Google Data Analytics Case Study Report

Aron Rana

2024-10-01

Contents

Introduction

The key to ensuring the establishment and success of Cyclistic, a fictional bike-share company, lies in expanding the number of annual subscription memberships. The marketing analytics team has been tasked to analyze historical company data from Cyclistic for the period of August 2023 to August 2024 in order to suggest marketing strategies that would increase the number of annual subscription memberships, and thus annual profit. My specific business task was to identify how subscription members and casual riders differ in terms of product use.

This report will include the steps taken to prepare data and the insights gained from analysis.

RStudio Desktop (Version 2024.09.0+375) for macOS Ventura 13.7 (22H123) was used for this project.

Step 1 - Load the required R packages

The following R packages were installed and loaded:

```
library(tidyverse)
library(skimr)
library(lubridate)
library(vctrs)
library(janitor)
library(dplyr)
library(ggplot2)
library(remotes)
library(lubritime)
library(knitr)
library(magrittr)
```

Step 2 - Import the required datasets

Historical company data from Cyclistic, for the period of August 2023 to August 2024, were all obtained from *https://divvy-tripdata.s3.amazonaws.com/index.html* in the form of downloadable CSV files, with each month having a separate CSV file. The data was made available using the appropriate license (*https://divvybikes.com/data-license-agreement*). The downloaded CSV files were stored in a subfolder on my password protected personal computer. This subfolder, for CSV files only, was contained in a main folder that was used for exclusively for this project. Backups of the CSV files were stored on an external flash drive in addition to Google Drive. Other than myself, no one had access to the datasets or their backups.

The required CSV files were imported to the global environment on RStudio Desktop using the code below:

```
aug_23 <- read.csv("202308-divvy-tripdata.csv")
sep_23 <- read.csv("202309-divvy-tripdata.csv")
oct_23 <- read.csv("202310-divvy-tripdata.csv")
nov_23 <- read.csv("202311-divvy-tripdata.csv")
dec_23 <- read.csv("202312-divvy-tripdata.csv")
jan_24 <- read.csv("202401-divvy-tripdata.csv")
feb_24 <- read.csv("202402-divvy-tripdata.csv")
mar_24 <- read.csv("202403-divvy-tripdata.csv")
apr_24 <- read.csv("202404-divvy-tripdata.csv")
may_24 <- read.csv("202405-divvy-tripdata.csv")
june_24 <- read.csv("202406-divvy-tripdata.csv")
july_24 <- read.csv("202407-divvy-tripdata.csv")
aug_24 <- read.csv("202408-divvy-tripdata.csv")
```

Step 3 - Clean the imported datasets

All downloaded datasets were previewed using the `skim_without_charts()` and `View()` functions. The comprehensive first party datasets already had personally identifiable information removed. In the available data, casual riders are referred to as users who purchase single or full-day passes. Members are referred to as users who purchase annual subscription passes.

Upon preview of all the datasets, I found that the CSV files for June 2024, July 2024 and August 2024 each had a number of records from the previous month. This discovery was made using the sort feature in the new window that was generated by running the appropriate `View()` function code. However, these records were not removed as all the dataframes were to be later combined to form one collective dataframe, and it would not matter if the aforementioned datasets had a number of records from previous months as long as there were no duplicates. The second issue I discovered during the initial preview of the available data was that the datasets did not contain information on how Cyclistic's products were used by customers in terms of purpose. Having data about the products being used for purposes such as leisure or daily commute would have provided valuable insight. The third issue with the available data was that some ride durations (which I added as a new column to the datasets) were negative values, values between 0 and 1 second and other very short durations that do not make sense in the context of bike rentals.

Records with trip durations less than 1 second were removed. Negative values for trip duration were removed because the end time was before the start time. Zero second values for trip duration were removed as no movement occurred. Trip duration values >0 seconds but <1 second were removed due to these values being impractical for bike rentals.

The `skim_without_charts()` function was used to generate comprehensive summaries of all imported datasets in order to preview each dataset. No records had white spaces. Missing (empty) values were under columns that would not be used for the analysis; these columns would later be removed from each dataset. These records with missing values were still used in the analysis as they contained valid and useful information for the analysis under the other columns.

Unique values for the product_type and type_of_user columns were verified using the `skim_without_chart()` and `unique()` functions.

The available data contained information (such as ride duration, start date/time values, type of user and product type) that was relevant to the business task I had to solve. The specific columns I needed for my analysis were selected using the `select()` function. Columns were renamed where necessary using the `rename()` function. Subsequent dataframes were named appropriately to maintain file organization.

All imported datasets were processed exactly in the following manner, using the CSV file for August 2023 as an example:

```
aug_23_clean <- aug_23 %>%
  select(ride_id, rideable_type, started_at, ended_at, member_casual) %>%
  rename(product_type = rideable_type) %>%
  rename(start_DateTime = started_at) %>%
  rename(end_DateTime = ended_at) %>%
  rename(type_of_user = member_casual)

aug_23_clean$start_DateTime <- ymd_hms(aug_23_clean$start_DateTime)
aug_23_clean$end_DateTime <- ymd_hms(aug_23_clean$end_DateTime)
```

```
aug_23_dt <- aug_23_clean %>%
  mutate(start_day = wday(aug_23_clean$start_DateTime, label = TRUE, abbr = FALSE)) %>%
  mutate(ride_duration = aug_23_clean$end_DateTime - aug_23_clean$start_DateTime)

aug_23_f <- aug_23_dt %>%
  filter(ride_duration >= 1)
```

The CSV file for August 2023 was used as an example for the code above. The above code was altered appropriately for each imported CSV file by copying and pasting the above code into the Pages app for macOS and then using the Find and Replace feature to make appropriate changes to the code. Thus, the data cleaning procedure for each imported dataset followed the exact same outline.

The final dataframes for each dataset were all combined to form one collective dataframe with all the required data in one place (files with the suffix "_f" were combined). This final dataframe was titled "total_data" using the code below:

```
total_data <- rbind(aug_23_f, sep_23_f, oct_23_f, nov_23_f, dec_23_f, jan_24_f, feb_24_f, mar_24_f, apr_24_f, may_24_f, june_24_f, july_24
```

The `ride_id` column of the `total_data` dataframe contained duplicate records. The duplicate records were all from 2024-05-31, 2024-06-01 and 2024-06-02. The start date/time for these duplicate records were on 2024-05-31 and the end date/time were on the following month on either 2024-06-01 or 2024-06-02. Because of this, the duplicate records in question will appear in the month of May 2024 and also in June 2024.

Duplicate records in the `ride_id` column of the `total_data` dataframe were identified and removed using the code below:

```
#Identifying duplicate records in total_data

vec_duplicate_any(total_data$ride_id)
#TRUE is returned if there are duplicate files, FALSE is returned if there are no duplicate records#

length(unique(total_data$ride_id)) == nrow(total_data)
#TRUE is returned if no duplicate records are present, FALSE is returned if there are duplicate records present#

n_occur_ride_id <- data.frame(table(total_data$ride_id))

duplicate_ride_id <- n_occur_ride_id[n_occur_ride_id$Freq > 1,]

duplicate_ride_id_records <- total_data[total_data$ride_id %in% n_occur_ride_id$Var1[n_occur_ride_id$Freq > 1],]
#Records with duplicate ‘ride_id‘ were extracted using the code above.

duplicate_ride_id_DayMonth <- duplicate_ride_id_records %>%
  mutate(start_month = month(duplicate_ride_id_records$start_DateTime, label = TRUE, abbr = FALSE)) %>%
```

```
  mutate(end_month = month(duplicate_ride_id_records$end_DateTime, label = TRUE, abbr = FALSE)) %>%
  mutate(start_day = day(duplicate_ride_id_records$start_DateTime)) %>%
  mutate(end_day = day(duplicate_ride_id_records$end_DateTime))
#The day and month for the duplicate records were extracted using the code above

unique(duplicate_ride_id_DayMonth$start_month)
unique(duplicate_ride_id_DayMonth$end_month)
unique(duplicate_ride_id_DayMonth$start_day)
unique(duplicate_ride_id_DayMonth$end_day)
#The day and month for the duplicate records were verified using the code above
```

Duplicate records from the `total_data` dataframe were removed. The resulting dataframe was titled "total_data_f":

```
total_data_f <- total_data %>%
      distinct(ride_id, .keep_all = TRUE)
```

To verify that the dataframe `total_data_f` no longer had duplicate records, the following code was run:

```
#Verifying that total_data_f has no duplicate records

vec_duplicate_any(total_data_f$ride_id)
#TRUE is returned if there are duplicate files, FALSE is returned if there are no duplicate records#

length(unique(total_data_f$ride_id)) == nrow(total_data_f)
#TRUE is returned if no duplicate records are present, FALSE is returned if there are duplicate records present#
```

Step 4 - Verify data cleaning procedures

I verified that data has been cleaned and merged accurately by reading all the `skim_without_charts()` summaries for all dataframes made. I made sure no records had been removed without reason, that there were no white spaces and made sure that the `product_type` and `type_of_user` columns maintained their unique values from their original dataframes using the `unique()` function.

Step 5 - Create summary statistics

The following summary statistics were made after cleaning the available data:

Number of rides by type of user

```
## # A tibble: 2 x 2
## # Groups:   type_of_user [2]
##   type_of_user       n
##   <chr>          <int>
## 1 casual       2355874
## 2 member       4113199
```

Number of rides by product type and type of user

```
## # A tibble: 5 x 3
## # Groups:   product_type, type_of_user [5]
##   product_type  type_of_user       n
##   <chr>         <chr>          <int>
## 1 classic_bike  casual       1130251
## 2 classic_bike  member       2084678
## 3 docked_bike   casual         15957
## 4 electric_bike casual       1209666
## 5 electric_bike member       2028521
```

Number of rides by day of rental, product type and type of user

```
## # A tibble: 35 x 4
## # Groups:   start_day, product_type, type_of_user [35]
##    start_day product_type  type_of_user      n
##    <ord>     <chr>         <chr>         <int>
##  1 Sunday    classic_bike  casual       204106
##  2 Sunday    classic_bike  member       237182
##  3 Sunday    docked_bike   casual         2903
##  4 Sunday    electric_bike casual       185898
##  5 Sunday    electric_bike member       221227
##  6 Monday    classic_bike  casual       120068
##  7 Monday    classic_bike  member       289463
##  8 Monday    docked_bike   casual         1357
##  9 Monday    electric_bike casual       134296
## 10 Monday    electric_bike member       267999
## # i 25 more rows
```

Number of rides by month of rental, day of rental, product type and type of user

```
## # A tibble: 343 x 5
## # Groups:   start_month, start_day, product_type, type_of_user [343]
##    start_month start_day product_type  type_of_user     n
##    <ord>       <ord>     <chr>         <chr>        <int>
##  1 January     Sunday    classic_bike  casual        1081
##  2 January     Sunday    classic_bike  member        5129
##  3 January     Sunday    electric_bike casual        1291
##  4 January     Sunday    electric_bike member        3899
##  5 January     Monday    classic_bike  casual        1686
##  6 January     Monday    classic_bike  member       10967
##  7 January     Monday    electric_bike casual        2367
##  8 January     Monday    electric_bike member        8807
##  9 January     Tuesday   classic_bike  casual        1546
## 10 January     Tuesday   classic_bike  member       10869
## # i 333 more rows
```

Overall minimum ride duration

```
##   minimum_ride_duration
## 1                    1S
```

Overall maximum ride duration

```
##   maximum_ride_duration
## 1         68d 9H 29M 4S
```

Overall average ride duration

```
##   average_ride_duration
## 1 18M 7.53430603472839S
```

Overall median ride duration

```
##   median_ride_duration
## 1 9M 51.7750000953674S
```

Minimum ride duration by type of user

```
## # A tibble: 2 x 2
##   type_of_user minimum_ride_duration
##   <chr>        <Period>
## 1 casual       1S
## 2 member       1S
```

Maximum ride duration by type of user

```
## # A tibble: 2 x 2
##   type_of_user maximum_ride_duration
##   <chr>        <Period>
## 1 casual       68d 9H 29M 4S
## 2 member       1d 1H 59M 48S
```

Average ride duration by type of user

```
## # A tibble: 2 x 2
##   type_of_user average_ride_duration
##   <chr>        <Period>
## 1 casual       27M 0.0160860928376678S
## 2 member       13M 2.55028238000045S
```

Median ride duration by type of user

```
## # A tibble: 2 x 2
##   type_of_user median_ride_duration
##   <chr>        <Period>
## 1 casual       12M 18.5479999780655S
## 2 member       8M 48S
```

Minimum ride duration by product type and type of user

```
## # A tibble: 5 x 3
## # Groups:   product_type [3]
```

```
##   product_type  type_of_user minimum_ride_duration
##   <chr>         <chr>        <Period>
## 1 classic_bike  casual       1S
## 2 classic_bike  member       1S
## 3 docked_bike   casual       1S
## 4 electric_bike casual       1S
## 5 electric_bike member       1S
```

Maximum ride duration by product type and type of user

```
## # A tibble: 5 x 3
## # Groups:   product_type [3]
##   product_type  type_of_user maximum_ride_duration
##   <chr>         <chr>        <Period>
## 1 classic_bike  casual       1d 1H 59M 56S
## 2 classic_bike  member       1d 1H 59M 48S
## 3 docked_bike   casual       68d 9H 29M 4S
## 4 electric_bike casual       8H 0M 26.3819999694824S
## 5 electric_bike member       8H 1M 30S
```

Average ride duration by product type and type of user

```
## # A tibble: 5 x 3
## # Groups:   product_type [3]
##   product_type  type_of_user average_ride_duration
##   <chr>         <chr>        <Period>
## 1 classic_bike  casual       37M 1.46452704223839S
## 2 classic_bike  member       14M 41.517092557283S
## 3 docked_bike   casual       4H 20M 36.9175910258818S
## 4 electric_bike casual       14M 33.1525724074755S
## 5 electric_bike member       11M 20.8437031004385S
```

Median ride duration by product type and type of user

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

```
## # A tibble: 5 x 3
## # Groups:   product_type [3]
##   product_type  type_of_user median_ride_duration
##   <chr>         <chr>        <Period>
## 1 classic_bike  casual       15M 43S
## 2 classic_bike  member       9M 11.5755001306534S
## 3 docked_bike   casual       29M 13S
## 4 electric_bike casual       9M 53S
## 5 electric_bike member       8M 25.0089998245239S
```

Minimum ride duration by day of rental, product type and type of user

```
## # A tibble: 35 x 4
## # Groups:   start_day, product_type [21]
##    start_day product_type  type_of_user minimum_ride_duration
##    <ord>     <chr>         <chr>        <Period>
##  1 Sunday    classic_bike  casual       1S
##  2 Sunday    classic_bike  member       1S
##  3 Sunday    docked_bike   casual       2S
##  4 Sunday    electric_bike casual       1S
##  5 Sunday    electric_bike member       1S
##  6 Monday    classic_bike  casual       1S
##  7 Monday    classic_bike  member       1S
##  8 Monday    docked_bike   casual       2S
##  9 Monday    electric_bike casual       1S
## 10 Monday    electric_bike member       1S
## # i 25 more rows
```

Maximum ride duration by day of rental, product type and type of user

```
## # A tibble: 35 x 4
## # Groups:   start_day, product_type [21]
##    start_day product_type  type_of_user maximum_ride_duration
##    <ord>     <chr>         <chr>        <Period>
##  1 Sunday    classic_bike  casual       1d 1H 7M 46S
##  2 Sunday    classic_bike  member       1d 1H 0M 29S
##  3 Sunday    docked_bike   casual       43d 15H 47M 6S
##  4 Sunday    electric_bike casual       8H 0M 26.3819999694824S
```

```
##  5 Sunday    electric_bike member       8H 0M 0S
##  6 Monday    classic_bike  casual       1d 1H 0M 25S
##  7 Monday    classic_bike  member       1d 0H 59M 57S
##  8 Monday    docked_bike   casual       57d 21H 42M 35S
##  9 Monday    electric_bike casual       7H 59M 54S
## 10 Monday    electric_bike member       8H 1M 29.7119998931885S
## # i 25 more rows
```

Average ride duration by day of rental, product type and type of user

```
## # A tibble: 35 x 4
## # Groups:   start_day, product_type [21]
##    start_day product_type  type_of_user average_ride_duration
##    <ord>     <chr>         <chr>        <Period>
##  1 Sunday    classic_bike  casual       41M 36.4315602774195S
##  2 Sunday    classic_bike  member       16M 33.8592735327019S
##  3 Sunday    docked_bike   casual       4H 42M 25.223217361352S
##  4 Sunday    electric_bike casual       16M 45.4004729367989S
##  5 Sunday    electric_bike member       12M 28.9547843166584S
##  6 Monday    classic_bike  casual       36M 22.6802720707437S
##  7 Monday    classic_bike  member       13M 56.8256189116485S
##  8 Monday    docked_bike   casual       5H 20M 1.07295504790O9S
##  9 Monday    electric_bike casual       13M 53.7338522666333S
## 10 Monday    electric_bike member       10M 44.2023040087619S
## # i 25 more rows
```

Median ride duration by day of rental, product type and type of user

```
## # A tibble: 35 x 4
## # Groups:   start_day, product_type [21]
##    start_day product_type  type_of_user median_ride_duration
##    <ord>     <chr>         <chr>        <Period>
##  1 Sunday    classic_bike  casual       18M 7.364500164985665
##  2 Sunday    classic_bike  member       10M 8S
##  3 Sunday    docked_bike   casual       31M 42S
##  4 Sunday    electric_bike casual       11M 13.8199999332428S
##  5 Sunday    electric_bike member       8M 53.8759999275208S
##  6 Monday    classic_bike  casual       14M 54.3090001344681S
```

```
##  7 Monday    classic_bike  member       8M 43S
##  8 Monday    docked_bike   casual       26M 14S
##  9 Monday    electric_bike casual       9M 13S
## 10 Monday    electric_bike member       8M 0S
## # i 25 more rows
```

Minimum ride duration by month of rental, product type and type of user

```
## # A tibble: 49 x 4
## # Groups:   start_month, product_type [25]
##    start_month product_type  type_of_user minimum_ride_duration
##    <ord>       <chr>         <chr>        <Period>
##  1 January     classic_bike  casual       1S
##  2 January     classic_bike  member       1S
##  3 January     electric_bike casual       1S
##  4 January     electric_bike member       1S
##  5 February    classic_bike  casual       1S
##  6 February    classic_bike  member       1S
##  7 February    electric_bike casual       1S
##  8 February    electric_bike member       1S
##  9 March       classic_bike  casual       1S
## 10 March       classic_bike  member       1S
## # i 39 more rows
```

Maximum ride duration by month of rental, product type and type of user

```
## # A tibble: 49 x 4
## # Groups:   start_month, product_type [25]
##    start_month product_type  type_of_user maximum_ride_duration
##    <ord>       <chr>         <chr>        <Period>
##  1 January     classic_bike  casual       1d 0H 59M 57S
##  2 January     classic_bike  member       1d 0H 59M 57S
##  3 January     electric_bike casual       7H 59M 54S
##  4 January     electric_bike member       7H 59M 58S
##  5 February    classic_bike  casual       1d 1H 0M 29S
##  6 February    classic_bike  member       1d 1H 0M 29S
##  7 February    electric_bike casual       7H 59M 59S
##  8 February    electric_bike member       8H 0M 27S
```

```
##  9 March       classic_bike  casual       1d 1H 59M 56S
## 10 March       classic_bike  member       1d 1H 59M 48S
## # i 39 more rows
```

Average ride duration by month of rental, product type and type of user

```
## # A tibble: 49 x 4
## # Groups:   start_month, product_type [25]
##    start_month product_type  type_of_user average_ride_duration
##    <ord>       <chr>         <chr>        <Period>
##  1 January     classic_bike  casual       37M 28.6898861352979S
##  2 January     classic_bike  member       17M 10.3426625293273S
##  3 January     electric_bike casual       9M 14.9178277956413S
##  4 January     electric_bike member       9M 41.8886759421358S
##  5 February    classic_bike  casual       34M 28.7143473570659S
##  6 February    classic_bike  member       14M 5.663321451746365
##  7 February    electric_bike casual       11M 50.7487982632966S
##  8 February    electric_bike member       10M 49.5640516698173S
##  9 March       classic_bike  casual       38M 8.810099734881995
## 10 March       classic_bike  member       13M 54.3280045517951S
## # i 39 more rows
```

Median ride duration by month of rental, product type and type of user

```
## # A tibble: 49 x 4
## # Groups:   start_month, product_type [25]
##    start_month product_type  type_of_user median_ride_duration
##    <ord>       <chr>         <chr>        <Period>
##  1 January     classic_bike  casual       8M 57S
##  2 January     classic_bike  member       7M 32S
##  3 January     electric_bike casual       6M 40S
##  4 January     electric_bike member       6M 45S
##  5 February    classic_bike  casual       11M 12S
##  6 February    classic_bike  member       8M 5S
##  7 February    electric_bike casual       7M 54S
##  8 February    electric_bike member       7M 44S
##  9 March       classic_bike  casual       14M 49S
## 10 March       classic_bike  member       8M 31S
```

```
## # i 39 more rows
```

Minimum ride duration by month of rental, day of rental, product type and type of user

```
## # A tibble: 343 x 5
## # Groups:   start_month, start_day, product_type [175]
##    start_month start_day product_type  type_of_user minimum_ride_duration
##    <ord>       <ord>     <chr>         <chr>        <Period>
##  1 January     Sunday    classic_bike  casual       2S
##  2 January     Sunday    classic_bike  member       2S
##  3 January     Sunday    electric_bike casual       1S
##  4 January     Sunday    electric_bike member       1S
##  5 January     Monday    classic_bike  casual       1S
##  6 January     Monday    classic_bike  member       1S
##  7 January     Monday    electric_bike casual       1S
##  8 January     Monday    electric_bike member       1S
##  9 January     Tuesday   classic_bike  casual       4S
## 10 January     Tuesday   classic_bike  member       1S
## # i 333 more rows
```

Maximum ride duration by month of rental, day of rental, product type and type of user

```
## # A tibble: 343 x 5
## # Groups:   start_month, start_day, product_type [175]
##    start_month start_day product_type  type_of_user maximum_ride_duration
##    <ord>       <ord>     <chr>         <chr>        <Period>
##  1 January     Sunday    classic_bike  casual       1d 0H 59M 56S
##  2 January     Sunday    classic_bike  member       1d 0H 59M 56S
##  3 January     Sunday    electric_bike casual       7H 59M 54S
##  4 January     Sunday    electric_bike member       7H 59M 56S
##  5 January     Monday    classic_bike  casual       1d 0H 59M 57S
##  6 January     Monday    classic_bike  member       1d 0H 59M 57S
##  7 January     Monday    electric_bike casual       4H 16M 4S
##  8 January     Monday    electric_bike member       7H 59M 51S
##  9 January     Tuesday   classic_bike  casual       1d 0H 59M 56S
## 10 January     Tuesday   classic_bike  member       1d 0H 59M 56S
## # i 333 more rows
```

Average ride duration by month of rental, day of rental, product type and type of user

```
## # A tibble: 343 x 5
## # Groups:   start_month, start_day, product_type [175]
##    start_month start_day product_type  type_of_user average_ride_duration
##    <ord>       <ord>     <chr>         <chr>        <Period>
##  1 January     Sunday    classic_bike  casual       32M 38.8991674375577S
##  2 January     Sunday    classic_bike  member       19M 6.90368492883613S
##  3 January     Sunday    electric_bike casual       10M 8.46010844306738S
##  4 January     Sunday    electric_bike member       10M 13.3416260579636S
##  5 January     Monday    classic_bike  casual       50M 48.7342823250297S
##  6 January     Monday    classic_bike  member       16M 41.4030272636095
##  7 January     Monday    electric_bike casual       9M 22.5771018166455S
##  8 January     Monday    electric_bike member       9M 39.686272283411S
##  9 January     Tuesday   classic_bike  casual       38M 4.446959896507S
## 10 January     Tuesday   classic_bike  member       18M 43.0291655166068S
## # i 333 more rows
```

Median ride duration by month of rental, day of rental, product type and type of user

```
## # A tibble: 343 x 5
## # Groups:   start_month, start_day, product_type [175]
##    start_month start_day product_type  type_of_user median_ride_duration
##    <ord>       <ord>     <chr>         <chr>        <Period>
##  1 January     Sunday    classic_bike  casual       9M 24S
##  2 January     Sunday    classic_bike  member       7M 26S
##  3 January     Sunday    electric_bike casual       6M 55S
##  4 January     Sunday    electric_bike member       6M 33S
##  5 January     Monday    classic_bike  casual       9M 19S
##  6 January     Monday    classic_bike  member       7M 33S
##  7 January     Monday    electric_bike casual       6M 44S
##  8 January     Monday    electric_bike member       6M 40S
##  9 January     Tuesday   classic_bike  casual       8M 39S
## 10 January     Tuesday   classic_bike  member       7M 40S
## # i 333 more rows
```

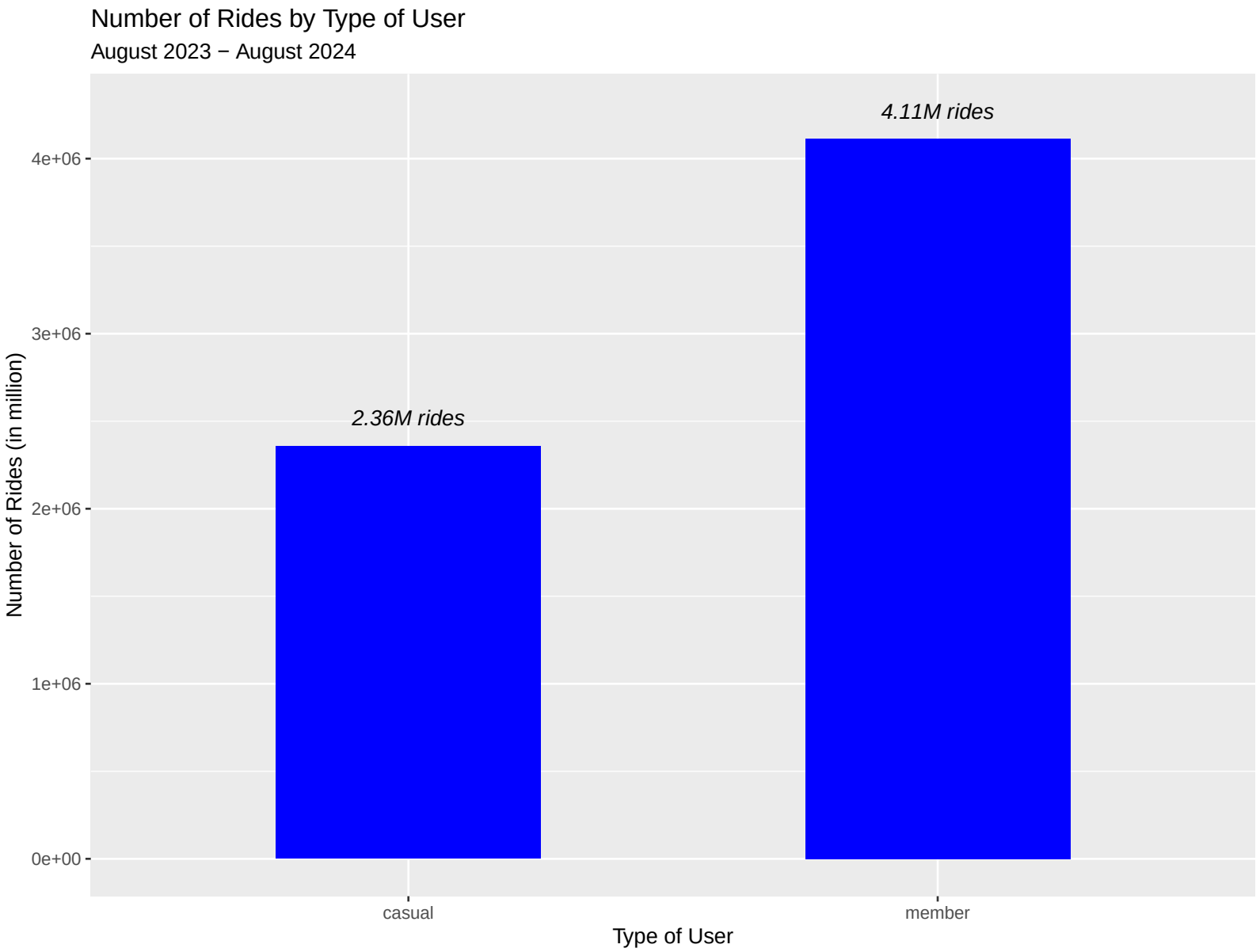
Step 6 - Create data visualizations

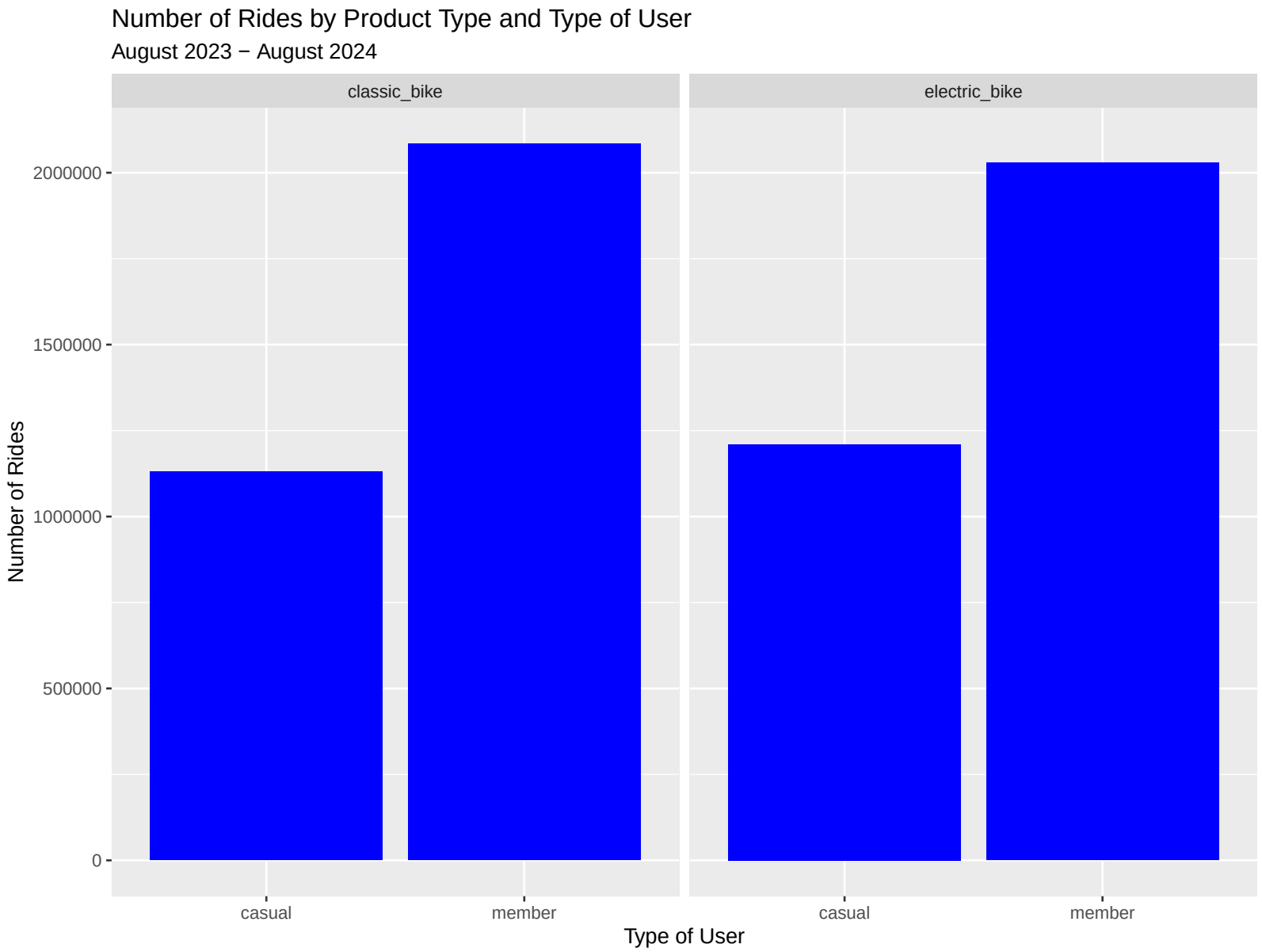
The number of docked_bike rides were significantly less compared to the other products. This made visualizing data about docked_bikes difficult. Therefore, to aid with visualization, the code below was used to create a dataframe for only **classic_bike** and **electric_bike**:

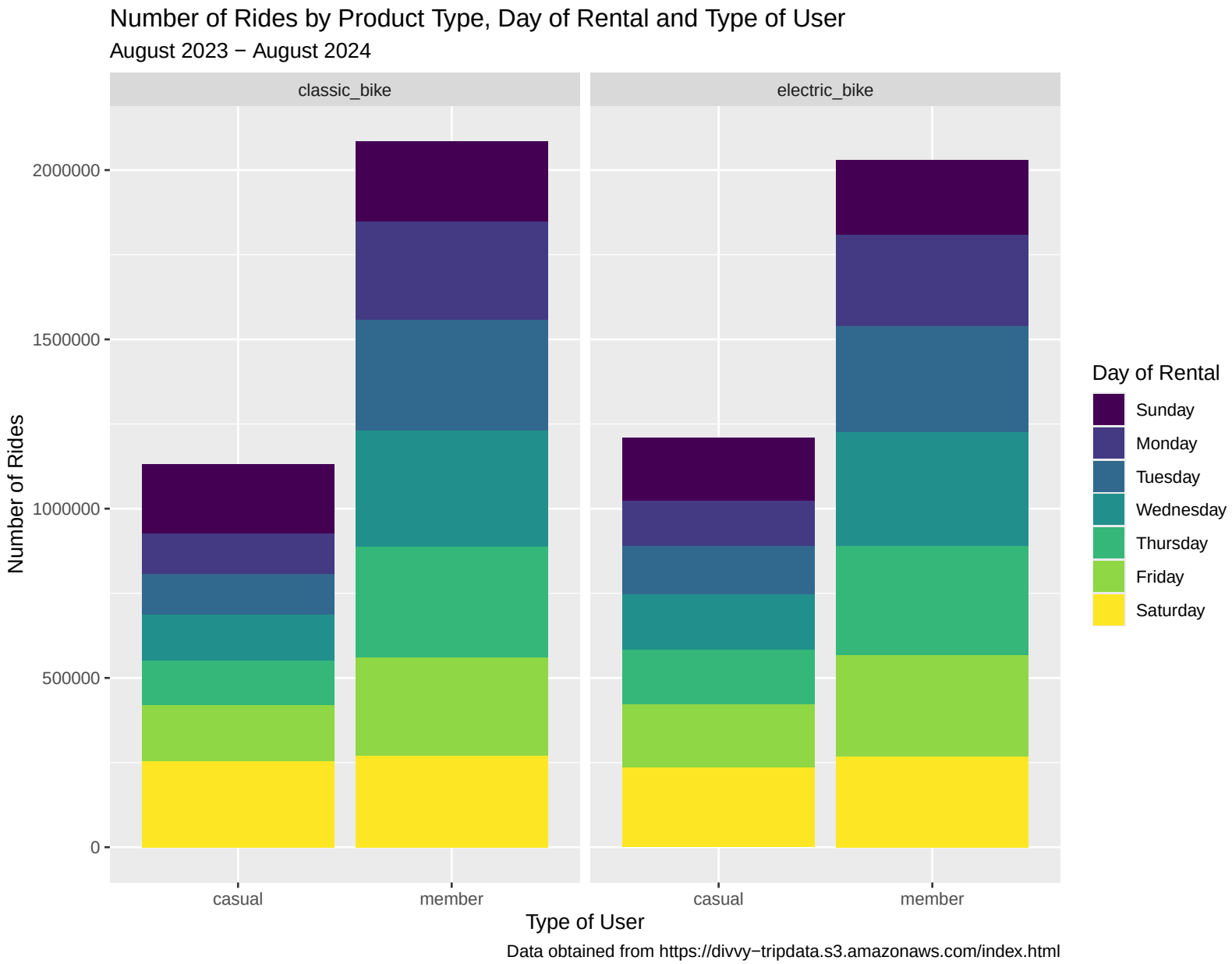
```
classic_and_electric <- user_month_data %>%
  filter(product_type!="docked_bike")
```

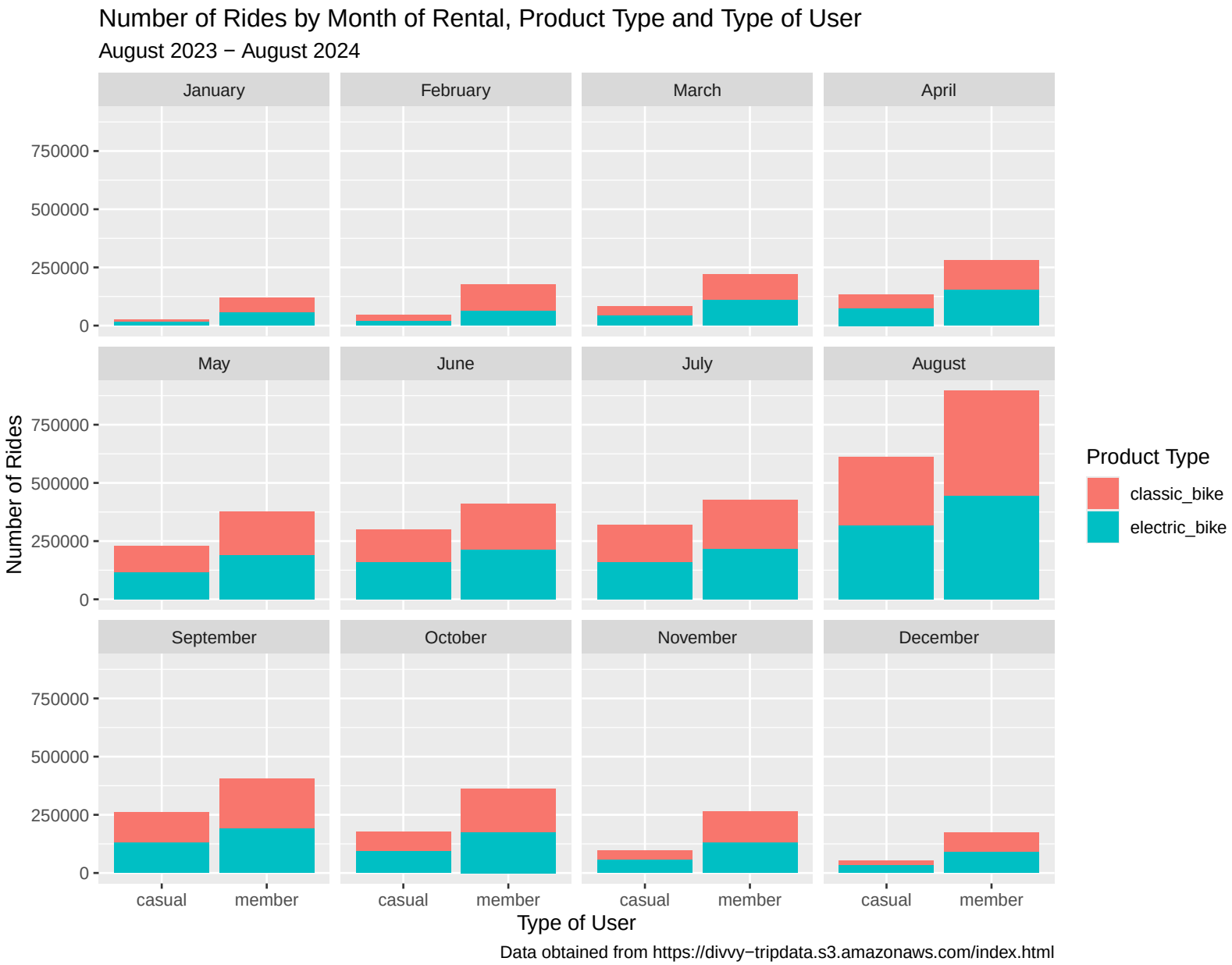
In addition to the above, to aid with visualization, the code below was used to create a dataframe for only **docked_bike**:

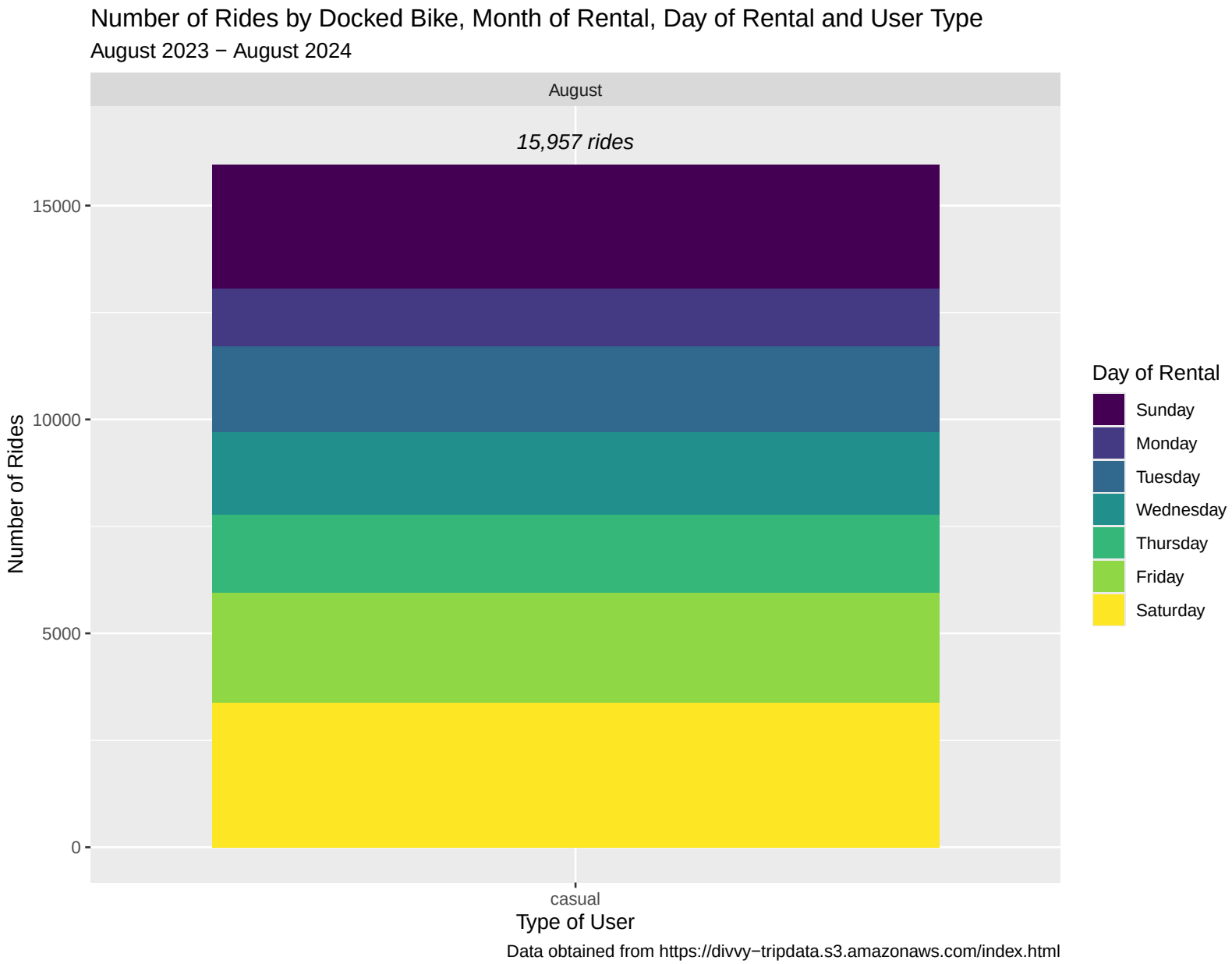
```
d_bike <- user_month_data %>%
  filter(product_type == "docked_bike")
```

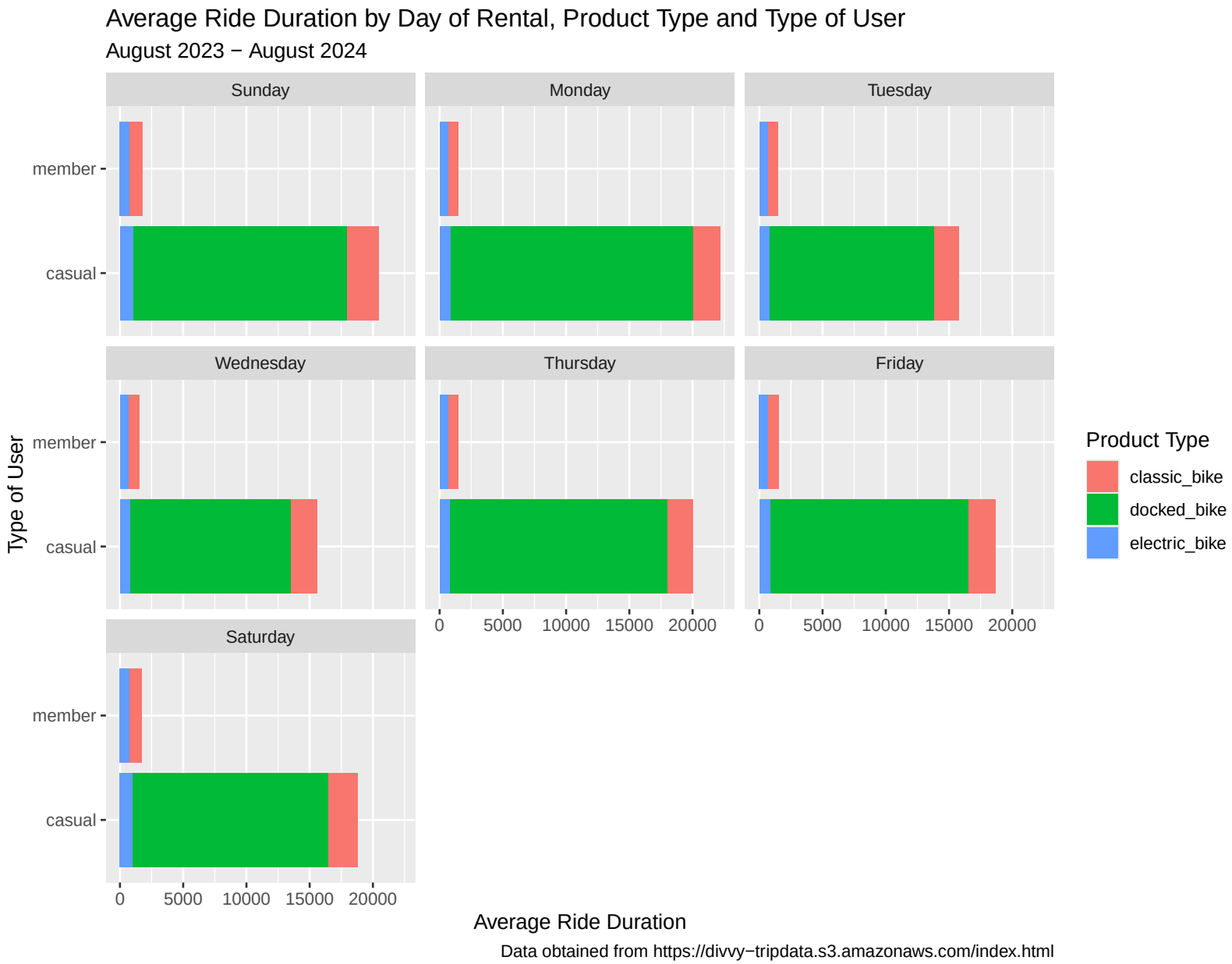

Number of Rides by Type of User
August 2023 − August 2024
4.11M rides
2.36M rides
4e+06
3e+06
2e+06
1e+06
0e+00
Number of Rides (in million)
casual
member
Type of User
Data obtained from https://divvy−tripdata.s3.amazonaws.com/index.html

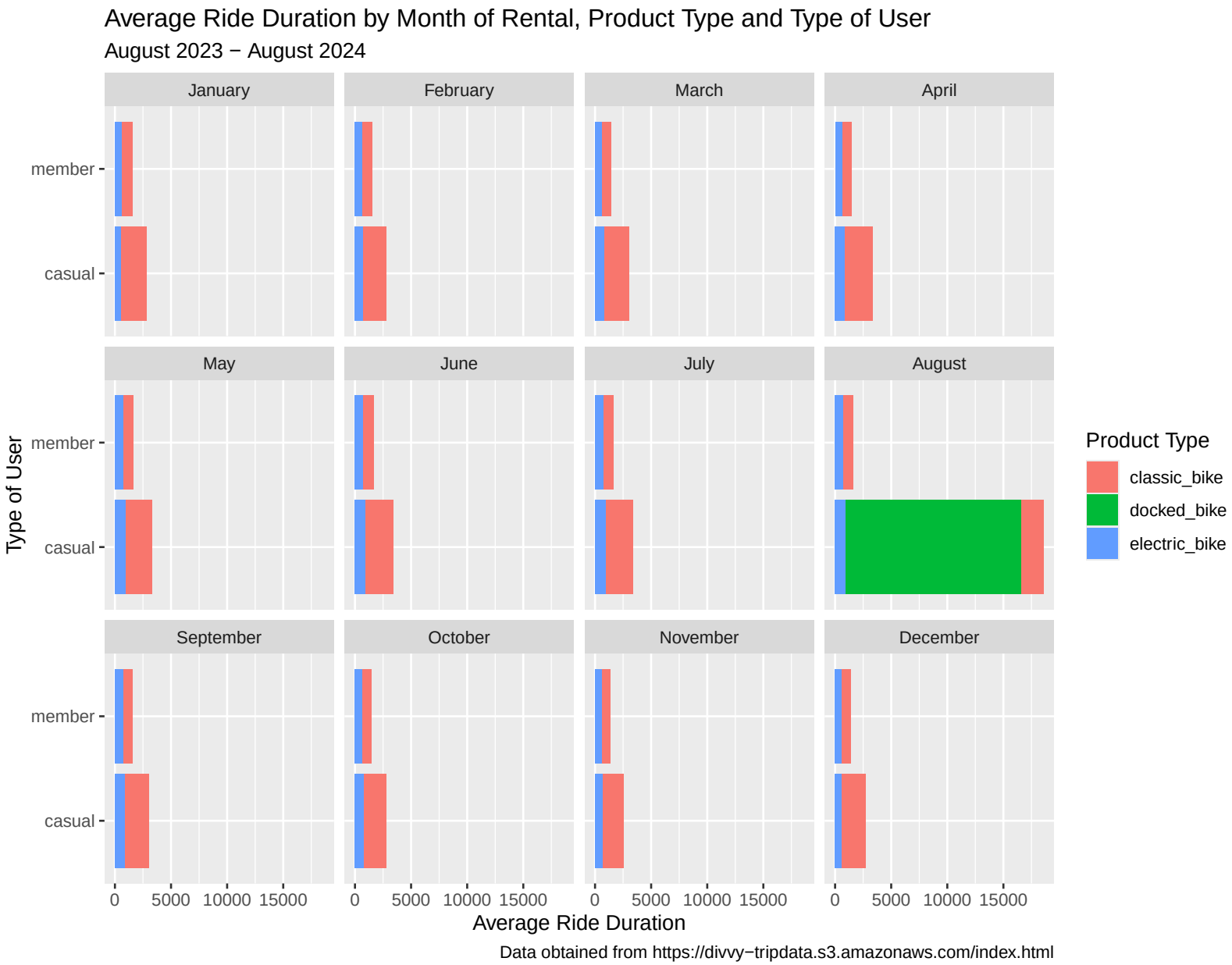

Number of Rides by Product Type and Type of User
August 2023 − August 2024
classic_bike
electric_bike
2000000
1500000
1000000
500000
0
Number of Rides
casual
member
casual
member
Type of User
Data obtained from https://divvy−tripdata.s3.amazonaws.com/index.html


Number of Rides by Product Type, Day of Rental and Type of User
August 2023 − August 2024
classic_bike
electric_bike
2000000
1500000
1000000
500000
0
Number of Rides
casual
member
casual
member
Type of User
Day of Rental
Sunday
Monday
Tuesday
Wednesday
Thursday
Friday
Saturday
Data obtained from https://divvy−tripdata.s3.amazonaws.com/index.html


Number of Rides by Month of Rental, Product Type and Type of User
August 2023 − August 2024
January
February
March
April
May
June
July
August
September
October
November
December
750000
500000
250000
0
Number of Rides
casual
member
Type of User
Product Type
classic_bike
electric_bike
Data obtained from https://divvy−tripdata.s3.amazonaws.com/index.html


Number of Rides by Docked Bike, Month of Rental, Day of Rental and User Type
August 2023 − August 2024
August
15,957 rides
15000
10000
5000
0
Number of Rides
casual
Type of User
Day of Rental
Sunday
Monday
Tuesday
Wednesday
Thursday
Friday
Saturday
Data obtained from https://divvy−tripdata.s3.amazonaws.com/index.html


Average Ride Duration by Day of Rental, Product Type and Type of User
August 2023 − August 2024
Sunday
Monday
Tuesday
Wednesday
Thursday
Friday
Saturday
member
casual
Type of User
0
5000
10000
15000
20000
Average Ride Duration
Product Type
classic_bike
docked_bike
electric_bike
Data obtained from https://divvy−tripdata.s3.amazonaws.com/index.html


Average Ride Duration by Month of Rental, Product Type and Type of User
August 2023 − August 2024
January
February
March
April
May
June
July
August
September
October
November
December
member
casual
Type of User
0
5000
10000
15000
Average Ride Duration
Product Type
classic_bike
docked_bike
electric_bike
Data obtained from https://divvy−tripdata.s3.amazonaws.com/index.html

Step 7 - List insights from data analysis

The following insights were gained from analysis of company data from Cyclistic for the period of August 2023 to August 2024:

1. There were 1.75 times more rides by subscription members than casual riders in the data analysed
2. There were more rides by subscription members than casual riders on all days of the week
3. There were 1.84 times more rides by subscription members who rode classic bikes than were casual riders
4. Casual riders were the only type of user that used docked bikes
5. There were 1.68 times more rides by subscription members who rode electric bikes than were casual riders
6. Subscription members rode more classic bikes and electric bikes than casual riders on all days of the week
7. The minimum ride duration for both subscription members and casual riders was 1 second
8. The longest ride duration in the data was for a casual rider
9. The average ride duration for casual riders was 2.07 times longer than that of subscription members
10. The average ride duration for casual riders was 1.49 times longer than the overall average ride duration
11. The average ride duration for subscription members was shorter than the overall average by a factor 0.72
12. On average, casual members rode classic bikes 2.52 times longer than subscription members
13. On average, casual members rode electric bikes 1.28 times longer than subscription members
14. On average, casual members rode classic bikes and electric bikes for a longer period of time than subscription members
15. The busiest months of the year, in terms of the most number of users, were May through September

Step 8 - Conclusions

1. There were more rides by subscription members than casual riders on all days of the week and all months of the year
2. Subscription members used more classic bikes and electric bikes than casual riders on all days of the week and all months of the year
3. On average, casual riders used classic bikes and electric bikes for longer periods of time than subscription members