Predict 411 Unit1 Moneyball

Jennifer Wanat

Contents

Introduction

In this project our objective is to be able to provide an ordinary least squares (OLS) regression model to predict the number of wins for a professional baseball team. The data set contains records of the team for a given year, ranging from 1871 through 2006. In order to build a model to provide a prediction of team wins, an exploratory data analysis (EDA) of the data set was completed prior to this analysis, which allowed for an evaluation and selection of the most promising predictor variables.

The data set was randomly split into a training set and test set to allow for cross-validation of the predictive models. Automated variable selection techniques utilized the data set for model identification. The generated model summaries and parameters were provided, and assessed for predictive accuracy by examing model selection criteria. The predictive accuracy of each model was assessed with the test data set. All models were assigned prediction grades for the training and test data sets. Then a model was selected, and used to score new data and predict the number of target wins. The analysis was conducted with the R programming language.

Data

The data was entered in an Excel spreadsheet and saved as a comma separated value file (.csv). The .csv file was opened in R and saved into a data frame called `moneyball`. The data frame contains 2276 integer observations of 17 variables measured between the years of 1871 through 2006. Refer to the data dictionary for variable definition and theoretical effect (positive or negative impact) on team wins. The variable *INDEX* is an identification variable and was not used for modeling purposes. An EDA was conducted on the remaining 16 variables.

Table 1: Statistical Summary of Variables.

	vars	n	mean	sd	median	trimmed	mad	min	max	range	skew	kurtosis	se
TARGET_WINS	1	2276	80.8	15.8	82.0	81.3	14.8	0	146	146	-0.4	1.0	0.3
TEAM_BATTING_H	2	2276	1469.3	144.6	1454.0	1459.0	114.2	891	2554	1663	1.6	7.3	3.0
TEAM_BATTING_2B	3	2276	241.2	46.8	238.0	240.4	47.4	69	458	389	0.2	0.0	1.0
TEAM_BATTING_3B	4	2276	55.2	27.9	47.0	52.2	23.7	0	223	223	1.1	1.5	0.6
TEAM_BATTING_HR	5	2276	99.6	60.5	102.0	97.4	78.6	0	264	264	0.2	-1.0	1.3
TEAM_BATTING_BB	6	2276	501.6	122.7	512.0	512.2	94.9	0	878	878	-1.0	2.2	2.6
TEAM_BATTING_SO	7	2174	735.6	248.5	750.0	742.3	284.7	0	1399	1399	-0.3	-0.3	5.3
TEAM_BASERUN_SB	8	2145	124.8	87.8	101.0	110.8	60.8	0	697	697	2.0	5.5	1.9
TEAM_BASERUN_CS	9	1504	52.8	23.0	49.0	50.4	17.8	0	201	201	2.0	7.6	0.6
TEAM_BATTING_HBP	10	191	59.4	13.0	58.0	58.9	11.9	29	95	66	0.3	-0.1	0.9
TEAM_PITCHING_H	11	2276	1779.2	1406.8	1518.0	1555.9	174.9	1137	30132	28995	10.3	141.8	29.5
TEAM_PITCHING_HR	12	2276	105.7	61.3	107.0	103.2	74.1	0	343	343	0.3	-0.6	1.3
TEAM_PITCHING_BB	13	2276	553.0	166.4	536.5	542.6	98.6	0	3645	3645	6.7	97.0	3.5
TEAM_PITCHING_SO	14	2174	817.7	553.1	813.5	796.9	257.2	0	19278	19278	22.2	671.2	11.9
TEAM_FIELDING_E	15	2276	246.5	227.8	159.0	193.4	62.3	65	1898	1833	3.0	11.0	4.8
TEAM_FIELDING_DP	16	1990	146.4	26.2	149.0	147.6	23.7	52	228	176	-0.4	0.2	0.6

Section 1: Data Exploration

A basic statistical summary of the variables was prepared. The descriptive summary includes the number of observations (n), mean, standard deviation (sd), median, trimmed mean (trimmed), median absolute deviation from the mean (mad), minimum value (min), maximum value (max), the difference between the minimum and maximum values (range), skewness (skew), kurtosis, and standard error (se). Six variables were noted to have missing values (NA), as determined by a n value less than 2276, and were examined further. These variables were: TEAM_BATTING_SO, TEAM_BASERUN_SB, TEAM_BASERUN_CS, TEAM_BATTING_HBP, TEAM_PITCHING_SO, TEAM_FIELDING_DP.

The number and percentage of missing values for these variables is examined in the table below. TEAM_BATTING_SO, TEAM_BASERUN_SB, and TEAM_PITCHING_SO have six percent or less missing values. TEAM_FIELDING_DP has 13% and TEAM_BASERUN_CS has 34% missing values. TEAM_BATTING_HBP has 92% missing values. Given the high percentage of missing values, TEAM_BATTING_HBP will not be used as a predictive variable in the models. The missing values will need to be imputed in order to build an OLS model.

Table 2: Percentage of Missing Values for Effected Variables.

	Number of missing values	Percent of missing values
TEAM_BATTING_SO	102	4
TEAM_BASERUN_SB	131	6
TEAM_BASERUN_CS	772	34
TEAM_BATTING_HBP	2085	92
TEAM_PITCHING_SO	102	4
TEAM_FIELDING_DP	286	13

Additionally, it was noted there were a very wide range of values for the variables TEAM_PITCHING_H and TEAM_PITCHING_SO. Both of these variables had a skew greater than 10, which indicates a right-skewed distribution and that there were outliers greater than the mean. This is visually observed with a histogram and boxplot of both variables. The variable mean will be affected more by outliers than the median. A transformation of these two variables should be conducted to minimize or eliminate the effect of the outliers on the data, or there will be a negative impact on the accuracy of the model.

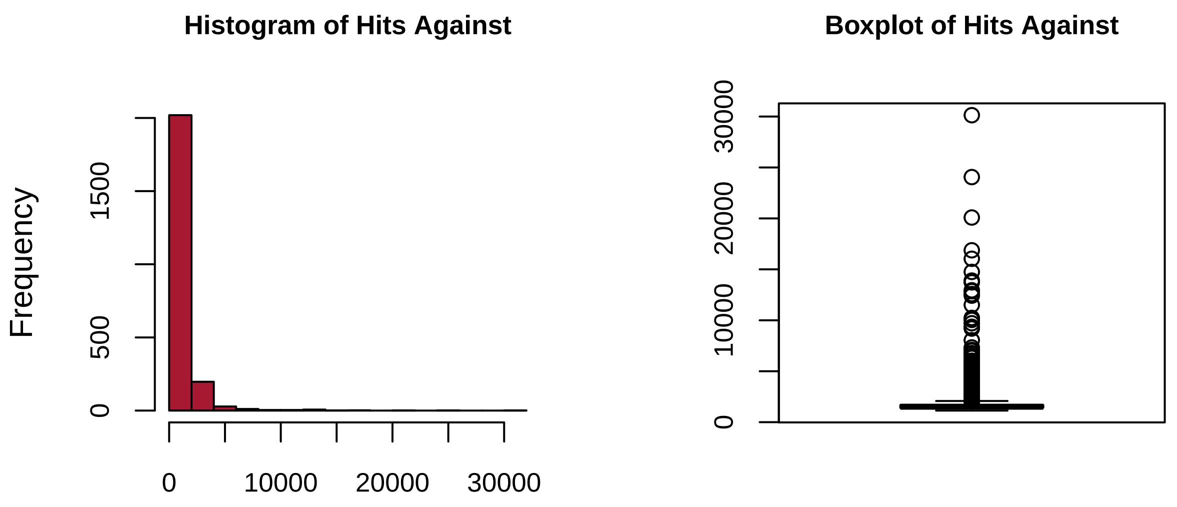


Figure 1: Visual Examination of Hits Against.

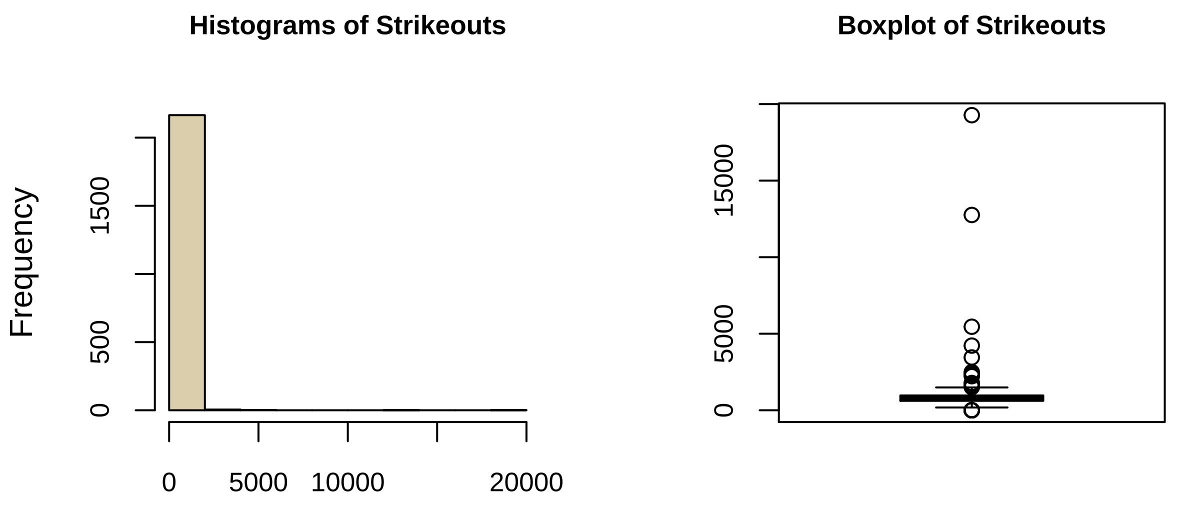


Figure 2: Visual Examination of Pitching Strikeouts.

A shadow matrix of the moneyball data set was created, which was generated with indicator variables using 1 for missing values and 0 for present values. The indicator variables for data with missing values is correlated with the original data to determine if there is a relationship between the missing values and variables.

TEAM_BATTING_SO and TEAM_PITCHING_SO have a strong positive correlation with a correlation coefficient of 1. These two variables tend to be missing values together. The correlation coefficient for TEAM_BASERUN_CS ranges from 0.22 to 0.48, which indicates that this variable tends to be missing values with the other variables that contain missing entries.

Further analysis of the variables that contained missing values included examination of the mode (most frequent value), quantiles, and the ten maximum and minimum values. This information can be useful to determine if the variables require a transformation, and what type of transformation should be used. Some variables had more than one mode. Utilizing the mode for a variable tranformation would likely not be useful.

Table 4: Quantiles of Variables of Interest.

	0%	1%	5%	10%	25%	50%	75%	90%	95%	99%	100%
TEAM_BATTING_SO	0	70.65	359.0	421.0	548.0	750.0	930.00	1049.0	1103.35	1192.27	1399
TEAM_BASERUN_SB	0	23.44	35.0	44.0	66.0	101.0	156.00	231.0	301.80	438.56	697
TEAM_BASERUN_CS	0	16.03	24.0	30.0	38.0	49.0	62.00	77.0	91.00	142.97	201
TEAM_BATTING_HBP	29	29.90	40.0	44.0	50.5	58.0	67.00	76.0	82.50	89.10	95
TEAM_PITCHING_SO	0	207.19	421.3	490.0	615.0	813.5	968.00	1095.0	1173.00	1466.70	19278
TEAM_FIELDING_DP	52	79.00	98.0	109.0	131.0	149.0	164.00	178.0	186.00	202.22	228
TEAM_PITCHING_H	1137	1244.00	1316.0	1356.0	1419.0	1518.0	1682.50	2057.5	2563.00	7054.00	30132
TEAM_PITCHING_BB	0	240.00	377.0	417.5	476.0	536.5	611.00	693.5	757.00	921.00	3645
TARGET_WINS	0	38.75	54.0	61.0	71.0	82.0	92.00	99.5	104.00	114.00	146
TEAM_FIELDING_E	65	86.00	100.0	109.0	127.0	159.0	249.25	542.0	716.00	1228.00	1898
TEAM_BATTING_H	891	1193.25	1281.5	1315.0	1383.0	1454.0	1537.25	1635.5	1695.25	1945.50	2554

[a] 0% = minimum, 25% = Q1, 50% = median, 75% = Q3, 100% = maximum

Quantile analysis divides the group of data into a specified number of equal parts. Q1 and Q3 are the first and third quartiles, and divide the data into the 25th and 75th percentiles, respectively. The median is also known as Q2 and is located at the 50th percentile. Together, the data between Q1 and Q3 represent the middle 50% of the data. Minimum is the smallest value in the data set and maximum is the largest value in the data set.

The maximum value of TEAM_PITCHING_SO is 19278 compared to the values of 1173 and 1467 at the 95th and 99th percentiles. The maximum value of TEAM_PITCHING_H is 30132 compared to the values of 2563 and 7054 at the 95th and 99th percentiles. If these variables are used for modeling purposes, then it is recommended that they are truncated so that the values do not exceed either the 95th or 99th percentiles. The ten maximum and minimum values were examined to observe the type of values at the very extremes of the data set (less than 1% of the total observations).

Table 3: Mode for Varibles with Missing Values.

TEAM_BATTING_SO	0			
TEAM_BASERUN_SB	65	69	73	100
TEAM_BASERUN_CS	52			
TEAM_BATTING_HBP	54	56	72	
TEAM_PITCHING_SO	0			
TEAM_FIELDING_DP	148	156		

Boxplots of the variables were created to help visualize the location of outliers. The two variables, TEAM_PITCHING_H and TEAM_PITCHING_SO, were previously examined and are not part of the boxplots below. Variables TEAM_BATTING_H, TEAM_PITCHING_BB, and TEAM_FIELDING_E all exhibited outliers that would benefit from a transformation such as truncating to the 1% and/or 99% quantile(s).

Finally, a correlation plot was constructed of variables that did not contain any missing values. It was noted that TEAM_PITCHING_HR and TEAM_BATTING_HR were strongly correlated with each other (r=0.97). Utilizing both of these variables in a model would cause multicollinearity issues. This would result in an increase in the variance of the regression coefficients, which would cause an increase in the standard errors of the coefficients. The regression model coefficients would not be as significant when you would expect them to be.

TEAM_BATTING_H was observed to be positively correlated with TEAM_BATTING_2B (r=0.56) and TEAM_BATTING_3B (r=0.43). This is expected as TEAM_BATTING_H is the total number of base hits. Predictive models will account for this colinearity by utilizing the individual variable components of base hits (singles, doubles, triples, and home runs).

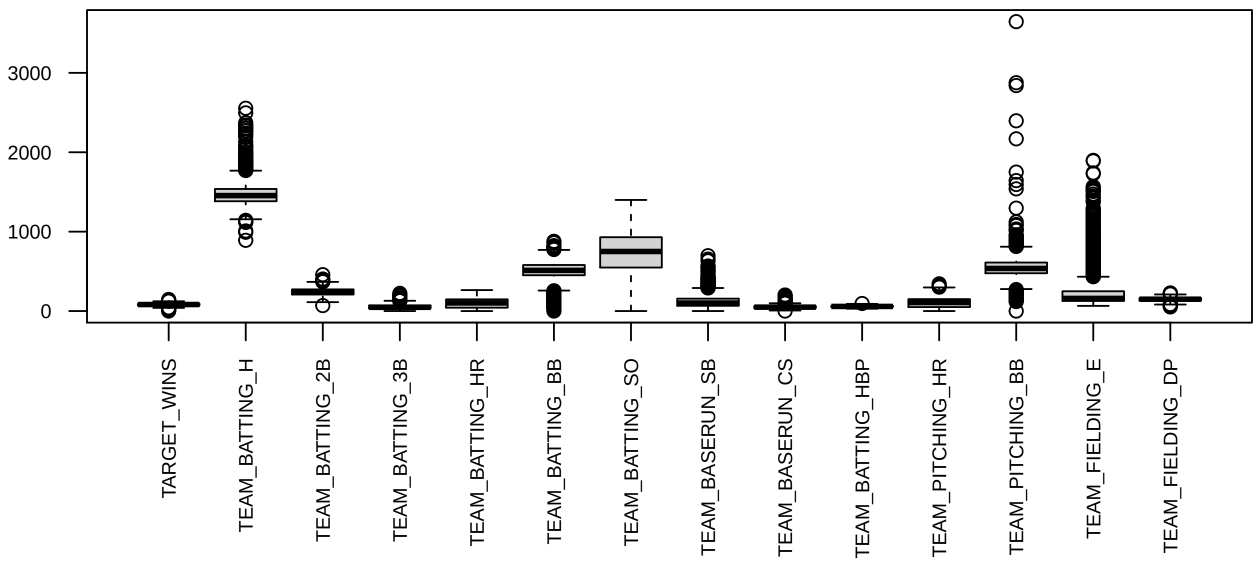
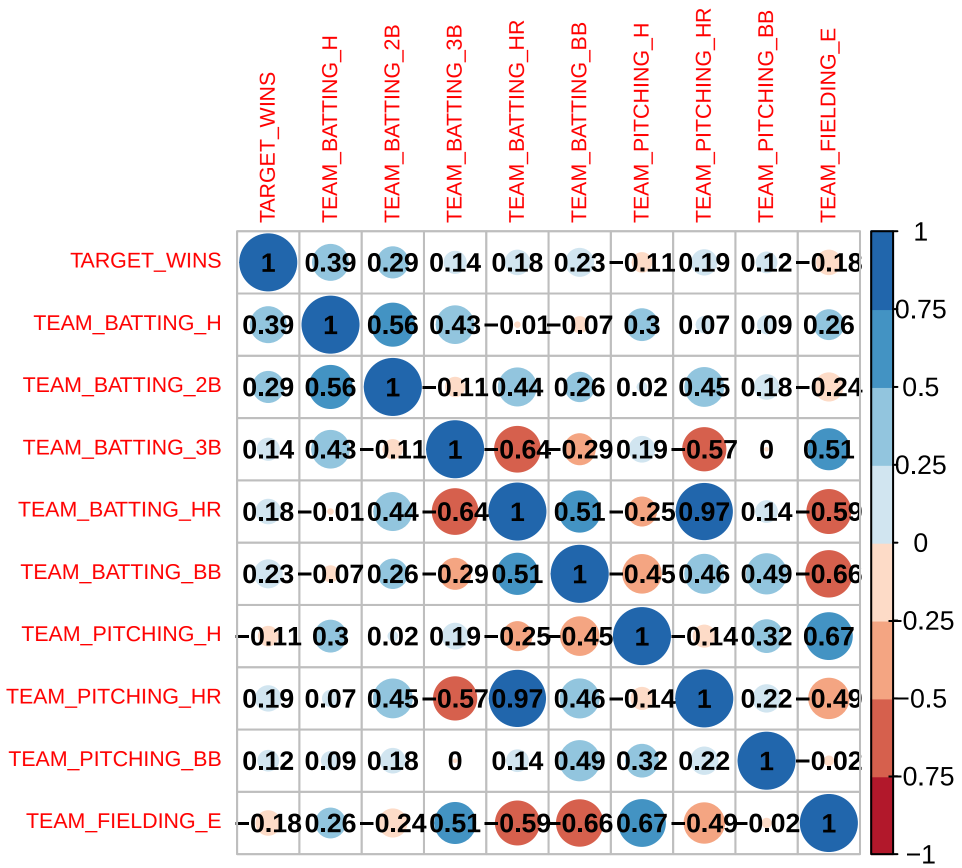


Figure 3: Boxplot of data set variables.

TEAM_PITCHING_H and TEAM_FIELDING_E were positively correlated with each other (r=0.67). Negative correlations were noted with TEAM_BATTING_HR and TEAM_BATTING_3B (r=-0.64), and TEAM_PITCHING_HR and TEAM_BATTING_3B (r=-0.57). TEAM_FIELDING_E had a negative correlation with both TEAM_BATTING_HR (r=-0.59) and TEAM_BATTING_BB (r=-0.66). These correlations would be monitored in the model building steps to minimize multicollinearity.



Section 2: Data Preparation

Before creating a predictive model, the data needed to be prepared. Missing values would need to be imputed, and variables with outliers would need a type of transformation in order to reduce the effect of the outlier on the model accuracy. After data preparation has been completed, then predictive models may be generated.

Section 2.1: Data Imputation

Variables with missing values were imputed with the `mice()` function in R. The `mice` (multivariate imputation by chained equations) package will fill in the missing values utilizing a specified imputation method and generate a specified number of completed data sets. The completed data set to be used for modeling can be specified (dependent upon the number generated), and stored for further tranformations and regression analysis. The imputation method selected for the *moneyball* data set was "pmm" (predictive mean matching). Five data sets would be generated, but only one would be used for modeling. The INDEX variable would be excluded from the imputation as it is only a numerical identification and should not be used as a predictor for the missing values. TEAM_BATTING_HBP will be exluded as 2085 of the 2276 observations (92%) are missing.

After imputation of the missing values, the data set was checked to confirm there were no longer any missing values. Then some of the variables were transformed to address outliers.

Table 5: Statistical Summary of Single Hits

	vars	n	mean	sd	median	trimmed	mad	min	max	range	skew	kurtosis	se
X1	1	2276	1073.2	128.9	1050	1059.1	99.3	709	2112	1403	2	8.5	2.7

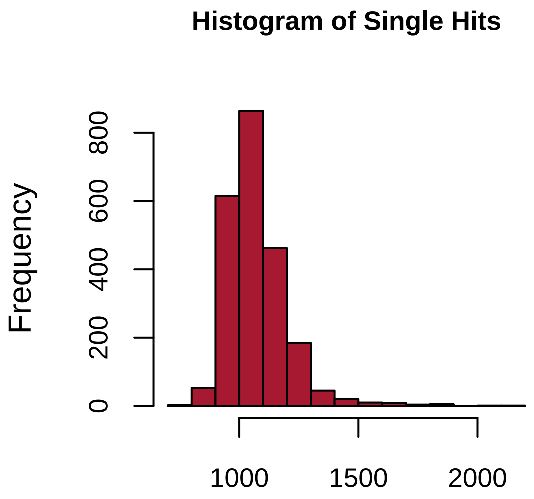
Table 6: Quantiles of Single Hits.

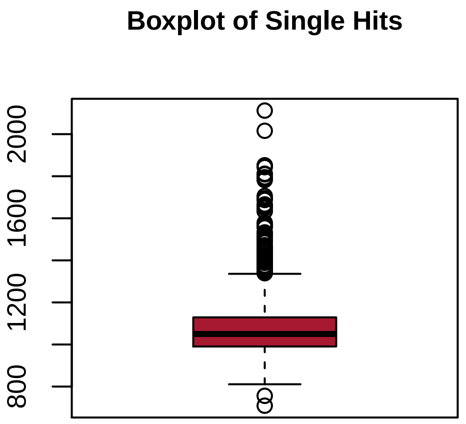
	0%	1%	5%	10%	25%	50%	75%	90%	95%	99%	100%
TEAM_BATTING_1B	709	881	919	945	990.75	1050	1129	1222.5	1282	1558.25	2112

Section 2.2: Outlier Transformation

There were variables for the total number of base hits by batters (TEAM_BATTING_H), the number of doubles (TEAM_BATTING_2B), the number of triples (TEAM_BATTING_3B), and the total number of home runs (TEAM_BATTING_HR). A new variable for the number of single hits, called TEAM_BATTING_1B, was created by subtracting the number of home runs, triples, and doubles from the total number of hits. This would allow for the variable TEAM_BATTING_H to not be used in the predictive model, due to the observed collinearity with the other hit variables.

The new variable structure, TEAM_BATTING_1B, was statistically examined. There were no missing values, although there were some outliers that would require a transformation.





The natural logarithm of the TEAM_BATTING_1B, TEAM_PITCHING_H, and TEAM_FIELDING_E variables was calculated utilizing the `log()` function in R. The natural logarithm tranformation can only be conducted on variables that do not contain the value of zero. The `log` transformed data was saved as a separate variable, so that the original data remained unchanged.

Five variables were selected to undergo a truncation transformation: TEAM_PITCHING_SO, TEAM_PITCHING_H, TEAM_PITCHING_BB, TEAM_FIELDING_E, and TEAM_BATTING_H. These variables were selected based upon the data distribution osberved in the EDA. These variables were truncated to either the 1% quantile (TEAM_PITCHING_SO and TEAM_PITCHING_BB) or 99% quantile (all five varibles listed).

The final variable transformation used was the z tranformation, in which the variable mean is subtracted from each data observation and then divided by the standard deviation. This was conducted for six variables: TEAM_BATTING_3B, TEAM_BASERUN_SB, TEAM_BASERUN_CS, TEAM_PITCHING_BB, TEAM_FIELDING_E, TEAM_BATTING_SO. The z transformed variable was saved as a separate variable, so that the original data remained unchanged.

Section 2.3: The Train/Test Split

In order to assess model performance, the sample population data set was split into a 70/30 train/test split. This would allow for cross-validation after a predictive model has been developed. The train data set would be used for in-sample model development and the test data set would be used for out-of-sample model assessment of predictive accuracy.

Table 7: Observation counts and percentage of train and test data sets.

	Number of Observations	Percentage
Train Set	1598	70.20
Test Set	678	29.79
Total	2276	100.00

Section 3: Models

Utilizing the training data set, three variable selection techniques were used to select predictor variables to generate the best models. The three techniques used were stepwise regression, subsets regression, and manual selection. Models were assessed with the metric Akaike Information Criterion (AIC), which will penalize model complexity with too many predictor variables. Small values of AIC are preferred. The model with the lowest AIC value generated from each variable selection technique was then assessed for model fit. Models were also assessed with the Mean Square Error (MSE). Small values of MSE are preferred.

Section 3.1: Stepwise Variable Selection

A linear regression model was created using all variables from the completed moneyball data set. The `stepAIC` function in R was used to add and remove predictor variables until a model with the lowest AIC was achieved. The following model was generated with the stepwise method:

$$\begin{aligned} E(TargetWins) = 21.742664 - 0.061358(TeamBattingH) + 0.066999(TeamBatting2B) + \\ 0.207489(TeamBattingHR) + 0.183708(TeamBatting3B) + 0.052708(TeamBaserunSB) + \\ 0.023797(TeamBaserunCS) - 0.049908(TeamPitchingHR) - 0.045895(TeamPitchingBB) + \\ 0.024564(TeamPitchingSO) - 0.018593(TeamFieldingE) + 0.089688(TeamBatting1B) + \\ 14.493899(logTeamPitchingH) - 15.137826(logTeamFieldingE) + 0.067146(TeamBattingBB) - \\ 0.041862(TeamBattingSO) - 0.130568(TeamFieldingDP) \end{aligned}$$

Table 8: Stepwise model summary.

	Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
Residuals	-53	-8	0	0	8	61

term	estimate	std.error	statistic	p.value
(Intercept)	21.7426640	24.7706516	0.8777591	0.3801678
TEAM_BATTING_H	-0.0613580	0.0113588	-5.4018180	0.0000001
TEAM_BATTING_2B	0.0669993	0.0133238	5.0285296	0.0000005
TEAM_BATTING_HR	0.2074885	0.0360647	5.7532301	0.0000000
TEAM_BATTING_3B	0.1837085	0.0192481	9.5442650	0.0000000
TEAM_BASERUN_SB	0.0527075	0.0049347	10.6809928	0.0000000
TEAM_BASERUN_CS	0.0237967	0.0102211	2.3281971	0.0199895
TEAM_PITCHING_HR	-0.0499078	0.0277335	-1.7995527	0.0720647
TEAM_PITCHING_BB	-0.0458954	0.0088092	-5.2099145	0.0000002
TEAM_PITCHING_SO	0.0245641	0.0044725	5.4922283	0.0000000
TEAM_FIELDING_E	-0.0185931	0.0059456	-3.1272288	0.0017871
TEAM_BATTING_1B	0.0896877	0.0106544	8.4178717	0.0000000
log_TEAM_PITCHING_H	14.4938988	3.0021323	4.8278681	0.0000015
log_TEAM_FIELDING_E	-15.1378259	2.1003169	-7.2074007	0.0000000
TEAM_BATTING_BB	0.0671464	0.0105088	6.3895287	0.0000000
TEAM_BATTING_SO	-0.0418624	0.0053398	-7.8396200	0.0000000
TEAM_FIELDING_DP	-0.1305683	0.0126637	-10.3104296	0.0000000

Residual Standard Error	Adjusted R-squared	MultipleR-squared	F-Statistic
12.2	0.3964	0.4006	94.37

Term	VIF
TEAM_BATTING_H	34.81
TEAM_BATTING_2B	5.91
TEAM_BATTING_HR	72.42
TEAM_BATTING_3B	4.39
TEAM_BASERUN_SB	3.9
TEAM_BASERUN_CS	4.02
TEAM_PITCHING_HR	43.9
TEAM_PITCHING_BB	15.9
TEAM_PITCHING_SO	17.48
TEAM_FIELDING_E	24.69
TEAM_BATTING_1B	28.66
log_TEAM_PITCHING_H	12.89
log_TEAM_FIELDING_E	25.28
TEAM_BATTING_BB	25.24
TEAM_BATTING_SO	26.35
TEAM_FIELDING_DP	2.11

The sign in front of the coefficient indicates whether it has a positive (+) or negative (-) effect on target wins. Variable effects were as expected, with four exceptions. TEAM_BATTING_H had a negative effect on target wins. This variable is the total number of base hits. As there are variables for single, double, triple and home run hits, the negative sign for TEAM_BATTING_H would correct for the duplicate counting of hits. TEAM_PITCHING_H and TEAM_BASERUN_CS had positive signs indicating a positive impact on wins and TEAM_FIELDING_DP had a negative sign indicating a negative impact on wins - all contrary to the theoretical effect. TEAM_BASERUN_CS and TEAM_FIELDING_DP variables contained missing values that required imputation. TEAM_PITCHING_H contained outliers and underwent truncation for transformation.

Seventeen predictors were included in the model and all variables were significant at the 95th significance level except for TEAM_PITCHING_HR (p-value 0.072) and the intercept (p-value 0.380). The multiple R-squared value is 0.4006. This indicates that 40.06% of the variation in carryover can be explained by the independent variables (predictors) in the regression model. R-squared has a monotonic relationship with model parameters. To account for this relationship, the adjusted R-squared will penalize models that have too many unimportant predictor variables and will allow models of different sizes to be compared. The adjusted R-squared value is 0.3964. The adjusted R-square value will always be lower than the multiple R-squared value.

The residual standard error of 12.24 is the average error in predicting target wins from the predictors in this model. The global F-statistic is 94.37, and tests whether the predictors as a group predict the target wins above chance levels.

The variance inflation factor (VIF) for each predictor variable was calculated. The VIF is a measurement of collinearity between predictors. Small values of VIF are preferred. There were high values of VIF (greater than 3) for all variables except two (TEAM_BASERUN_SB and TEAM_FIELDING_DP).

Section 3.2: Subset Variable Selection

The subset variable selection differs from the stepwise regression, as every possible model is inspected. The `regsubsets` function in R was used to add and remove predictor variables until a model with the highest Adjusted R-squared was achieved. The two best models for each subset are graphed, and a black bar indicates that the variable was used in the model. The resulting Adjusted R-squared is plotted on the y-axis. From this calculation, a regression model was specified based upon the variables indicated in the plot. The following model was generated with the Subset method:

$$E(TargetWins) = -177.9451636 + 0.1136369(TeamBattingHR) + 0.0112150(TeamPitchingBB) - 18.7842817(logTeamFieldingE) - 0.0155041(TeamBattingSO) - 0.1204914(TeamFieldingDP) + 52.8797335(logTeamBatting1B) + 3.6547292(ztransTeamBatting3B) + 5.6118841(ztransTeamBaserunSB)$$

Table 12: Subset model mummary.

	Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
Residuals	-51	-8	0	0	8	73

term	estimate	std.error	statistic	p.value
(Intercept)	-177.9451636	33.2282012	-5.355245	1.00e-07
TEAM_BATTING_HR	0.1136369	0.0093819	12.112372	0.00e+00
TEAM_PITCHING_BB	0.0112150	0.0028631	3.917103	9.34e-05
log_TEAM_FIELDING_E	-18.7842817	1.0291144	-18.252860	0.00e+00
TEAM_BATTING_SO	-0.0155041	0.0026968	-5.749031	0.00e+00
TEAM_FIELDING_DP	-0.1204914	0.0149738	-8.046808	0.00e+00
log_TEAM_BATTING_1B	52.8797335	4.5630742	11.588620	0.00e+00
ztrans_TEAM_BATTING_3B	3.6547292	0.4998901	7.311065	0.00e+00
ztrans_TEAM_BASERUN_SB	5.6118841	0.4903962	11.443571	0.00e+00

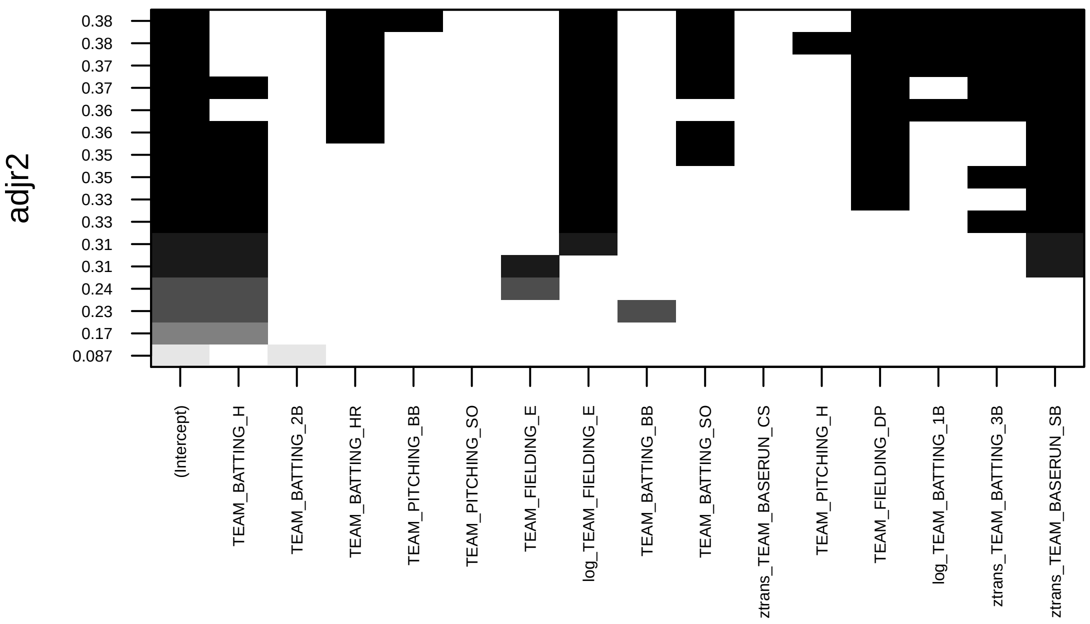


Figure 4: Plot of the best two models for each subset size assessed by Adjusted R-squared.

Residual Standard Error	Adjusted R-squared	Multiple R-squared	F-Statistic
12.6	0.3787	0.3818	122.66

Term	VIF
TEAM_BATTING_HR	3.22
TEAM_PITCHING_BB	1.14
log_TEAM_FIELDING_E	4.03
TEAM_BATTING_SO	4.46
TEAM_FIELDING_DP	1.96
log_TEAM_BATTING_1B	2.6
ztrans_TEAM_BATTING_3B	2.44
ztrans_TEAM_BASERUN_SB	2.44

Variable effects on target wins (positive or negative signs) were as expected, with one exception. TEAM_FIELDING_DP had a negative sign that indicates a negative impact on wins - contrary to the theoretical effect. TEAM_FIELDING_DP variables contained missing values that required imputation.

Nine predictors were included in the model and all variables were significant at the 99.9th significance level. The multiple R-squared value is 0.3818. This indicates that 38.18% of the variation in carryover can be explained by the independent variables (predictors) in the regression model. The adjusted R-squared value is 0.3787.

The residual standard error of 12.57 is the average error in predicting target wins from the predictors in this model. The global F-statistic is 122.7, and tests whether the predictors as a group predict the target wins above chance levels. The VIF values were improved with this model, as all were equal to 4.46 or less.

Section 3.3: Manual Variable Selection

Based upon the performance of the two prior models, a third model was created via manual selection of variables. The variables were selected based upon the performance of the coefficient (p-value) and VIF (collinearity) values.

$$\begin{aligned}E(TargetWins) = -230.4469990 + 0.0200439(TeamBatting2B) + 0.1235413(TeamBattingHR) - \\ 0.0119928(TeamBattingBB) - 0.0122523(TeamBattingSO) + 0.0055580(TeamPitchingH) - \\ 0.1242545(TeamFieldingDP) + 43.7635718(logTeamBatting1B) + \\ 2.7957019(ztransTeamBatting3B) - 11.8228106(ztransTeamFieldingE) + \\ 5.8305525(ztransTeamBaserunSB)\end{aligned}$$

Table 16: Manual selection model summary.

	Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
Residuals	-49	-8	0	0	8	53

term	estimate	std.error	statistic	p.value
(Intercept)	-230.4469990	34.8988043	-6.603292	0.0000000
TEAM_BATTING_2B	0.0200439	0.0085217	2.352095	0.0187896
TEAM_BATTING_HR	0.1235413	0.0099719	12.388902	0.0000000
TEAM_BATTING_BB	0.0119928	0.0036664	3.270993	0.0010947
TEAM_BATTING_SO	-0.0122523	0.0026710	-4.587216	0.0000048
TEAM_PITCHING_H	0.0055580	0.0007933	7.006608	0.0000000
TEAM_FIELDING_DP	-0.1242545	0.0151541	-8.199402	0.0000000
log_TEAM_BATTING_1B	43.7635718	4.9481545	8.844423	0.0000000
ztrans_TEAM_BATTING_3B	2.7957019	0.5117724	5.462784	0.0000001
ztrans_TEAM_FIELDING_E	-11.8228106	0.8213467	-14.394421	0.0000000
ztrans_TEAM_BASERUN_SB	5.8305525	0.4990192	11.684024	0.0000000

Residual Standard Error	Adjusted R-squared	Multiple R-squared	F-Statistic
12.6	0.3802	0.3841	98.96

Term	VIF
TEAM_BATTING_2B	1.56
TEAM_BATTING_HR	3.64
TEAM_BATTING_BB	2.05
TEAM_BATTING_SO	4.39
TEAM_PITCHING_H	3.94
TEAM_FIELDING_DP	2.01
log_TEAM_BATTING_1B	3.07
ztrans_TEAM_BATTING_3B	2.57
ztrans_TEAM_FIELDING_E	6.71
ztrans_TEAM_BASERUN_SB	2.54

Variable effects on target wins (positive or negative signs) were as expected, with two exceptions. TEAM_PITCHING_H had a positive sign indicating a positive impact on wins and TEAM_FIELDING_DP had a negative sign that indicates a negative impact on wins - contrary to the theoretical effect. TEAM_FIELDING_DP variables contained missing values that required imputation. TEAM_PITCHING_H contained outliers and underwent truncation for transformation.

Eleven predictors were included in the model and all variables were significant at the 99.9th significance level except for TEAM_BATTING_2B that was at the 95th significance level. The multiple R-squared value is 0.3841. This indicates that 38.41% of the variation in carryover can be explained by the independent variables (predictors) in the regression model. The adjusted R-squared value is 0.3802.

The residual standard error of 12.56 is the average error in predicting target wins from the predictors in this model. The global F-statistic is 98.96, and tests whether the predictors as a group predict the target wins above chance levels. The VIF values were improved with this model, as all were equal to 4.39 or less, except for ztrans_TEAM_FIELDING_E (6.71).

Section 4: Model Selection

For each of the models, the adjusted R-squared, AIC, Bayesian Information Criterion (BIC), mean standard error (MSE) and mean absolute error (MAE) were calculated. These metrics are used to evaluate the model fit with the training set data. A high value of adjusted R-squared and a small value of AIC, BIC, MSE and MAE is desired.

Section 4.1: Model Comparison

The Stepwise model had the highest Adjusted R-squared value, followed by the Manual Selection and then the Subset model. The AIC and BIC of the Subset and Manual models were nearly identical, and they were lower than the Stepwise model. The Stepwise model had a lower MSE. All models had the same MAE value of 10.

The model metrics were compared against each other and ranked. The Stepwise model ranked first in three of the metrics, while the Subset and Manual models both ranked first in two of the metrics. The second ranked model by metrics was the Manual model followed by the Subset.

The Stepwise model had the worst ranking for AIC and BIC. Both of these metrics will penalize a model with more predictors. The Stepwise model had the most predictors (17) compared to the Subset (9) and Manual (11) models.

Table 20: Train Model Performance.

	Adjusted R-squared	AIC	BIC	MSE	MAE
Stepwise	0.3964	17879	17982	149	10
Subset	0.3787	12637	12690	157	10
Manual Selection	0.3802	12635	12699	157	10

Table 21: Rank of model in-sample fit.

	Adjusted R-squared	AIC	BIC	MSE	MAE
First	Stepwise	Manual	Subset	Stepwise	All 3 Tied
Second	Manual	Subset	Manual	Subset & Manual	
Third	Subset	Stepwise	Stepwise		

The predictive accuracy was assessed with the out-of-sample test data set by calculating the MSE and MAE for each model. Lower values of MSE and MAE are desired, and the model criterions were compared against each other. The Stepwise model had the lowest MSE and MAE values of all three models. The MSE and MAE values of the test data set with the Stepwise model were lower than those calculated from the train data set. This indicates that the model was not overfit to the train data set. The Subset model had the next lowest MSE value (154) followed by the Manual model (158). Both the Subset and Manual models had the same MAE value (10).

Table 22: Predictive Accuracy of Each Model with the Test Data Set.

	MSE	MAE
Stepwise	143	9
Subset	154	10
Manual	158	10

Section 4.2: Model Validation

Prediction grades for the in-sample (train) data set and out-of-sample (test) data set were calculated. Grades were determined by the predictive accuracy of each model. A grade of 1 is ideal, followed by grade 2, grade 3 and grade 4.

The Stepwise, Subset and Manual selection models generated nearly the same predictive accuracy with the in-sample (train) data set. All three models were accurate to within 10% of target wins 50% of the time, with a maximum difference between model accuracy of only 0.44%. The three models were accurate within 10-15% of target wins between 17.79% to 19.40% of the time, and within 15-25% of target wins between 17.96% to 21.27% of the time.

Table 23: Prediction grades of in-sample data set by model.

	Grade 1: [0.0.10]	Grade 2: (0.10,0.15]	Grade 3: (0.15,0.25]	Grade 4: (0.25+]
Stepwise Train	0.5075	0.1779	0.2127	0.1019
Subset Train	0.5050	0.1921	0.1796	0.1233
Manual Train	0.5031	0.1940	0.1802	0.1227

The Stepwise selection model had the best predictive accuracy with the out-of-sample (train) data set. The Stepwise model was accurate to within 10% of target wins 49.71% of the time, followed by the Subset model with 48.67% and the Manual model with 46.90%. The three models were accurate within 10-15% of target wins between 19.47% to 20.94% of the time, and within 15-25% of target wins between 20.06% to 21.24% of the time.

Table 24: Prediction grades of out-of-sample data set by model.

	Grade 1: [0.0.10]	Grade 2: (0.10,0.15]	Grade 3: (0.15,0.25]	Grade 4: (0.25+]
Stepwise Test	0.4971	0.1947	0.2124	0.09587
Subset Test	0.4867	0.2021	0.2006	0.11060
Manual Test	0.4690	0.2094	0.2094	0.11210

Conclusion

The Stepwise model was selected as the best model. The Stepwise model performance on the train data set was similiar to the other two models, but performance was slightly improved with the test data set. It was a more complicated model with the most variables, and had the highest Adjusted R-squared, AIC and BIC values. However, the MSE was the lowest with the Stepwise model. The Stepwise model would be used with a new data set to predict the number of target wins.

References

Fox, J. and Weisberg, S. (2011). An R Companion to Applied Regression, Second Edition. Thousand Oaks, CA: Sage Publications, Inc.

Kabacoff, R. (2015). R in Action, Second Edition. Shelter Island, NY: Manning Publications Co.

Lander, J. (2014). R for Everyone. Upper Saddle River, NJ: Addison-Wesley.

Pardoe, I. (2012). Applied Regression Modeling, Second Edition. Hoboken, NJ: John Wiley & Sons, Inc.

Stowell, S. (2014). Using R for Statistics. New York, NY: Apress.

##Code

```
knitr::opts_chunk$set(echo = FALSE)
library(knitr)

######## Download appropriate packages and install
#install.packages("rJava")
#install.packages("readr")
#install.packages("pbkrtest")
#install.packages("car")
#install.packages("leaps")
#install.packages("MASS")
#install.packages("xlsxjars")
#install.packages("xlsx")
#install.packages("psych")
#install.packages("gridExtra")
#install.packages('corrplot', dependencies=TRUE)
#install.packages('mice')
install.packages('kableExtra')
install.packages('knitr')
install.packages('tinytex')

library(rJava)
library(readr)
library(pbkrtest)
library(car)
library(leaps)
library(MASS)
library(xlsxjars)
library(xlsx)
library(psych)
library(gridExtra)
library(corrplot)
library(mice)
library(VIM)
library(kableExtra)
library(RColorBrewer)
library(tinytex)
library(broom)

#Designated proper working environment
#Load moneyball file in R
setwd("~/Desktop/R/")
moneyball=read.csv("moneyball.csv",header=T)

#str(moneyball)


#table of n, mean, sd, median, trimmed, mad, min, max, range, skew, kurtosis, se
#from psych package
kable(round(describe(moneyball[2:17]), 1), caption = "Statistical Summary of Variables.",
      format = "latex", booktabs = T ) %>%
    kable_styling(latex_options = c("striped", "scale_down"))
```

```r
#function to describe variables with missing data
myNA.variable.summary <- function(variable){
  NAsum.summary <- sum(is.na(variable))
  NAmean.summary <- (round(mean(is.na(variable)), 2)*100)
  return(c(NAsum.summary, NAmean.summary))
}

#table of variables with missing data
NA.BSO <- myNA.variable.summary(moneyball$TEAM_BATTING_SO)
NA.SB <- myNA.variable.summary(moneyball$TEAM_BASERUN_SB)
NA.CS <- myNA.variable.summary(moneyball$TEAM_BASERUN_CS)
NA.HBP <- myNA.variable.summary(moneyball$TEAM_BATTING_HBP)
NA.PSO <- myNA.variable.summary(moneyball$TEAM_PITCHING_SO)
NA.DP <- myNA.variable.summary(moneyball$TEAM_FIELDING_DP)
overview.NA <- rbind(NA.BSO, NA.SB, NA.CS, NA.HBP, NA.PSO, NA.DP)
colnames(overview.NA) <- c("Number of missing values", "Percent of missing values")
rownames(overview.NA) <- c("TEAM_BATTING_SO", "TEAM_BASERUN_SB", "TEAM_BASERUN_CS",
                             "TEAM_BATTING_HBP", "TEAM_PITCHING_SO", "TEAM_FIELDING_DP")
kable(overview.NA, caption = 'Percentage of Missing Values for Effected Variables.')


# Pitching hits
par(mfrow=c(1,2))
hist(moneyball$TEAM_PITCHING_H, col = "#A71930", xlab = " ",
     main = "Histogram of Hits Against", cex.main = 0.8, cex.axis = 0.8)
boxplot(moneyball$TEAM_PITCHING_H, col = "#A71930",
        main = "Boxplot of Hits Against", cex.main = 0.8, cex.axis = 0.8)
par(mfrow=c(1,1))


#Pitching strikeouts
par(mfrow=c(1,2))
hist(moneyball$TEAM_PITCHING_SO, col = "#DBCEAC", xlab = " ",
     main = "Histograms of Strikeouts", cex.main = 0.8, cex.axis = 0.8)
boxplot(moneyball$TEAM_PITCHING_SO, col = "#DBCEAC",
        main = "Boxplot of Strikeouts", cex.main = 0.8, cex.axis = 0.8)
par(mfrow=c(1,1))


#Kabacoff page 422, Using correlations to exlore missing values
mb <- as.data.frame(abs(is.na(moneyball)))

#viewing the first 5 rows of each data frame
#head(moneyball, n=5)
#head(mb, n=5)

#extract the variables that have some (but not all) missing values
mb.y <- mb[which(apply(mb,2,sum) >0)]


mb.cor <- cor(mb.y[-c(7,8)])

corrplot(mb.cor, type="upper", order="hclust",
         addCoef.col = "black",
         col=brewer.pal(n=8, name="RdBu"),
         diag = FALSE, tl.cex = 0.7, number.cex = 0.8)

#mode of variables with NA values
mode.BSO <- names(table(moneyball$TEAM_BATTING_SO))[table(moneyball$TEAM_BATTING_SO)
                                                    ==max(table(moneyball$TEAM_BATTING_SO))]
mode.SB <- names(table(moneyball$TEAM_BASERUN_SB))[table(moneyball$TEAM_BASERUN_SB)
                                                   ==max(table(moneyball$TEAM_BASERUN_SB))]
mode.CS <- names(table(moneyball$TEAM_BASERUN_CS))[table(moneyball$TEAM_BASERUN_CS)
                                                   ==max(table(moneyball$TEAM_BASERUN_CS))]
mode.HBP <- names(table(moneyball$TEAM_BATTING_HBP))[table(moneyball$TEAM_BATTING_HBP)
                                                     ==max(table(moneyball$TEAM_BATTING_HBP))]
mode.PSO <- names(table(moneyball$TEAM_PITCHING_SO))[table(moneyball$TEAM_PITCHING_SO)
                                                     ==max(table(moneyball$TEAM_PITCHING_SO))]
mode.DP <- names(table(moneyball$TEAM_FIELDING_DP))[table(moneyball$TEAM_FIELDING_DP)
                                                    ==max(table(moneyball$TEAM_FIELDING_DP))]

#This makes each vector the same length, so that values will not be repeated
n <- max(length(mode.BSO), length(mode.SB), length(mode.CS),
         length(mode.HBP), length(mode.PSO), length(mode.DP))
length(mode.BSO) <- n
length(mode.SB) <- n
length(mode.CS) <- n
length(mode.HBP) <- n
length(mode.PSO) <- n
length(mode.DP) <- n

overview.mode <- rbind(mode.BSO, mode.SB, mode.CS, mode.HBP, mode.PSO, mode.DP)

#colnames(overview.mode) <- c("", "Mode", "", "")
rownames(overview.mode) <- c("TEAM_BATTING_SO", "TEAM_BASERUN_SB", "TEAM_BASERUN_CS",
                             "TEAM_BATTING_HBP", "TEAM_PITCHING_SO", "TEAM_FIELDING_DP")

options(knitr.kable.NA = '')
kable(overview.mode, caption = 'Mode for Varibles with Missing Values.')


#quantile calculations
#quantile(x,  probs = c(0.1, 0.5, 1, 2, 5, 10, 50, NA)/100)
quantile.BSO <- quantile(moneyball$TEAM_BATTING_SO,
                         probs = c(0,1,5,10,25,50,75,90,95,99,100)/100, na.rm=TRUE)
quantile.SB <- quantile(moneyball$TEAM_BASERUN_SB,
                        probs = c(0,1,5,10,25,50,75,90,95,99,100)/100, na.rm=TRUE)
quantile.CS <- quantile(moneyball$TEAM_BASERUN_CS,
                        probs = c(0,1,5,10,25,50,75,90,95,99,100)/100, na.rm=TRUE)
quantile.HBP <- quantile(moneyball$TEAM_BATTING_HBP,
                         probs = c(0,1,5,10,25,50,75,90,95,99,100)/100, na.rm=TRUE)
quantile.PSO <- quantile(moneyball$TEAM_PITCHING_SO,
                         probs = c(0,1,5,10,25,50,75,90,95,99,100)/100, na.rm=TRUE)
quantile.DP <- quantile(moneyball$TEAM_FIELDING_DP,
                        probs = c(0,1,5,10,25,50,75,90,95,99,100)/100, na.rm=TRUE)
quantile.H <- quantile(moneyball$TEAM_PITCHING_H,
```

```r
                            probs = c(0,1,5,10,25,50,75,90,95,99,100)/100, na.rm=TRUE)
quantile.PBB <- quantile(moneyball$TEAM_PITCHING_BB,
                              probs = c(0,1,5,10,25,50,75,90,95,99,100)/100, na.rm=TRUE)
quantile.TW <- quantile(moneyball$TARGET_WINS,
                             probs = c(0,1,5,10,25,50,75,90,95,99,100)/100, na.rm=TRUE)
quantile.FE <- quantile(moneyball$TEAM_FIELDING_E,
                             probs = c(0,1,5,10,25,50,75,90,95,99,100)/100, na.rm=TRUE)
quantile.BH <- quantile(moneyball$TEAM_BATTING_H,
                             probs = c(0,1,5,10,25,50,75,90,95,99,100)/100, na.rm=TRUE)

overview.quantile <- rbind(quantile.BSO,quantile.SB,quantile.CS,quantile.HBP,
                           quantile.PSO,quantile.DP,quantile.H, quantile.PBB,
                           quantile.TW, quantile.FE, quantile.BH)
colnames(overview.quantile) <- c("0%","1%","5%","10%","25%","50%",
                             "75%", "90%", "95%", "99%", "100%")
rownames(overview.quantile) <- c("TEAM_BATTING_SO", "TEAM_BASERUN_SB", "TEAM_BASERUN_CS",
                             "TEAM_BATTING_HBP", "TEAM_PITCHING_SO", "TEAM_FIELDING_DP",
                             "TEAM_PITCHING_H", "TEAM_PITCHING_BB", "TARGET_WINS",
                             "TEAM_FIELDING_E", "TEAM_BATTING_H")

kable(overview.quantile, caption = 'Quantiles of Variables of Interest.',
      format = "latex", booktabs = T) %>%
    kable_styling(latex_options = c("striped", "scale_down")) %>%
    add_footnote(c("0% = minimum, 25% = Q1, 50% = median, 75% = Q3, 100% = maximum"))


#list 10 maximum values of variables
max.BSO <- head(sort(moneyball$TEAM_BATTING_SO, decreasing = TRUE), 10)
max.SB <- head(sort(moneyball$TEAM_BASERUN_SB, decreasing = TRUE), 10)
max.CS <- head(sort(moneyball$TEAM_BASERUN_CS, decreasing = TRUE), 10)
max.HBP <- head(sort(moneyball$TEAM_BATTING_HBP, decreasing = TRUE), 10)
max.SO <- head(sort(moneyball$TEAM_PITCHING_SO, decreasing = TRUE), 10)
max.DP <- head(sort(moneyball$TEAM_FIELDING_DP, decreasing = TRUE), 10)
max.H <- head(sort(moneyball$TEAM_PITCHING_H, decreasing = TRUE), 10)
overview.max <- rbind(max.BSO, max.SB, max.CS, max.HBP, max.SO, max.DP, max.H)
rownames(overview.max) <- c("TEAM_BATTING_SO", "TEAM_BASERUN_SB", "TEAM_BASERUN_CS",
                            "TEAM_BATTING_HBP", "TEAM_PITCHING_SO", "TEAM_FIELDING_DP",
                            "TEAM_PITCHING_H")
kable(overview.max, caption = 'List of the Ten Highest Maximum Values.')


#list 10 minimum values of variables
min.BSO <- tail(sort(moneyball$TEAM_BATTING_SO, decreasing = TRUE), 10)
min.SB <- tail(sort(moneyball$TEAM_BASERUN_SB, decreasing = TRUE), 10)
min.CS <- tail(sort(moneyball$TEAM_BASERUN_CS, decreasing = TRUE), 10)
min.HBP <- tail(sort(moneyball$TEAM_BATTING_HBP, decreasing = TRUE), 10)
min.SO <- tail(sort(moneyball$TEAM_PITCHING_SO, decreasing = TRUE), 10)
min.DP <- tail(sort(moneyball$TEAM_FIELDING_DP, decreasing = TRUE), 10)
overview.min <- rbind(min.BSO, min.SB, min.CS, min.HBP, min.SO, min.DP)
rownames(overview.min) <- c("TEAM_BATTING_SO", "TEAM_BASERUN_SB", "TEAM_BASERUN_CS",
                            "TEAM_BATTING_HBP", "TEAM_PITCHING_SO", "TEAM_FIELDING_DP")
kable(overview.min, caption = 'List of the Ten Lowest Minimum Values.')
```

```r
#Boxplot of data set variables
boxplot(moneyball[, c(2:11,13,14,16, 17)],
        las=2, par(mar = c(12, 5, 4, 2)+ 0.1), col="lightgrey", cex.axis=0.7)


#correlation plot of variables that do not contain NA values
moneyball.cor2 <- cor(moneyball[,c(2:7,12:14, 16)])
corrplot(moneyball.cor2, addCoef.col = "black", col=brewer.pal(n=8, name="RdBu"),
         tl.cex = 0.7, number.cex = 0.8)
#corrplot(moneyball.cor2, method = "number", tl.cex = 0.7, number.cex = 0.6)


###################### Section 2 #############################################
###################### Data Preparation ######################################
#exclude BATTING_HBP from data set before imputation due to high number of missing values
exclude <- c('TEAM_BATTING_HBP')
include <- setdiff(names(moneyball), exclude)

#moneyball data set without BATTING_HBP variable now
moneyball <- moneyball[include]

#Impute the missing data with mice package
#Do not include INDEX during calculations
imp.moneyball <- mice(moneyball[, !names(moneyball) %in% "INDEX"], m=5,maxit=50,meth='pmm',seed=500)
summary(imp.moneyball)
completed.moneyball <- complete(imp.moneyball)
anyNA(completed.moneyball)


#################### Data Transformations #######################################

#Straighten Relationships for completed.moneyball - create single hits variable
completed.moneyball$TEAM_BATTING_1B <- completed.moneyball$TEAM_BATTING_H -
  completed.moneyball$TEAM_BATTING_HR - completed.moneyball$TEAM_BATTING_3B -
  completed.moneyball$TEAM_BATTING_2B


par(mfrow=c(1,2))
hist(completed.moneyball$TEAM_BATTING_1B, col = "#A71930", xlab = " ",
     main = "Histogram of Single Hits", cex.main = 0.8, cex.axis = 0.8)
boxplot(completed.moneyball$TEAM_BATTING_1B, col = "#A71930",
        main = "Boxplot of Single Hits", cex.main = 0.8, cex.axis = 0.8)
par(mfrow=c(1,1))


############# Examination of TEAM_BATTING_1B variable ##########################
#table of n, mean, sd, median, trimmed, mad, min, max, range, skew, kurtosis, se
#from psych package
kable(round(describe(completed.moneyball$TEAM_BATTING_1B), 1),
      caption = "Statistical Summary of Single Hits",
      format = "latex", booktabs = T ) %>%
    kable_styling(latex_options = c("striped", "scale_down"))
```

```r
#summary(completed.moneyball$TEAM_BATTING_1B)

#Quantile Analysis
quantile.1B <- quantile(completed.moneyball$TEAM_BATTING_1B,
                        probs = c(0,1,5,10,25,50,75,90,95,99,100)/100, na.rm=TRUE)
overview2.quantile <- rbind(quantile.1B)
colnames(overview2.quantile) <- c("0%","1%","5%","10%","25%","50%",
                             "75%", "90%", "95%", "99%", "100%")
rownames(overview2.quantile) <- c("TEAM_BATTING_1B")

kable(overview2.quantile, caption = 'Quantiles of Single Hits.',
      format = "latex", booktabs = T) %>%
    kable_styling(latex_options = c("striped", "scale_down"))


#Straighten Relationships for completed.moneyball - log transformation
#log transformation cannot be done on variables with values of zero
completed.moneyball$log_TEAM_BATTING_1B <- log(completed.moneyball$TEAM_BATTING_1B)
completed.moneyball$log_TEAM_PITCHING_H <- log(completed.moneyball$TEAM_PITCHING_H)
completed.moneyball$log_TEAM_FIELDING_E <- log(completed.moneyball$TEAM_FIELDING_E)

#Set to 1% truncation for lower limit and 99% truncation for upper limit - trimming data
completed.moneyball$TEAM_PITCHING_SO[(completed.moneyball$TEAM_PITCHING_SO > 1467)] = 1467
completed.moneyball$TEAM_PITCHING_SO[(completed.moneyball$TEAM_PITCHING_SO < 207)] = 207
completed.moneyball$TEAM_PITCHING_H[(completed.moneyball$TEAM_PITCHING_H > 7054)] = 7054
completed.moneyball$TEAM_PITCHING_BB[(completed.moneyball$TEAM_PITCHING_BB > 921)] = 921
completed.moneyball$TEAM_PITCHING_BB[(completed.moneyball$TEAM_PITCHING_BB < 240)] = 240
completed.moneyball$TEAM_FIELDING_E[(completed.moneyball$TEAM_FIELDING_E > 1228)] = 1228
completed.moneyball$TEAM_BATTING_H[(completed.moneyball$TEAM_BATTING_H > 1945)] = 1945

#Straighten Relationships for completed.moneyball - z tranformation / standardizing
completed.moneyball$ztrans_TEAM_BATTING_3B <-
(completed.moneyball$TEAM_BATTING_3B - mean(completed.moneyball$TEAM_BATTING_3B)) /
sd(completed.moneyball$TEAM_BATTING_3B)
completed.moneyball$ztrans_TEAM_BASERUN_SB <-
(completed.moneyball$TEAM_BASERUN_SB - mean(completed.moneyball$TEAM_BASERUN_SB)) /
sd(completed.moneyball$TEAM_BASERUN_SB)
completed.moneyball$ztrans_TEAM_BASERUN_CS <-
(completed.moneyball$TEAM_BASERUN_CS - mean(completed.moneyball$TEAM_BASERUN_CS)) /
sd(completed.moneyball$TEAM_BASERUN_CS)
completed.moneyball$ztrans_TEAM_PITCHING_BB <-
(completed.moneyball$TEAM_PITCHING_BB - mean(completed.moneyball$TEAM_PITCHING_BB)) /
sd(completed.moneyball$TEAM_PITCHING_BB)
completed.moneyball$ztrans_TEAM_FIELDING_E <-
(completed.moneyball$TEAM_FIELDING_E - mean(completed.moneyball$TEAM_FIELDING_E)) /
sd(completed.moneyball$TEAM_FIELDING_E)
completed.moneyball$ztrans_TEAM_BATTING_SO <-
(completed.moneyball$TEAM_BATTING_SO - mean(completed.moneyball$TEAM_BATTING_SO)) /
sd(completed.moneyball$TEAM_BATTING_SO)


####################Random Number seed generator and Train / Test split########
# Set the seed on the random number generator so you get the same split every time that
# you run the code.
set.seed(123)
completed.moneyball$u <- runif(n=dim(completed.moneyball)[1],min=0,max=1);

# Create train/test split;
train.df <- subset(completed.moneyball, u<0.70);
test.df <- subset(completed.moneyball, u>=0.70);

# Check your data split. The sum of the parts should equal the whole.
# Do your totals add up?
dim(completed.moneyball)[1]
dim(train.df)[1]
dim(test.df)[1]
dim(train.df)[1]+dim(test.df)[1]

#Create table of observations in each data frame and calculate percentage split
total <- dim(completed.moneyball)[1]
total.train <- dim(train.df)[1]
total.test <- dim(test.df)[1]

percent.total <- round((total/total)*100, 1)
percent.train <- round((total.train/total)*100, 1)
percent.test <- signif((total.test/total)*100, 4)

total.all <- c(total.train, total.test, total)
percent.all <- c(percent.train, percent.test, percent.total)


#Create table of Train and Test data sets - observation counts and percentages
overview.split <- cbind(total.all, percent.all)
colnames(overview.split) <- c("Number of Observations", "Percentage")
rownames(overview.split) <- c("Train Set", "Test Set", "Total")
kable(overview.split, caption = "Observation counts and percentage of train and test data sets.")

#################### Part 3: Model Creation ############################################
#Function for Mean Square Error Calculation
mse <- function(sm)
  mean(sm$residuals^2)

########################### Model 1: Stepwise #################################################

stepwisemodel3c <- lm(formula = TARGET_WINS ~ TEAM_BATTING_H + TEAM_BATTING_2B + TEAM_BATTING_HR +
                        TEAM_BATTING_3B + TEAM_BASERUN_SB + TEAM_BASERUN_CS + TEAM_PITCHING_HR +
                        TEAM_PITCHING_BB + TEAM_PITCHING_SO + TEAM_FIELDING_E + TEAM_BATTING_1B +
                        log_TEAM_PITCHING_H + log_TEAM_FIELDING_E + ztrans_TEAM_PITCHING_BB +
                        TEAM_BATTING_BB + TEAM_BATTING_SO + ztrans_TEAM_BASERUN_CS + TEAM_PITCHING_H +
                        TEAM_FIELDING_DP + log_TEAM_BATTING_1B + ztrans_TEAM_BATTING_3B +
                        ztrans_TEAM_BATTING_SO + ztrans_TEAM_FIELDING_E +
                        ztrans_TEAM_BASERUN_SB, data = completed.moneyball)
stepwise3c <- stepAIC(stepwisemodel3c, direction = "both")
summary(stepwise3c)
vif(stepwise3c)
sqrt(vif(stepwise3c)) > 2
```

```
mse(stepwise3c)



############################ Model 1: Stepwise Statistics ########################################

#Function for model summary statistics
model.summary <- function(model){
  residualse.summary <- round(summary(lm(model))$sigma,1)
  adjrs.summary <- round(summary(lm(model))$adj.r.squared,4)
  multrs.summary <- round(summary(lm(model))$r.squared,4)
  fstat.summary <- round(unname(summary(lm(model))$fstatistic)[1], 2)
  return(c(residualse.summary, adjrs.summary, multrs.summary, fstat.summary))
}

#Model residual summary table
stepwise3c.residuals <- t(round(summary(stepwise3c$residuals), 0))
rownames(stepwise3c.residuals) <- c("Residuals")
kable(stepwise3c.residuals, caption = 'Stepwise model summary.')

#Model Summary table of coefficients
stepwise3c.alternate <- tidy(stepwise3c)
kable(stepwise3c.alternate)

#Table of model ANOVA
stepwise3c.summary <- t(model.summary(stepwise3c))
colnames(stepwise3c.summary) <- c('Residual Standard Error','Adjusted R-squared','MultipleR-squared','F
kable(stepwise3c.summary)


#Table of model VIF
stepwise3c.vif <- round(vif(stepwise3c), 2)
stepwise3c.vif.term <- names(stepwise3c.vif)
stepwise3c.vif.numbers <- unname(stepwise3c.vif)
stepwise3c.vif.final <- cbind(stepwise3c.vif.term, stepwise3c.vif.numbers)
colnames(stepwise3c.vif.final) <- c("Term", "VIF")
kable(stepwise3c.vif.final)


############################ Model 2: Subset ##################################################

subsets <- regsubsets(TARGET_WINS ~ TEAM_BATTING_H + TEAM_BATTING_2B + TEAM_BATTING_HR +
                        TEAM_PITCHING_BB + TEAM_PITCHING_SO + TEAM_FIELDING_E +
                        log_TEAM_FIELDING_E + TEAM_BATTING_BB + TEAM_BATTING_SO +
                        ztrans_TEAM_BASERUN_CS + TEAM_PITCHING_H +TEAM_FIELDING_DP +
                        log_TEAM_BATTING_1B + ztrans_TEAM_BATTING_3B +
                        ztrans_TEAM_BASERUN_SB,  data = train.df, nbest = 2)
#par(mar=c(10, 4.1, 2.1, 2.1))
#plot(subsets, scale="adjr2")
summary(subsets)

subset <- lm(TARGET_WINS ~ TEAM_BATTING_HR + TEAM_PITCHING_BB + log_TEAM_FIELDING_E +
               TEAM_BATTING_SO + TEAM_FIELDING_DP + log_TEAM_BATTING_1B +
```

```r
               ztrans_TEAM_BATTING_3B + ztrans_TEAM_BASERUN_SB, data = train.df)
summary(subset)
mse(subset)
vif(subset)

#Plot of regsubsets results
par(cex.axis=0.5)
plot(subsets, scale="adjr2",  cex.subsets=0.5)


########################### Model 2: Subset Statistics #########################################

#Model residual summary table
subset.residuals <- t(round(summary(subset$residuals), 0))
rownames(subset.residuals) <- c("Residuals")
kable(subset.residuals, caption = 'Subset model mummary.')

#Model summary table of coefficient
subset.alternate <- tidy(subset)
kable(subset.alternate)

#Table of model ANOVA
subset.summary <- t(model.summary(subset))
colnames(subset.summary) <- c('Residual Standard Error',
                                  'Adjusted R-squared','Multiple R-squared','F-Statistic')
kable(subset.summary)

#Table of model VIF
subset.vif <- round(vif(subset), 2)
subset.vif.term <- names(subset.vif)
subset.vif.numbers <- unname(subset.vif)
subset.vif.final <- cbind(subset.vif.term, subset.vif.numbers)
colnames(subset.vif.final) <- c("Term", "VIF")
kable(subset.vif.final)



########################### Model 3: Manual ###################################################

model1 <- lm(formula = TARGET_WINS ~ TEAM_BATTING_2B + TEAM_BATTING_HR +
                       TEAM_BATTING_BB + TEAM_BATTING_SO + TEAM_PITCHING_H +
                       TEAM_FIELDING_DP + log_TEAM_BATTING_1B + ztrans_TEAM_BATTING_3B +
                       ztrans_TEAM_FIELDING_E + ztrans_TEAM_BASERUN_SB, data = train.df)


summary(model1)
mse(model1)
vif(model1)


########################### Model 3: Manual Statistics #########################################

#Model residual summary table
```

```r
model1.residuals <- t(round(summary(model1$residuals), 0))
rownames(model1.residuals) <- c("Residuals")
kable(model1.residuals, caption = 'Manual selection model summary.')

#Model summary table of coefficients
model1.alternate <- tidy(model1)
kable(model1.alternate)

#Table of model ANOVA
model1.summary <- t(model.summary(model1))
colnames(model1.summary) <- c('Residual Standard Error',
                                  'Adjusted R-squared','Multiple R-squared','F-Statistic')
kable(model1.summary)

#Table of model VIF
model1.vif <- round(vif(model1), 2)
model1.vif.term <- names(model1.vif)
model1.vif.numbers <- unname(model1.vif)
model1.vif.final <- cbind(model1.vif.term, model1.vif.numbers)
colnames(model1.vif.final) <- c("Term", "VIF")
kable(model1.vif.final)


######################## Section 4: Model Selection ##################################
#Model Performance function
Model.fit <- function(model){
  adj.rsquare <- round(summary(lm(model))$adj.r.squared, 4)
  AIC.fit <- round(AIC(model), 0)
  BIC.fit <- round(BIC(model), 0)
  MSE.fit <- round(mean(model$residuals^2), 0)
  MAE.fit <- round(mean(abs(model$residuals)), 0)
  return(c(adj.rsquare, AIC.fit, BIC.fit, MSE.fit, MAE.fit))
}

#Model Comparison
#Table of R-squared, AIC, BIC, MSE, and MAE for each Model
Stepwise.fit <- Model.fit(stepwise3c)
Subset.fit <- Model.fit(subset)
Model1.fit <- Model.fit(model1)

fit.compare <- rbind(Stepwise.fit, Subset.fit, Model1.fit)
colnames(fit.compare) <- c('Adjusted R-squared', 'AIC', 'BIC', 'MSE', 'MAE')
rownames(fit.compare) <- c('Stepwise', 'Subset', 'Manual Selection')
kable(fit.compare, caption = 'Train Model Performance.')


#Model Rank
#Table of Model Rank for each criterion above (R-squared, AIC, BIC, MSE, MAE)
First <- c('Stepwise', 'Manual', 'Subset', 'Stepwise', 'All 3 Tied')
Second <- c('Manual', 'Subset', 'Manual', 'Subset & Manual', ' ')
Third <- c('Subset', 'Stepwise', 'Stepwise', ' ', ' ')
model.rank <- rbind(First, Second, Third)
colnames(model.rank) <- c('Adjusted R-squared', 'AIC', 'BIC', 'MSE', 'MAE')
```

```r
kable(model.rank, caption = 'Rank of model in-sample fit.')


##############################Predictive Accuracy####################
stepwise.test <- predict(stepwise3c,newdata=test.df);
subset.test <- predict(subset, newdata=test.df)
manual.test <- predict(model1, newdata=test.df)

st.residuals <- (test.df$TARGET_WINS - stepwise.test)
sb.residuals <- (test.df$TARGET_WINS - subset.test)
ma.residuals <- (test.df$TARGET_WINS - manual.test)


#Calculate the MSE and MAE of each model
st.mse <- round(mean(st.residuals^2), 0)
st.mae <- round(mean(abs(st.residuals)), 0)

sb.mse <- round(mean(sb.residuals^2), 0)
sb.mae <- round(mean(abs(sb.residuals)), 0)

ma.mse <- round(mean(ma.residuals^2), 0)
ma.mae <- round(mean(abs(ma.residuals)), 0)

#Create table of predictive accuracy with model MSE and MAE
stepwise.predict.fit <- cbind(st.mse, st.mae)
subset.predict.fit <- cbind(sb.mse, sb.mae)
manual.predict.fit <- cbind(ma.mse, ma.mae)

Predict.fit.table <- rbind(stepwise.predict.fit, subset.predict.fit, manual.predict.fit)
colnames(Predict.fit.table) <- c('MSE', 'MAE')
rownames(Predict.fit.table) <- c('Stepwise', 'Subset', 'Manual')
kable(Predict.fit.table, caption = 'Predictive Accuracy of Each Model with the Test Data Set.')


#################################Operational Validation###########################

# Training Data
# Abs Pct Error
########Stepwise Train Test###################################
stepwise.pct <- abs(stepwise3c$residuals)/train.df$TARGET_WINS;
# Assign Prediction Grades;
stepwise.trainPredictionGrade <- ifelse(stepwise.pct<=0.10,'Grade 1: [0.0.10]',
                                  ifelse(stepwise.pct<=0.15,'Grade 2: (0.10,0.15]',
                                         ifelse(stepwise.pct<=0.25,'Grade 3: (0.15,0.25]',
                                                'Grade 4: (0.25+]')
                                  )
)
stepwise.trainTable <- table(stepwise.trainPredictionGrade)
stepwise.trainTable/sum(stepwise.trainTable)
#Create table for paper
stepwise.traingrade <- cbind(signif((stepwise.trainTable[1]/sum(stepwise.trainTable)),4),
                       signif((stepwise.trainTable[2]/sum(stepwise.trainTable)),4),
                       signif((stepwise.trainTable[3]/sum(stepwise.trainTable)),4),
                       signif((stepwise.trainTable[4]/sum(stepwise.trainTable)),4))
colnames(stepwise.traingrade) <- c('Grade 1: [0.0.10]', 'Grade 2: (0.10,0.15]',
                                   'Grade 3: (0.15,0.25]', 'Grade 4: (0.25+]')
rownames(stepwise.traingrade) <- c('Stepwise Train')
#kable(stepwise.traingrade)

########Subset Train Test####################################
subset.pct <- abs(subset$residuals)/train.df$TARGET_WINS;
# Assign Prediction Grades;
subset.trainPredictionGrade <- ifelse(subset.pct<=0.10,'Grade 1: [0.0.10]',
                                  ifelse(subset.pct<=0.15,'Grade 2: (0.10,0.15]',
                                         ifelse(subset.pct<=0.25,'Grade 3: (0.15,0.25]',
                                                'Grade 4: (0.25+]')
                                  )
)
subset.trainTable <- table(subset.trainPredictionGrade)
subset.trainTable/sum(subset.trainTable)
#Create table for paper
subset.traingrade <- cbind(signif((subset.trainTable[1]/sum(subset.trainTable)),4),
                        signif((subset.trainTable[2]/sum(subset.trainTable)),4),
                        signif((subset.trainTable[3]/sum(subset.trainTable)),4),
                        signif((subset.trainTable[4]/sum(subset.trainTable)),4))
colnames(subset.traingrade) <- c('Grade 1: [0.0.10]', 'Grade 2: (0.10,0.15]',
                                 'Grade 3: (0.15,0.25]', 'Grade 4: (0.25+]')
rownames(subset.traingrade) <- c('Subset Train')
#kable(subset.traingrade)

########Manual Train Test####################################
manual.pct <- abs(model1$residuals)/train.df$TARGET_WINS;
# Assign Prediction Grades;
manual.trainPredictionGrade <- ifelse(manual.pct<=0.10,'Grade 1: [0.0.10]',
                                        ifelse(manual.pct<=0.15,'Grade 2: (0.10,0.15]',
                                               ifelse(manual.pct<=0.25,'Grade 3: (0.15,0.25]',
                                                      'Grade 4: (0.25+]')
                                        )
)
manual.trainTable <- table(manual.trainPredictionGrade)
manual.trainTable/sum(manual.trainTable)
#Create table for paper
manual.traingrade <- cbind(signif((manual.trainTable[1]/sum(manual.trainTable)),4),
                        signif((manual.trainTable[2]/sum(manual.trainTable)),4),
                        signif((manual.trainTable[3]/sum(manual.trainTable)),4),
                        signif((manual.trainTable[4]/sum(manual.trainTable)),4))
colnames(manual.traingrade) <- c('Grade 1: [0.0.10]', 'Grade 2: (0.10,0.15]',
                                 'Grade 3: (0.15,0.25]', 'Grade 4: (0.25+]')
rownames(manual.traingrade) <- c('Manual Train')
#kable(manual.traingrade)




#####################Summary of all Train data grades###########
#Create table for all Train data grades
```

```r
all.traingrade <- rbind(stepwise.traingrade, subset.traingrade, manual.traingrade)
colnames(all.traingrade) <- c('Grade 1: [0.0.10]', 'Grade 2: (0.10,0.15]',
                              'Grade 3: (0.15,0.25]', 'Grade 4: (0.25+]')
rownames(all.traingrade) <- c('Stepwise Train','Subset Train','Manual Train')
kable(all.traingrade, caption = 'Prediction grades of in-sample data set by model.')


##############################################################
# Test Data
# Abs Pct Error
stepwise.testPCT <- abs(test.df$TARGET_WINS-stepwise.test)/test.df$TARGET_WINS;
subset.testPCT <- abs(test.df$TARGET_WINS-subset.test)/test.df$TARGET_WINS;
manual.testPCT <- abs(test.df$TARGET_WINS-manual.test)/test.df$TARGET_WINS;


# Assign Prediction Grades;
###############Stepwise.test##################################
stepwise.testPredictionGrade <- ifelse(stepwise.testPCT<=0.10,'Grade 1: [0.0.10]',
                                      ifelse(stepwise.testPCT<=0.15,'Grade 2: (0.10,0.15]',
                                             ifelse(stepwise.testPCT<=0.25,'Grade 3: (0.15,0.25]',
                                                    'Grade 4: (0.25+]')
                                      )
)
stepwise.testTable <-table(stepwise.testPredictionGrade)
stepwise.testTable/sum(stepwise.testTable)
#Create table for paper
stepwise.testgrade <- cbind(signif((stepwise.testTable[1]/sum(stepwise.testTable)),4),
                       signif((stepwise.testTable[2]/sum(stepwise.testTable)),4),
                       signif((stepwise.testTable[3]/sum(stepwise.testTable)),4),
                       signif((stepwise.testTable[4]/sum(stepwise.testTable)),4))
colnames(stepwise.testgrade) <- c('Grade 1: [0.0.10]', 'Grade 2: (0.10,0.15]',
                                  'Grade 3: (0.15,0.25]', 'Grade 4: (0.25+]')
rownames(stepwise.testgrade) <- c('Stepwise Test')
#kable(stepwise.testgrade)


#############Subset.test####################################
subset.testPredictionGrade <- ifelse(subset.testPCT<=0.10,'Grade 1: [0.0.10]',
                                      ifelse(subset.testPCT<=0.15,'Grade 2: (0.10,0.15]',
                                             ifelse(subset.testPCT<=0.25,'Grade 3: (0.15,0.25]',
                                                    'Grade 4: (0.25+]')
                                      )
)
subset.testTable <-table(subset.testPredictionGrade)
subset.testTable/sum(subset.testTable)
#Create table for paper
subset.testgrade <- cbind(signif((subset.testTable[1]/sum(subset.testTable)),4),
                        signif((subset.testTable[2]/sum(subset.testTable)),4),
                        signif((subset.testTable[3]/sum(subset.testTable)),4),
                        signif((subset.testTable[4]/sum(subset.testTable)),4))
colnames(subset.testgrade) <- c('Grade 1: [0.0.10]', 'Grade 2: (0.10,0.15]',
                                'Grade 3: (0.15,0.25]', 'Grade 4: (0.25+]')
rownames(subset.testgrade) <- c('Subset Test')
```

```
#kable(subset.testgrade)


############Manual.test################################
manual.testPredictionGrade <- ifelse(manual.testPCT<=0.10,'Grade 1: [0.0.10]',
                                       ifelse(manual.testPCT<=0.15,'Grade 2: (0.10,0.15]',
                                              ifelse(manual.testPCT<=0.25,'Grade 3: (0.15,0.25]',
                                                     'Grade 4: (0.25+]')
                                       )
)
manual.testTable <-table(manual.testPredictionGrade)
manual.testTable/sum(manual.testTable)
#Create table for paper
manual.testgrade <- cbind(signif((manual.testTable[1]/sum(manual.testTable)),4),
                        signif((manual.testTable[2]/sum(manual.testTable)),4),
                        signif((manual.testTable[3]/sum(manual.testTable)),4),
                        signif((manual.testTable[4]/sum(manual.testTable)),4))
colnames(manual.testgrade) <- c('Grade 1: [0.0.10]', 'Grade 2: (0.10,0.15]',
                                'Grade 3: (0.15,0.25]', 'Grade 4: (0.25+]')
rownames(manual.testgrade) <- c('Manual Test')
#kable(manual.testgrade)


#####################Summary of all Test data grades###########
#Create table for all Test data grades
all.testgrade <- rbind(stepwise.testgrade, subset.testgrade, manual.testgrade)
colnames(all.testgrade) <- c('Grade 1: [0.0.10]', 'Grade 2: (0.10,0.15]',
                             'Grade 3: (0.15,0.25]', 'Grade 4: (0.25+]')
rownames(all.testgrade) <- c('Stepwise Test','Subset Test','Manual Test')
kable(all.testgrade, caption = 'Prediction grades of out-of-sample data set by model.')


#################### Score Test Data ##########################
#setwd("D:/RFile/")
#moneyball_test=read.csv("moneyball_test.csv",header=T)
moneyball_test <- read.csv("~/Desktop/R/moneyball_test.csv", header = T)

# Fixing na's
#exclude BATTING_HBP from data set before imputation due to high number of missing values
exclude <- c('TEAM_BATTING_HBP')
include <- setdiff(names(moneyball_test), exclude)

#moneyball data set without BATTING_HBP variable now
moneyball_test <- moneyball_test[include]

#Impute the missing data with mice package
#Do not include INDEX during calculations
imp.moneyball_test <- mice(moneyball_test[, !names(moneyball_test) %in% "INDEX"],
                           m=5,maxit=50,meth='pmm',seed=500)
summary(imp.moneyball_test)
completed.moneyball_test <- complete(imp.moneyball_test)
anyNA(completed.moneyball_test)
```

```r
#Straighten Relationships for completed.moneyball - create single hits variable
completed.moneyball_test$TEAM_BATTING_1B <- completed.moneyball_test$TEAM_BATTING_H -
  completed.moneyball_test$TEAM_BATTING_HR - completed.moneyball_test$TEAM_BATTING_3B -
  completed.moneyball_test$TEAM_BATTING_2B

#Straighten Relationships for completed.moneyball - log transformation
#log transformation cannot be done on variables with values of zero
completed.moneyball_test$log_TEAM_BATTING_1B <- log(completed.moneyball_test$TEAM_BATTING_1B)
completed.moneyball_test$log_TEAM_PITCHING_H <- log(completed.moneyball_test$TEAM_PITCHING_H)
completed.moneyball_test$log_TEAM_FIELDING_E <- log(completed.moneyball_test$TEAM_FIELDING_E)

#Set to 1% truncation for lower limit and 99% truncation for upper limit - trimming data
completed.moneyball_test$TEAM_PITCHING_SO[(completed.moneyball_test$TEAM_PITCHING_SO > 1467)] = 1467
completed.moneyball_test$TEAM_PITCHING_SO[(completed.moneyball_test$TEAM_PITCHING_SO < 207)] = 207
completed.moneyball_test$TEAM_PITCHING_H[(completed.moneyball_test$TEAM_PITCHING_H > 7054)] = 7054
completed.moneyball_test$TEAM_PITCHING_BB[(completed.moneyball_test$TEAM_PITCHING_BB > 921)] = 921
completed.moneyball_test$TEAM_PITCHING_BB[(completed.moneyball_test$TEAM_PITCHING_BB < 240)] = 240
completed.moneyball_test$TEAM_FIELDING_E[(completed.moneyball_test$TEAM_FIELDING_E > 1228)] = 1228
completed.moneyball_test$TEAM_BATTING_H[(completed.moneyball_test$TEAM_BATTING_H > 1945)] = 1945

#Straighten Relationships for completed.moneyball - z tranformation / standardizing
completed.moneyball_test$ztrans_TEAM_BATTING_3B <-
(completed.moneyball_test$TEAM_BATTING_3B - mean(completed.moneyball_test$TEAM_BATTING_3B)) /
sd(completed.moneyball_test$TEAM_BATTING_3B)
completed.moneyball_test$ztrans_TEAM_BASERUN_SB <-
(completed.moneyball_test$TEAM_BASERUN_SB - mean(completed.moneyball_test$TEAM_BASERUN_SB)) /
sd(completed.moneyball_test$TEAM_BASERUN_SB)
completed.moneyball_test$ztrans_TEAM_BASERUN_CS <-
(completed.moneyball_test$TEAM_BASERUN_CS - mean(completed.moneyball_test$TEAM_BASERUN_CS)) /
sd(completed.moneyball_test$TEAM_BASERUN_CS)
completed.moneyball_test$ztrans_TEAM_PITCHING_BB <-
(completed.moneyball_test$TEAM_PITCHING_BB - mean(completed.moneyball_test$TEAM_PITCHING_BB)) /
sd(completed.moneyball_test$TEAM_PITCHING_BB)
completed.moneyball_test$ztrans_TEAM_FIELDING_E <-
(completed.moneyball_test$TEAM_FIELDING_E - mean(completed.moneyball_test$TEAM_FIELDING_E)) /
sd(completed.moneyball_test$TEAM_FIELDING_E)
completed.moneyball_test$ztrans_TEAM_BATTING_SO <-
(completed.moneyball_test$TEAM_BATTING_SO - mean(completed.moneyball_test$TEAM_BATTING_SO)) /
sd(completed.moneyball_test$TEAM_BATTING_SO)

completed.moneyball_test$INDEX <- moneyball_test$INDEX

#model  stepwise
completed.moneyball_test$P_TARGET_WINS <- 21.742664 -
  0.061358 * completed.moneyball_test$TEAM_BATTING_H +
  0.066999 * completed.moneyball_test$TEAM_BATTING_2B +
  0.207489 * completed.moneyball_test$TEAM_BATTING_HR +
  0.183708 * completed.moneyball_test$TEAM_BATTING_3B +
  0.052708 * completed.moneyball_test$TEAM_BASERUN_SB +
  0.023797 * completed.moneyball_test$TEAM_BASERUN_CS -
  0.049908 * completed.moneyball_test$TEAM_PITCHING_HR -
  0.045895 * completed.moneyball_test$TEAM_PITCHING_BB +
  0.024564 * completed.moneyball_test$TEAM_PITCHING_SO -
  0.018593 * completed.moneyball_test$TEAM_FIELDING_E +
  0.089688 * completed.moneyball_test$TEAM_BATTING_1B +
  14.493899 * completed.moneyball_test$log_TEAM_PITCHING_H -
  15.137826 * completed.moneyball_test$log_TEAM_FIELDING_E +
  0.067146 * completed.moneyball_test$TEAM_BATTING_BB -
  0.041862 * completed.moneyball_test$TEAM_BATTING_SO -
  0.130568 * completed.moneyball_test$TEAM_FIELDING_DP


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

#####
#Note, this next function will output an Excel file in your work environment called write.xlsx.
#####

#Prediction File
write.xlsx(prediction, file = "write.xlsx", sheetName = "Predictions",
           col.names = TRUE)
```