Multiple Linear Regression for Baseball Win Prediction

Homework #1

Naomi Buell Richie Rivera Alexander Simon

2026-03-01

1 Introduction

We analyze and model data on a professional baseball team from 1871 to 2006. 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. We build a multiple linear regression model on the training data to predict the number of wins for the team.

2 Methods

2.1 Data Exploration

The mondeyball training data set contains 2276 rows and 17 features, and is 91.01 percent complete. The data are all numeric, and the target variable is `TARGET_WINS`, which represents the number of games won by the team in a season. The other variables represent various performance statistics for the team. See Table 1 for descriptive characteristics of the training data. Variables range from 0% to 92% missing. We address missingness by dropping columns and imputing in the Data Preparation section.

> **Note**
>
> NB: Not sure if target_wins represents the number of games won per season. Is each row a season? Is this the correct interpretation of the target variable?

Table 1: Summary statistics for all variables in the training dataset

Variable	Completeness rate	Mean	Min	P25	P50	P75	Max
INDEX	1.000	1300	1	630	1300	1900	2500
TARGET_WINS	1.000	81	0	71	82	92	150

Table 1: Summary statistics for all variables in the training dataset

Variable	Completeness rate	Mean	Min	P25	P50	P75	Max
TEAM_BATTING_H	1.000	1500	890	1400	1500	1500	2600
TEAM_BATTING_2B	1.000	240	69	210	240	270	460
TEAM_BATTING_3B	1.000	55	0	34	47	72	220
TEAM_BATTING_HR	1.000	100	0	42	100	150	260
TEAM_BATTING_BB	1.000	500	0	450	510	580	880
TEAM_BATTING_SO	0.960	740	0	550	750	930	1400
TEAM_BASERUN_SB	0.940	120	0	66	100	160	700
TEAM_BASERUN_CS	0.660	53	0	38	49	62	200
TEAM_BATTING_HBP	0.084	59	29	50	58	67	95
TEAM_PITCHING_H	1.000	1800	1100	1400	1500	1700	30000
TEAM_PITCHING_HR	1.000	110	0	50	110	150	340
TEAM_PITCHING_BB	1.000	550	0	480	540	610	3600
TEAM_PITCHING_SO	0.960	820	0	620	810	970	19000
TEAM_FIELDING_E	1.000	250	65	130	160	250	1900
TEAM_FIELDING_DP	0.870	150	52	130	150	160	230

The correlation between all variables are shown Figure 1. The most correlated variables with the target variable `TARGET_WINS` are `TEAM_BATTING_H`, `TEAM_BATTING_2B`, and `TEAM_BATTING_BB`, which may be strong predictors of wins. The most autocorrelated variables are `TEAM_BATTING_HR` and `TEAM_PITCHING_HR`, which makes sense as teams that hit many homeruns may also allow many homeruns, for example.

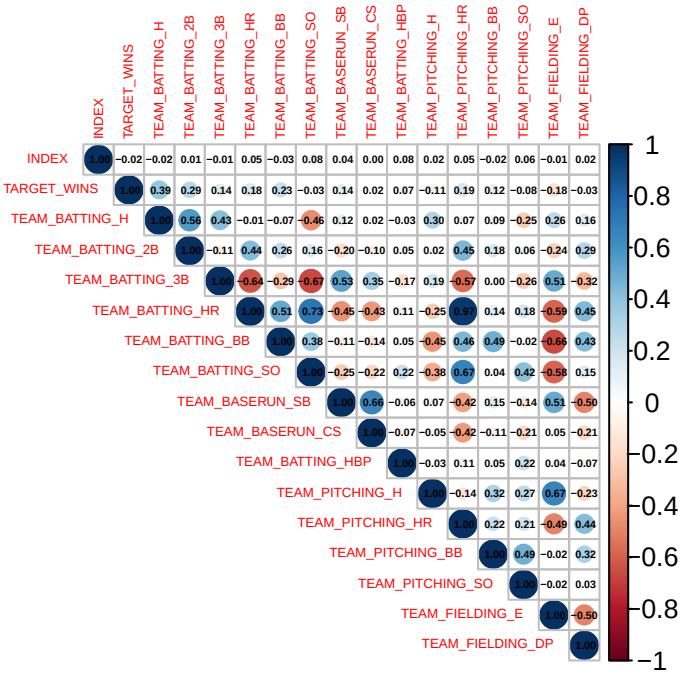


Figure 1: Correlation matrix of all variables in the training dataset

> **Note**
>
> NB: Consider adding Bar Chart or Box Plot of the data or anything else that you think is interesting.

2.2 Data Preparation

We handled high missingness in the variables `TEAM_BATTING_HBP`, `TEAM_BASERUN_CS`, and `TEAM_FIELDING_DP` (91.6%, 33.9%, and 12.6% missing, respectively), which are not very correlated with the target variable, by excluding these variables. The remaining variables are less than 10% missing, so we will impute the missing values with the median of each variable. We avoid dropping `TEAM_BASERUN_SB` as it is the most correlated variable with the target variable that has missingness, and instead impute it to preserve this potentially strong predictor of wins.

The median imputation was picked based on the distributions of the variables with missingness: `TEAM_BASERUN_SB` and `TEAM_PITCHING_SO` have a long right tail so the median is a better imputation method than the mean. `TEAM_BATTING_SO` is bimodal, so the median is also a better imputation method than the mean for this variable. See the histograms above for the distributions of these variables.

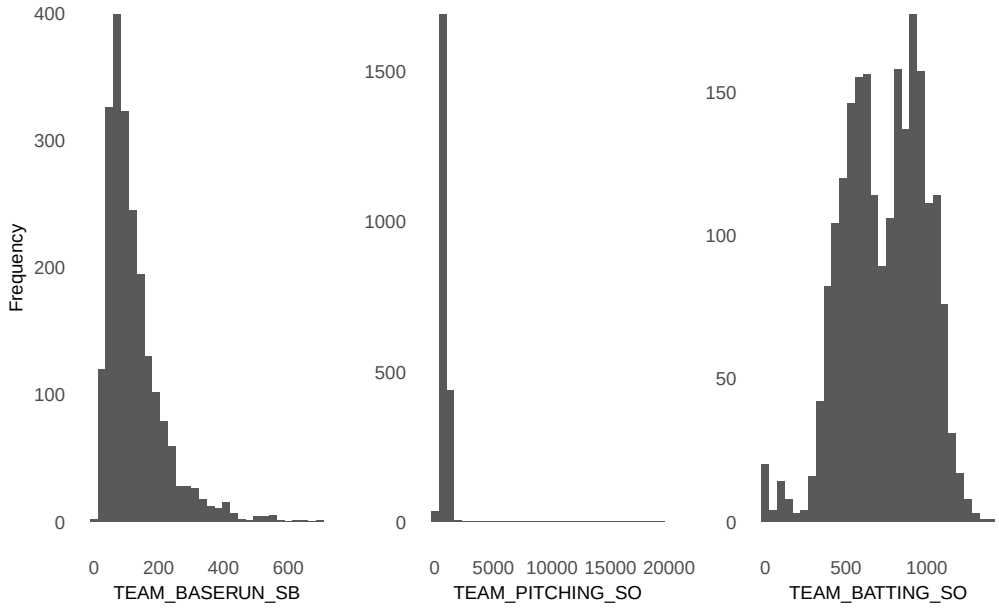


Figure 2: Histograms of variables with missingness selected for imputation

Describe how you have transformed the data by changing the original variables or creating new variables. If you did transform the data or create new variables, discuss why you did this. Here are some possible transformations.

b. *Create flags to suggest if a variable was missing*
c. *Transform data by putting it into buckets*
d. *Mathematical transforms such as log or square root (or use Box-Cox) e. Combine variables (such as ratios or adding or multiplying) to create new variables*

2.3 Build Models

Using the training data set, build at least three different multiple linear regression models, using different variables (or the same variables with different transformations). Since we have not yet covered automated variable selection methods, you should select the variables manually (unless you previously learned Forward or Stepwise selection, etc.). Since you manually selected a variable for inclusion into the model or exclusion into the model, indicate why this was done. Discuss the coefficients in the models, do they make sense? For example, if a team hits a lot of Home Runs, it would be reasonably expected that such a team would win more games. However, if the coefficient is negative (suggesting that the team would lose more games), then that needs to be discussed. Are you keeping the model even though it is counter intuitive? Why? The boss needs to know.

2.4 Select Models

Decide on the criteria for selecting the best multiple linear regression model. Will you select a model with slightly worse performance if it makes more sense or is more parsimonious? Discuss why you selected your model. For the multiple linear regression model, will you use a metric such as Adjusted R2 , RMSE, etc.? Be sure to explain how you can make inferences from the model, discuss multi-collinearity issues (if any), and discuss other relevant model output. Using the training data set, evaluate the multiple linear regression model based on (a) mean squared error, (b) R2 , (c) F-statistic, and (d) residual plots. Make predictions using the evaluation data set.

3 Results

We predict that…

4 Conclusion

In conclusion…

5 Appendix: R Code

5.1 1. Data Exploration

First, we preview the data.

```
moneyball_train |> head()
```

```
# A tibble: 6 x 17
  INDEX TARGET_WINS TEAM_BATTING_H TEAM_BATTING_2B TEAM_BATTING_3B
  <dbl>       <dbl>          <dbl>            <dbl>            <dbl>
1     1          39           1445              194               39
2     2          70           1339              219               22
3     3          86           1377              232               35
4     4          70           1387              209               38
5     5          82           1297              186               27
6     6          75           1279              200               36
# i 12 more variables: TEAM_BATTING_HR <dbl>, TEAM_BATTING_BB <dbl>,
#   TEAM_BATTING_SO <dbl>, TEAM_BASERUN_SB <dbl>, TEAM_BASERUN_CS <dbl>,
#   TEAM_BATTING_HBP <dbl>, TEAM_PITCHING_H <dbl>, TEAM_PITCHING_HR <dbl>,
#   TEAM_PITCHING_BB <dbl>, TEAM_PITCHING_SO <dbl>, TEAM_FIELDING_E <dbl>,
#   TEAM_FIELDING_DP <dbl>
```

The data are all numeric, and the target variable is `TARGET_WINS`, which represents the number of

wins for the team in that season. The other variables represent various performance statistics for the team.

We also check summary statistics for all variables in our training data, including checking missingness.

```
moneyball_train |> skim()
```

Table 2: Data summary

Name	moneyball_train
Number of rows	2276
Number of columns	17
_____	
Column type frequency:	
numeric	17
_____	
Group variables	None

Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
INDEX	0	1.00	1268.46	736.35	1	630.75	1270.5	1915.50	2535	
TARGET_WINS	0	1.00	80.79	15.75	0	71.00	82.0	92.00	146	
TEAM_BATTING_H	0	1.00	1469.27	144.59	891	1383.00	1454.0	1537.25	2554	
TEAM_BATTING_2B	0	1.00	241.25	46.80	69	208.00	238.0	273.00	458	
TEAM_BATTING_3B	0	1.00	55.25	27.94	0	34.00	47.0	72.00	223	
TEAM_BATTING_HR	0	1.00	99.61	60.55	0	42.00	102.0	147.00	264	
TEAM_BATTING_BB	0	1.00	501.56	122.67	0	451.00	512.0	580.00	878	
TEAM_BATTING_SO	102	0.96	735.61	248.53	0	548.00	750.0	930.00	1399	
TEAM_BASERUN_SB	131	0.94	124.76	87.79	0	66.00	101.0	156.00	697	
TEAM_BASERUN_CS	772	0.66	52.80	22.96	0	38.00	49.0	62.00	201	
TEAM_BATTING_HBP	2085	0.08	59.36	12.97	29	50.50	58.0	67.00	95	

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
TEAM_PITCHING_H	0	1.00	1779.21	1406.84	1137	1419.00	1518.0	1682.50	30132	
TEAM_PITCHING_HR	0	1.00	105.70	61.30	0	50.00	107.0	150.00	343	
TEAM_PITCHING_BB	0	1.00	553.01	166.36	0	476.00	536.5	611.00	3645	
TEAM_PITCHING_SO	102	0.96	817.73	553.09	0	615.00	813.5	968.00	19278	
TEAM_FIELDING_E	0	1.00	246.48	227.77	65	127.00	159.0	249.25	1898	
TEAM_FIELDING_DP	286	0.87	146.39	26.23	52	131.00	149.0	164.00	228	

```
nrows <- moneyball_train |> nrow()
ncols <- moneyball_train |> length()
perc_complete <- round(moneyball_train |> n_complete() / nrows / ncols * 100, 2)
```

This training dataset contains 2276 rows and 17 features, and is 91.01 percent complete. The variables with missingness are:

```
moneyball_train |>
    summarise(across(everything(), ~ sum(is.na(.)) / nrows * 100)) |>
    pivot_longer(everything(), names_to = "variable", values_to = "missing_percentage") |>
    arrange(desc(missing_percentage)) |>
    filter(missing_percentage > 0)
```

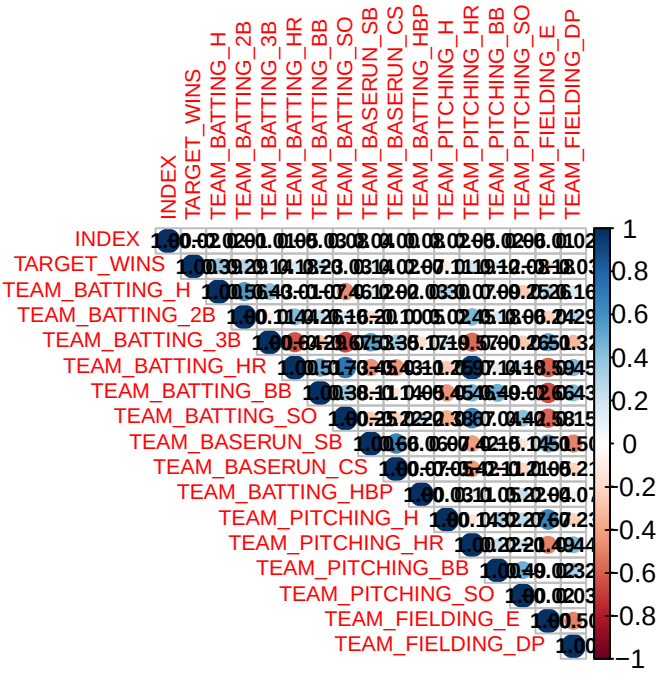
```
# A tibble: 6 x 2
  variable         missing_percentage
  <chr>                         <dbl>
1 TEAM_BATTING_HBP              91.6
2 TEAM_BASERUN_CS               33.9
3 TEAM_FIELDING_DP              12.6
4 TEAM_BASERUN_SB                5.76
5 TEAM_BATTING_SO                4.48
6 TEAM_PITCHING_SO               4.48
```

This missingness will need to be addressed in the data preparation step.

We also visualize correlations between all variables.

```
moneyball_train_cor <- moneyball_train |>
    cor(use = "pairwise.complete.obs")

corrplot(moneyball_train_cor, method = "circle", type = "upper", tl.cex = 0.7, sig.level = 0.05
```



The most correlated variables with the target variable `TARGET_WINS` are:

```
moneyball_train_cor |>
    as.data.frame() |>
    rownames_to_column(var = "variable") |>
    select(variable, TARGET_WINS) |>
    filter(variable != "TARGET_WINS") |>
    arrange(desc(abs(TARGET_WINS))) |>
    head(10)
```

```
           variable TARGET_WINS
1    TEAM_BATTING_H   0.3887675
2   TEAM_BATTING_2B   0.2891036
3   TEAM_BATTING_BB   0.2325599
4  TEAM_PITCHING_HR   0.1890137
5   TEAM_FIELDING_E  -0.1764848
6   TEAM_BATTING_HR   0.1761532
7   TEAM_BATTING_3B   0.1426084
8   TEAM_BASERUN_SB   0.1351389
9  TEAM_PITCHING_BB   0.1241745
```

```
10  TEAM_PITCHING_H  -0.1099371
```

These may be strong predictors of wins.

The most autocorrelated variables are:

```
moneyball_train_cor |>
    as.data.frame() |>
    rownames_to_column(var = "variable1") |>
    pivot_longer(-variable1, names_to = "variable2", values_to = "correlation") |>
    filter(variable1 != variable2) |>
    distinct(across(c(correlation)), .keep_all = TRUE) |>
    arrange(desc(abs(correlation))) |>
    head(20)
```

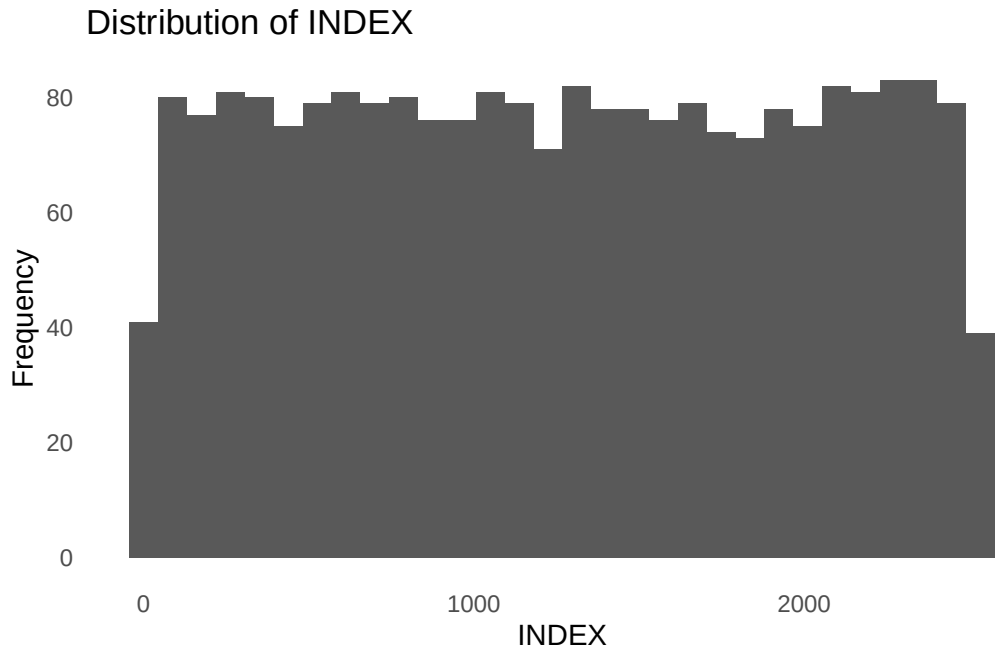
```
# A tibble: 20 x 3
   variable1        variable2        correlation
   <chr>            <chr>                  <dbl>
 1 TEAM_BATTING_HR  TEAM_PITCHING_HR       0.969
 2 TEAM_BATTING_HR  TEAM_BATTING_SO        0.727
 3 TEAM_BATTING_3B  TEAM_BATTING_SO       -0.670
 4 TEAM_PITCHING_H  TEAM_FIELDING_E        0.668
 5 TEAM_BATTING_SO  TEAM_PITCHING_HR       0.667
 6 TEAM_BATTING_BB  TEAM_FIELDING_E       -0.656
 7 TEAM_BASERUN_SB  TEAM_BASERUN_CS        0.655
 8 TEAM_BATTING_3B  TEAM_BATTING_HR       -0.636
 9 TEAM_BATTING_HR  TEAM_FIELDING_E       -0.587
10 TEAM_BATTING_SO  TEAM_FIELDING_E       -0.585
11 TEAM_BATTING_3B  TEAM_PITCHING_HR      -0.568
12 TEAM_BATTING_H   TEAM_BATTING_2B        0.563
13 TEAM_BATTING_3B  TEAM_BASERUN_SB        0.534
14 TEAM_BATTING_HR  TEAM_BATTING_BB        0.514
15 TEAM_BATTING_3B  TEAM_FIELDING_E        0.510
16 TEAM_BASERUN_SB  TEAM_FIELDING_E        0.510
17 TEAM_FIELDING_E  TEAM_FIELDING_DP      -0.498
18 TEAM_BASERUN_SB  TEAM_FIELDING_DP      -0.497
19 TEAM_PITCHING_HR TEAM_FIELDING_E       -0.493
20 TEAM_BATTING_BB  TEAM_PITCHING_BB       0.489
```

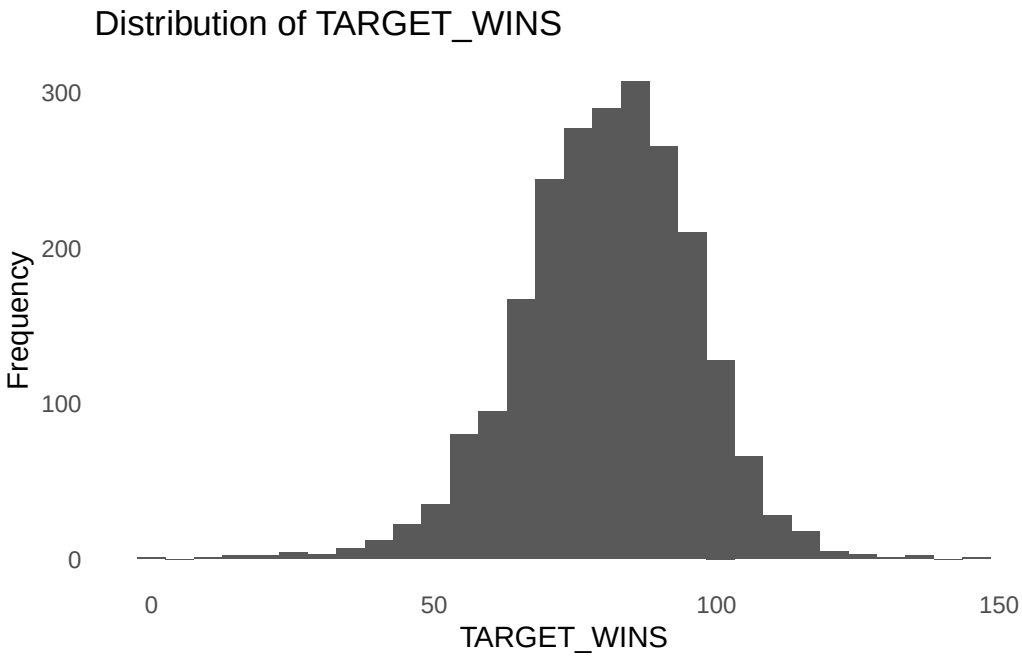
It makes sense that homeruns by batters (`TEAM_BATTING_HR`) and homeruns allowed (`TEAM_PITCHING_HR`) are correlated, as teams that hit many homeruns may also allow many homeruns, for example. We may need to avoid including these pairs of variables in the same model

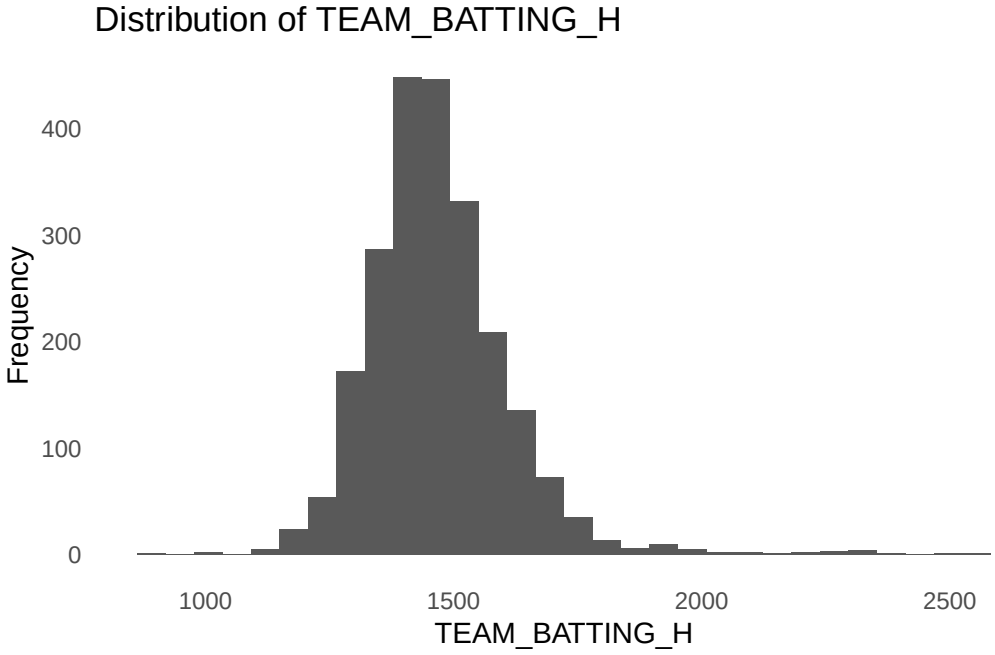
to prevent multicollinearity.

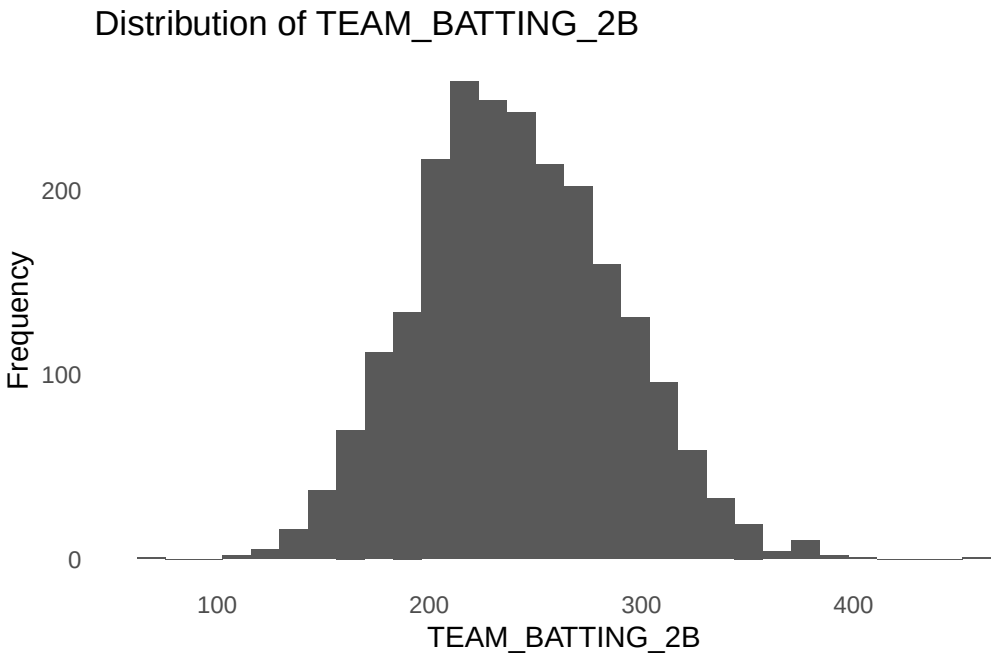
We also visualize the distribution of all variables.

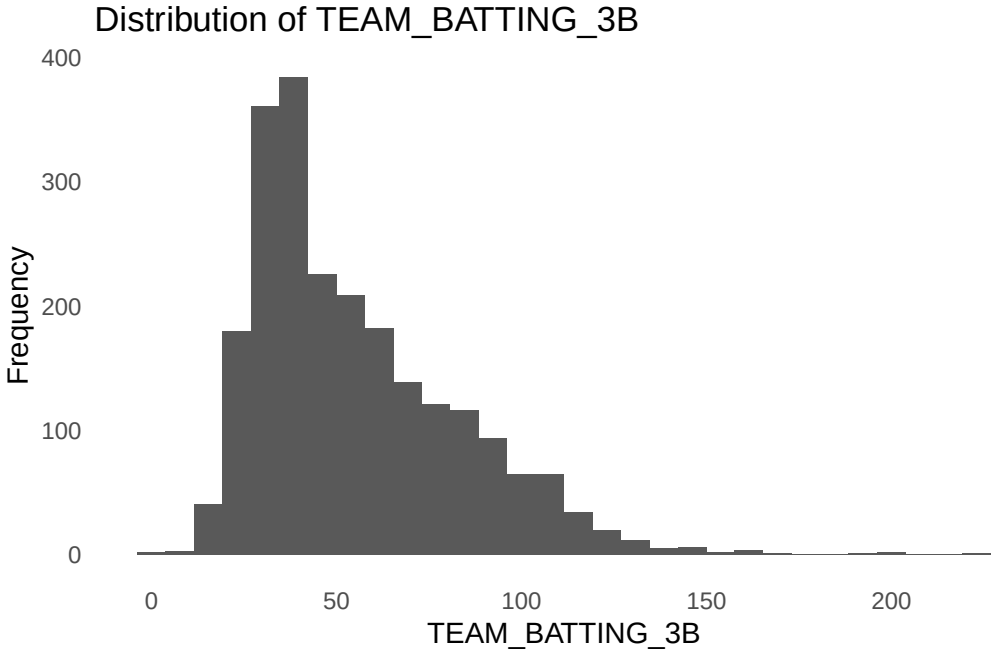
```
# Loop through all variables and create histograms
for (var in names(moneyball_train)) {
    p <- ggplot(moneyball_train, aes_string(x = var)) +
        geom_histogram() +
        labs(title = paste("Distribution of", var), x = var, y = "Frequency") +
        theme_minimal() +
        theme(panel.grid = element_blank())
    print(p)
}
```

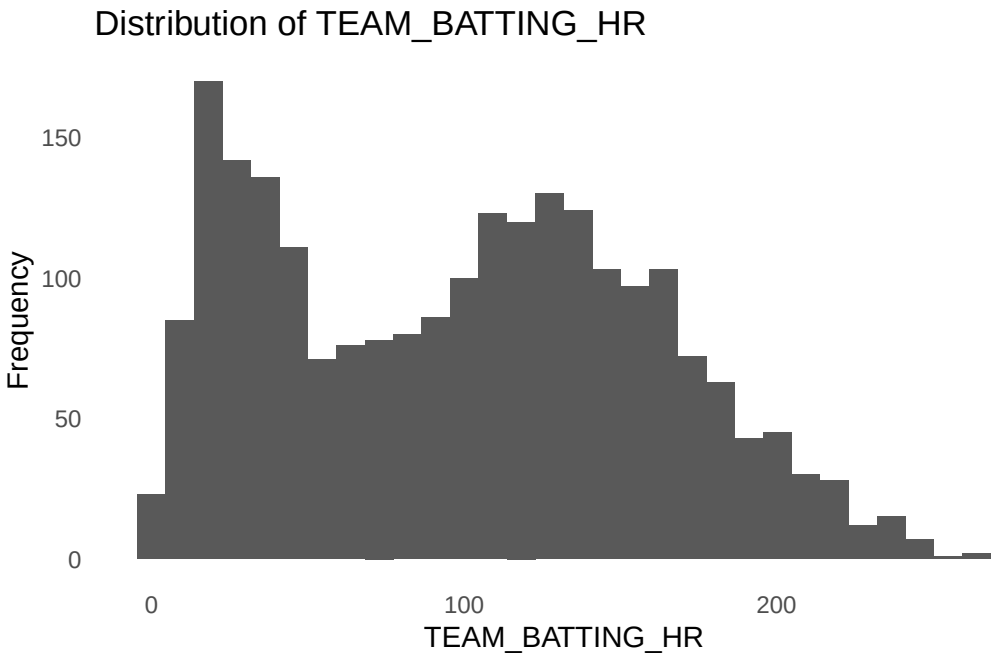


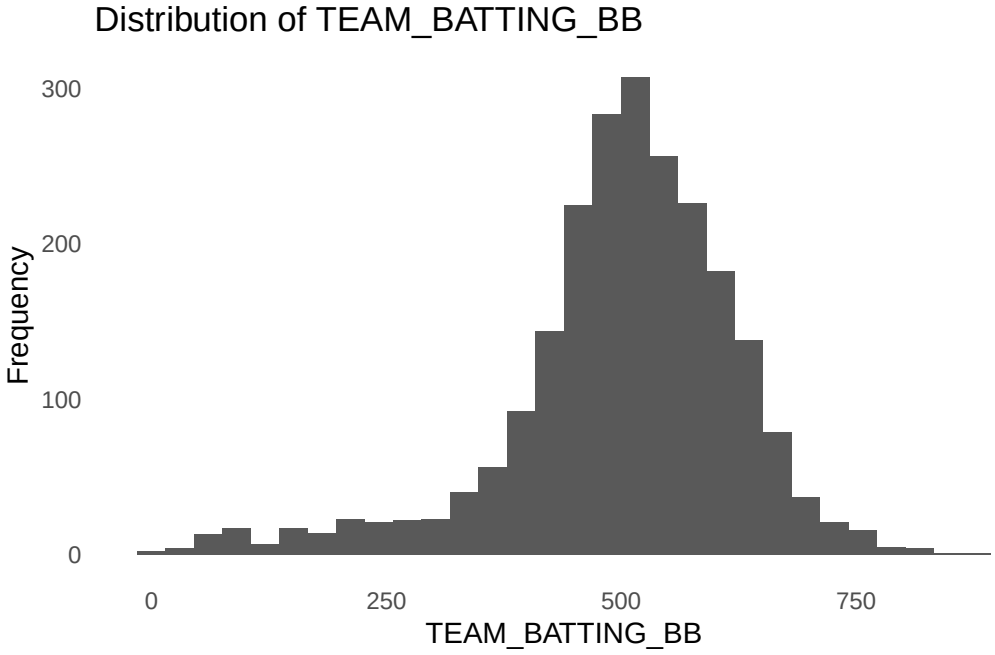

Distribution of TARGET_WINS
Frequency
300
200
100
0
0
50
100
150
TARGET_WINS

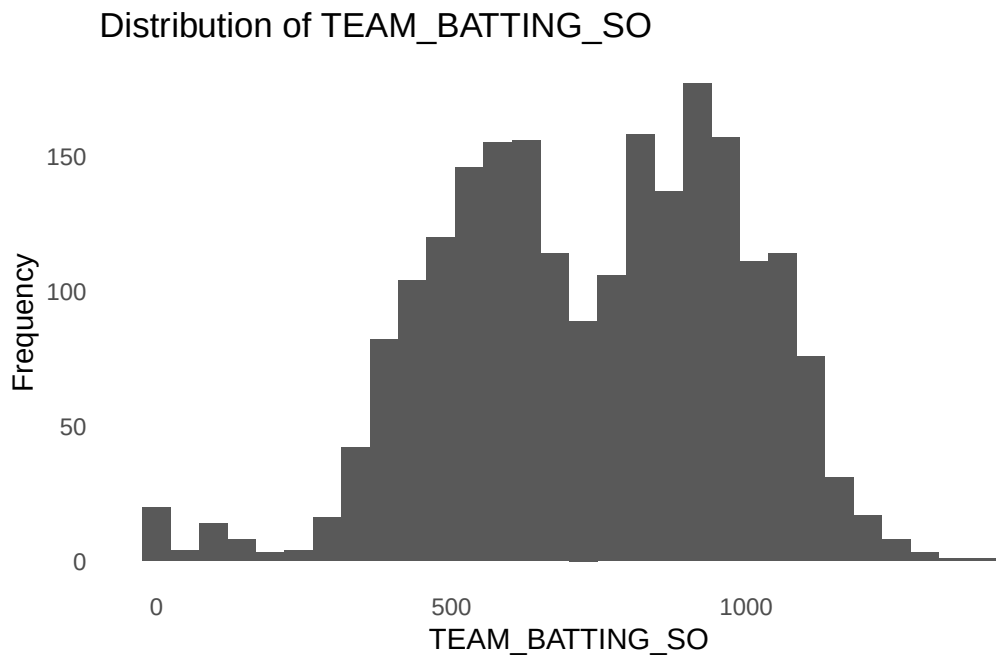

Distribution of TEAM_BATTING_H
Frequency
400
300
200
100
0
1000
1500
2000
2500
TEAM_BATTING_H

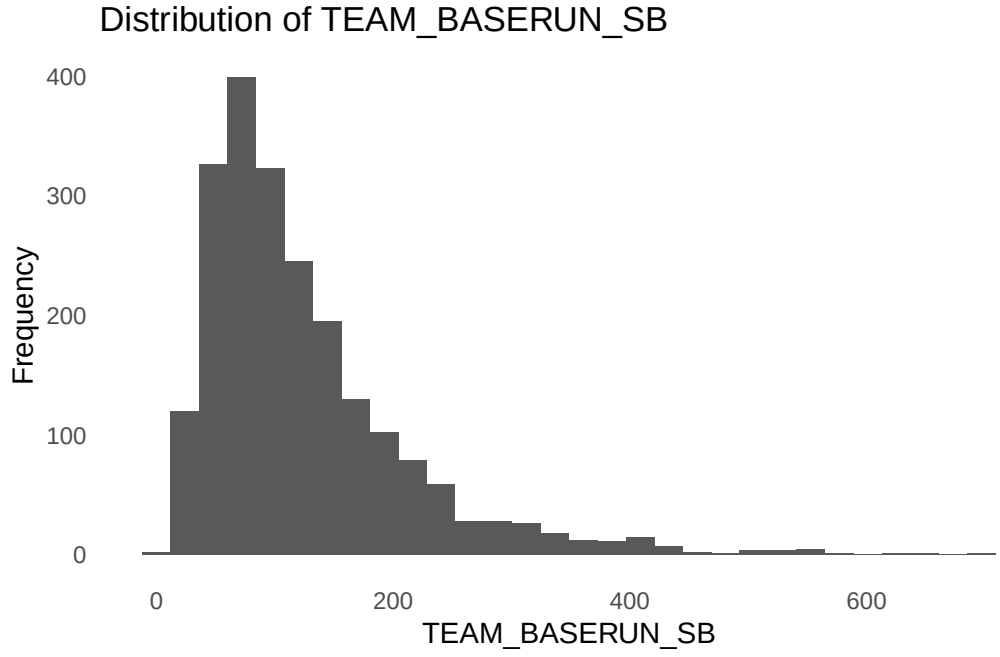

Distribution of TEAM_BATTING_2B
Frequency
0
100
200
100
200
300
400
TEAM_BATTING_2B

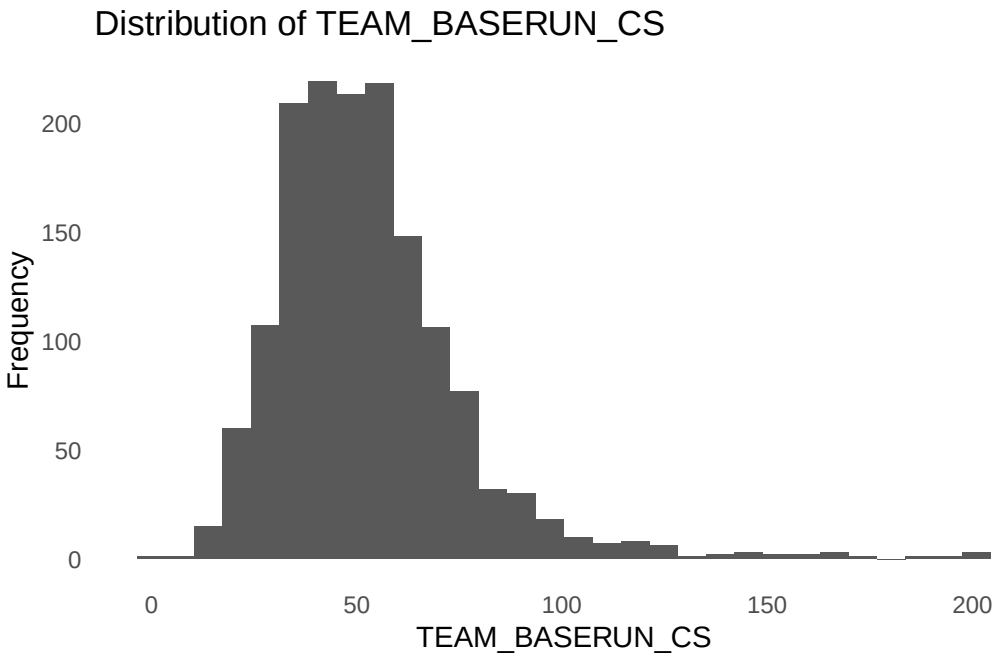

Distribution of TEAM_BATTING_3B
Frequency
0
100
200
300
400
0
50
100
150
200
TEAM_BATTING_3B

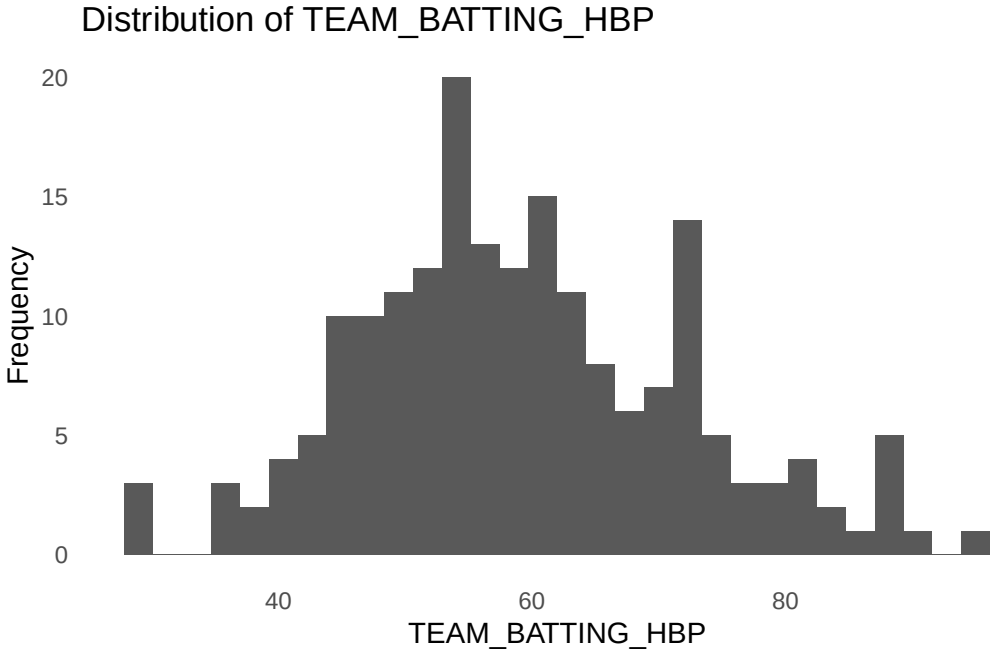

Distribution of TEAM_BATTING_HR
Frequency
150
100
50
0
0
100
200
TEAM_BATTING_HR

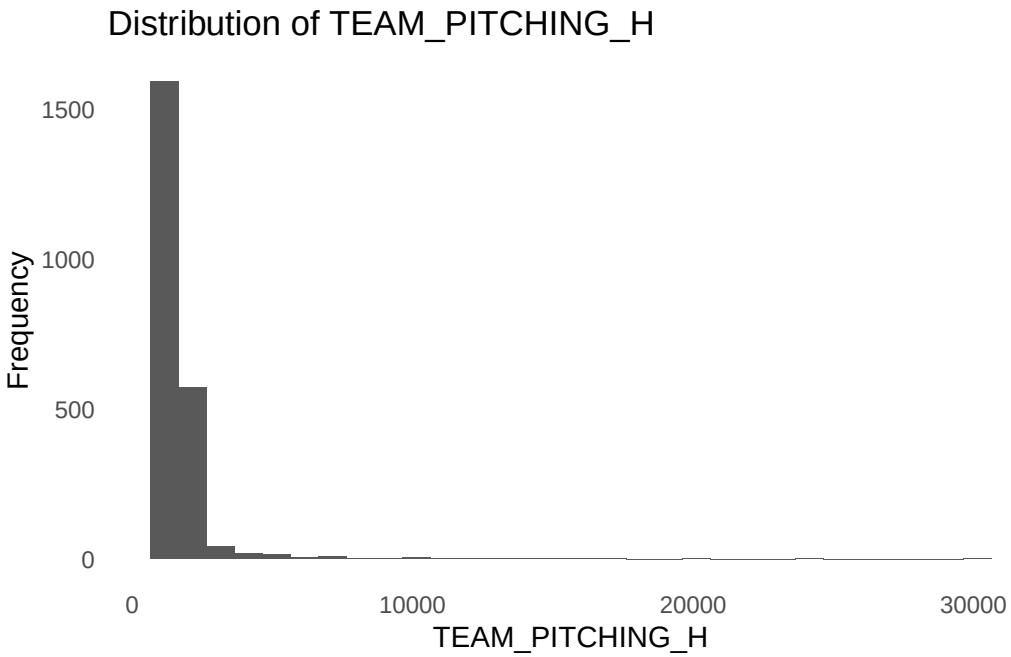

Distribution of TEAM_BATTING_BB
Frequency
300
200
100
0
0
250
500
750
TEAM_BATTING_BB

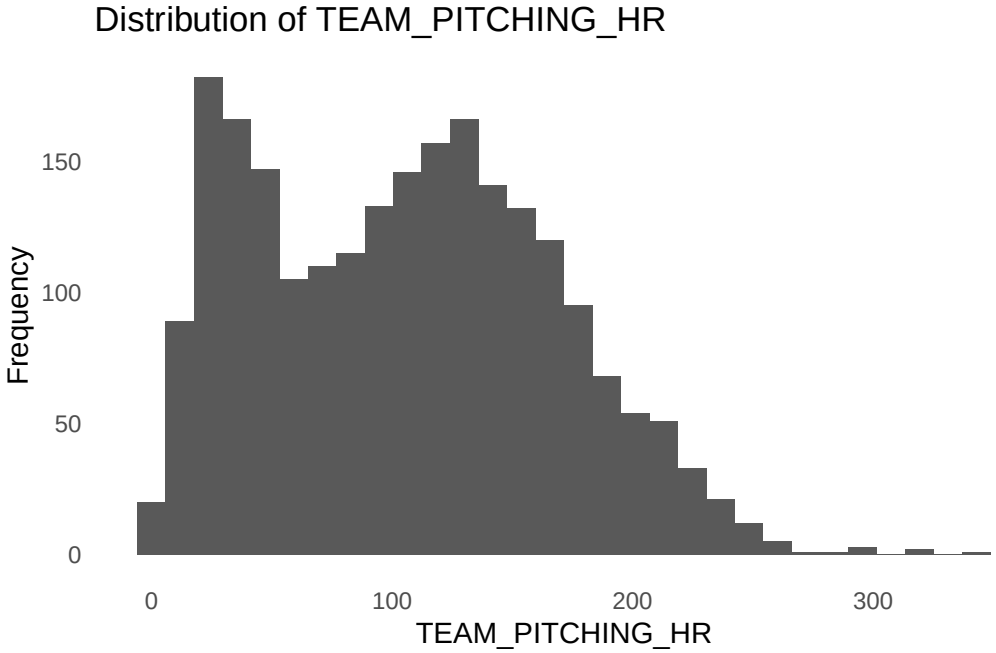

Distribution of TEAM_BATTING_SO
Frequency
150
100
50
0
0
500
1000
TEAM_BATTING_SO

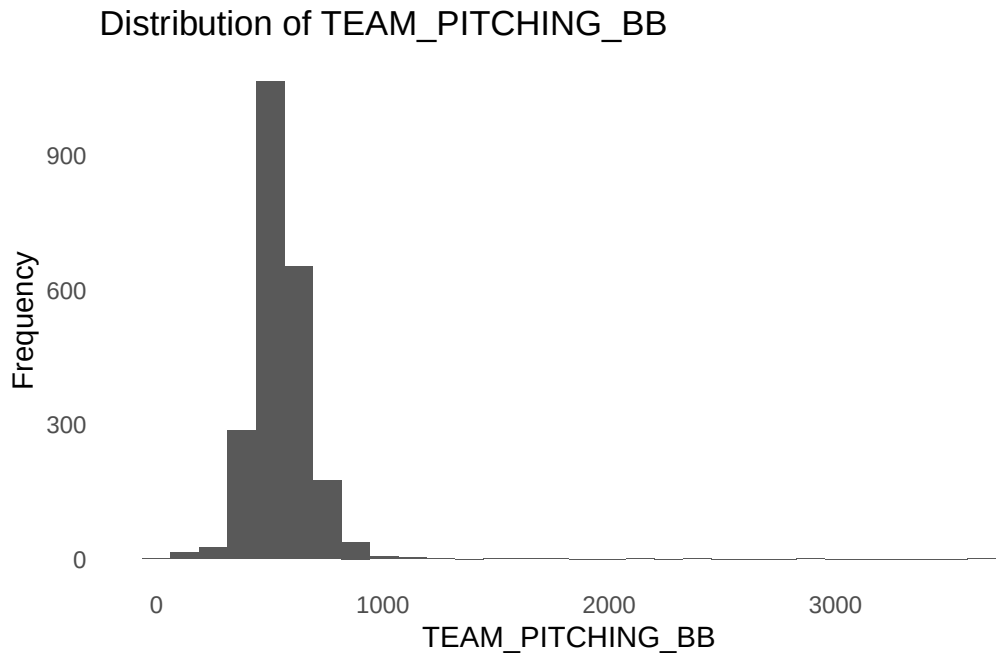

Distribution of TEAM_BASERUN_SB
Frequency
400
300
200
100
0
0
200
400
600
TEAM_BASERUN_SB

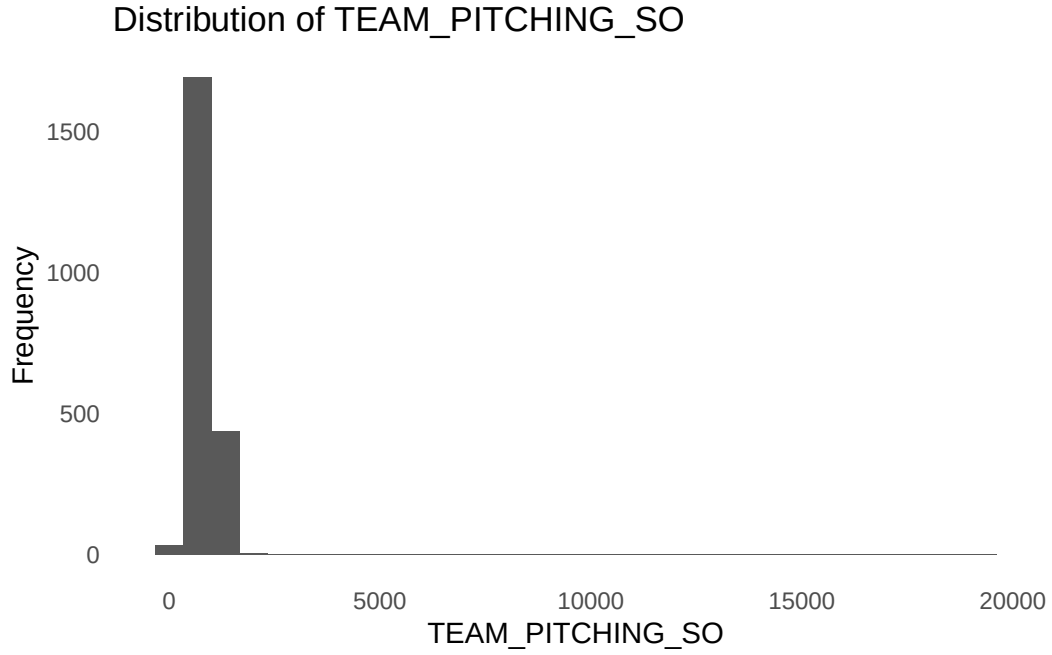

Distribution of TEAM_BASERUN_CS
Frequency
0
50
100
150
200
TEAM_BASERUN_CS

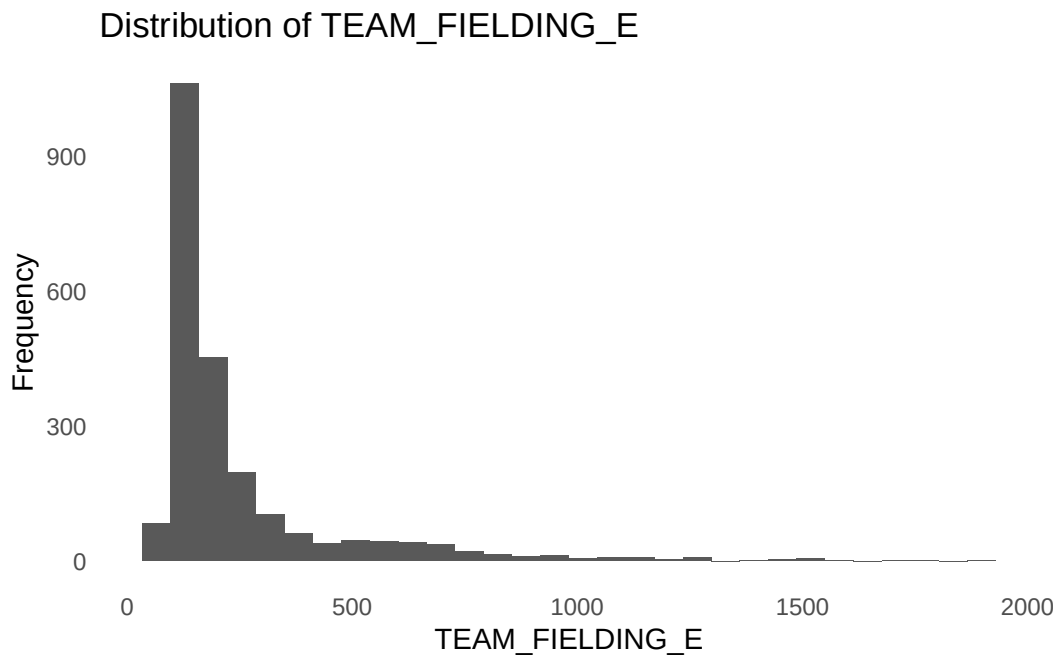

Distribution of TEAM_BATTING_HBP
Frequency
0
5
10
15
20
40
60
80
TEAM_BATTING_HBP

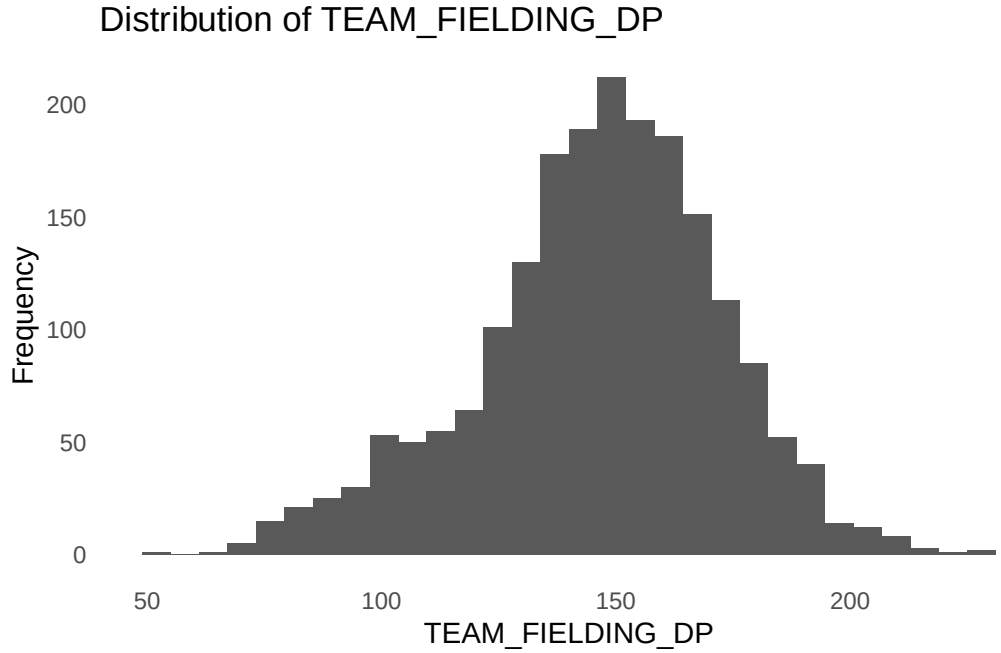

Distribution of TEAM_PITCHING_H
Frequency
1500
1000
500
0
0
10000
20000
30000
TEAM_PITCHING_H


Distribution of TEAM_PITCHING_HR
Frequency
150
100
50
0
0
100
200
300
TEAM_PITCHING_HR


Distribution of TEAM_PITCHING_BB
Frequency
900
600
300
0
0
1000
2000
3000
TEAM_PITCHING_BB


Distribution of TEAM_PITCHING_SO
Frequency
1500
1000
500
0
0
5000
10000
15000
20000
TEAM_PITCHING_SO





5.2 2. Data Preparation

Next, we will prepare the data for modeling.

First, we handle missing data. Since we saw in the EDA that `TEAM_BATTING_HBP`, `TEAM_BASERUN_CS`, and `TEAM_FIELDING_DP` were 91.6%, 33.9%, and 12.6% missing, respectively, and not very correlated with the target variable, so we will drop these variables. The remaining variables are less than 10% missing, so we will impute the missing values with the median of each variable. `TEAM_BASERUN_SB`

is the most correlated variable with the target variable that has missingness, so we will be sure to impute this variable rather than drop it.

The median imputation was picked based on the distributions of the variables with missingness: `TEAM_BASERUN_SB` and `TEAM_PITCHING_SO` have a long right tail so the median is a better imputation method than the mean. `TEAM_BATTING_SO` is bimodal, so the median is also a better imputation method than the mean for this variable. See the histograms above for the distributions of these variables.

```
moneyball_train_prepared <- moneyball_train |>
    select(-TEAM_BATTING_HBP, -TEAM_BASERUN_CS, -TEAM_FIELDING_DP) |>
    mutate(across(everything(), ~ ifelse(is.na(.), median(., na.rm = TRUE), .)))
```

5.3 3. Build Models

5.4 4. Select Models

Export assigned predictions (the number of wins for the team) for the evaluation data set for deliverable.