R Notebook

Code

The first thing to do it to find out how the hell to analyse data

Hide

```
library(corrplot)
library(tidyverse)
library(ggplot2)
library(maps)
library(ggmap)
library(tmap)
library(geosphere)
library(lubridate)
library(sf)
library(dplyr)
library(lubridate)
```

Hide

```
trips_df <- read_csv("../data/trips_data_cleaned.csv")
```

```
Rows: 6451628 Columns: 14
── Column specification ────────────────────────────────────────────────────────
Delimiter: ","
chr  (7): ride_id, rideable_type, start_station_name, start_station_id, end_station_name, end_station_id, m
ember_casual
dbl  (5): start_lat, start_lng, end_lat, end_lng, trip_duration
dttm (2): started_at, ended_at

ℹ Use `spec()` to retrieve the full column specification for this data.
ℹ Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

The goal is to find how members and casual users differ in behavior.

#Lost bikes vs not lost bikes

First question I have is with lost bikes. Do casuals lose more more bikes than members?

Hide

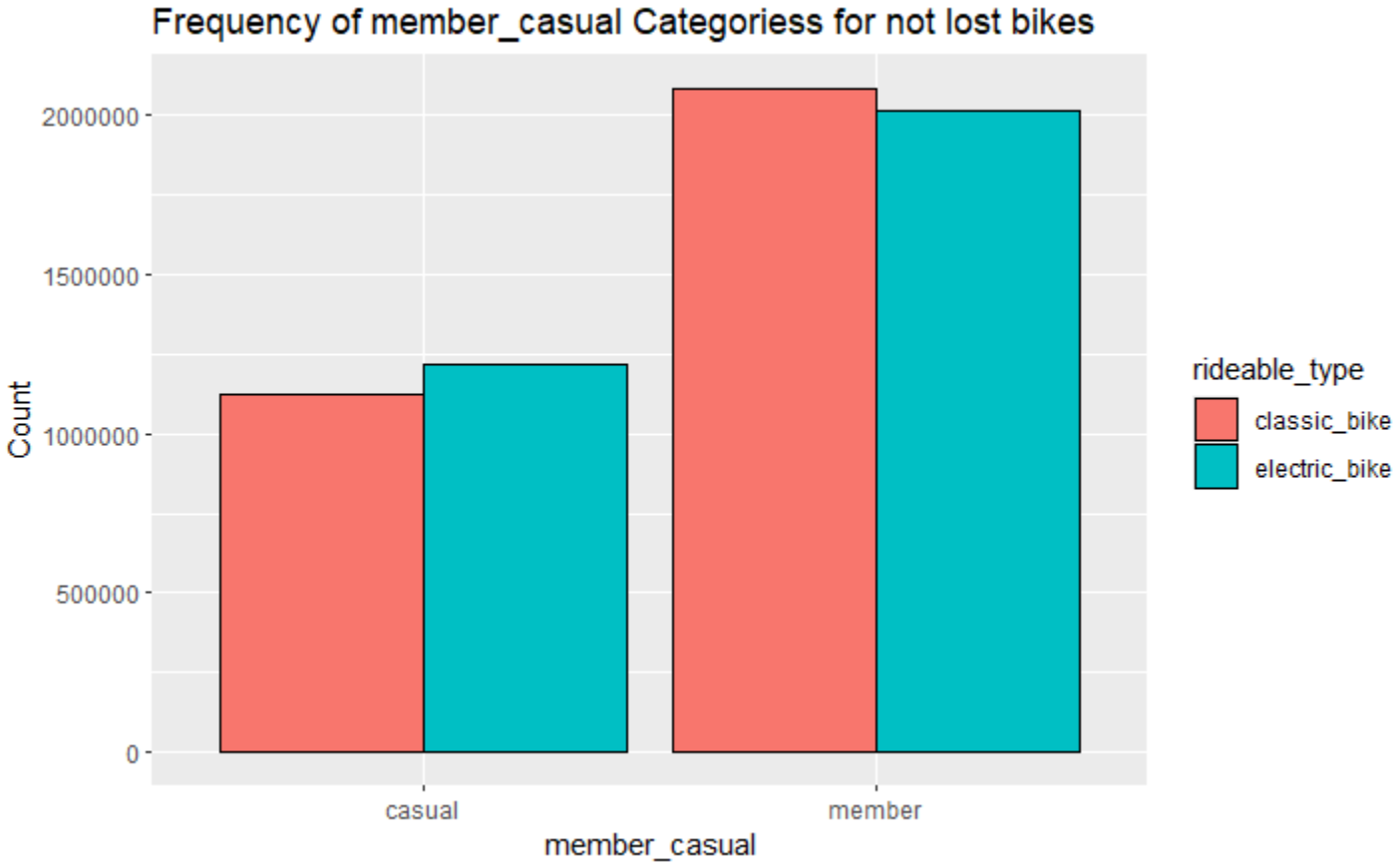
```
trips_df_no_lost <- trips_df  %>%
  select(everything()) %>%
  filter(end_station_name != "Lost")
```

Hide

```
trips_df_lost <- trips_df  %>%
  select(everything()) %>%
  filter(end_station_name == "Lost")
```

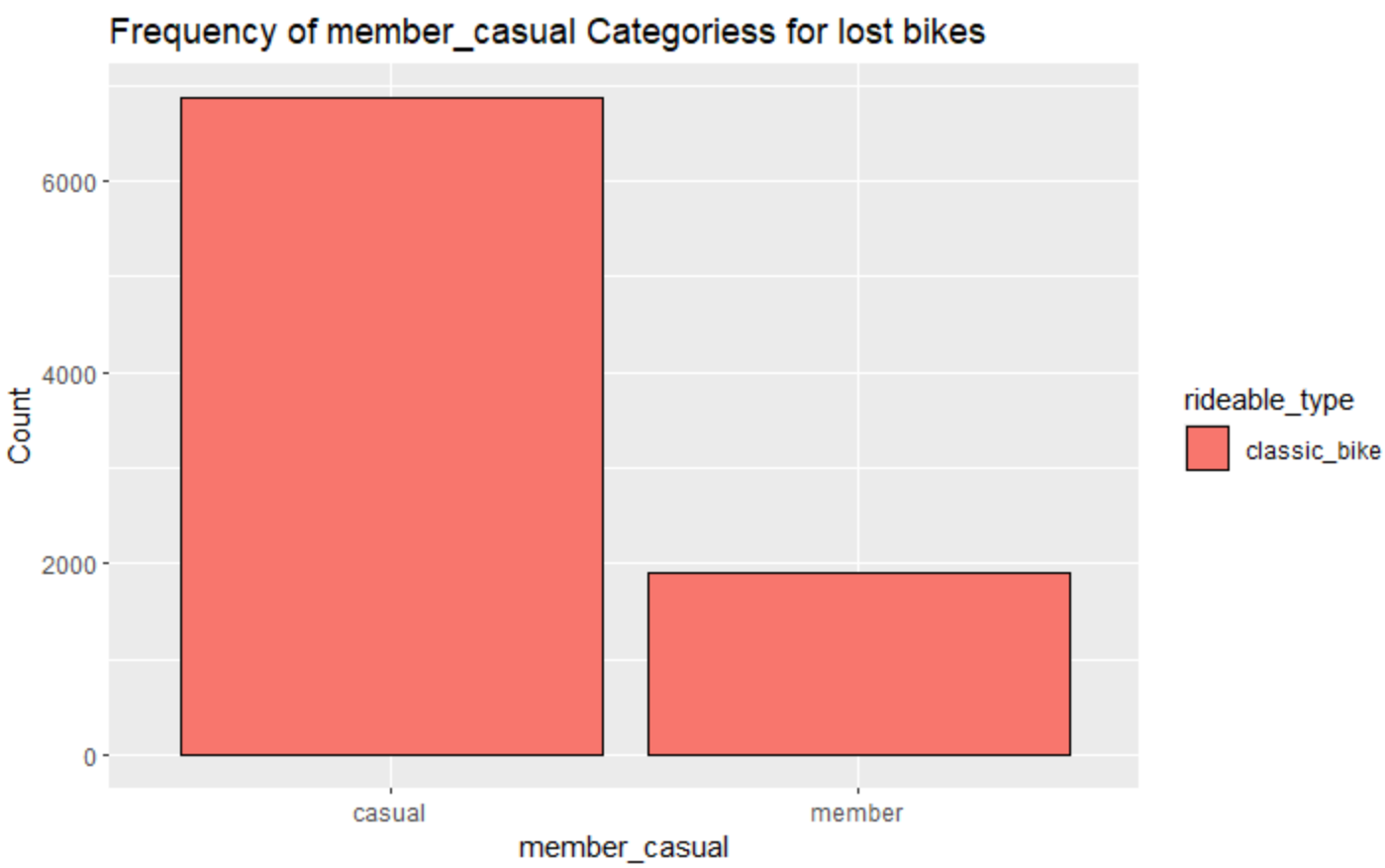
Hide

```
ggplot(trips_df_no_lost, aes(x = member_casual, fill = rideable_type)) +
    geom_bar(color = "black", position = "dodge") +
    labs(title = "Frequency of member_casual Categoriess for not lost bikes", x = "member_casual", y = "Cou
nt")
```



Hide

```
ggplot(trips_df_lost, aes(x = member_casual, fill = rideable_type)) +
    geom_bar(color = "black", position = "dodge") +
    labs(title = "Frequency of member_casual Categoriess for lost bikes", x = "member_casual", y = "Count")
```



Most of the lost bikes are used by casuals despite the casual population being about half of the member population. One thing to note is that for the lost bikes everything is red because only classic bikes can be lost since electric bikes have integrated GPS into them.

As a recommendation A good focus for future analysis is finding why casual users lose more bikes than members and use that for promoting the membership. Or find ways to avoid casual users from losing the bikes,

#Analysing trip durations.

Hide

```
trips_df_no_lost %>%
  select(member_casual, trip_duration) %>%  # Select only relevant columns
  group_by(member_casual) %>%               # Group by member vs casual
  summarise(
    min_trip_duration = min(trip_duration, na.rm = TRUE),
    first_quartile = quantile(trip_duration, 0.25, na.rm = TRUE),
    median_trip_duration = median(trip_duration, na.rm = TRUE),
    mean_trip_duration = mean(trip_duration, na.rm = TRUE),
    third_quartile = quantile(trip_duration, 0.75, na.rm = TRUE),
    max_trip_duration = max(trip_duration, na.rm = TRUE)
  )
```

member_casual <chr>	min_trip_duration <dbl>	first_quartile <dbl>	median_trip_duration <dbl>	mean_trip_duration <dbl>
casual	0.0017333349	6.816667	12.233333	21.21378
member	0.0006499966	5.111904	8.766667	12.36406

2 rows | 1-5 of 7 columns

Hide

```
NA
```

There are still some big outliers even removing trips that lasted 24 hours or longer. So I will use IQR to remove the outliers.

Hide

```
Q1 <- quantile(trips_df_no_lost$trip_duration, 0.25)
Q3 <- quantile(trips_df_no_lost$trip_duration, 0.75)
IQR <- Q3 - Q1

# Define outliers as values outside 1.5 * IQR from Q1 and Q3
lower_bound <- Q1 - 1.5 * IQR
upper_bound <- Q3 + 1.5 * IQR

# Filter outliers
trips_no_outliers <- trips_df_no_lost %>% filter(trip_duration > lower_bound & trip_duration < upper_bound)
```

Hide

```
trips_no_outliers %>%
  select(member_casual, trip_duration) %>%  # Select only relevant columns
  group_by(member_casual) %>%               # Group by member vs casual
  summarise(
    min_trip_duration = min(trip_duration, na.rm = TRUE),
    first_quartile = quantile(trip_duration, 0.25, na.rm = TRUE),
    median_trip_duration = median(trip_duration, na.rm = TRUE),
    mean_trip_duration = mean(trip_duration, na.rm = TRUE),
    third_quartile = quantile(trip_duration, 0.75, na.rm = TRUE),
    max_trip_duration = max(trip_duration, na.rm = TRUE)
  )
```

member_casual <chr>	min_trip_duration <dbl>	first_quartile <dbl>	median_trip_duration <dbl>	mean_trip_duration <dbl>
casual	0.0017333349	6.183333	10.53333	12.42604
member	0.0006499966	4.983333	8.45000	10.37876

2 rows | 1-5 of 7 columns

Hide

```
NA
```

After removing ouliers all trips are bellow 35 minutes with some of them still being shorter than 1 minute. I don't have minimum length for trips so I will work with this limit.

Hide

```
trips_no_outliers %>%
  count()
```

n <int>

n
<int>
5969914

1 row

Hide

```
trips_df %>%
  count()
```

n
<int>
6451628

1 row

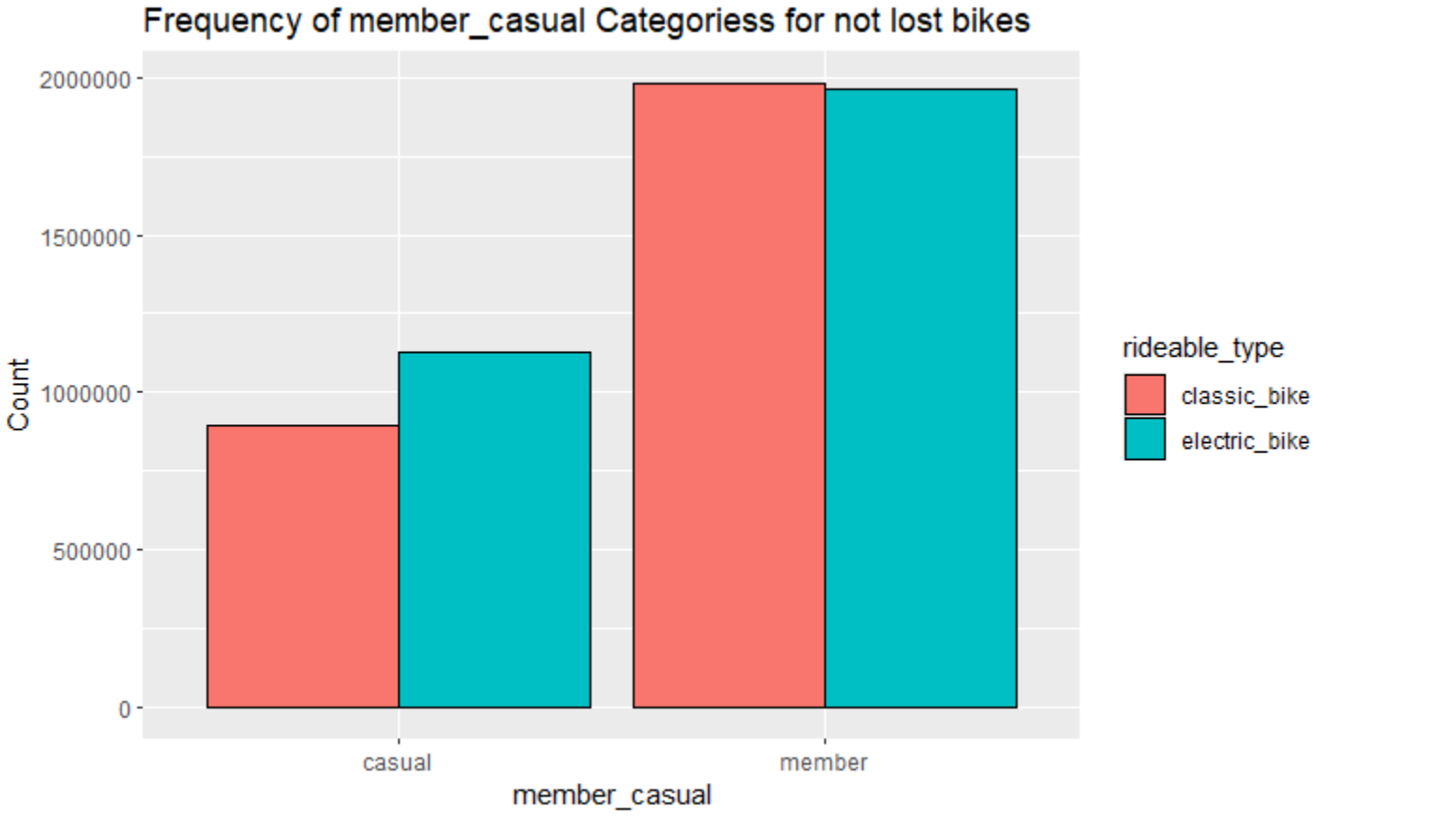
Hide

```
NA
```

481,714 rows were removed .

Hide

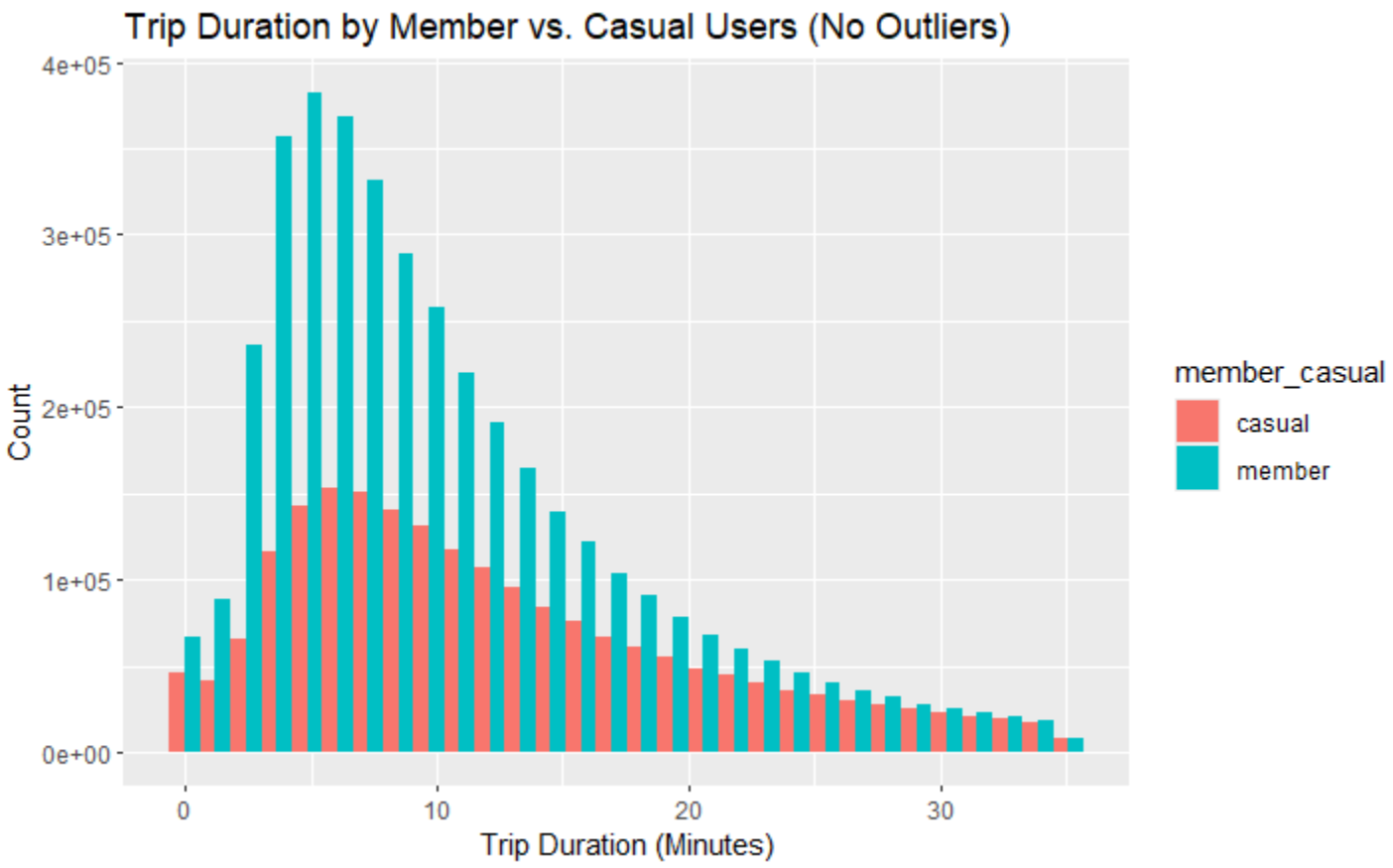
```
ggplot(trips_no_outliers, aes(x = member_casual, fill = rideable_type)) +
    geom_bar(color = "black", position = "dodge") +
    labs(title = "Frequency of member_casual Categoriess for not lost bikes", x = "member_casual", y = "Cou
nt")
```



Hide

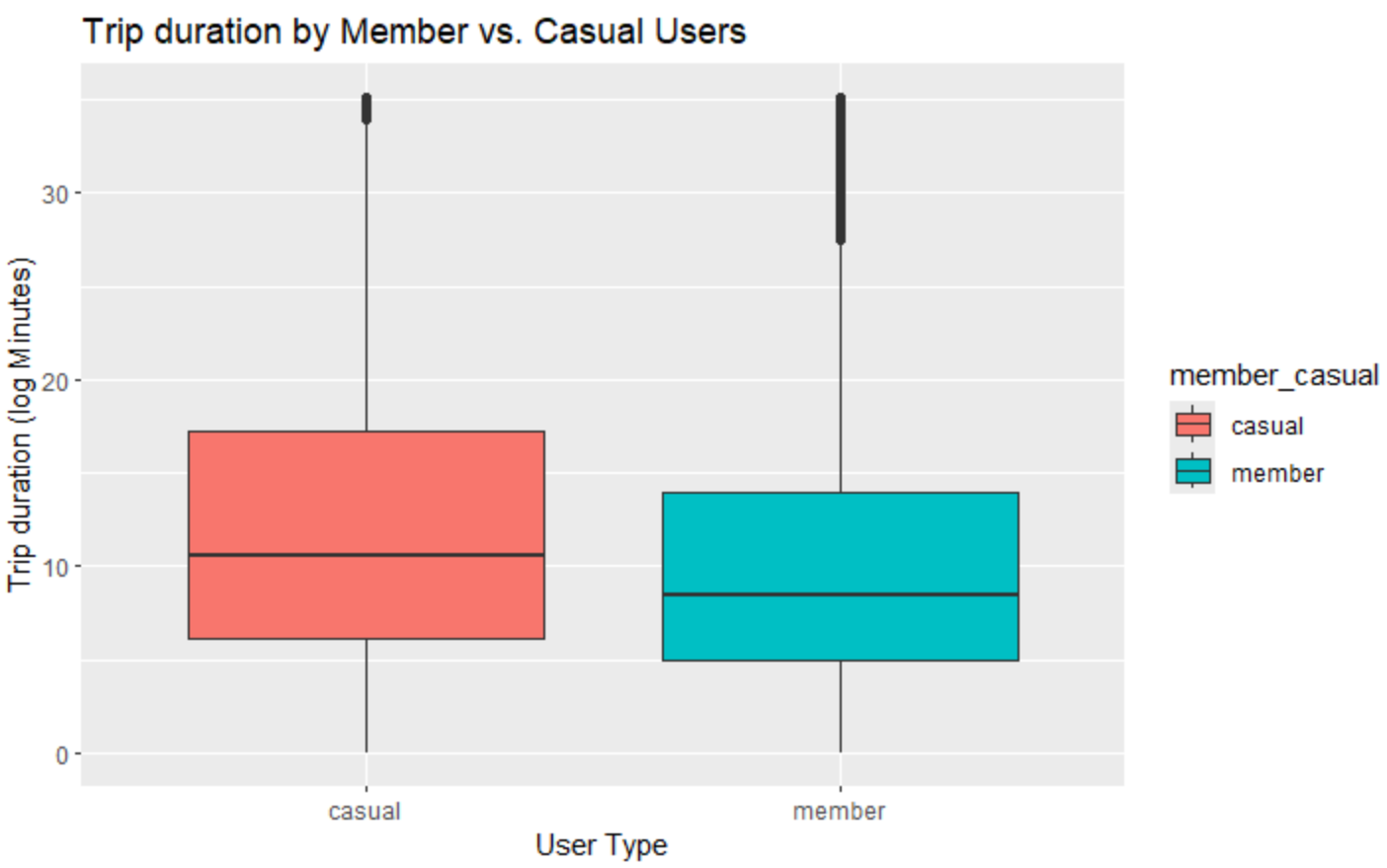
```
ggplot(trips_no_outliers, aes(x = trip_duration, fill = member_casual)) +
  geom_histogram(position = "dodge") +
  labs(title = "Trip Duration by Member vs. Casual Users (No Outliers)", x = "Trip Duration (Minutes)", y = "Count")
```

```
`stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



Hide

```
ggplot(trips_no_outliers, aes(x = member_casual, y = trip_duration, fill = member_casual)) +
  geom_boxplot() +
  labs(title = "Trip duration by Member vs. Casual Users", y = "Trip duration (log Minutes)", x = "User Type")
```


Trip duration by Member vs. Casual Users
Trip duration (log Minutes)
30
20
10
0
casual
member
User Type
member_casual
casual
member

Hide

```
start_station_medians <- trips_no_outliers %>%
  group_by(start_station_name) %>%
  summarise(
    start_median_lat = median(start_lat, na.rm = TRUE),
    start_median_lng = median(start_lng, na.rm = TRUE)
  )

end_station_medians <- trips_no_outliers %>%
  group_by(end_station_name) %>%
  summarise(
    end_median_lat = median(end_lat, na.rm = TRUE),
    end_median_lng = median(end_lng, na.rm = TRUE)
  )

# Classify stations into areas based on their median latitude and longitude
start_station_area <- start_station_medians %>%
  mutate(
    start_area = case_when(
      start_median_lat >= 41.90 & start_median_lng > -87.65 ~ "Northeast",    # North and East side (North
Chicago)
      start_median_lat >= 41.90 & start_median_lng <= -87.65 ~ "Northwest",   # North and West side
      start_median_lat < 41.90 & start_median_lat >= 41.83 & start_median_lng > -87.65 ~ "Centraleast",  #
Central area, East side
      start_median_lat < 41.90 & start_median_lat >= 41.83 & start_median_lng <= -87.65 ~ "Centralwest", #
Central area, West side
      start_median_lat < 41.83 & start_median_lng > -87.65 ~ "Southeast",     # South and East side
      start_median_lat < 41.83 & start_median_lng <= -87.65 ~ "Southwest"     # South and West side
    )
  )

end_station_area <- end_station_medians %>%
  mutate(
    end_area = case_when(
      end_median_lat >= 41.90 & end_median_lng > -87.65 ~ "Northeast",    # North and East side (North Chic
ago)
      end_median_lat >= 41.90 & end_median_lng <= -87.65 ~ "Northwest",   # North and West side
      end_median_lat < 41.90 & end_median_lat >= 41.83 & end_median_lng > -87.65 ~ "Centraleast",  # Centra
l area, East side
      end_median_lat < 41.90 & end_median_lat >= 41.83 & end_median_lng <= -87.65 ~ "Centralwest", # Centra
l area, West side
      end_median_lat < 41.83 & end_median_lng > -87.65 ~ "Southeast",     # South and East side
      end_median_lat < 41.83 & end_median_lng <= -87.65 ~ "Southwest"     # South and West side
    )
  )

# Merge the area classification back into the main dataset
trips_no_outliers <- trips_no_outliers %>%
  left_join(start_station_area, by = "start_station_name") %>%
  left_join(end_station_area, by = "end_station_name")
```

Hide

```
start_area_percentages <- trips_no_outliers %>%
  group_by(start_area, member_casual) %>%
  summarise(count = n()) %>%
  mutate(percentage = count / sum(count) * 100) %>%
  ungroup()
```

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

Hide

```
end_area_percentages <- trips_no_outliers %>%
  group_by(end_area, member_casual) %>%
  summarise(count = n()) %>%
  mutate(percentage = count / sum(count) * 100) %>%
  ungroup()
```

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

Hide

```
start_area_percentages %>%
  select()
```

12 rows

Hide

```
end_area_percentages %>%
  select()
```

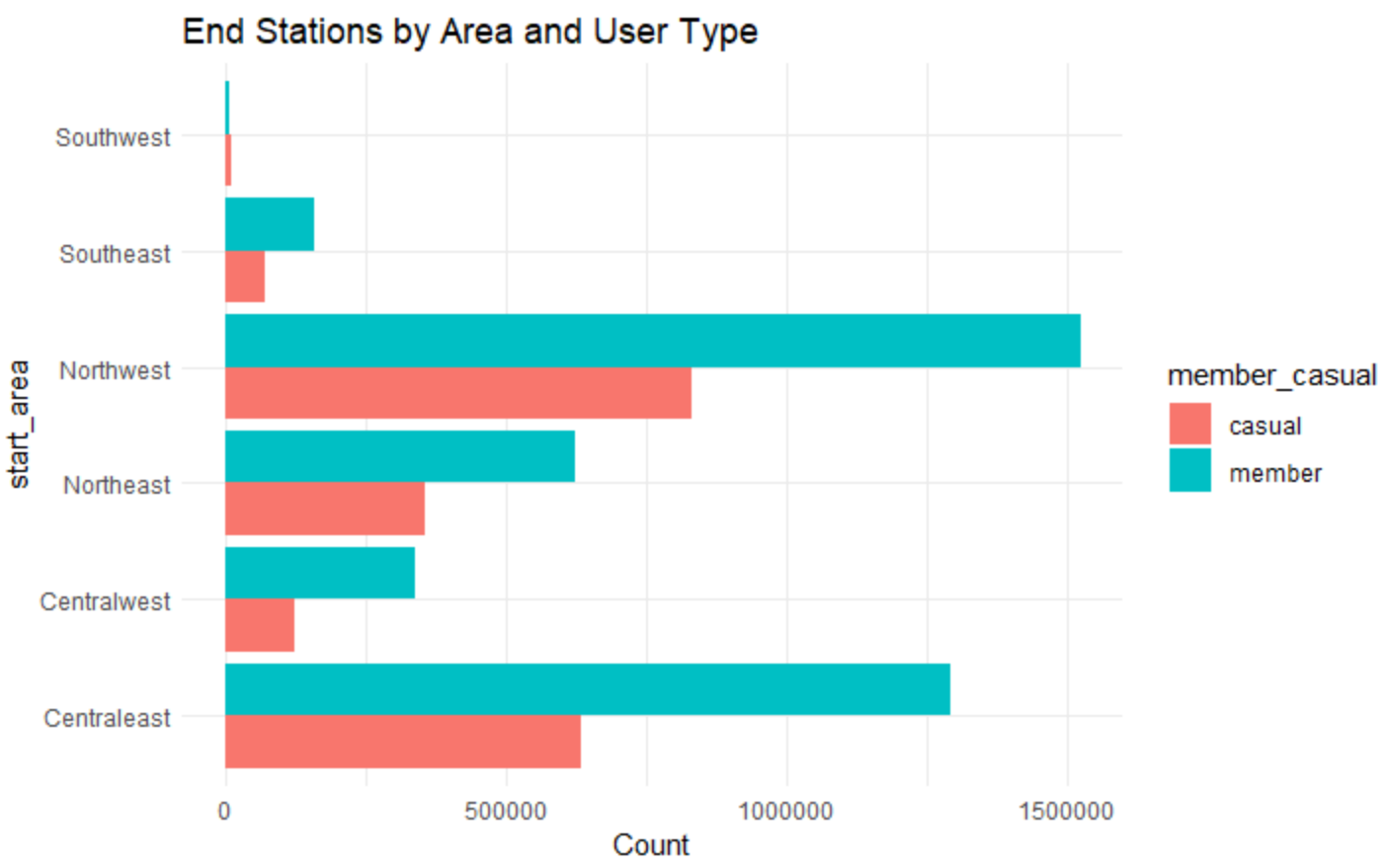
12 rows

Hide

```
NA
```

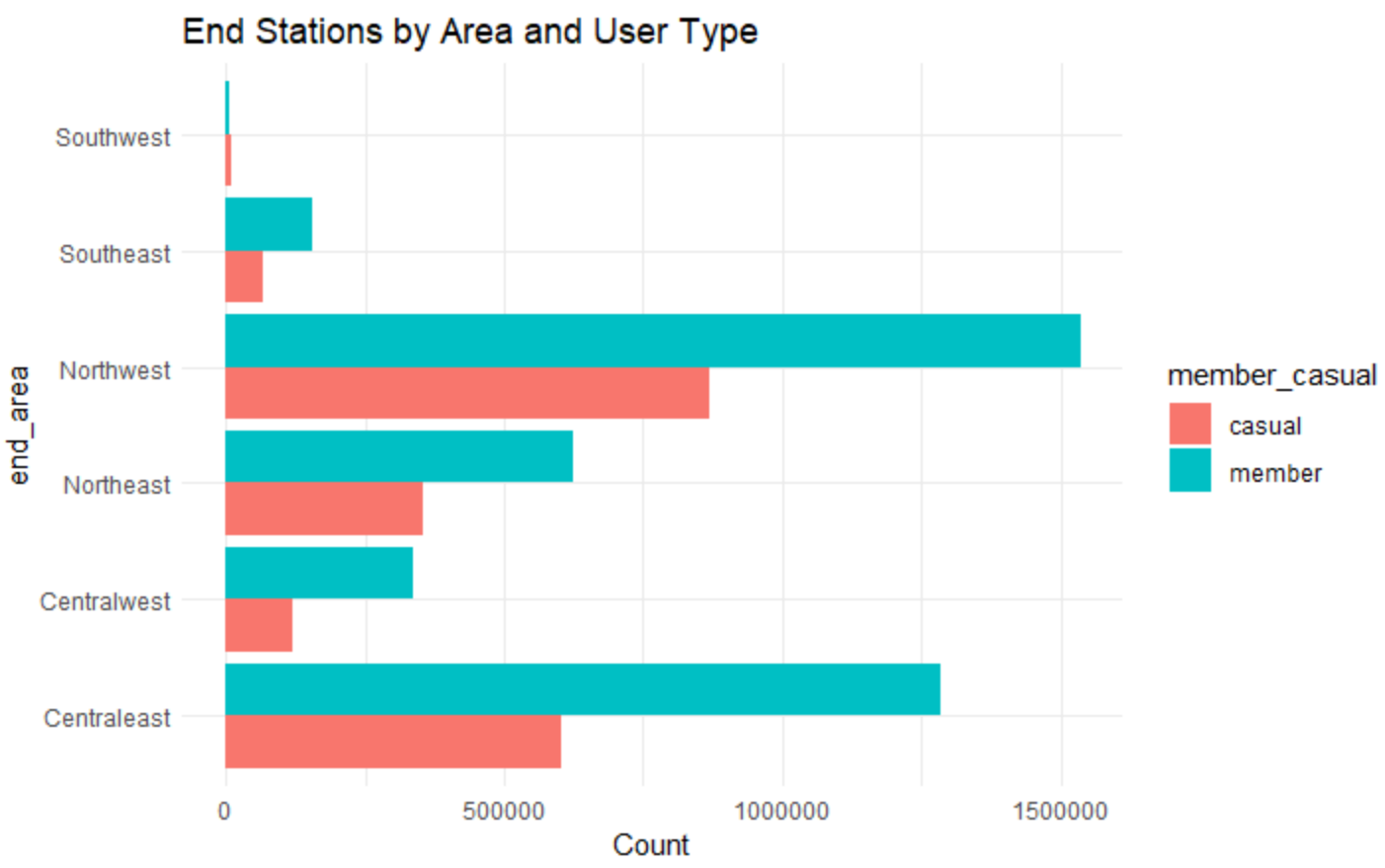
Hide

```
ggplot(start_area_percentages, aes(x = start_area, y = count, fill = member_casual)) +
  geom_bar(stat = "identity", position = "dodge") +
  ggtitle("End Stations by Area and User Type") +
  xlab("start_area") +
  ylab("Count") +
  coord_flip() +
  theme_minimal()
```



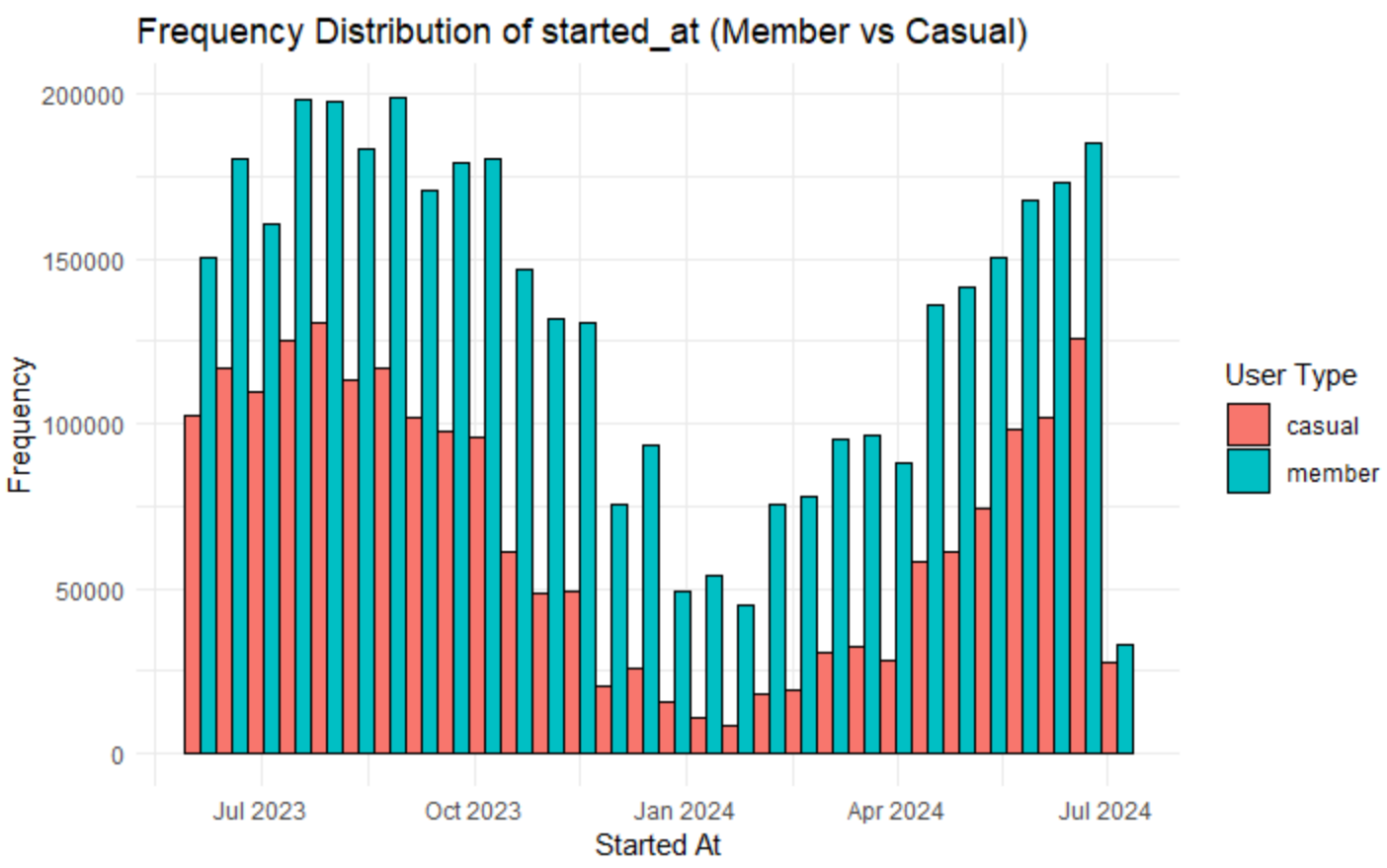
Hide

```
ggplot(end_area_percentages, aes(x = end_area, y = count, fill = member_casual)) +
  geom_bar(stat = "identity", position = "dodge") +
  ggtitle("End Stations by Area and User Type") +
  xlab("end_area") +
  ylab("Count") +
  coord_flip() +
  theme_minimal()
```



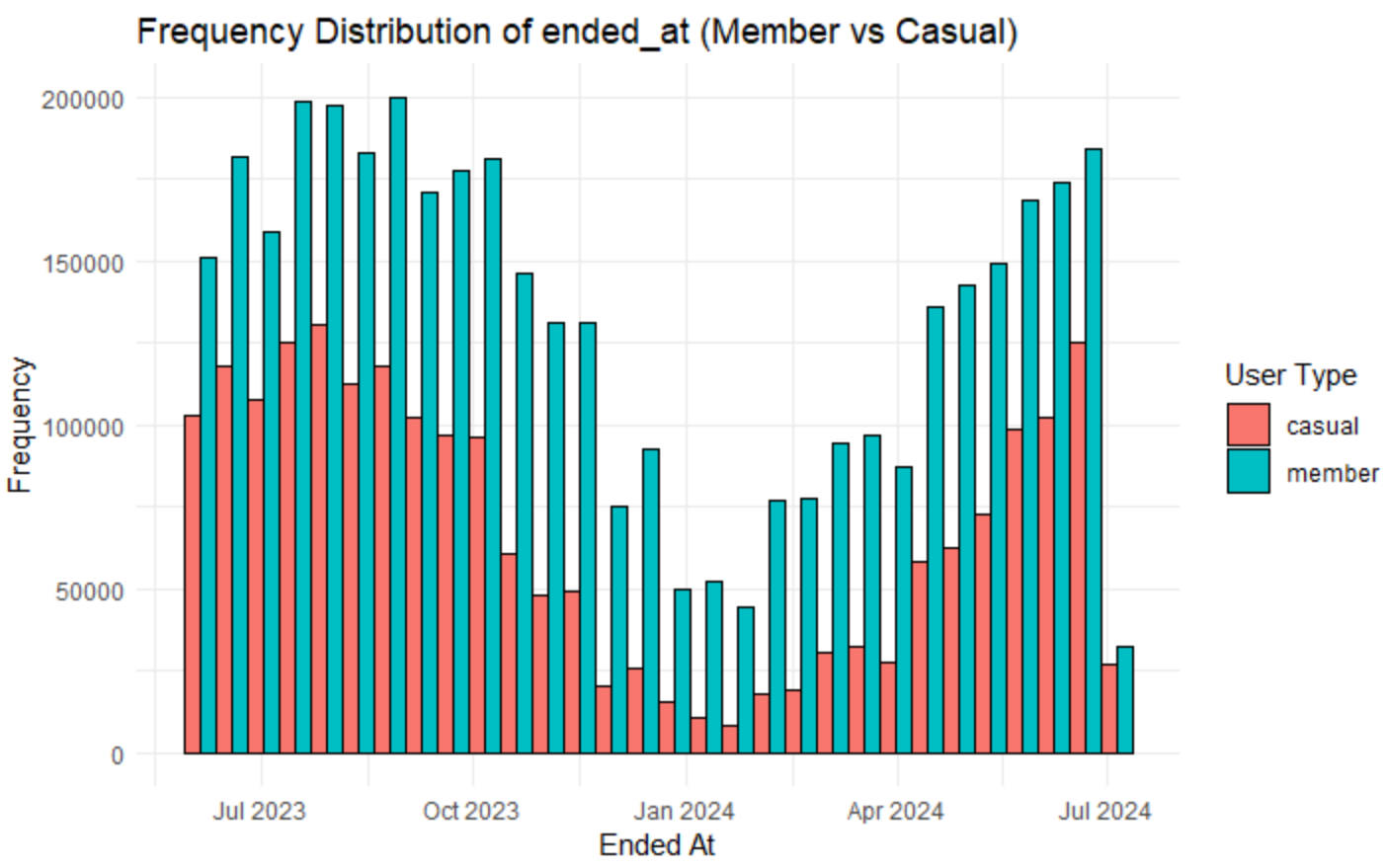
Hide

```
ggplot(trips_no_outliers, aes(x = started_at, fill = member_casual)) +
    geom_histogram(position = "dodge", color = "black", bins = 30) +
    labs(title = "Frequency Distribution of started_at (Member vs Casual)",
         x = "Started At",
         y = "Frequency",
         fill = "User Type") +
    theme_minimal()
```



Hide

```
ggplot(trips_no_outliers, aes(x = ended_at, fill = member_casual)) +
    geom_histogram(position = "dodge", color = "black", bins = 30) +
    labs(title = "Frequency Distribution of ended_at (Member vs Casual)",
         x = "Ended At",
         y = "Frequency",
         fill = "User Type") +
    theme_minimal()
```



Hide

```
# Create columns for the day of the week
trips_no_outliers <- trips_no_outliers %>%
  mutate(started_day = wday(started_at, label = TRUE),
         ended_day = wday(ended_at, label = TRUE))

trips_df_no_lost
```

ride_id <chr>	rideable_type <chr>	started_at <S3: POSIXct>	ended_at <S3: POSIXct>	start_station_name <chr>
6F1682AC40EB6F71	electric_bike	2023-06-05 13:34:12	2023-06-05 14:31:56	Out of station
622A1686D64948EB	electric_bike	2023-06-05 01:30:22	2023-06-05 01:33:06	Out of station
3C88859D926253B4	electric_bike	2023-06-20 18:15:49	2023-06-20 18:32:05	Out of station
EAD8A5E0259DEC88	electric_bike	2023-06-19 14:56:00	2023-06-19 15:00:35	Out of station
5A36F21930D6A55C	electric_bike	2023-06-19 15:03:34	2023-06-19 15:07:16	Out of station
CF682EA7D0F961DB	electric_bike	2023-06-09 21:30:25	2023-06-09 21:49:52	Out of station
4910FBB710157754	electric_bike	2023-06-03 13:34:09	2023-06-03 13:34:28	Out of station
EA19D850A42F56D8	electric_bike	2023-06-03 13:34:46	2023-06-03 13:35:00	Out of station
E68F43784662A2D0	electric_bike	2023-06-02 22:27:35	2023-06-02 22:35:26	Out of station
5A013E29CC001611	electric_bike	2023-06-02 21:18:31	2023-06-03 01:27:19	Out of station

1-10 of 6,442,856 rows | 1-5 of 20 columns

Previous 1 2 3 4 5 6 … 100 Next

the behaviors in locations and times are similar accounting for the fact that casuals are half of members.

Hide

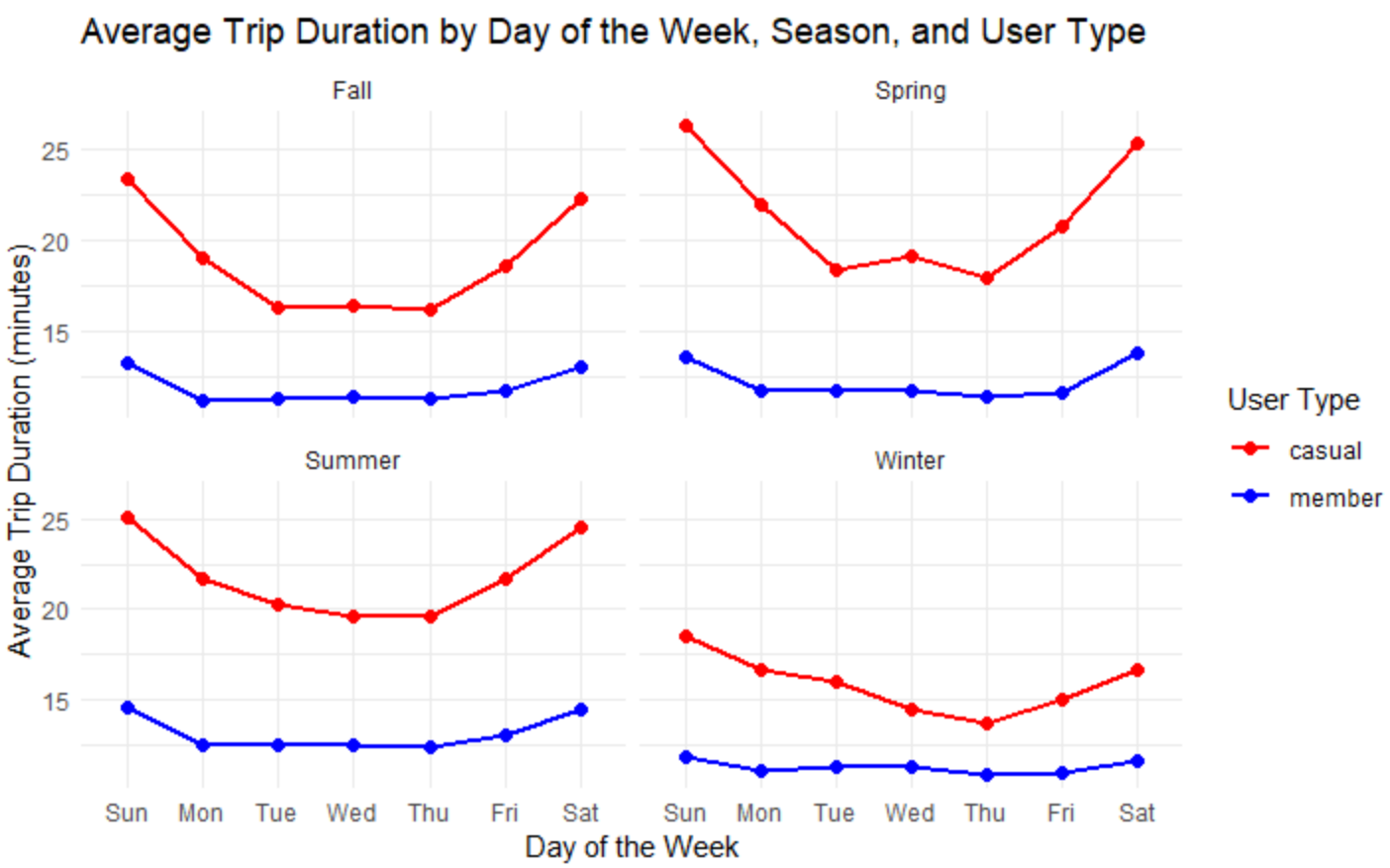
```
# Create a season column based on the date
trips_no_outliers <- trips_df_no_lost %>%
  mutate(started_day = wday(started_at, label = TRUE),
         season = case_when(
           month(started_at) %in% c(12, 1, 2) ~ "Winter",
           month(started_at) %in% c(3, 4, 5) ~ "Spring",
           month(started_at) %in% c(6, 7, 8) ~ "Summer",
           month(started_at) %in% c(9, 10, 11) ~ "Fall"
         ))

# Calculate average trip duration for each day, season, and user type
average_trip_duration <- trips_no_outliers %>%
  group_by(started_day, season, member_casual) %>%
  summarise(avg_duration = mean(trip_duration, na.rm = TRUE)) %>%
  ungroup()
```

```
`summarise()` has grouped output by 'started_day', 'season'. You can override using the `.groups` argument.
```

Hide

```
# Plot average trip duration by day of the week, season, and user type as a line graph
ggplot(average_trip_duration, aes(x = started_day, y = avg_duration, color = member_casual, group = member_
casual)) +
  geom_line(size = 1) +
  geom_point(size = 2) +
  facet_wrap(~ season) +
  labs(title = "Average Trip Duration by Day of the Week, Season, and User Type",
       x = "Day of the Week",
       y = "Average Trip Duration (minutes)",
       color = "User Type") +
  theme_minimal() +
  scale_color_manual(values = c("member" = "blue", "casual" = "red"))
```

```
Warning: Using `size` aesthetic for lines was deprecated in ggplot2 3.4.0.
ℹ Please use `linewidth` instead.
This warning is displayed once every 8 hours.
Call `lifecycle::last_lifecycle_warnings()` to see where this warning was generated.
```



Causal drivers have longer trips they also have an increase in times on weekends. Although members also have increase in weekends is not as significant. Another thing to notice is that on the summer there is an over all growth for everything with a big increase on the weekend numbers.

1. Focus on the casual drives on the week days. Even though not as significant as the weekends was there are a number of them and they do have longer trips than their member counterparts. We can advertise the low fee per minute that the membership offers. as well as the free unlocks
2. Focus on the weekend and summer riders and offer a service that could substitute the yearly subscription
3. Lastly we can focus on the weekend drivers. Even though they do not use it as often if they use it every week they could still benefit from the subscription. We can advertise from that angle.