Case Study: Cyclistic

Contents

Case Background 1

Problem Statement 1

Data Source and Organization 2

Preprocessing Coding Log 2

Transforming and cleaning the data 6

Conduct a descriptive analysis 9

Findings Summary: Inconclusive 12

Share Key Findings(ppt) 13

Act on Key Findings 13

Case Background

As a junior data analyst working in the marketing analyst team at Cyclistic (a fictional bike-sharing company active in Chicago), I am tasked with understanding how casual riders and annual members use Cyclistic bikes differently. Casual riders consist of customers that purchase single-ride or full-day passes, whereas annual members subscribe yearly for unlimited biking access. The marketing director theorizes that the company's future success depends on maximizing the number of yearly memberships by converting casual riders into annual members. Pending executive approval, my team will be designing a new marketing strategy that pursues this idea.

To inform any decision-making behind Cyclistic's new marketing strategy, the goal of this project will be to uncover and convey actionable insights.

If you would like to skip everything to view the results of this study, you can view my presentation here.

Problem Statement

Cyclistic is faced with an uncertain future and is no longer able to solely rely on its traditional marketing strategies of raising general awareness and appealing to a variety of customer needs. In the interest of company growth, the director of marketing believes that Cyclistic should capitalize on the lucrative profit margins of annual subscribers by marketing to existing casual customers and persuading them to become yearly subscribers. If that strategy is plausible, a well-executed marketing campaign may lead to more sustainable long-term revenue. To that end, we need to analyze how and why Cyclistic casual bikers and members differ to weigh any evidence, opportunities, and barriers to any future marketing strategy.

Data Source and Organization

The data we'll be using was extracted from here. This data is made available by Motivate International Inc. under this license. Note that Cyclistic is a fictional entity and Divvy's open data is used for the purpose of this case study.

The data available to us consists of a repository made up primarily of quantitative measurements collected over time. Each data point represents a single bike trip from one docking station to the next. At first glance, this data does not seem sufficient to fully comprehend how casuals and members use Cyclistic bikes differently. This data provides an overhead view of what they may be doing differently, but not the why.

We'll be examining the data of Quarter 1 of year 2019 and 2020. Each quarter has a separate comma-separated value file with the same headings. Each record consists of a bike trip under the bike-sharing program composed of several features: a unique hash ID serving as the table's primary key to identify each bike trip, the type of bike used, the type of customer (casual or member), details about the starting and ending docking station (name, ID, latitude, and longitude) and the DateTime for when the bike was picked up and dropped off.

Preprocessing Coding Log

Setup Libraries

```r
# helps wrangle data
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v readr     2.1.5
## v forcats   1.0.0     v stringr   1.5.1
## v ggplot2   3.5.1     v tibble    3.2.1
## v lubridate 1.9.3     v tidyr     1.3.1
## v purrr     1.0.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```r
# Use the conflicted package to manage conflicts
library(conflicted)

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

```
## [conflicted] Will prefer dplyr::filter over any other package.
```

```r
conflict_prefer("lag", "dplyr")
```

```
## [conflicted] Will prefer dplyr::lag over any other package.
```

Collect Data

Upload Divvy datasets (csv files) here

```
setwd('C:/Users/bhard/OneDrive/Documents/Cyclistic-CaseStudy/Data_2019_2020')
q1_2019 <- read_csv("Divvy_Trips_2019_Q1.csv")
```

```
## Rows: 365069 Columns: 12
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (4): from_station_name, to_station_name, usertype, gender
## dbl  (5): trip_id, bikeid, from_station_id, to_station_id, birthyear
## num  (1): tripduration
## dttm (2): start_time, end_time
##
## 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.
```

```
q1_2020 <- read_csv("Divvy_Trips_2020_Q1.csv")
```

```
## Rows: 426887 Columns: 13
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (7): ride_id, rideable_type, started_at, ended_at, start_station_name, e...
## dbl (6): start_station_id, end_station_id, start_lat, start_lng, end_lat, en...
##
## 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.
```

Wrangle data and combine into a single file

Compare column names each of the files

```
colnames(q1_2019)
```

```
##  [1] "trip_id"           "start_time"        "end_time"
##  [4] "bikeid"            "tripduration"      "from_station_id"
##  [7] "from_station_name" "to_station_id"     "to_station_name"
## [10] "usertype"          "gender"            "birthyear"
```

```
colnames(q1_2020)
```

```
##  [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"
```

Rename columns in q1_2019 to make them consistent with q1_2020

```r
(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> <dttm>              <dttm>                      <dbl>        <dbl>
##  1 21742443 2019-01-01 00:04:37 2019-01-01 00:11:07          2167          390
##  2 21742444 2019-01-01 00:08:13 2019-01-01 00:15:34          4386          441
##  3 21742445 2019-01-01 00:13:23 2019-01-01 00:27:12          1524          829
##  4 21742446 2019-01-01 00:13:45 2019-01-01 00:43:28           252         1783
##  5 21742447 2019-01-01 00:14:52 2019-01-01 00:20:56          1170          364
##  6 21742448 2019-01-01 00:15:33 2019-01-01 00:19:09          2437          216
##  7 21742449 2019-01-01 00:16:06 2019-01-01 00:19:03          2708          177
##  8 21742450 2019-01-01 00:18:41 2019-01-01 00:20:21          2796          100
##  9 21742451 2019-01-01 00:18:43 2019-01-01 00:47:30          6205         1727
## 10 21742452 2019-01-01 00:19:18 2019-01-01 00:24:54          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>
```

Inspect the dataframes

```r
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        : POSIXct[1:365069], format: "2019-01-01 00:04:37" "2019-01-01 00:08:13" ...
##  $ ended_at          : POSIXct[1:365069], format: "2019-01-01 00:11:07" "2019-01-01 00:15:34" ...
##  $ 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 (*)" "W
##  $ 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_datetime(format = ""),
##   ..   end_time = col_datetime(format = ""),
```

```
##   ..   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" "Wilt
##  $ 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>
```

Make data types consistent

```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
# Convert started_at and ended_at to datetime so that they can stack correctly
q1_2020 <-  mutate(q1_2020,
                   started_at = as_datetime(started_at),
                   ended_at = as_datetime(ended_at))
```

Combine the datasets

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

# 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"))
```

Transforming and cleaning the data

Inspecting Data

Inspect the new table that has been created

```r
colnames(all_trips)  #List of column names
```

```
## [1] "ride_id"            "started_at"         "ended_at"
## [4] "rideable_type"      "start_station_id"   "start_station_name"
## [7] "end_station_id"     "end_station_name"   "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>   <dttm>              <dttm>              <chr>                    <dbl>
## 1 217424~ 2019-01-01 00:04:37 2019-01-01 00:11:07 2167                       199
## 2 217424~ 2019-01-01 00:08:13 2019-01-01 00:15:34 4386                        44
## 3 217424~ 2019-01-01 00:13:23 2019-01-01 00:27:12 1524                        15
## 4 217424~ 2019-01-01 00:13:45 2019-01-01 00:43:28 252                        123
## 5 217424~ 2019-01-01 00:14:52 2019-01-01 00:20:56 1170                       173
## 6 217424~ 2019-01-01 00:15:33 2019-01-01 00:19:09 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        : POSIXct[1:791956], format: "2019-01-01 00:04:37" "2019-01-01 00:08:13" ...
##  $ ended_at          : POSIXct[1:791956], format: "2019-01-01 00:11:07" "2019-01-01 00:15:34" ...
##  $ 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] "Subscriber" "Subscriber" "Subscriber" "Subscriber" ...
```

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

```
##    ride_id            started_at                   
##  Length:791956      Min.   :2019-01-01 00:04:37.00  
##  Class :character   1st Qu.:2019-02-28 17:04:04.75  
##  Mode  :character   Median :2020-01-07 12:48:50.50  
##                     Mean   :2019-09-01 11:58:08.35  
##                     3rd Qu.:2020-02-19 19:31:54.75  
##                     Max.   :2020-03-31 23:51:34.00  
##                                                     
##     ended_at                      rideable_type      start_station_id
##  Min.   :2019-01-01 00:11:07.00   Length:791956      Min.   :  2.0   
##  1st Qu.:2019-02-28 17:15:58.75   Class :character   1st Qu.: 77.0   
##  Median :2020-01-07 13:02:50.00   Mode  :character   Median :174.0   
##  Mean   :2019-09-01 12:17:52.17                      Mean   :204.4   
##  3rd Qu.:2020-02-19 19:51:54.50                      3rd Qu.:291.0   
##  Max.   :2020-05-19 20:10:34.00                      Max.   :675.0   
##                                                                      
##  start_station_name end_station_id  end_station_name   member_casual     
##  Length:791956      Min.   :  2.0   Length:791956      Length:791956     
##  Class :character   1st Qu.: 77.0   Class :character   Class :character  
##  Mode  :character   Median :174.0   Mode  :character   Mode  :character  
##                     Mean   :204.4                                        
##                     3rd Qu.:291.0                                        
##                     Max.   :675.0                                        
##                     NA's   :1
```

Identifing Problems

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.

Data Cleaning

Begin by seeing how many observations fall under each usertype

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

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

Reassign to the desired values (we will go with the 2020 labels)

```
all_trips <-  all_trips %>%
  mutate(member_casual = recode(member_casual
                                ,"Subscriber" = "member"
                                ,"Customer" = "casual"))
```

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 op
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")
```

Add a "ride_length" calculation to all_trips (in seconds)

```
all_trips$ride_length <- difftime(all_trips$ended_at,all_trips$started_at)
```

Convert "ride_length" from Factor to numeric so we can run calculations on the data

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

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

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

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

Conduct a descriptive analysis

Summary

```r
summary(all_trips_v2$ride_length)
```

```
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
##        1      331      539     1189      912 10632022
```

Comparing Casual and Member users

```r
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                5372.7839
## 2                     member                 795.2523
```

```r
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                     1393
## 2                     member                      508
```

```r
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                 10632022
## 2                     member                  6096428
```

```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                        2
## 2                     member                        1
```

```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 all_trips_v2$ride_length
## 1                      casual                   Friday                6090.7373
## 2                      member                   Friday                 796.7338
## 3                      casual                   Monday                4752.0504
## 4                      member                   Monday                 822.3112
## 5                      casual                 Saturday                4950.7708
## 6                      member                 Saturday                 974.0730
## 7                      casual                   Sunday                5061.3044
## 8                      member                   Sunday                 972.9383
```

```
## 9                      casual                Thursday                8451.6669
## 10                     member                Thursday                 707.2093
## 11                     casual                 Tuesday                4561.8039
## 12                     member                 Tuesday                 769.4416
## 13                     casual               Wednesday                4480.3724
## 14                     member               Wednesday                 711.9838
```

```
# 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", "W
```

Average ride time by each day for Members vs Casual users

```
# 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 all_trips_v2$ride_length
## 1                      casual                  Sunday                5061.3044
## 2                      member                  Sunday                 972.9383
## 3                      casual                  Monday                4752.0504
## 4                      member                  Monday                 822.3112
## 5                      casual                 Tuesday                4561.8039
## 6                      member                 Tuesday                 769.4416
## 7                      casual               Wednesday                4480.3724
## 8                      member               Wednesday                 711.9838
## 9                      casual                Thursday                8451.6669
## 10                     member                Thursday                 707.2093
## 11                     casual                  Friday                6090.7373
## 12                     member                  Friday                 796.7338
## 13                     casual                Saturday                4950.7708
## 14                     member                Saturday                 974.0730
```

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
            ,average_duration = mean(ride_length)) %>%      # calculates the average duration
  arrange(member_casual, weekday)                              # sorts
```

```
## `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               18652            5061.
##  2 casual        Mon                5591            4752.
##  3 casual        Tue                7311            4562.
##  4 casual        Wed                7690            4480.
##  5 casual        Thu                7147            8452.
```

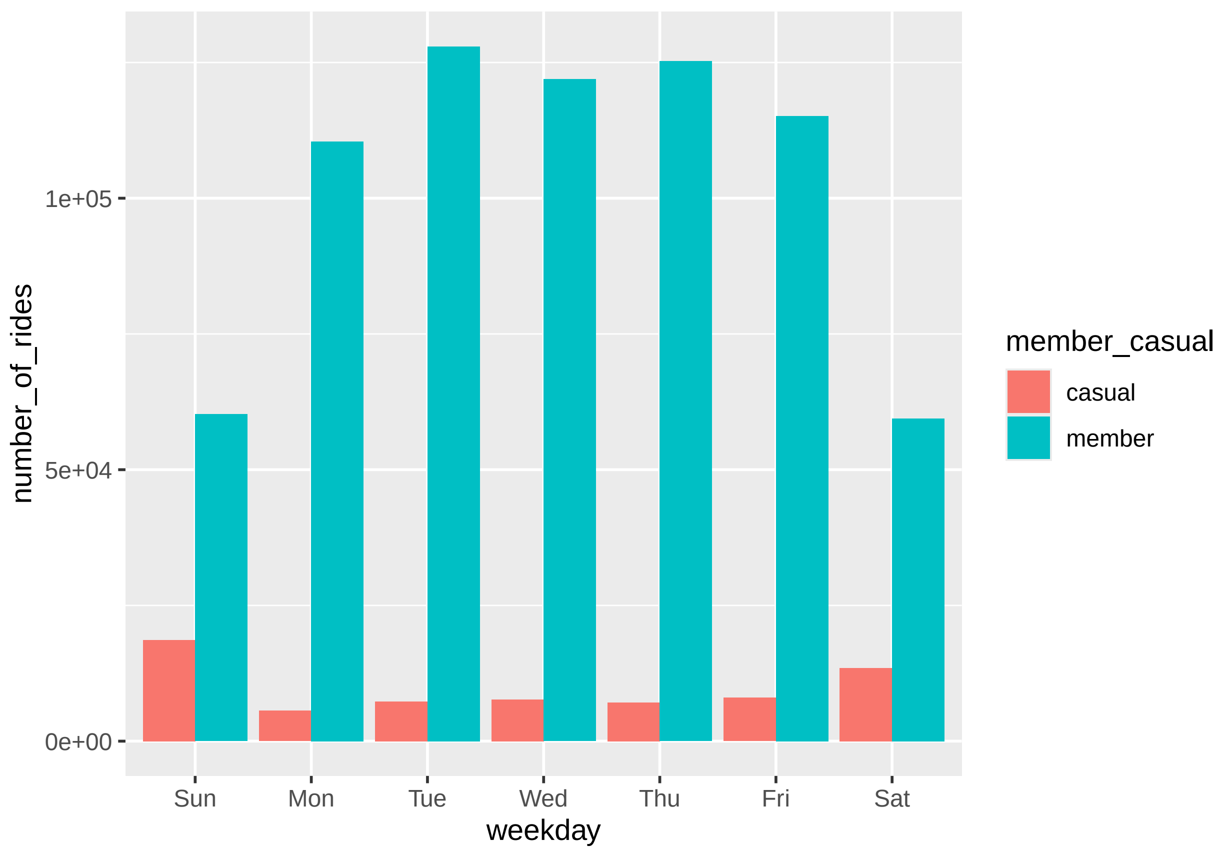
```
## 6 casual Fri 8013 6091.
## 7 casual Sat 13473 4951.
## 8 member Sun 60197 973.
## 9 member Mon 110430 822.
## 10 member Tue 127974 769.
## 11 member Wed 121902 712.
## 12 member Thu 125228 707.
## 13 member Fri 115168 797.
## 14 member Sat 59413 974.
```

Data Visualisations

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")
```

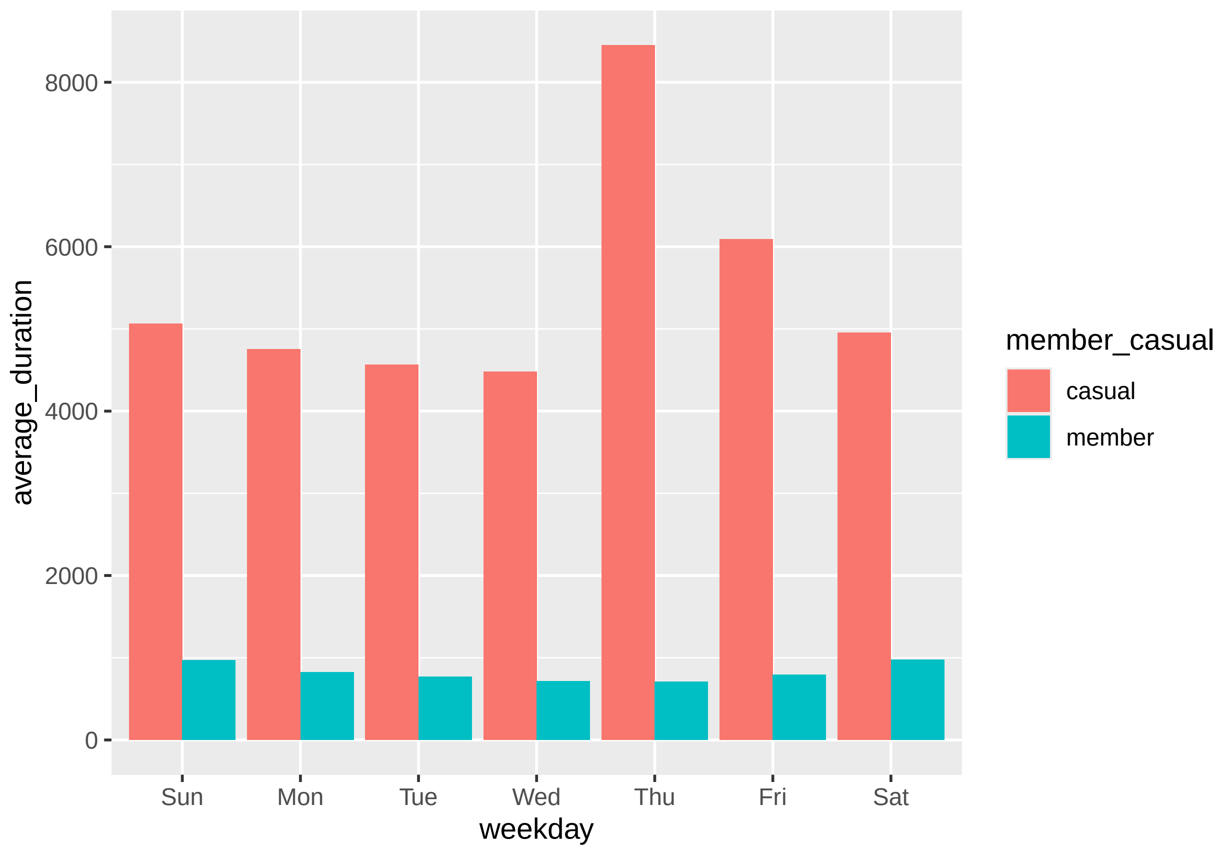
```
## `summarise()` has grouped output by 'member_casual'. You can override using the
## `.groups` argument.
```



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")
```

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



Findings Summary: Inconclusive

Our analysis reveals some important behavioral differences between Casuals and Members. These distinctions hint at some variations in fundamental values between the two groups, but at this stage, the information we've uncovered can only be used to make generalizations about the entire population of bike trips on both sides and is insufficient to draw any conclusions about your typical member or casual. Therefore, our findings cannot fully answer the question that started this project without collecting more relevant data and reiterating the data analysis process.

Speculations

A. Casuals primarily use Cyclistic bikes for leisure

We make that assumption based on the fact that casuals:

Bike twice as much on Saturdays and Sundays compared to any other day of the week Spend the majority of their bike trips near parks and water Spend significantly longer on average on every bike trip, suggesting that they spend time in-between docking stations doing leisurely activities Scarcely use Cyclistic bikes in the morning (6 am-noon) Do not use Cyclistic bikes often enough to warrant paying for an annual membership

B. Members get more out of Cyclistic bikes by using them for leisure and commuting consistently

We've drawn that conclusion based on the fact that members:

Rely on bikes consistently each week and year-round, with no notable preference on a single day of the week Use Cyclistic bikes often during the rush hours on a typical workday Have a large geographical spread in the downtown area, particularly in high urban dense areas Are motivated by the economics of an annual membership pass

Share Key Findings(ppt)

My presentation to stakeholders on my findings can be found here.

Act on Key Findings

Next Steps: Recommended Paths

A) Reiterating on Data Extraction and Analysis To confirm that our speculations are true and uncover any other notable behavioral differences, we need to survey a significant sample of our user population to discover what truly defines each user group's behavior. Gravitating towards qualitative data that would provide insights like opinions and motivations (what do they use Cyclistic bikes for?) gives some much-needed context behind our initial findings. In particular, finding out what can incentivize users will help us leverage consumer needs towards a more successful marketing strategy.In addition, collecting more quantitative data into user demographics would provide more information into your typical casual or member user with features like income, age, and weight.

The idea here is to get a good sense of the major obstacles and opportunities ahead of Cyclistic that could interfere or assist in a conversion strategy. For instance, if Casuals do not have enough disposable income, it would be incredibly difficult to persuade them to increase their fiscal commitment.

Ultimately, a successful conversion relies on a strategy that provides enough incentive to casual users to be willing to make the switch.

B) Forging Ahead with our Initial Findings Unfortunately, the safest choice often requires the investment of more time and resources. If Cyclistic executives can tolerate a certain level of risk, they can use the findings in this analysis to kick-start and form the basis of a few strategies. Regardless, due diligence is required to ensure the viability of any new marketing strategies.

The top three recommendations moving forward:

1. Consider alternatives to conversion, such as new service and pricing models The future of Cyclistic should not be limited to a single narrow strategy to achieve sustainable business growth. We should examine any options to innovate and enhance the current Cyclistic experience, even if it's radical. In particular, a three-tiered pricing approach could optimize revenue by offering a service that fits between single-use passes and an annual subscription that provides unlimited rides. This offering would seek a balance between the two, finding the sweet spot for casuals that cannot justify the economics of an expensive annual payment and the limitations of single-use passes. For example, a weekend pass would serve as a good compromise for users that typically only use Cyclistic for leisure on weekends. However, it's worth examining how this may inadvertently downgrade current annual members and if there's any way to mitigate any undesired effects from this strategy.

2. Explore ways to convey the benefits of biking more frequently If Cyclistic can incentivize Casual users to increase their biking tendencies, they may be willing to upgrade their commitment. This incentivization could come from various strategies, one of which could come through an effective marketing campaign that informs users of the main benefits of biking. As mentioned earlier, this strategy is risky without an in-depth understanding of casual users.

Convincing someone to change their habits has always been a monumental task. This strategy would need to consider how Cyclistic could realistically disrupt Casual user habits and instill a desire to bike year-round. For this reason, it may be best to narrow the marketing strategy's focus to a subset of Casuals that are particularly susceptible to influence and have relevant personal goals that are achievable with Cyclistic.

3. Explore models that reward higher-priced offerings with additional privileges As it stands, Casuals and Members experience a relatively similar level of service. They are both given the same privileges on a first-come-first-serve service basis for every available bike type. Naturally, some people will experience pain points under this system whenever they come up to a station and miss out on using their favorite bike type or are inconvenienced with a biking shortage during peak biking season.

By minimizing this inconvenience for Members, Casuals will bear the brunt of the inconvenience. For example, Members could have the privilege of knowing what bikes are available in real-time and can reserve them ahead of time, thereby eliminating Casuals' access to the first-come-first-serve experience. In turn, they will have to tolerate the inconvenience, upgrade their commitment, or switch to a competing product. Admittedly, this is risky in the sense that it could cause severe backlash and result in a high churn rate. However, Cyclistic could execute a strategy that slowly enacts these changes long-term and exposes users to negligible and tolerable differences in the Cyclistic experience.

These privileges would not necessarily convert a large portion of Casuals, as I doubt it would provide enough justification for them to pay more for the same amount of bike use with extra perks. Regardless, it is worth examining the pain threshold for Casuals and testing how they would react. This strategy could be synergized with the three-tiered pricing strategy, introducing different levels of perks for each service offering and encouraging most users to choose the service that provides the best experience at a reasonable price.