Google Data Analytics Capstone Project : Cyclistic Bike-Share Analysis Using R

Goal

Design marketing strategies aimed at converting casual riders into annual members.

Casual riders

Riders who purchased a single ride or full-day Passes.

Members

Riders who purchased an Annual Membership.

Stakeholders

The project stakeholder is **Lily Moreno**,the director of marketing and the manager of Cyclistic, a bike-share company in Chicago. She set the above goal and has a broader vision of the business. The analyst team has to communicate efficiently and frequently with *Lily* in all steps of the analysis process in order to achieve the goal.

I) Ask

Three questions will guide the future marketing program:

- How do annual members and casual riders use Cyclistic bikes differently?
- Why would casual riders buy Cyclistic annual memberships?
- How can Cyclistic use digital media to influence casual riders to become members?

II) Prepare

The public data is generated by Motivate International Inc under a license, this makes the data source reliable and original. Google Data Analytic provided this data through a link. Collected (downloaded) from Download the previous 12 months of Cyclistic trip data here.

- I have limit my analysis to historical data from Jan-2022 to Dec-2022.
- The data downloaded is stored in a CSV files.

- To get a sense of the data, I have open one file using Microsoft Excel and found that,it contains information on each ride's id, customers' type (casual or member), the start and end datetime of each trip, the start and end station names.
- Also found that, there is no private columns available, the entire data can be viewed by the whole team, no need to hide or give some special access to anyone.
- There are some columns with some missing values, in the "process step" I will tackle this issue.

III) Process

Now for data exploration, data cleaning & data shaping I have use the 'tidyverse' package of R.
Also I have document all the steps involve in the data processing.

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

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.2     v readr     2.1.4
## v forcats   1.0.0     v stringr   1.5.0
## v ggplot2   3.4.2     v tibble    3.2.1
## v lubridate 1.9.2     v tidyr     1.3.0
## v purrr     1.0.1
## -- 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 errors
```

1) Import the data

I have 12 CSV files of the same structure (column names and number of columns), stored in my local drive.

```
# Import each file and save it in the separate data frame.

data_01 <- read.csv("202201-divvy-tripdata.csv")
data_02 <- read.csv("202202-divvy-tripdata.csv")
data_03 <- read.csv("202203-divvy-tripdata.csv")
data_04 <- read.csv("202204-divvy-tripdata.csv")
data_05 <- read.csv("202205-divvy-tripdata.csv")
data_06 <- read.csv("202206-divvy-tripdata.csv")
data_07 <- read.csv("202207-divvy-tripdata.csv")
data_08 <- read.csv("202208-divvy-tripdata.csv")
data_09 <- read.csv("202209-divvy-tripdata.csv")
data_10 <- read.csv("202210-divvy-tripdata.csv")
data_11 <- read.csv("202211-divvy-tripdata.csv")
data_12 <- read.csv("202212-divvy-tripdata.csv")
```

Confirmed the structure of imported data by looking at the 1st row of data frame "data_01" as well as "data_12".

```
head(data_01,1)
```

```
##            ride_id rideable_type          started_at            ended_at
## 1 C2F7DD78E82EC875 electric_bike 2022-01-13 11:59:47 2022-01-13 12:02:44
##         start_station_name start_station_id     end_station_name end_station_id
## 1 Glenwood Ave & Touhy Ave              525 Clark St & Touhy Ave         RP-007
##   start_lat start_lng  end_lat   end_lng member_casual
## 1   42.0128 -87.66591 42.01256 -87.67437        casual
```

```
head(data_12,1)
```

```
##            ride_id rideable_type          started_at            ended_at
## 1 65DBD2F447EC51C2 electric_bike 2022-12-05 10:47:18 2022-12-05 10:56:34
##           start_station_name start_station_id          end_station_name
## 1 Clifton Ave & Armitage Ave     TA1307000163 Sedgwick St & Webster Ave
##   end_station_id start_lat start_lng  end_lat   end_lng member_casual
## 1          13191  41.91824 -87.65711 41.92217 -87.63889        member
```

- The results of theses queries: data_01 and data_12, give the same number of columns, the same type and the same name.
- Hence, I can confidently concatenate these data frames.

2) Concatenate the data frames

```
data_2022 <-  rbind(data_01,data_02,data_03,data_04,data_05,data_06,data_07,data_08,data_09,data_10,data
```

```
# To get the structure of the concatenated data set.

str(data_2022)
```

```
## 'data.frame':    5667717 obs. of  13 variables:
##  $ ride_id           : chr  "C2F7DD78E82EC875" "A6CF8980A652D272" "BD0F91DFF741C66D" "CBB80ED4191054
##  $ rideable_type     : chr  "electric_bike" "electric_bike" "classic_bike" "classic_bike" ...
##  $ started_at        : chr  "2022-01-13 11:59:47" "2022-01-10 08:41:56" "2022-01-25 04:53:40" "2022-
##  $ ended_at          : chr  "2022-01-13 12:02:44" "2022-01-10 08:46:17" "2022-01-25 04:58:01" "2022-
##  $ start_station_name: chr  "Glenwood Ave & Touhy Ave" "Glenwood Ave & Touhy Ave" "Sheffield Ave & Fu
##  $ start_station_id  : chr  "525" "525" "TA1306000016" "KA1504000151" ...
##  $ end_station_name  : chr  "Clark St & Touhy Ave" "Clark St & Touhy Ave" "Greenview Ave & Fullerton
##  $ end_station_id    : chr  "RP-007" "RP-007" "TA1307000001" "TA1309000021" ...
##  $ start_lat         : num  42 42 41.9 42 41.9 ...
##  $ start_lng         : num  -87.7 -87.7 -87.7 -87.7 -87.6 ...
##  $ end_lat           : num  42 42 41.9 42 41.9 ...
##  $ end_lng           : num  -87.7 -87.7 -87.7 -87.7 -87.6 ...
##  $ member_casual     : chr  "casual" "casual" "member" "casual" ...
```

- We have a total of 13 columns and 5,667,717 records.

3) Understand the Data type of each column

- ride_id, start_station_name, start_station_id, end_station_name, end_station_id, member_casual and rideable_type are of type "chr", so a string, which is as expected.
- started_at and ended_at are of type "Chr". I was expecting a datetime type, because the columns are made up of a date and a time.
- start_lat, start_lng, end_lat and end_lng are of type "num", it matches the expectation.

4) Change the data type of "started_at" and "ended_at"

```
# make the copy of the data set
data_2022_v001 <- data_2022

# Added an duplicate columns
data_2022_v001$started_at_dup <- data_2022_v001$started_at
data_2022_v001$ended_at_dup <- data_2022_v001$ended_at

# Change the data types from 'Str' to 'POSIXct'(datetime)
data_2022_v001$started_at_dup <- as.POSIXct(data_2022_v001$started_at_dup)
data_2022_v001$ended_at_dup <- as.POSIXct(data_2022_v001$ended_at_dup)

#validate the structure of data set after data type conversion
str(data_2022_v001)
```

```
## 'data.frame':    5667717 obs. of  15 variables:
##  $ ride_id           : chr  "C2F7DD78E82EC875" "A6CF8980A652D272" "BD0F91DFF741C66D" "CBB80ED4191054
##  $ rideable_type     : chr  "electric_bike" "electric_bike" "classic_bike" "classic_bike" ...
##  $ started_at        : chr  "2022-01-13 11:59:47" "2022-01-10 08:41:56" "2022-01-25 04:53:40" "2022-
##  $ ended_at          : chr  "2022-01-13 12:02:44" "2022-01-10 08:46:17" "2022-01-25 04:58:01" "2022-
##  $ start_station_name: chr  "Glenwood Ave & Touhy Ave" "Glenwood Ave & Touhy Ave" "Sheffield Ave & Fu
##  $ start_station_id  : chr  "525" "525" "TA1306000016" "KA1504000151" ...
##  $ end_station_name  : chr  "Clark St & Touhy Ave" "Clark St & Touhy Ave" "Greenview Ave & Fullerton
##  $ end_station_id    : chr  "RP-007" "RP-007" "TA1307000001" "TA1309000021" ...
##  $ start_lat         : num  42 42 41.9 42 41.9 ...
##  $ start_lng         : num  -87.7 -87.7 -87.7 -87.7 -87.6 ...
##  $ end_lat           : num  42 42 41.9 42 41.9 ...
##  $ end_lng           : num  -87.7 -87.7 -87.7 -87.7 -87.6 ...
##  $ member_casual     : chr  "casual" "casual" "member" "casual" ...
##  $ started_at_dup    : POSIXct, format: "2022-01-13 11:59:47" "2022-01-10 08:41:56" ...
##  $ ended_at_dup      : POSIXct, format: "2022-01-13 12:02:44" "2022-01-10 08:46:17" ...
```

```
# Select the 1st value from started_at
data_2022_v001$started_at_dup[1]
```

```
## [1] "2022-01-13 11:59:47 IST"
```

```
# Select the 1st value from ended_at
data_2022_v001$ended_at_dup[1]
```

```
## [1] "2022-01-13 12:02:44 IST"
```

```r
# get the ride_time
data_2022_v001$ended_at_dup[1]-data_2022_v001$started_at_dup[1]
```

```
## Time difference of 2.95 mins
```

- By subtracting the 1st value of "started_at" from "ended_at", I am getting the value of "ride_time" in the time format, which is as per my requirement.

5) Shape the dataframe

A ride time start can't be greater than a ride time end, nor equal. Let us verify the latter.

```r
sum(data_2022_v001$started_at_dup>=data_2022_v001$ended_at_dup)
```

```
## [1] 531
```

- We have 531 records that confirms the above.
- It means that the date time they started the ride is > date time ended the ride, this can't be possible.
- It might be an error during "Data collection".

```r
# get the Subset of the data frame after excluding the values where "started_at_dup" >= "endend_at_dup"

data_2022_v002 <- data_2022_v001[data_2022_v001$started_at_dup<data_2022_v001$ended_at_dup,]

# validate the results by re-checking the count of rows

str(data_2022_v002)
```

```
## 'data.frame':    5667186 obs. of  15 variables:
##  $ ride_id           : chr  "C2F7DD78E82EC875" "A6CF8980A652D272" "BD0F91DFF741C66D" "CBB80ED4191054
##  $ rideable_type     : chr  "electric_bike" "electric_bike" "classic_bike" "classic_bike" ...
##  $ started_at        : chr  "2022-01-13 11:59:47" "2022-01-10 08:41:56" "2022-01-25 04:53:40" "2022-
##  $ ended_at          : chr  "2022-01-13 12:02:44" "2022-01-10 08:46:17" "2022-01-25 04:58:01" "2022-
##  $ start_station_name: chr  "Glenwood Ave & Touhy Ave" "Glenwood Ave & Touhy Ave" "Sheffield Ave & Fu
##  $ start_station_id  : chr  "525" "525" "TA1306000016" "KA1504000151" ...
##  $ end_station_name  : chr  "Clark St & Touhy Ave" "Clark St & Touhy Ave" "Greenview Ave & Fullerton
##  $ end_station_id    : chr  "RP-007" "RP-007" "TA1307000001" "TA1309000021" ...
##  $ start_lat         : num  42 42 41.9 42 41.9 ...
##  $ start_lng         : num  -87.7 -87.7 -87.7 -87.7 -87.6 ...
##  $ end_lat           : num  42 42 41.9 42 41.9 ...
##  $ end_lng           : num  -87.7 -87.7 -87.7 -87.7 -87.6 ...
##  $ member_casual     : chr  "casual" "casual" "member" "casual" ...
##  $ started_at_dup    : POSIXct, format: "2022-01-13 11:59:47" "2022-01-10 08:41:56" ...
##  $ ended_at_dup      : POSIXct, format: "2022-01-13 12:02:44" "2022-01-10 08:46:17" ...
```

- We have total 5,667,186 rows which means 531 rows are successfully excluded.

```r
# calculate the ride time

data_2022_v002$ride_time <- ((data_2022_v002$ended_at_dup)-(data_2022_v002$started_at_dup))/60

str(data_2022_v002)
```

```
## 'data.frame':    5667186 obs. of  16 variables:
##  $ ride_id           : chr  "C2F7DD78E82EC875" "A6CF8980A652D272" "BD0F91DFF741C66D" "CBB80ED4191054
##  $ rideable_type     : chr  "electric_bike" "electric_bike" "classic_bike" "classic_bike" ...
##  $ started_at        : chr  "2022-01-13 11:59:47" "2022-01-10 08:41:56" "2022-01-25 04:53:40" "2022-
##  $ ended_at          : chr  "2022-01-13 12:02:44" "2022-01-10 08:46:17" "2022-01-25 04:58:01" "2022-
##  $ start_station_name: chr  "Glenwood Ave & Touhy Ave" "Glenwood Ave & Touhy Ave" "Sheffield Ave & Fu
##  $ start_station_id  : chr  "525" "525" "TA1306000016" "KA1504000151" ...
##  $ end_station_name  : chr  "Clark St & Touhy Ave" "Clark St & Touhy Ave" "Greenview Ave & Fullerton
##  $ end_station_id    : chr  "RP-007" "RP-007" "TA1307000001" "TA1309000021" ...
##  $ start_lat         : num  42 42 41.9 42 41.9 ...
##  $ start_lng         : num  -87.7 -87.7 -87.7 -87.7 -87.6 ...
##  $ end_lat           : num  42 42 41.9 42 41.9 ...
##  $ end_lng           : num  -87.7 -87.7 -87.7 -87.7 -87.6 ...
##  $ member_casual     : chr  "casual" "casual" "member" "casual" ...
##  $ started_at_dup    : POSIXct, format: "2022-01-13 11:59:47" "2022-01-10 08:41:56" ...
##  $ ended_at_dup      : POSIXct, format: "2022-01-13 12:02:44" "2022-01-10 08:46:17" ...
##  $ ride_time         : 'difftime' num  2.95 4.35 4.35 14.9333333333333 ...
##   ..- attr(*, "units")= chr "secs"
```

- The resulted ride_time is given in the 'Seconds' so we have converted it into 'Minutes' by dividing it by 60.

6) Check for the Null Values

```r
colSums(is.na(data_2022_v002))
```

```
##            ride_id      rideable_type         started_at           ended_at
##                  0                  0                  0                  0
## start_station_name   start_station_id   end_station_name     end_station_id
##                  0                  0                  0                  0
##          start_lat          start_lng            end_lat            end_lng
##                  0                  0               5858               5858
##      member_casual     started_at_dup       ended_at_dup          ride_time
##                  0                  0                  0                  0
```

- The quantitative column name "end_lat" and "end_lng" have null values.
- No Null values found in started_at, ended_at, member_casual and ride_d, ride_time. This is good because it helps to examine each ride.

7) Check for misspellings

- For the qualitative columns "rideable_type" and "member_casual" I have check for the possible typo errors.

- Example:
  + casual, casualll, caslaul
  + Electric bike, electrik_bike, clasic_bike

```
# To know the unique values from 'rideable_type'
unique(data_2022_v002$rideable_type)
```

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

```
# To know the unique values from 'member_casual'
unique(data_2022_v002$member_casual)
```

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

- No misspellings found

8) Check for duplicates

- Each ride is unique & I hope to have unique records (rows).
- The dataset, has the ride_id column which describes each ride taken by a casual or an annual member.

```
sum(duplicated(data_2022_v002$ride_id))
```

```
## [1] 0
```

- Zero value returned, hence there are no duplicates for the column "ride_id". This confirms that each record of the dataframe is unique.

9) Add a new Columns "Weekday" & "Month"

- I do this to have a data in shape for the **Analyse** step of data analysis process.

```
# add the name of the day from started_at
data_2022_v002$weekday <- wday(data_2022_v002$started_at_dup,week_start=1,label = TRUE)

# add the name of the month from started_at
data_2022_v002$month <- month(data_2022_v002$started_at_dup,label = TRUE)

str(data_2022_v002)
```

```
## 'data.frame':    5667186 obs. of  18 variables:
##  $ ride_id           : chr  "C2F7DD78E82EC875" "A6CF8980A652D272" "BD0F91DFF741C66D" "CBB80ED4191054
##  $ rideable_type     : chr  "electric_bike" "electric_bike" "classic_bike" "classic_bike" ...
##  $ started_at        : chr  "2022-01-13 11:59:47" "2022-01-10 08:41:56" "2022-01-25 04:53:40" "2022-
##  $ ended_at          : chr  "2022-01-13 12:02:44" "2022-01-10 08:46:17" "2022-01-25 04:58:01" "2022-
##  $ start_station_name: chr  "Glenwood Ave & Touhy Ave" "Glenwood Ave & Touhy Ave" "Sheffield Ave & Fu
##  $ start_station_id  : chr  "525" "525" "TA1306000016" "KA1504000151" ...
```

```
##  $ end_station_name  : chr  "Clark St & Touhy Ave" "Clark St & Touhy Ave" "Greenview Ave & Fullerton
##  $ end_station_id    : chr  "RP-007" "RP-007" "TA1307000001" "TA1309000021" ...
##  $ start_lat         : num  42 42 41.9 42 41.9 ...
##  $ start_lng         : num  -87.7 -87.7 -87.7 -87.7 -87.6 ...
##  $ end_lat           : num  42 42 41.9 42 41.9 ...
##  $ end_lng           : num  -87.7 -87.7 -87.7 -87.7 -87.6 ...
##  $ member_casual     : chr  "casual" "casual" "member" "casual" ...
##  $ started_at_dup    : POSIXct, format: "2022-01-13 11:59:47" "2022-01-10 08:41:56" ...
##  $ ended_at_dup      : POSIXct, format: "2022-01-13 12:02:44" "2022-01-10 08:46:17" ...
##  $ ride_time         : 'difftime' num  2.95 4.35 4.35 14.9333333333333 ...
##   ..- attr(*, "units")= chr "secs"
##  $ weekday           : Ord.factor w/ 7 levels "Mon"<"Tue"<"Wed"<..: 4 1 2 2 4 2 7 6 1 5 ...
##  $ month             : Ord.factor w/ 12 levels "Jan"<"Feb"<"Mar"<..: 1 1 1 1 1 1 1 1 1 1 ...
```

IV) Analyse

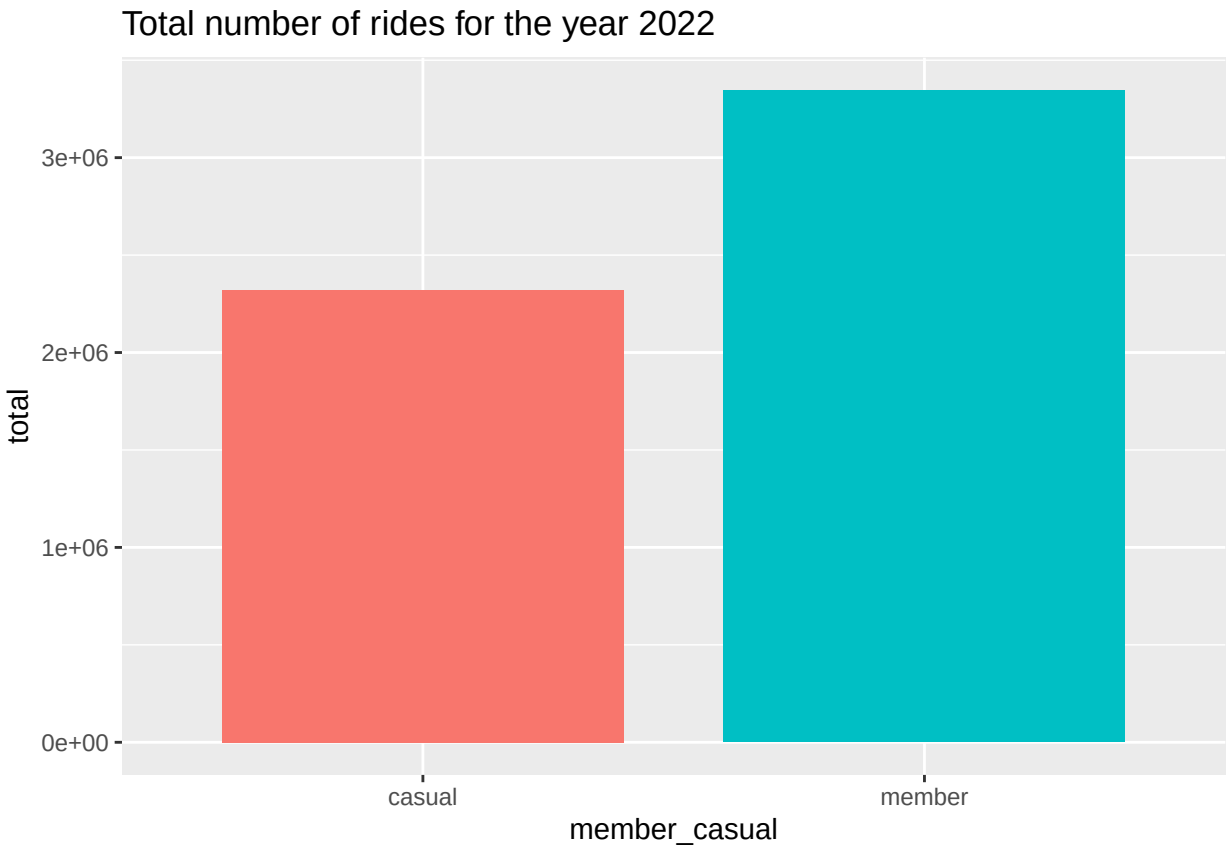
- In this step of the data analysis process, I will do calculations, data shaping to sketch visuals for the Share step.

1) Number of yearly rides for each type of customer

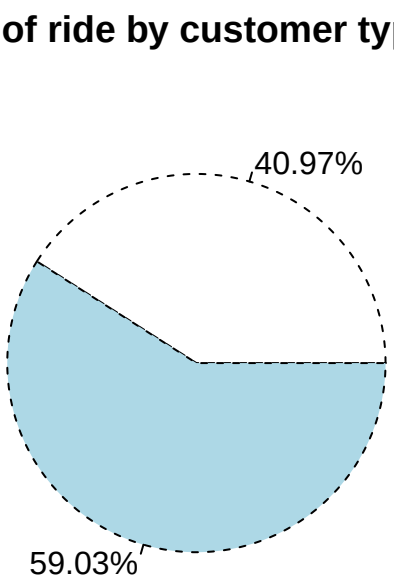
```
# Use pipes '%>%' to drill down the data
library(tidyverse)
data_2022_v002 %>%
  group_by(member_casual) %>%
  summarise(count=n()) %>%
  rename("total"="count")
```

```
## # A tibble: 2 x 2
##   member_casual   total
##   <chr>           <int>
## 1 casual        2321769
## 2 member        3345417
```

```
# Plot an bar_chart to compare the number of rides visually

library(ggplot2)
ggplot(data = data_2022_v002)+
  geom_bar(mapping = aes(x=member_casual,fill=member_casual),show.legend = FALSE,width = 0.8)+
  labs(y="total",title = "Total number of rides for the year 2022")
```


Total number of rides for the year 2022
3e+06
2e+06
1e+06
0e+00
total
casual
member
member_casual

```r
#Create a data frame of member_casual_count_summary
member_casual_summary<- data_2022_v002 %>%
                            group_by(member_casual) %>%
                            summarise(count=n())

#Create a labels for pie chart
pie_labels <- paste0(round(100*member_casual_summary$count/sum(member_casual_summary$count),2),"%")

pie(member_casual_summary$count,labels = pie_labels,lty = 2, main = "% of ride by customer type")
```

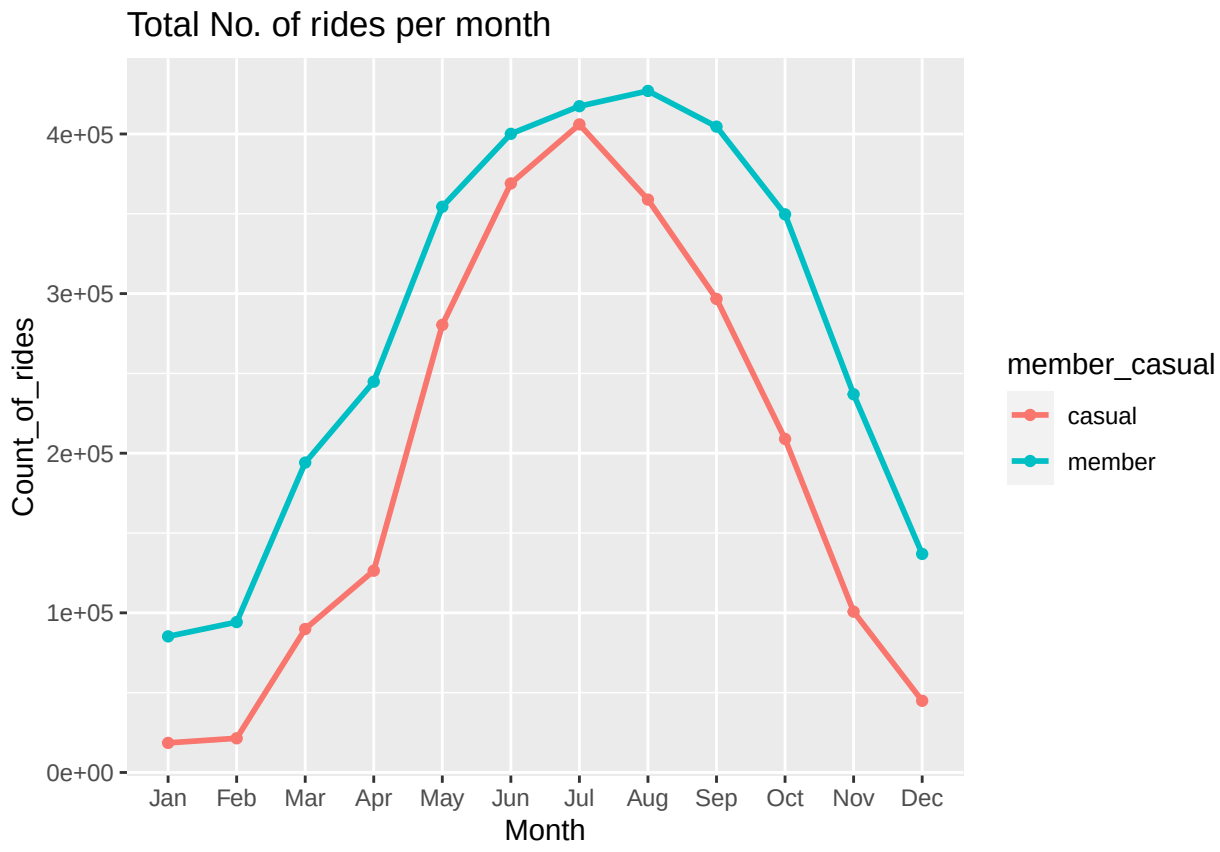


2) Total rides taken per month for each type of customers

```
# Select the required columns & reshape the data
monthly_ride_count <- data_2022_v002 %>%
                      group_by(month,member_casual) %>%
                      summarise(count_of_ride =n())
```

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

```
# Plot the line chart
ggplot(monthly_ride_count,aes(x=month,y=count_of_ride,group=member_casual,xlim(0,400000)))+
  geom_point(aes(color=member_casual),size=1.5)+
  geom_line(aes(color=member_casual),size=1)+
  labs(x="Month",y="Count_of_rides",title = "Total No. of rides per month")
```



Monthly ride count difference between member and casual riders

```
# Created an subset of data frame
monthly_ride_count_2 <- monthly_ride_count %>%
                        pivot_wider(names_from = member_casual,values_from = count_of_ride) %>% #use pi
                        mutate(ride_count_diff = member-casual)
print(monthly_ride_count_2)
```

```
## # A tibble: 12 x 4
## # Groups:   month [12]
##    month casual member ride_count_diff
##    <ord>  <int>  <int>           <int>
##  1 Jan    18517  85248           66731
##  2 Feb    21414  94190           72776
##  3 Mar    89874 194150          104276
##  4 Apr   126398 244820          118422
##  5 May   280387 354423           74036
##  6 Jun   369022 400116           31094
##  7 Jul   406013 417403           11390
##  8 Aug   358886 426969           68083
##  9 Sep   296664 404603          107939
## 10 Oct   208961 349659          140698
## 11 Nov   100742 236935          136193
## 12 Dec    44891 136901           92010
```

Average of monthly ride count difference between member and casual riders

```
mean(monthly_ride_count_2$ride_count_diff)
```

```
## [1] 85304
```

3) Total rides taken each day in a week for each type of customers

```
day_wise_ride_count <- data_2022_v002 %>%
                       group_by(weekday,member_casual) %>%
                       summarise(count_of_ride =n()) %>%
                       pivot_wider(names_from = member_casual,values_from = count_of_ride) %>%
                       mutate(total_rides = casual+member) %>%
                       mutate(casual_percentage= (casual/total_rides)*100) %>%
                       mutate(member_percentage= (member/total_rides)*100)
```

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

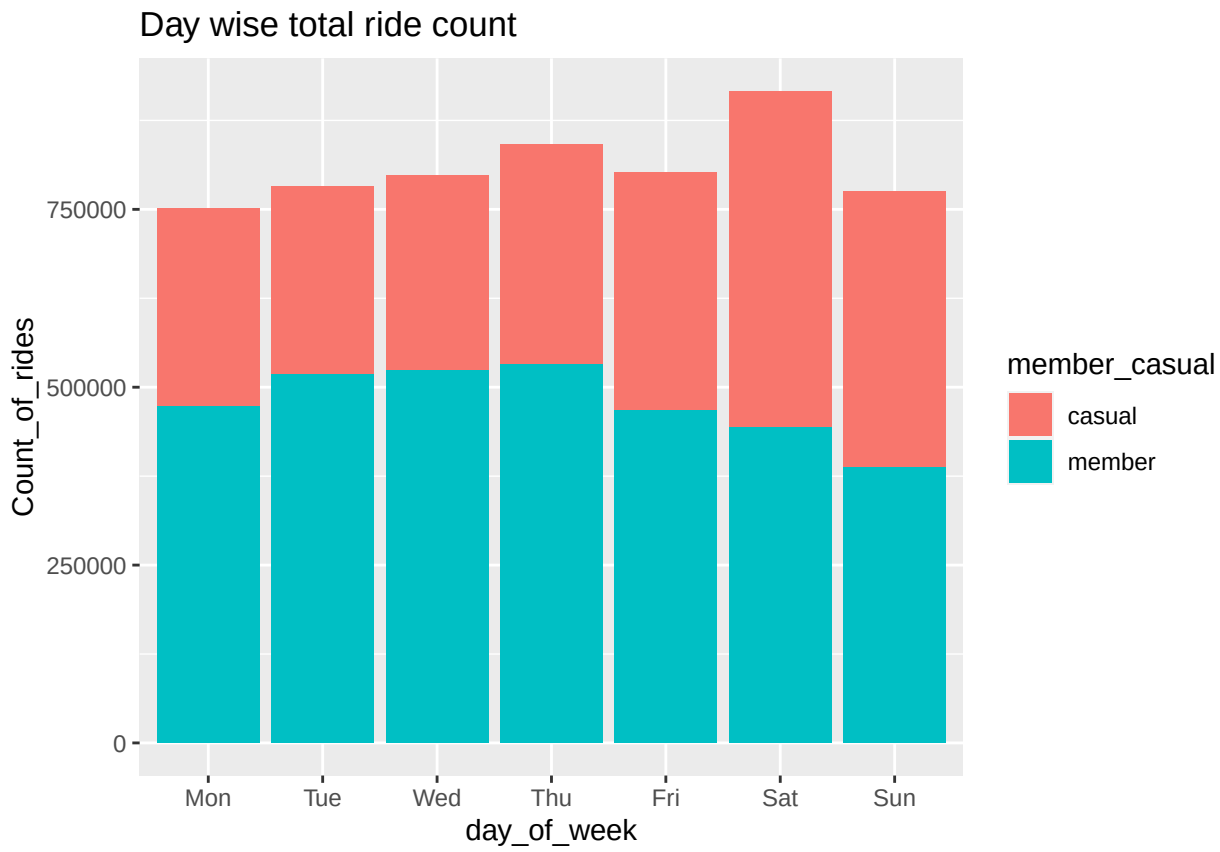
```
print(day_wise_ride_count)
```

```
## # A tibble: 7 x 6
## # Groups:   weekday [7]
##   weekday casual member total_rides casual_percentage member_percentage
##   <ord>    <int>  <int>       <int>              <dbl>              <dbl>
## 1 Mon     277649 473305      750954               37.0               63.0
## 2 Tue     263706 518584      782290               33.7               66.3
## 3 Wed     274339 523836      798175               34.4               65.6
## 4 Thu     309297 532215      841512               36.8               63.2
## 5 Fri     334667 467051      801718               41.7               58.3
## 6 Sat     473130 443246      916376               51.6               48.4
## 7 Sun     388981 387180      776161               50.1               49.9
```

```r
day_wise_ride_count_2 <- data_2022_v002 %>%
                      group_by(weekday,member_casual) %>%
                      summarise(count_of_ride =n())
```

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

```r
ggplot(data = day_wise_ride_count_2)+
  geom_col(mapping=aes(x=weekday,y=count_of_ride,fill=member_casual))+
  labs(x="day_of_week",y="Count_of_rides",title = "Day wise total ride count")
```

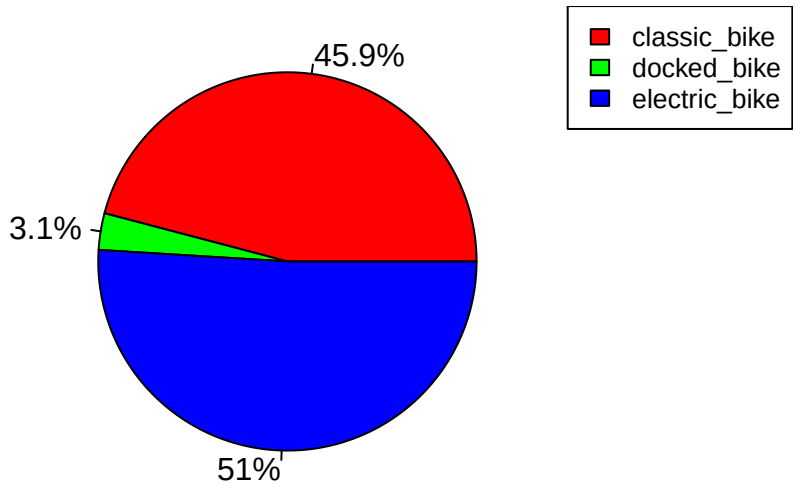


4) Types of bikes per type of customers

```
bike_type_count <- data_2022_v002 %>%
                   group_by(rideable_type) %>%
                   summarise(ride_count=n()) %>%
                   mutate(ride_count_percentage = round(100*ride_count/sum(ride_count),1))
print(bike_type_count)
```

```
## # A tibble: 3 x 3
##   rideable_type ride_count ride_count_percentage
##   <chr>              <int>                 <dbl>
## 1 classic_bike     2601088                  45.9
## 2 docked_bike       177468                   3.1
## 3 electric_bike    2888630                  51
```

```
pie(bike_type_count$ride_count,labels = paste0(bike_type_count$ride_count_percentage,"%"),col = rainbow
legend("topright",bike_type_count$rideable_type,cex = 0.8,fill = rainbow(length(bike_type_count$ride_co
```



```r
bike_type_count_2 <- data_2022_v002 %>%
                  group_by(rideable_type,member_casual) %>%
                  summarise(ride_count=n()) %>%
                  pivot_wider(names_from = member_casual,values_from = ride_count) %>%
                  mutate(total_ride_count = sum(casual,member,na.rm=TRUE)) %>%
                  mutate(casual_percentage = round(100*(casual/total_ride_count),2)) %>%
                  mutate(member_percentage = round(100*(member/total_ride_count),2))
```

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

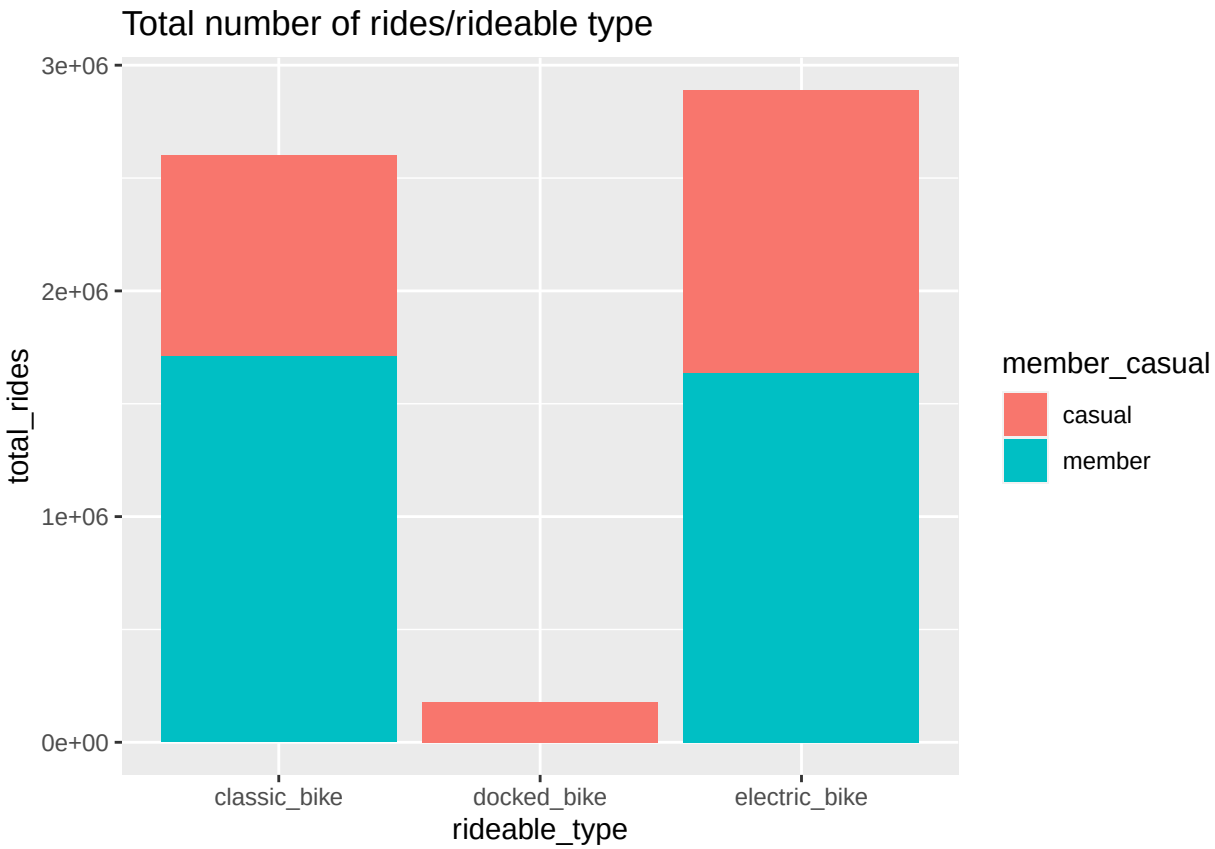
```r
print(bike_type_count_2)
```

```
## # A tibble: 3 x 6
## # Groups:   rideable_type [3]
##   rideable_type  casual  member total_ride_count casual_percentage
##   <chr>           <int>   <int>            <int>             <dbl>
## 1 classic_bike   891406 1709682          2601088              34.3
## 2 docked_bike    177468      NA           177468             100
## 3 electric_bike 1252895 1635735          2888630              43.4
## # i 1 more variable: member_percentage <dbl>
```

```r
bike_type_count_3 <- data_2022_v002 %>%
                  group_by(rideable_type,member_casual) %>%
                  summarise(ride_count=n())
```

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

```r
ggplot(data = bike_type_count_3)+
  geom_col(mapping=aes(x=rideable_type,y=ride_count,fill=member_casual))+
  labs(x="rideable_type",y="total_rides",title = "Total number of rides/rideable type")
```


Total number of rides/rideable type
3e+06
2e+06
1e+06
0e+00
total_rides
classic_bike
docked_bike
electric_bike
rideable_type
member_casual
casual
member

V) Share & Act

- I will skip the following steps **Share** and **Act** because, the analysis is a personal project.
- I will go directly to findings and recommendations.

VI) Findings and Recommendations

1) How do annual members and casual riders use Cyclistic bikes differently?

- For the year **2022**, which is our study time, we have more rides for annual members (**59%** rides) than casual riders (**41%**).
- When we go down to a finer level, at a month, we observe the following between the casual & annual members:
  - The difference in ride is of an average of **85,304** rides
  - For the months of **June** and **July** the total number of rides are fairly close as compared to the other months.

- At the week day level of granularity, for the days: **Saturday** and **Sunday**, casual riders have greater rides than annual members.
- Docked bikes are only used by casual riders, it represents **3%** of rides (electric 51% and classic 46%).
- 34.3% of rides by classic bikes are casual riders and 65.7% are annual members.
- 43.4% of rides by electric bikes are casual riders and 56.6% are annual members.

2) Why would casual riders buy Cyclistic annual memberships?

- We previously saw that casual riders ride more on weekends, if they have to ride at the same pace during the weekdays, it may motivate them to become annual members.
- If their preference for docked bikes shift to classic or electric bikes, Cyclistic can hope of having an increase in annual members.
- To understand more about the customer's choice of becoming annual members or casual riders and their ride time, the following information can help to do a finer analysis:
  - The reason behind each rides, example: home, work, leisure..
  - The cost details for rider type and bike type

3) How can Cyclistic use digital media to influence casual riders to become members?

Through influencer marketing, advertising and environmental awareness campaign on social media and TV,Cyclistic can work on the following:

- The advantages of using more electric bikes (environmentally friendly) than docked bikes (from our sample data, we do not have annual members for docked bikes only for electric & classic bikes).
- Encourage casual riders to ride throughout the week as they do during the weekend.
- The advantages in becoming an annual member, example: it can be less costly when we compare the average price (in a year, month, week, day) for each ride as an annual member as compared to a casual rider.