CSP571 Code

Juveriya

2024-04-17

CTA BUS ROUTE DATASET

```
# Import the required libraries
library(dplyr)
```

```
##
## 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(ggplot2)

# Load the Bus Route dataset

busRoute <- read.csv("CTA_Bus_Routes.csv")
summary(busRoute)
```

```
##     route             routename         Month_Beginning    Avg_Weekday_Rides 
##  Length:37238       Length:37238       Length:37238       Length:37238      
##  Class :character   Class :character   Class :character   Class :character  
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character  
##  Avg_Saturday_Rides Avg_Sunday.Holiday_Rides  MonthTotal       
##  Length:37238       Length:37238             Length:37238      
##  Class :character   Class :character         Class :character  
##  Mode  :character   Mode  :character         Mode  :character  
```

```
# Print column names
print(names(busRoute))
```

```
## [1] "route"                   "routename"
## [3] "Month_Beginning"         "Avg_Weekday_Rides"
## [5] "Avg_Saturday_Rides"      "Avg_Sunday.Holiday_Rides"
## [7] "MonthTotal"
```

Data Cleaning

```
# Checking the structure
str(busRoute)
```

```
## 'data.frame':    37238 obs. of  7 variables:
##  $ route                   : chr  "1" "2" "3" "4" ...
##  $ routename               : chr  "Indiana/Hyde Park" "Hyde Park Express" "King Drive" "Cotta
ge Grove" ...
##  $ Month_Beginning         : chr  "01/01/2001" "01/01/2001" "01/01/2001" "01/01/2001" ...
##  $ Avg_Weekday_Rides       : chr  "6,982.6" "1,000" "21,406.5" "22,432.2" ...
##  $ Avg_Saturday_Rides      : chr  "0" "0" "13,210.7" "17,994" ...
##  $ Avg_Sunday.Holiday_Rides: chr  "0" "0" "8,725.3" "10,662.2" ...
##  $ MonthTotal              : chr  "153,617" "22,001" "567,413" "618,796" ...
```

```
busRoute$Month_Beginning <- as.Date(busRoute$Month_Beginning, format = "%m/%d/%Y")
years <- unique(format(busRoute$Month_Beginning, "%Y"))
num_years <- length(years)
print(num_years)
```

```
## [1] 23
```

```
cat("List of years:", paste(years, collapse = ", "))
```

```
## List of years: 2001, 2003, 2017, 2004, 2020, 2021, 2002, 2022, 2009, 2005, 2011, 2006, 2010,
2012, 2007, 2008, 2014, 2013, 2015, 2016, 2018, 2019, 2023
```

```
# Filter the data for the years 2013 to 2023
busRoute_data <- busRoute[format(busRoute$Month_Beginning, "%Y") %in% c("2013", "2014", "2015",
"2016", "2017", "2018", "2019", "2020", "2021", "2022", "2023"), ]
print(head(busRoute_data))
```

```
##       route                 routename Month_Beginning Avg_Weekday_Rides
## 892      31                      31st      2017-02-01             669.2
## 1167    992                 ROAD CALL      2020-06-01               0.8
## 1510     96                      Lunt      2021-08-01               450
## 1716     96                      Lunt      2022-08-01             492.3
## 16207    48               South Damen      2014-04-01             1,129
## 20638     1 Bronzeville/Union Station      2013-01-01           2,069.4
##       Avg_Saturday_Rides Avg_Sunday.Holiday_Rides MonthTotal
## 892                    0                        0     13,384
## 1167                   0                        0          3
## 1510                   0                        0      9,900
## 1716                   0                        0     11,323
## 16207                  0                        0     24,837
## 20638                  0                        0     45,526
```

```
# Convert numerical columns to numeric, removing commas in the process
busRoute_data$Avg_Weekday_Rides <- as.numeric(gsub(",", "", busRoute_data$Avg_Weekday_Rides))
busRoute_data$Avg_Saturday_Rides <- as.numeric(gsub(",", "", busRoute_data$Avg_Saturday_Rides))
busRoute_data$Avg_Sunday.Holiday_Rides <- as.numeric(gsub(",", "", busRoute_data$Avg_Sunday.Holi
day_Rides))
busRoute_data$MonthTotal <- as.numeric(gsub(",", "", busRoute_data$MonthTotal))
```

```
# Check the number of rows and columns in the dataset
dim(busRoute_data)
```

```
## [1] 16606     7
```

```
# Check for missing values in the entire dataset
missing_values <- colSums(is.na(busRoute_data))
print(missing_values)
```

```
##                    route                routename          Month_Beginning
##                        0                        0                        0
##        Avg_Weekday_Rides       Avg_Saturday_Rides Avg_Sunday.Holiday_Rides
##                        0                        0                        0
##               MonthTotal
##                        0
```

```
# Check for duplicate rows
duplicate_rows <- busRoute_data[duplicated(busRoute_data), ]
print(duplicate_rows)
```

```
## [1] route                    routename                Month_Beginning
## [4] Avg_Weekday_Rides        Avg_Saturday_Rides       Avg_Sunday.Holiday_Rides
## [7] MonthTotal
## <0 rows> (or 0-length row.names)
```

```
length(unique(busRoute_data$routename))
```

```
## [1] 152
```

```
#unique(busRoute_data$routename)
```

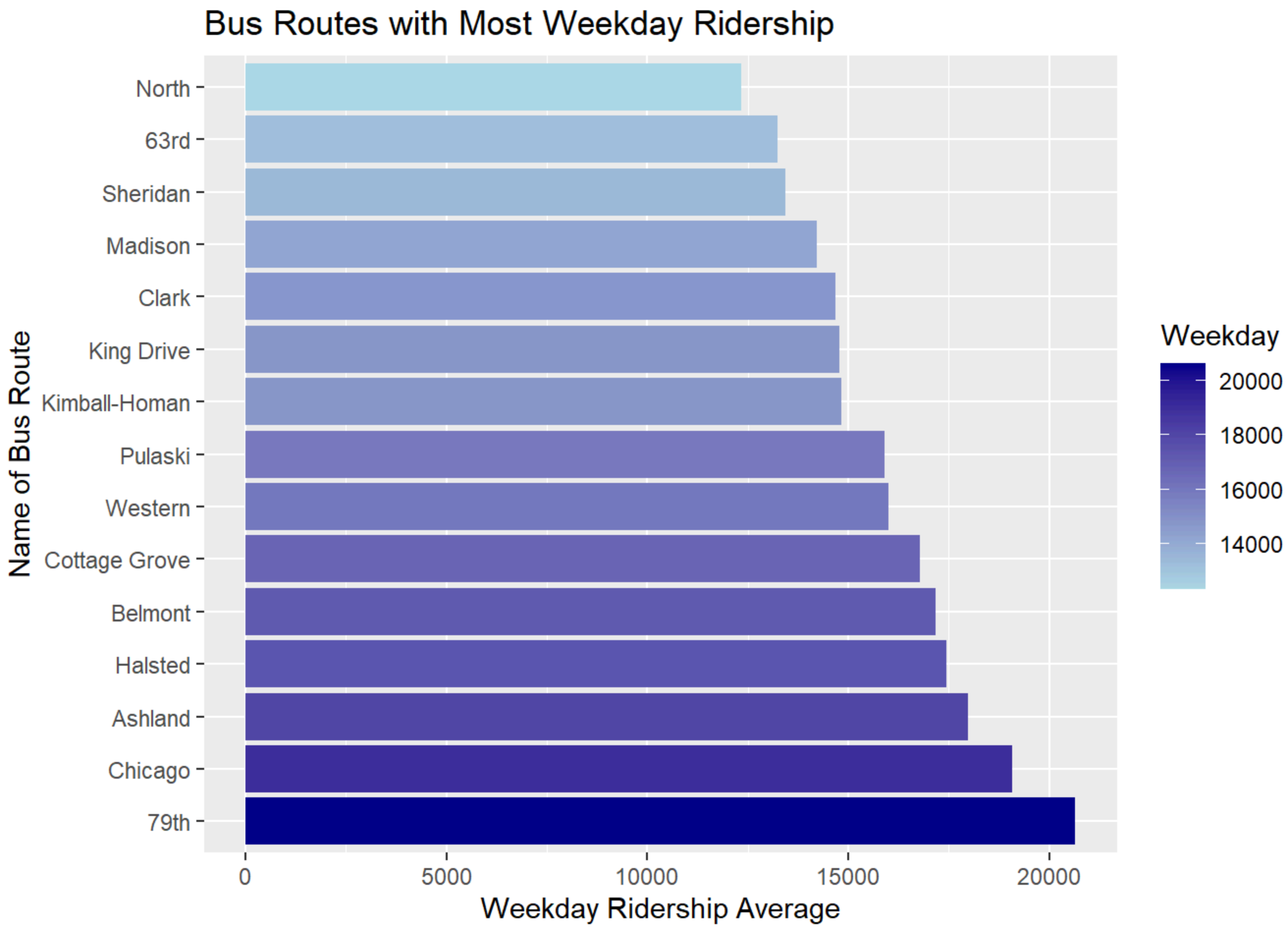
```
nrow(busRoute_data)
```

```
## [1] 16606
```

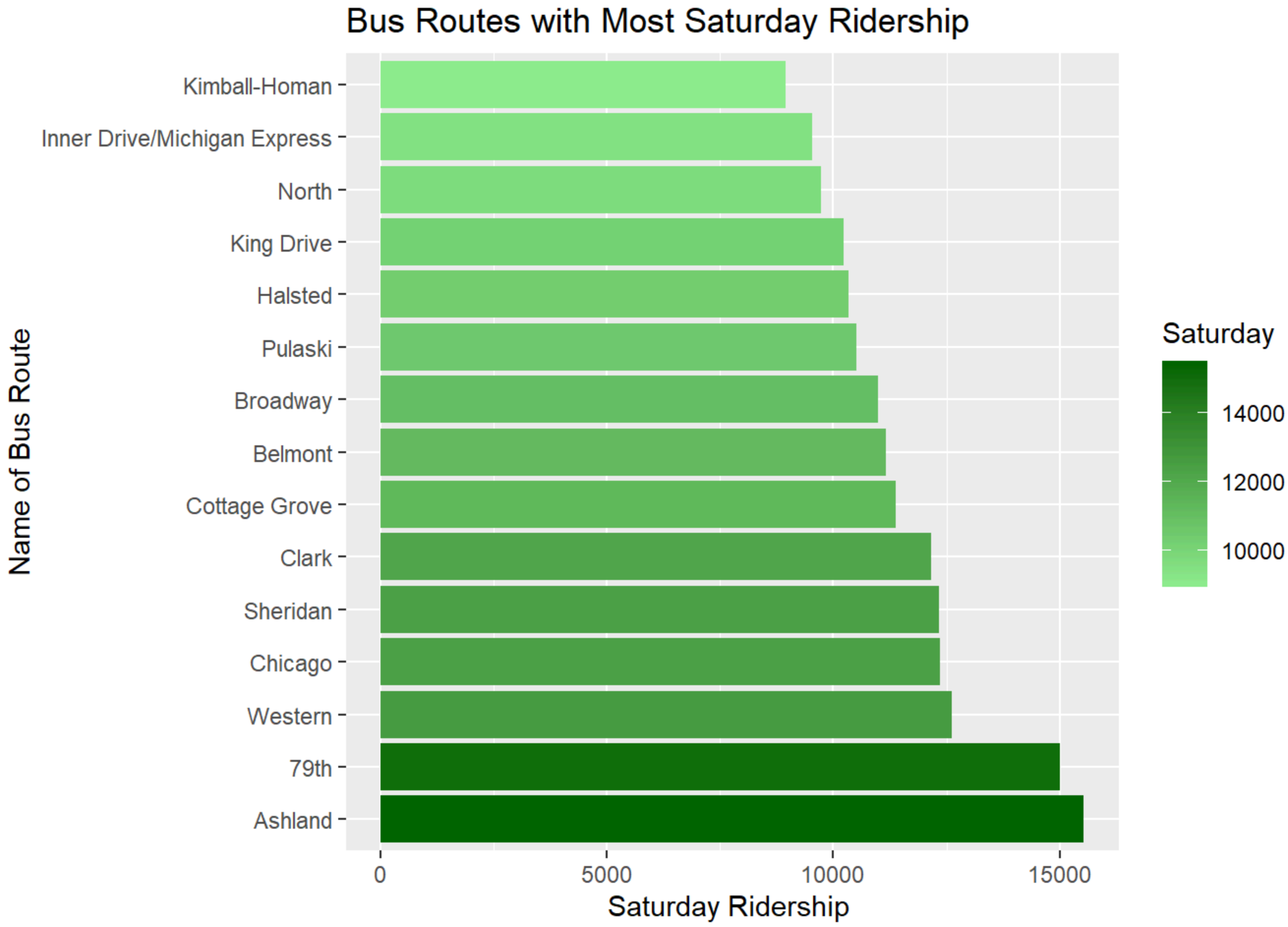
Exploratory Data Analysis

```
# Calculate average weekday, Saturday, and Sunday/holiday rides for each route
summary_df <- busRoute_data %>%
  group_by(routename) %>%
  summarize(Weekday = mean(Avg_Weekday_Rides),
            Saturday= mean(Avg_Saturday_Rides),
            Sunday_Holiday = mean(Avg_Sunday.Holiday_Rides))
```

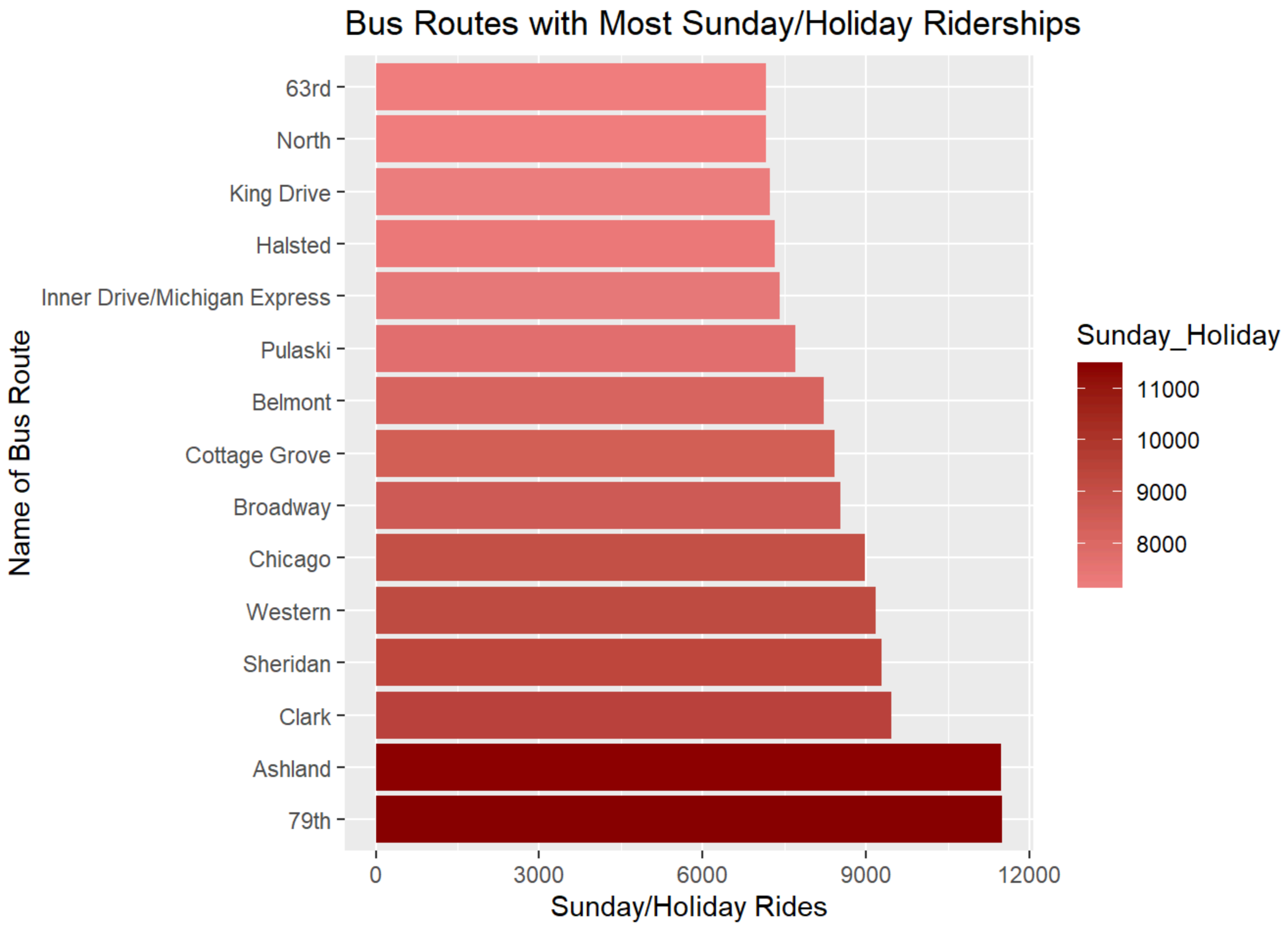
```
# Subset the sorted routes based on average weekday rides
top_15_weekday <- head(summary_df[order(summary_df$Weekday, decreasing = TRUE), ], n = 15)

# Create a horizontal bar plot for top 15 routes based on average weekday rides
ggplot(top_15_weekday, aes(x = Weekday, y = reorder(routename, -Weekday), fill = Weekday)) +
  geom_bar(stat = "identity") +
  labs(x = "Weekday Ridership Average", y = "Name of Bus Route") +
  ggtitle("Bus Routes with Most Weekday Ridership") +
  scale_fill_gradient(low = "lightblue", high = "darkblue") +  # Gradient color scale
  theme(axis.text.y = element_text(hjust = 1))
```



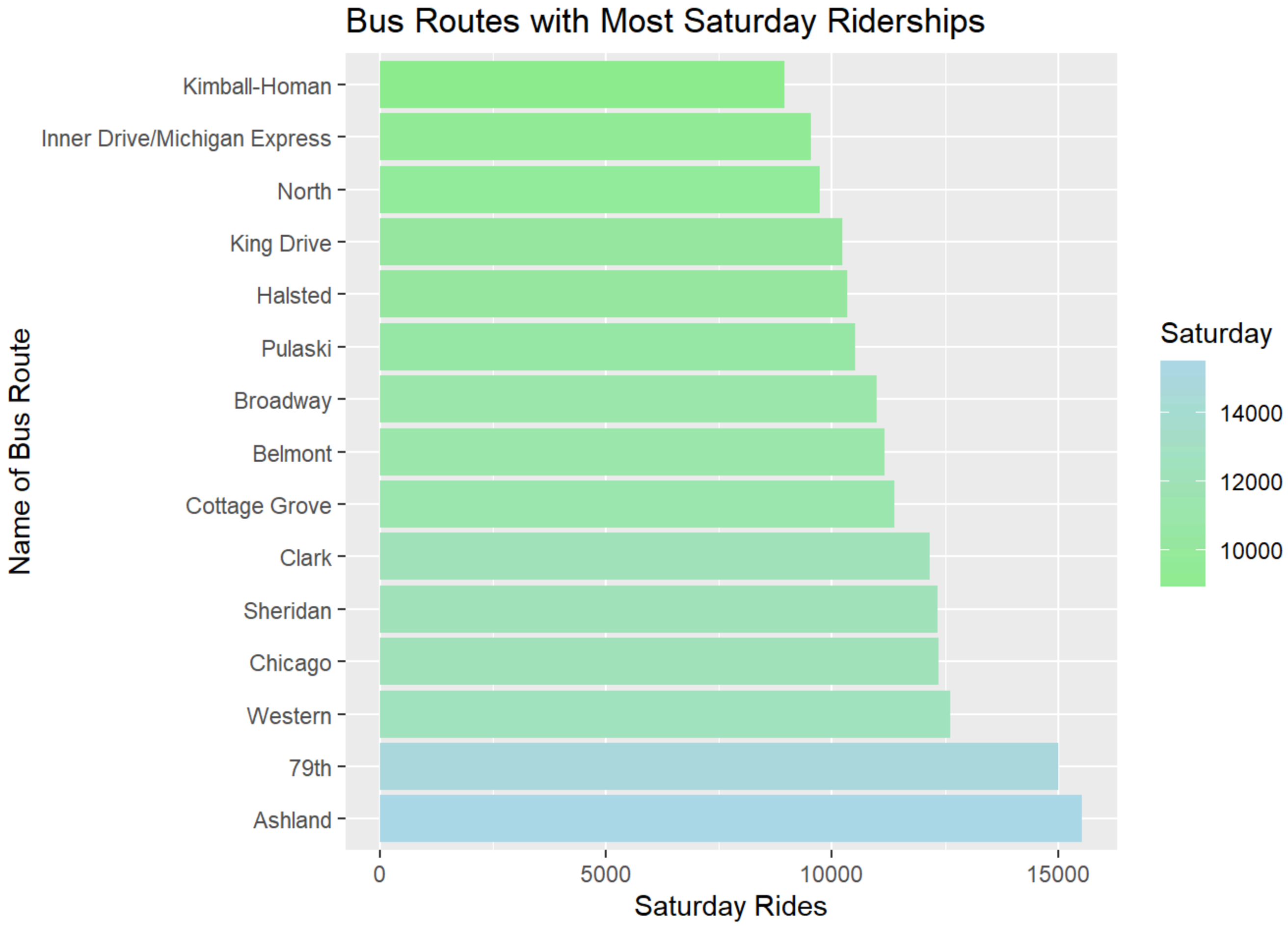
```
# Subset the sorted routes based on average Saturday rides
top_15_saturday <- head(summary_df[order(summary_df$Saturday, decreasing = TRUE), ], n = 15)
# Create a horizontal bar plot for top 15 routes based on average Saturday rides
ggplot(top_15_saturday, aes(x = Saturday, y = reorder(routename, -Saturday), fill = Saturday)) +
  geom_bar(stat = "identity") +
  labs(x = "Saturday Ridership ", y = "Name of Bus Route") +
  ggtitle("Bus Routes with Most Saturday Ridership") +
  scale_fill_gradient(low = "lightgreen", high = "darkgreen") +  # Gradient color scale
  theme(axis.text.y = element_text(hjust = 1))
```



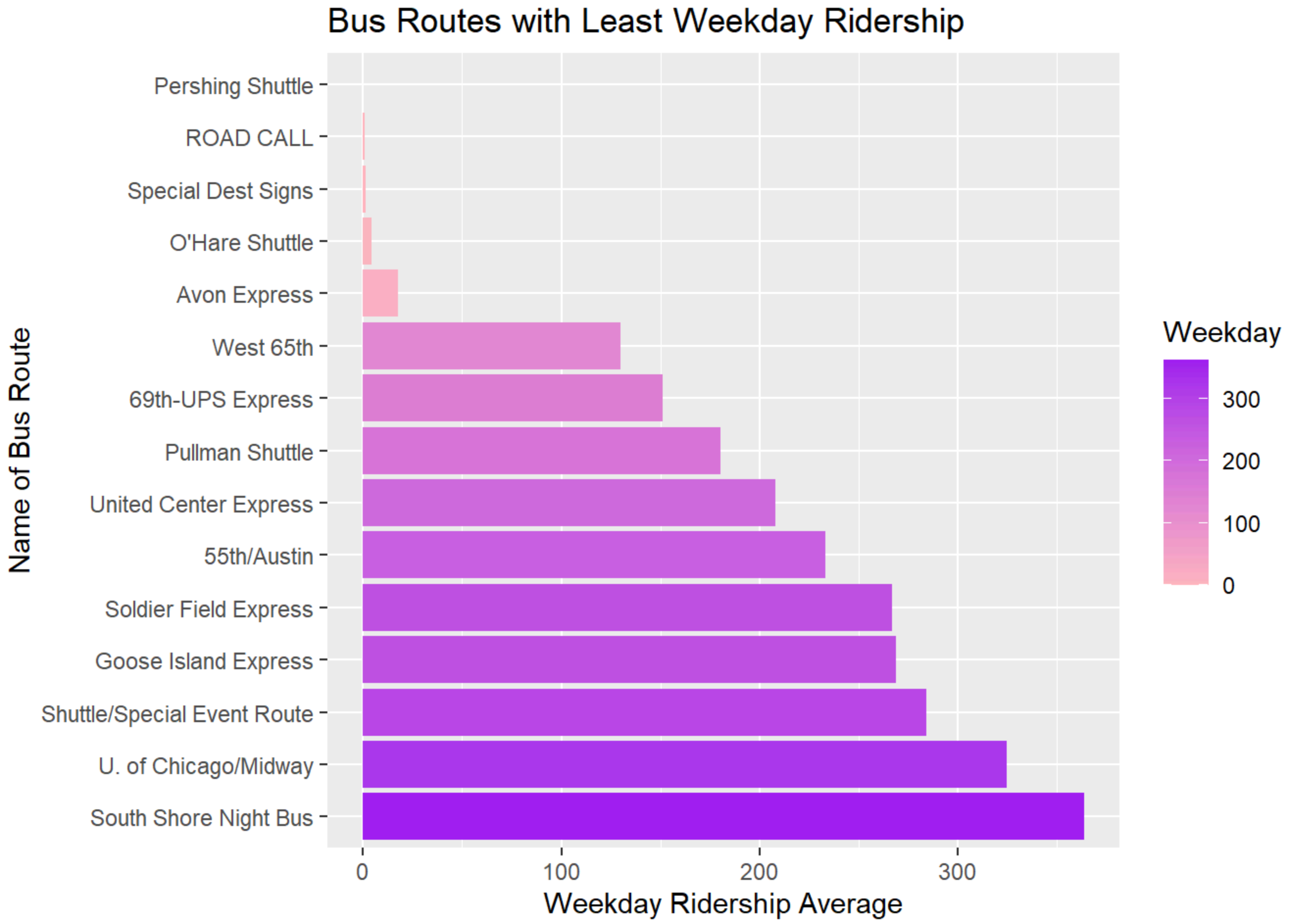
```
# Subset the sorted routes based on average Sunday/holiday rides
top_15_sunday_holiday <- head(summary_df[order(summary_df$Sunday_Holiday, decreasing = TRUE), ],
n = 15)

# Create a horizontal bar plot for top 15 routes based on average Sunday/holiday rides
ggplot(top_15_sunday_holiday, aes(x = Sunday_Holiday, y = reorder(routename, -Sunday_Holiday), f
ill = Sunday_Holiday)) +
  geom_bar(stat = "identity") +
  labs(x = "Sunday/Holiday Rides", y = "Name of Bus Route") +
  ggtitle("Bus Routes with Most Sunday/Holiday Riderships") +
  scale_fill_gradient(low = "lightcoral", high = "darkred") +  # Gradient color scale with diffe
rent colors
  theme(axis.text.y = element_text(hjust = 1))
```



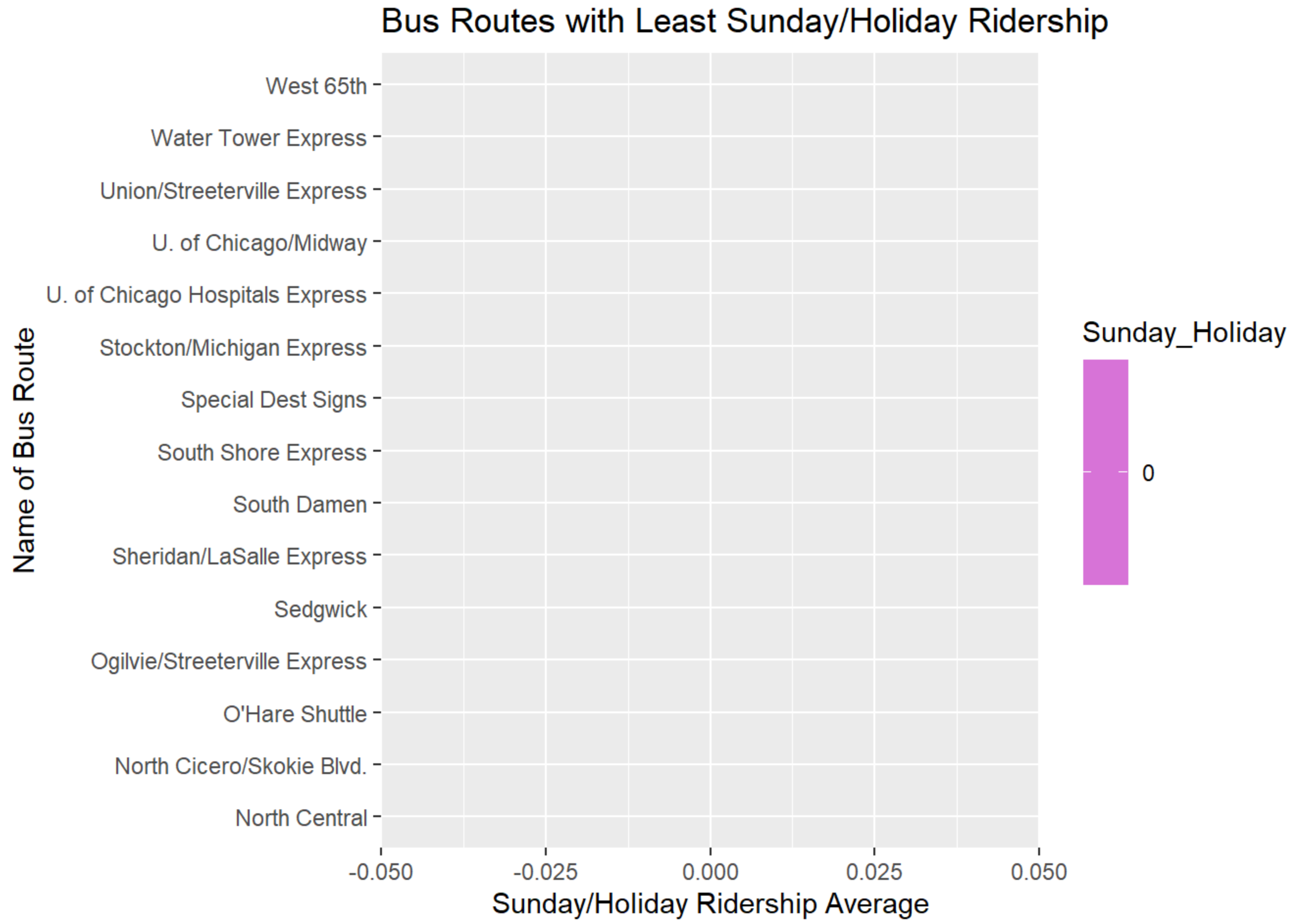
```
# Subset the sorted routes based on average Saturday rides
top_15_sunday_holiday <- head(summary_df[order(summary_df$Saturday, decreasing = TRUE), ], n = 15)

# Create a horizontal bar plot for top 15 routes based on average Sunday/holiday rides
ggplot(top_15_sunday_holiday, aes(x = Saturday, y = reorder(routename, -Saturday), fill = Saturday)) +
  geom_bar(stat = "identity") +
  labs(x = "Saturday Rides", y = "Name of Bus Route") +
  ggtitle("Bus Routes with Most Saturday Riderships") +
  scale_fill_gradient(low = "lightgreen", high = "lightblue") +  # Gradient color scale with different colors
  theme(axis.text.y = element_text(hjust = 1))
```



```
# Subset the sorted routes based on average weekday rides
bot_15_weekday <- tail(summary_df[order(summary_df$Weekday, decreasing = TRUE), ], n = 15)

# Create a horizontal bar plot for top 15 routes based on average weekday rides
ggplot(bot_15_weekday, aes(x = Weekday, y = reorder(routename, -Weekday), fill = Weekday)) +
  geom_bar(stat = "identity") +
  labs(x = "Weekday Ridership Average", y = "Name of Bus Route") +
  ggtitle("Bus Routes with Least Weekday Ridership") +
  scale_fill_gradient(low = "lightpink", high = "purple") +  # Gradient color scale
  theme(axis.text.y = element_text(hjust = 1))
```



```r
# Subset the sorted routes based on average sunday rides
bot_15_weekday <- tail(summary_df[order(summary_df$Sunday_Holiday, decreasing = TRUE), ], n = 15)

# Create a horizontal bar plot for top 15 routes based on average weekday rides
ggplot(bot_15_weekday, aes(x = Sunday_Holiday, y = reorder(routename, -Sunday_Holiday), fill = Sunday_Holiday)) +
  geom_bar(stat = "identity") +
  labs(x = "Sunday/Holiday Ridership Average", y = "Name of Bus Route") +
  ggtitle("Bus Routes with Least Sunday/Holiday Ridership") +
  scale_fill_gradient(low = "lightpink", high = "purple") +  # Gradient color scale
  theme(axis.text.y = element_text(hjust = 1))
```



##

CTA L Station Dataset

```
lstation_data <- read.csv("CTA_L_Station_Entries.csv", header = TRUE)
summary(lstation_data)
```

```
##    station_id     stationname        month_beginning    avg_weekday_rides 
##  Min.   :40010   Length:39053       Length:39053       Length:39053      
##  1st Qu.:40370   Class :character   Class :character   Class :character  
##  Median :40760   Mode  :character   Mode  :character   Mode  :character  
##  Mean   :40767                                                            
##  3rd Qu.:41160                                                            
##  Max.   :41700                                                            
##  avg_saturday_rides avg_sunday.holiday_rides  monthtotal       
##  Length:39053       Length:39053             Length:39053      
##  Class :character   Class :character         Class :character  
##  Mode  :character   Mode  :character         Mode  :character  
## 
## 
## 
```

Data Cleaning

```
print(names(lstation_data))
```

```
## [1] "station_id"               "stationname"             
## [3] "month_beginning"          "avg_weekday_rides"       
## [5] "avg_saturday_rides"       "avg_sunday.holiday_rides"
## [7] "monthtotal"
```

```
lstation_data$month_beginning <- as.Date(lstation_data$month_beginning, format = "%m/%d/%Y")
years <- unique(format(lstation_data$month_beginning, "%Y"))
num_years <- length(years)
print(num_years)
```

```
## [1] 23
```

```
cat("List of years:", paste(years, collapse = ", "))
```

```
## List of years: 2001, 2006, 2007, 2008, 2002, 2003, 2004, 2005, 2009, 2010, 2011, 2012, 2013,
2014, 2015, 2016, 2017, 2018, 2019, 2020, 2021, 2022, 2023
```

```
#Filter the data for the years 2013 to 2023
lstation <- lstation_data[format(lstation_data$month_beginning, "%Y") %in% c("2013", "2014", "20
15", "2016", "2017", "2018", "2019", "2020", "2021", "2022", "2023"), ]
print(tail(lstation))
```

```
##       station_id       stationname month_beginning avg_weekday_rides
## 39048      40160 LaSalle/Van Buren      2023-10-01           1,724.7
## 39049      40850           Library      2023-10-01           3,134.5
## 39050      40680      Adams/Wabash      2023-10-01           5,157.3
## 39051      41700 Washington/Wabash      2023-10-01           6,957.9
## 39052      40260        State/Lake      2023-10-01           8,378.3
## 39053      40380        Clark/Lake      2023-10-01           9,057.9
##       avg_saturday_rides avg_sunday.holiday_rides monthtotal
## 39048              591.8                    439.6     42,509
## 39049              2,256                  1,917.2     87,569
## 39050            2,901.8                    3,579    142,962
## 39051            5,125.5                  4,359.4    195,372
## 39052            7,134.8                  5,498.2    240,353
## 39053              5,088                    4,750    243,376
```

```
nrow(lstation)
```

```
## [1] 18639
```

```
str(lstation)
```

```
## 'data.frame':    18639 obs. of  7 variables:
##  $ station_id              : int  40900 41190 40100 41300 40760 40880 41380 40340 41200 40770
...
##  $ stationname             : chr  "Howard" "Jarvis" "Morse" "Loyola" ...
##  $ month_beginning         : Date, format: "2013-01-01" "2013-01-01" ...
##  $ avg_weekday_rides       : chr  "6,134.1" "1,295.2" "4,443.5" "4,641.5" ...
##  $ avg_saturday_rides      : chr  "4,252" "1,124.8" "3,314.8" "3,653.3" ...
##  $ avg_sunday.holiday_rides: chr  "2,928.6" "739.4" "2,462.6" "2,459" ...
##  $ monthtotal              : chr  "166,602" "36,690" "123,329" "129,021" ...
```

```
# Remove commas and convert numeric columns to numeric data types
lstation$avg_weekday_rides <- as.numeric(gsub(",", "", lstation$avg_weekday_rides))
lstation$avg_saturday_rides <- as.numeric(gsub(",", "", lstation$avg_saturday_rides))
lstation$avg_sunday.holiday_rides <- as.numeric(gsub(",", "", lstation$avg_sunday.holiday_ride
s))
lstation$monthtotal <- as.numeric(gsub(",", "", lstation$monthtotal))
```

```
# Check for missing values
missing_values <- colSums(is.na(lstation))
print(missing_values)
```

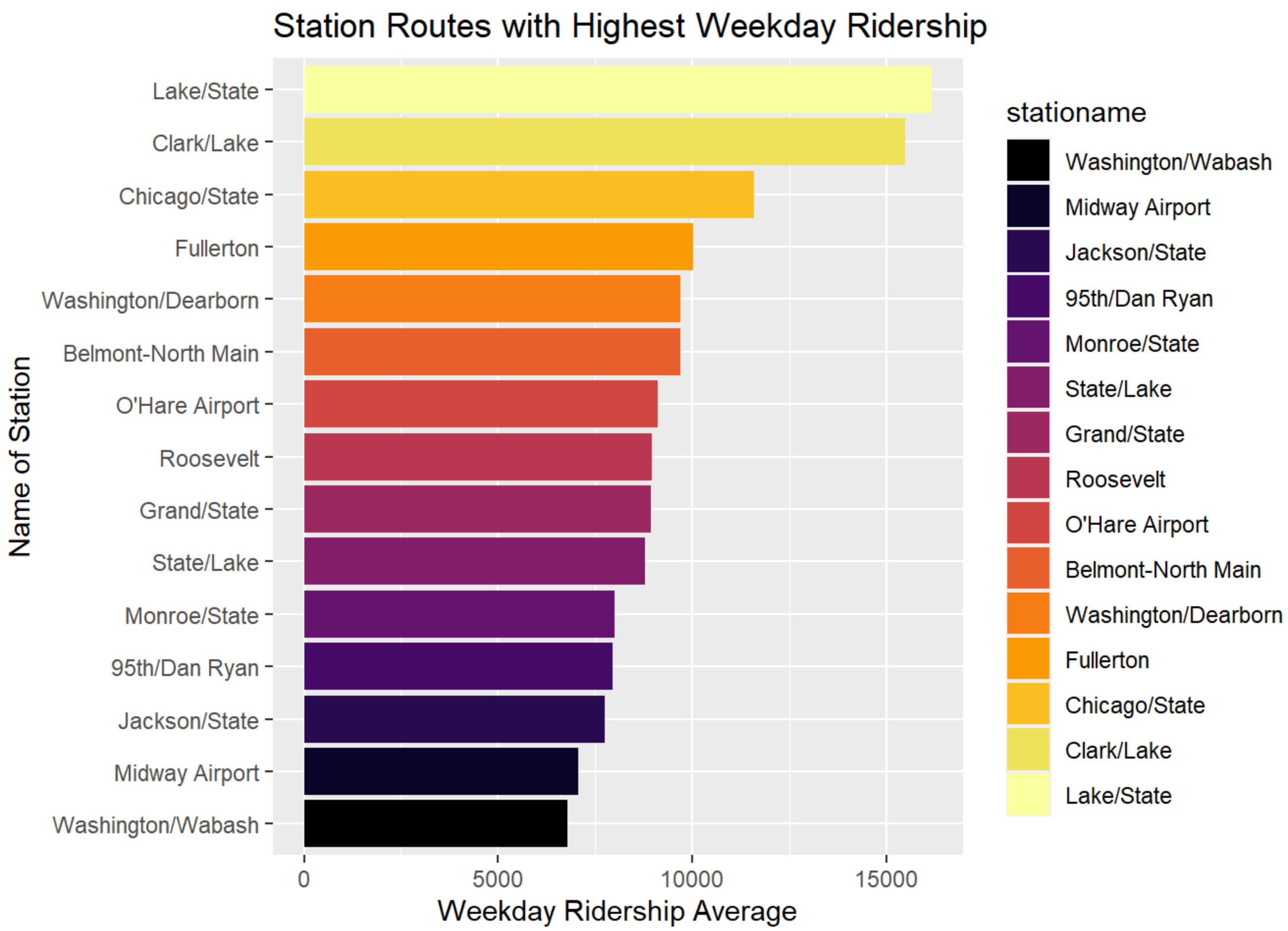
```
##               station_id              stationname          month_beginning 
##                        0                        0                        0 
##        avg_weekday_rides       avg_saturday_rides avg_sunday.holiday_rides 
##                        0                        0                        0 
##               monthtotal 
##                        0
```

```
# Check for duplicate rows
duplicate_rows <- lstation[duplicated(lstation), ]
print(duplicate_rows)
```

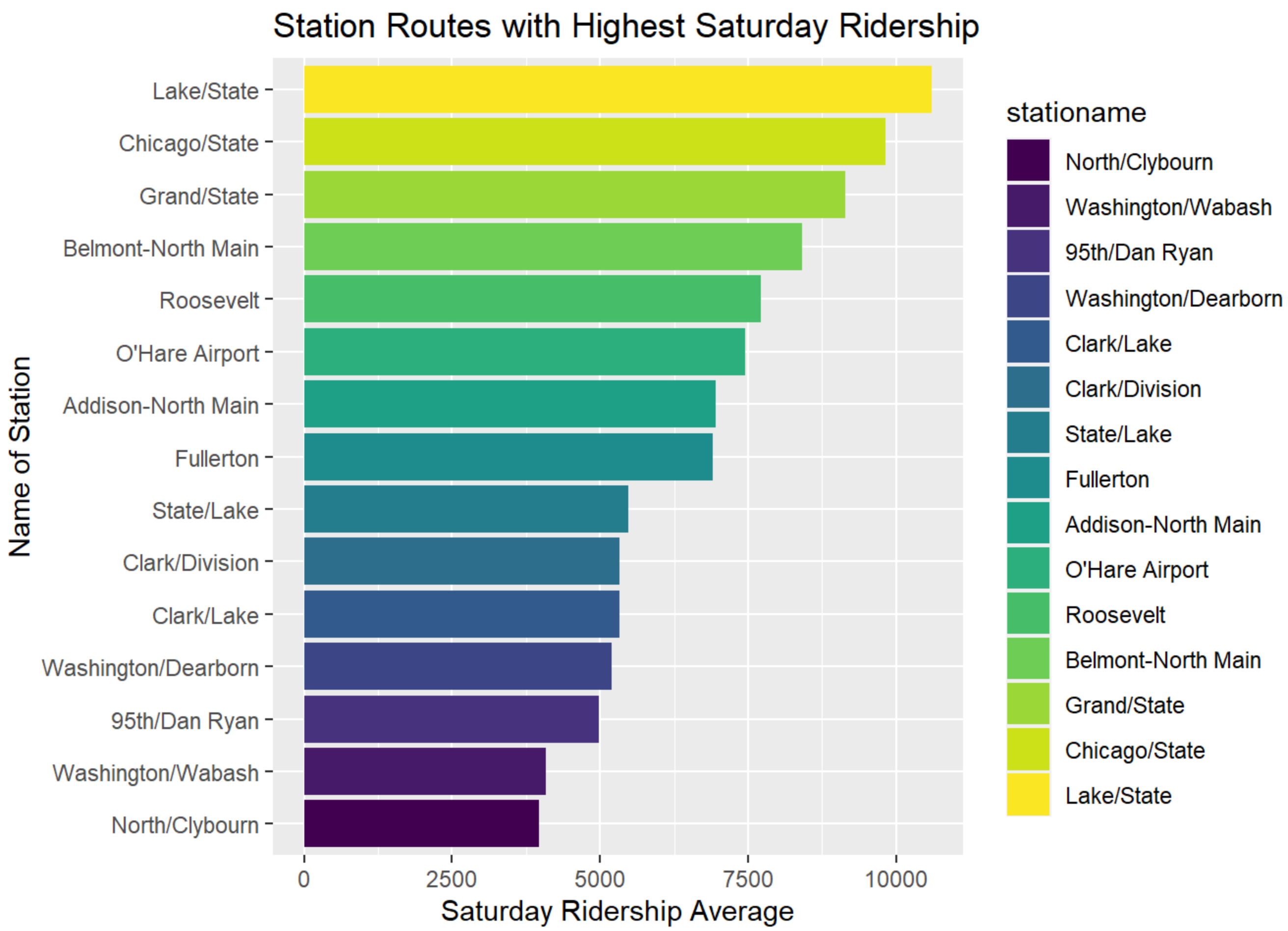
```
## [1] station_id               stationname              month_beginning         
## [4] avg_weekday_rides        avg_saturday_rides       avg_sunday.holiday_rides
## [7] monthtotal              
## <0 rows> (or 0-length row.names)
```

```
station_mean <- lstation %>%
  group_by(stationname) %>%
  summarize(avg_weekday = mean(avg_weekday_rides),
            avg_sat = mean(avg_saturday_rides),
            avg_sun = mean(avg_sunday.holiday_rides))
```

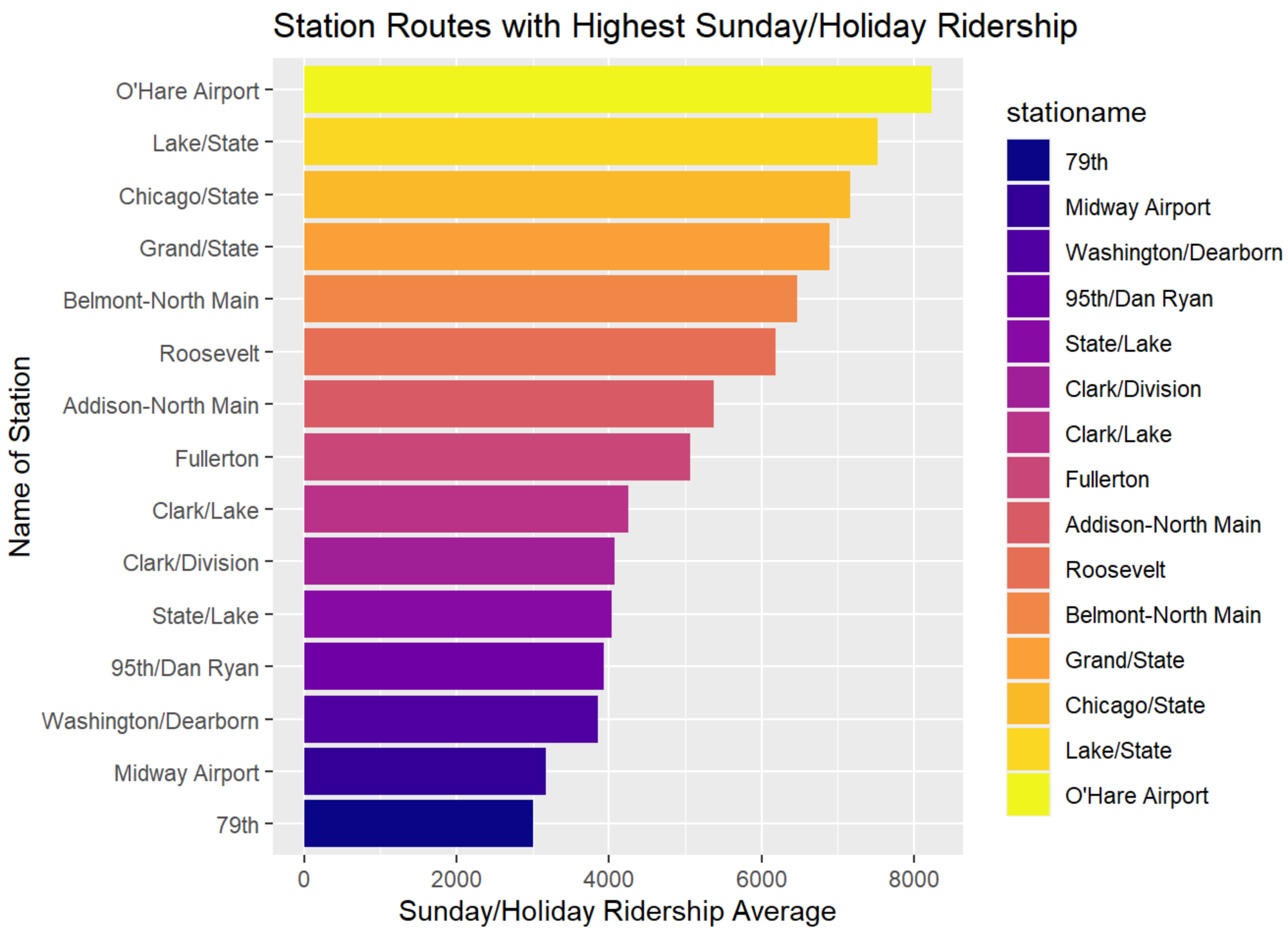
```
sum <- station_mean[order(station_mean$avg_weekday, decreasing = TRUE),]
# Find the top 15 stations on week day
sum <- head(sum, n = 15)
sum$stationame <- factor(sum$stationame, levels = sum$stationame[order(sum$avg_weekday)])
colors <- viridis::inferno(nrow(sum))

# Plot the graph
ggplot(sum, aes(x = avg_weekday, y = stationame, fill=stationame)) +
geom_bar(stat = "identity") +
labs(x = "Weekday Ridership Average", y = "Name of Station") +
ggtitle("Station Routes with Highest Weekday Ridership") +
scale_fill_manual(values = colors)
```



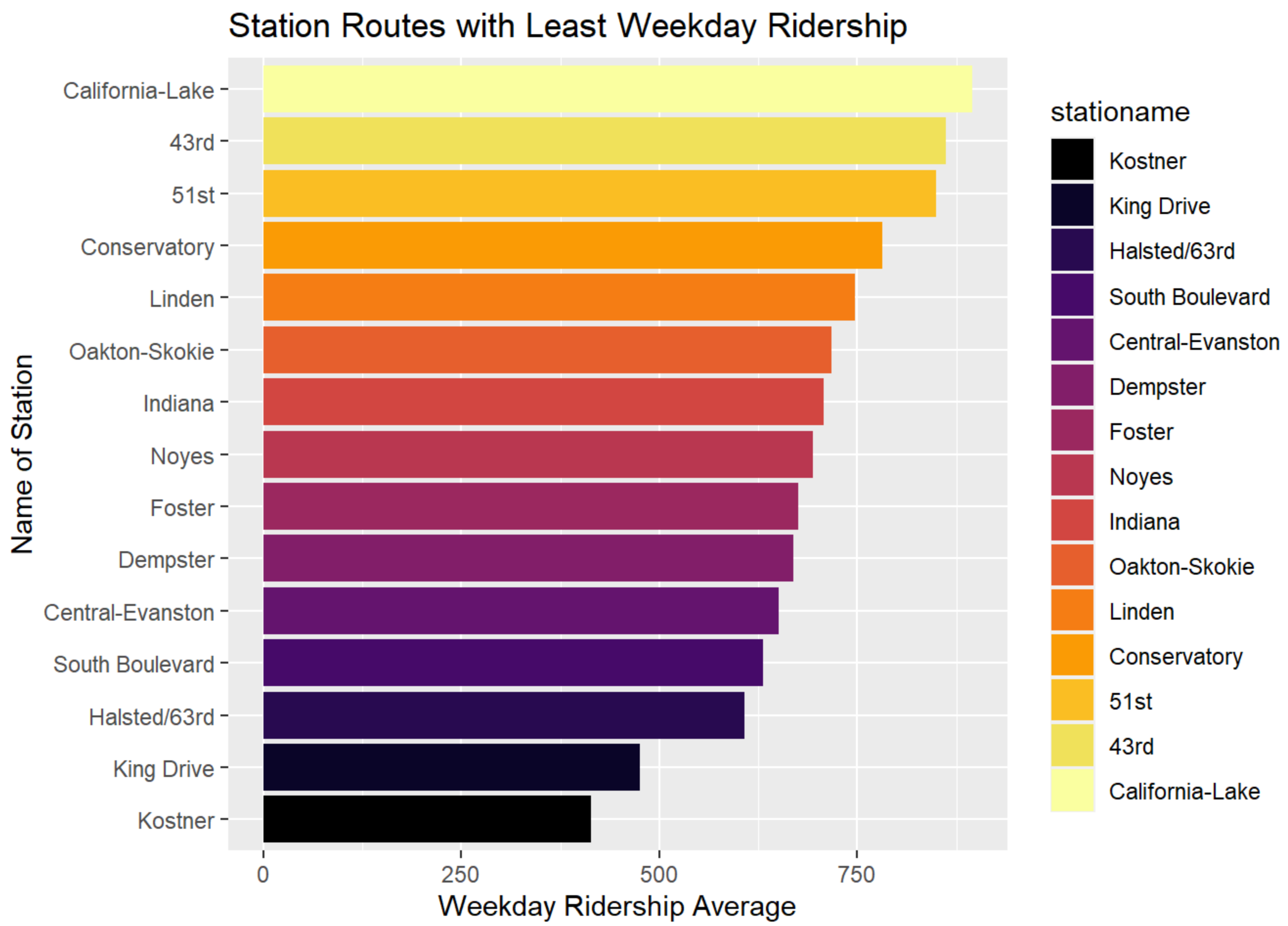
```
sum <- station_mean[order(station_mean$avg_sat, decreasing = TRUE),]
# Find the top 15 stations on Saturday
sum <- head(sum, n = 15)
sum$stationame <- factor(sum$stationame, levels = sum$stationame[order(sum$avg_sat)])
colors <- viridis::viridis(nrow(sum))

# Plot the graph
ggplot(sum, aes(x = avg_sat, y = stationame, fill=stationame)) +
geom_bar(stat = "identity") +
labs(x = "Saturday Ridership Average", y = "Name of Station") +
ggtitle("Station Routes with Highest Saturday Ridership") +
scale_fill_manual(values = colors)
```



```
sum <- station_mean[order(station_mean$avg_sun, decreasing = TRUE),]
# Find the top 15 stations on saturday
sum <- head(sum, n = 15)
sum$stationame <- factor(sum$stationame, levels = sum$stationame[order(sum$avg_sun)])
colors <- viridis::plasma(nrow(sum))

# Plot the graph
ggplot(sum, aes(x = avg_sun, y = stationame, fill=stationame)) +
geom_bar(stat = "identity") +
labs(x = "Sunday/Holiday Ridership Average", y = "Name of Station") +
ggtitle("Station Routes with Highest Sunday/Holiday Ridership") +
scale_fill_manual(values = colors)
```



```
sum <- station_mean[order(station_mean$avg_weekday, decreasing = TRUE),]
# Find the bottom 15 stations on week day
sum <- tail(sum, n = 15)
sum$stationame <- factor(sum$stationame, levels = sum$stationame[order(sum$avg_weekday)])
colors <- viridis::inferno(nrow(sum))

# Plot the graph
ggplot(sum, aes(x = avg_weekday, y = stationame, fill=stationame)) +
geom_bar(stat = "identity") +
labs(x = "Weekday Ridership Average", y = "Name of Station") +
ggtitle("Station Routes with Least Weekday Ridership") +
scale_fill_manual(values = colors)
```

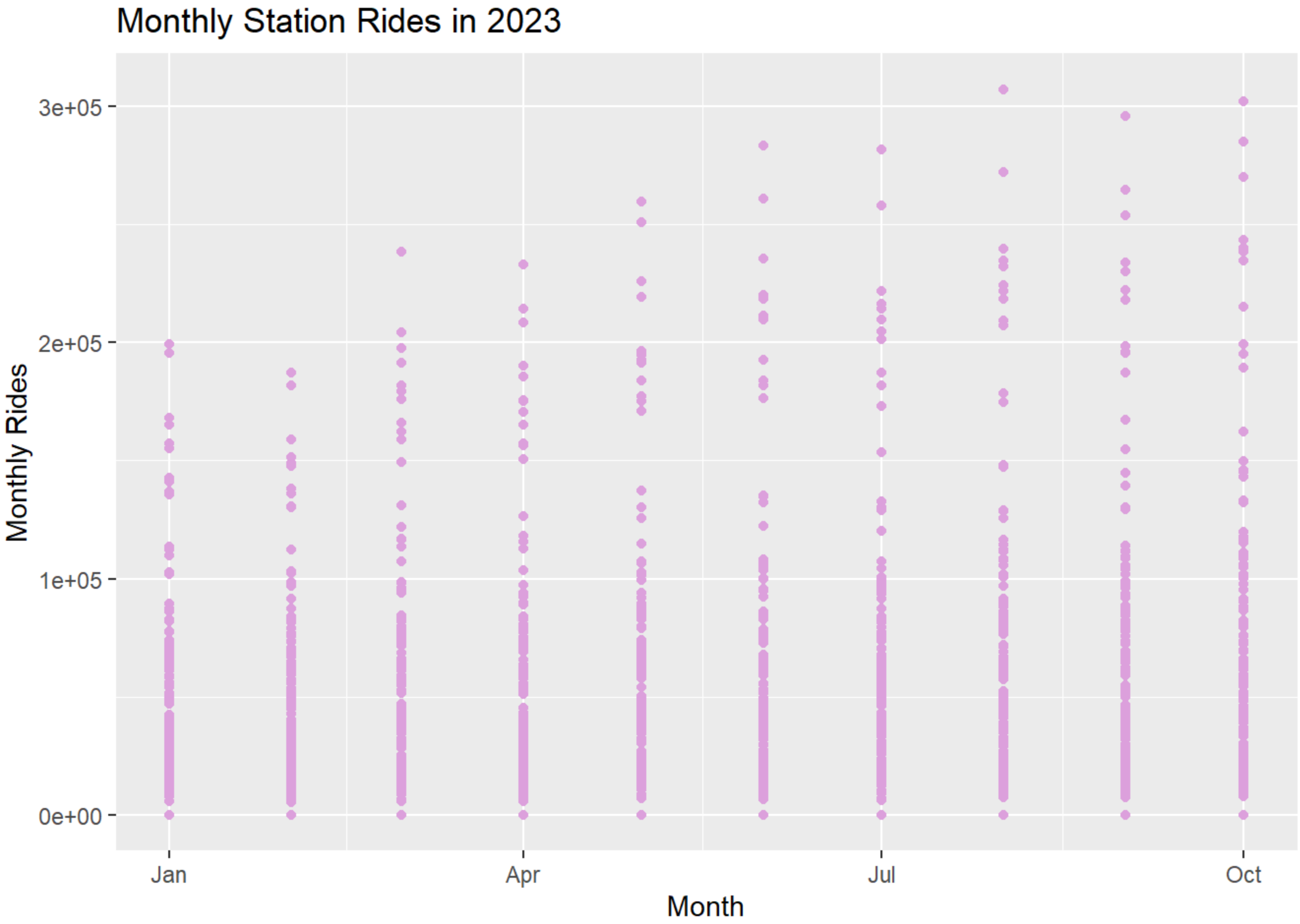


More EDA FOR Lstation & Bus Route

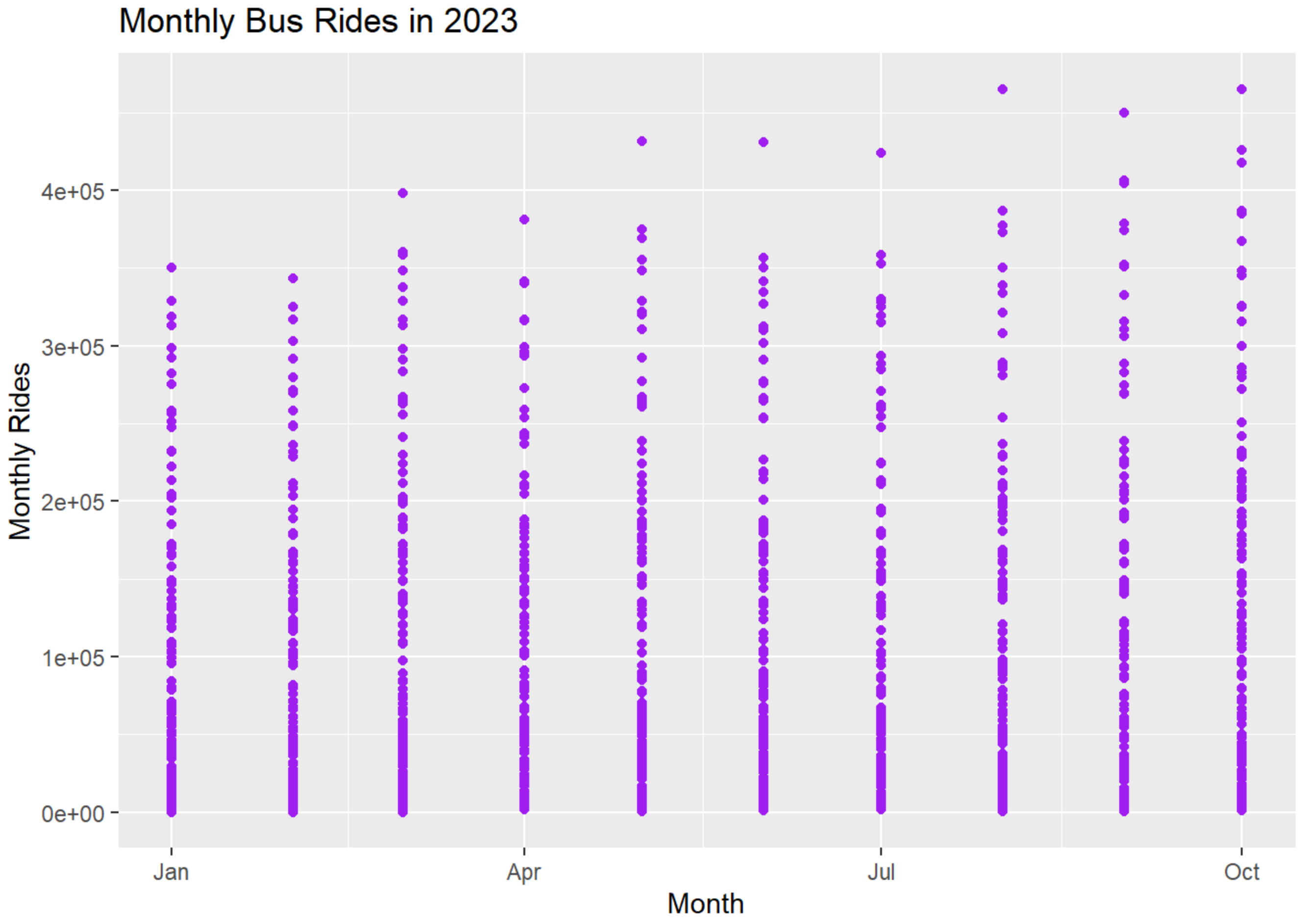
```
library(lubridate)
```

```
##
## Attaching package: 'lubridate'
## The following objects are masked from 'package:base':
##
##     date, intersect, setdiff, union
```

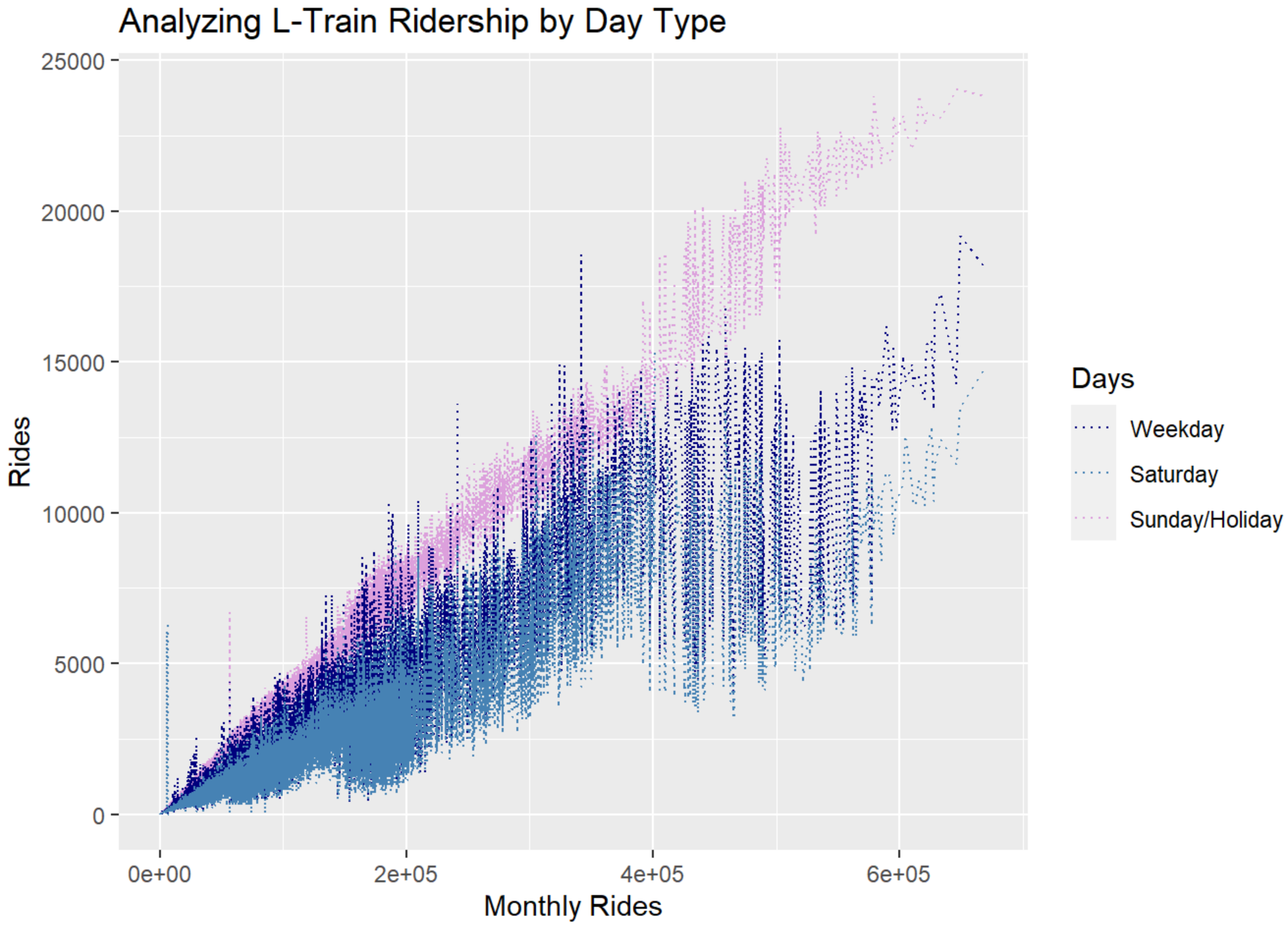
```
# Filter data for the year 2023
lstation_2023 <- lstation %>%
  filter(year(month_beginning) == 2023)
# Create a scatter plot
ggplot(data = lstation_2023, aes(x = month_beginning, y = monthtotal)) +

  geom_point(color = "plum") +
  labs(x = "Month", y = "Monthly Rides", title = "Monthly Station Rides in 2023")
```



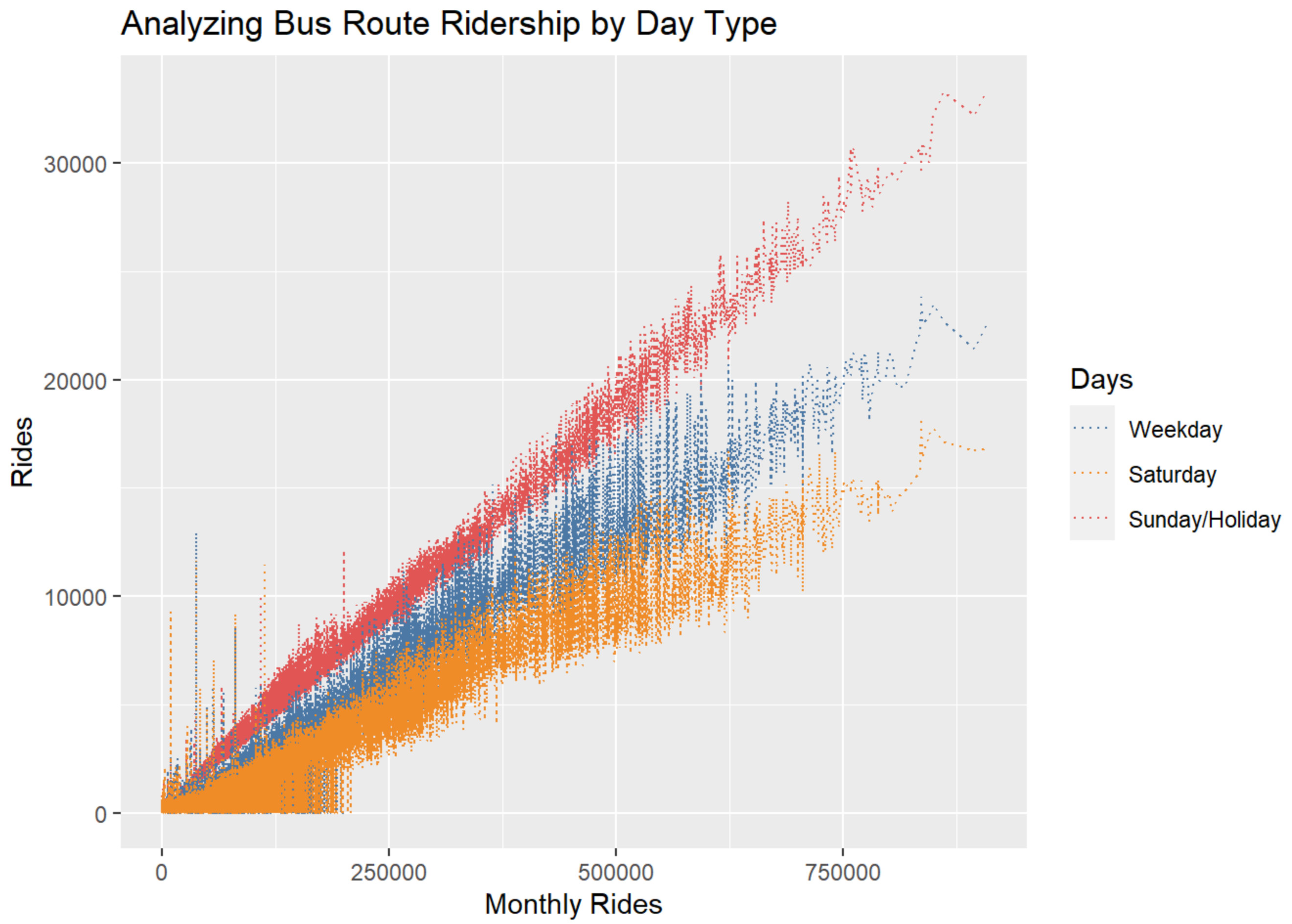
```
# Filter data for the year 2023
Bus_2023 <- busRoute_data %>%
  filter(year(Month_Beginning) == 2023)
# Create a scatter plot
ggplot(data = Bus_2023, aes(x = Month_Beginning, y = MonthTotal)) +

  geom_point(color = "purple") +
  labs(x = "Month", y = "Monthly Rides", title = "Monthly Bus Rides in 2023")
```



```
ggplot(lstation, aes(x = monthtotal)) +
  geom_line(aes(y = avg_weekday_rides, color = "Weekday"), linetype = "dotted") +
  geom_line(aes(y = avg_saturday_rides, color = "Saturday"), linetype = "dotted") +
  geom_line(aes(y = avg_sunday.holiday_rides, color = "Sunday/Holiday"), linetype = "dotted") +
  ggtitle("Analyzing L-Train Ridership by Day Type") +
  xlab("Monthly Rides") +
  ylab("Rides") +
  scale_color_manual(
    name = "Days",  # Set legend name
    values = c("navy", "steelblue", "plum"),  # Set colors for lines
    labels = c("Weekday", "Saturday", "Sunday/Holiday")  # Set labels for legend
  )
```



```
ggplot(busRoute_data, aes(x = MonthTotal)) +
  geom_line(aes(y = Avg_Weekday_Rides, color = "Weekday"), linetype = "dotted") +
  geom_line(aes(y = Avg_Saturday_Rides, color = "Saturday"), linetype = "dotted") +
  geom_line(aes(y = Avg_Sunday.Holiday_Rides, color = "Sunday/Holiday"), linetype = "dotted") +
  ggtitle("Analyzing Bus Route Ridership by Day Type") +
  xlab("Monthly Rides") +
  ylab("Rides") +
  scale_color_manual(
    name = "Days",  # Set legend name
    values = c("#4E79A7", "#F28E2B", "#E15759"),  # Set colors for lines
    labels = c("Weekday", "Saturday", "Sunday/Holiday")  # Set labels for legend
  )
```



###

Maximum and Minimum traffic on Bus Route

```
# Aggregate data to calculate total traffic for each day type
traffic_bus <- busRoute_data %>%
  summarise(
    total_weekday_traffic = sum(Avg_Weekday_Rides),
    total_saturday_traffic = sum(Avg_Saturday_Rides),
    total_sunday_holiday_traffic = sum(Avg_Sunday.Holiday_Rides)
  )

# Print the total traffic for each day type
print(traffic_bus)
```

```
##   total_weekday_traffic total_saturday_traffic total_sunday_holiday_traffic
## 1              89321754               55693662                     40857403
```

Maximum and Minimum traffic on L-Station

```
# Aggregate data to calculate total traffic for each day type
traffic_station <- lstation %>%
  summarise(
    total_weekday_traffic = sum(avg_weekday_rides),
    total_saturday_traffic = sum(avg_saturday_rides),
    total_sunday_holiday_traffic = sum(avg_sunday.holiday_rides)
  )

# Print the total traffic for each day type
print(traffic_station)
```

```
##   total_weekday_traffic total_saturday_traffic total_sunday_holiday_traffic
## 1              62151341               38473800                     29046345
```

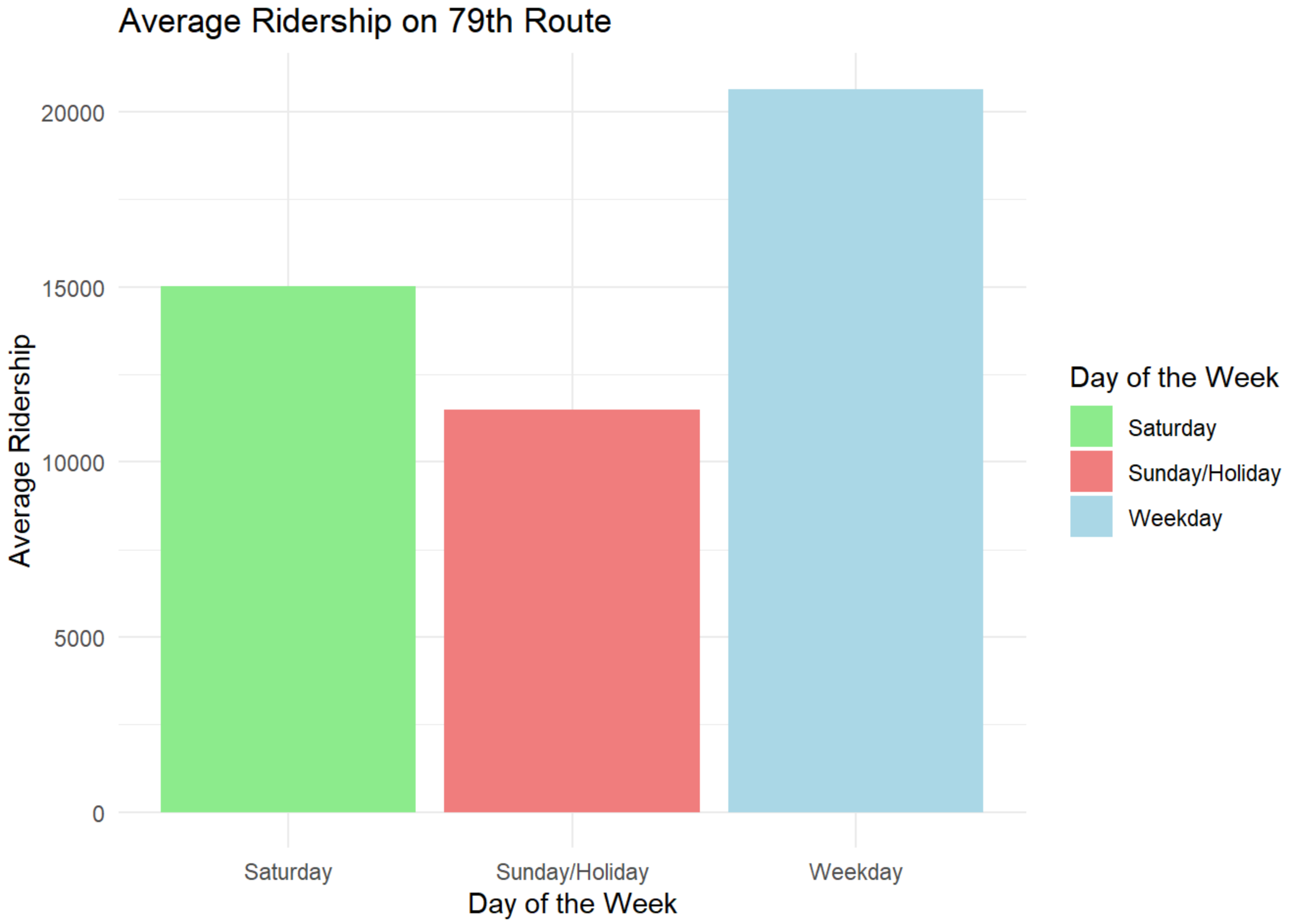
```
print(names(busRoute_data))
```

```
## [1] "route"                    "routename"
## [3] "Month_Beginning"          "Avg_Weekday_Rides"
## [5] "Avg_Saturday_Rides"       "Avg_Sunday.Holiday_Rides"
## [7] "MonthTotal"
```

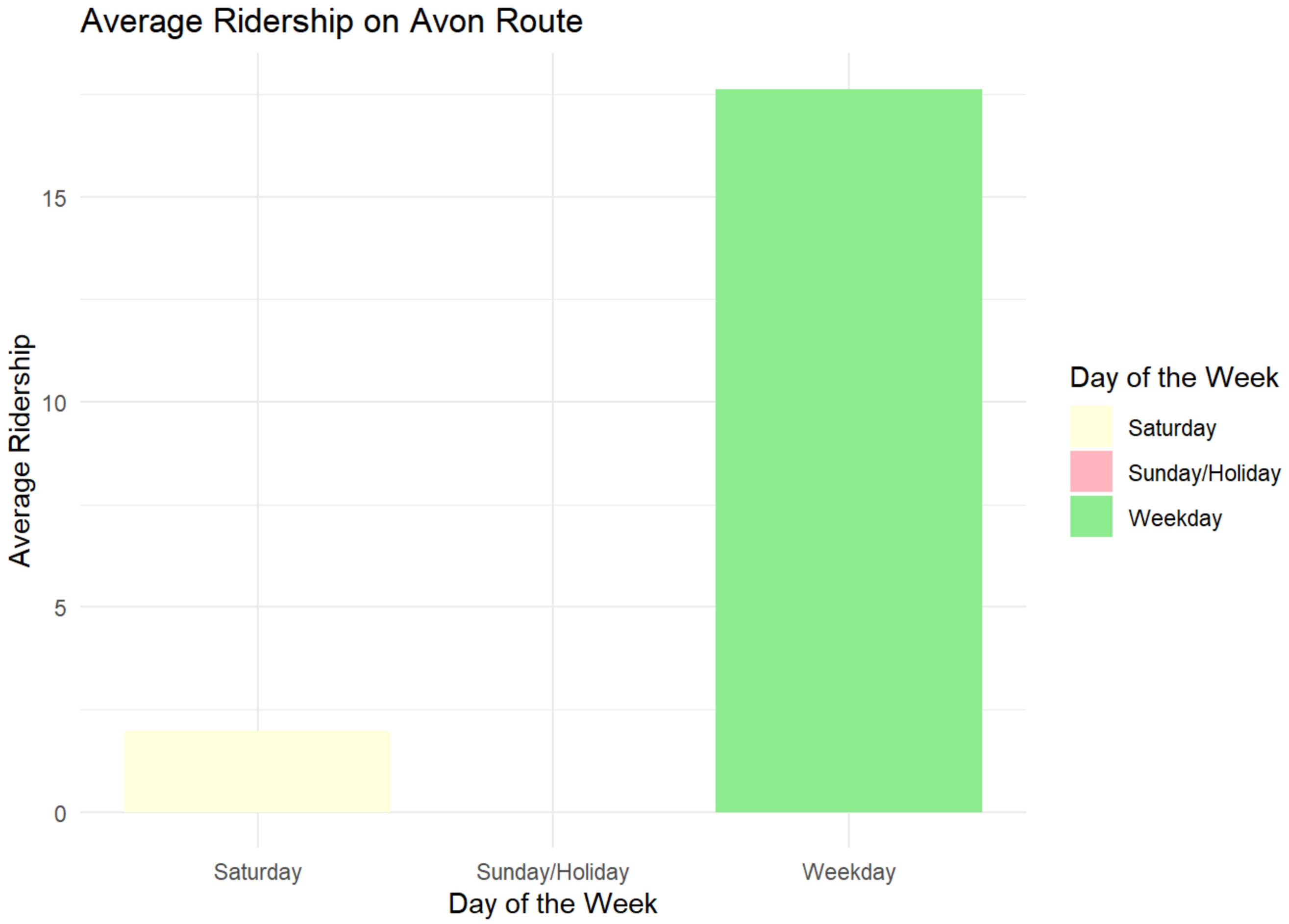
```
print(names(lstation))
```

```
## [1] "station_id"               "stationname"
## [3] "month_beginning"          "avg_weekday_rides"
## [5] "avg_saturday_rides"       "avg_sunday.holiday_rides"
## [7] "monthtotal"
```

```
# Subset data for the 79th route
route_79 <- summary_df[summary_df$routename == "79th", ]

# Create a data frame for plotting
plot_data <- data.frame(
  Day = c("Sunday/Holiday", "Saturday", "Weekday"),
  Ridership = c(route_79$Sunday_Holiday, route_79$Saturday, route_79$Weekday)
)

# Create the bar plot
ggplot(plot_data, aes(x = Day, y = Ridership, fill = Day)) +
  geom_bar(stat = "identity") +
  labs(x = "Day of the Week", y = "Average Ridership", fill = "Day of the Week") +
  ggtitle("Average Ridership on 79th Route") +
  scale_fill_manual(values = c("Sunday/Holiday" = "lightcoral", "Saturday" = "lightgreen", "Week
day" = "lightblue")) +
  theme_minimal()
```



```
# Subset data for the Avon
route_ashland <- summary_df[summary_df$routename == "Avon Express", ]

# Create a data frame for plotting
plot_data_ashland <- data.frame(
  Day = c("Sunday/Holiday", "Saturday", "Weekday"),
  Ridership = c(route_ashland$Sunday_Holiday, route_ashland$Saturday, route_ashland$Weekday)
)

# Create the bar plot for Avon route
ggplot(plot_data_ashland, aes(x = Day, y = Ridership, fill = Day)) +
  geom_bar(stat = "identity") +
  labs(x = "Day of the Week", y = "Average Ridership", fill = "Day of the Week") +
  ggtitle("Average Ridership on Avon Route") +
  scale_fill_manual(values = c("Sunday/Holiday" = "lightpink", "Saturday" = "lightyellow", "Weekday" = "lightgreen")) +
  theme_minimal()
```


Average Ridership on Avon Route
Average Ridership
0
5
10
15
Saturday
Sunday/Holiday
Weekday
Day of the Week
Day of the Week
Saturday
Sunday/Holiday
Weekday