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 Appendix **?@tbl-summary-statistics** for a preview of the data.

NOTE: Is this the correct interpretation of the target variable? If not, please correct it.

As shown in Figure 1, the correlation between all variables in the training data set reveals important relationships. The most correlated variables with the target variable `TARGET_WINS` are `TEAM_BASERUN_SB`, `TEAM_BATTING_HR`, and `TEAM_PITCHING_SO`, 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. We may need to avoid including these pairs of variables in the same model to prevent multicollinearity.

Describe the size and the variables in the moneyball training data set. Consider that too much detail will cause a manager to lose interest while too little detail will make the manager consider that you aren't doing your job. Some suggestions are given below. Please do NOT treat this as a check list of things to do to complete the assignment. You should have your own thoughts on what to tell the boss. These are just ideas.

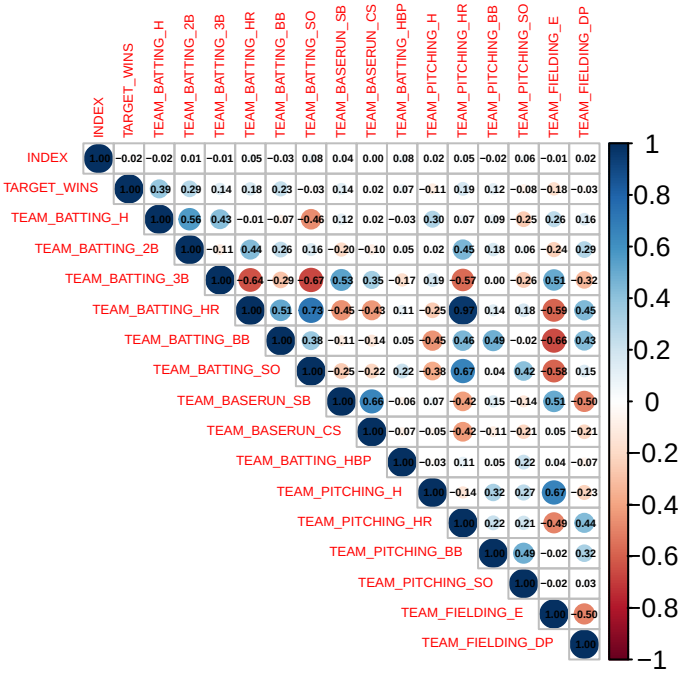


Figure 1: Correlation matrix of all variables in the training dataset

a. Mean / Standard Deviation / Median
b. Bar Chart or Box Plot of the data
c. Is the data correlated to the target variable (or to other variables?)
d. Are any of the variables missing and need to be imputed "fixed"?

2.2 Data Preparation

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.

a. Fix missing values (maybe with a Mean or Median value)
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 Conclusion

4 Appendix: R Code

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() |> as_tibble() |> knitr::kable()
```

skim_type	skim_variable	n_missing	complete_rate	numeric.mean	numeric.sd	numeric.p0	numeric.p25	numeric.p50	numeric.p75	numeric.p100	numeric.hist
numeric	INDEX	0	1.0000000	1268.46373	736.34904	1	630.75	1270.5	1915.50	2535	
numeric	TARGET_WINS	0	1.0000000	80.79086	15.75215	0	71.00	82.0	92.00	146	
numeric	TEAM_BATTING_H	0	1.0000000	1469.26977	144.59120	891	1383.00	1454.0	1537.25	2554	
numeric	TEAM_BATTING_2B	0	1.0000000	241.24692	46.80141	69	208.00	238.0	273.00	458	
numeric	TEAM_BATTING_3B	0	1.0000000	55.25000	27.93856	0	34.00	47.0	72.00	223	
numeric	TEAM_BATTING_HR	0	1.0000000	99.61204	60.54687	0	42.00	102.0	147.00	264	
numeric	TEAM_BATTING_BB	0	1.0000000	501.55888	122.67086	0	451.00	512.0	580.00	878	
numeric	TEAM_BATTING_SO	102	0.9551845	735.60534	248.52642	0	548.00	750.0	930.00	1399	
numeric	TEAM_BASERUN_SB	131	0.9484429	124.76177	87.79117	0	66.00	101.0	156.00	697	
numeric	TEAM_BASERUN_CS	772	0.6954377	52.80386	22.95634	0	38.00	49.0	62.00	201	
numeric	TEAM_BATTING_HBP	2085	0.0839192	59.35602	12.96712	29	50.50	58.0	67.00	95	
numeric	TEAM_PITCHING_H	0	1.0000000	1779.21046	1406.84293	1137	1419.00	1518.0	1682.50	30132	
numeric	TEAM_PITCHING_HR	0	1.0000000	105.69856	61.29875	0	50.00	107.0	150.00	343	
numeric	TEAM_PITCHING_BB	0	1.0000000	553.00791	166.35736	0	476.00	536.5	611.00	3645	
numeric	TEAM_PITCHING_SO	102	0.9551845	817.73045	553.08503	0	615.00	813.5	968.00	19278	
numeric	TEAM_FIELDING_E	0	1.0000000	246.48067	227.77097	65	127.00	159.0	249.25	1898	
numeric	TEAM_FIELDING_DP	286	0.8743409	146.38794	26.22639	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)
```

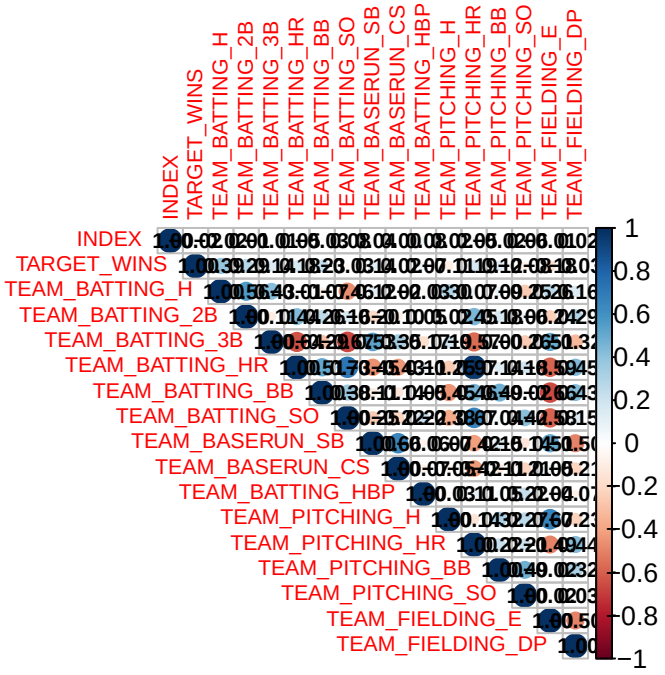
```
# A tibble: 6 x 2
  variable         missing_percentage
  <chr>                         <dbl>
1 TEAM_BATTING_HBP              91.6
2 TEAM_BASERUN_CS               33.9
3 TEAM_FIELDING_DP              12.6
4 TEAM_BASERUN_SB                5.76
5 TEAM_BATTING_SO                4.48
6 TEAM_PITCHING_SO               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) |>
    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_PITCHING_HR TEAM_BATTING_HR        0.969
 3 TEAM_BATTING_HR  TEAM_BATTING_SO        0.727
 4 TEAM_BATTING_SO  TEAM_BATTING_HR        0.727
 5 TEAM_BATTING_3B  TEAM_BATTING_SO       -0.670
 6 TEAM_BATTING_SO  TEAM_BATTING_3B       -0.670
 7 TEAM_PITCHING_H  TEAM_FIELDING_E        0.668
 8 TEAM_FIELDING_E  TEAM_PITCHING_H        0.668
 9 TEAM_BATTING_SO  TEAM_PITCHING_HR       0.667
10 TEAM_PITCHING_HR TEAM_BATTING_SO        0.667
11 TEAM_BATTING_BB  TEAM_FIELDING_E       -0.656
12 TEAM_FIELDING_E  TEAM_BATTING_BB       -0.656
13 TEAM_BASERUN_SB  TEAM_BASERUN_CS        0.655
14 TEAM_BASERUN_CS  TEAM_BASERUN_SB        0.655
15 TEAM_BATTING_3B  TEAM_BATTING_HR       -0.636
16 TEAM_BATTING_HR  TEAM_BATTING_3B       -0.636
17 TEAM_BATTING_HR  TEAM_FIELDING_E       -0.587
18 TEAM_FIELDING_E  TEAM_BATTING_HR       -0.587
19 TEAM_BATTING_SO  TEAM_FIELDING_E       -0.585
20 TEAM_FIELDING_E  TEAM_BATTING_SO       -0.585
```

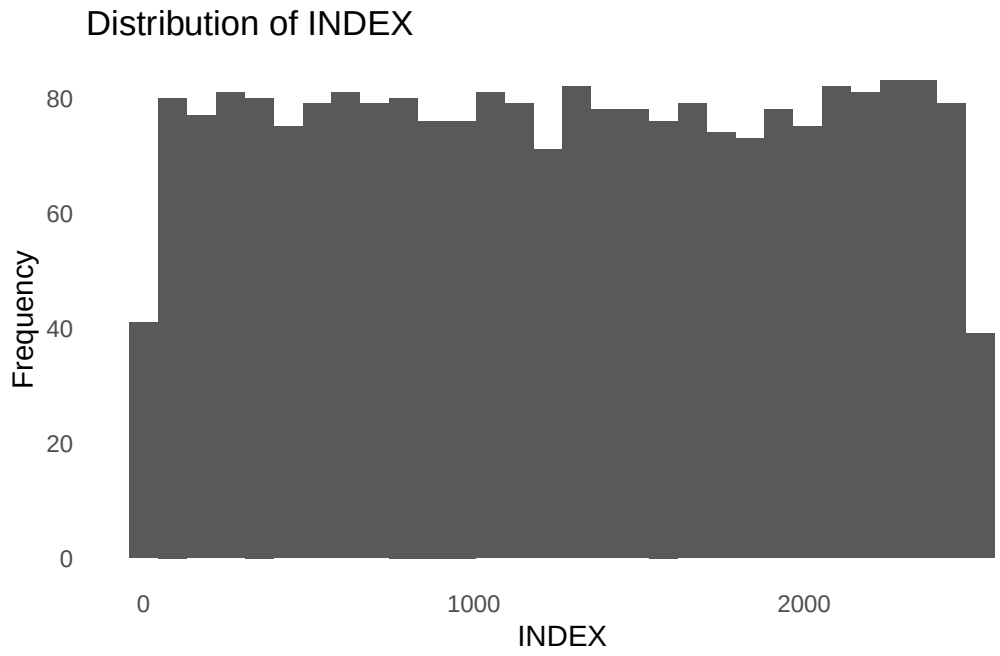
It makes sense that homeruns by batters (`TEAM_BATTING_HR`) and homeruns allowed (`TEAM_PITCHING_HR`) are correlated, as teams that hit many homeruns may also allow many homeruns, for example. We may need to avoid including these pairs of variables in the same model to prevent multicollinearity.

We also visualize the distribution of all variables.

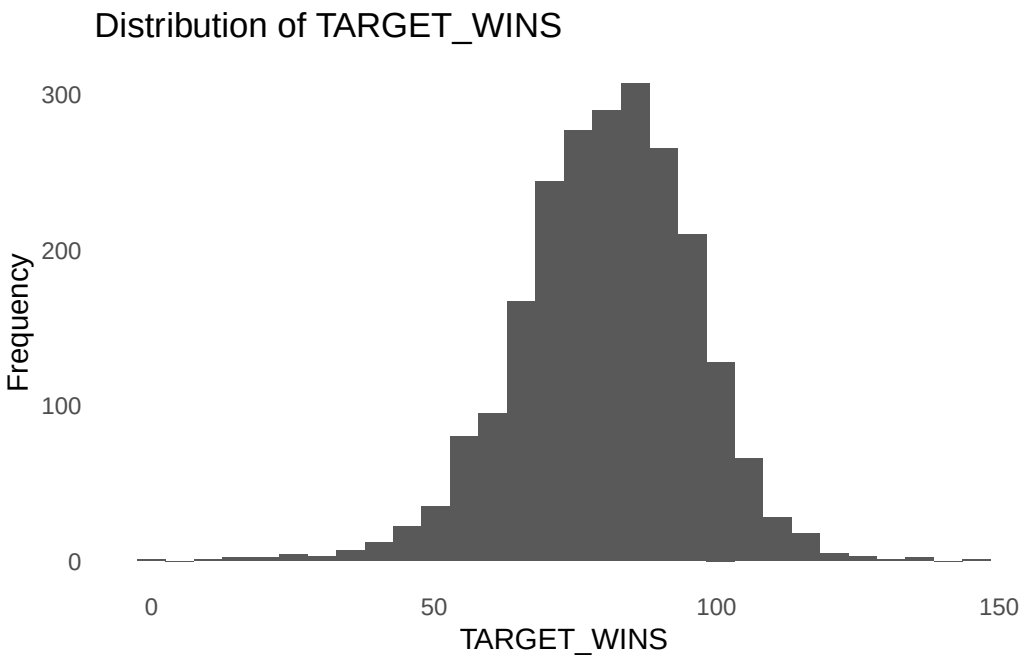
```
# Loop through all variables and create histograms
for (var in names(moneyball_train)) {
```

```
    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)
}
```

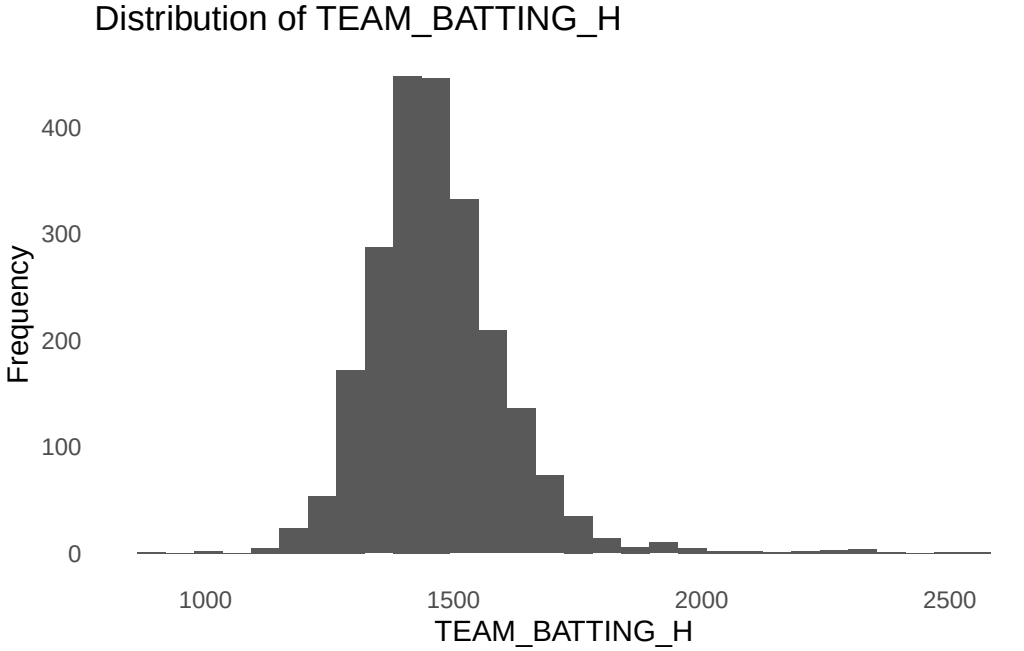
```
`stat_bin()` using `bins = 30`. Pick better value `binwidth`.
```



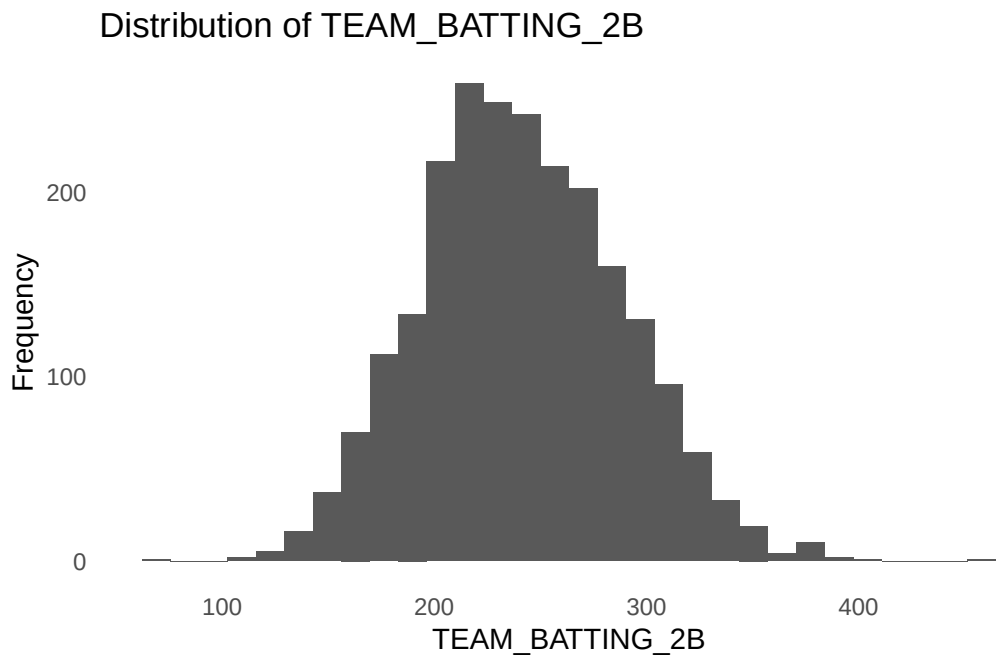

```
`stat_bin()` using `bins = 30`. Pick better value `binwidth`.
```



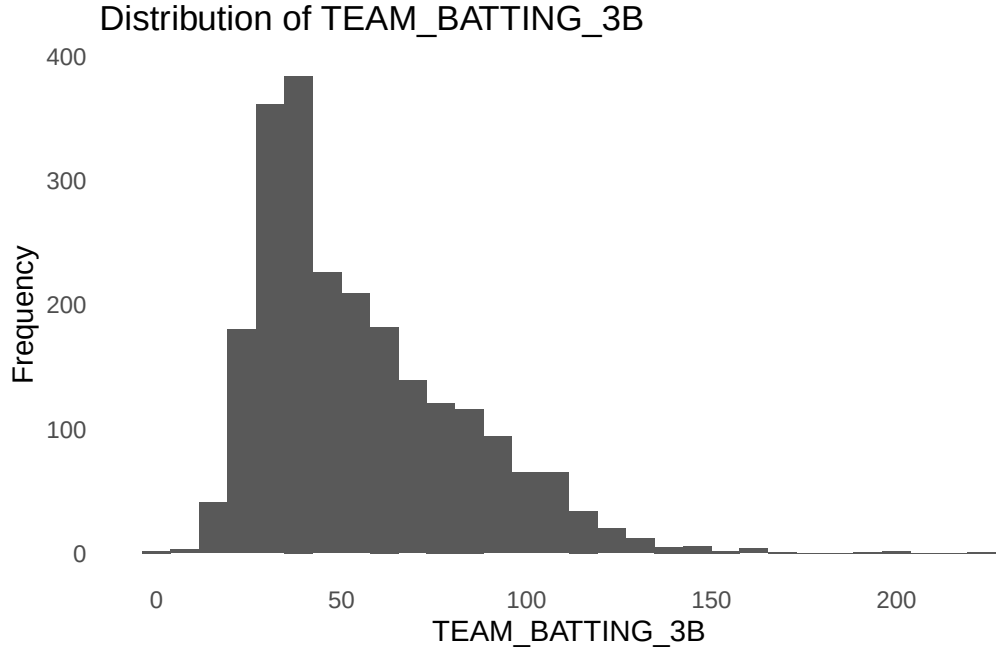

```
`stat_bin()` using `bins = 30`. Pick better value `binwidth`.
```



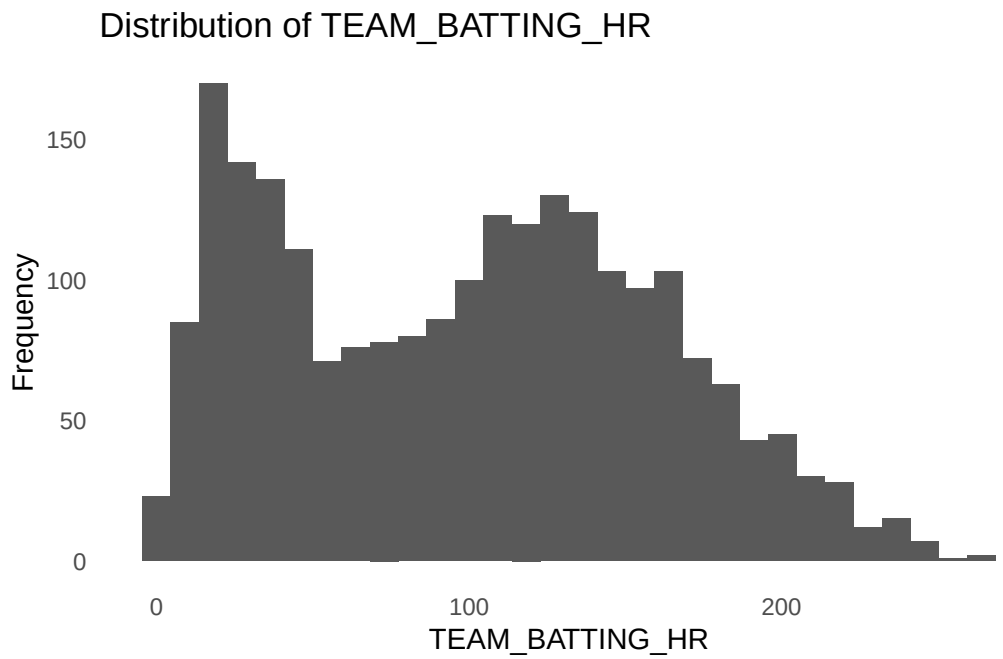

```
`stat_bin()` using `bins = 30`. Pick better value `binwidth`.
```



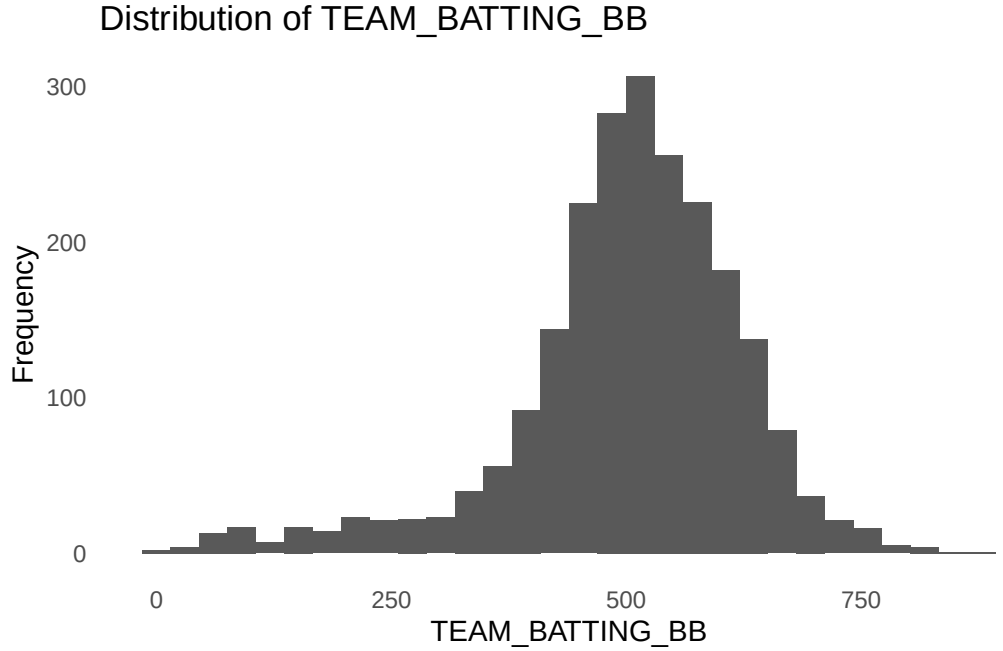

```
`stat_bin()` using `bins = 30`. Pick better value `binwidth`.
```



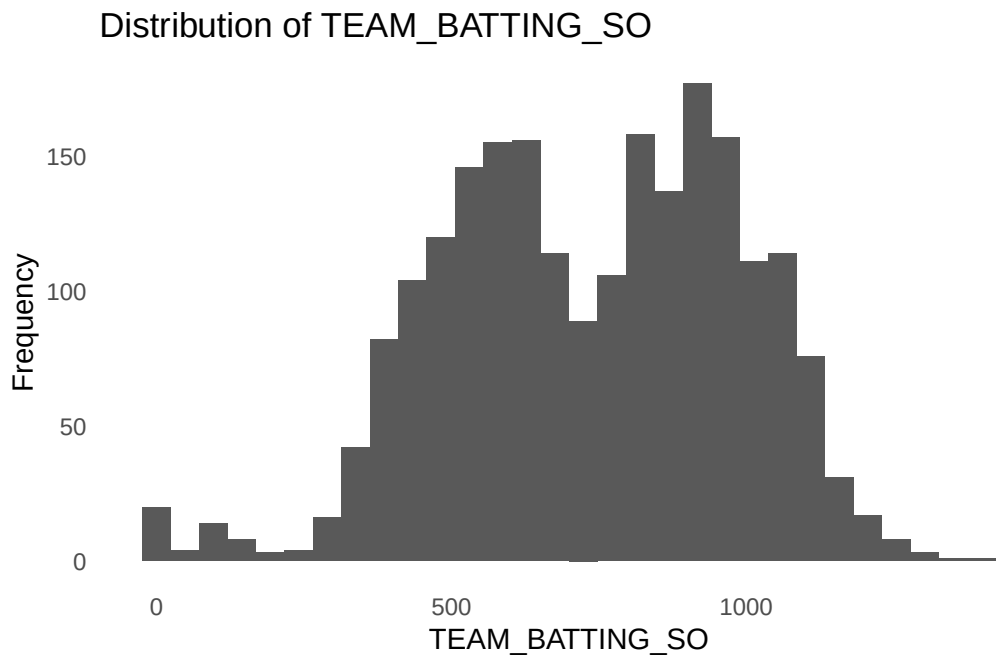

```
`stat_bin()` using `bins = 30`. Pick better value `binwidth`.
```



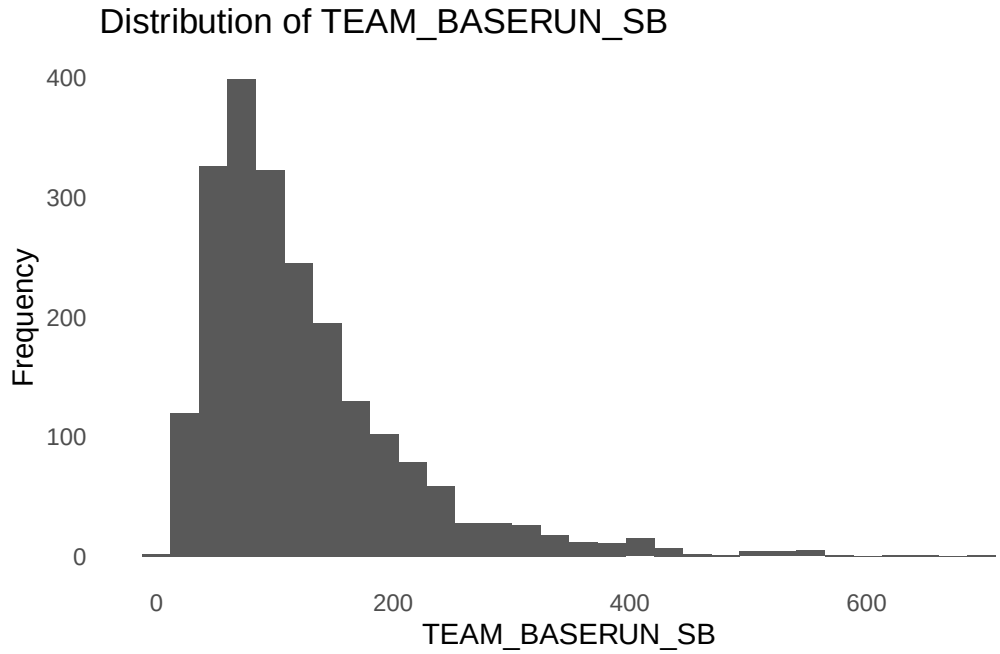

```
`stat_bin()` using `bins = 30`. Pick better value `binwidth`.
```



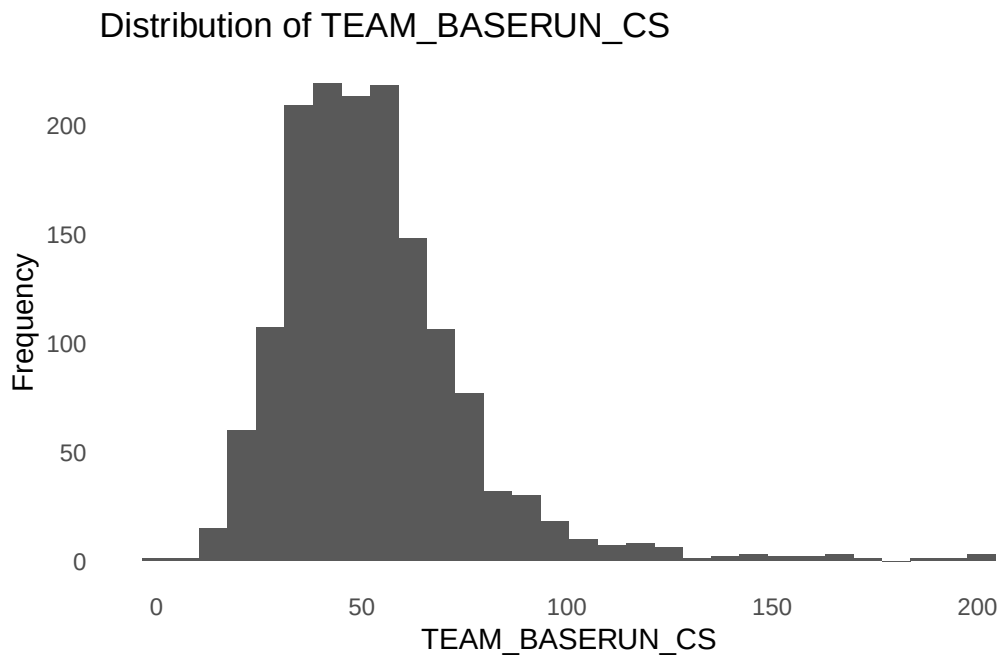

```
`stat_bin()` using `bins = 30`. Pick better value `binwidth`.
```



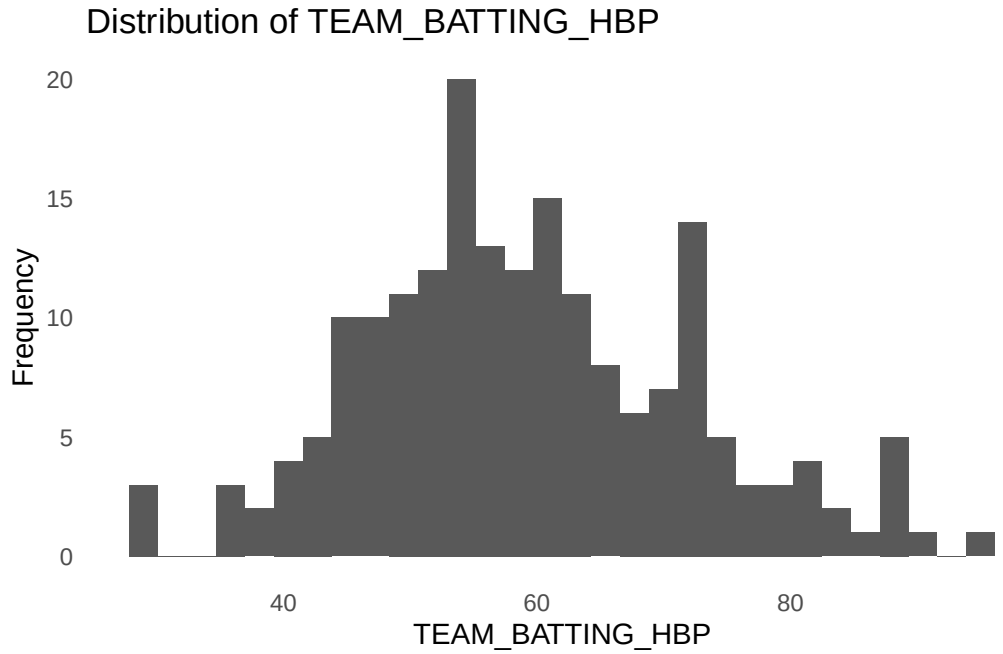

```
`stat_bin()` using `bins = 30`. Pick better value `binwidth`.
```



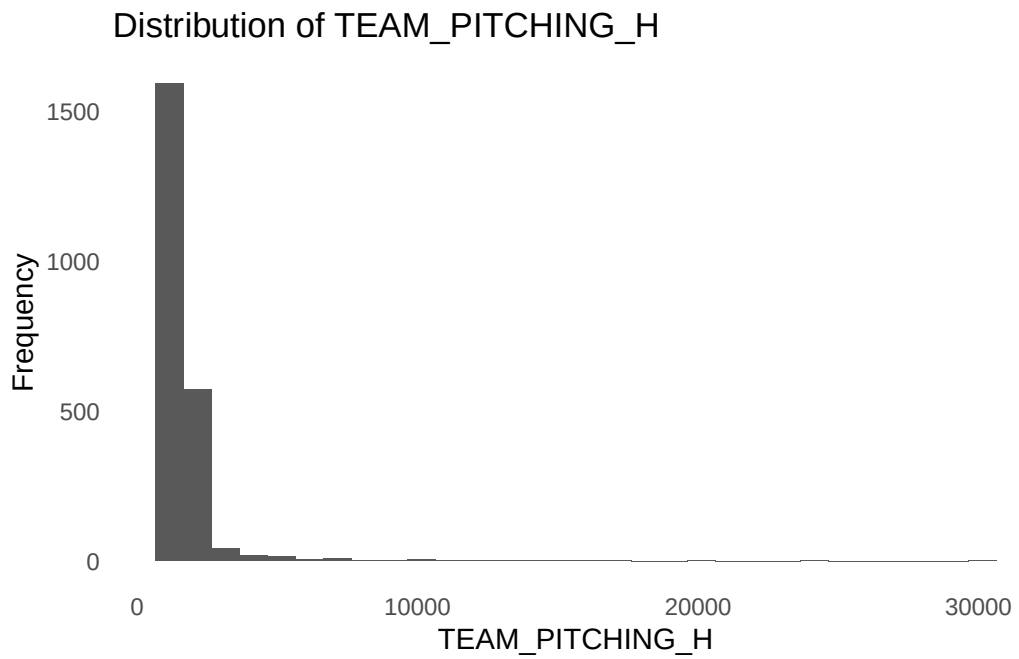

```
`stat_bin()` using `bins = 30`. Pick better value `binwidth`.
```



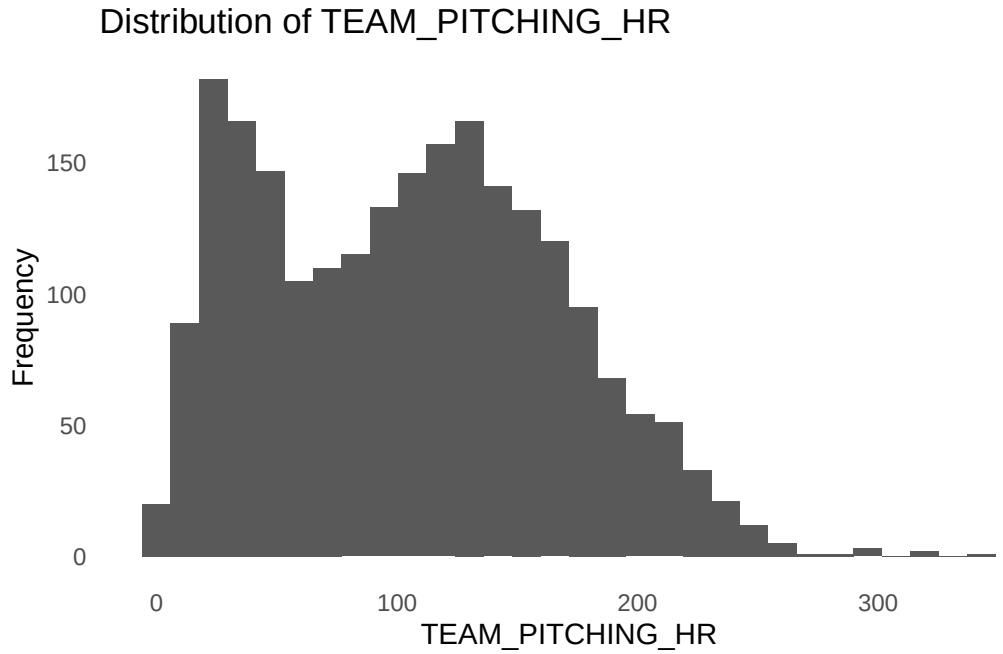

```
`stat_bin()` using `bins = 30`. Pick better value `binwidth`.
```



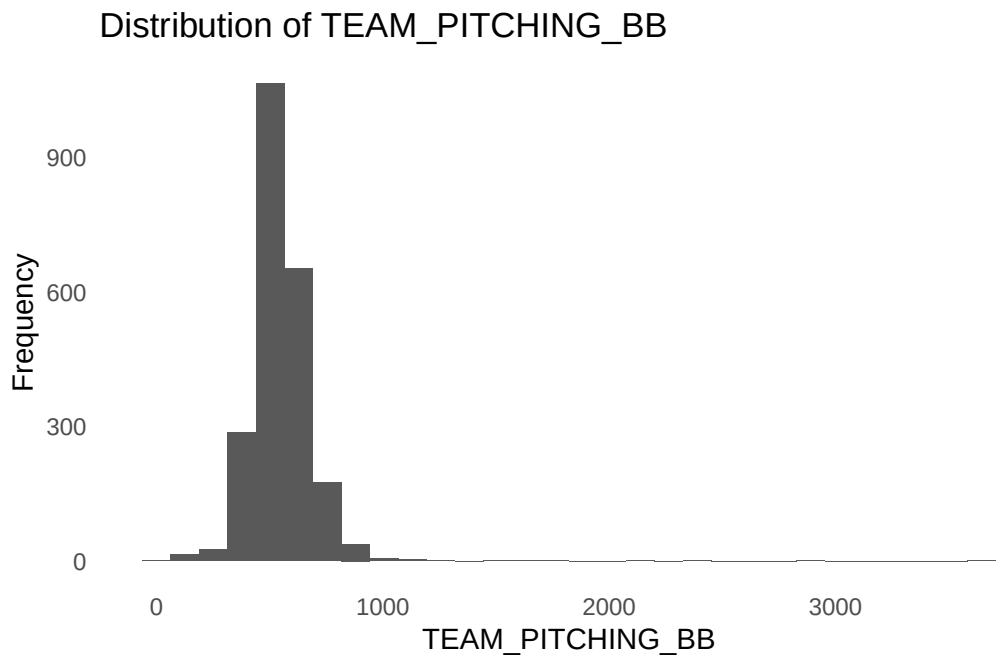

```
`stat_bin()` using `bins = 30`. Pick better value `binwidth`.
```



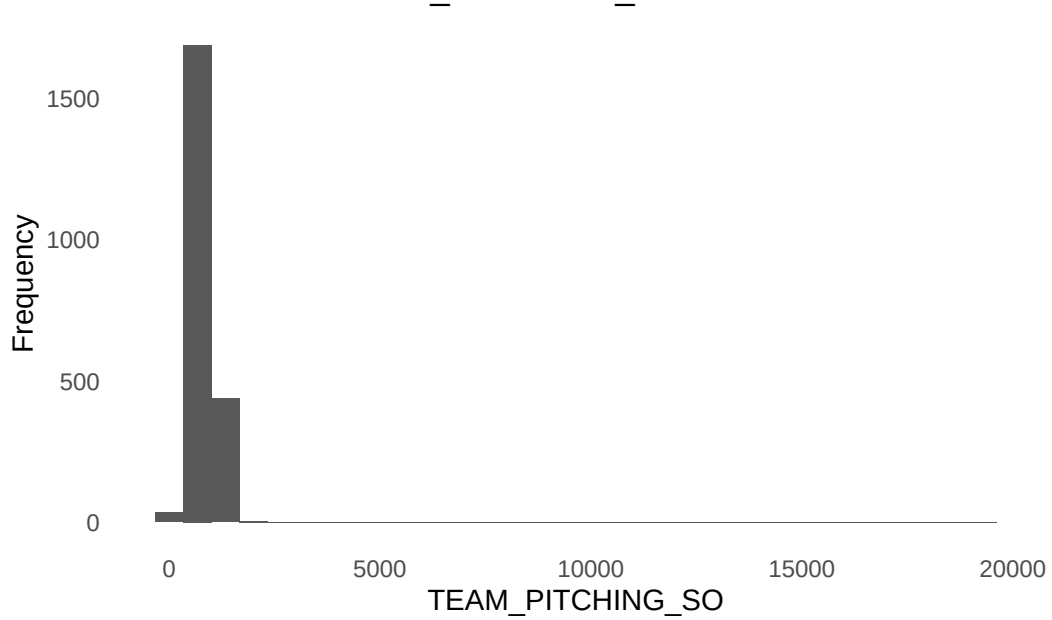

```
`stat_bin()` using `bins = 30`. Pick better value `binwidth`.
```



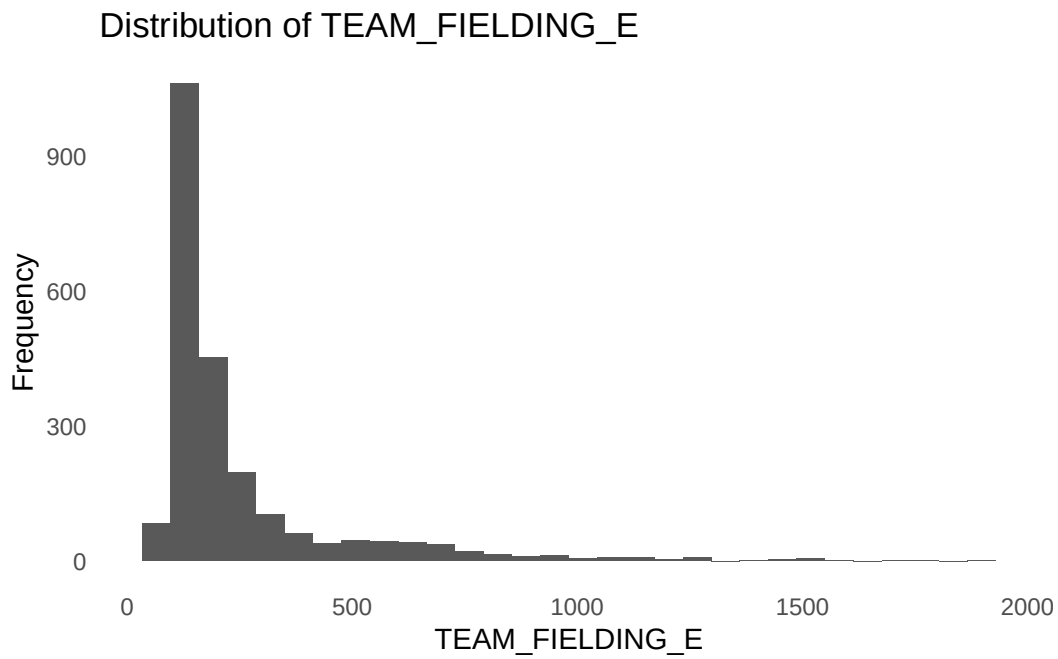

```
`stat_bin()` using `bins = 30`. Pick better value `binwidth`.
```



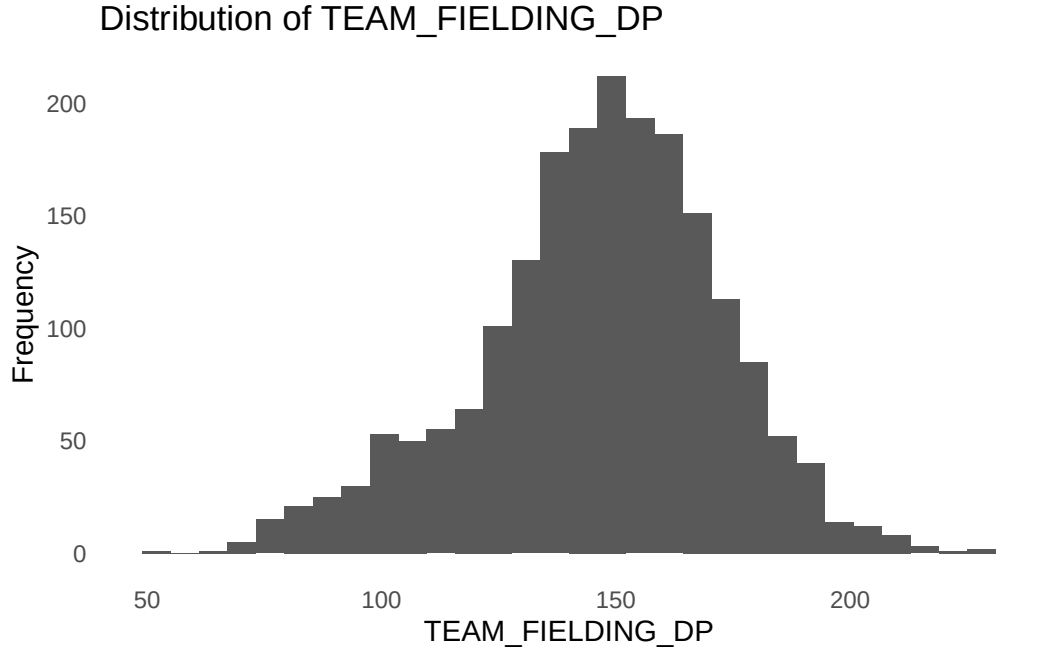

```
`stat_bin()` using `bins = 30`. Pick better value `binwidth`.
```




```
`stat_bin()` using `bins = 30`. Pick better value `binwidth`.
```




```
`stat_bin()` using `bins = 30`. Pick better value `binwidth`.
```



4.1 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), .)))
```

4.2 3. Build Models

4.3 4. Select Models

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