Data Analyst Capstone

Atakan KÖREZ

05 11 2021

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

The purpose of this document is to consolidate downloaded Divvy data into a single dataframe and then conduct simple analysis to help answer the key question: "In what ways do members and casual riders use Divvy bikes differently?"

Introduction

This exploratory analysis case study is towards Capstome project requirement for Google Data Analytics Professional Certificate. The case study involves a bikeshare company's data of its customer's trip details over a 12 month period (November 2020 - October 2021). The data has been made available by Motivate International Inc. under this license.

The analysis will follow the 6 phases of the Data Analysis process: Ask, Prepare, Process, Analyze, and Act. A brief explanation of these processes:

Ask

- Ask effective questions
- Define the scope of the analysis
- Define what success looks like

Prepare

- Verify data's integrity
- Check data credibility and reliability
- Check data types
- Merge datasets

Process

- Clean, Remove and Transform data
- Document cleaning processes and results

Analyze

- Identify patterns
- Draw conclusions
- Make predictions

Share

- Create effective visuals
- Create a story for data
- Share insights to stakeholders

Act

- Give recommendations based on insights
- Solve problems
- Create something new

1. Ask

Scenario Marketing team needs to design marketing strategies aimed at converting casual riders into annual members. In order to do that, however, the marketing analyst team needs to better understand how annual members and casual riders differ.

Stakeholders:

- Director of marketing
- Cyclistic executive team

Objective Hence, the objective for this analysis is to throw some light on how the two types of customers: annual members and casual riders, use Cyclistic bikeshare differently, based on few parameters that can be calculated/ obtained from existing data.

Deliverables:

- Insights on how annual members and casual riders use Cyclistic bikes differently
- Provide effective visuals and relevant data to support insights
- Use insights to give three recommendations to convert casual riders to member riders

2. Prepare

Data Source A total of **12 CSV files** have been made available for each month starting from **November 2020 to October 2021**. Each file captures the details of every ride logged by the customers of Cyclistic. This data that has been made publicly available has been scrubbed to omit rider's personal information.

The combined size of all the 12 CSV files is close to 950 MB. Data cleaning in spreadsheets will be time-consuming and slow compared to R. I am choosing R simply because I could do both data wrangling and analysis/ visualizations in the same platform.

```
library(tidyverse)
```

Load Libraries

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --

## v ggplot2 3.3.5     v purrr   0.3.4
## v tibble  3.1.5     v dplyr   1.0.7
## v tidyr   1.1.4     v stringr 1.4.0
## v readr   2.0.2     v forcats 0.5.1

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

```
library(ggplot2)
library(lubridate)
```

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

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

```
library(dplyr)
library(readr)
library(janitor)
```

```
##
## Attaching package: 'janitor'

## The following objects are masked from 'package:stats':
##
##     chisq.test, fisher.test
```

```
library(data.table)
```

```
##
## Attaching package: 'data.table'

## The following objects are masked from 'package:lubridate':
##
##     hour, isoweek, mday, minute, month, quarter, second, wday, week,
##     yday, year

## The following objects are masked from 'package:dplyr':
##
##     between, first, last

## The following object is masked from 'package:purrr':
##
##     transpose
```

```
library(tidyr)
```

```

tripdata_202011 <- read.csv("~/Data_Analytics_Capstone/202011-divvy-tripdata.csv")
tripdata_202012 <- read.csv("~/Data_Analytics_Capstone/202012-divvy-tripdata.csv")
tripdata_202101 <- read.csv("~/Data_Analytics_Capstone/202101-divvy-tripdata.csv")
tripdata_202102 <- read.csv("~/Data_Analytics_Capstone/202102-divvy-tripdata.csv")
tripdata_202103 <- read.csv("~/Data_Analytics_Capstone/202103-divvy-tripdata.csv")
tripdata_202104 <- read.csv("~/Data_Analytics_Capstone/202104-divvy-tripdata.csv")
tripdata_202105 <- read.csv("~/Data_Analytics_Capstone/202105-divvy-tripdata.csv")
tripdata_202106 <- read.csv("~/Data_Analytics_Capstone/202106-divvy-tripdata.csv")
tripdata_202107 <- read.csv("~/Data_Analytics_Capstone/202107-divvy-tripdata.csv")
tripdata_202108 <- read.csv("~/Data_Analytics_Capstone/202108-divvy-tripdata.csv")
tripdata_202109 <- read.csv("~/Data_Analytics_Capstone/202109-divvy-tripdata.csv")
tripdata_202110 <- read.csv("~/Data_Analytics_Capstone/202110-divvy-tripdata.csv")
```

Load dataset CSV files (Previous 12 months of Cyclistic trip data)

Data transformation and cleaning start_station_id & end_station_id are not consistent in one CSV file. The ones in tripdata_202011 is int vs. the others are char. Convert the inconsistent ones from int to char datatype.

```
tripdata_202011 <- tripdata_202011 %>% mutate(start_station_id = as.character(start_station_id), end_sta
```

3. Process

Combine all the datasets into one single dataframe

```
all_trips <- bind_rows(tripdata_202011,tripdata_202012,tripdata_202101,tripdata_202102,tripdata_202103,t

str(all_trips)
```

```
## 'data.frame':    5378834 obs. of  13 variables:
##  $ ride_id           : chr  "BD0A6FF6FFF9B921" "96A7A7A4BDE4F82D" "C61526D06582BDC5" "E533E89C32080B9
##  $ rideable_type     : chr  "electric_bike" "electric_bike" "electric_bike" "electric_bike" ...
##  $ started_at        : chr  "2020-11-01 13:36:00" "2020-11-01 10:03:26" "2020-11-01 00:34:05" "2020-1
##  $ ended_at          : chr  "2020-11-01 13:45:40" "2020-11-01 10:14:45" "2020-11-01 01:03:06" "2020-1
##  $ start_station_name: chr  "Dearborn St & Erie St" "Franklin St & Illinois St" "Lake Shore Dr & Mon
##  $ start_station_id  : chr  "110" "672" "76" "659" ...
##  $ end_station_name  : chr  "St. Clair St & Erie St" "Noble St & Milwaukee Ave" "Federal St & Polk St
##  $ end_station_id    : chr  "211" "29" "41" "185" ...
##  $ start_lat         : num  41.9 41.9 41.9 41.9 41.9 ...
##  $ start_lng         : num  -87.6 -87.6 -87.6 -87.7 -87.6 ...
##  $ end_lat           : num  41.9 41.9 41.9 41.9 41.9 ...
##  $ end_lng           : num  -87.6 -87.7 -87.6 -87.7 -87.6 ...
##  $ member_casual     : chr  "casual" "casual" "casual" "casual" ...
```

Clean-up further! Hold on! started_at & ended_at should be in datetime datatype instead of char. Convert all from char to datetime.

```
all_trips[['started_at']] <- ymd_hms(all_trips[['started_at']])
all_trips[['ended_at']] <- ymd_hms(all_trips[['ended_at']])
```

```
all_trips <- all_trips %>%
  select(-c(start_lat:end_lng))

glimpse(all_trips)
```

Remove columns not required or beyond the scope of project

```
## Rows: 5,378,834
## Columns: 9
## $ ride_id            <chr> "BD0A6FF6FFF9B921", "96A7A7A4BDE4F82D", "C61526D065~
## $ rideable_type      <chr> "electric_bike", "electric_bike", "electric_bike", ~
## $ started_at         <dttm> 2020-11-01 13:36:00, 2020-11-01 10:03:26, 2020-11-~
## $ ended_at           <dttm> 2020-11-01 13:45:40, 2020-11-01 10:14:45, 2020-11-~
## $ start_station_name <chr> "Dearborn St & Erie St", "Franklin St & Illinois St~
## $ start_station_id   <chr> "110", "672", "76", "659", "2", "72", "76", NA, "58~
## $ end_station_name   <chr> "St. Clair St & Erie St", "Noble St & Milwaukee Ave~
## $ end_station_id     <chr> "211", "29", "41", "185", "2", "76", "72", NA, "288~
## $ member_casual      <chr> "casual", "casual", "casual", "casual", "casual", "~
```

```
all_trips <- all_trips %>%
  rename(ride_type = rideable_type,
         start_time = started_at,
         end_time = ended_at,
         customer_type = member_casual)
```

```
glimpse(all_trips)
```

Rename columns for better readability

```
## Rows: 5,378,834
## Columns: 9
## $ ride_id            <chr> "BD0A6FF6FFF9B921", "96A7A7A4BDE4F82D", "C61526D065~
## $ ride_type          <chr> "electric_bike", "electric_bike", "electric_bike", ~
## $ start_time         <dttm> 2020-11-01 13:36:00, 2020-11-01 10:03:26, 2020-11--~
## $ end_time           <dttm> 2020-11-01 13:45:40, 2020-11-01 10:14:45, 2020-11--~
## $ start_station_name <chr> "Dearborn St & Erie St", "Franklin St & Illinois St~
## $ start_station_id   <chr> "110", "672", "76", "659", "2", "72", "76", NA, "58~
## $ end_station_name   <chr> "St. Clair St & Erie St", "Noble St & Milwaukee Ave~
## $ end_station_id     <chr> "211", "29", "41", "185", "2", "76", "72", NA, "288~
## $ customer_type      <chr> "casual", "casual", "casual", "casual", "casual", "~
```

```
# column for day of the week the trip started
all_trips$day_of_the_week <- format(as.Date(all_trips$start_time),'%a')
```

```
# column for month when the trip started
all_trips$month <- format(as.Date(all_trips$start_time),'%b_%y')
```

```
# The time is then converted back to POSIXct with today's date - the date is of no interest to us,only
all_trips$time <- format(all_trips$start_time, format = "%H:%M")
all_trips$time <- as.POSIXct(all_trips$time, format = "%H:%M")
```

```
# column for trip duration in min
all_trips$trip_duration <- (as.double(difftime(all_trips$end_time, all_trips$start_time)))/60
```

```
# check the dataframe
glimpse(all_trips)
```

```
## Rows: 5,378,834
## Columns: 13
## $ ride_id            <chr> "BD0A6FF6FFF9B921", "96A7A7A4BDE4F82D", "C61526D065~
## $ ride_type          <chr> "electric_bike", "electric_bike", "electric_bike", ~
## $ start_time         <dttm> 2020-11-01 13:36:00, 2020-11-01 10:03:26, 2020-11--~
## $ end_time           <dttm> 2020-11-01 13:45:40, 2020-11-01 10:14:45, 2020-11--~
## $ start_station_name <chr> "Dearborn St & Erie St", "Franklin St & Illinois St~
## $ start_station_id   <chr> "110", "672", "76", "659", "2", "72", "76", NA, "58~
## $ end_station_name   <chr> "St. Clair St & Erie St", "Noble St & Milwaukee Ave~
## $ end_station_id     <chr> "211", "29", "41", "185", "2", "76", "72", NA, "288~
## $ customer_type      <chr> "casual", "casual", "casual", "casual", "casual", "~
## $ day_of_the_week    <chr> "Paz", "Paz", "Paz", "Paz", "Paz", "Cmt", "Cmt", "C~
## $ month              <chr> "Kas_20", "Kas_20", "Kas_20", "Kas_20", "Kas_20", "~
## $ time               <dttm> 2021-11-06 13:36:00, 2021-11-06 10:03:00, 2021-11--~
## $ trip_duration      <dbl> 9.6666667, 11.3166667, 29.0166667, 9.2500000, 33.45~
```

Let's check to see if the trip_duration column has any negative values, as this may cause problem while creating visualizations. Also, we do not want to include the trips that were part of quality tests by the company. These trips are usually identified by string 'test' in the start_station_name column.

```
# checking for trip lengths less than 0
nrow(subset(all_trips,trip_duration < 0))
```

```
## [1] 1393
```

```r
#checking for testrides that were made by company for quality checks
nrow(subset(all_trips, start_station_name %like% "TEST"))
```

```
## [1] 105
```

```r
nrow(subset(all_trips, start_station_name %like% "test"))
```

```
## [1] 0
```

```r
nrow(subset(all_trips, start_station_name %like% "Test"))
```

```
## [1] 0
```

As there are 1393 rows with trip_dration less than 0 mins and 105 trips that were test rides, we will remove these observations from our dataframe as they contribute to only about 0.3% of the total rows. We will create a new dataframe deviod of these obseravtions without making any changes to the existing dataframe.

```r
# remove negative trip durations
all_trips_v2 <- all_trips[!(all_trips$trip_duration < 0),]

#remove test rides
all_trips_v2<- all_trips_v2[!((all_trips_v2$start_station_name %like% "TEST" | all_trips_v2$start_statio

#check dataframe
glimpse(all_trips_v2)
```

```
## Rows: 5,377,336
## Columns: 13
## $ ride_id            <chr> "BD0A6FF6FFF9B921", "96A7A7A4BDE4F82D", "C61526D065~
## $ ride_type          <chr> "electric_bike", "electric_bike", "electric_bike", ~
## $ start_time         <dttm> 2020-11-01 13:36:00, 2020-11-01 10:03:26, 2020-11-~
## $ end_time           <dttm> 2020-11-01 13:45:40, 2020-11-01 10:14:45, 2020-11-~
## $ start_station_name <chr> "Dearborn St & Erie St", "Franklin St & Illinois St~
## $ start_station_id   <chr> "110", "672", "76", "659", "2", "72", "76", NA, "58~
## $ end_station_name   <chr> "St. Clair St & Erie St", "Noble St & Milwaukee Ave~
## $ end_station_id     <chr> "211", "29", "41", "185", "2", "76", "72", NA, "288~
## $ customer_type      <chr> "casual", "casual", "casual", "casual", "casual", "~
## $ day_of_the_week    <chr> "Paz", "Paz", "Paz", "Paz", "Paz", "Cmt", "Cmt", "C~
## $ month              <chr> "Kas_20", "Kas_20", "Kas_20", "Kas_20", "Kas_20", "~
## $ time               <dttm> 2021-11-06 13:36:00, 2021-11-06 10:03:00, 2021-11-~
## $ trip_duration      <dbl> 9.6666667, 11.3166667, 29.0166667, 9.2500000, 33.45~
```

It is important to make sure that customer_type column has only two distinct values. Let's confirm the same.

```r
# checking count of distinct values
table(all_trips_v2$customer_type)
```

```
##
##  casual  member
## 2470117 2907219
```

```r
#aggregating total trip duration by customer type
setNames(aggregate(trip_duration ~ customer_type, all_trips_v2, sum), c("customer_type", "total_trip_du
```

```
##   customer_type total_trip_duration(mins)
## 1        casual                  80423286
## 2        member                  40594097
```

4&5. Analyze and Share Data

The dataframe is now ready for descriptive analysis that will help us uncover some insights on how the casual riders and members use Cyclistic rideshare differently.

First, let's try to get some simple statistics on trip_duration for all customers, and do the same by customer_type.

```
# staticitical summary of trip_duration for all trips
summary(all_trips_v2$trip_duration)
```

```
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
##     0.00     6.97    12.38    22.51    22.43 55944.15
```

```
#statistical summary of trip_duration by customer_type
all_trips_v2 %>%
  group_by(customer_type) %>%
  summarise(min_trip_duration = min(trip_duration),max_trip_duration = max(trip_duration),
            median_trip_duration = median(trip_duration), mean_trip_duration = mean(trip_duration))
```

```
## # A tibble: 2 x 5
##   customer_type min_trip_duration max_trip_duration median_trip_duration
##   <chr>                     <dbl>             <dbl>                <dbl>
## 1 casual                        0            55944.                16.4
## 2 member                        0             1560.                 9.92
## # ... with 1 more variable: mean_trip_duration <dbl>
```

The mean trip duration of member riders is lower than the mean trip duration of all trips, while it is exactly the opposite for casual riders, whose mean trip duration is higher than the the mean trip duration of all trips. This tells us that casual riders usually take the bikes out for a longer duration compared to members.

```
# fix the order for the day_of_the_week and month variable so that they show up in the same sequence in
all_trips_v2$day_of_the_week <- ordered(all_trips_v2$day_of_the_week, levels=c("Pzt", "Sal", "Çar", "Pe

all_trips_v2$month <- ordered(all_trips_v2$month, levels=c("Kas_20","Ara_20","Oca_21","Şub_21","Mar_21"

#Total number of trips by customer type and day of the week

all_trips_v2 %>%
  group_by(customer_type, day_of_the_week) %>%
  summarise(number_of_rides = n(),average_duration_mins = mean(trip_duration)) %>%
  arrange(customer_type, desc(number_of_rides))
```

Total number of trips by customer type and day of the week

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

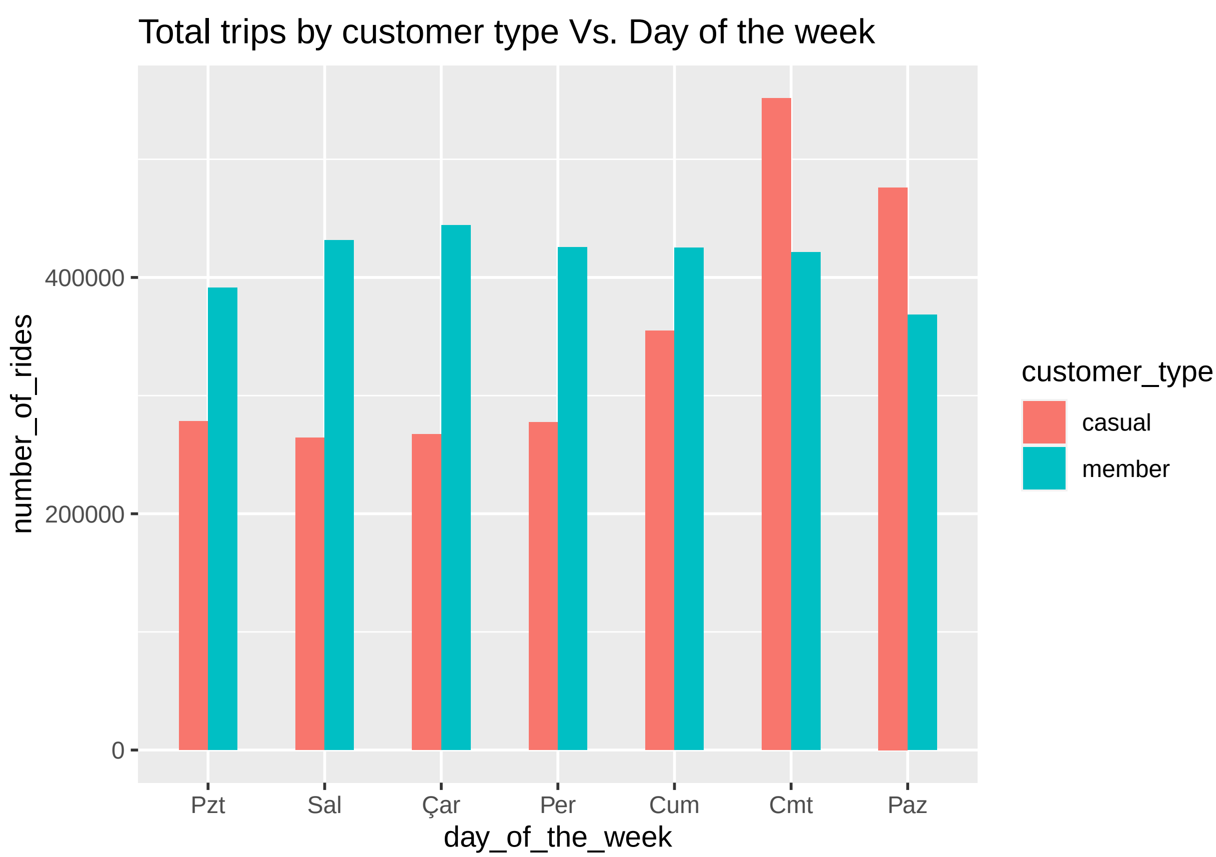
```
## # A tibble: 14 x 4
## # Groups:   customer_type [2]
##    customer_type day_of_the_week number_of_rides average_duration_mins
##    <chr>         <ord>                     <int>                 <dbl>
##  1 casual        Cmt                      551814                  35.2
##  2 casual        Paz                      476138                  38.1
##  3 casual        Cum                      354955                  30.9
##  4 casual        Pzt                      278245                  32.3
##  5 casual        Per                      277310                  28.2
##  6 casual        Çar                      267431                  28.2
```

```
## 7 casual      Sal              264224            28.6
## 8 member      Çar              444255            13.2
## 9 member      Sal              431459            13.1
## 10 member     Per              425562            13.1
## 11 member     Cum              425119            13.7
## 12 member     Cmt              421164            15.6
## 13 member     Pzt              391286            13.5
## 14 member     Paz              368374            15.9
```

```
#Total trips by customer type Vs. Day_of_Week
all_trips_v2 %>%
  group_by(customer_type, day_of_the_week) %>%
  summarise(number_of_rides = n()) %>%
  arrange(customer_type, day_of_the_week)  %>%
  ggplot(aes(x = day_of_the_week, y = number_of_rides, fill = customer_type)) +
  labs(title ="Total trips by customer type Vs. Day of the week") +
  geom_col(width=0.5, position = position_dodge(width=0.5)) +
  scale_y_continuous(labels = function(x) format(x, scientific = FALSE))
```

Visualization

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



From the table and graph above, casual customers are most busy on Sundays followed by Saturdays, while members are most busy on later half of the week extending into the weekend. Interesting pattern to note though is the consistent trip numbers among members with less spread over entire week as compared to casual riders who don't seem to use the bikeshare services much during weekdays.

```
all_trips_v2 %>%
  group_by(customer_type, month) %>%
  summarise(number_of_rides = n(),`average_duration_(mins)` = mean(trip_duration)) %>%
  arrange(customer_type,desc(number_of_rides))
```

Average number of trips by customer type and month

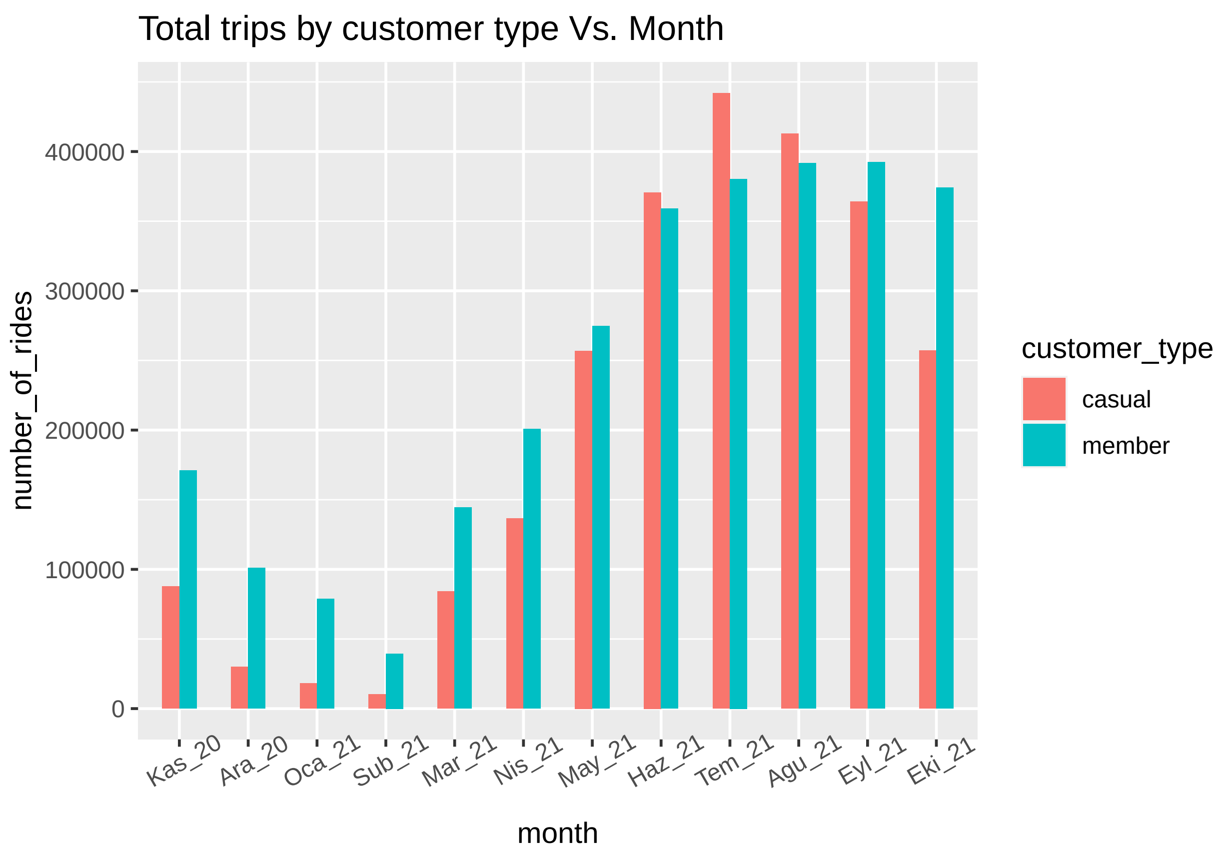
```
## `summarise()` has grouped output by 'customer_type'. You can override using the `.groups` argument.
```

```
## # A tibble: 24 x 4
## # Groups:   customer_type [2]
##    customer_type month  number_of_rides `average_duration_(mins)`
##    <chr>         <ord>            <int>                     <dbl>
##  1 casual        Tem_21          442048                      32.8
##  2 casual        Ağu_21          412662                      28.8
##  3 casual        Haz_21          370678                      37.1
##  4 casual        Eyl_21          363883                      27.8
##  5 casual        Eki_21          257242                      28.7
##  6 casual        May_21          256916                      38.2
##  7 casual        Nis_21          136601                      38.0
##  8 casual        Kas_20           87810                      31.9
##  9 casual        Mar_21           84032                      38.2
## 10 casual        Ara_20           29997                      26.8
## # ... with 14 more rows
```

```
#Total trips by customer type Vs. Month
all_trips_v2 %>%
  group_by(customer_type, month) %>%
  summarise(number_of_rides = n()) %>%
  arrange(customer_type, month)  %>%
  ggplot(aes(x = month, y = number_of_rides, fill = customer_type)) +
  labs(title ="Total trips by customer type Vs. Month") +
  theme(axis.text.x = element_text(angle = 30)) +
  geom_col(width=0.5, position = position_dodge(width=0.5)) +
  scale_y_continuous(labels = function(x) format(x, scientific = FALSE))
```

Visualization

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

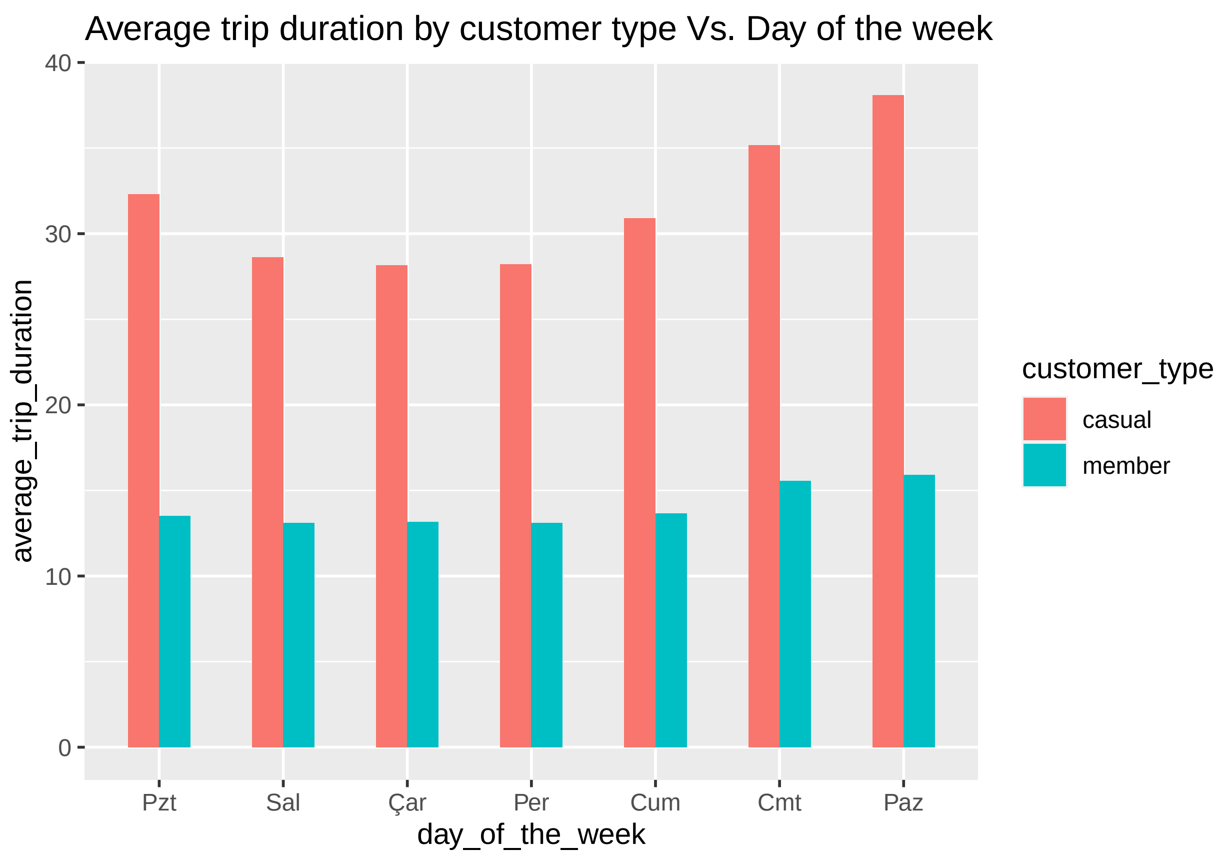


The data shows that the months of July, August, September and October are the most busy time of the year among both members and casual riders. This could be attributed to an external factor (eg. cold weather, major quality issue) that might have hindered with customer needs. 2021 is a tough year when Covid comes. People care more about their health. The charts shows that the no.of rides in 2021 is higher than 2020 in general. However, the number of trips made by members is always higher than the casual riders across all months of the year.

```
all_trips_v2 %>%
  group_by(customer_type, day_of_the_week) %>%
  summarise(average_trip_duration = mean(trip_duration)) %>%
  ggplot(aes(x = day_of_the_week, y = average_trip_duration, fill = customer_type)) +
  geom_col(width=0.5, position = position_dodge(width=0.5)) +
  labs(title ="Average trip duration by customer type Vs. Day of the week")
```

Visualizaton of average trip duration by customer type on each day of the week

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

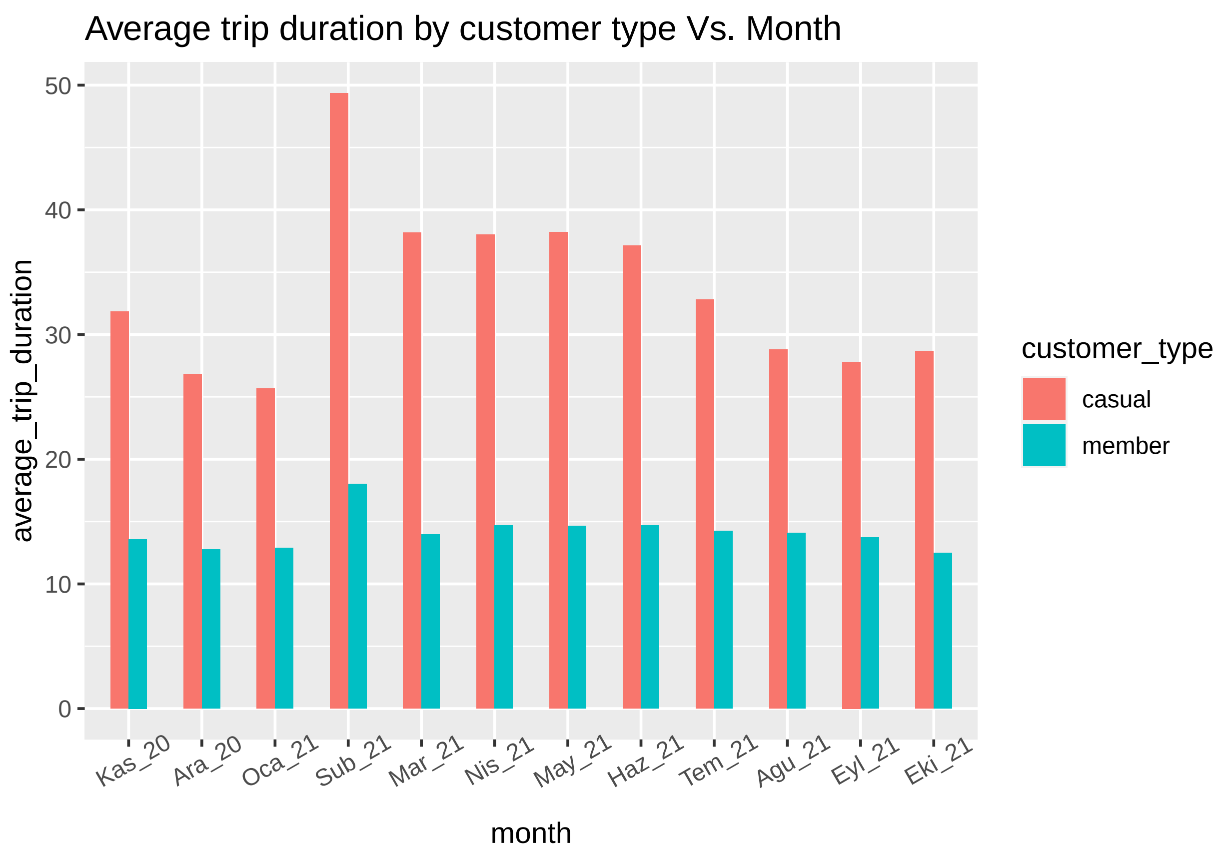


The average trip duration of a casual rider is more than twice that of a member. Note that this necessarily does not mean that casual riders travel farther distance. It is also interesting to note that weekends not only contribute to more number of trips but also longer trips on average when compared to weekdays.

```
all_trips_v2 %>%
  group_by(customer_type, month) %>%
  summarise(average_trip_duration = mean(trip_duration)) %>%
  ggplot(aes(x = month, y = average_trip_duration, fill = customer_type)) +
  geom_col(width=0.5, position = position_dodge(width=0.5)) +
  labs(title ="Average trip duration by customer type Vs. Month") +
  theme(axis.text.x = element_text(angle = 30))
```

Visualizaton of average trip duration by customer type Vs. month

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

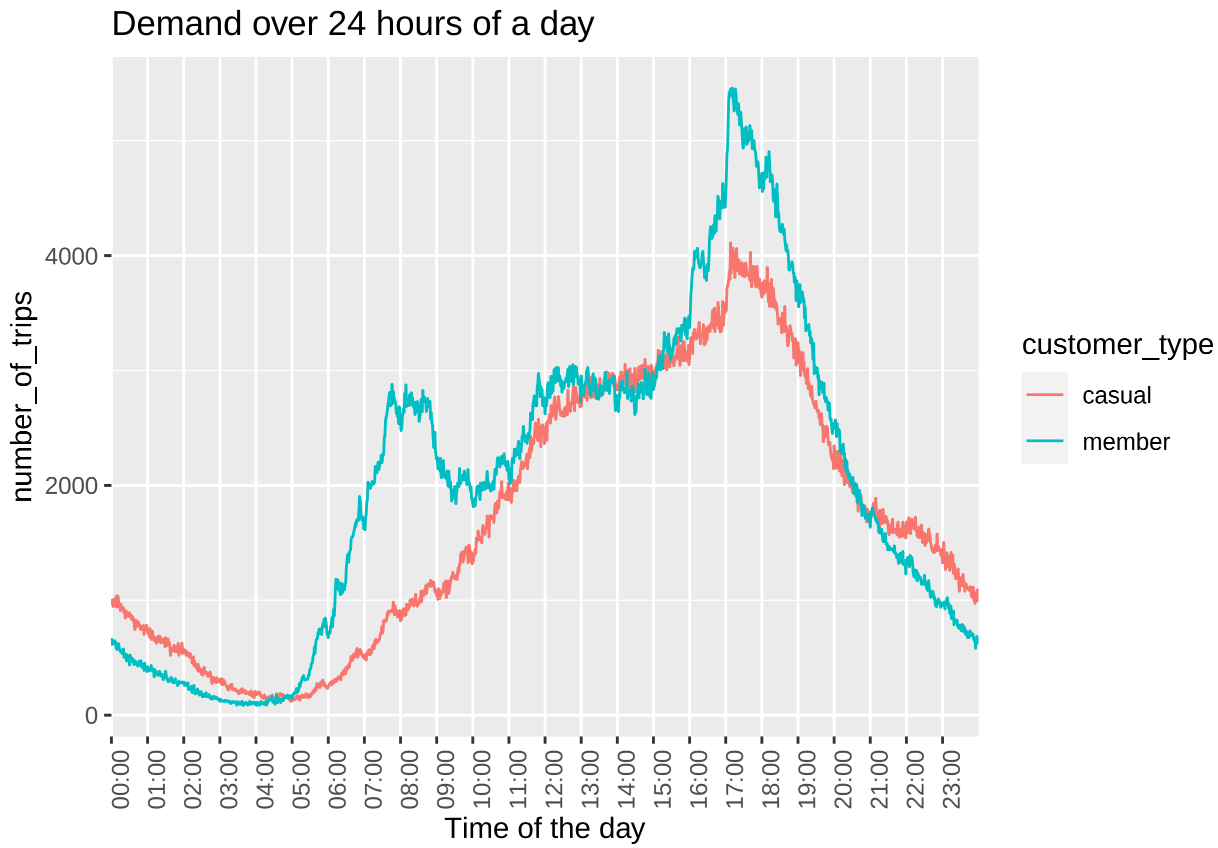


Average trip duration of member riders is anywhere between 10-30 minutes throughout the year, exception being February when it goes slightly over 20 minutes. However, there seems to be a distinct pattern when it comes to casual riders, whose average trip duration swings wildly from as low as ~25 minutes to more than an hour depending on time of the year. It is worth noting unusually long trip durations by casual riders in the month of February.

```
all_trips_v2 %>%
  group_by(customer_type, time) %>%
  summarise(number_of_trips = n()) %>%
  ggplot(aes(x = time, y = number_of_trips, color = customer_type, group = customer_type)) +
  geom_line() +
  scale_x_datetime(date_breaks = "1 hour", minor_breaks = NULL,
                   date_labels = "%H:%M", expand = c(0,0)) +
  theme(axis.text.x = element_text(angle = 90)) +
  labs(title ="Demand over 24 hours of a day", x = "Time of the day")
```

Visualizaton of bike demand over 24 hr period (a day)

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

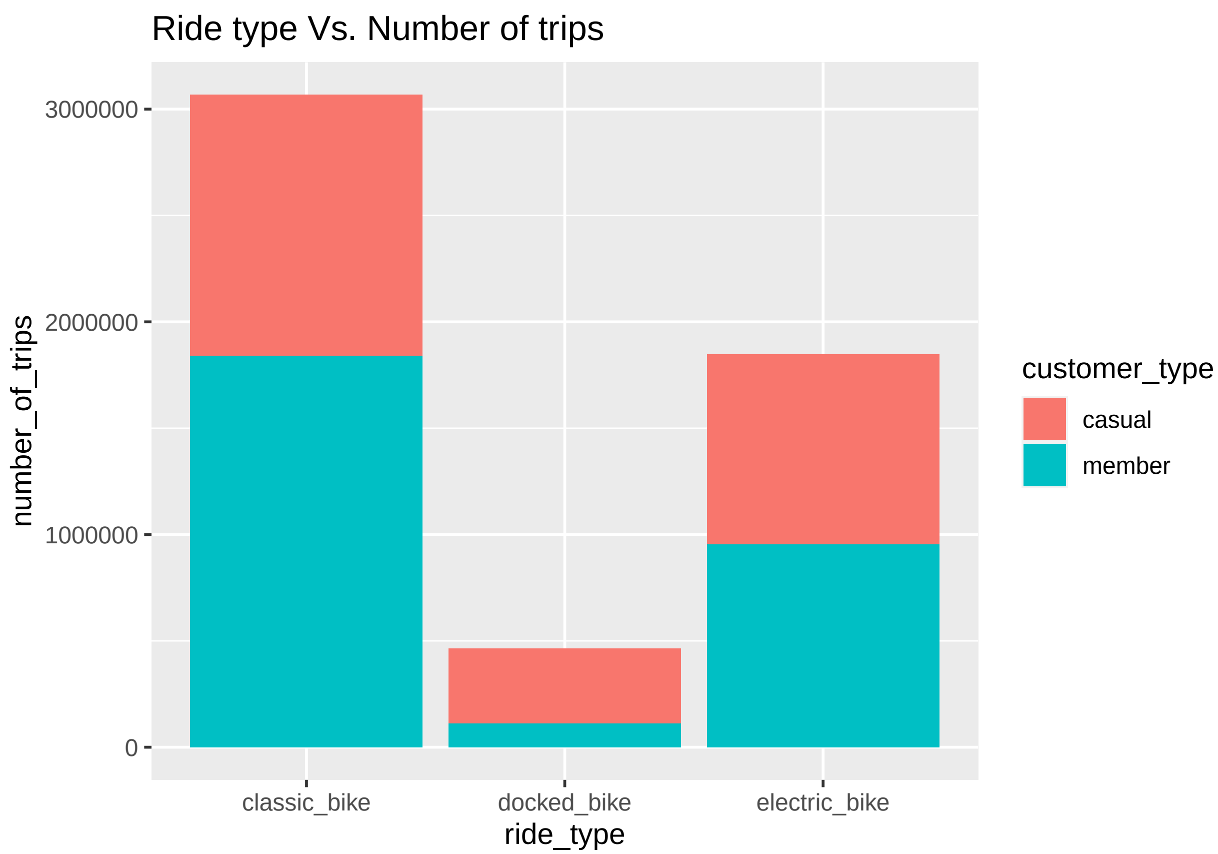


For the members, there seems to be two distict peak demand hours: 7-9 AM and 5-7 PM, the latter one coinciding with the peak demand hours of casual riders as well. One could probably hypothesize that office-goers make up majority of the members profile due to demand in both morning and evening hours, but we need more data to substabtiate this assumption.

```
all_trips_v2 %>%
  group_by(ride_type, customer_type) %>%
  summarise(number_of_trips = n()) %>%
  ggplot(aes(x= ride_type, y=number_of_trips, fill= customer_type))+
  geom_bar(stat='identity') +
  scale_y_continuous(labels = function(x) format(x, scientific = FALSE)) +
  labs(title ="Ride type Vs. Number of trips")
```

Visualizaton of ride type Vs. number of trips by customer type

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



Classic bikes are predominantly used by members. Docked bikes are almost never used compared to others. Electric bikes are equally used by both members as well as casual riders. If docked bikes cost the highest among all 3 types, it would be a financially sound move to increase their fleet while reducing docked bikes, as they are already preferred by members who make up for the majority of the trips.

6. Act

Important Findings

- Casual riders made 41% of total trips contributing to 66% of total trip duration between Nov'20 - Oct'21. Member riders make up 59% of total trips contributing to 34% of total trip duration between Nov'20 - Oct'21
- Usage (based on trip duration) of bikes by casual riders is almost twice that of member riders.
- Casual customers use bikeshare services more during weekends, while members use them consistently over the entire week.
- Average trip duration of casual riders is more than twice that of member rider over any given day of the week cumulatively.
- Casual riders ride longer during first half of the year compared to the second half, while members clock relatively similar average trip duration month over month.
- Casual riders prefer electric bikes the most while classic bikes are popular among members.

Recommendations

- Provide attractive promotions for casual riders on weekdays so that casual members use the bikeshare services ore uniformly across the entire week.

- Offer discounted membership fee for renewals after the first year. It might nudge casual riders to take up membership.
- Offer discounted pricing during non-busy hours so that casual riders might choose to use bikes more often and level out demand over the day.

Additonal data that could expand scope of analysis

- Age and gender profile - Again, this data could be used to study the category of riders who can be targeted for attracting new members.
- Address/ neighborhood details of members to investigate if there are any location specific parameters that encourage membership.
- Pricing details for members and casual riders - Based on this data, we might be to optimize cost structure for casual riders or provide discounts without affecting the profit margin.