Divvy Case Study

David Santos

19/01/2022

Abstract

Cyclistic is 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.

Case Study Scenario

I am 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.

Data Source

Data was provided by the Google Data Analytics Professional Certificate program, as part of one of it's capstone projects, from the link: https://divvy-tripdata.s3.amazonaws.com/index.html.

Setting up my environment and loading the data

I decided to only use the last 12 months of data available, since it was more reliable, would provide more insights on the actual trends and overall more complete.

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --
```

```
## v ggplot2 3.3.5     v purrr   0.3.4
## v tibble  3.1.6     v dplyr   1.0.7
## v tidyr   1.1.4     v stringr 1.4.0
## v readr   2.1.1     v forcats 0.5.1
```

```
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(lubridate)
```

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

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

```
library(ggplot2)
library(readxl)
library(dplyr)
library(ggmap)
```

```
## Google's Terms of Service: https://cloud.google.com/maps-platform/terms/.
```

```
## Please cite ggmap if you use it! See citation("ggmap") for details.
```

```
cyclist_df_1 <- read_excel("C:\\Users\\Dacs\\Documents\\R Working Directory\\Case Study Cyclist\\XLS Fil
cyclist_df_2 <- read_excel("C:\\Users\\Dacs\\Documents\\R Working Directory\\Case Study Cyclist\\XLS Fil
cyclist_df_3 <- read_excel("C:\\Users\\Dacs\\Documents\\R Working Directory\\Case Study Cyclist\\XLS Fil
cyclist_df_4 <- read_excel("C:\\Users\\Dacs\\Documents\\R Working Directory\\Case Study Cyclist\\XLS Fil
cyclist_df_5 <- read_excel("C:\\Users\\Dacs\\Documents\\R Working Directory\\Case Study Cyclist\\XLS Fil
cyclist_df_6 <- read_excel("C:\\Users\\Dacs\\Documents\\R Working Directory\\Case Study Cyclist\\XLS Fil
cyclist_df_7 <- read_excel("C:\\Users\\Dacs\\Documents\\R Working Directory\\Case Study Cyclist\\XLS Fil
cyclist_df_8 <- read_excel("C:\\Users\\Dacs\\Documents\\R Working Directory\\Case Study Cyclist\\XLS Fil
cyclist_df_9 <- read_excel("C:\\Users\\Dacs\\Documents\\R Working Directory\\Case Study Cyclist\\XLS Fil
cyclist_df_10 <- read_excel("C:\\Users\\Dacs\\Documents\\R Working Directory\\Case Study Cyclist\\XLS Fi
cyclist_df_11 <- read_excel("C:\\Users\\Dacs\\Documents\\R Working Directory\\Case Study Cyclist\\XLS Fi
cyclist_df_12 <- read_excel("C:\\Users\\Dacs\\Documents\\R Working Directory\\Case Study Cyclist\\XLS Fi


Sys.setlocale("LC_TIME", "C")
```

```
## [1] "C"
```

Joining all the data in one file.

Checking out the data.

```
# Statistical summary of data.

summary(cyclist_df)
```

```
##    ride_id          rideable_type        started_at
##  Length:5479096     Length:5479096     Min.   :2020-12-01 00:01:15
##  Class :character   Class :character   1st Qu.:2021-05-31 17:13:15
##  Mode  :character   Mode  :character   Median :2021-07-25 12:16:02
##                                        Mean   :2021-07-17 15:43:28
##                                        3rd Qu.:2021-09-15 16:30:21
##                                        Max.   :2021-11-30 23:59:56
##
##     ended_at                   start_station_name start_station_id
##  Min.   :2020-11-25 07:40:56   Length:5479096     Length:5479096
##  1st Qu.:2021-05-31 17:45:37   Class :character   Class :character
##  Median :2021-07-25 12:42:07   Mode  :character   Mode  :character
##  Mean   :2021-07-17 16:03:38
##  3rd Qu.:2021-09-15 16:48:19
##  Max.   :2021-12-02 06:41:33
##
##  end_station_name   end_station_id      start_lat          start_lng
##  Length:5479096     Length:5479096     Length:5479096     Length:5479096
##  Class :character   Class :character   Class :character   Class :character
```

```
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
##
##    end_lat            end_lng          member_casual
##  Length:5479096     Length:5479096     Length:5479096
##  Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character
##
##
##
##
##   ride_length                    day_of_week
##  Min.   :1899-12-30 23:01:58   Min.   :1.0
##  1st Qu.:1899-12-31 00:06:50   1st Qu.:2.0
##  Median :1899-12-31 00:12:08   Median :4.0
##  Mean   :1899-12-31 00:22:08   Mean   :4.1
##  3rd Qu.:1899-12-31 00:22:00   3rd Qu.:6.0
##  Max.   :1900-02-07 20:24:09   Max.   :7.0
##  NA's   :378
```

Understanding and Cleaning Data

I started out by separating the date from "started_at" into year, month and day, along with a new column specifying the day of the week.

```
# The default format is yyyy-mm-dd

cyclist_df$date <- as.Date(cyclist_df$started_at)
cyclist_df$month <- format(as.Date(cyclist_df$date), "%m")
cyclist_df$day <- format(as.Date(cyclist_df$date), "%d")
cyclist_df$year <- format(as.Date(cyclist_df$date), "%Y")
cyclist_df$day_of_week <- format(as.Date(cyclist_df$date), "%A")
```

Dropping NA's

```
cyclist_df <- drop_na(cyclist_df)
```

Fixing the ride_length column to show the time in seconds and converting it to numeric.

```
cyclist_df$ride_length <- difftime(cyclist_df$ended_at, cyclist_df$started_at)

str(cyclist_df$ride_length)
```

```
##  'difftime' num [1:4525527] 433 272 587 537 ...
##  - attr(*, "units")= chr "secs"
```

```
cyclist_df$ride_length <- as.numeric(cyclist_df$ride_length)
is.numeric(cyclist_df$ride_length)
```

```
## [1] TRUE
```

The data frame includes a few hundred entries when bikes were taken out of docks and checked for quality by Divvy or ride_length was negative.

```
cyclist_df_v2 <- cyclist_df [!(cyclist_df$start_station_name == 'HQ QR' | cyclist_df$ride_length <0),]
```

Comparing the nr of entries for casuals and members.

```
table(cyclist_df_v2$member_casual)
```

```
##
##  casual  member
## 2027751 2497605
```

Descriptive statistics for ride_length.

```
summary(cyclist_df_v2$ride_length)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##       0     421     739    1317    1338 3356649
```

Compare members and casual users.

```
aggregate(cyclist_df_v2$ride_length ~ cyclist_df_v2$member_casual, FUN = mean)
```

```
##   cyclist_df_v2$member_casual cyclist_df_v2$ride_length
## 1                      casual                  1957.260
## 2                      member                   797.444
```

```
aggregate(cyclist_df_v2$ride_length ~ cyclist_df_v2$member_casual, FUN = median)
```

```
##   cyclist_df_v2$member_casual cyclist_df_v2$ride_length
## 1                      casual                      1005
## 2                      member                       588
```

```
aggregate(cyclist_df_v2$ride_length ~ cyclist_df_v2$member_casual, FUN = max)
```

```
##   cyclist_df_v2$member_casual cyclist_df_v2$ride_length
## 1                      casual                   3356649
## 2                      member                     89990
```

See the average ride time by each day for members vs casual users.

```
aggregate(cyclist_df_v2$ride_length ~ cyclist_df_v2$member_casual + cyclist_df_v2$day_of_week, FUN = mea
```

```
##    cyclist_df_v2$member_casual cyclist_df_v2$day_of_week
## 1                       casual                    Friday
## 2                       member                    Friday
## 3                       casual                    Monday
## 4                       member                    Monday
## 5                       casual                  Saturday
## 6                       member                  Saturday
## 7                       casual                    Sunday
## 8                       member                    Sunday
## 9                       casual                  Thursday
## 10                      member                  Thursday
## 11                      casual                   Tuesday
## 12                      member                   Tuesday
## 13                      casual                 Wednesday
## 14                      member                 Wednesday
##    cyclist_df_v2$ride_length
## 1                  1864.8796
```

```
## 2                   774.1012
## 3                  1964.9903
## 4                   769.1446
## 5                  2100.9427
## 6                   893.4159
## 7                  2264.3143
## 8                   917.0686
## 9                  1683.5258
## 10                  749.3354
## 11                 1727.5484
## 12                  747.4657
## 13                 1695.0272
## 14                  755.2309
```

Since the days are out order, I will go ahead and fix that.

```
cyclist_df_v2$day_of_week <- ordered(cyclist_df_v2$day_of_week, levels=c("Sunday", "Monday", "Tuesday",
```

Checking if it is fixed.

```
aggregate(cyclist_df_v2$ride_length ~ cyclist_df_v2$member_casual + cyclist_df_v2$day_of_week, FUN = mea
```

```
##    cyclist_df_v2$member_casual cyclist_df_v2$day_of_week
## 1                       casual                    Sunday
## 2                       member                    Sunday
## 3                       casual                    Monday
## 4                       member                    Monday
## 5                       casual                   Tuesday
## 6                       member                   Tuesday
## 7                       casual                 Wednesday
## 8                       member                 Wednesday
## 9                       casual                  Thursday
## 10                      member                  Thursday
## 11                      casual                    Friday
## 12                      member                    Friday
## 13                      casual                  Saturday
## 14                      member                  Saturday
##    cyclist_df_v2$ride_length
## 1                  2264.3143
## 2                   917.0686
## 3                  1964.9903
## 4                   769.1446
## 5                  1727.5484
## 6                   747.4657
## 7                  1695.0272
## 8                   755.2309
## 9                  1683.5258
## 10                  749.3354
## 11                 1864.8796
## 12                  774.1012
## 13                 2100.9427
## 14                  893.4159
```

Analyze ridership data by type and weekday.

```
cyclist_df_v2 %>%
  # groups by user type and weekday
  group_by(member_casual, day_of_week) %>%
  # calculates the number of rides
  summarise(number_of_rides = n()
            # calculates the average duration
            ,average_duration = mean(ride_length)) %>%
  # sorts the data
  arrange(member_casual, day_of_week)
```

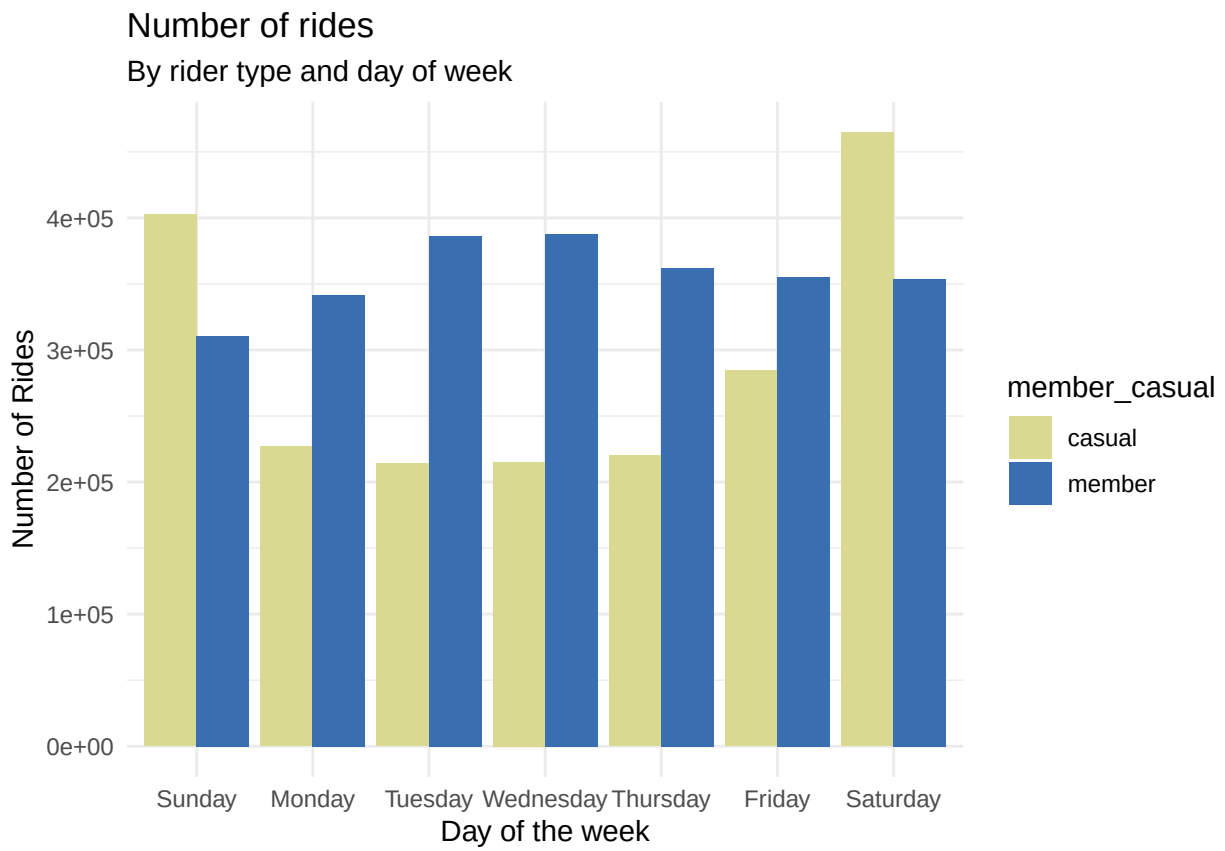
```
## `summarise()` has grouped output by 'member_casual'. You can override using the `.groups` argument.
```

```
## # A tibble: 14 x 4
## # Groups:   member_casual [2]
##    member_casual day_of_week number_of_rides average_duration
##    <chr>         <ord>                 <int>            <dbl>
##  1 casual        Sunday               402526            2264.
##  2 casual        Monday               226837            1965.
##  3 casual        Tuesday              214196            1728.
##  4 casual        Wednesday            215210            1695.
##  5 casual        Thursday             219883            1684.
##  6 casual        Friday               284477            1865.
##  7 casual        Saturday             464622            2101.
##  8 member        Sunday               310483             917.
##  9 member        Monday               341706             769.
## 10 member        Tuesday              386435             747.
## 11 member        Wednesday            387918             755.
## 12 member        Thursday             362173             749.
## 13 member        Friday               355300             774.
## 14 member        Saturday             353590             893.
```

Visualize the previous data.

```
# Let's visualize the number of rides by rider type
cyclist_df_v2 %>%
  group_by(member_casual, day_of_week) %>%
  summarise(number_of_rides = n()
            ,average_duration = mean(ride_length)) %>%
  arrange(member_casual, day_of_week)  %>%
  ggplot() +
  geom_col(aes(x = day_of_week, y = number_of_rides, fill = member_casual), position = "dodge") +
  labs(title = "Number of rides",
       subtitle="By rider type and day of week",
       x="Day of the week",
       y="Number of Rides") +
  scale_fill_manual(values = c("casual" = "#DAD992", "member" = "#396EB0")) +
  theme_minimal()
```

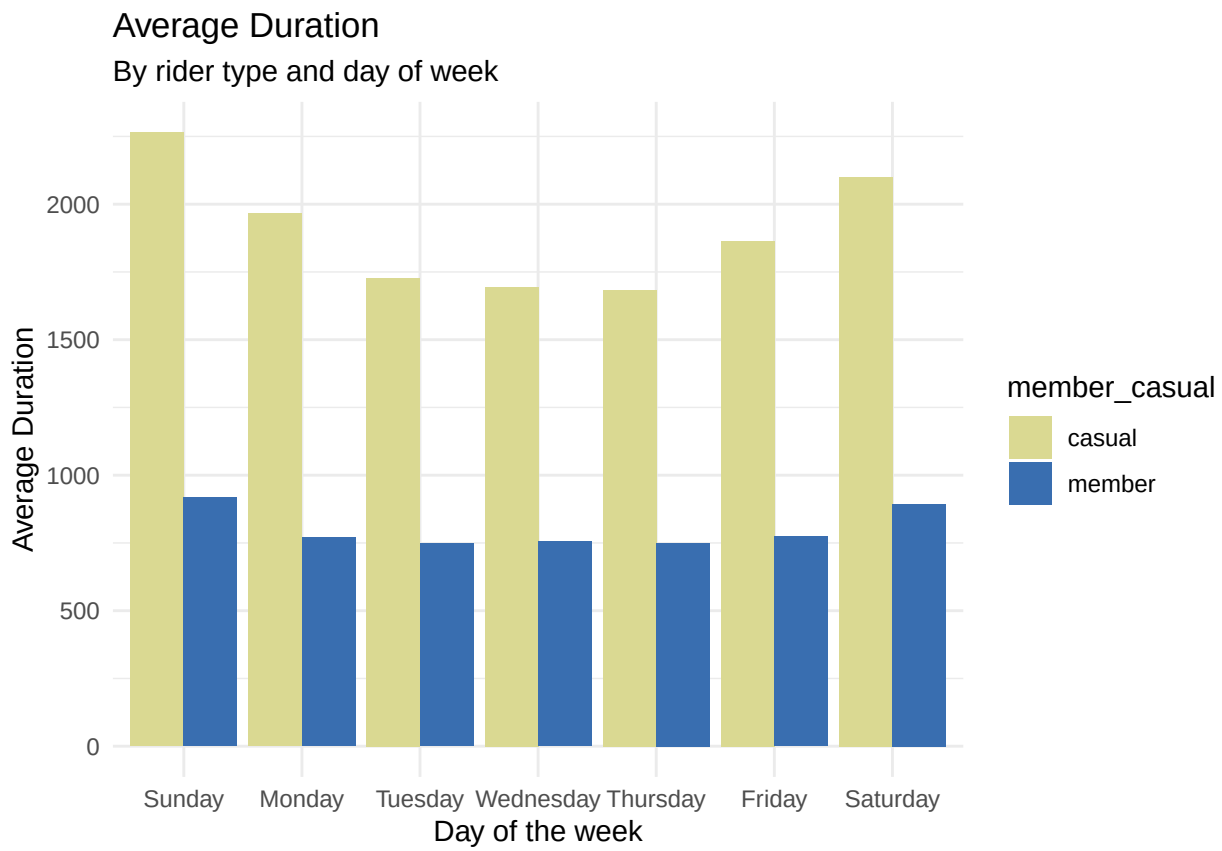
```
## `summarise()` has grouped output by 'member_casual'. You can override using the `.groups` argument.
```



Next, I will check the Average duration by rider type and day of the week to see on which days people take longer trips.

```
cyclist_df_v2 %>%
  group_by(member_casual, day_of_week) %>%
  summarise(number_of_rides = n()
            ,average_duration = mean(ride_length)) %>%
  ggplot() +
  geom_col(aes(x=day_of_week, y=average_duration, fill=member_casual), position = "dodge") +
  labs(title = "Average Duration",
       subtitle="By rider type and day of week",
       x="Day of the week",
       y="Average Duration") +
  scale_fill_manual(values = c("casual" = "#DAD992", "member" = "#396EB0")) +
  theme_minimal()
```

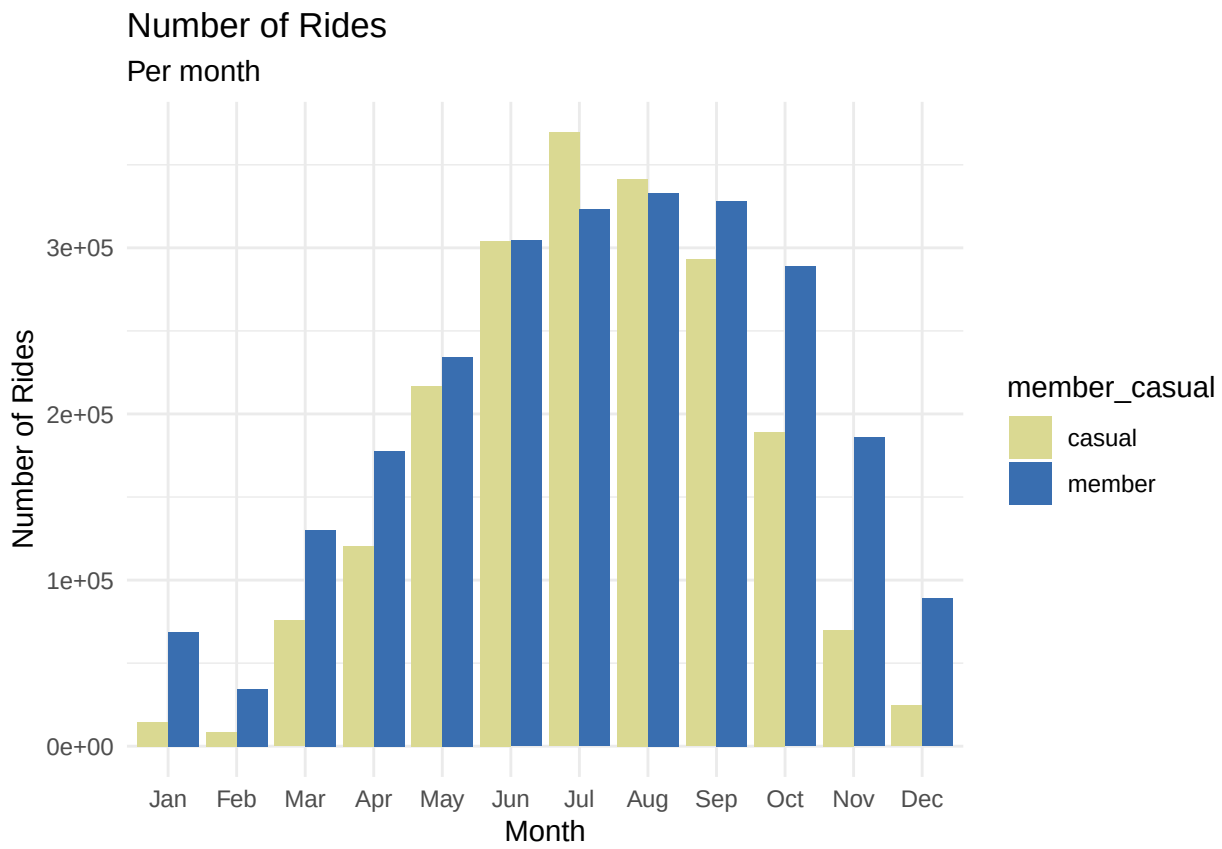
```
## `summarise()` has grouped output by 'member_casual'. You can override using the `.groups` argument.
```



Checking out the number of rides per month.

```
cyclist_df_v2 %>%
  # I use this line instead of the column "month" so I can get the name of the month instead of the num
  mutate(month = month(started_at, label = TRUE)) %>%
  group_by(member_casual, month) %>%
  summarise(number_of_rides = n()) %>%
  arrange(member_casual, month) %>%
  ggplot() +
  geom_col(aes(x= month, y = number_of_rides, fill = member_casual), position="dodge") +
  labs(title = "Number of Rides",
       subtitle = "Per month",
       x="Month",
       y="Number of Rides") +
  scale_fill_manual(values = c("casual" = "#DAD992", "member" = "#396EB0")) +
  theme_minimal()
```

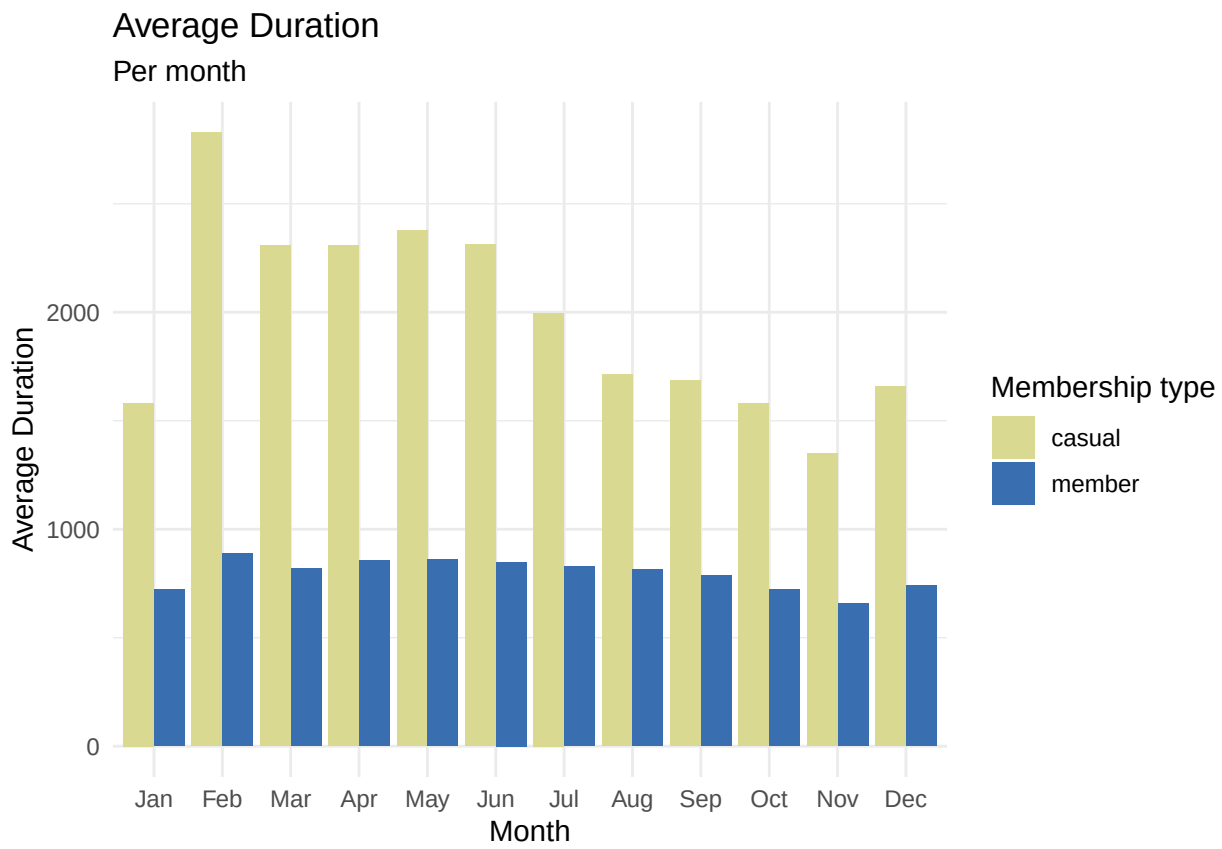
```
## `summarise()` has grouped output by 'member_casual'. You can override using the `.groups` argument.
```



Next, I will check the Average duration by rider type and month to see on which months people take longer trips.

```
cyclist_df_v2 %>%
  mutate(month = month(started_at, label = TRUE)) %>%
  group_by(member_casual, month) %>%
  summarise(average_duration = mean(ride_length)) %>%
  arrange(member_casual, month) %>%
  ggplot() +
  geom_col(aes(x= month, y = average_duration, fill = member_casual), position="dodge") +
  labs(title = "Average Duration",
       subtitle = "Per month",
       fill = "Membership type",
       x="Month",
       y="Average Duration") +
  scale_fill_manual(values = c("casual" = "#DAD992", "member" = "#396EB0")) +
  theme_minimal()
```

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



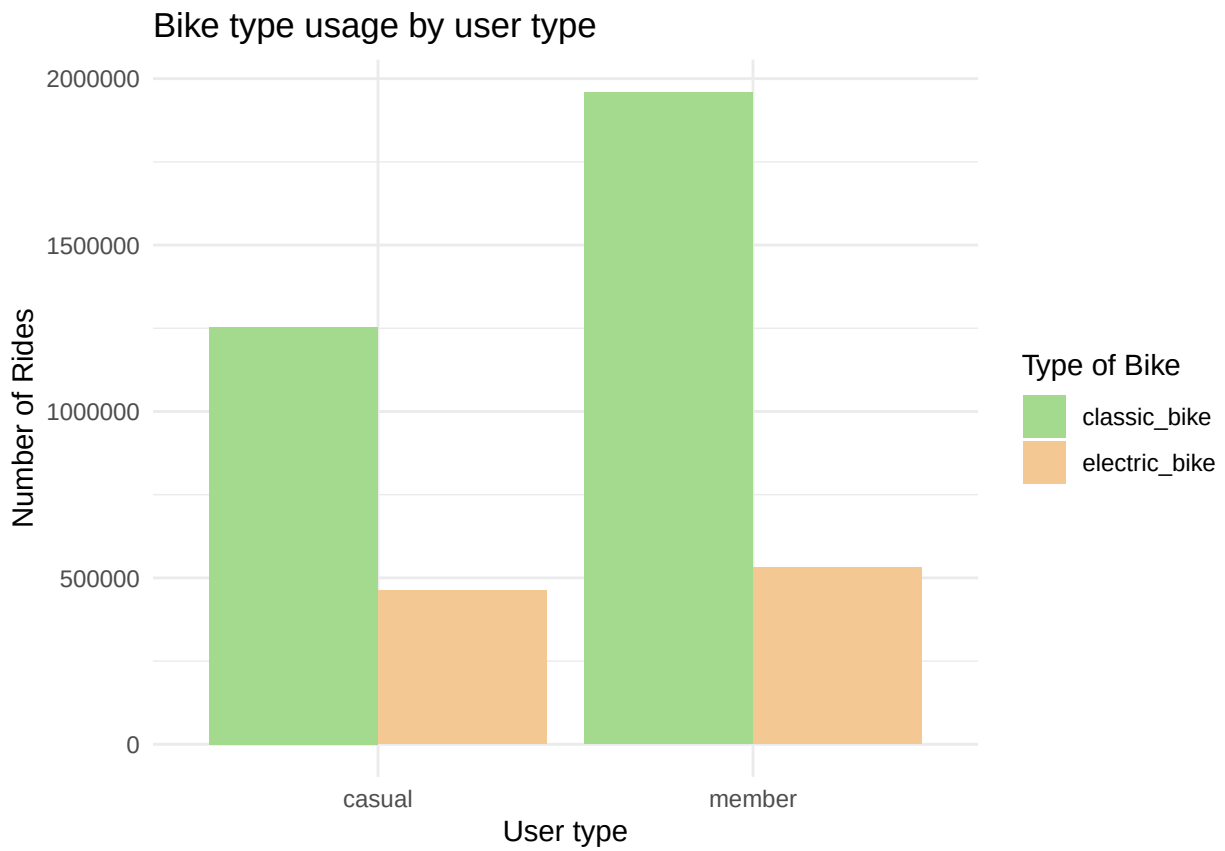
Creating a sub data set for bike type in order to exclude docked bikes.

```
bike_type <- cyclist_df_v2 %>% filter(rideable_type=="classic_bike" | rideable_type=="electric_bike")
```

Now we check the type of bike used by each user type.

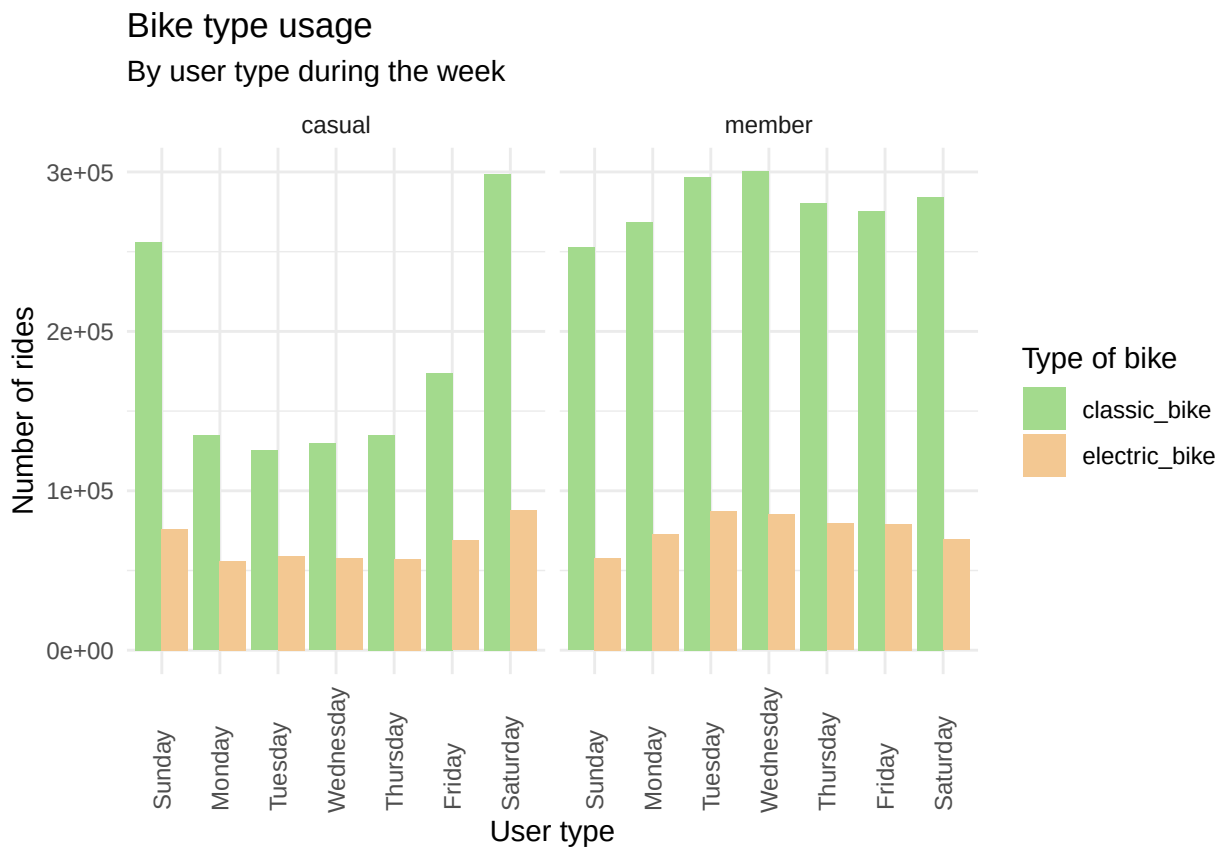
```
bike_type %>%
  group_by(member_casual,rideable_type) %>%
  summarise(totals=n())  %>%
  ggplot()+
  geom_col(aes(x=member_casual,y=totals,fill=rideable_type), position = "dodge") +
  labs(title = "Bike type usage by user type",
       x="User type",
       y="Number of Rides",
       fill="Type of Bike") +
  scale_fill_manual(values = c("classic_bike" = "#A3DA8D","electric_bike" = "#F3C892")) +
  theme_minimal()
```

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



Next, we use a facet wrap to compare the usage by both user types during a week.

```
bike_type %>%
  group_by(member_casual,rideable_type,day_of_week) %>%
  summarise(totals=n(), .groups="drop") %>%
  ggplot() +
  geom_col(aes(x=day_of_week,y=totals, fill=rideable_type), position = "dodge") +
  facet_wrap(~member_casual) +
  labs(title = "Bike type usage",
       subtitle="By user type during the week",
       x="User type",
       y="Number of rides",
       fill="Type of bike") +
  scale_fill_manual(values = c("classic_bike" = "#A3DA8D","electric_bike" = "#F3C892")) +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 90, vjust = 0.5, hjust=0.05))
```

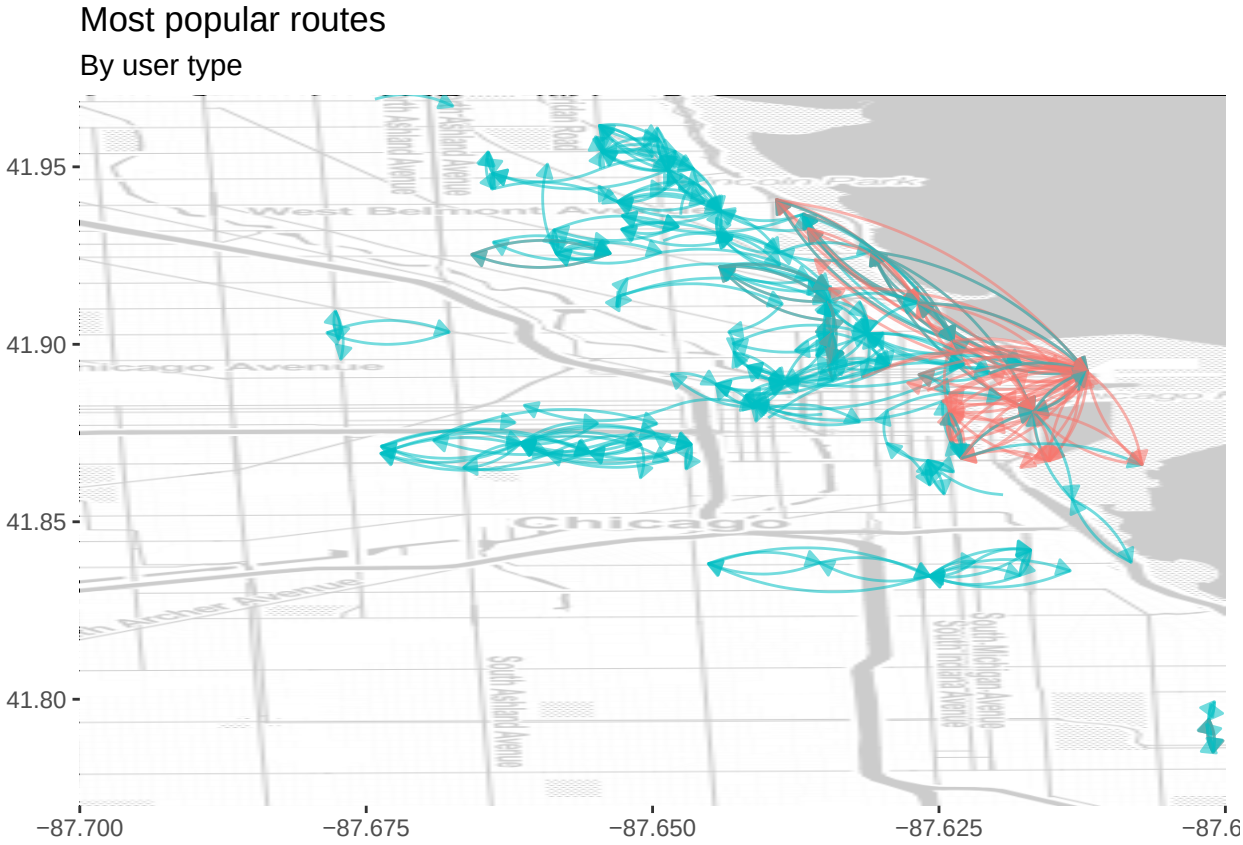



```
# Have to change both columns of latitude and both columns of longitude to numeric so it would work as

bike_type$start_lat = as.numeric(bike_type$start_lat)
bike_type$start_lng = as.numeric(bike_type$start_lng)
bike_type$end_lat = as.numeric(bike_type$end_lat)
bike_type$end_lng = as.numeric(bike_type$end_lng)
```

```
# Selecting the most common routes, otherwise the map would become unreadable with too many routes.

coordinates_table <- bike_type %>%
filter(start_lng != end_lng & start_lat != end_lat) %>%
group_by(start_lng, start_lat, end_lng, end_lat, member_casual, rideable_type) %>%
summarise(total = n()) %>%
filter(total > 300)

# Creating a boundary box for the map.

chi_bb <- c(
  left = -87.70,
  bottom = 41.77,
  right = -87.6,
  top = 41.97
)

# Getting Chicago's map with the get_statenmap function from the package ggmap with the boundary box at
```

```
chicago_stamen <- get_stamenmap(
  bbox = chi_bb,
  zoom = 12,
  maptype = "toner"
)

# Then we plot the data on the map.

ggmap(chicago_stamen,darken = c(0.8, "white")) +
  # Draws the curved lines representing the segments
   geom_curve(coordinates_table,
              mapping = aes(x = start_lng,
                            y = start_lat,
                            xend = end_lng,
                            yend = end_lat,
                            alpha= 0.1,
                            color=member_casual),
              size = 0.5,
              curvature = .2,
              # We use arrows so we could easily identify where the route starts and ends.
              arrow = arrow(length=unit(0.2,"cm"),
                            ends="first",
                            type = "closed")) +
    coord_cartesian() +
    labs(title = "Most popular routes",
         subtitle = "By user type",
         x=NULL,
         y=NULL,
         color="User type") +
    theme(legend.position="none")
```

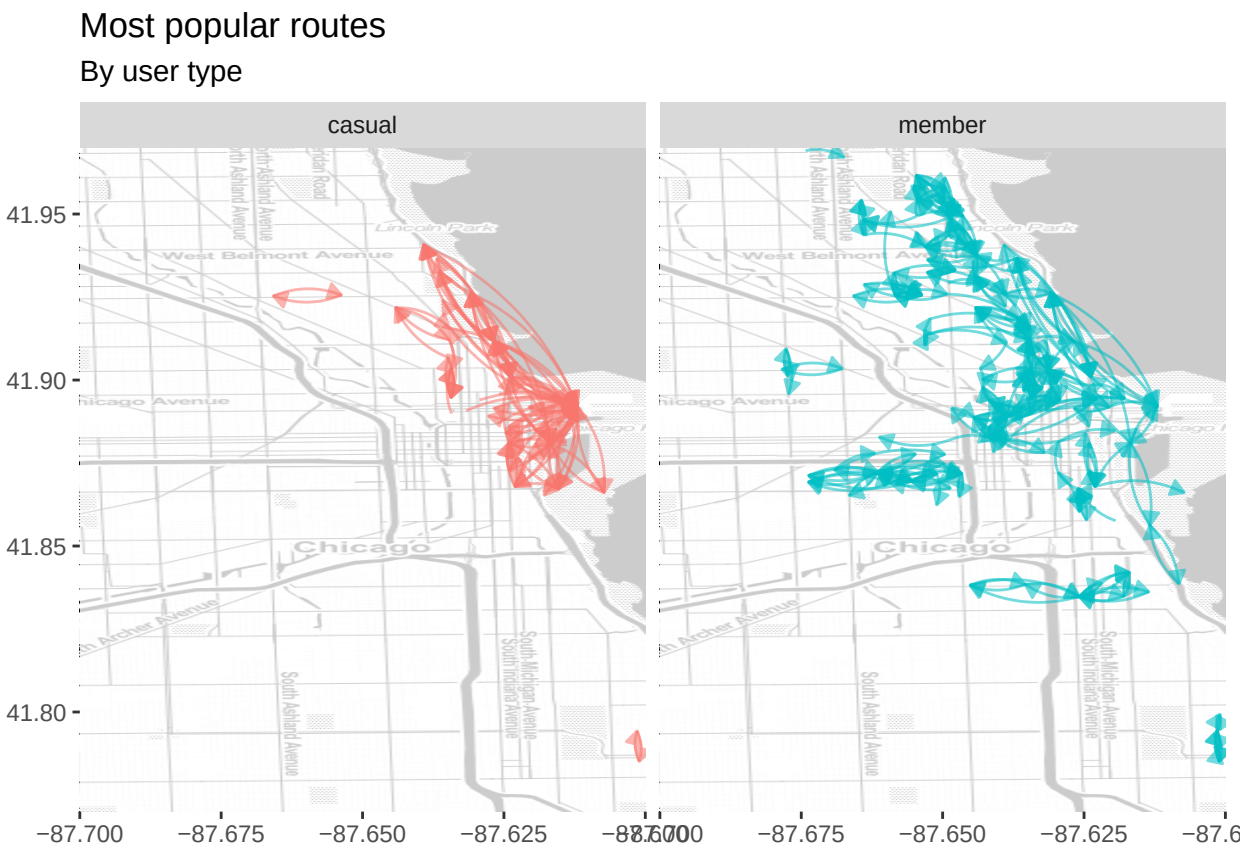
```
## Warning: Removed 74 rows containing missing values (geom_curve).
```



As you can see, even showing up only the most common routes, the map has alot of information, so we will divide it by user type.

```
# Then we plot both with the addition of the facet_wrap function.

ggmap(chicago_stamen,darken = c(0.8, "white")) +
   geom_curve(coordinates_table,
              mapping = aes(x = start_lng,
                            y = start_lat,
                            xend = end_lng,
                            yend = end_lat,
                            alpha= 0.5,
                            color=member_casual),
              size = 0.5,
              curvature = .2,
              arrow = arrow(length=unit(0.2,"cm"),
                            ends="first",
                            type = "closed")) +
  facet_wrap(~member_casual) +
    coord_cartesian() +
    labs(title = "Most popular routes",
         subtitle = "By user type",
         x=NULL,
         y=NULL,
         color="User type") +
    theme(legend.position="none")
```

```
## Coordinate system already present. Adding new coordinate system, which will replace the existing one
## Warning: Removed 74 rows containing missing values (geom_curve).
```



Conclusion

From this analysis we can conclude that: * * * *