Data_621_Homework1

Group 1

Objective

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

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

Load Packages

Run the below cell to laod the packges needed to use the notebook. Please note not all of the packages are included in base R so you may need to install some packages before running the code.

```
# Data Prep Libraries
library(knitr)
library(tidyverse)
library(reshape2)
library(naniar)
library(skimr)

# Modeling Libraries
library(fastDummies)
library(funModeling)
library(caret)
library(MASS)

# Data Visaulization Libraries
library(corrplot)
library(ggplot2)
library(VIM)
library(GGally)
library(gridExtra)
```

Exploratory Data Analysis

Code Author: Forhad Akbar

The below cells are for the intent of exploring the relationships that exist within the data. The training data contains data for 2,276 baseball teams with 16 features and one target variable. We explore the various relationships and statistics using a number of different plots. Some of the struggles we see with this data set is the amount of missing data and the effect outliers have on the models. Over the next couple cells we will examine that information and show you what we did to deal with that later on in the modeling development section.

Investigate Summary Statistics

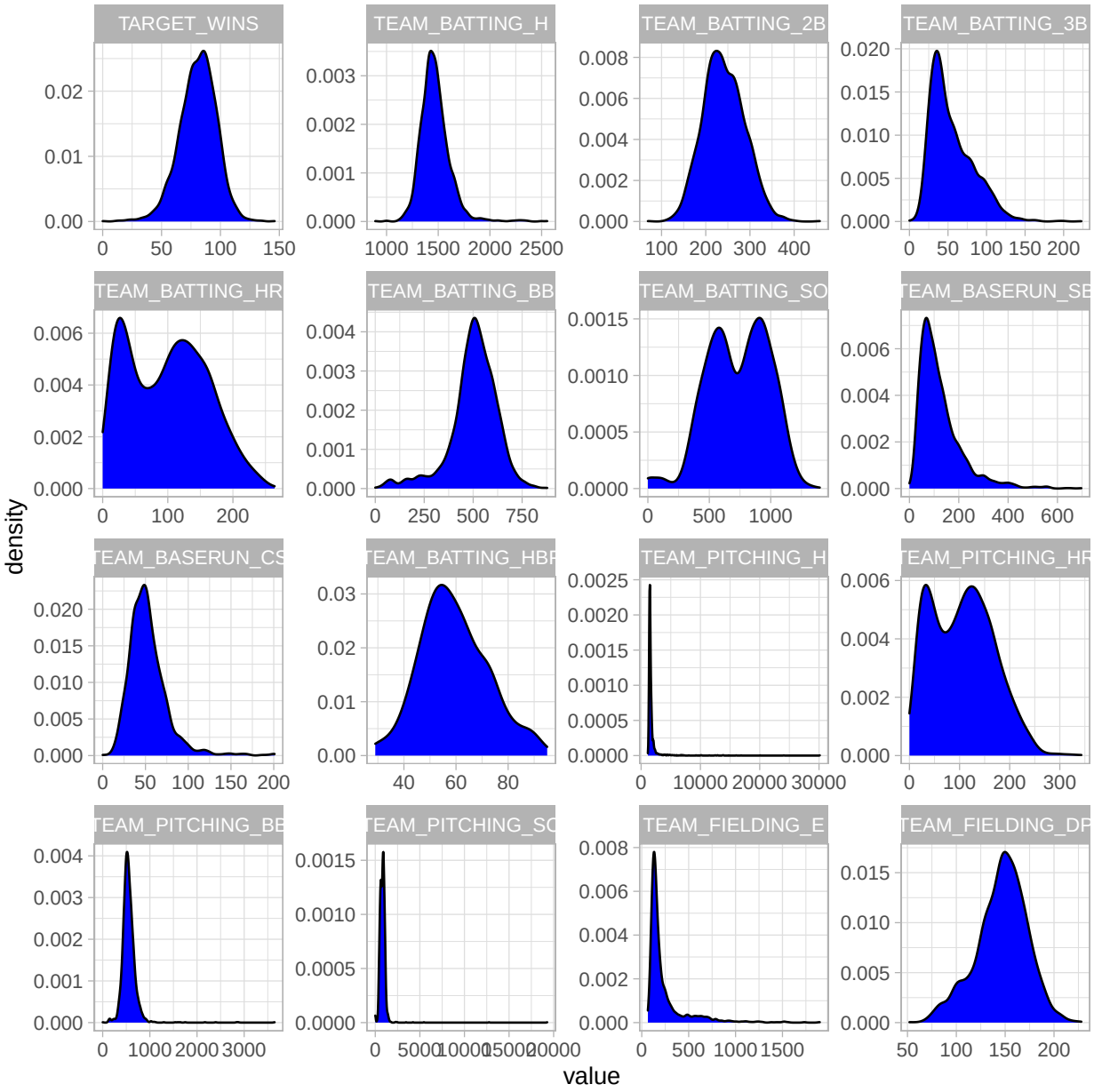
```
summary(train)
```

```
##   TARGET_WINS      TEAM_BATTING_H TEAM_BATTING_2B TEAM_BATTING_3B
##  Min.   :  0.00   Min.   : 891   Min.   : 69.0   Min.   :  0.00
##  1st Qu.: 71.00   1st Qu.:1383   1st Qu.:208.0   1st Qu.: 34.00
##  Median : 82.00   Median :1454   Median :238.0   Median : 47.00
##  Mean   : 80.79   Mean   :1469   Mean   :241.2   Mean   : 55.25
##  3rd Qu.: 92.00   3rd Qu.:1537   3rd Qu.:273.0   3rd Qu.: 72.00
##  Max.   :146.00   Max.   :2554   Max.   :458.0   Max.   :223.00
##
##  TEAM_BATTING_HR  TEAM_BATTING_BB TEAM_BATTING_SO  TEAM_BASERUN_SB
##  Min.   :  0.00   Min.   :  0.0   Min.   :   0.0   Min.   :  0.0
##  1st Qu.: 42.00   1st Qu.:451.0   1st Qu.: 548.0   1st Qu.: 66.0
##  Median :102.00   Median :512.0   Median : 750.0   Median :101.0
##  Mean   : 99.61   Mean   :501.6   Mean   : 735.6   Mean   :124.8
##  3rd Qu.:147.00   3rd Qu.:580.0   3rd Qu.: 930.0   3rd Qu.:156.0
##  Max.   :264.00   Max.   :878.0   Max.   :1399.0   Max.   :697.0
##                                   NA's   :102      NA's   :131
##  TEAM_BASERUN_CS TEAM_BATTING_HBP TEAM_PITCHING_H TEAM_PITCHING_HR
##  Min.   :  0.0   Min.   :29.00    Min.   : 1137   Min.   :  0.0
##  1st Qu.: 38.0   1st Qu.:50.50    1st Qu.: 1419   1st Qu.: 50.0
##  Median : 49.0   Median :58.00    Median : 1518   Median :107.0
##  Mean   : 52.8   Mean   :59.36    Mean   : 1779   Mean   :105.7
##  3rd Qu.: 62.0   3rd Qu.:67.00    3rd Qu.: 1682   3rd Qu.:150.0
##  Max.   :201.0   Max.   :95.00    Max.   :30132   Max.   :343.0
##  NA's   :772     NA's   :2085
##  TEAM_PITCHING_BB TEAM_PITCHING_SO  TEAM_FIELDING_E  TEAM_FIELDING_DP
##  Min.   :   0.0   Min.   :    0.0   Min.   :  65.0   Min.   : 52.0
##  1st Qu.: 476.0   1st Qu.:  615.0   1st Qu.: 127.0   1st Qu.:131.0
##  Median : 536.5   Median :  813.5   Median : 159.0   Median :149.0
##  Mean   : 553.0   Mean   :  817.7   Mean   : 246.5   Mean   :146.4
##  3rd Qu.: 611.0   3rd Qu.:  968.0   3rd Qu.: 249.2   3rd Qu.:164.0
##  Max.   :3645.0   Max.   :19278.0   Max.   :1898.0   Max.   :228.0
##                   NA's   :102                        NA's   :286
```

Check Data Distribution

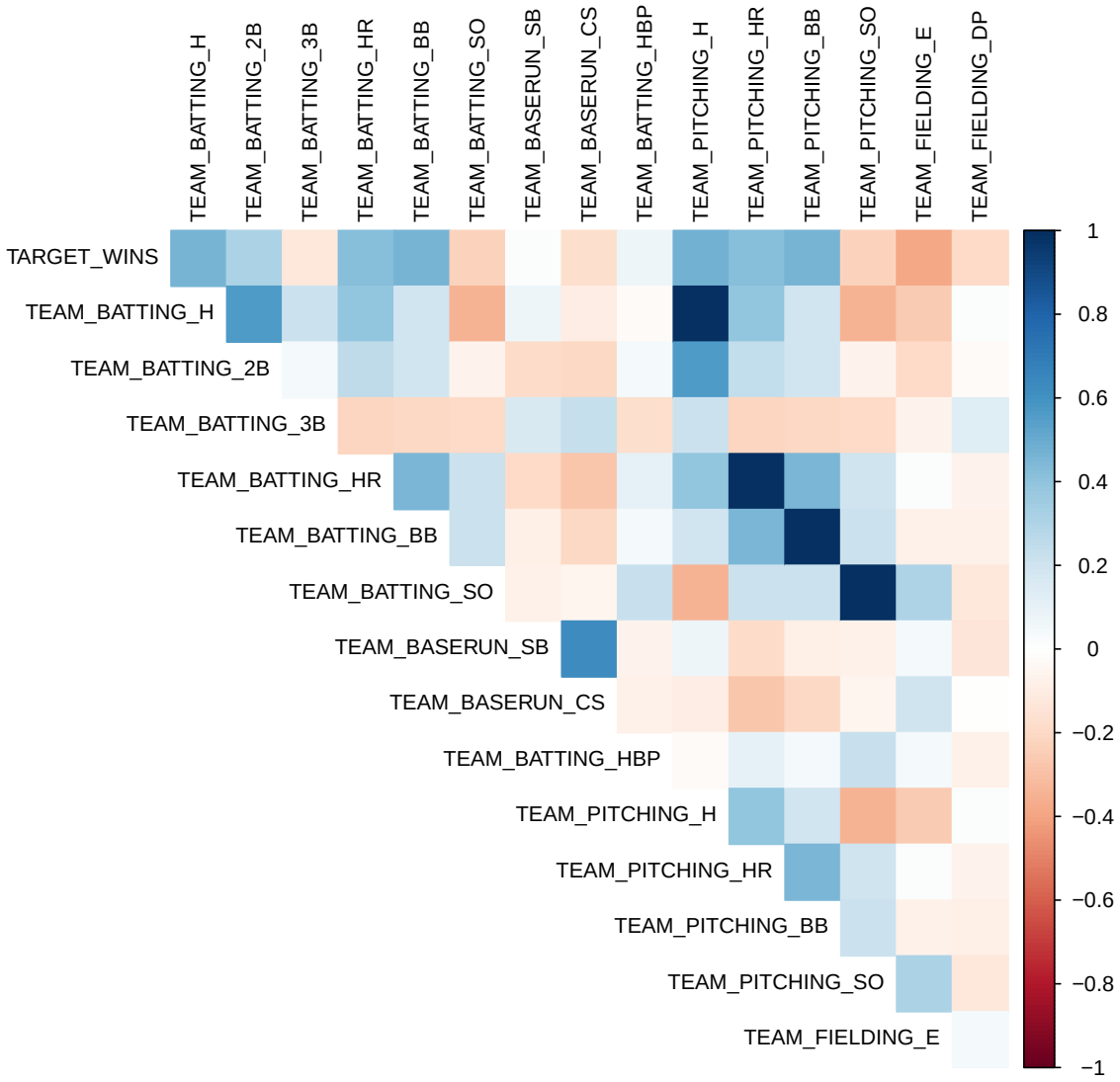
```
g = melt(train)
ggplot(g, aes(x = value)) +
    geom_density(fill = "blue") +
    facet_wrap(~variable,
```

```
        scales = "free") +
    theme_light()
```



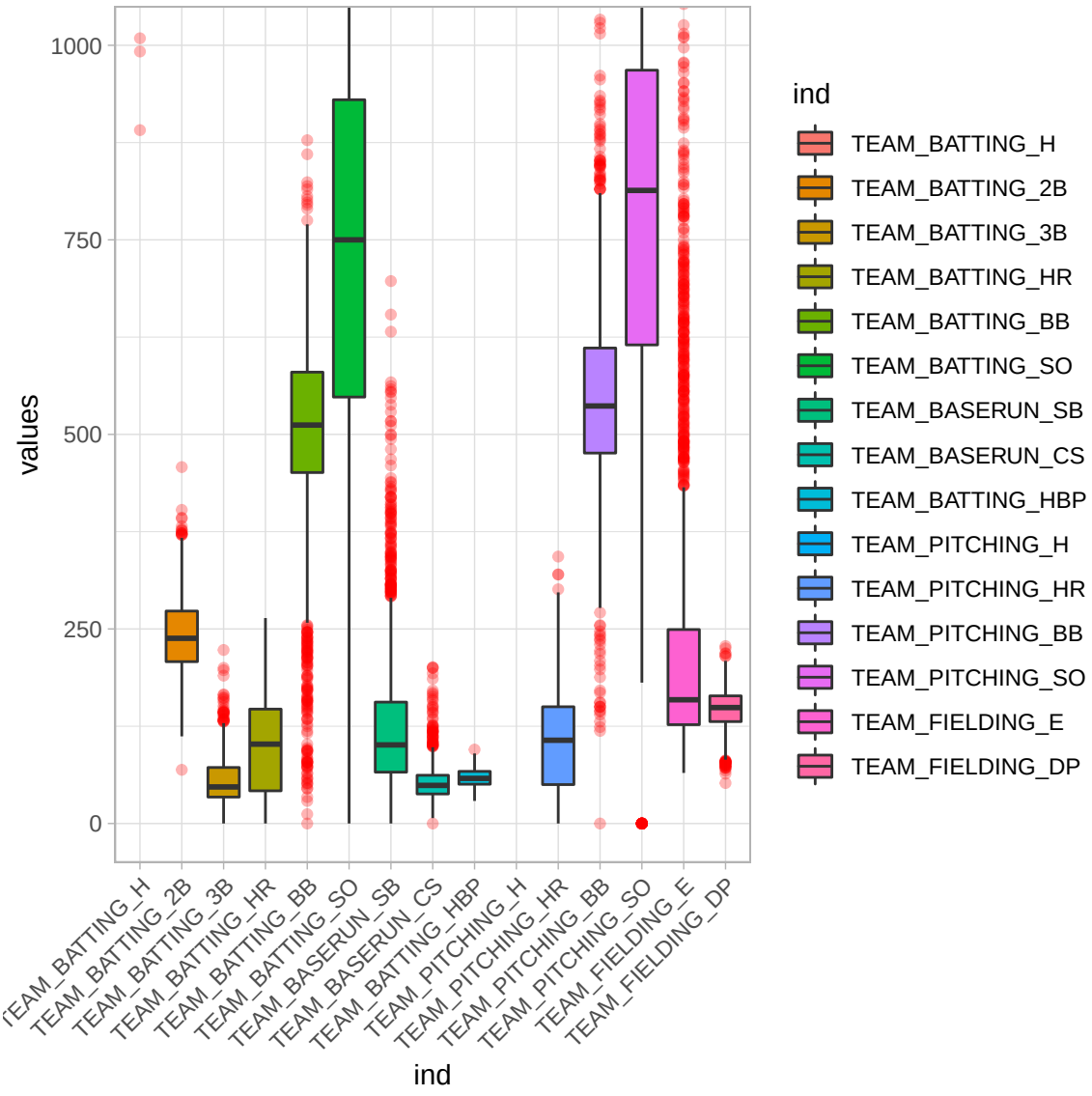
Look at Correltions between variable

```
train %>% cor(., use = "complete.obs") %>%
    corrplot(., method = "color",
        type = "upper",
        tl.col = "black",
        tl.cex = 0.8,
        diag = FALSE)
```



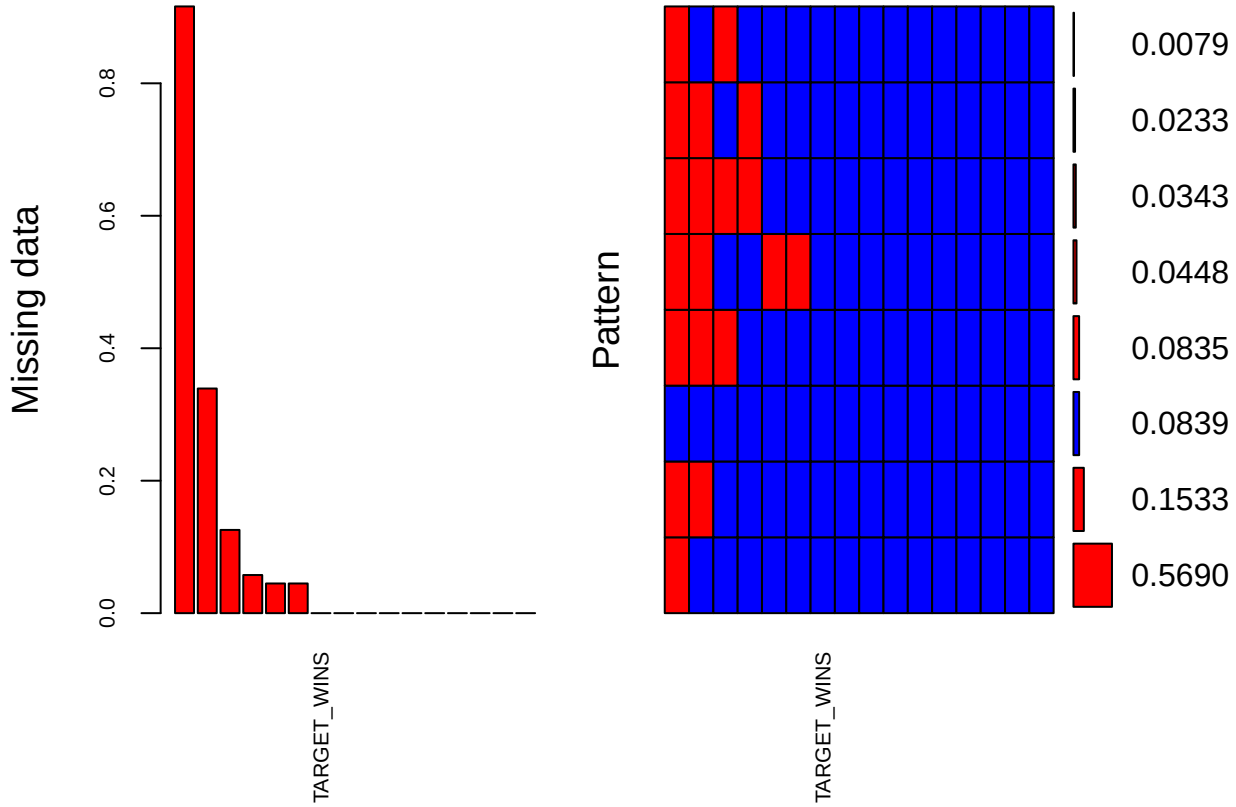
Explore the outliers

```
ggplot(stack(train[,
    -1]), aes(x = ind,
    y = values, fill = ind)) +
    geom_boxplot(outlier.colour = "red",
        outlier.alpha = 0.3) +
    coord_cartesian(ylim = c(0,
        1000)) + theme_light() +
    theme(axis.text.x = element_text(angle = 45,
        hjust = 1))
```



Plot Missing Data

```
missing_plot <- aggr(train,
    col = c("blue", "red"),
    numbers = TRUE, sortVars = TRUE,
    labels = names(train),
    cex.axis = 0.7, gap = 3,
    ylab = c("Missing data",
        "Pattern"))
```




```
##
##  Variables sorted by number of missings:
##          Variable      Count
##  TEAM_BATTING_HBP 0.91608084
##   TEAM_BASERUN_CS 0.33919156
##  TEAM_FIELDING_DP 0.12565905
##   TEAM_BASERUN_SB 0.05755712
##   TEAM_BATTING_SO 0.04481547
##  TEAM_PITCHING_SO 0.04481547
##       TARGET_WINS 0.00000000
##    TEAM_BATTING_H 0.00000000
##   TEAM_BATTING_2B 0.00000000
##   TEAM_BATTING_3B 0.00000000
##   TEAM_BATTING_HR 0.00000000
##   TEAM_BATTING_BB 0.00000000
##   TEAM_PITCHING_H 0.00000000
##  TEAM_PITCHING_HR 0.00000000
##  TEAM_PITCHING_BB 0.00000000
##   TEAM_FIELDING_E 0.00000000
```

Model Development

The below cells go through three different modeling methods and the result of each. Each method contains slightly different ways of preparing the data and we explore the impact that has on our evaluation metric R2. The reason we chose to use R2 is to compare the results of the model across these different approaches.

Method 1

Code Author: Christopher Bloome

This method cleans the data in a way where the missing data is imputed with the median and focusses on feature engineering. Over the next few cells you will look at a bunch of different features that were created from the existing one and then fit a linear model, and comparing the results.

Data Prep

```
# Remove
# TEAM_BATTING_HBP as
# it has 91.6%
# missing values
method1_train <- train[-10]

# Replace extreme
# outliers with
# median
method1_train <- method1_train %>%
    mutate(TEAM_PITCHING_H = ifelse(TEAM_PITCHING_H >
        5000, median(TEAM_PITCHING_H),
        TEAM_PITCHING_H),
        TEAM_PITCHING_SO = ifelse(TEAM_PITCHING_SO >
            1500, median(TEAM_PITCHING_SO),
            TEAM_PITCHING_SO))

method1_test <- test %>%
    mutate(TEAM_PITCHING_H = ifelse(TEAM_PITCHING_H >
        5000, median(TEAM_PITCHING_H),
        TEAM_PITCHING_H),
        TEAM_PITCHING_SO = ifelse(TEAM_PITCHING_SO >
            1500, median(TEAM_PITCHING_SO),
            TEAM_PITCHING_SO))

# Replace missing
# values with median
method1_train[] <- lapply(method1_train,
    function(x) ifelse(is.na(x),
        median(x, na.rm = TRUE),
        x))
method1_test[] <- lapply(method1_test,
    function(x) ifelse(is.na(x),
        median(x, na.rm = TRUE),
        x))
```

Feature Engineering

Now that we have the data reasonably clean, we can calculate some additional fields that may be of use in our models.

SLG 1

There exists a metric called "Slugging percentage" used by baseball teams in the modern era, that effectively determines the effectiveness of a hitter. This is traditionally calculated as follows: [Singles + Doubles x 2 + Triples x 3 + HRs x 4]/[Plate Appearances].

As we do not have Plate Appearances, lets use hits in its place, effectively calculating how many bases a team gets on average each time one of its players gets a hit.

```
method1_train$SLG1 <- (method1_train$TEAM_BATTING_H +
    method1_train$TEAM_BATTING_2B +
    2 * method1_train$TEAM_BATTING_3B +
    3 * method1_train$TEAM_BATTING_HR)/method1_train$TEAM_BATTING_H
```

SLG 2

While the above demonstrates the effectiveness of a team at the plate, each of the other statistics are aggregate sums. In that spirit, lets calculate a "slugging total" that is not scaled to the number of hits.

```
method1_train$SLG2 <- (method1_train$TEAM_BATTING_H +
    method1_train$TEAM_BATTING_2B +
    2 * method1_train$TEAM_BATTING_3B +
    3 * method1_train$TEAM_BATTING_HR)
```

Diffs

Due to the changes in the way baseball has been played since its inception, aggregate totals may be less predictive them ratios of a team's outcomes vs their opponents. Lets create 3 variables, that measure ratios in this way.

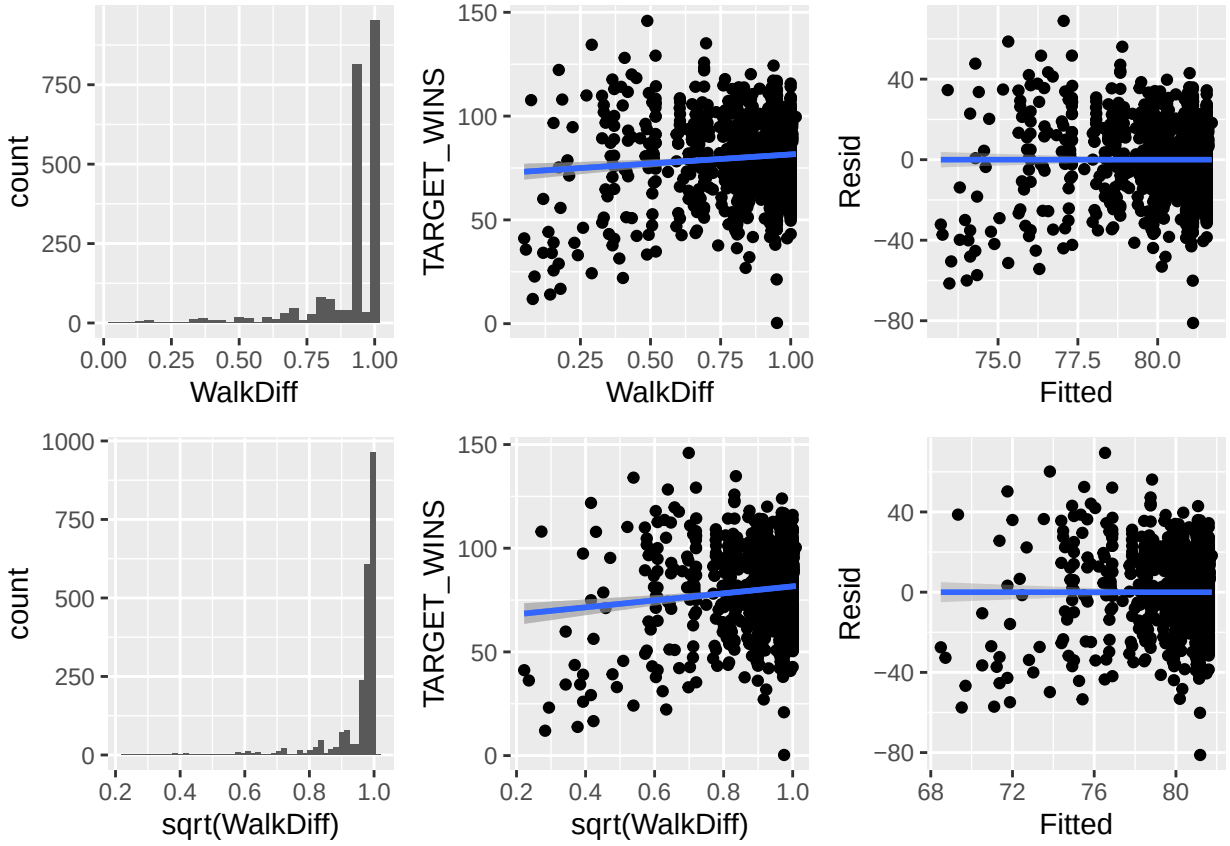
As we will be building a learning model, we would not want to generate differences by way of subtracting, as they would be deemed redundant later on.

```
method1_train$HitDiff <- method1_train$TEAM_BATTING_H/method1_train$TEAM_PITCHING_H

method1_train$WalkDiff <- method1_train$TEAM_BATTING_BB/method1_train$TEAM_PITCHING_BB

method1_train$HRDiff <- method1_train$TEAM_BATTING_HR/method1_train$TEAM_PITCHING_HR

# Change NAs to
# Median - While NAs
# are caused by a
# value of 0 - cases
# where the
# denominator are 0
# are more likely due
# to missing data
# than truly having
# no Hits, Walks or
# HRs in a season.
```

```
method1_train[] <- lapply(method1_train,
    function(x) ifelse(is.na(x),
        median(x, na.rm = TRUE),
        x))
```

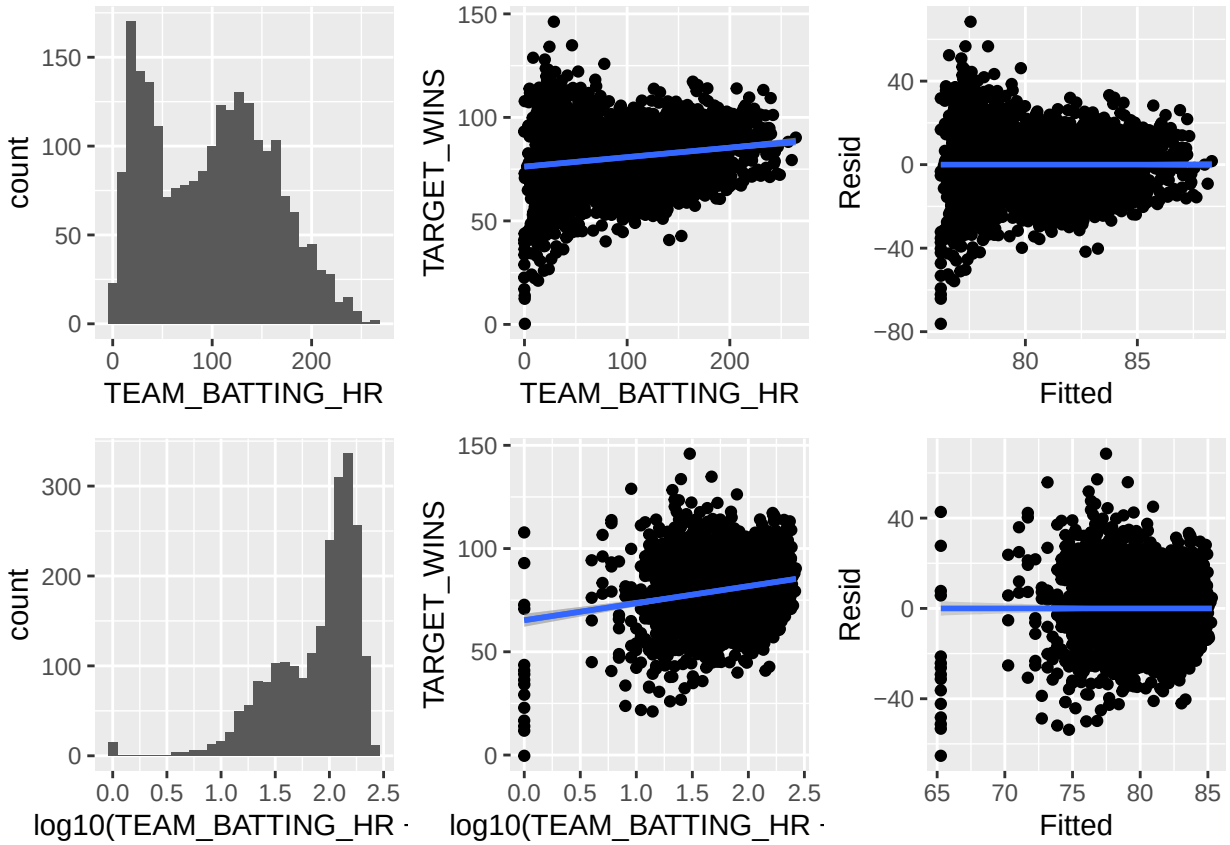
Walk Diff

Lets start with one of our calculated variables, WalkDiff. This variable will always be between 0 and 1. By taking the squareroot, are vaules remain between 0 and 1, but we effectively compress values such that the better performances become closer together, and worse performances become outliers. This results in a significantly more predictive variable.



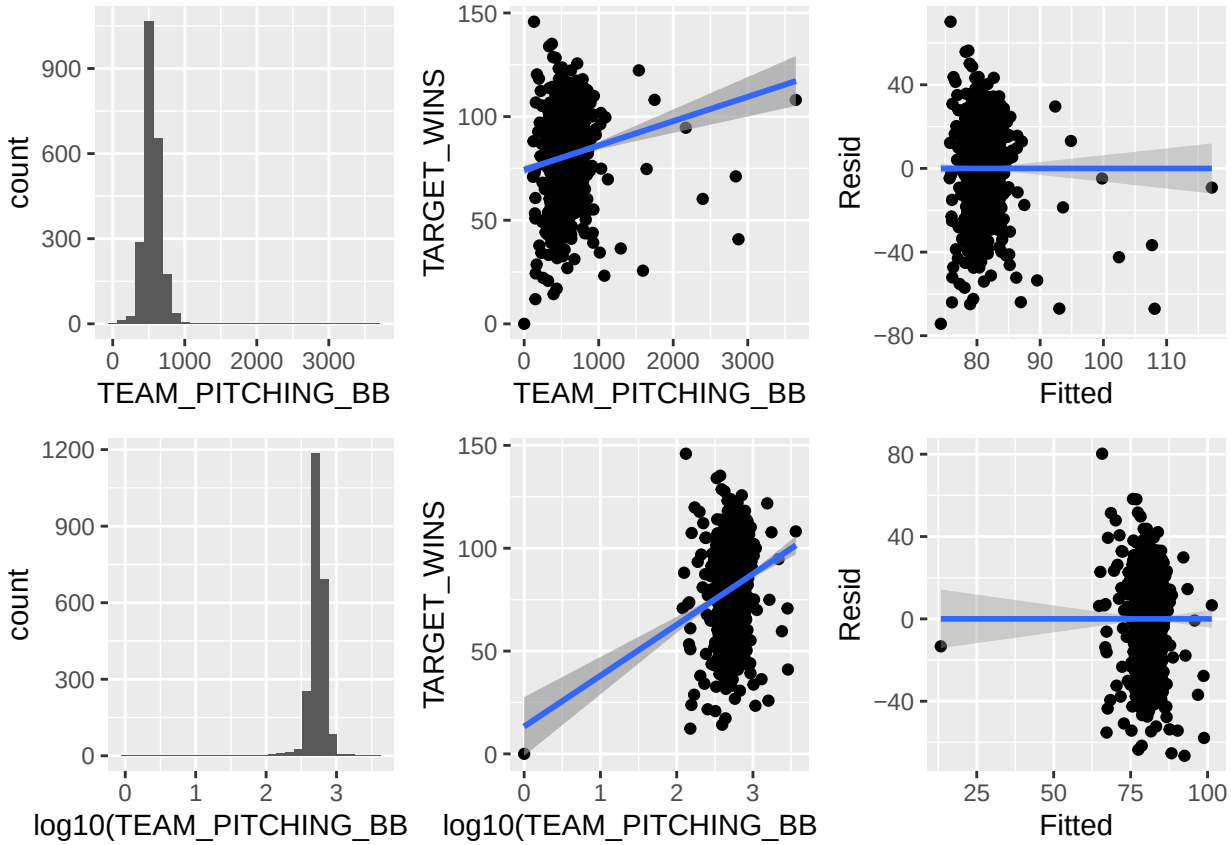
TEAM_BATTING_HR

We see that Home Runs is bi-model in nature, likely due to variation in the way the game has evolved. If we were to subset this data by era, we might see several distinct normal distributions. By taking the Log, we find the bi-model distribution becomes more normal, and that the range changes in a manner more fitting for a linear model. Reviewing our residual plots, we find that before the transformation, we have a much higher degree of variation for lower values. This is largely fixed after the transformation.



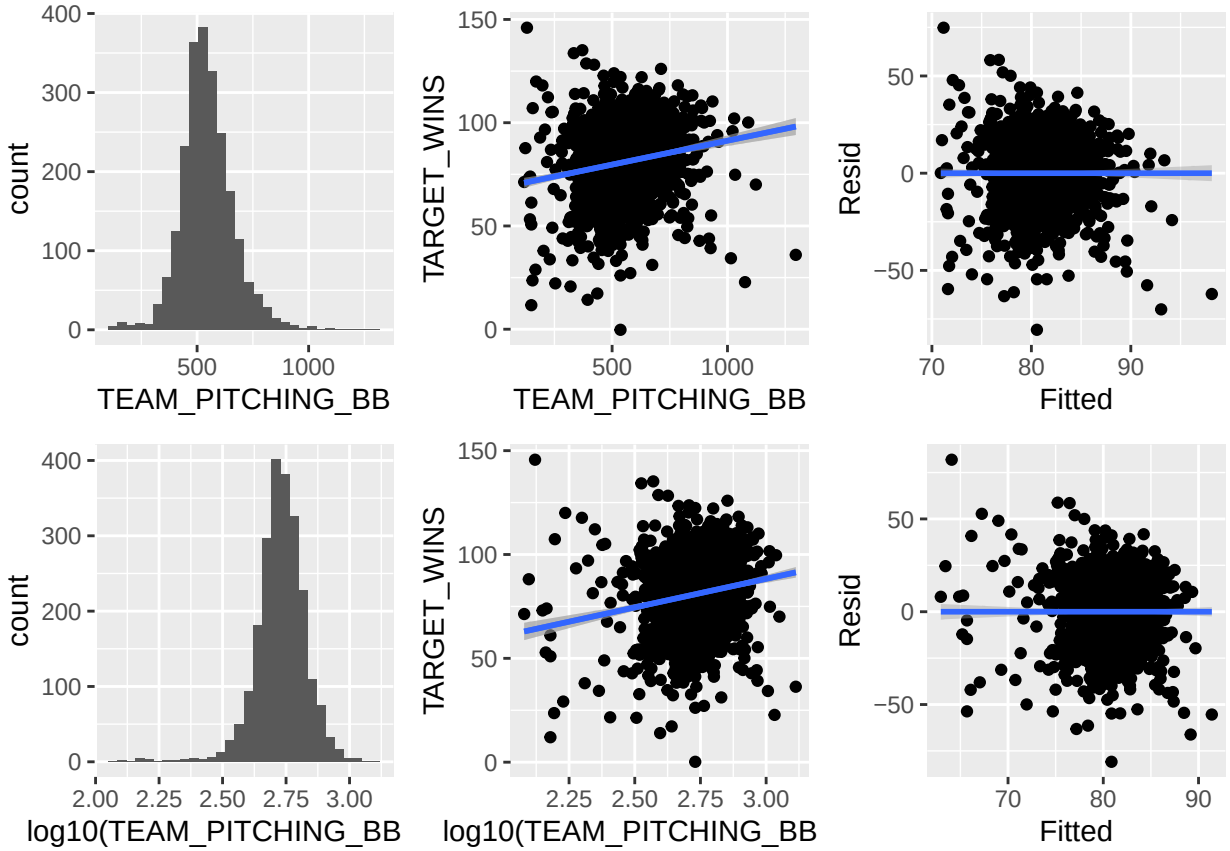
TEAM_PITCHING_BB

At first glance, it appears that a log transformation increases the predictive power of the pitching/walks metric.



However, after a closer look, we find that this is actually due to the treatment of outliers. Notice the extreme high values before the transformation, and the low value after the transformation.

We can accomplish an increased level of predictiveness by removing these extremes with the median of the set. It is likely that these are in fact errors in the data.



The log transformation does improve our predictive power but only slightly.

Train Model & Evaluate Results

In the end, we learned a few things from this method:

- We found that higher order hits were generally not predictive, likely due to their rarity. Triples were eliminated, though HR and the HR ratios remained.
- While the square root transformation seemed more predictive in our prep, it actually made the model worse and was ultimately removed.
- The Log transformations and the truncating of Walks proved very effective.

```
model.1c <- lm(TARGET_WINS ~
    TEAM_BATTING_H +
        TEAM_BATTING_2B +
        log10(TEAM_BATTING_HR +
            1) + TEAM_BATTING_BB +
        TEAM_PITCHING_H +
        TEAM_PITCHING_HR +
        log10(TEAM_PITCHING_BB +
            1) + TEAM_FIELDING_E +
```

```
        TEAM_BASERUN_SB +
        TEAM_PITCHING_SO +
        TEAM_FIELDING_DP +
        SLG1 + SLG2 +
        HRDiff + WalkDiff,
    data = method1_train)


my.summary.lm(summary(model.1c),
    my.rows = 1:4)
```

```
##
## Call:
## lm(formula = TARGET_WINS ~ TEAM_BATTING_H + TEAM_BATTING_2B +
##     log10(TEAM_BATTING_HR + 1) + TEAM_BATTING_BB + TEAM_PITCHING_H +
##     TEAM_PITCHING_HR + log10(TEAM_PITCHING_BB + 1) + TEAM_FIELDING_E +
##     TEAM_BASERUN_SB + TEAM_PITCHING_SO + TEAM_FIELDING_DP + SLG1 +
##     SLG2 + HRDiff + WalkDiff, data = method1_train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -49.407  -8.200   0.274   8.142  64.091
##
## Coefficients:
##                              Estimate Std. Error t value Pr(>|t|)
## (Intercept)                 -1.99764   43.45865  -0.046  0.96334
## TEAM_BATTING_H               0.07495    0.02598   2.885  0.00395 **
## TEAM_BATTING_2B             -0.06433    0.01117  -5.759 9.59e-09 ***
## log10(TEAM_BATTING_HR + 1) -6.21839    2.63625  -2.359  0.01842 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 12.81 on 2260 degrees of freedom
## Multiple R-squared:  0.3433, Adjusted R-squared:  0.339
## F-statistic: 78.77 on 15 and 2260 DF,  p-value: < 2.2e-16
```

Method 2

Code Author: Adam Gersowitz

This model's approach was to attempt to predict the year that these statistics took place using rates of key statistics that were found on baseball-reference.com. After creating a linear model based on these ratios to predict the year we will then use that model to predict the year of our test dataset. The Year prediction model had an R-squared > 0.96 which indicates it is extremely accurate and can be relied upon to predict the year of our dataset.

Once we have our predicted year we will then create dummy variables based on widely agreed upon eras in the history of baseball. Finally, our model will be based on the interaction of these eras and the counting statistics that were given to us. For example, TEAM_BATTING_HITS*era_modern will produce 0 if the team was not predicted to have played in the modern era but will be the number of hits the team had if they did play in the modern era. This essentially creates features that are "number of hits in the modern era" vs "number of hits". This allows us to get a better understanding of which statistics were more important in which eras.

Reference Links: Statistics by year
https://www.baseball-reference.com/leagues/MLB/bat.shtml
Three True Outcomes
https://www.mlb.com/glossary/idioms/three-true-outcomes#:~:text=
The%20%22three%20true%20outcomes%22%20in,the%20pitcher%20or%20the%20catcher
Eras of Baseball
https://thesportjournal.org/article/examining-perceptions-of-baseballs-eras/#
:~:text=A%20common%20list%20presented%20at,%2D2005)%20(17)

Train Model & Evaluate Results

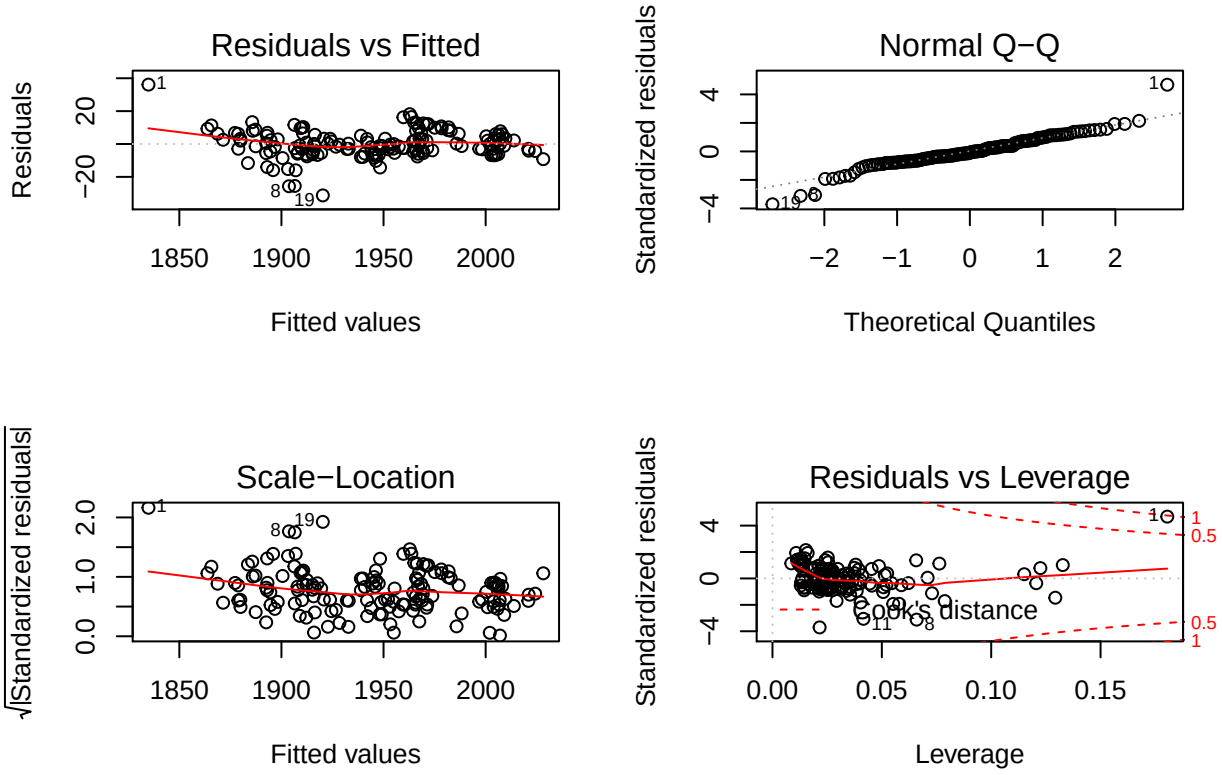
Baseball, perhaps more than any other sport, is defined by it's eras. These range from a version of the game in the Dead-Ball era that was focused on "small-ball" type plays such as stolen bases, bunts, singles etc. This is drastically different from the modern game which focuses on the 3 true outcomes of the game (home runs, walks, strike outs) as the important counting statistics to focus on. Unfortunately, in our data set there is no indication of the year these statistics took place. this is particularly troubling because the dataset ranges over 100 years of baseball which has seen its fair share of evolution.

```
method2_train$X2B = (method2_train$TEAM_BATTING_2B/162)
method2_train$X3B = (method2_train$TEAM_BATTING_3B/162)
method2_train$BB = ((method2_train$TEAM_BATTING_BB/162) +
    (method2_train$TEAM_PITCHING_BB/162))/2
method2_train$SO = ((method2_train$TEAM_BATTING_SO/162) +
    (method2_train$TEAM_PITCHING_SO/162))/2

year <- lm(Year ~ X2B +
    X3B + BB + SO, data = yr)

summary(year)
```

```
##
## Call:
## lm(formula = Year ~ X2B + X3B + BB + SO, data = yr)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
```

```
## -31.269  -4.950  -0.753   5.447  36.113
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 1853.9638     7.3246 253.113  < 2e-16 ***
## X2B           37.8715     3.4045  11.124  < 2e-16 ***
## X3B         -113.4110     8.5250 -13.303  < 2e-16 ***
## BB             9.9854     1.1284   8.849 2.79e-15 ***
## SO            10.5675     0.7793  13.561  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 8.526 on 145 degrees of freedom
## Multiple R-squared:  0.9625, Adjusted R-squared:  0.9615
## F-statistic: 930.9 on 4 and 145 DF,  p-value: < 2.2e-16
```

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

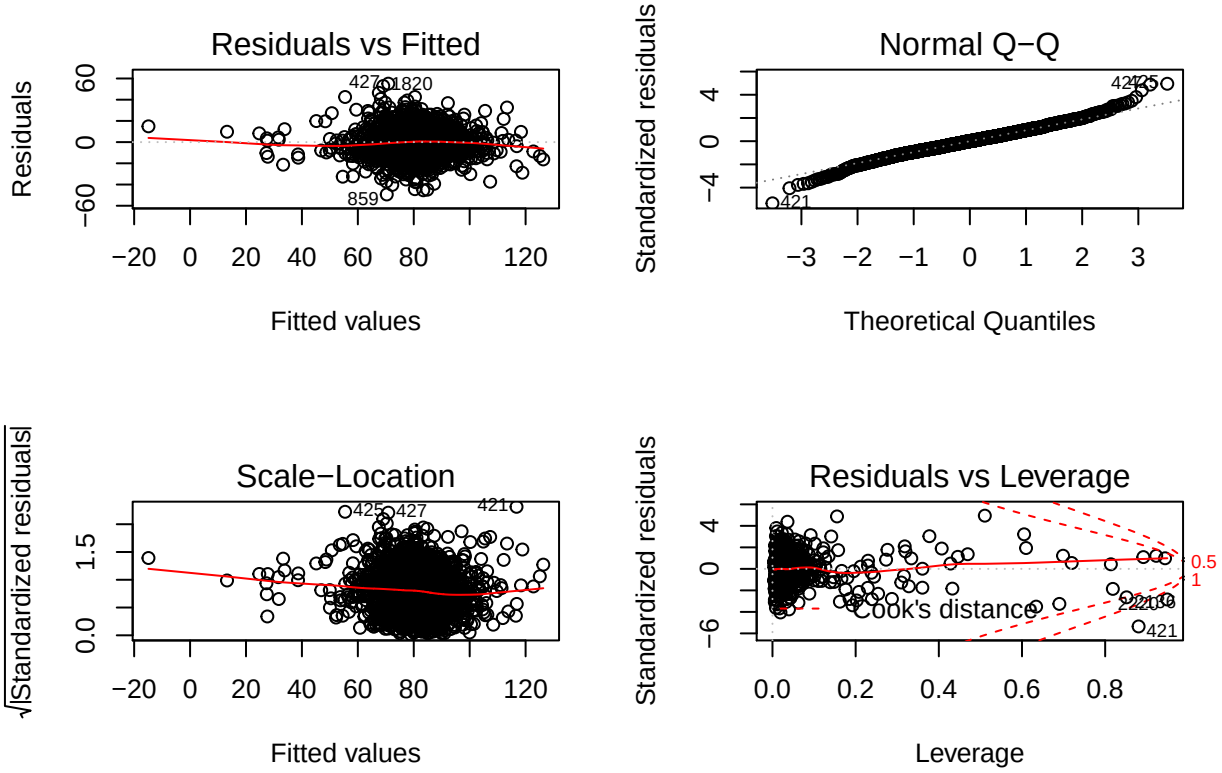



```
my.summary.lm(summary(era),
    my.rows = 1:4)
```

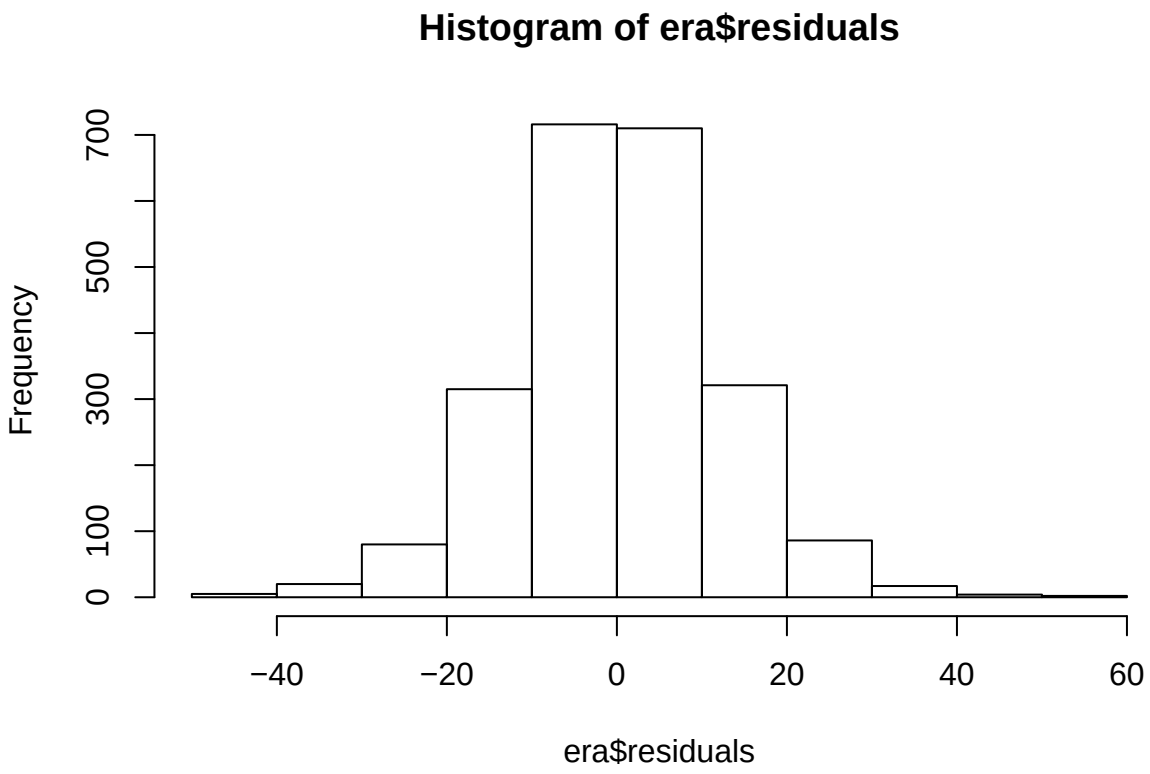
```
##
## Call:
## lm(formula = TARGET_WINS ~ H_era_m + H_era_fa + H_era_e + H_era_i +
```

```
##     H_era_lb + H_era_db + H_era_m_p + H_era_fa_p + H_era_e_p +
##     H_era_i_p + H_era_lb_p + H_era_db_p + bb_era_m + bb_era_fa +
##     bb_era_e + bb_era_i + bb_era_lb + bb_era_db + bb_era_m_p +
##     bb_era_fa_p + bb_era_e_p + bb_era_i_p + bb_era_lb_p + bb_era_db_p +
##     hr_era_m + hr_era_fa + hr_era_e + hr_era_i + hr_era_lb +
##     hr_era_db + hr_era_m_p + hr_era_fa_p + hr_era_e_p + hr_era_i_p +
##     hr_era_lb_p + hr_era_db_p + so_era_m + so_era_fa + so_era_e +
##     so_era_i + so_era_lb + so_era_db + so_era_m_p + so_era_fa_p +
##     so_era_e_p + so_era_i_p + so_era_lb_p + so_era_db_p + x2b_era_m +
##     x2b_era_fa + x2b_era_e + x2b_era_i + x2b_era_lb + x2b_era_db +
##     x3b_era_m + x3b_era_fa + x3b_era_e + x3b_era_i + x3b_era_lb +
##     x3b_era_db + e_era_m + e_era_fa + e_era_e + e_era_i + e_era_lb +
##     e_era_db + dp_era_m + dp_era_fa + dp_era_e + dp_era_i + dp_era_lb +
##     dp_era_db + sb_era_m + sb_era_fa + sb_era_e + sb_era_i +
##     sb_era_lb + sb_era_db, data = method2_train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -49.346  -7.716   0.025   7.535  55.109
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 26.514945   6.358395   4.170 3.16e-05 ***
## H_era_m      0.024951   0.012053   2.070   0.0386 *
## H_era_fa     0.006729   0.029309   0.230   0.8184
## H_era_e     -0.020592   0.024759  -0.832   0.4057
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 12.3 on 2197 degrees of freedom
## Multiple R-squared:  0.4115, Adjusted R-squared:  0.3906
## F-statistic:  19.7 on 78 and 2197 DF,  p-value: < 2.2e-16
```

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




```r
hist(era$residuals)
```



This model show promise with a relatively high r-squared of 0.4115. However, it needs cleaning and more advance feature selection. The residuals for this model seem close to normally distributed with QQ plot that has heavy tails which is driven by high leverage outliers. While this model takes advantage of publicly available data to achieve more accurate results, it still needs refining. In the next model we will take some of the elements from model 2 and apply them to a more thorough feature selection and deal with some other outstanding issues such as colinearity.

Method 3

Code Author: David Blumenstiel

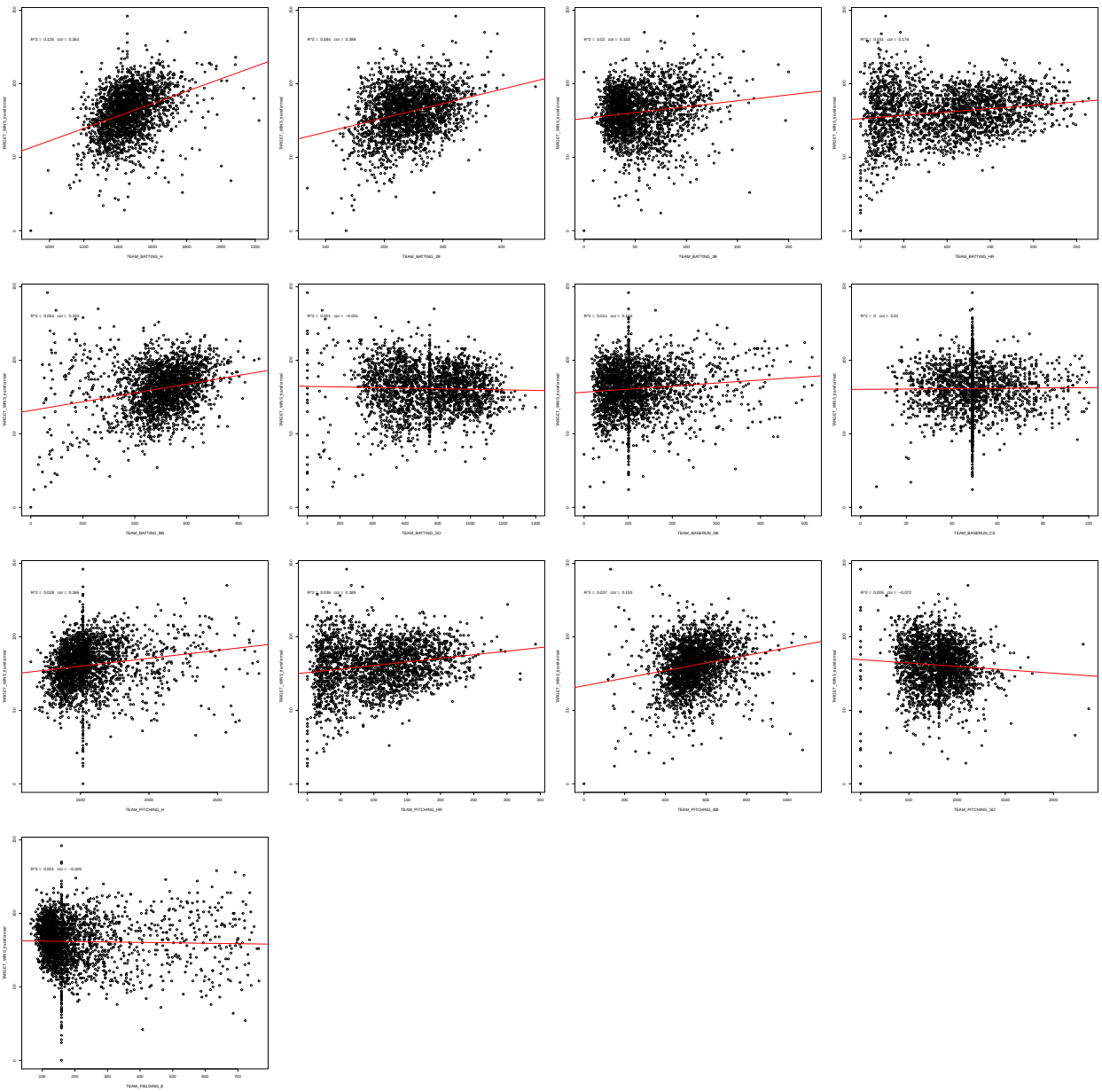
Data Prep & Feature Engineering

```
## [1] "set 35 outliers to median"
```

```
## [1] "set 277 outliers to median"
```

```
# Explore Transformed
# Relationships
layout(matrix(c(1, 2,
    3, 4, 5, 6, 7, 8,
    9, 10, 11, 12, 13,
    14, 15, 16), byrow = TRUE,
    ncol = 4, nrow = 4))
```

```r
# par(mfrow=c(4,4))
names = colnames(training)
for (i in 3:length(training) -
    1) {
    name = names[i]
    plot(training$TARGET_WINS ~
        training[, i],
        xlab = name,
        ylab = "TARGET_WINS_transformed")
    l = lm(training$TARGET_WINS ~
        training[, i])
    abline(l, col = "red")
    text(min(training[,
        i]), 130, paste("R^2 = ",
        round(summary(l)$r.squared,
            3), "  cor = ",
        round(cor(y = training$TARGET_WINS,
            x = training[,
                i]),
            3)), adj = 0)
}
```

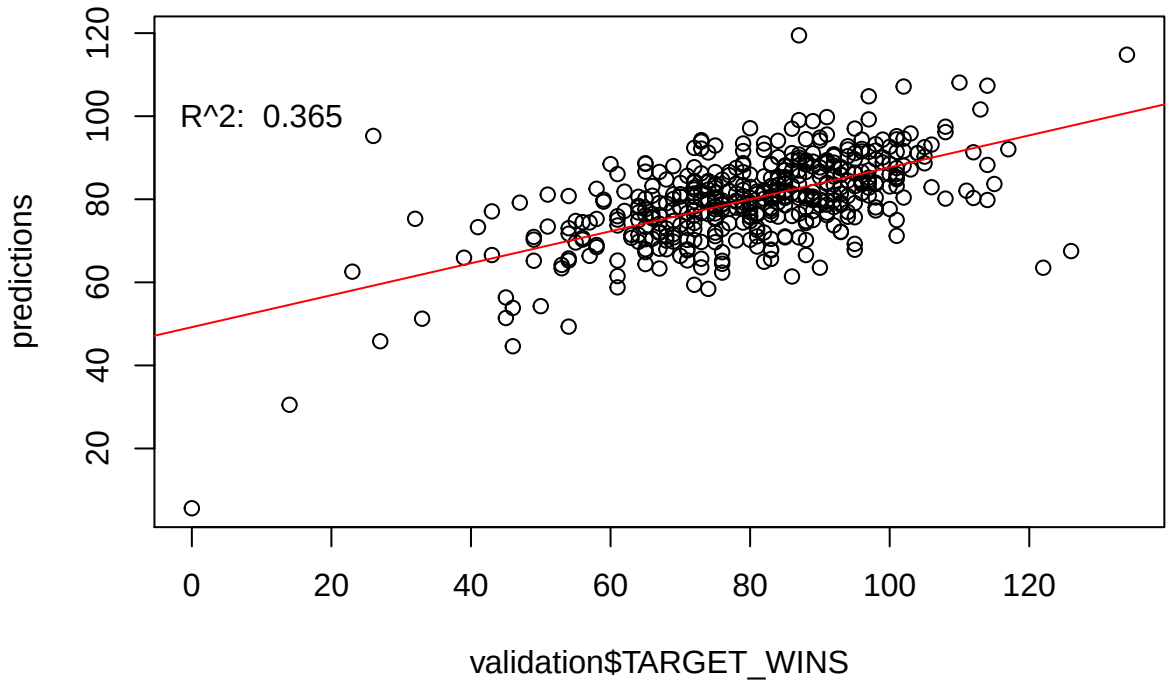


Model 1

This model utilizes all variables except the pitching ones, which are pretty much co-linear with the batting ones. This uses a stepwise method to narrow down the parameters, and checks all interactions terms 2 layers deep. 5 fold cv is used with 2 repeats. A seperate validation set is held in reserve to gauge accuracy (stepwise regression is known for overfitting).

```
set.seed(1234567890)  #So you see the same thing I do

# Set's up some
# K-fold validation
# and repetition
tr <- trainControl(method = "repeatedcv",
```

```
    number = 5, repeats = 2)

# Set's up
# validations set,
# which won't get
# touched during
# stepwise
# regression.  Gives
# us a better sense
# of performance
split <- createDataPartition(training$TARGET_WINS,
    p = 0.8, list = FALSE)
method3_train <- training[split,
    ]
validation <- training[-split,
    ]

# Set's up the model
model <- train(TARGET_WINS ~
    (TEAM_BATTING_H +
        TEAM_BATTING_2B +
        TEAM_BATTING_3B +
        TEAM_BATTING_HR +
        TEAM_BATTING_BB +
        TEAM_BATTING_SO +
        TEAM_BASERUN_SB +
        TEAM_BASERUN_CS +
        TEAM_FIELDING_E +
        TEAM_FIELDING_DP)^2,
    data = method3_train,
    method = "lmStepAIC",
    trControl = tr, trace = FALSE)  #Supress the trace or else this will eat your screen

predictions <- predict(model,
    validation)
l <- lm(predictions ~
    validation$TARGET_WINS)

# View at results on
# validation data
plot(predictions ~ validation$TARGET_WINS)
text(10, 100, paste("R^2: ",
    round(summary(l)$r.squared,
        3)))
abline(l$coefficients,
    col = "red")
```



This model performs fairly well, with R^2 near 0.365 on the holdout set. There seems to be a balancing act when it comes to outliers: a higher tolerance for keeping outliers leads to higher R^2 values, but more heteroscedascity. As is now (with an outlier tolerance of $median \pm 5 * IQR$, there's a slight heteroscedascity and heavy tails, but the model has a close fit.

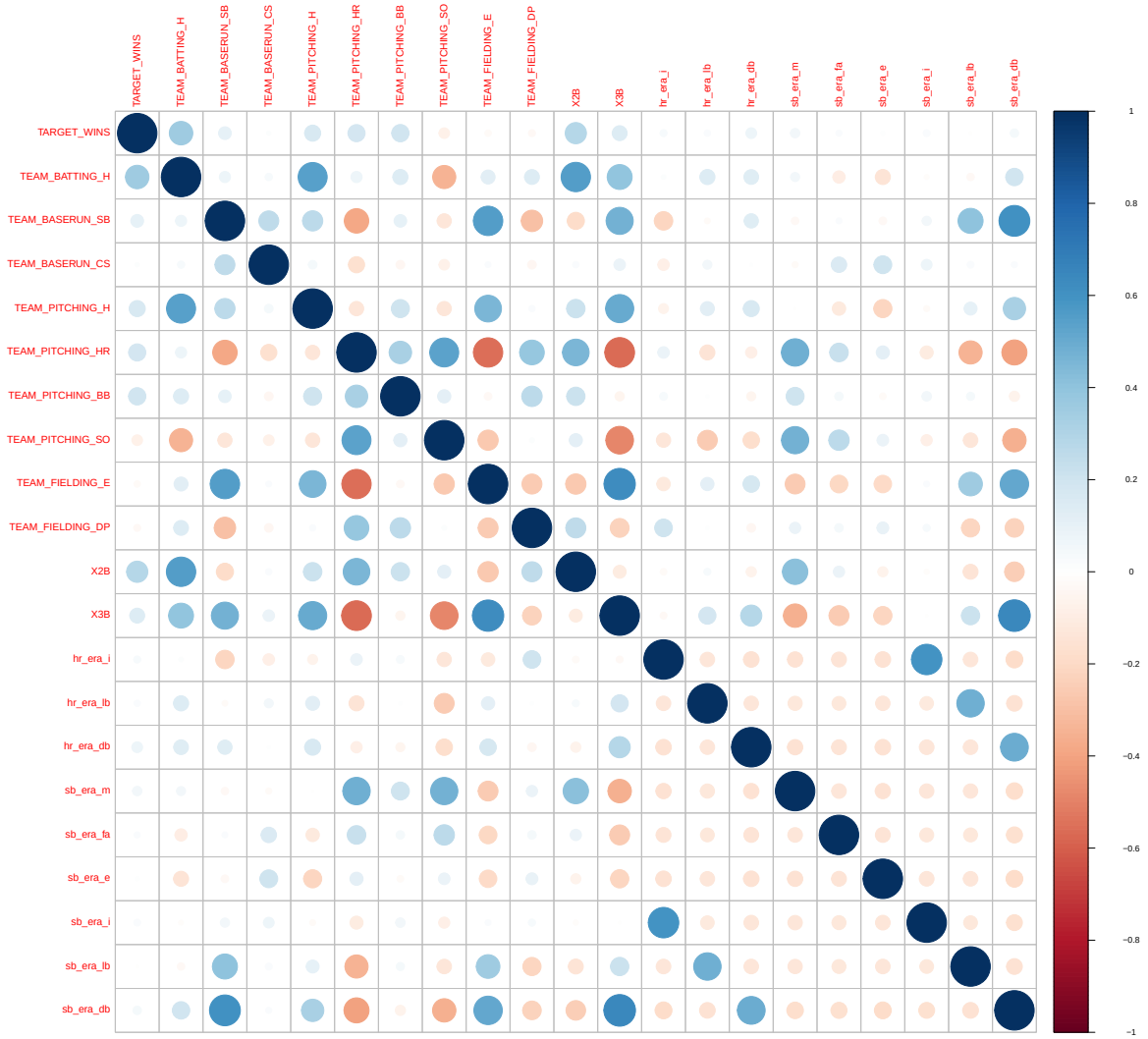
This model also uses some predictors which, by themselves, are not very predictive. In addition, Stepwise regression is also known to cause problems, and can introduce biases which inflate the R^2. However, the model performs well on the validation set.

Model 2

First, we prepair the new features

There's alot of collinearity among the new variables. Below we drop collinear terms.

```
method3_train$era <- NULL  #Gets rid of the only non numeric value
dropem <- findCorrelation(cor(method3_train),
    cutoff = 0.7)  #Finds collinear variables
remove_names <- colnames(method3_train)[dropem]  #Save for later
method3_train <- method3_train[,
    -dropem]  #Drops collinear variables
corrplot(cor(method3_train))
```



Now to actually make/download the model. This will look amongst both the origional variables and new ones, excluding colinear terms. It will look to interaction terms deep, and do stepwise selection to pick narrow down the amount of coefficients used in the final model. The stepwise selection process takes quite a while, so the chunk below is going to try to download the pre-trained model instead of training it again.

```
# Make predictions on
# validation set
predictions <- predict(era_model_stepped,
    validation_set)
```

```
## Warning in predict.lm(era_model_stepped, validation_set): prediction from a
## rank-deficient fit may be misleading
```

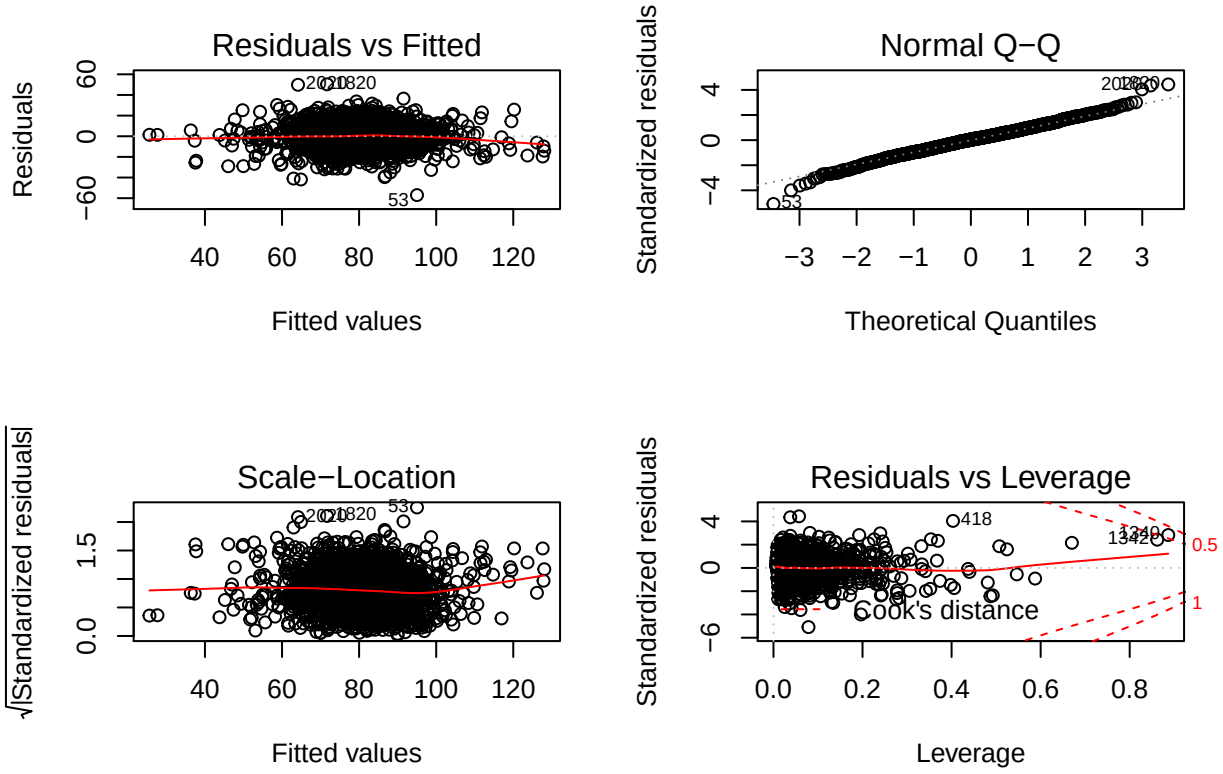
```r
l <- lm(predictions ~
    validation_set$TARGET_WINS)

# View at results on
# validation data
my.summary.lm(summary(era_model_stepped),
    my.rows = 1:4)
```

```
##
## Call:
## lm(formula = TARGET_WINS ~ TEAM_BATTING_H + TEAM_BASERUN_SB +
##     TEAM_BASERUN_CS + TEAM_PITCHING_H + TEAM_PITCHING_HR + TEAM_PITCHING_BB +
##     TEAM_PITCHING_SO + TEAM_FIELDING_E + TEAM_FIELDING_DP + X2B +
##     X3B + hr_era_i + hr_era_lb + hr_era_db + sb_era_m + sb_era_fa +
##     sb_era_e + sb_era_i + sb_era_lb + sb_era_db + TEAM_BATTING_H:TEAM_PITCHING_H +
##     TEAM_BATTING_H:TEAM_PITCHING_BB + TEAM_BATTING_H:TEAM_PITCHING_SO +
##     TEAM_BATTING_H:TEAM_FIELDING_E + TEAM_BATTING_H:X2B + TEAM_BATTING_H:X3B +
##     TEAM_BATTING_H:hr_era_lb + TEAM_BATTING_H:sb_era_fa + TEAM_BATTING_H:sb_era_e +
##     TEAM_BATTING_H:sb_era_i + TEAM_BATTING_H:sb_era_lb + TEAM_BASERUN_SB:TEAM_BASERUN_CS +
##     TEAM_BASERUN_SB:TEAM_PITCHING_H + TEAM_BASERUN_SB:TEAM_PITCHING_HR +
##     TEAM_BASERUN_SB:TEAM_FIELDING_E + TEAM_BASERUN_SB:X3B + TEAM_BASERUN_SB:hr_era_db +
##     TEAM_BASERUN_SB:sb_era_m + TEAM_BASERUN_SB:sb_era_lb + TEAM_BASERUN_SB:sb_era_db +
##     TEAM_BASERUN_CS:TEAM_FIELDING_E + TEAM_BASERUN_CS:X2B + TEAM_BASERUN_CS:X3B +
##     TEAM_BASERUN_CS:hr_era_lb + TEAM_BASERUN_CS:sb_era_m + TEAM_BASERUN_CS:sb_era_fa +
##     TEAM_BASERUN_CS:sb_era_e + TEAM_BASERUN_CS:sb_era_i + TEAM_BASERUN_CS:sb_era_lb +
##     TEAM_PITCHING_H:TEAM_PITCHING_HR + TEAM_PITCHING_H:TEAM_PITCHING_SO +
##     TEAM_PITCHING_H:TEAM_FIELDING_E + TEAM_PITCHING_H:TEAM_FIELDING_DP +
##     TEAM_PITCHING_H:X2B + TEAM_PITCHING_H:sb_era_m + TEAM_PITCHING_HR:TEAM_PITCHING_BB +
##     TEAM_PITCHING_HR:TEAM_PITCHING_SO + TEAM_PITCHING_HR:TEAM_FIELDING_E +
##     TEAM_PITCHING_HR:TEAM_FIELDING_DP + TEAM_PITCHING_HR:X2B +
##     TEAM_PITCHING_HR:hr_era_i + TEAM_PITCHING_HR:sb_era_fa +
##     TEAM_PITCHING_BB:X3B + TEAM_PITCHING_BB:hr_era_lb + TEAM_PITCHING_BB:hr_era_db +
##     TEAM_PITCHING_BB:sb_era_fa + TEAM_PITCHING_BB:sb_era_i +
##     TEAM_PITCHING_SO:TEAM_FIELDING_E + TEAM_PITCHING_SO:TEAM_FIELDING_DP +
##     TEAM_PITCHING_SO:X2B + TEAM_PITCHING_SO:X3B + TEAM_PITCHING_SO:hr_era_i +
##     TEAM_PITCHING_SO:sb_era_m + TEAM_PITCHING_SO:sb_era_fa +
##     TEAM_PITCHING_SO:sb_era_i + TEAM_PITCHING_SO:sb_era_lb +
##     TEAM_FIELDING_E:TEAM_FIELDING_DP + TEAM_FIELDING_E:X2B +
##     TEAM_FIELDING_E:X3B + TEAM_FIELDING_E:hr_era_i + TEAM_FIELDING_E:hr_era_lb +
##     TEAM_FIELDING_E:sb_era_m + TEAM_FIELDING_E:sb_era_fa + TEAM_FIELDING_E:sb_era_i +
##     TEAM_FIELDING_E:sb_era_lb + TEAM_FIELDING_DP:X2B + TEAM_FIELDING_DP:hr_era_i +
##     TEAM_FIELDING_DP:hr_era_db + TEAM_FIELDING_DP:sb_era_lb +
##     X2B:X3B + X2B:hr_era_lb + X2B:hr_era_db + X2B:sb_era_m +
##     X2B:sb_era_e + X2B:sb_era_i + X2B:sb_era_lb + X3B:hr_era_i +
##     X3B:hr_era_db + X3B:sb_era_m + X3B:sb_era_e + X3B:sb_era_i,
##     data = train_set)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -57.020  -7.507   0.476   7.257  50.309
##
## Coefficients: (1 not defined because of singularities)
##                                     Estimate Std. Error t value Pr(>|t|)
```

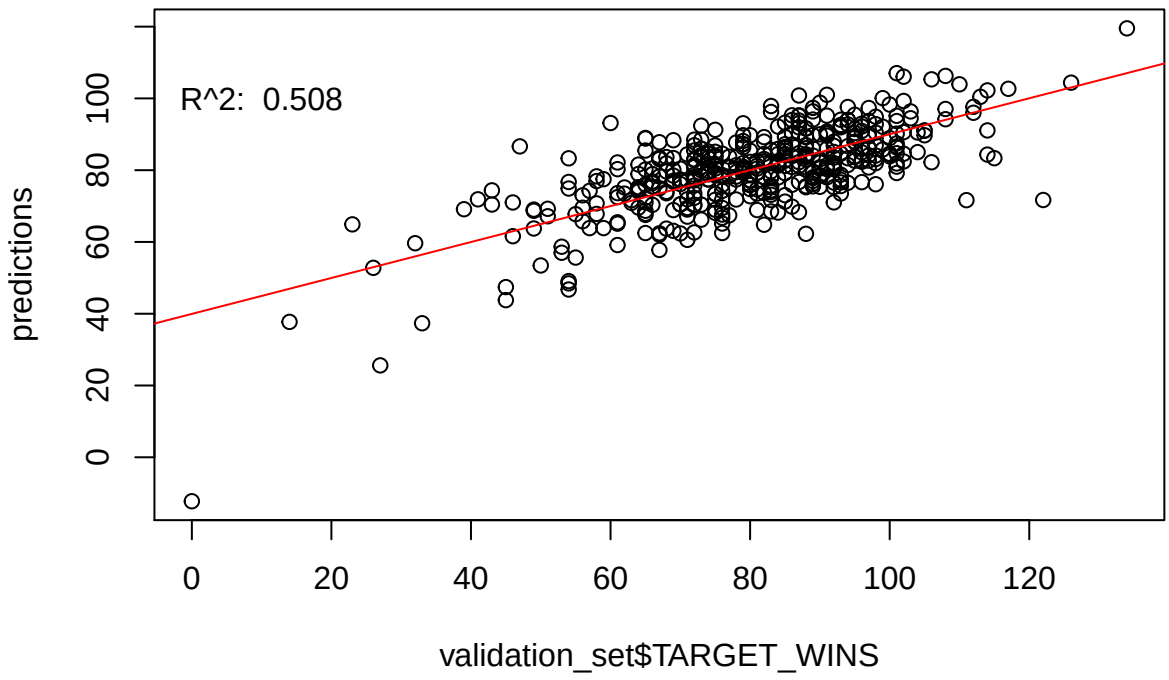
```
## (Intercept)                          1.006e+01  3.763e+01   0.267 0.789251
## TEAM_BATTING_H                      -1.242e-02  2.554e-02  -0.486 0.626812
## TEAM_BASERUN_SB                      4.335e-01  6.188e-02   7.005 3.54e-12 ***
## TEAM_BASERUN_CS                      6.391e-01  1.955e-01   3.270 0.001097 **
## TEAM_PITCHING_H                     -1.349e-01  2.071e-02  -6.515 9.55e-11 ***
## TEAM_PITCHING_HR                     3.574e-01  8.249e-02   4.333 1.56e-05 ***
## TEAM_PITCHING_BB                     1.108e-01  2.685e-02   4.125 3.88e-05 ***
## TEAM_PITCHING_SO                    -1.887e-02  2.161e-02  -0.873 0.382695
## TEAM_FIELDING_E                     -1.609e-02  6.129e-02  -0.263 0.792960
## TEAM_FIELDING_DP                    -8.760e-02  1.353e-01  -0.647 0.517497
## X2B                                  1.498e+02  1.832e+01   8.177 5.59e-16 ***
## X3B                                  4.234e+01  3.254e+01   1.301 0.193470
## hr_era_i                            -1.272e-01  1.148e-01  -1.108 0.267874
## hr_era_lb                           -1.359e+00  2.511e-01  -5.413 7.07e-08 ***
## hr_era_db                           -1.105e-01  1.179e-01  -0.937 0.348687
## sb_era_m                            -8.162e-01  1.517e-01  -5.381 8.45e-08 ***
## sb_era_fa                           -1.104e+00  1.758e-01  -6.281 4.26e-10 ***
## sb_era_e                            -3.099e-01  1.501e-01  -2.065 0.039120 *
## sb_era_i                            -8.414e-02  1.510e-01  -0.557 0.577386
## sb_era_lb                           -2.495e-01  1.103e-01  -2.262 0.023798 *
## sb_era_db                                   NA         NA      NA       NA
## TEAM_BATTING_H:TEAM_PITCHING_H       7.687e-05  1.525e-05   5.040 5.13e-07 ***
## TEAM_BATTING_H:TEAM_PITCHING_BB     -7.668e-05  1.970e-05  -3.893 0.000103 ***
## TEAM_BATTING_H:TEAM_PITCHING_SO      2.239e-05  1.166e-05   1.921 0.054952 .
## TEAM_BATTING_H:TEAM_FIELDING_E       6.556e-05  2.683e-05   2.443 0.014656 *
## TEAM_BATTING_H:X2B                  -3.729e-02  9.626e-03  -3.874 0.000111 ***
## TEAM_BATTING_H:X3B                  -6.372e-02  1.521e-02  -4.188 2.96e-05 ***
## TEAM_BATTING_H:hr_era_lb             9.840e-04  2.202e-04   4.468 8.42e-06 ***
## TEAM_BATTING_H:sb_era_fa             4.775e-04  1.045e-04   4.568 5.27e-06 ***
## TEAM_BATTING_H:sb_era_e              2.823e-04  1.255e-04   2.250 0.024605 *
## TEAM_BATTING_H:sb_era_i              2.066e-04  8.975e-05   2.302 0.021481 *
## TEAM_BATTING_H:sb_era_lb            -1.441e-04  6.348e-05  -2.269 0.023387 *
## TEAM_BASERUN_SB:TEAM_BASERUN_CS     -3.246e-03  8.399e-04  -3.865 0.000115 ***
## TEAM_BASERUN_SB:TEAM_PITCHING_H     -7.078e-05  2.725e-05  -2.597 0.009472 **
## TEAM_BASERUN_SB:TEAM_PITCHING_HR    -3.351e-04  1.515e-04  -2.213 0.027060 *
## TEAM_BASERUN_SB:TEAM_FIELDING_E     -3.639e-04  5.750e-05  -6.329 3.15e-10 ***
## TEAM_BASERUN_SB:X3B                 -1.915e-01  4.005e-02  -4.780 1.90e-06 ***
## TEAM_BASERUN_SB:hr_era_db            7.046e-04  2.909e-04   2.422 0.015537 *
## TEAM_BASERUN_SB:sb_era_m             4.834e-04  3.174e-04   1.523 0.127858
## TEAM_BASERUN_SB:sb_era_lb            3.555e-04  7.463e-05   4.764 2.06e-06 ***
## TEAM_BASERUN_SB:sb_era_db            3.595e-04  6.918e-05   5.197 2.27e-07 ***
## TEAM_BASERUN_CS:TEAM_FIELDING_E     -2.126e-03  7.834e-04  -2.714 0.006721 **
## TEAM_BASERUN_CS:X2B                 -2.954e-01  1.065e-01  -2.774 0.005599 **
## TEAM_BASERUN_CS:X3B                  8.972e-01  2.599e-01   3.453 0.000568 ***
## TEAM_BASERUN_CS:hr_era_lb           -4.943e-03  1.641e-03  -3.012 0.002630 **
## TEAM_BASERUN_CS:sb_era_m             2.713e-03  1.083e-03   2.504 0.012369 *
## TEAM_BASERUN_CS:sb_era_fa            2.452e-03  9.168e-04   2.674 0.007560 **
## TEAM_BASERUN_CS:sb_era_e             1.514e-03  8.490e-04   1.784 0.074629 .
## TEAM_BASERUN_CS:sb_era_i             2.518e-03  8.350e-04   3.016 0.002600 **
## TEAM_BASERUN_CS:sb_era_lb            3.453e-03  1.446e-03   2.388 0.017038 *
## TEAM_PITCHING_H:TEAM_PITCHING_HR    -1.802e-04  3.602e-05  -5.004 6.20e-07 ***
## TEAM_PITCHING_H:TEAM_PITCHING_SO     2.418e-05  7.775e-06   3.110 0.001903 **
## TEAM_PITCHING_H:TEAM_FIELDING_E      2.278e-05  1.029e-05   2.214 0.026977 *
## TEAM_PITCHING_H:TEAM_FIELDING_DP     2.340e-04  6.879e-05   3.402 0.000684 ***
```

```
## TEAM_PITCHING_H:X2B               -1.267e-02  7.674e-03  -1.651 0.098884 .
## TEAM_PITCHING_H:sb_era_m           1.151e-04  6.136e-05   1.876 0.060823 .
## TEAM_PITCHING_HR:TEAM_PITCHING_BB  2.991e-04  5.038e-05   5.936 3.52e-09 ***
## TEAM_PITCHING_HR:TEAM_PITCHING_SO -7.586e-05  3.262e-05  -2.325 0.020162 *
## TEAM_PITCHING_HR:TEAM_FIELDING_E  -6.787e-04  1.196e-04  -5.676 1.62e-08 ***
## TEAM_PITCHING_HR:TEAM_FIELDING_DP  1.143e-03  3.494e-04   3.273 0.001087 **
## TEAM_PITCHING_HR:X2B              -8.149e-02  3.167e-02  -2.573 0.010164 *
## TEAM_PITCHING_HR:hr_era_i         -5.509e-04  2.244e-04  -2.455 0.014181 *
## TEAM_PITCHING_HR:sb_era_fa        -5.302e-04  2.366e-04  -2.241 0.025161 *
## TEAM_PITCHING_BB:X3B              -3.734e-02  1.767e-02  -2.114 0.034685 *
## TEAM_PITCHING_BB:hr_era_lb         7.127e-04  1.448e-04   4.922 9.38e-07 ***
## TEAM_PITCHING_BB:hr_era_db         2.383e-04  1.213e-04   1.965 0.049608 *
## TEAM_PITCHING_BB:sb_era_fa         2.120e-04  8.633e-05   2.455 0.014171 *
## TEAM_PITCHING_BB:sb_era_i         -2.036e-04  8.643e-05  -2.355 0.018616 *
## TEAM_PITCHING_SO:TEAM_FIELDING_E   4.966e-05  1.570e-05   3.163 0.001591 **
## TEAM_PITCHING_SO:TEAM_FIELDING_DP -1.417e-04  7.494e-05  -1.891 0.058765 .
## TEAM_PITCHING_SO:X2B              -2.073e-02  7.193e-03  -2.881 0.004009 **
## TEAM_PITCHING_SO:X3B              -5.812e-02  1.195e-02  -4.863 1.26e-06 ***
## TEAM_PITCHING_SO:hr_era_i          1.580e-04  8.352e-05   1.892 0.058721 .
## TEAM_PITCHING_SO:sb_era_m         -2.081e-04  8.394e-05  -2.479 0.013282 *
## TEAM_PITCHING_SO:sb_era_fa         1.263e-04  5.341e-05   2.364 0.018187 *
## TEAM_PITCHING_SO:sb_era_i         -2.188e-04  6.831e-05  -3.202 0.001387 **
## TEAM_PITCHING_SO:sb_era_lb         1.758e-04  4.357e-05   4.034 5.72e-05 ***
## TEAM_FIELDING_E:TEAM_FIELDING_DP  -4.693e-04  1.526e-04  -3.075 0.002135 **
## TEAM_FIELDING_E:X2B               -2.666e-02  1.339e-02  -1.991 0.046611 *
## TEAM_FIELDING_E:X3B                2.006e-01  2.641e-02   7.596 5.00e-14 ***
## TEAM_FIELDING_E:hr_era_i          -7.326e-04  3.016e-04  -2.429 0.015256 *
## TEAM_FIELDING_E:hr_era_lb          7.370e-04  2.558e-04   2.881 0.004018 **
## TEAM_FIELDING_E:sb_era_m           1.727e-03  2.632e-04   6.563 6.98e-11 ***
## TEAM_FIELDING_E:sb_era_fa          1.122e-03  2.643e-04   4.245 2.30e-05 ***
## TEAM_FIELDING_E:sb_era_i           6.241e-04  1.009e-04   6.186 7.68e-10 ***
## TEAM_FIELDING_E:sb_era_lb         -1.248e-04  6.648e-05  -1.877 0.060641 .
## TEAM_FIELDING_DP:X2B              -2.332e-01  5.811e-02  -4.012 6.27e-05 ***
## TEAM_FIELDING_DP:hr_era_i          5.708e-04  3.866e-04   1.477 0.139965
## TEAM_FIELDING_DP:hr_era_db         9.245e-04  5.157e-04   1.793 0.073170 .
## TEAM_FIELDING_DP:sb_era_lb         9.217e-04  2.668e-04   3.454 0.000565 ***
## X2B:X3B                            3.042e+01  9.709e+00   3.134 0.001755 **
## X2B:hr_era_lb                     -2.560e-01  1.010e-01  -2.536 0.011311 *
## X2B:hr_era_db                     -1.995e-01  5.303e-02  -3.762 0.000174 ***
## X2B:sb_era_m                       1.904e-01  6.890e-02   2.763 0.005792 **
## X2B:sb_era_e                      -1.771e-01  6.719e-02  -2.636 0.008475 **
## X2B:sb_era_i                      -1.511e-01  5.565e-02  -2.716 0.006678 **
## X2B:sb_era_lb                      7.813e-02  3.815e-02   2.048 0.040720 *
## X3B:hr_era_i                       4.793e-01  1.402e-01   3.420 0.000642 ***
## X3B:hr_era_db                      3.863e-01  1.097e-01   3.522 0.000440 ***
## X3B:sb_era_m                       6.473e-01  1.702e-01   3.804 0.000147 ***
## X3B:sb_era_e                       5.190e-01  1.456e-01   3.564 0.000376 ***
## X3B:sb_era_i                       2.284e-01  1.630e-01   1.401 0.161457
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 11.68 on 1721 degrees of freedom
## Multiple R-squared:  0.4837, Adjusted R-squared:  0.4537
## F-statistic: 16.12 on 100 and 1721 DF,  p-value: < 2.2e-16
```

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




```
plot(predictions ~ validation_set$TARGET_WINS)
text(10, 100, paste("R^2: ",
    round(summary(l)$r.squared,
        3)))
abline(l$coefficients,
    col = "red")
```



Conclusion

This model has an adjusted R^2 of 0.45, and a higher R^2 of 0.508 on the validation set, which is suprising. The residuals seem okay, although there is perhaps somthing a slight curve to them. The QQ plot reveals heavy tails, although most deviation is among a few outliers. There's one leverage point with a Cook's distance score over 0.5. Overall, this model could be used for decent predictions (and seems valid), but could probably have it's coefficients narrowed down further. Stepwise selection has a tendency for overfitting, but this model seems to do better on the validation set, which is strange.

Predictions

Below, we make predictions using the the model above (after a little hidden data prep). It then saves them locally.

```
predictions <- as.data.frame(predict(era_model_stepped,
    method3_test))
```

```
## Warning in predict.lm(era_model_stepped, method3_test): prediction from a rank-
## deficient fit may be misleading
```

```
write.csv(predictions,
    "group1_predictions.csv")
print("Saved")
```

```
## [1] "Saved"
```