Cyclistic Biking Service Case Study

B M Morshed Sayeed

2022-05-06

Introduction

My name is B M Morshed Sayeed and I'm currently enrolled in an university program to get my Bachelor degree in Computer Science and Technology. During my undergraduate program, there are some courses focused on data analytics and statistics. These courses are most interesting and encouraging.

Now, I am about to finish Google Data Analytics Certificate. The only thing left to do is the capstone project. Here I've chosen R as the primary tool of my data analysis.

Background

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.

Until now, Cyclistic's marketing strategy relied on building general awareness and appealing to broad consumer segments. One approach that helped make these things possible was the flexibility of its pricing plans: single-ride passes, full-day passes, and annual memberships. Customers who purchase single-ride or full-day passes are referred to as casual riders. Customers who purchase annual memberships are Cyclistic members.

Business Problem

This report will examine the business question: 'what is the most effective marketing strategy to converting Cyclistic's casual riders to annul memberships?' Three questions will be guide the analysis program to achieve the business goal:

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

Primary Stakeholders

1. Cyclistic Executive Team
2. Lily Moreno, Director of Marketing and Manager

Data Sources

User data from the past 12 months, April 2021 - March 2022 has been made available. Each data set is in csv format and details every ride logged by Cyclistic customers. This data has been made publicly available via license by Motivate International Inc. and the city of Chicago available here. All user's personal data has been scrubbed for privacy.

Tools for analysis

R is being used due to the data size and visualizations needed to complete this analysis.

Preparation of data

Load packages

For basic data wrangling, manipulation and plotting we install the `tidyverse` package that itself contains a lot of useful packages. The `dplyr` and `janitor` package enables us to perform some common data manipulation task of cleaning and analysis. The `lubridate` package helps us to manipulate date-time format. And `ggplot2` package uses for data visualization task. You need to install them manually via:

- `install.packages("tidyverse")`
- `install.packages("dplyr")`
- `install.packages("janitor")`
- `install.packages("lubridate")`
- `install.packages("ggplot2")`

```r
#load the necessary libraries
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   1.0.8
## 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()
```

```r
library(dplyr)
library(janitor)
```

```
##
## Attaching package: 'janitor'
```

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

```r
library(lubridate)
```

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

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

```r
library(ggplot2)
```

Check and set current working directory

If you want to run this markdown file, it is important that you change the absolute file path to wherever you saved the data files.

Load all the data

```
t21_Apr <- read_csv("data/202104-divvy-tripdata.csv")
```

```
## Rows: 337230 Columns: 13
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (7): ride_id, rideable_type, start_station_name, start_station_id, end_...
## dbl  (4): start_lat, start_lng, end_lat, end_lng
## dttm (2): started_at, ended_at
##
## i Use ‘spec()‘ to retrieve the full column specification for this data.
## i Specify the column types or set ‘show_col_types = FALSE‘ to quiet this message.
```

```
t21_May <- read_csv("data/202105-divvy-tripdata.csv")
```

```
## Rows: 531633 Columns: 13
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (7): ride_id, rideable_type, start_station_name, start_station_id, end_...
## dbl  (4): start_lat, start_lng, end_lat, end_lng
## dttm (2): started_at, ended_at
##
## i Use ‘spec()‘ to retrieve the full column specification for this data.
## i Specify the column types or set ‘show_col_types = FALSE‘ to quiet this message.
```

```
t21_Jun <- read_csv("data/202106-divvy-tripdata.csv")
```

```
## Rows: 729595 Columns: 13
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (7): ride_id, rideable_type, start_station_name, start_station_id, end_...
## dbl  (4): start_lat, start_lng, end_lat, end_lng
## dttm (2): started_at, ended_at
##
## i Use ‘spec()‘ to retrieve the full column specification for this data.
## i Specify the column types or set ‘show_col_types = FALSE‘ to quiet this message.
```

```
t21_Jul <- read_csv("data/202107-divvy-tripdata.csv")
```

```
## Rows: 822410 Columns: 13
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (7): ride_id, rideable_type, start_station_name, start_station_id, end_...
## dbl  (4): start_lat, start_lng, end_lat, end_lng
## dttm (2): started_at, ended_at
##
## i Use ‘spec()‘ to retrieve the full column specification for this data.
## i Specify the column types or set ‘show_col_types = FALSE‘ to quiet this message.
```

```
t21_Aug <- read_csv("data/202108-divvy-tripdata.csv")
```

```
## Rows: 804352 Columns: 13
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (7): ride_id, rideable_type, start_station_name, start_station_id, end_...
## dbl  (4): start_lat, start_lng, end_lat, end_lng
## dttm (2): started_at, ended_at
##
## i Use ‘spec()‘ to retrieve the full column specification for this data.
## i Specify the column types or set ‘show_col_types = FALSE‘ to quiet this message.
```

```
t21_Sep <- read_csv("data/202109-divvy-tripdata.csv")
```

```
## Rows: 756147 Columns: 13
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (7): ride_id, rideable_type, start_station_name, start_station_id, end_...
## dbl  (4): start_lat, start_lng, end_lat, end_lng
## dttm (2): started_at, ended_at
##
## i Use ‘spec()‘ to retrieve the full column specification for this data.
## i Specify the column types or set ‘show_col_types = FALSE‘ to quiet this message.
```

```
t21_Oct <- read_csv("data/202110-divvy-tripdata.csv")
```

```
## Rows: 631226 Columns: 13
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (7): ride_id, rideable_type, start_station_name, start_station_id, end_...
## dbl  (4): start_lat, start_lng, end_lat, end_lng
## dttm (2): started_at, ended_at
##
## i Use ‘spec()‘ to retrieve the full column specification for this data.
## i Specify the column types or set ‘show_col_types = FALSE‘ to quiet this message.
```

```
t21_Nov <- read_csv("data/202111-divvy-tripdata.csv")
```

```
## Rows: 359978 Columns: 13
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (7): ride_id, rideable_type, start_station_name, start_station_id, end_...
## dbl  (4): start_lat, start_lng, end_lat, end_lng
## dttm (2): started_at, ended_at
##
## i Use ‘spec()‘ to retrieve the full column specification for this data.
## i Specify the column types or set ‘show_col_types = FALSE‘ to quiet this message.
```

```
t21_Dec <- read_csv("data/202112-divvy-tripdata.csv")
```

```
## Rows: 247540 Columns: 13
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (7): ride_id, rideable_type, start_station_name, start_station_id, end_...
## dbl  (4): start_lat, start_lng, end_lat, end_lng
## dttm (2): started_at, ended_at
##
## i Use ‘spec()‘ to retrieve the full column specification for this data.
## i Specify the column types or set ‘show_col_types = FALSE‘ to quiet this message.
```

```
t22_Jan <- read_csv("data/202201-divvy-tripdata.csv")
```

```
## Rows: 103770 Columns: 13
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (7): ride_id, rideable_type, start_station_name, start_station_id, end_...
## dbl  (4): start_lat, start_lng, end_lat, end_lng
## dttm (2): started_at, ended_at
##
## i Use ‘spec()‘ to retrieve the full column specification for this data.
## i Specify the column types or set ‘show_col_types = FALSE‘ to quiet this message.
```

```
t22_Fed <- read_csv("data/202202-divvy-tripdata.csv")
```

```
## Rows: 115609 Columns: 13
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (7): ride_id, rideable_type, start_station_name, start_station_id, end_...
## dbl  (4): start_lat, start_lng, end_lat, end_lng
## dttm (2): started_at, ended_at
##
## i Use ‘spec()‘ to retrieve the full column specification for this data.
## i Specify the column types or set ‘show_col_types = FALSE‘ to quiet this message.
```

```
t22_Mar <- read_csv("data/202203-divvy-tripdata.csv")
```

```
## Rows: 284042 Columns: 13
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (7): ride_id, rideable_type, start_station_name, start_station_id, end_...
## dbl  (4): start_lat, start_lng, end_lat, end_lng
## dttm (2): started_at, ended_at
##
## i Use ‘spec()‘ to retrieve the full column specification for this data.
## i Specify the column types or set ‘show_col_types = FALSE‘ to quiet this message.
```

Combine and inspect the dataset

```
trip_data <- rbind(t21_Apr, t21_May, t21_Jun, t21_Jul, t21_Aug, t21_Sep, t21_Oct,
                   t21_Nov, t21_Dec, t22_Jan, t22_Fed, t22_Mar)

glimpse(trip_data)
```

```
## Rows: 5,723,532
## Columns: 13
## $ ride_id            <chr> "6C992BD37A98A63F", "1E0145613A209000", "E498E15508~
## $ rideable_type      <chr> "classic_bike", "docked_bike", "docked_bike", "clas~
## $ started_at         <dttm> 2021-04-12 18:25:36, 2021-04-27 17:27:11, 2021-04--~
## $ ended_at           <dttm> 2021-04-12 18:56:55, 2021-04-27 18:31:29, 2021-04--~
## $ start_station_name <chr> "State St & Pearson St", "Dorchester Ave & 49th St"~
## $ start_station_id   <chr> "TA1307000061", "KA1503000069", "20121", "TA1305000~
## $ end_station_name   <chr> "Southport Ave & Waveland Ave", "Dorchester Ave & 4~
## $ end_station_id     <chr> "13235", "KA1503000069", "20121", "13235", "20121",~
## $ start_lat          <dbl> 41.89745, 41.80577, 41.74149, 41.90312, 41.74149, 4~
## $ start_lng          <dbl> -87.62872, -87.59246, -87.65841, -87.67394, -87.658~
## $ end_lat            <dbl> 41.94815, 41.80577, 41.74149, 41.94815, 41.74149, 4~
## $ end_lng            <dbl> -87.66394, -87.59246, -87.65841, -87.66394, -87.658~
## $ member_casual      <chr> "member", "casual", "casual", "member", "casual", "~
```

Data Cleaning

Firstly remove all the irrelevant columns that won't be used for analysis using pipe statement:

```
trip_data <- trip_data %>%
  select(-c(start_station_name, start_station_id, end_station_name, end_station_id,
            start_lat, start_lng, end_lat, end_lng))
```

Remove empty column and rows:

```
trip_data <- remove_empty(trip_data, which = c("rows", "cols"), quiet = FALSE)
```

```
## No empty rows to remove.
```

```
## No empty columns to remove.
```

Review of the data and its parameters:

```
head(trip_data)
```

```
## # A tibble: 6 x 5
##   ride_id    rideable_type started_at          ended_at            member_casual
##   <chr>      <chr>         <dttm>              <dttm>              <chr>
## 1 6C992BD37~ classic_bike  2021-04-12 18:25:36 2021-04-12 18:56:55 member
## 2 1E0145613~ docked_bike   2021-04-27 17:27:11 2021-04-27 18:31:29 casual
## 3 E498E1550~ docked_bike   2021-04-03 12:42:45 2021-04-07 11:40:24 casual
## 4 1887262AD~ classic_bike  2021-04-17 09:17:42 2021-04-17 09:42:48 member
## 5 C123548CA~ docked_bike   2021-04-03 12:42:25 2021-04-03 14:13:42 casual
## 6 097E76F36~ classic_bike  2021-04-25 18:43:18 2021-04-25 18:43:59 casual
```

Dimention of dataset:

```
dim(trip_data)
```

```
## [1] 5723532       5
```

Column names and row number of dataset:

```
colnames(trip_data)
```

```
## [1] "ride_id"       "rideable_type" "started_at"    "ended_at"
## [5] "member_casual"
```

```
nrow(trip_data)
```

```
## [1] 5723532
```

Structure of dataset:

```
str(trip_data)
```

```
## tibble [5,723,532 x 5] (S3: tbl_df/tbl/data.frame)
##  $ ride_id      : chr [1:5723532] "6C992BD37A98A63F" "1E0145613A209000" "E498E15508A80BAD" "1887262A
##  $ rideable_type: chr [1:5723532] "classic_bike" "docked_bike" "docked_bike" "classic_bike" ...
##  $ started_at   : POSIXct[1:5723532], format: "2021-04-12 18:25:36" "2021-04-27 17:27:11" ...
##  $ ended_at     : POSIXct[1:5723532], format: "2021-04-12 18:56:55" "2021-04-27 18:31:29" ...
##  $ member_casual: chr [1:5723532] "member" "casual" "casual" "member" ...
```

Summary of the dataset:

```
summary(trip_data)
```

```
##    ride_id          rideable_type        started_at
##  Length:5723532     Length:5723532     Min.   :2021-04-01 00:03:18
##  Class :character   Class :character   1st Qu.:2021-06-22 15:20:26
##  Mode  :character   Mode  :character   Median :2021-08-17 18:25:49
##                                        Mean   :2021-08-26 22:25:18
##                                        3rd Qu.:2021-10-14 19:48:10
##                                        Max.   :2022-03-31 23:59:47
##     ended_at                      member_casual
##  Min.   :2021-04-01 00:14:29   Length:5723532
##  1st Qu.:2021-06-22 15:47:37   Class :character
##  Median :2021-08-17 18:44:32   Mode  :character
##  Mean   :2021-08-26 22:46:50
##  3rd Qu.:2021-10-14 20:03:28
##  Max.   :2022-04-01 22:10:12
```

Convert date time to standard form:

```
trip_data$started_at <- ymd_hms(trip_data$started_at)
trip_data$ended_at <- ymd_hms(trip_data$ended_at)
```

Create additional columns for date and time

```
trip_data$day_of_week <- format(as.Date(trip_data$started_at), "%A")
trip_data$month <- format(as.Date(trip_data$started_at), "%m")
```

Calculated filed that shows the time of each unique ride in hour:

```
trip_data$ride_length <- (as.double(difftime(trip_data$ended_at, trip_data$started_at))) / 60
```

Remove the data where riding time is negative

```
trip_data <- trip_data[!(trip_data$ride_length <0),]
```

Now, check the updated dataset:

```
head(trip_data)
```

```
## # A tibble: 6 x 8
##   ride_id    rideable_type started_at          ended_at            member_casual
##   <chr>      <chr>         <dttm>              <dttm>              <chr>
## 1 6C992BD37~ classic_bike  2021-04-12 18:25:36 2021-04-12 18:56:55 member
## 2 1E0145613~ docked_bike   2021-04-27 17:27:11 2021-04-27 18:31:29 casual
## 3 E498E1550~ docked_bike   2021-04-03 12:42:45 2021-04-07 11:40:24 casual
## 4 1887262AD~ classic_bike  2021-04-17 09:17:42 2021-04-17 09:42:48 member
## 5 C123548CA~ docked_bike   2021-04-03 12:42:25 2021-04-03 14:13:42 casual
## 6 097E76F36~ classic_bike  2021-04-25 18:43:18 2021-04-25 18:43:59 casual
## # ... with 3 more variables: day_of_week <chr>, month <chr>, ride_length <dbl>
```

Structure of updated dataset:

```
str(trip_data)
```

```
## tibble [5,723,387 x 8] (S3: tbl_df/tbl/data.frame)
##  $ ride_id      : chr [1:5723387] "6C992BD37A98A63F" "1E0145613A209000" "E498E15508A80BAD" "1887262A
##  $ rideable_type: chr [1:5723387] "classic_bike" "docked_bike" "docked_bike" "classic_bike" ...
##  $ started_at   : POSIXct[1:5723387], format: "2021-04-12 18:25:36" "2021-04-27 17:27:11" ...
##  $ ended_at     : POSIXct[1:5723387], format: "2021-04-12 18:56:55" "2021-04-27 18:31:29" ...
##  $ member_casual: chr [1:5723387] "member" "casual" "casual" "member" ...
##  $ day_of_week  : chr [1:5723387] "Monday" "Tuesday" "Saturday" "Saturday" ...
##  $ month        : chr [1:5723387] "04" "04" "04" "04" ...
##  $ ride_length  : num [1:5723387] 31.3 64.3 5697.6 25.1 91.3 ...
```

Summary of the updated dataset:

```
summary(trip_data)
```

```
##    ride_id          rideable_type        started_at
##  Length:5723387     Length:5723387     Min.   :2021-04-01 00:03:18
##  Class :character   Class :character   1st Qu.:2021-06-22 15:19:19
##  Mode  :character   Mode  :character   Median :2021-08-17 18:25:00
##                                        Mean   :2021-08-26 22:24:29
##                                        3rd Qu.:2021-10-14 19:47:23
##                                        Max.   :2022-03-31 23:59:47
##     ended_at                     member_casual      day_of_week
##  Min.   :2021-04-01 00:14:29   Length:5723387     Length:5723387
##  1st Qu.:2021-06-22 15:46:31   Class :character   Class :character
##  Median :2021-08-17 18:43:54   Mode  :character   Mode  :character
##  Mean   :2021-08-26 22:46:01
##  3rd Qu.:2021-10-14 20:02:47
##  Max.   :2022-04-01 22:10:12
##     month             ride_length
##  Length:5723387     Min.   :    0.00
##  Class :character   1st Qu.:    6.57
##  Mode  :character   Median :   11.72
##                     Mean   :   21.54
##                     3rd Qu.:   21.33
##                     Max.   :55944.15
```

Analyze Data

Calculating the mean, max, min - figures to determine statistical value of membership type:

```
setNames(aggregate(trip_data$ride_length ~ trip_data$member_casual, FUN = mean),
         c("Member/Casual", "Avg_ride_time"))
```

```
##   Member/Casual Avg_ride_time
## 1        casual      31.74045
## 2        member      13.36978
```

```
setNames(aggregate(trip_data$ride_length ~ trip_data$member_casual, FUN = min),
         c("Member/Casual", "Min_ride_time"))
```

```
##   Member/Casual Min_ride_time
## 1        casual             0
## 2        member             0
```

```
setNames(aggregate(trip_data$ride_length ~ trip_data$member_casual, FUN = max),
         c("Member/Casual", "Max_ride_length"))
```

```
##   Member/Casual Max_ride_length
## 1        casual        55944.15
## 2        member         1559.90
```

Calculate the number of rides in each day for casual and annual members:

```
trip_data %>%
  mutate(days_week = wday(started_at, label = TRUE)) %>%   # label the days name.
  group_by(member_casual, days_week) %>%
  summarise(num_of_rides = n())  # Count the number of rides.
```

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

```
## # A tibble: 14 x 3
## # Groups:   member_casual [2]
##    member_casual days_week num_of_rides
##    <chr>         <ord>            <int>
##  1 casual        Sun             482801
##  2 casual        Mon             292993
##  3 casual        Tue             276371
##  4 casual        Wed             286400
##  5 casual        Thu             293632
##  6 casual        Fri             364277
##  7 casual        Sat             550008
##  8 member        Sun             387717
##  9 member        Mon             439428
## 10 member        Tue             490095
## 11 member        Wed             499901
## 12 member        Thu             475330
## 13 member        Fri             453108
## 14 member        Sat             431326
```

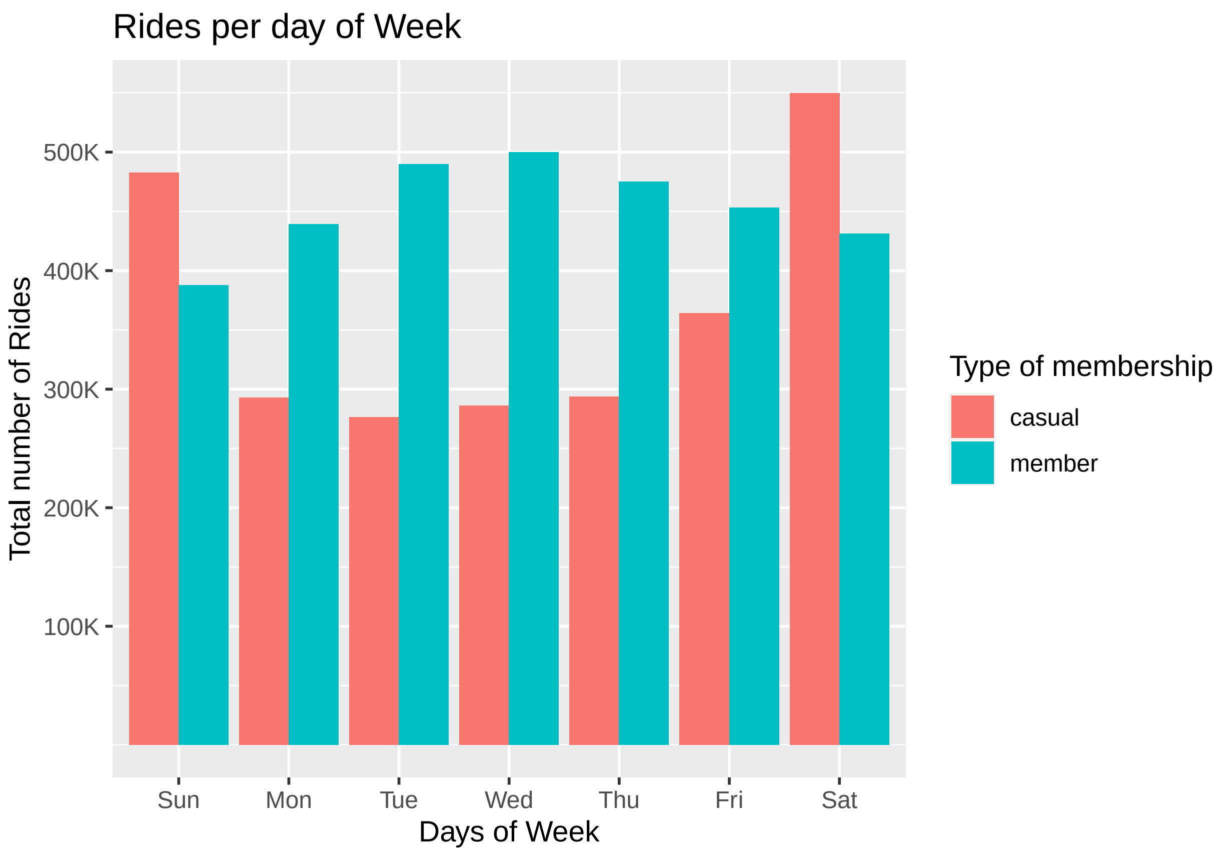
Data Visualization

We can visualize our data using ggplot2 package. Here, we are using column graph and bar graph to understand the relationship between variables.

Total rides broken down by weekday

```
trip_data %>%
  mutate(days_week = wday(started_at, label = TRUE)) %>%
  group_by(member_casual, days_week) %>%
  summarise(num_of_rides = n()) %>%
  ggplot(aes(x=days_week, y=num_of_rides, fill = member_casual)) +
  geom_col(position = "dodge") +
  labs(x="Days of Week", y="Total number of Rides", title = "Rides per day of Week",
       fill='Type of membership') +
  scale_y_continuous(breaks = c(100000, 200000, 300000, 400000, 500000, 600000),
                     labels = c("100K", "200K", "300K", "400K", "500K", "600K"))
```

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

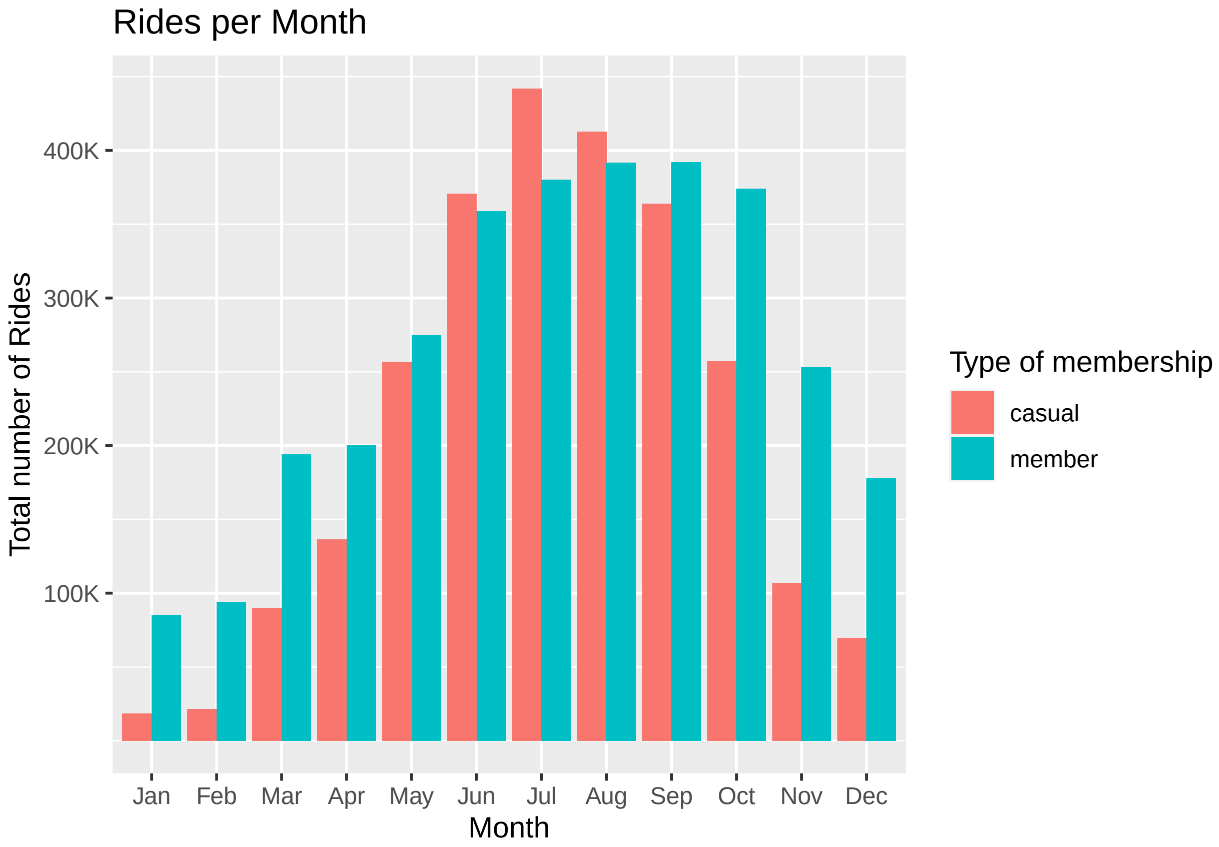


The rides per day of week shows casual riders peak on the Saturday and Sunday while members peak Monday through Friday. This indicates that members mainly use the bikes for their commutes and casual riders use the bikes main at their leisure time.

Total rides broken down by month

```
trip_data %>%
  mutate(month = month(started_at, label = TRUE)) %>%
  group_by(member_casual, month) %>%
  summarise(num_of_rides = n()) %>%
  ggplot(aes(x=month, y=num_of_rides, fill = member_casual)) +
  geom_col(position = "dodge") +
  labs(x="Month", y="Total number of Rides", title = "Rides per Month",
       fill='Type of membership') +
  scale_y_continuous(breaks = c(100000, 200000, 300000, 400000, 500000, 600000),
                     labels = c("100K", "200K", "300K", "400K", "500K", "600K"))
```

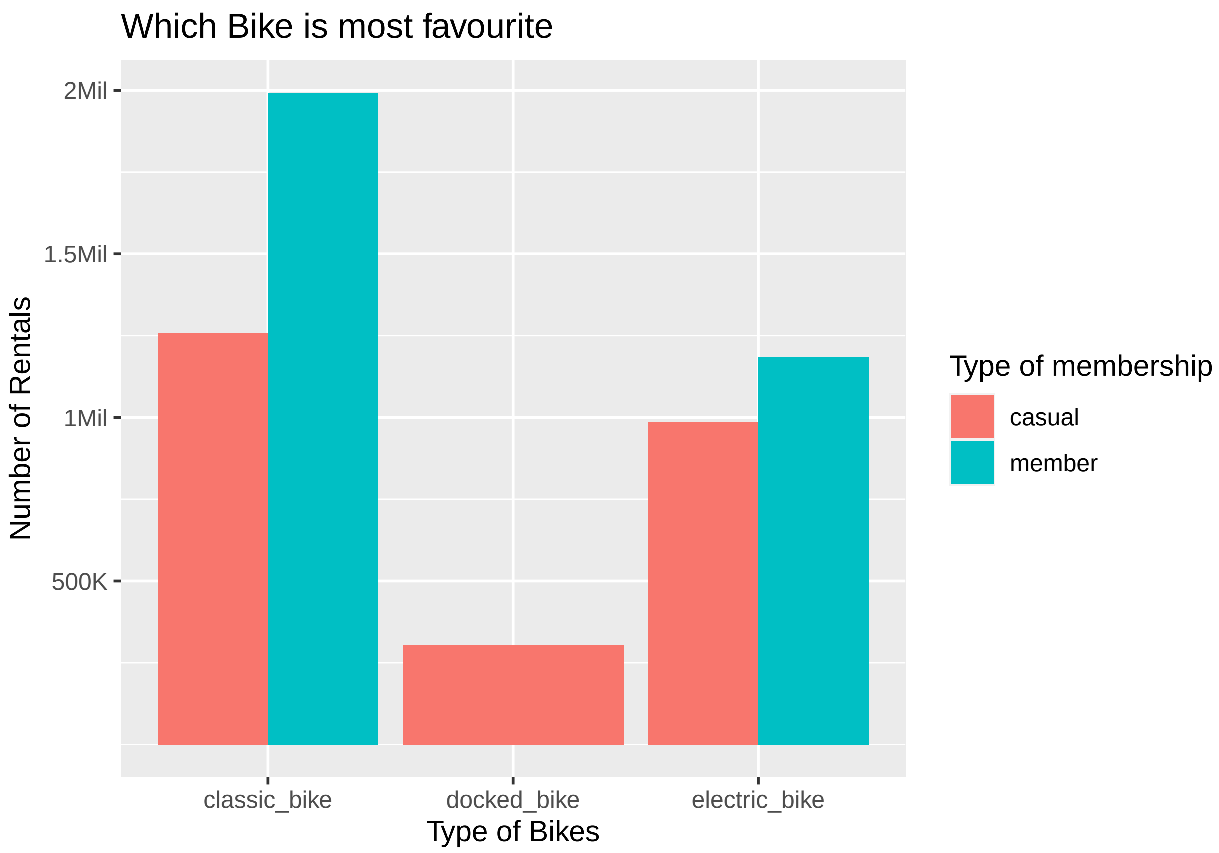
```
## ‘summarise()‘ has grouped output by ’member_casual’. You can override using the
## ‘.groups‘ argument.
```



The rides per month shows that casual riders are more active during the summer months (Jun-Aug) than the long-term members. Conversely, the winter months(Nov-Mar) show very little activity on the part of the casual users. The long-term users are more active in the fall(Sep-Oct), winter(Nov-Mar) and spring(Apr-May) months than the casual riders.

Total rides breakdown by bike types rented

```
trip_data %>%
  ggplot(aes(x= rideable_type, fill = member_casual)) + geom_bar(position = "dodge") +
  labs(x = "Type of Bikes", y="Number of Rentals", title = "Which Bike is most favourite",
       fill="Type of membership") +
  scale_y_continuous(breaks = c(500000, 1000000, 1500000, 2000000),
                     labels = c("500K", "1Mil", "1.5Mil", "2Mil"))
```

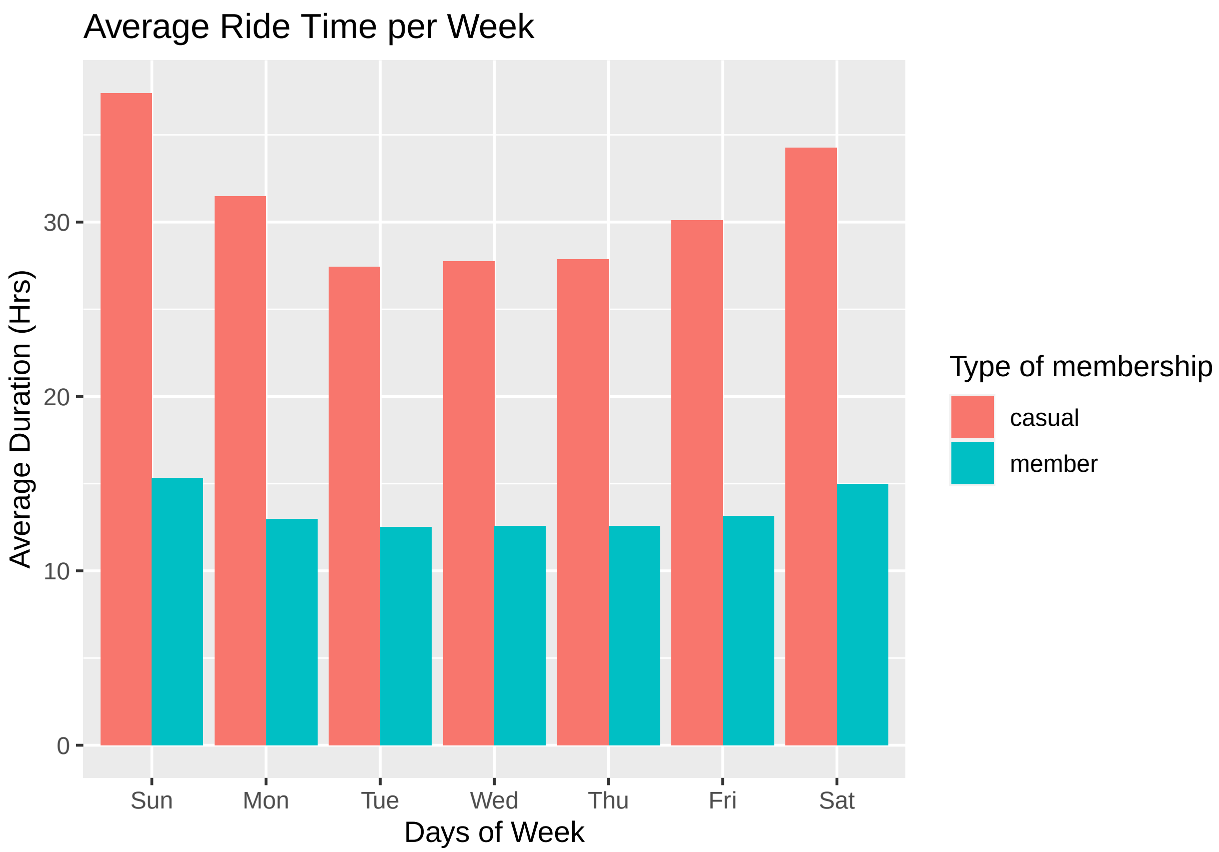


The breakdown of which type of bike is the most popular among either type of user. It shows that two types of bikes classic and electric are most popular. Both types of memberships prefer using the classic bike more than the electric bike. The classic bike is most popular to the long-term users for their commutes. They didn't like docked bike.

Find the average time spent riding by each membership type per individul day

```
trip_data %>%
  mutate(days_week = wday(started_at, label = TRUE)) %>%
  group_by(member_casual, days_week) %>%
  summarise(average_duration = mean(ride_length)) %>%
  ggplot(aes(x=days_week, y=average_duration, fill = member_casual)) +
  geom_col(position = "dodge") +
  labs(x="Days of Week", y="Average Duration (Hrs)",
      title = "Average Ride Time per Week", fill='Type of membership')
```

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



The average ride time shows a stark difference between the casuals and members. Casuals riders overall spend more time using the service than long-term members. They spend near 35 hours at the weekend (Sat-Sun).

What does the data tell us?

Key findings:

- Casual users tended to ride more so in the summer months of Chicago, namely June- August. Their participation exceeded that of the long term members.
- To further that the Casual users spent on average a lot longer time per ride than their long-term counter-parts.
- Long term riders tended to stick more so to classic bikes as opposed to the docked or electric bikes.
- The days of the week also further shows that causal riders prefer to use the service during the weekends as their usage peaked then. Conversely, the long term members use the service more-so throughout the typical work day.

Recommendation

- Giving incentives or rewards for achieving members' milestones to attract casual riders to become members.
- Offer casual riders a membership package with promotions and discounts.
- Host fun biking competitions with prizes at intervals for members on the weekends. Since there are lot of casual riders on weekends,this will also attract them to get a membership.

Other Considerations for Further Exploration

- Include data on whether a casual rider uses a single-ride pass or full-day pass, to analyze how pricing may impact usage.
- Know the preferred type of bikes, locations or most popular routes by casual users. This will help the company by to get insights which might lead them to offer the best promotions in those specific routes to convert more casual users to member users.

Thank you for your time!