Data 621 Assignment 1

Group2 - Gabriella Martinez, Maliat Islam, Ken Popkin, George Cruz Deschamps,

Matthew Lucich, and Karim Hammoud

2021-09-22

Table of Contents

DATA 621 – Business Analytics and Data Mining
Homework #1 Assignment Requirements

Overview

In this homework assignment, you will explore, analyze and model a data set containing approximately 2200 records. Each record represents a professional baseball team from the years 1871 to 2006 inclusive. Each record has the performance of the team for the given year, with all of the statistics adjusted to match the performance of a 162 game season.

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

VARIABLE NAME	DEFINITION	THEORETICAL EFFECT
INDEX	Identification Variable (do not use)	None
TARGET_WINS	Number of wins	
TEAM_BATTING_H	Base Hits by batters (1B,2B,3B,HR)	Positive Impact on Wins
TEAM_BATTING_2B	Doubles by batters (2B)	Positive Impact on Wins
TEAM_BATTING_3B	Triples by batters (3B)	Positive Impact on Wins
TEAM_BATTING_HR	Homeruns by batters (4B)	Positive Impact on Wins
TEAM_BATTING_BB	Walks by batters	Positive Impact on Wins
TEAM_BATTING_HBP	Batters hit by pitch (get a free base)	Positive Impact on Wins
TEAM_BATTING_SO	Strikeouts by batters	Negative Impact on Wins
TEAM_BASERUN_SB	Stolen bases	Positive Impact on Wins
TEAM_BASERUN_CS	Caught stealing	Negative Impact on Wins
TEAM_FIELDING_E	Errors	Negative Impact on Wins
TEAM_FIELDING_DP	Double Plays	Positive Impact on Wins
TEAM_PITCHING_BB	Walks allowed	Negative Impact on Wins
TEAM_PITCHING_H	Hits allowed	Negative Impact on Wins
TEAM_PITCHING_HR	Homeruns allowed	Negative Impact on Wins
TEAM_PITCHING_SO	Strikeouts by pitchers	Positive Impact on Wins

Deliverables:

- A write-up submitted in PDF format. Your write-up should have four sections. Each one is described below. You may assume you are addressing me as a fellow data scientist, so do not need to shy away from technical details.

- Assigned predictions (the number of wins for the team) for the evaluation data set.

- Include your R statistical programming code in an Appendix.

In this data set we are trying to identify good and bad teams in major league baseball team's season. We are assuming some of the predictors will be higher for good teams. We will try to predict how many times a team will win in this season.

DATA EXPLORATION:

We can observe the response variable (TARGET_WINS) looks to be normally distributed. This supports the working theory that there are good teams and bad teams. There are also a lot of average teams.

There are also quite a few variables with missing values. and,Some variables are right skewed (TEAM_BASERUN_CS, TEAM_BASERUN_SB, etc.). This might support the good team theory. It may also introduce non-normally distributed residuals in the model. We shall see.

Load the Data

Summary of the train data

```
##      INDEX         TARGET_WINS     TEAM_BATTING_H TEAM_BATTING_2B
##  Min.   :   1.0   Min.   :  0.00   Min.   : 891   Min.   : 69.0  
##  1st Qu.: 630.8   1st Qu.: 71.00   1st Qu.:1383   1st Qu.:208.0  
##  Median :1270.5   Median : 82.00   Median :1454   Median :238.0  
##  Mean   :1268.5   Mean   : 80.79   Mean   :1469   Mean   :241.2  
##  3rd Qu.:1915.5   3rd Qu.: 92.00   3rd Qu.:1537   3rd Qu.:273.0  
##  Max.   :2535.0   Max.   :146.00   Max.   :2554   Max.   :458.0  
##                                                                  
##  TEAM_BATTING_3B  TEAM_BATTING_HR  TEAM_BATTING_BB TEAM_BATTING_SO 
##  Min.   :  0.00   Min.   :  0.00   Min.   :  0.0   Min.   :   0.0  
##  1st Qu.: 34.00   1st Qu.: 42.00   1st Qu.:451.0   1st Qu.: 548.0  
##  Median : 47.00   Median :102.00   Median :512.0   Median : 750.0  
##  Mean   : 55.25   Mean   : 99.61   Mean   :501.6   Mean   : 735.6  
##  3rd Qu.: 72.00   3rd Qu.:147.00   3rd Qu.:580.0   3rd Qu.: 930.0  
##  Max.   :223.00   Max.   :264.00   Max.   :878.0   Max.   :1399.0  
##                                                    NA's   :102     
##  TEAM_BASERUN_SB TEAM_BASERUN_CS TEAM_BATTING_HBP TEAM_PITCHING_H
##  Min.   :  0.0   Min.   :  0.0   Min.   :29.00    Min.   : 1137  
##  1st Qu.: 66.0   1st Qu.: 38.0   1st Qu.:50.50    1st Qu.: 1419  
##  Median :101.0   Median : 49.0   Median :58.00    Median : 1518  
##  Mean   :124.8   Mean   : 52.8   Mean   :59.36    Mean   : 1779  
##  3rd Qu.:156.0   3rd Qu.: 62.0   3rd Qu.:67.00    3rd Qu.: 1682  
##  Max.   :697.0   Max.   :201.0   Max.   :95.00    Max.   :30132  
##  NA's   :131     NA's   :772     NA's   :2085                    
##  TEAM_PITCHING_HR TEAM_PITCHING_BB TEAM_PITCHING_SO  TEAM_FIELDING_E 
##  Min.   :  0.0    Min.   :   0.0   Min.   :    0.0   Min.   :  65.0  
##  1st Qu.: 50.0    1st Qu.: 476.0   1st Qu.:  615.0   1st Qu.: 127.0  
##  Median :107.0    Median : 536.5   Median :  813.5   Median : 159.0  
##  Mean   :105.7    Mean   : 553.0   Mean   :  817.7   Mean   : 246.5  
##  3rd Qu.:150.0    3rd Qu.: 611.0   3rd Qu.:  968.0   3rd Qu.: 249.2  
##  Max.   :343.0    Max.   :3645.0   Max.   :19278.0   Max.   :1898.0  
##                                    NA's   :102                       
##  TEAM_FIELDING_DP
##  Min.   : 52.0   
##  1st Qu.:131.0   
##  Median :149.0   
##  Mean   :146.4   
##  3rd Qu.:164.0   
##  Max.   :228.0   
##  NA's   :286
```

Summary of the test data

```
##      INDEX      TEAM_BATTING_H TEAM_BATTING_2B TEAM_BATTING_3B
##  Min.   :   9   Min.   : 819   Min.   : 44.0   Min.   : 14.00
##  1st Qu.: 708   1st Qu.:1387   1st Qu.:210.0   1st Qu.: 35.00
##  Median :1249   Median :1455   Median :239.0   Median : 52.00
##  Mean   :1264   Mean   :1469   Mean   :241.3   Mean   : 55.91
##  3rd Qu.:1832   3rd Qu.:1548   3rd Qu.:278.5   3rd Qu.: 72.00
##  Max.   :2525   Max.   :2170   Max.   :376.0   Max.   :155.00
##
##  TEAM_BATTING_HR  TEAM_BATTING_BB TEAM_BATTING_SO  TEAM_BASERUN_SB
##  Min.   :  0.00   Min.   : 15.0   Min.   :   0.0   Min.   :  0.0
##  1st Qu.: 44.50   1st Qu.:436.5   1st Qu.: 545.0   1st Qu.: 59.0
##  Median :101.00   Median :509.0   Median : 686.0   Median : 92.0
##  Mean   : 95.63   Mean   :499.0   Mean   : 709.3   Mean   :123.7
##  3rd Qu.:135.50   3rd Qu.:565.5   3rd Qu.: 912.0   3rd Qu.:151.8
##  Max.   :242.00   Max.   :792.0   Max.   :1268.0   Max.   :580.0
##                                   NA's   :18       NA's   :13
##  TEAM_BASERUN_CS  TEAM_BATTING_HBP TEAM_PITCHING_H TEAM_PITCHING_HR
##  Min.   :  0.00   Min.   :42.00    Min.   : 1155   Min.   :  0.0
##  1st Qu.: 38.00   1st Qu.:53.50    1st Qu.: 1426   1st Qu.: 52.0
##  Median : 49.50   Median :62.00    Median : 1515   Median :104.0
##  Mean   : 52.32   Mean   :62.37    Mean   : 1813   Mean   :102.1
##  3rd Qu.: 63.00   3rd Qu.:67.50    3rd Qu.: 1681   3rd Qu.:142.5
##  Max.   :154.00   Max.   :96.00    Max.   :22768   Max.   :336.0
##  NA's   :87       NA's   :240
##  TEAM_PITCHING_BB TEAM_PITCHING_SO TEAM_FIELDING_E  TEAM_FIELDING_DP
##  Min.   : 136.0   Min.   :   0.0   Min.   :  73.0   Min.   : 69.0
##  1st Qu.: 471.0   1st Qu.: 613.0   1st Qu.: 131.0   1st Qu.:131.0
##  Median : 526.0   Median : 745.0   Median : 163.0   Median :148.0
##  Mean   : 552.4   Mean   : 799.7   Mean   : 249.7   Mean   :146.1
##  3rd Qu.: 606.5   3rd Qu.: 938.0   3rd Qu.: 252.0   3rd Qu.:164.0
##  Max.   :2008.0   Max.   :9963.0   Max.   :1568.0   Max.   :204.0
##                   NA's   :18                        NA's   :31
```

Glimpse of the train data

```
## Rows: 2,276
## Columns: 17
## $ INDEX           <int> 1, 2, 3, 4, 5, 6, 7, 8, 11, 12, 13, 15, 16, 17, 1
8, 1…
## $ TARGET_WINS     <int> 39, 70, 86, 70, 82, 75, 80, 85, 86, 76, 78, 68, 7
2, 7…
## $ TEAM_BATTING_H  <int> 1445, 1339, 1377, 1387, 1297, 1279, 1244, 1273, 1
391,…
## $ TEAM_BATTING_2B <int> 194, 219, 232, 209, 186, 200, 179, 171, 197, 213,
179…
## $ TEAM_BATTING_3B <int> 39, 22, 35, 38, 27, 36, 54, 37, 40, 18, 27, 31, 4
1, 2…
## $ TEAM_BATTING_HR <int> 13, 190, 137, 96, 102, 92, 122, 115, 114, 96, 82,
95,…
## $ TEAM_BATTING_BB <int> 143, 685, 602, 451, 472, 443, 525, 456, 447, 441,
374…
## $ TEAM_BATTING_SO <int> 842, 1075, 917, 922, 920, 973, 1062, 1027, 922, 8
27, …
## $ TEAM_BASERUN_SB <int> NA, 37, 46, 43, 49, 107, 80, 40, 69, 72, 60, 119,
221…
## $ TEAM_BASERUN_CS <int> NA, 28, 27, 30, 39, 59, 54, 36, 27, 34, 39, 79, 1
09, …
## $ TEAM_BATTING_HBP <int> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, N
A, N…
## $ TEAM_PITCHING_H <int> 9364, 1347, 1377, 1396, 1297, 1279, 1244, 1281, 1
391,…
## $ TEAM_PITCHING_HR <int> 84, 191, 137, 97, 102, 92, 122, 116, 114, 96, 86,
95,…
## $ TEAM_PITCHING_BB <int> 927, 689, 602, 454, 472, 443, 525, 459, 447, 441,
391…
## $ TEAM_PITCHING_SO <int> 5456, 1082, 917, 928, 920, 973, 1062, 1033, 922,
827,…
## $ TEAM_FIELDING_E <int> 1011, 193, 175, 164, 138, 123, 136, 112, 127, 131
, 11…
## $ TEAM_FIELDING_DP <int> NA, 155, 153, 156, 168, 149, 186, 136, 169, 159,
141,…
```

Glimpse of the test data

```
## Rows: 259
## Columns: 16
## $ INDEX           <int> 9, 10, 14, 47, 60, 63, 74, 83, 98, 120, 123, 135,
138…
## $ TEAM_BATTING_H  <int> 1209, 1221, 1395, 1539, 1445, 1431, 1430, 1385, 1
259,…
## $ TEAM_BATTING_2B <int> 170, 151, 183, 309, 203, 236, 219, 158, 177, 212,
243…
## $ TEAM_BATTING_3B <int> 33, 29, 29, 29, 68, 53, 55, 42, 78, 42, 40, 55, 5
7, 2…
## $ TEAM_BATTING_HR <int> 83, 88, 93, 159, 5, 10, 37, 33, 23, 58, 50, 164,
186,…
## $ TEAM_BATTING_BB <int> 447, 516, 509, 486, 95, 215, 568, 356, 466, 452,
495,…
## $ TEAM_BATTING_SO <int> 1080, 929, 816, 914, 416, 377, 527, 609, 689, 584
, 64…
## $ TEAM_BASERUN_SB <int> 62, 54, 59, 148, NA, NA, 365, 185, 150, 52, 64, 4
8, 3…
## $ TEAM_BASERUN_CS <int> 50, 39, 47, 57, NA, NA, NA, NA, NA, NA, NA, 28, 2
1, 8…
## $ TEAM_BATTING_HBP <int> NA, NA, NA, 42, NA, NA, NA, NA, NA, NA, NA, NA, N
A, N…
## $ TEAM_PITCHING_H <int> 1209, 1221, 1395, 1539, 3902, 2793, 1544, 1626, 1
342,…
## $ TEAM_PITCHING_HR <int> 83, 88, 93, 159, 14, 20, 40, 39, 25, 62, 53, 173,
196…
## $ TEAM_PITCHING_BB <int> 447, 516, 509, 486, 257, 420, 613, 418, 497, 482,
521…
## $ TEAM_PITCHING_SO <int> 1080, 929, 816, 914, 1123, 736, 569, 715, 734, 62
2, 6…
## $ TEAM_FIELDING_E <int> 140, 135, 156, 124, 616, 572, 490, 328, 226, 184,
200…
## $ TEAM_FIELDING_DP <int> 156, 164, 153, 154, 130, 105, NA, 104, 132, 145,
183,…
```

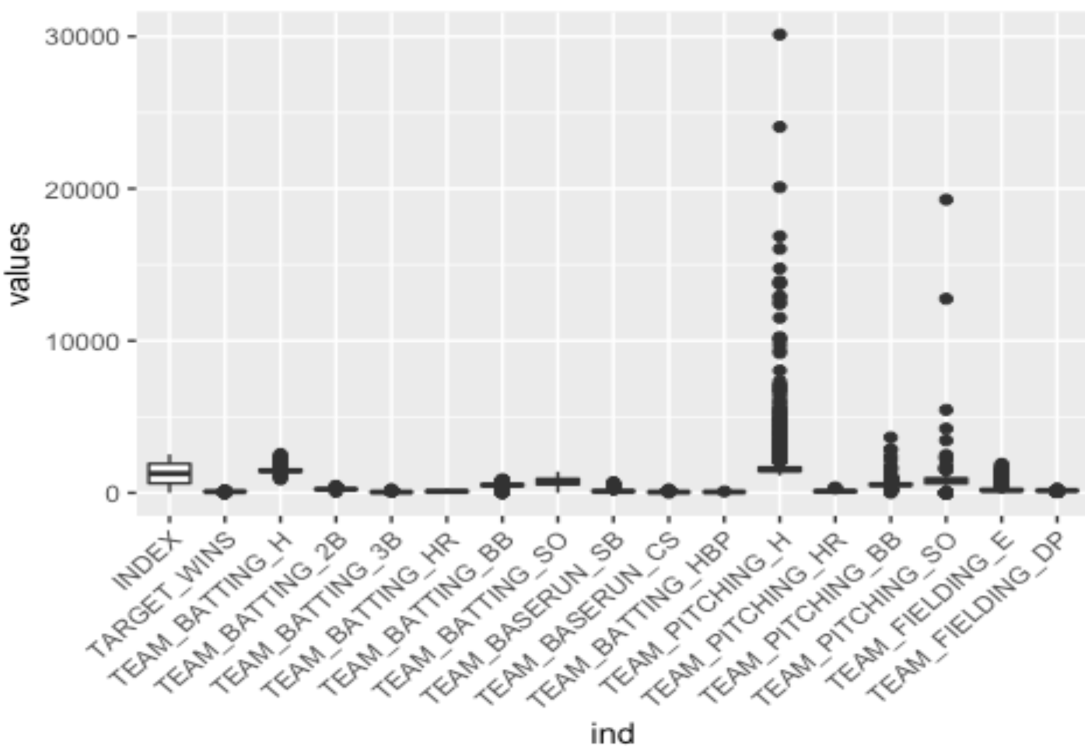
Find SD for all of the train data

```
##            INDEX   TEAM_BATTING_H  TEAM_BATTING_2B  TEAM_BATTING_3B
##        693.28867        150.65523         49.51612         27.14410
##  TEAM_BATTING_HR  TEAM_BATTING_BB  TEAM_BATTING_SO  TEAM_BASERUN_SB
##         56.33221        120.59215        243.11114         93.38796
##  TEAM_BASERUN_CS TEAM_BATTING_HBP  TEAM_PITCHING_H TEAM_PITCHING_HR
##         23.10457         12.70700       1662.91308         57.65490
## TEAM_PITCHING_BB TEAM_PITCHING_SO  TEAM_FIELDING_E TEAM_FIELDING_DP
##        172.95006        634.30585        230.90260         25.88387
```

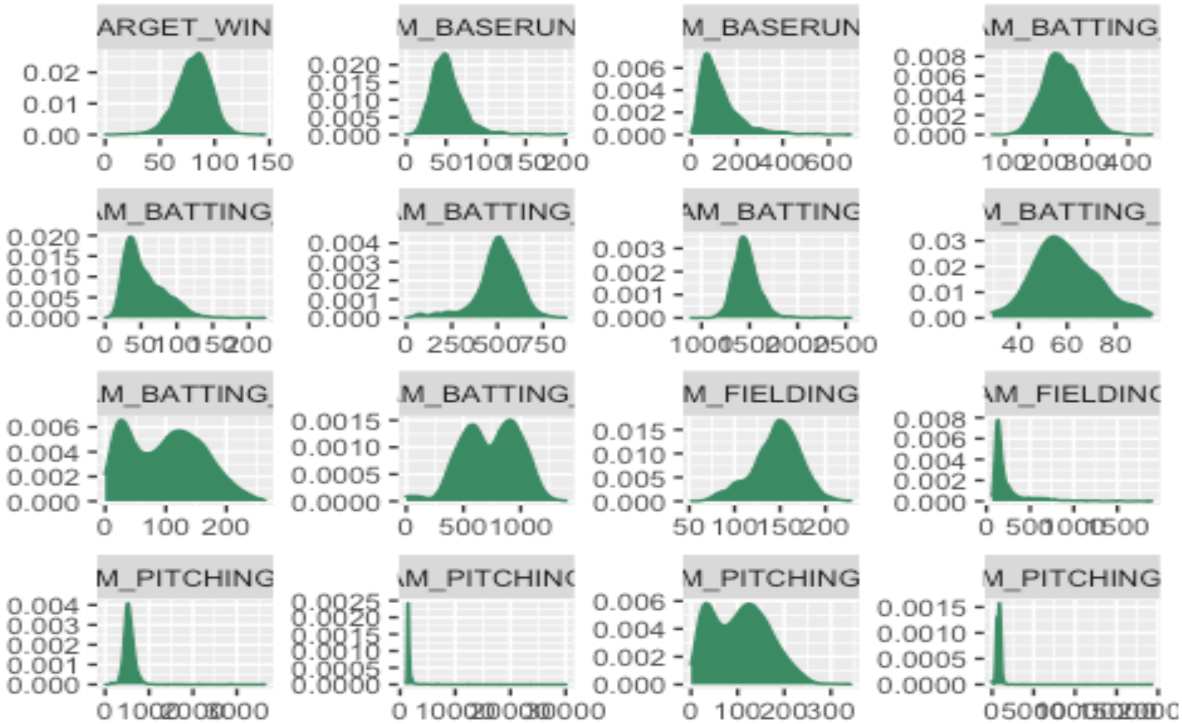
Find SD for all of the test data

```
##            INDEX      TARGET_WINS   TEAM_BATTING_H  TEAM_BATTING_2B
##        736.34904         15.75215        144.59120         46.80141
##  TEAM_BATTING_3B  TEAM_BATTING_HR  TEAM_BATTING_BB  TEAM_BATTING_SO
##         27.93856         60.54687        122.67086        248.52642
##  TEAM_BASERUN_SB  TEAM_BASERUN_CS TEAM_BATTING_HBP  TEAM_PITCHING_H
##         87.79117         22.95634         12.96712       1406.84293
## TEAM_PITCHING_HR TEAM_PITCHING_BB TEAM_PITCHING_SO  TEAM_FIELDING_E
##         61.29875        166.35736        553.08503        227.77097
## TEAM_FIELDING_DP
##         26.22639
```

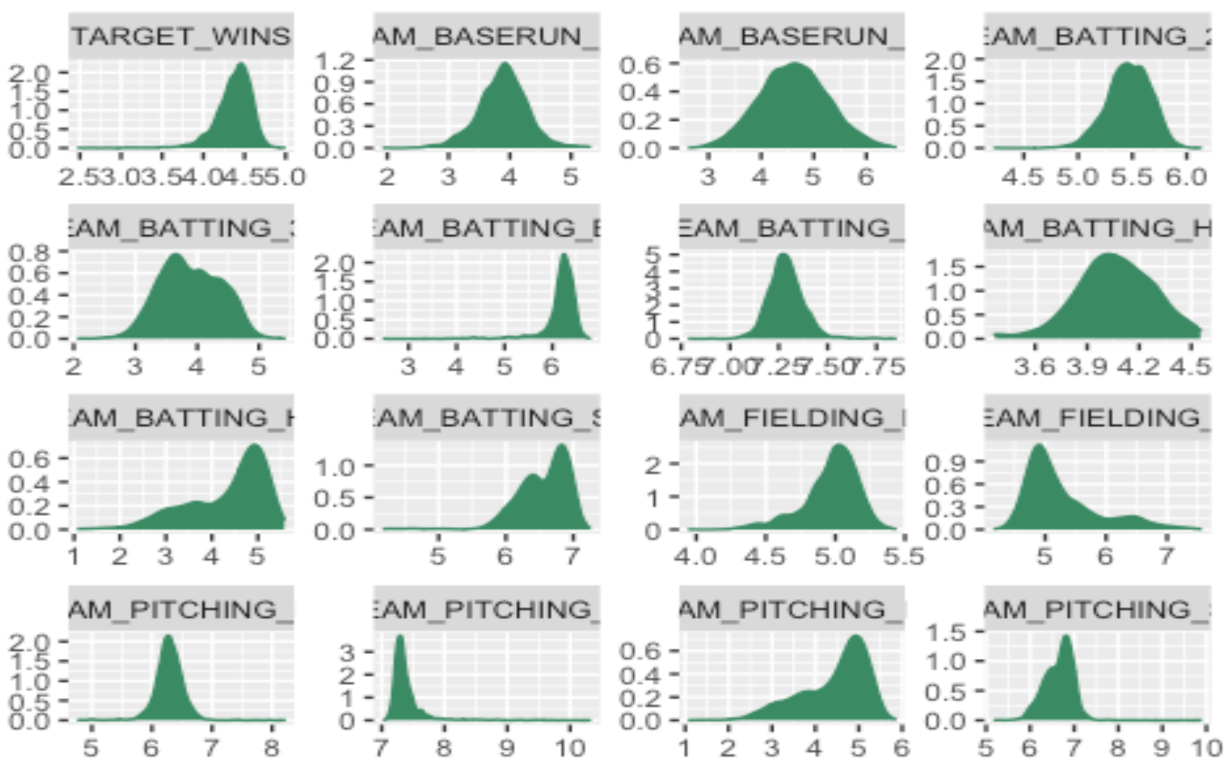
Box plot the train data



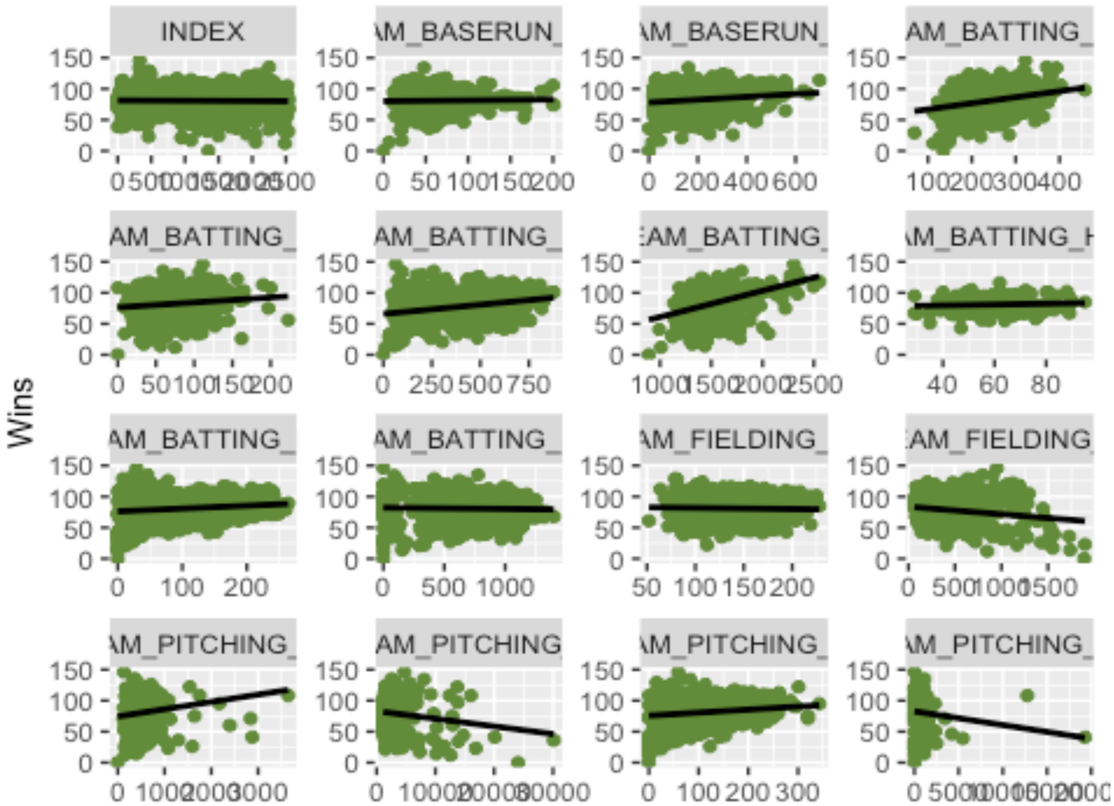
Variable Distributions

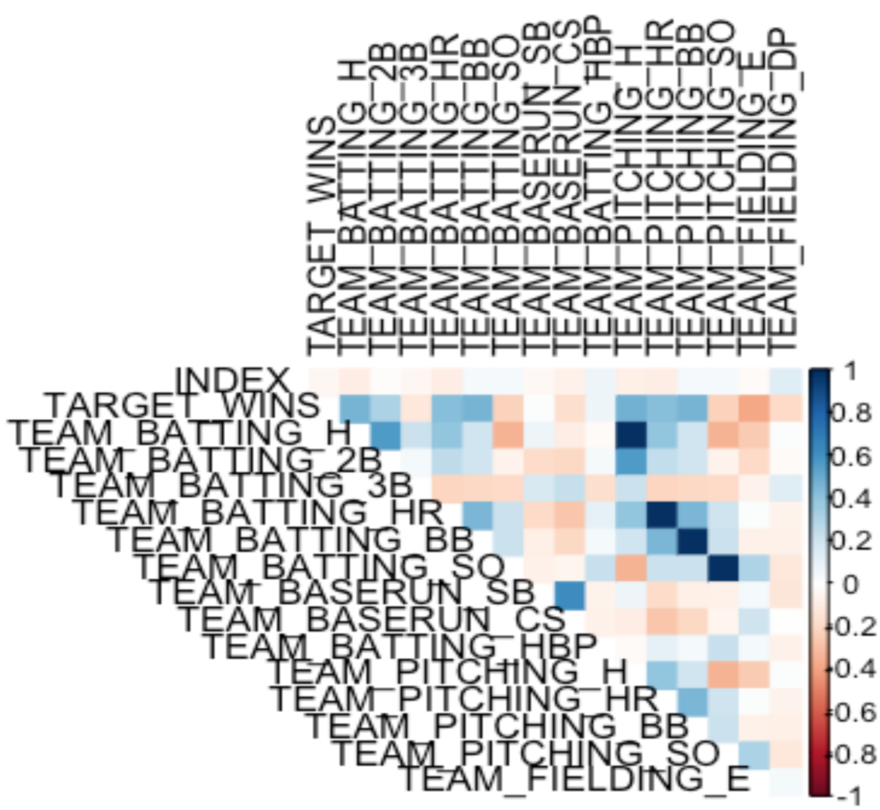


Log Variable Distributions



Correlations with Response Variable



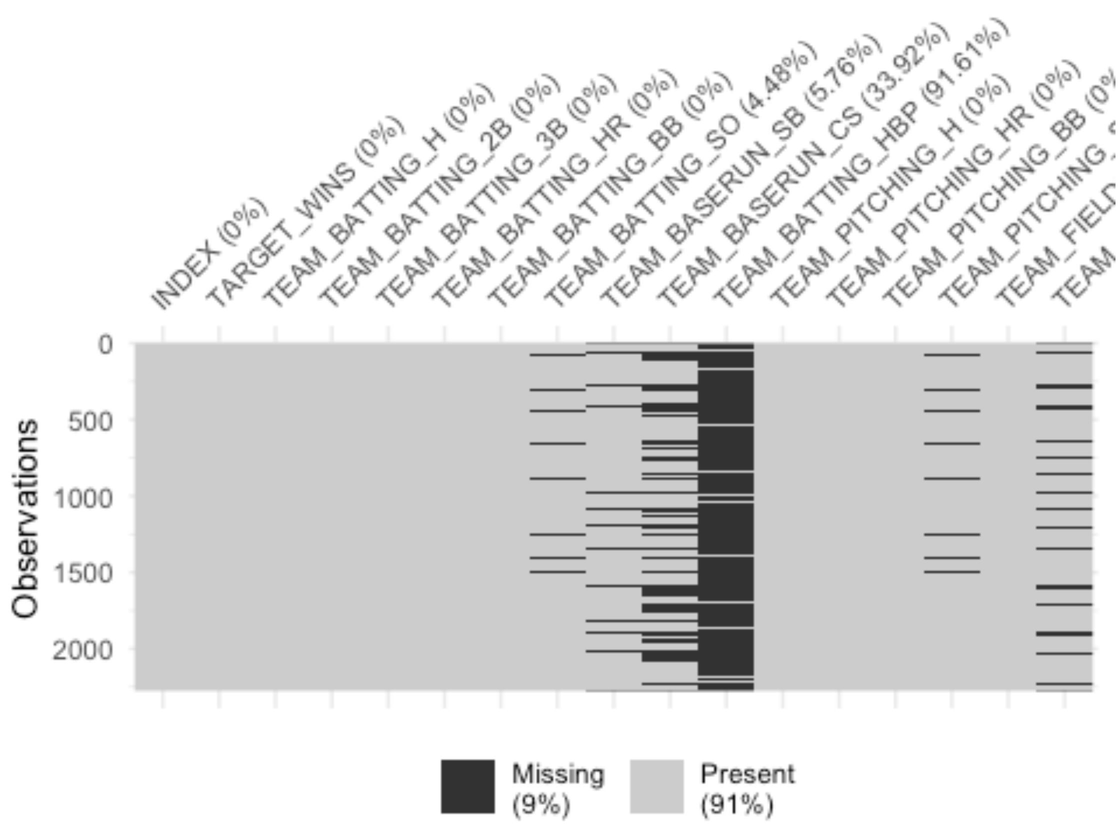


DATA PREPARATION

NA counts for the train data set [1]

```
##            INDEX      TARGET_WINS   TEAM_BATTING_H  TEAM_BATTING_2B
##                0                0                0                0
##  TEAM_BATTING_3B  TEAM_BATTING_HR  TEAM_BATTING_BB  TEAM_BATTING_SO
##                0                0                0              102
##  TEAM_BASERUN_SB  TEAM_BASERUN_CS TEAM_BATTING_HBP  TEAM_PITCHING_H
##              131              772             2085                0
## TEAM_PITCHING_HR TEAM_PITCHING_BB TEAM_PITCHING_SO  TEAM_FIELDING_E
##                0                0              102                0
## TEAM_FIELDING_DP
##              286
```

visulaization and percentage of NA values [2]

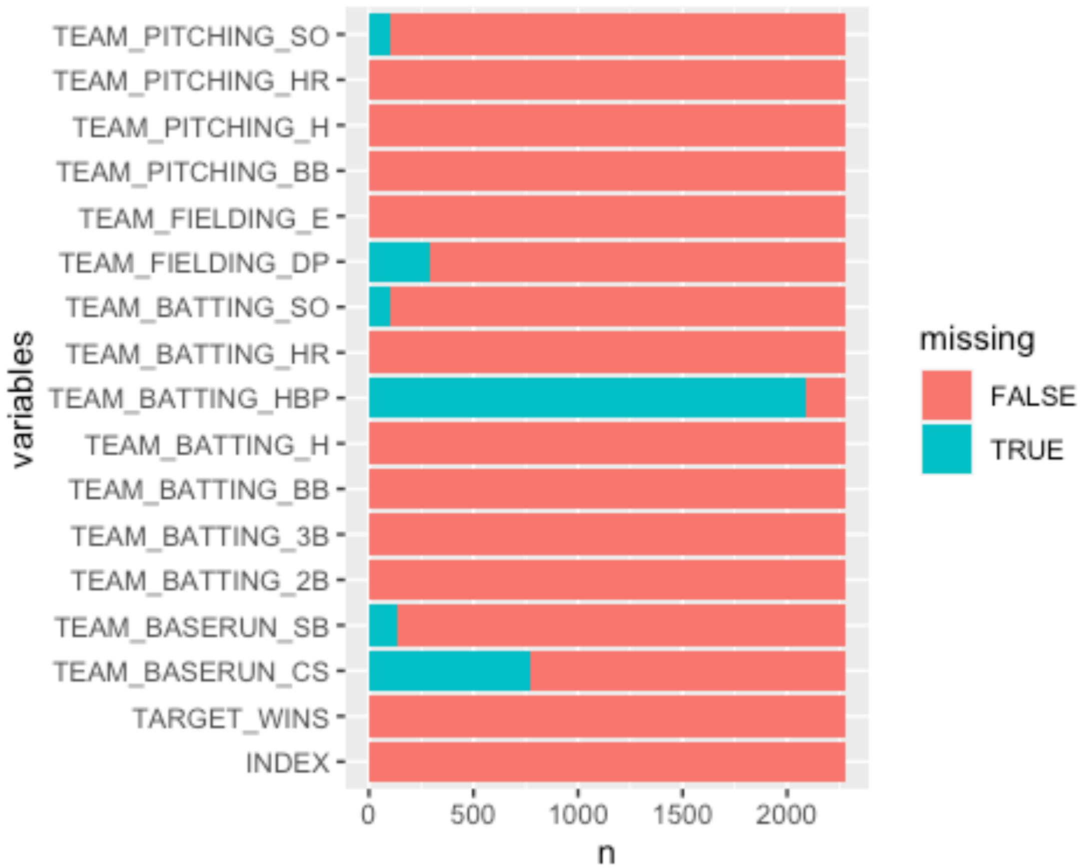


---

[1] https://statisticsglobe.com/count-number-of-na-values-in-vector-and-column-in-r

[2] https://cran.r-project.org/web/packages/naniar/vignettes/naniar-visualisation.html

alternative NA values visualization [3]



Since 92% of the data for the TEAM_BATTING_HBP is missing, the variable has been removed from both test and train data. TEAM_BASERUN_CS is a runner up with the next highest amount of NA at 34%.

[3] https://datavizpyr.com/visualizing-missing-data-with-barplot-in-r/

BUILD MODELS

Model #1

Two predictors: Base hits by batters and Hits allowed

Using a manual review, below are the features selected for the first model and the supporting reason/s.

TEAM_BATTING_H = Base hits by batters: it's impossible to win in baseball without getting to the bases and hitting the ball is the primary means to accomplish this.

TEAM_PITCHING_H = Hits allowed: winning without a good defense is difficult and in baseball preventing the other team from getting hits is a good defense strategy.

Only two features are selected for the first model - start small and build up seems like a good approach.

Create the Regression Model

```
##
## Call:
## lm(formula = TARGET_WINS ~ TEAM_BATTING_H + TEAM_PITCHING_H,
##     data = train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -57.444  -9.113   0.613   9.677  77.744
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)     15.6425241  3.5652886   4.387 1.22e-05 ***
## TEAM_BATTING_H   0.0475405  0.0024789  19.178  < 2e-16 ***
## TEAM_PITCHING_H -0.0026104  0.0002462 -10.603  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 14.16 on 1704 degrees of freedom
## Multiple R-squared:  0.189,  Adjusted R-squared:  0.1881
## F-statistic: 198.6 on 2 and 1704 DF,  p-value: < 2.2e-16
```

The p values are 0, which per the criteria of "keep a feature if the p-value is <0.05" recommends that we keep both these features. But, the adjusted R-squared is TERRIBLE at around 21%. Even though the R-squared is poor it's simple to run this model with the test data, so we'll do that next.

Evaluate the second model results using RMSE

```
## [1] 13.6336
```

Model #2

Four predictors: Base hits by batters, Hits allowed, Errors, and Walks allowed

Using a manual review, below are the features selected for the second model and the supporting reason/s.

We'll keep the features from the first model (due to low p-values) and add two more features… TEAM_FIELDING_E = Errors: errors are costly in terms of immediate impact, but could also impact the team in other ways (i.e. a high occurrence could impact team comraderie and confidence in each other)

TEAM_PITCHING_BB = Walks allowed: putting players on base for "free" is more opportunity for points

Create the Regression Model

```
##
## Call:
## lm(formula = TARGET_WINS ~ TEAM_BATTING_H + TEAM_PITCHING_H +
##     TEAM_FIELDING_E + TEAM_PITCHING_BB, data = train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -51.927  -9.234   0.174   9.477  47.151
##
## Coefficients:
##                    Estimate Std. Error t value Pr(>|t|)
## (Intercept)       7.2608113  3.6224175   2.004   0.0452 *
## TEAM_BATTING_H    0.0495348  0.0024231  20.443  < 2e-16 ***
## TEAM_PITCHING_H  -0.0018244  0.0003488  -5.231 1.89e-07 ***
## TEAM_FIELDING_E  -0.0129111  0.0020960  -6.160 9.07e-10 ***
## TEAM_PITCHING_BB  0.0130541  0.0023301   5.602 2.46e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 13.73 on 1702 degrees of freedom
## Multiple R-squared:  0.2382, Adjusted R-squared:  0.2364
## F-statistic:   133 on 4 and 1702 DF,  p-value: < 2.2e-16
```

Evaluate the second model results using RMSE

```
## [1] 13.30535
```

The increase from two features in the first model to four features in the second model did not yield a noticeable improvement. The Adjusted R2 on the training data improved slightly, but the RMSE for all practical purposes stayed the same at around 13; which is a poor RMSE implying that both models have poor predictive capability.

Model #3

BSR Model (SaberMetrics) (data imputation)

Base runs (BsR) is a baseball statistic invented by sabermetrician David Smyth to estimate the number of runs a team "should have" scored given their component offensive statistics, as well as the number of runs a hitter or pitcher creates or allows. It measures essentially the same thing as Bill James runs created, but as sabermetrician Tom M. Tango points out, base runs models the reality of the run-scoring process "significantly better than any other run estimator".

Cleaning Data

```
##
##  iter imp variable
##   1   1  TEAM_BATTING_SO  TEAM_BASERUN_SB  TEAM_BASERUN_CS  TEAM_PITCHING_
SO  TEAM_FIELDING_DP
##   2   1  TEAM_BATTING_SO  TEAM_BASERUN_SB  TEAM_BASERUN_CS  TEAM_PITCHING_
SO  TEAM_FIELDING_DP
##   3   1  TEAM_BATTING_SO  TEAM_BASERUN_SB  TEAM_BASERUN_CS  TEAM_PITCHING_
SO  TEAM_FIELDING_DP
##   4   1  TEAM_BATTING_SO  TEAM_BASERUN_SB  TEAM_BASERUN_CS  TEAM_PITCHING_
SO  TEAM_FIELDING_DP
##   5   1  TEAM_BATTING_SO  TEAM_BASERUN_SB  TEAM_BASERUN_CS  TEAM_PITCHING_
SO  TEAM_FIELDING_DP
```

The simplest, uses only the most common batting statistics[2]

$A = H + BB - HR$ $B = (1.4 * TB - .6 * H - 3 * HR + .1 * BB) * 1.02$ $C = AB - H$ $D = HR$

$$BsR = \frac{(A * B)}{(B + C)} + D$$

Create the Regression Model

BSR

```
##
## Call:
## lm(formula = TARGET_WINS ~ BSR + TEAM_PITCHING_SO + TEAM_FIELDING_E +
##     TEAM_FIELDING_DP, data = rmdata3)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -63.776  -8.418   0.410   8.537  50.024
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)      45.237755   3.161743  14.308  < 2e-16 ***
## BSR               0.049301   0.002015  24.469  < 2e-16 ***
## TEAM_PITCHING_SO  0.009641   0.001308   7.369 2.67e-13 ***
## TEAM_FIELDING_E  -0.039350   0.001995 -19.728  < 2e-16 ***
```

```
## TEAM_FIELDING_DP -0.169143   0.012539 -13.489  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 13.18 on 1702 degrees of freedom
## Multiple R-squared:  0.3157, Adjusted R-squared:  0.3141
## F-statistic: 196.3 on 4 and 1702 DF,  p-value: < 2.2e-16
```

Evaluate the model results using RMSE

```
## [1] 14.25345
```

Model #4

(Modified) Backward Elimination Model (omitting NAs)

Due to previously learning how to perform Backward Elimination and it being possible to perform manually, we decided to include a model that resulted from the procedure. The process was performed with imputed data (via MICE) as well as data with NAs removed. The latter showed stronger results, therefore the final model was fitted with the NA omitted data.

According to Faraway, Backward Elimination is when you start with all predictors in the model, then remove the predictor with the highest p-value as long as it is above your p-value threshold (e.g. 0.05). Then refit the model and continue the process until only predictors with p-values below your threshold remain.

Additionally, we took steps to remove variables with non-intuitive coefficients. For instance, TEAM_FIELDING_DP and TEAM_PITCHING_SO were unexpectedly showing negative effects on wins. While there could be potential intervening variables giving these variables true predictive power, we opted to remove the variables from the model due to the possibility they were significant by chance and due to our bias towards parsimony. Further, RMSE did not drastically worsen when removed.

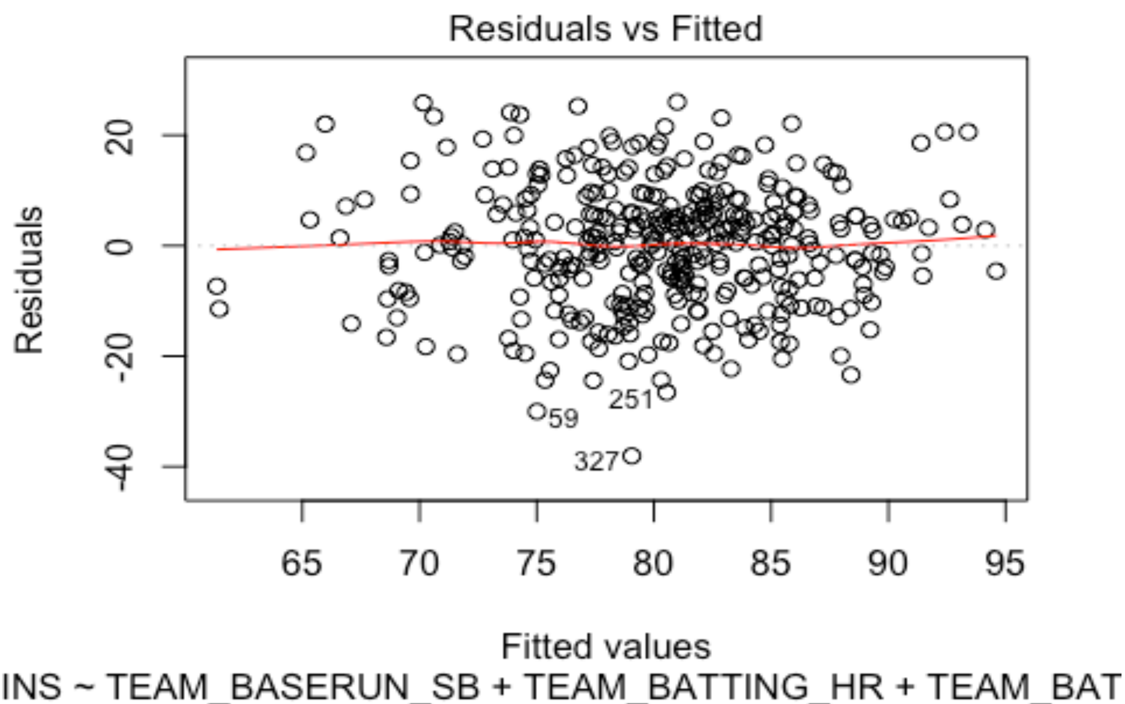
```
##
## Call:
## lm(formula = TARGET_WINS ~ TEAM_BASERUN_SB + TEAM_BATTING_HR +
##     TEAM_BATTING_BB + TEAM_FIELDING_E, data = test_no_na)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -38.055  -7.316   0.722   6.485  26.008
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)     56.249572   6.027751   9.332  < 2e-16 ***
## TEAM_BASERUN_SB  0.047422   0.013540   3.502 0.000520 ***
## TEAM_BATTING_HR  0.055258   0.016007   3.452 0.000622 ***
## TEAM_BATTING_BB  0.035161   0.007576   4.641 4.87e-06 ***
## TEAM_FIELDING_E -0.045877   0.018742  -2.448 0.014853 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 11.16 on 359 degrees of freedom
## Multiple R-squared:  0.2096, Adjusted R-squared:  0.2008
## F-statistic:  23.8 on 4 and 359 DF,  p-value: < 2.2e-16

## [1] 11.081
```

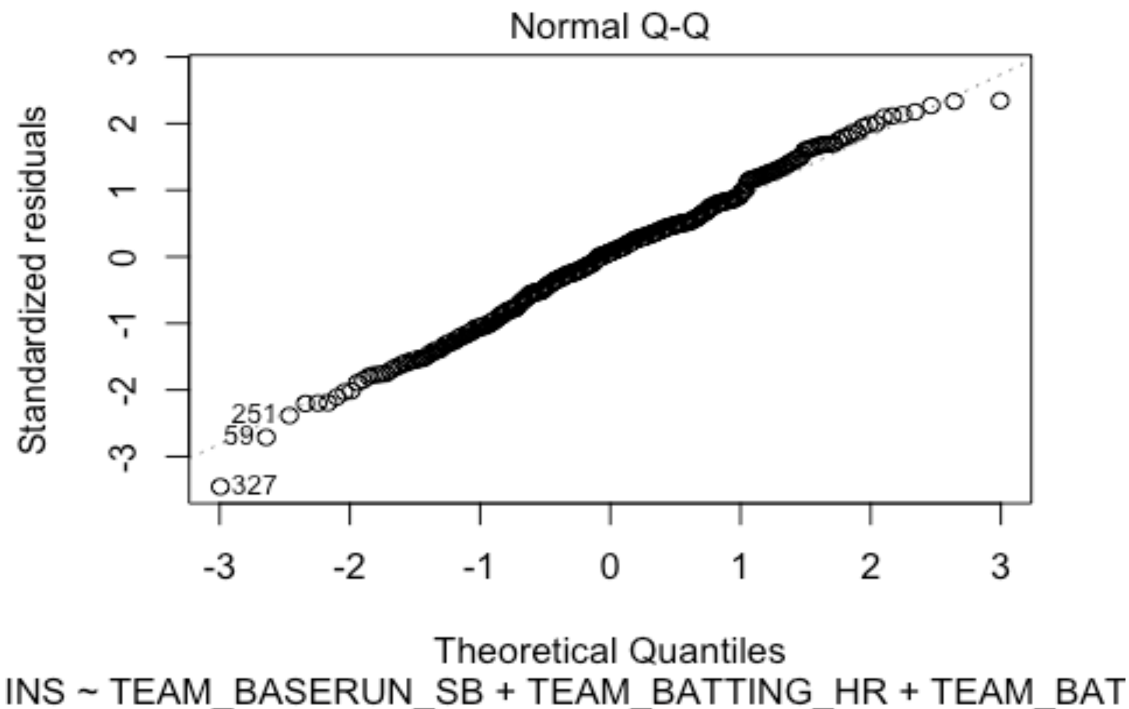
SELECT MODELS

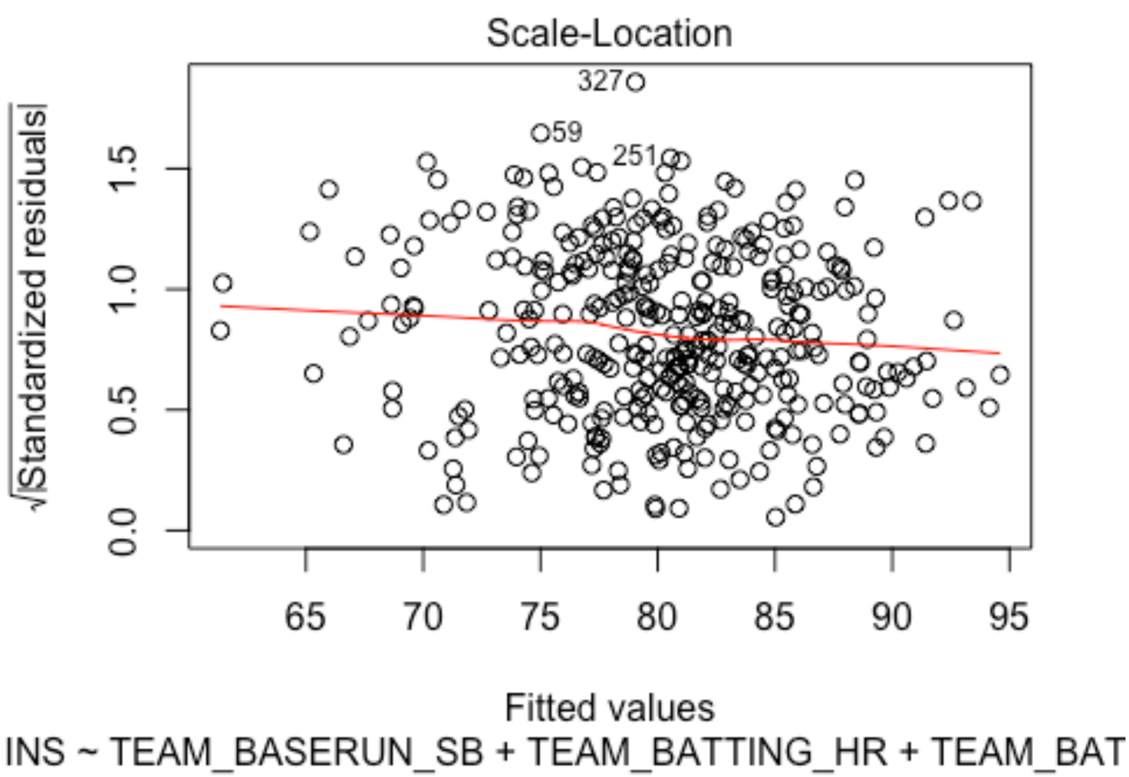
Verifying OLS Regression Assumptions

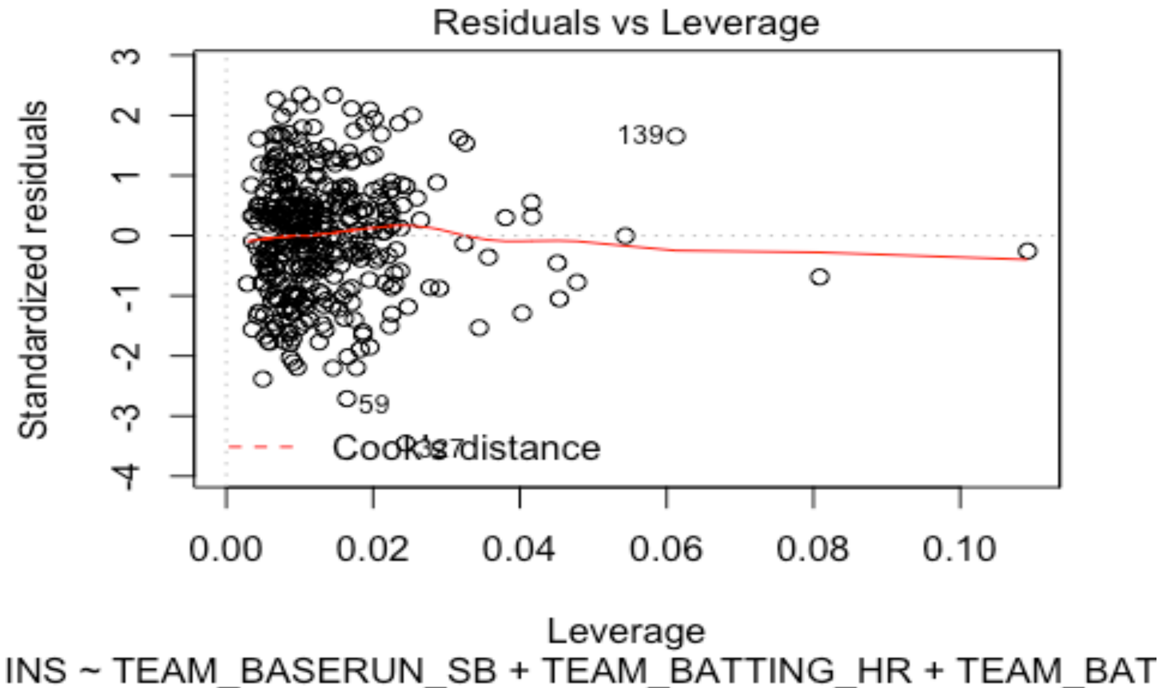
Assumption: Mean of residuals is zero

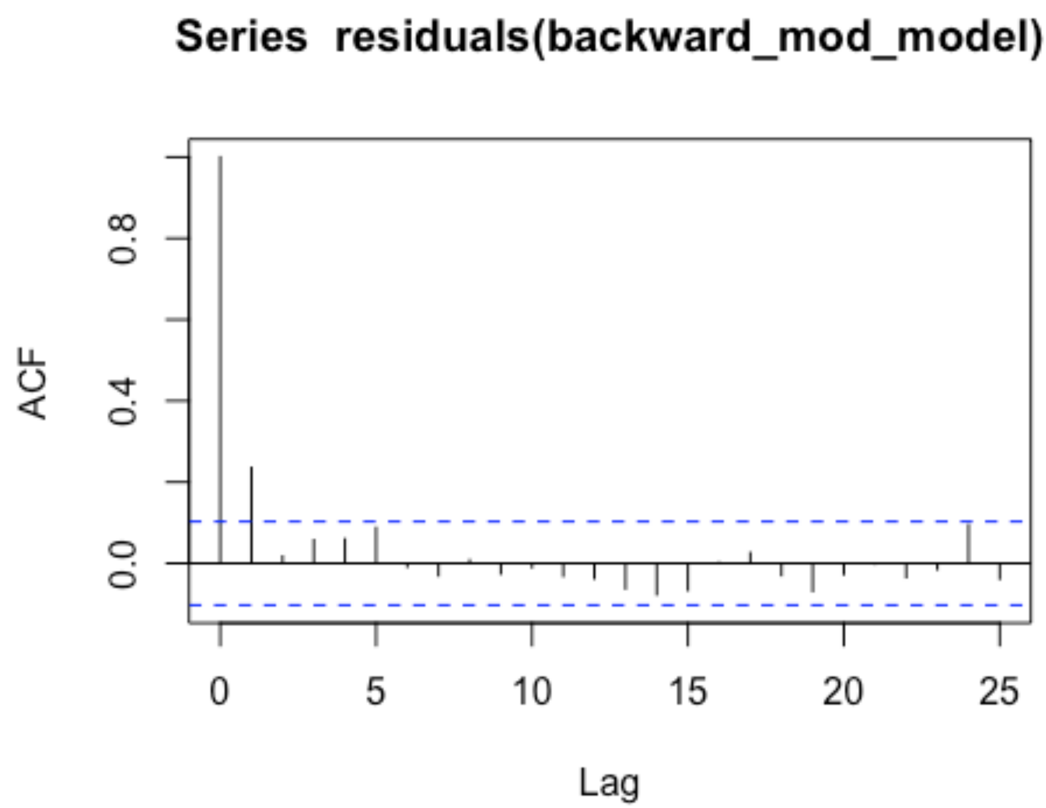
```
## [1] 2.396206e-17
```




Normal Q-Q
Standardized residuals
Theoretical Quantiles
INS ~ TEAM_BASERUN_SB + TEAM_BATTING_HR + TEAM_BAT
251
59
327


Scale-Location
√|Standardized residuals|
Fitted values
INS ~ TEAM_BASERUN_SB + TEAM_BATTING_HR + TEAM_BAT
327
59
251


Residuals vs Leverage
Standardized residuals
3
2
1
0
-1
-2
-3
-4
139
59
Cook's distance
0.00
0.02
0.04
0.06
0.08
0.10
Leverage
INS ~ TEAM_BASERUN_SB + TEAM_BATTING_HR + TEAM_BAT


Series residuals(backward_mod_model)
ACF
0.8
0.4
0.0
0
5
10
15
20
25
Lag

Model Selection

First, before fully evaluating models we validated that all individual predictors had p-values below 0.05, the cutoff for a 95% confidence level. Additionally, we validated that the models F-statistics were also significant at a 95% confidence level.

Then, the two primary statistics used to choose our final model were adjusted R-squared and root mean square error (RMSE). Adjusted R-squared helped guide model selection since, like R-squared, adjusted R-squared measures the amount of variation in the dependent variable explained by the independent variables, except with a correction to ensure only independent variables with predictive power raise the statistic. RMSE was perhaps even more crucial to model selection as it is the measure of the standard deviation of the residuals, essentially a measure of accuracy in the same units as the response variable. To ensure the model can generalize to unobserved data, we calculated the RMSE on our test set.

Backward elimination saw a RMSE of approximately 10, noticeably outperforming other models. Therefore, we chose the backward elimination model even with a slightly worse adjusted R-squared. Additionally, since all top performing models included four predictors, parsimony was not a consideration.

Lastly, we verified the forward selection model meets OLS regression assumptions. These included: no significant multicollinearity, the mean of residuals is zero, homoscedasticity of residuals, and no significant auto-correlation. We deemed all assumptions had been met, but note, there is a slight trend in the residuals vs fitted plot (Assumption: Homoscedasticity of residuals) which may indicate a small nonlinear trend.

Matt's References

Bhandari, Aniruddha, "Key Difference between R-squared and Adjusted R-squared for Regression Analysis", Analytics Vidhya, 2020 https://www.analyticsvidhya.com/blog/2020/07/difference-between-r-squared-and-adjusted-r-squared/

Glen., Stephanie "RMSE: Root Mean Square Error", StatisticsHowTo.com https://www.statisticshowto.com/probability-and-statistics/regression-analysis/rmse-root-mean-square-error/

Gupta, Aryansh, "Linear Regression Assumptions and Diagnostics in R", RPubs, https://rpubs.com/aryn999/LinearRegressionAssumptionsAndDiagnosticsInR

Kim, Bommae, "Understanding Diagnostic Plots for Linear Regression Analysis", University of Virginia Library, https://data.library.virginia.edu/diagnostic-plots/

Code Appendix

```
## DATA EXPLORATION:
#We can observe the response variable (TARGET_WINS) looks to be normally dist
ributed. This supports the working theory that there are good teams and bad t
eams. There are also a lot of average teams.

#There are also quite a few variables with missing values. and,Some variables
are right skewed (TEAM_BASERUN_CS, TEAM_BASERUN_SB, etc.). This might support
the good team theory. It may also introduce non-normally distributed residual
s in the model. We shall see.

### Load the Data


# Set seed for reproducibility
set.seed(621)

train <-read.csv("https://raw.githubusercontent.com/akarimhammoud/Data_621/ma
in/Assignment_1/data/moneyball-training-data.csv")

evaluation <-read.csv("https://raw.githubusercontent.com/akarimhammoud/Data_6
21/main/Assignment_1/data/moneyball-evaluation-data.csv")


# Summary of the data

summary(train)
summary(evaluation)


# Glimpse of the data

glimpse(train)

glimpse(evaluation)


# Find SD for all of the train and test data

apply(train,2,sd, na.rm=TRUE)

apply(evaluation,2,sd, na.rm=TRUE)
```

```r
# Box plot the data

ggplot(stack(train), aes(x = ind, y = values)) +
  geom_boxplot() +
  theme(legend.position="none") +
  theme(axis.text.x=element_text(angle=45, hjust=1))

# Variable Distributions

train %>%
  gather(variable, value, TARGET_WINS:TEAM_FIELDING_DP) %>%
  ggplot(., aes(value)) +
  geom_density(fill = "#3A8B63", color="#3A8B63") +
  facet_wrap(~variable, scales ="free", ncol = 4) +
  labs(x = element_blank(), y = element_blank())


#Log Variable Distributions


train_log <- log(train)

train_log %>%
  gather(variable, value, TARGET_WINS:TEAM_FIELDING_DP) %>%
  ggplot(., aes(value)) +
  geom_density(fill = "#3A8B63", color="#3A8B63") +
  facet_wrap(~variable, scales ="free", ncol = 4) +
  labs(x = element_blank(), y = element_blank())


# Correlations with Response Variable

train %>%
  gather(variable, value, -TARGET_WINS) %>%
  ggplot(., aes(value, TARGET_WINS)) +
  geom_point(fill = "#628B3A", color="#628B3A") +
  geom_smooth(method = "lm", se = FALSE, color = "black") +
  facet_wrap(~variable, scales ="free", ncol = 4) +
  labs(x = element_blank(), y = "Wins")



train %>%
  cor(., use = "complete.obs") %>%
  corrplot(., method = "color", type = "upper", tl.col = "black", diag = FALS
E)
```

```
### DATA PREPARATION

# ^[https://statisticsglobe.com/count-number-of-na-values-in-vector-and-colum
n-in-r]

#NA counts for the train data set
colSums(is.na(train))

# ^[https://cran.r-project.org/web/packages/naniar/vignettes/naniar-visualisa
tion.html]

#visulaization and percentage of NA values
vis_miss(train)



# ^[https://datavizpyr.com/visualizing-missing-data-with-barplot-in-r/]

#alternative NA values visualization
train  %>%
  summarise_all(list(~is.na(.)))%>%
  pivot_longer(everything(),
               names_to = "variables", values_to="missing") %>%
  count(variables, missing) %>%
  ggplot(aes(y=variables,x=n,fill=missing))+
  geom_col()


#Since 92% of the data for the TEAM_BATTING_HBP is missing, the variable has
been removed from both test #and train data. TEAM_BASERUN_CS is a runner up w
ith the next highest amount of NA at 34%.


#removes the TEAM_BATTING_HBP due to high # of NAs
train_full <- train %>% dplyr::select(-c(TEAM_BATTING_HBP))
evaluation <- evaluation %>% dplyr::select(-c(TEAM_BATTING_HBP))


# ^[https://sphweb.bumc.bu.edu/otlt/MPH-Modules/BS/R/R-Manual/R-Manual5.html]

#creates CSV in your current working directory of R
write.csv(train_full,'hw1_train_data.csv')
write.csv(evaluation, 'hw1_evaluation_data.csv')



# Create train, test split
```

```r
train <- train_full %>% dplyr::sample_frac(.75)
test  <- dplyr::anti_join(train_full, train, by = 'INDEX')



## BUILD MODELS


## Model #1
### Two predictors: Base hits by batters and Hits allowed
#Using a manual review, below are the features selected for the first model a
nd the supporting reason/s.

#TEAM_BATTING_H = Base hits by batters:  it's impossible to win in baseball w
ithout getting to the bases # and hitting the ball is the primary means to ac
complish this.

#TEAM_PITCHING_H = Hits allowed: winning without a good defense is difficult
and in baseball preventing #the other team from getting hits is a good defens
e strategy.

#Only two features are selected for the first model - start small and build u
p seems like a good approach.

#<B> Create the Regression Model </B>

# Build the first model and produce a summary
first_model <- lm(TARGET_WINS ~ TEAM_BATTING_H + TEAM_PITCHING_H, data = trai
n)
summary(first_model)



#The p values are 0, which per the criteria of "keep a feature if the p-value
is <0.05" recommends that #we keep both these features.  But, the adjusted R-
squared is TERRIBLE at around 21%.  Even though the #R-squared is poor it's s
imple to run this model with the test data, so we'll do that next.


#Predict with the first model training data
first_model_predictions = predict(first_model, test)

#Evaluate the first model results using RMSE
rmse(test$TARGET_WINS, first_model_predictions)



## Model #2
```

```
### Four predictors: Base hits by batters, Hits allowed, Errors, and Walks al
lowed
#Using a manual review, below are the features selected for the second model
and the supporting reason/s.

#We'll keep the features from the first model (due to low p-values) and add t
wo more features...
#TEAM_FIELDING_E = Errors: errors are costly in terms of immediate impact, bu
t could also impact the team in other ways (i.e. a high occurrence could impa
ct team comraderie and confidence in each other)

#TEAM_PITCHING_BB = Walks allowed: putting players on base for "free" is more
opportunity for points

#<B> Create the Regression Model </B>


# Build the second model and produce a summary
second_model <- lm(TARGET_WINS ~ TEAM_BATTING_H + TEAM_PITCHING_H + TEAM_FIEL
DING_E + TEAM_PITCHING_BB, data = train)
summary(second_model)


#Predict with the second model training data
second_model_predictions = predict(second_model,test)

#Evaluate the second model results using RMSE
rmse(test$TARGET_WINS, second_model_predictions)

#The increase from two features in the first model to four features in the se
cond model did not yield a noticeable improvement.  The Adjusted R2 on the tr
aining data improved slightly, but the RMSE for all practical purposes stayed
the same at around 13; which is a poor RMSE implying that both models have po
or predictive capability.

## Model #3
### BSR Model (SaberMetrics) (data imputation)
# *Base runs (BsR) is a baseball statistic invented by sabermetrician David S
myth to estimate the number of runs a team "should have"*
#*scored given their component offensive statistics, as well as the number of
runs a hitter or pitcher #creates or allows.*
#*It measures essentially the same thing as Bill James runs created, but as s
abermetrician Tom M. Tango points out, base*
#*runs models the reality of the run-scoring process "significantly better th
an any other run estimator".*

#*Cleaning Data*

# load data
```

```
data <- read.csv('hw1_train_data.csv')

#imput data by regression:
data_imp <- mice(data, method = "norm.predict", m = 1)

#complete data
data_complete <- complete(data_imp)


# The simplest, uses only the most common batting statistics[2]

#$A = H + BB - HR$
#$B = (1.4 * TB - .6 * H - 3 * HR + .1 * BB) * 1.02$
#$C = AB - H$
#$D = HR$

#$BsR = \frac{(A * B)}{(B + C)} + D$


data3 <- data_complete %>%
  rowwise() %>%
  mutate(TEAM_BATTING_AB = sum( TEAM_BATTING_H,TEAM_BATTING_BB,TEAM_BATTING_S
O, na.rm=TRUE),
         TEAM_BATTING_1B = TEAM_BATTING_H - (TEAM_BATTING_2B + TEAM_BATTING_3
B + TEAM_BATTING_HR),
         TEAM_BATTING_TB = TEAM_BATTING_1B + (2 * TEAM_BATTING_2B) + (3 * TEA
M_BATTING_3B) + (4 * TEAM_BATTING_HR),
         BSR_A = TEAM_BATTING_H + TEAM_BATTING_BB - TEAM_BATTING_HR,
         BSR_B = (( 1.4 * TEAM_BATTING_TB) - ( 0.6 * TEAM_BATTING_H) - (3 * T
EAM_BATTING_HR) + (0.1 * TEAM_BATTING_BB)) * 1.02,
         BSR_C = TEAM_BATTING_AB - TEAM_BATTING_H,
         BSR = ((BSR_A*BSR_B)/(BSR_B + BSR_C)) + TEAM_BATTING_HR
         )

data3 <- as.data.frame(data3)
train3 <- data3 %>% dplyr::sample_frac(.75)
test3  <- dplyr::anti_join(data3, train3, by = 'X')


#<B> Create the Regression Model </B>
#*BSR*

rmdata3 <- train3 %>%
  dplyr::select(BSR, TEAM_PITCHING_SO, TEAM_FIELDING_E, TEAM_FIELDING_DP, TAR
GET_WINS)

#Build the second model and produce a summary
GModel3 <- lm(TARGET_WINS ~ BSR + TEAM_PITCHING_SO + TEAM_FIELDING_E + TEAM_F
IELDING_DP, data = rmdata3)
```

```
summary(GModel3)



#Predict with the second model training data
GModel3_predictions = predict(GModel3,test3)

#Evaluate the second model results using RMSE
rmse(test3$TARGET_WINS, GModel3_predictions)





## Model #4
### (Modified) Backward Elimination Model (omitting NAs)

#Due to previously learning how to perform Backward Elimination and it being
possible to perform manually, we decided to include a model that resulted fro
m the procedure. The process was performed with imputed data (via MICE) as we
ll as data with NAs removed. The latter showed stronger results, therefore th
e final model was fitted with the NA omitted data.

#According to Faraway, Backward Elimination is when you start with all predic
tors in the model, then remove the predictor with the highest p-value as long
as it is above your p-value threshold (e.g. 0.05). Then refit the model and c
ontinue the process until only predictors with p-values below your threshold
remain.

#Additionally, we took steps to remove variables with non-intuitive coefficie
nts. For instance, TEAM_FIELDING_DP and TEAM_PITCHING_SO were unexpectedly sh
owing negative effects on wins. While there could be potential intervening va
riables giving these variables true predictive power, we opted to remove the
variables from the model due to the possibility they were significant by chan
ce and due to our bias towards parsimony. Further, RMSE did not drastically w
orsen when removed.


# Remove NAs
train_no_na <- na.omit(train)
test_no_na <- na.omit(test)

# Fit model
backward_model <- lm(TARGET_WINS ~ TEAM_BASERUN_SB + TEAM_BATTING_HR + TEAM_B
ATTING_BB + TEAM_BASERUN_SB
                     + TEAM_PITCHING_SO + TEAM_FIELDING_E + TEAM_FIELDING_DP,
data = test_no_na)

# Fit modified model
```

```r
backward_mod_model <- lm(TARGET_WINS ~ TEAM_BASERUN_SB + TEAM_BATTING_HR + TE
AM_BATTING_BB + TEAM_FIELDING_E,
                     data = test_no_na)


# View summary
summary(backward_mod_model)



# Make predictions on test set
backward_model_predictions = predict(backward_mod_model, test_no_na)

# Obtain RMSE between actuals and predicted
rmse(test_no_na$TARGET_WINS, backward_model_predictions)



# Make predictions on evaluation data
backward_model_predictions_evaluation = predict(backward_mod_model, evaluatio
n)

# Final predictions on evaluation set
write.csv(backward_model_predictions_evaluation, 'evaluation_predictions.csv'
)




## SELECT MODELS


### Verifying OLS Regression Assumptions

# Assumption: No Multicollinearity (VIF under 5)
vif(backward_mod_model)

# Assumption: Mean of residuals is zero
mean(residuals(backward_mod_model))

# Assumption: Homoscedasticity of residuals
plot(backward_mod_model)

# Assumption: No auto-correlation
acf(residuals(backward_mod_model), lags=20)
```