A2

Max

2023-10-06

R Markdown

1.1 (Q1)

```
# Load the tidyverse package
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.3     v readr     2.1.4
## v forcats   1.0.0     v stringr   1.5.0
## v ggplot2   3.4.3     v tibble    3.2.1
## v lubridate 1.9.3     v tidyr     1.3.0
## v purrr     1.0.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become error
```

```
# Load the Stat2Data package and the Hawks dataset
library(Stat2Data)
data("Hawks")

# Create the hSF data frame
hSF <- Hawks %>%
  filter(Species == "RT", Weight >= 1000) %>%
  select(Wing, Weight, Tail)

# Display the first few rows of the resulting data frame
head(hSF)
```

```
##   Wing Weight Tail
## 1  412   1090  230
## 2  412   1210  210
## 3  405   1120  238
## 4  393   1010  222
## 5  371   1010  217
## 6  390   1120  213
```

1.1(Q2)

The data frame hSF has 3 variables. frame contains 398 examples or observations

1.2

```
# Sort the hSF data frame by increasing wing span
hSF_sorted <- hSF %>%
  arrange(Wing)

# Display the top five rows of the sorted data frame
head(hSF_sorted)
```

```
##    Wing Weight Tail
## 1  37.2   1180  210
## 2 111.0   1340  226
## 3 199.0   1290  222
## 4 241.0   1320  235
## 5 262.0   1020  200
## 6 277.0   1500  207
```

1.3(Q1)

```
# Create the hawkSpeciesNameCodes data frame
hawkSpeciesNameCodes <- data.frame(species_code = c("CH", "RT", "SS"),
                                   species_name_full = c("Cooper's", "Red-tailed", "Sharp-shinned")

)
```

1.3(Q2)

```
hawksFullName <- Hawks %>%
  left_join(hawkSpeciesNameCodes, by = c("Species" = "species_code")) %>%
  select(-Species) %>%
  rename(Species = species_name_full)
```

1.3(Q3)

```
hawksFullName %>%
  select(Species, Wing, Weight) %>%
  head(7)
```

```
##         Species Wing Weight
## 1    Red-tailed  385    920
## 2    Red-tailed  376    930
## 3    Red-tailed  381    990
## 4       Cooper's  265    470
## 5 Sharp-shinned  205    170
## 6    Red-tailed  412   1090
## 7    Red-tailed  370    960
```

1.4

```
# Calculate bird BMI and create the hawksWithBMI data frame
hawksWithBMI <- Hawks %>%
  mutate(bird_BMI = 1000 * Weight / (Wing^2)) %>%
  select(Species, bird_BMI) %>%
  arrange(desc(bird_BMI))

# Display the top 8 rows of the hawksWithBMI data frame
head(hawksWithBMI, 8)
```

```
##   Species  bird_BMI
## 1      RT 852.69973
## 2      RT 108.75741
## 3      RT  32.57493
## 4      RT  22.72688
## 5      CH  22.40818
## 6      RT  19.54932
## 7      CH  15.21998
## 8      RT  14.85927
```

1.5(Q1)

```
# Group by Species and calculate the summary quantities
summary_table <- hawksFullName %>%
  group_by(Species) %>%
  summarize(
    num_rows = n(),                              # Number of rows
    mn_wing = mean(Wing, na.rm = TRUE),          # Mean wing span
    md_wing = median(Wing, na.rm = TRUE),        # Median wing span
    t_mn_wing = mean(Wing, trim = 0.1, na.rm = TRUE),  # Trimmed mean wing span
    b_wt_ratio = max(Wing / Weight, na.rm = TRUE)  # Biggest ratio between wing span and weight
  )

# Print the summary table
print(summary_table)
```

```
## # A tibble: 3 x 6
##   Species       num_rows mn_wing md_wing t_mn_wing b_wt_ratio
##   <chr>            <int>   <dbl>   <dbl>     <dbl>      <dbl>
## 1 Cooper's            70    244.     240      243.       4.79
## 2 Red-tailed         577    383.     384      385.       4.11
## 3 Sharp-shinned      261    185.     191      184.       2.05
```

1.5(Q2)

```r
# Define the columns you want to analyze
selected_columns <- c("Wing", "Weight", "Culmen", "Hallux", "Tail", "StandardTail", "Tarsus", "Crop")

# Group the data by Hawk species and count missing values for selected columns
missing_summary_table <- Hawks %>%
  group_by(Species) %>%
  summarize(
    across(
      all_of(selected_columns),
      list(missing_count = ~sum(is.na(.)))
    ),
    .groups = "drop"
  )

# Print the missing value summary table
print(missing_summary_table)
```

```
## # A tibble: 3 x 9
##   Species Wing_missing_count Weight_missing_count Culmen_missing_count
##   <fct>                <int>                <int>                <int>
## 1 CH                       1                    0                    0
## 2 RT                       0                    5                    4
## 3 SS                       0                    5                    3
## # i 5 more variables: Hallux_missing_count <int>, Tail_missing_count <int>,
## #   StandardTail_missing_count <int>, Tarsus_missing_count <int>,
## #   Crop_missing_count <int>
```

2.1(Q1)

library(dplyr) library(purrr) ## 2.1(Q2)

```r
impute_by_median <- function(x) {
  median_x <- median(x, na.rm = TRUE)  # Compute the median of x
  impute_f <- function(z) {  # Coordinate-wise imputation
    if (is.na(z)) {
      return(median_x)  # If z is NA, replace it with the median
    } else {
      return(z)  # Otherwise, leave it in place
    }
  }
  return(map_dbl(x, impute_f))  # Apply the map function to impute across the vector
}
v<-c(1,2,NA,4)
impute_by_median(v)
```

```
## [1] 1 2 2 4
```

2.1(Q3)

```
n <- 101  # Number of data points
x <- seq(0, 10, by = 0.1)  # Generate the sequence for x
y <- 5 * x + 1  # Generate the sequence for y based on the given formula
df_xy <- data.frame(x, y)  # Create the data frame
df_xy %>% head(5)
```

```
##     x   y
## 1 0.0 1.0
## 2 0.1 1.5
## 3 0.2 2.0
## 4 0.3 2.5
## 5 0.4 3.0
```

2.1(Q4)

```
# Function to generate missing values based on index
sometimes_missing <- function(index, value) {
  if (index %% 5 == 0) {
    return(NA)
  } else {
    return(value)
  }
}

# Generate the data frame with missing data
df_xy_missing <- data.frame(
  x = seq(0, 10, by = 0.1),
  y = 5 * seq(0, 10, by = 0.1) + 1
)

# Apply the sometimes_missing function to create missing values in y
df_xy_missing$y <- mapply(sometimes_missing, 1:nrow(df_xy_missing), df_xy_missing$y)

# Check the first ten rows of the data frame
head(df_xy_missing, 10)
```

```
##      x   y
## 1  0.0 1.0
## 2  0.1 1.5
## 3  0.2 2.0
## 4  0.3 2.5
## 5  0.4  NA
## 6  0.5 3.5
## 7  0.6 4.0
## 8  0.7 4.5
## 9  0.8 5.0
## 10 0.9  NA
```

2.1(Q5)

```
# Function to impute missing values with the median
impute_by_median <- function(x) {
  median_x <- median(x, na.rm = TRUE)
  return(ifelse(is.na(x), median_x, x))
}

# Create df_xy_imputed by applying impute_by_median to the y column of df_xy_missing
df_xy_imputed <- df_xy_missing %>%
  mutate(y = impute_by_median(y))

# Check the first few rows of the df_xy_imputed data frame
head(df_xy_imputed)
```

```
##     x    y
## 1 0.0  1.0
## 2 0.1  1.5
## 3 0.2  2.0
## 4 0.3  2.5
## 5 0.4 26.0
## 6 0.5  3.5
```

2.2

```
library(readxl)
library(dplyr)
library(tidyr)

# Read the "Wins" sheet from the Excel file
file_path <- "C:/Users/dell/Desktop/Bristol/RStudio/RStudio/week2/HockeyLeague.xlsx"
wins_data_frame <- read_excel(file_path, sheet = "Wins")

# Transform the data into a tidy format
wins_tidy <- wins_data_frame %>%
  pivot_longer(cols = -Team, names_to = "Year", values_to = "Wins_Total") %>%
  separate(Wins_Total, into = c("Wins", "Total"), sep = " of ") %>%
  mutate(Year = as.integer(Year),
         Wins = as.integer(Wins),
         Total = as.integer(Total))

# Check dimensions and inspect the top 5 rows
print(dim(wins_tidy))
```

```
## [1] 248   4
```

```
print(head(wins_tidy, 5))
```

```
## # A tibble: 5 x 4
##   Team   Year  Wins Total
##   <chr> <int> <int> <int>
## 1 Ducks  1990    30    50
## 2 Ducks  1991    11    50
## 3 Ducks  1992    30    50
## 4 Ducks  1993    12    50
## 5 Ducks  1994    24    50
```

2.2(Q1)

```
# Read the "Losses" sheet from the Excel file
losses_data_frame <- read_excel(file_path, sheet = "Losses")

# Transform the data into a tidy format
losses_tidy <- losses_data_frame %>%
  pivot_longer(cols = -Team, names_to = "Year", values_to = "Losses_Total") %>%
  separate(Losses_Total, into = c("Losses", "Total"), sep = " of ") %>%
  mutate(Year = as.integer(Year),
         Losses = as.integer(Losses),
         Total = as.integer(Total))

# Check dimensions and inspect the top 5 rows
print(dim(losses_tidy))
```

```
## [1] 248   4
```

```
print(head(losses_tidy, 5))
```

```
## # A tibble: 5 x 4
##   Team   Year Losses Total
##   <chr> <int>  <int> <int>
## 1 Ducks  1990     20    50
## 2 Ducks  1991     37    50
## 3 Ducks  1992      1    50
## 4 Ducks  1993     30    50
## 5 Ducks  1994      7    50
```

```
# Set your folder path and file name
folder_path <- "C:/Users/dell/Desktop/Bristol/RStudio/RStudio/week2/"
file_name <- "HockeyLeague.xlsx"
file_path <- paste(folder_path, file_name, sep="")

# Read the "Losses" sheet from the Excel file
losses_data_frame <- read_excel(file_path, sheet = "Losses")

# Rename the columns to remove spaces and make them more descriptive
losses_data_frame <- losses_data_frame %>%
  rename_with(~gsub("\\s", "_", .), -Team)

# Pivot the data to long format
losses_tidy <- losses_data_frame %>%
  pivot_longer(cols = -Team, names_to = "Year", values_to = "Losses_Total")

# Separate the "Losses_Total" column into "Losses" and "Total" columns
losses_tidy <- losses_tidy %>%
  separate(Losses_Total, into = c("Losses", "Total"), sep = " of ", convert = TRUE)

# Change the data types of columns
losses_tidy <- losses_tidy %>%
  mutate(
    Team = as.character(Team),
    Year = as.integer(Year),
    Losses = as.integer(Losses),
    Total = as.integer(Total)
  )

# Check the dimensions and first five rows of the tidy data frame
dim(losses_tidy)
```

```
## [1] 248   4
```

```
head(losses_tidy, 5)
```

```
## # A tibble: 5 x 4
##   Team   Year Losses Total
##   <chr> <int>  <int> <int>
## 1 Ducks  1990     20    50
## 2 Ducks  1991     37    50
## 3 Ducks  1992      1    50
## 4 Ducks  1993     30    50
## 5 Ducks  1994      7    50
```

2.2(Q3)

```
# Combine wins_tidy and losses_tidy
hockey_df <- wins_tidy %>%
  inner_join(losses_tidy, by = c("Team", "Year", "Total")) %>%
  mutate(Draws = Total - Wins - Losses,
         Wins_rt = Wins / Total,
         Losses_rt = Losses / Total,
         Draws_rt = Draws / Total) %>%
  select(Team, Year, Wins, Total, Losses, Draws, Wins_rt, Losses_rt, Draws_rt)

# Display the top 5 rows
print(head(hockey_df, 5))
```

```
## # A tibble: 5 x 9
##   Team   Year  Wins Total Losses Draws Wins_rt Losses_rt Draws_rt
##   <chr> <int> <int> <int>  <int> <int>   <dbl>     <dbl>    <dbl>
## 1 Ducks  1990    30    50     20     0    0.6       0.4      0
## 2 Ducks  1991    11    50     37     2    0.22      0.74     0.04
## 3 Ducks  1992    30    50      1     19    0.6        0.02       0.38
## 4 Ducks  1993    12    50     30      8    0.24       0.6        0.16
## 5 Ducks  1994    24    50      7     19    0.48       0.14       0.38
```

2.2(Q4)

```
summary_df <- hockey_df %>%
  group_by(Team) %>%
  summarise(W_md = round(median(Wins_rt), 3),
            W_mn = round(mean(Wins_rt), 3),
            L_md = round(median(Losses_rt), 3),
            L_mn = round(mean(Losses_rt), 3),
            D_md = round(median(Draws_rt), 3),
            D_mn = round(mean(Draws_rt), 3)) %>%
  arrange(desc(W_md))

# Display the summary data frame
print(summary_df)
```

```
## # A tibble: 8 x 7
##   Team         W_md  W_mn  L_md  L_mn  D_md  D_mn
##   <chr>       <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
## 1 Eagles      0.45  0.437 0.25  0.279 0.317 0.284
## 2 Penguins    0.45  0.457 0.3   0.31  0.133 0.232
## 3 Hawks       0.417 0.388 0.233 0.246 0.32  0.366
## 4 Ducks       0.383 0.362 0.34  0.333 0.25  0.305
## 5 Owls        0.32  0.333 0.3   0.33  0.383 0.337
## 6 Ostriches   0.3   0.309 0.4   0.395 0.267 0.296
## 7 Storks      0.3   0.284 0.22  0.283 0.48  0.433
## 8 Kingfishers 0.233 0.245 0.34  0.36  0.4   0.395
```

3 (Q1)

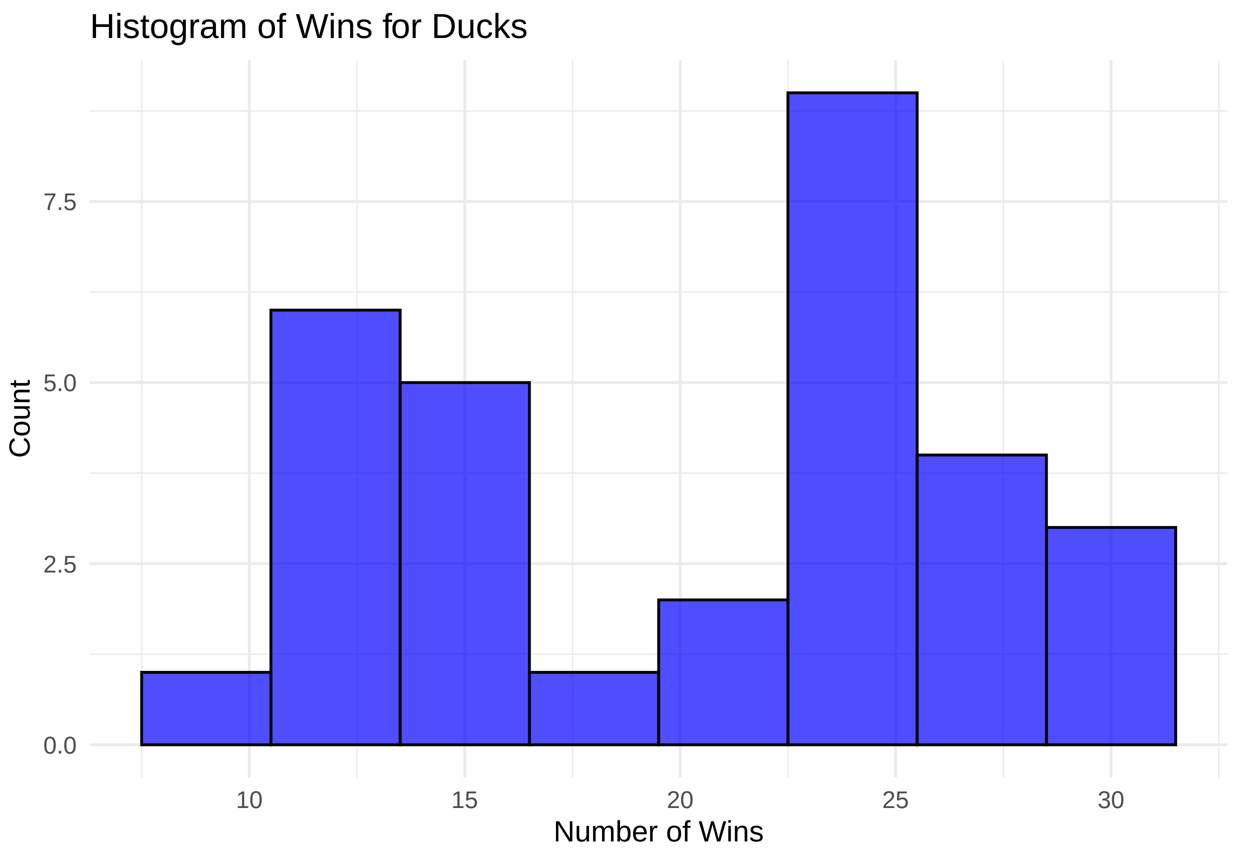
```
# Load the ggplot2 library
install.packages("ggplot2")
```

```
## Warning: package 'ggplot2' is in use and will not be installed
```

```
library(ggplot2)

# Filter the data for Ducks
ducks_data <- wins_tidy %>% filter(Team == "Ducks")

# Create the histogram
ggplot(ducks_data, aes(x = Wins)) +
  geom_histogram(binwidth = 3, fill = "blue", color = "black", alpha = 0.7) +
  labs(title = "Histogram of Wins for Ducks",
       x = "Number of Wins",
       y = "Count") +
  theme_minimal()
```

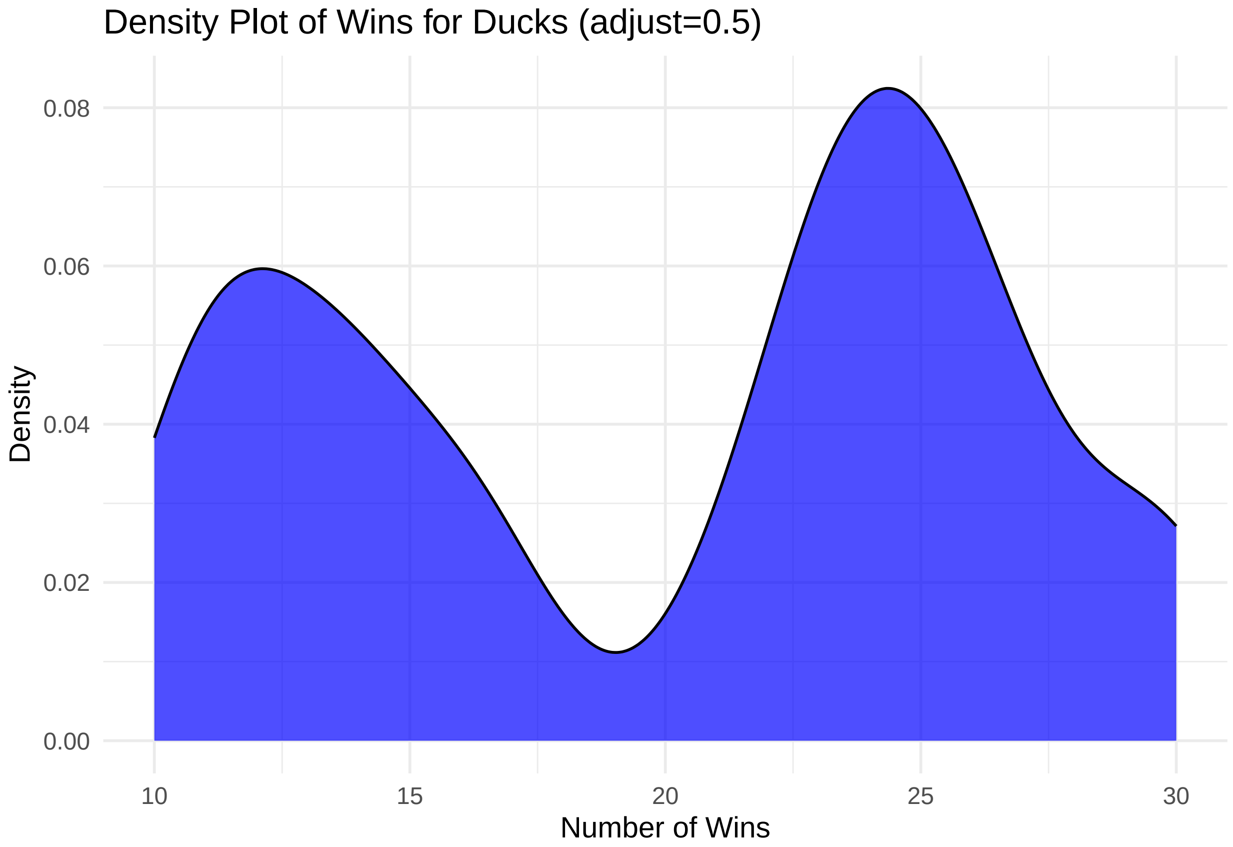


3 (Q2)

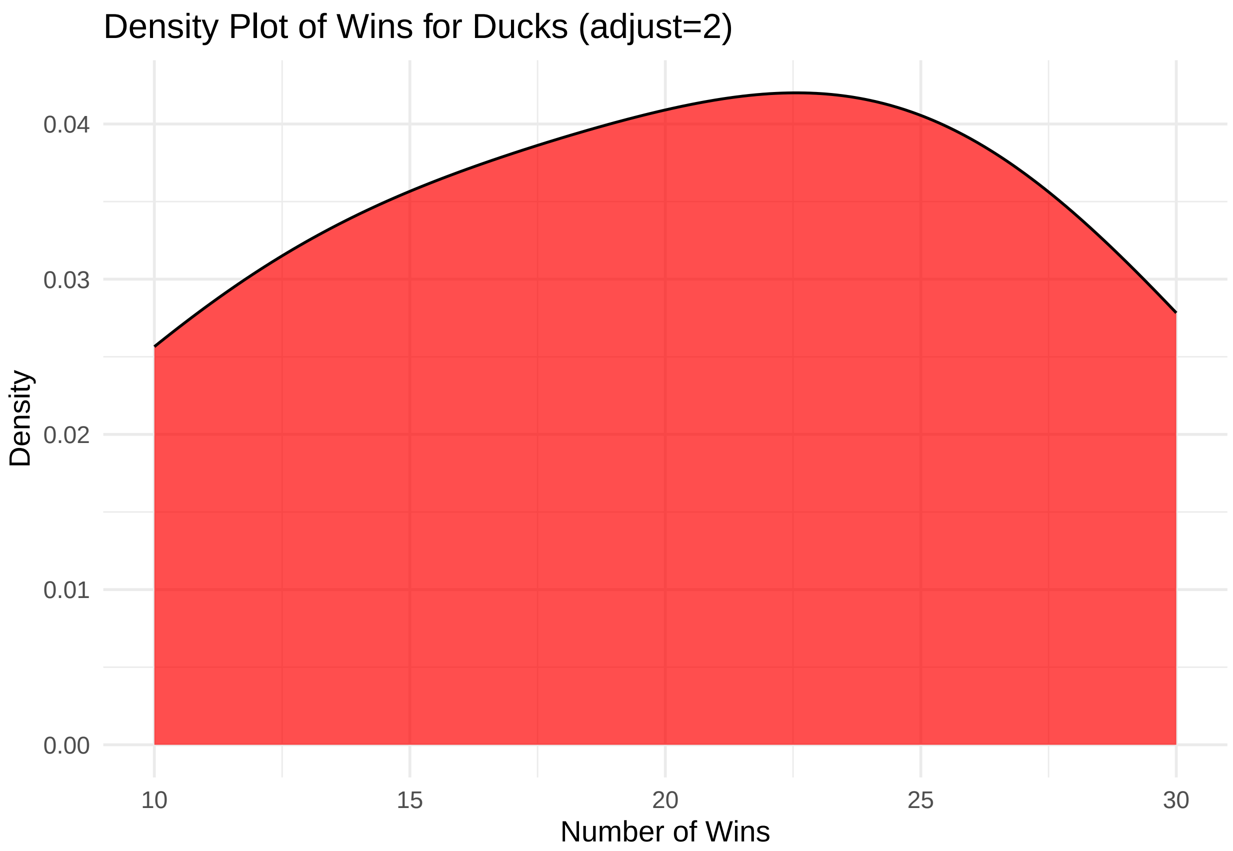
```r
# Density plot with adjust = 0.5
p1 <- ggplot(ducks_data, aes(x = Wins)) +
  geom_density(adjust = 0.5, fill = "blue", alpha = 0.7) +
  labs(title = "Density Plot of Wins for Ducks (adjust=0.5)",
       x = "Number of Wins",
       y = "Density") +
  theme_minimal()

# Density plot with adjust = 2
p2 <- ggplot(ducks_data, aes(x = Wins)) +
  geom_density(adjust = 2, fill = "red", alpha = 0.7) +
  labs(title = "Density Plot of Wins for Ducks (adjust=2)",
       x = "Number of Wins",
       y = "Density") +
  theme_minimal()

# Display the plots
print(p1)
```


Density Plot of Wins for Ducks (adjust=0.5)
Density
0.08
0.06
0.04
0.02
0.00
10
15
20
25
30
Number of Wins

```
print(p2)
```



3 (Q3)

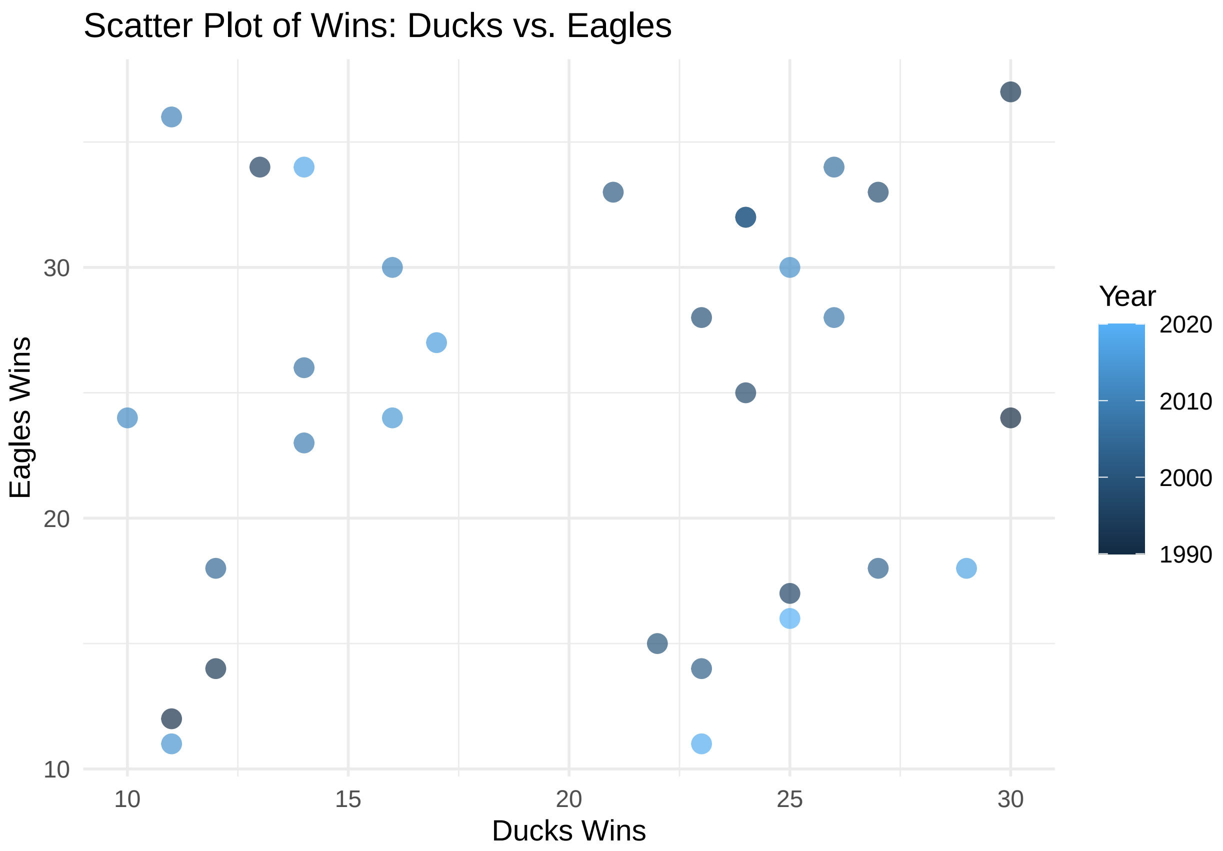
```
# Reshape the wins_tidy dataframe
wins_teams <- wins_tidy %>%
  select(Year, Team, Wins) %>%
  pivot_wider(names_from = Team, values_from = Wins)

# Display the first 10 rows of wins_teams
print(head(wins_teams, 10))
```

```
## # A tibble: 10 x 9
##     Year Ducks Eagles Hawks Kingfishers Ostriches  Owls Penguins Storks
##    <int> <int>  <int> <int>       <int>     <int> <int>    <int>  <int>
##  1  1990    30     24    20          16        13    19       23     20
##  2  1991    11     12    22          19        13    13       29     13
##  3  1992    30     37    33          12        10    18       30     18
##  4  1993    12     14    11          10        25    16       32     22
##  5  1994    24     32    20          17        10    13       33     19
##  6  1995    13     34    18          11        21    24       36     11
##  7  1996    25     17    21          11        13    24       12     15
##  8  1997    24     25    23          12        18    10       16     15
##  9  1998    27     33    18          11        24    20       34     14
## 10  1999    23     28    28          19        18    21       20     13
```

```r
# Create the scatter plot
scatter_plot <- ggplot(wins_teams, aes(x = Ducks, y = Eagles)) +
  geom_point(aes(color = Year), size = 3, alpha = 0.7) +
  labs(title = "Scatter Plot of Wins: Ducks vs. Eagles",
       x = "Ducks Wins",
       y = "Eagles Wins") +
  theme_minimal() +
  scale_color_continuous(name = "Year")

# Display the scatter plot
print(scatter_plot)
```


Scatter Plot of Wins: Ducks vs. Eagles
Eagles Wins
Ducks Wins
Year