DATA 621 Homework 1 - Spring2020

Abdelmalek Hajjam / Monu Chacko

Purpose

This report covers an attempt to build a model to predict number of wins of a baseball team in a season based on several offensive and deffensive statistics. Resulting model explained about 36% of variability in the target variable and included most of the provided explanatory variables. Some potentially helpful variables were not included in the data set. For instance, number of At Bats can be used to calculate on-base percentage which may correlate strongly with winning percentage. The model can be revised with additional variables or further analysis.

Data Exploration

The data set describes baseball team statistics for the years 1871 to 2006 inclusive. Each record in the data set represents the performance of the team for the given year adjusted to the current length of the season - 162 games. The data set includes 16 variables (excluding the index) and the training set includes 2,276 records.

Dimensions For reproducibility purposes, we have put the original data sets in git-hub account and then we will read them from there. Let's load our data and look at the the dimensions of our data set.

```
#BTraining <- read.csv("moneyball-training-data.csv")
BTraining <- read.csv("https://raw.githubusercontent.com/theoracley/Data621/master/Homework1/moneyball-
dim(BTraining)
```

```
## [1] 2276   17
```

As we can notice, the training data set has a total of 17 different variables. The total number or records available are 2276.

Structure The below structure is currently present in the data, for simplicity purposes, we have loaded and treated this data set as a data frame in which all the variables are integers.

```
str(BTraining)
```

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

Summary Let's look at the summary of our data.

```
BTraining.summary <- data.frame(unclass(summary(BTraining[2:17])),
check.names = FALSE,
row.names = NULL,
stringsAsFactors = FALSE)
BTraining.summary
```

```
##        TARGET_WINS TEAM_BATTING_H TEAM_BATTING_2B  TEAM_BATTING_3B
## 1 Min.   :  0.00   Min.   : 891   Min.   : 69.0   Min.   :  0.00
## 2 1st Qu.: 71.00   1st Qu.:1383   1st Qu.:208.0   1st Qu.: 34.00
## 3 Median : 82.00   Median :1454   Median :238.0   Median : 47.00
## 4 Mean   : 80.79   Mean   :1469   Mean   :241.2   Mean   : 55.25
## 5 3rd Qu.: 92.00   3rd Qu.:1537   3rd Qu.:273.0   3rd Qu.: 72.00
## 6 Max.   :146.00   Max.   :2554   Max.   :458.0   Max.   :223.00
## 7             <NA>           <NA>            <NA>             <NA>
##    TEAM_BATTING_HR TEAM_BATTING_BB  TEAM_BATTING_SO TEAM_BASERUN_SB
## 1 Min.   :  0.00   Min.   :  0.0   Min.   :   0.0   Min.   :  0.0
## 2 1st Qu.: 42.00   1st Qu.:451.0   1st Qu.: 548.0   1st Qu.: 66.0
## 3 Median :102.00   Median :512.0   Median : 750.0   Median :101.0
## 4 Mean   : 99.61   Mean   :501.6   Mean   : 735.6   Mean   :124.8
## 5 3rd Qu.:147.00   3rd Qu.:580.0   3rd Qu.: 930.0   3rd Qu.:156.0
## 6 Max.   :264.00   Max.   :878.0   Max.   :1399.0   Max.   :697.0
## 7             <NA>            <NA>    NA's   :102     NA's   :131
##   TEAM_BASERUN_CS TEAM_BATTING_HBP TEAM_PITCHING_H TEAM_PITCHING_HR
## 1 Min.   :  0.0    Min.   :29.00   Min.   : 1137    Min.   :  0.0
## 2 1st Qu.: 38.0    1st Qu.:50.50   1st Qu.: 1419    1st Qu.: 50.0
## 3 Median : 49.0    Median :58.00   Median : 1518    Median :107.0
## 4 Mean   : 52.8    Mean   :59.36   Mean   : 1779    Mean   :105.7
## 5 3rd Qu.: 62.0    3rd Qu.:67.00   3rd Qu.: 1682    3rd Qu.:150.0
## 6 Max.   :201.0    Max.   :95.00   Max.   :30132    Max.   :343.0
## 7   NA's   :772     NA's   :2085             <NA>             <NA>
##   TEAM_PITCHING_BB  TEAM_PITCHING_SO  TEAM_FIELDING_E TEAM_FIELDING_DP
## 1 Min.   :   0.0   Min.   :    0.0   Min.   :  65.0    Min.   : 52.0
## 2 1st Qu.: 476.0   1st Qu.:  615.0   1st Qu.: 127.0    1st Qu.:131.0
## 3 Median : 536.5   Median :  813.5   Median : 159.0    Median :149.0
## 4 Mean   : 553.0   Mean   :  817.7   Mean   : 246.5    Mean   :146.4
## 5 3rd Qu.: 611.0   3rd Qu.:  968.0   3rd Qu.: 249.2    3rd Qu.:164.0
## 6 Max.   :3645.0   Max.   :19278.0   Max.   :1898.0    Max.   :228.0
## 7             <NA>     NA's   :102               <NA>    NA's   :286
```

From the statistics above, we can see that there are 6 variables that contain some **NA**'s values, and we need to deal with them later.

```
BTraining.summary$TEAM_BATTING_SO[7]
```

```
## [1] "NA's   :102  "
```

```
BTraining.summary$TEAM_BASERUN_SB[7]
```

```
## [1] "NA's   :131  "
```

```
BTraining.summary$TEAM_BASERUN_CS[7]
```

```
## [1] "NA's   :772  "
```

```
BTraining.summary$TEAM_BATTING_HBP[7]
```

```
## [1] "NA's   :2085  "
```

```
BTraining.summary$TEAM_PITCHING_SO[7]
```

```
## [1] "NA's   :102  "
```

```
BTraining.summary$TEAM_FIELDING_DP[7]
```

```
## [1] "NA's   :286  "
```

Each variable is presented below with corresponding basic statistics (minimum, median and maximum values, mean and standard deviation, number of records with missing values and zero values), boxplot, density plot with highlighted mean value, and scatterplot against outcome variable (`TARGET_WINS`) with best fit line. This information is used to check general validity of data and adjust as necessary.

Let's create a statistical container that will hold this information for each variable

```
sumBTraining = data.frame(Variable = character(),
                  Min = integer(),
                  Median = integer(),
                  Mean = double(),
                  SD = double(),
                  Max = integer(),
                  Num_NAs = integer(),
                  Num_Zeros = integer())
for (i in 2:17) {
  sumBTraining <- rbind(sumBTraining, data.frame(Variable = colnames(BTraining)[i],
                                    Min = min(BTraining[,i], na.rm=TRUE),
                                    Median = median(BTraining[,i], na.rm=TRUE),
                                    Mean = mean(BTraining[,i], na.rm=TRUE),
                                    SD = sd(BTraining[,i], na.rm=TRUE),
                                    Max = max(BTraining[,i], na.rm=TRUE),
```

```
                                    Num_NAs = sum(is.na(BTraining[,i])),
                                    Num_Zeros = length(which(BTraining[,i]==0)))
                  )
}
colnames(sumBTraining) <- c("", "Min", "Median", "Mean", "SD", "Max", "Num of NAs", "Num of Zeros")
```

Let's go through every variable and check on the information output.

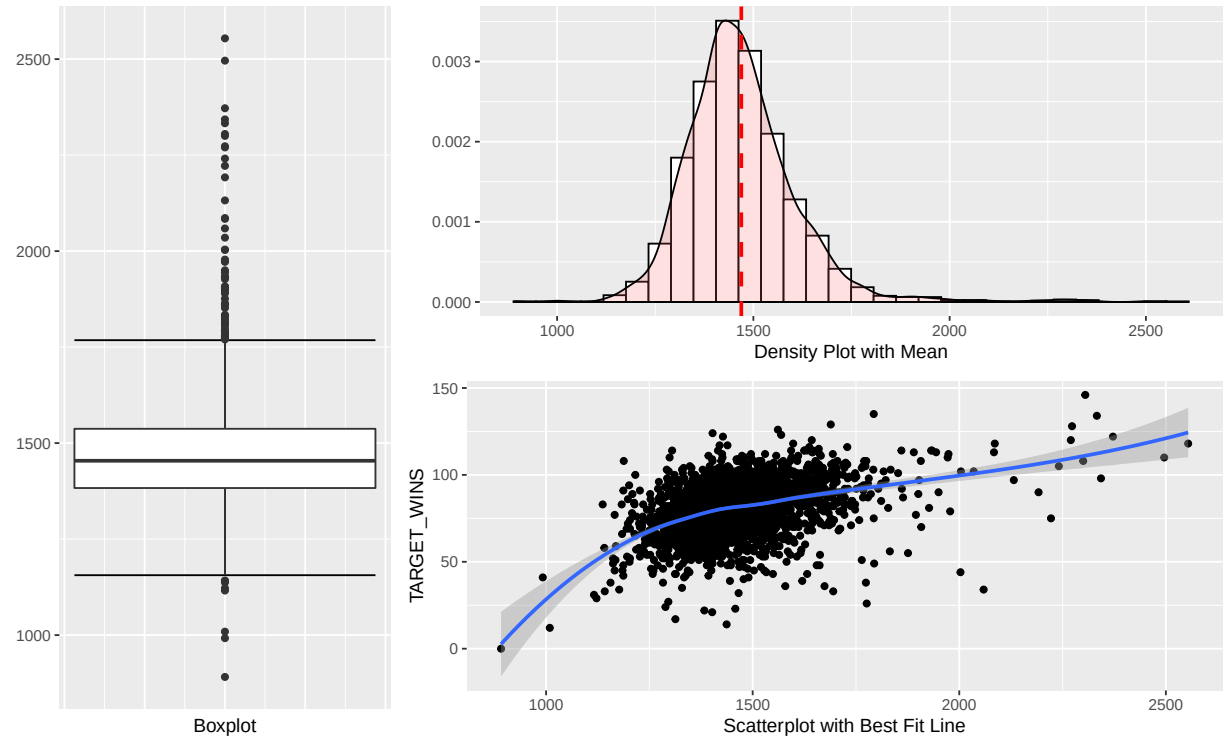
```
kable(sumBTraining[sumBTraining[,1]=="TEAM_BATTING_H",2:8], row.names=FALSE)
```

TEAM_BATTING_H: Number of team base hits (includes singles, doubles, triples and home runs)

Min	Median	Mean	SD	Max	Num of NAs	Num of Zeros
891	1454	1469.27	144.5912	2554	0	0

```
# Boxplot
bp <- ggplot(BTraining, aes(x = 1, y = TEAM_BATTING_H)) +
  stat_boxplot(geom ='errorbar') + geom_boxplot() +
  xlab("Boxplot") + ylab("") + theme(axis.text.x=element_blank(), axis.ticks.x=element_blank())

# Density plot
hp <- ggplot(BTraining, aes(x = TEAM_BATTING_H)) +
  geom_histogram(aes(y=..density..), colour="black", fill="white") +
  geom_density(alpha=.2, fill="#FF6666") + ylab("") + xlab("Density Plot with Mean") +
  geom_vline(aes(xintercept=mean(TEAM_BATTING_H, na.rm=TRUE)), color="red", linetype="dashed", size=1)

# Scatterplot
sp <- ggplot(data=BTraining, aes(x=TEAM_BATTING_H, y=TARGET_WINS)) +
  geom_point() + geom_smooth(method = "loess") +
  xlab("Scatterplot with Best Fit Line")

grid.arrange(bp, hp, sp, layout_matrix=rbind(c(1,2,2),c(1,3,3)))
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



Analysis: There are no missing values. The range and distribution are reasonable.

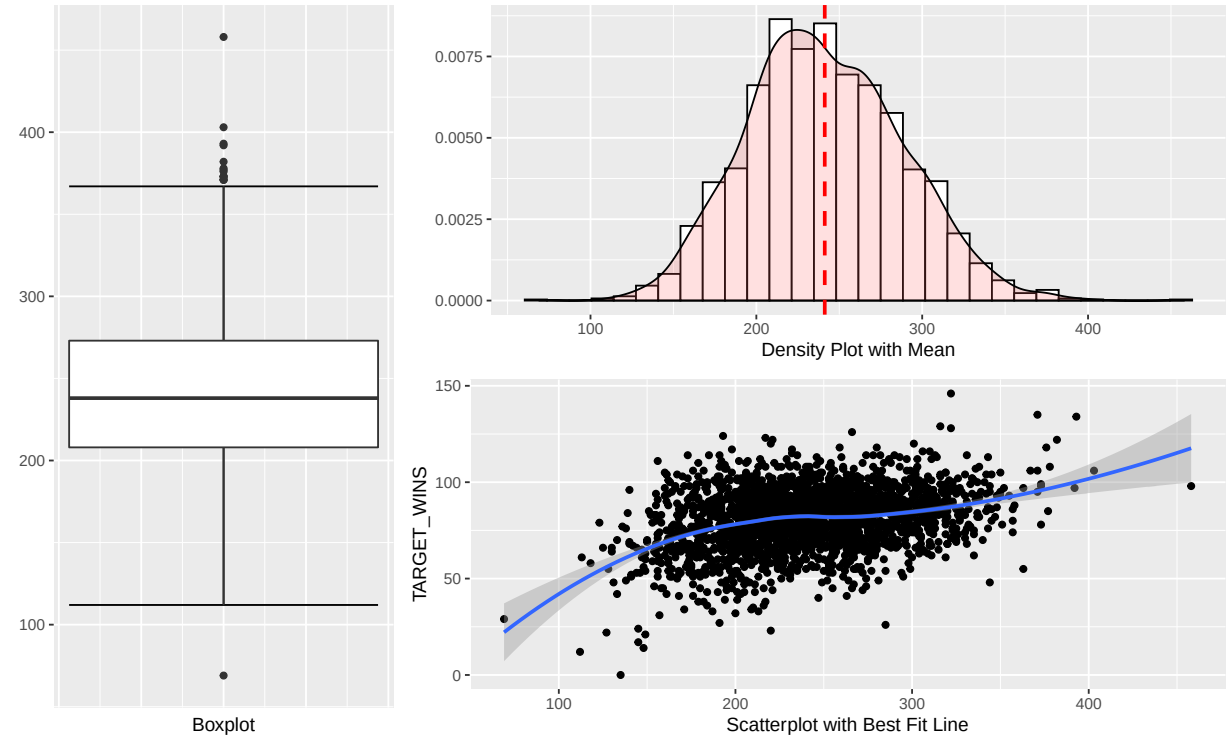
```
kable(sumBTraining[sumBTraining[,1]=="TEAM_BATTING_2B",2:8], row.names=FALSE)
```

TEAM_BATTING_2B: Number of team doubles

Min	Median	Mean	SD	Max	Num of NAs	N
69	238	241.2469	46.80141	458	0	

```
# Boxplot
bp <- ggplot(BTraining, aes(x = 1, y = TEAM_BATTING_2B)) +
  stat_boxplot(geom ='errorbar') + geom_boxplot() +
  xlab("Boxplot") + ylab("") + theme(axis.text.x=element_blank(), axis.ticks.x=element_blank())

# Density plot
hp <- ggplot(BTraining, aes(x = TEAM_BATTING_2B)) +
  geom_histogram(aes(y=..density..), colour="black", fill="white") +
  geom_density(alpha=.2, fill="#FF6666") + ylab("") + xlab("Density Plot with Mean") +
  geom_vline(aes(xintercept=mean(TEAM_BATTING_2B, na.rm=TRUE)), color="red", linetype="dashed", size=1)

# Scatterplot
sp <- ggplot(data=BTraining, aes(x=TEAM_BATTING_2B, y=TARGET_WINS)) +
  geom_point() + geom_smooth(method = "loess") +
  xlab("Scatterplot with Best Fit Line")

grid.arrange(bp, hp, sp, layout_matrix=rbind(c(1,2,2),c(1,3,3)))
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



Analysis: There are no missing values. The range and distribution are reasonable.

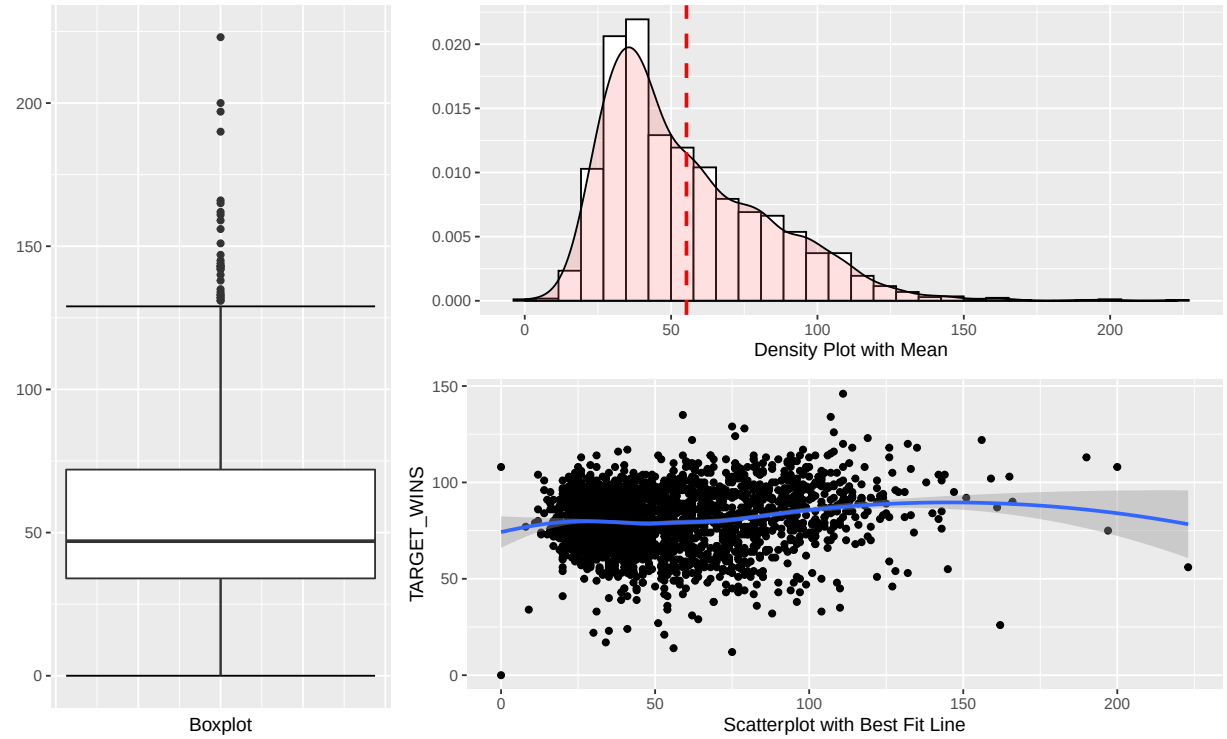
```
kable(sumBTraining[sumBTraining[,1]=="TEAM_BATTING_3B",2:8], row.names=FALSE)
```

TEAM_BATTING_3B: Number of team triples

Min	Median	Mean	SD	Max	Num of NAs	Num
0	47	55.25	27.93856	223	0	

```
# Boxplot
bp <- ggplot(BTraining, aes(x = 1, y = TEAM_BATTING_3B)) +
  stat_boxplot(geom ='errorbar') + geom_boxplot() +
  xlab("Boxplot") + ylab("") + theme(axis.text.x=element_blank(), axis.ticks.x=element_blank())

# Density plot
hp <- ggplot(BTraining, aes(x = TEAM_BATTING_3B)) +
  geom_histogram(aes(y=..density..), colour="black", fill="white") +
  geom_density(alpha=.2, fill="#FF6666") + ylab("") + xlab("Density Plot with Mean") +
  geom_vline(aes(xintercept=mean(TEAM_BATTING_3B, na.rm=TRUE)), color="red", linetype="dashed", size=1)

# Scatterplot
sp <- ggplot(data=BTraining, aes(x=TEAM_BATTING_3B, y=TARGET_WINS)) +
  geom_point() + geom_smooth(method = "loess") +
  xlab("Scatterplot with Best Fit Line")

grid.arrange(bp, hp, sp, layout_matrix=rbind(c(1,2,2),c(1,3,3)))
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



Analysis: The range and distribution are reasonable. There are 2 records with zero values which is unrealistic for a team in a season. One record (index 1347) has 12 variables with missing values, including the

outcome variable. This record will be deleted from the data set. Second record (index 1494) has 7 missing variables, but it does have some recorded values in all categories - batting, pitching and fielding. Zero value for `TEAM_BATTING_3B` can be replaced with the median (because the distribution is right-skewed, median value will provide more realistic estimate).

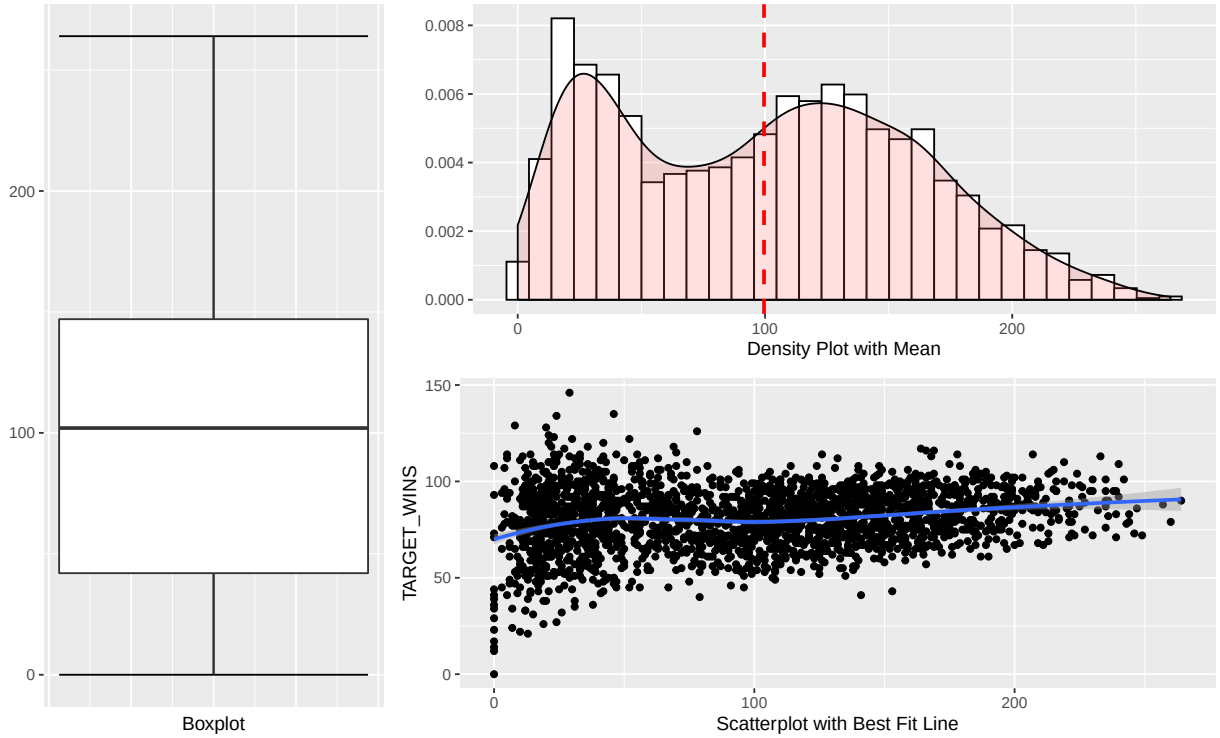
```
kable(sumBTraining[sumBTraining[,1]=="TEAM_BATTING_HR",2:8], row.names=FALSE)
```

TEAM_BATTING_HR: Number of team home runs

Min	Median	Mean	SD	Max	Num of NAs
0	102	99.61204	60.54687	264	0

```
# Boxplot
bp <- ggplot(BTraining, aes(x = 1, y = TEAM_BATTING_HR)) +
  stat_boxplot(geom ='errorbar') + geom_boxplot() +
  xlab("Boxplot") + ylab("") + theme(axis.text.x=element_blank(), axis.ticks.x=element_blank())

# Density plot
hp <- ggplot(BTraining, aes(x = TEAM_BATTING_HR)) +
  geom_histogram(aes(y=..density..), colour="black", fill="white") +
  geom_density(alpha=.2, fill="#FF6666") + ylab("") + xlab("Density Plot with Mean") +
  geom_vline(aes(xintercept=mean(TEAM_BATTING_HR, na.rm=TRUE)), color="red", linetype="dashed", size=1)

# Scatterplot
sp <- ggplot(data=BTraining, aes(x=TEAM_BATTING_HR, y=TARGET_WINS)) +
  geom_point() + geom_smooth(method = "loess") +
  xlab("Scatterplot with Best Fit Line")

grid.arrange(bp, hp, sp, layout_matrix=rbind(c(1,2,2),c(1,3,3)))
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



Analysis: The range is reasonable. The distribution is interesting because it is multimodal. Most likely this indicates major changes in game dynamics - perhaps, some rule adjustments started favoring batters.

Or perhaps, this is an affect of steroid era. There are 15 records with zero values which is unrealistic for this variable. They can be imputed from other values.

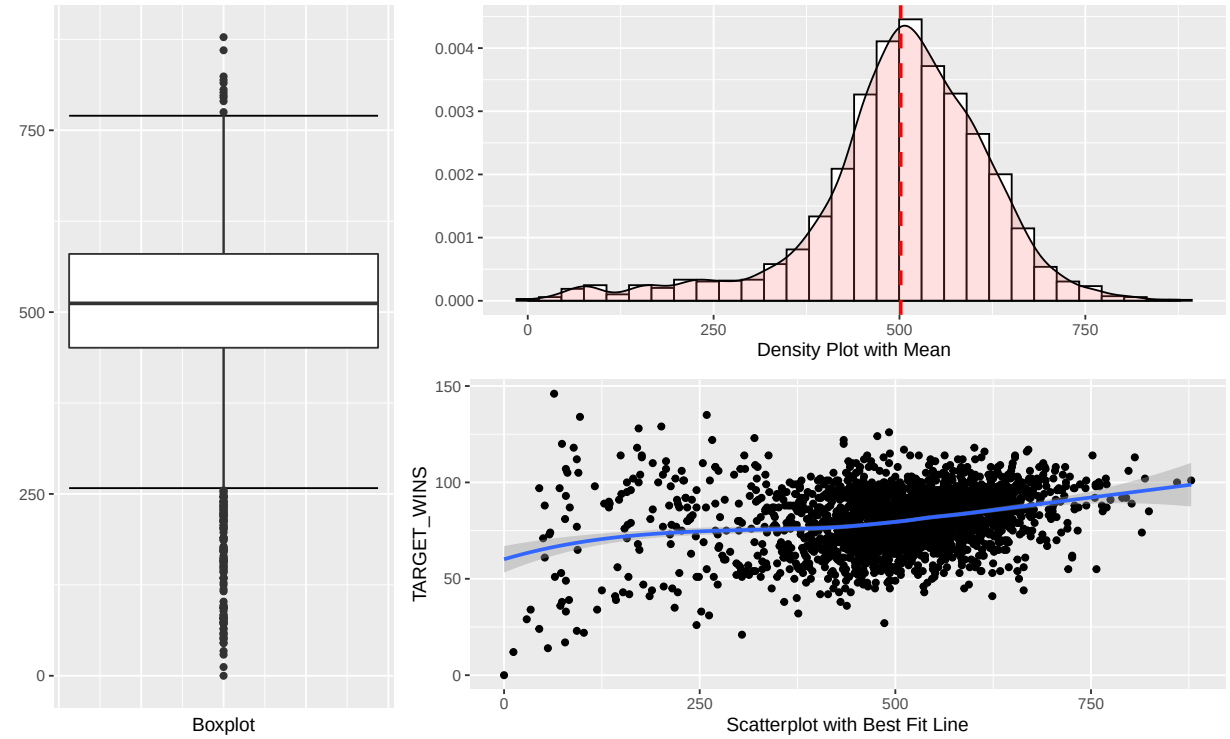
```
kable(sumBTraining[sumBTraining[,1]=="TEAM_BATTING_BB",2:8], row.names=FALSE)
```

TEAM_BATTING_BB: Number of team walks

Min	Median	Mean	SD	Max	Num of NAs	Nu
0	512	501.5589	122.6709	878	0	

```
# Boxplot
bp <- ggplot(BTraining, aes(x = 1, y = TEAM_BATTING_BB)) +
  stat_boxplot(geom ='errorbar') + geom_boxplot() +
  xlab("Boxplot") + ylab("") + theme(axis.text.x=element_blank(), axis.ticks.x=element_blank())

# Density plot
hp <- ggplot(BTraining, aes(x = TEAM_BATTING_BB)) +
  geom_histogram(aes(y=..density..), colour="black", fill="white") +
  geom_density(alpha=.2, fill="#FF6666") + ylab("") + xlab("Density Plot with Mean") +
  geom_vline(aes(xintercept=mean(TEAM_BATTING_BB, na.rm=TRUE)), color="red", linetype="dashed", size=1)

# Scatterplot
sp <- ggplot(data=BTraining, aes(x=TEAM_BATTING_BB, y=TARGET_WINS)) +
  geom_point() + geom_smooth(method = "loess") +
  xlab("Scatterplot with Best Fit Line")

grid.arrange(bp, hp, sp, layout_matrix=rbind(c(1,2,2),c(1,3,3)))
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



Analysis: The range and distribution are reasonable. There is one record (index 1347) that has a zero value. This record was discussed above (under `TEAM_BATTING_3B`) and it will be deleted from the data set.

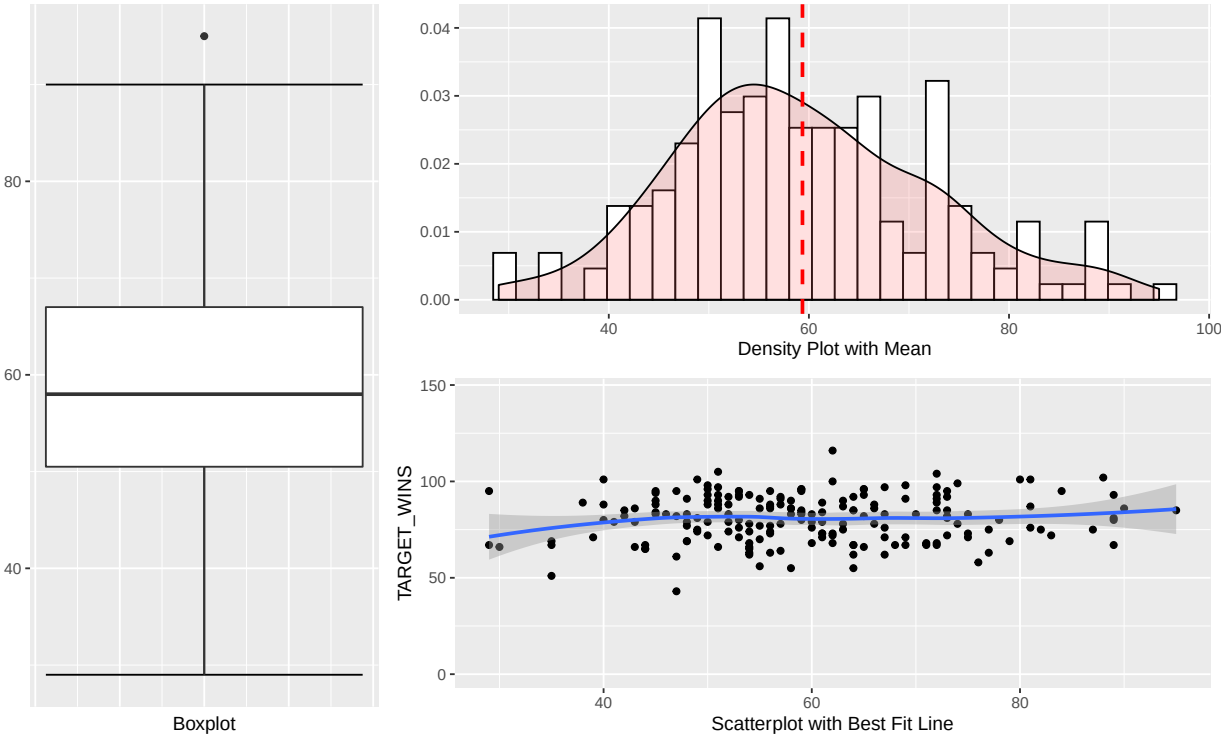
```
kable(sumBTraining[sumBTraining[,1]=="TEAM_BATTING_HBP",2:8], row.names=FALSE)
```

TEAM_BATTING_HBP: Number of team batters hit by pitch

Min	Median	Mean	SD	Max	N
29	58	59.35602	12.96712	95	

```
# Boxplot
bp <- ggplot(BTraining, aes(x = 1, y = TEAM_BATTING_HBP)) +
  stat_boxplot(geom ='errorbar') + geom_boxplot() +
  xlab("Boxplot") + ylab("") + theme(axis.text.x=element_blank(), axis.ticks.x=element_blank())

# Density plot
hp <- ggplot(BTraining, aes(x = TEAM_BATTING_HBP)) +
  geom_histogram(aes(y=..density..), colour="black", fill="white") +
  geom_density(alpha=.2, fill="#FF6666") + ylab("") + xlab("Density Plot with Mean") +
  geom_vline(aes(xintercept=mean(TEAM_BATTING_HBP, na.rm=TRUE)), color="red", linetype="dashed", size=1

# Scatterplot
sp <- ggplot(data=BTraining, aes(x=TEAM_BATTING_HBP, y=TARGET_WINS)) +
  geom_point() + geom_smooth(method = "loess") +
  xlab("Scatterplot with Best Fit Line")

grid.arrange(bp, hp, sp, layout_matrix=rbind(c(1,2,2),c(1,3,3)))
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



Analysis: There are 2,085 records - 91.6% of data set - that are missing value. Because this variable is missing for majority of records, it will not be imputed and will be left out from the regression model.

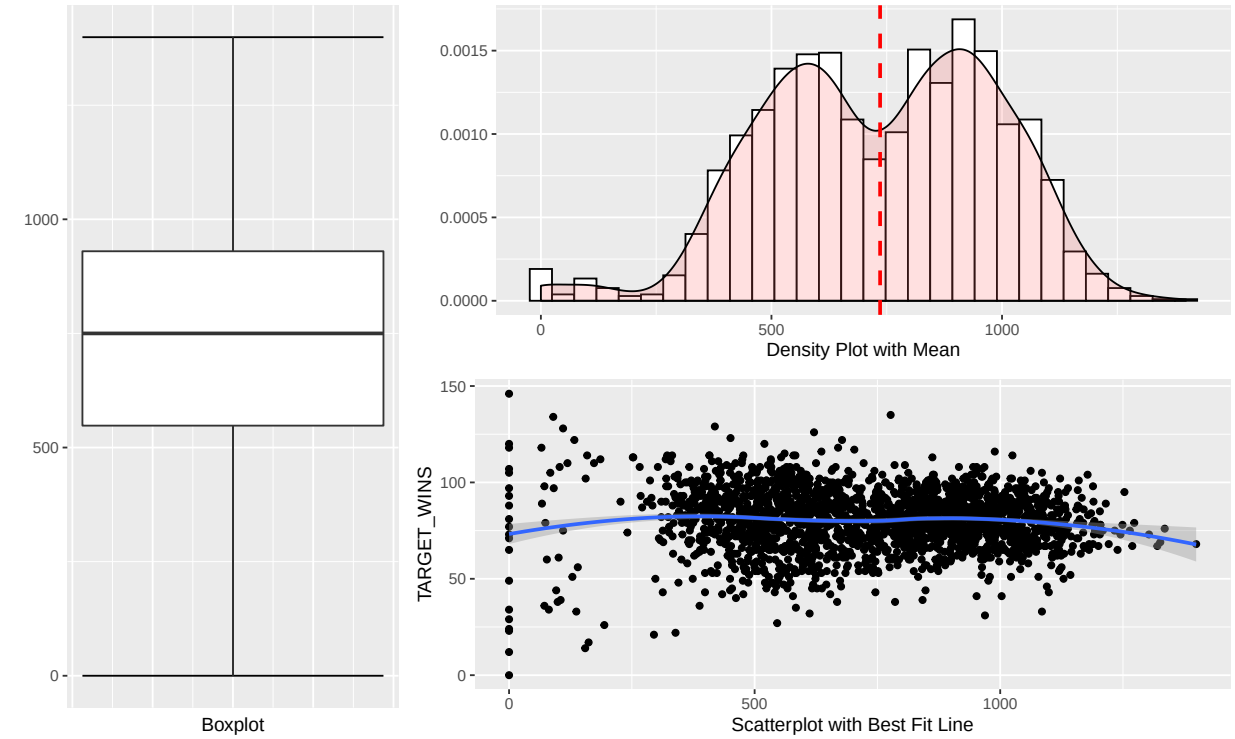
```
kable(sumBTraining[sumBTraining[,1]=="TEAM_BATTING_SO",2:8], row.names=FALSE)
```

TEAM_BATTING_SO: Number of team strikeouts by batters

Min	Median	Mean	SD	Max	N
0	750	735.6053	248.5264	1399	

```
# Boxplot
bp <- ggplot(BTraining, aes(x = 1, y = TEAM_BATTING_SO)) +
  stat_boxplot(geom ='errorbar') + geom_boxplot() +
  xlab("Boxplot") + ylab("") + theme(axis.text.x=element_blank(), axis.ticks.x=element_blank())

# Density plot
hp <- ggplot(BTraining, aes(x = TEAM_BATTING_SO)) +
  geom_histogram(aes(y=..density..), colour="black", fill="white") +
  geom_density(alpha=.2, fill="#FF6666") + ylab("") + xlab("Density Plot with Mean") +
  geom_vline(aes(xintercept=mean(TEAM_BATTING_SO, na.rm=TRUE)), color="red", linetype="dashed", size=1)

# Scatterplot
sp <- ggplot(data=BTraining, aes(x=TEAM_BATTING_SO, y=TARGET_WINS)) +
  geom_point() + geom_smooth(method = "loess") +
  xlab("Scatterplot with Best Fit Line")

grid.arrange(bp, hp, sp, layout_matrix=rbind(c(1,2,2),c(1,3,3)))
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



Analysis: There are 122 records with missing or zero value (as wtih other variables a zero value is unrealistic). These values can be imputed. Similarly to homeruns, the distribution is multimodal, which is

interesting enough for additional analysis. Another area of concern is a noticeable left tail. It is highly unlikely to have games without any strikeouts, so anything lower than 162 (average of 1 strikeout per game) is definitely suspect.

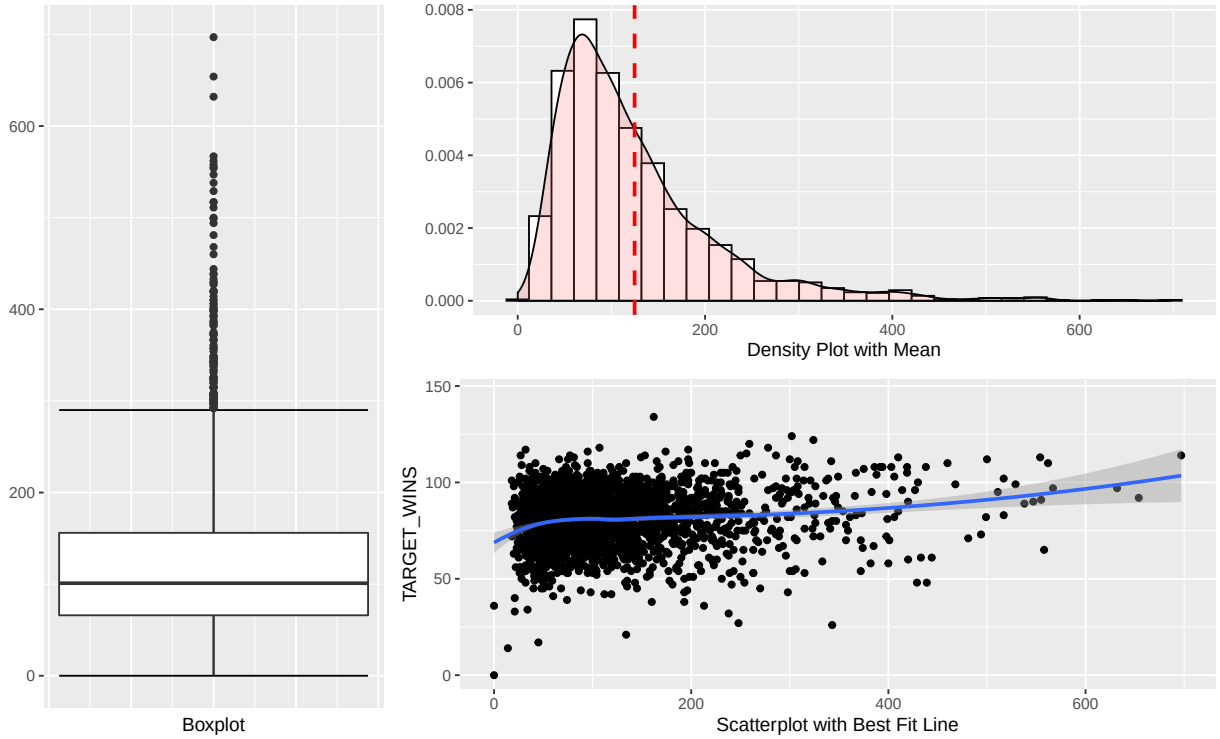
```
kable(sumBTraining[sumBTraining[,1]=="TEAM_BASERUN_SB",2:8], row.names=FALSE)
```

TEAM_BASERUN_SB: Number of team stolen bases

Min	Median	Mean	SD	Max	Num of NA
0	101	124.7618	87.79117	697	13

```
# Boxplot
bp <- ggplot(BTraining, aes(x = 1, y = TEAM_BASERUN_SB)) +
  stat_boxplot(geom ='errorbar') + geom_boxplot() +
  xlab("Boxplot") + ylab("") + theme(axis.text.x=element_blank(), axis.ticks.x=element_blank())

# Density plot
hp <- ggplot(BTraining, aes(x = TEAM_BASERUN_SB)) +
  geom_histogram(aes(y=..density..), colour="black", fill="white") +
  geom_density(alpha=.2, fill="#FF6666") + ylab("") + xlab("Density Plot with Mean") +
  geom_vline(aes(xintercept=mean(TEAM_BASERUN_SB, na.rm=TRUE)), color="red", linetype="dashed", size=1)

# Scatterplot
sp <- ggplot(data=BTraining, aes(x=TEAM_BASERUN_SB, y=TARGET_WINS)) +
  geom_point() + geom_smooth(method = "loess") +
  xlab("Scatterplot with Best Fit Line")

grid.arrange(bp, hp, sp, layout_matrix=rbind(c(1,2,2),c(1,3,3)))
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



Analysis: The range and distribution are reasonable. The only issue are 133 records with missing or zero value. These values can be imputed in order to use these records in model building.

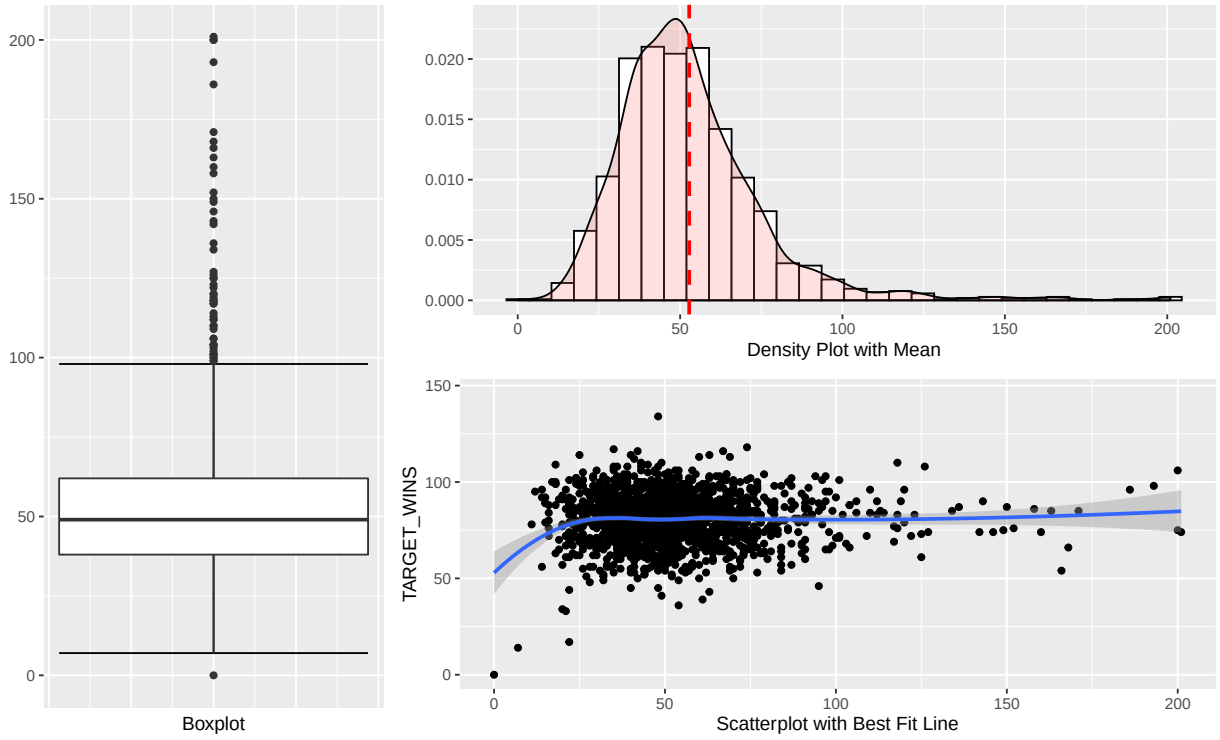
```
kable(sumBTraining[sumBTraining[,1]=="TEAM_BASERUN_CS",2:8], row.names=FALSE)
```

TEAM_BASERUN_CS: Number of team runners caught stealing

Min	Median	Mean	SD	Max
0	49	52.80386	22.95634	201

```
# Boxplot
bp <- ggplot(BTraining, aes(x = 1, y = TEAM_BASERUN_CS)) +
  stat_boxplot(geom ='errorbar') + geom_boxplot() +
  xlab("Boxplot") + ylab("") + theme(axis.text.x=element_blank(), axis.ticks.x=element_blank())

# Density plot
hp <- ggplot(BTraining, aes(x = TEAM_BASERUN_CS)) +
  geom_histogram(aes(y=..density..), colour="black", fill="white") +
  geom_density(alpha=.2, fill="#FF6666") + ylab("") + xlab("Density Plot with Mean") +
  geom_vline(aes(xintercept=mean(TEAM_BASERUN_CS, na.rm=TRUE)), color="red", linetype="dashed", size=1)

# Scatterplot
sp <- ggplot(data=BTraining, aes(x=TEAM_BASERUN_CS, y=TARGET_WINS)) +
  geom_point() + geom_smooth(method = "loess") +
  xlab("Scatterplot with Best Fit Line")

grid.arrange(bp, hp, sp, layout_matrix=rbind(c(1,2,2),c(1,3,3)))
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



Analysis: The range and distribution are reasonable; however, there is significant number of missing values - 773, including one zero value. This represents a third of the entire data set. It may be possible to impute this value, but it may be necessary to leave this variable out of model building.

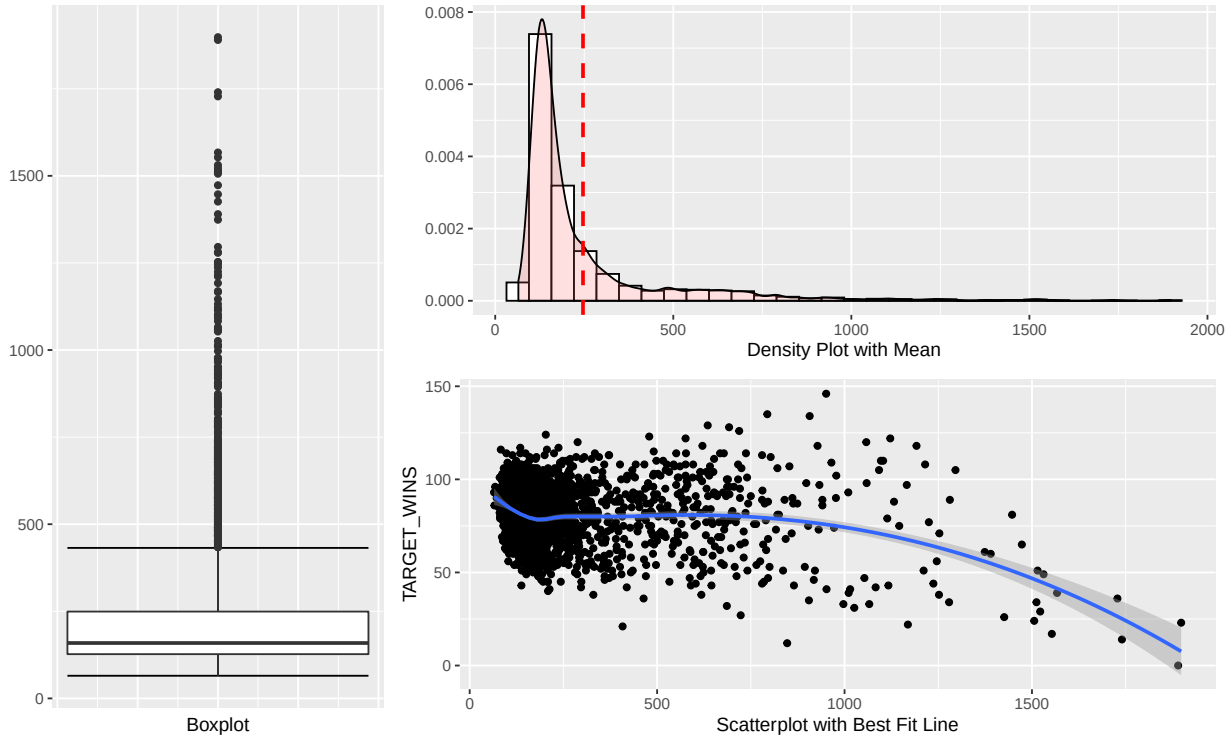
```
kable(sumBTraining[sumBTraining[,1]=="TEAM_FIELDING_E",2:8], row.names=FALSE)
```

TEAM_FIELDING_E: Number of team fielding errors

Min	Median	Mean	SD	Max	Num of NA
65	159	246.4807	227.771	1898	

```
# Boxplot
bp <- ggplot(BTraining, aes(x = 1, y = TEAM_FIELDING_E)) +
  stat_boxplot(geom ='errorbar') + geom_boxplot() +
  xlab("Boxplot") + ylab("") + theme(axis.text.x=element_blank(), axis.ticks.x=element_blank())

# Density plot
hp <- ggplot(BTraining, aes(x = TEAM_FIELDING_E)) +
  geom_histogram(aes(y=..density..), colour="black", fill="white") +
  geom_density(alpha=.2, fill="#FF6666") + ylab("") + xlab("Density Plot with Mean") +
  geom_vline(aes(xintercept=mean(TEAM_FIELDING_E, na.rm=TRUE)), color="red", linetype="dashed", size=1)

# Scatterplot
sp <- ggplot(data=BTraining, aes(x=TEAM_FIELDING_E, y=TARGET_WINS)) +
  geom_point() + geom_smooth(method = "loess") +
  xlab("Scatterplot with Best Fit Line")

grid.arrange(bp, hp, sp, layout_matrix=rbind(c(1,2,2),c(1,3,3)))
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



Analysis: There are no missing values. Distribution has a very long right tail. Values in the 1,000 and above range are highly suspect. One of the highest historical number of errors is 867 errors by Washington

in 1886 for 122 games. That is equal about 1,151 errors for 162 game season. There are multiple values above that number. This may unfavorably influence a model.

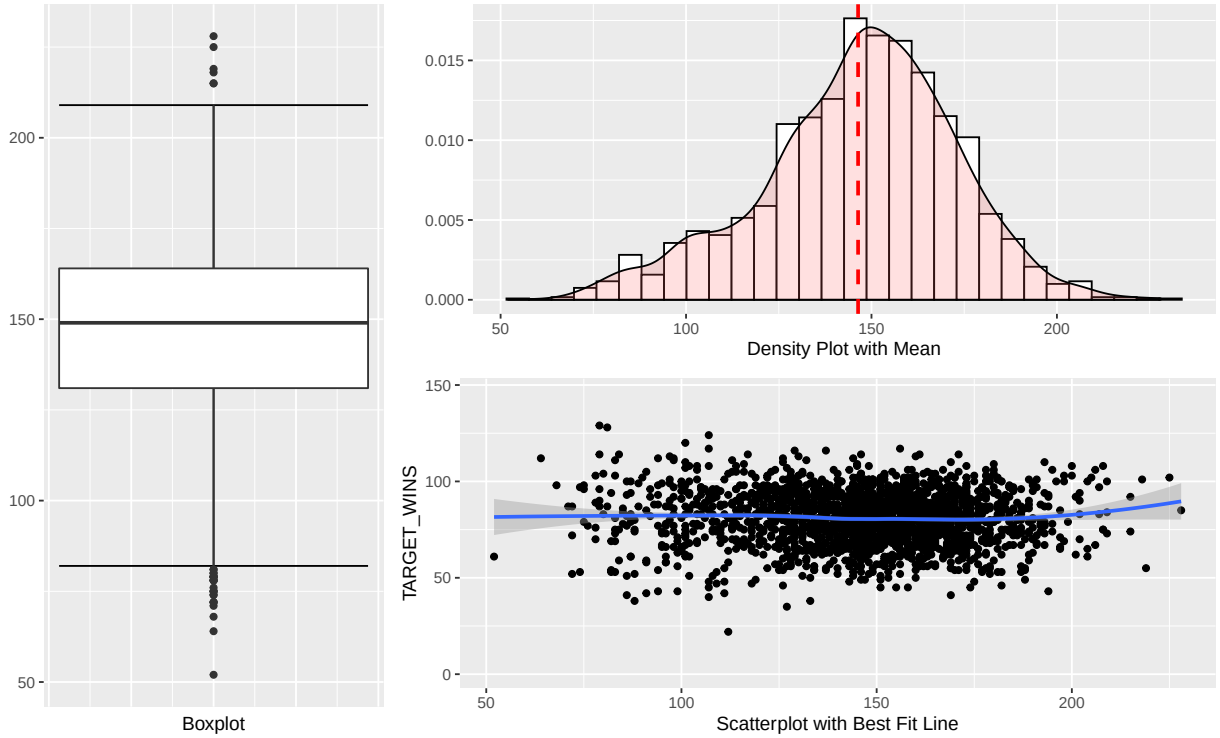
```
kable(sumBTraining[sumBTraining[,1]=="TEAM_FIELDING_DP",2:8], row.names=FALSE)
```

TEAM_FIELDING_DP: Number of team fielding double plays

Min	Median	Mean	SD	Max	N
52	149	146.3879	26.22639	228	

```
# Boxplot
bp <- ggplot(BTraining, aes(x = 1, y = TEAM_FIELDING_DP)) +
  stat_boxplot(geom ='errorbar') + geom_boxplot() +
  xlab("Boxplot") + ylab("") + theme(axis.text.x=element_blank(), axis.ticks.x=element_blank())

# Density plot
hp <- ggplot(BTraining, aes(x = TEAM_FIELDING_DP)) +
  geom_histogram(aes(y=..density..), colour="black", fill="white") +
  geom_density(alpha=.2, fill="#FF6666") + ylab("") + xlab("Density Plot with Mean") +
  geom_vline(aes(xintercept=mean(TEAM_FIELDING_DP, na.rm=TRUE)), color="red", linetype="dashed", size=1

# Scatterplot
sp <- ggplot(data=BTraining, aes(x=TEAM_FIELDING_DP, y=TARGET_WINS)) +
  geom_point() + geom_smooth(method = "loess") +
  xlab("Scatterplot with Best Fit Line")

grid.arrange(bp, hp, sp, layout_matrix=rbind(c(1,2,2),c(1,3,3)))
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



Analysis: The range and distribution are reasonable. Similar to a few other variables there is a medium number off missing values - 286 records. This value can be imputed.

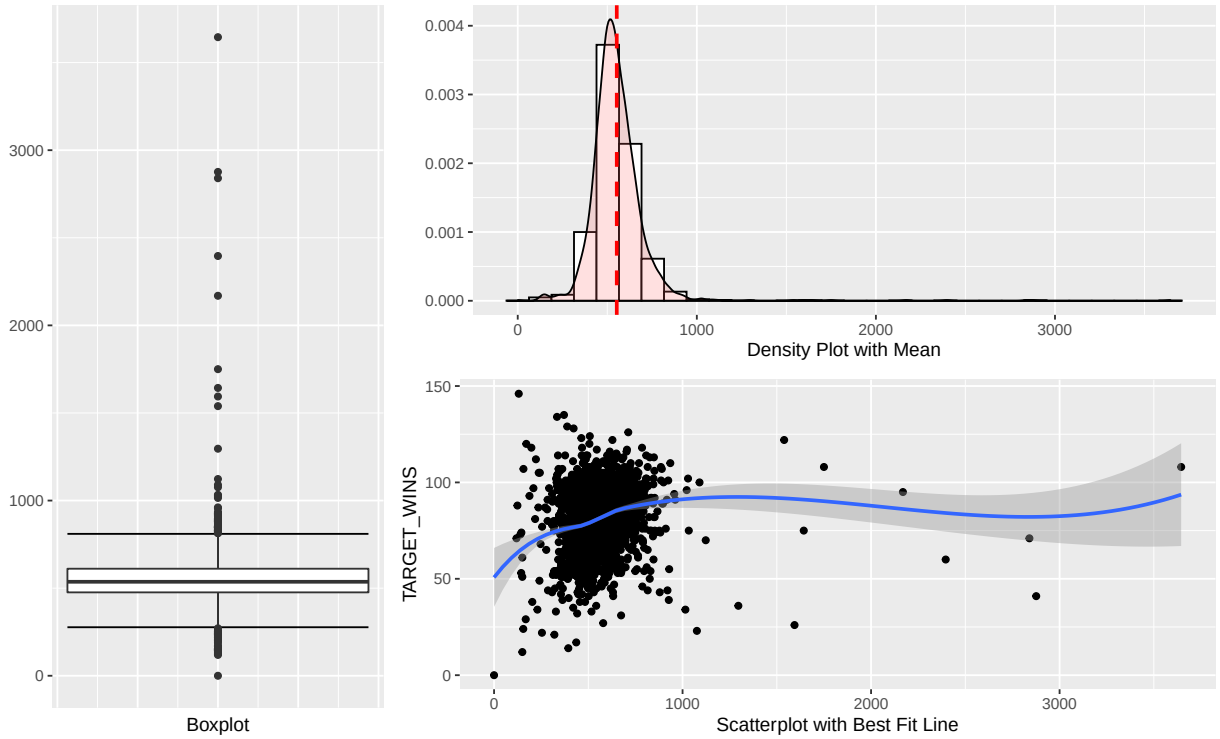
```
kable(sumBTraining[sumBTraining[,1]=="TEAM_PITCHING_BB",2:8], row.names=FALSE)
```

TEAM_PITCHING_BB: Number of walks given up by pitchers

Min	Median	Mean	SD	Max
0	536.5	553.0079	166.3574	3645

```
# Boxplot
bp <- ggplot(BTraining, aes(x = 1, y = TEAM_PITCHING_BB)) +
  stat_boxplot(geom ='errorbar') + geom_boxplot() +
  xlab("Boxplot") + ylab("") + theme(axis.text.x=element_blank(), axis.ticks.x=element_blank())

# Density plot
hp <- ggplot(BTraining, aes(x = TEAM_PITCHING_BB)) +
  geom_histogram(aes(y=..density..), colour="black", fill="white") +
  geom_density(alpha=.2, fill="#FF6666") + ylab("") + xlab("Density Plot with Mean") +
  geom_vline(aes(xintercept=mean(TEAM_PITCHING_BB, na.rm=TRUE)), color="red", linetype="dashed", size=1

# Scatterplot
sp <- ggplot(data=BTraining, aes(x=TEAM_PITCHING_BB, y=TARGET_WINS)) +
  geom_point() + geom_smooth(method = "loess") +
  xlab("Scatterplot with Best Fit Line")

grid.arrange(bp, hp, sp, layout_matrix=rbind(c(1,2,2),c(1,3,3)))
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



Analysis: There are no missing values with the exception of record 1347 which will be deleted from model building. There are some unrealistic outliers. Current record of walks by a team in a season is held by 1949

Boston Red Sox - 835 walks in 155 games. For a 162 game season, this number is 873. This variable will be capped at 1,100 and any value over this will be set to this cap.

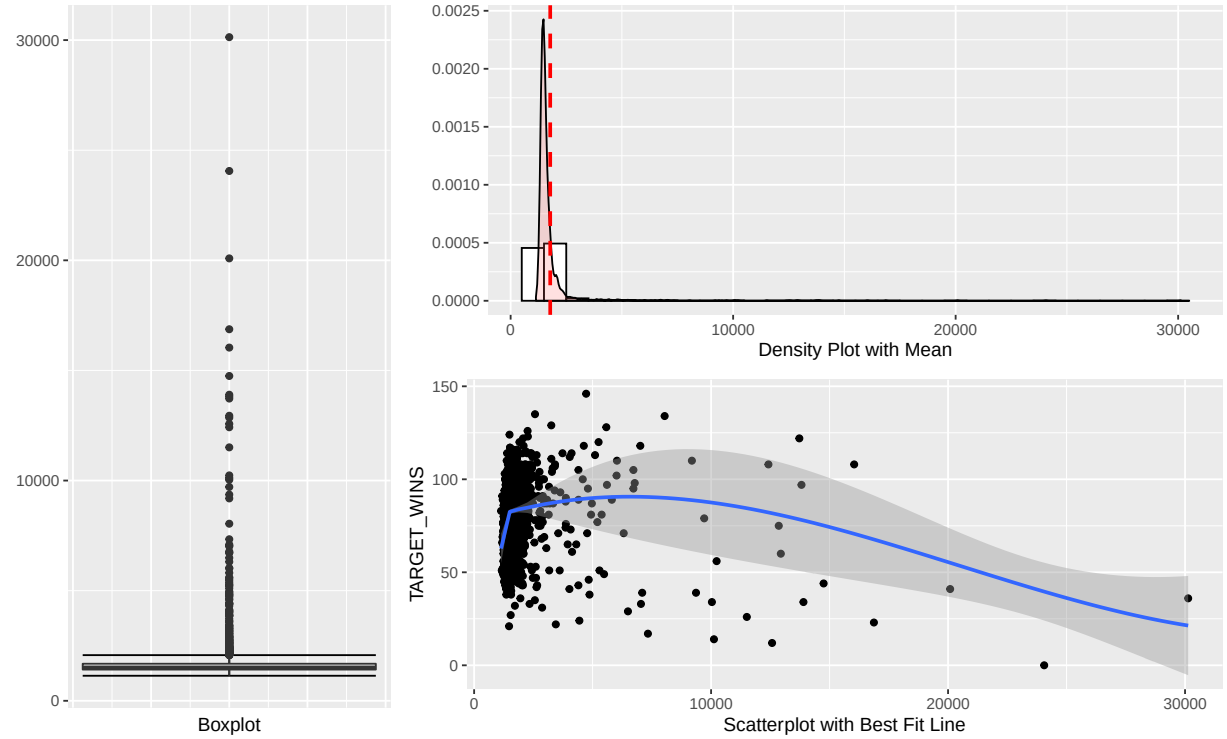
```
kable(sumBTraining[sumBTraining[,1]=="TEAM_PITCHING_H",2:8], row.names=FALSE)
```

TEAM_PITCHING_H: Number of base hits given up by pitchers

Min	Median	Mean	SD	Max
1137	1518	1779.21	1406.843	30132

```
# Boxplot
bp <- ggplot(BTraining, aes(x = 1, y = TEAM_PITCHING_H)) +
  stat_boxplot(geom ='errorbar') + geom_boxplot() +
  xlab("Boxplot") + ylab("") + theme(axis.text.x=element_blank(), axis.ticks.x=element_blank())

# Density plot
hp <- ggplot(BTraining, aes(x = TEAM_PITCHING_H)) +
  geom_histogram(aes(y=..density..), colour="black", fill="white") +
  geom_density(alpha=.2, fill="#FF6666") + ylab("") + xlab("Density Plot with Mean") +
  geom_vline(aes(xintercept=mean(TEAM_PITCHING_H, na.rm=TRUE)), color="red", linetype="dashed", size=1)

# Scatterplot
sp <- ggplot(data=BTraining, aes(x=TEAM_PITCHING_H, y=TARGET_WINS)) +
  geom_point() + geom_smooth(method = "loess") +
  xlab("Scatterplot with Best Fit Line")

grid.arrange(bp, hp, sp, layout_matrix=rbind(c(1,2,2),c(1,3,3)))
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



Analysis: Similar to TEAM_PITCHING_BB above, there are no missing value, but there issues with outliers. Based on visualizations, this variable will be capped at 13,000 and any value over this will be set to this cap.

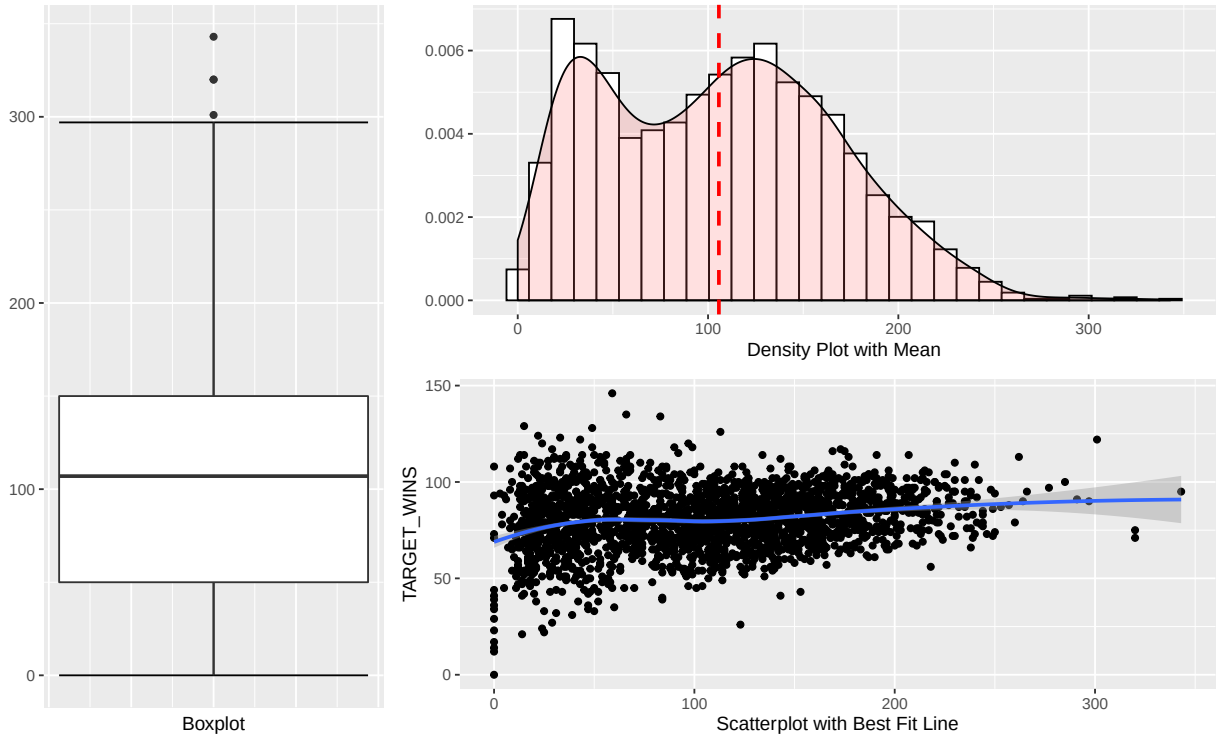
```
kable(sumBTraining[sumBTraining[,1]=="TEAM_PITCHING_HR",2:8], row.names=FALSE)
```

TEAM_PITCHING_HR: Number of home runs given up by pitchers

Min	Median	Mean	SD	M
0	107	105.6986	61.29875	3

```
# Boxplot
bp <- ggplot(BTraining, aes(x = 1, y = TEAM_PITCHING_HR)) +
  stat_boxplot(geom ='errorbar') + geom_boxplot() +
  xlab("Boxplot") + ylab("") + theme(axis.text.x=element_blank(), axis.ticks.x=element_blank())

# Density plot
hp <- ggplot(BTraining, aes(x = TEAM_PITCHING_HR)) +
  geom_histogram(aes(y=..density..), colour="black", fill="white") +
  geom_density(alpha=.2, fill="#FF6666") + ylab("") + xlab("Density Plot with Mean") +
  geom_vline(aes(xintercept=mean(TEAM_PITCHING_HR, na.rm=TRUE)), color="red", linetype="dashed", size=1

# Scatterplot
sp <- ggplot(data=BTraining, aes(x=TEAM_PITCHING_HR, y=TARGET_WINS)) +
  geom_point() + geom_smooth(method = "loess") +
  xlab("Scatterplot with Best Fit Line")

grid.arrange(bp, hp, sp, layout_matrix=rbind(c(1,2,2),c(1,3,3)))
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



Analysis: This variable is more consistent than other pitching variables. The range and distribution are reasonable. Multimodality is interesting similar to a few other variables above. There are 15 zero values which can be imputed as needed.

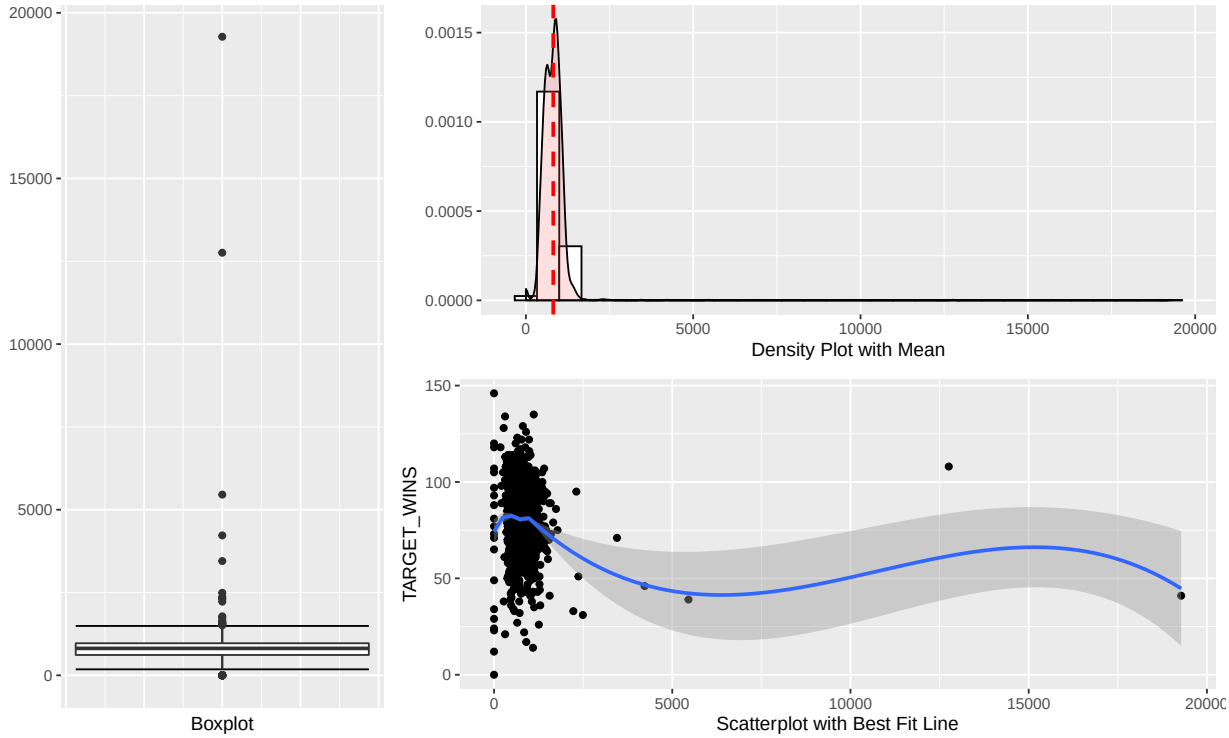
```
kable(sumBTraining[sumBTraining[,1]=="TEAM_PITCHING_SO",2:8], row.names=FALSE)
```

TEAM_PITCHING_SO: Number of strikeouts by pitchers

Min	Median	Mean	SD	Max	Num
0	813.5	817.7305	553.085	19278	

```
# Boxplot
bp <- ggplot(BTraining, aes(x = 1, y = TEAM_PITCHING_SO)) +
  stat_boxplot(geom ='errorbar') + geom_boxplot() +
  xlab("Boxplot") + ylab("") + theme(axis.text.x=element_blank(), axis.ticks.x=element_blank())

# Density plot
hp <- ggplot(BTraining, aes(x = TEAM_PITCHING_SO)) +
  geom_histogram(aes(y=..density..), colour="black", fill="white") +
  geom_density(alpha=.2, fill="#FF6666") + ylab("") + xlab("Density Plot with Mean") +
  geom_vline(aes(xintercept=mean(TEAM_PITCHING_SO, na.rm=TRUE)), color="red", linetype="dashed", size=1

# Scatterplot
sp <- ggplot(data=BTraining, aes(x=TEAM_PITCHING_SO, y=TARGET_WINS)) +
  geom_point() + geom_smooth(method = "loess") +
  xlab("Scatterplot with Best Fit Line")

grid.arrange(bp, hp, sp, layout_matrix=rbind(c(1,2,2),c(1,3,3)))
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



Analysis: This variable has 122 missing or zero values. They can be imputed as needed. There is also an outlier issue. Based on visualizations, this variable will be capped at 2,500 and any value over this will be set to this cap.

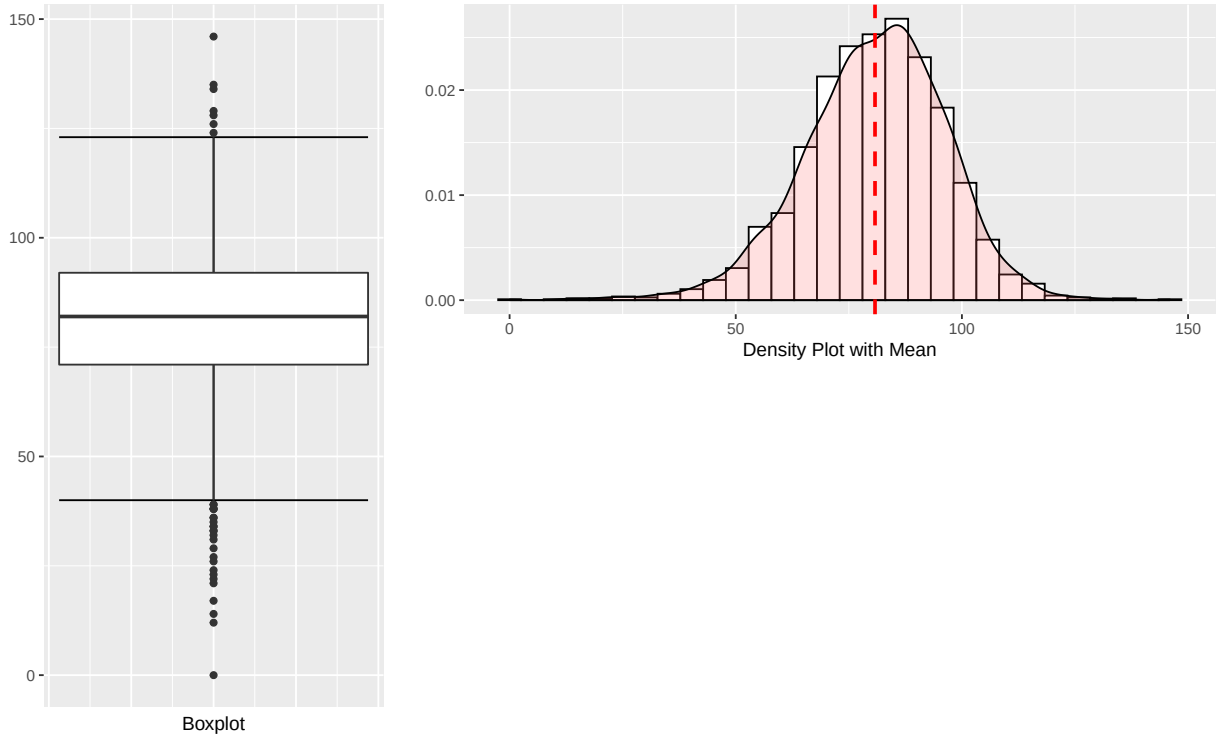
```
kable(sumBTraining[sumBTraining[,1]=="TARGET_WINS",2:8], row.names=FALSE)
```

TARGET_WINS: Number of wins (Outcome)

Min	Median	Mean	SD	Max	Num of NAs	Num
0	82	80.79086	15.75215	146	0	

```
# Boxplot
bp <- ggplot(BTraining, aes(x = 1, y = TARGET_WINS)) +
  stat_boxplot(geom ='errorbar') + geom_boxplot() +
  xlab("Boxplot") + ylab("") + theme(axis.text.x=element_blank(), axis.ticks.x=element_blank())

# Density plot
hp <- ggplot(BTraining, aes(x = TARGET_WINS)) +
  geom_histogram(aes(y=..density..), colour="black", fill="white") +
  geom_density(alpha=.2, fill="#FF6666") + ylab("") + xlab("Density Plot with Mean") +
  geom_vline(aes(xintercept=mean(TARGET_WINS, na.rm=TRUE)), color="red", linetype="dashed", size=1)

grid.arrange(bp, hp, layout_matrix=rbind(c(1,2,2),c(1,3,3)))
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



Analysis: The range and distribution are reasonable. There are no missing values with the exception of record 1347.

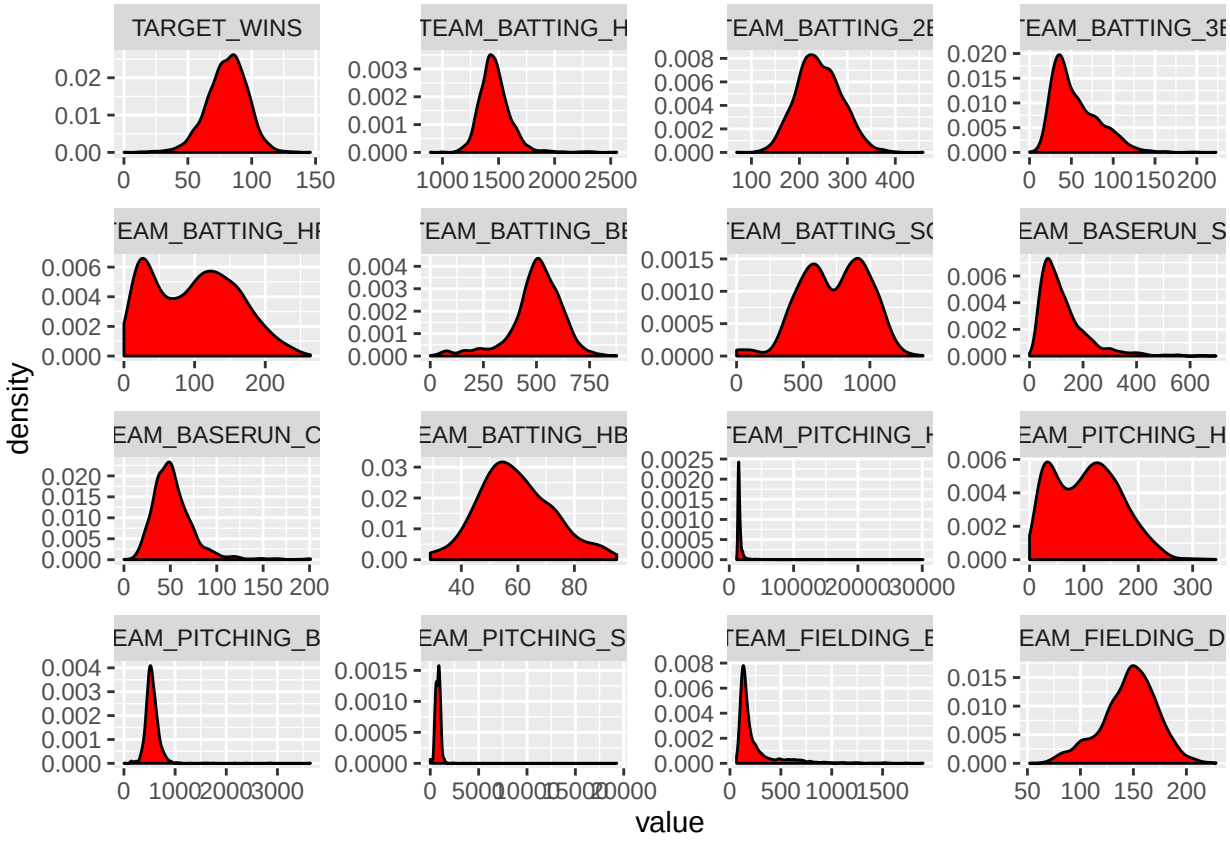
```
par(mfrow = c(3, 3))

datasub = melt(BTraining[c(2:17)])
```

Density Plots

```
## Using  as id variables
```

```
ggplot(datasub, aes(x= value)) +
    geom_density(fill='red') + facet_wrap(~variable, scales = 'free')
```

```
## Warning: Removed 3478 rows containing non-finite values (stat_density).
```



- The density plots show various issues with skew an non-normality

Box Plots Box plots can provide a visual representation of the variance of the data

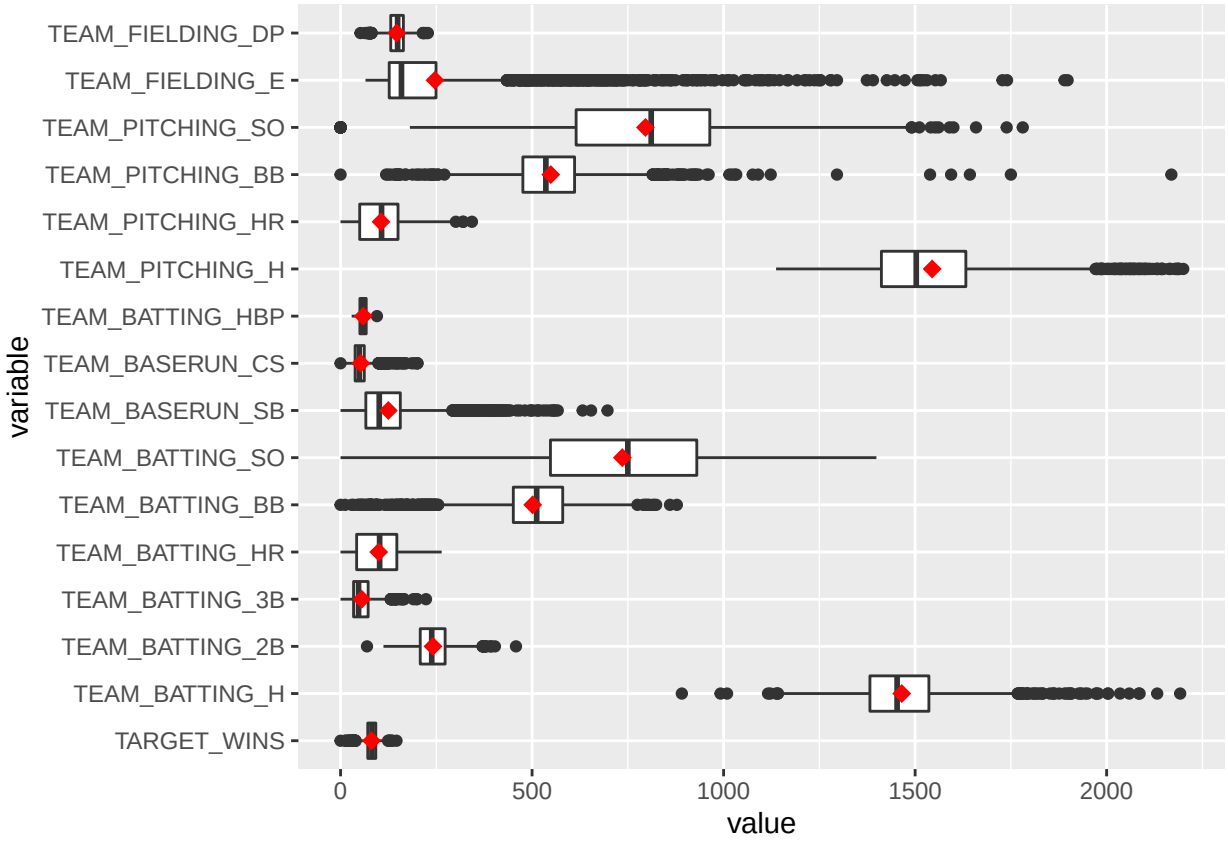
```
vis <- melt(BTraining) %>%
   dplyr::filter(variable != "INDEX")
```

```
## Using  as id variables
```

```
ggplot(vis, aes(x = variable, y = value)) +
  geom_boxplot(show.legend = T) +
  stat_summary(fun.y = mean, color = "red", geom = "point", shape = 18, size = 3) +
  coord_flip() +
  ylim(0, 2200)
```

```
## Warning: Removed 3671 rows containing non-finite values (stat_boxplot).
```

```
## Warning: Removed 3671 rows containing non-finite values (stat_summary).
```



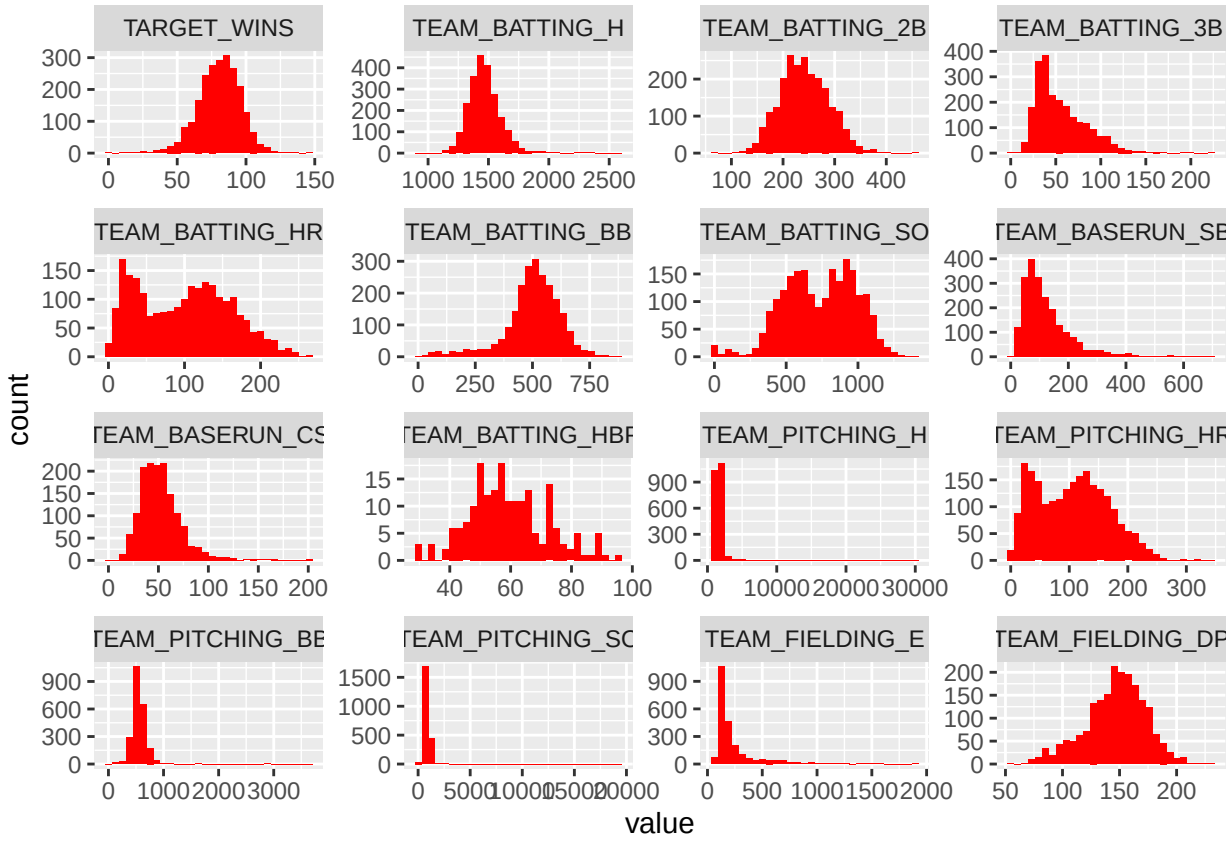
- The box plots reveal that a great majority of the explanatory variables have high variances
- Some of the variables contain extreme outliers that this graph does not show because i had to reduce the limits on the graph to get clear box plots
- Many of the medians and means are also not aligned which demonstrates the outliers' effects

```
ggplot(vis, aes(value)) +
  geom_histogram(fill = "red")  +
  facet_wrap(~ variable, scales = "free")
```

Histograms

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

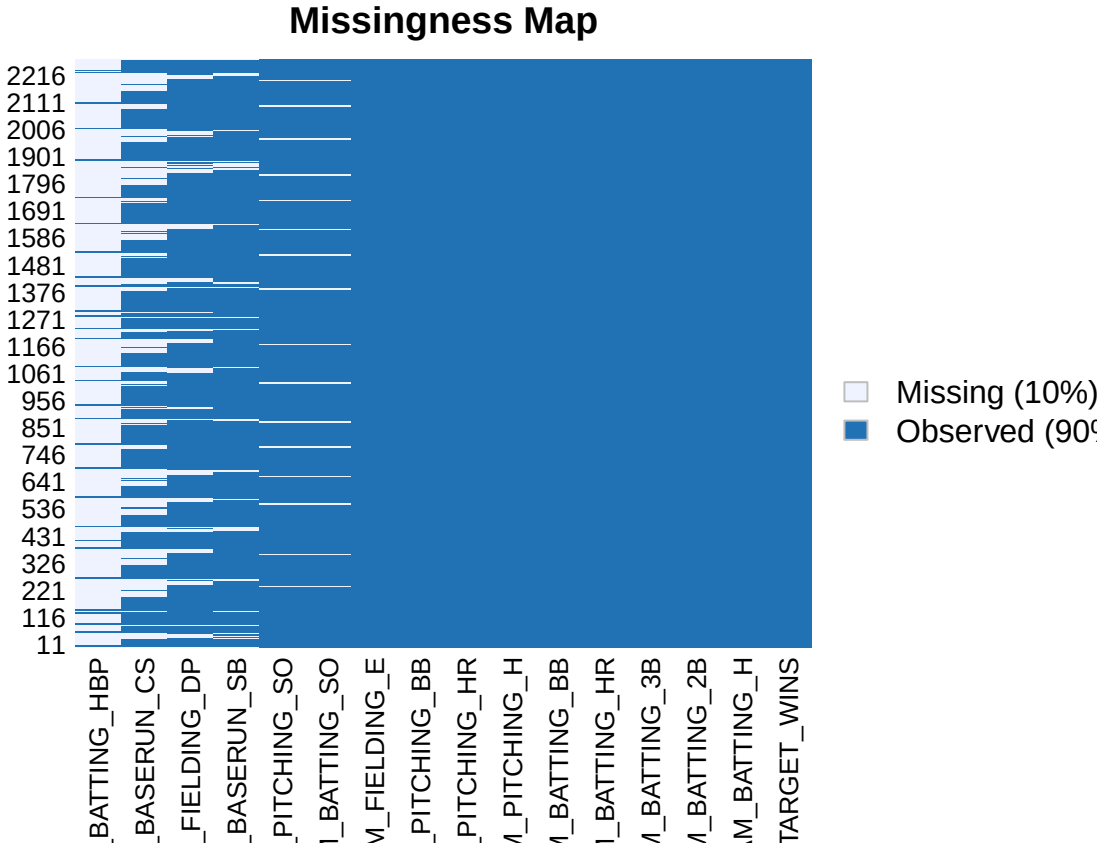
```
## Warning: Removed 3478 rows containing non-finite values (stat_bin).
```



- The histograms reveal that very few of the variables are normally distributed
- A few variables are multi-modal
- Some of the variable exhibit a lot of skew (e.g. BASERUN_SB)

```
missmap(BTraining[2:17])
```

Missing Values



```
# Correlation matrix
cm <- cor(BTraining, use="pairwise.complete.obs")
cm <- cm[2:17,2:17]
names <- c("Wins", "H", "2B", "3B", "HR", "BB", "SO", "SB", "CS", "HBP", "P-H", "P-HR", "P-BB", "P-SO",
colnames(cm) <- names; rownames(cm) <- names
cm <- round(cm, 2)
cmout <- as.data.frame(cm) %>% mutate_all(function(x) {
  cell_spec(x, "latex", color = ifelse(x>0.5 | x<(-0.5),"blue","black"))
  })
rownames(cmout) <- names
cmout %>%
  kable("latex", escape = F, align = "c", row.names = TRUE) %>%
  kable_styling("striped", full_width = F) %>%
  row_spec(0, angle = -90)
```

	Wins	H	2B	3B	HR	BB	SO	SB	CS	HBP	P-H	P-HR	P-BB	P-SO	E
Wins	1	0.39	0.29	0.14	0.18	0.23	-0.03	0.14	0.02	0.07	-0.11	0.19	0.12	-0.08	-0.18
H	0.39	1	0.56	0.43	-0.01	-0.07	-0.46	0.12	0.02	-0.03	0.3	0.07	0.09	-0.25	0.26
2B	0.29	0.56	1	-0.11	0.44	0.26	0.16	-0.2	-0.1	0.05	0.02	0.45	0.18	0.06	-0.24
3B	0.14	0.43	-0.11	1	-0.64	-0.29	-0.67	0.53	0.35	-0.17	0.19	-0.57	0	-0.26	0.51
HR	0.18	-0.01	0.44	-0.64	1	0.51	0.73	-0.45	-0.43	0.11	-0.25	0.97	0.14	0.18	-0.59
BB	0.23	-0.07	0.26	-0.29	0.51	1	0.38	-0.11	-0.14	0.05	-0.45	0.46	0.49	-0.02	-0.66
SO	-0.03	-0.46	0.16	-0.67	0.73	0.38	1	-0.25	-0.22	0.22	-0.38	0.67	0.04	0.42	-0.58
SB	0.14	0.12	-0.2	0.53	-0.45	-0.11	-0.25	1	0.66	-0.06	0.07	-0.42	0.15	-0.14	0.51
CS	0.02	0.02	-0.1	0.35	-0.43	-0.14	-0.22	0.66	1	-0.07	-0.05	-0.42	-0.11	-0.21	0.05
HBP	0.07	-0.03	0.05	-0.17	0.11	0.05	0.22	-0.06	-0.07	1	-0.03	0.11	0.05	0.22	0.04
P-H	-0.11	0.3	0.02	0.19	-0.25	-0.45	-0.38	0.07	-0.05	-0.03	1	-0.14	0.32	0.27	0.67
P-HR	0.19	0.07	0.45	-0.57	0.97	0.46	0.67	-0.42	-0.42	0.11	-0.14	1	0.22	0.21	-0.49
P-BB	0.12	0.09	0.18	0	0.14	0.49	0.04	0.15	-0.11	0.05	0.32	0.22	1	0.49	-0.02
P-SO	-0.08	-0.25	0.06	-0.26	0.18	-0.02	0.42	-0.14	-0.21	0.22	0.27	0.21	0.49	1	-0.02
E	-0.18	0.26	-0.24	0.51	-0.59	-0.66	-0.58	0.51	0.05	0.04	0.67	-0.49	-0.02	-0.02	1
DP	-0.03	0.16	0.29	-0.32	0.45	0.43	0.15	-0.5	-0.21	-0.07	-0.23	0.44	0.32	0.03	-0.5

```
# cmout %>%
#   kable("html", escape = F, align = "c", row.names = TRUE) %>%
#   kable_styling("striped", full_width = F)
#
```

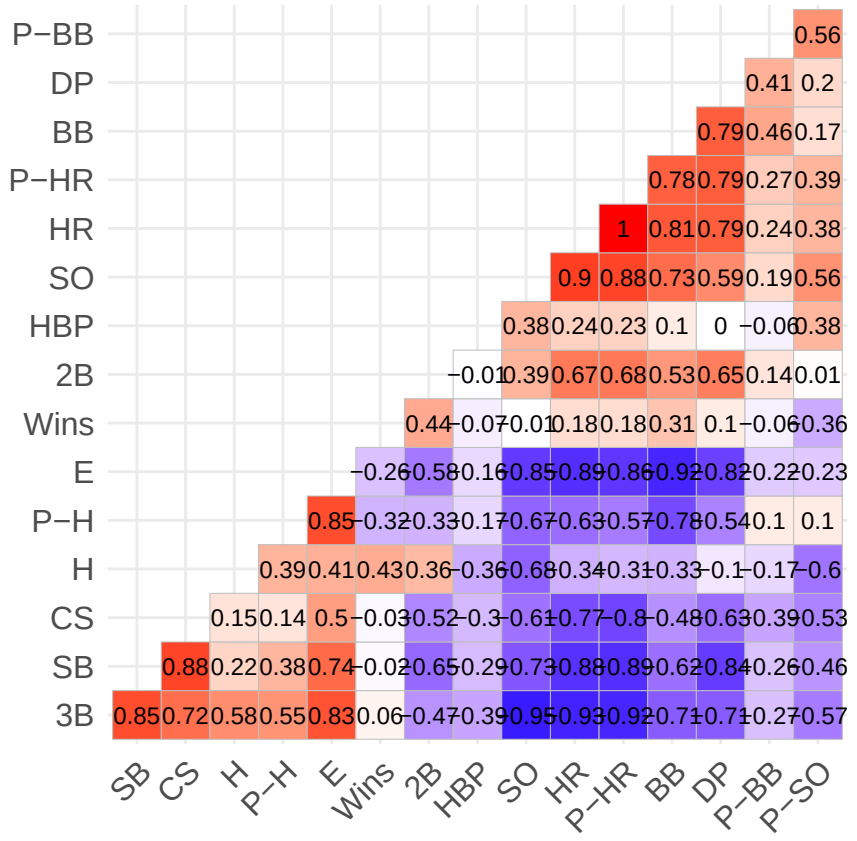
Correlation Matrix Anything over 0.5 or under -0.5 is highlighted in blue. The matrix was created using complete pairwise observations.

A few conclusions:

- Not surprisingly there is a very strong correlation between home runs batted in and home runs given up by pitching.
- There is a negative correlation between number of triples and home runs. A less powerful team may not have enough power to hit home runs, but they get a lot of triples.

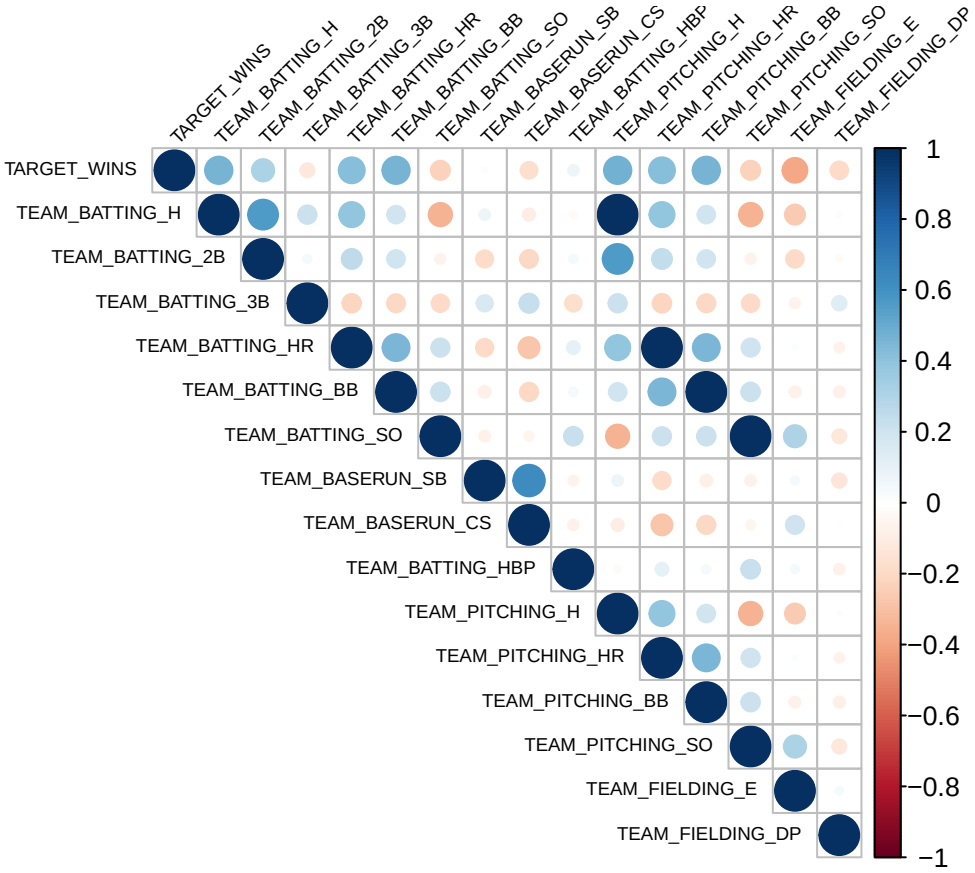
- There is a strong positive correlation between number of strikeouts and home runs. More swings of the bat results in more home runs.

```
#chart.Correlation(BTraining)

corr <- cor(cm, use="complete.ob")
ggcorrplot(corr, hc.order = TRUE, type = "lower",
   lab = TRUE, lab_size = 3, digits = 2)
```



Other correlation visualization

```
cor_res <- cor(BTraining[c(2:17)], use = "na.or.complete")
corrplot(cor_res,
  type = "upper",
  order = "original",
  tl.col = "black",
  tl.srt = 45,
  tl.cex = 0.55)
```



Let's read the correlations respective values:

```
cor_res.df <- data.frame(cor_res)
cor_res.df[1]
```

```
##                  TARGET_WINS
## TARGET_WINS       1.00000000
## TEAM_BATTING_H    0.46994665
## TEAM_BATTING_2B   0.31298400
## TEAM_BATTING_3B  -0.12434586
## TEAM_BATTING_HR   0.42241683
## TEAM_BATTING_BB   0.46868793
## TEAM_BATTING_SO  -0.22889273
## TEAM_BASERUN_SB   0.01483639
## TEAM_BASERUN_CS  -0.17875598
## TEAM_BATTING_HBP  0.07350424
## TEAM_PITCHING_H   0.47123431
## TEAM_PITCHING_HR  0.42246683
## TEAM_PITCHING_BB  0.46839882
## TEAM_PITCHING_SO -0.22936481
## TEAM_FIELDING_E  -0.38668800
## TEAM_FIELDING_DP -0.19586601
```

We notice also that there are some very strong positive correlations between other variables, represented in the correlation graphic above, among themselves. some of these are:

- **TEAM_BATTING_H** is strongly correlated in a positive way with **TEAM_PITCHING_H**

```
cor_res.df['TEAM_BATTING_H','TEAM_PITCHING_H']
```

```
## [1] 0.9991927
```

- **TEAM_BATTING_HR** is strongly correlated in a positive way with **TEAM_PITCHING_HR**

```
cor_res.df['TEAM_BATTING_HR','TEAM_PITCHING_HR']
```

```
## [1] 0.9999326
```

- **TEAM_BATTING_BB** is strongly correlated in a positive way with **TEAM_PITCHING_BB**

```
cor_res.df['TEAM_BATTING_BB','TEAM_PITCHING_BB']
```

```
## [1] 0.9998814
```

- **TEAM_BATTING_SO** is strongly correlated in a positive way with **TEAM_PITCHING_SO**

```
cor_res.df['TEAM_BATTING_SO','TEAM_PITCHING_SO']
```

```
## [1] 0.9997684
```

Primary insigths Based on the above data exploration, we could note the following:

- **Missing values**

It is confirmed the presence of missing values and these need to be address in a case by case - (will be taken care of in data Preparation section).

- **Zero values**

It is confirmed the presence of Zero values as minimum entries in the data - (will be taken care of in data Preparation section).

- **Correlations**

There seems to be very strong correlations to the target variable.

In regards to correlations related to other variables, it is confirmed that some variables have stronh correlation between each others.

Data Preparation

The following steps and/or assumptions will be considered. The idea is to make our given training data set more homogeneous and workable. The final goal is to be able to predict the TARGET_WINS with our data.

As noted in the **Data Exploration** section, the following adjustments have been performed:

- Record 1347 having 0 for outcome variable `TARGET_WINS` has been removed.
- Variable `TEAM_BATTING_HBP` has been removed.
- Any zero value in all variables has been converted to `NA`.

- Any `NA` value has been imputed using `aregImpute` function of `Hmisc` R package. Please see https://www.rdocumentation.org/packages/Hmisc/versions/4.3-1/topics/aregImpute for a full documentation of this function.
- Imputation and R^2 business:

```
# Remove observations with no target
BTraining <- BTraining[which(BTraining$TARGET_WINS!=0), ]

# Reset zero values
BTraining[which(BTraining$TEAM_BATTING_H==0),"TEAM_BATTING_H"] <- NA
BTraining[which(BTraining$TEAM_BATTING_2B==0),"TEAM_BATTING_2B"] <- NA
BTraining[which(BTraining$TEAM_BATTING_3B==0),"TEAM_BATTING_3B"] <- NA
BTraining[which(BTraining$TEAM_BATTING_HR==0),"TEAM_BATTING_HR"] <- NA
BTraining[which(BTraining$TEAM_BATTING_BB==0),"TEAM_BATTING_BB"] <- NA
BTraining[which(BTraining$TEAM_BATTING_SO==0),"TEAM_BATTING_SO"] <- NA
BTraining[which(BTraining$TEAM_BASERUN_SB==0),"TEAM_BASERUN_SB"] <- NA
BTraining[which(BTraining$TEAM_BASERUN_CS==0),"TEAM_BASERUN_CS"] <- NA
BTraining[which(BTraining$TEAM_FIELDING_E==0),"TEAM_FIELDING_E"] <- NA
BTraining[which(BTraining$TEAM_FIELDING_DP==0),"TEAM_FIELDING_DP"] <- NA
BTraining[which(BTraining$TEAM_PITCHING_BB==0),"TEAM_PITCHING_BB"] <- NA
BTraining[which(BTraining$TEAM_PITCHING_H==0),"TEAM_PITCHING_H"] <- NA
BTraining[which(BTraining$TEAM_PITCHING_HR==0),"TEAM_PITCHING_HR"] <- NA
BTraining[which(BTraining$TEAM_PITCHING_SO==0),"TEAM_PITCHING_SO"] <- NA

# Impute missing values
BTrainingImpute <- aregImpute(~ 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_DP + TEAM_FIELDING_E + TEAM_PITCHING_BB +
                          TEAM_PITCHING_H + TEAM_PITCHING_HR + TEAM_PITCHING_SO,
                        data = BTraining, n.impute = 10)
BTrainingI <- impute.transcan(BTrainingImpute, imputation=10, data=BTraining,
                      list.out=TRUE, pr=FALSE, check=FALSE)
BTraining$TEAM_BASERUN_SB <- BTrainingI$TEAM_BASERUN_SB
BTraining$TEAM_BASERUN_CS <- BTrainingI$TEAM_BASERUN_CS
BTraining$TEAM_BATTING_3B <- BTrainingI$TEAM_BATTING_3B
BTraining$TEAM_BATTING_HR <- BTrainingI$TEAM_BATTING_HR
BTraining$TEAM_BATTING_SO <- BTrainingI$TEAM_BATTING_SO
BTraining$TEAM_FIELDING_DP <- BTrainingI$TEAM_FIELDING_DP
BTraining$TEAM_PITCHING_HR <- BTrainingI$TEAM_PITCHING_HR
BTraining$TEAM_PITCHING_SO <- BTrainingI$TEAM_PITCHING_SO
```

```
BTrainingImpute$rsq
```

```
##  TEAM_BATTING_3B  TEAM_BATTING_HR  TEAM_BATTING_SO  TEAM_BASERUN_SB
##        0.7197856        0.9844894        0.9167682        0.7281020
##  TEAM_BASERUN_CS TEAM_FIELDING_DP TEAM_PITCHING_HR TEAM_PITCHING_SO
##        0.6623715        0.5206226        0.9888505        0.7413410
```

- Outliers for several variables have been capped: `TEAM_PITCHING_SO` at 2,500, `TEAM_PITCHING_H` at 13,000, and `TEAM_PITCHING_BB` at 1100.
- To even out the spread of `TEAM_FIELDING_E` which has a long right tail with low median value, it has been log-transformed.
- A new variable has been created to calculate number of singles batting in. It is equal to number of base hits minus doubles, triples and home runs.

```
# Adjust outliers
BTraining[which(BTraining$TEAM_PITCHING_SO>2500),"TEAM_PITCHING_SO"] <- 2500
BTraining[which(BTraining$TEAM_PITCHING_H>13000),"TEAM_PITCHING_H"] <- 13000
BTraining[which(BTraining$TEAM_PITCHING_BB>1100),"TEAM_PITCHING_BB"] <- 1100

# Create singles
BTraining$TEAM_BATTING_S <- BTraining$TEAM_BATTING_H - BTraining$TEAM_BATTING_2B - BTraining$TEAM_BATTI
summary(BTraining$TEAM_BATTING_S)

# Create log fielding error
BTraining$TEAM_FIELDING_E_LOG <- log(BTraining$TEAM_FIELDING_E)
```

Model Building

Model 1

The first model includes several variables, selected manually, that have higher than average correlation to the target variable. They cover hitting, walking and fielding errors.

```
model1 <- lm(TARGET_WINS ~ TEAM_BATTING_H + TEAM_BATTING_BB + TEAM_FIELDING_E, data=BTraining)
summary(model1)
```

```
##
## Call:
## lm(formula = TARGET_WINS ~ TEAM_BATTING_H + TEAM_BATTING_BB +
##     TEAM_FIELDING_E, data = BTraining)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -50.971  -9.022   0.101   9.062  51.557
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)      4.089556   3.311975   1.235    0.217
## TEAM_BATTING_H   0.049143   0.002100  23.403  < 2e-16 ***
## TEAM_BATTING_BB  0.016107   0.003136   5.137 3.03e-07 ***
## TEAM_FIELDING_E -0.014493   0.001768  -8.196 4.11e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 13.7 on 2271 degrees of freedom
## Multiple R-squared:  0.2356, Adjusted R-squared:  0.2346
## F-statistic: 233.3 on 3 and 2271 DF,  p-value: < 2.2e-16
```

All variables are significant, but the R^2 value is relatively small at 0.2356.

Model 2

The second model expand the base hit variable, `TEAM_BATTING_H`, into its components - singles, doubles, triples and home runs.

```
model2 <- lm(TARGET_WINS ~ TEAM_BATTING_S + TEAM_BATTING_2B + TEAM_BATTING_3B + TEAM_BATTING_HR +
           TEAM_BATTING_BB + TEAM_FIELDING_E, data=BTraining)
summary(model2)
```

```
##
## Call:
## lm(formula = TARGET_WINS ~ TEAM_BATTING_S + TEAM_BATTING_2B +
##     TEAM_BATTING_3B + TEAM_BATTING_HR + TEAM_BATTING_BB + TEAM_FIELDING_E,
##     data = BTraining)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -52.268  -8.767   0.133   8.770  58.611
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)      8.116455   3.443787   2.357  0.01852 *
## TEAM_BATTING_S   0.045455   0.003157  14.400  < 2e-16 ***
## TEAM_BATTING_2B  0.022516   0.007421   3.034  0.00244 **
## TEAM_BATTING_3B  0.161195   0.015132  10.652  < 2e-16 ***
## TEAM_BATTING_HR  0.078670   0.007762  10.135  < 2e-16 ***
## TEAM_BATTING_BB  0.012554   0.003201   3.922 9.05e-05 ***
## TEAM_FIELDING_E -0.018589   0.001977  -9.405  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 13.53 on 2268 degrees of freedom
## Multiple R-squared:  0.2564, Adjusted R-squared:  0.2545
## F-statistic: 130.4 on 6 and 2268 DF,  p-value: < 2.2e-16
```

All variables are still significant and R^2 is slightly improved at 0.2574. Another variation of this model - with log-transformed fielding error variable - produced slightly worse results.

Model 3

The third model includes several variables manually selected to try and cover different aspects of the game:

- `TEAM_BATTING_SO:TEAM_BATTING_H` interaction covers offensive successes (hits) and failures (strike-outs).
- Similarly `TEAM_BATTING_BB:TEAM_BATTING_H` interaction covers interaction between hits and walks.
- `TEAM_BASERUN_SB` covers base running.
- `TEAM_FIELDING_DP` and `TEAM_FIELDING_E_LOG` cover fielding performance.
- `TEAM_PITCHING_HR` covers pitching performance.

```
model3 <- lm(TARGET_WINS ~ TEAM_BATTING_SO:TEAM_BATTING_H + TEAM_BATTING_BB:TEAM_BATTING_H + TEAM_BATTI
           TEAM_BASERUN_SB + TEAM_FIELDING_DP + TEAM_PITCHING_HR + TEAM_FIELDING_E_LOG, data=BTraining)
summary(model3)
```

```
##
## Call:
## lm(formula = TARGET_WINS ~ TEAM_BATTING_SO:TEAM_BATTING_H + TEAM_BATTING_BB:TEAM_BATTING_H +
##     TEAM_BATTING_SO + TEAM_BASERUN_SB + TEAM_FIELDING_DP + TEAM_PITCHING_HR +
##     TEAM_FIELDING_E_LOG, data = BTraining)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -53.456  -8.285  -0.077   8.308  66.305
##
## Coefficients:
##                                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)                    1.768e+02  6.906e+00  25.609  < 2e-16 ***
## TEAM_BATTING_SO               -8.802e-02  5.528e-03 -15.924  < 2e-16 ***
## TEAM_BASERUN_SB                5.005e-02  4.289e-03  11.669  < 2e-16 ***
## TEAM_FIELDING_DP              -1.522e-01  1.328e-02 -11.458  < 2e-16 ***
## TEAM_PITCHING_HR               4.727e-02  7.513e-03   6.292 3.75e-10 ***
## TEAM_FIELDING_E_LOG           -1.473e+01  9.296e-01 -15.849  < 2e-16 ***
## TEAM_BATTING_SO:TEAM_BATTING_H  4.672e-05  4.245e-06  11.005  < 2e-16 ***
## TEAM_BATTING_H:TEAM_BATTING_BB  8.588e-06  1.939e-06   4.429 9.92e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 13.06 on 2267 degrees of freedom
## Multiple R-squared:  0.3074, Adjusted R-squared:  0.3053
## F-statistic: 143.7 on 7 and 2267 DF,  p-value: < 2.2e-16
```

All variables are statistically significant and there is noticeable improvement of the R^2 value at 0.3027.

Model 4

The fourth model started with all variables and used backward elimination to arrive at the optimal model. It started with the following variables: TEAM_BATTING_S, TEAM_BATTING_2B, TEAM_BATTING_3B, TEAM_BATTING_HR, TEAM_BATTING_BB, TEAM_BATTING_SO, TEAM_BASERUN_SB, TEAM_BASERUN_CS, TEAM_FIELDING_DP, TEAM_FIELDING_E, TEAM_PITCHING_BB, TEAM_PITCHING_H, TEAM_PITCHING_SO, and TEAM_PITCHING_HR. It was necessary to remove only one variable - TEAM_BASERUN_CS - to arrive at a model with all significant variables.

```
#model started with all variables
model4 <- lm(TARGET_WINS ~ TEAM_BATTING_S + TEAM_BATTING_2B + TEAM_BATTING_3B +
           TEAM_BATTING_HR + TEAM_BATTING_BB + TEAM_BATTING_SO + TEAM_BASERUN_SB +
           TEAM_BASERUN_CS + TEAM_FIELDING_DP + TEAM_FIELDING_E + TEAM_PITCHING_BB +
           TEAM_PITCHING_H + TEAM_PITCHING_SO + TEAM_PITCHING_HR, data=BTraining)

#then removed TEAM_BASERUN_CS variable, the model becomes
model4 <- lm(TARGET_WINS ~ TEAM_BATTING_S + TEAM_BATTING_2B + TEAM_BATTING_3B +
           TEAM_BATTING_HR + TEAM_BATTING_BB + TEAM_BATTING_SO + TEAM_BASERUN_SB +
           TEAM_FIELDING_DP + TEAM_FIELDING_E + TEAM_PITCHING_BB +
           TEAM_PITCHING_H + TEAM_PITCHING_SO + TEAM_PITCHING_HR, data=BTraining)

summary(model4)
```

```
##
## Call:
## lm(formula = TARGET_WINS ~ TEAM_BATTING_S + TEAM_BATTING_2B +
##     TEAM_BATTING_3B + TEAM_BATTING_HR + TEAM_BATTING_BB + TEAM_BATTING_SO +
##     TEAM_BASERUN_SB + TEAM_FIELDING_DP + TEAM_FIELDING_E + TEAM_PITCHING_BB +
##     TEAM_PITCHING_H + TEAM_PITCHING_SO + TEAM_PITCHING_HR, data = BTraining)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -51.215  -8.326   0.095   8.036  48.804
##
## Coefficients:
##                    Estimate Std. Error t value Pr(>|t|)
## (Intercept)      32.2151709  5.2378921   6.150 9.11e-10 ***
## TEAM_BATTING_S    0.0444436  0.0036692  12.113  < 2e-16 ***
## TEAM_BATTING_2B   0.0209810  0.0070465   2.978 0.002937 **
## TEAM_BATTING_3B   0.0972874  0.0153353   6.344 2.70e-10 ***
## TEAM_BATTING_HR   0.2013376  0.0321212   6.268 4.36e-10 ***
## TEAM_BATTING_BB   0.0417369  0.0095448   4.373 1.28e-05 ***
## TEAM_BATTING_SO  -0.0334824  0.0044007  -7.608 4.05e-14 ***
## TEAM_BASERUN_SB   0.0489973  0.0042811  11.445  < 2e-16 ***
## TEAM_FIELDING_DP -0.1414384  0.0127090 -11.129  < 2e-16 ***
## TEAM_FIELDING_E  -0.0462308  0.0028226 -16.379  < 2e-16 ***
## TEAM_PITCHING_BB -0.0279301  0.0079589  -3.509 0.000458 ***
## TEAM_PITCHING_H   0.0030870  0.0005723   5.394 7.60e-08 ***
## TEAM_PITCHING_SO  0.0206029  0.0033822   6.092 1.31e-09 ***
## TEAM_PITCHING_HR -0.0743654  0.0288251  -2.580 0.009946 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 12.52 on 2261 degrees of freedom
```

```
## Multiple R-squared:  0.3648, Adjusted R-squared:  0.3611
## F-statistic: 99.87 on 13 and 2261 DF,  p-value: < 2.2e-16
```

The R^2 value is 0.3581.

Model 5

Additionally, several models were created by trying out some variables and there interactions. Variables were selected either based on theoretical expectation or correlation information from the first section. The following model has R^2 values of 0.3279, which is relatively close to the fourth model; however, this model has fewer variables and may be preferential because of its simplicity.

```
model5 <- lm(TARGET_WINS ~ TEAM_BATTING_S  + TEAM_BATTING_3B +
           TEAM_BATTING_HR + TEAM_BASERUN_SB  +
           TEAM_FIELDING_E_LOG*TEAM_PITCHING_H, data=BTraining)
summary(model5)
```

```
##
## Call:
## lm(formula = TARGET_WINS ~ TEAM_BATTING_S + TEAM_BATTING_3B +
##     TEAM_BATTING_HR + TEAM_BASERUN_SB + TEAM_FIELDING_E_LOG *
##     TEAM_PITCHING_H, data = BTraining)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -47.499  -8.831   0.046   8.528  54.478
##
## Coefficients:
##                                     Estimate Std. Error t value Pr(>|t|)
## (Intercept)                       38.1148230  8.1565697   4.673 3.14e-06 ***
## TEAM_BATTING_S                     0.0463385  0.0030777  15.056  < 2e-16 ***
## TEAM_BATTING_3B                    0.1475886  0.0153459   9.617  < 2e-16 ***
## TEAM_BATTING_HR                    0.0737792  0.0072236  10.214  < 2e-16 ***
## TEAM_BASERUN_SB                    0.0530925  0.0039374  13.484  < 2e-16 ***
## TEAM_FIELDING_E_LOG               -7.7716343  1.4022366  -5.542 3.33e-08 ***
## TEAM_PITCHING_H                    0.0299094  0.0045297   6.603 5.01e-11 ***
## TEAM_FIELDING_E_LOG:TEAM_PITCHING_H -0.0042761  0.0006399  -6.683 2.94e-11 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 12.92 on 2267 degrees of freedom
## Multiple R-squared:  0.3222, Adjusted R-squared:  0.3201
## F-statistic:   154 on 7 and 2267 DF,  p-value: < 2.2e-16
```

The R^2 value is 0.3294.

Model Selection

Based on R^2 value, the fourth model (R^2=0.358)was selected for further analysis. This model also has the lowest AIC score.

```
AIC(model1, model2, model3, model4, model5)
```

```
##        df      AIC
## model1  5 18372.69
## model2  8 18315.83
## model3  9 18156.32
## model4 15 17971.67
## model5  9 18107.12
```

```
summary(model4)
```

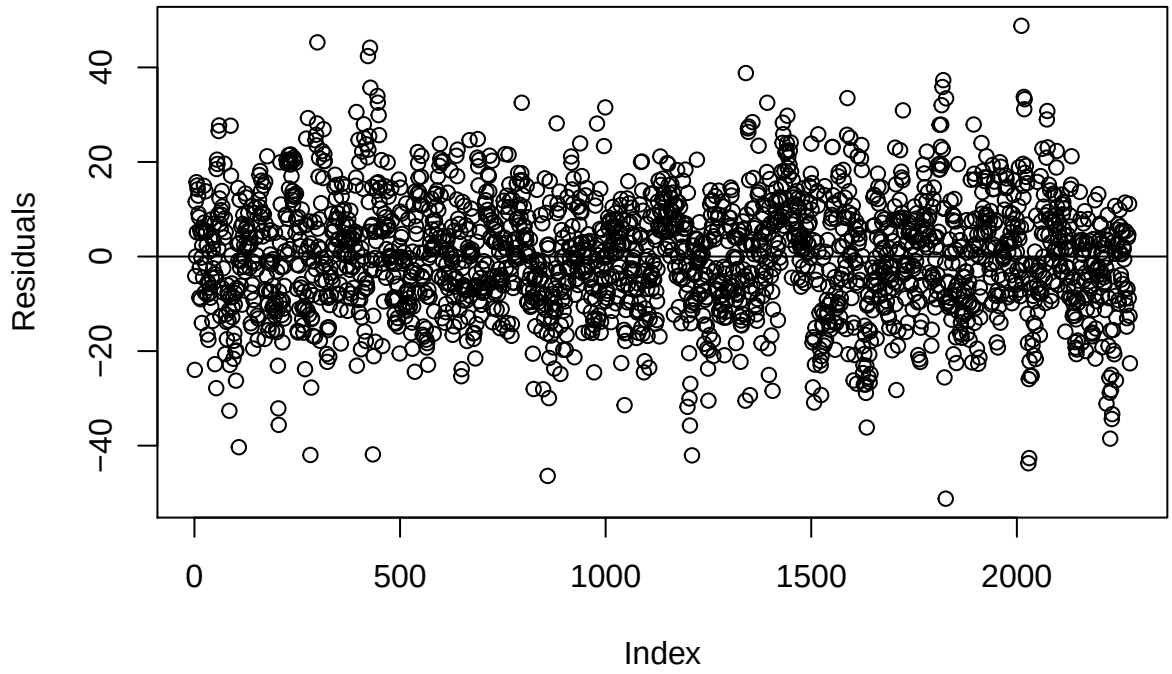
```
##
## Call:
## lm(formula = TARGET_WINS ~ TEAM_BATTING_S + TEAM_BATTING_2B +
##     TEAM_BATTING_3B + TEAM_BATTING_HR + TEAM_BATTING_BB + TEAM_BATTING_SO +
##     TEAM_BASERUN_SB + TEAM_FIELDING_DP + TEAM_FIELDING_E + TEAM_PITCHING_BB +
##     TEAM_PITCHING_H + TEAM_PITCHING_SO + TEAM_PITCHING_HR, data = BTraining)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -51.215  -8.326   0.095   8.036  48.804
##
## Coefficients:
##                    Estimate Std. Error t value Pr(>|t|)
## (Intercept)      32.2151709  5.2378921   6.150 9.11e-10 ***
## TEAM_BATTING_S    0.0444436  0.0036692  12.113  < 2e-16 ***
## TEAM_BATTING_2B   0.0209810  0.0070465   2.978 0.002937 **
## TEAM_BATTING_3B   0.0972874  0.0153353   6.344 2.70e-10 ***
## TEAM_BATTING_HR   0.2013376  0.0321212   6.268 4.36e-10 ***
## TEAM_BATTING_BB   0.0417369  0.0095448   4.373 1.28e-05 ***
## TEAM_BATTING_SO  -0.0334824  0.0044007  -7.608 4.05e-14 ***
## TEAM_BASERUN_SB   0.0489973  0.0042811  11.445  < 2e-16 ***
## TEAM_FIELDING_DP -0.1414384  0.0127090 -11.129  < 2e-16 ***
## TEAM_FIELDING_E  -0.0462308  0.0028226 -16.379  < 2e-16 ***
## TEAM_PITCHING_BB -0.0279301  0.0079589  -3.509 0.000458 ***
## TEAM_PITCHING_H   0.0030870  0.0005723   5.394 7.60e-08 ***
## TEAM_PITCHING_SO  0.0206029  0.0033822   6.092 1.31e-09 ***
## TEAM_PITCHING_HR -0.0743654  0.0288251  -2.580 0.009946 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 12.52 on 2261 degrees of freedom
## Multiple R-squared:  0.3648, Adjusted R-squared:  0.3611
## F-statistic: 99.87 on 13 and 2261 DF,  p-value: < 2.2e-16
```

All variables used in this model have statistical significance at 0.01 level. The F-statistic is high with a p-value nearly 0 and, therefore, is significant. Median value of residuals is close to 0 and they are equally distributed. Standard errors are significantly smaller than estimated coefficients.

Only 4 variables are negatively correlated - strikeouts, double plays, errors, and home runs allowed. Remaining variables are positively correlated. Some correlation is counter-intuitive. For example, double plays are considered successful defensive moves and should increase the winning percentage. Similarly, allowing base hits should decrease winning percentage. The model indicates otherwise and there are probably there factors that influence these variables.

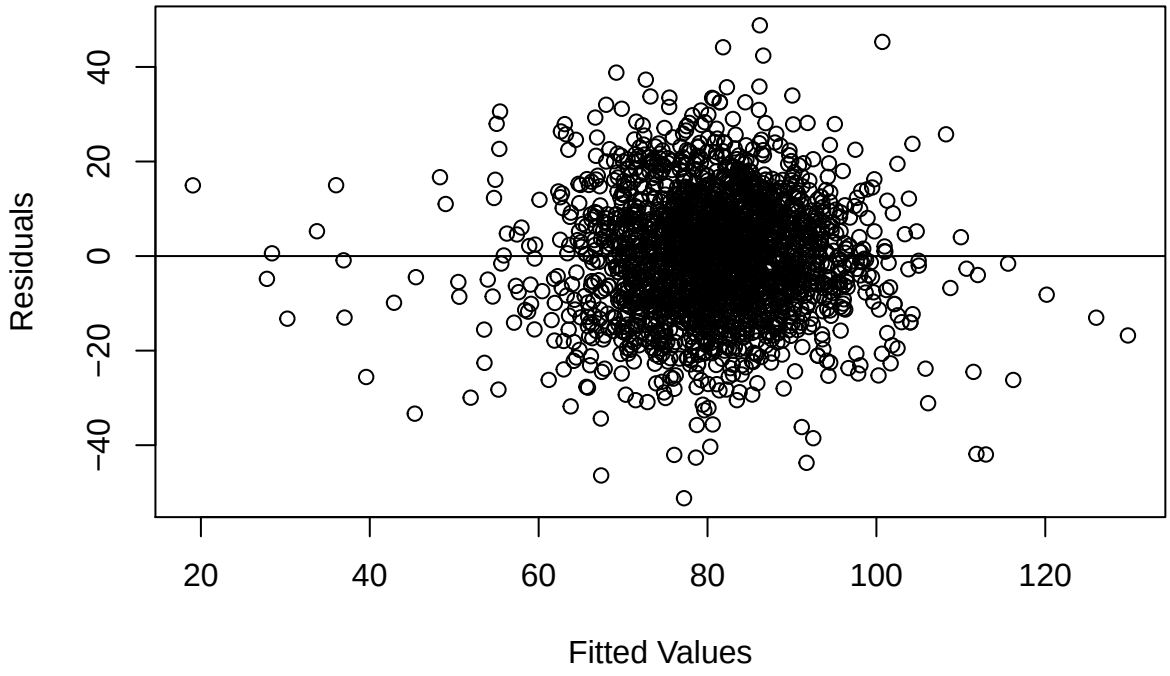
Consider residuals plotted against data index. There is no pattern.

```
plot(model4$residuals, ylab="Residuals")
abline(h=0)
```



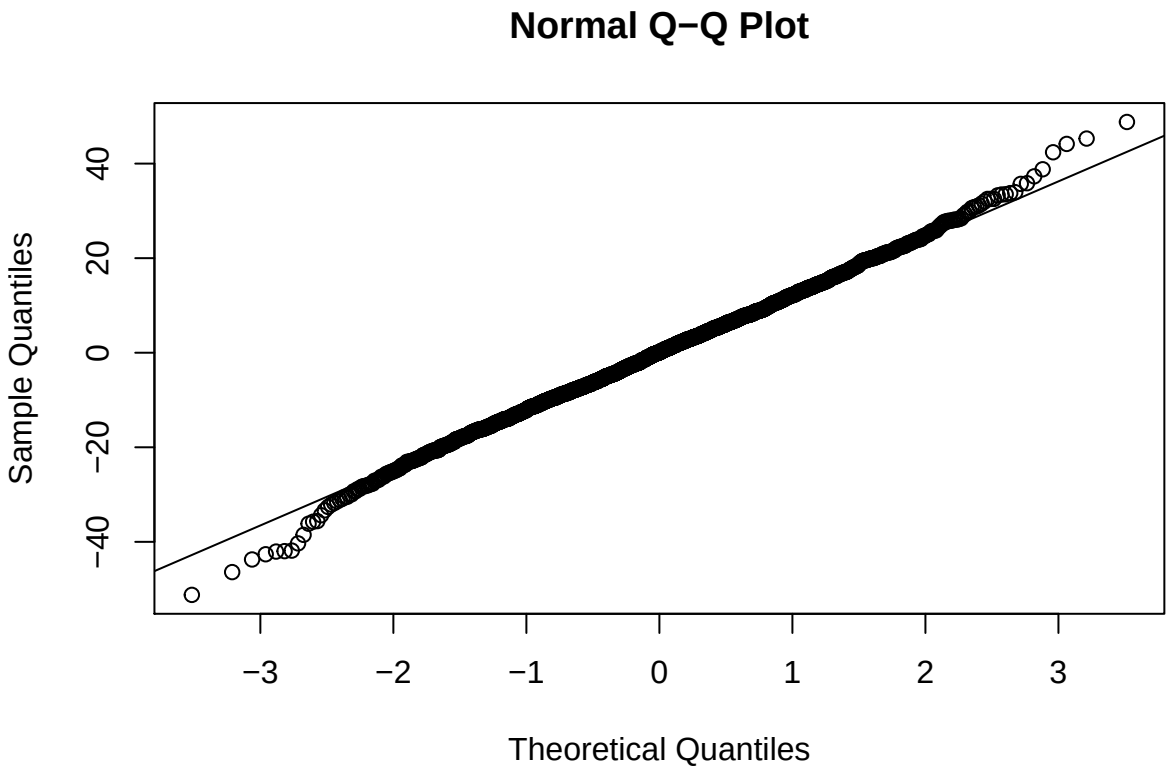
Plotting fitted values against the residuals is more problematic. Although there is no pattern among residuals, there are some outliers and variability does not appear to be constant across the entire range.

```
plot(model4$fitted.values, model4$residuals, xlab="Fitted Values", ylab="Residuals")
abline(h=0)
```



Q-Q plot confirms that residuals are normally distributed.

```
qqnorm(model4$residuals)
qqline(model4$residuals)
```



Prediction

Using selected model and evaluation data (transformed similarly to training data), prediction table is as follows. It includes predicted number of wins along with confidence interval(CI).

```
#BEvaluation <- read.csv("moneyball-evaluation-data.csv")
BEvaluation <- read.csv("https://raw.githubusercontent.com/theoracley/Data621/master/Homework1/moneyball

BEvaluation[which(BEvaluation$TEAM_BATTING_H==0),"TEAM_BATTING_H"] <- NA
BEvaluation[which(BEvaluation$TEAM_BATTING_2B==0),"TEAM_BATTING_2B"] <- NA
BEvaluation[which(BEvaluation$TEAM_BATTING_3B==0),"TEAM_BATTING_3B"] <- NA
BEvaluation[which(BEvaluation$TEAM_BATTING_HR==0),"TEAM_BATTING_HR"] <- NA
BEvaluation[which(BEvaluation$TEAM_BATTING_BB==0),"TEAM_BATTING_BB"] <- NA
BEvaluation[which(BEvaluation$TEAM_BATTING_SO==0),"TEAM_BATTING_SO"] <- NA
BEvaluation[which(BEvaluation$TEAM_BASERUN_SB==0),"TEAM_BASERUN_SB"] <- NA
BEvaluation[which(BEvaluation$TEAM_BASERUN_CS==0),"TEAM_BASERUN_CS"] <- NA
BEvaluation[which(BEvaluation$TEAM_FIELDING_E==0),"TEAM_FIELDING_E"] <- NA
BEvaluation[which(BEvaluation$TEAM_FIELDING_DP==0),"TEAM_FIELDING_DP"] <- NA
BEvaluation[which(BEvaluation$TEAM_PITCHING_BB==0),"TEAM_PITCHING_BB"] <- NA
BEvaluation[which(BEvaluation$TEAM_PITCHING_H==0),"TEAM_PITCHING_H"] <- NA
BEvaluation[which(BEvaluation$TEAM_PITCHING_HR==0),"TEAM_PITCHING_HR"] <- NA
BEvaluation[which(BEvaluation$TEAM_PITCHING_SO==0),"TEAM_PITCHING_SO"] <- NA

# Impute mimssing values
BTrainingImpute <- aregImpute(~ 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_DP + TEAM_FIELDING_E + TEAM_PITCHING_BB +
                          TEAM_PITCHING_H + TEAM_PITCHING_HR + TEAM_PITCHING_SO,
                        data = BEvaluation, n.impute = 10)
BTrainingImpute
BTrainingImpute$rsq

BTrainingI <- impute.transcan(BTrainingImpute, imputation=10, data=BEvaluation,
                        list.out=TRUE, pr=FALSE, check=FALSE)
BEvaluation$TEAM_BATTING_HR <- BTrainingI$TEAM_BATTING_HR
BEvaluation$TEAM_BATTING_SO <- BTrainingI$TEAM_BATTING_SO
BEvaluation$TEAM_BASERUN_SB <- BTrainingI$TEAM_BASERUN_SB
BEvaluation$TEAM_BASERUN_CS <- BTrainingI$TEAM_BASERUN_CS
BEvaluation$TEAM_FIELDING_DP <- BTrainingI$TEAM_FIELDING_DP
BEvaluation$TEAM_PITCHING_HR <- BTrainingI$TEAM_PITCHING_HR
BEvaluation$TEAM_PITCHING_SO <- BTrainingI$TEAM_PITCHING_SO

# Adjust outliers
BEvaluation[which(BEvaluation$TEAM_PITCHING_SO>2500),"TEAM_PITCHING_SO"] <- 2500
BEvaluation[which(BEvaluation$TEAM_PITCHING_H>13000),"TEAM_PITCHING_H"] <- 13000
BEvaluation[which(BEvaluation$TEAM_PITCHING_BB>1100),"TEAM_PITCHING_BB"] <- 1100

BEvaluation$TEAM_BATTING_S <- BEvaluation$TEAM_BATTING_H - BEvaluation$TEAM_BATTING_2B - BEvaluation$TEA

BEvaluation$PREDICT_WIN <- predict(model4, newdata=BEvaluation, interval="confidence")
```

```
BTrainingPredict <- cbind(BEvaluation$INDEX, BEvaluation$PREDICT_WIN[, 1], BEvaluation$PREDICT_WIN[, 2]
colnames(BTrainingPredict) <- c("Index", "Predicted Wins", "CI Lower", "CI Upper")
kable(round(BTrainingPredict,0))
```

Index	Predicted Wins	CI Lower	CI Upper
9	61	59	63
10	64	62	65
14	74	73	75
47	87	86	89
60	69	66	73
63	63	61	66
74	84	82	86
83	76	75	78
98	70	69	71
120	73	72	74
123	67	66	69
135	82	80	83
138	81	80	83
140	83	81	84
151	87	85	89
153	77	75	79
171	74	73	75
184	78	77	79
193	63	60	66
213	90	89	92
217	81	80	83
226	83	82	85
230	80	78	81
241	71	70	72
291	82	81	83
294	88	87	90
300	54	50	58
348	73	72	75
350	83	81	85
357	74	72	76
367	90	89	92
368	85	84	86
372	81	80	83
382	83	81	84
388	79	78	80
396	87	86	89
398	75	75	76
403	91	89	92
407	82	79	85
410	93	91	94
412	83	81	85
414	93	91	94
436	-18	-26	-10
440	106	103	110
476	95	93	97
479	97	95	99
481	93	90	95
501	77	75	78
503	68	66	69
506	80	78	81
519	76	74	77
522	85	84	87
550	76	75	77
554	73	72	75
566	74	73	75
578	79	78	80
596	92	90	94

Saving our prediction to a prediction file (ourPrediction.csv)

```
write.csv(BTrainingPredict, file = "ourPrediction.csv")
```

APPENDIX:

```
# Required libraries
library(tidyverse)
library(kableExtra)
library(ggplot2)
library(gridExtra)
library(dplyr)
library(Hmisc)
library(PerformanceAnalytics)
library(ggcorrplot)
library(corrplot)
library(Amelia)
library(reshape)
library(ggplot2)

# Import data
BTraining <- read.csv("moneyball-training-data.csv")

#structure
str(BTraining)

#Summary
BTraining.summary <- data.frame(unclass(summary(BTraining[2:17])),
check.names = FALSE,
row.names = NULL,
stringsAsFactors = FALSE)
BTraining.summary

#Columns having NA values
BTraining.summary$TEAM_BATTING_SO[7]
BTraining.summary$TEAM_BASERUN_SB[7]
BTraining.summary$TEAM_BASERUN_CS[7]
BTraining.summary$TEAM_BATTING_HBP[7]
BTraining.summary$TEAM_PITCHING_SO[7]
BTraining.summary$TEAM_FIELDING_DP[7]

# Get summary table
sumBTraining = data.frame(Variable = character(),
                    Min = integer(),
                    Median = integer(),
                    Mean = double(),
                    SD = double(),
                    Max = integer(),
                    Num_NAs = integer(),
                    Num_Zeros = integer())

for (i in 2:17) {
  sumBTraining <- rbind(sumBTraining, data.frame(Variable = colnames(BTraining)[i],
                                    Min = min(BTraining[,i], na.rm=TRUE),
                                    Median = median(BTraining[,i], na.rm=TRUE),
                                    Mean = mean(BTraining[,i], na.rm=TRUE),
                                    SD = sd(BTraining[,i], na.rm=TRUE),
```

```
                                    Max = max(BTraining[,i], na.rm=TRUE),
                                    Num_NAs = sum(is.na(BTraining[,i])),
                                    Num_Zeros = length(which(BTraining[,i]==0)))
                  )
}

# ----This section is for Exploration of Data used for every variable----
kable(sumBTraining[sumBTraining[,1]=="TEAM_BASERUN_SB",2:8], row.names=FALSE)

# Boxplot
bp <- ggplot(BTraining, aes(x = 1, y = TEAM_BASERUN_SB)) +
  stat_boxplot(geom ='errorbar') + geom_boxplot() +
  xlab("Boxplot") + ylab("") + theme(axis.text.x=element_blank(),
                                     axis.ticks.x=element_blank())

# Density plot
hp <- ggplot(BTraining, aes(x = TEAM_BASERUN_SB)) +
  geom_histogram(aes(y=..density..), colour="black", fill="white") +
  geom_density(alpha=.2, fill="#FF6666") + ylab("") + xlab("Density Plot with Mean") +
  geom_vline(aes(xintercept=mean(TEAM_BASERUN_SB, na.rm=TRUE)), color="red",
             linetype="dashed", size=1)

# Scatterplot
sp <- ggplot(data=BTraining, aes(x=TEAM_BASERUN_SB, y=TARGET_WINS)) +
  geom_point() + geom_smooth(method = "loess") +
  xlab("Scatterplot with Best Fit Line")

grid.arrange(bp, hp, sp, layout_matrix=rbind(c(1,2,2),c(1,3,3)))

##-----End of exploratory Data for every variable--------


#Density plots
par(mfrow = c(3, 3))
datasub = melt(BTraining[c(2:17)])
ggplot(datasub, aes(x= value)) +
    geom_density(fill='red') + facet_wrap(~variable, scales = 'free')


# Correlation matrix
cm <- cor(BTraining, use="pairwise.complete.obs")
cm <- cm[2:17,2:17]
names <- c("Wins", "H", "2B", "3B", "HR", "BTraining", "SO", "SB", "CS", "HBP", "P-H",
           "P-HR", "P-BTraining", "P-SO", "E", "DP")
colnames(cm) <- names; rownames(cm) <- names
cm <- round(cm, 2)
cmout <- as.data.frame(cm) %>% mutate_all(function(x) {
  cell_spec(x, "html", color = ifelse(x>0.5 | x<(-0.5),"blue","black"))
  })
rownames(cmout) <- names
cmout %>%
  kable("html", escape = F, align = "c", row.names = TRUE) %>%
  kable_styling("striped", full_width = F)
```

```r
#Box plots
vis <- melt(BTraining) %>%
   dplyr::filter(variable != "INDEX")

ggplot(vis, aes(x = variable, y = value)) +
  geom_boxplot(show.legend = T) +
  stat_summary(fun.y = mean, color = "red", geom = "point", shape = 18, size = 3) +
  coord_flip() +
  ylim(0, 2200)

#Histograms
ggplot(vis, aes(value)) +
  geom_histogram(fill = "red")  +
  facet_wrap(~ variable, scales = "free")

#missing values
missmap(BTraining[2:17])


# Correlation matrix
cm <- cor(BTraining, use="pairwise.complete.obs")
cm <- cm[2:17,2:17]
names <- c("Wins", "H", "2B", "3B", "HR", "BB", "SO", "SB", "CS", "HBP", "P-H", "P-HR", "P-BB", "P-SO",
colnames(cm) <- names; rownames(cm) <- names
cm <- round(cm, 2)
cmout <- as.data.frame(cm) %>% mutate_all(function(x) {
  cell_spec(x, "latex", color = ifelse(x>0.5 | x<(-0.5),"blue","black"))
  })
rownames(cmout) <- names
cmout %>%
  kable("latex", escape = F, align = "c", row.names = TRUE) %>%
  kable_styling("striped", full_width = F) %>%
  row_spec(0, angle = -90)


#Other visualization of correlation
#1.
corr <- cor(cm, use="complete.ob")
ggcorrplot(corr, hc.order = TRUE, type = "lower",
   lab = TRUE, lab_size = 3, digits = 2)

#2.
cor_res <- cor(BTraining[c(2:17)], use = "na.or.complete")
corrplot(cor_res,
  type = "upper",
  order = "original",
  tl.col = "black",
  tl.srt = 45,
  tl.cex = 0.55)

#correlation values
cor_res.df <- data.frame(cor_res)
```

```
cor_res.df[1]


# Remove observations with no target
BTraining <- BTraining[which(BTraining$TARGET_WINS!=0), ]

# Reset zero values
BTraining[which(BTraining$TEAM_BATTING_H==0),"TEAM_BATTING_H"] <- NA
BTraining[which(BTraining$TEAM_BATTING_2B==0),"TEAM_BATTING_2B"] <- NA
BTraining[which(BTraining$TEAM_BATTING_3B==0),"TEAM_BATTING_3B"] <- NA
BTraining[which(BTraining$TEAM_BATTING_HR==0),"TEAM_BATTING_HR"] <- NA
BTraining[which(BTraining$TEAM_BATTING_BB==0),"TEAM_BATTING_BB"] <- NA
BTraining[which(BTraining$TEAM_BATTING_SO==0),"TEAM_BATTING_SO"] <- NA
BTraining[which(BTraining$TEAM_BASERUN_SB==0),"TEAM_BASERUN_SB"] <- NA
BTraining[which(BTraining$TEAM_BASERUN_CS==0),"TEAM_BASERUN_CS"] <- NA
BTraining[which(BTraining$TEAM_FIELDING_E==0),"TEAM_FIELDING_E"] <- NA
BTraining[which(BTraining$TEAM_FIELDING_DP==0),"TEAM_FIELDING_DP"] <- NA
BTraining[which(BTraining$TEAM_PITCHING_BB==0),"TEAM_PITCHING_BB"] <- NA
BTraining[which(BTraining$TEAM_PITCHING_H==0),"TEAM_PITCHING_H"] <- NA
BTraining[which(BTraining$TEAM_PITCHING_HR==0),"TEAM_PITCHING_HR"] <- NA
BTraining[which(BTraining$TEAM_PITCHING_SO==0),"TEAM_PITCHING_SO"] <- NA

# Impute mimssing values
BTrainingImpute <- aregImpute(~ 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_DP + TEAM_FIELDING_E + TEAM_PITCHING_BB +
                        TEAM_PITCHING_H + TEAM_PITCHING_HR + TEAM_PITCHING_SO,
                      data = BTraining, n.impute = 10)

BTrainingI <- impute.transcan(BTrainingImpute, imputation=10, data=BTraining,
                      list.out=TRUE, pr=FALSE, check=FALSE)
BTraining$TEAM_BASERUN_SB <- BTrainingI$TEAM_BASERUN_SB
BTraining$TEAM_BASERUN_CS <- BTrainingI$TEAM_BASERUN_CS
BTraining$TEAM_BATTING_3B <- BTrainingI$TEAM_BATTING_3B
BTraining$TEAM_BATTING_HR <- BTrainingI$TEAM_BATTING_HR
BTraining$TEAM_BATTING_SO <- BTrainingI$TEAM_BATTING_SO
BTraining$TEAM_FIELDING_DP <- BTrainingI$TEAM_FIELDING_DP
BTraining$TEAM_PITCHING_HR <- BTrainingI$TEAM_PITCHING_HR
BTraining$TEAM_PITCHING_SO <- BTrainingI$TEAM_PITCHING_SO

BTrainingImpute$rsq

# Adjust outliers
BTraining[which(BTraining$TEAM_PITCHING_SO>2500),"TEAM_PITCHING_SO"] <- 2500
BTraining[which(BTraining$TEAM_PITCHING_H>13000),"TEAM_PITCHING_H"] <- 13000
BTraining[which(BTraining$TEAM_PITCHING_BB>1100),"TEAM_PITCHING_BB"] <- 1100

# Creat singles
BTraining$TEAM_BATTING_S <- BTraining$TEAM_BATTING_H - BTraining$TEAM_BATTING_2B -
  BTraining$TEAM_BATTING_3B - BTraining$TEAM_BATTING_HR
summary(BTraining$TEAM_BATTING_S)
```

```r
# Create log fielding error
BTraining$TEAM_FIELDING_E_LOG <- log(BTraining$TEAM_FIELDING_E)

# Model building
model1 <- lm(TARGET_WINS ~ TEAM_BATTING_H + TEAM_BATTING_BB + TEAM_FIELDING_E, data=BTraining)
summary(model1)

model2 <- lm(TARGET_WINS ~ TEAM_BATTING_S + TEAM_BATTING_2B + TEAM_BATTING_3B + TEAM_BATTING_HR +
           TEAM_BATTING_BB + TEAM_FIELDING_E, data=BTraining)
summary(model2)
model2b <- lm(TARGET_WINS ~ TEAM_BATTING_S + TEAM_BATTING_2B + TEAM_BATTING_3B +
            TEAM_BATTING_HR + TEAM_BATTING_BB + TEAM_FIELDING_E_LOG, data=BTraining)
summary(model2b)

model3 <- lm(TARGET_WINS ~ TEAM_BATTING_SO:TEAM_BATTING_H + TEAM_BATTING_BB:TEAM_BATTING_H + TEAM_BATTI
           TEAM_BASERUN_SB + TEAM_FIELDING_DP + TEAM_PITCHING_HR + TEAM_FIELDING_E_LOG, data=BTraining)
summary(model3)

model4 <- lm(TARGET_WINS ~ TEAM_BATTING_S + TEAM_BATTING_2B + TEAM_BATTING_3B +
           TEAM_BATTING_HR + TEAM_BATTING_BB + TEAM_BATTING_SO + TEAM_BASERUN_SB +
           TEAM_BASERUN_CS + TEAM_FIELDING_DP + TEAM_FIELDING_E + TEAM_PITCHING_BB +
           TEAM_PITCHING_H + TEAM_PITCHING_SO + TEAM_PITCHING_HR, data=BTraining)
summary(model4)
model4 <- lm(TARGET_WINS ~ TEAM_BATTING_S + TEAM_BATTING_2B + TEAM_BATTING_3B +
           TEAM_BATTING_HR + TEAM_BATTING_BB + TEAM_BATTING_SO + TEAM_BASERUN_SB +
           TEAM_FIELDING_DP + TEAM_FIELDING_E + TEAM_PITCHING_BB +
           TEAM_PITCHING_H + TEAM_PITCHING_SO + TEAM_PITCHING_HR, data=BTraining)
summary(model4)

model5 <- lm(TARGET_WINS ~ TEAM_BATTING_S  + TEAM_BATTING_3B +
           TEAM_BATTING_HR + TEAM_BASERUN_SB  +
           TEAM_FIELDING_E_LOG*TEAM_PITCHING_H, data=BTraining)
summary(model5)

#AIC Score
AIC(model1, model2, model3, model4, model5)
summary(model4)

# Residuals plots
plot(model4$residuals, ylab="Residuals")
abline(h=0)

plot(model4$fitted.values, m4$residuals, xlab="Fitted Values", ylab="Residuals")
abline(h=0)

qqnorm(model4$residuals)
qqline(model4$residuals)

# Test evaluation data for prediction
#BEvaluation <- read.csv("moneyball-evaluation-data.csv")
BEvaluation <- read.csv("https://raw.githubusercontent.com/theoracley/Data621/master/Homework1/moneybal

BEvaluation[which(BEvaluation$TEAM_BATTING_H==0),"TEAM_BATTING_H"] <- NA
```

```
BEvaluation[which(BEvaluation$TEAM_BATTING_2B==0),"TEAM_BATTING_2B"] <- NA
BEvaluation[which(BEvaluation$TEAM_BATTING_3B==0),"TEAM_BATTING_3B"] <- NA
BEvaluation[which(BEvaluation$TEAM_BATTING_HR==0),"TEAM_BATTING_HR"] <- NA
BEvaluation[which(BEvaluation$TEAM_BATTING_BB==0),"TEAM_BATTING_BB"] <- NA
BEvaluation[which(BEvaluation$TEAM_BATTING_SO==0),"TEAM_BATTING_SO"] <- NA
BEvaluation[which(BEvaluation$TEAM_BASERUN_SB==0),"TEAM_BASERUN_SB"] <- NA
BEvaluation[which(BEvaluation$TEAM_BASERUN_CS==0),"TEAM_BASERUN_CS"] <- NA
BEvaluation[which(BEvaluation$TEAM_FIELDING_E==0),"TEAM_FIELDING_E"] <- NA
BEvaluation[which(BEvaluation$TEAM_FIELDING_DP==0),"TEAM_FIELDING_DP"] <- NA
BEvaluation[which(BEvaluation$TEAM_PITCHING_BB==0),"TEAM_PITCHING_BB"] <- NA
BEvaluation[which(BEvaluation$TEAM_PITCHING_H==0),"TEAM_PITCHING_H"] <- NA
BEvaluation[which(BEvaluation$TEAM_PITCHING_HR==0),"TEAM_PITCHING_HR"] <- NA
BEvaluation[which(BEvaluation$TEAM_PITCHING_SO==0),"TEAM_PITCHING_SO"] <- NA

# Impute mimssing values
BTrainingImpute <- aregImpute(~ 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_DP +
                        TEAM_FIELDING_E + TEAM_PITCHING_BB + TEAM_PITCHING_H +
                        TEAM_PITCHING_HR + TEAM_PITCHING_SO,
                      data = BEvaluation, n.impute = 10)

BTrainingImpute$rsq

BTrainingI <- impute.transcan(BTrainingImpute, imputation=10, data=BEvaluation,
                      list.out=TRUE, pr=FALSE, check=FALSE)
BEvaluation$TEAM_BATTING_HR <- BTrainingI$TEAM_BATTING_HR
BEvaluation$TEAM_BATTING_SO <- BTrainingI$TEAM_BATTING_SO
BEvaluation$TEAM_BASERUN_SB <- BTrainingI$TEAM_BASERUN_SB
BEvaluation$TEAM_BASERUN_CS <- BTrainingI$TEAM_BASERUN_CS
BEvaluation$TEAM_FIELDING_DP <- BTrainingI$TEAM_FIELDING_DP
BEvaluation$TEAM_PITCHING_HR <- BTrainingI$TEAM_PITCHING_HR
BEvaluation$TEAM_PITCHING_SO <- BTrainingI$TEAM_PITCHING_SO

# Adjust outliers
BEvaluation[which(BEvaluation$TEAM_PITCHING_SO>2500),"TEAM_PITCHING_SO"] <- 2500
BEvaluation[which(BEvaluation$TEAM_PITCHING_H>13000),"TEAM_PITCHING_H"] <- 13000
BEvaluation[which(BEvaluation$TEAM_PITCHING_BB>1100),"TEAM_PITCHING_BB"] <- 1100

BEvaluation$TEAM_BATTING_S <- BEvaluation$TEAM_BATTING_H - BEvaluation$TEAM_BATTING_2B -
  BEvaluation$TEAM_BATTING_3B - BEvaluation$TEAM_BATTING_HR

BEvaluation$PREDICT_WIN <- predict(model4, newdata=BEvaluation, interval="confidence")

BTrainingPredict <- cbind(BEvaluation$INDEX, BEvaluation$PREDICT_WIN[, 1], BEvaluation$PREDICT_WIN[, 2]
                    BEvaluation$PREDICT_WIN[, 3])
colnames(BTrainingPredict) <- c("Index", "Predicted Wins", "CI Lower", "CI Upper")
round(BTrainingPredict,0)

#write our prediction to a file
write.csv(BTrainingPredict, file = "ourPrediction.csv")
```