HW1

Team 1

September 13, 2020

```
# 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
```

```
#library(tidyr)
library(MASS)
```

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

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

```
library(caret)
```

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

```
library(RCurl)
```

```
# Loading the data
git_dir <- 'https://raw.github.com/odonnell31/data621-HW1/master/data'
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

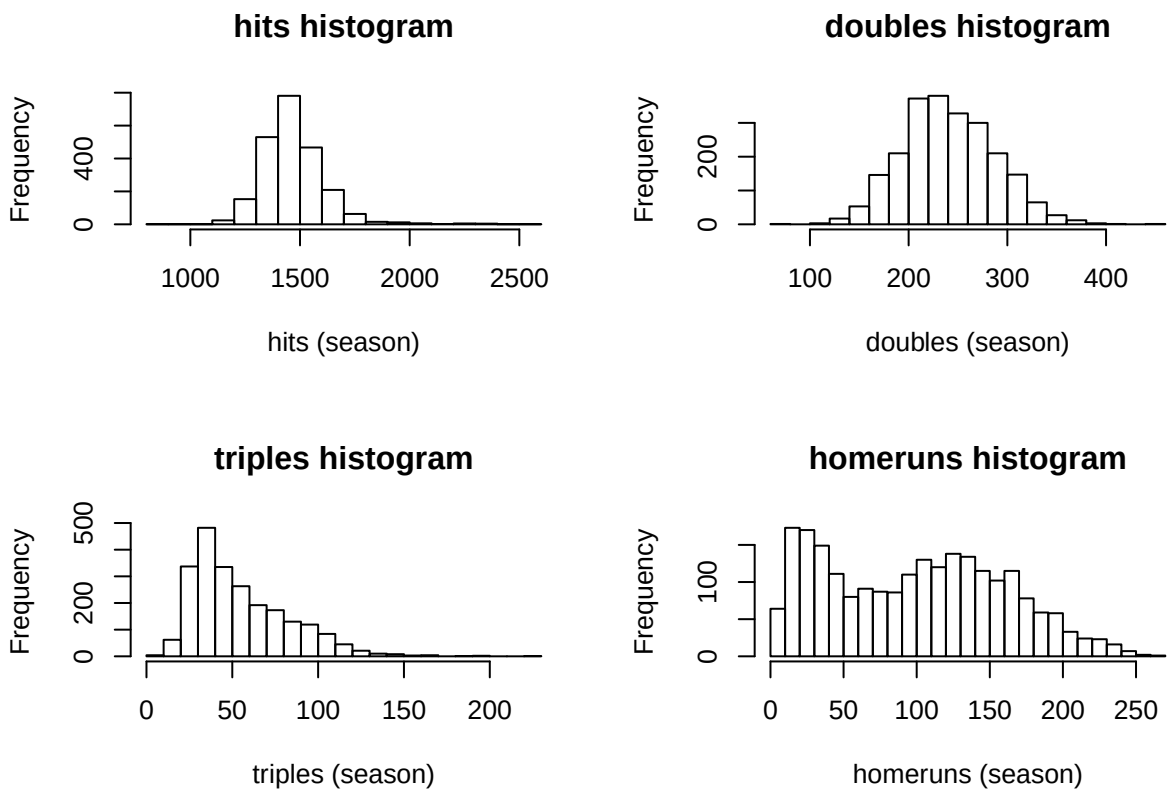
```
# view a summary of all columns
summary(train_df)
```

```
##      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
```

For types of hits, see a histogram of each

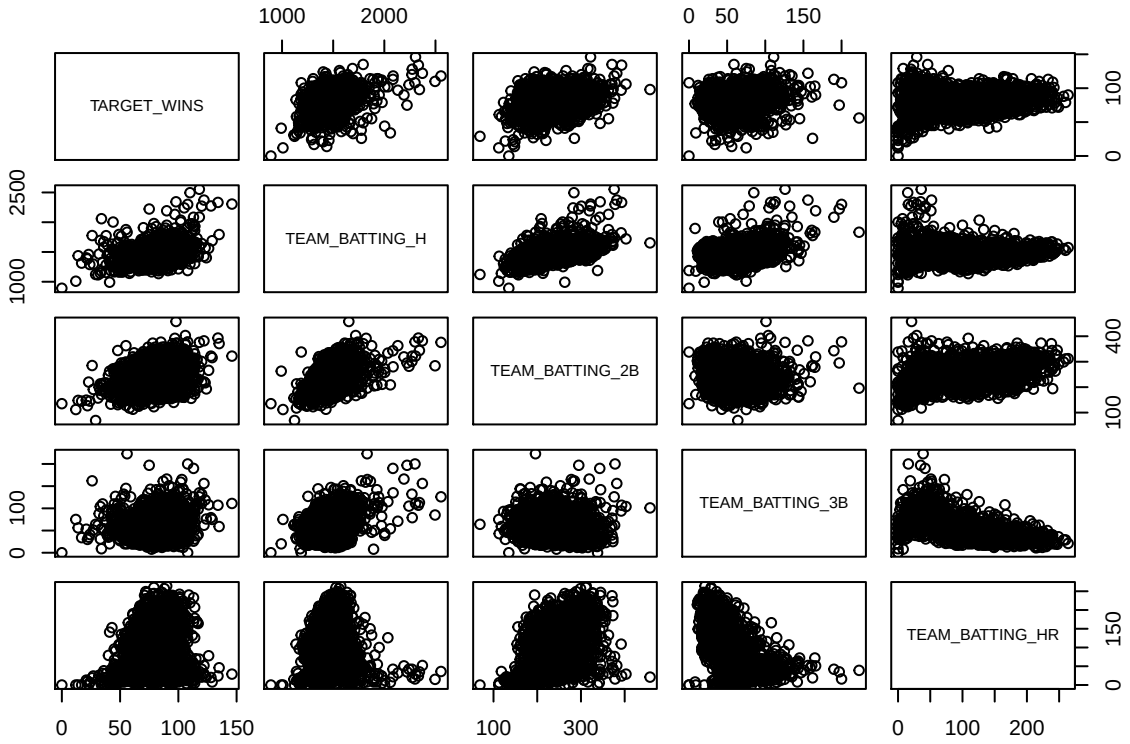
```
par(mfrow=c(2,2))
hist(train_df$TEAM_BATTING_H,
     main = "hits histogram", xlab = "hits (season)",
```

```
     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)
```



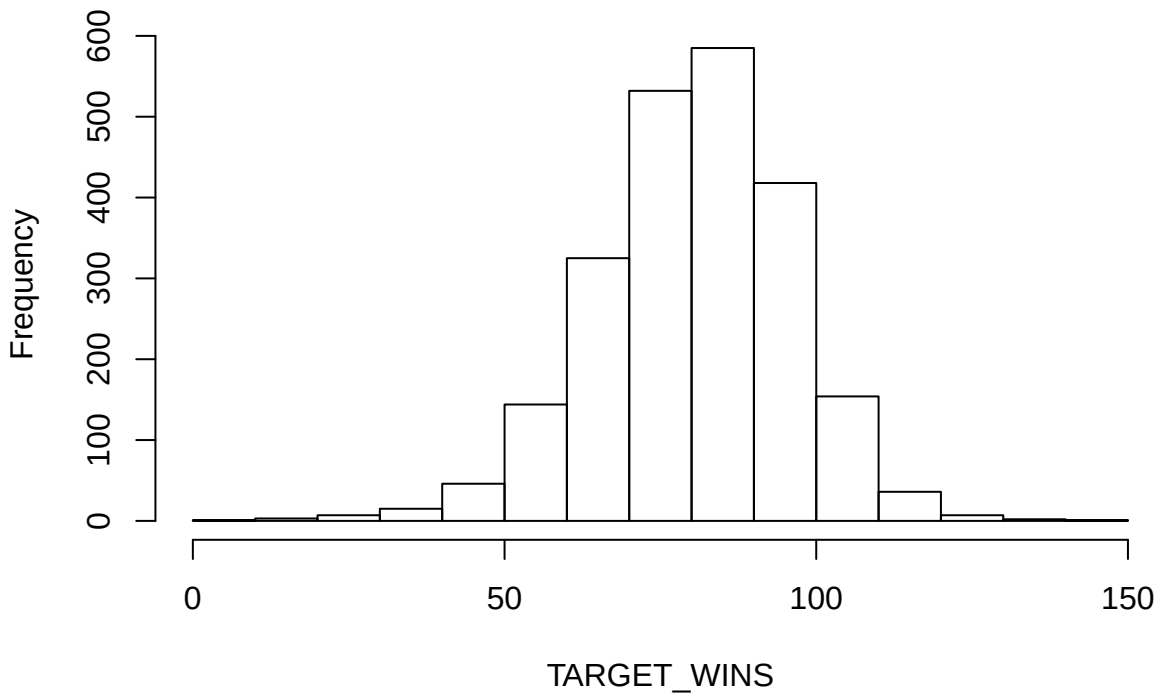

```
# look at the structure of the variables
str(train_df)
```

```
## '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 ...
```

```
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="")
```

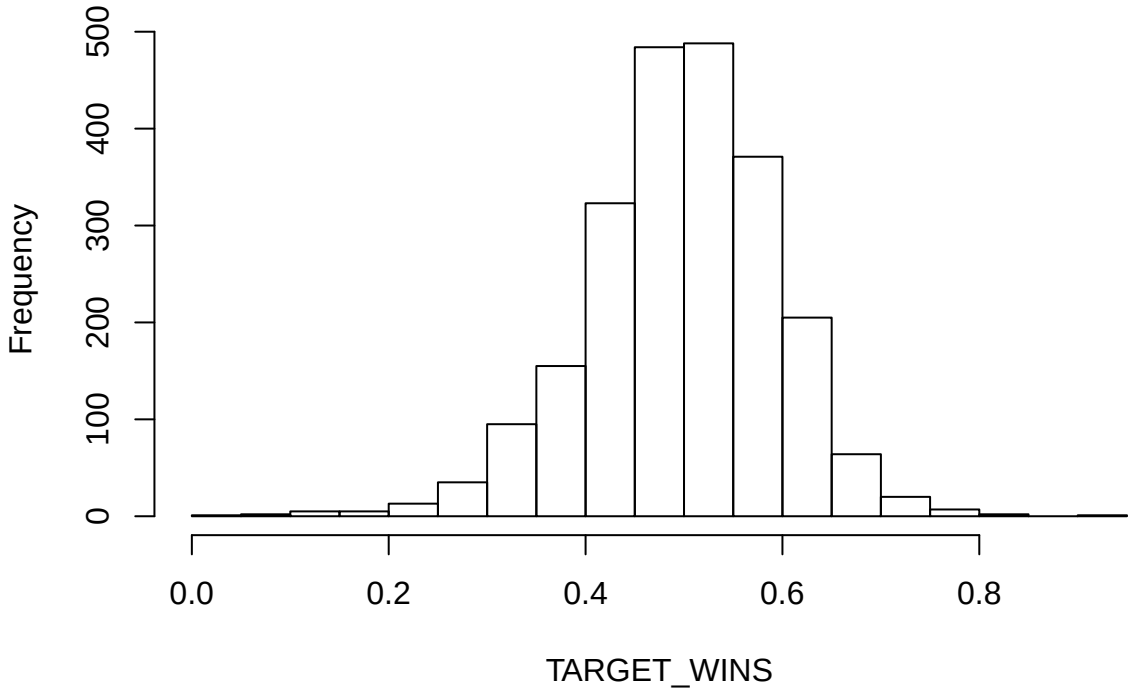



```r
# we have no TARGET_WINS from eval
# hist(eval$TARGET_WINS,xlab="TARGET_WINS",main="")
```

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="")
```




```
str(train_df)
```

```
## 'data.frame':    2276 obs. of  17 variables:
##  $ INDEX           : int  1 2 3 4 5 6 7 8 11 12 ...
##  $ TARGET_WINS     : num  0.241 0.432 0.531 0.432 0.506 ...
##  $ 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)
# Random sample indexes
# train_df_idx <- sample(1:nrow(train_df), 0.9 * nrow(train_df))
# val_idx <- setdiff(1:nrow(train_df), train_df_idx)
# val = train_df[val_idx,]
# train_df = train_df[train_df_idx,]
```

```
#model = lm(TARGET_WINS ~ ., data=train_df)
#summary(model)
# drop TEAM_PITCHING_SO_NA because model thinks its correlated with another var
#train_df = subset(train_df, select = -c(TEAM_PITCHING_SO_NA))
#model = lm(TARGET_WINS ~ ., data=train_df)
#summary(model)
```

```
#layout(matrix(c(1,2,3,4),2,2)) # optional 4 graphs/page
#plot(model)
```

following ideas for model selecting taken from http://www.sthda.com/english/articles/37-model-selection-essentials-in-r/154-stepwise-regression-essentials-in-r/

```
# Try different selection methods
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
## -0.39317 -0.04980  0.00204  0.04861  0.30817
##
## Coefficients:
##                        Estimate Std. Error t value Pr(>|t|)
## (Intercept)          1.123e-01  2.588e-02   4.340 1.49e-05 ***
## TEAM_BATTING_H       2.890e-04  1.983e-05  14.578  < 2e-16 ***
## TEAM_BATTING_2B     -1.702e-04  5.539e-05  -3.073 0.002147 **
## TEAM_BATTING_3B      3.348e-04  9.547e-05   3.507 0.000461 ***
## TEAM_BATTING_HR      4.660e-04  5.334e-05   8.736  < 2e-16 ***
## TEAM_BATTING_BB      1.480e-04  2.000e-05   7.404 1.86e-13 ***
## TEAM_BATTING_SO     -6.325e-05  1.096e-05  -5.771 8.97e-09 ***
## TEAM_BASERUN_SB      3.095e-04  2.751e-05  11.249  < 2e-16 ***
## TEAM_PITCHING_H      1.222e-05  2.061e-06   5.930 3.49e-09 ***
## TEAM_PITCHING_SO    -6.764e-06  4.082e-06  -1.657 0.097666 .
## TEAM_FIELDING_E     -3.510e-04  2.080e-05 -16.873  < 2e-16 ***
## TEAM_FIELDING_DP    -6.453e-04  8.081e-05  -7.985 2.21e-15 ***
## TEAM_BASERUN_SB_NA   2.450e-01  1.264e-02  19.385  < 2e-16 ***
## TEAM_BATTING_HBP_NA  2.023e-02  6.613e-03   3.059 0.002244 **
## TEAM_FIELDING_DP_NA -6.622e-02  1.203e-02  -5.507 4.07e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.0747 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
```

```
# 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 0.075692 0.3901083 0.05951661 0.003462334 0.06517879 0.002015675
```

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

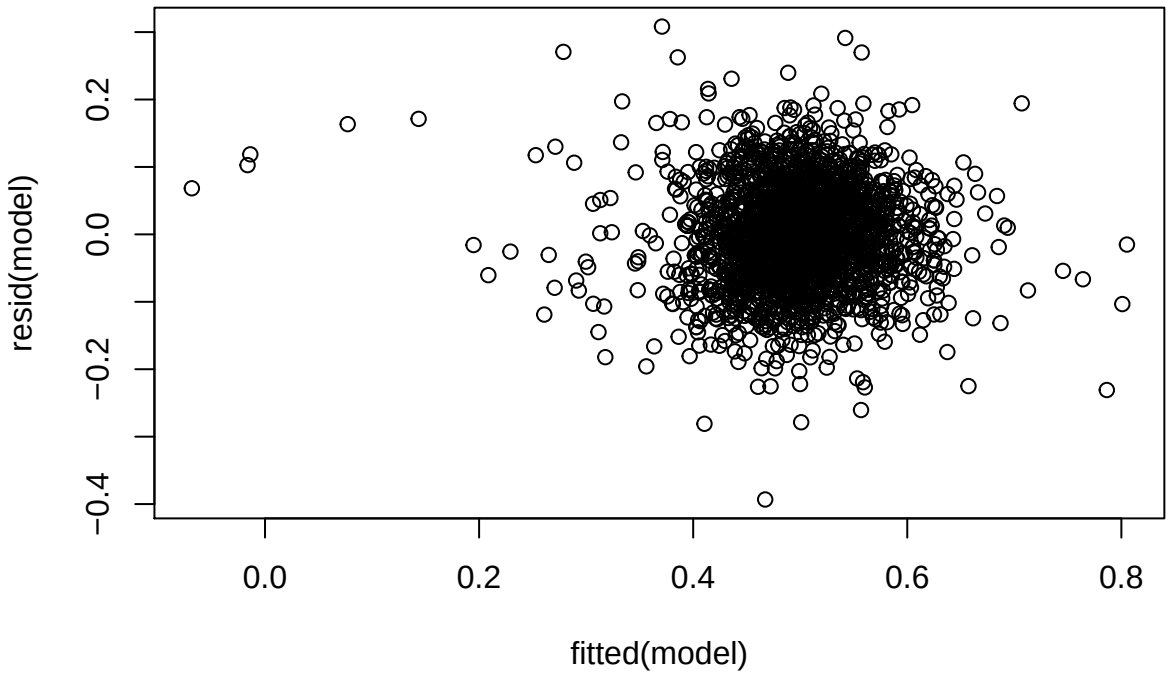
```
##
## Call:
## lm(formula = .outcome ~ TEAM_BATTING_H + TEAM_BATTING_2B + TEAM_BATTING_3B +
##     TEAM_BATTING_HR + TEAM_BATTING_BB + TEAM_BATTING_SO + TEAM_BASERUN_SB +
##     TEAM_PITCHING_H + TEAM_PITCHING_SO + TEAM_FIELDING_E + TEAM_FIELDING_DP +
##     TEAM_BASERUN_SB_NA + TEAM_BATTING_HBP_NA + TEAM_FIELDING_DP_NA,
```

```
##     data = dat)
##
## Coefficients:
##         (Intercept)       TEAM_BATTING_H      TEAM_BATTING_2B
##           1.123e-01            2.890e-04           -1.702e-04
##     TEAM_BATTING_3B      TEAM_BATTING_HR      TEAM_BATTING_BB
##           3.348e-04            4.660e-04            1.480e-04
##     TEAM_BATTING_SO      TEAM_BASERUN_SB      TEAM_PITCHING_H
##          -6.325e-05            3.095e-04            1.222e-05
##    TEAM_PITCHING_SO      TEAM_FIELDING_E     TEAM_FIELDING_DP
##          -6.764e-06           -3.510e-04           -6.453e-04
##  TEAM_BASERUN_SB_NA  TEAM_BATTING_HBP_NA  TEAM_FIELDING_DP_NA
##           2.450e-01            2.023e-02           -6.622e-02
```

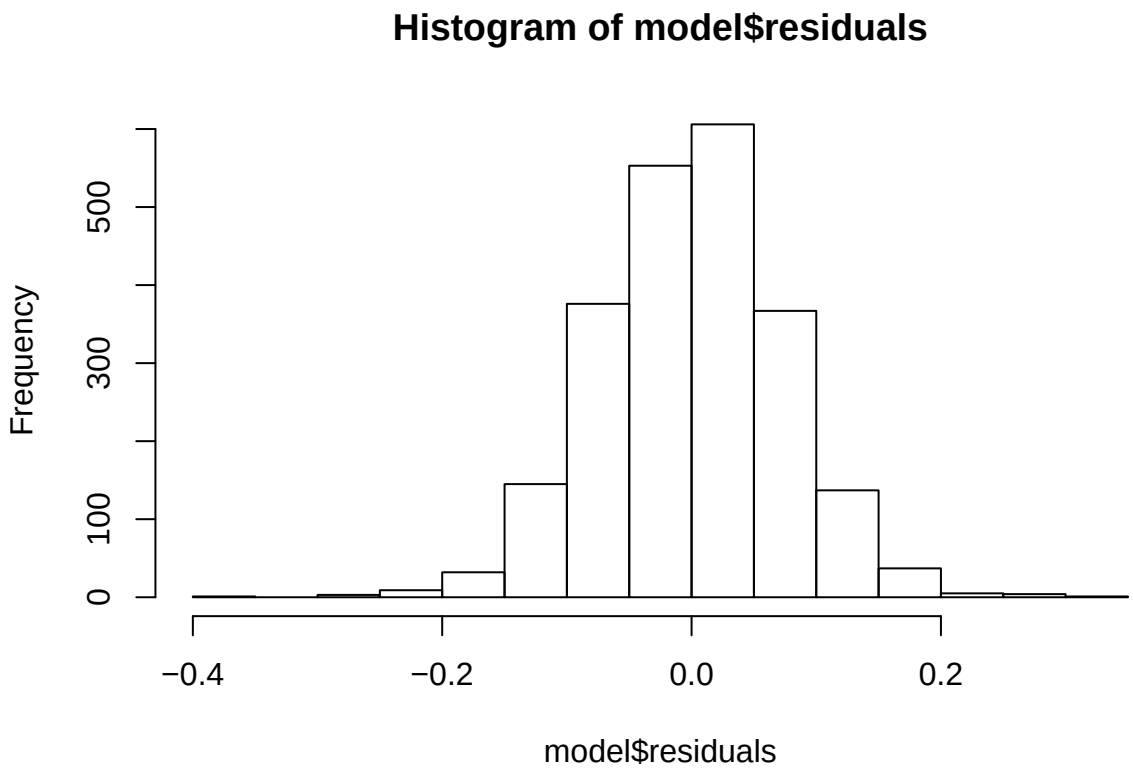
```
# Summary of model
summary(step_model$finalModel)
```

```
##
## 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
## -0.39317 -0.04980  0.00204  0.04861  0.30817
##
## Coefficients:
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)          1.123e-01  2.588e-02   4.340 1.49e-05 ***
## TEAM_BATTING_H       2.890e-04  1.983e-05  14.578  < 2e-16 ***
## TEAM_BATTING_2B     -1.702e-04  5.539e-05  -3.073 0.002147 **
## TEAM_BATTING_3B      3.348e-04  9.547e-05   3.507 0.000461 ***
## TEAM_BATTING_HR      4.660e-04  5.334e-05   8.736  < 2e-16 ***
## TEAM_BATTING_BB      1.480e-04  2.000e-05   7.404 1.86e-13 ***
## TEAM_BATTING_SO     -6.325e-05  1.096e-05  -5.771 8.97e-09 ***
## TEAM_BASERUN_SB      3.095e-04  2.751e-05  11.249  < 2e-16 ***
## TEAM_PITCHING_H      1.222e-05  2.061e-06   5.930 3.49e-09 ***
## TEAM_PITCHING_SO    -6.764e-06  4.082e-06  -1.657 0.097666 .
## TEAM_FIELDING_E     -3.510e-04  2.080e-05 -16.873  < 2e-16 ***
## TEAM_FIELDING_DP    -6.453e-04  8.081e-05  -7.985 2.21e-15 ***
## TEAM_BASERUN_SB_NA   2.450e-01  1.264e-02  19.385  < 2e-16 ***
## TEAM_BATTING_HBP_NA  2.023e-02  6.613e-03   3.059 0.002244 **
## TEAM_FIELDING_DP_NA -6.622e-02  1.203e-02  -5.507 4.07e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.0747 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
```

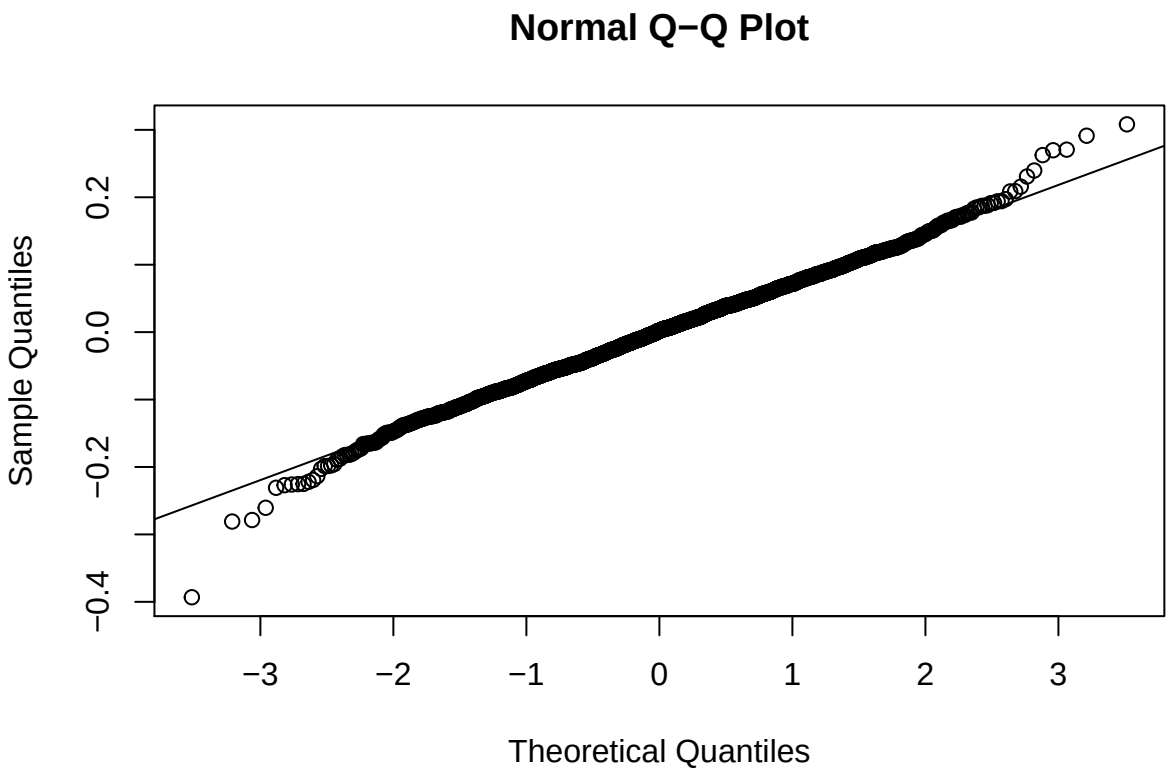
```r
model = step_model$finalModel
plot(fitted(model), resid(model))
```




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

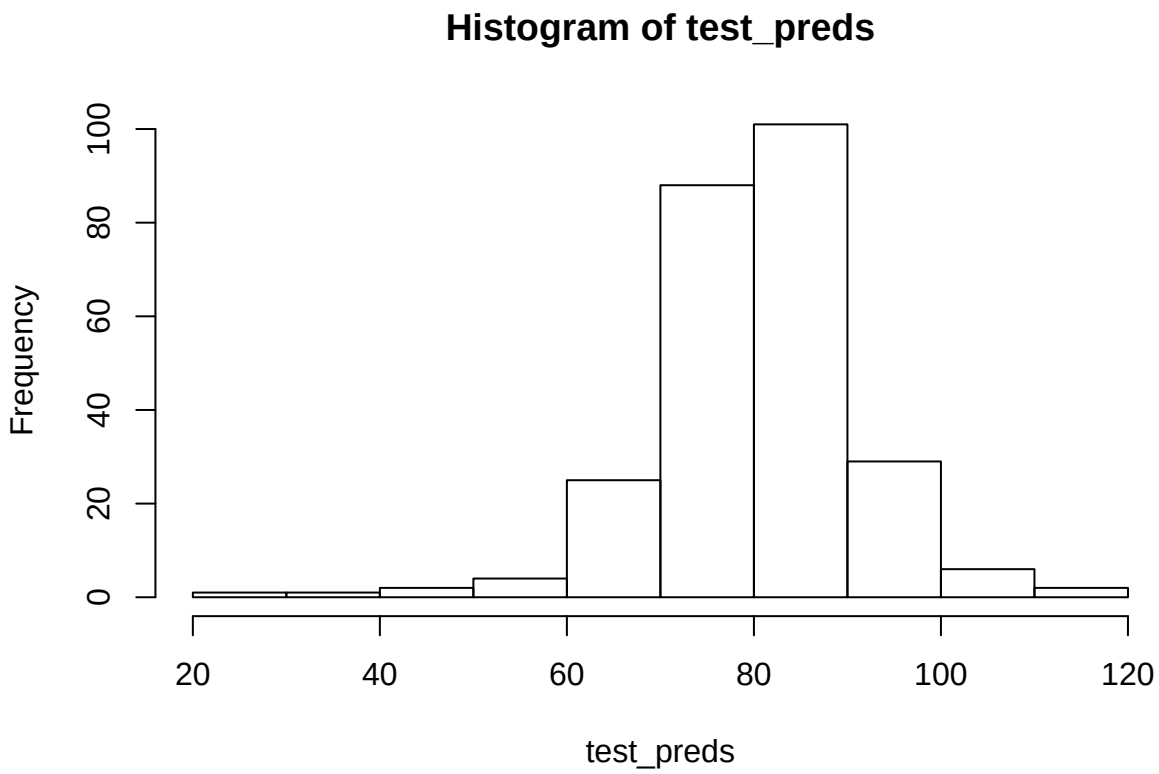



```
qqnorm(resid(model))
qqline(resid(model))
```

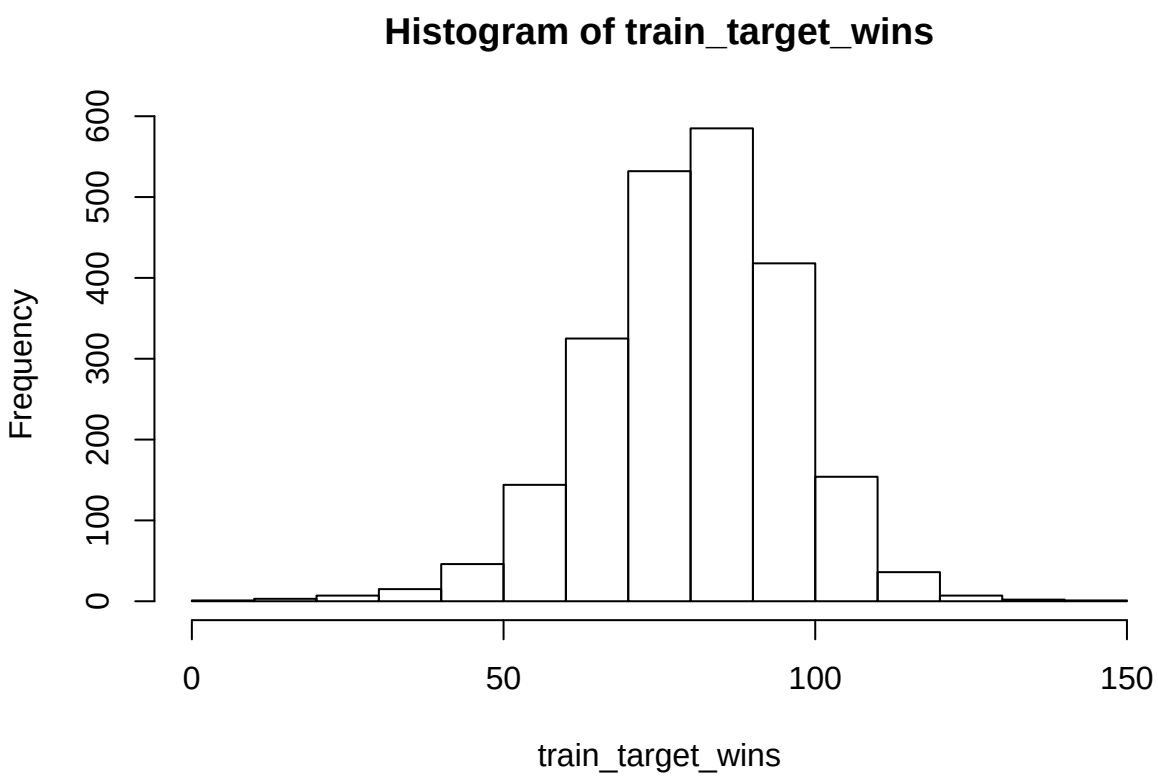


Predictions on Evaluation Set

```
# 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, 'data/eval_with_preds.csv')
hist(test_preds)
```




```
hist(train_target_wins)
```


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