HMW 1- Data 621

OMER OZEREN

INTRODUCTION

I have been given a dataset with 2276 records summarizing a major league baseball team's season. All statistics have been adjusted to match the performance of a 162 game season. 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. The objective is to build a linear regression model to predict the number of wins for a team.

This report covers an attempt to build a model to predict number of wins of a baseball team in a season based on several offensive and deffensive statistics. Resulting model explained about 39% of variability in the target variable and included most of the provided explanatory variables. Some potentially variables were not included in the data set due to missing values.I used KNN for variable missing values imputtion.

DATA EXPLORATION

Each record in the data set represents the performance of the team for the given year adjusted to the current length of the season - 162 games. The data set includes 16 variables and the training set includes 2,276 records. Following Table show variable Statistical DEscriptions :

		Min	Median	Mean	SD	Max	Num_Zeros	Num_NaN
1	TARGET_WINS	0	82.0	81	16	146	1	0
2	TEAM_BATTING_H	891	1454.0	1469	145	2554	0	0
3	TEAM_BATTING_2B	69	238.0	241	47	458	0	0
4	TEAM_BATTING_3B	0	47.0	55	28	223	2	0
5	TEAM_BATTING_HR	0	102.0	100	61	264	15	0
6	TEAM_BATTING_BB	0	512.0	502	123	878	1	0
7	TEAM_BATTING_SO	0	750.0	736	249	1399	20	102
8	TEAM_BASERUN_SB	0	101.0	125	88	697	2	131
9	TEAM_BASERUN_CS	0	49.0	53	23	201	1	772
10	TEAM_BATTING_HBP	29	58.0	59	13	95	0	2085
11	TEAM_PITCHING_H	1137	1518.0	1779	1407	30132	0	0
12	TEAM_PITCHING_HR	0	107.0	106	61	343	15	0
13	TEAM_PITCHING_BB	0	536.5	553	166	3645	1	0
14	TEAM_PITCHING_SO	0	813.5	818	553	19278	20	102
15	TEAM_FIELDING_E	65	159.0	246	228	1898	0	0
16	TEAM_FIELDING_DP	52	149.0	146	26	228	0	286

Some initial observations:

- The response variable (`TARGET_WINS`) looks to be normally distributed. This supports the working theory that there are good teams and bad teams. There are also a lot of average teams.
- There are also quite a few variables with missing values. I may need to deal with these in order to have the largest data set possible for modeling.
- A couple variables are bimodal (`TEAM_BATTING_HR`, `TEAM_BATTING_SO` `TEAM_PITCHING_HR`). This may be a challenge as some of them are missing values and that may be a challenge in filling in missing values.
- Some variables are right skewed (`TEAM_BASERUN_CS`, `TEAM_BASERUN_SB`, etc.). This might support the good team theory. It may also introduce non-normally distributed residuals in the model.

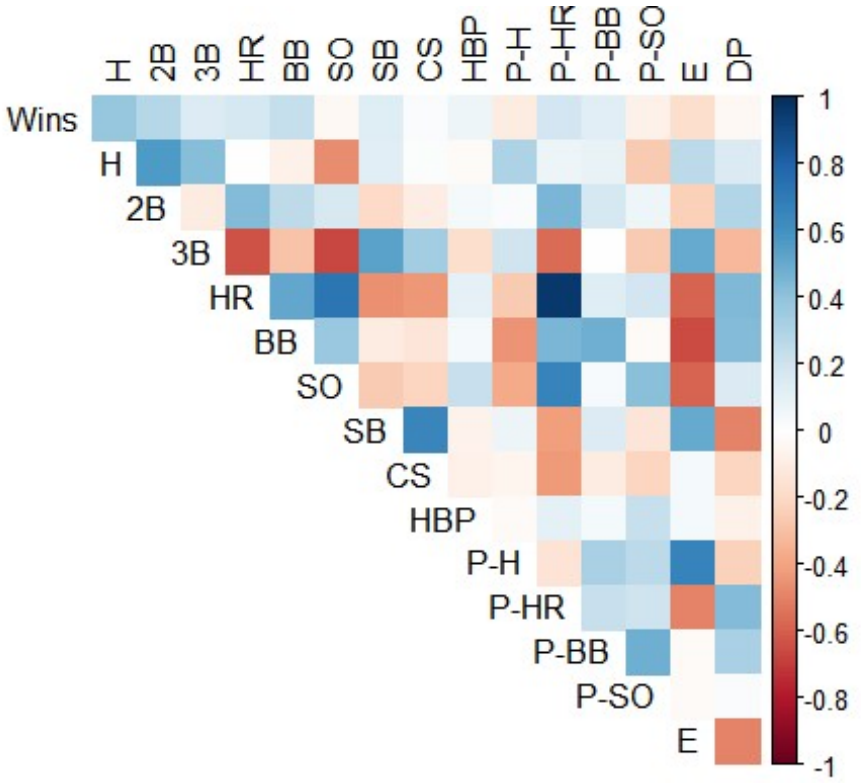
Correlations Matrix Table

	Wins	H	2B	3B	HR	BB	SO	SB	CS	HBP	P-H	P-HR
Wins	1.00	0.39	0.29	0.14	0.18	0.23	-0.03	0.14	0.02	0.07	-0.11	0.19
H	0.39	1.00	0.56	0.43	-0.01	-0.07	-0.46	0.12	0.02	-0.03	0.30	0.07
2B	0.29	0.56	1.00	-0.11	0.44	0.26	0.16	-0.20	-0.10	0.05	0.02	0.45
3B	0.14	0.43	-0.11	1.00	-0.64	-0.29	-0.67	0.53	0.35	-0.17	0.19	-0.57
HR	0.18	-0.01	0.44	-0.64	1.00	0.51	0.73	-0.45	-0.43	0.11	-0.25	0.97
BB	0.23	-0.07	0.26	-0.29	0.51	1.00	0.38	-0.11	-0.14	0.05	-0.45	0.46
SO	-0.03	-0.46	0.16	-0.67	0.73	0.38	1.00	-0.25	-0.22	0.22	-0.38	0.67
SB	0.14	0.12	-0.20	0.53	-0.45	-0.11	-0.25	1.00	0.66	-0.06	0.07	-0.42
CS	0.02	0.02	-0.10	0.35	-0.43	-0.14	-0.22	0.66	1.00	-0.07	-0.05	-0.42
HBP	0.07	-0.03	0.05	-0.17	0.11	0.05	0.22	-0.06	-0.07	1.00	-0.03	0.11
P-H	-0.11	0.30	0.02	0.19	-0.25	-0.45	-0.38	0.07	-0.05	-0.03	1.00	-0.14
P-HR	0.19	0.07	0.45	-0.57	0.97	0.46	0.67	-0.42	-0.42	0.11	-0.14	1.00
P-BB	0.12	0.09	0.18	0.00	0.14	0.49	0.04	0.15	-0.11	0.05	0.32	0.22
P-SO	-0.08	-0.25	0.06	-0.26	0.18	-0.02	0.42	-0.14	-0.21	0.22	0.27	0.21
E	-0.18	0.26	-0.24	0.51	-0.59	-0.66	-0.58	0.51	0.05	0.04	0.67	-0.49
DP	-0.03	0.16	0.29	-0.32	0.45	0.43	0.15	-0.50	-0.21	-0.07	-0.23	0.44

	P-BB	P-SO	E	DP
Wins	0.12	-0.08	-0.18	-0.03
H	0.09	-0.25	0.26	0.16
2B	0.18	0.06	-0.24	0.29
3B	0.00	-0.26	0.51	-0.32
HR	0.14	0.18	-0.59	0.45
BB	0.49	-0.02	-0.66	0.43
SO	0.04	0.42	-0.58	0.15
SB	0.15	-0.14	0.51	-0.50
CS	-0.11	-0.21	0.05	-0.21
HBP	0.05	0.22	0.04	-0.07
P-H	0.32	0.27	0.67	-0.23
P-HR	0.22	0.21	-0.49	0.44
P-BB	1.00	0.49	-0.02	0.32
P-SO	0.49	1.00	-0.02	0.03
E	-0.02	-0.02	1.00	-0.50
DP	0.32	0.03	-0.50	1.00

Correlations Matrix Plots

Let's take a look at the correlations. The following is the correlations from the complete cases only:



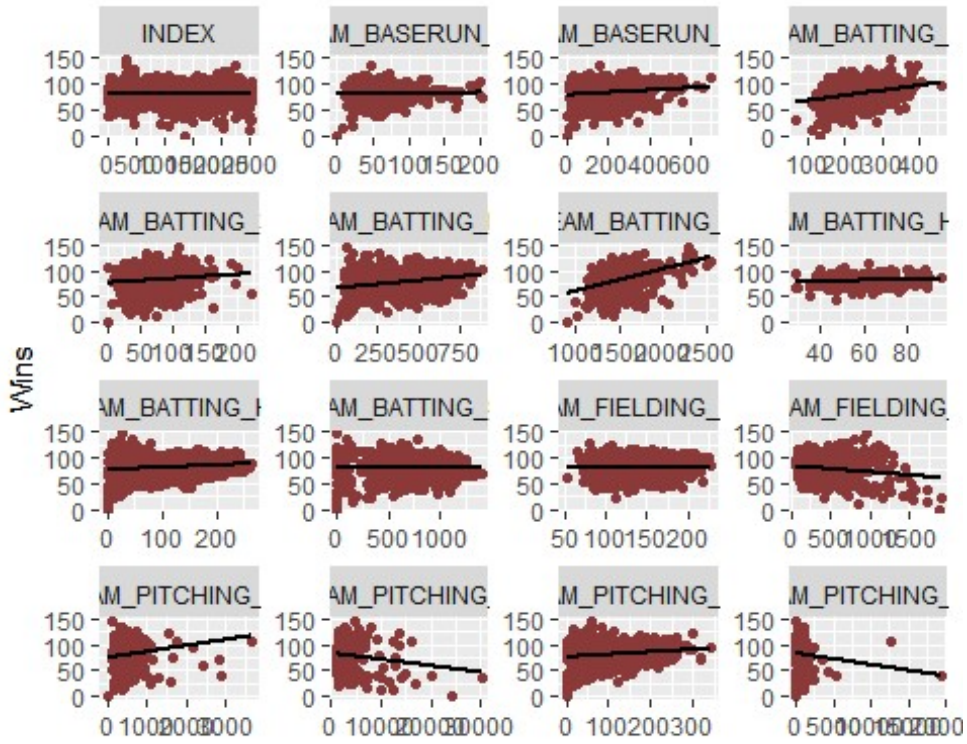
Anything over 0.5 or under -0.5 is highlighted in blue. The matrix was created using complete pairwise observations.

A few conclusions:

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

Correlations: Endogenous and Exogenous Variables

Let's take a look at how the Exogenous(Model Inputs) are correlated with the response variable(Endogenous):



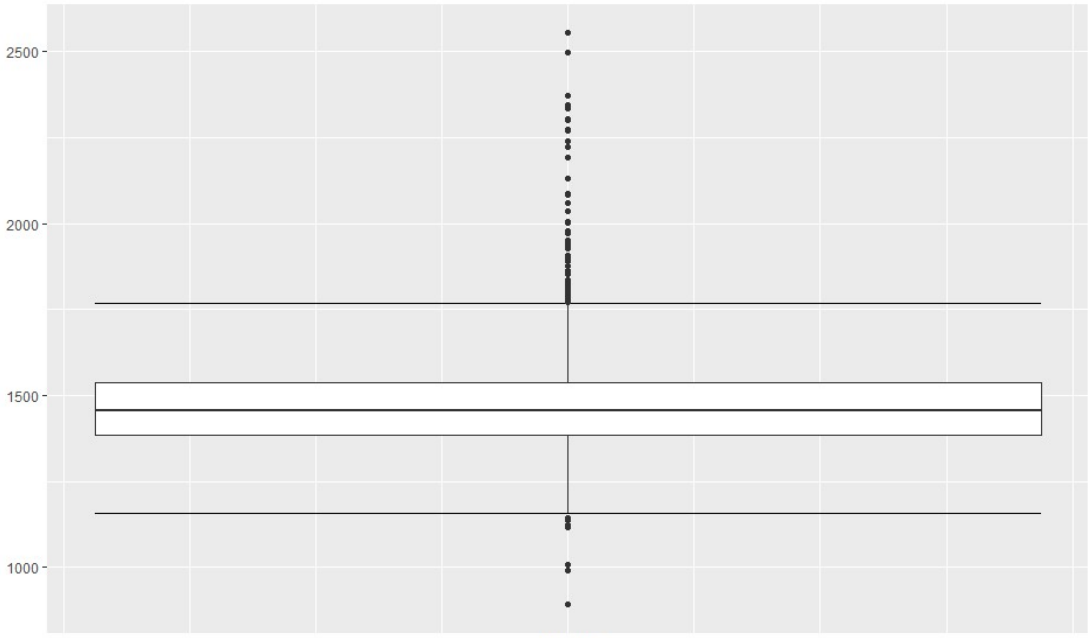
Variable chacteristics

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

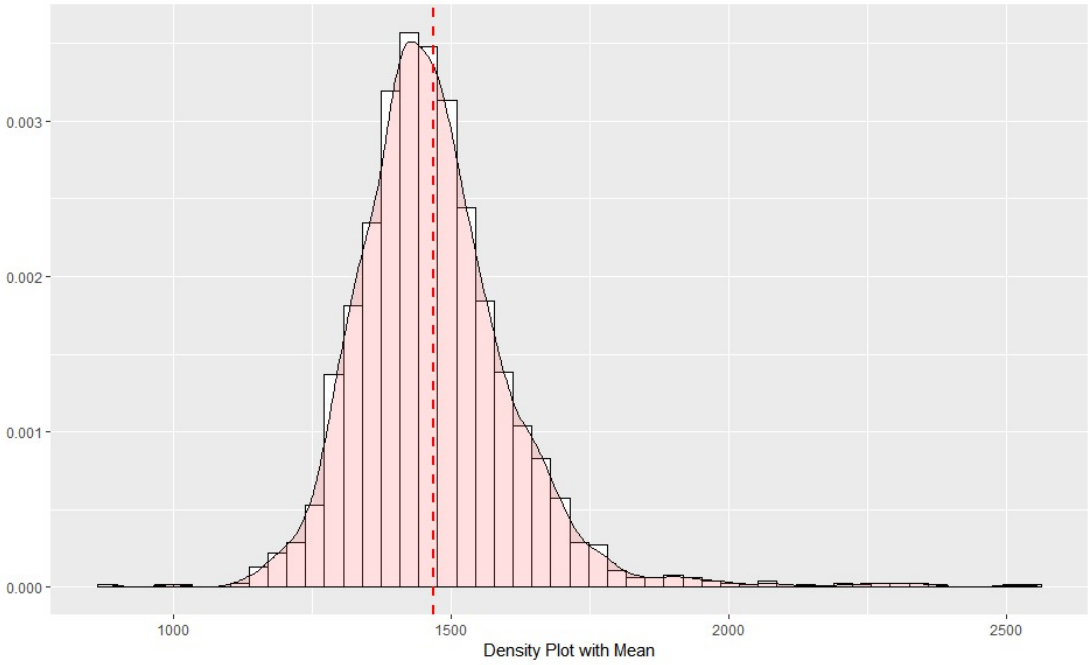
TEAM_BATTING_H:

This variable represents number of team base hits:

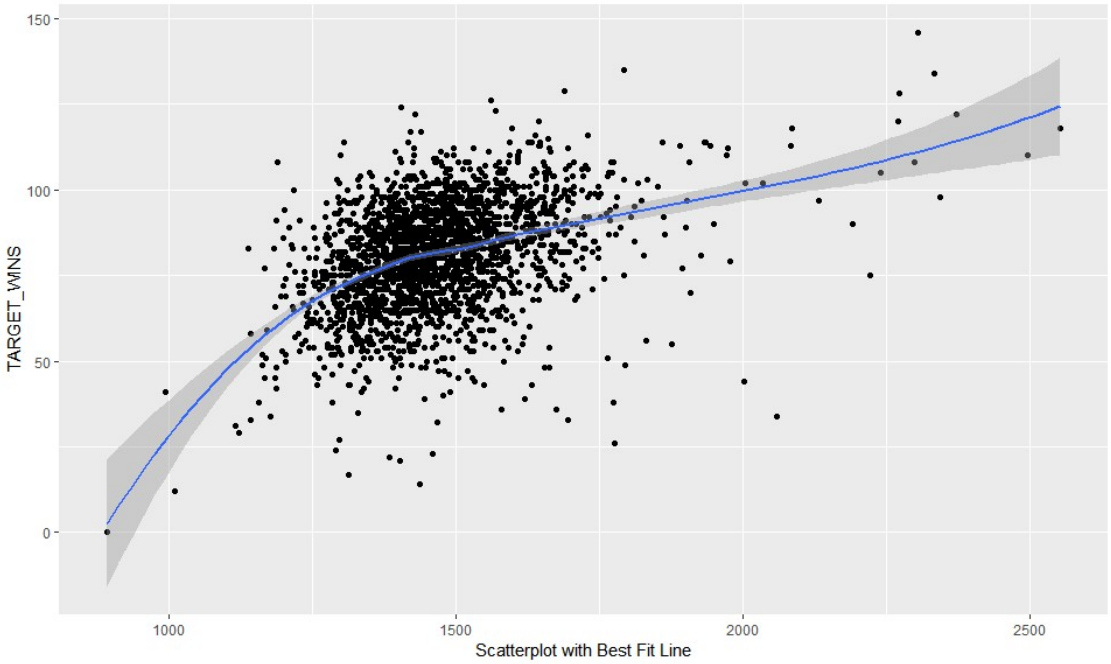
```
  Min Median Mean  SD  Max Num_Zeros Num_NaN
2 891   1454 1469 145 2554         0       0
```



Boxplot



Density Plot with Mean

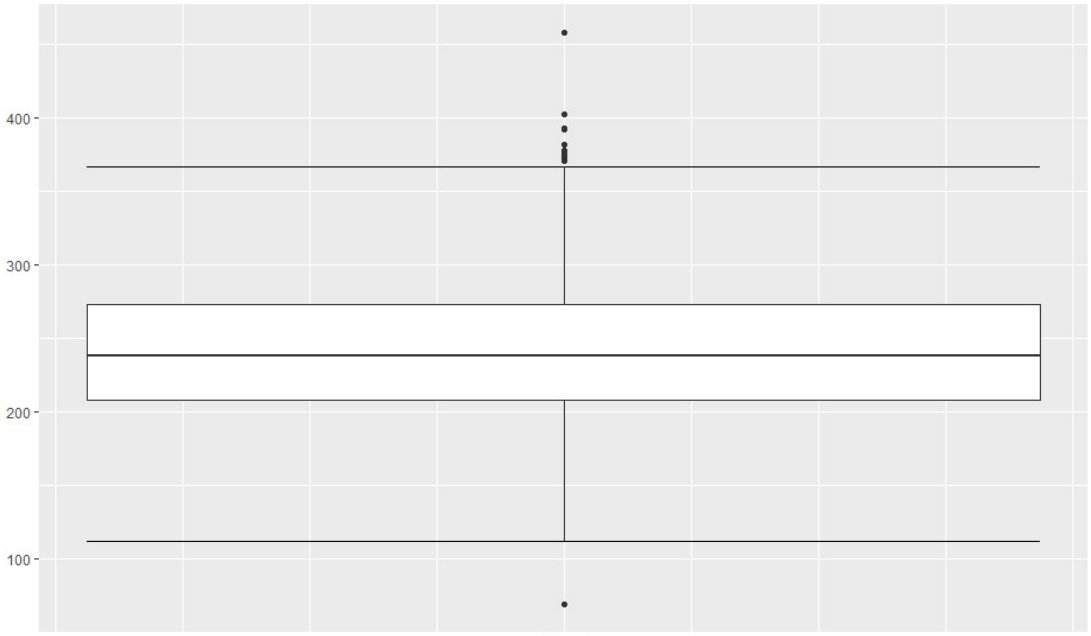


Data Overview: There are no missing values. The range and distribution are reasonable.

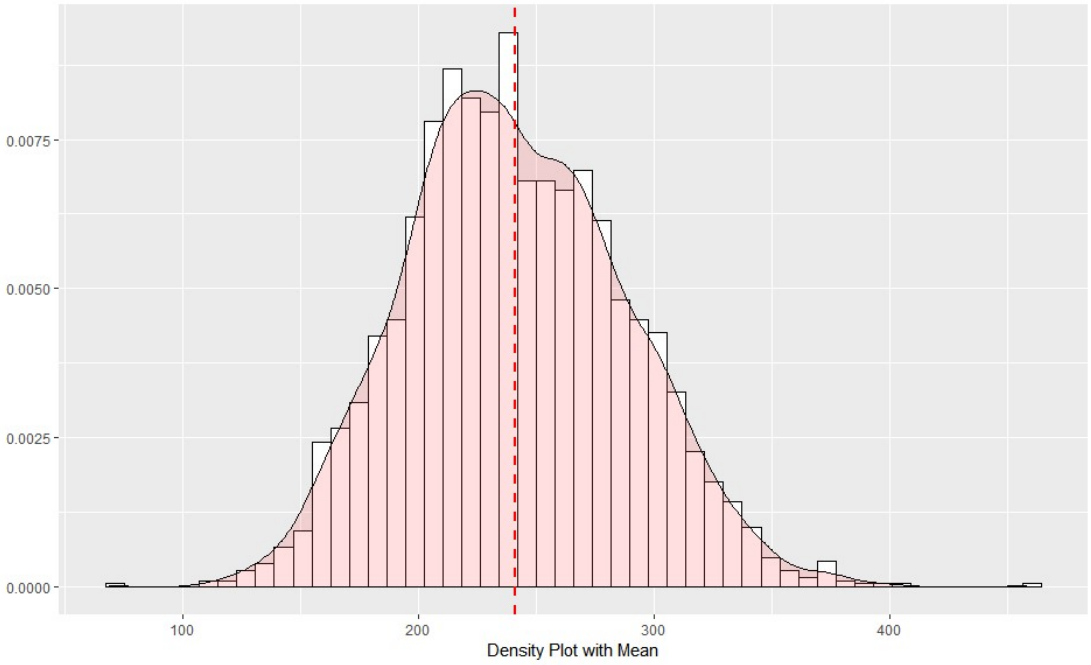
TEAM_BATTING_2B:

This variable represents number of team doubles:

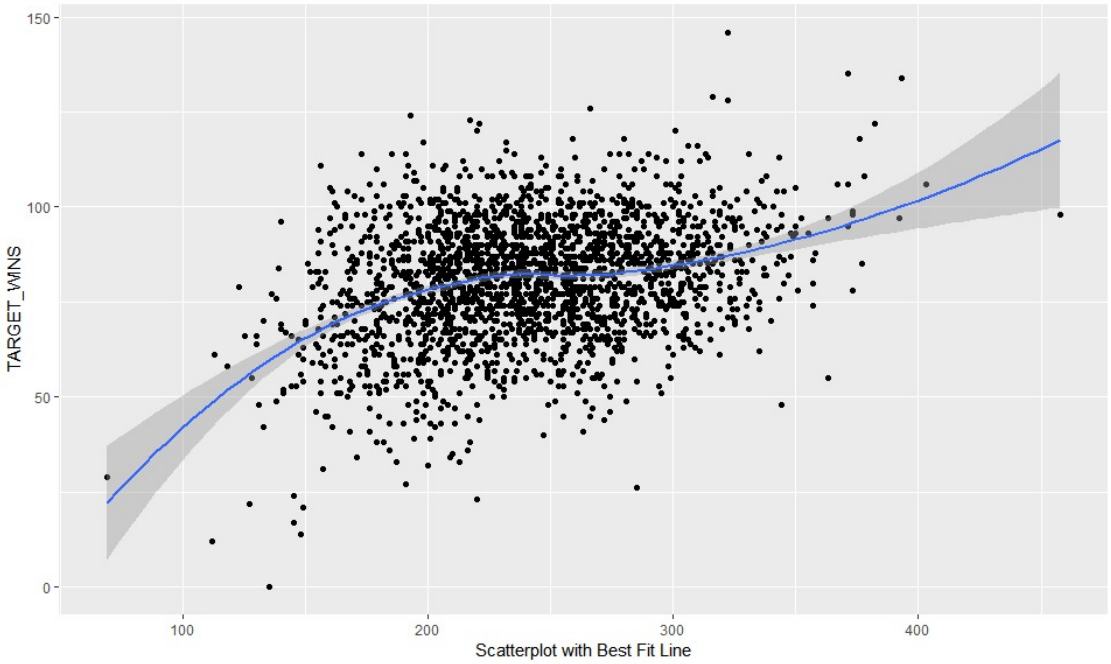
```
  Min Median Mean SD Max Num_Zeros Num_NaN
3  69    238  241 47 458         0       0
```



Boxplot



Density Plot with Mean

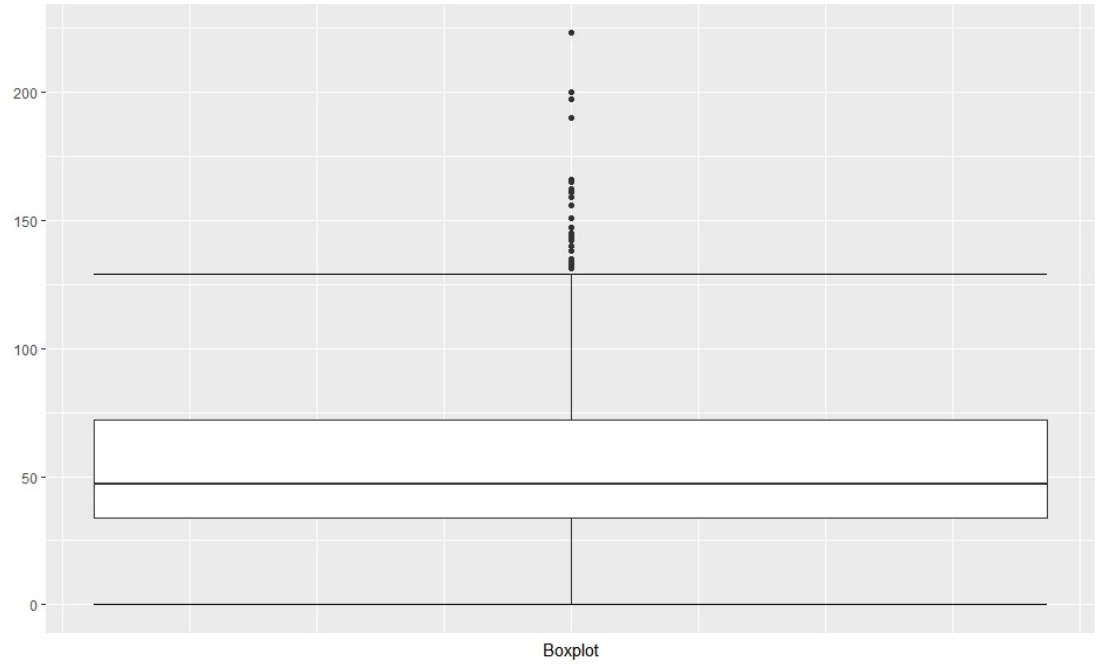


Data Overview: There are no missing values. The range and distribution are reasonable.

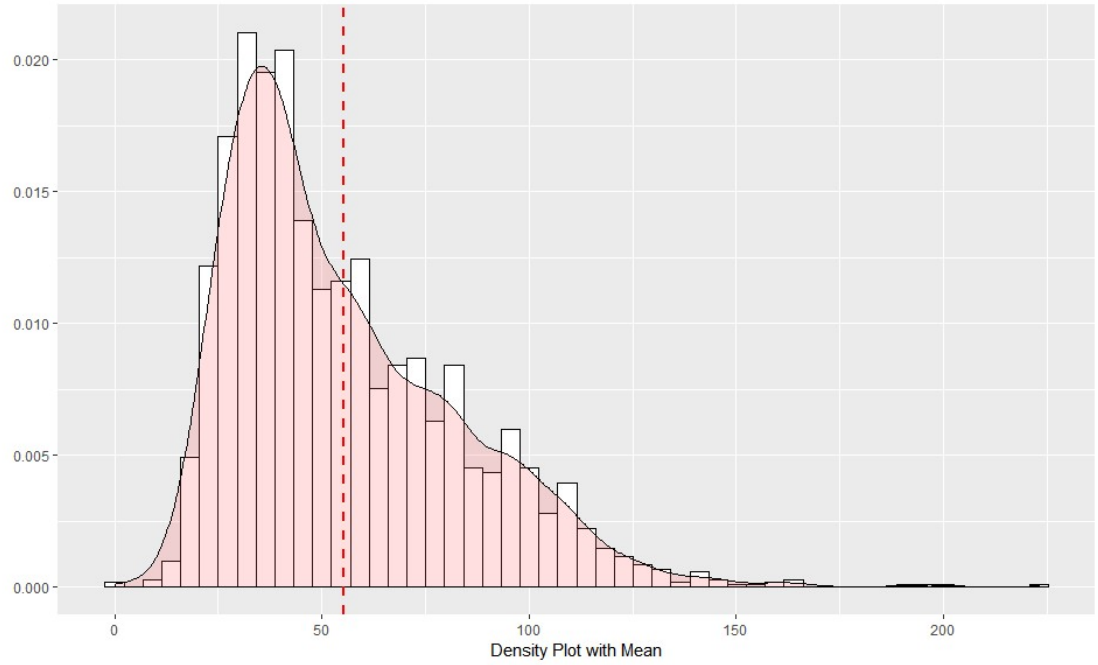
TEAM_BATTING_3B:

This variable represents number of team triples:

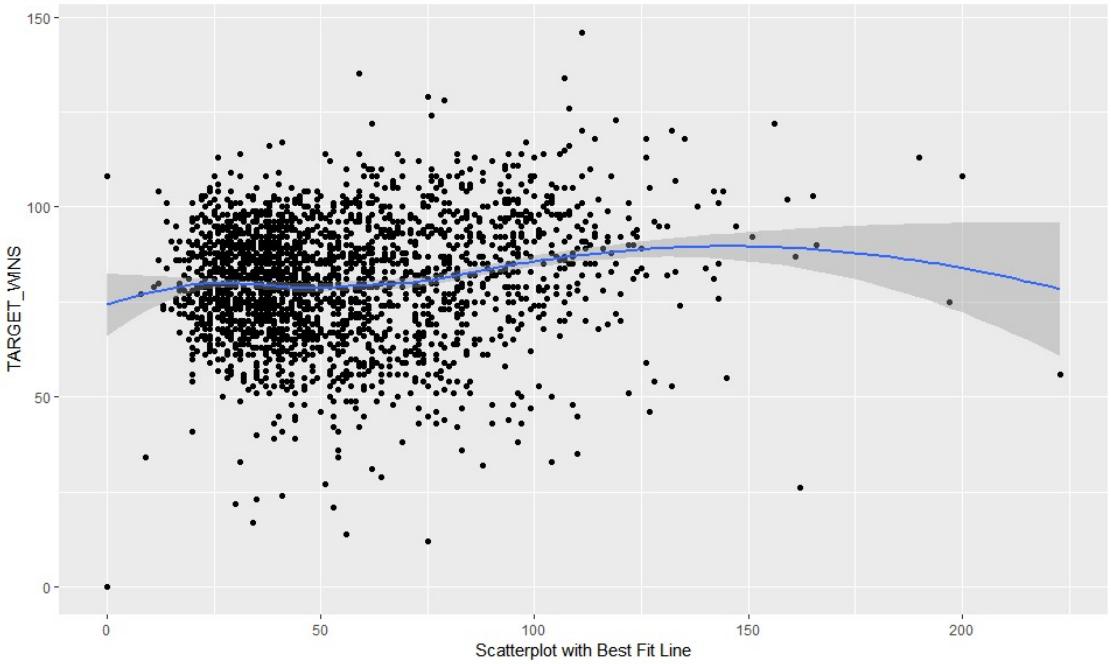
```
  Min Median Mean SD Max Num_Zeros Num_NaN
4   0     47   55 28 223         2       0
```



Boxplot



Density Plot with Mean

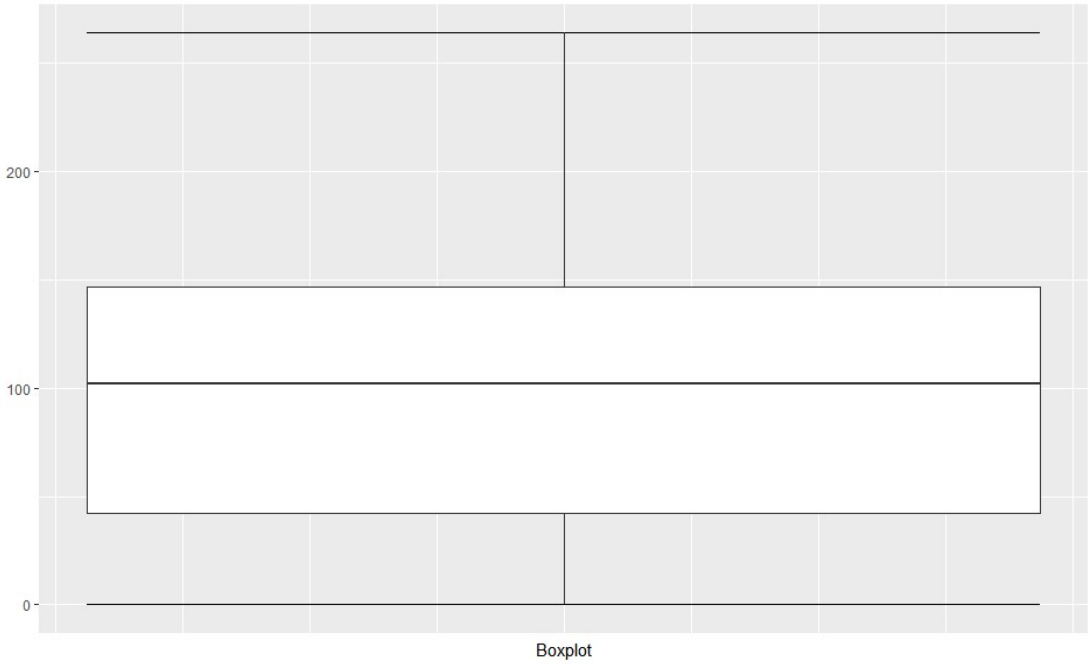


Data Overview: The range and distribution are reasonable. There are 2 records with zero values which is unrealistic for a team in a season. One record (index 1347) has 12 variables with missing values, including the outcome variable. This record will be deleted from the data set. Second record (index 1494) has 7 missing variables, but it does have some recorded values in all categories - batting, pitching and fielding. Zero value for `TEAM_BATTING_3B` can be replaced with the median (because the distribution is right-skewed, median value will provide more realistic estimate).

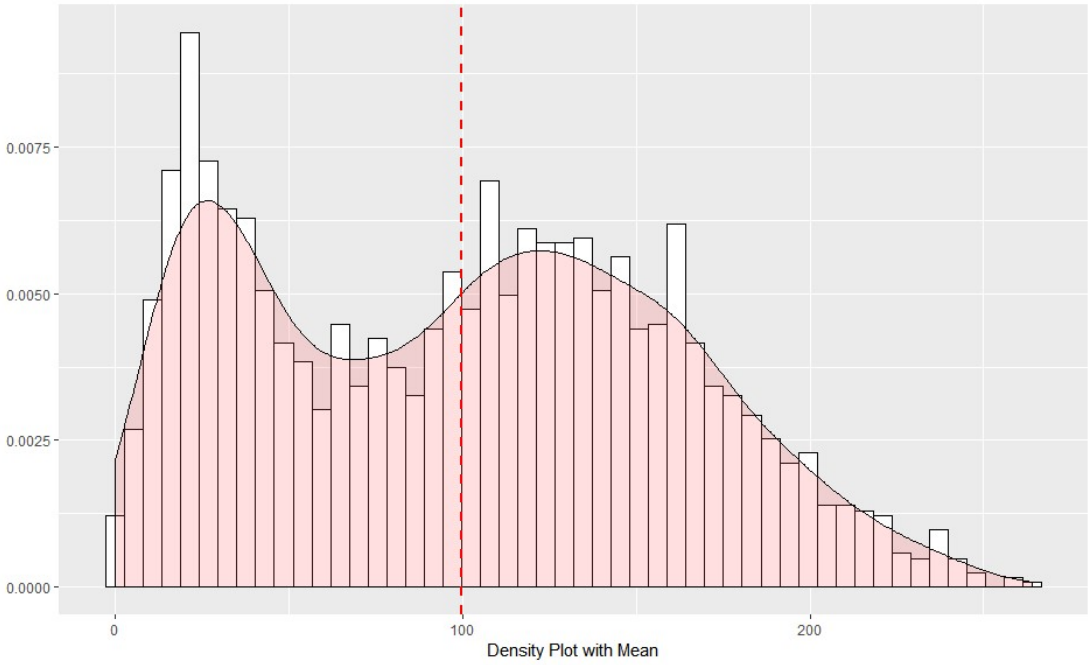
TEAM_BATTING_HR:

This variable represents number of team triples:

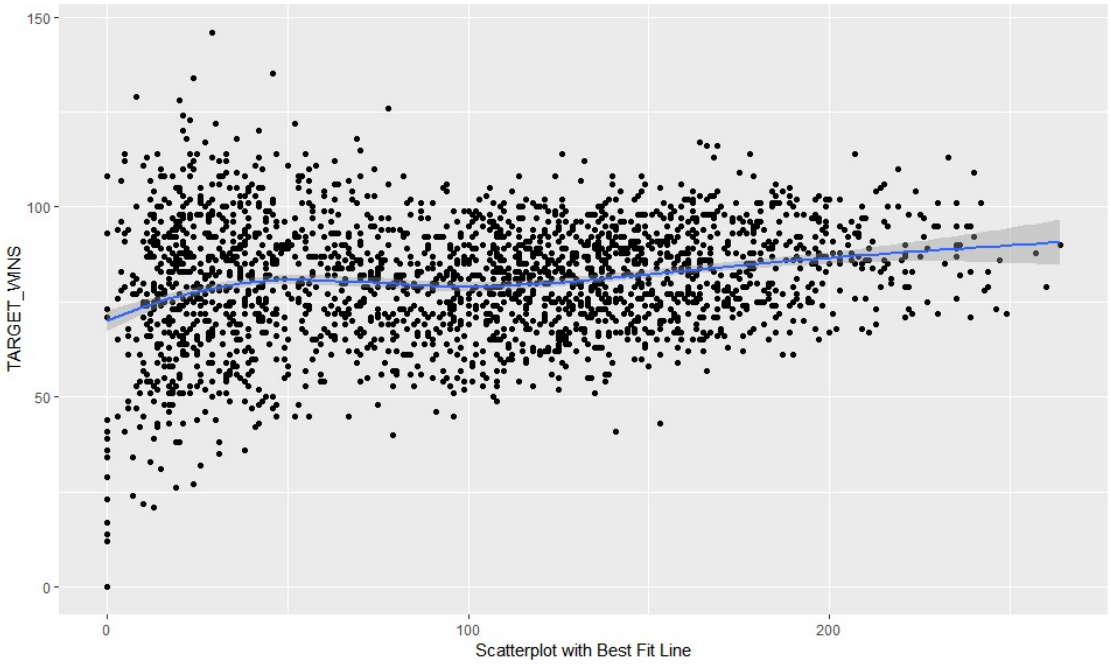
```
  Min Median Mean SD Max Num_Zeros Num_NaN
5   0    102  100 61 264        15       0
```



Boxplot



Density Plot with Mean

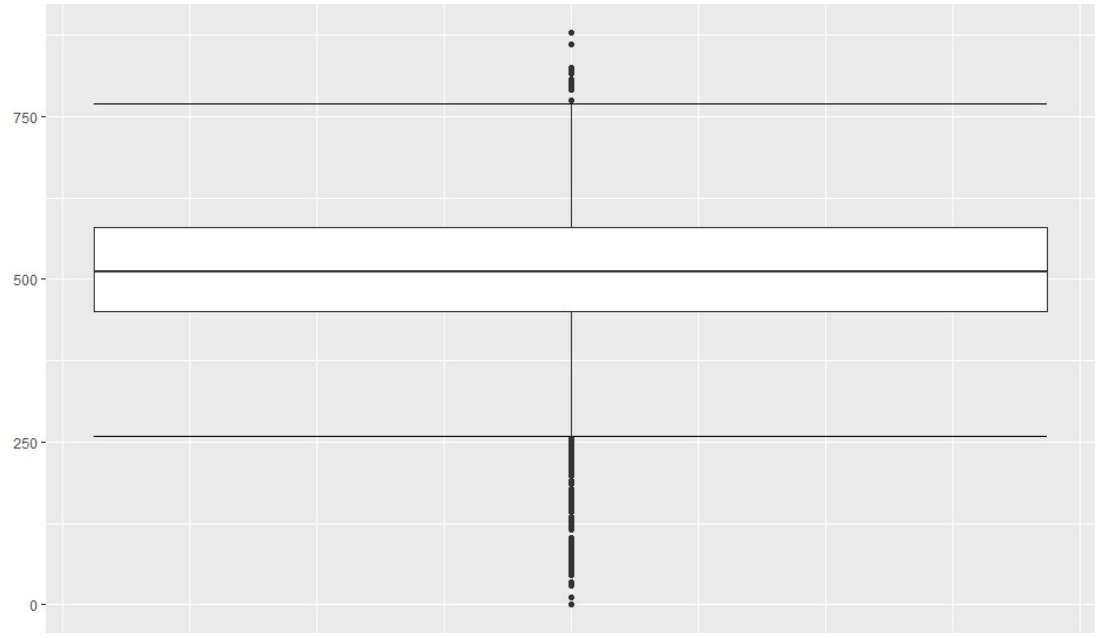


Data Overview: There are some low values in the data. So zero doesn't seem too unusual here either.

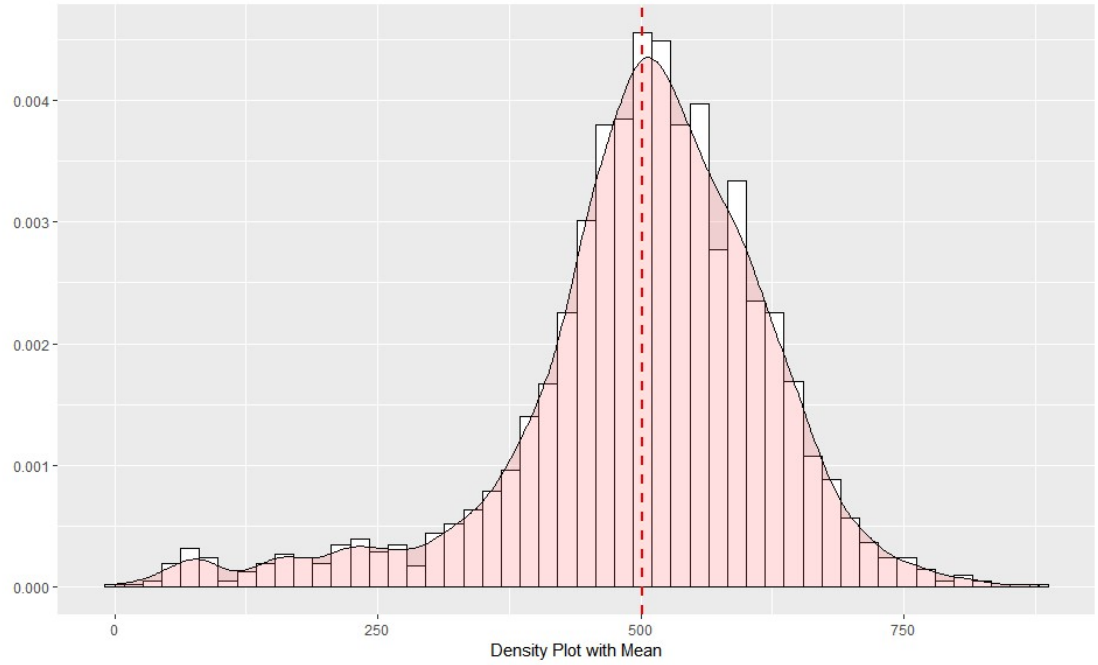
TEAM_BATTING_BB:

This variable represents Number of team walks

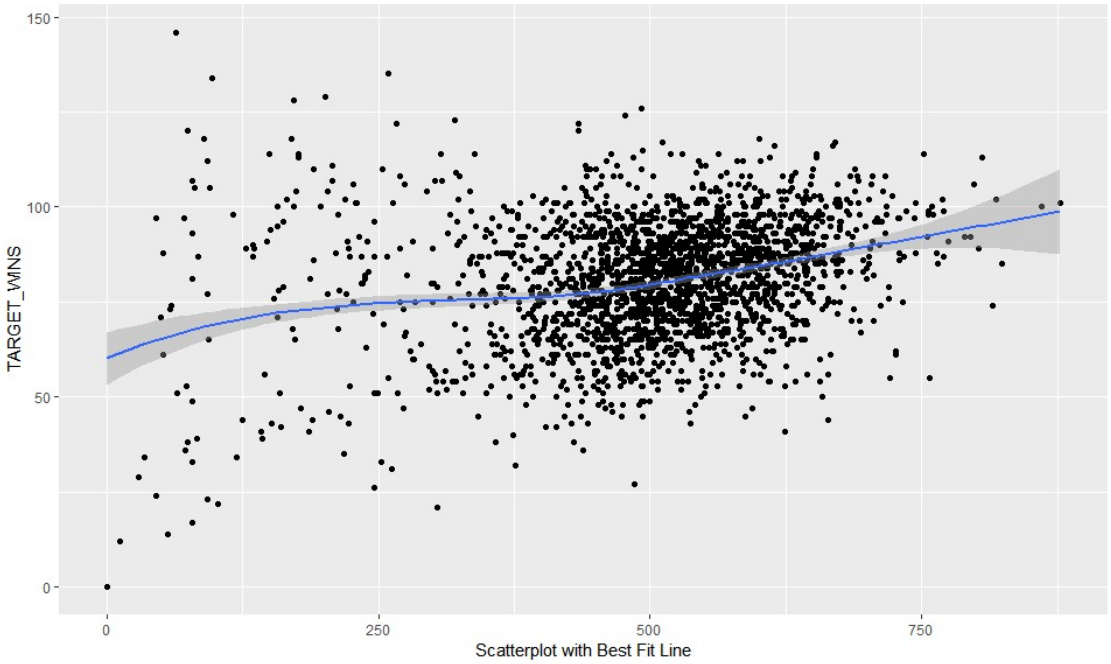
	Min	Median	Mean	SD	Max	Num_Zeros	Num_NaN
6	0	512	502	123	878	1	0



Boxplot



Density Plot with Mean

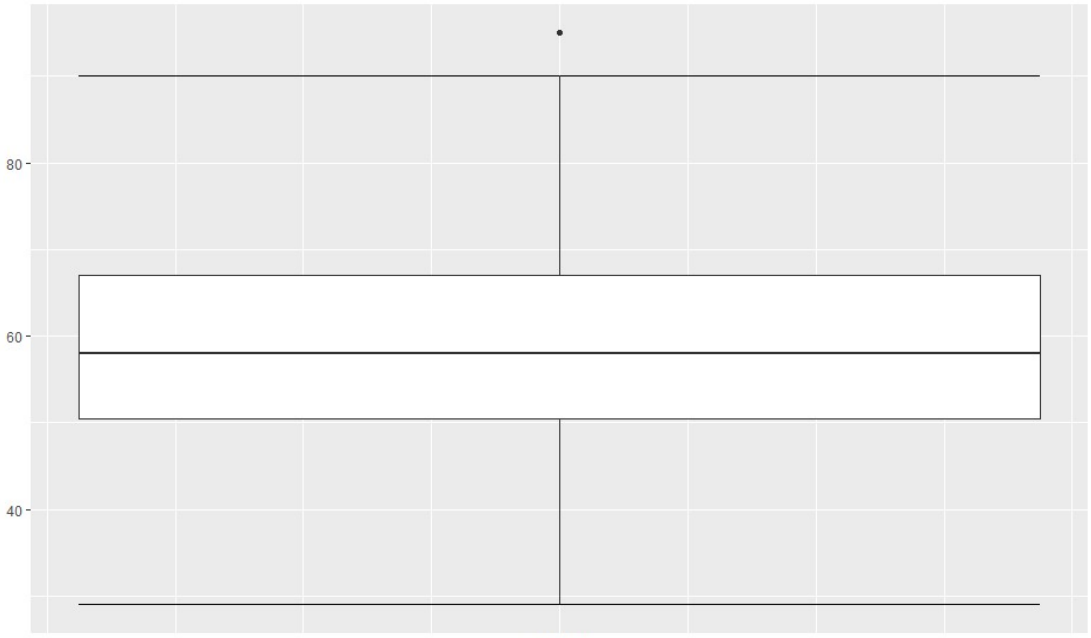


Data Overview: The range and distribution are reasonable. There is one record (index 1347) that has a zero value. This record was discussed above (under `TEAM_BATTING_3B`) and it will be deleted from the data set.

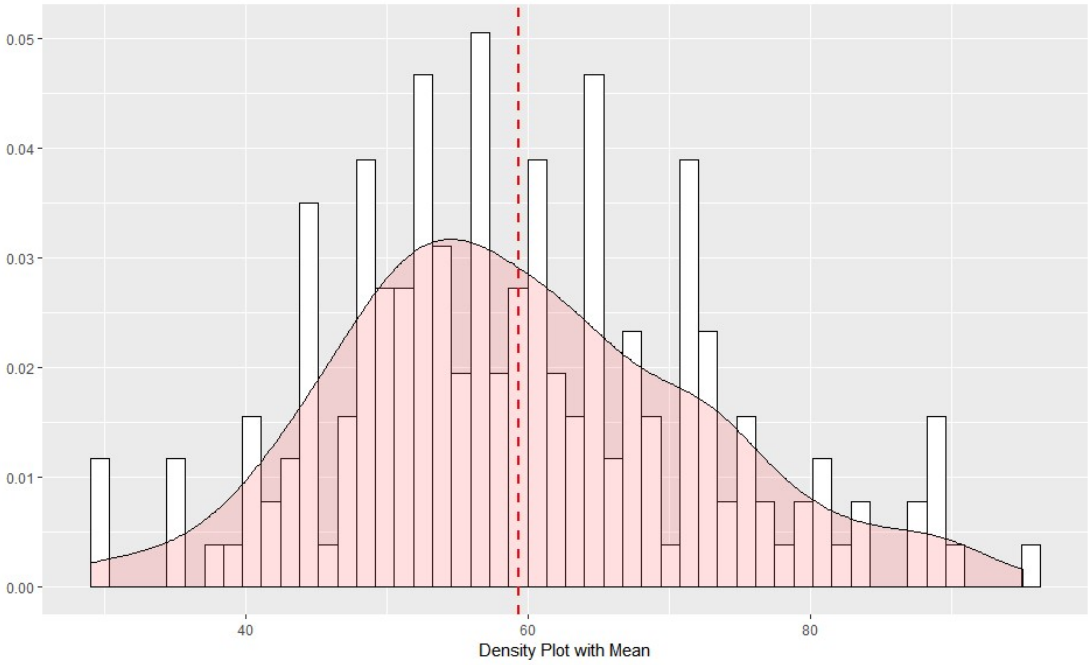
TEAM_BATTING_HBP:

This variable represents Number of team batters hit by pitch

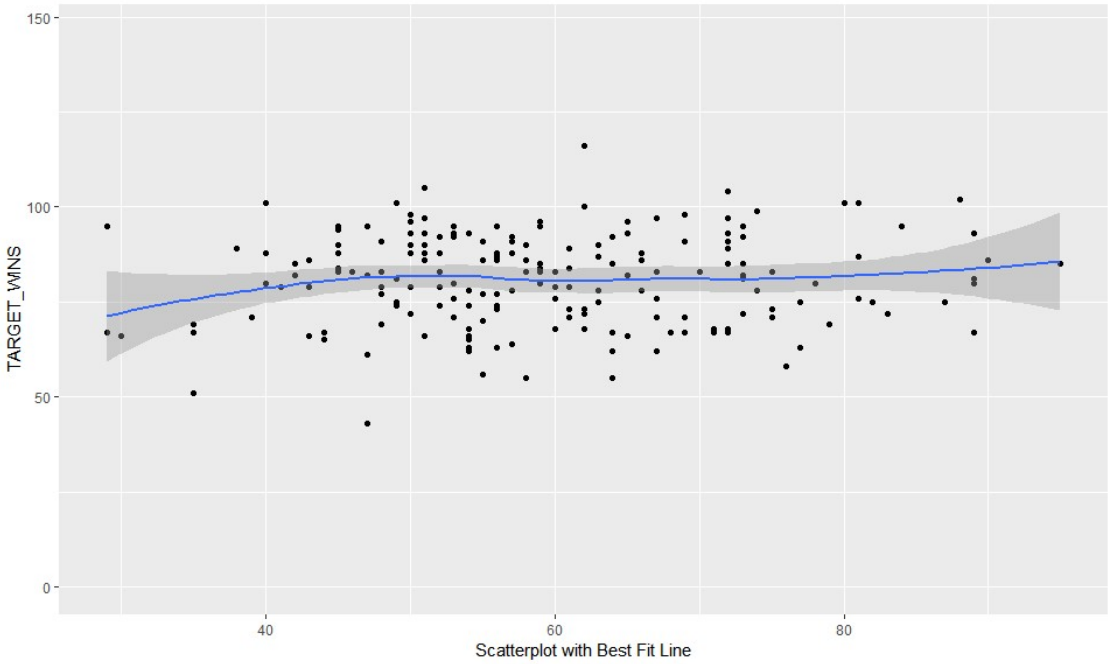
```
   Min Median Mean SD Max Num_Zeros Num_NaN
10  29     58   59 13  95         0    2085
```



Boxplot



Density Plot with Mean

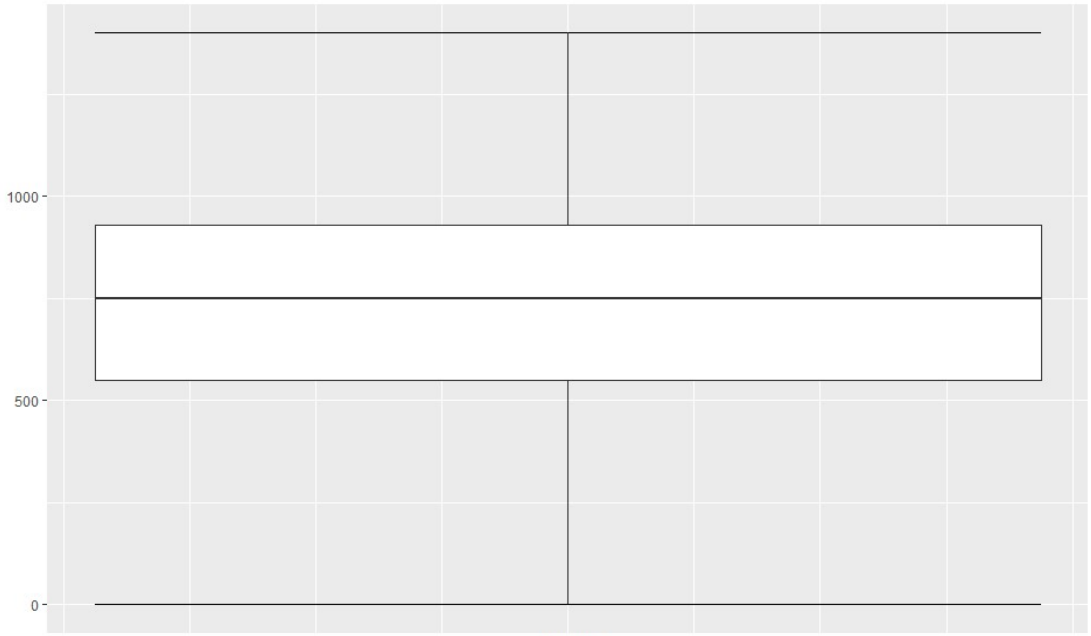


Data Overview: There are 2,085 records - 91.6% of data set - that are missing value. Because this variable is missing for majority of records, I wont consider this variable as input for regression model.

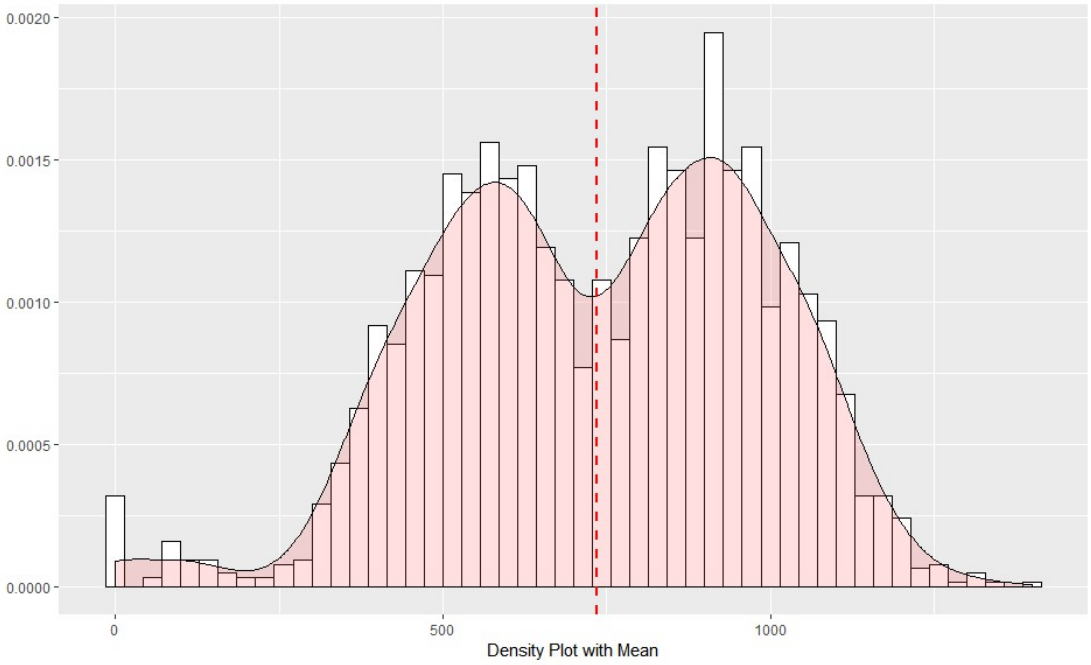
TEAM_BATTING_SO:

This variable represents Number of team strikeouts by batters

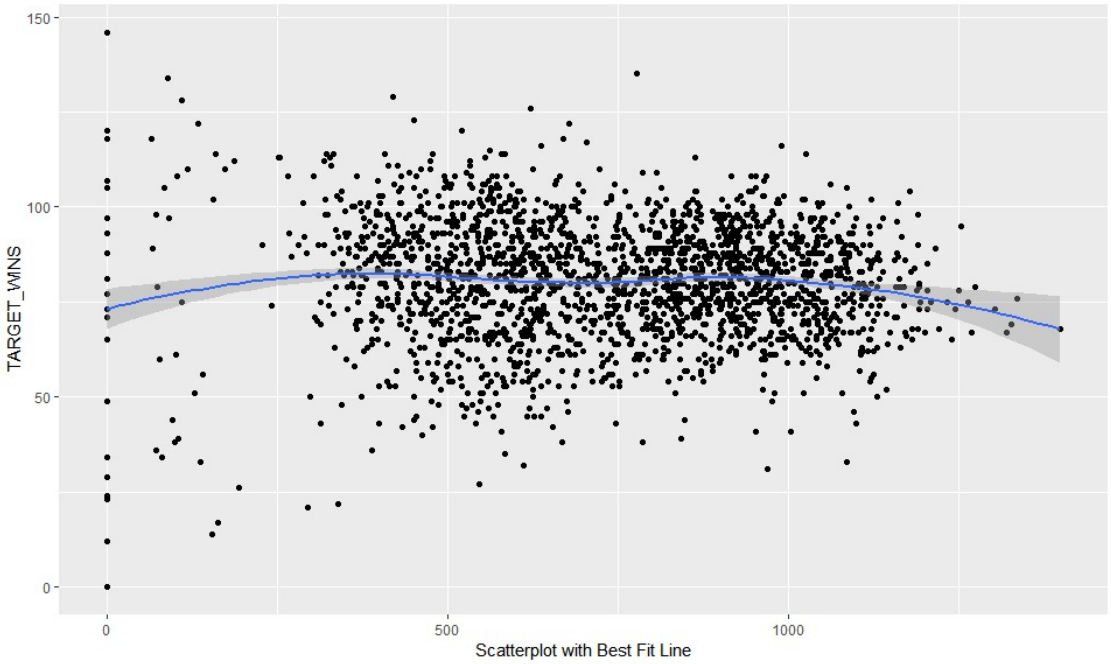
	Min	Median	Mean	SD	Max	Num_Zeros	Num_NaN
7	0	750	736	249	1399	20	102



Boxplot



Density Plot with Mean

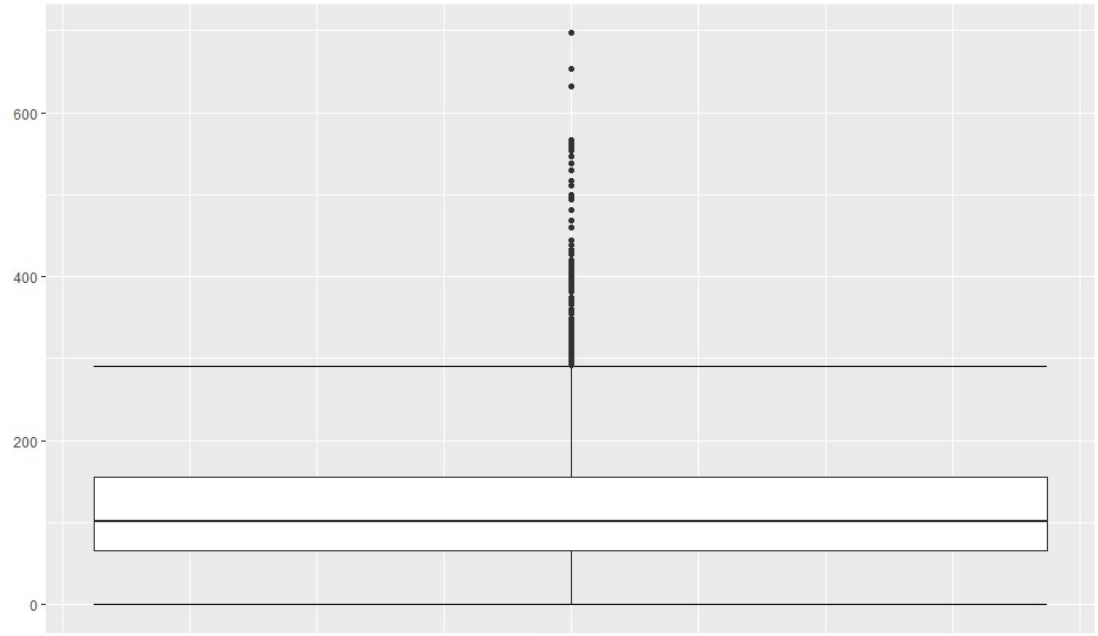


Data Overview: There are 122 records with missing or zero value (as wtih other variables a zero value is unrealistic). These values can be imputed. Similarly to homeruns, the distribution is multimodal, which is interesting enough for additional analysis. Another area of concern is a noticeable left tail. It is highly unlikely to have games without any strikeouts, so anything lower than 162 (average of 1 strikeout per game) is definitely suspect.

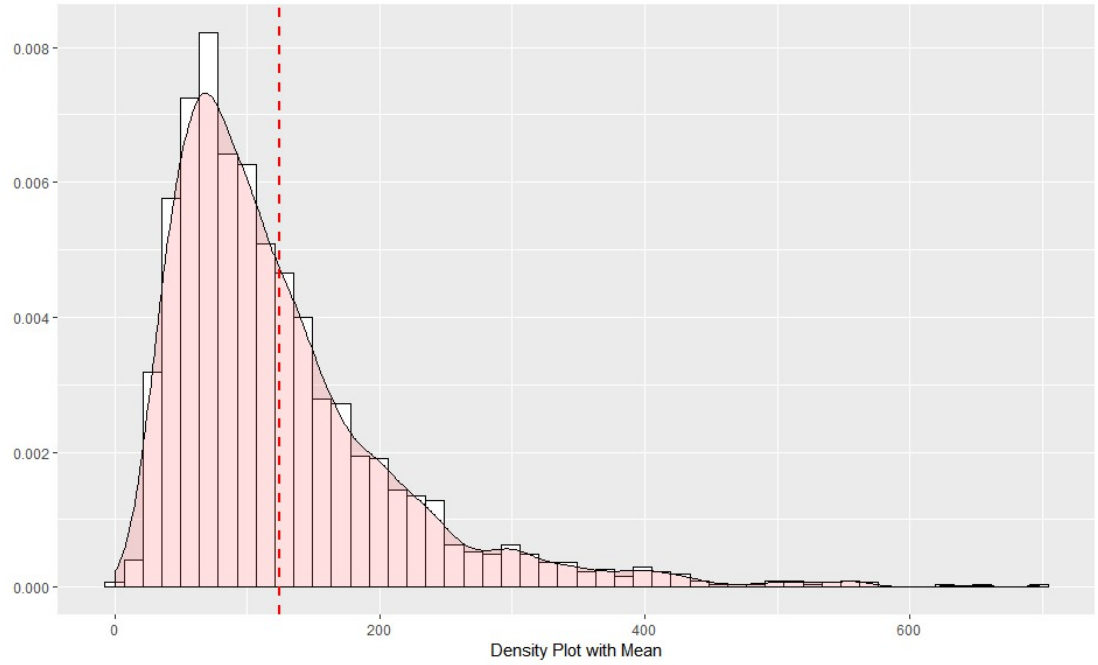
TEAM_BASERUN_SB:

This variable represents Number of team stolen bases

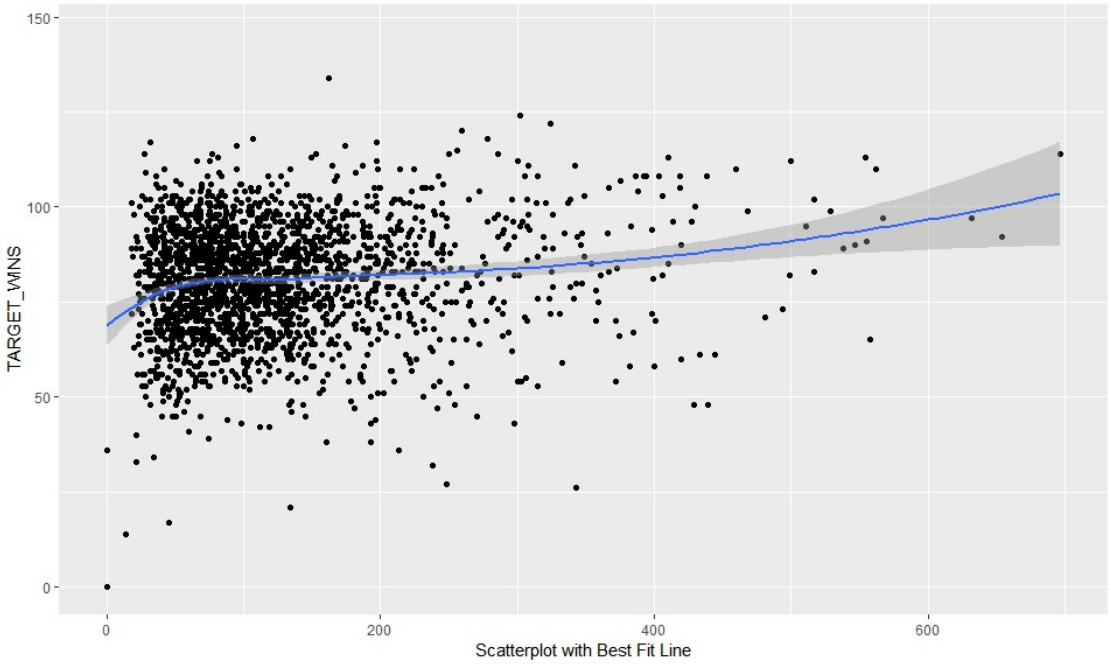
```
  Min Median Mean SD Max Num_Zeros Num_NaN
8   0    101  125 88 697         2     131
```



Boxplot



Density Plot with Mean

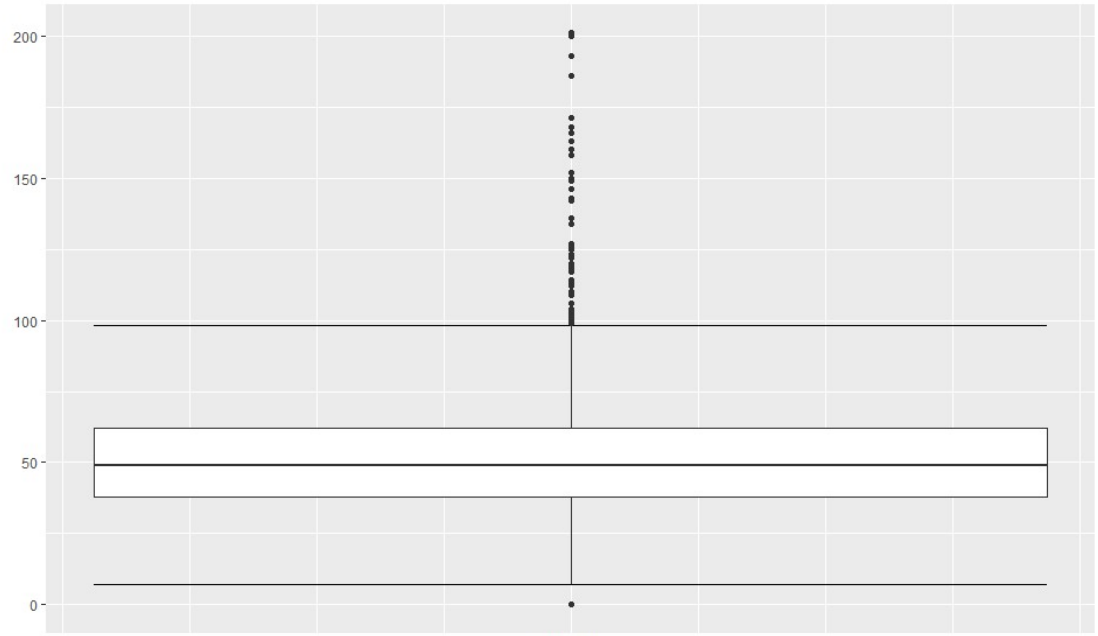


Data Overview: The range and distribution are reasonable. The only issue are 133 records with missing or zero value. These values can be imputed in order to use these records in model building.

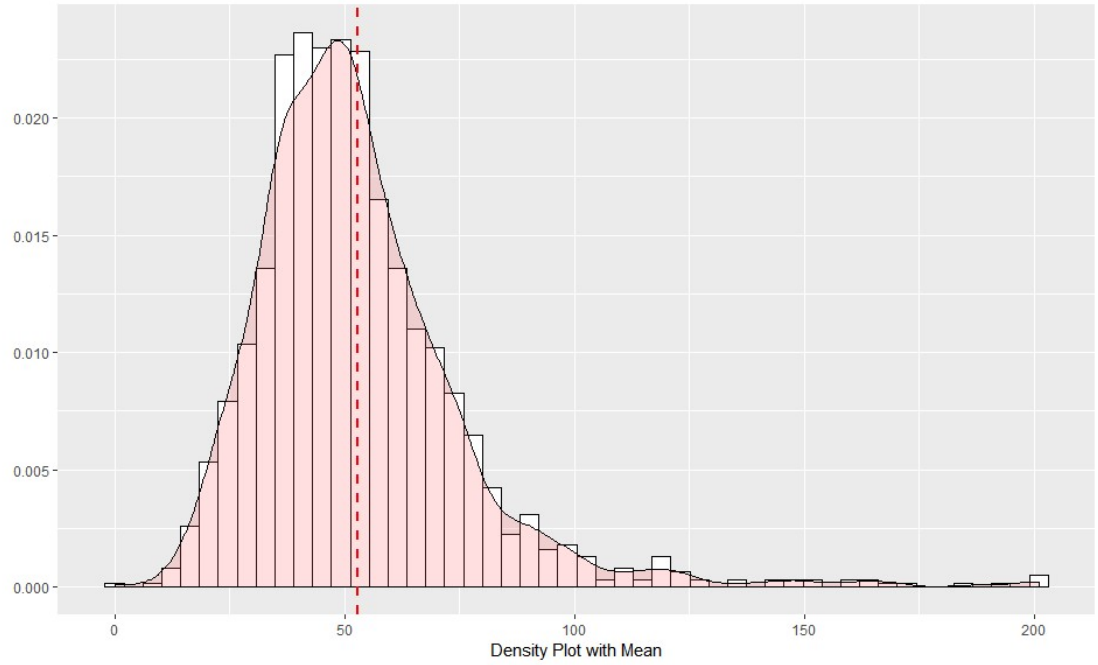
TEAM_BASERUN_CS:

This variable represents Number of team runners caught stealing

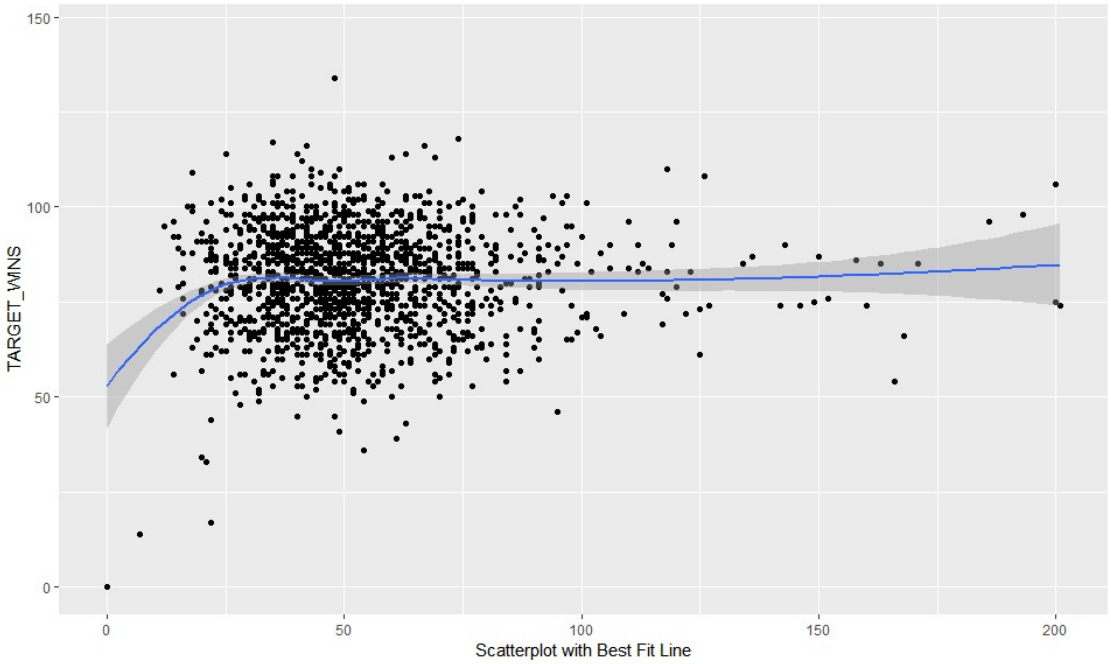
```
  Min Median Mean SD Max Num_Zeros Num_NaN
9   0     49   53 23 201         1     772
```



Boxplot



Density Plot with Mean

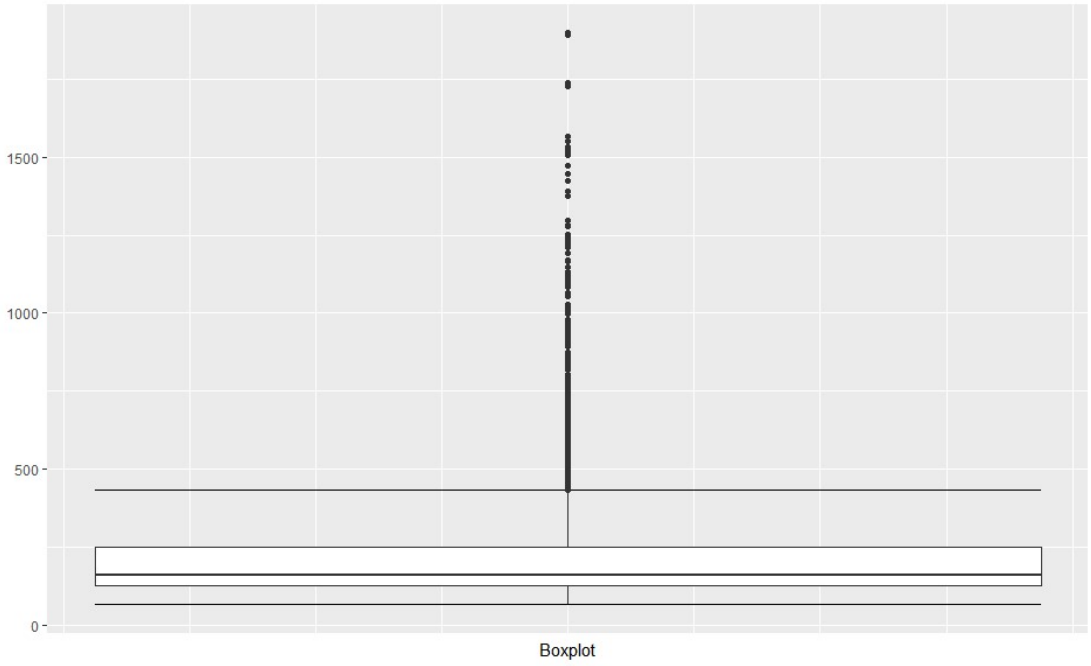


Data Overview: The range and distribution are reasonable; however, there is significant number of missing values - 773, including one zero value. This represents a third of the entire data set. It may be possible to impute this value, but it may be necessary to leave this variable out of model building.

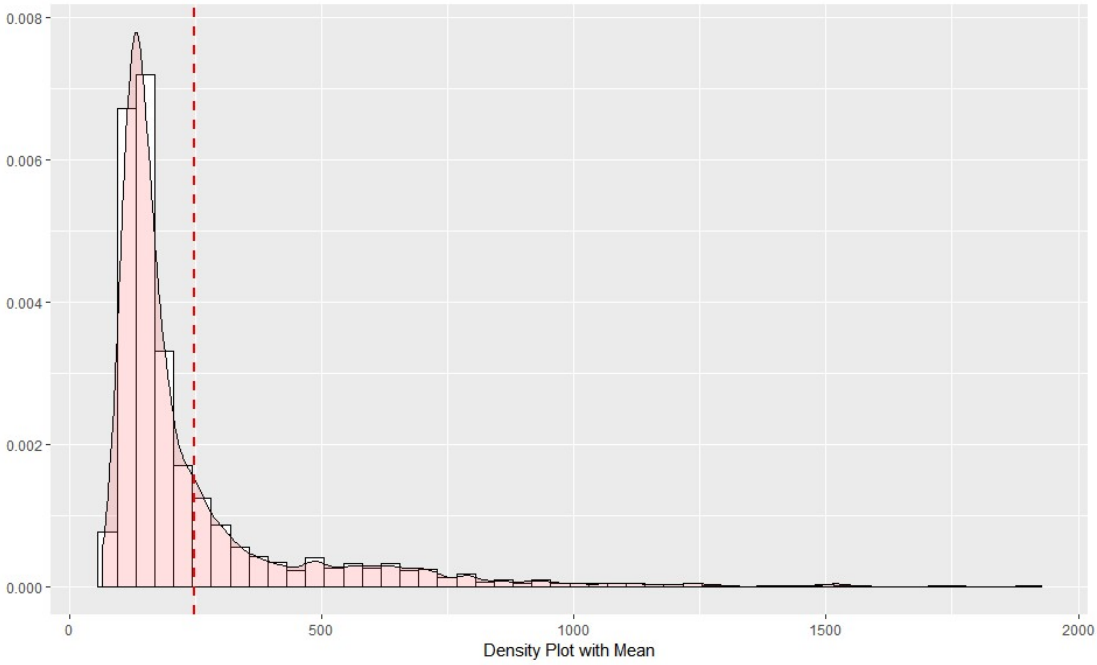
TEAM_FIELDING_E:

This variable represents Number of team fielding errors

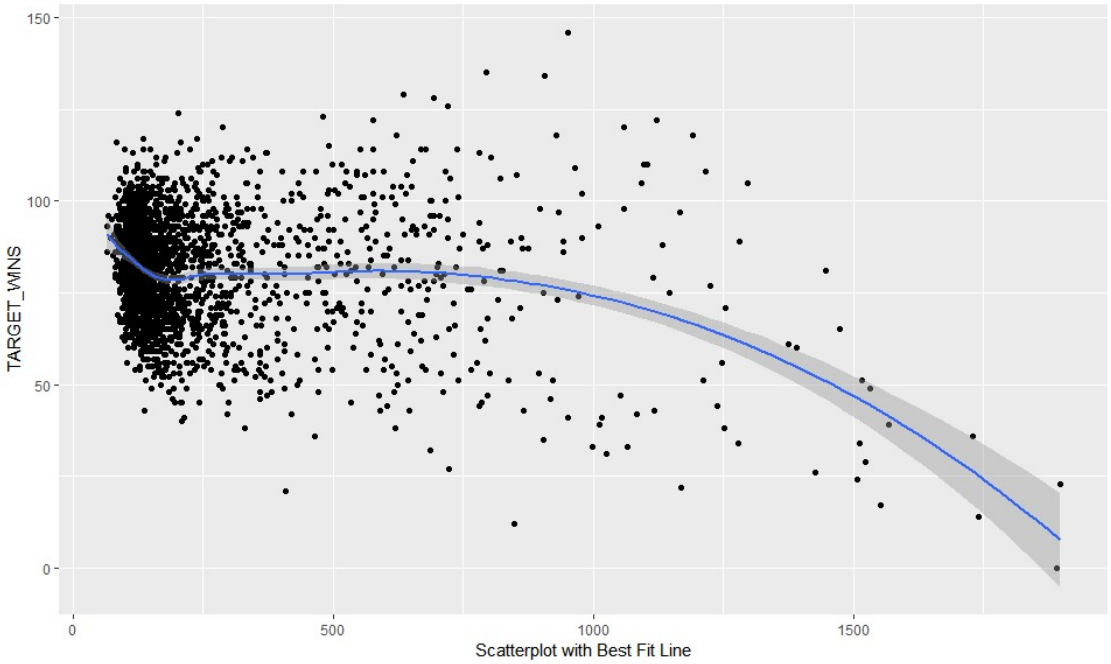
```
   Min Median Mean  SD  Max Num_Zeros Num_NaN
15  65    159  246 228 1898         0       0
```



Boxplot



Density Plot with Mean

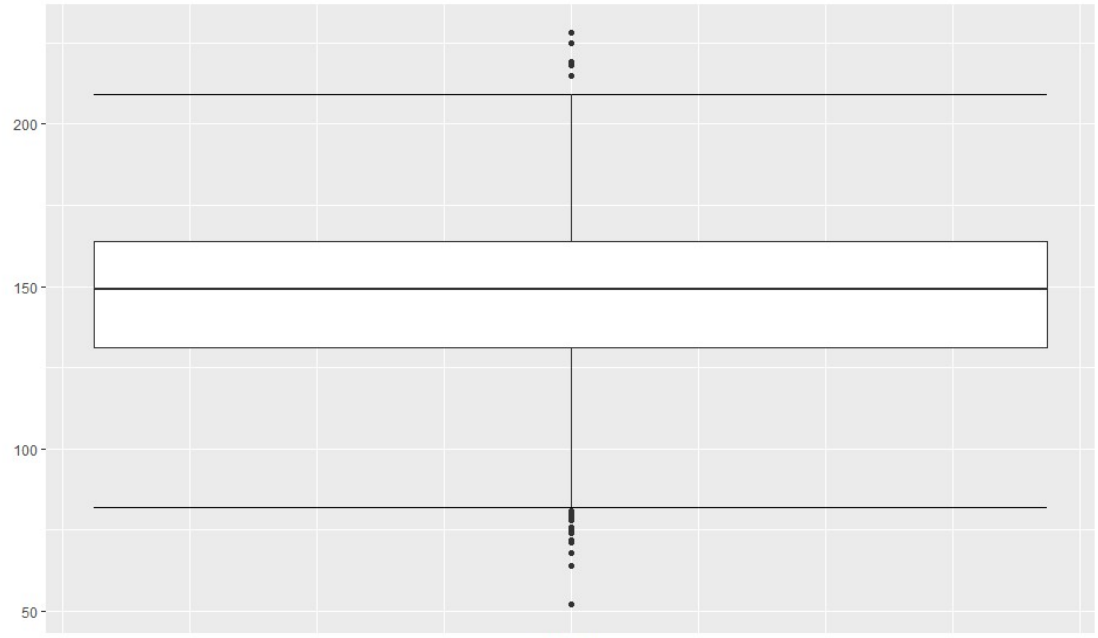


Data Overview: There are no missing values. Distribution has a very long right tail. Values in the 1,000 and above range are highly suspect. One of the highest historical number of errors is 867 errors by Washington in 1886 for 122 games. That is equal about 1,151 errors for 162 game season. There are multiple values above that number. This may unfavorably influence a model.

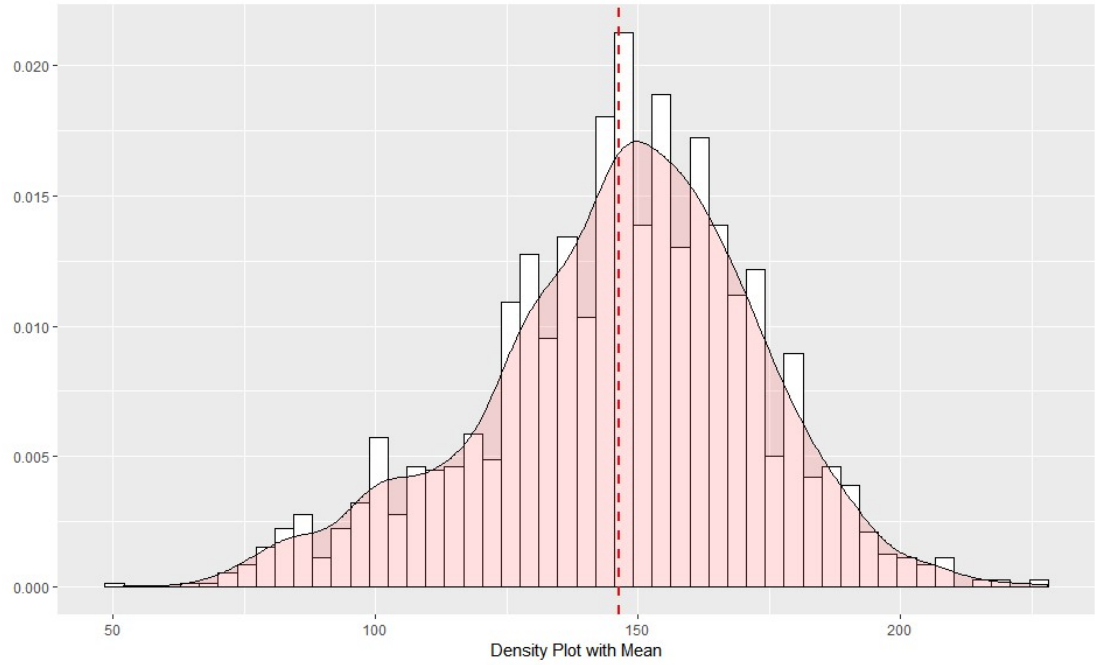
TEAM_FIELDING_DP:

This variable represents Number of team fielding double plays

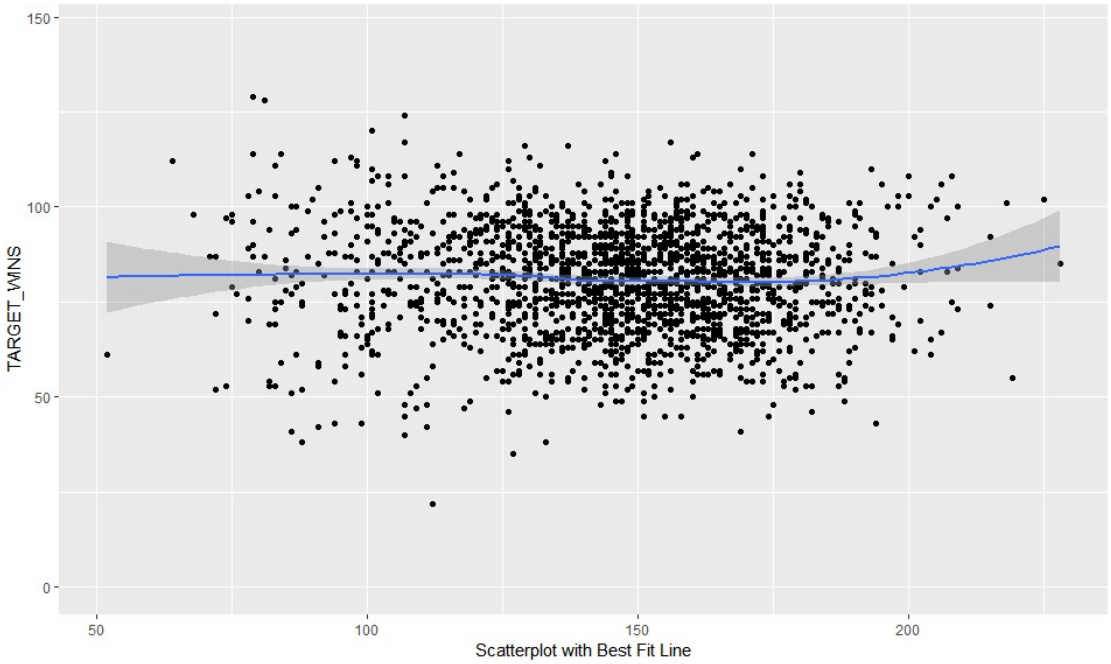
	Min	Median	Mean	SD	Max	Num_Zeros	Num_NaN
16	52	149	146	26	228	0	286



Boxplot



Density Plot with Mean

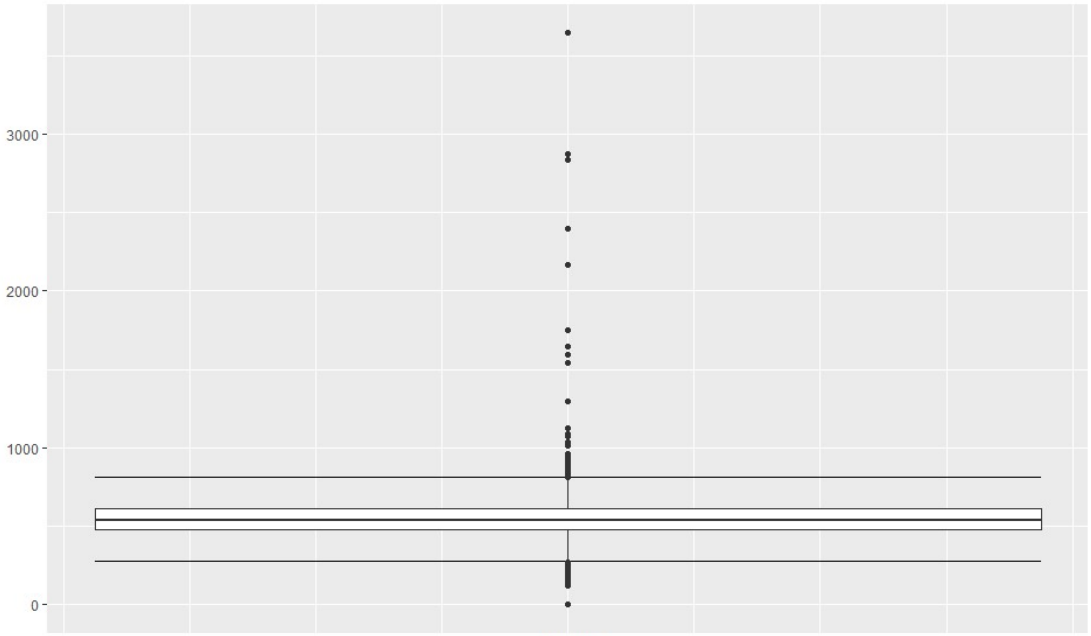


Data Overview: The range and distribution are reasonable. Similar to a few other variables there is a medium number off missing values - 286 records. This value can be imputed.

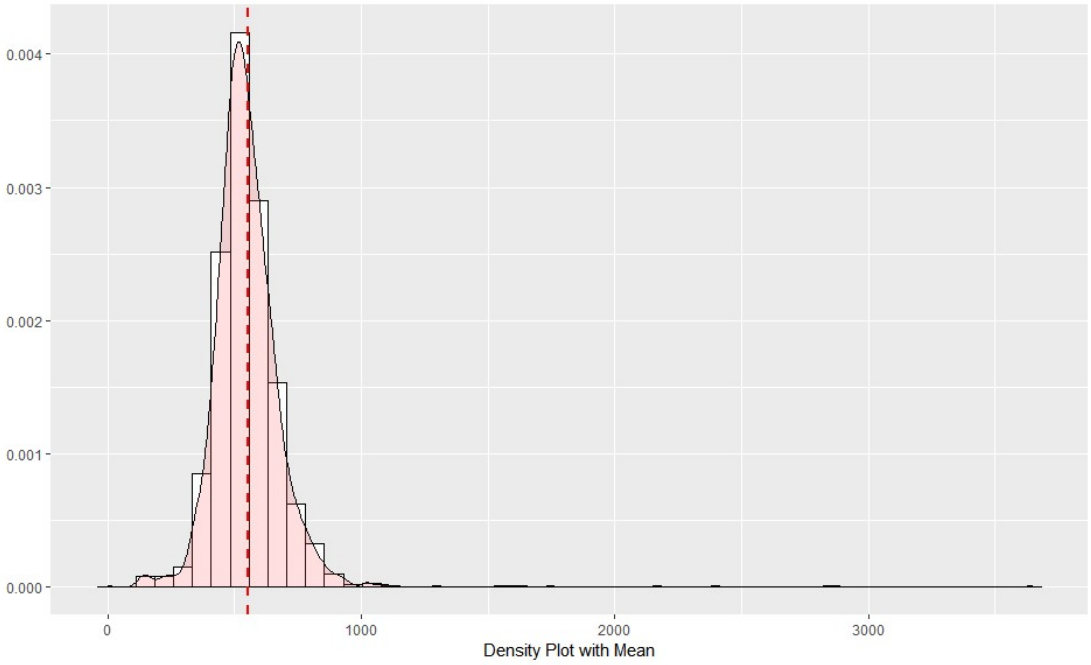
TEAM_PITCHING_BB:

This variable represents Number of walks given up by pitchers

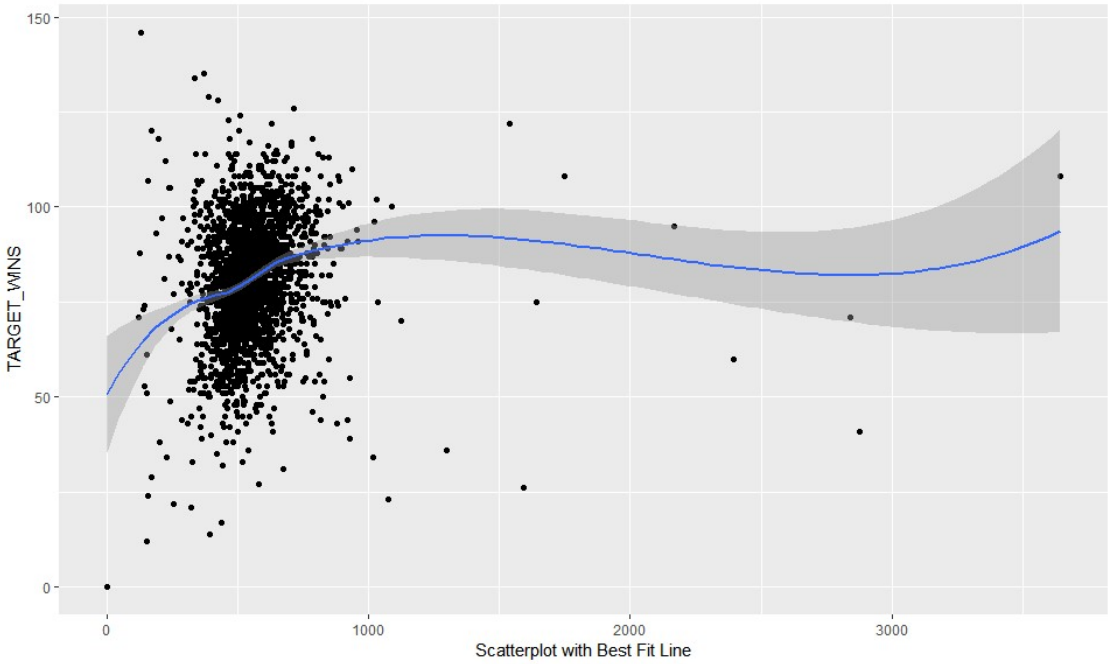
	Min	Median	Mean	SD	Max	Num_Zeros	Num_NaN
13	0	536.5	553	166	3645	1	0



Boxplot



Density Plot with Mean

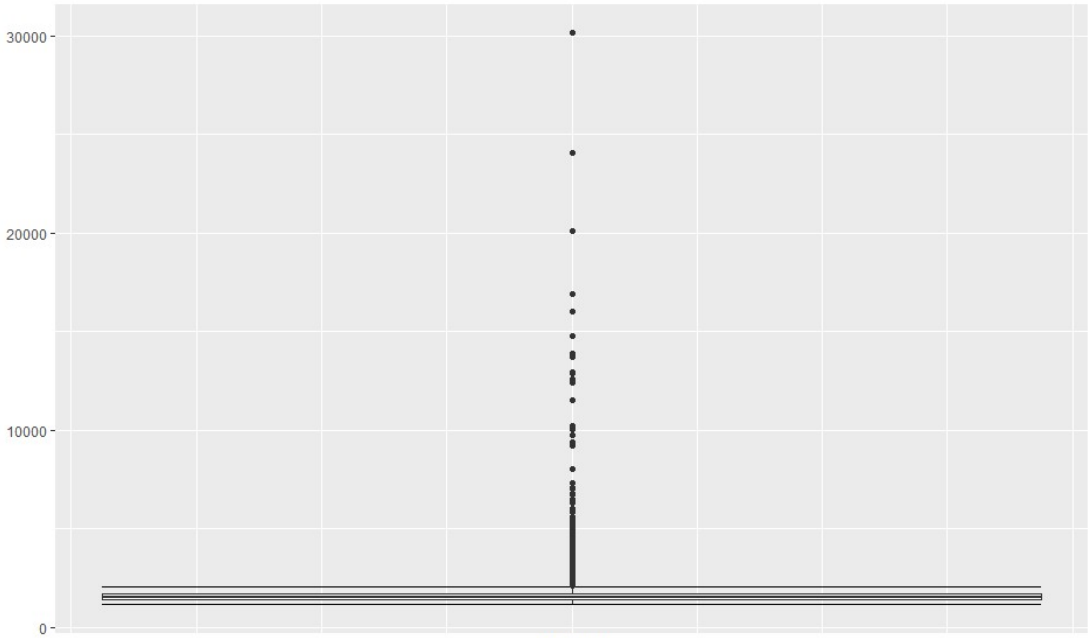


Data Overview: There are no missing values with the exception of record 1347 which will be deleted from model building. There are some unrealistic outliers.

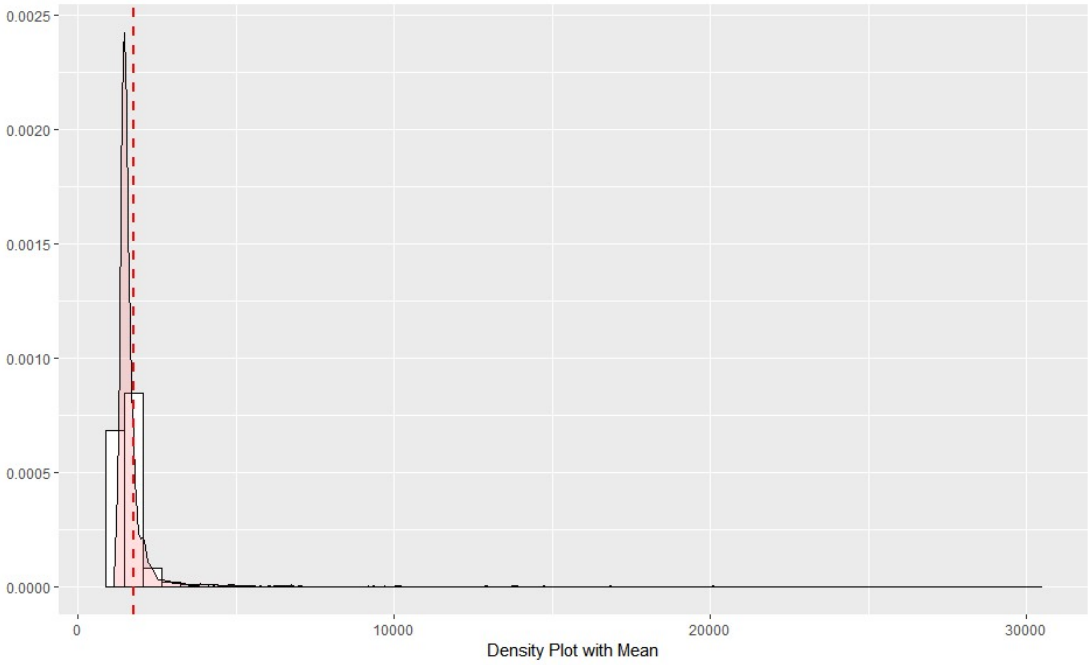
TEAM_PITCHING_H:

This variable represents Number of base hits given up by pitchers

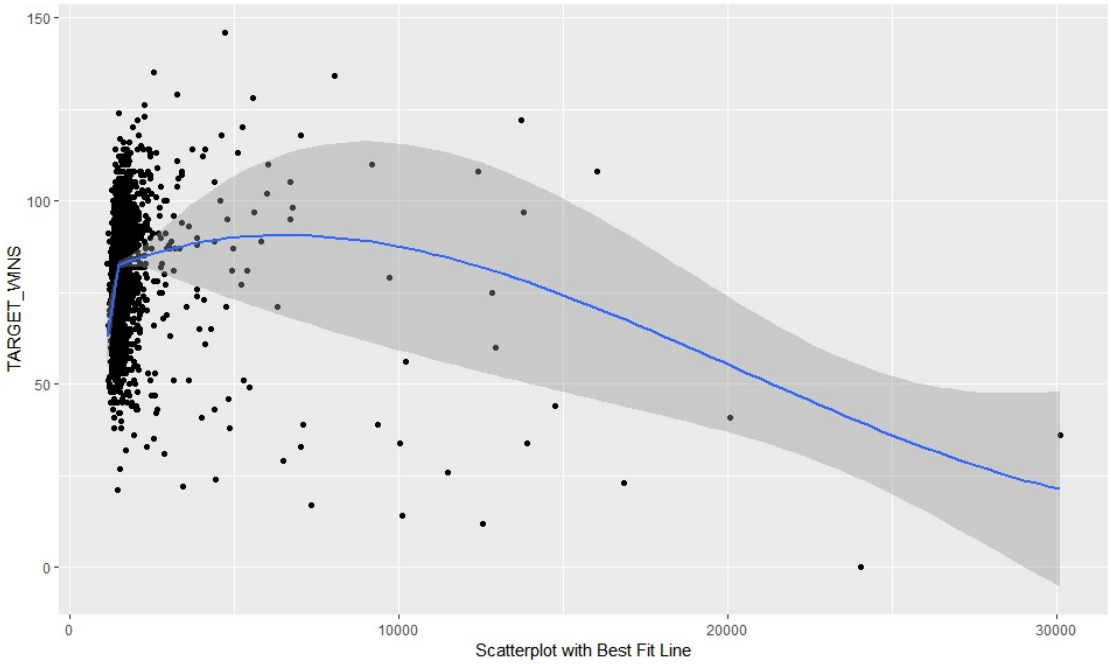
```
    Min Median Mean   SD   Max Num_Zeros Num_NaN
11 1137   1518 1779 1407 30132         0       0
```



Boxplot



Density Plot with Mean

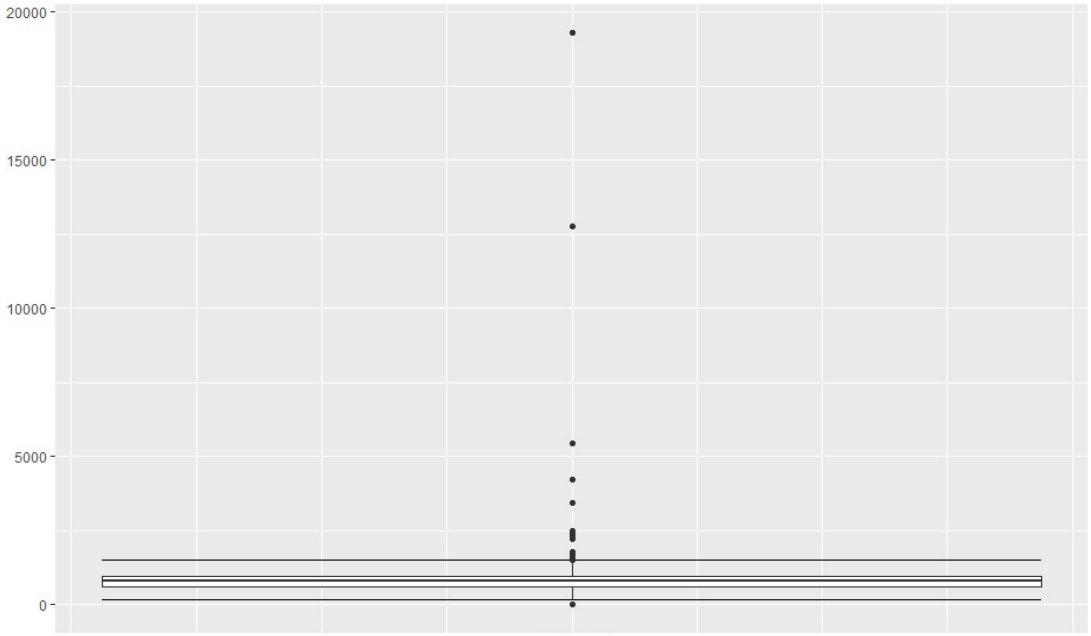


Data Overview: Similar to `TEAM_PITCHING_BB` above, there are no missing value, but there issues with outliers. Based on visualizations, this variable will be capped at 13,000 and any value over this will be set to this cap.

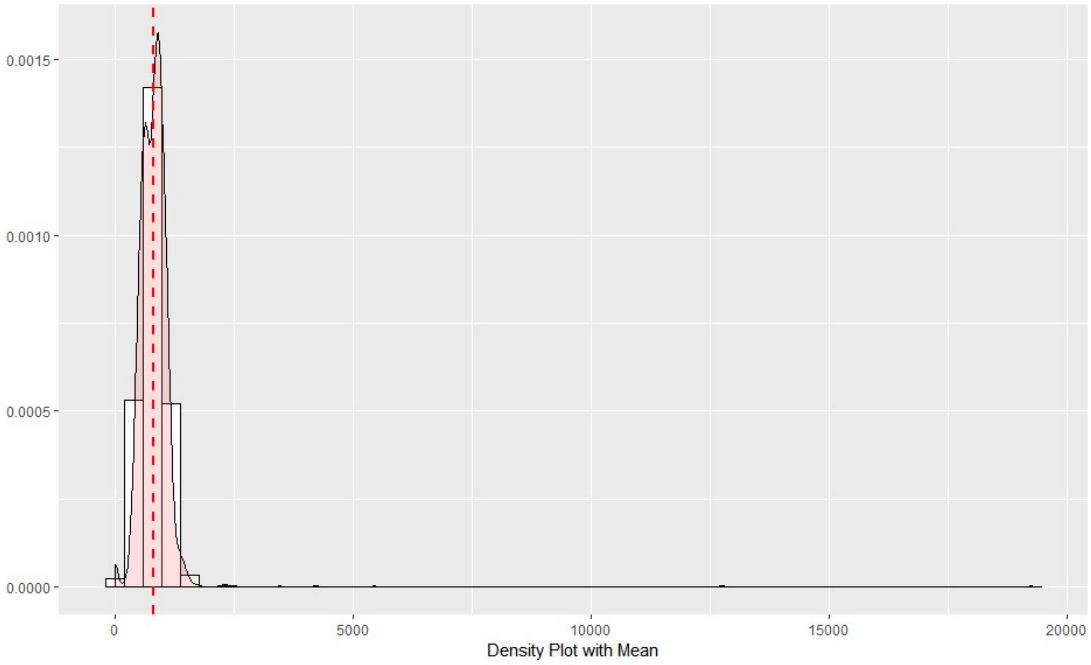
TEAM_PITCHING_SO:

This variable represents Number of strikeouts by pitchers

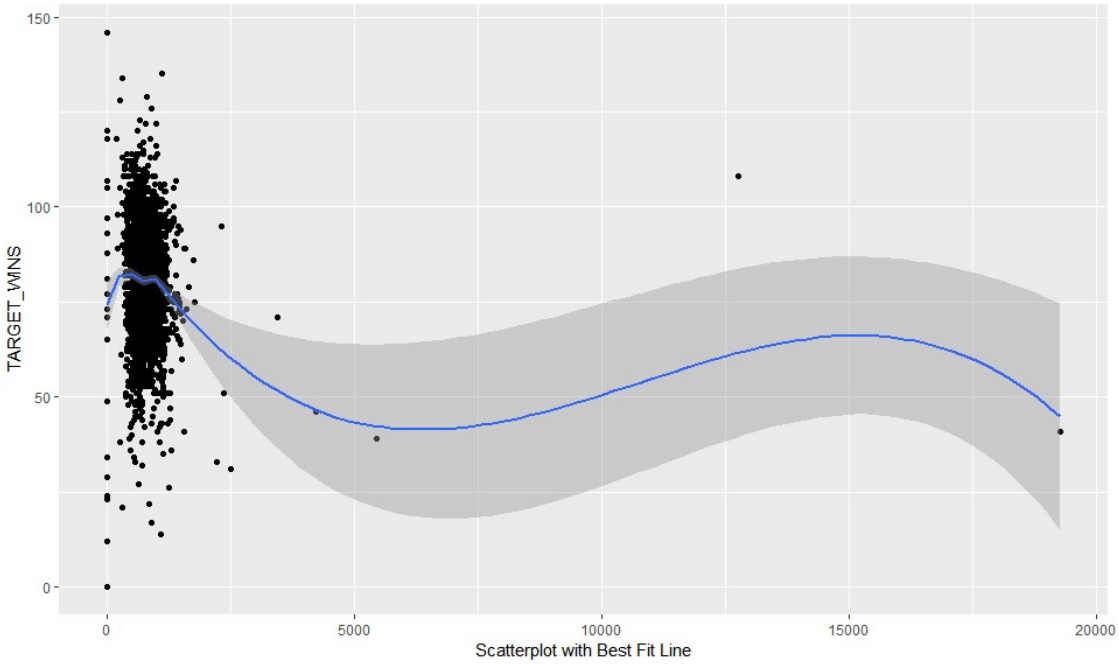
```
   Min Median Mean  SD   Max Num_Zeros Num_NaN
14   0  813.5  818 553 19278        20     102
```



Boxplot



Density Plot with Mean

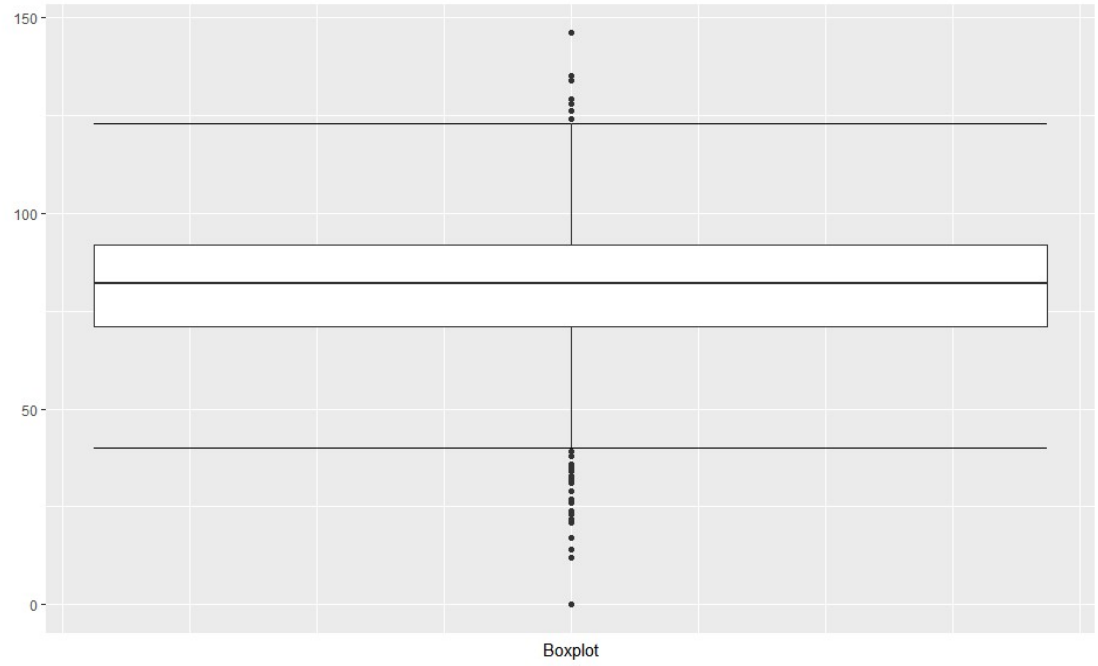


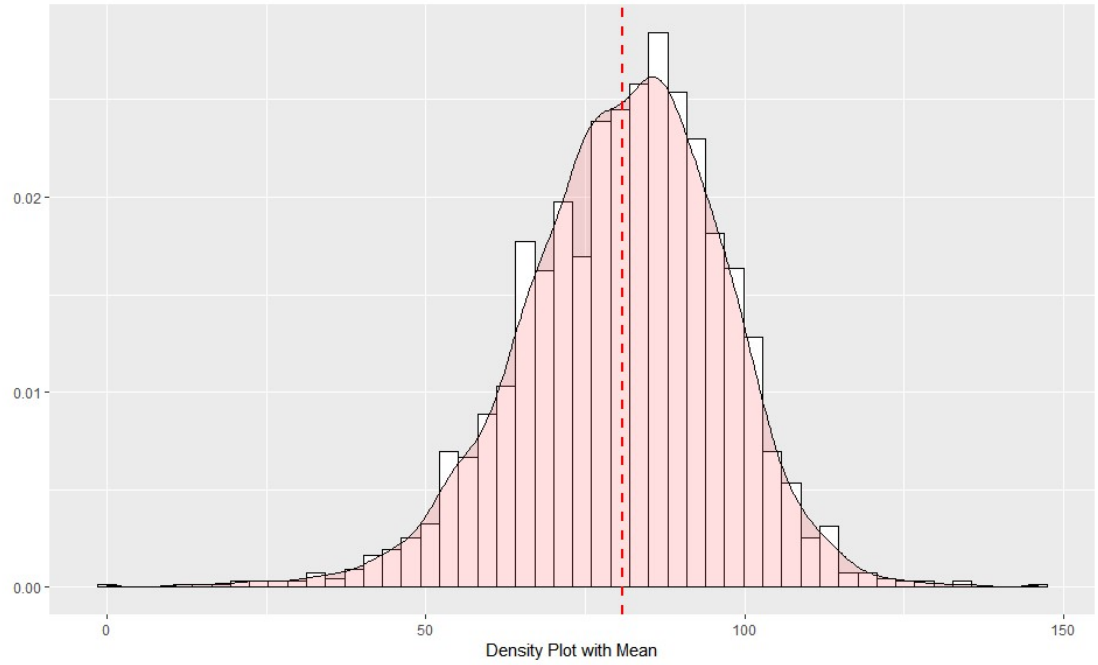
Data Overview: This variable has 122 missing or zero values. They can be imputed as needed. There is also an outlier issue as graph shows.

TARGET_WINS:

This variable represents Number of wins **(Outcome)**

```
  Min Median Mean SD Max Num_Zeros Num_NaN
1   0     82   81 16 146         1       0
```





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

DATA PREPARATION

Fixing Missing/Zero Values- TRAINING DATA

First I will remove the invalid data and prep it for imputation. I will drop the hit by pitcher variable from the dataset.

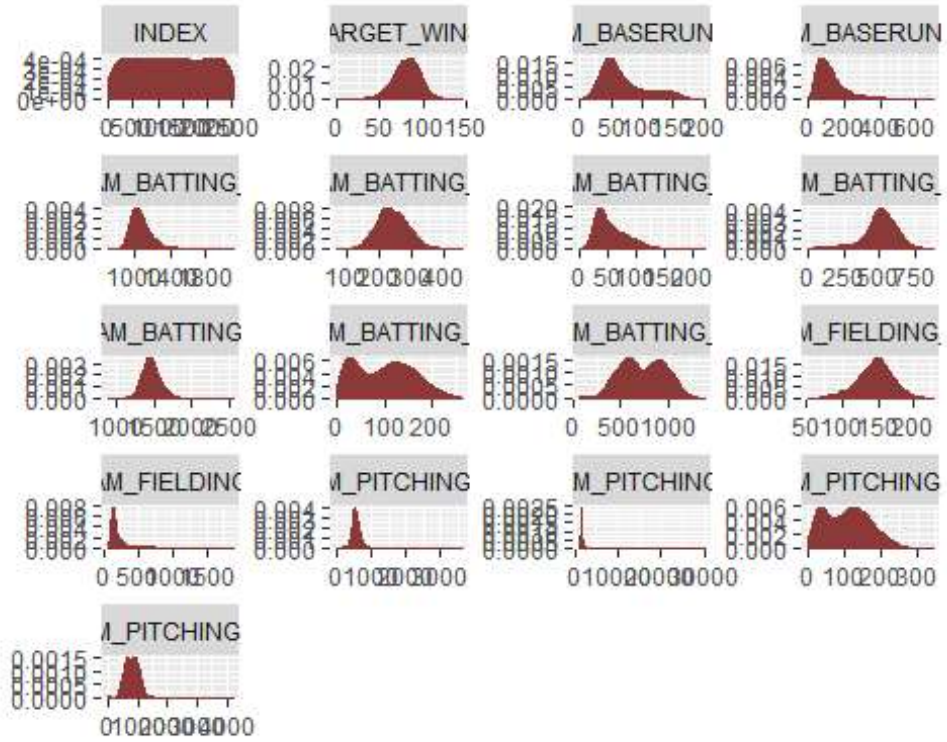
KNN imputation - TRAINING DATA

Feature Engineering - TRAINING DATA

The batting singles is not included but I can back it out of the hits.

Model Data Look

Here's what the data look like after imputation and correction:




```
    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.   :  66
1st Qu.: 34.00   1st Qu.: 42.00   1st Qu.:451.0   1st Qu.: 554
Median : 47.00   Median :102.00   Median :512.0   Median : 733
Mean   : 55.25   Mean   : 99.61   Mean   :501.6   Mean   : 735
3rd Qu.: 72.00   3rd Qu.:147.00   3rd Qu.:580.0   3rd Qu.: 925
Max.   :223.00   Max.   :264.00   Max.   :878.0   Max.   :1399
TEAM_BASERUN_SB TEAM_BASERUN_CS TEAM_PITCHING_H TEAM_PITCHING_HR
Min.   :  0.0   Min.   :  0.0   Min.   : 1137   Min.   :  0.0
1st Qu.: 67.0   1st Qu.: 43.0   1st Qu.: 1419   1st Qu.: 50.0
```

```
 Median :104.0   Median : 58.0   Median : 1518   Median :107.0
 Mean   :124.7   Mean   : 69.7   Mean   : 1779   Mean   :105.7
 3rd Qu.:153.2   3rd Qu.: 89.0   3rd Qu.: 1682   3rd Qu.:150.0
 Max.   :697.0   Max.   :201.0   Max.   :30132   Max.   :343.0
 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.: 618.5   1st Qu.: 127.0   1st Qu.:130.0
 Median : 536.5   Median : 797.0   Median : 159.0   Median :147.0
 Mean   : 553.0   Mean   : 795.8   Mean   : 246.5   Mean   :145.4
 3rd Qu.: 611.0   3rd Qu.: 957.0   3rd Qu.: 249.2   3rd Qu.:162.0
 Max.   :3645.0   Max.   :4224.0   Max.   :1898.0   Max.   :228.0
 TEAM_BATTING_1B
 Min.   : 709.0
 1st Qu.: 990.8
 Median :1050.0
 Mean   :1073.2
 3rd Qu.:1129.0
 Max.   :2112.0
```

BUILD MODELS

I split training dat 70 %training purposes and 30 % for testing purposes.

Model 1

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

```
Call:
lm(formula = TARGET_WINS ~ TEAM_BATTING_H + TEAM_BATTING_BB +
    TEAM_FIELDING_E, data = train)

Residuals:
    Min      1Q  Median      3Q     Max
-50.366  -9.091  -0.009   9.193  50.035

Coefficients:
                Estimate Std. Error t value Pr(>|t|)
(Intercept)     0.191724   4.089763   0.047    0.963
TEAM_BATTING_H  0.050059   0.002587  19.349  < 2e-16 ***
TEAM_BATTING_BB 0.020486   0.003840   5.335 1.09e-07 ***
TEAM_FIELDING_E -0.012959   0.002100  -6.170 8.64e-10 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 13.82 on 1591 degrees of freedom
Multiple R-squared:  0.2496,    Adjusted R-squared:  0.2482
F-statistic: 176.4 on 3 and 1591 DF,  p-value: < 2.2e-16
```

All variables are significant, but the R^2 value is relatively small at 0.2427.

Model 2

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

```
Call:
lm(formula = TARGET_WINS ~ TEAM_BATTING_1B + TEAM_BATTING_2B +
    TEAM_BATTING_3B + TEAM_BATTING_HR + TEAM_BATTING_BB + TEAM_FIELDING_E,
    data = train)

Residuals:
    Min      1Q  Median      3Q     Max
-49.132  -8.827   0.021   8.908  58.695

Coefficients:
                 Estimate Std. Error t value Pr(>|t|)
(Intercept)      4.881325   4.207715   1.160    0.246
TEAM_BATTING_1B  0.045001   0.003881  11.594  < 2e-16 ***
TEAM_BATTING_2B  0.015157   0.008794   1.724    0.085 .
TEAM_BATTING_3B  0.193543   0.018105  10.690  < 2e-16 ***
TEAM_BATTING_HR  0.092140   0.009316   9.890  < 2e-16 ***
TEAM_BATTING_BB  0.016433   0.003871   4.245 2.31e-05 ***
TEAM_FIELDING_E -0.016851   0.002305  -7.309 4.23e-13 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 13.52 on 1588 degrees of freedom
Multiple R-squared:  0.2832,    Adjusted R-squared:  0.2805
F-statistic: 104.6 on 6 and 1588 DF,  p-value: < 2.2e-16
```

All variables are still significant and R^2 is slightly improved at 0.2628.

Model 3 :Higher Order Stepwise Regression

For the third model I will use a stepwise regression method using a backwards elimination process. I also introduce some higher order polynomial variables.

```
Call:
lm(formula = poly_call[2], data = train)

Residuals:
    Min      1Q  Median      3Q     Max
-38.548  -8.039  -0.351   7.703  63.122

Coefficients:
                        Estimate Std. Error t value Pr(>|t|)
```

```
(Intercept)             8.463e+01  1.803e+01   4.694 2.91e-06 ***
TEAM_BATTING_2B         1.246e-01  5.604e-02   2.223 0.026354 *
TEAM_BATTING_3B         1.562e-01  2.050e-02   7.619 4.39e-14 ***
TEAM_BATTING_BB        -2.367e-01  3.142e-02  -7.533 8.34e-14 ***
TEAM_BATTING_SO         4.425e-02  1.087e-02   4.070 4.93e-05 ***
TEAM_BASERUN_SB         2.578e-02  6.347e-03   4.061 5.12e-05 ***
TEAM_PITCHING_H        -7.289e-03  2.086e-03  -3.494 0.000490 ***
TEAM_PITCHING_HR        1.645e-01  2.695e-02   6.104 1.30e-09 ***
TEAM_PITCHING_BB        6.279e-02  1.525e-02   4.116 4.05e-05 ***
TEAM_PITCHING_SO        1.107e-02  5.053e-03   2.191 0.028583 *
TEAM_FIELDING_E        -6.283e-02  7.523e-03  -8.352  < 2e-16 ***
TEAM_FIELDING_DP       -8.851e-02  1.647e-02  -5.375 8.83e-08 ***
TEAM_BATTING_1B        -4.447e-02  2.481e-02  -1.792 0.073251 .
I(TEAM_BATTING_2B^2)   -1.639e-04  1.102e-04  -1.487 0.137087
I(TEAM_BATTING_HR^2)    4.864e-04  1.091e-04   4.456 8.92e-06 ***
I(TEAM_BATTING_BB^2)    1.949e-04  2.279e-05   8.555  < 2e-16 ***
I(TEAM_BATTING_SO^2)   -3.712e-05  6.471e-06  -5.736 1.16e-08 ***
I(TEAM_BASERUN_CS^2)    5.412e-04  1.051e-04   5.149 2.95e-07 ***
I(TEAM_PITCHING_H^2)    2.052e-07  7.929e-08   2.588 0.009749 **
I(TEAM_PITCHING_HR^2) -5.886e-04  1.109e-04  -5.309 1.26e-07 ***
I(TEAM_PITCHING_BB^2) -1.309e-05  4.293e-06  -3.048 0.002338 **
I(TEAM_PITCHING_SO^2) -2.983e-06  1.267e-06  -2.354 0.018713 *
I(TEAM_FIELDING_E^2)    1.834e-05  5.211e-06   3.519 0.000446 ***
I(TEAM_BATTING_1B^2)    4.353e-05  1.029e-05   4.231 2.46e-05 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 12.41 on 1571 degrees of freedom
Multiple R-squared:  0.4032,    Adjusted R-squared:  0.3944
F-statistic: 46.14 on 23 and 1571 DF,  p-value: < 2.2e-16
```

This model has the highest adjusted R-squared value at 0.3944 .Some variables p-values are not in 95 % siginificant level but they are in 90 % significant level which is acceptable.

Model 4

For the fourth model, Variables were selected either based on correlation information from the first section. The following model has R^2 values of 0.2606, which is relatively close to the fourth model; however, this model has fewer variables and may be preferential because of its simplicity.

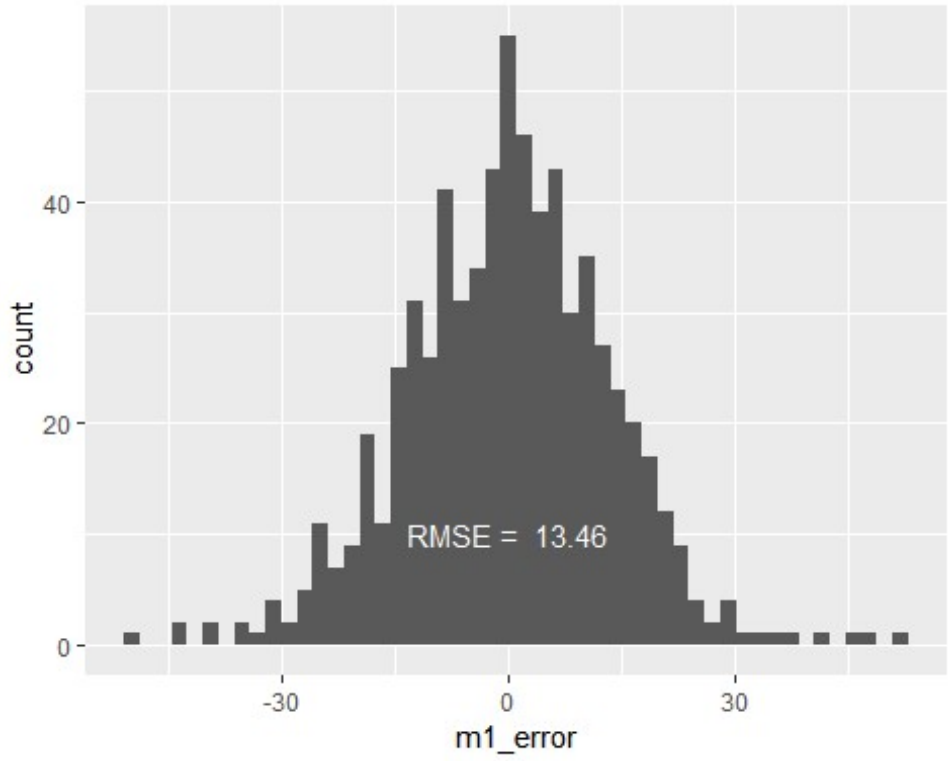
```
Call:
lm(formula = TARGET_WINS ~ TEAM_BATTING_SO + TEAM_BATTING_3B +
    TEAM_BATTING_HR + TEAM_BASERUN_SB + TEAM_FIELDING_E_LOG *
    TEAM_PITCHING_H, data = train)

Residuals:
    Min      1Q  Median      3Q     Max
-67.376  -8.286  -0.003   8.323  75.209
```

```
Coefficients:
                                  Estimate Std. Error t value Pr(>|t|)
(Intercept)                     73.0506213  9.3014936   7.854 7.39e-15
***
TEAM_BATTING_SO                 -0.0190443  0.0024547  -7.758 1.53e-14
***
TEAM_BATTING_3B                  0.1886994  0.0210281   8.974  < 2e-16
***
TEAM_BATTING_HR                  0.1145298  0.0098324  11.648  < 2e-16
***
TEAM_BASERUN_SB                  0.0461917  0.0049564   9.320  < 2e-16
***
TEAM_FIELDING_E_LOG             -3.5210552  1.4774803  -2.383   0.0173 *
TEAM_PITCHING_H                  0.0312198  0.0049147   6.352 2.76e-10
***
TEAM_FIELDING_E_LOG:TEAM_PITCHING_H -0.0043771  0.0006843  -6.397 2.09e-10
***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 13.55 on 1587 degrees of freedom
Multiple R-squared:  0.2813,    Adjusted R-squared:  0.2781
F-statistic: 88.73 on 7 and 1587 DF,  p-value: < 2.2e-16
```

SELECT MODELS

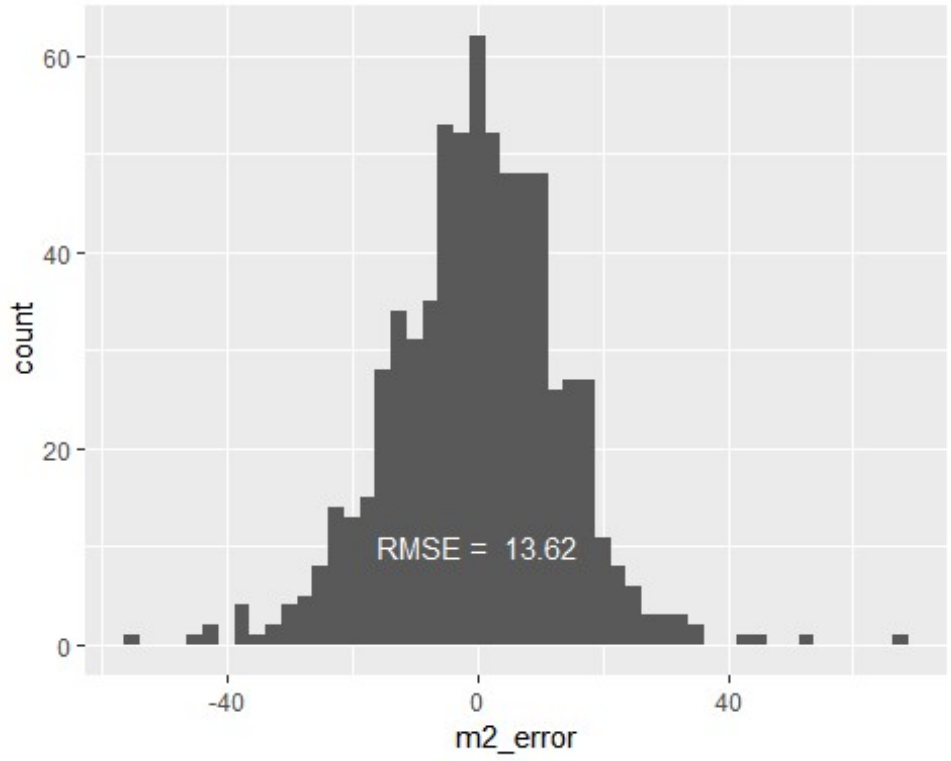
In order to select which model is the “best” I will test it against a validation (test) set. I will examine the difference between the predicted and actual values.

Model 1 Results



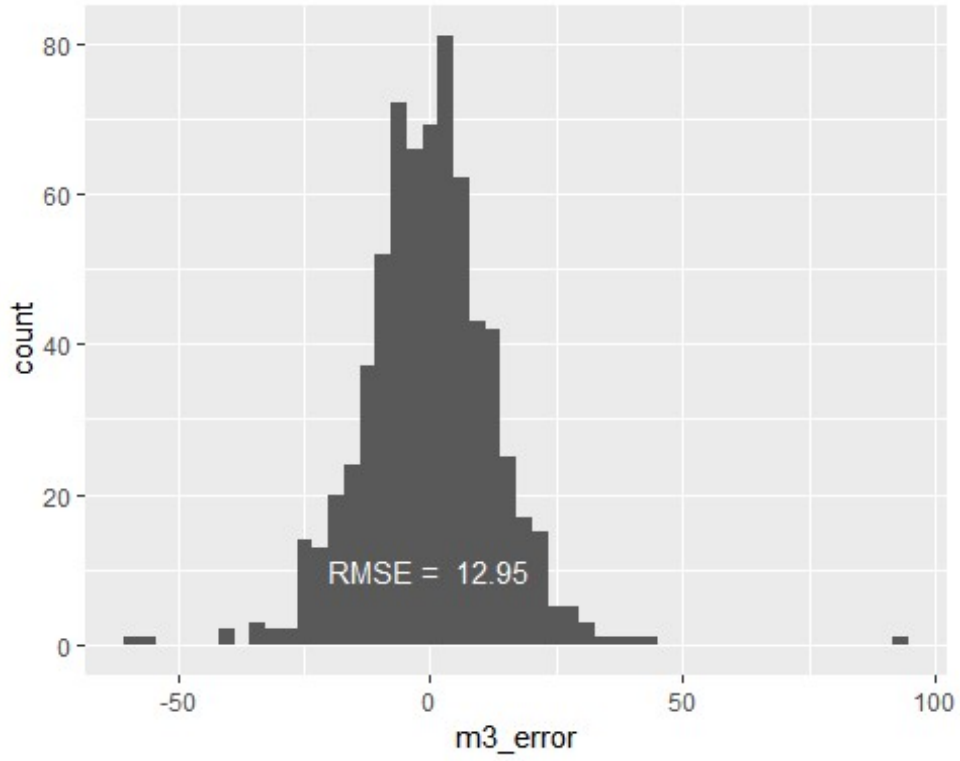

```
     Min.   1st Qu.    Median      Mean   3rd Qu.      Max. 
-49.65700  -8.58739   0.01945  -0.10902   8.50826  52.08518 
```

Model 2 Results



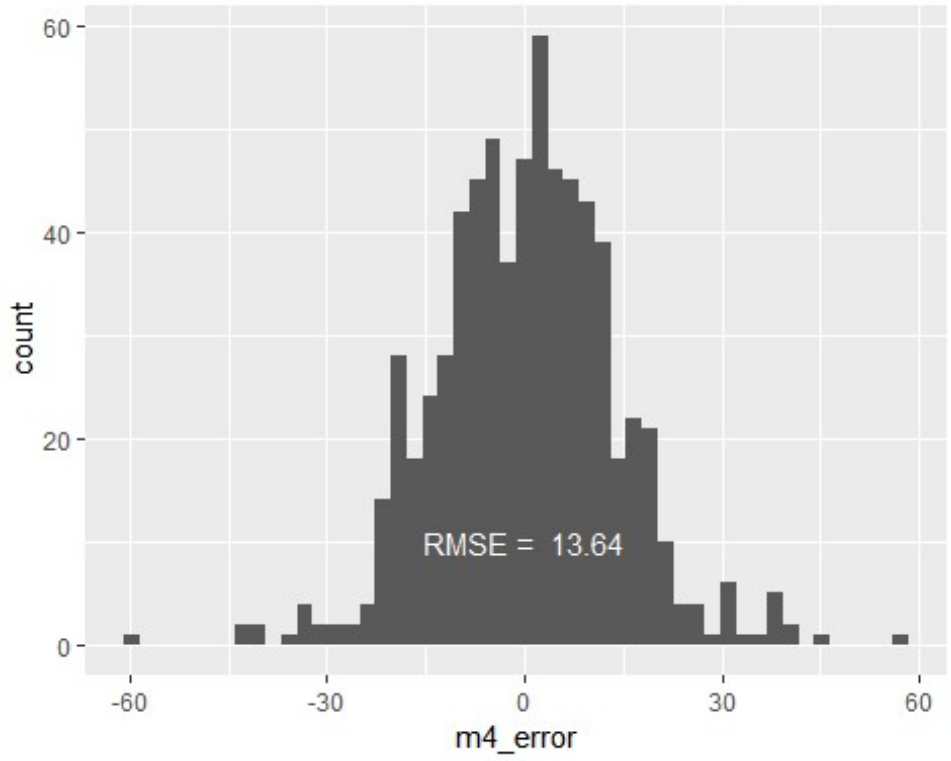

```
     Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
-55.27014  -8.12926   0.26431  -0.00126   8.53172  67.37311
```

Model 3 Results




```
     Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
-59.70232  -7.77596  -0.03296  -0.27844   7.16360  92.51153
```

Model 4 Results

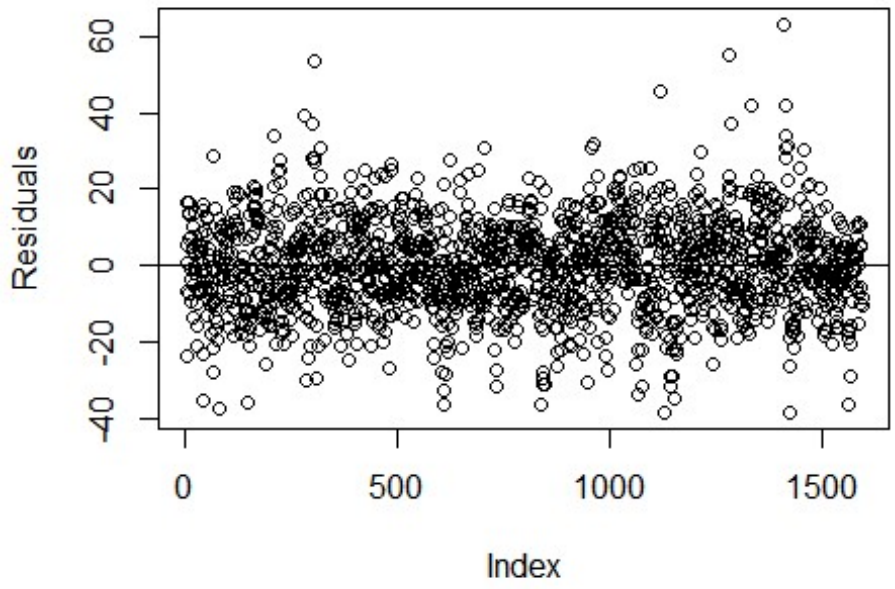



```
    Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
-59.8164  -8.4904   0.6244   0.1682   8.7404  56.9616
```

Based on R^2 value and RMSE results, the third model (M3) was selected for further analysis. This model also has the lowest AIC score.

```
   df      AIC
m1  5 12910.59
m2  8 12843.45
m3 25 12585.36
m4  9 12849.71


Call:
lm(formula = poly_call[2], data = train)

Residuals:
    Min      1Q  Median      3Q     Max
-38.548  -8.039  -0.351   7.703  63.122

Coefficients:
                       Estimate Std. Error t value Pr(>|t|)
(Intercept)           8.463e+01  1.803e+01   4.694 2.91e-06 ***
TEAM_BATTING_2B       1.246e-01  5.604e-02   2.223 0.026354 *
TEAM_BATTING_3B       1.562e-01  2.050e-02   7.619 4.39e-14 ***
```

```
TEAM_BATTING_BB       -2.367e-01  3.142e-02  -7.533 8.34e-14 ***
TEAM_BATTING_SO        4.425e-02  1.087e-02   4.070 4.93e-05 ***
TEAM_BASERUN_SB        2.578e-02  6.347e-03   4.061 5.12e-05 ***
TEAM_PITCHING_H       -7.289e-03  2.086e-03  -3.494 0.000490 ***
TEAM_PITCHING_HR       1.645e-01  2.695e-02   6.104 1.30e-09 ***
TEAM_PITCHING_BB       6.279e-02  1.525e-02   4.116 4.05e-05 ***
TEAM_PITCHING_SO       1.107e-02  5.053e-03   2.191 0.028583 *
TEAM_FIELDING_E       -6.283e-02  7.523e-03  -8.352  < 2e-16 ***
TEAM_FIELDING_DP      -8.851e-02  1.647e-02  -5.375 8.83e-08 ***
TEAM_BATTING_1B       -4.447e-02  2.481e-02  -1.792 0.073251 .
I(TEAM_BATTING_2B^2)  -1.639e-04  1.102e-04  -1.487 0.137087
I(TEAM_BATTING_HR^2)   4.864e-04  1.091e-04   4.456 8.92e-06 ***
I(TEAM_BATTING_BB^2)   1.949e-04  2.279e-05   8.555  < 2e-16 ***
I(TEAM_BATTING_SO^2)  -3.712e-05  6.471e-06  -5.736 1.16e-08 ***
I(TEAM_BASERUN_CS^2)   5.412e-04  1.051e-04   5.149 2.95e-07 ***
I(TEAM_PITCHING_H^2)   2.052e-07  7.929e-08   2.588 0.009749 **
I(TEAM_PITCHING_HR^2) -5.886e-04  1.109e-04  -5.309 1.26e-07 ***
I(TEAM_PITCHING_BB^2) -1.309e-05  4.293e-06  -3.048 0.002338 **
I(TEAM_PITCHING_SO^2) -2.983e-06  1.267e-06  -2.354 0.018713 *
I(TEAM_FIELDING_E^2)   1.834e-05  5.211e-06   3.519 0.000446 ***
I(TEAM_BATTING_1B^2)   4.353e-05  1.029e-05   4.231 2.46e-05 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 12.41 on 1571 degrees of freedom
Multiple R-squared:  0.4032,    Adjusted R-squared:  0.3944
F-statistic: 46.14 on 23 and 1571 DF,  p-value: < 2.2e-16
```



Prediction

In order to make prediction, I need to impute missing values on “evalution” dataset by using the same as training imputation.

Fixing Missing/Zero Values

First I will remove the invalid data and prep it for imputation. I will drop the hit by pitcher variable from the dataset.

KNN imputation

Feature Engineering

The batting singles is not included but I can back it out of the hits.

Prediction

	Index	Predicted Wins	CI Lower	CI Upper
1	9	59	56	62
2	10	62	59	64
3	14	71	69	73
4	47	86	84	87
5	60	76	69	82
6	63	70	67	74
7	74	72	68	75
8	83	74	71	76

9	98	69	66	71
10	120	71	70	73
11	123	67	65	69
12	135	83	81	85
13	138	84	81	86
14	140	81	79	83
15	151	82	80	84
16	153	77	75	79
17	171	71	69	73
18	184	78	76	79
19	193	68	65	70
20	213	87	84	90
21	217	80	78	82
22	226	85	83	87
23	230	85	83	87
24	241	72	70	73
25	291	82	81	84
26	294	86	84	88
27	300	46	38	54
28	348	72	70	74
29	350	79	77	82
30	357	71	69	74
31	367	97	95	100
32	368	88	86	89
33	372	92	89	94
34	382	96	93	99
35	388	81	79	82
36	396	87	85	89
37	398	76	75	78
38	403	91	89	94
39	407	92	88	96
40	410	91	88	93
41	412	83	81	85
42	414	92	90	94
43	436	30	20	41
44	440	114	109	120
45	476	90	87	93
46	479	85	82	88
47	481	92	90	95
48	501	81	79	83
49	503	67	65	69
50	506	79	77	82
51	519	75	73	77
52	522	81	79	83
53	550	75	74	77
54	554	75	74	77
55	566	74	73	75
56	578	80	78	81
57	596	87	85	90
58	599	76	73	78

59	605	56	51	60
60	607	76	74	78
61	614	83	80	85
62	644	84	81	88
63	692	87	86	89
64	699	85	82	89
65	700	85	83	88
66	716	88	84	92
67	721	76	74	77
68	722	81	78	83
69	729	74	71	76
70	731	90	87	92
71	746	81	78	84
72	763	68	65	70
73	774	80	78	82
74	776	89	86	92
75	788	83	80	85
76	789	82	80	85
77	792	86	85	88
78	811	81	80	83
79	835	72	70	74
80	837	75	73	77
81	861	79	76	82
82	862	88	85	90
83	863	94	90	98
84	871	74	72	77
85	879	90	88	91
86	887	78	76	80
87	892	81	79	83
88	904	85	84	87
89	909	90	87	92
90	925	90	88	93
91	940	73	70	76
92	951	60	42	78
93	976	63	60	66
94	981	85	82	88
95	983	82	79	84
96	984	81	79	83
97	989	99	96	103
98	995	104	101	107
99	1000	86	84	88
100	1001	87	84	89
101	1007	78	76	80
102	1016	69	67	71
103	1027	84	82	85
104	1033	85	83	87
105	1070	71	68	75
106	1081	74	71	78
107	1084	47	43	51
108	1098	74	72	77

109	1150	87	85	89
110	1160	56	52	59
111	1169	86	84	88
112	1172	85	83	88
113	1174	92	90	94
114	1176	92	90	94
115	1178	84	83	86
116	1184	79	77	81
117	1193	86	84	88
118	1196	82	80	83
119	1199	74	72	76
120	1207	68	65	71
121	1218	79	76	82
122	1223	63	60	66
123	1226	68	65	70
124	1227	67	62	71
125	1229	67	64	69
126	1241	84	81	86
127	1244	86	83	88
128	1246	76	74	77
129	1248	88	86	90
130	1249	92	90	94
131	1253	85	83	87
132	1261	77	75	79
133	1305	76	74	78
134	1314	80	77	83
135	1323	85	83	87
136	1328	68	65	72
137	1353	77	75	78
138	1363	77	76	79
139	1371	90	88	93
140	1372	82	80	84
141	1389	67	64	70
142	1393	69	66	73
143	1421	88	86	90
144	1431	72	71	74
145	1437	73	71	75
146	1442	74	73	76
147	1450	76	75	77
148	1463	79	78	81
149	1464	79	77	81
150	1470	83	82	85
151	1471	84	82	86
152	1484	79	78	81
153	1495	15	-10	40
154	1507	71	68	73
155	1514	74	72	76
156	1526	67	65	70
157	1549	87	85	90
158	1552	61	58	64

159	1556	88	85	91
160	1564	67	64	70
161	1585	110	107	113
162	1586	123	119	127
163	1590	96	94	99
164	1591	114	110	117
165	1592	108	105	111
166	1603	93	91	95
167	1612	85	83	87
168	1634	81	79	83
169	1645	72	70	73
170	1647	80	79	81
171	1673	86	84	89
172	1674	88	86	90
173	1687	78	76	80
174	1688	87	84	89
175	1700	80	78	82
176	1708	75	73	77
177	1713	84	81	86
178	1717	71	69	73
179	1721	75	73	76
180	1730	79	78	80
181	1737	91	88	94
182	1748	87	85	89
183	1749	85	84	87
184	1763	86	84	88
185	1768	90	81	100
186	1778	85	80	90
187	1780	83	80	86
188	1782	52	46	57
189	1784	54	51	57
190	1794	108	104	113
191	1803	64	61	67
192	1804	78	75	80
193	1819	82	79	84
194	1832	75	73	77
195	1833	77	75	79
196	1844	61	59	63
197	1847	75	74	77
198	1854	92	89	94
199	1855	82	80	83
200	1857	86	85	88
201	1864	72	69	74
202	1865	78	77	80
203	1869	73	71	76
204	1880	98	94	101
205	1881	81	79	82
206	1882	86	84	87
207	1894	79	77	81
208	1896	76	75	78

209	1916	74	72	77
210	1918	63	60	66
211	1921	96	92	99
212	1926	85	82	88
213	1938	82	79	84
214	1979	64	63	66
215	1982	70	67	72
216	1987	85	83	87
217	1997	81	79	83
218	2004	97	95	100
219	2011	78	76	79
220	2015	79	77	81
221	2022	77	75	79
222	2025	73	71	75
223	2027	80	78	81
224	2031	74	72	76
225	2036	95	86	103
226	2066	75	73	76
227	2073	81	79	82
228	2087	76	74	78
229	2092	82	81	84
230	2125	63	59	67
231	2148	71	68	75
232	2162	95	92	97
233	2191	84	82	86
234	2203	88	86	90
235	2218	80	78	81
236	2221	73	71	74
237	2225	80	78	82
238	2232	78	77	80
239	2267	82	79	85
240	2291	71	68	74
241	2299	87	86	89
242	2317	86	84	88
243	2318	80	79	82
244	2353	82	80	84
245	2403	59	57	62
246	2411	85	83	87
247	2415	81	79	82
248	2424	85	84	87
249	2441	73	71	74
250	2464	83	80	85
251	2465	78	76	80
252	2472	66	61	71
253	2481	90	87	92
254	2487	54	37	71
255	2500	68	66	70
256	2501	82	79	84
257	2520	80	78	82

```
258  2521              81        79        83
259  2525              77        74        80
```

APPENDIX: R Script

Read in the training data

training <- read.csv("https://raw.githubusercontent.com/omerozeren/DATA621/master/moneyball-training-data.csv") # Read in the evaluation data evaluation <- read.csv("https://raw.githubusercontent.com/omerozeren/DATA621/master/moneyball-evaluation-data.csv")

sumtable = data.frame(Variable = character(), Min = integer(), Median = integer(), Mean = double(), SD = double(), Max = integer(), Num_Zeros = integer(), Num_NaN = integer()) for (i in 2:17) { sumtable <- rbind(sumtable, data.frame(Variable = colnames(training)[i], Min = min(training[,i], na.rm=TRUE), Median = median(training[,i], na.rm=TRUE), Mean = round(mean(training[,i], na.rm=TRUE)), SD = round(sd(training[,i], na.rm=TRUE)), Max = max(training[,i], na.rm=TRUE), Num_Zeros = length(which(training[,i]==0)), Num_NaN = sum(is.na(training[,i]))))) } colnames(sumtable) <- c("", "Min", "Median", "Mean", "SD", "Max","Num_Zeros", "Num_NaN") sumtable

cm <- cor(training, use="pairwise.complete.obs") cm <- cm[2:17,2:17] names <- c("Wins", "H", "2B", "3B", "HR", "BB", "SO", "SB", "CS", "HBP", "P-H", "P-HR", "P-BB", "P-SO", "E", "DP") colnames(cm) <- names; rownames(cm) <- names round(cm,2)

cm <- cor(training, use="pairwise.complete.obs") cm <- cm[2:17,2:17] names <- c("Wins", "H", "2B", "3B", "HR", "BB", "SO", "SB", "CS", "HBP", "P-H", "P-HR", "P-BB", "P-SO", "E", "DP") colnames(cm) <- names; rownames(cm) <- names corrplot(cm, method = "color", type = "upper", tl.col = "black", diag = FALSE)

training %>% gather(variable, value, -TARGET_WINS) %>% ggplot(., aes(value, TARGET_WINS)) + geom_point(fill = "indianred4", color="indianred4") + geom_smooth(method = "lm", se = FALSE, color = "black") + facet_wrap(~variable, scales ="free", ncol = 4) + labs(x = element_blank(), y = "Wins")

sumtable[sumtable[,1]=="TEAM_BATTING_H",2:8]

Boxplot

ggplot(training, aes(x = 1, y = TEAM_BATTING_H)) + stat_boxplot(geom ='errorbar') + geom_boxplot() + xlab("Boxplot") + ylab("") + theme(axis.text.x=element_blank(), axis.ticks.x=element_blank())

Density plot

ggplot(training, aes(x = TEAM_BATTING_H)) + geom_histogram(aes(y=..density..), colour="black", fill="white",bins=50) + geom_density(alpha=.2, fill="#FF6666") + ylab("") + xlab("Density Plot with Mean") + geom_vline(aes(xintercept=mean(TEAM_BATTING_H, na.rm=TRUE)), color="red", linetype="dashed", size=1)

Scatterplot

ggplot(data=training, aes(x=TEAM_BATTING_H, y=TARGET_WINS)) + geom_point() + geom_smooth(method = "loess") + xlab("Scatterplot with Best Fit Line")

TEAM_BATTING_2B:

sumtable[sumtable[,1]=="TEAM_BATTING_2B",2:8] # Boxplot ggplot(training, aes(x = 1, y = TEAM_BATTING_2B)) + stat_boxplot(geom ='errorbar') + geom_boxplot() + xlab("Boxplot") + ylab("") + theme(axis.text.x=element_blank(), axis.ticks.x=element_blank()) # Density plot ggplot(training, aes(x = TEAM_BATTING_2B)) + geom_histogram(aes(y=..density..), colour="black", fill="white",bins=50) + geom_density(alpha=.2, fill="#FF6666") + ylab("") + xlab("Density Plot with Mean") + geom_vline(aes(xintercept=mean(TEAM_BATTING_2B, na.rm=TRUE)), color="red", linetype="dashed", size=1) # Scatterplot ggplot(data=training, aes(x=TEAM_BATTING_2B, y=TARGET_WINS)) + geom_point() + geom_smooth(method = "loess") + xlab("Scatterplot with Best Fit Line")

TEAM_BATTING_3B:

sumtable[sumtable[,1]=="TEAM_BATTING_3B",2:8] # Boxplot ggplot(training, aes(x = 1, y = TEAM_BATTING_3B)) + stat_boxplot(geom ='errorbar') + geom_boxplot() + xlab("Boxplot") + ylab("") + theme(axis.text.x=element_blank(), axis.ticks.x=element_blank()) # Density plot ggplot(training, aes(x = TEAM_BATTING_3B)) + geom_histogram(aes(y=..density..), colour="black", fill="white",bins=50) + geom_density(alpha=.2, fill="#FF6666") + ylab("") + xlab("Density Plot with Mean") + geom_vline(aes(xintercept=mean(TEAM_BATTING_3B, na.rm=TRUE)), color="red", linetype="dashed", size=1) # Scatterplot ggplot(data=training, aes(x=TEAM_BATTING_3B, y=TARGET_WINS)) + geom_point() + geom_smooth(method = "loess") + xlab("Scatterplot with Best Fit Line")

TEAM_BATTING_HR:

sumtable[sumtable[,1]=="TEAM_BATTING_HR",2:8] # Boxplot ggplot(training, aes(x = 1, y = TEAM_BATTING_HR)) + stat_boxplot(geom ='errorbar') + geom_boxplot() + xlab("Boxplot") + ylab("") + theme(axis.text.x=element_blank(), axis.ticks.x=element_blank()) # Density plot ggplot(training, aes(x = TEAM_BATTING_HR)) + geom_histogram(aes(y=..density..), colour="black", fill="white",bins=50) + geom_density(alpha=.2, fill="#FF6666") + ylab("") + xlab("Density Plot with Mean") + geom_vline(aes(xintercept=mean(TEAM_BATTING_HR, na.rm=TRUE)), color="red",

linetype="dashed", size=1) # Scatterplot ggplot(data=training, aes(x=TEAM_BATTING_HR, y=TARGET_WINS)) + geom_point() + geom_smooth(method = "loess") + xlab("Scatterplot with Best Fit Line")

TEAM_BATTING_BB:

This variable represents Number of team walks

sumtable[sumtable[,1]=="TEAM_BATTING_BB",2:8] # Boxplot ggplot(training, aes(x = 1, y = TEAM_BATTING_BB)) + stat_boxplot(geom ='errorbar') + geom_boxplot() + xlab("Boxplot") + ylab("") + theme(axis.text.x=element_blank(), axis.ticks.x=element_blank()) # Density plot ggplot(training, aes(x = TEAM_BATTING_BB)) + geom_histogram(aes(y=..density..), colour="black", fill="white",bins=50) + geom_density(alpha=.2, fill="#FF6666") + ylab("") + xlab("Density Plot with Mean") + geom_vline(aes(xintercept=mean(TEAM_BATTING_BB, na.rm=TRUE)), color="red", linetype="dashed", size=1) # Scatterplot ggplot(data=training, aes(x=TEAM_BATTING_BB, y=TARGET_WINS)) + geom_point() + geom_smooth(method = "loess") + xlab("Scatterplot with Best Fit Line")

TEAM_BATTING_HBP:

sumtable[sumtable[,1]=="TEAM_BATTING_HBP",2:8] # Boxplot ggplot(training, aes(x = 1, y = TEAM_BATTING_HBP)) + stat_boxplot(geom ='errorbar') + geom_boxplot() + xlab("Boxplot") + ylab("") + theme(axis.text.x=element_blank(), axis.ticks.x=element_blank()) # Density plot ggplot(training, aes(x = TEAM_BATTING_HBP)) + geom_histogram(aes(y=..density..), colour="black", fill="white",bins=50) + geom_density(alpha=.2, fill="#FF6666") + ylab("") + xlab("Density Plot with Mean") + geom_vline(aes(xintercept=mean(TEAM_BATTING_HBP, na.rm=TRUE)), color="red", linetype="dashed", size=1) # Scatterplot ggplot(data=training, aes(x=TEAM_BATTING_HBP, y=TARGET_WINS)) + geom_point() + geom_smooth(method = "loess") + xlab("Scatterplot with Best Fit Line")

TEAM_BATTING_SO:

sumtable[sumtable[,1]=="TEAM_BATTING_SO",2:8] # Boxplot ggplot(training, aes(x = 1, y = TEAM_BATTING_SO)) + stat_boxplot(geom ='errorbar') + geom_boxplot() + xlab("Boxplot") + ylab("") + theme(axis.text.x=element_blank(), axis.ticks.x=element_blank()) # Density plot ggplot(training, aes(x = TEAM_BATTING_SO)) + geom_histogram(aes(y=..density..), colour="black", fill="white",bins=50) + geom_density(alpha=.2, fill="#FF6666") + ylab("") + xlab("Density Plot with Mean") + geom_vline(aes(xintercept=mean(TEAM_BATTING_SO, na.rm=TRUE)), color="red", linetype="dashed", size=1) # Scatterplot ggplot(data=training, aes(x=TEAM_BATTING_SO, y=TARGET_WINS)) + geom_point() + geom_smooth(method = "loess") + xlab("Scatterplot with Best Fit Line")

TEAM_BASERUN_SB:

sumtable[sumtable[,1]==“TEAM_BASERUN_SB”,2:8] # Boxplot ggplot(training, aes(x = 1, y = TEAM_BASERUN_SB)) + stat_boxplot(geom =‘errorbar’) + geom_boxplot() + xlab(“Boxplot”) + ylab(“”) + theme(axis.text.x=element_blank(), axis.ticks.x=element_blank()) # Density plot ggplot(training, aes(x = TEAM_BASERUN_SB)) + geom_histogram(aes(y=..density..), colour=“black”, fill=“white”,bins=50) + geom_density(alpha=.2, fill=“#FF6666”) + ylab(“”) + xlab(“Density Plot with Mean”) + geom_vline(aes(xintercept=mean(TEAM_BASERUN_SB, na.rm=TRUE)), color=“red”, linetype=“dashed”, size=1) # Scatterplot ggplot(data=training, aes(x=TEAM_BASERUN_SB, y=TARGET_WINS)) + geom_point() + geom_smooth(method = “loess”) + xlab(“Scatterplot with Best Fit Line”)

TEAM_BASERUN_CS:

sumtable[sumtable[,1]==“TEAM_BASERUN_CS”,2:8] # Boxplot ggplot(training, aes(x = 1, y = TEAM_BASERUN_CS)) + stat_boxplot(geom =‘errorbar’) + geom_boxplot() + xlab(“Boxplot”) + ylab(“”) + theme(axis.text.x=element_blank(), axis.ticks.x=element_blank()) # Density plot ggplot(training, aes(x = TEAM_BASERUN_CS)) + geom_histogram(aes(y=..density..), colour=“black”, fill=“white”,bins=50) + geom_density(alpha=.2, fill=“#FF6666”) + ylab(“”) + xlab(“Density Plot with Mean”) + geom_vline(aes(xintercept=mean(TEAM_BASERUN_CS, na.rm=TRUE)), color=“red”, linetype=“dashed”, size=1) # Scatterplot ggplot(data=training, aes(x=TEAM_BASERUN_CS, y=TARGET_WINS)) + geom_point() + geom_smooth(method = “loess”) + xlab(“Scatterplot with Best Fit Line”)

TEAM_FIELDING_E:

sumtable[sumtable[,1]==“TEAM_FIELDING_E”,2:8] # Boxplot ggplot(training, aes(x = 1, y = TEAM_FIELDING_E)) + stat_boxplot(geom =‘errorbar’) + geom_boxplot() + xlab(“Boxplot”) + ylab(“”) + theme(axis.text.x=element_blank(), axis.ticks.x=element_blank()) # Density plot ggplot(training, aes(x = TEAM_FIELDING_E)) + geom_histogram(aes(y=..density..), colour=“black”, fill=“white”,bins=50) + geom_density(alpha=.2, fill=“#FF6666”) + ylab(“”) + xlab(“Density Plot with Mean”) + geom_vline(aes(xintercept=mean(TEAM_FIELDING_E, na.rm=TRUE)), color=“red”, linetype=“dashed”, size=1) # Scatterplot ggplot(data=training, aes(x=TEAM_FIELDING_E, y=TARGET_WINS)) + geom_point() + geom_smooth(method = “loess”) + xlab(“Scatterplot with Best Fit Line”)

TEAM_FIELDING_DP:

sumtable[sumtable[,1]==“TEAM_FIELDING_DP”,2:8] # Boxplot ggplot(training, aes(x = 1, y = TEAM_FIELDING_DP)) + stat_boxplot(geom =‘errorbar’) + geom_boxplot() + xlab(“Boxplot”) + ylab(“”) + theme(axis.text.x=element_blank(), axis.ticks.x=element_blank()) # Density plot ggplot(training, aes(x = TEAM_FIELDING_DP)) + geom_histogram(aes(y=..density..), colour=“black”, fill=“white”,bins=50) + geom_density(alpha=.2, fill=“#FF6666”) + ylab(“”) + xlab(“Density Plot with Mean”) + geom_vline(aes(xintercept=mean(TEAM_FIELDING_DP, na.rm=TRUE)), color=“red”, linetype=“dashed”, size=1) # Scatterplot ggplot(data=training, aes(x=TEAM_FIELDING_DP,

y=TARGET_WINS)) + geom_point() + geom_smooth(method = “loess”) + xlab(“Scatterplot with Best Fit Line”)

TEAM_PITCHING_BB:

sumtable[sumtable[,1]==“TEAM_PITCHING_BB”,2:8] # Boxplot ggplot(training, aes(x = 1, y = TEAM_PITCHING_BB)) + stat_boxplot(geom =‘errorbar’) + geom_boxplot() + xlab(“Boxplot”) + ylab(“”) + theme(axis.text.x=element_blank(), axis.ticks.x=element_blank()) # Density plot ggplot(training, aes(x = TEAM_PITCHING_BB)) + geom_histogram(aes(y=..density..), colour=“black”, fill=“white”,bins=50) + geom_density(alpha=.2, fill=“#FF6666”) + ylab(“”) + xlab(“Density Plot with Mean”) + geom_vline(aes(xintercept=mean(TEAM_PITCHING_BB, na.rm=TRUE)), color=“red”, linetype=“dashed”, size=1) # Scatterplot ggplot(data=training, aes(x=TEAM_PITCHING_BB, y=TARGET_WINS)) + geom_point() + geom_smooth(method = “loess”) + xlab(“Scatterplot with Best Fit Line”)

TEAM_PITCHING_H:

sumtable[sumtable[,1]==“TEAM_PITCHING_H”,2:8] # Boxplot ggplot(training, aes(x = 1, y = TEAM_PITCHING_H)) + stat_boxplot(geom =‘errorbar’) + geom_boxplot() + xlab(“Boxplot”) + ylab(“”) + theme(axis.text.x=element_blank(), axis.ticks.x=element_blank()) # Density plot ggplot(training, aes(x = TEAM_PITCHING_H)) + geom_histogram(aes(y=..density..), colour=“black”, fill=“white”,bins=50) + geom_density(alpha=.2, fill=“#FF6666”) + ylab(“”) + xlab(“Density Plot with Mean”) + geom_vline(aes(xintercept=mean(TEAM_PITCHING_H, na.rm=TRUE)), color=“red”, linetype=“dashed”, size=1) # Scatterplot ggplot(data=training, aes(x=TEAM_PITCHING_H, y=TARGET_WINS)) + geom_point() + geom_smooth(method = “loess”) + xlab(“Scatterplot with Best Fit Line”)

TEAM_PITCHING_SO:

sumtable[sumtable[,1]==“TEAM_PITCHING_SO”,2:8] # Boxplot ggplot(training, aes(x = 1, y = TEAM_PITCHING_SO)) + stat_boxplot(geom =‘errorbar’) + geom_boxplot() + xlab(“Boxplot”) + ylab(“”) + theme(axis.text.x=element_blank(), axis.ticks.x=element_blank()) # Density plot ggplot(training, aes(x = TEAM_PITCHING_SO)) + geom_histogram(aes(y=..density..), colour=“black”, fill=“white”,bins=50) + geom_density(alpha=.2, fill=“#FF6666”) + ylab(“”) + xlab(“Density Plot with Mean”) + geom_vline(aes(xintercept=mean(TEAM_PITCHING_SO, na.rm=TRUE)), color=“red”, linetype=“dashed”, size=1) # Scatterplot ggplot(data=training, aes(x=TEAM_PITCHING_SO, y=TARGET_WINS)) + geom_point() + geom_smooth(method = “loess”) + xlab(“Scatterplot with Best Fit Line”)

TARGET_WINS:

sumtable[sumtable[,1]==“TARGET_WINS”,2:8] # Boxplot ggplot(training, aes(x = 1, y = TARGET_WINS)) + stat_boxplot(geom =‘errorbar’) + geom_boxplot() + xlab(“Boxplot”) + ylab(“”) + theme(axis.text.x=element_blank(), axis.ticks.x=element_blank()) # Density plot ggplot(training, aes(x = TARGET_WINS)) + geom_histogram(aes(y=..density..), colour=“black”, fill=“white”,bins=50) + geom_density(alpha=.2, fill=“#FF6666”) + ylab(“”) +

```
xlab(“Density Plot with Mean”) + geom_vline(aes(xintercept=mean(TARGET_WINS, na.rm=TRUE)), color=“red”, linetype=“dashed”, size=1)

clean_data <- function(df){ # Change 0’s to NA so they too can be imputed df <- df %>% mutate(TEAM_BATTING_SO = ifelse(TEAM_BATTING_SO == 0, NA, TEAM_BATTING_SO)) # Remove the high pitching strikeout values df[which(df$TEAM_PITCHING_SO > 5346),“TEAM_PITCHING_SO”] <- NA # Drop the hit by pitcher variable df %>% select(-TEAM_BATTING_HBP) } training <- clean_data(training)

set.seed(42) knn <- training %>% knnImputation() apply_func <- function(df, knn){ impute_me <- is.na(df$TEAM_BATTING_SO)  df[impute_me,"TEAM_BATTING_SO"] <- knn[impute_me,"TEAM_BATTING_SO"]  impute_me <- is.na(df$TEAM_BASERUN_SB) df[impute_me,“TEAM_BASERUN_SB”] <- knn[impute_me,“TEAM_BASERUN_SB”] impute_me <- is.na(df$TEAM_BASERUN_CS)  df[impute_me,"TEAM_BASERUN_CS"] <- knn[impute_me,"TEAM_BASERUN_CS"]  impute_me <- is.na(df$TEAM_PITCHING_SO) df[impute_me,“TEAM_PITCHING_SO”] <- knn[impute_me,“TEAM_PITCHING_SO”] impute_me <- is.na(df$TEAM_FIELDING_DP) df[impute_me,“TEAM_FIELDING_DP”] <- knn[impute_me,“TEAM_FIELDING_DP”] return(df) } training <- apply_func(training, knn)

add_features <- function(df){ df %>% mutate(TEAM_BATTING_1B = TEAM_BATTING_H - TEAM_BATTING_2B - TEAM_BATTING_3B - TEAM_BATTING_HR) } training <- add_features(training)

Here’s what the data look like after imputation and correction:

training %>% gather(variable, value) %>% ggplot(., aes(value)) + geom_density(fill = “indianred4”, color=“indianred4”) + facet_wrap(~variable, scales =“free”, ncol = 4) + labs(x = element_blank(), y = element_blank())

quick_summary <- function(df){ df %>% summary() } quick_summary(training)

set.seed(42) train_index <- createDataPartition(training$TARGET_WINS, p = .7, list = FALSE, times = 1) train <- training[train_index,] test <- training[-train_index,]

m1 <- lm(TARGET_WINS ~ TEAM_BATTING_H + TEAM_BATTING_BB + TEAM_FIELDING_E, data=train) summary(m1)

m2 <- lm(TARGET_WINS ~ TEAM_BATTING_1B + TEAM_BATTING_2B + TEAM_BATTING_3B + TEAM_BATTING_HR + TEAM_BATTING_BB + TEAM_FIELDING_E, data=train) summary(m2)

A full_formula <- “TARGET_WINS ~ TEAM_BATTING_2B + TEAM_BATTING_3B + TEAM_BATTING_HR + TEAM_BATTING_BB + TEAM_BATTING_SO + TEAM_BASERUN_SB + TEAM_BASERUN_CS + TEAM_PITCHING_H + TEAM_PITCHING_HR + TEAM_PITCHING_BB + TEAM_PITCHING_SO + TEAM_FIELDING_E + TEAM_FIELDING_DP + TEAM_BATTING_1B + I(TEAM_BATTING_2B^2) + I(TEAM_BATTING_3B^2) + I(TEAM_BATTING_HR^2) + I(TEAM_BATTING_BB^2) + I(TEAM_BATTING_SO^2) + I(TEAM_BASERUN_SB^2) + I(TEAM_BASERUN_CS^2) + I(TEAM_PITCHING_H^2) + I(TEAM_PITCHING_HR^2) + I(TEAM_PITCHING_BB^2) + I(TEAM_PITCHING_SO^2) + I(TEAM_FIELDING_E^2) +
```

I(TEAM_FIELDING_DP^2) + I(TEAM_BATTING_1B^2)" full_model <- lm(full_formula, train) step_back <- MASS::stepAIC(full_model, direction="backward", trace = F) poly_call <- summary(step_back)$call m3 <- lm(poly_call[2], train) summary(m3)

Create log fielding error

train$TEAM_F IELDING_{E_L} OG < -log(train$TEAM_FIELDING_E) m4 <- lm(TARGET_WINS ~ TEAM_BATTING_SO + TEAM_BATTING_3B + TEAM_BATTING_HR + TEAM_BASERUN_SB + TEAM_FIELDING_E_LOG*TEAM_PITCHING_H, data=train) summary(m4)

test$m1 < -predict(m1, test)test < -test$m1_error^2)),2)), color="white") summary(test$m1_error)

test$m2 < -predict(m2, test)test < -test$m2_error^2)),2)), color="white") summary(test$m2_error)

test$m3 < -predict(m3, test)test < -test$m3_error^2)),2)), color="white") summary(test$m3_error)

Create log fielding error

test$TEAM_F IELDING_{E_L} OG < -log(test$TEAM_FIELDING_E) test$m4 < -predict(m4, test)test < -test$m4_error^2)),2)), color="white") summary(test$m4_error)

AIC(m1, m2, m3, m4) summary(m3)

plot(m3$residuals, ylab="Residuals") abline(h=0)

clean_data <- function(df){ # Change 0's to NA so they too can be imputed df <- df %>% mutate(TEAM_BATTING_SO = ifelse(TEAM_BATTING_SO == 0, NA, TEAM_BATTING_SO)) # Remove the high pitching strikeout values df[which(df$TEAM_PITCHING_SO > 5346),"TEAM_PITCHING_SO"] <- NA # Drop the hit by pitcher variable df %>% select(-TEAM_BATTING_HBP) } evaluation <- clean_data(evaluation)

set.seed(42) knn <- evaluation %>% knnImputation() apply_func <- function(df, knn){ impute_me <- is.na(df$TEAM_BATTING_SO) df[impute_me,"TEAM_BATTING_SO"] <- knn[impute_me,"TEAM_BATTING_SO"] impute_me <- is.na(df$TEAM_BASERUN_SB) df[impute_me,"TEAM_BASERUN_SB"] <- knn[impute_me,"TEAM_BASERUN_SB"] impute_me <- is.na(df$TEAM_BASERUN_CS) df[impute_me,"TEAM_BASERUN_CS"] <- knn[impute_me,"TEAM_BASERUN_CS"] impute_me <- is.na(df$TEAM_PITCHING_SO) df[impute_me,"TEAM_PITCHING_SO"] <- knn[impute_me,"TEAM_PITCHING_SO"] impute_me <- is.na(df$TEAM_FIELDING_DP) df[impute_me,"TEAM_FIELDING_DP"] <- knn[impute_me,"TEAM_FIELDING_DP"] return(df) } evaluation <- apply_func(evaluation, knn)

```
add_features <- function(df){ df %>% mutate(TEAM_BATTING_1B = TEAM_BATTING_H - TEAM_BATTING_2B - TEAM_BATTING_3B - TEAM_BATTING_HR) } evaluation <- add_features(evaluation)

evaluation$PREDICT_WIN <- predict(m3, newdata=evaluation, interval="confidence")
Forecast <- cbind(evaluation$INDEX,
evaluation$PREDICT_WIN[,1], evaluation$PREDICT_WIN[, 2], evaluation$PREDICT_WIN[, 3])
colnames(Forecast) <- c("Index", "Predicted Wins", "CI Lower", "CI Upper")
round(Forecast,0)
```