Home Work Assignment - 01

Critical Thinking Group 5

Contents

Data Exploration

In this section we will explore how the data looks like. The goal of data exploration is to look at summaries / descriptives for each variable, shape of the distribution, identify variables of interest, decide on how to treat missing values and outliers.

First lets look at the variables.

VARIABLE_NAME	DEFINITION	THEORETICAL_EFFECT
INDEX	Identification Variable (do not use)	None
TARGET_WINS	Number of wins	Target
TEAM_BATTING_H	Base Hits by batters (1B,2B,3B,HR)	Positive Impact on Wins
TEAM_BATTING_2B	Doubles by batters (2B)	Positive Impact on Wins
TEAM_BATTING_3B	Triples by batters (3B)	Positive Impact on Wins
TEAM_BATTING_HR	Homeruns by batters (4B)	Positive Impact on Wins
TEAM_BATTING_BB	Walks by batters	Positive Impact on Wins
TEAM_BATTING_HBP	Batters hit by pitch (get a free base)	Positive Impact on Wins
TEAM_BATTING_SO	Strikeouts by batters	Negative Impact on Wins
TEAM_BASERUN_SB	Stolen bases	Positive Impact on Wins
TEAM_BASERUN_CS	Caught stealing	Negative Impact on Wins
TEAM_FIELDING_E	Errors	Negative Impact on Wins
TEAM_FIELDING_DP	Double Plays	Positive Impact on Wins
TEAM_PITCHING_BB	Walks allowed	Negative Impact on Wins
TEAM_PITCHING_H	Hits allowed	Negative Impact on Wins
TEAM_PITCHING_HR	Homeruns allowed	Negative Impact on Wins
TEAM_PITCHING_SO	Strikeouts by pitchers	Positive Impact on Wins

We notice that all variables are numeric. The variable names seem to follow certain naming pattern to highlight certain arithmetic relationships. In other words, we can compute the number of '1B' hits by taking the difference between overall hits and '2B', '3B', 'HR'. Although such naming and construct is not recommended in normalized database design (as it violates third normal form), it is very frequent practice in the data analytics.

Our predictor input is made of 15 variables. And our dependent variable is one variable called TARGET_WINS.

Below are the variable that have been identified and their respective type and category:

Next we start with a summary of the variables and see what we can infer from the same. The goal is to look at measures of central tendancy and dispersion to see how the variables are currently placed in their structure.

Summary / Descriptives / Correlation

```
ds_stats <- psych::describe(moneyball2, skew = FALSE, na.rm = TRUE)[c(3:6)]
ds_stats <- cbind(VARIABLE_NAME = rownames(ds_stats), ds_stats)
#rownames(ds_stats) <- NULL

Variable<- rownames(ds_stats)

fun <- function(x) sum(!complete.cases(x))
Missing <- sapply(moneyball2[Variable], FUN = fun)

#ds_stats <- cbind(ds_stats, Missing)


# fun <- function(x) mean(x, na.rm=T)
# Mean <- sapply(moneyball2[Variable], FUN = fun)

fun <- function(x, y) cor(y, x, use = "na.or.complete")
Correlation <- sapply(moneyball2[Variable], FUN = fun, y=moneyball2$TARGET_WINS)

ds_stats <- data.frame(cbind(ds_stats, Missing, Correlation))
ds_stats <- left_join(ds_stats, moneyballvars, by="VARIABLE_NAME")
kable(ds_stats)
```

VARIABLE_NAME	mean	sd	median	trimmed	Missing	Correlation	DEFINITION
TARGET_WINS	80.79086	15.75215	82.0	81.31229	0	1.0000000	Number of wins
TEAM_BATTING_H	1469.26977	144.59120	1454.0	1459.04116	0	0.3887675	Base Hits by batter
TEAM_BATTING_2B	241.24692	46.80141	238.0	240.39627	0	0.2891036	Doubles by batters
TEAM_BATTING_3B	55.25000	27.93856	47.0	52.17563	0	0.1426084	Triples by batters (
TEAM_BATTING_HR	99.61204	60.54687	102.0	97.38529	0	0.1761532	Homeruns by batter
TEAM_BATTING_BB	501.55888	122.67086	512.0	512.18331	0	0.2325599	Walks by batters
TEAM_BATTING_SO	735.60534	248.52642	750.0	742.31322	102	-0.0317507	Strikeouts by batter
TEAM_BASERUN_SB	124.76177	87.79117	101.0	110.81188	131	0.1351389	Stolen bases
TEAM_BASERUN_CS	52.80386	22.95634	49.0	50.35963	772	0.0224041	Caught stealing
TEAM_BATTING_HBP	59.35602	12.96712	58.0	58.86275	2085	0.0735042	Batters hit by pitch
TEAM_PITCHING_H	1779.21046	1406.84293	1518.0	1555.89517	0	-0.1099371	Hits allowed
TEAM_PITCHING_HR	105.69859	61.29875	107.0	103.15697	0	0.1890137	Homeruns allowed

VARIABLE_NAME	mean	sd	median	trimmed	Missing	Correlation	DEFINITION
TEAM_PITCHING_BB	553.00791	166.35736	536.5	542.62459	0	0.1241745	Walks allowed
TEAM_PITCHING_SO	817.73045	553.08503	813.5	796.93391	102	-0.0784361	Strikeouts by pitche
TEAM_FIELDING_E	246.48067	227.77097	159.0	193.43798	0	-0.1764848	Errors
TEAM_FIELDING_DP	146.38794	26.22639	149.0	147.57789	286	-0.0348506	Double Plays

Based on the table for the variables listed above, there are some things that stand out:

1. Some of the variables like TEAM_PITCHING_H, TEAM_PITCHING_SO and TEAM_FIELDING_E seem to have outliers which is evident from the mean, median and trimmed mean values.

2. TEAM_BATTING_HBP and TEAM_BASERUN_CS seems to be missing a lot of values which casts doubt on its usefulness as a predictor. Maybe a flag for presense or absense of TEAM_BATTING_HBP and TEAM_BASERUN_CS might be a better predictor. Also given the fact that there is low correlation, we decided to exclude these 2 variables from any missing value or outlier treatment.

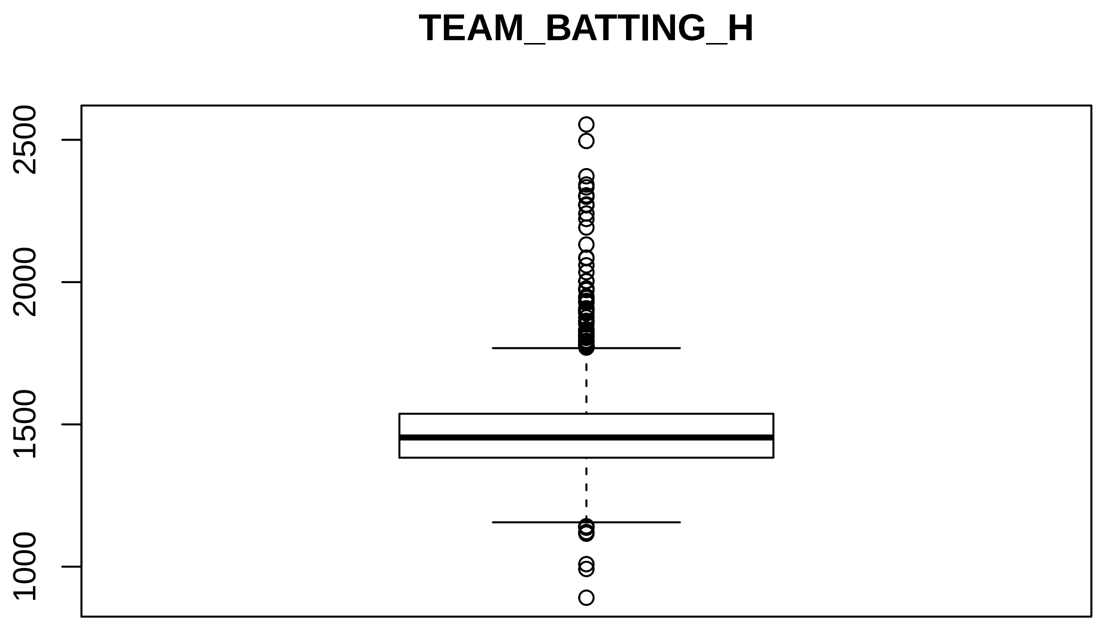
3. Most of the variables seem to indicate a positive / negative correlation in line with the theoretical effect. However, the following stand out as they show a correlation opposite to the theoretical impact: TEAM_BASERUN_CS, TEAM_PITCHING_HR, TEAM_PITCHING_BB, TEAM_PITCHING_SO and TEAM_FIELDING_DP. Lets evaluate these variables further once we fix any missing values or outliers.

4. We will impute the missing values in TEAM_BATTING_SO, FIELDING_DP, BASERUN_SB and TEAM_PITCHING_SO since it has lesser missing values even though there is low correlation. So we will create new variables that will have the respective missing values handled.

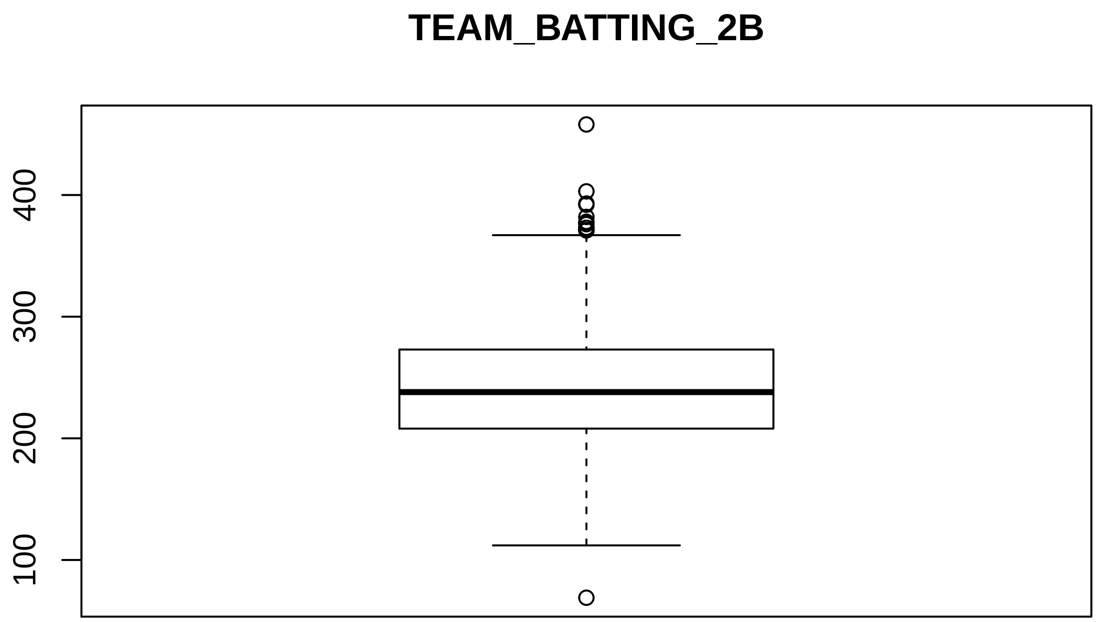
Distribution and Correlation

In this section we look at boxplots to determine the outliers in variables and decide on whether to act on the outliers.

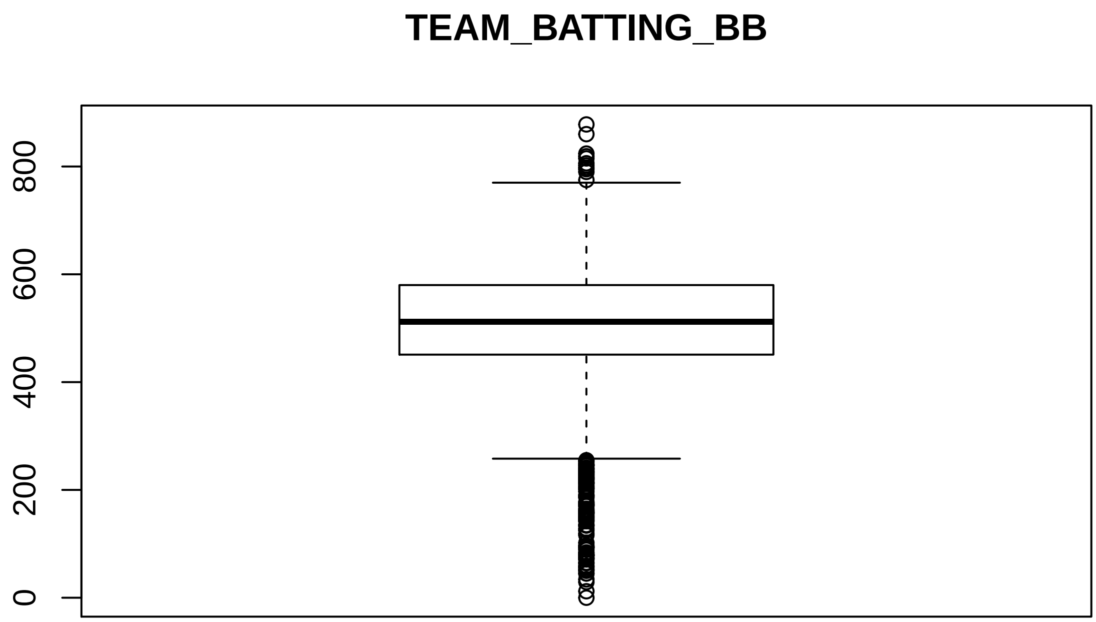
```
#par(mfrow=c(6,2))
boxplot(moneyball2$TEAM_BATTING_H,main="TEAM_BATTING_H")
```



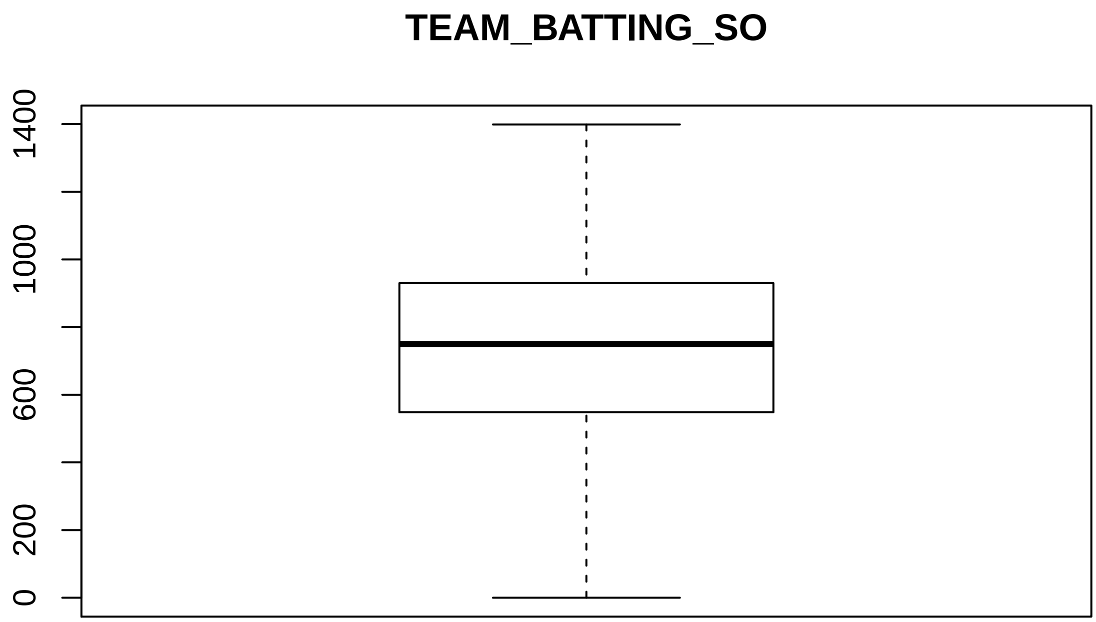

```r
boxplot(moneyball2$TEAM_BATTING_2B,main="TEAM_BATTING_2B")
```



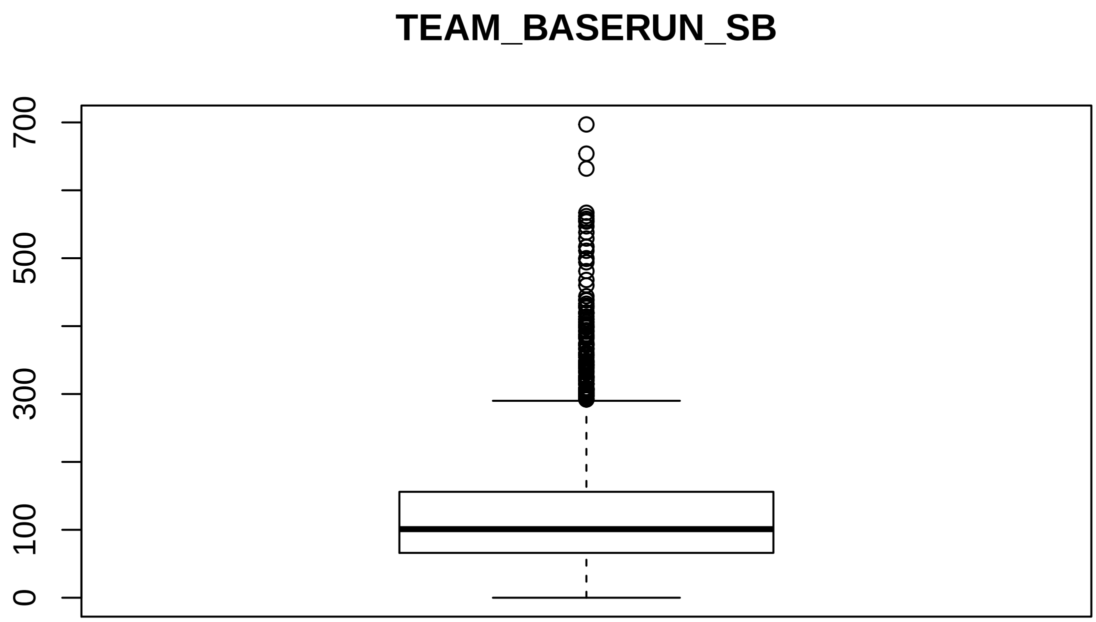

```
boxplot(moneyball2$TEAM_BATTING_BB,main="TEAM_BATTING_BB")
```



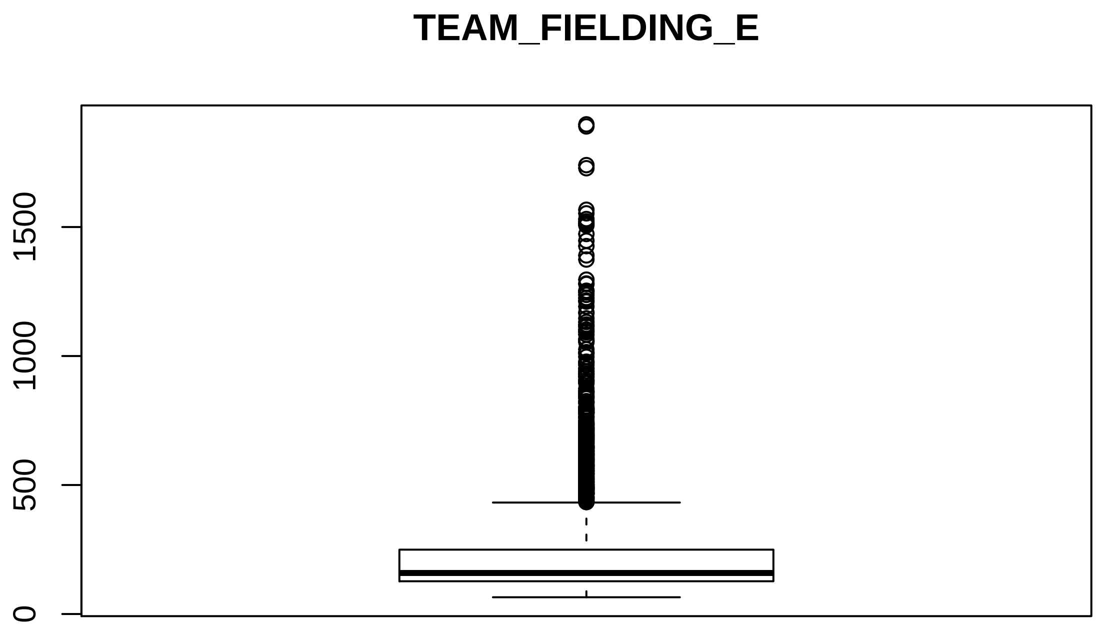

```r
boxplot(moneyball2$TEAM_BATTING_SO,main="TEAM_BATTING_SO")
```



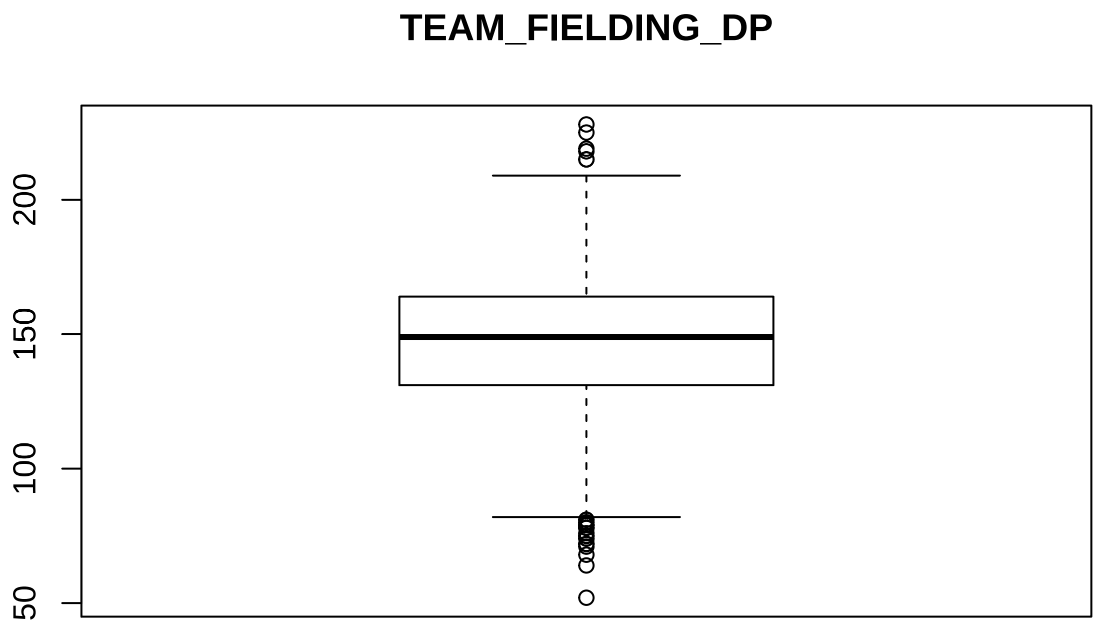

```
boxplot(moneyball2$TEAM_BASERUN_SB,main="TEAM_BASERUN_SB")
```



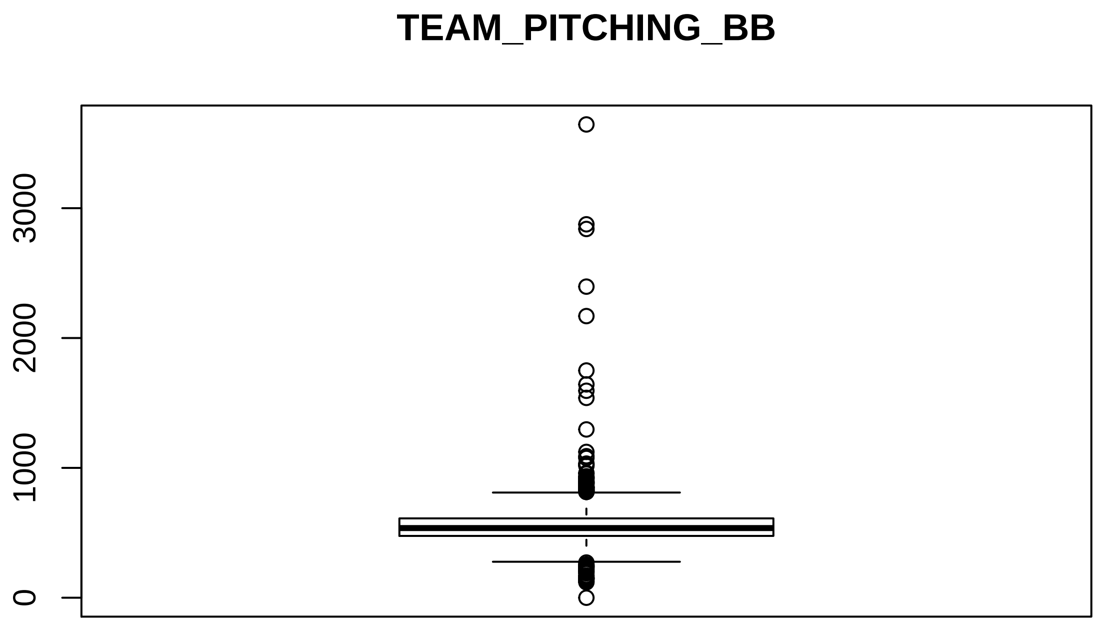

```
boxplot(moneyball2$TEAM_FIELDING_E,main="TEAM_FIELDING_E")
```



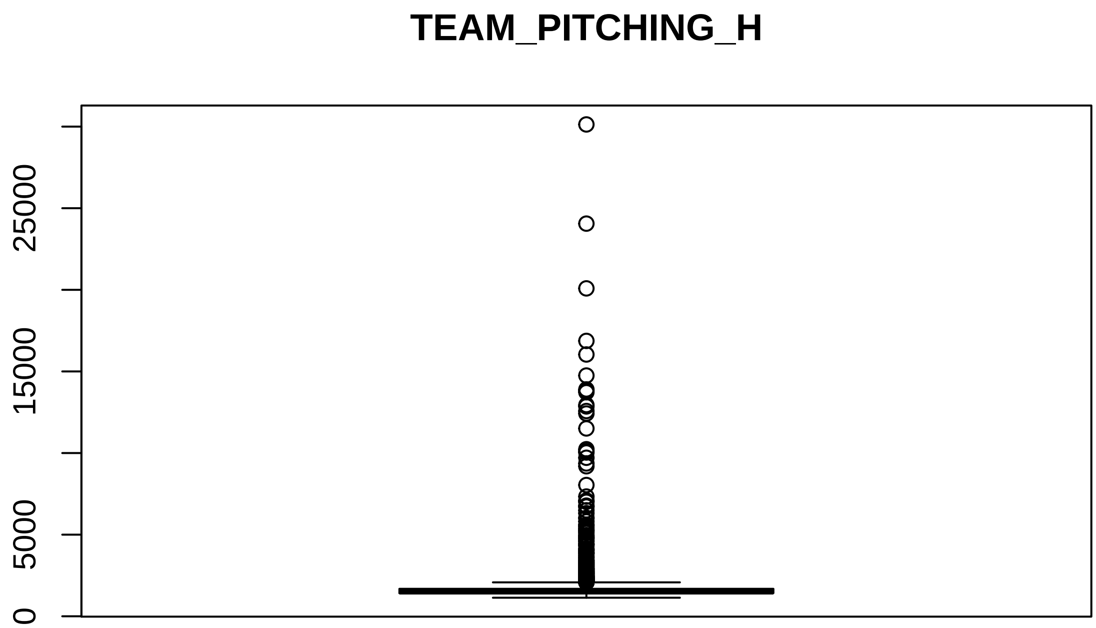

```
boxplot(moneyball2$TEAM_FIELDING_DP,main="TEAM_FIELDING_DP")
```



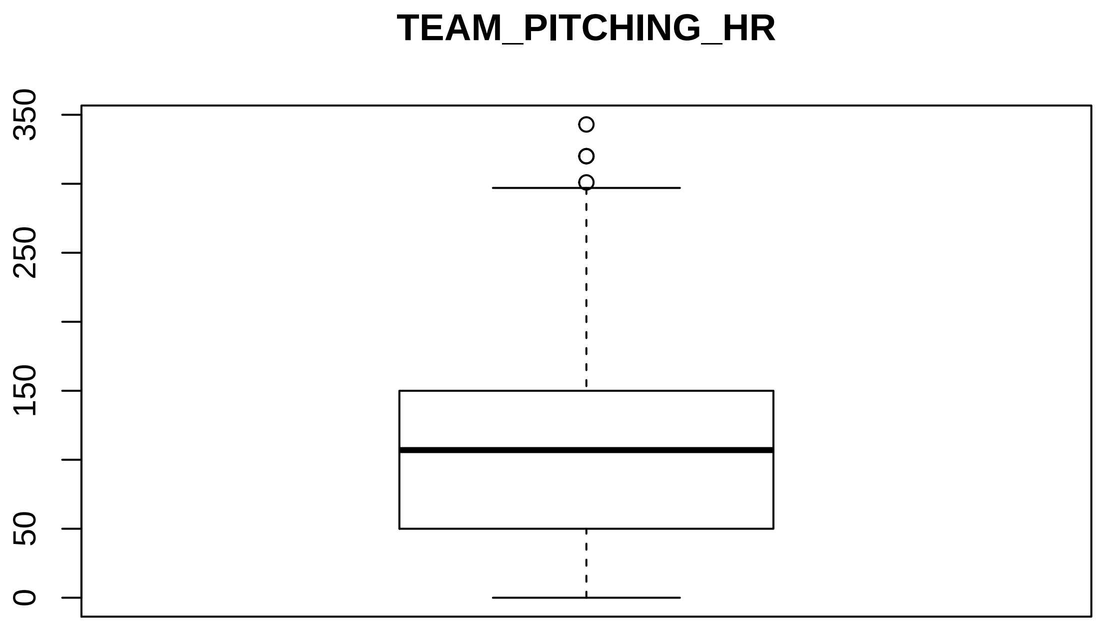

```
boxplot(moneyball2$TEAM_PITCHING_BB,main="TEAM_PITCHING_BB")
```



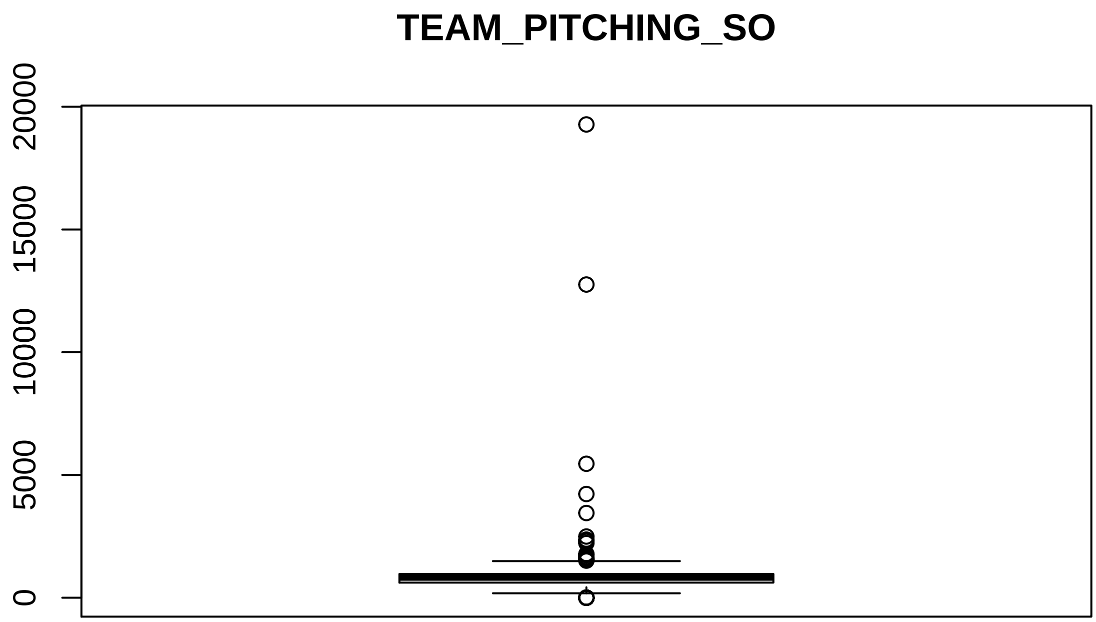

```r
boxplot(moneyball2$TEAM_PITCHING_H,main="TEAM_PITCHING_H")
```




```
boxplot(moneyball2$TEAM_PITCHING_HR,main="TEAM_PITCHING_HR")
```




```
boxplot(moneyball2$TEAM_PITCHING_SO,main="TEAM_PITCHING_SO")
```



For TEAM_BATTING_H, we can see that there are quite a few outliers, both at the upper and lower end. For this variable we decide to create a new variable that will have the outlier fixed.

For TEAM_BATTING_2B, we can see that there are quite a few outliers, both at the upper and a single outlier at the lower end. For this variable we decide to create a new variable that will have the outliers fixed.

For TEAM_BATTING_BB, we can see that there are quite a few outliers, both at the upper and lower end. For this variable we decide to create a new variable that will have the outlier fixed.

For TEAM_BATTING_SO, we can see that there are no outliers. No further action needed for this variable.

For TEAM_BASERUN_SB, we can see that there are quite a few outliers at the upper end. For this variable we decide to create a new variable that will have the outlier fixed.

For TEAM_FIELDING_E, we can see that there are quite a few outliers at the upper end. For this variable we decide to create a new variable that will have the outlier fixed.

For TEAM_FIELDING_DP, we can see that there are quite a few outliers, both at the upper and lower end. For this variable we decide to create a new variable that will have the outlier fixed.

For TEAM_PITCHING_BB, we can see that there are quite a few outliers, both at the upper and lower end. For this variable we decide to create a new variable that will have the outlier fixed.

For TEAM_PITCHING_H, we can see that there are quite a few outliers at the upper end. For this variable we decide to create a new variable that will have the outlier fixed.

For TEAM_PITCHING_HR, we can see that there only 3 outliers at the upper end. For this variable we decide to create a new variable that will have the outlier fixed.

For TEAM_PITCHING_SO, we can see that there are quite a few outliers at the upper and a single outlier on the lower end. For this variable we decide to create a new variable that will have the outlier fixed.

Data Preparation

Now that we have the preliminary analysis ready, we will go ahead and carry out the necessary transformations to the data.

This will primarily take care of Missing Values, Handle Outliers and create some additional variables.

Outliers

For outliers, we will use the capping method. In this method, we will replace all outliers that lie outside the 1.5 times of IQR limits. We will cap it by replacing those observations less than the lower limit with the value of 5th %ile and those that lie above the upper limit with the value of 95th %ile.

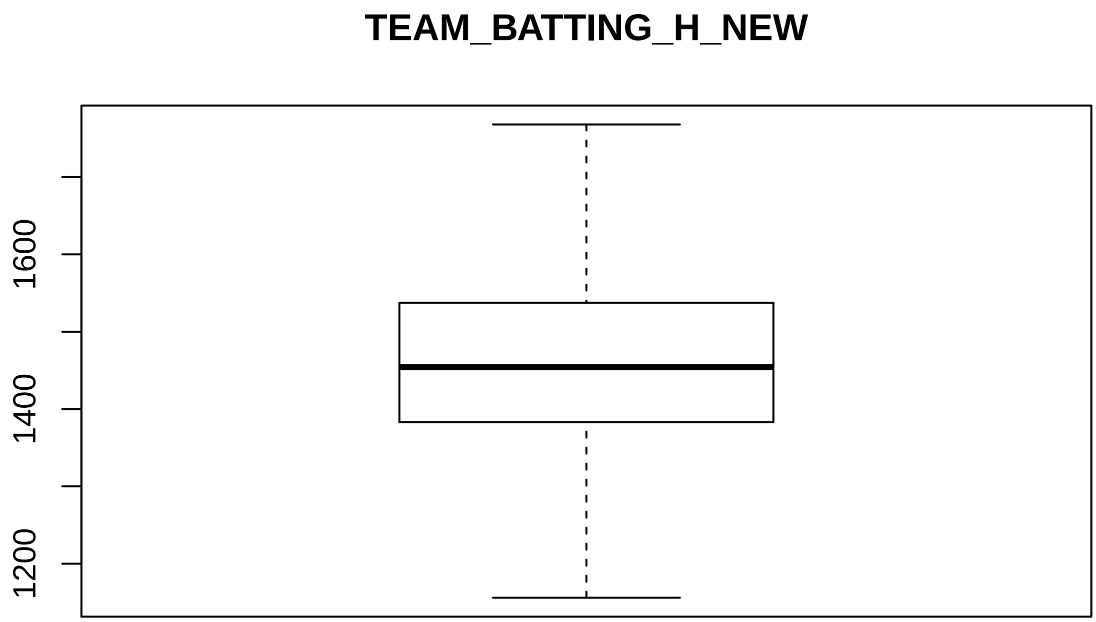
Accordingly we create the following new variables while retaining the original variables as is.

TEAM_BATTING_H TEAM_BATTING_2B TEAM_BATTING_BB TEAM_BASERUN_SB TEAM_FIELDING_E TEAM_FIELDING_DP TEAM_PITCHING_BB TEAM_PITCHING_H TEAM_PITCHING_HR TEAM_PITCHING_SO

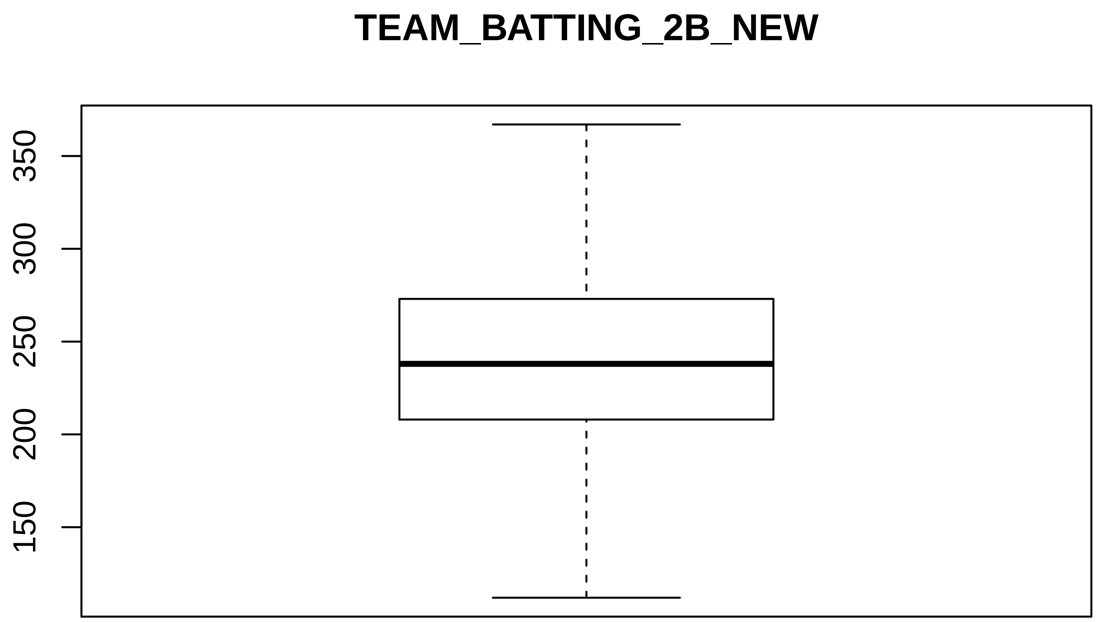
```
# function for removing outliers - http://r-statistics.co/Outlier-Treatment-With-R.html

treat_outliers <- function(x) {
qnt <- quantile(x, probs=c(.25, .75), na.rm = T)
caps <- quantile(x, probs=c(.05, .95), na.rm = T)
H <- 1.5 * IQR(x, na.rm = T)
x[x < (qnt[1] - H)] <- caps[1]
x[x > (qnt[2] + H)] <- caps[2]

return(x)
}

TEAM_BATTING_H_NEW <- treat_outliers(moneyball2$TEAM_BATTING_H)
TEAM_BATTING_2B_NEW <- treat_outliers(moneyball2$TEAM_BATTING_2B)
TEAM_BATTING_BB_NEW <- treat_outliers(moneyball2$TEAM_BATTING_BB)
TEAM_BASERUN_SB_NEW <- treat_outliers(moneyball2$TEAM_BASERUN_SB)
TEAM_FIELDING_E_NEW <- treat_outliers(moneyball2$TEAM_FIELDING_E)
TEAM_FIELDING_DP_NEW <- treat_outliers(moneyball2$TEAM_FIELDING_DP)
TEAM_PITCHING_BB_NEW <- treat_outliers(moneyball2$TEAM_PITCHING_BB)
TEAM_PITCHING_H_NEW <- treat_outliers(moneyball2$TEAM_PITCHING_H)
TEAM_PITCHING_HR_NEW <- treat_outliers(moneyball2$TEAM_PITCHING_HR)
TEAM_PITCHING_SO_NEW <- treat_outliers(moneyball2$TEAM_PITCHING_SO)
```

Lets see how the new variables look in boxplots.

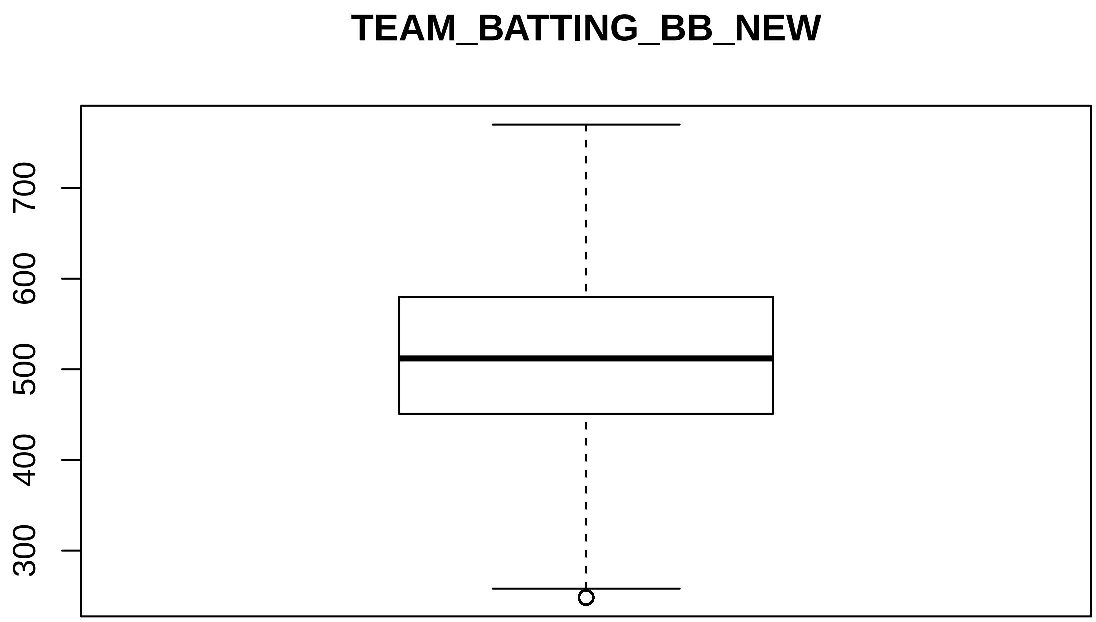
```
#par(mfrow=c(5,2))

boxplot(TEAM_BATTING_H_NEW,main="TEAM_BATTING_H_NEW")
```



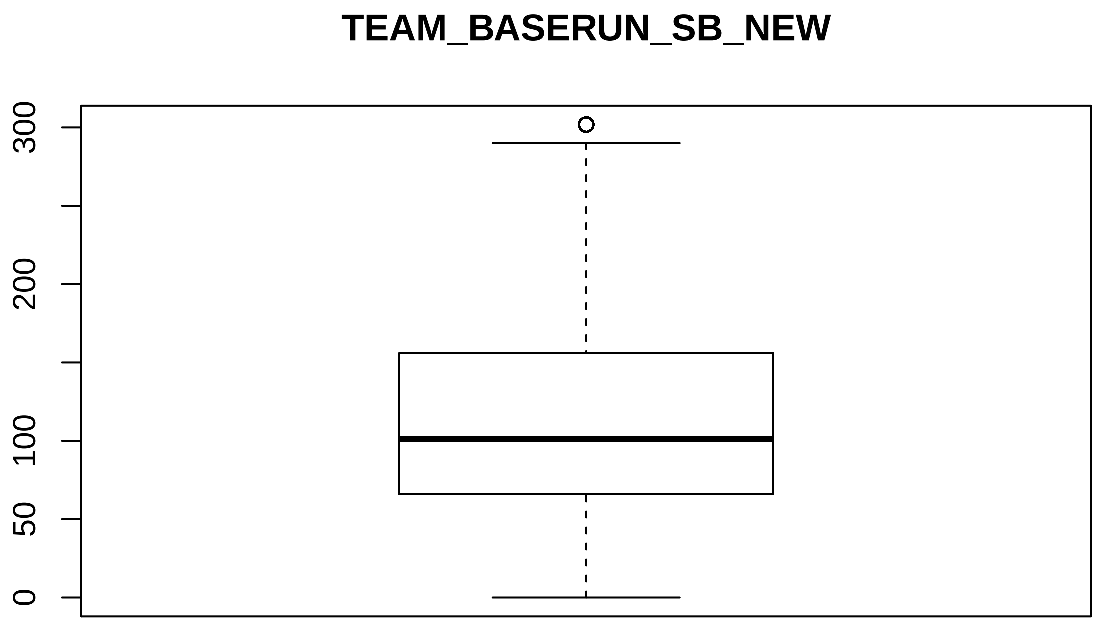

```
boxplot(TEAM_BATTING_2B_NEW,main="TEAM_BATTING_2B_NEW")
```



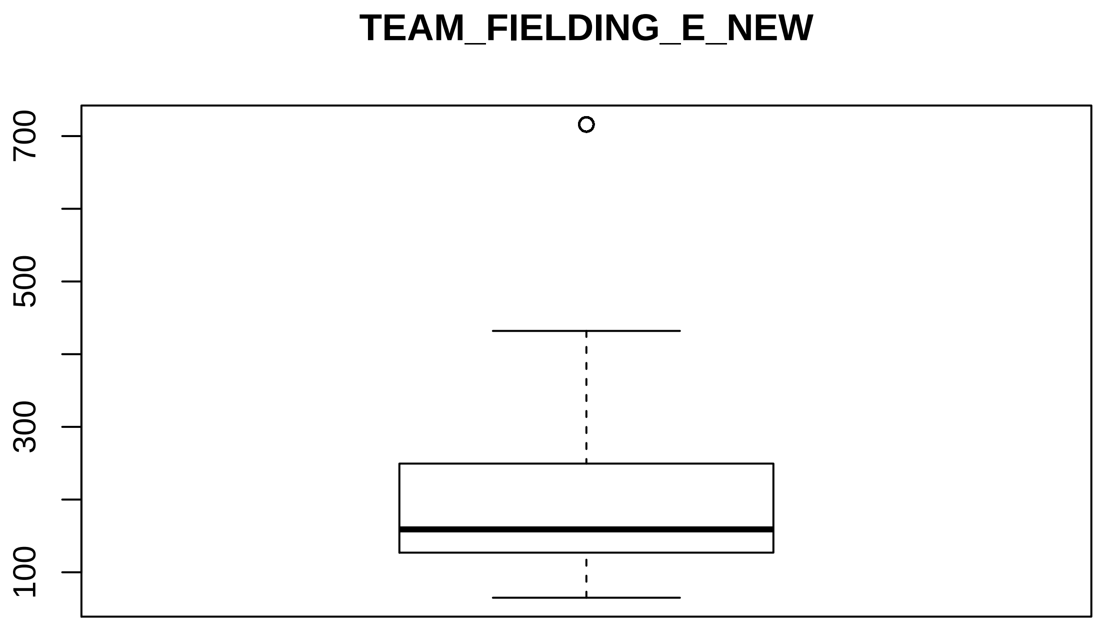

```r
boxplot(TEAM_BATTING_BB_NEW,main="TEAM_BATTING_BB_NEW")
```



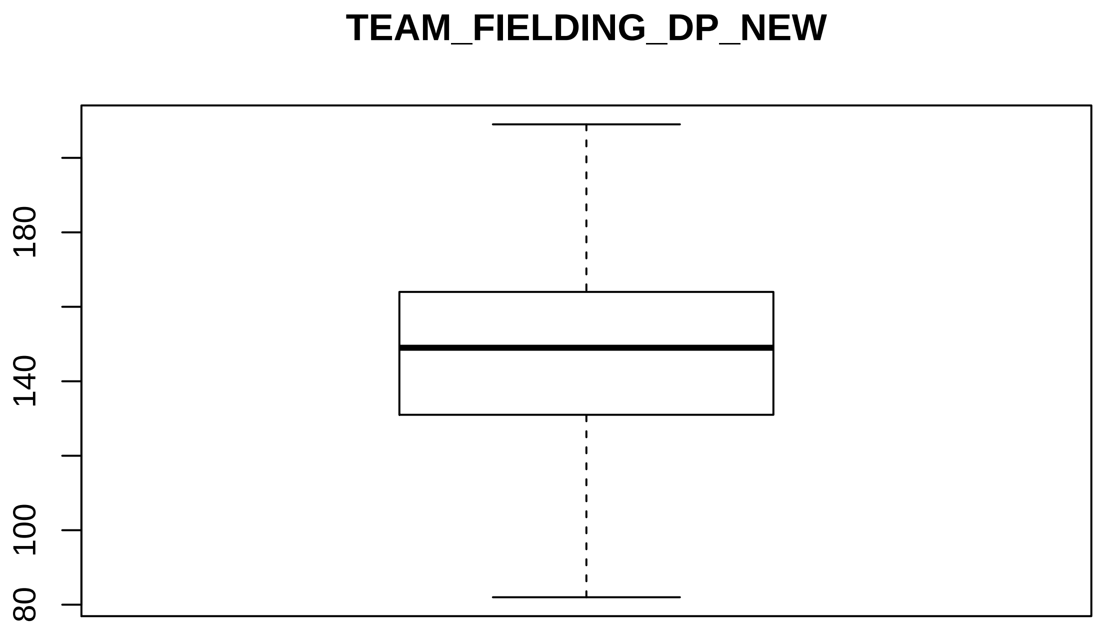

```
boxplot(TEAM_BASERUN_SB_NEW,main="TEAM_BASERUN_SB_NEW")
```



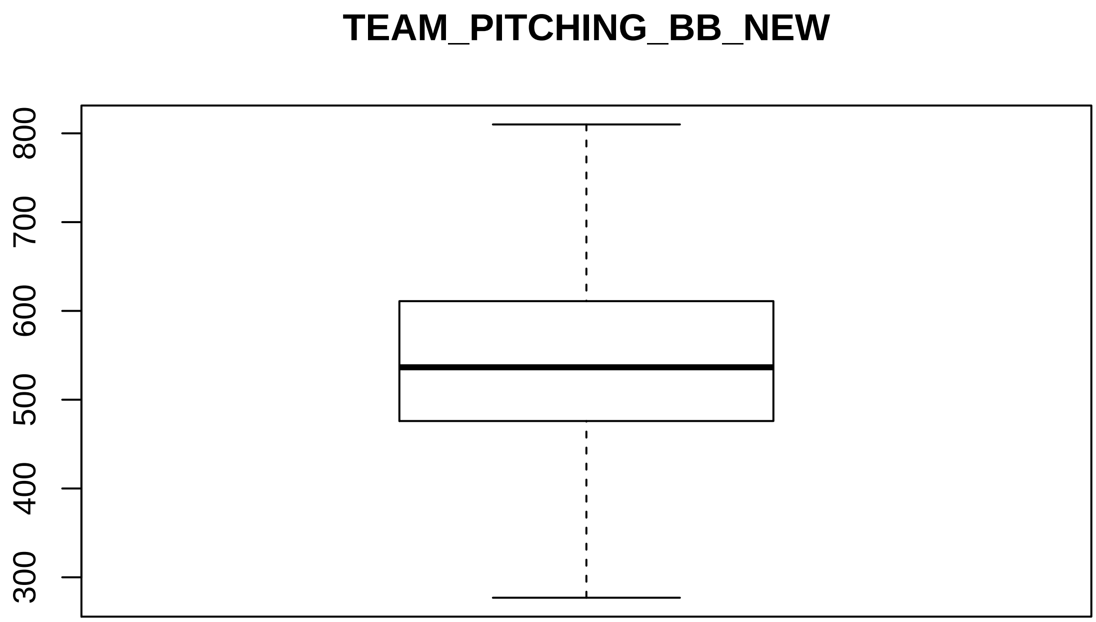

```
boxplot(TEAM_FIELDING_E_NEW,main="TEAM_FIELDING_E_NEW")
```



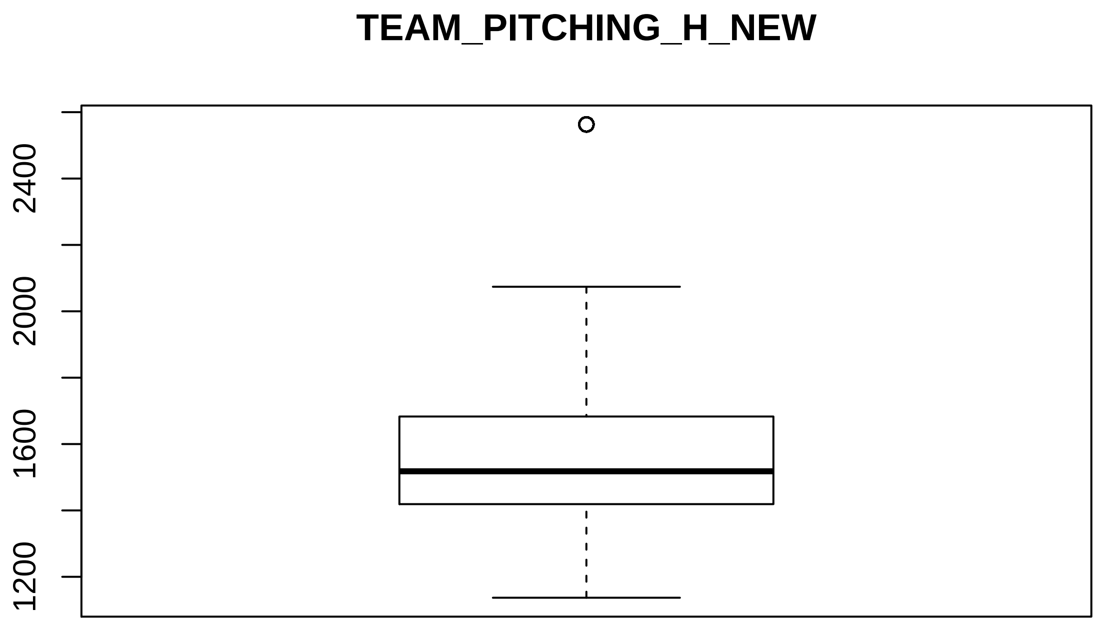

```r
boxplot(TEAM_FIELDING_DP_NEW,main="TEAM_FIELDING_DP_NEW")
```



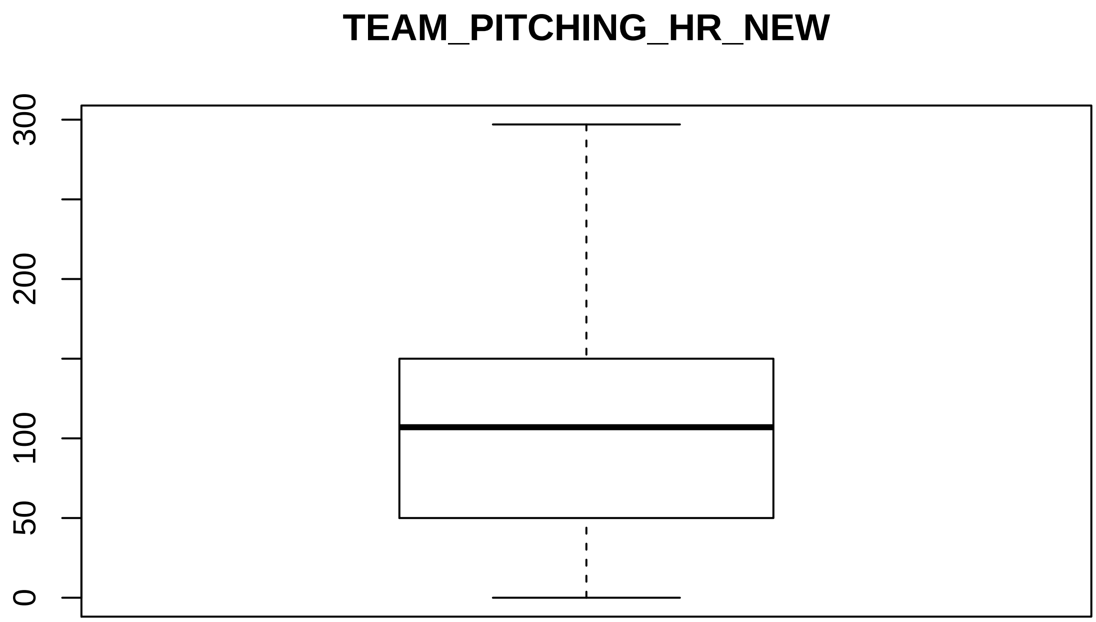

```
boxplot(TEAM_PITCHING_BB_NEW,main="TEAM_PITCHING_BB_NEW")
```



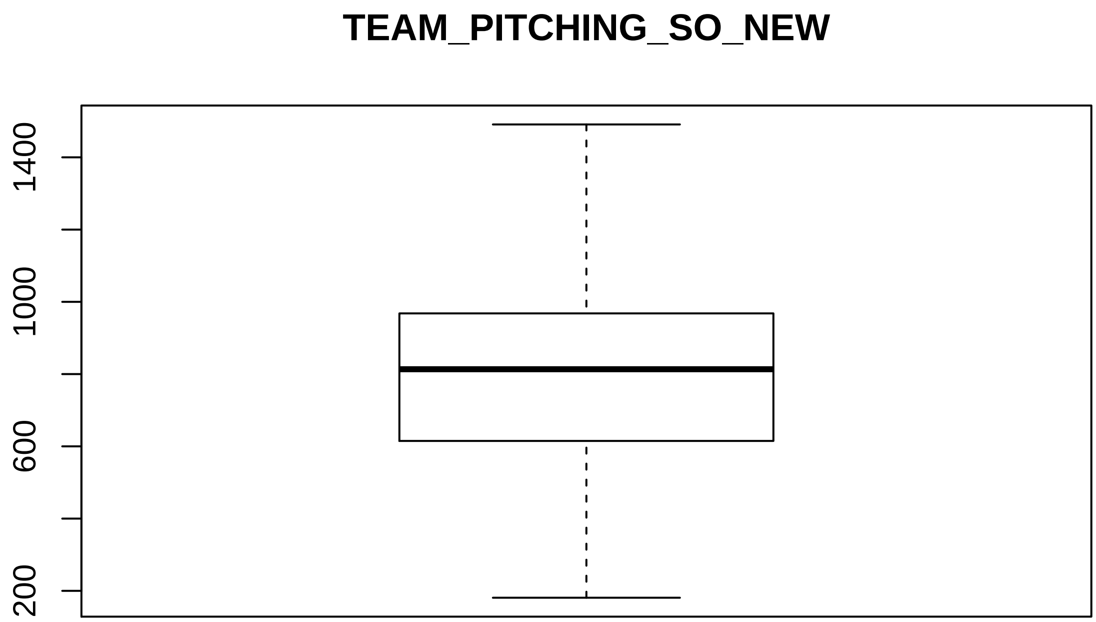

```r
boxplot(TEAM_PITCHING_H_NEW,main="TEAM_PITCHING_H_NEW")
```




```r
boxplot(TEAM_PITCHING_HR_NEW,main="TEAM_PITCHING_HR_NEW")
```




```
boxplot(TEAM_PITCHING_SO_NEW,main="TEAM_PITCHING_SO_NEW")
```



Missing Values

Next we impute missing values. Since we have handled outliers, we can go ahead and use the mean as impute values. As with outliers, we will go ahead and create new variables for the following:

TEAM_BATTING_SO

We will re-use the already created new variables for fixing the missing values for the below:

TEAM_PITCHING_SO TEAM_BASERUN_SB TEAM_FIELDING_DP

```
TEAM_BATTING_SO_NEW <- moneyball2$TEAM_BATTING_SO
TEAM_BATTING_SO_NEW[is.na(TEAM_BATTING_SO_NEW)] <- mean(TEAM_BATTING_SO_NEW, na.rm = T)

TEAM_PITCHING_SO_NEW[is.na(TEAM_PITCHING_SO_NEW)] <- mean(TEAM_PITCHING_SO_NEW, na.rm = T)
TEAM_BASERUN_SB_NEW[is.na(TEAM_BASERUN_SB_NEW)] <- mean(TEAM_BASERUN_SB_NEW, na.rm = T)
TEAM_FIELDING_DP_NEW[is.na(TEAM_FIELDING_DP_NEW)] <- mean(TEAM_FIELDING_DP_NEW, na.rm = T)
```

Additional Variables

Lets now create some additional variables that might help us in out analysis.

Missing Flags

First we create flag variables to indicate whether TEAM_BATTING_HBP and TEAM_BASERUN_CS and missing. If the value is missing, we code it with 1 and if the value is present we code it with 0.

```
TEAM_BATTING_HBP_Missing <- ifelse(complete.cases(moneyball2$TEAM_BATTING_HBP),1,0)
TEAM_BASERUN_CS_Missing <- ifelse(complete.cases(moneyball2$TEAM_BASERUN_CS),1,0)
```

Ratios

moneyball2$Hits_R < -moneyball$2TEAM_BATTING_H/moneyball2$TEAM_PITCHING_H moneyball$2Walks_R <- moneyball2$TEAM_BATTING_BB/moneyball$2TEAM_PITCHING_BB moneyball2$HomeRuns_R < -moneyball$2TEAM_BATTING_HR/moneyball2$TEAM_PITCHING_HR moneyball$2Strikeout_R <- moneyball2$TEAM_BATTING_SO/moneyball$2TEAM_PITCHING_SO

Build Models

Using the training data set, build at least three different multiple linear regression models, using different variables (or the same variables with different transformations). Since we have not yet covered automated variable selection methods, you should select the variables manually (unless you previously learned Forward or Stepwise selection, etc.). Since you manually selected a variable for inclusion into the model or exclusion into the model, indicate why this was done. Discuss the coefficients in the models, do they make sense? For example, if a team hits a lot of Home Runs, it would be reasonably expected that such a team would win more games. However, if the coefficient is negative (suggesting that the team would lose more games), then that needs to be discussed. Are you keeping the model even though it is counter intuitive? Why? The boss needs to know.

Select Models

Decide on the criteria for selecting the best multiple linear regression model. Will you select a model with slightly worse performance if it makes more sense or is more parsimonious? Discuss why you selected your model. For the multiple linear regression model, will you use a metric such as Adjusted R2, RMSE, etc.? Be sure to explain how you can make inferences from the model, discuss multi-collinearity issues (if any), and discuss other relevant model output. Using the training data set, evaluate the multiple linear regression model based on (a) mean squared error, (b) R2, (c) F-statistic, and (d) residual plots. Make predictions using the evaluation data set.

Model One with original data

```
library(car)
```

```
##
## Attaching package: 'car'

## The following object is masked from 'package:psych':
##
##     logit
```

```r
mod1<- lm(TARGET_WINS ~
     TEAM_BATTING_H +
     TEAM_BATTING_2B +
     TEAM_BATTING_3B +
     TEAM_BATTING_HR +
     TEAM_BATTING_BB +
     TEAM_BATTING_HBP +
     TEAM_BATTING_SO +
     TEAM_BASERUN_SB +
     TEAM_BASERUN_CS +
     TEAM_FIELDING_E +
     TEAM_FIELDING_DP +
     TEAM_PITCHING_BB +
     TEAM_PITCHING_H +
     TEAM_PITCHING_HR +
     TEAM_PITCHING_SO, moneyball2
   )

summary(mod1)
```

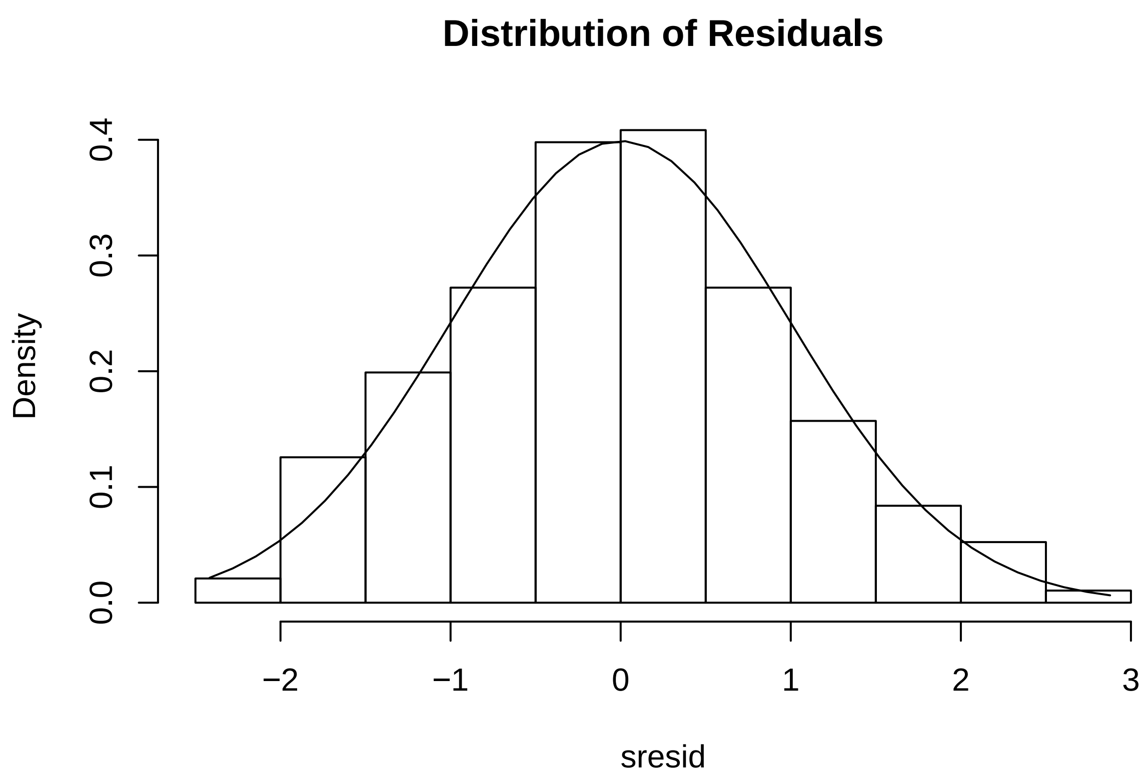
```
##
## Call:
## lm(formula = TARGET_WINS ~ TEAM_BATTING_H + TEAM_BATTING_2B +
##     TEAM_BATTING_3B + TEAM_BATTING_HR + TEAM_BATTING_BB + TEAM_BATTING_HBP +
##     TEAM_BATTING_SO + TEAM_BASERUN_SB + TEAM_BASERUN_CS + TEAM_FIELDING_E +
##     TEAM_FIELDING_DP + TEAM_PITCHING_BB + TEAM_PITCHING_H + TEAM_PITCHING_HR +
##     TEAM_PITCHING_SO, data = moneyball2)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -19.8708  -5.6564  -0.0599   5.2545  22.9274
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)      60.28826   19.67842   3.064  0.00253 **
## TEAM_BATTING_H    1.91348    2.76139   0.693  0.48927
## TEAM_BATTING_2B   0.02639    0.03029   0.871  0.38484
## TEAM_BATTING_3B  -0.10118    0.07751  -1.305  0.19348
## TEAM_BATTING_HR  -4.84371   10.50851  -0.461  0.64542
## TEAM_BATTING_BB  -4.45969    3.63624  -1.226  0.22167
## TEAM_BATTING_HBP  0.08247    0.04960   1.663  0.09815 .
## TEAM_BATTING_SO   0.34196    2.59876   0.132  0.89546
## TEAM_BASERUN_SB   0.03304    0.02867   1.152  0.25071
## TEAM_BASERUN_CS  -0.01104    0.07143  -0.155  0.87730
## TEAM_FIELDING_E  -0.17204    0.04140  -4.155 5.08e-05 ***
## TEAM_FIELDING_DP -0.10819    0.03654  -2.961  0.00349 **
## TEAM_PITCHING_BB  4.51089    3.63372   1.241  0.21612
## TEAM_PITCHING_H  -1.89096    2.76095  -0.685  0.49432
## TEAM_PITCHING_HR  4.93043   10.50664   0.469  0.63946
## TEAM_PITCHING_SO -0.37364    2.59705  -0.144  0.88577
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## Residual standard error: 8.467 on 175 degrees of freedom
##   (2085 observations deleted due to missingness)
## Multiple R-squared:  0.5501, Adjusted R-squared:  0.5116
## F-statistic: 14.27 on 15 and 175 DF,  p-value: < 2.2e-16
```

```r
#library(faraway)
#sumary(mod1)
```

Normality check of Residuals

```r
#  First let plot residuals to see if they look like a normal distribution:

library(MASS)
sresid <- studres(mod1)
hist(sresid, freq=FALSE,
     main="Distribution of Residuals")
xfit<-seq(min(sresid),max(sresid),length=40)
yfit<-dnorm(xfit)
lines(xfit, yfit)
```



The residuals are normally distributed, this indicates That the mean of the difference between our predictions

and the actual values is close to 0 which is good for our analysis.

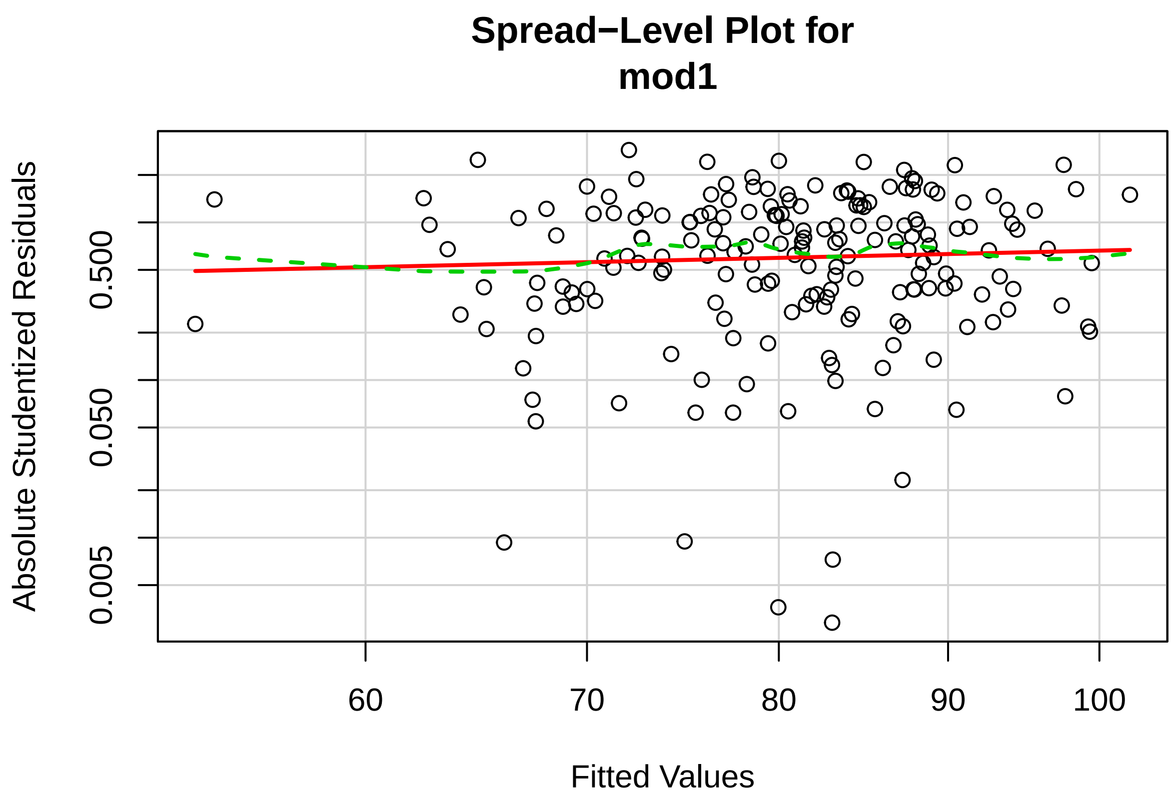
Also, it's unlikely that no relationship exists between TEAM_FIELDING_E and TARGET_WINS.

homoscedasticity check or non-constant error variance test

```
# Evaluate homoscedasticity
ncvTest(mod1)
```

```
## Non-constant Variance Score Test
## Variance formula: ~ fitted.values
## Chisquare = 0.03994848    Df = 1     p = 0.8415813
```

```
# plot studentized residuals vs. fitted values
spreadLevelPlot(mod1)
```




```
##
## Suggested power transformation:  0.5267026
```

The test confirms the non-constant error variance test. It also has a p-value higher than a significance level of 0.05.

Therefore we can accept the null hypothesis that the variance of the residuals is constant and infer that heteroscedasticity is not present.

Collinearity Check

```
# Evaluate Collinearity
vif(mod1) # variance inflation factors
```

```
##    TEAM_BATTING_H  TEAM_BATTING_2B  TEAM_BATTING_3B  TEAM_BATTING_HR
##      1.171824e+05     1.685623e+00     1.302198e+00     3.074804e+05
##   TEAM_BATTING_BB TEAM_BATTING_HBP  TEAM_BATTING_SO  TEAM_BASERUN_SB
##      1.962853e+05     1.096334e+00     1.941752e+05     1.950069e+00
##   TEAM_BASERUN_CS  TEAM_FIELDING_E TEAM_FIELDING_DP TEAM_PITCHING_BB
##      1.914415e+00     1.256819e+00     1.097611e+00     1.964039e+05
##   TEAM_PITCHING_H TEAM_PITCHING_HR TEAM_PITCHING_SO
##      1.160417e+05     3.069624e+05     1.946316e+05
```

```
Collinearity<- sqrt(vif(mod1)) > 3 # 3 problem?
data.frame(Collinearity)
```

```
##                  Collinearity
## TEAM_BATTING_H           TRUE
## TEAM_BATTING_2B         FALSE
## TEAM_BATTING_3B         FALSE
## TEAM_BATTING_HR          TRUE
## TEAM_BATTING_BB          TRUE
## TEAM_BATTING_HBP        FALSE
## TEAM_BATTING_SO          TRUE
## TEAM_BASERUN_SB         FALSE
## TEAM_BASERUN_CS         FALSE
## TEAM_FIELDING_E         FALSE
## TEAM_FIELDING_DP        FALSE
## TEAM_PITCHING_BB         TRUE
## TEAM_PITCHING_H          TRUE
## TEAM_PITCHING_HR         TRUE
## TEAM_PITCHING_SO         TRUE
```

Test for Autocorrelated Errors

durbinWatsonTest(mod1)

```
durbinWatsonTest(mod1)
```

```
##  lag Autocorrelation D-W Statistic p-value
##    1       0.2128921      1.567453       0
##  Alternative hypothesis: rho != 0
```

goodness of fit of your model

using R-squared and adjusted R-squared, our model is about 55% predicts the TARGET_WINS