Bike analysis 2023

Alexandre Ferreira

2024-01-20

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v forcats   1.0.0     v readr     2.1.5
## v ggplot2   3.4.4     v stringr   1.5.1
## v lubridate 1.9.3     v tibble    3.2.1
## v purrr     1.0.2     v tidyr     1.3.0
```

```
## -- 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 error
```

```
library(tidyr)
library(ggplot2)
```

About the problem

In 2016, Cyclistic launched a successful bike-share offering. Since then, the program has grown to a fleet of 5,824 bicycles that are geotracked and locked into a network of 692 stations acrros Chicago.

The customers are divided between members (annual memberships) and casual (single-ride passes and full-day passes).

The primary problem is to design marketing strategies to convert casual riders into annual members.

The business task is to analyze Cyclistic's historical bike trip data to discern patterns differentiating annual members from casual riders.

About the Data

The data is available as public datasets provided by *Motivate International Inc.*, representing 12 months of Cyclistic trip data. Available here.

It was collected data from **January/2023 to December/2023**.

All 12 .CSV files contain the same structure. They have these columns:

1. ride_id
2. rideable_type
3. started_at
4. ended_at
5. start_station_name
6. start_station_id
7. end_station_name
8. end_station_id
9. start_lat
10. start_lng
11. end_lat
12. end_lng
13. member_casual

Analysis

Setting the working directory to the location of the CSV files

```
setwd("D:/codigos/R/DATA")
```

As we have 12 distinct tables, let's create a list with the name of each CSV file to use later.

```
# Create a list of all CSV files in the directory
file_list <- list.files(pattern = "*.csv")
```

Print the List of Files

```
# File list
print(file_list)
```

```
##  [1] "202301_tripdata.csv" "202302_tripdata.csv" "202303_tripdata.csv"
##  [4] "202304_tripdata.csv" "202305_tripdata.csv" "202306_tripdata.csv"
##  [7] "202307_tripdata.csv" "202308_tripdata.csv" "202309_tripdata.csv"
## [10] "202310_tripdata.csv" "202311_tripdata.csv" "202312_tripdata.csv"
```

Combining files

1. Create the Dataframe **"bike_trip"**;
2. **"do.call"** is a function that applies another function to a list of arguments;
3. **"rbind"** which binds data frames row-wise
4. **"lapply"**, the result of it is a list of data frames, each corresponding to a CSV file in **"file_list"**;
5. **"function(file)"** is defined here to read a CSV file using **"read_csv"**;
6. **"stringAsFactors"** as **FALSE** to ensure that character columns are not automatically converted to factors.

```
# Read each file and combine them
bike_trip <- do.call(rbind, lapply(file_list, function(file) {
  read.csv(file, stringsAsFactors = FALSE)
}))
```

Preview the Dataset Viewing the first elements of the dataframe.

```
# Print a preview
head(bike_trip)
```

```
##            ride_id rideable_type          started_at            ended_at
## 1 F96D5A74A3E41399 electric_bike 2023-01-21 20:05:42 2023-01-21 20:16:33
## 2 13CB7EB698CEDB88  classic_bike 2023-01-10 15:37:36 2023-01-10 15:46:05
## 3 BD88A2E670661CE5 electric_bike 2023-01-02 07:51:57 2023-01-02 08:05:11
## 4 C90792D034FED968  classic_bike 2023-01-22 10:52:58 2023-01-22 11:01:44
## 5 3397017529188E8A  classic_bike 2023-01-12 13:58:01 2023-01-12 14:13:20
## 6 58E68156DAE3E311 electric_bike 2023-01-31 07:18:03 2023-01-31 07:21:16
##              start_station_name start_station_id              end_station_name
## 1   Lincoln Ave & Fullerton Ave     TA1309000058      Hampden Ct & Diversey Ave
## 2         Kimbark Ave & 53rd St     TA1309000037        Greenwood Ave & 47th St
## 3        Western Ave & Lunt Ave           RP-005 Valli Produce - Evanston Plaza
## 4         Kimbark Ave & 53rd St     TA1309000037        Greenwood Ave & 47th St
## 5         Kimbark Ave & 53rd St     TA1309000037        Greenwood Ave & 47th St
## 6 Lakeview Ave & Fullerton Pkwy     TA1309000019      Hampden Ct & Diversey Ave
##   end_station_id start_lat start_lng  end_lat   end_lng member_casual
## 1       202480.0  41.92407 -87.64628 41.93000 -87.64000        member
## 2   TA1308000002  41.79957 -87.59475 41.80983 -87.59938        member
## 3            599  42.00857 -87.69048 42.03974 -87.69941        casual
## 4   TA1308000002  41.79957 -87.59475 41.80983 -87.59938        member
## 5   TA1308000002  41.79957 -87.59475 41.80983 -87.59938        member
## 6       202480.0  41.92607 -87.63886 41.93000 -87.64000        member
```

```
# Structure of the dataset
str(bike_trip)
```

```
## 'data.frame':    5719877 obs. of  13 variables:
##  $ ride_id           : chr  "F96D5A74A3E41399" "13CB7EB698CEDB88" "BD88A2E670661CE5" "C90792D034FED9
##  $ rideable_type     : chr  "electric_bike" "classic_bike" "electric_bike" "classic_bike" ...
##  $ started_at        : chr  "2023-01-21 20:05:42" "2023-01-10 15:37:36" "2023-01-02 07:51:57" "2023-
##  $ ended_at          : chr  "2023-01-21 20:16:33" "2023-01-10 15:46:05" "2023-01-02 08:05:11" "2023-
##  $ start_station_name: chr  "Lincoln Ave & Fullerton Ave" "Kimbark Ave & 53rd St" "Western Ave & Lun
##  $ start_station_id  : chr  "TA1309000058" "TA1309000037" "RP-005" "TA1309000037" ...
##  $ end_station_name  : chr  "Hampden Ct & Diversey Ave" "Greenwood Ave & 47th St" "Valli Produce - E
##  $ end_station_id    : chr  "202480.0" "TA1308000002" "599" "TA1308000002" ...
##  $ start_lat         : num  41.9 41.8 42 41.8 41.8 ...
##  $ start_lng         : num  -87.6 -87.6 -87.7 -87.6 -87.6 ...
##  $ end_lat           : num  41.9 41.8 42 41.8 41.8 ...
##  $ end_lng           : num  -87.6 -87.6 -87.7 -87.6 -87.6 ...
##  $ member_casual     : chr  "member" "member" "casual" "member" ...
```

```
# Check missing datas
colSums(is.na(bike_trip))
```

Checking missing data

```
##            ride_id      rideable_type         started_at           ended_at
##                  0                  0                  0                  0
## start_station_name   start_station_id   end_station_name     end_station_id
##                  0                  0                  0                  0
##          start_lat          start_lng            end_lat            end_lng
##                  0                  0               6990               6990
##      member_casual
##                  0
```

We can observe that only **2 columns have missing values**, namely, the **'end_lat'** and **'end_lng'** columns.

If these columns are important for the analysis, we can handle them. If they don't affect the analysis, that's not a problem.

Checking suplicates

1. Creating column: Ride Length

We will create a column named 'ride_length,' which will be the result of the following equation:

ended_at - started_at

I have converted the 'started_at' and 'ended_at' columns to Datetime format using the **'lubridate'** library.

```
# Creating column: ride_length
library(lubridate)
bike_trip$started_at = ymd_hms(bike_trip$started_at)
bike_trip$ended_at = ymd_hms(bike_trip$ended_at)

bike_trip$ride_length <- (bike_trip$ended_at - bike_trip$started_at)
```

2. Creating column: Day of Week

We have created a 'day_of_week' column that represents the days of the week, numbered as follows:

- 1 = Sunday
- 2 = Monday
- 3 = Tuesday
- 4 = Wednesday
- 5 = Thursday
- 6 = Friday
- 7 = Saturday

```
# Day of Week (Sun:1, Mon:2, ... Sat: 7)
bike_trip$day_of_week <- wday(bike_trip$started_at, week_start = 1)
```

3. Calculate the AVERAGE of ride_length

```
# Calculate the mean of ride_length
mean(bike_trip$ride_length)
```

```
## Time difference of 1090.159 secs
```

That means the average is **1090 seconds**, which, when converted to minutes, is **18 minutes**.

3. Calculate the MAX of ride_length

```
# Calculate the max ride_length
max(bike_trip$ride_length)
```

```
## Time difference of 5909344 secs
```

The maximum is 5,909,344 seconds, which represents a total of approximately **68 days**.

3. Calculate the MODE of day_of_week

```
# Calculate the mode of week_day
bike_trip %>%
```

```
  count(day_of_week) %>%
  arrange(desc(n)) %>%
  slice(1) %>%
  pull(day_of_week)
```

```
## [1] 6
```

The day of the week that appears the most is 6, which represents Friday.

We can see below the values corresponding to each day of the week:

```
bike_trip %>%
  group_by(day_of_week) %>%
  summarise(total = n()) %>%
  arrange(desc(total))
```

```
## # A tibble: 7 x 2
##   day_of_week  total
##         <dbl>  <int>
## 1           6 883566
## 2           4 860202
## 3           5 843524
## 4           3 835625
## 5           2 822978
## 6           7 744578
## 7           1 729404
```

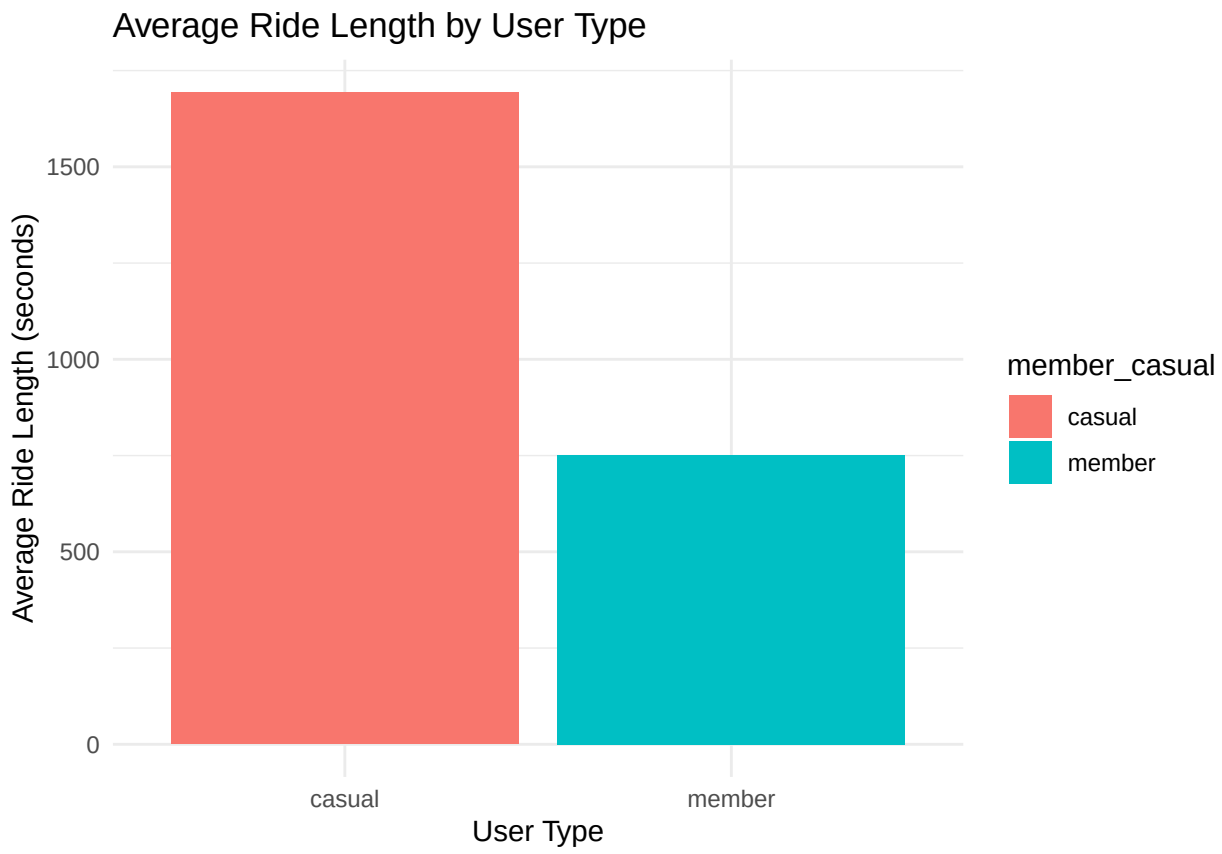
4. Compare the Average: Casual x Members

We can observe that casual users have an average that is 2.2 times higher than members.

```
# Calculate the average ride_length by members and casual riders
bike_trip %>%
  group_by(member_casual) %>%
  summarize(media = mean(ride_length))
```

```
## # A tibble: 2 x 2
##   member_casual media
##   <chr>         <drtn>
## 1 casual        1693.4718 secs
## 2 member         750.7899 secs
```

```
bike_trip %>%
  group_by(member_casual) %>%
  summarize(media = mean(ride_length)) %>%
  # Convert 'media' from difftime to numeric (e.g., minutes)
  mutate(media_min = as.numeric(media, units = "secs")) %>%
  # Creating the plot
  ggplot(aes(x = member_casual, y = media_min, fill = member_casual)) +
  geom_bar(stat = "identity", position = position_dodge()) +
  labs(x = "User Type", y = "Average Ride Length (seconds)", title = "Average Ride Length by User Type"
  theme_minimal()
```



5. Compare the Day of the Week: Casual x Members

We can see that Friday is the busiest day for both user types.

```
# Calculate average by day_of_week and type of user
bike_trip %>%
  group_by(day_of_week, member_casual) %>%
  summarize(Average_ride_length = mean(ride_length)) %>%
  pivot_wider(names_from = member_casual, values_from = Average_ride_length)
```

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

```
## # A tibble: 7 x 3
## # Groups:   day_of_week [7]
##   day_of_week casual        member
##         <dbl> <drtn>        <drtn>
## 1           1 1662.811 secs 714.1153 secs
## 2           2 1504.998 secs 720.7540 secs
## 3           3 1455.630 secs 715.7563 secs
## 4           4 1483.497 secs 721.1627 secs
## 5           5 1635.647 secs 748.6873 secs
## 6           6 1928.233 secs 836.2612 secs
## 7           7 1965.455 secs 834.3816 secs
```

6. Compare the number of riders: Casual x Members

We can observe that the number of members is greater than that of casual users on all days.

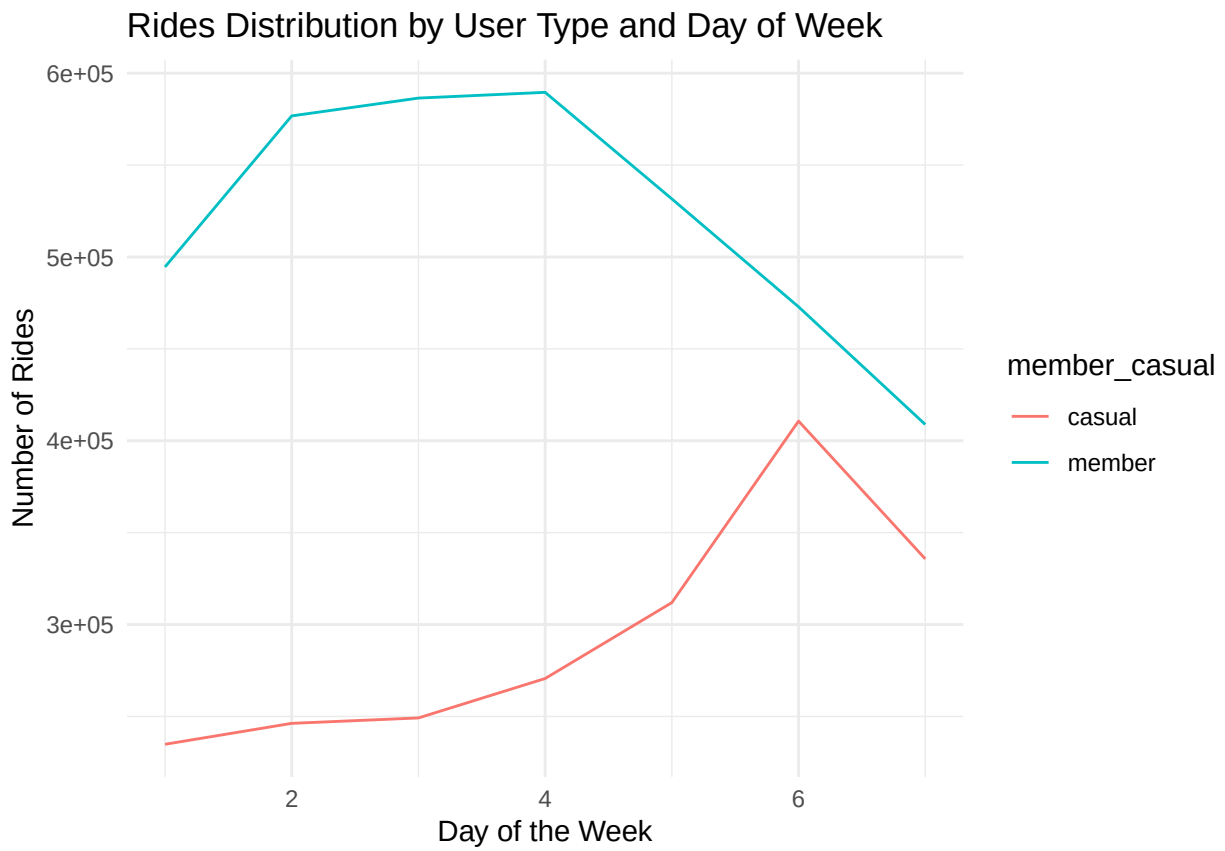
Above, we analyzed that the average for members is lower, which is likely due to the large number of subscribers.

```
# Calculate the number of rides by users
bike_trip %>%
  group_by(day_of_week, member_casual) %>%
  summarize(Count_of_rides = n()) %>%
  pivot_wider(names_from = member_casual, values_from = Count_of_rides)
```

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

```
## # A tibble: 7 x 3
## # Groups:   day_of_week [7]
##   day_of_week casual member
##         <dbl>  <int>  <int>
## 1           1 234828 494576
## 2           2 246224 576754
## 3           3 249166 586459
## 4           4 270612 589590
## 5           5 311925 531599
## 6           6 410706 472860
## 7           7 335718 408860
```

```
bike_trip %>%
  group_by(day_of_week, member_casual) %>%
  summarise(ride_count = n(), .groups = "drop") %>%
  ggplot(aes(x = day_of_week, y = ride_count, color = member_casual, group = member_casual)) +
  geom_line() +
  labs(x = "Day of the Week", y = "Number of Rides", title = "Rides Distribution by User Type and Day o
  theme_minimal()
```



7. Conclusions

Casual riders and members exhibit distinct patterns in their bike usage. Casual riders tend to have longer rides on average, possibly for leisure, while members show more consistent and frequent usage, indicative of regular commuting or routine use. Understanding these differences is crucial for Cyclistic in tailoring their services and marketing strategies to convert casual riders into members and to cater to the distinct needs of both groups.

Your team and business can apply these insights in several ways:

1. Targeted Marketing Strategies: Use the insight about casual riders' longer average ride lengths to develop targeted marketing campaigns focused on leisure and exploration aspects of biking.

2. Membership Conversion: Implement strategies to convert casual riders into members by emphasizing the benefits of membership for frequent riders, possibly through loyalty programs or discounts.

3. Service Customization: Tailor services to meet the differing needs of casual riders and members. For casual riders, consider offering guided tours or leisure-oriented experiences, while for members, focus on convenience and efficiency for regular commutes.

4. Operational Planning: Adjust bike availability and station services based on the usage patterns throughout the week to efficiently cater to the peak usage times of both member types.

By leveraging these insights, the business can enhance user satisfaction and potentially increase the conversion of casual riders to annual members, thereby boosting overall usage and profitability.