Introduction

This analysis case study is the Capstone project for the Google Data Analytics Professional Certificate. The case study involves a fictional bike-share company based in Chicago, Cyclistic, that wants to start a new marketing strategy that will be key for future growth of the company.

Scenario

Lily Moreno, the director of marketing, wants to start a new marketing strategy to convert casual riders into annual members. She believes with the right campaign there is a very good chance of conversion with the company's name recognition and access to user-friendly bike options. These user-friendly bike options include reclining bikes, hand tricycles, and cargo bikes which makes Cyclistic more inclusive to people with disabilities and riders who can't use a standard two-wheeled bike. Lily has tasked the marketing analytics team to analyze past user data to find trends and habits of Cyclistic's users to help create this marketing campaign.

Stakeholders

Cyclistic - The bike-share company with more than 5.800 bikes and 600 docking stations all over Chicago.

Lily Moreno - The director of marketing who has requested the analysis for her new marketing strategy.

Cyclistic Marketing Analytics Team - A team of data analysts who are responsible for collecting, analyzing, and reporting data that helps guide marketing strategy.

Cyclistic Executive Team - Detail-oriented executive team that will decide whether to approve the recommended marketing program.

Identify the business task

The new marketing strategy requires three components: behavioral differences between annual members and casual riders, reasons why a casual rider would buy Cyclistic annual memberships, and how digital media can influence casual riders to becoming members. The marketing analytics team is tasked with using past user data to find the behavioral differences between annual members and casual riders and report their findings.

Data sources

User data from the past 12 months, November 2020 - October 2021 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.

Documentation, cleaning and preparation of data for analysis

Tools for analysis

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

Start documentation and preparation of the data

Install the correct packages to start

```
library(tidyverse)
## ── Attaching packages ─────────────────────────────────────── tidyverse
1.3.1 ──
## ✓ ggplot2 3.3.5     ✓ purrr   0.3.4
## ✓ tibble  3.1.6     ✓ dplyr   1.0.7
## ✓ tidyr   1.1.4     ✓ stringr 1.4.0
## ✓ readr   2.1.1     ✓ forcats 0.5.1
## ── Conflicts ────────────────────────────────────────── tidyverse_confli
cts() ──
## 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(janitor)
##
## Attaching package: 'janitor'
## The following objects are masked from 'package:stats':
##
##     chisq.test, fisher.test
```

Load the data and combine all data sets for easier analysis

```
Nov2020 <- read_csv("~/Desktop/divvy data/csv/202011-divvy-tripdata.csv")
## Rows: 259716 Columns: 13
## ── Column specification ────────────────────────────────────────────────
────────
## Delimiter: ","
## chr  (5): ride_id, rideable_type, start_station_name, end_station_name,
memb...
## dbl  (6): start_station_id, end_station_id, start_lat, start_lng, end_la
t, e...
```

```
## 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
Dec2020 <- read_csv("~/Desktop/divvy data/csv/202012-divvy-tripdata.csv")
```

```
## Rows: 131573 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
Jan2021 <- read_csv("~/Desktop/divvy data/csv/202101-divvy-tripdata.csv")
```

```
## Rows: 96834 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
Feb2021 <- read_csv("~/Desktop/divvy data/csv/202102-divvy-tripdata.csv")
```

```
## Rows: 49622 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.
```

```
Mar2021 <- read_csv("~/Desktop/divvy data/csv/202103-divvy-tripdata.csv")
```

```
## Rows: 228496 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.
```

```
Apr2021 <- read_csv("~/Desktop/divvy data/csv/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.
```

```
May2021 <- read_csv("~/Desktop/divvy data/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.
```

```
Jun2021 <- read_csv("~/Desktop/divvy data/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.
```

```
Jul2021 <- read_csv("~/Desktop/divvy data/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.
```

```
Aug2021 <- read_csv("~/Desktop/divvy data/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.
```

```
Sep2021 <- read_csv("~/Desktop/divvy data/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.
```

```
Oct2021 <- read_csv("~/Desktop/divvy data/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.
```

```
bike_data <- rbind(Nov2020, Dec2020, Jan2021, Feb2021, Mar2021, Apr2021,
                   May2021, Jun2021, Jul2021, Aug2021, Sep2021, Oct2021)
```

Make a copy of the data to have a backup

```
bike_data1 <- bike_data
```

Add relevant columns

```
bike_data1$date <- as.Date(bike_data1$started_at) #add a date column
bike_data1$month <- format(as.Date(bike_data$started_at), "%b_%y")   #add a month column formatted as short hand_year (e.g. feb_2021)
bike_data1$day <- format(as.Date(bike_data1$date), "%d")  #add a day column
bike_data1$year <- format(as.Date(bike_data1$date), "%Y")  #add a year column
```

```
bike_data1$weekday <- format(as.Date(bike_data1$date), "%A")  #add a day of
week column

bike_data1$time <- format(bike_data1$started_at, format = "%H:%M")  #add a
time started column

bike_data1$time <- as.POSIXct(bike_data1$time, format = "%H:%M")  #change f
ormat for the time column for purposes later

bike_data1$ride_length <- (as.double(difftime(bike_data1$ended_at, bike_dat
a1$started_at))) /60  #calculate ride length in minutes
```

Look at specifics of the data

```
str(bike_data1)  #check the structure of the data
```

```
## spec_tbl_df [5,378,834 × 20] (S3: spec_tbl_df/tbl_df/tbl/data.frame)

##  $ ride_id           : chr [1:5378834] "BD0A6FF6FFF9B921" "96A7A7A4BDE4F
82D" "C61526D06582BDC5" "E533E89C32080B9E" ...

##  $ rideable_type     : chr [1:5378834] "electric_bike" "electric_bike" "
electric_bike" "electric_bike" ...

##  $ started_at        : POSIXct[1:5378834], format: "2020-11-01 13:36:00"
"2020-11-01 10:03:26" ...

##  $ ended_at          : POSIXct[1:5378834], format: "2020-11-01 13:45:40"
"2020-11-01 10:14:45" ...

##  $ start_station_name: chr [1:5378834] "Dearborn St & Erie St" "Franklin
St & Illinois St" "Lake Shore Dr & Monroe St" "Leavitt St & Chicago Ave" ..
.

##  $ start_station_id  : chr [1:5378834] "110" "672" "76" "659" ...

##  $ end_station_name  : chr [1:5378834] "St. Clair St & Erie St" "Noble S
t & Milwaukee Ave" "Federal St & Polk St" "Stave St & Armitage Ave" ...

##  $ end_station_id    : chr [1:5378834] "211" "29" "41" "185" ...

##  $ start_lat         : num [1:5378834] 41.9 41.9 41.9 41.9 41.9 ...

##  $ start_lng         : num [1:5378834] -87.6 -87.6 -87.6 -87.7 -87.6 ...

##  $ end_lat           : num [1:5378834] 41.9 41.9 41.9 41.9 41.9 ...

##  $ end_lng           : num [1:5378834] -87.6 -87.7 -87.6 -87.7 -87.6 ...

##  $ member_casual     : chr [1:5378834] "casual" "casual" "casual" "casua
l" ...

##  $ date              : Date[1:5378834], format: "2020-11-01" "2020-11-01
" ...

##  $ month             : chr [1:5378834] "Nov_20" "Nov_20" "Nov_20" "Nov_2
0" ...

##  $ day               : chr [1:5378834] "01" "01" "01" "01" ...

##  $ year              : chr [1:5378834] "2020" "2020" "2020" "2020" ...

##  $ weekday           : chr [1:5378834] "Sunday" "Sunday" "Sunday" "Sunda
y" ...

##  $ time              : POSIXct[1:5378834], format: "2021-12-16 13:36:00"
"2021-12-16 10:03:00" ...
```

```
##  $ ride_length       : num [1:5378834] 9.67 11.32 29.02 9.25 33.45 ...
##  - attr(*, "spec")=
##   .. cols(
##   ..   ride_id = col_character(),
##   ..   rideable_type = col_character(),
##   ..   started_at = col_datetime(format = ""),
##   ..   ended_at = col_datetime(format = ""),
##   ..   start_station_name = col_character(),
##   ..   start_station_id = col_double(),
##   ..   end_station_name = col_character(),
##   ..   end_station_id = col_double(),
##   ..   start_lat = col_double(),
##   ..   start_lng = col_double(),
##   ..   end_lat = col_double(),
##   ..   end_lng = col_double(),
##   ..   member_casual = col_character()
##   .. )
##  - attr(*, "problems")=<externalptr>
```

```
colnames(bike_data1)  #check the column names
```

```
##  [1] "ride_id"            "rideable_type"      "started_at"
##  [4] "ended_at"           "start_station_name" "start_station_id"
##  [7] "end_station_name"   "end_station_id"     "start_lat"
## [10] "start_lng"          "end_lat"            "end_lng"
## [13] "member_casual"      "date"               "month"
## [16] "day"                "year"               "weekday"
## [19] "time"               "ride_length"
```

```
dim(bike_data1)  #check dimensions of the data
```

```
## [1] 5378834      20
```

```
nrow(bike_data1)  #check the number of rows
```

```
## [1] 5378834
```

```
summary(bike_data1)  #summary for the data
```

```
##    ride_id          rideable_type        started_at
##  Length:5378834     Length:5378834     Min.   :2020-11-01 00:00:08
##  Class :character   Class :character   1st Qu.:2021-05-17 12:45:18
##  Mode  :character   Mode  :character   Median :2021-07-13 22:33:14
##                                        Mean   :2021-06-27 18:37:41
##                                        3rd Qu.:2021-09-02 18:18:14
##                                         Max.   :2021-10-31 23:59:49
##
##     ended_at                       start_station_name start_station_id
##  Min.   :2020-11-01 00:02:20   Length:5378834     Length:5378834
##  1st Qu.:2021-05-17 13:07:36   Class :character   Class :character
##  Median :2021-07-13 22:57:23   Mode  :character   Mode  :character
##  Mean   :2021-06-27 18:58:10
##  3rd Qu.:2021-09-02 18:35:16
##  Max.   :2021-11-03 21:45:48
##
##  end_station_name   end_station_id       start_lat       start_lng
##  Length:5378834     Length:5378834     Min.   :41.64   Min.   :-87.84
##  Class :character   Class :character   1st Qu.:41.88   1st Qu.:-87.66
##  Mode  :character   Mode  :character   Median :41.90   Median :-87.64
##                                         Mean   :41.90   Mean   :-87.65
##                                         3rd Qu.:41.93   3rd Qu.:-87.63
##                                         Max.   :42.08   Max.   :-87.52
##
##     end_lat         end_lng       member_casual           date
##  Min.   :41.51   Min.   :-88.07   Length:5378834     Min.   :2020-11-01
##  1st Qu.:41.88   1st Qu.:-87.66   Class :character   1st Qu.:2021-05-17
##  Median :41.90   Median :-87.64   Mode  :character   Median :2021-07-13
##  Mean   :41.90   Mean   :-87.65                      Mean   :2021-06-27
##  3rd Qu.:41.93   3rd Qu.:-87.63                      3rd Qu.:2021-09-02
##  Max.   :42.17   Max.   :-87.44                      Max.   :2021-10-31
##  NA's   :4831    NA's   :4831
##     month                day                 year               weekday
##  Length:5378834     Length:5378834     Length:5378834     Length:5378834
##  Class :character   Class :character   Class :character   Class :charact
er
##  Mode  :character   Mode  :character   Mode  :character   Mode  :charact
er
##
##
##
##
##      time                         ride_length
##  Min.   :2021-12-16 00:00:00   Min.   :-29049.97
##  1st Qu.:2021-12-16 11:37:00   1st Qu.:     6.97
##  Median :2021-12-16 15:32:00   Median :    12.38
##  Mean   :2021-12-16 14:45:03   Mean   :    20.49
##  3rd Qu.:2021-12-16 18:22:00   3rd Qu.:    22.43
##  Max.   :2021-12-16 23:59:00   Max.   : 55944.15
##
```

Start Cleaning the data

```
bike_data1 <- distinct(bike_data1)  #remove any duplicates
bike_data1 <- bike_data1[!bike_data1$ride_length<1,] #get rid of negative r
ides
bike_data1 <- bike_data1[!bike_data1$ride_length>1440,] #get rid of too lon
g rides - rides should be limited to 1 day or 1440 minutes


#change a few column names for clarification
bike_data1 <- rename(bike_data1, customer_type = member_casual)
bike_data1 <- rename(bike_data1, bike_type = rideable_type)


## Filter out data we will not be using and remove missing data
bike_data1 <- bike_data1 %>% select(bike_type, customer_type, started_at, d
ate, month, day, year, weekday, time, ride_length)
drop_na(bike_data1)
```

```
## # A tibble: 5,292,949 × 10
##    bike_type     customer_type started_at          date       month  day  
year
##    <chr>         <chr>         <dttm>              <date>     <chr>  <ch
r> <chr>
##  1 electric_bike casual        2020-11-01 13:36:00 2020-11-01 Nov_20 01   
2020
##  2 electric_bike casual        2020-11-01 10:03:26 2020-11-01 Nov_20 01   
2020
##  3 electric_bike casual        2020-11-01 00:34:05 2020-11-01 Nov_20 01   
2020
##  4 electric_bike casual        2020-11-01 00:45:16 2020-11-01 Nov_20 01   
2020
##  5 electric_bike casual        2020-11-01 15:43:25 2020-11-01 Nov_20 01   
2020
##  6 electric_bike casual        2020-11-14 15:55:17 2020-11-14 Nov_20 14   
2020
##  7 electric_bike casual        2020-11-14 16:47:29 2020-11-14 Nov_20 14   
2020
```

```
##  8 electric_bike casual        2020-11-14 16:04:15 2020-11-14 Nov_20 14
2020

##  9 electric_bike casual        2020-11-14 16:24:09 2020-11-14 Nov_20 14
2020

## 10 electric_bike casual        2020-11-14 01:24:22 2020-11-14 Nov_20 14
2020

## # … with 5,292,939 more rows, and 3 more variables: weekday <chr>, time
<dttm>,

## #   ride_length <dbl>
```

```
remove_empty(bike_data1)
```

```
## value for "which" not specified, defaulting to c("rows", "cols")

## # A tibble: 5,292,949 × 10

##    bike_type     customer_type started_at          date       month  day
year

##    <chr>         <chr>         <dttm>              <date>     <chr>  <ch
r> <chr>

##  1 electric_bike casual        2020-11-01 13:36:00 2020-11-01 Nov_20 01
2020

##  2 electric_bike casual        2020-11-01 10:03:26 2020-11-01 Nov_20 01
2020

##  3 electric_bike casual        2020-11-01 00:34:05 2020-11-01 Nov_20 01
2020

##  4 electric_bike casual        2020-11-01 00:45:16 2020-11-01 Nov_20 01
2020

##  5 electric_bike casual        2020-11-01 15:43:25 2020-11-01 Nov_20 01
2020

##  6 electric_bike casual        2020-11-14 15:55:17 2020-11-14 Nov_20 14
2020

##  7 electric_bike casual        2020-11-14 16:47:29 2020-11-14 Nov_20 14
2020

##  8 electric_bike casual        2020-11-14 16:04:15 2020-11-14 Nov_20 14
2020

##  9 electric_bike casual        2020-11-14 16:24:09 2020-11-14 Nov_20 14
2020

## 10 electric_bike casual        2020-11-14 01:24:22 2020-11-14 Nov_20 14
2020

## # … with 5,292,939 more rows, and 3 more variables: weekday <chr>, time
<dttm>,

## #   ride_length <dbl>
```

```
remove_missing(bike_data1)
```

```
## # A tibble: 5,292,949 × 10

##    bike_type     customer_type started_at          date       month  day
year

##    <chr>         <chr>         <dttm>              <date>     <chr>  <ch
r> <chr>
```

```
##  1 electric_bike casual        2020-11-01 13:36:00 2020-11-01 Nov_20 01
2020

##  2 electric_bike casual        2020-11-01 10:03:26 2020-11-01 Nov_20 01
2020

##  3 electric_bike casual        2020-11-01 00:34:05 2020-11-01 Nov_20 01
2020

##  4 electric_bike casual        2020-11-01 00:45:16 2020-11-01 Nov_20 01
2020

##  5 electric_bike casual        2020-11-01 15:43:25 2020-11-01 Nov_20 01
2020

##  6 electric_bike casual        2020-11-14 15:55:17 2020-11-14 Nov_20 14
2020

##  7 electric_bike casual        2020-11-14 16:47:29 2020-11-14 Nov_20 14
2020

##  8 electric_bike casual        2020-11-14 16:04:15 2020-11-14 Nov_20 14
2020

##  9 electric_bike casual        2020-11-14 16:24:09 2020-11-14 Nov_20 14
2020

## 10 electric_bike casual        2020-11-14 01:24:22 2020-11-14 Nov_20 14
2020

## # … with 5,292,939 more rows, and 3 more variables: weekday <chr>, time
<dttm>,

## #   ride_length <dbl>
```

Put data in order

```
#this will help keep the results of the analysis in order based on day of w
eek and by month to avoid confusion

bike_data1$weekday <- ordered(bike_data1$weekday, levels=c("Monday", "Tuesd
ay", "Wednesday", "Thursday",

                                                         "Friday", "Satur
day", "Sunday"))

bike_data1$month <- ordered(bike_data1$month, levels=c("Nov_20", "Dec_20",
"Jan_21", "Feb_21", "Mar_21", "Apr_21",  "May_21", "Jun_21","Jul_21", "Aug_
21", "Sep_21", "Oct_21"))
```

Analyzing the data

Look at the specifics of what the data shows

```
#shows the min, max, median, and average ride lengths

summary(bike_data1$ride_length)

##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.

##    1.000    7.183   12.567   20.327   22.633 1439.367

##looks at total number of customers broken down by membership details
```

```r
table(bike_data1$customer_type)
```

```
## 
##  casual  member 
## 2433787 2859162
```

```r
##looks at total rides for each customer type in minutes
setNames(aggregate(ride_length ~ customer_type, bike_data1, sum), c("customer_type", "total_ride_length(mins)"))
```

```
##   customer_type total_ride_length(mins)
## 1        casual                67707491
## 2        member                39881898
```

```r
##look at rides based on customer type
bike_data1 %>%
  group_by(customer_type) %>%
  summarise(min_length = min(ride_length), max_length = max(ride_length),
            median_length = median(ride_length), mean_length = mean(ride_length))
```

```
## # A tibble: 2 × 5
##   customer_type min_length max_length median_length mean_length
##   <chr>              <dbl>      <dbl>         <dbl>       <dbl>
## 1 casual                 1      1439.          16.6        27.8
## 2 member                 1      1434.          10.1        13.9
```

```r
#look at ride lengths broken down by day of week and customer type
aggregate(bike_data1$ride_length ~ bike_data1$customer_type + bike_data1$weekday, FUN = median)
```

```
##    bike_data1$customer_type bike_data1$weekday bike_data1$ride_length
## 1                    casual             Monday              16.616667
## 2                    member             Monday               9.666667
## 3                    casual            Tuesday              14.850000
## 4                    member            Tuesday               9.533333
## 5                    casual          Wednesday              14.500000
## 6                    member          Wednesday               9.650000
## 7                    casual           Thursday              14.333333
## 8                    member           Thursday               9.566667
## 9                    casual             Friday              15.516667
## 10                   member             Friday               9.900000
## 11                   casual           Saturday              18.450000
## 12                   member           Saturday              11.316667
## 13                   casual             Sunday              19.300000
## 14                        member                Sunday                 11.350000
```

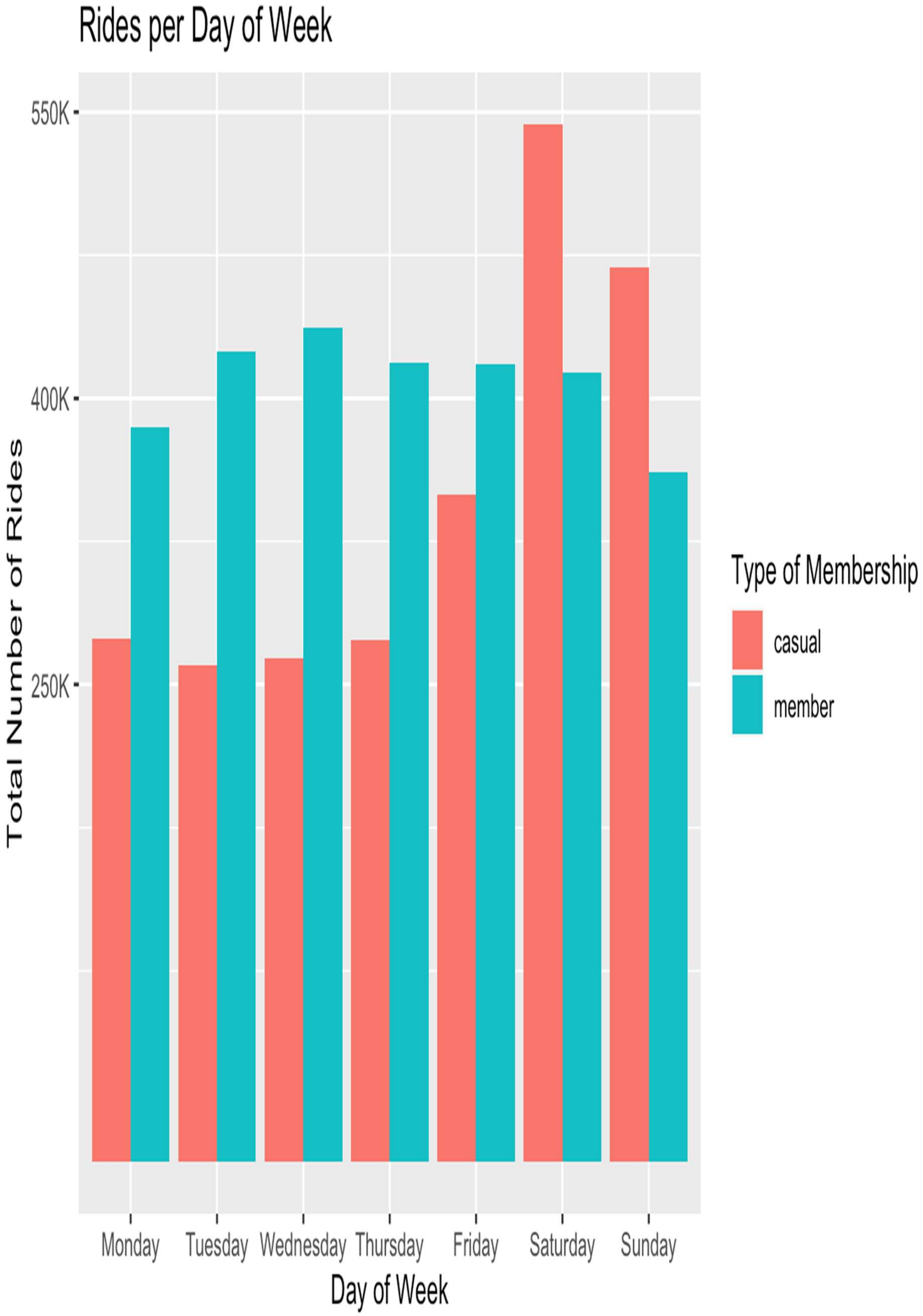
```
##look at total number of rides and averages based on day of week and customer type
bike_data1 %>%
  group_by(customer_type, weekday) %>%
  summarise(total_rides = n(), avg_ride = mean(ride_length)) %>%
  arrange(weekday)
```

```
## `summarise()` has grouped output by 'customer_type'. You can override using the `.groups` argument.
## # A tibble: 14 × 4
## # Groups:   customer_type [2]
##    customer_type weekday   total_rides avg_ride
##    <chr>         <ord>           <int>    <dbl>
##  1 casual        Monday         274179     28.2
##  2 member        Monday         384887     13.5
##  3 casual        Tuesday        260389     25.4
##  4 member        Tuesday        424811     13.1
##  5 casual        Wednesday      263521     24.2
##  6 member        Wednesday      437336     13.2
##  7 casual        Thursday       273325     24.0
##  8 member        Thursday       418922     13.1
##  9 casual        Friday         349734     25.8
## 10 member        Friday         418158     13.6
## 11 casual        Saturday       543678     30.2
## 12 member        Saturday       413470     15.5
## 13 casual        Sunday         468961     31.9
## 14 member        Sunday         361578     15.9
```

Data findings with visualizations

```
bike_data1 %>%    #total rides broken down by weekday
  group_by(customer_type, weekday) %>%
  summarise(number_of_rides = n() ) %>%
  arrange(customer_type, weekday) %>%
  ggplot(aes(x = weekday, y = number_of_rides, fill = customer_type)) + geom_col(position = "dodge") +
  labs(x= 'Day of Week', y='Total Number of Rides', title='Rides per Day of Week', fill = 'Type of Membership') +
  scale_y_continuous(breaks = c(250000, 400000, 550000), labels = c("250K",
"400K", "550K"))
```

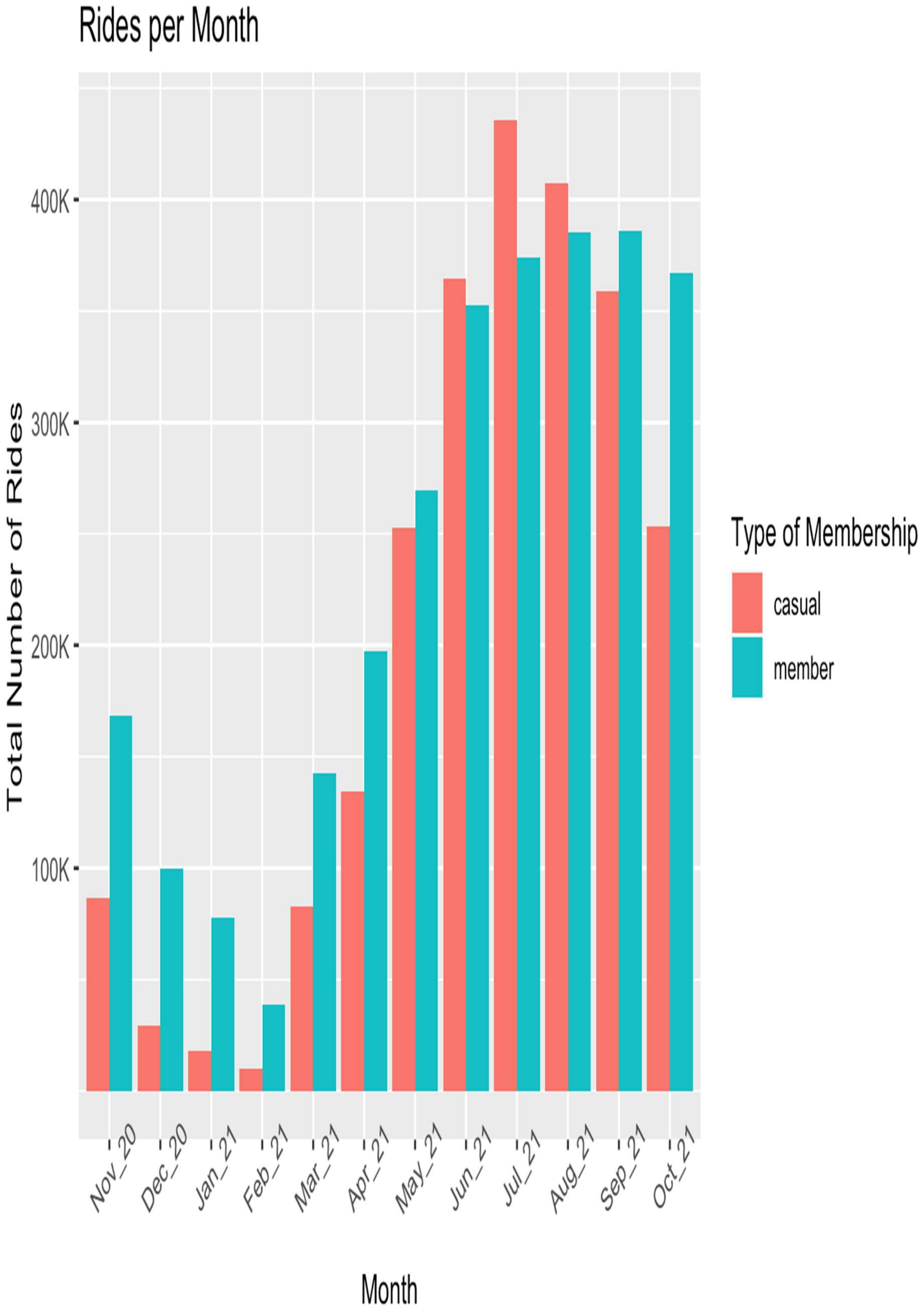
```
## `summarise()` has grouped output by 'customer_type'. You can override us
ing the `.groups` argument.
```



The rides per day of week show casual riders peak on the Saturday and Sunday while members peak Monday through Friday. This indicates members mainly use the bikes for their commutes and not leisure.

```r
bike_data1 %>%   #total rides broken down by month
  group_by(customer_type, month) %>%
  summarise(total_rides = n(),`average_duration_(mins)` = mean(ride_length)) %>%
  arrange(customer_type) %>%
  ggplot(aes(x=month, y=total_rides, fill = customer_type)) + 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), labels = c("100K", "200K", "300K", "400K")) + theme(axis.text.x = element_text(angle = 45))
```

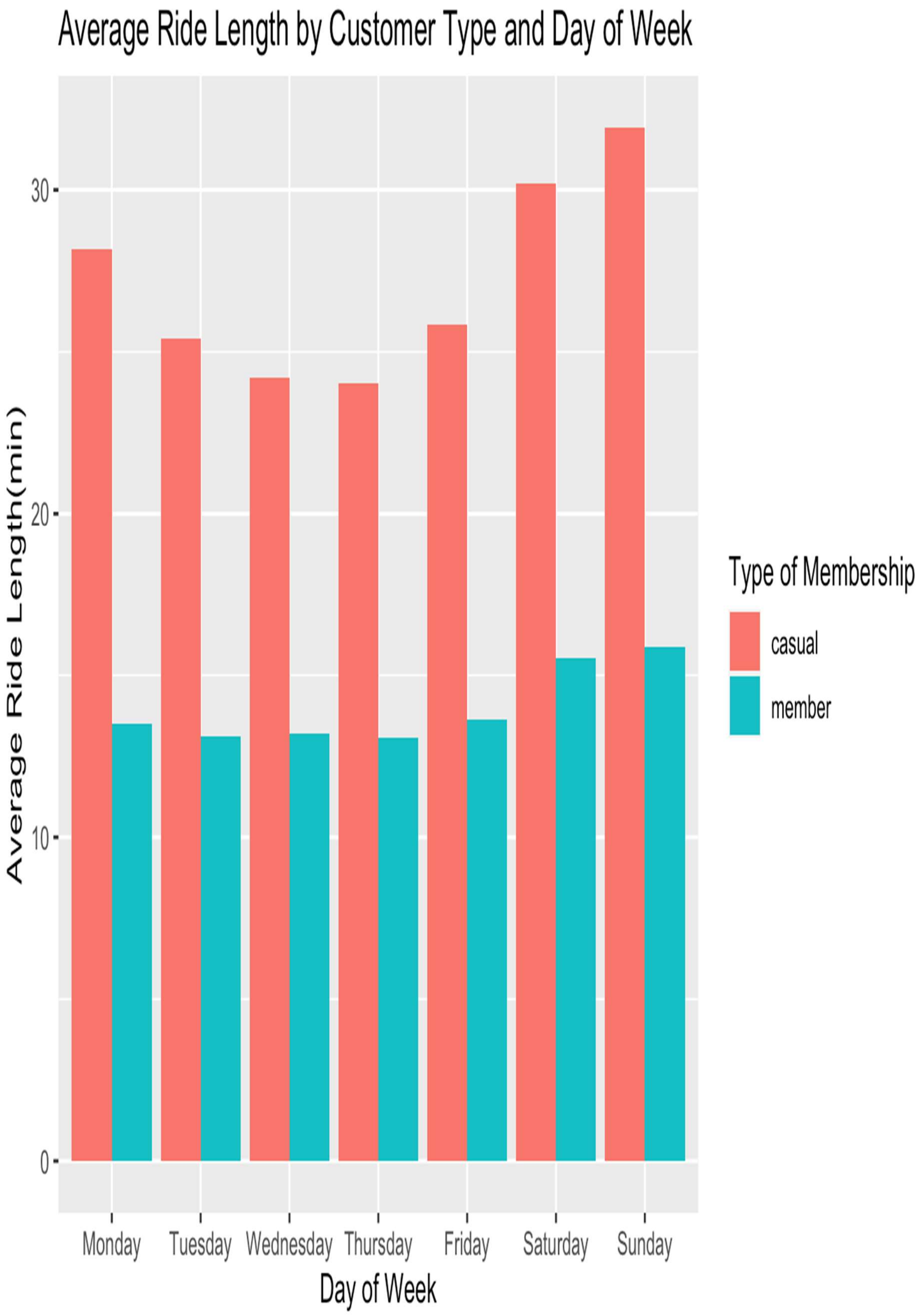
```
## `summarise()` has grouped output by 'customer_type'. You can override using the `.groups` argument.
```



The summer months bring more riders in total but casual rider usage is nearly nonexistent in the winter months. There are multiple factors which contribute to this result but annual members still use the service at a good rate in those months.

```r
bike_data1 %>%   #Average length by Customer Type and Day of Week
  group_by(customer_type, weekday) %>%
  summarise(average_ride_length = mean(ride_length)) %>%
  ggplot(aes(x=weekday, y = average_ride_length, fill = customer_type))+
  geom_col(position = "dodge") + labs (x="Day of Week", y="Average Ride Length(min)",
                                      title = "Average Ride Length by Customer Type and Day of Week",
                                      fill = "Type of Membership")
```

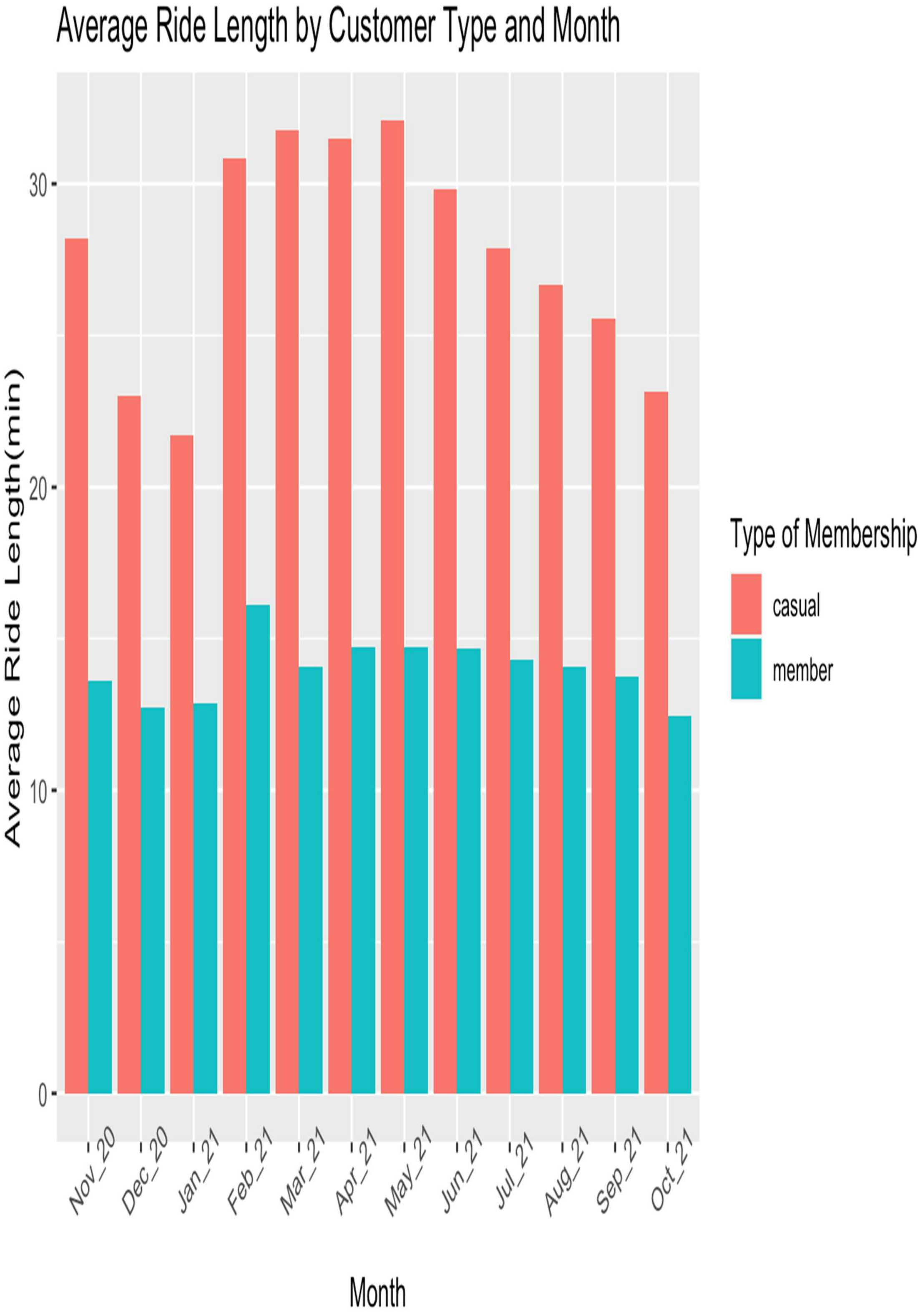
```
## `summarise()` has grouped output by 'customer_type'. You can override using the `.groups` argument.
```



The average ride length of casual riders is considerably longer than those of members and peak on Saturday and Sunday. Annual members bike a near constant length regardless of day of week.

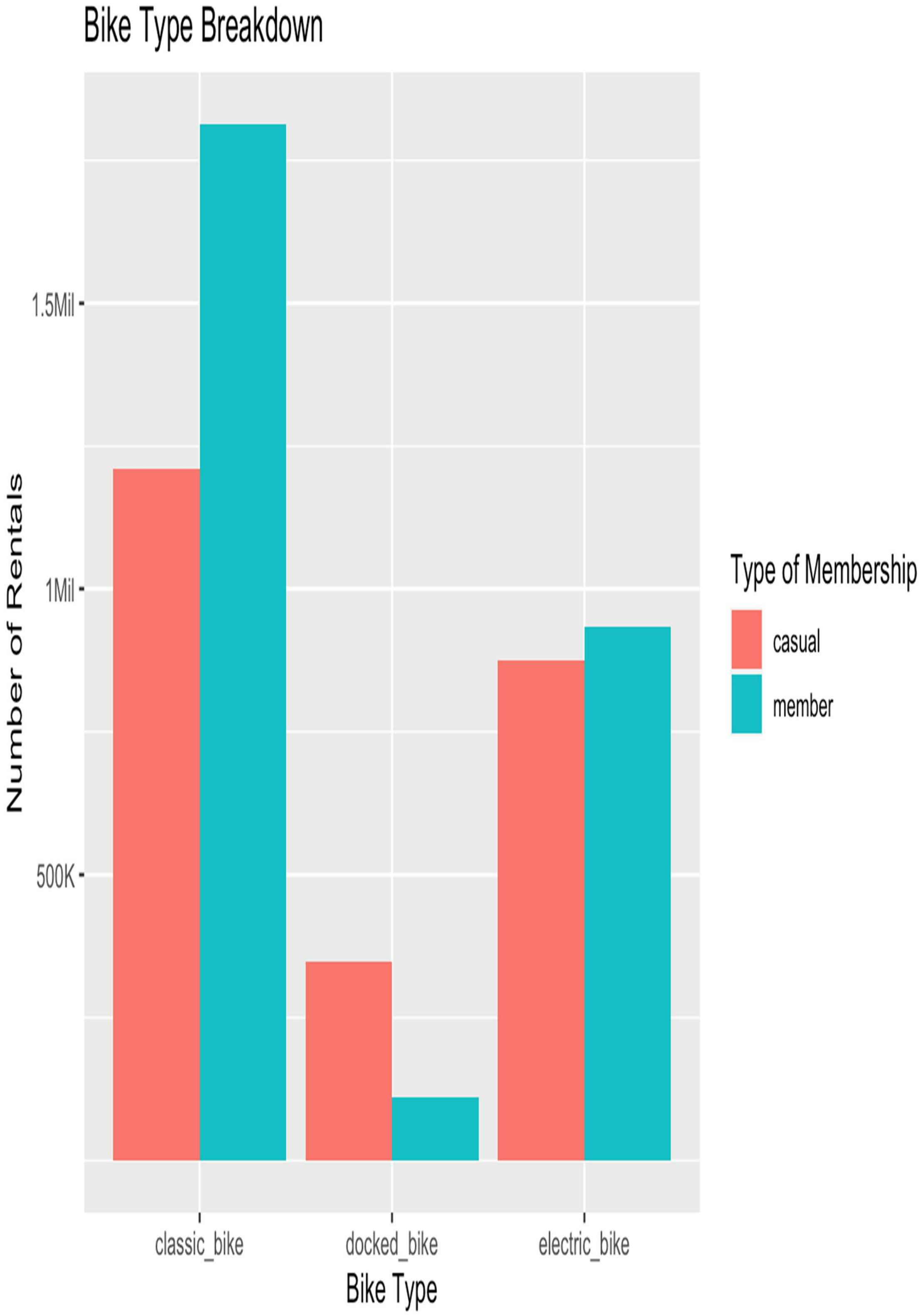
```r
bike_data1 %>%  #Average ride length by customer type and month
  group_by(customer_type, month) %>%
  summarise(average_ride_length = mean(ride_length)) %>%
  ggplot(aes(x=month, y = average_ride_length, fill = customer_type))+
  geom_col(position = "dodge") +
  labs (x="Month", y = "Average Ride Length(min)", title = "Average Ride Length by Customer Type and Month",
        fill = "Type of Membership") + theme(axis.text.x = element_text(angle = 45))
```

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



The average ride length by casual riders is still considerably longer than members even broken by month. While this would confirms the average ride length by day of week, it nearly contradicts the rides per month.

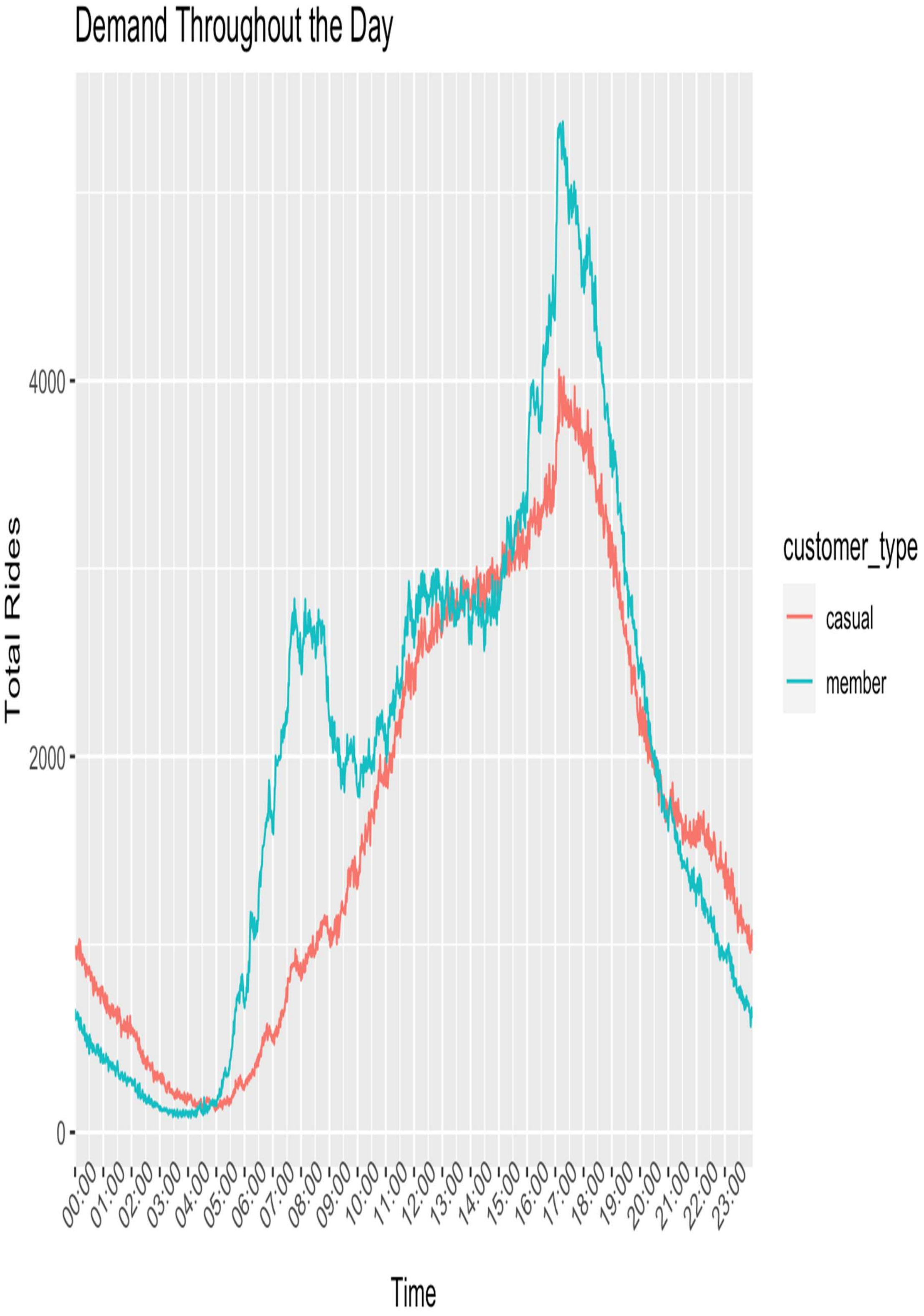
```r
bike_data1 %>%    #looking at breakdown of bike types rented
  ggplot(aes(x = bike_type, fill = customer_type)) + geom_bar(position = "dodge") +
  labs(x= 'Bike Type', y='Number of Rentals', title='Bike Type Breakdown', fill = 'Type of Membership') +
   scale_y_continuous(breaks = c(500000, 1000000, 1500000), labels = c("500K", "1Mil", "1.5Mil"))
```



The bike type breakdown shows members use classic bikes much more than casual members. The electric bike use is nearly identical but casual riders are more willing to use docked bikes.

```r
bike_data1 %>%     #Looking at demand over a 24 hour day
  group_by(customer_type, time) %>%
  summarise(total_rides = n()) %>%
  ggplot(aes(x=time, y=total_rides, color = customer_type, group = customer_type)) +
  geom_line() + scale_x_datetime(date_breaks = "1 hour",
                                 date_labels = "%H:%M", expand = c(0,0)) +
  theme(axis.text.x = element_text(angle = 45)) +
  labs(title ="Demand Throughout the Day", x = "Time", y = "Total Rides")
```

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



The bike demand in a 24 hour span shows that usage by annual members peak during rush hour to indicate many use the bikes for commutes to and from work especially with the steep drop after the peak at 5pm. Casual riders are not as volatile as there is a steady increase throughout the day with a steady decrease after the peak at 5pm.

Key takeaways

- Casual riders ride nearly 50% longer than members on average
- Annual members mainly use the bikes for their commutes as their usage peaks on weekdays during rush hour
- Casual riders use the bikes more for leisure based on the peak usage in summer months and weekends
- Casual riders do not use the service during the winter months as much as annual members
- Annual members mainly use classic bikes and rarely use docked bikes but casual riders are more open to riding all kinds of bikes