Homework 1

Sam Kuhn

11/20/22

Table of contents

Homework 1: Baseball Analysis

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

Data Exploration:

Load data

```
# library(tidyverse)
# library(here)
# library(tidymodels)
# library(corrplot)
# library(MASS)
# library(gt)

#Install pacman package and load libraries
# install.packages("pacman")
pacman::p_load(tidyverse, here, tidymodels, corrplot, MASS, gt)

#Makes sure dplyr::filter and dplyr::select will be used
conflicted::conflict_prefer("select", "dplyr")
conflicted::conflict_prefer("filter", "dplyr")

#Load training set from data folder and clean variable names
training_set <- readr::read_csv(here::here("data", "moneyball-training-data (1).csv")) |>
  janitor::clean_names()
```

Check for missing values

To check for NA values, we are going to take the sum of every value matching `NA` across the entire data-frame and print the results. Then, replace all the `NA` values with the median value of the corresponding variable. The variables with the most `NA` observations are: `team_batting_hbp`, `team_baserun_cs`, and `team_fielding_dp`.

```r
#Sum NAs across columns
training_set |>
  summarise(across(everything(), ~ sum(is.na(.)))) |>
  glimpse()
```

```
Rows: 1
Columns: 17
$ index            <int> 0
$ target_wins      <int> 0
$ team_batting_h   <int> 0
$ team_batting_2b  <int> 0
$ team_batting_3b  <int> 0
$ team_batting_hr  <int> 0
$ team_batting_bb  <int> 0
$ team_batting_so  <int> 102
$ team_baserun_sb  <int> 131
$ team_baserun_cs  <int> 772
$ team_batting_hbp <int> 2085
$ team_pitching_h  <int> 0
$ team_pitching_hr <int> 0
$ team_pitching_bb <int> 0
$ team_pitching_so <int> 102
$ team_fielding_e  <int> 0
$ team_fielding_dp <int> 286
```

```r
#Replace missing values (NAs) with median values
training_set <- training_set |>
  mutate(across(everything(), ~tidyr::replace_na(., median(., na.rm = TRUE)))) |>
  glimpse()
```

Just as a check, we will print out the data frame again to ensure no `NA` values remain.

```r
#Verify results
#Sum NAs across columns
training_set |>
  summarise(across(everything(), ~ sum(is.na(.)))) |>
  glimpse()
```

```
Rows: 1
Columns: 17
$ index            <int> 0
$ target_wins      <int> 0
$ team_batting_h   <int> 0
$ team_batting_2b  <int> 0
$ team_batting_3b  <int> 0
$ team_batting_hr  <int> 0
$ team_batting_bb  <int> 0
$ team_batting_so  <int> 0
$ team_baserun_sb  <int> 0
$ team_baserun_cs  <int> 0
$ team_batting_hbp <int> 0
$ team_pitching_h  <int> 0
$ team_pitching_hr <int> 0
$ team_pitching_bb <int> 0
$ team_pitching_so <int> 0
$ team_fielding_e  <int> 0
$ team_fielding_dp <int> 0
```

Summary statistics

Now that we do not have any missing values, we can perform some summary statistics to get a better sense of the data. Some key variables are interest are the regressand `target_wins`, where we can see the median value is slightly higher than the mean, suggesting that there is a possible left-tail distribution. Other key variables include: `team_batting_h` which can help predict total runs, and `team_batting_hr` which are homeruns.

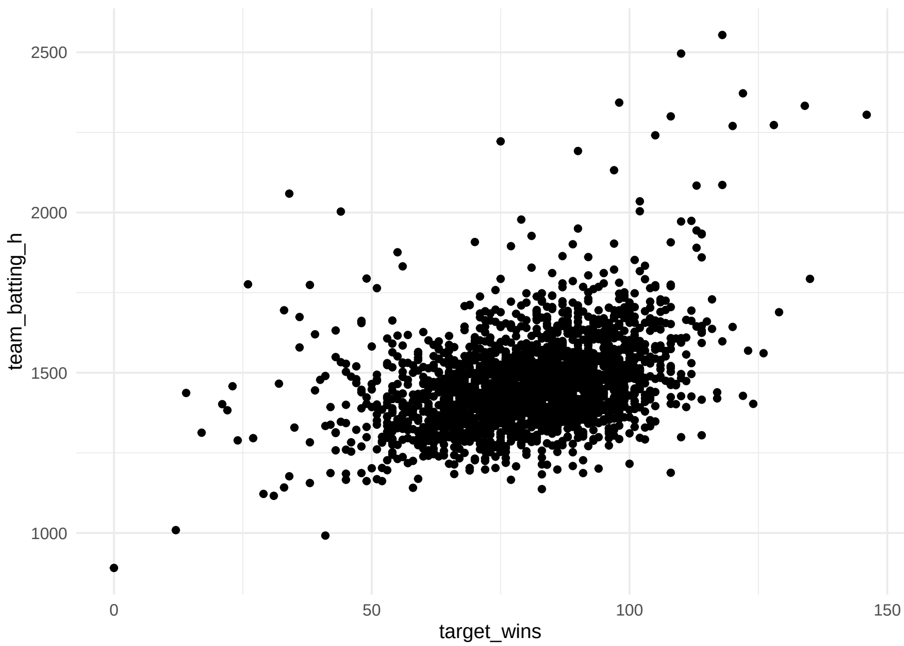
Table 1: Summary Statistics

Variable	Median	Mean	Standard Deviation
target_wins:	82	80.79	15.75215
team_batting_h	1454	1469.27	144.5912
team_batting_2b	238	241.2469	46.80141
team_batting_3b	47	55.25	27.93856
team_batting_hr	102	99.61204	60.54687
team_batting_bb	512	501.5589	122.6709
team_batting_so	750	736.2504	242.9094
team_batting_sb	101	123.3941	85.40565
team_baserun_cs	49	51.51362	18.74587
team_batting_hbp	58	58.1138	3.766219
team_pitching_h	1518	1779.21	1406.843
team_pitching_hr	107	105.6986	61.29875
team_pitching_bb	536.5	553.0079	166.3574
team_pitching_so	813.5	817.5409	540.5447
team_pitching_e	159	246.4807	227.771
team_fielding_dp	149	146.7162	24.53781

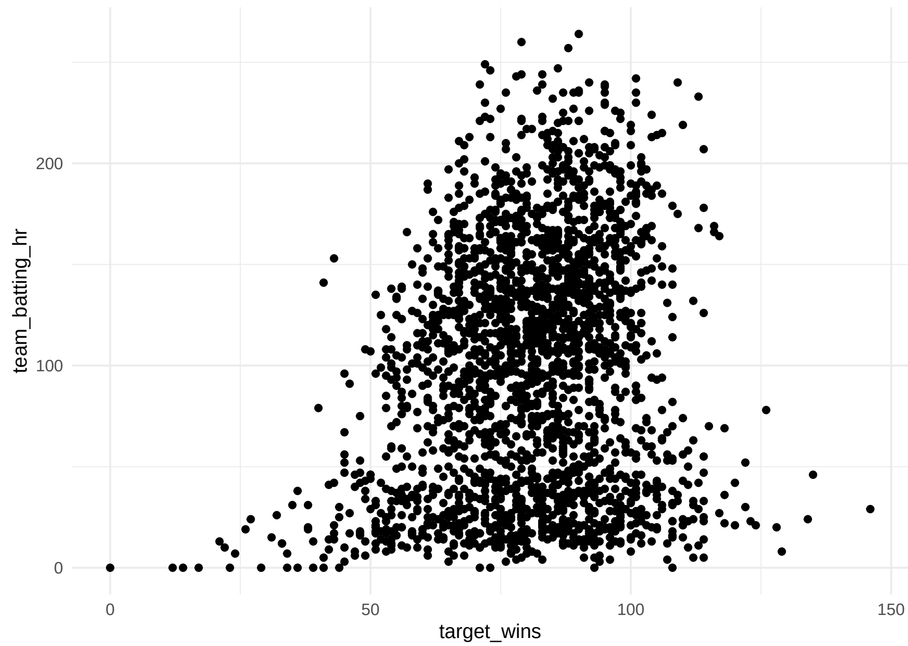
Summary plots

Let's look at some plots to visually inspect the data:

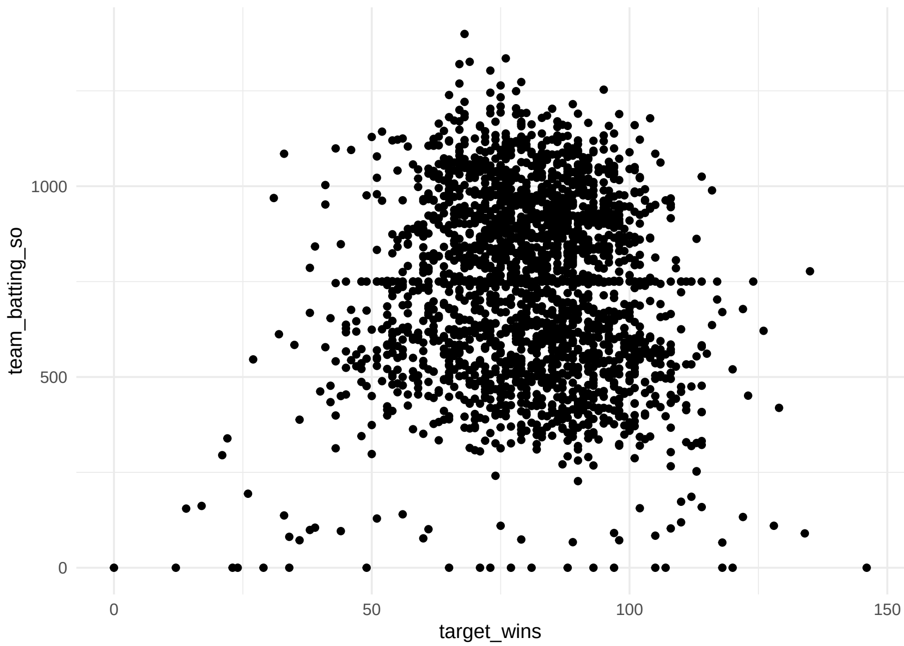
This first plot is ***base hits by batters vs. number of wins***. We can see that most of the observations are centered around 1500 hits, and ~80 wins, with a positive linear relationship.



Let's look at the relationship between ***home runs and wins*** as well. From this plot, it almost has a normal distribution, where the mean is centered around 80 wins, and with a slightly longer left-tail.



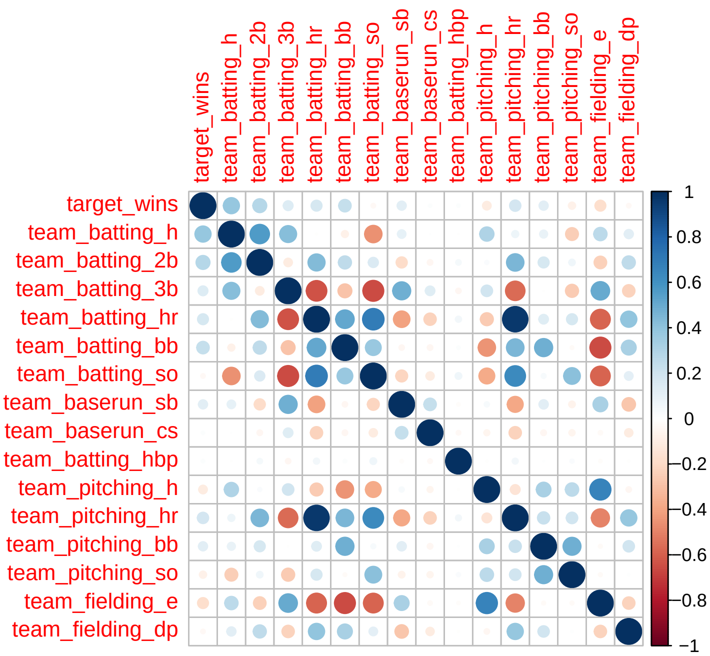
Lastly, let's look at a variable that has a negative impact on wins - ***strikeouts by batters***. This plot looks fairly similar to the plot above, however its shows a pattern that teams win more than 100 games generally don't give up more than 1,000 strikeouts a season.



1000
500
0
0
50
100
150
team_batting_so
target_wins

Correlation Plot

Now that we have a good idea about the distribution of our key variables, let's check the statistical correlation between all the variables and `target_wins`, to understand how each variable is impact it.



Correlation between variables and Target Wins

Pearson correlation

Target Wins	Variable	Correlation
1	team_batting_h	0.38876752
1	team_batting_2b	0.28910365
1	team_batting_bb	0.23255986
1	team_pitching_hr	0.18901373
1	team_batting_hr	0.17615320
1	team_batting_3b	0.14260841
1	team_pitching_bb	0.12417454
1	team_baserun_sb	0.12361087
1	team_batting_hbp	0.01651641
1	team_baserun_cs	0.01595982
1	index	-0.02105643
1	team_fielding_dp	-0.03008630
1	team_batting_so	-0.03058135
1	team_pitching_so	-0.07579967

1	team_pitching_h	-0.10993705
1	team_fielding_e	-0.17648476

Data Preparation

Build Models

Select Models

Appendix: R code

Summary statistics

```
##Median
training_set |>
  dplyr::select(-index) |>
  summarise(across(everything(), ~ median(.))) |>
  glimpse()

##Mean
training_set |>
  dplyr::select(-index) |>
  summarise(across(everything(), ~ mean(.))) |>
  glimpse()

##Standard Deviation
training_set |>
  dplyr::select(-index) |>
  summarise(across(everything(), ~ sd(.))) |>
  glimpse()
```

Plots

```
#Create named character vector of variables
vars <- training_set |>
  #select(-index) |>
  names() |>
  set_names()

#Use map function to create a sequence of plots
plots <-  map(vars, ~ggplot(data = training_set) +
              geom_point(aes(x = target_wins, y = .data[[.x]]) ) +
              theme_minimal() +
              labs(y = .x)
)

#Correlation matrix
cor_matrix <- training_set |>
  dplyr::select(-index) |>
  cor() |>
  as.matrix()

corrplot(cor_matrix)
```

```
#Get correlation values as a table, sorted highest to lowest
purrr::map_df(vars, ~cor(training_set$target_wins, training_set[[.x]])) |>
  pivot_longer(cols = !c("target_wins"), names_to = "correlation") |>
  arrange(desc(value)) |>
  gt::gt()
```