Google Data Analytics Capstone

Emilio_Acosta

2022-05-11

Scenario

You are a junior data analyst working in the marketing analyst team in 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.

Business Task

Use historical data derived for the purposes of this project to run analysis and generate insights on how casual and annual members use the platform differently and use those insights to help the marketing team influence casual riders into converting to an annual membership.

Prepare/Process

Data Source: The mock historical data was provided by Cyclistic a fictional company derived for the purposes of this capstone.

Install packages

Install all the necessary packages for analysis.

```
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(lubridate)
```

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

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

```r
library(janitor)
```

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

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

```r
library(tinytex)
```

Load Data

Load all the relevant data and validate the column structure of each data set, once columns are verified combine all the sets into one for easy analysis.

```r
may21 <- read_csv("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.
```

```r
june21 <- read_csv("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.
```

```r
july21 <- read_csv("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.
```

```r
aug21 <- read_csv("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.
```

```r
sep21 <- read_csv("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.
```

```r
oct21 <- read_csv("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.
```

```r
nov21 <- read_csv("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.
```

```
dec21 <- read_csv("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.
```

```
jan22 <- read_csv("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.
```

```
feb22 <- read_csv("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.
```

```
mar22 <- read_csv("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.
```

```
apr22 <- read_csv("202204-divvy-tripdata.csv")
```

```
## Rows: 371249 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.
```

```
#Combining the twelve data sets into one.
all_trips <- bind_rows(may21, june21, july21, aug21, sep21, oct21, nov21, dec21, jan22, feb22, mar22, ap
View(all_trips)
```

Data Cleaning

Make a copy of the data and add the necessary columns and remove the errors in the data set to proceed.

```
distinct(all_trips)   #Removes duplicates
```

```
## # A tibble: 5,757,551 x 13
##    ride_id          rideable_type started_at          ended_at
##    <chr>            <chr>         <dttm>              <dttm>
##  1 C809ED75D6160B2A electric_bike 2021-05-30 11:58:15 2021-05-30 12:10:39
##  2 DD59FDCE0ACACAF3 electric_bike 2021-05-30 11:29:14 2021-05-30 12:14:09
##  3 0AB83CB88C43EFC2 electric_bike 2021-05-30 14:24:01 2021-05-30 14:25:13
##  4 7881AC6D39110C60 electric_bike 2021-05-30 14:25:51 2021-05-30 14:41:04
##  5 853FA701B4582BAF electric_bike 2021-05-30 18:15:39 2021-05-30 18:22:32
##  6 F5E63DFD96B2A737 electric_bike 2021-05-30 11:33:41 2021-05-30 11:57:17
##  7 C884951E36656727 electric_bike 2021-05-30 10:51:37 2021-05-30 11:06:20
##  8 48B60B250FE75AF9 electric_bike 2021-05-05 13:57:03 2021-05-05 14:14:58
##  9 E3D0CC2FE1359880 electric_bike 2021-05-05 11:31:26 2021-05-05 11:34:03
## 10 4382735758ABF2CE electric_bike 2021-05-04 19:51:05 2021-05-04 20:17:26
## # ... with 5,757,541 more rows, and 9 more variables: start_station_name <chr>,
## #   start_station_id <chr>, end_station_name <chr>, end_station_id <chr>,
## #   start_lat <dbl>, start_lng <dbl>, end_lat <dbl>, end_lng <dbl>,
## #   member_casual <chr>
```

```
remove_empty(all_trips)   #Remove empty cells
```

```
## value for "which" not specified, defaulting to c("rows", "cols")
## # A tibble: 5,757,551 x 13
##    ride_id          rideable_type started_at          ended_at
##    <chr>            <chr>         <dttm>              <dttm>
##  1 C809ED75D6160B2A electric_bike 2021-05-30 11:58:15 2021-05-30 12:10:39
##  2 DD59FDCE0ACACAF3 electric_bike 2021-05-30 11:29:14 2021-05-30 12:14:09
##  3 0AB83CB88C43EFC2 electric_bike 2021-05-30 14:24:01 2021-05-30 14:25:13
##  4 7881AC6D39110C60 electric_bike 2021-05-30 14:25:51 2021-05-30 14:41:04
##  5 853FA701B4582BAF electric_bike 2021-05-30 18:15:39 2021-05-30 18:22:32
##  6 F5E63DFD96B2A737 electric_bike 2021-05-30 11:33:41 2021-05-30 11:57:17
##  7 C884951E36656727 electric_bike 2021-05-30 10:51:37 2021-05-30 11:06:20
##  8 48B60B250FE75AF9 electric_bike 2021-05-05 13:57:03 2021-05-05 14:14:58
##  9 E3D0CC2FE1359880 electric_bike 2021-05-05 11:31:26 2021-05-05 11:34:03
## 10 4382735758ABF2CE electric_bike 2021-05-04 19:51:05 2021-05-04 20:17:26
## # ... with 5,757,541 more rows, and 9 more variables: start_station_name <chr>,
## #   start_station_id <chr>, end_station_name <chr>, end_station_id <chr>,
## #   start_lat <dbl>, start_lng <dbl>, end_lat <dbl>, end_lng <dbl>,
## #   member_casual <chr>
```

```
#Removes the unwanted columns and create a copy names all_trips2
all_trips2 <- all_trips %>%
  select(-c(start_station_name, start_station_id, end_station_name, end_station_id))
#Adding columns necessary for analysis
all_trips2$date <- as.Date(all_trips2$started_at)
all_trips2$month <- format(as.Date(all_trips2$date), "%m")
all_trips2$day <- format(as.Date(all_trips2$date), "%d")
all_trips2$year <- format(as.Date(all_trips2$date), "%Y")
all_trips2$day_of_week <- format(as.Date(all_trips2$date), "%A")
all_trips2$time <- format(all_trips2$started_at, format = "%H:%M")
all_trips2$time <- as.POSIXct(all_trips2$time, format = "%H:%M")    #Change format for time column for
all_trips2$ride_length <- as.double(difftime(all_trips2$ended_at, all_trips2$started_at))/60  #Calculat
#Remove the docked bike type and any trip duration that is negative or longer than 24hrs
all_trips2 <- all_trips2[!(all_trips2$rideable_type == "docked_bike" | all_trips2$ride_length < 0),]
all_trips2 <- all_trips2[!(all_trips2$ride_length > 1440),]
```

Reorder

The data is not in order so it needs to be reordered to reflect the chronological order of the week.

```
#Set the date order
all_trips2$day_of_week <- ordered(all_trips2$day_of_week, levels=c("Monday", "Tuesday", "Wednesday","Thu
```

Analyze data

```
summary(all_trips2$ride_length)
```

```
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
##    0.000    6.267   11.050   17.076   19.717 1439.950
```

```
table(all_trips2$member_casual)
```

```
##
##  casual  member
## 2242623 3220621
```

```
table(all_trips2$rideable_type)
```

```
##
##  classic_bike electric_bike
##       3199922       2263322
```

```
aggregate(all_trips2$ride_length ~ all_trips2$member_casual, all_trips2, sum)
```

```
##   all_trips2$member_casual all_trips2$ride_length
## 1                   casual               51688171
## 2                   member               41604355
```

```
aggregate(all_trips2$ride_length ~ all_trips2$member_casual, FUN = mean)
```

```
##   all_trips2$member_casual all_trips2$ride_length
## 1                   casual               23.04809
## 2                   member               12.91812
```

```
aggregate(all_trips2$ride_length ~ all_trips2$member_casual, FUN = median)
```

```
##   all_trips2$member_casual all_trips2$ride_length
## 1                   casual              14.450000
## 2                   member               9.183333
```

```
aggregate(all_trips2$ride_length ~ all_trips2$member_casual, FUN = max)
```

```
##   all_trips2$member_casual all_trips2$ride_length
## 1                   casual               1439.917
## 2                   member               1439.950
```

```
aggregate(all_trips2$ride_length ~ all_trips2$member_casual, FUN = min)
```

```
##   all_trips2$member_casual all_trips2$ride_length
## 1                   casual                      0
## 2                   member                      0
```

```
aggregate(all_trips2$ride_length ~ all_trips2$member_casual + all_trips2$day_of_week, FUN = mean)
```

```
##    all_trips2$member_casual all_trips2$day_of_week all_trips2$ride_length
## 1                    casual                 Monday               23.06257
## 2                    member                 Monday               12.46918
## 3                       casual                Tuesday               20.22223
## 4                       member                Tuesday               12.06886
## 5                       casual              Wednesday               20.27521
## 6                       member              Wednesday               12.27530
## 7                       casual               Thursday               20.39349
## 8                       member               Thursday               12.22453
## 9                       casual                 Friday               21.60241
## 10                      member                 Friday               12.66182
## 11                      casual               Saturday               25.37129
## 12                      member               Saturday               14.49584
## 13                      casual                 Sunday               26.58824
## 14                      member                 Sunday               14.73290
```

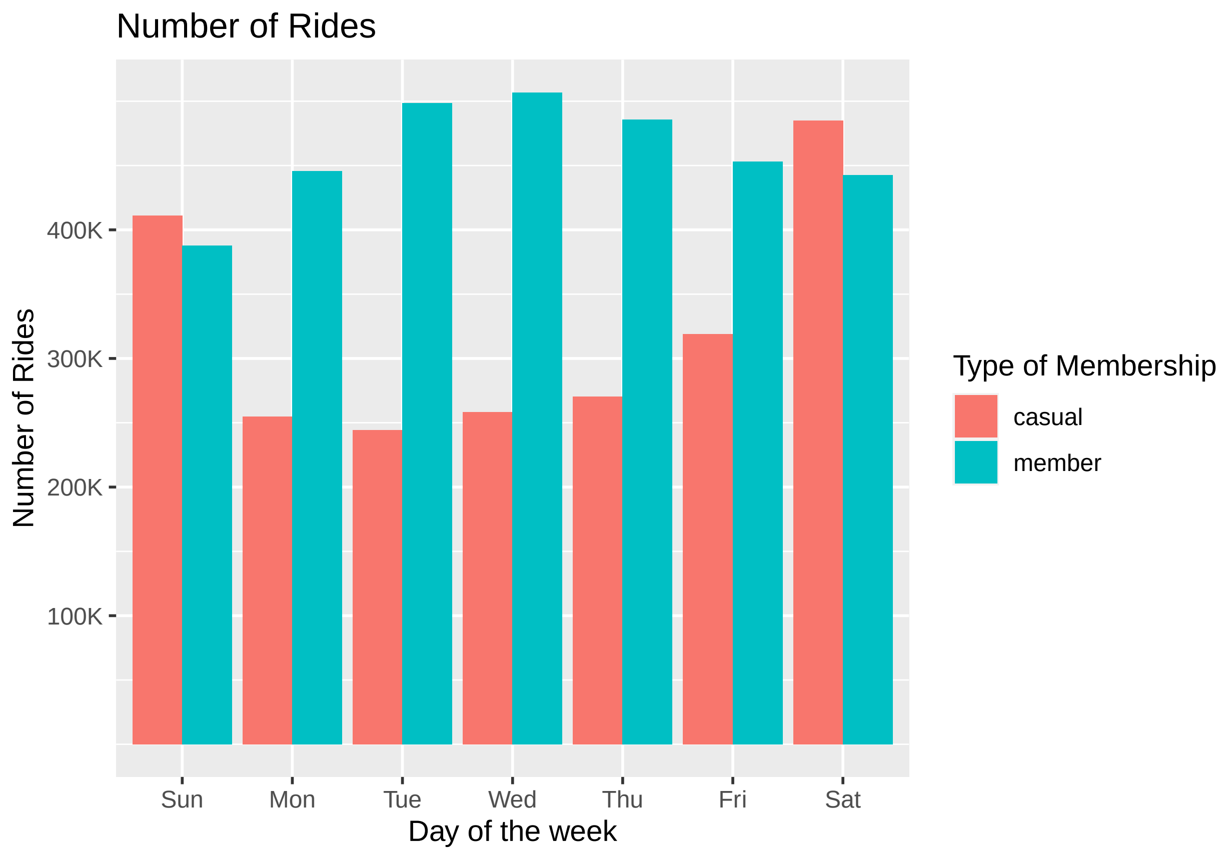
```
all_trips2 %>%
  mutate(weekday = wday(started_at, label = TRUE)) %>%
  group_by(member_casual, weekday) %>%
  summarize(number_of_rides = n(), average_duration = mean(ride_length)) %>%
  arrange(member_casual, weekday)
```

```
## `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 weekday number_of_rides average_duration
##    <chr>         <ord>             <int>            <dbl>
##  1 casual        Sun              411042             26.6
##  2 casual        Mon              254779             23.1
##  3 casual        Tue              244366             20.2
##  4 casual        Wed              258437             20.3
##  5 casual        Thu              270231             20.4
##  6 casual        Fri              318832             21.6
##  7 casual        Sat              484936             25.4
##  8 member        Sun              387931             14.7
##  9 member        Mon              445555             12.5
## 10 member        Tue              498615             12.1
## 11 member        Wed              506887             12.3
## 12 member        Thu              485769             12.2
## 13 member        Fri              453212             12.7
## 14 member        Sat              442652             14.5
```

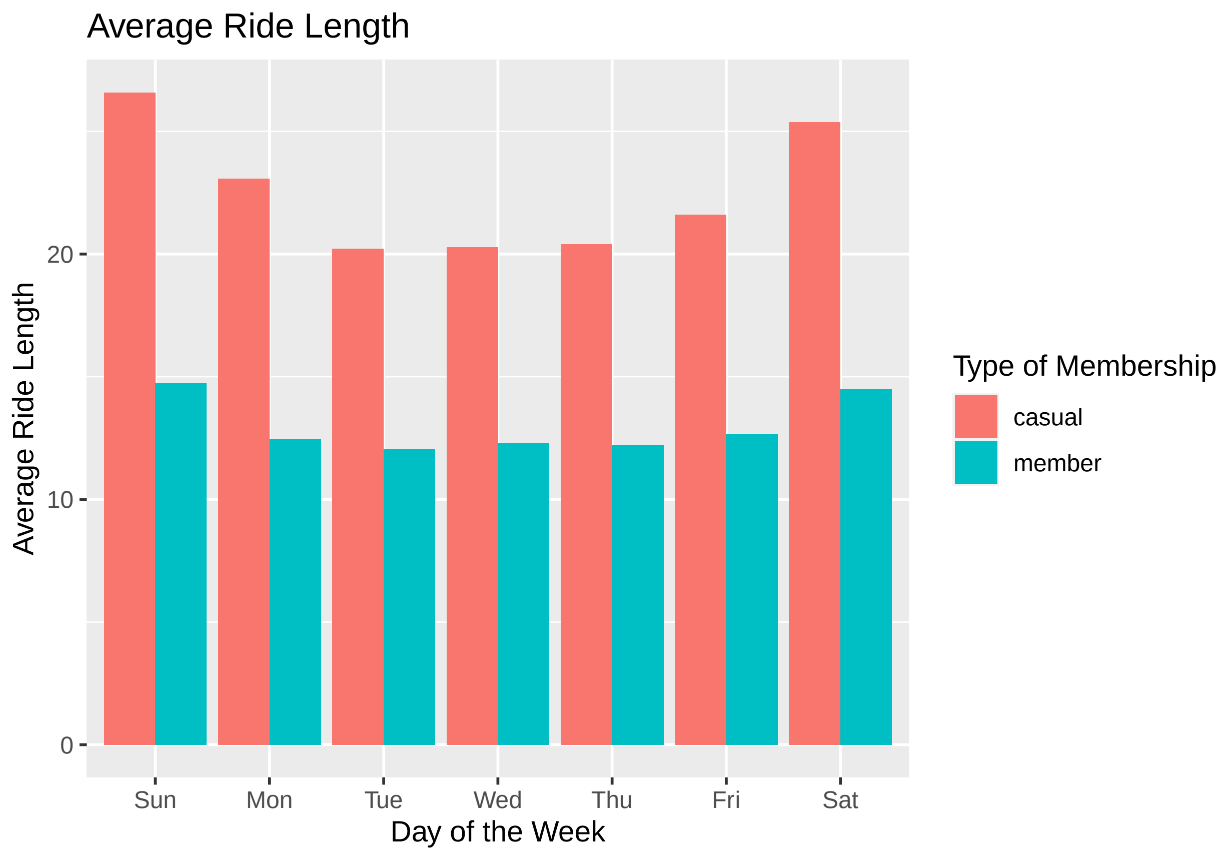
Visualize findings

```
all_trips2 %>%
  mutate(weekday = wday(started_at, label = TRUE)) %>%
  group_by(member_casual, weekday) %>%
  summarize(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 = "Number of Rides", x = "Day of the week", y = "Number of Rides", fill = "Type of Membersh
  scale_y_continuous(breaks = c(100000, 200000, 300000, 400000), labels = c("100K", "200K", "300K", "40
## 'summarise()' has grouped output by 'member_casual'. You can override using the
## '.groups' argument.
```

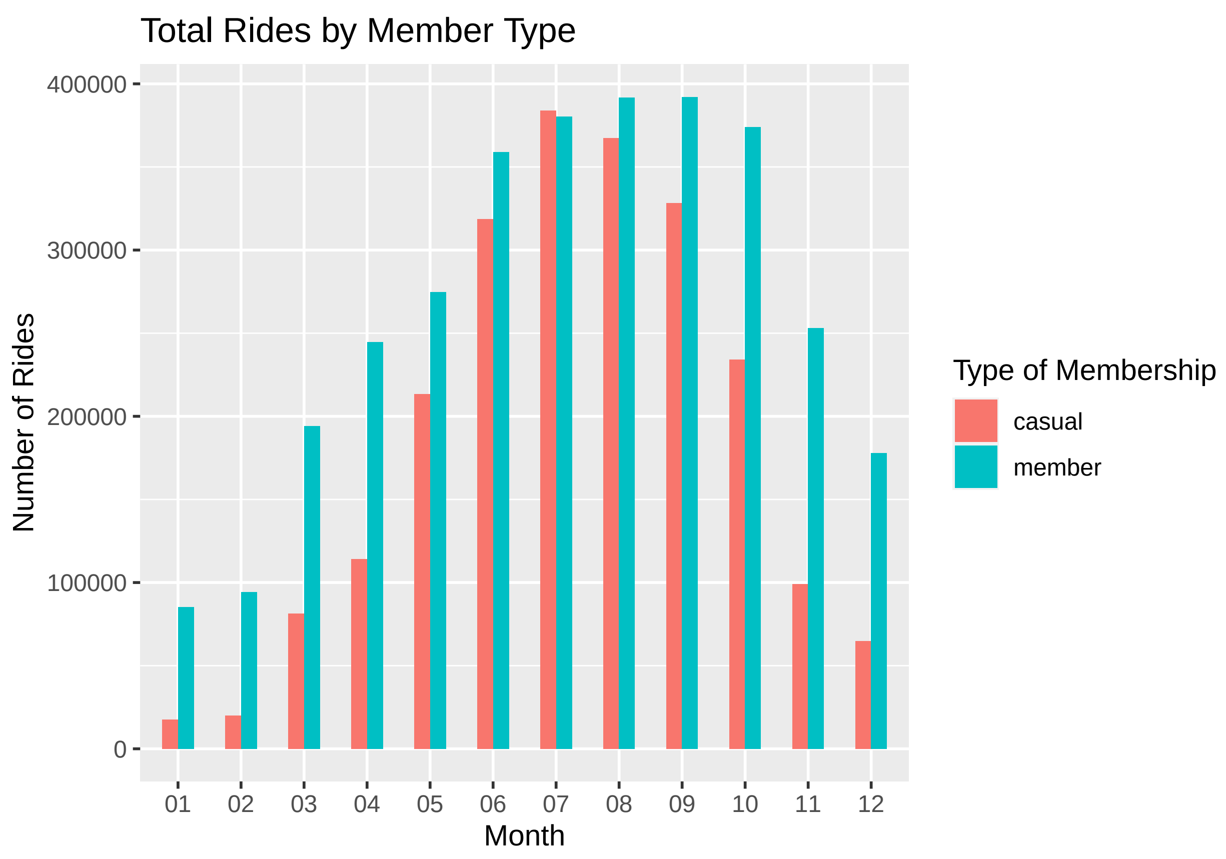


```
all_trips2 %>%
  mutate(weekday = wday(started_at, label = TRUE)) %>%
  group_by(member_casual, weekday) %>%
  summarize(number_of_rides = n(), average_duration = mean(ride_length)) %>%
  arrange(member_casual, weekday) %>%
  ggplot(aes(x = weekday, y = average_duration, fill = member_casual)) +
  geom_col(position = "dodge") +
  labs(title = "Average Ride Length", x = "Day of the Week", y = "Average Ride Length", fill = "Type of
```

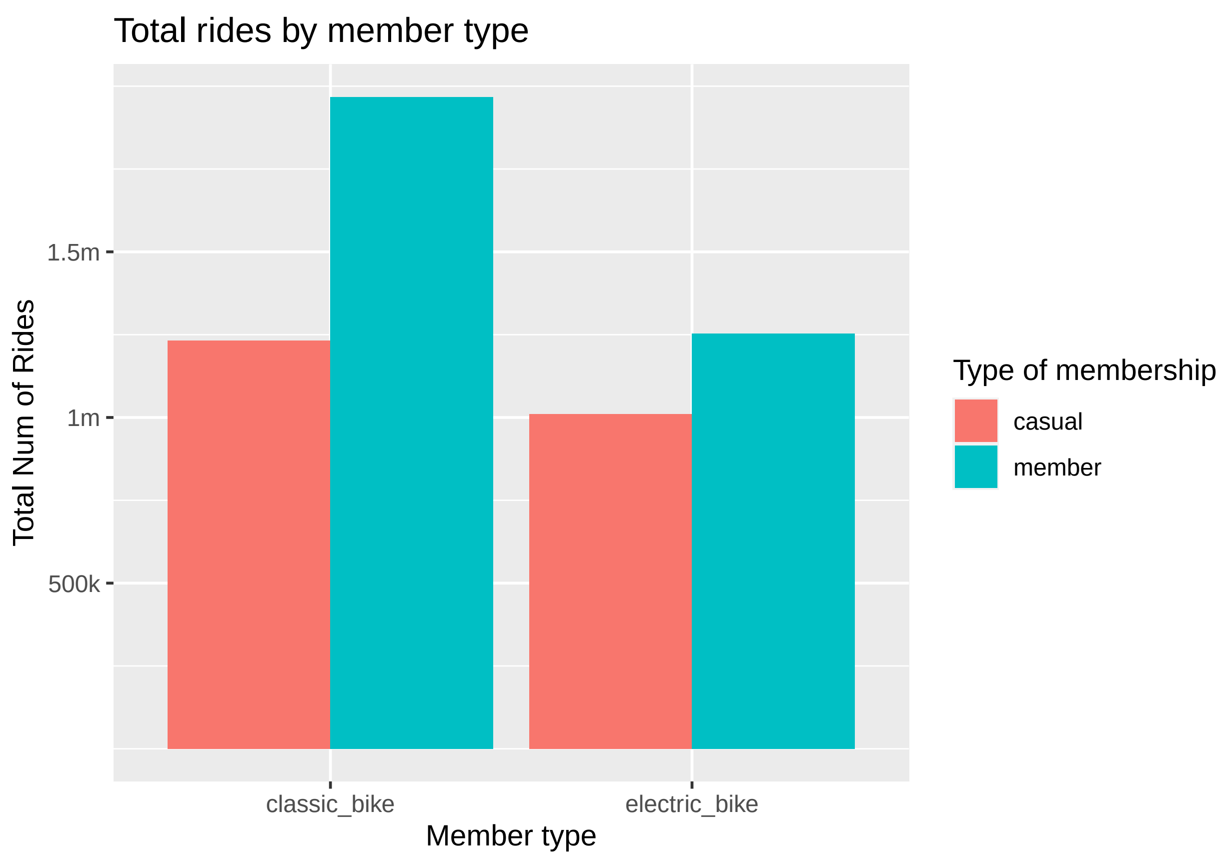
```
## 'summarise()' has grouped output by 'member_casual'. You can override using the
## '.groups' argument.
```



```
all_trips2 %>%
  group_by(member_casual, month) %>%
  summarize(number_of_rides = n(), .groups = "drop") %>%
  arrange(member_casual, month) %>%
  ggplot(aes(x = month, y = number_of_rides, fill = member_casual)) +
  geom_col(width = 0.5, position = position_dodge(width = 0.5)) +
  labs(title = "Total Rides by Member Type", x = "Month", y = "Number of Rides", fill = "Type of Members
  scale_y_continuous(labels = function(x) format(x, scientific = FALSE))
```

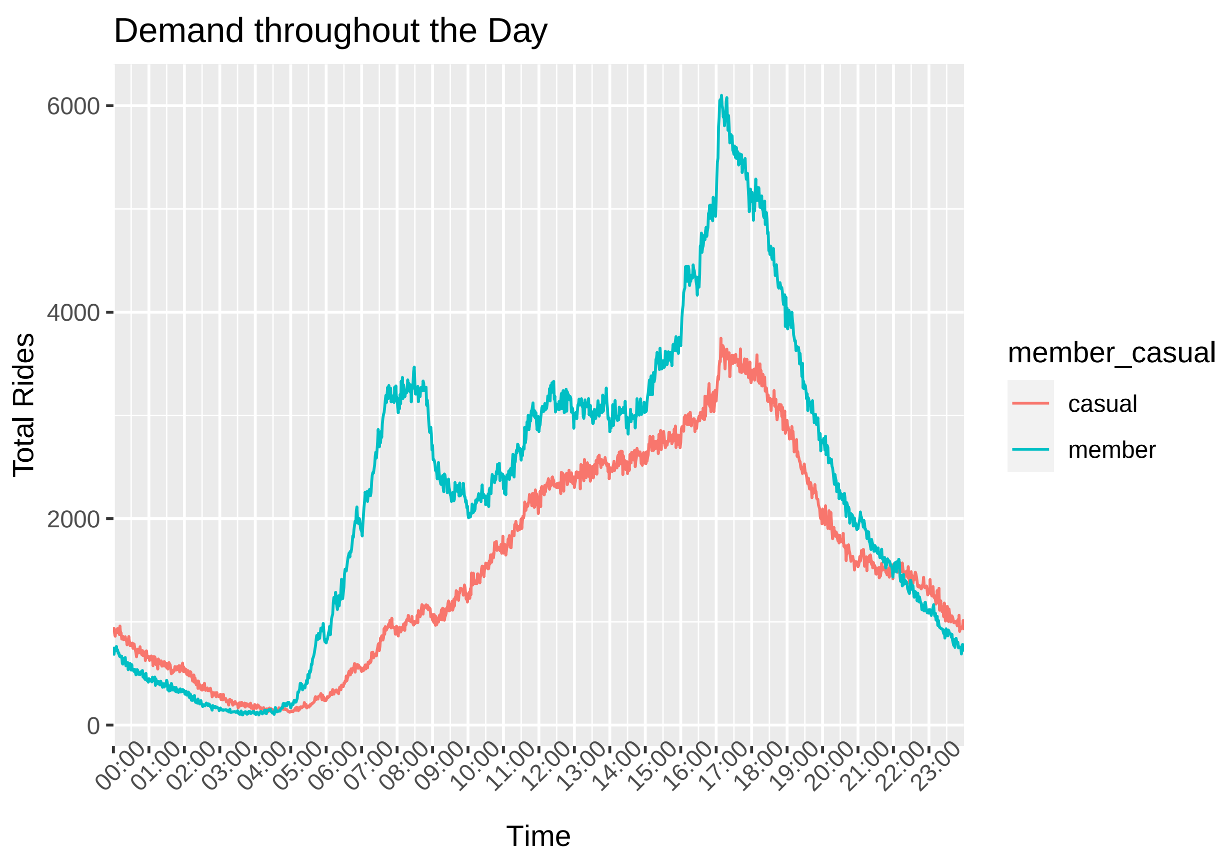


```
all_trips2 %>%
  ggplot(aes(x = rideable_type, fill = member_casual)) +
  geom_bar(position = "dodge") +
  labs(title = "Total rides by member type", x = "Member type", y = "Total Num of Rides", fill = "Type o
  scale_y_continuous(breaks = c(500000, 1000000, 1500000), labels = c("500k", "1m", "1.5m"))
```



```
all_trips2 %>%
  group_by(member_casual, time) %>%
  summarize(total_rides = n()) %>%
  ggplot(aes(x=time, y=total_rides, color = member_casual, group = member_casual)) +
  geom_line() + scale_x_datetime(date_breaks = "1 hour",
                                 date_labels = "%H:%M", expand = c(0,0)) +
  labs(title = "Demand throughout the Day", x = "Time", y = "Total Rides") +
  theme(axis.text.x = element_text(angle = 45))
```

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



Key takeaways

- Members ride at a higher rate where casual riders ride for almost double the duration.
- Casual rider usage peaks during summer months and weekends.
- Members use the service mostly to commute based on peaks during rush hour.

Recommendations

- The marketing team can raise the annual membership sales by running promotions during winter months.
- They can also run additional ads after evening rush hour when demand for service is at it's lowest.
- A change in the pay structure may also persuade casual riders to convert into annual members. **Ex.** Discounted fare during the week may cause a boost in the number of casual riders during the week, however with increased prices during the weekends when the demand is at it's highest may nudge casual riders to considering an annual membership.

Additional Data for Future Analysis

- Price - The pricing structure of the memberships may allow for better recommendations.
- Age - Can be a key indicator on how one uses the service and could further serve to create more insights.
- Gender - Seeing how the different genders use the service can help target certain demographics.