Google Data Analytics Capstone Project

Olisa Unegbu

02/02/2022

Project Goal:

To determine how annual members and casual riders use Cyclistic bikes differently with the aim of converting casual riders into annual members.

Characters and teams

Cylistic: The bike-share company,

Lily Moreno: My manager, the director of marketing,

Cyclistic marketing analytics team: A team of data analysts, and

Cyclist executive team: The detail-oriented executive team.

Ask Phase

I have being asked to help the marketing analyst team better understand how annual members and casual riders differ, why causal riders would buy Cyclistic annual memberships, and how digital media could affect their marketing tactics.

Tools used:

Excel, SQL and R

Prepare Phase

The Cyclistic's historical trip data is used to analyze and identify trends for this project. The 12 months of Cyclistic trip datasets used for this project were made available by Motivate International Inc. who granted me a non-exclusive, royalty-free, limited, perpetual license to access, reproduce, analyze, copy, modify, distribute in my product or service and use the Data for any lawful purpose("License"). There are no issues with bias or credibility in this data. The datasets are reliable, original, comprehensive, currrent and cited.

Setting up my environment

I will set up my environment by loading the "tidyverse" package and the 12 months Cyclistic trip csv datasets. Firstly, I set my current working directory to my datasets csv folder and used the read_csv to import all the 12 months needed datasets and the rbind to bind all the rows in the datasets into a dataframe.

```
library(tidyverse)
```

```
## -- 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
```

```
## Warning: package ’tibble’ was built under R version 4.1.2
```

```
## Warning: package ’readr’ was built under R version 4.1.2
```

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

```r
df1 <- read_csv("202004-cyclist-tripdata.csv")
```

```
## Rows: 84776 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.
```

```r
df2 <- read_csv("202005-cyclist-tripdata.csv")
```

```
## Rows: 200274 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.
```

```r
df3 <- read_csv("202006-cyclist-tripdata.csv")
```

```
## Rows: 343005 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.
```

```r
df4 <- read_csv("202007-cyclist-tripdata.csv")
```

```
## Rows: 551480 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.
```

```
df5 <- read_csv("202008-cyclist-tripdata.csv")
```

```
## Rows: 622361 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.
```

```
df6 <- read_csv("202009-cyclist-tripdata.csv")
```

```
## Rows: 532958 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.
```

```
df7 <- read_csv("202010-cyclist-tripdata.csv")
```

```
## Rows: 388653 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.
```

```
df8 <- read_csv("202011-cyclist-tripdata.csv")
```

```
## Rows: 259716 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.
```

```
df9 <- read_csv("202012-cyclist-tripdata.csv")
```

```
## Rows: 131573 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.
```

```
df10 <- read_csv("202101-cyclist-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.
```

```
df11 <- read_csv("202102-cyclist-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.
```

```
df12 <- read_csv("202103-cyclist-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.
```

```
cyclist_data <- rbind(df1,df2,df3,df4,df5,df6,df7,df8,df9,df10,df11,df12)

cyclist_data
```

```
## # A tibble: 3,489,748 x 13
##    ride_id          rideable_type started_at          ended_at           
##    <chr>            <chr>         <dttm>              <dttm>             
##  1 A847FADBBC638E45 docked_bike   2020-04-26 17:45:14 2020-04-26 18:12:03
##  2 5405B80E996FF60D docked_bike   2020-04-17 17:08:54 2020-04-17 17:17:03
##  3 5DD24A79A4E006F4 docked_bike   2020-04-01 17:54:13 2020-04-01 18:08:36
##  4 2A59BBDF5CDBA725 docked_bike   2020-04-07 12:50:19 2020-04-07 13:02:31
##  5 27AD306C119C6158 docked_bike   2020-04-18 10:22:59 2020-04-18 11:15:54
##  6 356216E875132F61 docked_bike   2020-04-30 17:55:47 2020-04-30 18:01:11
##  7 A2759CB06A81F2BC docked_bike   2020-04-02 14:47:19 2020-04-02 14:52:32
##  8 FC8BC2E2D54F35ED docked_bike   2020-04-07 12:22:20 2020-04-07 13:38:09
##  9 9EC5648678DE06E6 docked_bike   2020-04-15 10:30:11 2020-04-15 10:35:55
## 10 A8FFF89140C33017 docked_bike   2020-04-04 15:02:28 2020-04-04 15:19:47
## # ... with 3,489,738 more rows, and 9 more variables: start_station_name <chr>,
## #   start_station_id <chr>, end_station_name <chr>, end_station_id <chr>,
## #   start_lat <dbl>, start_lng <dbl>, end_lat <dbl>, end_lng <dbl>,
## #   member_casual <chr>
```

Understanding my data

I will have a preview of my data in order to understand its format.

```
head(cyclist_data)
```

```
## # A tibble: 6 x 13
##   ride_id          rideable_type started_at          ended_at            start_station_n~
##   <chr>            <chr>         <dttm>              <dttm>              <chr>           
## 1 A847FADBBC638E45 docked_bike   2020-04-26 17:45:14 2020-04-26 18:12:03 Eckhart Park    
## 2 5405B80E996FF60D docked_bike   2020-04-17 17:08:54 2020-04-17 17:17:03 Drake Ave & Ful~
## 3 5DD24A79A4E006F4 docked_bike   2020-04-01 17:54:13 2020-04-01 18:08:36 McClurg Ct & Er~
## 4 2A59BBDF5CDBA725 docked_bike   2020-04-07 12:50:19 2020-04-07 13:02:31 California Ave ~
## 5 27AD306C119C6158 docked_bike   2020-04-18 10:22:59 2020-04-18 11:15:54 Rush St & Hubba~
## 6 356216E875132F61 docked_bike   2020-04-30 17:55:47 2020-04-30 18:01:11 Mies van der Ro~
## # ... with 8 more variables: start_station_id <chr>, end_station_name <chr>,
## #   end_station_id <chr>, start_lat <dbl>, start_lng <dbl>, end_lat <dbl>,
## #   end_lng <dbl>, member_casual <chr>
```

```
dim(cyclist_data)
```

```
## [1] 3489748      13
```

The dimension of my data shows that I have 3,489,748 rows with 13 columns.

```
glimpse(cyclist_data)
```

```
## Rows: 3,489,748
## Columns: 13
## $ ride_id            <chr> "A847FADBBC638E45", "5405B80E996FF60D", "5DD24A79A4~
## $ rideable_type      <chr> "docked_bike", "docked_bike", "docked_bike", "docke~
## $ started_at         <dttm> 2020-04-26 17:45:14, 2020-04-17 17:08:54, 2020-04-~
## $ ended_at           <dttm> 2020-04-26 18:12:03, 2020-04-17 17:17:03, 2020-04-~
## $ start_station_name <chr> "Eckhart Park", "Drake Ave & Fullerton Ave", "McClu~
## $ start_station_id   <chr> "86", "503", "142", "216", "125", "173", "35", "434~
## $ end_station_name   <chr> "Lincoln Ave & Diversey Pkwy", "Kosciuszko Park", "~
## $ end_station_id     <chr> "152", "499", "255", "657", "323", "35", "635", "38~
## $ start_lat          <dbl> 41.8964, 41.9244, 41.8945, 41.9030, 41.8902, 41.896~
## $ start_lng          <dbl> -87.6610, -87.7154, -87.6179, -87.6975, -87.6262, -~
## $ end_lat            <dbl> 41.9322, 41.9306, 41.8679, 41.8992, 41.9695, 41.892~
## $ end_lng            <dbl> -87.6586, -87.7238, -87.6230, -87.6722, -87.6547, -~
## $ member_casual      <chr> "member", "member", "member", "member", "casual", "~
```

```
str(cyclist_data)
```

```
## spec_tbl_df [3,489,748 x 13] (S3: spec_tbl_df/tbl_df/tbl/data.frame)
##  $ ride_id           : chr [1:3489748] "A847FADBBC638E45" "5405B80E996FF60D" "5DD24A79A4E006F4" "2A59
##  $ rideable_type     : chr [1:3489748] "docked_bike" "docked_bike" "docked_bike" "docked_bike" ...
##  $ started_at        : POSIXct[1:3489748], format: "2020-04-26 17:45:14" "2020-04-17 17:08:54" ...
##  $ ended_at          : POSIXct[1:3489748], format: "2020-04-26 18:12:03" "2020-04-17 17:17:03" ...
##  $ start_station_name: chr [1:3489748] "Eckhart Park" "Drake Ave & Fullerton Ave" "McClurg Ct & Erie
##  $ start_station_id  : chr [1:3489748] "86" "503" "142" "216" ...
##  $ end_station_name  : chr [1:3489748] "Lincoln Ave & Diversey Pkwy" "Kosciuszko Park" "Indiana Ave &
##  $ end_station_id    : chr [1:3489748] "152" "499" "255" "657" ...
##  $ start_lat         : num [1:3489748] 41.9 41.9 41.9 41.9 41.9 ...
##  $ start_lng         : num [1:3489748] -87.7 -87.7 -87.6 -87.7 -87.6 ...
##  $ end_lat           : num [1:3489748] 41.9 41.9 41.9 41.9 42 ...
##  $ end_lng           : num [1:3489748] -87.7 -87.7 -87.6 -87.7 -87.7 ...
##  $ member_casual     : chr [1:3489748] "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>
```

```
colnames(cyclist_data)
```

```
##  [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"
```

Process Phase

Here, I will be checking the data for errors and cleaning the data as well using the RStudio.

```
cleaned_data <- cyclist_data %>%
  filter(!is.na(started_at)) %>%
  filter(!is.na(ended_at))
is.null(cleaned_data)
```

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

```
dim(cleaned_data)
```

```
## [1] 3489748      13
```

Since the dimension of my cleaned_data, 3,489,748 observations and 13 attributes is same with my cyclist_data, it shows that my most needed attributes for the analysis are free from NAs. I will be renaming some attributes like "rideable_type" to "bike_type", "started_at" to "start_time", "ended_at" to "end_at", and "member_casual" to "user_status". And will also check for consistency in some attribute names using the unique() function.

```
col_rename <- cleaned_data %>%
  dplyr::rename(bike_type = rideable_type, start_time = started_at,
         end_time = ended_at, user_status = member_casual)
colnames(col_rename)
```

```
##  [1] "ride_id"            "bike_type"          "start_time"
##  [4] "end_time"           "start_station_name" "start_station_id"
##  [7] "end_station_name"   "end_station_id"     "start_lat"
## [10] "start_lng"          "end_lat"            "end_lng"
## [13] "user_status"
```

Consistency check using the unique() function.

```
unique(col_rename$bike_type)
```

```
## [1] "docked_bike"   "electric_bike" "classic_bike"
```

```
unique(col_rename$user_status)
```

```
## [1] "member" "casual"
```

I will create an attribute called "ride_length" and calculate the length of each ride by subtracting the attribute "start_time" from the "end_time", another attribute called "day_of_week" and calculate the day of the week that each ride started, and another attribute called "month" to get the month each ride starts by using some functions in the 'lubridate' and 'dplyr' packages.

```
library(lubridate)
```

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

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

```
cyclist_tripdata <-col_rename %>%
  mutate(ride_length = as.duration(interval(ymd_hms(col_rename$start_time),
                                    ymd_hms(col_rename$end_time)))) %>%
  mutate(day_of_week = weekdays(as.Date(col_rename$start_time))) %>%
  mutate(month_of_ride = month.name[as.numeric(format(col_rename$start_time, "%m"))])
dim(cyclist_tripdata)
```

```
## [1] 3489748      16
```

```
glimpse(cyclist_tripdata)
```

```
## Rows: 3,489,748
## Columns: 16
## $ ride_id            <chr> "A847FADBBC638E45", "5405B80E996FF60D", "5DD24A79A4~
## $ bike_type          <chr> "docked_bike", "docked_bike", "docked_bike", "docke~
## $ start_time         <dttm> 2020-04-26 17:45:14, 2020-04-17 17:08:54, 2020-04--~
## $ end_time           <dttm> 2020-04-26 18:12:03, 2020-04-17 17:17:03, 2020-04--~
## $ start_station_name <chr> "Eckhart Park", "Drake Ave & Fullerton Ave", "McClu~
## $ start_station_id   <chr> "86", "503", "142", "216", "125", "173", "35", "434~
## $ end_station_name   <chr> "Lincoln Ave & Diversey Pkwy", "Kosciuszko Park", "~
## $ end_station_id     <chr> "152", "499", "255", "657", "323", "35", "635", "38~
## $ start_lat          <dbl> 41.8964, 41.9244, 41.8945, 41.9030, 41.8902, 41.896~
## $ start_lng          <dbl> -87.6610, -87.7154, -87.6179, -87.6975, -87.6262, --~
## $ end_lat            <dbl> 41.9322, 41.9306, 41.8679, 41.8992, 41.9695, 41.892~
## $ end_lng            <dbl> -87.6586, -87.7238, -87.6230, -87.6722, -87.6547, --~
## $ user_status        <chr> "member", "member", "member", "member", "casual", "~
## $ ride_length        <Duration> 1609s (~26.82 minutes), 489s (~8.15 minutes), ~
## $ day_of_week        <chr> "Sunday", "Friday", "Wednesday", "Tuesday", "Saturd~
## $ month_of_ride      <chr> "April", "April", "April", "April", "April", "April~
```

My glimpse now shows that I have 16 columns with same number of rows. The changes occurred as a result of the 3 new columns that I added.

Analyze Phase

I will run a few calculations here in order to identify trends and relationships.

```
table(cyclist_tripdata$user_status)
```

```
##
##  casual  member
## 1430376 2059372
```

```
table(cyclist_tripdata$bike_type)
```

```
##
##  classic_bike   docked_bike electric_bike
##        319873       2558469        611406
```

The analysis above shows that we have more annual member riders than the casual riders. And it also shows that the 'docked-bike' type is used more than the others as 73.31% of the bikes used is attributed to it.

```
  cyclist_tripdata %>%
  group_by(user_status) %>%
  dplyr::summarize(avg = mean(ride_length)/60)
```

```
## # A tibble: 2 x 2
##   user_status   avg
##   <chr>       <dbl>
## 1 casual       43.3
## 2 member       11.9
```

The analysis here shows that casual riders ride longer than member riders with 43.29 minutes and 11.90 minutes respectively. Which is averagely 31.39mins longer.

```
day_of_ride <- cyclist_tripdata %>%
  group_by(day_of_week, user_status) %>%
  select(user_status, day_of_week) %>%
  dplyr::summarize(number_of_day = table(day_of_week), .groups = 'drop')
day_of_ride
```

```
## # A tibble: 14 x 3
##    day_of_week user_status number_of_day
##    <chr>       <chr>       <table>
##  1 Friday      casual      209131
##  2 Friday      member      307671
##  3 Monday      casual      151460
##  4 Monday      member      268096
##  5 Saturday    casual      335901
##  6 Saturday    member      324283
##  7 Sunday      casual      262861
##  8 Sunday      member      266256
##  9 Thursday    casual      166672
## 10 Thursday    member      301321
## 11 Tuesday     casual      145660
## 12 Tuesday     member      285632
## 13 Wednesday   casual      158691
## 14 Wednesday   member      306113
```

```
month_of_ride <- cyclist_tripdata %>%
  group_by(month_of_ride, user_status) %>%
  select(user_status, month_of_ride) %>%
  dplyr::summarize(number_of_day = table(month_of_ride), .groups = 'drop')
month_of_ride
```

```
## # A tibble: 24 x 3
##    month_of_ride user_status number_of_day
##    <chr>         <chr>       <table>
##  1 April         casual       23628
##  2 April         member       61148
##  3 August        casual      289661
##  4 August        member      332700
##  5 December      casual       30080
##  6 December      member      101493
##  7 February      casual       10131
##  8 February      member       39491
##  9 January       casual       18117
## 10 January       member       78717
## # ... with 14 more rows
```

```
bike_type_users <- cyclist_tripdata %>%
  group_by(bike_type, user_status) %>%
  select(bike_type, user_status) %>%
  dplyr::summarize(number_of_bike_users = table(bike_type), .groups = 'drop')
bike_type_users
```

```
## # A tibble: 6 x 3
##   bike_type     user_status number_of_bike_users
##   <chr>         <chr>       <table>
## 1 classic_bike  casual        70801
## 2 classic_bike  member       249072
## 3 docked_bike   casual      1116583
## 4 docked_bike   member      1441886
## 5 electric_bike casual       242992
## 6 electric_bike member       368414
```

I will save my outcomes “day_of_ride”, “month_of_ride”, and “bike_type_users” on my working directory as a csv file and export both to SQL in order to separate ‘casual’ and ‘member’ with their corresponding ‘day_of_ride’, ‘month_of_ride’, and ‘bike_type’ respectively. I will do that by performing the command below; SELECT * FROM day_of_ride WHERE user_status = "casual" Performed same for member and will also use same format to extract that of 'month_of_ride', and 'bike_type_users'.

```
write.csv(day_of_ride, file = "day_of_ride.csv", row.names = FALSE)
write.csv(month_of_ride, file = "month_of_ride.csv", row.names = FALSE)
write.csv(bike_type_users, file = "bike_type_users.csv", row.names = FALSE)
```

I will import my query results back into my RStudio for further analysis.

```
{casual_day_of_ride <- read.csv("casual_day_of_ride.csv")}
member_day_of_ride <- read.csv("member_day_of_ride.csv")

casual_month_of_ride <- read.csv("casual_month_of_ride.csv")
member_month_of_ride <- read.csv("member_month_of_ride.csv")

casual_bike_type_users <- read.csv("casual_bike_type_users.csv")
member_bike_type_users <- read.csv("member_bike_type_users.csv")

casual_day_of_ride
```

```
##   day_of_week user_status number_of_days
## 1      Friday      casual         197030
## 2      Monday      casual         142332
## 3    Saturday      casual         319858
## 4      Sunday      casual         250344
## 5    Thursday      casual         156446
## 6     Tuesday      casual         136588
## 7   Wednesday      casual         148641
```

```
member_day_of_ride
```

```
##   day_of_week user_status number_of_days
## 1      Friday      member         290165
## 2      Monday      member         252664
## 3    Saturday      member         305769
## 4      Sunday      member         251416
## 5    Thursday      member         284598
## 6     Tuesday      member         269439
## 7   Wednesday      member         289401
```

```
casual_month_of_ride
```

```
##    month_of_ride user_status number_of_days
## 1          April      casual          23584
## 2         August      casual         283404
## 3       December      casual          24489
## 4       February      casual           8608
## 5        January      casual          14698
## 6           July      casual         268421
## 7           June      casual         154289
## 8          March      casual          75633
## 9            May      casual          86783
## 10      November      casual          73130
## 11       October      casual         122821
## 12     September      casual         215379
```

```
member_month_of_ride
```

```
##    month_of_ride user_status number_of_days
## 1          April      member          61095
## 2         August      member         325547
## 3       December      member          89096
## 4       February      member          34383
## 5        January      member          68823
## 6           July      member         281649
## 7           June      member         188004
## 8          March      member         130040
## 9            May      member         113196
## 10      November      member         150024
## 11       October      member         216492
## 12     September      member         285103
```

```
casual_bike_type_users
```

```
##       bike_type user_status number_of_bike_users
## 1  classic_bike      casual                70801
## 2   docked_bike      casual              1116583
## 3 electric_bike      casual               242992
```

```
member_bike_type_users
```

```
##       bike_type user_status number_of_bike_users
## 1  classic_bike      member               249072
## 2   docked_bike      member              1441886
## 3 electric_bike      member               368414
```

I will sort the above outcomes in a descending order.

```
sort_casual_day_of_ride <- casual_day_of_ride %>%
  arrange(desc(number_of_days))
sort_member_day_of_ride <- member_day_of_ride %>%
  arrange(desc(number_of_days))

sort_casual_month_of_ride <- casual_month_of_ride %>%
  arrange(desc(number_of_days))
sort_member_month_of_ride <- member_month_of_ride %>%
  arrange(desc(number_of_days))

sort_casual_bike_type_users <- casual_bike_type_users %>%
  arrange(desc(number_of_bike_users))
sort_member_bike_type_users <- member_bike_type_users %>%
  arrange(desc(number_of_bike_users))
```

Identifying trends and relationships in the sorted data.

```
sort_casual_day_of_ride
```

```
##   day_of_week user_status number_of_days
## 1    Saturday      casual         319858
## 2      Sunday      casual         250344
## 3      Friday      casual         197030
## 4    Thursday      casual         156446
## 5   Wednesday      casual         148641
## 6      Monday      casual         142332
## 7     Tuesday      casual         136588
```

```
sort_member_day_of_ride
```

```
##   day_of_week user_status number_of_days
## 1    Saturday      member         305769
## 2      Friday      member         290165
## 3   Wednesday      member         289401
## 4    Thursday      member         284598
## 5     Tuesday      member         269439
## 6      Monday      member         252664
## 7      Sunday      member         251416
```

The above analysis shows that casual riders use the bikes more on weekends than weekdays with Saturday, Sunday, and Friday occurring most while member riders have theirs to be Saturday, Friday, and Wednesday.

```
{sort_casual_month_of_ride}
```

```
##    month_of_ride user_status number_of_days
## 1         August      casual         283404
## 2           July      casual         268421
## 3      September      casual         215379
## 4           June      casual         154289
## 5        October      casual         122821
## 6            May      casual          86783
## 7          March      casual          75633
## 8       November      casual          73130
## 9       December      casual          24489
## 10         April      casual          23584
## 11       January      casual          14698
## 12      February      casual           8608
```

```
sort_member_month_of_ride
```

```
##    month_of_ride user_status number_of_days
## 1         August      member         325547
## 2      September      member         285103
## 3           July      member         281649
## 4        October      member         216492
## 5           June      member         188004
## 6       November      member         150024
## 7          March      member         130040
## 8            May      member         113196
## 9       December      member          89096
## 10       January      member          68823
## 11         April      member          61095
## 12      February      member          34383
```

Casual riders were shown to ride more in summer months which are basically from June to September. Annual members have theirs to fall between July to October.

```
sort_casual_bike_type_users
```

```
##       bike_type user_status number_of_bike_users
## 1   docked_bike      casual              1116583
## 2 electric_bike      casual               242992
## 3  classic_bike      casual                70801
```

```
sort_member_bike_type_users
```

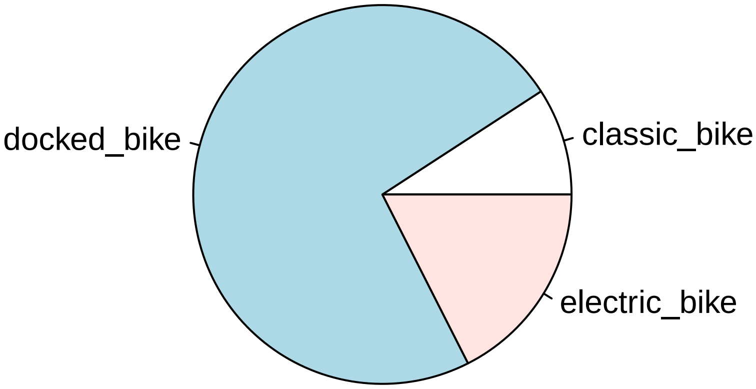
```
##       bike_type user_status number_of_bike_users
## 1   docked_bike      member              1441886
## 2 electric_bike      member               368414
## 3  classic_bike      member               249072
```

As shown above, casual riders ride more of docked_bike which is same with annual member riders.

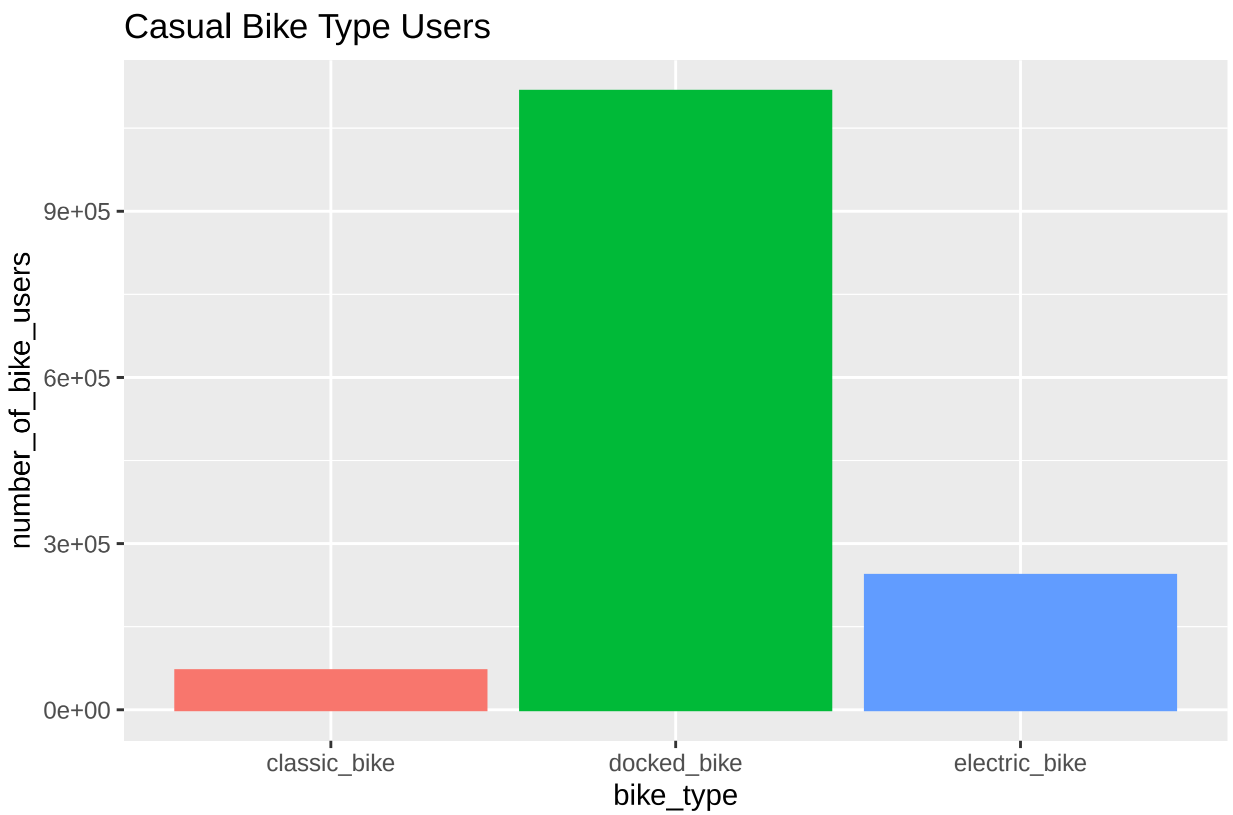
Share Phase

I will create visuals to tell my data story and also use them to communicate my findings. Since our project goal is on converting casual riders to annual members, I will focus more on casual riders charts.

```
pie(table(cyclist_tripdata$bike_type))
```

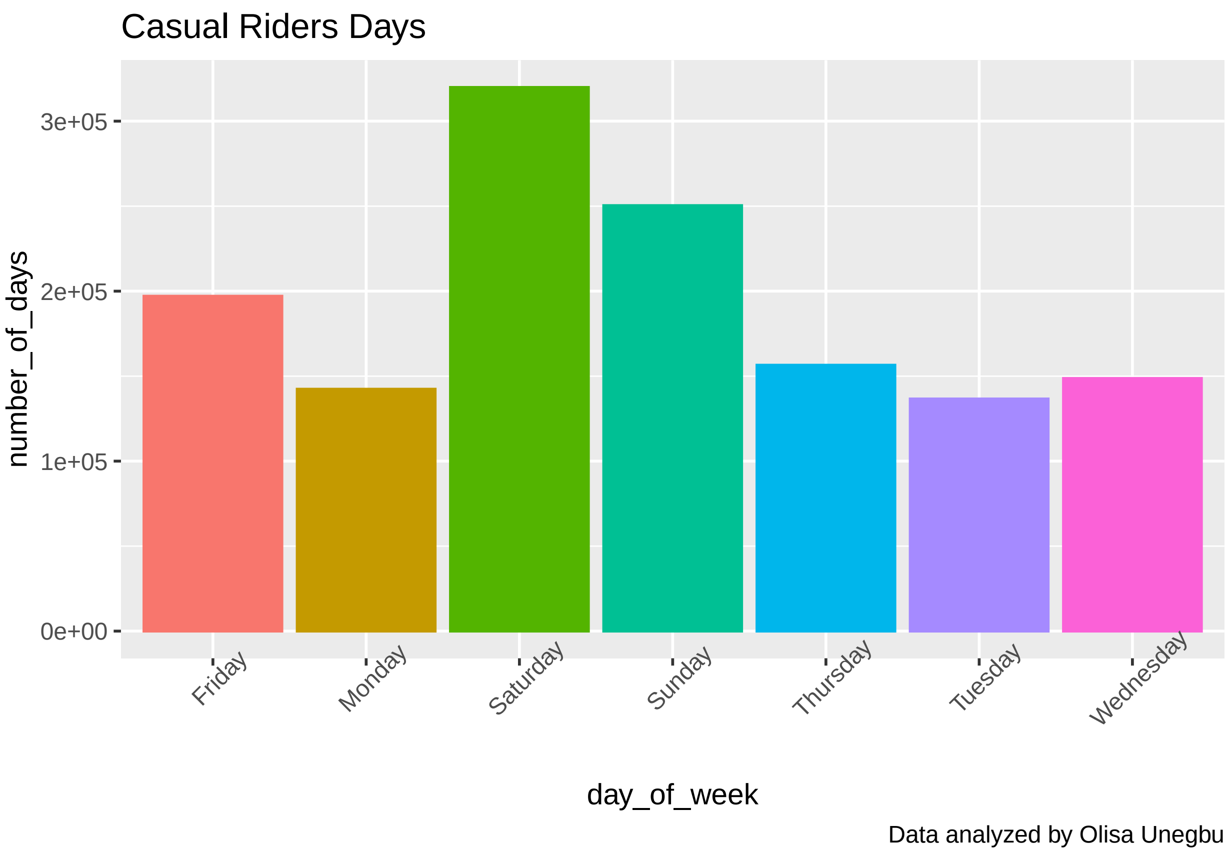


```
ggplot(casual_bike_type_users, aes(x = bike_type, y = number_of_bike_users,
                                   color = bike_type, fill = bike_type)) +
  geom_bar(stat = "identity")  +
  theme(legend.position = "none") +
  labs(title = "Casual Bike Type Users",
       caption = "Data analyzed by Olisa Unegbu")
```

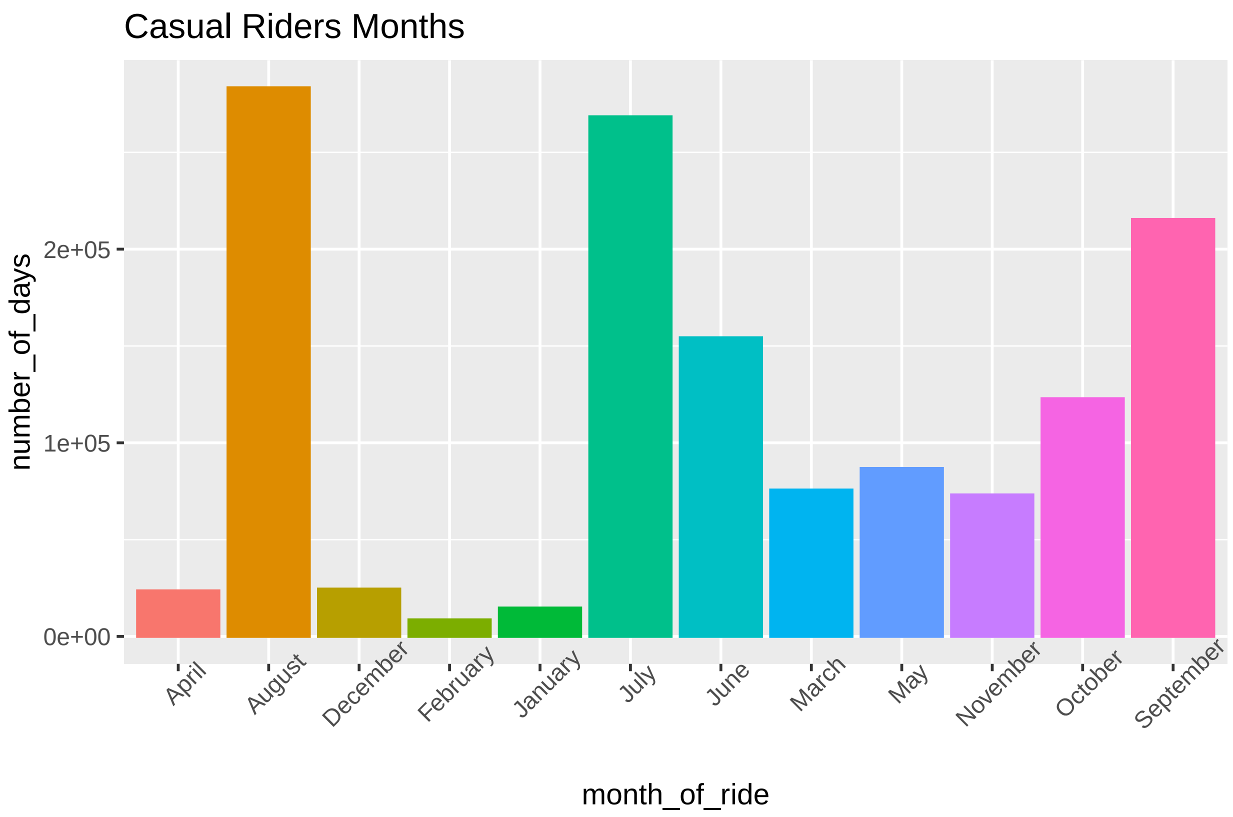


Data analyzed by Olisa Unegbu

```
ggplot(casual_day_of_ride, aes(x = day_of_week, y = number_of_days,
                               color = day_of_week, fill = day_of_week)) +
    geom_bar(stat = "identity")  +
     theme(legend.position = "none",axis.text.x = element_text(angle=45)) +
      labs(title = "Casual Riders Days",
      caption = "Data analyzed by Olisa Unegbu")
```



```
ggplot(casual_month_of_ride, aes(x = month_of_ride, y = number_of_days,
                                 color = month_of_ride, fill = month_of_ride)) +
  geom_bar(stat = "identity")  +
  theme(legend.position = "none", axis.text.x = element_text(angle=45)) +
  labs(title = "Casual Riders Months",
       caption = "Data analyzed by Olisa Unegbu")
```



Data analyzed by Olisa Unegbu

Act Phase

Based on the above analysis, my top three recommendations are stated as follows;

1. I will recommend you offer discounts to annual members using docked_bike.
2. Offer special discounts to annual members for rides that last longer than 15 minutes.
3. Offer special summer promo packages to annual members.

Olisa Unegbu

02/02/2022