Case_Study

Mudassar Afzal

2023-12-12

Introduction

In this project, a group of analysts is studying data for a bike-sharing company in Chicago. The head of marketing thinks that the company's success in the future depends on getting more people to sign up for yearly memberships. So, the team is trying to figure out how people who use the bikes occasionally and those with yearly memberships use the bikes in different ways. With this information, the team plans to create a new way to encourage occasional riders to become yearly members. However, before the company leaders agree to these ideas, the team needs to show them strong evidence and professional visual aids based on the data.

We will follow 6 steps to accomplesh our task
- Ask
- Prepare
- Process
- Analyze
- Share
- Act

Ask

Business Task

We're figuring out how people who just use the bikes sometimes and those who have subscriptions use Cyclist bikes in different ways. The goal is to get more customers and turn the ones who use the bikes casually into subscription members. This way, we can make more money and offer better services.

Prepare

- Determine the credibility
  This dataset containing information about cyclists in Chicago for the year 2023. The dataset used in this case study is provided by Motivate International Inc. under the Lyft Bikes and Scooters, LLC license and is owned by the City of Chicago. Because of privacy concerns, information that could identify riders personally is not included. This limitation affects the analysis because we can't find out whether casual riders reside in the Cyclistic service area in Chicago or if they have bought several single passes.

```r
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   1.5.0
## v ggplot2   3.4.2     v tibble    3.2.1
## v lubridate 1.9.2     v tidyr     1.3.0
## v purrr     1.0.1
## -- 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 errors
```

```r
sep_2023 <- read_csv("202309-divvy-tripdata.csv")
```

```
## Rows: 666371 Columns: 13
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (7): ride_id, rideable_type, start_station_name, start_station_id, end_...
## dbl  (4): start_lat, start_lng, end_lat, end_lng
## 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.
```

```r
oct_2023 <- read_csv("202310-divvy-tripdata.csv")
```

```
## Rows: 537113 Columns: 13
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (7): ride_id, rideable_type, start_station_name, start_station_id, end_...
## dbl  (4): start_lat, start_lng, end_lat, end_lng
## 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.
```

```r
nov_2023 <- read_csv("202311-divvy-tripdata.csv")
```

```
## Rows: 362518 Columns: 13
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (7): ride_id, rideable_type, start_station_name, start_station_id, end_...
## dbl  (4): start_lat, start_lng, end_lat, end_lng
## 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.
```

- Data's integrity
  Name of the columns must match so that we can combine the data

```r
colnames(sep_2023)
```

```
##  [1] "ride_id"            "rideable_type"      "started_at"
##  [4] "ended_at"           "start_station_name" "start_station_id"
##  [7] "end_station_name"   "end_station_id"     "start_lat"
## [10] "start_lng"          "end_lat"            "end_lng"
## [13] "member_casual"
```

```r
colnames(oct_2023)
```

```
##  [1] "ride_id"            "rideable_type"      "started_at"
##  [4] "ended_at"           "start_station_name" "start_station_id"
##  [7] "end_station_name"   "end_station_id"     "start_lat"
## [10] "start_lng"          "end_lat"            "end_lng"
## [13] "member_casual"
```

```r
colnames(nov_2023)
```

```
##  [1] "ride_id"            "rideable_type"      "started_at"
##  [4] "ended_at"           "start_station_name" "start_station_id"
##  [7] "end_station_name"   "end_station_id"     "start_lat"
## [10] "start_lng"          "end_lat"            "end_lng"
## [13] "member_casual"
```

We can combine the data

```r
data <- bind_rows(sep_2023,oct_2023,nov_2023)
```

Process

```r
head(data)
```

```
## # A tibble: 6 x 13
##   ride_id          rideable_type started_at          ended_at
##   <chr>            <chr>         <dttm>              <dttm>
## 1 011C1903BF4E2E28 classic_bike  2023-09-23 00:27:50 2023-09-23 00:33:27
## 2 87DB80E048A1BF9F classic_bike  2023-09-02 09:26:43 2023-09-02 09:38:19
## 3 7C2EB7AF669066E3 electric_bike 2023-09-25 18:30:11 2023-09-25 18:41:39
## 4 57D197B010269CE3 classic_bike  2023-09-13 15:30:49 2023-09-13 15:39:18
## 5 8A2CEA7C8C8074D8 classic_bike  2023-09-18 15:58:58 2023-09-18 16:05:04
## 6 03F7044D1304CD58 electric_bike 2023-09-15 20:19:25 2023-09-15 20:30:27
## # i 9 more variables: start_station_name <chr>, start_station_id <chr>,
## #   end_station_name <chr>, end_station_id <chr>, start_lat <dbl>,
## #   start_lng <dbl>, end_lat <dbl>, end_lng <dbl>, member_casual <chr>
```

Drop the irrelevant columns and na values from the data set

```
data<-na.omit(data)
data <- subset(data, select = c('rideable_type',
                                'started_at',
                                'ended_at',
                                'start_station_id',
                                'start_station_name',
                                'end_station_id',
                                'end_station_name',
                                'member_casual') )
```

Renaming the columns for better understanding

```
data <- data %>%
  rename(ride_type=rideable_type,
         start_time=started_at,
         end_time=ended_at,
         usertype=member_casual)
```

```
dplyr::glimpse(data)
```

```
## Rows: 1,185,214
## Columns: 8
## $ ride_type          <chr> "classic_bike", "classic_bike", "electric_bike", "c~
## $ start_time         <dttm> 2023-09-23 00:27:50, 2023-09-02 09:26:43, 2023-09-~
## $ end_time           <dttm> 2023-09-23 00:33:27, 2023-09-02 09:38:19, 2023-09-~
## $ start_station_id   <chr> "TA1309000061", "TA1307000142", "SL-010", "TA130700~
## $ start_station_name <chr> "Halsted St & Wrightwood Ave", "Clark St & Drummond~
## $ end_station_id     <chr> "TA1307000052", "TA1306000026", "13304", "TA1308000~
## $ end_station_name   <chr> "Sheffield Ave & Wellington Ave", "Racine Ave & Ful~
## $ usertype           <chr> "member", "member", "member", "member", "member", "~
```

Create new column 'date' from 'start time' and then split into month, day and day of week

```
data$date<-as.Date(data$start_time)

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

Creating new column name 'ride_length'

```
data$ride_length <- difftime(data$end_time,data$start_time)
```

Convert ride_length from seconds to minutes.

```
data <- data %>%
  mutate(ride_length = minute(seconds_to_period(ride_length)))
```

```
summary(data$ride_length)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  -54.00    5.00    9.00   12.09   16.00   59.00
```

We need to remove rows with negative ride length

```
new_data <- data[!(data$end_station_id == "Hubbard Bike-checking (LBS-WH-TEST)" &
                         !is.na(data$start_station_id) &
                         !is.na(data$start_station_name) &
                         !is.na(data$end_station_name) &
                         !is.na(data$end_station_id) | data$ride_length < 0),]
```

Analysis

```
summary(new_data)
```

```
##   ride_type           start_time
##  Length:1185165     Min.   :2023-09-01 00:00:44.00
##  Class :character   1st Qu.:2023-09-18 14:28:01.00
##  Mode  :character   Median :2023-10-05 18:40:47.00
##                     Mean   :2023-10-09 14:00:37.64
##                     3rd Qu.:2023-10-29 02:11:36.00
##                     Max.   :2023-11-30 23:58:38.00
##     end_time                       start_station_id   start_station_name
##  Min.   :2023-09-01 00:03:06.00   Length:1185165     Length:1185165
##  1st Qu.:2023-09-18 14:46:57.00   Class :character   Class :character
##  Median :2023-10-05 18:54:57.00   Mode  :character   Mode  :character
##  Mean   :2023-10-09 14:15:51.73
##  3rd Qu.:2023-10-29 02:32:38.00
##  Max.   :2023-12-01 09:47:35.00
##  end_station_id     end_station_name     usertype              date
##  Length:1185165     Length:1185165     Length:1185165     Min.   :2023-09-01
##  Class :character   Class :character   Class :character   1st Qu.:2023-09-18
##  Mode  :character   Mode  :character   Mode  :character   Median :2023-10-05
##                                                           Mean   :2023-10-08
##                                                           3rd Qu.:2023-10-29
##                                                           Max.   :2023-11-30
##     month               day            day_of_week         ride_length
##  Length:1185165     Length:1185165     Length:1185165     Min.   : 0.0
##  Class :character   Class :character   Class :character   1st Qu.: 5.0
##  Mode  :character   Mode  :character   Mode  :character   Median : 9.0
##                                                           Mean   :12.1
##                                                           3rd Qu.:16.0
##                                                           Max.   :59.0
```

```
aggregate(new_data$ride_length ~ new_data$usertype, FUN = mean)
```

```
##   new_data$usertype new_data$ride_length
## 1            casual             14.99677
## 2            member             10.62170
```

```r
aggregate(new_data$ride_length ~ new_data$usertype, FUN = median)
```

```
##   new_data$usertype new_data$ride_length
## 1            casual                  11
## 2            member                   8
```

```r
aggregate(new_data$ride_length ~ new_data$usertype, FUN = max)
```

```
##   new_data$usertype new_data$ride_length
## 1            casual                  59
## 2            member                  59
```

```r
aggregate(new_data$ride_length ~ new_data$usertype, FUN = min)
```

```
##   new_data$usertype new_data$ride_length
## 1            casual                   0
## 2            member                   0
```

Now lets see the length of ride on each day of week

```r
aggregate(new_data$ride_length ~ new_data$usertype + new_data$day_of_week,
          FUN = mean)
```

```
##    new_data$usertype new_data$day_of_week new_data$ride_length
## 1             casual               Friday             14.60549
## 2             member               Friday             10.50119
## 3             casual               Monday             14.55144
## 4             member               Monday             10.09954
## 5             casual             Saturday             16.97101
## 6             member             Saturday             11.72184
## 7             casual               Sunday             17.05075
## 8             member               Sunday             11.73521
## 9             casual             Thursday             12.92276
## 10            member             Thursday             10.23588
## 11            casual              Tuesday             13.16082
## 12            member              Tuesday             10.24128
## 13            casual            Wednesday             12.98660
## 14            member            Wednesday             10.27729
```

Sort by days

```r
new_data$day_of_week <- ordered(new_data$day_of_week, levels=c("Sunday", "Monday",
                                                               "Tuesday", "Wednesday", "Thursda

aggregate(new_data$ride_length ~ new_data$usertype + new_data$day_of_week,
          FUN = mean)
```

```
##    new_data$usertype new_data$day_of_week new_data$ride_length
## 1             casual               Sunday             17.05075
## 2             member               Sunday             11.73521
```

```
## 3             casual               Monday             14.55144
## 4             member               Monday             10.09954
## 5             casual              Tuesday             13.16082
## 6             member              Tuesday             10.24128
## 7             casual            Wednesday             12.98660
## 8             member            Wednesday             10.27729
## 9             casual             Thursday             12.92276
## 10            member             Thursday             10.23588
## 11            casual               Friday             14.60549
## 12            member               Friday             10.50119
## 13            casual             Saturday             16.97101
## 14            member             Saturday             11.72184
```

Now we'll see what type of rider rides how much on each day.

```
new_data %>%
  mutate(weekday = wday(start_time, label = TRUE)) %>%
  group_by(usertype, day_of_week) %>%
  summarise(number_of_rides = n()
            ,average_duration = mean(ride_length)) %>%
  arrange(usertype, day_of_week)
```

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

```
## # A tibble: 14 x 4
## # Groups:   usertype [2]
##    usertype day_of_week number_of_rides average_duration
##    <chr>    <ord>                <int>            <dbl>
##  1 casual   Sunday               72116             17.1
##  2 casual   Monday               45682             14.6
##  3 casual   Tuesday              44366             13.2
##  4 casual   Wednesday            45817             13.0
##  5 casual   Thursday             48008             12.9
##  6 casual   Friday               58191             14.6
##  7 casual   Saturday             85153             17.0
##  8 member   Sunday               88341             11.7
##  9 member   Monday              109100             10.1
## 10 member   Tuesday             124198             10.2
## 11 member   Wednesday           126566             10.3
## 12 member   Thursday            125090             10.2
## 13 member   Friday              111518             10.5
## 14 member   Saturday            101019             11.7
```

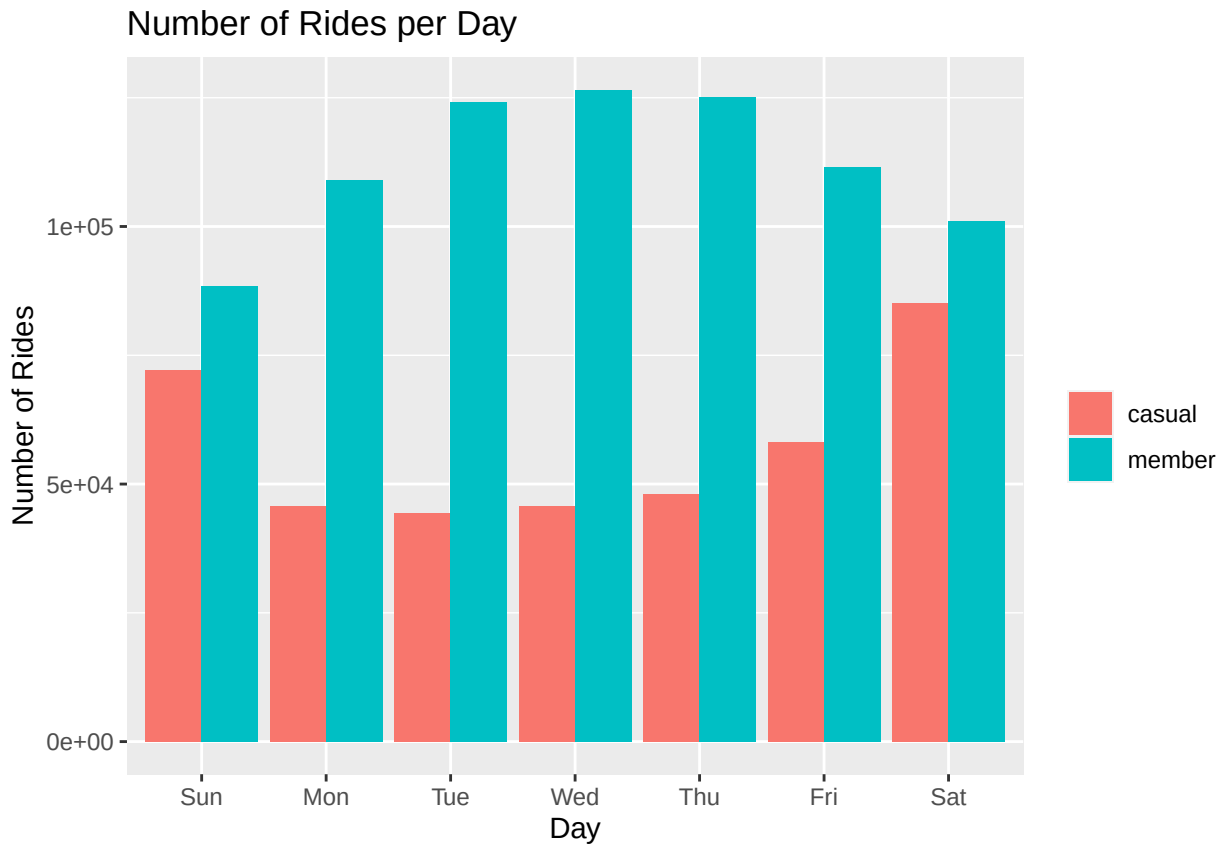
We can see thatnumber of casual rider are more than member but casual rider rides more average length of ride than members and also on Saturday and Sunday both members and casual rider take longer rides

Share

Data Visualization

```
# Visualization for number of rides per day
new_data %>%
  mutate(weekday = wday(start_time, label = TRUE)) %>%
  group_by(usertype, weekday) %>%
  summarise(number_of_rides = n()
            ,average_duration = mean(ride_length)) %>%
  arrange(usertype, weekday)  %>%
  ggplot(aes(x = weekday, y = number_of_rides, fill = usertype)) +
  geom_col(position = "dodge") +
  labs(title = "Number of Rides per Day", x = "Day", y = "Number of Rides", fill = "")
```

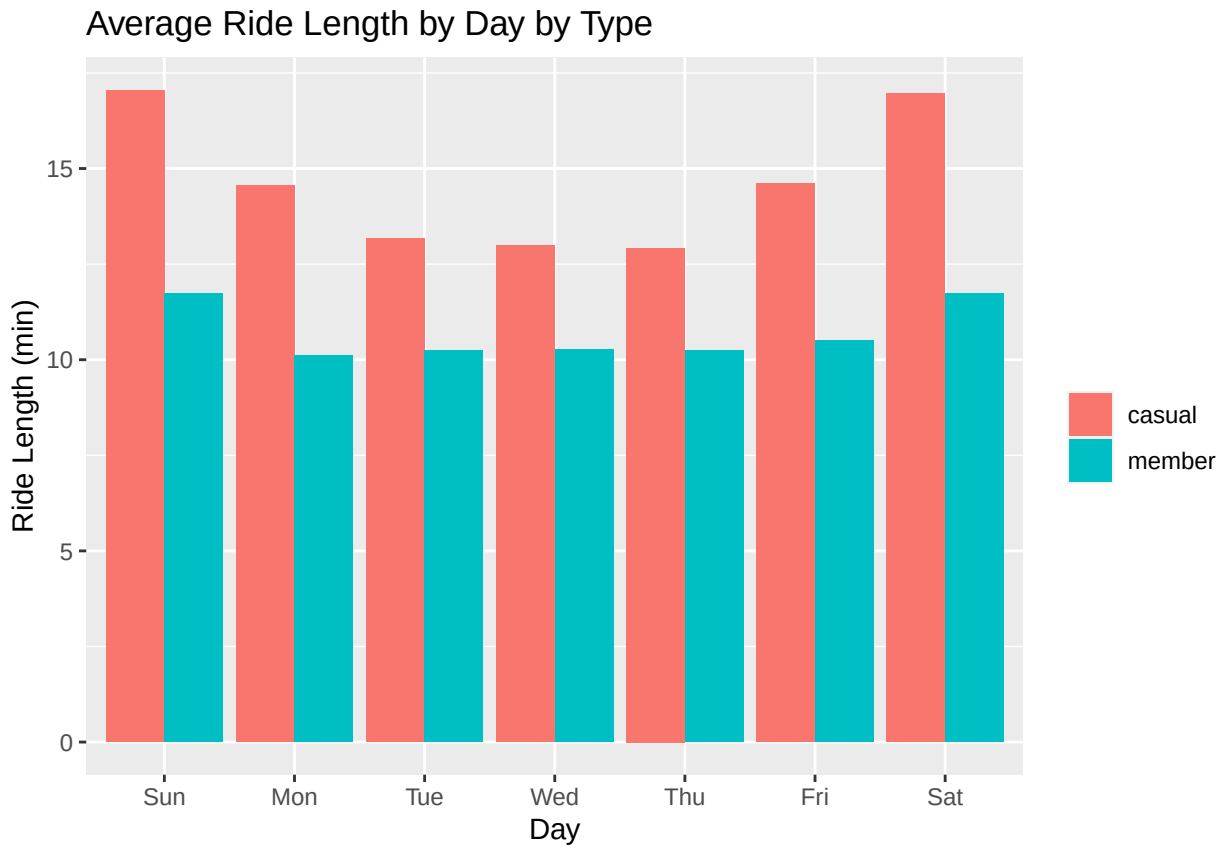
```
## `summarise()` has grouped output by 'usertype'. You can override using the
## `.groups` argument.
```




```
# Visualization for number of rides per day by membership type
new_data %>%
  mutate(weekday = wday(start_time, label = TRUE)) %>%
  group_by(usertype, weekday) %>%
  summarise(number_of_rides = n()
```

```
            ,average_duration = mean(ride_length)) %>%
  arrange(usertype, weekday)  %>%
  ggplot(aes(x = weekday, y = average_duration, fill = usertype)) +
  geom_col(position = "dodge") +
  labs(title = "Average Ride Length by Day by Type", x = "Day", y = "Ride Length (min)", fill = "")
```

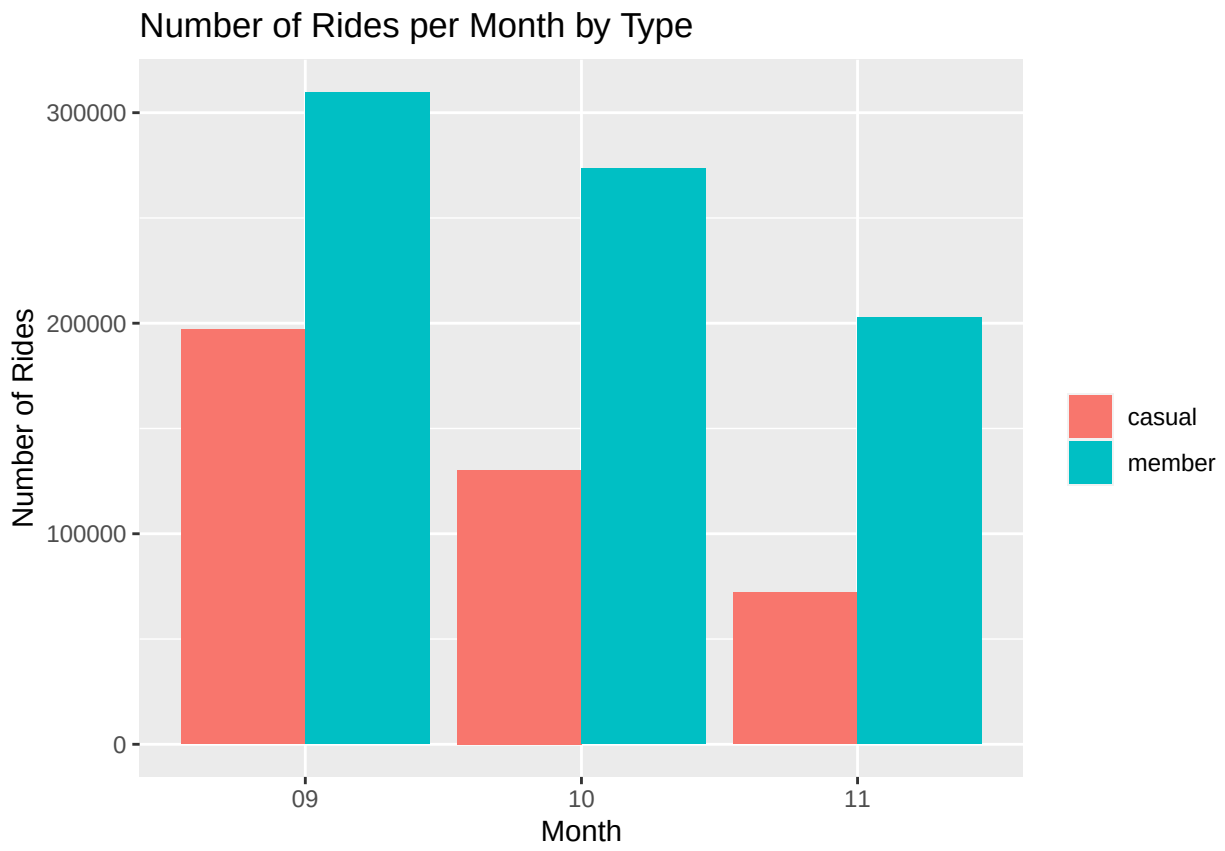
```
## `summarise()` has grouped output by 'usertype'. You can override using the
## `.groups` argument.
```



Casual riders usually go on longer rides compared to members. Both members and casual tend to take longer rides on the weekends than they do on weekdays.

```
options(scipen = 600000)
# Visualization of number of rides per month by membership type
new_data %>%
  group_by(usertype, month) %>%
  summarise(number_of_rides = n()) %>%
  arrange(usertype, month) %>%
    ggplot(aes(x = month, y = number_of_rides, fill = usertype)) +
    geom_col(position = "dodge") +
  labs(title = "Number of Rides per Month by Type", x = "Month", y = "Number of Rides", fill = "")
```

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




```
# Determine top 10 stations trips are STARTED from
start_station <- new_data %>%
  select(start_station_name) %>%
  count(start_station_name, sort = TRUE)

start_station <- start_station[c(1,2,3,4,5,6,7,8,9,10), ]

print(start_station)
```

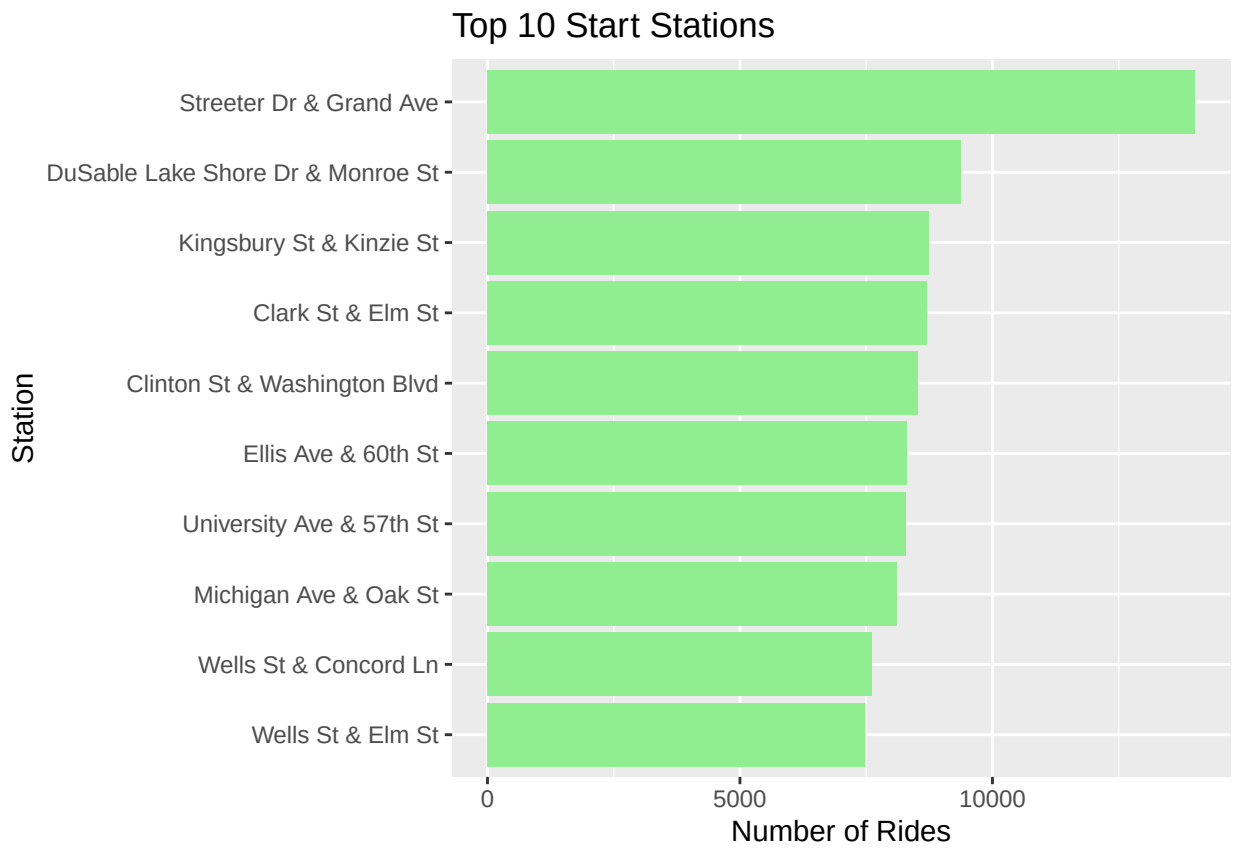
```
## # A tibble: 10 x 2
##    start_station_name                    n
##    <chr>                             <int>
##  1 Streeter Dr & Grand Ave           14012
##  2 DuSable Lake Shore Dr & Monroe St  9382
##  3 Kingsbury St & Kinzie St           8749
##  4 Clark St & Elm St                  8706
##  5 Clinton St & Washington Blvd       8517
##  6 Ellis Ave & 60th St                8303
##  7 University Ave & 57th St           8283
##  8 Michigan Ave & Oak St              8104
##  9 Wells St & Concord Ln              7615
## 10 Wells St & Elm St                  7473
```

```
# Determine top 10 stations trips are ENDED at
end_station <- new_data %>%
```

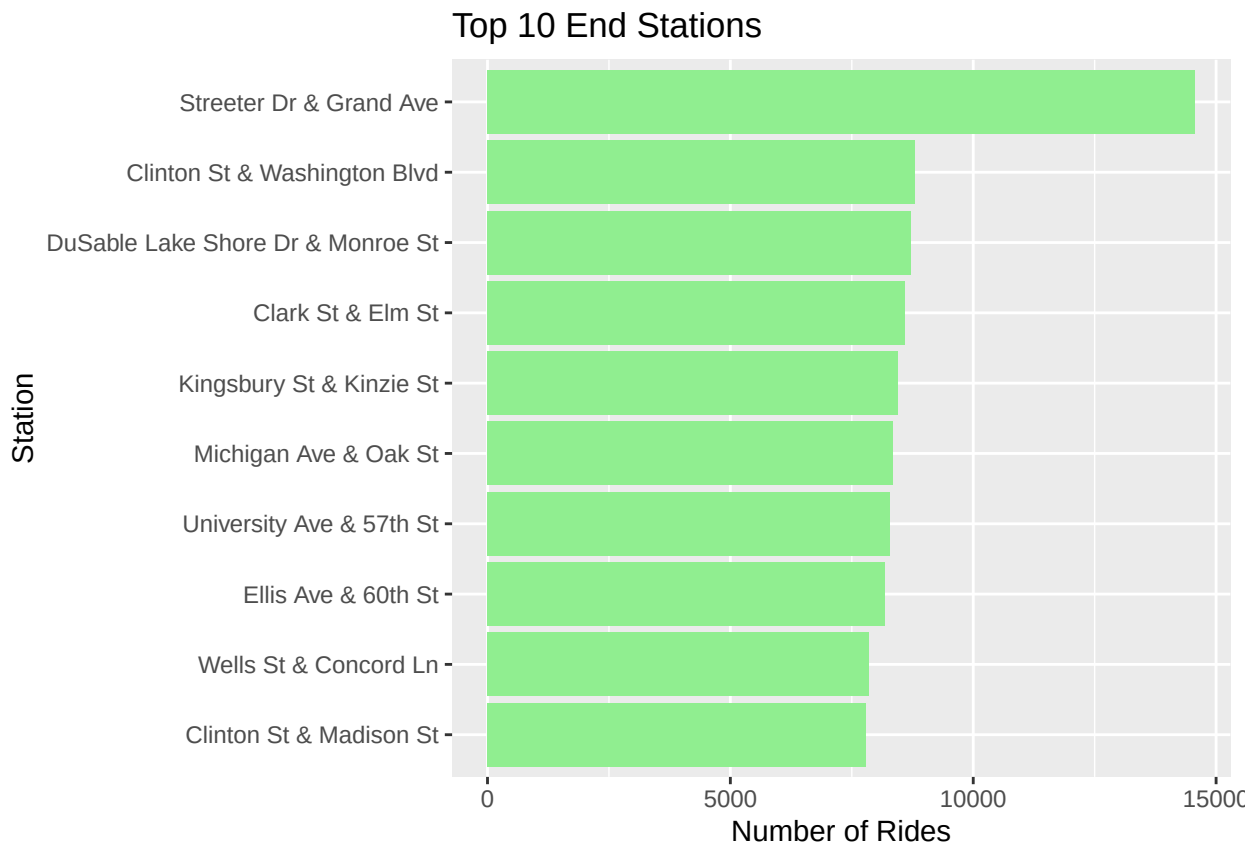
```
  select(end_station_name) %>%
  count(end_station_name, sort = TRUE)

end_station <- end_station[c(1,2,3,4,5,6,7,8,9,10), ]

print(end_station)
```

```
## # A tibble: 10 x 2
##    end_station_name                      n
##    <chr>                             <int>
##  1 Streeter Dr & Grand Ave           14566
##  2 Clinton St & Washington Blvd       8797
##  3 DuSable Lake Shore Dr & Monroe St  8725
##  4 Clark St & Elm St                  8592
##  5 Kingsbury St & Kinzie St           8446
##  6 Michigan Ave & Oak St              8357
##  7 University Ave & 57th St           8295
##  8 Ellis Ave & 60th St                8189
##  9 Wells St & Concord Ln              7861
## 10 Clinton St & Madison St            7792
```

```
# Visualization for high traffic STARTING and ENDING stations

start_station %>%
  mutate(start_station_name = fct_reorder(start_station_name, n)) %>%
  ggplot(aes(x = start_station_name, y = n)) +
  geom_col(fill = "lightgreen") +
  coord_flip() +
  labs(title = "Top 10 Start Stations", x = "Station", y = "Number of Rides")
```




```
end_station %>%
  mutate(end_station_name = fct_reorder(end_station_name, n)) %>%
  ggplot(aes(x = end_station_name, y = n)) +
  geom_col(fill = "lightgreen") +
  coord_flip() +
  labs(title = "Top 10 End Stations", x = "Station", y = "Number of Rides")
```



Act

Conclusion

- Throughout all months, we see more members riding compared to casual riders.
- Casual riders go on longer trips.
- Members ride more on weekdays, but casual riders also ride a lot on the weekends (Saturday and Sunday) compared to the rest of the week.
- Casual riders cover more distance.

Suggestions

- Offer discounts or rewards based on the monthly bike usage.
- Encourage them with rewards when they reach certain milestones to motivate more bike usage.
- A marketing campaign aimed at residents not using Cyclistic bikes for commuting might encourage them to go for annual memberships.