621 HW1

Team 1, Justin Hsi, Tony Mei, Lin Li, Santosh Cheruku, Michael ODonnell

September 13, 2020

1. Data Exploration

The "moneyball" training data set contains 2276 rows and 17 columns, including variables such as TARGET_WINS, TEAM_BATTING, TEAN_BASERUN, etc. The variables are thought to have a positive or negative effect on the number of games the baseball team won during the season. Running a summary() function on the data set, we are able to get the mean, median, first and third quartile and the minimum and maximum values for each variable. We included a correlation plot and pairs plot to visualize the relationship among the variables. Histograms were created for each type of hits to observe the normality of the variables. We explored the structure of the variables for both the training and evaluation data sets and finally observed how TARGET_WINS are affected by other factors. Interestingly, the number of wins seems positively correlated with all hits by batters except triples by batters, which the correlation plot shows as slightly negatively correlated. One potential explanation may be that getting triples, while good, is actually always worse than getting homeruns, so having a large number of triples may actually mean the team is just barely falling short. Nothing from the correlation plot can be used to conclude this, but it is something that can be investigated further in the future. Also surprising is that stolen bases barely has any positive correlation with wins, but that may just be due to the rarity of the event (stolen bases). TEAM_PITCHING_H, TEAM_PITCHING_BB, and TEAM_PITCHING_HR surprisingly shows a positive correlation with team wins, but maybe this alludes to having good batters and getting runs being more important to winning than stopping the opponent from getting runs. Similarly, TEAM_PITCHING_SO and TEAM_PITCHING_DP are events of denying the opponent runs, but they show a negative correlation with number of wins and may also point to getting runs for your team as the key to winning.

2. Data Preparation

We addressed issues with imperfect data before building models or performing statistical analysis. We observed that several variables have high numbers of NA or missing values. TEAM_BATTING_HBP has the highest number of missing cases i.e., 2085 (~ 90%). Based on the variable definitions given in the assignment, it seemed reasonable that NA values meant that there were no occurrences of that event. So we chose to create additional columns flagging whether the original variable was NA or not (1 if NA, 0 if not NA), and then filled NAs with 0.

3. Build Models

First we built a model using the backward elimination process. In this process, we rejected predictors with p-value greater than 0.05 and stopped after all remaining model predictors had p-values of less than 0.05. For our second model we decided to use stepwise selection. Stepwise selection uses an automated process of building a model by adding or removing predictors repeatedly based on an improvement of a criterion (Akaike information criterion in our case). We noticed one of the variables, TEAM_PITCHING_SO, had a p-value greater than 0.05 in the second model so we decided to build a third model using stepwise regression with the TEAM_PITCHING_SO predictor removed. The third model's R squared dropped slightly, so we decided to stick with our second model.

4. Select Models

Out of the three models we created, the second model with stepwise selection was the best of the three. The Adjusted R squared is 0.4098 which translates to approximately 41% of variation in Target Wins can be explained by our model. The F statistic tells us if there is a relationship between the dependent and independent variables we are testing. Generally, a large F indicates a stronger relationship and we have 113.9. The normal quantile quantile plot for residuals displays an approximately straight line so the residuals are approximately normally distributed. However, there is slight deviation at the extreme values, meaning our model does have a bit of trouble predicting a very high or low number of wins accurately. The MSE is 743.6606. Using this model we were able to make predictions for the test dataset. Finally, we made a histogram of wins from the training and evaluation set to see if the prediction distribution looked fairly similar to the training distribution, which it does.

Appendix

```r
# load required packages
library(ggplot2)
library(dplyr)
```

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

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

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

```r
#library(tidyr)
library(corrplot)
```

```
## corrplot 0.84 loaded
```

```r
library(MASS)
```

```
##
## Attaching package: 'MASS'

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

```r
library(caret)
```

```
## Loading required package: lattice
```

```r
library(RCurl)
```

```r
# Loading the data
git_dir <- 'https://raw.githubusercontent.com/san123i/CUNY/master/Semester4/621-BusinessAnalytics/HW1/da
train_df = read.csv(paste(git_dir, "/moneyball-training-data.csv", sep=""))
test_df = read.csv(paste(git_dir, "/moneyball-evaluation-data.csv", sep = ""))
```

1. Data Exploration

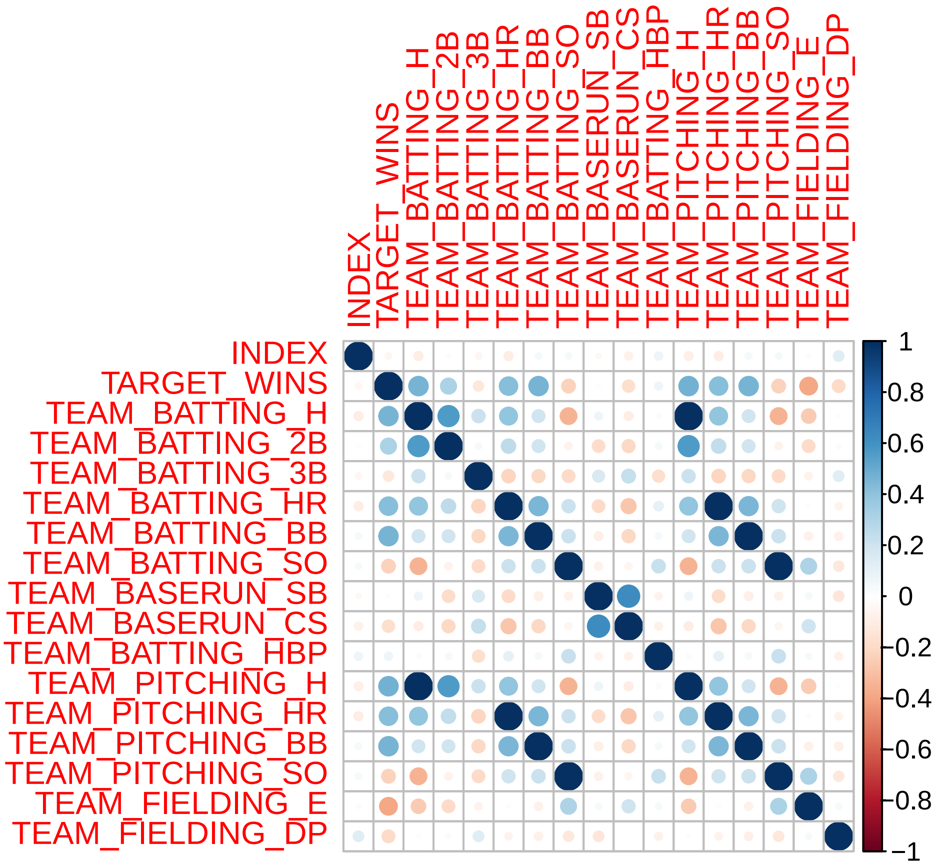
See a summary of each column in the train_dfing set

```r
# view a summary of all columns
summary(train_df)
```

```
##      INDEX         TARGET_WINS     TEAM_BATTING_H TEAM_BATTING_2B
##  Min.   :   1.0   Min.   :  0.00   Min.   : 891   Min.   : 69.0  
##  1st Qu.: 630.8   1st Qu.: 71.00   1st Qu.:1383   1st Qu.:208.0  
##  Median :1270.5   Median : 82.00   Median :1454   Median :238.0  
##  Mean   :1268.5   Mean   : 80.79   Mean   :1469   Mean   :241.2  
##  3rd Qu.:1915.5   3rd Qu.: 92.00   3rd Qu.:1537   3rd Qu.:273.0  
##  Max.   :2535.0   Max.   :146.00   Max.   :2554   Max.   :458.0  
##                                                                  
##  TEAM_BATTING_3B  TEAM_BATTING_HR  TEAM_BATTING_BB TEAM_BATTING_SO
##  Min.   :  0.00   Min.   :  0.00   Min.   :  0.0   Min.   :   0.0  
##  1st Qu.: 34.00   1st Qu.: 42.00   1st Qu.:451.0   1st Qu.: 548.0  
##  Median : 47.00   Median :102.00   Median :512.0   Median : 750.0  
##  Mean   : 55.25   Mean   : 99.61   Mean   :501.6   Mean   : 735.6  
##  3rd Qu.: 72.00   3rd Qu.:147.00   3rd Qu.:580.0   3rd Qu.: 930.0  
##  Max.   :223.00   Max.   :264.00   Max.   :878.0   Max.   :1399.0  
##                                                    NA's   :102     
##  TEAM_BASERUN_SB TEAM_BASERUN_CS TEAM_BATTING_HBP TEAM_PITCHING_H
##  Min.   :  0.0   Min.   :  0.0   Min.   :29.00    Min.   : 1137  
##  1st Qu.: 66.0   1st Qu.: 38.0   1st Qu.:50.50    1st Qu.: 1419  
##  Median :101.0   Median : 49.0   Median :58.00    Median : 1518  
##  Mean   :124.8   Mean   : 52.8   Mean   :59.36    Mean   : 1779  
##  3rd Qu.:156.0   3rd Qu.: 62.0   3rd Qu.:67.00    3rd Qu.: 1682  
##  Max.   :697.0   Max.   :201.0   Max.   :95.00    Max.   :30132  
##  NA's   :131     NA's   :772     NA's   :2085                    
##  TEAM_PITCHING_HR TEAM_PITCHING_BB TEAM_PITCHING_SO  TEAM_FIELDING_E
##  Min.   :  0.0    Min.   :   0.0   Min.   :    0.0   Min.   :  65.0  
##  1st Qu.: 50.0    1st Qu.: 476.0   1st Qu.:  615.0   1st Qu.: 127.0  
##  Median :107.0    Median : 536.5   Median :  813.5   Median : 159.0  
##  Mean   :105.7    Mean   : 553.0   Mean   :  817.7   Mean   : 246.5  
##  3rd Qu.:150.0    3rd Qu.: 611.0   3rd Qu.:  968.0   3rd Qu.: 249.2  
##  Max.   :343.0    Max.   :3645.0   Max.   :19278.0   Max.   :1898.0  
##                                    NA's   :102                       
```

```
##  TEAM_FIELDING_DP
##  Min.   : 52.0
##  1st Qu.:131.0
##  Median :149.0
##  Mean   :146.4
##  3rd Qu.:164.0
##  Max.   :228.0
##  NA's   :286
```

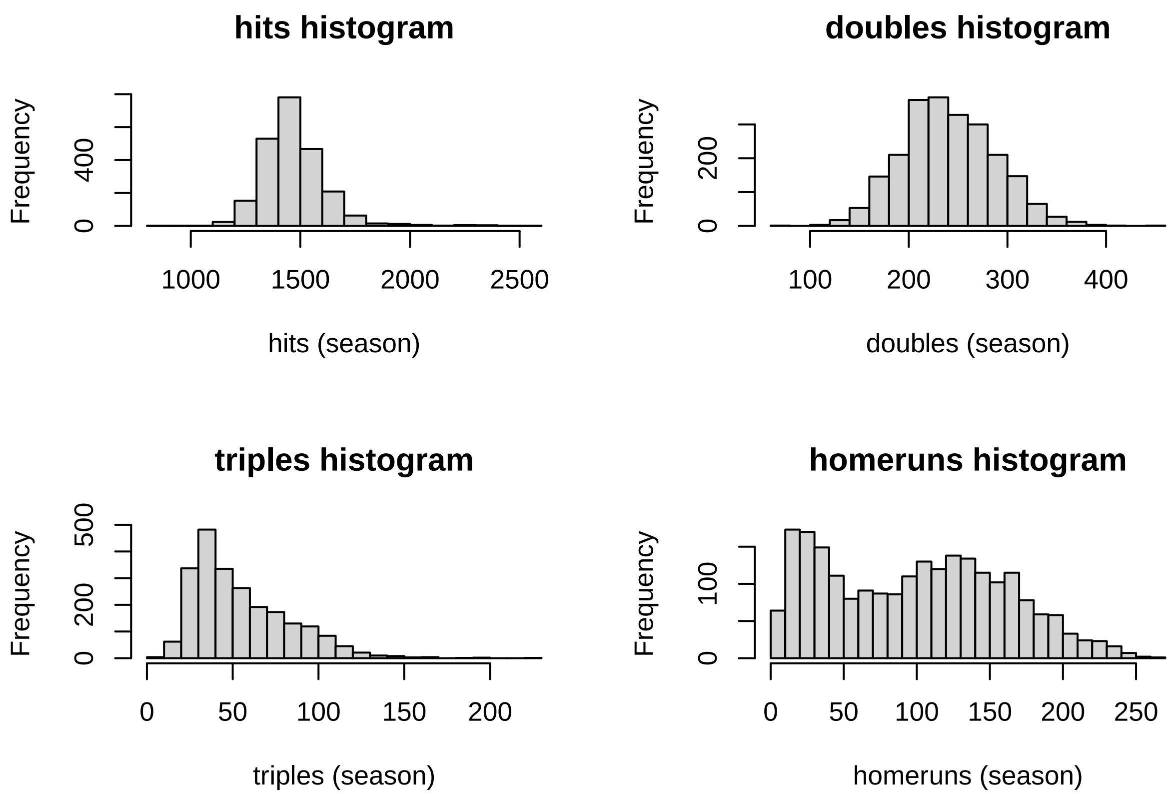
```r
# Correlations
cor_train = cor(train_df,  use = "na.or.complete")
corrplot(cor_train)
```



For types of hits, see a histogram of each

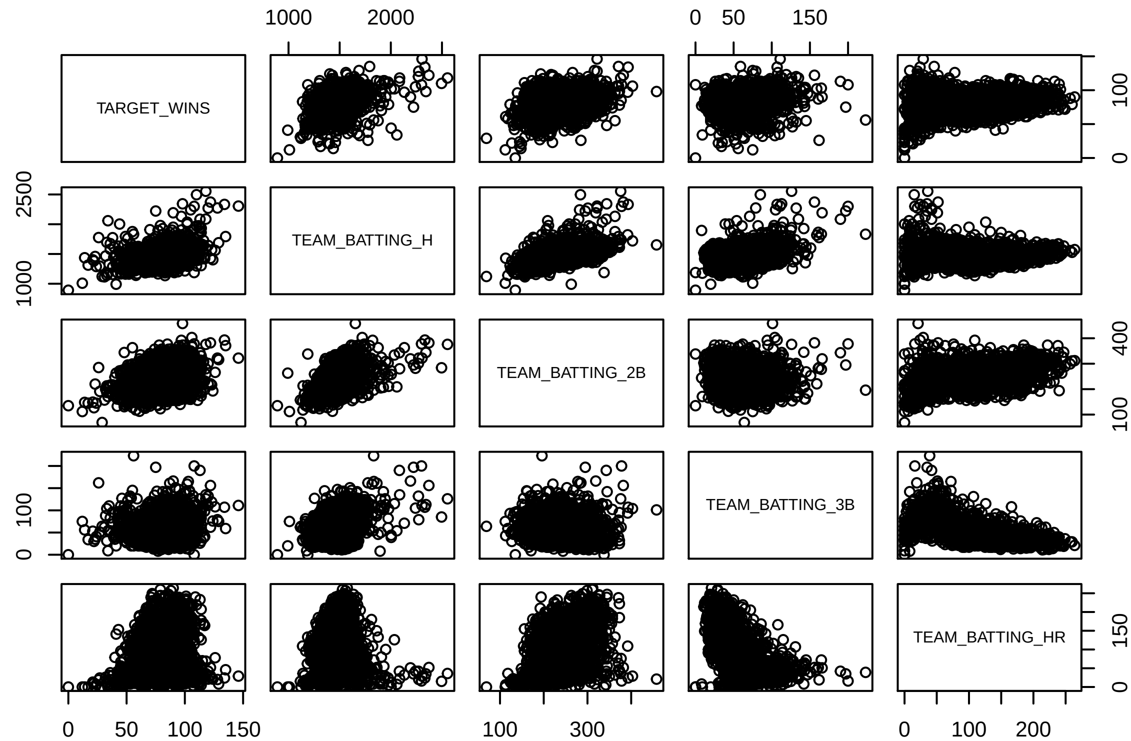
```r
par(mfrow=c(2,2))
hist(train_df$TEAM_BATTING_H,
     main = "hits histogram", xlab = "hits (season)",
     breaks = 20)
hist(train_df$TEAM_BATTING_2B,
     main = "doubles histogram", xlab = "doubles (season)",
     breaks = 20)
hist(train_df$TEAM_BATTING_3B,
     main = "triples histogram", xlab = "triples (season)",
     breaks = 20)
hist(train_df$TEAM_BATTING_HR,
```

```
     main = "homeruns histogram", xlab = "homeruns (season)",
     breaks = 20)
```




```
par(mfrow=c(1,1))
```

```
pairs(~ TARGET_WINS + TEAM_BATTING_H + TEAM_BATTING_2B
      + TEAM_BATTING_3B + TEAM_BATTING_HR, data = train_df)
```



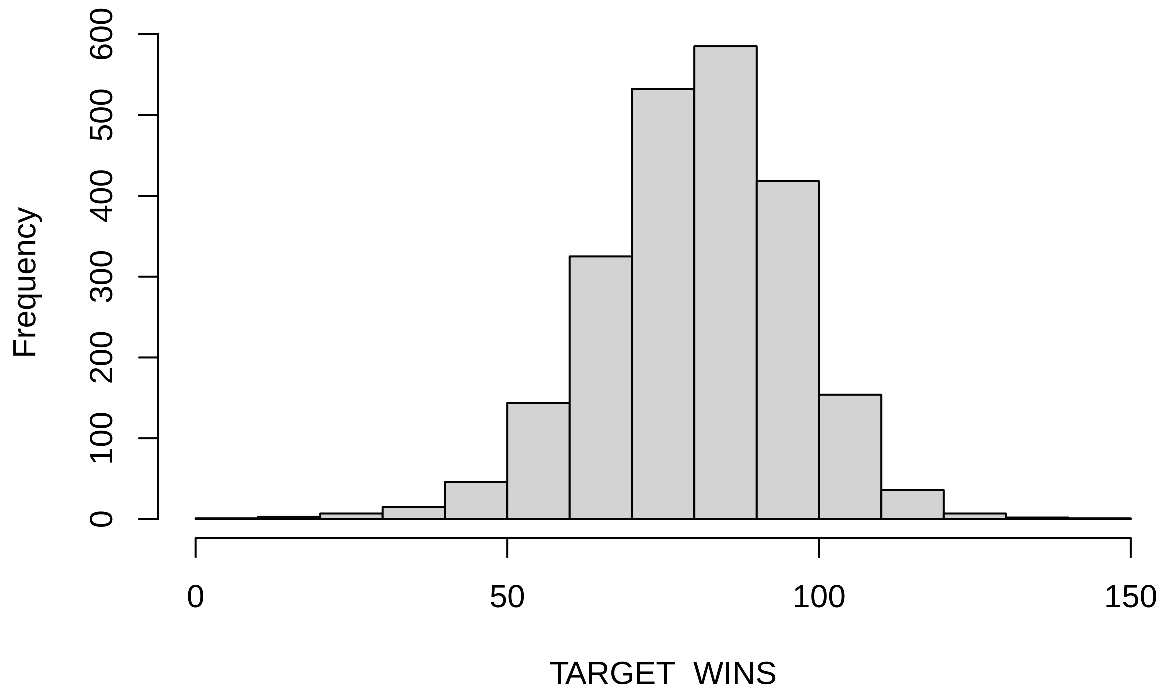

```r
# look at the structure of the variables
str(train_df)
```

```
## 'data.frame':    2276 obs. of  17 variables:
##  $ INDEX           : int  1 2 3 4 5 6 7 8 11 12 ...
##  $ TARGET_WINS     : int  39 70 86 70 82 75 80 85 86 76 ...
##  $ TEAM_BATTING_H  : int  1445 1339 1377 1387 1297 1279 1244 1273 1391 1271 ...
##  $ TEAM_BATTING_2B : int  194 219 232 209 186 200 179 171 197 213 ...
##  $ TEAM_BATTING_3B : int  39 22 35 38 27 36 54 37 40 18 ...
##  $ TEAM_BATTING_HR : int  13 190 137 96 102 92 122 115 114 96 ...
##  $ TEAM_BATTING_BB : int  143 685 602 451 472 443 525 456 447 441 ...
##  $ TEAM_BATTING_SO : int  842 1075 917 922 920 973 1062 1027 922 827 ...
##  $ TEAM_BASERUN_SB : int  NA 37 46 43 49 107 80 40 69 72 ...
##  $ TEAM_BASERUN_CS : int  NA 28 27 30 39 59 54 36 27 34 ...
##  $ TEAM_BATTING_HBP: int  NA NA NA NA NA NA NA NA NA NA ...
##  $ TEAM_PITCHING_H : int  9364 1347 1377 1396 1297 1279 1244 1281 1391 1271 ...
##  $ TEAM_PITCHING_HR: int  84 191 137 97 102 92 122 116 114 96 ...
##  $ TEAM_PITCHING_BB: int  927 689 602 454 472 443 525 459 447 441 ...
##  $ TEAM_PITCHING_SO: int  5456 1082 917 928 920 973 1062 1033 922 827 ...
##  $ TEAM_FIELDING_E : int  1011 193 175 164 138 123 136 112 127 131 ...
##  $ TEAM_FIELDING_DP: int  NA 155 153 156 168 149 186 136 169 159 ...
```

```r
str(eval)
```

```
## function (expr, envir = parent.frame(), enclos = if (is.list(envir) ||
##     is.pairlist(envir)) parent.frame() else baseenv())
```

```r
# lets observe how targets_win are effected by other factors
hist(train_df$TARGET_WINS,xlab="TARGET_WINS",main="")
```

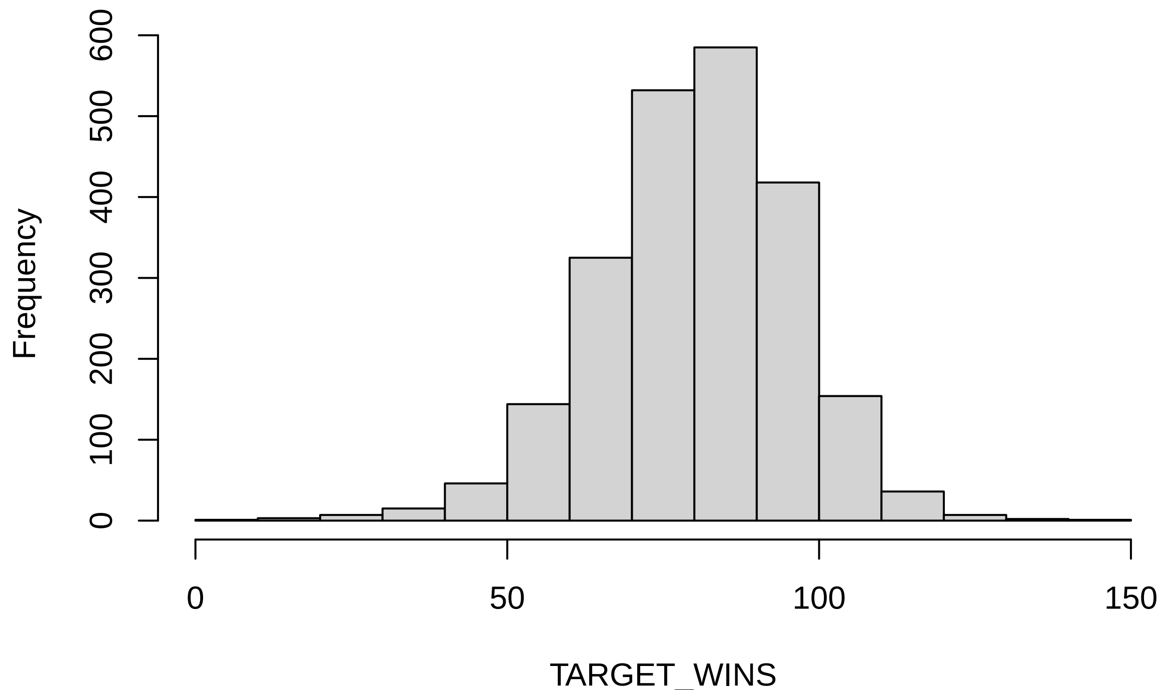



```r
# we have no TARGET_WINS from eval
# hist(eval$TARGET_WINS,xlab="TARGET_WINS",main="")
```

2. Data Preparation

1. We are told everything is standardized to match a 162 game season, so it is my preference to make TARGET_WINS a decimal of 162

```r
train_target_wins = train_df$TARGET_WINS
#train_df$TARGET_WINS = train_df$TARGET_WINS/162.
# TARGET_WINS now a decimal of games won in 162 game season
hist(train_df$TARGET_WINS,xlab="TARGET_WINS",main="")
```




```r
str(train_df)
```

```
## 'data.frame':    2276 obs. of  17 variables:
##  $ INDEX           : int  1 2 3 4 5 6 7 8 11 12 ...
##  $ TARGET_WINS     : int  39 70 86 70 82 75 80 85 86 76 ...
##  $ TEAM_BATTING_H  : int  1445 1339 1377 1387 1297 1279 1244 1273 1391 1271 ...
##  $ TEAM_BATTING_2B : int  194 219 232 209 186 200 179 171 197 213 ...
##  $ TEAM_BATTING_3B : int  39 22 35 38 27 36 54 37 40 18 ...
##  $ TEAM_BATTING_HR : int  13 190 137 96 102 92 122 115 114 96 ...
##  $ TEAM_BATTING_BB : int  143 685 602 451 472 443 525 456 447 441 ...
##  $ TEAM_BATTING_SO : int  842 1075 917 922 920 973 1062 1027 922 827 ...
##  $ TEAM_BASERUN_SB : int  NA 37 46 43 49 107 80 40 69 72 ...
##  $ TEAM_BASERUN_CS : int  NA 28 27 30 39 59 54 36 27 34 ...
##  $ TEAM_BATTING_HBP: int  NA NA NA NA NA NA NA NA NA NA ...
##  $ TEAM_PITCHING_H : int  9364 1347 1377 1396 1297 1279 1244 1281 1391 1271 ...
##  $ TEAM_PITCHING_HR: int  84 191 137 97 102 92 122 116 114 96 ...
##  $ TEAM_PITCHING_BB: int  927 689 602 454 472 443 525 459 447 441 ...
##  $ TEAM_PITCHING_SO: int  5456 1082 917 928 920 973 1062 1033 922 827 ...
##  $ TEAM_FIELDING_E : int  1011 193 175 164 138 123 136 112 127 131 ...
##  $ TEAM_FIELDING_DP: int  NA 155 153 156 168 149 186 136 169 159 ...
```

2. Assuming that everything that is NA can be filled by 0 based on the description of variables, create columns flagging if original values were NA (e.g. create TEAM_BATTING_HBP_NA column and value is 1 if TEAM_BATTING_HBP is NA and 0 otherwise meaning it wasn't NA and had a value. Do this for all columns)

```
#
has_NA = names(which(sapply(train_df, anyNA)))
for (col in has_NA)
{
   new_col = (paste(col,"_NA", sep=""))
   train_df[,new_col] = as.numeric(is.na(train_df[,col]))
   test_df[,new_col] = as.numeric(is.na(test_df[,col]))
}
train_df[is.na(train_df)] = 0
test_df[is.na(test_df)] = 0
```

3. Build Models

```
# set seed for reproducibility
n_records = nrow(train_df)
set.seed(1)
```

Model 1 - Backward Elimination Process

We will be rejecting predictors with p-value greater than 0.05 with the backward elimination process. We will stop after all the predictors are less than 0.05

```
model <- lm(TARGET_WINS ~ TEAM_BATTING_H + TEAM_BATTING_2B + TEAM_BATTING_3B + TEAM_BATTING_HR + TEAM_BA
summary(train_df)
```

```
##      INDEX         TARGET_WINS     TEAM_BATTING_H TEAM_BATTING_2B
##  Min.   :   1.0   Min.   :  0.00   Min.   : 891   Min.   : 69.0  
##  1st Qu.: 630.8   1st Qu.: 71.00   1st Qu.:1383   1st Qu.:208.0  
##  Median :1270.5   Median : 82.00   Median :1454   Median :238.0  
##  Mean   :1268.5   Mean   : 80.79   Mean   :1469   Mean   :241.2  
##  3rd Qu.:1915.5   3rd Qu.: 92.00   3rd Qu.:1537   3rd Qu.:273.0  
##  Max.   :2535.0   Max.   :146.00   Max.   :2554   Max.   :458.0  
##  TEAM_BATTING_3B  TEAM_BATTING_HR  TEAM_BATTING_BB TEAM_BATTING_SO 
##  Min.   :  0.00   Min.   :  0.00   Min.   :  0.0   Min.   :   0.0  
##  1st Qu.: 34.00   1st Qu.: 42.00   1st Qu.:451.0   1st Qu.: 524.0  
##  Median : 47.00   Median :102.00   Median :512.0   Median : 728.0  
##  Mean   : 55.25   Mean   : 99.61   Mean   :501.6   Mean   : 702.6  
##  3rd Qu.: 72.00   3rd Qu.:147.00   3rd Qu.:580.0   3rd Qu.: 925.0  
##  Max.   :223.00   Max.   :264.00   Max.   :878.0   Max.   :1399.0  
##  TEAM_BASERUN_SB TEAM_BASERUN_CS  TEAM_BATTING_HBP TEAM_PITCHING_H
##  Min.   :  0.0   Min.   :  0.00   Min.   : 0.000   Min.   : 1137  
##  1st Qu.: 60.0   1st Qu.:  0.00   1st Qu.: 0.000   1st Qu.: 1419  
##  Median : 97.0   Median : 38.00   Median : 0.000   Median : 1518  
##  Mean   :117.6   Mean   : 34.89   Mean   : 4.981   Mean   : 1779  
##  3rd Qu.:151.0   3rd Qu.: 54.25   3rd Qu.: 0.000   3rd Qu.: 1682  
##  Max.   :697.0   Max.   :201.00   Max.   :95.000   Max.   :30132  
##  TEAM_PITCHING_HR TEAM_PITCHING_BB TEAM_PITCHING_SO  TEAM_FIELDING_E 
##  Min.   :  0.0    Min.   :   0.0   Min.   :    0.0   Min.   :  65.0  
##  1st Qu.: 50.0    1st Qu.: 476.0   1st Qu.:  587.8   1st Qu.: 127.0
```

```
##  Median :107.0    Median : 536.5   Median :  797.0   Median : 159.0  
##  Mean   :105.7    Mean   : 553.0   Mean   :  781.1   Mean   : 246.5  
##  3rd Qu.:150.0    3rd Qu.: 611.0   3rd Qu.:  957.0   3rd Qu.: 249.2  
##  Max.   :343.0    Max.   :3645.0   Max.   :19278.0   Max.   :1898.0  
##  TEAM_FIELDING_DP TEAM_BATTING_SO_NA TEAM_BASERUN_SB_NA TEAM_BASERUN_CS_NA
##  Min.   :  0.0    Min.   :0.00000    Min.   :0.00000    Min.   :0.0000    
##  1st Qu.:118.0    1st Qu.:0.00000    1st Qu.:0.00000    1st Qu.:0.0000    
##  Median :145.0    Median :0.00000    Median :0.00000    Median :0.0000    
##  Mean   :128.0    Mean   :0.04482    Mean   :0.05756    Mean   :0.3392    
##  3rd Qu.:161.2    3rd Qu.:0.00000    3rd Qu.:0.00000    3rd Qu.:1.0000    
##  Max.   :228.0    Max.   :1.00000    Max.   :1.00000    Max.   :1.0000    
##  TEAM_BATTING_HBP_NA TEAM_PITCHING_SO_NA TEAM_FIELDING_DP_NA
##  Min.   :0.0000      Min.   :0.00000     Min.   :0.0000     
##  1st Qu.:1.0000      1st Qu.:0.00000     1st Qu.:0.0000     
##  Median :1.0000      Median :0.00000     Median :0.0000     
##  Mean   :0.9161      Mean   :0.04482     Mean   :0.1257     
##  3rd Qu.:1.0000      3rd Qu.:0.00000     3rd Qu.:0.0000     
##  Max.   :1.0000      Max.   :1.00000     Max.   :1.0000
```

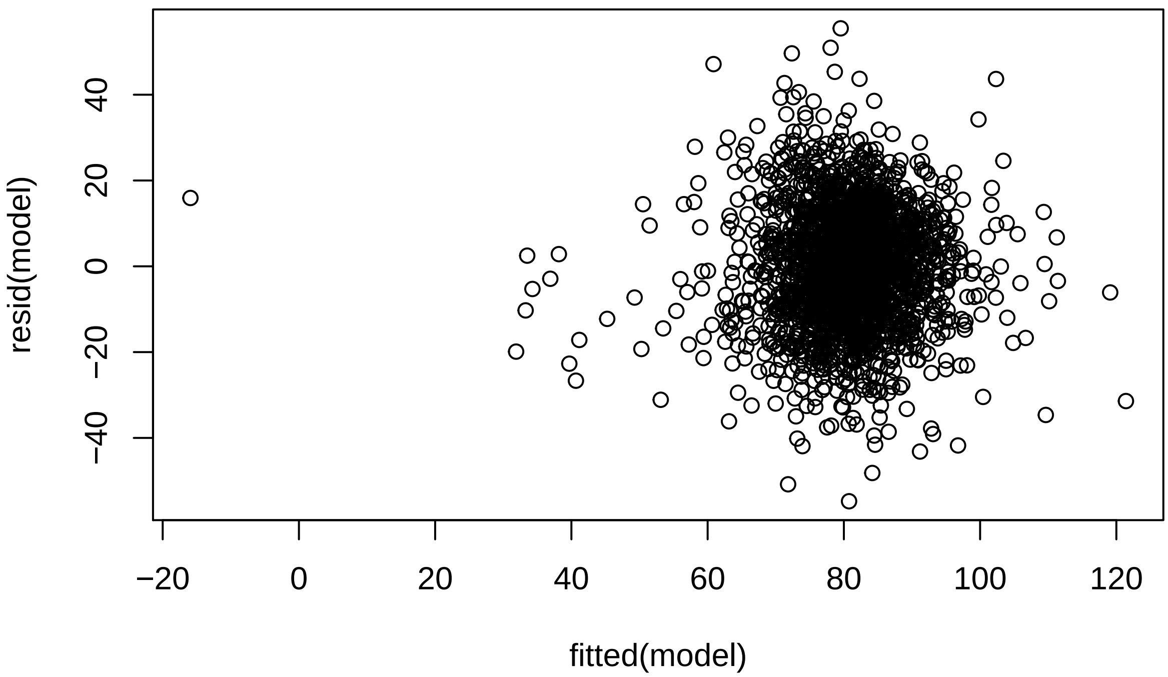
```r
model <- update(model, .~. - TEAM_BATTING_BB, data=train_df)
summary(model)
```

```
## 
## Call:
## lm(formula = TARGET_WINS ~ TEAM_BATTING_H + TEAM_BATTING_2B + 
##     TEAM_BATTING_3B + TEAM_BATTING_HR + TEAM_PITCHING_H + TEAM_PITCHING_HR + 
##     TEAM_PITCHING_BB + TEAM_FIELDING_E, data = train_df)
## 
## Residuals:
##     Min      1Q  Median      3Q     Max 
## -54.273  -8.832   0.127   8.886  55.587 
## 
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)    
## (Intercept)       6.526453   3.423988   1.906   0.0568 .  
## TEAM_BATTING_H    0.048766   0.003208  15.200  < 2e-16 ***
## TEAM_BATTING_2B  -0.026072   0.009050  -2.881   0.0040 ** 
## TEAM_BATTING_3B   0.102196   0.016708   6.116 1.12e-09 ***
## TEAM_BATTING_HR   0.054383   0.024691   2.203   0.0277 *  
## TEAM_PITCHING_H  -0.001282   0.000327  -3.922 9.05e-05 ***
## TEAM_PITCHING_HR -0.016991   0.022575  -0.753   0.4517    
## TEAM_PITCHING_BB  0.010755   0.002036   5.283 1.40e-07 ***
## TEAM_FIELDING_E  -0.016351   0.002287  -7.149 1.18e-12 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## Residual standard error: 13.48 on 2267 degrees of freedom
## Multiple R-squared:  0.2702, Adjusted R-squared:  0.2677 
## F-statistic: 104.9 on 8 and 2267 DF,  p-value: < 2.2e-16
```

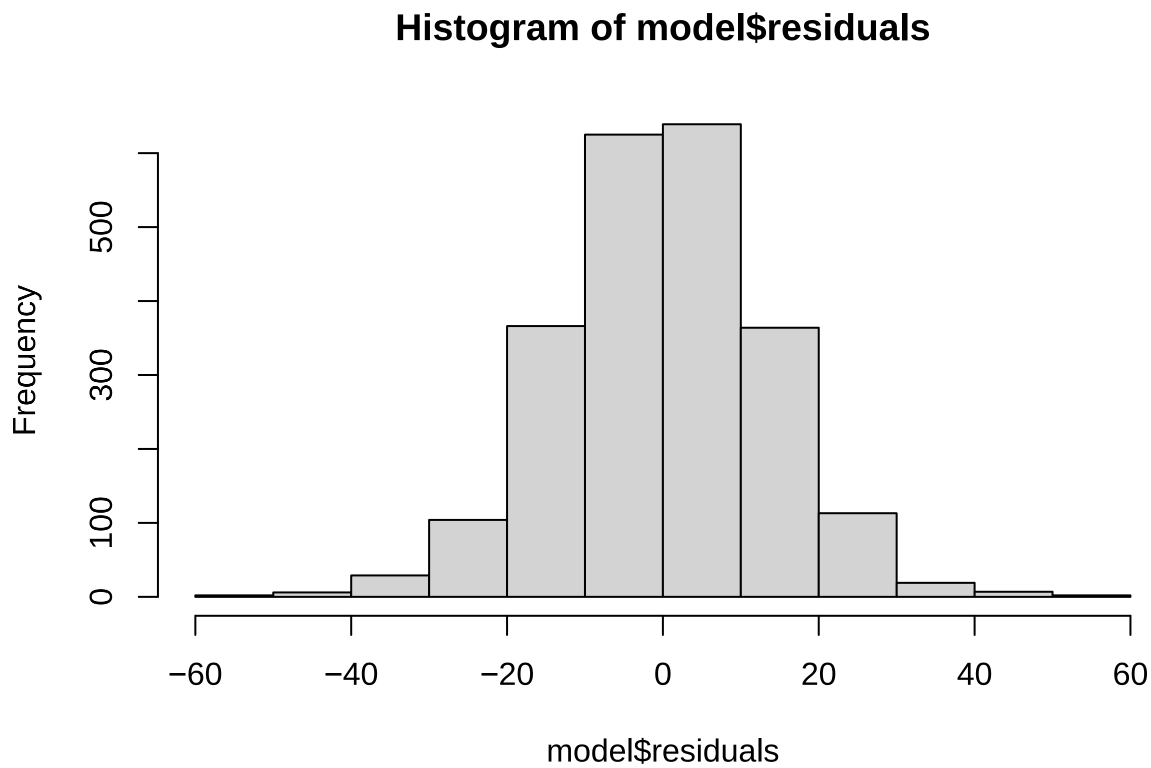
```r
model <- update(model, .~. - TEAM_PITCHING_HR, data=train_df)
summary(model)
```

```
##
## Call:
## lm(formula = TARGET_WINS ~ TEAM_BATTING_H + TEAM_BATTING_2B +
##     TEAM_BATTING_3B + TEAM_BATTING_HR + TEAM_PITCHING_H + TEAM_PITCHING_BB +
##     TEAM_FIELDING_E, data = train_df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -54.763  -8.861   0.095   8.860  55.469
##
## Coefficients:
##                    Estimate Std. Error t value Pr(>|t|)
## (Intercept)       7.2713462  3.2775220   2.219  0.02662 *
## TEAM_BATTING_H    0.0484775  0.0031849  15.221  < 2e-16 ***
## TEAM_BATTING_2B  -0.0258127  0.0090430  -2.854  0.00435 **
## TEAM_BATTING_3B   0.1010776  0.0166406   6.074 1.46e-09 ***
## TEAM_BATTING_HR   0.0366916  0.0075591   4.854 1.29e-06 ***
## TEAM_PITCHING_H  -0.0013088  0.0003251  -4.026 5.87e-05 ***
## TEAM_PITCHING_BB  0.0103207  0.0019522   5.287 1.36e-07 ***
## TEAM_FIELDING_E  -0.0166263  0.0022577  -7.364 2.48e-13 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 13.48 on 2268 degrees of freedom
## Multiple R-squared:   0.27,  Adjusted R-squared:  0.2678
## F-statistic: 119.9 on 7 and 2268 DF,  p-value: < 2.2e-16
```

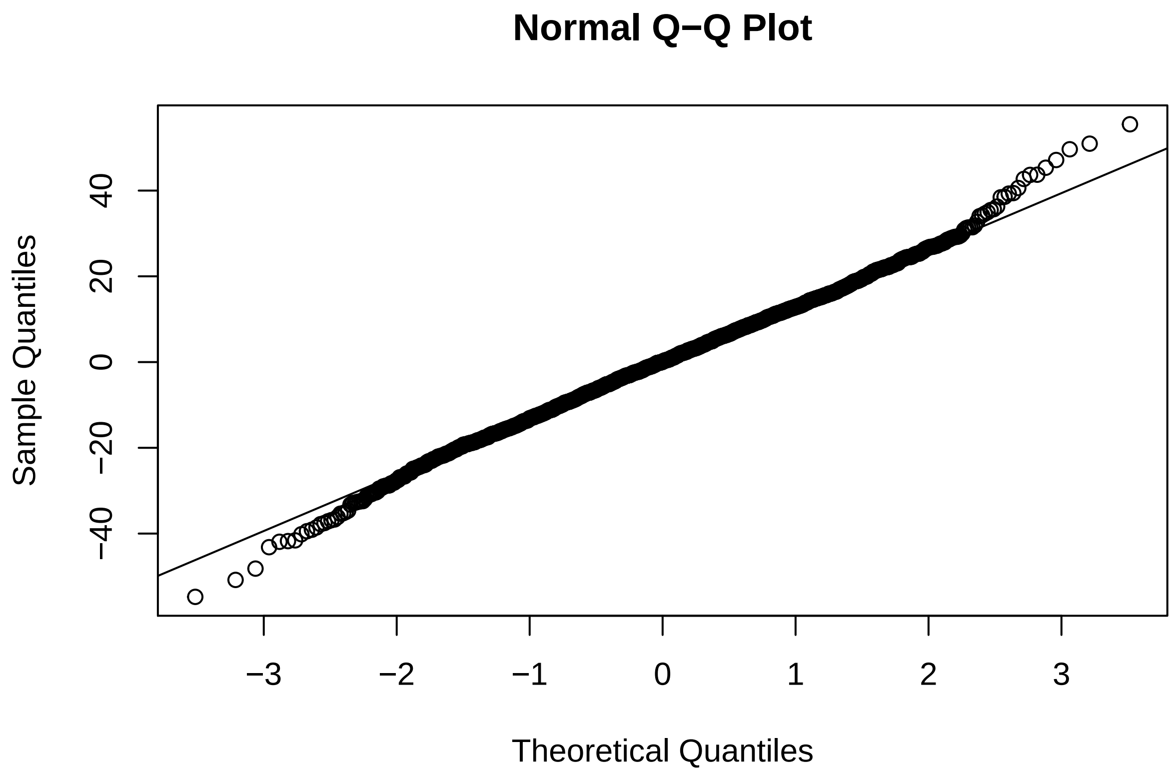
```r
plot(fitted(model), resid(model))
```




```
hist(model$residuals)
```




```r
qqnorm(resid(model))
qqline(resid(model))
```




```
#predict the model on the eval
colnames(test_df)
```

```
##  [1] "INDEX"              "TEAM_BATTING_H"     "TEAM_BATTING_2B"
##  [4] "TEAM_BATTING_3B"    "TEAM_BATTING_HR"    "TEAM_BATTING_BB"
##  [7] "TEAM_BATTING_SO"    "TEAM_BASERUN_SB"    "TEAM_BASERUN_CS"
## [10] "TEAM_BATTING_HBP"   "TEAM_PITCHING_H"    "TEAM_PITCHING_HR"
## [13] "TEAM_PITCHING_BB"   "TEAM_PITCHING_SO"   "TEAM_FIELDING_E"
## [16] "TEAM_FIELDING_DP"   "TEAM_BATTING_SO_NA" "TEAM_BASERUN_SB_NA"
## [19] "TEAM_BASERUN_CS_NA" "TEAM_BATTING_HBP_NA" "TEAM_PITCHING_SO_NA"
## [22] "TEAM_FIELDING_DP_NA"
```

```
#remove the predictors that have negative effect to the target wins

new_eval_model = subset(test_df, select=c(TEAM_BATTING_H, TEAM_BATTING_2B, TEAM_BATTING_3B, TEAM_BATTIN
# Turn the NA values in 0
new_eval_model[is.na(new_eval_model)] = 0

# prediction model
prediction_model <- predict(model, newdata=new_eval_model)
prediction_model
```

```
##        1        2        3        4        5        6        7        8
## 68.57679 70.20767 77.35107 83.60728 66.44188 67.44392 74.01699 72.52290
##        9       10       11       12       13       14       15       16
```

```
##  72.07908  75.86204  76.14127  85.66302  84.25863  82.11244  79.28366  80.65313
##        17        18        19        20        21        22        23        24
##  72.72498  80.73209  68.24429  93.15727  84.03790  86.72537  83.94422  76.45507
##        25        26        27        28        29        30        31        32
##  82.33443  84.46690  53.99437  77.34772  83.55037  76.54752  89.64897  87.49762
##        33        34        35        36        37        38        39        40
##  86.39979  88.63464  83.07959  82.97654  76.59917  90.98962  88.25264  89.93392
##        41        42        43        44        45        46        47        48
##  81.06430  86.65244  32.00565  93.94542  84.49850  91.12091  95.25990  72.55215
##        49        50        51        52        53        54        55        56
##  70.71842  77.42567  80.56279  86.18097  79.54452  75.66770  76.77920  78.91475
##        57        58        59        60        61        62        63        64
##  87.00232  70.24445  62.43238  76.94456  85.57690  82.32992  84.10415  84.08464
##        65        66        67        68        69        70        71        72
##  81.72510  88.61128  77.01994  84.45808  75.03575  84.58887  93.11545  78.11656
##        73        74        75        76        77        78        79        80
##  83.60987  87.48446  83.25982  87.59647  81.10361  79.45530  69.17038  75.34361
##        81        82        83        84        85        86        87        88
##  86.58620  91.02278  98.65784  83.24041  86.29588  81.38914  77.81345  83.29427
##        89        90        91        92        93        94        95        96
##  82.14307  85.78844  77.31626  90.17090  74.92238  80.27929  76.63840  76.41073
##        97        98        99       100       101       102       103       104
##  83.76351 101.49146  90.66066  91.80633  85.67709  75.74458  85.85636  82.51112
##       105       106       107       108       109       110       111       112
##  80.28514  75.74648  59.21657  80.05705  83.36447  63.89810  81.69559  80.89442
##       113       114       115       116       117       118       119       120
##  90.51339  88.42404  82.00004  79.88766  89.12636  79.28716  78.32773  70.56117
##       121       122       123       124       125       126       127       128
##  88.18073  64.83877  68.79647  62.89740  70.53486  89.14903  93.52098  77.13546
##       129       130       131       132       133       134       135       136
##  89.76420  96.00349  87.87496  79.55286  74.18762  83.65916  84.63120  67.92567
##       137       138       139       140       141       142       143       144
##  76.76088  79.31622  80.25903  79.00221  65.97271  70.88566  93.96534  80.09868
##       145       146       147       148       149       150       151       152
##  75.63502  76.66057  79.09194  81.58381  85.45157  81.03183  83.18578  79.69117
##       153       154       155       156       157       158       159       160
##  32.00533  74.74922  76.72696  73.53798  83.62346  70.38656  90.86799  71.82949
##       161       162       163       164       165       166       167       168
## 103.86302 102.94796  91.40787 103.43996  96.25437  92.15061  87.44536  83.28689
##       169       170       171       172       173       174       175       176
##  73.88550  80.44850  87.53529  83.90489  81.81791  91.73197  83.62750  78.62979
##       177       178       179       180       181       182       183       184
##  78.72177  78.62720  77.61974  80.23747  75.75891  82.42463  82.50687  83.40560
##       185       186       187       188       189       190       191       192
##  93.86719  84.08224  84.88270  59.90440  62.71131 106.61875  70.30532  79.80179
##       193       194       195       196       197       198       199       200
##  77.50981  80.91032  82.33698  71.26555  77.85090  81.87750  80.77272  86.39044
##       201       202       203       204       205       206       207       208
##  80.67028  82.42010  76.24716  85.64095  77.63218  78.86158  80.18659  76.75479
##       209       210       211       212       213       214       215       216
##  78.45877  74.04968 102.73424  94.95937  83.50166  71.04174  76.47425  88.75922
##       217       218       219       220       221       222       223       224
##  87.14607  86.32952  77.07959  76.85061  79.78128  75.26852  82.97115  79.90219
##       225       226       227       228       229       230       231       232
```

```
##  88.18673  76.73958  79.38705  80.08819  80.21071  76.68290  71.78995  94.28243
##       233       234       235       236       237       238       239       240
##  83.96099  86.59268  79.03424  74.35427  81.44309  78.25418  92.42152  75.30730
##       241       242       243       244       245       246       247       248
##  90.78059  88.86296  85.17144  83.49939  63.68535  86.98493  79.74425  82.77956
##       249       250       251       252       253       254       255       256
##  76.14821  84.12894  82.43395  59.19506  90.43189  46.23627  70.80823  77.18288
##       257       258       259
##  75.82183  77.87520  77.54615
```

Model 2 - Stepwise Regression

```
# Try stepwise regression as mentioned in http://www.sthda.com/english/articles/37-model-selection-esse
full_model = lm(TARGET_WINS ~ ., data=train_df)
step.model <- stepAIC(full_model, direction = "both",
                      trace = FALSE)
summary(step.model)
```

```
##
## Call:
## lm(formula = TARGET_WINS ~ TEAM_BATTING_H + TEAM_BATTING_2B +
##     TEAM_BATTING_3B + TEAM_BATTING_HR + TEAM_BATTING_BB + TEAM_BATTING_SO +
##     TEAM_BASERUN_SB + TEAM_PITCHING_H + TEAM_PITCHING_SO + TEAM_FIELDING_E +
##     TEAM_FIELDING_DP + TEAM_BASERUN_SB_NA + TEAM_BATTING_HBP_NA +
##     TEAM_FIELDING_DP_NA, data = train_df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -63.693  -8.067   0.330   7.875  49.924
##
## Coefficients:
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)          1.820e+01  4.192e+00   4.340 1.49e-05 ***
## TEAM_BATTING_H       4.682e-02  3.212e-03  14.578  < 2e-16 ***
## TEAM_BATTING_2B     -2.757e-02  8.973e-03  -3.073 0.002147 **
## TEAM_BATTING_3B      5.424e-02  1.547e-02   3.507 0.000461 ***
## TEAM_BATTING_HR      7.549e-02  8.642e-03   8.736  < 2e-16 ***
## TEAM_BATTING_BB      2.398e-02  3.239e-03   7.404 1.86e-13 ***
## TEAM_BATTING_SO     -1.025e-02  1.776e-03  -5.771 8.97e-09 ***
## TEAM_BASERUN_SB      5.014e-02  4.457e-03  11.249  < 2e-16 ***
## TEAM_PITCHING_H      1.980e-03  3.339e-04   5.930 3.49e-09 ***
## TEAM_PITCHING_SO    -1.096e-03  6.613e-04  -1.657 0.097666 .
## TEAM_FIELDING_E     -5.685e-02  3.370e-03 -16.873  < 2e-16 ***
## TEAM_FIELDING_DP    -1.045e-01  1.309e-02  -7.985 2.21e-15 ***
## TEAM_BASERUN_SB_NA   3.969e+01  2.048e+00  19.385  < 2e-16 ***
## TEAM_BATTING_HBP_NA  3.277e+00  1.071e+00   3.059 0.002244 **
## TEAM_FIELDING_DP_NA -1.073e+01  1.948e+00  -5.507 4.07e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 12.1 on 2261 degrees of freedom
```

```
## Multiple R-squared:  0.4135, Adjusted R-squared:  0.4098
## F-statistic: 113.9 on 14 and 2261 DF,  p-value: < 2.2e-16
```

```r
# Train model
train_control = trainControl(method = "cv", number = 10)
step_model = train(TARGET_WINS ~ ., data=train_df,
                   method = "lmStepAIC",
                   trControl = train_control,
                   trace=FALSE)
# Model accuracy
step_model$results
```

```
##   parameter    RMSE  Rsquared     MAE    RMSESD RsquaredSD     MAESD
## 1      none 12.2621 0.3901083 9.64169 0.5608981 0.06517879 0.3265393
```

```r
# Final model coefficients
step_model$finalModel
```

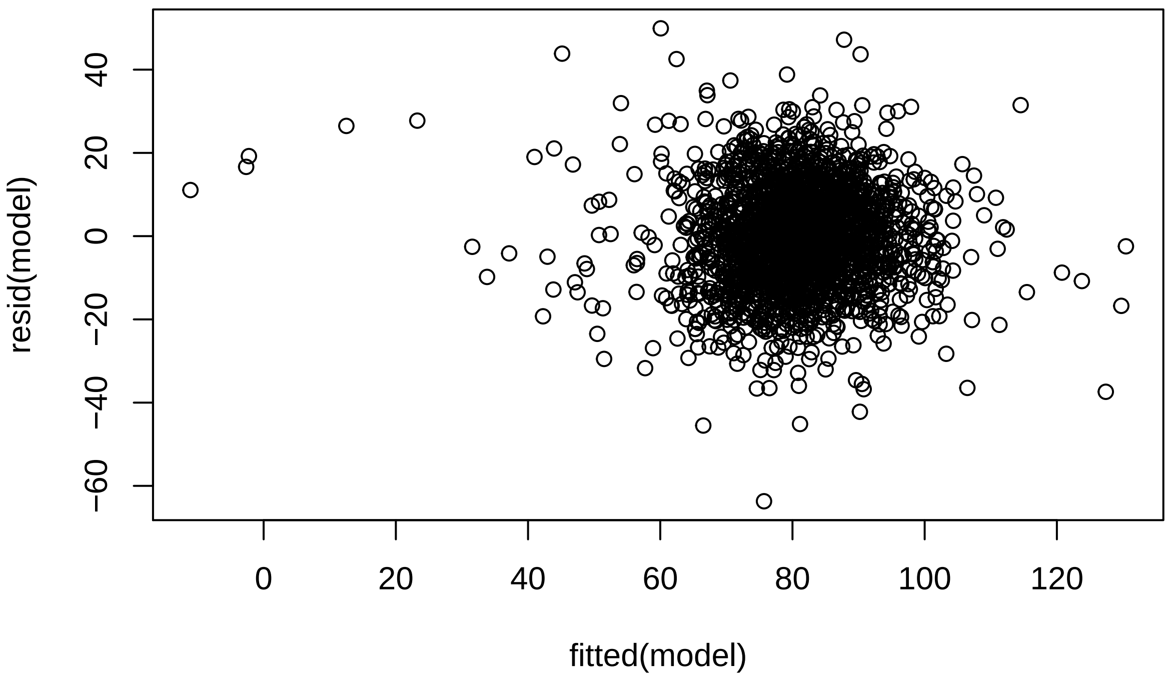
```
##
## Call:
## lm(formula = .outcome ~ TEAM_BATTING_H + TEAM_BATTING_2B + TEAM_BATTING_3B +
##     TEAM_BATTING_HR + TEAM_BATTING_BB + TEAM_BATTING_SO + TEAM_BASERUN_SB +
##     TEAM_PITCHING_H + TEAM_PITCHING_SO + TEAM_FIELDING_E + TEAM_FIELDING_DP +
##     TEAM_BASERUN_SB_NA + TEAM_BATTING_HBP_NA + TEAM_FIELDING_DP_NA,
##     data = dat)
##
## Coefficients:
##         (Intercept)       TEAM_BATTING_H      TEAM_BATTING_2B
##           18.196340             0.046820            -0.027572
##     TEAM_BATTING_3B      TEAM_BATTING_HR      TEAM_BATTING_BB
##            0.054244             0.075494             0.023983
##     TEAM_BATTING_SO      TEAM_BASERUN_SB      TEAM_PITCHING_H
##           -0.010247             0.050139             0.001980
##    TEAM_PITCHING_SO      TEAM_FIELDING_E     TEAM_FIELDING_DP
##           -0.001096            -0.056855            -0.104532
##  TEAM_BASERUN_SB_NA  TEAM_BATTING_HBP_NA  TEAM_FIELDING_DP_NA
##           39.693780             3.277467           -10.727882
```

```r
# Summary of model
summary(step_model$finalModel)
```

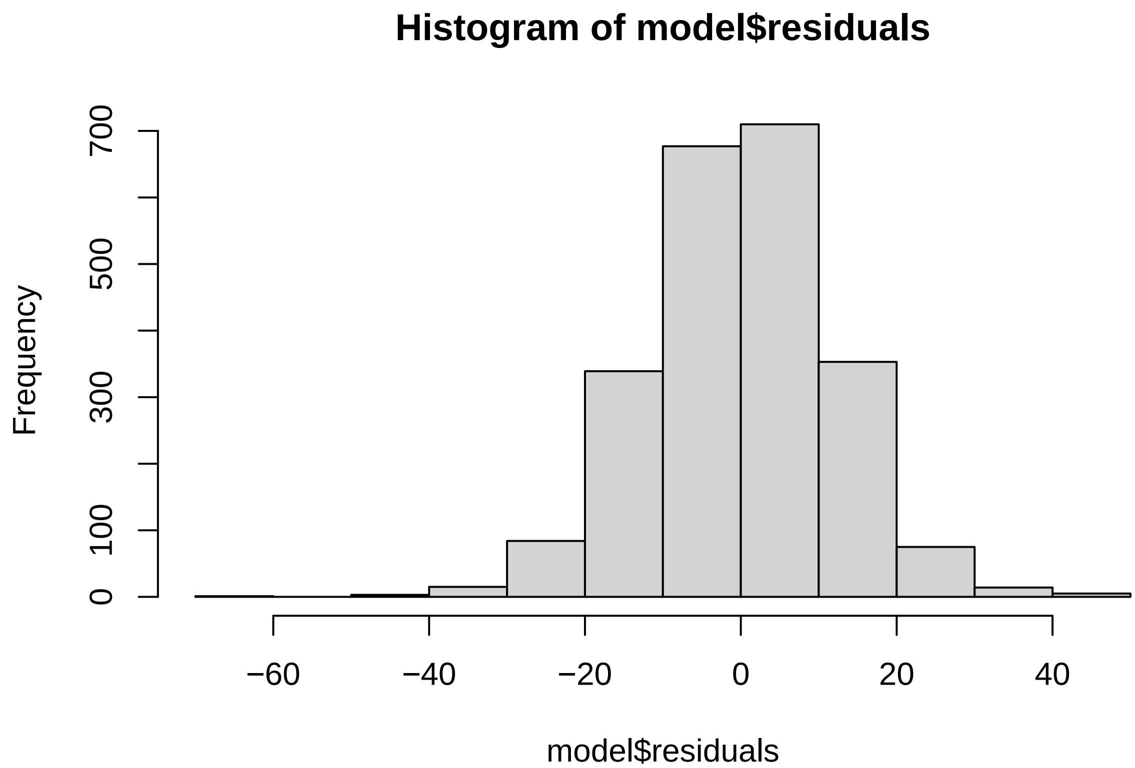
```
##
## Call:
## lm(formula = .outcome ~ TEAM_BATTING_H + TEAM_BATTING_2B + TEAM_BATTING_3B +
##     TEAM_BATTING_HR + TEAM_BATTING_BB + TEAM_BATTING_SO + TEAM_BASERUN_SB +
##     TEAM_PITCHING_H + TEAM_PITCHING_SO + TEAM_FIELDING_E + TEAM_FIELDING_DP +
##     TEAM_BASERUN_SB_NA + TEAM_BATTING_HBP_NA + TEAM_FIELDING_DP_NA,
##     data = dat)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -63.693  -8.067   0.330   7.875  49.924
```

```
##
## Coefficients:
##                      Estimate Std. Error t value Pr(>|t|)
## (Intercept)         1.820e+01  4.192e+00   4.340 1.49e-05 ***
## TEAM_BATTING_H      4.682e-02  3.212e-03  14.578  < 2e-16 ***
## TEAM_BATTING_2B    -2.757e-02  8.973e-03  -3.073 0.002147 **
## TEAM_BATTING_3B     5.424e-02  1.547e-02   3.507 0.000461 ***
## TEAM_BATTING_HR     7.549e-02  8.642e-03   8.736  < 2e-16 ***
## TEAM_BATTING_BB     2.398e-02  3.239e-03   7.404 1.86e-13 ***
## TEAM_BATTING_SO    -1.025e-02  1.776e-03  -5.771 8.97e-09 ***
## TEAM_BASERUN_SB     5.014e-02  4.457e-03  11.249  < 2e-16 ***
## TEAM_PITCHING_H     1.980e-03  3.339e-04   5.930 3.49e-09 ***
## TEAM_PITCHING_SO   -1.096e-03  6.613e-04  -1.657 0.097666 .
## TEAM_FIELDING_E    -5.685e-02  3.370e-03 -16.873  < 2e-16 ***
## TEAM_FIELDING_DP   -1.045e-01  1.309e-02  -7.985 2.21e-15 ***
## TEAM_BASERUN_SB_NA  3.969e+01  2.048e+00  19.385  < 2e-16 ***
## TEAM_BATTING_HBP_NA  3.277e+00  1.071e+00   3.059 0.002244 **
## TEAM_FIELDING_DP_NA -1.073e+01  1.948e+00  -5.507 4.07e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 12.1 on 2261 degrees of freedom
## Multiple R-squared:  0.4135, Adjusted R-squared:  0.4098
## F-statistic: 113.9 on 14 and 2261 DF,  p-value: < 2.2e-16
```

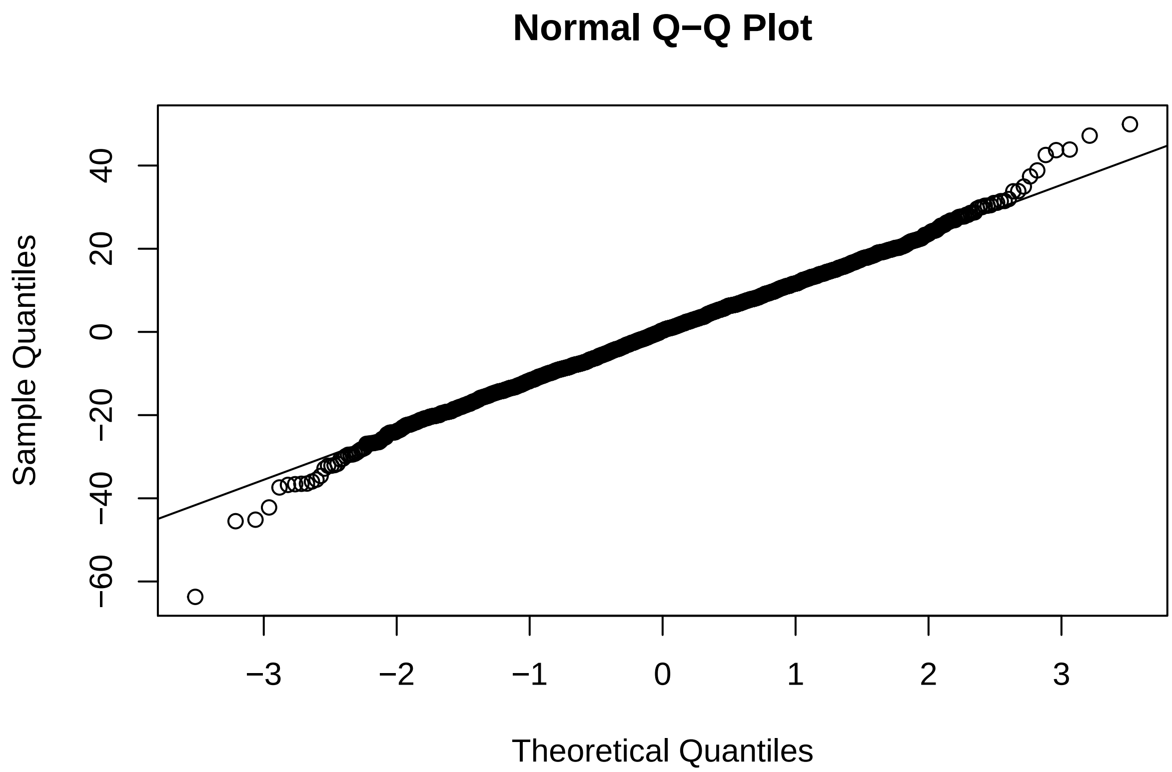
```r
model = step_model$finalModel
plot(fitted(model), resid(model))
```




```r
hist(model$residuals)
```




```r
qqnorm(resid(model))
qqline(resid(model))
```




```r
# Check MSE
mean(summary(model$residuals^2))
```

```
## [1] 743.6606
```

```r
# 743.6606
```

Model 3 - Try removing TEAM_PITCHING_SO

```r
# Train model without TEAM_PITCHING_SO since it has a relatively high p-value
train_control = trainControl(method = "cv", number = 10)
no_TPS = subset(train_df, select=-c(TEAM_PITCHING_SO))
step_model_noTPS = train(TARGET_WINS ~ ., data=no_TPS,
                   method = "lmStepAIC",
                   trControl = train_control,
                   trace=FALSE)
# Model accuracy
step_model_noTPS$results
```

```
##   parameter    RMSE  Rsquared      MAE   RMSESD RsquaredSD     MAESD
## 1      none 12.2715 0.3955766 9.654802 0.559686 0.04342146 0.3603521
```

```r
# Final model coefficients
step_model_noTPS$finalModel
```

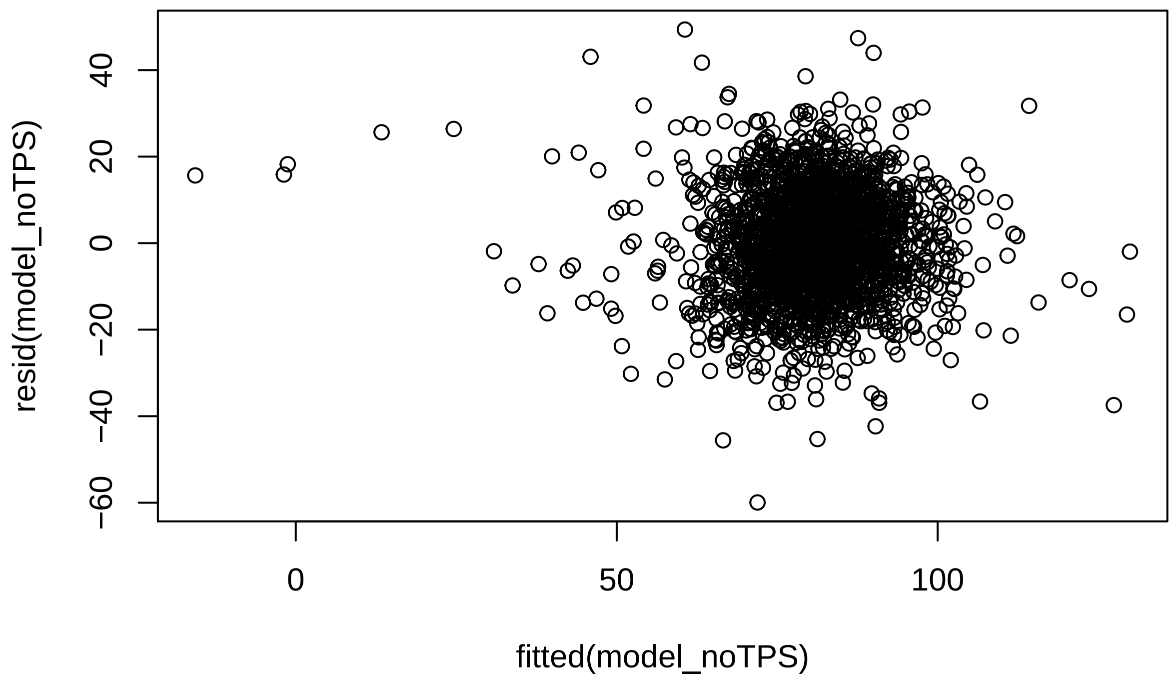
```
##
## Call:
## lm(formula = .outcome ~ TEAM_BATTING_H + TEAM_BATTING_2B + TEAM_BATTING_3B +
##     TEAM_BATTING_HR + TEAM_BATTING_BB + TEAM_BATTING_SO + TEAM_BASERUN_SB +
##     TEAM_PITCHING_H + TEAM_FIELDING_E + TEAM_FIELDING_DP + TEAM_BASERUN_SB_NA +
##     TEAM_BATTING_HBP_NA + TEAM_FIELDING_DP_NA, data = dat)
##
## Coefficients:
##         (Intercept)       TEAM_BATTING_H      TEAM_BATTING_2B
##           17.475160             0.047857            -0.029176
##     TEAM_BATTING_3B      TEAM_BATTING_HR      TEAM_BATTING_BB
##            0.053032             0.076969             0.023764
##     TEAM_BATTING_SO      TEAM_BASERUN_SB      TEAM_PITCHING_H
##           -0.011516             0.049503             0.001683
##     TEAM_FIELDING_E     TEAM_FIELDING_DP   TEAM_BASERUN_SB_NA
##           -0.055278            -0.104090            38.683727
## TEAM_BATTING_HBP_NA  TEAM_FIELDING_DP_NA
##            3.170394           -11.012818
```

```r
# Summary of model
summary(step_model_noTPS$finalModel)
```

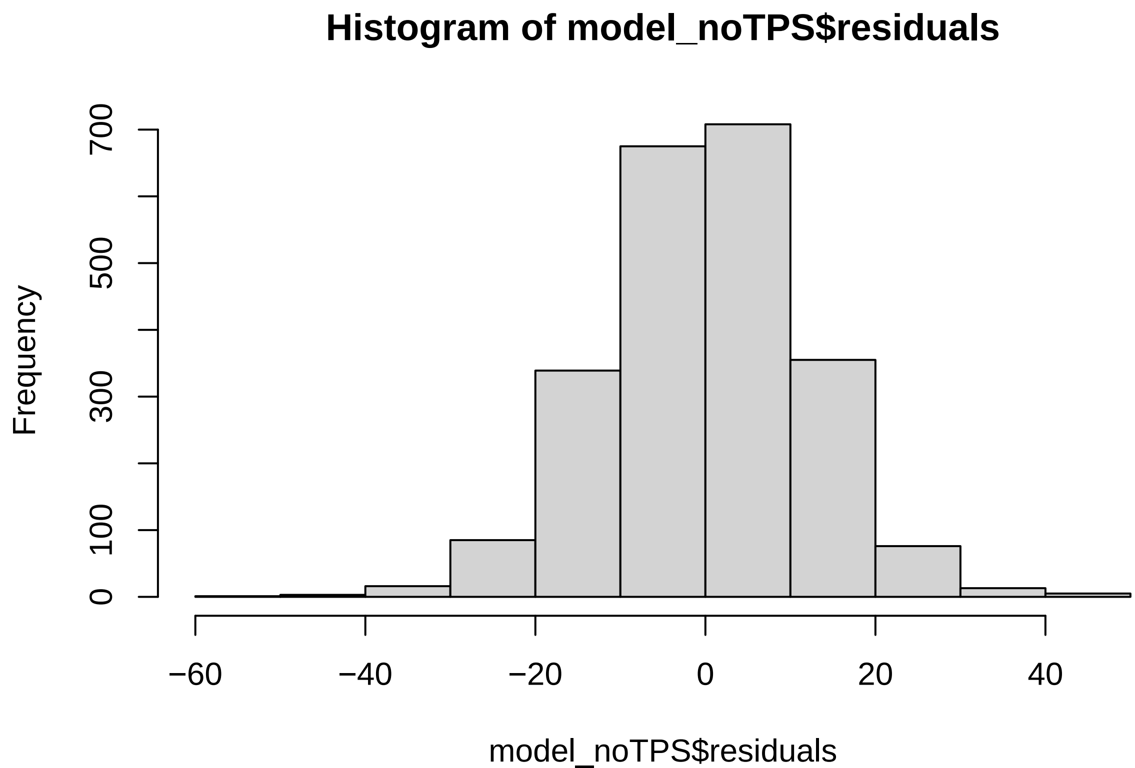
```
##
## Call:
## lm(formula = .outcome ~ TEAM_BATTING_H + TEAM_BATTING_2B + TEAM_BATTING_3B +
##     TEAM_BATTING_HR + TEAM_BATTING_BB + TEAM_BATTING_SO + TEAM_BASERUN_SB +
##     TEAM_PITCHING_H + TEAM_FIELDING_E + TEAM_FIELDING_DP + TEAM_BASERUN_SB_NA +
##     TEAM_BATTING_HBP_NA + TEAM_FIELDING_DP_NA, data = dat)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -59.938  -8.049   0.369   7.904  49.371
##
## Coefficients:
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)          1.748e+01  4.171e+00   4.189 2.91e-05 ***
## TEAM_BATTING_H       4.786e-02  3.151e-03  15.187  < 2e-16 ***
## TEAM_BATTING_2B     -2.918e-02  8.924e-03  -3.269 0.001094 **
## TEAM_BATTING_3B      5.303e-02  1.545e-02   3.432 0.000611 ***
## TEAM_BATTING_HR      7.697e-02  8.599e-03   8.951  < 2e-16 ***
## TEAM_BATTING_BB      2.376e-02  3.238e-03   7.339 2.98e-13 ***
## TEAM_BATTING_SO     -1.152e-02  1.603e-03  -7.185 9.12e-13 ***
## TEAM_BASERUN_SB      4.950e-02  4.442e-03  11.144  < 2e-16 ***
## TEAM_PITCHING_H      1.683e-03  2.817e-04   5.974 2.68e-09 ***
## TEAM_FIELDING_E     -5.528e-02  3.234e-03 -17.095  < 2e-16 ***
## TEAM_FIELDING_DP    -1.041e-01  1.309e-02  -7.950 2.92e-15 ***
## TEAM_BASERUN_SB_NA   3.868e+01  1.956e+00  19.782  < 2e-16 ***
## TEAM_BATTING_HBP_NA  3.170e+00  1.070e+00   2.964 0.003072 **
## TEAM_FIELDING_DP_NA -1.101e+01  1.941e+00  -5.673 1.58e-08 ***
## ---
```

```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 12.11 on 2262 degrees of freedom
## Multiple R-squared:  0.4128, Adjusted R-squared:  0.4094
## F-statistic: 122.3 on 13 and 2262 DF,  p-value: < 2.2e-16
```

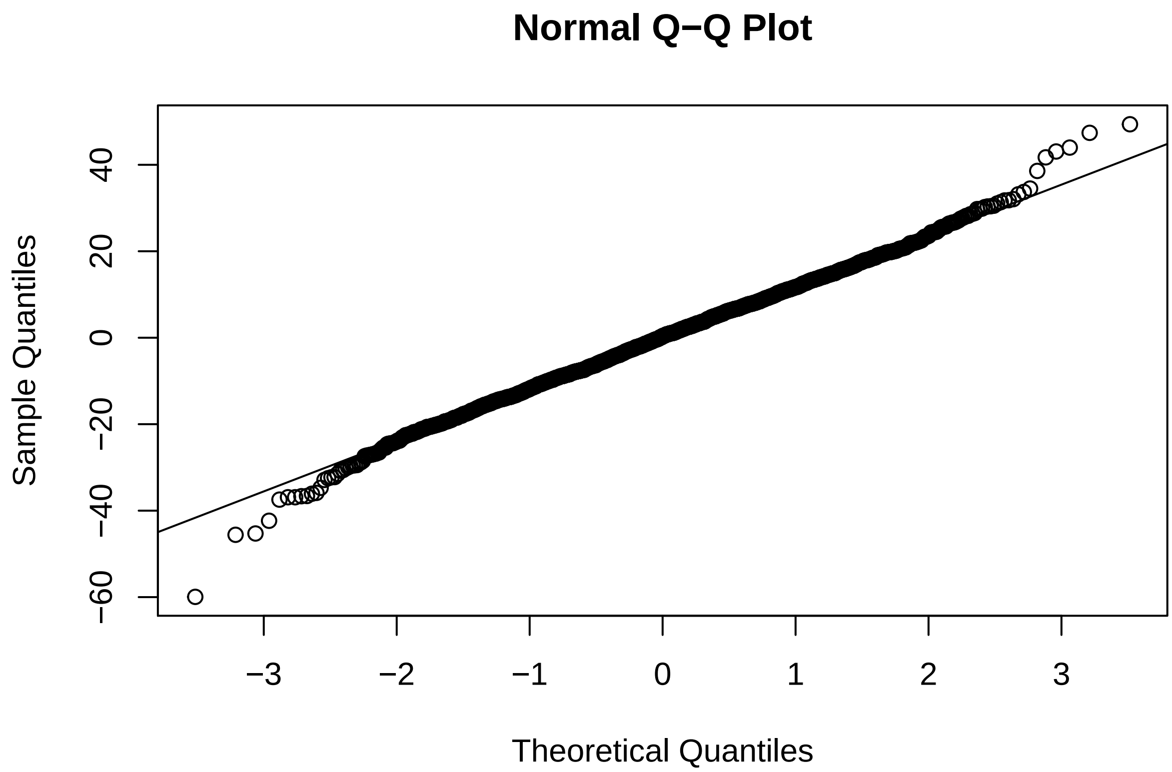
```r
model_noTPS = step_model_noTPS$finalModel
plot(fitted(model_noTPS), resid(model_noTPS))
```




```r
hist(model_noTPS$residuals)
```



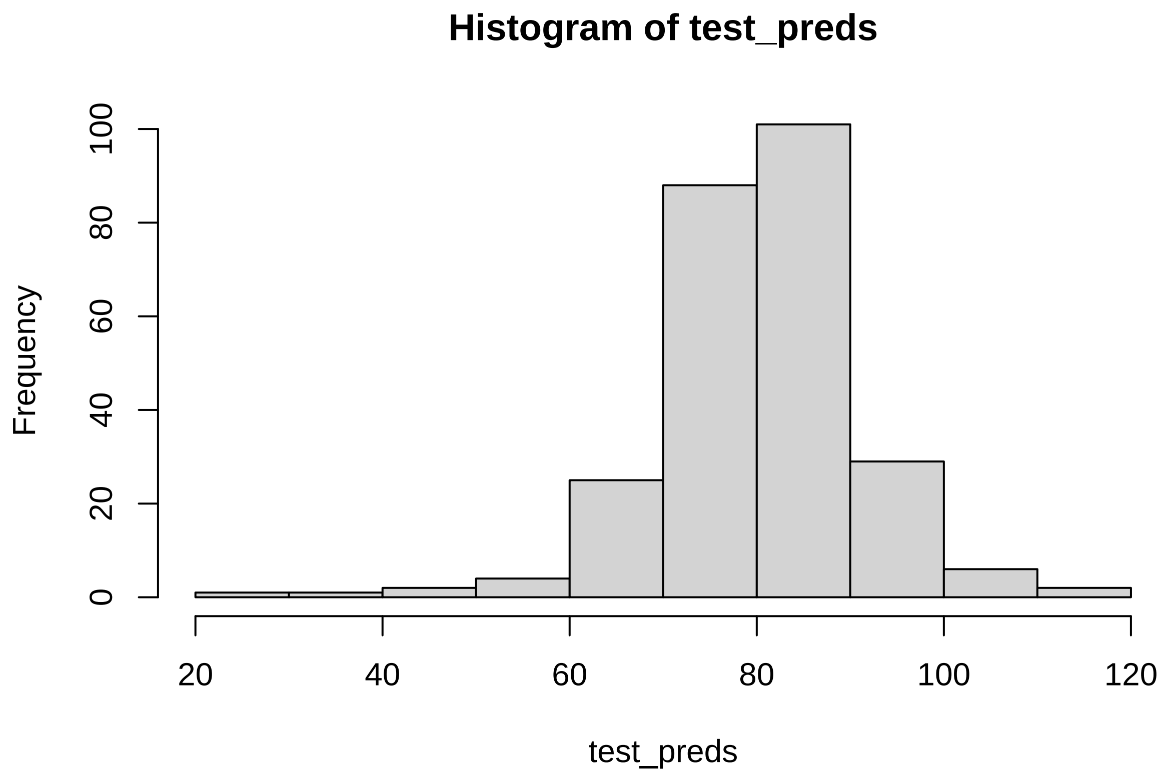

```
qqnorm(resid(model_noTPS))
qqline(resid(model_noTPS))
```



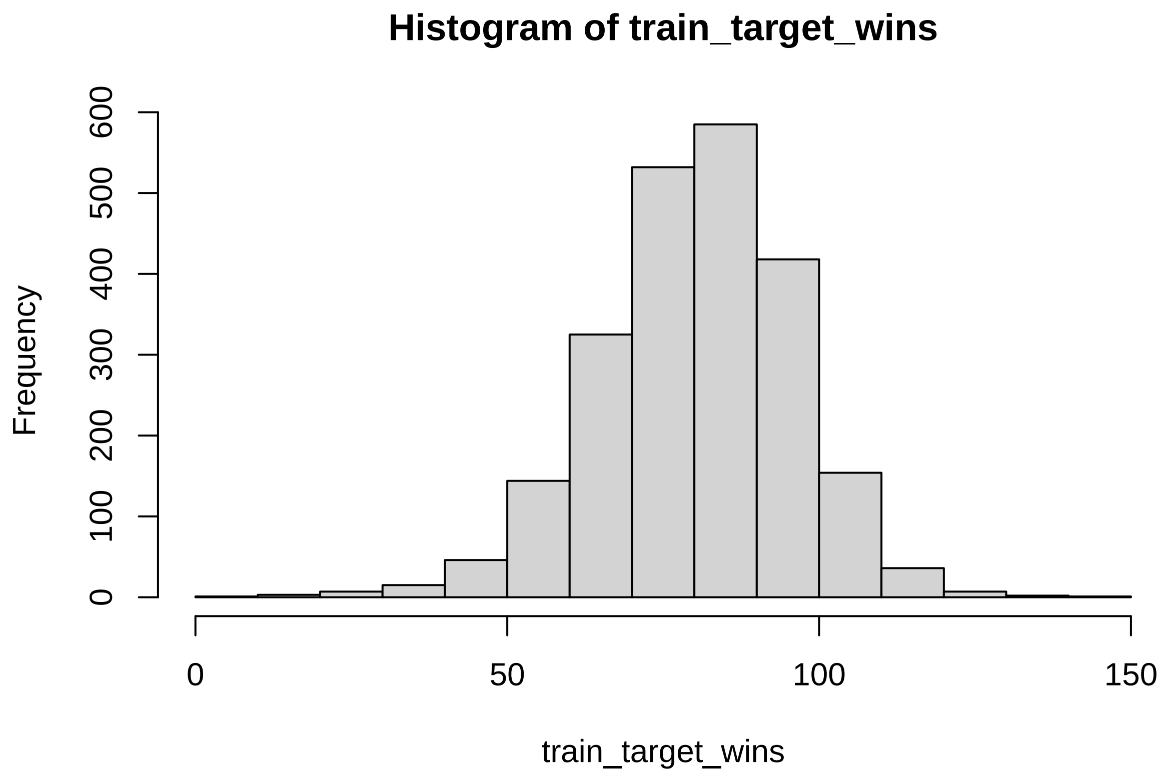
Predictions on Evaluation Set

```
# convert decimals of wins back to number of wins, rounded
test_preds = round(predict(model, newdata=test_df)) #*162
test_df$PRED_TARGET_WINS = test_preds
# write out evaluation data with predictions
write.csv(test_df, 'datasets/eval_with_preds.csv')

# visually inspect the distribution of predictions for test and wins from the training set
hist(test_preds)
```




```r
hist(train_target_wins)
```


Histogram of train_target_wins
Frequency
0
100
200
300
400
500
600
0
50
100
150
train_target_wins