Bike case study

Rodrigo

2023-10-07

```
options(repos = "https://cran.rstudio.com/")
library(installr)
```

Bike Case Study

We will start by installing the packages we will use

```
install.packages("tidyverse")
```

```
## package 'tidyverse' successfully unpacked and MD5 sums checked
##
## The downloaded binary packages are in
##  C:\Users\USER\AppData\Local\Temp\RtmpM5CKkO\downloaded_packages
```

```
install.packages("ggplot2")
```

```
## package 'ggplot2' successfully unpacked and MD5 sums checked
##
## The downloaded binary packages are in
##  C:\Users\USER\AppData\Local\Temp\RtmpM5CKkO\downloaded_packages
```

```
install.packages("dplyr")
```

```
## package 'dplyr' successfully unpacked and MD5 sums checked
##
## The downloaded binary packages are in
##  C:\Users\USER\AppData\Local\Temp\RtmpM5CKkO\downloaded_packages
```

```
install.packages("here")
```

```
## package 'here' successfully unpacked and MD5 sums checked
##
## The downloaded binary packages are in
##  C:\Users\USER\AppData\Local\Temp\RtmpM5CKkO\downloaded_packages
```

```
library(tidyverse)
library(ggplot2)
library(dplyr)
library(readr)
library(here)
```

We continue by importing the data, in this case we will import the trips data sets from 2022 and they are separated in months.

```
jan <- here("Data trip 2022","202201-divvy-tripdata.csv")
feb <- here("Data trip 2022","202202-divvy-tripdata.csv")
mar <- here("Data trip 2022","202203-divvy-tripdata.csv")
apr <- here("Data trip 2022","202204-divvy-tripdata.csv")
mayy <- here("Data trip 2022","202205-divvy-tripdata.csv")
ju <- here("Data trip 2022","202206-divvy-tripdata.csv")
jul <- here("Data trip 2022","202207-divvy-tripdata.csv")
aug <- here("Data trip 2022","202208-divvy-tripdata.csv")
sep <- here("Data trip 2022","202209-divvy-publictripdata.csv")
oct <- here("Data trip 2022","202210-divvy-tripdata.csv")
nov <- here("Data trip 2022","202211-divvy-tripdata.csv")
dec <- here("Data trip 2022","202212-divvy-tripdata.csv")

January <- read_csv(jan)
February <-read_csv(feb)
March <-read_csv(mar)
April <-read_csv(apr)
May <-read_csv(mayy)
June <-read_csv(ju)
July <-read_csv(jul)
August <-read_csv(aug)
September <-read_csv(sep)
October <-read_csv(oct)
November <-read_csv(nov)
December <-read_csv(dec)
```

Combine the data

We will use a function to combine all the data from each month into one data set so we can work with only one data frame.

```
bike_data_Set<-rbind(January,February,March,April,May,June,July,August,September,October,November,Decem
```

Now that we have all the data together we can start the cleaning process.

Cleaning the data

We start by eliminating the columns we won't use.

```
New_bike_dset <-subset(bike_data_Set,select=-c(start_lat,end_lat,start_lng,end_lng))
```

Then we change the name of the columns, we will change the name of the columns that shows the user types and the bike types

```
New_bike_dset <- New_bike_dset %>% rename(ride_type=rideable_type)
New_bike_dset <- New_bike_dset %>% rename(user_type=member_casual)
```

We take a look at our data after changing some columns and see what contains now

```
glimpse(New_bike_dset) # We see what type of data are the variables
nrow(New_bike_dset) # We see how many rows have our data
colnames(New_bike_dset) # We see what are the names of the columns
dim(New_bike_dset) # We see how many rows and colums have our data
```

For analysis it will be a good a idea to have different columns with the month, day and year for the trips

```
New_bike_dset$date <- as.Date(New_bike_dset$started_at)
New_bike_dset$month <- format(as.Date(New_bike_dset$date), "%m")
New_bike_dset$day <- format(as.Date(New_bike_dset$date), "%d")
New_bike_dset$year <- format(as.Date(New_bike_dset$date), "%Y")
New_bike_dset$day_of_week <- format(as.Date(New_bike_dset$date), "%A")
```

We transform the dates so we can create a new column to determine the lenght of the ride

We turn the ride_length column into a numeric

We eliminate the trips that don't contain the Stations name and create a New data frame with only the data doesn't have NA

After the cleaning of the columns and adding columns that we will use we can start the next phase.

Analyse the data

We start by finding some important insights from the data with these functions to get a better sense of the usage of the bikes

```
summary(Trips_data$ride_length)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  -10122     363     636    1026    1141 2061244
```

Average trip length compare from the user types

```
aggregate(Trips_data$ride_length ~ Trips_data$user_type, FUN = mean)
```

```
##   Trips_data$user_type Trips_data$ride_length
## 1               casual              1439.5232
## 2               member               747.0765
```

Median trip length comprare from the user types

```
aggregate(Trips_data$ride_length ~ Trips_data$user_type, FUN = median)
```

```
##   Trips_data$user_type Trips_data$ride_length
## 1               casual                    831
## 2               member                    539
```

Min length trip from the user type

```
aggregate(Trips_data$ride_length ~ Trips_data$user_type, FUN = min)
```

```
##   Trips_data$user_type Trips_data$ride_length
## 1               casual                  -7621
## 2               member                 -10122
```

We see the average length ride over the days of the week per type of user

```
aggregate(Trips_data$ride_length ~Trips_data$user_type +Trips_data$day_of_week, FUN=mean)
```

```
##    Trips_data$user_type Trips_data$day_of_week Trips_data$ride_length
## 1                casual                domingo              1633.3988
## 2                member                domingo               830.9191
## 3                casual                 jueves              1284.1945
## 4                member                 jueves               721.8816
## 5                casual                  lunes              1490.0326
## 6                member                  lunes               721.9661
## 7                casual                 martes              1286.4023
## 8                member                 martes               707.4372
## 9                casual              miércoles              1243.0783
## 10               member              miércoles               710.8126
## 11               casual                 sábado              1605.9442
## 12               member                 sábado               838.8782
## 13               casual                viernes              1341.4220
## 14               member                viernes               733.6143
```

We can make it better and show us the number of ride per day and the average length arrange by user type

```
Trips_data %>%
group_by(user_type,day_of_week) %>%
summarise(total_rides=n(),
average_length=mean(ride_length)) %>%
arrange(user_type,day_of_week)
```

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

```
## # A tibble: 14 x 4
## # Groups:   user_type [2]
##    user_type day_of_week total_rides average_length
##    <chr>     <chr>             <int>          <dbl>
##  1 casual    domingo          301316          1633.
##  2 casual    jueves           230007          1284.
##  3 casual    lunes            210759          1490.
##  4 casual    martes           196390          1286.
##  5 casual    miércoles        203576          1243.
##  6 casual    sábado           367344          1606.
##  7 casual    viernes          248797          1341.
##  8 member    domingo          297733           831.
##  9 member    jueves           415890           722.
## 10 member    lunes            375171           722.
## 11 member    martes           411249           707.
## 12 member    miércoles        412795           711.
## 13 member    sábado           338279           839.
## 14 member    viernes          360054           734.
```

We've created a table that shows what type of vehicle is used on each day

```
Trips_data %>%
group_by(user_type,day_of_week,ride_type) %>%
summarise(table(ride_type))
```

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

```
## # A tibble: 35 x 4
## # Groups:   user_type, day_of_week [14]
##    user_type day_of_week ride_type     ‘table(ride_type)‘
##    <chr>     <chr>       <chr>         <table[1d]>
##  1 casual    domingo     classic_bike  158094
##  2 casual    domingo     docked_bike    35192
##  3 casual    domingo     electric_bike 108030
##  4 casual    jueves      classic_bike  113477
##  5 casual    jueves      docked_bike    19486
##  6 casual    jueves      electric_bike  97044
##  7 casual    lunes       classic_bike  103966
##  8 casual    lunes       docked_bike    22197
##  9 casual    lunes       electric_bike  84596
## 10 casual    martes      classic_bike   95811
## # i 25 more rows
```

We see who has more trips in total in the 12 months

```
Trips_data %>%
group_by(user_type) %>%
summarise(table(user_type))
```

```
## # A tibble: 2 x 2
##   user_type ‘table(user_type)‘
##   <chr>     <table[1d]>
## 1 casual    1758189
## 2 member    2611171
```

We see which type of bike is the most used

```
Trips_data %>%
group_by(ride_type) %>%
summarise(table(ride_type))
```

```
## # A tibble: 3 x 2
##   ride_type     ‘table(ride_type)‘
##   <chr>         <table[1d]>
## 1 classic_bike  2597426
## 2 docked_bike    174858
## 3 electric_bike 1597076
```

Most visited stations by casual users top 10 stations

```
Trips_data %>% group_by(start_station_name,user_type) %>%
filter(user_type=="casual") %>%
summarise(average_ride=mean(ride_length),rides_by_station=n()) %>%
arrange(-rides_by_station) %>% head(10)
```

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

```
## # A tibble: 10 x 4
## # Groups:   start_station_name [10]
##    start_station_name                 user_type average_ride rides_by_station
##    <chr>                              <chr>            <dbl>            <int>
##  1 Streeter Dr & Grand Ave            casual           2155.            55061
##  2 DuSable Lake Shore Dr & Monroe St  casual           2236.            30262
##  3 Millennium Park                    casual           2456.            23951
##  4 Michigan Ave & Oak St              casual           2173.            23761
##  5 DuSable Lake Shore Dr & North Blvd casual           1758.            22157
##  6 Shedd Aquarium                     casual           1835.            19421
##  7 Theater on the Lake                casual           1838.            17333
##  8 Wells St & Concord Ln              casual           1063.            14834
##  9 Dusable Harbor                     casual           2173.            13271
## 10 Clark St & Armitage Ave            casual           1339.            12779
```

Most visited stations by member users top 10 stations

```
Trips_data %>% group_by(start_station_name,user_type) %>%
filter(user_type=="member") %>%
summarise(average_ride=mean(ride_length),rides_by_station=n()) %>%
arrange(-rides_by_station) %>% head(10)
```

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

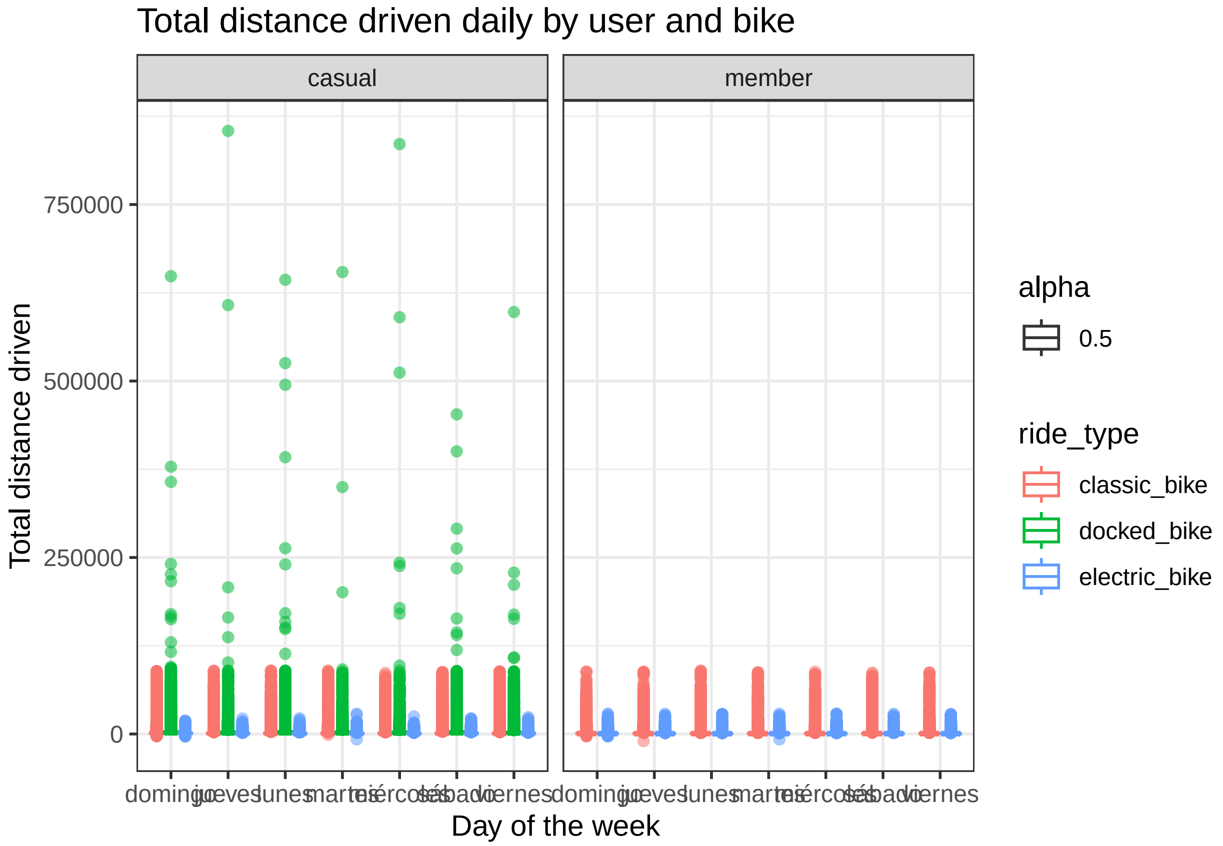
```
## # A tibble: 10 x 4
## # Groups:   start_station_name [10]
##    start_station_name           user_type average_ride rides_by_station
##    <chr>                        <chr>            <dbl>            <int>
##  1 Kingsbury St & Kinzie St     member            549.            23523
##   2 Clark St & Elm St             member          709.            20581
##   3 Wells St & Concord Ln         member          698.            19674
##   4 Clinton St & Washington Blvd member          636.            18828
##   5 Loomis St & Lexington St      member          564.            18252
##   6 Clinton St & Madison St       member          614.            18007
##   7 University Ave & 57th St      member          487.            17581
##   8 Ellis Ave & 60th St           member          410.            17504
##   9 Wells St & Elm St             member          644.            17496
##  10 Streeter Dr & Grand Ave       member         1245.            16208
```

Once we have the data analysed and we've got the information we need we will begin the visualitions:

Visualization

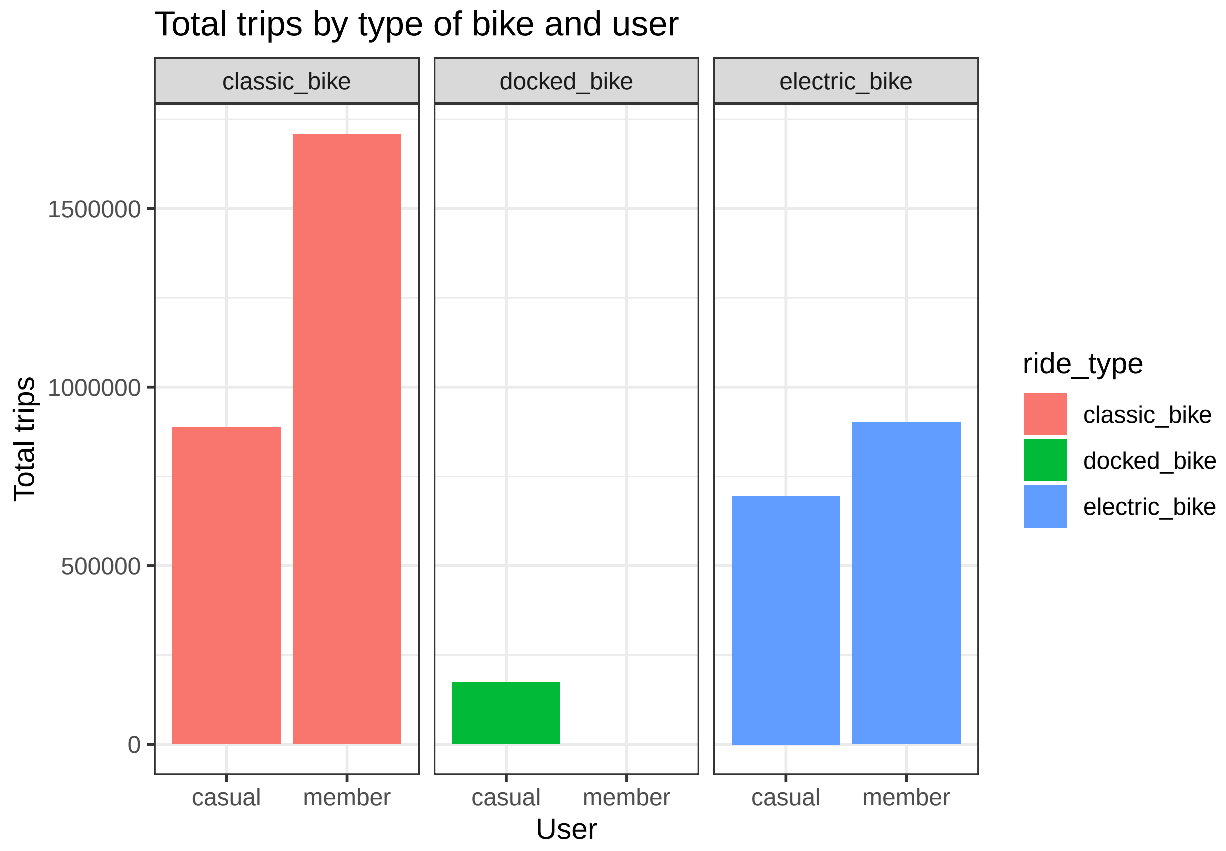
Total distance driven daily by user and bike

```
Trips_data %>% filter(ride_length <1500000) %>%
  ggplot(aes(x= day_of_week, y=ride_length))+
  geom_boxplot(aes(colour= ride_type, alpha = 0.5))+
  facet_wrap(~user_type)+theme_bw()+labs(y="Total distance driven",x="Day of the week", title = "Total d
```



Total rides by user and separated by type of bike

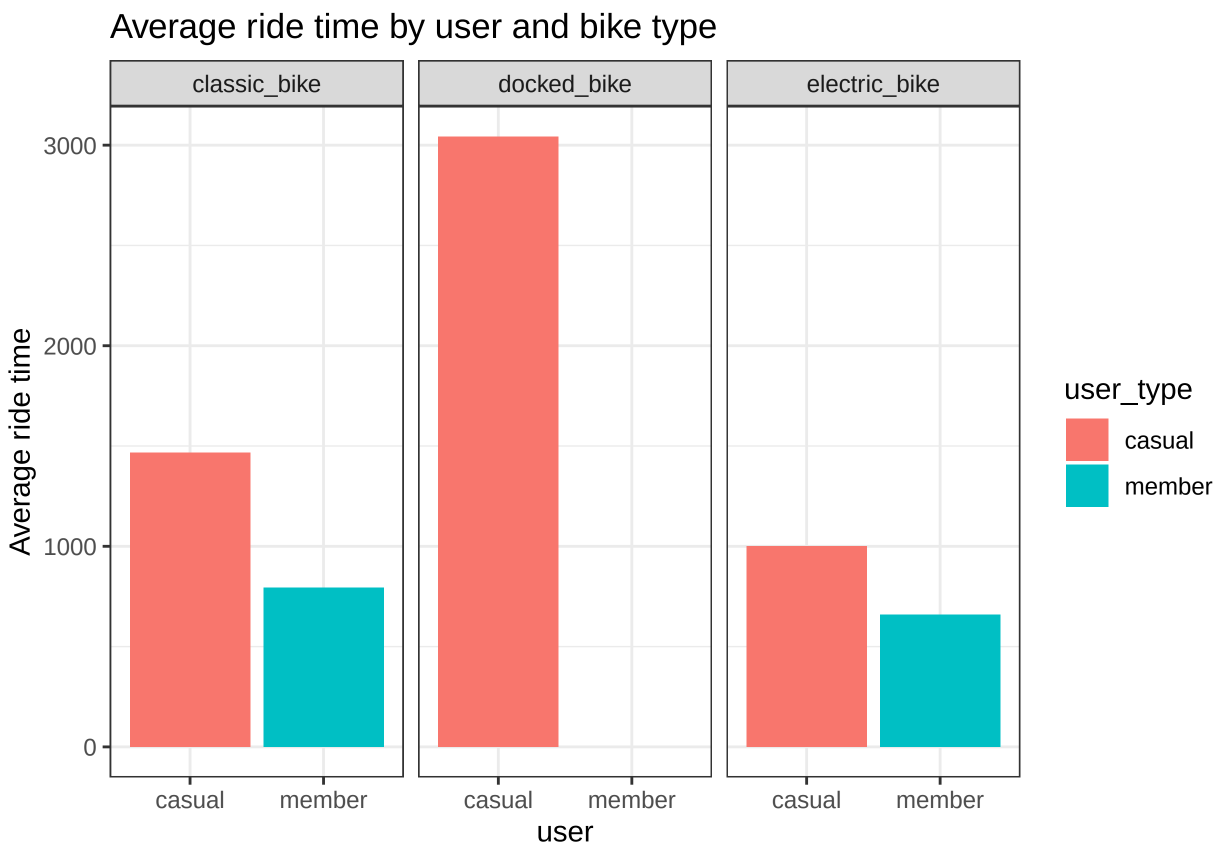
```
Trips_data %>%
  ggplot(aes(x=user_type))+
  geom_bar(aes(fill= ride_type))+
  facet_wrap(~ride_type)+theme_bw()+labs(y="Total trips",x="User",title="Total trips by type of bike an
```



Average ride time by user with separated by type of bike

```
Trips_data %>%  group_by(user_type, ride_type) %>%
  summarise(total_rides=n(),
            average_ride= mean(ride_length)) %>%
  ggplot() + geom_bar(aes(x= user_type, y=average_ride,fill=user_type),
                      stat="summary")+ facet_wrap(~ride_type)+
  theme_bw()+labs(y="Average ride time",x="user",title="Average ride time by user and bike type")
```

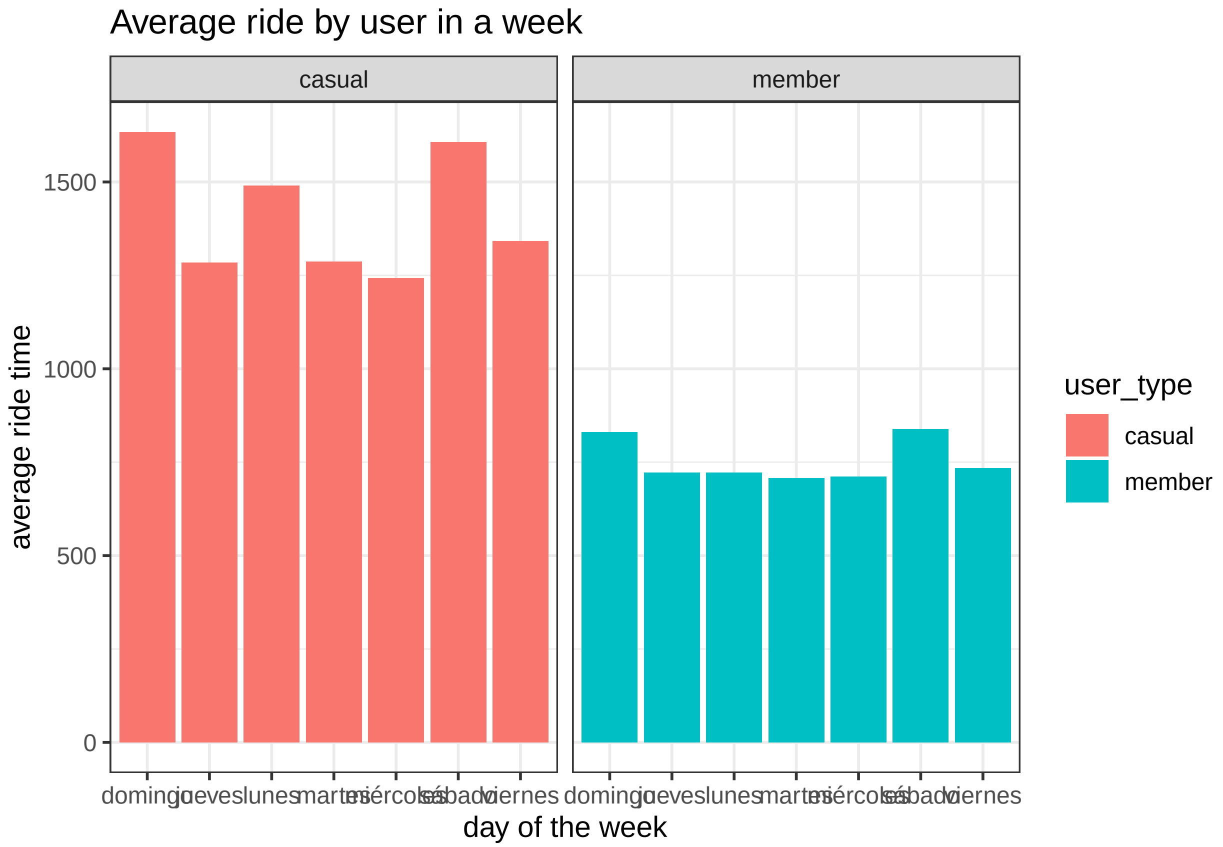
```
## ‘summarise()‘ has grouped output by ’user_type’. You can override using the
## ‘.groups‘ argument.
## No summary function supplied, defaulting to ‘mean_se()‘
## No summary function supplied, defaulting to ‘mean_se()‘
## No summary function supplied, defaulting to ‘mean_se()‘
```



Average ride time by user daily

```
Trips_data %>%  group_by(day_of_week,user_type) %>%
  summarise(total_rides=n(),
            average_ride=mean(ride_length)) %>%
  ggplot()+ geom_bar(aes(x=day_of_week,
                         y=average_ride,
                         fill=user_type),
                     stat="summary")+ facet_wrap(~user_type)+
  theme_bw()+
  labs(y="average ride time",x="day of the week",title="Average ride by user in a week")
```

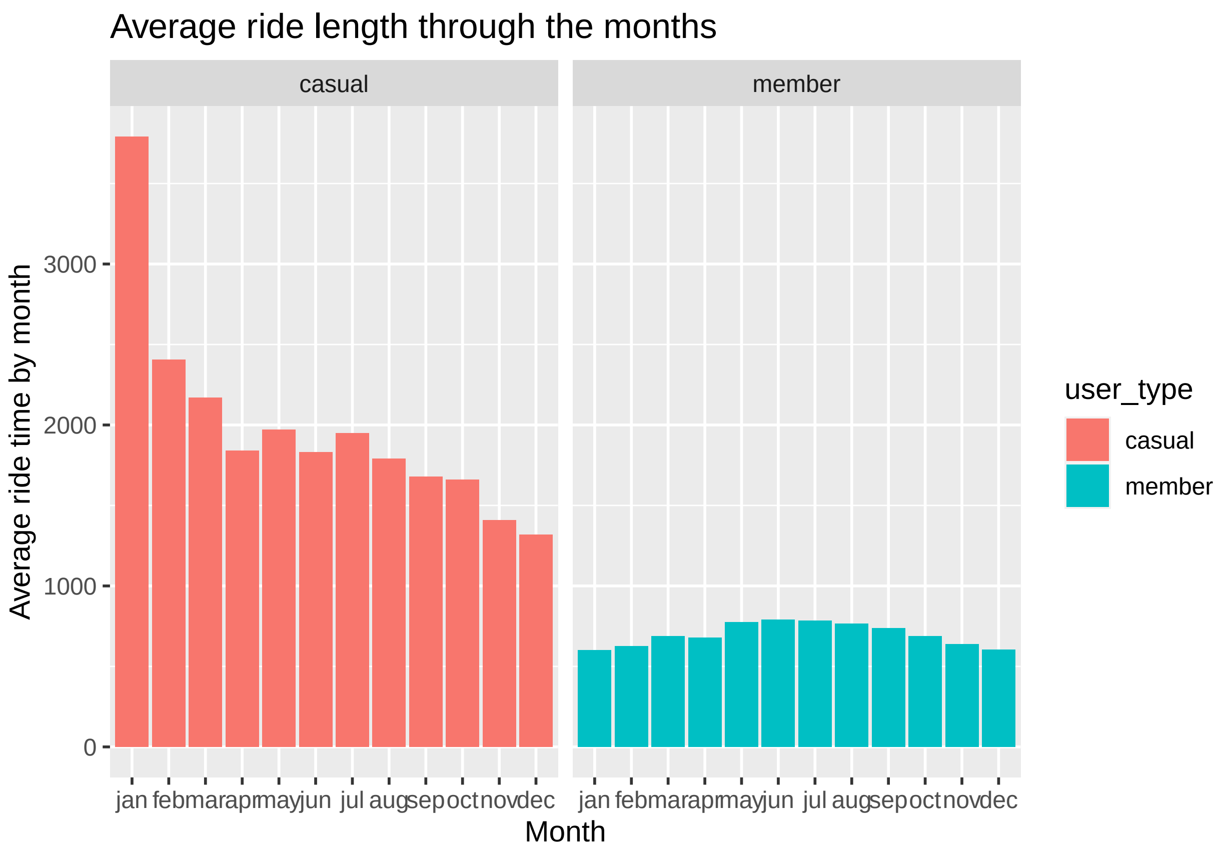
```
## ‘summarise()‘ has grouped output by ’day_of_week’. You can override using the
## ‘.groups‘ argument.
## No summary function supplied, defaulting to ‘mean_se()‘
## No summary function supplied, defaulting to ‘mean_se()‘
```



Average ride time by user monthly

```
Trips_data %>% group_by(month,user_type,ride_type) %>%
  summarise(total_rides=n(),
            average_ride=mean(ride_length)) %>%
  ggplot()+ geom_histogram(aes(x= month, y=average_ride,fill=user_type),
                           stat = "summary")+facet_wrap(~user_type) +labs(y="Average ride time by month
  scale_x_discrete(breaks=c("01","02","03","04","05","06","07","08","09","10","11","12"),
                   labels=c("jan","feb","mar","apr","may","jun","jul","aug","sep","oct","nov","dec"))
```

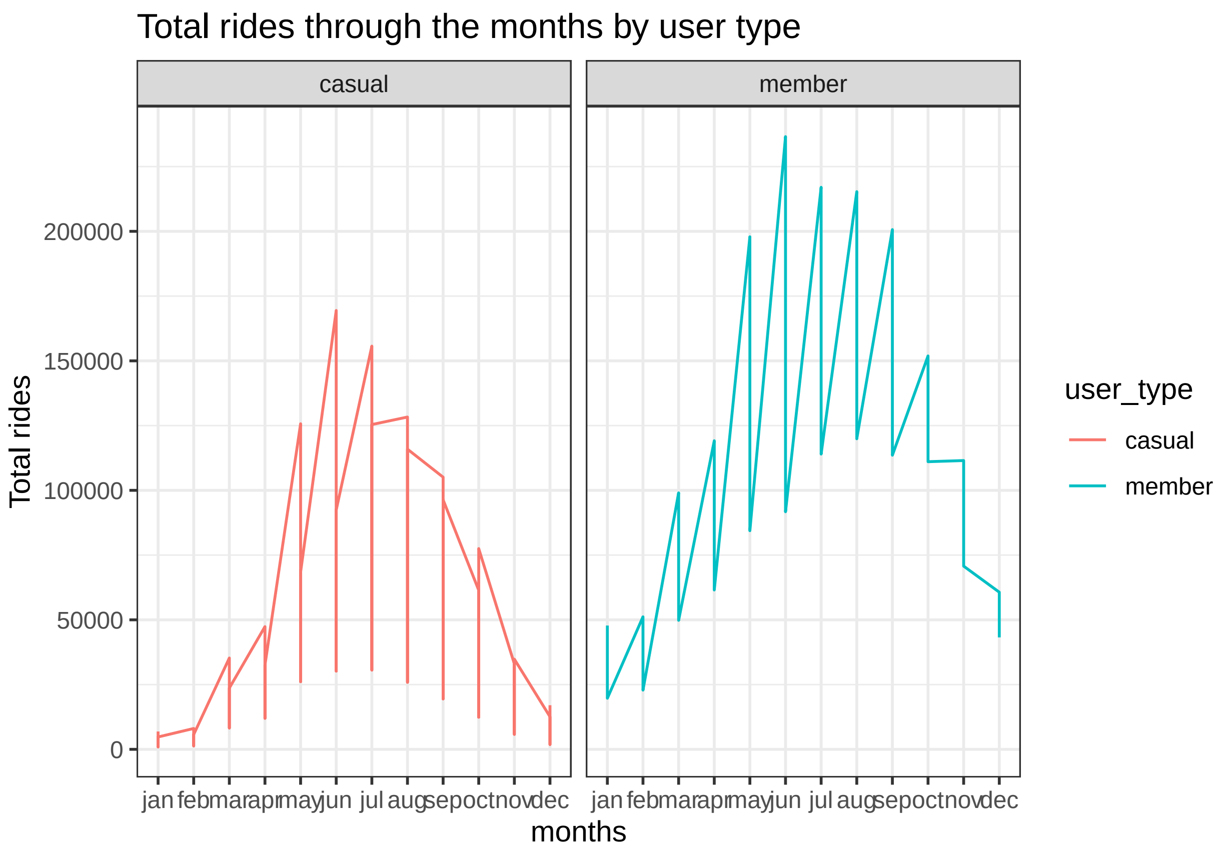
```
## `summarise()` has grouped output by 'month', 'user_type'. You can override
## using the `.groups` argument.

## Warning in geom_histogram(aes(x = month, y = average_ride, fill = user_type), :
## Ignoring unknown parameters: `binwidth`, `bins`, and `pad`

## No summary function supplied, defaulting to `mean_se()`
## No summary function supplied, defaulting to `mean_se()`
```



Total rides by user monthly

```
Trips_data %>% group_by(month,user_type,ride_type) %>%
  summarise(total_rides=n(),
            average_ride=mean(ride_length)) %>%
  ggplot(aes(month, total_rides,group=user_type))+ geom_line(aes(colour=user_type))+facet_wrap(~user_typ
  theme_bw()+ scale_x_discrete(breaks=c("01","02","03","04","05","06","07","08","09","10","11","12"),
                               labels=c("jan","feb","mar","apr","may","jun","jul","aug","sep","oct","no
  labs(y="Total rides",x="months",title = "Total rides through the months by user type")
```

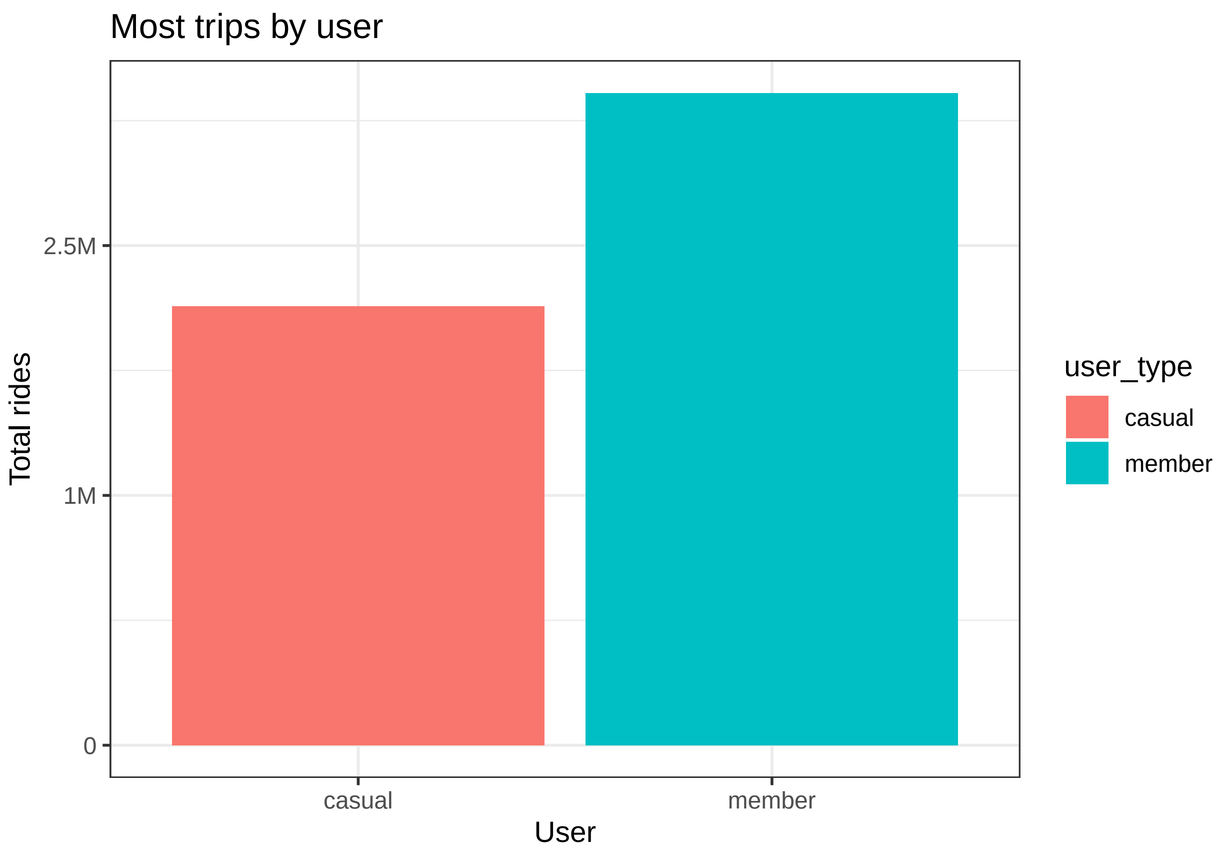
```
## `summarise()` has grouped output by 'month', 'user_type'. You can override
## using the `.groups` argument.
```



Total rides throughout the year by user

```
Trips_data %>% group_by(user_type) %>%
  summarise(total_rides=n()) %>%
  ggplot(aes(user_type,total_rides,fill=user_type))+geom_bar(stat = "summary")+scale_y_continuous(breaks
```

```
## No summary function supplied, defaulting to ‘mean_se()‘
```


Most trips by user
Total rides
2.5M
1M
0
casual
member
User
user_type
casual
member