Cyclistic bike-share analysis case study!

Ameyaw Ahmed Analysis

2022-07-15

Case Study: Does Cyclistic future successs depends on annual membership riders?

The main goal of this document is to consolidate the downloaded Divvy data into one data frame and then conduct a simple analysis which would help the marketing team to understand **how casual riders usage of Divvy bikes differ from annual membership riders** by finding insights that will help that could be turned into recommendation to help the marketing team design a new marketing strategy to convert casual riders into annual members.

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 **13 month period (Jul 2021 - Jul 2022)**. 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 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: The cyclistic marketing team needs to design strategies aimed atconverting casual riders into annual members. Howver the marketing team needs to under how the casual riders and annual members differ.

Stakeholders

- Director of marketing -Cyclistic executive team

Objective

The main goal is therefore to analysis the 13-month period data to extract some insight on how the two main customer sgegment use the Cyclistic bikes differently.

Deliverables

- Provide insights on how annual riders and casual riders use the Cyclistic bikes differently
- Support the findings with effective visualizations
- Based on the insights found, give recommendations on how to convert casual riders to member riders.

2. Prepare and Process

Source of data

The data has been made available by Motivate International Inc. under this license. The data is a monthly data which begins at July 11th, 2021 and ends at July 14th 2021. The data can be found here. The data consist of a total of **13 CSV files** starting from July 2021 to July 2022. Each of the files contain data of the monthly rides by customers of Cyclistic. The combined data is over 6 million observations hence doing data cleaning and wrangling in excel will be slow that's why I have chosen to conduct all the analysis process in R.

Load Libraries

```
library(tidyverse)
library(janitor)
library(here)
library(lubridate)
library(hms)
library(anytime)
library(scales)
library(zoo)
library(ggcharts)
library(showtext)
library(glue)
```

```
font_add_google(family = "josefin-new", "Josefin Sans")
showtext_auto()
```

Importing files

Comparaing datatype of the monthly data to further combine them

```
compare_df_cols(
  Apr_secyr,
  Dec_firstyr,
  Aug_firstyr,
  Feb_secyr,
  Jan_secyr,
  Jul_firstyr,
  Jul_secyr,
  Jun_secyr,
  Mar_secyr,
  May_secyr,
  Nov_firstyr,
  Oct_firstyr,
  Sep_firstyr
)
```

```
##            column_name Apr_secyr     Dec_firstyr     Aug_firstyr       Feb_secyr
```

```
## 1             end_lat   numeric         numeric         numeric         numeric
## 2             end_lng   numeric         numeric         numeric         numeric
## 3      end_station_id character       character       character       character
## 4    end_station_name character       character       character       character
## 5            ended_at character POSIXct, POSIXt POSIXct, POSIXt POSIXct, POSIXt
## 6       member_casual character       character       character       character
## 7             ride_id character       character       character       character
## 8       rideable_type character       character       character       character
## 9           start_lat   numeric         numeric         numeric         numeric
## 10          start_lng   numeric         numeric         numeric         numeric
## 11   start_station_id character       character       character       character
## 12 start_station_name character       character       character       character
## 13         started_at character POSIXct, POSIXt POSIXct, POSIXt POSIXct, POSIXt
##          Jan_secyr     Jul_firstyr       Jul_secyr       Jun_secyr
## 1          numeric         numeric         numeric         numeric
## 2          numeric         numeric         numeric         numeric
## 3        character       character       character       character
## 4        character       character       character       character
## 5  POSIXct, POSIXt POSIXct, POSIXt POSIXct, POSIXt POSIXct, POSIXt
## 6        character       character       character       character
## 7        character       character       character       character
## 8        character       character       character       character
## 9          numeric         numeric         numeric         numeric
## 10         numeric         numeric         numeric         numeric
## 11       character       character       character       character
## 12       character       character       character       character
## 13 POSIXct, POSIXt POSIXct, POSIXt POSIXct, POSIXt POSIXct, POSIXt
##          Mar_secyr       May_secyr     Nov_firstyr Oct_firstyr     Sep_firstyr
## 1          numeric         numeric         numeric     numeric         numeric
## 2          numeric         numeric         numeric     numeric         numeric
## 3        character       character       character   character       character
## 4        character       character       character   character       character
## 5  POSIXct, POSIXt POSIXct, POSIXt POSIXct, POSIXt   character POSIXct, POSIXt
## 6        character       character       character   character       character
## 7        character       character       character   character       character
## 8        character       character       character   character       character
## 9          numeric         numeric         numeric     numeric         numeric
## 10         numeric         numeric         numeric     numeric         numeric
## 11       character       character       character   character       character
## 12       character       character       character   character       character
## 13 POSIXct, POSIXt POSIXct, POSIXt POSIXct, POSIXt   character POSIXct, POSIXt
```

Observing the structure of the data, there are inconsistencies in the datatype, the **started_at** and **ended_at** of **Apr_secyr and Oct_firstyr** are stored as characters instead of dates hence I convert these variables into the right type of datatype.

```
Apr_secyr$started_at <- ymd_hms(Apr_secyr$started_at)
Apr_secyr$ended_at <- ymd_hms(Apr_secyr$ended_at)
Oct_firstyr$started_at <- ymd_hms(Oct_firstyr$started_at)
Oct_firstyr$ended_at <- ymd_hms(Oct_firstyr$ended_at)
```

Now that all datatype are consistent lets combine all dataframe into a single dataframe ### Combining all the monthly dataframe into one dataframe

```r
bike_rides_df <-
  rbind(
    Apr_secyr,
    Aug_firstyr,
    Dec_firstyr,
    Feb_secyr,
    Jan_secyr,
    Jul_firstyr,
    Jul_secyr,
    Jun_secyr,
    Mar_secyr,
    May_secyr,
    Nov_firstyr,
    Oct_firstyr,
    Sep_firstyr
  )
```

Since we don't need the individual dataframe, lets remove them to have a clean environment to work with. ### Reomve individual dataframe

```r
rm(
  Apr_secyr,
  Dec_firstyr,
  Aug_firstyr,
  Feb_secyr,
  Jan_secyr,
  Jul_firstyr,
  Jul_secyr,
  Jun_secyr,
  Mar_secyr,
  May_secyr,
  Nov_firstyr,
  Oct_firstyr,
  Sep_firstyr
)
```

viewing the data combined data

```r
glimpse(bike_rides_df)
```

```
## Rows: 6,629,980
## Columns: 13
## $ ride_id            <chr> "47EC0A7F82E65D52", "8494861979B0F477", "EFE527AF80~
## $ rideable_type      <chr> "classic_bike", "electric_bike", "classic_bike", "c~
## $ started_at         <dttm> 2022-03-21 13:45:01, 2022-03-16 09:37:16, 2022-03-~
## $ ended_at           <dttm> 2022-03-21 13:51:18, 2022-03-16 09:43:34, 2022-03-~
## $ start_station_name <chr> "Wabash Ave & Wacker Pl", "Michigan Ave & Oak St", ~
## $ start_station_id   <chr> "TA1307000131", "13042", "13109", "TA1307000131", "~
## $ end_station_name   <chr> "Kingsbury St & Kinzie St", "Orleans St & Chestnut ~
## $ end_station_id     <chr> "KA1503000043", "620", "15578", "TA1305000025", "13~
## $ start_lat          <dbl> 41.88688, 41.90100, 41.97835, 41.88688, 41.91172, 4~
```

```
## $ start_lng          <dbl> -87.62603, -87.62375, -87.65975, -87.62603, -87.626~
## $ end_lat            <dbl> 41.88918, 41.89820, 41.98404, 41.87771, 41.87794, 4~
## $ end_lng            <dbl> -87.63851, -87.63754, -87.66027, -87.63532, -87.662~
## $ member_casual      <chr> "member", "member", "member", "member", "member", "~
```

```
head(bike_rides_df)
```

```
##           ride_id rideable_type          started_at            ended_at
## 1 47EC0A7F82E65D52  classic_bike 2022-03-21 13:45:01 2022-03-21 13:51:18
## 2 8494861979B0F477 electric_bike 2022-03-16 09:37:16 2022-03-16 09:43:34
## 3 EFE527AF80B66109  classic_bike 2022-03-23 19:52:02 2022-03-23 19:54:48
## 4 9F446FD9DEE3F389  classic_bike 2022-03-01 19:12:26 2022-03-01 19:22:14
## 5 431128AD9AFFEDC0  classic_bike 2022-03-21 18:37:01 2022-03-21 19:19:11
## 6 9AA8A13AF7A85325  classic_bike 2022-03-07 17:10:22 2022-03-07 17:15:04
##                 start_station_name start_station_id
## 1             Wabash Ave & Wacker Pl     TA1307000131
## 2              Michigan Ave & Oak St            13042
## 3              Broadway & Berwyn Ave            13109
## 4             Wabash Ave & Wacker Pl     TA1307000131
## 5 DuSable Lake Shore Dr & North Blvd           LF-005
## 6          Bissell St & Armitage Ave            13059
##                     end_station_name end_station_id start_lat start_lng
## 1             Kingsbury St & Kinzie St   KA1503000043  41.88688 -87.62603
## 2 Orleans St & Chestnut St (NEXT Apts)            620  41.90100 -87.62375
## 3                 Broadway & Ridge Ave          15578  41.97835 -87.65975
## 4           Franklin St & Jackson Blvd   TA1305000025  41.88688 -87.62603
## 5             Loomis St & Jackson Blvd          13206  41.91172 -87.62680
## 6         Southport Ave & Clybourn Ave   TA1309000030  41.91802 -87.65218
##    end_lat   end_lng member_casual
## 1 41.88918 -87.63851        member
## 2 41.89820 -87.63754        member
## 3 41.98404 -87.66027        member
## 4 41.87771 -87.63532        member
## 5 41.87794 -87.66201        member
## 6 41.92077 -87.66371        member
```

```
str(bike_rides_df)
```

```
## 'data.frame':    6629980 obs. of  13 variables:
##  $ ride_id           : chr  "47EC0A7F82E65D52" "8494861979B0F477" "EFE527AF80B66109" "9F446FD9DEE3F38
##  $ rideable_type     : chr  "classic_bike" "electric_bike" "classic_bike" "classic_bike" ...
##  $ started_at        : POSIXct, format: "2022-03-21 13:45:01" "2022-03-16 09:37:16" ...
##  $ ended_at          : POSIXct, format: "2022-03-21 13:51:18" "2022-03-16 09:43:34" ...
##  $ start_station_name: chr  "Wabash Ave & Wacker Pl" "Michigan Ave & Oak St" "Broadway & Berwyn Ave"
##  $ start_station_id  : chr  "TA1307000131" "13042" "13109" "TA1307000131" ...
##  $ end_station_name  : chr  "Kingsbury St & Kinzie St" "Orleans St & Chestnut St (NEXT Apts)" "Broadw
##  $ end_station_id    : chr  "KA1503000043" "620" "15578" "TA1305000025" ...
##  $ start_lat         : num  41.9 41.9 42 41.9 41.9 ...
##  $ start_lng         : num  -87.6 -87.6 -87.7 -87.6 -87.6 ...
##  $ end_lat           : num  41.9 41.9 42 41.9 41.9 ...
##  $ end_lng           : num  -87.6 -87.6 -87.7 -87.6 -87.7 ...
##  $ member_casual     : chr  "member" "member" "member" "member" ...
```

To compute the length of each ride, I need to exxtract start and end time from the started__at and ended__at colimns. I extracted hours, mins and secs from the started__at and ended__at columns and add new columns which will be start__time and end__time

```
bike_rides_df$start_time <-
  hms::as_hms(bike_rides_df$started_at)
bike_rides_df$end_time <-
  hms::as_hms(bike_rides_df$ended_at)
```

Extraction of start date and end date

```
bike_rides_df$start_date <-
  as_date(bike_rides_df$started_at)
bike_rides_df$end_date <-
  as_date(bike_rides_df$ended_at)
```

```
bike_rides_df$start_hour <-
  strptime(bike_rides_df$start_time, "%H") %>%
  hour()
```

Extracting months from the date and converting them to month names

```
bike_rides_df$month <-
  as.yearmon(bike_rides_df$start_date)
```

Some of the variables are irrelevant in answering the questions I aim to answer. Hence I will remove such colunns

```
bike_rides_df <- bike_rides_df %>%
  select(
    rideable_type,
    started_at,
    ended_at,
    start_station_name,
    end_station_name,
    member_casual,
    start_time,
    end_time,
    start_date,
    month,
    start_hour
  ) %>%
  rename(
    "bike_type" = rideable_type,
    "customer_type" = member_casual,
    "ride_date" = start_date
  )
```

Extract day of the week from date the trip started

```
bike_rides_df$week_day <- weekdays(bike_rides_df$ride_date)
```

Create trip length from the started__at and ended__at

```
options(scipen = 999)
bike_rides_df$trip_duration <-
  as.double(difftime(bike_rides_df$end_time, bike_rides_df$start_time)) / 60
```

To avoid errors in our data visualization and error in the analysis, we need to check if there is and trip duration that was less than 0. If there is, we will exclude them. These trips with less than 0 trip duration were / could be as a result of the test trip conducted by the company hence we will exclude them. The trips with their start station containing **"test"** in their names are the test trip the company made.

Examining if there is a trip duration that was less than 0

```
bike_rides_df %>%
  select(trip_duration) %>%
  filter(trip_duration < 0) %>%
  count()
```

```
##       n
## 1 44046
```

```
# count how many trips were test trips made by the company
grep("test", bike_rides_df$start_station_name, value = TRUE)
```

```
## character(0)
```

```
grep("Test", bike_rides_df$start_station_name, value = TRUE)
```

```
## [1] "Pawel Bialowas - Test- PBSC charging station"
```

```
grep("TEST", bike_rides_df$start_station_name, value = TRUE)
```

```
## character(0)
```

Removing all trip duration that were less than 0

```
bike_rides_df_new <- bike_rides_df[!(bike_rides_df$trip_duration < 0), ]

# Check if the test trip and station is removed
grep("Test", bike_rides_df_new$start_station_name, value = TRUE)
```

```
## [1] "Pawel Bialowas - Test- PBSC charging station"
```

```
rm(bike_rides_df)
```

Removing test trip

```
bike_rides_df_new <-
  bike_rides_df_new[!grepl("Test", bike_rides_df_new$start_station_name), ]
```

view final dataframe before analysis and give levels to customer type and bike type

```
bike_rides_df_new$customer_type <-
  factor(
    bike_rides_df_new$customer_type,
    levels = c("member", "casual")
  )
bike_rides_df_new$bike_type <-
  factor(
    bike_rides_df_new$bike_type,
    levels = c("classic_bike", "electric_bike", "docked_bike")
  )
glimpse(bike_rides_df_new)
```

```
## Rows: 6,585,933
## Columns: 13
## $ bike_type          <fct> classic_bike, electric_bike, classic_bike, classic_~
## $ started_at         <dttm> 2022-03-21 13:45:01, 2022-03-16 09:37:16, 2022-03-~
## $ ended_at           <dttm> 2022-03-21 13:51:18, 2022-03-16 09:43:34, 2022-03-~
## $ start_station_name <chr> "Wabash Ave & Wacker Pl", "Michigan Ave & Oak St", ~
## $ end_station_name   <chr> "Kingsbury St & Kinzie St", "Orleans St & Chestnut ~
## $ customer_type      <fct> member, member, member, member, member, member, mem~
## $ start_time         <time> 13:45:01, 09:37:16, 19:52:02, 19:12:26, 18:37:01, ~
## $ end_time           <time> 13:51:18, 09:43:34, 19:54:48, 19:22:14, 19:19:11, ~
## $ ride_date          <date> 2022-03-21, 2022-03-16, 2022-03-23, 2022-03-01, 20~
## $ month              <yearmon> Mar 2022, Mar 2022, Mar 2022, Mar 2022, Mar 202~
## $ start_hour         <int> 13, 9, 19, 19, 18, 17, 17, 12, 17, 19, 21, 7, 16, 2~
## $ week_day           <chr> "Monday", "Wednesday", "Wednesday", "Tuesday", "Mon~
## $ trip_duration      <dbl> 6.283333, 6.300000, 2.766667, 9.800000, 42.166667, ~
```

4 & 5. Analyze & Share

Our data frame is now ready to uncover some insights on how casual riders use Cyclistic bikes differently from member riders. Lets first of all take a statistical summary of the trip duration

```
attach(bike_rides_df_new)
summary(trip_duration)
```

```
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
##    0.000    6.383   11.350   17.660   20.483 1424.700
```

Agreggate summary by customer type

```
bike_rides_df_new %>%
  select(trip_duration, customer_type) %>%
  group_by(customer_type) %>%
  summarise(
    min_trip = min(trip_duration),
    mean_trip = mean(trip_duration),
    max_trip = max(trip_duration),
    median_trip = median(trip_duration),
    n = n()
  ) %>%
  mutate(propor = n / sum(n)) %>%
  group_by(customer_type)
```

```
## # A tibble: 2 x 7
## # Groups:   customer_type [2]
##   customer_type min_trip mean_trip max_trip median_trip       n propor
##   <fct>            <dbl>     <dbl>    <dbl>       <dbl>   <int>  <dbl>
## 1 member               0      12.7    1249.        9.18 3689039  0.560
## 2 casual               0      24.0    1425.       15.0  2896894  0.440
```

Looking at the mean for the overall trip duration which is 17.6mins, we can say that member riders has less trip duration than the overall mean trip duration. Casual riders on the other hand have a trip duration greater than the overall trip duration.
It is also necessary to mention that whiles the duration of trip taken by casual riders on average is longer than the average trip duration of member riders, member riders take more trip than casual riders

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

```
bike_rides_df_new %>%
  select(
    customer_type,
    week_day,
    trip_duration
  ) %>%
  group_by(customer_type, week_day) %>%
  summarise(
    number_of_trips = n(),
    average_trip_mins = mean(trip_duration)
  ) %>%
  arrange(customer_type, desc(number_of_trips))
```

```
## `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 week_day  number_of_trips average_trip_mins
```

```
##    <fct>         <chr>              <int>              <dbl>
##  1 member        Wednesday          580003              12.0
##  2 member        Tuesday            578550              11.9
##  3 member        Thursday           573575              12.1
##  4 member        Friday             514537              12.4
##  5 member        Monday             511911              12.2
##  6 member        Saturday           487822              14.1
##  7 member        Sunday             442641              14.4
##  8 casual        Saturday           602110              26.3
##  9 casual        Sunday             531861              27.8
## 10 casual        Friday             409279              22.4
## 11 casual        Thursday           362640              21.2
## 12 casual        Monday             336550              24.5
## 13 casual        Wednesday          331301              20.9
## 14 casual        Tuesday            323153              21.4
```

Visualize total trips by customer type and day of the week

```
trip_week_day <- bike_rides_df_new %>%
  select(
    customer_type,
    week_day,
    trip_duration
  ) %>%
  group_by(customer_type, week_day) %>%
  summarise(
    number_of_trips = n(),
    average_trip_mins = mean(trip_duration)
  ) %>%
  arrange(customer_type, desc(number_of_trips))
```

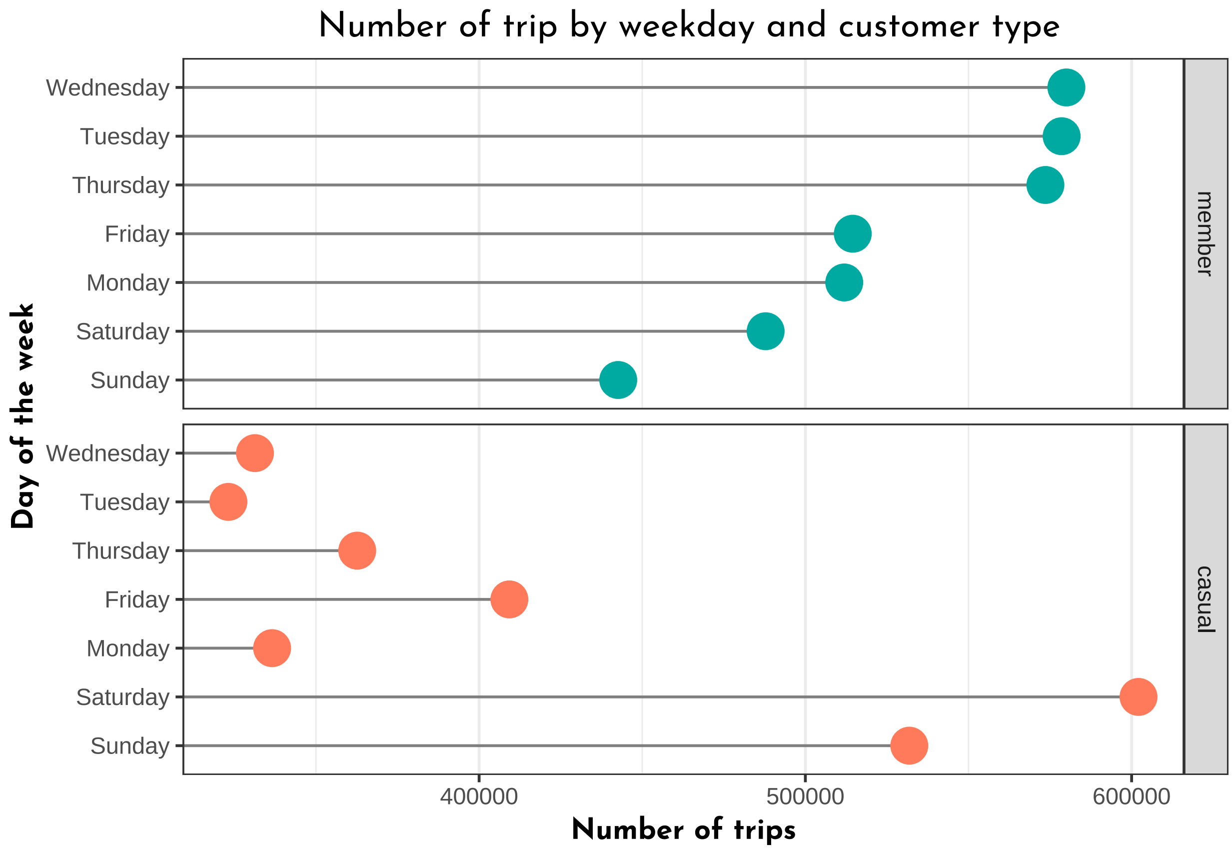
```
## `summarise()` has grouped output by 'customer_type'. You can override using the
## `.groups` argument.
```

```
weekorder <- trip_week_day$week_day[
  order(
    trip_week_day$customer_type,
    trip_week_day$number_of_trips
  )
]

trip_week_day$week_day <- factor(trip_week_day$week_day, levels = unique(weekorder))

ggplot(
  trip_week_day,
  aes(x = number_of_trips, y = week_day)
) +
  geom_segment(
    aes(yend = week_day),
    xend = 0, colour = "grey50"
  ) +
```

```
geom_point(
  size = 6, aes(colour = customer_type)
) +
scale_colour_manual(
  values = c("#00AAA0", "#FF7A5A"),
  limits = c("member", "casual"),
  guide = "none"
) +
theme_bw() +
theme(
  panel.grid.major.y = element_blank(),
  axis.title = element_text(
    face = "bold",
    family = "josefin-new"
  ),
  plot.title = element_text(
    family = "josefin-new",
    hjust = 0.5
  )
) +
labs(
  title = " Number of trip by weekday and customer type",
  x = "Number of trips",
  y = "Day of the week"
) +
facet_grid(customer_type ~ ., scales = "free_y", space = "free_y") +
scale_x_continuous(labels = ~ format(.x, scientific = FALSE))
```



Looking at the table and graph above, it can be concluded that, ***members*** are very active users of the bike during week days. **Members** have most of their trips on Wednesday, Tuesday and Thursday and they have less trips during the weekends. On the hand casual riders are mostly using the service during the weekends. An interesting pattern to take note is that, apart from the weekends where casual riders uses the bike service more than the member riders, member riders are consistently using the service through out the week. Additional customer personal data could be collected to figure out the underlying reasons. The assumption made here is that, members use the bikes for work related activities while casual workers normal ride for fun or for other reasons than work related reasons. Additional data would be needed to text this assumption.

Total number of trips by customer type and bike type

```
bike_rides_df_new %>%
  select(
    bike_type,
    trip_duration,
    customer_type
  ) %>%
  group_by(bike_type, customer_type) %>%
  summarise(
    average_trip_mins = mean(trip_duration),
    number_of_trips = n()
  ) %>%
  arrange(
```

```
    customer_type,
    desc(average_trip_mins)
  )
```

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

```
## # A tibble: 5 x 4
## # Groups:   bike_type [3]
##   bike_type     customer_type average_trip_mins number_of_trips
##   <fct>         <fct>                     <dbl>           <int>
## 1 classic_bike  member                     13.1         2211063
## 2 electric_bike member                     12.0         1477976
## 3 docked_bike   casual                     45.0          297828
## 4 classic_bike  casual                     24.3         1391789
## 5 electric_bike casual                     18.5         1207277
```

Visualize Total number of trips by bike and customer type and average duration

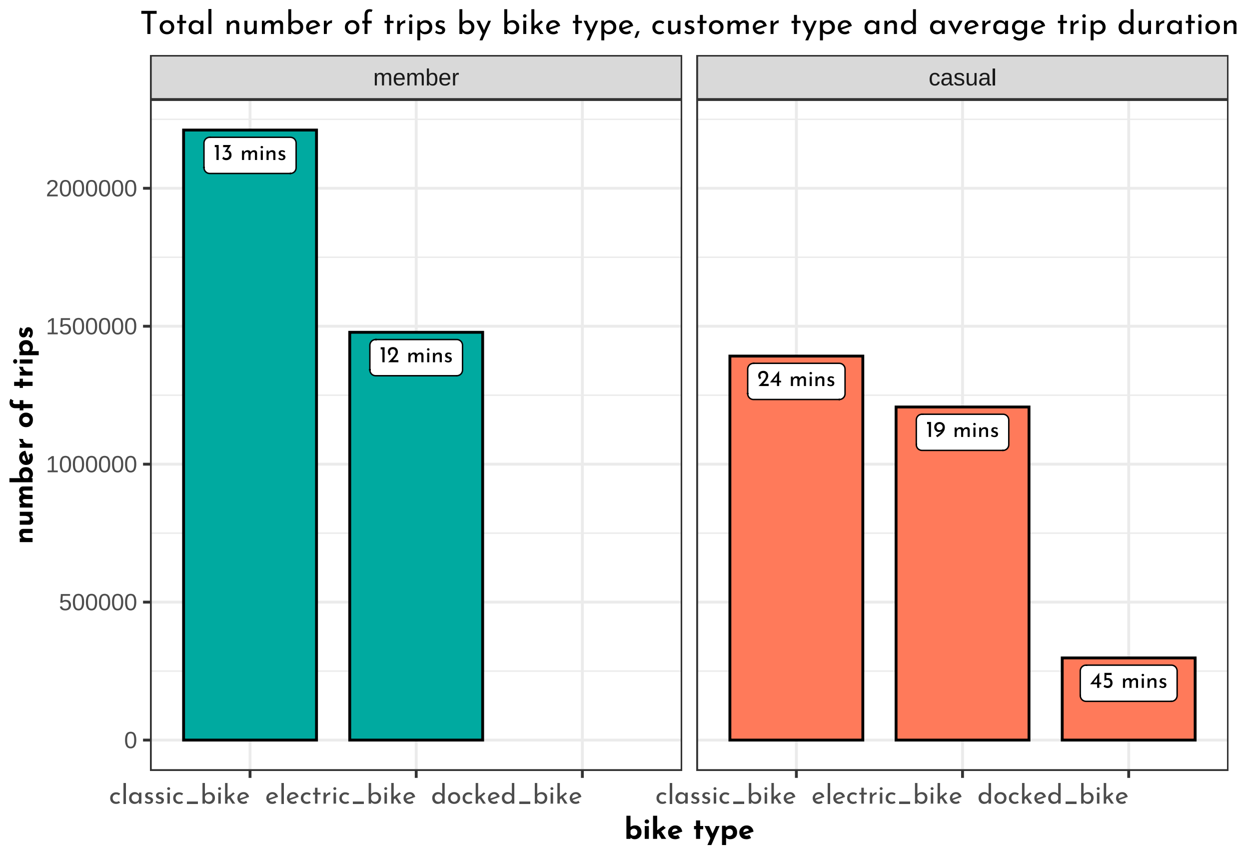
```
bike_type_number_trips <- bike_rides_df_new %>%
  select(
    bike_type,
    trip_duration,
    customer_type
  ) %>%
  group_by(bike_type, customer_type) %>%
  summarise(
    average_trip_mins = mean(trip_duration),
    number_of_trips = n()
  ) %>%
  arrange(
    customer_type,
    (average_trip_mins)
  )
```

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

```
bike_type_number_trips <- bike_type_number_trips %>%
  mutate(percentage = number_of_trips / sum(number_of_trips))

ggplot(
  bike_type_number_trips, aes(x = bike_type, y = number_of_trips, fill = customer_type)
) +
  geom_col(position = "dodge", color = "black", width = 0.8) +
  labs(
    title = "Total number of trips by bike type, customer type and average trip duration",
    x = "bike type",
    y = "number of trips",
    fill = "customer type"
```

```r
) +
theme_bw() +
scale_fill_manual(values = c("#00AAA0", "#FF7A5A"), guide = "none") +
geom_label(aes(label = paste(round(average_trip_mins, 0), "mins")),
  position = position_dodge(0.8),
  size = 3,
  vjust = 1.2,
  hjust = 0.5,
  colour = "black",
  family = "josefin-new",
  fill = "white"
) +
facet_wrap(~customer_type) +
theme(
  axis.text.x = element_text(
    hjust = 1,
    family = "josefin-new",
    size = 10
  ),
  axis.title = element_text(
    face = "bold",
    family = "josefin-new"
  ),
  plot.title = element_text(
    family = "josefin-new",
    size = 12,
    hjust = 0.5,
    vjust = 1
  )
)
```



From the table and graph above, we can see that majority of members usually use class classic bikes and their average trip duration is lesser than the average trip duration for casual bike users who uses the same type of bike. An interesting pattern to notice here is that, member riders hardly use dock bikes instead casual riders uses this type of bike and they have the longest average trip duration of about 45mins but also people tend to use it less among all both customer types.

Total monthly trips by customer type

```
bike_rides_df_new %>%
  select(
    month,
    customer_type,
    trip_duration
  ) %>%
  group_by(month, customer_type) %>%
  summarise(
    average_trip_mins = mean(trip_duration),
    number_of_trips = n()
  ) %>%
  arrange(desc(number_of_trips))
```

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

```
## # A tibble: 26 x 4
## # Groups:   month [13]
##    month     customer_type average_trip_mins number_of_trips
##    <yearmon> <fct>                     <dbl>           <int>
##  1 Jul 2021  casual                     26.3          436318
##  2 Aug 2021  casual                     25.2          408229
##  3 Jun 2022  member                     13.4          398688
##  4 Sep 2021  member                     13.3          390954
##  5 Aug 2021  member                     13.7          390303
##  6 Jul 2021  member                     13.8          378778
##  7 Oct 2021  member                     12.0          372780
##  8 Jun 2022  casual                     22.5          365377
##  9 Jun 2021  casual                     27.6          364684
## 10 Sep 2021  casual                     24.1          360200
## # ... with 16 more rows
```

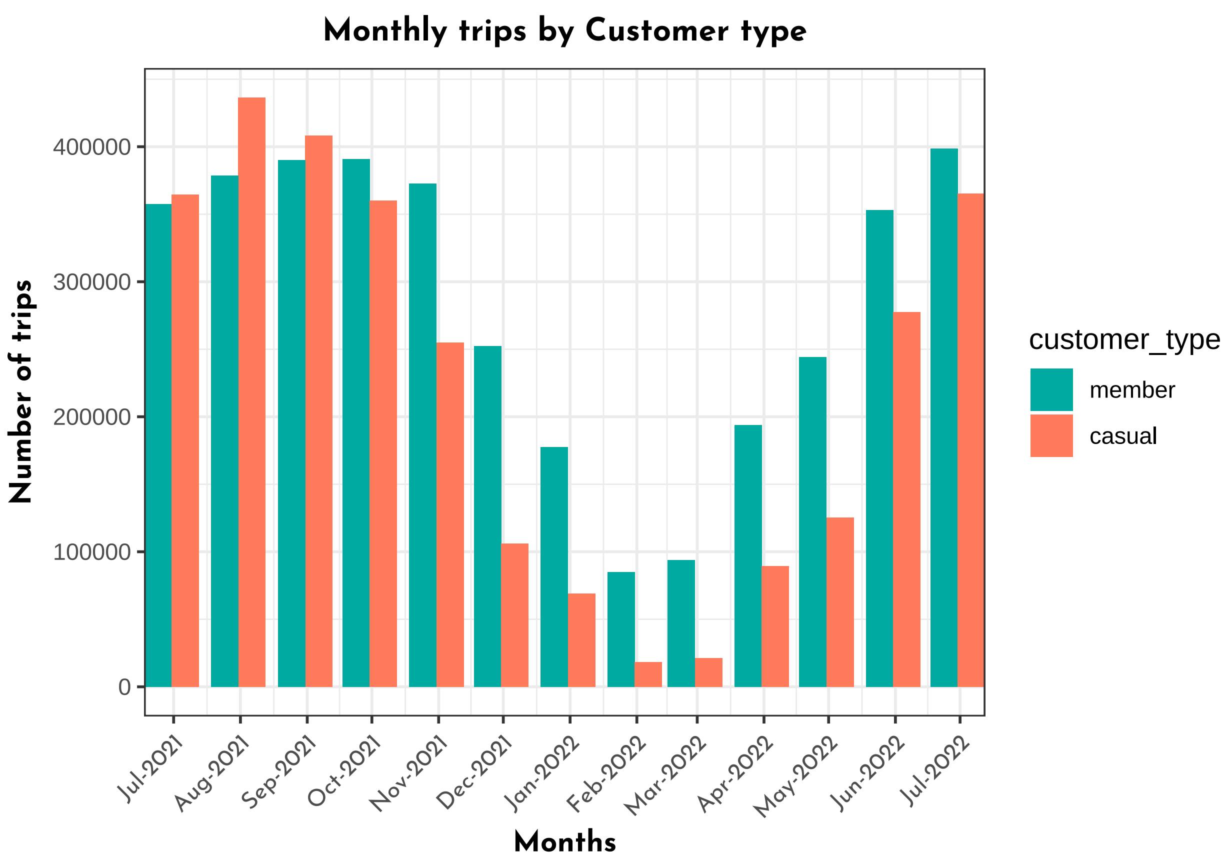
Visualize the total number of trips per month by customer type

```
monthly_ride_df <- bike_rides_df_new %>%
  select(
    month,
    customer_type,
    trip_duration
  ) %>%
  group_by(month, customer_type) %>%
  summarise(
    average_trip_mins = mean(trip_duration),
    number_of_trips = n()
  ) %>%
  arrange(desc(number_of_trips))
```

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

```
ggplot(
  monthly_ride_df, aes(
    x = (as.Date.yearmon(monthly_ride_df$month, 1)),
    y = number_of_trips, fill = customer_type
  )
) +
  geom_col(position = "dodge") +
  scale_x_date(
    date_breaks = "month",
    date_labels = "%b-%Y",
    expand = c(0, 0)
  ) +
  theme_bw() +
  theme(
    axis.title.y = element_text(
      face = "bold",
      family = "josefin-new",
```

```
      vjust = 2.5
    ),
    axis.text.x = element_text(
      angle = 45, hjust = 1.03,
      family = "josefin-new"
    ),
    axis.title.x = element_text(
      family = "josefin-new",
      face = "bold"
    ),
    plot.title = element_text(
      hjust = 0.5,
      vjust = 2,
      family = "josefin-new",
      size = 12,
      face = "bold"
    )
  ) +
  labs(title = "Monthly trips by Customer type", x = "Months", y = "Number of trips") +
  scale_fill_manual(values = c("#00AAA0", "#FF7A5A")) +
  scale_y_continuous(labels = ~ format(.x, scientific = FALSE))
```



From the graph above we can see an upward trend between July and until somewhere in November. Within this period, both casual and member riders have high demand of the bikes. Another important trend to notice is that, from December until the beginning of July of the subsequent year, demand of bikes by both customer segment falls. This could be the a seasonality factor. Since it is expected that the weather becomes cold from December (changes from summer) until June. Further data could be collected to test the impact of factors such as weather, seasonality etc on the demand of bikes by both groups. However comparing the number of trips made in 2021 is relatively higher than 2022. Most trips made between Jul 2021 and September 2021 are made by casual riders hence casual riders demand are mostly between the summer time and this trend changes as most trips between from October 2021 are made by member riders. Additional analysis could be done to figure out why this happened.

Top most demanding hours (24 hours) of bikes by customer type

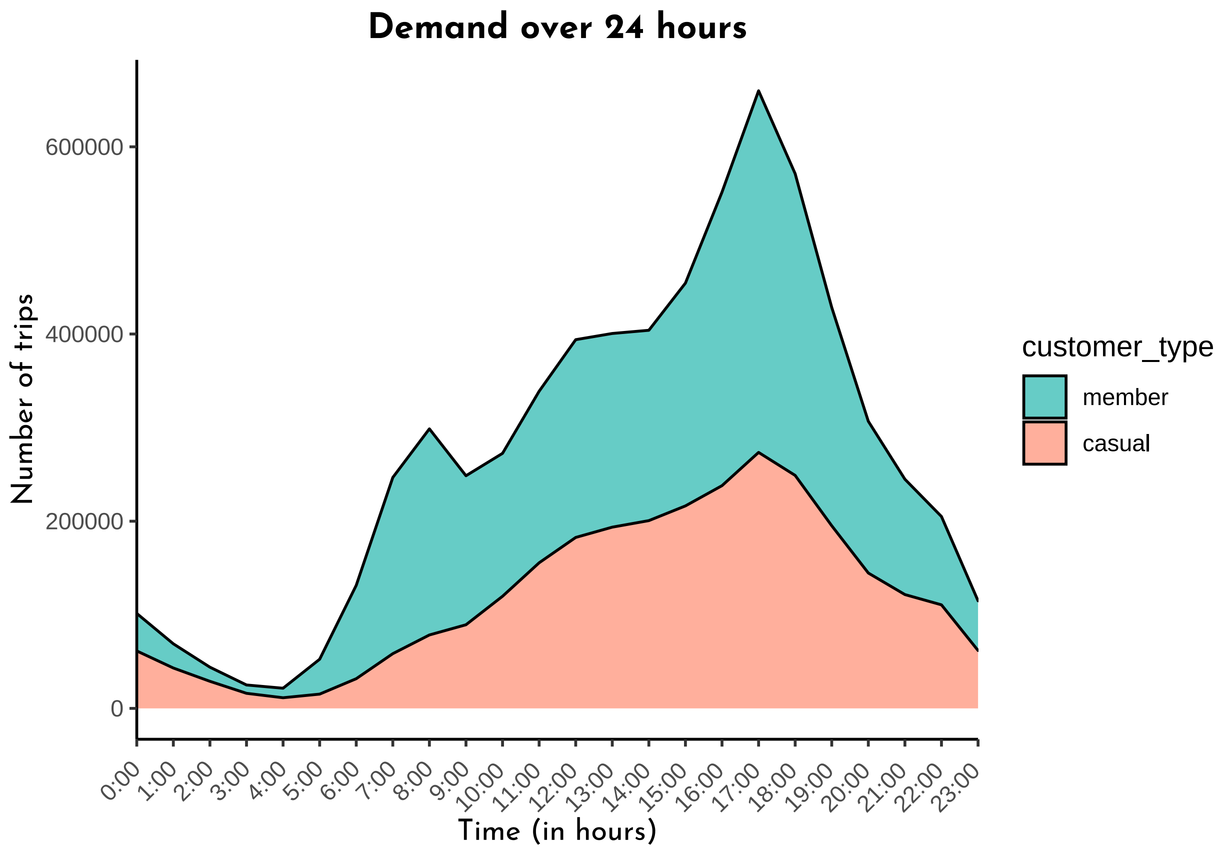
```
bike_rides_df_new %>%
  select(start_hour, trip_duration, customer_type) %>%
  group_by(start_hour, customer_type) %>%
  summarise(number_of_trips = n(), mean_trip = mean(trip_duration)) %>%
  arrange(desc(number_of_trips))
```

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

```
## # A tibble: 48 x 4
## # Groups:   start_hour [24]
##    start_hour customer_type number_of_trips mean_trip
##         <int> <fct>                   <int>     <dbl>
##  1         17 member                 386406      13.3
##  2         18 member                 322340      13.2
##  3         16 member                 313637      13.2
##  4         17 casual                 273501      22.8
##  5         18 casual                 248906      22.1
##  6         15 member                 238080      13.1
##  7         16 casual                 237952      24.5
##  8         19 member                 233258      12.9
##  9          8 member                 220234      11.7
## 10         15 casual                 216349      26.3
## # ... with 38 more rows
```

```
ggplot(
  bike_rides_df_new, aes(x = start_hour)
) +
  geom_area(
    aes(fill = customer_type),
    stat = "count",
    size = 0.5,
    linetype = 1,
    color = "black",
    alpha = 0.6
  ) +
  scale_x_continuous(
    breaks = seq(0, 23, by = 1),
    expand = c(0, 0),
    labels = function(x) glue("{x}:00")
  ) +
  scale_y_continuous(labels = ~ format(.x, scientific = FALSE)) +
```

```
  theme_classic() +
  theme(
    axis.title = element_text(family = "josefin-new"),
    plot.title = element_text(family = "josefin-new", face = "bold", hjust = 0.5),
    axis.text.x = element_text(angle = 45, hjust = 1.2)
  ) +
  labs(title = "Demand over 24 hours", x = "Time (in hours)", y = "Number of trips") +
  scale_fill_manual(values = c("#00AAA0", "#FF7A5A"))
```



Based on the graph and table above, members have two peaks periods where demand is high. The number of trips made by members are at peak around 7am -9am and then fall afterwards and the demand rises again around 5pm - 7pm. The peak demand of 5pm - 7pm also coincide with the peak demand of casual riders as well. The trend in demand by member riders support my initial assumption that member riders are mostly office workers and they may use the bike for commuting to and from work. More data would be needed to confirm this assumption

Average trip duration by day of the week and customer type

```
bike_rides_df_new %>%
  select(week_day, customer_type, trip_duration) %>%
  group_by(week_day, customer_type) %>%
  summarise(average_trip_duration = mean(trip_duration)) %>%
  arrange(desc(average_trip_duration))
```

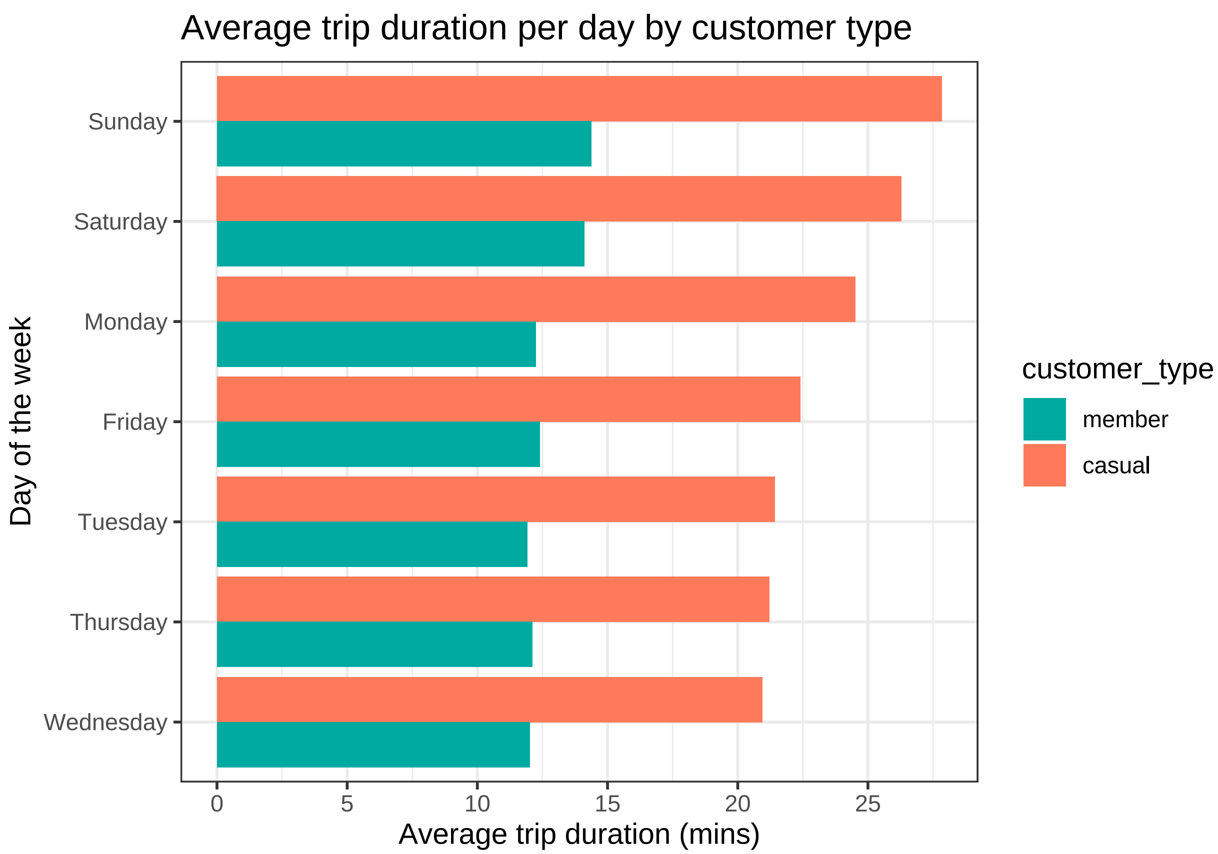
```
## ‘summarise()‘ has grouped output by ’week_day’. You can override using the
## ‘.groups‘ argument.

## # A tibble: 14 x 3
## # Groups:   week_day [7]
##    week_day  customer_type average_trip_duration
##    <chr>     <fct>                         <dbl>
##  1 Sunday    casual                         27.8
##  2 Saturday  casual                         26.3
##  3 Monday    casual                         24.5
##  4 Friday    casual                         22.4
##  5 Tuesday   casual                         21.4
##  6 Thursday  casual                         21.2
##  7 Wednesday casual                         20.9
##  8 Sunday    member                         14.4
##  9 Saturday  member                         14.1
## 10 Friday    member                         12.4
## 11 Monday    member                         12.2
## 12 Thursday  member                         12.1
## 13 Wednesday member                         12.0
## 14 Tuesday   member                         11.9
```

```
trip_dur_by_day <- bike_rides_df_new %>%
  select(week_day, customer_type, trip_duration) %>%
  group_by(week_day, customer_type) %>%
  summarise(average_trip_duration = mean(trip_duration)) %>%
  arrange(desc(average_trip_duration))
```

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

```
trip_dur_by_day %>%
  ggplot(
    aes(x = reorder(week_day, +average_trip_duration), y = average_trip_duration)
  ) +
  geom_col(aes(fill = customer_type), position = "dodge") +
  scale_y_continuous(breaks = seq(0, 28, by = 5)) +
  coord_flip() +
  labs(
    title = "Average trip duration per day by customer type",
    x = "Day of the week",
    y = "Average trip duration (mins)"
  ) +
  theme(
    axis.title = element_text(family = "josefin-new"),
    plot.title = element_text(family = "josefin-new", hjust = 0.5),
  ) +
  theme_bw() +
  scale_fill_manual(values = c("#00AAA0", "#FF7A5A"))
```



On a average the trip duration in mins of casual riders is as twice as the trip duration of member riders. This could be attribute to several factors such as the type of bike used. As indicated earlier, majority of member riders do use electric bikes and classic bikes than the number of casual riders who use similar bikes. The mins traveled doesn't equate to how far user went. Interestingly, casual riders do not have only their longest trip on weekends but also on average they have most of their trips on weekends.

6. Act

Important findings

- 44% of the total rides were taken by casual riders while 56% of the total ride made between July 2021 t0 July 2022 were made by member riders.
- The overall trip duration was 17.6 mins. Casual riders average trip duration exceeded the overall trip duration whiles member riders had less trip duration as compared to the overall trip duration.
- Number of trips by casual riders are higher during the weekends whiles member riders are consistently riding throughout the week.
- The highest number of trips made by member riders are made with the classic bike and the average duration is 13mins. The least trip duration of member riders on average is 12mins and its made with the electric bike. On the other hand, casual riders uses all three types of bikes and the average trip duration takes 24mins for classic bike. The highest average duration for casual riders are those who uses docked bike.
- In 2021 Casual riders were riding more in August and September whiles member riders were taking trips in October and November. There is a downward trend for both riders during Fall and Winter.

- The most demanding hour for the bikes are between 5pm and 7pm. Though member riders have two peaks which is demand is high in the morning between 7am and 9am but casual riders have just one peak.

Recommendations

- Since casual riders demands are high in certain months and drops during other months, offer membership for at least three months. This will gradually turn casual members into annual riders after they seen the number of benefits they would enjoy for these short monthly membership.
- Marketing message should focus on paying same amount for more trip duration. Since casual riders pay service fee based on the duration they use the bike (assumption is pay as you use service) and the average duration of casual riders are higher than member riders, key marketing message should leverage on the trip duration by casual riders
- Offer discount during weekdays for casual riders since the number of trips made by this segment of riders are mostly high on weekends but low during weekends.
- Use peak pricing strategy during the peak demand periods. Since both member and casual riders have a peak demand between 5pm and 7pm, peak pricing strategy could be used where prices of casual riders who pay for the service as they use would be increased. This will force casual riders to opt for membership since the increase in price at peak hours won't affect members

Future goals and data that could help dive deep to better understand and make more concrete recommendations

- Data about demographics could help dive deep into how certain demographics such as age, gender, income, locate etc affect each customer segment usage of the bike.
- Pricing and sales data could help understand build models to know other factors which affect these two customer segment.
- Certain assumptions were made about how the employment status affect both segment differently as well how seasonality have impact differently on these customer segments hence data about customers, weather etc