Ciclystic Trip Data Case Study

Javier Mas Ruiz

2022-10-24

Contents

Case Study for Google's Data Analytics Professional Certificate

Introduction

Welcome to the Cyclistic bike share analysis case study. In this case study, you will perform many real-world tasks typical of a junior data analyst. You will work for a fictional company called Cyclistic and meet different characters and team members. To answer the company's key questions, you will follow the steps of the data analysis process: ask, prepare, process, analyze, share and act. In this process, the case study roadmap charts, including guiding questions and key tasks, will help you stay on track.

Scenario

You are a junior data analyst working on the marketing analyst team at Cyclistic, a bike-sharing company in Chicago. The marketing director believes that the company's future success depends on maximizing the number of annual memberships. Therefore, your team wants to understand what differences exist in the use of Cyclistic bikes between casual riders and annual members. Through this knowledge, your team will design a new marketing strategy to convert casual riders into annual members. Before that, however, Cyclistic executives must approve your recommendations, so back up your proposal with compelling data insights and professional data visualizations.

Characters and teams

- **Cyclistic:** A bike share program that includes 5,800 bikes and 600 stations. Cyclistic is notable for also offering recumbent bikes, manual tricycles and cargo bikes that offer a more inclusive use of shared bikes for people with disabilities and riders who cannot use a standard two-wheeled bicycle. The majority of cyclists choose traditional bikes, about 8% of cyclists use the assisted options. Cyclistic users are most likely to use the bicycle for recreation, but about 30% use it to commute to work each day.
- **Lily Moreno:** The marketing director and your manager. Moreno is responsible for developing campaigns and initiatives to promote the bike share program. Campaigns may include email, social media and other channels.
- **Cyclistic's marketing data computational analytics team:** A team of data analysts who are responsible for collecting, analyzing and reporting data that helps drive Cyclistic's marketing strategy. You joined this team six months ago and have dedicated yourself to not only learning about Cyclistic's mission and business goals, but also seeing how you can help Cyclistic achieve it, from your position as a junior data analyst.
- **Cyclistic's executive team:** The highly detailed executive team will decide whether to approve the recommended marketing program.

About the Company

Cyclistic is a company that in 2016 launched a successful bike sharing program. Since then, the program grew to a fleet of 5,824 geo-tagged and locked bikes at a network of 692 stations across Chicago. Bikes can be unlocked from one station and returned to any other station in the system at any time.

Analysis Stages and Report

Business objective to be faced

Moreno (marketing director) believes that maximizing the number of annual members will be key to future growth. So a clear goal is set: **Design marketing strategies aimed at converting occasional cyclists into annual members.** In order to response *How do annual members and occasional riders differ in their use of Cyclistic bicycles?*

Description of data sources used

The data used in this analysis is public and was provided by Motivate International Inc. under this license and it can be downloaded here.

Cleaning and Manipulating Data Process

```
# Importing and installing relevant packages used for the analisys
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --

## v ggplot2 3.3.6     v purrr   0.3.4
## v tibble  3.1.7     v dplyr   1.0.9
## v tidyr   1.2.0     v stringr 1.4.0
## v readr   2.1.2     v forcats 0.5.1

## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(lubridate)
```

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

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

```
library(janitor)
```

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

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

```
library(tidyr)
library(ggplot2)
library(dplyr)
getwd() # display my working directory
```

```
## [1] "D:/Data_Science/Data_Analysis_by_Google_2022_julio/Curso 8. Curso Final de Analisis Computacion
```

```
setwd("D:/Data_Science/Data_Analysis_by_Google_2022_julio/Curso 8. Curso Final de Analisis Computacional
```

```
# Importing all .csv files from PC
# STEP 1: COLLECT DATA

q2_2019 <- read_csv("Divvy_Trips_2019_Q2.csv")
```

```
## Rows: 1108163 Columns: 12
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (4): 03 - Rental Start Station Name, 02 - Rental End Station Name, User...
## dbl  (5): 01 - Rental Details Rental ID, 01 - Rental Details Bike ID, 03 - R...
## dttm (2): 01 - Rental Details Local Start Time, 01 - Rental Details Local 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.
```

```
q3_2019 <- read_csv("Divvy_Trips_2019_Q3.csv")
```

```
## Rows: 1640718 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
## 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.
```

```
q4_2019 <- read_csv("Divvy_Trips_2019_Q4.csv")
```

```
## Rows: 704054 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
## 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  (5): ride_id, rideable_type, start_station_name, end_station_name, memb...
## dbl  (6): start_station_id, end_station_id, start_lat, start_lng, end_lat, e...
## dttm (2): started_at, ended_at
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
# ====================================================
# STEP 2: 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

colnames(q3_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(q4_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(q2_2019)
```

```
##  [1] "01 - Rental Details Rental ID"
##  [2] "01 - Rental Details Local Start Time"
##  [3] "01 - Rental Details Local End Time"
##  [4] "01 - Rental Details Bike ID"
##  [5] "01 - Rental Details Duration In Seconds Uncapped"
##  [6] "03 - Rental Start Station ID"
##  [7] "03 - Rental Start Station Name"
##  [8] "02 - Rental End Station ID"
##  [9] "02 - Rental End Station Name"
## [10] "User Type"
## [11] "Member Gender"
## [12] "05 - Member Details Member Birthday Year"
```

```
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 (as this will be the supposed going-forward tabl
(q3_2019 <- rename(q3_2019, ride_id = trip_id, rideable_type = bikeid, started_at = start_time, ended_a
```

```
## # A tibble: 1,640,718 x 12
##     ride_id started_at          ended_at            rideable_type tripduration
##       <dbl> <dttm>              <dttm>                      <dbl>        <dbl>
##  1 23479388 2019-07-01 00:00:27 2019-07-01 00:20:41          3591         1214
##  2 23479389 2019-07-01 00:01:16 2019-07-01 00:18:44          5353         1048
##  3 23479390 2019-07-01 00:01:48 2019-07-01 00:27:42          6180         1554
##  4 23479391 2019-07-01 00:02:07 2019-07-01 00:27:10          5540         1503
##  5 23479392 2019-07-01 00:02:13 2019-07-01 00:22:26          6014         1213
##  6 23479393 2019-07-01 00:02:21 2019-07-01 00:07:31          4941          310
##  7 23479394 2019-07-01 00:02:24 2019-07-01 00:23:12          3770         1248
##  8 23479395 2019-07-01 00:02:26 2019-07-01 00:28:16          5442         1550
##  9 23479396 2019-07-01 00:02:34 2019-07-01 00:28:57          2957         1583
## 10 23479397 2019-07-01 00:02:45 2019-07-01 00:29:14          6091         1589
```

```
## # ... with 1,640,708 more rows, and 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>
```

```
(q4_2019 <- rename(q4_2019, ride_id = trip_id, rideable_type = bikeid, started_at = start_time, ended_at
```

```
## # A tibble: 704,054 x 12
##     ride_id started_at          ended_at            rideable_type tripduration
##       <dbl> <dttm>              <dttm>                      <dbl>        <dbl>
##  1 25223640 2019-10-01 00:01:39 2019-10-01 00:17:20          2215          940
##  2 25223641 2019-10-01 00:02:16 2019-10-01 00:06:34          6328          258
##  3 25223642 2019-10-01 00:04:32 2019-10-01 00:18:43          3003          850
##  4 25223643 2019-10-01 00:04:32 2019-10-01 00:43:43          3275         2350
##  5 25223644 2019-10-01 00:04:34 2019-10-01 00:35:42          5294         1867
##  6 25223645 2019-10-01 00:04:38 2019-10-01 00:10:51          1891          373
##  7 25223646 2019-10-01 00:04:52 2019-10-01 00:22:45          1061         1072
##  8 25223647 2019-10-01 00:04:57 2019-10-01 00:29:16          1274         1458
##  9 25223648 2019-10-01 00:05:20 2019-10-01 00:29:18          6011         1437
## 10 25223649 2019-10-01 00:05:20 2019-10-01 02:23:46          2957         8306
## # ... with 704,044 more rows, and 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>
```

```
(q2_2019 <- rename(q2_2019, ride_id = '01 - Rental Details Rental ID', rideable_type = "01 - Rental Deta
```

```
## # A tibble: 1,108,163 x 12
##     ride_id started_at          ended_at            rideable_type
##       <dbl> <dttm>              <dttm>                      <dbl>
##  1 22178529 2019-04-01 00:02:22 2019-04-01 00:09:48          6251
##  2 22178530 2019-04-01 00:03:02 2019-04-01 00:20:30          6226
##  3 22178531 2019-04-01 00:11:07 2019-04-01 00:15:19          5649
##  4 22178532 2019-04-01 00:13:01 2019-04-01 00:18:58          4151
##  5 22178533 2019-04-01 00:19:26 2019-04-01 00:36:13          3270
##  6 22178534 2019-04-01 00:19:39 2019-04-01 00:23:56          3123
##  7 22178535 2019-04-01 00:26:33 2019-04-01 00:35:41          6418
##  8 22178536 2019-04-01 00:29:48 2019-04-01 00:36:11          4513
##  9 22178537 2019-04-01 00:32:07 2019-04-01 01:07:44          3280
## 10 22178538 2019-04-01 00:32:19 2019-04-01 01:07:39          5534
## # ... with 1,108,153 more rows, and 8 more variables:
## #   `01 - Rental Details Duration In Seconds Uncapped` <dbl>,
## #   start_station_id <dbl>, start_station_name <chr>, end_station_id <dbl>,
## #   end_station_name <chr>, member_casual <chr>, `Member Gender` <chr>,
## #   `05 - Member Details Member Birthday Year` <dbl>
```

```
# Inspect the dataframes and look for incongruencies
str(q1_2020)
```

```
## spec_tbl_df [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>
```

```
str(q4_2019)
```

```
## spec_tbl_df [704,054 x 12] (S3: spec_tbl_df/tbl_df/tbl/data.frame)
##  $ ride_id           : num [1:704054] 25223640 25223641 25223642 25223643 25223644 ...
##  $ started_at        : POSIXct[1:704054], format: "2019-10-01 00:01:39" "2019-10-01 00:02:16" ...
##  $ ended_at          : POSIXct[1:704054], format: "2019-10-01 00:17:20" "2019-10-01 00:06:34" ...
##  $ rideable_type     : num [1:704054] 2215 6328 3003 3275 5294 ...
##  $ tripduration      : num [1:704054] 940 258 850 2350 1867 ...
##  $ start_station_id  : num [1:704054] 20 19 84 313 210 156 84 156 156 336 ...
##  $ start_station_name: chr [1:704054] "Sheffield Ave & Kingsbury St" "Throop (Loomis) St & Taylor St
##  $ end_station_id    : num [1:704054] 309 241 199 290 382 226 142 463 463 336 ...
##  $ end_station_name  : chr [1:704054] "Leavitt St & Armitage Ave" "Morgan St & Polk St" "Wabash Ave &
##  $ member_casual     : chr [1:704054] "Subscriber" "Subscriber" "Subscriber" "Subscriber" ...
##  $ gender            : chr [1:704054] "Male" "Male" "Female" "Male" ...
##  $ birthyear         : num [1:704054] 1987 1998 1991 1990 1987 ...
##  - 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(q3_2019)
```

```
## spec_tbl_df [1,640,718 x 12] (S3: spec_tbl_df/tbl_df/tbl/data.frame)
##  $ ride_id           : num [1:1640718] 23479388 23479389 23479390 23479391 23479392 ...
##  $ started_at        : POSIXct[1:1640718], format: "2019-07-01 00:00:27" "2019-07-01 00:01:16" ...
##  $ ended_at          : POSIXct[1:1640718], format: "2019-07-01 00:20:41" "2019-07-01 00:18:44" ...
##  $ rideable_type     : num [1:1640718] 3591 5353 6180 5540 6014 ...
##  $ tripduration      : num [1:1640718] 1214 1048 1554 1503 1213 ...
##  $ start_station_id  : num [1:1640718] 117 381 313 313 168 300 168 313 43 43 ...
##  $ start_station_name: chr [1:1640718] "Wilton Ave & Belmont Ave" "Western Ave & Monroe St" "Lakevie
##  $ end_station_id    : num [1:1640718] 497 203 144 144 62 232 62 144 195 195 ...
##  $ end_station_name  : chr [1:1640718] "Kimball Ave & Belmont Ave" "Western Ave & 21st St" "Larrabee
##  $ member_casual     : chr [1:1640718] "Subscriber" "Customer" "Customer" "Customer" ...
##  $ gender            : chr [1:1640718] "Male" NA NA NA ...
##  $ birthyear         : num [1:1640718] 1992 NA NA NA NA ...
##  - 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(q2_2019)
```

```
## spec_tbl_df [1,108,163 x 12] (S3: spec_tbl_df/tbl_df/tbl/data.frame)
##  $ ride_id                                          : num [1:1108163] 22178529 22178530 22178531 2217
##  $ started_at                                       : POSIXct[1:1108163], format: "2019-04-01 00:02:2
##  $ ended_at                                         : POSIXct[1:1108163], format: "2019-04-01 00:09:4
##  $ rideable_type                                    : num [1:1108163] 6251 6226 5649 4151 3270 ...
##  $ 01 - Rental Details Duration In Seconds Uncapped: num [1:1108163] 446 1048 252 357 1007 ...
##  $ start_station_id                                 : num [1:1108163] 81 317 283 26 202 420 503 260 2
##  $ start_station_name                               : chr [1:1108163] "Daley Center Plaza" "Wood St &
##  $ end_station_id                                   : num [1:1108163] 56 59 174 133 129 426 500 499 2
##  $ end_station_name                                 : chr [1:1108163] "Desplaines St & Kinzie St" "Wal
##  $ member_casual                                    : chr [1:1108163] "Subscriber" "Subscriber" "Subs
##  $ Member Gender                                    : chr [1:1108163] "Male" "Female" "Male" "Male" .
##  $ 05 - Member Details Member Birthday Year         : num [1:1108163] 1975 1984 1990 1993 1992 ...
##  - attr(*, "spec")=
```

```
##   .. cols(
##   ..   `01 - Rental Details Rental ID` = col_double(),
##   ..   `01 - Rental Details Local Start Time` = col_datetime(format = ""),
##   ..   `01 - Rental Details Local End Time` = col_datetime(format = ""),
##   ..   `01 - Rental Details Bike ID` = col_double(),
##   ..   `01 - Rental Details Duration In Seconds Uncapped` = col_number(),
##   ..   `03 - Rental Start Station ID` = col_double(),
##   ..   `03 - Rental Start Station Name` = col_character(),
##   ..   `02 - Rental End Station ID` = col_double(),
##   ..   `02 - Rental End Station Name` = col_character(),
##   ..   `User Type` = col_character(),
##   ..   `Member Gender` = col_character(),
##   ..   `05 - Member Details Member Birthday Year` = col_double()
##   .. )
##  - attr(*, "problems")=<externalptr>
```

```
# Convert ride_id and rideable_type to character so that they can stack correctly
q4_2019 <- mutate(q4_2019, ride_id = as.character(ride_id) ,rideable_type = as.character(rideable_type))
q3_2019 <- mutate(q3_2019, ride_id = as.character(ride_id) ,rideable_type = as.character(rideable_type))
q2_2019 <- mutate(q2_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(q2_2019, q3_2019, q4_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, "01 - R
```

```
#=====================================================
# 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"
```

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

```
## [1] 3879822
```

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

```
## [1] 3879822       9
```

```
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 221785~ 2019-04-01 00:02:22 2019-04-01 00:09:48 6251                          81
## 2 221785~ 2019-04-01 00:03:02 2019-04-01 00:20:30 6226                         317
## 3 221785~ 2019-04-01 00:11:07 2019-04-01 00:15:19 5649                         283
## 4 221785~ 2019-04-01 00:13:01 2019-04-01 00:18:58 4151                          26
## 5 221785~ 2019-04-01 00:19:26 2019-04-01 00:36:13 3270                         202
## 6 221785~ 2019-04-01 00:19:39 2019-04-01 00:23:56 3123                         420
## # ... with 4 more variables: start_station_name <chr>, end_station_id <dbl>,
## #   end_station_name <chr>, member_casual <chr>
```

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

```
## tibble [3,879,822 x 9] (S3: tbl_df/tbl/data.frame)
##  $ ride_id           : chr [1:3879822] "22178529" "22178530" "22178531" "22178532" ...
##  $ started_at        : POSIXct[1:3879822], format: "2019-04-01 00:02:22" "2019-04-01 00:03:02" ...
##  $ ended_at          : POSIXct[1:3879822], format: "2019-04-01 00:09:48" "2019-04-01 00:20:30" ...
##  $ rideable_type     : chr [1:3879822] "6251" "6226" "5649" "4151" ...
##  $ start_station_id  : num [1:3879822] 81 317 283 26 202 420 503 260 211 211 ...
##  $ start_station_name: chr [1:3879822] "Daley Center Plaza" "Wood St & Taylor St" "LaSalle St & Jacks
##  $ end_station_id    : num [1:3879822] 56 59 174 133 129 426 500 499 211 211 ...
##  $ end_station_name  : chr [1:3879822] "Desplaines St & Kinzie St" "Wabash Ave & Roosevelt Rd" "Canal
##  $ member_casual     : chr [1:3879822] "Subscriber" "Subscriber" "Subscriber" "Subscriber" ...
```

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

```
##    ride_id            started_at
##  Length:3879822     Min.   :2019-04-01 00:02:22.00
##  Class :character   1st Qu.:2019-06-23 07:49:09.25
##  Mode  :character   Median :2019-08-14 17:43:38.00
##                     Mean   :2019-08-26 00:49:59.38
##                     3rd Qu.:2019-10-12 12:10:21.00
##                     Max.   :2020-03-31 23:51:34.00
##
##     ended_at                      rideable_type      start_station_id
##  Min.   :2019-04-01 00:09:48.00   Length:3879822     Min.   :  1.0
##  1st Qu.:2019-06-23 08:20:27.75   Class :character   1st Qu.: 77.0
##  Median :2019-08-14 18:02:04.00   Mode  :character   Median :174.0
##  Mean   :2019-08-26 01:14:37.06                      Mean   :202.9
##  3rd Qu.:2019-10-12 12:36:16.75                      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:3879822     Min.   :  1.0   Length:3879822     Length:3879822
##  Class :character   1st Qu.: 77.0   Class :character   Class :character
##  Mode  :character   Median :174.0   Mode  :character   Mode  :character
##                     Mean   :203.8
##                     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 tw
# (2) The data can only be aggregated at the ride-level, which is too granular. We will want to add som
```

```
# (3) We will want to add a calculated field for length of ride since the 2020Q1 data did not have the
# (4) There are some rides where tripduration shows up as negative, including several hundred rides whe

# 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 d

# N.B.: "Level" is a special property of a column that is retained even if a subset does not contain an
# Begin by seeing how many observations fall under each usertype

table(all_trips$member_casual)
```

```
##
##     casual   Customer     member Subscriber
##      48480     857474     378407    2595461
```

```
# 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", "Custo

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

```
##
##  casual  member
##  905954 2973868
```

```
# 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
# 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 [3,879,822 x 15] (S3: tbl_df/tbl/data.frame)
##  $ ride_id           : chr [1:3879822] "22178529" "22178530" "22178531" "22178532" ...
##  $ started_at        : POSIXct[1:3879822], format: "2019-04-01 00:02:22" "2019-04-01 00:03:02" ...
##  $ ended_at          : POSIXct[1:3879822], format: "2019-04-01 00:09:48" "2019-04-01 00:20:30" ...
##  $ rideable_type     : chr [1:3879822] "6251" "6226" "5649" "4151" ...
##  $ start_station_id  : num [1:3879822] 81 317 283 26 202 420 503 260 211 211 ...
##  $ start_station_name: chr [1:3879822] "Daley Center Plaza" "Wood St & Taylor St" "LaSalle St & Jacks
##  $ end_station_id    : num [1:3879822] 56 59 174 133 129 426 500 499 211 211 ...
##  $ end_station_name  : chr [1:3879822] "Desplaines St & Kinzie St" "Wabash Ave & Roosevelt Rd" "Canal
```

```
##  $ member_casual     : chr [1:3879822] "member" "member" "member" "member" ...
##  $ date              : Date[1:3879822], format: "2019-04-01" "2019-04-01" ...
##  $ month             : chr [1:3879822] "04" "04" "04" "04" ...
##  $ day               : chr [1:3879822] "01" "01" "01" "01" ...
##  $ year              : chr [1:3879822] "2019" "2019" "2019" "2019" ...
##  $ day_of_week       : chr [1:3879822] "Monday" "Monday" "Monday" "Monday" ...
##  $ ride_length       : 'difftime' num [1:3879822] 446 1048 252 357 ...
##   ..- attr(*, "units")= chr "secs"
```

```
# Convert "ride_length" from Factor 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
# The dataframe includes a few hundred entries when bikes were taken out of docks and checked for quali
# 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/
all_trips_v2 <- all_trips[!(all_trips$start_station_name == "HQ QR" | all_trips$ride_length<0),]
dim(all_trips_v2)
```

```
## [1] 3876042      15
```

Analyzing Process

```
# STEP 4: 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] 1479.139
```

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

```
## [1] 712
```

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

```
## [1] 9387024
```

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

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

```
# You can condense the four lines above to one line using summary() on the specific attribute
summary(all_trips_v2$ride_length)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##       1     412     712    1479    1289 9387024
```

```
# 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                3552.7502
## 2                     member                 850.0662
```

```
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                      1546
## 2                     member                       589
```

```
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                   9387024
## 2                     member                   9056634
```

```
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
```

```
# 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                3773.8351
## 2                      member                   Friday                 824.5305
## 3                      casual                   Monday                3372.2869
## 4                      member                   Monday                 842.5726
## 5                      casual                 Saturday                3331.9138
## 6                      member                 Saturday                 968.9337
## 7                      casual                   Sunday                3581.4054
## 8                      member                   Sunday                 919.9746
## 9                      casual                 Thursday                3682.9847
```

```
## 10                    member                Thursday                823.9278
## 11                    casual                 Tuesday               3596.3599
## 12                    member                 Tuesday                826.1427
## 13                    casual               Wednesday               3718.6619
## 14                    member               Wednesday                823.9996
```

```
# 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", "
```

```
# 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               3581.4054
## 2                      member                   Sunday                919.9746
## 3                      casual                   Monday               3372.2869
## 4                      member                   Monday                842.5726
## 5                      casual                  Tuesday               3596.3599
## 6                      member                  Tuesday                826.1427
## 7                      casual                Wednesday               3718.6619
## 8                      member                Wednesday                823.9996
## 9                      casual                 Thursday               3682.9847
## 10                     member                 Thursday                823.9278
## 11                     casual                   Friday               3773.8351
## 12                     member                   Friday                824.5305
## 13                     casual                 Saturday               3331.9138
## 14                     member                 Saturday                968.9337
```

```
# 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              181293            3581.
##  2 casual        Mon              103296            3372.
##  3 casual        Tue               90510            3596.
##  4 casual        Wed               92457            3719.
##  5 casual        Thu              102679            3683.
##  6 casual        Fri              122404            3774.
##  7 casual        Sat              209543            3332.
##  8 member        Sun              267965             920.
##  9 member        Mon              472196             843.
```

```
## 10 member        Tue              508445            826.
## 11 member        Wed              500329            824.
## 12 member        Thu              484177            824.
## 13 member        Fri              452790            825.
## 14 member        Sat              287958            969.
```

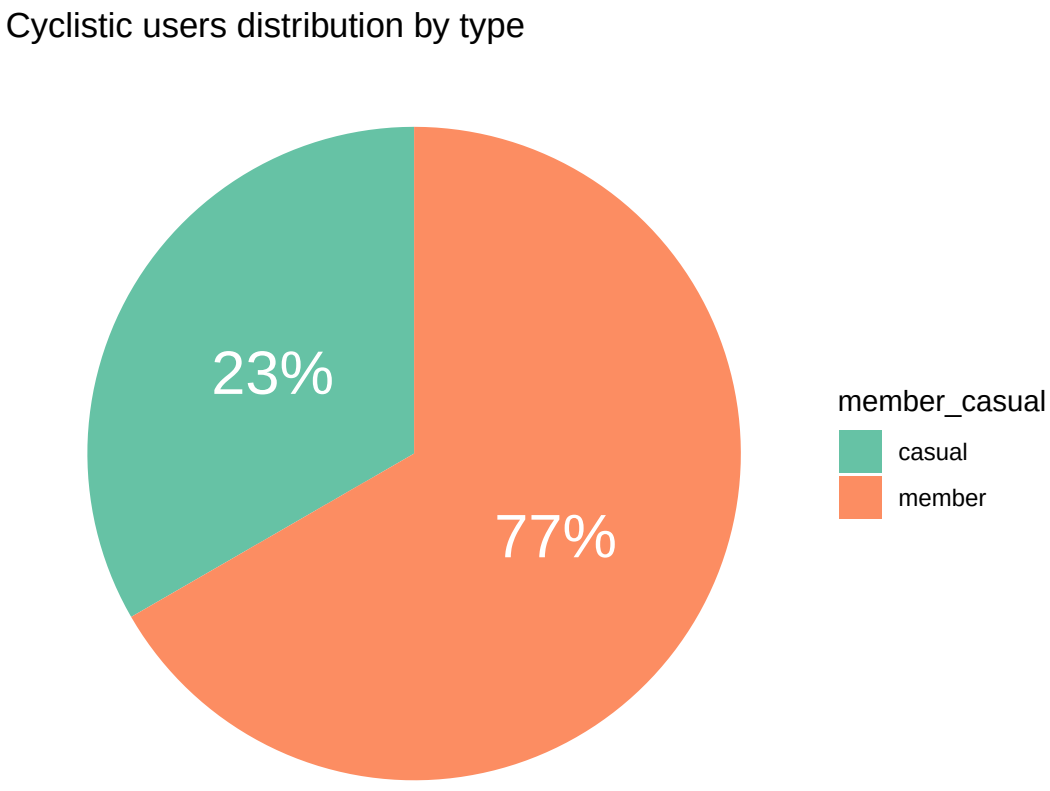
Supporting visualizations and key findings

```
#Transforming  data to create the Viz
cyclistic_users <- all_trips_v2 %>% group_by(member_casual) %>% count() %>%  ungroup() %>% mutate(pcnt =

head(cyclistic_users)
```

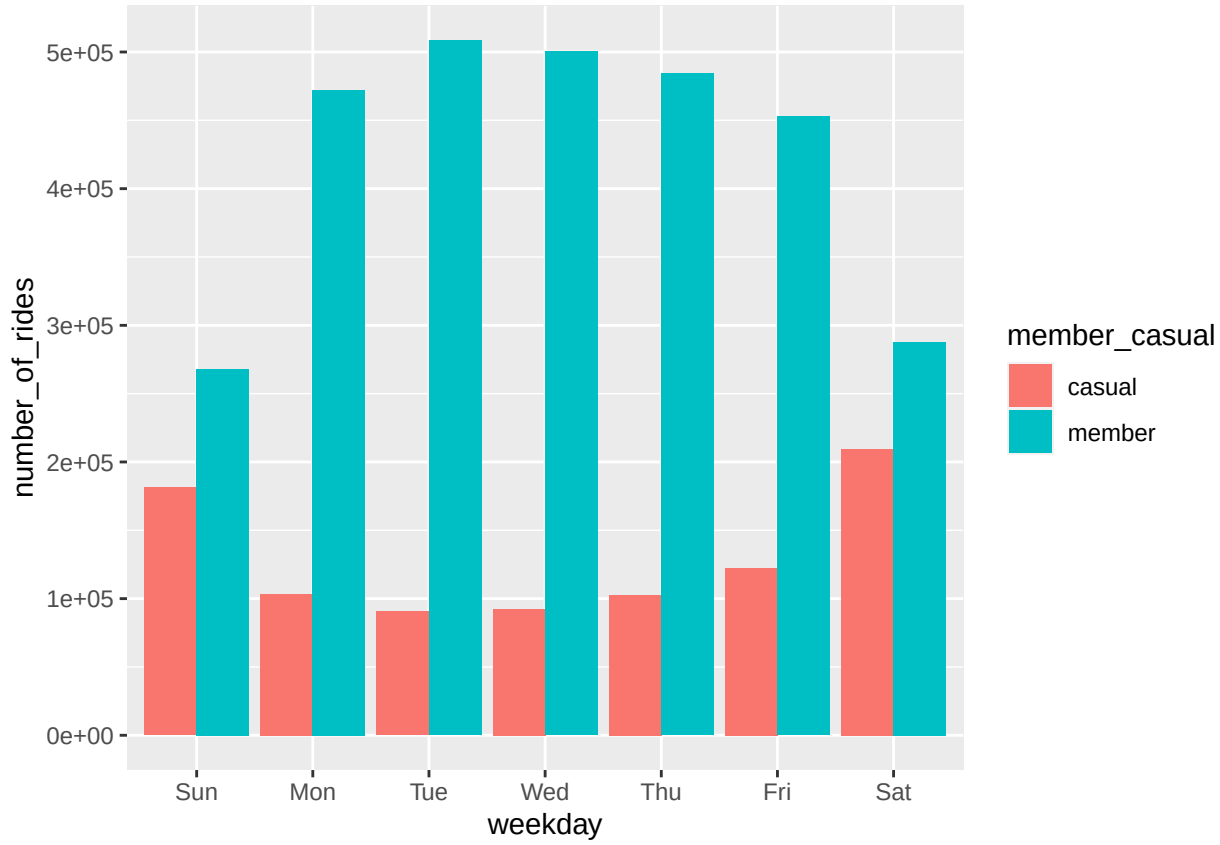
```
## # A tibble: 2 x 4
##   member_casual       n  pcnt labels
##   <chr>           <int> <dbl> <chr>
## 1 casual         902182 0.233 23%
## 2 member        2973860 0.767 77%
```

```
#Visualizing the distribution using a pie chart with ggplot
ggplot(cyclistic_users, aes(x="", y=labels, fill=member_casual)) + geom_col() + geom_text(aes(label = la
```



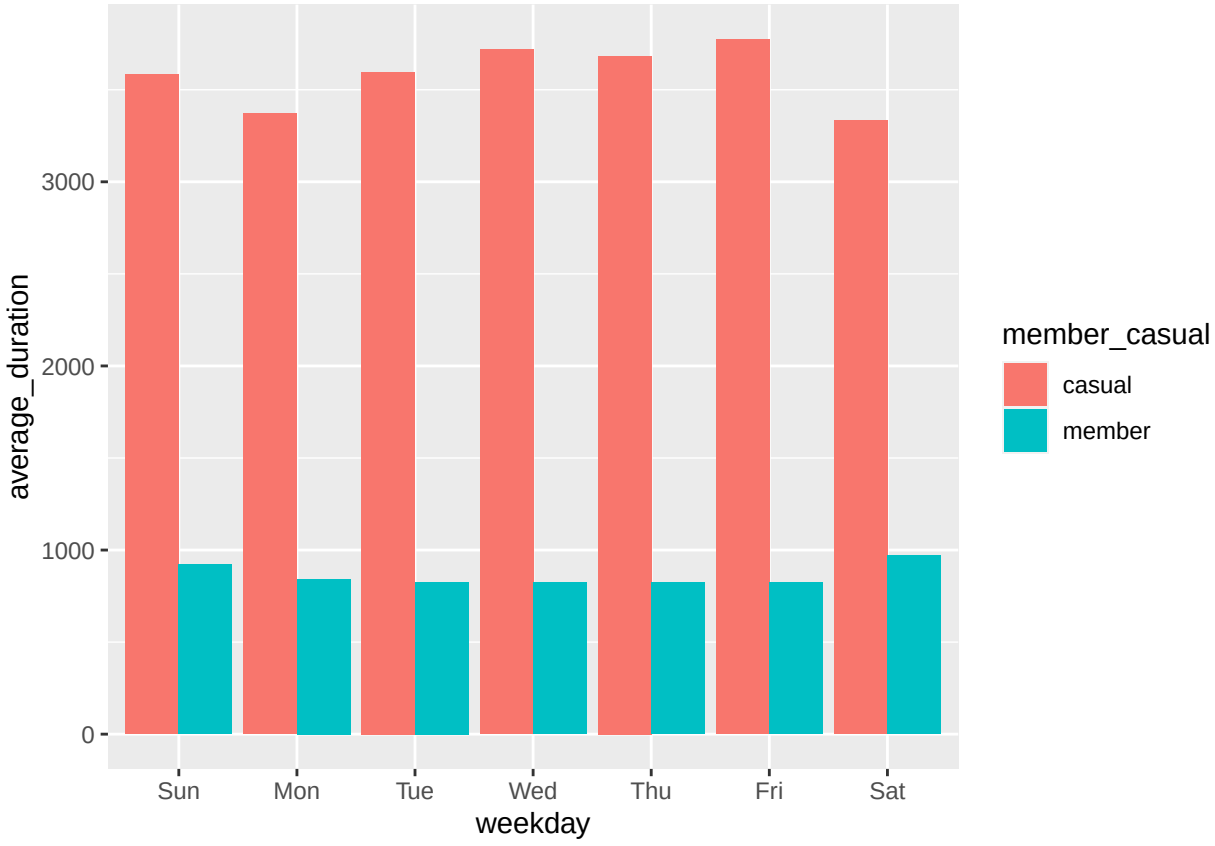
```
# Let's visualize the number of rides by rider type
all_trips_v2 %>% mutate(weekday = wday(started_at, label = TRUE)) %>% group_by(member_casual, weekday) %
```

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




```
# 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")
```

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



Conclusions

During the analysis, we discovered several insights. Only 1% of the data were removed from the analysis including empty data, repeated data, NA data and data used for testing purposes.

- The number of users with membership is 54% higher than the number of casual users.
- Membership cyclists make 5 times more trips than casual cyclists, on weekdays, on weekends casual cyclists increase the number of trips while members reduce their number of trips reaching a ratio of 4:5 (casual:members).
- For the case of trip length, it is appreciated that the average trip time in casual cyclists has a higher ratio of 7:2 with respect to cyclists with membership. Evidencing that casual cyclists make trips of longer duration than member cyclists.

Recommendations

Based on the analysis of the available data and the conclusions obtained, we suggest some recommendations to be taken into account:

- Some type of special short-term subscription can be offered to encourage casual cyclists to become members.

- Survey casual riders to see what incentives would make them become annual members
- Conduct a "zero emissions" campaign through family bike rides.

```
#=============================================
# STEP 5: EXPORT SUMMARY FILE FOR FURTHER ANALYSIS
#=============================================
# Create a csv file that we will visualize in Excel, Tableau, or my presentation software
# N.B.: This file location is for a Mac. If you are working on a PC, change the file location according

counts <- aggregate(all_trips_v2$ride_length ~ all_trips_v2$member_casual + all_trips_v2$day_of_week, FU
write.csv(counts, file = "D:/Data_Science/Data_Analysis_by_Google_2022_julio/Curso 8. Curso Final de Ana
```

Session Info:

```
# printing session information for compatibility
sessionInfo()
```

```
## R version 4.2.1 (2022-06-23 ucrt)
## Platform: x86_64-w64-mingw32/x64 (64-bit)
## Running under: Windows 10 x64 (build 19044)
##
## Matrix products: default
##
## locale:
## [1] LC_COLLATE=English_United Kingdom.utf8
## [2] LC_CTYPE=English_United Kingdom.utf8
## [3] LC_MONETARY=English_United Kingdom.utf8
## [4] LC_NUMERIC=C
## [5] LC_TIME=English_United Kingdom.utf8
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
##  [1] janitor_2.1.0   lubridate_1.8.0 forcats_0.5.1   stringr_1.4.0
##  [5] dplyr_1.0.9     purrr_0.3.4     readr_2.1.2     tidyr_1.2.0
##  [9] tibble_3.1.7    ggplot2_3.3.6   tidyverse_1.3.1
##
## loaded via a namespace (and not attached):
##  [1] assertthat_0.2.1   digest_0.6.29      utf8_1.2.2         R6_2.5.1
##  [5] cellranger_1.1.0   backports_1.4.1    reprex_2.0.1       evaluate_0.15
##  [9] httr_1.4.3         highr_0.9          pillar_1.7.0       rlang_1.0.3
## [13] readxl_1.4.0       rstudioapi_0.13    rmarkdown_2.17     labeling_0.4.2
## [17] bit_4.0.4          munsell_0.5.0      broom_1.0.0        compiler_4.2.1
## [21] modelr_0.1.8       xfun_0.31          pkgconfig_2.0.3    htmltools_0.5.2
## [25] tidyselect_1.1.2   fansi_1.0.3        crayon_1.5.1       tzdb_0.3.0
## [29] dbplyr_2.2.1       withr_2.5.0        grid_4.2.1         jsonlite_1.8.0
## [33] gtable_0.3.0       lifecycle_1.0.1    DBI_1.1.3          magrittr_2.0.3
## [37] scales_1.2.0       cli_3.3.0          stringi_1.7.6      vroom_1.5.7
## [41] farver_2.1.0       fs_1.5.2           snakecase_0.11.0   xml2_1.3.3
```

```
## [45] ellipsis_0.3.2     generics_0.1.3     vctrs_0.4.1        RColorBrewer_1.1-3
## [49] tools_4.2.1        bit64_4.0.5        glue_1.6.2         hms_1.1.1
## [53] parallel_4.2.1     fastmap_1.1.0      yaml_2.3.5         colorspace_2.0-3
## [57] rvest_1.0.2        knitr_1.40         haven_2.5.0
```