Cyclistic - Google Data Analytics Capstone Project

Ishan Amin

2022-05-09

Background Information

Cyclisitc is a bike-sharing company based out of Chicago. It currently offers 5824 bikes for its riders and has 692 mounting stations all across Chicago. The company offers three plans to its riders: The single-ride pass, full-day pass, and the annual membership.

The company currently categorizes its customers into two segments, the annual members and the casual riders (those that use the single-ride and full-day passes). The Director of Marketing wants to make a strong push to maximize the number of annual memberships, as that is what he believes is the driving factor for the sucess of the company.

Ask

Business Objective

The main objective of the business is to covert casual riders to annual members in order to increase revenue.

Business Task

In order to accomplish the business objective, Cyclistic must first understand the differences between their 2 customers.

###Stakeholders

Lily Moreno: Director of Marketing, responsible for the promotion of the bike sharing program through campaigns and initiatives.

Analytics Team: Data Analysts who collect, analyse, and report data to help guide marketing decisions and strategy.

Executive Team: Responsible for approving any recommendations that come out of this report.

Prepare

To conduct this investigation, we will be using information from April 2021 to April 2022.

The files have csv formatting with 13 columns: * ride_id: Unique ID for each ride. * rideable_type: Type of bicycle used (docked, electric, classical). * started_at: datetime of when the ride started. * ended_at: datetime of when the ride ended. * start_station_name: Name of the station where the customer started the ride. * start_station_id: ID of the station the rider picked up the bike. * end_station_name: Name of the station where the customer ended the ride. * end_station_id: ID of the station the rider dropped off the bike. * start_lat: Starting latitude of the ride. * start_lng: Starting longitude of the ride. * end_lat: Ending latitude of the ride. * end_lng: Ending longitude of the ride. * member_casual: Type of membership of the rider (member, casual)

The data is located at the following link: https://divvy-tripdata.s3.amazonaws.com/index.html

The data was collected by Motivate International Inc. and is available for public use. However, as per the licensing agreement (https://ride.divvybikes.com/data-license-agreement), all identifiable information has been removed from the data set. This poses several challenges, the first being that we won't be able to identify any demographic information about the riders, making it harder to market to a specific target audience. It also hinders the investigation considerably as it will not allow us to determine how frequently casual riders use the services, or if they live in the Chicago area.

Process

Loading the packages:

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --
```

```
## v ggplot2 3.3.6     v purrr   0.3.4
## v tibble  3.1.7     v dplyr   1.0.9
## v tidyr   1.2.0     v stringr 1.4.0
## v readr   2.1.2     v forcats 0.5.1
```

```
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(dplyr)
library(lubridate)
```

```
##
## Attaching package: 'lubridate'
```

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

```
library(skimr)
library(janitor)
```

```
##
## Attaching package: 'janitor'
```

```
## The following objects are masked from 'package:stats':
##
##     chisq.test, fisher.test
```

```
library(mapview)
```

Loading the data:

```
Apr_22 <- read_csv("202204-divvy-tripdata.csv")
```

```
## Rows: 371249 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.
```

```
Mar_22 <- read_csv("202203-divvy-tripdata.csv")
```

```
## Rows: 284042 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.
```

```
Feb_22 <- read_csv("202202-divvy-tripdata.csv")
```

```
## Rows: 115609 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.
```

```
Jan_22 <- read_csv("202201-divvy-tripdata.csv")
```

```
## Rows: 103770 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.
```

```
Dec_21 <- read_csv("202112-divvy-tripdata.csv")
```

```
## Rows: 247540 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.
```

```
Nov_21 <- read_csv("202111-divvy-tripdata.csv")
```

```
## Rows: 359978 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.
```

```
Oct_21 <- read_csv("202110-divvy-tripdata.csv")
```

```
## Rows: 631226 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.
```

```
Sep_21 <- read_csv("202109-divvy-tripdata.csv")
```

```
## Rows: 756147 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.
```

```
Aug_21 <- read_csv("202108-divvy-tripdata.csv")
```

```
## Rows: 804352 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.
```

```
Jul_21 <- read_csv("202107-divvy-tripdata.csv")
```

```
## Rows: 822410 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.
```

```
Jun_21 <- read_csv("202106-divvy-tripdata.csv")
```

```
## Rows: 729595 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.
```

```
May_21 <- read_csv("202105-divvy-tripdata.csv")
```

```
## Rows: 531633 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.
```

To make sure all the column names are the same, we compare the columns of each of the data frames:

```
compare_df_cols(May_21,Jun_21,Jul_21,Aug_21,Sep_21,Oct_21,Nov_21,Dec_21,Jan_22,Mar_22,Apr_22)
```

```
##           column_name          May_21          Jun_21          Jul_21
## 1             end_lat         numeric         numeric         numeric
## 2             end_lng         numeric         numeric         numeric
## 3      end_station_id       character       character       character
## 4    end_station_name       character       character       character
## 5           ended_at POSIXct, POSIXt POSIXct, POSIXt POSIXct, POSIXt
## 6       member_casual       character       character       character
## 7             ride_id       character       character       character
## 8       rideable_type       character       character       character
## 9           start_lat         numeric         numeric         numeric
## 10          start_lng         numeric         numeric         numeric
## 11   start_station_id       character       character       character
## 12 start_station_name       character       character       character
## 13         started_at POSIXct, POSIXt POSIXct, POSIXt POSIXct, POSIXt
##             Aug_21          Sep_21          Oct_21          Nov_21
## 1          numeric         numeric         numeric         numeric
## 2          numeric         numeric         numeric         numeric
## 3        character       character       character       character
## 4        character       character       character       character
## 5  POSIXct, POSIXt POSIXct, POSIXt POSIXct, POSIXt POSIXct, POSIXt
## 6        character       character       character       character
## 7        character       character       character       character
## 8        character       character       character       character
## 9          numeric         numeric         numeric         numeric
## 10         numeric         numeric         numeric         numeric
## 11       character       character       character       character
## 12       character       character       character       character
## 13 POSIXct, POSIXt POSIXct, POSIXt POSIXct, POSIXt POSIXct, POSIXt
##             Dec_21          Jan_22          Mar_22          Apr_22
## 1          numeric         numeric         numeric         numeric
## 2          numeric         numeric         numeric         numeric
## 3        character       character       character       character
## 4        character       character       character       character
## 5  POSIXct, POSIXt POSIXct, POSIXt POSIXct, POSIXt POSIXct, POSIXt
```

```
## 6        character       character       character       character
## 7        character       character       character       character
## 8        character       character       character       character
## 9          numeric         numeric         numeric         numeric
## 10         numeric         numeric         numeric         numeric
## 11       character       character       character       character
## 12       character       character       character       character
## 13 POSIXct, POSIXt POSIXct, POSIXt POSIXct, POSIXt POSIXct, POSIXt
```

As all the column names are the same, and all data frames match the type, the data frames were combined into one.

```
trips <- bind_rows(May_21,Jun_21,Jul_21,Aug_21,Sep_21,Oct_21,Nov_21,Dec_21,Jan_22,Mar_22,Apr_22)
glimpse(trips)
```

```
## Rows: 5,641,942
## Columns: 13
## $ ride_id            <chr> "C809ED75D6160B2A", "DD59FDCE0ACACAF3", "0AB83CB88C~
## $ rideable_type      <chr> "electric_bike", "electric_bike", "electric_bike", ~
## $ started_at         <dttm> 2021-05-30 11:58:15, 2021-05-30 11:29:14, 2021-05-~
## $ ended_at           <dttm> 2021-05-30 12:10:39, 2021-05-30 12:14:09, 2021-05-~
## $ start_station_name <chr> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ start_station_id   <chr> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ end_station_name   <chr> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ end_station_id     <chr> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ start_lat          <dbl> 41.90000, 41.88000, 41.92000, 41.92000, 41.94000, 4~
## $ start_lng          <dbl> -87.63000, -87.62000, -87.70000, -87.70000, -87.690~
## $ end_lat            <dbl> 41.89000, 41.79000, 41.92000, 41.94000, 41.94000, 4~
## $ end_lng            <dbl> -87.61000, -87.58000, -87.70000, -87.69000, -87.700~
## $ member_casual      <chr> "casual", "casual", "casual", "casual", "casual", "~
```

Remove all the data with empty rows and columns

```
trips = trips %>%
  remove_empty(which = c("cols", "rows"))
```

Checking for duplicates:

```
#get_dupes(trips,ride_id)
```

No duplicates to report.

The company also conducts testing on their docking stations. The following code ensures that the tests are not a part of the analysis.

Renamed columns for better comprehension:

```
trips = trips %>%
  rename(
    bike_type = rideable_type,
    user_type = member_casual
  ) %>%
  mutate(
    bike_type = as_factor(bike_type),
    user_type = as_factor(user_type)
  )
```

To see how many null values there are in the data set.

```
colSums(is.na(trips))
```

```
##            ride_id          bike_type         started_at           ended_at
##                  0                  0                  0                  0
## start_station_name   start_station_id   end_station_name     end_station_id
##             771627             771624             823006             823006
##          start_lat          start_lng            end_lat            end_lng
##                  0                  0               4689               4689
##          user_type
##                  0
```

From this we can see that there is an alarming number of missing values for both the start stations and the end stations. Out of the 5,641,942 unique bike rides over the past 12 months, it is unclear where 13.67% of rides originated from and where 14.58% of the rides ended.

In terms of the end latitude and longitude, it can be assumed that the riders did not return their bikes to the station, and as such the longitude and latitude was not registered.

To determine what time riders most often use the bike, I created new columns to better understand and track this data. I also created a column to track the amount of time, in minutes, each ride lasted.

```
trips = trips %>%
  mutate(
    hour_start = hour(started_at),
    weekday = wday(started_at, label = T, abbr = F),
    month = month(started_at, label = T, abbr =F),
    day = day(started_at),
    duration = difftime(ended_at, started_at, units = "mins")
  )

glimpse(trips)
```

```
## Rows: 5,641,942
## Columns: 18
## $ ride_id            <chr> "C809ED75D6160B2A", "DD59FDCE0ACACAF3", "0AB83CB88C~
## $ bike_type          <fct> electric_bike, electric_bike, electric_bike, electr~
## $ started_at         <dttm> 2021-05-30 11:58:15, 2021-05-30 11:29:14, 2021-05-~
## $ ended_at           <dttm> 2021-05-30 12:10:39, 2021-05-30 12:14:09, 2021-05-~
## $ start_station_name <chr> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ start_station_id   <chr> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ end_station_name   <chr> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ end_station_id     <chr> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ start_lat          <dbl> 41.90000, 41.88000, 41.92000, 41.92000, 41.94000, 4~
## $ start_lng          <dbl> -87.63000, -87.62000, -87.70000, -87.70000, -87.690~
## $ end_lat            <dbl> 41.89000, 41.79000, 41.92000, 41.94000, 41.94000, 4~
## $ end_lng            <dbl> -87.61000, -87.58000, -87.70000, -87.69000, -87.700~
## $ user_type          <fct> casual, casual, casual, casual, casual, casual, cas~
## $ hour_start         <int> 11, 11, 14, 14, 18, 11, 10, 13, 11, 19, 16, 0, 16, ~
## $ weekday            <ord> Sunday, Sunday, Sunday, Sunday, Sunday, Sunday, Sun~
## $ month              <ord> May, May, May, May, May, May, May, May, May, May, M~
## $ day                <int> 30, 30, 30, 30, 30, 30, 30, 5, 5, 4, 5, 31, 31, 30,~
## $ duration           <drtn> 12.400000 mins, 44.916667 mins, 1.200000 mins, 15.~
```

Furthermore, logic dictates that no trip can be under 0 minutes, as such a filter was created to remove bad data collection:

```
trips = filter(trips,duration > 0)
```

Finally, a descriptive analysis is conducted on the dataframe:

```
str(trips)
```

```
## tibble [5,641,295 x 18] (S3: tbl_df/tbl/data.frame)
##  $ ride_id           : chr [1:5641295] "C809ED75D6160B2A" "DD59FDCE0ACACAF3" "0AB83CB88C43EFC2" "788
##  $ bike_type         : Factor w/ 3 levels "electric_bike",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ started_at        : POSIXct[1:5641295], format: "2021-05-30 11:58:15" "2021-05-30 11:29:14" ...
##  $ ended_at          : POSIXct[1:5641295], format: "2021-05-30 12:10:39" "2021-05-30 12:14:09" ...
##  $ start_station_name: chr [1:5641295] NA NA NA NA ...
##  $ start_station_id  : chr [1:5641295] NA NA NA NA ...
##  $ end_station_name  : chr [1:5641295] NA NA NA NA ...
##  $ end_station_id    : chr [1:5641295] NA NA NA NA ...
##  $ start_lat         : num [1:5641295] 41.9 41.9 41.9 41.9 41.9 ...
##  $ start_lng         : num [1:5641295] -87.6 -87.6 -87.7 -87.7 -87.7 ...
##  $ end_lat           : num [1:5641295] 41.9 41.8 41.9 41.9 41.9 ...
##  $ end_lng           : num [1:5641295] -87.6 -87.6 -87.7 -87.7 -87.7 ...
##  $ user_type         : Factor w/ 2 levels "casual","member": 1 1 1 1 1 1 1 1 1 1 ...
##  $ hour_start        : int [1:5641295] 11 11 14 14 18 11 10 13 11 19 ...
##  $ weekday           : Ord.factor w/ 7 levels "Sunday"<"Monday"<..: 1 1 1 1 1 1 1 4 4 3 ...
##  $ month             : Ord.factor w/ 12 levels "January"<"February"<..: 5 5 5 5 5 5 5 5 5 5 ...
##  $ day               : int [1:5641295] 30 30 30 30 30 30 30 5 5 4 ...
##  $ duration          : 'difftime' num [1:5641295] 12.4 44.9166666666667 1.2 15.2166666666667 ...
##   ..- attr(*, "units")= chr "mins"
```

```
summary(trips)
```

```
##    ride_id                  bike_type         started_at
##  Length:5641295     electric_bike:2208167   Min.   :2021-05-01 00:00:11.00
##  Class :character   classic_bike :3143106   1st Qu.:2021-07-06 09:55:00.00
##  Mode  :character   docked_bike  : 290022   Median :2021-08-29 04:40:26.00
##                                             Mean   :2021-09-15 16:13:03.18
##                                             3rd Qu.:2021-10-29 11:22:23.00
##                                             Max.   :2022-04-30 23:59:54.00
##
##     ended_at                      start_station_name start_station_id
##  Min.   :2021-05-01 00:03:26.00   Length:5641295     Length:5641295
##  1st Qu.:2021-07-06 10:24:37.50   Class :character   Class :character
##  Median :2021-08-29 05:21:31.00   Mode  :character   Mode  :character
##  Mean   :2021-09-15 16:34:20.13
##  3rd Qu.:2021-10-29 11:36:02.00
##  Max.   :2022-05-02 00:35:01.00
##
##  end_station_name   end_station_id       start_lat       start_lng
##  Length:5641295     Length:5641295     Min.   :41.64   Min.   :-87.84
##  Class :character   Class :character   1st Qu.:41.88   1st Qu.:-87.66
##  Mode  :character   Mode  :character   Median :41.90   Median :-87.64
##                                        Mean   :41.90   Mean   :-87.65
##                                        3rd Qu.:41.93   3rd Qu.:-87.63
##                                        Max.   :45.64   Max.   :-73.80
##
##     end_lat         end_lng        user_type         hour_start
##  Min.   :41.39   Min.   :-88.97   casual:2514588   Min.   : 0.00
```

```
##  1st Qu.:41.88   1st Qu.:-87.66   member:3126707   1st Qu.:11.00
##  Median :41.90   Median :-87.64                    Median :15.00
##  Mean   :41.90   Mean   :-87.65                    Mean   :14.21
##  3rd Qu.:41.93   3rd Qu.:-87.63                    3rd Qu.:18.00
##  Max.   :42.17   Max.   :-87.49                    Max.   :23.00
##  NA's   :4689    NA's   :4689
##       weekday             month              day          duration
##  Sunday   :849025   July     : 822328   Min.   : 1.00   Length:5641295
##  Monday   :711816   August   : 804245   1st Qu.: 8.00   Class :difftime
##  Tuesday  :750108   September: 756040   Median :15.00   Mode  :numeric
##  Wednesday:774501   June     : 729529   Mean   :15.46
##  Thursday :770334   October  : 631156   3rd Qu.:23.00
##  Friday   :796743   May      : 531579   Max.   :31.00
##  Saturday :988768   (Other)  :1366418
```

```
skim_without_charts(trips)
```

Table 1: Data summary

Name	trips
Number of rows	5641295
Number of columns	18
Column type frequency:	
character	5
difftime	1
factor	4
numeric	6
POSIXct	2
Group variables	None

Variable type: character

skim_variable	n_missing	complete_rate	min	max	empty	n_unique	whitespace
ride_id	0	1.00	16	16	0	5641295	0
start_station_name	771604	0.86	3	53	0	865	0
start_station_id	771601	0.86	3	44	0	856	0
end_station_name	822578	0.85	10	53	0	864	0
end_station_id	822578	0.85	3	44	0	856	0

Variable type: difftime

skim_variable	n_missing	complete_rate	min	max	median	n_unique
duration	0	1	0.02 mins	55944.15 mins	11.6 mins	25120

Variable type: factor

skim_variable	n_missing	complete_rate	ordered	n_unique	top_counts
bike_type	0	1	FALSE	3	cla: 3143106, ele: 2208167, doc: 290022
user_type	0	1	FALSE	2	mem: 3126707, cas: 2514588
weekday	0	1	TRUE	7	Sat: 988768, Sun: 849025, Fri: 796743, Wed: 774501
month	0	1	TRUE	11	Jul: 822328, Aug: 804245, Sep: 756040, Jun: 729529

Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100
start_lat	0	1	41.90	0.05	41.64	41.88	41.90	41.93	45.64
start_lng	0	1	-87.65	0.03	-87.84	-87.66	-87.64	-87.63	-73.80
end_lat	4689	1	41.90	0.05	41.39	41.88	41.90	41.93	42.17
end_lng	4689	1	-87.65	0.03	-88.97	-87.66	-87.64	-87.63	-87.49
hour_start	0	1	14.21	5.08	0.00	11.00	15.00	18.00	23.00
day	0	1	15.46	8.74	1.00	8.00	15.00	23.00	31.00

Variable type: POSIXct

skim_variable	n_missing	complete_rate	min	max	median	n_unique
started_at	0	1	2021-05-01 00:00:11	2022-04-30 23:59:54	2021-08-29 04:40:26	4709965
ended_at	0	1	2021-05-01 00:03:26	2022-05-02 00:35:01	2021-08-29 05:21:31	4704795

The two most important factors to consider when conducting the analysis is the time and location of the rides, this is probably the most important information that we have from the data provided.

In order to further analyse these aspects of the data, we can create two additional data frames.

```
time_trips = trips %>%
  select(ride_id, user_type, bike_type, hour_start, weekday, month, day, duration)

colSums(is.na(time_trips))
```

```
##    ride_id  user_type  bike_type hour_start    weekday      month        day
##          0          0          0          0          0          0          0
##   duration
##          0
```

```
location_trips = trips %>%
  drop_na(start_station_name, end_station_name) %>%
  drop_na(end_lat,end_lng) %>%
  select(ride_id, start_station_name, end_station_name, start_lat, start_lng, end_lat, end_lng, user_ty

colSums(is.na(location_trips))
```

```
##            ride_id start_station_name   end_station_name          start_lat
##                  0                  0                  0                  0
##          start_lng            end_lat            end_lng          user_type
```

```
##                 0                 0                 0                 0
##          duration
##                 0
```

Analyse and Share

Time Analysis

To determine how long the casual riders rode bikes compared to the members, I first obtained summary statistics on the duration of the trips.

```
mean(time_trips$duration)
```

```
## Time difference of 21.2825 mins
```

```
median(time_trips$duration)
```

```
## Time difference of 11.6 mins
```

```
max(time_trips$duration)
```

```
## Time difference of 55944.15 mins
```

```
min(time_trips$duration)
```

```
## Time difference of 0.01666667 mins
```

Comparing members and casual riders

```
aggregate(time_trips$duration ~ time_trips$user_type, FUN = mean)
```

```
##   time_trips$user_type time_trips$duration
## 1               casual       31.33473 mins
## 2               member       13.19820 mins
```

```
aggregate(time_trips$duration ~ time_trips$user_type, FUN = median)
```

```
##   time_trips$user_type time_trips$duration
## 1               casual       15.61667 mins
## 2               member        9.25000 mins
```

```
aggregate(time_trips$duration ~ time_trips$user_type, FUN = max)
```

```
##   time_trips$user_type time_trips$duration
## 1               casual       55944.15 mins
## 2               member        1559.90 mins
```

```
aggregate(time_trips$duration ~ time_trips$user_type, FUN = min)
```

```
##   time_trips$user_type time_trips$duration
## 1               casual     0.01666667 mins
## 2               member     0.01666667 mins
```

The average ride time by each day for members vs casual users

```
aggregate(time_trips$duration ~ time_trips$user_type + time_trips$weekday, FUN = mean)
```

```
##    time_trips$user_type time_trips$weekday time_trips$duration
## 1                casual             Sunday       37.00891 mins
## 2                member             Sunday       15.15032 mins
## 3                casual             Monday       31.16528 mins
```

```
## 4                member             Monday       12.77111 mins
## 5                casual            Tuesday       26.46311 mins
## 6                member            Tuesday       12.29403 mins
## 7                casual          Wednesday       27.13261 mins
## 8                member          Wednesday       12.46359 mins
## 9                casual           Thursday       27.89279 mins
## 10               member           Thursday       12.47280 mins
## 11               casual             Friday       29.26609 mins
## 12               member             Friday       12.91149 mins
## 13               casual           Saturday       34.23300 mins
## 14               member           Saturday       14.85206 mins
```

Analyze ridership data by type and weekday

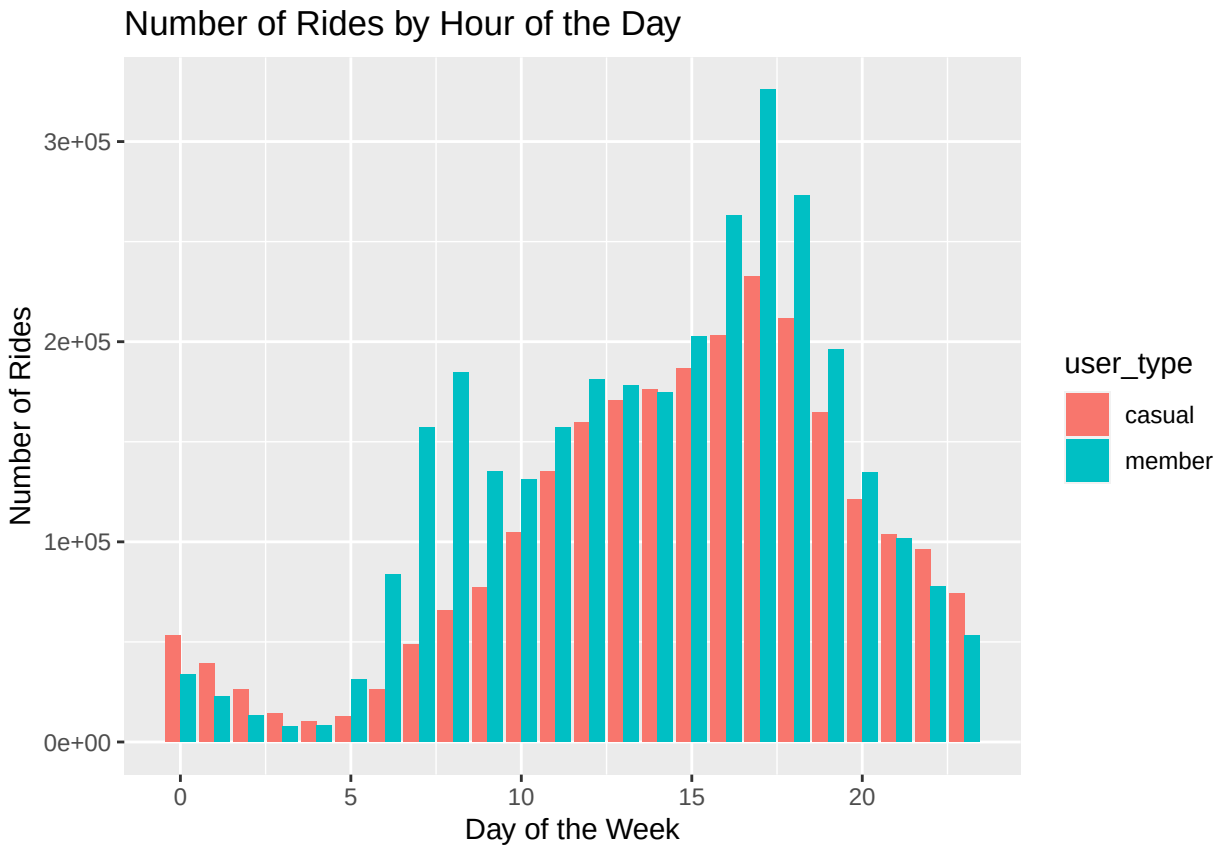
```
time_trips %>%
  group_by(user_type, weekday) %>%  #groups by usertype and weekday
  summarise(number_of_rides = n()                              #calculates the number of rides and average
  ,average_duration = mean(duration))       # calculates the average duration
```

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

```
## # A tibble: 14 x 4
## # Groups:   user_type [2]
##    user_type weekday   number_of_rides average_duration
##    <fct>     <ord>               <int> <drtn>
##  1 casual    Sunday             472730 37.00891 mins
##  2 casual    Monday             284586 31.16528 mins
##  3 casual    Tuesday            267722 26.46311 mins
##  4 casual    Wednesday          282210 27.13261 mins
##  5 casual    Thursday           296155 27.89279 mins
##  6 casual    Friday             355459 29.26609 mins
##  7 casual    Saturday           555726 34.23300 mins
##  8 member    Sunday             376295 15.15032 mins
##  9 member    Monday             427230 12.77111 mins
## 10 member    Tuesday            482386 12.29403 mins
## 11 member    Wednesday          492291 12.46359 mins
## 12 member    Thursday           474179 12.47280 mins
## 13 member    Friday             441284 12.91149 mins
## 14 member    Saturday           433042 14.85206 mins
```

```
time_trips %>%
  group_by(user_type, hour_start) %>%
  summarise(number_of_rides = n()
            ,average_duration = mean(duration)) %>%
  ggplot(aes(x = hour_start, y = number_of_rides, fill = user_type)) +
  geom_col(position = "dodge") +
  labs(
    title = "Number of Rides by Hour of the Day",
    x = "Day of the Week",
    y = "Number of Rides"
  )
```

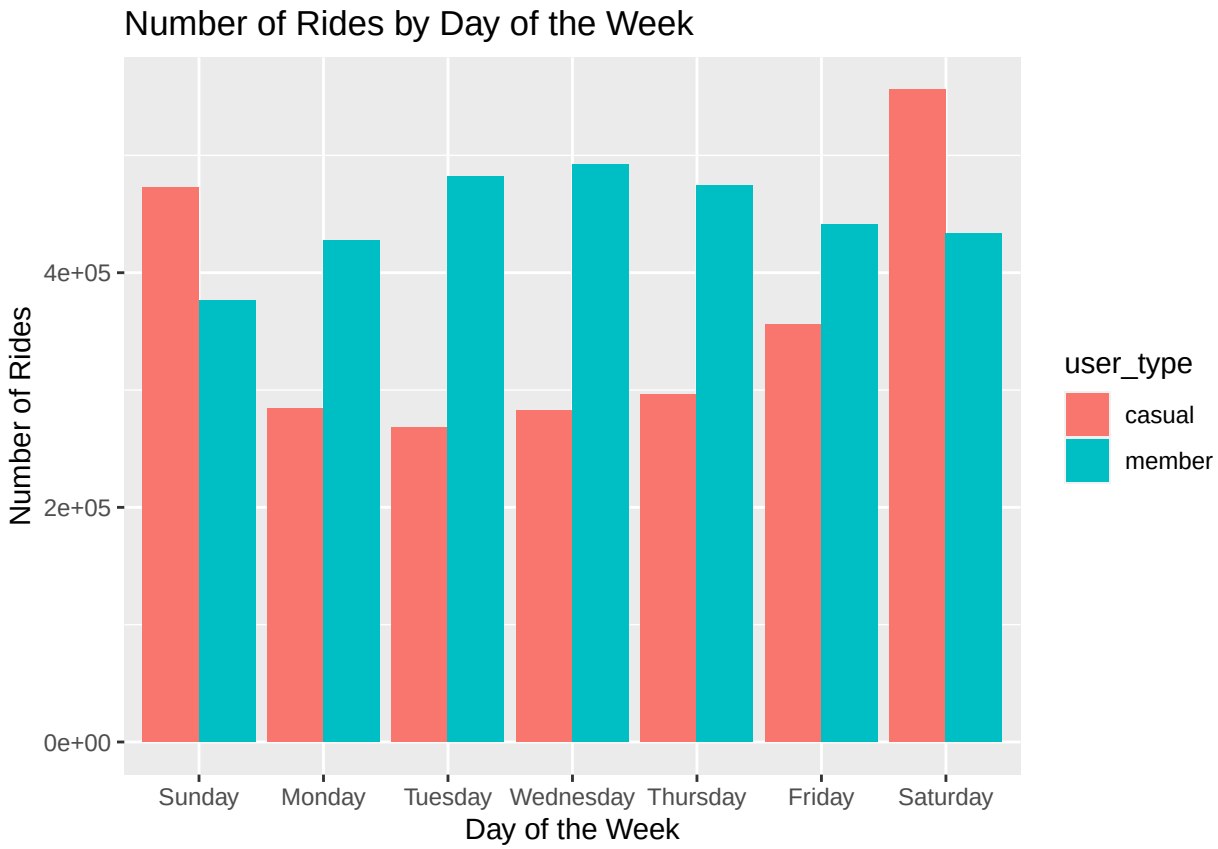
```
## `summarise()` has grouped output by 'user_type'. You can override using the
## `.groups` argument.
```



Casual members are more active between late mornings and early evenings.

```
time_trips %>%
  group_by(user_type, weekday) %>%
  summarise(number_of_rides = n()
            ,average_duration = mean(duration)) %>%
  ggplot(aes(x = weekday, y = number_of_rides, fill = user_type)) +
  geom_col(position = "dodge") +
  labs(
    title = "Number of Rides by Day of the Week",
    x = "Day of the Week",
    y = "Number of Rides"
  )
```

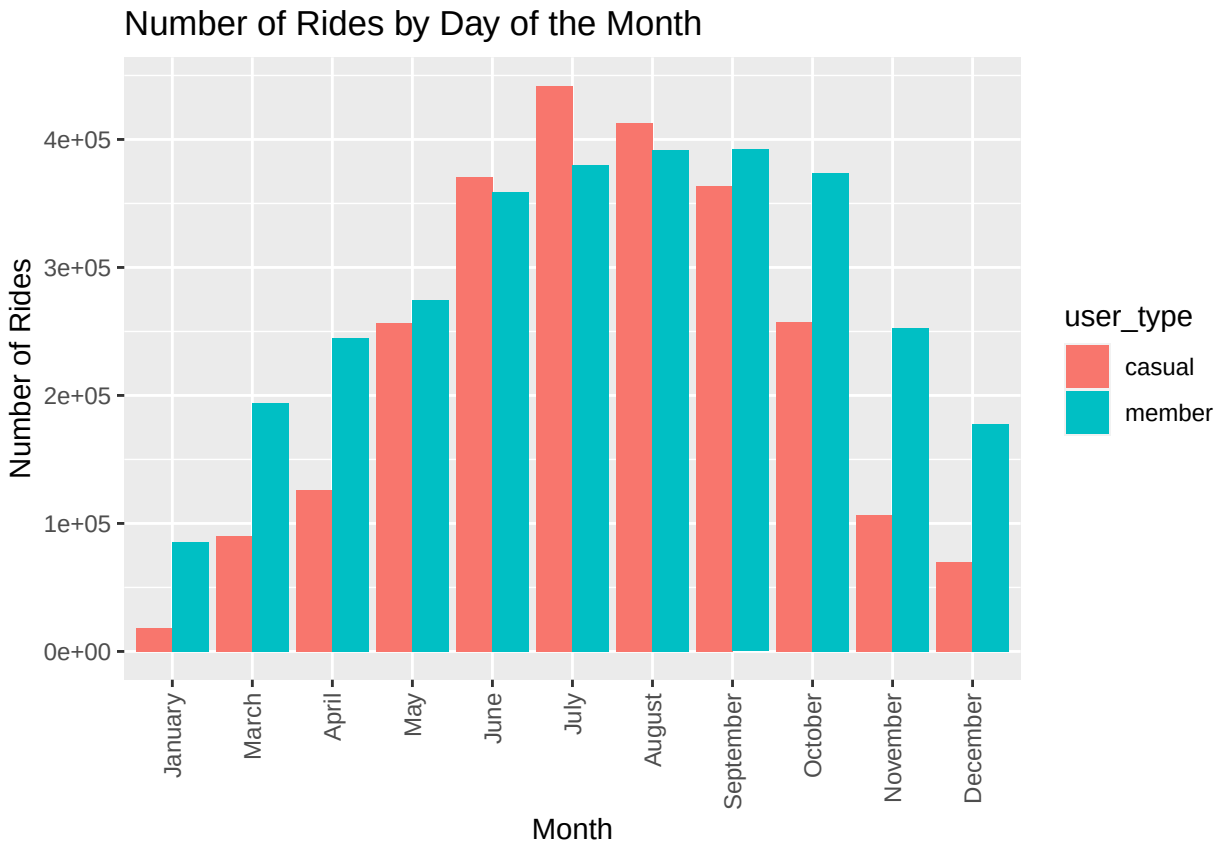
```
## `summarise()` has grouped output by 'user_type'. You can override using the
## `.groups` argument.
```



From this graph, it is evident that casual riders use Cyclistic's bikes on the weekend more than the weekdays. Members and casual riders have a inverse relationship in terms of bike usage throughout the week.

```
time_trips %>%
  group_by(user_type, month) %>%
  summarise(number_of_rides = n()
            ,average_duration = mean(duration)) %>%
  ggplot(aes(x = month, y = number_of_rides, fill = user_type)) +
  geom_col(position = "dodge") +
  labs(title = "Number of Rides by Day of the Month", x = "Month",y = "Number of Rides") +
  theme(axis.text.x = element_text(angle = 90, vjust = 0.5, hjust=1))
```

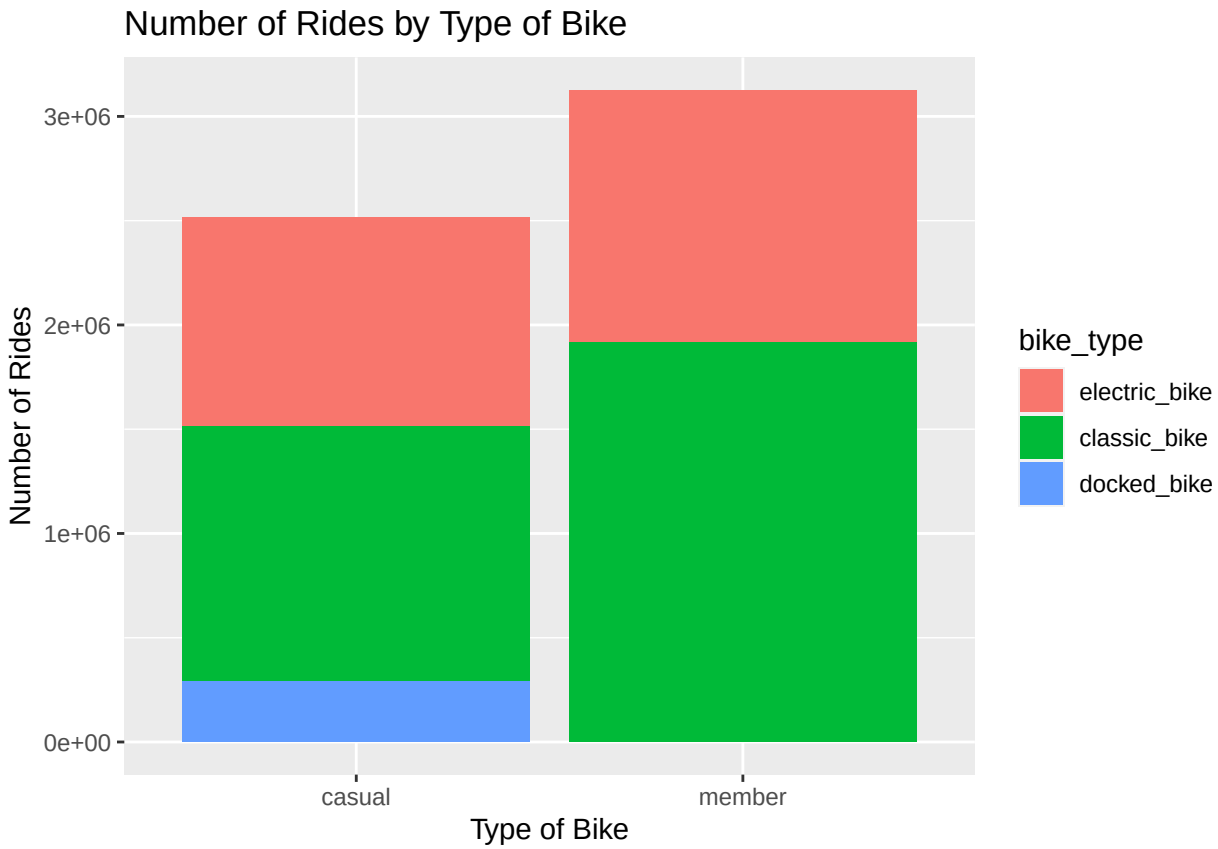
```
## `summarise()` has grouped output by 'user_type'. You can override using the
## `.groups` argument.
```



The distribution of both casual riders and members follows a normal distribution as it relates to the number of rides taken throughout the year. Casual riders ride more than members in the summer months: June, July, and August. This may due to an influx of tourists in the area, and their desire to explore city on bike.

```
time_trips %>%
  group_by(user_type, bike_type) %>%
  summarise(number_of_rides = n()
            ,average_duration = mean(duration)) %>%
  ggplot(aes(x = user_type, y = number_of_rides, fill = bike_type)) +
  geom_bar(position = "stack", stat = "identity") +
  labs(
    title = "Number of Rides by Type of Bike",
    x = "Type of Bike",
    y = "Number of Rides"
  )
```

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

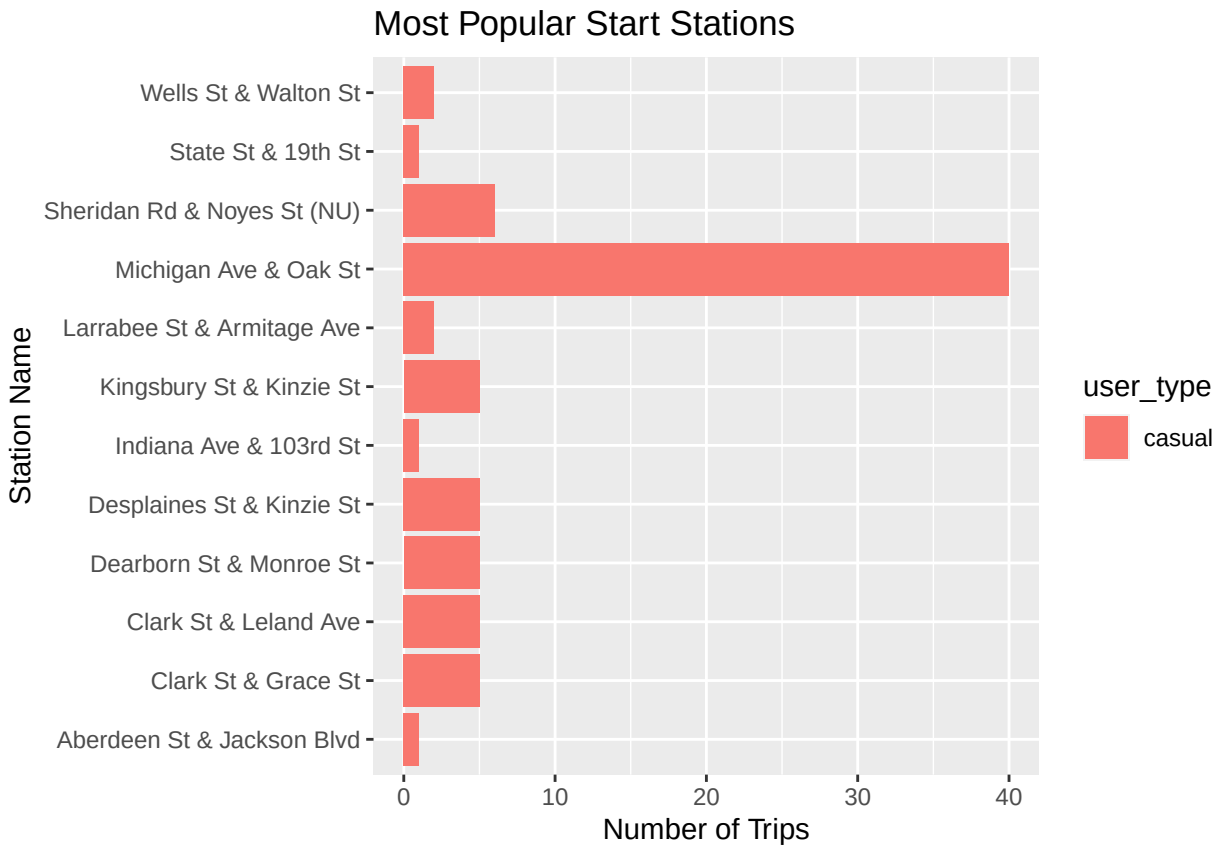


From this we can see that both groups have a preference for classic bikes, with electric bikes being a close second. However, we see that few casual riders choose docked bikes, and no members like docked bikes.

Location Analysis

```
location_trips [1:100, ] %>%
  group_by(user_type, start_station_name, start_lat, start_lng) %>%
  summarise(number_of_rides = n()) %>%
  ggplot(aes(start_station_name, number_of_rides, fill = user_type))+
  geom_col(position = "dodge")+
  coord_flip()+
  labs(
    title = "Most Popular Start Stations",
    x = "Station Name",
    y = "Number of Trips")
```

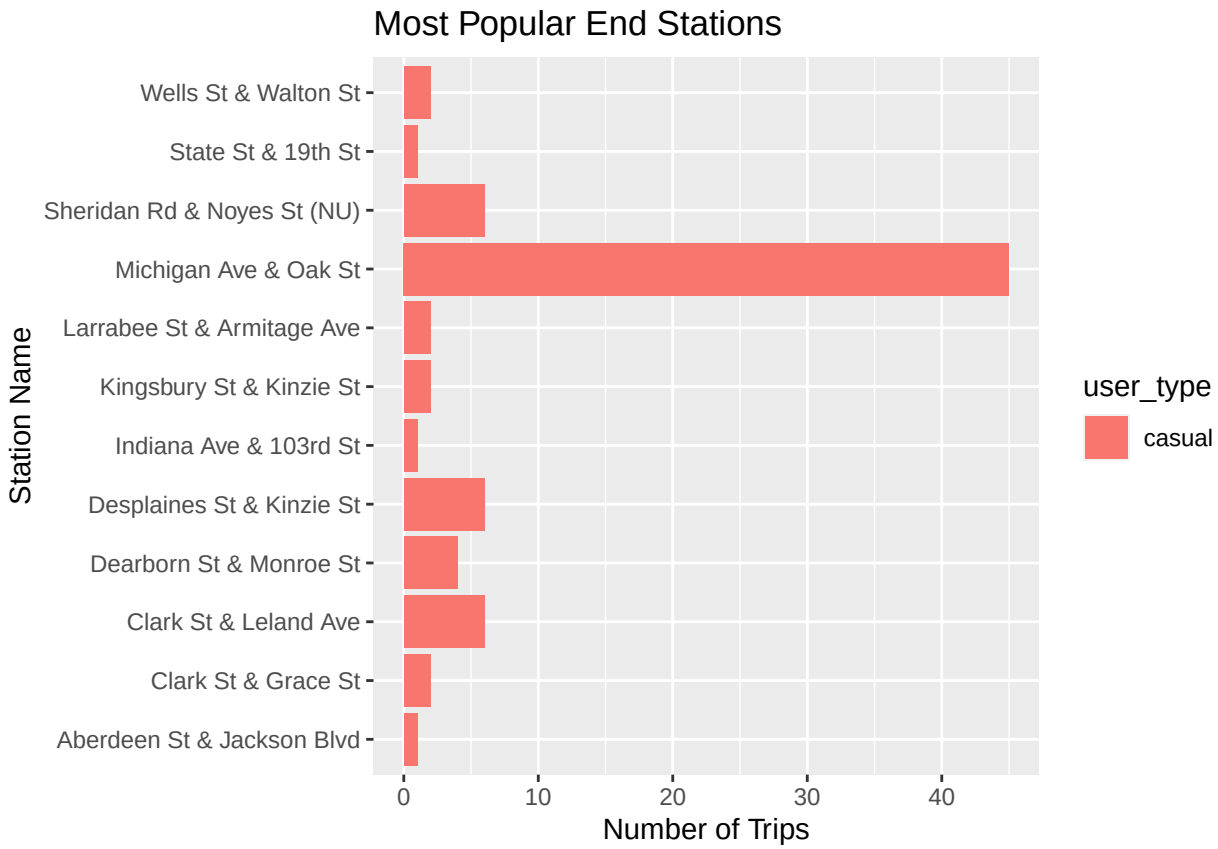
```
## `summarise()` has grouped output by 'user_type', 'start_station_name',
## 'start_lat'. You can override using the `.groups` argument.
```




```
#location_trips[1:100, ] %>%
#  group_by(user_type, start_station_name, start_lat, start_lng) %>%
#  summarise(number_of_rides = n()) %>%
#  mapview(
#    xcol = "start_lng",
#        ycol = "start_lat",
#        cex = "number_of_rides",
#        alpha = 0.9,
#        crs = 4269,
#        color = "#8b0000",
#        grid = F,
#        legend = T)
```

```
location_trips [1:100, ] %>%
  group_by(user_type, end_station_name, end_lat, end_lng) %>%
  summarise(number_of_rides = n()) %>%
  ggplot(aes(end_station_name, number_of_rides, fill = user_type))+
  geom_col(position = "dodge")+
  coord_flip()+
  labs(
    title = "Most Popular End Stations",
    x = "Station Name",
    y = "Number of Trips")
```

```
## `summarise()` has grouped output by 'user_type', 'end_station_name', 'end_lat'.
## You can override using the `.groups` argument.
```




```
#location_trips[1:100, ] %>%
#  group_by(user_type, end_station_name, end_lat, end_lng) %>%
#  summarise(number_of_rides = n()) %>%
#  mapview(
#    xcol = "end_lng",
#        ycol = "end_lat",
#        cex = "number_of_rides",
#        alpha = 0.9,
#        crs = 4269,
#        color = "#8b0000",
#        grid = F,
#        legend = T)
```

From the charts above, we see that the most active station for casual riders is the one located on Michigan Ave & Oak Street. We can also see that most of the locations for casual riders are near Lake Michigan.

Act

Reccomendations

Due to the lack of demographic information, it was difficult to determine how to specifically target the casual riders. However, the ride data did provide some insight into how the casual riders differed from the members, and how that can be used to market to them.

1. Cyclistic should increase advertisments around the Chicago Lake Front. From my analysis, the station on Michigan Ave & Oak Street recieved a lot of footfall from casual riders.
2. Cyclistic should ramp adverstisments during the summer months, namely June, July, August, and

September. These are when casual riders are the most active.

3. Only casual riders opt to use the docked bikes, interestingly, some of the longest trips in the data set are a result of the docked bikes. As such Cyclistic should create membership packages that would appeal to those who take longer bike rides.

```
#write_csv(trips, "all_trips.csv")
```