Cyclistic Bike Share Company

A CASE STUDY BY PIERPAOLO ZOTTI

Ilha de Guaratiba - Rio de Janeiro

+55 21 98130 2437 | zotti.pierpaolo@outlook.it | pierpaolozotti.com | PierpaoloZotti | pierpaolo-zotti-186399253

Introduction

Welcome to my personal analysis of the ***Cyclistic Bike Share Company***. In order to answer the *key business questions*, I will follow the steps of the data analysis process: **ask**, **prepare**, **process**, **analyze**, **share**, and **act**.

About the company

In 2016, **Cyclistic** launched a successful bike-share offering. Since then, the program has grown to a fleet of 5,824 bicycles that are geotracked and locked into a network of 692 stations across Chicago. The bikes can be unlocked from one station and returned to any other station in the system anytime.

Cyclistic's finance analysts have concluded that annual members are much more profitable than casual riders. Although the pricing flexibility helps Cyclistic attract more customers, Moreno believes that maximizing the number of annual members will be key to future growth. Rather than creating a marketing campaign that targets all-new customers, is a very good chance to convert casual riders into members.

Company goal

The company final goal is to increase the number of annual member. My task as junior analyst is presentig a clear visualization and some recommendations about the strategy to accomplish that goal.

Data source

The data for this analysis is the Divvy dataset, and we will use the last 12 months to accomplish our scope.

Installing required packages

For this case study we will use`tidyverse` for importing and wrangling data, `lubridate` for data manipulation, and `ggplot2` for data visualizations.

```
install.packages("tidyverse")
```

```
## pacchetto 'tidyverse' aperto con successo con controllo somme MD5
##
## I pacchetti binari scaricati sono in
##  C:\Users\zotti\AppData\Local\Temp\Rtmp4UTGDU\downloaded_packages
```

```
install.packages("lubridate")
```

```
## pacchetto 'lubridate' aperto con successo con controllo somme MD5
##
## I pacchetti binari scaricati sono in
##  C:\Users\zotti\AppData\Local\Temp\Rtmp4UTGDU\downloaded_packages
```

```
install.packages("ggplot2")
```

```
## pacchetto 'ggplot2' aperto con successo con controllo somme MD5
##
## I pacchetti binari scaricati sono in
##  C:\Users\zotti\AppData\Local\Temp\Rtmp4UTGDU\downloaded_packages
```

```
install.packages("scales")
```

```
## pacchetto 'scales' aperto con successo con controllo somme MD5
##
## I pacchetti binari scaricati sono in
##  C:\Users\zotti\AppData\Local\Temp\Rtmp4UTGDU\downloaded_packages
```

```
library(tidyverse)
library(lubridate)
library(ggplot2)
library(scales)
```

Check and set the working directoy

```
getwd() #Display the actual working directory
```

```
## [1] "C:/Users/zotti/OneDrive/Desktop/Case Study 1"
```

```
setwd("/Users/zotti/OneDrive/Desktop/Case Study 1/csv")
```

Collect data from csv files

```
td2021_10 <- read_csv("csv/202110-divvy-tripdata.csv")
td2021_11 <- read_csv("csv/202111-divvy-tripdata.csv")
td2021_12 <- read_csv("csv/202112-divvy-tripdata.csv")
td2022_01 <- read_csv("csv/202201-divvy-tripdata.csv")
td2022_02 <- read_csv("csv/202202-divvy-tripdata.csv")
td2022_03 <- read_csv("csv/202203-divvy-tripdata.csv")
td2022_04 <- read_csv("csv/202204-divvy-tripdata.csv")
td2022_05 <- read_csv("csv/202205-divvy-tripdata.csv")
td2022_06 <- read_csv("csv/202206-divvy-tripdata.csv")
td2022_07 <- read_csv("csv/202207-divvy-tripdata.csv")
td2022_08 <- read_csv("csv/202208-divvy-tripdata.csv")
td2022_09 <- read_csv("csv/202209-divvy-publictripdata.csv")
```

Check data consistency

Checking column names with`colnames()` function and the structure with `str()` I notice that all the columns of each dataset has a consistent name and data type

Join all dataset together

```
trips_df <- bind_rows(td2021_10,
                      td2021_11,
                      td2021_12,
                      td2022_01,
                      td2022_02,
                      td2022_03,
                      td2022_04,
                      td2022_05,
                      td2022_06,
                      td2022_07,
                      td2022_08,
                      td2022_09)
```

Checking for NULL values

Using the summary() function on the dataset I notice that there are 5844 rows of NULL values only on the Lat and Lng Columns.This is only the 1% of the total rows of our dataset. We can exclude these rows. But looking at the table with th View() function I notice that there are other N/A values on the start_station_name, end_statiom_name, start_station_id.

I will use the colSums(is.na()) function to count how many N/A values we have:

```
colSums(is.na(trips_df))
```

```
##            ride_id      rideable_type         started_at           ended_at
##                  0                  0                  0                  0
## start_station_name   start_station_id   end_station_name     end_station_id
##             895032             895032             958227             958227
##          start_lat          start_lng            end_lat            end_lng
##                  0                  0               5844               5844
##      member_casual
##                  0
```

This is 16% of total rows.

Excluding unnecessary columns

For this case study we don't need the information start_station_id and end_station_id

```
trips_df <- trips_df %>%
  select(-c(start_station_id,
            end_station_id))
```

Check member_casual and rideable_type field for unwanted records

```
print("Bike type:")
```

```
## [1] "Bike type:"
```

```
table(trips_df$rideable_type)
```

```
##
##  classic_bike   docked_bike electric_bike
##       2740516        192475       2895244
```

```
print("User type:")
```

```
## [1] "User type:"
```

```
table(trips_df$member_casual)
```

```
##
##  casual  member
## 2401286 3426949
```

Add column date, year, month, day, and day_of_week

This is for aggregation purpose. By now only ride level aggregation is possible.

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

Add column ride_lengthusing difftime() function

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

Now we assure that ride_length must be numeric

```
trips_df$ride_length <- as.numeric(as.character(trips_df$ride_length))
```

Clenaing data

I noticed that some rows of started_at is greater or equals than ended_at, generating negative or zero values of ride_length (Inconsistent values). In addition I will remove all the rides with more than 24 hours, cause based on Divvy site, after a 24 hours period the customer may be charged with a lost or stolen bike fee of 1200$

```
trips_df <- trips_df[!(trips_df$ride_length<=1),]
trips_df <- trips_df[!(trips_df$ride_length>24*60*60),]
```

Analysis

I will begin analyzing some values of trip duration for casual and member user:

```
aggregate(trips_df$ride_length ~ trips_df$member_casual, FUN = min) # Shortest ride
```

```
##   trips_df$member_casual trips_df$ride_length
## 1                 casual                    2
## 2                 member                    2
```

```
aggregate(trips_df$ride_length ~ trips_df$member_casual, FUN = mean) #Straight average
```

```
##   trips_df$member_casual trips_df$ride_length
## 1                 casual            1346.3957
## 2                 member             748.9114
```

```
aggregate(trips_df$ride_length ~ trips_df$member_casual, FUN = median) # Midpoint
```

```
##   trips_df$member_casual trips_df$ride_length
## 1                 casual                  805
## 2                 member                  533
```

```
aggregate(trips_df$ride_length ~ trips_df$member_casual, FUN = max) # Longest ride
```

```
##   trips_df$member_casual trips_df$ride_length
## 1                 casual                86395
## 2                 member                86397
```

Now I will calculate the average ride_length for users by day_of_week.

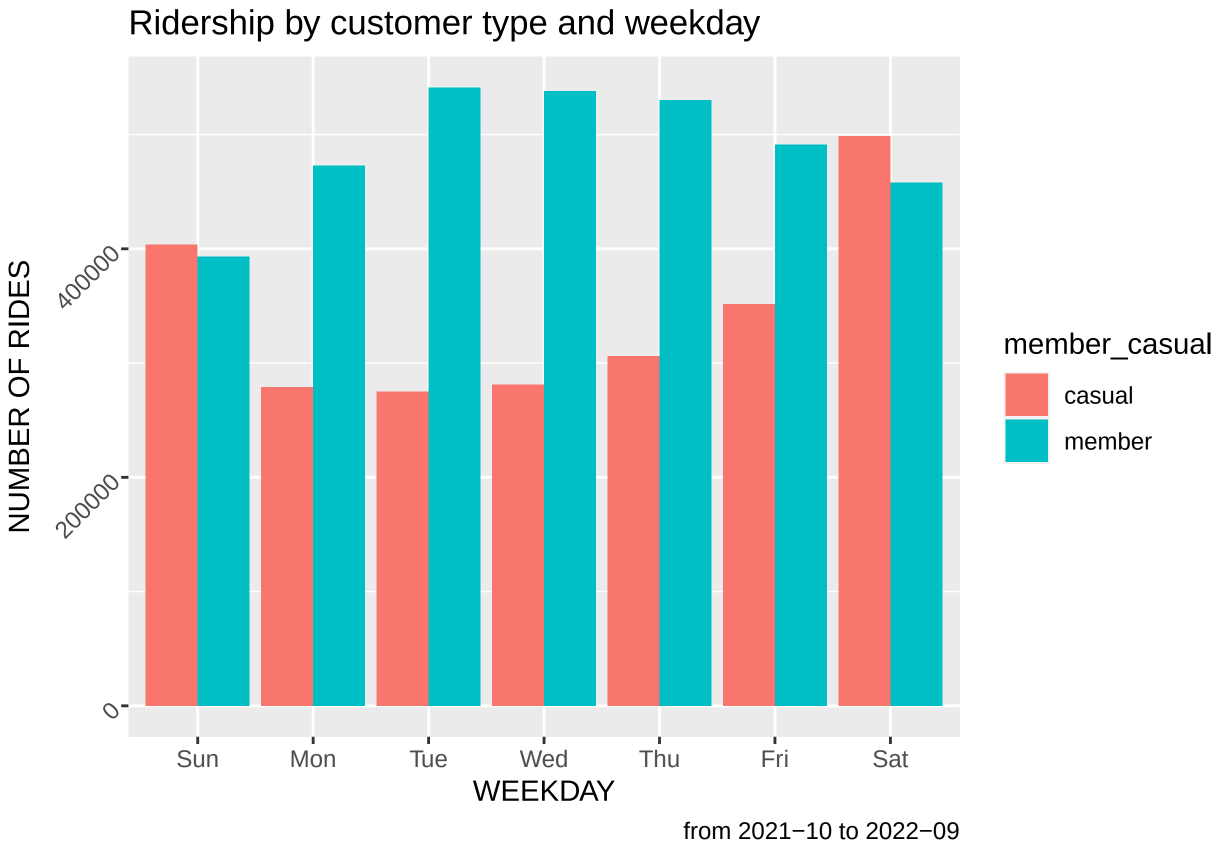
```
trips_df$day_of_week <- ordered(trips_df$day_of_week, levels=c("Sunday", "Monday", "Tuesday", "We

trips_df %>%
  aggregate(ride_length ~ member_casual + day_of_week, FUN = mean)
```

```
##    member_casual day_of_week ride_length
## 1         casual      Sunday   1547.2623
## 2         member      Sunday    830.0375
## 3         casual      Monday   1373.4595
## 4         member      Monday    724.4181
## 5         casual     Tuesday   1205.8526
## 6         member     Tuesday    713.0536
## 7         casual   Wednesday   1162.8130
## 8         member   Wednesday    712.9834
## 9         casual    Thursday   1189.5318
## 10        member    Thursday    720.8804
## 11        casual      Friday   1260.9086
## 12        member      Friday    734.5120
## 13        casual    Saturday   1506.1281
## 14        member    Saturday    837.0219
```

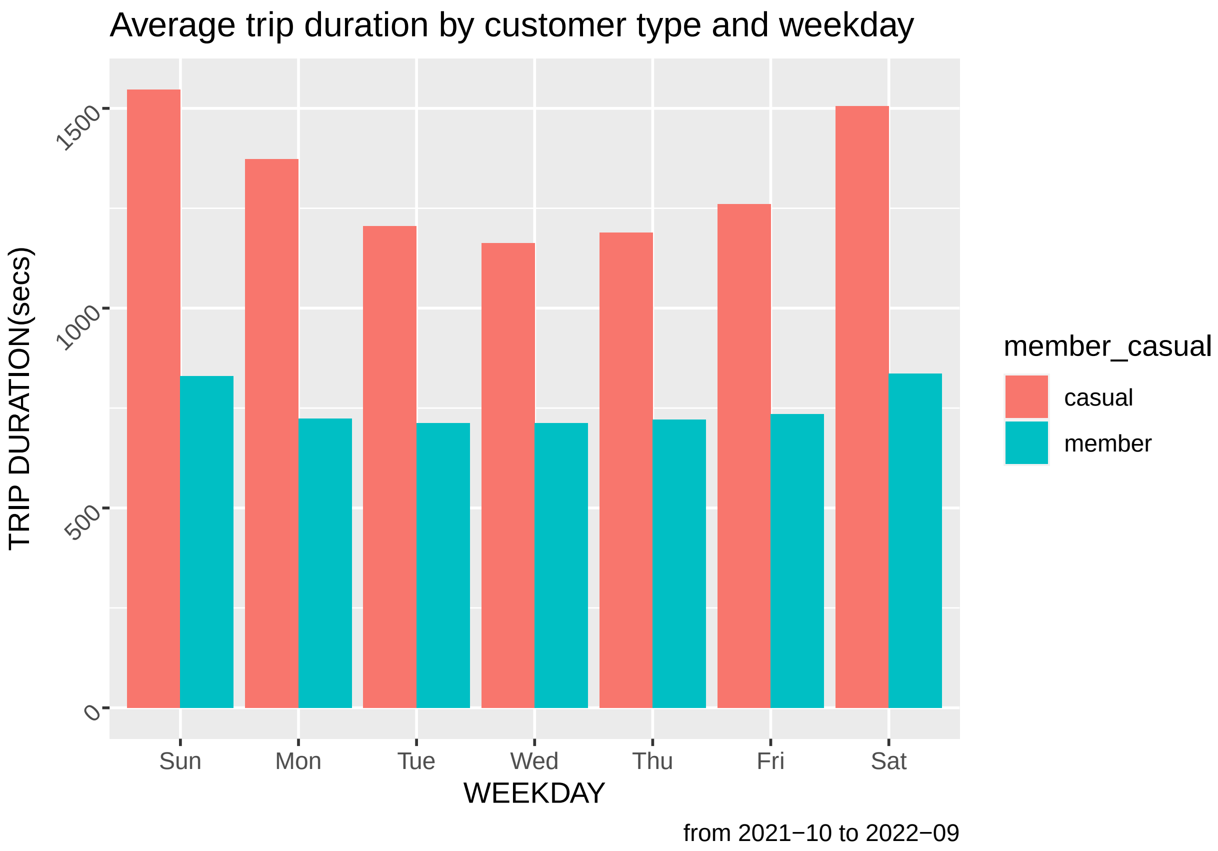
```
trips_df %>%
  mutate(weekday = wday(started_at, label = TRUE)) %>%  #creates weekday field
  group_by(member_casual, weekday) %>%  #groups by usertype and weekday
  summarise(number_of_rides = n(),average_duration = mean(ride_length)) %>%
  arrange(member_casual, weekday) %>%
  as.data.frame()
```

```
##    member_casual weekday number_of_rides average_duration
## 1         casual     Sun          403941        1547.2623
## 2         casual     Mon          279141        1373.4595
## 3         casual     Tue          275175        1205.8526
## 4         casual     Wed          281075        1162.8130
## 5         casual     Thu          306049        1189.5318
## 6         casual     Fri          351658        1260.9086
## 7         casual     Sat          498658        1506.1281
## 8         member     Sun          393373         830.0375
## 9         member     Mon          472830         724.4181
## 10        member     Tue          541236         713.0536
## 11        member     Wed          538255         712.9834
## 12        member     Thu          530302         720.8804
## 13        member     Fri          491218         734.5120
## 14        member     Sat          457976         837.0219
```

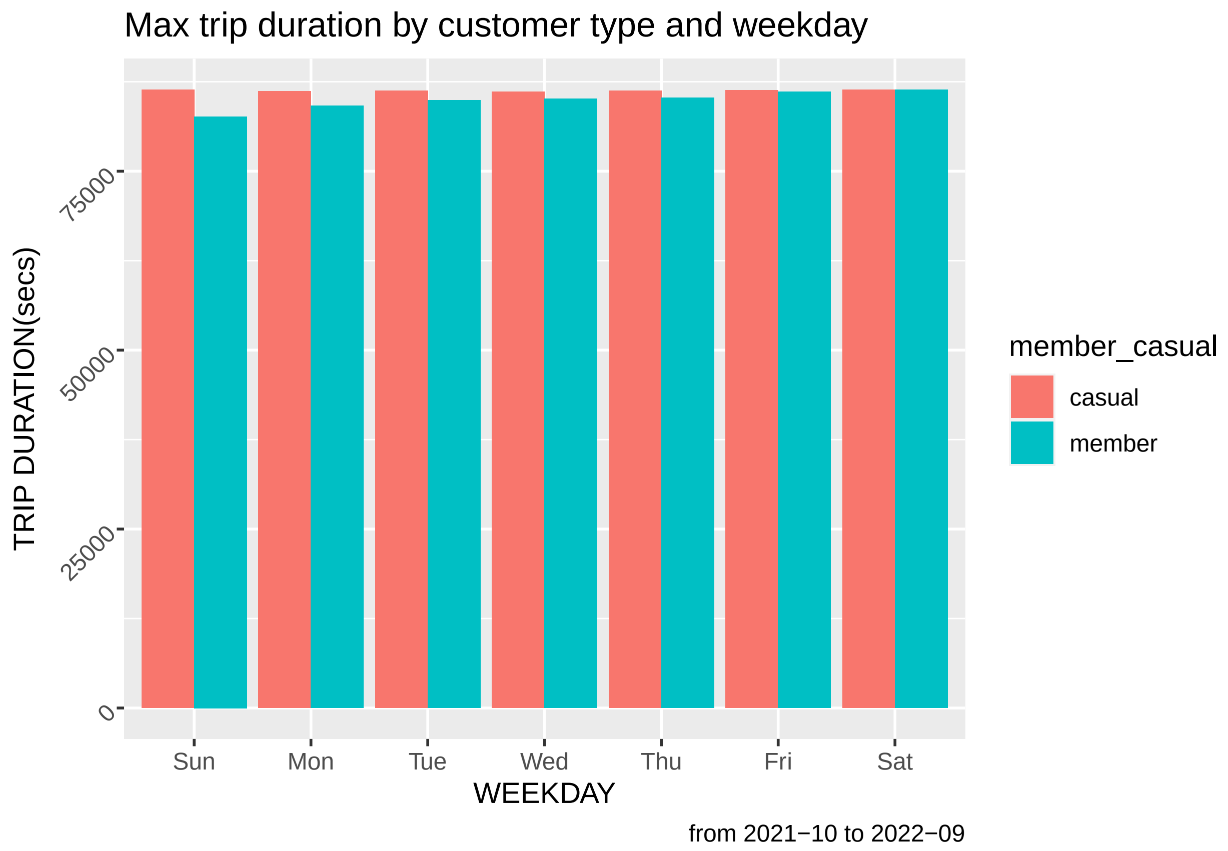
```
trips_df %>%
  mutate(weekday = wday(started_at, label = TRUE)) %>%
  group_by(member_casual, weekday) %>%
  summarise(number_of_rides = n()
            ,average_duration = mean(ride_length)) %>%
  arrange(member_casual, weekday)  %>%
  ggplot(aes(x = weekday, y = number_of_rides, fill = member_casual)) +
  geom_col(position = "dodge")+
  labs(title = "Ridership by customer type and weekday",
       caption = "from 2021-10 to 2022-09", x="WEEKDAY" , y="NUMBER OF RIDES")+
  theme(axis.text.y = element_text(angle = 45))+
  scale_y_continuous(labels = function(x) format(x, scientific = FALSE))
```



```
trips_df %>%
  mutate(weekday = wday(started_at, label = TRUE)) %>%
  group_by(member_casual, weekday) %>%
  summarise(number_of_rides = n()
           ,average_duration = mean(ride_length)) %>%
  arrange(member_casual, weekday)  %>%
  ggplot(aes(x = weekday, y = average_duration, fill = member_casual)) +
  geom_col(position = "dodge")+
  labs(title = "Average trip duration by customer type and weekday",
       caption = "from 2021-10 to 2022-09", x="WEEKDAY" , y="TRIP DURATION(secs)")+
  theme(axis.text.y = element_text(angle = 45))+
  scale_y_continuous(labels = function(x) format(x, scientific = FALSE))
```



```
trips_df %>%
  mutate(weekday = wday(started_at, label = TRUE)) %>%
  group_by(member_casual, weekday) %>%
  summarise(number_of_rides = n()
           ,max_duration = max(ride_length)) %>%
  arrange(member_casual, weekday)  %>%
  ggplot(aes(x = weekday, y = max_duration, fill = member_casual)) +
  geom_col(position = "dodge")+
  labs(title = "Max trip duration by customer type and weekday",
       caption = "from 2021-10 to 2022-09", x="WEEKDAY" , y="TRIP DURATION(secs)")+
  theme(axis.text.y = element_text(angle = 45))+
  scale_y_continuous(labels = function(x) format(x, scientific = FALSE))
```



Exporting data for further analysis

```
counts <- aggregate(trips_df$ride_length ~ trips_df$member_casual +
                      trips_df$day_of_week, FUN = mean)
df_by_month <- trips_df %>%
  group_by(month, member_casual) %>%
  drop_na() %>%
  summarize(avg_ride_length = mean(ride_length),
            max_ride_length = max(ride_length))
```

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

```
df_by_bike_user <- trips_df %>%
  group_by(rideable_type, member_casual) %>%
  summarize(number_of_ride = n(), max_duration = max(ride_length),
            avg_duration_secs = round(mean(ride_length), digits = 2))
```

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

```
df_by_lat_lng <- trips_df %>%
  group_by(start_lat, start_lng) %>%
  summarize(number_of_ride = n()) %>%
  filter(number_of_ride>50) %>%
  arrange(-number_of_ride)
```

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

```
df_by_start_station <- trips_df %>%
  group_by(start_station_name, end_station_name) %>%
  summarize(number_of_ride = n()) %>%
  filter(number_of_ride>50) %>%
  arrange(-number_of_ride)
```

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

```
# Create the roundtrip column

trips_df$roundtrip <- ifelse(trips_df$start_station_name == trips_df$end_station_name,1,0)

# Create a dataframe with sum of roundtrip grouped by station, usertype, month and day

df_roundtrip <- trips_df %>%
  filter(roundtrip==1) %>%
  group_by(start_station_name, member_casual, month, day_of_week) %>%
  summarize(number_of_ride = n()) %>%
  arrange(-number_of_ride)
```

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

```
# Create a df with rides longer than 45 min to study income from member's rides

df_renv_member<- trips_df[!(trips_df$ride_length < 45*60),]
df_renv_member <- df_renv_member %>%
  filter(member_casual=="member")

# Create a df with rides longer than 180 min to study income from casual's rides

df_renv_casual<- trips_df[!(trips_df$ride_length < 180*60),]
df_renv_casual <- df_renv_casual %>%
  filter(member_casual=="casual")

# Create a csv file to analize with other software

write.csv(counts, file = 'C:/Users/zotti/OneDrive/Desktop/Case Study 1/csv/avg_ride_lenght.csv')
write.csv(df_by_month, file= 'C:/Users/zotti/OneDrive/Desktop/Case Study 1/csv/df_by_month.csv')
write.csv(df_by_bike_user, file= 'C:/Users/zotti/OneDrive/Desktop/Case Study 1/csv/df_by_bike.csv
write.csv(df_by_start_station, file= 'C:/Users/zotti/OneDrive/Desktop/Case Study 1/csv/df_by_stat
write.csv(df_by_lat_lng, file= 'C:/Users/zotti/OneDrive/Desktop/Case Study 1/csv/df_by_latlng.csv
write.csv(df_roundtrip, file= 'C:/Users/zotti/OneDrive/Desktop/Case Study 1/csv/df_roundtrip.csv'
write.csv(df_renv_casual, file= 'C:/Users/zotti/OneDrive/Desktop/Case Study 1/csv/df_renv_casual.
write.csv(df_renv_member, file= 'C:/Users/zotti/OneDrive/Desktop/Case Study 1/csv/df_renv_member.
```

‘‘‘