# Case Study -1

Zhantore Nurbay

## Introduction

This is my version of the Google Data Analytics Capstone - Case Study 1. The full document to the case study can be found in the Google Data Analytics Capstone: Complete a Case Study course. This is my data preparation,analysis and data visualzation phase. The result of this phase will go directly to my Business case presentation. The company is focused on renting bikes and our main goal is to find diffrence between two customer types: member - a subscriber, and a casual member ## R Markdown

## Prepare

The project will use the data provided by Google. We have 12 month worth of data.

#### First step - prepare the data for analysis. All the csv files will be merged into one file

Loading libraries

The main libraries

```
library(tidyverse)

library(janitor)

library(lubridate)

library(readr)
library(ggplot2)
```

### Concatenating

All the csvs files will be concatenated into one dataframe. My data is located in folder 2021

```
setwd("D:/data/education/Personal/google/case_study_bike_data/2021")
getwd()

## [1] "D:/data/education/Personal/google/case_study_bike_data/2021"
```

### Giving each month a variable for further consolidation

### COMBINE DATA INTO A SINGLE FILE

merging files to check wheter the names in data set are the same

```
dflist <- mget(paste0("m", 1:12))
nametest <- sapply(dflist,function(x){names(x) %in% names(m1)})
table(nametest)

## nametest
## TRUE
##  156
```

Usually you would use view function to see where exactly is the mistake in column name was made. However, we can see that all values are true which means all columns share the same name

## Consolidating into one data frame

```
bike_data <- rbind(m1,m2,m3,m4,m5,m6,m7,m8,m9,m10,m11,m12)
```

## Data cleaning

Quick look at the data,finding NA’s, empty columns, duplicates

First, we will take a look at the data

```
summary(bike_data)

##    ride_id          rideable_type        started_at
##  Length:5595063     Length:5595063     Min.   :2021-01-01 00:02:05
##  Class :character   Class :character   1st Qu.:2021-06-06 23:52:40
##  Mode  :character   Mode  :character   Median :2021-07-31 19:52:11
##                                        Mean   :2021-07-29 05:07:43
##                                        3rd Qu.:2021-09-24 10:36:16
##                                        Max.   :2021-12-31 23:59:48
##
##     ended_at                   start_station_name start_station_id
##  Min.   :2021-01-01 00:08:39   Length:5595063     Length:5595063
##  1st Qu.:2021-06-07 00:44:21   Class :character   Class :character
##  Median :2021-07-31 20:21:55   Mode  :character   Mode  :character
##  Mean   :2021-07-29 05:29:39
##  3rd Qu.:2021-09-24 10:54:05
##  Max.   :2022-01-03 17:32:18
##
##  end_station_name   end_station_id       start_lat       start_lng
##  Length:5595063     Length:5595063     Min.   :41.64   Min.   :-87.84
##  Class :character   Class :character   1st Qu.:41.88   1st Qu.:-87.66
##  Mode  :character   Mode  :character   Median :41.90   Median :-87.64
##                                        Mean   :41.90   Mean   :-87.65
##                                        3rd Qu.:41.93   3rd Qu.:-87.63
##                                        Max.   :42.07   Max.   :-87.52
##
##     end_lat          end_lng        member_casual
##  Min.   :41.39   Min.   :-88.97   Length:5595063
##  1st Qu.:41.88   1st Qu.:-87.66   Class :character
##  Median :41.90   Median :-87.64   Mode  :character
##  Mean   :41.90   Mean   :-87.65
##  3rd Qu.:41.93   3rd Qu.:-87.63
##  Max.   :42.17   Max.   :-87.49
##  NA's   :4771    NA's   :4771
```

```
str(bike_data)
```

```
## spec_tbl_df [5,595,063 x 13] (S3: spec_tbl_df/tbl_df/tbl/data.frame)
##  $ ride_id           : chr [1:5595063] "E19E6F1B8D4C42ED" "DC88F20C2C55F27F"
"EC45C94683FE3F27" "4FA453A75AE377DB" ...
##  $ rideable_type     : chr [1:5595063] "electric_bike" "electric_bike" "elect
ric_bike" "electric_bike" ...
##  $ started_at        : POSIXct[1:5595063], format: "2021-01-23 16:14:19" "202
1-01-27 18:43:08" ...
##  $ ended_at          : POSIXct[1:5595063], format: "2021-01-23 16:24:44" "202
1-01-27 18:47:12" ...
##  $ start_station_name: chr [1:5595063] "California Ave & Cortez St" "Californ
ia Ave & Cortez St" "California Ave & Cortez St" "California Ave & Cortez St" ..
.
##  $ start_station_id  : chr [1:5595063] "17660" "17660" "17660" "17660" ...
##  $ end_station_name  : chr [1:5595063] NA NA NA NA ...
##  $ end_station_id    : chr [1:5595063] NA NA NA NA ...
##  $ start_lat         : num [1:5595063] 41.9 41.9 41.9 41.9 41.9 ...
##  $ start_lng         : num [1:5595063] -87.7 -87.7 -87.7 -87.7 -87.7 ...
##  $ end_lat           : num [1:5595063] 41.9 41.9 41.9 41.9 41.9 ...
##  $ end_lng           : num [1:5595063] -87.7 -87.7 -87.7 -87.7 -87.7 ...
##  $ member_casual     : chr [1:5595063] "member" "member" "member" "member" ..
.
##  - attr(*, "spec")=
##   .. cols(
##   ..   ride_id = col_character(),
##   ..   rideable_type = col_character(),
##   ..   started_at = col_datetime(format = ""),
##   ..   ended_at = col_datetime(format = ""),
##   ..   start_station_name = col_character(),
##   ..   start_station_id = col_character(),
##   ..   end_station_name = col_character(),
##   ..   end_station_id = col_character(),
##   ..   start_lat = col_double(),
##   ..   start_lng = col_double(),
##   ..   end_lat = col_double(),
##   ..   end_lng = col_double(),
##   ..   member_casual = col_character()
##   .. )
##  - attr(*, "problems")=<externalptr>
```

Overall, formatting and data looks fine, except maybe for NA’s. Now we will check for any duplicates We have to remember the number **5,595,063** that’s a number of rows we have in the start

```
bike_data <- janitor::remove_empty(bike_data,which = c("cols"))
bike_data <- janitor::remove_empty(bike_data,which = c("rows"))
dim(bike_data)
```

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

We didnt find any empty rows.

### Manipulating the data

New columns that are needed to perform ananlysis, we will make a seperate date column and divide it in month,year,day and which day of the week the the operation has been recorded. At the end will find the amount of time of a rent for each record

## Ride date

Add columns that list the date, month, day, and year of each ride

```
bike_data$date <- as.Date(bike_data$started_at)
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")
bike_data$day_of_week <-format(as.Date(bike_data$date), "%A")
bike_data <- bike_data %>%
  mutate(start_hour = strftime(bike_data$started_at, "%H"))
bike_data <- bike_data %>%
  mutate(end_hour = strftime(bike_data$ended_at, "%H"))
```

## Ride length

Add a “ride_length” calculation to all_trips (in minutes and hours) and changing format to difftime

```
bike_data$hour_length <- difftime(bike_data$ended_at,bike_data$started_at,units
= c("hours"))
bike_data$min_length <- difftime(bike_data$ended_at,bike_data$started_at,units =
c("mins"))
str(bike_data)
```

```
## tibble [5,595,063 x 22] (S3: tbl_df/tbl/data.frame)
##  $ ride_id           : chr [1:5595063] "E19E6F1B8D4C42ED" "DC88F20C2C55F27F"
"EC45C94683FE3F27" "4FA453A75AE377DB" ...
##  $ rideable_type     : chr [1:5595063] "electric_bike" "electric_bike" "elect
ric_bike" "electric_bike" ...
##  $ started_at        : POSIXct[1:5595063], format: "2021-01-23 16:14:19" "202
1-01-27 18:43:08" ...
##  $ ended_at          : POSIXct[1:5595063], format: "2021-01-23 16:24:44" "202
1-01-27 18:47:12" ...
##  $ start_station_name: chr [1:5595063] "California Ave & Cortez St" "Californ
ia Ave & Cortez St" "California Ave & Cortez St" "California Ave & Cortez St" ..
.
##  $ start_station_id  : chr [1:5595063] "17660" "17660" "17660" "17660" ...
##  $ end_station_name  : chr [1:5595063] NA NA NA NA ...
##  $ end_station_id    : chr [1:5595063] NA NA NA NA ...
##  $ start_lat         : num [1:5595063] 41.9 41.9 41.9 41.9 41.9 ...
##  $ start_lng         : num [1:5595063] -87.7 -87.7 -87.7 -87.7 -87.7 ...
##  $ end_lat           : num [1:5595063] 41.9 41.9 41.9 41.9 41.9 ...
##  $ end_lng           : num [1:5595063] -87.7 -87.7 -87.7 -87.7 -87.7 ...
##  $ member_casual     : chr [1:5595063] "member" "member" "member" "member" ..
```

```
.
##  $ date              : Date[1:5595063], format: "2021-01-23" "2021-01-27" ...
##  $ month             : chr [1:5595063] "01" "01" "01" "01" ...
##  $ day               : chr [1:5595063] "23" "27" "21" "07" ...
##  $ year              : chr [1:5595063] "2021" "2021" "2021" "2021" ...
##  $ day_of_week       : chr [1:5595063] "суббота" "среда" "четверг" "четверг"
...
##  $ start_hour        : chr [1:5595063] "22" "00" "04" "19" ...
##  $ end_hour          : chr [1:5595063] "22" "00" "04" "19" ...
##  $ hour_length       : 'difftime' num [1:5595063] 0.173611111111111 0.0677777
777777778 0.0222222222222222 0.195 ...
##   ..- attr(*, "units")= chr "hours"
##  $ min_length        : 'difftime' num [1:5595063] 10.4166666666667 4.06666666
666667 1.33333333333333 11.7 ...
##   ..- attr(*, "units")= chr "mins"
```

Date columns are made in chr format, but ride length is made in difftime format

## Remove “bad” data

The dataframe includes a few hundred entries when bikes were taken out of docks and checked for quality by or ride_length was negative We will create a new version of the dataframe (v2) since data is being removed

```
bike_data1 <- bike_data  %>% filter(min_length > 0) %>% drop_na()

print(paste("Removed", nrow(bike_data1) - nrow(bike_data),
            "rows that are either NA's or where ride length is less than 60 seco
nds"))

## [1] "Removed -614151 rows that are either NA's or where ride length is less t
han 60 seconds"
```

### CONDUCT DESCRIPTIVE ANALYSIS

In this step we will analyze our processed or cleaned data.

```
summary(bike_data1)

##    ride_id          rideable_type        started_at
##  Length:4980912     Length:4980912     Min.   :2021-01-01 00:02:24
##  Class :character   Class :character   1st Qu.:2021-06-09 06:33:43
##  Mode  :character   Mode  :character   Median :2021-07-31 07:47:13
##                                        Mean   :2021-07-26 22:12:16
##                                        3rd Qu.:2021-09-17 02:42:56
##                                        Max.   :2021-12-31 23:59:48
##     ended_at                  start_station_name start_station_id
##  Min.   :2021-01-01 00:08:39   Length:4980912     Length:4980912
##  1st Qu.:2021-06-09 06:53:06   Class :character   Class :character
##  Median :2021-07-31 08:16:22   Mode  :character   Mode  :character
##  Mean   :2021-07-26 22:33:50
##  3rd Qu.:2021-09-17 02:59:18
##  Max.   :2022-01-03 17:32:18                                          
##  end_station_name   end_station_id       start_lat       start_lng     
##  Length:4980912     Length:4980912     Min.   :41.65   Min.   :-87.84  
##  Class :character   Class :character   1st Qu.:41.88   1st Qu.:-87.66  
##  Mode  :character   Mode  :character   Median :41.90   Median :-87.64  
##                                        Mean   :41.90   Mean   :-87.64  
##                                        3rd Qu.:41.93   3rd Qu.:-87.63  
##                                        Max.   :42.07   Max.   :-87.52  
##     end_lat         end_lng       member_casual            date           
##  Min.   :41.57   Min.   :-87.87   Length:4980912     Min.   :2021-01-01  
##  1st Qu.:41.88   1st Qu.:-87.66   Class :character   1st Qu.:2021-06-09  
##  Median :41.90   Median :-87.64   Mode  :character   Median :2021-07-31  
##  Mean   :41.90   Mean   :-87.64                      Mean   :2021-07-26  
##  3rd Qu.:41.93   3rd Qu.:-87.63                      3rd Qu.:2021-09-17  
##  Max.   :42.17   Max.   :-87.49                      Max.   :2021-12-31  
##     month               day                year            day_of_week       
##  Length:4980912     Length:4980912     Length:4980912     Length:4980912    
##  Class :character   Class :character   Class :character   Class :character  
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character  
##                                                                            
##                                                                            
##                                                                            
##   start_hour          end_hour          hour_length        min_length      
##  Length:4980912     Length:4980912     Length:4980912    Length:4980912    
##  Class :character   Class :character   Class :difftime   Class :difftime  
##  Mode  :character   Mode  :character   Mode  :numeric    Mode  :numeric   
##                                                                            
##                                                                            
## 
```

```
mean(bike_data1$min_length) #straight average (total ride length / rides)
```

```
## Time difference of 21.56625 mins
```

```
median(bike_data1$min_length) #midpoint number in the ascending array of ride le
ngths
```

```
## Time difference of 12.23333 mins
```

```
max(bike_data1$min_length) #longest ride
```

```
## Time difference of 55944.15 mins
```

```
min(bike_data1$min_length) #shortest ride
```

```
## Time difference of 0.01666667 mins
```

But this data needs to be checked for outliers

## Day of Week

See the average ride time by each day for members vs casual users

```
bike_data1 <- bike_data1 %>%      #renaming from russian to english values in da
ys column
```

```
  mutate(day_of_week =recode(day_of_week
                             ,"воскресенье" = "sunday"
                             ,"вторник" = "tuesday"
                             ,"понедельник" = "monday"
                             ,"пятница" = "friday"
                             ,"среда" = "wednesday"
                             ,"суббота" = "saturday"
                             ,"четверг" = "thursday"))
```

```
aggregate(bike_data1$min_length~bike_data1$member_casual+bike_data1$day_of_week,
FUN = mean)
```

```
##    bike_data1$member_casual bike_data1$day_of_week bike_data1$min_length
## 1                    casual                 friday         29.57035 mins
## 2                    member                 friday         13.00604 mins
## 3                    casual                 monday         31.27836 mins
## 4                    member                 monday         12.85887 mins
## 5                    casual               saturday         34.04025 mins
## 6                    member               saturday         14.95688 mins
## 7                    casual                 sunday         37.07130 mins
## 8                    member                 sunday         15.26209 mins
## 9                    casual               thursday         27.28032 mins
## 10                   member               thursday         12.52844 mins
## 11                   casual                tuesday         27.97978 mins
## 12                   member                tuesday         12.51782 mins
## 13                   casual              wednesday         27.18089 mins
## 14                   member              wednesday         12.60128 mins
```

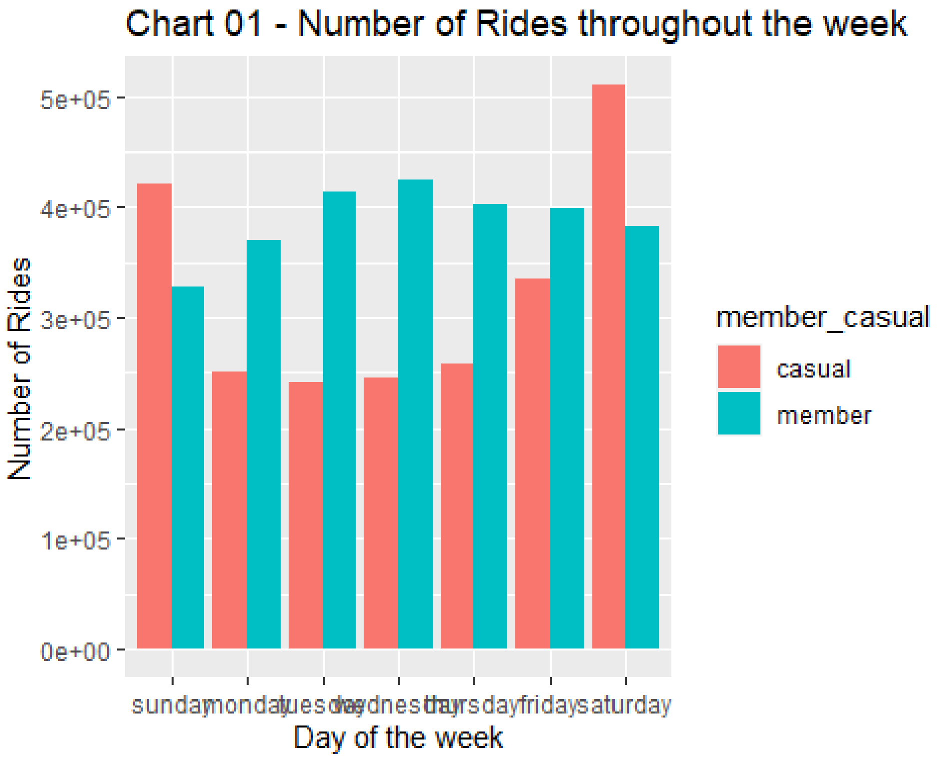
### next we will order data

```
bike_data1$day_of_week <-
  ordered(bike_data1$day_of_week, levels=c("sunday","monday","tuesday","wednesda
y","thursday","friday","saturday"))
```

### Analyze ridership data by type and weekday

```
bike_data1 %>%
  group_by(member_casual,day_of_week) %>% #group by weekday and customers
  summarise(number_of_rides = n(),average_duration = mean(min_length),) %>%
  arrange(member_casual,day_of_week) %>%
  ggplot(aes(x=day_of_week, y=number_of_rides, fill = member_casual ))+
  labs(x= "Day of the week", y ="Number of Rides", title = "Chart 01 - Number of
Rides throughout the week")+
  geom_col(position = "dodge")
```

```
## `summarise()` has grouped output by 'member_casual'. You can override using t
he `.groups` argument.
```



## We can clearly see that members usually dominate the weekdays, but on weekends casuals take more rides * Saturday has the biggest data points.

## Month data

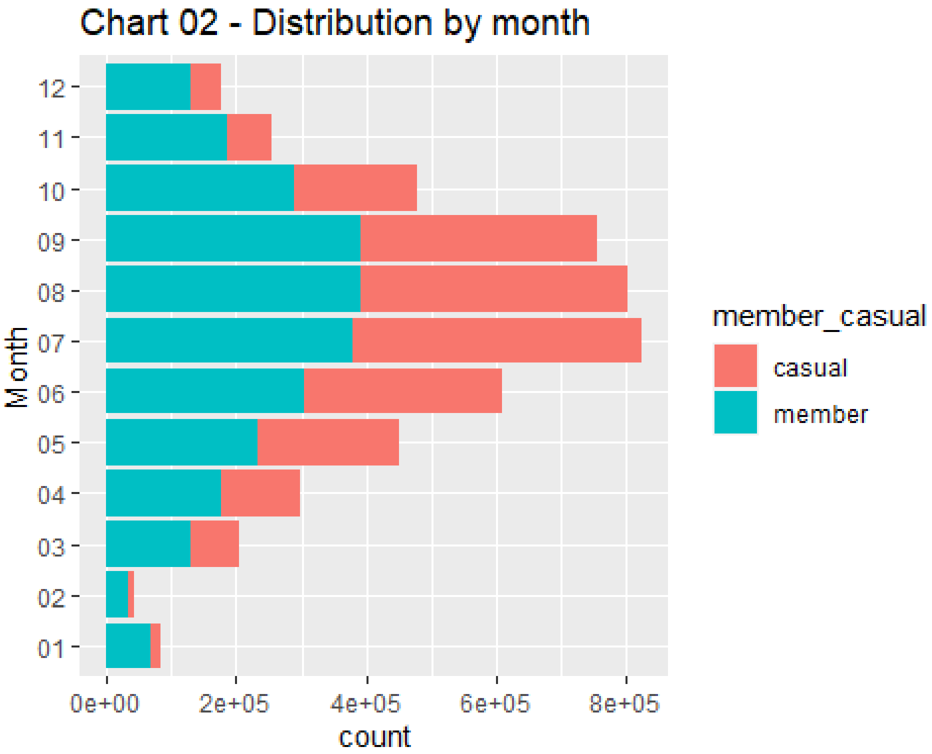
We grouped columns by month to see how much of members or casuals in any given month

```
bike_data1 %>%
  group_by(month) %>%
  summarise(count = length(ride_id),
            '%' = (length(ride_id) / nrow(bike_data1)) * 100,
            'members_p' = (sum(member_casual == "member") / length(ride_id)) * 1
00,
            'casual_p' = (sum(member_casual == "casual") / length(ride_id)) * 10
0,
            'Member x Casual Perc Difer' = members_p - casual_p)
```

```
## # A tibble: 12 x 6
##    month  count    `%` members_p casual_p `Member x Casual Perc Difer`
##    <chr>  <int>  <dbl>     <dbl>    <dbl>                        <dbl>
##  1 01     83508  1.68       82.4     17.6                        64.8
##  2 02     42994  0.863      80.0     20.0                        59.9
##  3 03    205687  4.13       63.2     36.8                        26.5
##  4 04    298199  5.99       59.6     40.4                        19.2
##  5 05    450978  9.05       51.9     48.1                         3.84
##  6 06    609597 12.2        50.0     50.0                         0.0441
##  7 07    823757 16.5        46.2     53.8                        -7.57
##  8 08    801191 16.1        48.8     51.2                        -2.42
##  9 09    754799 15.2        51.9     48.1                         3.79
## 10 10     477966  9.60        60.4      39.6                         20.9
## 11 11     255867  5.14        72.7      27.3                         45.3
## 12 12     176369  3.54        74.4      25.6                         48.9
```

### viz of month data

```
bike_data1 %>%
  ggplot(aes(month, fill=member_casual)) +
  geom_bar() +
  labs(x="Month", title="Chart 02 - Distribution by month") +
  coord_flip()
```



Some considerations can be taken by this chart: * The months with the biggest count of data points was August and July with ~25% of data * Almost n all months we have more members' rides than casual rides (Maybe because of returning members). * Temperature heavily influence the volume of rides in the month.

### Hour of the day

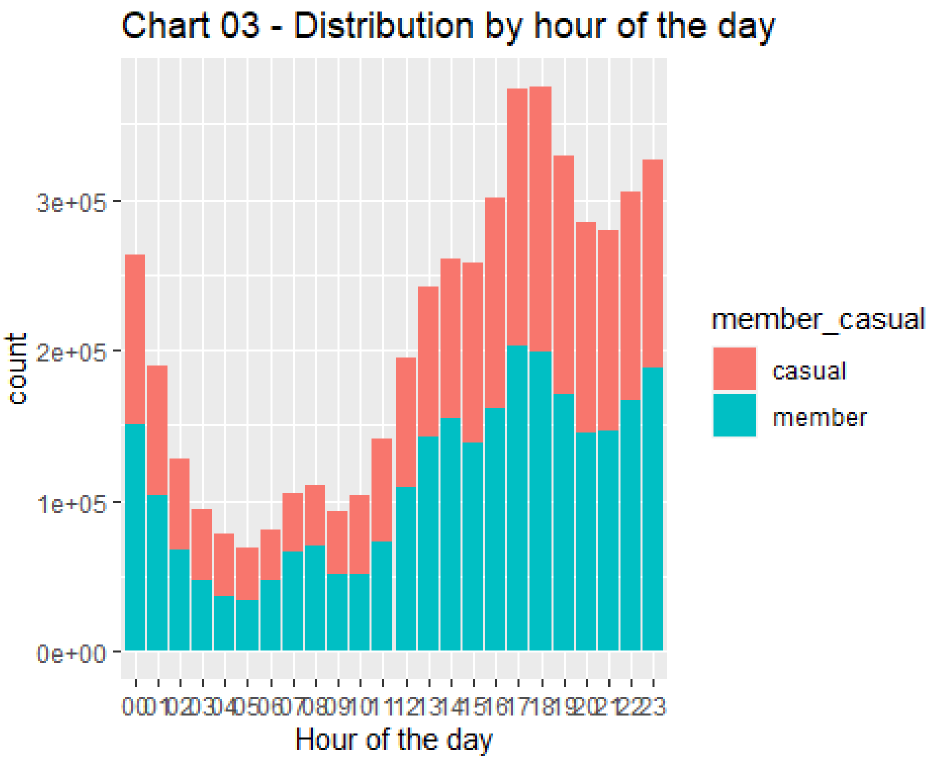
In this part will find what's the percentage of customers in any given hour

```
bike_data1 %>%
  group_by(start_hour) %>%
  summarise(count = length(ride_id),
            '%' = (length(ride_id) / nrow(bike_data1)) * 100,
            'members_p' = (sum(member_casual == "member") / length(ride_id)) * 1
00,
            'casual_p' = (sum(member_casual == "casual") / length(ride_id)) * 10
0,
            'member_casual_perc_difer' = members_p - casual_p)
```

```
## # A tibble: 24 x 6
##    start_hour  count   `%` members_p casual_p member_casual_perc_difer
##    <chr>       <int> <dbl>     <dbl>    <dbl>                    <dbl>
##  1 00         263886  5.30      57.1     42.9                   14.1
##  2 01         189214  3.80      55.0     45.0                    9.92
##  3 02         127913  2.57      52.4     47.6                    4.81
##  4 03          94048  1.89      49.9     50.1                   -0.196
##  5 04          77547  1.56      46.7     53.3                   -6.50
##  6 05          68456  1.37      49.7     50.3                   -0.666
##  7 06          81071  1.63      57.6     42.4                   15.3
##  8 07         104666  2.10      63.3     36.7                   26.6
##  9 08         110279  2.21      63.6     36.4                   27.2
## 10 09          92462  1.86      55.4     44.6                   10.8
## # ... with 14 more rows
```

### viz of hour data

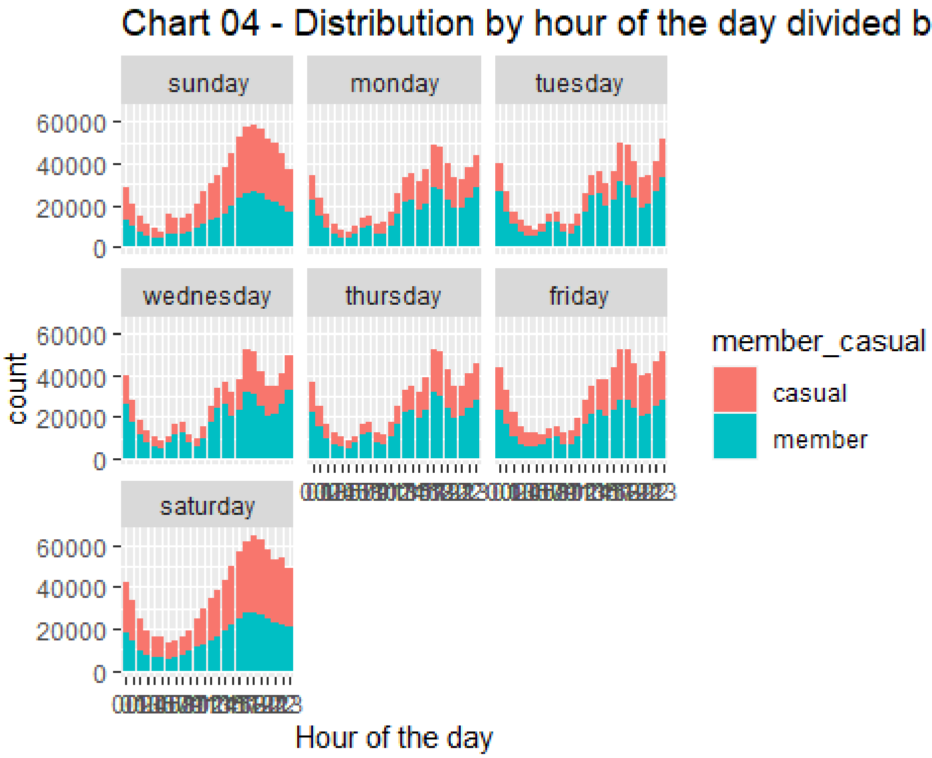
```
bike_data1 %>%
  ggplot(aes(start_hour, fill=member_casual)) +
  labs(x="Hour of the day", title="Chart 03 - Distribution by hour of the day")
+
  geom_bar()
```



## From this chart, we can see: * There's a bigger volume of bikers in the evening around 17-19 * We have more members during the morning, mainly in between 5am and 11am * And more casuals between 16pm 19pm

This chart can be expanded lets see it divided by day of the week.

```
bike_data1 %>%
  ggplot(aes(start_hour, fill=member_casual)) +
  geom_bar() +
  labs(x="Hour of the day", title="Chart 04 - Distribution by hour of the day di
vided by weekday") +
  facet_wrap(~ day_of_week)
```



### We can clearly see that members tend to use the bike during working days

## Rideable type

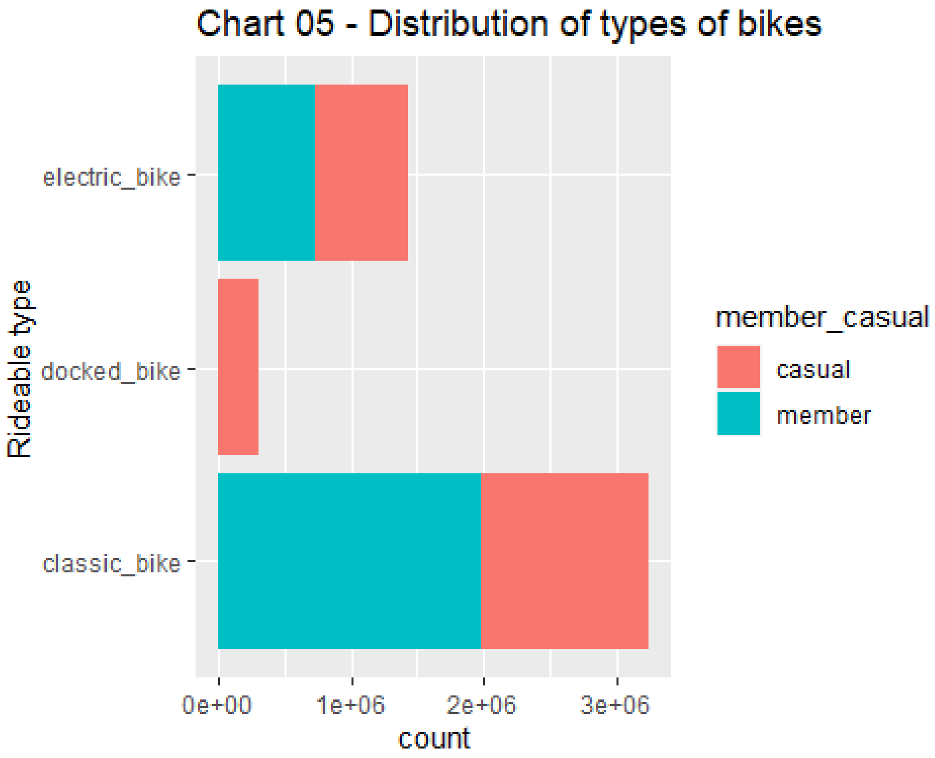
Now, we will focus on what type of bike do customers prefer

```
bike_data1 %>%
  group_by(rideable_type) %>%
  summarise(count = length(ride_id),
            '%' = (length(ride_id) / nrow(bike_data1)) * 100,
            'members_p' = (sum(member_casual == "member") / length(ride_id)) * 1
00,
            'casual_p' = (sum(member_casual == "casual") / length(ride_id)) * 10
0,
            'member_casual_perc_difer' = members_p - casual_p)

## # A tibble: 3 x 6
##   rideable_type   count   `%` members_p casual_p member_casual_perc_difer
##   <chr>           <int> <dbl>     <dbl>    <dbl>                    <dbl>
## 1 classic_bike  3243526 65.1   61.1         38.9                     22.2
## 2 docked_bike    312041  6.26   0.000320   100.                    -100.
## 3 electric_bike 1425345 28.6   51.8         48.2                      3.60
```

### Let’s viz

```
ggplot(bike_data1, aes(rideable_type, fill=member_casual)) +
  labs(x="Rideable type", title="Chart 05 - Distribution of types of bikes") +
  geom_bar() +
  coord_flip()
```



### It’s important to note that: * Classic bikes have the biggest volume of rides, but this can be that the company may have more docked bikes. * Also for electric bikes.

## Ride time

Lastly we will focus on ride time

```
min(bike_data1$min_length)

## Time difference of 0.01666667 mins

max(bike_data1$min_length)

## Time difference of 55944.15 mins
```

The min and the max may be a problem to plot some charts.Therefore we will find outliers

```
ventiles = quantile(bike_data1$min_length, seq(0, 1, by=0.05))
ventiles

## Time differences in mins
##           0%           5%          10%          15%          20%          25% 
## 1.666667e-02 2.916667e+00 4.133333e+00 5.100000e+00 6.016667e+00 6.933333e+00 
##          30%          35%          40%          45%          50%          55% 
## 7.866667e+00 8.850000e+00 9.883333e+00 1.100000e+01 1.223333e+01 1.361667e+01
##          60%          65%          70%          75%          80%          85% 
## 1.521667e+01 1.708333e+01 1.933333e+01 2.211667e+01 2.568333e+01 3.048333e+01 
##          90%          95%         100% 
## 3.833333e+01 5.706667e+01 5.594415e+04
```

We can see that: * The difference between 0% and 100% is 55944.13 mins * The difference between 5% and 95% is -54.15 mins. Because of that, in the analysis of this variable we are going to use a subset of the dataset without outliners. The subset will contain 95% of the dataset.

```
bike_data1_noout <- bike_data1 %>%
  filter(min_length > as.numeric(ventiles['5%'])) %>%
  filter(min_length < as.numeric(ventiles['95%']))

print(paste("Removed", nrow(bike_data1) - nrow(bike_data1_noout), "rows as outli
ners" ))
```

```
## [1] "Removed 498795 rows as outliners"
```

### Ride time closer look

```
bike_data1_noout %>%
  group_by(member_casual) %>%
  summarise(mean = mean(min_length),
            'first_quarter' = as.numeric(quantile(min_length, .25)),
            'median' = median(min_length),
            'third_quarter' = as.numeric(quantile(min_length, .75)),
            'IR' = third_quarter - first_quarter)
```

```
## # A tibble: 2 x 6
##   member_casual mean          first_quarter median        third_quarter    IR
##   <chr>         <drtn>                <dbl> <drtn>                <dbl> <dbl>
## 1 casual        18.62580 mins          9.27 15.18333 mins          25.1  15.8
## 2 member        13.27149 mins          6.42 10.35000 mins          17.1  10.7
```

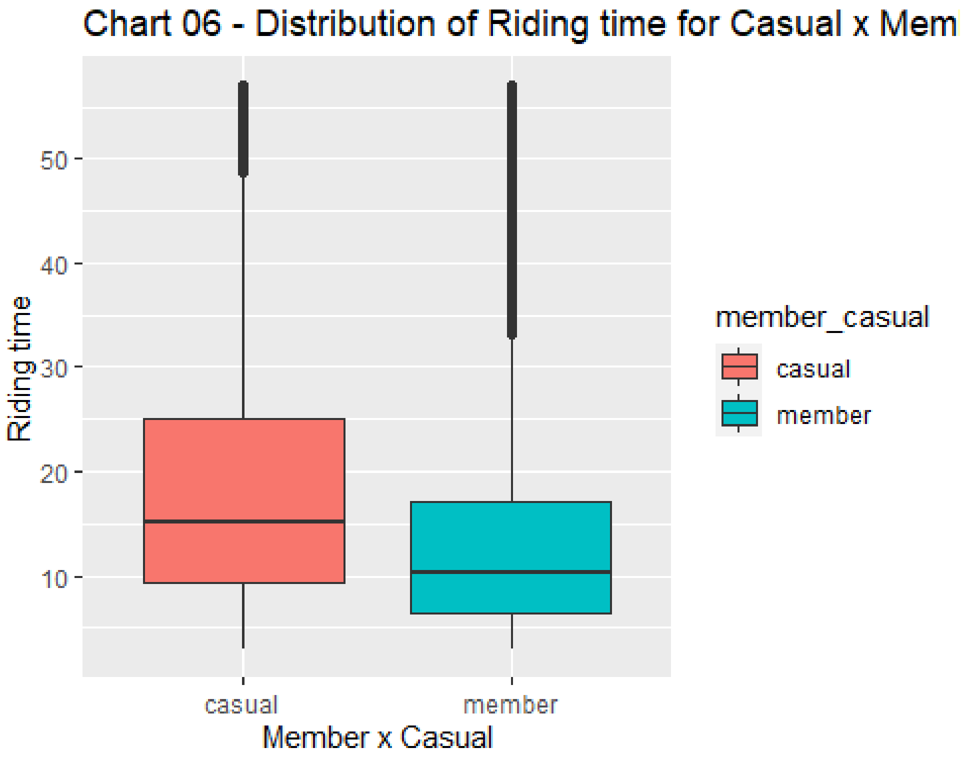
It’s important to note that: * Casual have more riding time thant members. * Mean and IQR is also bigger for casual.

Let’s see if we can extract more informations when ploting

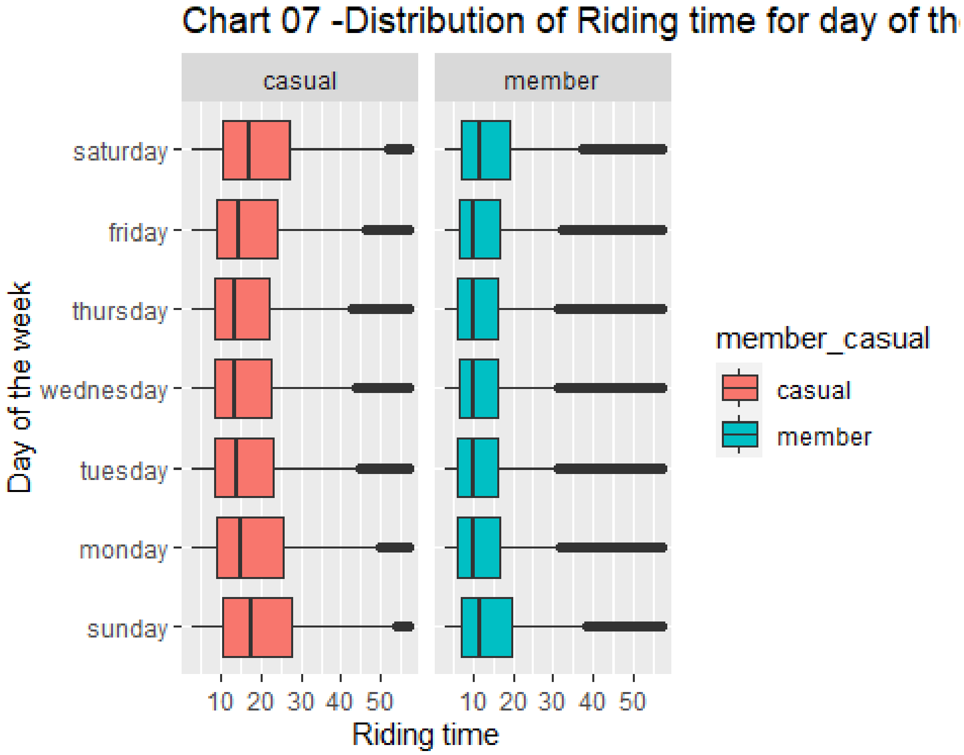
### Distribution of Riding time for Casual x Member

```
ggplot(bike_data1_noout, aes(x=member_casual, y=min_length, fill=member_casual))
+
  labs(x="Member x Casual", y="Riding time", title="Chart 06 - Distribution of R
iding time for Casual x Member") +
  geom_boxplot()
```

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



## Distribution of Riding time for day of the week



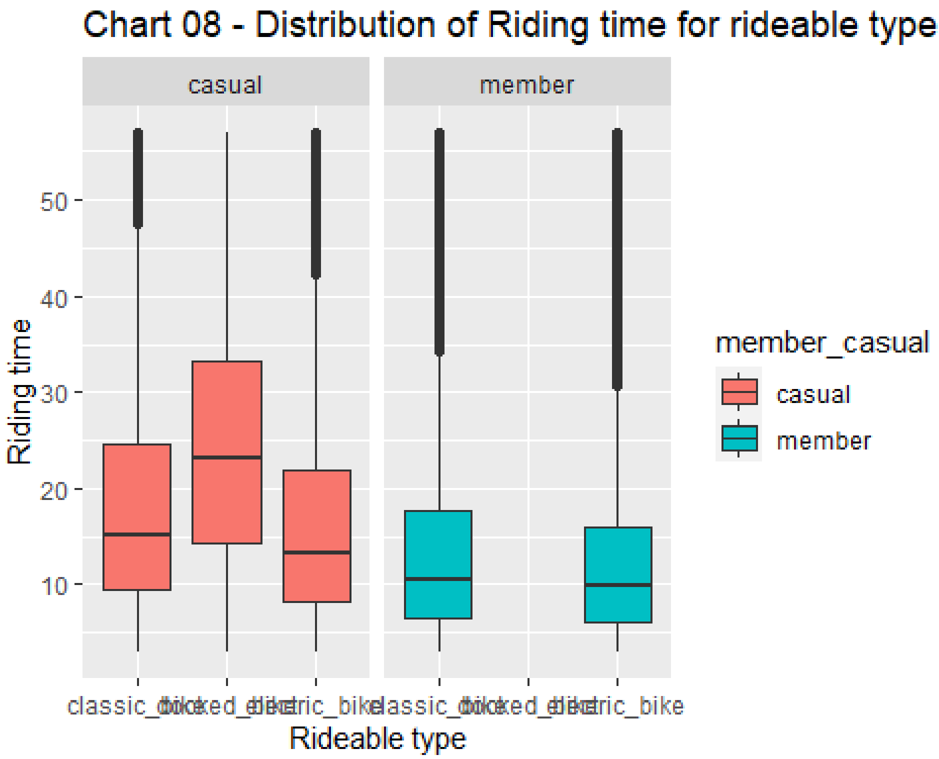
* Riding time for members is stable during the midweek, increasing during weekends * Casuals peak on sundays and slightly increasing on wednesday/thursday.

Last plot will cover distribution of time by type of a bicycle

### Distribution of Riding time for rideable type

```
ggplot(bike_data1_noout, aes(x=rideable_type, y=min_length,fill=member_casual))+
  geom_boxplot()+
  facet_wrap(~member_casual) +
  labs(x="Rideable type", y = "Riding time", title = "Chart 08 - Distribution of
Riding time for rideable type")

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



Electric bikes have less riding time than other bikes, for both members and casuals.

- Docked bikes have more riding time. And for docked bikes, members have more riding time while casuals don't.

## Deliverable

After tons of codes and analysis, it's time to share our results and to answer the question "How can we convert casuals to members?"

Our data has one limitation, it doesnt show us how and why a certain rider decided to become a member.

However, we made some observations and can suggest few things One of our observations was that casuals peak on sundays and also on wednesday/thursday. This data combined with the knowledge that August and July are our most profitable months, and that Casual riders prefer 16pm - 19pm. We could develop a packeage that can pique their interest. For instance, membership rides between 16-19 pm is few dollars cheaper, or make a promotion on August and July. Another, way to approach this is to make an incentive for casuals to purchase memberships through offering memberships based around the time they drive. For instance, on average casuals drive for around 18.62 min when members only drive for 13.27 min. Therefore, introducing a package where first 13 min of a ride is free can potetntially lure new customers