Predictive Model for Baseball Victories

Setu Madhavi Namburu

1/19/2019

BINGO BONUS ATTEMPTED

1. Use of Mice package to impute data and visualize - 20
2. Use of Decision trees for variable selection - 20
3. Use of glm on champion model - 10

1 INTRODUCTION

One of the major baseball league's team leader is interested to build strategies for his team to improve their probability of winning. In order to understand the performance characteristics of winning teams quantitatively rather than qualitatively, he hired a data scientist to build predictive model using historical performance records of professional teams. This data set contains approximately 2200 records. Each record represents a professional baseball team from the years 1871 to 2006 inclusive. Each record has the performance of the team for the given year, with all of the statistics adjusted to match the performance of a 162 game season. The team names and years are not provided in the data set for ananymous purposes. The data scientist worked on this project has no knowledge about baseball and hence decided to use theoretical knowledge of the variables (including grouping the variables, creation of derived variables etc) shared by league leader. The model is built purely based on insights observed from the data and warrants a detailed review from the leader to derive practical sense before putting it into application.

The data scientist followed below steps to build and validate the models (typical CRISP-DM process):

- Business problem understanding
  - Clarify the purpose and intension of the model, ask questions about available data and gather theoretical knowledge
- Data understanding
  - Gather and examine the data and available variables
  - Perform data quality checks
  - Conduct exploratary data analysis
  - Note down the insights observed
- Data Preparation
  - Address outliers and missing values
  - Create transformed variables based on EDA or based on knowledge gathered from subject matter experts
- Model Building
  - Experiment with various models
  - Calculate various performance metrics including predictive metrics on hold out data set
  - Perform model diagnostic checks
  - Iterate with EDA to add or modify variables until feasible model is derived
- Model Evaluation
  - Perform cross-validation
  - Run the model in proof of concept mode to make practical sense from experts
  - Iterate with above steps if practical significance is not met
- Model Deployment
  - Prepare data pipelining for new data
  - Use the predictive model built above to score the new data

2 DATA EXPLORATION

The dataset consists of 2276 observations with 17 variables. The first variable is index variable which can be used as a unique identifier for each row.

2.1 Data description

The variables can be further grouped into 3 subsets - Batting, pitching and fielding metrics based on how the game is played. Table 1 shows all the variable names and their theoretical effect on the win variable, TARGET_WINS.

Table 1: Variable names and descriptions

VAR_NAME	DEFINITION	THEORETICAL_EFFECT
INDEX	Identification Variable (do not use)	None
TARGET_WINS	NA	NA
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

2.2 Descriptive statistics

Table 2 presents the descriptive statistics of all the variables after excluding INDEX variable. The ranges for each variable (min,max) and percentage of valid observations (Pct.Valid) are noteworthy to get some idea about potential data quality issues in the data. The variable TEAM_BATTING_HBP is deleted from further analysis due to more than 90% missingness.

Table 2: Descriptive statistics

	Mean	Std.Dev	Min	Median	Max	N.Valid	Pct.Valid
TARGET_WINS	80.79	15.75	0	82.0	146	2276	100.00
TEAM_BATTING_H	1469.27	144.59	891	1454.0	2554	2276	100.00
TEAM_BATTING_2B	241.25	46.80	69	238.0	458	2276	100.00
TEAM_BATTING_3B	55.25	27.94	0	47.0	223	2276	100.00
TEAM_BATTING_HR	99.61	60.55	0	102.0	264	2276	100.00
TEAM_BATTING_BB	501.56	122.67	0	512.0	878	2276	100.00
TEAM_BATTING_SO	735.61	248.53	0	750.0	1399	2174	95.52
TEAM_BASERUN_SB	124.76	87.79	0	101.0	697	2145	94.24
TEAM_BASERUN_CS	52.80	22.96	0	49.0	201	1504	66.08
TEAM_BATTING_HBP	59.36	12.97	29	58.0	95	191	8.39
TEAM_PITCHING_H	1779.21	1406.84	1137	1518.0	30132	2276	100.00
TEAM_PITCHING_HR	105.70	61.30	0	107.0	343	2276	100.00
TEAM_PITCHING_BB	553.01	166.36	0	536.5	3645	2276	100.00
TEAM_PITCHING_SO	817.73	553.09	0	813.5	19278	2174	95.52
TEAM_FIELDING_E	246.48	227.77	65	159.0	1898	2276	100.00
TEAM_FIELDING_DP	146.39	26.23	52	149.0	228	1990	87.43

2.3 Exploratory data analysis

This section showcases exploratory data analysis conducted on the dataset where visual plots are examined for insights. The target variable of interest (TARGET_WINS) in this model is visualized using normality plots as shown in figure1. The box plot and qq plot clearly show some outliers at both the tail ends. While transformations may help with violations in normality for building robust models, it is hard to interpret their practical significance and make inferences. It is possible to restrict the range of the variables to work with typical ranges observed from these variables in baseball games, but that may have other effects on the model which are out of scope here.

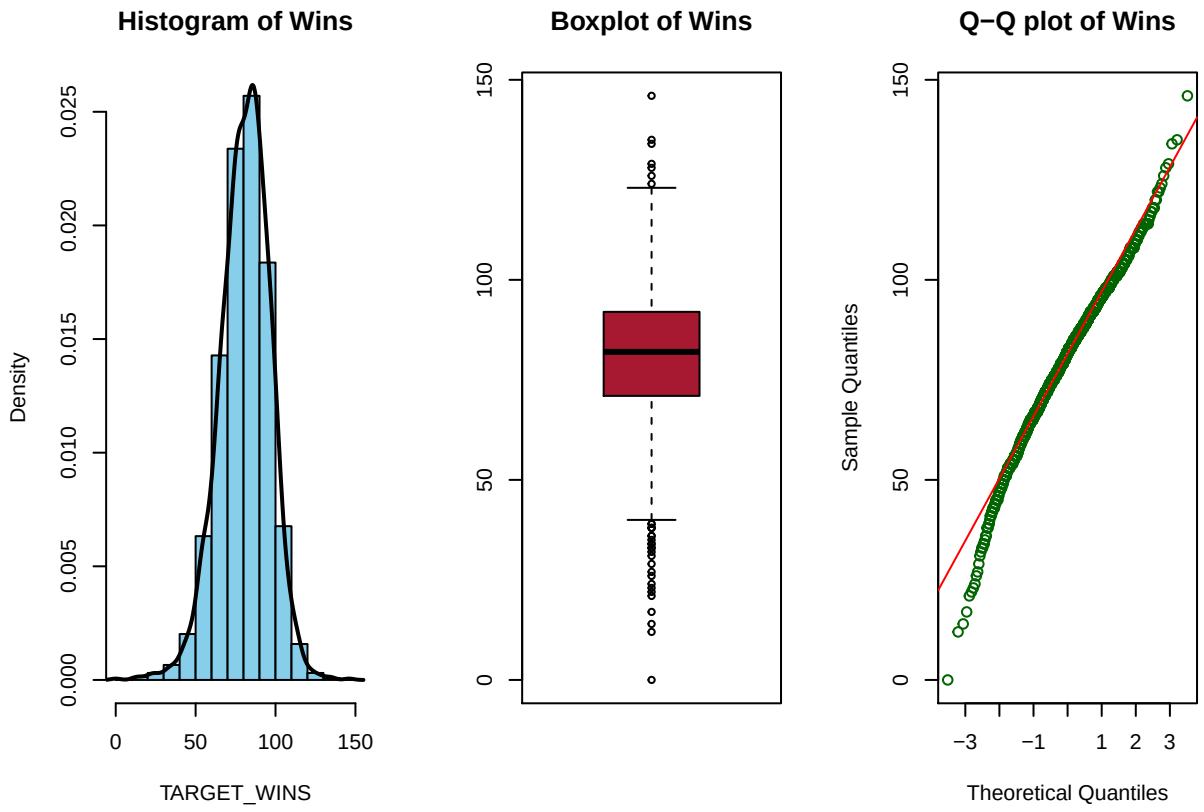

Figure 1: Normality plots of target variable

Figure 2-4 showcase pairs plots of batting, pitching, baserun and fielding variables. The scatter plots show potential multicollinearity between variables, nonlinearities and the density plots show bi-modalities and non-normal behaviors in each of the variables warranting careful feature engineering before proceeding with modeling step. There is also no clear linear relationship between target variable and any of the predictor variables observed from these plots which may indicate difficulty in capturing variability in the data.

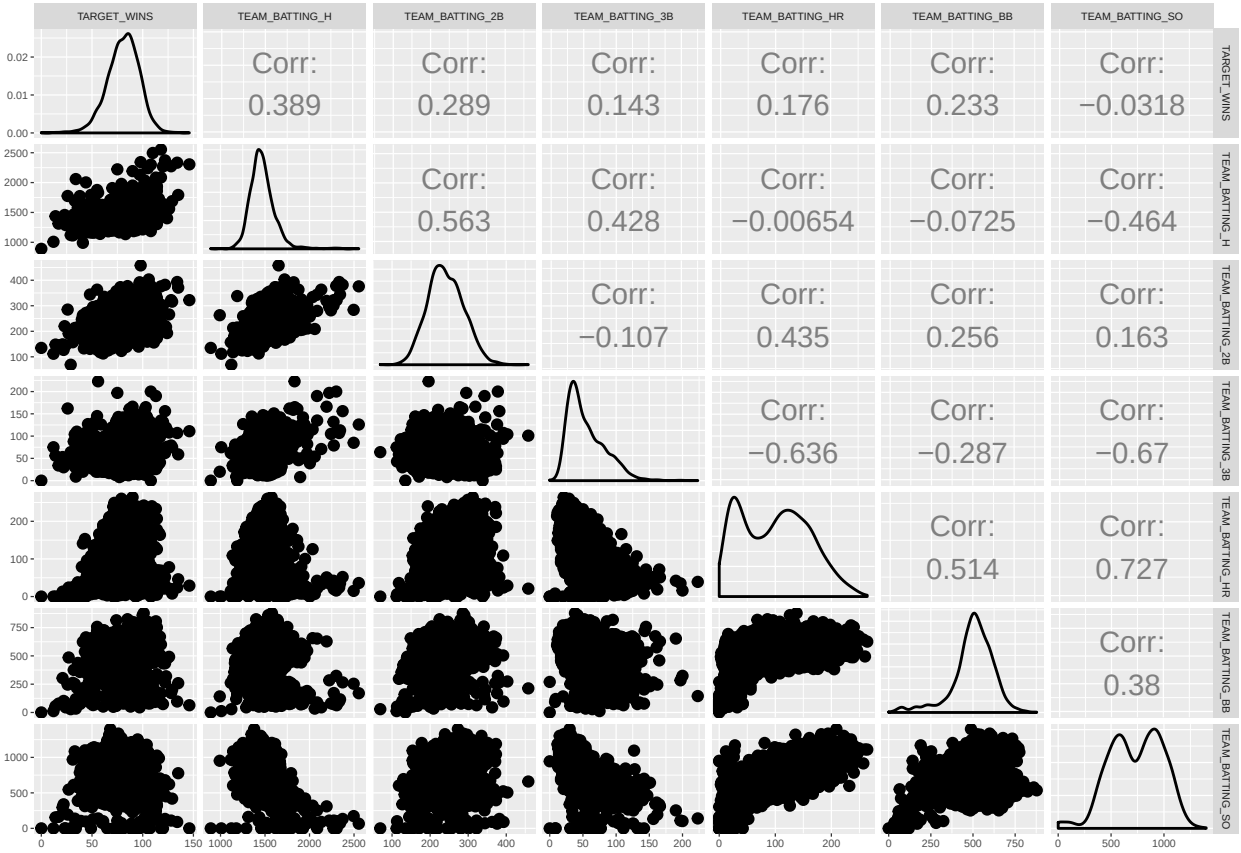

Figure 2: Pairs plot of batting stats and wins

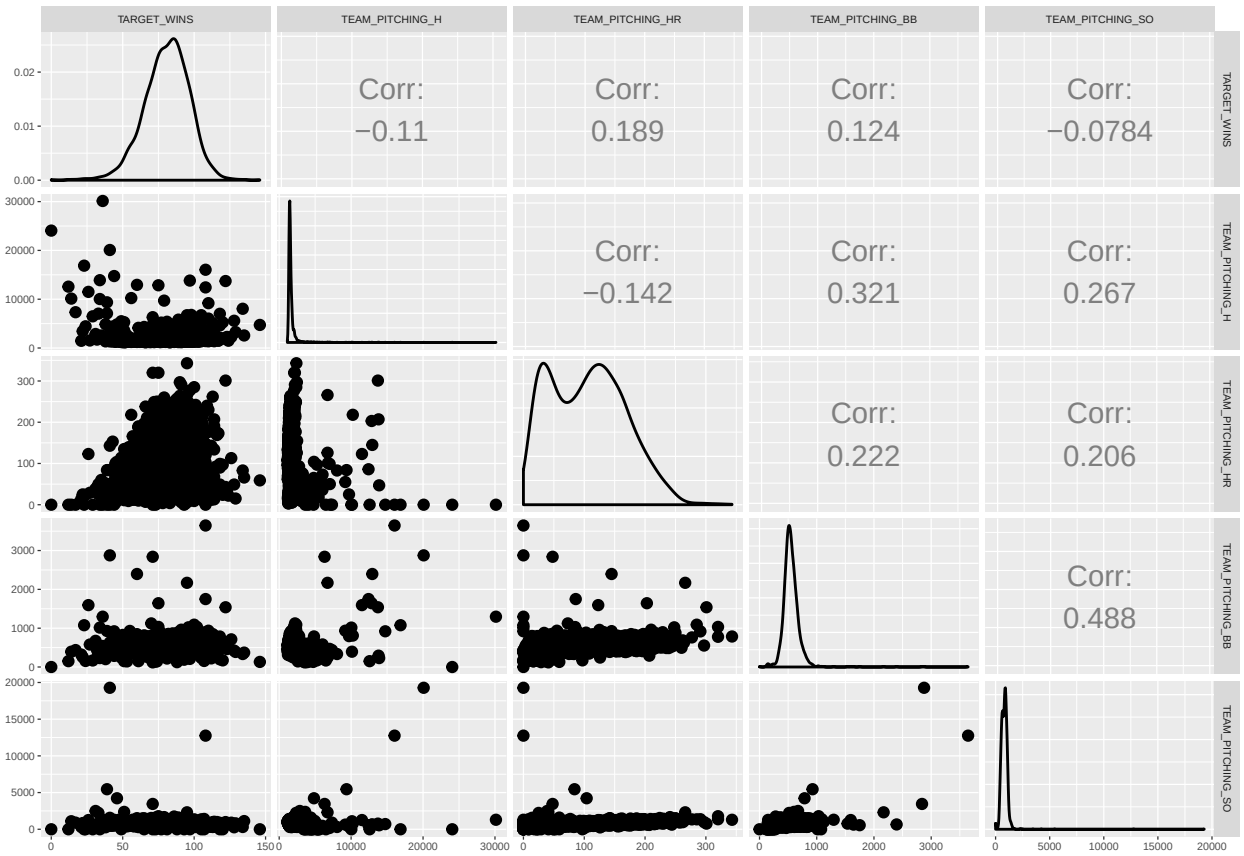

Figure 3: Pairs plot of pitcher stats and wins

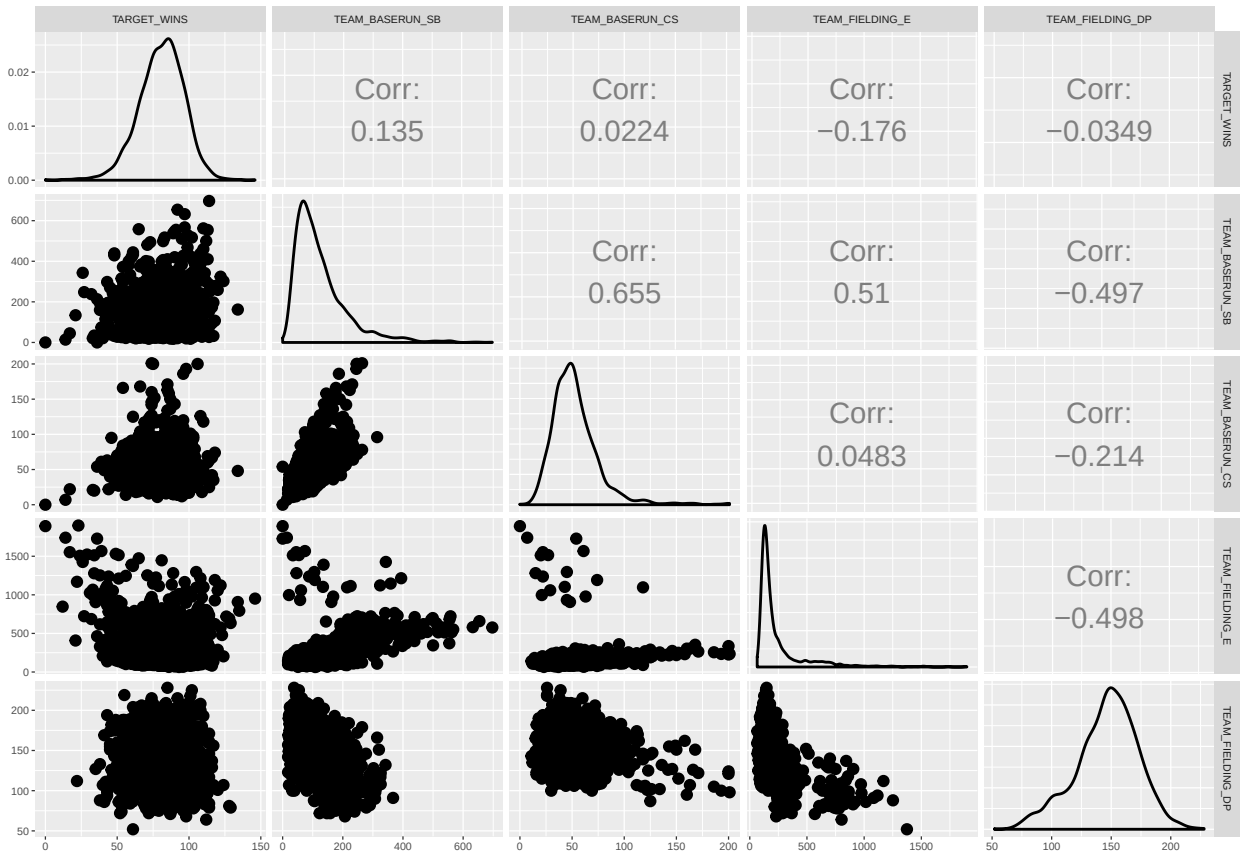

Figure 4: Pairs plot of baserun, fielding and win stats

While the scatter plots and density plots are helpful to examine the variables in detail, correlation matrix is another way to quickly capture relationship between among all the variables. Figure 5 shows correlation plot of the variables where the darker numbers (positive or negative) indicate a strong correlation and lighter indicate weak correlation. None of the predictor variables seem to be correlated with wins while some strong correlations are observed among themselves - as mentioned earlier these may lead to multi-collinear issues.

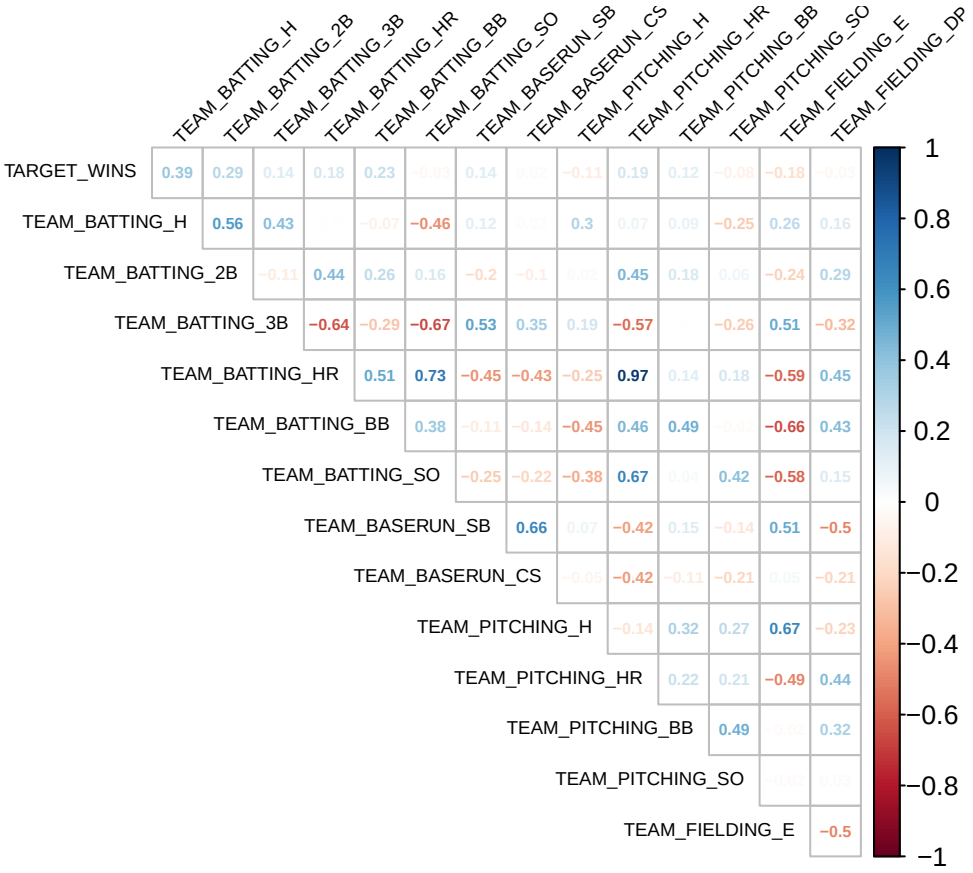

Figure 5: Correlation plot of the variables

2.4 Data quality checks

Data quality is the most important component of any data analysis project and need to be addressed before making major haul into modeling activity.

2.4.1 Missing values

In figure 6, missing values are shown in two dimensions (red indicating missing, blue complete observations respectvely). The rows indicate 65% of the observations are complete (blue), 15.33% of the observations missing TEAM_BASERUN_CS alone etc. The columns indicate, the % of missingness in each variable where TEAM_BASERUN_CS has the highest % of missing values. There is no particular pattern observed as time data is not available. The missing values need to be imputed as it is not advisable to throw away the observations.

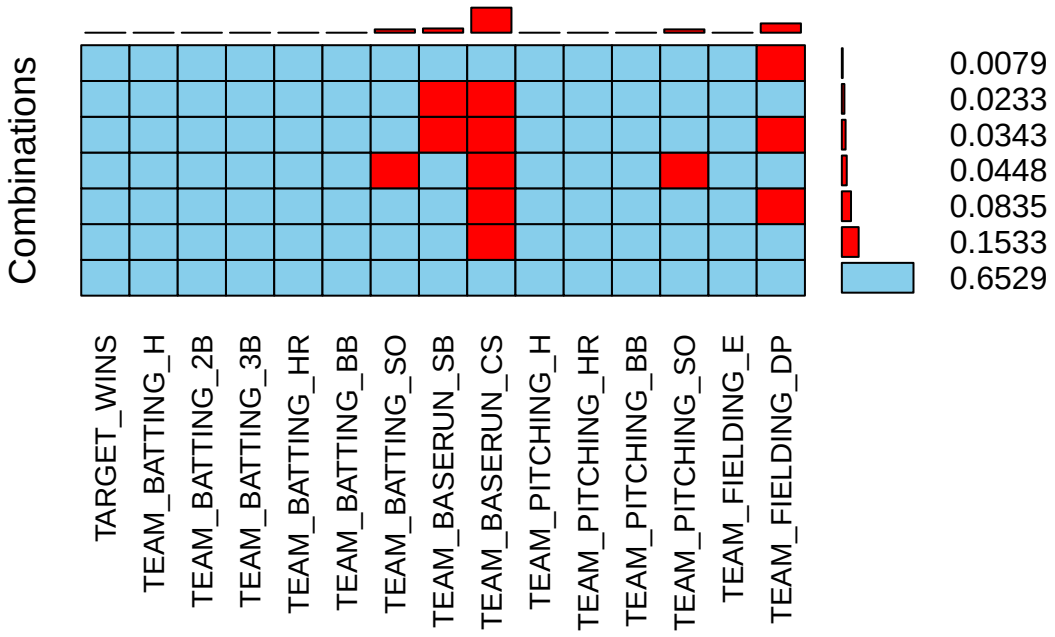

Figure 6: Pattern plot of missing observations

2.4.2 Outliers

Addressing outliers can be handled by winsorizing the variables wherein top and bottom tail values are replaced with respective quantile values to restrict the variables to typical ranges. Below table 3 presents the spread of the variables in the form of quantiles. Clearly there are huge outliers in some variables, e.g., TEAM_PITCHING_H has 99% value of 7054 whereas 99.5% value shows 12516. Due to lack of knowledge in the subject, it is decided not to throw away the outlier observations.

Table 3: Different quantile values of the variables

vars	0.5%	1%	10%	50%	90%	99%	99.5%
TARGET_WINS	31.375	38.75	61.0	82.0	99.5	114.00	119.250
TEAM_BATTING_H	1166.000	1193.25	1315.0	1454.0	1635.5	1945.50	2169.500
TEAM_BATTING_2B	133.000	141.75	182.0	238.0	303.0	351.25	371.000
TEAM_BATTING_3B	14.000	17.00	27.0	47.0	96.0	133.25	146.250
TEAM_BATTING_HR	0.000	4.75	20.0	102.0	179.5	235.00	239.625
TEAM_BATTING_BB	60.875	79.00	363.5	512.0	635.0	752.75	770.000
TEAM_BATTING_SO	0.000	70.65	421.0	750.0	1049.0	1192.27	1239.810
TEAM_BASERUN_SB	21.000	23.44	44.0	101.0	231.0	438.56	520.360
TEAM_BASERUN_CS	15.000	16.03	30.0	49.0	77.0	142.97	164.455
TEAM_PITCHING_H	1224.375	1244.00	1356.0	1518.0	2057.5	7054.00	12516.250
TEAM_PITCHING_HR	0.000	8.00	25.0	107.0	187.0	244.00	258.875
TEAM_PITCHING_BB	160.500	240.00	417.5	536.5	693.5	921.00	1084.750
TEAM_PITCHING_SO	0.000	207.19	490.0	813.5	1095.0	1466.70	1669.800
TEAM_FIELDING_E	81.375	86.00	109.0	159.0	542.0	1228.00	1463.250
TEAM_FIELDING_DP	75.000	79.00	109.0	149.0	178.0	202.22	208.000

3 DATA PREPARATION

Following steps are used in data preparation:

- Addressing outliers
  - The variables are winsorized to restrict the ranges to 1-99% values, the quantile values are stored for applying on new data.

- Imputing missing values
  - To keep it simple, the missing values from five variables with missing values shown in figure 6, are replaced by their respective means and the values are stored for applying on new data later. Figure 7 shows the densities of variables before and after imputation. It introduced spikes in two of the variables which makes sense as they have high % of missing values. This may warrant a different scheme for imputation which may be complex in nature.
- Flag variables
  - For all the predictor variables, flags are created to indiacte where the missing values and outliers are altered as 1, 0 otherwise
- Creating transformed variables Using theoretical knowledge from leader, following 7 new variables are created
  - TEAM_BATTING_1B.TREATED: TEAM_BATTING_H.TREATED - TEAM_BATTING_HR.TREATED - TEAM_BATTING_3B.TREATED - TEAM_BATTING_2B.TREATED
  - log_TEAM_BATTING_1B.TREATED: log(TEAM_BATTING_1B.TREATED)
  - log_TEAM_BATTING_3B.TREATED: log(TEAM_BATTING_3B.TREATED)
  - log_TEAM_BASERUN_SB.TREATED: log(TEAM_BASERUN_SB.TREATED)
  - log_TEAM_BASERUN_CS.TREATED: log(TEAM_BASERUN_CS.TREATED)
  - sqrt_TEAM_PITCHING_HR.TREATED: sqrt(TEAM_PITCHING_HR.TREATED)
  - SB_PCT.TREATED: TEAM_BASERUN_SB.TREATED/(1.0XTEAM_BASERUN_SB.TREATED+ TEAM_BASERUN_CS.TREATED)

After applying all the data transformation steps, it resulted in 36 variables (original imputed variables, tranformed variables and flags) and retained all 2276 observations. This final dataset is used for variable selection and model building.

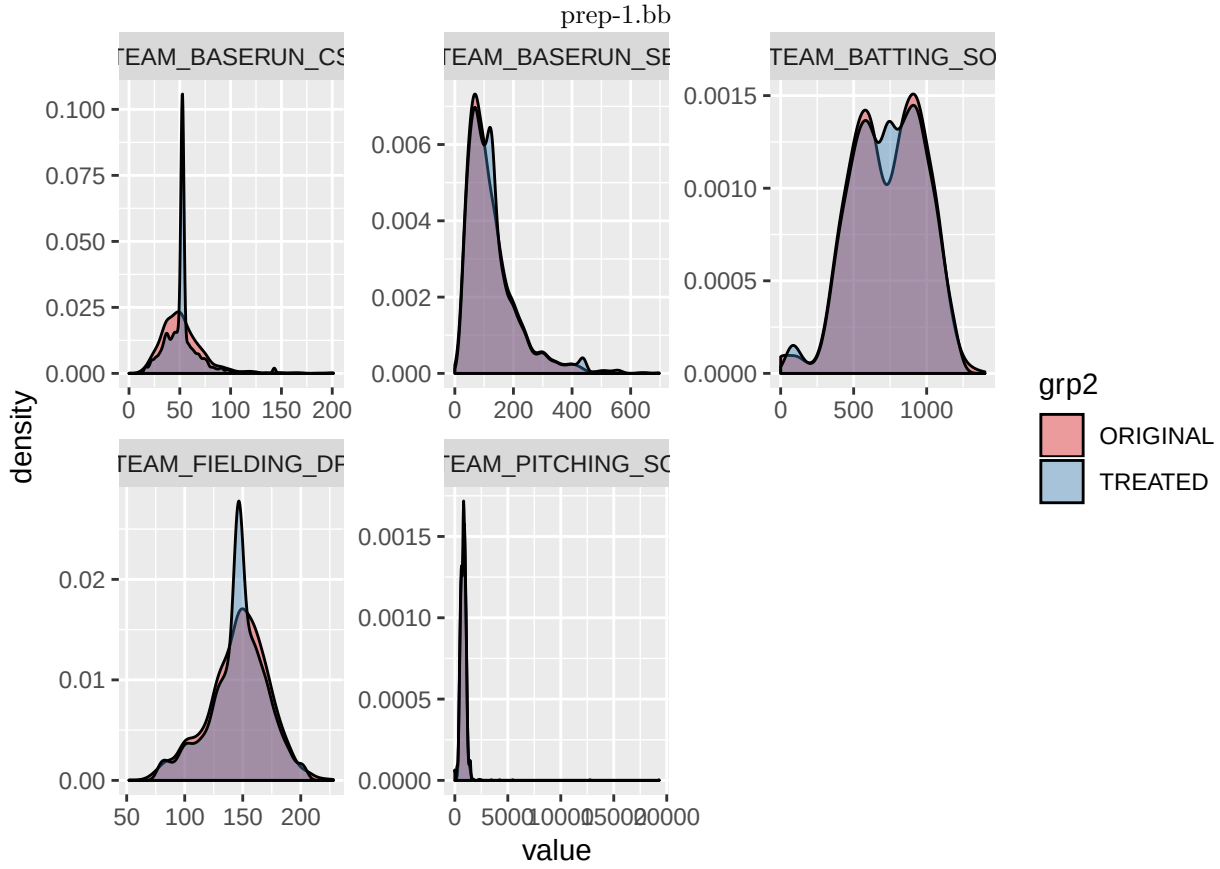

Figure 7: Densities of variables with missing values before and after imputation

4 BUILD MODELS

Since lack of knowledge in baseball, the 35 potential variables are ran through Gradient Boosting Machines (GBM) to get variable importance score with respect to wins target. It is interesting to see as shown in table 4, TEAM_FIELDING_E seem to be the top variable in the list which has negative theoretical effect on wins. Six models are built with variables selected using different criteria.

Table 4: Variable importance using GBM

	var	rel.inf
TEAM_FIELDING_E.TREATED	TEAM_FIELDING_E.TREATED	13.4654864
TEAM_BATTING_H.TREATED	TEAM_BATTING_H.TREATED	10.7233059
TEAM_BATTING_BB.TREATED	TEAM_BATTING_BB.TREATED	7.3256650
TEAM_BATTING_2B.TREATED	TEAM_BATTING_2B.TREATED	6.9951454
TEAM_BATTING_3B.TREATED	TEAM_BATTING_3B.TREATED	6.2511491
SB_PCT.TREATED	SB_PCT.TREATED	5.9631870
TEAM_BASERUN_SB.TREATED	TEAM_BASERUN_SB.TREATED	5.7179736
TEAM_PITCHING_SO.TREATED	TEAM_PITCHING_SO.TREATED	5.7072079
TEAM_FIELDING_DP.TREATED	TEAM_FIELDING_DP.TREATED	5.5152028
TEAM_PITCHING_H.TREATED	TEAM_PITCHING_H.TREATED	5.2842848
TEAM_PITCHING_BB.TREATED	TEAM_PITCHING_BB.TREATED	5.1173034
TEAM_BATTING_SO.TREATED	TEAM_BATTING_SO.TREATED	4.8029107
TEAM_BATTING_1B.TREATED	TEAM_BATTING_1B.TREATED	4.4977931
TEAM_PITCHING_HR.TREATED	TEAM_PITCHING_HR.TREATED	4.2923790
TEAM_BATTING_HR.TREATED	TEAM_BATTING_HR.TREATED	3.5455482
TEAM_BASERUN_CS.TREATED	TEAM_BASERUN_CS.TREATED	2.9657399
TEAM_BASERUN_SB.TREAT_FLAG	TEAM_BASERUN_SB.TREAT_FLAG	0.4636892
TEAM_PITCHING_SO.TREAT_FLAG	TEAM_PITCHING_SO.TREAT_FLAG	0.3510811
TEAM_BASERUN_CS.TREAT_FLAG	TEAM_BASERUN_CS.TREAT_FLAG	0.3107294
TEAM_FIELDING_DP.TREAT_FLAG	TEAM_FIELDING_DP.TREAT_FLAG	0.2614516
TEAM_BATTING_H.TREAT_FLAG	TEAM_BATTING_H.TREAT_FLAG	0.0762422
TEAM_BATTING_3B.TREAT_FLAG	TEAM_BATTING_3B.TREAT_FLAG	0.0666151
TEAM_BATTING_SO.TREAT_FLAG	TEAM_BATTING_SO.TREAT_FLAG	0.0654844
TEAM_PITCHING_BB.TREAT_FLAG	TEAM_PITCHING_BB.TREAT_FLAG	0.0572791
TEAM_BATTING_BB.TREAT_FLAG	TEAM_BATTING_BB.TREAT_FLAG	0.0428962
TEAM_PITCHING_HR.TREAT_FLAG	TEAM_PITCHING_HR.TREAT_FLAG	0.0402468
TEAM_BATTING_HR.TREAT_FLAG	TEAM_BATTING_HR.TREAT_FLAG	0.0394252
TEAM_FIELDING_E.TREAT_FLAG	TEAM_FIELDING_E.TREAT_FLAG	0.0288106
TEAM_BATTING_2B.TREAT_FLAG	TEAM_BATTING_2B.TREAT_FLAG	0.0250500
TEAM_PITCHING_H.TREAT_FLAG	TEAM_PITCHING_H.TREAT_FLAG	0.0007168
log_TEAM_BATTING_1B.TREATED	log_TEAM_BATTING_1B.TREATED	0.0000000
log_TEAM_BATTING_3B.TREATED	log_TEAM_BATTING_3B.TREATED	0.0000000
log_TEAM_BASERUN_SB.TREATED	log_TEAM_BASERUN_SB.TREATED	0.0000000
log_TEAM_BASERUN_CS.TREATED	log_TEAM_BASERUN_CS.TREATED	0.0000000
sqrt_TEAM_PITCHING_HR.TREATED	sqrt_TEAM_PITCHING_HR.TREATED	0.0000000

5 MODEL SELECTION

Model names and descriptions are shown in table 5. Data is split into 80%-20% for training and testing purposes to perform validation. All 35 variables are first ran through MLR model using backword selection search (backward.lm.allvars) as it is claimed to be good at dealing with multicollinearity. 21 (df-1) variables are selected of which 12 of them have <10 for variable inflation factor (VIF >10 is an indicative of multicollinearity). For the second model (backward.lm.orig.imp.w_flags) only the imputed variables and flag variables are ran through MLR using backward search in which 17 are selected of which 16 have valid vifs. Next model is ran using 14 imputed variables (backward.lm.orig.imp) of which 12 are selected by backward search. 4th model is ran using 7 variables from model 4 that have valid vifs. 5th model(orig.imp.no.vif) is ran using 11 variables selected based on EDA observations, and importance shown by GBMs. It showed 7 variables having valid VIFs. Last model (selected.vars.no.vif) is ran using these 7 variables alone. Table 5 shows various performance metrics of six models: The goodness of fit statistics such as Adj-Rsq, AIC, BIC and predicitive measures such as mean absolute errors (MAE) from train and test splits, mean squared error (MSE) from test split and degrees of freedom (df-1 gives number of variables used) and number of variables with valid VIFs.

Table 5: Different models and metrics

model_name	model_description	adj.r.squared	AIC	BIC	train.MAE	test.MAE	test.MSE	df	valid_vif
backward.lm.allvars	treated+flag+transformed variables - backward selection of variables	0.36	14343.51	14470.21	9.65	9.38	142.86	22	12
backward.lm.orig.imp.w_flags	treated+flag variables - backward selection of variables	0.35	14363.59	14468.25	9.71	9.45	144.75	18	16
backward.lm.orig.imp	treated variables - backward selection of variables	0.30	14492.79	14569.91	10.03	9.85	156.28	13	8
orig.imp.no.vif	treated variables, vifs addressed - lm	0.29	14515.73	14570.81	10.19	10.07	161.17	9	8
selected.vars	selected variables through trees/observation - lm	0.32	14431.42	14503.03	9.96	9.83	154.47	12	7
selected.vars.no.vif	selected variables, vifs addressed - lm	0.25	14612.68	14662.26	10.39	10.03	164.30	8	7

The model metrics are compared against each other using ranking on the respective metrics. While the models with more variables performed relatively better, principle of parsimoney says the model should be as simple as possible. While "selected_vars" model has 11 variables and some VIF concerns, this model has been selected as the champion model in terms of performance. It is acknowledged that two variables selected are transformed variables which may be hard to interpret for practical purposes. The model need to undergo excruciating review with baseball experts before putting into production. The analyst was not able to use judgement in addressing outliers/selecting variables due to lack of knowledge in the subject and may need to be revisted. But the significance of the variables and predictive power of the model can provide some insights to the league leader that may be useful.

Table 6: Model fit metrics - ranks

model_name	Rank.adjR	Rank.AIC	Rank.BIC	Rank.train.MAE	Rank.test.MAE	Rank.test.MSE	df	valid_vif
backward.lm.allvars	1	1	2	1	1	1	22	12
backward.lm.orig.imp.w_flags	2	2	1	2	2	2	18	16
backward.lm.orig.imp	4	4	4	4	4	4	13	8
orig.imp.no.vif	5	5	5	5	6	5	9	8
selected.vars	3	3	3	3	3	3	12	7
selected.vars.no.vif	6	6	6	6	5	6	8	7

5.1 Final selected model

The selected model is rerun using the entire training data set before scoring new data. The model parameters significance and summary are shown below. It captured 31.4% variation in the data and the F-statistic says the model is significant. While most of the selected variables show significance, the variable with squareroot transformation is insignificant. But removing any of the variables is effecting model performance so it is decided to keep all of them. It certainly indicates an unstable model and further diagnostics checks need to be done before revisitng the model.

```
##
## The best model selected
## ===========================================================
##                                    Dependent variable:
##                                ---------------------------
##                                    TARGET_WINS.TREATED
## -----------------------------------------------------------
## TEAM_FIELDING_E.TREATED                 -0.047
##                                          (0.003)
##                                        t = -14.665
##                                       p = 0.000***
##
## TEAM_BATTING_H.TREATED                   0.084
##                                          (0.020)
##                                         t = 4.266
##                                      p = 0.00003***
##
## TEAM_BATTING_BB.TREATED                  0.017
##                                          (0.003)
##                                         t = 5.434
##                                      p = 0.00000***
##
## TEAM_BATTING_3B.TREATED                  0.078
##                                          (0.021)
##                                         t = 3.719
##                                       p = 0.0003***
##
## TEAM_PITCHING_H.TREATED                  0.004
##                                          (0.001)
##                                         t = 6.233
##                                       p = 0.000***
##
## SB_PCT.TREATED                          35.327
##                                          (2.932)
##                                        t = 12.047
##                                       p = 0.000***
##
## TEAM_BATTING_1B.TREATED                 -0.046
##                                          (0.020)
##                                        t = -2.338
##                                        p = 0.020**
##
## sqrt_TEAM_PITCHING_HR.TREATED           -0.091
##                                          (0.353)
##                                        t = -0.257
```

```
##                                          p = 0.798
##
## TEAM_BASERUN_SB.TREAT_FLAG                17.941
##                                           (1.441)
##                                         t = 12.448
##                                        p = 0.000***
##
## TEAM_BATTING_2B.TREATED                   -0.075
##                                           (0.022)
##                                         t = -3.448
##                                        p = 0.001***
##
## TEAM_PITCHING_SO.TREATED                  -0.006
##                                           (0.002)
##                                         t = -3.275
##                                        p = 0.002***
##
## Constant                                  -3.249
##                                           (4.827)
##                                         t = -0.673
##                                          p = 0.501
##
## ---------------------------------------------------------
## Observations                               2,276
## R2                                         0.318
## Adjusted R2                                0.314
## Residual Std. Error                  12.551 (df = 2264)
## F Statistic                       95.804*** (df = 11; 2264)
## =========================================================
## Note:                           *p<0.1; **p<0.05; ***p<0.01
```

The equation for the selected model:

TARGET_WINS.TREATED = -3.249 - 0.047 x TEAM_FIELDING_E.TREATED + 0.084 x TEAM_BATTING_H.TREATED + 0.017 x TEAM_BATTING_BB.TREATED + 0.078 x TEAM_BATTING_3B.TREAT + 0.004 x TEAM_PITCHING_H.TREATED + 35.327 x SB_PCT.TREATED - 0.046 x TEAM_BATTING_1B.TREATED - 0.091 x sqrt_TEAM_PITCHING_HR.TREATED + 17.941 x TEAM_BASERUN_SB.TREAT_FLAG - 0.075 x TEAM_BATTING_2B.TREATED - 0.006 x TEAM_PITCHING_SO.TREATED

While the sign of coefficient for TEAM_FIELDING_E parameter is negative tallying with the theoretical effect, signs for some variables do not match theoretical effect which can drive lot of questions about this model. It is also noted that missing values in TEAM_BASERUN_SB seem to have significant effect on the model.

5.1.1 Diagnostic checks

Further the model is checked for any unsual patterns in the residuals using diagnostic plots and influential points using influence plot as shown in figures 8 & 9. The diamond shape of residuals seem to be result of winsorizing the variables for outliers. Clearly there are also several leverage points which when removed can effect the model performance significantly. These points should be reviewed with experts before removing them.

checks-1.bb

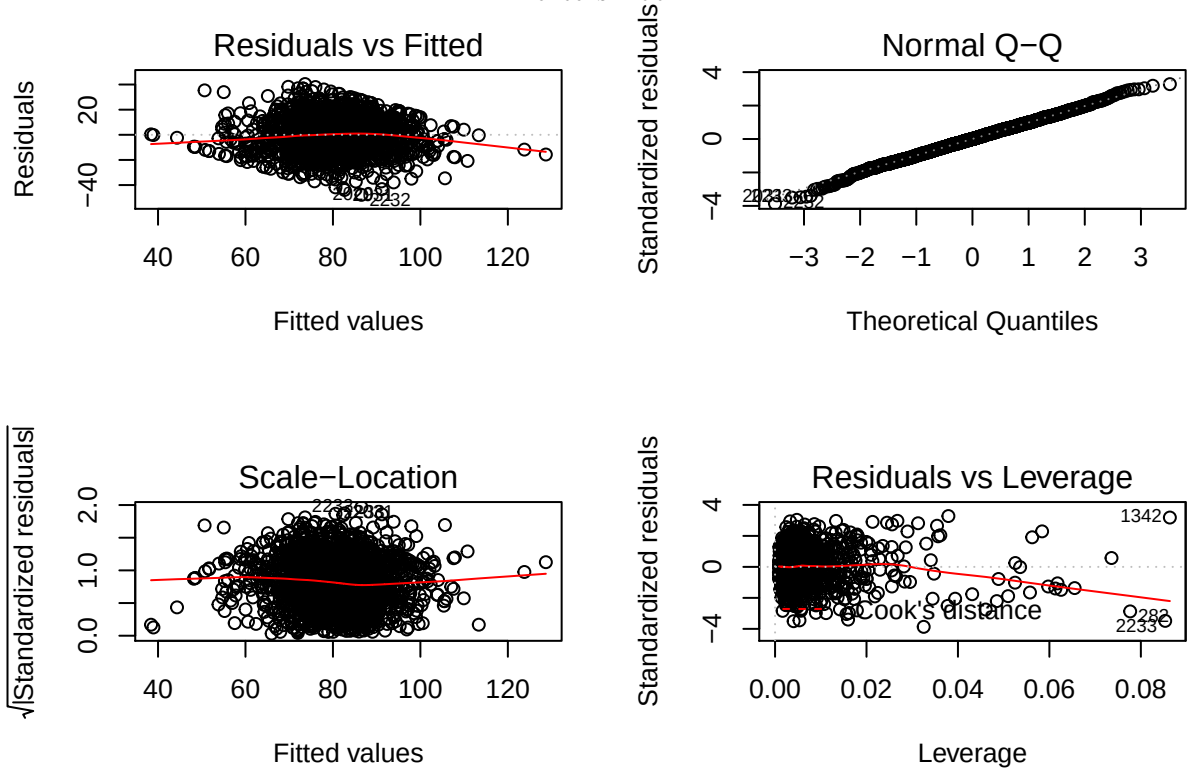

Figure 8: Diagnostic plot of selected best model

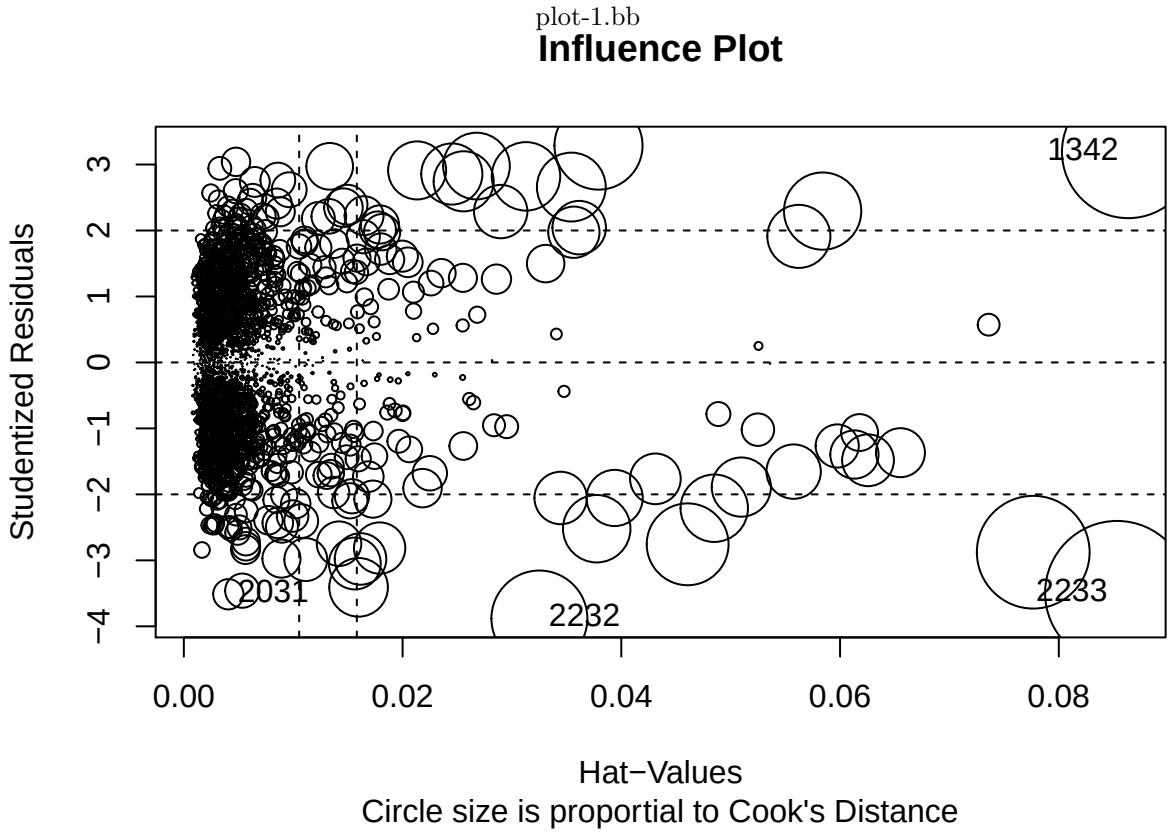

Figure 9: Influence plot of selected best model

```
##        StudRes         Hat       CookD
## 1342  3.191225 0.086347476 0.079881126
## 2031 -3.513838 0.004068954 0.004182776
## 2232 -3.881456 0.032509493 0.041925847
## 2233 -3.493806 0.085329778 0.094429557
```

6 STAND ALONE SCORING PROGRAM / MODEL DEPLOYMENT CODE

The new dataset is processed through same data preprocessing steps used on training data.

```
##read test data file
df.test <- read.csv("moneyball_test.csv", header = TRUE)
df.test<- within(df.test, rm(TEAM_BATTING_HBP))
moneyball.test <- df.test

##Winsorize using train data
varnames <- colnames(df.test)

replace.values <- function(df.col,val1, val2) {
  df.col[df.col<val1] <- val1
  df.col[df.col>val2] <- val2
  return(df.col)
```

```
  }

df.test[] <- lapply(varnames, function(x){ifelse(x=="INDEX", df.test[x],replace.values(df.test[x],train

###Replace missing values with train means

df.test$TEAM_BATTING_SO[is.na(df.test$TEAM_BATTING_SO)] = missing.var.means$TEAM_BATTING_SO
df.test$TEAM_BASERUN_SB[is.na(df.test$TEAM_BASERUN_SB)] = missing.var.means$TEAM_BASERUN_SB
df.test$TEAM_BASERUN_CS[is.na(df.test$TEAM_BASERUN_CS)] = missing.var.means$TEAM_BASERUN_CS
df.test$TEAM_FIELDING_DP[is.na(df.test$TEAM_FIELDING_DP)] = missing.var.means$TEAM_FIELDING_DP
df.test$TEAM_PITCHING_SO[is.na(df.test$TEAM_PITCHING_SO)] = missing.var.means$TEAM_PITCHING_SO

###Create flags table to store flag for where data is treated
flag_table <- data.frame()
flag_table <- (moneyball.test==df.test)
flag_table[which(is.na(flag_table))] <- FALSE
test.flag.table<-as.data.frame(ifelse(flag_table == FALSE, 1,0))
test.flag.table<- within(test.flag.table, rm(INDEX))

###add indicators to columns
colnames(df.test) <- paste(colnames(df.test), "TREATED", sep = ".")
#colnames(moneyball.test) <- paste(colnames(moneyball.test), "ORIGINAL", sep = ".")
colnames(test.flag.table) <- paste(colnames(test.flag.table), "TREAT_FLAG", sep = ".")

#Straighten Relationships
df.test$TEAM_BATTING_1B.TREATED <- df.test$TEAM_BATTING_H.TREATED - df.test$TEAM_BATTING_HR.TREATED - d
df.test$log_TEAM_BATTING_1B.TREATED <- log(df.test$TEAM_BATTING_1B.TREATED)
df.test$log_TEAM_BATTING_3B.TREATED <- log(df.test$TEAM_BATTING_3B.TREATED)
df.test$log_TEAM_BASERUN_SB.TREATED <- log(df.test$TEAM_BASERUN_SB.TREATED)
df.test$log_TEAM_BASERUN_CS.TREATED <- log(df.test$TEAM_BASERUN_CS.TREATED)
df.test$sqrt_TEAM_PITCHING_HR.TREATED <- sqrt(df.test$TEAM_PITCHING_HR.TREATED)
df.test$SB_PCT.TREATED <- df.test$TEAM_BASERUN_SB.TREATED/(1.0*df.test$TEAM_BASERUN_SB.TREATED+df.test$

names(coef(selected.vars)) -> bestmodel.vars

#df.test1 <- cbind(df.test, test.flag.table)
newtest <- cbind(df.test, test.flag.table)  %>% select(bestmodel.vars[2:12])
```

7 SCORED DATA FILE (SCORE THE MONEYBALL_TEST DATA SET)

The trained model is is used to score the processed new data and predicted wins are stored into a csv file.

```
moneyball.test$P_TARGET_WINS <- round(predict(object=selected.vars,newdata=newtest))

predict.file <- moneyball.test %>% select(INDEX,P_TARGET_WINS)
write.csv(predict.file,"P_TARGET_WINS_SETUNAMBURU.csv",row.names = FALSE )
```

8 CONCLUSION

A Multiple linear regression model has been built to predict the number of wins a team can achieve given their performance statistics such as doubles, tripples etc. Six models are built and compared against each other using fit statistics and predictive performance measures. While it is not simple, a model with 11 variables has been selected as a champion model among the models built after taking all performance measures into consideration. Due to lack of knowledge in the subject, outliers and missing values are addressed using simple methods.

The model did not pass diagnostic checks but provided several insights into the data and the variables. Some knowledge about the subject may have helped the analyst with better preprocessing of the data (here very cautionary appraoch has been taken) that may have helped to build much robust model. The selected model was able to achieve Adj. R-sq value of 32% and mean absolute error of 9.83% on test split of the data. While signs of some of the coefficients did not make sense from theoretical perspective it is worthwhile to investigate the effects as they seem to be statistically significant. If they make practical sense, strategies can be developed to help the players to increase their probability of winning.

After all George Box said: " All models are wrong, some are useful"

9 BINGO BONUS

9.1 Imputation using MICE package

R MICE package is used to impute missing values from variables in the given dataset using predictive mean method. 5 different imputed datasets are generated and the below density plot in figure 10 shows density of the variables before (below) and after imputation (magenta). If the shapes match we could tell that the imputed values are indeed plausible. In this case, the original shapes of the densities seem to be distorted indicating this imputation did not result in similar data.

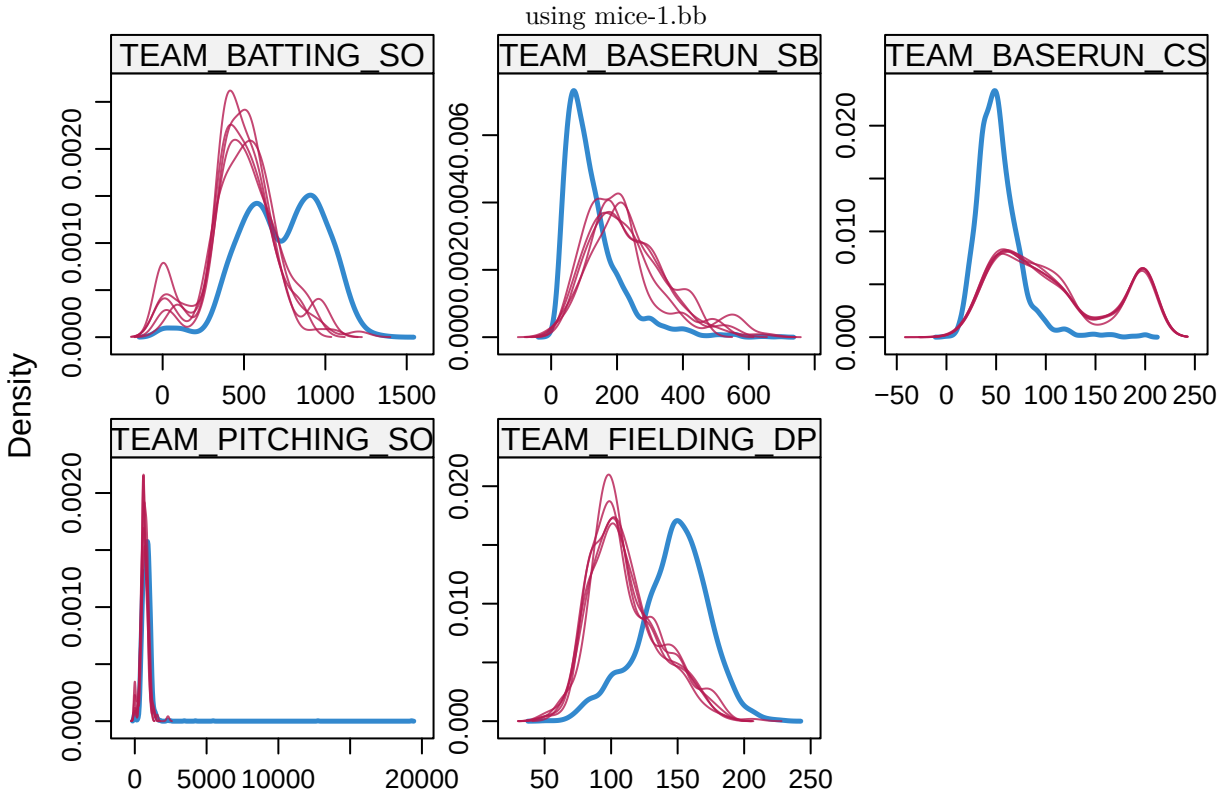

Figure 10: Before/After density plot of missing values imputation using MICE

9.2 Variable selection using decision trees

Decision trees are widely used for variable selection especially when we don't know where to start. A simple decision tree is built using tree package with WINS as the target variable and is shown in figure 11. This can guide the analyst to devide variables into segments (for example does TEAM_FIELDING_E< 326 drive more wins?) and can be turned into dummy variables to use as predictors. Trees that are too deep can run into overfitting problems. So judgement and practical sense to segment the data is required to turn branches into predictor variables.

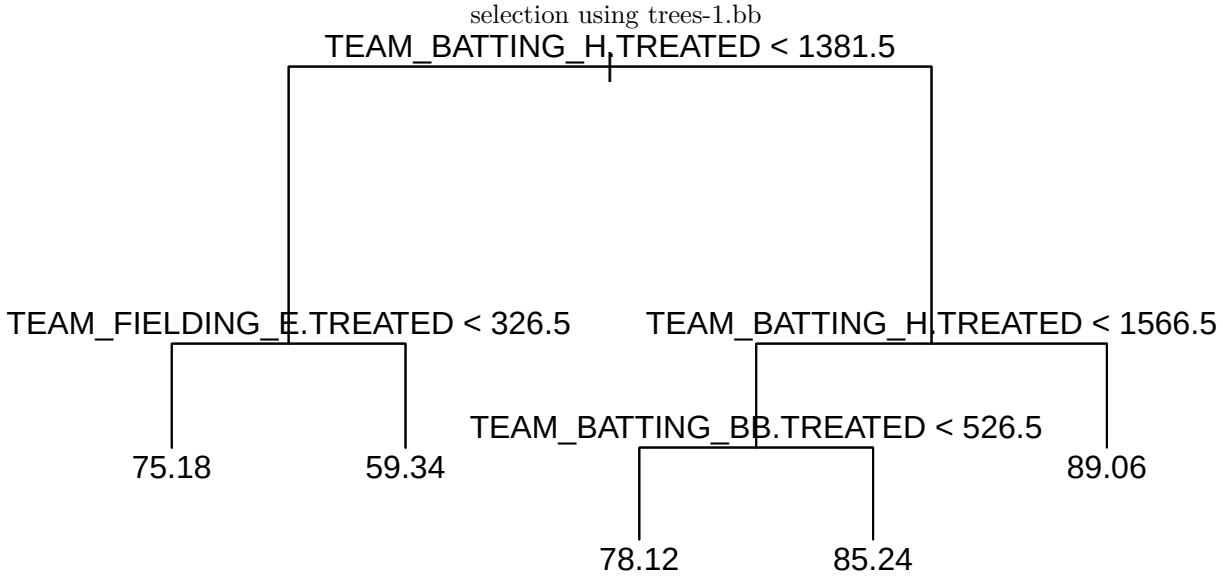

Figure 11: Simple decision tree

Random forest algorithm is used for identifying important variables, the output is shown in table 7. It shows that TEAM_BATTING_H and TEAM_FIELDING_E have most influence on WINS, in other words error is most reduced if these two variables are present in the model.

Table 7: Variable importance using Random Forests

	%IncMSE	IncNodePurity
TEAM_BATTING_H.TREATED	52.6475952	48439.1234
TEAM_BATTING_2B.TREATED	15.0034091	29034.2267
TEAM_BATTING_3B.TREATED	18.0908524	20398.2336
TEAM_BATTING_HR.TREATED	17.2071058	20107.3983
TEAM_BATTING_BB.TREATED	22.2991660	32447.1983
TEAM_BATTING_SO.TREATED	14.0153939	20130.4396
TEAM_BASERUN_SB.TREATED	12.8267087	18220.2851
TEAM_BASERUN_CS.TREATED	2.8756999	8226.8357
TEAM_PITCHING_H.TREATED	20.5025325	26937.1728
TEAM_PITCHING_HR.TREATED	18.4000250	19704.0768
TEAM_PITCHING_BB.TREATED	13.9380728	23726.1378
TEAM_PITCHING_SO.TREATED	15.2543710	21770.4497
TEAM_FIELDING_E.TREATED	50.1039820	47784.6651
TEAM_FIELDING_DP.TREATED	5.6297982	17200.2864
TEAM_BATTING_1B.TREATED	17.9553493	21582.5568
log_TEAM_BATTING_1B.TREATED	19.5466791	21925.2045
log_TEAM_BATTING_3B.TREATED	19.7616063	20622.5947
log_TEAM_BASERUN_SB.TREATED	11.8680147	17550.4701
log_TEAM_BASERUN_CS.TREATED	2.9724834	8032.3695
sqrt_TEAM_PITCHING_HR.TREATED	19.2047606	19876.3014
SB_PCT.TREATED	8.7864033	18276.4229
TEAM_BATTING_H.TREAT_FLAG	0.8421492	1776.7728
TEAM_BATTING_2B.TREAT_FLAG	0.0829044	480.9169
TEAM_BATTING_3B.TREAT_FLAG	0.0767150	434.9997
TEAM_BATTING_HR.TREAT_FLAG	0.0530149	348.6762
TEAM_BATTING_BB.TREAT_FLAG	0.1789500	736.2677
TEAM_BATTING_SO.TREAT_FLAG	0.6680671	1527.3147
TEAM_BASERUN_SB.TREAT_FLAG	1.0678246	1739.4542
TEAM_BASERUN_CS.TREAT_FLAG	1.6233402	1501.5407
TEAM_PITCHING_H.TREAT_FLAG	0.8019632	2076.0890
TEAM_PITCHING_HR.TREAT_FLAG	0.0923533	662.5016
TEAM_PITCHING_BB.TREAT_FLAG	0.0020469	466.8476
TEAM_PITCHING_SO.TREAT_FLAG	0.6502757	1421.7701
TEAM_FIELDING_E.TREAT_FLAG	1.2788241	2353.0365
TEAM_FIELDING_DP.TREAT_FLAG	0.7668386	1797.5205

9.3 Performance of best model using GLM

The champion model selected is rerun using GLM routine in R. With family = "Gaussian", GLM is nothing but OLS and the model metrics in table 8 indeed match with OLS metrics shown in above sections for this model.

Table 8: Metrics using GLM

Rsq	MAE	MSE	AIC
0.3244925	9.834689	154.4734	14431.42