Data 621 Assignment 1

Mark Gonsalves, Joshua Hummell, Claire Meyer, Chinedu Onyeka, Rathish Parayil Sasidharan

3/6/2022

Assignment One: Baseball

1. Data Exploration

This assignment reviews a baseball dataset, which looks at teams across a number of different features. We'll be building a number of linear regression models and comparing their efficacy.

To start, we will explore the dataset: its shape, composition, and any information that may help with future data processing and model building.

```
#eval <- getURL("https://raw.githubusercontent.com/cmm6/data608/main/moneyball-evaluation-data.csv",.op
#train <- getURL("https://raw.githubusercontent.com/cmm6/data608/main/moneyball-training-data.csv",.opt
baseball_eval <- read.csv("https://raw.githubusercontent.com/cmm6/data608/main/moneyball-evaluation-data
baseball_training <- read.csv("https://raw.githubusercontent.com/cmm6/data608/main/moneyball-training-da

baseball_eval <- subset(baseball_eval)
baseball_training = subset(baseball_training, select = -c(INDEX) )

print(dim(baseball_training))
```

```
## [1] 2276   16
```

```
print(head(baseball_training))
```

```
##   TARGET_WINS TEAM_BATTING_H TEAM_BATTING_2B TEAM_BATTING_3B TEAM_BATTING_HR
## 1          39           1445             194              39              13
## 2          70           1339             219              22             190
## 3          86           1377             232              35             137
## 4          70           1387             209              38              96
## 5          82           1297             186              27             102
## 6          75           1279             200              36              92
##   TEAM_BATTING_BB TEAM_BATTING_SO TEAM_BASERUN_SB TEAM_BASERUN_CS
## 1             143             842              NA              NA
## 2             685            1075              37              28
## 3             602             917              46              27
## 4             451             922              43              30
## 5             472             920              49              39
## 6             443             973             107              59
##   TEAM_BATTING_HBP TEAM_PITCHING_H TEAM_PITCHING_HR TEAM_PITCHING_BB
## 1               NA            9364               84              927
```

```
## 2               NA            1347              191              689
## 3               NA            1377              137              602
## 4               NA            1396               97              454
## 5               NA            1297              102              472
## 6               NA            1279               92              443
##   TEAM_PITCHING_SO TEAM_FIELDING_E TEAM_FIELDING_DP
## 1             5456            1011               NA
## 2             1082             193              155
## 3              917             175              153
## 4              928             164              156
## 5              920             138              168
## 6              973             123              149
```

We've dropped the index, which the data dictionary confirms is irrelevant to the target variable. As such, the evaluation set has 2276 observations, with 16 columns. The first, `Target_Wins` is the target of future linear regression modeling.

First, we'll use summary() to get a sense of the type and values for each field.

Right away, we can see several fields have NA values, including the majority of TEAM_BATTING_HBP. We can try different ways to handle these in later model development, e.g. removing entirely vs. replacing with mean, etc.

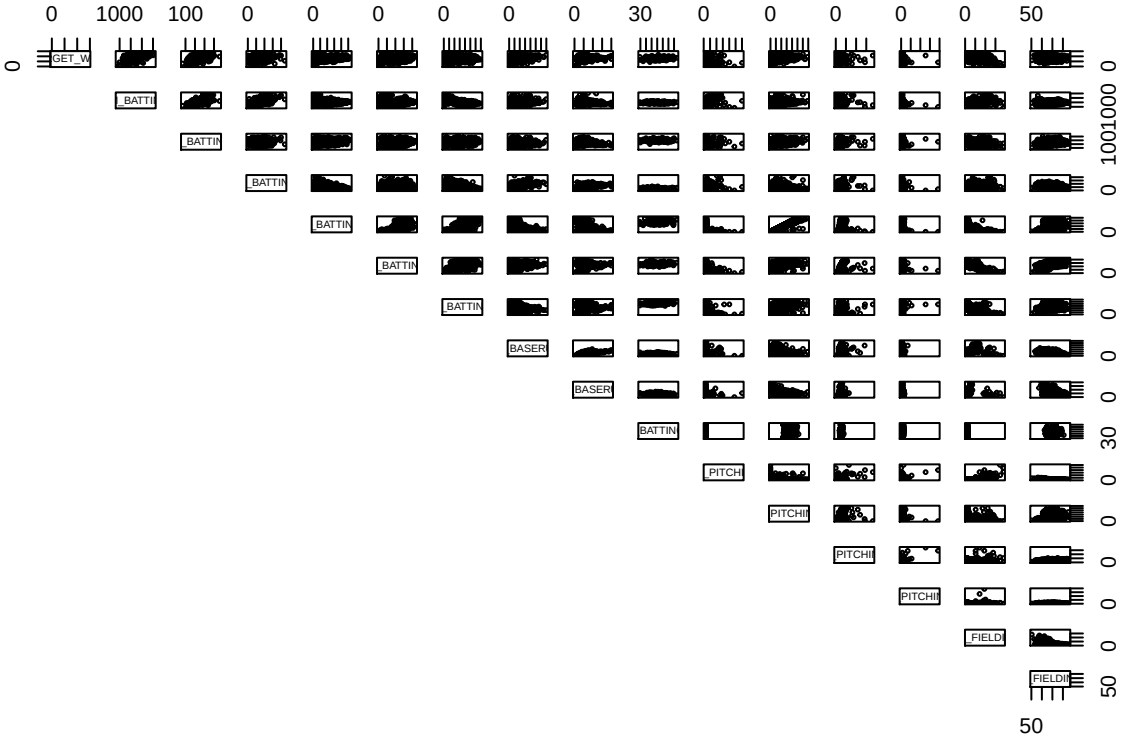
```
summary(baseball_training)
```

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

```
##  3rd Qu.: 611.0   3rd Qu.:  968.0   3rd Qu.: 249.2   3rd Qu.:164.0
##  Max.   :3645.0   Max.   :19278.0   Max.   :1898.0   Max.   :228.0
##                   NA's   :102                        NA's   :286
```

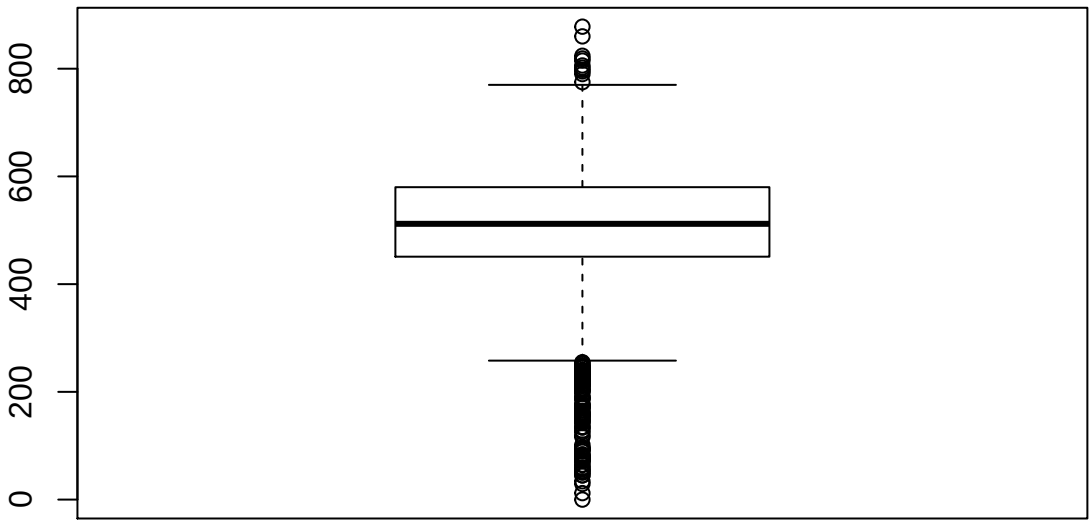
One of the key requirements of linear regression is a linear relationship between the explanatory and target variables. Digging into these relationships using scatter plot, it looks like TEAM_BATTING_SO, TEAM_BATTING_BB, and TEAM_PITCHING_SO have clear clustering around 0 and may not be linear.

```
pairs(baseball_training, lower.panel = NULL, cex = 0.4, cex.labels=0.5)
```

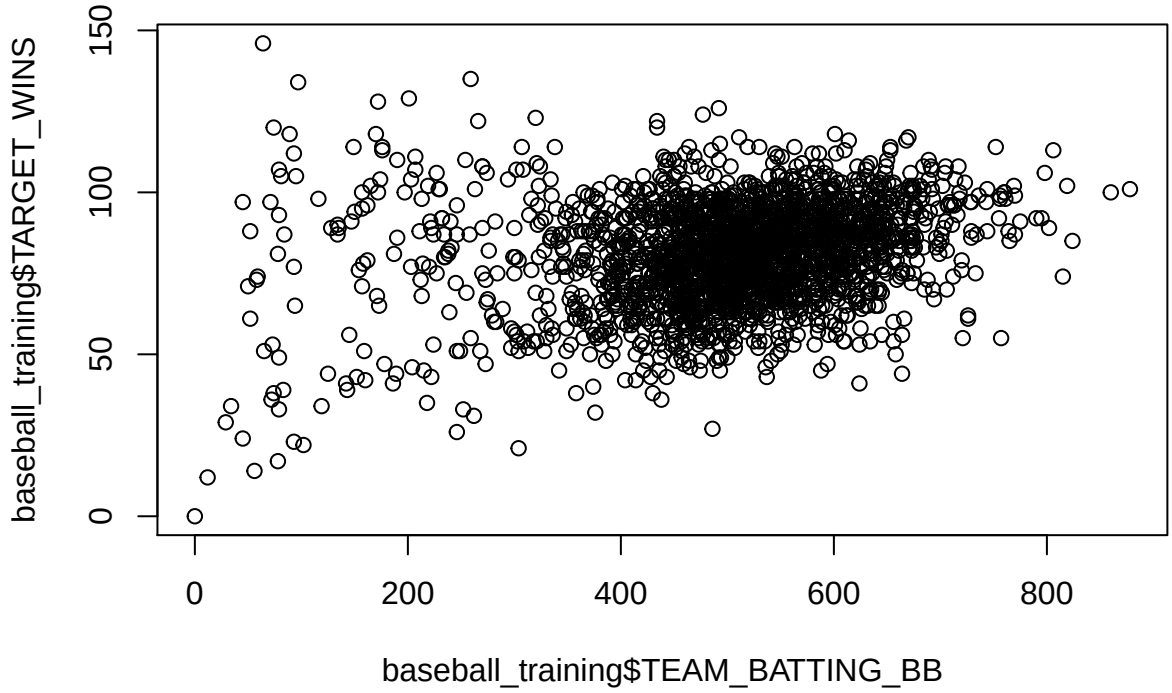


Digging further in with scatterplots versus the target, as well as box plots, there is clear clustering of some values around 0 for both TEAM_BATTING_BB and TEAM_BATTING_SO, and nearly all TEAM_PITCHING_SO values are 0. The boxplot for the latter shows a few outliers, but otherwise the majority of the data tightly clustered at low value.

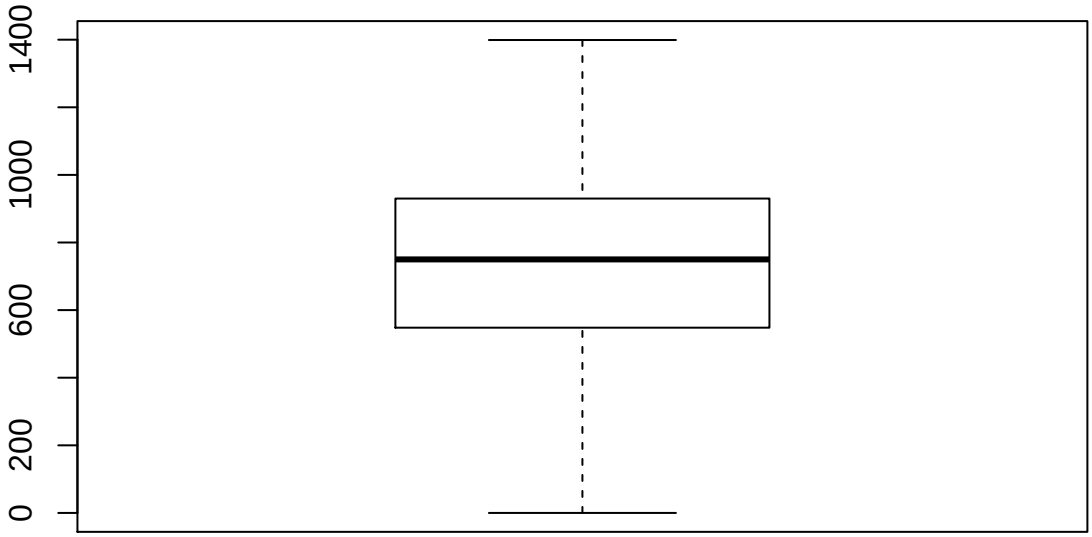
```
boxplot(baseball_training$TEAM_BATTING_BB)
```



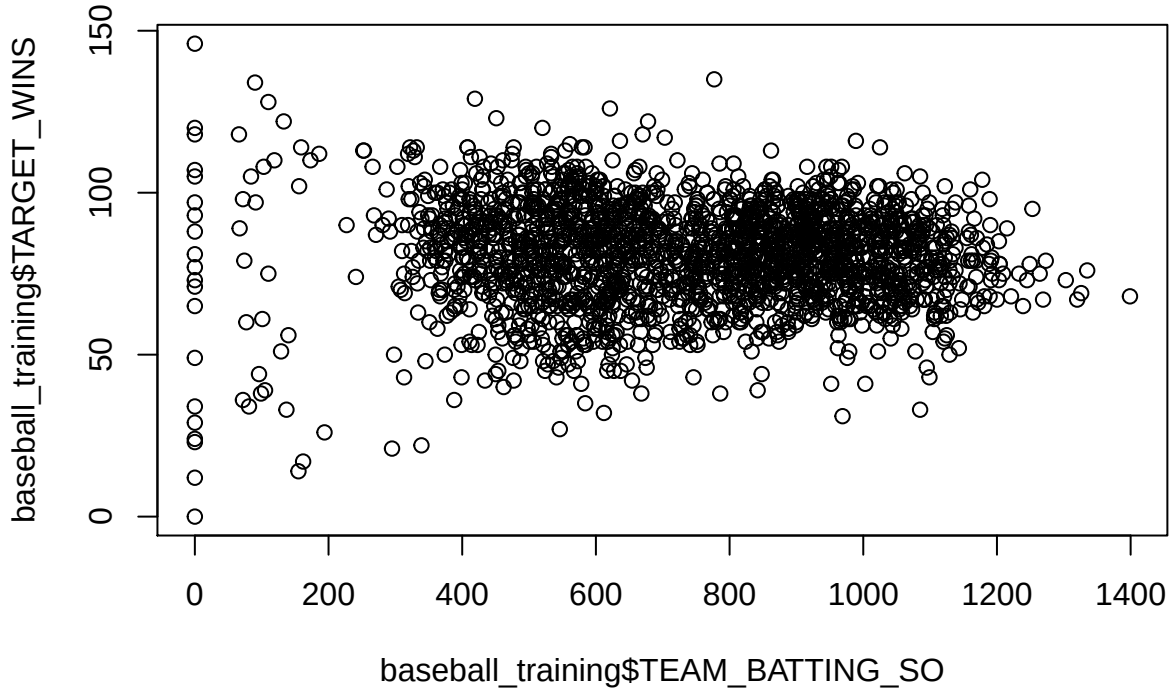

```
plot(baseball_training$TEAM_BATTING_BB,baseball_training$TARGET_WINS)
```



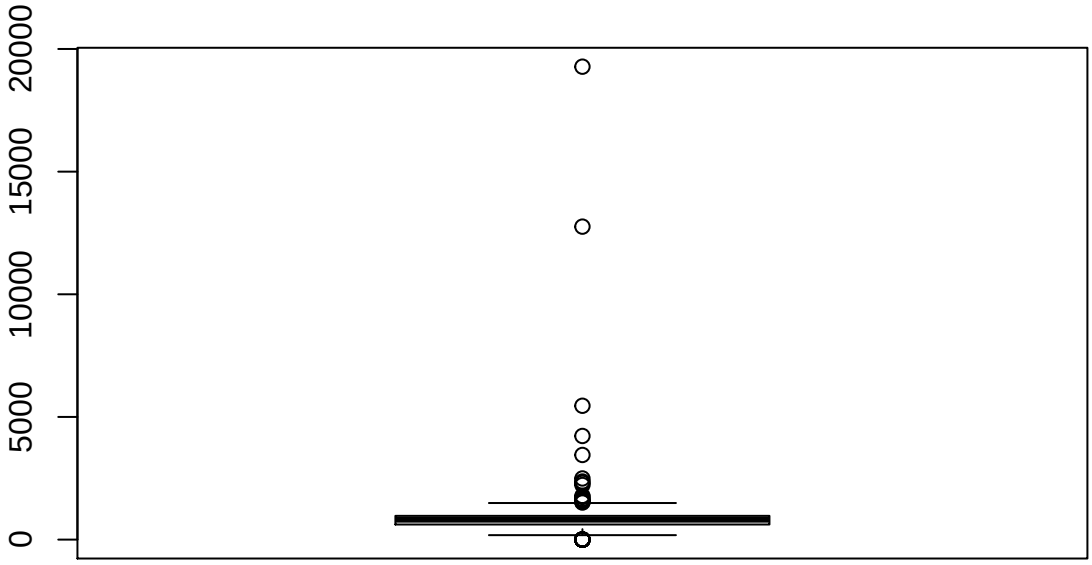

```
boxplot(baseball_training$TEAM_BATTING_SO)
```



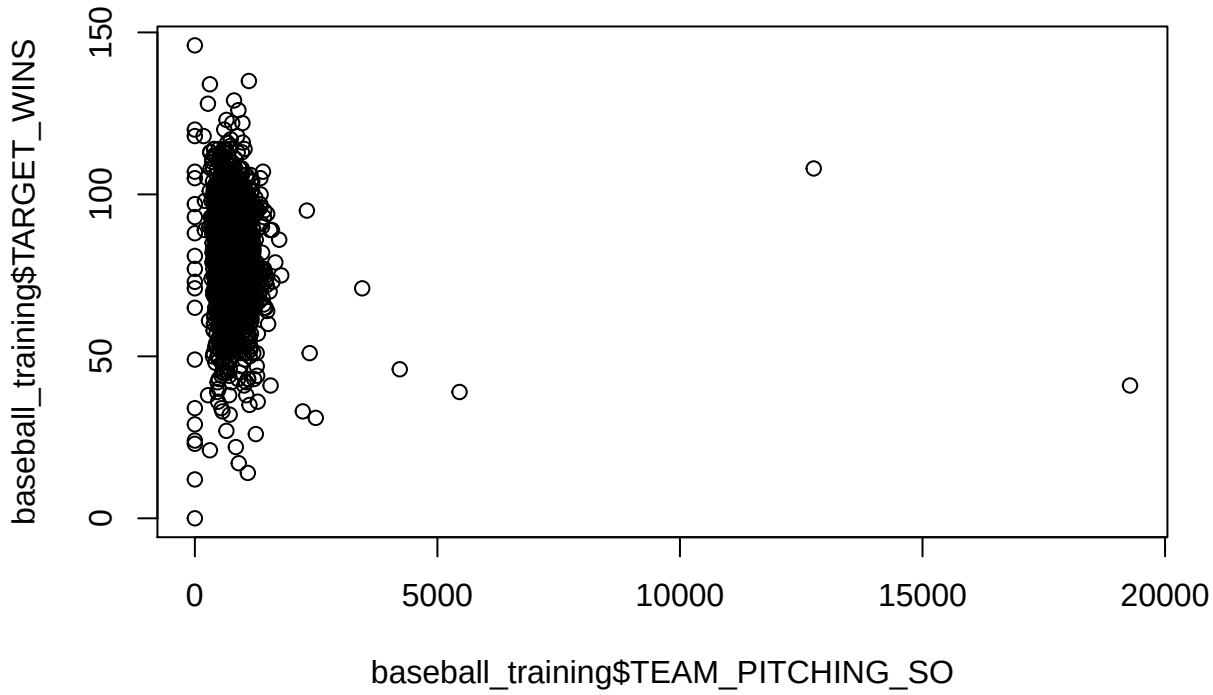

```
plot(baseball_training$TEAM_BATTING_SO,baseball_training$TARGET_WINS)
```




```
boxplot(baseball_training$TEAM_PITCHING_SO)
```



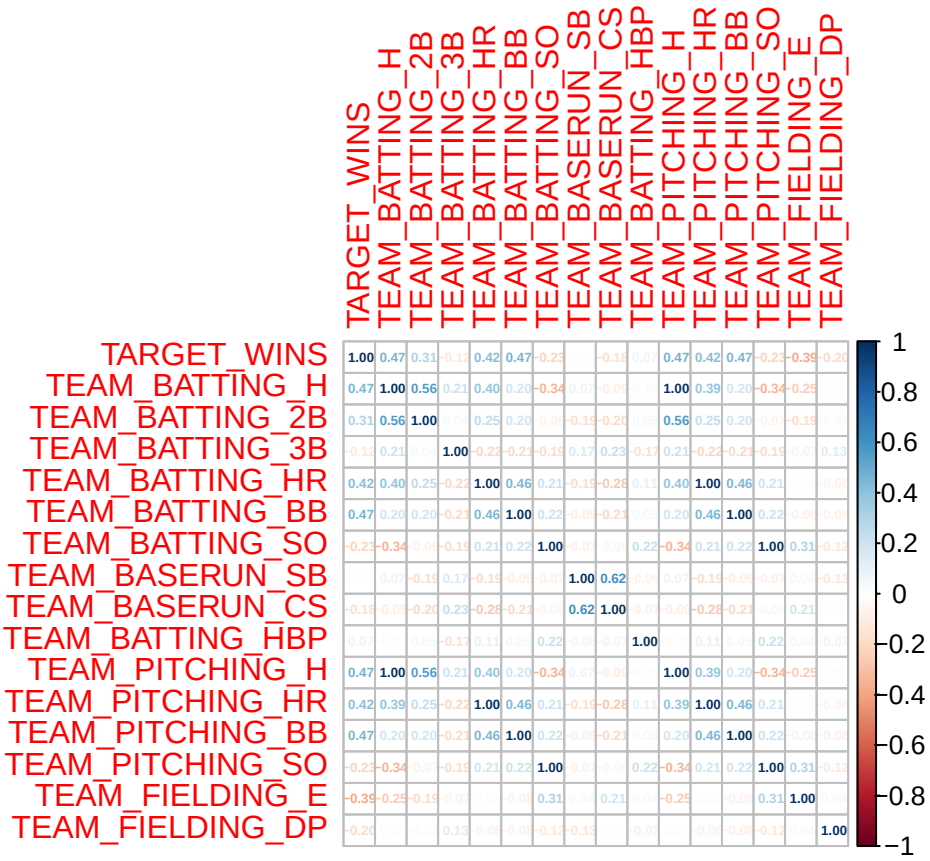

```
plot(baseball_training$TEAM_PITCHING_SO,baseball_training$TARGET_WINS)
```



These features and other nonlinearity can be kept in mind in executing the model.

Also valuable is building a correlation matrix. This serves two purposes - to show which features correlate highly to the Target variable, and to reduce potential for collinearity if two features are not offering distinct value to the model. NA values are omitted.

```
training_cor <- cor(na.omit(baseball_training))
corrplot(training_cor, method = 'number',number.cex=7/ncol(baseball_training))
```



In terms of collinearity, there is large correlation between: * TEAM_PITCHING_H and TEAM_BATTING_H * TEAM_PITCHING_HR and TEAM_BATTING_HR * TEAM_PITCHING_BB and TEAM_BATTING_BB * TEAM_PITCHING_SO and TEAM_BATTING_SO

In the final model, we can explore keeping just 1 of each of these pairs.

In terms of correlation to the TARGET_WINS, TEAM_BATTING_H, TEAM_BATTING_BB, TEAM_PITCHING_H, and TEAM_PITCHING_BB have highest positive correlation. These are also fields with mutual correlation, which is helpful to note going into the model development and data preparation.

Finally, we can create a baseline model, that takes every variable unadjusted. This can act as a baseline to outperform as we iterate with more elaborate models.

```
lm_baseline <- lm(TARGET_WINS~.,baseball_training)
summary(lm_baseline)
```

```
##
## Call:
## lm(formula = TARGET_WINS ~ ., data = baseball_training)
##
## 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_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_BATTING_HBP  0.08247    0.04960   1.663  0.09815 .
## TEAM_PITCHING_H  -1.89096    2.76095  -0.685  0.49432
## TEAM_PITCHING_HR  4.93043   10.50664   0.469  0.63946
## TEAM_PITCHING_BB  4.51089    3.63372   1.241  0.21612
## TEAM_PITCHING_SO -0.37364    2.59705  -0.144  0.88577
## TEAM_FIELDING_E  -0.17204    0.04140  -4.155 5.08e-05 ***
## TEAM_FIELDING_DP -0.10819    0.03654  -2.961  0.00349 **
## ---
## 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
```

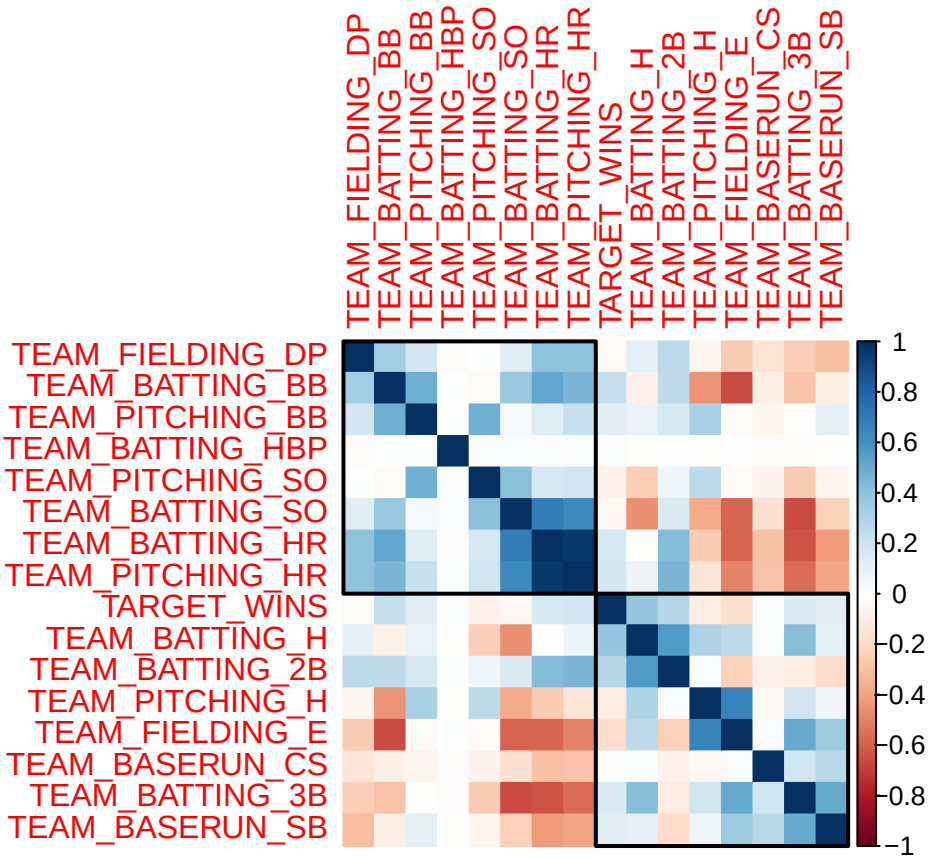
2. Data Preparation

For Data preparation, there are two things we want to focus on, imputing missing values and making sure the data is ready for the models. The first item we can tackle is finding and replacing missing values. First, we want to find the percentage of missing values. The largest is TEAM_BATTING_HBP with 91% missing data while the second largest variables is TEAM_BASERUN_CS with 33% missing. We also have TEAM_FIELDING_DP with 12% missing, TEAM_BASERUN_SB with roughly 5% and both TEAM_PITCHING_SO and TEAM_BATTING_SO with 4.5% each.

For Team Batting HBP, I was originally going to fill in the NAs with 0 since there is a limited chance that it occurs and seems more like a data input error (the min is 29 but if it is not common should be 0), But since it is an issue that relates to how the games are recorded (this was not recoreded in the early days of baseball) we will get the mean data instead. And for TEAM_FIELDING_DP, I will do the same since it seems to have the same error. For all the other NAs it would make sense to fill in with the mean since there is not too much missing data or the min is 0.0.

Now, another thing we want to check for is if there are any issues with collinearity.

```
training_cor <- cor(na.omit(baseball_training))
corrplot(training_cor,method = 'color' ,order = 'hclust', addrect = 2)
```



From this heatmap we can see that there is a pairwise relationship between TEAM_PITCHING_HR and TEAM_BATTING_HR (close to 1), but since they are integral to our data, we will not get rid of either.

Finally, I will add several new columns, One for Predicted runs for season, Team Fielding, and Team Pitching

We need to do the same for baseball eval

And now, we are ready to split the data and build the model.

```
set.seed(678)

split <- sample.split(baseball_training$TARGET_WINS, SplitRatio = 0.8)
training_set <- subset(baseball_training, split == TRUE)
test_set <- subset(baseball_training, split == FALSE)
```

We split the data into train (80%) and test data (20%)

3. Build Models

Model 1

```
lm2 <- lm(TARGET_WINS ~ TEAM_BATTING_H + TEAM_BATTING_2B + TEAM_BATTING_3B +
           TEAM_BATTING_HR + TEAM_BATTING_BB + TEAM_BATTING_SO +
           TEAM_BASERUN_SB +  TEAM_PITCHING_H +
            TEAM_PITCHING_BB + TEAM_PITCHING_SO +
           TEAM_FIELDING_E + TEAM_FIELDING_DP, data = training_set)
```

```
summary(lm2)
```

```
##
## Call:
## lm(formula = TARGET_WINS ~ TEAM_BATTING_H + TEAM_BATTING_2B +
##     TEAM_BATTING_3B + TEAM_BATTING_HR + TEAM_BATTING_BB + TEAM_BATTING_SO +
##     TEAM_BASERUN_SB + TEAM_PITCHING_H + TEAM_PITCHING_BB + TEAM_PITCHING_SO +
##     TEAM_FIELDING_E + TEAM_FIELDING_DP, data = training_set)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -49.627  -8.472   0.049   8.515  56.544
##
## Coefficients:
##                     Estimate Std. Error t value Pr(>|t|)
## (Intercept)      22.2410579  5.9344238   3.748 0.000184 ***
## TEAM_BATTING_H    0.0490852  0.0041243  11.902  < 2e-16 ***
## TEAM_BATTING_2B  -0.0155517  0.0103227  -1.507 0.132100
## TEAM_BATTING_3B   0.0658366  0.0183529   3.587 0.000343 ***
## TEAM_BATTING_HR   0.0568688  0.0110121   5.164 2.68e-07 ***
## TEAM_BATTING_BB   0.0075975  0.0056821   1.337 0.181357
## TEAM_BATTING_SO  -0.0062301  0.0028160  -2.212 0.027066 *
## TEAM_BASERUN_SB   0.0244755  0.0048265   5.071 4.36e-07 ***
## TEAM_PITCHING_H  -0.0009467  0.0003790  -2.498 0.012590 *
## TEAM_PITCHING_BB  0.0023327  0.0036950   0.631 0.527908
## TEAM_PITCHING_SO  0.0025050  0.0008933   2.804 0.005100 **
## TEAM_FIELDING_E  -0.0199673  0.0026315  -7.588 5.17e-14 ***
## TEAM_FIELDING_DP -0.1234703  0.0146783  -8.412  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 13.23 on 1813 degrees of freedom
## Multiple R-squared:  0.3232, Adjusted R-squared:  0.3187
## F-statistic: 72.15 on 12 and 1813 DF,  p-value: < 2.2e-16
```

```
vif(lm2)
```

```
##   TEAM_BATTING_H  TEAM_BATTING_2B  TEAM_BATTING_3B  TEAM_BATTING_HR
##         3.780657         2.473079         2.888639         4.667396
##  TEAM_BATTING_BB  TEAM_BATTING_SO  TEAM_BASERUN_SB  TEAM_PITCHING_H
##         5.480579         5.016713         1.804216         3.561706
## TEAM_PITCHING_BB TEAM_PITCHING_SO  TEAM_FIELDING_E TEAM_FIELDING_DP
##         4.540903         2.903665         4.161496         1.333660
```

The most common way to detect multicollinearity is by using the variance inflation factor (VIF), which measures the correlation and strength of correlation between the predictor variables in a regression model.

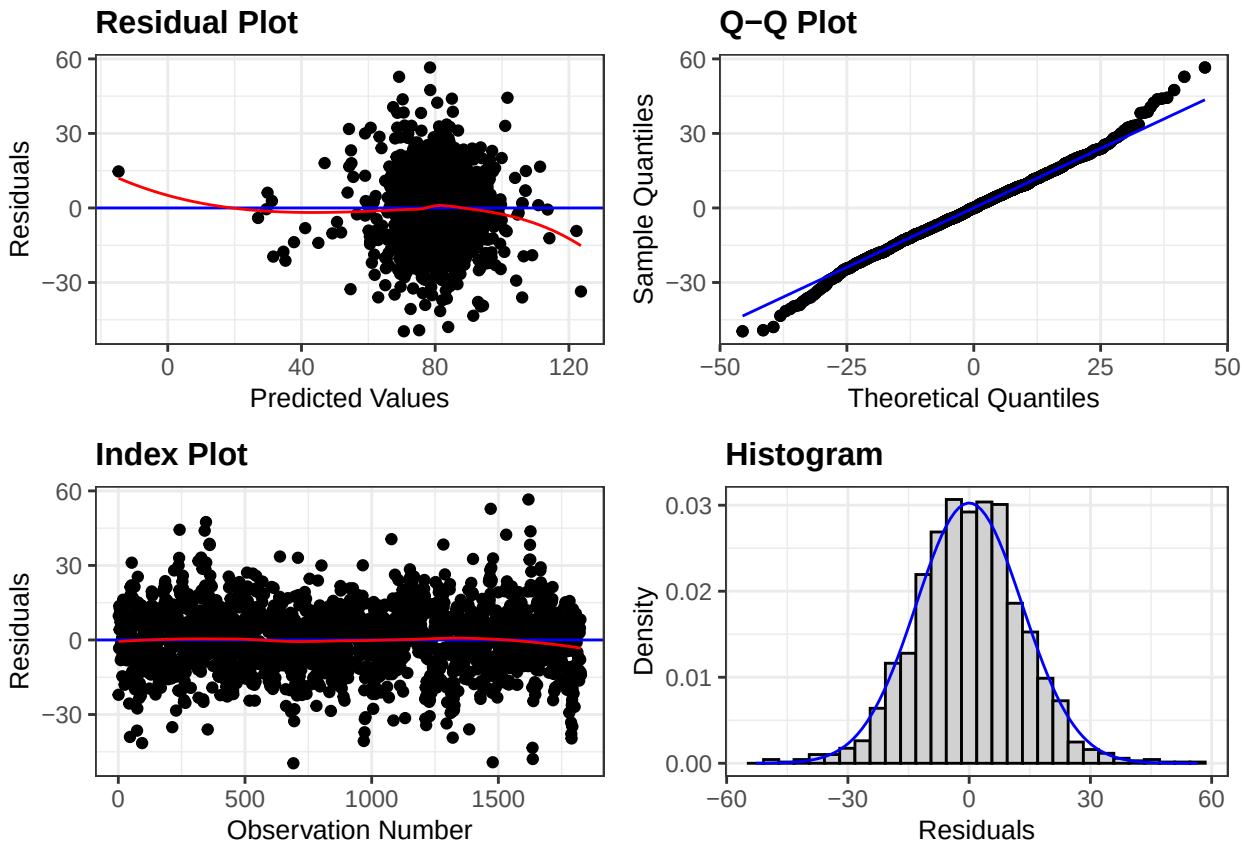
A value greater than 5 indicates potentially severe correlation between a given predictor variable and other predictor variables in the model.

From our earlier analysis , we already noticed that a correlation exist between TEAM_PITCHING_H * TEAM_BATTING_H and TEAM_BATTING_BB * TEAM_PITCHING_BB and TEAM_PITCHING_SO * TEAM_BATTING_SO

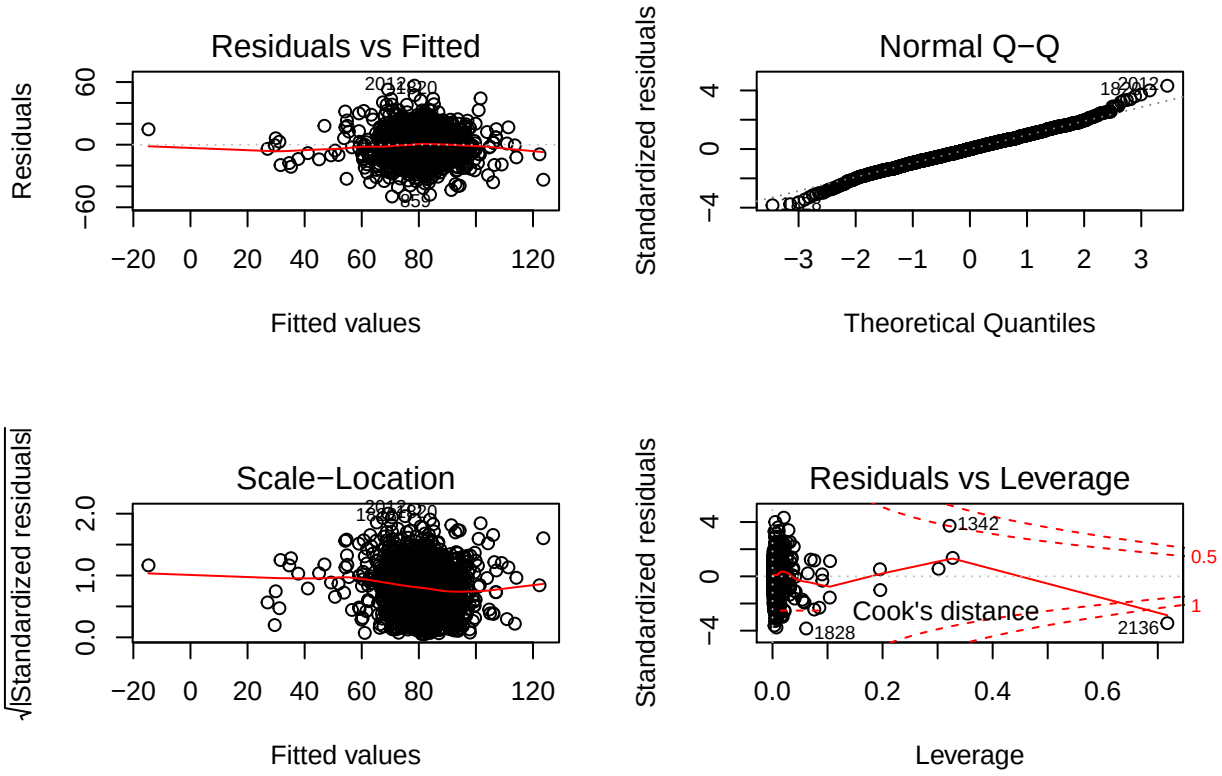
Model Diagnostics

```
resid_panel(lm2, plots='default', smoother = TRUE)
```

```
## `geom_smooth()` using formula 'y ~ x'
## `geom_smooth()` using formula 'y ~ x'
```




```
par(mfrow=c(2,2))
plot(lm2)
```



Model 2

In this model we are going to remove the correlated predictors

```
lm3 <- lm(TARGET_WINS ~ TEAM_BATTING_H + TEAM_BATTING_2B + TEAM_BATTING_3B +
            TEAM_BATTING_HR + TEAM_BATTING_BB + TEAM_BATTING_SO +
            TEAM_BASERUN_SB + TEAM_FIELDING_E, data = training_set)


summary(lm3)
```

```
##
## Call:
## lm(formula = TARGET_WINS ~ TEAM_BATTING_H + TEAM_BATTING_2B +
##     TEAM_BATTING_3B + TEAM_BATTING_HR + TEAM_BATTING_BB + TEAM_BATTING_SO +
##     TEAM_BASERUN_SB + TEAM_FIELDING_E, data = training_set)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -53.078  -9.154   0.002   8.508  60.798
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)      8.0105200  5.7444624   1.394  0.16334
## TEAM_BATTING_H   0.0460224  0.0041572  11.071  < 2e-16 ***
## TEAM_BATTING_2B -0.0134959  0.0103968  -1.298  0.19442
## TEAM_BATTING_3B  0.0813442  0.0183731   4.427 1.01e-05 ***
```

```
## TEAM_BATTING_HR  0.0352787  0.0109966   3.208  0.00136 **
## TEAM_BATTING_BB  0.0033577  0.0037405   0.898  0.36949
## TEAM_BATTING_SO  0.0005925  0.0025871   0.229  0.81889
## TEAM_BASERUN_SB  0.0323915  0.0047867   6.767 1.77e-11 ***
## TEAM_FIELDING_E -0.0229460  0.0021954 -10.452  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 13.53 on 1817 degrees of freedom
## Multiple R-squared:  0.2909, Adjusted R-squared:  0.2877
## F-statistic: 93.15 on 8 and 1817 DF,  p-value: < 2.2e-16
```

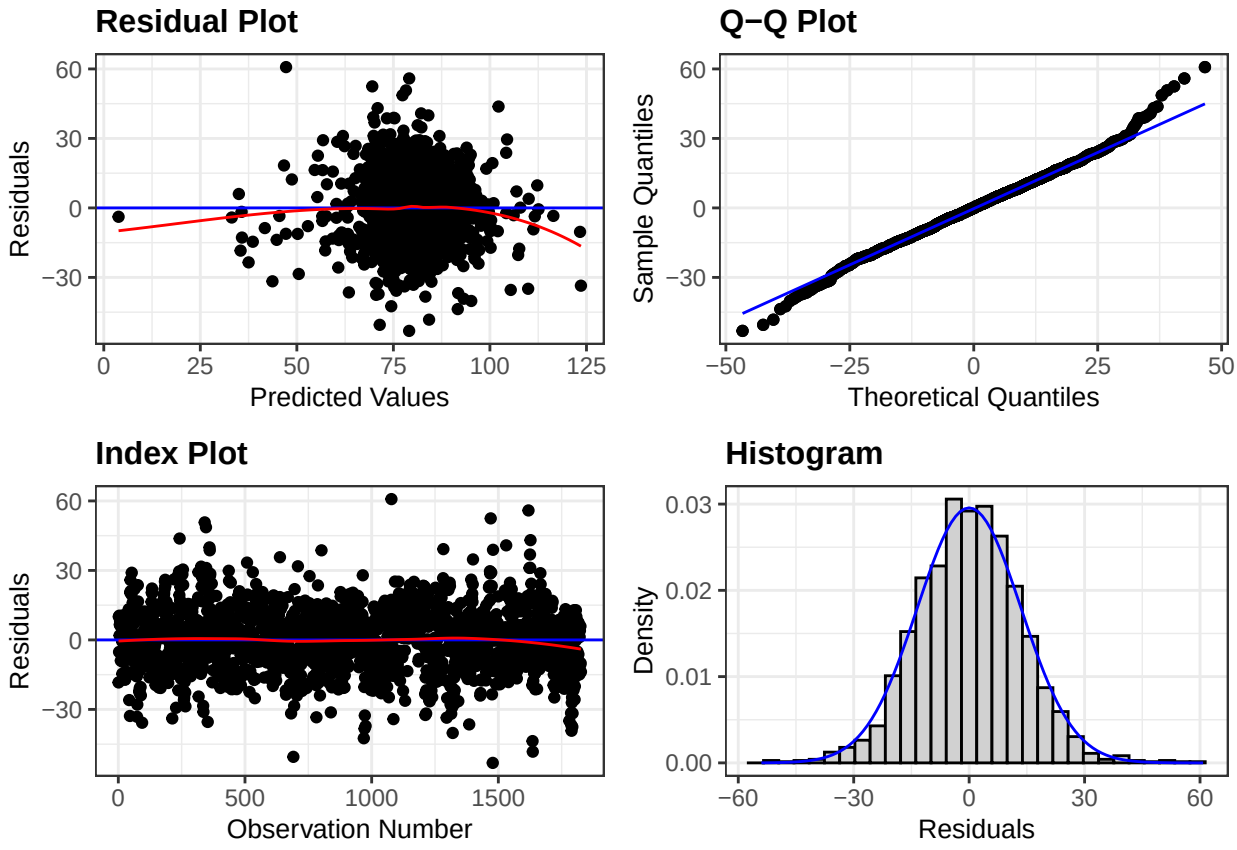
```
vif(lm3)
```

```
##  TEAM_BATTING_H TEAM_BATTING_2B TEAM_BATTING_3B TEAM_BATTING_HR TEAM_BATTING_BB
##        3.674111        2.399495        2.769016        4.451733        2.271714
## TEAM_BATTING_SO TEAM_BASERUN_SB TEAM_FIELDING_E
##        4.050009        1.697371        2.770376
```

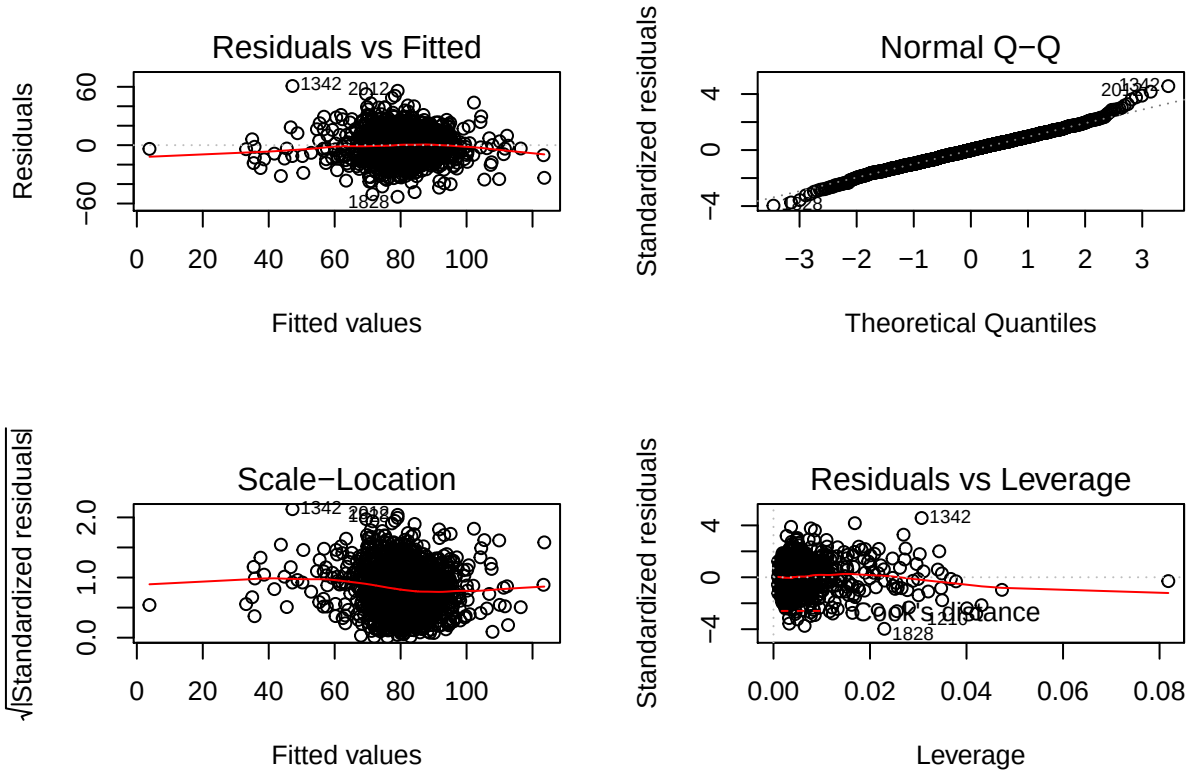
Model Diagnostics

```
resid_panel(lm3, plots='default', smoother = TRUE)
```

```
## `geom_smooth()` using formula 'y ~ x'
## `geom_smooth()` using formula 'y ~ x'
```



```
par(mfrow=c(2,2))
plot(lm3)
```



Model 3

We will do further clean up based on p-value

```
lm4 <- lm(TARGET_WINS ~ TEAM_BATTING_H + TEAM_BATTING_2B + TEAM_BATTING_3B +
           TEAM_BATTING_HR +  TEAM_BASERUN_SB + TEAM_FIELDING_E, data = training_set)


summary(lm4)
```

```
##
## Call:
## lm(formula = TARGET_WINS ~ TEAM_BATTING_H + TEAM_BATTING_2B +
##     TEAM_BATTING_3B + TEAM_BATTING_HR + TEAM_BASERUN_SB + TEAM_FIELDING_E,
##     data = training_set)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -52.943  -9.046  -0.028   8.581  60.883
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)     10.376668   3.450126   3.008  0.00267 **
## TEAM_BATTING_H   0.045480   0.003481  13.067  < 2e-16 ***
```

```
## TEAM_BATTING_2B -0.012592   0.010003  -1.259  0.20827
## TEAM_BATTING_3B  0.082751   0.018220   4.542 5.94e-06 ***
## TEAM_BATTING_HR  0.038935   0.008380   4.647 3.62e-06 ***
## TEAM_BASERUN_SB  0.033813   0.004331   7.806 9.86e-15 ***
## TEAM_FIELDING_E -0.024096   0.001771 -13.603  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 13.52 on 1819 degrees of freedom
## Multiple R-squared:  0.2905, Adjusted R-squared:  0.2882
## F-statistic: 124.2 on 6 and 1819 DF,  p-value: < 2.2e-16
```

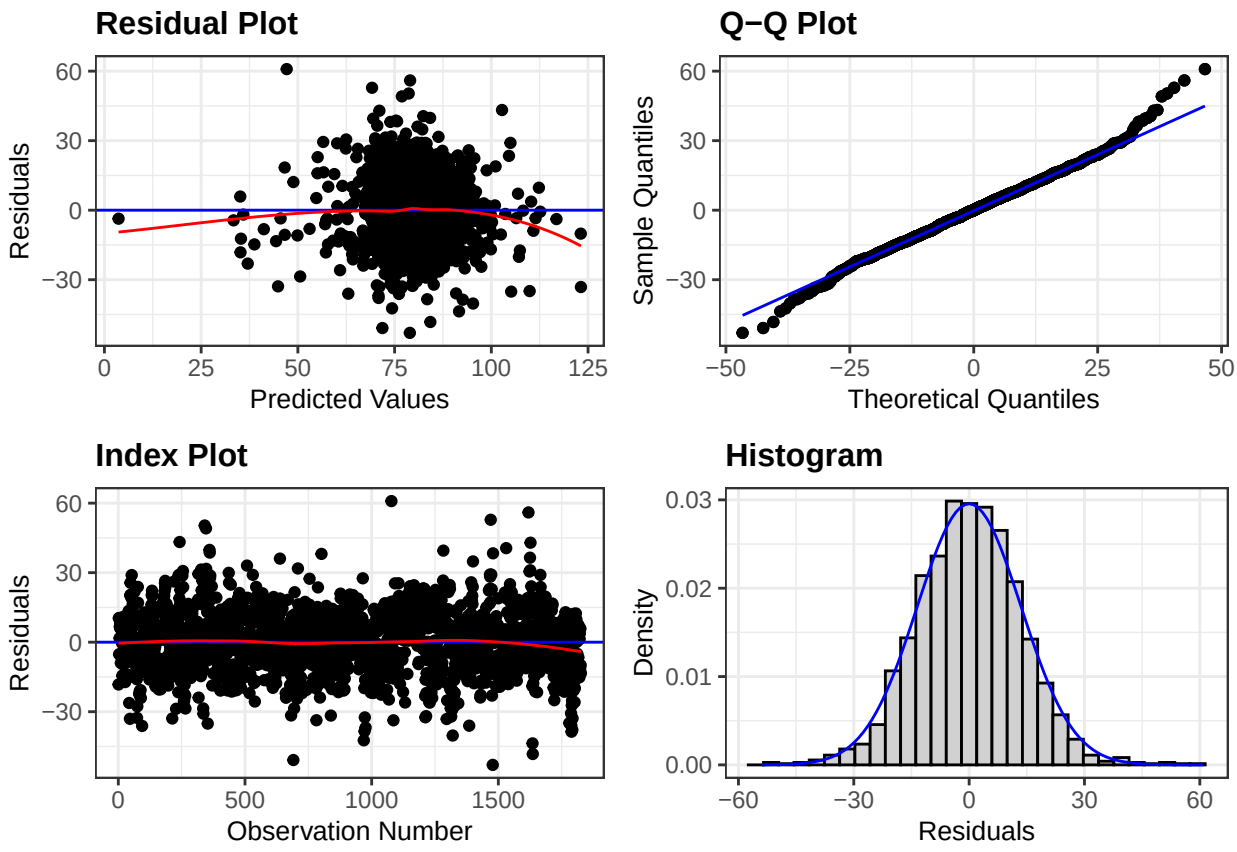
```
vif(lm4)
```

```
##  TEAM_BATTING_H TEAM_BATTING_2B TEAM_BATTING_3B TEAM_BATTING_HR TEAM_BASERUN_SB
##        2.577105        2.222817        2.724804        2.586646        1.390762
## TEAM_FIELDING_E
##        1.804790
```

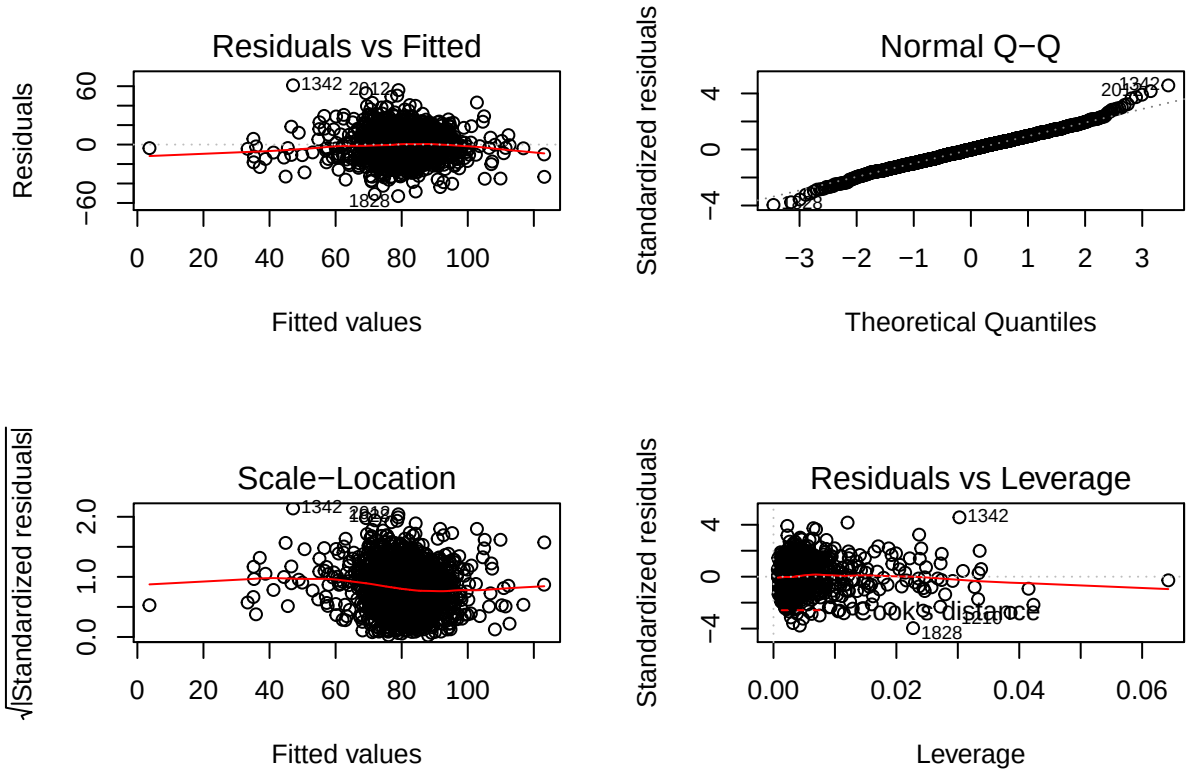
Model Diagnostics

```
resid_panel(lm4, plots='default', smoother = TRUE)
```

```
## `geom_smooth()` using formula 'y ~ x'
## `geom_smooth()` using formula 'y ~ x'
```



```
par(mfrow=c(2,2))
plot(lm4)
```



4. Select Models

In order to select the best model, we will look at the evaluation metrics (RSE, R-Squared, Adj. R-Squared, F-Statistic, and AIC) for all three models and compare them.

Extract Model Evaluation Metrics

Model 1

```
# extract the rse, r.squared, adj.r.squared, F-statistic, and AIC for model 1
model1_rse <- round(summary(lm2)$sigma, 4)
model1_r_squared <- round(summary(lm2)$r.squared, 4)
model1_adj_r_squared <- round(summary(lm2)$adj.r.squared, 4)
model1_f_statistic <- round(summary(lm2)$fstatistic[1], 4)
model1_aic <- round(AIC(lm2), 4)

model1_metrics <- c(model1_rse, model1_r_squared, model1_adj_r_squared, model1_f_statistic,
                    model1_aic)
```

Model 2

```
# extract the rse, r.squared, adj.r.squared, F-statistic, and AIC for model 2
model2_rse <- round(summary(lm3)$sigma, 4)
model2_r_squared <- round(summary(lm3)$r.squared, 4)
model2_adj_r_squared <- round(summary(lm3)$adj.r.squared, 4)
```

	Model1	Model2	Model3
RSE	13.2276	13.5252	13.5207
RSquared	0.3232	0.2909	0.2905
Adj-RSquared	0.3187	0.2877	0.2882
F-Statistic	72.1506	93.1534	124.1515
AIC	14627.4980	14704.7697	14701.5802

```
model2_f_statistic <- round(summary(lm3)$fstatistic[1], 4)
model2_aic <- round(AIC(lm3), 4)

model2_metrics <- c(model2_rse, model2_r_squared, model2_adj_r_squared, model2_f_statistic,
                    model2_aic)
```

Model 3

```
# extract the rse, r.squared, adj.r.squared, F-statistic, and AIC for model 3
model3_rse <- round(summary(lm4)$sigma, 4)
model3_r_squared <- round(summary(lm4)$r.squared, 4)
model3_adj_r_squared <- round(summary(lm4)$adj.r.squared, 4)
model3_f_statistic <- round(summary(lm4)$fstatistic[1], 4)
model3_aic <- round(AIC(lm4), 4)

model3_metrics <- c(model3_rse, model3_r_squared, model3_adj_r_squared, model3_f_statistic,
                    model3_aic)
```

Combine all metrics

```
metrics <- data.frame(model1_metrics, model2_metrics, model3_metrics)
metrics_rownames <- c("RSE", "RSquared", "Adj-RSquared", "F-Statistic", "AIC")
metrics_headers <- c("Model1", "Model2", "Model3")
rownames(metrics) <- metrics_rownames
colnames(metrics) <- metrics_headers
metrics <- metrics %>% kbl() %>% kable_styling()
metrics
```

The model diagnostic plots for all three models appear to be fairly similar. The Q-Q plot shows that the distribution is nearly normal and the residual vs fitted plot also shows no specific patter to worry about. Also, looking at the metrics of all three models, we can see that the values are fairly close. The RSE values are around 13 for all three models and the R-Square and Adj. R-Squared are about 30% for all three models. Furthermore, the AIC (Akaike Information Criteria) for all models are fairly close as well. However, the F-Statistic for Model3 is well above those for models 1 and 2. We can see that as we kept improving the model by selecting features whose p-values are significant, other model metrics remain fairly similar but the F-Statistic improved significantly from model1 to model3. Hence, we select model3 since it has a higher F-Statistic and it's a simpler model than the others and contains less features that are almost all statistically significant.

Summary

Now that we have the model selected, let's run it against the training data and see how accurate all of them are to make sure that model three is the best is by looking at the RSME. Then run it against the data we have in the Eval Training Set.

```
test_predictions = predict(lm2, newdata=test_set, interval ="predict")


test_set_1 <- cbind(test_set,test_predictions)

# RMSE
paste0("The Root Sqaure Mean Error for model one is: ", round(sqrt(mean((test_set_1$TARGET_WINS - test_s
```

```
## [1] "The Root Sqaure Mean Error for model one is: 12.3"
```

```
test_predictions = predict(lm3, newdata=test_set, interval ="predict")

test_set_2 <- cbind(test_set,test_predictions)

# RMSE
paste0("The Root Sqaure Mean Error for model two is: ", round(sqrt(mean((test_set_2$TARGET_WINS - test_s
```

```
## [1] "The Root Sqaure Mean Error for model two is: 12.5"
```

```
test_predictions = predict(lm4, newdata=test_set, interval ="predict")

test_set_3 <- cbind(test_set,test_predictions)

# RMSE
paste0("The Root Sqaure Mean Error for model three is: ", round(sqrt(mean((test_set_3$TARGET_WINS - tes
```

```
## [1] "The Root Sqaure Mean Error for model three is: 12.51"
```

Linear Model One has the lowest RSME, but overall they are not too far apart, meaning they will perform similar so we will still work with Linear Model Three.

```
test_predictions = data.frame(predict(lm4, newdata=baseball_eval, interval ="predict"))

test_predictions$predictions <- test_predictions %>% select(fit) %>% mutate_if(is.numeric, round)

test_predictions <- test_predictions %>% select(-fit)

baseball_eval_final <- cbind(test_predictions,baseball_eval)
```

Appendix

Here is the model run with Baseball Eval. Fit is the predicted wins value, while lwr and upr are the values that fall within a 95% confidence degree.

```
baseball_eval_final %>%
  kbl() %>%
  kable_paper("hover", full_width = F, html_font = "Times New Roman", font_size = 10) %>%
  kable_styling(bootstrap_options = c("striped", "hover", "condensed"))
```

Here is the full code for the project.

lwr	upr	predictions	INDEX	TEAM_BATTING_H	TEAM_BATTING_2B	TEAM_BATTING_
41.350848	94.46208	68	9	1209	170	
41.839095	94.97114	68	10	1221	151	
49.214303	102.33247	76	14	1395	183	
60.533402	113.63861	87	47	1539	309	
42.145847	95.25464	69	60	1445	203	
41.094911	94.22833	68	63	1431	236	
52.572600	105.79078	79	74	1430	219	
47.913359	101.06453	74	83	1385	158	
45.820956	98.94553	72	98	1259	177	
47.753170	100.84848	74	120	1397	212	
48.264837	101.37137	75	123	1427	243	
57.800401	110.90012	84	135	1496	239	
55.276351	108.43522	82	138	1420	223	
53.878734	106.98109	80	140	1460	232	
52.069829	105.18523	79	151	1411	195	
53.464345	106.58761	80	153	1434	192	
46.265018	99.34858	73	171	1297	204	
54.389187	107.46861	81	184	1446	284	
42.189647	95.30478	69	193	1276	162	
65.691883	118.80279	92	213	1715	322	
55.363319	108.49636	82	217	1520	295	
57.839352	110.95030	84	226	1597	291	
53.361572	106.45265	80	230	1453	256	
47.172487	100.25511	74	241	1378	225	
57.678895	110.76008	84	291	1516	277	
60.637675	113.74369	87	294	1556	288	
26.628296	80.11148	53	300	1499	183	
50.265541	103.36815	77	348	1464	263	
54.898005	108.08389	81	350	1558	318	
49.426308	102.63963	76	357	1502	308	
60.322680	113.42118	87	367	1596	320	
58.457558	111.52281	85	368	1546	260	
56.169930	109.24534	83	372	1516	282	
57.133747	110.21204	84	382	1550	275	
54.786096	107.87231	81	388	1447	260	
54.747265	107.85626	81	396	1450	252	
49.191270	102.25896	76	398	1347	239	
63.511035	116.66855	90	403	1561	260	
58.142547	111.25795	85	407	1578	252	
61.845815	114.95878	88	410	1598	259	
53.480584	106.60448	80	412	1497	322	
59.294736	112.39145	86	414	1569	310	
2.252219	56.12103	29	436	1119	118	
73.507943	126.94855	100	440	1609	196	
62.607181	115.87249	89	476	1514	175	
65.666171	118.82528	92	479	1657	237	
70.858826	124.07677	97	481	1746	213	
48.077078	101.17221	75	501	1319	224	
44.311333	97.42191	71	503	1293	204	
51.031722	104.11565	78	506	1420	235	
53.507420	106.59165	80	519	1496	269	
59.054371	112.19629	86	522	1625	289	
52.337103	105.41508	79	550	1391	239	
48.401519	101.49272	75	554	1319	203	
50.870833	103.93379	77	566	1411	251	
52.940251	106.00741	79	578	1420	221	
63.432941	116.58409	90	596	1552	206	

```r
# Libraries used in this project
library(RCurl)
library(tidyverse)
# install.packages("caret")
library(corrplot)
library("caret")
library(kableExtra)
library(caTools)
library(car)
library(ggResidpanel)

# 1. Data Exploration

# Importing of data and a quick header check of the data

#eval <- getURL("https://raw.githubusercontent.com/cmm6/data608/main/moneyball-evaluation-data.csv",.op
#train <- getURL("https://raw.githubusercontent.com/cmm6/data608/main/moneyball-training-data.csv",.opt
baseball_eval <- read.csv("https://raw.githubusercontent.com/cmm6/data608/main/moneyball-evaluation-data
baseball_training <- read.csv("https://raw.githubusercontent.com/cmm6/data608/main/moneyball-training-da

baseball_eval <- subset(baseball_eval)
baseball_training = subset(baseball_training, select = -c(INDEX) )

print(dim(baseball_training))
print(head(baseball_training))

# Summary of the baseball_training data
summary(baseball_training)

# Scatterplot of the baseball_training data
pairs(baseball_training, lower.panel = NULL, cex = 0.4, cex.labels=0.5)

# Boxplot of the baseball_training data
boxplot(baseball_training$TEAM_BATTING_BB)
plot(baseball_training$TEAM_BATTING_BB,baseball_training$TARGET_WINS)
boxplot(baseball_training$TEAM_BATTING_SO)
plot(baseball_training$TEAM_BATTING_SO,baseball_training$TARGET_WINS)
boxplot(baseball_training$TEAM_PITCHING_SO)
plot(baseball_training$TEAM_PITCHING_SO,baseball_training$TARGET_WINS)

# Correlation of the data
training_cor <- cor(na.omit(baseball_training))
corrplot(training_cor, method = 'number',number.cex=7/ncol(baseball_training))

# Linear Model
lm_baseline <- lm(TARGET_WINS~.,baseball_training)
summary(lm_baseline)

# 2. Data Preparation

# Summary of baseball_training
summary(baseball_training)
```

```r
# A look at the NA's as a percentage of total data
sapply(baseball_training, function(x) (sum(is.na(x) / nrow(baseball_training) *100)))

# The mean
baseball_training <- baseball_training %>% mutate_at(vars(-group_cols()),~ifelse(is.na(.) | is.nan(.),
                                        mean(.,na.rm=TRUE),.))

# Verify there are no more NAs

sum(is.na(baseball_training))

# Check if there are any issues with collinearity.
training_cor <- cor(na.omit(baseball_training))
corrplot(training_cor,method = 'color' ,order = 'hclust', addrect = 2)

#  New variables created for baseball_training

baseball_training$PRED_RUNS <- baseball_training$TEAM_PITCHING_BB + baseball_training$TEAM_BATTING_HBP +


baseball_training$TEAM_FIELDING <- baseball_training$TEAM_FIELDING_DP - baseball_training$TEAM_FIELDING_

# new variables for baseball_eval

baseball_eval <- baseball_eval %>% mutate_at(vars(-group_cols()),~ifelse(is.na(.) | is.nan(.),
                                        mean(.,na.rm=TRUE),.))


baseball_eval$PRED_RUNS <- baseball_eval$TEAM_PITCHING_BB + baseball_eval$TEAM_BATTING_HBP + baseball_ev


baseball_eval$TEAM_FIELDING <- baseball_eval$TEAM_FIELDING_DP - baseball_eval$TEAM_FIELDING_E

# Split the data

set.seed(678)

split <- sample.split(baseball_training$TARGET_WINS, SplitRatio = 0.8)
training_set <- subset(baseball_training, split == TRUE)
test_set <- subset(baseball_training, split == FALSE)

# 3. Build Models


baseball_training %>%
  gather(variable, value, TARGET_WINS:TEAM_FIELDING_DP) %>%
  ggplot(., aes(value)) +
  geom_density(fill = "dodgerblue4", color="dodgerblue4") +
  facet_wrap(~variable, scales ="free", ncol = 4) +
  labs(x = element_blank(), y = element_blank())


# Model 1
```

```
lm2 <- lm(TARGET_WINS ~ TEAM_BATTING_H + TEAM_BATTING_2B + TEAM_BATTING_3B +
           TEAM_BATTING_HR + TEAM_BATTING_BB + TEAM_BATTING_SO +
           TEAM_BASERUN_SB +  TEAM_PITCHING_H +
            TEAM_PITCHING_BB + TEAM_PITCHING_SO +
           TEAM_FIELDING_E + TEAM_FIELDING_DP, data = training_set)

summary(lm2)
vif(lm2)

# Diagnostics for model 1

resid_panel(lm2, plots='default', smoother = TRUE)

par(mfrow=c(2,2))
plot(lm2)

# Model 2

lm3 <- lm(TARGET_WINS ~ TEAM_BATTING_H + TEAM_BATTING_2B + TEAM_BATTING_3B +
           TEAM_BATTING_HR + TEAM_BATTING_BB + TEAM_BATTING_SO +
           TEAM_BASERUN_SB + TEAM_FIELDING_E, data = training_set)


summary(lm3)
vif(lm3)

# Diagnostics for model 2

resid_panel(lm3, plots='default', smoother = TRUE)
par(mfrow=c(2,2))
plot(lm3)

# Model 3

lm4 <- lm(TARGET_WINS ~ TEAM_BATTING_H + TEAM_BATTING_2B + TEAM_BATTING_3B +
           TEAM_BATTING_HR +  TEAM_BASERUN_SB + TEAM_FIELDING_E, data = training_set)


summary(lm4)
vif(lm4)

# Diagnostics for model 3

resid_panel(lm4, plots='default', smoother = TRUE)
par(mfrow=c(2,2))
plot(lm4)


# Select Models

# Model 1

# extract the rse, r.squared, adj.r.squared, F-statistic, and AIC for model 1
```

```r
model1_rse <- round(summary(lm2)$sigma, 4)
model1_r_squared <- round(summary(lm2)$r.squared, 4)
model1_adj_r_squared <- round(summary(lm2)$adj.r.squared, 4)
model1_f_statistic <- round(summary(lm2)$fstatistic[1], 4)
model1_aic <- round(AIC(lm2), 4)

model1_metrics <- c(model1_rse, model1_r_squared, model1_adj_r_squared, model1_f_statistic,
                    model1_aic)


# Model 2

# extract the rse, r.squared, adj.r.squared, F-statistic, and AIC for model 2
model2_rse <- round(summary(lm3)$sigma, 4)
model2_r_squared <- round(summary(lm3)$r.squared, 4)
model2_adj_r_squared <- round(summary(lm3)$adj.r.squared, 4)
model2_f_statistic <- round(summary(lm3)$fstatistic[1], 4)
model2_aic <- round(AIC(lm3), 4)

model2_metrics <- c(model2_rse, model2_r_squared, model2_adj_r_squared, model2_f_statistic,
                    model2_aic)


# Model 3

# extract the rse, r.squared, adj.r.squared, F-statistic, and AIC for model 3
model3_rse <- round(summary(lm4)$sigma, 4)
model3_r_squared <- round(summary(lm4)$r.squared, 4)
model3_adj_r_squared <- round(summary(lm4)$adj.r.squared, 4)
model3_f_statistic <- round(summary(lm4)$fstatistic[1], 4)
model3_aic <- round(AIC(lm4), 4)

model3_metrics <- c(model3_rse, model3_r_squared, model3_adj_r_squared, model3_f_statistic,
                    model3_aic)


# Combine all metrics

metrics <- data.frame(model1_metrics, model2_metrics, model3_metrics)
metrics_rownames <- c("RSE", "RSquared", "Adj-RSquared", "F-Statistic", "AIC")
metrics_headers <- c("Model1", "Model2", "Model3")
rownames(metrics) <- metrics_rownames
colnames(metrics) <- metrics_headers
metrics <- metrics %>% kbl() %>% kable_styling()
metrics


# Summary

test_predictions = predict(lm2, newdata=test_set, interval ="predict")


test_set_1 <- cbind(test_set,test_predictions)
```

```
# RMSE
paste0("The Root Sqaure Mean Error for model one is: ", round(sqrt(mean((test_set_1$TARGET_WINS - test_s


test_predictions = predict(lm3, newdata=test_set, interval ="predict")

test_set_2 <- cbind(test_set,test_predictions)

# RMSE
paste0("The Root Sqaure Mean Error for model two is: ", round(sqrt(mean((test_set_2$TARGET_WINS - test_s


test_predictions = predict(lm4, newdata=test_set, interval ="predict")

test_set_3 <- cbind(test_set,test_predictions)

# RMSE
paste0("The Root Sqaure Mean Error for model three is: ", round(sqrt(mean((test_set_3$TARGET_WINS - tes


test_predictions = data.frame(predict(lm4, newdata=baseball_eval, interval ="predict"))

test_predictions$predictions <- test_predictions %>% select(fit) %>% mutate_if(is.numeric, round)

test_predictions <- test_predictions %>% select(-fit)

baseball_eval_final <- cbind(test_predictions,baseball_eval)


# Appendix


baseball_eval_final %>%
  kbl() %>%
  kable_paper("hover", full_width = F, html_font = "Times New Roman", font_size = 10) %>%
  kable_styling(bootstrap_options = c("striped", "hover", "condensed"))
```

‘‘‘