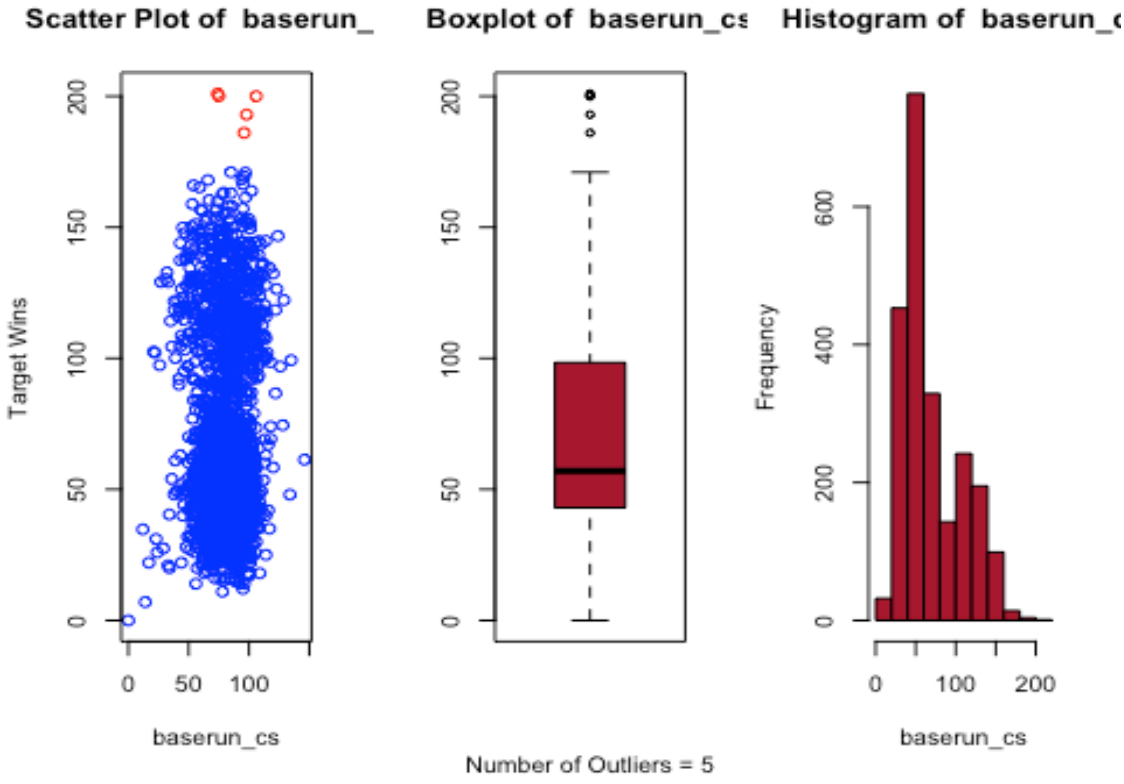

Histogram of batting_3
Frequency
0
200
400
600
800
0
100
200
batting_3b

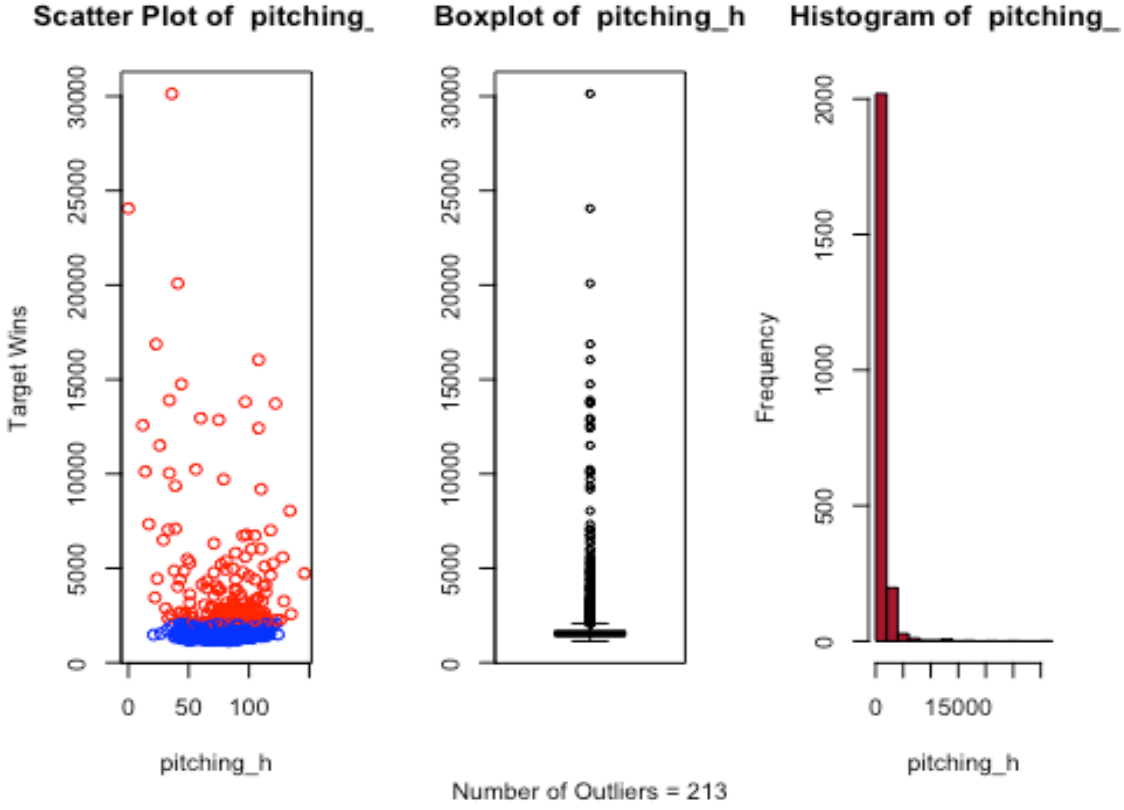

Scatter Plot of batting_l
Boxplot of batting_hr
Histogram of batting_h
Target Wins
batting_hr
Frequency
batting_hr
Number of Outliers = 0

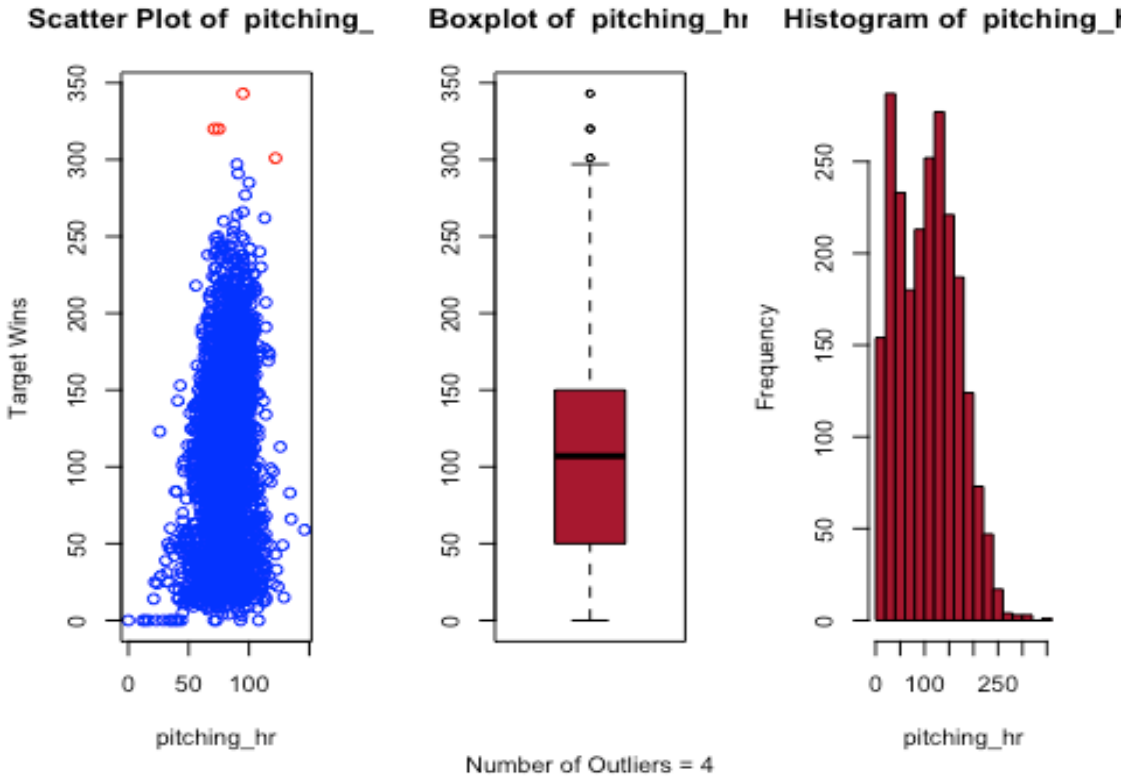

Scatter Plot of batting_b
Boxplot of batting_bb
Histogram of batting_b
Target Wins
batting_bb
Frequency
batting_bb
Number of Outliers = 129

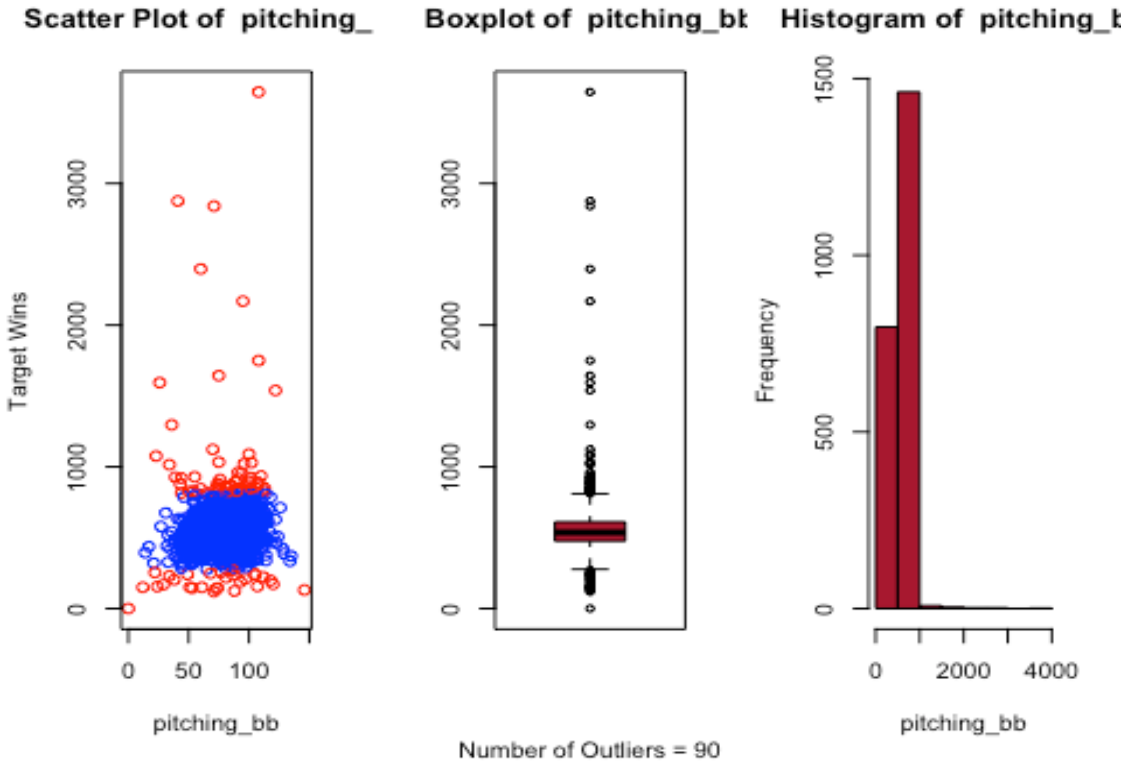

Scatter Plot of batting_s
Boxplot of batting_so
Histogram of batting_s
Target Wins
0
200
400
600
800
1000
1200
1400
0
50
100
batting_so
Number of Outliers = 0
Frequency
0
50
100
150
200
250
300
350
0
400
1000
batting_so

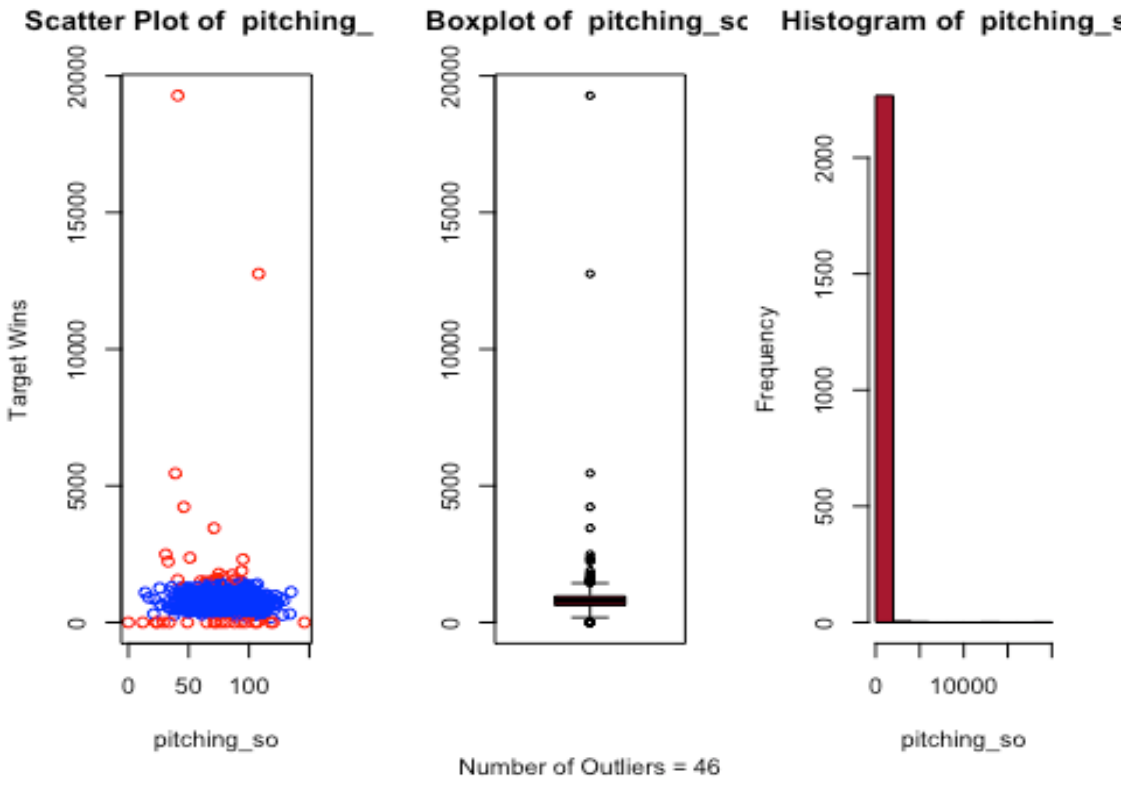

Scatter Plot of baserun_
Boxplot of baserun_st
Histogram of baserun_s
Target Wins
0
100
200
300
400
500
600
700
0
50
100
baserun_sb
Number of Outliers = 109
Frequency
0
200
400
600
0
200
500
baserun_sb

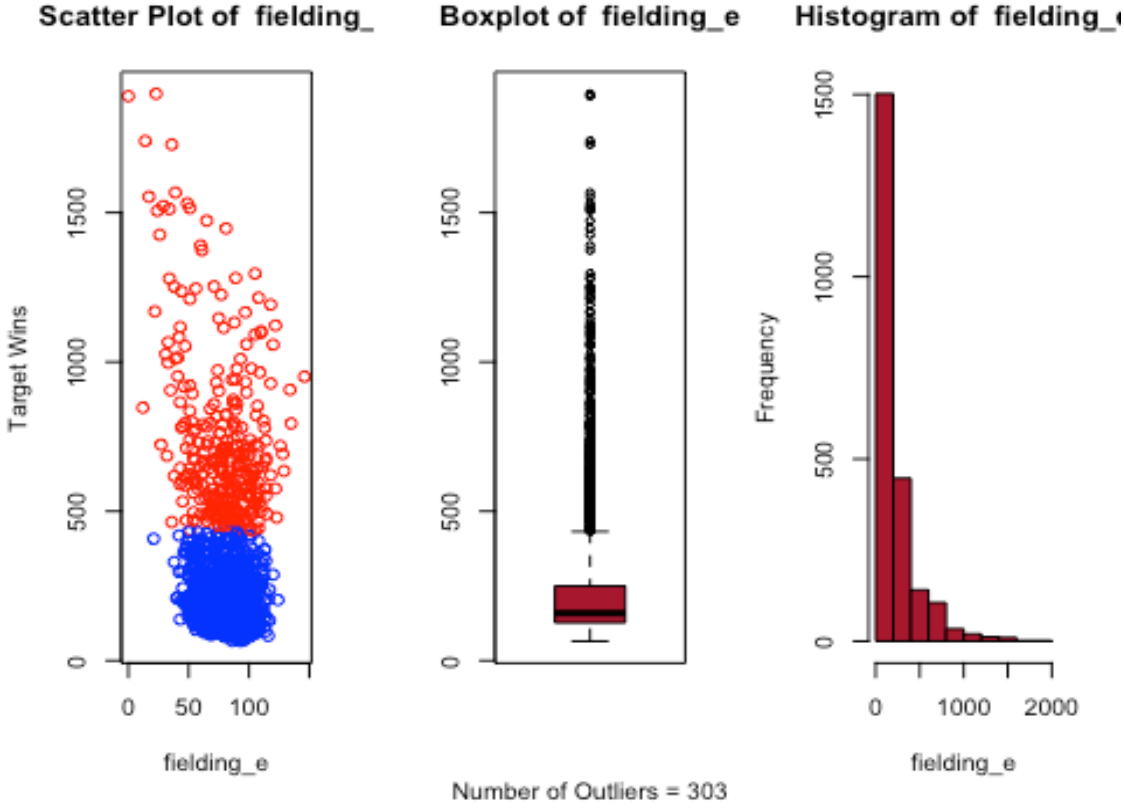

Scatter Plot of baserun_
Boxplot of baserun_cs
Histogram of baserun_c
Target Wins
baserun_cs
Number of Outliers = 5
Frequency
baserun_cs

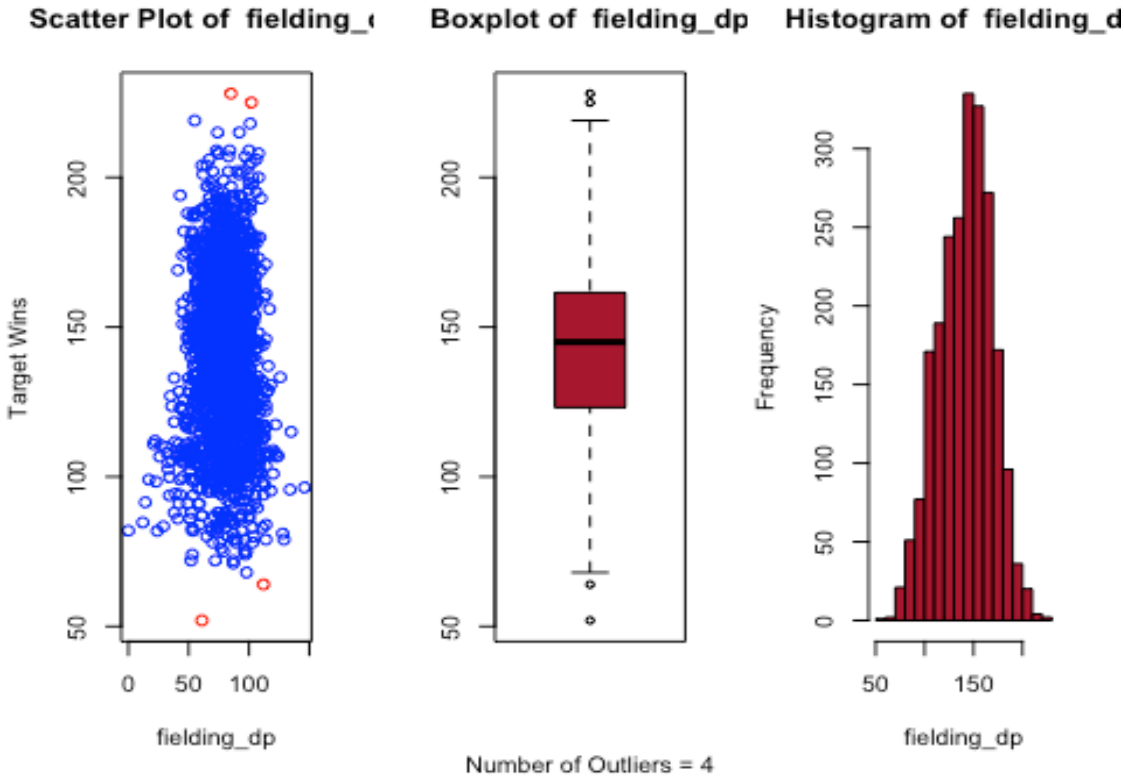

Scatter Plot of pitching_
Boxplot of pitching_h
Histogram of pitching_
Target Wins
pitching_h
Number of Outliers = 213
Frequency
pitching_h

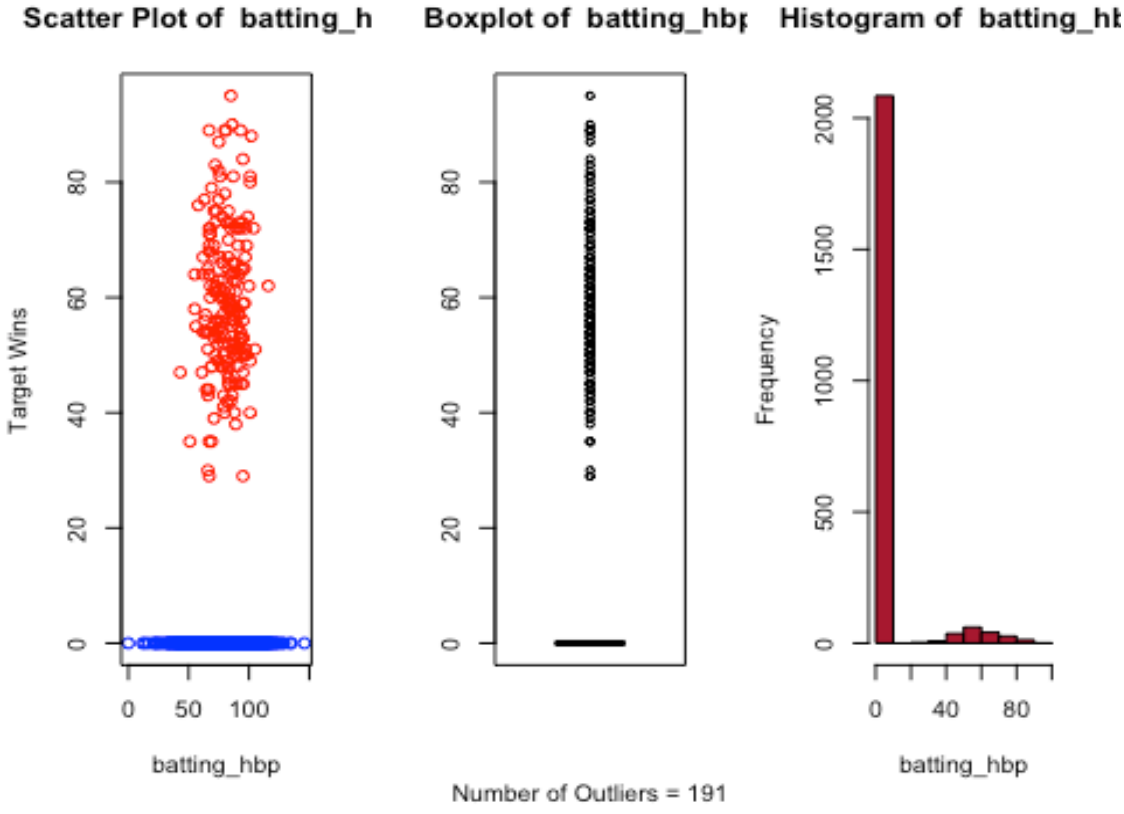

Scatter Plot of pitching_
Boxplot of pitching_hr
Histogram of pitching_h
Target Wins
pitching_hr
Frequency
pitching_hr
Number of Outliers = 4


Scatter Plot of pitching_
Boxplot of pitching_bb
Histogram of pitching_b
Target Wins
pitching_bb
Frequency
pitching_bb
Number of Outliers = 90


Scatter Plot of pitching_
Target Wins
pitching_so
Boxplot of pitching_sc
Number of Outliers = 46
Histogram of pitching_s
Frequency
pitching_so


Scatter Plot of fielding_
Target Wins
fielding_e
Boxplot of fielding_e
Number of Outliers = 303
Histogram of fielding_
Frequency
fielding_e


Scatter Plot of fielding_
Boxplot of fielding_dp
Histogram of fielding_d
Target Wins
fielding_dp
Frequency
fielding_dp
Number of Outliers = 4


Scatter Plot of batting_h
Boxplot of batting_hbp
Histogram of batting_hb
Target Wins
batting_hbp
Frequency
batting_hbp
Number of Outliers = 191

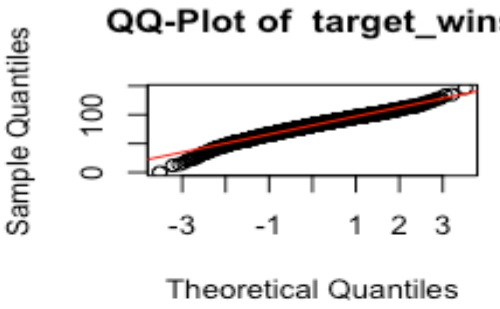
It looks like the outliers are going to be a problem for this model. Multiple techniques will be used to remediate this issue.

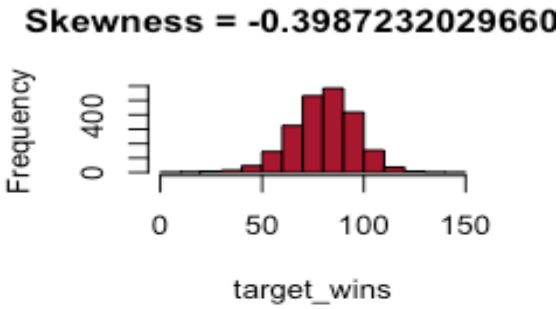
Now that step one is done, let's take a look at step 2.

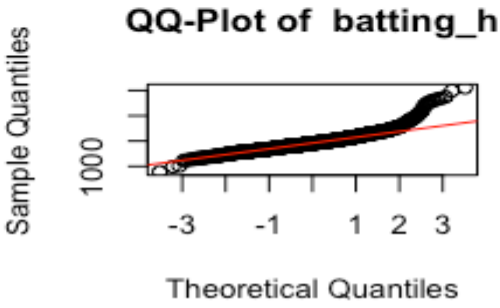
Step 3: Are the data normally distributed?

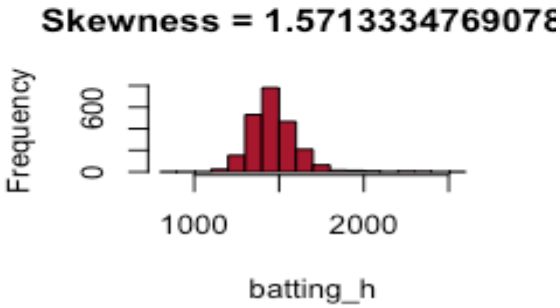
From the histogram above, we clearly see the data is not normal, with the exception of some that seems to sort of follow a normal distribution. Let's use QQ-plot to test each column for normality, while adding a histogram and a Skewness number.
- If skewness is less than −1 or greater than +1, the distribution is highly skewed.
- If skewness is between −1 and −½ or between +½ and +1, the distribution is moderately skewed.
- If skewness is between −½ and +½, the distribution is approximately symmetric.

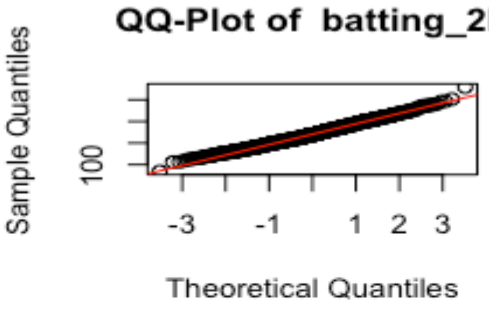
```
par(mfrow = c(2, 2))
i = 2
while (i %in% c(2:18)) {
  qqnorm(moneyball_MF[,i], main = paste("QQ-Plot of
",colnames(moneyball_MF)[i]));qqline(moneyball_MF[,i], col = 2)

  hist(
  moneyball_MF[,i],
  col = "#A71930",
  xlab = colnames(moneyball_MF)[i],
  main = paste0("Skewness = ",skewness(moneyball_MF[,i]))
)

  i = i + 1

}
```


QQ-Plot of target_wins
Sample Quantiles
0 100
-3 -1 1 2 3
Theoretical Quantiles

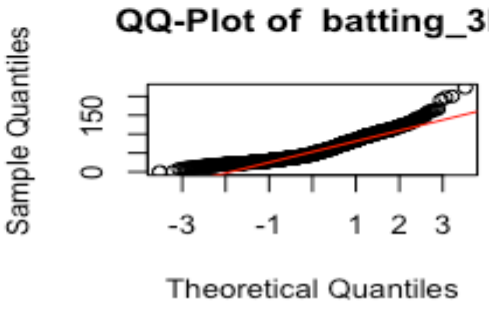

Skewness = -0.3987232029660
Frequency
0 400
0 50 100 150
target_wins

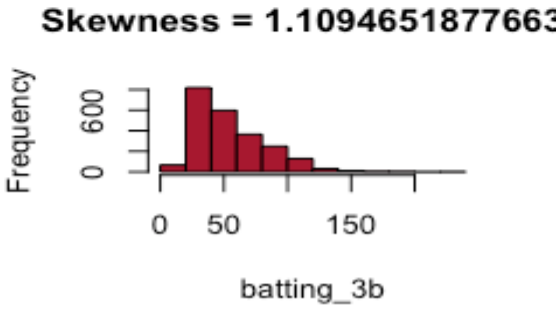

QQ-Plot of batting_h
Sample Quantiles
1000
-3 -1 1 2 3
Theoretical Quantiles

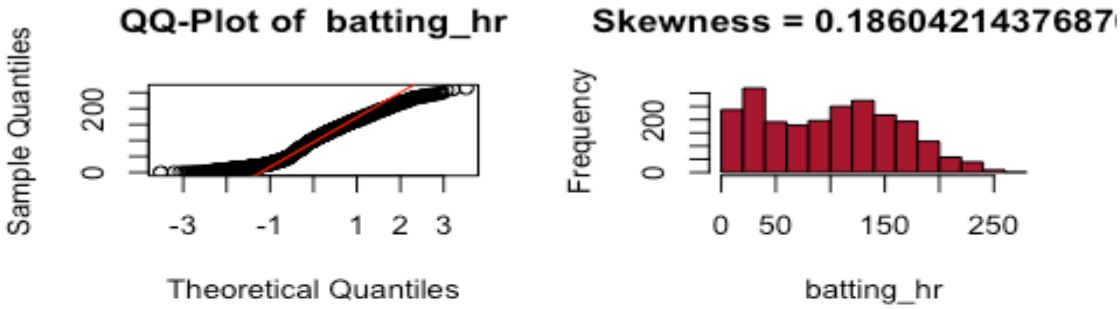

Skewness = 1.5713334769078
Frequency
0 600
1000 2000
batting_h


QQ-Plot of batting_2b
Sample Quantiles
100
-3 -1 1 2 3
Theoretical Quantiles

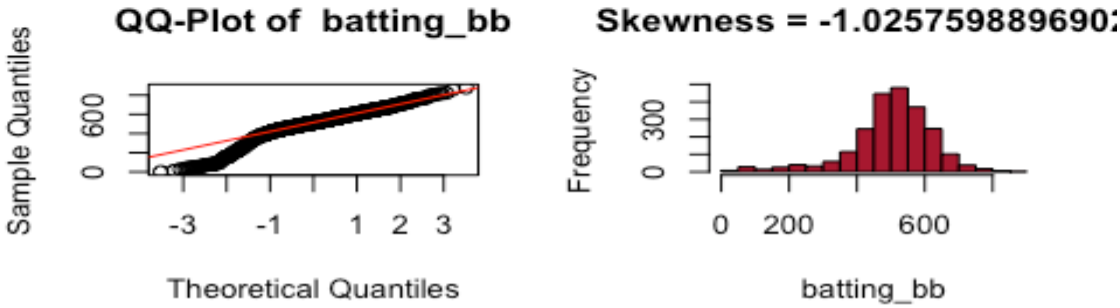

Skewness = 0.2151018023286
Frequency
0 600
100 300 500
batting_2b


QQ-Plot of batting_3b
Sample Quantiles
0 150
-3 -1 1 2 3
Theoretical Quantiles

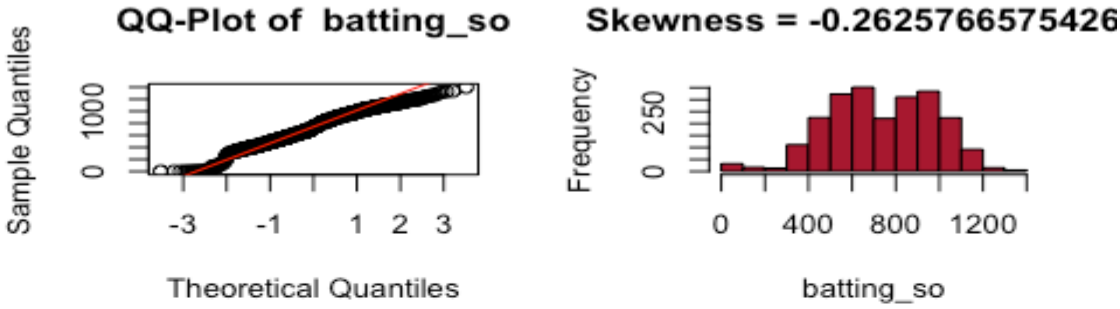

Skewness = 1.1094651877663
Frequency
0 600
0 50 150
batting_3b


QQ-Plot of batting_hr
Sample Quantiles
0 200
-3 -1 1 2 3
Theoretical Quantiles

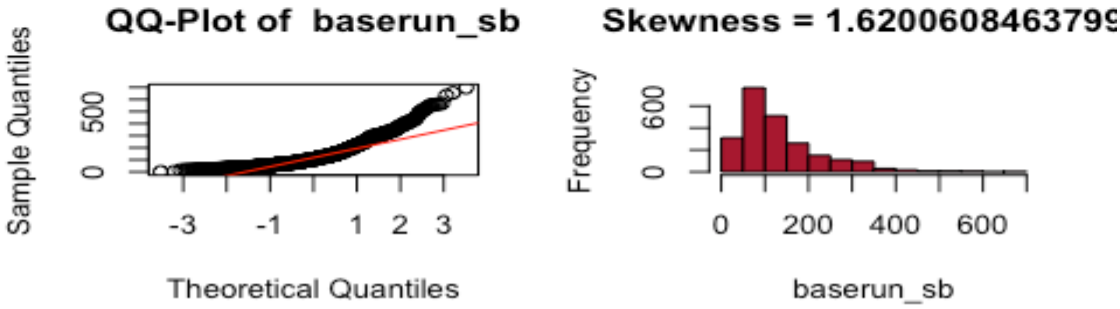
Skewness = 0.1860421437687
Frequency
0 200
0 50 150 250
batting_hr


QQ-Plot of batting_bb
Sample Quantiles
0 600
-3 -1 1 2 3
Theoretical Quantiles

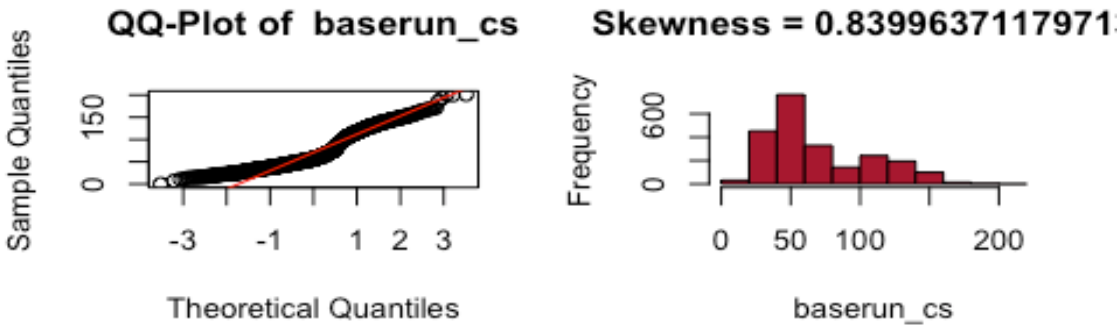
Skewness = -1.025759889690
Frequency
0 300
0 200 600
batting_bb


QQ-Plot of batting_so
Sample Quantiles
0 1000
-3 -1 1 2 3
Theoretical Quantiles

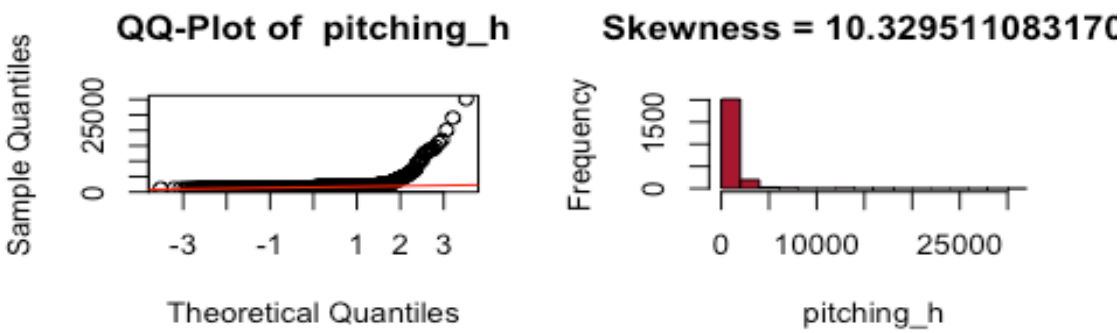
Skewness = -0.2625766575426
Frequency
0 250
0 400 800 1200
batting_so


QQ-Plot of baserun_sb
Sample Quantiles
0 500
-3 -1 1 2 3
Theoretical Quantiles

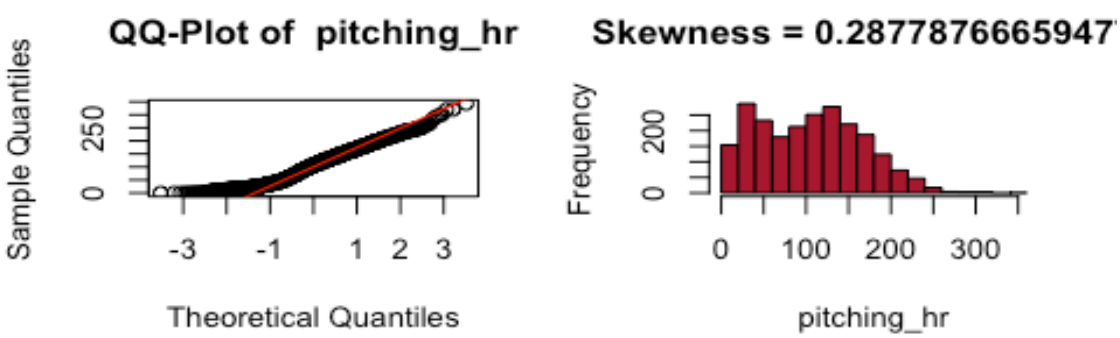
Skewness = 1.6200608463799
Frequency
0 600
0 200 400 600
baserun_sb


QQ-Plot of baserun_cs
Sample Quantiles
0 150
-3 -1 1 2 3
Theoretical Quantiles

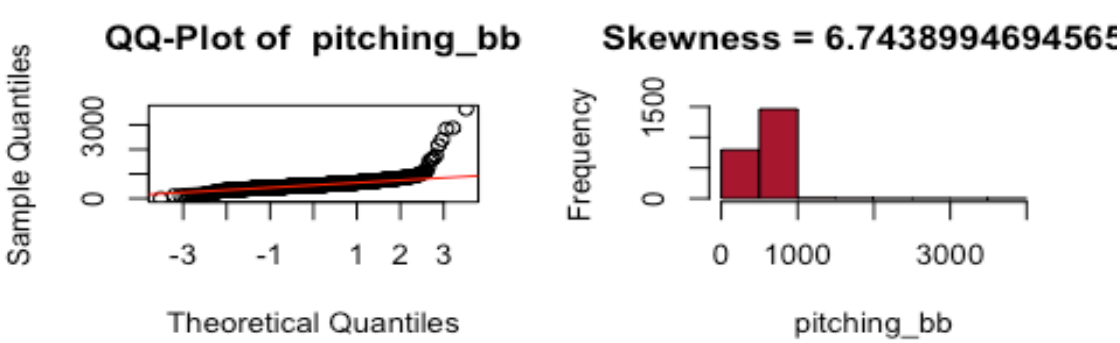
Skewness = 0.8399637117971
Frequency
0 600
0 50 100 200
baserun_cs


QQ-Plot of pitching_h
Sample Quantiles
0 25000
-3 -1 1 2 3
Theoretical Quantiles

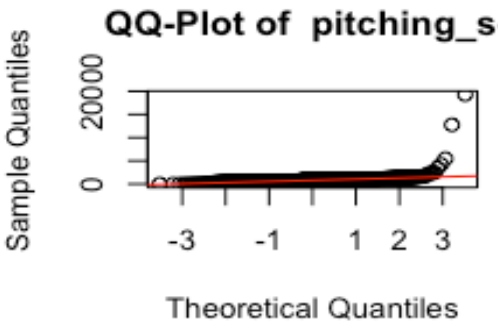
Skewness = 10.329511083170
Frequency
0 1500
0 10000 25000
pitching_h

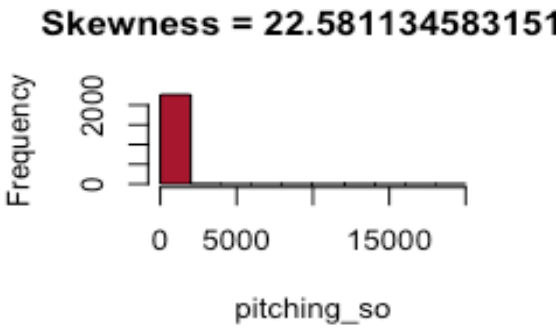

QQ-Plot of pitching_hr
Sample Quantiles
0 250
-3 -1 1 2 3
Theoretical Quantiles

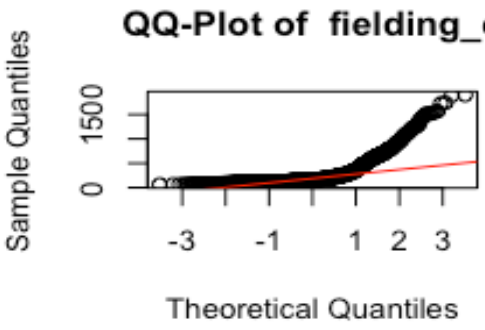
Skewness = 0.2877876665947
Frequency
0 200
0 100 200 300
pitching_hr

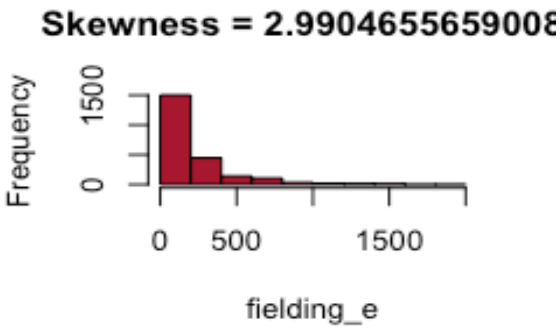

QQ-Plot of pitching_bb
Sample Quantiles
0 3000
-3 -1 1 2 3
Theoretical Quantiles

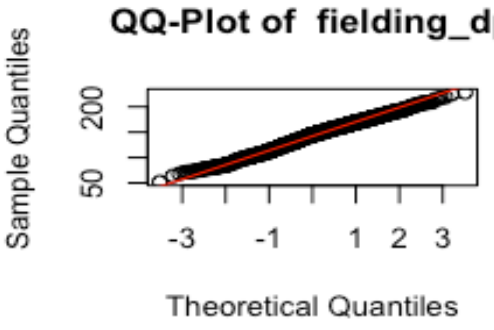
Skewness = 6.7438994694565
Frequency
0 1500
0 1000 3000
pitching_bb

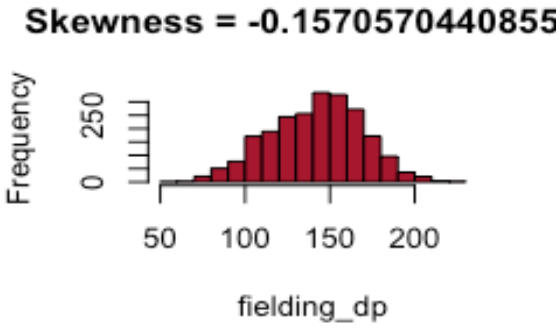

QQ-Plot of pitching_so
Sample Quantiles
0
20000
-3
-1
1
2
3
Theoretical Quantiles

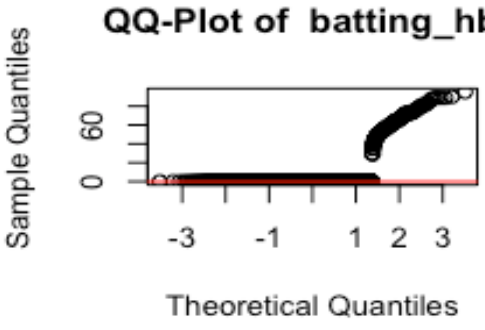

Skewness = 22.581134583151
Frequency
0
2000
0
5000
15000
pitching_so

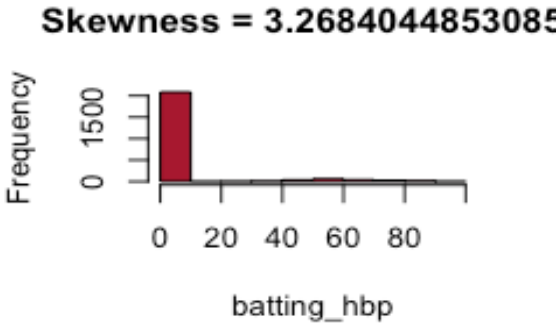

QQ-Plot of fielding_e
Sample Quantiles
0
1500
-3
-1
1
2
3
Theoretical Quantiles

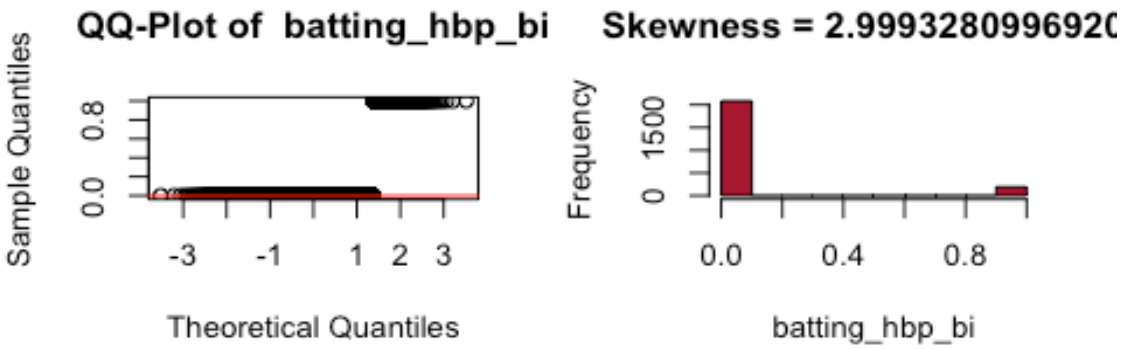

Skewness = 2.9904655659008
Frequency
0
1500
0
500
1500
fielding_e


QQ-Plot of fielding_dp
Sample Quantiles
50
200
-3
-1
1
2
3
Theoretical Quantiles


Skewness = -0.1570570440855
Frequency
0
250
50
100
150
200
fielding_dp


QQ-Plot of batting_hbp
Sample Quantiles
0
60
-3
-1
1
2
3
Theoretical Quantiles


Skewness = 3.2684044853085
Frequency
0
1500
0
20
40
60
80
batting_hbp



We would need to try certain transformation to correct for Skewness, with Box-Cox being the number one choice.QQ-plots are a great way to quickly gauge the normality of the variables.

Step 4: Is there collinearity among the covariates?

Let's create a series of correlation matix to understand how each independent variable interacts with the dependent variable. This correlation matix will help us spot any infrigement of the assupmtions needed to develop a robust OLS model, namely multicollinearity. The `caret` package can help the user find those pairs and even suggest which one to remove.

The Caret package offers the `findcorrelation()`, which takes the correlation matrix as an input and finds the fields causing multicollinearity based on a threshold, the `cutoff` parameter. It in turns returns a vector with values that would need to be removed from our dataset due to correlation.

```
colnames(moneyball_MF)[findCorrelation(cor(moneyball_MF))]

## [1] "batting_hr"  "batting_hbp"
```

Per caret's suggestion, we need to remove two variables in order to deal with the multicollinearity issue, `batting_hr and batting_hbp`. We will keep that in mind for

when we start the data transformation phase. For now, let's keep them since we need them for more feature engineering. ## Data Transformation

Let's introduce new variables through transformation:

1. `batting_1B = batting_h-(batting_2b + batting_3b + batting_hr)`
2. `free_bases_num = batting_hbp + batting_bb`
3. `total_bases = batting_1B + 2 * batting_2b + 3 * batting_3b + 4 * batting_hr + batting_bb + batting_hbp + baserun_sb`
4. `total_bases_allowed = pitching_bb + 4 * pitching_hr + pitching_h`
5. `HR_over_OP = batting_hr - pitching_hr`
6. `walks_over_OP = batting_bb - pitching_bb`
7. `SO_over_OP = pitching_so - batting_so`

```
moneyball_MF$batting_1B <- moneyball_MF$batting_h-
(moneyball_MF$batting_2b + moneyball_MF$batting_3b +
moneyball_MF$batting_hr)
moneyball_MF$free_bases_num <-
if_else(is.na(moneyball_MF$batting_hbp),0,as.numeric(moneyball_MF$batti
ng_hbp)) + moneyball_MF$batting_bb
moneyball_MF$total_bases <- moneyball_MF$batting_1B + 2 *
moneyball_MF$batting_2b + 3 * moneyball_MF$batting_3b + 4 *
moneyball_MF$batting_hr + moneyball_MF$batting_bb +
if_else(is.na(moneyball_MF$batting_hbp),0,as.numeric(moneyball_MF$batti
ng_hbp)) + moneyball_MF$baserun_sb
moneyball_MF$total_bases_allowed = moneyball_MF$pitching_bb + 4 *
moneyball_MF$pitching_hr + moneyball_MF$pitching_h
moneyball_MF$HR_over_OP = moneyball_MF$batting_hr -
moneyball_MF$pitching_hr
moneyball_MF$walks_over_OP = moneyball_MF$batting_bb -
moneyball_MF$pitching_bb
moneyball_MF$SO_over_OP = moneyball_MF$pitching_so -
moneyball_MF$batting_so
```

Now that we have imputed and created new variables, let's look at the correlation matrix to understand the correlation between the variables and the traget_wins. Remember when `caret` suggested to delete `batting_hr and batting_hbp` from our model? Let's build a correlaion matrix to understand why.

```
moneyball_MF <- subset(moneyball_MF, select = -c(batting_hbp))
cor(moneyball_MF)
```

Now that we created new variables, let's see what `caret` has to say about which variables to remove.

```
colnames(moneyball_MF)[findCorrelation(cor(moneyball_MF), cutoff =
0.9)]

## [1] "batting_hr"     "free_bases_num" "pitching_h"
```

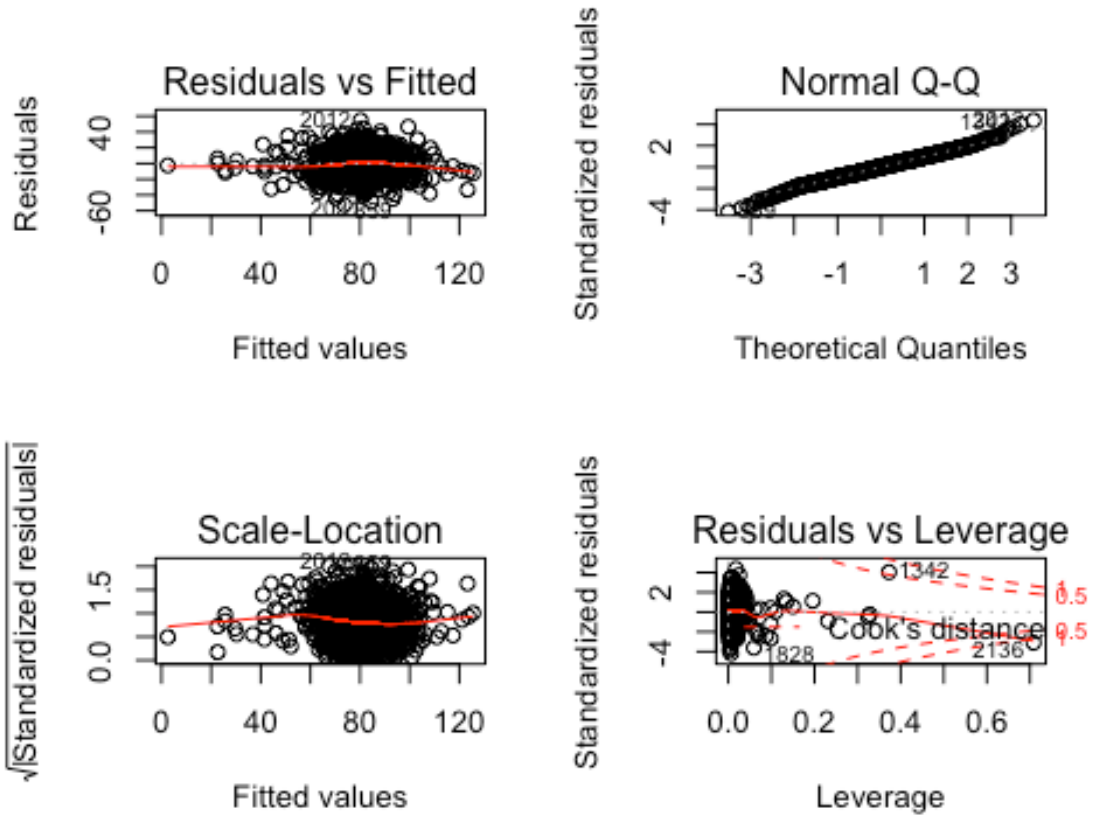
It suggesting `batting_hr` together with `free_bases_num and pitching_h`. According to the correlation matrix, `batting_hr` has a coefficient of correlation of 0.96 related to `pitching_hr`, `free_bases_num` has a coefficient of correlation of 0.99 related to `batting_bb`, and `pitching_h` has a coefficient of correlation of 0.99 related to `total_bases_allowed`. All these variables had a correlation of above th cuttoff point, 0.9. Let's remove those variables.

```
moneyball_MF <- subset(moneyball_MF, select = -c(batting_hr,
free_bases_num, pitching_h))
```

Build a Model

Let's test a model to establish a baseline

```
base_model_all <- lm(target_wins ~ batting_h + batting_2b + batting_3b
+ batting_bb + batting_so + baserun_sb + baserun_cs  + pitching_hr +
pitching_bb + pitching_so + fielding_e + fielding_dp  + batting_hbp_bi
+ batting_1B + total_bases + total_bases_allowed + HR_over_OP +
walks_over_OP + SO_over_OP, data = moneyball_MF)
par(mfrow = c(2,2))
plot(base_model_all)
```



```r
summary(base_model_all)
```

```
##
## Call:
## lm(formula = target_wins ~ batting_h + batting_2b + batting_3b +
##     batting_bb + batting_so + baserun_sb + baserun_cs + pitching_hr
+
##     pitching_bb + pitching_so + fielding_e + fielding_dp +
batting_hbp_bi +
##     batting_1B + total_bases + total_bases_allowed + HR_over_OP +
##     walks_over_OP + SO_over_OP, data = moneyball_MF)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -52.652  -8.409   0.127   8.244  54.758
##
## Coefficients: (3 not defined because of singularities)
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)         31.0416928  5.5994557   5.544 3.31e-08 ***
## batting_h           -0.1683958  0.2842424  -0.592   0.5536
## batting_2b           0.0551870  0.1442124   0.383   0.7020
## batting_3b           0.0128822  0.0765507   0.168   0.8664
## batting_bb          -0.0622595  0.0709777  -0.877   0.3805
## batting_so          -0.0147684  0.0025808  -5.723 1.19e-08 ***
## baserun_sb          -0.0246874  0.0708188  -0.349   0.7274
## baserun_cs           0.0209196  0.0157071   1.332   0.1830
## pitching_hr          0.0085637  0.0237397   0.361   0.7183
## pitching_bb         -0.0039146  0.0042004  -0.932   0.3515
## pitching_so          0.0017889  0.0008893   2.012   0.0444 *
## fielding_e          -0.0352387  0.0025861 -13.626  < 2e-16 ***
## fielding_dp         -0.1256495  0.0139520  -9.006  < 2e-16 ***
## batting_hbp_bi      -8.3004969  4.3301434  -1.917   0.0554 .
## batting_1B           0.1384228  0.2139085   0.647   0.5176
## total_bases          0.0740487  0.0707177   1.047   0.2952
## total_bases_allowed  0.0004301  0.0003704   1.161   0.2457
## HR_over_OP                  NA         NA      NA       NA
## walks_over_OP               NA         NA      NA       NA
## SO_over_OP                  NA         NA      NA       NA
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 12.62 on 2259 degrees of freedom
## Multiple R-squared:  0.3623, Adjusted R-squared:  0.3578
## F-statistic: 80.21 on 16 and 2259 DF,  p-value: < 2.2e-16
```

```r
mse <- function(sm)
  mean(sm$residuals^2)

paste('MSE equal ', mse(base_model_all), "and RMSE is ",
sqrt(mse(base_model_all)))
```

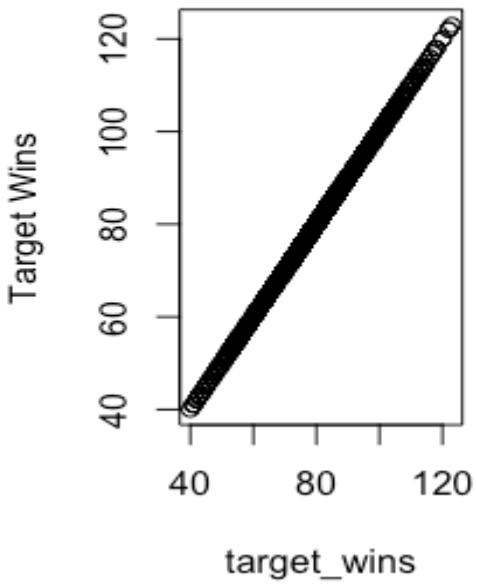
```
## [1] "MSE equal  158.166007039155 and RMSE is  12.576406761836"
```

Though R-squared and adjusted R-square is decent, we can clearly see that this model is not optmal. Let's try to forget about the new additions, and build a model without them.

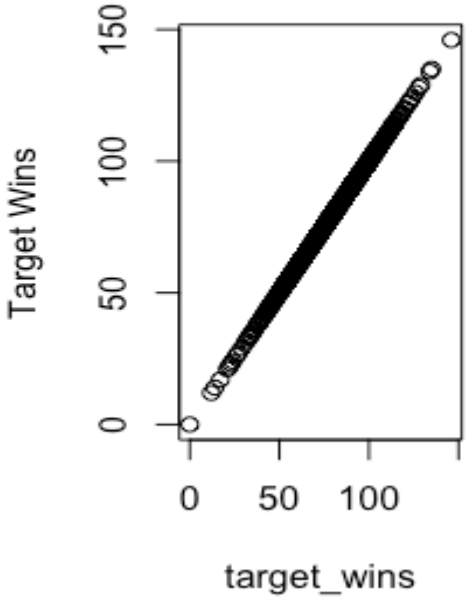
Let's fix the issue with outliers and see if we get any improvements. For the first approach we will use Winsoring approch.
For every outlier we will impute it with `Q1 - 1.5*IQR` or `Q3 + 1.5*IQR`, the cutoff for outliers.

```
outlier_treat <- moneyball_MF[,-c(1,15)]
comp_data <- moneyball_MF[,-c(1,15)]
i = 1
while (i %in% seq_along(outlier_treat)) {

qnt <- quantile(outlier_treat[,i], probs = c(.25, .75), na.rm = T)
caps <- quantile(outlier_treat[,i], probs = c(.05, .95), na.rm = T)
H <- 1.5 * IQR(outlier_treat[,i], na.rm = T)
outlier_treat[,i][outlier_treat[,i] < (qnt[1] - H)] <- caps[1]
outlier_treat[,i][outlier_treat[,i] > (qnt[2] + H)] <- caps[2]
 par(mfrow = c(1,2))
  plot(outlier_treat$target_wins, outlier_treat[,i],  xlab =
colnames(outlier_treat)[i] , ylab = "Target Wins", main =
paste("Treated Scatter Plot of ",colnames(outlier_treat)[i]))
  plot(comp_data$target_wins, comp_data[,i],xlab =
colnames(comp_data)[i] , ylab = "Target Wins", main = paste("Scatter
Plot of ",colnames(comp_data)[i]))
   i = i + 1
}
```

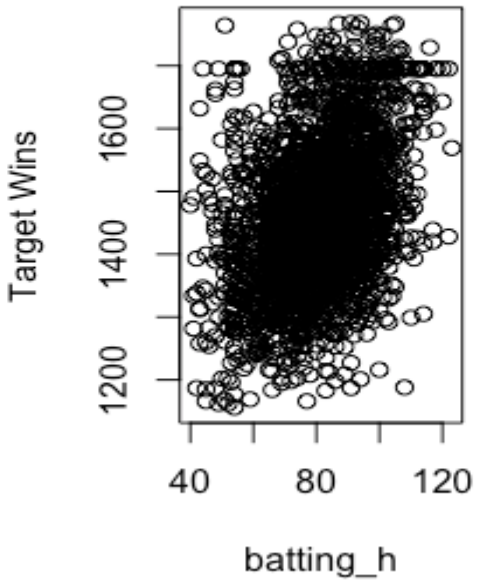
eated Scatter Plot of targe



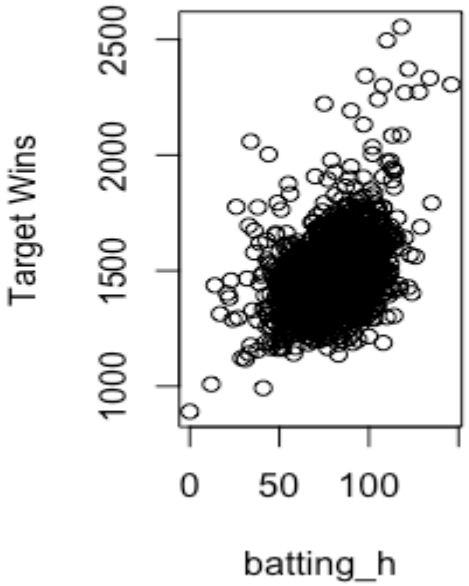
Scatter Plot of target_wi

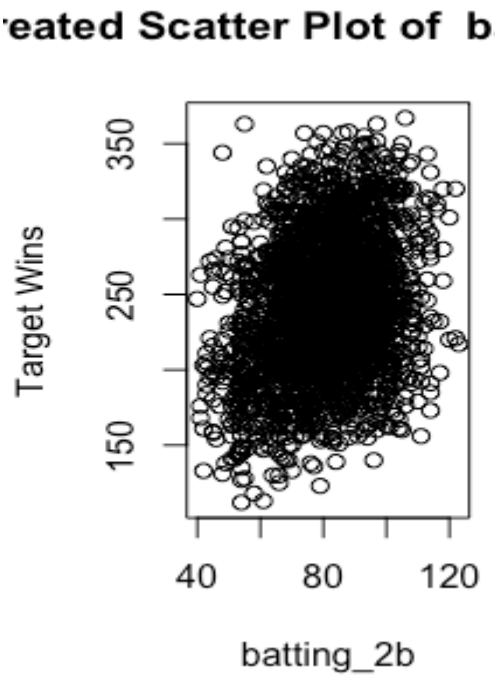


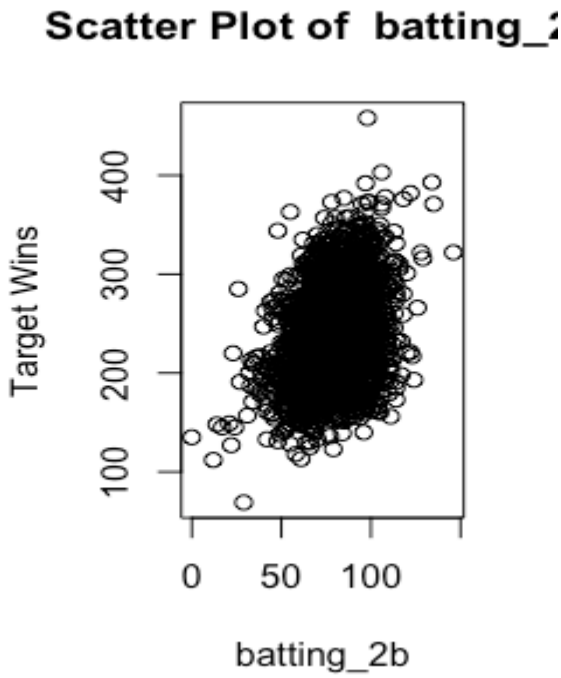
reated Scatter Plot of batt

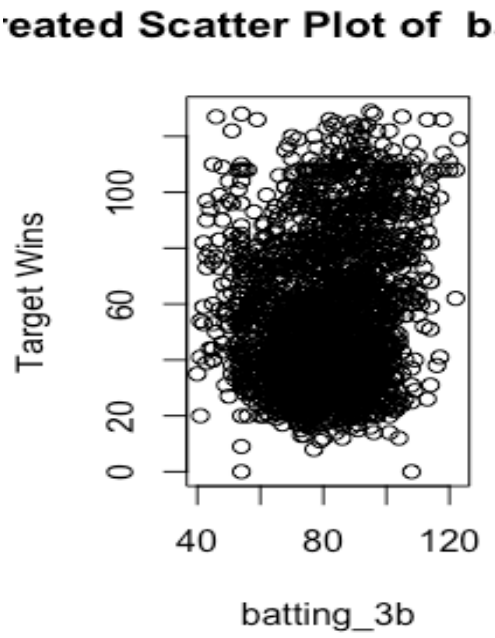


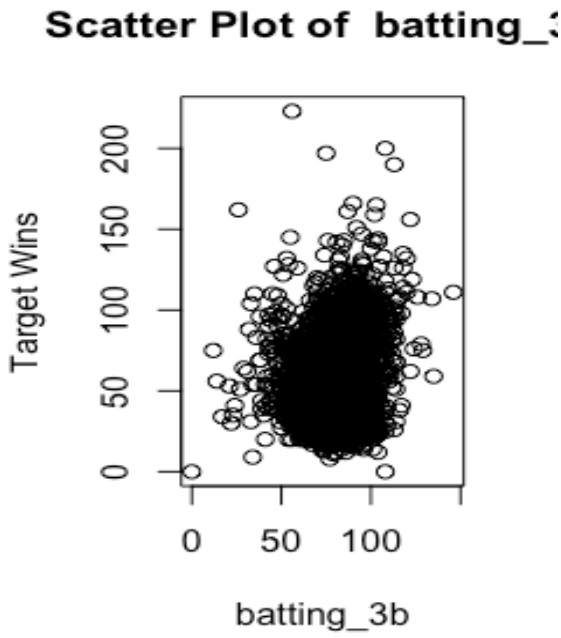
Scatter Plot of batting_

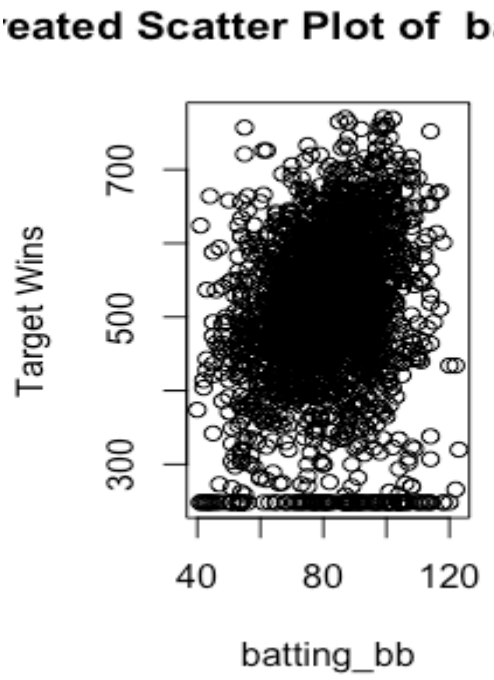


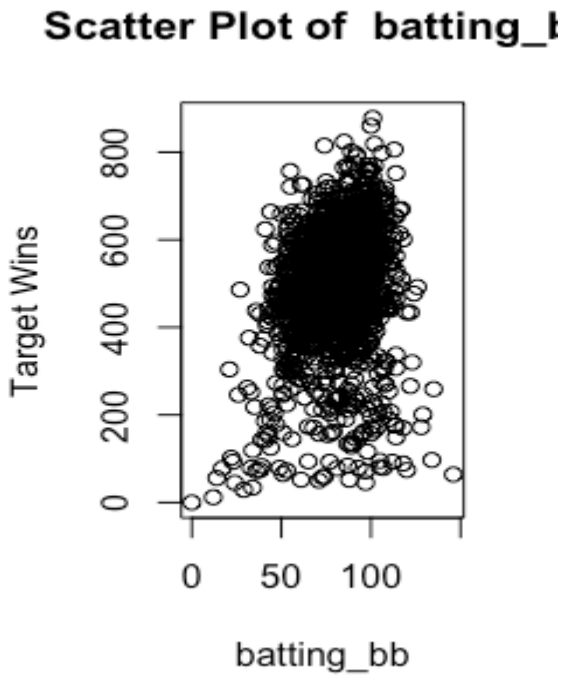

eated Scatter Plot of batti
Target Wins
150
250
350
40
80
120
batting_2b

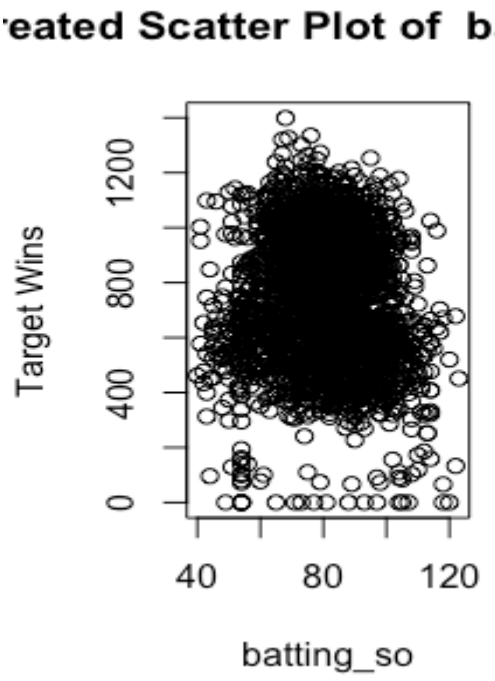

Scatter Plot of batting_2
Target Wins
100
200
300
400
0
50
100
batting_2b

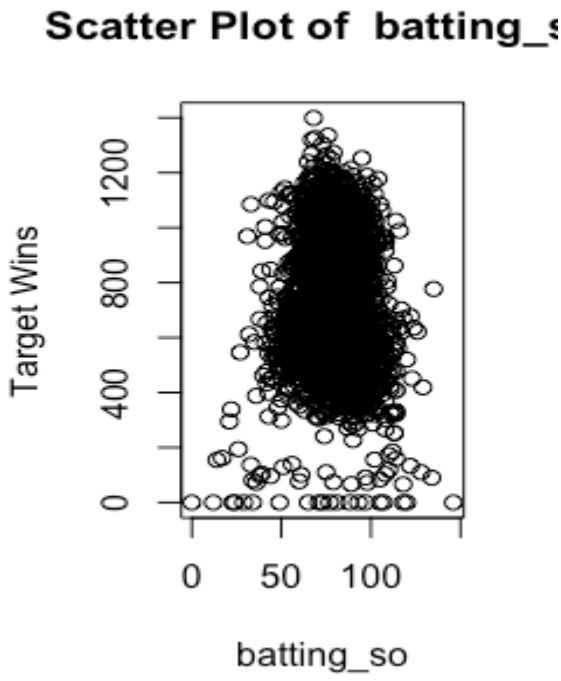

eated Scatter Plot of batti
Target Wins
0
20
60
100
40
80
120
batting_3b

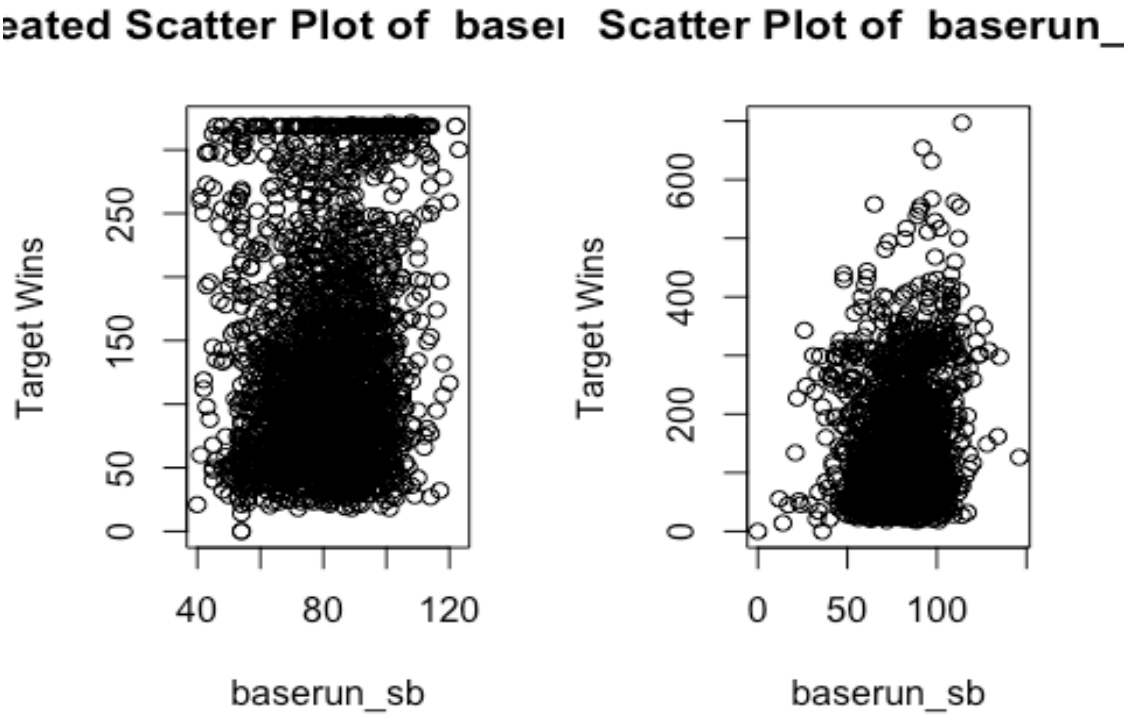

Scatter Plot of batting_3
Target Wins
0
50
100
150
200
0
50
100
batting_3b

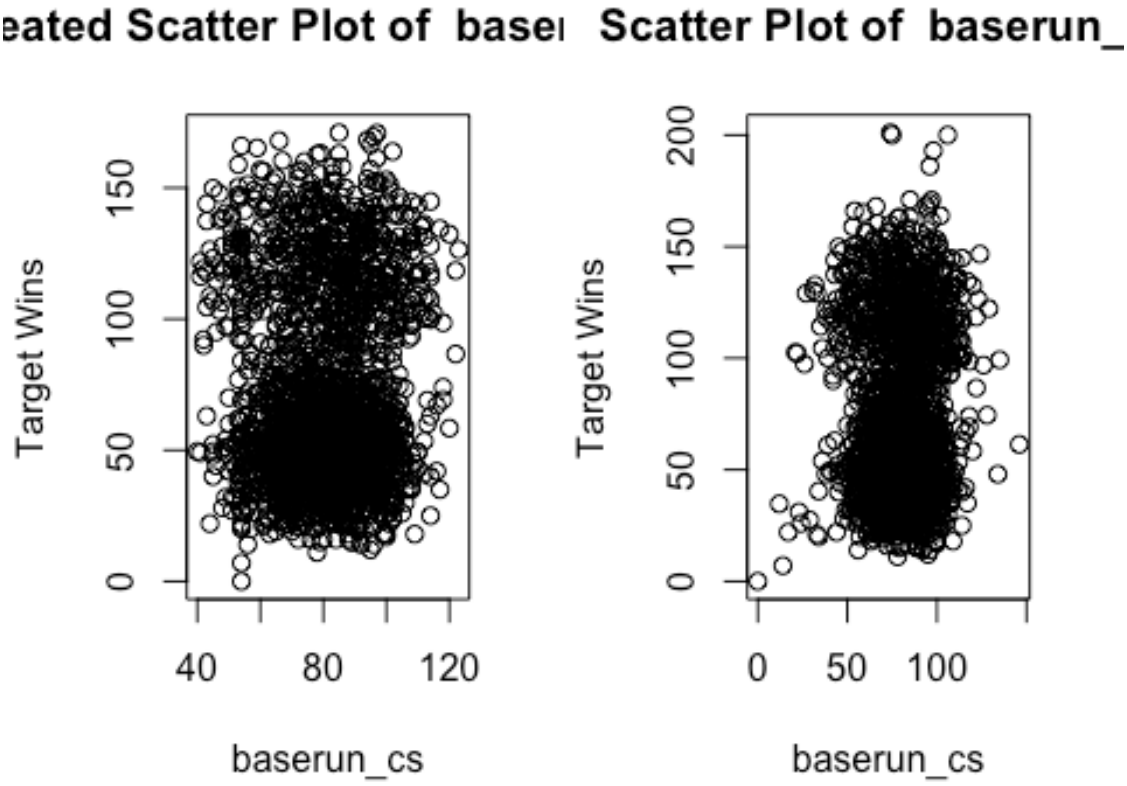

eated Scatter Plot of batti
Target Wins
300
500
700
40
80
120
batting_bb

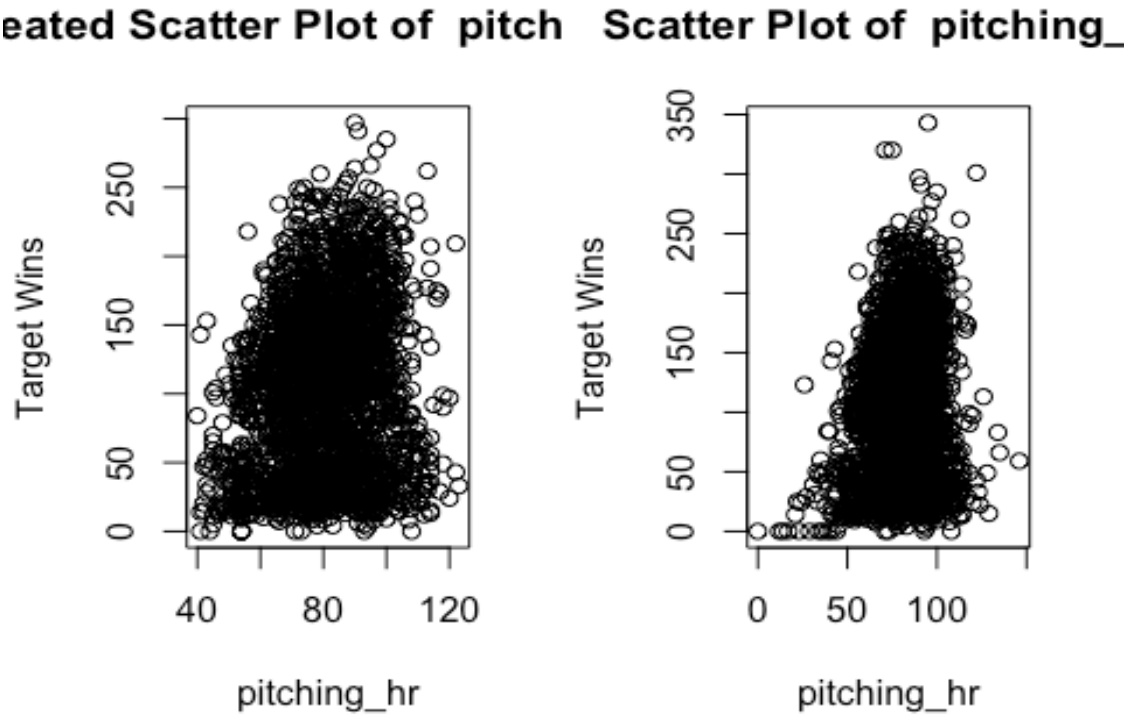

Scatter Plot of batting_b
Target Wins
0
200
400
600
800
0
50
100
batting_bb

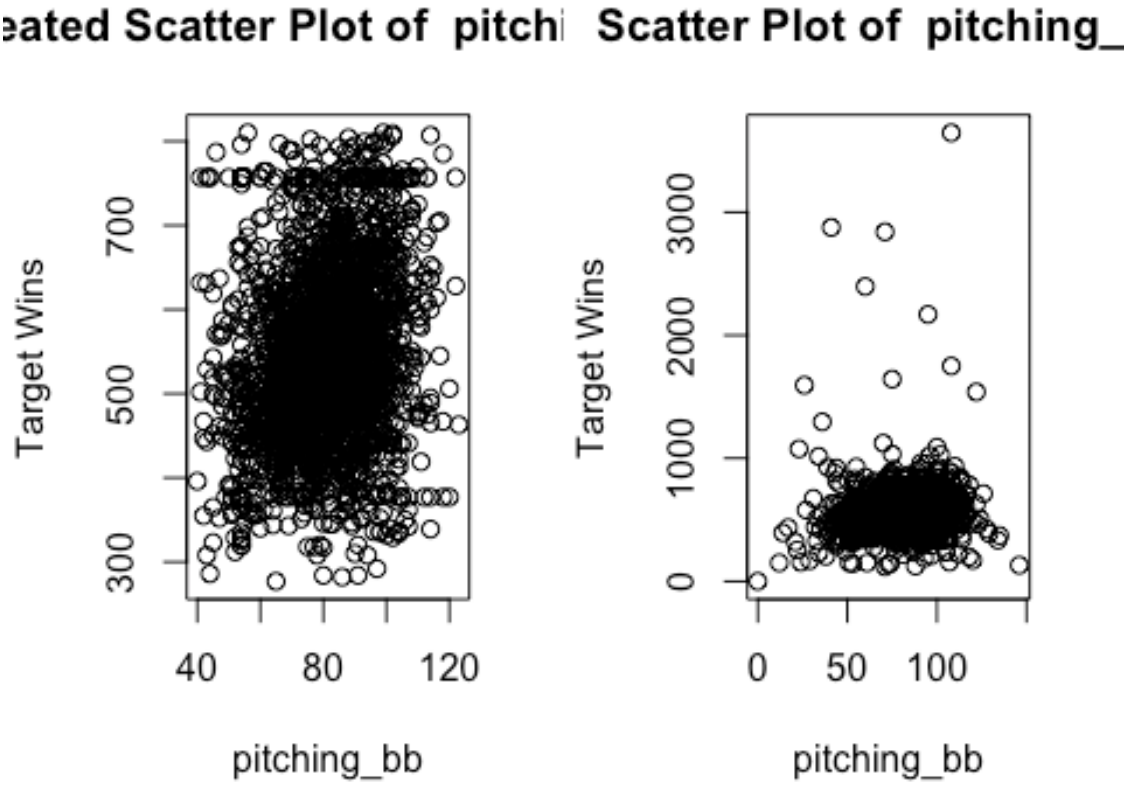

eated Scatter Plot of batti
Target Wins
0
400
800
1200
40
80
120
batting_so


Scatter Plot of batting_s
Target Wins
0
400
800
1200
0
50
100
batting_so

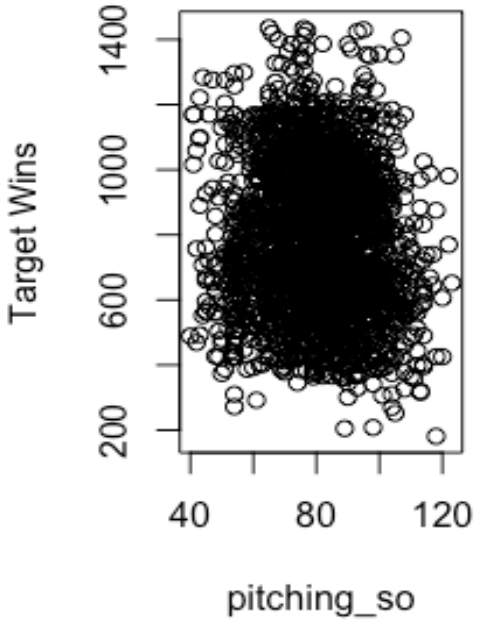

eated Scatter Plot of basel
Scatter Plot of baserun_
Target Wins
0
50
150
250
40
80
120
baserun_sb
Target Wins
0
200
400
600
0
50
100
baserun_sb


eated Scatter Plot of basel
Scatter Plot of baserun_
Target Wins
0
50
100
150
40
80
120
baserun_cs
Target Wins
0
50
100
150
200
0
50
100
baserun_cs

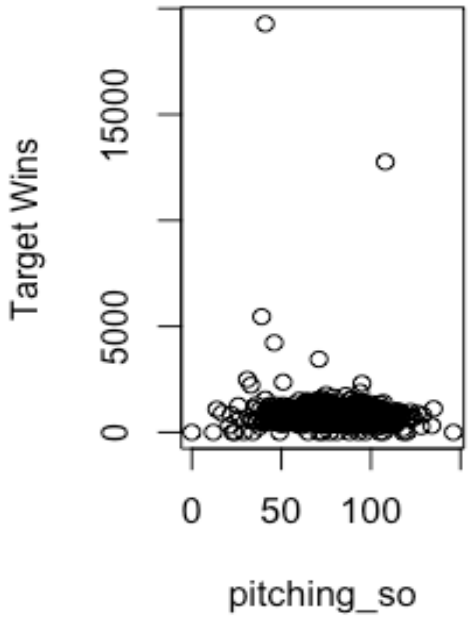

eated Scatter Plot of pitch
Scatter Plot of pitching_
Target Wins
0
50
150
250
40
80
120
pitching_hr
Target Wins
0
50
150
250
350
0
50
100
pitching_hr


eated Scatter Plot of pitch
Scatter Plot of pitching_
Target Wins
300
500
700
40
80
120
pitching_bb
Target Wins
0
1000
2000
3000
0
50
100
pitching_bb

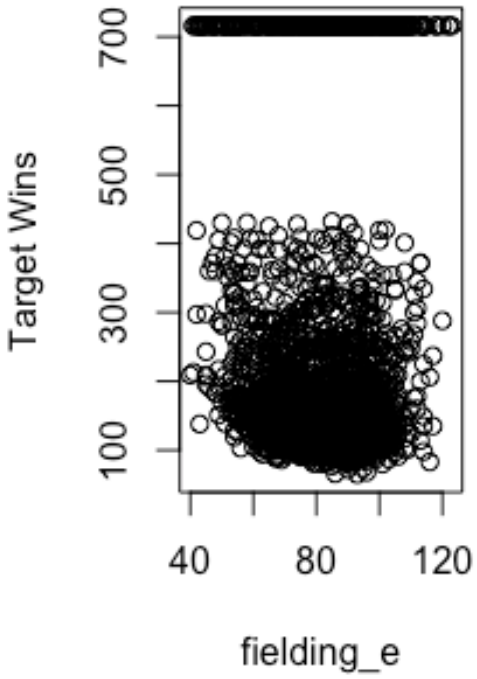
eated Scatter Plot of pitch



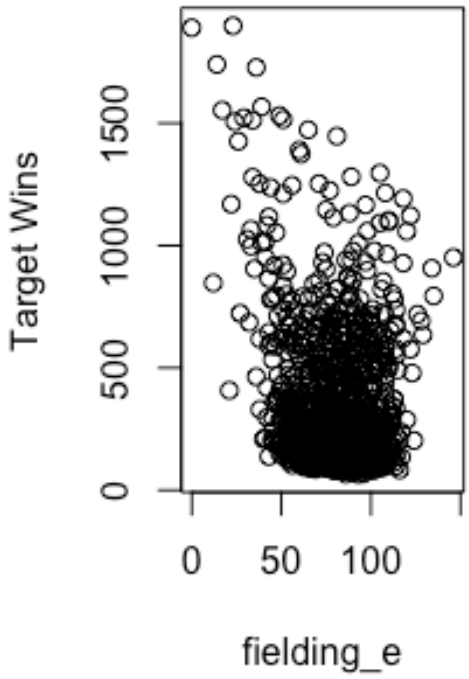
Scatter Plot of pitching_



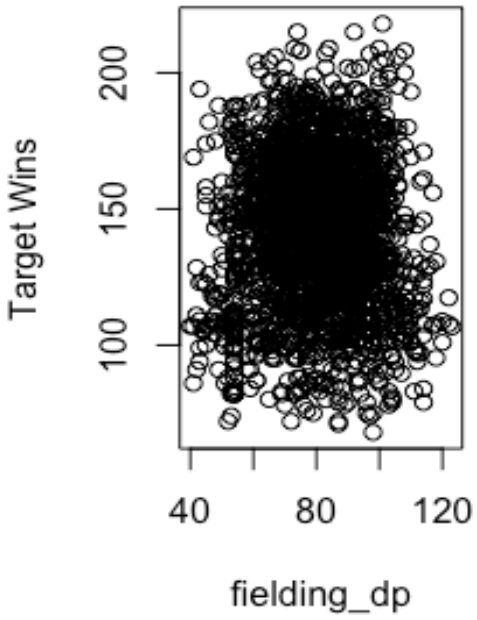
reated Scatter Plot of field



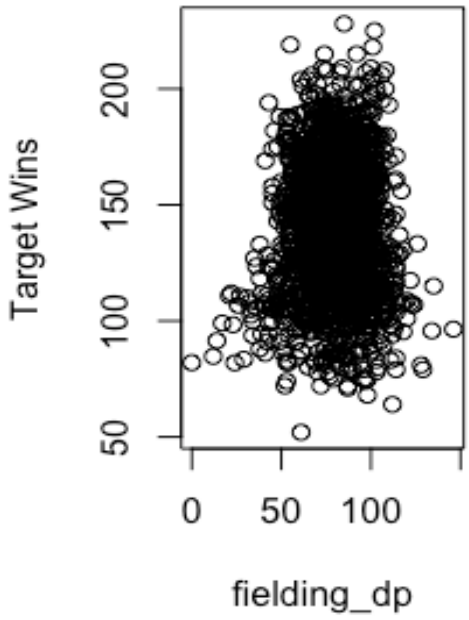
Scatter Plot of fielding_



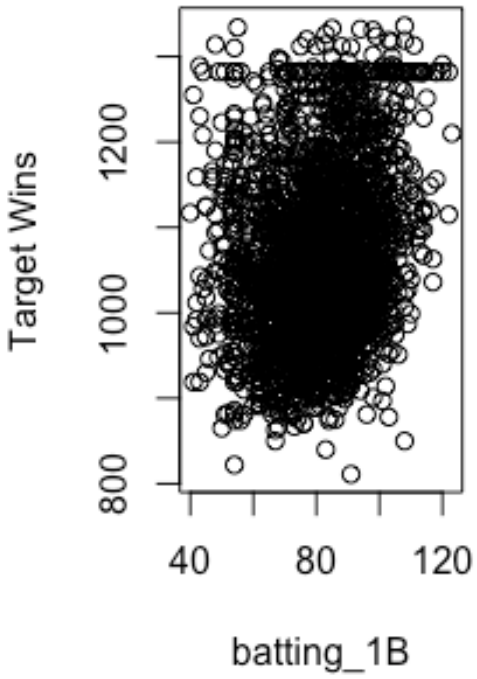
eated Scatter Plot of fieldi



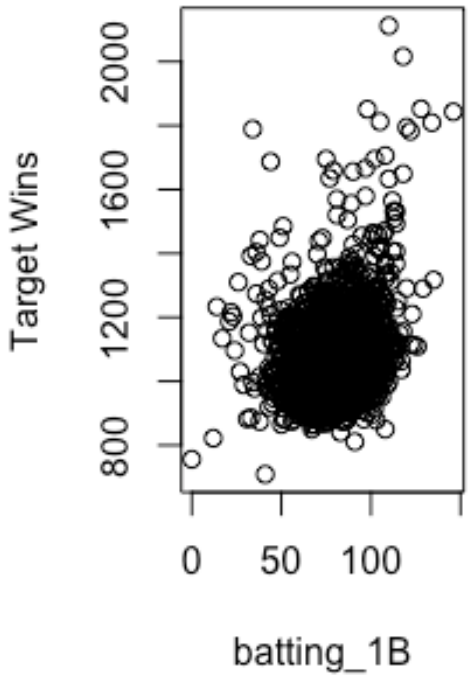
Scatter Plot of fielding_

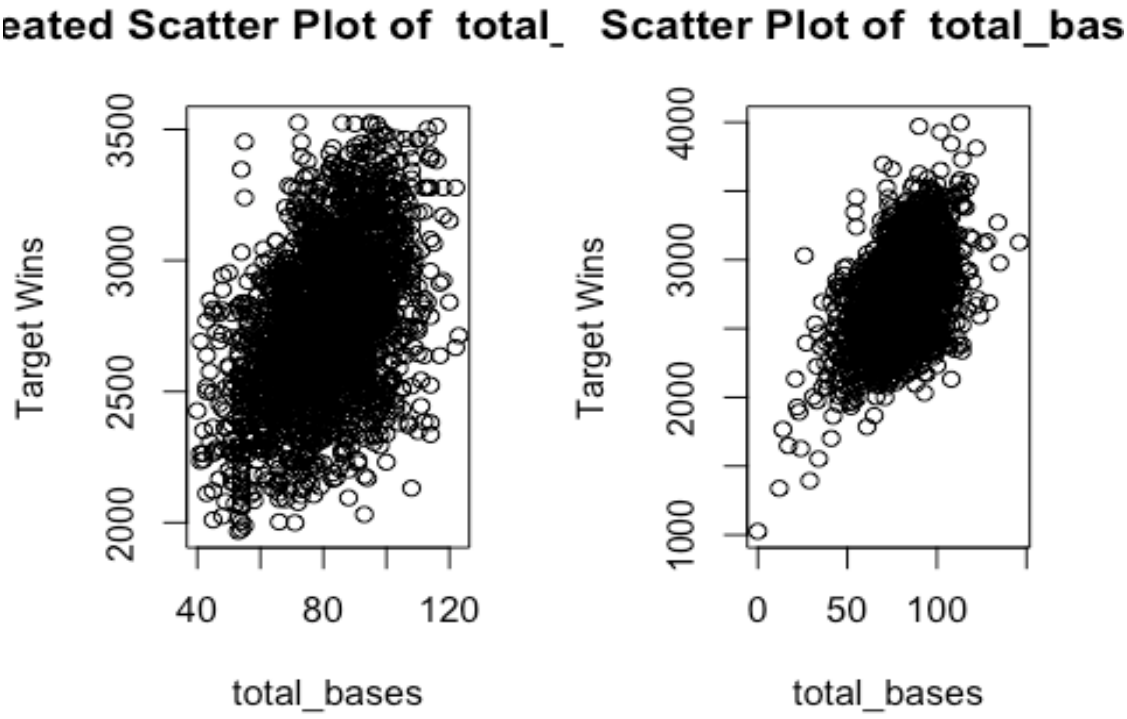


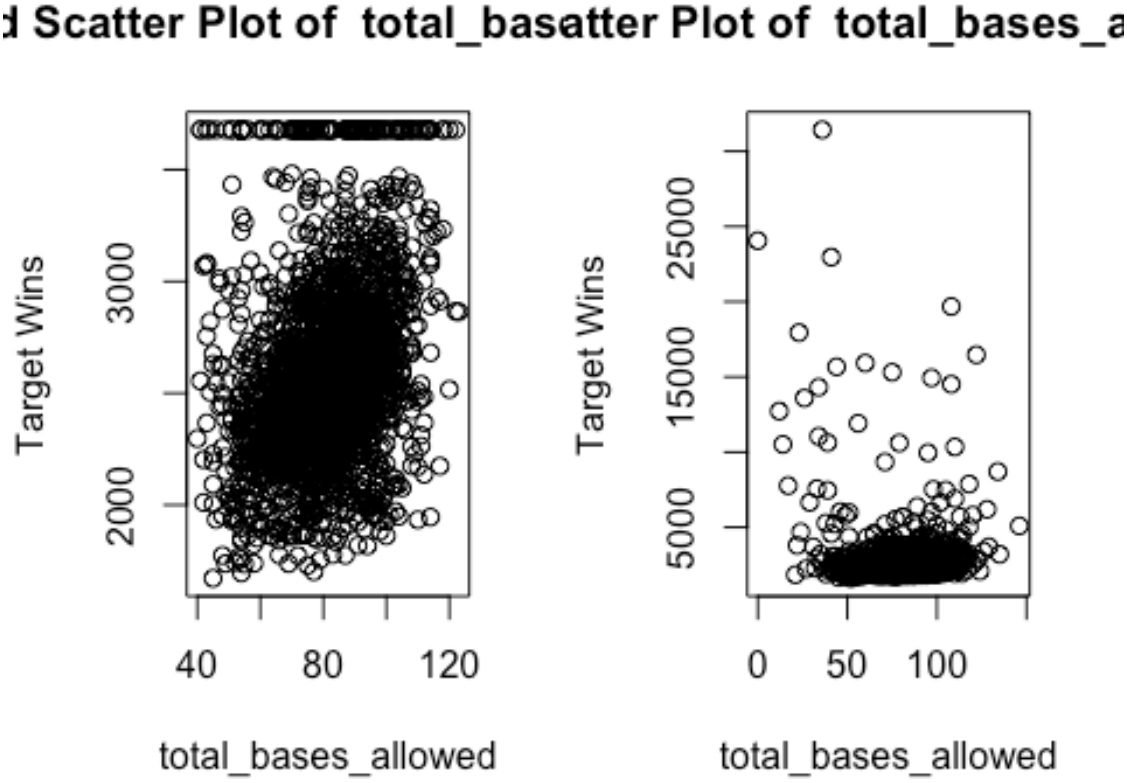
eated Scatter Plot of batti



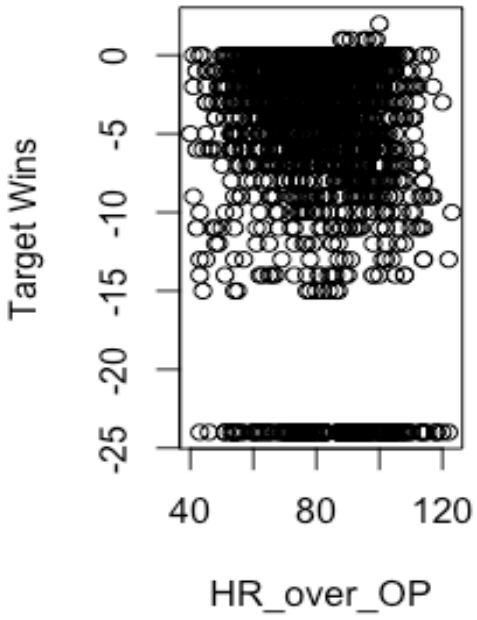
Scatter Plot of batting_1

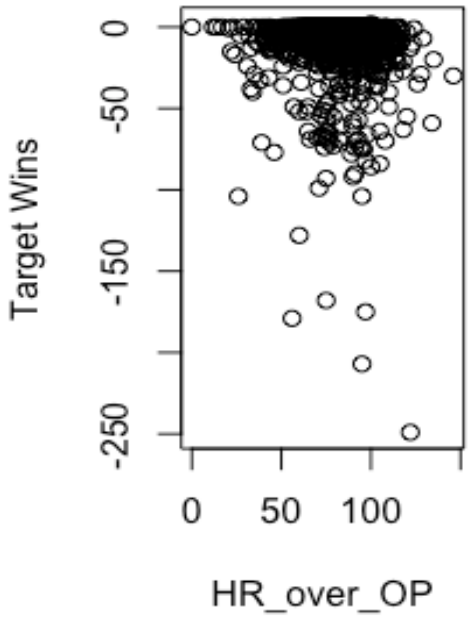



eated Scatter Plot of total_
Scatter Plot of total_bas
Target Wins
2000
2500
3000
3500
40
80
120
total_bases
Target Wins
1000
2000
3000
4000
0
50
100
total_bases

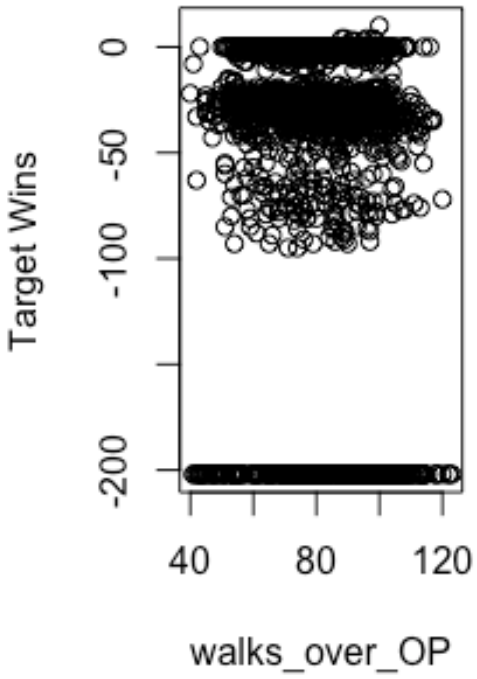

d Scatter Plot of total_basatter Plot of total_bases_a
Target Wins
2000
3000
40
80
120
total_bases_allowed
Target Wins
5000
15000
25000
0
50
100
total_bases_allowed

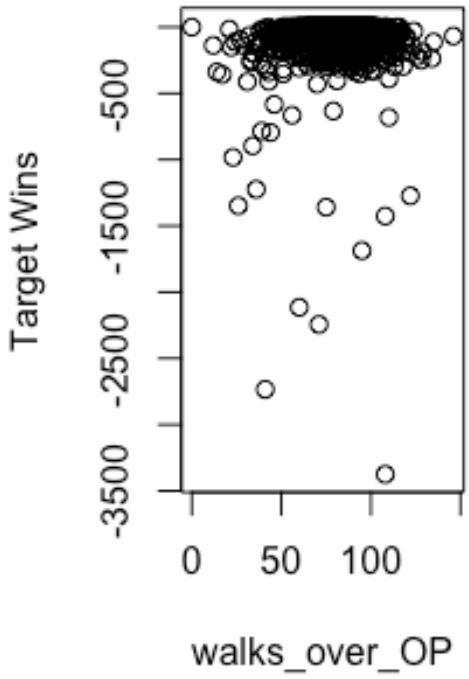
ated Scatter Plot of HR_o' Scatter Plot of HR_over_

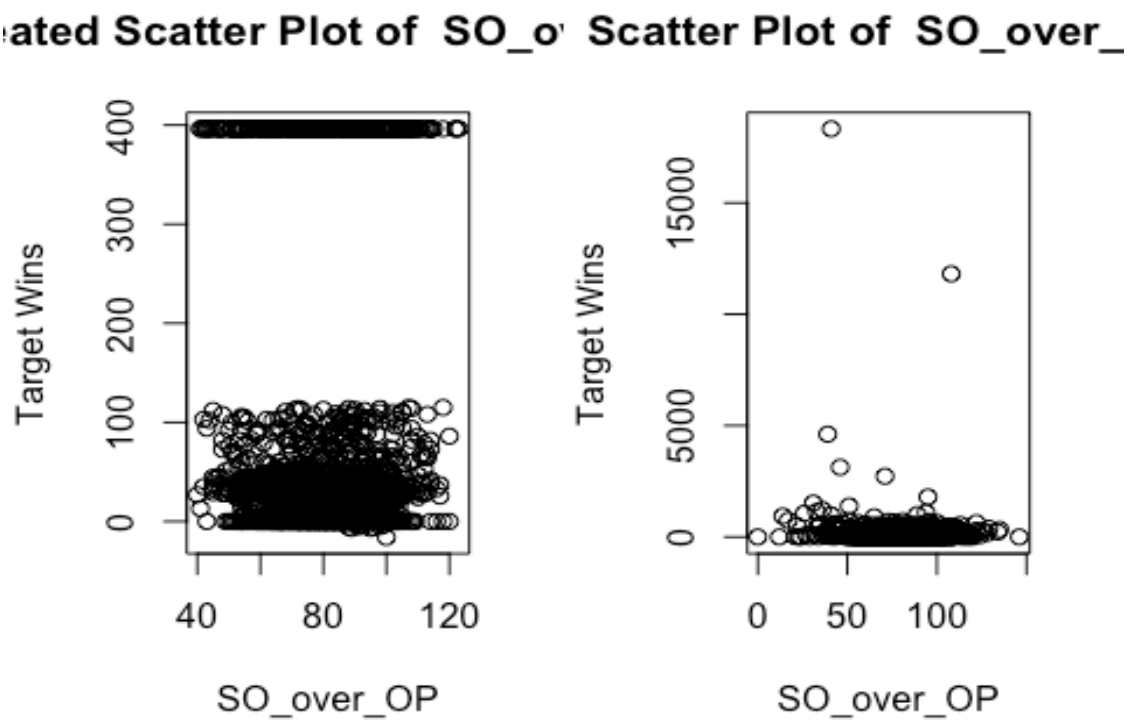




ted Scatter Plot of walks_Scatter Plot of walks_over



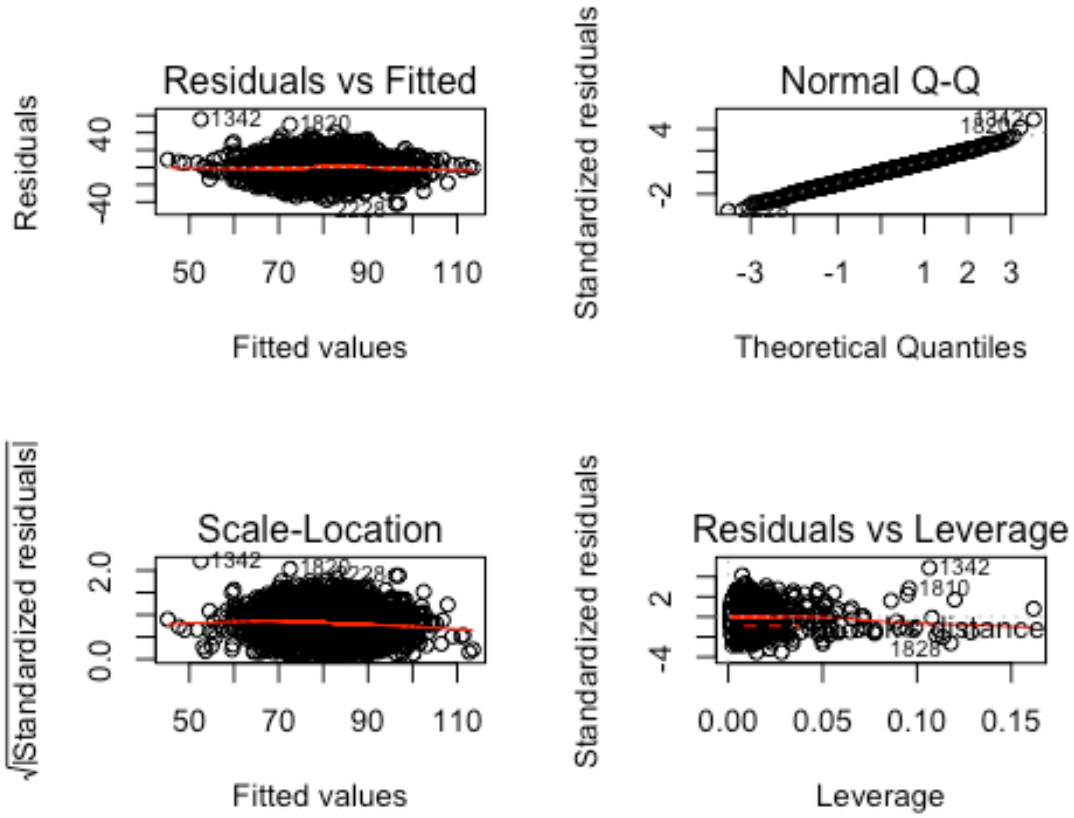





```
#add back the columns that was dropped prior to the outlier treatment
outlier_treat <- cbind(outlier_treat,moneyball_MF[,c(1,15)])
```

Let's try different models using the new data.

```
base_model_orig <-
  lm(target_wins ~ batting_h + batting_2b + batting_3b + batting_bb +
batting_so + baserun_sb + baserun_cs  + pitching_hr + pitching_bb +
pitching_so + fielding_e + fielding_dp  + batting_hbp_bi + batting_1B +
total_bases + total_bases_allowed + HR_over_OP + walks_over_OP +
SO_over_OP, data = outlier_treat)
  par(mfrow = c(2, 2))
  plot(base_model_orig)
```




```
  summary(base_model_orig)

##
## Call:
## lm(formula = target_wins ~ batting_h + batting_2b + batting_3b +
##     batting_bb + batting_so + baserun_sb + baserun_cs + pitching_hr
+
##     pitching_bb + pitching_so + fielding_e + fielding_dp +
batting_hbp_bi +
##     batting_1B + total_bases + total_bases_allowed + HR_over_OP +
##     walks_over_OP + SO_over_OP, data = outlier_treat)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -42.233  -8.088   0.253   7.745  55.427
##
## Coefficients:
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)         35.123619   6.138975   5.721 1.20e-08 ***
## batting_h            0.004055   0.013627   0.298  0.76608
## batting_2b          -0.035640   0.014628  -2.437  0.01491 *
## batting_3b           0.054027   0.022752   2.375  0.01765 *
## batting_bb           0.026962   0.009036   2.984  0.00288 **
```

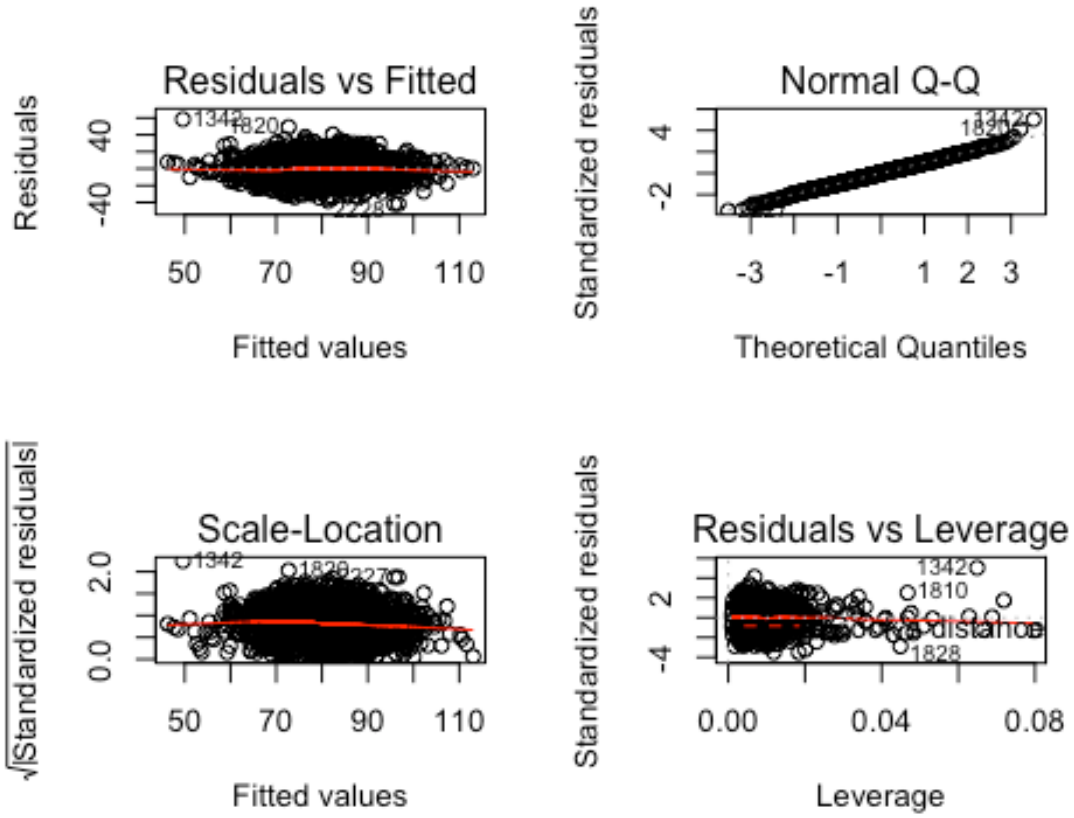
```
## batting_so          -0.011793   0.005336  -2.210  0.02721 *
## baserun_sb           0.063622   0.008905   7.145 1.21e-12 ***
## baserun_cs          -0.001510   0.017519  -0.086  0.93134
## pitching_hr         -0.013255   0.018338  -0.723  0.46987
## pitching_bb         -0.035640   0.007652  -4.658 3.38e-06 ***
## pitching_so         -0.005737   0.004952  -1.159  0.24674
## fielding_e          -0.042457   0.003071 -13.827  < 2e-16 ***
## fielding_dp         -0.098734   0.013254  -7.449 1.33e-13 ***
## batting_hbp_bi      -4.420561   1.132281  -3.904 9.73e-05 ***
## batting_1B          -0.009107   0.012942  -0.704  0.48173
## total_bases          0.022028   0.004502   4.893 1.06e-06 ***
## total_bases_allowed  0.012397   0.002050   6.047 1.72e-09 ***
## HR_over_OP           0.010129   0.081686   0.124  0.90132
## walks_over_OP        0.034089   0.010948   3.114  0.00187 **
## SO_over_OP           0.012261   0.004447   2.757  0.00588 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 12 on 2256 degrees of freedom
## Multiple R-squared:  0.3459, Adjusted R-squared:  0.3404
## F-statistic:  62.8 on 19 and 2256 DF,  p-value: < 2.2e-16
```

```
  paste('MSE equal ', mse(base_model_orig))
```

```
## [1] "MSE equal  142.763337742595"
```

This model looks good, from a performance point of view(r2), but when I look at the variance of the residual I don't feel secure. Specially after analyising Cook's distance graph. There are several observations that are way out from the rest. Let's build another model including only those with low p-Values.

```
base_model_lp <-
  lm(target_wins ~ batting_2b + batting_3b + batting_bb + batting_so +
baserun_sb + pitching_bb +
       fielding_e + fielding_dp + batting_hbp_bi + total_bases +
total_bases_allowed + walks_over_OP + SO_over_OP, data = outlier_treat)
  par(mfrow = c(2, 2))
  plot(base_model_lp)
```




```
  summary(base_model_lp)

##
## Call:
## lm(formula = target_wins ~ batting_2b + batting_3b + batting_bb +
##     batting_so + baserun_sb + pitching_bb + fielding_e + fielding_dp
+
##     batting_hbp_bi + total_bases + total_bases_allowed +
walks_over_OP +
##     SO_over_OP, data = outlier_treat)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -41.832  -8.064   0.175   7.833  58.399
##
## Coefficients:
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)         32.042515   3.269463   9.801  < 2e-16 ***
## batting_2b          -0.028986   0.009041  -3.206 0.001364 **
## batting_3b           0.058833   0.017710   3.322 0.000908 ***
## batting_bb           0.030852   0.008469   3.643 0.000276 ***
## batting_so          -0.016710   0.001804  -9.261  < 2e-16 ***
## baserun_sb           0.063457   0.005851  10.845  < 2e-16 ***
```

```
## pitching_bb         -0.036726   0.007442  -4.935 8.59e-07 ***
## fielding_e          -0.041706   0.002934 -14.214  < 2e-16 ***
## fielding_dp         -0.099706   0.012883  -7.739 1.50e-14 ***
## batting_hbp_bi      -4.226860   1.092614  -3.869 0.000113 ***
## total_bases          0.020971   0.002438   8.602  < 2e-16 ***
## total_bases_allowed  0.011086   0.001513   7.326 3.29e-13 ***
## walks_over_OP        0.034041   0.010551   3.226 0.001272 **
## SO_over_OP           0.010053   0.003898   2.579 0.009966 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 11.99 on 2262 degrees of freedom
## Multiple R-squared:  0.3452, Adjusted R-squared:  0.3414
## F-statistic: 91.72 on 13 and 2262 DF,  p-value: < 2.2e-16
```

```
  paste('MSE equal ', mse(base_model_lp))
```

```
## [1] "MSE equal  142.927674156311"
```

Though the rsquared value went down, there are some improvements on the Cook's distance chart. Now let's try to use use the `caret` package to apply the transformations we discussed earlier in our exploration phase. I will include all the variables minus the ones cause Multicollinearity issues.

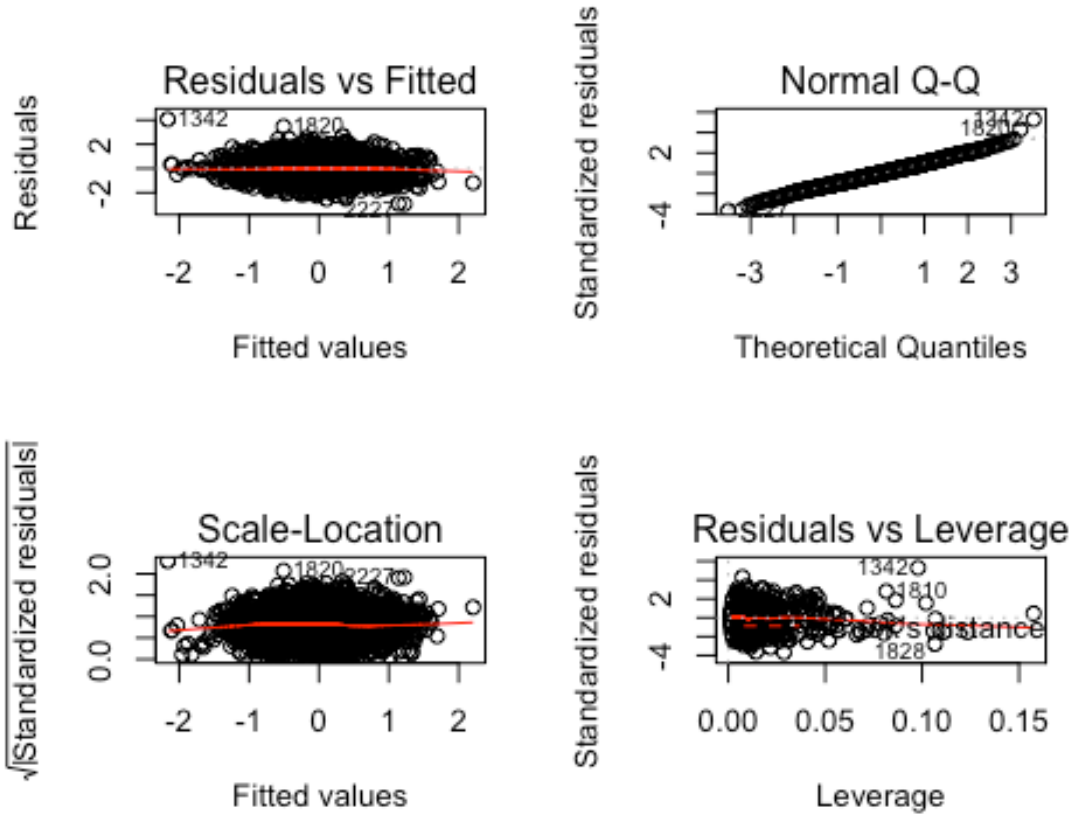
1. Center and Scale the data
2. Fix the the problem with outliers by using spatial sign Transformation

3. Last but not least a boxcox transformation to take car of the skewness

```
trans <- preProcess(outlier_treat, method = c("BoxCox","center",
"scale"))
transformed <- predict(trans, outlier_treat)
head(transformed)
```

```
##   target_wins  batting_h batting_2b batting_3b batting_bb batting_so
## 1 -1.76727657 -0.1111056 -1.0260875 -0.5976418 -2.0900610  0.4509070
## 2 -0.76118502 -1.0523406 -0.4551888 -1.2453898  1.8582866  1.4070416
## 3  0.31687312 -0.7066133 -0.1661113 -0.7500531  0.9112549  0.7586757
## 4 -0.76118502 -0.6172179 -0.6810714 -0.6357446 -0.5850077  0.7791936
## 5  0.04120325 -1.4460647 -1.2133573 -1.0548757 -0.3951198  0.7709864
## 6 -0.43149374 -1.6187221 -0.8871532 -0.7119502 -0.6557485  0.9884763
##   baserun_sb baserun_cs pitching_hr pitching_bb pitching_so
fielding_e
## 1  1.6766913  0.8365039 -0.35375893   1.8815144   1.6531432
1.73665912
## 2 -1.1245933 -1.1553448  1.40662888   1.3308308   1.2673652
0.25915028
## 3 -1.0147342 -1.1829825  0.51820887   0.5763990   0.5378137
0.06267378
## 4 -1.0513539 -1.1000694 -0.13988003  -0.8749743   0.5864504 -
0.07748718
```

```
## 5 -0.9781145 -0.8513302 -0.05761892  -0.6845719   0.5510782 -
0.49255669
## 6 -0.2701337 -0.2985763 -0.22214115  -0.9935698   0.7854190 -
0.80732240
##   fielding_dp batting_1B total_bases total_bases_allowed HR_over_OP
## 1  -1.1301341  1.2688903 -2.02196615          2.11040379 -2.9180993
## 2   0.4543355 -1.7915976  0.45456779          0.59390081  0.5315340
## 3   0.3777646 -0.9158042 -0.07574655         -0.05571369  0.6815181
## 4   0.4927324 -0.1066855 -1.04119059         -0.88758315  0.5315340
## 5   0.9591689 -0.8054425 -1.34714116         -1.08694911  0.6815181
## 6   0.2255249 -1.1961690 -1.30294830         -1.38831137  0.6815181
##   walks_over_OP SO_over_OP     index batting_hbp_bi
## 1    -2.2942806  2.5117883 -2.185553     -0.3025995
## 2     0.6102467 -0.5049697 -2.175875     -0.3025995
## 3     0.6689240 -0.5592210 -2.167613     -0.3025995
## 4     0.6249160 -0.5127199 -2.160153     -0.3025995
## 5     0.6689240 -0.5592210 -2.153239     -0.3025995
## 6     0.6689240 -0.5592210 -2.146731     -0.3025995

trans_model_all <-
  lm(target_wins ~ batting_h + batting_2b + batting_3b + batting_bb +
batting_so + baserun_sb + baserun_cs  + pitching_hr + pitching_bb +
pitching_so + fielding_e + fielding_dp  + batting_hbp_bi + batting_1B +
total_bases + total_bases_allowed + HR_over_OP + walks_over_OP +
SO_over_OP, data = transformed)
  par(mfrow = c(2, 2))
  plot(trans_model_all)
```




```
  summary(trans_model_all)

##
## Call:
## lm(formula = target_wins ~ batting_h + batting_2b + batting_3b +
##     batting_bb + batting_so + baserun_sb + baserun_cs + pitching_hr
+
##     pitching_bb + pitching_so + fielding_e + fielding_dp +
batting_hbp_bi +
##     batting_1B + total_bases + total_bases_allowed + HR_over_OP +
##     walks_over_OP + SO_over_OP, data = transformed)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.9324 -0.5430 -0.0133  0.5176  4.0620
##
## Coefficients:
##                         Estimate Std. Error t value Pr(>|t|)
## (Intercept)          1.343e-13  1.694e-02   0.000 1.000000
## batting_h           -4.445e-02  1.009e-01  -0.441 0.659536
## batting_2b          -4.135e-02  4.174e-02  -0.991 0.321924
## batting_3b           1.657e-01  3.972e-02   4.171 3.14e-05 ***
## batting_bb           1.950e-01  5.587e-02   3.490 0.000492 ***
```

```
## batting_so          -2.226e-01  8.274e-02  -2.690 0.007200 **
## baserun_sb           1.882e-01  4.701e-02   4.002 6.48e-05 ***
## baserun_cs           1.472e-01  4.353e-02   3.382 0.000733 ***
## pitching_hr          1.491e-02  7.246e-02   0.206 0.836935
## pitching_bb         -1.950e-01  4.696e-02  -4.152 3.42e-05 ***
## pitching_so         -9.011e-02  7.167e-02  -1.257 0.208772
## fielding_e          -5.770e-01  3.884e-02 -14.858  < 2e-16 ***
## fielding_dp         -1.951e-01  2.375e-02  -8.216 3.52e-16 ***
## batting_hbp_bi      -1.375e-01  2.145e-02  -6.413 1.73e-10 ***
## batting_1B           4.819e-02  7.787e-02   0.619 0.536041
## total_bases          3.530e-01  8.664e-02   4.075 4.77e-05 ***
## total_bases_allowed  2.413e-01  5.831e-02   4.138 3.64e-05 ***
## HR_over_OP          -5.382e-02  3.604e-02  -1.493 0.135467
## walks_over_OP        2.077e-01  5.003e-02   4.151 3.43e-05 ***
## SO_over_OP           6.148e-02  3.833e-02   1.604 0.108868
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.808 on 2256 degrees of freedom
## Multiple R-squared:  0.3526, Adjusted R-squared:  0.3472
## F-statistic: 64.68 on 19 and 2256 DF,  p-value: < 2.2e-16
```

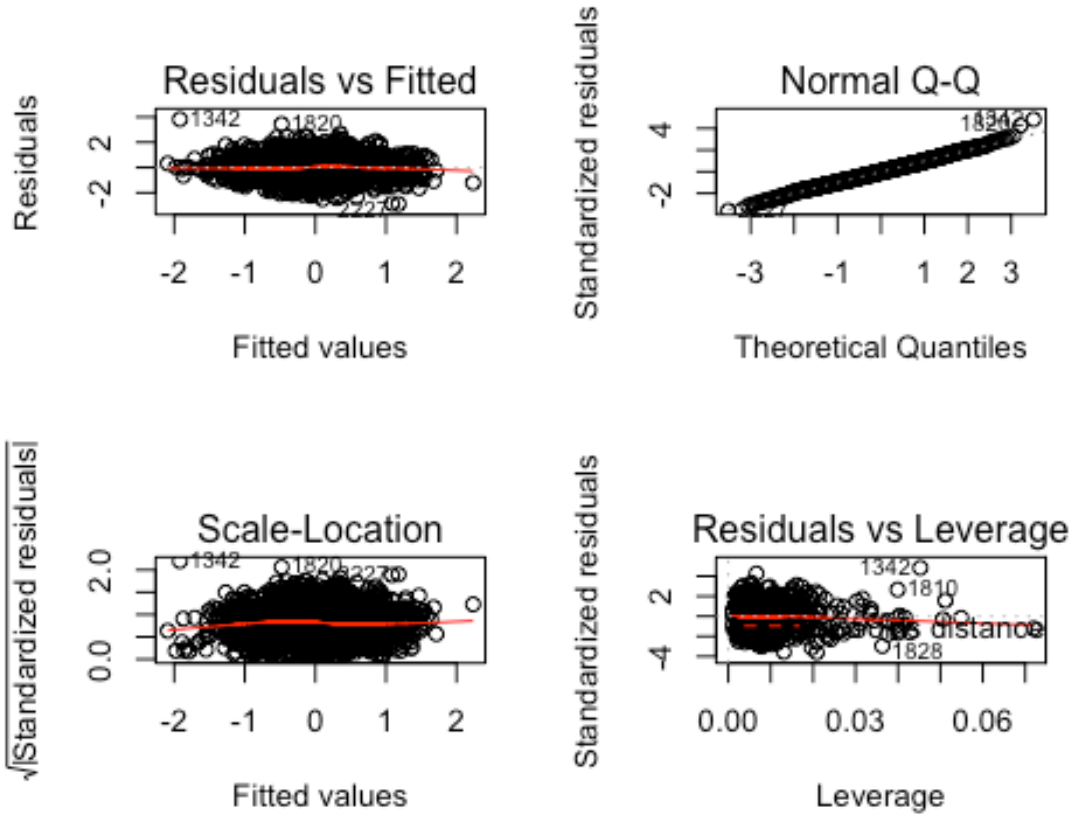
```
  paste('MSE equal ', mse(trans_model_all))
```

```
## [1] "MSE equal  0.647083867565306"
```

The residual plots look pretty good, with the exception of some possibly influential observation. Looking at Cook's Distance, it's clear that we have influential data, but the other charts look right where they should be.

Let's look at another model using the same transformed data, but now looking only on the columns with low p-value.

```
trans_model_lp <-
  lm(target_wins ~ batting_3b + batting_bb + batting_so + baserun_sb +
baserun_cs + pitching_bb + fielding_e + fielding_dp  + batting_hbp_bi +
total_bases + total_bases_allowed + walks_over_OP + SO_over_OP, data =
transformed)
  par(mfrow = c(2, 2))
  plot(trans_model_lp)
```




```
  summary(trans_model_lp)

##
## Call:
## lm(formula = target_wins ~ batting_3b + batting_bb + batting_so +
##     baserun_sb + baserun_cs + pitching_bb + fielding_e + fielding_dp
+
##     batting_hbp_bi + total_bases + total_bases_allowed +
walks_over_OP +
##     SO_over_OP, data = transformed)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.8851 -0.5568 -0.0115  0.5256  3.8180
##
## Coefficients:
##                         Estimate Std. Error t value Pr(>|t|)
## (Intercept)         -2.213e-15  1.695e-02   0.000  1.00000
## batting_3b           1.711e-01  3.287e-02   5.203 2.13e-07 ***
## batting_bb           2.165e-01  5.059e-02   4.280 1.94e-05 ***
## batting_so          -3.174e-01  3.045e-02 -10.426  < 2e-16 ***
## baserun_sb           1.919e-01  4.007e-02   4.790 1.77e-06 ***
## baserun_cs           1.388e-01  4.267e-02   3.253  0.00116 **
```

```
## pitching_bb          -1.972e-01  4.608e-02  -4.278 1.96e-05 ***
## fielding_e           -5.694e-01  3.797e-02 -14.996  < 2e-16 ***
## fielding_dp          -1.903e-01  2.341e-02  -8.129 7.06e-16 ***
## batting_hbp_bi       -1.529e-01  2.027e-02  -7.545 6.53e-14 ***
## total_bases           3.296e-01  4.125e-02   7.990 2.12e-15 ***
## total_bases_allowed   2.218e-01  4.262e-02   5.204 2.12e-07 ***
## walks_over_OP         1.903e-01  4.811e-02   3.955 7.89e-05 ***
## SO_over_OP            6.804e-02  3.372e-02   2.018  0.04374 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.8086 on 2262 degrees of freedom
## Multiple R-squared:  0.3499, Adjusted R-squared:  0.3462
## F-statistic: 93.65 on 13 and 2262 DF,  p-value: < 2.2e-16
```

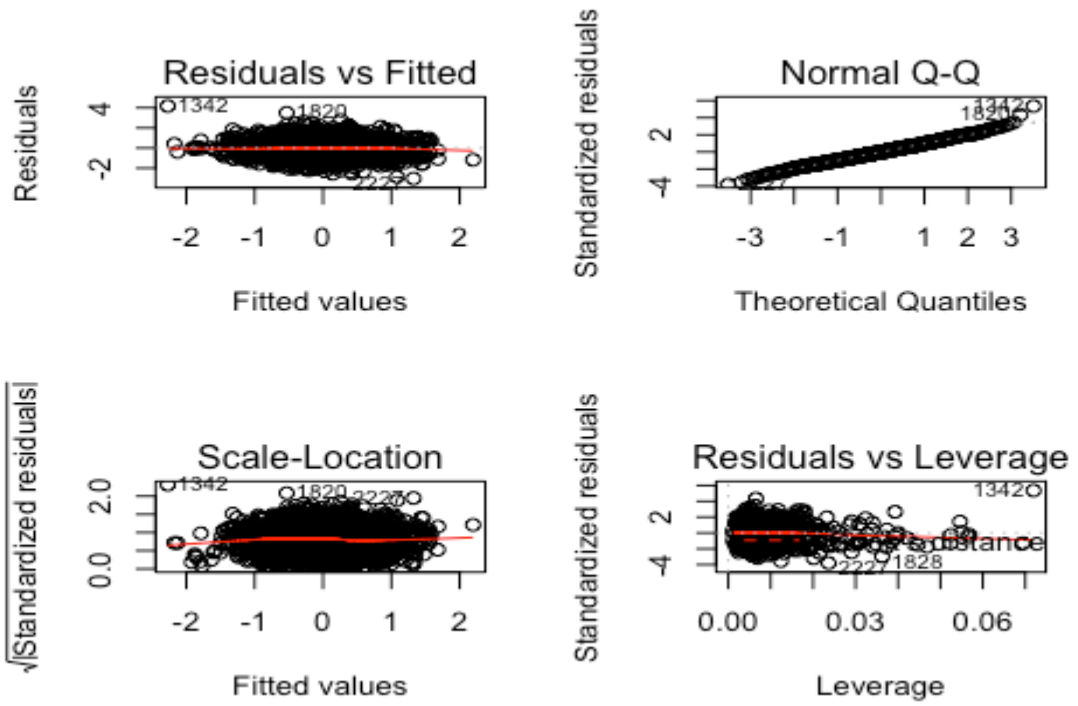
```
  paste('MSE equal ', mse(trans_model_lp))
```

```
## [1] "MSE equal  0.649826477602929"
```

This model seems to be on par with the other models. I'll try stepwise approaches and then I'll see if removing "influencial" observations will improve the model. Let's try, stepwise approach.

1. Both direction

```
stepwise_base_model_bd <- stepAIC(trans_model_all, direction = "both")

par(mfrow = c(2, 2))
  plot(stepwise_base_model_bd)
```



```
  summary(stepwise_base_model_bd)
```

```
##
## Call:
## lm(formula = target_wins ~ batting_2b + batting_3b + batting_bb +
##     batting_so + baserun_sb + baserun_cs + pitching_bb + fielding_e
+
##     fielding_dp + batting_hbp_bi + total_bases + total_bases_allowed
+
##     HR_over_OP + walks_over_OP, data = transformed)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.0325 -0.5448 -0.0108  0.5230  4.1704
##
## Coefficients:
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)         -1.764e-15  1.693e-02   0.000 1.000000
## batting_2b          -5.547e-02  2.728e-02  -2.033 0.042159 *
## batting_3b           1.549e-01  3.337e-02   4.642 3.64e-06 ***
## batting_bb           1.950e-01  5.158e-02   3.781 0.000160 ***
## batting_so          -3.124e-01  2.924e-02 -10.684  < 2e-16 ***
## baserun_sb           1.851e-01  4.059e-02   4.560 5.38e-06 ***
## baserun_cs           1.552e-01  4.219e-02   3.679 0.000239 ***
## pitching_bb         -1.952e-01  4.605e-02  -4.239 2.34e-05 ***
## fielding_e          -5.754e-01  3.812e-02 -15.095  < 2e-16 ***
## fielding_dp         -1.910e-01  2.345e-02  -8.146 6.15e-16 ***
## batting_hbp_bi      -1.423e-01  2.068e-02  -6.880 7.72e-12 ***
## total_bases          3.721e-01  4.813e-02   7.731 1.60e-14 ***
## total_bases_allowed  2.054e-01  4.323e-02   4.752 2.14e-06 ***
## HR_over_OP          -7.286e-02  3.029e-02  -2.406 0.016226 *
## walks_over_OP        1.805e-01  4.580e-02   3.941 8.35e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.8078 on 2261 degrees of freedom
## Multiple R-squared:  0.3515, Adjusted R-squared:  0.3475
## F-statistic: 87.55 on 14 and 2261 DF,  p-value: < 2.2e-16
```

```
paste('MSE equal ', mse(stepwise_base_model_bd))
```

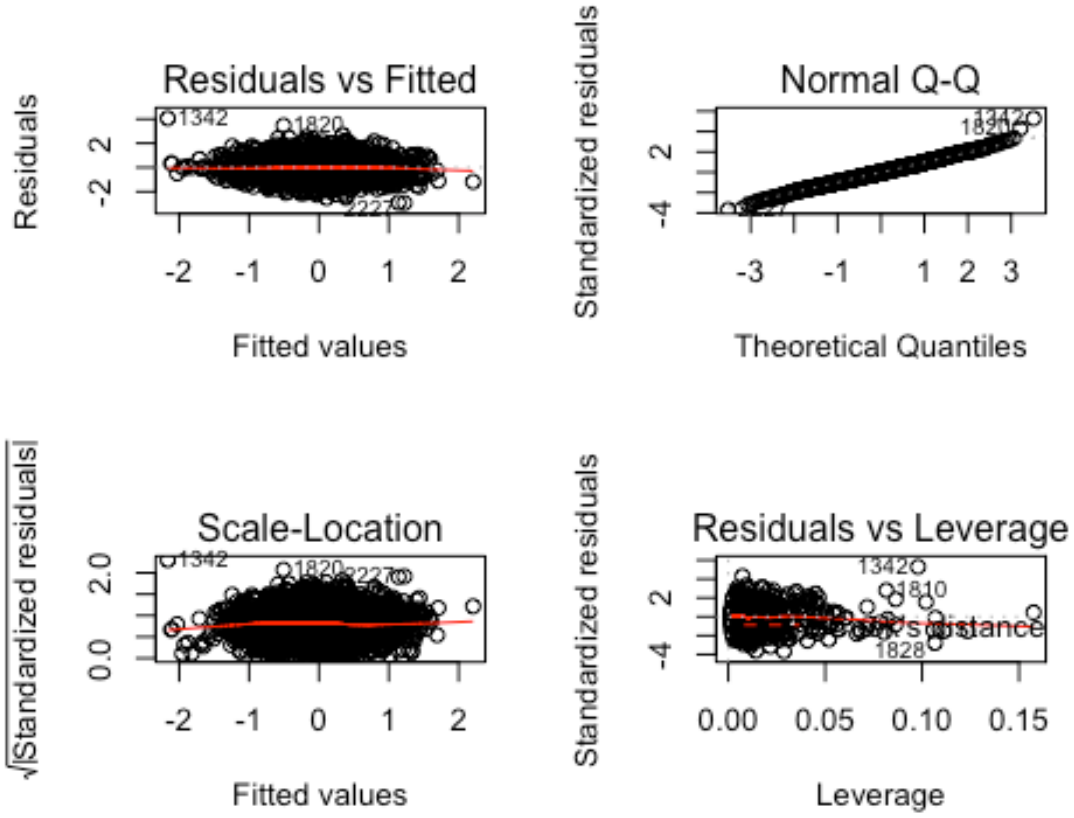
```
## [1] "MSE equal  0.648179170553685"
```

2. Forward direction

```
stepwise_base_model_fw <- stepAIC(trans_model_all, direction =
"forward")
```

```
## Start:  AIC=-950.7
## target_wins ~ batting_h + batting_2b + batting_3b + batting_bb +
##     batting_so + baserun_sb + baserun_cs + pitching_hr + pitching_bb
+
```

```
##     pitching_so + fielding_e + fielding_dp + batting_hbp_bi +
##     batting_1B + total_bases + total_bases_allowed + HR_over_OP +
##     walks_over_OP + SO_over_OP

par(mfrow = c(2, 2))
  plot(stepwise_base_model_fw)
```




```
  summary(stepwise_base_model_fw)

##
## Call:
## lm(formula = target_wins ~ batting_h + batting_2b + batting_3b +
##     batting_bb + batting_so + baserun_sb + baserun_cs + pitching_hr
+
##     pitching_bb + pitching_so + fielding_e + fielding_dp +
batting_hbp_bi +
##     batting_1B + total_bases + total_bases_allowed + HR_over_OP +
##     walks_over_OP + SO_over_OP, data = transformed)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.9324 -0.5430 -0.0133  0.5176  4.0620
##
## Coefficients:
```

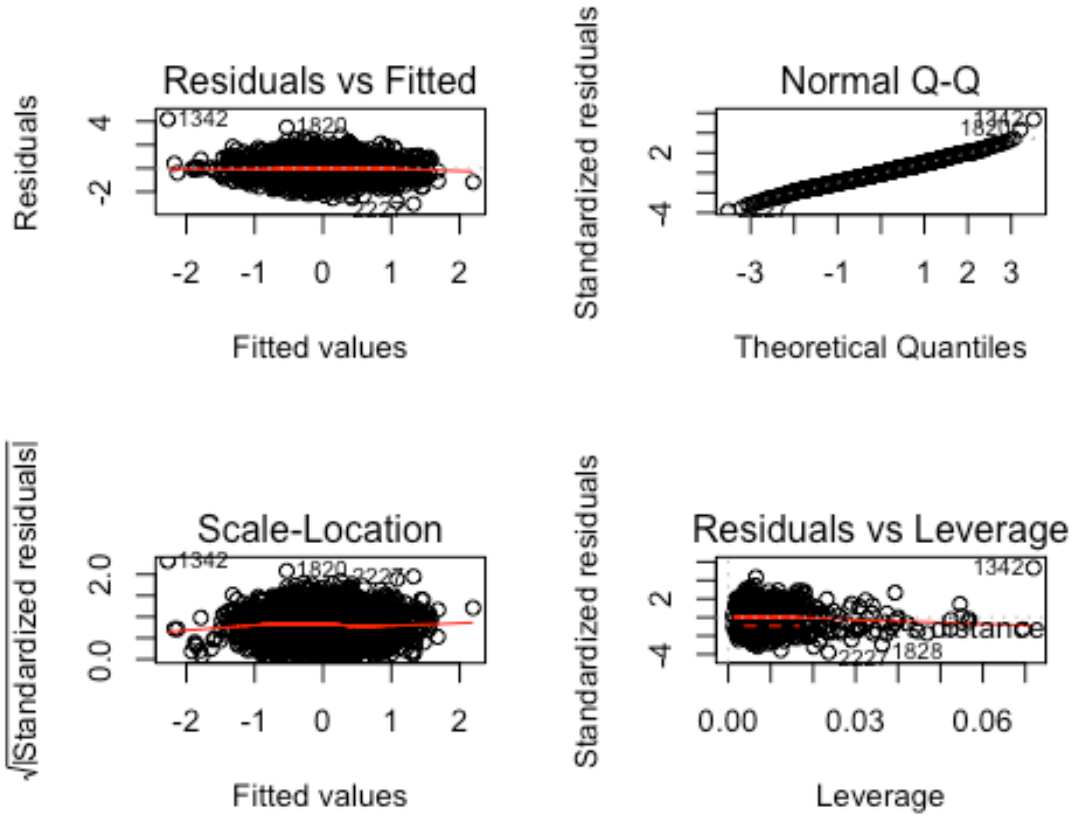
```
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)          1.343e-13  1.694e-02   0.000 1.000000
## batting_h           -4.445e-02  1.009e-01  -0.441 0.659536
## batting_2b          -4.135e-02  4.174e-02  -0.991 0.321924
## batting_3b           1.657e-01  3.972e-02   4.171 3.14e-05 ***
## batting_bb           1.950e-01  5.587e-02   3.490 0.000492 ***
## batting_so          -2.226e-01  8.274e-02  -2.690 0.007200 **
## baserun_sb           1.882e-01  4.701e-02   4.002 6.48e-05 ***
## baserun_cs           1.472e-01  4.353e-02   3.382 0.000733 ***
## pitching_hr          1.491e-02  7.246e-02   0.206 0.836935
## pitching_bb         -1.950e-01  4.696e-02  -4.152 3.42e-05 ***
## pitching_so         -9.011e-02  7.167e-02  -1.257 0.208772
## fielding_e          -5.770e-01  3.884e-02 -14.858  < 2e-16 ***
## fielding_dp         -1.951e-01  2.375e-02  -8.216 3.52e-16 ***
## batting_hbp_bi      -1.375e-01  2.145e-02  -6.413 1.73e-10 ***
## batting_1B           4.819e-02  7.787e-02   0.619 0.536041
## total_bases          3.530e-01  8.664e-02   4.075 4.77e-05 ***
## total_bases_allowed  2.413e-01  5.831e-02   4.138 3.64e-05 ***
## HR_over_OP          -5.382e-02  3.604e-02  -1.493 0.135467
## walks_over_OP        2.077e-01  5.003e-02   4.151 3.43e-05 ***
## SO_over_OP           6.148e-02  3.833e-02   1.604 0.108868
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.808 on 2256 degrees of freedom
## Multiple R-squared:  0.3526, Adjusted R-squared:  0.3472
## F-statistic: 64.68 on 19 and 2256 DF,  p-value: < 2.2e-16
```

```
paste('MSE equal ', mse(stepwise_base_model_fw))
```

```
## [1] "MSE equal  0.647083867565306"
```

3. Backwards direction

```
stepwise_base_model_bw <- stepAIC(trans_model_all, direction =
"backward")

par(mfrow = c(2, 2))
  plot(stepwise_base_model_bw)
```



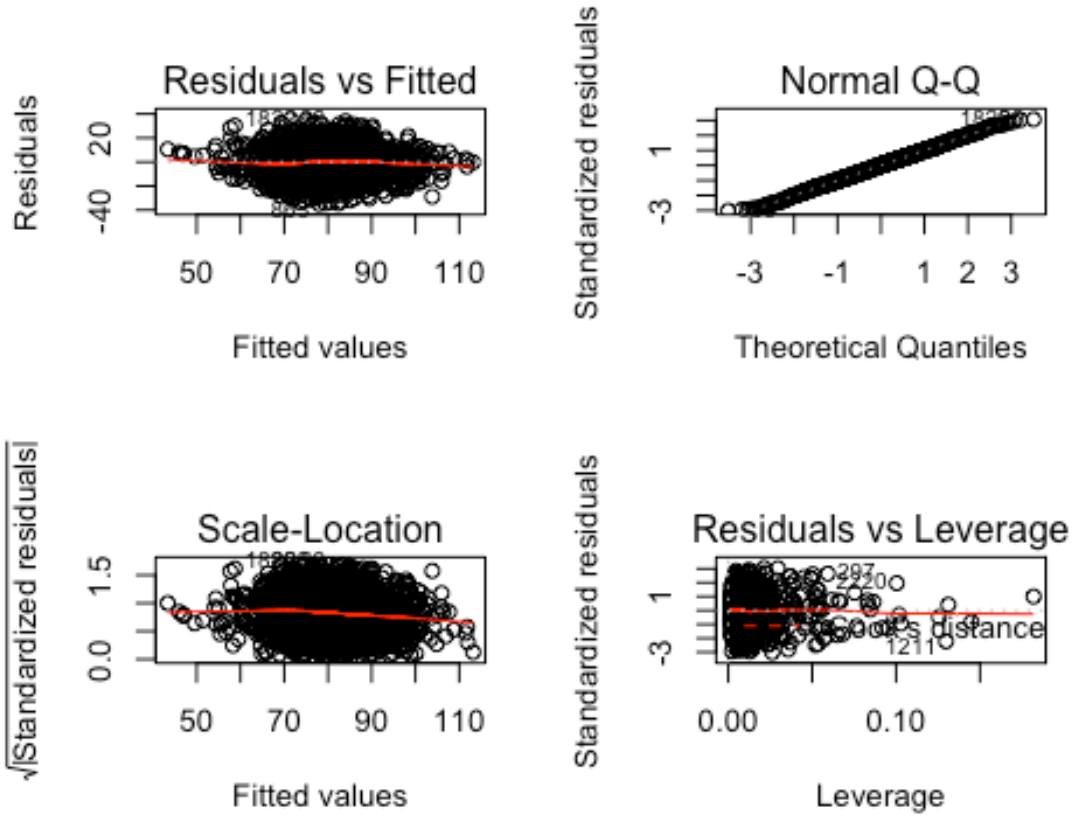

```
  summary(stepwise_base_model_bw)

##
## Call:
## lm(formula = target_wins ~ batting_2b + batting_3b + batting_bb +
##     batting_so + baserun_sb + baserun_cs + pitching_bb + fielding_e
+
##     fielding_dp + batting_hbp_bi + total_bases + total_bases_allowed
+
##     HR_over_OP + walks_over_OP, data = transformed)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.0325 -0.5448 -0.0108  0.5230  4.1704
##
## Coefficients:
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)         -1.764e-15  1.693e-02   0.000 1.000000
## batting_2b          -5.547e-02  2.728e-02  -2.033 0.042159 *
## batting_3b           1.549e-01  3.337e-02   4.642 3.64e-06 ***
## batting_bb           1.950e-01  5.158e-02   3.781 0.000160 ***
## batting_so          -3.124e-01  2.924e-02 -10.684  < 2e-16 ***
## baserun_sb           1.851e-01  4.059e-02   4.560 5.38e-06 ***
```

```
## baserun_cs            1.552e-01  4.219e-02   3.679 0.000239 ***
## pitching_bb          -1.952e-01  4.605e-02  -4.239 2.34e-05 ***
## fielding_e           -5.754e-01  3.812e-02 -15.095  < 2e-16 ***
## fielding_dp          -1.910e-01  2.345e-02  -8.146 6.15e-16 ***
## batting_hbp_bi       -1.423e-01  2.068e-02  -6.880 7.72e-12 ***
## total_bases           3.721e-01  4.813e-02   7.731 1.60e-14 ***
## total_bases_allowed   2.054e-01  4.323e-02   4.752 2.14e-06 ***
## HR_over_OP           -7.286e-02  3.029e-02  -2.406 0.016226 *
## walks_over_OP         1.805e-01  4.580e-02   3.941 8.35e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.8078 on 2261 degrees of freedom
## Multiple R-squared:  0.3515, Adjusted R-squared:  0.3475
## F-statistic: 87.55 on 14 and 2261 DF,  p-value: < 2.2e-16

paste('MSE equal ', mse(stepwise_base_model_bw))

## [1] "MSE equal  0.648179170553685"
```

Let's remove influential observations based on cook's distance chart. We will remove the following observations: 1342, 1810, 1828, 2136, 1820, 2227,1340, 1811, 2233, 1896, 2020, 2228.

Those observations will be removed from these datasets: transformed and outlier_treat

```
outlier_treat_rm <- outlier_treat[-c(1342, 1810, 1828, 2136, 1820,
2227,1340, 1811, 2233, 1896, 2020, 2228),]
transformed_rm <- transformed[-c(1342, 1810, 1828, 2136, 1820,
2227,1340, 1811, 2233, 1896, 2020, 2228),]

base_model_orig_rm <-
  lm(target_wins ~ batting_h + batting_2b + batting_3b + batting_bb +
batting_so + baserun_sb + baserun_cs  + pitching_hr + pitching_bb +
pitching_so + fielding_e + fielding_dp  + batting_hbp_bi + batting_1B +
total_bases + total_bases_allowed + HR_over_OP + walks_over_OP +
SO_over_OP, data = outlier_treat_rm)
  par(mfrow = c(2, 2))
  plot(base_model_orig_rm)
```



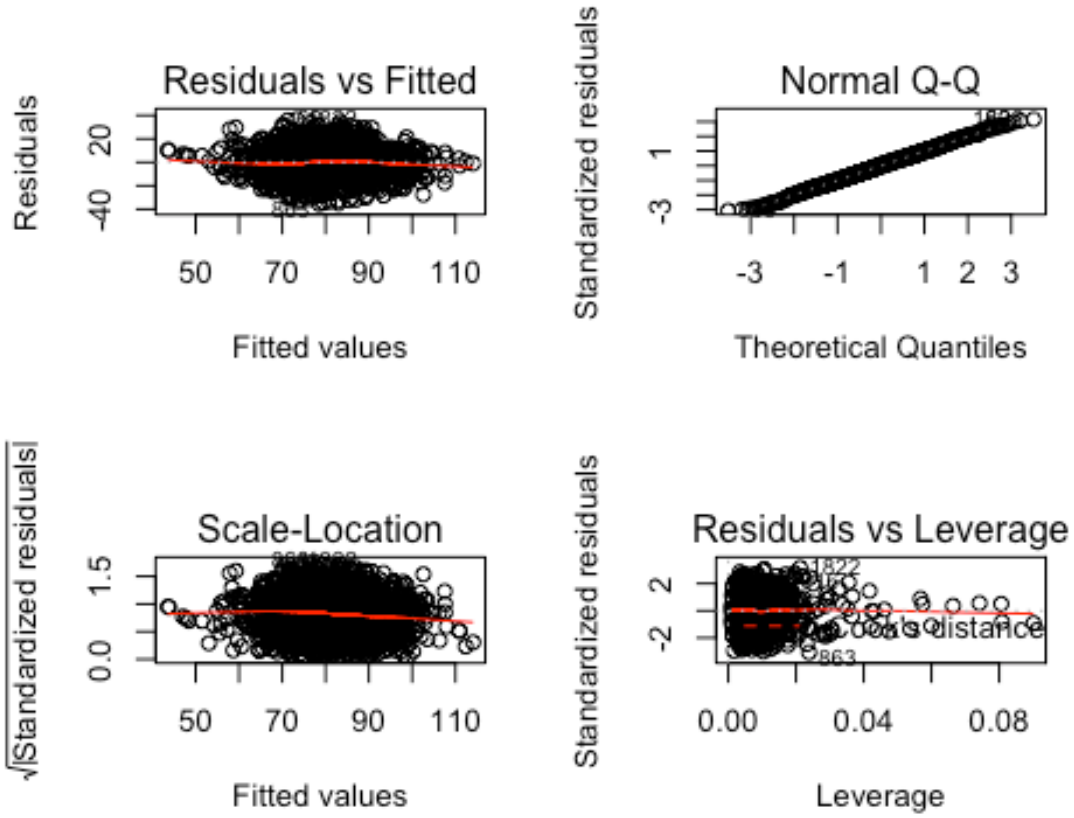

```
  summary(base_model_orig_rm)

##
## Call:
## lm(formula = target_wins ~ batting_h + batting_2b + batting_3b +
##     batting_bb + batting_so + baserun_sb + baserun_cs + pitching_hr
+
##     pitching_bb + pitching_so + fielding_e + fielding_dp +
batting_hbp_bi +
##     batting_1B + total_bases + total_bases_allowed + HR_over_OP +
##     walks_over_OP + SO_over_OP, data = outlier_treat_rm)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -34.935  -7.981   0.192   7.807  35.524
##
## Coefficients:
##                        Estimate Std. Error t value Pr(>|t|)
## (Intercept)         34.3291551  6.0483138   5.676 1.56e-08 ***
## batting_h            0.0218895  0.0136362   1.605 0.108580
## batting_2b          -0.0627070  0.0152440  -4.114 4.04e-05 ***
## batting_3b           0.0405900  0.0230262   1.763 0.078074 .
## batting_bb           0.0411676  0.0097664   4.215 2.59e-05 ***
```

```
## batting_so          -0.0160349  0.0056546  -2.836 0.004613 **
## baserun_sb           0.0605082  0.0089420   6.767 1.67e-11 ***
## baserun_cs          -0.0039357  0.0172741  -0.228 0.819795
## pitching_hr         -0.0410934  0.0200349  -2.051 0.040374 *
## pitching_bb         -0.0502903  0.0081714  -6.154 8.90e-10 ***
## pitching_so         -0.0009172  0.0052280  -0.175 0.860747
## fielding_e          -0.0441196  0.0030316 -14.553  < 2e-16 ***
## fielding_dp         -0.0993091  0.0129565  -7.665 2.65e-14 ***
## batting_hbp_bi      -4.2752572  1.1134687  -3.840 0.000127 ***
## batting_1B          -0.0287308  0.0133420  -2.153 0.031392 *
## total_bases          0.0248649  0.0047017   5.288 1.35e-07 ***
## total_bases_allowed  0.0117947  0.0021119   5.585 2.62e-08 ***
## HR_over_OP          -0.1019108  0.0816525  -1.248 0.212123
## walks_over_OP        0.0266495  0.0108719   2.451 0.014313 *
## SO_over_OP           0.0091144  0.0043901   2.076 0.037994 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 11.71 on 2244 degrees of freedom
## Multiple R-squared:  0.3642, Adjusted R-squared:  0.3588
## F-statistic: 67.64 on 19 and 2244 DF,  p-value: < 2.2e-16
```

```
  paste('MSE equal ', mse(base_model_orig_rm))
```

```
## [1] "MSE equal  136.021498431333"
```

```
base_model_lp_rm <-
  lm(target_wins ~ batting_2b + batting_3b + batting_bb + batting_so +
baserun_sb + pitching_bb +
       fielding_e + fielding_dp + batting_hbp_bi + total_bases +
total_bases_allowed + walks_over_OP + SO_over_OP, data =
outlier_treat_rm)
  par(mfrow = c(2, 2))
  plot(base_model_lp_rm)
```



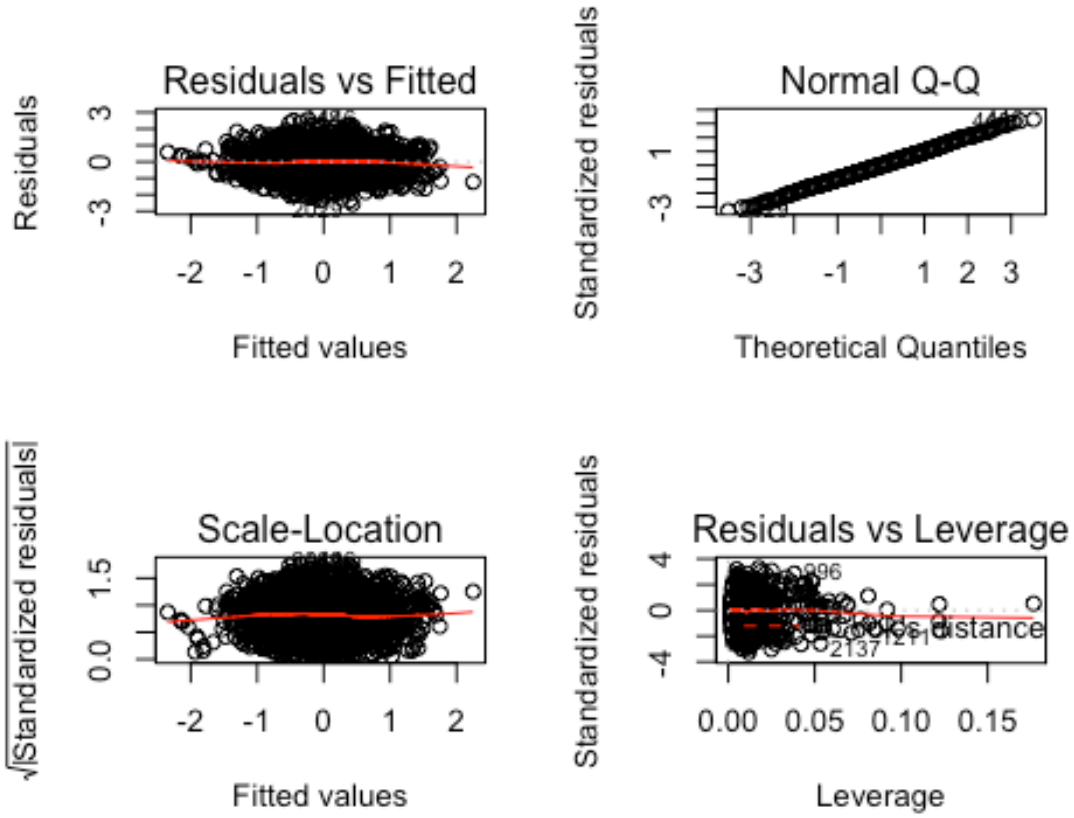

```
  summary(base_model_lp_rm)

##
## Call:
## lm(formula = target_wins ~ batting_2b + batting_3b + batting_bb +
##     batting_so + baserun_sb + pitching_bb + fielding_e + fielding_dp
+
##     batting_hbp_bi + total_bases + total_bases_allowed +
walks_over_OP +
##     SO_over_OP, data = outlier_treat_rm)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -35.554  -7.946   0.194   7.746  36.628
##
## Coefficients:
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)         31.423369   3.214253   9.776  < 2e-16 ***
## batting_2b          -0.034378   0.008980  -3.828 0.000133 ***
## batting_3b           0.071554   0.017427   4.106 4.17e-05 ***
## batting_bb           0.045871   0.009117   5.031 5.26e-07 ***
## batting_so          -0.016883   0.001772  -9.528  < 2e-16 ***
## baserun_sb           0.062321   0.005752  10.834  < 2e-16 ***
```

```
## pitching_bb         -0.050756   0.008068  -6.291 3.78e-10 ***
## fielding_e          -0.042639   0.002883 -14.787  < 2e-16 ***
## fielding_dp         -0.100316   0.012606  -7.958 2.75e-15 ***
## batting_hbp_bi      -3.980375   1.069020  -3.723 0.000201 ***
## total_bases          0.020573   0.002454   8.382  < 2e-16 ***
## total_bases_allowed  0.012093   0.001508   8.018 1.71e-15 ***
## walks_over_OP        0.027087   0.010490   2.582 0.009879 **
## SO_over_OP           0.009524   0.003845   2.477 0.013320 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 11.72 on 2250 degrees of freedom
## Multiple R-squared:  0.3623, Adjusted R-squared:  0.3586
## F-statistic: 98.33 on 13 and 2250 DF,  p-value: < 2.2e-16
```

```
  paste('MSE equal ', mse(base_model_lp_rm))
```

```
## [1] "MSE equal  136.418202030273"
```

```
trans_model_all_rm <-
  lm(target_wins ~ batting_h + batting_2b + batting_3b + batting_bb +
batting_so + baserun_sb + baserun_cs  + pitching_hr + pitching_bb +
pitching_so + fielding_e + fielding_dp  + batting_hbp_bi + batting_1B +
total_bases + total_bases_allowed + HR_over_OP + walks_over_OP +
SO_over_OP, data = transformed_rm)
  par(mfrow = c(2, 2))
  plot(trans_model_all_rm)
```



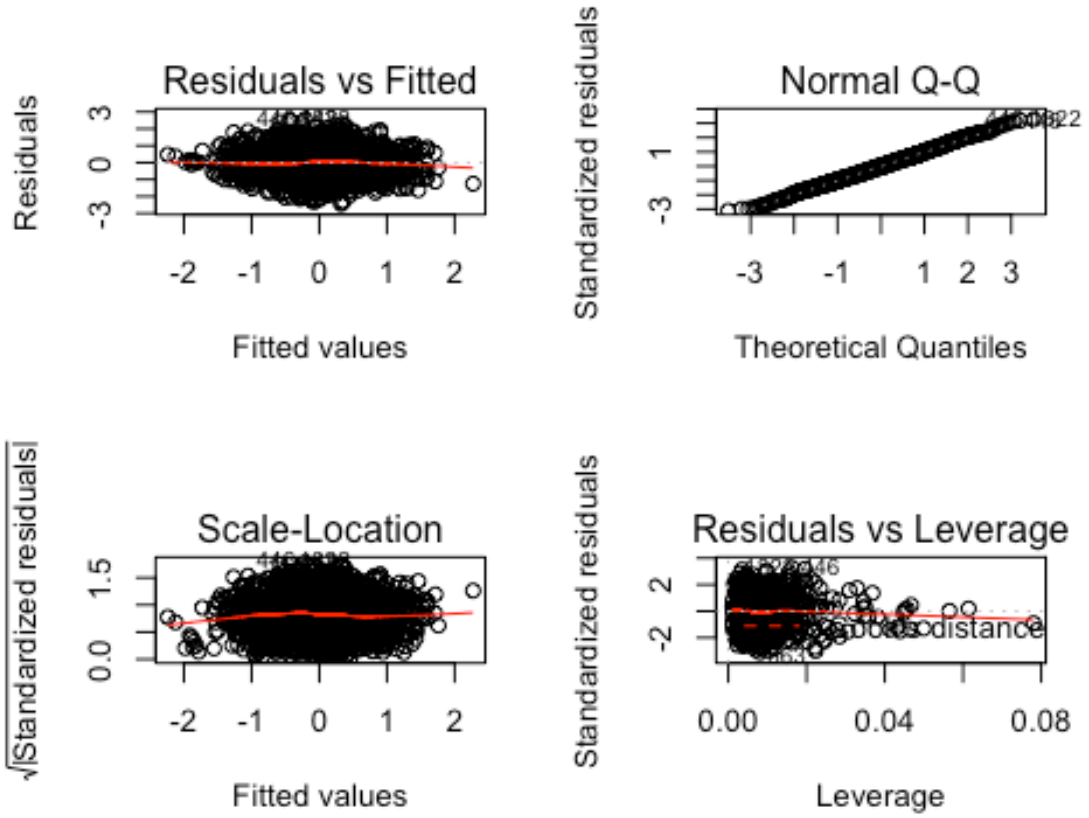

```
  summary(trans_model_all_rm)

##
## Call:
## lm(formula = target_wins ~ batting_h + batting_2b + batting_3b +
##     batting_bb + batting_so + baserun_sb + baserun_cs + pitching_hr
+
##     pitching_bb + pitching_so + fielding_e + fielding_dp +
batting_hbp_bi +
##     batting_1B + total_bases + total_bases_allowed + HR_over_OP +
##     walks_over_OP + SO_over_OP, data = transformed_rm)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.56019 -0.54095 -0.00762  0.51300  2.57939
##
## Coefficients:
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)         -0.004208   0.016568  -0.254 0.799545
## batting_h            0.074177   0.100439   0.739 0.460270
## batting_2b          -0.109759   0.043200  -2.541 0.011130 *
## batting_3b           0.150447   0.039975   3.764 0.000172 ***
## batting_bb           0.292214   0.058825   4.968 7.29e-07 ***
```

```
## batting_so           -0.310532   0.087329  -3.556 0.000384 ***
## baserun_sb            0.172427   0.047514   3.629 0.000291 ***
## baserun_cs            0.135214   0.043083   3.138 0.001720 **
## pitching_hr          -0.054814   0.078107  -0.702 0.482888
## pitching_bb          -0.283055   0.048945  -5.783 8.36e-09 ***
## pitching_so           0.001335   0.075176   0.018 0.985830
## fielding_e           -0.595271   0.038211 -15.578  < 2e-16 ***
## fielding_dp          -0.194105   0.023177  -8.375  < 2e-16 ***
## batting_hbp_bi       -0.137895   0.021106  -6.534 7.92e-11 ***
## batting_1B           -0.043597   0.079796  -0.546 0.584874
## total_bases           0.405897   0.089732   4.523 6.40e-06 ***
## total_bases_allowed   0.175163   0.059663   2.936 0.003360 **
## HR_over_OP           -0.104410   0.035925  -2.906 0.003693 **
## walks_over_OP         0.174683   0.049432   3.534 0.000418 ***
## SO_over_OP            0.031538   0.037757   0.835 0.403647
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.7881 on 2244 degrees of freedom
## Multiple R-squared:  0.3709, Adjusted R-squared:  0.3656
## F-statistic: 69.64 on 19 and 2244 DF,  p-value: < 2.2e-16
```

```
  paste('MSE equal ', mse(trans_model_all_rm))
```

```
## [1] "MSE equal  0.615551935849093"
```

```
trans_model_lp_rm <-
  lm(target_wins ~ batting_3b + batting_bb + batting_so + baserun_sb +
baserun_cs + pitching_bb + fielding_e + fielding_dp  + batting_hbp_bi +
total_bases + total_bases_allowed + walks_over_OP + SO_over_OP, data =
transformed_rm)
  par(mfrow = c(2, 2))
  plot(trans_model_lp_rm)
```




```
  summary(trans_model_lp_rm)

##
## Call:
## lm(formula = target_wins ~ batting_3b + batting_bb + batting_so +
##     baserun_sb + baserun_cs + pitching_bb + fielding_e + fielding_dp
+
##     batting_hbp_bi + total_bases + total_bases_allowed +
walks_over_OP +
##     SO_over_OP, data = transformed_rm)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.44678 -0.54976 -0.00499  0.51233  2.56018
##
## Coefficients:
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)         -0.003734   0.016609  -0.225  0.82213
## batting_3b           0.193489   0.032339   5.983 2.54e-09 ***
## batting_bb           0.305651   0.053250   5.740 1.08e-08 ***
## batting_so          -0.318539   0.029906 -10.651  < 2e-16 ***
## baserun_sb           0.190847   0.039415   4.842 1.37e-06 ***
## baserun_cs           0.126372   0.041996   3.009  0.00265 **
```

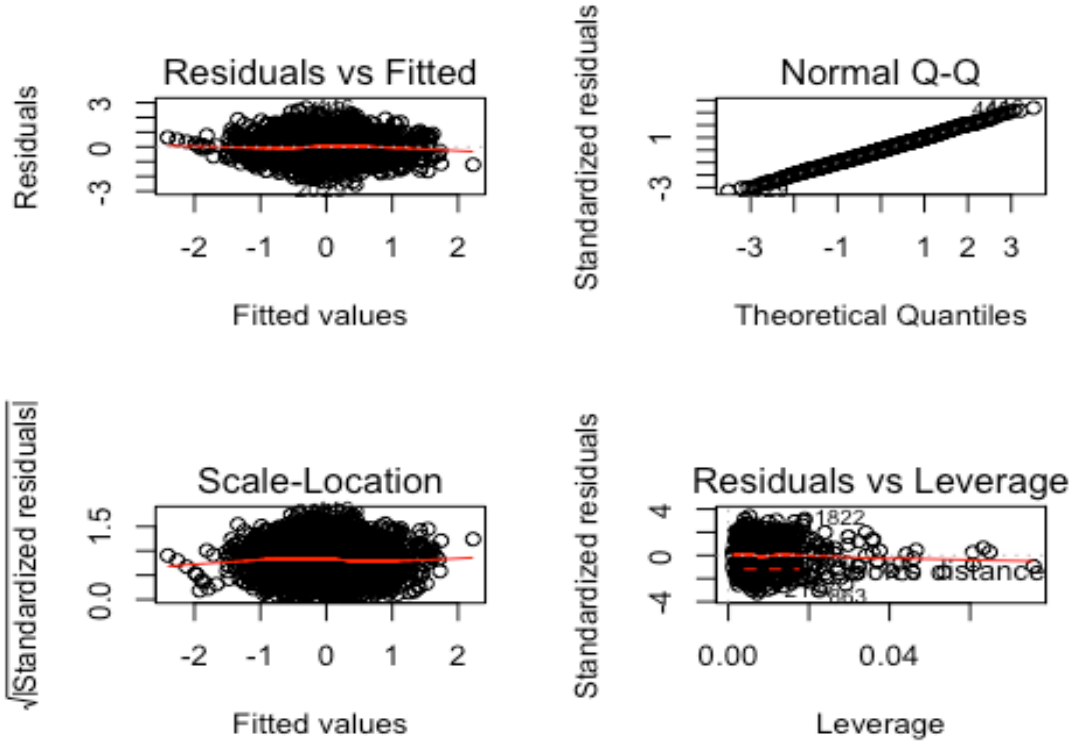
```
## pitching_bb          -0.275581   0.048522  -5.680 1.53e-08 ***
## fielding_e           -0.579511   0.037258 -15.554  < 2e-16 ***
## fielding_dp          -0.192350   0.022915  -8.394  < 2e-16 ***
## batting_hbp_bi       -0.152738   0.019816  -7.708 1.91e-14 ***
## total_bases           0.320545   0.041004   7.817 8.21e-15 ***
## total_bases_allowed   0.223549   0.042162   5.302 1.26e-07 ***
## walks_over_OP         0.156725   0.047703   3.285  0.00103 **
## SO_over_OP            0.060558   0.033262   1.821  0.06879 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.7901 on 2250 degrees of freedom
## Multiple R-squared:  0.366,  Adjusted R-squared:  0.3623
## F-statistic:  99.9 on 13 and 2250 DF,  p-value: < 2.2e-16
```

```
  paste('MSE equal ', mse(trans_model_lp_rm))
```

```
## [1] "MSE equal  0.620413235273429"
```

1. Stepwise Both direction

```
stepwise_base_model_bd_rm <- stepAIC(trans_model_all_rm, direction =
"both")

par(mfrow = c(2, 2))
  plot(stepwise_base_model_bd_rm)
```



```r
  summary(stepwise_base_model_bd_rm)
```

```
##
## Call:
## lm(formula = target_wins ~ batting_2b + batting_3b + batting_bb +
##     batting_so + baserun_sb + baserun_cs + pitching_bb + fielding_e
+
##     fielding_dp + batting_hbp_bi + total_bases + total_bases_allowed
+
##     HR_over_OP + walks_over_OP, data = transformed_rm)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.5712 -0.5427 -0.0066  0.5141  2.6560
##
## Coefficients:
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)         -0.004106   0.016556  -0.248 0.804132
## batting_2b          -0.076989   0.027057  -2.845 0.004476 **
## batting_3b           0.170960   0.032731   5.223 1.92e-07 ***
## batting_bb           0.290798   0.054075   5.378 8.33e-08 ***
## batting_so          -0.322686   0.028711 -11.239  < 2e-16 ***
## baserun_sb           0.181398   0.039890   4.547 5.72e-06 ***
## baserun_cs           0.141448   0.041390   3.417 0.000643 ***
## pitching_bb         -0.284971   0.048500  -5.876 4.84e-09 ***
## fielding_e          -0.589229   0.037362 -15.771  < 2e-16 ***
## fielding_dp         -0.194402   0.022892  -8.492  < 2e-16 ***
## batting_hbp_bi      -0.136648   0.020182  -6.771 1.63e-11 ***
## total_bases          0.388993   0.047997   8.105 8.59e-16 ***
## total_bases_allowed  0.189220   0.042893   4.411 1.08e-05 ***
## HR_over_OP          -0.101620   0.030242  -3.360 0.000792 ***
## walks_over_OP        0.161277   0.045189   3.569 0.000366 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.7876 on 2249 degrees of freedom
## Multiple R-squared:  0.3703, Adjusted R-squared:  0.3664
## F-statistic: 94.48 on 14 and 2249 DF,  p-value: < 2.2e-16
```

```r
paste('MSE equal ', mse(stepwise_base_model_bd_rm))
```

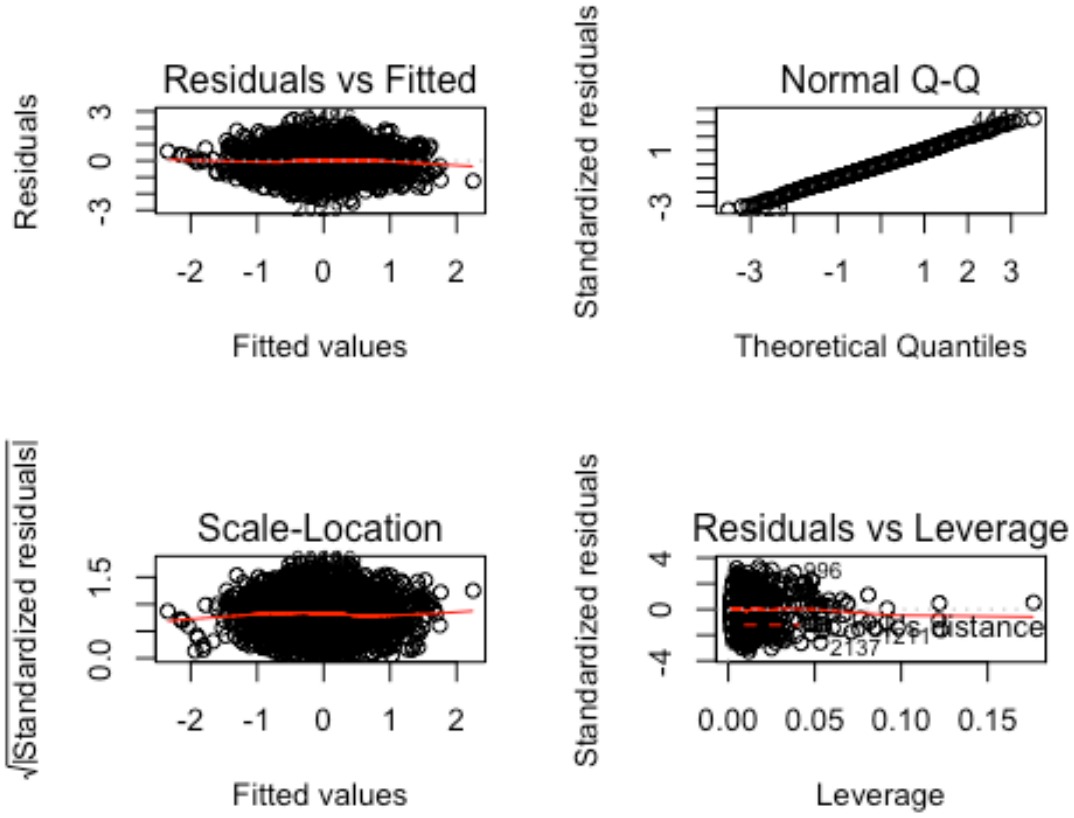
```
## [1] "MSE equal  0.616142519793743"
```

2. Forward direction

```r
stepwise_base_model_fw_rm <- stepAIC(trans_model_all_rm, direction =
"forward")
```

```
## Start:  AIC=-1058.57
## target_wins ~ batting_h + batting_2b + batting_3b + batting_bb +
##     batting_so + baserun_sb + baserun_cs + pitching_hr + pitching_bb
+
```

```
##     pitching_so + fielding_e + fielding_dp + batting_hbp_bi +
##     batting_1B + total_bases + total_bases_allowed + HR_over_OP +
##     walks_over_OP + SO_over_OP

par(mfrow = c(2, 2))
  plot(stepwise_base_model_fw_rm)
```




```
  summary(stepwise_base_model_fw_rm)

##
## Call:
## lm(formula = target_wins ~ batting_h + batting_2b + batting_3b +
##     batting_bb + batting_so + baserun_sb + baserun_cs + pitching_hr
+
##     pitching_bb + pitching_so + fielding_e + fielding_dp +
batting_hbp_bi +
##     batting_1B + total_bases + total_bases_allowed + HR_over_OP +
##     walks_over_OP + SO_over_OP, data = transformed_rm)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.56019 -0.54095 -0.00762  0.51300  2.57939
##
## Coefficients:
```

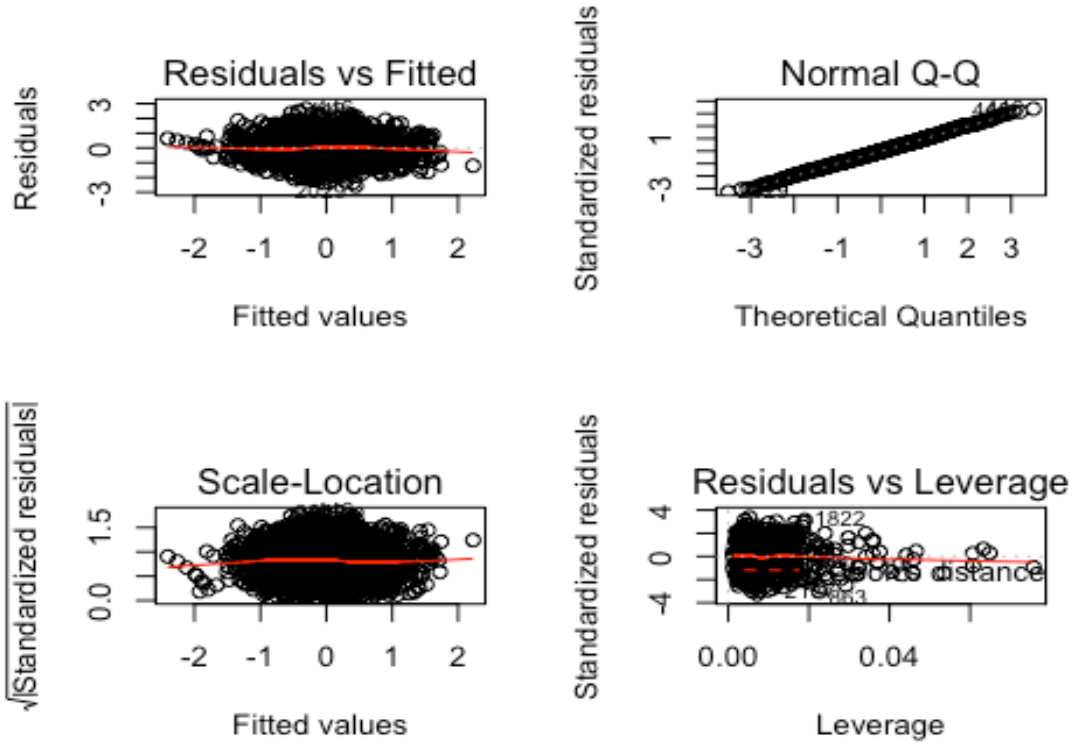
```
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)         -0.004208   0.016568  -0.254 0.799545
## batting_h            0.074177   0.100439   0.739 0.460270
## batting_2b          -0.109759   0.043200  -2.541 0.011130 *
## batting_3b           0.150447   0.039975   3.764 0.000172 ***
## batting_bb           0.292214   0.058825   4.968 7.29e-07 ***
## batting_so          -0.310532   0.087329  -3.556 0.000384 ***
## baserun_sb           0.172427   0.047514   3.629 0.000291 ***
## baserun_cs           0.135214   0.043083   3.138 0.001720 **
## pitching_hr         -0.054814   0.078107  -0.702 0.482888
## pitching_bb         -0.283055   0.048945  -5.783 8.36e-09 ***
## pitching_so          0.001335   0.075176   0.018 0.985830
## fielding_e          -0.595271   0.038211 -15.578  < 2e-16 ***
## fielding_dp         -0.194105   0.023177  -8.375  < 2e-16 ***
## batting_hbp_bi      -0.137895   0.021106  -6.534 7.92e-11 ***
## batting_1B          -0.043597   0.079796  -0.546 0.584874
## total_bases          0.405897   0.089732   4.523 6.40e-06 ***
## total_bases_allowed  0.175163   0.059663   2.936 0.003360 **
## HR_over_OP          -0.104410   0.035925  -2.906 0.003693 **
## walks_over_OP        0.174683   0.049432   3.534 0.000418 ***
## SO_over_OP           0.031538   0.037757   0.835 0.403647
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.7881 on 2244 degrees of freedom
## Multiple R-squared:  0.3709, Adjusted R-squared:  0.3656
## F-statistic: 69.64 on 19 and 2244 DF,  p-value: < 2.2e-16
```

```
paste('MSE equal ', mse(stepwise_base_model_fw_rm))
```

```
## [1] "MSE equal  0.615551935849093"
```

3. Backwards direction

```
stepwise_base_model_bw_rm <- stepAIC(trans_model_all_rm, direction =
"backward")

par(mfrow = c(2, 2))
  plot(stepwise_base_model_bw_rm)
```




```
  summary(stepwise_base_model_bw_rm)

##
## Call:
## lm(formula = target_wins ~ batting_2b + batting_3b + batting_bb +
##     batting_so + baserun_sb + baserun_cs + pitching_bb + fielding_e
+
##     fielding_dp + batting_hbp_bi + total_bases + total_bases_allowed
+
##     HR_over_OP + walks_over_OP, data = transformed_rm)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.5712 -0.5427 -0.0066  0.5141  2.6560
##
## Coefficients:
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)         -0.004106   0.016556  -0.248 0.804132
## batting_2b          -0.076989   0.027057  -2.845 0.004476 **
## batting_3b           0.170960   0.032731   5.223 1.92e-07 ***
## batting_bb           0.290798   0.054075   5.378 8.33e-08 ***
## batting_so          -0.322686   0.028711 -11.239  < 2e-16 ***
## baserun_sb           0.181398   0.039890   4.547 5.72e-06 ***
## baserun_cs           0.141448   0.041390   3.417 0.000643 ***
## pitching_bb         -0.284971   0.048500  -5.876 4.84e-09 ***
## fielding_e          -0.589229   0.037362 -15.771  < 2e-16 ***
```

```
## fielding_dp         -0.194402   0.022892  -8.492  < 2e-16 ***
## batting_hbp_bi      -0.136648   0.020182  -6.771 1.63e-11 ***
## total_bases          0.388993   0.047997   8.105 8.59e-16 ***
## total_bases_allowed  0.189220   0.042893   4.411 1.08e-05 ***
## HR_over_OP          -0.101620   0.030242  -3.360 0.000792 ***
## walks_over_OP        0.161277   0.045189   3.569 0.000366 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.7876 on 2249 degrees of freedom
## Multiple R-squared:  0.3703, Adjusted R-squared:  0.3664
## F-statistic: 94.48 on 14 and 2249 DF,  p-value: < 2.2e-16
```

```
paste('MSE equal ', mse(stepwise_base_model_bw_rm))
```

```
## [1] "MSE equal  0.616142519793743"
```

Conclusion

It definitely made a difference when the transformation was applied. The only problem that I faced was that my prediction when in the thousands if I used the models created on the transformed data set. After paying a close attention on the Cook's distance for the models' residual, I removed certain observation that led to an improved model.

After testing more than 10 models, using different techniques and transformation, I settled with a model built after I capped outliers, removed variables causing multicollinearity, variables with low p-value, and removed influencial observations.

Here is the model `base_model_lp_rm`: `Target Wins = 32.157432 - 0.035903 * moneyball_imp_test$batting_2b + 0.068862 * moneyball_imp_test$batting_3b + 0.044466 * moneyball_imp_test$batting_bb - 0.016966 * moneyball_imp_test$batting_so + 0.060647 * moneyball_imp_test$baserun_sb - 0.050230 * moneyball_imp_test$pitching_bb - 0.043364 * moneyball_imp_test$fielding_e - 0.105258 * moneyball_imp_test$fielding_dp - 4.089404 * moneyball_imp_test$batting_hbp_bi + 0.021326 * moneyball_imp_test$total_bases + 0.011782 * moneyball_imp_test$total_bases_allowed + 0.023997 * moneyball_imp_test$walks_over_OP + 0.008021 * moneyball_imp_test$SO_over_OP`

When looking at the Rsquared and Adjusted Rsquared together with the residual plots, the `base_model_lp_rm` model was not the best model. The stepwise model after removing influencial observation were the best model, but when tested on the test dataset, the numbers were in the thousands. It could be a step I missed, but `base_model_lp_rm` will be my final model.