```r
# This R environment comes with many helpful analytics packages
installed
# It is defined by the kaggle/rstats Docker image:
https://github.com/kaggle/docker-rstats
# For example, here's a helpful package to load

library(tidyverse) # metapackage of all tidyverse packages
install.packages("ggplot2")
install.packages("readr")
install.packages("fastdigest")
install.packages("validate")
install.packages("lubridate")
install.packages("dplyr")

# Input data files are available in the read-only "../input/"
directory
# For example, running this (by clicking run or pressing Shift+Enter)
will list all files under the input directory

list.files(path = "/kaggle/input/cyclistic-bike-share")

# You can write up to 20GB to the current directory (/kaggle/working/)
that gets preserved as output when you create a version using "Save &
Run All"
# You can also write temporary files to /kaggle/temp/, but they won't
be saved outside of the current session
```

```
── Attaching core tidyverse packages ──────────────────────── tidyverse 2.0.0 ──
✔ dplyr     1.1.2     ✔ readr     2.1.4
✔ forcats   1.0.0     ✔ stringr   1.5.0
✔ ggplot2   3.4.2     ✔ tibble    3.2.1
✔ lubridate 1.9.2     ✔ tidyr     1.3.0
✔ purrr     1.0.1
── Conflicts ────────────────────────────────────────── tidyverse_conflicts() ──
✖ dplyr::filter() masks stats::filter()
✖ dplyr::lag()    masks stats::lag()
ⓘ Use the conflicted package (<http://conflicted.r-lib.org/>) to
force all conflicts to become errors
Installing package into ‘/usr/local/lib/R/site-library’
(as ‘lib’ is unspecified)

Installing package into ‘/usr/local/lib/R/site-library’
(as ‘lib’ is unspecified)

Installing package into ‘/usr/local/lib/R/site-library’
(as ‘lib’ is unspecified)

Installing package into ‘/usr/local/lib/R/site-library’
```

```
(as ‘lib’ is unspecified)

Installing package into ‘/usr/local/lib/R/site-library’
(as ‘lib’ is unspecified)

Installing package into ‘/usr/local/lib/R/site-library’
(as ‘lib’ is unspecified)

Warning message in install.packages("dplyr"):
“installation of package ‘dplyr’ had non-zero exit status”

 [1] "202207-divvy-tripdata.csv"       "202208-divvy-tripdata.csv"

 [3] "202209-divvy-publictripdata.csv" "202210-divvy-tripdata.csv"

 [5] "202211-divvy-tripdata.csv"       "202212-divvy-tripdata.csv"

 [7] "202301-divvy-tripdata.csv"       "202302-divvy-tripdata.csv"

 [9] "202303-divvy-tripdata.csv"       "202304-divvy-tripdata.csv"

[11] "202305-divvy-tripdata.csv"       "202306-divvy-tripdata.csv"
```

Case Study

How does a Bike Share Navigate Speedy Success?

Author: Hero Clement Gomes

Date: August 12, 2023

Table of contents:

Introduction: The goal of this case study is to design marketing strategies aimed at converting casual riders into annual members. Here the client is Cyclistic a bike-share company. The director of marketing believes the company's future success depends on maximizing the number of annual memberships. Therefore, your team wants to understand how casual riders and annual members use Cyclistic bikes differently.

About Cyclistic: Cyclistic bike-share company lunched their business in 2016. Since then, the program has grown to a fleet of 5,824 bicycles that are into a network of 692 stations across Chicago.

Ask step:

Here we have identified three questions need to be answered:

```
How do annual members and casual riders use Cyclistic bikes differently?

Why would casual riders buy Cyclistic annual memberships?

How can Cyclistic use digital media to influence casual riders to become members?
```

Manager has assigned me the first question to answer: How do annual members and casual riders use Cyclistic bikes differently?

I have to produce a report with the following deliverables:

```
A clear statement of the business task

A description of all data sources used

Documentation of any cleaning or manipulation of data

A summary of your analysis

Supporting visualizations and key findings

Your top three recommendations based on your analysis
```

Defining the problem: Here we find that there are two issues regarding the growth of the business-1) Do we need to take initiative to increase new customers or 2) convert casual customer into annual member. I have another question-Whether the pricings are correct? I mean pricing depends on cost price of asset, its longevity, repair & maintenance costs. The goal of this project is to design marketing strategies aimed at converting casual riders into annual members.

Help from the insights of data: I will analyze the data of casual customers and annual members. We will try to find what type of services they like or dislike or what they deserve. Also, will analyze the trends of the business.

Key tasks: Find out the best option among these two—1) do we need to take initiative to increase new customers or 2) convert casual customer into annual member. To find out this we will prepare the data and process it for analysis and visualization to find the trends, share our analysis and then act on it.

Here another important task is to consider my stakeholders. My stakeholders are:

```
Cyclistic executive team: The highest level of stakeholders,
responsible for making decisions about the company's overall strategy.

Lily Moreno, director of marketing: The next level of stakeholders,
responsible for developing and implementing marketing campaigns.

Cyclistic marketing analytics team: The next level of stakeholders,
responsible for collecting and analyzing data that helps guide
marketing strategy.
```

Me, a junior data analyst: The lowest level of stakeholders, responsible for collecting and analyzing data.

Deliverable: Purpose of this case study is to find how casual riders and annual member riders act on the bike-share services. Collect data on both casual riders and annual members.

Prepare step:

Data source: Data location is https://divvy-tripdata.s3.amazonaws.com/index.html

Data organization: From above location I have downloaded datasets ranging from July 2022 to June 2023. I will combine 12 datasets to create a single data.

Bias or credibility issues: Cyclistic is a fictional company. datasets are appropriate and will enable us to answer the business questions. The data has been made available by Motivate International Inc. under license https://ride.divvybikes.com/data-license-agreement.

Licensing, privacy, security, and accessibility: About licensing, privacy, security and accessibility all are mentioned here https://ride.divvybikes.com/data-license-agreement.

Verification of data integrity: In RStudio[case_study1] checked all 12 datasets. I have used fastdigest() function to find out the data integrity.

Data merge and other issues: 12 datasets need to be merged and later on it will be processed in process step.

```r
# We will load essential libraries
library("tidyverse")
library("ggplot2")
library("readr")
library("fastdigest")
library("dplyr")
library("validate")
library("lubridate")
```

```
Attaching package: ‘validate’


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

    expr


The following object is masked from ‘package:ggplot2’:

    expr
```

```r
# Now we will load 12 datasets (202207 to 202306)

df_july22 <- read.csv("/kaggle/input/cyclistic-bike-share/202207-divvy-tripdata.csv")
df_aug22 <- read.csv("/kaggle/input/cyclistic-bike-share/202208-divvy-tripdata.csv")
df_sept22 <- read.csv("/kaggle/input/cyclistic-bike-share/202209-divvy-publictripdata.csv")
df_oct22 <- read.csv("/kaggle/input/cyclistic-bike-share/202210-divvy-tripdata.csv")
df_nov22 <- read.csv("/kaggle/input/cyclistic-bike-share/202211-divvy-tripdata.csv")
df_dec22 <- read.csv("/kaggle/input/cyclistic-bike-share/202212-divvy-tripdata.csv")
df_jan23 <- read.csv("/kaggle/input/cyclistic-bike-share/202301-divvy-tripdata.csv")
df_feb23 <- read.csv("/kaggle/input/cyclistic-bike-share/202302-divvy-tripdata.csv")
df_mar23 <- read.csv("/kaggle/input/cyclistic-bike-share/202303-divvy-tripdata.csv")
df_apr23 <- read.csv("/kaggle/input/cyclistic-bike-share/202304-divvy-tripdata.csv")
df_may23 <- read.csv("/kaggle/input/cyclistic-bike-share/202305-divvy-tripdata.csv")
df_june23 <- read.csv("/kaggle/input/cyclistic-bike-share/202306-divvy-tripdata.csv")

# Calculate the hash value of datasets
hash1 <- fastdigest(df_july22)
hash2 <- fastdigest(df_aug22)
hash3 <- fastdigest(df_sept22)
hash4 <- fastdigest(df_oct22)
hash5 <- fastdigest(df_nov22)
hash6 <- fastdigest(df_dec22)
hash7 <- fastdigest(df_jan23)
hash8 <- fastdigest(df_feb23)
```

```
hash9 <- fastdigest(df_mar23)
hash10 <- fastdigest(df_apr23)
hash11 <- fastdigest(df_may23)
hash12 <- fastdigest(df_june23)

# Save the hash values
hash_value1 <- hash1
hash_value2 <- hash2
hash_value3 <- hash3
hash_value4 <- hash4
hash_value5 <- hash5
hash_value6 <- hash6
hash_value7 <- hash7
hash_value8 <- hash8
hash_value9 <- hash9
hash_value10 <- hash10
hash_value11 <- hash11
hash_value12 <- hash12
```

Now we will check the integrity of the data sets

```
if (hash1 == hash_value1) {
  print("The dataset has not been modified.")
} else {
  print("The dataset has been modified.")
}

if (hash2 == hash_value2) {
  print("The dataset has not been modified.")
} else {
  print("The dataset has been modified.")
}

if (hash3 == hash_value3) {
  print("The dataset has not been modified.")
} else {
  print("The dataset has been modified.")
}

if (hash4 == hash_value4) {
  print("The dataset has not been modified.")
} else {
  print("The dataset has been modified.")
}

if (hash5 == hash_value5) {
  print("The dataset has not been modified.")
} else {
  print("The dataset has been modified.")
}
```

```
if (hash6 == hash_value6) {
  print("The dataset has not been modified.")
} else {
  print("The dataset has been modified.")
}

if (hash7 == hash_value7) {
  print("The dataset has not been modified.")
} else {
  print("The dataset has been modified.")
}

if (hash8 == hash_value8) {
  print("The dataset has not been modified.")
} else {
  print("The dataset has been modified.")
}

if (hash9 == hash_value9) {
  print("The dataset has not been modified.")
} else {
  print("The dataset has been modified.")
}

if (hash10 == hash_value10) {
  print("The dataset has not been modified.")
} else {
  print("The dataset has been modified.")
}

if (hash11 == hash_value11) {
  print("The dataset has not been modified.")
} else {
  print("The dataset has been modified.")
}

if (hash12 == hash_value12) {
  print("The dataset has not been modified.")
} else {
  print("The dataset has been modified.")
}
```

```
[1] "The dataset has not been modified."
[1] "The dataset has not been modified."
[1] "The dataset has not been modified."
[1] "The dataset has not been modified."
[1] "The dataset has not been modified."
[1] "The dataset has not been modified."
[1] "The dataset has not been modified."
```

```
[1] "The dataset has not been modified."
[1] "The dataset has not been modified."
[1] "The dataset has not been modified."
[1] "The dataset has not been modified."
[1] "The dataset has not been modified."
```

After running above code chunks I found that datasets were not modified. So, I am confident that the datasets are secure and usable.

Now we need to join our 12 datasets.

1st we will check if column names of each dataset are same

```
# Importing 12 datasets
df_datasets <- list.files("/kaggle/input/cyclistic-bike-share",
pattern="*.csv")

# Using the colnames() function to get the column names of each
dataset.
col_names <- lapply(df_datasets, colnames)

# Storing column names of all 12 datasets
all_col_names <- unlist(col_names)

# Now we will compare the column names of the vector to the column
names of each dataset by using identical() function
for (df_dataset in df_datasets) {
  if(identical(all_col_names, colnames(df_dataset))) {
    print(paste(df_dataset, "has the same columns"))
  } else {
    print(paste(df_dataset, "does not have the same columns"))
  }
}
```

```
[1] "202207-divvy-tripdata.csv has the same columns"
[1] "202208-divvy-tripdata.csv has the same columns"
[1] "202209-divvy-publictripdata.csv has the same columns"
[1] "202210-divvy-tripdata.csv has the same columns"
[1] "202211-divvy-tripdata.csv has the same columns"
[1] "202212-divvy-tripdata.csv has the same columns"
[1] "202301-divvy-tripdata.csv has the same columns"
[1] "202302-divvy-tripdata.csv has the same columns"
[1] "202303-divvy-tripdata.csv has the same columns"
[1] "202304-divvy-tripdata.csv has the same columns"
[1] "202305-divvy-tripdata.csv has the same columns"
[1] "202306-divvy-tripdata.csv has the same columns"
```

```
# Now we will create a data list of 12 datasets
data_list <- list(df_july22, df_aug22, df_sept22, df_oct22, df_nov22,
df_dec22, df_jan23, df_feb23, df_mar23, df_apr23, df_may23, df_june23)
```

```r
# Join the data frames (here we will join 1st 2 datasets)
combined_data <- dplyr::full_join(data_list[[1]], data_list[[2]])
```

```
Joining with `by = join_by(ride_id, rideable_type, started_at,
ended_at,
start_station_name, start_station_id, end_station_name,
end_station_id,
start_lat, start_lng, end_lat, end_lng, member_casual)`
```

```r
# Now we will join rest of the 10 datasets with above combined_data to
complete joining
for (i in 3:12) {
  new_data <- data_list[[i]]
  combined_data <- dplyr::full_join(combined_data, new_data)
}
```

```
Joining with `by = join_by(ride_id, rideable_type, started_at,
ended_at,
start_station_name, start_station_id, end_station_name,
end_station_id,
start_lat, start_lng, end_lat, end_lng, member_casual)`
Joining with `by = join_by(ride_id, rideable_type, started_at,
ended_at,
start_station_name, start_station_id, end_station_name,
end_station_id,
start_lat, start_lng, end_lat, end_lng, member_casual)`
Joining with `by = join_by(ride_id, rideable_type, started_at,
ended_at,
start_station_name, start_station_id, end_station_name,
end_station_id,
start_lat, start_lng, end_lat, end_lng, member_casual)`
Joining with `by = join_by(ride_id, rideable_type, started_at,
ended_at,
start_station_name, start_station_id, end_station_name,
end_station_id,
start_lat, start_lng, end_lat, end_lng, member_casual)`
Joining with `by = join_by(ride_id, rideable_type, started_at,
ended_at,
start_station_name, start_station_id, end_station_name,
end_station_id,
start_lat, start_lng, end_lat, end_lng, member_casual)`
Joining with `by = join_by(ride_id, rideable_type, started_at,
ended_at,
start_station_name, start_station_id, end_station_name,
end_station_id,
start_lat, start_lng, end_lat, end_lng, member_casual)`
Joining with `by = join_by(ride_id, rideable_type, started_at,
ended_at,
start_station_name, start_station_id, end_station_name,
end_station_id,
```

```
start_lat, start_lng, end_lat, end_lng, member_casual)`
Joining with `by = join_by(ride_id, rideable_type, started_at,
ended_at,
start_station_name, start_station_id, end_station_name,
end_station_id,
start_lat, start_lng, end_lat, end_lng, member_casual)`
Joining with `by = join_by(ride_id, rideable_type, started_at,
ended_at,
start_station_name, start_station_id, end_station_name,
end_station_id,
start_lat, start_lng, end_lat, end_lng, member_casual)`
Joining with `by = join_by(ride_id, rideable_type, started_at,
ended_at,
start_station_name, start_station_id, end_station_name,
end_station_id,
start_lat, start_lng, end_lat, end_lng, member_casual)`
```

Process Step:

```
# Now we will add new coloumns to our dataset for future analysis.
Columns are date, month, day and year.
combined_data <- combined_data %>%
  mutate(
    date=as.Date(started_at),
    month=format(as.Date(date), "%m"),
    day=format(as.Date(date), "%d"),
    year=format(as.Date(date), "%y"),
    day_of_week=format(as.Date(date), "%A")
  )

# Checking column names
colnames(combined_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"      "date"               "month"
[16] "day"                "year"               "day_of_week"
```

```
# Add a column 'ride_length' of each ride to calculate ride duration
of each ride.
combined_data <- combined_data %>%
  mutate(ride_length=difftime(ended_at, started_at, units="mins"))

# Now we will convert "ride_length" from Double to numeric so we can
run calculations on the data
combined_data <- combined_data %>%
  mutate(ride_length=as.numeric(ride_length))
is.numeric(combined_data$ride_length)
```

```
[1] TRUE
```

```
# Check the combined_data
summary(combined_data)
```

```
   ride_id           rideable_type        started_at           ended_at
 Length:5779444     Length:5779444     Length:5779444
Length:5779444
 Class :character   Class :character   Class :character
Class :character
 Mode  :character   Mode  :character   Mode  :character
Mode  :character


 start_station_name start_station_id   end_station_name
end_station_id
 Length:5779444     Length:5779444     Length:5779444
Length:5779444
 Class :character   Class :character   Class :character
Class :character
 Mode  :character   Mode  :character   Mode  :character
Mode  :character


   start_lat       start_lng         end_lat         end_lng
 Min.   :41.64   Min.   :-87.87   Min.   : 0.00   Min.   :-88.16
 1st Qu.:41.88   1st Qu.:-87.66   1st Qu.:41.88   1st Qu.:-87.66
 Median :41.90   Median :-87.64   Median :41.90   Median :-87.64
 Mean   :41.90   Mean   :-87.65   Mean   :41.90   Mean   :-87.65
 3rd Qu.:41.93   3rd Qu.:-87.63   3rd Qu.:41.93   3rd Qu.:-87.63
 Max.   :42.07   Max.   :-87.52   Max.   :42.37   Max.   :  0.00
                                  NA's   :5795    NA's   :5795
 member_casual           date                month                day

 Length:5779444     Min.   :2022-07-01   Length:5779444
Length:5779444
 Class :character   1st Qu.:2022-08-25   Class :character
Class :character
```

```
 Mode  :character   Median :2022-11-02   Mode  :character
Mode  :character
                    Mean   :2022-12-12

                    3rd Qu.:2023-04-21

                    Max.   :2023-06-30



     year            day_of_week          ride_length
 Length:5779444     Length:5779444     Min.   :-10353.35
 Class :character   Class :character   1st Qu.:     5.52
 Mode  :character   Mode  :character   Median :     9.70
                                       Mean   :    18.34
                                       3rd Qu.:    17.32
                                       Max.   : 41387.25


# Now we will remove those rows of ride_length which which are less
than 0 minute and above 1440 minutes to make our data more accurate.
combined_data_v2 <- combined_data[combined_data$ride_length >= 0 &
combined_data$ride_length <= 1440, ]

# Checking the combined_data_v2
dim(combined_data_v2)
summary(combined_data_v2)

[1] 5774250      19

   ride_id           rideable_type       started_at           ended_at

 Length:5774250     Length:5774250     Length:5774250
Length:5774250
 Class :character   Class :character   Class :character
Class :character
 Mode  :character   Mode  :character   Mode  :character
Mode  :character









 start_station_name start_station_id   end_station_name
end_station_id
 Length:5774250     Length:5774250     Length:5774250
Length:5774250
 Class :character   Class :character   Class :character
```

```
Class :character
 Mode  :character   Mode  :character   Mode  :character
Mode  :character
```

```
   start_lat       start_lng         end_lat         end_lng
 Min.   :41.64   Min.   :-87.87   Min.   : 0.00   Min.   :-88.16
 1st Qu.:41.88   1st Qu.:-87.66   1st Qu.:41.88   1st Qu.:-87.66
 Median :41.90   Median :-87.64   Median :41.90   Median :-87.64
 Mean   :41.90   Mean   :-87.65   Mean   :41.90   Mean   :-87.65
 3rd Qu.:41.93   3rd Qu.:-87.63   3rd Qu.:41.93   3rd Qu.:-87.63
 Max.   :42.07   Max.   :-87.52   Max.   :42.37   Max.   :  0.00
                                  NA's   :846     NA's   :846
 member_casual           date                month                day

 Length:5774250     Min.   :2022-07-01   Length:5774250
Length:5774250
 Class :character   1st Qu.:2022-08-25   Class :character
Class :character
 Mode  :character   Median :2022-11-02   Mode  :character
Mode  :character
                    Mean   :2022-12-12

                    3rd Qu.:2023-04-21

                    Max.   :2023-06-30
```

```
     year           day_of_week          ride_length
 Length:5774250     Length:5774250     Min.   :   0.00
 Class :character   Class :character   1st Qu.:   5.50
 Mode  :character   Mode  :character   Median :   9.70
                                       Mean   :  15.34
                                       3rd Qu.:  17.28
                                       Max.   :1439.93
```

So, after removal ride_length data >=0 & <=1440, 5194 data reduced which is .09%. 846 NAs still existing in end_lng and end_lat. After doing this our data is much more clean now.

Analyze Step:

Now Our data is clean and organized and ready to use for descriptive analysis

```r
# Create a new dataframe with the mean ride_length for casual and
member riders
combined_data_v2_mean <- combined_data_v2 %>%
  group_by(member_casual) %>%
  summarise(mean_ride_length=mean(ride_length))
print(combined_data_v2_mean)
```

```
# A tibble: 2 × 2
  member_casual mean_ride_length
  <chr>                    <dbl>
1 casual                    20.5
2 member                    12.1
```

```r
# Calculation of max ride_length for casual and member riders
combined_data_v2 %>%
  group_by(member_casual) %>%
  summarise(max_ride_length=max(ride_length))
```

```
  member_casual max_ride_length
1 casual              1439.933
2 member              1439.833
```

```r
# Create a new dataframe with the day_of_week and member_casual
columns

combined_data_v2_day_of_week <- combined_data_v2 %>%
select(day_of_week, member_casual)

# Calculation of mode of day_of_week for casual riders
casual_mode_day_of_week <- combined_data_v2_day_of_week %>%
  filter(member_casual=="casual") %>%
  count(day_of_week) %>%
  arrange(desc(n)) %>%
  slice(1) %>%
  pull(day_of_week)

# Calculation of mode of day_of_week for member riders
member_mode_day_of_week <- combined_data_v2_day_of_week %>%
  filter(member_casual=="member") %>%
  count(day_of_week) %>%
  arrange(desc(n)) %>%
  slice(1) %>%
  pull(day_of_week)

# Print the results
print(paste("The mode of day of week for casual riders is",
casual_mode_day_of_week))
print(paste("The mode of day of week for member riders is",
member_mode_day_of_week))
```

```
[1] "The mode of day of week for casual riders is Saturday"
[1] "The mode of day of week for member riders is Wednesday"
```

```
## Calculate casual and member rides number
casual_ride_number <- combined_data_v2 %>%
  filter(member_casual == "casual") %>%
  nrow()

member_ride_number <- combined_data_v2 %>%
  filter(member_casual == "member") %>%
  nrow()

print(paste("Number of casual riders:", casual_ride_number))
print(paste("Number of member riders:", member_ride_number))
```

```
[1] "Number of casual riders: 2239899"
[1] "Number of member riders: 3534351"
```

Share Step:

We have already find some answers of how annual members and casual riders use Cyclistic bike differently.

Our question was how do annual members and casual riders use Cyclistic bikes differently.

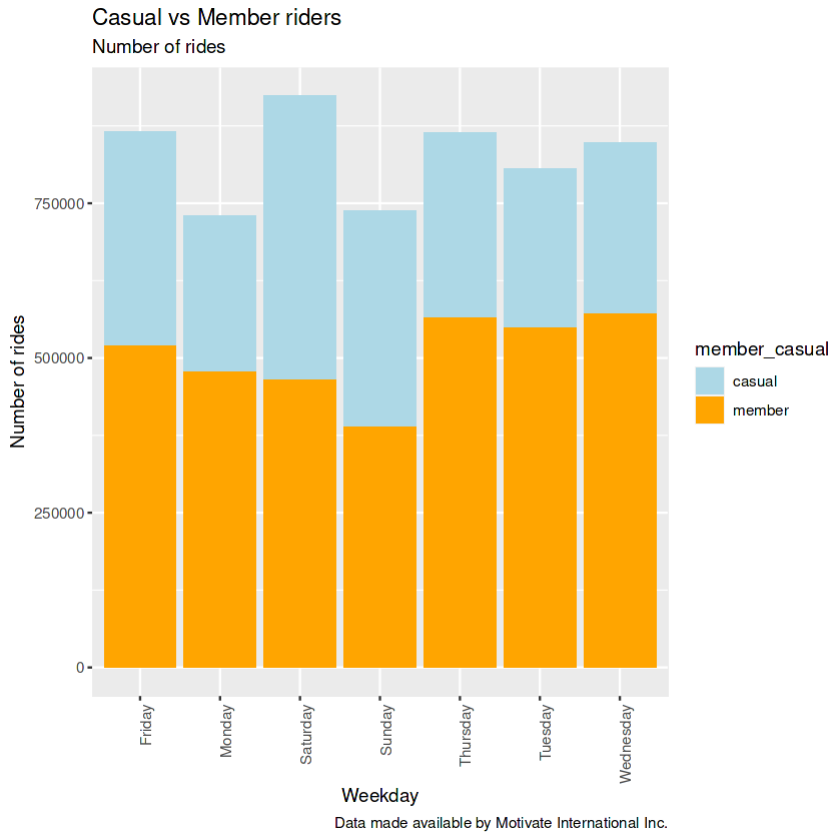
Now we will create some visualization.

```
# Visualization of weekday vs number of rides as per type of riders
# We need to create a data frame for the plot
rider_weekday <- combined_data_v2 %>%
  group_by(member_casual, day_of_week) %>%
  summarise(number_of_rides=n())

# 1st plot

ggplot(rider_weekday, aes(x=day_of_week, y=number_of_rides,
fill=member_casual))+
  geom_bar(stat="Identity")+
  labs(title="Casual vs Member riders", subtitle="Number of rides",
       caption=paste0("Data made available by Motivate International
Inc."))+
       xlab("Weekday")+
       ylab("Number of rides")+
         scale_fill_manual(values=c("lightblue", "orange"))+
  theme(axis.text.x=element_text(angle=90, hjust=1))
```

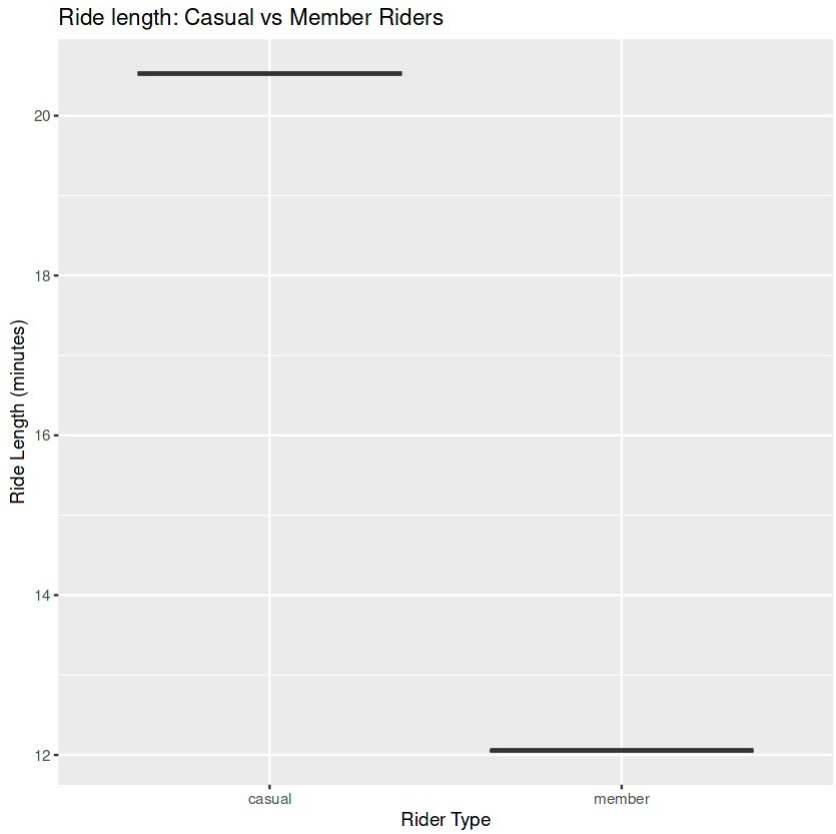
```
`summarise()` has grouped output by 'member_casual'. You can override
using the
`.groups` argument.
```



Plot is showing that number of rides of member riders are much higher than casual riders.

```
# Visualization of Box plot of the ride length by type of rider
# 1st we will create a data frame for the plot
ride_length_by_rider_type <- combined_data_v2 %>%
  group_by(member_casual) %>%
  summarise(ride_length=mean(ride_length))

# 2nd plot
ggplot(data=ride_length_by_rider_type)+
  geom_boxplot(mapping=aes(x=member_casual, y=ride_length))+
  labs(title="Ride length: Casual vs Member Riders",
       x="Rider Type",
       y="Ride Length (minutes)")
```

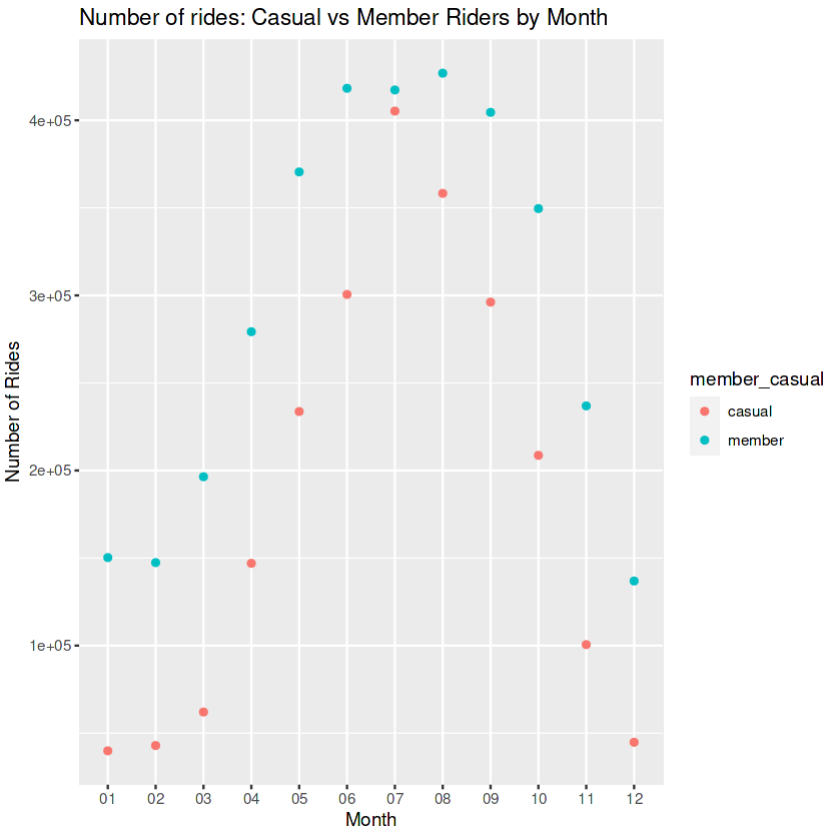


Box plot is showing that mean ride length of casual riders is well above member riders. Casual riders mean ride length is 20.5 minutes and member riders mean ride length is 12.1

```
# Visualization of point plot of the number of rides by months and
rider type
# Data frame for plot
rides_by_month_rider_type <- combined_data_v2 %>%
  group_by(month, member_casual) %>%
  summarise(number_of_rides=n())

# 3rd plot
ggplot(rides_by_month_rider_type, aes(x=month, y=number_of_rides,
color=member_casual))+
  geom_point()+
  labs(title="Number of rides: Casual vs Member Riders by Month",
       x="Month",
       y="Number of Rides")
```

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



It's clear that casual riders are more likely to use Cyclistic bikes in the warmer months, while member riders are more likely to use Cyclistic bikes in the colder months.

Act Step:

I have three recommendations for how to take action based on my key findings in analysis. Each of these recommendations takes the original problem into consideration.

The problem to be solved: While Cyclistic's flexible pricing schemes are successful at attracting new customers, many of these remain casual customers and do not bring in the same profit that Cyclistic members do.

3 Recommendations: Based on the analysis and visualizations of Cyclistic bike-share 12 months data, here are my top 3 recommendations:

1. Target casual riders with marketing campaigns during the warmer months. Casual riders are more likely to use Cyclistic bikes in the warmer months, so it makes sense to target them with marketing campaigns during this time. This could include things like advertising in local digital magazines and newspapers, or sponsoring events that are popular with casual riders.

2. Offer discounts or incentives for annual memberships. Annual members are more profitable for Cyclistic than casual riders, so it makes sense to encourage more people to sign up for annual memberships. This could include offering discounts for annual memberships, or giving members access to exclusive benefits, such as free

bike repairs or discounts on bike accessories. Also, members may categorize as Platinum, Gold and Silver.

3. Make it easier for casual riders to become annual members. Some casual riders may be hesitant to sign up for an annual membership because they're not sure if they'll use it enough. Cyclistic could make it easier for these riders to become annual members by offering a free trial period, or by allowing them to cancel their membership at any time.