DATA 621 Business Analytics & Data Mining

Homework #1 - Moneyball

Kyle Gilde

2/18/2018


```
##            installed_and_loaded.packages.
## prettydoc                              TRUE
## psych                                  TRUE
## knitr                                  TRUE
## tidyverse                              TRUE
## ggthemes                               TRUE
## corrplot                               TRUE
## Hmisc                                  TRUE
## data.table                             TRUE
## missForest                             TRUE
## mltools                                TRUE
## htmlTable                              TRUE
## broom                                  TRUE
## MLmetrics                              TRUE
```

Prompt and Data Overview

In this homework assignment, you will explore, analyze and model a data set containing approximately 2200 records.

Each record represents a professional baseball team from the years 1871 to 2006 inclusive. Each record has the performance of the team for the given year, with all of the statistics adjusted to match the performance of a 162 game season.

Your objective is to build a multiple linear regression model on the training data to predict the number of wins for the team. You can only use the variables given to you (or variables that you derive from the variables provided). Below is a short description of the variables of interest in the data set:

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

1. DATA EXPLORATION

Non-visual exploration

The data set contains 15 explanatory variables, and they can be categorized into four groups:

1. 7 batting metrics
2. 2 baserunning metrics
3. 2 fielding metrics
4. 4 pitching metrics

Data Inspection

Let's take a look at the data's non-statistical aspects.

- In the table below, we that the data set contains all integers.
- The data set has less than 10% complete cases.
- Six variables are missing values with TEAM_BATTING_HBP, TEAM_BASERUN_CS & TEAM_FIELDING_DP being the most sparse with 92%, 34% & 13% NA s, respectively.
- We will have to drop them or try to impute the missing values.

vars	class_type	n_rows	complete_cases	NA_ct	NA_pct	unique_value_ct	most_
TARGET_WINS	integer	2276	191	0	0.0	108	83; 79
TEAM_BATTING_H	integer	2276	191	0	0.0	569	1458;
TEAM_BATTING_2B	integer	2276	191	0	0.0	240	227; 2
TEAM_BATTING_3B	integer	2276	191	0	0.0	144	35; 29
TEAM_BATTING_HR	integer	2276	191	0	0.0	243	21; 10
TEAM_BATTING_BB	integer	2276	191	0	0.0	533	502; 4
TEAM_BATTING_SO	integer	2276	191	102	4.5	823	NA's; 0
TEAM_BASERUN_SB	integer	2276	191	131	5.8	349	NA's; 6
TEAM_BASERUN_CS	integer	2276	191	772	33.9	129	NA's; 5
TEAM_BATTING_HBP	integer	2276	191	2085	91.6	56	NA's; 5
TEAM_PITCHING_H	integer	2276	191	0	0.0	843	1494;
TEAM_PITCHING_HR	integer	2276	191	0	0.0	256	114; 2
TEAM_PITCHING_BB	integer	2276	191	0	0.0	535	536; 4
TEAM_PITCHING_SO	integer	2276	191	102	4.5	824	NA's; 0
TEAM_FIELDING_E	integer	2276	191	0	0.0	549	122; 1
TEAM_FIELDING_DP	integer	2276	191	286	12.6	145	NA's; 1

Statistical Summary

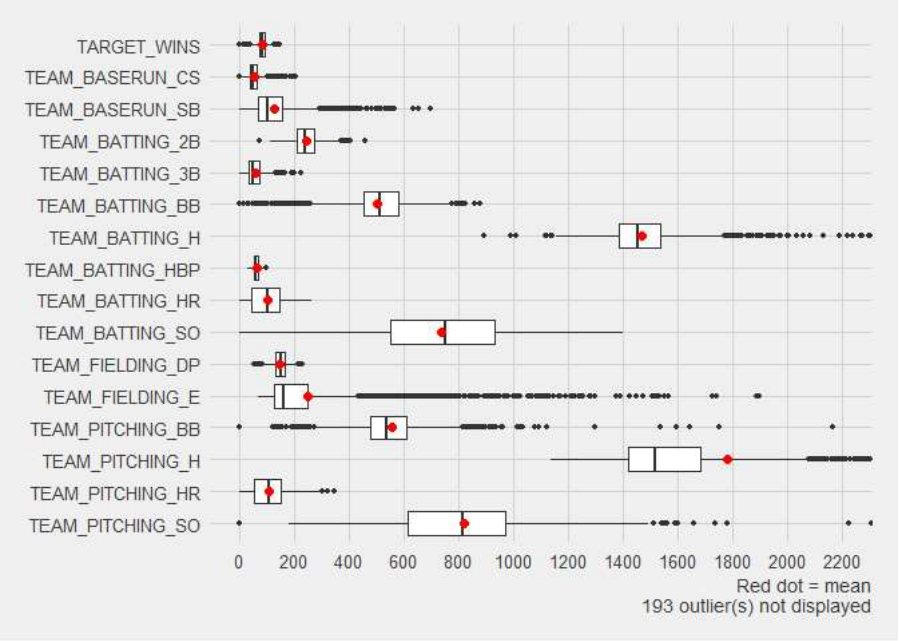
	n	min	Q.1st	median	mean	Q.3rd	max	range	sd	s
TARGET_WINS	2,276	0	71	82	80.8	92	146	146	15.8	0
TEAM_BATTING_H	2,276	891	1,383	1,454	1,469.3	1,537.2	2,554	1,663	144.6	3
TEAM_BATTING_2B	2,276	69	208	238	241.2	273	458	389	46.8	1
TEAM_BATTING_3B	2,276	0	34	47	55.2	72	223	223	27.9	0
TEAM_BATTING_HR	2,276	0	42	102	99.6	147	264	264	60.5	1
TEAM_BATTING_BB	2,276	0	451	512	501.6	580	878	878	122.7	2
TEAM_BATTING_SO	2,174	0	548	750	735.6	930	1,399	1,399	248.5	5
TEAM_BASERUN_SB	2,145	0	66	101	124.8	156	697	697	87.8	1
TEAM_BASERUN_CS	1,504	0	38	49	52.8	62	201	201	23	0
TEAM_BATTING_HBP	191	29	50.5	58	59.4	67	95	66	13	0
TEAM_PITCHING_H	2,276	1,137	1,419	1,518	1,779.2	1,682.5	30,132	28,995	1,406.8	2
TEAM_PITCHING_HR	2,276	0	50	107	105.7	150	343	343	61.3	1
TEAM_PITCHING_BB	2,276	0	476	536.5	553	611	3,645	3,645	166.4	3
TEAM_PITCHING_SO	2,174	0	615	813.5	817.7	968	19,278	19,278	553.1	1

	n	min	Q.1st	median	mean	Q.3rd	max	range	sd	
TEAM_FIELDING_E	2,276	65	127	159	246.5	249.2	1,898	1,833	227.8	
TEAM_FIELDING_DP	1,990	52	131	149	146.4	164	228	176	26.2	

Visual Exploration

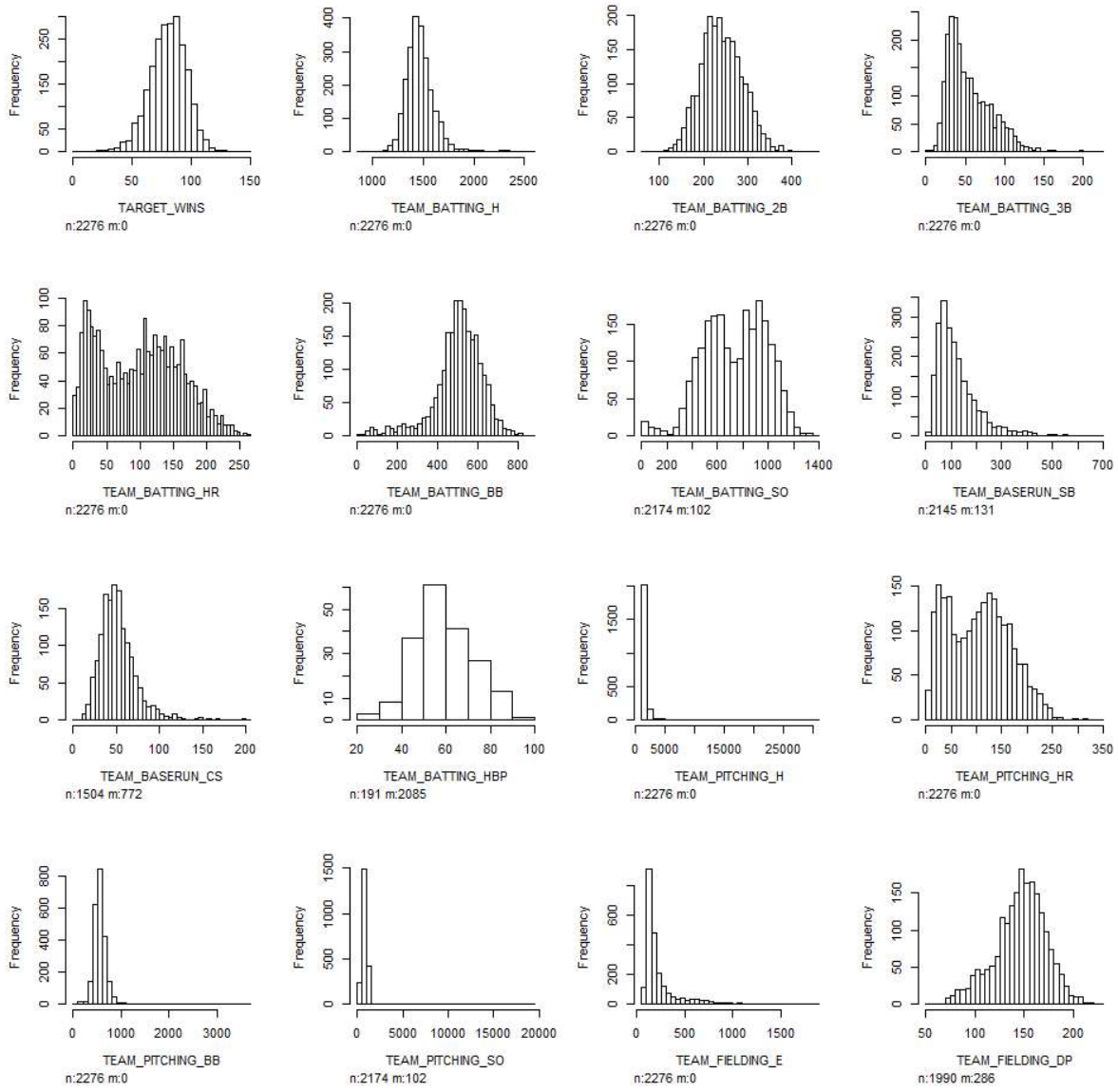
Boxplots

- In the boxplots below, we see that the variance of some of the explanatory variables greatly exceeds the variance of the response games-won variable.
- According to the summary statistics table above, the TARGET_WIN's standard deviation is the 2nd smallest of all the variables.
- A few of them have so many outliers that we may be dealing with non-unimodal distributions.
- The data set has 193 observations that are more extreme than the 1.5 * IQR of the boxplot whiskers.



Histograms

- The histograms below confirm that both the batting & pitching home-run variables are bimodal as well as the batting strike-out variable.
- Many variables have significant right skews.
- TEAM_BATTING_BB & TEAM_FIELDING_DP are left-skewed.
- The distributions for TEAM_BATTING_HBP, TEAM_BASERUN_CS & TEAM_FIELDING_DP are centered away from zero, so it seems unlikely that all the `NA`s are mislabelled zero values.



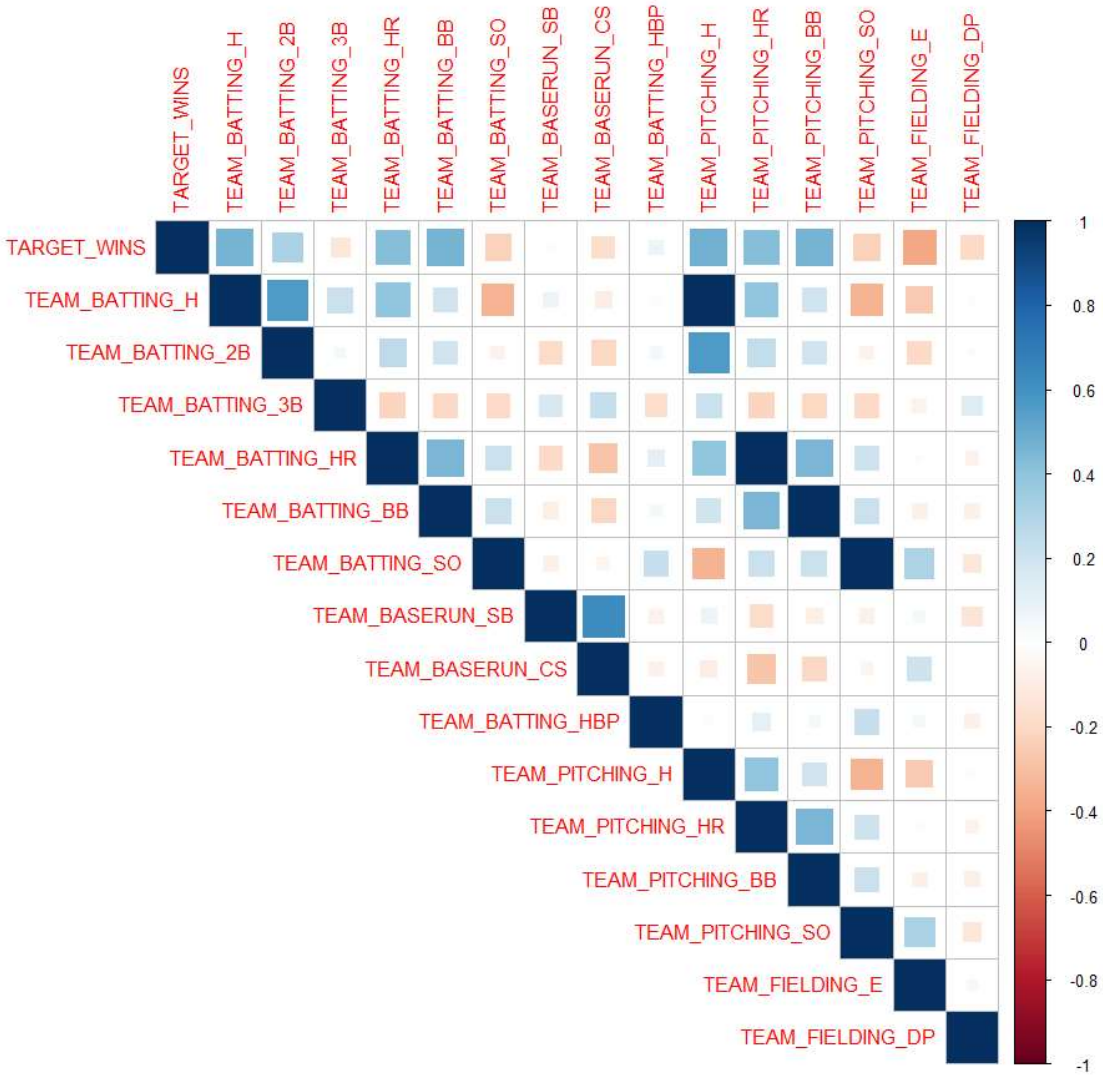
Correlations

- In the correlation table and plot below, we see 4 pairs of highly correlated variables. They are corresponding metrics for batting and pitching: home runs, walks, strike outs & hits (rows 1-4). These may present multicollinearity issues in our modeling.
- Rows 8 to 11 show batting & pitching hits and walks are moderately correlated to our response variable. Let's consider using them for one our models.

Top 12 Correlated Variable Pairs

	Var1	Var2	Correlation
1	TEAM_PITCHING_HR	TEAM_BATTING_HR	0.9999
2	TEAM_PITCHING_BB	TEAM_BATTING_BB	0.9999
3	TEAM_PITCHING_SO	TEAM_BATTING_SO	0.9998
4	TEAM_PITCHING_H	TEAM_BATTING_H	0.9992
5	TEAM_BASERUN_CS	TEAM_BASERUN_SB	0.6247

	Var1	Var2	Correlation
6	TEAM_BATTING_2B	TEAM_BATTING_H	0.5618
7	TEAM_PITCHING_H	TEAM_BATTING_2B	0.5605
8	TEAM_PITCHING_H	TARGET_WINS	0.4712
9	TEAM_BATTING_H	TARGET_WINS	0.4699
10	TEAM_BATTING_BB	TARGET_WINS	0.4687
11	TEAM_PITCHING_BB	TARGET_WINS	0.4684
12	TEAM_PITCHING_HR	TEAM_BATTING_BB	0.4566



2. DATA PREPARATION

Creating Bins

From the histograms and summary statitics, TEAM_PITCHING_H seems to have the extreme outliers, so let's put these values into quintile bins to mitigate their effect.

```
train_raw$TEAM_PITCHING_H <- bin_data(train_raw$TEAM_PITCHING_H, bins = 5, binType = "quantile")

levels(train_raw$TEAM_PITCHING_H) <- c("One","Two","Three","Four","Five")
```

Missing Value Imputation

Let's use the missForest package to do nonparametric missing-value imputation using Random Forest.

First, let's try leaving in TEAM_BATTING_HPB, even though it has all of those `NA`s.

When we check the out-of-the-bag error, we would like to the normalized root mean-squares error (NRMSE) near zero, which would indicate a well-fitted imputation. However, as suspected, the normalized root mean-squares error (NRMSE) for TEAM_BATTING_HBP is nearly twice as large as the next highest NRMSE. Let's remove it and impute the missing values again.

Source: Accuracy and Errors for Models

```
##   missForest iteration 1 in progress...done!
##   missForest iteration 2 in progress...done!
##   missForest iteration 3 in progress...done!
##   missForest iteration 4 in progress...done!
##   missForest iteration 5 in progress...done!
```

variable	NA_ct	NA_pct	MSE	RMSE	NRMSE
TEAM_BATTING_HBP	2085	91.61	194.34	13.94	0.21
TEAM_FIELDING_DP	286	12.57	341.17	18.47	0.10
TEAM_BASERUN_CS	772	33.92	178.53	13.36	0.07
TEAM_BASERUN_SB	131	5.76	1637.94	40.47	0.06
TEAM_BATTING_SO	102	4.48	1827.35	42.75	0.03
TEAM_PITCHING_SO	102	4.48	130365.14	361.06	0.02

The remaining 5 imputed variables have NRMSE values have better fits and range from .02 to .11.

```
##   missForest iteration 1 in progress...done!
##   missForest iteration 2 in progress...done!
##   missForest iteration 3 in progress...done!
##   missForest iteration 4 in progress...done!
##   missForest iteration 5 in progress...done!
```

variable	NA_ct	NA_pct	MSE	RMSE	NRMSE
TEAM_FIELDING_DP	286	12.57	346.87	18.62	0.11
TEAM_BASERUN_CS	772	33.92	185.08	13.60	0.07
TEAM_BASERUN_SB	131	5.76	1629.75	40.37	0.06

variable	NA_ct	NA_pct	MSE	RMSE	NRMSE
TEAM_BATTING_SO	102	4.48	2467.60	49.67	0.04
TEAM_PITCHING_SO	102	4.48	172102.31	414.85	0.02

3. BUILD MODELS

Model 1: All Variables

First, let's build a model with all variables.

```
##
## Call:
## lm(formula = TARGET_WINS ~ ., data = train_imputed)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -55.389  -8.298   0.195   8.167  54.913
##
## Coefficients:
##                         Estimate Std. Error t value Pr(>|t|)
## (Intercept)           28.6409850  5.7749125   4.960 7.59e-07 ***
## TEAM_BATTING_H         0.0467974  0.0036512  12.817  < 2e-16 ***
## TEAM_BATTING_2B       -0.0080999  0.0091590  -0.884  0.37659
## TEAM_BATTING_3B        0.0248277  0.0166190   1.494  0.13533
## TEAM_BATTING_HR        0.0655984  0.0277124   2.367  0.01801 *
## TEAM_BATTING_BB        0.0123925  0.0052297   2.370  0.01789 *
## TEAM_BATTING_SO       -0.0195892  0.0026891  -7.285 4.43e-13 ***
## TEAM_BASERUN_SB        0.0470115  0.0054356   8.649  < 2e-16 ***
## TEAM_BASERUN_CS        0.0262674  0.0156964   1.673  0.09437 .
## TEAM_PITCHING_HTwo    -0.9872101  0.8882665  -1.111  0.26652
## TEAM_PITCHING_HThree  -1.0974513  0.9532332  -1.151  0.24973
## TEAM_PITCHING_HFour   -4.3120963  1.1048503  -3.903 9.78e-05 ***
## TEAM_PITCHING_HFive   -3.1433220  1.4102628  -2.229  0.02592 *
## TEAM_PITCHING_HR       0.0291162  0.0247491   1.176  0.23954
## TEAM_PITCHING_BB      -0.0031523  0.0035931  -0.877  0.38041
## TEAM_PITCHING_SO       0.0024559  0.0009026   2.721  0.00656 **
## TEAM_FIELDING_E       -0.0322373  0.0023362 -13.799  < 2e-16 ***
## TEAM_FIELDING_DP      -0.1123643  0.0140611  -7.991 2.11e-15 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 12.65 on 2258 degrees of freedom
## Multiple R-squared:  0.3602, Adjusted R-squared:  0.3554
## F-statistic: 74.77 on 17 and 2258 DF,  p-value: < 2.2e-16
```

In the model output above, we notice the following:

- Of the 14 continuous & intercept variables, 9 have statistically significant p-values at the 5% significance level. In the 3rd model, we will explore which of the variables are actually necessary to a parsimonious model.
- Only 2 out of the 4 TEAM_PITCHING_H quintile dummy variables are statistically significant, so let's check an ANOVA summary to make sure that the group means are significantly different. The F-statistic's p-value is near zero, so we would reject the null hypothesis that the means are equal and keep the dummy variables in the model.

```
##                   Df Sum Sq Mean Sq F value Pr(>F)
## TEAM_PITCHING_H    4  22687    5672   23.77 <2e-16 ***
## Residuals       2271 541810     239
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

- While we would expect negative coefficients for Fielding Errors, Walks Allowed & Batting Strikeouts, we wouldn't expect negative values for Double Plays & Batting Doubles. Additionally, it's surprising that Caught Stealing & Home Runs Allowed do not have negative coefficients. While none of the variables are perfectly collinear and were automatically removed from the model, these unexpected signs may indicate that we have some.
- At 0.3602, the adjusted R^2 is not as high as we would prefer, and it indicates that only 36% of the variance in the response variable can be explained by the predictor variables.
- At 75, the F-statistic is large, and the model's p-value is near zero. If the model's diagnostics are sufficient, these values indicate that we would reject the null hypothesis that there is no relationship between the explanatory & response variables.

Model 2: Only Highly Correlated Variables

From our Correlated Variable Pairs table, let's use only the 4 explanatory variables that were most correlated with the response variable.

```
##               Var1        Var2 Correlation
## 1  TEAM_PITCHING_H TARGET_WINS   0.4712343
## 2   TEAM_BATTING_H TARGET_WINS   0.4699467
## 3  TEAM_BATTING_BB TARGET_WINS   0.4686879
## 4 TEAM_PITCHING_BB TARGET_WINS   0.4683988
```

```
##
## Call:
## lm(formula = TARGET_WINS ~ TEAM_PITCHING_H + TEAM_BATTING_H +
##     TEAM_BATTING_BB + TEAM_PITCHING_BB, data = train_imputed)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -59.728  -8.963   0.394   9.085  65.012
##
## Coefficients:
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)          -6.052105   3.788542  -1.597  0.11030
## TEAM_PITCHING_HTwo    0.092667   0.955466   0.097  0.92275
## TEAM_PITCHING_HThree -0.034295   1.005300  -0.034  0.97279
## TEAM_PITCHING_HFour  -3.315098   1.095721  -3.025  0.00251 **
## TEAM_PITCHING_HFive  -1.341073   1.301417  -1.030  0.30290
## TEAM_BATTING_H        0.049052   0.002722  18.018  < 2e-16 ***
## TEAM_BATTING_BB       0.036671   0.003172  11.563  < 2e-16 ***
## TEAM_PITCHING_BB     -0.004882   0.002201  -2.218  0.02667 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 13.87 on 2268 degrees of freedom
## Multiple R-squared:  0.2275, Adjusted R-squared:  0.2251
## F-statistic: 95.41 on 7 and 2268 DF,  p-value: < 2.2e-16
```

In the Model #2 output above, we notice the following:

- All 3 continuous variables are statistically significant, but this time the intercept is not.

- Only 1 of the 4 TEAM_PITCHING_H dummy variables is statistically significant.
- We would expect a negative coefficient for Walks Allowed and positive ones for Batting Walks & Batting Hits. However, 1 of the 4 Hits Allowed dummy variables does not have a negative coefficient, and this unexpected coefficient sign may indicate that we still have some collinearity.
- At 0.2251, the adjusted R^2 indicates that this model explains less of variance in the response variable than Model #1.
- At 95, the F-statistic is larger than model #1, and the model's p-value is near zero. If the model's diagnostics are sufficient, these values indicate that we would reject the null hypothesis that there is no relationship between the explanatory & response variables.

Model 3: Backwards Elimination

- Let's start with Model #1, but let's remove the 4 variables that appeared to have multicollinearity issues because of their unexpected coefficient signs.
- Then we will remove one variable at a time by greatest coefficient p-value until we have a parsimonious model with only statistically significant variables.
- Along the way, we will take note if the removal of a variable causes a significant changes in adjusted R-squared and coefficient estimates.
- Lastly, TEAM_PITCHING_SO was removed because it didn't have the expected sign.

```
##
## Call:
## lm(formula = TARGET_WINS ~ TEAM_BATTING_H + TEAM_BATTING_HR +
##     TEAM_BATTING_SO + TEAM_BASERUN_SB + TEAM_FIELDING_E, data = train_imputed)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -52.551  -8.678   0.128   8.514  53.122
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)     25.629114   4.237642   6.048 1.71e-09 ***
## TEAM_BATTING_H   0.039267   0.002512  15.631  < 2e-16 ***
## TEAM_BATTING_HR  0.071958   0.008755   8.219 3.41e-16 ***
## TEAM_BATTING_SO -0.014925   0.002190  -6.814 1.21e-11 ***
## TEAM_BASERUN_SB  0.066361   0.003767  17.616  < 2e-16 ***
## TEAM_FIELDING_E -0.030775   0.001665 -18.482  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 12.93 on 2270 degrees of freedom


## Multiple R-squared:  0.3273, Adjusted R-squared:  0.3258
## F-statistic: 220.9 on 5 and 2270 DF,  p-value: < 2.2e-16
```

In the Model #3 output above, we notice the following:

- All 5 continuous variables & and the intercept are statistically significant. Their p-values are near zero.
- Unlike the previous models, we do not have any unexpected coefficient signs, which means that we have removed at least some of the multilinearity.
- At 0.3258, the adjusted R^2 indicates that this model explains 33% of the variance in the response variable.

- At 221, the F-statistic is larger than the other 2 models, and the model's p-value is near zero. If the model's diagnostics are sufficient, these values indicate that we would reject the null hypothesis that there is no relationship between the explanatory & response variables.

4. SELECT MODEL

Evaluation

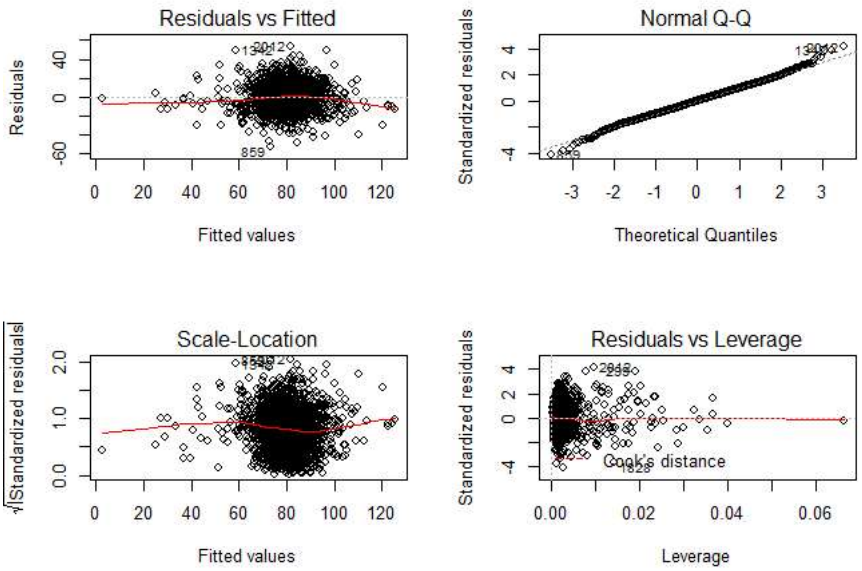
The table below summarizes the 3 models.

- While Model #1 has the highest adjusted R-squared, it also had 5 variables that were not statistically significant, and 4 variables with unexpected signs, which indicated multicollinearity issues.
- Model #2 used fewer variables, but the intercept coefficient was not statistically significant. We also saw some evidence of collinearity.
- **Of the 3, Model #3 is the best.** Removing the non-significant and collinear variables makes it more parsimonious with an adjusted R-squared that is less than Model #1 but much greater than Model #2. This model's F-statistic is also significantly larger than the other 2. Additionally, the difference between Model #3's predicted R-squared and R-squared is smaller than the other models, giving us more confidence that the model is not overfit.

model_name	n_predictors	pred.r.squared	r.squared	adj.r.squared	sigma	statistic	p.value
mod1	14	0.3326	0.3602	0.3554	12.6474	74.7676	0
mod2	4	0.2146	0.2275	0.2251	13.8663	95.4145	0
mod3	5	0.3225	0.3273	0.3258	12.9336	220.9222	0

Residual Plots

- In the Q-Q plot, we see some deviations for the normal distribution at the ends. Some of the extreme cases have been labeled. However, as the LMR text suggests, perhaps we can relax the normality assumption since we have more than 2200 observations.
- In the residuals-fitted and standardzied residuals-fitted plots, the best fitted line has a curve in the main cluster, which indicates that we do not have constant variance.
- Ideally, prior to making any inferences, we would want to attempt to transform the data in order to better meet the regression assumptions.



Test Model

Explore

Let's load the data, and look at our `NA` rates.

vars	class_type	n_rows	complete_cases	NA_ct	NA_pct	unique_value_ct	most_
TEAM_BATTING_H	integer	259	19	0	0.0	199	1420;
TEAM_BATTING_2B	integer	259	19	0	0.0	137	218; 2
TEAM_BATTING_3B	integer	259	19	0	0.0	91	36; 39
TEAM_BATTING_HR	integer	259	19	0	0.0	141	130; 2
TEAM_BATTING_BB	integer	259	19	0	0.0	185	506; 5
TEAM_BATTING_SO	integer	259	19	18	6.9	207	NA's; 6
TEAM_BASERUN_SB	integer	259	19	13	5.0	154	NA's; 7
TEAM_BASERUN_CS	integer	259	19	87	33.6	69	NA's; 5
TEAM_BATTING_HBP	integer	259	19	240	92.7	18	NA's; 6
TEAM_PITCHING_H	integer	259	19	0	0.0	210	1450;
TEAM_PITCHING_HR	integer	259	19	0	0.0	144	62; 11
TEAM_PITCHING_BB	integer	259	19	0	0.0	199	521; 4
TEAM_PITCHING_SO	integer	259	19	18	6.9	206	NA's; 5
TEAM_FIELDING_E	integer	259	19	0	0.0	168	130; 1
TEAM_FIELDING_DP	integer	259	19	31	12.0	91	NA's; 1 157

Transform & Impute Missing Values

We see similar patterns of missing values, so let's do the following:

- bin TEAM_PITCHING_H
- drop the sparsely populated TEAM_BATTING_HBP
- and impute the missing values

Since we have fewer cases, the NRMSE values are a little higher with the test data.

```
##   missForest iteration 1 in progress...done!
##   missForest iteration 2 in progress...done!
##   missForest iteration 3 in progress...done!
##   missForest iteration 4 in progress...done!
```

variable	NA_ct	NA_pct	MSE	RMSE	NRMSE
TEAM_FIELDING_DP	31	11.97	345.79	18.60	0.14
TEAM_BASERUN_CS	87	33.59	298.59	17.28	0.11
TEAM_BASERUN_SB	13	5.02	2924.34	54.08	0.09
TEAM_BATTING_SO	18	6.95	7168.02	84.66	0.07
TEAM_PITCHING_SO	18	6.95	377291.26	614.24	0.06

Predict

```
test_results <- predict(mod3, newdata = test_imputed)
```

Code Appendix

```
knitr::opts_chunk$set(
                      error = FALSE
                      ,message = FALSE
                      #,tidy = TRUE
                      ,cache = TRUE
                      )
#required packages
packages <- c("prettydoc", "psych", "knitr", "tidyverse", "ggthemes", "corrplot", "Hmisc", "data.tabl
#, "pastecs"

#see if we need to install any of them
installed_and_loaded <- function(pkg){
  #CODE SOURCE: https://gist.github.com/stevenworthington/3178163
  new.pkg <- pkg[!(pkg %in% installed.packages()[, "Package"])]
  if (length(new.pkg)) install.packages(new.pkg, dependencies = TRUE)
  sapply(pkg, require, character.only = TRUE, quietly = TRUE, warn.conflicts = FALSE)
}

#excute function and display the loaded packages
data.frame(installed_and_loaded(packages))
train_raw <- read.csv("https://raw.githubusercontent.com/kylegilde/D621-Data-Mining/master/HW1%20Mone

train_raw <- train_raw %>% select(-INDEX)

test_raw <- dplyr::select(read.csv("https://raw.githubusercontent.com/kylegilde/D621-Data-Mining/mast

set.seed(5)
```

```
metadata <- function(df){
  #Takes a data frame & Checks NAs, class types, inspects the unique values
  df_len <- nrow(df)
  NA_ct = as.vector(rapply(df, function(x) sum(is.na(x))))

  #create dataframe
  df_metadata <- data.frame(
    vars = names(df),
    class_type = rapply(lapply(df, class), function(x) x[[1]]),
    n_rows = rapply(df, length),
    complete_cases = sum(complete.cases(df)),
    NA_ct = NA_ct,
    NA_pct = NA_ct / df_len * 100,
    unique_value_ct = rapply(df, function(x) length(unique(x))),
    most_common_values = rapply(df, function(x) str_replace(paste(names(sort(summary(as.factor(x)),
  )
 rownames(df_metadata) <- NULL
 return(df_metadata)
}

meta_df <- metadata(train_raw)


kable(meta_df, digits = 1)
metrics <- function(df){
  ###Creates summary metrics table
  metrics_only <- df[, which(rapply(lapply(df, class), function(x) x[[1]]) %in% c("numeric", "intege

  df_metrics <- psych::describe(metrics_only, quant = c(.25,.75))

  df_metrics <-
    dplyr::select(df_metrics, n, min, Q.1st = Q0.25, median, mean, Q.3rd = Q0.75,
    max, range, sd, se, skew, kurtosis
  )

  return(df_metrics)
}

metrics_df <- metrics(train_raw)

kable(metrics_df, digits = 1, format.args = list(big.mark = ',', scientific = F, drop0trailing = T))
#calculate some parameters to deal with the outliers
train_stacked <- na.omit(stack(train_raw))
bpstats <- boxplot(values ~ ind, data = train_stacked, plot = F)$stats
ylimits <- c(0, ceiling(max(bpstats) / 200)) * 200
ybreaks <- seq(ylimits[1], ylimits[2], by = 200)
outliers_not_shown <- paste(sum(train_stacked$values > max(ylimits)), "outlier(s) not displayed")

ggplot(data = train_stacked, mapping = aes(x = ind, y = values)) +
  geom_boxplot(outlier.size = 1) +
  labs(caption = paste("Red dot = mean", outliers_not_shown, sep = "\n")) +
  scale_x_discrete(limits = rev(levels(train_stacked$ind))) +
  scale_y_continuous(breaks = ybreaks) +
  stat_summary(fun.y=mean, geom="point", size=2, color = "red") +
  coord_flip(ylim = ylimits) +
  theme_fivethirtyeight()


hist.data.frame(train_raw)
cormatrix <- cor(drop_na(train_raw))

#find the top correlations
cor_df <- data.frame(Var1=rownames(cormatrix)[row(cormatrix)],
```

```
                    Var2=colnames(cormatrix)[col(cormatrix)],
                    Correlation=c(cormatrix))


corr_list <-
  cor_df %>%
  filter(Var1 != Var2) %>%
  arrange(-Correlation)

#dedupe the rows
sort_rows <- t(apply(corr_list, 1, sort, decreasing = T))
fin_list <- corr_list[!duplicated(sort_rows), ]
rownames(fin_list) <- 1:nrow(fin_list)
#print table
kable(head(fin_list, 12), digits=4, row.names = T, caption = "Top 12 Correlated Variable Pairs")

#https://stackoverflow.com/questions/28035001/transform-correlation-matrix-into-dataframe-with-recor

#plot
corrplot(cormatrix, method = "square", type = "upper")

train_raw$TEAM_PITCHING_H <- bin_data(train_raw$TEAM_PITCHING_H, bins = 5, binType = "quantile")

levels(train_raw$TEAM_PITCHING_H) <- c("One","Two","Three","Four","Five")

impute_missing <- missForest(train_raw, variablewise = T)

# check imputation error
impute_df <- cbind(meta_df,
                   range = metrics_df$range,
                   MSE = as.numeric(impute_missing$OOBerror),
                   variable = names(impute_missing$ximp)) %>%
  select(variable, NA_ct, NA_pct, MSE) %>%
  mutate(RMSE = sqrt(as.numeric(impute_missing$OOBerror)),
         NRMSE = sqrt(as.numeric(impute_missing$OOBerror))/metrics_df$range) %>%
  filter(MSE > 0) %>%
  arrange(-NRMSE)

kable(impute_df, digits = 2)
train_raw_less_one <- select(train_raw, -TEAM_BATTING_HBP)
impute_missing <- missForest(train_raw_less_one, variablewise = T)

# check imputation error
impute_df <- cbind(metadata(train_raw_less_one),
                   range = rapply(impute_missing$ximp, max) - rapply(impute_missing$ximp, min),
                   MSE = as.numeric(impute_missing$OOBerror),
                   variable = names(impute_missing$ximp)) %>%
  select(variable, NA_ct, NA_pct, MSE) %>%
  mutate(RMSE = sqrt(as.numeric(impute_missing$OOBerror)),
         NRMSE = sqrt(as.numeric(impute_missing$OOBerror))/(rapply(impute_missing$ximp, max) - rapply
  filter(MSE > 0) %>%
  arrange(-NRMSE)

kable(impute_df, digits = 2)

train_imputed <- impute_missing$ximp
class(train_imputed$TEAM_PITCHING_H) <- "factor"

mod1 <- lm(TARGET_WINS ~ ., data = train_imputed)

summary(mod1)
summary(aov(TARGET_WINS~TEAM_PITCHING_H, data = train_imputed))
filter(fin_list, Var2 == "TARGET_WINS")[1:4,]
```

```
mod2 <- lm(TARGET_WINS ~ TEAM_PITCHING_H + TEAM_BATTING_H + TEAM_BATTING_BB + TEAM_PITCHING_BB, data

summary(mod2)

mod3 <- update(mod1, .~. -TEAM_FIELDING_DP -TEAM_BATTING_2B -TEAM_BASERUN_CS -TEAM_PITCHING_HR, data
#summary(mod3) #R-sq = 0.334

#remove TEAM_PITCHING_BB at 0.598258
mod3 <- update(mod3, .~. -TEAM_PITCHING_BB, data = train_imputed)
#summary(mod3) #new R-sq = 0.3342

#remove TEAM_BATTING_BB at 0.169551
mod3 <- update(mod3, .~. -TEAM_BATTING_BB, data = train_imputed)
#summary(mod3) #new R-sq = 0.3341

#remove TEAM_PITCHING_H
mod3 <- update(mod3, .~. -TEAM_PITCHING_H, data = train_imputed)
#summary(mod3) #new R-sq = 0.3284

#remove TEAM_PITCHING_SO, it appears collinear
mod3 <- update(mod3, .~. -TEAM_PITCHING_SO, data = train_imputed)
#summary(mod3) #new R-sq = 0.3273

#remove TEAM_BATTING_3B,
mod3 <- update(mod3, .~. -TEAM_BATTING_3B, data = train_imputed)
summary(mod3) #new R-sq = 0.3273


PRESS <- function(linear.model) {
  #source:  https://gist.github.com/tomhopper/8c204d978c4a0cbcb8c0#file-press-r
  #' calculate the predictive residuals
  pr <- residuals(linear.model)/(1-lm.influence(linear.model)$hat)
  #' calculate the PRESS
  PRESS <- sum(pr^2)

  return(PRESS)
}
pred_r_squared <- function(linear.model) {
  #source: https://gist.github.com/tomhopper/8c204d978c4a0cbcb8c0#file-pred_r_squared-r
  #' Use anova() to get the sum of squares for the linear model
  lm.anova <- anova(linear.model)
  #' Calculate the total sum of squares
  tss <- sum(lm.anova$'Sum Sq')
  # Calculate the predictive R^2
  pred.r.squared <- 1-PRESS(linear.model)/(tss)

  return(pred.r.squared)
}


model_summary <- function(model, y_var) {
    ### Summarizes the model's key statistics in one row
    df_summary <- glance(summary(model))
    model_name <- deparse(substitute(model))
    n_predictors <- ncol(model$model) - 1
    pred.r.squared <- pred_r_squared(model)
    df_summary <- cbind(model_name, n_predictors, pred.r.squared, df_summary)
    return(df_summary)
}
mod_sum_df1 <- model_summary(mod1, "TARGET_WINS")
mod_sum_df2 <- model_summary(mod2, "TARGET_WINS")
mod_sum_df3 <- model_summary(mod3, "TARGET_WINS")
```

```
kable(all_results <- rbind(mod_sum_df1, mod_sum_df2, mod_sum_df3), digits = 4)
par(mfrow=c(2,2))
plot(mod3)
kable(metadata(test_raw), digits = 1)

test_raw$TEAM_PITCHING_H <- bin_data(test_raw$TEAM_PITCHING_H, bins = 5, binType = "quantile")

levels(test_raw$TEAM_PITCHING_H) <- c("One","Two","Three","Four","Five")

test_raw_less_one <- select(test_raw, -TEAM_BATTING_HBP)

impute_missing_test <- missForest(test_raw_less_one, variablewise = T)

# check imputation error
impute_df <- cbind(metadata(test_raw_less_one),
                   range = rapply(impute_missing_test$ximp, max) - rapply(impute_missing_test$ximp, r
                   MSE = as.numeric(impute_missing_test$OOBerror),
                   variable = names(impute_missing_test$ximp)
                   ) %>%
  select(variable, NA_ct, NA_pct, MSE) %>%
  mutate(RMSE = sqrt(as.numeric(impute_missing_test$OOBerror)),
         NRMSE = sqrt(as.numeric(impute_missing_test$OOBerror))/(rapply(impute_missing_test$ximp, ma
  filter(MSE > 0) %>%
  arrange(-NRMSE)

kable(impute_df, digits = 2)

test_imputed <- impute_missing$ximp
class(test_imputed$TEAM_PITCHING_H) <- "factor"
test_results <- predict(mod3, newdata = test_imputed)
```