DATA 621 Homework 1

Critical Thinking Group 1

September 26, 2021

Contents

Prepared for:

Prof. Dr. Nasrin Khansari

City University of New York, School of Professional Studies - Data 621

DATA 621 – Business Analytics and Data Mining

Prepared by:

Critical Thinking Group 1

Vic Chan

Gehad Gad

Evan McLaughlin

Bruno de Melo

Anjal Hussan

Zhouxin Shi

Overview

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

Objective

The objective is to build a multiple linear regression model on the training data to predict the number of wins for the team. we can only use the variables given to us (or variables that we derive from the variables provided).

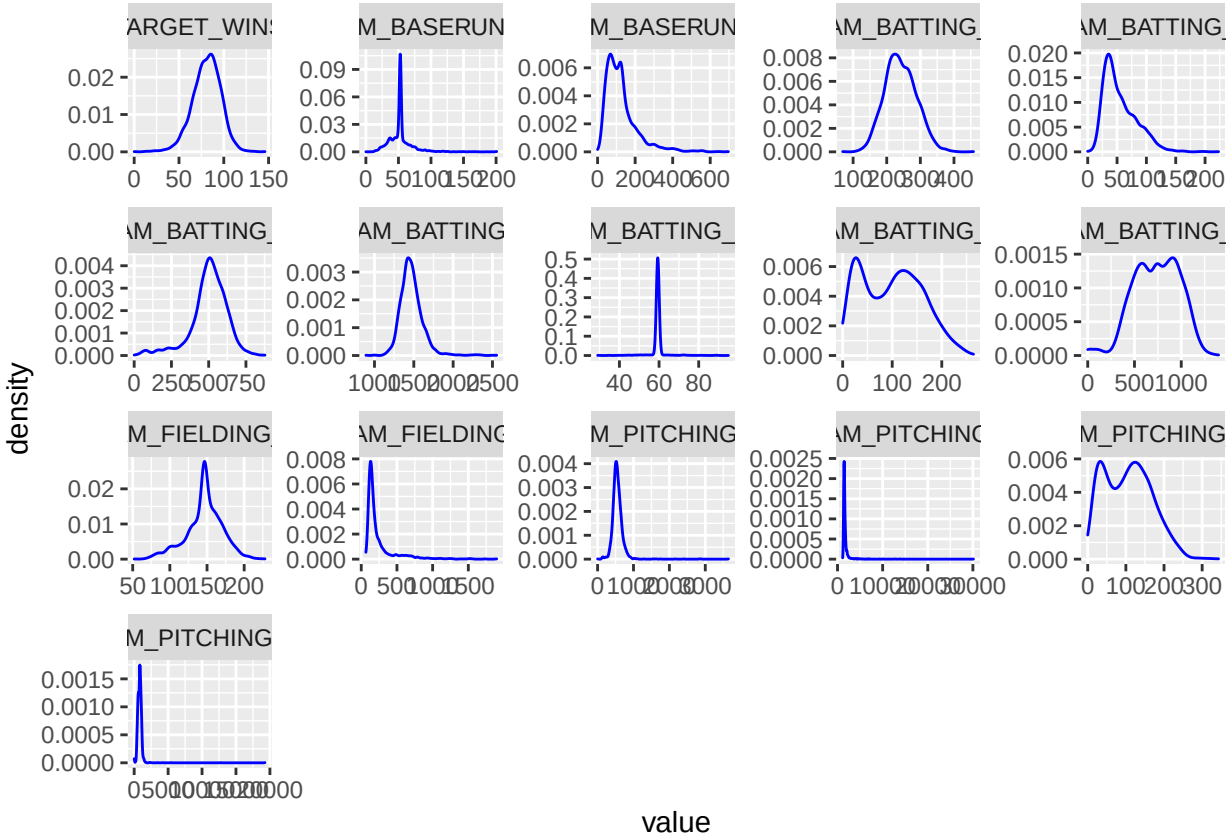
Data Exploration

Data Summary

The dataset contain 17 columns and 2276 observations or records. The first column is the index which will be deleted as it has no use in the analysis. The target variable is the **TARGET_WINS** column. The dataset is all numerical and does include any categorical variables.

At a glance we can see that the data a significant number of NA values and the average wins for a team is about 81.

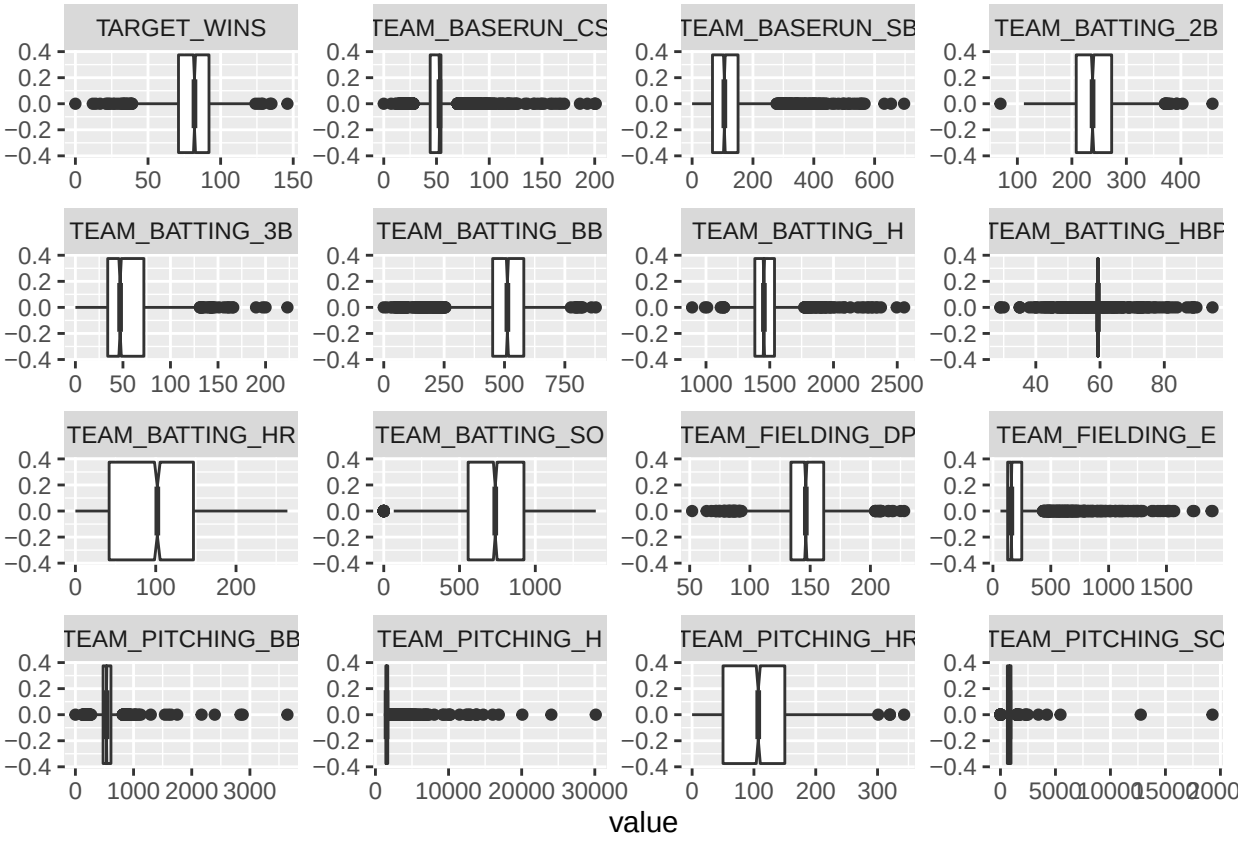
Distribution



The distribution of the target variable **TARGET_WINS** is normally distributed.

The distribution also show that **BASERUN_SB** and **BATTING_3B** are right skewed, and additional **BATTING_HR** and **PITCHING_HR** are bimodal distributions.

Boxplot

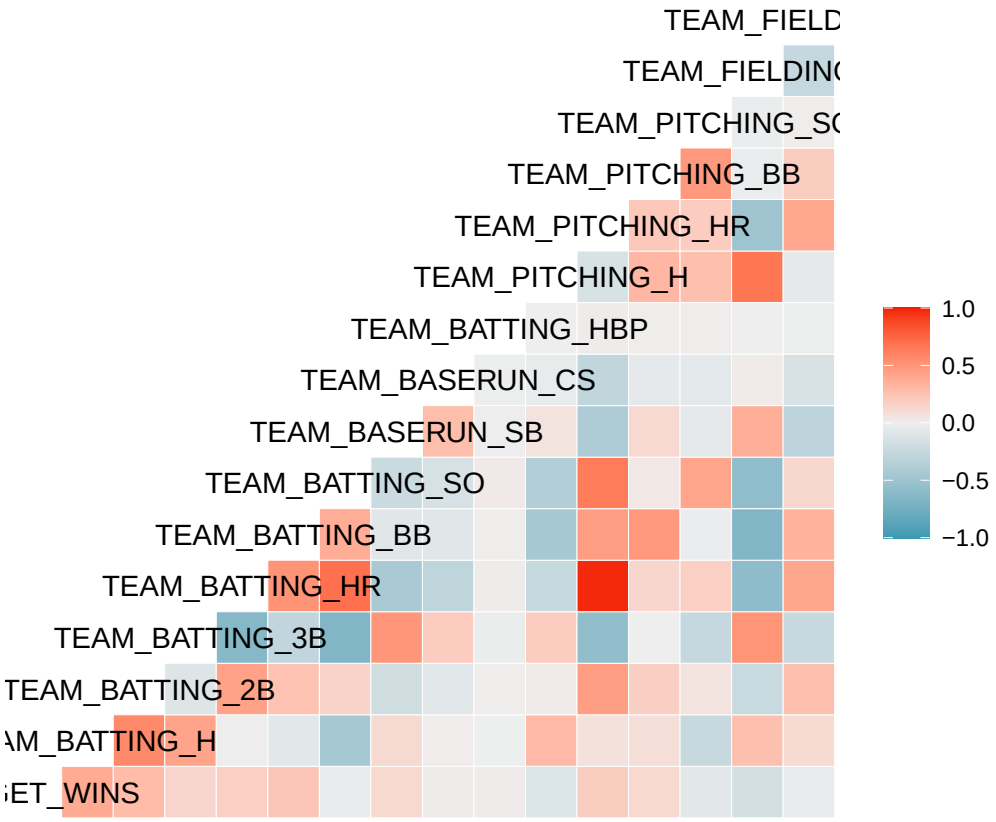


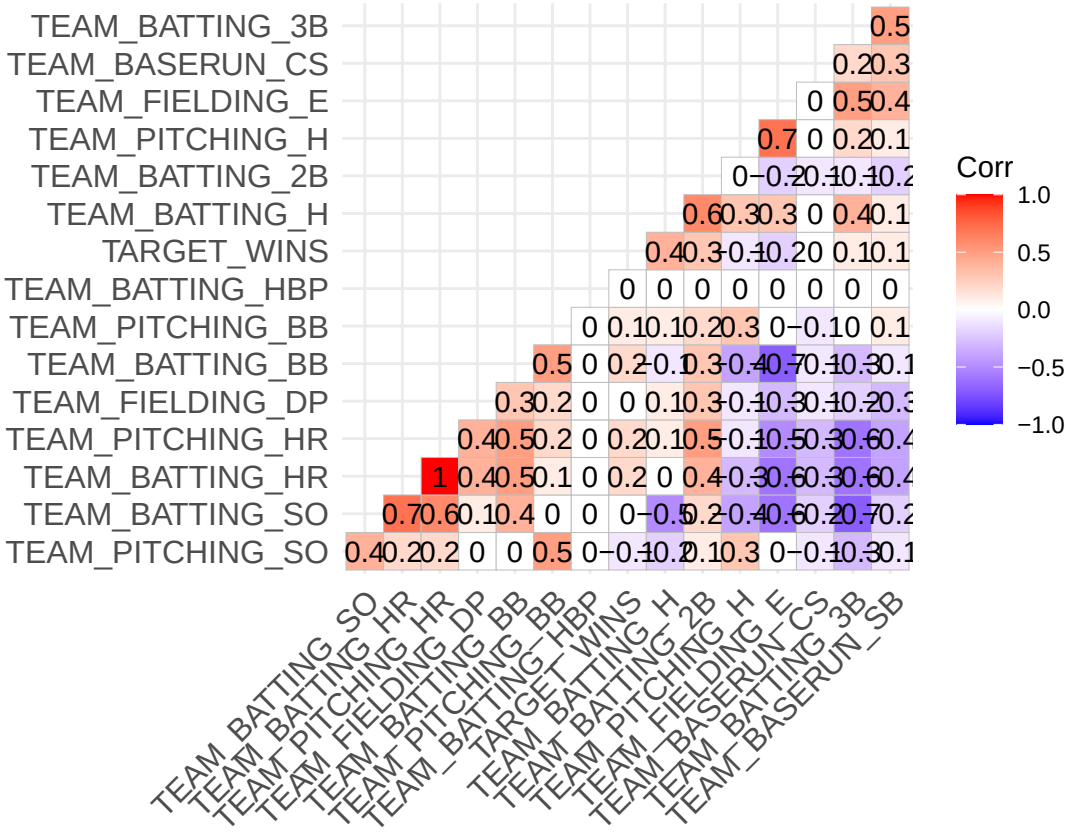
The box-plots above give us idea about the spread of each variable in the data which reveal significant outliers in a lot of the columns.

Correlation

```
## # A tibble: 15 x 2
##    term             TARGET_WINS
##    <chr>                  <dbl>
##  1 TEAM_BATTING_H         0.389
##  2 TEAM_BATTING_2B        0.289
##  3 TEAM_BATTING_3B        0.143
##  4 TEAM_BATTING_HR        0.176
##  5 TEAM_BATTING_BB        0.233
##  6 TEAM_BATTING_SO       -0.0307
##  7 TEAM_BASERUN_SB        0.123
##  8 TEAM_BASERUN_CS        0.0156
##  9 TEAM_BATTING_HBP       0.0163
## 10 TEAM_PITCHING_H       -0.110
## 11 TEAM_PITCHING_HR       0.189
## 12 TEAM_PITCHING_BB       0.124
## 13 TEAM_PITCHING_SO      -0.0758
## 14 TEAM_FIELDING_E       -0.176
```

```
## 15 TEAM_FIELDING_DP     -0.0288
```





These correlations plots do not show a strong relationship between any two variables. This indicate presence of 'noise' in these relationships. It is interesting to note that allowing hits have little positive impacts on wins. It is also noteworthy that pitching strikeouts by batters and hits allowed are negatively correlated with winning. I assume the correlations are affected by the outliers.

Data Preparation

In this section we will be looking at the different ways to prepare the data for modeling. We will show the different steps that we took and the reasoning on why we did certain transformations, replacement and creation of columns.

Finding All NA

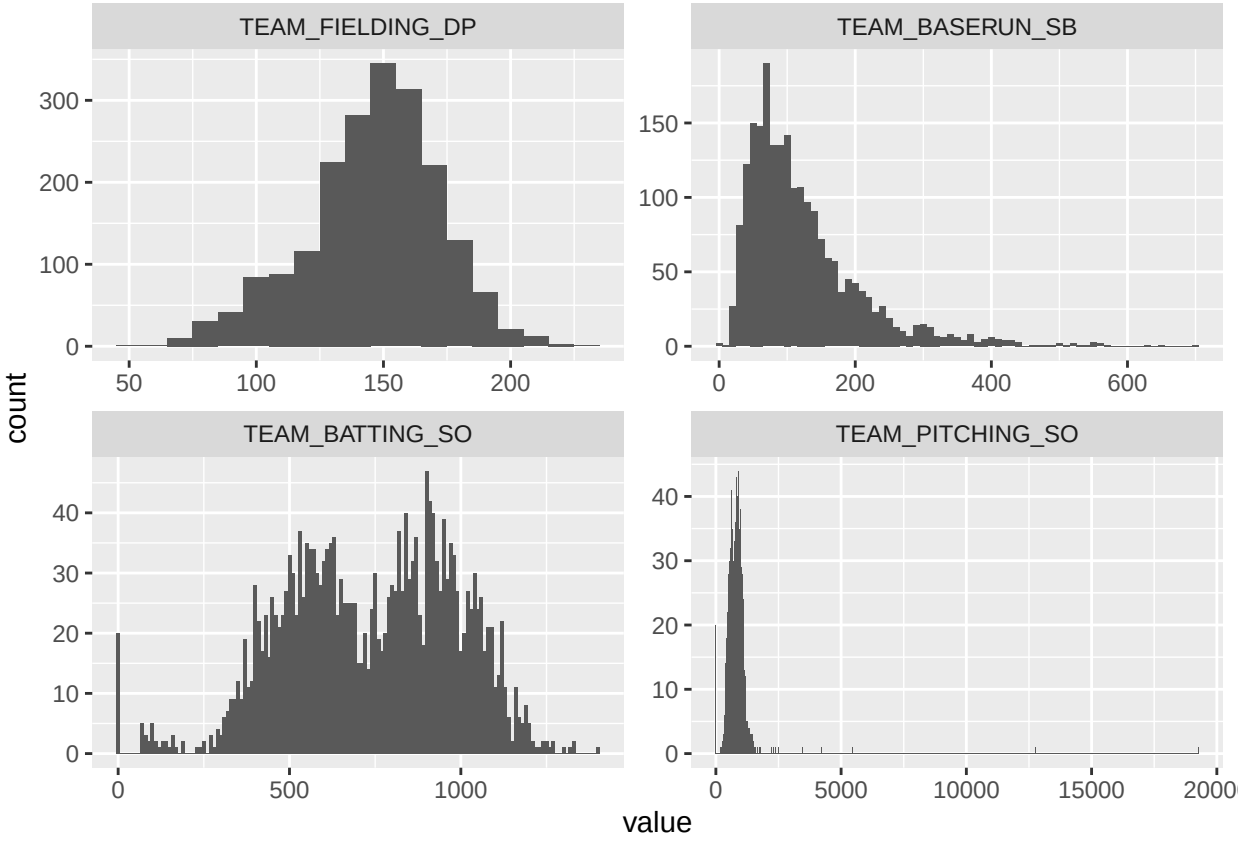
```
##    na_count total_rows percent_missing
## 1      2085       2276      0.91608084
## 2       772       2276      0.33919156
## 3       286       2276      0.12565905
## 4       131       2276      0.05755712
## 5       102       2276      0.04481547
## 6       102       2276      0.04481547
## 7         0       2276      0.00000000
## 8         0       2276      0.00000000
## 9         0       2276      0.00000000
## 10        0       2276      0.00000000
## 11        0       2276      0.00000000
## 12        0       2276      0.00000000
## 13        0       2276      0.00000000
```

```
## 14          0        2276       0.00000000
## 15          0        2276       0.00000000
## 16          0        2276       0.00000000
## 17          0        2276       0.00000000
```

Initially when looking at the data we can see that **TEAM_BATTING_HBP** is missing 91% of its data and **TEAM_BASERUN_CS** is missing around 34% of its data. This is a lot of data missing which is why those columns will be removing these. Based on different studies there is no definite percentage of for how much data one should be missing before removing the column, but it is always better to have more data. The columns **TEAM_FIELDING_DP**, **TEAM_BASERUN_SB**, **TEAM_BATTING_SO**, and **TEAM_PITCHING_SO** are missing around 12% - 4% of its data and can fill those in with using mean and median. In the next section we will look at to see whether using the mean or median would be the better choice in filling the missing data.

Replacing NA with Mean or Median

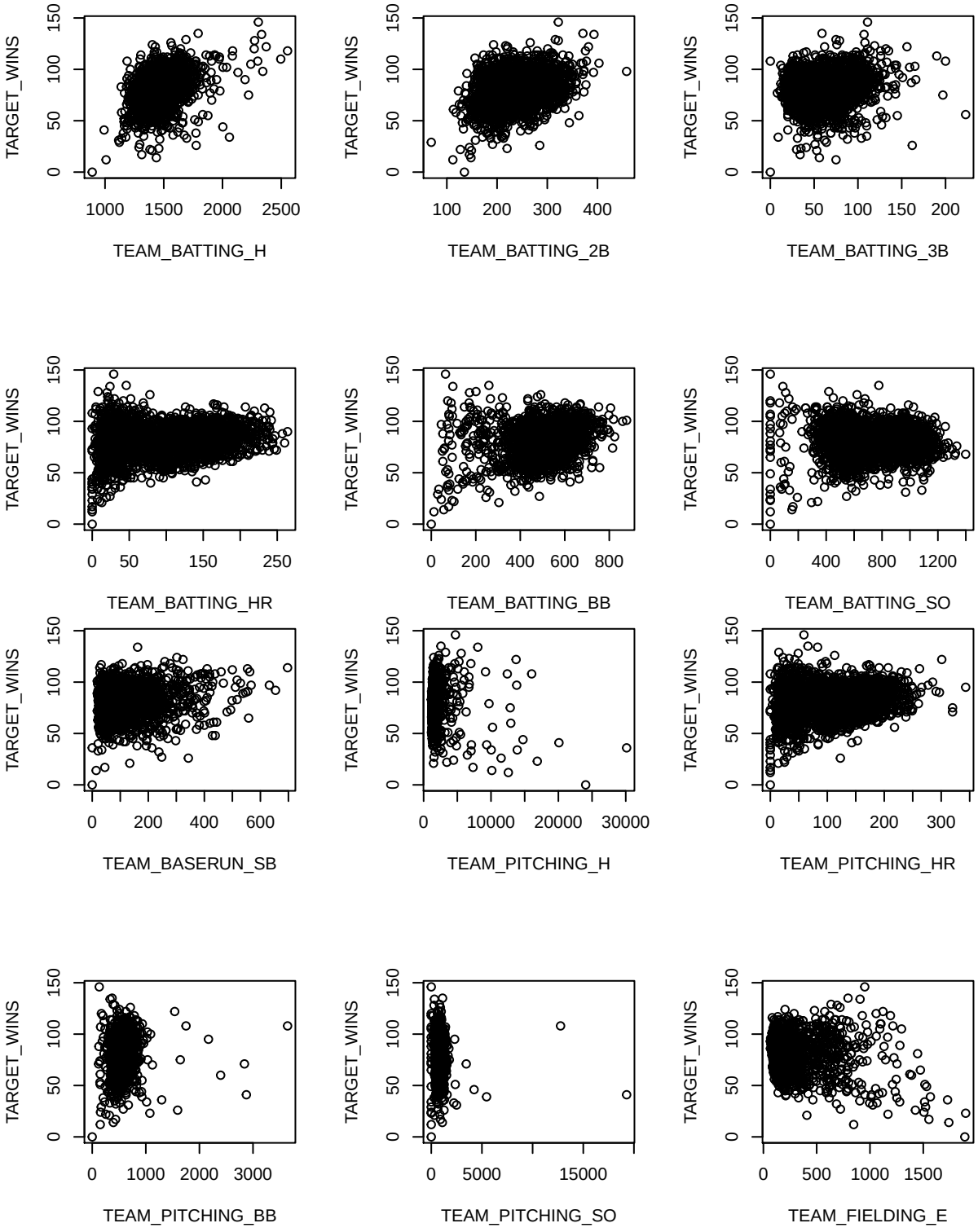
In this section we will need to decide whether to fill the missing data using the mean or median. We will need to look at the distribution of each of the columns with missing data in order to decide if we will be using the median or mean to fill in the missing data.

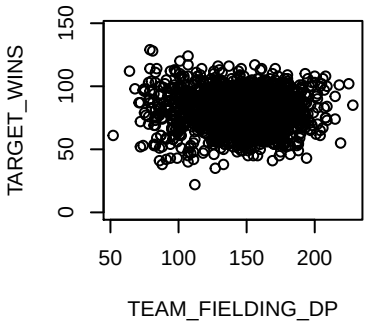


Looking at the above graphs we can see that not all the distribution are uniform distribution. We can see that **TEAM_BATTING_SO** is a bimodal distribution, **TEAM_BASERUN_SB** is skewed to the right, and **TEAM_PITCHING_SO** has very large outliers. For this reason we will be using the median to replace all the missing data as the median is less susceptible to outliers and non-uniform distributions.

Transformation

We will also be needing to check all of the columns to see if they will need any type of transformation in order to create a linear line. We will be be graphing all the columns with **TARGET_WINS** as the response variable. This will allow us to see if there are any columns that can be transformed in order to improve the model.





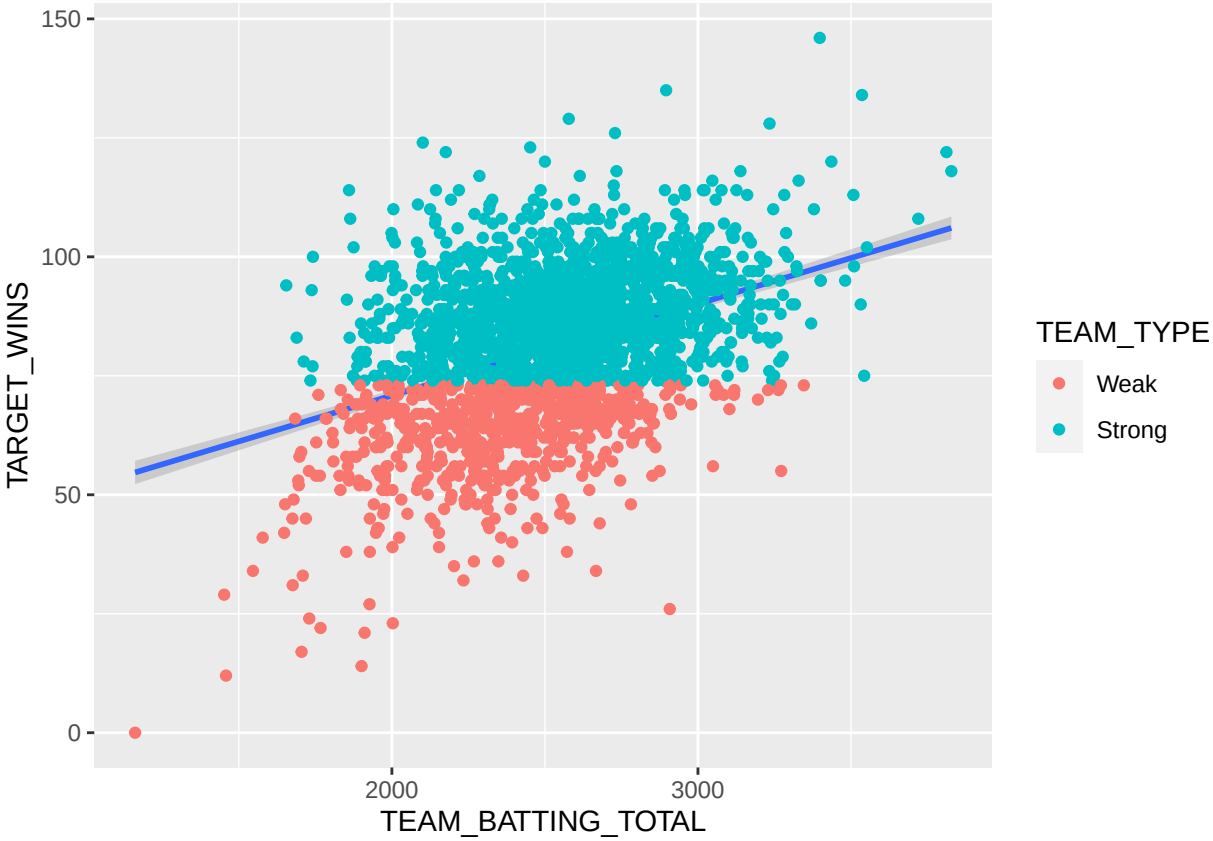
Looking at the graphs above we can see that none of the columns are real good candidates for transformation.

Putting Teams Into Buckets

We will be putting the dataset into buckets based on the teams winning score as this will allow us to see if there is any patterns between weak and strong teams. The teams will be split into two groups **Strong** and **Weak** based on the **TARGET_WINS** column. The maximum **TARGET_WINS** is 146 and the minimum **TARGET_WINS** is 0 therefore the split is 73.

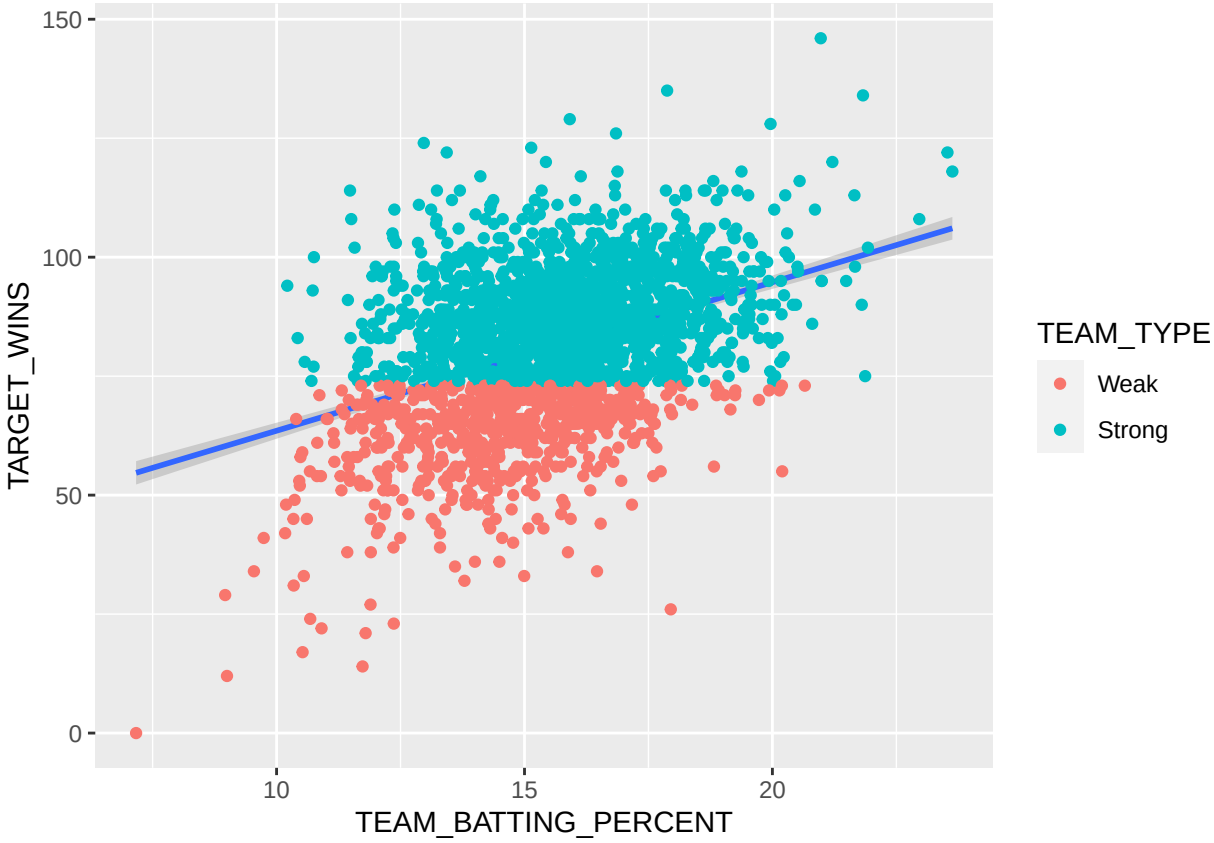
Creating Total Hits

We needed to create a column which include all the different batting statistics. By combining **TEAM_BATTING_H**, **TEAM_BATTING_2B**, **TEAM_BATTING_3B** AND **TEAM_BATTING_HR** we are able to measure the total amount bases each team scored. The reason why we needed this column is because there can be teams that score more home runs than single bases. Combining this information all into one column will make it easier to build a model as we will not need to put as many variables.



Base Percentage

We would like to create a column which measures the total amount of bases a team get per game. This will be calculated by using the new created column **TEAM_BATTING_TOTAL** dividing 162 game season.



Model Building

At the beginning, we were presented with 16 independent variables. It makes sense to exclude index since it is not relevant. It also makes sense to exclude **TEAM_BATTING_HBP** and **TEAM_BASERUN_CS** since they are comprised of so many N/As. We are thus able to concentrate on the 13 remaining variables, pursuing continuous incremental model improvement.

To start with, our first three models are outlined below.

lmodel1 - an "all-in" model that includes all 13 remaining variables
lmodel2 - a model that strips out outliers
lmodel3 - a model that eliminates impertinent attributes

We'll start with the all-in model

```
##
## Call:
## lm(formula = target_wins ~ ., data = train1)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -49.827  -8.580   0.103   8.432  58.544
##
## Coefficients: (2 not defined because of singularities)
```

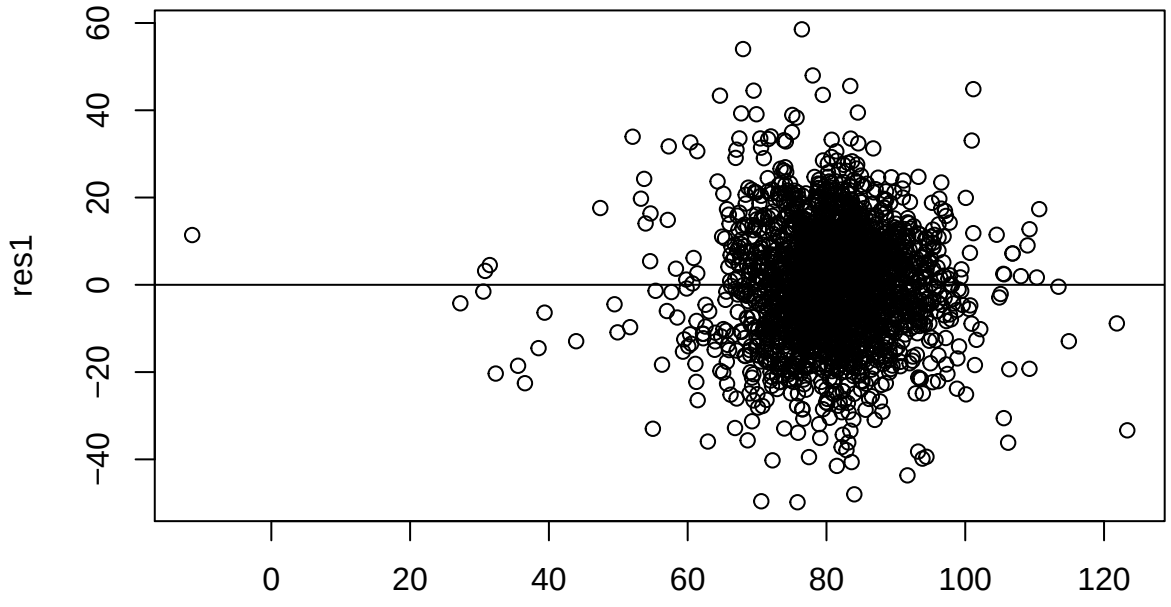
```
##                          Estimate Std. Error t value Pr(>|t|)
## (Intercept)          22.9775583  5.3046349   4.332 1.54e-05 ***
## team_batting_h        0.0488787  0.0036941  13.232  < 2e-16 ***
## team_batting_2b      -0.0212136  0.0091699  -2.313 0.020791 *
## team_batting_3b       0.0649302  0.0167897   3.867 0.000113 ***
## team_batting_hr       0.0545602  0.0273630   1.994 0.046279 *
## team_batting_bb       0.0105502  0.0058352   1.808 0.070734 .
## team_batting_so      -0.0084176  0.0025457  -3.307 0.000959 ***
## team_baserun_sb       0.0247806  0.0042572   5.821 6.69e-09 ***
## team_pitching_h      -0.0008598  0.0003668  -2.344 0.019147 *
## team_pitching_hr      0.0123395  0.0243703   0.506 0.612672
## team_pitching_bb      0.0008863  0.0041539   0.213 0.831065
## team_pitching_so      0.0028087  0.0009218   3.047 0.002338 **
## team_fielding_e      -0.0191590  0.0024016  -7.978 2.35e-15 ***
## team_fielding_dp     -0.1219877  0.0129372  -9.429  < 2e-16 ***
## team_batting_total           NA         NA      NA       NA
## team_batting_percent         NA         NA      NA       NA
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 13.07 on 2262 degrees of freedom
## Multiple R-squared:  0.3152, Adjusted R-squared:  0.3113
## F-statistic:  80.1 on 13 and 2262 DF,  p-value: < 2.2e-16
```

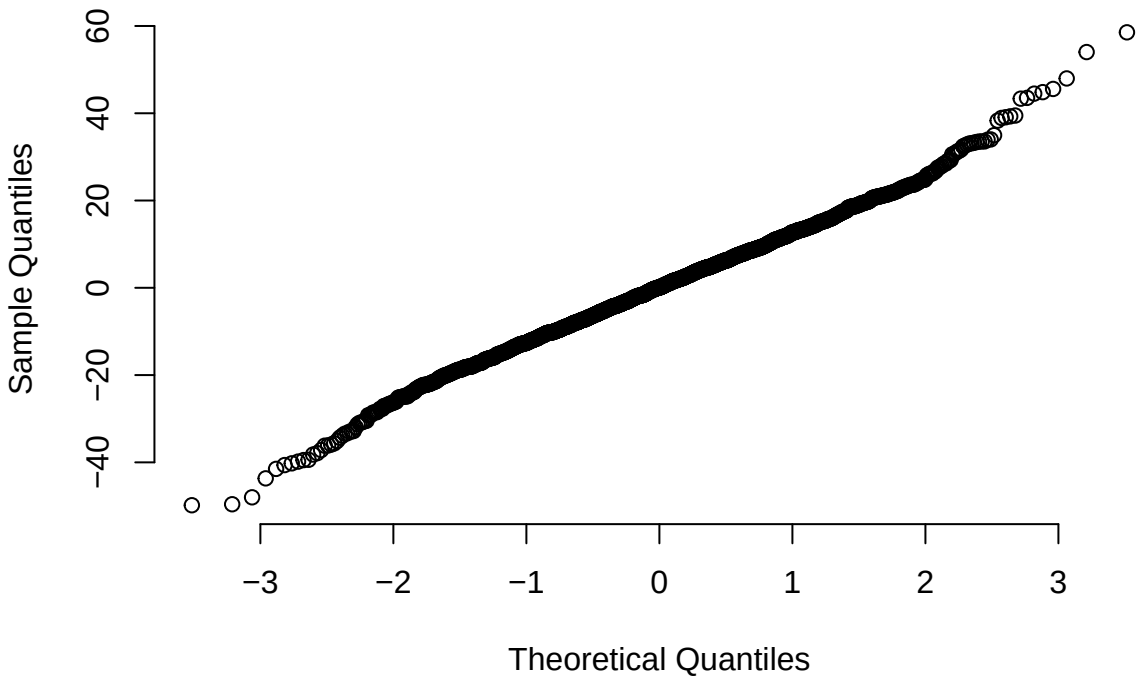
So in looking at the all-in model, we can identify how the model behaves intuitively and not-so-intuitively. For example, we see the following variables as having positive coefficients: **TEAM_BATTING_H**, **TEAM_BATTING_3B**, **TEAM_BASERUN_SB**, and **BEAM_PITCH_STRIKEOUT**. These make sense, as you'd expect a team to win games that gets hits, hits triples, steals bases efficiently, and strikes out opponents. However, some of the positive coefficients don't make as much sense. For example, we would expect teams whose pitchers give up lots of home runs to not win very many games. This certainly warrants further analysis.

For negative coefficients, we'd obviously expect teams whose players make a lot of errors to not win at a high rate. However, hitting doubles and fielding double plays have negative coefficients as well, which are not intuitive at all.

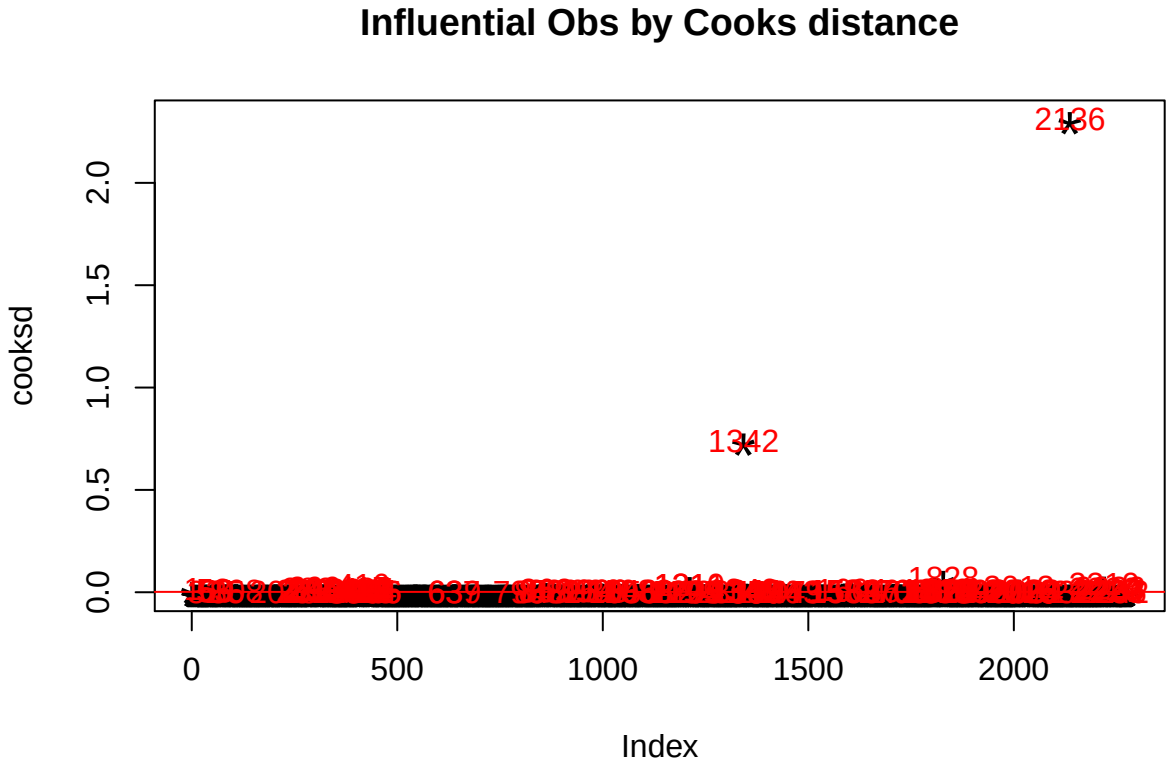
A majority of the variables that we are assessing appear to contribute to predicting wins. We can gain some comfort in our model due to the low RSE (13.07) and satisfactory F-statistic (80.1), and we should feel ok about the overall efficacy of our model. However, the Adjusted R-square well under 1 is cause for some concern, but we can look to improve that in future iterations of the model.

What else can we do to improve our model? Well, its predictive value might be enhanced by eliminating some problematic outliers. So let's take a look at if it makes sense to do so.





The data is not evenly scattered but we don't detect any unexpected non-linear pattern. The normal QQ looks good as well with a relatively straight line. We can spot some outliers that we should drill down on using Cook's Distance. Then, we can then attempt to strip them out to improve our model somewhat.



We can spot two that breach our threshold, so now we set about removing them. Next, we can re-run our initial all-in model to see if dropping the outliers has any impact on improving the model.

```
##
## Call:
## lm(formula = target_wins ~ ., data = train1_strip)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -36.469  -7.796   0.246   7.405  34.488
##
## Coefficients: (2 not defined because of singularities)
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)          30.440842   4.948518   6.152 9.13e-10 ***
## team_batting_h        0.031941   0.003887   8.218 3.55e-16 ***
## team_batting_2b      -0.038693   0.008314  -4.654 3.46e-06 ***
## team_batting_3b       0.110678   0.016304   6.788 1.46e-11 ***
## team_batting_hr       0.079148   0.043669   1.812   0.0701 .
## team_batting_bb       0.099555   0.012268   8.115 8.08e-16 ***
## team_batting_so      -0.047069   0.005498  -8.561  < 2e-16 ***
## team_baserun_sb       0.050541   0.004185  12.076  < 2e-16 ***
## team_pitching_h       0.008201   0.001069   7.670 2.59e-14 ***
## team_pitching_hr      0.008620   0.040704   0.212   0.8323
## team_pitching_bb     -0.069371   0.010792  -6.428 1.59e-10 ***
## team_pitching_so      0.034187   0.004537   7.536 7.13e-14 ***
## team_fielding_e      -0.040194   0.003121 -12.880  < 2e-16 ***
## team_fielding_dp     -0.119672   0.011334 -10.559  < 2e-16 ***
## team_batting_total          NA         NA      NA       NA
## team_batting_percent        NA         NA      NA       NA
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
##
## Residual standard error: 10.98 on 2140 degrees of freedom
## Multiple R-squared:  0.3925, Adjusted R-squared:  0.3888
## F-statistic: 106.4 on 13 and 2140 DF,  p-value: < 2.2e-16
```

This looks like good news. Our RSE is down, and our F-statistic is up. Even our Adjusted R-Squared value is up slightly from .31. Nevertheless, the explanatory value of our model remains limited without this last number increasing significantly. And we can clearly see some variables with high p-values that ought to be removed in order to improve our model. Let's proceed with removing `team_batting_hr` and `team_picthing_hr`.

```
##
## Call:
## lm(formula = target_wins ~ ., data = train3)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -36.462  -7.810   0.229   7.392  34.504
##
## Coefficients: (1 not defined because of singularities)
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)          30.418573   4.946297   6.150 9.23e-10 ***
## team_batting_h        0.010015   0.005093   1.966   0.0494 *
## team_batting_2b      -0.082926   0.009262  -8.953  < 2e-16 ***
## team_batting_3b       0.044487   0.015645   2.844   0.0045 **
## team_batting_bb       0.098449   0.011100   8.869  < 2e-16 ***
## team_batting_so      -0.047460   0.005179  -9.164  < 2e-16 ***
## team_baserun_sb       0.050534   0.004184  12.077  < 2e-16 ***
## team_pitching_h       0.008148   0.001039   7.842 6.92e-15 ***
## team_pitching_bb     -0.068337   0.009620  -7.103 1.65e-12 ***
## team_pitching_so      0.034524   0.004248   8.127 7.36e-16 ***
## team_fielding_e      -0.040318   0.003064 -13.159  < 2e-16 ***
## team_fielding_dp     -0.119690   0.011331 -10.563  < 2e-16 ***
## team_batting_total    0.022054   0.002145  10.282  < 2e-16 ***
## team_batting_percent        NA         NA      NA       NA
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 10.98 on 2141 degrees of freedom
## Multiple R-squared:  0.3925, Adjusted R-squared:  0.3891
## F-statistic: 115.3 on 12 and 2141 DF,  p-value: < 2.2e-16
```

We've improved the model incrementally by removing variables with high p-values, and our RSE and F-stat look better The explanatory power of our model, however, remains in doubt due to the Adjusted R-Squared value that remains low, even though it's improved from the previous model. What stands out here is that triples hit, bases stolen, and gaining walks remain the overall strongest positive coefficients, while `team_fielding_dp` remains the largest negative coefficient, which is counter-intuitive at first blush. However, one thing necessary for a double play is at least one opponent runner on base. Those teams that earn a high number of double plays are only able to do so because their pitchers are allowing runners on base to begin with.

In addition to the aforementioned three models, we will be analysing two additional models as follows:

`1: lmodel4 - model containing three variables which we believe are more related to winning games: 'team_batting_total` + `team_pitching_h` + `team_batting_3b`

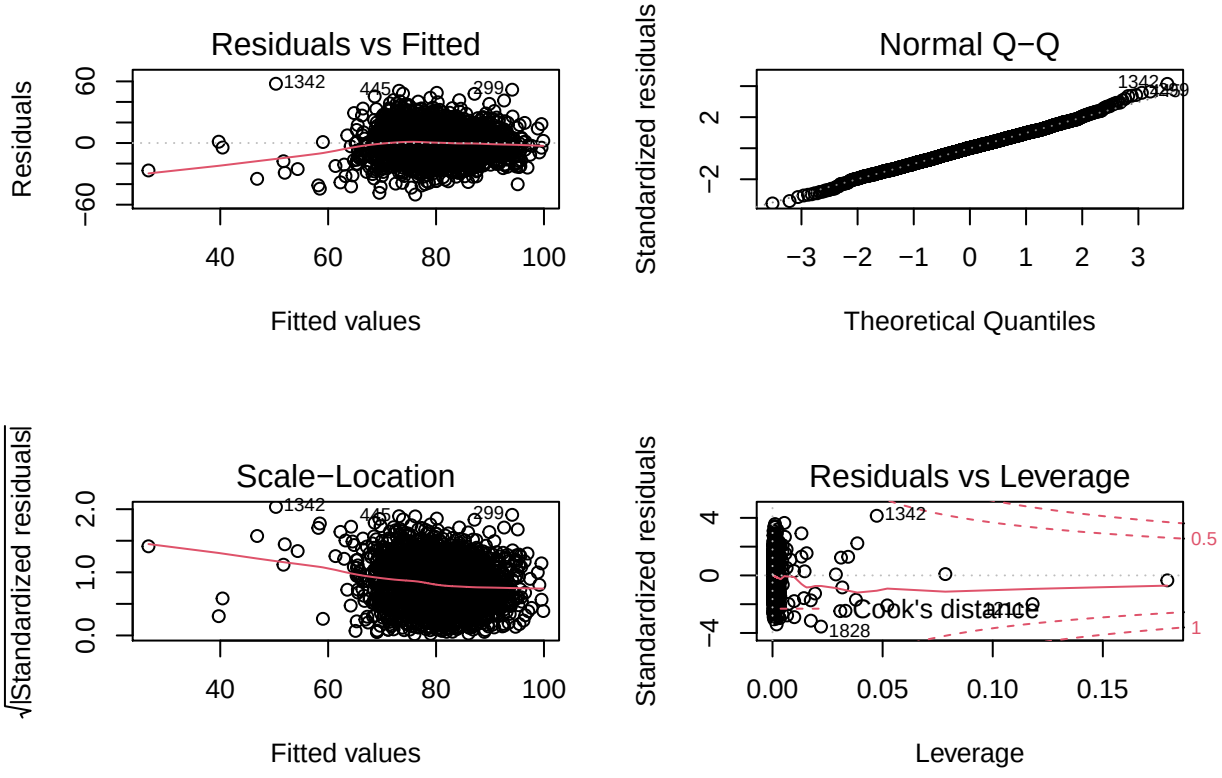
2: lmodel5 - a model put together by someone who does not understand the game of baseball;

We will fit a model on two variables which we believe are more related to winning games: 'team_batting_total+team_pitching_h

```
##
## Call:
## lm(formula = target_wins ~ team_batting_total + team_pitching_h,
##     data = train1)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -50.147  -9.644   0.310   9.452  57.678
##
## Coefficients:
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)         34.4849847  2.2895718  15.062  < 2e-16 ***
## team_batting_total   0.0192922  0.0008904  21.667  < 2e-16 ***
## team_pitching_h     -0.0012548  0.0002125  -5.904 4.07e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 14.26 on 2273 degrees of freedom
## Multiple R-squared:  0.1812, Adjusted R-squared:  0.1805
## F-statistic: 251.5 on 2 and 2273 DF,  p-value: < 2.2e-16
```

Adjusted R-squared is lower then what we have in model3. As for interpretation, batting total has a positive impact on wins while hits allowed have a negative impact, as expected.

Residual plots are below:



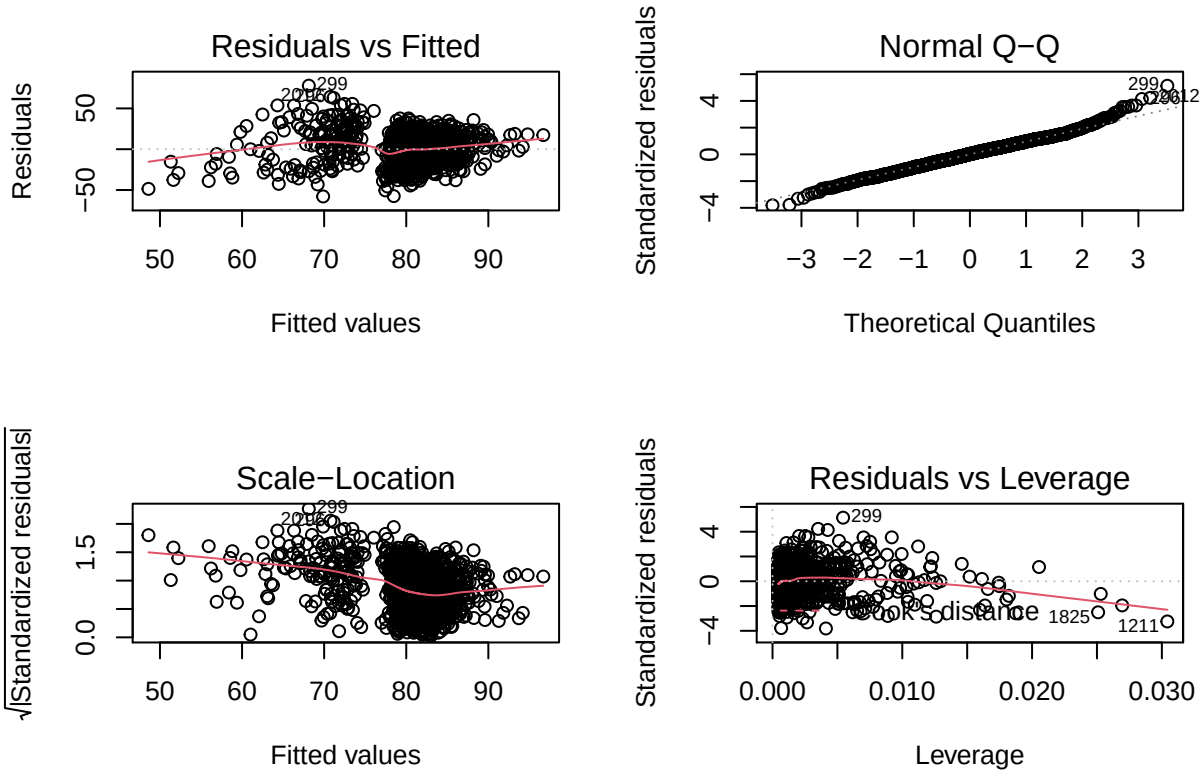
Now we will do a random model. The subject matter *non-expert* does not understand batters, pitchers, walks,

double plays, etc. What (s)he knows is that in every game, not losing is related to a low number of errors - so the variable 'team_fielding_e' is picked and also (s)he thinks that stolen is not good so the variable `team_baserun_sb` is also chosen .

```
##
## Call:
## lm(formula = target_wins ~ team_baserun_sb + team_fielding_e,
##     data = train1)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -57.878  -9.807   0.398   9.827  77.867
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)     80.312320   0.597172 134.488   <2e-16 ***
## team_baserun_sb  0.037393   0.003951   9.464   <2e-16 ***
## team_fielding_e -0.016778   0.001481 -11.325   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 15.21 on 2273 degrees of freedom
## Multiple R-squared:  0.06788,    Adjusted R-squared:  0.06706
## F-statistic: 82.76 on 2 and 2273 DF,  p-value: < 2.2e-16
```

As expected, such a random model does not yield very good results, despite all coefficients being statistically different from zero. As we can see from the very low R-squared and high RSE. In terms of interpretation, number of stolen bases have a positive impact on winnings, contradicting what the *non-expert* thought and number of errors has a negative impact on winnings, which is intuitive, as expected by our *non-expert*.

In addition, analyzing the residual plots we see that the residuals are not normally distributed and present heterocedasticity.



Select Models

Model comparison

Now, before we select our model, let's find out the statistics of our models so that we can compare the models using the indicators i.e. R2, MSE, F-statistic(f), Number of Variables (k), Number of Observations (n)

name	rsquared	adjustedR2	mse	f	pvalue	k	n
model1	0.3152383	0.3113029	169.8355	80.10299	0.0247806	13	2262
model2	0.3924991	0.3888087	119.7359	106.35604	0.0505407	13	2140
model3	0.3924864	0.3890813	119.7384	115.26673	0.0081481	12	2141
model4	0.1811960	0.1804756	203.0808	251.50011	21.6667731	2	2273
model5	0.0678773	0.0670572	231.1863	82.76013	9.4640368	2	2273

In order to select on model we need to consider few aspects of the models:

R-squared:

R-squared evaluates the scatter of the data points around the fitted regression line. It is also called the coefficient of determination. For the same data set, higher R-squared values represent smaller differences between the observed data and the fitted values.

R-squared is the percentage of the dependent variable variation that a linear model explains.

R-squared is always between 0 and 100%. 0% represents a model that does not explain any of the variation in the response variable around its mean. The mean of the dependent variable predicts the dependent variable as well as the regression model. 100% represents a model that explains all the variation in the response variable around its mean. Usually, the larger the R2, the better the regression model fits your observations.

Among the models, the r-Squared of Model2 and Model3 are almost similar and has highest R2. Based on R2, we can select model2 and model3 as our best model candidate.

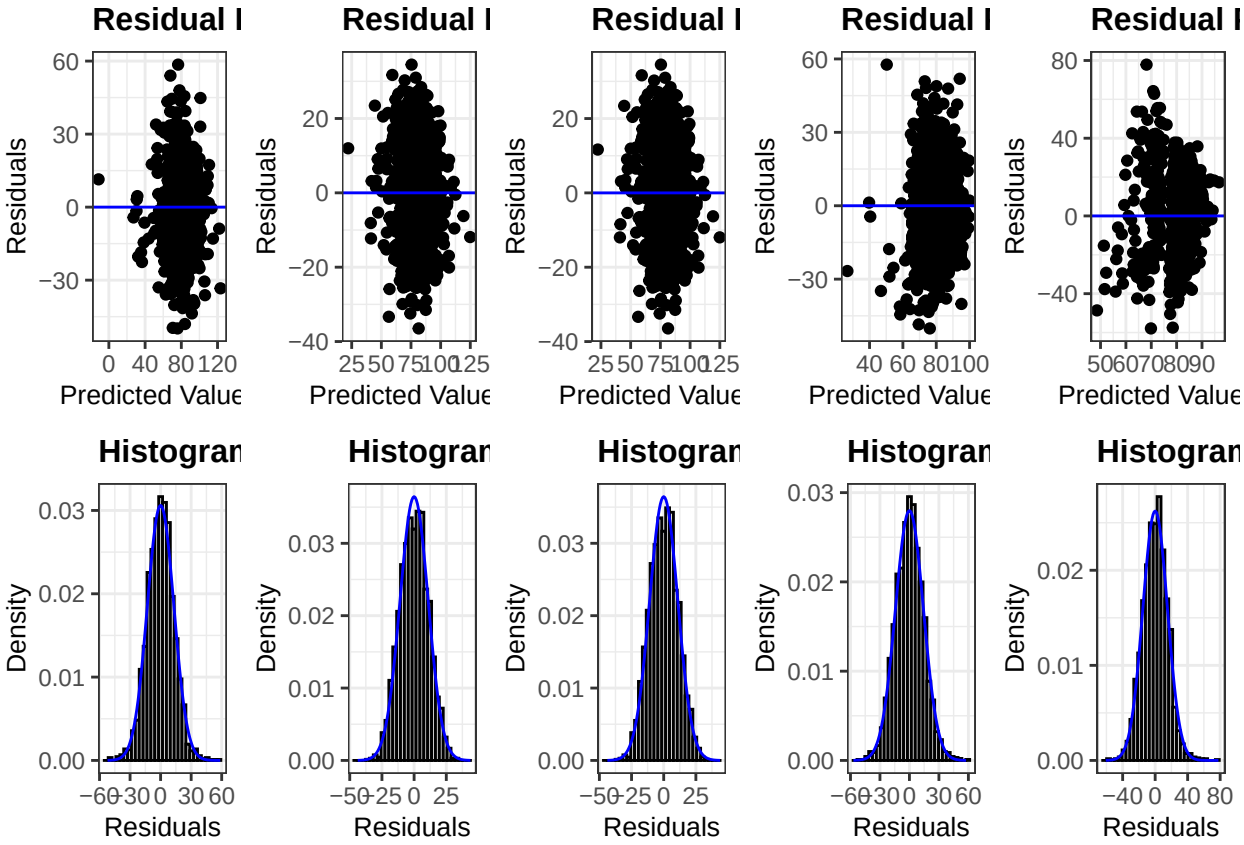
Adjusted R-squared:

The adjusted R-squared is a modified version of R-squared that adjusts for predictors. when we compare Adj-R-Squared among the input variables, the lower adjusted R-squared indicates that the additional input variables are not adding value to the model. A higher adjusted R-squared indicates that the changes in input variables are adding value to the model. Among the models, model3 has the highest adjusted R2 which indicates that model3 could be the best model. The model4 and model5 have the lowest adjusted R2 compare to the first three models.

Residuals:

Residuals are estimates of experimental error obtained by subtracting the observed responses from the predicted responses. The predicted response is calculated from the chosen model. Since this is a form of error, the same general assumptions apply to the group of residuals that we typically use for errors in general: one expects them to be (roughly) normal and (approximately) independently distributed with a mean of 0 and some constant variance.in other words, we should not see any pattern in the residuals when plotting. A simple plot is suitable for displaying the normality of the distribution of a group of residuals

Using ggResidpanel package, we can quickly visualize residuals from all models. In this visuals, the first plot is for model1, second plot is for model2 and so on.



From the plots and histogram, it looks like the residuals from model2 and model3 have no obvious pattern. residuals from model4 and model 5 are left skewed.Hence we can select model2 and model3 as the candidate for the best model

p-value:

we need to evaluate p-value and if the p-value is much less than 0.05 then we can reject the null hypothesis. That will indicate that there is a significant relationship between the variables in the linear regression model of the data set. Based on the p-value shown in the table, we can pick model1, model2, and model3 as best model candidate. p-value of the model4 and model5 are not significant. Hence we can not select model4 and model5 as our best model

Although model2 and model3 are the best candidates. Both models have highest R2 and increasing Adjusted R2. p-value for both of the models are significant. But we are going to select model3 as our final model based on the statistics and diagnostics especially adjusted R2 which is slightly higher and the p-value which is low in comparison with model2

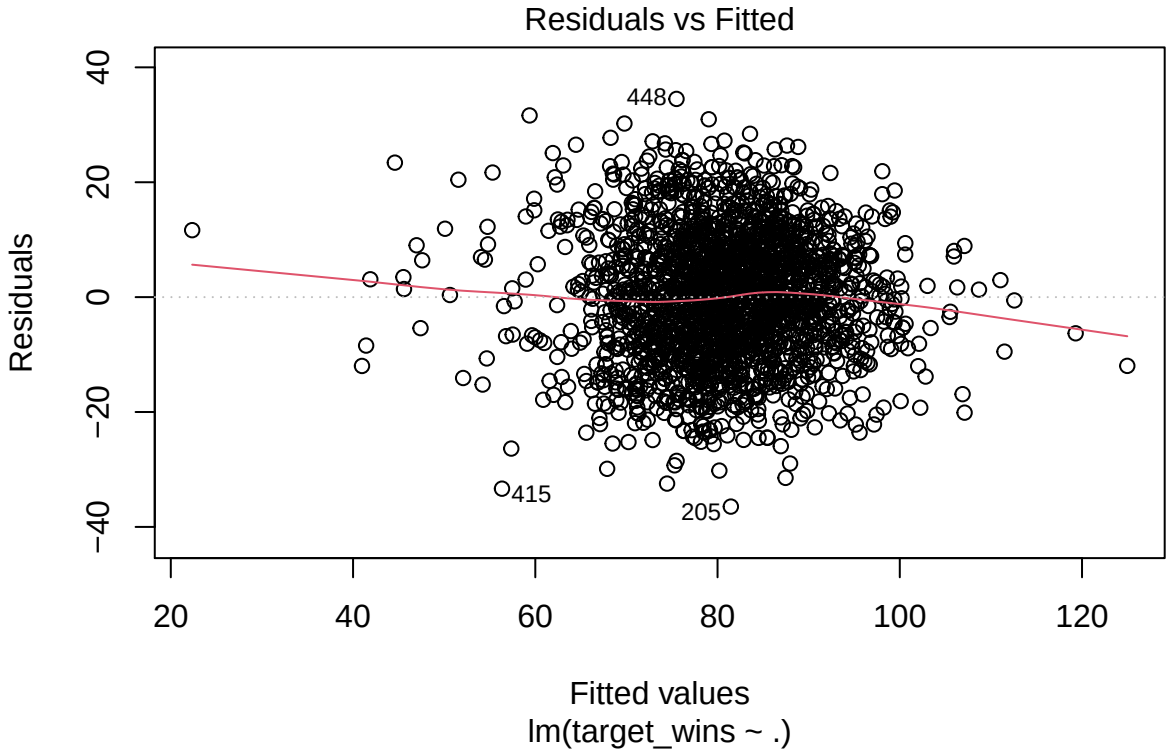
Final Model Review

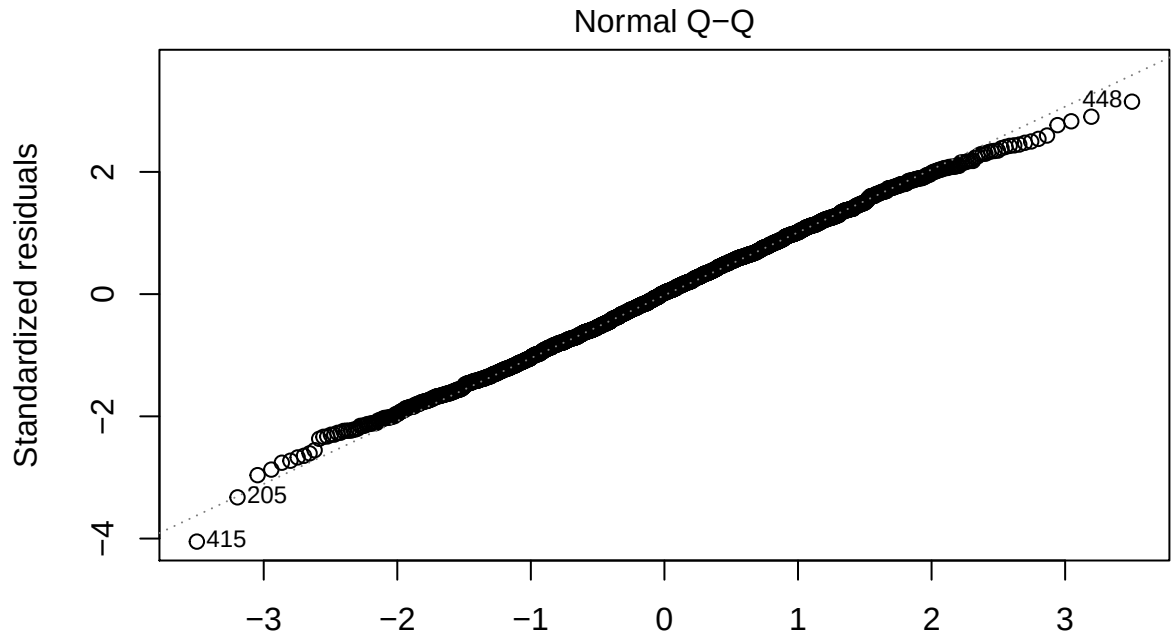
Now that we have selected model3 as our best model, let's take a look to some other aspects of our final model.

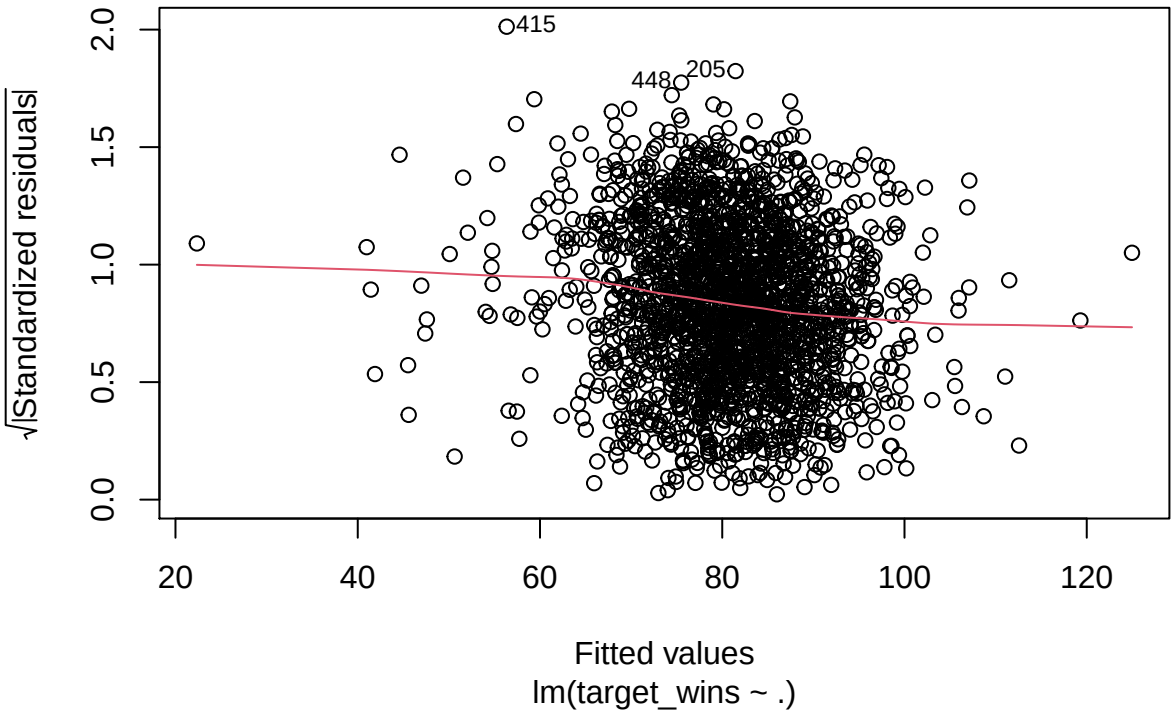
The dependents columns of our selected model are:

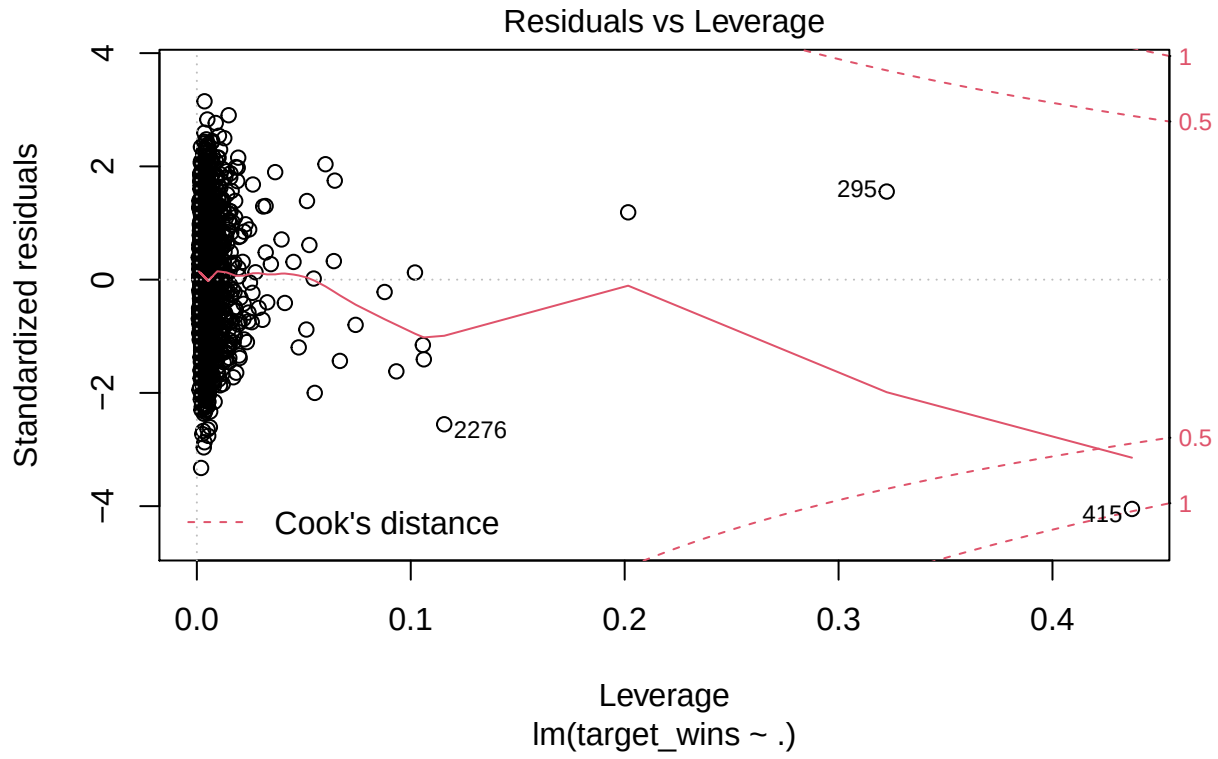
```
##  [1] "target_wins"          "team_batting_h"       "team_batting_2b"
##  [4] "team_batting_3b"      "team_batting_bb"      "team_batting_so"
##  [7] "team_baserun_sb"      "team_pitching_h"      "team_pitching_bb"
## [10] "team_pitching_so"     "team_fielding_e"      "team_fielding_dp"
## [13] "team_batting_total"   "team_batting_percent"
```

Let's review the diagnostic plots and a plot of the residuals.

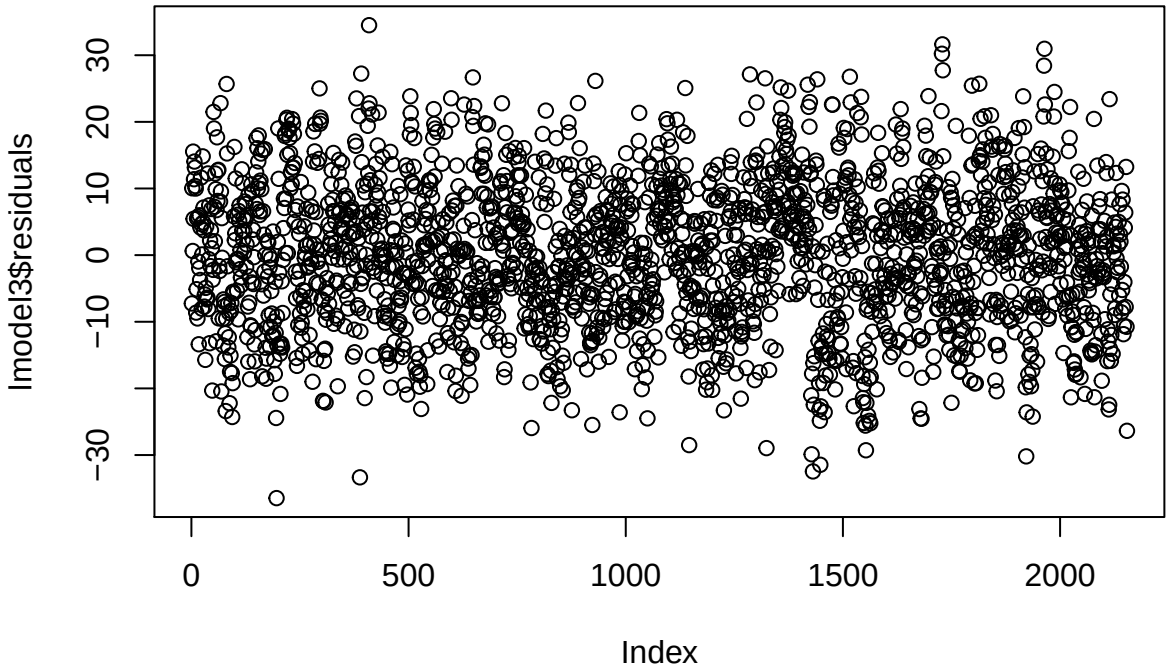



Normal Q−Q
Standardized residuals
Theoretical Quantiles
lm(target_wins ~ .)
448
205
415
−3
−2
−1
0
1
2
3
−4
−2
0
2


Scale−Location
√|Standardized residuals|
Fitted values
lm(target_wins ~ .)
415
448
205
20
40
60
80
100
120
0.0
0.5
1.0
1.5
2.0

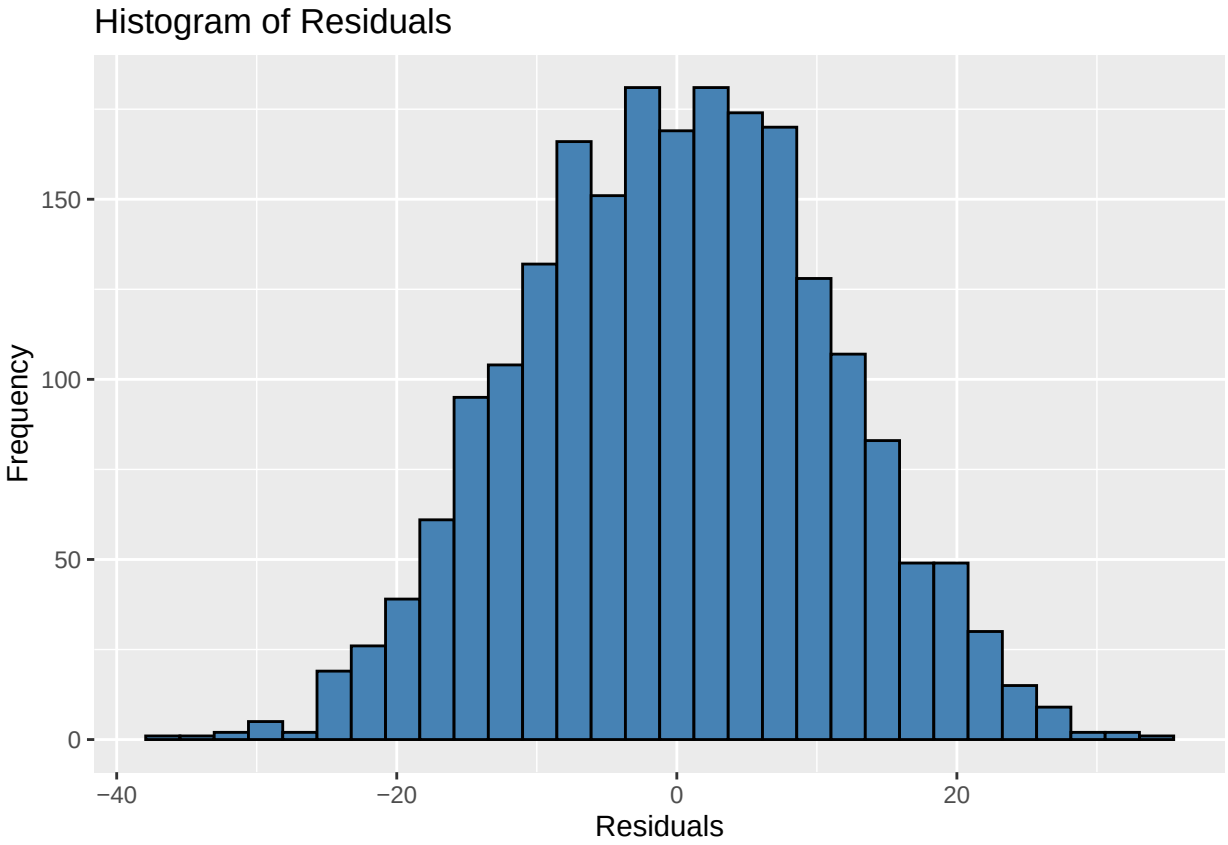


Plot the residual of selected model



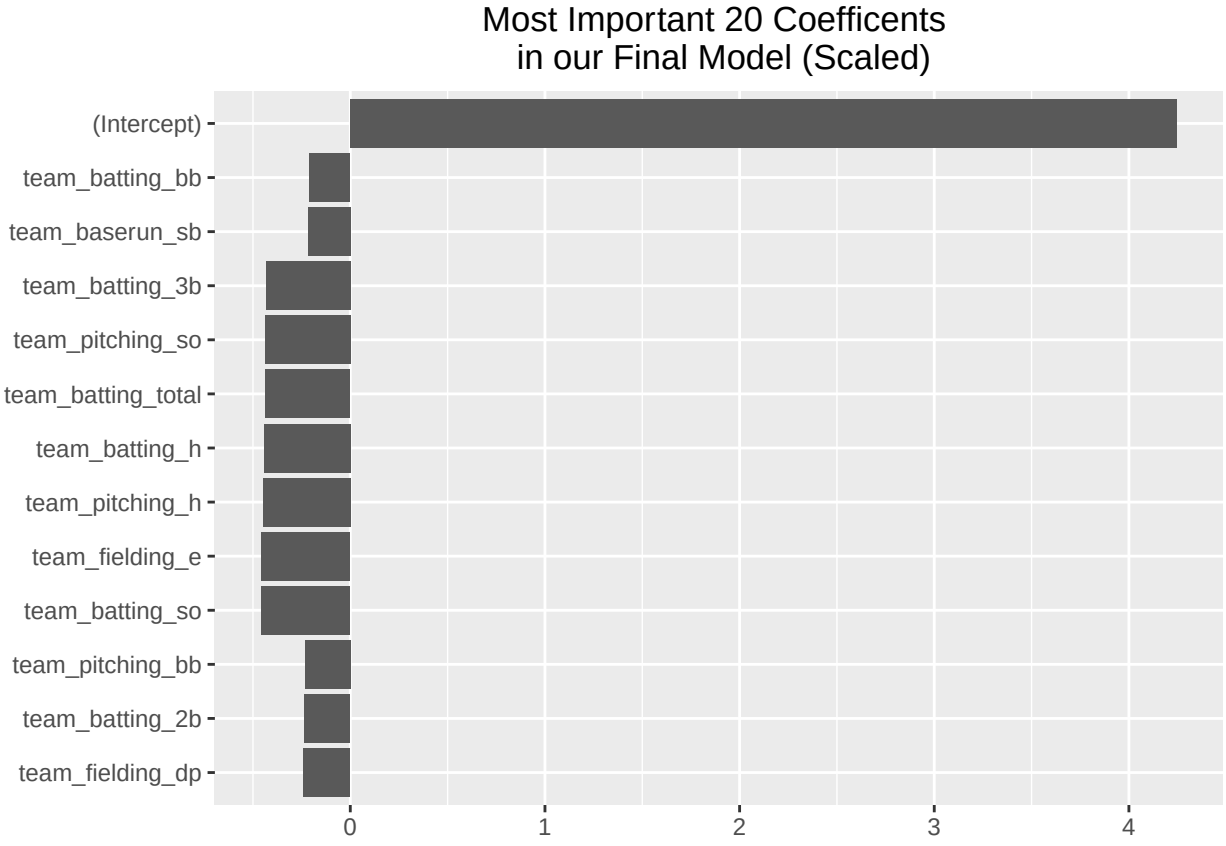
The residual plot looks perfect and there is no obvious pattern noticed in this plot

Create histogram of the residuals of selected model



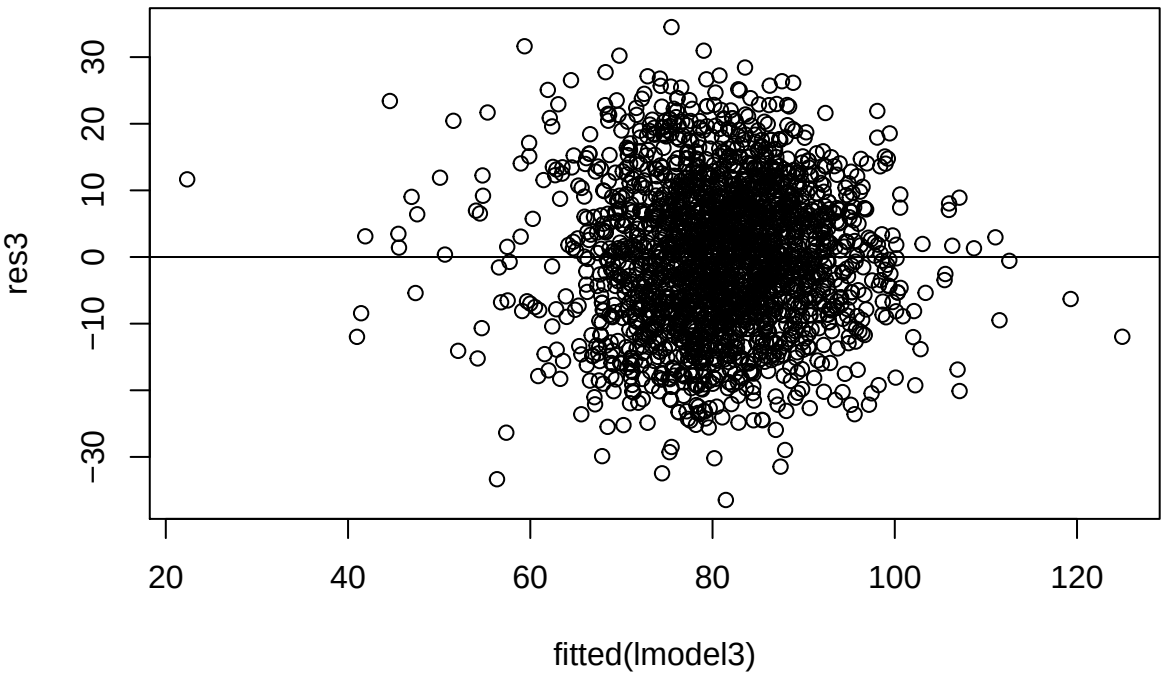
From this histogram we can visualize that the distribution is normal.

Plot the top Coefficients of our model



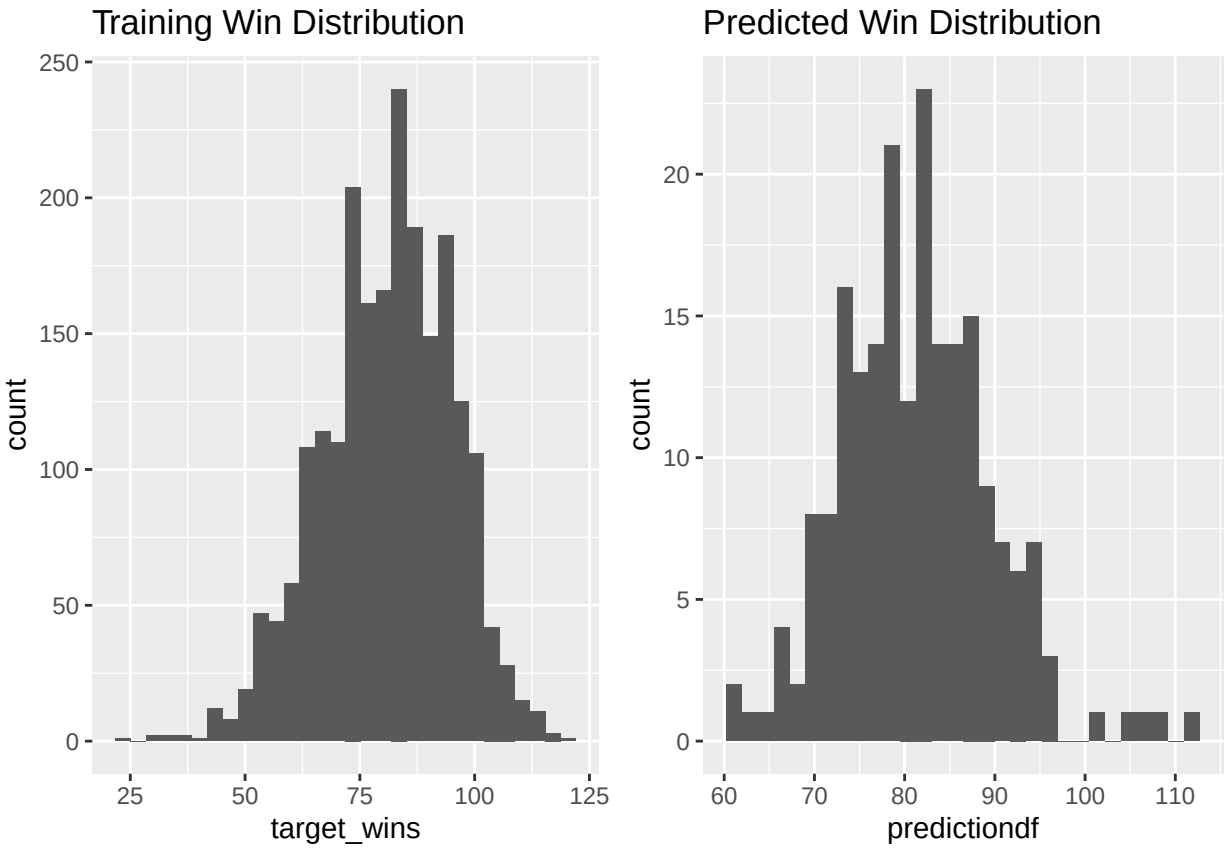
Predictions using evaluation data

Let's use our evaluation data to predict and evaluate our selected model.



Similar to the training data, the evaluation data also needs some prep work.The NA values imputed used the same method as what we done before. We successfully predicted the number of wins with our selected model3 which has a higher R2 and F-statistic. But the residuals are unevenly dispersed relative to the fitted values, indicating that the variance of the residuals is not constant. And the model lacks a higher order term for one variable to explain the curvature.

Compare predicted to original distribution



The Training win distribution and predicted win distribution look similar.

Appendix

- For full output code visit: https://github.com/ahussan/DATA_621_Group1/blob/main/HW1/HW1.Rmd
- For predicted values over test set visit: https://github.com/ahussan/DATA_621_Group1/blob/main/HW1/testPredictions.csv