Divvy Cyclistic Case Study

Dr Inam ur Rehman

2024-08-16

A. ASK Phase

1. Scenario

As a 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, marketing team wants to understand how casual riders and annual members use Cyclistic bikes differently. From these insights, design a new marketing strategy to convert casual riders into annual members. But first, Cyclistic executives must approve recommendations, so they must be backed up with compelling data insights and professional data visualizations.

2. About the company

Lyft Bikes and Scooters, LLC ("Bikeshare") operates the City of Chicago's ("City") Divvy bicycle sharing service. Bikeshare and the City are committed to supporting bicycling as an alternative transportation option. In 2016, Cyclistic launched a successful bike-share offering. Since then, the program has grown to a fleet of 5,824 bicycles that are geotracked and locked into a network of 692 stations across Chicago. The bikes can be unlocked from one station and returned to any other station in the system anytime.

3. Data License Agreement

City permits Bikeshare to make certain Divvy system data owned by the City ("Data") available to the public, subject to the terms and conditions of this License Agreement ("Agreement"). By accessing or using any of the Data, you agree to all of the terms and conditions of this Agreement.

License. Bikeshare hereby grants us a non-exclusive, royalty-free, limited, perpetual license to access, reproduce, analyze, copy, modify, distribute in your product or service and use the Data for any lawful purpose ("License"). https://www.divvybikes.com/data-license-agreement

4. Characters and teams

Cyclistic: A bike-share program that features more than 5,800 bicycles and 600 docking stations. 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 and my manager. 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. I joined this team six months ago and have been busy learning about Cyclistic's mission and business goals—as well as how I, as a junior data analyst, can help Cyclistic achieve them.

Cyclistic executive team: The notoriously detail-oriented executive team will decide whether to approve the recommended marketing program.

5. Business Task

Design a new marketing strategy to convert casual riders into annual members.

B.Prepare Phase

STEP 1: Load Packages & COLLECT DATA

Load Packages

```
library(readr)
library(janitor)
library(lubridate)
library(tidyverse)
library(ggplot2)
library(dplyr)
library(tibble)
library(ggpubr)
```

Upload Divvy data-sets (csv files)

```
q1_2019 <- read_csv("../data/Divvy_Trips_2019_Q1.csv")
q1_2020 <- read_csv("../data/Divvy_Trips_2020_Q1.csv")
```

STEP 2: 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 to make them consistent with q1_2020

```
(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 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        : 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>
```

```
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" "C9A38
##  $ rideable_type     : chr [1:426887] "docked_bike" "docked_bike" "docked_bike" "docked_bike" ...
##  $ started_at        : POSIXct[1:426887], format: "2020-01-21 20:06:59" "2020-01-30 14:22:39" ...
##  $ ended_at          : POSIXct[1:426887], format: "2020-01-21 20:14:30" "2020-01-30 14:26:22" ...
##  $ 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_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>
```

ride_id and rideable_type are numeric in q1_2019, so we convert ride_id and rideable_type to character so that they can stack correctly with q1_2020 data frame.

```
q1_2019 <- mutate(q1_2019, ride_id = as.character(ride_id)
,rideable_type = as.character(rideable_type))
```

Stack individual quarter's data frames into one big data frame

```
all_trips <- bind_rows(q1_2019, q1_2020)
glimpse(all_trips)
```

```
## Rows: 791,956
## Columns: 16
## $ ride_id            <chr> "21742443", "21742444", "21742445", "21742446", "21~
## $ started_at         <dttm> 2019-01-01 00:04:37, 2019-01-01 00:08:13, 2019-01-~
## $ ended_at           <dttm> 2019-01-01 00:11:07, 2019-01-01 00:15:34, 2019-01-~
## $ rideable_type      <chr> "2167", "4386", "1524", "252", "1170", "2437", "270~
## $ tripduration       <dbl> 390, 441, 829, 1783, 364, 216, 177, 100, 1727, 336,~
## $ start_station_id   <dbl> 199, 44, 15, 123, 173, 98, 98, 211, 150, 268, 299, ~
## $ start_station_name <chr> "Wabash Ave & Grand Ave", "State St & Randolph St",~
## $ end_station_id     <dbl> 84, 624, 644, 176, 35, 49, 49, 142, 148, 141, 295, ~
## $ end_station_name   <chr> "Milwaukee Ave & Grand Ave", "Dearborn St & Van Bur~
## $ member_casual      <chr> "Subscriber", "Subscriber", "Subscriber", "Subscrib~
## $ gender             <chr> "Male", "Female", "Female", "Male", "Male", "Female~
## $ birthyear          <dbl> 1989, 1990, 1994, 1993, 1994, 1983, 1984, 1990, 199~
## $ start_lat          <dbl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ start_lng          <dbl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ end_lat            <dbl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ end_lng            <dbl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
```

Remove lat, long, birthyear, and gender fields as they are not required in current study.

```
all_trips <- all_trips %>%
select(-c(start_lat, start_lng, end_lat, end_lng, birthyear, gender, tripduration))
glimpse(all_trips)
```

```
## Rows: 791,956
## Columns: 9
## $ ride_id            <chr> "21742443", "21742444", "21742445", "21742446", "21~
## $ started_at         <dttm> 2019-01-01 00:04:37, 2019-01-01 00:08:13, 2019-01-~
## $ ended_at           <dttm> 2019-01-01 00:11:07, 2019-01-01 00:15:34, 2019-01-~
## $ rideable_type      <chr> "2167", "4386", "1524", "252", "1170", "2437", "270~
## $ start_station_id   <dbl> 199, 44, 15, 123, 173, 98, 98, 211, 150, 268, 299, ~
## $ start_station_name <chr> "Wabash Ave & Grand Ave", "State St & Randolph St",~
## $ end_station_id     <dbl> 84, 624, 644, 176, 35, 49, 49, 142, 148, 141, 295, ~
## $ end_station_name   <chr> "Milwaukee Ave & Grand Ave", "Dearborn St & Van Bur~
## $ member_casual      <chr> "Subscriber", "Subscriber", "Subscriber", "Subscrib~
```

STEP 3: CLEAN UP AND ADD DATA TO PREPARE FOR ANALYSIS

Inspect the new table that has been created

```
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.
```

```
## # 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
tail(all_trips)  #See the last 6 rows of data frame.
```

```
## # A tibble: 6 x 9
##   ride_id started_at          ended_at            rideable_type start_station_id
##   <chr>   <dttm>              <dttm>              <chr>                     <dbl>
## 1 6F4D22~ 2020-03-10 10:40:27 2020-03-10 10:40:29 docked_bike                675
## 2 ADDAA3~ 2020-03-10 10:40:06 2020-03-10 10:40:07 docked_bike                675
## 3 82B10F~ 2020-03-07 15:25:55 2020-03-07 16:14:03 docked_bike                161
## 4 AA0D5A~ 2020-03-01 13:12:38 2020-03-01 13:38:29 docked_bike                141
## 5 329636~ 2020-03-07 18:02:45 2020-03-07 18:13:18 docked_bike                672
## 6 064EC7~ 2020-03-08 13:03:57 2020-03-08 13:32:27 docked_bike                110
## # i 4 more variables: start_station_name <chr>, end_station_id <dbl>,
## #   end_station_name <chr>, member_casual <chr>
```

See list of columns and data types (numeric, character, etc)

```r
str(all_trips)
```

```
## 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" ...
```

Statistical summary of data. Mainly for numerics

```
summary(all_trips)
```

```
##    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
```

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. Seeing how many observations fall under each usertype

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

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

In the "member_casual" column, replace "Subscriber" with "member" and "Customer" with "casual"

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

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

```
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) 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)
```

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        : 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 (*)" "U
##  $ member_casual     : chr [1:791956] "member" "member" "member" "member" ...
##  $ date              : Date[1:791956], format: "2019-01-01" "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] 390 441 829 1783 ...
##   ..- attr(*, "units")= chr "secs"
```

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

```
is.factor(all_trips$ride_length)
```

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

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

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

Remove "bad" data

```r
table(all_trips$start_station_name == "HQ QR")
```

```
##
##  FALSE   TRUE
## 788189   3767
```

```r
table(all_trips$ride_length < 0)
```

```
##
##  FALSE   TRUE
## 791839    117
```

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 https://www.datasciencemadesimple.com/delete-or-drop-rows-in-r-with-conditions-2/

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

```
##
##  FALSE
## 788189
```

```r
table(all_trips_v2$ride_length < 0)
```

```
##
##  FALSE
## 788189
```

C. Analysis Phase

STEP 4: CONDUCT DESCRIPTIVE ANALYSIS

Descriptive analysis on ride_length (all figures in seconds)

```r
mean(all_trips_v2$ride_length) #straight average (total ride length / rides)
```

```
## [1] 1189.459
```

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

```
## [1] 539
```

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

```
## [1] 10632022
```

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

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

Compare members and casual 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
```

For Pie chart

```r
member_casual_percen <- all_trips_v2 |>
  group_by(member_casual) |>
  summarise( total= n()) |>
  mutate(totals = sum(total)) %>%
group_by(member_casual) %>%
summarise(total_percent = total / totals) %>%
mutate(labels = scales::percent(total_percent))
head(member_casual_percen)
```

```
## # A tibble: 2 x 3
##   member_casual total_percent labels
##   <chr>                 <dbl> <chr>
## 1 casual               0.0861 9%
## 2 member               0.914  91%
```

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
```

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", "
```

Analyze ridership data by type and weekday. To see behavior of users on weekend

```
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.
```

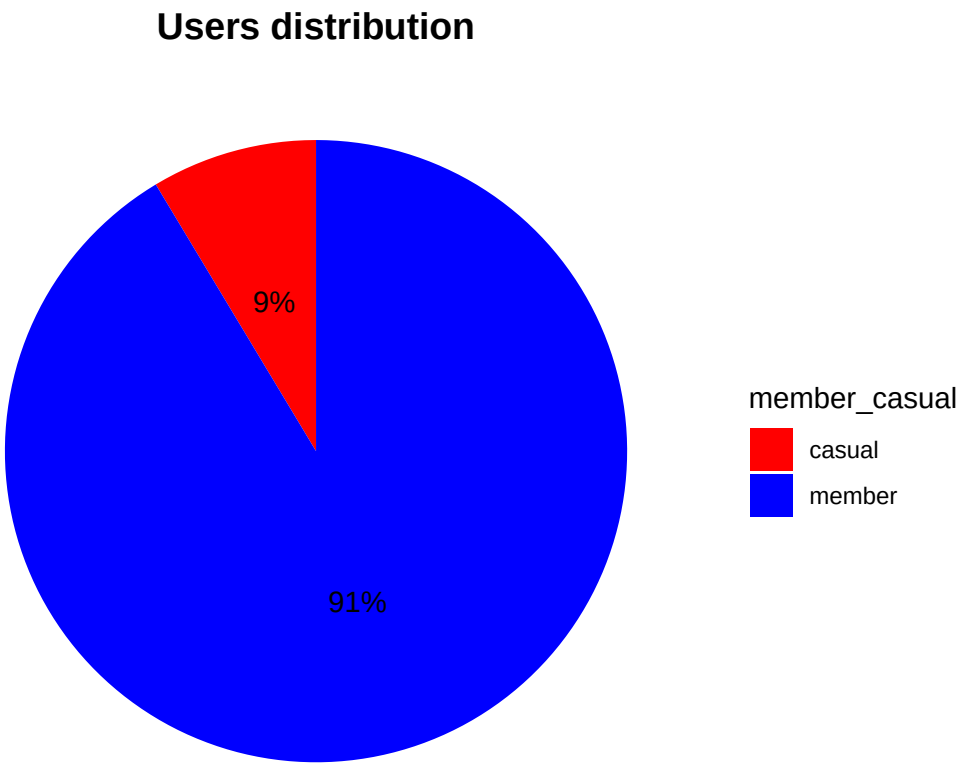
D.Visualization Phase

Users distribution

```
head(member_casual_percen)
```

```
## # A tibble: 2 x 3
##   member_casual total_percent labels
##   <chr>                 <dbl> <chr>
## 1 casual               0.0861 9%
## 2 member               0.914  91%
```

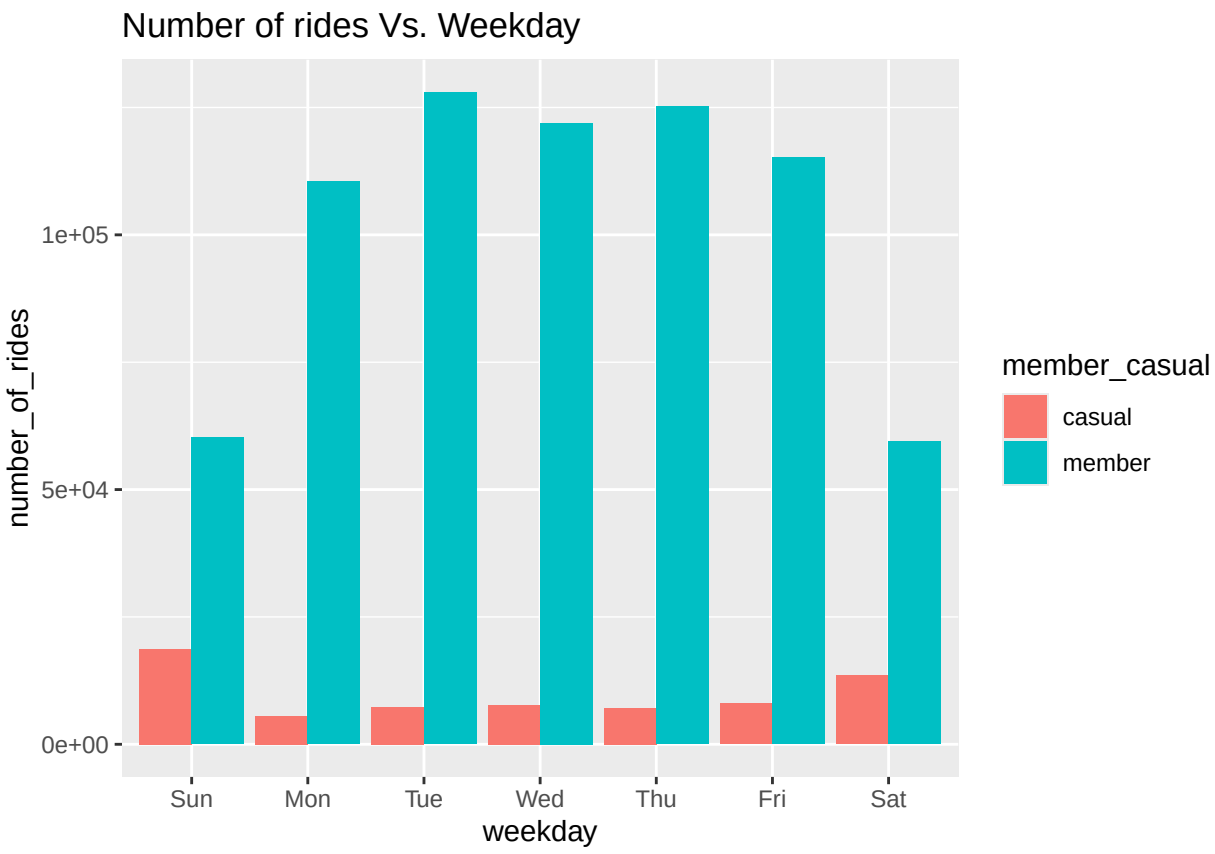
```
member_casual_percen %>%
ggplot(aes(x="",y=total_percent, fill=member_casual)) +
geom_bar(stat = "identity", width = 1)+
coord_polar("y", start=0)+
theme_minimal()+
theme(axis.title.x= element_blank(),
axis.title.y = element_blank(),
panel.border = element_blank(),
panel.grid = element_blank(),
axis.ticks = element_blank(),
axis.text.x = element_blank(),
plot.title = element_text(hjust = 0.5, size=14, face = "bold")) +
scale_fill_manual(values = c("red", "blue")) +
geom_text(aes(label = labels),
position = position_stack(vjust = 0.5))+
labs(title="Users distribution")
```



v 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 Vs. Weekday")
```

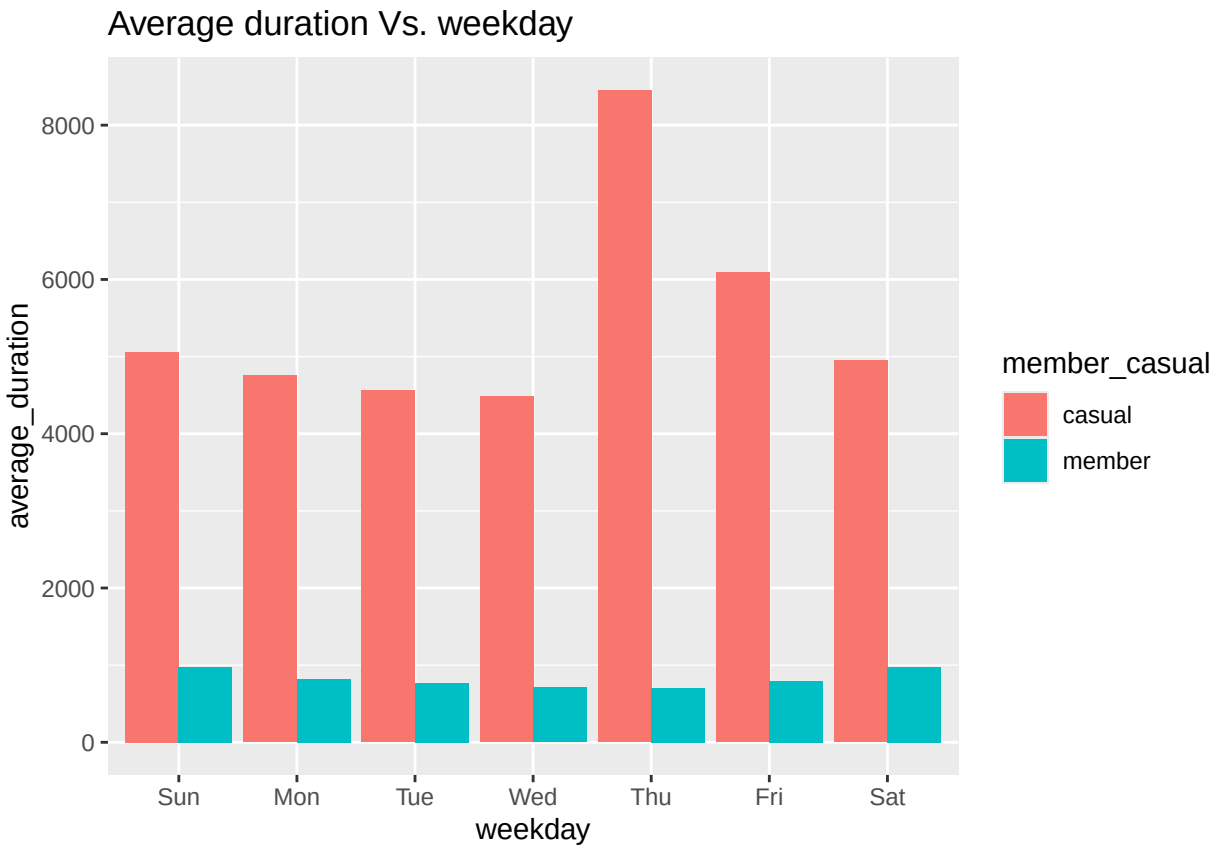
```
## `summarise()` has grouped output by 'member_casual'. You can override using the
## `.groups` argument.
```



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") +
labs(title="Average duration Vs. weekday")
```

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



STEP 5: EXPORT SUMMARY FILE FOR FURTHER ANALYSIS

```
counts <- aggregate(all_trips_v2$ride_length ~ all_trips_v2$member_casual + all_trips_v2$day_of_week, FU
write.csv(counts,  'C:\\Users\\Inam\\Desktop\\Data Analytic Professional\\Case1usingguidlines\\avg_ride_
```

E. ACT Phase

Recommedations

1.As casual riders are already aware of the Cyclistic program and have chosen Cyclistic for their mobility needs so their change to annual member will be easy as compare to new customers.

2.From users distribution pie chart annual members are 91% while casual are 9%. This implies that company gave overall more or equal benefits to member as well as casual riders. To convert casual users into members promotions schemes should focused on members more as compare to single and day riders.

3.From number of rides Vs. weekday plot, number of rides of members are far more throughout the week than casual riders.It seems that members are using this facility for commute to work or go to their job sites because in working days their numbers of rides are more than weekend. While number of rides increased on weekend days for casual riders which implies that they use this facility to explore the city or some other adventures beside their mobility needs . At this moment membership incentive should be offered to casual riders through social media, emails and other proper advertising sources.

4.Some detailed financial benefits worked out and share with casual users for membership as finance analysts already worked out.

5.From Average duration Vs. weekday graph average duration of casual riders is far more than members riders. This suggests two things. Firstly Casual riders not care about financial benefits and they think they required this facility occasional so no need of membership. Secondly they calculated and found trip or day rider options are more saving than membership. Finance analyst can answer this question. In the light of that analysis casual rider can be convinced better for membership.

6.Although causal members are 9% but their ride length is much more than member. Members average maximum ride length is 1000 seconds while causal rider more than 9000 seconds. This suggest new survey whats the hurdles the casual rider feels not becoming member. Manager can help in removing that hurdles.