Task_1

Intro

Importing the data set from github repo and loading libraries

```
library(readr)
```

```
Warning: package 'readr' was built under R version 4.2.3
```

```
library(ggplot2)
```

```
Warning: package 'ggplot2' was built under R version 4.2.3
```

```
library(reshape2)
library(corrplot)
```

```
corrplot 0.94 loaded
```

```
library(dplyr)
```

```
Warning: package 'dplyr' was built under R version 4.2.3
```

```

Attaching package: 'dplyr'

The following objects are masked from 'package:stats':

    filter, lag
```

```
The following objects are masked from 'package:base':

    intersect, setdiff, setequal, union
```

```
library(tidyr)
```

```
Warning: package 'tidyr' was built under R version 4.2.3
```

```
Attaching package: 'tidyr'

The following object is masked from 'package:reshape2':

    smiths
```

```
library(ranger)
```

```
Warning: package 'ranger' was built under R version 4.2.3
```

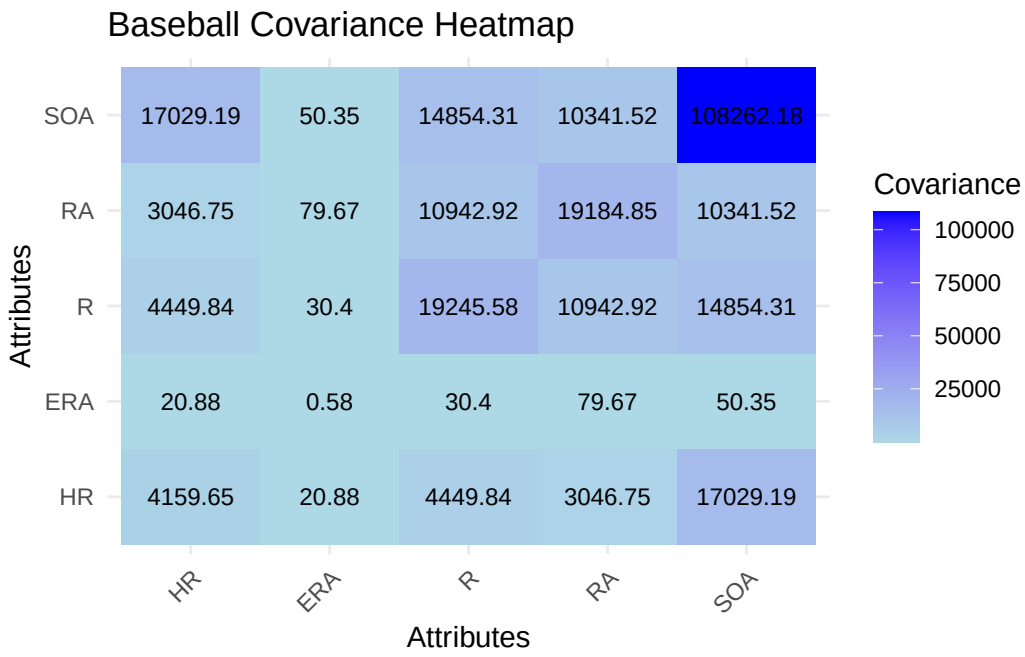
```
url = "https://raw.githubusercontent.com/RaunakDune/3337_Fall2024_EDA/bf7b9ca17ed5e08ff25677

data = read.csv(url)
head(data, 3)
attach(data)
```

1. Covariance Matrices

Covariance matrices for HR, ERA, R, RA, SOA

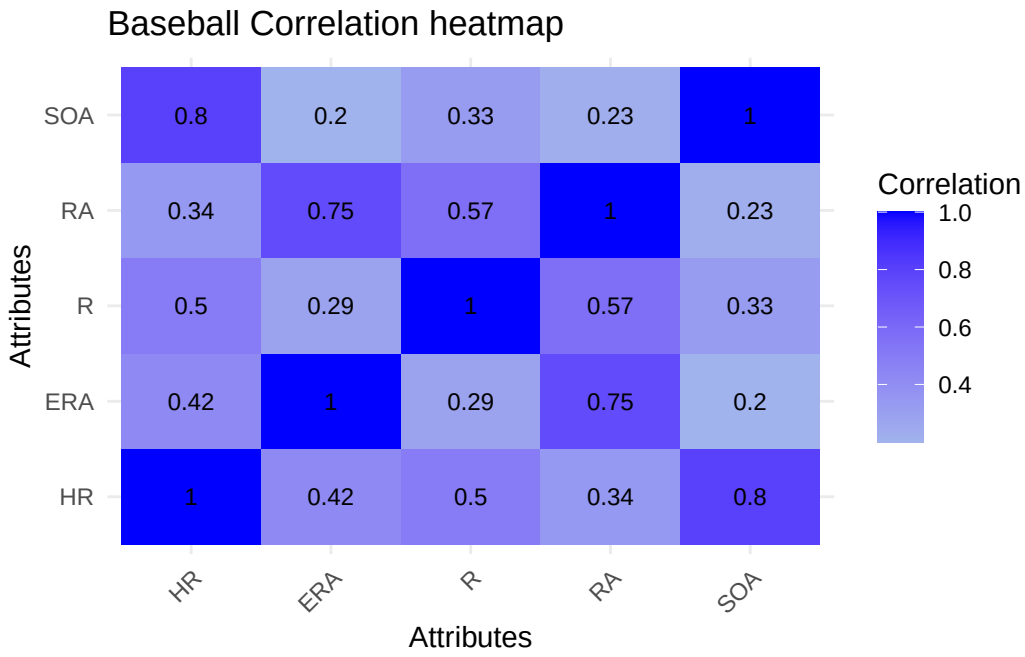
```
selected_data = data[, c("HR", "ERA","R", "RA", "SOA")]
covariance_matrix = cov(selected_data)

cov_melted = melt(covariance_matrix)

ggplot(cov_melted, aes(Var1, Var2, fill = value)) +
  geom_tile() +
  scale_fill_gradient2(low = "white", mid = "lightblue", high = "blue",
                       midpoint = 0, limit = c(min(cov_melted$value), max(cov_melted$value))
```

```
  theme_minimal() +
  labs(title = "Baseball Covariance Heatmap", x = "Attributes", y = "Attributes", fill = "Cov
  theme(axis.text.x = element_text(angle = 45, hjust = 1)) +
  geom_text(aes(label = round(value, 2)), color = "black", size = 3)
```



1. Correlation Matrices

```
correlation_matrix = cor(selected_data)

cor_melted = melt(correlation_matrix)

ggplot(cor_melted, aes(Var1, Var2, fill = value)) +
  geom_tile() +
  scale_fill_gradient2(low = "white", mid = "lightblue", high = "blue",
                       limit = c(min(cor_melted$value), max(cor_melted$value))) +
  theme_minimal() +
  labs(title = "Baseball Correlation heatmap", x = "Attributes", y = "Attributes", fill = "Co
  theme(axis.text.x = element_text(angle = 45, hjust = 1)) +
  geom_text(aes(label = round(value, 2)), color = "black", size = 3)
```



1. Interpretation of Correlation and Covariance Matrices

SOA has a correlation of .80 with HR, suggesting that teams that strike out a lot of batters, also hit a lot of home runs. This could be because the strike outs are due to good pitchers, the same pitchers that batters will be practicing with, iron sharpening iron.

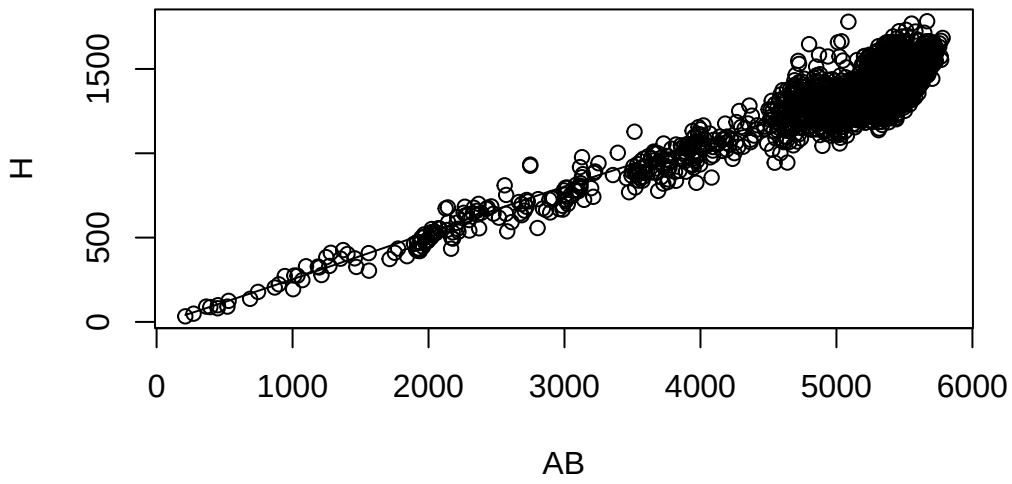
The next highest correlation is 0.75 between ERA and RA. This is expected since ERA is a measure of a pitcher's effectiveness. If the pitcher is less effective at striking batters out, meaning the pitcher has a higher ERA, then there will be more runs scored, and a higher RA.

Everything else is roughly 0.50 or less, and can be explained by domain knowledge.

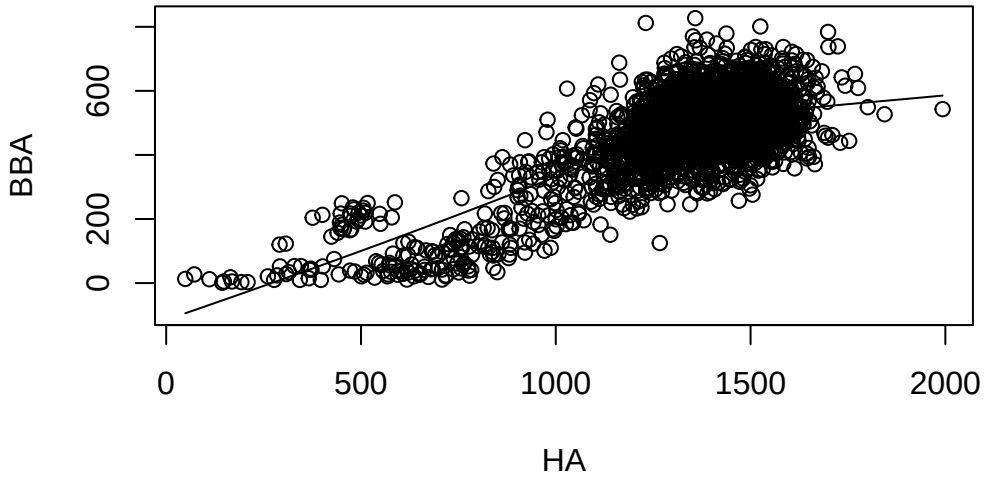
2. Scatter Plot for AB/H and HA/BBA

AB = At bats, H = hits by batters, HA = hits allows, BBA = walks allowed

```
scatter.smooth(AB, H)
```




```
scatter.smooth(HA, BBA)
```



At bats is a measure of how many times a team has had a batter at the plate. Hits tells us

how many times a batter got a hit AND got to first base. If a batter hit the ball, and it was foul, it is not considered a hit. Only hits that result in placement on base count.

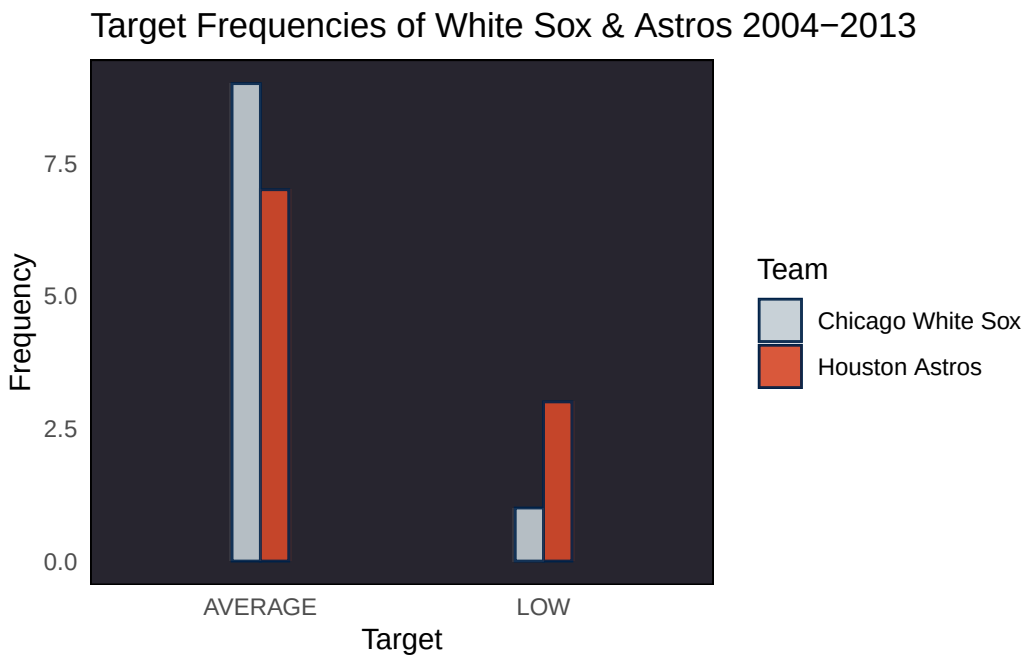
Based on our AB/H plot, there is a strong positive correlation between the two, which is to be expected. There seems to be more variability in clusters around 2000, 4000, and 4700+ at bats.

Walks allowed are recorded when a pitcher either hits a batter with the ball, or throws 4 balls outside of the strike zone without the batter swinging at them. Hits allowed is the offensive component of Hits. There seems to be a lot of variation between these two variables. I can't think of why right now.

3. Two Teams + Astros Histograms

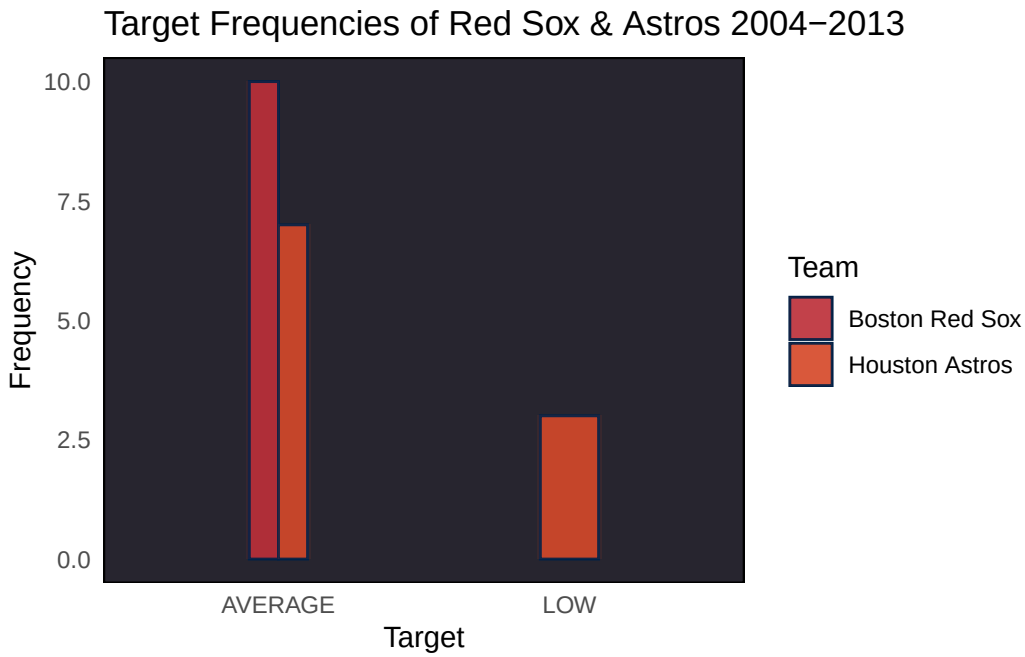
I will use the White Sox, Red Sox, and Astros. The histograms will be for the years 2004-2013 and 2014-2023. I will measure the High, Low, and Average.

```
team_hist_data_2013_Boston = data %>%
  filter(yearID <= 2013 & yearID >= 2004 & (name == "Boston Red Sox" | name == "Houston Astro
  select( name, TARGET)


team_hist_data_2013_Chicago = data %>%
  filter(yearID <= 2013 & yearID >= 2004 & (name == "Chicago White Sox" | name == "Houston As
  select(name, TARGET)


ggplot(team_hist_data_2013_Chicago, aes(x = TARGET, fill = name)) +
  geom_bar(position = "dodge", color = "#002147", width = 0.2, alpha = 0.9) +
  scale_fill_manual(values = c("Chicago White Sox" = "#C4CED4", "Houston Astros" = "#D6492A")
  labs(x = "Target", y = "Frequency", title = "Target Frequencies of White Sox & Astros 2004-
  labs(fill = "Team") +
  theme_minimal() +
    theme(
    panel.grid.major = element_blank(),
    panel.grid.minor = element_blank(),
    panel.background = element_rect(fill = "#27252F")
    ) +
    geom_bar(aes(x = TARGET, fill = name), position = "dodge", width = 0.22, alpha = 0.1)
```



The White Sox had more average targets than the Astros and few low targets as well.

```
ggplot(team_hist_data_2013_Boston, aes(x = TARGET, fill = name)) +
  geom_bar(position = "dodge", color = "#002147", width = 0.2, alpha = 0.9) +
  scale_fill_manual(values = c("Boston Red Sox" = "#BD3039", "Houston Astros" = "#D6492A")) +
  labs(x = "Target", y = "Frequency", title = "Target Frequencies of Red Sox & Astros 2004-20
  labs(fill = "Team") +
  theme_minimal() +
    theme(
    panel.grid.major = element_blank(),
    panel.grid.minor = element_blank(),
    panel.background = element_rect(fill = "#27252F")
    ) +
    geom_bar(aes(x = TARGET, fill = name), position = "dodge", width = 0.22, alpha = 0.1)
```



The Red Sox had no low season targets within this 10 year period, while the astros had mixed seasons.

4. Box plots for BB and SB

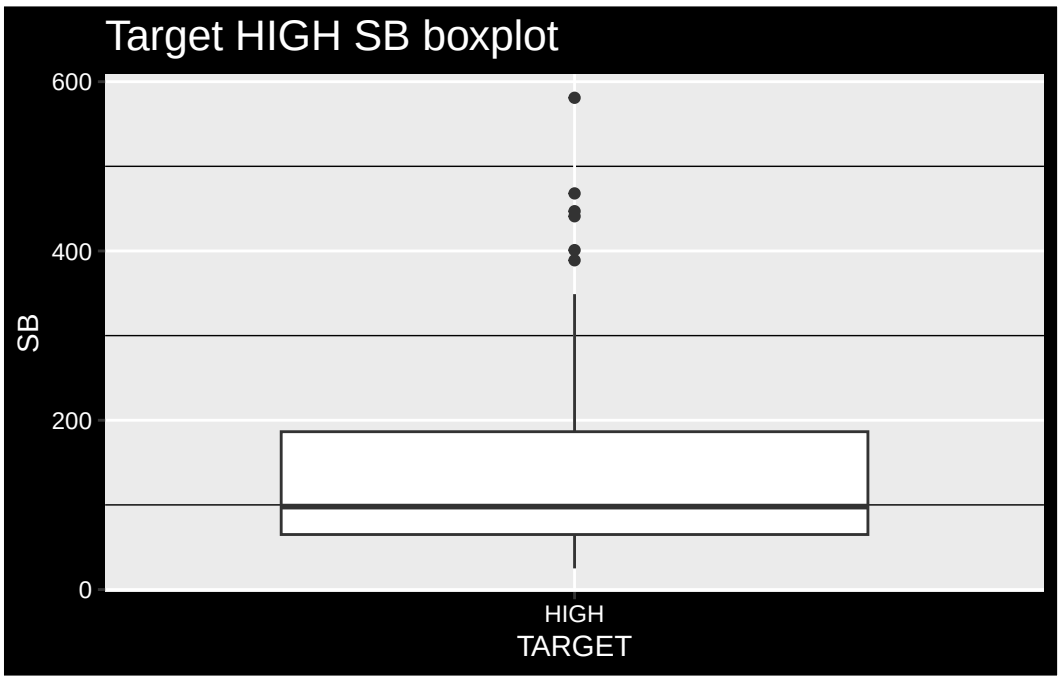
I will create box plots for BB and SB, each with TARGET ave, low, hi, and then each with the whole data set.

BB is walks by batters, while SB is stolen bases.

```
box_sb_hi = data %>%
  filter(TARGET == "HIGH") %>%
  select(TARGET, SB)

box_sb_ave = data %>%
  filter(TARGET == "AVERAGE") %>%
  select(TARGET, SB)

box_sb_low = data %>%
  filter(TARGET == "LOW") %>%
  select(TARGET, SB)
```

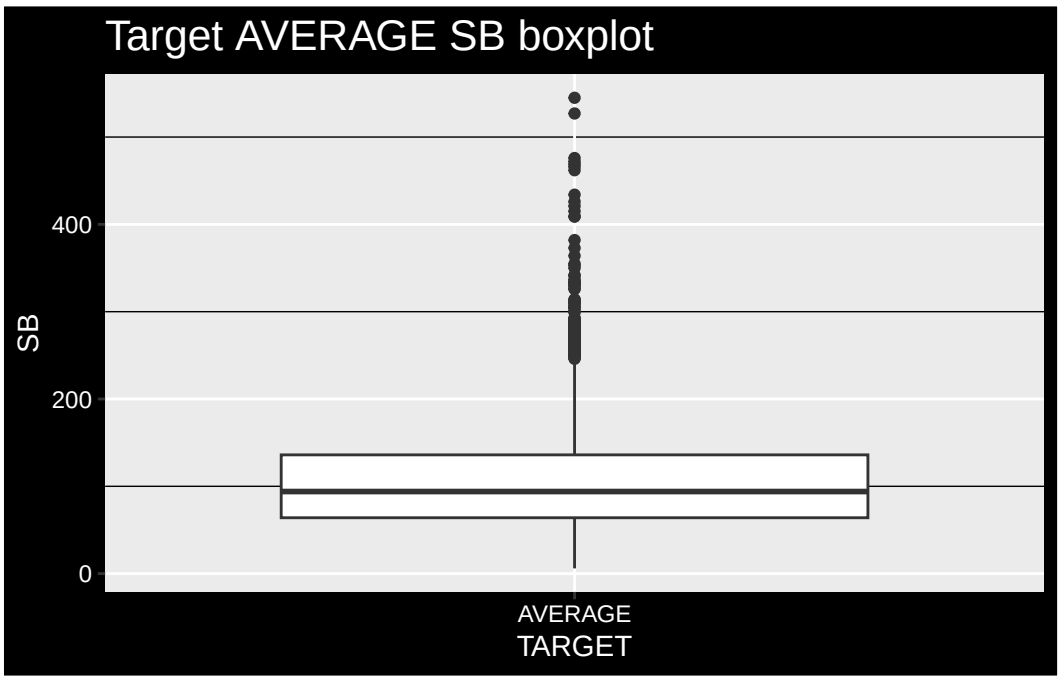
```r
box_bb_hi = data %>%
  filter(TARGET == "HIGH") %>%
  select(TARGET, BB)

box_bb_ave = data %>%
  filter(TARGET == "AVERAGE") %>%
  select(TARGET, BB)

box_bb_low = data %>%
  filter(TARGET == "LOW") %>%
  select(TARGET, BB)


box_bb_all = data %>%
  select(TARGET, BB) %>%
  arrange(TARGET)

box_sb_all = data %>%
  select(TARGET, SB) %>%
  arrange(TARGET)


ggplot(box_sb_hi, aes(x = TARGET, y = SB)) +
  geom_boxplot() + labs(title ="Target HIGH SB boxplot") +
  theme(
    panel.grid.minor = element_line(color = "black"),
    plot.background = element_rect(fill = "black"),
    axis.text = element_text(color = "white"),
    axis.title = element_text(color = "white"),
    plot.title = element_text(color = "white", size = 16),
    legend.text = element_text(color = "white")
  )
```

```
Warning: Removed 24 rows containing non-finite outside the scale range
(`stat_boxplot()`).
```



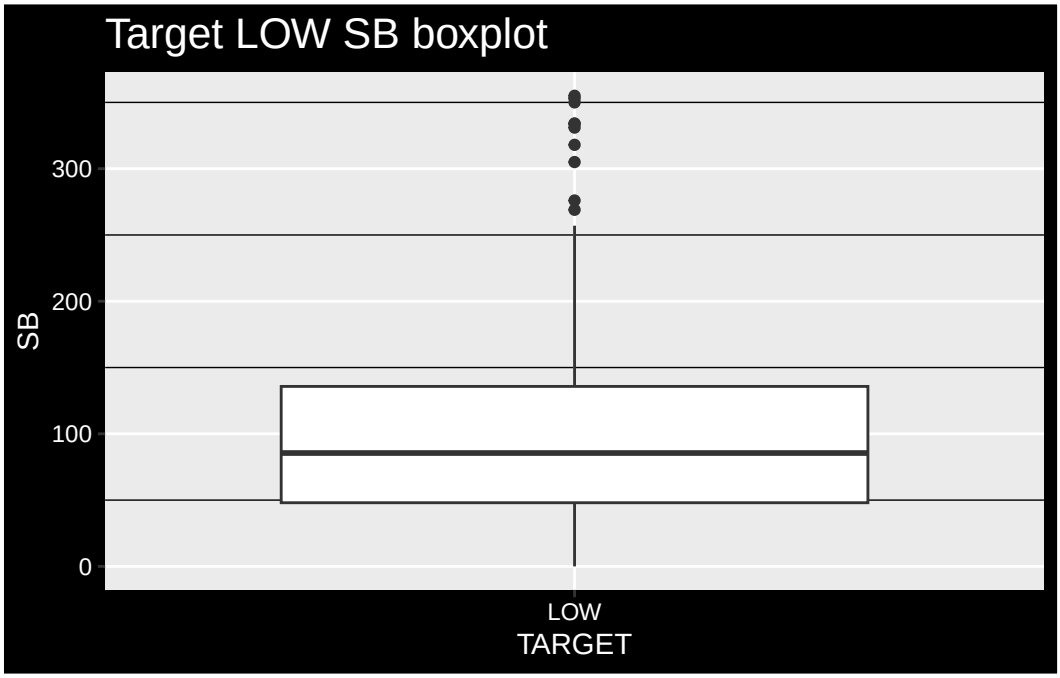

```
ggplot(box_sb_ave, aes(x = TARGET, y = SB)) +
  geom_boxplot() +
  labs(title ="Target AVERAGE SB boxplot") +
  theme(
    panel.grid.minor = element_line(color = "black"),
    plot.background = element_rect(fill = "black"),
    axis.text = element_text(color = "white"),
    axis.title = element_text(color = "white"),
    plot.title = element_text(color = "white", size = 16),
    legend.text = element_text(color = "white")
  )
```

```
Warning: Removed 64 rows containing non-finite outside the scale range
(`stat_boxplot()`).
```

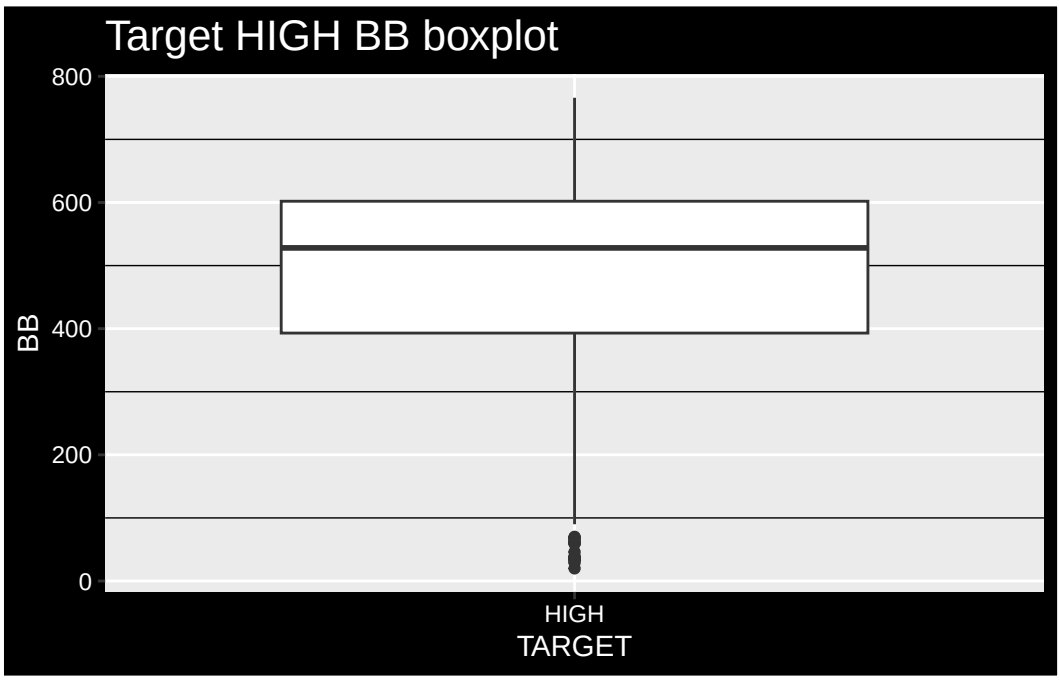



```
ggplot(box_sb_low, aes(x = TARGET, y = SB)) +
  geom_boxplot() +
  labs(title ="Target LOW SB boxplot") +
  theme(
    panel.grid.minor = element_line(color = "black"),
    plot.background = element_rect(fill = "black"),
    axis.text = element_text(color = "white"),
    axis.title = element_text(color = "white"),
    plot.title = element_text(color = "white", size = 16),
    legend.text = element_text(color = "white")
  )
```

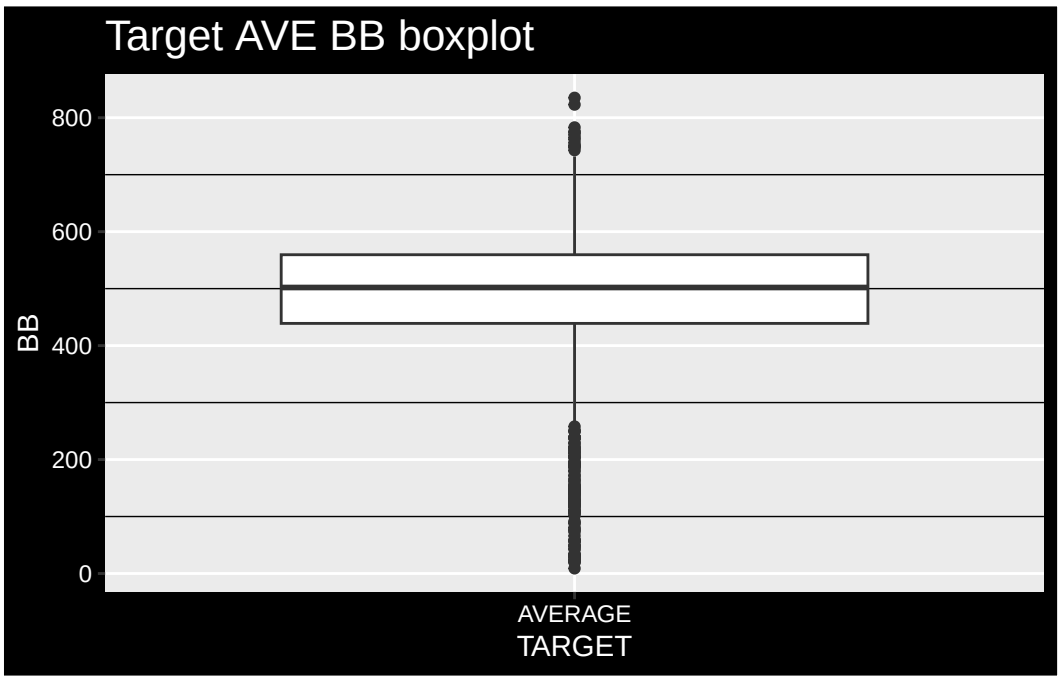
```
Warning: Removed 37 rows containing non-finite outside the scale range
(`stat_boxplot()`).
```



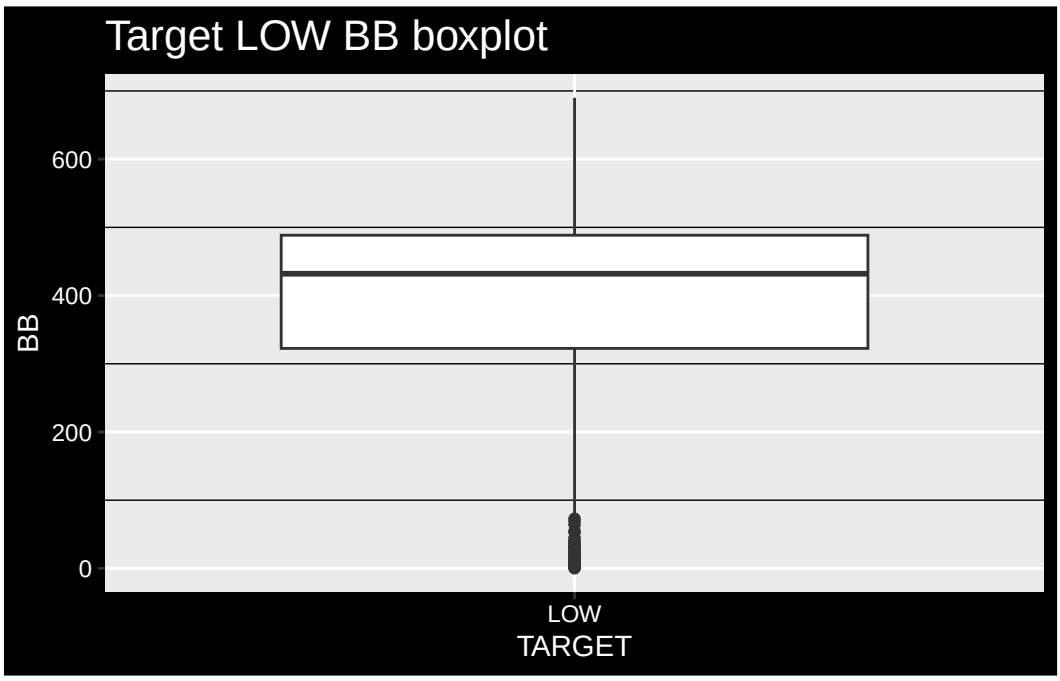

```
ggplot(box_bb_hi, aes(x = TARGET, y = BB)) +
  geom_boxplot() + labs(title ="Target HIGH BB boxplot") +
  theme(
    panel.grid.minor = element_line(color = "black"),
    plot.background = element_rect(fill = "black"),
    axis.text = element_text(color = "white"),
    axis.title = element_text(color = "white"),
    plot.title = element_text(color = "white", size = 16),
    legend.text = element_text(color = "white")
  )
```



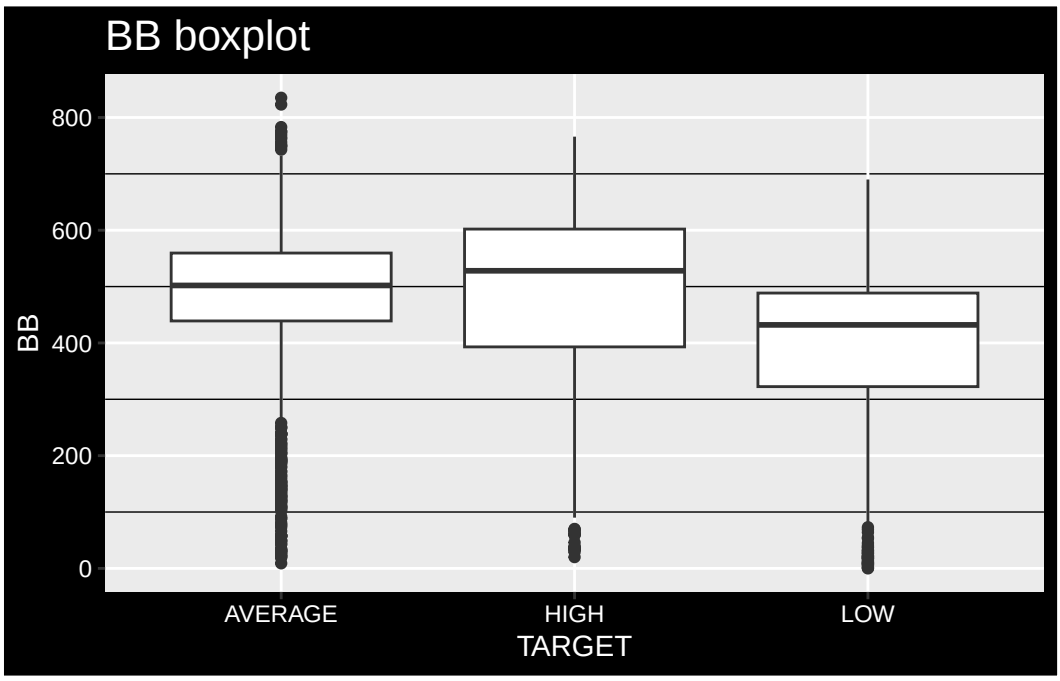

```
ggplot(box_bb_ave, aes(x = TARGET, y = BB)) +
  geom_boxplot() + labs(title ="Target AVE BB boxplot") +
  theme(
    panel.grid.minor = element_line(color = "black"),
    plot.background = element_rect(fill = "black"),
    axis.text = element_text(color = "white"),
    axis.title = element_text(color = "white"),
    plot.title = element_text(color = "white", size = 16),
    legend.text = element_text(color = "white")
  )
```




```
ggplot(box_bb_low, aes(x = TARGET, y = BB)) +
  geom_boxplot() + labs(title ="Target LOW BB boxplot") +
  theme(
    panel.grid.minor = element_line(color = "black"),
    plot.background = element_rect(fill = "black"),
    axis.text = element_text(color = "white"),
    axis.title = element_text(color = "white"),
    plot.title = element_text(color = "white", size = 16),
    legend.text = element_text(color = "white")
  )
```

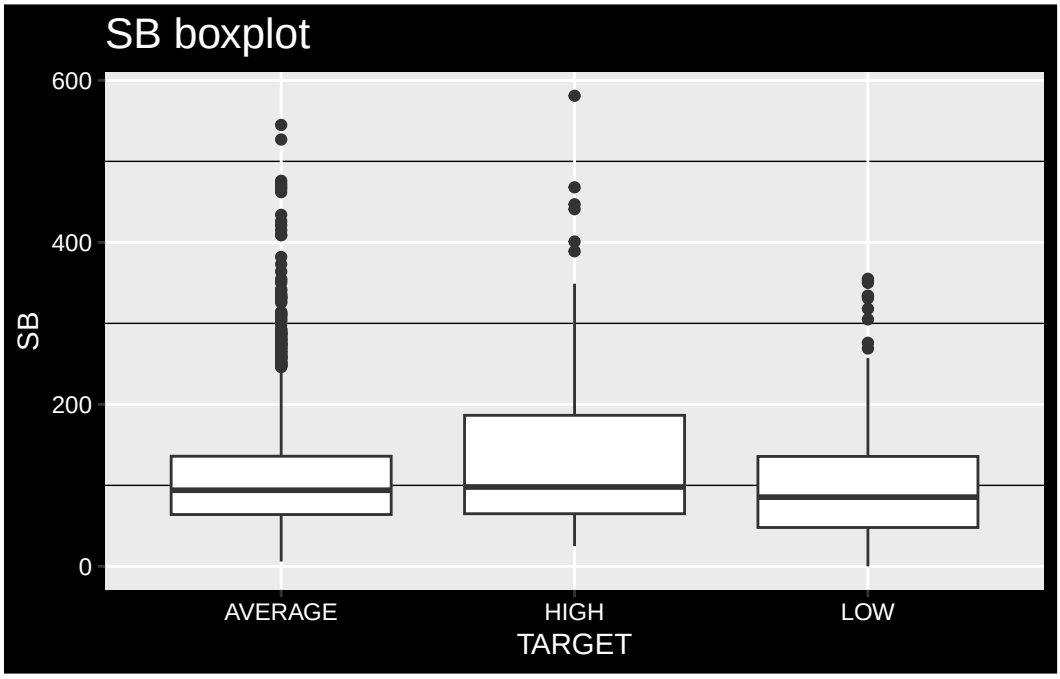



```
ggplot(box_bb_all, aes(x = TARGET, y = BB)) +
  geom_boxplot() + labs(title ="BB boxplot") +
  theme(
    panel.grid.minor = element_line(color = "black"),
    plot.background = element_rect(fill = "black"),
    axis.text = element_text(color = "white"),
    axis.title = element_text(color = "white"),
    plot.title = element_text(color = "white", size = 16),
    legend.text = element_text(color = "white")
  )
```




```
ggplot(box_sb_all, aes(x = TARGET, y = SB)) +
  geom_boxplot() + labs(title ="SB boxplot") +
  theme(
    panel.grid.minor = element_line(color = "black"),
    plot.background = element_rect(fill = "black"),
    axis.text = element_text(color = "white"),
    axis.title = element_text(color = "white"),
    plot.title = element_text(color = "white", size = 16),
    legend.text = element_text(color = "white")
  )
```

```
Warning: Removed 125 rows containing non-finite outside the scale range
(`stat_boxplot()`).
```



For seasons with the HIGH target, stolen bases seems to reach its median at roughly 100 stolen bases, which is slightly above the AVERAGE target stolen bases median. The average target seasons seem to have a much tighter distribution and far more outliers than the others.

The average target seasons seemed to have a very tight distribution with a median barely above 500. The low target seasons had a median of 425 and the high 525. The latter two also had much wider distributions. There are very few outliers for the high target group.

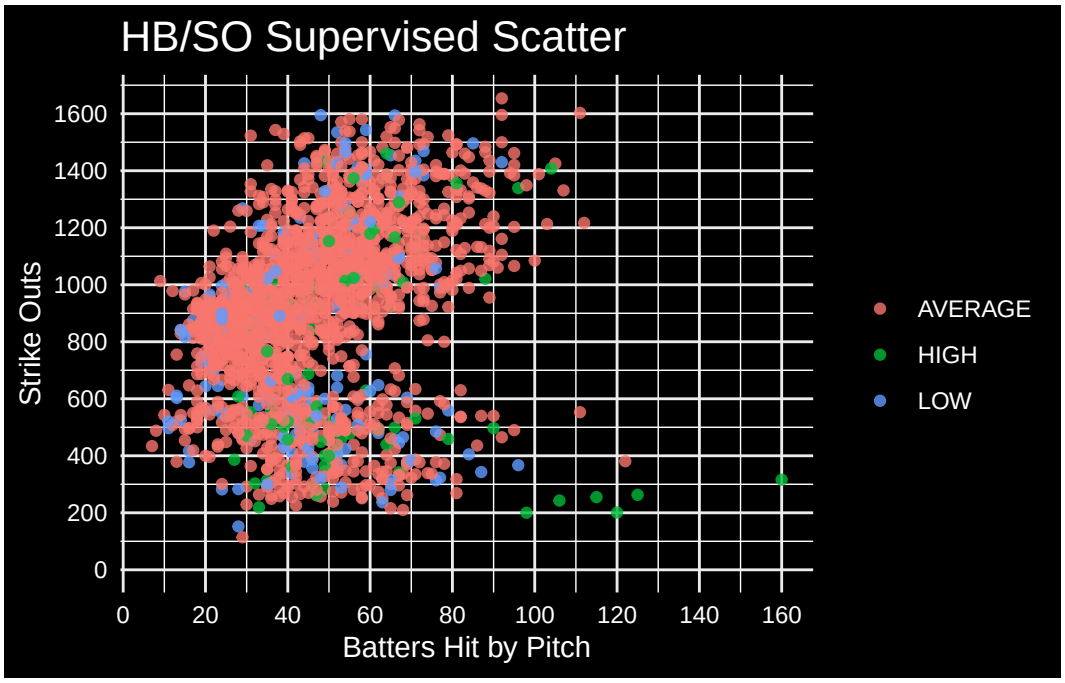
On average, BB seems to have a fairly wide distribution with the fewest amount of outliers, all being below the lower quartile by a significant difference. The Average seems to vary widely, with lots of outliers.

5. Supervised Scatterplot for HB/SO, CG/SHO, IPOuts/DP

```
supervised_data = data %>%
  select(HBP, SO, CG, SHO, IPouts, DP, TARGET)

ggplot(supervised_data, aes(x = HBP, y = SO, color = TARGET)) +
  geom_point(alpha = 0.8) +
  labs(title = "HB/SO Supervised Scatter", y = "Strike Outs", x = "Batters Hit by Pitch") +
  theme_minimal() +
    theme(
```

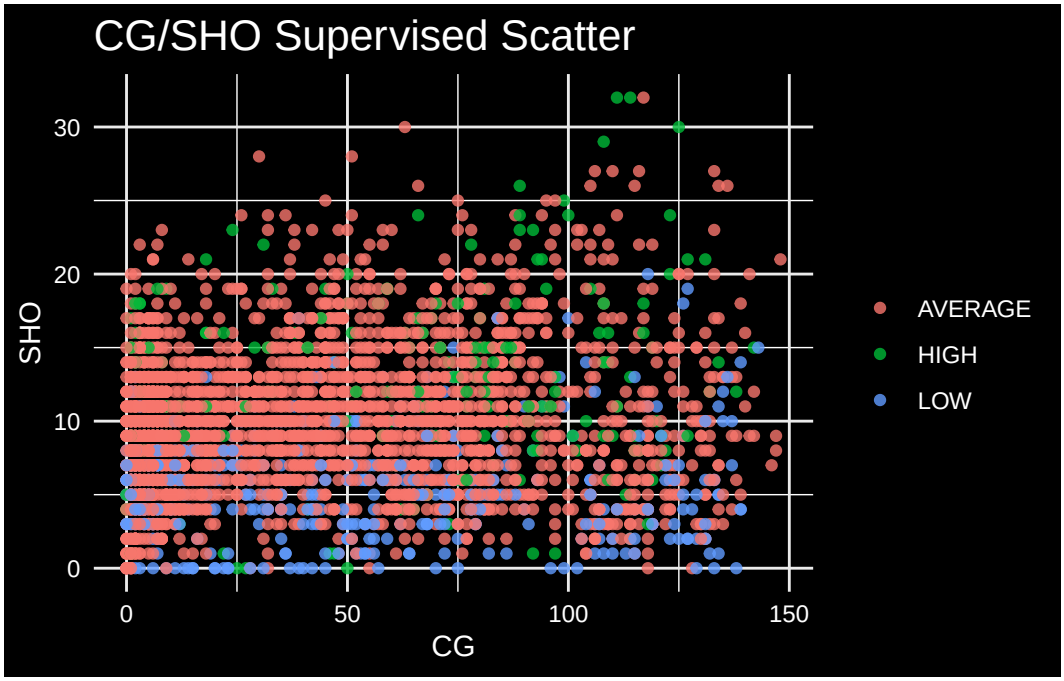
```
    panel.grid.minor = element_line(color = "white"),
    plot.background = element_rect(fill = "black"),
    axis.text = element_text(color = "white"),
    axis.title = element_text(color = "white"),
    plot.title = element_text(color = "white", size = 16),
    legend.text = element_text(color = "white")
  ) +
  scale_x_continuous(breaks = seq(0, 200, 20)) +
  scale_y_continuous(breaks = seq(0, 2000, 200))
```

```
Warning: Removed 1158 rows containing missing values or values outside the scale range
(`geom_point()`).
```

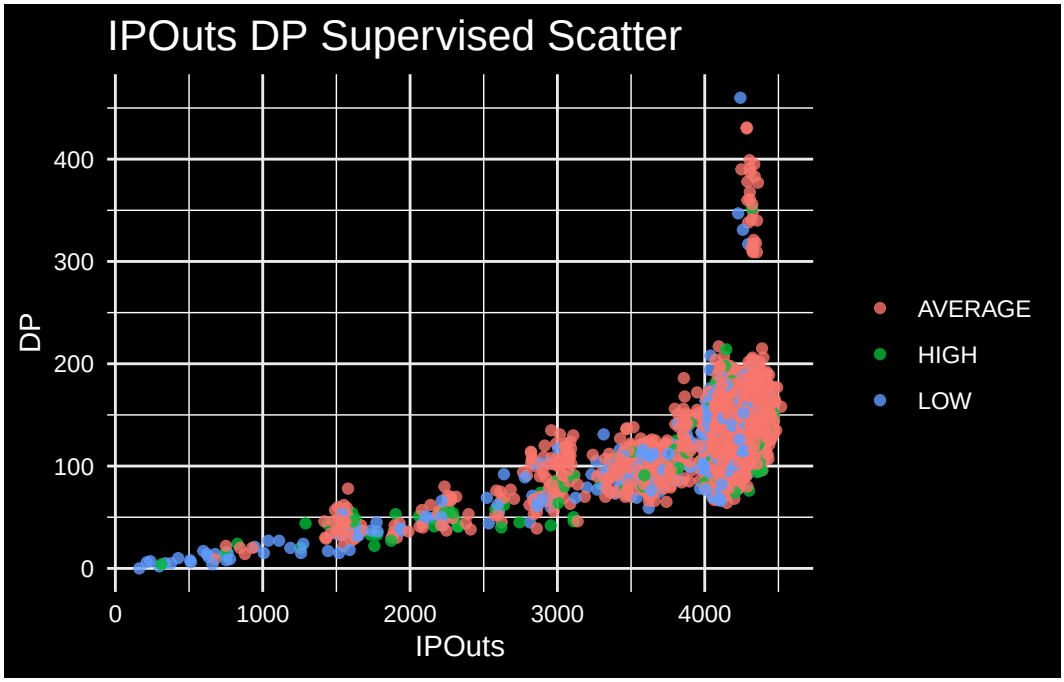



```
ggplot(supervised_data, aes(x = CG, y = SHO, color = TARGET)) +
  geom_point(alpha = 0.8) +
  labs(title = "CG/SHO Supervised Scatter", y = "SHO", x = "CG") +
  theme_minimal() +
  theme(
    panel.grid.minor = element_line(color = "white"),
    plot.background = element_rect(fill = "black"),
    axis.text = element_text(color = "white"),
```

```
    axis.title = element_text(color = "white"),
    plot.title = element_text(color = "white", size = 16),
    legend.text = element_text(color = "white")
  )
```




```
ggplot(supervised_data, aes(x = IPouts, y = DP, color = TARGET)) +
  geom_point(alpha = 0.8) +
  labs(title = "IPOuts DP Supervised Scatter", y = "DP", x = "IPOuts") +
  theme_minimal() +
  theme(
    panel.grid.minor = element_line(color = "white"),
    plot.background = element_rect(fill = "black"),
    axis.text = element_text(color = "white"),
    axis.title = element_text(color = "white"),
    plot.title = element_text(color = "white", size = 16),
    legend.text = element_text(color = "white")
  )
```



Predicting target, based on the above scatterplots, will be very difficult. Each target is fairly well distributed through the x and y axes. It may be slightly easier to predict using the IPOuts and DP plot, with the exception of values above 4200

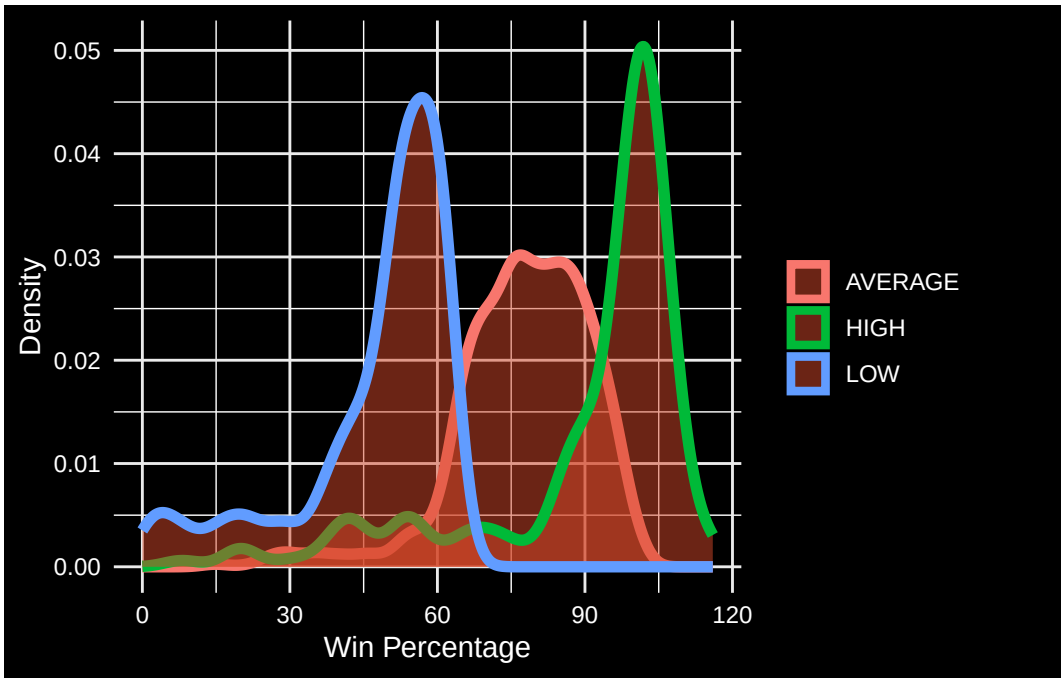
6. Density Plots W v E

We will view density plots against Win Percentage W, and Errors per game E

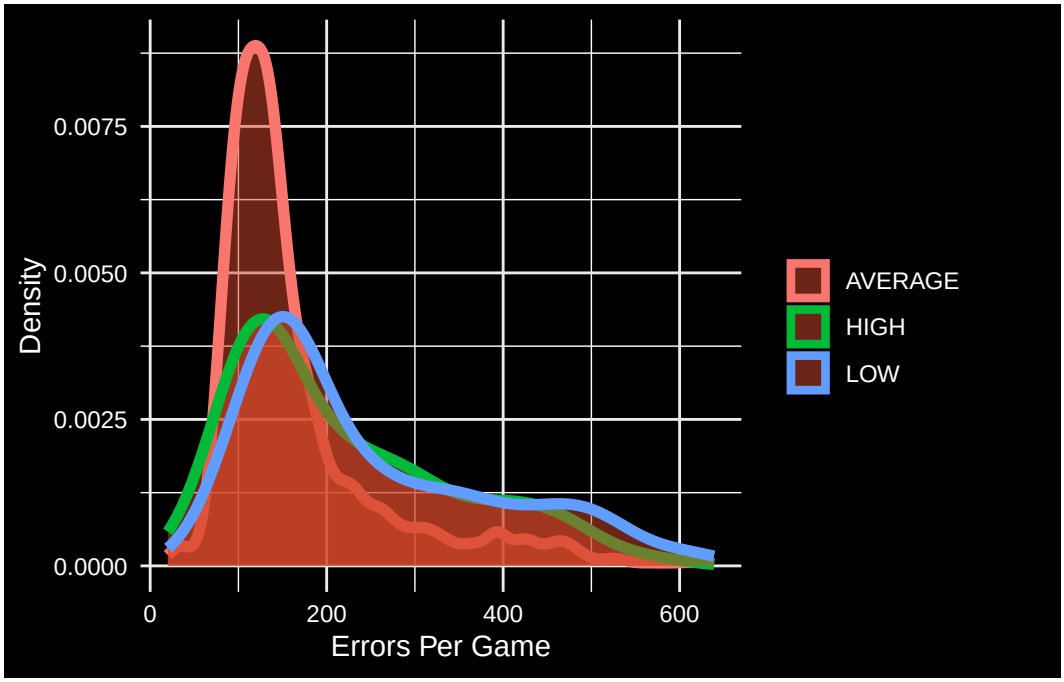
```
density_data = data %>%
  select(W, E, TARGET)

ggplot(density_data, aes(x = W, color = as.factor(TARGET))) +
  geom_density(fill = "#D6492A", alpha = 0.5, size = 2) +
  theme_minimal() +
  labs(x = "Win Percentage", y = "Density") +
  theme(
    panel.grid.minor = element_line(color = "white"),
    plot.background = element_rect(fill = "black"),
    axis.text = element_text(color = "white"),
    axis.title = element_text(color = "white"),
    plot.title = element_text(color = "white", size = 16),
```

```
    legend.text = element_text(color = "white")
  )
```

```
Warning: Using `size` aesthetic for lines was deprecated in ggplot2 3.4.0.
i Please use `linewidth` instead.
```




```
ggplot(density_data, aes(x = E, color = as.factor(TARGET))) +
  geom_density(fill = "#D6492A", alpha = 0.5, size = 2) +
  theme_minimal() +
  labs(x = "Errors Per Game", y = "Density") +
  theme(
    panel.grid.minor = element_line(color = "white"),
    plot.background = element_rect(fill = "black"),
    axis.text = element_text(color = "white"),
    axis.title = element_text(color = "white"),
    plot.title = element_text(color = "white", size = 16),
    legend.text = element_text(color = "white")
  )
```



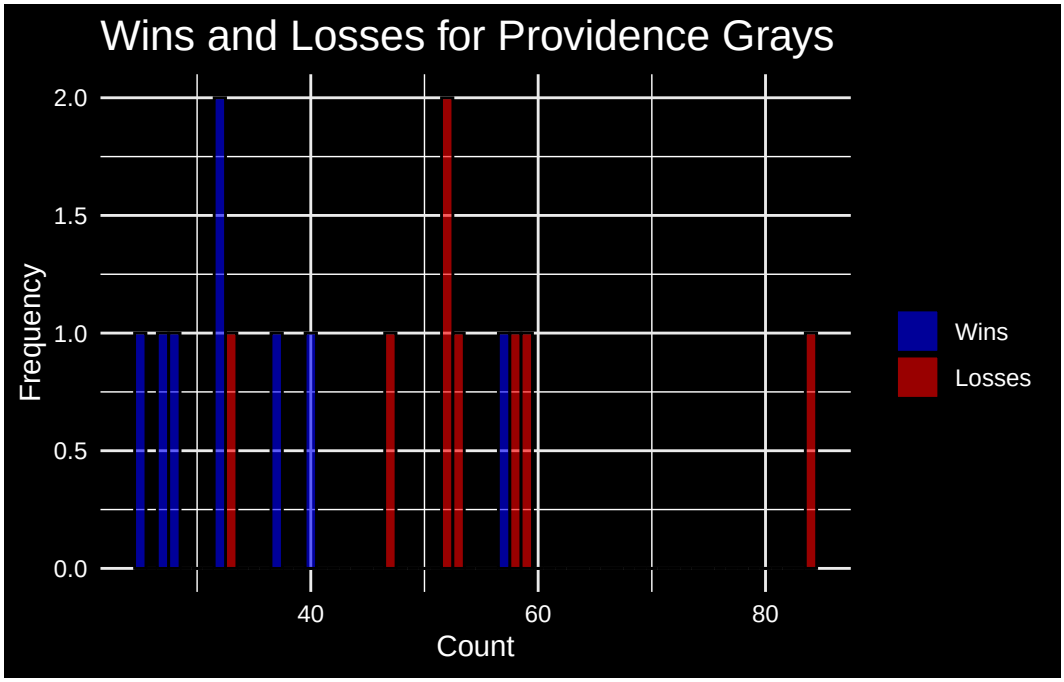
There seems to be a very small difference in density for E when comparing high and low Target's. Maybe because of more conservative play? If a variable can be engineered that represents the aggressiveness of a team, it can be used to validate that hypothesis. The win percentage density is as expected, a bimodal density plot where the average is in the middle of both peaks.

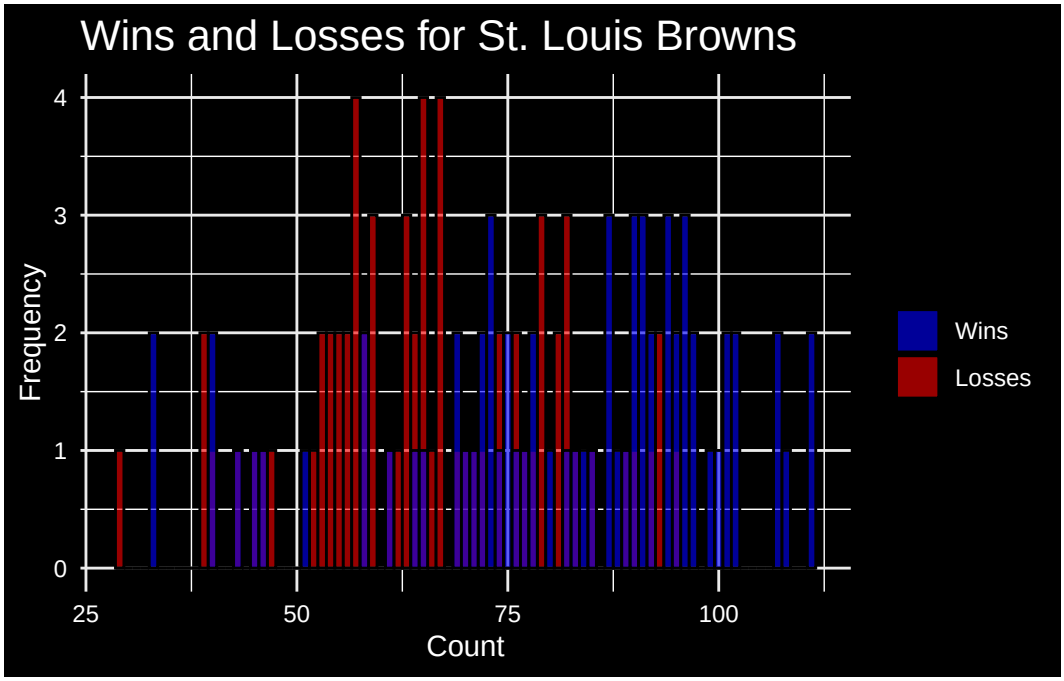
7. World Series Wins Table

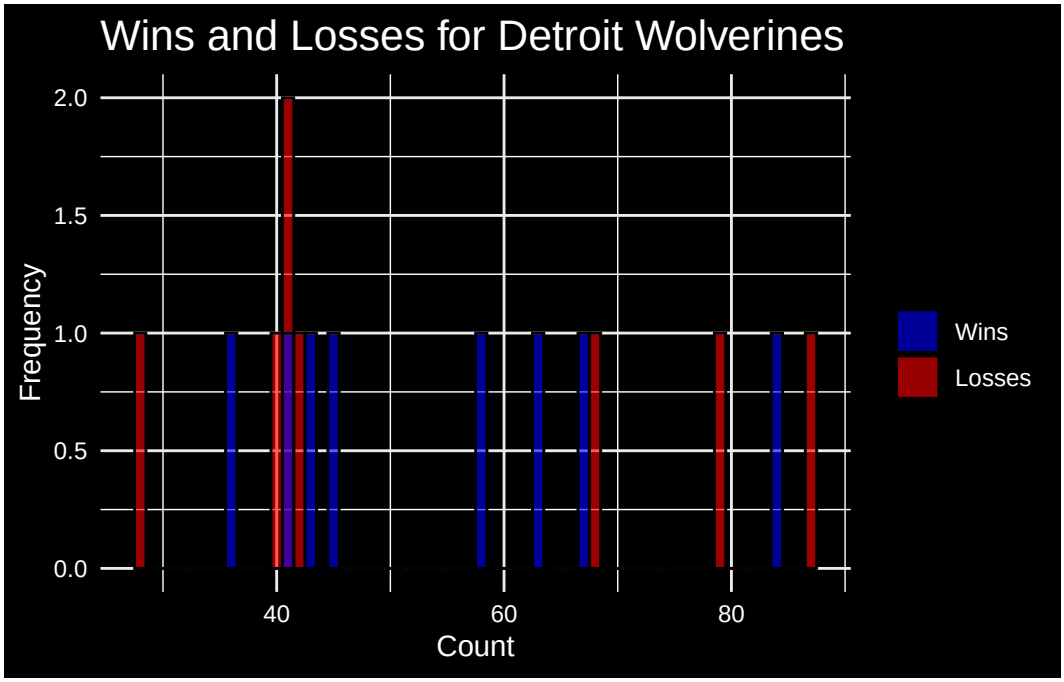
```
world_series <- data %>%
  filter(WSWin == "Y") %>%
  select(name, WSWin) %>%
  mutate(
    LOW = sum(TARGET == "LOW"),
    AVERAGE = sum(TARGET == "AVERAGE"),
    HIGH = sum(TARGET == "HIGH")
  )


world_series = world_series %>%
  left_join(data %>% select(name, W, L), by = "name")
```

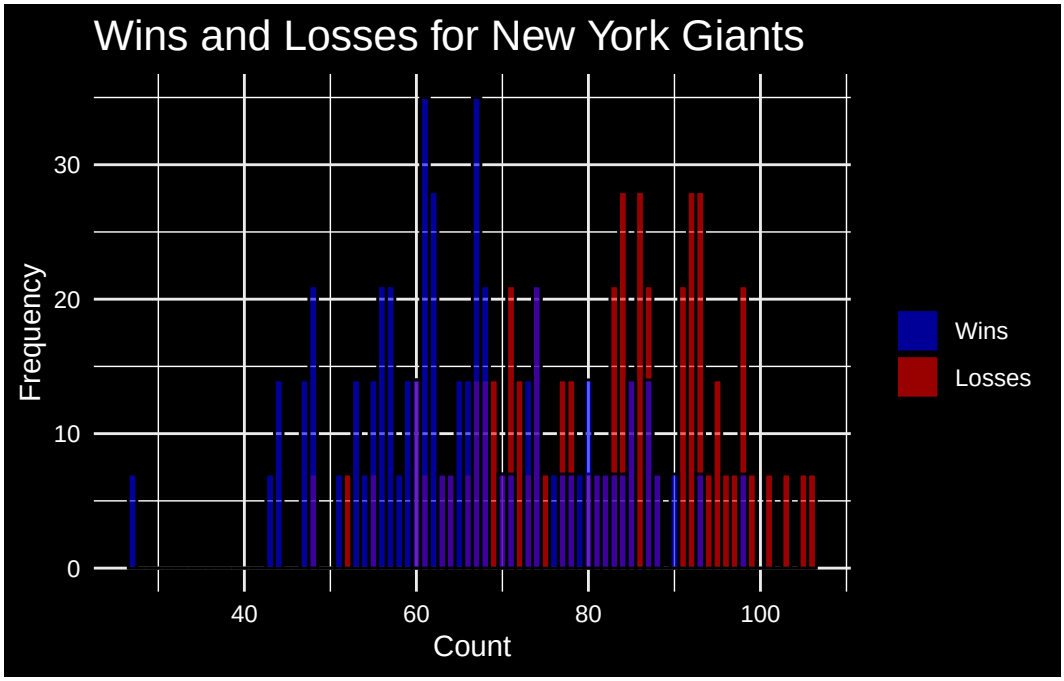
```
Warning in left_join(., data %>% select(name, W, L), by = "name"): Detected an unexpected ma
i Row 1 of `x` matches multiple rows in `y`.
i Row 170 of `y` matches multiple rows in `x`.
i If a many-to-many relationship is expected, set `relationship =
  "many-to-many"` to silence this warning.
```

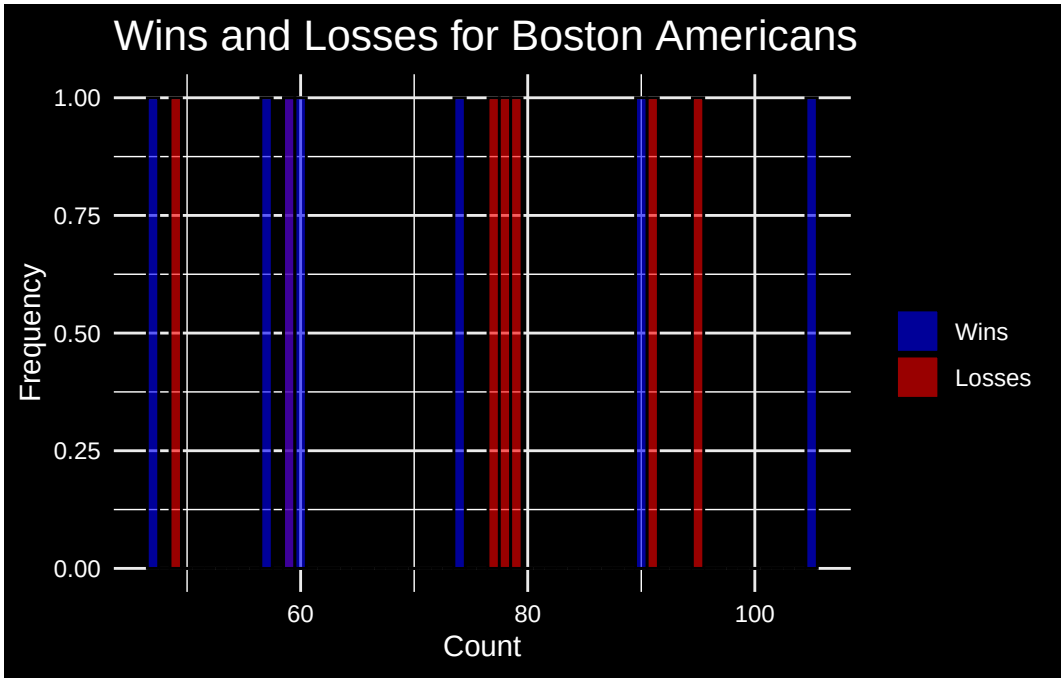
```r
teams = unique(world_series$name)

for (team in teams) {
  team_data = world_series %>% filter(name == team)

  looped_plots = ggplot(team_data) +
    geom_histogram(aes(x = W, fill = "Wins"), binwidth = 1, alpha = 0.6, color = "black") +
    geom_histogram(aes(x = L, fill = "Losses"), binwidth = 1, alpha = 0.6, color = "black") +
    scale_fill_manual(values = c("blue", "red"), name = "Type", labels = c("Wins", "Losses")) +
    labs(x = "Count", y = "Frequency", title = paste("Wins and Losses for", team)) +
    theme_minimal() +
    theme(
    panel.grid.minor = element_line(color = "white"),
    plot.background = element_rect(fill = "black"),
    axis.text = element_text(color = "white"),
    axis.title = element_text(color = "white"),
    plot.title = element_text(color = "white", size = 16),
    legend.text = element_text(color = "white")
  )

  print(looped_plots)
}
```

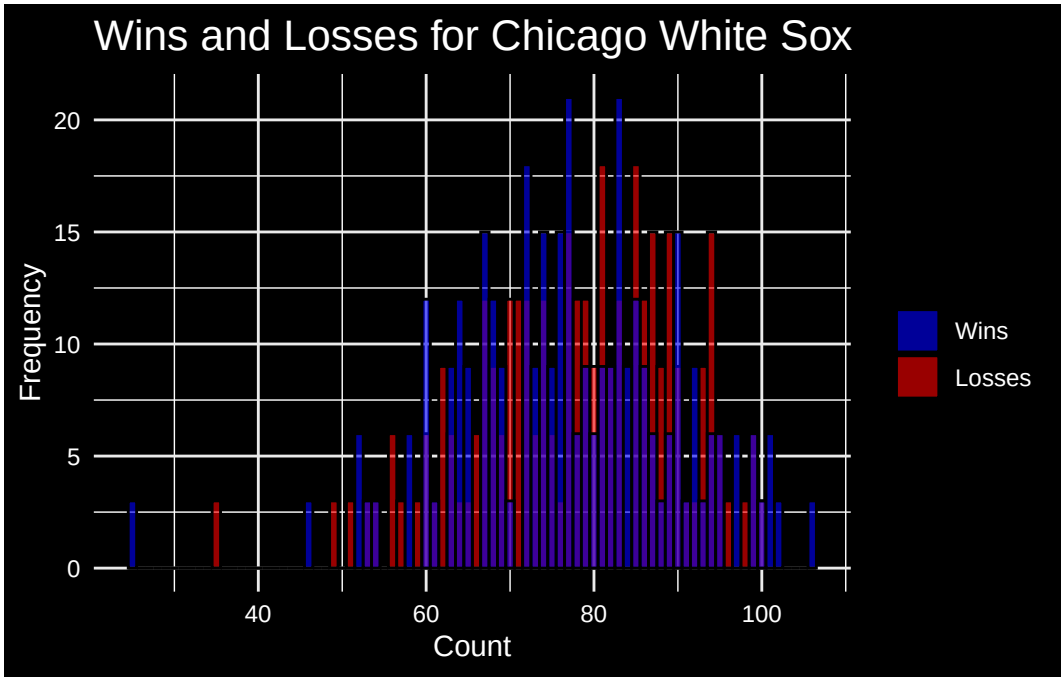

Wins and Losses for Providence Grays
2.0
1.5
1.0
0.5
0.0
Frequency
40
60
80
Count
Wins
Losses

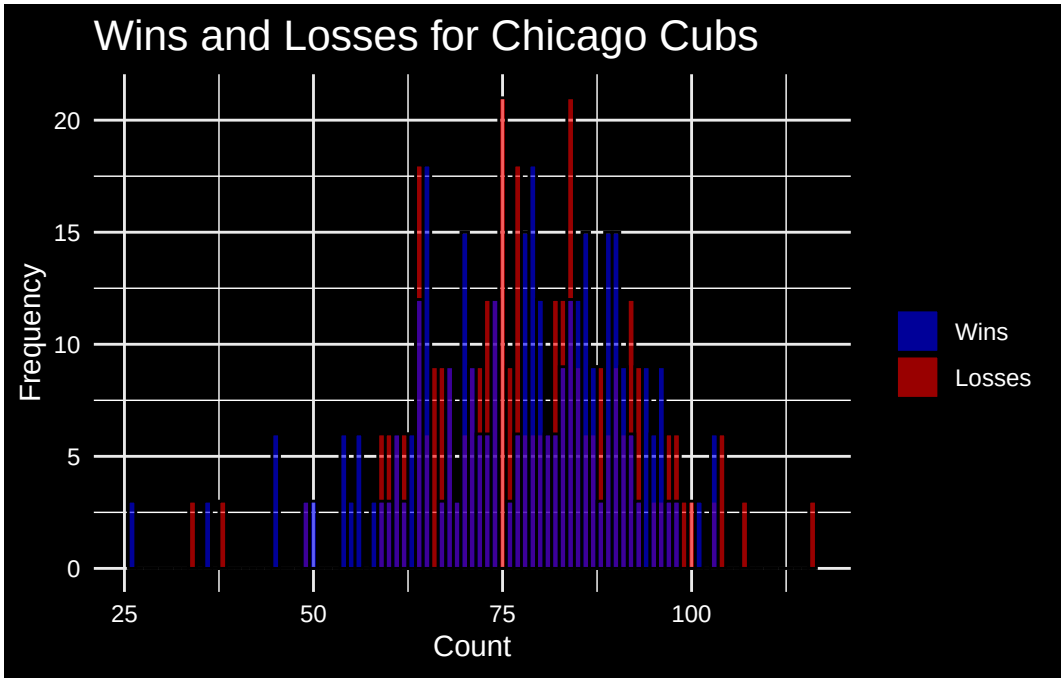

Wins and Losses for St. Louis Browns
4
3
2
1
0
Frequency
25
50
75
100
Count
Wins
Losses

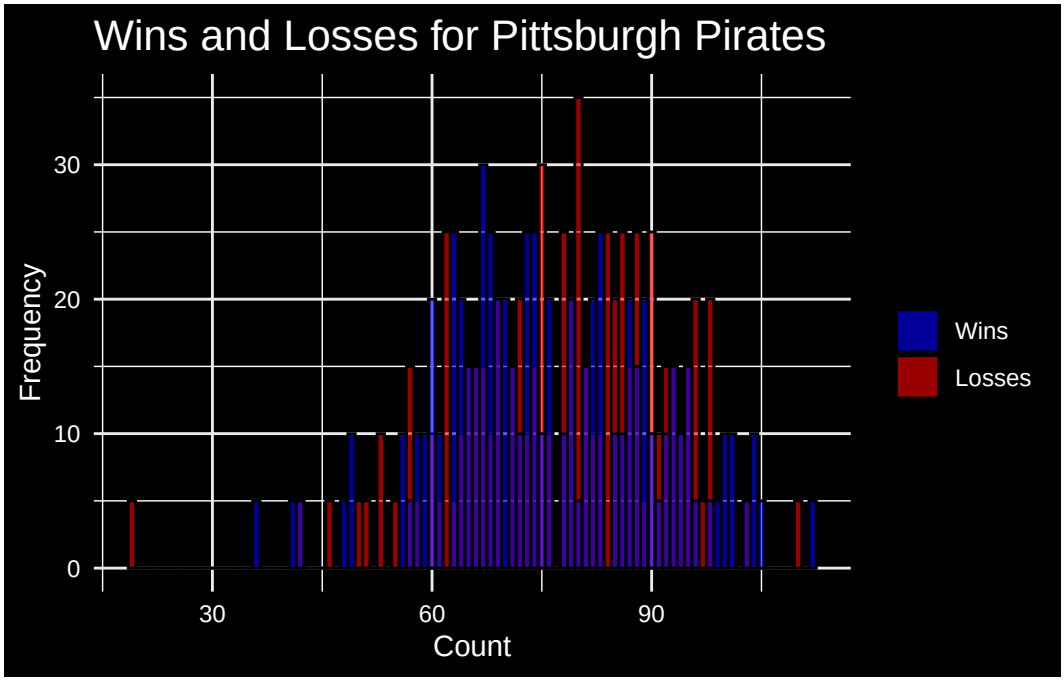

Wins and Losses for Detroit Wolverines
2.0
1.5
1.0
0.5
0.0
Frequency
40
60
80
Count
Wins
Losses

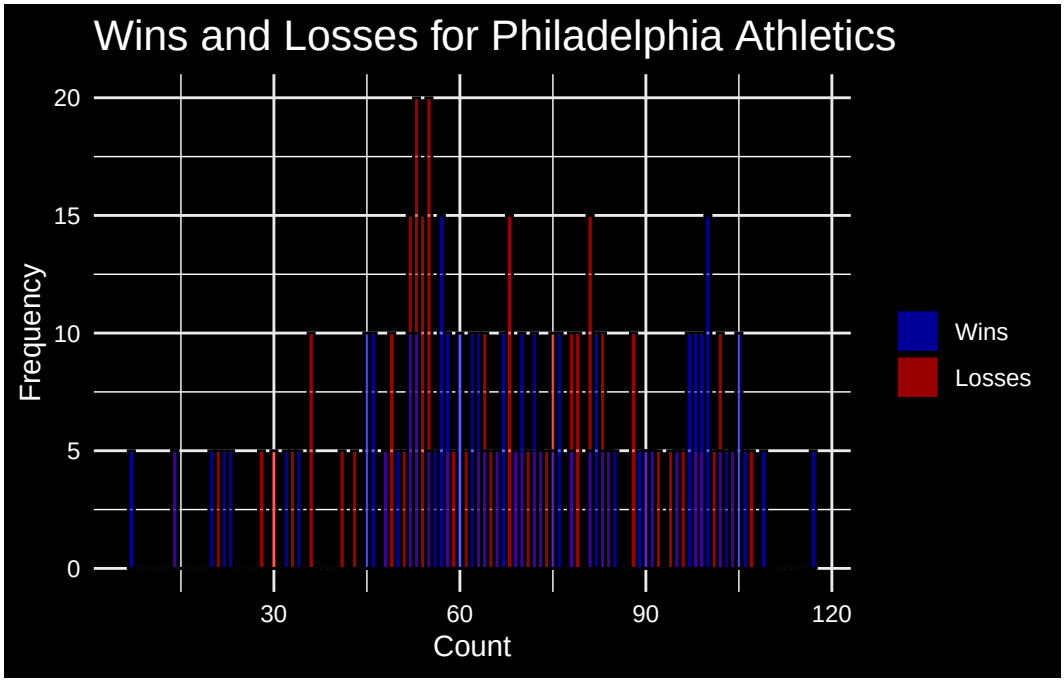

Wins and Losses for New York Giants
30
20
10
0
Frequency
40
60
80
100
Count
Wins
Losses

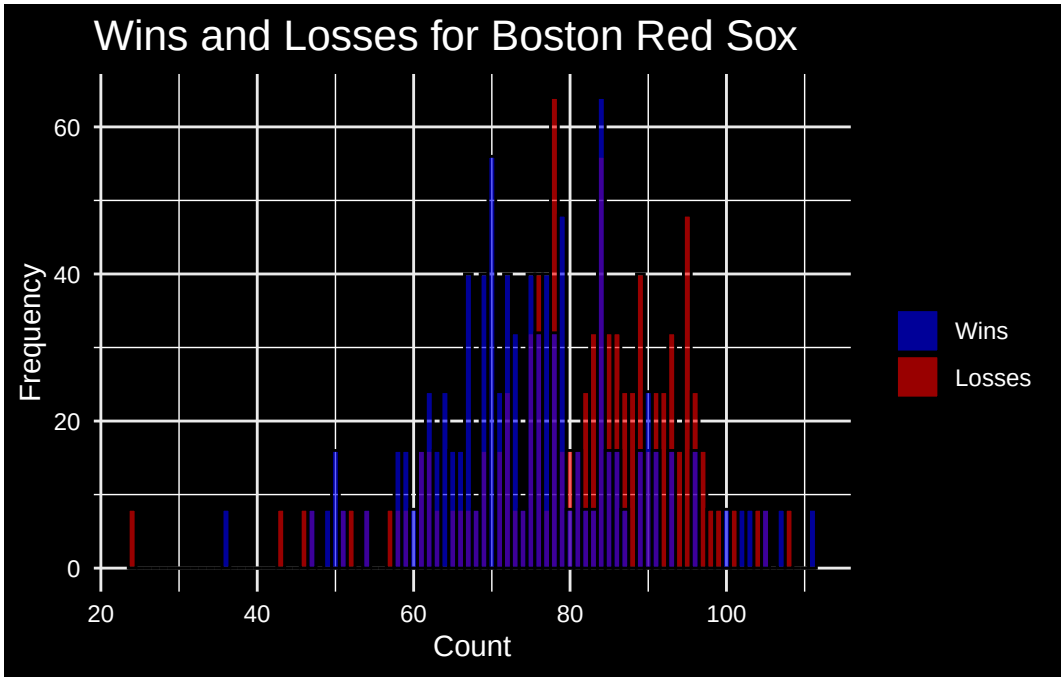

Wins and Losses for Boston Americans
1.00
0.75
0.50
0.25
0.00
Frequency
60
80
100
Count
Wins
Losses

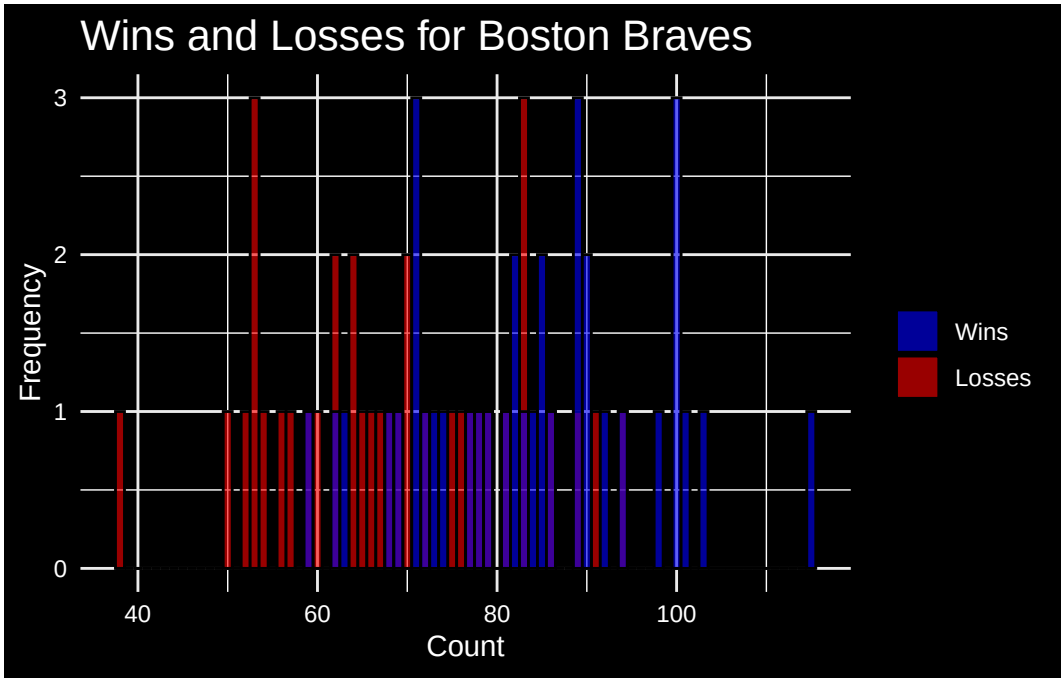

Wins and Losses for Chicago White Sox
20
15
10
5
0
Frequency
40
60
80
100
Count
Wins
Losses

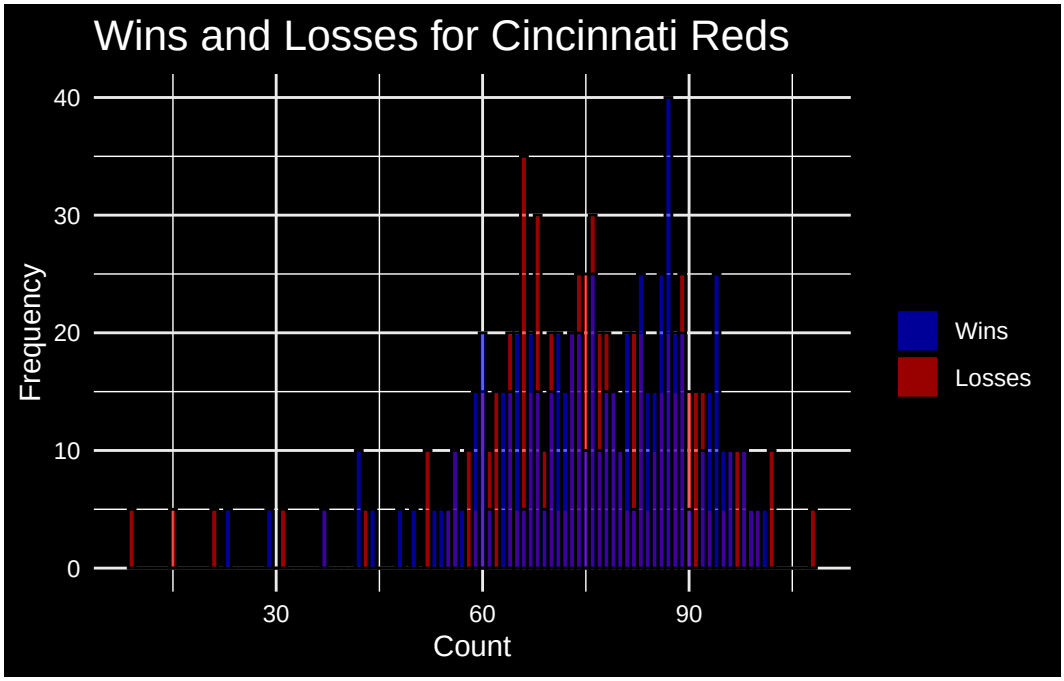

Wins and Losses for Chicago Cubs
Frequency
20
15
10
5
0
25
50
75
100
Count
Wins
Losses

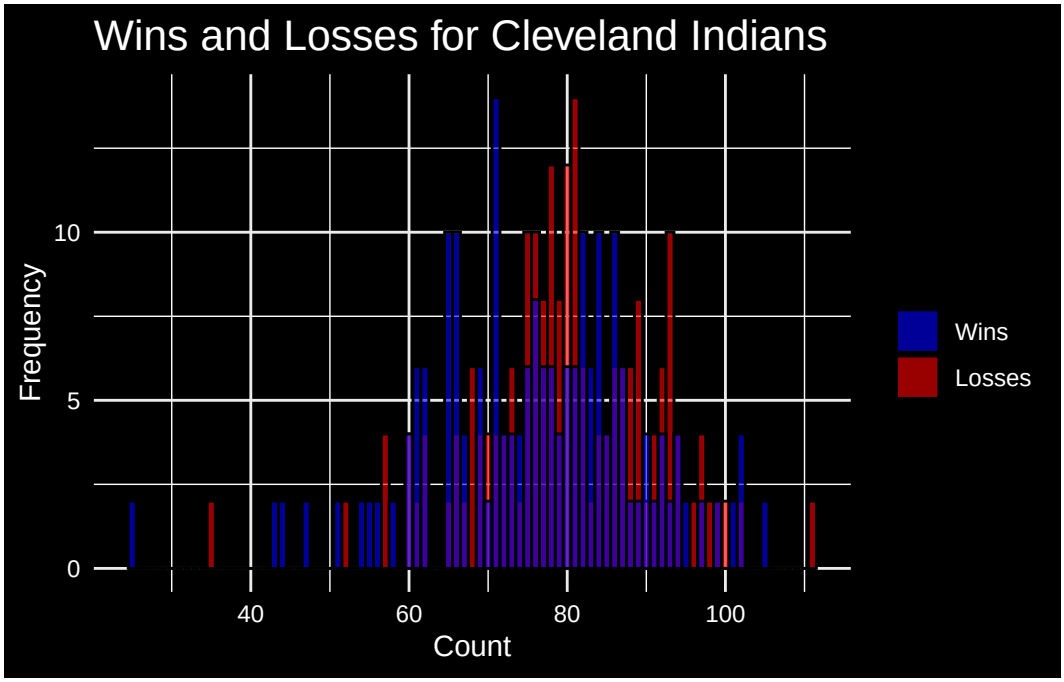

Wins and Losses for Pittsburgh Pirates
Frequency
30
20
10
0
30
60
90
Count
Wins
Losses

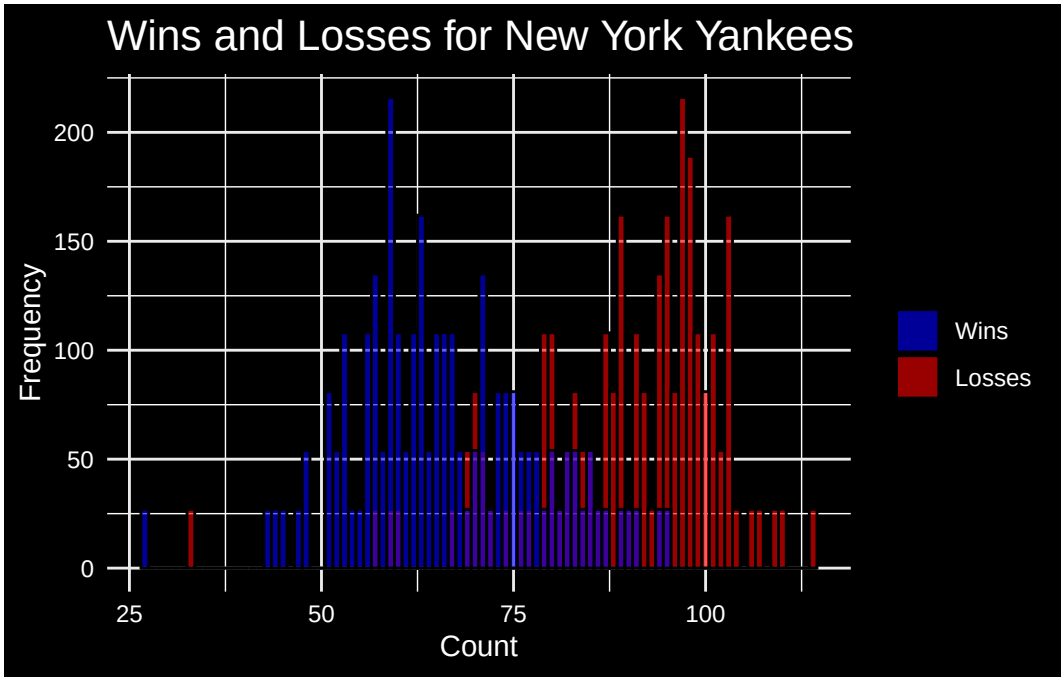

Wins and Losses for Philadelphia Athletics
Frequency
20
15
10
5
0
30
60
90
120
Count
Wins
Losses

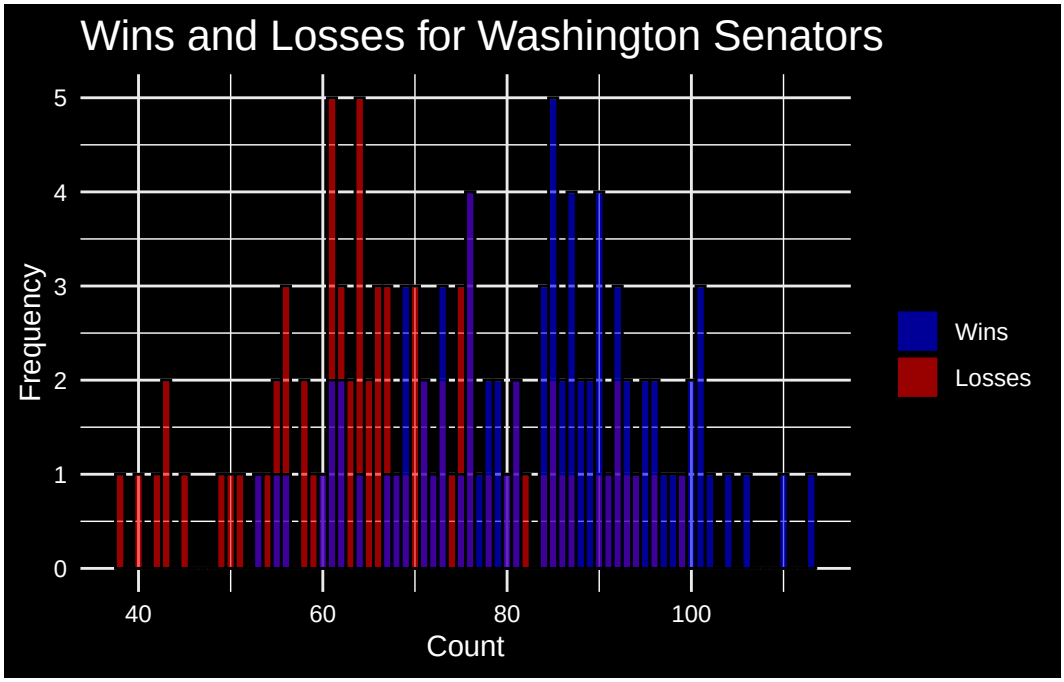

Wins and Losses for Boston Red Sox
Frequency
60
40
20
0
20
40
60
80
100
Count
Wins
Losses

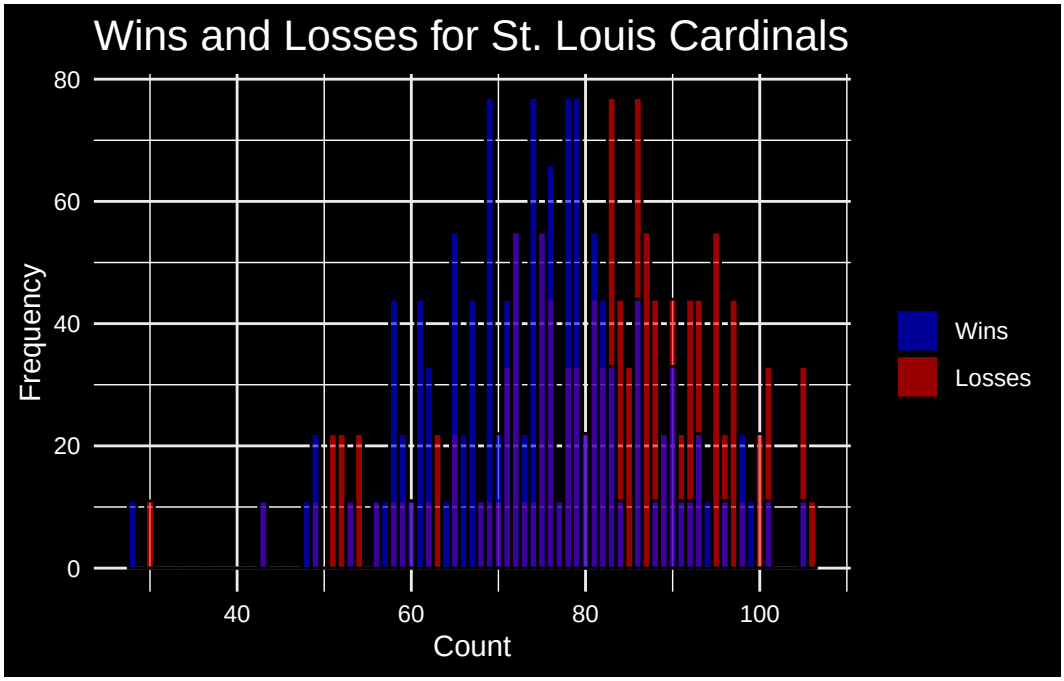

Wins and Losses for Boston Braves
Frequency
Count
0
1
2
3
40
60
80
100
Wins
Losses

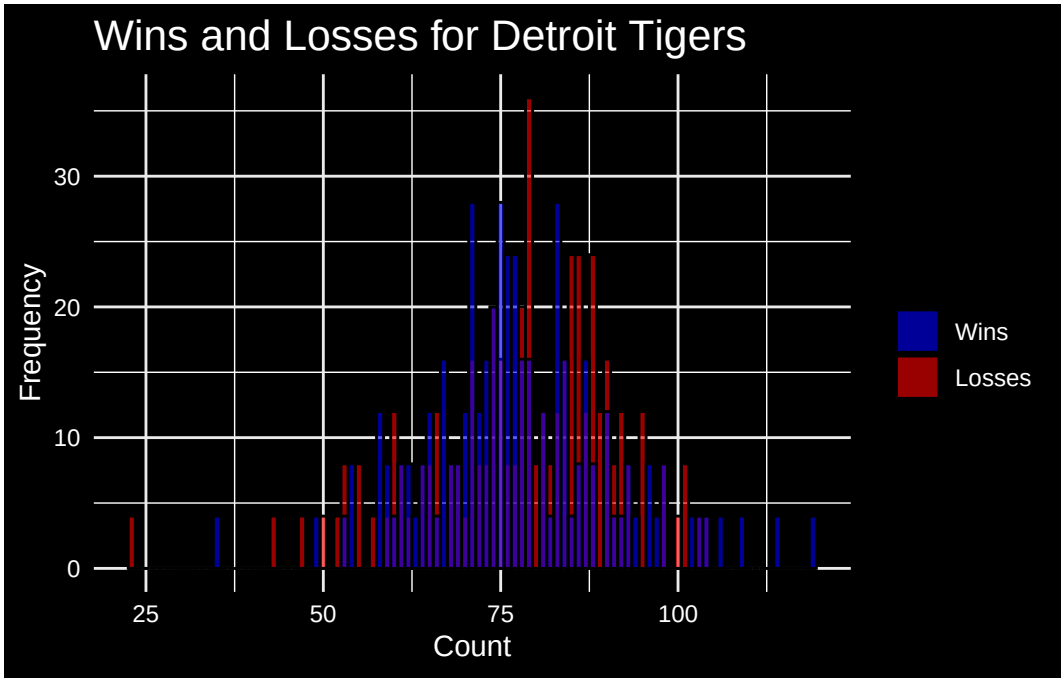

Wins and Losses for Cincinnati Reds
Frequency
Count
0
10
20
30
40
30
60
90
Wins
Losses

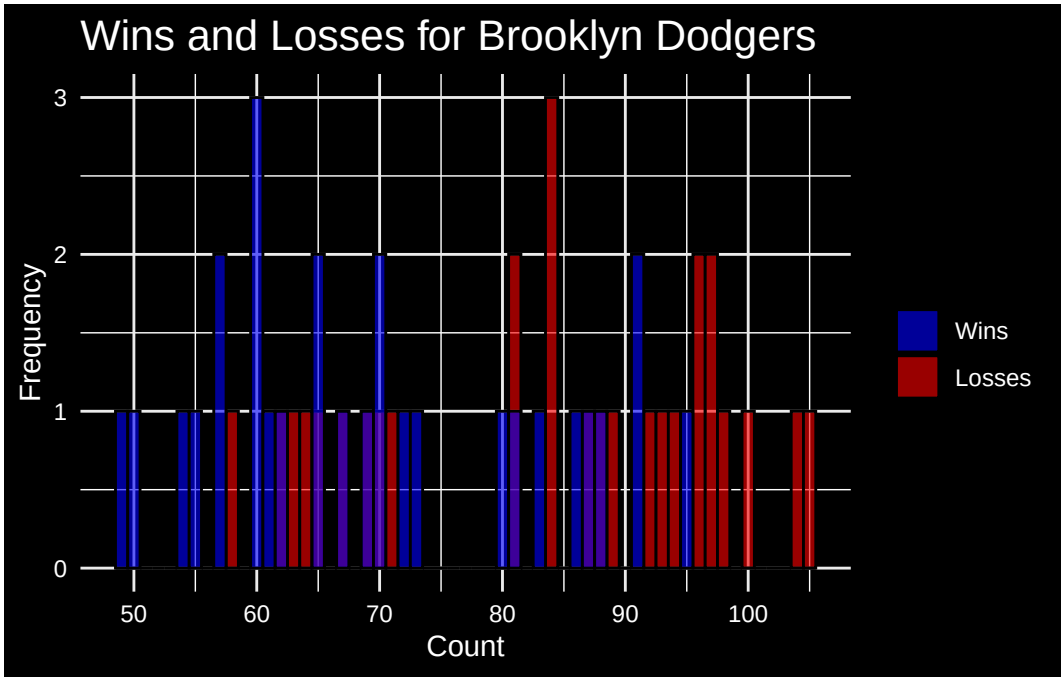

Wins and Losses for Cleveland Indians
Frequency
10
5
0
40
60
80
100
Count
Wins
Losses

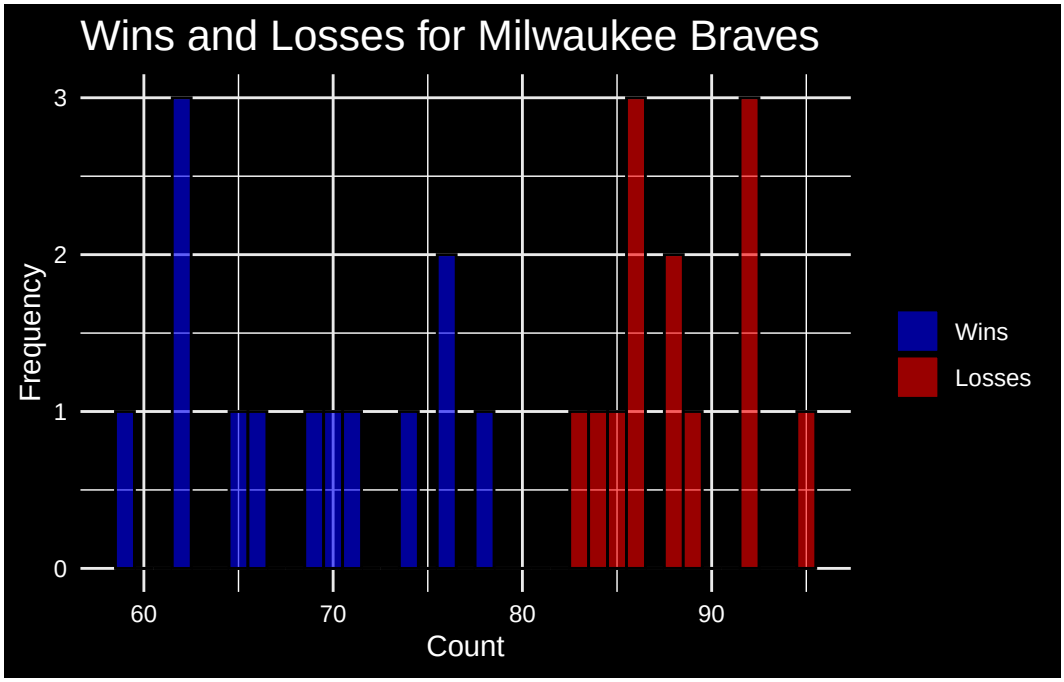

Wins and Losses for New York Yankees
Frequency
200
150
100
50
0
25
50
75
100
Count
Wins
Losses

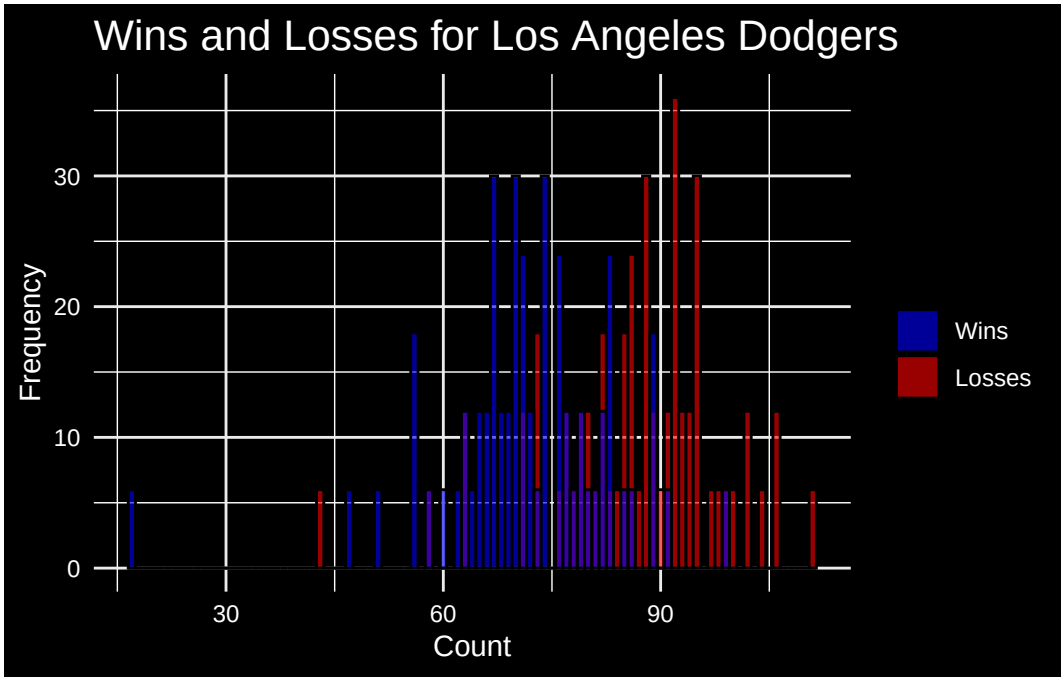

Wins and Losses for Washington Senators
Frequency
0
1
2
3
4
5
40
60
80
100
Count
Wins
Losses

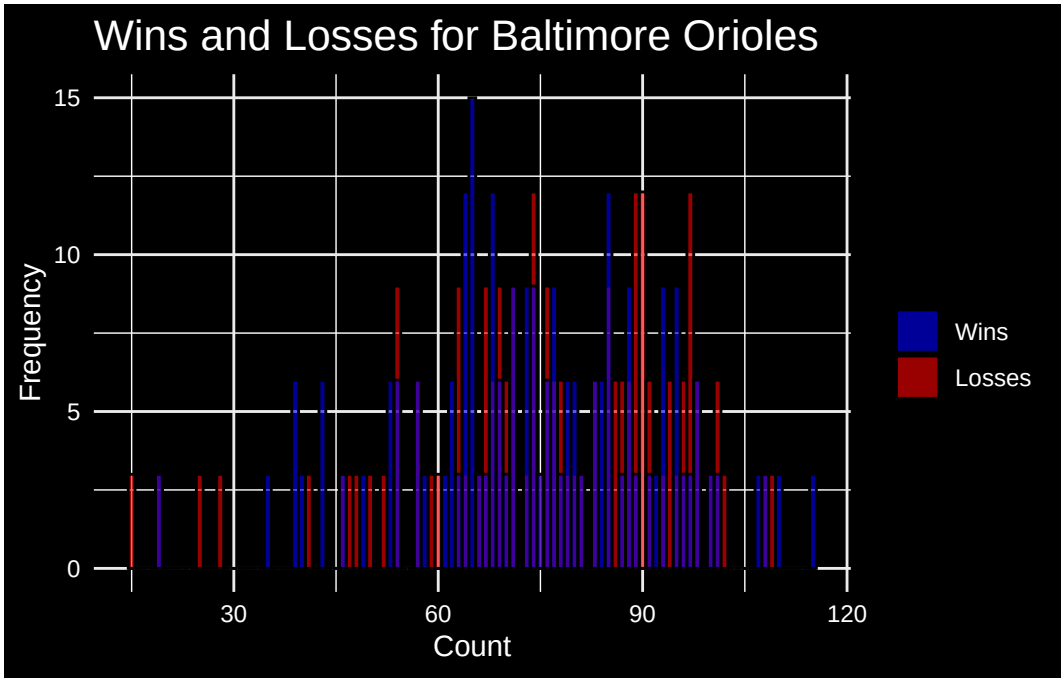

Wins and Losses for St. Louis Cardinals
Frequency
0
20
40
60
80
40
60
80
100
Count
Wins
Losses

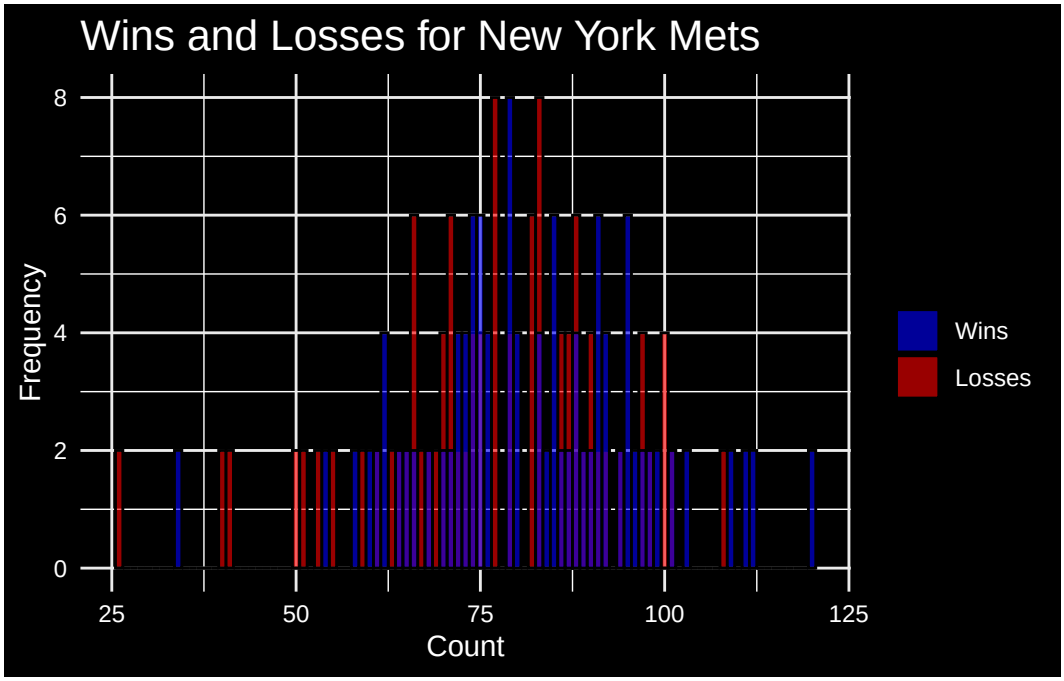

Wins and Losses for Detroit Tigers
Frequency
30
20
10
0
25
50
75
100
Count
Wins
Losses

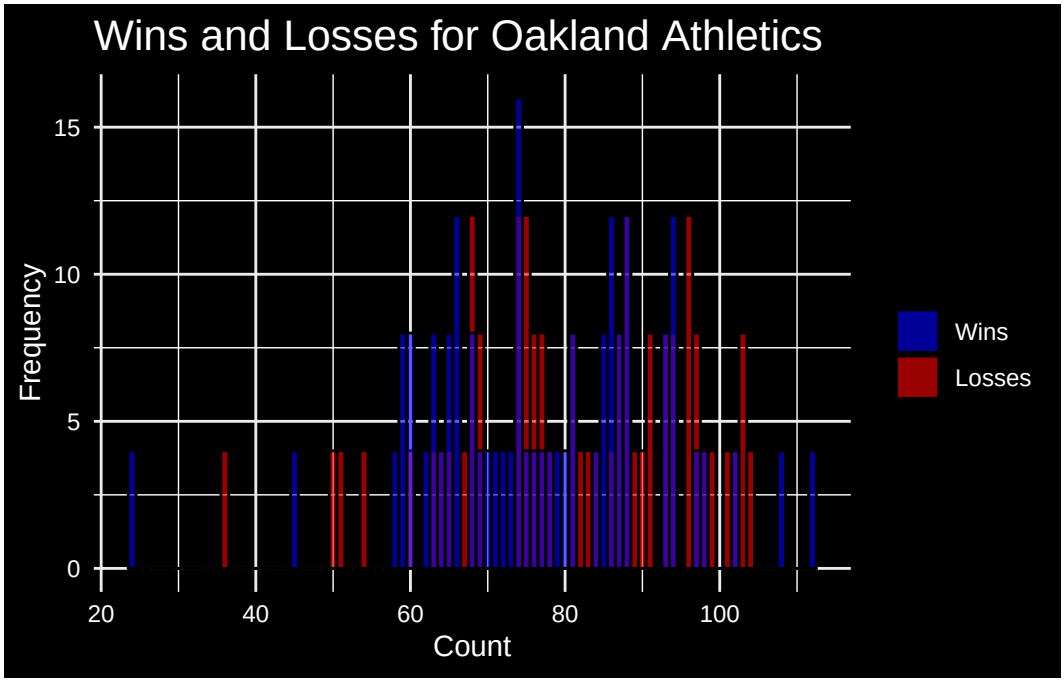

Wins and Losses for Brooklyn Dodgers
Frequency
3
2
1
0
50
60
70
80
90
100
Count
Wins
Losses

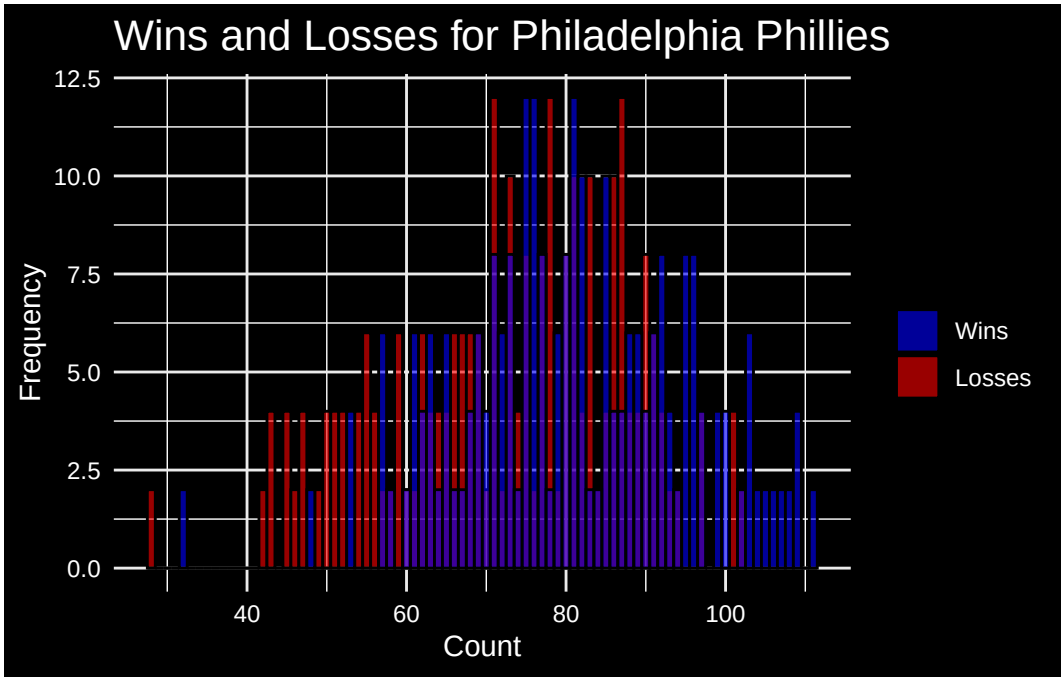

Wins and Losses for Milwaukee Braves
Frequency
Count
Wins
Losses
0
1
2
3
60
70
80
90

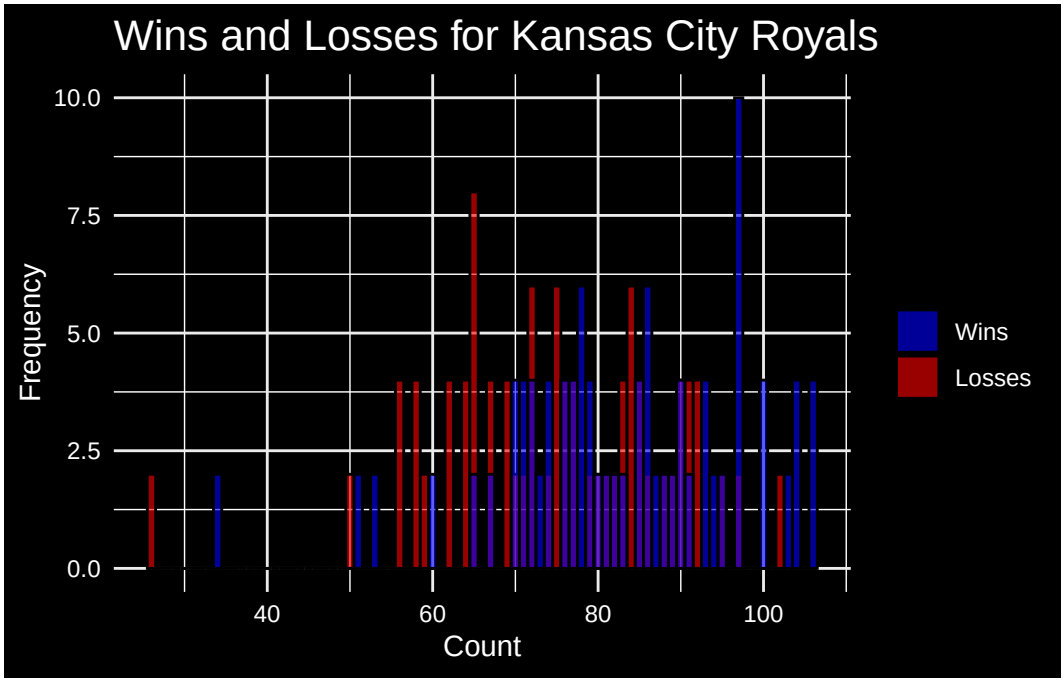

Wins and Losses for Los Angeles Dodgers
Frequency
Count
Wins
Losses
0
10
20
30
30
60
90

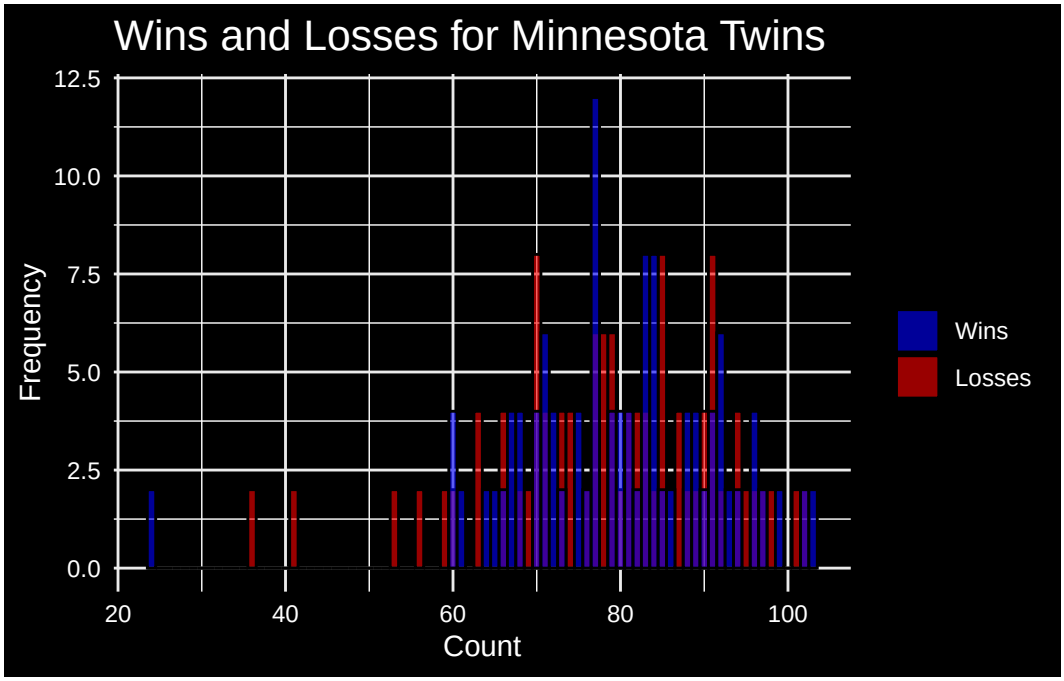

Wins and Losses for Baltimore Orioles
Frequency
15
10
5
0
30
60
90
120
Count
Wins
Losses

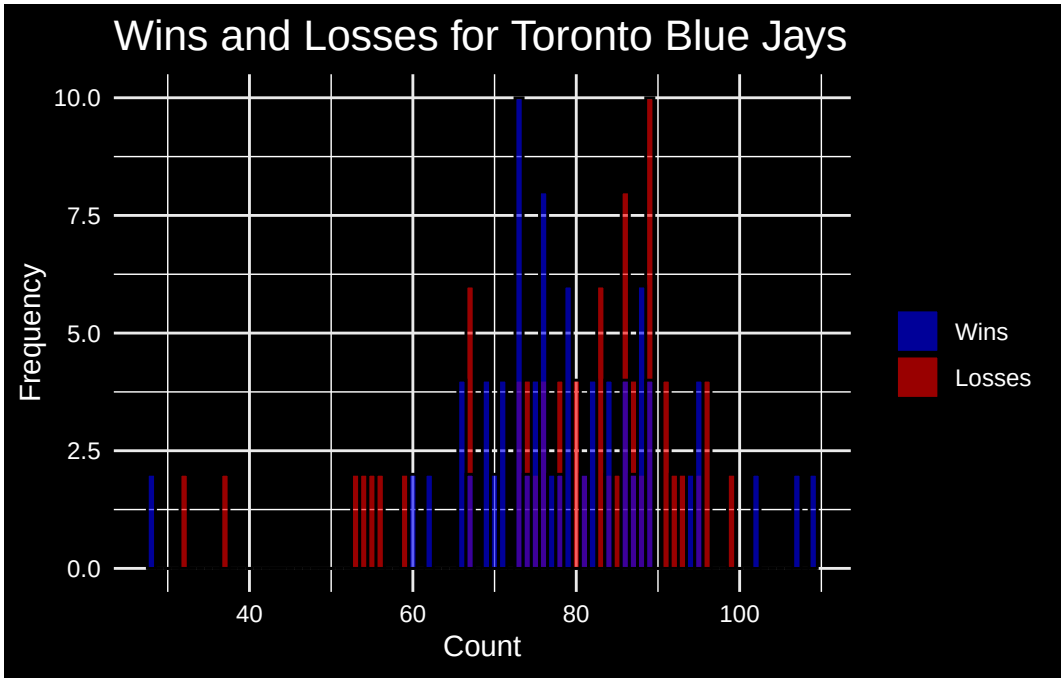

Wins and Losses for New York Mets
Frequency
8
6
4
2
0
25
50
75
100
125
Count
Wins
Losses

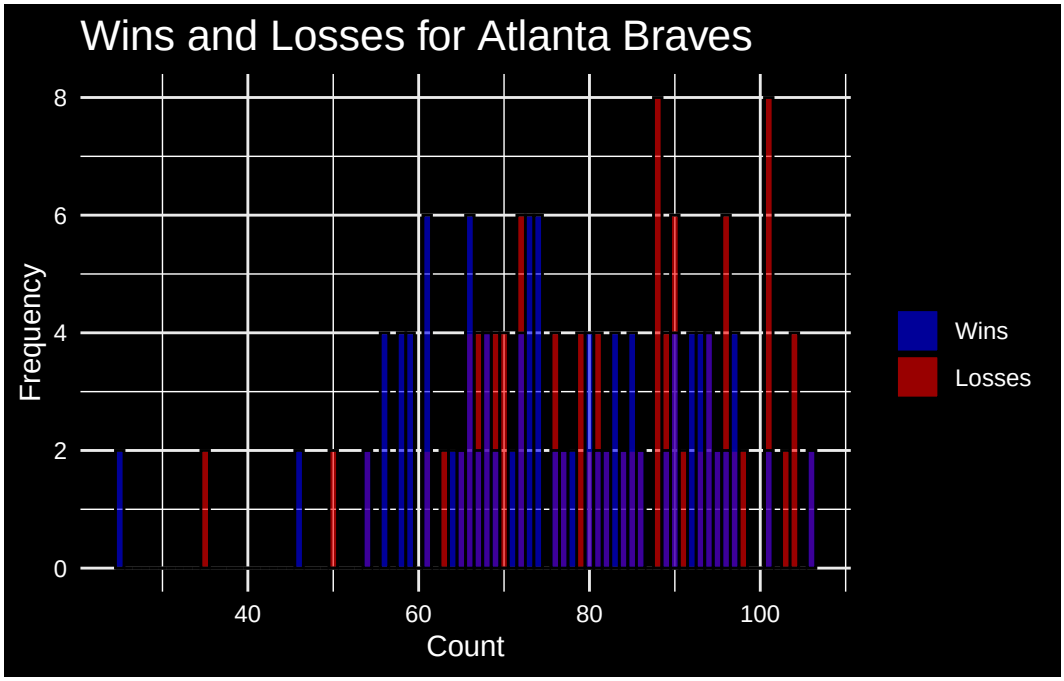

Wins and Losses for Oakland Athletics
Frequency
15
10
5
0
20
40
60
80
100
Count
Wins
Losses

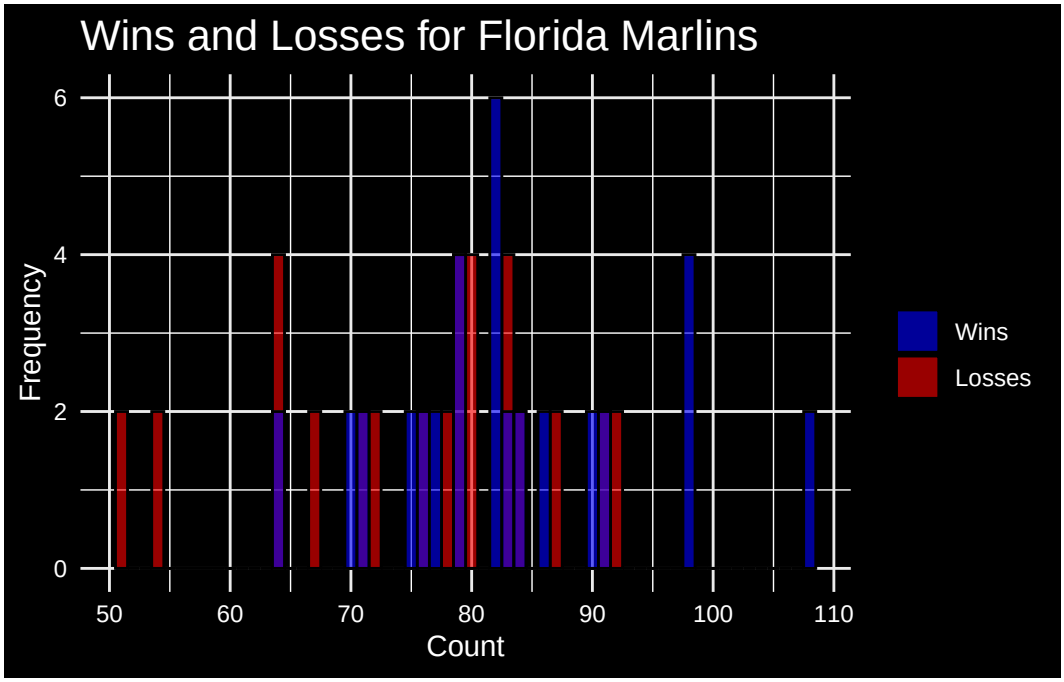

Wins and Losses for Philadelphia Phillies
Frequency
12.5
10.0
7.5
5.0
2.5
0.0
40
60
80
100
Count
Wins
Losses

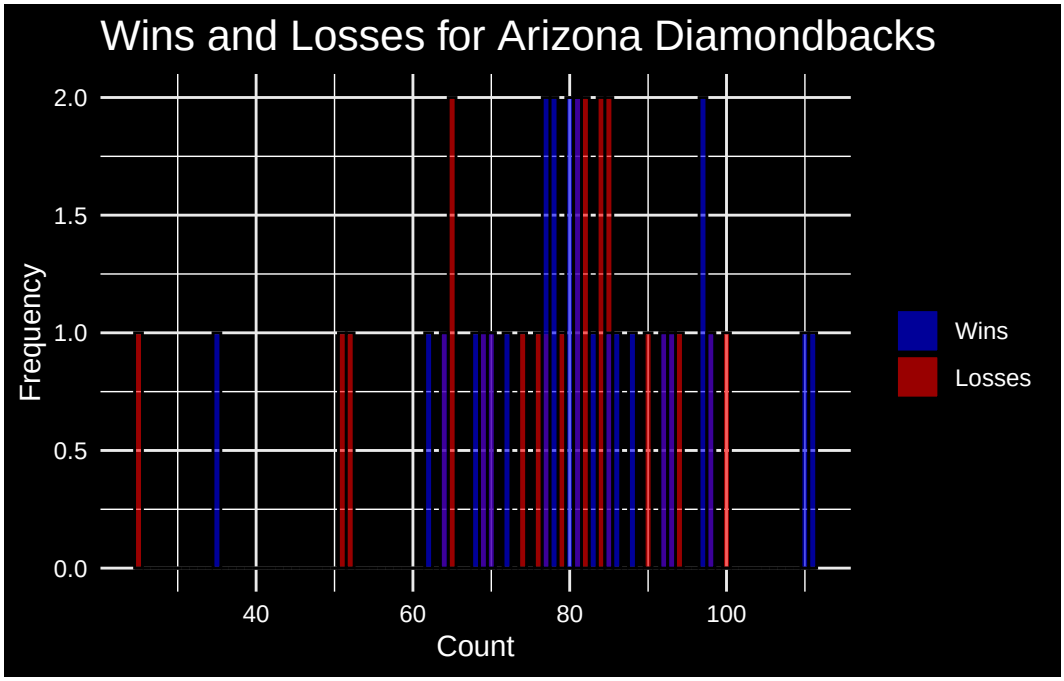

Wins and Losses for Kansas City Royals
10.0
7.5
5.0
2.5
0.0
Frequency
40
60
80
100
Count
Wins
Losses

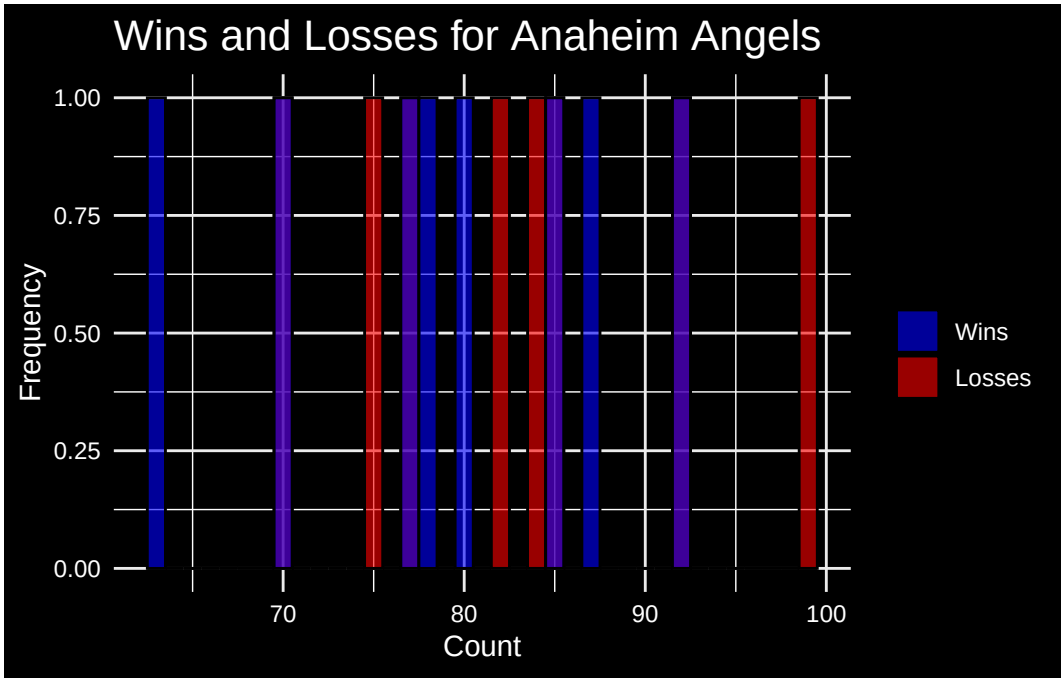

Wins and Losses for Minnesota Twins
12.5
10.0
7.5
5.0
2.5
0.0
Frequency
20
40
60
80
100
Count
Wins
Losses

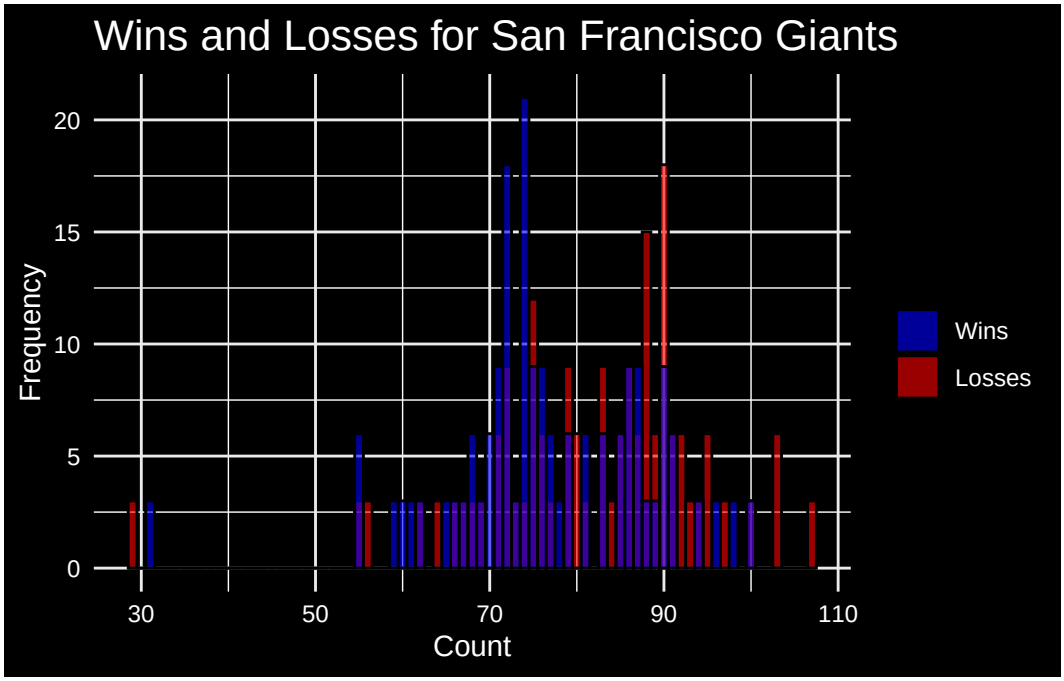

Wins and Losses for Toronto Blue Jays
10.0
7.5
5.0
2.5
0.0
Frequency
40
60
80
100
Count
Wins
Losses

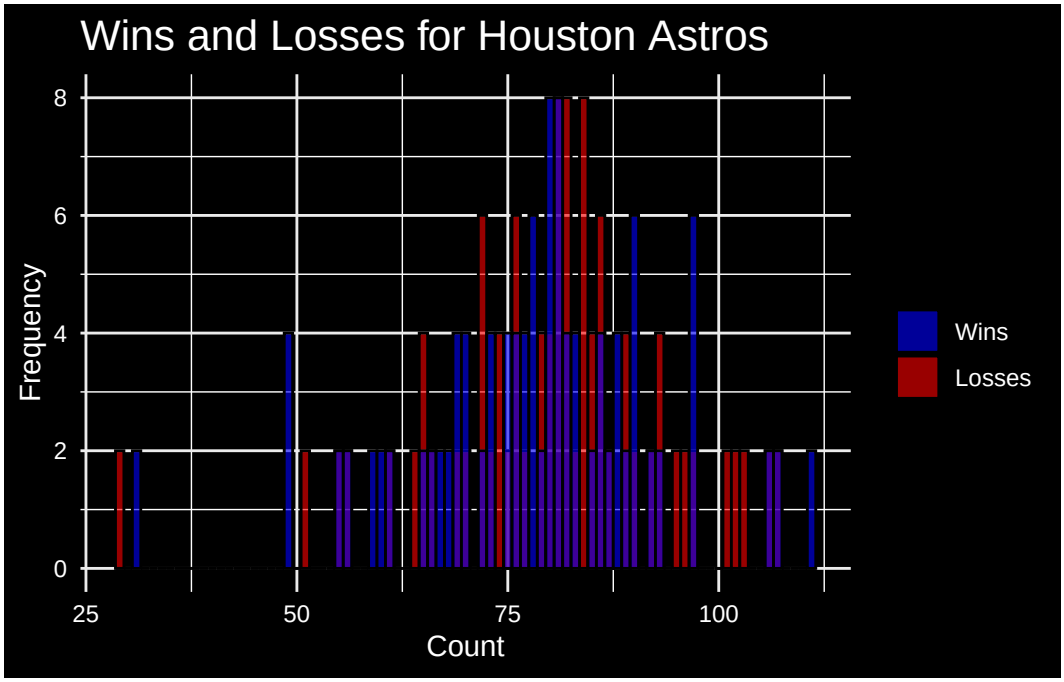

Wins and Losses for Atlanta Braves
8
6
4
2
0
Frequency
40
60
80
100
Count
Wins
Losses

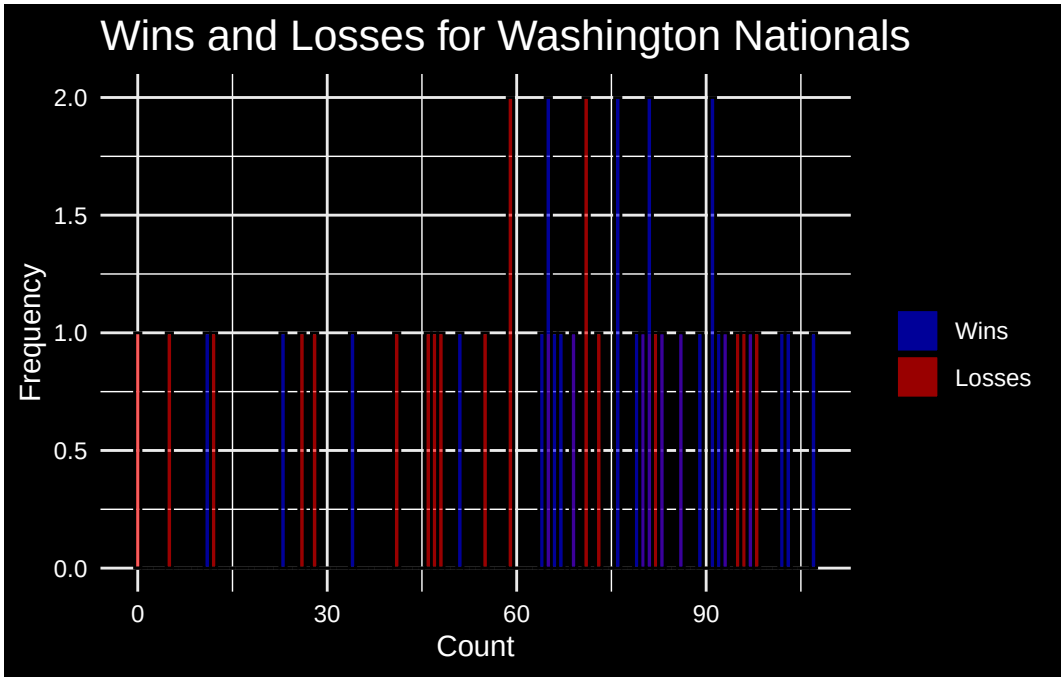

Wins and Losses for Florida Marlins
Frequency
0
2
4
6
50
60
70
80
90
100
110
Count
Wins
Losses

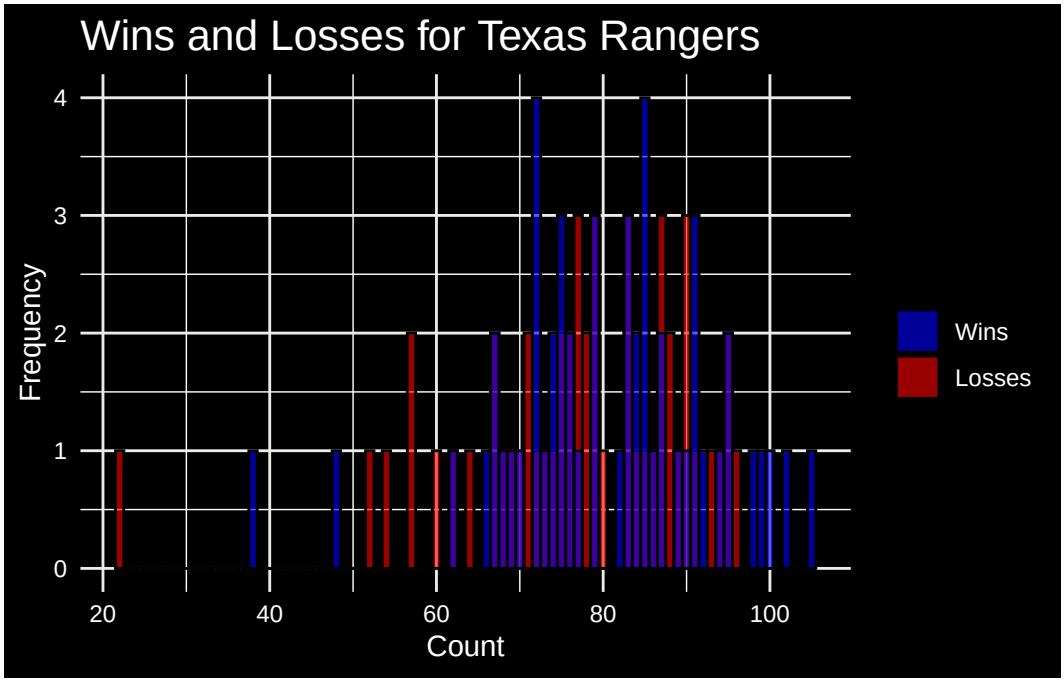

Wins and Losses for Arizona Diamondbacks
Frequency
0.0
0.5
1.0
1.5
2.0
40
60
80
100
Count
Wins
Losses


Wins and Losses for Anaheim Angels
1.00
0.75
0.50
0.25
0.00
Frequency
70
80
90
100
Count
Wins
Losses


Wins and Losses for San Francisco Giants
20
15
10
5
0
Frequency
30
50
70
90
110
Count
Wins
Losses


Wins and Losses for Houston Astros
Frequency
Count
Wins
Losses
0
2
4
6
8
25
50
75
100


Wins and Losses for Washington Nationals
Frequency
Count
Wins
Losses
0.0
0.5
1.0
1.5
2.0
0
30
60
90



8. Transformed Data

```
transformed_data = data %>%
  mutate(
    z_H = (H - mean(H, na.rm = TRUE)) / sd(H, na.rm = TRUE),
    z_SO = (SO - mean(SO, na.rm = TRUE)) / sd(SO, na.rm = TRUE),
    z_SOA = (SOA - mean(SOA, na.rm = TRUE)) / sd(SOA, na.rm = TRUE),
    z_SHO = (SHO - mean(SHO, na.rm = TRUE)) / sd(SHO, na.rm = TRUE),
    z_FP = (FP - mean(FP, na.rm = TRUE)) / sd(FP, na.rm = TRUE)
  )

lm_model = lm(WP ~ z_H + z_SO + z_SOA + z_SHO + z_FP, data = transformed_data)

model_summary = summary(lm_model)

summary_table = data.frame(c(model_summary$r.squared, model_summary$coefficients))

model_summary
```

```
Call:
lm(formula = WP ~ z_H + z_SO + z_SOA + z_SHO + z_FP, data = transformed_data)

Residuals:
     Min       1Q   Median       3Q      Max
-0.34912 -0.05030 -0.00125  0.04825  0.39146

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  0.495559   0.001419 349.136   <2e-16 ***
z_H          0.019678   0.001943  10.128   <2e-16 ***
z_SO        -0.076725   0.004212 -18.215   <2e-16 ***
z_SOA        0.073244   0.004197  17.453   <2e-16 ***
z_SHO        0.030149   0.001545  19.509   <2e-16 ***
z_FP        -0.001228   0.002412  -0.509    0.611
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.07812 on 3023 degrees of freedom
  (16 observations deleted due to missingness)
Multiple R-squared:  0.3061,    Adjusted R-squared:  0.3049
F-statistic: 266.7 on 5 and 3023 DF,  p-value: < 2.2e-16
```

z_FP and z_SO are negatively correlated with the win percentage attribute, while the others are positively correlated. z_FP is not significant in the model. And the model only accounts for 30.6% of the variation.

9. Decision Trees

```r
subset_data = data[, 9:ncol(data)]
subset_data = na.omit(subset_data)
subset_data$TARGET = as.factor(subset_data$TARGET)

set.seed(123)
train_index = sample(1:nrow(subset_data), 0.8 * nrow(subset_data))
train_data = subset_data[train_index, ]
test_data = subset_data[-train_index, ]

tree_model1 = ranger(TARGET ~ ., data = train_data, num.trees = 1, max.depth = 5)
tree_model2 = ranger(TARGET ~ ., data = train_data, num.trees = 1, min.node.size = 10)
```

```r
tree_model3 = ranger(TARGET ~ ., data = train_data, num.trees = 1, sample.fraction = 0.7)

calculate_accuracy = function(model, data) {
  predictions = predict(model, data)$predictions
  accuracy = sum(predictions == data$TARGET) / nrow(data)
  return(accuracy)
}

train_acc_model1 = calculate_accuracy(tree_model1, train_data)
test_acc_model1 = calculate_accuracy(tree_model1, test_data)

train_acc_model2 = calculate_accuracy(tree_model2, train_data)
test_acc_model2 = calculate_accuracy(tree_model2, test_data)

train_acc_model3 = calculate_accuracy(tree_model3, train_data)
test_acc_model3 = calculate_accuracy(tree_model3, test_data)

cat("model 1 training accuracy: ", train_acc_model1, "\n")
```

```
model 1 training accuracy:  0.9326683
```

```r
cat("model 1 testing accuracy: ", test_acc_model1, "\n")
```

```
model 1 testing accuracy:  0.8936877
```

```r
cat("model 2 training accuracy: ", train_acc_model2, "\n")
```

```
model 2 training accuracy:  0.9509559
```

```r
cat("model 2 testing accuracy: ", test_acc_model2, "\n")
```

```
model 2 testing accuracy:  0.89701
```

```r
cat("model 3  training accuracy: ", train_acc_model3, "\n")
```

```
model 3  training accuracy:  0.9326683
```

```r
cat("model 3  testing accuracy: ", test_acc_model3, "\n")
```

```
model 3  testing accuracy:  0.8903654
```

Conclusion

Write a conclusion (at most 13 sentences!) summarizing the most important findings of this task; in particular, address the findings obtained related to predicting a successful team (both by Rank and Target) using attributes 7-30. If possible, write about which attributes seem useful for predicting good teams and what you as an individual can learn from this dataset! **6 points (and up to 4 extra points)**