Data 621 - Homework 1

Group 4 Layla Quinones, Ian Costello, Dmitriy Burtsev & Esteban Aramayo

September 26, 2021

About the Data

The data set consists of 2,276 records and 17 different variables, with each observation corresponding to a baseball teams performance in a single year. The time horizon of these data are from 1871, the same year as the first recorded professional baseball game through 2006.

General objective

Through linear regression, train the data to predict the number of wins.

Challenges Right Off the Bat (so to speak)

The data set covers a very large time period. The rules and play style of baseball have changed a great deal from the late 19th-century. The season year of the team would be an important factor in improving these models. Additionally, certain clubs have bucked trends in winning or losing despite these metrics. The Boston Red Sox and Chicago Cubs had very long dry spells, even with good team statistics.

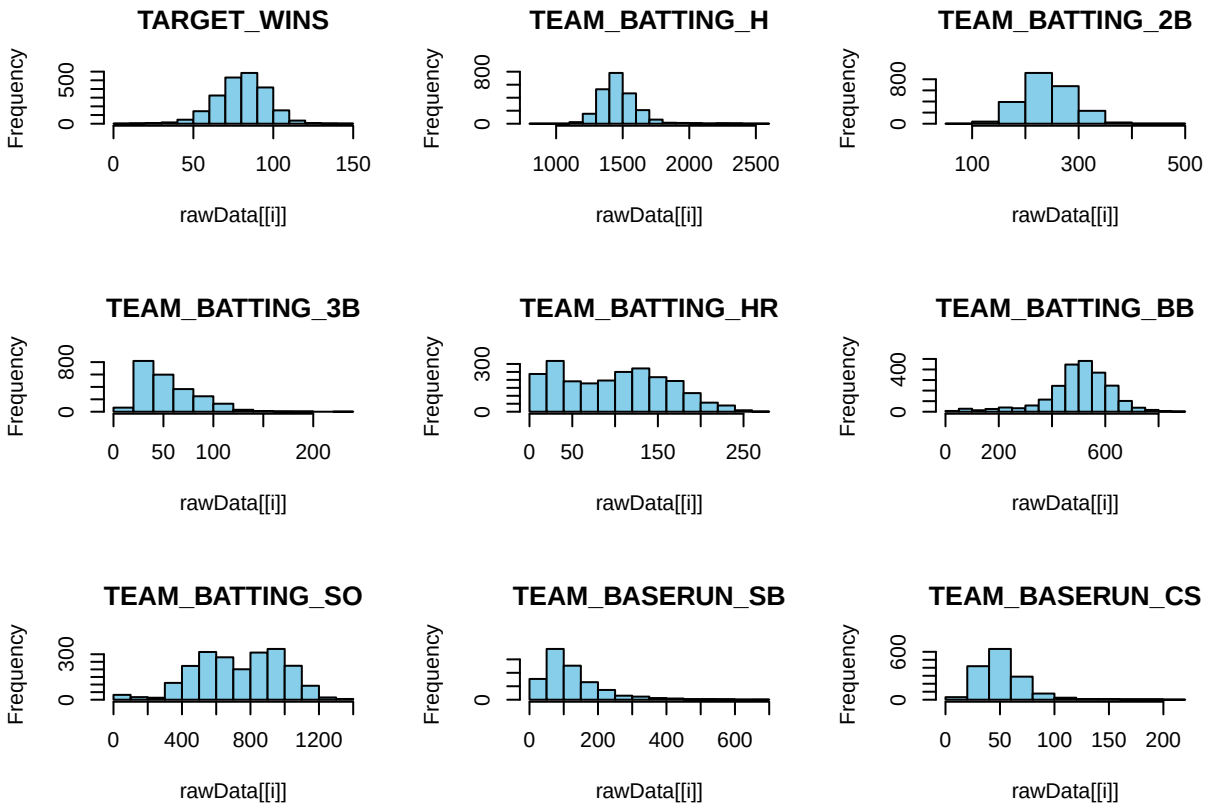
Data Exploration

Our data is stored for easy reference among the team on GitHub. We use a number of packages **(Code Appendix 1.1)** to complete this work, including the ever-useful `tiyverse` and `caret`. With 2,276 team observations and 17 variables. Of those, 15 are features, 1 is an index, and the remaining is our target variable for number of wins. Right away, by just reading out the raw data **(Code Appendix 2.1)** we already know that missing values will need to be accounted for in all of these features.

In order to properly train and test our data, we create a data partition for each at 80% training data **(Code Appendix 3.1)**.

Summary Statistics

For each of the variables, these summary statistics **(Code Appendix 4.1)** provide a nice overview of each feature, its variation, and paths for potential transformations later on for model construction. The histograms in figure 1 are a quick way to see the shape of the distributions for each feature. Of note are the normally distributed variables, like our target variable for wins, base hits by batters, doubles by batters, walks by batters, and batters hit by pitches. The more skewed features include hits allowed, strike outs by pitchers (a very difficult thing to do consistently), and team fielding errors. Once again, we can observe the extent of the N/As and outliers that we'll have to account for.


TARGET_WINS
Frequency
0
500
0
50
100
150
rawData[[i]]
TEAM_BATTING_H
Frequency
0
800
1000
1500
2000
2500
rawData[[i]]
TEAM_BATTING_2B
Frequency
0
800
100
300
500
rawData[[i]]
TEAM_BATTING_3B
Frequency
0
800
0
50
100
200
rawData[[i]]
TEAM_BATTING_HR
Frequency
0
300
0
50
150
250
rawData[[i]]
TEAM_BATTING_BB
Frequency
0
400
0
200
600
rawData[[i]]
TEAM_BATTING_SO
Frequency
0
300
0
400
800
1200
rawData[[i]]
TEAM_BASERUN_SB
Frequency
0
0
200
400
600
rawData[[i]]
TEAM_BASERUN_CS
Frequency
0
600
0
50
100
150
200
rawData[[i]]

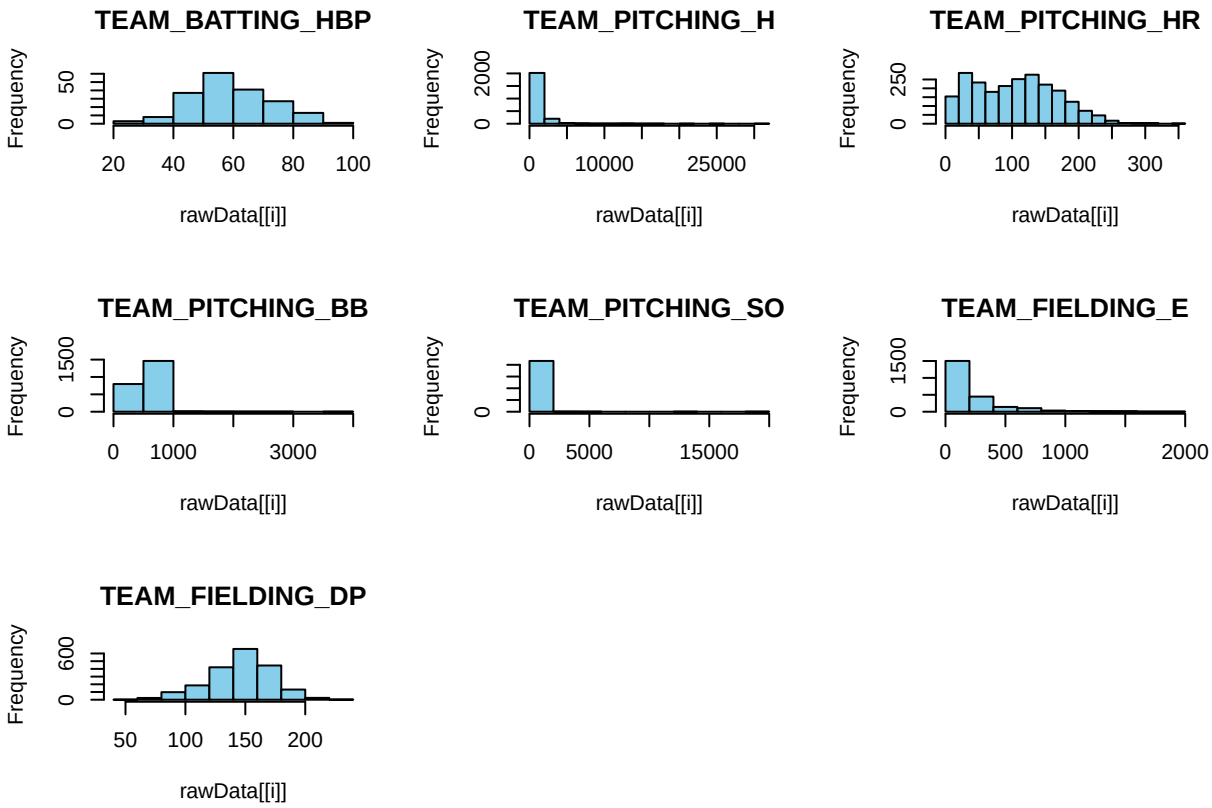


Figure 1, Feature Histograms (Code Appendix 4.2)

Box Plots

This box plot visualization (Figure 2) gives us an idea of the outliers we have in each variable, but does not give us a good sense of the distribution. We can use the histograms (Figure 1) above to interpret shape. From the box plots, we see that the variable `TEAM_PITCHING_H` has the greatest number of outliers. This may mean we throw that variable out altogether and not consider it in our models.

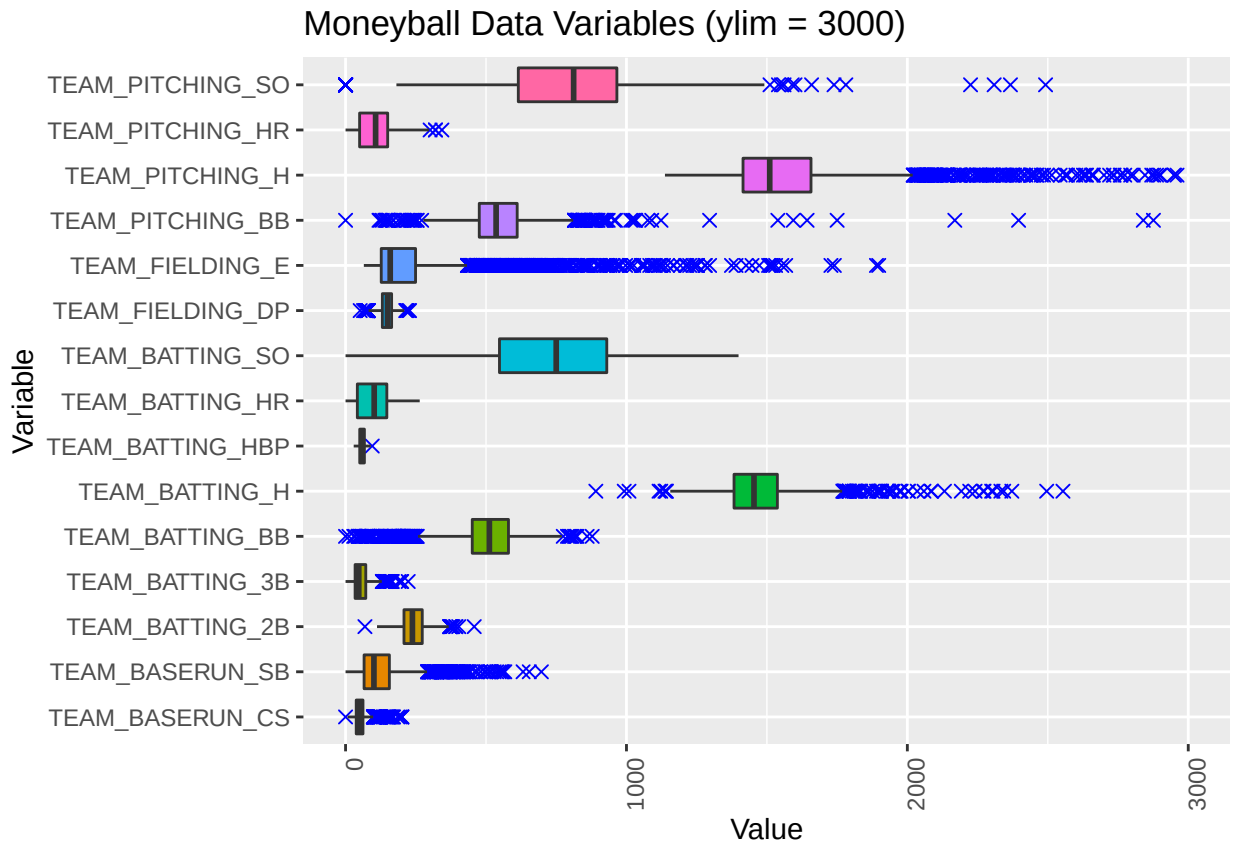


Figure 2, Feature Box Plot with Outliers (Code Appendix 4.3)

Data Pre-Processing

Missing Data

In figure 3, we can see immidiately that there are a a few variables with lots of missing data. Even with imputation without at least 40% to 50% actual data, it would not be that informative to use in any model. Indeed, `TEAM_BATTING_HBP` has about 92% missing data **(Appendix 5.2)** and so is removed for the data set and will not be considered for the models. We will be using the histograms (figure 1) for each other variable to decide how and whether to impute during the pre-processing stage.

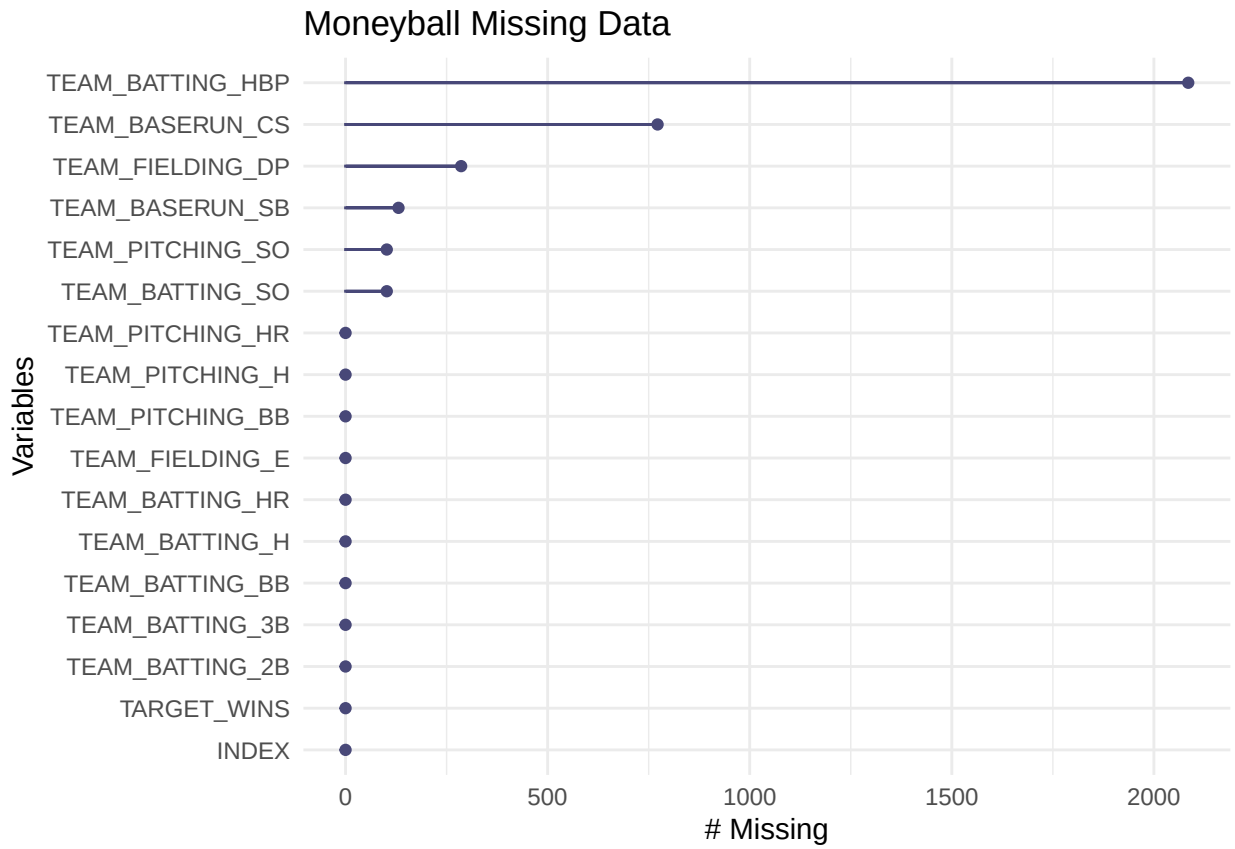


Figure 3, Missing Data

Correlation Matrix and Multicollinearity

It is important to check for features which may also be correlated. Simply, having multiple features relate to themselves can cause overfitting, reduced p values, and strange variances in the data. To avoid this, we exclude one or more of the variables. In the correlation matrix (Figure 4), we see that `TEAM_BATTING_HR` and `TEAM_PITCHING_SO` are very intertwined, showing up as bright red. We'll take care when constructing our models not to use both. We determined later **(Code Appendix 5.4)** that `TEAM_BATTING_HR` had very weak effects on the model and so selected that feature to be removed entirely.

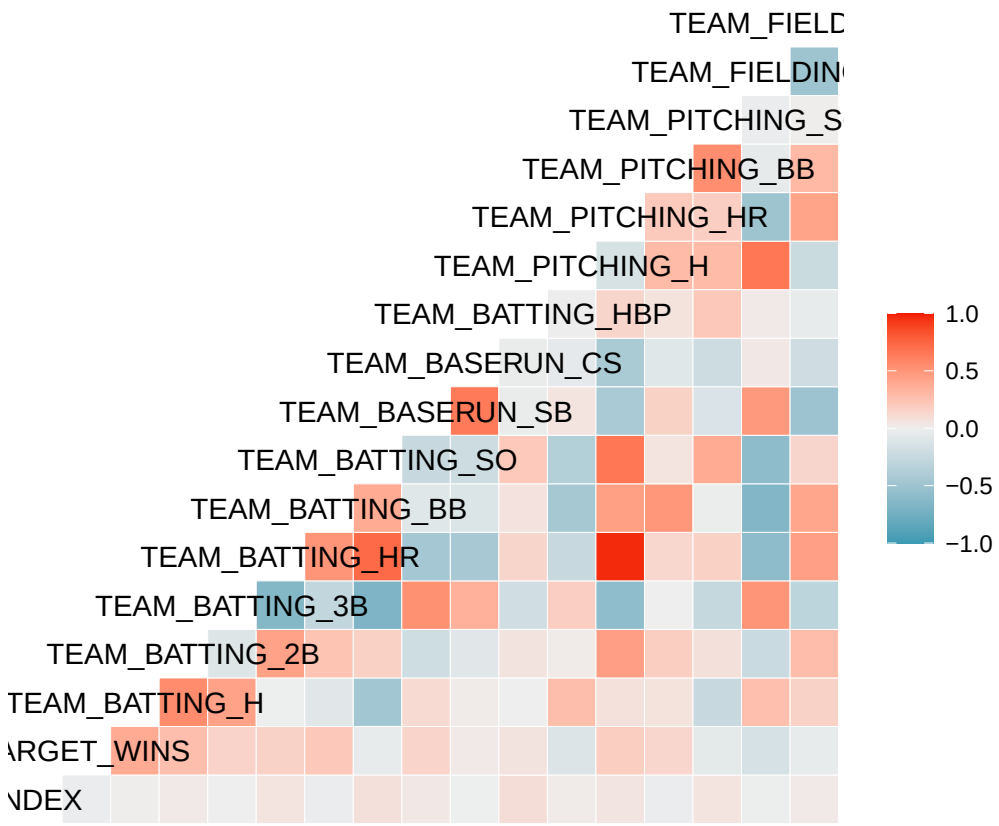


Figure 4, Correlation Matrix (Code Appendix 5.3)

Imputing Missing Data

For the other features with significant amounts missing data, we can impute using either the mean or median of the feature. In observing the histograms (Figure 1), if the shape is more skewed, we would seek to use the median. If the distribution appears more normal, we can use the mean (average) **(Code Appendix 5.5 & 5.5.1)**.

Feature Plots

With the data cleaned and imputed, we can again review the features and begin selecting them for our models. The feature plots (Figure 5) below summarize the potential effect of each feature on the target variable. Obviously, Target wins is basically a straight line since it is itself the target variable and a perfect line.

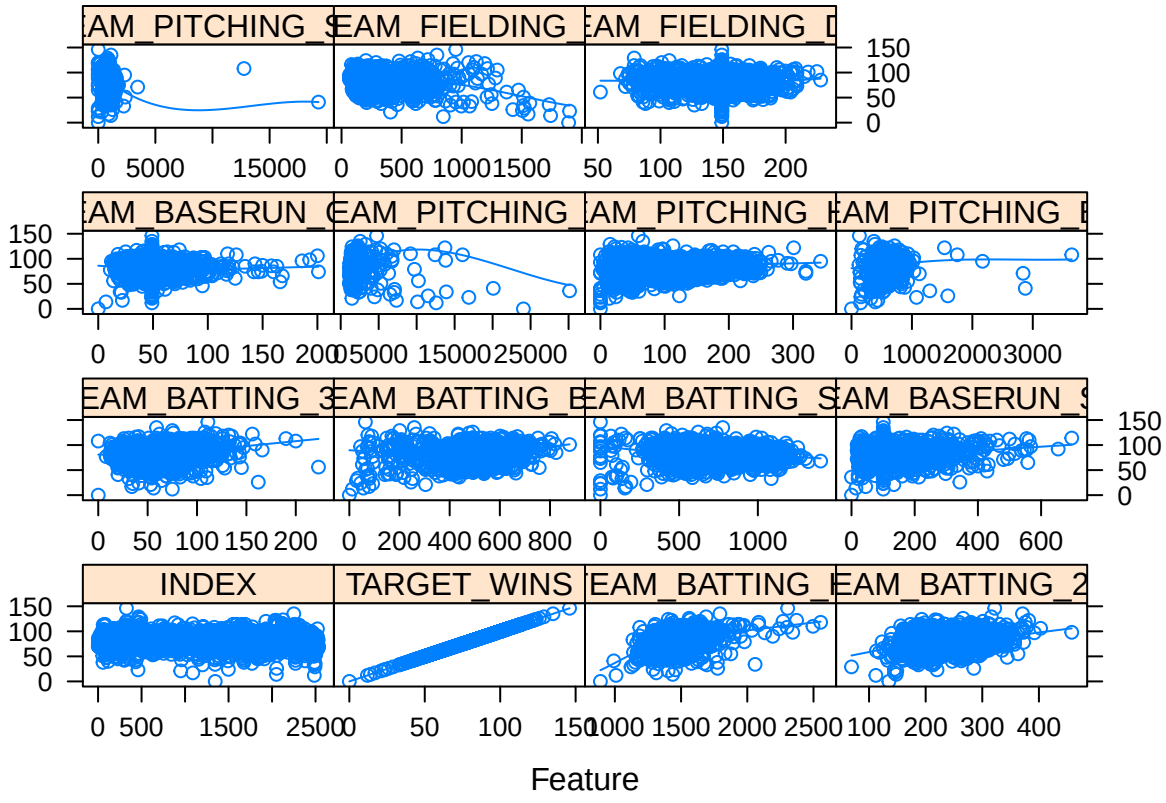


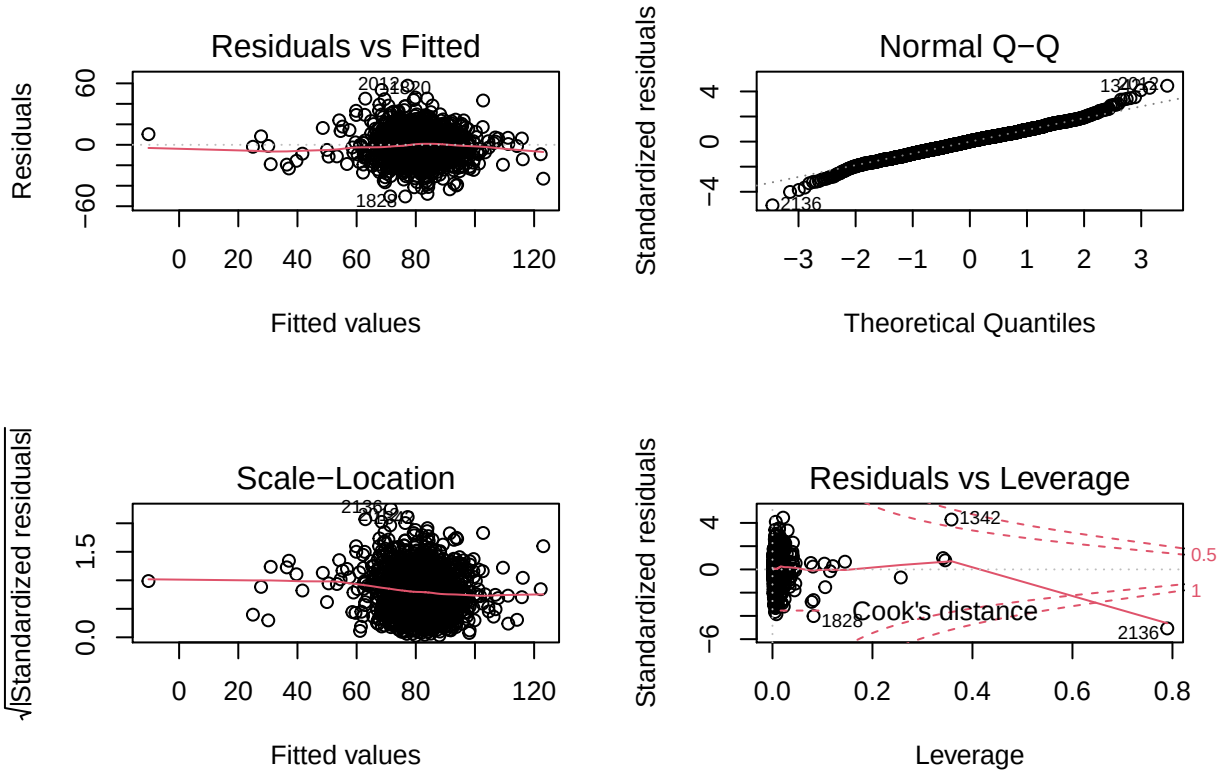
Figure 5, Feature Plots (Code Appendix 5.6)

Build Models

Model 1: "The Kitchen Sink"

$R^2 = 0.303$

Our first model is the so-called kitchen sing approach, where all features are included. We get a pretty lousy R^2 value, even though the residuals appears to be quite random, if a bit clumped and the F-statistic indicated our model does say *something*.




```
##
## Call:
## lm(formula = TARGET_WINS ~ ., data = trainData)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -50.444  -8.364   0.404   8.140  57.744
##
## Coefficients:
##                    Estimate Std. Error t value Pr(>|t|)
## (Intercept)      23.4416666  6.0755834   3.858 0.000118 ***
## INDEX            -0.0006326  0.0004206  -1.504 0.132699
## TEAM_BATTING_H    0.0490246  0.0041222  11.893  < 2e-16 ***
## TEAM_BATTING_2B  -0.0227145  0.0102269  -2.221 0.026471 *
## TEAM_BATTING_3B   0.0605813  0.0187539   3.230 0.001259 **
## TEAM_BATTING_BB   0.0150025  0.0059409   2.525 0.011645 *
## TEAM_BATTING_SO  -0.0083507  0.0028737  -2.906 0.003706 **
## TEAM_BASERUN_SB   0.0252307  0.0048857   5.164 2.68e-07 ***
## TEAM_BASERUN_CS  -0.0122584  0.0176581  -0.694 0.487640
## TEAM_PITCHING_H  -0.0009030  0.0003871  -2.333 0.019775 *
## TEAM_PITCHING_HR  0.0588300  0.0097636   6.025 2.04e-09 ***
## TEAM_PITCHING_BB -0.0027049  0.0042154  -0.642 0.521174
## TEAM_PITCHING_SO  0.0036413  0.0010092   3.608 0.000317 ***
## TEAM_FIELDING_E  -0.0183293  0.0027725  -6.611 5.00e-11 ***
## TEAM_FIELDING_DP -0.1154644  0.0144517  -7.990 2.39e-15 ***
## ---
```

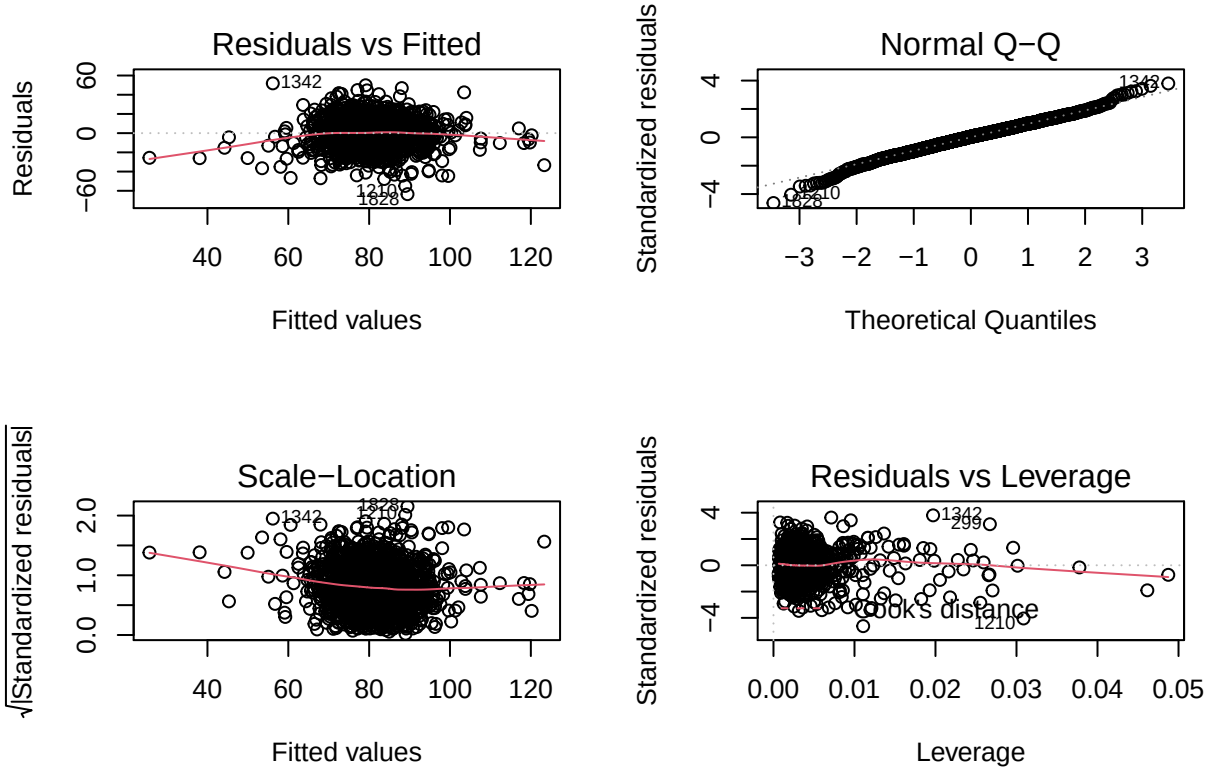
```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 13.11 on 1809 degrees of freedom
## Multiple R-squared:  0.3085, Adjusted R-squared:  0.3032
## F-statistic: 57.66 on 14 and 1809 DF,  p-value: < 2.2e-16
```

Figure 6, Model 1 Residuals vs Fitted, QQ, Scale-Location, Residuals vs Leverage (Code Appendix 6.2)

Model 2: Targeting Most Impactful Features

Reviewing the standard errors, and the correlation matrix, we what we think may be the most impactful metrics. Unfortunately, is has the opposite effect on our R^2, which is reduced roughly 8%.

$R^2 = 0.230$




```
##
## Call:
## lm(formula = TARGET_WINS ~ TEAM_BATTING_H + TEAM_BATTING_2B +
##     TEAM_BATTING_3B + TEAM_BATTING_BB + TEAM_BATTING_SO, data = trainData)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -63.489  -8.737   0.549   8.865  51.831
##
## Coefficients:
```

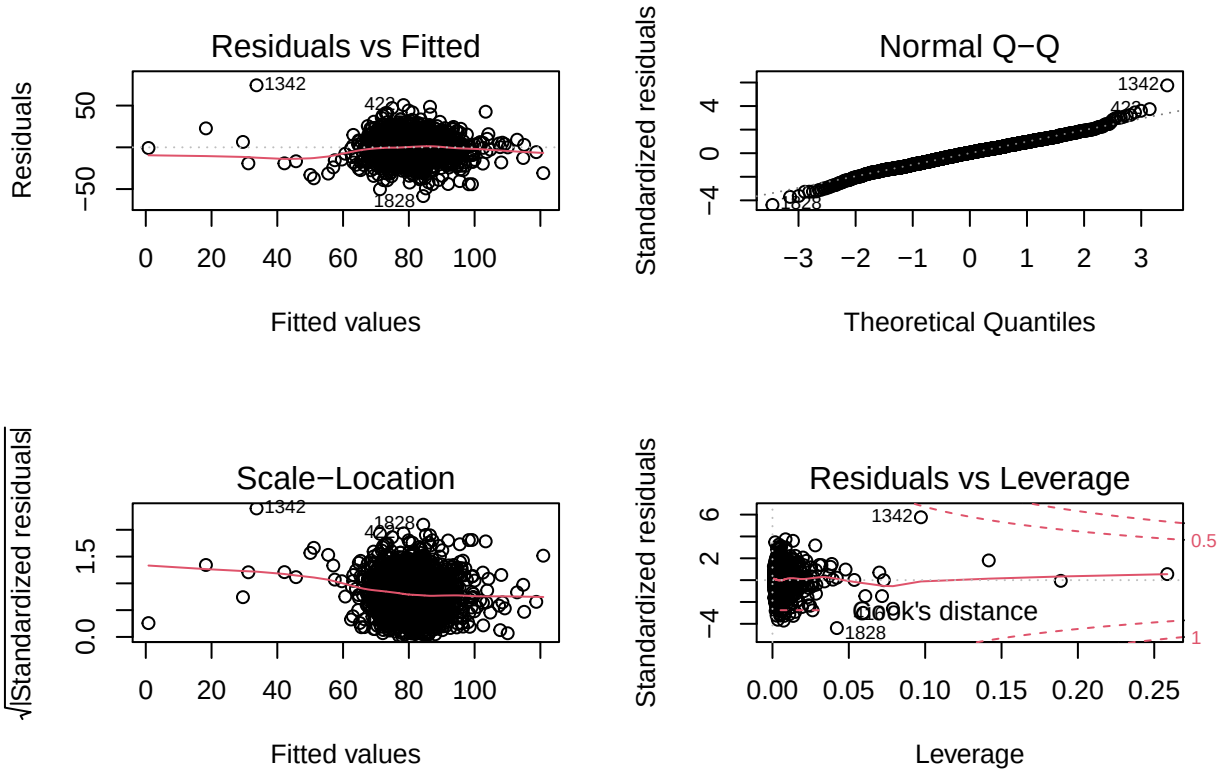
```
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)     -16.507104   4.906108  -3.365 0.000782 ***
## TEAM_BATTING_H    0.050124   0.003724  13.460  < 2e-16 ***
## TEAM_BATTING_2B  -0.018508   0.010346  -1.789 0.073804 .
## TEAM_BATTING_3B   0.075932   0.016238   4.676 3.14e-06 ***
## TEAM_BATTING_BB   0.031327   0.002939  10.660  < 2e-16 ***
## TEAM_BATTING_SO   0.011561   0.002065   5.597 2.51e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 13.79 on 1818 degrees of freedom
## Multiple R-squared:  0.2317, Adjusted R-squared:  0.2296
## F-statistic: 109.7 on 5 and 1818 DF,  p-value: < 2.2e-16
```

Figure 7, Model 2 Residuals vs Fitted, QQ, Scale-Location, Residuals vs Leverage (Code Appendix 6.3)

Model 3: Adding in a Few More

At this point, we're looking for something better than our first model. Steadily, we're adding additional features to see if we can break and R^2 of 30%. Adding in these additional features increased the value from our previous model, but not better than the kitchen sink.

$R^2 = 0.230$



##

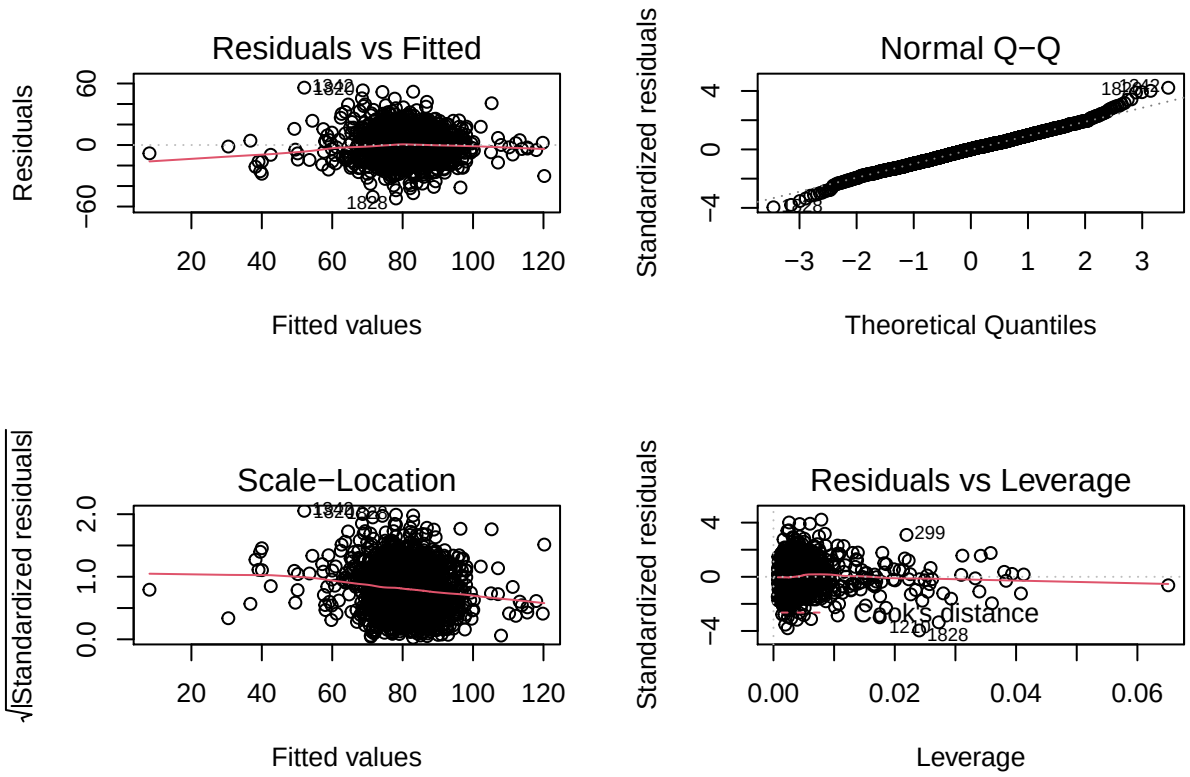
```
## Call:
## lm(formula = TARGET_WINS ~ TEAM_BATTING_H + TEAM_BATTING_2B +
##     TEAM_BATTING_3B + TEAM_BATTING_BB + TEAM_BATTING_SO + TEAM_BASERUN_SB +
##     TEAM_BASERUN_CS + TEAM_PITCHING_H + TEAM_PITCHING_HR, data = trainData)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -58.385  -8.926   0.477   8.994  74.310
##
## Coefficients:
##                    Estimate Std. Error t value Pr(>|t|)
## (Intercept)       1.1535194  5.8554891   0.197  0.84385
## TEAM_BATTING_H    0.0433115  0.0041997  10.313  < 2e-16 ***
## TEAM_BATTING_2B  -0.0084978  0.0104092  -0.816  0.41439
## TEAM_BATTING_3B   0.0573080  0.0190131   3.014  0.00261 **
## TEAM_BATTING_BB   0.0183362  0.0033997   5.393 7.82e-08 ***
## TEAM_BATTING_SO   0.0017697  0.0025771   0.687  0.49237
## TEAM_BASERUN_SB   0.0204570  0.0046655   4.385 1.23e-05 ***
## TEAM_BASERUN_CS   0.0094712  0.0178925   0.529  0.59664
## TEAM_PITCHING_H  -0.0015705  0.0002668  -5.886 4.70e-09 ***
## TEAM_PITCHING_HR  0.0421675  0.0097678   4.317 1.67e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 13.59 on 1814 degrees of freedom
## Multiple R-squared:  0.255,  Adjusted R-squared:  0.2514
## F-statistic: 69.01 on 9 and 1814 DF,  p-value: < 2.2e-16
```

Figure 8, Model 3 Residuals vs Fitted, QQ, Scale-Location, Residuals vs Leverage (Code Appendix 6.4)

Mode 4: Lets start transforming some of these variables

For this model, we attempted to do a few transformations. For three of the features which we found to be left-skewed (`TEAM_FIELDING_E`, `TEAM_BASERUN_SB`, and `TEAM_BASERUN_CS`) we add in the median value. The R^2 value is a bit better than our earlier attempts, but still not as high as the kitchen sink.

$R^2 = 0.282$




```
##
## Call:
## lm(formula = TARGET_WINS ~ TEAM_BATTING_H + TEAM_PITCHING_HR +
##     TEAM_FIELDING_DP + fieldingE + baserunSB + baserunCS, data = trainData_model4)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -52.101  -8.869   0.216   8.282  55.937
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)      21.796978   4.002272   5.446 5.85e-08 ***
## TEAM_BATTING_H    0.050854   0.002353  21.609  < 2e-16 ***
## TEAM_PITCHING_HR  0.029207   0.006657   4.387 1.21e-05 ***
## TEAM_FIELDING_DP -0.101189   0.014093  -7.180 1.01e-12 ***
## fieldingE        -0.022605   0.001754 -12.886  < 2e-16 ***
## baserunSB         0.035437   0.004109   8.625  < 2e-16 ***
## baserunCS        -0.025096   0.017767  -1.412    0.158
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 13.31 on 1817 degrees of freedom
## Multiple R-squared:  0.2842, Adjusted R-squared:  0.2818
## F-statistic: 120.2 on 6 and 1817 DF,  p-value: < 2.2e-16
```

Figure 9, Model 4 Residuals vs Fitted, QQ, Scale-Location, Residuals vs Leverage (Code

Appendix 6.5)

Model Selection

Based on the R^2 value of the models, our kitchen sink model (model 1) seemed to perform best, but by no means perfect for this exercise. With some additional features, perhaps team and season year we could improve our predictions.

According to our F-statistics, all our models had some effect and the residuals appeared more random throughout. Still, even with some tuning, we couldn't get the R^2 to be higher than our first model. We'll use that for our predictions and conclude the assignment by writing those to a csv **(Code Appendix 7.1)**.

Code Appendix

1.1 Libraries

r, warning = FALSE, message = FALSE, echo=FALSE, message=FALSE, warning=FALSE

library(kableExtra) library(tidyverse) library(tidymodels) library(VIM) library(naniar) library(GGally) library(caret) library(psych)

2.1 Data Import

```
urlTraining = "https://raw.githubusercontent.com/MsQCompSci/Data621Group4/main/HW1/moneyball-training-da

rawData <- read.csv(urlTraining)

str(rawData)
```

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

3.1 Train and Test Data Split

set.seed(123) trainRowNumbers <- createDataPartition(rawData$INDEX, p=0.8, list=FALSE) trainData <- rawData[trainRowNumbers,] testData <- rawData[-trainRowNumbers,]

4.1 Summary Statistics

```
summary(rawData)
```

```
##      INDEX         TARGET_WINS     TEAM_BATTING_H TEAM_BATTING_2B
##  Min.   :   1.0   Min.   :  0.00   Min.   : 891   Min.   : 69.0
##  1st Qu.: 630.8   1st Qu.: 71.00   1st Qu.:1383   1st Qu.:208.0
##  Median :1270.5   Median : 82.00   Median :1454   Median :238.0
##  Mean   :1268.5   Mean   : 80.79   Mean   :1469   Mean   :241.2
##  3rd Qu.:1915.5   3rd Qu.: 92.00   3rd Qu.:1537   3rd Qu.:273.0
##  Max.   :2535.0   Max.   :146.00   Max.   :2554   Max.   :458.0
##
##  TEAM_BATTING_3B  TEAM_BATTING_HR  TEAM_BATTING_BB TEAM_BATTING_SO
##  Min.   :  0.00   Min.   :  0.00   Min.   :  0.0   Min.   :   0.0
##  1st Qu.: 34.00   1st Qu.: 42.00   1st Qu.:451.0   1st Qu.: 548.0
##  Median : 47.00   Median :102.00   Median :512.0   Median : 750.0
##  Mean   : 55.25   Mean   : 99.61   Mean   :501.6   Mean   : 735.6
##  3rd Qu.: 72.00   3rd Qu.:147.00   3rd Qu.:580.0   3rd Qu.: 930.0
##  Max.   :223.00   Max.   :264.00   Max.   :878.0   Max.   :1399.0
##                                                    NA's   :102
##  TEAM_BASERUN_SB TEAM_BASERUN_CS TEAM_BATTING_HBP TEAM_PITCHING_H
##  Min.   :  0.0   Min.   :  0.0   Min.   :29.00    Min.   : 1137
##  1st Qu.: 66.0   1st Qu.: 38.0   1st Qu.:50.50    1st Qu.: 1419
##  Median :101.0   Median : 49.0   Median :58.00    Median : 1518
##  Mean   :124.8   Mean   : 52.8   Mean   :59.36    Mean   : 1779
##  3rd Qu.:156.0   3rd Qu.: 62.0   3rd Qu.:67.00    3rd Qu.: 1682
##  Max.   :697.0   Max.   :201.0   Max.   :95.00    Max.   :30132
##  NA's   :131     NA's   :772     NA's   :2085
##  TEAM_PITCHING_HR TEAM_PITCHING_BB TEAM_PITCHING_SO  TEAM_FIELDING_E
##  Min.   :  0.0    Min.   :   0.0   Min.   :    0.0   Min.   :  65.0
##  1st Qu.: 50.0    1st Qu.: 476.0   1st Qu.:  615.0   1st Qu.: 127.0
##  Median :107.0    Median : 536.5   Median :  813.5   Median : 159.0
##  Mean   :105.7    Mean   : 553.0   Mean   :  817.7   Mean   : 246.5
##  3rd Qu.:150.0    3rd Qu.: 611.0   3rd Qu.:  968.0   3rd Qu.: 249.2
##  Max.   :343.0    Max.   :3645.0   Max.   :19278.0   Max.   :1898.0
##                                    NA's   :102
##  TEAM_FIELDING_DP
##  Min.   : 52.0
##  1st Qu.:131.0
##  Median :149.0
##  Mean   :146.4
##  3rd Qu.:164.0
##  Max.   :228.0
##  NA's   :286
```

```
describe(rawData)
```

```
##                  vars    n    mean      sd median trimmed    mad  min   max
## INDEX               1 2276 1268.46  736.35 1270.5 1268.57 952.57    1  2535
## TARGET_WINS         2 2276   80.79   15.75   82.0   81.31  14.83    0   146
## TEAM_BATTING_H      3 2276 1469.27  144.59 1454.0 1459.04 114.16  891  2554
## TEAM_BATTING_2B     4 2276  241.25   46.80  238.0  240.40  47.44   69   458
## TEAM_BATTING_3B     5 2276   55.25   27.94   47.0   52.18  23.72    0   223
## TEAM_BATTING_HR     6 2276   99.61   60.55  102.0   97.39  78.58    0   264
## TEAM_BATTING_BB     7 2276  501.56  122.67  512.0  512.18  94.89    0   878
## TEAM_BATTING_SO     8 2174  735.61  248.53  750.0  742.31 284.66    0  1399
## TEAM_BASERUN_SB     9 2145  124.76   87.79  101.0  110.81  60.79    0   697
## TEAM_BASERUN_CS    10 1504   52.80   22.96   49.0   50.36  17.79    0   201
## TEAM_BATTING_HBP   11  191   59.36   12.97   58.0   58.86  11.86   29    95
## TEAM_PITCHING_H    12 2276 1779.21 1406.84 1518.0 1555.90 174.95 1137 30132
## TEAM_PITCHING_HR   13 2276  105.70   61.30  107.0  103.16  74.13    0   343
## TEAM_PITCHING_BB   14 2276  553.01  166.36  536.5  542.62  98.59    0  3645
## TEAM_PITCHING_SO   15 2174  817.73  553.09  813.5  796.93 257.23    0 19278
## TEAM_FIELDING_E    16 2276  246.48  227.77  159.0  193.44  62.27   65  1898
## TEAM_FIELDING_DP   17 1990  146.39   26.23  149.0  147.58  23.72   52   228
##                  range  skew kurtosis    se
## INDEX             2534  0.00    -1.22 15.43
## TARGET_WINS        146 -0.40     1.03  0.33
## TEAM_BATTING_H    1663  1.57     7.28  3.03
## TEAM_BATTING_2B    389  0.22     0.01  0.98
## TEAM_BATTING_3B    223  1.11     1.50  0.59
## TEAM_BATTING_HR    264  0.19    -0.96  1.27
## TEAM_BATTING_BB    878 -1.03     2.18  2.57
## TEAM_BATTING_SO   1399 -0.30    -0.32  5.33
## TEAM_BASERUN_SB    697  1.97     5.49  1.90
## TEAM_BASERUN_CS    201  1.98     7.62  0.59
## TEAM_BATTING_HBP    66  0.32    -0.11  0.94
## TEAM_PITCHING_H  28995 10.33   141.84 29.49
## TEAM_PITCHING_HR   343  0.29    -0.60  1.28
## TEAM_PITCHING_BB  3645  6.74    96.97  3.49
## TEAM_PITCHING_SO 19278 22.17   671.19 11.86
## TEAM_FIELDING_E   1833  2.99    10.97  4.77
## TEAM_FIELDING_DP   176 -0.39     0.18  0.59
```

4.2 Histograms

par(mfrow = c(3,3)) for(i in 2:ncol(rawData)) {#distribution of each variable hist(rawData[[i]], main = colnames(rawData[i]), col = "skyblue")

4.3 Box Plots

trainDataLONG <- rawData %>% select(-INDEX, -TARGET_WINS) %>% gather(key = Variable, value = Value)

ggplot(trainDataLONG, aes(Variable, Value, fill = Variable)) + geom_boxplot(outlier.colour="blue", outlier.shape=4, outlier.size=2, show.legend=FALSE) + ylim(0,3000) + theme(axis.text.x = element_text(angle = 90, hjust = 1)) + coord_flip()+ labs(title="Moneyball Data Variables (ylim = 3000)")

5.1 Data Pre-Processing

5.2 Missing Data

```
#Lets see how we can throw away (as a general rule )
#What percent is missing? (must be below 60))
data.frame(TEAM_BATTING_HBP = sum(is.na(trainData$TEAM_BATTING_HBP))/nrow(trainData), TEAM_BASERUN_CS =
```

```
##   TEAM_BATTING_HBP TEAM_BASERUN_CS TEAM_FIELDING_DP TEAM_BASERUN_SB
## 1                0               0                0               0
##   TEAM_BATTING_SO TEAM_PITCHING_SO
## 1               0                0
```

5.3 Correlation Matrix and Multicollinearity

corMat <- cor(trainData, use = "complete.obs") ggcorr(trainData)

5.4 Drop Variables

```
#Identify variable to drop
findCorrelation(cor(trainData),
                cutoff = 0.75,
                verbose = TRUE,
                names = TRUE)
```

```
## All correlations <= 0.75
```

```
## character(0)
```

```
#TEAM_PITCHING_HR will be omitted due to correlation (when we tested adding and subtracting this variab
#There are a number of correlated variables which may affect the model - we want to make sure that coli
```

5.5 Imputing Missing Data

trainData <- trainData %>% select(-TEAM_BATTING_HR, -TEAM_BATTING_HBP)

5.5.1 Impute Missing Data based on distribution trainData <- trainData %>% mutate(TEAM_BATTING_SO = ifelse(is.na(trainData$TEAM_BATTING_SO), mean(trainData$TEAM_BATTING_SO,na.rm=TRUE), trainData$TEAM_BATTING_SO), $TEAM_PITCHING_SO = ifelse(is.na(trainData$TEAM_PITCHING_SO), mean(trainData$TEAM_PITCHING_SO, na.rm = TRUE), trainData$TEAM_PITCHING_SO), TEAM_FIELDING_DP = ifelse(is.na(trainData$TEAM_FIELDING_DP), median(trainData$TEAM_FIELDING_DP,na.rm=TRUE), trainData$TEAM_FIELDING_DP), $TEAM_BASERUN_SB = ifelse(is.na(trainData$TEAM_BASERUN_SB), median(trainData$TEAM_BASERUN_SB, na.rm = TRUE), trainData$TEAM_BASERUN_SB), TEAM_BASERUN_CS = ifelse(is.na(trainData$TEAM_BASERUN_CS),

5.6 Feature Plots

featurePlot(y = unlist(trainData$TARGET_WINS), x = trainData, plot = "scatter", type = c("p", "smooth"), span = .5, layout = c(4, 3))

6.1 Build Models

6.2 Model 1: "The Kitchen Sink"

model_1 <- lm(TARGET_WINS ~ ., data = trainData) par(mfrow = c(2,2)) plot(model_1) summary(model_1)

6.3 Model 2: Targeting Most Impactful Features

model_2 <- lm(TARGET_WINS ~ TEAM_BATTING_H + TEAM_BATTING_2B + TEAM_BATTING_3B + TEAM_BATTING_BB + TEAM_BATTING_SO, data = trainData) par(mfrow = c(2,2)) plot(model_2) summary(model_2)

6.4 Model 3: Adding in a Few More

model_3 <- lm(TARGET_WINS ~ TEAM_BATTING_H + TEAM_BATTING_2B + TEAM_BATTING_3B + TEAM_BATTING_BB + TEAM_BATTING_SO + TEAM_BASERUN_SB + TEAM_BASERUN_CS + TEAM_PITCHING_H + TEAM_PITCHING_HR, data = trainData) par(mfrow = c(2,2)) plot(model_3) summary(model_3)

6.5 Model 4: Lets start transforming some of these variables

fieldingE <- trainData$TEAM_FIELDING_E + median(trainData$TEAM_FIELDING_E) baserunSB <- trainData$TEAM_BASERUN_SB + median(trainData$TEAM_BASERUN_SB) baserunCS <- trainData$TEAM_BASERUN_CS + median(trainData$TEAM_BASERUN_CS)

trainData_model4 <- mutate(trainData, fieldingE = unlist(fieldingE), TEAM_BASERUN_SB = baserunSB, TEAM_BASERUN_CS = baserunCS)

model_4 <- lm(TARGET_WINS ~ TEAM_BATTING_H + TEAM_PITCHING_HR + TEAM_FIELDING_DP + fieldingE + baserunSB + baserunCS, data = trainData_model4) par(mfrow = c(2,2)) plot(model_4) summary(model_4)

7.1 Model Selection and Write Predictions

testData$TARGET_WINS_PRED <- predict(model_1, newdata = testData) testData %>% select(INDEX, TARGET_WINS, TARGET_WINS_PRED) %>% write.csv(., "moneyball-predictions.csv", row.names = F)