Google Data Analytics Professional Certificate Capstone Project: Cyclistic - Bike-share company

Filipe Balseiro

23/01/2022

## Case Study: Help a bike-share company to convert casual riders into annual members

In this article i showcase my approach to solve the case study of Google Data Analytics Professional Certificate Capstone Project.

As I learned from the Google Data Analytics program, I will the six phases of the data analysis process: **ask**, **prepare**, **process**, **analyze**, **share and act**.

### About the Company

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.

The director of marketing believes the company’s future success depends on maximizing the number of annual memberships. Therefore, team wants to understand how casual riders and annual members use Cyclistic bikes differently.

### Business Task

To answer this problem: **How do annual members and casual riders use Cyclistic bikes differenty?**

### Ask

Three questions need to be answered:

1. How do annual members and casual riders use Cyclistic bikes differently?
2. Why would casual riders buy Cyclistic annual memberships?
3. How can Cyclistic use digital media to influence casual riders to become members?

### Prepare

In this phase, I collected the data organized in monthly .csv files from this source [here](https://divvy-tripdata.s3.amazonaws.com/index.html). I downloaded the files from 2021 to perform this analysis.

#### Are there issues with bias or credibility in this data?

The data is from last year and is collected by a bike-share company directly. It includes all rides, so it’s not a sample from the whole data. Therefore, it’s possible to conclude that there are no credibility and bias issues with this data.

#### How are you addressing licensing, privacy, security, and accessibility?

The data is under this [license](https://ride.divvybikes.com/data-license-agreement). There are no privacy concerns since the data does not contain personal information.

### Process

In this phase I processed the data and get it ready for the next phase where I will look for insights that help me answer to our stakeholders questions and business task. I used R to perform this step since the data is too big to merge and process (5.595.063 records).

I started by importing the packages needed to this task.

library(tidyverse) #helps wrangle data

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

## v ggplot2 3.3.5 v purrr 0.3.4  
## v tibble 3.1.6 v dplyr 1.0.7  
## v tidyr 1.1.4 v stringr 1.4.0  
## v readr 2.1.1 v forcats 0.5.1

## -- Conflicts ------------------------------------------ tidyverse\_conflicts() --  
## x dplyr::filter() masks stats::filter()  
## x dplyr::lag() masks stats::lag()

library(lubridate) #helps wrangle date attributes

##   
## Attaching package: 'lubridate'

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

Next, I collected the data from the 12 months of 2021 and combine to a single file.

# Upload Cyclistic 2021 datasets (csv files) here  
  
m01\_2021 <- read\_csv("202101-divvy-tripdata.csv")

## Rows: 96834 Columns: 13

## -- Column specification --------------------------------------------------------  
## Delimiter: ","  
## chr (7): ride\_id, rideable\_type, start\_station\_name, start\_station\_id, end\_...  
## dbl (4): start\_lat, start\_lng, end\_lat, end\_lng  
## dttm (2): started\_at, ended\_at

##   
## i Use `spec()` to retrieve the full column specification for this data.  
## i Specify the column types or set `show\_col\_types = FALSE` to quiet this message.

m02\_2021 <- read\_csv("202102-divvy-tripdata.csv")

## Rows: 49622 Columns: 13

## -- Column specification --------------------------------------------------------  
## Delimiter: ","  
## chr (7): ride\_id, rideable\_type, start\_station\_name, start\_station\_id, end\_...  
## dbl (4): start\_lat, start\_lng, end\_lat, end\_lng  
## dttm (2): started\_at, ended\_at

##   
## i Use `spec()` to retrieve the full column specification for this data.  
## i Specify the column types or set `show\_col\_types = FALSE` to quiet this message.

m03\_2021 <- read\_csv("202103-divvy-tripdata.csv")

## Rows: 228496 Columns: 13

## -- Column specification --------------------------------------------------------  
## Delimiter: ","  
## chr (7): ride\_id, rideable\_type, start\_station\_name, start\_station\_id, end\_...  
## dbl (4): start\_lat, start\_lng, end\_lat, end\_lng  
## dttm (2): started\_at, ended\_at

##   
## i Use `spec()` to retrieve the full column specification for this data.  
## i Specify the column types or set `show\_col\_types = FALSE` to quiet this message.

m04\_2021 <- read\_csv("202104-divvy-tripdata.csv")

## Rows: 337230 Columns: 13

## -- Column specification --------------------------------------------------------  
## Delimiter: ","  
## chr (7): ride\_id, rideable\_type, start\_station\_name, start\_station\_id, end\_...  
## dbl (4): start\_lat, start\_lng, end\_lat, end\_lng  
## dttm (2): started\_at, ended\_at

##   
## i Use `spec()` to retrieve the full column specification for this data.  
## i Specify the column types or set `show\_col\_types = FALSE` to quiet this message.

m05\_2021 <- read\_csv("202105-divvy-tripdata.csv")

## Rows: 531633 Columns: 13

## -- Column specification --------------------------------------------------------  
## Delimiter: ","  
## chr (7): ride\_id, rideable\_type, start\_station\_name, start\_station\_id, end\_...  
## dbl (4): start\_lat, start\_lng, end\_lat, end\_lng  
## dttm (2): started\_at, ended\_at

##   
## i Use `spec()` to retrieve the full column specification for this data.  
## i Specify the column types or set `show\_col\_types = FALSE` to quiet this message.

m06\_2021 <- read\_csv("202106-divvy-tripdata.csv")

## Rows: 729595 Columns: 13

## -- Column specification --------------------------------------------------------  
## Delimiter: ","  
## chr (7): ride\_id, rideable\_type, start\_station\_name, start\_station\_id, end\_...  
## dbl (4): start\_lat, start\_lng, end\_lat, end\_lng  
## dttm (2): started\_at, ended\_at

##   
## i Use `spec()` to retrieve the full column specification for this data.  
## i Specify the column types or set `show\_col\_types = FALSE` to quiet this message.

m07\_2021 <- read\_csv("202107-divvy-tripdata.csv")

## Rows: 822410 Columns: 13

## -- Column specification --------------------------------------------------------  
## Delimiter: ","  
## chr (7): ride\_id, rideable\_type, start\_station\_name, start\_station\_id, end\_...  
## dbl (4): start\_lat, start\_lng, end\_lat, end\_lng  
## dttm (2): started\_at, ended\_at

##   
## i Use `spec()` to retrieve the full column specification for this data.  
## i Specify the column types or set `show\_col\_types = FALSE` to quiet this message.

m08\_2021 <- read\_csv("202108-divvy-tripdata.csv")

## Rows: 804352 Columns: 13

## -- Column specification --------------------------------------------------------  
## Delimiter: ","  
## chr (7): ride\_id, rideable\_type, start\_station\_name, start\_station\_id, end\_...  
## dbl (4): start\_lat, start\_lng, end\_lat, end\_lng  
## dttm (2): started\_at, ended\_at

##   
## i Use `spec()` to retrieve the full column specification for this data.  
## i Specify the column types or set `show\_col\_types = FALSE` to quiet this message.

m09\_2021 <- read\_csv("202109-divvy-tripdata.csv")

## Rows: 756147 Columns: 13

## -- Column specification --------------------------------------------------------  
## Delimiter: ","  
## chr (7): ride\_id, rideable\_type, start\_station\_name, start\_station\_id, end\_...  
## dbl (4): start\_lat, start\_lng, end\_lat, end\_lng  
## dttm (2): started\_at, ended\_at

##   
## i Use `spec()` to retrieve the full column specification for this data.  
## i Specify the column types or set `show\_col\_types = FALSE` to quiet this message.

m10\_2021 <- read\_csv("202110-divvy-tripdata.csv")

## Rows: 631226 Columns: 13

## -- Column specification --------------------------------------------------------  
## Delimiter: ","  
## chr (7): ride\_id, rideable\_type, start\_station\_name, start\_station\_id, end\_...  
## dbl (4): start\_lat, start\_lng, end\_lat, end\_lng  
## dttm (2): started\_at, ended\_at

##   
## i Use `spec()` to retrieve the full column specification for this data.  
## i Specify the column types or set `show\_col\_types = FALSE` to quiet this message.

m11\_2021 <- read\_csv("202111-divvy-tripdata.csv")

## Rows: 359978 Columns: 13

## -- Column specification --------------------------------------------------------  
## Delimiter: ","  
## chr (7): ride\_id, rideable\_type, start\_station\_name, start\_station\_id, end\_...  
## dbl (4): start\_lat, start\_lng, end\_lat, end\_lng  
## dttm (2): started\_at, ended\_at

##   
## i Use `spec()` to retrieve the full column specification for this data.  
## i Specify the column types or set `show\_col\_types = FALSE` to quiet this message.

m12\_2021 <- read\_csv("202112-divvy-tripdata.csv")

## Rows: 247540 Columns: 13

## -- Column specification --------------------------------------------------------  
## Delimiter: ","  
## chr (7): ride\_id, rideable\_type, start\_station\_name, start\_station\_id, end\_...  
## dbl (4): start\_lat, start\_lng, end\_lat, end\_lng  
## dttm (2): started\_at, ended\_at

##   
## i Use `spec()` to retrieve the full column specification for this data.  
## i Specify the column types or set `show\_col\_types = FALSE` to quiet this message.

# Stack individual monthly's data frames into one big data frame  
all\_trips <- bind\_rows(m01\_2021, m02\_2021, m03\_2021, m04\_2021,  
 m05\_2021, m06\_2021, m07\_2021, m08\_2021,  
 m09\_2021, m10\_2021, m11\_2021, m12\_2021)

Let’s start to observe the number of rows and columns

dim(all\_trips)

## [1] 5595063 13

Let’s remove the columns that won’t be used in the analysis process: latitude, longitude, start\_station\_id and end\_station\_id.

#remove the latitude, longitude, start\_station\_id and end\_station\_id as these are not required for analysis  
all\_trips <- all\_trips %>%  
 select(-c(start\_lat, start\_lng, end\_lat, end\_lng, start\_station\_id, end\_station\_id))

Next, I renamed the column member\_casual to a more suitable name.

#rename column member\_casual to rider\_type  
all\_trips <- all\_trips %>%  
 rename(rider\_type = member\_casual)

I created new columns that list the date, year, month, day and start hour of each ride.

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$day\_of\_week <- c("Sunday", "Monday", "Tuesday", "Wednesday", "Thursday",   
 "Friday", "Saturday")[as.POSIXlt(all\_trips$date)$wday + 1]  
all\_trips$hour\_of\_day <- format(as.POSIXct(all\_trips$started\_at), format = "%H")

Added a new column called ride\_length to calculate each trip (in minutes).

all\_trips$ride\_length <- difftime(all\_trips$ended\_at,all\_trips$started\_at, units = "mins")

I sorted the dataset based on ride\_length to check if there any errors, like negative values.

# sort the dataset based on ride\_length value to check if there are any errors  
all\_trips %>%  
 arrange(ride\_length) %>%  
 select(ride\_id, started\_at, ended\_at, ride\_length) %>%  
 filter(ride\_length < 0)

## # A tibble: 147 x 4  
## ride\_id started\_at ended\_at ride\_length   
## <chr> <dttm> <dttm> <drtn>   
## 1 7CA158F5F050156E 2021-11-07 01:58:08 2021-11-07 01:00:06 -58.03333 mins  
## 2 FD8AF7324ABAE9DA 2021-11-07 01:56:51 2021-11-07 01:00:57 -55.90000 mins  
## 3 508B09A5FB0737DC 2021-11-07 01:54:50 2021-11-07 01:00:45 -54.08333 mins  
## 4 6F9E76F5EDAAC1B8 2021-11-07 01:55:42 2021-11-07 01:01:55 -53.78333 mins  
## 5 7AECC76D1562B51C 2021-11-07 01:54:58 2021-11-07 01:01:29 -53.48333 mins  
## 6 B506DCD44974C575 2021-11-07 01:53:34 2021-11-07 01:00:42 -52.86667 mins  
## 7 CDB307B8494885AD 2021-11-07 01:55:09 2021-11-07 01:02:26 -52.71667 mins  
## 8 FFD5A2DDE1FAAA90 2021-11-07 01:54:36 2021-11-07 01:03:11 -51.41667 mins  
## 9 7E24361D78747AF6 2021-11-07 01:58:06 2021-11-07 01:06:43 -51.38333 mins  
## 10 53222CFE6657D53D 2021-11-07 01:52:22 2021-11-07 01:01:29 -50.88333 mins  
## # ... with 137 more rows

The wrong data was removed from the dataset.

all\_trips\_v2 <- all\_trips[!(all\_trips$ride\_length<0),]  
nrow(all\_trips\_v2)

## [1] 5594916

As we can see below dataset as a considerable amount of NA values in **start\_station\_name** and **end\_station\_name** columns.

# check for NA values in start\_station\_name and end\_station\_name columns  
all\_trips\_v2 %>%   
 group\_by(start\_station\_name) %>%  
 summarise(number\_of\_rides = n()) %>%  
 arrange(-number\_of\_rides)

## # A tibble: 848 x 2  
## start\_station\_name number\_of\_rides  
## <chr> <int>  
## 1 <NA> 690789  
## 2 Streeter Dr & Grand Ave 82714  
## 3 Michigan Ave & Oak St 44347  
## 4 Wells St & Concord Ln 43609  
## 5 Millennium Park 42223  
## 6 Clark St & Elm St 41217  
## 7 Wells St & Elm St 37690  
## 8 Theater on the Lake 36840  
## 9 Kingsbury St & Kinzie St 33581  
## 10 Clark St & Lincoln Ave 33382  
## # ... with 838 more rows

all\_trips\_v2 %>%   
 group\_by(end\_station\_name) %>%  
 summarise(number\_of\_rides = n()) %>%  
 arrange(-number\_of\_rides)

## # A tibble: 845 x 2  
## end\_station\_name number\_of\_rides  
## <chr> <int>  
## 1 <NA> 739149  
## 2 Streeter Dr & Grand Ave 83389  
## 3 Michigan Ave & Oak St 44833  
## 4 Wells St & Concord Ln 43850  
## 5 Millennium Park 42933  
## 6 Clark St & Elm St 40530  
## 7 Wells St & Elm St 37348  
## 8 Theater on the Lake 37046  
## 9 Clark St & Lincoln Ave 33295  
## 10 Wabash Ave & Grand Ave 33132  
## # ... with 835 more rows

Instead of removing the records with NA values, I assigned all NA values as “Missing Data” so that I can analyze missing values as well.

# replace NA values in start\_station\_name and end\_station\_name columns with Missing Data values  
all\_trips\_v2$start\_station\_name <-  
 replace(all\_trips\_v2$start\_station\_name, is.na(all\_trips\_v2$start\_station\_name), "Missing Data")  
all\_trips\_v2$end\_station\_name <-  
 replace(all\_trips\_v2$end\_station\_name, is.na(all\_trips\_v2$end\_station\_name), "Missing Data")

Finally I exported the dataset as a csv file to analyze in Tableau.

# Create a csv file that we will visualize in Tableau  
counts <- write.csv(all\_trips\_v2, file = 'all\_trips.csv')

### Tableau Dashboard

[Link](https://public.tableau.com/app/profile/filipe7270/viz/Cyclistic_16429571092230/StoryBike-Sharing)