Cyclistic Data Analysis About The Differences of Casual Riders And Members

Pedro Henrique Warken Ramos

2024-01-10

1.Introduction

This data analysis report uses the historical data from all months of 2023 to determine differences between how casual riders and annual members use Cyclistic bikes differently. With this analysis it will be possible to create a market strategy to target casual riders and encourage them to buy the annual membership. This report contains all the code used to analyse the data provided along with data visualizations and explanations about the findings all the way from creating the data frame, cleaning the data and manipulating it. At the end of the report there will be the conclusions that can be made about the findings and a data driven market strategy to convert casual riders into annual members.

2.Data analysis walkthrough

2.1 Create a data frame with all cyclistic data from 2023

```
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v readr     2.1.4
## v forcats   1.0.0     v stringr   1.5.1
## v ggplot2   3.4.4     v tibble    3.2.1
## v lubridate 1.9.3     v tidyr     1.3.0
## 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 error
```

```
data_list = list()
for (i in 1:12) {
  if (i < 10) {
    file_name = paste0("20230", i, "-divvy-tripdata.csv")
  } else {
    file_name = paste0("2023", i, "-divvy-tripdata.csv")
  }
  data_list[[i]] = read.csv(file_name)
}
cyclistic_2023 = do.call(rbind, data_list)
rm(data_list)
```

2.2 Verifying the integrity of the data:

```
empty_counts = sapply(cyclistic_2023, function(x) sum(is.null(x) | x == ""))
```

With that is possible to conclude that there are lots of rows with missing values for the station names and ids (for the end and start), also end_lat and end_lng returned NA which should be further investigated.

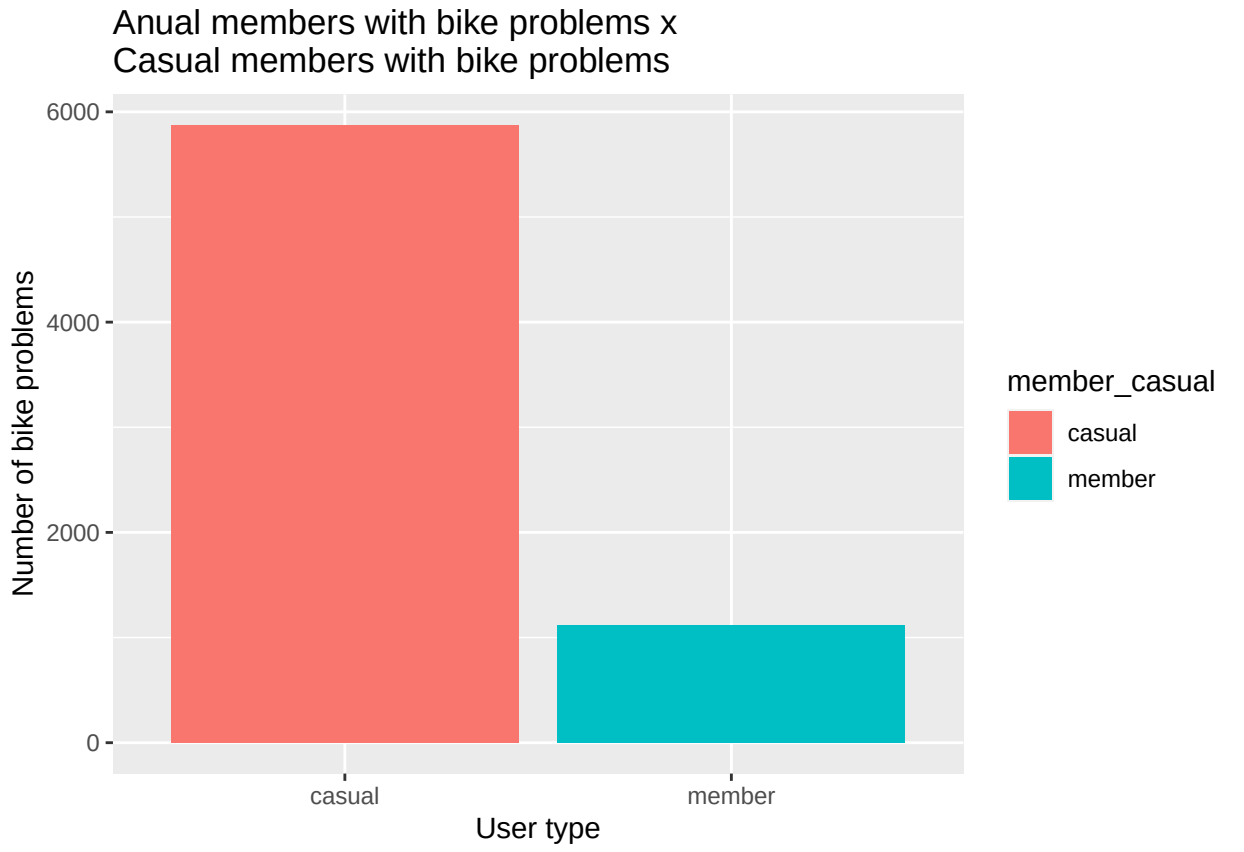
```
empty_counts = sapply(cyclistic_2023, function(x) sum(is.na(x)))
rm(empty_counts)
```

We can verify that there are several rows for which there is no end_lat and end_lng which are likely bikes that were broken or stolen before reaching their destinations:

```
na_end_positions = cyclistic_2023 %>%
  filter(is.na(end_lat))
```

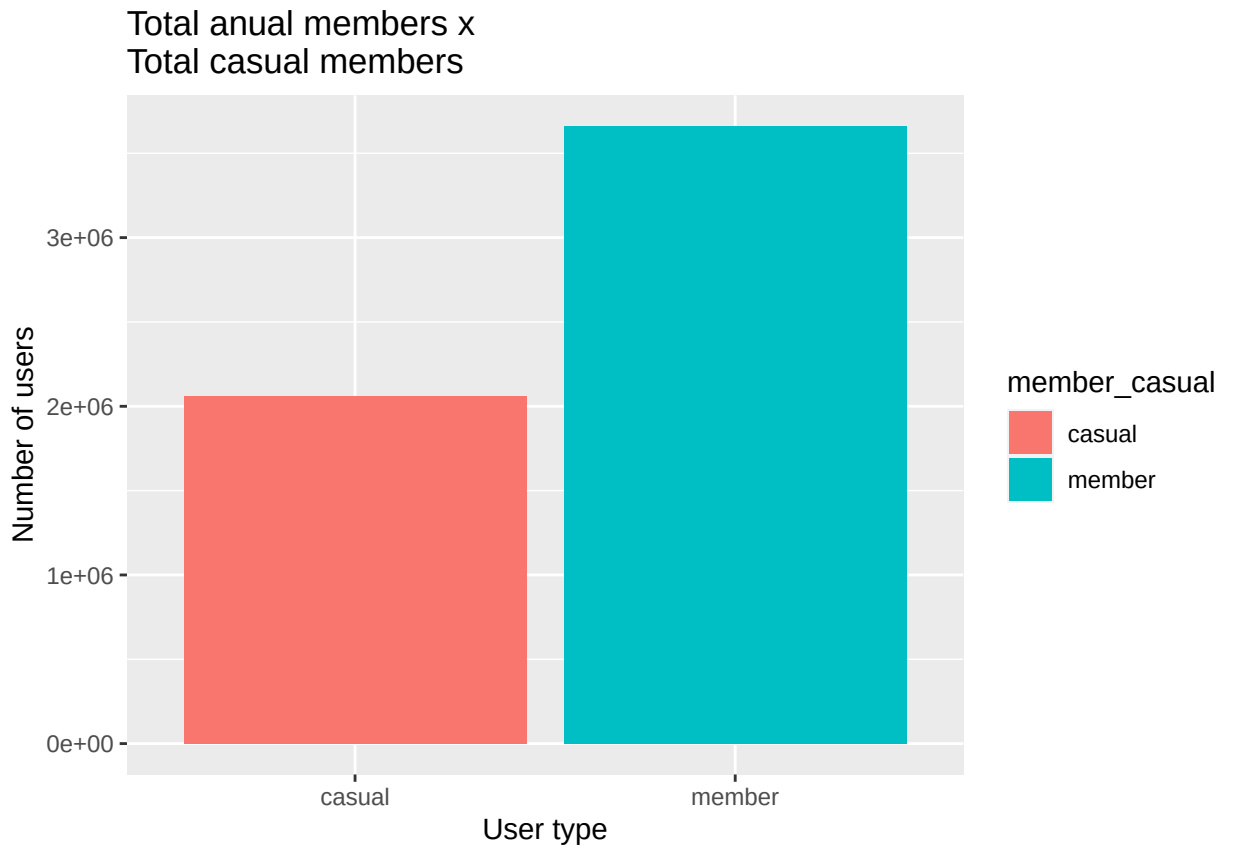
Graph comparing the amount of problematic bikes from casual riders x annual members against the total amount of each group:

```
ggplot(na_end_positions, aes(
  x = member_casual,
  fill=member_casual,
)) + geom_bar() + labs(
  x = "User type",
  y = "Number of bike problems",
  title="Anual members with bike problems x\nCasual members with bike problems")
```




```
rm(na_end_positions)
ggplot(cyclistic_2023, aes(
```

```
  x = member_casual,
  fill=member_casual
)) + geom_bar() + labs(
  x = "User type",
  y = "Number of users",
  title="Total anual members x\nTotal casual members"
  )
```



With these graphs is clear that more bike problems occur in casual riders than members, since there are more members in total but more casual riders with bike problems. Nevertheless, with the data provided it is not clear exactly what happens to the bikes or why, they might have broken, stolen or maybe the casual riders simply did not understood correctly how to retrieve the bikes.

2.3 Manipulating the data

To further discover differences between casual riders and members we can analyze calculated columns like total_trip_seconds and trip_distance_km, which represent the total amount of seconds in each trip and the total distance in kilometers between the starting station and the ending station respectively.

```
# Total trip seconds:
library(lubridate)
library(dplyr)
manipulated_data = cyclistic_2023
rm(cyclistic_2023)
manipulated_data$started_at = ymd_hms(manipulated_data$started_at)
manipulated_data$ended_at = ymd_hms(manipulated_data$ended_at)
```

```r
manipulated_data$total_trip_seconds = as.numeric(manipulated_data$ended_at -
                                                   manipulated_data$started_at)
# Convert the numeric column to the 'mm:ss' format for better visualization
manipulated_data = manipulated_data %>%
  mutate(formatted_time = seconds_to_period(total_trip_seconds) %>%
           as.character() %>%
           sprintf("%M:%S"))
```

```
## Warning: There was 1 warning in `mutate()`.
## i In argument: `formatted_time = `%>%`(...)`.
## Caused by warning in `sprintf()`:
## ! one argument not used by format
```

```r
# Add weekday
manipulated_data = manipulated_data %>%
  mutate(weekday_start = weekdays(started_at), weekday_end = weekdays(ended_at))

# Distance from bike retrieval to delivery:
# Function to calculate Haversine distance
haversine_distance = function(delta_latitude, delta_longitude) {
  # Radius of the Earth in kilometers
  earth_radius = 6371

  # Convert latitude and longitude differences to radians
  delta_lat_rad = delta_latitude * pi / 180
  delta_lon_rad = delta_longitude * pi / 180

  # Haversine formula
  a = sin(delta_lat_rad/2)^2 + cos((0) * pi / 180) *
    cos((0 + delta_latitude) * pi / 180) * sin(delta_lon_rad/2)^2
  c = 2 * atan2(sqrt(a), sqrt(1 - a))
  distance = earth_radius * c

  return(distance)
}

distance_time_trips = manipulated_data %>%
  # Removing unfinished trips since they were already analyzed previously
  filter(!is.na(end_lat) & !is.na(end_lng)) %>%
  mutate(delta_lat = abs(end_lat - start_lat),
         delta_lng = abs(end_lng - start_lng)) %>%
  mutate(trip_distance_km = haversine_distance(delta_lat, delta_lng))

rm(manipulated_data)
```

Preview of the first few rows of the manipulated data (only the columns added are being shown for simplicity)

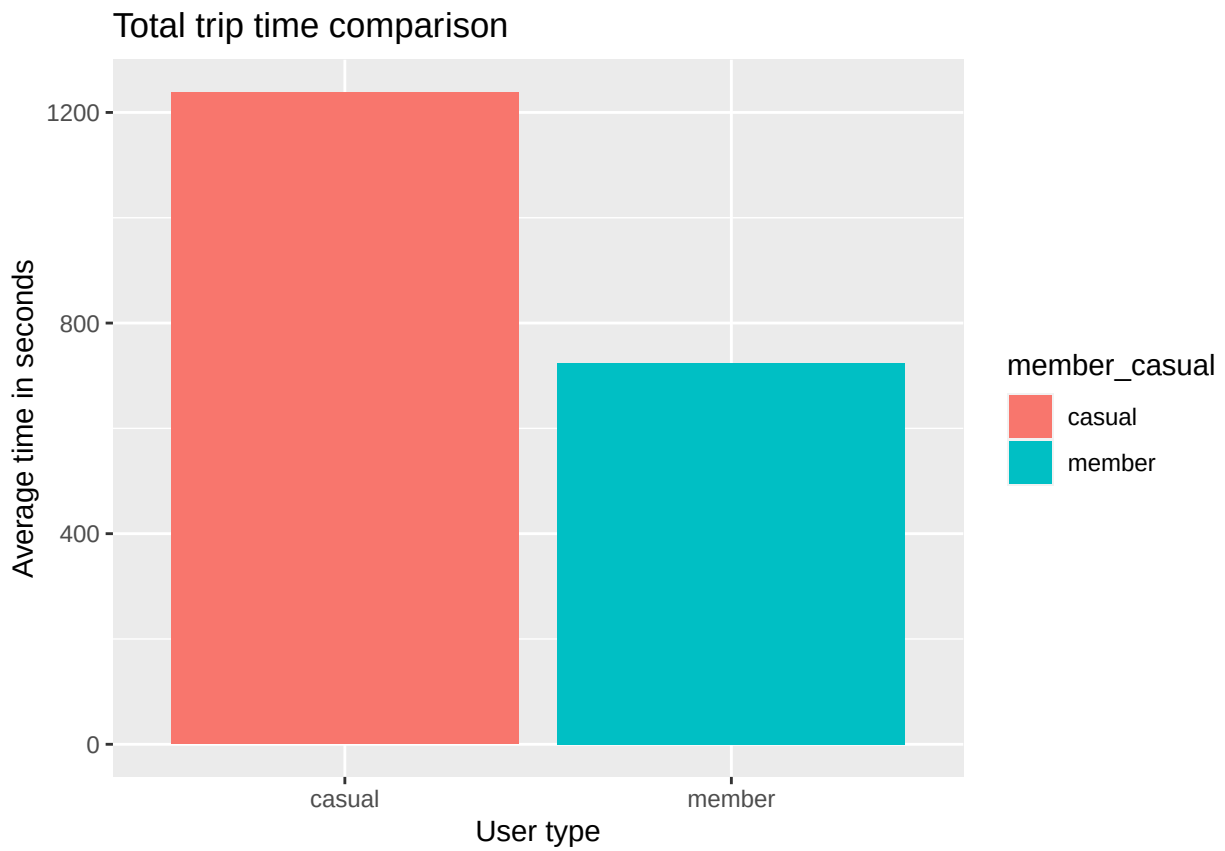
```r
head(distance_time_trips[, c(
  "total_trip_seconds", "formatted_time", "weekday_start", "weekday_end",
  "delta_lat", "delta_lng", "trip_distance_km"
  )])
```

```
##   total_trip_seconds formatted_time weekday_start weekday_end   delta_lat
## 1                651        10M 51S      Saturday    Saturday 0.005926065
## 2                509         8M 29S       Tuesday     Tuesday 0.010267000
```

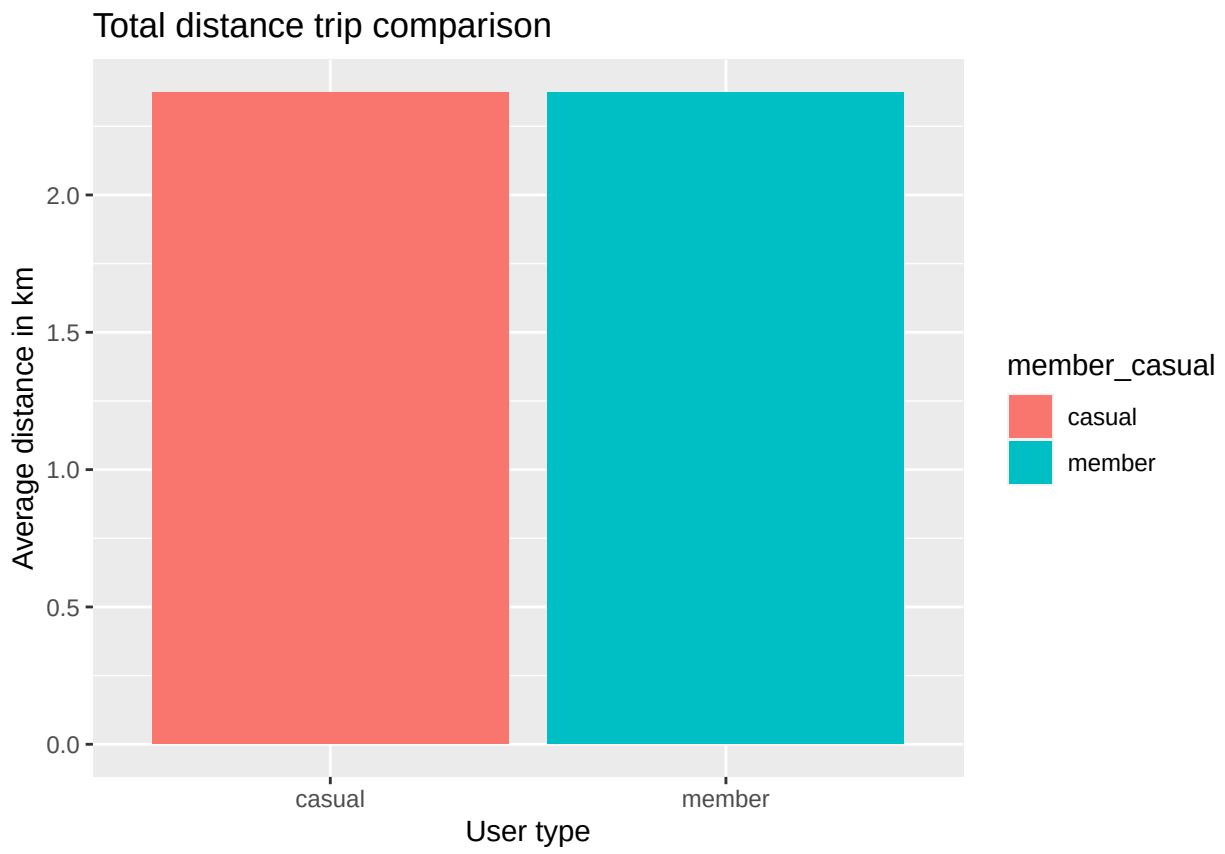
```
## 3                794        13M 14S        Monday      Monday 0.031171000
## 4                526         8M 46S        Sunday      Sunday 0.010267000
## 5                919        15M 19S      Thursday    Thursday 0.010267000
## 6                193         3M 13S       Tuesday     Tuesday 0.003931098
##     delta_lng trip_distance_km
## 1 0.006278381        0.9599949
## 2 0.004636000        1.2526284
## 3 0.008930167        3.6054929
## 4 0.004636000        1.2526284
## 5 0.004636000        1.2526284
## 6 0.001141801        0.4551832
```

2.4 Use additional calculated data to compare casual riders and members:

```
grouped_data = distance_time_trips %>%
  group_by(member_casual) %>%
  summarize(average_time = mean(total_trip_seconds))

ggplot(grouped_data, aes(
  x = member_casual,
  y = average_time,
  fill = member_casual
)) + geom_bar(stat = "identity") +
  labs(
    x = "User type",
    y = "Average time in seconds",
    title="Total trip time comparison"
    )
```

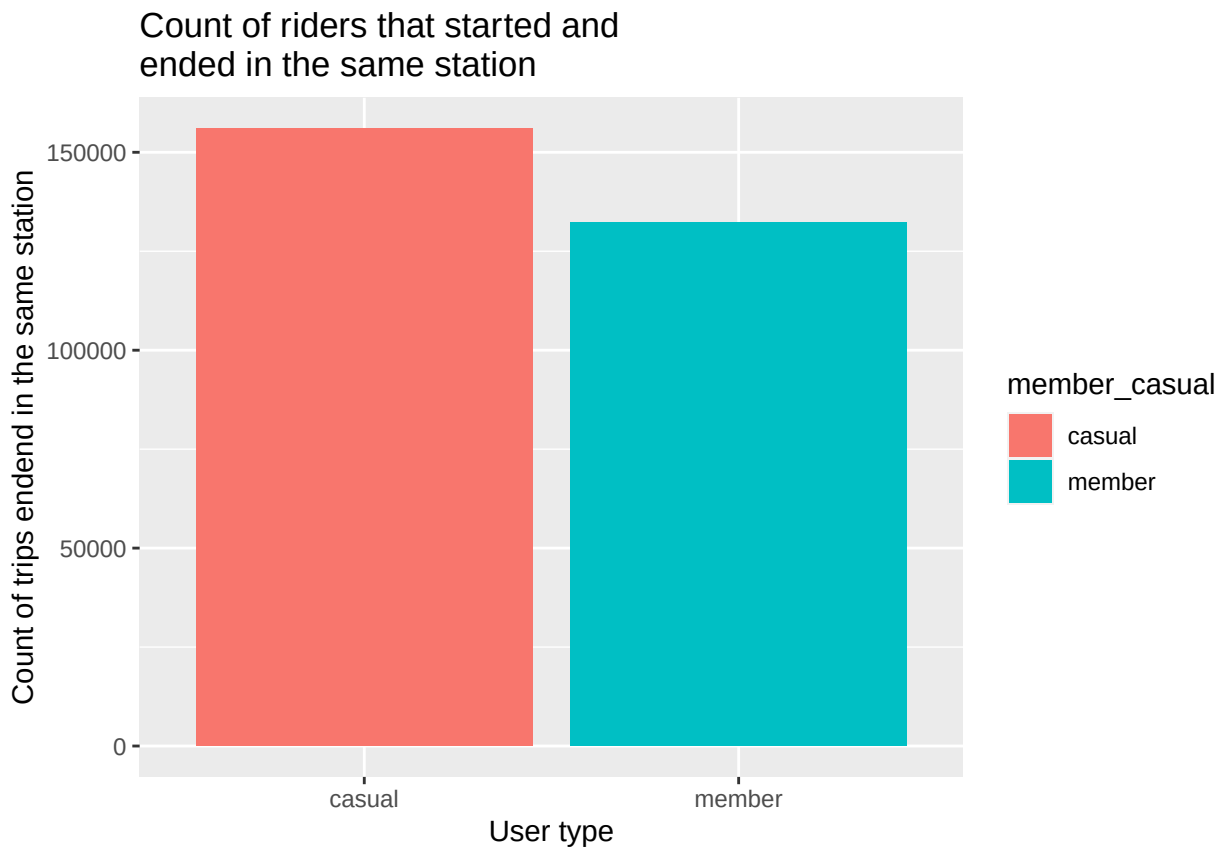



```
grouped_data = distance_time_trips %>%
  group_by(member_casual) %>%
  summarize(average_distance = mean(trip_distance_km))

ggplot(grouped_data, aes(
  x = member_casual,
  y = average_distance,
  fill = member_casual
)) + geom_bar(stat = "identity") +
  labs(
    x = "User type",
    y = "Average distance in km",
    title = "Total distance trip comparison"
    )
```




```
rm(grouped_data)
```

```
trips_0_distance = distance_time_trips %>%
  filter(trip_distance_km == 0)

ggplot(trips_0_distance, aes(
  x = member_casual,
  fill = member_casual
)) + geom_bar() + labs(
  x = "User type",
  y = "Count of trips endend in the same station",
  title = "Count of riders that started and\nended in the same station"
  )
```




```
rm(trips_0_distance)
```

Comparison of trips that started on weekdays between groups:

```
trips_in_weekdays = distance_time_trips %>%
  group_by(member_casual) %>%
  summarize(
    total_trips = n(),  # Count total trips
    weekday_trips = sum(!(weekday_start %in% c("Saturday", "Sunday"))),
    weekend_trips = sum(weekday_start %in% c("Saturday", "Sunday"))
  )
rm(distance_time_trips)
print(trips_in_weekdays)
```

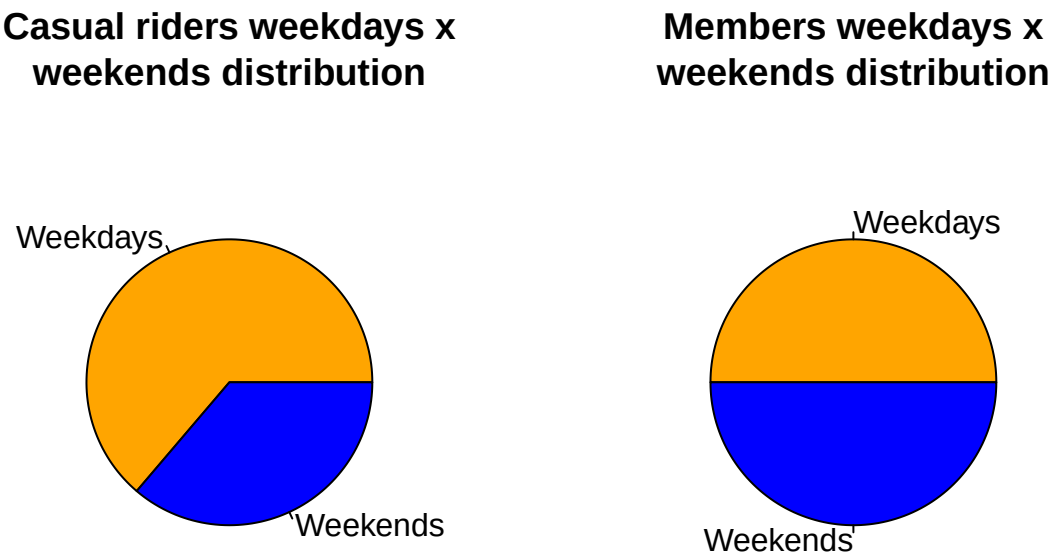
```
## # A tibble: 2 x 4
##   member_casual total_trips weekday_trips weekend_trips
##   <chr>               <int>         <int>         <int>
## 1 casual            2053307       1309367        743940
## 2 member            3659580       2778169        881411
```

```
casual_data = subset(trips_in_weekdays, member_casual == "casual")
member_data = subset(trips_in_weekdays, member_casual == "member")

print(trips_in_weekdays)
```

```
## # A tibble: 2 x 4
##   member_casual total_trips weekday_trips weekend_trips
##   <chr>               <int>         <int>         <int>
```

```
## 1 casual            2053307       1309367        743940
## 2 member            3659580       2778169        881411
```

```
casual_data = c(casual_data$weekday_trips, casual_data$weekend_trips)
member_data = c(member_data$weekday_trips, member_data$weekday_trips)
labels =
  c("Weekdays", "Weekends")
colors = c("orange", "blue")

par(mfrow = c(1, 2))
pie(
  casual_data,
  labels = labels,
  col = colors,
  main = "Casual riders weekdays x\nweekends distribution")
pie(member_data,
    labels = labels,
    col = colors,
    main = "Members weekdays x\nweekends distribution")
```



3. Conclusions and market strategy propose

Analyzing the first two graphs it is clear that more casual riders have problems with the bikes than annual members although there are more annual members in total. Although the problem that happens with the bikes is unknown with the data provided it is likely a cause for the casual riders not wanting to continue using the app and buying an annual subscription. This could be remedied by inserting clear instructions in

the app and in each bike station of how to get the bike, retrieve it and use it.

In the comparison of average time spent for each group the casual riders used the bikes considerably longer the annual members. This is likely because many casual riders buy the single-ride pass, thus they spend more time in the bikes to better harness the amount payed, since the charge is by ride and not by time spent, whereas annual riders are can get as many bikes as they want because they already payed for it. For the comparison of average distance between groups there wasn't a significant difference enough to draw any conclusions about it.

About the analysis of rides that started and ended in the same station there is a small difference between each group, which likely just indicates that the users are testing the bike out for the first time or just riding for fun which is more common in casual riders.

The last graph is the most significant finding. Opposite to what might be expected, there are more casual riders using the bikes in weekdays than annual members. Since it is more common for people to use bikes for fun in weekends it indicates that there are several casual riders that use the bikes for locomotion for which an annual subscription would be more appropriate. This indicates two possible options: the users fear committing for such a long time or the annual subscription does not present a good enough discount compared to casual rides. This could be solved with a cancel and refund policy in the first three months of use for example, solving the first issue and also implementing cheaper annual plans that can be used only in weekdays, for example, or for a limited amount of days per week, solving the second issue.

Using the data-driven marketing campaigns presented in this report the Cyclistic company will successfully be able to convert a vast amount of casual riders into annual members providing becoming more lucrative and offering a better service to their clients.