Cyclistic: Google Data Analytics Capstone Project

Bernard Bamidele Aghedo

Background I'll be performing the role of a junior data analyst working in the marketing analyst team at Cyclistic, a bike share company in Chicago. The director of marketing believes the company's future success depends on maximizing the number of annual memberships. Therefore, your team wants to understand how casual riders and annual members use Cyclistic bikes differently. From these insights, your team will design a new marketing strategy to convert casual riders into annual members. But first, Cyclistic executives must approve your recommendations, so they must be backed up with compelling data insights and professional data visualizations.

Cyclistic A bike-share program that features more than 5,800 bicycles and 600 docking stations. Cyclistic sets itself apart by also offering reclining bikes, hand tricycles, and cargo bikes, making bike-share more inclusive to people with disabilities and riders who can't use a standard two-wheeled bike. The majority of riders opt for traditional bikes; about 8% of riders use the assistive options. Cyclistic users are more likely to ride for leisure, but about 30% use them to commute to work each day.

Loading the available dataset into R

```
# Loading packages

library("tidyverse")
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.2 --
## v ggplot2 3.3.6     v purrr   0.3.4
## v tibble  3.1.8     v dplyr   1.0.9
## v tidyr   1.2.0     v stringr 1.4.0
## v readr   2.1.2     v forcats 0.5.1
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library("janitor")
```

```
##
## Attaching package: 'janitor'
##
## The following objects are masked from 'package:stats':
##
##     chisq.test, fisher.test
```

```
library("lubridate")
```

```
##
## Attaching package: 'lubridate'
```

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

```
library("ggplot2")
```

```
# Set working directory
setwd("~/Desktop/Cyclistic Case Study")
```

```
# Reading .csv divvy-tripdata files

df1 <- read.csv("202004-divvy-tripdata.csv")
df2 <- read.csv("202005-divvy-tripdata.csv")
df3 <- read.csv("202006-divvy-tripdata.csv")
df4 <- read.csv("202007-divvy-tripdata.csv")
df5 <- read.csv("202008-divvy-tripdata.csv")
df6 <- read.csv("202009-divvy-tripdata.csv")
df7 <- read.csv("202010-divvy-tripdata.csv")
df8 <- read.csv("202011-divvy-tripdata.csv")
df9 <- read.csv("202012-divvy-tripdata.csv")
df10 <- read.csv("202101-divvy-tripdata.csv")
df11 <- read.csv("202103-divvy-tripdata.csv")
df12 <- read.csv("202104-divvy-tripdata.csv")
df13 <- read.csv("202105-divvy-tripdata.csv")
df14 <- read.csv("202106-divvy-tripdata.csv")
df15 <- read.csv("202107-divvy-tripdata.csv")
df16 <- read.csv("202108-divvy-tripdata.csv")
df17 <- read.csv("202109-divvy-tripdata.csv")
df18 <- read.csv("202110-divvy-tripdata.csv")
df19 <- read.csv("202111-divvy-tripdata.csv")
df20 <- read.csv("202112-divvy-tripdata.csv")
df21 <- read.csv("202203-divvy-tripdata.csv")
df22 <- read.csv("202201-divvy-tripdata.csv")
df23 <- read.csv("202202-divvy-tripdata.csv")
df24 <- read.csv("202204-divvy-tripdata.csv")
df25 <- read.csv("202205-divvy-tripdata.csv")
df26 <- read.csv("202206-divvy-tripdata.csv")
```

```
# binding all csv files

bike_rides <- rbind(df1,df2,df3,df4,df5,df6,df7,
                    df8,df9,df10,df11,df12,df13,
                    df14,df15,df16,df17,df18,df19,
                    df20,df21,df22,df23,df24,df25,df26)
```

```
# Cleaning the data: removing empty columns and changing data types
bike_rides <- remove_empty(bike_rides, which = c("cols","rows"))

unique(bike_rides$member_casual)
```

```
## [1] "member" "casual"
```

```r
unique(bike_rides$rideable_type)
```

```
## [1] "docked_bike"   "electric_bike" "classic_bike"
```

```r
bike_rides$start_station_id <- as.integer(bike_rides$start_station_id)
```

```
## Warning: NAs introduced by coercion
```

```r
bike_rides$end_station_id <- as.integer(bike_rides$end_station_id)
```

```
## Warning: NAs introduced by coercion
```

```r
# Parsing date/time

bike_rides$started_at <- ymd_hms(bike_rides$started_at)

bike_rides$ended_at <- ymd_hms(bike_rides$ended_at)

bike_rides$start_hour <- hour(bike_rides$started_at)

bike_rides$end_hour <- hour(bike_rides$ended_at)

bike_rides$trip_date <- as.Date(bike_rides$started_at)
bike_rides$trip_month <- format(as.Date(bike_rides$trip_date), "%B")
bike_rides$trip_day <- format(as.Date(bike_rides$trip_date), "%d")
bike_rides$trip_year <- format(as.Date(bike_rides$trip_date), "%Y")
bike_rides$trip_weekday <- weekdays(bike_rides$trip_date)
```

```r
# Checking for test stations

unique(bike_rides$start_station_name[grep("test", bike_rides$start_station_name)])
```

```
## [1] "hubbard_test_lws"
```

```r
# Filtering test stations

bike_rides <- (filter(bike_rides, !(start_station_name == "hubbard_test_lws" |
                                      start_station_name == "")))
```

```r
# To analyze TOP stations

all_trip_stations <- bike_rides[,c(5,9,10)]

all_trip_stations <- all_trip_stations[!duplicated(all_trip_stations$start_station_name),]
```

```r
# Total number of trip stations

unique(all_trip_stations)
```

```r
NROW(unique(all_trip_stations))
```

```
## [1] 1321
```

```r
# Calculating Ride Lengths in Minutes

bike_rides$ride_length <- difftime(bike_rides$ended_at, bike_rides$started_at)
bike_rides$ride_length <- bike_rides$ride_length/60
bike_rides$ride_length <- round(bike_rides$ride_length, 2)
```

```r
# Remove observations where ride length is below 0

bike_rides <- filter(bike_rides, ride_length > 0)
```

```r
# Compare members and casual users

bike_rides %>%
  group_by(member_casual) %>%
  summarise(avg_ride_length = mean(ride_length), median_ride_length = median(ride_length),
            max_ride_length = max(ride_length), min_ride_length = min(ride_length))
```

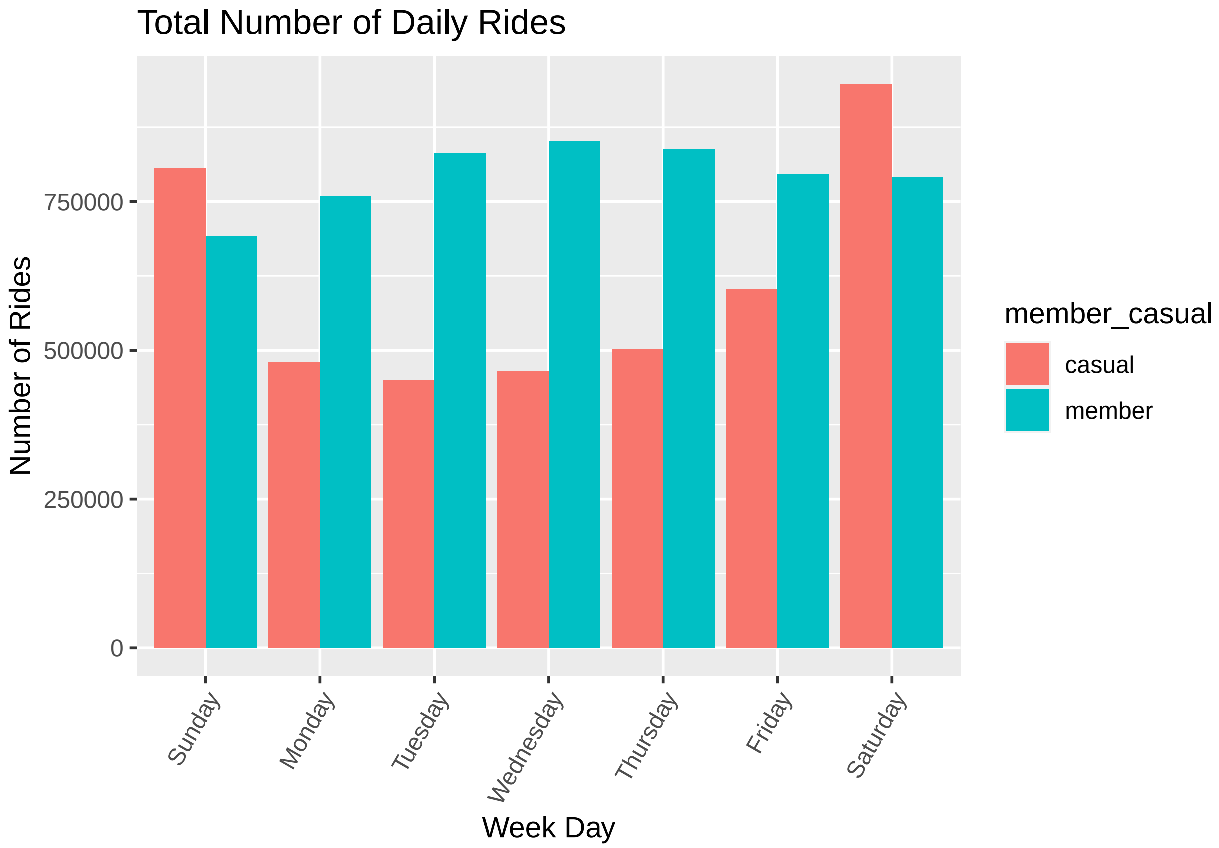
```
## # A tibble: 2 x 5
##   member_casual avg_ride_length median_ride_length max_ride_length min_ride_le~1
##   <chr>         <drtn>          <drtn>             <drtn>          <drtn>
## 1 casual        37.59273 secs   17.58 secs         55944.15 secs   0.02 secs
## 2 member        14.40054 secs   10.15 secs         58720.03 secs   0.02 secs
## # ... with abbreviated variable name 1: min_ride_length
```

```r
# Order the days of the week. Will also order the month

bike_rides$trip_weekday <- ordered(bike_rides$trip_weekday,
        levels=c("Sunday", "Monday", "Tuesday", "Wednesday", "Thursday",
                 "Friday", "Saturday"))

bike_rides$trip_month <- ordered(bike_rides$trip_month,
        levels=c("January", "February", "March", "April", "May",
                 "June", "July", "August", "September", "October",
                 "November", "December"))
```

```r
# Avg ride time by each day for members & casual users

casual_member_avg_ride <- aggregate(bike_rides$ride_length ~
                          bike_rides$member_casual + bike_rides$trip_weekday,
                                                FUN = mean)
```

```r
# Relationship between Rider type and No. of rides in each day of the week

bike_rides %>%
  group_by(member_casual, trip_weekday) %>%
  summarise(number_of_rides = n(), average_duration = mean(ride_length)) %>%
  arrange(member_casual, trip_weekday) %>%
ggplot(aes(x = trip_weekday, y = number_of_rides, fill = member_casual)) +
  geom_bar(position = "dodge", stat = 'identity') +
  labs(title="Total Number of Daily Rides",
            x = "Week Day", y = "Number of Rides") +
  theme(axis.text.x = element_text(angle = 60, hjust = 1))
```

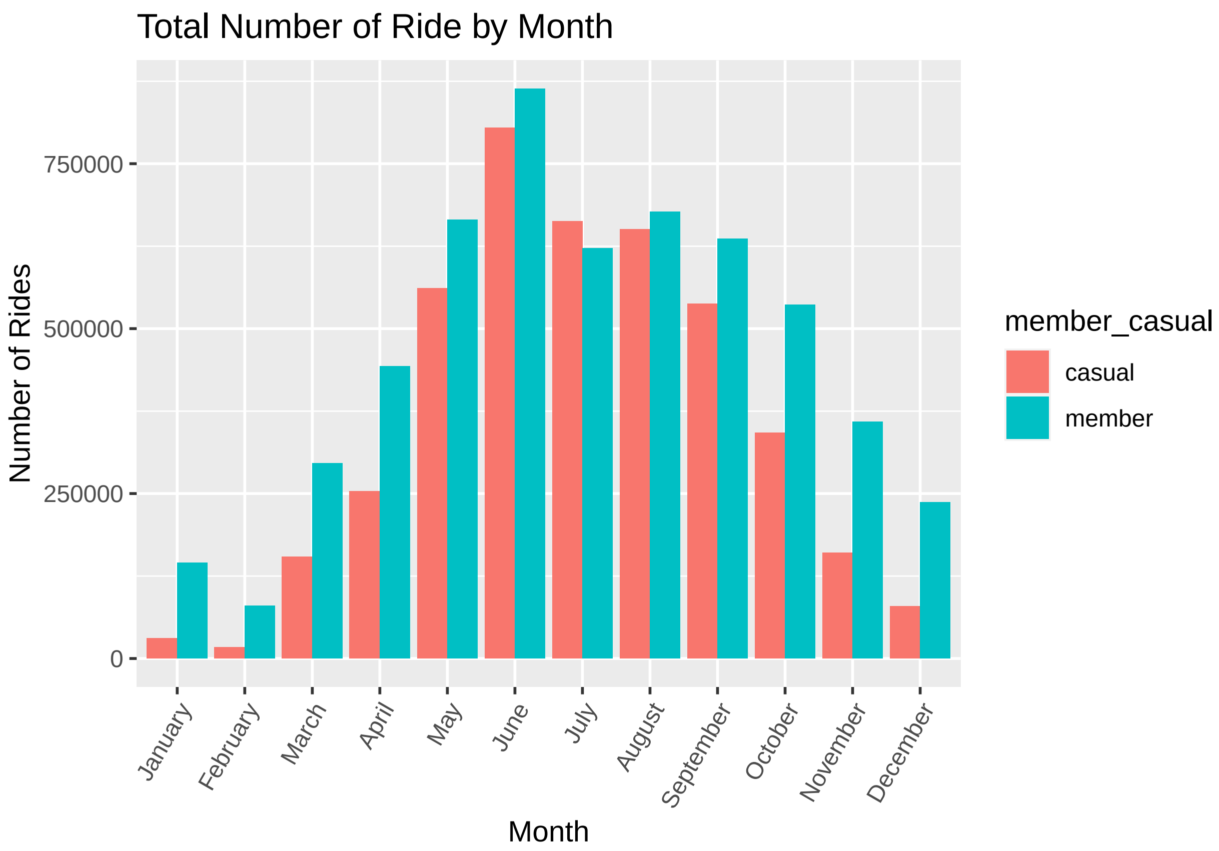
```
## ‘summarise()‘ has grouped output by ’member_casual’. You can override using the
## ‘.groups‘ argument.
```



```
# Relationship between Rider type and No of rides monthly

bike_rides %>%
  group_by(member_casual, trip_month) %>%
  summarise(number_of_rides = n(), average_duration = mean(ride_length)) %>%
  arrange(member_casual, trip_month)    %>%
  ggplot(aes(x = trip_month, y = number_of_rides, fill = member_casual)) +
  geom_col(position = "dodge", stat = 'identity') +
  labs(title="Total Number of Ride by Month", x = "Month", y = "Number of Rides") +
  theme(axis.text.x = element_text(angle = 60, hjust = 1))
```

```
## ‘summarise()‘ has grouped output by ’member_casual’. You can override using the
## ‘.groups‘ argument.
```

```
## Warning: Ignoring unknown parameters: stat
```



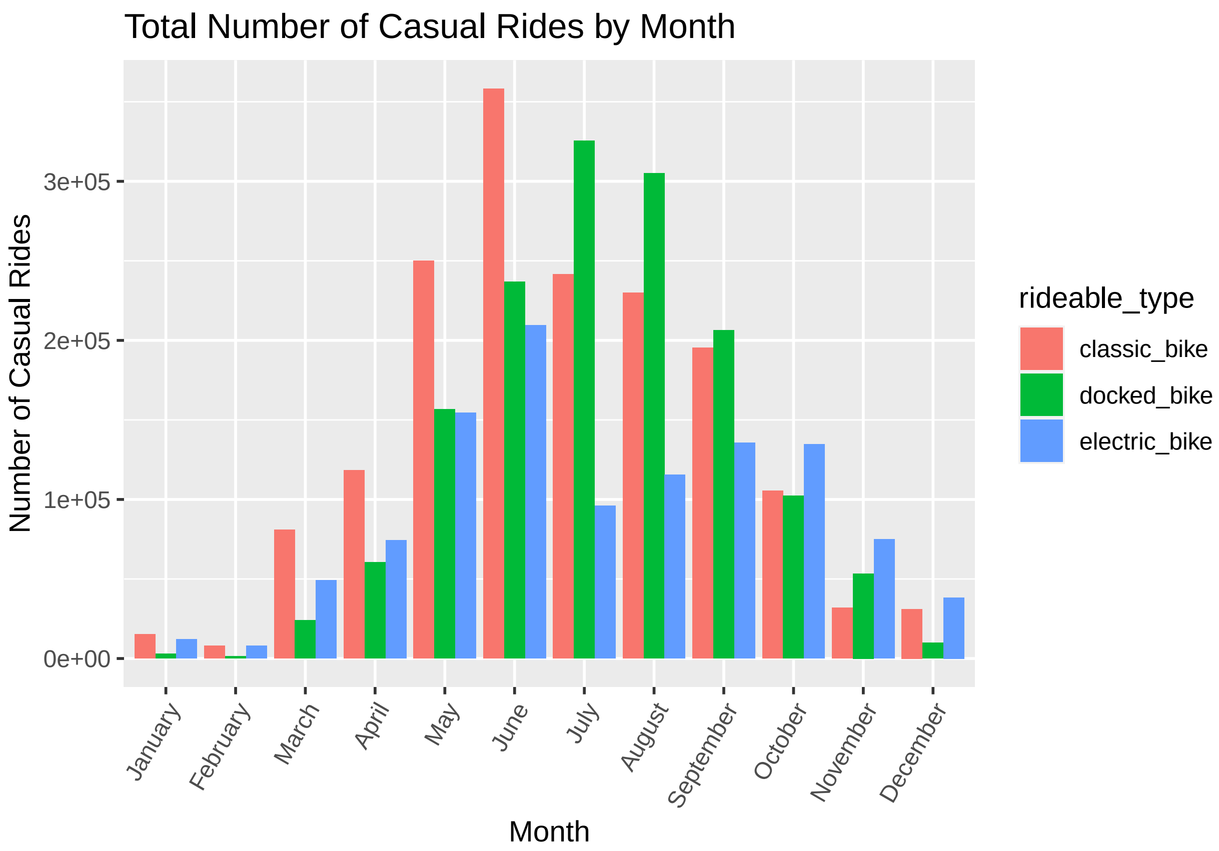
```
# Analyze Casual Bike type and rides monthly & daily

casual_bike_riders <-  filter(bike_rides, member_casual == "casual")

unique(casual_bike_riders$rideable_type)
```

```
## [1] "docked_bike"   "electric_bike" "classic_bike"
```

```
## Casual rides and bike type relationship Monthly

casual_bike_riders %>%
  group_by(rideable_type, trip_month) %>%
  summarise(number_of_rides = n(), average_duration = mean(ride_length)) %>%
  arrange(rideable_type, trip_month)    %>%
  ggplot(aes(x = trip_month, y = number_of_rides, fill = rideable_type)) +
  geom_col(position = "dodge", stat = 'identity') +
  labs(title="Total Number of Casual Rides by Month", x = "Month", y = "Number of Casual Rides") +
  theme(axis.text.x = element_text(angle = 60, hjust = 1))
```

```
## `summarise()` has grouped output by 'rideable_type'. You can override using the
## `.groups` argument.
```

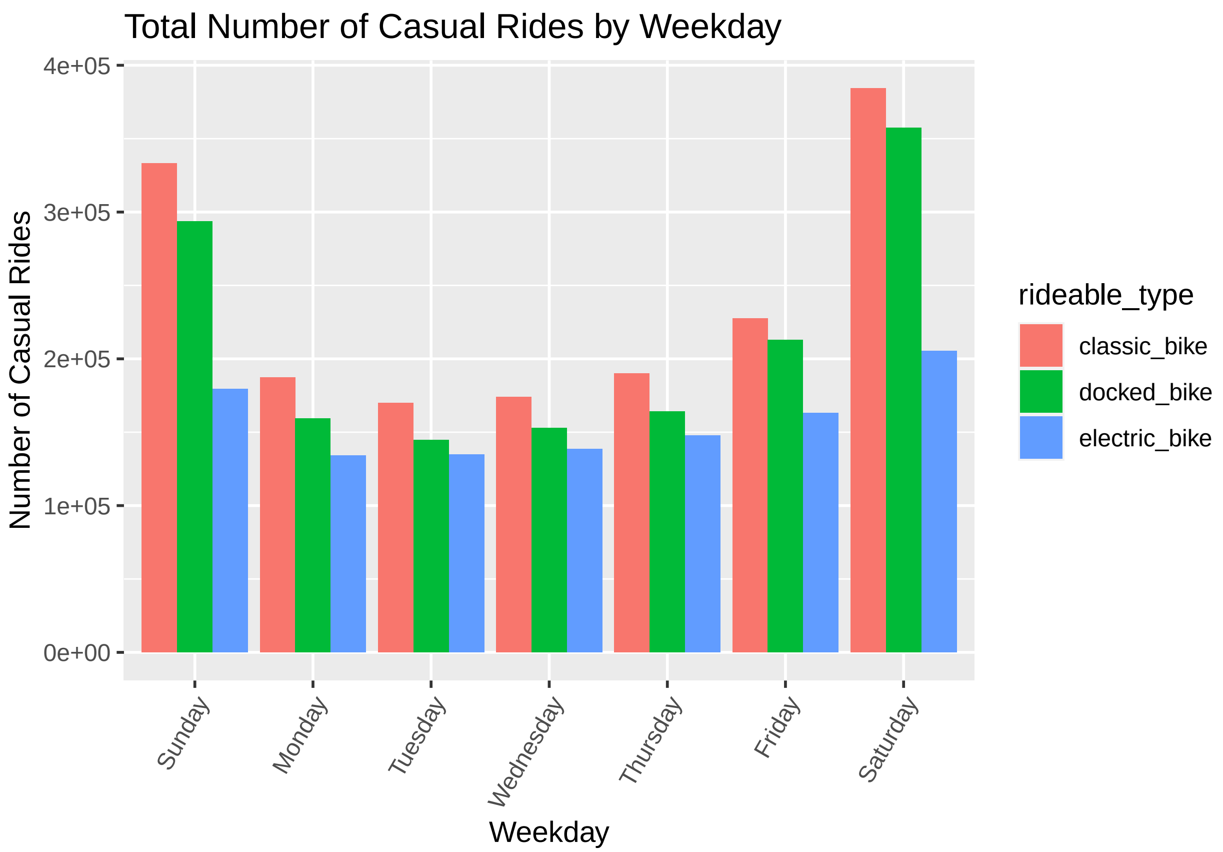
```
## Warning: Ignoring unknown parameters: stat
```



```
## Casual rides and bike type relationship Daily

casual_bike_riders %>%
  group_by(rideable_type, trip_weekday) %>%
  summarise(number_of_rides = n(), average_duration = mean(ride_length)) %>%
  arrange(rideable_type, trip_weekday)  %>%
  ggplot(aes(x = trip_weekday, y = number_of_rides, fill = rideable_type)) +
  geom_col(position = "dodge", stat = 'identity') +
  labs(title="Total Number of Casual Rides by Weekday", x = "Weekday", y = "Number of Casual Rides") +
  theme(axis.text.x = element_text(angle = 60, hjust = 1))
```

```
## `summarise()` has grouped output by 'rideable_type'. You can override using the
## `.groups` argument.
```

```
## Warning: Ignoring unknown parameters: stat
```

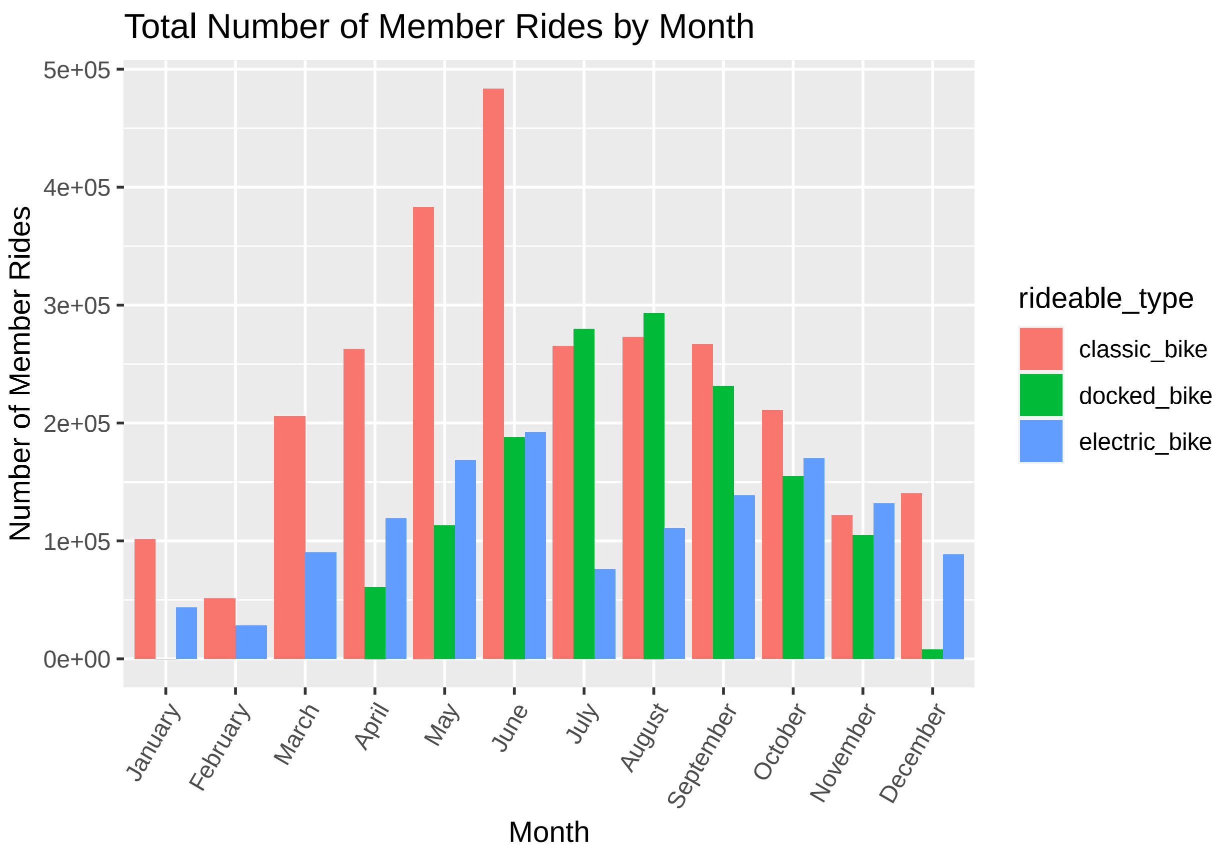


```
# Analyze Member Bike type and rides monthly & daily

member_bike_riders <- filter(bike_rides, member_casual == "member")
```

```
## Member rides and bike type relationship Monthly

member_bike_riders %>%
  group_by(rideable_type, trip_month) %>%
  summarise(number_of_rides = n(), average_duration = mean(ride_length)) %>%
  arrange(rideable_type, trip_month)    %>%
  ggplot(aes(x = trip_month, y = number_of_rides, fill = rideable_type)) +
  geom_col(position = "dodge", stat = 'identity') +
  labs(title="Total Number of Member Rides by Month", x = "Month", y = "Number of Member Rides") +
  theme(axis.text.x = element_text(angle = 60, hjust = 1))
```

```
## ‘summarise()‘ has grouped output by ’rideable_type’. You can override using the
## ‘.groups‘ argument.
```

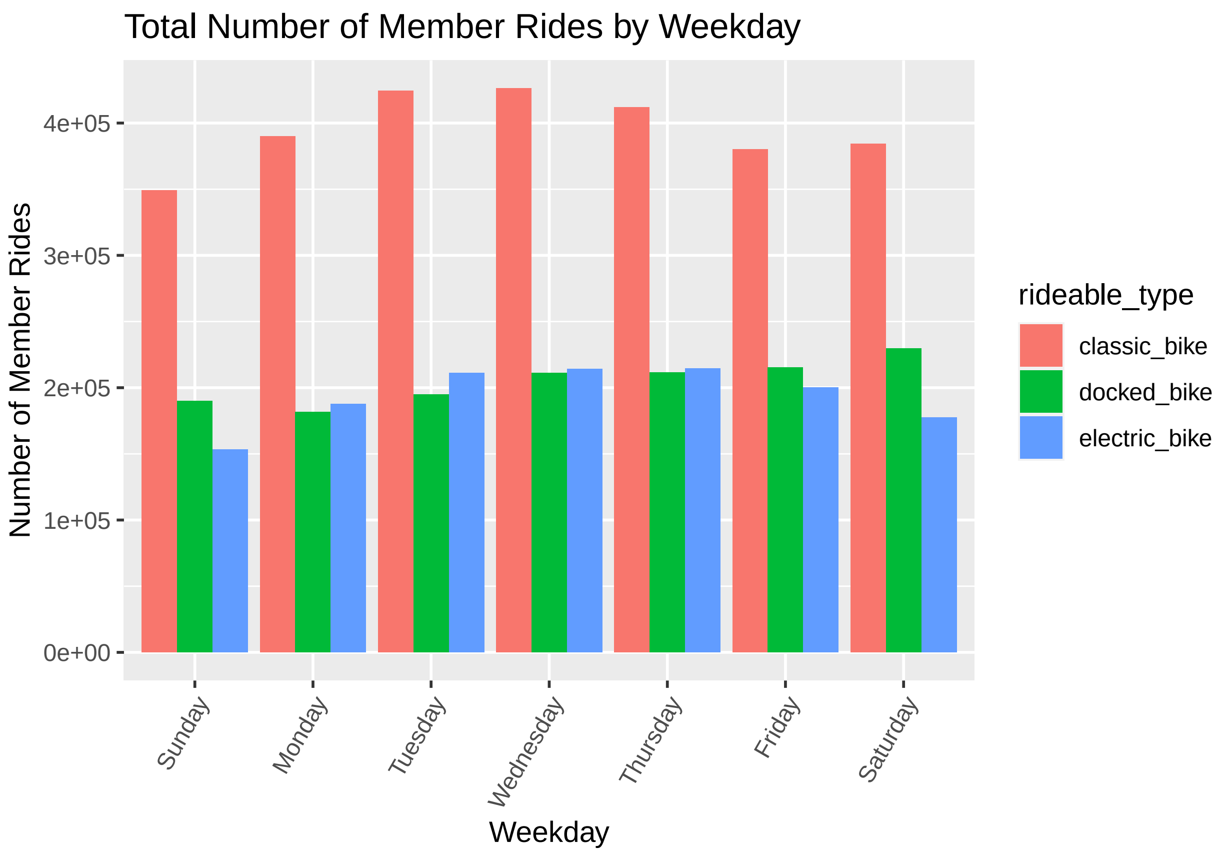
```
## Warning: Ignoring unknown parameters: stat
```



```
## Member rides and bike types relationship Daily

member_bike_riders %>%
  group_by(rideable_type, trip_weekday) %>%
  summarise(number_of_rides = n(), average_duration = mean(ride_length)) %>%
  arrange(rideable_type, trip_weekday)  %>%
  ggplot(aes(x = trip_weekday, y = number_of_rides, fill = rideable_type)) +
  geom_col(position = "dodge", stat = 'identity') +
  labs(title="Total Number of Member Rides by Weekday", x = "Weekday", y = "Number of Member Rides") +
  theme(axis.text.x = element_text(angle = 60, hjust = 1))
```

```
## `summarise()` has grouped output by 'rideable_type'. You can override using the
## `.groups` argument.
```

```
## Warning: Ignoring unknown parameters: stat
```



```
# Popular Routes taken by casual riders

casual_bike_riders <- casual_bike_riders %>%
  mutate(route = paste(start_station_name, "To", sep=" "))

casual_bike_riders <- casual_bike_riders %>%
  mutate(route = paste(route, end_station_name, sep =" "))

casual_pop_routes <-  casual_bike_riders %>%
  group_by(route) %>%
  summarise(number_of_rides  = n(), average_duration_minutes = mean(ride_length)) %>%
  arrange(route, number_of_rides, average_duration_minutes)
```

```
## Calculating Top 10 stations for casual riders

top10_casual_pop_routes <- head(arrange(casual_pop_routes, desc(number_of_rides)), 10)

top10_casual_pop_routes <- top10_casual_pop_routes %>%
  separate(route, c("start_station_name", "end_station_name"), sep = " To ")

unique(top10_casual_pop_routes)
```

```
## # A tibble: 10 x 4
##    start_station_name                end_station_name            numbe~1 avera~2
##    <chr>                             <chr>                         <int> <drtn>
##  1 Streeter Dr & Grand Ave           Streeter Dr & Grand Ave       21572 50.234~
##  2 Millennium Park                   Millennium Park               12204 51.790~
##  3 Michigan Ave & Oak St             Michigan Ave & Oak St         11111 53.578~
##  4 Lake Shore Dr & Monroe St         Lake Shore Dr & Monroe St      9895 50.188~
##  5 Buckingham Fountain               Buckingham Fountain            8945 67.655~
##  6 Indiana Ave & Roosevelt Rd        Indiana Ave & Roosevelt Rd     7390 56.264~
##  7 Theater on the Lake               Theater on the Lake            6854 50.732~
##  8 Michigan Ave & 8th St             Michigan Ave & 8th St          6399 58.087~
##  9 Fort Dearborn Dr & 31st St        Fort Dearborn Dr & 31st St     6392 68.953~
## 10 DuSable Lake Shore Dr & Monroe St DuSable Lake Shore Dr & Mo~    5908 39.420~
## # ... with abbreviated variable names 1: number_of_rides,
## #   2: average_duration_minutes
```

```
# Merging the Casual popular routes with all_trips_stations for more geo location details

top10_casual_pop_routes_start <- top10_casual_pop_routes[c(1,3,4)]

top10_stations_casual <- merge(top10_casual_pop_routes_start, all_trip_stations)

head(top10_stations_casual, 10)
```

```
##                   start_station_name number_of_rides average_duration_minutes
## 1                Buckingham Fountain            8945            67.65532 secs
## 2  DuSable Lake Shore Dr & Monroe St            5908            39.42059 secs
## 3         Fort Dearborn Dr & 31st St            6392            68.95358 secs
## 4         Indiana Ave & Roosevelt Rd            7390            56.26464 secs
## 5          Lake Shore Dr & Monroe St            9895            50.18860 secs
## 6              Michigan Ave & 8th St            6399            58.08737 secs
## 7              Michigan Ave & Oak St           11111            53.57885 secs
## 8                    Millennium Park           12204            51.79009 secs
## 9            Streeter Dr & Grand Ave           21572            50.23436 secs
## 10               Theater on the Lake            6854            50.73278 secs
##    start_lat start_lng
## 1   41.87650 -87.62050
## 2   41.88096 -87.61674
## 3   41.83860 -87.60820
## 4   41.86790 -87.62300
## 5   41.88100 -87.61670
## 6   41.87280 -87.62400
## 7   41.90100 -87.62380
## 8   41.88100 -87.62410
## 9   41.89230 -87.61200
## 10  41.92630 -87.63080
```

```
# Top 10 popular Routes taken by Member riders

member_bike_riders <- member_bike_riders %>%
  mutate(route = paste(start_station_name, "To", sep=" "))

member_bike_riders <- member_bike_riders %>%
  mutate(route = paste(route, end_station_name, sep =" "))

member_pop_routes <-  member_bike_riders %>%
  group_by(route) %>%
  summarise(number_of_rides  = n(), average_duration_minutes = mean(ride_length)) %>%
  arrange(route, number_of_rides, average_duration_minutes)
```

```
# Calculating Top 10 stations for member riders

top10_member_pop_routes <- head(arrange(member_pop_routes, desc(number_of_rides)), 10)

top10_member_pop_routes <- top10_member_pop_routes %>%
  separate(route, c("start_station_name", "end_station_name"), sep = " To ")

unique(top10_member_pop_routes)
```

```
## # A tibble: 10 x 4
##    start_station_name       end_station_name           number_of_rides average~1
##    <chr>                    <chr>                                <int> <drtn>
##  1 Ellis Ave & 60th St      "Ellis Ave & 55th St"                 6752  5.71927~
##  2 Ellis Ave & 60th St      "University Ave & 57th St"            6410  5.60451~
##  3 University Ave & 57th St "Ellis Ave & 60th St"                 6097  5.06073~
##  4 Ellis Ave & 55th St      "Ellis Ave & 60th St"                 6032  5.85549~
##  5 University Ave & 57th St ""                                    3483 15.17137~
##  6 Ellis Ave & 60th St      ""                                    3374  8.62238~
##  7 Calumet Ave & 33rd St    "State St & 33rd St"                  3368  4.07918~
##  8 State St & 33rd St       "Calumet Ave & 33rd St"               3294  4.76965~
##  9 Loomis St & Lexington St "Morgan St & Polk St"                 3072  5.80585~
## 10 MLK Jr Dr & 29th St      "State St & 33rd St"                  2988  7.78151~
## # ... with abbreviated variable name 1: average_duration_minutes
```

```
#  Merging the popular routes with all_trips_stations for more geo location details

top10_member_pop_routes_start <- top10_member_pop_routes[c(1,3,4)]

top10_stations_member <- merge(top10_member_pop_routes_start, all_trip_stations)
head(top10_stations_member, 10)
```

```
##          start_station_name number_of_rides average_duration_minutes start_lat
## 1     Calumet Ave & 33rd St            3368            4.079186 secs   41.8349
## 2       Ellis Ave & 55th St            6032            5.855494 secs   41.7943
## 3       Ellis Ave & 60th St            6752            5.719271 secs   41.7851
## 4       Ellis Ave & 60th St            6410            5.604510 secs   41.7851
## 5       Ellis Ave & 60th St            3374            8.622380 secs   41.7851
## 6  Loomis St & Lexington St            3072            5.805859 secs   41.8722
## 7       MLK Jr Dr & 29th St            2988            7.781513 secs   41.8421
## 8        State St & 33rd St            3294            4.769651 secs   41.8347
## 9  University Ave & 57th St            6097            5.060733 secs   41.7915
## 10 University Ave & 57th St            3483           15.171375 secs   41.7915
##    start_lng
## 1   -87.6179
## 2   -87.6015
## 3   -87.6011
## 4   -87.6011
## 5   -87.6011
## 6   -87.6615
## 7   -87.6170
## 8   -87.6258
## 9   -87.5999
## 10  -87.5999
```

```
# Export "top10_stations_casual" and "top10_stations_member" into Tableau for visualization

write.csv(top10_stations_casual, "top10_stations_casual.csv")
write.csv(top10_stations_member, "top10_stations_member.csv")
```

link to Dashboard Tableau Viz: Top 10 casual and member stations https://public.tableau.com/views/DashboardCyclisticBikeShare-Top10Member_CasualStation/CyclisticBikeShare-Top10Member_CasualStations

Trends and Correlations Classic bike is being used the most by both casual and member riders with most rides at the first half of the year Docked bike is the next used bikes to classic bike

Member riders have more rides during the week which is between Mondays to Fridays While Casual have more rides during weekends which is Saturdays and Sundays