Project #1

Introduction

In this report we examine professional baseball team data with the ultimate intent to build OLS regression models to describe baseball team performance, wins. To accomplish this, we performed exploratory data analysis, selected a sample population of typical baseball seasons, derived new variables, trained multiple regression models using different variable selection methods, evaluated in-sample and out-of-sample error measures as well as evidence of multicollinearity in the regressors, and finally selected the model which was most satisfactory.

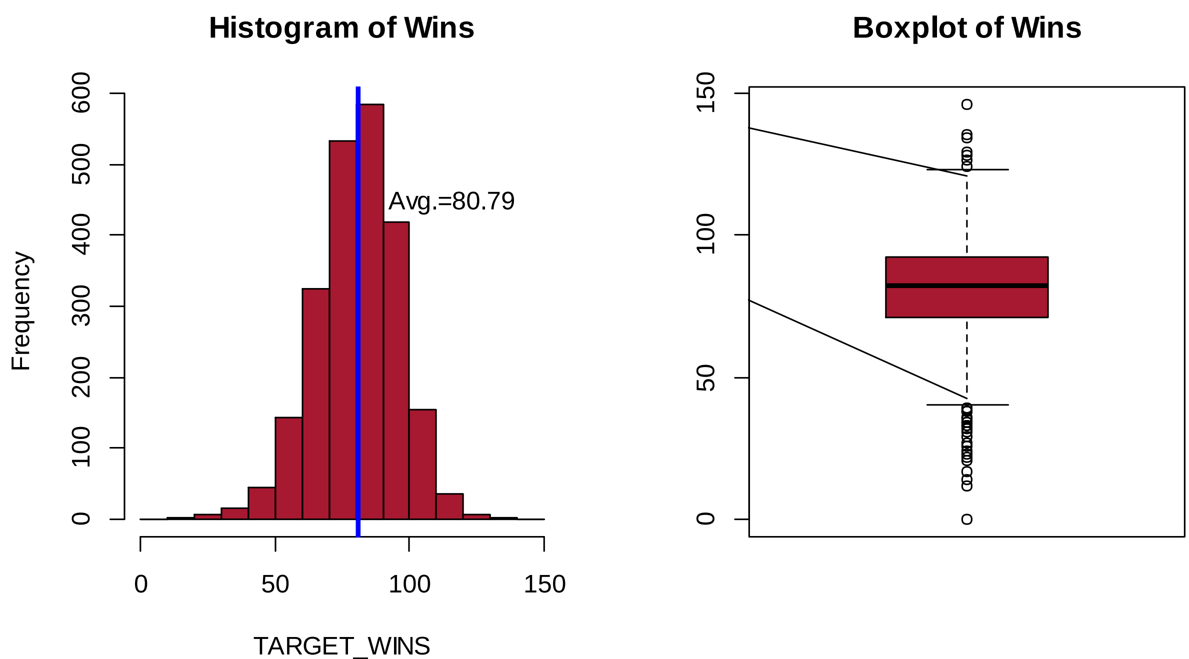
Three linear regression models were built using different variable selection methods. The best simple linear regression model only explained ~15% of the variation in wins, where both multiple linear regressions explained ~30%. The all regressors method yielded the best fit in terms of least MSE for the training data however, this model had signs of severe multicollinearity in its regressors which may explain the overfitting of the model that led to poorer testing error measures. The parsimonious model seemed to be the most balanced across in-sample and out-of-sample, and had fewer regressors to interpret and would be considered the leader of the models.

Data Exploration

Our source data set contains 2,276 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. This data set contains 17 variables, all of which are integers. The target variable of interest is TARGET_WINS ("wins"), which provides for a given team in a given year, how many games out of 162 they won. The independent variables are a mixture of both theoretical positive and negative drivers of wins.

Analyzing our response “wins”. As a first step in data exploration, we observe the distribution of wins in Figure 1. We observe the response variable is centered at an average of 80.79 wins with a slight skew to the left (skewness= -0.40) and higher peak (kurtosis= 4.03) than that of the normal (Gaussian) distribution. However, we will proceed with the assumption that our response is continuous and normally distributed.

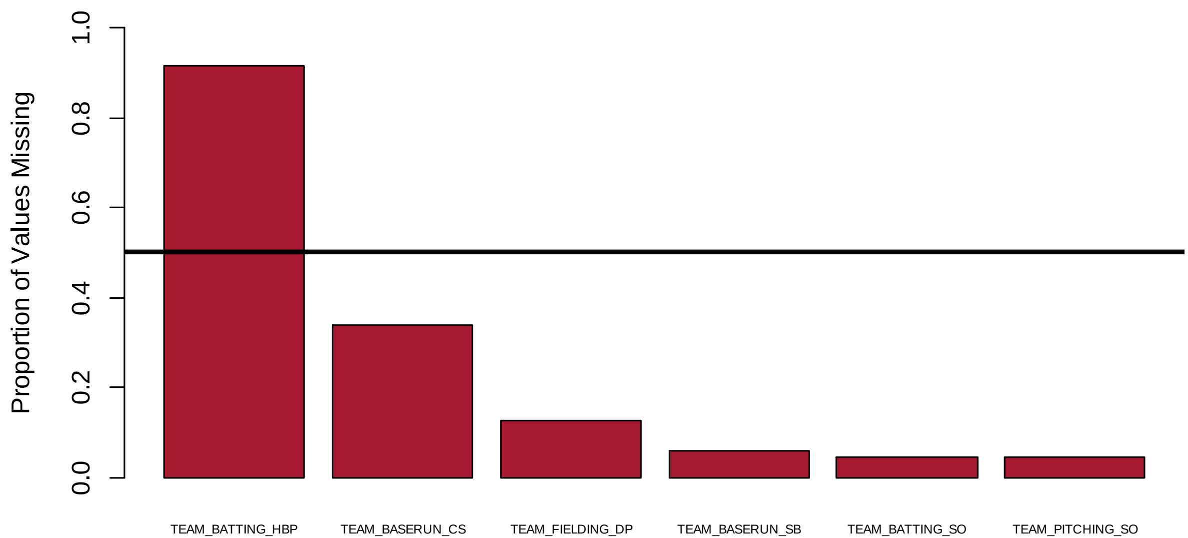
Figure 1: Distribution of wins



In the Figure 1 boxplot, we also note there are several data points that exceed the whiskers, this suggests there are outlier seasons based on wins that we may wish to consider removing as they might impede out model fit.

Assessing data completeness. As a first step to analyzing our explanatory variables, we check for the completeness of the data in Figure 2. Here we see that 6 of our 15 possible predictors contain missing values. TEAM_BATTING_HBP, the number of batters hit by a pitch, has missing values for nearly 92% of the records. The lack of completeness for this variable makes it a candidate for removal from the dataset. The other predictors have at least 50% completeness, and will be candidates for imputing missing values.

Figure 2: Missing values in predictors



Predictor relationships with wins. There are theoretical relationships between our predictors and our response variable, wins. Table 1 displays the hypothesized relationships between the predictors and wins, as well as the Pearson correlation coefficient so explore these relationships. We observe that for 5 of the 15 possible predictors, there is a mismatch in the sign of the correlation coefficient and the theoretical effect, namely: homeruns allowed, walks allowed, caught stealing, double plays, and strikeouts by pitchers show an opposite relationship from what is believed to be the effect on wins. We will track these relationships in our models to see how they relate to the driver variables' theoretical effects.

Table 1: Theoretical effects and Pearson correlations with wins

Driver Variable	Definition	Theoretical Effect	Pearson Coefficient	Mismatch
TEAM_BATTING_H	Base Hits by batters (1B,2B,3B,HR)	Positive Impact on Wins	0.389	
TEAM_BATTING_2B	Doubles by batters (2B)	Positive Impact on Wins	0.289	
TEAM_BATTING_BB	Walks by batters	Positive Impact on Wins	0.233	
TEAM_PITCHING_HR	Homeruns allowed	Negative Impact on Wins	0.189	X
TEAM_BATTING_HR	Homeruns by batters (4B)	Positive Impact on Wins	0.176	
TEAM_BATTING_3B	Triples by batters (3B)	Positive Impact on Wins	0.143	
TEAM_BASERUN_SB	Stolen bases	Positive Impact on Wins	0.135	
TEAM_PITCHING_BB	Walks allowed	Negative Impact on Wins	0.124	X
TEAM_BATTING_HBP	Batters hit by pitch (get a free base)	Positive Impact on Wins	0.074	
TEAM_BASERUN_CS	Caught stealing	Negative Impact on Wins	0.022	X
TEAM_BATTING_SO	Strikeouts by batters	Negative Impact on Wins	-0.032	
TEAM_FIELDING_DP	Double Plays	Positive Impact on Wins	-0.035	X
TEAM_PITCHING_SO	Strikeouts by pitchers	Positive Impact on Wins	-0.078	X
TEAM_PITCHING_H	Hits allowed	Negative Impact on Wins	-0.110	
TEAM_FIELDING_E	Errors	Negative Impact on Wins	-0.176	

Data Preparation

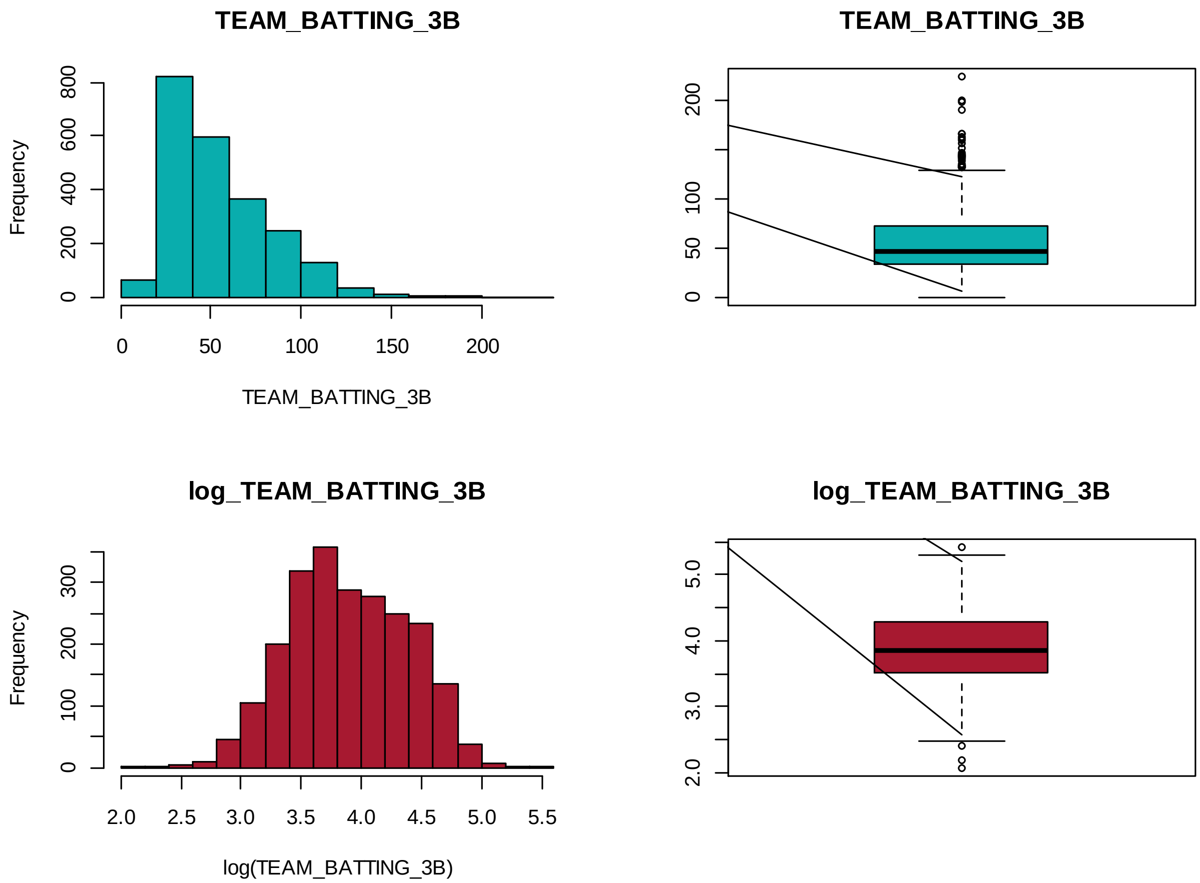
In this section we discuss the transformations performed on our data set, which includes imputing missing values for certain variables, log and square-root transformations to straighten variable distributions, creation of some potentially useful variables, and finally sub setting our data to remove outlier records from our modeling data set.

Impute missing values. As seen in Figure 2, we have 6 variables which contain missing values. To handle these variables we will replace them with their respective mean value, with the exception of TEAM_BATTING_HBP, which we will remove from our data set due to its severe lack of completeness. Although missing indicator values might yield a more accurate model, in this case we will not create indicator variables for missing values since they will not support the business objective, which is to understand how baseball statistical drivers impact wins.

Variable creations. Two additional variables were developed to potentially provide additional explanation to wins. First a variable for base hits was created, “TEAM_BATTING_1B”, by taking total hits and subtracting out home runs, triples and doubles. Second, a variable to represent stolen base success percentage was created, “SB_PCT”, by taking dividing total stolen bases by total attempted stolen bases or, TEAM_BASERUN_SB+ TEAM_BASERUN_CS.

Variable transformations. Through exploratory analysis it was discovered for a select set of predictor distributions that they did not show desirable Gaussian-like distributions. E.g. In Figure 3 we show the before and after log transformation for the variable TEAM_BATTING_3B. In similar fashion we derived log transformations on: TEAM_BATTING_1B, TEAM_BASERUN_SB, and TEAM_BASERUN_CS. We also performed a square root transformation on TEAM_PITCHING_HR to create a more desirable distribution.

Figure 3: Example transformation – Log transformation on triples by batters



Variable value adjustments. Additionally a couple variables showed values high on their distributions, namely: TEAM_FIELDING_E and TEAM_PITCHING_SO had their values capped at 500 and 5,000 respectively so that there would not be too much divergence from the average of their distribution, which can potentially create issues when fitting our models.

Creating our modeling data set. As mentioned with Figure 1, there are records that deviate from the norm and can cause problems in our model fitting. To account for this, in Table 2 we describe the drop conditions used to create our sample data set we to model from.

Table 2: Waterfall of conditions to select sample data set for modeling

Drop_Condition	Frequency	Percent
01: Irregular win total	21	0.92
02: Extreme number of hits allowed	11	0.48
03: Sample Population	2244	98.59

Model Creation

In this section we describe the creation of three linear regression models, using different approaches. First we build a model with a single predictor that has the highest correlation with our response, wins. Next we build a model using all possible predictors. Finally, we attempt to build a parsimonious model that seeks to serve the business understanding.

Model 1: Best Simple Linear Regression Model

This model answers the question if we could only use one baseball figure, how much could we say about baseball wins? From Table 1, we saw that the predictor with the highest correlation to wins was base hits by batters (1B,2B,3B,HR), "TEAM_BATTING_H". We fit a simple linear regression of TARGET_WINS ~ TEAM_BATTING_H. We see from the output in Figure 4 that the F value is statistically significant at the $p < 0.05$ level, indicating that our model_simple has predictive power in explaining the variability in TARGET_WINS. In addition we see the R-squared value of 0.1457, which indicates that 14.57% of the variability in TARGET_WINS is explained by the TEAM_BATTING_H. We note that the single regressor TEAM_BATTING_H is statistically significant at the $p < 0.05$ level. Plugging the parameter estimates from Figure 4 into the simple linear regression model yields:

$$model\_simple: TARGET\_WINS = 19.733574 + 0.041740 * TEAM\_BATTING\_H$$

Figure 4: Best simple linear regression model (model_simple)

```
Call:
lm(formula = TARGET_WINS ~ TEAM_BATTING_H, data = moneyball2)
```

```
Residuals:
    Min      1Q  Median      3Q     Max 
-57.482  -8.790   0.668   9.505  39.796 

Coefficients:
                Estimate Std. Error t value Pr(>|t|)    
(Intercept)    19.733574   3.143981   6.277 4.14e-10 ***
TEAM_BATTING_H  0.041740   0.002134  19.557  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 13.62 on 2242 degrees of freedom
Multiple R-squared:  0.1457,	Adjusted R-squared:  0.1454 
F-statistic: 382.5 on 1 and 2242 DF,  p-value: < 2.2e-16
```

This model aligns with theoretical effect that base hits by batters has a positive effect on wins.

Model 2: All Predictors Regression Model

Next we consider all possible predictors to fit a multiple linear regression model to wins. This model answers the question if we threw everything we had into the model, what kind of results would return. We see from the output in Figure 5 that the F value is statistically significant at the $p < 0.05$ level, indicating that our model_all has predictive power in explaining the variability in TARGET_WINS. In addition we see the R-squared value of 0.3083, which indicates that 30.83% of the variability in TARGET_WINS is explained by the model_all regressors.

Figure 5: All predictors overall model results (model_all)

```
Call:
lm(formula = TARGET_WINS ~ . - TEAM_BATTING_1B, data = moneyball2)

Residuals:
    Min      1Q  Median      3Q     Max 
-52.160  -8.148   0.164   7.861  47.833 

Residual standard error: 12.31 on 2223 degrees of freedom
Multiple R-squared:  0.3083,	Adjusted R-squared:  0.3021 
F-statistic: 49.54 on 20 and 2223 DF,  p-value: < 2.2e-16
```

Note: TEAM_BATTING_1B is excluded from the model since it is a linear combination of 3 other predictors.

In Figure 6, we observe the results for the individual regressors. We note that 9 of the 20 regressors are statistically significant at the $p < 0.05$ level. Plugging the parameter estimates from Figure 6 into the multiple linear regression model yields:

$$
\begin{aligned}
model\_all: TARGET\_WINS \\
&= -1.934999e+02 + 3.647428e+01 * log\_TEAM\_BASERUN\_CS \\
&- 4.094005e+01 * log\_TEAM\_BASERUN\_SB + 2.823224e+01 \\
&* log\_TEAM\_BATTING\_1B - 6.238451e-01 * log\_TEAM\_BATTING\_3B \\
&+ 1.686558e+02 * SB\_PCT - 5.797343e-01 \\
&* sqrt\_TEAM\_PITCHING\_HR - 3.677270e-02 * TEAM\_BASERUN\_CS \\
&+ 9.800566e-02 * TEAM\_BASERUN\_SB + 2.503545e-03 \\
&* TEAM\_BATTING\_2B + 1.573213e-01 * TEAM\_BATTING\_3B \\
&+ 2.453788e-02 * TEAM\_BATTING\_BB + 8.863799e-03 \\
&* TEAM\_BATTING\_H + 7.632586e-02 * TEAM\_BATTING\_HR \\
&- 1.465469e-02 * TEAM\_BATTING\_SO - 1.085427e-01 \\
&* TEAM\_FIELDING\_DP - 5.113062e-02 * TEAM\_FIELDING\_E \\
&- 8.303566e-03 * TEAM\_PITCHING\_BB + 2.014600e-03 \\
&* TEAM\_PITCHING\_H + 5.197448e-02 * TEAM\_PITCHING\_HR \\
&+ 2.063505e-03 * TEAM\_PITCHING\_SO
\end{aligned}
$$

We further observe several predictors with evidence of severe multicollinearity (VIF>10). This suggests our model most likely has misleading coefficient estimates in sign and magnitude.

Figure 6: All predictors regressor results (model_all)

variable	Estimate	Pr...t..	vif	P<0.05	VIF>10
(Intercept)	-1.934999e+02	2.951214e-01	NA		<NA>
log_TEAM_BASERUN_CS	3.647428e+01	3.374958e-03	246.667234	*	MC
log_TEAM_BASERUN_SB	-4.094005e+01	2.351439e-03	1038.001464	*	MC

```
   log_TEAM_BATTING_1B  2.823224e+01 3.571645e-01  154.199479              MC
   log_TEAM_BATTING_3B -6.238451e-01 7.765013e-01   17.031070              MC
                SB_PCT  1.686558e+02 9.966923e-04  497.434455        *     MC
 sqrt_TEAM_PITCHING_HR -5.797343e-01 2.515762e-01   38.985957              MC
       TEAM_BASERUN_CS -3.677270e-02 3.939937e-01    9.633189
       TEAM_BASERUN_SB  9.800566e-02 1.684044e-05   55.240423        *     MC
       TEAM_BATTING_2B  2.503545e-03 9.298707e-01   25.257410              MC
       TEAM_BATTING_3B  1.573213e-01 2.890768e-03   30.475477        *     MC
       TEAM_BATTING_BB  2.453788e-02 3.348083e-03   14.019696        *     MC
        TEAM_BATTING_H  8.863799e-03 7.525402e-01  212.424125              MC
       TEAM_BATTING_HR  7.632586e-02 7.378871e-02   97.494858              MC
       TEAM_BATTING_SO -1.465469e-02 1.333022e-03   17.070370        *     MC
      TEAM_FIELDING_DP -1.085427e-01 8.271192e-17    1.498038        *
       TEAM_FIELDING_E -5.113062e-02 2.451084e-26    5.202718        *
      TEAM_PITCHING_BB -8.303566e-03 2.174500e-01   11.120766              MC
       TEAM_PITCHING_H  2.014600e-03 8.601525e-02    8.127305
      TEAM_PITCHING_HR  5.197448e-02 1.886695e-01   85.520313              MC
      TEAM_PITCHING_SO  2.063505e-03 5.072671e-01   10.910118              MC
```

For example, we see that log_TEAM_BASERUN_SB, a theoretical positive impact on wins has returned a significant negative coefficient. This kind of model result would be difficult to explain to the business. Similarly, we see opposite results from expected for:

log_TEAM_BASERUN_CS, log_TEAM_BATTING_3B, TEAM_FIELDING_DP, TEAM_PITCHING_H, TEAM_PITCHING_HR.

Model 3: Parsimonious Regression Model

This final model attempts to provide the greatest clarity to the business, potentially sacrificing accuracy for a model that is easier to interpret and is more parsimonious. Prior to fitting such a model we need to take steps to ensure we remove likelihood of multicollinearity, which as we observed in the model_all, can create problems in our interpretation of the model coefficients. To do this we identify groups of predictors that are highly correlated with one another, we'll use Pearson correlation coefficient of 0.7 or greater. Figure 7 displays these groups of predictors. The highest correlated predictor with wins from each group was selected for inclusion in the model.

Figure 7: Highly correlated predictors

```
$`1`
[1] "log_TEAM_BATTING_1B" "TEAM_BATTING_H"

$`2`
[1] "sqrt_TEAM_PITCHING_HR" "TEAM_BATTING_HR"

$`3`
[1] "TEAM_PITCHING_BB" "TEAM_BATTING_BB"

$`4`
[1] "TEAM_PITCHING_SO" "TEAM_BATTING_SO"

$`5`
[1] "log_TEAM_BASERUN_SB" "SB_PCT"
```

In addition, the following variables were excluded from the model as they were reproduced with log or square root transformations: "TEAM_BATTING_1B", "TEAM_BATTING_3B", "TEAM_BASERUN_SB", "TEAM_BASERUN_CS", "TEAM_PITCHING_HR". We next employ a stepwise F-test method for variable selection. We see from the output in Figure 8 that the F value is statistically significant at the $p < 0.05$ level, indicating that our model_stepwise has predictive power in explaining the variability in TARGET_WINS. In addition we see the R-squared value of 0.2915, which indicates that 29.15% of the variability in TARGET_WINS is explained by the model_stepwise regressors.

Figure 8: Parsimonious model results (model_stepwise)

```
Call:
lm(formula = TARGET_WINS ~ TEAM_BATTING_H + TEAM_BATTING_HR +
    TEAM_BATTING_BB + TEAM_PITCHING_SO + SB_PCT + TEAM_BATTING_2B +
    TEAM_FIELDING_E + TEAM_FIELDING_DP + log_TEAM_BATTING_3B +
    log_TEAM_BASERUN_CS + TEAM_PITCHING_H, data = moneyball3)

Residuals:
    Min      1Q  Median      3Q     Max
-52.671  -8.108   0.089   8.300  46.087

Residual standard error: 12.43 on 2232 degrees of freedom
Multiple R-squared:  0.2915,   Adjusted R-squared:  0.288
F-statistic: 83.48 on 11 and 2232 DF,  p-value: < 2.2e-16
```

In Figure 9, we observe the results for the individual regressors. We note that all 11 of the variables were selected for inclusion by the stepwise method with all 11regressors found to be statistically significant at the $p < 0.05$ level. Plugging the parameter estimates from Figure 9 into the multiple linear regression model yields:

$$
\begin{aligned}
model\_stepwise: TARGET\_WINS \sim &- 15.609918869 + 2.428889761 \\
&* log\_TEAM\_BASERUN\_CS + 6.766976869\, log\_TEAM\_BATTING\_3B \\
&+ 27.388885755 * SB\_PCT - 0.030872646 * TEAM\_BATTING\_2B \\
&+ 0.020348380 * TEAM\_BATTING\_BB + 0.038741791 \\
&* TEAM\_BATTING\_H + 0.063340013 * TEAM\_BATTING\_HR \\
&- 0.102864749 * TEAM\_FIELDING\_DP - 0.038749390 \\
&* TEAM\_FIELDING\_E + 0.002383322 * TEAM\_PITCHING\_H \\
&- 0.005164621 * TEAM\_PITCHING\_SO
\end{aligned}
$$

Unlike our previous model, Figure 9 shows there are no concerns of multicollinearity in the parsimonious model, model_stepwise.

Figure 9: Parsimonious model regressor results (model_stepwise)

variable	Estimate	Pr...t..	vif	P<0.05	VIF>10
(Intercept)	-15.609918869	1.281408e-02	NA	*	<NA>
log_TEAM_BASERUN_CS	2.428889761	6.208916e-03	1.230625	*	
log_TEAM_BATTING_3B	6.766976869	4.565734e-13	2.986684	*	
SB_PCT	27.388885755	1.887177e-20	1.594557	*	
TEAM_BATTING_2B	-0.030872646	6.743000e-04	2.516283	*	
TEAM_BATTING_BB	0.020348380	2.845898e-11	1.822388	*	
TEAM_BATTING_H	0.038741791	4.818899e-18	5.187782	*	
TEAM_BATTING_HR	0.063340013	5.432037e-12	4.383125	*	
TEAM_FIELDING_DP	-0.102864749	6.927148e-16	1.405044	*	
TEAM_FIELDING_E	-0.038749390	2.733311e-20	3.901924	*	
TEAM_PITCHING_H	0.002383322	1.583065e-03	3.287914	*	
TEAM_PITCHING_SO	-0.005164621	4.768029e-04	2.407274	*	

Although we've seemed to solve the problem of multicollinearity, we still see variables exhibiting opposite relationships with wins than expected. The regressors: log_TEAM_BASERUN_CS, TEAM_BATTING_2B, TEAM_FIELDING_DP, TEAM_PITCHING_H, TEAM_PITCHING_SO all exhibited contradictory coefficient estimate signs from the theoretical effect. A possible explanation for this occurrence may be attributable to interaction effects that were not explored. For example hits allowed showed as positive impact on wins however theoretically it should be negative. This might be that more often than not teams that garner wins more so from batting hits rather than their pitching allowed hits.

Model Selection

With our three models built, we need to select one to move forward with and use for making business decisions. An important feature for the selected model, will be its ability to make predictions for wins on new data. To simulate how our models will perform, we will calculate out-of-sample error for each model using 10-fold cross validation sets, and evaluating with the Mean Squared Error ("MSE"). In figure 10 we see how our models stack up against one another both on the model training sample and the 10-fold hold out samples.

Table 3: Comparison of In-sample and out-of-sample MSEs for the models

Model	In-Sample MSE	K=10 Fold Out-of-Sample MSE
model_simple	185.3	185.6
model_all	150.0	155.1
model_stepwise	153.7	156.3

From Table 3, we see our model_all with all predictors yields the lowest in-sample MSE, however looking at the out-of-sample MSEs the model_stepwise is nearly as low as model_all. This means these models are expected to perform similarly on new data. Since our model_stepwise is a more robust and more interpretable model, we would recommend moving forward with the model_stepwise model for business use.

Conclusions

This report examined baseball season data for the purpose of building regression models to describe baseball team wins. Three linear regression models were built using different variable selection methods. The best simple linear regression model only explained ~15% of the variation in wins, where both multiple linear regressions explained ~30%. The all regressors method yielded the best fit in terms of least MSE for the training data however, this model had signs of severe multicollinearity in its regressors which may explain the overfitting of the model that led to poorer testing error measures. The parsimonious model, model_stepwise, seemed to be the most balanced across in-sample and out-of-sample, and had fewer regressors to interpret.

Additional considerations may be applied in future study of the baseball season data. For one, additional variable transformations for the both the response and the regressors can be considered in the model to describe wins. Rigorous outlier analysis may also be considered to further improve the model for sale price. Also, developing additional predictors that might remove confusion in the predictor estimates directions should be evaluated.

Code

```
###########################################################################
#
#			title: Project #1
#			author: STUDENT NAME
#			info: corresponds to file Project_1.docx
#
###########################################################################

######## Load necessary packages
library(readr)
library(pbkrtest)
library(car)
library(leaps)
library(MASS)
```

```
library(openxlsx)
library(moments)
library(boot)
library(igraph)

#####
#Designated proper working environment
#####

setwd("C:/Users/NAME/OneDrive/MSPA/PREDICT411/Project 1")
moneyball=read.csv("moneyball.csv",header=T)



############## Part 1: Data Exploration ###############################

str(moneyball)
summary(moneyball)

############## Analyze response variable: TARGET_WINS

# Missing values in Y?
sum(is.na(moneyball$TARGET_WINS))

# Distribution of Y
mean(moneyball$TARGET_WINS)
skewness(moneyball$TARGET_WINS); kurtosis(moneyball$TARGET_WINS)
# slightly skewed to the left and higher peaked at the center of the
# distribution

# Distribution plot of Y
par(mfrow=c(1,2))
hist(moneyball$TARGET_WINS, col = "#A71930", xlab = "TARGET_WINS",
    main = "Histogram of Wins")
abline(v=mean(mean(moneyball$TARGET_WINS)),lwd=3,col="blue")
text(x=mean(moneyball$TARGET_WINS),y=450,
    labels = paste0("Avg.=",round(mean(moneyball$TARGET_WINS),2)),
    pos = 4,offset = 1)
boxplot(moneyball$TARGET_WINS, col ="#A71930", main ="Boxplot of Wins")
par(mfrow = c(1,1))

# Outliers in Y?
quantile(x = moneyball$TARGET_WINS, probs = c(0.005, 0.01,0.05,0.5,0.95,
                                    0.99,0.995))
# few values below 31 and few values above 120



############## Analyze predictor variables:

# Data completeness
prop.miss <- function(x){sum(is.na(x))/length(x)}
```

```
miss.summary <- apply(moneyball[,3:length(names(moneyball))],MARGIN = 2,
                 FUN = prop.miss)
miss.summary <- sort(miss.summary,decreasing = T)
miss.summary[miss.summary > 0]
barplot(miss.summary[miss.summary > 0],cex.names = 0.5, ylim = c(0,1),
      ylab = "Proportion of Values Missing",col ="#A71930")
abline(h = 0.5, col = "black", lwd = 3)

# Predictor distributions
for(i in 3:length(names(moneyball))){
      par(mfrow=c(1,2))
      hist(moneyball[,i], col = "#09ADAD",xlab=names(moneyball)[i],
          main=names(moneyball)[i])
      boxplot(moneyball[,i], col = "#09ADAD",main=names(moneyball)[i])
      par(mfrow=c(1,1))
}

######## Correlation Matrix vs. Theoretical Impacts ##########

data.dictionary<-read.xlsx("DataDictionary_Baseball.xlsx")
data.d <- data.dictionary[-(1:2),]

cor.matrix <- data.frame(cor(moneyball, use = "pairwise.complete.obs"))
cor.matrix <- data.frame(VARIABLE = names(cor.matrix),
                   Pearson.coef = as.vector(t(cor.matrix[2,]))
                   )
cor.matrix <- cor.matrix[-(1:2),]
theory <- merge(data.d,cor.matrix,by.x="VARIABLE.NAME",by.y="VARIABLE")
names(theory) <- c("Driver","Definition","Theoretical.effect",
               "Pearson.coefficient")
theory = theory[order(theory$Pearson.coefficient, decreasing = T),]
theory$Pearson.coefficient = round(theory$Pearson.coefficient,3)
theory$Mismatch = ifelse(
      theory$Theoretical.effect == "Positive Impact on Wins" &
            sign(theory$Pearson.coefficient)== -1, "X",
      ifelse(theory$Theoretical.effect == "Negative Impact on Wins" &
                  sign(theory$Pearson.coefficient)== 1, "X",""))
write.csv(theory,"TABLE_1_theory_table.csv",row.names = F)
theory




############## Part 2: Data Preparation ################################

#Fix Missing Values Using Mean of All Seasons
moneyball$TEAM_BATTING_SO[is.na(moneyball$TEAM_BATTING_SO)] =
      mean(moneyball$TEAM_BATTING_SO, na.rm = TRUE)
moneyball$TEAM_BASERUN_SB[is.na(moneyball$TEAM_BASERUN_SB)] =
      mean(moneyball$TEAM_BASERUN_SB, na.rm = TRUE)
moneyball$TEAM_BASERUN_CS[is.na(moneyball$TEAM_BASERUN_CS)] =
      mean(moneyball$TEAM_BASERUN_CS, na.rm = TRUE)
```

```
moneyball$TEAM_FIELDING_DP[is.na(moneyball$TEAM_FIELDING_DP)] =
      mean(moneyball$TEAM_FIELDING_DP, na.rm = TRUE)
moneyball$TEAM_PITCHING_SO[is.na(moneyball$TEAM_PITCHING_SO)] =
      mean(moneyball$TEAM_PITCHING_SO, na.rm = TRUE)

#Straighten Relationships

# Define new variables
moneyball$TEAM_BATTING_1B <- moneyball$TEAM_BATTING_H -
      moneyball$TEAM_BATTING_HR - moneyball$TEAM_BATTING_3B -
      moneyball$TEAM_BATTING_2B
moneyball$SB_PCT <- moneyball$TEAM_BASERUN_SB /
      (1.0*moneyball$TEAM_BASERUN_SB+moneyball$TEAM_BASERUN_CS+0.00001)
# Transform variables
moneyball$log_TEAM_BATTING_1B <- log(moneyball$TEAM_BATTING_1B)
moneyball$log_TEAM_BATTING_3B <- log(moneyball$TEAM_BATTING_3B)
moneyball$log_TEAM_BASERUN_SB <- log(moneyball$TEAM_BASERUN_SB)
moneyball$log_TEAM_BASERUN_CS <- log(moneyball$TEAM_BASERUN_CS)
moneyball$sqrt_TEAM_PITCHING_HR <- sqrt(moneyball$TEAM_PITCHING_HR)
# Cap extreme divergance values
moneyball$TEAM_FIELDING_E[(moneyball$TEAM_FIELDING_E > 500)] = 500
moneyball$TEAM_PITCHING_SO[(moneyball$TEAM_PITCHING_SO > 5000)] = 5000

#Check that na's are gone.
summary(moneyball)

# Before and After Transformations: Example of TEAM_BATTING_3B
par(mfrow=c(2,2))
hist(moneyball$TEAM_BATTING_3B, col = "#09ADAD",
    main="TEAM_BATTING_3B", xlab = "TEAM_BATTING_3B")
boxplot(moneyball$TEAM_BATTING_3B, col = "#09ADAD",
      main="TEAM_BATTING_3B")
hist(moneyball$log_TEAM_BATTING_3B, col = "#A71930",
    main="log_TEAM_BATTING_3B", xlab = "log(TEAM_BATTING_3B)")
boxplot(moneyball$log_TEAM_BATTING_3B, col = "#A71930",
      main="log_TEAM_BATTING_3B")
par(mfrow=c(1,1))

#Remove bad data from data set
moneyball$Drop_Condition <- ifelse(
      moneyball$TARGET_WINS < 31 | moneyball$TARGET_WINS > 120,
            "01: Irregular win total",
      ifelse(moneyball$TEAM_PITCHING_H > 10000,
            "02: Extreme number of hits allowed",
            "03: Sample Population"))

drop.table <- data.frame(table(moneyball$Drop_Condition))
names(drop.table) <- c("Drop_Condition","Frequency")
drop.table$Percent <- round(100*(drop.table$Frequency/
                        sum(drop.table$Frequency)),2)
write.csv(drop.table,"TABLE_2_Drop_Conditions.csv",row.names = F)

moneyball2 <- moneyball[moneyball$Drop_Condition ==
```

```
                "03: Sample Population",
            -c(1,11,25)] #remove index, HBP & drop_condition



############## Part 3: Model Creation ##################################

#Function for Mean Square Error Calculation
mse <- function(sm){mean(sm$residuals^2)}


############## Best Single Linear Regression Model

# Identify Predictor with strongest correlation to TARGET_WINS
cor.matrix <- data.frame(cor(moneyball2))
cor.vector <- as.vector(t(cor.matrix[1,]))
names(cor.vector) <- names(cor.matrix)
cor.vector <- sort(cor.vector)
cor.vector <- cor.vector[1:(length(cor.vector)-1)]
par(mfrow=c(1,1),mar=c(4,8,1,1))
barplot(cor.vector,horiz = T,las=1,cex.names = 0.5,col="#A71930",
    xlim = c(-0.2,0.4))

# Create Single Linear Regression Model
model_simple <- lm(TARGET_WINS ~ TEAM_BATTING_H, data=moneyball2)
summary(model_simple)





############## All Predictors Multiple Regression Model
model_all <- lm(TARGET_WINS ~ . - TEAM_BATTING_1B, data=moneyball2)
summary.model_all <- summary(model_all)
coef.model_all <- data.frame(summary.model_all$coefficients)
coef.model_all$variable <- row.names(coef.model_all)
vif.model_all <- vif(model_all)
vif.model_all <- data.frame(variable = names(vif.model_all),
                vif=vif.model_all)

eval.model_all <- merge(coef.model_all[,c(1,4,5)],vif.model_all,
            by="variable",all.x = T)
eval.model_all$`P<0.05` <- ifelse(eval.model_all$Pr...t..<0.05,"*","")
eval.model_all$`VIF>10` <- ifelse(eval.model_all$vif>10,"MC","")

model_all <- glm(TARGET_WINS ~ . - TEAM_BATTING_1B, data=moneyball2)

print(eval.model_all,row.names=F)
print(eval.model_all[,c(2,1)],row.names = F)
```

```
############## Parsimonious Model

# Drop predictors that were transformed
drop <- c("TEAM_BATTING_1B","TEAM_BATTING_3B","TEAM_BASERUN_SB",
        "TEAM_BASERUN_CS","TEAM_PITCHING_HR")
moneyball3 <- moneyball2[,!(names(moneyball2) %in% drop)]

# Select representative predictor from highly correlated group of
# predictors
var.corelation <- cor(as.matrix(moneyball3), method="pearson")
# prevent duplicated pairs
var.corelation <- var.corelation*lower.tri(var.corelation)
check.corelation <- which(var.corelation>0.7, arr.ind=TRUE)
graph.cor <- graph.data.frame(check.corelation, directed = FALSE)
groups.cor <- split(unique(as.vector(check.corelation)),
                clusters(graph.cor)$membership)
lapply(groups.cor,FUN=function(list.cor){rownames(var.corelation)[list.cor]})

# Create stepwise model
stepwisemodel <- lm(TARGET_WINS ~ TEAM_BATTING_H + TEAM_BATTING_HR +
                TEAM_BATTING_BB + TEAM_PITCHING_SO + SB_PCT +
                TEAM_BATTING_2B +
                TEAM_FIELDING_E + TEAM_FIELDING_DP +
                log_TEAM_BATTING_3B + log_TEAM_BASERUN_CS +
                TEAM_PITCHING_H
                , data = moneyball3)
model_stepwise <- step(stepwisemodel, direction = "both", test="F")

summary.model_stepwise <- summary(model_stepwise)
coef.model_stepwise <- data.frame(summary.model_stepwise$coefficients)
coef.model_stepwise$variable <- row.names(coef.model_stepwise)
vif.model_stepwise <- vif(model_stepwise)
vif.model_stepwise <- data.frame(variable = names(vif.model_stepwise),
                          vif=vif.model_stepwise)

eval.model_stepwise <- merge(coef.model_stepwise[,c(1,4,5)],
                       vif.model_stepwise,by="variable",all.x = T)
eval.model_stepwise$`P<0.05` <- ifelse(eval.model_stepwise$Pr...t..<0.05,"*","")
eval.model_stepwise$`VIF>10` <- ifelse(eval.model_stepwise$vif>10,"MC","")

model_stepwise <- glm(TARGET_WINS ~ . - TEAM_BATTING_1B, data=moneyball2)

print(eval.model_stepwise,row.names=F)
print(eval.model_stepwise[,c(2,1)],row.names = F)




############## Part 4: Model Selection ##################################
```

```
model_simple <- glm(TARGET_WINS ~ TEAM_BATTING_H, data=moneyball2)

model_all <- glm(TARGET_WINS ~ . - TEAM_BATTING_1B, data=moneyball2)

stepwisemodel <- glm(TARGET_WINS ~ TEAM_BATTING_H + TEAM_BATTING_HR +
                    TEAM_BATTING_BB + TEAM_PITCHING_SO + SB_PCT
                    + TEAM_BATTING_2B +
                    TEAM_FIELDING_E + TEAM_FIELDING_DP +
                    log_TEAM_BATTING_3B + log_TEAM_BASERUN_CS +
                    TEAM_PITCHING_H
               ,data = moneyball3)
model_stepwise <- step(stepwisemodel, direction = "both", test="F")

models <-list(model_simple,model_all,model_stepwise)
df <- data.frame()
for(i in models){
    #train.aic <- AIC(i)
    train.mse <- mse(i)
    set.seed(1234)
    k10_fold_test.mse <- cv.glm(moneyball2, i, K=10)$delta[1]
    mod <- cbind(train.mse,k10_fold_test.mse)
    df <- rbind(df,mod)
}
row.names(df) <- c("model_simple","model_all","model_stepwise")
write.csv(df,"TABLE_3_Model_Selection.csv")




############## Test Data ################################################

setwd("C:/Users/NAME/OneDrive/MSPA/PREDICT411/Project 1")
moneyball_test=read.csv("moneyball_test.csv",header=T)

# Fixing na's
moneyball_test$TEAM_BATTING_SO[is.na(moneyball_test$TEAM_BATTING_SO)] =
    mean(moneyball_test$TEAM_BATTING_SO, na.rm = TRUE)
moneyball_test$TEAM_BASERUN_SB[is.na(moneyball_test$TEAM_BASERUN_SB)] =
    mean(moneyball_test$TEAM_BASERUN_SB, na.rm = TRUE)
moneyball_test$TEAM_BASERUN_CS[is.na(moneyball_test$TEAM_BASERUN_CS)] =
    mean(moneyball_test$TEAM_BASERUN_CS, na.rm = TRUE)
moneyball_test$TEAM_FIELDING_DP[is.na(moneyball_test$TEAM_FIELDING_DP)] =
    mean(moneyball_test$TEAM_FIELDING_DP, na.rm = TRUE)
moneyball_test$TEAM_PITCHING_SO[is.na(moneyball_test$TEAM_PITCHING_SO)] =
    mean(moneyball_test$TEAM_PITCHING_SO, na.rm = TRUE)
moneyball_test$TEAM_BASERUN_CS[moneyball_test$TEAM_BASERUN_CS < 1] = 1
moneyball_test$log_TEAM_BASERUN_CS <- log(moneyball_test$TEAM_BASERUN_CS)
moneyball_test$log_TEAM_BATTING_3B <- log(moneyball_test$TEAM_BATTING_3B)
moneyball_test$SB_PCT <- moneyball_test$TEAM_BASERUN_SB/
    (1.0*moneyball_test$TEAM_BASERUN_SB+moneyball_test$TEAM_BASERUN_CS)
moneyball_test$SB_PCT[is.na(moneyball_test$SB_PCT)] =
    mean(moneyball_test$SB_PCT)
```

```
# Stand Alone Scoring
moneyball_test$P_TARGET_WINS <- -15.609918869 +
      2.428889761* moneyball_test$log_TEAM_BASERUN_CS +
      6.766976869* moneyball_test$log_TEAM_BATTING_3B +
      27.388885755* moneyball_test$SB_PCT -
      0.030872646* moneyball_test$TEAM_BATTING_2B +
      0.020348380* moneyball_test$TEAM_BATTING_BB +
      0.038741791* moneyball_test$TEAM_BATTING_H +
      0.063340013* moneyball_test$TEAM_BATTING_HR -
      0.102864749* moneyball_test$TEAM_FIELDING_DP -
      0.038749390* moneyball_test$TEAM_FIELDING_E +
      0.002383322* moneyball_test$TEAM_PITCHING_H -
      0.005164621* moneyball_test$TEAM_PITCHING_SO

#subset of data set for the deliverable "Scored data file"
prediction <- moneyball_test[c("INDEX","P_TARGET_WINS")]

# Prediction file
write.csv(x = prediction, file = "Project_1_Prediction.csv",
        row.names = F)
```