Money Ball Assignment

4/27/2018

Contents

Introduction

The purpose of this report is to analyze data collected from professional baseball teams between 1871 and 2006. The data contains information on the performance of baseballteams during a given year; with data adjusted to reflect the performance of a 162 game season. In this paper I will perform an exploratory data analysis, prepare data for modeling, explore various models from a predictive modeling prespective, quantify the predictive accuracy of the models using both in-sample and out-of-sample data, and create an optimal model for deployment.

Sample Population

The Money Ball data set has 2,276 obervations and 17 variables. For this paper, I am interested in using an OLS ("Linear") Regression to predict the number of wins for the professional baseball team. For the purpose of exploratory data analysis, I will use the data as is. Later on, I will split my data 70/30 for the purpose of quantifying the predictive accuracy of the models.

Listed below are the descriptions of the 17 variables in the Money Ball dataset.

Variable Name	Definition	Theoretical Effect
TARGET_WINS	Number of wins in a given year	Response Variable
TEAM_BATTING_H	Base Hits by batters (1B,2B,3B,HR)	Positive Impact on Wins
TEAM_BATTING_2B	Doubles by batters (2B)	Positive Impact on Wins
TEAM_BATTING_3B	Triples by batters (3B)	Positive Impact on Wins
TEAM_BATTING_HR	Homeruns by batters (4B)	Positive Impact on Wins
TEAM_BATTING_BB	Walks by batters	Positive Impact on Wins
TEAM_BATTING_HBP	Batters hit by pitch (get a free base)	Positive Impact on Wins
TEAM_BATTING_SO	Strikeouts by batters	Negative Impact on Wins
TEAM_BASERUN_SB	Stolen bases	Positive Impact on Wins
TEAM_BASERUN_CS	Caught stealing	Negative Impact on Wins
TEAM_FIELDING_E	Errors	Negative Impact on Wins
TEAM_FIELDING_DP	Double Plays	Positive Impact on Wins
TEAM_PITCHING_BB	Walks allowed	Negative Impact on Wins
TEAM_PITCHING_H	Hits allowed	Negative Impact on Wins
TEAM_PITCHING_HR	Homeruns allowed	Negative Impact on Wins
TEAM_PITCHING_SO	Strikeouts by pitchers	Positive Impact on Wins

Data Preparation

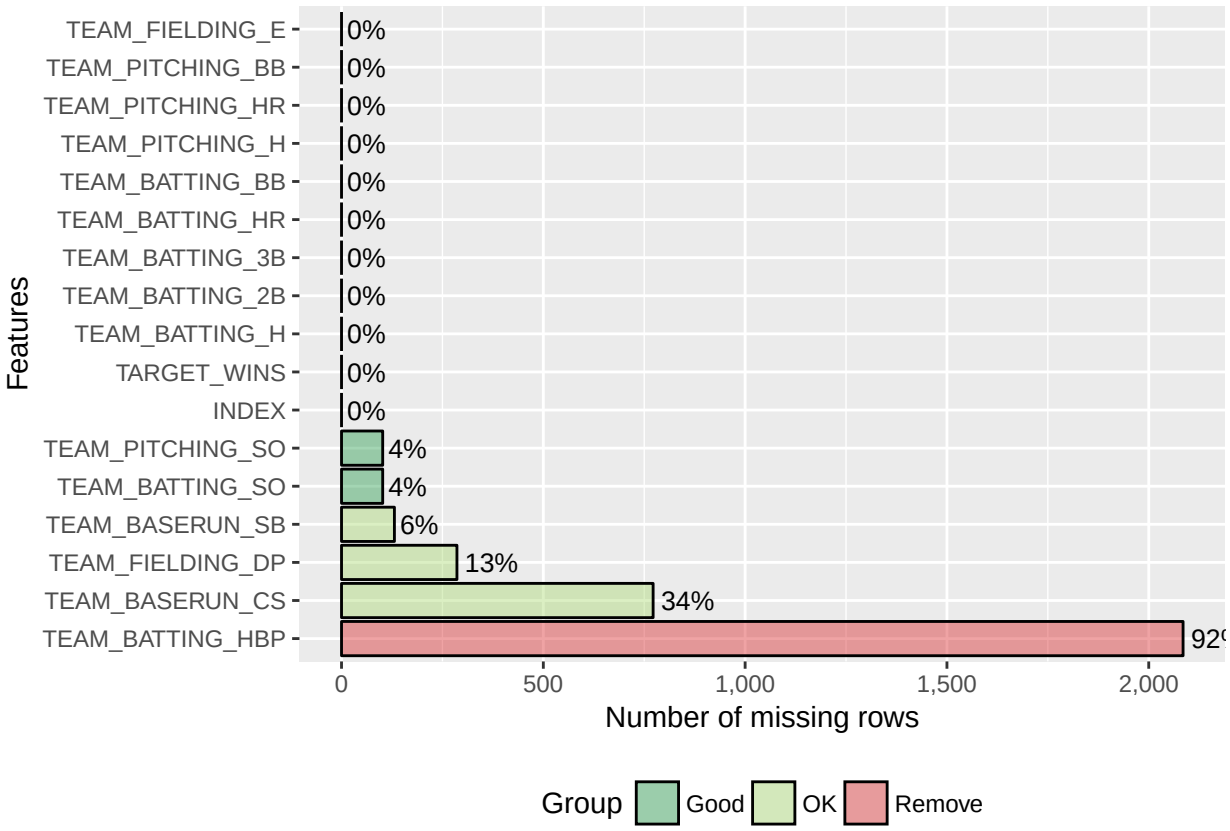
Taking a quick look at the summary statistics, we see that some variables in our dataset have missing values; we'll want to take care of missing values. To gain a deeper understanding of our data, lets go ahead and visually inspect variables with missing values.

	Min.	1st Qu.	Median	Mean	3rd Qu.	Max.	NA's
TARGET_WINS	0	71	82	81	92	146	0
TEAM_BATTING_H	891	1383	1454	1469	1537	2554	891
TEAM_BATTING_2B	69	208	238	241	273	458	69
TEAM_BATTING_3B	0	34	47	55	72	223	0
TEAM_BATTING_HR	0	42	102	100	147	264	0
TEAM_BATTING_BB	0	451	512	502	580	878	0
TEAM_BATTING_SO	0	548	750	736	930	1399	102
TEAM_BASERUN_SB	0	66	101	125	156	697	131
TEAM_BASERUN_CS	0	38	49	53	62	201	772
TEAM_BATTING_HBP	29	50	58	59	67	95	2085
TEAM_PITCHING_H	1137	1419	1518	1779	1682	30132	1137
TEAM_PITCHING_HR	0	50	107	106	150	343	0
TEAM_PITCHING_BB	0	476	536	553	611	3645	0
TEAM_PITCHING_SO	0	615	814	818	968	19278	102
TEAM_FIELDING_E	65	127	159	246	249	1898	65
TEAM_FIELDING_DP	52	131	149	146	164	228	286

Missing Variables

Below we have a plot and a table of the missing values in our dataset. As you can see below, the variable 'Team Batting HBP' has around 92% of the data missing; we'll probably want to exclude this from our analysis. Next we notcie that 'Team Baserun CS' more than a third of its values missing; maybe we can find a way to estimate these missing values. The other variables with missing values ('Team Fielding DP', 'Team Baserun SB', Team Batting SO', and 'Team Pitching SO') have less than 13% of its data missing so we'll have to estimate these values.

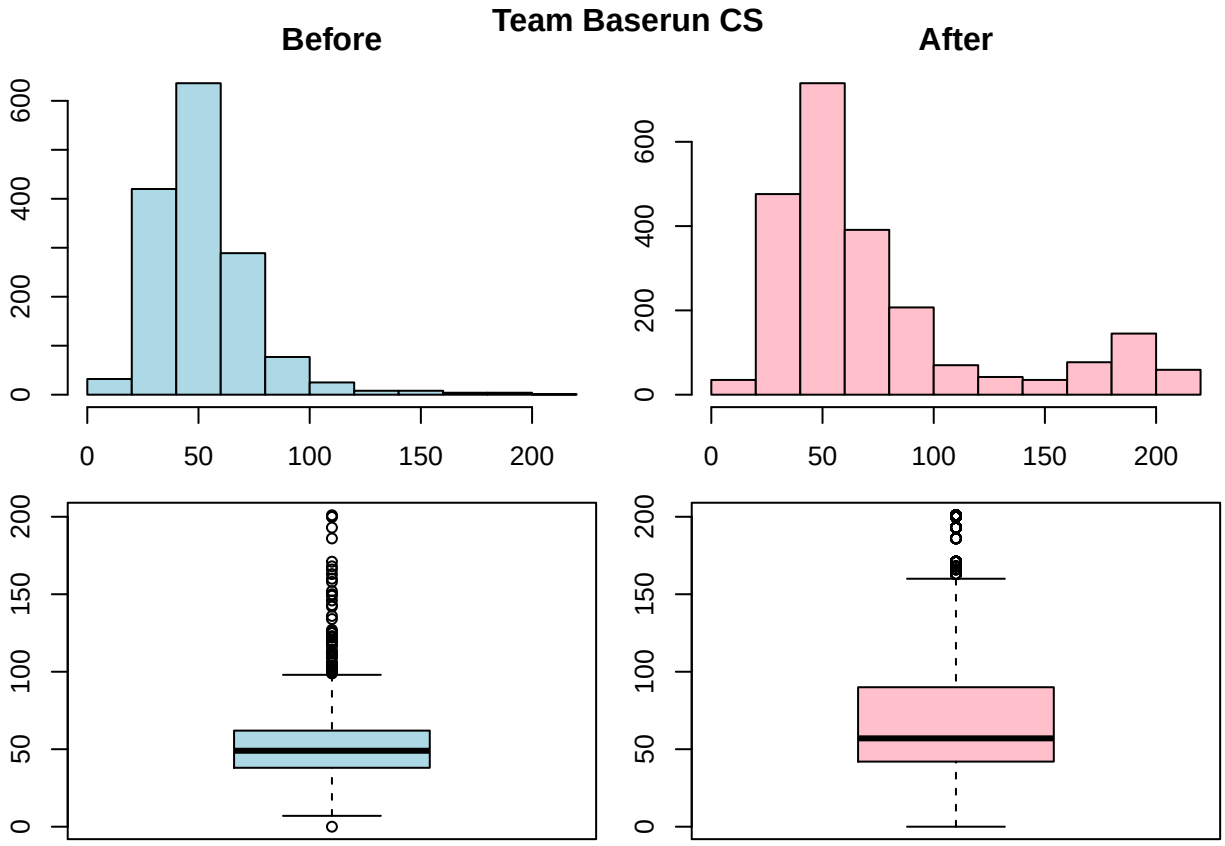
	feature	num_missing	pct_missing	group
11	TEAM_BATTING_HBP	2085	0.9160808	Remove
10	TEAM_BASERUN_CS	772	0.3391916	OK
17	TEAM_FIELDING_DP	286	0.1256591	OK
9	TEAM_BASERUN_SB	131	0.0575571	OK
8	TEAM_BATTING_SO	102	0.0448155	Good
15	TEAM_PITCHING_SO	102	0.0448155	Good
1	INDEX	0	0.0000000	Good
2	TARGET_WINS	0	0.0000000	Good
3	TEAM_BATTING_H	0	0.0000000	Good
4	TEAM_BATTING_2B	0	0.0000000	Good
5	TEAM_BATTING_3B	0	0.0000000	Good
6	TEAM_BATTING_HR	0	0.0000000	Good
7	TEAM_BATTING_BB	0	0.0000000	Good
12	TEAM_PITCHING_H	0	0.0000000	Good
13	TEAM_PITCHING_HR	0	0.0000000	Good
14	TEAM_PITCHING_BB	0	0.0000000	Good
16	TEAM_FIELDING_E	0	0.0000000	Good



Estimate Missing Values

Now that we have found the variables with missing values, lets go ahead and estimate these missing values. To predict these missing values, I am going to use predictive mean matching method (PMM). PMM, according to SAS, is an an imputation method that imputes a value randomly from a set of observed values whose predicted values are closest to the predicted value for the missing value from the simulated regression model - that was a mouthful. Basically, it calculates the predicted value using a regression model and picks the closest 'real' value to the predicted value.

Now that we've estimated the missing values, lets show a before and after picture of the 'Team Baserun CS' variable.



As you can see above, the imputation process resulted in the concentration of data around the peak of the histogram and expanded the boxplot spread between the 25th and 7th percentile. With that said, its important to note that the imputation process did not change the orginal skewness nor did it change the lower or upper bound of the original data.

Additional Features

Before moving on to doing Exploratory Data Analysis, lets go ahead and create dummary variables to identify the imputed/missing values. While we are at it, lets create a postive and negative dummary variable which are just the total sum of variables that have postive and negative impacts, respectively.

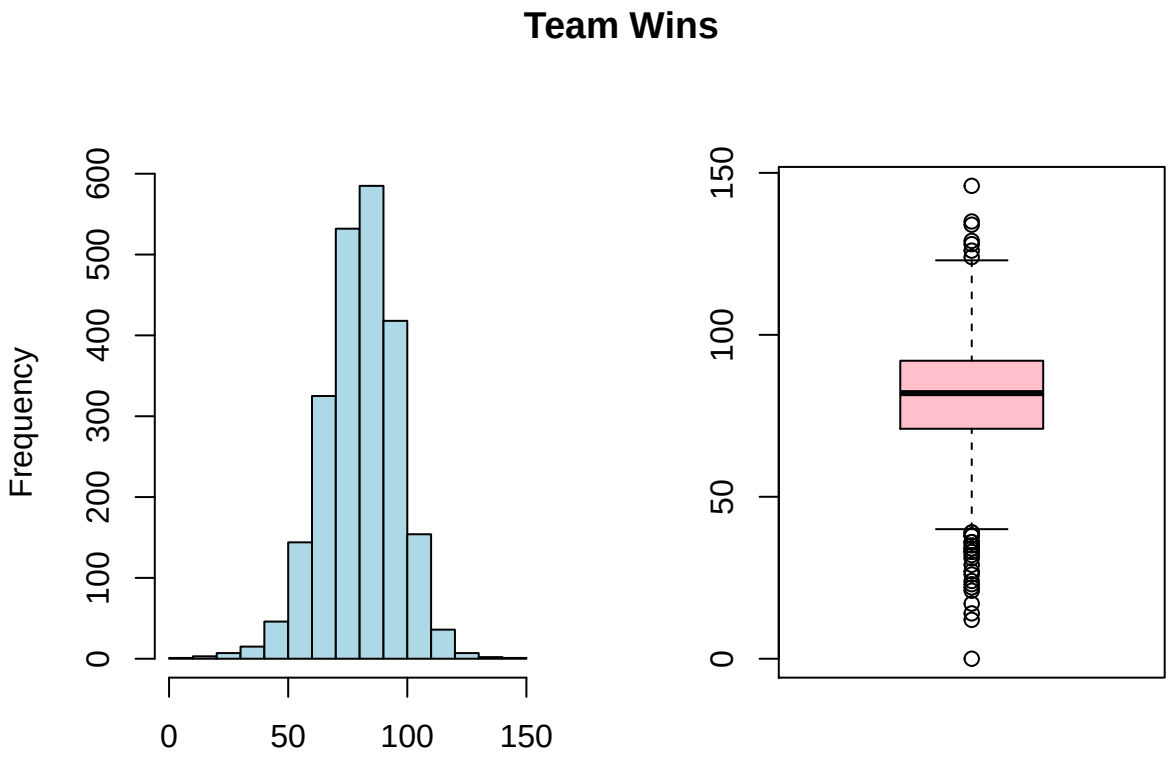
Dummy Variable	Logic
Team_Baserun_CS_NA_Ind	Value equal to 1 if estimated, 0 if not.
Team_Fielding_DP_NA_Ind	Value equal to 1 if estimated, 0 if not.
Team_Baserun_SB_NA_Ind	Value equal to 1 if estimated, 0 if not.
Team_Batting_SO_NA_Ind	Value equal to 1 if estimated, 0 if not.
Team_Pitching_SO_NA_Ind	Value equal to 1 if estimated, 0 if not.
Postive_Impact_Var	Sum of Team_Batting_H, Team_Batting_2B, Team_Batting_3B, Team_Batting_HR, Team_Batting_BB, Team_Baserun_SB, Team_Fielding_DP, Team_Pitching_SO
Negative_Impact_Var	Sum of Team_Batting_SO, Team_Baserun_CS, Team_Fielding_E, Team_Pitching_BB, Team_Pitching_H, Team_Pitching_HR

Exploratory Data Analysis

Target Variables

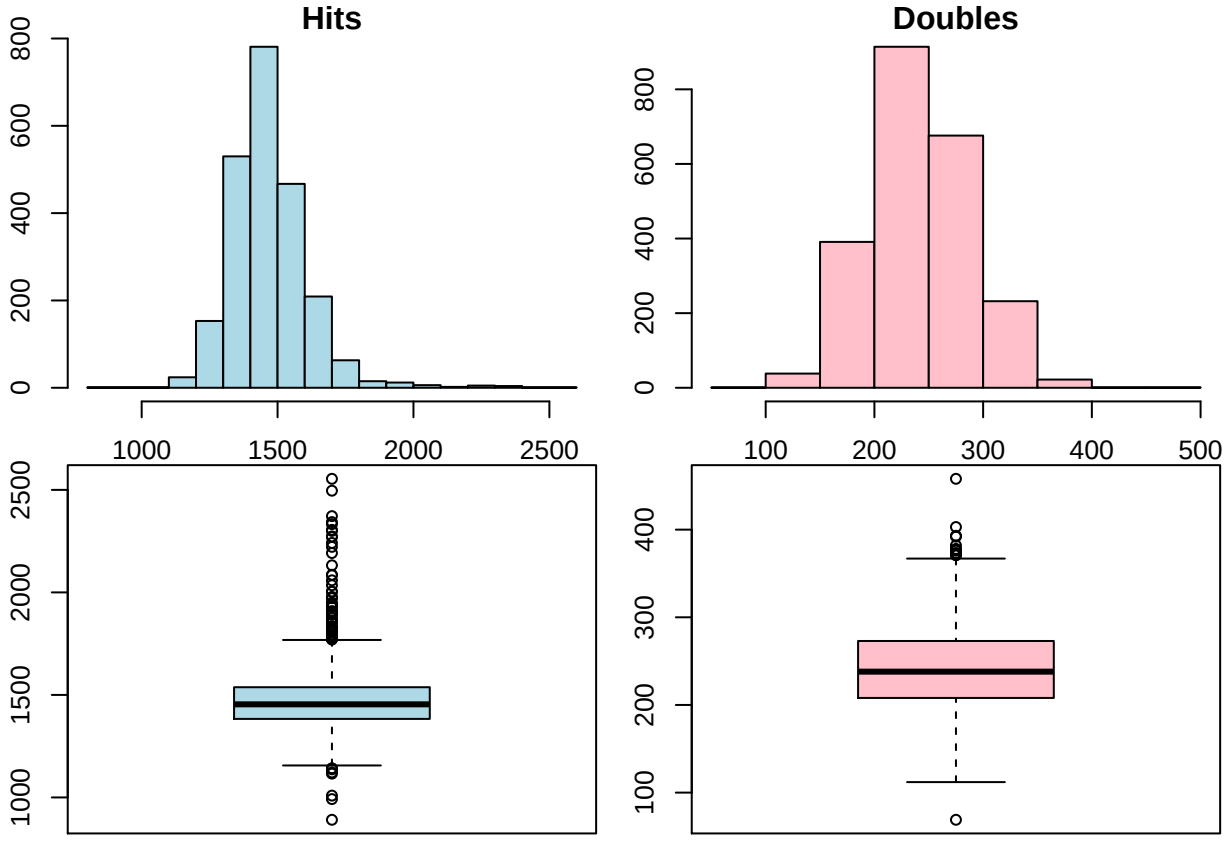
Target Wins

Now that we've taken care of the missing values, lets start our exploratory data analysis by looking at what we want to predict, the 'Target_Wins' variable. As you can see below, the plot seems to indicate that the 'Target_Wins' variable is normally distributed with mean of 81.



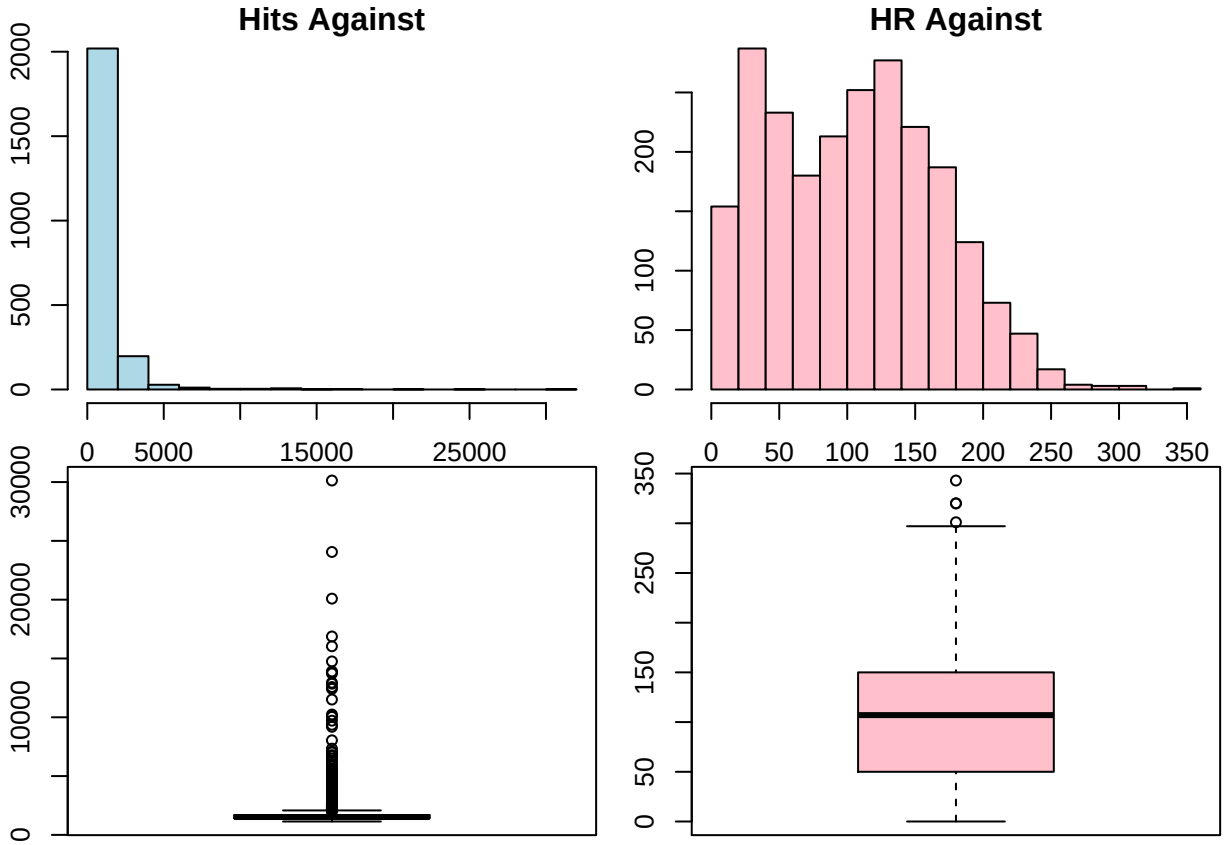
Batting Variables

Next, lets pick a couple of batting variables. For this analysis, I've picked the 'Hits' and 'Doubles' variables. As you can see below, the Hits variable is skewed to the right with some teams having disproportionately larger number of hits in a given year. The boxplot seems to confirm this On the other hand, the Doubles variable seems to follow a some what narrow normall distribution; some skewness to the right is visable.



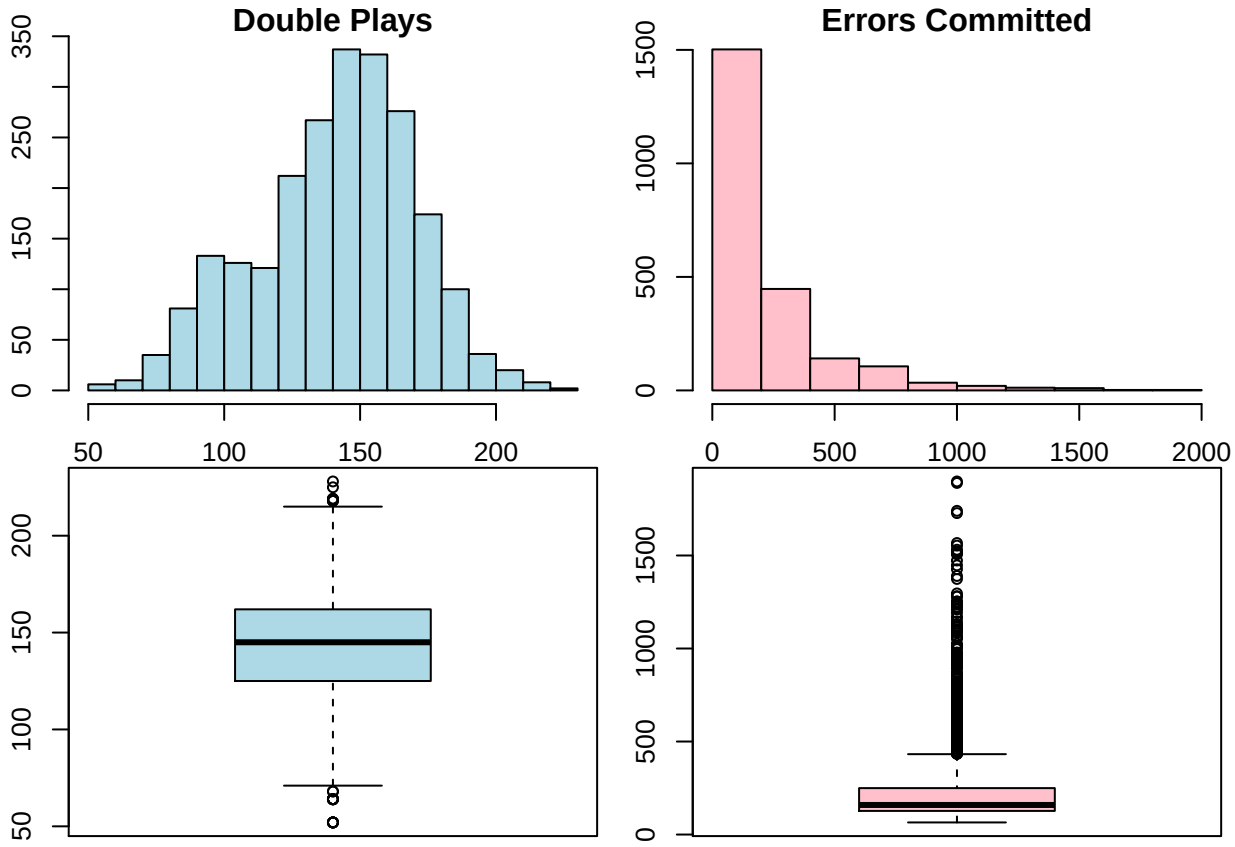
Pitching Variables

Next, lets pick a couple of pitching variables. For this analysis, I've picked the 'Hits Against' and 'Home Run Against' variables.As you can see below, the 'Hits Against' variable is skewed to the right with some teams having disproportionately larger number of hits in a given year. The boxplot seems to confirm this. On the other hand, the 'Home Run Against' variable seems to have some slight skewness to the right. Further, there seems to be some concentration of data between the mid to lower bound section of the distribution.



Fielding Variables

At last, lets pick a couple of fielding variables. For this analysis, I've picked the 'Double Plays' and 'Errors Committed' variables. As you can see below, the 'Double Plays' variable is probably the most 'normal distribution like' variable we have explored in this paper so far. The histogram seems to indicate that there is a concentration of data towards the middle of the distribution with almost symmetric deviation from the mean on both sides of the tail. A further examination of the boxplot seems to show that there is some slight skewness to the right. Taking a look at the 'Errors Committed' variable, you can see that the variable is clearly skewed heavily to the right.

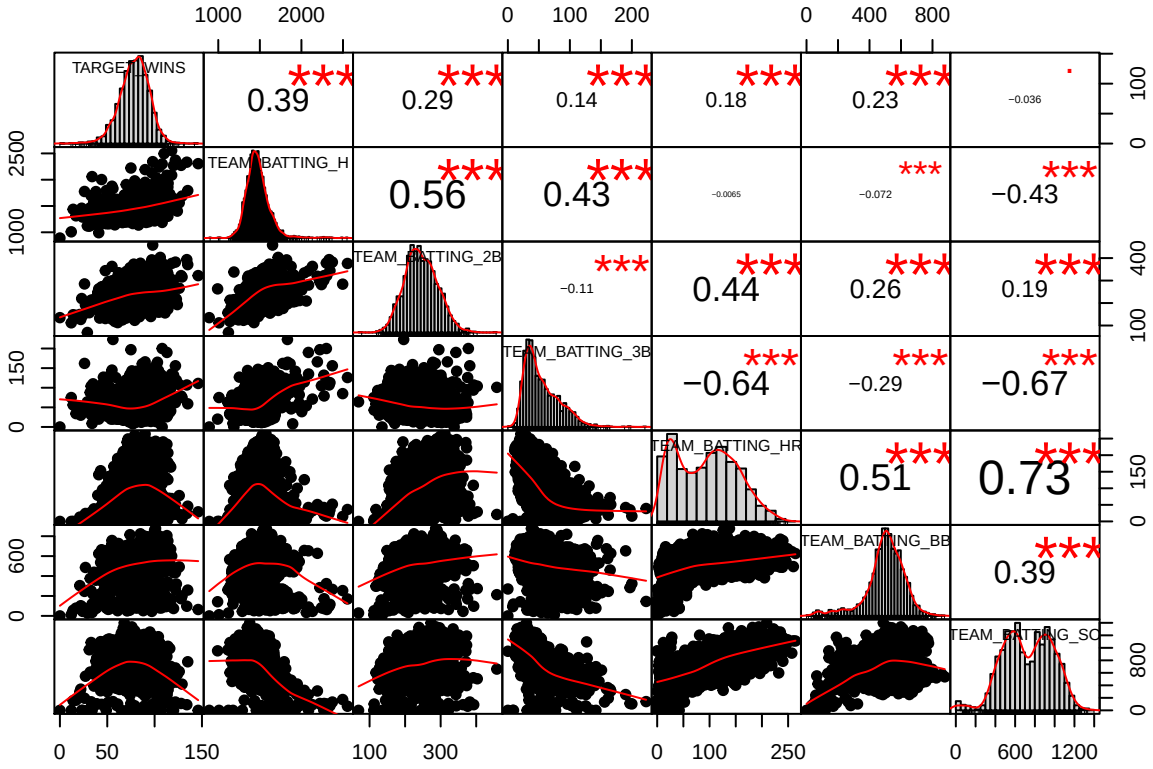


Correlation

Now that we have visually explored our dataset, lets now see how these variables correlate to our response variable and to each other.

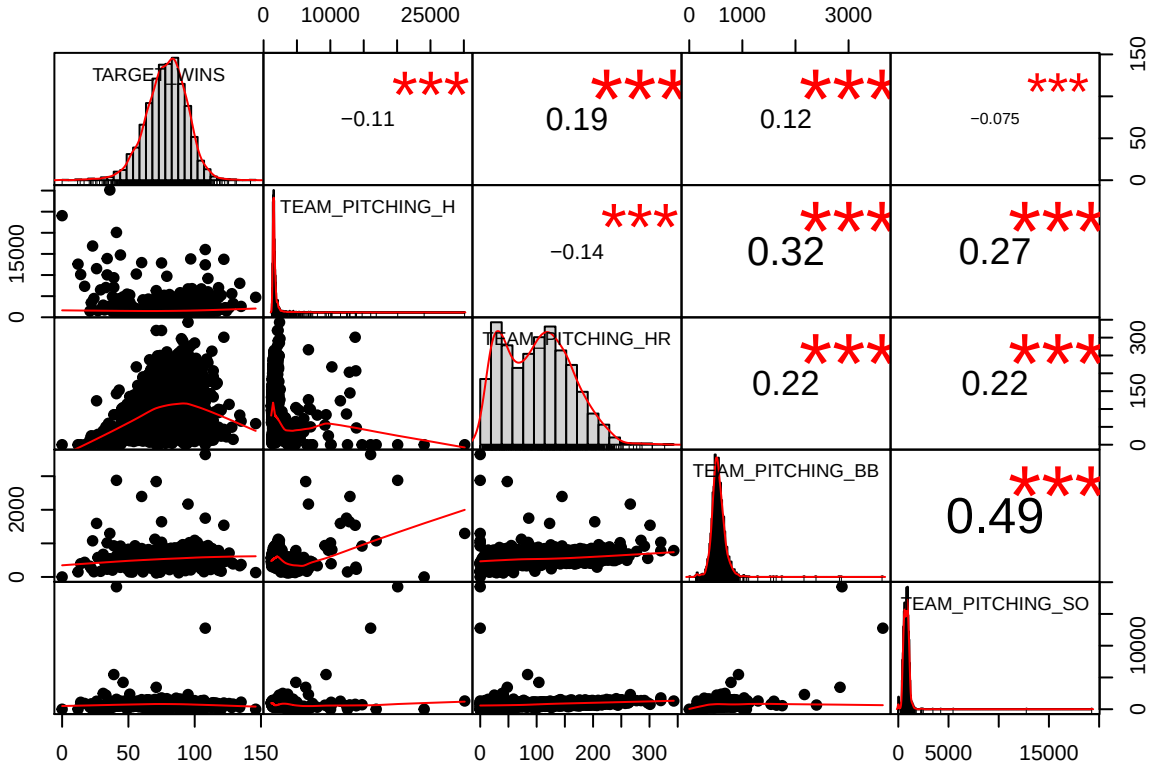
Batting Variables

Starting off with the batting variables, we can see that the variable 'Team Batting H' is the highest correlated variable with the 'Target Wins' variable. Further, taking a look at how the batting variables relate to each other, we can see that 'Team Batting SO' and 'Team Batting HR' are highly correlated; we might want to consider keeping one of the other variable in our model to minimize multicollinearity. 'Team Batting SO' is also highly correlated with other batting variables like 'Team Batting 3B' and 'Team Batting H'; it might be worth just keeping the 'Team Batting SO' and leaving the other variables.



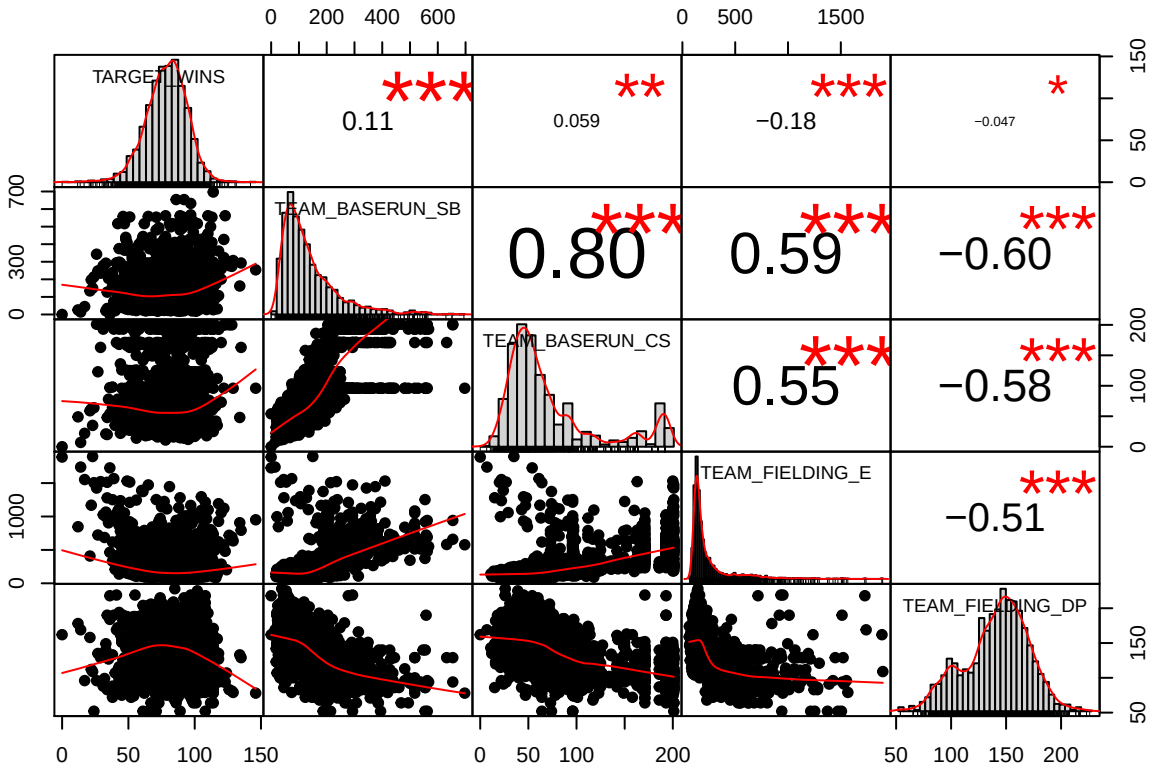
Pitching Variables

Moving on to our pitching variables, we can see that the 'Target Wins' variable doesn't have any 'significant' correlation with the pitching variables. Further, the 'Team Pitching SO' variable seems to have a relatively high correlation with the other pitching variables; in particular, 'Team Pitching BB' and 'Team Pitching H'. We probably want to keep 'Team Pitching H' in mind when constructing our model since the inclusion of this variable may lead to some multicollinearity in our model.



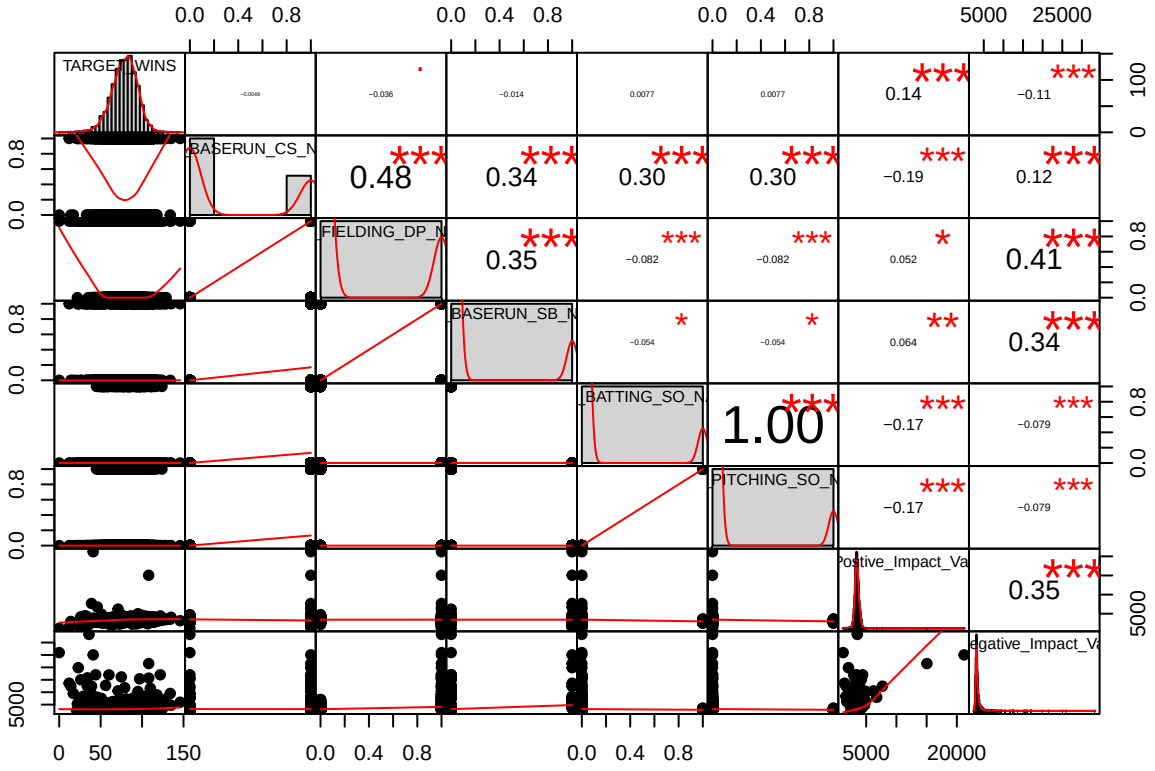
Fielding & Baserun Variables

Next, lets take a quick look at at our fielding and baserun variables. As you can see below, the 'Target Wins' variable doesn't have any 'significant' correlation with the fielding or baserun variables. Interestingly, the 'Team Baserun SB' variable has a really high correlating with the 'Team Baserun CS'. In addition, 'Team Baserun SB' is highly correlated with the 'Team Fielding E' and 'Team Fielding DP' variables; we'll definitely have to cherrypick the 'Team Baserun SB' to avoid multicollinearity.



Feature Engineered Variables

Finally, lets take a look at the variables we created and how they correlate with the 'Team Wins' variable and to each other. As a quick recap of these variables; with the exception of the 'Positive Impact' and 'Negative Impact' variables, these are summary variables used to flag values that were imputed. At a quick glance of the correlation chat, the variable seem to have almost no correlation with our response variable; big surprise! Further, the 'Team Batting SO Ind' and 'Team Pitching SO Ind' have perfect correlation indicating that when one variable was omitted, so was the other.



Model Development

Train/Test Split

To assess the Predictive Accuracy and Validity of our models, our moneyball data set was split into two groups; training and testing datasets. The spilt was done using a random uniform distribution such that roughly 70% of our data is used to train our models and the remaining 30% is used to test our models out-of-sample accuracy. The number of observations in each table is shown below.

Data	Count
Moneyball	2,276
Training	1,598
Testing	678

Variable Selection

In this paper, I will use the backwards, forwards, and stepwise variable selection methods to build regression models.

Backward Variable Selection

I will start building my first model using Backward Variable Selection (BVS). With BVS, we start off with a current model that includes all of the explanatory variables. Next, we consider a candidate model that is different than the current model by the removal of one explanatory variable. If the candidate model has an AIC value that is less than our current model, we accept the candidate model as our current model. We then repeat the same steps until all explanatory variables are tested such that the current model has the lowest AIC value relative to all candidate models.

Using our training dataset, we've built a BVS model below. Recall that our training dataset has 1,598 observations, 21 numerical explanatory variables, and 1 response variable. Taking a look at the summary results below, we can see that our BVS model has 13 out of the 21 explanatory variables included in the model and contributes to an adjusted R-squared of around 0.3973 and an AIC value of around 12,541. Note that not every explanatory variable is statistically significant at the 0.01 significance level.

Taking a closer look at our BVS model, we can see that the MAE value is around 9.6 and the Root MSE value is around 12. This implies that our BVS model on average is within 9.6 to 12 of the actual team win for a given year.

Model	Adjusted R Squared	AIC	BIC	RMSE	MAE
Backward Variable Selection	0.3973	12541	12617	12	9.6

Next, lets examine the explanatory variables to see if there are multicollinearity between variables. The Variance Inflation Factor (VIF) is a great way to quantify the severity of multicollinearity in our BVS model. Taking a quick look at the top 5 variables with high VIF values, we see that we don't have much multicollinearity across the explanatory variables. 'Team Batting HR' seems to be the variable with the highest VIF value of around 4.58. Personally, I like my VIF values to be around 3 since this corresponds to an R squared value of 0.67 between explanatory variables. In the case of 'Team Batting HR', the VIF value of 4.58 corresponds to an R squared value of 0.78 which is relatively high for my liking.

	Backward Variable Selection
TEAM_BATTING_HR	4.582812
TEAM_BATTING_H	3.596138
TEAM_BATTING_3B	2.725529
TEAM_BATTING_2B	2.460671
TEAM_BATTING_BB	2.344735

Now that we've built our BVS model, calculated the MAE and Root MSE amount, and confirmed that we our model doesn't show extremely high multicollinearity; its time that we test our model predictive accuracy. Recalculating the MAE and Root MSE amount using the testing data set below, the results seem to indicate that our BVS model on average is within 9.4 to 12 of the actual team win for a given year.

Nice! Our model performed fairly well on the testing data set! Now lets check if there is a better automated variable selection method that we haven't tested. Lets move on to the Forward Variable Selection method and see if we could build a better model.

Model	RMSE	MAE
Backward Variable Selection	12	9.4

Forward Variable Selection

Next, we will build our second model using Forward Variable Selection (FVS). With FVS, we start off with a current model that includes no explanatory variables but just the y-intercept. We then consider a candidate model that is different than the current model by the addition of one explanatory variable. If the candidate model has an AIC value that is less than our current model, we accept the candidate model as our current model. We then repeat the same steps until all explanatory variables are tested such that the current model has the lowest AIC value relative to all candidate models.

Using our training dataset, we've built a FVS model below. Taking a look at the summary results below, we can see that our FVS model has 14 out of the 21 explanatory variables included in the model and contributes to an adjusted R-squared of around 0.3969 and an AIC value of around 12,543. Note that not every explanatory variable is statistically significant at the 0.01 significance level. Further, there are some explanatory variables included in the model with significantly high p-values; in particular 'Team Pitching HR' with a p-value of 0.6202.

Taking a closer look at our FVS model, we can see that the MAE value is around 9.6 and the Root MSE value is around 12. This implies that our FVS model on average is within 9.6 to 12 of the actual team wins in a given year.

Model	Adjusted R Squared	AIC	BIC	RMSE	MAE
Forward Variable Selection	0.3969	12543	12624	12	9.6

Next, lets examine the explanatory variables to see if there is multicollinearity between variables. Taking a quick look at the top 5 variables with high VIF values, we see that we might have multicollinearity across the explanatory variables.The 'Team Fielding E' variable seems to have a hight VIF value of around 6.51; this corresponds to an R squared value of 0.85 across explanatory variables. With that said, it might be worth addressing this issue before moving forward with this model as is.

	Forward Variable Selection
TEAM_FIELDING_E	6.507384
TEAM_BATTING_H	3.616458
TEAM_BASERUN_SB	2.542020
TEAM_BATTING_BB	2.344738
TEAM_BASERUN_SB_NA_IND	2.331826

Now that we've built our FVS model, calculated the MAE and Root MSE amount, and somewhat confirmed that our model doesn't show extremely high multicollinearity; its time to test our models predictive accuracy. Recalculating the MAE and Root MSE amount using the testing data set below, the results seem to indicate that our FVS model on average is within 9.4 to 12 of the actual team wins for a given year.

Nice! Our model performed fairly well on the testing data set! Now lets check if there is a better automated variable selection method that we haven't tested. Lets move on to the Stepwise Variable Selection method and see if we could build a better model.

Model	RMSE	MAE
Forward Variable Selection	12	9.4

Stepwise Variable Selection

Next, we will build our third model using Stepwise Variable Selection (SVS). With SVS, we start off with a current model that includes no explanatory variables but just the y-intercept. We then consider a candidate model that is different than the current model by the addition and subtraction of one explanatory variable. If the candidate model has an AIC value that is less than our current model, then we accept the candidate model as our current model. We then repeat the same steps until all explanatory variables are tested such that the current model has the lowest AIC value relative to all candidate models.

Using our training dataset, we've built a SVS model below. Taking a look at the summary results below, we can see that our SVS model has 13 out of the 21 explanatory variables included in the model and contributes to an adjusted R-squared of around 0.3973 and an AIC value of around 12,541. Note that not every explanatory variable is statistically significant at the 0.01 significance level.

Taking a closer look at our SVS model, we can see that the MAE value is around 9.6 and the Root MSE value is around 12. This implies that our SVS model on average is within 9.6 to 12 of the actual team wins in a given year.

Model	Adjusted R Squared	AIC	BIC	RMSE	MAE
Stepwise Variable Selection	0.3973	12541	12617	12	9.6

Next, lets examine the explanatory variables to see if there is multicollinearity between variables. Taking a quick look at the top 5 variables with high VIF values, we see that we might have multicollinearity across the explanatory variables.The 'Team Fielding E' variable seems to have a hight VIF value of around 6.50; this corresponds to an R squared value of 0.85 across explanatory variables. With that said, it might be worth addressing this issue before moving forward with this model as is.

	Stepwise Variable Selection
TEAM_FIELDING_E	6.503584
TEAM_BATTING_H	3.596138
TEAM_BASERUN_SB	2.540797
TEAM_BATTING_BB	2.344735
TEAM_BASERUN_SB_NA_IND	2.331532

Now that we've built our SVS model, calculated the MAE and Root MSE amount, and confirmed that our model doesn't show extremely high multicollinearity; its time that we test our models predictive accuracy. Recalculating the MAE and Root MSE amount using the testing data set below, the results seem to indicate that our SVS model on average is within 9.4 to 12,1 of the actual team wins in a given year.

Nice! Our model performed fairly well on the testing data set! Now lets check if there is a better automated variable selection method that we haven't tested. Lets move on to our User Model and see if it performs better.

Model	RMSE	MAE
Stepwise Variable Selection	12	9.4

User Defined Model

Next, we will build our final model which we will cal our User model; which will be a model that uses predefined explanatory variables chosen by me.The User model has just 6 out of the 21 explanatory variables included in the model and contributes to an adjusted R-squared of 0.3135.

Taking a closer look at our User model, we can see that the MAE value is around 10.2 and the Root MSE value is around 12.9. This implies that our Junk model on average is within 10.2 to 12.9 of the actual team wins in a given year.

Model	Adjusted R Squared	AIC	BIC	RMSE	MAE
User Model	0.3135	12744	12787	12.9	10.2

Next, lets examine the explanatory variables to see if there is multicollinearity between variables. Taking a quick look at the 6 variables with high VIF values, we see that we don't have multicollinearity across the explanatory variables. The 'Team Batting 3B' variable seems to have a hight VIF value of around 2.35; this corresponds to an R squared value of 0.57 across explanatory variables, which is great!

	User Variable Selection
TEAM_BATTING_3B	2.345127
TEAM_BASERUN_SB	2.071671
TEAM_PITCHING_HR	2.022323
TEAM_FIELDING_E	1.934565
TEAM_FIELDING_DP	1.781817
TEAM_BATTING_H	1.561949

Now that we've built our User Model, calculated the MAE and Root MSE amount, and confirmed that our model doesn't have multicollinearity; its time that we test our models predictive accuracy. Recalculating the MAE and Root MSE amount using the testing data set below, the results seem to indicate that our User Model on average is within 10 to 12.7 of the actual team wins in a given year.

Model	RMSE	MAE
User Model	12.7	10

Model Comparison

Now that we've built all four of our models, lets see how they compare relative to each other using our training data set.

Looking at the the Rank of Adjust R-Squared column, we can see that our BVS and SVS models performed the best. While its nice that these models had the highest adjusted R-squared, we know that high Adjust R-Squared values could easily be lipstick on a pig. With this in mind, we'll note that our BVS and SVS models have the highest Adjust R-Squared value, but will look to validate this through other metrics.

Looking at the AIC values, we see that the SVS model had the lowest AIC value. BVS and SVS models both came in first while our User model came in last. Taking a look at the BIC values, we see the exact same thing; BVS and SVS models both came in first while our User model came in last.

Taking a look at the RMSE amounts We see that the BVS, SVS, and FVS models came in first. Looking at the MAE amounts, we see that BVS and SVS came in first place.

At a high level, it seems that our BVS and SVS models had the highest Adjust R-Squared, lowest MAE and RMSE values, and the lowest AIC and BIC values. Our in-sample modeling seems to indicate that the BVS and SVS models are the best performers.

Model	Adjusted R Squared	AIC	BIC	RMSE	MAE
Backward Variable Selection	0.3973	12541	12617	12.049	9.553
Forward Variable Selection	0.3969	12543	12624	12.049	9.554
Stepwise Variable Selection	0.3973	12541	12617	12.049	9.553
User Variable Selection	0.3135	12744	12787	12.883	10.229

Predictive Accuracy

Now that we know which model perform well on our training data set, the logical next step is to see how they perform on our testing data set.

Below I show how our models performed against each other with the testing dataset. Clearly, our BVS and SVS models performed the best and our User model performed the worst. Surprisingly, the model that predicted best in-sample also predicted best out-of-sample. With this in mind, I'll select the SVS model as our optimal model.

Model	Rank of MSE	Rank of MAE
Backward Variable Selection	11.951	9.431
Forward Variable Selection	11.954	9.435
Stepwise Variable Selection	11.951	9.431
User Variable Selection	12.654	9.967

Model Selection

As mentioned above, I've selected the SVS model as the model. The SVS was tied for first place for having the largest adjusted R-squared, the lowest AIC, and low MAE and RMSE values in our in-sample analysis. When looking the out-of-sample analysis, we see that the SVS model lied for first place for having the lowest MAE and RMSE values. With this data at hand, I've selected the SVS model as our preferred model; please see coefficients of the model below.

Variables	Coefficients
Intercept	+ 30.7026
TEAM_BATTING_H	+ 0.0431
TEAM_BATTING_2B	- 0.0338
TEAM_BATTING_3B	+ 0.0647
TEAM_BATTING_HR	+ 0.0795
TEAM_BATTING_BB	+ 0.0251
TEAM_BATTING_SO	- 0.0155
TEAM_BASERUN_SB	+ 0.0481
TEAM_PITCHING_H	+ 0.0024
TEAM_FIELDING_E	- 0.0601
TEAM_FIELDING_DP	- 0.1102
TEAM_BASERUN_SB_NA_IND	+ 24.6870
TEAM_BATTING_SO_NA_IND	+ 6.8564

Appendix

BVS

```
##
## Call:
## lm(formula = TARGET_WINS ~ TEAM_BATTING_H + TEAM_BATTING_2B +
##     TEAM_BATTING_3B + TEAM_BATTING_HR + TEAM_BATTING_BB + TEAM_BATTING_SO +
##     TEAM_BASERUN_SB + TEAM_PITCHING_H + TEAM_FIELDING_E + TEAM_FIELDING_DP +
##     TEAM_BASERUN_SB_NA_IND + TEAM_BATTING_SO_NA_IND, data = train.df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -48.302  -8.194   0.068   7.974  49.654
##
## Coefficients:
##                           Estimate Std. Error t value Pr(>|t|)
## (Intercept)            30.7025931  6.0649606   5.062 4.62e-07 ***
## TEAM_BATTING_H          0.0431198  0.0040594  10.622  < 2e-16 ***
## TEAM_BATTING_2B        -0.0338159  0.0101798  -3.322 0.000915 ***
## TEAM_BATTING_3B         0.0646564  0.0182209   3.548 0.000399 ***
## TEAM_BATTING_HR         0.0795278  0.0106159   7.491 1.13e-13 ***
## TEAM_BATTING_BB         0.0251296  0.0038272   6.566 6.98e-11 ***
## TEAM_BATTING_SO        -0.0155244  0.0025628  -6.058 1.72e-09 ***
## TEAM_BASERUN_SB         0.0480798  0.0049736   9.667  < 2e-16 ***
## TEAM_PITCHING_H         0.0023612  0.0003327   7.097 1.92e-12 ***
## TEAM_FIELDING_E        -0.0605403  0.0034597 -17.499  < 2e-16 ***
## TEAM_FIELDING_DP       -0.1102293  0.0147761  -7.460 1.42e-13 ***
## TEAM_BASERUN_SB_NA_IND 24.6870362  2.0367283  12.121  < 2e-16 ***
## TEAM_BATTING_SO_NA_IND  6.8564136  1.6451900   4.168 3.24e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 12.1 on 1588 degrees of freedom
## Multiple R-squared:  0.4018, Adjusted R-squared:  0.3973
## F-statistic: 88.88 on 12 and 1588 DF,  p-value: < 2.2e-16
```

FVS

```
##
## Call:
## lm(formula = TARGET_WINS ~ TEAM_BATTING_H + TEAM_FIELDING_E +
##     TEAM_BASERUN_SB + TEAM_BASERUN_SB_NA_IND + TEAM_BATTING_BB +
##     TEAM_FIELDING_DP + TEAM_PITCHING_H + TEAM_BATTING_2B + TEAM_BATTING_SO_NA_IND +
##     TEAM_PITCHING_HR + TEAM_BATTING_SO + TEAM_BATTING_3B + TEAM_BATTING_HR,
##     data = train.df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -48.064  -8.174   0.054   7.982  49.539
##
## Coefficients:
##                           Estimate Std. Error t value Pr(>|t|)
## (Intercept)            30.8847006  6.1061497   5.058 4.73e-07 ***
```

```
## TEAM_BATTING_H          0.0430396  0.0040720  10.570  < 2e-16 ***
## TEAM_FIELDING_E        -0.0605623  0.0034617 -17.495  < 2e-16 ***
## TEAM_BASERUN_SB         0.0480511  0.0049763   9.656  < 2e-16 ***
## TEAM_BASERUN_SB_NA_IND 24.6810236  2.0374539  12.114  < 2e-16 ***
## TEAM_BATTING_BB         0.0251285  0.0038283   6.564 7.08e-11 ***
## TEAM_FIELDING_DP       -0.1101045  0.0147881  -7.445 1.58e-13 ***
## TEAM_PITCHING_H         0.0023381  0.0003442   6.793 1.54e-11 ***
## TEAM_BATTING_2B        -0.0337475  0.0101861  -3.313 0.000943 ***
## TEAM_BATTING_SO_NA_IND  6.8641764  1.6459373   4.170 3.21e-05 ***
## TEAM_PITCHING_HR        0.0057859  0.0220062   0.263 0.792646
## TEAM_BATTING_SO        -0.0155439  0.0025646  -6.061 1.69e-09 ***
## TEAM_BATTING_3B         0.0639407  0.0184284   3.470 0.000535 ***
## TEAM_BATTING_HR         0.0734490  0.0254424   2.887 0.003944 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 12.1 on 1587 degrees of freedom
## Multiple R-squared:  0.4018, Adjusted R-squared:  0.3969
## F-statistic:    82 on 13 and 1587 DF,  p-value: < 2.2e-16
```

SVS

```
##
## Call:
## lm(formula = TARGET_WINS ~ TEAM_BATTING_H + TEAM_FIELDING_E +
##     TEAM_BASERUN_SB + TEAM_BASERUN_SB_NA_IND + TEAM_BATTING_BB +
##     TEAM_FIELDING_DP + TEAM_PITCHING_H + TEAM_BATTING_2B + TEAM_BATTING_SO_NA_IND +
##     TEAM_BATTING_SO + TEAM_BATTING_3B + TEAM_BATTING_HR, data = train.df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -48.302  -8.194   0.068   7.974  49.654
##
## Coefficients:
##                          Estimate Std. Error t value Pr(>|t|)
## (Intercept)            30.7025931  6.0649606   5.062 4.62e-07 ***
## TEAM_BATTING_H          0.0431198  0.0040594  10.622  < 2e-16 ***
## TEAM_FIELDING_E        -0.0605403  0.0034597 -17.499  < 2e-16 ***
## TEAM_BASERUN_SB         0.0480798  0.0049736   9.667  < 2e-16 ***
## TEAM_BASERUN_SB_NA_IND 24.6870362  2.0367283  12.121  < 2e-16 ***
## TEAM_BATTING_BB         0.0251296  0.0038272   6.566 6.98e-11 ***
## TEAM_FIELDING_DP       -0.1102293  0.0147761  -7.460 1.42e-13 ***
## TEAM_PITCHING_H         0.0023612  0.0003327   7.097 1.92e-12 ***
## TEAM_BATTING_2B        -0.0338159  0.0101798  -3.322 0.000915 ***
## TEAM_BATTING_SO_NA_IND  6.8564136  1.6451900   4.168 3.24e-05 ***
## TEAM_BATTING_SO        -0.0155244  0.0025628  -6.058 1.72e-09 ***
## TEAM_BATTING_3B         0.0646564  0.0182209   3.548 0.000399 ***
## TEAM_BATTING_HR         0.0795278  0.0106159   7.491 1.13e-13 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 12.1 on 1588 degrees of freedom
## Multiple R-squared:  0.4018, Adjusted R-squared:  0.3973
## F-statistic: 88.88 on 12 and 1588 DF,  p-value: < 2.2e-16
```

User Model

```
##
## Call:
## lm(formula = TARGET_WINS ~ TEAM_BATTING_H + TEAM_BATTING_3B +
##     TEAM_BASERUN_SB + TEAM_PITCHING_HR + TEAM_FIELDING_E + TEAM_FIELDING_DP,
##     data = train.df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -46.634  -8.612   0.108   8.689  63.430
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)      21.248903   4.024775   5.280 1.47e-07 ***
## TEAM_BATTING_H    0.044708   0.002855  15.659  < 2e-16 ***
## TEAM_BATTING_3B   0.055455   0.018037   3.074  0.00214 **
## TEAM_BASERUN_SB   0.045316   0.004793   9.455  < 2e-16 ***
## TEAM_PITCHING_HR  0.041460   0.007415   5.592 2.64e-08 ***
## TEAM_FIELDING_E  -0.035092   0.002014 -17.426  < 2e-16 ***
## TEAM_FIELDING_DP -0.079196   0.014685  -5.393 7.97e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 12.91 on 1594 degrees of freedom
## Multiple R-squared:  0.3161, Adjusted R-squared:  0.3135
## F-statistic: 122.8 on 6 and 1594 DF,  p-value: < 2.2e-16
```