Google Data Analytics Capstone Project

Shaun Partridge

2023-06-01

Ask

Cyclistic, an inclusive bike-sharing company with over 5,800 bicycles and 600 dock stations throughout Chicago, believes the key to the companies future growth is to convert casual riders into annual members. The business task for this data analysis is to answer the following question: How do annual members and casual riders use Cyclistic bikes differently?

Prepare

The data used in this analysis is made available by Motivate International Inc. under this license (https://ride.divvybikes.com/data-license-agreement). The files downloaded consist of 12 data sets of bike sharing data ranging from May 2022 to April 2023 and are stored in a single folder. The data was found to have high integrity after the columns of each data set were examined, as well as being unbiased and ROCCC (https://www.coursera.org/learn/data-preparation/lecture/lHirM/what-is-bad-data). The code below will install and load the appropriate libraries for analysis, in addition to reading in each .csv file into a data frame.

```
install.packages('tidyverse', repos = "http://cran.us.r-project.org")
```

```
## package 'tidyverse' successfully unpacked and MD5 sums checked
##
## The downloaded binary packages are in
##  C:\Users\spart\AppData\Local\Temp\RtmpieTTGV\downloaded_packages
```

```
install.packages('janitor', repos = "http://cran.us.r-project.org")
```

```
## package 'janitor' successfully unpacked and MD5 sums checked
##
## The downloaded binary packages are in
##  C:\Users\spart\AppData\Local\Temp\RtmpieTTGV\downloaded_packages
```

```
install.packages('lubridate', repos = "http://cran.us.r-project.org")
```

```
## package 'lubridate' successfully unpacked and MD5 sums checked
```

```
## Warning: cannot remove prior installation of package 'lubridate'
```

```
## Warning in file.copy(savedcopy, lib, recursive = TRUE): problem copying
## C:\Users\spart\AppData\Local\R\win-library\4.3\00LOCK\lubridate\libs\x64\lubridate.dll
## to
## C:\Users\spart\AppData\Local\R\win-library\4.3\lubridate\libs\x64\lubridate.dll:
## Permission denied
```

```
## Warning: restored 'lubridate'
```

```
##
## The downloaded binary packages are in
##  C:\Users\spart\AppData\Local\Temp\RtmpieTTGV\downloaded_packages
```

```
library(tidyverse)
library(janitor)
library(lubridate)
```

```
# Reading in each .csv file into its own data frame
df_202205_trip_data <- read.csv('C:/Users/spart/OneDrive/Desktop/BikeShareData/202205-divvy-tripdata.csv')
df_202206_trip_data <- read.csv('C:/Users/spart/OneDrive/Desktop/BikeShareData/202206-divvy-tripdata.csv')
df_202207_trip_data <- read.csv('C:/Users/spart/OneDrive/Desktop/BikeShareData/202207-divvy-tripdata.csv')
df_202208_trip_data <- read.csv('C:/Users/spart/OneDrive/Desktop/BikeShareData/202208-divvy-tripdata.csv')
df_202209_trip_data <- read.csv('C:/Users/spart/OneDrive/Desktop/BikeShareData/202209-divvy-tripdata.csv')
df_202210_trip_data <- read.csv('C:/Users/spart/OneDrive/Desktop/BikeShareData/202210-divvy-tripdata.csv')
df_202211_trip_data <- read.csv('C:/Users/spart/OneDrive/Desktop/BikeShareData/202211-divvy-tripdata.csv')
df_202212_trip_data <- read.csv('C:/Users/spart/OneDrive/Desktop/BikeShareData/202212-divvy-tripdata.csv')
df_202301_trip_data <- read.csv('C:/Users/spart/OneDrive/Desktop/BikeShareData/202301-divvy-tripdata.csv')
df_202302_trip_data <- read.csv('C:/Users/spart/OneDrive/Desktop/BikeShareData/202302-divvy-tripdata.csv')
df_202303_trip_data <- read.csv('C:/Users/spart/OneDrive/Desktop/BikeShareData/202303-divvy-tripdata.csv')
df_202304_trip_data <- read.csv('C:/Users/spart/OneDrive/Desktop/BikeShareData/202304-divvy-tripdata.csv')
```

Process

RStudio will be used for this data analysis due to some of the files being too large to upload into Google Cloud BigQuery, Excel, or Google Sheets unless they were broken down into smaller files, in addition to the data cleaning and visualization packages available in R. The data cleaning process involved combining all data frames into one data frame, removing rows with empty columns, transforming data types of columns, and creating new columns to help with the analysis.

The following code snippets will:

- Combine all of the data frames into one large data frame
- Convert the data types of columns, **started_at** and **ended_at**, to calculate the time difference of each ride
- Create new columns: **trip_duration**, **day_of_week**, and **month**

```
# Creating a new data frame to combine all data frames into one
all_trip_data_df <- rbind(df_202205_trip_data,df_202206_trip_data,df_202207_trip_data,df_202208_trip_data,d
f_202209_trip_data,df_202210_trip_data,df_202211_trip_data,df_202212_trip_data,df_202301_trip_data,df_20230
2_trip_data,df_202303_trip_data,df_202304_trip_data)
```

```
# Creating trip_duration, day_of_week, and month column

# trip_duration value comes from using difftime function and passing in ended_at and started_at cols as arg
s and making the units minutes; then rounding the result to decimal places
all_trip_data_df$trip_duration <- round(difftime(all_trip_data_df$ended_at,all_trip_data_df$started_at,unit
s = "mins"),2)

# day_of_week value comes from using wday function and passing in started_at column as arg; label and abbr
= TRUE means store values as 'Mon', 'Tues', etc.
all_trip_data_df$day_of_week <- wday(all_trip_data_df$started_at, label=TRUE, abbr=TRUE)

# month value comes from formatting the started_at column as a Month, ie. '01', '02', .. '12'
all_trip_data_df$month <- format(as.Date(all_trip_data_df$started_at),'%m')
```

```
# Final clean data frame with incomplete rows and rows with trip duration <= 0 removed
cleaned_df <- all_trip_data_df[!(all_trip_data_df$trip_duration<=0) & all_trip_data_df$start_station_name !
= "",]
```

Analyze

We'll start off the analyzing phase by first calculating the average trip duration for every day of the week for each type of rider. The result will be stored in a new data frame, **res**, to plot the results later. Then, we'll calculate the total number of each type of rider for both, each day of the week and month.

```
# Creating a result data frame to plot the average trip duration of each rider group for each day of the we
ek
res <- setNames(aggregate(cleaned_df$trip_duration,by = list(cleaned_df$day_of_week,cleaned_df$member_casua
l),FUN=mean),
                c("day_of_week","member_casual","avg_trip_duration"))
res
```

```
##    day_of_week member_casual avg_trip_duration
## 1          Sun        casual     36.29056 mins
## 2          Mon        casual     30.82467 mins
## 3          Tue        casual     27.53345 mins
## 4          Wed        casual     26.22851 mins
## 5          Thu        casual     26.78919 mins
## 6          Fri        casual     30.01001 mins
## 7          Sat        casual     34.96278 mins
## 8          Sun        member     13.99083 mins
## 9          Mon        member     12.04542 mins
## 10         Tue        member     11.99733 mins
## 11         Wed        member     11.93944 mins
## 12         Thu        member     12.16597 mins
## 13         Fri        member     12.42425 mins
## 14         Sat        member     14.11994 mins
```

```
# Group the count of each type of rider by day_of_week and member_casual
cleaned_df %>%
  group_by(day_of_week, member_casual) %>%
  count()
```

```
## # A tibble: 14 × 3
## # Groups:   day_of_week, member_casual [14]
##    day_of_week member_casual      n
##    <ord>       <chr>          <int>
##  1 Sun         casual        334664
##  2 Sun         member        342158
##  3 Mon         casual        235851
##  4 Mon         member        420866
##  5 Tue         casual        231546
##  6 Tue         member        474093
##  7 Wed         casual        241226
##  8 Wed         member        483167
##  9 Thu         casual        269648
## 10 Thu         member        484053
## 11 Fri         casual        296006
## 12 Fri         member        425717
## 13 Sat         casual        402793
## 14 Sat         member        384747
```

```
# Group the count of each type of rider by month and member_casual
cleaned_df %>%
  group_by(month, member_casual) %>%
  count()
```

```
## # A tibble: 24 × 3
## # Groups:   month, member_casual [24]
##    month member_casual      n
##    <chr> <chr>          <int>
##  1 01    casual         33453
##  2 01    member        130119
##  3 02    casual         36696
##  4 02    member        128268
##  5 03    casual         52900
##  6 03    member        169849
##  7 04    casual        125091
##  8 04    member        237650
##  9 05    casual        243105
## 10 05    member        305001
## # ℹ 14 more rows
```

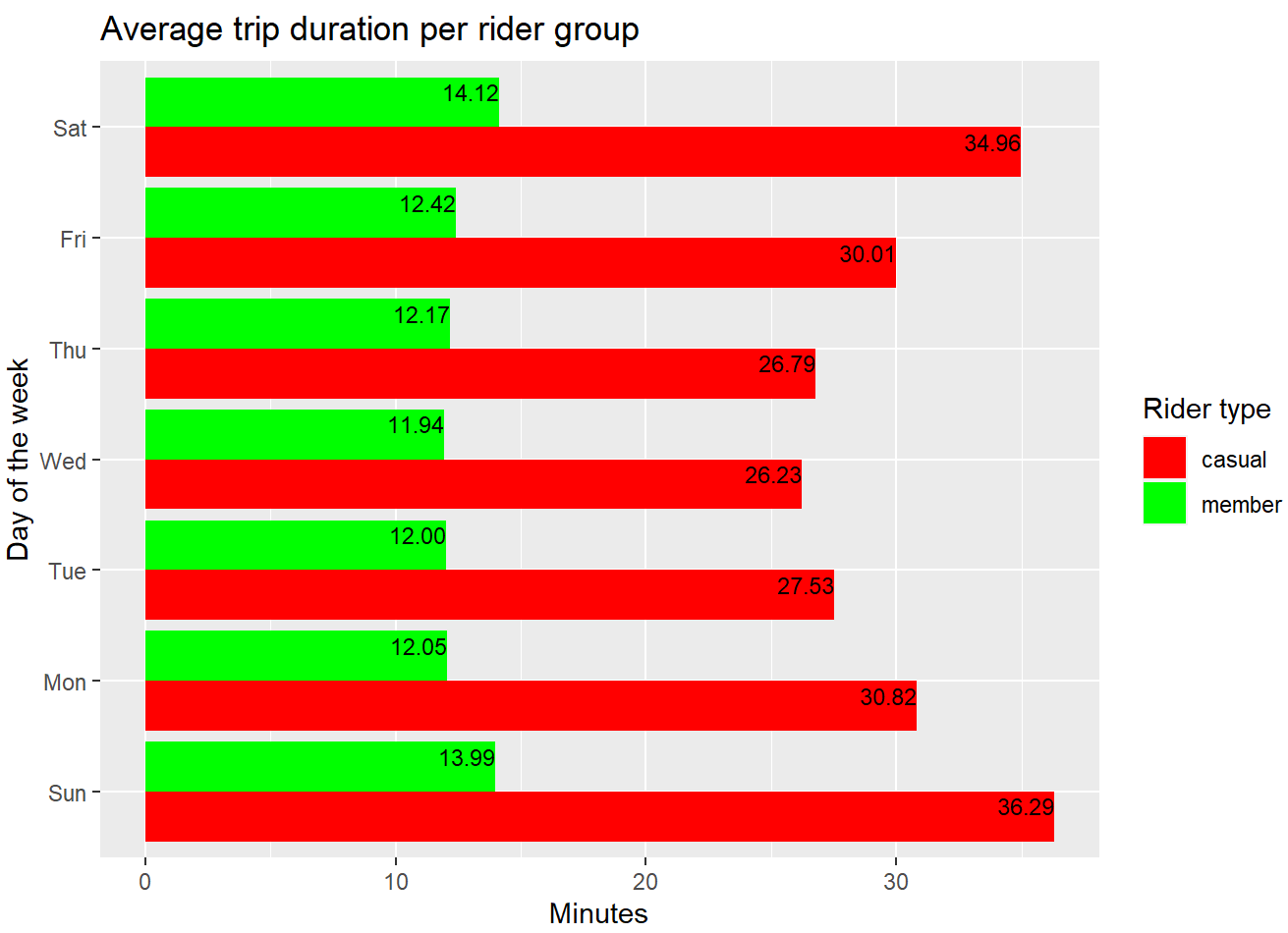
After running the above code snippets we notice the following about the data:

- Casual riders on average, have longer bike rides on a daily basis compared to member riders
- There are more member riders that bike everyday of the week except for Saturday
- Member riders bike more than casual riders monthly, however there is an increase of both types of riders during the warmer months of the year (June-September).

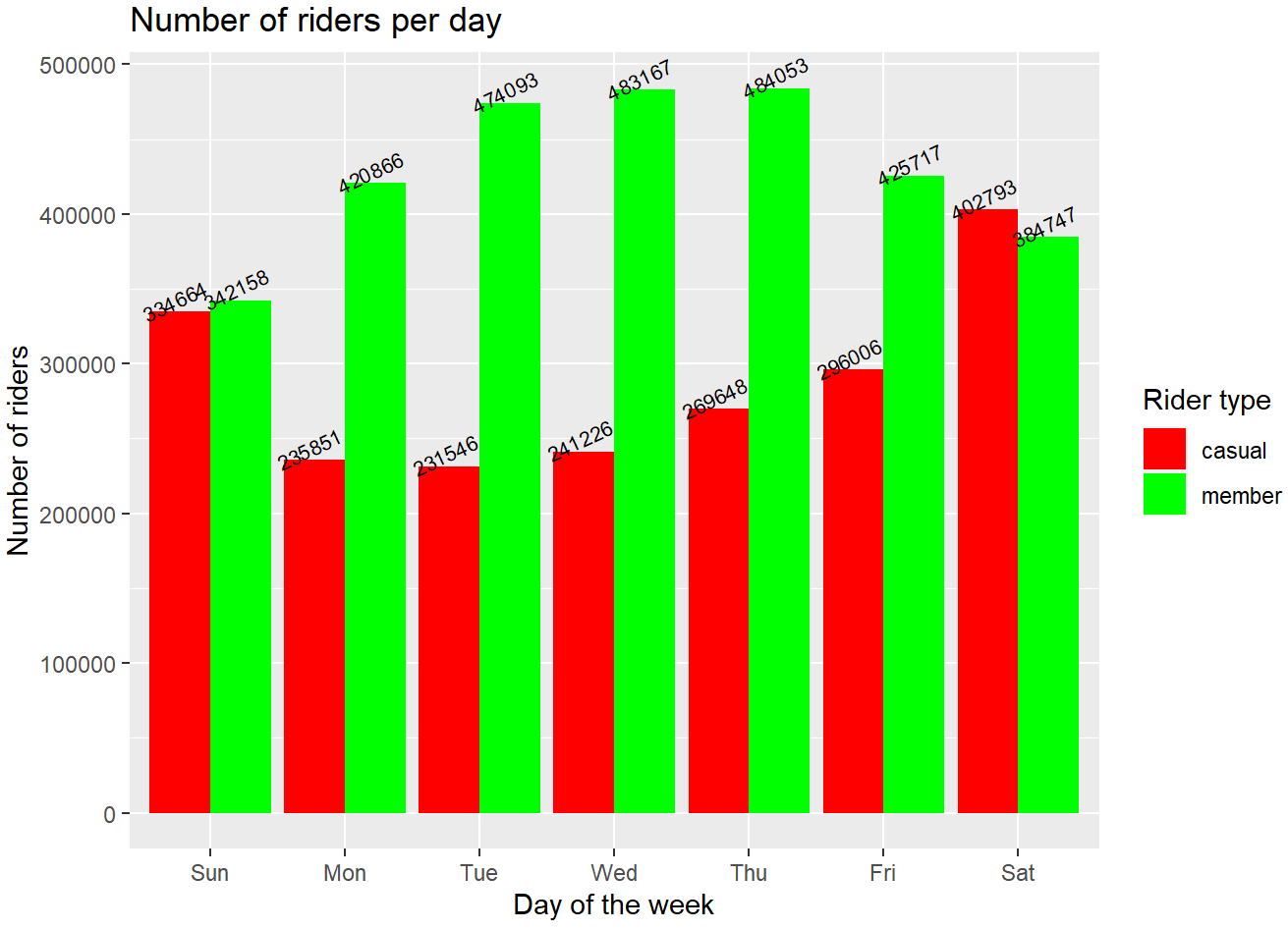
Share

Now we'll use data visualizations to display our insights using ggplot2, a data visualization library, and create a plot for each one.

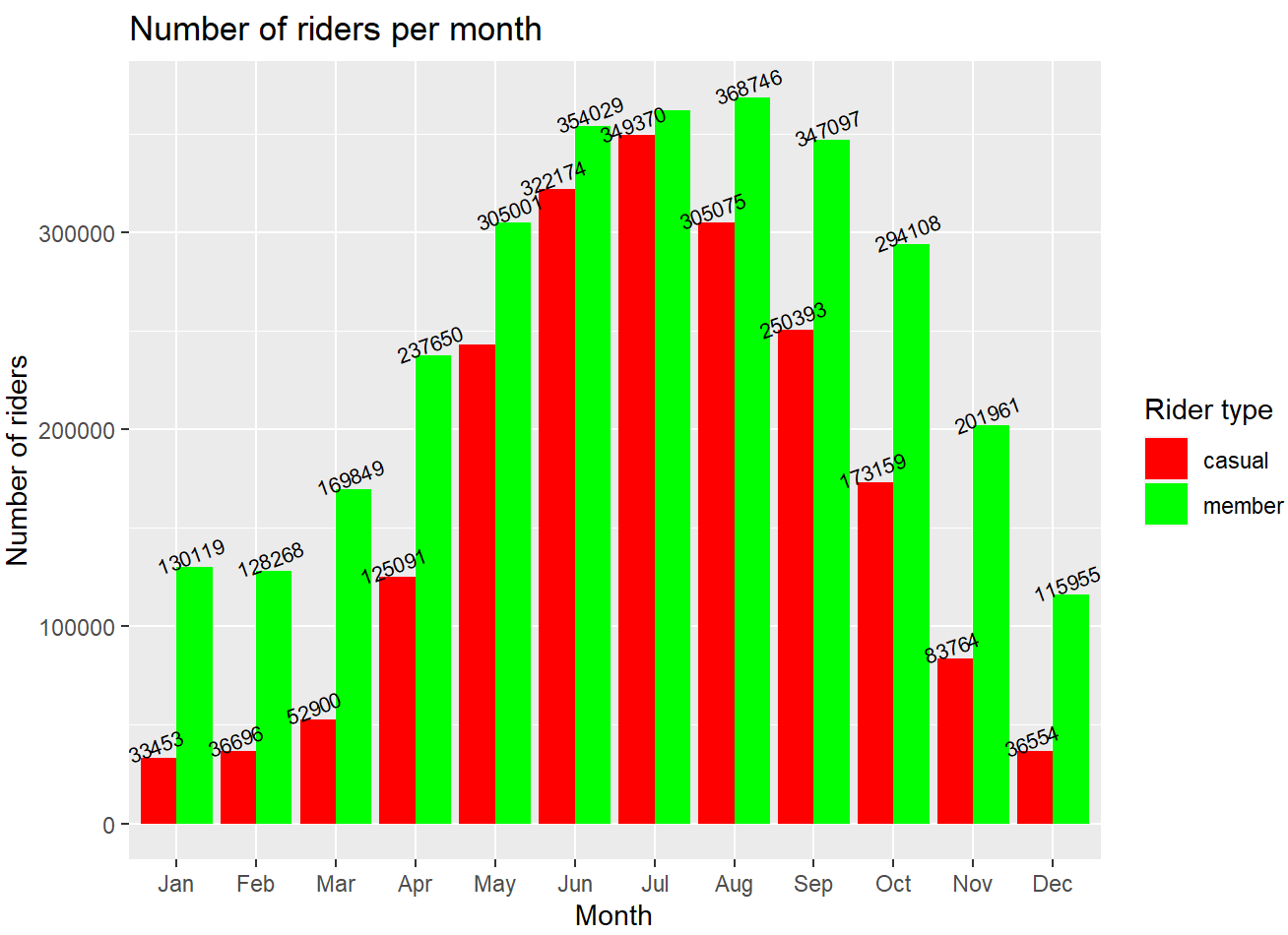
Plot to display the average ride duration per rider group for each day of the week



Plot to display the count of each rider group for each day of the week



Plot to display the count of each rider group for every month



Act

We can conclude from the analysis that member riders bike more frequently than casual riders. However, casual riders bike for much longer periods of time compared to member riders. It's also worth mentioning that there are more of both groups of riders during the months, June, July, August, and September. With these insights, stakeholders should consider using ads and promotions geared towards casual riders during the summer at bike stations to get them to sign up for a membership. That way they can bike whenever and how long they want instead of trying to get their money's worth with every ride, as well as help secure the company's future growth.