**Case Study Roadmap**

**- Ask Guiding questions**

**● What is the problem you are trying to solve?**

Design marketing strategies aimed at converting casual riders into annual members

**● How can your insights drive business decisions?**

Using historical data for past 12 months from June 2020 to May 2021 to understand what annual members and casual riders use Cyclistic bikes differently

Key tasks

Identify the business task

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?

2. Consider key stakeholders

Director of Marketing Lily Moreno, Cyclistic marketing analytics team, and Cyclistic executive team

Deliverable

A clear statement of the business task

**PREPARE**

**● Where is your data located?**

The data collected is from <https://divvy-tripdata.s3.amazonaws.com/index.html>

**2.How is the data organized?**

The data is considered structured data because is organized in a certain format, like rows and columns.

**● Are there issues with bias or credibility in this data? Does your data ROCCC?**

Data has been downloaded from [Motivate International Inc.](https://divvy-tripdata.s3.amazonaws.com/index.html) Local copies have been stored securely on Google Drive and here on Kaggle.

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

The data has been made available by Motivate International Inc. under this <https://www.divvybikes.com/data-license-agreement>

● How did you verify the data’s integrity?

This is public data that you can use to explore how different customer types are using Cyclistic bikes. We are going to assume the data is credible.

How does it help you answer your question?

●Are there any problems with the data?

**3. Sort and filter the data.**

I filtered and use the “find and select” > “Go to special” > “Blanks” allowed to delete blanks row. If I found blanks row in any column, I erased the whole row because the data is public and could not find the reason of why the data was empty.

Link for erasing blanks rows and steps

<https://www.howtoexcel.org/delete-blank-rows/>

**Process**

**Key tasks**

* **Choose your tools**

R for cleaning, analysis and data visualization

* **Check the data for errors**

The null values were eliminate in the spreadsheets by filtering and “find and select.”

* **Transform the data into the right type**
  + - The datasets or data frames from tripdata\_202006 to tripdata\_202011 the started\_at and ended\_at were considered character instead of datetime, so I changed in there too.
    - Additionally, start\_station\_id and end\_station\_id from 202006 to 202011 were consider double or numeric instead of characters; and 202012 to 202105 were consider data type characters already.
* **Document the cleaning process**

**Installing the package tidy verse and loading**

install.packages("tidyverse")

library(tidyverse)

> library(tidyverse)

── Attaching packages ───────────────── tidyverse 1.3.2 ──

✔ ggplot2 3.3.6 ✔ purrr 0.3.4

✔ tibble 3.1.8 ✔ dplyr 1.0.9

✔ tidyr 1.2.0 ✔ stringr 1.4.0

✔ readr 2.1.2 ✔ forcats 0.5.1

── Conflicts ──────────────────── tidyverse\_conflicts() ──

✖ dplyr::filter() masks stats::filter()

✖ dplyr::lag() masks stats::lag()

>**install.packages("janitor")**

**> library(janitor)**

Attaching package: ‘janitor’

The following objects are masked from ‘package:stats’:

chisq.test, fisher.test

**> install.packages(lubridate)**

**Library(lubridate)**

The following objects are masked from ‘package:base’:

date, intersect, setdiff, union

**library (dplyr)**

The following objects are masked from ‘package:data.table’:

between, first, last

The following objects are masked from ‘package:stats’:

filter, lag

The following objects are masked from ‘package:base’:

intersect, setdiff, setequal, union

install.packages("here")

library(here)

library(scales)

Attaching package: ‘scales’

The following object is masked from ‘package:purrr’:

discard

The following object is masked from ‘package:readr’:

col\_factor

2.1 **Loading datasets using readr**

tripdata\_202006 <- read\_csv("Bike data/202006-divvy-tripdata.csv",

+ col\_types = cols(started\_at = col\_datetime(format = "%m/%d/%Y %H:%M"),

+ ended\_at = col\_datetime(format = "%m/%d/%Y %H:%M"),

+ start\_station\_id = col\_character(),

+ end\_station\_id = col\_character()))

> View(tripdata\_202006) july\_2020 <- read\_csv("Bike data 12 months/202007-divvy-tripdata/202007-divvy-tripdata.csv")

> tripdata\_202007 <- read\_csv("Bike data/202007-divvy-tripdata.csv",

+ col\_types = cols(started\_at = col\_datetime(format = "%m/%d/%Y %H:%M"),

+ ended\_at = col\_datetime(format = "%m/%d/%Y %H:%M"),

+ start\_station\_id = col\_character(),

+ end\_station\_id = col\_character()))

> View(tripdata\_202007)

> tripdata\_202008 <- read\_csv("Bike data/202008-divvy-tripdata.csv",

+ col\_types = cols(started\_at = col\_datetime(format = "%m/%d/%Y %H:%M"),

+ ended\_at = col\_datetime(format = "%m/%d/%Y %H:%M"),

+ start\_station\_id = col\_character(),

+ end\_station\_id = col\_character()))

> View(tripdata\_202008)

> tripdata\_202009 <- read\_csv("Bike data/202009-divvy-tripdata.csv",

+ col\_types = cols(started\_at = col\_datetime(format = "%m/%d/%Y %H:%M"),

+ ended\_at = col\_datetime(format = "%m/%d/%Y %H:%M"),

+ start\_station\_id = col\_character(),

+ end\_station\_id = col\_character()))

> View(tripdata\_202009)

> tripdata\_202010 <- read\_csv("Bike data/202010-divvy-tripdata.csv",

+ col\_types = cols(started\_at = col\_datetime(format = "%m/%d/%Y %H:%M"),

+ ended\_at = col\_datetime(format = "%m/%d/%Y %H:%M"),

+ start\_station\_id = col\_character(),

+ end\_station\_id = col\_character()))

> View(tripdata\_202010)

> tripdata\_202011 <- read\_csv("Bike data/202011-divvy-tripdata.csv",

+ col\_types = cols(started\_at = col\_datetime(format = "%m/%d/%Y %H:%M"),

+ ended\_at = col\_datetime(format = "%m/%d/%Y %H:%M"),

+ start\_station\_id = col\_character(),

+ end\_station\_id = col\_character()))

> View(tripdata\_202011)

> tripdata\_202012 <- read\_csv("Bike data/202012-divvy-tripdata.csv",

+ col\_types = cols(started\_at = col\_datetime(format = "%m/%d/%Y %H:%M"),

+ ended\_at = col\_datetime(format = "%m/%d/%Y %H:%M")))

> View(tripdata\_202012)

> tripdata\_202101 <- read\_csv("Bike data/202101-divvy-tripdata.csv",

+ col\_types = cols(started\_at = col\_datetime(format = "%m/%d/%Y %H:%M"),

+ ended\_at = col\_datetime(format = "%m/%d/%Y %H:%M")))

> View(tripdata\_202101)

> tripdata\_202102 <- read\_csv("Bike data/202102-divvy-tripdata.csv",

+ col\_types = cols(started\_at = col\_datetime(format = "%m/%d/%Y %H:%M"),

+ ended\_at = col\_datetime(format = "%m/%d/%Y %H:%M")))

> View(tripdata\_202102)

> tripdata\_202103 <- read\_csv("Bike data/202103-divvy-tripdata.csv",

+ col\_types = cols(started\_at = col\_datetime(format = "%m/%d/%Y %H:%M"),

+ ended\_at = col\_datetime(format = "%m/%d/%Y %H:%M")))

> View(tripdata\_202103)

> tripdata\_202104 <- read\_csv("Bike data/202104-divvy-tripdata.csv",

+ col\_types = cols(started\_at = col\_datetime(format = "%m/%d/%Y %H:%M"),

+ ended\_at = col\_datetime(format = "%m/%d/%Y %H:%M")))

> View(tripdata\_202104)

> tripdata\_202105 <- read\_csv("Bike data/202105-divvy-tripdata.csv",

+ col\_types = cols(started\_at = col\_datetime(format = "%m/%d/%Y %H:%M"),

+ ended\_at = col\_datetime(format = "%m/%d/%Y %H:%M")))

> View(tripdata\_202105)

*2.2 ## Combining each data frame to get twelve months of data bike*

3.1 combined\_databike <- bind\_rows(tripdata\_202006,tripdata\_202007,tripdata\_202008,tripdata\_202009,

tripdata\_202010,tripdata\_202011,tripdata\_202012,tripdata\_202101,

tripdata\_202102,tripdata\_202103,tripdata\_202104,tripdata\_202105)

View(combined\_databike)

*> str(combined\_databike)*

spec\_tbl\_df [3,745,465 × 13] (S3: spec\_tbl\_df/tbl\_df/tbl/data.frame)

$ ride\_id : chr [1:3745465] "8CD5DE2C2B6C4CFC" "9A191EB2C751D85D" "F37D14B0B5659BCF" "C41237B506E85FA1" ...

$ rideable\_type : chr [1:3745465] "docked\_bike" "docked\_bike" "docked\_bike" "docked\_bike" ...

$ started\_at : POSIXct[1:3745465], format: "2020-06-13 23:24:00" "2020-06-26 07:26:00" "2020-06-23 17:12:00" "2020-06-20 01:09:00" ...

$ ended\_at : POSIXct[1:3745465], format: "2020-06-13 23:36:00" "2020-06-26 07:31:00" "2020-06-23 17:21:00" "2020-06-20 01:28:00" ...

$ start\_station\_name: chr [1:3745465] "Wilton Ave & Belmont Ave" "Federal St & Polk St" "Daley Center Plaza" "Broadway & Cornelia Ave" ...

$ start\_station\_id : chr [1:3745465] "117" "41" "81" "303" ...

$ end\_station\_name : chr [1:3745465] "Damen Ave & Clybourn Ave" "Daley Center Plaza" "State St & Harrison St" "Broadway & Berwyn Ave" ...

$ end\_station\_id : chr [1:3745465] "163" "81" "5" "294" ...

$ start\_lat : num [1:3745465] 41.9 41.9 41.9 41.9 41.9 ...

$ start\_lng : num [1:3745465] -87.7 -87.6 -87.6 -87.6 -87.7 ...

$ end\_lat : num [1:3745465] 41.9 41.9 41.9 42 41.9 ...

$ end\_lng : num [1:3745465] -87.7 -87.6 -87.6 -87.7 -87.7 ...

$ member\_casual : chr [1:3745465] "casual" "member" "member" "casual" ...

- attr(\*, "spec")=

.. cols(

.. ride\_id = col\_character(),

.. rideable\_type = col\_character(),

.. started\_at = col\_datetime(format = "%m/%d/%Y %H:%M"),

.. ended\_at = col\_datetime(format = "%m/%d/%Y %H:%M"),

.. start\_station\_name = col\_character(),

.. start\_station\_id = col\_character(),

.. end\_station\_name = col\_character(),

.. end\_station\_id = col\_character(),

.. start\_lat = col\_double(),

.. start\_lng = col\_double(),

.. end\_lat = col\_double(),

.. end\_lng = col\_double(),

.. member\_casual = col\_character()

.. )

- attr(\*, "problems")=<externalptr>

After installing the “here” package and loaded

I used

*3.2 glimpse(combined\_databike)*

Rows: 3,745,465

Columns: 13

$ ride\_id <chr> "8CD5DE2C2B6C4CFC", "9A191EB2C751D85D", "F37D14B0B5659BCF", "C41237B506E85FA1", "4B51B3B0BDA…

$ rideable\_type <chr> "docked\_bike", "docked\_bike", "docked\_bike", "docked\_bike", "docked\_bike", "docked\_bike", "d…

$ started\_at <dttm> 2020-06-13 23:24:00, 2020-06-26 07:26:00, 2020-06-23 17:12:00, 2020-06-20 01:09:00, 2020-06…

$ ended\_at <dttm> 2020-06-13 23:36:00, 2020-06-26 07:31:00, 2020-06-23 17:21:00, 2020-06-20 01:28:00, 2020-06…

$ start\_station\_name <chr> "Wilton Ave & Belmont Ave", "Federal St & Polk St", "Daley Center Plaza", "Broadway & Cornel…

$ start\_station\_id <chr> "117", "41", "81", "303", "327", "327", "41", "115", "338", "84", "317", "225", "303", "126"…

$ end\_station\_name <chr> "Damen Ave & Clybourn Ave", "Daley Center Plaza", "State St & Harrison St", "Broadway & Berw…

$ end\_station\_id <chr> "163", "81", "5", "294", "117", "117", "81", "303", "164", "53", "168", "160", "324", "627",…

$ start\_lat <dbl> 41.94018, 41.87208, 41.88424, 41.94553, 41.92154, 41.92154, 41.87208, 41.93627, 41.85761, 41…

$ start\_lng <dbl> -87.65304, -87.62954, -87.62963, -87.64644, -87.65382, -87.65382, -87.62954, -87.65266, -87.…

$ end\_lat <dbl> 41.93193, 41.88424, 41.87405, 41.97835, 41.94018, 41.94018, 41.88424, 41.94553, 41.88584, 41…

$ end\_lng <dbl> -87.67786, -87.62963, -87.62772, -87.65975, -87.65304, -87.65304, -87.62963, -87.64644, -87.…

$ member\_casual <chr> "casual", "member", "member", "casual", "casual", "casual", "member", "casual", "member", "m…

Then

3.3 --

> head(combined\_databike)

# A tibble: 6 × 13

ride\_id ridea…¹ started\_at ended\_at start…² start…³ end\_s…⁴ end\_s…⁵ start…⁶ start…⁷ end\_lat end\_lng

<chr> <chr> <dttm> <dttm> <chr> <chr> <chr> <chr> <dbl> <dbl> <dbl> <dbl>

1 8CD5DE… docked… 2020-06-13 23:24:00 2020-06-13 23:36:00 Wilton… 117 Damen … 163 41.9 -87.7 41.9 -87.7

2 9A191E… docked… 2020-06-26 07:26:00 2020-06-26 07:31:00 Federa… 41 Daley … 81 41.9 -87.6 41.9 -87.6

3 F37D14… docked… 2020-06-23 17:12:00 2020-06-23 17:21:00 Daley … 81 State … 5 41.9 -87.6 41.9 -87.6

4 C41237… docked… 2020-06-20 01:09:00 2020-06-20 01:28:00 Broadw… 303 Broadw… 294 41.9 -87.6 42.0 -87.7

5 4B51B3… docked… 2020-06-25 16:59:00 2020-06-25 17:08:00 Sheffi… 327 Wilton… 117 41.9 -87.7 41.9 -87.7

6 D50DF2… docked… 2020-06-17 18:07:00 2020-06-17 18:18:00 Sheffi… 327 Wilton… 117 41.9 -87.7 41.9 -87.7

# … with 1 more variable: member\_casual <chr>, and abbreviated variable names ¹​rideable\_type, ²​start\_station\_name,

# ³​start\_station\_id, ⁴​end\_station\_name, ⁵​end\_station\_id, ⁶​start\_lat, ⁷​start\_lng

# ℹ Use `colnames()` to see all variable names

**Process**

#======================================================

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

#======================================================

**Check the data for errors**

Originally all dataframes tripdata\_202006 to tripdata\_202105 started\_at and ended\_at were considered as a character instead of datetime. I did the changes using readr().

Empty rows or columns were cleaned back excel. But only to add extra step of cleaning, I used the janitor to remove empty columns and rows.

**Choose your tools.**

Janitor, Here,Dplyr

**Transform the data so you can work with it effectively.**

Started\_at and ended\_at have date and time. I focused in creating new columns for date, month, day, year, and day\_week.

**Document the cleaning process.**

# Using Janitor to remove empty spaces

combined\_databike <- janitor::remove\_empty(combined\_databike, which = c("cols"))

combined\_databike <- janitor::remove\_empty(combined\_databike, which = c("rows"))

dim(combined\_databike) # to check if there was any changes after checking for empty spaces

I already took the null spaces back with spreadsheets but wanted to make sure it was empty.

## Remove columns not required or beyond the scope of project

combined\_databike <- combined\_databike %>%

select(-start\_station\_id,-end\_station\_id) %>%

combined\_databike <- combined\_databike %>%

select(-start\_lat,-start\_lng,-end\_lat,-end\_lng)

## More data cleaning

glimpse(combined\_databike) ## With glimpse I have 3,745,465 rows and 13 columns

colnames(combined\_databike) #List of column names

[1] "ride\_id" "rideable\_type" "started\_at" "ended\_at"

[5] "start\_station\_name" "end\_station\_name" "member\_casual"

nrow(combined\_databike) #How many rows are in data frame?

3745465

dim(combined\_databike) #Dimensions of the data frame?

3745465 7

head(combined\_databike) #See the first 6 rows of data frame.

# A tibble: 6 × 7

ride\_id rideable\_type started\_at ended\_at start\_station\_name end\_s…¹ membe…²

<chr> <chr> <dttm> <dttm> <chr> <chr> <chr>

1 8CD5DE2C2B6C4CFC docked\_bike 2020-06-13 23:24:00 2020-06-13 23:36:00 Wilton Ave & Belm… Damen … casual

2 9A191EB2C751D85D docked\_bike 2020-06-26 07:26:00 2020-06-26 07:31:00 Federal St & Polk… Daley … member

3 F37D14B0B5659BCF docked\_bike 2020-06-23 17:12:00 2020-06-23 17:21:00 Daley Center Plaza State … member

4 C41237B506E85FA1 docked\_bike 2020-06-20 01:09:00 2020-06-20 01:28:00 Broadway & Cornel… Broadw… casual

5 4B51B3B0BDA7787C docked\_bike 2020-06-25 16:59:00 2020-06-25 17:08:00 Sheffield Ave & W… Wilton… casual

6 D50DF288196B53BE docked\_bike 2020-06-17 18:07:00 2020-06-17 18:18:00 Sheffield Ave & W… Wilton… casual

# … with abbreviated variable names ¹​end\_station\_name, ²​member\_casual

str(combined\_databike) # See list of columns and data types (numeric, character, etc)

tibble [3,745,465 × 7] (S3: tbl\_df/tbl/data.frame)

$ ride\_id : chr [1:3745465] "8CD5DE2C2B6C4CFC" "9A191EB2C751D85D" "F37D14B0B5659BCF" "C41237B506E85FA1" ...

$ rideable\_type : chr [1:3745465] "docked\_bike" "docked\_bike" "docked\_bike" "docked\_bike" ...

$ started\_at : POSIXct[1:3745465], format: "2020-06-13 23:24:00" "2020-06-26 07:26:00" "2020-06-23 17:12:00" ...

$ ended\_at : POSIXct[1:3745465], format: "2020-06-13 23:36:00" "2020-06-26 07:31:00" "2020-06-23 17:21:00" ...

$ start\_station\_name: chr [1:3745465] "Wilton Ave & Belmont Ave" "Federal St & Polk St" "Daley Center Plaza" "Broadway & Cornelia Ave" ...

$ end\_station\_name : chr [1:3745465] "Damen Ave & Clybourn Ave" "Daley Center Plaza" "State St & Harrison St" "Broadway & Berwyn Ave" ...

$ member\_casual : chr [1:3745465] "casual" "member" "member" "casual" ...

summary(combined\_databike) # Statistical summary of data. Mainly for numerics

ride\_id rideable\_type started\_at ended\_at

Length:3745465 Length:3745465 Min. :2020-06-03 05:59:00.00 Min. :2020-06-03 06:03:00.00

Class :character Class :character 1st Qu.:2020-08-03 13:43:00.00 1st Qu.:2020-08-03 14:13:00.00

Mode :character Mode :character Median :2020-09-23 13:34:00.00 Median :2020-09-23 13:57:00.00

Mean :2020-11-02 01:40:18.89 Mean :2020-11-02 02:05:03.25

3rd Qu.:2021-03-05 16:48:00.00 3rd Qu.:2021-03-05 17:05:00.00

Max. :2021-05-31 23:59:00.00 Max. :2021-06-10 22:17:00.00

start\_station\_name end\_station\_name member\_casual

Length:3745465 Length:3745465 Length:3745465

Class :character Class :character Class :character

Mode :character Mode :character Mode :character

## transforming to

combined\_databike$date <- as.Date(combined\_databike$started\_at) # The default format is yyyy-mm-dd

> combined\_databike$month <- format(as.Date(combined\_databike$date),"%B")

> combined\_databike$day <- format(as.Date(combined\_databike$date), "%d")

combined\_databike$year <- format(as.Date(combined\_databike$date),"%Y")

> combined\_databike$day\_of\_week <- format(as.Date(combined\_databike$date), "%A")

## ## Add a "ride\_length" calculation to all\_trips (in seconds)

combined\_databike$ride\_length <-difftime(combined\_databike$ended\_at,combined\_databike$started\_at)

# Inspect the structure of the columns

str(combined\_databike)

tibble [3,745,465 × 13] (S3: tbl\_df/tbl/data.frame)

$ ride\_id : chr [1:3745465] "8CD5DE2C2B6C4CFC" "9A191EB2C751D85D" "F37D14B0B5659BCF" "C41237B506E85FA1" ...

$ rideable\_type : chr [1:3745465] "docked\_bike" "docked\_bike" "docked\_bike" "docked\_bike" ...

$ started\_at : POSIXct[1:3745465], format: "2020-06-13 23:24:00" "2020-06-26 07:26:00" "2020-06-23 17:12:00" "2020-06-20 01:09:00" ...

$ ended\_at : POSIXct[1:3745465], format: "2020-06-13 23:36:00" "2020-06-26 07:31:00" "2020-06-23 17:21:00" "2020-06-20 01:28:00" ...

$ start\_station\_name: chr [1:3745465] "Wilton Ave & Belmont Ave" "Federal St & Polk St" "Daley Center Plaza" "Broadway & Cornelia Ave" ...

$ end\_station\_name : chr [1:3745465] "Damen Ave & Clybourn Ave" "Daley Center Plaza" "State St & Harrison St" "Broadway & Berwyn Ave" ...

$ member\_casual : chr [1:3745465] "casual" "member" "member" "casual" ...

$ date : Date[1:3745465], format: "2020-06-13" "2020-06-26" "2020-06-23" "2020-06-20" ...

$ month : chr [1:3745465] "June" "June" "June" "June" ...

$ day : chr [1:3745465] "13" "26" "23" "20" ...

$ year : chr [1:3745465] "2020" "2020" "2020" "2020" ...

$ day\_of\_week : chr [1:3745465] "Saturday" "Friday" "Tuesday" "Saturday" ...

$ ride\_length : 'difftime' num [1:3745465] 720 300 540 1140 ...

..- attr(\*, "units")= chr "secs"

# Convert "ride\_length" from Factor to numeric so we can run calculations on the data

is.factor(combined\_databike$ride\_length)

combined\_databike$ride\_length <- as.numeric(as.character(combined\_databike$ride\_length))

is.numeric(combined\_databike$ride\_length)

> combined\_databike\_2 <- combined\_databike [!(combined\_databike$start\_station\_name == "HQ QR" | combined\_databike$ride\_length<0),]

> View(combined\_databike\_2)

**Analyze**

# STEP 4: CONDUCT DESCRIPTIVE ANALYSIS

#=====================================

summary(combined\_databike\_2$ride\_length)

Min. 1st Qu. Median Mean 3rd Qu. Max.

0 480 840 1626 1560 3256980

**Observation**

The average time of ride is 1626.10 seconds between casual customer and member customer

> mean(combined\_databike\_2$ride\_length) #straight average (total ride length / rides)

[1] 1626.104

> median(combined\_databike\_2$ride\_length) #midpoint number in the ascending array of ride lengths

[1] 840

> max(combined\_databike\_2$ride\_length) #longest ride

[1] 3256980

> min(combined\_databike\_2$ride\_length) #shortest ride

[1] 0

# Compare members and casual users

**Observation**

We can observe that casual customers spend more time (2603.78) in their ride from started\_at to ended\_at than member customers (915.8)

> aggregate(combined\_databike\_2$ride\_length ~ combined\_databike\_2$member\_casual, FUN = mean)

combined\_databike\_2$member\_casual

1 casual

2 member

combined\_databike\_2$ride\_length

1 2603.7754

2 915.8042

Discovery and relationships: on average casual member spent more time driving bikes than members.

> aggregate(combined\_databike\_2$ride\_length ~ combined\_databike\_2$member\_casual, FUN = median)

combined\_databike\_2$member\_casual

1 casual

2 member

combined\_databike\_2$ride\_length

1 1260

2 660

> aggregate(combined\_databike\_2$ride\_length ~ combined\_databike\_2$member\_casual, FUN = max)

combined\_databike\_2$member\_casual

1 casual

2 member

combined\_databike\_2$ride\_length

1 3256980

2 2476260

> aggregate(combined\_databike\_2$ride\_length ~ combined\_databike\_2$member\_casual, FUN = min)

combined\_databike\_2$member\_casual

1 casual

2 member

combined\_databike\_2$ride\_length

1 0

2 0

# See the average ride time by each day for members vs casual users

**Observations**

The average ride time by members vs casuals customers by each day were disorganized. I used ordered and levels to start from Sunday to Saturday. Each day casual customers spend more time than member

customers

# See the average ride time by each day for members vs casual users

aggregate(combined\_databike\_2$ride\_length ~ combined\_databike\_2$member\_casual + combined\_databike\_2$day\_of\_week, FUN = mean)

# Notice that the days of the week are out of order. Let's fix that.

combined\_databike\_2$day\_of\_week <-ordered(combined\_databike\_2$day\_of\_week, levels=c("Sunday", "Monday", "Tuesday", "Wednesday", "Thursday", "Friday", "Saturday"))

# Now, let's run the average ride time by each day for members vs casual users

> aggregate(combined\_databike\_2$ride\_length ~ combined\_databike\_2$member\_casual + combined\_databike\_2$day\_of\_week, FUN = mean)

combined\_databike\_2$member\_casual combined\_databike\_2$day\_of\_week combined\_databike\_2$ride\_length

1 casual Sunday 2974.7040

2 member Sunday 1039.1421

3 casual Monday 2575.7317

4 member Monday 875.5580

5 casual Tuesday 2316.4070

6 member Tuesday 863.0295

7 casual Wednesday 2346.7785

8 member Wednesday 870.6256

9 casual Thursday 2447.7436

10 member Thursday 864.1151

11 casual Friday 2473.0872

12 member Friday 893.6181

13 casual Saturday 2702.2623

14 member Saturday 1008.4317

**Observations**

The average ride time by members vs casuals customers by each month were disorganized. I used ordered and levels to start from Sunday to Saturday. The month of July was the highest average time for casual members with a time spend of 3550.78 seconds compared to member customers with 1050.98 seconds of that same month. The lowest for casual customers was October with 1874.6899 seconds. The lowest for members customers was 833.7450

## See the average ride time by each month for members vs casuals users

aggregate(combined\_databike\_2$ride\_length ~ combined\_databike\_2$member\_casual + combined\_databike\_2$month, FUN = mean)

# Notice that the days of the week are out of order. Let's fix that.

combined\_databike\_2$month <- ordered(combined\_databike\_2$month,c("June", "July", "August", "September", "October", "November", "December","January","February","March","April","May"))

# Now, let's run the average ride time by each day for members vs casual users

aggregate(combined\_databike\_2$ride\_length ~ combined\_databike\_2$member\_casual + combined\_databike\_2$month, FUN = mean)

> # Notice that the days of the week are out of order. Let's fix that.

> combined\_databike\_2$day\_of\_week <- ordered(combined\_databike\_2$day\_of\_week, levels=c("Sunday", "Monday", "Tuesday", "Wednesday", "Thursday", "Friday", "Saturday"))

1 casual June 3070.8678 secs

2 member June 1110.8220 secs

3 casual July 3550.7742 secs

4 member July 1050.9757 secs

5 casual August 2646.9551 secs

6 member August 990.2215 secs

7 casual September 2300.0574 secs

8 member September 913.0426 secs

9 casual October 1874.6899 secs

10 member October 833.7450 secs

11 casual November 2005.3134 secs

12 member November 807.0817 secs

13 casual December 1659.7466 secs

14 member December 738.3983 secs

15 casual January 1612.4128 secs

16 member January 720.7699 secs

17 casual February 2828.5983 secs

18 member February 886.7673 secs

19 casual March 2308.9062 secs

20 member March 819.8550 secs

21 casual April 2306.3654 secs

22 member April 855.8439 secs

23 casual May 2378.0678 secs

24 member May 860.6578 secs

## # Let's visualize the number of rides by rider type

**Share**

*## Number of rides completed by user type*

ggplot(combined\_databike\_2, aes(x=member\_casual))+

geom\_bar(fill="Red") +

labs(

title = "Number of rides completed by user type",

subtitle = "For the period between June 2020 and May 2021",

x = "User type",

y = "Number of rides (in millions)") +

scale\_y\_continuous(labels = label\_number(suffix = " M", scale = 1e-6)) +

geom\_text(stat='count', aes(label=..count..), vjust=+2, color="white")

Chart, treemap chart

Description automatically generated

**Observations:** There are more members customers by 2,167,516 (58%) than casual customers 1,574,747 (42%).

*## Number of rides each day by rider type*

combined\_databike\_2 %>%

group\_by(member\_casual, day\_of\_week) %>%

summarise(number\_of\_rides = n()

,average\_duration = mean(ride\_length)) %>%

arrange(member\_casual, day\_of\_week) %>%

ggplot(aes(x = day\_of\_week, y = number\_of\_rides, fill = member\_casual)) +

geom\_col(position = "dodge") +

labs(title= "Number of rides each day by rider type",

subtitle= "For the period between June 2020 and May 2021") +

scale\_y\_continuous(labels = label\_number(suffix = " M", scale = 1e-6)) +

geom\_text(aes(label=number\_of\_rides),position = position\_dodge(0.9),hjust=+3, color="black",angle= 90)

Chart, bar chart

Description automatically generated

**Observations:** The highest number of casual customers was on Saturday with 368790 and Tuesday with 159261 the lowest for the casual members. The highest number of member customers was on Saturday with 331289 and Sunday with 283609 the lowest for the members customers. Also the members customers number of rides per day from Sunday to Saturday does not change as much as the casual members that only Sunday and Saturday are the highest and Monday to Friday drop, but increases by day until reach Saturday.

*# Average rides each day by rider type*

combined\_databike\_2 %>%

group\_by(member\_casual, day\_of\_week) %>%

summarise(number\_of\_rides = n()

,average\_duration = mean(ride\_length)) %>%

arrange(member\_casual, day\_of\_week) %>%

ggplot(aes(x = day\_of\_week, y = average\_duration, fill = member\_casual)) +

geom\_col(position = "dodge") +

labs(title= "Average rides each day by rider type",

subtitle= "For the period between June 2020 and May 2021") +

scale\_y\_continuous(labels = label\_number(suffix = " M", scale = 1e-6)) +

geom\_text(aes(label=average\_duration),position = position\_dodge(0.85),hjust=0.9, color="black",angle= 90)

Chart, bar chart

Description automatically generated

**Observation:** The casual customers spent more time than members customers by a differences of 1000 seconds more than member customers.

*##* *Bike preference by user type.*

combined\_databike\_2 %>%

group\_by(member\_casual, rideable\_type) %>%

summarise(number\_of\_rides = n()) %>%

arrange(member\_casual, rideable\_type) %>%

ggplot(aes(x = member\_casual, y = number\_of\_rides, fill = rideable\_type)) +

geom\_bar(stat = "identity") + labs(title= "Bike preference and member\_casual",

subtitle= "June 2020 to May 2021") +

coord\_flip() +

geom\_text(aes(label=number\_of\_rides),position = position\_stack(vjust = .5), color="black") +

scale\_y\_continuous(labels = label\_number(suffix = " M", scale = 1e-6))

Chart, treemap chart

Description automatically generated

**Observations:** The docked bike is the most chosen rideable type by casual and member customers. The electric bike is the least chosen bike by casual and member customers.

combined\_databike\_2 %>%

group\_by(member\_casual, month) %>%

summarise(number\_of\_rides = n()) %>%

arrange(member\_casual, month) %>%

ggplot(aes(x = month, y = number\_of\_rides, fill = member\_casual)) +

geom\_col(position = "dodge") + labs(title= "Numbers of rides completed by month",

subtitle= "June 2020 to May 2021")

Chart, bar chart

Description automatically generated

**Observations:** The casual and member customers highest month is August. The lowest number of rides completed by casual customers are between December to February where it picks up and the same applies to member customers too. The highest number of rides completed by month for members were July, August and September. The casual customers highest months were July and August.

## Average of rides duration completed by month and member\_casual

combined\_databike\_2 %>%

group\_by(member\_casual, month) %>%

summarise(number\_of\_rides = n()

,average\_duration = mean(ride\_length)) %>%

arrange(member\_casual, month) %>%

ggplot(aes(x = month, y = average\_duration, fill = member\_casual)) +

geom\_col(position = "dodge") + labs(title= "Average of rides completed by month by member\_casual",

subtitle= "June 2020 to May 2021")

Chart, bar chart

Description automatically generated

**Observations:** Again, in each month the casual customers spent more time than members customers. The differences are casual customers time spent reduces from August to January until it hits February.

###

###

*## Top 10 start station by user types*

Table1 <- combined\_databike\_2 %>%

group\_by(member\_casual, start\_station\_name) %>%

summarise(count\_of=n()) %>%

arrange(desc(count\_of)) %>%

na.omit(start\_station\_name)

## Table 1.1 - By casual riders ##

Table1.1 <- filter(Table1, member\_casual =="casual") %>%

rename(number\_of\_rides = count\_of) %>%

slice(1:10)

#Table 1.2 - By members ##

Table1.2 <- filter(Table1, member\_casual =="member") %>%

rename(number\_of\_rides = count\_of) %>%

slice(1:10) Top 5 end station by user types

*print(Table1.1)*

# A tibble: 10 × 3

# Groups: member\_casual [1]

member\_casual start\_station\_name number\_of\_rides

<chr> <chr> <int>

1 casual Streeter Dr & Grand Ave 36107

2 casual Lake Shore Dr & Monroe St 27882

3 casual Millennium Park 24338

4 casual Theater on the Lake 18363

5 casual Michigan Ave & Oak St 18017

6 casual Lake Shore Dr & North Blvd 16645

7 casual Indiana Ave & Roosevelt Rd 15567

8 casual Shedd Aquarium 13681

9 casual Michigan Ave & Lake St 13634

10 casual Clark St & Elm St 13046

## Top 10 start station by casual customer

ggplot(Table1.1,aes(x = start\_station\_name, y = number\_of\_rides, fill = member\_casual)) +

geom\_bar(stat = "identity", width = 0.2) +

coord\_flip() +

labs(title = "Top 10 start station by casual customer", subtitle = "June 2020 to May 2021")

Bar chart

Description automatically generated with medium confidence

**Observations:** The “Streeter Dr & Grand Ave” was the most use start station by casual customers making 36107 rides from where to start.

print(Table1.2)

# A tibble: 10 × 3

# Groups: member\_casual [1]

member\_casual start\_station\_name number\_of\_rides

<chr> <chr> <int>

1 member Clark St & Elm St 22243

2 member Theater on the Lake 17375

3 member Wells St & Concord Ln 17310

4 member Broadway & Barry Ave 16949

5 member Kingsbury St & Kinzie St 16651

6 member St. Clair St & Erie St 16570

7 member Dearborn St & Erie St 16568

8 member Wells St & Elm St 16160

9 member Wells St & Huron St 15657

10 member Lake Shore Dr & North Blvd 15421

## top 10 start station by member customer

ggplot(Table1.2,aes(x = start\_station\_name, y = number\_of\_rides, fill = member\_casual)) +

geom\_bar(stat = "identity", width = 0.2) +

coord\_flip() +

labs(title = "Top 10 start station by member customer",subtitle= "June 2020 to May 2021")

Table

Description automatically generated

**Observations:** The “Clark St & Elm St” is the most use start station by members customers making 22243 rides from where to start. Also, for members customers Clark St & Elm St is the most used compared to casual customers that is the least used for them.

*##Top 10 end station by user types*

Table2 <- combined\_databike\_2 %>%

group\_by(member\_casual, end\_station\_name) %>%

summarise(count\_of=n()) %>%

arrange(desc(count\_of)) %>%

na.omit(end\_station\_name)

## Table 1.1 - By casual riders ##

Table2.1 <- filter(Table2, member\_casual =="casual") %>%

rename(number\_of\_rides = count\_of) %>%

slice(1:10)

#Table 1.2 - By members ##

Table2.2 <- filter(Table2, member\_casual =="member") %>%

rename(number\_of\_rides = count\_of) %>%

slice(1:10)

print(Table2.1)

# A tibble: 10 × 3

# Groups: member\_casual [1]

member\_casual end\_station\_name number\_of\_rides

<chr> <chr> <int>

1 casual Streeter Dr & Grand Ave 39288

2 casual Lake Shore Dr & Monroe St 26958

3 casual Millennium Park 25547

4 casual Theater on the Lake 20697

5 casual Michigan Ave & Oak St 18770

6 casual Lake Shore Dr & North Blvd 17860

7 casual Indiana Ave & Roosevelt Rd 15730

8 casual Michigan Ave & Lake St 13159

9 casual Michigan Ave & Washington St 12828

10 casual Clark St & Elm St 12559

ggplot(Table2.1,aes(x = end\_station\_name, y = number\_of\_rides, fill = member\_casual)) +

geom\_bar(stat = "identity", width = 0.2) +

coord\_flip() +

labs(title = "Top 10 end station by casual customer", subtitle= "June 2020 to May 2021")

Graphical user interface, application, table

Description automatically generated with medium confidence

**Observations:** The “Streeter Dr & Grand Ave” was the most use end station by casual customers making 39288 rides. Also this station name is the most used from start to end by casual customers.

*print(Table2.2)*

# A tibble: 10 × 3

# Groups: member\_casual [1]

member\_casual end\_station\_name number\_of\_rides

<chr> <chr> <int>

1 member Clark St & Elm St 22599

2 member Wells St & Concord Ln 17791

3 member St. Clair St & Erie St 17433

4 member Dearborn St & Erie St 17190

5 member Broadway & Barry Ave 17171

6 member Kingsbury St & Kinzie St 16814

7 member Theater on the Lake 16808

8 member Wells St & Elm St 15457

9 member Lake Shore Dr & North Blvd 14941

10 member Wells St & Huron St 14843

## top 10 start station by member customer

ggplot(Table2.2,aes(x = end\_station\_name, y = number\_of\_rides, fill = member\_casual)) +

geom\_bar(stat = "identity", width = 0.2) +

coord\_flip() +

labs(title = "Top 10 end station name by member customer",subtitle= "June 2020 to May 2021")

Graphical user interface, table

Description automatically generated

**Observations:** The “Clark St & Elm St” is the most use start station by members customers making 22599 rides. Also, for members customers Clark St & Elm St is the most used from start to end station name by member customers.

Reference list

<https://www.kaggle.com/code/zobuffy/case-study-how-does-a-bike-share-navigate-speedy/notebook>

<https://medium.com/@ypurutcu/how-does-a-bike-share-navigate-speedy-success-c99848743d42>

<https://www.linkedin.com/pulse/case-study-how-does-bike-share-navigate-speedy-success-richard-leung/?trk=public_profile_article_view>

<https://anniesanalytics.wixsite.com/annie-nelson/post/capstone-project-1-complete>

<https://www.youtube.com/watch?v=rpPt0JCzL6Q&t=1036s>

<https://ourcodingclub.github.io/tutorials/intro-to-r/>

<https://cran.r-project.org/doc/contrib/Paradis-rdebuts_en.pdf>

https://ourcodingclub.github.io/tutorials/intro-to-r/

<https://r-graph-gallery.com/ggplot2-package.html> Ggplot

<http://www.sthda.com/english/wiki/be-awesome-in-ggplot2-a-practical-guide-to-be-highly-effective-r-software-and-data-visualization>

<https://www.geeksforgeeks.org/select-top-n-highest-values-by-group-in-r/> ## for data frame and only use for the top stations