Case study: How does a bike-share navigate speedy success?

Md. Dider Hossain

2025-06-10

Scenario

As an acting junior data analyst working on the marketing analyst team at 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, my team wants to understand how casual riders and annual members use Cyclistic bikes dierently. From these insights, my team will design a new marketing strategy to convert casual riders into annual members. But first, Cyclistic executives must approve our recommendations, so they must be backed up with compelling data insights and professional data visualizations.

Characters and teams

- **Cyclistic:** A bike-share program that features more than 5,800 bicycles and 600 docking stations. Cyclistic sets itself apart by also offering reclining bikes, hand tricycles, and cargo bikes, making bike-share more inclusive to people with disabilities and riders who can't use a standard two-wheeled bike. The majority of riders opt for traditional bikes; about 8% of riders use the assistive options. Cyclistic users are more likely to ride for leisure, but about 30% use the bikes to commute to work each day.
- **Lily Moreno:** The director of marketing. Moreno is responsible for the development of campaigns and initiatives to promote the bike-share program. These may include email, social media, and other channels.
- **Cyclistic marketing analytics team:** A team of data analysts who are responsible for collecting, analyzing, and reporting data that helps guide Cyclistic marketing strategy.
- **Cyclistic executive team:** The notoriously detail-oriented executive team will decide whether to approve the recommended marketing program.

Ask phase

Three questions will guide the future marketing program:

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 inuence casual riders to become members?

Problem

Cyclistic aims to increase profitability by converting casual riders (single/day-pass users) into annual members, who generate higher recurring revenue. However, the marketing team lacks data-driven insights into how these user groups differ in their bike usage patterns.

Business Task

Analyze Cyclistic's historical trip data to:

1. Compare usage patterns between casual riders and annual members (e.g., ride duration, frequency, time/day preferences).
2. Identify key behavioral differences that could inform targeted marketing strategies.
3. Provide actionable recommendations to convert casual riders into members.

Prepare and process phase

Cyclistic's trip data is stored in two separate CSV files, one containing 2019 data and another containing 2020 data. These datasets were accessed through the **Google Data Analytics Course**. Both files include information such as ride IDs, bike types, start and end times, station names, and user types (casual or member).

The files are organized in a structured project directory with separate folders for raw data, cleaned data, and documentation to ensure organization and reproducibility. The goal is to combine these files into a single dataset for analysis. During preprocessing, key calculated fields like ride_length (trip duration) and day_of_week were added.

The data is sourced directly from **Motivate International Inc.**, the operator of Chicago's Divvy bikes. As the original, unmodified source, this data hasn't been aggregated or altered by third parties. The combined dataset contains more than 725,000 observations across 13 variables. It is used under Motivate's public **license**, which notes commercial use restrictions. No personally identifiable information (PII) such as credit card numbers is included, allowing the analysis to focus on aggregate trends.

Data integrity was ensured through the following steps:

- Checking for duplicate ride_id entries
- Validating timestamps (ensuring started_at < ended_at)
- Removing test rides (where ride_length   0)

R Code

```r
library(tidyverse)  #helps wrangle data
# Use the conflicted package to manage conflicts
library(conflicted)
```

```r
# Set dplyr::filter and dplyr::lag as the default choices
conflict_prefer("filter", "dplyr")
conflict_prefer("lag", "dplyr")
```

```r
# # Upload Divvy datasets (csv files)
q1_2019 <- read_csv("Divvy_Trips_2019_Q1.csv")
q1_2020 <- read_csv("Divvy_Trips_2020_Q1.csv")
```

Wrangle data and combine into a single file

- Compare column names each of the files
- While the names don't have to be in the same order, they DO need to match perfectly before we can use a command to join them into one file

```
colnames(q1_2019)
```

```
##  [1] "trip_id"           "start_time"
##  [3] "end_time"          "bikeid"
##  [5] "tripduration"      "from_station_id"
##  [7] "from_station_name" "to_station_id"
##  [9] "to_station_name"   "usertype"
## [11] "gender"            "birthyear"
```

```
colnames(q1_2020)
```

```
##  [1] "ride_id"            "rideable_type"
##  [3] "started_at"         "ended_at"
##  [5] "start_station_name" "start_station_id"
##  [7] "end_station_name"   "end_station_id"
##  [9] "start_lat"          "start_lng"
## [11] "end_lat"            "end_lng"
## [13] "member_casual"
```

- Rename columns to make them consistent with q1_2020 (as this will be the supposed going-forward table design for Divvy)

```
(q1_2019 <- rename(q1_2019
                   ,ride_id = trip_id
                   ,rideable_type = bikeid
                   ,started_at = start_time
                   ,ended_at = end_time
                   ,start_station_name = from_station_name
                   ,start_station_id = from_station_id
                   ,end_station_name = to_station_name
                   ,end_station_id = to_station_id
                   ,member_casual = usertype
))
```

```
## # A tibble: 365,069 x 12
##     ride_id started_at   ended_at rideable_type tripduration
##       <dbl> <chr>        <chr>            <dbl>        <dbl>
##  1 21742443 2019-01-01 ~ 2019-01~          2167          390
##  2 21742444 2019-01-01 ~ 2019-01~          4386          441
##  3 21742445 2019-01-01 ~ 2019-01~          1524          829
##  4 21742446 2019-01-01 ~ 2019-01~           252         1783
##  5 21742447 2019-01-01 ~ 2019-01~          1170          364
##  6 21742448 2019-01-01 ~ 2019-01~          2437          216
##  7 21742449 2019-01-01 ~ 2019-01~          2708          177
##  8 21742450 2019-01-01 ~ 2019-01~          2796          100
##  9 21742451 2019-01-01 ~ 2019-01~          6205         1727
## 10 21742452 2019-01-01 ~ 2019-01~          3939          336
## # i 365,059 more rows
## # i 7 more variables: start_station_id <dbl>,
## #   start_station_name <chr>, end_station_id <dbl>,
## #   end_station_name <chr>, member_casual <chr>,
## #   gender <chr>, birthyear <dbl>
```

```r
# Inspect the dataframes and look for incongruencies
str(q1_2019)
```

```
## spc_tbl_ [365,069 x 12] (S3: spec_tbl_df/tbl_df/tbl/data.frame)
##  $ ride_id           : num [1:365069] 21742443 21742444 21742445 21742446 21742447 ...
##  $ started_at        : chr [1:365069] "2019-01-01 0:04:37" "2019-01-01 0:08:13" "2019-01-01 0:13:23"
##  $ ended_at          : chr [1:365069] "2019-01-01 0:11:07" "2019-01-01 0:15:34" "2019-01-01 0:27:12"
##  $ rideable_type     : num [1:365069] 2167 4386 1524 252 1170 ...
##  $ tripduration      : num [1:365069] 390 441 829 1783 364 ...
##  $ start_station_id  : num [1:365069] 199 44 15 123 173 98 98 211 150 268 ...
##  $ start_station_name: chr [1:365069] "Wabash Ave & Grand Ave" "State St & Randolph St" "Racine Ave &
##  $ end_station_id    : num [1:365069] 84 624 644 176 35 49 49 142 148 141 ...
##  $ end_station_name  : chr [1:365069] "Milwaukee Ave & Grand Ave" "Dearborn St & Van Buren St (*)" "
##  $ member_casual     : chr [1:365069] "Subscriber" "Subscriber" "Subscriber" "Subscriber" ...
##  $ gender            : chr [1:365069] "Male" "Female" "Female" "Male" ...
##  $ birthyear         : num [1:365069] 1989 1990 1994 1993 1994 ...
##  - attr(*, "spec")=
##   .. cols(
##   ..   trip_id = col_double(),
##   ..   start_time = col_character(),
##   ..   end_time = col_character(),
##   ..   bikeid = col_double(),
##   ..   tripduration = col_number(),
##   ..   from_station_id = col_double(),
##   ..   from_station_name = col_character(),
##   ..   to_station_id = col_double(),
##   ..   to_station_name = col_character(),
##   ..   usertype = col_character(),
##   ..   gender = col_character(),
##   ..   birthyear = col_double()
##   .. )
##  - attr(*, "problems")=<externalptr>
```

```r
str(q1_2020)
```

```
## spc_tbl_ [426,887 x 13] (S3: spec_tbl_df/tbl_df/tbl/data.frame)
##  $ ride_id           : chr [1:426887] "EACB19130B0CDA4A" "8FED874C809DC021" "789F3C21E472CA96" "C9A3
##  $ rideable_type     : chr [1:426887] "docked_bike" "docked_bike" "docked_bike" "docked_bike" ...
##  $ started_at        : chr [1:426887] "2020-01-21 20:06:59" "2020-01-30 14:22:39" "2020-01-09 19:29:
##  $ ended_at          : chr [1:426887] "2020-01-21 20:14:30" "2020-01-30 14:26:22" "2020-01-09 19:32:
##  $ start_station_name: chr [1:426887] "Western Ave & Leland Ave" "Clark St & Montrose Ave" "Broadway
##  $ start_station_id  : num [1:426887] 239 234 296 51 66 212 96 96 212 38 ...
##  $ end_station_name  : chr [1:426887] "Clark St & Leland Ave" "Southport Ave & Irving Park Rd" "Wilto
##  $ end_station_id    : num [1:426887] 326 318 117 24 212 96 212 212 96 100 ...
##  $ start_lat         : num [1:426887] 42 42 41.9 41.9 41.9 ...
##  $ start_lng         : num [1:426887] -87.7 -87.7 -87.6 -87.6 -87.6 ...
##  $ end_lat           : num [1:426887] 42 42 41.9 41.9 41.9 ...
##  $ end_lng           : num [1:426887] -87.7 -87.7 -87.7 -87.6 -87.6 ...
##  $ member_casual     : chr [1:426887] "member" "member" "member" "member" ...
##  - attr(*, "spec")=
##   .. cols(
##   ..   ride_id = col_character(),
##   ..   rideable_type = col_character(),
##   ..   started_at = col_character(),
##   ..   ended_at = col_character(),
##   ..   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>
```

```r
# Convert ride_id and rideable_type to character so that they can stack correctly
q1_2019 <-  mutate(q1_2019, ride_id = as.character(ride_id)
                   ,rideable_type = as.character(rideable_type))
```

```r
# Stack individual quarter's data frames into one big data frame
all_trips <- bind_rows(q1_2019, q1_2020)#, q3_2019)#, q4_2019, q1_2020)
```

```r
# Remove lat, long, birthyear, and gender fields as this data was dropped beginning in 2020
all_trips <- all_trips %>%
  select(-c(start_lat, start_lng, end_lat, end_lng, birthyear, gender,  "tripduration"))
```

Clean up data and add data to prepare for analysis

```r
# Inspect the new table that has been created
colnames(all_trips)  #List of column names
```

```
## [1] "ride_id"            "started_at"
## [3] "ended_at"           "rideable_type"
## [5] "start_station_id"   "start_station_name"
## [7] "end_station_id"     "end_station_name"
## [9] "member_casual"
```

```r
nrow(all_trips)  #How many rows are in data frame?
```

```
## [1] 791956
```

```r
dim(all_trips)  #Dimensions of the data frame?
```

```
## [1] 791956      9
```

```r
head(all_trips)  #See the first 6 rows of data frame.  Also tail(all_trips)
```

```
## # A tibble: 6 x 9
##   ride_id started_at ended_at rideable_type start_station_id
##   <chr>   <chr>      <chr>    <chr>                    <dbl>
## 1 217424~ 2019-01-0~ 2019-01~ 2167                       199
## 2 217424~ 2019-01-0~ 2019-01~ 4386                        44
## 3 217424~ 2019-01-0~ 2019-01~ 1524                        15
## 4 217424~ 2019-01-0~ 2019-01~ 252                        123
## 5 217424~ 2019-01-0~ 2019-01~ 1170                       173
## 6 217424~ 2019-01-0~ 2019-01~ 2437                        98
## # i 4 more variables: start_station_name <chr>,
## #   end_station_id <dbl>, end_station_name <chr>,
## #   member_casual <chr>
```

```r
str(all_trips)  #See list of columns and data types (numeric, character, etc)
```

```
## tibble [791,956 x 9] (S3: tbl_df/tbl/data.frame)
##  $ ride_id           : chr [1:791956] "21742443" "21742444" "21742445" "21742446" ...
##  $ started_at        : chr [1:791956] "2019-01-01 0:04:37" "2019-01-01 0:08:13" "2019-01-01 0:13:23"
##  $ ended_at          : chr [1:791956] "2019-01-01 0:11:07" "2019-01-01 0:15:34" "2019-01-01 0:27:12"
##  $ rideable_type     : chr [1:791956] "2167" "4386" "1524" "252" ...
##  $ start_station_id  : num [1:791956] 199 44 15 123 173 98 98 211 150 268 ...
##  $ start_station_name: chr [1:791956] "Wabash Ave & Grand Ave" "State St & Randolph St" "Racine Ave
##  $ end_station_id    : num [1:791956] 84 624 644 176 35 49 49 142 148 141 ...
##  $ end_station_name  : chr [1:791956] "Milwaukee Ave & Grand Ave" "Dearborn St & Van Buren St (*)" "
##  $ member_casual     : chr [1:791956] "Subscriber" "Subscriber" "Subscriber" "Subscriber" ...
```

```r
summary(all_trips)  #Statistical summary of data. Mainly for numerics
```

```
##    ride_id           started_at          ended_at
##  Length:791956      Length:791956      Length:791956
##  Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character
##
##
##
##
##  rideable_type      start_station_id start_station_name
##  Length:791956      Min.   :  2.0    Length:791956
##  Class :character   1st Qu.: 77.0    Class :character
##  Mode  :character   Median :174.0    Mode  :character
##                     Mean   :204.4
##                     3rd Qu.:291.0
##                     Max.   :675.0
##
##  end_station_id  end_station_name   member_casual
##  Min.   :  2.0   Length:791956      Length:791956     
##  1st Qu.: 77.0   Class :character   Class :character  
##  Median :174.0   Mode  :character   Mode  :character  
##  Mean   :204.4                                        
##  3rd Qu.:291.0                                        
##  Max.   :675.0                                        
##  NA's   :1
```

There are a few problems we will need to fix:

1. In the "member_casual" column, there are two names for members ("member" and "Subscriber") and two names for casual riders ("Customer" and "casual"). We will need to consolidate that from four to two labels.
2. The data can only be aggregated at the ride-level, which is too granular. We will want to add some additional columns of data – such as day, month, year – that provide additional opportunities to aggregate the data.
3. We will want to add a calculated field for length of ride since the 2020Q1 data did not have the "tripduration" column. We will add "ride_length" to the entire dataframe for consistency.
4. There are some rides where tripduration shows up as negative, including several hundred rides where Divvy took bikes out of circulation for Quality Control reasons. We will want to delete these rides.

- In the "member_casual" column, replace "Subscriber" with "member" and "Customer" with "casual"
- Before 2020, Divvy used different labels for these two types of riders. We will want to make our dataframe consistent with their current nomenclature
- Begin by seeing how many observations fall under each usertype

```r
table(all_trips$member_casual)
```

```
## 
##     casual   Customer     member Subscriber 
##      48480      23163     378407     341906
```

```r
# Reassign to the desired values (we will go with the current 2020 labels)
all_trips <-  all_trips %>%
  mutate(member_casual = recode(member_casual
                                ,"Subscriber" = "member"
                                ,"Customer" = "casual"))
```

```r
# Check to make sure the proper number of observations were reassigned
table(all_trips$member_casual)
```

```
## 
## casual member 
##  71643 720313
```

- Add columns that list the date, month, day, and year of each ride
- This will allow us to aggregate ride data for each month, day, or year … before completing these operations we could only aggregate at the ride level
- (https://www.statmethods.net/input/dates.html) more on date formats in R found at that link

```r
all_trips$date <- as.Date(all_trips$started_at) #The default format is yyyy-mm-dd
all_trips$month <- format(as.Date(all_trips$date), "%m")
all_trips$day <- format(as.Date(all_trips$date), "%d")
all_trips$year <- format(as.Date(all_trips$date), "%Y")
all_trips$day_of_week <- format(as.Date(all_trips$date), "%A")
```

```r
# Add a "ride_length" calculation to all_trips (in seconds)
# https://stat.ethz.ch/R-manual/R-devel/library/base/html/difftime.html
all_trips$ride_length <- difftime(all_trips$ended_at,all_trips$started_at)
```

```r
# Inspect the structure of the columns
str(all_trips)
```

```
## tibble [791,956 x 15] (S3: tbl_df/tbl/data.frame)
##  $ ride_id           : chr [1:791956] "21742443" "21742444" "21742445" "21742446" ...
##  $ started_at        : chr [1:791956] "2019-01-01 0:04:37" "2019-01-01 0:08:13" "2019-01-01 0:13:23"
##  $ ended_at          : chr [1:791956] "2019-01-01 0:11:07" "2019-01-01 0:15:34" "2019-01-01 0:27:12"
##  $ rideable_type     : chr [1:791956] "2167" "4386" "1524" "252" ...
##  $ start_station_id  : num [1:791956] 199 44 15 123 173 98 98 211 150 268 ...
##  $ start_station_name: chr [1:791956] "Wabash Ave & Grand Ave" "State St & Randolph St" "Racine Ave &
##  $ end_station_id    : num [1:791956] 84 624 644 176 35 49 49 142 148 141 ...
##  $ end_station_name  : chr [1:791956] "Milwaukee Ave & Grand Ave" "Dearborn St & Van Buren St (*)" "W
##  $ member_casual     : chr [1:791956] "member" "member" "member" "member" ...
##  $ date              : Date[1:791956], format: "2019-01-01" ...
##  $ month             : chr [1:791956] "01" "01" "01" "01" ...
##  $ day               : chr [1:791956] "01" "01" "01" "01" ...
##  $ year              : chr [1:791956] "2019" "2019" "2019" "2019" ...
##  $ day_of_week       : chr [1:791956] "Tuesday" "Tuesday" "Tuesday" "Tuesday" ...
##  $ ride_length       : 'difftime' num [1:791956] 0 0 0 0 ...
##   ..- attr(*, "units")= chr "secs"
```

```r
# Convert "ride_length" from Factor to numeric so we can run calculations on the data
is.factor(all_trips$ride_length)
```

```
## [1] FALSE
```

```r
all_trips$ride_length <- as.numeric(as.character(all_trips$ride_length))
is.numeric(all_trips$ride_length)
```

```
## [1] TRUE
```

- Remove "bad" data
- The dataframe includes a few hundred entries when bikes were taken out of docks and checked for quality by Divvy or ride_length was negative
- We will create a new version of the dataframe (v2) since data is being removed
- Check this link (https://www.datasciencemadesimple.com/delete-or-drop-rows-in-r-with-conditions-2/)

```
all_trips_v2 <- all_trips[!(all_trips$start_station_name == "HQ QR" | all_trips$ride_length<0),]
```

Conduct descriptive analysis

```
# Descriptive analysis on ride_length (all figures in seconds)
mean(all_trips_v2$ride_length) #straight average (total ride length / rides)
```

```
## [1] 532.5352
```

```
median(all_trips_v2$ride_length) #midpoint number in the ascending array of ride lengths
```

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

```
max(all_trips_v2$ride_length) #longest ride
```

```
## [1] 10623600
```

```
min(all_trips_v2$ride_length) #shortest ride
```

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

```
# Or by summary function
summary(all_trips_v2$ride_length)
```

```
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
##        0        0        0      533        0 10623600
```

```
# Compare members and casual users
aggregate(all_trips_v2$ride_length ~ all_trips_v2$member_casual, FUN = mean)
```

```
##   all_trips_v2$member_casual all_trips_v2$ride_length
## 1                     casual                3859.9879
## 2                     member                 218.9801
```

```
aggregate(all_trips_v2$ride_length ~ all_trips_v2$member_casual, FUN = median)
```

```
##   all_trips_v2$member_casual all_trips_v2$ride_length
## 1                     casual                        0
## 2                     member                        0
```

```
aggregate(all_trips_v2$ride_length ~ all_trips_v2$member_casual, FUN = max)
```

```
##   all_trips_v2$member_casual all_trips_v2$ride_length
## 1                     casual                 10623600
## 2                     member                  6130800
```

```r
aggregate(all_trips_v2$ride_length ~ all_trips_v2$member_casual, FUN = min)
```

```
##   all_trips_v2$member_casual all_trips_v2$ride_length
## 1                     casual                        0
## 2                     member                        0
```

```r
# See the average ride time by each day for members vs casual users
aggregate(all_trips_v2$ride_length ~ all_trips_v2$member_casual + all_trips_v2$day_of_week, FUN = mean)
```

```
##    all_trips_v2$member_casual all_trips_v2$day_of_week
## 1                      casual                   Friday
## 2                      member                   Friday
## 3                      casual                   Monday
## 4                      member                   Monday
## 5                      casual                 Saturday
## 6                      member                 Saturday
## 7                      casual                   Sunday
## 8                      member                   Sunday
## 9                      casual                 Thursday
## 10                     member                 Thursday
## 11                     casual                  Tuesday
## 12                     member                  Tuesday
## 13                     casual                Wednesday
## 14                     member                Wednesday
##    all_trips_v2$ride_length
## 1                 5076.3010
## 2                  248.3190
## 3                 3476.3727
## 4                  237.8158
## 5                 3499.2652
## 6                  475.4717
## 7                 3114.9689
## 8                  338.4886
## 9                 7165.2442
## 10                 135.1998
## 11                3201.6414
## 12                 174.8043
## 13                2864.5514
## 14                 122.6165
```

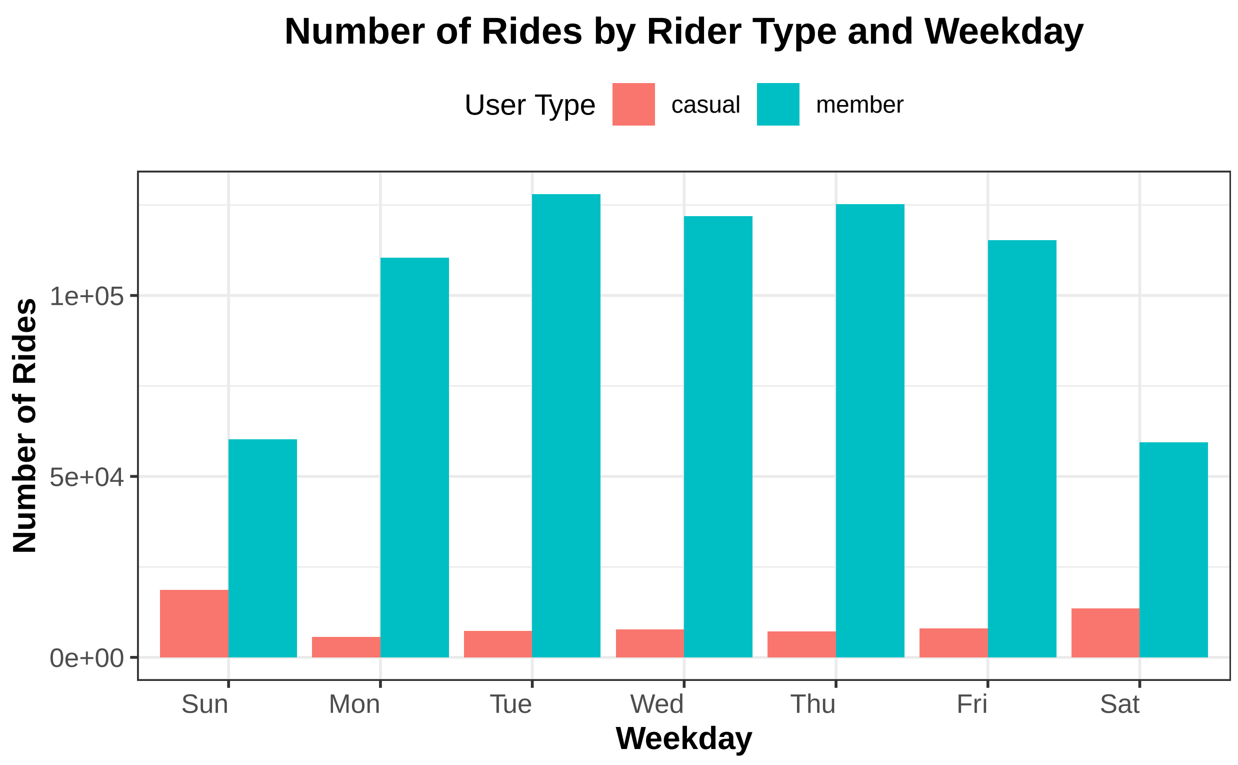
```r
# Notice that the days of the week are out of order. Let's fix that.
all_trips_v2$day_of_week <- ordered(all_trips_v2$day_of_week,
                                    levels=c("Sunday", "Monday", "Tuesday", "Wednesday", "Thursday",
                                             "Friday", "Saturday"))
# Now, let's run the average ride time by each day for members vs casual users
aggregate(all_trips_v2$ride_length ~ all_trips_v2$member_casual + all_trips_v2$day_of_week, FUN = mean)
```

```
##    all_trips_v2$member_casual all_trips_v2$day_of_week
## 1                      casual                   Sunday
## 2                      member                   Sunday
## 3                      casual                   Monday
## 4                      member                   Monday
## 5                      casual                 Tuesday
## 6                      member                 Tuesday
## 7                      casual               Wednesday
## 8                      member               Wednesday
## 9                      casual                Thursday
## 10                     member                Thursday
## 11                     casual                  Friday
## 12                     member                  Friday
## 13                     casual                Saturday
## 14                     member                Saturday
##    all_trips_v2$ride_length
## 1                3114.9689
## 2                 338.4886
## 3                3476.3727
## 4                 237.8158
## 5                3201.6414
## 6                 174.8043
## 7                2864.5514
## 8                 122.6165
## 9                7165.2442
## 10                135.1998
## 11               5076.3010
## 12                248.3190
## 13               3499.2652
## 14                475.4717
```

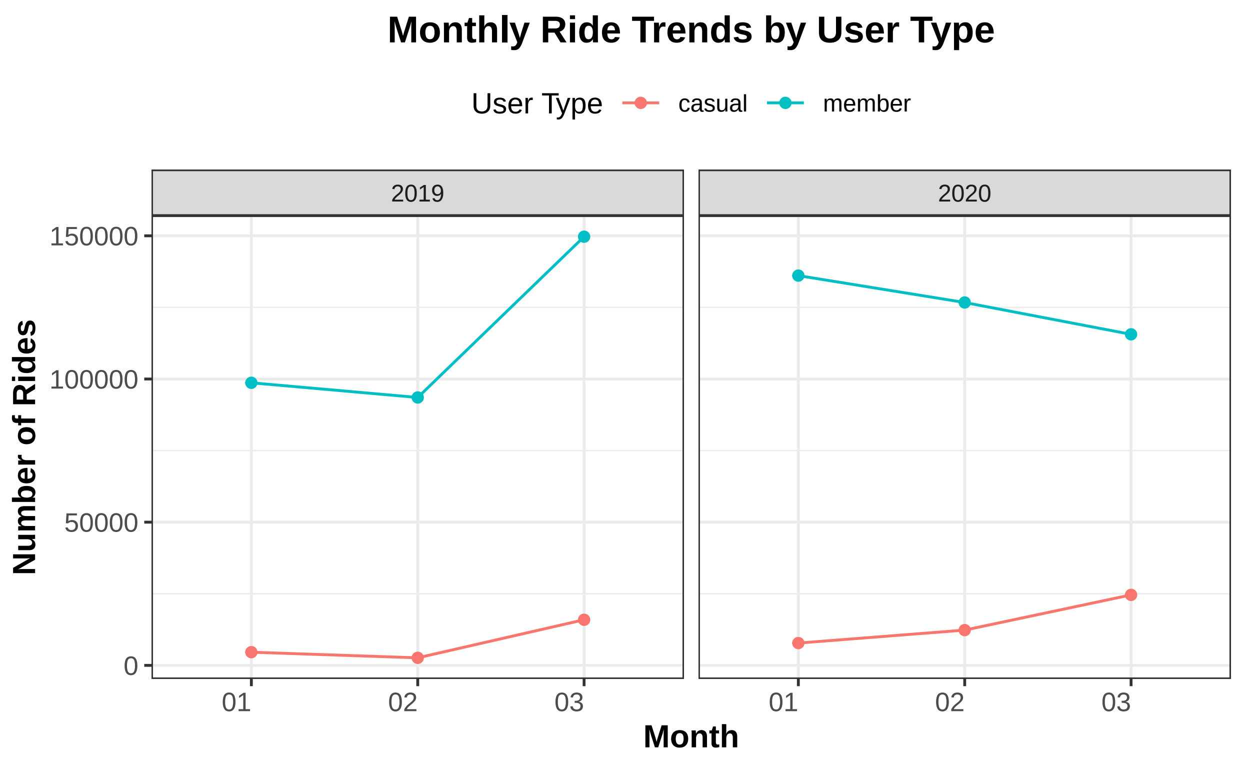
```r
# analyze ridership data by type and weekday
all_trips_v2 %>%
  mutate(weekday = wday(started_at, label = TRUE)) %>%  #creates weekday field using wday()
  group_by(member_casual, weekday) %>%  #groups by usertype and weekday
  summarise(number_of_rides = n()  #calculates the number of rides and average duration
          ,average_duration = mean(ride_length)) %>%  # calculates the average duration
  arrange(member_casual, weekday)
```

```
## # A tibble: 14 x 4
## # Groups:   member_casual [2]
##    member_casual weekday number_of_rides average_duration
##    <chr>         <ord>             <int>            <dbl>
##  1 casual        Sun               18652            3115.
##  2 casual        Mon                5591            3476.
##  3 casual        Tue                7311            3202.
##  4 casual        Wed                7690            2865.
##  5 casual        Thu                7147            7165.
##  6 casual        Fri                8013            5076.
##  7 casual        Sat               13473            3499.
##  8 member        Sun               60197             338.
##  9 member        Mon              110430             238.
## 10 member        Tue              127974             175.
## 11 member        Wed              121902             123.
## 12 member        Thu              125228             135.
## 13 member        Fri              115168             248.
## 14 member        Sat               59413             475.
```

```r
# Let's visualize the number of rides by rider type
all_trips_v2 %>%
  mutate(weekday = wday(started_at, label = TRUE)) %>%
  group_by(member_casual, weekday) %>%
  summarise(number_of_rides = n()
            ,average_duration = mean(ride_length)) %>%
  arrange(member_casual, weekday)  %>%
  ggplot(aes(x = weekday, y = number_of_rides, fill = member_casual)) +
  geom_col(position = "dodge") +
  labs(title = "Number of Rides by Rider Type and Weekday",
       x = "Weekday",
       y = "Number of Rides",
       fill = "User Type") +
  theme_bw() +
  theme(legend.position = "top",
        plot.title = element_text(hjust = 0.5, size = 14, face = "bold"),
        axis.text.x = element_text(hjust = 1, size = 10),
        axis.text.y = element_text(size = 10),
        plot.background = element_blank(),
        axis.title.x = element_text(size = 12, face = "bold"),
        axis.title.y = element_text(size = 12, face = "bold"))
```

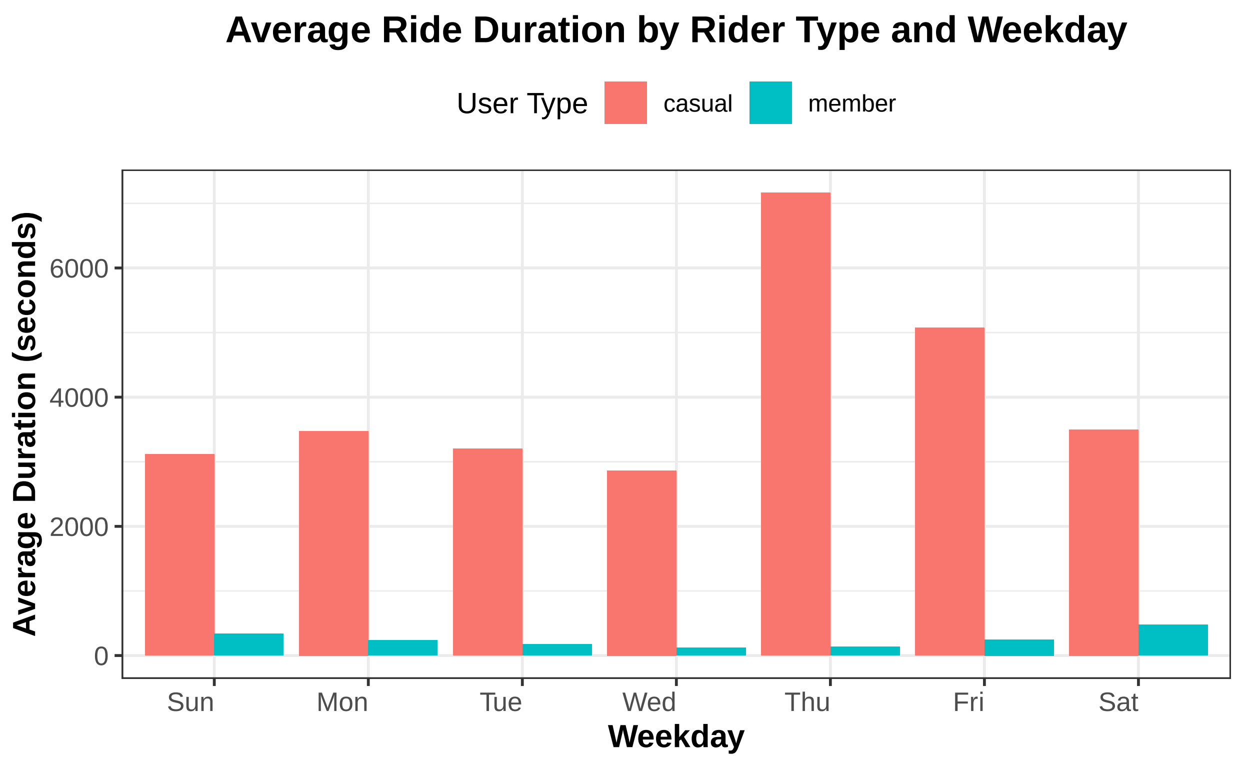


```r
all_trips_v2 %>%
  group_by(year, month, member_casual) %>%
  summarise(rides = n()) %>%
  ggplot(aes(x = month, y = rides, color = member_casual, group = member_casual)) +
  geom_line() +
  geom_point() +
  facet_wrap(~year) +
  theme_bw() +
  theme(legend.position = "top",
        plot.title = element_text(hjust = 0.5, size = 14, face = "bold"),
        axis.text.x = element_text(hjust = 1, size = 10),
        axis.text.y = element_text(size = 10),
        plot.background = element_blank(),
        axis.title.x = element_text(size = 12, face = "bold"),
        axis.title.y = element_text(size = 12, face = "bold")) +
  labs(title = "Monthly Ride Trends by User Type",
       x = "Month",
       y = "Number of Rides",
       color = "User Type")
```



```
# Let's create a visualization for average duration
all_trips_v2 %>%
  mutate(weekday = wday(started_at, label = TRUE)) %>%
  group_by(member_casual, weekday) %>%
  summarise(number_of_rides = n()
            ,average_duration = mean(ride_length)) %>%
  arrange(member_casual, weekday)  %>%
  ggplot(aes(x = weekday, y = average_duration, fill = member_casual)) +
  geom_col(position = "dodge") +
  labs(title = "Average Ride Duration by Rider Type and Weekday",
       x = "Weekday",
       y = "Average Duration (seconds)",
       fill = "User Type") +
  theme_bw() +
```

```r
  theme(legend.position = "top",
        plot.title = element_text(hjust = 0.5, size = 14, face = "bold"),
        axis.text.x = element_text(hjust = 1, size = 10),
        axis.text.y = element_text(size = 10),
        plot.background = element_blank(),
        axis.title.x = element_text(size = 12, face = "bold"),
        axis.title.y = element_text(size = 12, face = "bold"))
```



```r
t.test(ride_length ~ member_casual, data = all_trips_v2)
```

```
##
##  Welch Two Sample t-test
##
## data:  ride_length by member_casual
## t = 9.4545, df = 68232, p-value < 2.2e-16
## alternative hypothesis: true difference in means between group casual and group member is not equal t
## 95 percent confidence interval:
##  2886.193 4395.822
## sample estimates:
## mean in group casual mean in group member
##            3859.9879             218.9801
```

Conclusion

Casual riders take significantly longer trips, averaging 3,860 seconds per ride, compared to members, who average just 219 seconds. Interestingly, while casual riders show higher activity on Thursdays and Fridays, members are more active on weekends (Saturday and Sunday). However, the total number of rides is higher for members on weekdays, likely due to commuting patterns, whereas casual riders exhibit a noticeable summer peak, suggesting seasonal leisure use. A t-test ($p < 2.2e-16$) confirms that the difference in ride durations between the two groups is highly significant, reinforcing that casual riders consistently take longer trips on average. These insights highlight distinct usage behaviors—casual riders leaning toward leisure and members toward routine travel.

Recommendations

- Target casual riders who frequently use bikes on Thursdays and Fridays with weekday-specific discounts for membership sign-ups. Example: **Ride 3+ weekdays? Save 30% with an annual membership!**
- Capitalize on the summer peak by offering limited-time summer membership deals (**e.g., 3-month trial at a reduced rate**).
- Identify casual riders with longer trip durations and offer them bonus ride credits upon membership conversion.
- Deploy on-the-spot membership sign-up kiosks at high-traffic casual rider stations, especially near leisure hotspots.

References

- Methodology and dataset sourced from the **Google Data Analytics Professional Certificate** Capstone Project.
- Data provided by **Motivate International Inc.**, the operator of Chicago's Divvy bikes, under their public **license**.