Business Analytics Home Work 1

John Deblase, Sekhar Mekala, Sonya Hong

Sunday, September 18, 2016

Project requirements

The main goal of this project is to perform data analysis of baseball team performance and predict the number of wins based on a given set of inputs. We are given two data sets. The training data set and the test data set. The training data has input variables along with the observed response variable (number of wins). We will use the training data set to train our model, and the predictions obtained on the test data will be submitted as a project deliverable.

Data Exploration

TARGET_WINS will be the dependent variable, and the remaining variables will be independent variables for our predictive models:

Table 1: Training data set's variables

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

In the test data set (moneyball-evaluation-data.csv), we have the the same set of variables, except the TARGET_WINS variable. Our goal is to predict this variable's value in the test data as accurately as possible. We will use 5 fold cross validation technique to estimate our models accuracy using the training data.

A summary of all the variables in the training data is given below:

Figure 1: Summary of training data set

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

In the above display, the INDEX Variable contains the observation number. The ranges of two variables, TEAM_PITCHING_SO and TEAM_PITCHING_H is questionable, since their maximum values are very large.

A number of outliers might need to be removed from these variables in order to build more accurate models.

The following variables have NA values (NA represents unknown values).

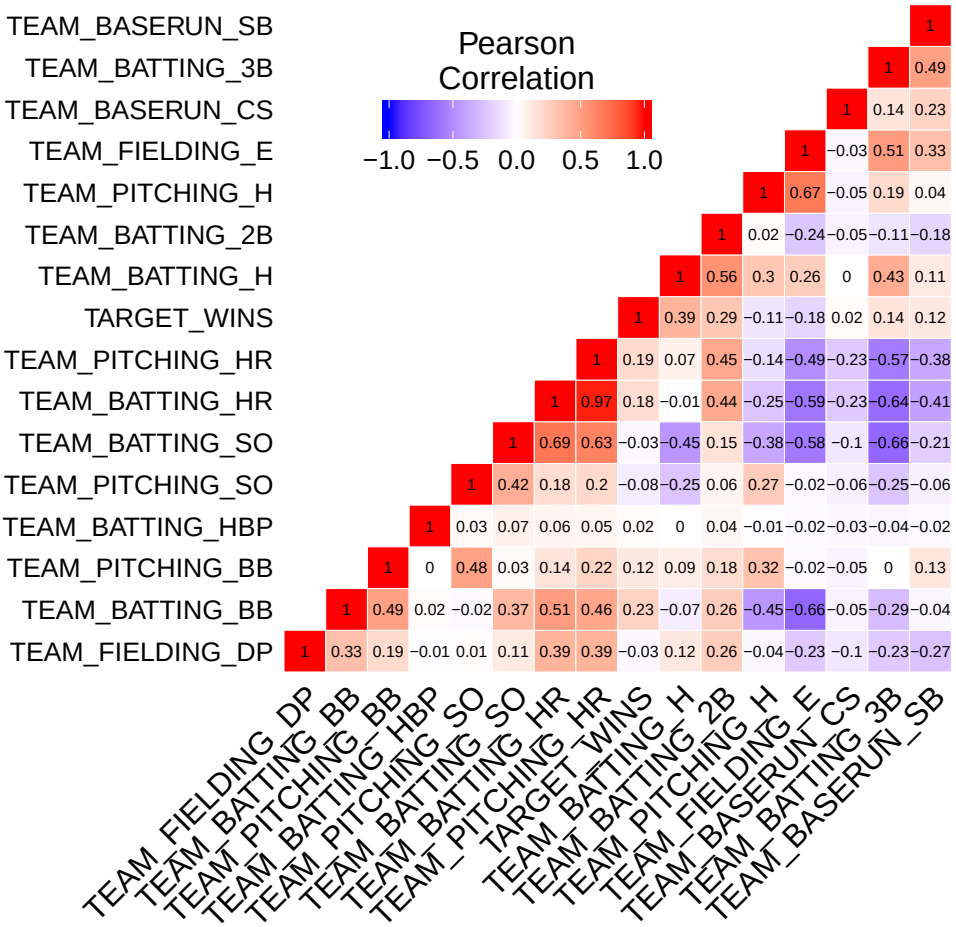
Table 2: NA values summary in training data

Variable	Number_of_NA	NA_Percentage
TEAM_BATTING_SO	102	4.48%
TEAM_BASERUN_SB	131	5.76%
TEAM_BASERUN_CS	772	33.92%
TEAM_BATTING_HBP	2085	91.61%
TEAM_PITCHING_SO	102	4.48%
TEAM_FIELDING_DP	286	12.57%

Since we have considerable percentage of unknown values, we will build the models in two ways. In the first approach, we will use the median value of the respective variable to represent the unknown value, and in the second approach we will use dummy variables to represent the unknown variables. The approach that results in best model's performance will be finally used to calculate the TARGET_WINS using the test data.

We will now analyze the correlation between the variables in the training data.

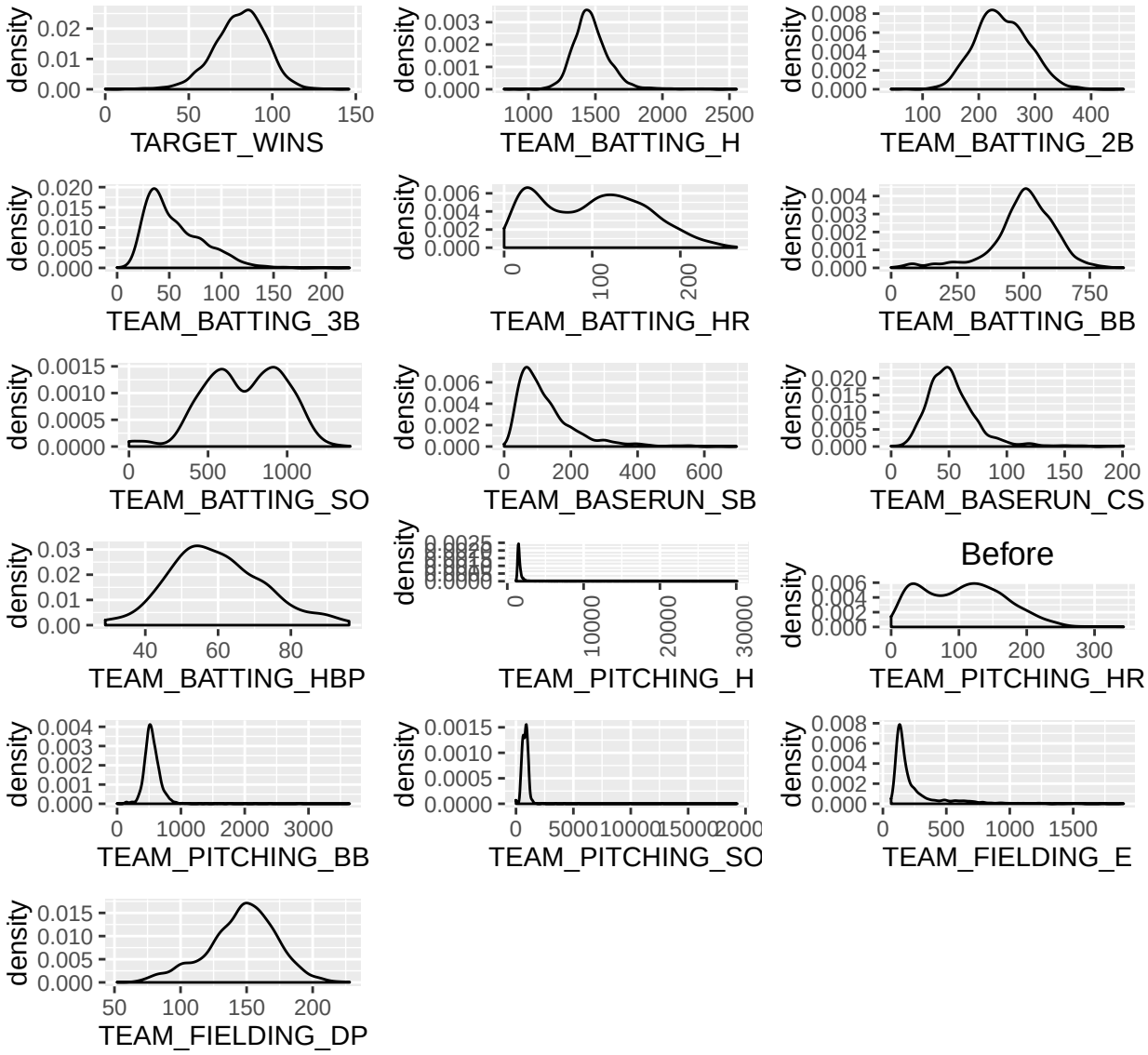
Figure 2: Heat map showing overall correlations



The heat map shows that TEAM_BATTING_HR and TEAM_PITCHING_HR are highly correlated with each other. In fact the correlation is so strong that, we may have to consider only one of this variable in our model, to avoid collinearity. We will refer to this correlation map while building predictive models in the *Model building* section of this document.

The density plots of the combined training and test data is shown below. These plots will show if any of the variables have skewed data that can potentially be fixed via transformation.

Figure 3: Density plots of variables



Based on these density plots we can apply some simple transformations to different variables in order to normalize the data.

Most of the variables data are either skewed right or approximately normal. For the skewed right variables, we applied 3 different types of transformations. If a variable is skewed right, apply the log (base e) transformation. But if the variable is having 0 values, then use a square root transformation. For one variable we had to use cube root to make the density approximately normal. See *Appendix-A* for a comparision of the variables densities before and after transformation. The below table shows the summary of the proposed transformations.

Table 3: Suggested variable transformations summary

Variable	Transformation
TARGET_WINS	None
TEAM_BATTING_H	ln
TEAM_BATTING_2B	None
TEAM_BATTING_3B	Square root
TEAM_BATTING_HR	Square root
TEAM_BATTING_BB	None
TEAM_BATTING_SO	None
TEAM_BASERUN_SB	Cubic root
TEAM_BASERUN_CS	Square root
TEAM_BATTING_HBP	None
TEAM_PITCHING_H	ln
TEAM_PITCHING_HR	None
TEAM_PITCHING_BB	Square root
TEAM_PITCHING_SO	Square root
TEAM_FIELDING_E	ln
TEAM_FIELDING_DP	None

Summary of data exploration

- We have unknown values in 6 variables, and one of these 6 variables have more than 90% of the unknown data.
- TEAM_PITCHING_H and TEAM_PITCHING_SO have very big values, which are questionable and we may have to eliminate those values.
- TEAM_BATTING_HR is highly correlated to TEAM_PITCHING_HR, and we may have to drop one of these variables to avoid collinearity in the model. TEAM_BATTING_HR is correlated to most of the variables in the data set.
- Variables will be transformed according to table 3

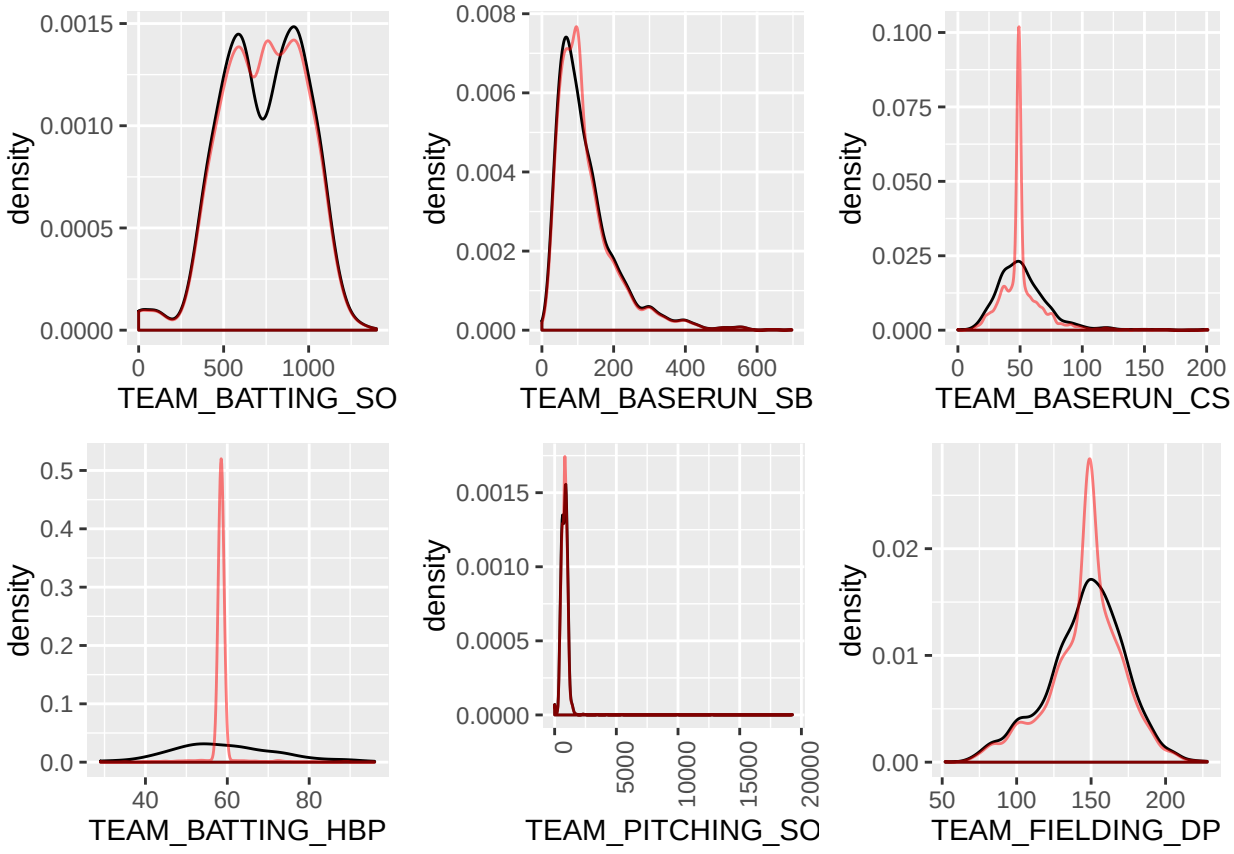
Data Preparation

We will handle the NA values in 2 methods. In the first method, we will replace all the NA values in a variable with its median value. This modified data will be used to build linear models. In the second method we will use dummy variables to handle NA values, and use the resulting data set for building another set of models. We will apply the same data changes to the test data as we apply to the training data, so that the model developed on the training data can be easily applied to the test data, without any change to the model coefficients.

Replacing the missing values with median values

We have the 6 variables in the training data that have NA values. The density plots of these variables change the following way, if we replace each of the variable's NA values with the median value (of the respective variable):

Figure 4: Replacing NAs with Median values (Red colored line shows the distribution after replacement)



The following data frames are created by replacing the NA values with the median values (the median values are obtained by combining the test and training data sets), and also transforming the variables as suggested in Table 3:

- *median_train_df* - data frame containing the training data
- *median_test_df* - data frame containing the test data

The *median_train_df* is further modified to exclude records for which the variables data TEAM_PITCHING_H is greater than 3000 and TEAM_PITCHING_SO is greater than 1600. The 3000 and 1600 values are the approximate cut-off values, which are obtained by adding 1.5 times IQR (Inter Quartile Range) to the value at 75th percentile, and rounding off the number to the nearest 100.

Creating dummy variables to handle NA values

We will now prepare 6 dummy variables to handle NA values for each of the 6 variables. Each of the dummy variable contains 1 or 0, with the following notation:

- *For each variable with at least one NA value, we will have a dummy variable*
- *If the variable has a NA at a specific row, then the corresponding row in the dummy variable will have 1, else the dummy variable's row will have 0*
- *All the NA values will be replaced by 1*

The following data frames are created by creating dummy variables for variables with NA values, and also transforming the variables as suggested in Table 3:

- *mark_na_train_df* - data frame containing the training data
- *mark_na_test_df* - data frame containing the test data

The *mark_na_train_df* is further modified to exclude records for which the variables data TEAM_PITCHING_H is greater than 3000 and TEAM_PITCHING_SO is greater than 1600

In the next section we will fit linear models using the data sets created in this section.

Model building

Our first linear model is built from the imputed median dataset.

We call this model as Model-0.

Model-0

$$TARGET\_WINS = -393 + 68.56(TEAM\_BATTING\_H) - 0.02602(TEAM\_BATTING\_2B)$$
$$+1.485(TEAM\_BATTING\_3B) - 0.1673(TEAM\_BATTING\_HR)$$
$$+0.03631(TEAM\_BATTING\_BB) - 0.01754(TEAM\_BATTING\_SO)$$
$$+3.167(TEAM\_BASERUN\_SB) - 0.6282(TEAM\_BASERUN\_CS)$$
$$+0.07934(TEAM\_BATTING\_HBP) + 3.653(TEAM\_PITCHING\_H)$$
$$+0.0514(TEAM\_PITCHING\_HR) - 0.9378(TEAM\_PITCHING\_BB)$$
$$+0.5379(TEAM\_PITCHING\_SO) - 10.99(TEAM\_FIELDING\_E)$$
$$-0.1164(TEAM\_FIELDING\_DP)$$

The above model's $Adj.R^2$, Residual Sum of Squares, and F-statistic details are given in the following table:

Table 4: Model-0 statistics

Quantity	Value
Adj. R-Squared	0.3157
F-statistic	70.9800
F-Statistic's p-Value	0.0000
Residual Standard Error	13.0300
5-fold cross validation error	173.8388

The p-value of the F-Statistic is zero, which suggests that the dependent variable (TARGET_WINS) is associated with at least one of the independent variable. The $Adj.R^2$ is not high, but we will evaluate our model's performance using the cross validation method (5 fold cross validation method using R's *boot* package). The cross validation error for this model is 173.8388.

Cross validation method

The *cross validation* method will hold 5 observations, fit the same model on the remaining observations, and evaluate the model's Mean Squared Error (on the held 5 observations). This is repeated till all the observations are in the hold set once. The mean of all the mean squared errors calculated is the cross validation error. Cross validation error gives an estimate of model's performance when the model is used to predict the unseen data. Lesser the cross validation error, better is the model.

Model-0 coefficients analysis

Using the p-value of F-Statistic we confirmed that the TARGET_WINS is highly associated with the inependent variables, and the independent variables can be used to estimate the TARGET_WINS value for unseen data.

Model-0 suggests that TEAM_BATTING_H (Base Hits by batters (1B,2B,3B,HR)) has a huge positive impact on the TARGET_WINS (number of wins) variable. For every 1 unit increase in TEAM_BATTING_H, the target wins will increase by 68.56 (holding all the other variables constant). The TEAM_FIELDING_E (Errors) has a huge negative impact. For every 1 unit increase in TEAM_FIELDING_E the TARGET_WINS variable decrease by 10.99 units (holding all the other variables constant). Surprisingly the TEAM_BATTING_2B (Doubles by batters), TEAM_BATTING_HR (Homeruns by batters (4B)) is having negative impact on the TARGET_WINS, while these should increase the TARGET_WINS in the real game. The reason could be due to high correlation coefficient between TEAM_BATTING_H and TEAM_BATTING_2B (a correlation coefficient of 0.56), and TEAM_PITCHING_HR (Homeruns allowed) and TEAM_BATTING_HR (a correlation coefficient of 0.97). The TEAM_PITCHING_HR is supposed to have negative impact, but in Model-1, we see a positive coefficient for this variable. Such surprising relationship of the independent variables, with TARGET_WINS is due to strong correlation coefficients among the variables. If the effect of a variable "x" on TARGET_WINS is represented by another highly correlated variable "y", then the variable "x" should have a high p-value, and also its coefficient should vary in the vicinity of 0 (the coefficient range at 95% confidence level can be obtained by adding and subtracting 2 times the std. error to/from the variable coefficient). The coefficients of correlation among all the variables is given in 2. The p-values and standard errors of the Model-0 coefficients are given in the below table (Table 5)

Table 5: Model-0 coefficients p-values and standard errors

Variable	Coefficient	Std_Error	p_value
(Intercept)	-393	39.83	0.0000000000000002
TEAM_BATTING_H	68.56	6.182	0.0000000000000002
TEAM_BATTING_2B	-0.02602	0.00928	0.005085
TEAM_BATTING_3B	1.485	0.27	0.0000000418
TEAM_BATTING_HR	-0.1673	0.4729	0.723608
TEAM_BATTING_BB	0.03631	0.007173	0.000000449
TEAM_BATTING_SO	-0.01754	0.003531	0.000000728
TEAM_BASERUN_SB	3.167	0.3913	0.000000000000000931
TEAM_BASERUN_CS	-0.6282	0.2619	0.016516
TEAM_BATTING_HBP	0.07934	0.07303	0.277424
TEAM_PITCHING_H	3.653	2.595	0.159392
TEAM_PITCHING_HR	0.0514	0.02212	0.020218
TEAM_PITCHING_BB	-0.9378	0.2575	0.000277
TEAM_PITCHING_SO	0.5379	0.1124	0.00000183
TEAM_FIELDING_E	-10.99	1.097	0.0000000000000002
TEAM_FIELDING_DP	-0.1164	0.0134	0.0000000000000002

The p-value of TEAM_BATTING_2B is around 0.5%,but its coefficient range is between -0.00746 and 0.04458 (obtained using $Coefficient \pm 2 * Std.Error$). The TEAM_BATTING_HR's p-value is around 72%, and hence we obtained the coefficient of TEAM_BATTING_HR due to randomness. Based on the p-values, we can see that not all the variables are significant. We can certainly eliminate some of these variables, while not compromising on the model's $Adj.R^2$. NOTE that we cannot use just p-value to determine which variables are important, since there is always a chance that 5% of the p-values associated with the variables is always below 5% by chance. Hence we have to use a variable selection method to select the important variables. We will use the *Mixed selection* method to select the variables. In this method, we start with no variables in the model, and we add the variable that provides the best fit. We continue to add variables one-by-one. As new variables are added to the model, the p-values of the existing variables in the model may become larger. We take out the variables which have larger p-values from the model. We continue to perform these forward and backward steps until all variables in the model have sufficiently low p-values, and all variables outside the model would have a large p-value if added to the model.

We will use stepAIC() function from the MASS library to determine which variables should be in the final model. This function performs mixed selection, and gives us the list of variables that should be in the final model. When we executed the variable selection method on Model-0, we obtained the following model.

We call this model as Model-1. This model is obtained by eliminating the insignificant variables from Model-0

Model-1

$$TARGET\_WINS = -388 + 71.46(TEAM\_BATTING\_H)$$
$$-0.02458(TEAM\_BATTING\_2B) + 1.36(TEAM\_BATTING\_3B)$$
$$+0.02818(TEAM\_BATTING\_BB) - 0.01794(TEAM\_BATTING\_SO)$$
$$+3.085(TEAM\_BASERUN\_SB) - 0.644(TEAM\_BASERUN\_CS)$$
$$+0.04743(TEAM\_PITCHING\_HR) - 0.669(TEAM\_PITCHING\_BB)$$
$$+0.5143(TEAM\_PITCHING\_SO) - 10.06(TEAM\_FIELDING\_E)$$
$$-0.1183(TEAM\_FIELDING\_DP)$$

Table 6: Model-1 statistics

Quantity	Value
Adj. R-Squared	0.3154
F-statistic	88.3500
F-Statistic's p-Value	0.0000
Residual Standard Error	13.0300
5-fold cross validation error	172.1424

We can see that Model-1 is obtained by eliminating the 3 variables TEAM_PITCHING_H, TEAM_BATTING_HR and TEAM_BATTING_HBP from Model-0. Although these 3 variables were eliminated, the $Adj.R^2$ has not changed much, and the Cross validation error has slightly decreased.

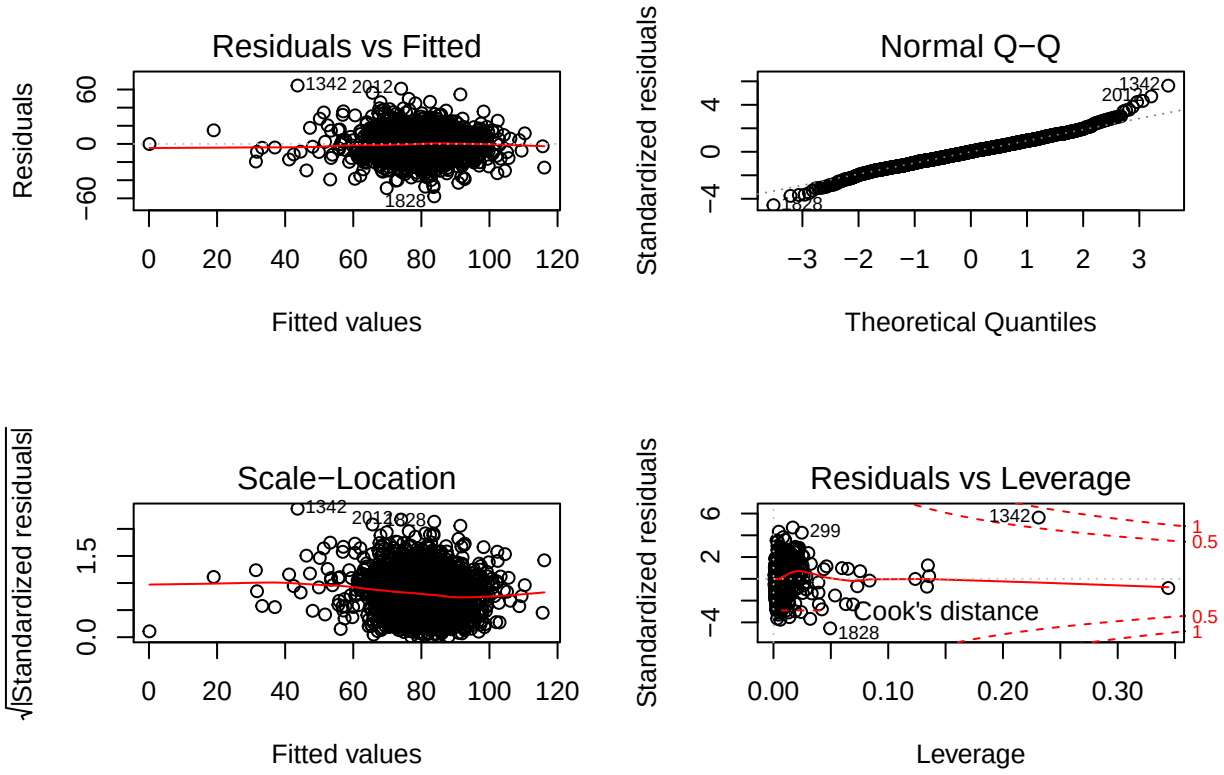
Table 7: Model-1's p-values, std. error and coefficients. Model-1 is obtained by eliminating the unnecessary variables from Model-0

Variable	Coefficient	Std_error	p_value
(Intercept)	-388.4	39.66	0.0000000000000002
TEAM_BATTING_H	71.46	5.638	0.0000000000000002
TEAM_BATTING_2B	-0.02458	0.009237	0.00784
TEAM_BATTING_3B	1.36	0.2553	0.00000011
TEAM_BATTING_BB	0.02818	0.005298	0.000000115
TEAM_BATTING_SO	-0.01794	0.003511	0.000000348
TEAM_BASERUN_SB	3.085	0.3819	0.00000000000000106
TEAM_BASERUN_CS	-0.644	0.2618	0.01396
TEAM_PITCHING_HR	0.04743	0.008873	0.0000000992
TEAM_PITCHING_BB	-0.669	0.2032	0.00101
TEAM_PITCHING_SO	0.5143	0.1117	0.00000435
TEAM_FIELDING_E	-10.06	0.9623	0.0000000000000002
TEAM_FIELDING_DP	-0.1183	0.013	0.0000000000000002

The p-values of all the coefficients are below 5%. This model (Model-1) shows the important variables that effect the TARGET_WINS variable.

Let us plot the residual plot for Model-1, to determine if we have any abnormalities such as non-constant variance in the errors or if a non-linear model is appropriate.

Figure 6: Residual plots of Model-1



The residual plot of Model-1 does not really have any specific pattern, although we have some high leverage points and non-constant variance. Also the quantile plot shows deviations from the normal distribution at extreme values.

Model-2

The final model will be built using the modified dataset that contains dummy variables. Applying the lm() function, and subsequently applying the same stepAIC() for variable selection has given the following model.

We call this model as Model-2

$$TARGET\_WINS = -356.5 + 73.64(TEAM\_BATTING\_H)$$
$$-0.02576(TEAM\_BATTING\_2B) + 1.764(TEAM\_BATTING\_3B)$$
$$+3.523(TEAM\_BASERUN\_SB) + 40.25(MARK\_NA\_TEAM\_BASERUN\_SB)$$
$$-0.5665(TEAM\_BASERUN\_CS) + 6.811(MARK\_NA\_TEAM\_BATTING\_HBP)$$
$$0.03137(TEAM\_PITCHING\_HR) + 0.8118(TEAM\_PITCHING\_BB)$$

$$-0.3175(TEAM\_PITCHING\_SO) - 24(TEAM\_FIELDING\_E)$$
$$-0.1079(TEAM\_FIELDING\_DP) - 9.159(MARK\_NA\_TEAM\_FIELDING\_DP)$$

In the above model, we have new variables (not present in the given test/training data sets). These variables are dummy variables to handle NA values. The dummy variables used in the above model have the following significance:

MARK_NA_TEAM_BASERUN_SB - This variable will have 1 if TEAM_BASERUN_SB is having NA value, else, it will have 0

MARK_NA_TEAM_BATTING_HBP - This variable will have 1 if TEAM_BATTING_HBP is having NA value, else, it will have 0

MARK_NA_TEAM_FIELDING_DP - This vriable will have 1 if TEAM_FIELDING_DP is having NA value, else, it will have 0

In the above model (Model-2), we can see that the variable TEAM_BATTING_HBP is not used, but its dummy variable MARK_NA_TEAM_BATTING_HBP is used. The variable TEAM_BATTING_HBP has more than 90% of its values as NA values. The Model-2's $Adj.R^2$, F-statistic and its p-vale, Cross validation error are given in table 8.

Table 8: Model-2 statistics

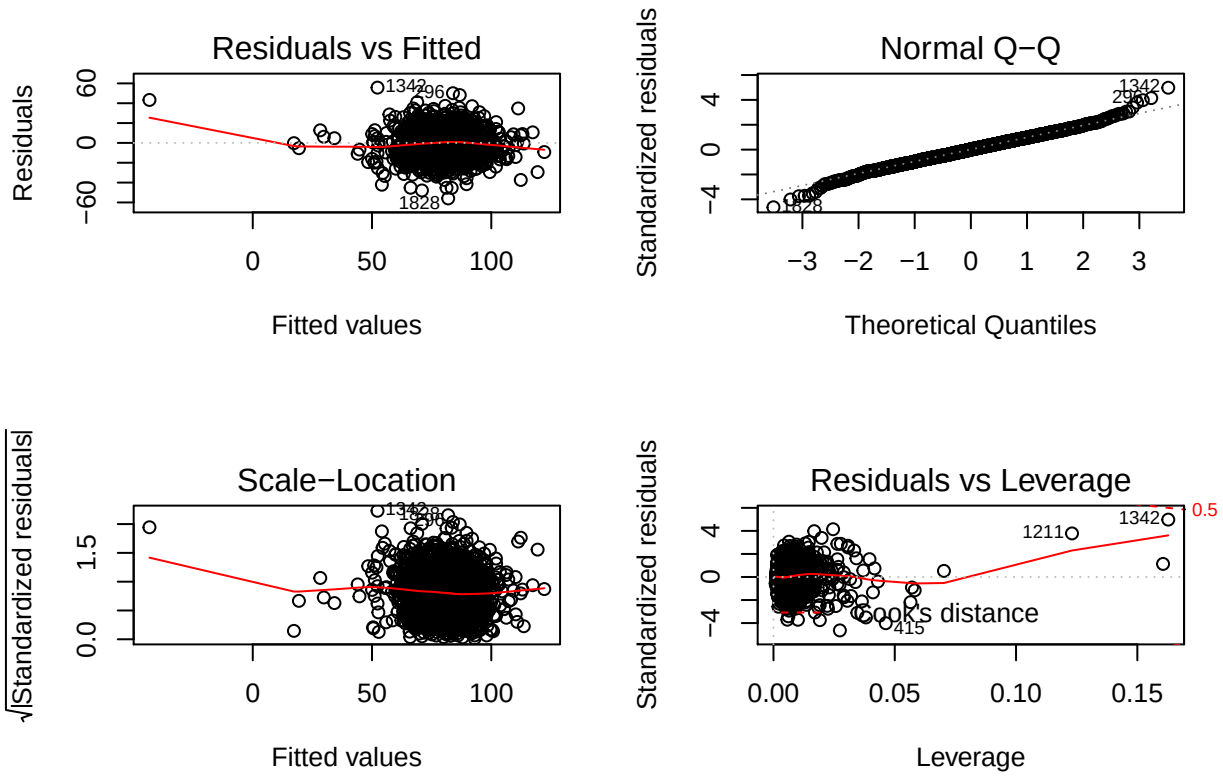
Quantity	Value
Adj. R-Squared	0.3974
F-statistic	116.4000
F-Statistic's p-Value	0.0000
Residual Standard Error	12.2300
5-fold cross validation error	152.5300

This model has obtained a cross validation error of approximately 152.53. The p-values of the variable are given below:

Table 9: Model-2 p-values. This model uses dummy variables for NA values

Variable	Coefficients	Std_error	p_value
(Intercept)	-356.5	31.32	0.0000000000000002
TEAM_BATTING_H	73.64	4.681	0.0000000000000002
TEAM_BATTING_2B	-0.02576	0.008983	0.00418
TEAM_BATTING_3B	1.764	0.2378	0.000000000000168
TEAM_BASERUN_SB	3.523	0.3425	0.0000000000000002
MARK_NA_TEAM_BASERUN_SB	40.25	2.275	0.0000000000000002
TEAM_BASERUN_CS	-0.5665	0.112	0.000000453
MARK_NA_TEAM_BATTING_HBP	6.811	1.084	0.00000000039
TEAM_PITCHING_HR	0.03137	0.007372	0.0000217
TEAM_PITCHING_BB	0.8118	0.1029	0.00000000000000466
TEAM_PITCHING_SO	-0.3175	0.0441	0.000000000000812
TEAM_FIELDING_E	-24	1.115	0.0000000000000002
TEAM_FIELDING_DP	-0.1079	0.01372	0.00000000000000567
MARK_NA_TEAM_FIELDING_DP	-9.159	1.9	0.00000152

Figure 7: Model-2 (lm.fit2) residual plot



The residual plot for Model-2 is almost similar to Model-1. We have the same problems (the non-constant variance, high leverage points) as in Model-1. But the Q-Q plot for Model-2 is better than Model-1.

Model selection

We developed the following 2 models:

Model-1

$$TARGET\_WINS = -388 + 71.46(TEAM\_BATTING\_H)$$
$$-0.02458(TEAM\_BATTING\_2B) + 1.36(TEAM\_BATTING\_3B)$$
$$+0.02818(TEAM\_BATTING\_BB) - 0.01794(TEAM\_BATTING\_SO)$$
$$+3.085(TEAM\_BASERUN\_SB) - 0.644(TEAM\_BASERUN\_CS)$$
$$+0.04743(TEAM\_PITCHING\_HR) - 0.669(TEAM\_PITCHING\_BB)$$
$$+0.5143(TEAM\_PITCHING\_SO) - 10.06(TEAM\_FIELDING\_E)$$
$$-0.1183(TEAM\_FIELDING\_DP)$$

Model-2

$$TARGET\_WINS = -356.5 + 73.64(TEAM\_BATTING\_H)$$
$$-0.02576(TEAM\_BATTING\_2B) + 1.764(TEAM\_BATTING\_3B)$$
$$+3.523(TEAM\_BASERUN\_SB) + 40.25(MARK\_NA\_TEAM\_BASERUN\_SB)$$
$$-0.5665(TEAM\_BASERUN\_CS) + 6.811(MARK\_NA\_TEAM\_BATTING\_HBP)$$
$$0.03137(TEAM\_PITCHING\_HR) + 0.8118(TEAM\_PITCHING\_BB)$$
$$-0.3175(TEAM\_PITCHING\_SO) - 24(TEAM\_FIELDING\_E)$$
$$-0.1079(TEAM\_FIELDING\_DP) - 9.159(MARK\_NA\_TEAM\_FIELDING\_DP)$$

Model-1 was developed by using the training data set with imputed median value and Model-2 was developed by using dummy variables to handle NA values. The cross validation error of Model-1 is approximately 172 and the cross validation error of Model-2 is approximately 153. Given this huge cross validation error difference, we propose Model-2 to predict the TARGET_WINS value in the test data set.

We will use *mark_na_test_df* to predict the TARGET_WINS variable. The predicted values will be appended to the test data set (moneyball-evaluation-data.csv) for final submission. The R command to generate the TARGET_WINS prediction and preparing the file for final submission is listed in Appendix-B. Appendix-B also contains the R code used to prepare the predictive models.

Future work

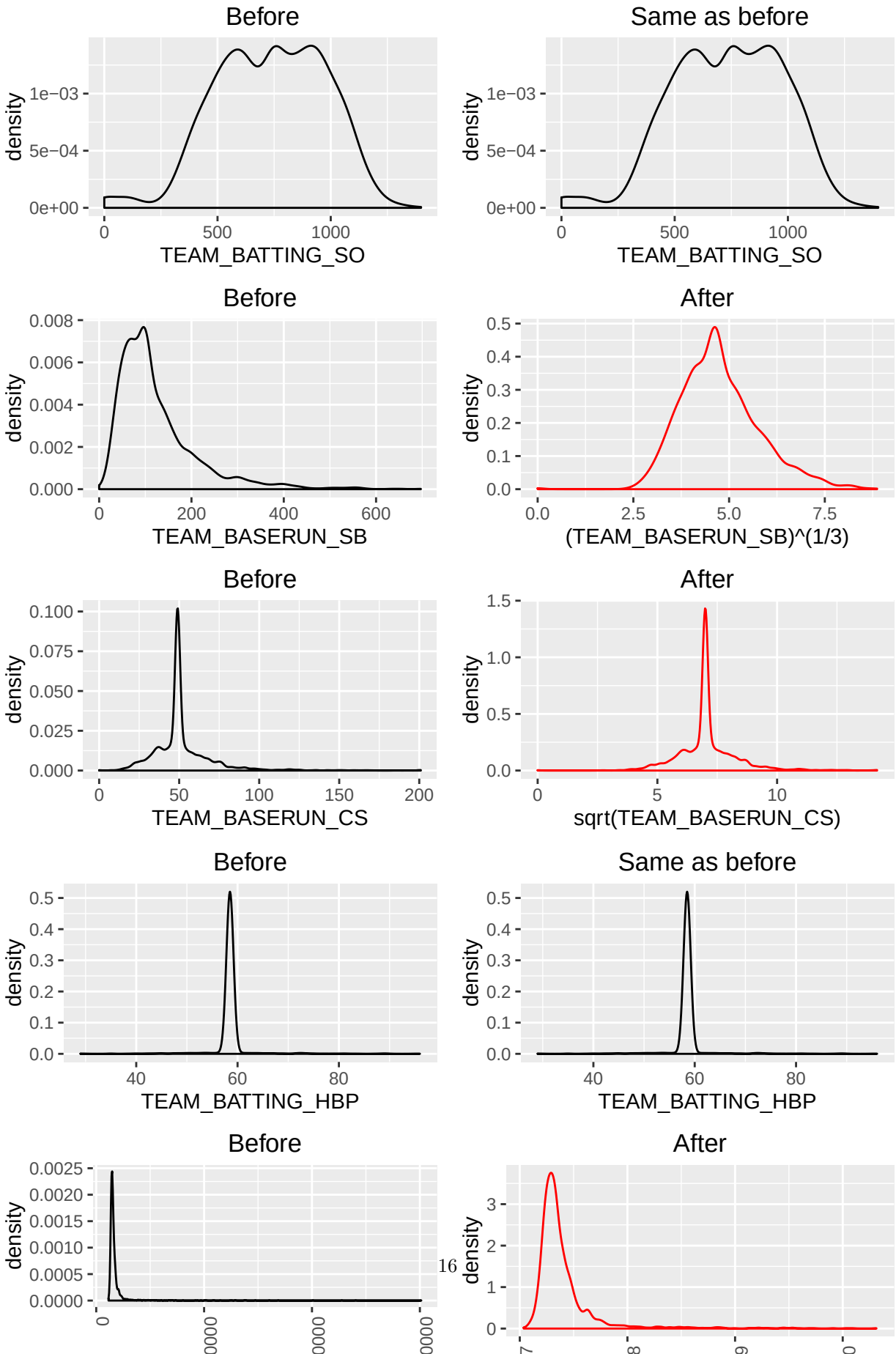
We did not test the interaction effect of the variables. We would like to test the model's performance based on variables interaction. We would also like to work further on predicting the TARGET_WINS using non-parametric methods (such as KNN, Random Forests etc). Given that the data is highly distorted, we believe non-parametric methods might perform better than the parametric methods. The Q-Q plots of the models show that the models might perform poorly at the extreme data points. Random forests would efficiently deal with the variation in data.

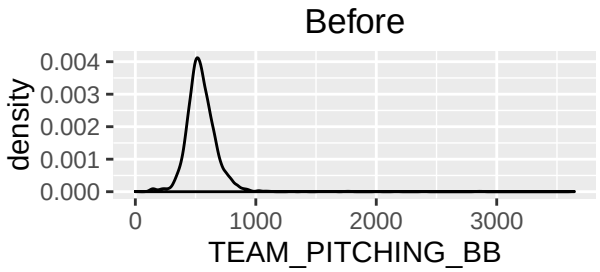
Appendix-A

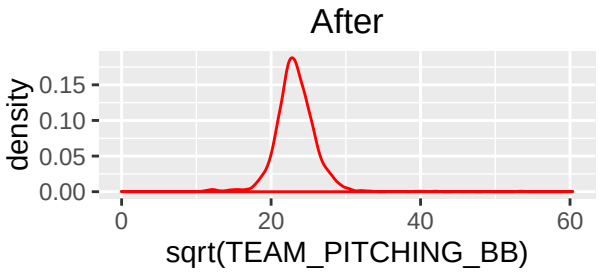
Variables and their transformations

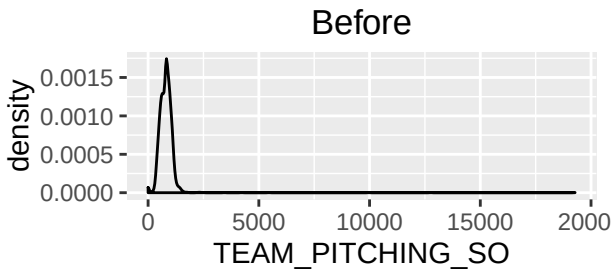
We used the ln (log_e) and square root transformations for some of our input variables. Most of these transformed variables were having right skewed data, and if the variable has non-zero value, then we used ln transformation, else we used square root transformation. For one variable (TEAM_BASERUN_SB) we used cube root transformation. All these transformations have helped to get approximate normal distribution for the variable data.

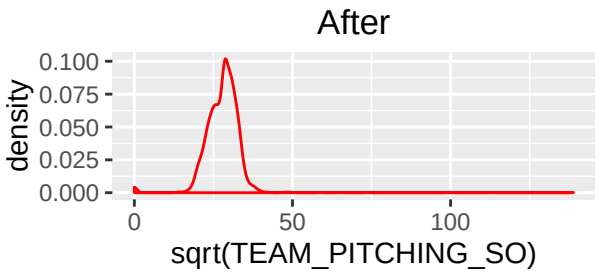
Figure A-1: Variables and their transformations (wherever applied)

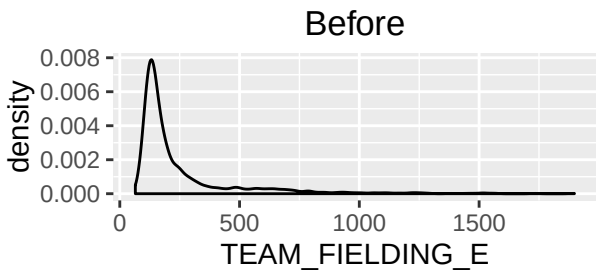


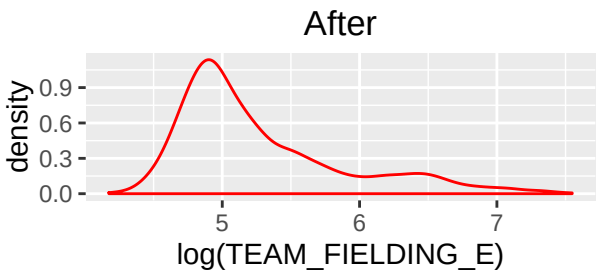

Before
density
0.004
0.003
0.002
0.001
0.000
0
1000
2000
3000
TEAM_PITCHING_BB

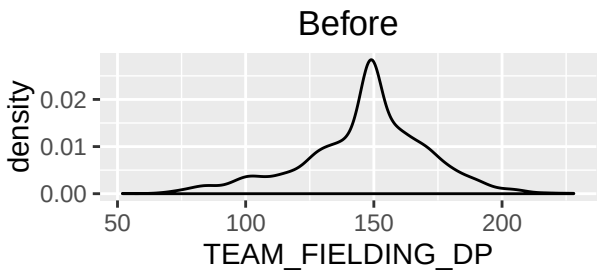

After
density
0.15
0.10
0.05
0.00
0
20
40
60
sqrt(TEAM_PITCHING_BB)

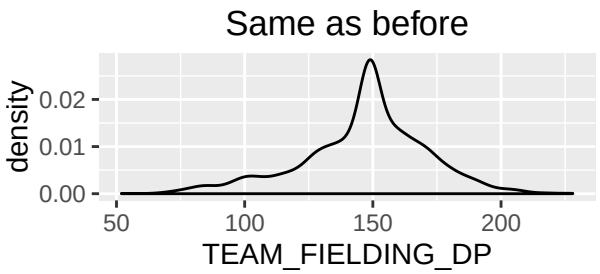

Before
density
0.0015
0.0010
0.0005
0.0000
0
5000
10000
15000
2000
TEAM_PITCHING_SO


After
density
0.100
0.075
0.050
0.025
0.000
0
50
100
sqrt(TEAM_PITCHING_SO)


Before
density
0.008
0.006
0.004
0.002
0.000
0
500
1000
1500
TEAM_FIELDING_E


After
density
0.9
0.6
0.3
0.0
5
6
7
log(TEAM_FIELDING_E)


Before
density
0.02
0.01
0.00
50
100
150
200
TEAM_FIELDING_DP


Same as before
density
0.02
0.01
0.00
50
100
150
200
TEAM_FIELDING_DP

Appendix-B

The R code used for this project is given below:

```
library(knitr)
library(ggplot2)
library(reshape2)
library(gridExtra)
library(MASS)
library(boot)



train_df <- read.csv("moneyball-training-data.csv")
test_df <- read.csv("moneyball-evaluation-data.csv")


###Table 2: NA values summary in training data

Number_of_NA = c(102,131,772,2085,102,286)
df <- data.frame(Variable =
                   c('TEAM_BATTING_SO','TEAM_BASERUN_SB',
                     'TEAM_BASERUN_CS','TEAM_BATTING_HBP',
                     'TEAM_PITCHING_SO','TEAM_FIELDING_DP'),
       Number_of_NA =
         Number_of_NA, NA_Percentage =
         paste0(round((Number_of_NA/nrow(train_df) * 100),2),"%"))
kable(df)


###Figure 2: Heat map showing overall correlations

#Code taken from
#http://www.sthda.com/english/wiki/ggplot2-quick-correlation-matrix-heatmap-r-software-and-data-visuali

library(reshape2)

train_df$TEAM_BATTING_HBP[is.na(train_df$TEAM_BATTING_HBP)]
          <- median(train_df$TEAM_BATTING_HBP,na.rm=TRUE)
test_df$TEAM_BATTING_HBP[is.na(test_df$TEAM_BATTING_HBP)]
     <- median(train_df$TEAM_BATTING_HBP,na.rm=TRUE)


train_df$TEAM_BASERUN_CS[is.na(train_df$TEAM_BASERUN_CS)]
<- median(train_df$TEAM_BASERUN_CS,na.rm=TRUE)

test_df$TEAM_BASERUN_CS[is.na(test_df$TEAM_BASERUN_CS)]
<- median(train_df$TEAM_BASERUN_CS,na.rm=TRUE)

train_df$TEAM_PITCHING_SO[is.na(train_df$TEAM_PITCHING_SO)]
<- median(train_df$TEAM_PITCHING_SO,na.rm=TRUE)
test_df$TEAM_PITCHING_SO[is.na(test_df$TEAM_PITCHING_SO)]
<- median(train_df$TEAM_PITCHING_SO,na.rm=TRUE)
```

```
train_df$TEAM_BATTING_SO[is.na(train_df$TEAM_BATTING_SO)]
<- median(train_df$TEAM_BATTING_SO,na.rm=TRUE)
test_df$TEAM_BATTING_SO[is.na(test_df$TEAM_BATTING_SO)]
<- median(train_df$TEAM_BATTING_SO,na.rm=TRUE)

train_df$TEAM_BASERUN_SB[is.na(train_df$TEAM_BASERUN_SB)]
<- median(train_df$TEAM_BASERUN_SB,na.rm=TRUE)
test_df$TEAM_BASERUN_SB[is.na(test_df$TEAM_BASERUN_SB)]
<- median(train_df$TEAM_BASERUN_SB,na.rm=TRUE)

train_df$TEAM_FIELDING_DP[is.na(train_df$TEAM_FIELDING_DP)]
<- median(train_df$TEAM_FIELDING_DP,na.rm=TRUE)
test_df$TEAM_FIELDING_DP[is.na(test_df$TEAM_FIELDING_DP)]
<- median(train_df$TEAM_FIELDING_DP,na.rm=TRUE)

train_df <- train_df[,-1]

cormat <- round(cor(train_df),2)

##
# Get lower triangle of the correlation matrix
get_lower_tri<-function(cormat){
  cormat[upper.tri(cormat)] <- NA
  return(cormat)
}
# Get upper triangle of the correlation matrix
get_upper_tri <- function(cormat){
  cormat[lower.tri(cormat)]<- NA
  return(cormat)
}

upper_tri <- get_upper_tri(cormat)
#upper_tri


reorder_cormat <- function(cormat){
  # Use correlation between variables as distance
  dd <- as.dist((1-cormat)/2)
  hc <- hclust(dd)
  cormat <-cormat[hc$order, hc$order]
}

# Reorder the correlation matrix
cormat <- reorder_cormat(cormat)
upper_tri <- get_upper_tri(cormat)
# Melt the correlation matrix
melted_cormat <- melt(upper_tri, na.rm = TRUE)

# Create a ggheatmap
ggheatmap <- ggplot(melted_cormat, aes(Var2, Var1, fill = value))+
  geom_tile(color = "white")+
  scale_fill_gradient2(low = "blue", high = "red", mid = "white",
                       midpoint = 0, limit = c(-1,1), space = "Lab",
```

```
                      name="Pearson\nCorrelation") +
  theme_minimal()+ # minimal theme
  theme(axis.text.x = element_text(angle = 45, vjust = 1,
                                   size = 12, hjust = 1))+
  coord_fixed()
ggheatmap +
  geom_text(aes(Var2, Var1, label = value), color = "black", size = 2) +
  theme(
    axis.title.x = element_blank(),
    axis.title.y = element_blank(),
    panel.grid.major = element_blank(),
    panel.border = element_blank(),
    panel.background = element_blank(),
    axis.ticks = element_blank(),
    legend.justification = c(1, 0),
    legend.position = c(0.6, 0.7),
    legend.direction = "horizontal")+
  guides(fill = guide_colorbar(barwidth = 7, barheight = 1,
                               title.position = "top", title.hjust = 0.5))
```

```
###Figure 3: Density plots of variables

#names(test_df)
#names(train_df)
setwd("C:/Users/Sekhar/Documents/R Programs/Business Analytics/HW1")

train_df <- read.csv("moneyball-training-data.csv")
test_df <- read.csv("moneyball-evaluation-data.csv")

#test_df <- test_df[,-1]
#train_df <- train_df[,-1]

TARGET_WINS <- vector(length=nrow(test_df))
data_type <- rep("Test",nrow(test_df))

test_df <- data.frame(data_type,TARGET_WINS,test_df)
data_type <- rep("Train",nrow(train_df))
train_df <- data.frame(data_type,train_df)
combined_df <- rbind(train_df,test_df)
#head(combined_df)


#Drawing the density plots

g1 <- ggplot(data=train_df,aes(TARGET_WINS))+
  geom_density()

g2 <- ggplot(data=combined_df,aes(TEAM_BATTING_H))+
  geom_density()
```

```
g3 <- ggplot(data=combined_df,aes(TEAM_BATTING_2B))+
  geom_density()

g4 <- ggplot(data=combined_df,aes(TEAM_BATTING_3B))+
  geom_density()

g5 <- ggplot(data=combined_df,aes(TEAM_BATTING_HR))+
  geom_density()+
  theme(axis.text.x = element_text(angle = 90, hjust = 1))

g6 <- ggplot(data=combined_df,aes(TEAM_BATTING_BB))+
  geom_density()

#grid.arrange(g1, g2, g3,g4,g5,g6, ncol=3)


g7 <- ggplot(data=combined_df,aes(TEAM_BATTING_SO))+
  geom_density()

g8 <- ggplot(data=combined_df,aes(TEAM_BASERUN_SB))+
  geom_density()

g9 <- ggplot(data=combined_df,aes(TEAM_BASERUN_CS))+
  geom_density()


g10 <- ggplot(data=combined_df,aes(TEAM_BATTING_HBP))+
  geom_density()


g11 <- ggplot(data=combined_df,aes(TEAM_PITCHING_H))+
  geom_density()+
  theme(axis.text.x = element_text(angle = 90, hjust = 1))


g12 <- ggplot(data=combined_df,aes(TEAM_PITCHING_HR))+
  geom_density()+
  labs(title="Before")


#grid.arrange(g1,g2,g3,g4,g5,g6,g7, g8,g9,g10,g11,g12, ncol=3)


g13 <- ggplot(data=combined_df,aes(TEAM_PITCHING_BB))+
  geom_density()

g14 <- ggplot(data=combined_df,aes(TEAM_PITCHING_SO))+
  geom_density()

g15 <- ggplot(data=combined_df,aes(TEAM_FIELDING_E))+
  geom_density()

g16 <- ggplot(data=combined_df,aes(TEAM_FIELDING_DP))+
```

```
  geom_density()

#grid.arrange(g13, g14,g15,g16, ncol=2)
grid.arrange(g1,g2,g3,g4,g5,g6,g7, g8,g9,g10,g11,g12,g13,g14,g15,g16, ncol=3)

###Table 3: Suggested variable transformations summary

df <- data.frame(Variable =
                   names(train_df)[c(-1,-2)],
                 Transformation=c("None","ln","None","Square root",
                                  "Square root", "None","None",
                                  "Cubic root","Square root","None",
                                  "ln","None","Square root",
                                  "Square root","ln","None"))
kable(df)



g1 <- ggplot(data=combined_df,aes(TEAM_BATTING_SO))+
  geom_density()

g2 <- ggplot(data=combined_df,aes(TEAM_BASERUN_SB))+
  geom_density()

g3 <- ggplot(data=combined_df,aes(TEAM_BASERUN_CS))+
  geom_density()

g4 <- ggplot(data=combined_df,aes(TEAM_BATTING_HBP))+
  geom_density()

g5 <- ggplot(data=combined_df,aes(TEAM_PITCHING_SO))+
  geom_density()+
  theme(axis.text.x = element_text(angle = 90, hjust = 1))

g6 <- ggplot(data=combined_df,aes(TEAM_FIELDING_DP))+
  geom_density()

combined_df$TEAM_BATTING_HBP[is.na(combined_df$TEAM_BATTING_HBP)]
<- median(combined_df$TEAM_BATTING_HBP,na.rm=TRUE)
test_df$TEAM_BATTING_HBP[is.na(test_df$TEAM_BATTING_HBP)]
<- median(combined_df$TEAM_BATTING_HBP,na.rm=TRUE)


combined_df$TEAM_BASERUN_CS[is.na(combined_df$TEAM_BASERUN_CS)]
<- median(combined_df$TEAM_BASERUN_CS,na.rm=TRUE)
test_df$TEAM_BASERUN_CS[is.na(test_df$TEAM_BASERUN_CS)]
<- median(combined_df$TEAM_BASERUN_CS,na.rm=TRUE)

combined_df$TEAM_PITCHING_SO[is.na(combined_df$TEAM_PITCHING_SO)]
<- median(combined_df$TEAM_PITCHING_SO,na.rm=TRUE)
test_df$TEAM_PITCHING_SO[is.na(test_df$TEAM_PITCHING_SO)]
<- median(combined_df$TEAM_PITCHING_SO,na.rm=TRUE)
```

```
combined_df$TEAM_BATTING_SO[is.na(combined_df$TEAM_BATTING_SO)]
<- median(combined_df$TEAM_BATTING_SO,na.rm=TRUE)
test_df$TEAM_BATTING_SO[is.na(test_df$TEAM_BATTING_SO)]
<- median(combined_df$TEAM_BATTING_SO,na.rm=TRUE)

combined_df$TEAM_BASERUN_SB[is.na(combined_df$TEAM_BASERUN_SB)]
<- median(combined_df$TEAM_BASERUN_SB,na.rm=TRUE)
test_df$TEAM_BASERUN_SB[is.na(test_df$TEAM_BASERUN_SB)]
<- median(combined_df$TEAM_BASERUN_SB,na.rm=TRUE)

combined_df$TEAM_FIELDING_DP[is.na(combined_df$TEAM_FIELDING_DP)]
<- median(combined_df$TEAM_FIELDING_DP,na.rm=TRUE)
test_df$TEAM_FIELDING_DP[is.na(test_df$TEAM_FIELDING_DP)]
<- median(combined_df$TEAM_FIELDING_DP,na.rm=TRUE)


g1 <- g1 + geom_density(
  data=combined_df,aes(TEAM_BATTING_SO),color="red",alpha=0.5)

g2 <- g2 + geom_density(
  data=combined_df,aes(TEAM_BASERUN_SB),color="red",alpha=0.5)

g3 <- g3 + geom_density(
  data=combined_df,aes(TEAM_BASERUN_CS),color="red",alpha=0.5)

g4 <- g4 + geom_density(
  data=combined_df,aes(TEAM_BATTING_HBP),color="red",alpha=0.5)

g5 <- g5 + geom_density(

  data=combined_df,aes(TEAM_PITCHING_SO),color="red",alpha=0.5)

g6 <- g6 + geom_density(data=combined_df,
                        aes(TEAM_FIELDING_DP),color="red",alpha=0.5)

grid.arrange(g1, g2, g3,g4,g5,g6, ncol=3)

transformed_df <- combined_df

combined_df$TEAM_BATTING_H <- log(combined_df$TEAM_BATTING_H)
combined_df$TEAM_BATTING_3B <- sqrt(combined_df$TEAM_BATTING_3B)
combined_df$TEAM_BATTING_HR <- sqrt(combined_df$TEAM_BATTING_HR)
combined_df$TEAM_BASERUN_SB <- (combined_df$TEAM_BASERUN_SB)^(1/3)
combined_df$TEAM_BASERUN_CS <- sqrt(combined_df$TEAM_BASERUN_CS)
combined_df$TEAM_PITCHING_H <- log(combined_df$TEAM_PITCHING_H)
combined_df$TEAM_PITCHING_BB <- sqrt(combined_df$TEAM_PITCHING_BB)
combined_df$TEAM_PITCHING_SO <- sqrt(combined_df$TEAM_PITCHING_SO)
combined_df$TEAM_FIELDING_E <- log(combined_df$TEAM_FIELDING_E)



median_train_df <- combined_df[combined_df$data_type=="Train",]
median_test_df <- combined_df[combined_df$data_type=="Test",]
```

```
#tail(combined_df)
#dim(median_train_df)
#dim(median_test_df)
#dim(test_df)
#dim(train_df)
#names(median_train_df)
#names(median_test_df)
median_train_df <-
  median_train_df[median_train_df$TEAM_PITCHING_H <= 3000,]
median_train_df <-
  median_train_df[median_train_df$TEAM_PITCHING_SO <= 1600,]


setwd("C:/Users/Sekhar/Documents/R Programs/Business Analytics/HW1")

train_df <- read.csv("moneyball-training-data.csv")
test_df <- read.csv("moneyball-evaluation-data.csv")

#test_df <- test_df[,-1]
#train_df <- train_df[,-1]

TARGET_WINS <- vector(length=nrow(test_df))
data_type <- rep("Test",nrow(test_df))

test_df <- data.frame(data_type,TARGET_WINS,test_df)
data_type <- rep("Train",nrow(train_df))
train_df <- data.frame(data_type,train_df)
combined_df <- rbind(train_df,test_df)
#head(combined_df)

MARK_NA_TEAM_BATTING_SO <-
  vector(length=nrow(combined_df))
MARK_NA_TEAM_BATTING_SO[which(is.na(combined_df$TEAM_BATTING_SO))] <- 1
combined_df$MARK_NA_TEAM_BATTING_SO <-
  MARK_NA_TEAM_BATTING_SO


MARK_NA_TEAM_BASERUN_SB <- vector(length=nrow(combined_df))
MARK_NA_TEAM_BASERUN_SB[which(is.na(combined_df$TEAM_BASERUN_SB))] <- 1
combined_df$MARK_NA_TEAM_BASERUN_SB <-
  MARK_NA_TEAM_BASERUN_SB


MARK_NA_TEAM_BASERUN_CS <- vector(length=nrow(combined_df))
MARK_NA_TEAM_BASERUN_CS[which(is.na(combined_df$TEAM_BASERUN_CS))] <- 1
combined_df$MARK_NA_TEAM_BASERUN_CS <- MARK_NA_TEAM_BASERUN_CS


MARK_NA_TEAM_BATTING_HBP <- vector(length=nrow(combined_df))
MARK_NA_TEAM_BATTING_HBP[which(is.na(combined_df$TEAM_BATTING_HBP))] <- 1
combined_df$MARK_NA_TEAM_BATTING_HBP <- MARK_NA_TEAM_BATTING_HBP
```

```
MARK_NA_TEAM_PITCHING_SO <- vector(length=nrow(combined_df))
MARK_NA_TEAM_PITCHING_SO[which(is.na(combined_df$TEAM_PITCHING_SO))] <- 1
combined_df$MARK_NA_TEAM_PITCHING_SO <- MARK_NA_TEAM_PITCHING_SO


MARK_NA_TEAM_FIELDING_DP <- vector(length=nrow(combined_df))
MARK_NA_TEAM_FIELDING_DP[which(is.na(combined_df$TEAM_FIELDING_DP))] <- 1
combined_df$MARK_NA_TEAM_FIELDING_DP <- MARK_NA_TEAM_FIELDING_DP

combined_df$TEAM_BATTING_SO[which(is.na(combined_df$TEAM_BATTING_SO))] <- 1
combined_df$TEAM_BASERUN_SB[which(is.na(combined_df$TEAM_BASERUN_SB))] <- 1
combined_df$TEAM_BASERUN_CS[which(is.na(combined_df$TEAM_BASERUN_CS))] <- 1
combined_df$TEAM_BATTING_HBP[which(is.na(combined_df$TEAM_BATTING_HBP))] <- 1
combined_df$TEAM_PITCHING_SO[which(is.na(combined_df$TEAM_PITCHING_SO))] <- 1
combined_df$TEAM_FIELDING_DP[which(is.na(combined_df$TEAM_FIELDING_DP))] <- 1




combined_df$TEAM_BATTING_H <- log(combined_df$TEAM_BATTING_H)
combined_df$TEAM_BATTING_3B <- sqrt(combined_df$TEAM_BATTING_3B)
combined_df$TEAM_BATTING_HR <- sqrt(combined_df$TEAM_BATTING_HR)
combined_df$TEAM_BASERUN_SB <- (combined_df$TEAM_BASERUN_SB)^(1/3)
combined_df$TEAM_BASERUN_CS <- sqrt(combined_df$TEAM_BASERUN_CS)
combined_df$TEAM_PITCHING_H <- log(combined_df$TEAM_PITCHING_H)
combined_df$TEAM_PITCHING_BB <- sqrt(combined_df$TEAM_PITCHING_BB)
combined_df$TEAM_PITCHING_SO <- sqrt(combined_df$TEAM_PITCHING_SO)
combined_df$TEAM_FIELDING_E <- log(combined_df$TEAM_FIELDING_E)

mark_na_train_df <- combined_df[combined_df$data_type=="Train",]
mark_na_test_df <- combined_df[combined_df$data_type=="Test",]
#head(mark_na_test_df)
#dim(mark_na_test_df)
mark_na_train_df <- mark_na_train_df[mark_na_train_df$TEAM_PITCHING_H <= 3000,]
mark_na_train_df <- mark_na_train_df[mark_na_train_df$TEAM_PITCHING_SO <= 1600,]

display_df <-
  data.frame(Quantity=c("Adj. R-Squared", "F-statistic",
                        "F-Statistic's p-Value","Residual Standard Error",
                        "5-fold cross validation error"),
                        Value=c(0.3157,70.98,2.2e-16,13.03,173.8388))
kable(display_df)



###Table 5: Model-0 coefficients p-values and standard errors


display_df <- data.frame(
Variable=c(
"(Intercept)",
"TEAM_BATTING_H",
"TEAM_BATTING_2B",
```

```
"TEAM_BATTING_3B",
"TEAM_BATTING_HR",
"TEAM_BATTING_BB",
"TEAM_BATTING_SO",
"TEAM_BASERUN_SB",
"TEAM_BASERUN_CS",
"TEAM_BATTING_HBP",
"TEAM_PITCHING_H",
"TEAM_PITCHING_HR",
"TEAM_PITCHING_BB",
"TEAM_PITCHING_SO",
"TEAM_FIELDING_E",
"TEAM_FIELDING_DP"
), Coefficient=c(
  "-393",
"68.56",
"-0.02602",
"1.485",
"-0.1673",
"0.03631",
"-0.01754",
"3.167",
"-0.6282",
"0.07934",
"3.653",
"0.0514",
"-0.9378",
"0.5379",
"-10.99",
"-0.1164"
  ),
Std_Error=c(
  "39.83",
"6.182",
"0.00928",
"0.27",
"0.4729",
"0.007173",
"0.003531",
"0.3913",
"0.2619",
"0.07303",
"2.595",
"0.02212",
"0.2575",
"0.1124",
"1.097",
"0.0134"
),
p_value=c("0.0000000000000002",
"0.0000000000000002",
"0.005085",
"0.0000000418",
```

```
"0.723608",
"0.000000449",
"0.000000728",
"0.000000000000000931",
"0.016516",
"0.277424",
"0.159392",
"0.020218",
"0.000277",
"0.00000183",
"0.0000000000000002",
"0.0000000000000002"

)
)

kable(display_df)

#Cross validation of Model 1
df <- median_train_df[,c(-1,-2)]
set.seed(1)
#glm.fit <- glm(data=df,TARGET_WINS~.)
glm.fit = glm(data=df,TARGET_WINS ~
                TEAM_BATTING_H + TEAM_BA
              TTING_2B + TEAM_BATTING_3B +
    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)

#cv.glm(df,glm.fit,K=5)$delta[1]
#names(df)


display_df <- data.frame(Quantity=c("Adj. R-Squared", "F-statistic",
                                    "F-Statistic's p-Value",
                                    "Residual Standard Error",
                                    "5-fold cross validation error"),
                         Value=c(0.3154,88.35,2.2e-16,13.03,172.1424))
kable(display_df)

###Table 7: Model-1's p-values, std.
##error and coefficients. Model-1 is
##obtained by eliminating the unnecessary variables from Model-0


df <- median_train_df[,c(-1,-2)]
#set.seed(1)
#lm.fit1 <- lm(data=df,TARGET_WINS~.)

set.seed(1)
lm.fit1 <- lm(data=df,TARGET_WINS ~
```

```
                TEAM_BATTING_H + TEAM_BATTING_2B + TEAM_BATTING_3B +
    TEAM_BATTING_BB + TEAM_BATTING_SO +
      TEAM_BASERUN_SB + TEAM_BASERUN_CS +
    #TEAM_PITCHING_H +
      TEAM_PITCHING_HR + TEAM_PITCHING_BB +
      TEAM_PITCHING_SO +
    TEAM_FIELDING_E + TEAM_FIELDING_DP
)
#summary(lm.fit1)


display_df <- data.frame(
Variable=c(
"(Intercept)",
"TEAM_BATTING_H",
"TEAM_BATTING_2B",
"TEAM_BATTING_3B",
"TEAM_BATTING_BB",
"TEAM_BATTING_SO",
"TEAM_BASERUN_SB",
"TEAM_BASERUN_CS",
"TEAM_PITCHING_HR",
"TEAM_PITCHING_BB",
"TEAM_PITCHING_SO",
"TEAM_FIELDING_E",
"TEAM_FIELDING_DP"
),
Coefficient=c(
  "-388.4",
"71.46",
"-0.02458",
"1.36",
"0.02818",
"-0.01794",
"3.085",
"-0.644",
"0.04743",
"-0.669",
"0.5143",
"-10.06",
"-0.1183"
  ),
Std_error=c(
  "39.66",
"5.638",
"0.009237",
"0.2553",
"0.005298",
"0.003511",
"0.3819",
"0.2618",
"0.008873",
"0.2032",
```

```
"0.1117",
"0.9623",
"0.013"),
p_value=c("0.0000000000000002",
"0.0000000000000002",
"0.00784",
"0.00000011",
"0.000000115",
"0.000000348",
"0.00000000000000106",
"0.01396",
"0.0000000992",
"0.00101",
"0.00000435",
"0.0000000000000002",
"0.0000000000000002"
)
)
kable(display_df)

###Figure 6: Residual plots of Model-1

par(mfrow=c(2,2))
plot(lm.fit1)

###Table 8: Model-2 statistics

display_df <- data.frame(Quantity=c("Adj. R-Squared",
                                    "F-statistic", "F-Statistic's p-Value",
                                    "Residual Standard Error",
                                    "5-fold cross validation error"),
                         Value=c(0.3974,116.4,2.2e-16,12.23,152.53))
kable(display_df)




#names(mark_na_train_df)

df <- mark_na_train_df[,c(-1,-2)]
#dim(df)
#names(df)
#dim(mark_na_train_df)

lm.fit2 <- lm(data=df,TARGET_WINS~TEAM_BATTING_H
+TEAM_BATTING_2B
+TEAM_BATTING_3B
+TEAM_BATTING_HR
+TEAM_BATTING_BB
+TEAM_BATTING_SO*MARK_NA_TEAM_BATTING_SO
+TEAM_BASERUN_SB*MARK_NA_TEAM_BASERUN_SB
+TEAM_BASERUN_CS*MARK_NA_TEAM_BASERUN_CS
+TEAM_BATTING_HBP*MARK_NA_TEAM_BATTING_HBP
```

```
+TEAM_PITCHING_H
+TEAM_PITCHING_HR
+TEAM_PITCHING_BB
+TEAM_PITCHING_SO*MARK_NA_TEAM_PITCHING_SO
+TEAM_FIELDING_E
+TEAM_FIELDING_DP*MARK_NA_TEAM_FIELDING_DP)

#summary(lm.fit2)



library(MASS)
#step <- stepAIC(lm.fit2)
#step$anova
#anova(lm.fit2)

lm.fit2 <- lm(data=df,TARGET_WINS ~ TEAM_BATTING_H +
                TEAM_BATTING_2B + TEAM_BATTING_3B +
    #TEAM_BATTING_BB +
      #TEAM_BATTING_SO +
      TEAM_BASERUN_SB + MARK_NA_TEAM_BASERUN_SB +
    TEAM_BASERUN_CS +
      #MARK_NA_TEAM_BASERUN_CS +
      #TEAM_BATTING_HBP +
    MARK_NA_TEAM_BATTING_HBP + TEAM_PITCHING_HR +
      TEAM_PITCHING_BB +
    TEAM_PITCHING_SO + TEAM_FIELDING_E + TEAM_FIELDING_DP +
      MARK_NA_TEAM_FIELDING_DP)
#summary(lm.fit2)

#mean((df$TARGET_WINS-predict(lm.fit3))^2)
#predict(lm.fit2,mark_na_test_df)

#dim(df)
#names(df)

#predict(lm.fit3,df1[-1,])
set.seed(1)
glm.fit = glm(data=df,TARGET_WINS ~ TEAM_BATTING_H +
                TEAM_BATTING_2B + TEAM_BATTING_3B +
      TEAM_BASERUN_SB + MARK_NA_TEAM_BASERUN_SB +
    TEAM_BASERUN_CS +
    MARK_NA_TEAM_BATTING_HBP + TEAM_PITCHING_HR +
      TEAM_PITCHING_BB +
    TEAM_PITCHING_SO + TEAM_FIELDING_E + TEAM_FIELDING_DP +
      MARK_NA_TEAM_FIELDING_DP)


#cv.glm(df,glm.fit,K=5)$delta[1]

display_df <- data.frame(Variable=
c("(Intercept)",
"TEAM_BATTING_H",
```

```
"TEAM_BATTING_2B",
"TEAM_BATTING_3B",
"TEAM_BASERUN_SB",
"MARK_NA_TEAM_BASERUN_SB",
"TEAM_BASERUN_CS",
"MARK_NA_TEAM_BATTING_HBP",
"TEAM_PITCHING_HR",
"TEAM_PITCHING_BB",
"TEAM_PITCHING_SO",
"TEAM_FIELDING_E",
"TEAM_FIELDING_DP",
"MARK_NA_TEAM_FIELDING_DP"
),
Coefficients=c(
  "-356.5",
"73.64",
"-0.02576",
"1.764",
"3.523",
"40.25",
"-0.5665",
"6.811",
"0.03137",
"0.8118",
"-0.3175",
"-24",
"-0.1079",
"-9.159"
  ),
Std_error=c(
  "31.32",
"4.681",
"0.008983",
"0.2378",
"0.3425",
"2.275",
"0.112",
"1.084",
"0.007372",
"0.1029",
"0.0441",
"1.115",
"0.01372",
"1.9"

  ),
p_value=c("0.0000000000000002",
"0.0000000000000002",
"0.00418",
"0.000000000000168",
"0.0000000000000002",
"0.0000000000000002",
"0.000000453",
```

```
"0.00000000039",
"0.0000217",
"0.00000000000000466",
"0.000000000000812",
"0.0000000000000002",
"0.00000000000000567",
"0.00000152"))

kable(display_df)



###Figure 7: Model-2 (lm.fit2) residual plot

par(mfrow=c(2,2))
plot(lm.fit2)

combined_df <- transformed_df
g7 <- ggplot(data=combined_df,aes(TEAM_BATTING_SO))+
  geom_density()+
  labs(title="Before")

g7b <- ggplot(data=combined_df,aes(TEAM_BATTING_SO))+
  geom_density()+
  labs(title="Same as before")


g8 <- ggplot(data=combined_df,aes(TEAM_BASERUN_SB))+
  geom_density()+
  labs(title="Before")

g8b <- ggplot(data=combined_df,aes((TEAM_BASERUN_SB)^(1/3)))+
  geom_density(col="red")+
  labs(title="After")

g9 <- ggplot(data=combined_df,aes(TEAM_BASERUN_CS))+
  geom_density()+
  labs(title="Before")

g9b <- ggplot(data=combined_df,aes(sqrt(TEAM_BASERUN_CS)))+
  geom_density(col="red")+
  labs(title="After")
grid.arrange(g7, g7b, g8,g8b,g9,g9b, ncol=2)

g10 <- ggplot(data=combined_df,aes(TEAM_BATTING_HBP))+
  geom_density()+
  labs(title="Before")

g10b <- ggplot(data=combined_df,aes(TEAM_BATTING_HBP))+
  geom_density()+
  labs(title="Same as before")

g11 <- ggplot(data=combined_df,aes(TEAM_PITCHING_H))+
```

```
  geom_density()+
  theme(axis.text.x = element_text(angle = 90, hjust = 1))+
  labs(title="Before")


g11b <- ggplot(data=combined_df,aes((log(TEAM_PITCHING_H))))+
  geom_density(col="red")+
  theme(axis.text.x = element_text(angle = 90, hjust = 1))+
  labs(title="After")



g12 <- ggplot(data=combined_df,aes(TEAM_PITCHING_HR))+
  geom_density()+
  labs(title="Before")


g12b <- ggplot(data=combined_df,aes((TEAM_PITCHING_HR)))+
  geom_density()+
  labs(title="Same as before")

grid.arrange(g10, g10b, g11,g11b,g12,g12b, ncol=2)


g13 <- ggplot(data=combined_df,aes(TEAM_PITCHING_BB))+
  geom_density()+
  labs(title="Before")

g13b <- ggplot(data=combined_df,aes(sqrt(TEAM_PITCHING_BB)))+
  geom_density(col="red")+
  labs(title="After")

g14 <- ggplot(data=combined_df,aes(TEAM_PITCHING_SO))+
  geom_density()+
  labs(title="Before")


g14b <- ggplot(data=combined_df,aes(sqrt(TEAM_PITCHING_SO)))+
  geom_density(col="red")+
  labs(title="After")


g15 <- ggplot(data=combined_df,aes(TEAM_FIELDING_E))+
  geom_density()+
  labs(title="Before")


g15b <- ggplot(data=combined_df,aes(log(TEAM_FIELDING_E)))+
  geom_density(col="red")+
  labs(title="After")
```

```
g16 <- ggplot(data=combined_df,aes(TEAM_FIELDING_DP))+
  geom_density()+
  labs(title="Before")

g16b <- ggplot(data=combined_df,aes(TEAM_FIELDING_DP))+
  geom_density()+
  labs(title="Same as before")

grid.arrange(g13, g13b, g14,g14b,g15,g15b,g16,g16b, ncol=2)
```

To prepare the final submission file

- Set current working directory to the location where you downloaded the test data file.
- Execute the code given below
- Verify the final_sub file. Compare the INDEX column with MY_INDX values. If you find the same values in both the columns, then, copy the TARGET_WINS column from this file, and paste it into the evaluation data set, and save the file as predictions.csv
- Submit the predictions.csv file as project deliverable

```
## ::include statistical code here::

## code for predictions
test_df <- read.csv("moneyball-evaluation-data.csv")

df <- data.frame(TARGET_WINS=round(predict(lm.fit2,mark_na_test_df)),
                 MY_INDX = mark_na_test_df$INDEX)

df <- cbind(test_df,df)

write.csv(df,file="final_sub.csv")
```

References

1. Gareth James, Daniela Witten, Trevor Hastie, Robert Tibshirani - An Introduction to Statistical Learning with Applications in R, Sprienger 2013
2. The heat map was generated using the code from http://www.sthda.com/english/wiki/ggplot2-quick-correlation-matrix-heatmap-r-software-and-data-visualization