Cyclistic Bike-Share case study

Adeyemi

2024-05-08

How does a bike-share navigate speedy success?

Overall Task

Design marketing strategies aimed at converting casual riders into annual members

Industry Focus

Marketing

Problem statement

How do annual members and casual riders use Cyclistics bikes differently? Why would casual riders buy Cyclistics annual memberships? How can Cyclistics use digital media to influence casual riders to become members?

BuisnessTask/Problem assignment

How do annual members and Casual riders use Cyclistics bikes differently?

Data

Download the previous 12 months of Cyclistics trip data for 2021.link https://divvy-tripdata.s3.amazonaws.com/index.htm

Type of Data

Public Data

Data Provider

Motivate International Inc license https://www.divvybikes.com/data-license-agreement

Data Limitation

Due to data-privacy issues one will not be able to connect past purchases to credit card numbers to determine if casual riders live in the Cyclistics service area or if they have purchased multiple single passes.

Step 1: Setting up my enviroment

Note: Setting up my environment involves loading the packages, 'tidyverse', 'skimr' and 'bike trip data'.

```
library(tidyverse)
```

```
## Warning: package 'tidyverse' was built under R version 4.3.3
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v readr     2.1.5
## v forcats   1.0.0     v stringr   1.5.1
## v ggplot2   3.4.4     v tibble    3.2.1
## v lubridate 1.9.3     v tidyr     1.3.1
## v purrr     1.0.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```
library(dplyr)
library(lubridate)
library(skimr)
```

```
## Warning: package 'skimr' was built under R version 4.3.3
```

```
library(ggplot2)
library(readr)
```

Use the conflicted package to manage conflicts

```
library(conflicted)
```

Set dplyr::filter and dplyr::lag as the default choice

```
conflict_prefer("filter", "dplyr")
```

```
## [conflicted] Will prefer dplyr::filter over any other package.
```

```
conflict_prefer("lag", "dplyr")
```

```
## [conflicted] Will prefer dplyr::lag over any other package.
```

Step 2: Collect data for the previous 12 months and combine in one data frame

Upload Divvy datasets (csv files) for 2021

```
q1_202101<-read_csv("~/Case Study1-Bike-share speedy Succes/Data/Raw Data/cyclistic data download file/
```

```
## 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.
```

```
q1_202102<-read_csv("~/Case Study1-Bike-share speedy Succes/Data/Raw Data/cyclistic data download file/
```

```
## 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.
```

```
q1_202103<-read_csv("~/Case Study1-Bike-share speedy Succes/Data/Raw Data/cyclistic data download file/
```

```
## 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.
```

```
q2_202104<-read_csv("~/Case Study1-Bike-share speedy Succes/Data/Raw Data/cyclistic data download file/
```

```
## Rows: 337230 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.
```

```
q2_202105<-read_csv("~/Case Study1-Bike-share speedy Succes/Data/Raw Data/cyclistic data download file/
```

```
## Rows: 531633 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.
```

```
q2_202106<-read_csv("~/Case Study1-Bike-share speedy Succes/Data/Raw Data/cyclistic data download file/
```

```
## Rows: 729595 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.
```

```
q3_202107<-read_csv("~/Case Study1-Bike-share speedy Succes/Data/Raw Data/cyclistic data download file/
```

```
## Rows: 822410 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.
```

```
q3_202108<-read_csv("~/Case Study1-Bike-share speedy Succes/Data/Raw Data/cyclistic data download file/
```

```
## Rows: 804352 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.
```

```
q3_202109<-read_csv("~/Case Study1-Bike-share speedy Succes/Data/Raw Data/cyclistic data download file/
## Rows: 756147 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.
```

```
q4_202110<-read_csv("~/Case Study1-Bike-share speedy Succes/Data/Raw Data/cyclistic data download file/2
```

```
## Rows: 631226 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.
```

```
q4_202111<-read_csv("~/Case Study1-Bike-share speedy Succes/Data/Raw Data/cyclistic data download file/2
```

```
## Rows: 359978 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.
```

```
q4_202112<-read_csv("~/Case Study1-Bike-share speedy Succes/Data/Raw Data/cyclistic data download file/2
```

```
## Rows: 247540 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.
```

Compare column names in each of the files.

While the names don't need to be in the same order, they do need to match perfectly before we can command to join them into one file

```
# colnames(q1_202101)
# colnames(q1_202102)
# colnames(q1_202103)
# colnames(q2_202104)
# colnames(q2_202105)
# colnames(q2_202106)
# colnames(q3_202107)
# colnames(q3_202108)
# colnames(q3_202109)
# colnames(q4_202110)
# colnames(q4_202111)
# colnames(q4_202112)
```

Inspect the dataframes and look for incongruencies

```
# str(q1_202101)
# str(q1_202102)
# str(q1_202103)
# str(q2_202104)
# str(q2_202105)
# str(q2_202106)
# str(q3_202107)
# str(q3_202108)
# str(q3_202109)
# str(q4_202110)
# str(q4_202111)
# str(q4_202112)
```

Combine the data from January 2021 to December 2021 into one data frame

```
bike_data<-bind_rows(q1_202101,q1_202102,q1_202103,q2_202104,q2_202105,q2_202106,q3_202107,q3_202108,q3_
```

Step 3: Clean Up and Add Data (new columns) to prepare for Analysis

Inspect the new table that has been created

```
# colnames(bike_data)        # List of column names
nrow(bike_data)             # How many rows are in data frame
```

```
## [1] 5595063
```

```
dim(bike_data)              # Dimensions of the data frame
```

```
## [1] 5595063      13
```

```r
head(bike_data)            # To see the first 6 rows of the data frame
```

```
## # A tibble: 6 x 13
##   ride_id          rideable_type started_at          ended_at
##   <chr>            <chr>         <dttm>              <dttm>
## 1 E19E6F1B8D4C42ED electric_bike 2021-01-23 16:14:19 2021-01-23 16:24:44
## 2 DC88F20C2C55F27F electric_bike 2021-01-27 18:43:08 2021-01-27 18:47:12
## 3 EC45C94683FE3F27 electric_bike 2021-01-21 22:35:54 2021-01-21 22:37:14
## 4 4FA453A75AE377DB electric_bike 2021-01-07 13:31:13 2021-01-07 13:42:55
## 5 BE5E8EB4E7263A0B electric_bike 2021-01-23 02:24:02 2021-01-23 02:24:45
## 6 5D8969F88C773979 electric_bike 2021-01-09 14:24:07 2021-01-09 15:17:54
## # i 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>
```

```r
# str(bike_data)            # See the list of columns and data types (numeric, character, dttm, dbl, e
# summary(bike_data)        # Statistical summary of data, mainly for numeric
```

There are a few observations and problems we will need to fix

(1) In the "member_casual" column, there are two names for members, "member" and "casual"- which is OK

(2) The data can only be aggregated/group_by at the rider level which it's to small to help our analysis.

N.B.:- "Level" is a special property of a column that is retained even if a subset does not contain any value from a specific level

We will want to add some additional columns of data – such as day, month, year– that provides additional opportunities to aggregate/group_by the data.

(3) We will want to add a calculated field for length of ride since from 2020 data did not have the "trip_duration" column. We will add "ride_length" to the entire dataframe for consistency.

Step 4: Check for NA values

```r
sum(is.na(bike_data))
```

```
## [1] 2869497
```

We choose not to remove NA because most NA are related to the start and end station name and their percentage of the overall observation is low.

Step 5: Drop columns we don't need for analysis: start_lat, start_lng, end_lat, end_lng

```
bike_data<- bike_data %>%  select(-c(start_lat, start_lng, end_lat, end_lng))
# colnames(bike_data)
```

Step 6a: Add column “ride_length”, which is the duration of each ride from ended_at minus started_at, and format as time, HH:MM:SS

Step 6b: Add column “day_of_week”, and calculate the day of the week that each ride started

```
bike_data<- bike_data %>%  mutate(ride_length=ended_at - started_at) %>% mutate(day_of_week = weekdays(
# glimpse(bike_data)
```

ride_length comes up as “drtn” (character on dataframe) and we need to convert it to numeric to do calculations.

Convert ride_length from difftime object to numeric then from seconds to minutes

```
bike_data$ride_length<- as.numeric(bike_data$ride_length)
bike_data$ride_length<- as.numeric(bike_data$ride_length / 60)
# head(bike_data)
```

Create date, month, day, and year column

```
bike_data$date<-as.Date(bike_data$started_at)   # default format is yyyy-mm-dd
bike_data$month<-format(as.Date(bike_data$date), "%m")
bike_data$day<-format(as.Date(bike_data$date), "%d")
bike_data$year<-format(as.Date(bike_data$date), "%y")
```

Check columns created

```
# colnames(bike_data)
```

Step 7: Cleaning - Remove the bad data and do analysis on the ride length

The data frame includes a few hundred entries when bikes were taken out of docks and checked for quality by Divvy or ride_length as negative

check for data with negative ride_length and remove

We will create a new version of the dataframe (v2) since data is being removed.

```r
bike_data_v2 <- bike_data[ bike_data$ride_length > 0,]
nrow(bike_data_v2)             # How many rows are in data frame
```

```
## [1] 5594410
```

Insight from bike_data_v2 data table after cleaning

There are 5,594,410 rides (rows) in 2021

check for data with ride length more than 1 day (86400 seconds or 1440 mins)

```r
sum(bike_data_v2$ride_length > 1440)
```

```
## [1] 4016
```

Insight from bike_data_v2 data table after cleaning

The number of rides with ride duration more than 1 day (86400 seconds or 1440 mins) is 4016
This is 0.72% of the total rides (5,594,410), low percentage

Check for extreme outliner

```r
max(bike_data_v2$ride_length)
```

```
## [1] 55944.15
```

```r
min(bike_data_v2$ride_length)
```

```
## [1] 0.01666667
```

Insight from bike_data_v2 data table after cleaning

We have 55944.15 minutes or 38.85 days for max ride duration and 1 second (0.01666667 min.) for least ride duration

Check for mean and median

```r
mean(bike_data_v2$ride_length)
```

```
## [1] 21.93831
```

```r
median(bike_data_v2$ride_length)
```

```
## [1] 12
```

Insight from bike_data_v2 data table after cleaning

The average duration of bike rides is 21.93831 min.
The median ride duration is 12 min.

Using summary to check min,max,median and mean

```r
# summary(bike_data_v2$ride_length)
```

Insight from bike_data_v2 data table after cleaning

min-0.02, 1st Qtr-6.75, Median-21.94, 3rd Qtr-21.78, max-55944.15
Minimum duration is 1 second (0.01666667 min.)
Maximum duration is 55944.15 minutes or 38.85 days

Step 8: Aggregrate/group_by to analyse the data based on user types: member vs casual

```r
aggregate(bike_data_v2$ride_length ~ bike_data_v2$member_casual, FUN = mean)
```

```
##   bike_data_v2$member_casual bike_data_v2$ride_length
## 1                     casual                 32.00578
## 2                     member                 13.63455
```

```r
aggregate(bike_data_v2$ride_length ~ bike_data_v2$member_casual, FUN = median)
```

```
##   bike_data_v2$member_casual bike_data_v2$ride_length
## 1                     casual                 15.98333
## 2                     member                  9.60000
```

```r
average_median_duration_userType<-bike_data_v2 %>% group_by(member_casual) %>% summarise(number_of_rides
# head(average_median_duration_userType)
```

Insight bike_data_v2 data table after cleaning-Statistical analysis

The number of rides casuals- 2,528,664: member- 3,065,746 The total number of rides - 5,594,410
Percentages of rides:members- 55% Percentage of rides: casual - 45%
Average ride duration- casual- 32 min.:members-13.6 min. Median ride duration- casual- 16 min.:members-9.6 min.

See the average ride_time by each day of the week for members vs casual users

```
aggregate(bike_data_v2$ride_length ~ bike_data_v2$member_casual + bike_data_v2$day_of_week, FUN =  mean
```

```
##    bike_data_v2$member_casual bike_data_v2$day_of_week bike_data_v2$ride_length
## 1                      casual                   Friday                 30.35177
## 2                      member                   Friday                 13.32608
## 3                      casual                   Monday                 31.87912
## 4                      member                   Monday                 13.24826
## 5                      casual                 Saturday                 34.70997
## 6                      member                 Saturday                 15.26552
## 7                      casual                   Sunday                 37.57096
## 8                      member                   Sunday                 15.65923
## 9                      casual                 Thursday                 27.70578
## 10                     member                 Thursday                 12.77703
## 11                     casual                  Tuesday                 27.97549
## 12                     member                  Tuesday                 12.78908
## 13                     casual                Wednesday                 27.66107
## 14                     member                Wednesday                 12.82021
```

Since the days of the week are out of order, let's order it

```
bike_data_v2$day_of_week<-ordered(bike_data_v2$day_of_week, levels=c("Sunday", "Monday", "Tuesday", "We
```

Now the average ride time by each day of the week for member_casual

```
aggregate(bike_data_v2$ride_length ~ bike_data_v2$member_casual + bike_data_v2$day_of_week, FUN =  mean
```

```
##    bike_data_v2$member_casual bike_data_v2$day_of_week bike_data_v2$ride_length
## 1                      casual                   Sunday                 37.57096
## 2                      member                   Sunday                 15.65923
## 3                      casual                   Monday                 31.87912
## 4                      member                   Monday                 13.24826
## 5                      casual                  Tuesday                 27.97549
## 6                      member                  Tuesday                 12.78908
## 7                      casual                Wednesday                 27.66107
## 8                      member                Wednesday                 12.82021
## 9                      casual                 Thursday                 27.70578
## 10                     member                 Thursday                 12.77703
## 11                     casual                   Friday                 30.35177
## 12                     member                   Friday                 13.32608
## 13                     casual                 Saturday                 34.70997
## 14                     member                 Saturday                 15.26552
```

Average ride duration (using group_by) by each day of the week for members vs casual

```
bike_data_v2  %>% group_by(member_casual, day_of_week) %>% summarise(average_ride_length = mean(ride_len
```

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

```
## # A tibble: 14 x 3
## # Groups:   member_casual [2]
##    member_casual day_of_week average_ride_length
##    <chr>         <ord>                     <dbl>
##  1 casual        Sunday                     37.6
##  2 casual        Monday                     31.9
##  3 casual        Tuesday                    28.0
##  4 casual        Wednesday                  27.7
##  5 casual        Thursday                   27.7
##  6 casual        Friday                     30.4
##  7 casual        Saturday                   34.7
##  8 member        Sunday                     15.7
##  9 member        Monday                     13.2
## 10 member        Tuesday                    12.8
## 11 member        Wednesday                  12.8
## 12 member        Thursday                   12.8
## 13 member        Friday                     13.3
## 14 member        Saturday                   15.3
```

dataframe for the analysis of average ride duration by each day of the week for members vs casual

```
# average_duration_userType_weekday<-bike_data_v2  %>% group_by(member_casual, day_of_week) %>% summari
```

Step 9: Further analysis into the stations which shows where NA belong to in step 4.

```
head(count(bike_data_v2, start_station_name, member_casual, rideable_type, sort = TRUE))
```

```
## # A tibble: 6 x 4
##   start_station_name      member_casual rideable_type      n
##   <chr>                   <chr>         <chr>          <int>
## 1 <NA>                    member        electric_bike 373106
## 2 <NA>                    casual        electric_bike 317678
## 3 Streeter Dr & Grand Ave casual        classic_bike   37478
## 4 Clark St & Elm St       member        classic_bike   19058
## 5 Streeter Dr & Grand Ave casual        docked_bike    18139
## 6 Millennium Park         casual        classic_bike   17490
```

```r
head(count(bike_data_v2,end_station_name, member_casual, rideable_type, sort = TRUE))
```

```
## # A tibble: 6 x 4
##   end_station_name        member_casual rideable_type      n
##   <chr>                   <chr>         <chr>          <int>
## 1 <NA>                    member        electric_bike 370008
## 2 <NA>                    casual        electric_bike 359514
## 3 Streeter Dr & Grand Ave casual        classic_bike   37919
## 4 Streeter Dr & Grand Ave casual        docked_bike    20150
## 5 Clark St & Elm St       member        classic_bike   19168
## 6 Michigan Ave & Oak St   casual        classic_bike   18275
```

Insight on NA

The NA are occurring at start_station_name (12% of rides)and end_station_name (13% of rides) columns

Step 10: Analyse ridership data by type and weekday- create weekday column using wday()

```r
bike_data_v2 %>% mutate(weekday= wday(started_at, label=TRUE)) %>% group_by(member_casual, weekday) %>%
  summarise(number_of_ride = n(), average_duration= mean(ride_length)) %>% arrange(member_casual,weekda
```

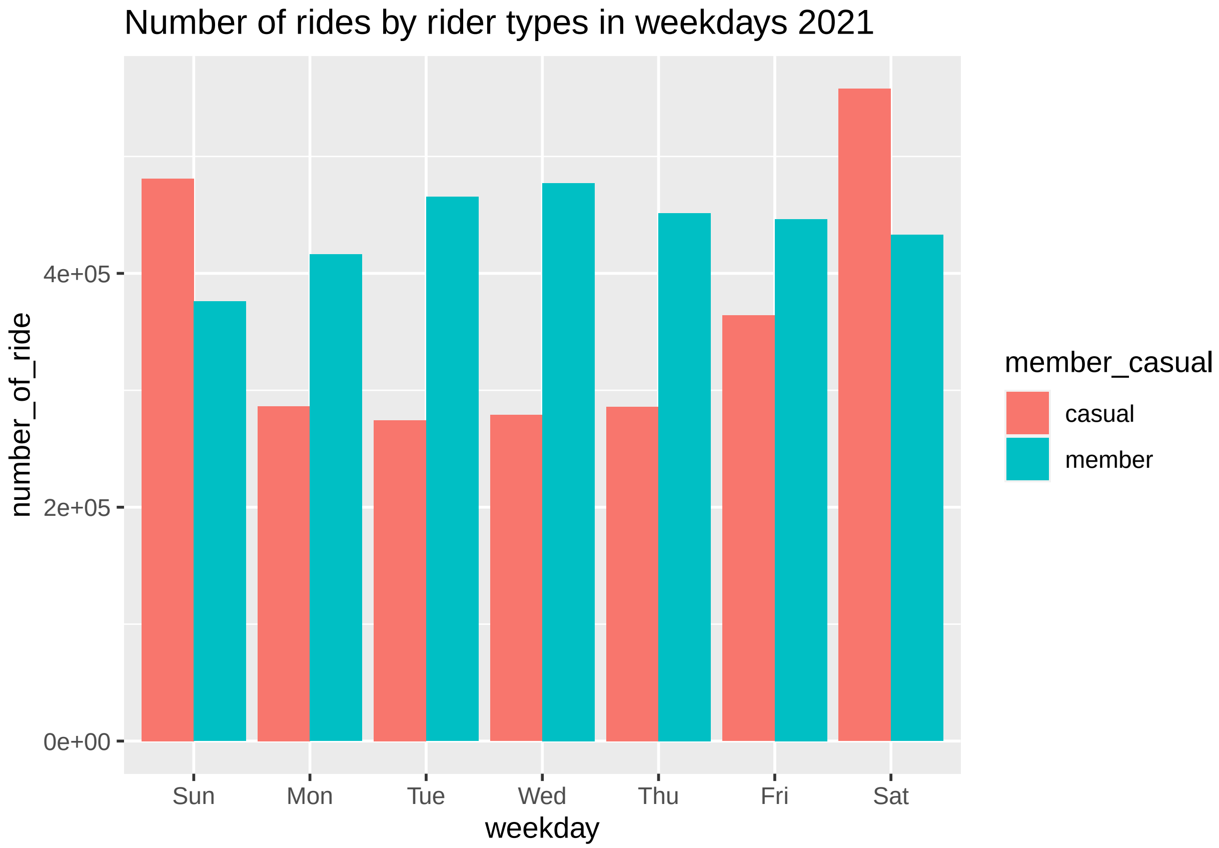
```
## `summarise()` has grouped output by 'member_casual'. You can override using the
## `.groups` argument.
```

```
## # A tibble: 14 x 4
## # Groups:   member_casual [2]
##    member_casual weekday number_of_ride average_duration
##    <chr>         <ord>            <int>            <dbl>
##  1 casual        Sun             481048             37.6
##  2 casual        Mon             286340             31.9
##  3 casual        Tue             274357             28.0
##  4 casual        Wed             278910             27.7
##  5 casual        Thu             286038             27.7
##  6 casual        Fri             364037             30.4
##  7 casual        Sat             557934             34.7
##  8 member        Sun             376086             15.7
##  9 member        Mon             416181             13.2
## 10 member        Tue             465474             12.8
## 11 member        Wed             477117             12.8
## 12 member        Thu             451490             12.8
## 13 member        Fri             446384             13.3
## 14 member        Sat             433014             15.3
```

Step 11: Visualize the number of rides by rider types AND average duration on weekdays

```
bike_data_v2 %>% mutate(weekday= wday(started_at, label= TRUE)) %>% group_by(member_casual, weekday) %>
  summarise(number_of_ride = n(), average__duration = mean(ride_length)) %>% arrange(member_casual, wee
  ggplot(mapping = aes(x= weekday, y= number_of_ride, fill= member_casual)) +
  geom_col(position = "dodge") + labs(title = "Number of rides by rider types in weekdays 2021")
```

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

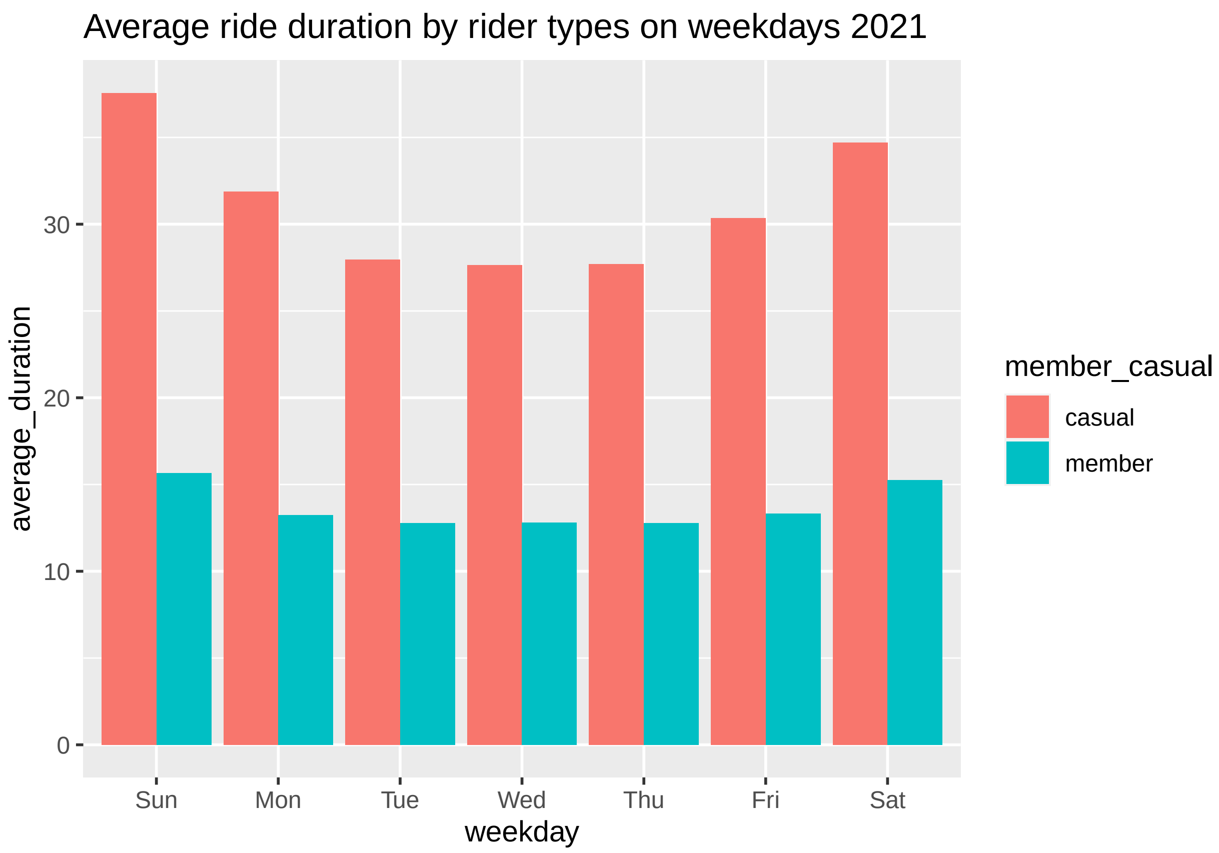


Insight from "Number of rides by rider types in weekdays 2021"

Members have more number of bike rides from Monday to Friday than casuals
Casuals members have more number of rides on Saturday and Sunday (weekend)

```
bike_data_v2 %>% mutate(weekday= wday(started_at, label= TRUE)) %>% group_by(member_casual, weekday) %>
  summarise(number_of_ride= n(), average_duration= mean(ride_length)) %>%
  arrange(member_casual, weekday) %>% ggplot(mapping = aes(x= weekday,y = average_duration, fill= membe
  geom_col(position = "dodge") + labs(title = "Average ride duration by rider types on weekdays 2021")
```

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

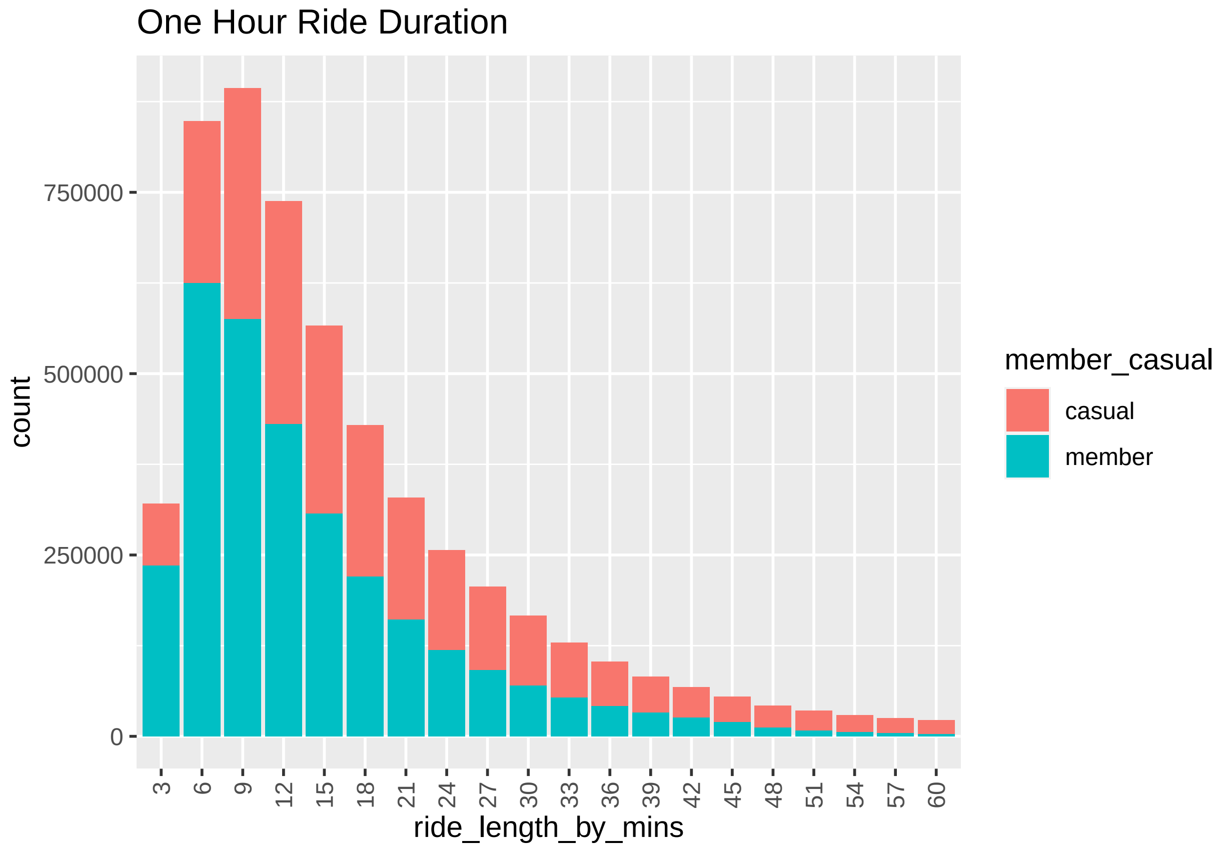


Insight from "Average ride duration by rider types on weekdays"

Casuals have longer rides than members through out the weekdays
They both stay longer on their rides during weekends

Step 12: Visualize member vs casual on short rides (less than one hour)

```
one_hour_data<- bike_data_v2 %>% filter(ride_length < 60)
one_hour_data$ride_length_by_mins<- cut(one_hour_data$ride_length, breaks = 20)
ggplot(one_hour_data) +
  geom_bar(mapping = aes(x= ride_length_by_mins, fill= member_casual)) +
  labs(title = "One Hour Ride Duration") +
  theme(axis.text.x = element_text(angle = 90, vjust = 0.5, hjust = 1)) +
  scale_x_discrete(labels= c("3","6","9","12","15","18","21","24","27","30","33","36","39","42","45","4
```

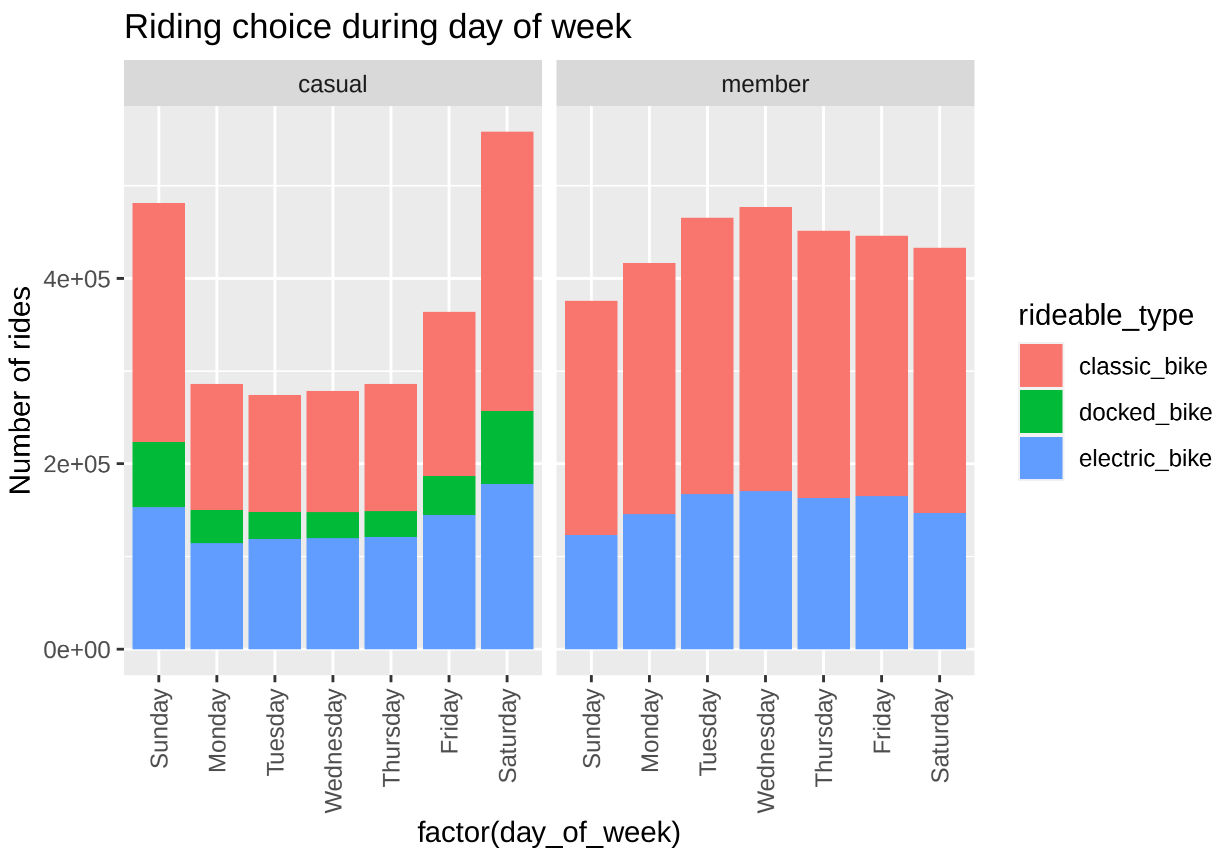


Insight from "One Hour Ride Duration"

The ride duration of 9 minutes is the duration that most rides less than one hour cover

Step 13: Visualize day of the week ride choices between member vs casual

```r
ggplot(data = bike_data_v2) +
  geom_bar(mapping = aes(x= factor(day_of_week), fill= rideable_type)) +
  facet_wrap(~ member_casual) +
  labs(title = 'Riding choice during day of week', y="Number of rides") +
  theme(axis.text.x = element_text(angle = 90, vjust = 0.5, hjust=1))
```



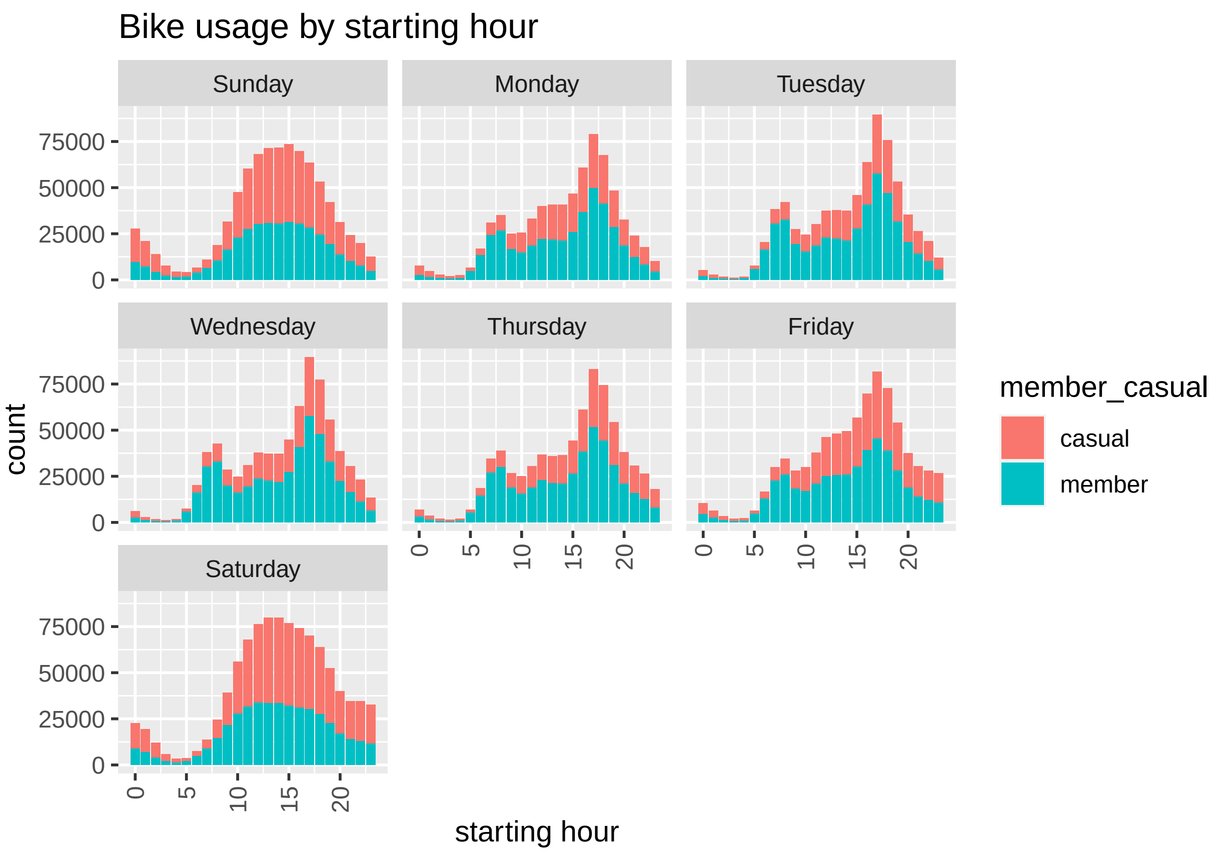
Insight from "Riding choice during day of week"

The ride used by casual members are classic, docked, and electric bikes
Members used classic and electric bikes
Casual have maximum rides on Saturday and minimum on Tuesday The members have maximum rides on Wednesday and and minimum on Sunday

Step 14: Check for peak time for bike usage between member vs casual

```r
hour_data <- bike_data_v2
hour_data$start_hour <- as.numeric(format(strptime(bike_data_v2$started_at,"%Y-%m-%d %H:%M:%S"),'%H'))
ggplot(data = hour_data) +
  geom_bar(mapping = aes(x = start_hour, fill = member_casual), stat = 'count') +
  facet_wrap(~factor(day_of_week)) +
  labs(title = "Bike usage by starting hour", x = "starting hour") +
  theme(axis.text.x = element_text(angle = 90, vjust = 0.5, hjust=1))
```

```
## Warning: Removed 29 rows containing non-finite values (‘stat_count()‘).
```



Insight from "Bike usage by starting hour"

We have the peak bike usage at the following hours

Sunday - 15hrs
Saturday - 13/14hrs
Monday to Friday - 17hrs

Step 15: Save as csv for further analysis and visualization in Tableau

bike_ride_v2 dataframe

```
write_csv(bike_data_v2, "bikedata_v2.csv")
```

total and average weekly rides by rider type

```
summary_ride_weekly<- bike_data_v2 %>%
  mutate(weekday= wday(started_at, label= TRUE)) %>%
  group_by(member_casual, weekday) %>%
  summarise(number_of_rides = n(), average_duration= mean(ride_length)) %>%
  arrange(member_casual, weekday)
```

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

```
write_csv(summary_ride_weekly, "summary_ride_weekly.csv")
```

total and avg monthly rides by rider type

```
summary_month<- bike_data_v2 %>%
  mutate(month = month(started_at, label = TRUE)) %>%
  group_by(month, member_casual) %>%
  summarise(number_of_rides = n(), average_duration = mean(ride_length)) %>%
  arrange(month, member_casual)
```

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

```
write_csv(summary_month, "summary_ride_monthly.csv")
```

most popular stations

```
popular_stations<- bike_data_v2 %>%
  mutate(station = start_station_name) %>%
  drop_na(start_station_name) %>%
  group_by(start_station_name, member_casual) %>%
  summarise(number_of_rides = n()) %>%
  arrange(-number_of_rides)
```

```
## ‘summarise()‘ has grouped output by ’start_station_name’. You can override
## using the ‘.groups‘ argument.
```

```
head(popular_stations)
```

```
## # A tibble: 6 x 3
## # Groups:   start_station_name [6]
##   start_station_name       member_casual number_of_rides
##   <chr>                    <chr>                   <int>
## 1 Streeter Dr & Grand Ave  casual                  66353
## 2 Millennium Park          casual                  33578
## 3 Michigan Ave & Oak St    casual                  29778
## 4 Clark St & Elm St        member                  24739
## 5 Wells St & Concord Ln    member                  23716
## 6 Kingsbury St & Kinzie St member                  23562
```

```
write_csv(popular_stations, "popular_stations.csv")
```

Insight from “popular_stations” dataframe

The most popular station is Streeter Dr & Grand Ave

total membership types and rideable types

```
total_riders <- data.frame(table(bike_data$member_casual))
total_types <- data.frame(table(bike_data$rideable_type))
write_csv(total_riders, "total_riders.csv")
write_csv(total_types, "total_types.csv")
count_rides<- bike_data %>%
  group_by(member_casual, rideable_type) %>%
  summarise(number_of_rides= n(), average_duration= mean(ride_length)) %>%
  arrange(member_casual)
```

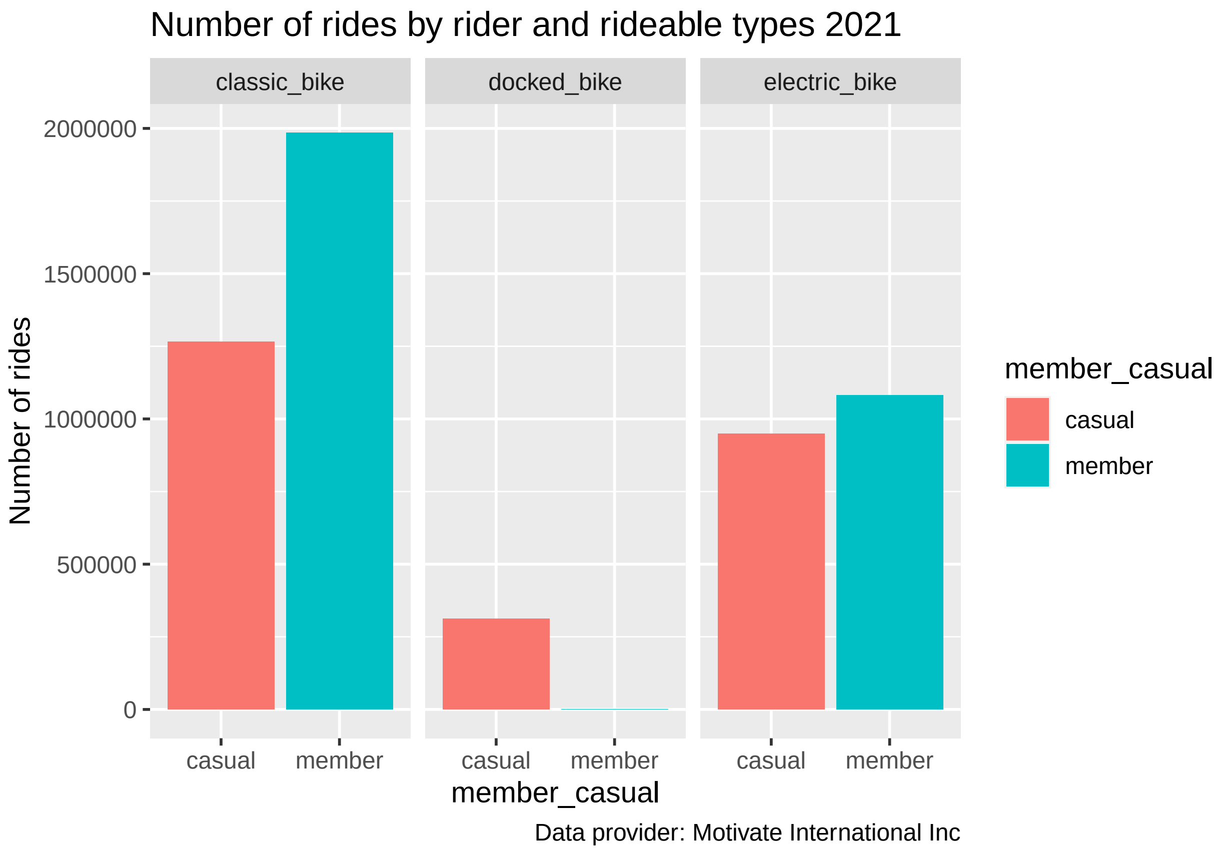
```
## ‘summarise()‘ has grouped output by ’member_casual’. You can override using the
## ‘.groups‘ argument.
```

```
write_csv(count_rides, "count_rides.csv")
```

Step 16 Visulization to get trends and findings

Plot the graph of number of rides by member vs casual and rideable types in 2021

```
ggplot(data = bike_data) + geom_bar(mapping = aes(x = member_casual, fill= member_casual)) + labs(title
```

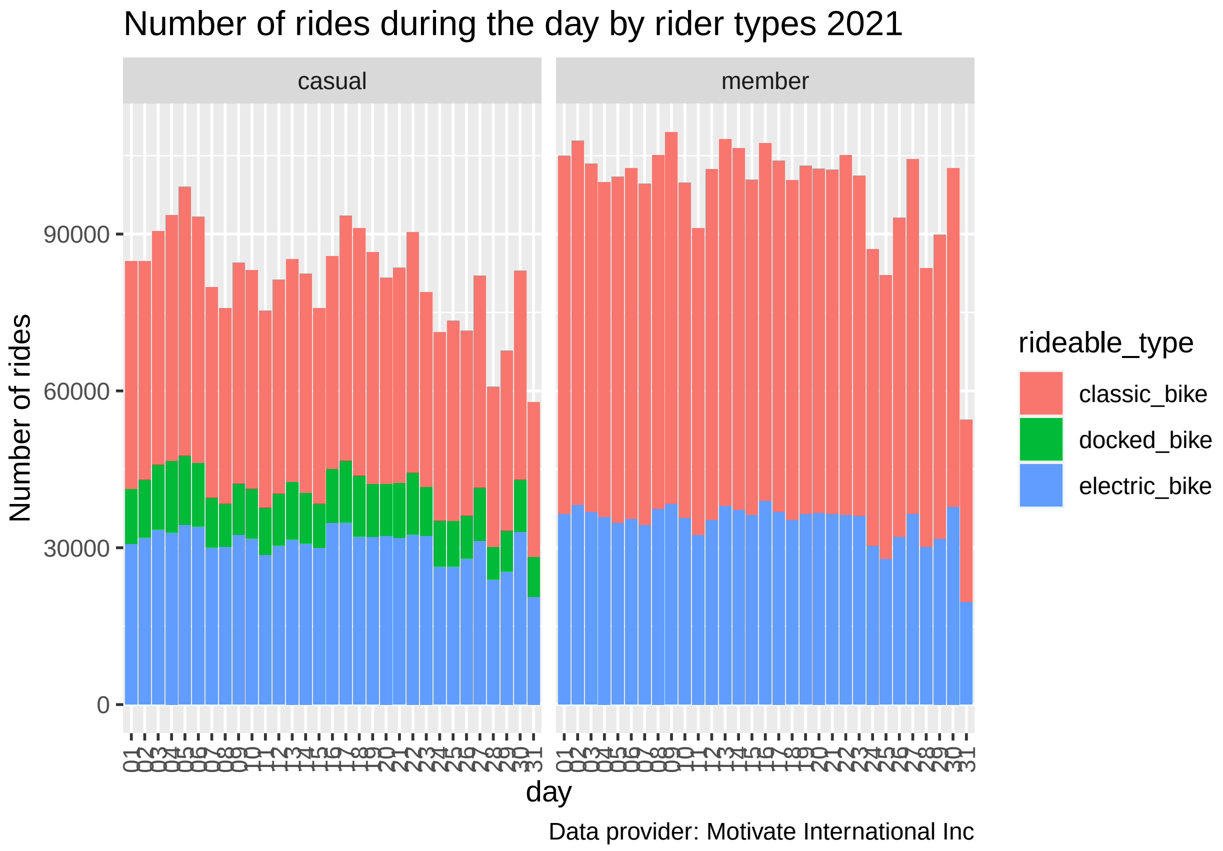


Findings from "Number of rides by rider and rideable types 2021"

The casual members rides classic, electric and docked bikes in decreasing order of usage
The members ride classic and electric bike in decreasing order of usage
Members have more number of bike rides than casuals
Members have no ride on docked bikes

Plot the graph of number of rides during the day 2021

```
ggplot(data = bike_data) + geom_bar(mapping = aes(x = day, fill= rideable_type), stat = 'count') + labs
```



Number of rides during the day by rider types 2021

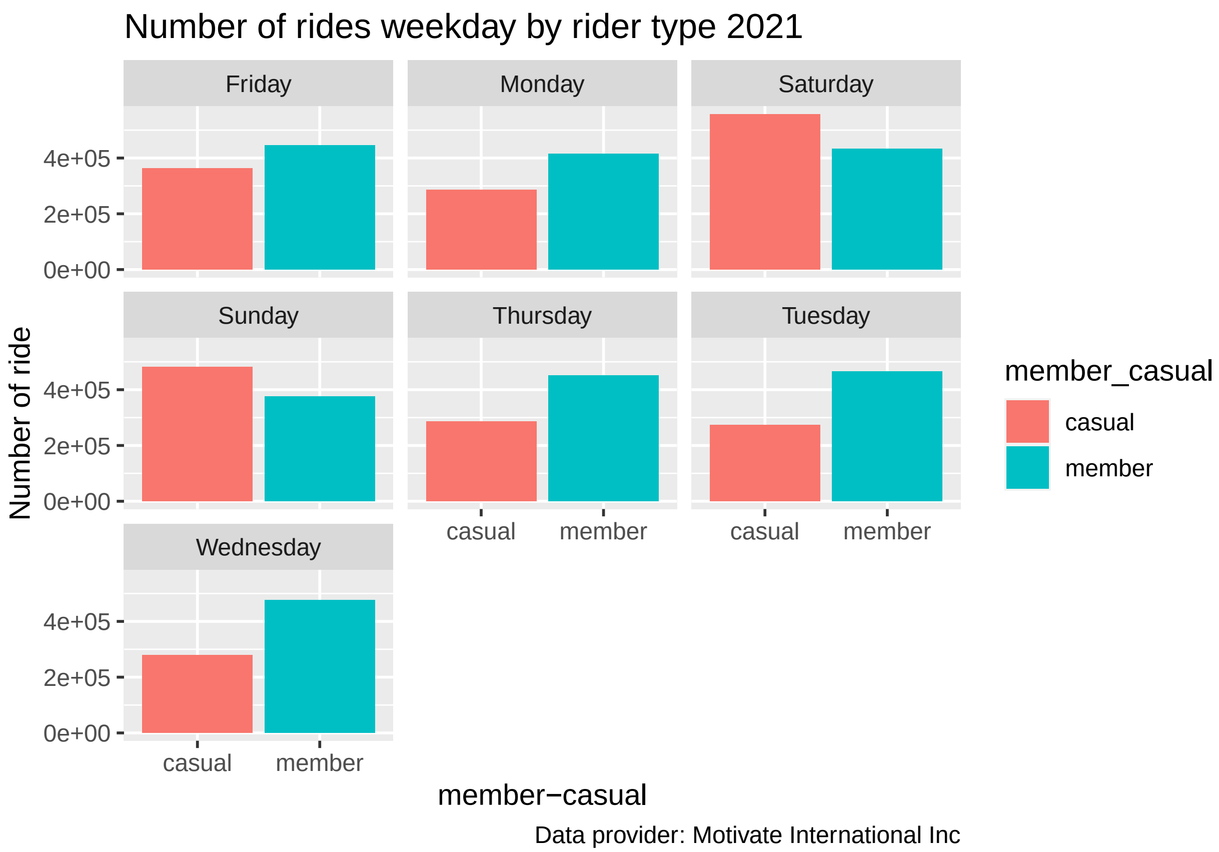
Data provider: Motivate International Inc

Findings from "Number of rides during the day by rider types 2021"

Casual riders maximum number of rides is on the 5th and minimum on 31st of the month
Members have the highest number of rides on the 9th and minimum on the 31st

Plot number of rides during the weekday

```
ggplot(data = bike_data) + geom_bar(mapping = aes(x = member_casual, fill= member_casual)) + facet_wrap
```



Number of rides weekday by rider type 2021

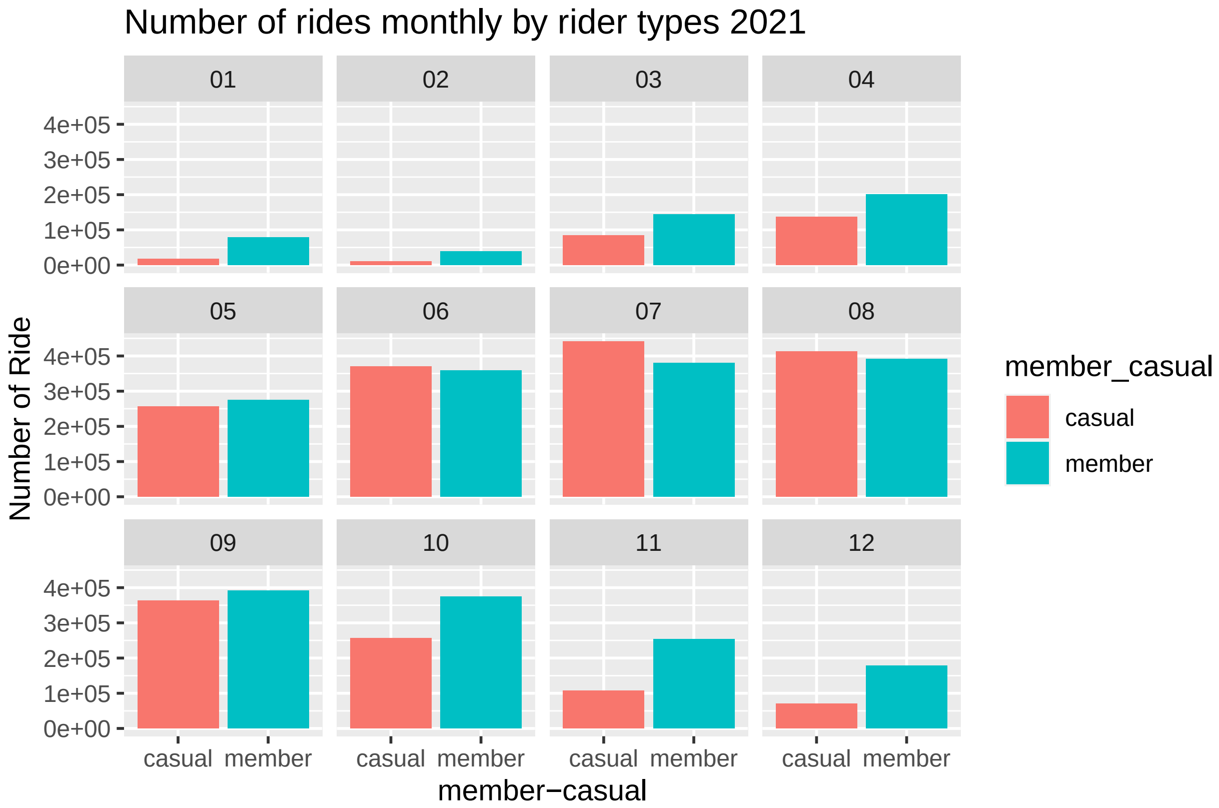
Data provider: Motivate International Inc

Findings from "Number of rides weekday by rider type 2021" ***The casual members have the maximum number of rides on Saturday then Sunday***

The casual have their minimum number of rides on Wednesday /Thursday
The members have the maximum number of rides on Wednesday followed by Tuesday
The members minimum number of rides are on Sunday
There are more member riders out riding from Mondays to Fridays
The casuals come out more than members to ride bikes on Saturdays and Sundays

Plot the number of rides monthly by rider types 2021

```
ggplot(data = bike_data) + geom_bar(mapping = aes(x = member_casual, fill= member_casual)) + facet_wrap
```



Data provider: Motivate International Inc

Findings from "Number of rides monthly by rider types 2021"

Casual riders maximum number of rides is in July and minimum in February
Members maximum number of rides is in September and minimum in February

Top Three Recommendations based on the Analysis

(1)The marketing campaign aimed at converting casual riders into annual members should commence from March when we have an up swing in the number of casual riders to October.

(2)The intensity of the campaign should be greatest on Saturday and Sunday when we have most casual riders coming out for bike rides.

(3)The campaign should be from 5hrs to 20hrs during which most of the casual riders would be out for their bike ride.