Data analysis

María del Carmen Martín Rodríguez

2025-01-13

Divvy Exercise Full Year Analysis

This analysis is based on the Divvy case study " 'Sophisticated, Clear, and Polished': Divvy and Data Visualization" written by Kevin Hartman found in https://artscience.blog/home/divvy-dataviz-case-study. The purpose of this script 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?"

Install required packages

- tidyverse for data import and wrangling
- lubridate for date functions
- ggplot for visualization
- dplur for data manipulation

```
library(tidyverse)  #helps wrangle data
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v readr     2.1.5
## v forcats   1.0.0     v stringr   1.5.1
## v ggplot2   3.5.1     v tibble    3.2.1
## v lubridate 1.9.4     v tidyr     1.3.1
## v purrr     1.0.2
## -- Conflicts ---------------------------------------- tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```
library(lubridate)  #helps wrangle date attributes
library(ggplot2)  #helps visualize data
library(dplyr) # helps Data Manipulation
```

STEP 1: COLLECT DATA

Data was collected from this site

```r
setwd("/Users/carme/Desktop/Datos_divvy/CSV") #sets your working directory to simplify calls to data ..
# Upload Divvy datasets (csv files) here
jan <- read_csv("202401-divvy-tripdata.csv")
feb <- read_csv("202402-divvy-tripdata.csv")
mar <- read_csv("202403-divvy-tripdata.csv")
apr <- read_csv("202404-divvy-tripdata.csv")
may <- read_csv("202405-divvy-tripdata.csv")
jun <- read_csv("202406-divvy-tripdata.csv")
jul <- read_csv("202407-divvy-tripdata.csv")
aug <- read_csv("202408-divvy-tripdata.csv")
sep <- read_csv("202409-divvy-tripdata.csv")
oct <- read_csv("202410-divvy-tripdata.csv")
nov <- read_csv("202411-divvy-tripdata.csv")
dec <- read_csv("202412-divvy-tripdata.csv")
```

STEP 2: COMBINE DATA INTO A SINGLE DATAFRAME

Stack individual data frames into one big data frame:

```r
all_trips <- bind_rows(jan, feb, mar, apr, may, jun, jul, aug, sep, oct, nov, dec)
```

Inspect the new table that has been created:

```r
colnames(all_trips)  #List of column names
```

```
##  [1] "ride_id"            "rideable_type"      "started_at"
##  [4] "ended_at"           "start_station_name" "start_station_id"
##  [7] "end_station_name"   "end_station_id"     "start_lat"
## [10] "start_lng"          "end_lat"            "end_lng"
## [13] "member_casual"
```

```r
nrow(all_trips)  #How many rows are in data frame?
```

```
## [1] 5860568
```

```r
dim(all_trips)  #Dimensions of the data frame?
```

```
## [1] 5860568      13
```

```r
head(all_trips)  #See the first 6 rows of data frame.  Also tail(all_trips)
```

```
## # A tibble: 6 x 13
##   ride_id          rideable_type started_at          ended_at
##   <chr>            <chr>         <dttm>              <dttm>
## 1 C1D650626C8C899A electric_bike 2024-01-12 15:30:27 2024-01-12 15:37:59
## 2 EECD38BDB25BFCB0 electric_bike 2024-01-08 15:45:46 2024-01-08 15:52:59
## 3 F4A9CE78061F17F7 electric_bike 2024-01-27 12:27:19 2024-01-27 12:35:19
## 4 0A0D9E15EE50B171 classic_bike  2024-01-29 16:26:17 2024-01-29 16:56:06
## 5 33FFC9805E3EFF9A classic_bike  2024-01-31 05:43:23 2024-01-31 06:09:35
```

```
## 6 C96080812CD285C5 classic_bike  2024-01-07 11:21:24 2024-01-07 11:30:03
## # i 9 more variables: start_station_name <chr>, start_station_id <chr>,
## #   end_station_name <chr>, end_station_id <chr>, start_lat <dbl>,
## #   start_lng <dbl>, end_lat <dbl>, end_lng <dbl>, member_casual <chr>
```

```
str(all_trips)  #See list of columns and data types (numeric, character, etc)
```

```
## spc_tbl_ [5,860,568 x 13] (S3: spec_tbl_df/tbl_df/tbl/data.frame)
##  $ ride_id           : chr [1:5860568] "C1D650626C8C899A" "EECD38BDB25BFCB0" "F4A9CE78061F17F7" "0A0
##  $ rideable_type     : chr [1:5860568] "electric_bike" "electric_bike" "electric_bike" "classic_bike
##  $ started_at        : POSIXct[1:5860568], format: "2024-01-12 15:30:27" "2024-01-08 15:45:46" ...
##  $ ended_at          : POSIXct[1:5860568], format: "2024-01-12 15:37:59" "2024-01-08 15:52:59" ...
##  $ start_station_name: chr [1:5860568] "Wells St & Elm St" "Wells St & Elm St" "Wells St & Elm St" "
##  $ start_station_id  : chr [1:5860568] "KA1504000135" "KA1504000135" "KA1504000135" "TA1305000030" .
##  $ end_station_name  : chr [1:5860568] "Kingsbury St & Kinzie St" "Kingsbury St & Kinzie St" "Kingsbu
##  $ end_station_id    : chr [1:5860568] "KA1503000043" "KA1503000043" "KA1503000043" "13193" ...
##  $ start_lat         : num [1:5860568] 41.9 41.9 41.9 41.9 41.9 ...
##  $ start_lng         : num [1:5860568] -87.6 -87.6 -87.6 -87.6 -87.7 ...
##  $ end_lat           : num [1:5860568] 41.9 41.9 41.9 41.9 41.9 ...
##  $ end_lng           : num [1:5860568] -87.6 -87.6 -87.6 -87.6 -87.6 ...
##  $ member_casual     : chr [1:5860568] "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>
```

```
summary(all_trips)  #Statistical summary of data. Mainly for numerics
```

```
##    ride_id          rideable_type        started_at
##  Length:5860568     Length:5860568     Min.   :2024-01-01 00:00:39.00
##  Class :character   Class :character   1st Qu.:2024-05-20 19:47:53.00
##  Mode  :character   Mode  :character   Median :2024-07-22 20:36:16.27
##                                        Mean   :2024-07-17 07:55:47.61
##                                        3rd Qu.:2024-09-17 20:14:22.56
##                                        Max.   :2024-12-31 23:56:49.84
##
##     ended_at                      start_station_name start_station_id
##  Min.   :2024-01-01 00:04:20.00   Length:5860568     Length:5860568
##  1st Qu.:2024-05-20 20:07:54.75   Class :character   Class :character
##  Median :2024-07-22 20:53:59.16   Mode  :character   Mode  :character
```

```
##  Mean   :2024-07-17 08:13:06.54
##  3rd Qu.:2024-09-17 20:27:46.02
##  Max.   :2024-12-31 23:59:55.70
##
##  end_station_name   end_station_id       start_lat       start_lng
##  Length:5860568     Length:5860568     Min.   :41.64   Min.   :-87.91
##  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.   :16.06   Min.   :-144.05   Length:5860568
##  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.   :87.96   Max.   : 152.53
##  NA's   :7232    NA's   :7232
```

STEP 3: CLEAN UP AND ADD DATA TO PREPARE FOR ANALYSIS

Remove fields with location coordinates because they are irrelevant:

```
all_trips <- all_trips %>%
  select(-c(start_lat, start_lng, end_lat, end_lng))
```

Add columns that list the date, month, day, and year of each ride. This will allow us to aggregate ride data for each month, day, or year ... before completing these operations we could only aggregate at the ride level:

```
all_trips$date <- as.Date(all_trips$started_at) #The default format is yyyy-mm-dd
all_trips$month <- format(as.Date(all_trips$date), "%m")
all_trips$day <- format(as.Date(all_trips$date), "%d")
all_trips$year <- format(as.Date(all_trips$date), "%Y")
all_trips$day_of_week <- format(as.Date(all_trips$date), "%A")
```

Add a "ride_length" calculation to all_trips (in seconds):

```
all_trips$ride_length <- difftime(all_trips$ended_at,all_trips$started_at)
```

Inspect the structure of the columns:

```
str(all_trips)
```

```
## tibble [5,860,568 x 15] (S3: tbl_df/tbl/data.frame)
##  $ ride_id           : chr [1:5860568] "C1D650626C8C899A" "EECD38BDB25BFCB0" "F4A9CE78061F17F7" "0A0
##  $ rideable_type     : chr [1:5860568] "electric_bike" "electric_bike" "electric_bike" "classic_bike
##  $ started_at        : POSIXct[1:5860568], format: "2024-01-12 15:30:27" "2024-01-08 15:45:46" ...
##  $ ended_at          : POSIXct[1:5860568], format: "2024-01-12 15:37:59" "2024-01-08 15:52:59" ...
```

```
##  $ start_station_name: chr [1:5860568] "Wells St & Elm St" "Wells St & Elm St" "Wells St & Elm St" "
##  $ start_station_id  : chr [1:5860568] "KA1504000135" "KA1504000135" "KA1504000135" "TA1305000030" .
##  $ end_station_name  : chr [1:5860568] "Kingsbury St & Kinzie St" "Kingsbury St & Kinzie St" "Kingsb
##  $ end_station_id    : chr [1:5860568] "KA1503000043" "KA1503000043" "KA1503000043" "13193" ...
##  $ member_casual     : chr [1:5860568] "member" "member" "member" "member" ...
##  $ date              : Date[1:5860568], format: "2024-01-12" "2024-01-08" ...
##  $ month             : chr [1:5860568] "01" "01" "01" "01" ...
##  $ day               : chr [1:5860568] "12" "08" "27" "29" ...
##  $ year              : chr [1:5860568] "2024" "2024" "2024" "2024" ...
##  $ day_of_week       : chr [1:5860568] "viernes" "lunes" "sábado" "lunes" ...
##  $ ride_length       : 'difftime' num [1:5860568] 452 433 480 1789 ...
##   ..- attr(*, "units")= chr "secs"
```

Convert "ride_length" from Factor to numeric so we can run calculations on the data:

```
is.factor(all_trips$ride_length)
```

```
## [1] FALSE
```

```
all_trips$ride_length <- as.numeric(as.character(all_trips$ride_length))
is.numeric(all_trips$ride_length)
```

```
## [1] TRUE
```

Check that there are no duplicate rows:

```
all_trips <- unique(all_trips)
```

We will delete "bad" data. The data frame includes a few hundred entries where the walk time was a negative value or less than 60 seconds. We will create a new version of the data frame (v2) since data is being removed:

```
all_trips_v2 <- all_trips %>%
  filter(!is.na(ride_length) & ride_length >= 60)
```

Let's make the month column more coherent:

```
all_trips_v2 <- all_trips_v2 %>%
  mutate(month = recode(month,
                        "01" = "enero",
                        "02" = "febrero",
                        "03" = "marzo",
                        "04" = "abril",
                        "05" = "mayo",
                        "06" = "junio",
                        "07" = "julio",
                        "08" = "agosto",
                        "09" = "septiembre",
                        "10" = "octubre",
                        "11" = "noviembre",
                        "12" = "diciembre"))
```

STEP 4: CONDUCT DESCRIPTIVE ANALYSIS

Descriptive analysis on ride_length (all figures in seconds):

```
summary(all_trips_v2$ride_length)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    60.0   347.3   596.7  1062.3  1051.6 93596.0
```

Compare members and casual users:

```
aggregate(all_trips_v2$ride_length ~ all_trips_v2$member_casual, FUN = mean)
```

```
##   all_trips_v2$member_casual all_trips_v2$ride_length
## 1                     casual                1555.6221
## 2                     member                 779.8342
```

```
aggregate(all_trips_v2$ride_length ~ all_trips_v2$member_casual, FUN = median)
```

```
##   all_trips_v2$member_casual all_trips_v2$ride_length
## 1                     casual                   748.13
## 2                     member                   531.00
```

```
aggregate(all_trips_v2$ride_length ~ all_trips_v2$member_casual, FUN = max)
```

```
##   all_trips_v2$member_casual all_trips_v2$ride_length
## 1                     casual                    93596
## 2                     member                    93588
```

```
aggregate(all_trips_v2$ride_length ~ all_trips_v2$member_casual, FUN = min)
```

```
##   all_trips_v2$member_casual all_trips_v2$ride_length
## 1                     casual                       60
## 2                     member                       60
```

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

```
aggregate(all_trips_v2$ride_length ~ all_trips_v2$member_casual + all_trips_v2$day_of_week, FUN = mean)
```

```
##    all_trips_v2$member_casual all_trips_v2$day_of_week all_trips_v2$ride_length
## 1                      casual                 domingo                1822.3651
## 2                      member                 domingo                 872.9023
## 3                      casual                  jueves                1354.3948
## 4                      member                  jueves                 747.5375
## 5                      casual                   lunes                1489.0019
## 6                      member                   lunes                 744.5072
## 7                      casual                  martes                1326.5486
## 8                      member                  martes                 749.2442
## 9                      casual               miércoles                1376.3531
```

```
## 10                    member               miércoles                759.9032
## 11                    casual                  sábado               1748.3825
## 12                    member                  sábado                862.1790
## 13                    casual                 viernes               1517.2393
## 14                    member                 viernes                758.6460
```

Notice that the days of the week are out of order. Let's fix that:

```
all_trips_v2$day_of_week <- ordered(all_trips_v2$day_of_week, levels=c("domingo", "lunes", "martes", "mi
```

Now, let's run the average ride time by each day for members vs casual users:

```
aggregate(all_trips_v2$ride_length ~ all_trips_v2$member_casual + all_trips_v2$day_of_week, FUN = mean)
```

```
##    all_trips_v2$member_casual all_trips_v2$day_of_week all_trips_v2$ride_length
## 1                      casual                 domingo               1822.3651
## 2                      member                 domingo                872.9023
## 3                      casual                   lunes               1489.0019
## 4                      member                   lunes                744.5072
## 5                      casual                  martes               1326.5486
## 6                      member                  martes                749.2442
## 7                      casual                miércoles               1376.3531
## 8                      member                miércoles                759.9032
## 9                      casual                  jueves               1354.3948
## 10                     member                  jueves                747.5375
## 11                     casual                 viernes               1517.2393
## 12                     member                 viernes                758.6460
## 13                     casual                  sábado               1748.3825
## 14                     member                  sábado                862.1790
```

Analyze ridership data by type and weekday:

```
all_trips_v2 %>%
  mutate(weekday = wday(started_at, label = TRUE)) %>%
  group_by(member_casual, weekday) %>%
  summarise(number_of_rides = n(),average_duration = mean(ride_length)) %>%
  arrange(member_casual, weekday)
```

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

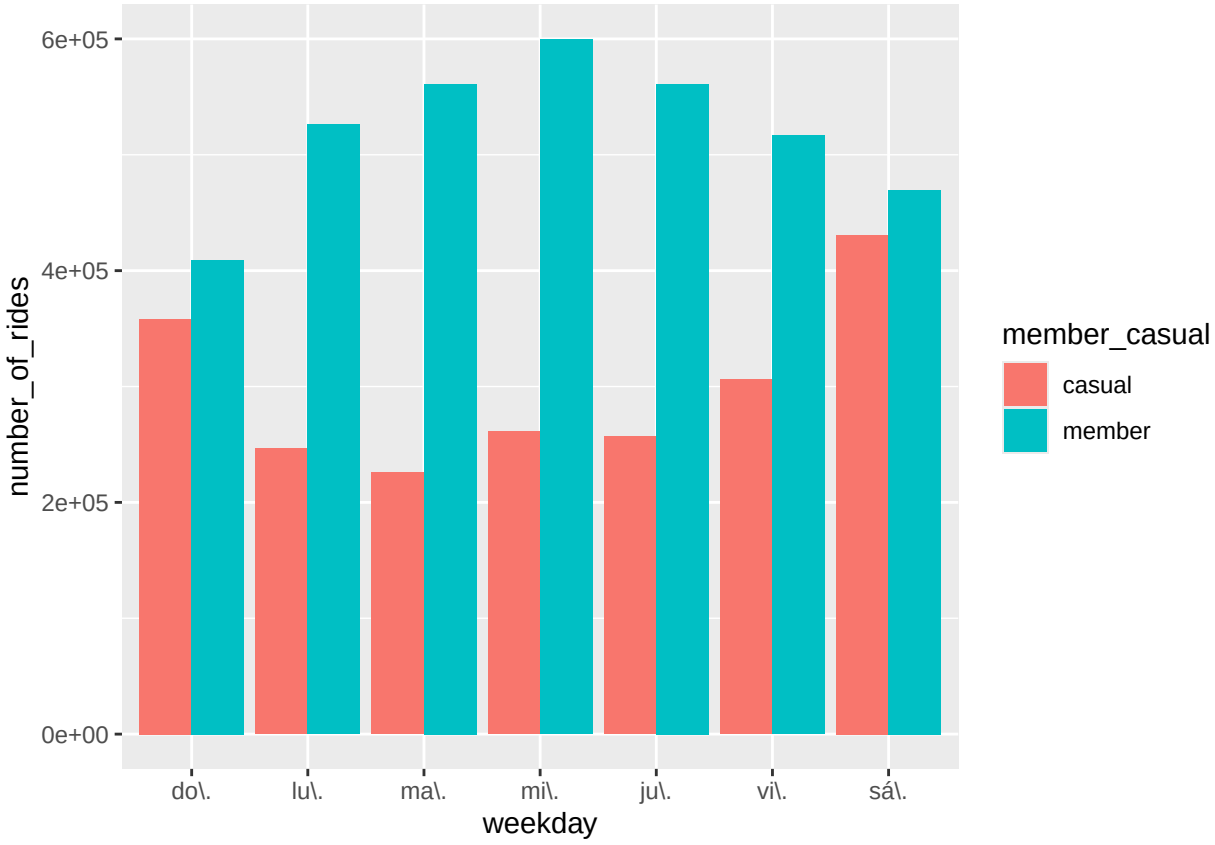
```
## # A tibble: 14 x 4
## # Groups:   member_casual [2]
##    member_casual weekday number_of_rides average_duration
##    <chr>         <ord>             <int>            <dbl>
##  1 casual        "do\\."          358189            1822.
##  2 casual        "lu\\."          246546            1489.
##  3 casual        "ma\\."          226028            1327.
##  4 casual        "mi\\."          261441            1376.
##  5 casual        "ju\\."          257153            1354.
##  6 casual        "vi\\."          306227            1517.
```

```
##  7 casual        "sá\\."          430721            1748.
##  8 member        "do\\."          409150             873.
##  9 member        "lu\\."          525798             745.
## 10 member        "ma\\."          561056             749.
## 11 member        "mi\\."          599649             760.
## 12 member        "ju\\."          561059             748.
## 13 member        "vi\\."          516387             759.
## 14 member        "sá\\."          469634             862.
```

Let's visualize the number of rides by rider type:

```
all_trips_v2 %>%
  mutate(weekday = wday(started_at, label = TRUE)) %>%
  group_by(member_casual, weekday) %>%
  summarise(number_of_rides = n() ,average_duration = mean(ride_length)) %>%
  arrange(member_casual, weekday)  %>%
  ggplot(aes(x = weekday, y = number_of_rides, fill = member_casual)) +
  geom_col(position = "dodge")
```

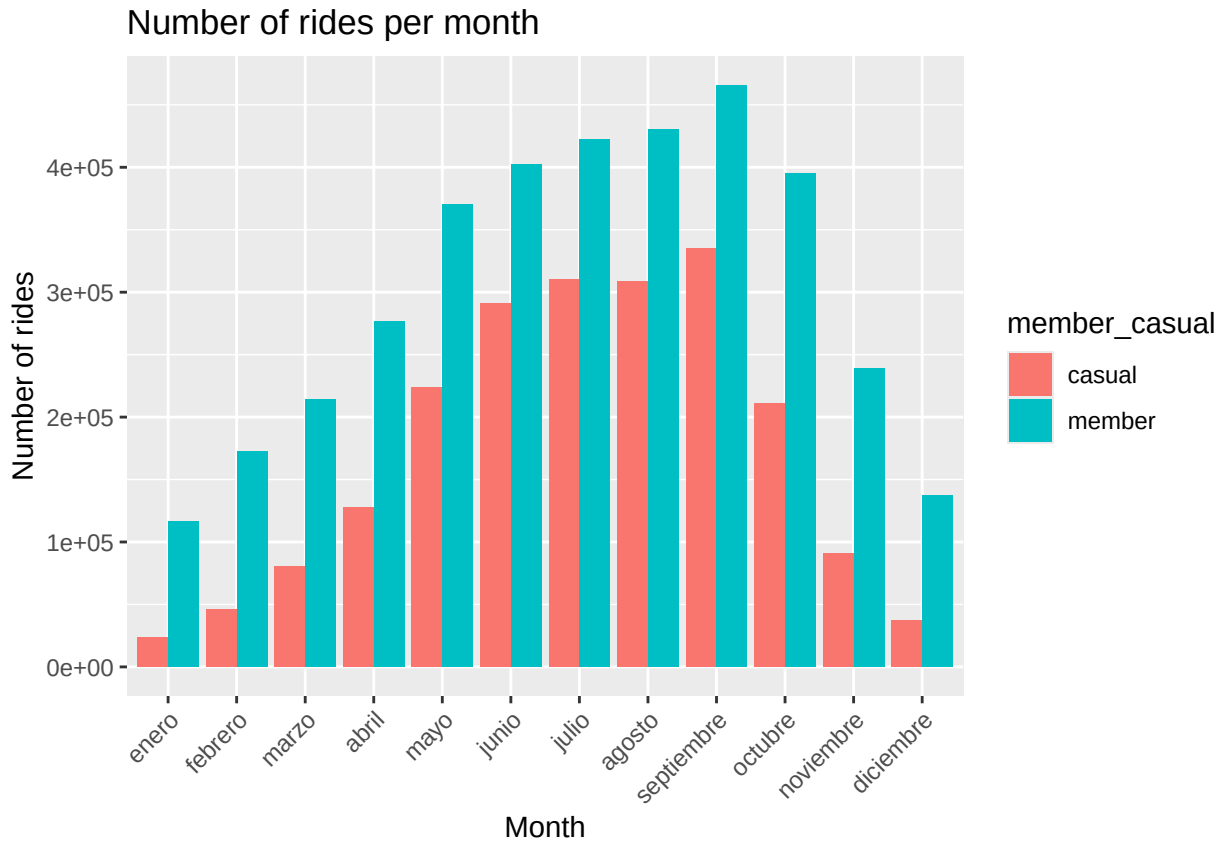
```
## `summarise()` has grouped output by 'member_casual'. You can override using the
## `.groups` argument.
```




```
all_trips_v2 %>%
  mutate(month = factor(month, levels = c("enero", "febrero", "marzo", "abril", "mayo", "junio", "julio
```

```
  group_by(member_casual, month) %>%
  summarise(number_of_rides = n(),
            average_duration = mean(ride_length, na.rm = TRUE)) %>%
  arrange(member_casual, month) %>%
  ggplot(aes(x = month, y = number_of_rides, fill = member_casual)) +
  geom_col(position = "dodge") +
  labs(x = "Month", y = "Number of rides", title = "Number of rides per month") +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```

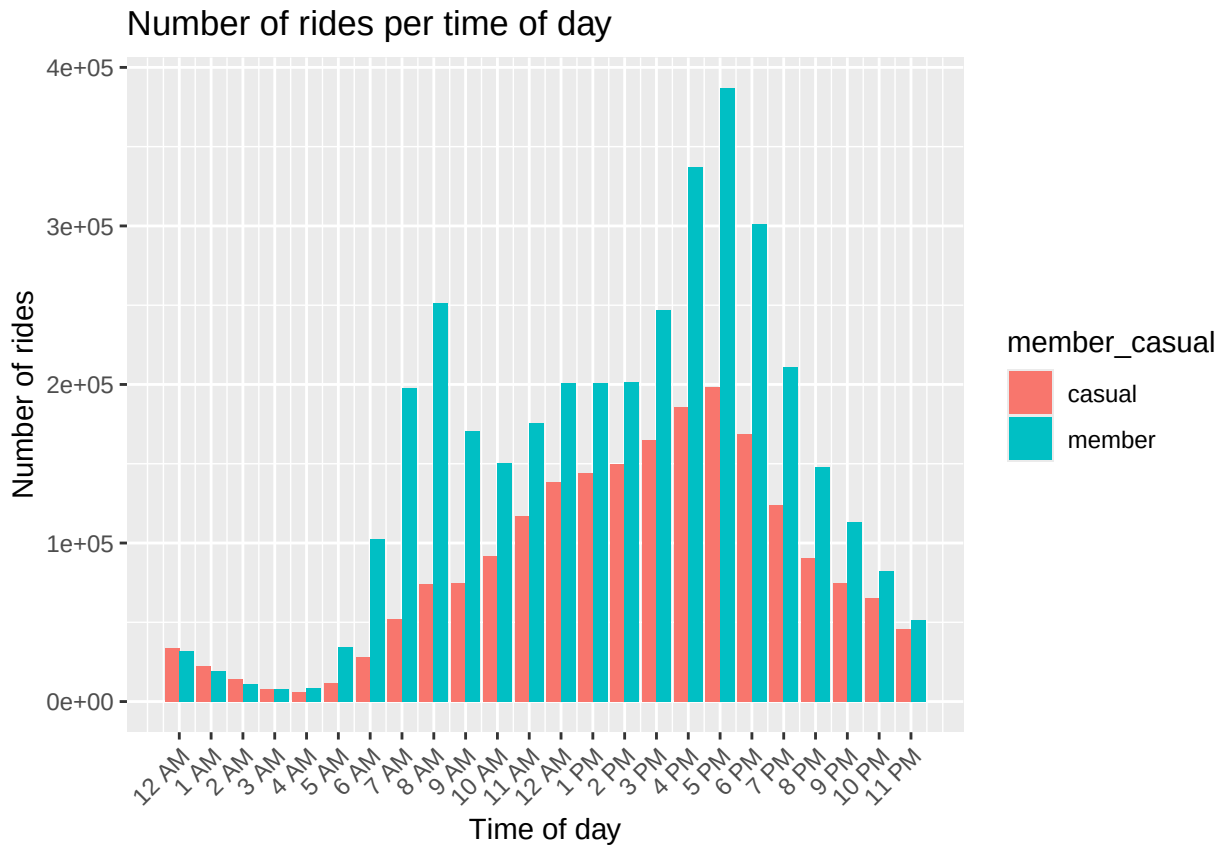
```
## `summarise()` has grouped output by 'member_casual'. You can override using the
## `.groups` argument.
```




```
all_trips_v2 %>%
  mutate(hour_of_day = hour(started_at)) %>%
  group_by(member_casual, hour_of_day) %>%
  summarise(number_of_rides = n(),
            average_duration = mean(ride_length, na.rm = TRUE)) %>%
  arrange(member_casual, hour_of_day) %>%
  ggplot(aes(x = hour_of_day, y = number_of_rides, fill = member_casual)) +
  geom_col(position = "dodge") +
  scale_x_continuous(breaks = 0:23,
                     labels = function(x) {
                       hour_labels <- ifelse(x %% 12 == 0, "12 AM",
                                             ifelse(x < 12, paste0(x, " AM"),
```

```
                                                  paste0(x - 12, " PM")))
                    return(hour_labels)
                  }) +
  labs(x = "Time of day", y = "Number of rides", title = "Number of rides per time of day") +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```

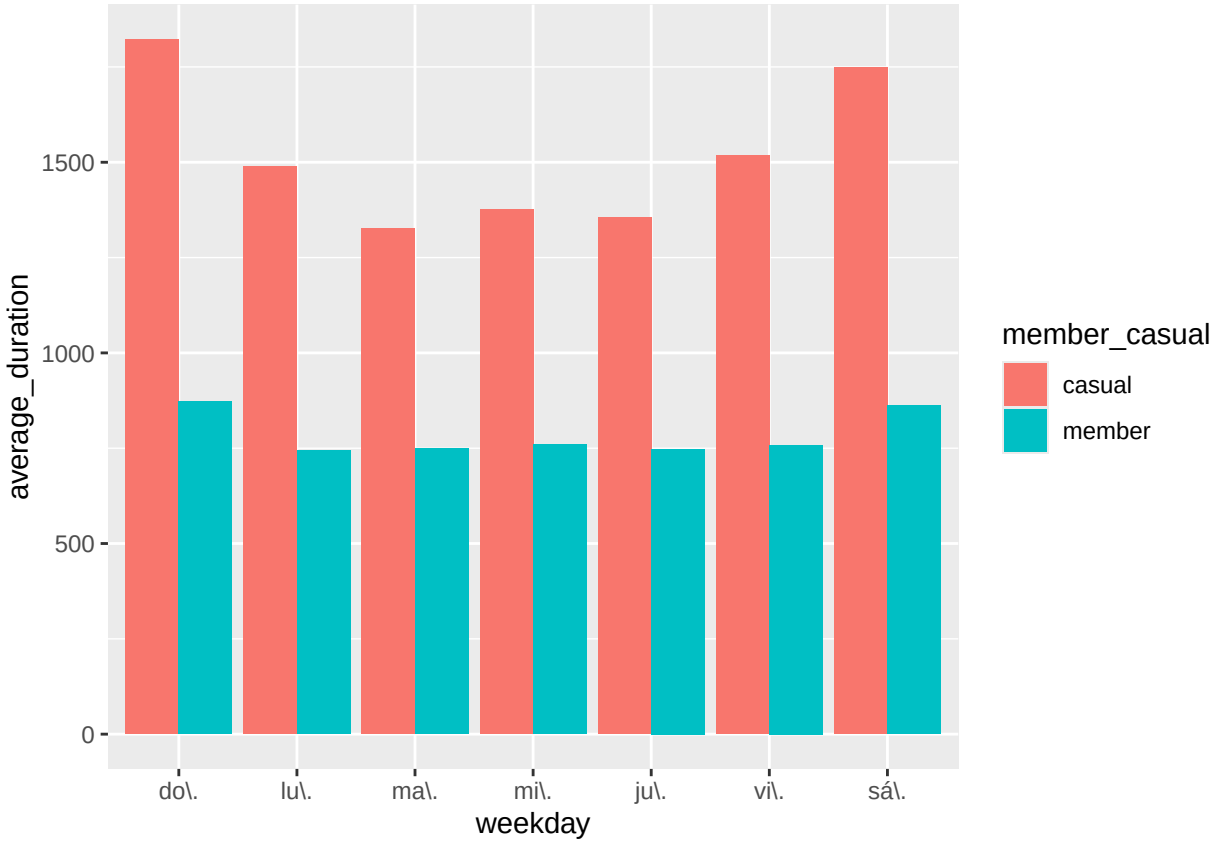
```
## `summarise()` has grouped output by 'member_casual'. You can override using the
## `.groups` argument.
```



Let's create a visualization for average duration:

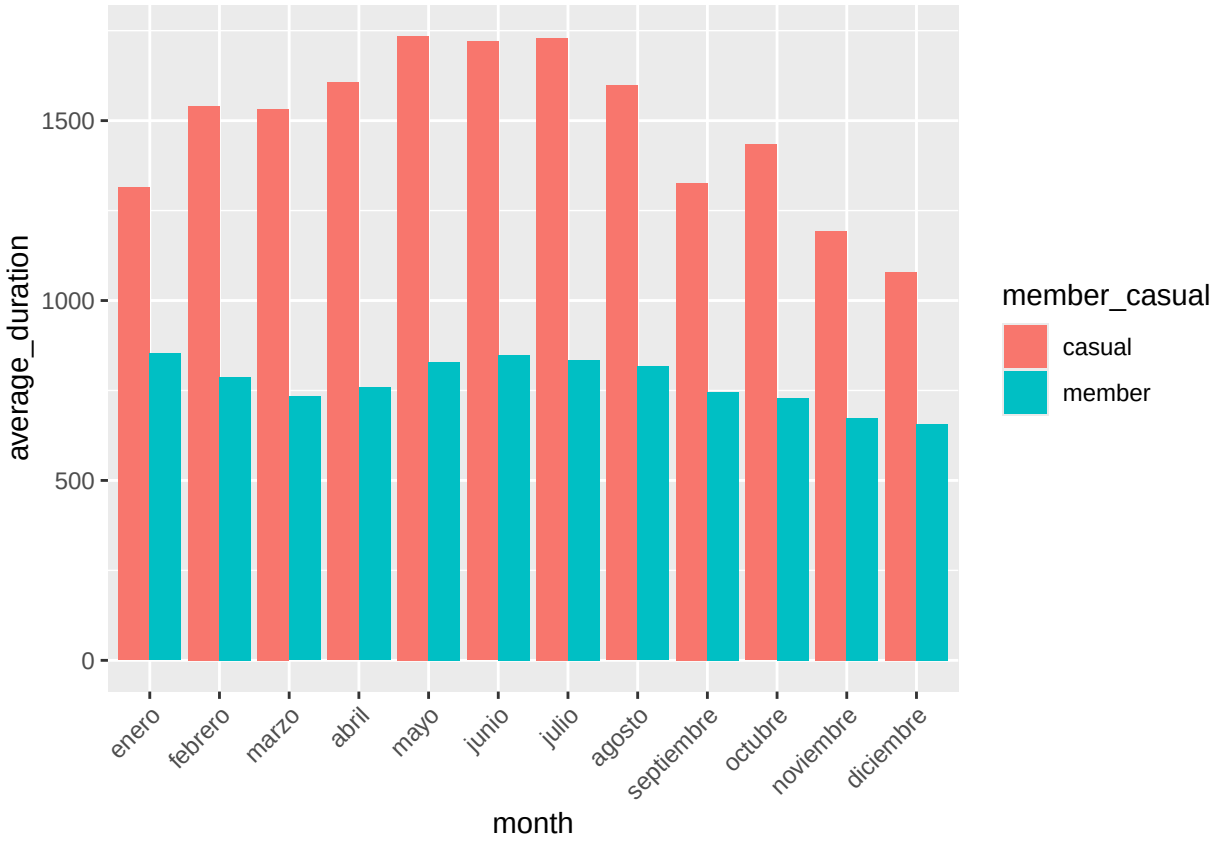
```
all_trips_v2 %>%
  mutate(weekday = wday(started_at, label = TRUE)) %>%
  group_by(member_casual, weekday) %>%
  summarise(number_of_rides = n() ,average_duration = mean(ride_length)) %>%
  arrange(member_casual, weekday)  %>%
  ggplot(aes(x = weekday, y = average_duration, fill = member_casual)) +
  geom_col(position = "dodge")
```

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



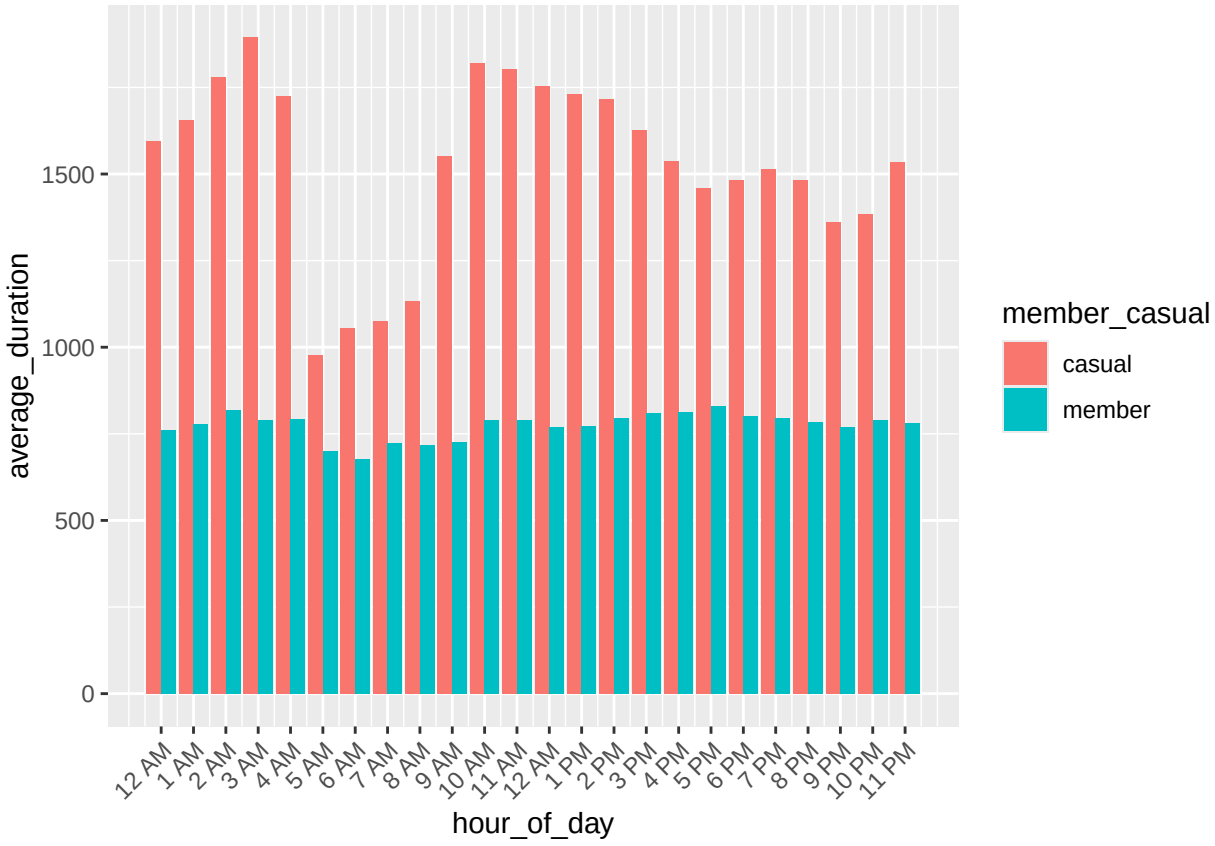

```
all_trips_v2 %>%
  mutate(month = factor(month, levels = c("enero", "febrero", "marzo", "abril", "mayo", "junio",
                                          "julio", "agosto", "septiembre", "octubre", "noviembre", "dici
  group_by(member_casual, month) %>%
  summarise(number_of_rides = n(),
            average_duration = mean(ride_length, na.rm = TRUE)) %>%
  arrange(member_casual, month) %>%
  ggplot(aes(x = month, y = average_duration, fill = member_casual)) +
  geom_col(position = "dodge") +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```

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




```
all_trips_v2 %>%
  mutate(hour_of_day = hour(started_at)) %>%
  group_by(member_casual, hour_of_day) %>%
  summarise(number_of_rides = n(),
            average_duration = mean(ride_length, na.rm = TRUE)) %>%
  arrange(member_casual, hour_of_day) %>%
  ggplot(aes(x = hour_of_day, y = average_duration, fill = member_casual)) +
  geom_col(position = "dodge") +
  scale_x_continuous(breaks = 0:23,
                     labels = function(x) {
                       hour_labels <- ifelse(x %% 12 == 0, "12 AM",
                                             ifelse(x < 12, paste0(x, " AM"),
                                                    paste0(x - 12, " PM")))
                       return(hour_labels)
                     }) +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```

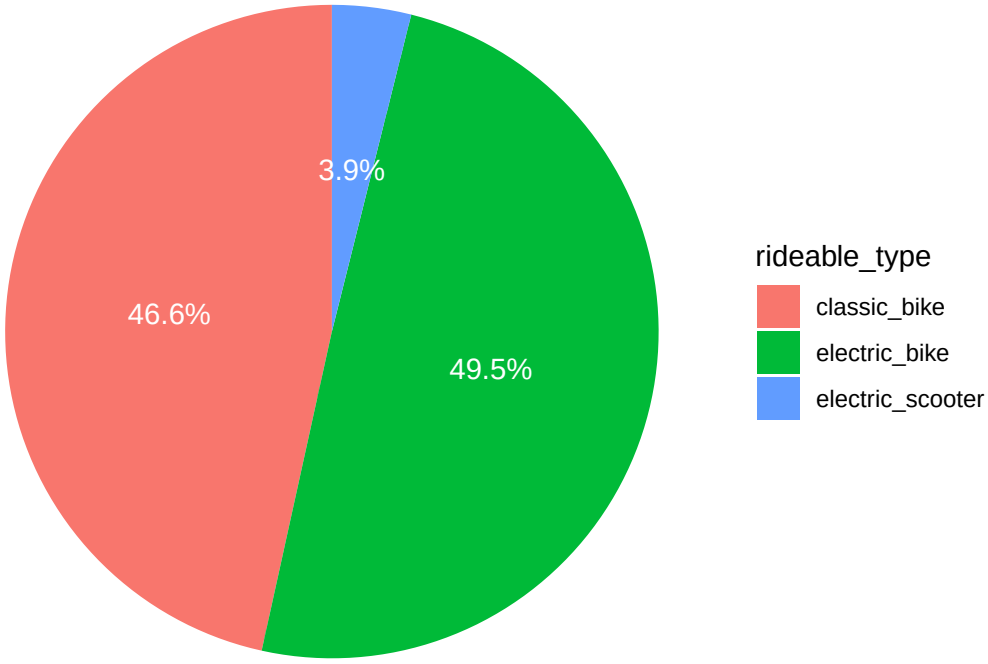
```
## `summarise()` has grouped output by 'member_casual'. You can override using the
## `.groups` argument.
```



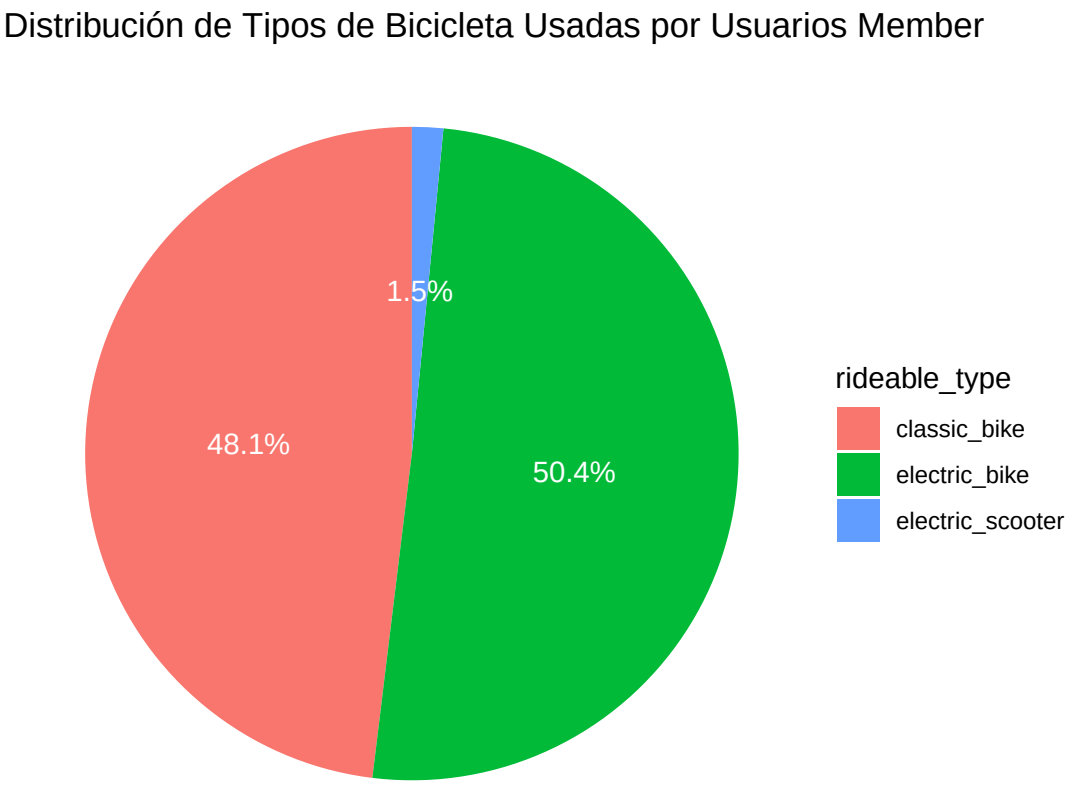
Let's try to visualize the type of bicycle used by each type of user:

```
all_trips_v2 %>%
  filter(member_casual == "casual") %>%
  count(rideable_type) %>%
  mutate(percentage = round(n / sum(n) * 100, 1)) %>%
  ggplot(aes(x = "", y = n, fill = rideable_type)) +
  geom_bar(stat = "identity", width = 1) +
  coord_polar(theta = "y") +
  labs(title = "Casual users") +
  theme_void() +
  theme(axis.text.x = element_blank()) +
  geom_text(aes(label = paste0(percentage, "%")),
            position = position_stack(vjust = 0.5), color = "white")
```

Casual users




```
all_trips_v2 %>%
  filter(member_casual == "member") %>%
  count(rideable_type) %>%
  mutate(percentage = round(n / sum(n) * 100, 1)) %>%
  ggplot(aes(x = "", y = n, fill = rideable_type)) +
  geom_bar(stat = "identity", width = 1) +
  coord_polar(theta = "y") +
  labs(title = "Distribución de Tipos de Bicicleta Usadas por Usuarios Member") +
  theme_void() +
  theme(axis.text.x = element_blank()) +
  geom_text(aes(label = paste0(percentage, "%")),
            position = position_stack(vjust = 0.5), color = "white")
```



STEP 5: EXPORT SUMMARY FILE FOR FURTHER ANALYSIS

Create a csv file that we will visualize in Excel, Tableau, or my presentation software:

```
counts <- aggregate(all_trips_v2$ride_length ~ all_trips_v2$member_casual + all_trips_v2$day_of_week, FU
write.csv(counts, file ="C:/Users/carme/Desktop/avg_ride_length.csv")
```