Case Study:How Does a Bike-Share Navigate Speedy Success?

Clarke Li

2022-08-18

Case Study: How Does a Bike-Share Navigate Speedy Sucess?

Outdoor activities vector created by pch.vector - www.freepik.com _____ ## Introduction:

This project is a data analysis project for a fictional company, Cyclists, A bike-share company in Chicago. The marketing director believes the company's future success depends on converting the casual customer to Annual subscribed users.

As the data analyst of the company, the tasks in the project include discovering the different usage between casual users and the subscribed user; Basic on the insight from the analysis, provide a recommendation on Marketing strategy to turn the casual users into subscribed users.

Background of the fiction company

Cyclistic started its bike-share offering in 2016. They have grown to own a fleet of 5,284 bikes that are tracked by GPS and located in 692 stations in Chicago.

The company's CEO has set a clear goal to boost the business's profits: "Design marketing strategies aimed at converting casual riders into annual members. The business task of designing market strategies has been handed down to the Cyclistic marketing analytics team. The team will make a recommendation on the marketing program. And the program will be approved by the executive team in Cyclistic before rollout

The Bussiness Case Study

As the goal of the project is to develop effective marketing strategies for the business, there will be an in-depth data analysis on the historical trip data in previous 12 month, which is collected from the customer by the Cyclistic over years of operation(in the fiction company time line). The analytic team should answer three questions that will guide the future marketing program:

- 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?

Task One: Preparation

This session is the beginning of the whole project. The main tasks include collecting and related information, checking the bias and credibility of the data collected, and preparing the data by reviewing and further processing data.

Organise the data directary

The data set is download from following LINK.

The datasets are downloaded in Zip format. There are two types of data in the unzipped dataset folder, the Divvy Trips data and the Divvy stations data. The Divvy Trips data has all the user and trip information and is saved in the CSV format. The Divvy Stations data, on the other hand, contains all the existing bicycle stations' geo-location information and is saved as the ".xsl".

For the analytic need, only Divvy_Trips data from previous years are needed. Therefore, all the Divvy Trips will be copied and transferred to a separate folder in the repository. A new folder is created in the root directory, and the folder is renamed to **Cyclistic_trip_data**. Then, Trips data files were moved to this folder.

Data credibility, Integrity and its issues

Because this is the friction company, the actual data is made available by Motivate International Inc. under this license. Moreover, the information is collected under a different entity, Lyft Bikes and Scooters, LLC("Bikeshare"). This public data allowed the user to explore the different types of customers using Cyclistic. In the notice, the license prohibits using personally identifiable information in the dataset.

In the data folder, the time of data is range from 2013 to present. However, in the time line of our frinction company, it started the business in the year of 2016. Hence the data before the 2016 will be automatically inore. On the hand, there is some data format different in previous years of data collected. Therefore, as preset in the business case, only the data from past 12 months will be used in this analynadic process.

In these dataset, it has information about user type and travel time, and the travel distance. Those infomation will be used to find the patent of different account usage.

Tools for analytic

In this project, the R studio will be the primary tool working.Because The R studio is a comprehensive equipment for data analysis and data engineering. It has all the functions available in data cleaning, filtering, algebraic calculation, graphic drawing, etc.

On the other hand, Git will be used as version control. Git will help track the change and prevent unintended alternation on the work saved. That measure will provide better security for the project.

Use R studio to load the relatived data into the dataframe. But right before all of this, the configuration of the working enviroment shall be setup, installed neccessary package, laod all necssary liberary.

Task two: Data Processing

Setting R

Use R studio to load the relatived data into the dataframe. But right before all of this, the configuration of the working enviroment shall be setup, installed neccessary package, laod all necssary liberary.

```
##*** Setup the working enviroment *****##
## This code is for installing and loading required package for the analyltic

#*** install required package
#install.packages("tidyverse")
#install.package("data")
#install.packages("here")
#install.packages("skimr")
#install.packages("lubridate")
#install.packages("ggplot2")
#** Load the database liberary

library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.2 --
## v ggplot2 3.3.6     v purrr   0.3.4
## v tibble  3.1.8     v dplyr   1.0.9
## v tidyr   1.2.0     v stringr 1.4.0
## v readr   2.1.2     v forcats 0.5.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
  #coherent system of packages for data manipulation, exploration and visualization

library(readr)
  #read csv function

library(data.table)
```

```
##
## Attaching package: 'data.table'
##
## The following objects are masked from 'package:dplyr':
##
##     between, first, last
##
## The following object is masked from 'package:purrr':
##
##     transpose
```

```
library("skimr")
  #provide summary statistics about variables in data frames, tibbles, data tables and vectors.

library("lubridate")
```

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

```
##
## The following objects are masked from 'package:data.table':
##
##     hour, isoweek, mday, minute, month, quarter, second, wday, week,
##     yday, year
##
## The following objects are masked from 'package:base':
##
##     date, intersect, setdiff, union
```

```
#tools to manipulate dates times to the forms easy to work with.

#* Package 'here'
#* Constructs paths to your project's files.
#* Declare the relative path of a file within your project with 'i_am()'. Use the 'here()' function as

library(here)
```

```
## here() starts at /Users/junli/Desktop/Data Science Hands-on course/Google_analytic_cert/Modual_8_Goog
```

```
#* Visalization **#
library(ggplot2)
```

Load dataset

Loading all the CSV files trip data from the **Cyclistic__trip__data** folder.

Problem in loading the CSV files At the begging of the process, all the CSV files had been unzipped and moved into the new folder. But there was a problem spotted in their naming system. That was, all the files did not follow a unique naming convention. Most importantly, there is one more problem with the name of the files. No conventional charter has been used in the files name, which is the dash "-" had been, which should be replaced with the underscore "_". And the file name starts shall start from a letter instead of the number

For example, the files name "202004-divvy-tripdata.csv" is not recommand in the naming method, it is recommanded to check to "Divvy_tripdata_202004.csv".

```
#loading multiple CSV files into one dataframe
#** list.files() functions produce a character vector of the names of files or directories in the named
##*

trips_df <-
  list.files(path="./Cyclistic_trip_data/",pattern="*.csv",full.names = TRUE)%>%
  map_df(~fread(.))
head(trips_df)
```

```
##             ride_id rideable_type          started_at            ended_at
## 1: 0A1B623926EF4E16   docked_bike 2021-07-02 14:44:36 2021-07-02 15:19:58
## 2: B2D5583A5A5E76EE  classic_bike 2021-07-07 16:57:42 2021-07-07 17:16:09
## 3: 6F264597DDBF427A  classic_bike 2021-07-25 11:30:55 2021-07-25 11:48:45
## 4: 379B58EAB20E8AA5  classic_bike 2021-07-08 22:08:30 2021-07-08 22:23:32
## 5: 6615C1E4EB08E8FB electric_bike 2021-07-28 16:08:06 2021-07-28 16:27:09
## 6: 62DC2B32872F9BA8 electric_bike 2021-07-29 17:09:08 2021-07-29 17:15:00
##              start_station_name start_station_id               end_station_name
## 1: Michigan Ave & Washington St            13001   Halsted St & North Branch St
## 2:   California Ave & Cortez St            17660           Wood St & Hubbard St
## 3:         Wabash Ave & 16th St           SL-012           Rush St & Hubbard St
```

```
## 4:   California Ave & Cortez St            17660        Carpenter St & Huron St
## 5:   California Ave & Cortez St            17660 Elizabeth (May) St & Fulton St
## 6:   California Ave & Cortez St            17660  Albany Ave & Bloomingdale Ave
##    end_station_id start_lat start_lng  end_lat   end_lng member_casual
## 1:   KA1504000117  41.88398 -87.62468 41.89937 -87.64848        casual
## 2:          13432  41.90036 -87.69670 41.88990 -87.67147        casual
## 3:   KA1503000044  41.86038 -87.62581 41.89017 -87.62619        member
## 4:          13196  41.90036 -87.69670 41.89456 -87.65345        member
## 5:          13197  41.90035 -87.69668 41.88659 -87.65839        casual
## 6:          15655  41.90033 -87.69674 41.91389 -87.70513        casual
```

```
#colnames(trips_df)
```

Clean and review the dataframe

This process include clean and inspect the data. It is preprocess before furture analysit.

```
number_of_colums <- length(colnames(trips_df))
# number_of_colums # The number of columes in the dataframe

#check the if there is any missing value in any of the columns
for(y in 1:number_of_colums){

  missing_number <- sum(is.na(subset(x=trips_df,,y)))
  #missing_number <- sum(is.null(trips_df[,x]) == TRUE)
  cat("There are ",missing_number , "miss values in this column",colnames(trips_df)[y],"\n")
}
```

```
## There are  0 miss values in this column ride_id
## There are  0 miss values in this column rideable_type
## There are  0 miss values in this column started_at
## There are  0 miss values in this column ended_at
## There are  0 miss values in this column start_station_name
## There are  0 miss values in this column start_station_id
## There are  0 miss values in this column end_station_name
## There are  0 miss values in this column end_station_id
## There are  0 miss values in this column start_lat
## There are  0 miss values in this column start_lng
## There are  6321 miss values in this column end_lat
## There are  6321 miss values in this column end_lng
## There are  0 miss values in this column member_casual
```

The result show there are valuse in end_lat and end_lng. This might means there is no bike parking back to the station. However, this is no important for the analysis.

```
str(trips_df)
```

```
## Classes 'data.table' and 'data.frame':   6723873 obs. of  13 variables:
##  $ ride_id           : chr  "0A1B623926EF4E16" "B2D5583A5A5E76EE" "6F264597DDBF427A" "379B58EAB20E8A
##  $ rideable_type     : chr  "docked_bike" "classic_bike" "classic_bike" "classic_bike" ...
##  $ started_at        : POSIXct, format: "2021-07-02 14:44:36" "2021-07-07 16:57:42" ...
##  $ ended_at          : POSIXct, format: "2021-07-02 15:19:58" "2021-07-07 17:16:09" ...
##  $ start_station_name: chr  "Michigan Ave & Washington St" "California Ave & Cortez St" "Wabash Ave &
##  $ start_station_id  : chr  "13001" "17660" "SL-012" "17660" ...
##  $ end_station_name  : chr  "Halsted St & North Branch St" "Wood St & Hubbard St" "Rush St & Hubbard
##  $ end_station_id    : chr  "KA1504000117" "13432" "KA1503000044" "13196" ...
```

```
## $ start_lat         : num  41.9 41.9 41.9 41.9 41.9 ...
## $ start_lng         : num  -87.6 -87.7 -87.6 -87.7 -87.7 ...
## $ end_lat           : num  41.9 41.9 41.9 41.9 41.9 ...
## $ end_lng           : num  -87.6 -87.7 -87.6 -87.7 -87.7 ...
## $ member_casual     : chr  "casual" "casual" "member" "member" ...
## - attr(*, ".internal.selfref")=<externalptr>
```

Process with dataframe

In this part of work, there will be further process to the raw data, extracting useful information, perfoming calculation and reorgranise the dataframe.

create new columns Worked the duration for each ride in seconds, and store the result at the new column called ride_length. Use the "wday" function to workout the day of the week when people used the bike rental service, and creat a new column call "day of week". Created started_time column to record time the trip started.

```
#trips_df_14 <- mutate(trips_df, ride_length = ended_at - started_at)
#work out the duration by calculate the ended time minus started time
#creat another column record the time service started
#wday() is function of work the day in a week from the date of the year
trips_df_14 <- trips_df %>%
    mutate(
      ride_length = ended_at - started_at,
      day_of_week = wday(trips_df$started_at),
      time_of_day = hour(trips_df$started_at)
    )
```

Task Three: Analysis

Choosing paremeter of the analysis

In the previous task, there was preprocess on these data frames. The data frame is ready for the analysis task. There is some missing number in the columns of the end station coordination. That could be why the bike did not return to the station or illegal parking etc. However, this information is not necessarily crucial to our results. Therefore, these two columns will not store in the new data frame for the analytic progress.

Moreover, the "started_at" and "ended_at" columns recorded the of travel for each single trip. The started_at column could used to find out which location of the station is busier than the other. However, the ended_at has less signication in this praticular project. For the same reason, the "start_station_id","end_station_name","end_station_id", start_lat","start_lng","end_lat","end_lng" will be drop out from the dataframe for the final analysis. There are 8 column remained in the dataframe.

```
#create a new dataframe for the analysis.
analysis_df <- subset.data.frame(trips_df_14,select = c(ride_id,rideable_type,start_station_name,end_sta
colnames(analysis_df)
```

```
## [1] "ride_id"             "rideable_type"       "start_station_name"
## [4] "end_station_name"    "started_at"          "member_casual"
## [7] "ride_length"         "day_of_week"         "time_of_day"
```

The duration of ride

The duration of ride is too records the time each user used the service. However, there were some duration of trips is less than two minus(120 seconds). The user who is in these kind of ride is not on their purpose of communiting, they could just on a trial or simple change their mind after taken the service. Therefore, those samples were not the ideal sample to study the user behavior.

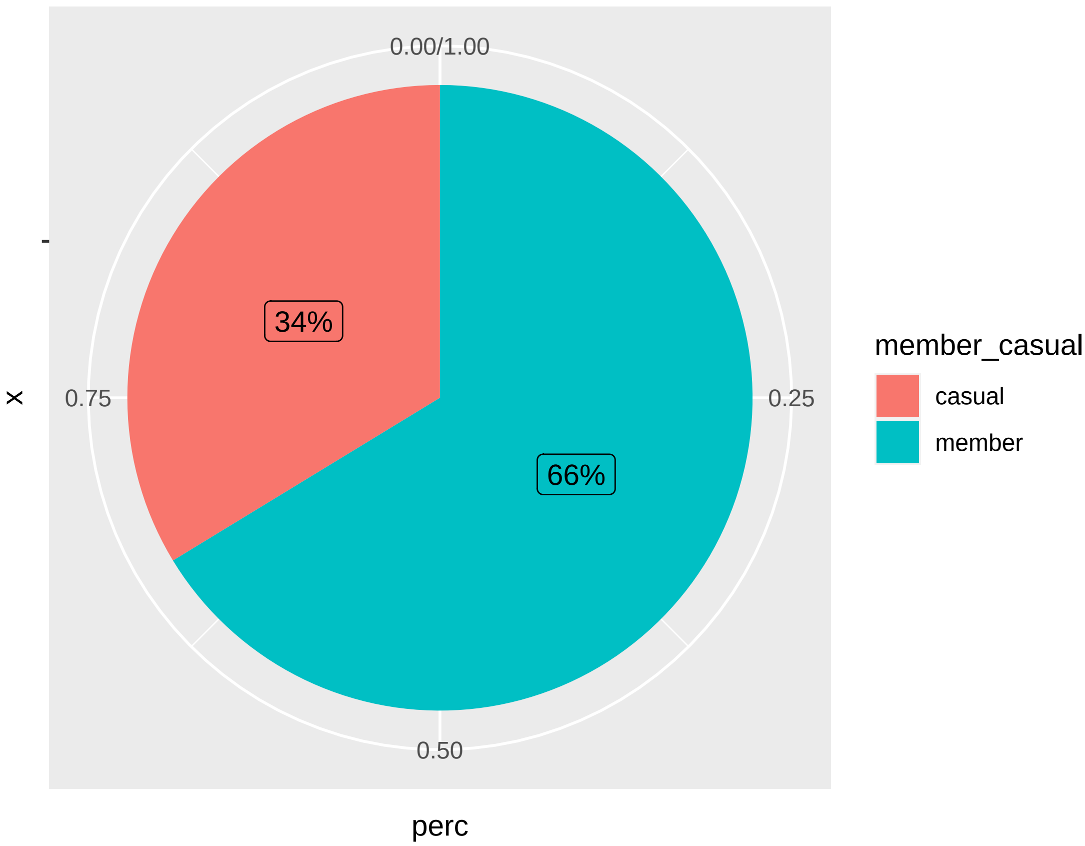
```
# The travel duration are less than 120 seconds
#analysis_df%>%
#  filter(ride_length <= 120)%>%
#  group_by(member_casual)

# work the percentage of the vast account with less 2 mins of ride
shot_trips_count <- analysis_df%>%
  filter(ride_length <= 120)%>%
  group_by(member_casual) %>%
  count()%>%
  ungroup()%>%
  mutate(perc = (n/sum(n)))%>%
  arrange(perc)%>%
  mutate(percents = scales::percent(perc))

rename(shot_trips_count, total_count = n)
```

```
## # A tibble: 2 x 4
##   member_casual total_count  perc percents
##   <chr>               <int> <dbl> <chr>
## 1 casual              75718 0.337 34%
## 2 member             148903 0.663 66%
```

```
#draw pie chart
ggplot(data = shot_trips_count, aes(x="", y=perc , fill=member_casual)) +
  geom_bar(stat="identity", width=1) +
  geom_label(aes(label = percents),
             position = position_stack(vjust = 0.5),
             show.legend = FALSE) +
  coord_polar("y", start=0)
```

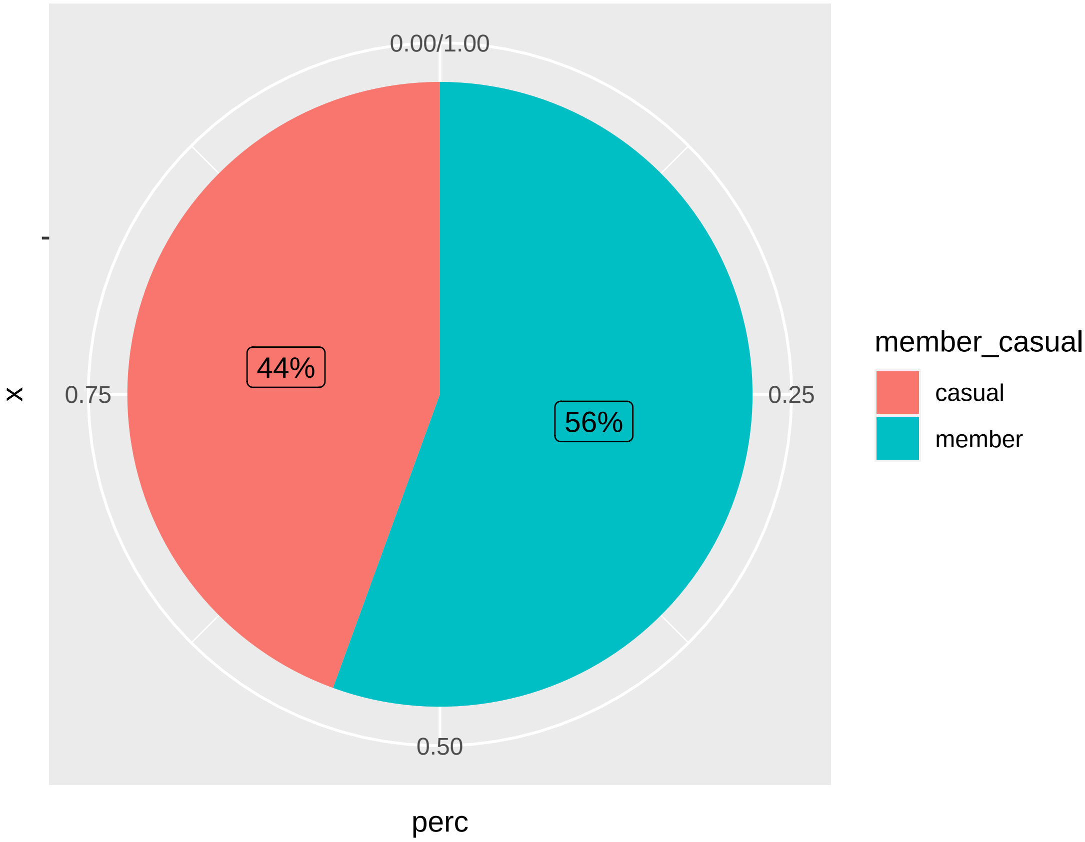



```r
#ggplot(data = shot_trips_count) +
#geom_col(mapping = aes(member_casual,n,fill = member_casual ))

# The durations are longer than 2 mins
long_trips_count <- analysis_df%>%
  filter(ride_length > 120)%>%
  group_by(member_casual) %>%
  count()%>%
  ungroup()%>%
  mutate(perc = (n/sum(n)))%>%
  arrange(perc)%>%
  mutate(percents = scales::percent(perc))
rename(long_trips_count,total_count = n)
```

```
## # A tibble: 2 x 4
##   member_casual total_count  perc percents
##   <chr>               <int> <dbl> <chr>
## 1 casual            2888564 0.444 44%
## 2 member            3610688 0.556 56%
```

```r
ggplot(data = long_trips_count, aes(x="", y=perc , fill=member_casual)) +
  geom_bar(stat="identity", width=1) +
  geom_label(aes(label = percents),
             position = position_stack(vjust = 0.5),
             show.legend = FALSE) +
  coord_polar("y", start=0)
```




```
#gplot(data = long_trips_count) +
#geom_col(mapping = aes(member_casual,n,fill = member_casual ))
#ggplot(data = shot_trips_count) +
#  geom_col(mapping = aes(x=member_casual,  fill = member_casual))
```

The result showed there are 2888564 ride have duration longer than 2 mins last 12 month for casual user, where as there are 3610688 ride for members. The percentage of causual rider who did less than 2 mins is about 34%. Meanwhile, it is about 44% for duration logner than 2 mins.

```
# skim_without_charts() function provides a detailed summary of the data
analysis_df%>%
  group_by(member_casual)%>%
  filter(ride_length > 120)%>%
  select(ride_length,member_casual)%>%
  skim_without_charts()
```

Table 1: Data summary

Name	Piped data
Number of rows	6499252
Number of columns	2
Column type frequency:	
difftime	1
Group variables	member_casual

Variable type: difftime

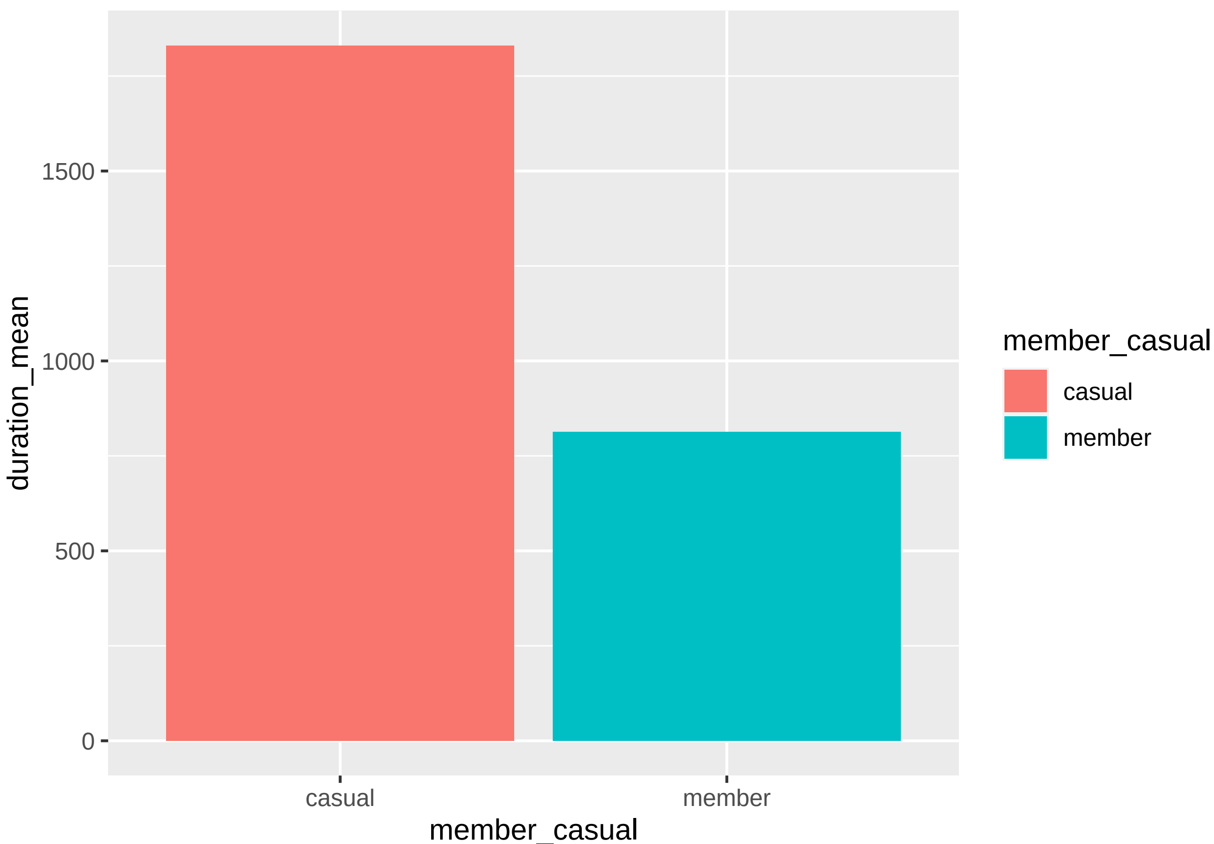
skim_variable	member_casual	n_missing	complete_rate	min	max	median	n_unique
ride_length	casual	0	1	121 secs	2946429 secs	909 secs	23345
ride_length	member	0	1	121 secs	93594 secs	572 secs	12103

```
# Work the mean of the duration for different type of users.
mean_of_ride <- analysis_df%>%
  group_by(member_casual)%>%
  filter(ride_length > 120)%>%
  select(ride_length,member_casual)%>%
  summarize(duration_mean = mean(ride_length))
mean_of_ride
```

```
## # A tibble: 2 x 2
##   member_casual duration_mean
##   <chr>         <drtn>
## 1 casual        1830.2333 secs
## 2 member         813.6321 secs
```

```
ggplot(data = mean_of_ride) +
geom_col(mapping = aes(member_casual,duration_mean,fill = member_casual ))
```

```
## Don't know how to automatically pick scale for object of type difftime. Defaulting to continuous.
```



In the graph of the average trip duration, the casual customer will use the service longer than the subscribe customer.

The day of ride

In this session, the usage statistic will be grouped by the day of the week. The total number of rides on each day of the week over 12 month period will be added. Then, it will be straightforward to see which day in the week is busier.

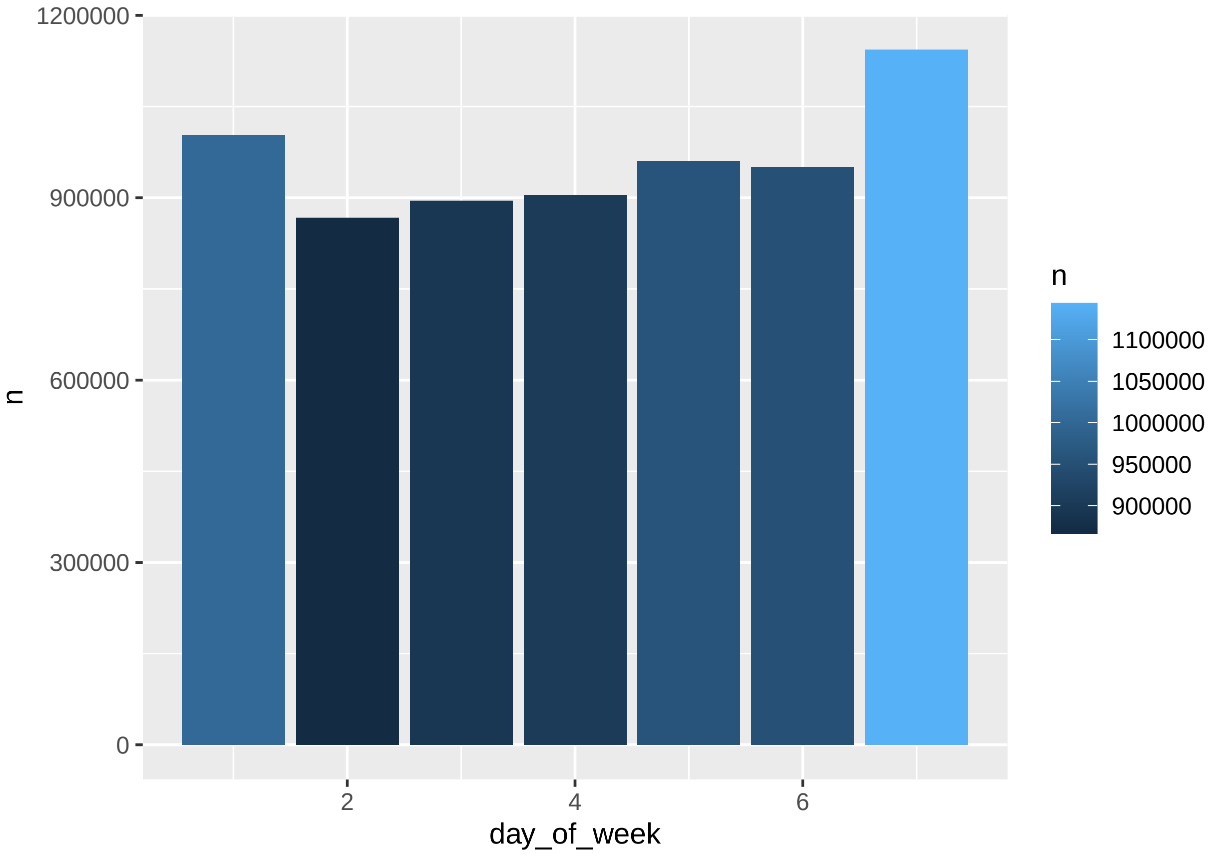
```
colnames(analysis_df)
```

```
## [1] "ride_id"            "rideable_type"      "start_station_name"
## [4] "end_station_name"   "started_at"         "member_casual"
## [7] "ride_length"        "day_of_week"        "time_of_day"
```

```
# To check which has the most of usage
Ride_weekly <- analysis_df%>%
  group_by( day_of_week)%>%
  select(ride_id, member_casual,day_of_week)%>%
  count()%>%
  ungroup()%>%
  mutate( perc = n/sum(n))%>%
  arrange(desc(perc))%>%
  mutate(percents = scales::percent(perc))
rename(Ride_weekly, trips = n)
```

```
## # A tibble: 7 x 4
##   day_of_week   trips  perc percents
##         <dbl>   <int> <dbl> <chr>
## 1           7 1143825 0.170 17.01%
## 2           1 1003094 0.149 14.92%
## 3           5  960055 0.143 14.28%
## 4           6  950510 0.141 14.14%
## 5           4  904523 0.135 13.45%
## 6           3  894999 0.133 13.31%
## 7           2  866867 0.129 12.89%
```

```
#print(Ride_weekly)
# Plotting the chart
ggplot(data = Ride_weekly) +
geom_col(mapping = aes(day_of_week,n,fill = n ))
```



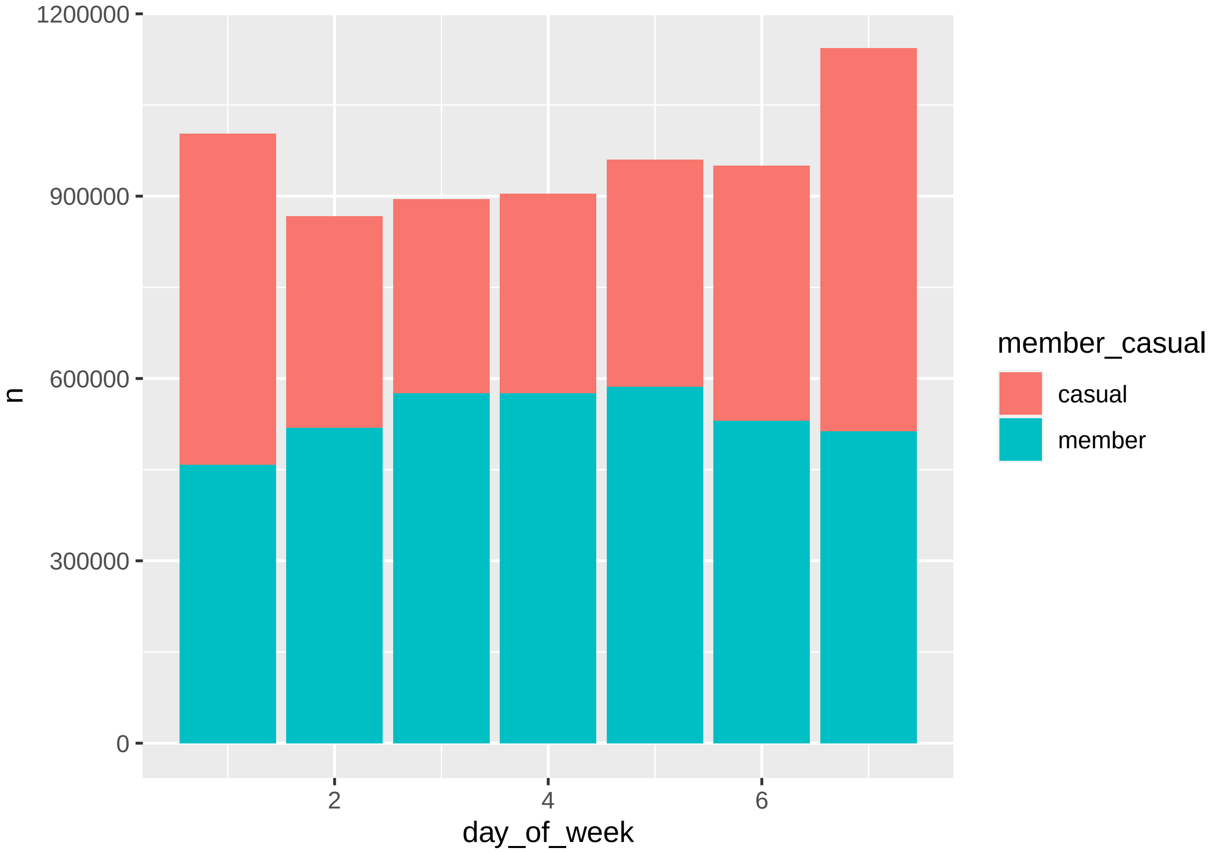

```r
# To check which has the most of usage by groups
Ride_weekly_by_group <- analysis_df%>%
  group_by(member_casual, day_of_week)%>%
  select(ride_id, member_casual,day_of_week)%>%
  count()%>%
  ungroup()%>%
  mutate( perc = n/sum(n))%>%
  arrange(desc(perc))%>%
  mutate(percents = scales::percent(perc))
rename(Ride_weekly_by_group, trips = n)
```

```
## # A tibble: 14 x 5
##    member_casual day_of_week  trips   perc percents
##    <chr>               <dbl>  <int>  <dbl> <chr>
##  1 casual                  7 630802 0.0938 9.3815%
##  2 member                  5 587091 0.0873 8.7314%
##  3 member                  4 576188 0.0857 8.5693%
##  4 member                  3 575770 0.0856 8.5631%
##  5 casual                  1 545370 0.0811 8.1110%
##  6 member                  6 530900 0.0790 7.8957%
##  7 member                  2 518895 0.0772 7.7172%
##  8 member                  7 513023 0.0763 7.6299%
##  9 member                  1 457724 0.0681 6.8074%
## 10 casual                  6 419610 0.0624 6.2406%
## 11 casual                  5 372964 0.0555 5.5469%
## 12 casual                  2 347972 0.0518 5.1752%
## 13 casual                  4 328335 0.0488 4.8831%
```

```
## 14 casual                    3 319229 0.0475 4.7477%
```

```
#print(Ride_weekly_by_group)

# Plotting the chart
ggplot(data = Ride_weekly_by_group) +
geom_col(mapping = aes(day_of_week,n,fill = member_casual ))
```



The time of ride

In this session, the usage statistic will be grouped by the time period of the day. The total number of rides in each hour over 12 month period will be added. Then, it will be straightforward to see the peak hour for the service.

```
colnames(analysis_df)
```

```
## [1] "ride_id"             "rideable_type"      "start_station_name"
## [4] "end_station_name"    "started_at"         "member_casual"
## [7] "ride_length"         "day_of_week"        "time_of_day"
```

```
#created new dataframe to group the usage in each hour of the day
hourly_usage_df <- analysis_df%>%
  group_by(time_of_day,member_casual)%>%
  select(member_casual,time_of_day)%>%
  arrange(time_of_day)%>%
  count()

#plot the graph with coloumn
ggplot(data = hourly_usage_df) +
```

```
geom_col(mapping = aes(time_of_day,n,fill = member_casual ))
```