Data 621 Project 01
Section 01 Group 5

Farhana Zahir, Habib Khan, Vijaya Cherukuri, Scott Reed, Shovon Biswas

9/22/2020

Contents

Executive Summary

The team has developed a model using historical baseball data to determine a team's performance based on statistics of their performance. While correlation does not equal causation it is suggested that a focus on some of the variables such as a focus on either single hits or triple or more hits to the exclusion of doubles might be worth pursuing. Also the data suggests that a focus on home runs allowed may not be worth giving up a number of more normal hits.

Data Used

In this project, we are asked to explore, analyze and model a baseball dataset. Each record represents a particular professional baseball team from the years 1871 to 2006 inclusive. Each record has the performance of the given team for the given year, with all of the statistics adjusted as if the team were playing a 162 game season.

Our objective is to build a multiple linear regression model on the training data to predict the number of wins for the team, based on their other performance variables. This data set is split between a training dataset of 2276 observation for model development and a test dataset of 259 observations for model validation.

A short description of the variables and their theoretical effect on the response variable 'Number of wins' is provided.

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

Data exploration

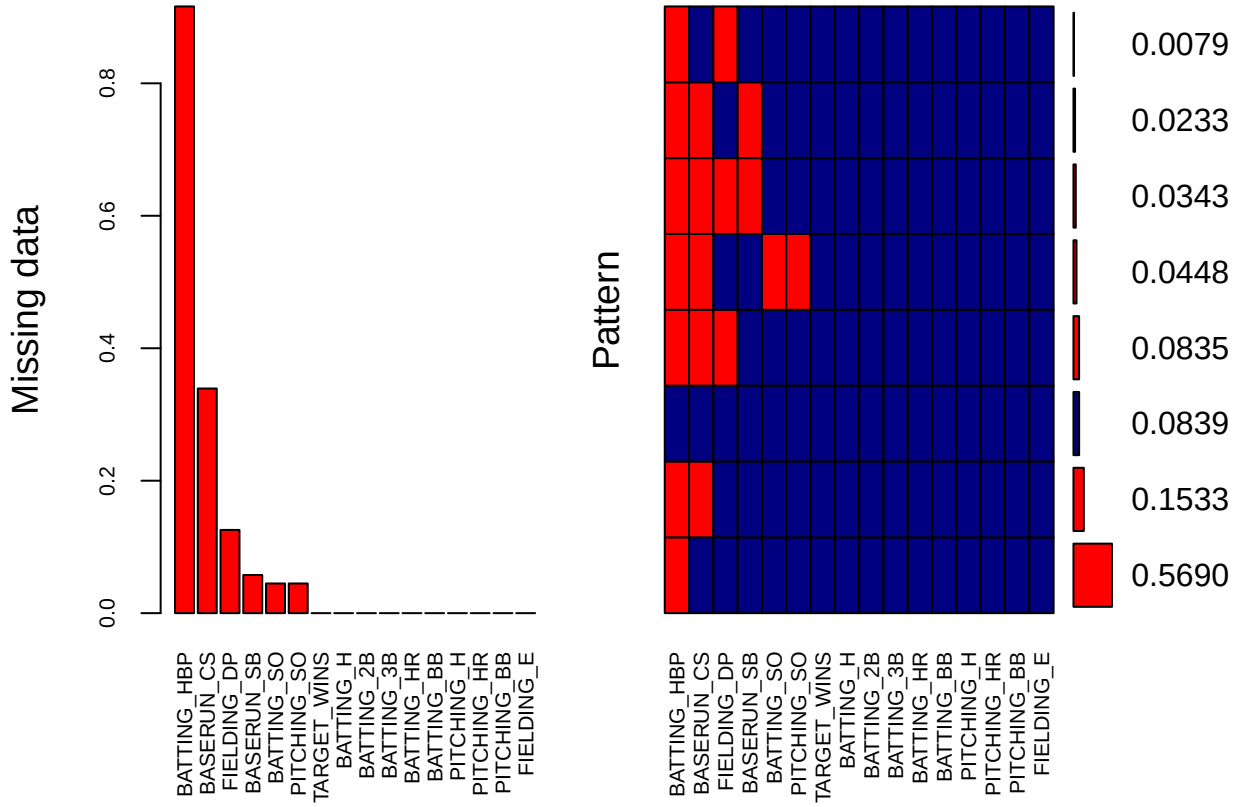
Looking at the data it is clear that all of the data is numeric, and that Hit By Pitch has a lot of missing variables.

```
##   TARGET_WINS       BATTING_H      BATTING_2B      BATTING_3B
##  Min.   :  0.00   Min.   : 891   Min.   : 69.0   Min.   :  0.00
##  1st Qu.: 71.00   1st Qu.:1383   1st Qu.:208.0   1st Qu.: 34.00
##  Median : 82.00   Median :1454   Median :238.0   Median : 47.00
##  Mean   : 80.79   Mean   :1469   Mean   :241.2   Mean   : 55.25
##  3rd Qu.: 92.00   3rd Qu.:1537   3rd Qu.:273.0   3rd Qu.: 72.00
##  Max.   :146.00   Max.   :2554   Max.   :458.0   Max.   :223.00
##
##    BATTING_HR       BATTING_BB      BATTING_SO       BASERUN_SB
##  Min.   :  0.00   Min.   :  0.0   Min.   :   0.0   Min.   :  0.0
##  1st Qu.: 42.00   1st Qu.:451.0   1st Qu.: 548.0   1st Qu.: 66.0
##  Median :102.00   Median :512.0   Median : 750.0   Median :101.0
##  Mean   : 99.61   Mean   :501.6   Mean   : 735.6   Mean   :124.8
##  3rd Qu.:147.00   3rd Qu.:580.0   3rd Qu.: 930.0   3rd Qu.:156.0
##  Max.   :264.00   Max.   :878.0   Max.   :1399.0   Max.   :697.0
##                                   NA's   :102      NA's   :131
##    BASERUN_CS     BATTING_HBP      PITCHING_H     PITCHING_HR
##  Min.   :  0.0   Min.   :29.00   Min.   : 1137   Min.   :  0.0
##  1st Qu.: 38.0   1st Qu.:50.50   1st Qu.: 1419   1st Qu.: 50.0
##  Median : 49.0   Median :58.00   Median : 1518   Median :107.0
##  Mean   : 52.8   Mean   :59.36   Mean   : 1779   Mean   :105.7
##  3rd Qu.: 62.0   3rd Qu.:67.00   3rd Qu.: 1682   3rd Qu.:150.0
##  Max.   :201.0   Max.   :95.00   Max.   :30132   Max.   :343.0
##  NA's   :772     NA's   :2085
##   PITCHING_BB      PITCHING_SO        FIELDING_E      FIELDING_DP
##  Min.   :   0.0   Min.   :    0.0   Min.   :  65.0   Min.   : 52.0
##  1st Qu.: 476.0   1st Qu.:  615.0   1st Qu.: 127.0   1st Qu.:131.0
##  Median : 536.5   Median :  813.5   Median : 159.0   Median :149.0
##  Mean   : 553.0   Mean   :  817.7   Mean   : 246.5   Mean   :146.4
##  3rd Qu.: 611.0   3rd Qu.:  968.0   3rd Qu.: 249.2   3rd Qu.:164.0
##  Max.   :3645.0   Max.   :19278.0   Max.   :1898.0   Max.   :228.0
##                   NA's   :102                        NA's   :286
```

6 of the variables have missing values:

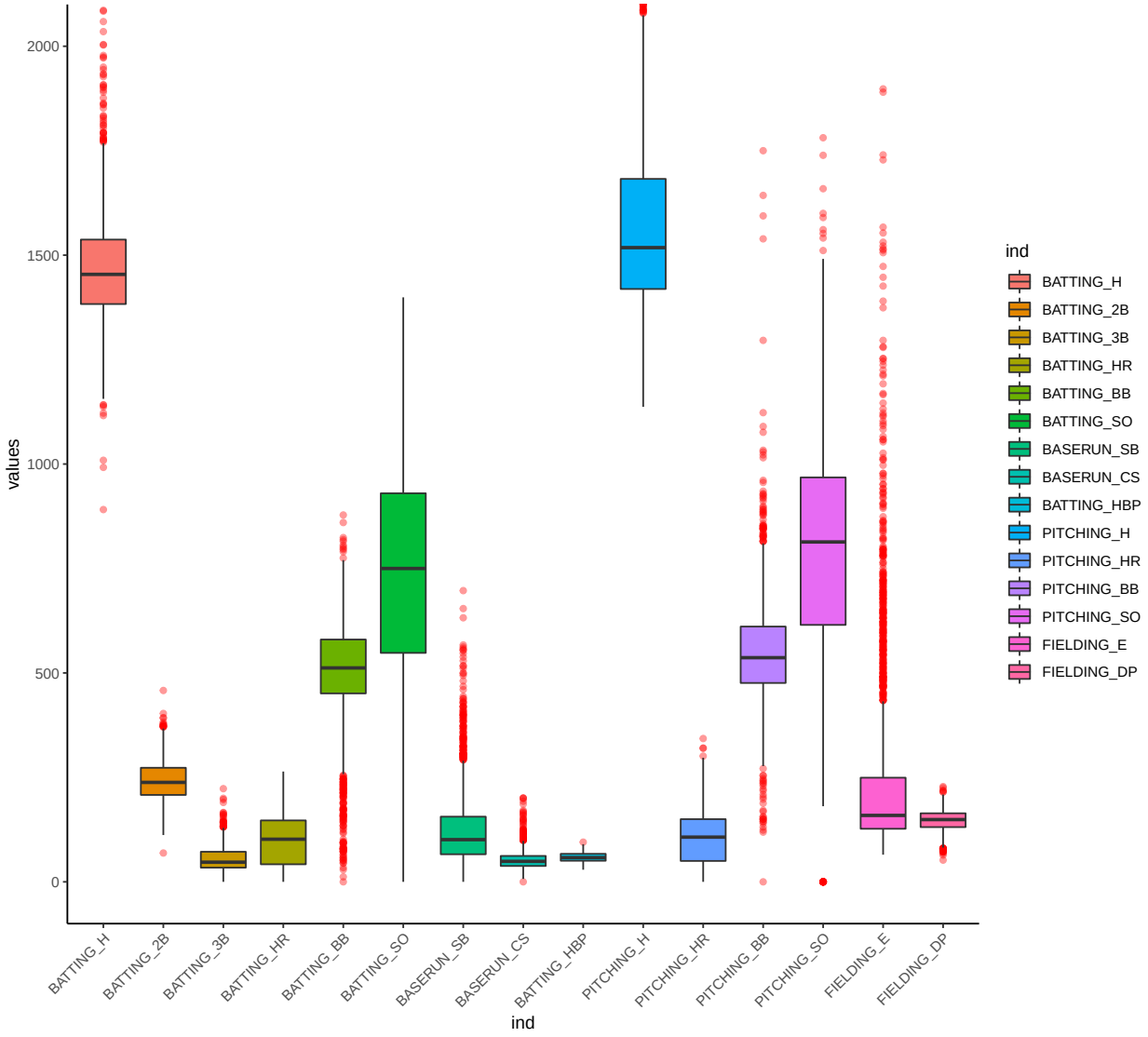
- BATTING_SO: 102 (4.5% of total)
- PITCHING_SO: 102 (4.5% of total)
- BASERUN_SB: 131 (5.8% of total)
- BASERUN_CS: 772 (34% of total, needs to be looked at closely, impute or exclude?)
- BATTING_HBP: 2085 (92% of total, exclude from analysis?)
- FIELDING_DP: 286 (12.6% of total,needs to be looked at closely, impute or exclude?)

The following diagram shows the missingness pattern by variables.


Missing data
0.0
0.2
0.4
0.6
0.8
BATTING_HBP
BASERUN_CS
FIELDING_DP
BASERUN_SB
BATTING_SO
PITCHING_SO
TARGET_WINS
BATTING_H
BATTING_2B
BATTING_3B
BATTING_HR
BATTING_BB
PITCHING_H
PITCHING_HR
PITCHING_BB
FIELDING_E
Pattern
0.0079
0.0233
0.0343
0.0448
0.0835
0.0839
0.1533
0.5690

Outliers

The following diagram shows the outliers for all the variables, both dependent and independent.

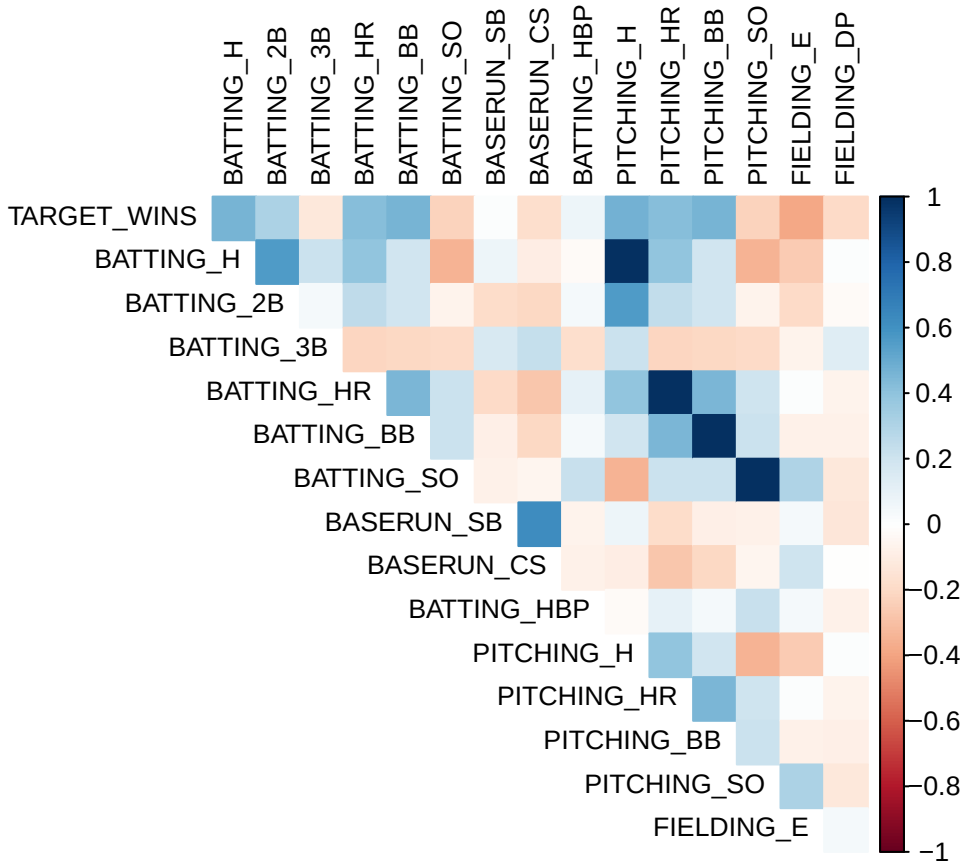


We can see that 12 of the variables have a significant skew. only 4 of the 16 variables are normally or close to normally distributed. The response variable Target_wins seems to be normally distributed. Batting_Hr, Batting_SO and Pitching_HR are bi-modal.

10 of the 16 variables have a minimum value of 0. This is not a major concern as the total % of 0 in each column is less than 1%.

The variables Batting_BB, Batting_CS, Baserun_SB, Pitching_BB and Fielding_E have a significant number of outliers.

Correlations among predictors



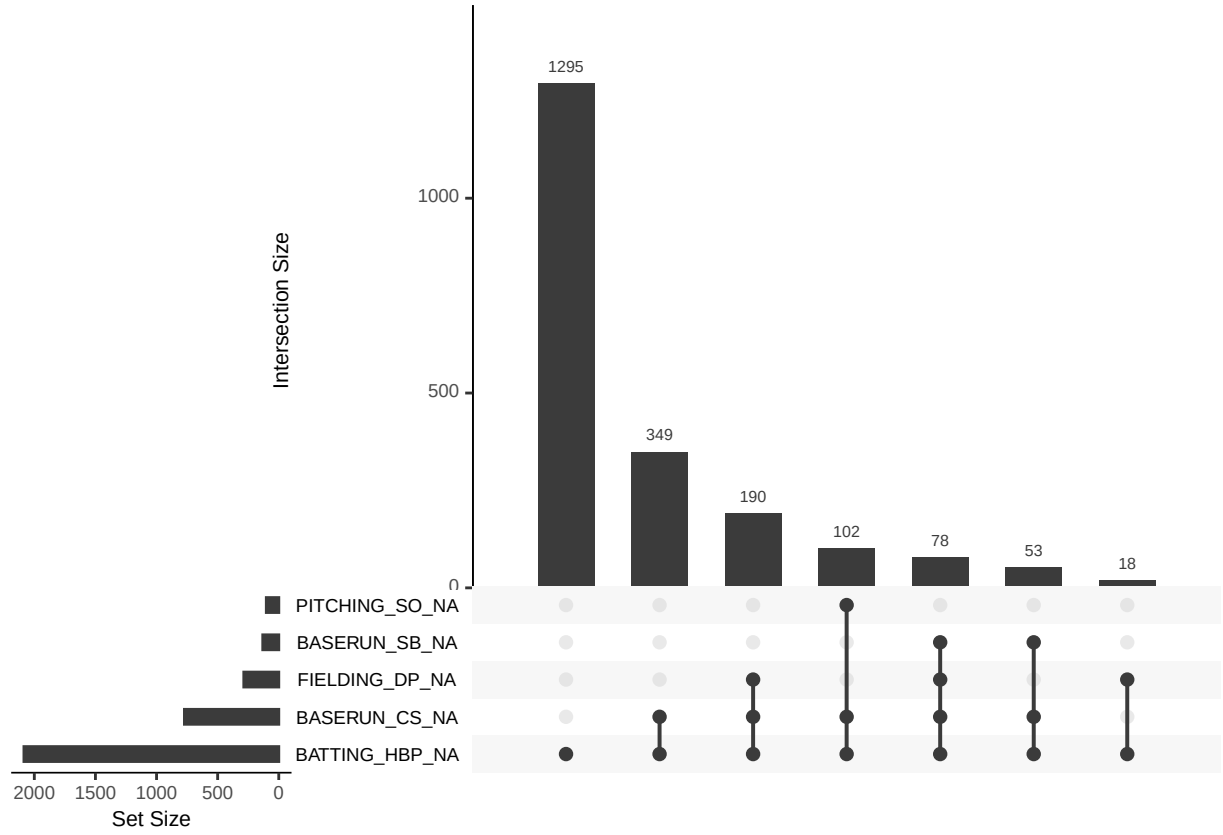
There are positive or negative correlations among the predictors. A small number of values are not correlated. Let us look at the numerical correlations with the response variable. We see that the predictors Batting_H, Batting_HR, Batting_BB, Pitching_H, and Pitching_HR are more correlated and should be included in our regression.

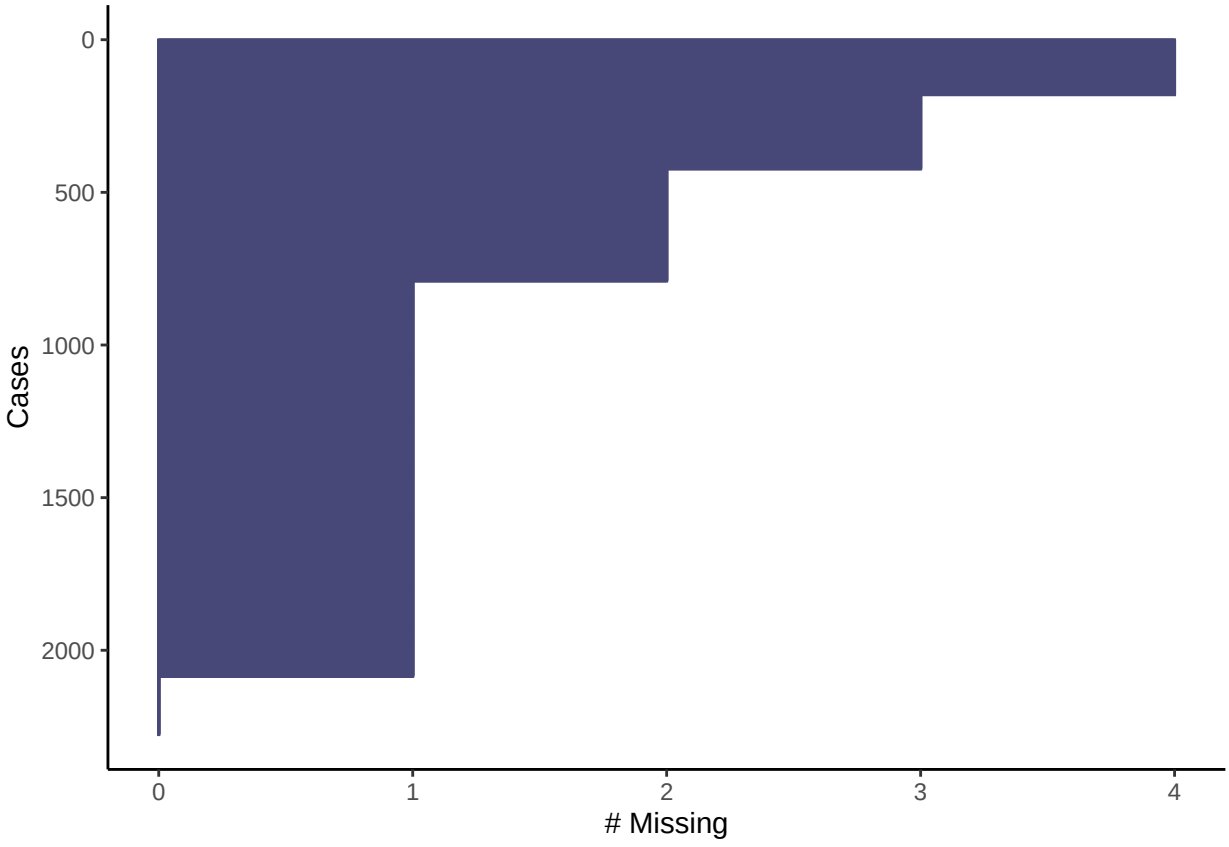
rowname	Variable	Correlation
TARGET_WINS	TARGET_WINS	1.0000000
PITCHING_H	TARGET_WINS	0.4712343
BATTING_H	TARGET_WINS	0.4699467
BATTING_BB	TARGET_WINS	0.4686879
PITCHING_BB	TARGET_WINS	0.4683988
PITCHING_HR	TARGET_WINS	0.4224668
BATTING_HR	TARGET_WINS	0.4224168
BATTING_2B	TARGET_WINS	0.3129840
BATTING_HBP	TARGET_WINS	0.0735042
BASERUN_SB	TARGET_WINS	0.0148364
BATTING_3B	TARGET_WINS	-0.1243459
BASERUN_CS	TARGET_WINS	-0.1787560
FIELDING_DP	TARGET_WINS	-0.1958660
BATTING_SO	TARGET_WINS	-0.2288927
PITCHING_SO	TARGET_WINS	-0.2293648
FIELDING_E	TARGET_WINS	-0.3866880

rowname	Variable	Correlation
BATTING_2B	BATTING_H	0.5617729
PITCHING_H	BATTING_H	0.9991927
PITCHING_H	BATTING_2B	0.5604535
PITCHING_HR	BATTING_HR	0.9999326
PITCHING_BB	BATTING_BB	0.9998814
PITCHING_SO	BATTING_SO	0.9997684
BASERUN_CS	BASERUN_SB	0.6247378

Examining significant correlations among the independent variables, we see that four of the pairs have a correlation close to 1. This can lead to multicollinearity issues in our analysis.

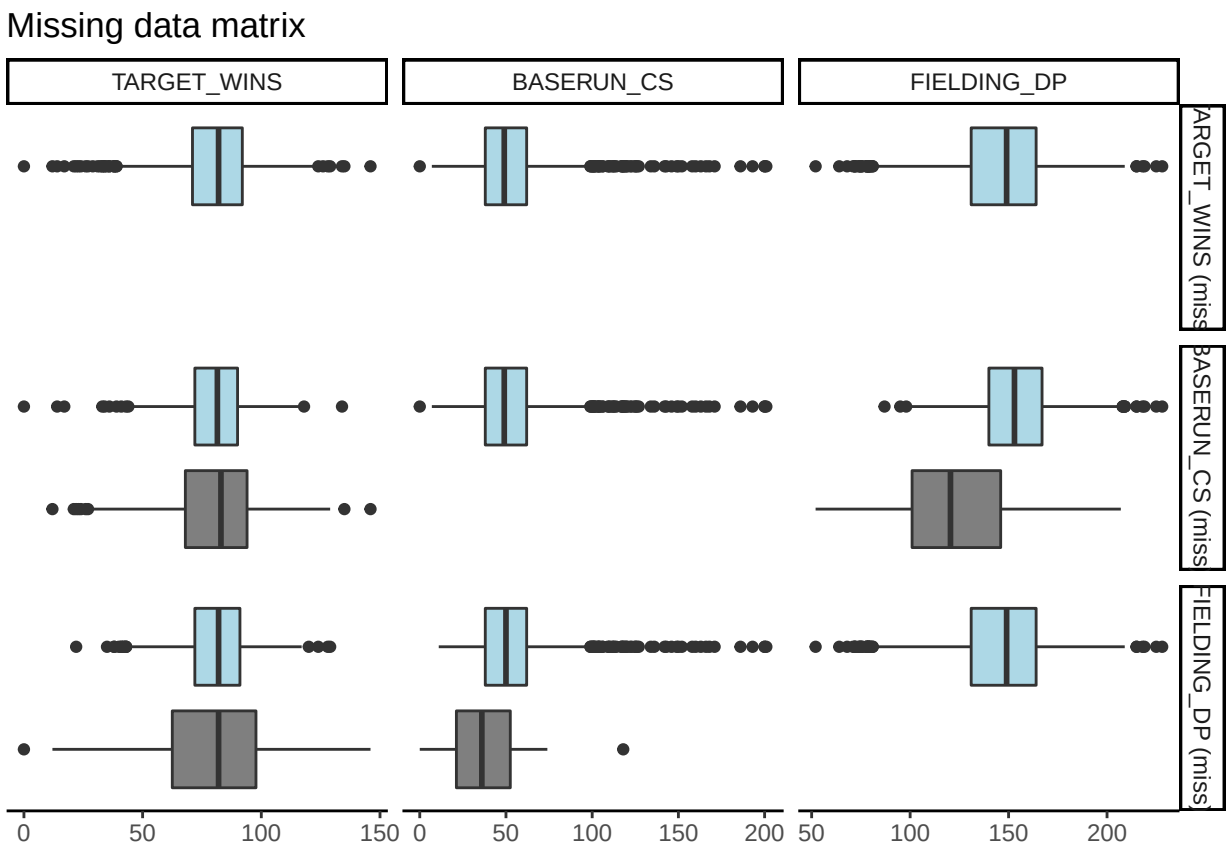
Data Preparation





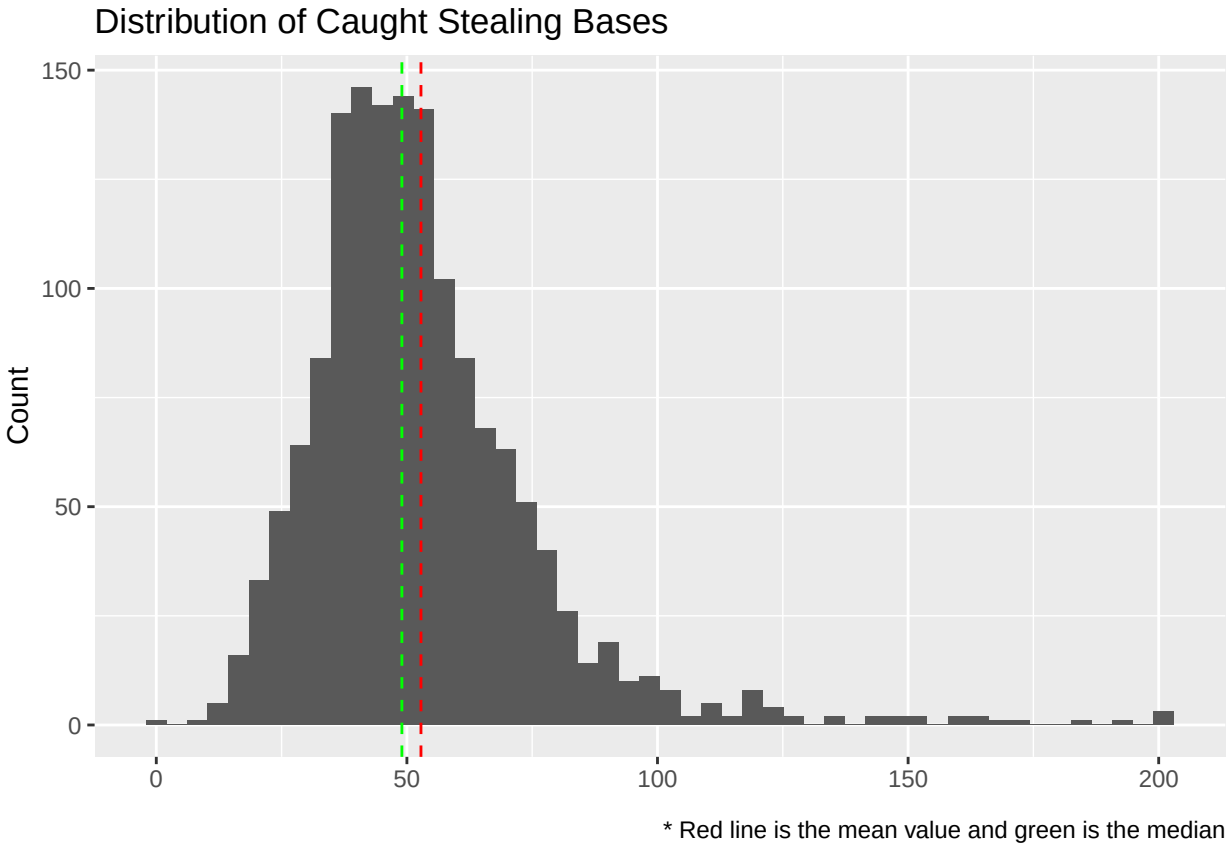
We will look at the patterns and intersections of missing data among the variables. We can see that only 22 of the observations have all 5 variables missing, so we are completely removing these observations. Overall, the pattern suggests that the variables are Missing at Random (MAR)

In deciding whether data is MCAR (Missing Completely At Random) or MAR, one approach is to explore patterns of missingness between variables. This is particularly important for a primary outcome measure/dependent variable. We will test the two cases with most missing values.



The wins for teams whose Caught Stealing (Baserun_cs) data is known is the blue box plot, and the wins with missing Baserun_cs data is the gray box plot. We do not see a lot of difference in the mean, but the Interquartile range has changed. Also the mean changes significantly between Double Play (Fielding_DP) and Baserun_cs if we do not include missing data.

102 of the observations have 3 of the variables missing.The highest case missing is in the BASERUN_CS variable, so this deserves an analysis of its own.We find a highly positively skewed distribution with most missing data after the 125 on the x axis. We want to make sure that the shape of our distribution does not change as we impute the data.


Distribution of Caught Stealing Bases
Count
150
100
50
0
0
50
100
150
200
* Red line is the mean value and green is the median

Data Preparation for Modeling

Model 1 - Basic Model

This model is created through removing basic extreme outliers from Hits Allowed and Strikeouts by Pitchers Also, the data for hit by pitcher had 92% missing values requiring its removal.

In this model, we replace the missing values for each column with their median.

All the missing values were replaced by the median of it's column. Most of the variables don't have extreme outliers but still it has some sort of skewness but we won't remove them because it might change the model's fitness. We will fix the model in next steps and check which model performs better. Variables like Hits Allowed and Walks by Batters are skewed significantly which we have to transform in next models.

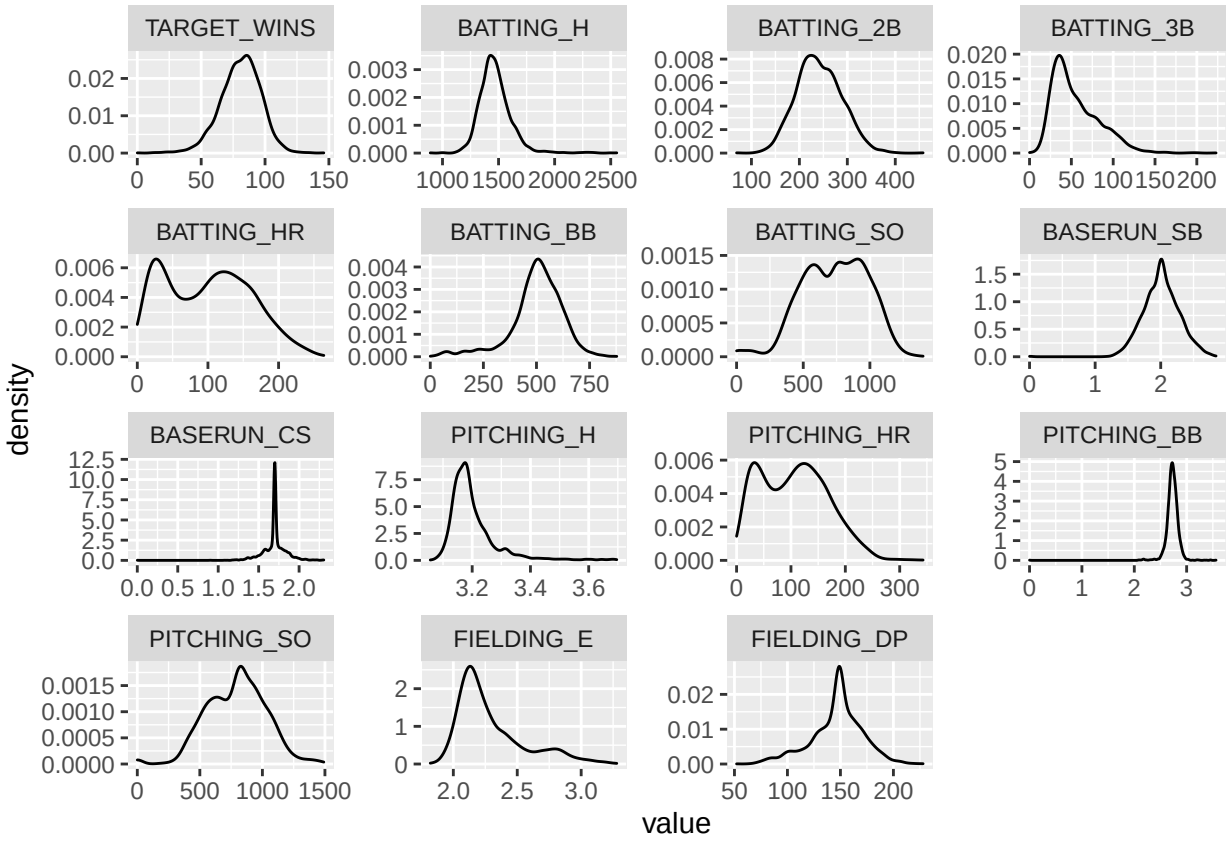
Model 2 - Log Transformation

In this model, we use log transformation to transform skewed distributions into normally distributed shape. Skewness improved after applying log transformation on highly skewed variables. We must apply log transformation to both training and test datasets to maintain consistency. We used log10 and added 1 on each skewed variable to avoid log having an undefined result for our many 0 values. We applied this transformation on the following variables:

- Walks By Batters
- Hits Allowed by Pitcher
- Fielding Errors
- Stolen Bases
- Caught Stealing

Skewness is comparatively much better now as it was in previous model.

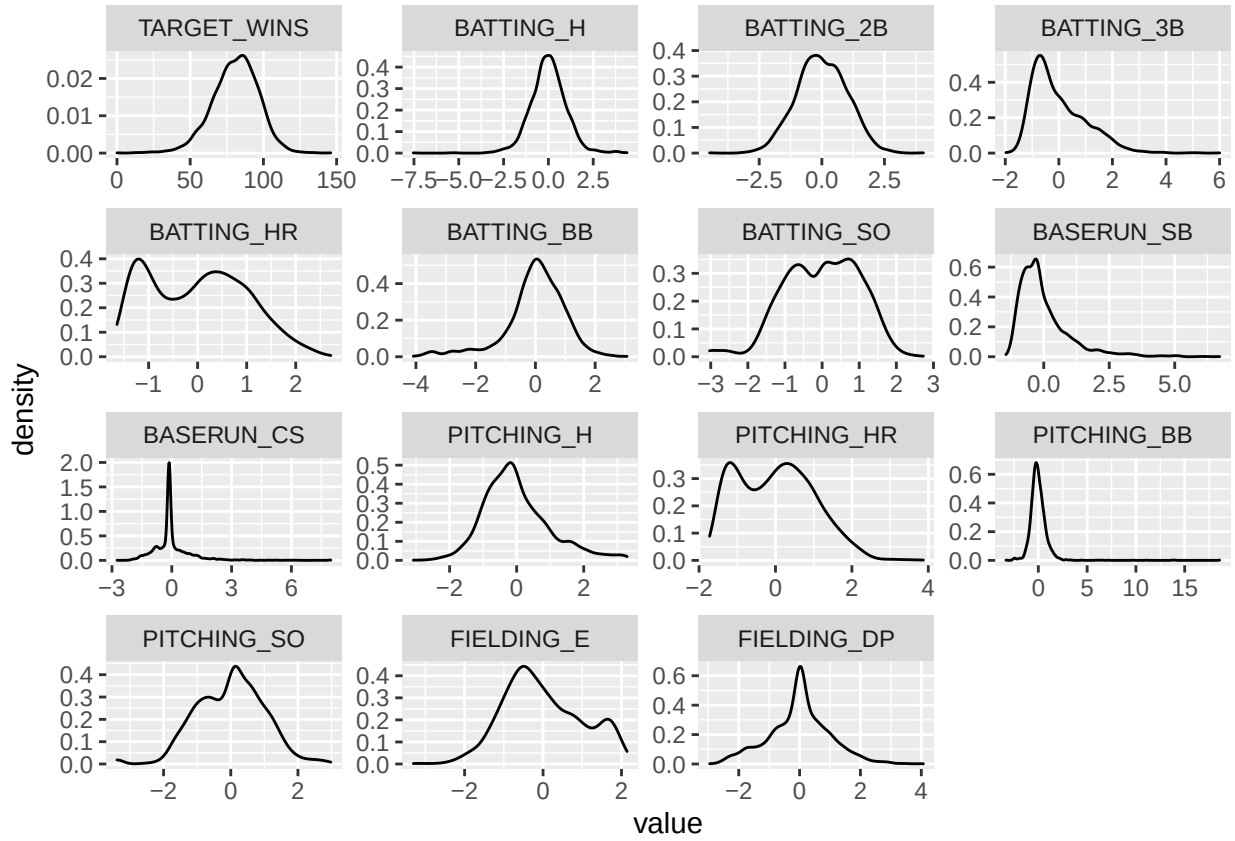
```
## TARGET_WINS   BATTING_H  BATTING_2B  BATTING_3B  BATTING_HR  BATTING_BB
##  -0.3987232   1.5713335   0.2151018   1.1094652   0.1860421  -1.0257599
##  BATTING_SO  BASERUN_SB  BASERUN_CS  PITCHING_H PITCHING_HR PITCHING_BB
##  -0.3126012  -0.2565572  -0.7870606   2.5388764   0.2877877  -5.0646903
## PITCHING_SO  FIELDING_E FIELDING_DP
##  -0.1600733   1.2547515  -0.4551750
```



Model 3 - BoxCox Transformation

In this model, we are going to apply a BoxCox transformation but also center and scale. This means that we want to subtract the mean of predictor's data from predictor's values while scale divides by the standard deviation. We were hoping this model to perform better but the resultant data skewness was not remediated as well as it was log transformation.

```
## [1] 259  14
```




```
## TARGET_WINS   BATTING_H  BATTING_2B  BATTING_3B  BATTING_HR  BATTING_BB
## -0.39872320 -0.08517001 -0.01738229  1.10946519  0.18604214 -1.02575989
##  BATTING_SO  BASERUN_SB  BASERUN_CS  PITCHING_H PITCHING_HR PITCHING_BB
## -0.31260120  2.06582822  2.60217224  0.77405438  0.28778767  6.74389947
## PITCHING_SO  FIELDING_E FIELDING_DP
## -0.16007329  0.21740959  0.05371277
```

Model 4 - kNN Imputation on missing values

In model 4, we are going to replace the missing values with kNN using the Caret R package. We decided to remove Hit By Pitcher and Caught Stealing as they had 92% and 33% missing values respectively. We want to see if kNN Imputation along with other selected variables might give better results. After imputation, we used log transformation with the kNN imputed values to see if it will improve the model's fitness. Again for consistency, we have to apply all transformation on both training and datasets

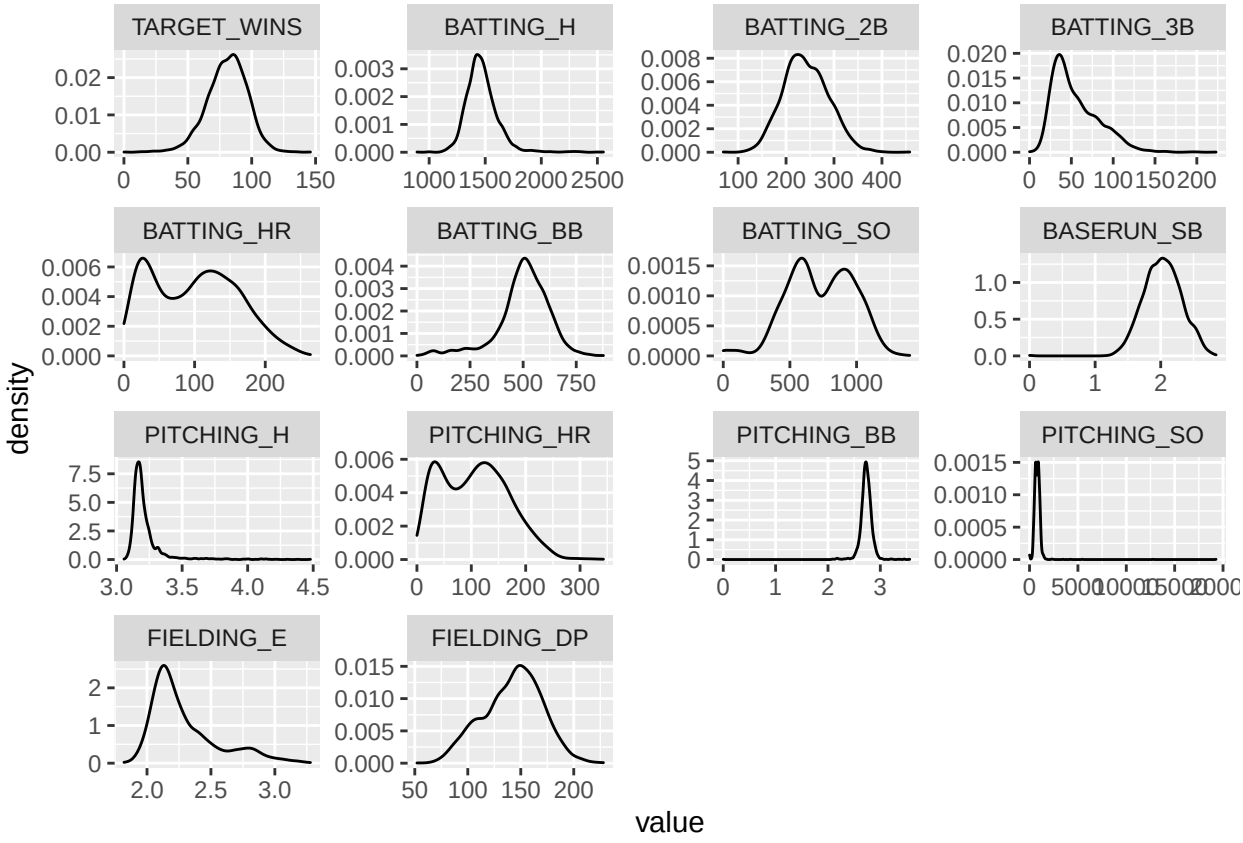
kNN Imputation

As it may be unfamiliar to some, kNN is a method of building a model to impute data. While in theory a more sophisticated model could be used, kNN has some advantages in being simple and well understood.

```
##     TARGET_WINS       BATTING_H      BATTING_2B      BATTING_3B      BATTING_HR
##               0               0               0               0               0
##      BATTING_BB      BATTING_SO      BASERUN_SB      PITCHING_H     PITCHING_HR
##               0               0               0               0               0
##     PITCHING_BB     PITCHING_SO      FIELDING_E     FIELDING_DP  BATTING_SO_imp
##               0               0               0               0               0
```

```
##  BASERUN_SB_imp PITCHING_SO_imp FIELDING_DP_imp
##               0               0               0
```

```
##     TARGET_WINS       BATTING_H      BATTING_2B      BATTING_3B      BATTING_HR
##      -0.3987232       1.5713335       0.2151018       1.1094652       0.1860421
##      BATTING_BB      BATTING_SO      BASERUN_SB      PITCHING_H     PITCHING_HR
##      -1.0257599      -0.2319776      -0.2442570       4.2997637       0.2877877
##     PITCHING_BB     PITCHING_SO      FIELDING_E     FIELDING_DP  BATTING_SO_imp
##      -5.0646903      22.5814355       1.2547515      -0.2142498       4.3971736
##  BASERUN_SB_imp PITCHING_SO_imp FIELDING_DP_imp
##       3.7968539       4.3971736       2.2572193
```



Build Models

Model 0 . . . a Sanity check

Using the imputed data we can simply run a linear regression against all the variables.

First of all the significance of the model is significant as per the p-value of F-statistics. It means that overall the model is significant. Adjusted r2 is 0.32 which means these variables can explain 32% of TARGET_WINS. All the variables other than BASERUN_CS and PITCHING_SO are insignificant. BASERUN_CS has 33% missing values which should have negative impact and it has negative but insignificant. It means Caught Stealing and it does not impact the chances of winning significantly. PITCHING_SO means strikeouts by pitchers which also has insigificant impact on TARGET_WINS. BATTING_2B should have positive impact but in model it shows significant but with very little coefficient. BATTING_HR means homeruns by batters should've and have positive impact on chances of winning. BATTING_BB means walks by batter. It should have positive impact but in our model it has insignificant impact. BATTING_SO means strikeouts by batters which shoul've negative impact but in our model it does not play significant role. BASERUN_SB should've significant impact and it has positive role. BASERUN_CS, PITCHING_HR and PITCHING_SO have insignificant impact while PITCHING_H, PITCHING_BB, FIELDING_E and FIELDING_DP have significant roles.

$$TARGET_WINS = 19.02 + BATTING_H(0.04) - BATTING_2B(0.02) + BATTING_3B(0.07) +$$
$$BATTING_HR(0.09) + BASERUN_SB(0.03) + PITCHING_H(0.01) + PITCHING_BB(0.01)$$
$$-FIELDING_E(0.03) - FIELDING_DP(0.11)$$

Table 1:

	Dependent variable:
	TARGET_WINS
BATTING_H	0.044***
	(0.004)
BATTING_2B	−0.024***
	(0.009)
BATTING_3B	0.072***
	(0.016)
BATTING_HR	0.091***
	(0.029)
BATTING_BB	0.006
	(0.004)
BATTING_SO	−0.003
	(0.004)
BASERUN_SB	0.027***
	(0.004)
BASERUN_CS	−0.012
	(0.016)
PITCHING_H	0.006***
	(0.001)
PITCHING_HR	−0.029
	(0.026)
PITCHING_BB	0.007***
	(0.002)
PITCHING_SO	−0.0005
	(0.003)
FIELDING_E	−0.027***
	(0.002)
FIELDING_DP	−0.111***
	(0.013)
Constant	19.022***
	(5.417)
Observations	2,276
R^2	0.324
Adjusted R^2	0.319
Residual Std. Error	12.995 (df = 2261)
F Statistic	77.286*** (df = 14; 2261)
Note:	*p<0.1; **p<0.05; ***p<0.01

Model 1 Basic Model

We select variable that seem correlated for this simple direct model.

Namely:

- Base Hits by Batters Batting_H
- Doubles by Batters Batting_2H
- Triples By Batters Batting_3B
- Home Runes Batting_HR
- Stolen Bases Baserun_SB
- Hits Allowed Pitching_H
- Walks Allowed Pitching_BB
- Errors in Fielding Fielding_E
- Double Plays Fielding_DP

All have a fairly direct impact on scoring or runs allowed.

This model's Adjusted R2 is also .32 which suggests our model is performing perfectly well without the other variables. This makes the model a bit less pointlessly confusing.

Table 2:

	Dependent variable:
	TARGET_WINS
BATTING_H	0.047***
	(0.003)
BATTING_2B	−0.028***
	(0.009)
BATTING_3B	0.076***
	(0.016)
BATTING_HR	0.055***
	(0.007)
BASERUN_SB	0.027***
	(0.004)
PITCHING_H	0.005***
	(0.001)
PITCHING_BB	0.008***
	(0.002)
FIELDING_E	−0.029***
	(0.002)
FIELDING_DP	−0.104***
	(0.013)
Constant	15.505***
	(3.311)
Observations	2,276
R^2	0.321
Adjusted R^2	0.318
Residual Std. Error	13.010 (df = 2266)
F Statistic	118.774*** (df = 9; 2266)
Note:	*p<0.1; **p<0.05; ***p<0.01

Model 2 Log Transformation

For this model we use the log adjusted values adding in the adjusted Caught Stealing. In addition to the previously present, but now adjusted:

- Walks By Batters
- Hits Allowed by Pitcher
- Fielding Errors
- Stolen Bases

Adjusted-r2 did not improve overall as compared with model 1 and intercept is no longer significant in this model. The additional Caught Stealing variable is significant, but with the loss of a significant intercept it is a wash. This does not seem a preferred model because log transformations are complex to intuit, and it doesn't seem to offer a meaningful benefit.

Table 3:

	Dependent variable:
	TARGET_WINS
BATTING_H	0.045***
	(0.003)
BATTING_2B	−0.037***
	(0.009)
BATTING_3B	0.120***
	(0.016)
BATTING_HR	0.031***
	(0.008)
BASERUN_SB	10.123***
	(1.307)
PITCHING_H	28.610***
	(4.576)
PITCHING_BB	9.274***
	(2.463)
FIELDING_E	−31.029***
	(2.058)
FIELDING_DP	−0.104***
	(0.013)
BASERUN_CS	−6.792***
	(2.265)
Constant	−25.409*
	(14.552)
Observations	2,276
R^2	0.319
Adjusted R^2	0.316
Residual Std. Error	13.023 (df = 2265)
F Statistic	106.328*** (df = 10; 2265)
Note:	*p<0.1; **p<0.05; ***p<0.01

Model 3 BoxCox Transformation

For the BoxCox transformed data set we will continue with the same variables as previously.

Table 4:

	Dependent variable:
	TARGET_WINS
BATTING_H	6.230***
	(0.488)
BATTING_2B	−1.276***
	(0.432)
BATTING_3B	3.744***
	(0.481)
BATTING_HR	2.005***
	(0.525)
BASERUN_SB	2.118***
	(0.343)
PITCHING_H	0.469
	(0.422)
PITCHING_BB	1.170***
	(0.297)
FIELDING_E	−6.594***
	(0.571)
FIELDING_DP	−2.709***
	(0.314)
BASERUN_CS	0.022
	(0.297)
Constant	80.791***
	(0.279)
Observations	2,276
R^2	0.291
Adjusted R^2	0.287
Residual Std. Error	13.297 (df = 2265)
F Statistic	92.752*** (df = 10; 2265)
Note:	*p<0.1; **p<0.05; ***p<0.01

With boxcox transformation, intercept is now again significant but adjusted r-square dropped to .29. We have lost two variables to insignificance with Caught Stealing becoming particular meaningless.

Model 4 kNN Imputation

With our log adjusted KNN we will take all 13 variables.

It appears that knn-imputation along with log transformation improved the adjusted r-square from 0.33 to 0.39 from model 3. As discussed merely depending upon adjusted r-square is not advised. According to new model BATTING_HR is insignificant all the other variables are significant at 10 % confidence interval which means result did improve here. BATTING_2B is negative though but consistent with all models but to very low extent. BATTING_SO, PITCHING_H, PITCHING_BB and PITCHING_E have also negative impact which comports with the theoretical impact.

Our model will be:

$$TARGET_WINS = 154.048 + BATTING_H(0.05) - BATTING_2B(0.04) + BATTING_3B(0.10) +$$

$$BATTING_BB(0.04) - BATTING_SO(0.02) + BASERUN_SB(12.89) - PITCHING_H(9.25) +$$

$$PITCHING_HR(0.08) - PITCHING_BB(9.83) + PITCHING_SO(0.00) - FIELDING_E(46.98) -$$

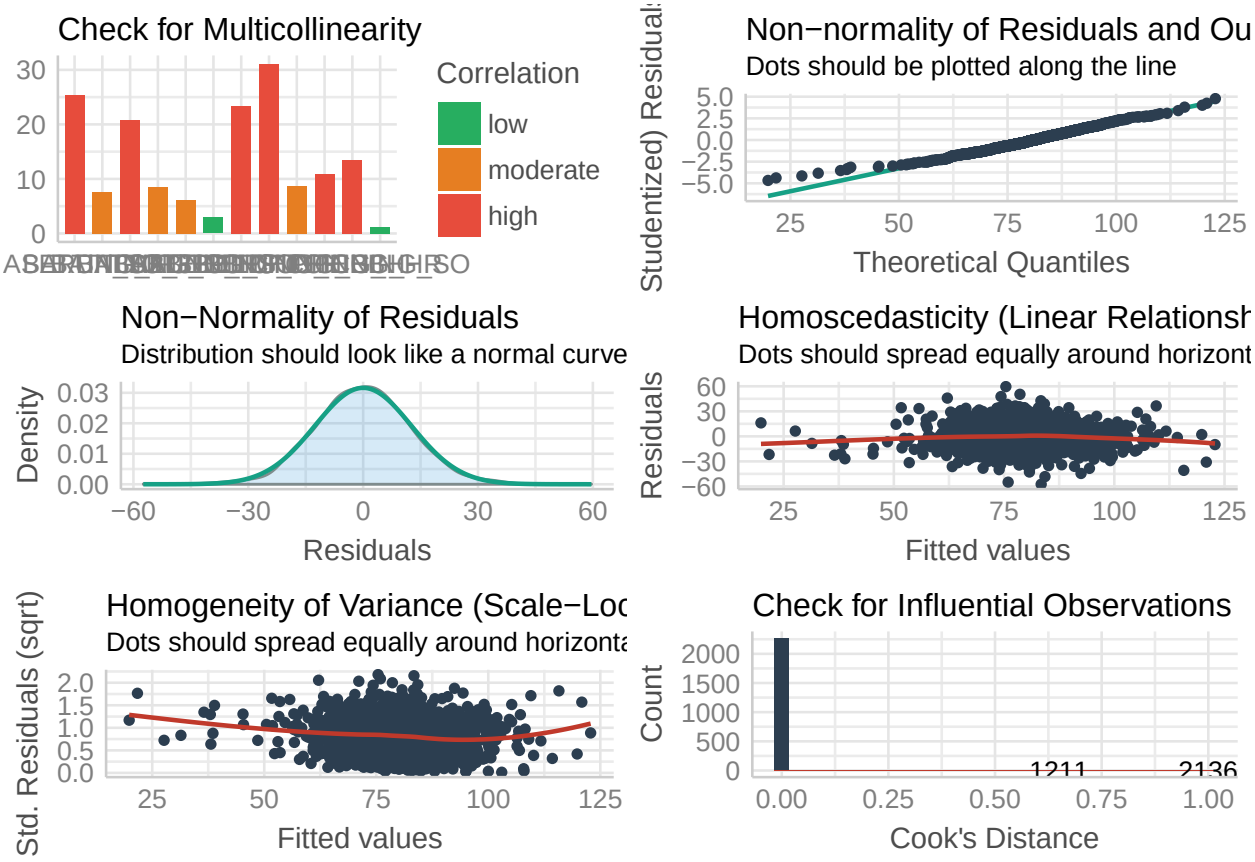
$$FIELDING_DP(0.14)$$

Table 5:

	Dependent variable:
	TARGET_WINS
BATTING_H	0.050***
	(0.004)
BATTING_2B	−0.043***
	(0.009)
BATTING_3B	0.105***
	(0.016)
BATTING_HR	0.016
	(0.026)
BATTING_BB	0.041***
	(0.005)
BATTING_SO	−0.022***
	(0.002)
BASERUN_SB	12.885***
	(1.303)
PITCHING_H	−9.252*
	(4.724)
PITCHING_HR	0.040*
	(0.023)
PITCHING_BB	−9.830**
	(3.888)
PITCHING_SO	0.003***
	(0.001)
FIELDING_E	−46.981***
	(2.928)
FIELDING_DP	−0.139***
	(0.013)
BATTING_SO_imp	7.456***
	(1.431)
BASERUN_SB_imp	24.571***
	(1.610)
PITCHING_SO_imp	
FIELDING_DP_imp	2.298*
	(1.302)
Constant	154.481***
	(16.373)
Observations	2,276

Select Model

As discussed we would prefer to select Model 4 because it not only improved adjusted r-square but the intercept also became significant which was not the case in model 3. In addition, the significance and directions of each variable's impact on TARGET_WINS mostly accords with the theoretical effect, and intuition. We found that knn imputation along with log transformation improved the model significantly in comparison with other techniques. It also shows that we cannot randomly just replace the missing values with mean or median. The analystm must check our models with different criteria to see which transformation techniques do well overall in terms of not only r-square but also we have to see if the result actually makes sense or not. At this point we are convinced about performance of model 4 but we'll take one step extra and double check.



Data looks normally distributed but looks like there are high collinearity among some of the variables as shown below which might be affect their significance or coefficients in final model. According to Jim Frost, multicollinearity does not influence the predictions, precision and goodness of fit. He adds that if primary goal is to make predictions then you do not have to understand the role of each independent variable. There is some homoskedasticity in residuals as shown in graph 2 in second row.

```
## # A tibble: 12 x 3
##    Parameter     VIF SE_factor
##    <chr>       <dbl>     <dbl>
##  1 BATTING_H    6.19      2.49
##  2 BATTING_2B   7.60      2.76
##  3 BATTING_3B  20.8       4.56
##  4 BATTING_BB   8.50      2.91
##  5 BATTING_SO   2.92      1.71
##  6 BASERUN_SB  25.5       5.05
```

```
##  7 PITCHING_H  10.9       3.30
##  8 PITCHING_HR 13.5       3.68
##  9 PITCHING_BB  8.76      2.96
## 10 PITCHING_SO  1.11      1.05
## 11 FIELDING_E  31.1       5.58
## 12 FIELDING_DP 23.4       4.84
```

```
## # A tibble: 1 x 5
##      AIC    BIC    R2 R2_adjusted  RMSE
##    <dbl>  <dbl> <dbl>       <dbl> <dbl>
## 1 18026. 18106. 0.358       0.355  12.6
```

It seems that model 4's performance is almost consistent with other models in terms of collinearity and RMSE but adjusted r2 is comparatively better than others. The model, also makes intuitive sense. This suggests it should be selected. Now that we have selected model 4 as a best among the others, let's move ahead and predict the target wins by predicting the test dataset on Model 4.

TARGET_WINS	BATTING_H	BATTING_2B	BATTING_3B	BATTING_HR	BATTING_BB	BATTING_
61	1209	170	33	83	447	1
64	1221	151	29	88	516	
73	1395	183	29	93	509	
87	1539	309	29	159	486	
62	1445	203	68	5	95	
65	1431	236	53	10	215	

Appendix 1 R Code

Since all the data is for the same team, we will remove the word 'TEAM' from all variables for ease of reading.

```
#Clean train dataset
name_list <- names(train)
name_list <- gsub("TEAM_", "", name_list)
names(train) <- name_list

#We do the same for the test dataset
name_list <- names(test)
name_list <- gsub("TEAM_", "", name_list)
names(test) <- name_list
```

Here is a preview of what the train dataset contains:

```
head(train[1], 6)
```

```
## # A tibble: 6 x 1
##   TARGET_WINS
##         <dbl>
## 1          39
## 2          70
## 3          86
## 4          70
## 5          82
## 6          75
```

The train dataset has 2276 records with 16 variables.Let us look at the structure of the data.

```
str(train)
```

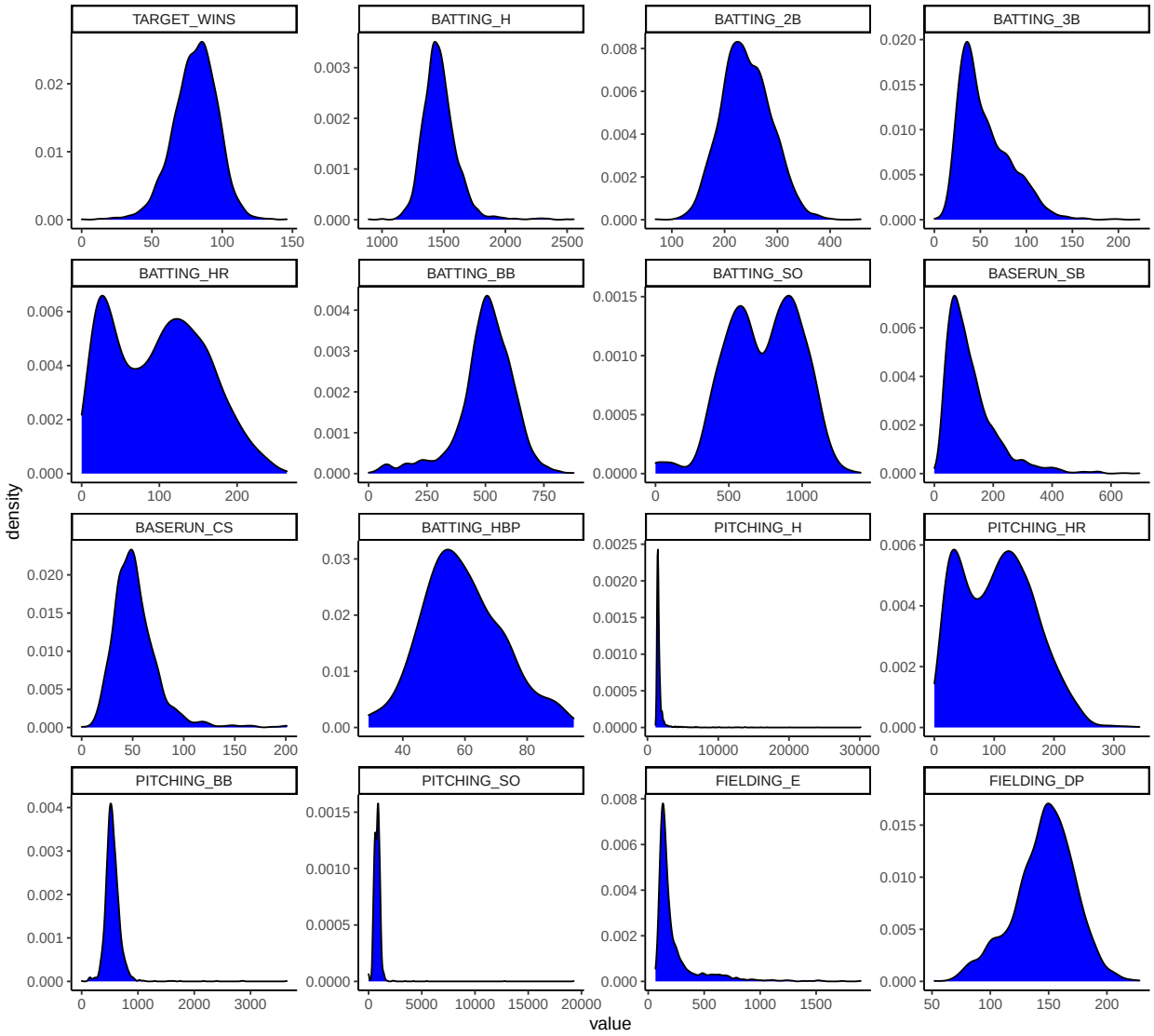
```
## tibble [2,276 x 16] (S3: tbl_df/tbl/data.frame)
##  $ TARGET_WINS: num [1:2276] 39 70 86 70 82 75 80 85 86 76 ...
##  $ BATTING_H  : num [1:2276] 1445 1339 1377 1387 1297 ...
##  $ BATTING_2B : num [1:2276] 194 219 232 209 186 200 179 171 197 213 ...
##  $ BATTING_3B : num [1:2276] 39 22 35 38 27 36 54 37 40 18 ...
##  $ BATTING_HR : num [1:2276] 13 190 137 96 102 92 122 115 114 96 ...
##  $ BATTING_BB : num [1:2276] 143 685 602 451 472 443 525 456 447 441 ...
##  $ BATTING_SO : num [1:2276] 842 1075 917 922 920 ...
##  $ BASERUN_SB : num [1:2276] NA 37 46 43 49 107 80 40 69 72 ...
##  $ BASERUN_CS : num [1:2276] NA 28 27 30 39 59 54 36 27 34 ...
##  $ BATTING_HBP: num [1:2276] NA NA NA NA NA NA NA NA NA NA ...
##  $ PITCHING_H : num [1:2276] 9364 1347 1377 1396 1297 ...
##  $ PITCHING_HR: num [1:2276] 84 191 137 97 102 92 122 116 114 96 ...
##  $ PITCHING_BB: num [1:2276] 927 689 602 454 472 443 525 459 447 441 ...
##  $ PITCHING_SO: num [1:2276] 5456 1082 917 928 920 ...
##  $ FIELDING_E : num [1:2276] 1011 193 175 164 138 ...
##  $ FIELDING_DP: num [1:2276] NA 155 153 156 168 149 186 136 169 159 ...
```

We can see that all of the variables are integers, and BATTING_HBP has a lot of missing values.We will look at the summary of the data to find some idea about the distributions.

```
summary(train)
```

```
##   TARGET_WINS      BATTING_H       BATTING_2B      BATTING_3B
##  Min.   :  0.00   Min.   : 891   Min.   : 69.0   Min.   :  0.00
##  1st Qu.: 71.00   1st Qu.:1383   1st Qu.:208.0   1st Qu.: 34.00
##  Median : 82.00   Median :1454   Median :238.0   Median : 47.00
##  Mean   : 80.79   Mean   :1469   Mean   :241.2   Mean   : 55.25
##  3rd Qu.: 92.00   3rd Qu.:1537   3rd Qu.:273.0   3rd Qu.: 72.00
##  Max.   :146.00   Max.   :2554   Max.   :458.0   Max.   :223.00
##
##    BATTING_HR      BATTING_BB      BATTING_SO      BASERUN_SB
##  Min.   :  0.00   Min.   :  0.0   Min.   :   0.0   Min.   :  0.0
##  1st Qu.: 42.00   1st Qu.:451.0   1st Qu.: 548.0   1st Qu.: 66.0
##  Median :102.00   Median :512.0   Median : 750.0   Median :101.0
##  Mean   : 99.61   Mean   :501.6   Mean   : 735.6   Mean   :124.8
##  3rd Qu.:147.00   3rd Qu.:580.0   3rd Qu.: 930.0   3rd Qu.:156.0
##  Max.   :264.00   Max.   :878.0   Max.   :1399.0   Max.   :697.0
##                                   NA's   :102      NA's   :131
##    BASERUN_CS      BATTING_HBP      PITCHING_H     PITCHING_HR
##  Min.   :  0.0   Min.   :29.00   Min.   : 1137   Min.   :  0.0
##  1st Qu.: 38.0   1st Qu.:50.50   1st Qu.: 1419   1st Qu.: 50.0
##  Median : 49.0   Median :58.00   Median : 1518   Median :107.0
##  Mean   : 52.8   Mean   :59.36   Mean   : 1779   Mean   :105.7
##  3rd Qu.: 62.0   3rd Qu.:67.00   3rd Qu.: 1682   3rd Qu.:150.0
##  Max.   :201.0   Max.   :95.00   Max.   :30132   Max.   :343.0
##  NA's   :772     NA's   :2085
##   PITCHING_BB      PITCHING_SO        FIELDING_E      FIELDING_DP
##  Min.   :   0.0   Min.   :    0.0   Min.   :  65.0   Min.   : 52.0
##  1st Qu.: 476.0   1st Qu.:  615.0   1st Qu.: 127.0   1st Qu.:131.0
##  Median : 536.5   Median :  813.5   Median : 159.0   Median :149.0
##  Mean   : 553.0   Mean   :  817.7   Mean   : 246.5   Mean   :146.4
##  3rd Qu.: 611.0   3rd Qu.:  968.0   3rd Qu.: 249.2   3rd Qu.:164.0
##  Max.   :3645.0   Max.   :19278.0   Max.   :1898.0   Max.   :228.0
##                   NA's   :102                        NA's   :286
```

```
m = melt(train)
ggplot(m, aes(x= value)) +
   geom_density(fill='blue') +
   facet_wrap(~variable, scales = 'free') +
   theme_classic()
```



We can see that 12 of the variables have a significant skew. only 4 of the 16 variables are normally or close to normally distributed. The response variable Target_wins seems to be normally distributed.Batting_Hr, Batting_SO and Pitching_HR are bi-modal.

10 of the 16 variables have a minimum value of 0. This is not a major concern as the total % of 0 in each column is less than 1%.

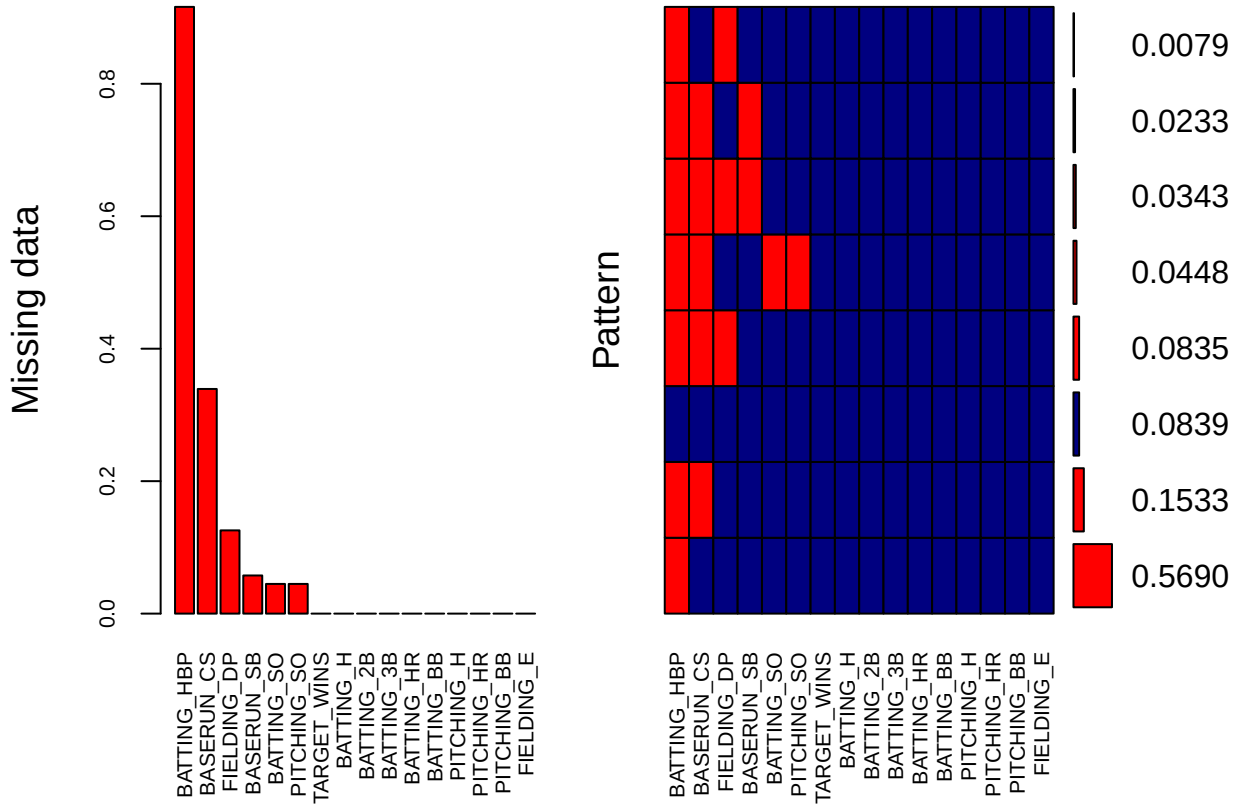
```
train %>%
  gather(variable, value) %>%
  filter(value == 0) %>%
  group_by(variable) %>%
  tally() %>%
  mutate(percent = n / nrow(train) * 100) %>%
  arrange(desc(n)) %>%
  rename(`Variable With Zeros` = variable,
         `Number of Records` = n,
         `Share of Total` = percent) %>%
  kable() %>%
```

```
  kable_styling()
```

6 of the variables have above 4% missing values:

- BATTING_SO: 102 (4.5% of total)
- PITCHING_SO: 102 (4.5% of total)
- BASERUN_SB: 131 (5.8% of total)
- BASERUN_CS: 772 (34% of total, needs to be looked at closely, impute or exclude?)
- BATTING_HBP: 2085 (92% of total, exclude from analysis?)
- FIELDING_DP: 286 (12.6% of total,needs to be looked at closely, impute or exclude?)

```
#Plot missing values using library VIM
mice_plot <- aggr(train, col=c('navyblue','red'),numbers=TRUE, sortVars=TRUE,labels=names(train), cex.a
```




```
##
##  Variables sorted by number of missings:
##      Variable       Count
##  BATTING_HBP 0.91608084
##    BASERUN_CS 0.33919156
##  FIELDING_DP 0.12565905
##    BASERUN_SB 0.05755712
##    BATTING_SO 0.04481547
```

```
##  PITCHING_SO 0.04481547
##  TARGET_WINS 0.00000000
##    BATTING_H 0.00000000
##   BATTING_2B 0.00000000
##   BATTING_3B 0.00000000
##   BATTING_HR 0.00000000
##   BATTING_BB 0.00000000
##   PITCHING_H 0.00000000
##  PITCHING_HR 0.00000000
##  PITCHING_BB 0.00000000
##   FIELDING_E 0.00000000
```

```
#Test for complete rows
print('Percentage of complete case is')
```

```
## [1] "Percentage of complete case is"
```

```
print((sum(complete.cases(train))/nrow(train))*100)
```

```
## [1] 8.391916
```

Outliers

```
ggplot(stack(train[,-1]), aes(x = ind, y = values, fill=ind)) +
  geom_boxplot(outlier.colour = "red",  outlier.alpha=.4) +
  coord_cartesian(ylim = c(0, 1000)) +
  theme_classic()+
  theme(axis.text.x=element_text(angle=45, hjust=1))
```

The variables Batting_BB, Batting_CS, Baserun_SB, Pitching_BB and Fielding_E have a significant number of outliers.

Correlations among predictors

```
train %>%
  cor(., use = "complete.obs") %>%
  corrplot(., method = "color", type = "upper", tl.col = "black", tl.cex=.8, diag = FALSE)
```

There are positive or negative correlations among the predictors. A small number of values are not correlated. Let us look at the numerical correlations with the response variable. We see that the predictors Batting_H, Batting_HR, Batting_BB, Pitching_H, and Pitching_HR are more correlated and should be included in our regression.

```
correlation <- train %>%
  cor(., use = "complete.obs") %>%
  as.data.frame() %>%
  rownames_to_column()%>%
  gather(Variable, Correlation, -rowname)

correlation %>%
  filter(Variable == "TARGET_WINS") %>%
     arrange(desc(Correlation)) %>%
  kable() %>%
  kable_styling()
```

Let us look at significant correlations among the independent variables. We see that four of the pairs have a correlation close to 1. This can lead to autocorrelation errors in our analysis.

```
correlation %>%
  filter(abs(Correlation) > .5 & Correlation!=1 ) %>%
  distinct(Correlation, .keep_all = T)%>%
  kable() %>%
  kable_styling()
```

Data Preparation

First we will remove Batting_HBP (Hit by Pitch) which has 92% missing values.

```
train <- train[-10]
```

We will look at the patterns and intersections of missingness among the variables, using the naniar package. We can see that only 22 of the observations have all 5 variables missing, we will just delete these cases. The pattern suggests that the variables are Missing at Random (MAR)

```
par(mfrow=c(1,2))
gg_miss_upset(train,
              nsets = 5,
              nintersects = NA)
gg_miss_case(train)+
  theme_classic()
```

```
train<-train%>%
   mutate(rowsum=rowSums(is.na(train)))%>%
   filter(rowsum<5)%>%
   select(-rowsum)
```

In deciding whether data is MCAR or MAR, one approach is to explore patterns of missingness between variables. This is particularly important for a primary outcome measure/dependent variable. We will test the two cases with most missing values.

The wins for players who's Baserun_cs data is known is the blue box plot, and the wins with missing Baserun_cs data is the gray box plot.We do not see a lot of difference in the mean, but the Interquartile range has changed. Also the mean changes significantly between Fielding_DP and Baserun_cs if we do not include missing data.

```
library(finalfit)
explanatory <- c('BASERUN_CS','FIELDING_DP' )
dependent <- 'TARGET_WINS'
train %>%
  missing_pairs(dependent, explanatory)
```

102 of the observations have 3 of the variables missing.The highest case missing is in the BASERUN_CS variable, so this deserves an analysis of its own.We find a highly positively skewed distribution with most missing data after the 125 on the x axis. We want to make sure that the shape of our distribution does not change as we impute the data.

```
train %>%
  ggplot(aes(BASERUN_CS)) +
  geom_histogram(bins = 50) +
  geom_vline(aes(xintercept = mean(BASERUN_CS, na.rm = T)), col = "red", lty = 2) +
  geom_vline(aes(xintercept = median(BASERUN_CS, na.rm = T)),col = "green", lty = 2) +
  labs(x = element_blank(),
       y = "Count",
       title = "Distribution of Caught Stealing Bases",
       caption = "* Red line is the mean value and green is the median")
```

Modeling

Model 1 - Basic Model

```
# Replacing extreme values with median for training & evaluation datasets and removing BATTING_HBP whic
train2 <- train %>% mutate(PITCHING_H = if_else(PITCHING_H > 5000, median(PITCHING_H), PITCHING_H),
                           PITCHING_SO = if_else(PITCHING_SO > 1500, median(PITCHING_SO), PITCHING_SO))
  select(-BATTING_HBP)

test2 <- test %>% mutate(PITCHING_H = if_else(PITCHING_H > 5000, median(PITCHING_H), PITCHING_H),
                           PITCHING_SO = if_else(PITCHING_SO > 1500, median(PITCHING_SO), PITCHING_SO))
  select(-BATTING_HBP)
```

```
# Replacing missing values with median
train2[] <- lapply(train2, function(x) ifelse(is.na(x), median(x, na.rm=TRUE), x))
test2[] <- lapply(test2, function(x) ifelse(is.na(x), median(x, na.rm=TRUE), x))


# Verifying for missing values
#sapply(train2, function(x) sum(is.na(x)))
#vis_miss(train2) # From naniar library

# Checking skewness
#sapply(train2, function(x) skewness(x))
```

Model 2 - Log Transformation

```
train_log <- train2 # Model 2
test_log <- test2

# Applying log transformation for highly skewed variables
#training
train_log$PITCHING_BB <- log10(train_log$PITCHING_BB + 1)
train_log$PITCHING_H <- log10(train_log$PITCHING_H + 1)
train_log$FIELDING_E <- log10(train_log$FIELDING_E + 1)
train_log$BASERUN_SB <- log10(train_log$BASERUN_SB + 1)
train_log$BASERUN_CS <- log10(train_log$BASERUN_CS + 1)
#test
test_log$PITCHING_BB <- log10(test_log$PITCHING_BB + 1)
test_log$PITCHING_H <- log10(test_log$PITCHING_H + 1)
test_log$FIELDING_E <- log10(test_log$FIELDING_E + 1)
test_log$BASERUN_SB <- log10(test_log$BASERUN_SB + 1)
test_log$BASERUN_CS <- log10(test_log$BASERUN_CS + 1)
```

```
# CHecking skewness
sapply(train_log, function(x) skewness(x))

# Printing summary statistics
#summary(train_log)

# Skewness
ggplot(melt(train_log), aes(x=value))+geom_density()+facet_wrap(~variable, scales='free')
```

Model 3 - BoxCox Transformation

```
# Converting tibble to df
train_data_bx <- data.frame(train2)
test_data_bx <- data.frame(test2)

dim(test2)
# PreProcess function for boxcox transformation
preproc_value <- preProcess(train_data_bx[,-1], c("BoxCox", "center", "scale"))

# Transformation
train_bx_transformed <- predict(preproc_value, train_data_bx)
test_bx_transformed <- predict(preproc_value, test_data_bx)

# Normality and skewness
ggplot(melt(train_bx_transformed), aes(x=value))+geom_density()+facet_wrap(~variable, scales='free')
sapply(train_bx_transformed, function(x) skewness(x))
```

Model 4 - kNN Imputation on missing values

kNN Imputation

```
set.seed(1100)
library(VIM)

# knn imputation for missing values
# training
train_data_knn <- train %>% # TRAINING
  select(-BATTING_HBP, -BASERUN_CS)
train_knn <- kNN(train_data_knn, variable = c("BATTING_SO", "BASERUN_SB", "PITCHING_SO","FIELDING_DP"),

#test
test_data_knn <- test %>% # TEST
  select(-BATTING_HBP, -BASERUN_CS)
test_knn <- kNN(test_data_knn, variable = c("BATTING_SO", "BASERUN_SB", "PITCHING_SO","FIELDING_DP"),k=

# CHecking for missing values
colSums(is.na(train_knn))

# Log transformation on knn imputed dataset
train_knn_log <- train_data_knn # Model 2
test_knn_log <- test_data_knn
#test_log <- test2
```

```r
# Applying log transformation for highly skewed variables
#training
train_knn_log$PITCHING_BB <- log10(train_knn_log$PITCHING_BB + 1)
train_knn_log$PITCHING_H <- log10(train_knn_log$PITCHING_H + 1)
train_knn_log$FIELDING_E <- log10(train_knn_log$FIELDING_E + 1)
train_knn_log$BASERUN_SB <- log10(train_knn_log$BASERUN_SB + 1)

# TEST DATASET TRANSFORMATION
test_knn_log$PITCHING_BB <- log10(test_knn_log$PITCHING_BB + 1)
test_knn_log$PITCHING_H <- log10(test_knn_log$PITCHING_H + 1)
test_knn_log$FIELDING_E <- log10(test_knn_log$FIELDING_E + 1)
test_knn_log$BASERUN_SB <- log10(test_knn_log$BASERUN_SB + 1)

# CHecking skewness and normality
sapply(train_knn_log, function(x) skewness(x))
ggplot(melt(train_knn_log), aes(x=value))+geom_density()+facet_wrap(~variable, scales='free')
```

References

https://easystats.github.io/performance/

https://statisticsbyjim.com/regression/multicollinearity-in-regression-analysis/#:~:text=Multicollinearity%20makes%20it%