

# Cyclistic case study- Victor

Victor Lee

2023-08-16

## PHASE 1: IDENTIFYING BUSINESS TASK

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 across Chicago. The bikes can be unlocked from one station and returned to any other station in the system anytime.

Until now, Cyclistic's marketing strategy relied on building general awareness and appealing to broad consumer segments. One approach that helped make these things possible was the flexibility of its pricing plans: single-ride passes, full-day passes, and annual memberships. Customers who purchase single-ride or full-day passes are referred to as casual riders. Customers who purchase annual memberships are Cyclistic members.

Cyclistic's finance analysts have concluded that annual members are much more profitable than casual riders. Although the pricing flexibility helps Cyclistic attract more customers, Moreno believes that maximizing the number of annual members will be key to future growth. Rather than creating a marketing campaign that targets all-new customers, Moreno believes there is a very good chance to convert casual riders into members. She notes that casual riders are already aware of the Cyclistic program and have chosen Cyclistic for their mobility needs.

Moreno (The director of marketing) has set a clear goal: Design marketing strategies aimed at converting casual riders into annual members. In order to do that, however, the marketing analyst team needs to better understand how annual members and casual riders differ, why casual riders would buy a membership, and how digital media could affect their marketing tactics. Moreno and her team are interested in analyzing the Cyclistic historical bike trip data to identify trends. Moreno wants our team to answer: ***How do annual members and casual riders use Cyclistic bikes differently?***

The business question assigned to me is quite clear: "How do annual members and casual riders use Cyclistic bikes differently?" Cyclistic Company seeks to understand the main differences between these two user groups - casual riders (who pay for each ride) and annual members (who pay for a yearly subscription) - through data analysis. To address this question, I will examine Cyclistic's Rides data to gain insights into the distinct usage patterns of members and casual riders. Additionally, if there is sufficient data available, I will explore the factors that motivate casual riders to purchase Cyclistic annual memberships and consider how digital media can be utilized to influence them to become members. If data is insufficient, I will prioritize addressing these questions in subsequent analyses by collecting additional relevant data.

To approach this business question effectively, I will consider both the background information highlighting the higher profitability of annual members compared to casual riders and the dataset provided, which includes valuable information such as bike types, total number of rides, ride months, ride days, ride hours, latitude and longitude of stations, and member-causal type. Based on this information, I will formulate several hypotheses to guide my analysis:

H1: Members use bikes more often than casuals.

H2: Members' average ride duration is higher than casuals'.

H3: Member and casuals are significantly different in bike type choice.

H4: Due to the weather in Chicago, both members and casuals prefer to use bikes in summer than winter.

H5: Members' total rides per month are higher than casuals.

H6: Members' total ride duration per month is higher than casuals.

H7: Members are more active on weekdays rather than weekends, casual riders are highly active on weekends.

H8: Members' total ride duration is higher than casuals on weekdays while lower than casuals on weekends.

H9: Members' average ride duration is higher than casuals on weekdays while lower than casuals on weekends.

H10: Members have a ride peak at 8 mornings and at 17 afternoons during weekdays while casual users tend to ride more in the afternoon during weekends.

H11: Members' total rides per hour of the day are higher than casuals.

H12: Members' average ride duration per hour of the day is higher than casuals.

H13: Members' starting and ending locations are clustered in the downtown area, while casual customers' locations are more dispersed and along the beach.

The goal of this project is to get insights that could support the marketing director's strategy. My stakeholder in this project is the marketing director and I will report to her directly.

## PHASE 2: PREPARING and Combining DATA

The dataset provided for this case study project includes 39 '.csv' files representing each month from April 2020 to June 2023 and is located on the company's cloud storage (Amazon Web Services : <https://divvy-tripdata.s3.amazonaws.com/index.html> (<https://divvy-tripdata.s3.amazonaws.com/index.html>)).

I download these files to my local folder. As a practice, I plan to use R to process and analyse these data. To input these data into R, I setup my work directory with code in chunk 1.

Each table uses 13 columns ("ride\_id", "rideable\_type", "started\_at", "ended\_at", "start\_station\_name", "start\_station\_id", "end\_station\_name", "end\_station\_id", "start\_lat", "start\_lng", "end\_lat", "end\_lng", "member\_casual") to record each month's rides.

Further inspection of the files showed that there are multiple null values across tables, also columns have inconsistent naming and formatting that causes duplicates and the inability to merge tables without additional formatting.

Thus, to prepare data for cleaning, we need to change the type of two columns (start\_station\_id, end\_station\_id) of the first 8 months to match the most recent data.

Now that we have all the data in one place saved as a new virtual table, we can start to clean the data off possible inconsistencies and/or errors. We will also add new columns based on existing data calculations to get more in-depth insights about user behaviour.

## Setting Work Directory

## Adding Packages

```
library(tidyverse)
```

```
## — Attaching core tidyverse packages — tidyverse 2.0.0 —
## ✓ dplyr      1.1.2      ✓ readr      2.1.4
## ✓ forcats    1.0.0      ✓ stringr   1.5.0
## ✓ ggplot2    3.4.2      ✓ tibble    3.2.1
## ✓ lubridate  1.9.2      ✓ tidyr     1.3.0
## ✓ purrr      1.0.1
## — Conflicts — tidyverse_conflicts() —
## ✖ dplyr::filter() masks stats::filter()
## ✖ dplyr::lag()     masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflict
s to become errors
```

```
library(lubridate)
```

```
library(skimr) # Summarize data
library(readr) # Import data
library(janitor) # Cleaning
```

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

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

```
library(leaflet)
library(htmlwidgets)
library(highcharter)
```

```
## Registered S3 method overwritten by 'quantmod':
##   method             from
##   as.zoo.data.frame zoo
```

```
## Highcharts (www.highcharts.com) is a Highsoft software product which is
```

```
## not free for commercial and Governmental use
```

```
library(htmltools)
library(leaflet.extras)
library(geosphere)
```

```
## The legacy packages maptools, rgdal, and rgeos, underpinning the sp package,
## which was just loaded, will retire in October 2023.
## Please refer to R-spatial evolution reports for details, especially
## https://r-spatial.org/r/2023/05/15/evolution4.html.
## It may be desirable to make the sf package available;
## package maintainers should consider adding sf to Suggests:.
## The sp package is now running under evolution status 2
##   (status 2 uses the sf package in place of rgdal)
```

```
##  
## Attaching package: 'geosphere'
```

```
## The following object is masked from 'package:htmltools':  
##  
##      span
```

```
library(leafsync)  
library(mapview)  
library(tinytex)
```

## Checking Work Directory

```
getwd()
```

```
## [1] "/Users/rachel_sun/Desktop/divvy"
```

## Importing data

```
divvy_202004 <- read.csv("202004-divvy-tripdata.csv")
```

```
divvy_202005 <- read.csv("202005-divvy-tripdata.csv")
```

```
divvy_202006 <- read.csv("202006-divvy-tripdata.csv")
```

```
divvy_202007 <- read.csv("202007-divvy-tripdata.csv")
```

```
divvy_202008 <- read.csv("202008-divvy-tripdata.csv")
```

```
divvy_202009 <- read.csv("202009-divvy-tripdata.csv")
```

```
divvy_202010 <- read.csv("202010-divvy-tripdata.csv")
```

```
divvy_202011 <- read.csv("202011-divvy-tripdata.csv")
```

```
divvy_202012 <- read.csv("202012-divvy-tripdata.csv")
```

```
divvy_202101 <- read.csv("202101-divvy-tripdata.csv")
```

```
divvy_202102 <- read.csv("202102-divvy-tripdata.csv")
```

```
divvy_202103 <- read.csv("202103-divvy-tripdata.csv")
```

```
divvy_202104 <- read.csv("202104-divvy-tripdata.csv")
```

```
divvy_202105 <- read.csv("202105-divvy-tripdata.csv")
```

```
divvy_202106 <- read.csv("202106-divvy-tripdata.csv")
```

```
divvy_202107 <- read.csv("202107-divvy-tripdata.csv")
```

```
divvy_202108 <- read.csv("202108-divvy-tripdata.csv")
```

```
divvy_202109 <- read.csv("202109-divvy-tripdata.csv")
```

## Checking Data Structure

After importing the 12 files, I compared the naming conventions of the columns to check compatibility for a full union.

```
colnames(divvy_202004)
```

```
## [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"
```

```
colnames(divvy_202005)
```

```
## [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"
```

```
colnames(divvy_202006)
```

```
## [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"
```

```
colnames(divvy_202007)
```

```
## [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"
```

```
colnames(divvy_202008)
```

```
## [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"
```

```
colnames(divvy_202009)
```

```
## [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"
```

```
colnames(divvy_202010)
```

```
## [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"
```

```
colnames(divvy_202011)
```

```
## [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"
```

```
colnames(divvy_202012)
```

```
## [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"
```

```
colnames(divvy_202101)
```

```
## [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"
```

```
colnames(divvy_202102)
```

```
## [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"
```

```
colnames(divvy_202103)
```

```
## [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"
```

```
colnames(divvy_202104)
```

```
## [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"
```

```
colnames(divvy_202105)
```

```
## [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"
```

```
colnames(divvy_202106)
```

```
## [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"
```

```
colnames(divvy_202107)
```

```
## [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"
```

```
colnames(divvy_202108)
```

```
## [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"
```

```
colnames(divvy_202109)
```

```
## [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"
```

## Checking Inconsistencies

Given that the column names are identical, I will use the `compare_df_cols()` function to probe deeper for any inconsistencies in data formatting. I have requested R to return any data type mismatches between the tables.

```
compare_df_cols(divvy_202004, divvy_202005, divvy_202006, divvy_202007, divvy_202008, divvy_202009, divvy_202010, divvy_202011, divvy_202012, divvy_202101, divvy_202102, divvy_202103, divvy_202104, divvy_202105, divvy_202106, divvy_202107, divvy_202108, divvy_202109, return = "mismatch")
```

```
##      column_name divvy_202004 divvy_202005 divvy_202006 divvy_202007
## 1  end_station_id      integer      integer      integer      integer
## 2 start_station_id      integer      integer      integer      integer
##      divvy_202008 divvy_202009 divvy_202010 divvy_202011 divvy_202012 divvy_202101
## 1      integer      integer      integer      integer      character      character
## 2      integer      integer      integer      integer      character      character
##      divvy_202102 divvy_202103 divvy_202104 divvy_202105 divvy_202106 divvy_202107
## 1      character      character      character      character      character      character
## 2      character      character      character      character      character      character
##      divvy_202108 divvy_202109
## 1      character      character
## 2      character      character
```

## Correcting Mismatch Columns

There are mismatched data types regard to `end_station_id` and `start_station_id`. From April 2020 to October 2020, the end and start station id data types are integer. The left months are character. We need transfer these two columns into character type for 202004 - 202411. I will use `as.character()` to cope with that.



```
divvy_202004$start_station_id <- as.character(divvy_202004$start_station_id)
divvy_202004$end_station_id <- as.character(divvy_202004$end_station_id)
```

```
divvy_202005$start_station_id <- as.character(divvy_202005$start_station_id)
divvy_202005$end_station_id <- as.character(divvy_202005$end_station_id)
```

```
divvy_202006$start_station_id <- as.character(divvy_202006$start_station_id)
divvy_202006$end_station_id <- as.character(divvy_202006$end_station_id)
```

```
divvy_202007$start_station_id <- as.character(divvy_202007$start_station_id)
divvy_202007$end_station_id <- as.character(divvy_202007$end_station_id)
```

```
divvy_202008$start_station_id <- as.character(divvy_202008$start_station_id)
divvy_202008$end_station_id <- as.character(divvy_202008$end_station_id)
```

```
divvy_202009$start_station_id <- as.character(divvy_202009$start_station_id)
divvy_202009$end_station_id <- as.character(divvy_202009$end_station_id)
```

```
divvy_202010$start_station_id <- as.character(divvy_202010$start_station_id)
divvy_202010$end_station_id <- as.character(divvy_202010$end_station_id)
```

```
divvy_202011$start_station_id <- as.character(divvy_202011$start_station_id)
divvy_202011$end_station_id <- as.character(divvy_202011$end_station_id)
```

## Checking Inconsistencies Again

I will use the `compare_df_cols()` function again to check if there is any inconsistency.

```
compare_df_cols(divvy_202004, divvy_202005, divvy_202006, divvy_202007, divvy_202008, divvy_202009,
divvy_202010, divvy_202011, divvy_202012, divvy_202101, divvy_202102, divvy_202103,
return = "mismatch")
```

```
## [1] column_name divvy_202004 divvy_202005 divvy_202006 divvy_202007
## [6] divvy_202008 divvy_202009 divvy_202010 divvy_202011 divvy_202012
## [11] divvy_202101 divvy_202102 divvy_202103
## <0 rows> (or 0-length row.names)
```

## Combining Data

Since there are no mismatched data types between the tables now, combining the 12 tables into a single one will be relatively straightforward using the `bind_rows()` function. The column names offer sufficient context for the data, hence, they will require minimal renaming for this analysis. Vertical merge data with `rbind()` as all of dataset have the same columns

```
trip_data <- rbind(divvy_202010, divvy_202011, divvy_202012, divvy_202101, divvy_202102,
divvy_202103, divvy_202104, divvy_202105, divvy_202106, divvy_202107, divvy_202108,
divvy_202109)
```

# Exploring Data

## preview data

```
#View(trip_data)
```

## Checking Data Structure

```
str(trip_data)
```

```
## 'data.frame':    5136261 obs. of  13 variables:
##  $ ride_id          : chr  "ACB6B40CF5B9044C" "DF450C72FD109C01" "B6396B54A15AC0DF"
##  "44A4AEE261B9E854" ...
##  $ rideable_type     : chr  "electric_bike" "electric_bike" "electric_bike" "electri
##  c_bike" ...
##  $ started_at        : chr  "2020-10-31 19:39:43" "2020-10-31 23:50:08" "2020-10-31
##  23:00:01" "2020-10-31 22:16:43" ...
##  $ ended_at          : chr  "2020-10-31 19:57:12" "2020-11-01 00:04:16" "2020-10-31
##  23:08:22" "2020-10-31 22:19:35" ...
##  $ start_station_name: chr  "Lakeview Ave & Fullerton Pkwy" "Southport Ave & Wavelan
##  d Ave" "Stony Island Ave & 67th St" "Clark St & Grace St" ...
##  $ start_station_id  : chr  "313" "227" "102" "165" ...
##  $ end_station_name  : chr  "Rush St & Hubbard St" "Kedzie Ave & Milwaukee Ave" "Uni
##  versity Ave & 57th St" "Broadway & Sheridan Rd" ...
##  $ end_station_id    : chr  "125" "260" "423" "256" ...
##  $ start_lat         : num  41.9 41.9 41.8 42 41.9 ...
##  $ start_lng         : num  -87.6 -87.7 -87.6 -87.7 -87.7 ...
##  $ end_lat           : num  41.9 41.9 41.8 42 41.9 ...
##  $ end_lng           : num  -87.6 -87.7 -87.6 -87.7 -87.7 ...
##  $ member_casual     : chr  "casual" "casual" "casual" "casual" ...
```

## Summary of Data Combining and Exploration

I merged twelve CSV files containing historical trip data from October 2020 to September 2021. Upon examining the data, we checked its structure, completeness, whitespace, and the number of unique values for categorical variables, along with preliminary summary statistics. This examination highlighted numerous incomplete fields. Notably, the station id, latitude, longitude variables were frequently missing.

## Data Cleaning

### Rename Columns

I use rename() function to change the name of rideable\_type to bike\_type, member\_casual to customer\_type.

```
trip_data <- trip_data %>%
  rename(bike_type = rideable_type,
         customer_type = member_casual)
```

## Verifying Column Names

```
colnames(trip_data)
```

```
## [1] "ride_id"          "bike_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] "customer_type"
```

## Cleaning Names

clean names; include only characters, numbers, and underscores in names.

```
trip_data <- clean_names(trip_data)
```

## Removing Empty Rows and Columns

```
trip_data <- remove_empty(trip_data)
```

```
## value for "which" not specified, defaulting to c("rows", "cols")
```

## Removing Duplicates

```
trip_data <- distinct(trip_data)
```

## Checking Anomalies

check for naming anomalies, the variables should only include limited types.

```
count(trip_data, bike_type)
```

```
##      bike_type      n
## 1 classic_bike 2750831
## 2  docked_bike  677980
## 3 electric_bike 1707450
```

```
count(trip_data, customer_type)
```

```
## customer_type      n
## 1      casual 2358287
## 2      member 2777974
```

## Summarizing Data Set

```
summary(trip_data)
```

```
##      ride_id          bike_type      started_at      ended_at
## Length:5136261      Length:5136261      Length:5136261      Length:5136261
## Class :character      Class :character      Class :character      Class :character
## Mode  :character      Mode  :character      Mode  :character      Mode  :character
##
##
##
##
## start_station_name start_station_id  end_station_name  end_station_id
## Length:5136261      Length:5136261      Length:5136261      Length:5136261
## Class :character      Class :character      Class :character      Class :character
## Mode  :character      Mode  :character      Mode  :character      Mode  :character
##
##
##
##      start_lat      start_lng      end_lat      end_lng
## Min.   :41.64      Min.   : -87.84      Min.   :41.51      Min.   : -88.07
## 1st Qu.:41.88      1st Qu.: -87.66      1st Qu.:41.88      1st Qu.: -87.66
## Median :41.90      Median : -87.64      Median :41.90      Median : -87.64
## Mean   :41.90      Mean   : -87.65      Mean   :41.90      Mean   : -87.65
## 3rd Qu.:41.93      3rd Qu.: -87.63      3rd Qu.:41.93      3rd Qu.: -87.63
## Max.   :42.08      Max.   : -87.52      Max.   :42.17      Max.   : -87.44
##
##                      NA's      :4821      NA's      :4821
## customer_type
## Length:5136261
## Class :character
## Mode  :character
##
##
##
##
```

## Checking Dataset Dimensions

```
dim(trip_data)
```

```
## [1] 5136261      13
```

## Checking Data Samples

```
head(trip_data)
```

```
##          ride_id      bike_type      started_at      ended_at
## 1 ACB6B40CF5B9044C electric_bike 2020-10-31 19:39:43 2020-10-31 19:57:12
## 2 DF450C72FD109C01 electric_bike 2020-10-31 23:50:08 2020-11-01 00:04:16
## 3 B6396B54A15AC0DF electric_bike 2020-10-31 23:00:01 2020-10-31 23:08:22
## 4 44A4AEE261B9E854 electric_bike 2020-10-31 22:16:43 2020-10-31 22:19:35
## 5 10B7DD76A6A2EB95 electric_bike 2020-10-31 19:38:19 2020-10-31 19:54:32
## 6 DA6C3759660133DA electric_bike 2020-10-29 17:38:04 2020-10-29 17:45:43
##          start_station_name start_station_id      end_station_name
## 1 Lakeview Ave & Fullerton Pkwy          313      Rush St & Hubbard St
## 2 Southport Ave & Waveland Ave          227 Kedzie Ave & Milwaukee Ave
## 3 Stony Island Ave & 67th St          102 University Ave & 57th St
## 4 Clark St & Grace St          165 Broadway & Sheridan Rd
## 5 Southport Ave & Wrightwood Ave          190 Stave St & Armitage Ave
## 6 Larrabee St & Division St          359 Wells St & Huron St
## end_station_id start_lat start_lng end_lat end_lng customer_type
## 1          125  41.92610 -87.63898 41.89035 -87.62607      casual
## 2          260  41.94817 -87.66391 41.92953 -87.70782      casual
## 3          423  41.77346 -87.58537 41.79145 -87.60005      casual
## 4          256  41.95085 -87.65924 41.95281 -87.65010      casual
## 5          185  41.92886 -87.66396 41.91778 -87.69143      casual
## 6           53  41.90353 -87.64335 41.89440 -87.63431      casual
```

```
tail(trip_data)
```

```
##          ride_id      bike_type      started_at      ended_at
## 5136256 0A6AA3B1A1EC5FF4 classic_bike 2021-09-14 23:00:37 2021-09-14 23:10:55
## 5136257 FA66BCAB0D73DDC2 classic_bike 2021-09-22 15:46:57 2021-09-22 16:01:15
## 5136258 1D44DEFB5D36CA04 classic_bike 2021-09-25 16:25:23 2021-09-25 16:40:29
## 5136259 6A346EA57FC23C45 classic_bike 2021-09-25 16:26:05 2021-09-25 16:40:30
## 5136260 49360AFD771100A6 classic_bike 2021-09-15 17:57:48 2021-09-15 18:24:06
## 5136261 343190A2DC023FED electric_bike 2021-09-11 18:01:06 2021-09-11 18:08:26
##          start_station_name start_station_id      end_station_name
## 5136256 Ellis Ave & 60th St      KA1503000014      Shore Dr & 55th St
## 5136257 Ellis Ave & 83rd St          584 Stony Island Ave & 75th St
## 5136258 Ellis Ave & 60th St      KA1503000014      Shore Dr & 55th St
## 5136259 Ellis Ave & 60th St      KA1503000014      Shore Dr & 55th St
## 5136260 Ellis Ave & 60th St      KA1503000014      Shore Dr & 55th St
## 5136261 Wells St & Huron St      TA1306000012      Clinton St & Lake St
##          end_station_id start_lat start_lng end_lat end_lng customer_type
## 5136256 TA1308000009  41.78510 -87.60107 41.79521 -87.58071      member
## 5136257 KA1503000019  41.74412 -87.59903 41.75867 -87.58688      casual
## 5136258 TA1308000009  41.78510 -87.60107 41.79521 -87.58071      casual
## 5136259 TA1308000009  41.78510 -87.60107 41.79521 -87.58071      casual
## 5136260 TA1308000009  41.78510 -87.60107 41.79521 -87.58071      casual
## 5136261          13021  41.89489 -87.63434 41.88577 -87.64137      member
```

## Checking Data Types

```
glimpse(trip_data)
```

```
## Rows: 5,136,261
## Columns: 13
## $ ride_id          <chr> "ACB6B40CF5B9044C", "DF450C72FD109C01", "B6396B54A1...
## $ bike_type        <chr> "electric_bike", "electric_bike", "electric_bike", ...
## $ started_at       <chr> "2020-10-31 19:39:43", "2020-10-31 23:50:08", "2020...
## $ ended_at         <chr> "2020-10-31 19:57:12", "2020-11-01 00:04:16", "2020...
## $ start_station_name <chr> "Lakeview Ave & Fullerton Pkwy", "Southport Ave & W...
## $ start_station_id  <chr> "313", "227", "102", "165", "190", "359", "313", "1...
## $ end_station_name  <chr> "Rush St & Hubbard St", "Kedzie Ave & Milwaukee Ave...
## $ end_station_id    <chr> "125", "260", "423", "256", "185", "53", "125", "31...
## $ start_lat         <dbl> 41.92610, 41.94817, 41.77346, 41.95085, 41.92886, 4...
## $ start_lng         <dbl> -87.63898, -87.66391, -87.58537, -87.65924, -87.663...
## $ end_lat           <dbl> 41.89035, 41.92953, 41.79145, 41.95281, 41.91778, 4...
## $ end_lng           <dbl> -87.62607, -87.70782, -87.60005, -87.65010, -87.691...
## $ customer_type     <chr> "casual", "casual", "casual", "casual", "casual", "..."
```

## Checking Data Attributes

```
#attributes(trip_data)
```

## Checking Dataset Names Again

```
attr(x = trip_data, which = "names")
```

```
## [1] "ride_id"          "bike_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] "customer_type"
```

## Checking Data Structure

```
str(trip_data)
```

```
## 'data.frame':    5136261 obs. of  13 variables:
## $ ride_id      : chr  "ACB6B40CF5B9044C" "DF450C72FD109C01" "B6396B54A15AC0DF"
"44A4AEE261B9E854" ...
## $ bike_type    : chr  "electric_bike" "electric_bike" "electric_bike" "electri
c_bike" ...
## $ started_at   : chr  "2020-10-31 19:39:43" "2020-10-31 23:50:08" "2020-10-31
23:00:01" "2020-10-31 22:16:43" ...
## $ ended_at     : chr  "2020-10-31 19:57:12" "2020-11-01 00:04:16" "2020-10-31
23:08:22" "2020-10-31 22:19:35" ...
## $ start_station_name: chr  "Lakeview Ave & Fullerton Pkwy" "Southport Ave & Wavelan
d Ave" "Stony Island Ave & 67th St" "Clark St & Grace St" ...
## $ start_station_id : chr  "313" "227" "102" "165" ...
## $ end_station_name : chr  "Rush St & Hubbard St" "Kedzie Ave & Milwaukee Ave" "Uni
versity Ave & 57th St" "Broadway & Sheridan Rd" ...
## $ end_station_id   : chr  "125" "260" "423" "256" ...
## $ start_lat        : num  41.9 41.9 41.8 42 41.9 ...
## $ start_lng        : num  -87.6 -87.7 -87.6 -87.7 -87.7 ...
## $ end_lat          : num  41.9 41.9 41.8 42 41.9 ...
## $ end_lng          : num  -87.6 -87.7 -87.6 -87.7 -87.7 ...
## $ customer_type    : chr  "casual" "casual" "casual" "casual" ...
```

## Full Summary of the Dataset

```
skim(trip_data)
```

### Data summary

Name	trip_data
Number of rows	5136261
Number of columns	13
Column type frequency:	
character	9
numeric	4
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	5136052	0
bike_type	0	1.00	11	13	0	3	0
started_at	0	1.00	19	19	0	4301706	0
ended_at	0	1.00	19	19	0	4291553	0
start_station_name	0	1.00	0	53	523467	785	0

skim_variable	n_missing	complete_rate	min	max	empty	n_unique	whitespace
start_station_id	55839	0.99	0	36	467942	1300	0
end_station_name	0	1.00	0	53	567268	782	0
end_station_id	62613	0.99	0	36	504888	1300	0
customer_type	0	1.00	6	6	0	2	0

Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
start_lat	0	1	41.90	0.04	41.64	41.88	41.90	41.93	42.08	
start_lng	0	1	-87.65	0.03	-87.84	-87.66	-87.64	-87.63	-87.52	
end_lat	4821	1	41.90	0.04	41.51	41.88	41.90	41.93	42.17	
end_lng	4821	1	-87.65	0.03	-88.07	-87.66	-87.64	-87.63	-87.44	

As we can see above, there are some missing data in the column of start\_station\_id, end\_station\_id, end\_lat, and end\_lng. And, in the column of start station name and end station name, there are some empty values. I wonder if it is possible to create unique tables through station names, and then use the unique tables as a reference to fill the missing data. Let's try it.

## Creating a Unique Start Stations Table

```
stations <- distinct(trip_data, start_station_name, start_station_id)
```

```
summary(stations)
```

```
## start_station_name start_station_id
## Length:1372      Length:1372
## Class :character  Class :character
## Mode  :character  Mode  :character
```

```
#View(stations)
```

## Summary of Unique Stations Table

```
skim(stations)
```

Data summary

Name	stations
Number of rows	1372
Number of columns	2

Column type frequency:



character	2
<hr/>	
Group variables	None

**Variable type: character**

skim_variable	n_missing	complete_rate	min	max	empty	n_unique	whitespace
start_station_name	0	1	0	53	5	785	0
start_station_id	4	1	0	36	1	1300	0

Here we notice that there are four missing and one empty in start\_station\_id. It means that there are five station names without their own id. And there are also 5 empty values in the start\_station\_name. It seems that it is impossible to fill the missing data by using the distinct station name.

## Creating the Unique End Stations Table

```
end_station <- distinct(trip_data, end_station_name, end_station_id)
```

```
skim(end_station)
```

Data summary

Name	end_station
Number of rows	1364
Number of columns	2
<hr/>	
Column type frequency:	
character	2
<hr/>	
Group variables	None

**Variable type: character**

skim_variable	n_missing	complete_rate	min	max	empty	n_unique	whitespace
end_station_name	0	1	0	53	2	782	0
end_station_id	3	1	0	36	1	1300	0

Here we notice that there are 3 missing and 1 empty in end\_station\_id. It means that there 4 end station name without their own id. There are also 2 empty values in the end\_station\_name. It is impossible to fill the missing data by using the distinct station name.

# Creating a Table to Further Check NA Values

```
start_id_null <- filter(trip_data, is.na(start_station_id))
```

```
summary(start_id_null)
```

```
##      ride_id      bike_type      started_at      ended_at
## Length:55839 Length:55839 Length:55839 Length:55839
## Class :character Class :character Class :character Class :character
## Mode :character Mode :character Mode :character Mode :character
##
##
##
## start_station_name start_station_id end_station_name end_station_id
## Length:55839 Length:55839 Length:55839 Length:55839
## Class :character Class :character Class :character Class :character
## Mode :character Mode :character Mode :character Mode :character
##
##
##
## start_lat start_lng end_lat end_lng
## Min. :41.64 Min. : -87.80 Min. :41.54 Min. : -87.87
## 1st Qu.:41.80 1st Qu.: -87.69 1st Qu.:41.81 1st Qu.: -87.69
## Median :41.90 Median : -87.65 Median :41.90 Median : -87.65
## Mean :41.88 Mean : -87.65 Mean :41.88 Mean : -87.65
## 3rd Qu.:41.93 3rd Qu.: -87.62 3rd Qu.:41.93 3rd Qu.: -87.62
## Max. :42.08 Max. : -87.52 Max. :42.15 Max. : -87.52
## customer_type
## Length:55839
## Class :character
## Mode :character
##
##
##
```

```
#View(start_id_null)
```

```
trip_data %>%
  select(start_station_name, start_station_id) %>%
  filter(is.na(start_station_id) | start_station_id == "") %>%
  filter(!start_station_name == "")
```

##	start_station_name	start_station_id
## 1	W Oakdale Ave & N Broadway	<NA>
## 2	W Oakdale Ave & N Broadway	<NA>
## 3	W Oakdale Ave & N Broadway	<NA>
## 4	W Oakdale Ave & N Broadway	<NA>
## 5	W Oakdale Ave & N Broadway	<NA>
## 6	W Oakdale Ave & N Broadway	<NA>
## 7	W Oakdale Ave & N Broadway	<NA>
## 8	W Oakdale Ave & N Broadway	<NA>
## 9	W Oakdale Ave & N Broadway	<NA>
## 10	W Oakdale Ave & N Broadway	<NA>
## 11	W Oakdale Ave & N Broadway	<NA>
## 12	W Armitage Ave & N Sheffield Ave	<NA>
## 13	W Oakdale Ave & N Broadway	<NA>
## 14	W Oakdale Ave & N Broadway	<NA>
## 15	W Oakdale Ave & N Broadway	<NA>
## 16	W Armitage Ave & N Sheffield Ave	<NA>
## 17	W Armitage Ave & N Sheffield Ave	<NA>
## 18	W Oakdale Ave & N Broadway	<NA>
## 19	W Oakdale Ave & N Broadway	<NA>
## 20	W Oakdale Ave & N Broadway	<NA>
## 21	W Oakdale Ave & N Broadway	<NA>
## 22	W Armitage Ave & N Sheffield Ave	<NA>
## 23	W Oakdale Ave & N Broadway	<NA>
## 24	W Oakdale Ave & N Broadway	<NA>
## 25	W Oakdale Ave & N Broadway	<NA>
## 26	W Oakdale Ave & N Broadway	<NA>
## 27	W Armitage Ave & N Sheffield Ave	<NA>
## 28	W Oakdale Ave & N Broadway	<NA>
## 29	W Armitage Ave & N Sheffield Ave	<NA>
## 30	W Oakdale Ave & N Broadway	<NA>
## 31	W Oakdale Ave & N Broadway	<NA>
## 32	W Oakdale Ave & N Broadway	<NA>
## 33	W Oakdale Ave & N Broadway	<NA>
## 34	W Armitage Ave & N Sheffield Ave	<NA>
## 35	W Armitage Ave & N Sheffield Ave	<NA>
## 36	W Oakdale Ave & N Broadway	<NA>
## 37	W Oakdale Ave & N Broadway	<NA>
## 38	W Oakdale Ave & N Broadway	<NA>
## 39	W Oakdale Ave & N Broadway	<NA>
## 40	W Oakdale Ave & N Broadway	<NA>
## 41	W Oakdale Ave & N Broadway	<NA>
## 42	W Armitage Ave & N Sheffield Ave	<NA>
## 43	W Oakdale Ave & N Broadway	<NA>
## 44	W Oakdale Ave & N Broadway	<NA>
## 45	W Oakdale Ave & N Broadway	<NA>
## 46	W Oakdale Ave & N Broadway	<NA>
## 47	W Oakdale Ave & N Broadway	<NA>
## 48	W Armitage Ave & N Sheffield Ave	<NA>
## 49	W Oakdale Ave & N Broadway	<NA>
## 50	W Oakdale Ave & N Broadway	<NA>
## 51	W Oakdale Ave & N Broadway	<NA>
## 52	W Oakdale Ave & N Broadway	<NA>
## 53	W Oakdale Ave & N Broadway	<NA>
## 54	W Oakdale Ave & N Broadway	<NA>
## 55	W Oakdale Ave & N Broadway	<NA>

[illegible]

[illegible]

[illegible]

[illegible]

## 284	W Oakdale Ave & N Broadway	<NA>
## 285	W Oakdale Ave & N Broadway	<NA>
## 286	W Oakdale Ave & N Broadway	<NA>
## 287	W Oakdale Ave & N Broadway	<NA>
## 288	W Oakdale Ave & N Broadway	<NA>
## 289	W Oakdale Ave & N Broadway	<NA>
## 290	W Oakdale Ave & N Broadway	<NA>
## 291	W Oakdale Ave & N Broadway	<NA>
## 292	W Oakdale Ave & N Broadway	<NA>
## 293	W Oakdale Ave & N Broadway	<NA>
## 294	W Oakdale Ave & N Broadway	<NA>
## 295	W Oakdale Ave & N Broadway	<NA>
## 296	W Oakdale Ave & N Broadway	<NA>
## 297	W Oakdale Ave & N Broadway	<NA>
## 298	W Oakdale Ave & N Broadway	<NA>
## 299	W Oakdale Ave & N Broadway	<NA>
## 300	W Oakdale Ave & N Broadway	<NA>
## 301	W Oakdale Ave & N Broadway	<NA>
## 302	W Oakdale Ave & N Broadway	<NA>
## 303	W Oakdale Ave & N Broadway	<NA>
## 304	W Oakdale Ave & N Broadway	<NA>
## 305	W Oakdale Ave & N Broadway	<NA>
## 306	W Oakdale Ave & N Broadway	<NA>
## 307	W Oakdale Ave & N Broadway	<NA>
## 308	W Oakdale Ave & N Broadway	<NA>
## 309	W Oakdale Ave & N Broadway	<NA>
## 310	W Oakdale Ave & N Broadway	<NA>
## 311	W Oakdale Ave & N Broadway	<NA>
## 312	W Oakdale Ave & N Broadway	<NA>
## 313	W Oakdale Ave & N Broadway	<NA>
## 314	W Oakdale Ave & N Broadway	<NA>
## 315	W Oakdale Ave & N Broadway	<NA>
## 316	W Oakdale Ave & N Broadway	<NA>
## 317	W Oakdale Ave & N Broadway	<NA>

We can see that, when the id is NA, the station name is usually empty. There are only 317 rows where the station\_id is NA or empty while the station\_name is not empty.

```
str(start_id_null)
```



```
## 'data.frame':    55839 obs. of  13 variables:
## $ ride_id      : chr  "285D224410C101C5" "774087F3F7887F0C" "5260DF288DA43EF9"
##               : chr  "1C4BF0BF80B84E90" ...
## $ bike_type    : chr  "electric_bike" "electric_bike" "electric_bike" "electric_bike" ...
## $ started_at   : chr  "2020-10-28 23:12:03" "2020-10-25 22:55:10" "2020-10-13 14:38:50"
##               : chr  "2020-10-13 07:51:13" ...
## $ ended_at     : chr  "2020-10-28 23:24:32" "2020-10-25 23:14:26" "2020-10-13 14:39:35"
##               : chr  "2020-10-13 08:27:09" ...
## $ start_station_name: chr  "" "" "" "" ...
## $ start_station_id : chr  NA NA NA NA ...
## $ end_station_name : chr  "Wabash Ave & Grand Ave" "Damen Ave & Pierce Ave" "" "Daley Center Plaza" ...
## $ end_station_id   : chr  "199" "69" NA "81" ...
## $ start_lat        : num  41.9 41.9 41.9 42 41.7 ...
## $ start_lng        : num  -87.7 -87.7 -87.7 -87.7 -87.7 ...
## $ end_lat          : num  41.9 41.9 41.9 41.9 41.7 ...
## $ end_lng          : num  -87.6 -87.7 -87.7 -87.6 -87.6 ...
## $ customer_type    : chr  "casual" "casual" "casual" "casual" ...
```

```
unique(start_id_null$bike_type)
```

```
## [1] "electric_bike"
```

It is only the electric bikes has NA or empty start\_station\_id values.

```
end_id_null <- trip_data %>%
  filter(is.na(end_station_id))
```

```
unique(end_id_null$bike_type)
```

```
## [1] "electric_bike" "docked_bike"
```

However, there are electric bikes and docked bikes have NA or empty start\_station\_id values. It is impossible to determine what causes the NA or empty station name and id values.

```
trip_data %>%
  select(end_station_name, end_station_id) %>%
  filter(is.na(end_station_id) | end_station_id == "") %>%
  filter(!end_station_name == "")
```

##		end_station_name	end_station_id
## 1		W Oakdale Ave & N Broadway	<NA>
## 2		W Oakdale Ave & N Broadway	<NA>
## 3	W Armitage Ave & N Sheffield Ave		<NA>
## 4		W Oakdale Ave & N Broadway	<NA>
## 5		W Oakdale Ave & N Broadway	<NA>
## 6		W Oakdale Ave & N Broadway	<NA>
## 7		W Oakdale Ave & N Broadway	<NA>
## 8		W Oakdale Ave & N Broadway	<NA>
## 9		W Oakdale Ave & N Broadway	<NA>
## 10		W Oakdale Ave & N Broadway	<NA>
## 11		W Oakdale Ave & N Broadway	<NA>
## 12		W Oakdale Ave & N Broadway	<NA>
## 13		W Oakdale Ave & N Broadway	<NA>
## 14		W Oakdale Ave & N Broadway	<NA>
## 15		W Oakdale Ave & N Broadway	<NA>
## 16		W Oakdale Ave & N Broadway	<NA>
## 17	W Armitage Ave & N Sheffield Ave		<NA>
## 18		W Oakdale Ave & N Broadway	<NA>
## 19		W Oakdale Ave & N Broadway	<NA>
## 20	W Armitage Ave & N Sheffield Ave		<NA>
## 21		W Oakdale Ave & N Broadway	<NA>
## 22		W Oakdale Ave & N Broadway	<NA>
## 23		W Oakdale Ave & N Broadway	<NA>
## 24		W Oakdale Ave & N Broadway	<NA>
## 25		W Oakdale Ave & N Broadway	<NA>
## 26	W Armitage Ave & N Sheffield Ave		<NA>
## 27	W Armitage Ave & N Sheffield Ave		<NA>
## 28		W Oakdale Ave & N Broadway	<NA>
## 29		W Oakdale Ave & N Broadway	<NA>
## 30		W Oakdale Ave & N Broadway	<NA>
## 31		W Oakdale Ave & N Broadway	<NA>
## 32		W Oakdale Ave & N Broadway	<NA>
## 33	W Armitage Ave & N Sheffield Ave		<NA>
## 34		W Oakdale Ave & N Broadway	<NA>
## 35		W Oakdale Ave & N Broadway	<NA>
## 36		W Oakdale Ave & N Broadway	<NA>
## 37	W Armitage Ave & N Sheffield Ave		<NA>
## 38		W Oakdale Ave & N Broadway	<NA>
## 39		W Oakdale Ave & N Broadway	<NA>
## 40		W Oakdale Ave & N Broadway	<NA>
## 41	W Armitage Ave & N Sheffield Ave		<NA>
## 42		W Oakdale Ave & N Broadway	<NA>
## 43		W Oakdale Ave & N Broadway	<NA>
## 44		W Oakdale Ave & N Broadway	<NA>
## 45		W Oakdale Ave & N Broadway	<NA>
## 46	W Armitage Ave & N Sheffield Ave		<NA>
## 47	W Armitage Ave & N Sheffield Ave		<NA>
## 48		W Oakdale Ave & N Broadway	<NA>
## 49		W Oakdale Ave & N Broadway	<NA>
## 50		W Oakdale Ave & N Broadway	<NA>
## 51	W Armitage Ave & N Sheffield Ave		<NA>
## 52		W Oakdale Ave & N Broadway	<NA>
## 53		W Oakdale Ave & N Broadway	<NA>
## 54		W Oakdale Ave & N Broadway	<NA>
## 55		W Oakdale Ave & N Broadway	<NA>

[illegible]

[illegible]

[illegible]

```
## 227      W Oakdale Ave & N Broadway      <NA>
## 228      W Oakdale Ave & N Broadway      <NA>
## 229      W Oakdale Ave & N Broadway      <NA>
## 230      W Oakdale Ave & N Broadway      <NA>
## 231      W Oakdale Ave & N Broadway      <NA>
## 232      W Oakdale Ave & N Broadway      <NA>
## 233      W Oakdale Ave & N Broadway      <NA>
```

There are only 233 rows where the end\_station\_id is NA or empty and the station\_name is not empty. I tend to remove all of the values where the start\_station\_name and end\_station\_name is empty as I will use these two columns to analyze the top stations.

```
unique(end_id_null$customer_type)
```

```
## [1] "casual" "member"
```

```
end_lat_null <- trip_data %>%
  filter(is.na(end_lat))
```

```
unique(end_lat_null$bike_type)
```

```
## [1] "docked_bike" "classic_bike"
```

As shown above, I apply filters to explore the dataset and examine the null rows and corresponding columns. I notice that there is only electric bike with missing start\_station\_id. For the missing end\_station\_id, end\_lat, and end\_lng, it includes docked bike and electric bike. As I couldn't access the stakeholders, it is impossible to confirm with them what causes the missing values in these columns. As I am going to use the geographical points to map stations trends, I will remove those rows with missing values in end\_lat and end\_lng.

```
bike_rides <- trip_data %>%
  filter_at(vars(end_lat, end_lng), all_vars(!is.na(.))) %>%
  filter(!start_station_name == "" & !end_station_name == "")
```

## Full summary of combined table

```
skim(bike_rides)
```

### Data summary

Name	bike_rides
Number of rows	4354488
Number of columns	13
Column type frequency:	
character	9
numeric	4

Group variables

None

### Variable type: character

skim_variable	n_missing	complete_rate	min	max	empty	n_unique	whitespace
ride_id	0	1	16	16	0	4354280	0
bike_type	0	1	11	13	0	3	0
started_at	0	1	19	19	0	3732831	0
ended_at	0	1	19	19	0	3721211	0
start_station_name	0	1	3	53	0	781	0
start_station_id	259	1	1	36	0	1296	0
end_station_name	0	1	10	53	0	776	0
end_station_id	194	1	1	36	0	1294	0
customer_type	0	1	6	6	0	2	0

### Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
start_lat	0	1	41.90	0.04	41.65	41.88	41.90	41.93	42.06	
start_lng	0	1	-87.64	0.02	-87.83	-87.66	-87.64	-87.63	-87.53	
end_lat	0	1	41.90	0.04	41.65	41.88	41.90	41.93	42.17	
end_lng	0	1	-87.64	0.02	-87.83	-87.66	-87.64	-87.63	-87.52	

```
summary(bike_rides)
```

```
##      ride_id          bike_type      started_at      ended_at
## Length:4354488      Length:4354488      Length:4354488      Length:4354488
## Class :character    Class :character    Class :character    Class :character
## Mode  :character    Mode  :character    Mode  :character    Mode  :character
##
##
##
## start_station_name start_station_id  end_station_name  end_station_id
## Length:4354488      Length:4354488      Length:4354488      Length:4354488
## Class :character    Class :character    Class :character    Class :character
## Mode  :character    Mode  :character    Mode  :character    Mode  :character
##
##
##
##      start_lat      start_lng      end_lat      end_lng
## Min.      :41.65    Min.      :-87.83    Min.      :41.65    Min.      :-87.83
## 1st Qu.:41.88      1st Qu.: -87.66    1st Qu.:41.88      1st Qu.: -87.66
## Median :41.90      Median : -87.64    Median :41.90      Median : -87.64
## Mean   :41.90      Mean   : -87.64    Mean   :41.90      Mean   : -87.64
## 3rd Qu.:41.93      3rd Qu.: -87.63    3rd Qu.:41.93      3rd Qu.: -87.63
## Max.    :42.06      Max.    : -87.53    Max.    :42.17      Max.    : -87.52
## customer_type
## Length:4354488
## Class :character
## Mode  :character
##
##
##
```

Right now, there is no missing values in the columns of start\_station\_name, end\_station\_name, end\_lat and end\_lng. I will not remove the null values in the start\_station\_id and end\_station\_id as I would not use these two columns in the following analysis.

Next, I will add some calculation columns to the bike\_rides data set, like year, month, day, hour and ride duration to build some granularity for my analysis.

First, I will format the date type of started\_at and ended\_at into POSIXct type.

## Converting to Date-time Format

```
bike_rides$started_at <- as.POSIXct(bike_rides$started_at)
```

## Format ended\_at into POSIXct

```
bike_rides$ended_at <- as.POSIXct(bike_rides$ended_at)
```

## Check data samples

```
head(bike_rides)
```



```
##          ride_id      bike_type      started_at      ended_at
## 1 ACB6B40CF5B9044C electric_bike 2020-10-31 19:39:43 2020-10-31 19:57:12
## 2 DF450C72FD109C01 electric_bike 2020-10-31 23:50:08 2020-11-01 00:04:16
## 3 B6396B54A15AC0DF electric_bike 2020-10-31 23:00:01 2020-10-31 23:08:22
## 4 44A4AEE261B9E854 electric_bike 2020-10-31 22:16:43 2020-10-31 22:19:35
## 5 10B7DD76A6A2EB95 electric_bike 2020-10-31 19:38:19 2020-10-31 19:54:32
## 6 DA6C3759660133DA electric_bike 2020-10-29 17:38:04 2020-10-29 17:45:43
##          start_station_name start_station_id      end_station_name
## 1 Lakeview Ave & Fullerton Pkwy          313      Rush St & Hubbard St
## 2 Southport Ave & Waveland Ave          227 Kedzie Ave & Milwaukee Ave
## 3 Stony Island Ave & 67th St          102 University Ave & 57th St
## 4 Clark St & Grace St          165 Broadway & Sheridan Rd
## 5 Southport Ave & Wrightwood Ave          190 Stave St & Armitage Ave
## 6 Larrabee St & Division St          359 Wells St & Huron St
## end_station_id start_lat start_lng end_lat end_lng customer_type
## 1          125  41.92610 -87.63898 41.89035 -87.62607      casual
## 2          260  41.94817 -87.66391 41.92953 -87.70782      casual
## 3          423  41.77346 -87.58537 41.79145 -87.60005      casual
## 4          256  41.95085 -87.65924 41.95281 -87.65010      casual
## 5          185  41.92886 -87.66396 41.91778 -87.69143      casual
## 6           53  41.90353 -87.64335 41.89440 -87.63431      casual
```

## View data structure.

started\_at and ended\_at have been tranfered into POSIXct type.

```
str(bike_rides)
```

```
## 'data.frame':    4354488 obs. of  13 variables:
## $ ride_id      : chr  "ACB6B40CF5B9044C" "DF450C72FD109C01" "B6396B54A15AC0DF"
## $ bike_type    : chr  "electric_bike" "electric_bike" "electric_bike" "electric_bike" ...
## $ started_at   : POSIXct, format: "2020-10-31 19:39:43" "2020-10-31 23:50:08"
## $ ended_at     : POSIXct, format: "2020-10-31 19:57:12" "2020-11-01 00:04:16"
## $ start_station_name: chr  "Lakeview Ave & Fullerton Pkwy" "Southport Ave & Waveland Ave"
## $ start_station_id : chr  "313" "227" "102" "165" ...
## $ end_station_name : chr  "Rush St & Hubbard St" "Kedzie Ave & Milwaukee Ave" "University Ave & 57th St"
## $ end_station_id   : chr  "125" "260" "423" "256" ...
## $ start_lat        : num  41.9 41.9 41.8 42 41.9 ...
## $ start_lng        : num  -87.6 -87.7 -87.6 -87.7 -87.7 ...
## $ end_lat          : num  41.9 41.9 41.8 42 41.9 ...
## $ end_lng          : num  -87.6 -87.7 -87.6 -87.7 -87.7 ...
## $ customer_type    : chr  "casual" "casual" "casual" "casual" ...
```

## Summary of Data Exploration and Cleaning:

- Reformatted variable names to comprise only characters, numbers, and underscores.
- Deleted empty rows and columns.
- Eliminated duplicate entries.

- Ensured observations had consistent naming and included only valid entries for categorical variables or those with restricted responses.
- Standardized entries using a consistent date-time format.
- Removed rows with blank or “NA” fields.

## Phase 3: Data Manipulation & Preparation for Analysis

Create columns for year, month, day, hour; ensure case of “Y/y, M/m, D/d, H/h, M/m, S/s” is correct.

### Extracting Year (4 number: “%Y”, 2 number: “%y”)

```
bike_rides$started_year <- format(bike_rides$started_at, "%Y")
str(bike_rides)
```

```
## 'data.frame':    4354488 obs. of  14 variables:
##  $ ride_id          : chr  "ACB6B40CF5B9044C" "DF450C72FD109C01" "B6396B54A15AC0DF"
##  $ bike_type        : chr  "electric_bike" "electric_bike" "electric_bike" "electric_bike" ...
##  $ started_at       : POSIXct, format: "2020-10-31 19:39:43" "2020-10-31 23:50:08" ...
##  $ ended_at         : POSIXct, format: "2020-10-31 19:57:12" "2020-11-01 00:04:16" ...
##  $ start_station_name: chr  "Lakeview Ave & Fullerton Pkwy" "Southport Ave & Wavelan
##  $ start_station_id  : chr  "313" "227" "102" "165" ...
##  $ end_station_name  : chr  "Rush St & Hubbard St" "Kedzie Ave & Milwaukee Ave" "Uni
##  $ end_station_id    : chr  "125" "260" "423" "256" ...
##  $ start_lat         : num  41.9 41.9 41.8 42 41.9 ...
##  $ start_lng         : num  -87.6 -87.7 -87.6 -87.7 -87.7 ...
##  $ end_lat           : num  41.9 41.9 41.8 42 41.9 ...
##  $ end_lng           : num  -87.6 -87.7 -87.6 -87.7 -87.7 ...
##  $ customer_type     : chr  "casual" "casual" "casual" "casual" ...
##  $ started_year      : chr  "2020" "2020" "2020" "2020" ...
```

### Extracting Month (abbreviation: “%h”, for number of month: “%m”, for full month name: “%B”)

```
bike_rides$started_month <- format(bike_rides$started_at, "%h")
str(bike_rides)
```

```
## 'data.frame':    4354488 obs. of  15 variables:
## $ ride_id      : chr  "ACB6B40CF5B9044C" "DF450C72FD109C01" "B6396B54A15AC0DF"
"44A4AEE261B9E854" ...
## $ bike_type    : chr  "electric_bike" "electric_bike" "electric_bike" "electric_bike" ...
## $ started_at   : POSIXct, format: "2020-10-31 19:39:43" "2020-10-31 23:50:08"
...
## $ ended_at     : POSIXct, format: "2020-10-31 19:57:12" "2020-11-01 00:04:16"
...
## $ start_station_name: chr  "Lakeview Ave & Fullerton Pkwy" "Southport Ave & Wavelan
d Ave" "Stony Island Ave & 67th St" "Clark St & Grace St" ...
## $ start_station_id : chr  "313" "227" "102" "165" ...
## $ end_station_name : chr  "Rush St & Hubbard St" "Kedzie Ave & Milwaukee Ave" "Uni
versity Ave & 57th St" "Broadway & Sheridan Rd" ...
## $ end_station_id   : chr  "125" "260" "423" "256" ...
## $ start_lat        : num  41.9 41.9 41.8 42 41.9 ...
## $ start_lng        : num  -87.6 -87.7 -87.6 -87.7 -87.7 ...
## $ end_lat          : num  41.9 41.9 41.8 42 41.9 ...
## $ end_lng          : num  -87.6 -87.7 -87.6 -87.7 -87.7 ...
## $ customer_type    : chr  "casual" "casual" "casual" "casual" ...
## $ started_year     : chr  "2020" "2020" "2020" "2020" ...
## $ started_month    : chr  "Oct" "Oct" "Oct" "Oct" ...
```

## Extracting Day of Week (Abbreviation “%a”, for full day of week name: “%A”, number: “%u”)

```
bike_rides$day_of_week <- format(bike_rides$started_at, "%a")
str(bike_rides)
```

```
## 'data.frame':    4354488 obs. of  16 variables:
## $ ride_id      : chr  "ACB6B40CF5B9044C" "DF450C72FD109C01" "B6396B54A15AC0DF"
"44A4AEE261B9E854" ...
## $ bike_type    : chr  "electric_bike" "electric_bike" "electric_bike" "electric_bike" ...
## $ started_at   : POSIXct, format: "2020-10-31 19:39:43" "2020-10-31 23:50:08"
...
## $ ended_at     : POSIXct, format: "2020-10-31 19:57:12" "2020-11-01 00:04:16"
...
## $ start_station_name: chr  "Lakeview Ave & Fullerton Pkwy" "Southport Ave & Wavelan
d Ave" "Stony Island Ave & 67th St" "Clark St & Grace St" ...
## $ start_station_id : chr  "313" "227" "102" "165" ...
## $ end_station_name : chr  "Rush St & Hubbard St" "Kedzie Ave & Milwaukee Ave" "Uni
versity Ave & 57th St" "Broadway & Sheridan Rd" ...
## $ end_station_id   : chr  "125" "260" "423" "256" ...
## $ start_lat        : num  41.9 41.9 41.8 42 41.9 ...
## $ start_lng        : num  -87.6 -87.7 -87.6 -87.7 -87.7 ...
## $ end_lat          : num  41.9 41.9 41.8 42 41.9 ...
## $ end_lng          : num  -87.6 -87.7 -87.6 -87.7 -87.7 ...
## $ customer_type    : chr  "casual" "casual" "casual" "casual" ...
## $ started_year     : chr  "2020" "2020" "2020" "2020" ...
## $ started_month    : chr  "Oct" "Oct" "Oct" "Oct" ...
## $ day_of_week      : chr  "Sat" "Sat" "Sat" "Sat" ...
```

Reset day of week levels.

```
bike_rides$day_of_week <- factor(bike_rides$day_of_week, levels = c("Sun", "Mon", "Tue", "Wed", "Thu", "Fri", "Sat"))
```

## Extracting Hour

```
bike_rides$started_hour <- format(bike_rides$started_at, "%H")
str(bike_rides)
```

```
## 'data.frame':    4354488 obs. of  17 variables:
##  $ ride_id          : chr  "ACB6B40CF5B9044C" "DF450C72FD109C01" "B6396B54A15AC0DF"
##  $ bike_type        : chr  "electric_bike" "electric_bike" "electric_bike" "electric_bike" ...
##  $ started_at       : POSIXct, format: "2020-10-31 19:39:43" "2020-10-31 23:50:08"
##  $ ended_at         : POSIXct, format: "2020-10-31 19:57:12" "2020-11-01 00:04:16"
##  $ start_station_name: chr  "Lakeview Ave & Fullerton Pkwy" "Southport Ave & Wavelan
##  $ start_station_id  : chr  "313" "227" "102" "165" ...
##  $ end_station_name  : chr  "Rush St & Hubbard St" "Kedzie Ave & Milwaukee Ave" "Uni
##  $ end_station_id    : chr  "125" "260" "423" "256" ...
##  $ start_lat         : num  41.9 41.9 41.8 42 41.9 ...
##  $ start_lng         : num  -87.6 -87.7 -87.6 -87.7 -87.7 ...
##  $ end_lat           : num  41.9 41.9 41.8 42 41.9 ...
##  $ end_lng           : num  -87.6 -87.7 -87.6 -87.7 -87.7 ...
##  $ customer_type     : chr  "casual" "casual" "casual" "casual" ...
##  $ started_year      : chr  "2020" "2020" "2020" "2020" ...
##  $ started_month     : chr  "Oct" "Oct" "Oct" "Oct" ...
##  $ day_of_week       : Factor w/ 7 levels "Sun","Mon","Tue",...: 7 7 7 7 7 5 5 5 5 4
##  $ started_hour      : chr  "19" "23" "23" "22" ...
```

## Calculating ride duration, calculated in minutes

```
bike_rides$ride_duration <- round(difftime(bike_rides$ended_at, bike_rides$started_at,
units = "mins"), 2)
str(bike_rides)
```

```
## 'data.frame':    4354488 obs. of  18 variables:
## $ ride_id      : chr  "ACB6B40CF5B9044C" "DF450C72FD109C01" "B6396B54A15AC0DF"
##               "44A4AEE261B9E854" ...
## $ bike_type    : chr  "electric_bike" "electric_bike" "electric_bike" "electric_bike" ...
## $ started_at   : POSIXct, format: "2020-10-31 19:39:43" "2020-10-31 23:50:08"
##               ...
## $ ended_at     : POSIXct, format: "2020-10-31 19:57:12" "2020-11-01 00:04:16"
##               ...
## $ start_station_name: chr  "Lakeview Ave & Fullerton Pkwy" "Southport Ave & Wavelan
##               d Ave" "Stony Island Ave & 67th St" "Clark St & Grace St" ...
## $ start_station_id : chr  "313" "227" "102" "165" ...
## $ end_station_name : chr  "Rush St & Hubbard St" "Kedzie Ave & Milwaukee Ave" "Uni
##               versity Ave & 57th St" "Broadway & Sheridan Rd" ...
## $ end_station_id   : chr  "125" "260" "423" "256" ...
## $ start_lat        : num  41.9 41.9 41.8 42 41.9 ...
## $ start_lng        : num  -87.6 -87.7 -87.6 -87.7 -87.7 ...
## $ end_lat          : num  41.9 41.9 41.8 42 41.9 ...
## $ end_lng          : num  -87.6 -87.7 -87.6 -87.7 -87.7 ...
## $ customer_type    : chr  "casual" "casual" "casual" "casual" ...
## $ started_year      : chr  "2020" "2020" "2020" "2020" ...
## $ started_month     : chr  "Oct" "Oct" "Oct" "Oct" ...
## $ day_of_week       : Factor w/ 7 levels "Sun","Mon","Tue",...: 7 7 7 7 7 5 5 5 5 4
##               ...
## $ started_hour      : chr  "19" "23" "23" "22" ...
## $ ride_duration     : 'difftime' num  17.48 14.13 8.35 2.87 ...
## $ .. attr(*, "units")= chr "mins"
```

```
class(bike_rides$ride_duration)
```

```
## [1] "difftime"
```

```
is.numeric(bike_rides$ride_duration)
```

```
## [1] FALSE
```

I would like to convert the ride duration into numeric type, as I will use it to analyze the difference between members and casual users.

## Changing the Type of ride\_duration

```
bike_rides$ride_duration <- as.numeric(bike_rides$ride_duration)
```

```
is.numeric(bike_rides$ride_duration)
```

```
## [1] TRUE
```


# Full Summary of the Table

```
skim(bike_rides$ride_duration)
```

## Data summary

Name	bike_rides\$ride_duration
Number of rows	4354488
Number of columns	1
Column type frequency:	
numeric	1
Group variables	
None	

## Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
data	0	1	20.57	308.8	-29049.97	7.2	12.7	22.98	55944.15	

As we can see that, there is some negative values for ride duration. It means that ended date is earlier than started date. It must be errors of the data set. We have to delete these errors to ensure the following analysis correct. I use filter to delete these errors.

# Removing Rides Duration < 0


```
bike_rides_2 <- bike_rides %>%  
  filter(ride_duration >= 0)
```

```
skim(bike_rides_2$ride_duration)
```

## Data summary

Name	bike_rides_2\$ride_duratio...
Number of rows	4351282
Number of columns	1
Column type frequency:	
numeric	1
Group variables	
None	

Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
data	0	1	22.65	190.07	0	7.22	12.7	22.98	55944.15	

There is no neagtive values in ride\_duration.

## Checking outlier

```
#table(bike_rides_2$ride_duration)
```

From the frequencies of less than 1 minutes, it is fairly represented. As there are hundreds of data points for rides less than 1 minutes, it means that it is not random errors. So I will keep these data points which less than 1 minutes.

There are still data points which are larger than 1440 minutes (1 day). The maximum rides is 58720.03 minutes, it means 40 days. It seems that it is impossible. I will treat these outliers as errors and should be deleted. I will delete the values which is tested as outliers.

## Checking for outlier data: Rosner test to detect multiple outliers.

```
#install.packages("EnvStats")  
library(EnvStats)
```

```
##  
## Attaching package: 'EnvStats'
```

```
## The following objects are masked from 'package:stats':  
##  
## predict, predict.lm
```

## Checking Outliers

```
outliers <- rosnerTest(bike_rides_2$ride_duration, k = 10)  
outliers
```

```
##
## Results of Outlier Test
## -----
##
## Test Method:                Rosner's Test for Outliers
##
## Hypothesized Distribution:   Normal
##
## Data:                       bike_rides_2$ride_duration
##
## Sample Size:                4351282
##
## Test Statistics:            R.1  = 294.2206
##                             R.2  = 295.8499
##                             R.3  = 289.3681
##                             R.4  = 285.5782
##                             R.5  = 268.6228
##                             R.6  = 263.5360
##                             R.7  = 261.4041
##                             R.8  = 233.3958
##                             R.9  = 234.7783
##                             R.10 = 231.1427
##
## Test Statistic Parameter:   k = 10
##
## Alternative Hypothesis:     Up to 10 observations are not
##                             from the same Distribution.
##
## Type I Error:              5%
##
## Number of Outliers Detected: 10
##
##      i   Mean.i      SD.i    Value Obs.Num   R.i+1 lambda.i+1 Outlier
## 1  0 22.65300 190.0666 55944.15 2261412 294.2206   5.707102   TRUE
## 2  1 22.64014 188.1665 55691.68 2214891 295.8499   5.707102   TRUE
## 3  2 22.62735 186.2644 53921.60 1345521 289.3681   5.707102   TRUE
## 4  3 22.61496 184.4635 52701.38 1824058 285.5782   5.707102   TRUE
## 5  4 22.60286 182.7267 49107.15 2914851 268.6228   5.707102   TRUE
## 6  5 22.59158 181.2053 47776.70 1014997 263.5360   5.707102   TRUE
## 7  6 22.58060 179.7533 47010.85 2216083 261.4041   5.707102   TRUE
## 8  7 22.56980 178.3364 41645.52 2377354 233.3958   5.707102   TRUE
## 9  8 22.56024 177.2166 41629.17 3206609 234.7783   5.707102   TRUE
## 10 9 22.55068 176.0905 40724.60 2677751 231.1427   5.707102   TRUE
```

There are many outliers presented above. I will use IQR method to remove outliers.

## Checking Outliers

```
bike_rides_2 <- arrange(bike_rides_2, desc(ride_duration))
```

In the column of ride. duration, there are many values which are over 1440 minutes.



# Calculating upper and lower limit

```
quartiles <- quantile(bike_rides_2$ride_duration, probs = c(0.25, 0.75), na.rm = F)
IQR <- IQR(bike_rides_2$ride_duration)
```

```
quartiles
```

```
##      25%      75%  
##  7.22 22.98
```

```
IQR
```

```
## [1] 15.76
```

```
lower <- quartiles[1] - 1.5*IQR  
upper <- quartiles[2] + 1.5*IQR
```

```
lower
```

```
##      25%  
## -16.42
```

```
upper
```

```
##      75%  
## 46.62
```

## Removing Outliers

```
bike_rides_rm_outlier <- subset(bike_rides_2, ride_duration > lower & ride_duration < upper)
```

## Checking Outliers Again

```
outliers <- rosnerTest(bike_rides_rm_outlier$ride_duration, k = 3)  
outliers
```

```
##
## Results of Outlier Test
## -----
##
## Test Method:                Rosner's Test for Outliers
##
## Hypothesized Distribution:   Normal
##
## Data:                       bike_rides_rm_outlier$ride_duration
##
## Sample Size:                4022037
##
## Test Statistics:            R.1 = 3.203800
##                             R.2 = 3.203805
##                             R.3 = 3.203809
##
## Test Statistic Parameter:    k = 3
##
## Alternative Hypothesis:      Up to 3 observations are not
##                             from the same Distribution.
##
## Type I Error:               5%
##
## Number of Outliers Detected: 0
##
##   i   Mean.i      SD.i Value Obs.Num   R.i+1 lambda.i+1 Outlier
## 1 0 14.46386 10.03687 46.62      1 3.203800   5.693689  FALSE
## 2 1 14.46385 10.03686 46.62      2 3.203805   5.693689  FALSE
## 3 2 14.46384 10.03685 46.62      3 3.203809   5.693689  FALSE
```

The Rosner's test result show that there are no significant outliers in our trimmed data set. However, I would like to take a look the data which has been deleted to check if it is reasonable to remove them. So I create several datasets to compare the proportion of members and casuals.

```
bike_less_47 <- bike_rides_rm_outlier
```

```
bike_test <- bike_rides_2 %>%
  mutate(duration_bins
    = case_when(ride_duration >=0 & ride_duration<=60 ~ "0_1_h",
                ride_duration >60 & ride_duration<=120 ~ "1_2_h",
                ride_duration >120 & ride_duration<=180 ~ "2_3_h",
                ride_duration >180 & ride_duration<=240 ~ "3_4_h",
                ride_duration >240 & ride_duration<=300 ~ "4_5_h",
                ride_duration >300 & ride_duration<=360 ~ "5_6_h",
                ride_duration >360 & ride_duration<=420 ~ "6_7_h",
                ride_duration >420 & ride_duration<=480 ~ "7_8_h",
                ride_duration >480 ~ "8_h_plus"))
```

```
str(bike_test)
```

```
## 'data.frame':    4351282 obs. of  19 variables:
## $ ride_id      : chr  "F043F0F6A1AA4F85" "7F0578ABF030FC83" "BDA1217EC8532C7B"
## $ bike_type    : chr  "docked_bike" "docked_bike" "docked_bike" "docked_bike"
## $ started_at   : POSIXct, format: "2021-06-05 02:27:26" "2021-06-04 22:03:33"
## $ ended_at     : POSIXct, format: "2021-07-13 22:51:35" "2021-07-13 14:15:14"
## $ start_station_name: chr  "Michigan Ave & Lake St" "Streeter Dr & Grand Ave" "Stat
## $ start_station_id : chr  "TA1305000011" "13022" "TA1305000035" "KA1503000012" ...
## $ end_station_name : chr  "Malcolm X College Vaccination Site" "Base - 2132 W Hubb
## $ end_station_id   : chr  "631" "Hubbard Bike-checking (LBS-WH-TEST)" "SL-011" "Hu
## $ start_lat        : num  41.9 41.9 41.9 41.9 41.9 ...
## $ start_lng        : num  -87.6 -87.6 -87.6 -87.6 -87.7 ...
## $ end_lat          : num  41.9 41.9 41.9 41.9 41.9 ...
## $ end_lng          : num  -87.7 -87.7 -87.6 -87.7 -87.7 ...
## $ customer_type    : chr  "casual" "casual" "casual" "casual" ...
## $ started_year      : chr  "2021" "2021" "2021" "2021" ...
## $ started_month     : chr  "Jun" "Jun" "May" "Jun" ...
## $ day_of_week       : Factor w/ 7 levels "Sun","Mon","Tue",...: 7 6 1 7 5 6 7 5 1 7
## $ started_hour      : chr  "02" "22" "02" "23" ...
## $ ride_duration     : num  55944 55692 53922 52701 49107 ...
## $ duration_bins     : chr  "8_h_plus" "8_h_plus" "8_h_plus" "8_h_plus" ...
```

```
bike_over_47 <- bike_rides_2 %>%
  filter(ride_duration > 46.62)
```

```
bike_47_12h <- bike_rides_2 %>%
  filter(ride_duration > 46.62 & ride_duration <= 720)
nrow(bike_47_12h)
```

```
## [1] 325280
```

There are 325280 records where ride duration is between 46.62 minutes and 720 minutes (12 hour)

```
bike_47_8h <- bike_rides_2 %>%
  filter(ride_duration > 46.62 & ride_duration <= 480)
nrow(bike_47_8h)
```

```
## [1] 323723
```

```
bike_over_12h <- bike_over_47 %>%
  filter(ride_duration > 720)
nrow(bike_over_12h)
```

```
## [1] 3965
```

There are 3965 records where ride duration are over 12 hour.

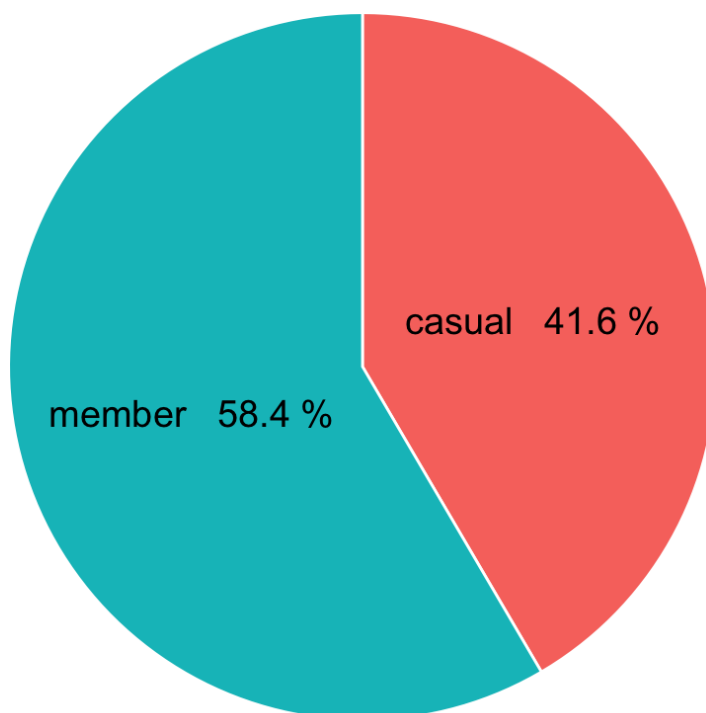
```
bike_over_day <- bike_over_47 %>%  
  filter(ride_duration > 1440)  
nrow(bike_over_day)
```

```
## [1] 1389
```

There are only 1389 records where ride duration are over a day.

```
bike_less_47 %>%  
  group_by(customer_type) %>%  
  summarise(count = n()) %>%  
  ggplot(aes(x = "", y=count, fill = customer_type)) +  
  geom_bar(stat = "identity", width = 1, color = "white") +  
  coord_polar("y", start = 0, direction = -1) +  
  theme_void() +  
  theme(axis.title = element_blank(), axis.ticks = element_blank(), plot.title = element_text(hjust = 1)) +  
  ggtitle("Rides Per Customer Type") +  
  theme(plot.title = element_text(hjust = 0.5, vjust = 0.5, face = "bold", size = 20)) +  
  geom_text(aes(label = paste(customer_type, " ", round(count / sum(count) * 100, 1), "%")), position = position_stack(vjust = 0.5), size = 5) +  
  theme(legend.position = "none")
```

## Rides Per Customer Type



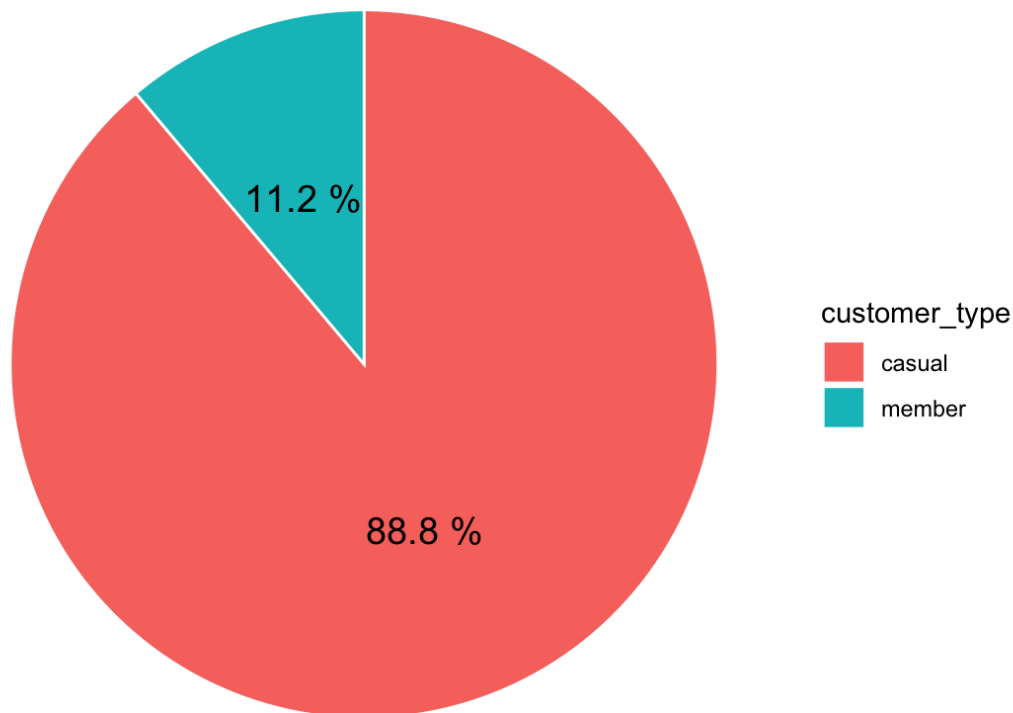
#### Checking the outliers distribution

```

bike_over_47 %>%
  group_by(customer_type) %>%
  summarise(count = n()) %>%
  ggplot(aes(x = "", y=count, fill = customer_type +
    geom_bar(stat = "identity", width = 1, color = "white" +
    coord_polar("y", start = 0, direction = -1) +
    theme_void( +
      theme(axis.title = element_blank(), axis.ticks = element_blank(), plot.title = element_text(hjust = 1)) +
      ggtitle("Rides Per Customer Type") +
      theme(plot.title = element_text(hjust = 0.5, vjust = 0.5, face = "bold", size = 20))
    +
      geom_text(aes(label = paste(round(count / sum(count) * 100, 1), "%")), position = position_stack(vjust = 0.5), size=5)

```

## Rides Per Customer Type

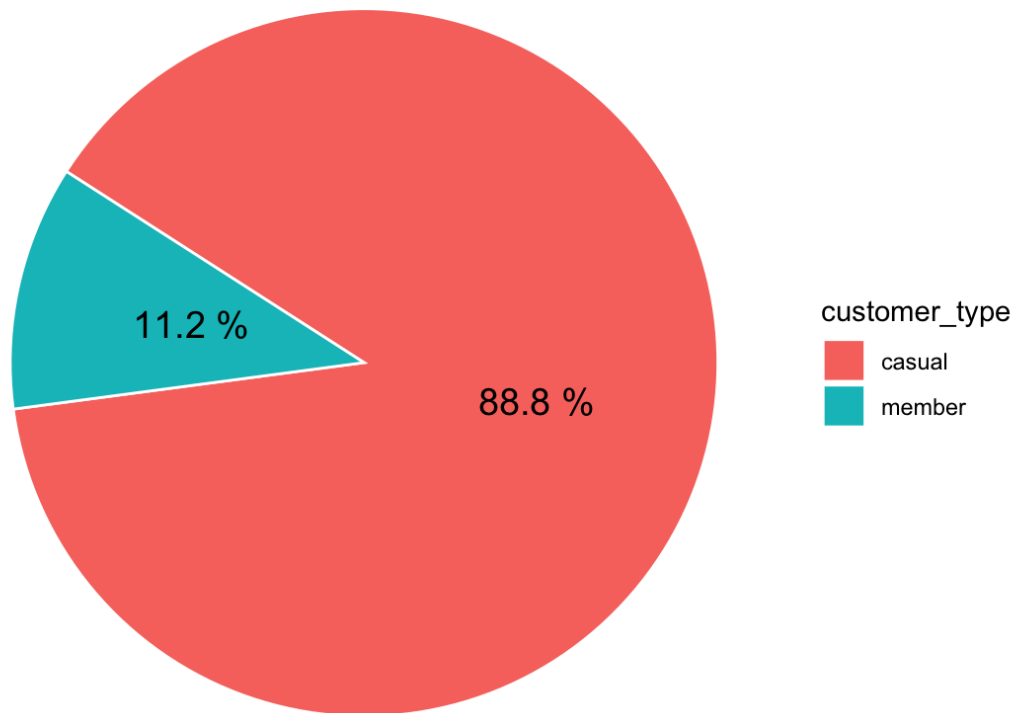


```

bike_47_12h %>%
  group_by(customer_type) %>%
  summarise(count = n()) %>%
  ggplot(aes(x = "", y = count, fill = customer_type)) +
  geom_bar(stat = "identity", width = 1, color = "white") +
  coord_polar("y", start = 1, direction = -1) +
  theme_void() +
  theme(axis.title = element_blank(), axis.ticks = element_blank(), plot.title = element_text(hjust = 1)) +
  ggtitle("Rides per Customer Type (Over 47 and less 12h)") +
  theme(plot.title = element_text(hjust = 0.4, vjust = 0.5, size = 20, face = "bold")) +
  geom_text(aes(label = paste(round(count / sum(count)*100, 1), "%")), position = position_stack(vjust = 0.5), size = 5)

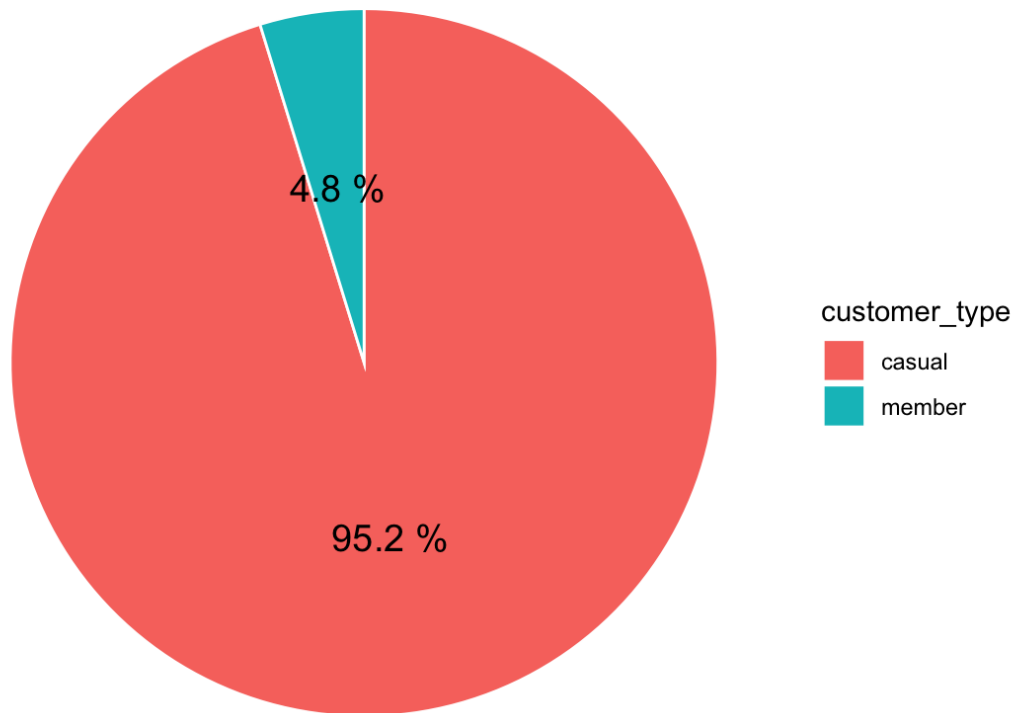
```

## Rides per Customer Type (Over 47 and less 12h)

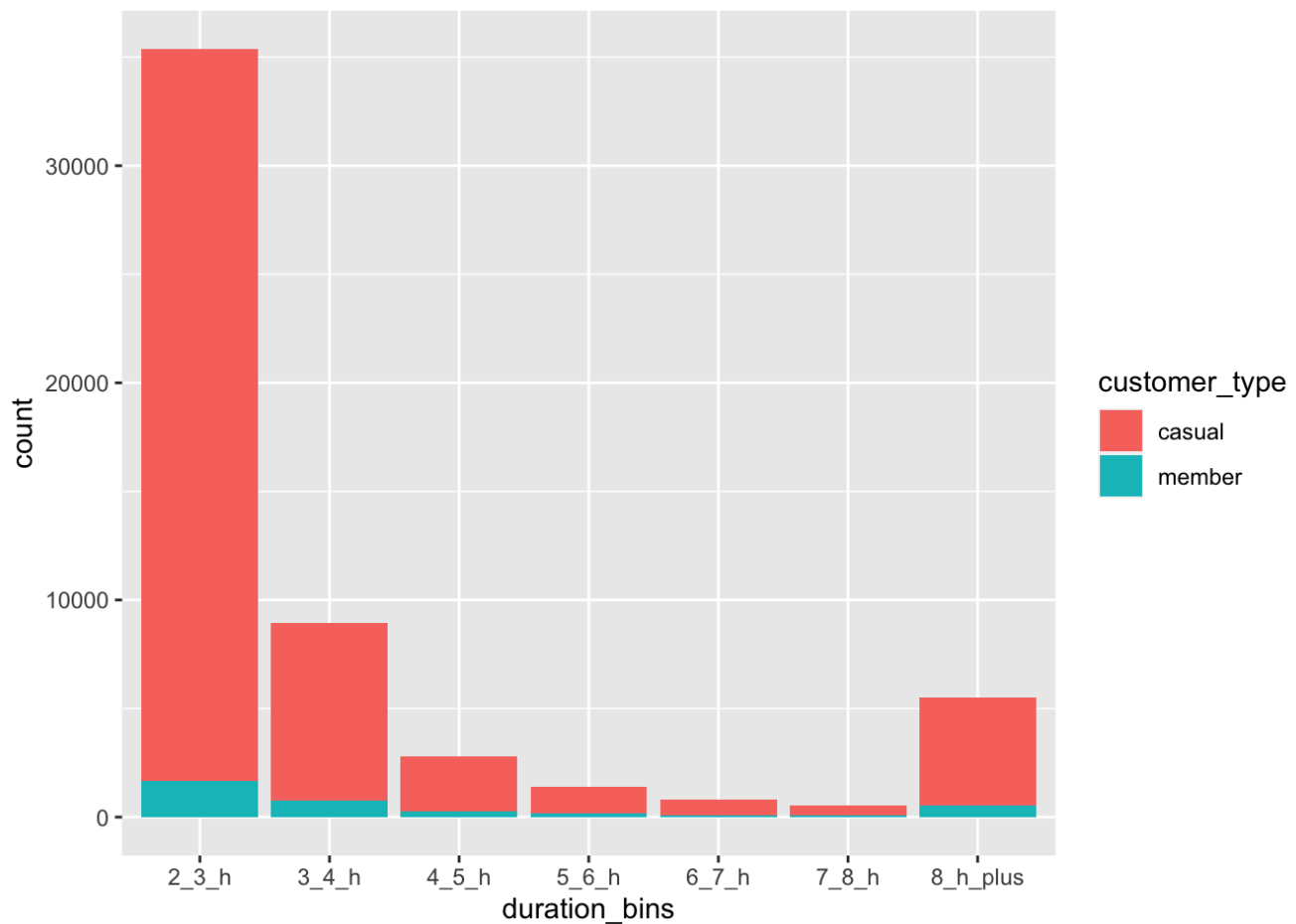


```
bike_over_day %>%
  group_by(customer_type) %>%
  summarise(count = n()) %>%
  ggplot(aes(x="", y = count, fill = customer_type)) +
  geom_bar(stat = "identity", width = 1, color = "white") +
  coord_polar("y", start = 0, direction = -1) +
  theme_void() +
  theme(axis.title = element_blank(), axis.ticks = element_blank(), plot.title = element_text(hjust = 1)) +
  ggtitle("Rides per Customer Type (Over a Day)") +
  theme(plot.title = element_text(hjust = 0.5, vjust = 0.5, face = "bold", size = 20))
+
  geom_text(aes(label = paste(round(count / sum(count)*100, 1), "%")), position = position_stack(vjust = 0.5), size = 5)
```

# Rides per Customer Type (Over a Day)



```
bike_test %>%  
  filter(!duration_bins == "0_1_h" & !duration_bins == "1_2_h") %>%  
  group_by(customer_type) %>%  
  ggplot(aes(x = duration_bins, fill = customer_type)) +  
  geom_bar()
```



```
bike_test_5h <- bike_test %>%
  filter(duration_bins == "4_5_h")
```

As we can see that, from 4 hour, the rides number drop sharply. Between 240 and 300 minutes, there are only 2785 rides. So, I think it is reasonable to delete outlier which are longer than 300 minutes, rather than IQR. With IQR, it will remove over 300,000 rides and over 90% of the outliers are casuals. If I remove these values which are over 46.62 minutes, I may lose the chance to discover the casual customer's behavior.

With this, we can then proceed with the succeeding statistical analysis using our trimmed data.

```
bike_rides_less_5h <- bike_rides_2 %>%
  filter(ride_duration <= 300)
```

```
rosnerTest(bike_rides_less_5h$ride_duration, k = 10)
```



```
##
## Results of Outlier Test
## -----
##
## Test Method:                Rosner's Test for Outliers
##
## Hypothesized Distribution:   Normal
##
## Data:                       bike_rides_less_5h$ride_duration
##
## Sample Size:                4343050
##
## Test Statistics:            R.1  = 12.10182
##                             R.2  = 12.10159
##                             R.3  = 12.10007
##                             R.4  = 12.09898
##                             R.5  = 12.09832
##                             R.6  = 12.09810
##                             R.7  = 12.09658
##                             R.8  = 12.09635
##                             R.9  = 12.09569
##                             R.10 = 12.09503
##
## Test Statistic Parameter:   k = 10
##
## Alternative Hypothesis:     Up to 10 observations are not
##                             from the same Distribution.
##
## Type I Error:              5%
##
## Number of Outliers Detected: 10
##
##      i   Mean.i      SD.i  Value Obs.Num   R.i+1 lambda.i+1 Outlier
## 1  0 19.55842 23.16772 299.93      1 12.10182   5.70678   TRUE
## 2  1 19.55835 23.16734 299.92      2 12.10159   5.70678   TRUE
## 3  2 19.55829 23.16695 299.88      3 12.10007   5.70678   TRUE
## 4  3 19.55822 23.16656 299.85      4 12.09898   5.70678   TRUE
## 5  4 19.55816 23.16617 299.83      5 12.09832   5.70678   TRUE
## 6  5 19.55809 23.16579 299.82      6 12.09810   5.70678   TRUE
## 7  6 19.55803 23.16540 299.78      7 12.09658   5.70678   TRUE
## 8  7 19.55797 23.16501 299.77      8 12.09635   5.70678   TRUE
## 9  8 19.55790 23.16462 299.75      9 12.09569   5.70678   TRUE
## 10 9 19.55784 23.16423 299.73     10 12.09503   5.70678   TRUE
```

Even the outlier check shows that there are some outliers, I won't remove them based on the previous analysis.

```
skim(bike_rides_less_5h$ride_duration)
```

#### Data summary

Name	bike_rides_less_5h\$ride_d...
Number of rows	4343050
Number of columns	1

---


Column type frequency:

numeric	1
---------	---

---

Group variables	None
-----------------	------

**Variable type: numeric**

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
data	0	1	19.56	23.17	0	7.2	12.68	22.9	299.93	

## Checking duplicate

```
bike_rides_less_5h$ride_id[duplicated(bike_rides_less_5h$ride_id)]
```

```
## character(0)
```

There is no duplicates in ride\_id.

Now, I have finished the data cleaning and transforming. I will start to analyze the data in the following part.

## Name the dataset for the following analysis: bike\_cleaned

```
bike_cleaned <- bike_rides_less_5h  
skim(bike_cleaned)
```

Data summary

Name	bike_cleaned
Number of rows	4343050
Number of columns	18

---

Column type frequency:

character	10
factor	1
numeric	5
POSIXct	2

---

Group variables	None
-----------------	------

**Variable type: character**

skim_variable	n_missing	complete_rate	min	max	empty	n_unique	whitespace
ride_id	0	1	16	16	0	4343050	0
bike_type	0	1	11	13	0	3	0
start_station_name	0	1	3	53	0	781	0
start_station_id	259	1	1	36	0	1296	0
end_station_name	0	1	10	53	0	776	0
end_station_id	194	1	1	36	0	1294	0
customer_type	0	1	6	6	0	2	0
started_year	0	1	4	4	0	2	0
started_month	0	1	3	3	0	12	0
started_hour	0	1	2	2	0	24	0

#### Variable type: factor

skim_variable	n_missing	complete_rate	ordered	n_unique	top_counts
day_of_week	0	1	FALSE	7	Sat: 789472, Sun: 675808, Fri: 623443, Thu: 583858

#### Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
start_lat	0	1	41.90	0.04	41.65	41.88	41.90	41.93	42.06	
start_lng	0	1	-87.64	0.02	-87.83	-87.66	-87.64	-87.63	-87.53	
end_lat	0	1	41.90	0.04	41.65	41.88	41.90	41.93	42.17	
end_lng	0	1	-87.64	0.02	-87.83	-87.66	-87.64	-87.63	-87.52	
ride_duration	0	1	19.56	23.17	0.00	7.20	12.68	22.90	299.93	

#### Variable type: POSIXct

skim_variable	n_missing	complete_rate	min	max	median	n_unique
started_at	0	1	2020-10-01 00:00:06	2021-09-30 23:59:44	2021-06-19 19:51:09	3724611
ended_at	0	1	2020-10-01 00:05:09	2021-10-01 01:23:49	2021-06-19 20:16:56	3712429

## Checking Ride Duration

```
bike_cleaned_drd <- arrange(bike_cleaned, desc(ride_duration))
summary(bike_cleaned$ride_duration)
```

##	Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
##	0.00	7.20	12.68	19.56	22.90	299.93

Creating Bins for Ride Duration

```
bike_cleaned_2 <- bike_cleaned %>%
  mutate(duration_bins
    = case_when(ride_duration >=0 & ride_duration<=5 ~ "0_5_mins",
      ride_duration >5 & ride_duration<=10 ~ "5_10_mins",
      ride_duration >10 & ride_duration<=15 ~ "10_15_mins",
      ride_duration >15 & ride_duration<=20 ~ "15_20_mins",
      ride_duration >20 & ride_duration<=25 ~ "20_25_mins",
      ride_duration >25 & ride_duration<=30 ~ "25_30_mins",
      ride_duration >30 & ride_duration<=35 ~ "30_35_mins",
      ride_duration >35 & ride_duration<=40 ~ "35_40_mins",
      ride_duration >40 & ride_duration<=45 ~ "40_45_mins",
      ride_duration >45 & ride_duration<=50 ~ "45_50_mins",
      ride_duration >50 & ride_duration<=60 ~ "50_60_mins",
      ride_duration >60 & ride_duration<=90 ~ "60_90_mins",
      ride_duration >90 & ride_duration<=150 ~ "90_150_mins",
      ride_duration >150 ~ "over_150_mins"))
```

```
skim(bike_cleaned_2)
```

Data summary

Name	bike_cleaned_2
Number of rows	4343050
Number of columns	19
Column type frequency:	
character	11
factor	1
numeric	5
POSIXct	2
Group variables	
None	

Variable type: character

skim_variable	n_missing	complete_rate	min	max	empty	n_unique	whitespace
ride_id	0	1	16	16	0	4343050	0
bike_type	0	1	11	13	0	3	0
start_station_name	0	1	3	53	0	781	0

skim_variable	n_missing	complete_rate	min	max	empty	n_unique	whitespace
start_station_id	259	1	1	36	0	1296	0
end_station_name	0	1	10	53	0	776	0
end_station_id	194	1	1	36	0	1294	0
customer_type	0	1	6	6	0	2	0
started_year	0	1	4	4	0	2	0
started_month	0	1	3	3	0	12	0
started_hour	0	1	2	2	0	24	0
duration_bins	0	1	8	13	0	14	0

Variable type: factor

skim_variable	n_missing	complete_rate	ordered	n_unique	top_counts
day_of_week	0	1	FALSE	7	Sat: 789472, Sun: 675808, Fri: 623443, Thu: 583858

Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
start_lat	0	1	41.90	0.04	41.65	41.88	41.90	41.93	42.06	
start_lng	0	1	-87.64	0.02	-87.83	-87.66	-87.64	-87.63	-87.53	
end_lat	0	1	41.90	0.04	41.65	41.88	41.90	41.93	42.17	
end_lng	0	1	-87.64	0.02	-87.83	-87.66	-87.64	-87.63	-87.52	
ride_duration	0	1	19.56	23.17	0.00	7.20	12.68	22.90	299.93	

Variable type: POSIXct

skim_variable	n_missing	complete_rate	min	max	median	n_unique
started_at	0	1	2020-10-01 00:00:06	2021-09-30 23:59:44	2021-06-19 19:51:09	3724611
ended_at	0	1	2020-10-01 00:05:09	2021-10-01 01:23:49	2021-06-19 20:16:56	3712429

colnames(bike\_cleaned\_2)

```
## [1] "ride_id"          "bike_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] "customer_type"    "started_year"     "started_month"
## [16] "day_of_week"      "started_hour"     "ride_duration"
## [19] "duration_bins"
```

```
head(bike_cleaned_2)
```

```
##          ride_id    bike_type      started_at      ended_at
## 1 E5CF6494F2CF5635 classic_bike 2021-05-02 17:00:30 2021-05-02 22:00:26
## 2 8C616C19B29AC3F9 docked_bike 2021-09-06 13:33:38 2021-09-06 18:33:33
## 3 71CD7586C02F9D58 docked_bike 2021-09-04 10:41:12 2021-09-04 15:41:05
## 4 5C2A0EBF288CF749 classic_bike 2021-07-27 12:35:56 2021-07-27 17:35:47
## 5 BA59BFC7426B7D5E docked_bike 2021-07-23 12:15:39 2021-07-23 17:15:29
## 6 531BEF5239B4D05C classic_bike 2021-06-29 16:13:09 2021-06-29 21:12:58
##          start_station_name start_station_id
## 1      Fort Dearborn Dr & 31st St      TA1307000048
## 2 DuSable Lake Shore Dr & North Blvd      LF-005
## 3 DuSable Lake Shore Dr & North Blvd      LF-005
## 4      Clarendon Ave & Gordon Ter      13379
## 5      Streeter Dr & Grand Ave      13022
## 6      Rush St & Superior St      15530
##          end_station_name      end_station_id
## 1      McClurg Ct & Erie St      KA1503000041
## 2 DuSable Lake Shore Dr & North Blvd      LF-005
## 3 Orleans St & Merchandise Mart Plaza      TA1305000022
## 4      Clarendon Ave & Gordon Ter      13379
## 5      State St & Randolph St      TA1305000029
## 6      Base - 2132 W Hubbard Warehouse Hubbard Bike-checking (LBS-WH-TEST)
## start_lat start_lng end_lat end_lng customer_type started_year
## 1 41.83856 -87.60822 41.89450 -87.61785      casual      2021
## 2 41.91172 -87.62680 41.91172 -87.62680      casual      2021
## 3 41.91172 -87.62680 41.88824 -87.63639      casual      2021
## 4 41.95787 -87.64951 41.95787 -87.64951      member      2021
## 5 41.89228 -87.61204 41.88468 -87.62798      casual      2021
## 6 41.89576 -87.62591 41.88995 -87.68065      member      2021
## started_month day_of_week started_hour ride_duration duration_bins
## 1      May      Sun      17      299.93 over_150_mins
## 2      Sep      Mon      13      299.92 over_150_mins
## 3      Sep      Sat      10      299.88 over_150_mins
## 4      Jul      Tue      12      299.85 over_150_mins
## 5      Jul      Fri      12      299.83 over_150_mins
## 6      Jun      Tue      16      299.82 over_150_mins
```

```
tail(bike_cleaned_2)
```

##	ride_id	bike_type	started_at	ended_at		
## 4343045	7E8BDEE052DDB6B9	classic_bike	2021-09-28 17:39:07	2021-09-28 17:39:07		
## 4343046	1C00E7445599C55A	classic_bike	2021-09-18 20:26:23	2021-09-18 20:26:23		
## 4343047	AB0227900FBBA279	classic_bike	2021-09-04 15:53:04	2021-09-04 15:53:04		
## 4343048	84D8C8E8FDCBBFB0	electric_bike	2021-09-12 17:29:00	2021-09-12 17:29:00		
## 4343049	0C32DAA0DDCAA66B	electric_bike	2021-09-05 15:16:10	2021-09-05 15:16:10		
## 4343050	8590ABC3CD2BBC35	classic_bike	2021-09-22 18:51:46	2021-09-22 18:51:46		
##	start_station_name	start_station_id				
## 4343045	Halsted St & Clybourn Ave	331				
## 4343046	Racine Ave & Fullerton Ave	TA1306000026				
## 4343047	DuSable Lake Shore Dr & North Blvd	LF-005				
## 4343048	Wabash Ave & 9th St	TA1309000010				
## 4343049	Sheffield Ave & Wrightwood Ave	TA1309000023				
## 4343050	Sedgwick St & Webster Ave	13191				
##	end_station_name	end_station_id	start_lat	start_lng		
## 4343045	Halsted St & Clybourn Ave	331	41.90967	-87.64813		
## 4343046	Racine Ave & Fullerton Ave	TA1306000026	41.92556	-87.65840		
## 4343047	DuSable Lake Shore Dr & North Blvd	LF-005	41.91172	-87.62680		
## 4343048	Wabash Ave & 9th St	TA1309000010	41.87077	-87.62580		
## 4343049	Sheffield Ave & Wrightwood Ave	TA1309000023	41.92864	-87.65378		
## 4343050	Sedgwick St & Webster Ave	13191	41.92217	-87.63889		
##	end_lat	end_lng	customer_type	started_year	started_month	day_of_week
## 4343045	41.90967	-87.64813	casual	2021	Sep	Tue
## 4343046	41.92556	-87.65840	member	2021	Sep	Sat
## 4343047	41.91172	-87.62680	member	2021	Sep	Sat
## 4343048	41.87083	-87.62583	member	2021	Sep	Sun
## 4343049	41.92865	-87.65378	casual	2021	Sep	Sun
## 4343050	41.92217	-87.63889	member	2021	Sep	Wed
##	started_hour	ride_duration	duration_bins			
## 4343045	17	0	0_5_mins			
## 4343046	20	0	0_5_mins			
## 4343047	15	0	0_5_mins			
## 4343048	17	0	0_5_mins			
## 4343049	15	0	0_5_mins			
## 4343050	18	0	0_5_mins			

## Summary of Data Preparation

- We generated a column to categorize the year, the month, days of the week, and the hour. The ride duration, measured in minutes, was subsequently calculated.
- To identify outlier data, we utilized the Rosner Test.
- The upper and lower limits of data inclusion were computed using the following formulas:
  - Lower Limit = Quartile[1] - 1.5IQR
  - Upper Limit = Quartile[2] + 1.5IQR
- When I attempted to exclude outliers using the Interquartile Range (IQR) method, I observed an imbalance in the removed values, with a disproportionate number of casual values being affected. Upon examining the data for ride durations longer than 46.62 minutes (the threshold for the upper limit), I discovered over 300,000 rows, with 90% of them being casual rides. Therefore, I made the decision to delete only the rides lasting more than 300 minutes. This approach seemed reasonable, as it is not uncommon for people to rent bikes for periods of less than 5 hours.
- The rides were classified into bins in increments of minutes for ease of analysis.

```
# options(max.print=1000000000)
```

# Phase 4: Data Analysis

## Descriptive Analysis

### H1 Rides difference by customer type

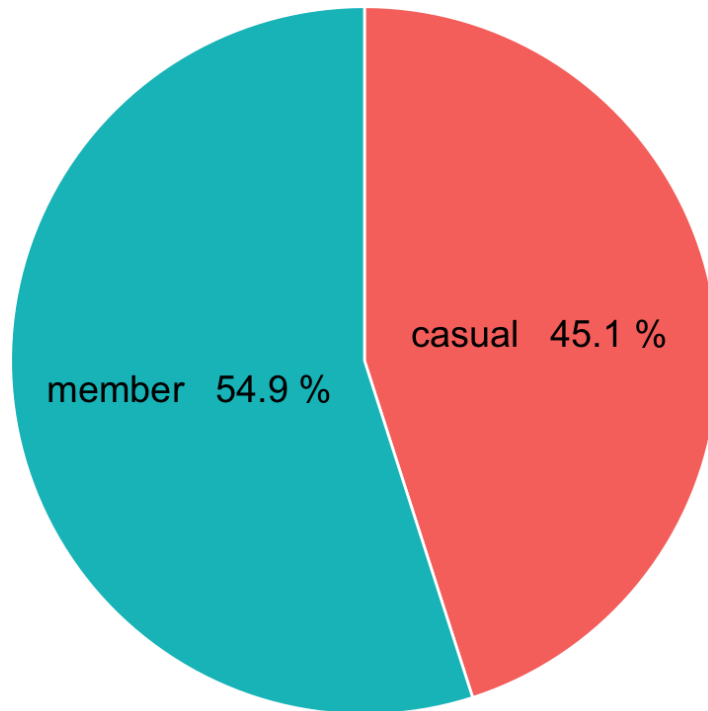
**H1: Members use bikes more often than casuals.**

#### Pie chart of percent per customer type

```
bike_cleaned_2 %>%
  group_by(customer_type) %>%
  summarise(count = n()) %>%
  ggplot(aes(x = "", y=count, fill = customer_type)) +
  geom_bar(stat = "identity", width = 1, color = "white") +
  coord_polar("y", start = 0, direction = -1) +
  theme_void() +
  theme(axis.title = element_blank(), axis.ticks = element_blank(), plot.title = element_text(hjust = 1)) +
  ggtitle("Rides Percent per Customer Type") +
  theme(plot.title = element_text(hjust = 0.5, vjust = 0.5, face = "bold", size = 20))
+
  geom_text(aes(label = paste(customer_type, " ", round(count / sum(count) * 100, 1), "%")), position = position_stack(vjust = 0.5), size=5) +
  theme(legend.position = "none")
```



# Rides Percent per Customer Type

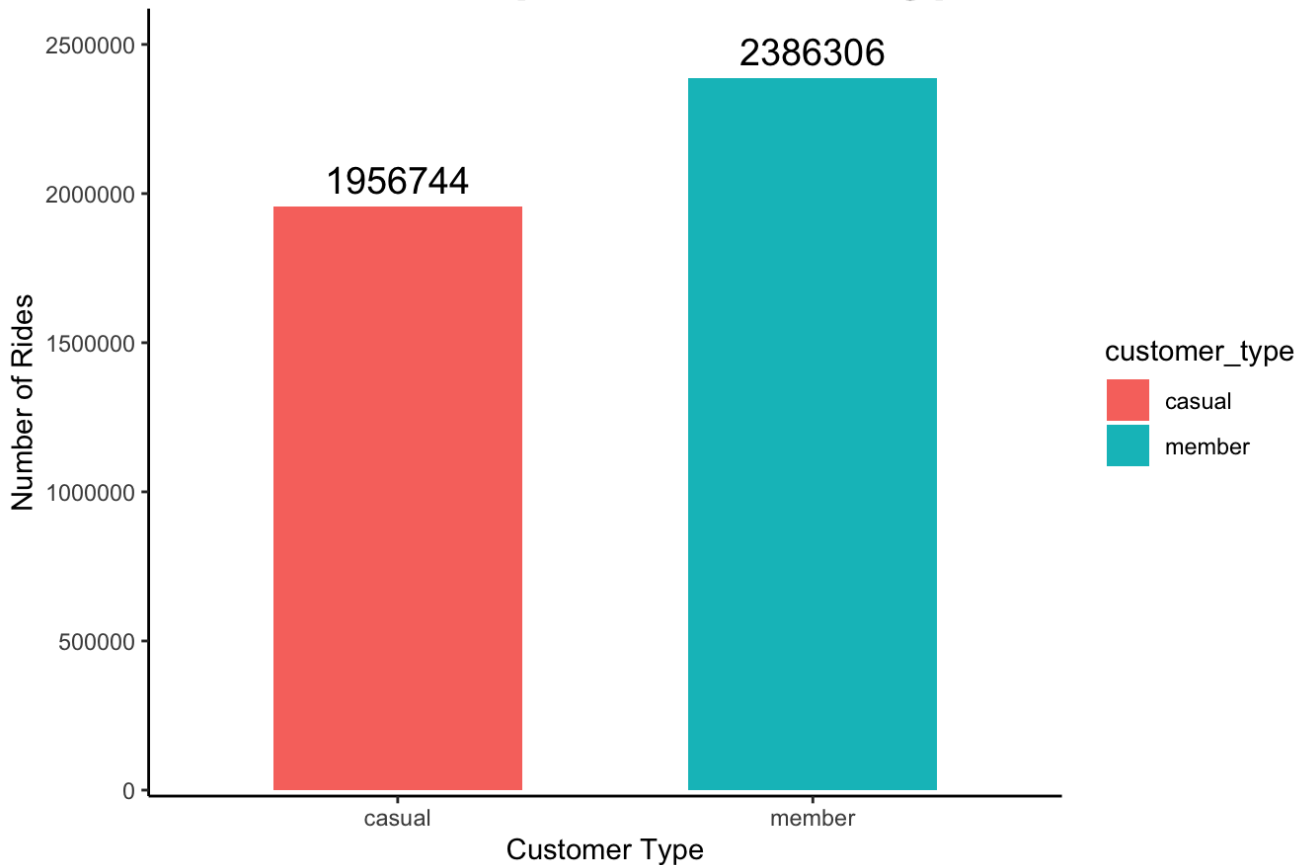


Member riders account for 54.9% while casual riders account for 45.1%.

## Bar chart of rides by customer type

```
ggplot(bike_cleaned_2, aes(x = customer_type, fill = customer_type)) +  
  geom_bar(position = "dodge", width = 0.6) +  
  scale_y_continuous(name = "Number of Rides", labels = function(x) format(x, big.mark =  
= ",", scientific = F)) +  
  theme_classic() +  
  labs(x = "Customer Type") +  
  coord_cartesian(ylim = c(100000, 2500000)) +  
  ggtitle("Total Rides per Customer Type") +  
  theme(plot.title = element_text(hjust = 0.5, vjust = 0.5, face = "bold", size = 20)) +  
  geom_text(aes(label = after_stat(count)), stat = "count", vjust = -0.5, color = "black", size = 5)
```

# Total Rides per Customer Type



Member riders took more than 2 million trips while casual riders took more than 1.95 million rides.

## Chi-square test for customer rides

I use chi-square test to examine the difference of these two groups frequencies.

```
chisq.test(table(bike_cleaned_2$customer_type))
```

```
##  
## Chi-squared test for given probabilities  
##  
## data:  table(bike_cleaned_2$customer_type)  
## X-squared = 42487, df = 1, p-value < 2.2e-16
```

Chi-square test result show that p-value is less than 0.05, means that there is a significant difference between members and casuals rides. There was evidence that at the 5% level, the hypothesis being tested was correct, it is that members use bikes more often than casuals. **The hypothesis H1 was supported.**

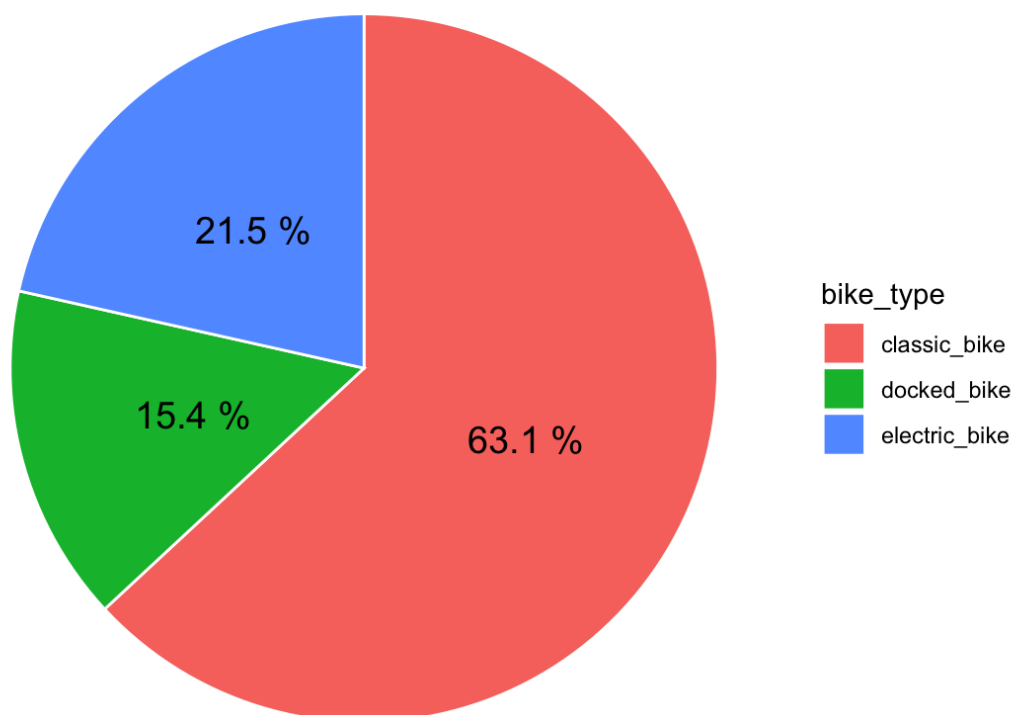
## Pie chart of rides by bike type

```

bike_cleaned_2 %>%
  group_by(bike_type) %>%
    summarise(bike_rides = n() %>%
      ggplot(aes(x = "", y = bike_rides, fill = bike_type +
        geom_bar(stat = "identity", width = 1, color = "white" +
        coord_polar("y", start = 0, direction = -1 +
        theme_void( +
        ggtitle("Rides Percent Per Bike Type" +
        theme(plot.title = element_text(hjust = 0.5, vjust = 0.5, face = "bold", size = 20
+
    geom_text(aes(label = paste(round(bike_rides / sum(bike_rides)*100, 1), "%")), position
on = position_stack(vjust = 0.5), size = 5)

```

## Rides Percent Per Bike Type



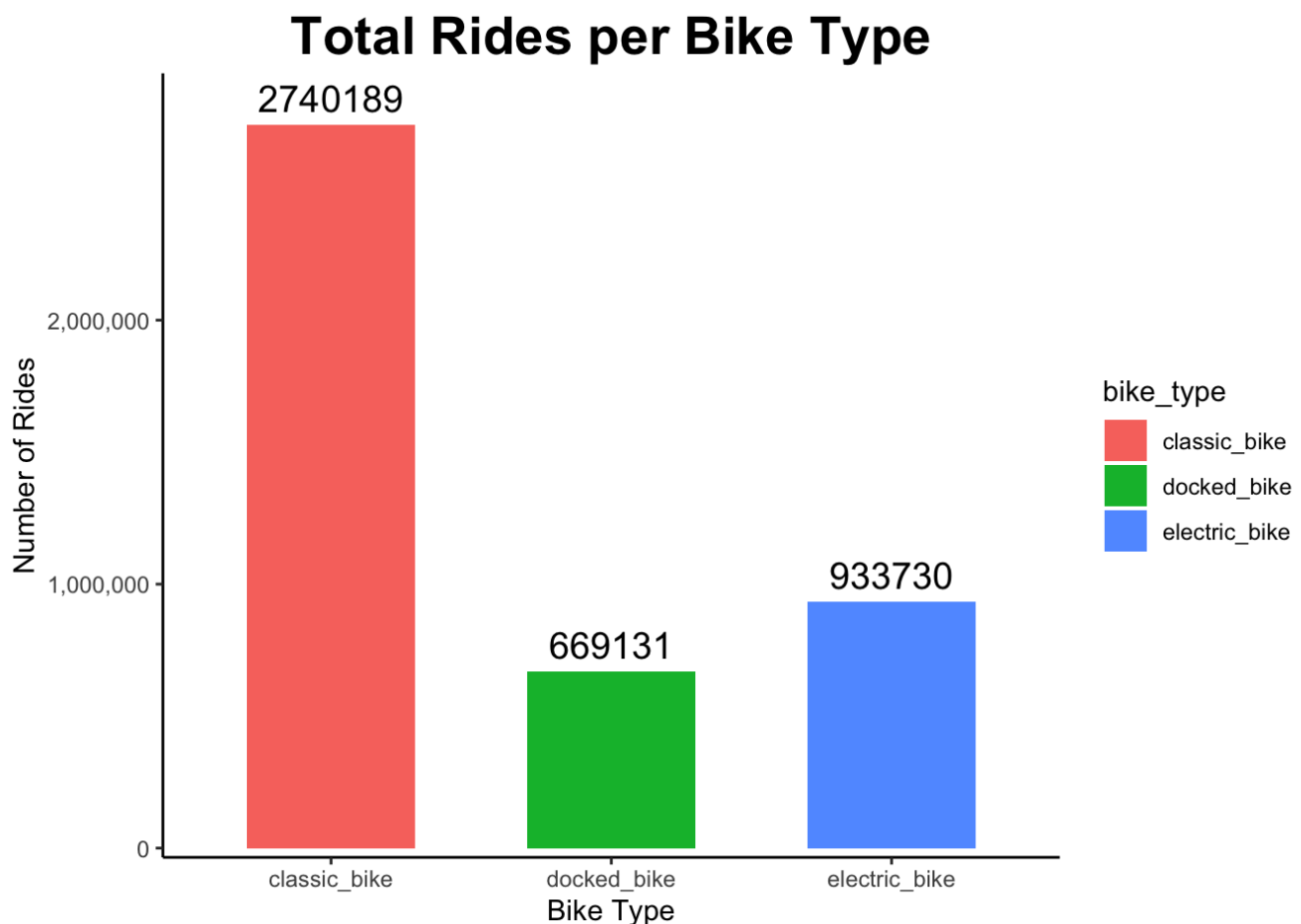
Based on the data I selected, classic bikes account for 63.1% of the total rides, electric bikes account for 21.5%, and docked bikes account for 15.4%. Since we lack sufficient information, it's challenging to determine definitively whether this distribution reflects customer preference or the company's strategy in launching more classic bikes than the other two options.

### Bar chart of rides by bike type

```
ggplot(bike_cleaned_2, aes(x= bike_type, fill = bike_type)) +
  geom_bar(position="dodge", width = 0.6

  scale_y_continuous(name = "Number of Rides", labels = function(x) format(x, big.mark
= ",", scientific = F)) +
  coord_cartesian(ylim = c(100000, 2800000)) +
  theme_classic() +
  labs(x = "Bike Type") +
  ggtitle("Total Rides per Bike Type") +
  theme(plot.title = element_text(hjust = 0.5, vjust = 0.5, face = "bold", size = 20))
+
  geom_text(aes(label = ..count..), stat = "count", vjust = -0.5, color = "black", size
= 5)
```

```
## Warning: The dot-dot notation (`..count..`) was deprecated in ggplot2 3.4.0.
## i Please use `after_stat(count)` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```



Based on the data I selected, classic bikes reach rides to 2,740,198, docked bikes hit 669,131 rides, and electric bikes hit 933,730 rides.

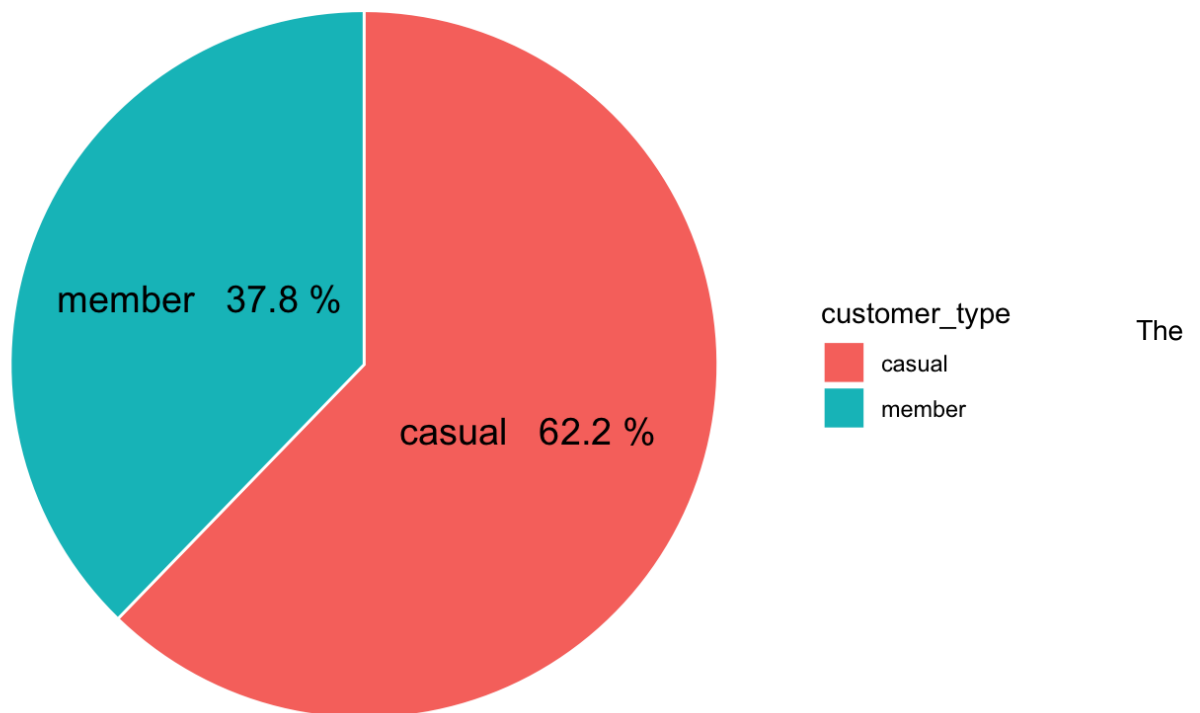
**Pie chart of ride duration by customer type**

```

bike_cleaned_2 %>%
  group_by(customer_type) %>%
  summarise(customer_sum = sum(ride_duration %>%
  ggplot(aes(x = "", y = customer_sum, fill = customer_type +
  geom_bar(stat = "identity", width = 1, color = "white" +
  coord_polar("y", start = 0, direction = -1 +
  theme_void( +
  ggtitle("Ride Duration Percent per Bike Type" +
  theme(plot.title = element_text(hjust = 0.5, vjust = 0.5, size =20, face = "bold" +
  geom_text(aes(label = paste(customer_type, " ", round(customer_sum / sum(customer_su
m)*100, 1), "%")), position = position_stack(vjust = 0.5),size = 5)

```

## Ride Duration Percent per Bike Type

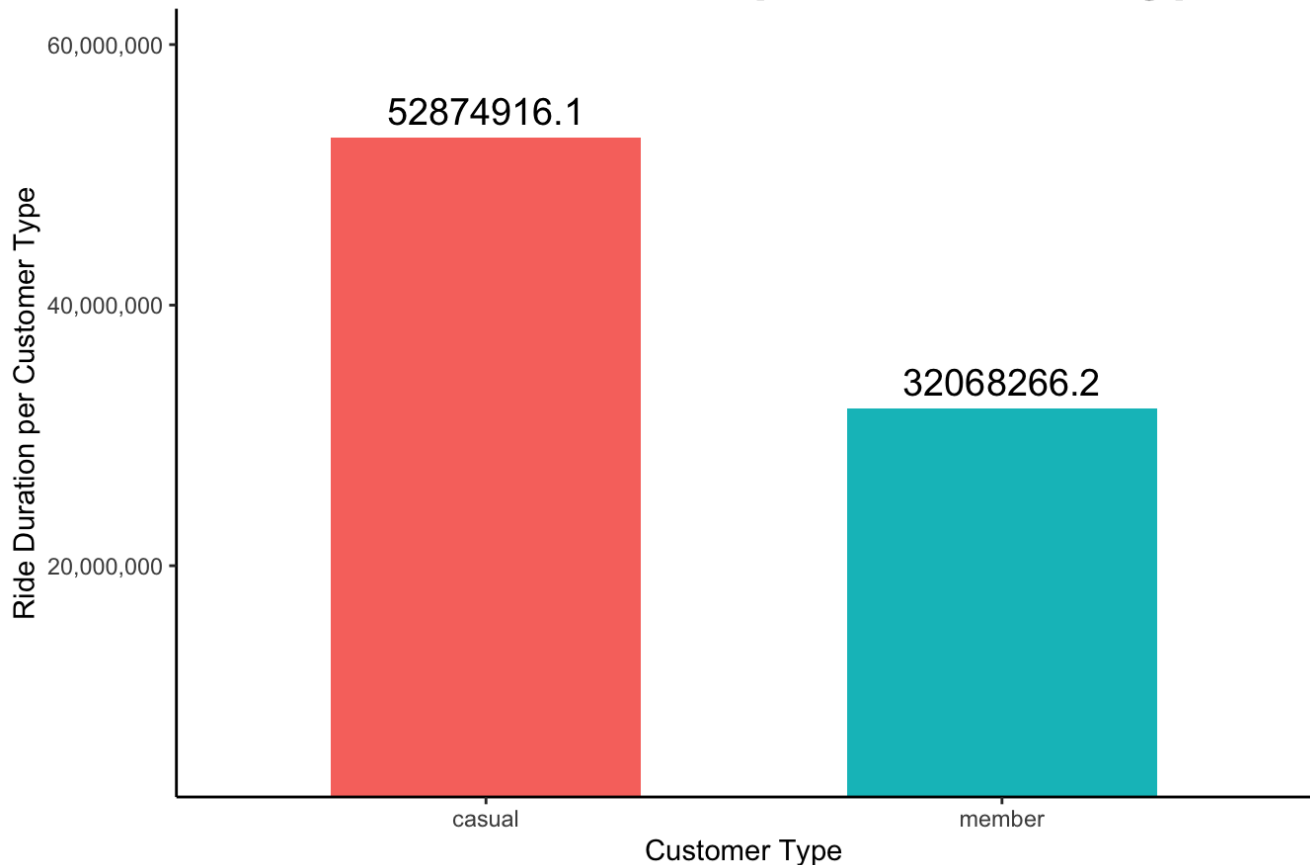


proportion of ride duration for casual users has dramatically increased to 62.2% compared to the proportion of rides (45.1%). This suggests that casual users tend to use bikes for longer duration per ride than members.

### Bar chart of ride duration by customer type

```
bike_cleaned_2 %>%
  group_by(customer_type) %>%
    summarise(customer_duration = round(sum(ride_duration, 1 %>%
      ggplot(aes(x = customer_type, y = customer_duration, fill = customer_type, label = cu
stomer_duration)) +
    geom_bar(stat = "identity", position = "dodge", width = 0.6) +
    scale_y_continuous(name = "Ride Duration per Customer Type", labels = function(x) for
mat(x, big.mark = ",", scientific = F)) +
    coord_cartesian(ylim = c(5000000, 60000000)) +
    theme_classic() +
    labs(x = "Customer Type") +
    ggtitle("Total Ride Duration per Customer Type") +
    theme(plot.title = element_text(hjust = 0.5, vjust = 0.5, face = "bold", size = 20))
+
    geom_text(aes(label = customer_duration), vjust = -0.5, color = "black", size = 5) +
    theme(legend.position = "none")
```

## Total Ride Duration per Customer Type



## H2 Ride duration difference by customer type

H2: Members' average ride duration is higher than casuals'.

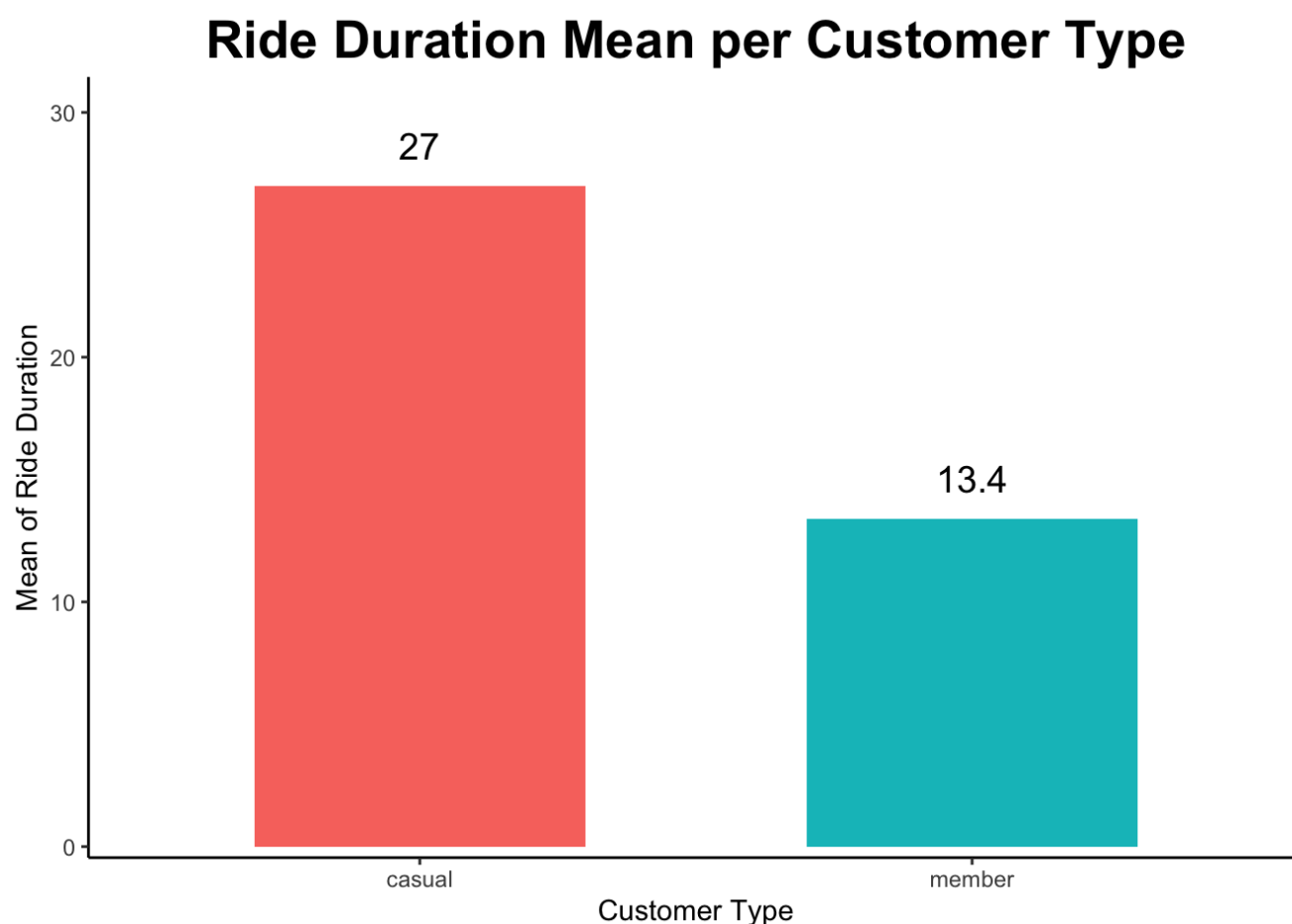
### Ride duration mean by customer type

```
aggregate(bike_cleaned_2, ride_duration~customer_type, FUN = mean)
```

```
## customer_type ride_duration
## 1 casual 27.02189
## 2 member 13.43846
```

## Bar chart of ride duration mean by customer type

```
bike_cleaned_2 %>%
  group_by(customer_type) %>%
  summarise(mean_duration_c = round(mean(ride_duration), 1)) %>%
  ggplot(aes(x = customer_type, y = mean_duration_c, fill = customer_type)) +
  geom_col(width = 0.6, position = "dodge") +
  theme_classic() +
  scale_y_continuous(name = "Mean of Ride Duration") +
  coord_cartesian(ylim = c(1, 30)) +
  ggtitle("Ride Duration Mean per Customer Type") +
  theme(plot.title = element_text(hjust = 0.5, vjust = 0.5, face = "bold", size = 20),
        legend.position = "none") +
  labs(x = "Customer Type") +
  geom_text(aes(label = mean_duration_c), vjust = -1, color = "black", size = 5)
```



## T test for ride duration by customer type

```
t.test(ride_duration~customer_type, data = bike_cleaned_2)
```

```
##
#### Welch Two Sample t-test
## data: ride_duration by customer_type
## t = 592.12, df = 2488536, p-value < 2.2e-16
## alternative hypothesis: true difference in means between group casual and group member is not equal to 0
## 95 percent confidence interval:
## 13.53847 13.62839
## sample estimates:
## mean in group casual mean in group member
## 27.02189 13.43846
```

Based on the very low p-value and the confidence interval that does not include 0, the t-test result indicates a highly significant difference in ride duration between the “casual” and “member” customer types. **H2: Members’ average ride duration is higher than casuals, was not supported. Casual users’ average ride duration is longer than members.**

## Median of ride duration by customer type

```
aggregate(bike_cleaned_2, ride_duration~customer_type, FUN = median)
```

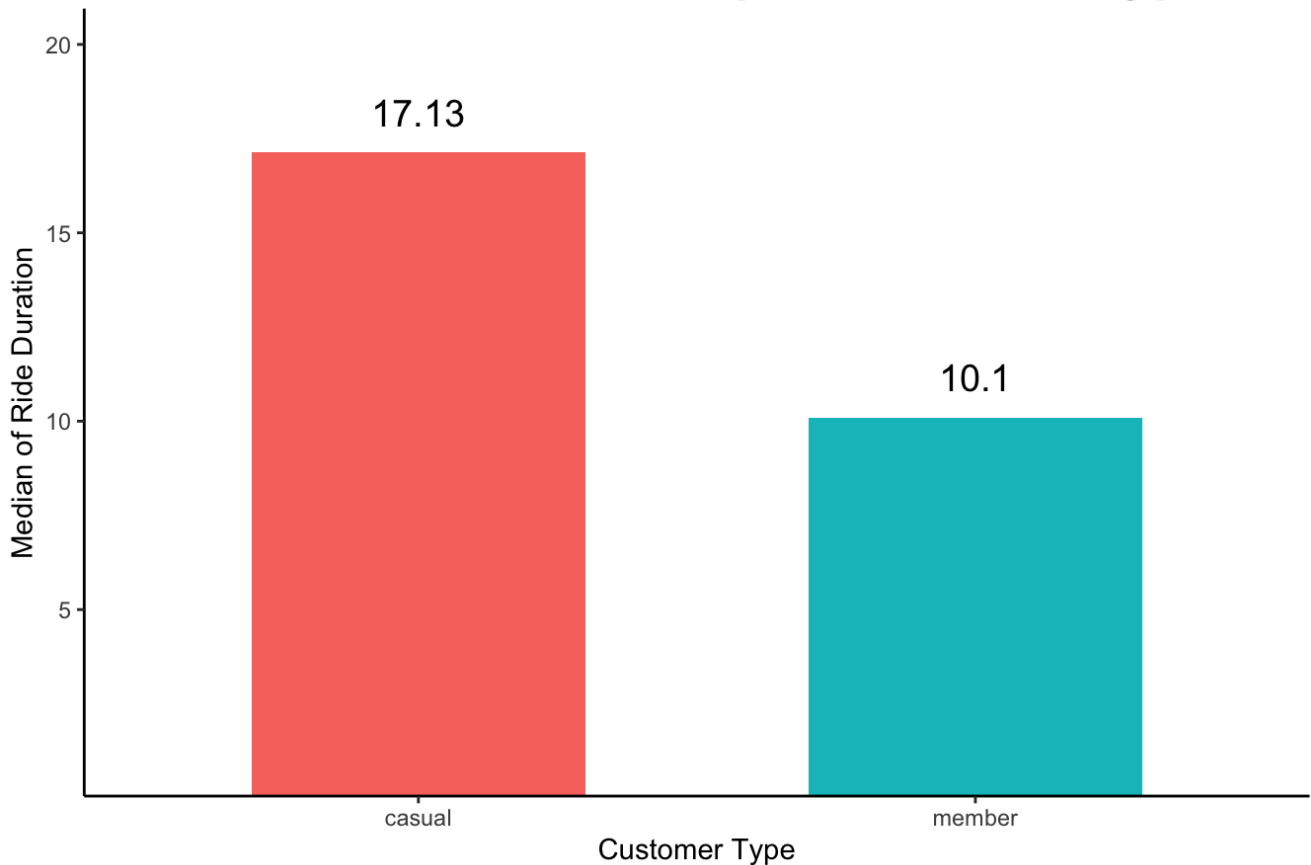
```
## customer_type ride_duration
## 1 casual 17.13
## 2 member 10.10
```

## Bar chart of ride duration median by customer type

```
bike_cleaned_2 %>%
  group_by(customer_type) %>%
  summarise(median_duration_c = median(ride_duration)) %>%
  ggplot(aes(x = customer_type, y = median_duration_c, fill = customer_type)) +
  theme_classic() +
  geom_col(width = 0.6, position = "dodge") +
  scale_y_continuous(name = "Median of Ride Duration") +
  coord_cartesian(ylim = c(1, 20)) +
  labs(x = "Customer Type") +
  ggtitle("Ride Duration Median per Customer Type") +
  theme(plot.title = element_text(hjust = 0.5, vjust = 0.5, face = "bold", size = 20),
  legend.position = "none") +
  geom_text(aes(label = median_duration_c), vjust = -1, color = "black", size = 5)
```



## Ride Duration Median per Customer Type



### Wilcox test for ride duration median by customer type

```
wilcox.test(bike_cleaned_2$ride_duration~bike_cleaned_2$customer_type)
```

```
##  
## Wilcoxon rank sum test with continuity correction  
##  
## data: bike_cleaned_2$ride_duration by bike_cleaned_2$customer_type  
## W = 3.1832e+12, p-value < 2.2e-16  
## alternative hypothesis: true location shift is not equal to 0
```

The Wilcoxon rank sum test result suggests strong evidence of a significant difference in the medians of ride duration between the two customer types (“casual” and “member”).

### maximum of ride duration per customer type

```
aggregate(bike_cleaned_2, ride_duration~customer_type, FUN = max)
```

```
## customer_type ride_duration  
## 1 casual 299.93  
## 2 member 299.85
```

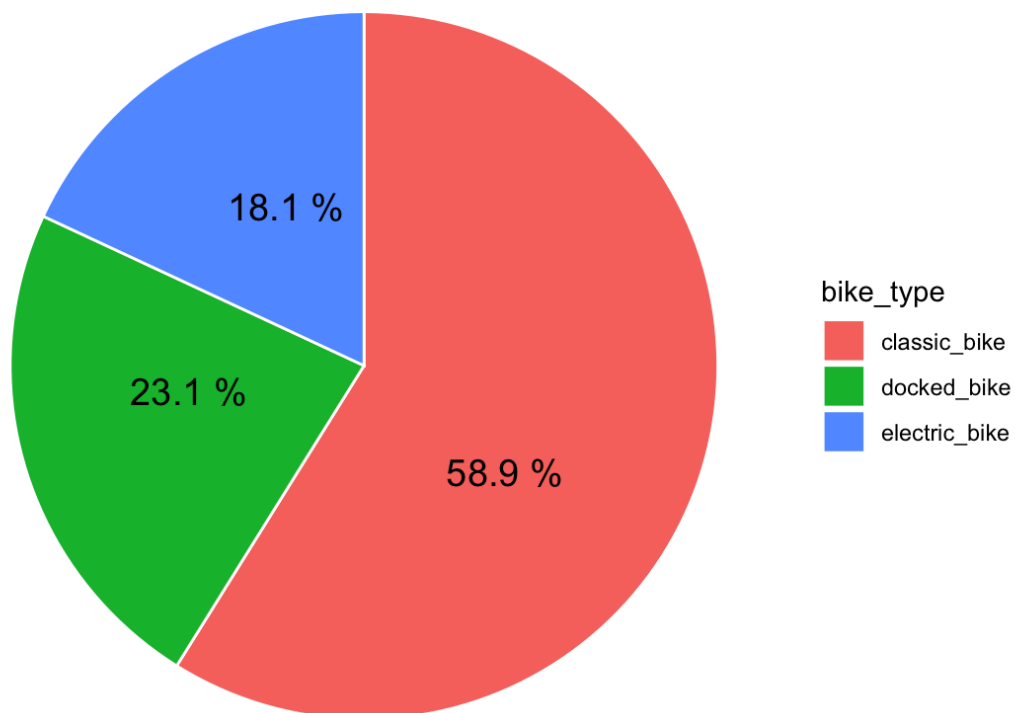
### Pie chart of ride duration by bike type

```
bike_sum_duration <- bike_cleaned_2 %>%  
  group_by(bike_type) %>%  
  summarise(sum_duration = sum(ride_duration))  
bike_sum_duration
```

```
## # A tibble: 3 × 2
##   bike_type      sum_duration
##   <chr>          <dbl>
## 1 classic_bike    50013721.
## 2 docked_bike    19583950.
## 3 electric_bike  15345511.
```

```
ggplot(bike_sum_duration, aes(x = "", y = sum_duration, fill = bike_type)) +
  geom_bar(stat = "identity", width = 1, color = "white") +
  coord_polar("y", start = 0, direction = -1) +
  theme_void() +
  theme(axis.title = element_blank(), axis.ticks = element_blank(), plot.title = element_text(hjust = 1)) +
  ggtitle("Ride Duration Percent per Bike Type") +
  theme(plot.title = element_text(hjust = 0.5, vjust = 0.5, face = "bold", size = 20)) +
  geom_text(aes(label = paste(round(sum_duration / sum(sum_duration)*100, 1), "%")), position = position_stack(vjust = 0.4), size = 5)
```

## Ride Duration Percent per Bike Type



The classic bike account for 58.9% of the total ride duration, docked bike account for 23.1%, and electric bike account for 18.1%.

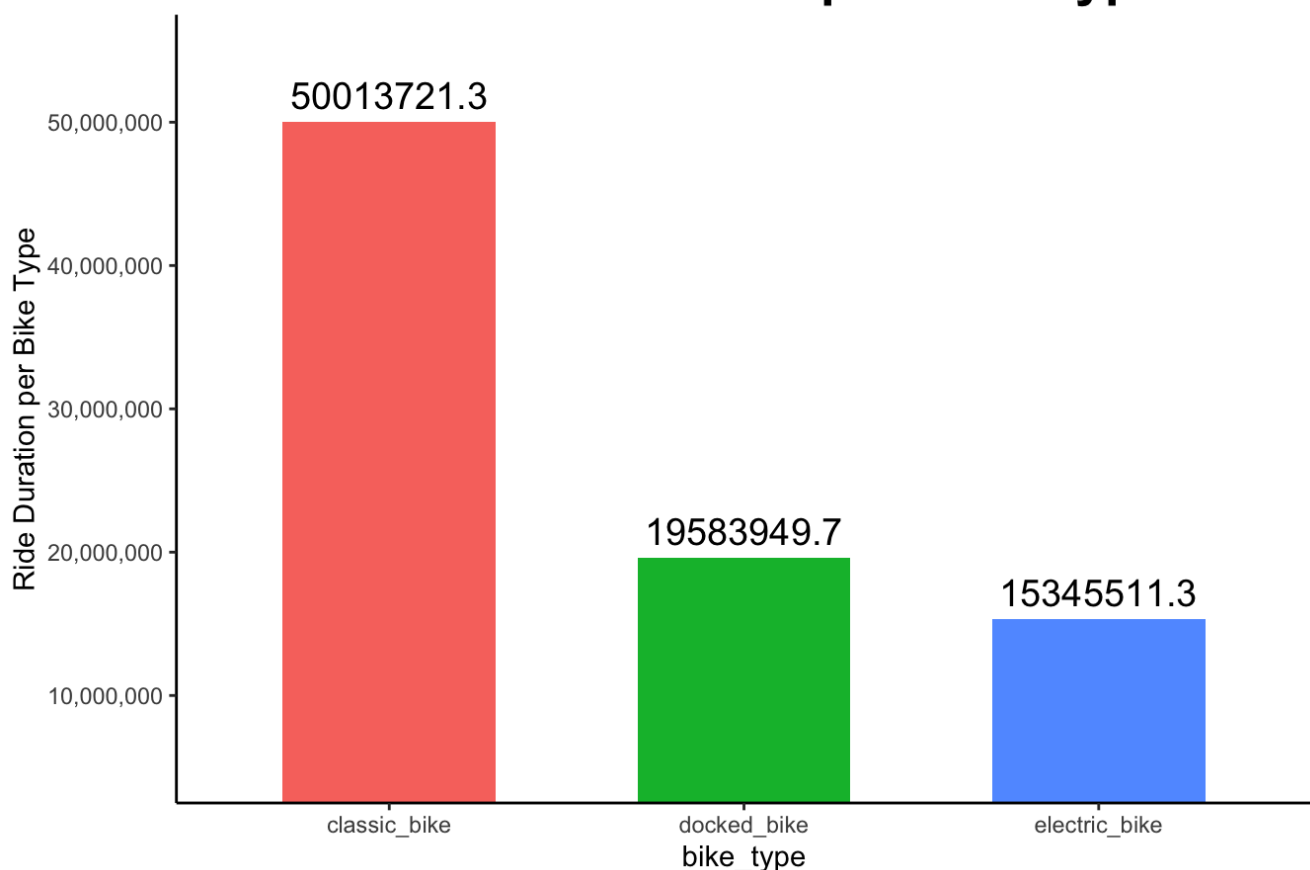
**Bar chart of ride duration by bike type**

```

bike_cleaned_2 %>%
  group_by(bike_type) %>%
    summarise(bike_type_duration= round(sum(ride_duration), 1)) %>%
      ggplot(aes(x = bike_type, y = bike_type_duration, fill = bike_type, label = bike_type
_duration)) +
        geom_bar(stat = "identity", position = "dodge", width = 0.6) +
        scale_y_continuous(name = "Ride Duration per Bike Type", labels = function(x) format
(x, big.mark = ",", scientific = F)) +
        coord_cartesian(ylim = c(5000000, 55000000)) +
        theme_classic() +
        ggtitle("Total Ride Duration per Bike Type") +
        theme(plot.title = element_text(hjust = 0.5, vjust = 0.5, face= "bold", size = 20), 1
legend.position = "none") +
        geom_text(vjust = -0.5, color = "black", size = 5)

```

## Total Ride Duration per Bike Type



Based on the data, it's evident that classic bikes have the highest total ride duration, surpassing 50 million minutes. Docked bikes have a ride duration of around 19 million minutes, while electric bikes have achieved over 15 million minutes of ride duration. However, we can't definitively determine which bike type is the most popular, as the ride duration is influenced by the number of bikes the company introduced in the area. Utilizing the ride duration mean could potentially serve as a more reliable indicator of bike popularity.

## Mean of ride duration by bike type

```

aggregate(bike_cleaned_2, ride_duration~bike_type, FUN = mean)

```

```

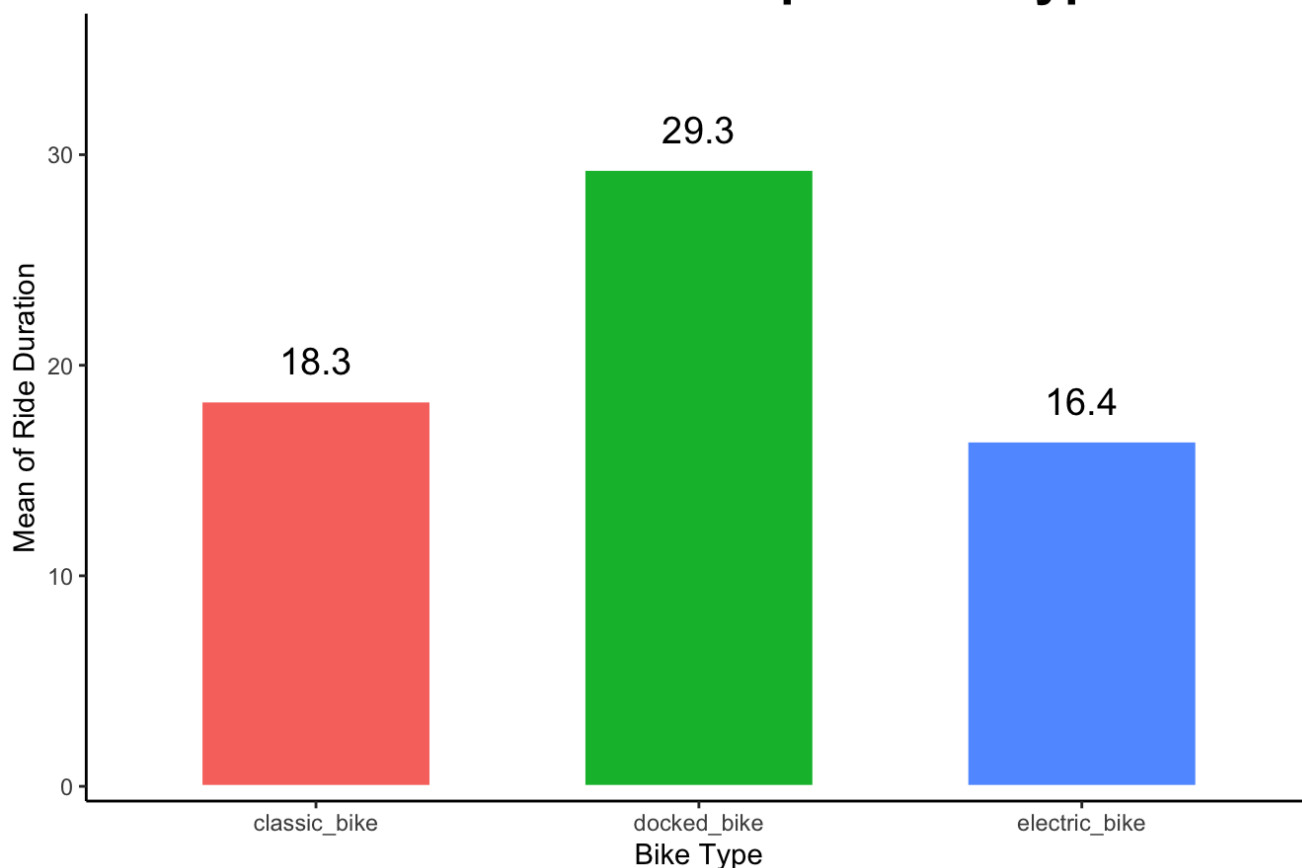
##      bike_type ride_duration
## 1 classic_bike      18.25192
## 2 docked_bike      29.26774
## 3 electric_bike     16.43463

```

## Bar chart of ride duration mean by bike type

```
bike_cleaned_2 %>%
  group_by(bike_type) %>%
  summarise(mean_duration = round(mean(ride_duration), 1)) %>%
  ggplot(aes(x = bike_type, y = mean_duration, fill = bike_type)) +
  geom_col(position = "dodge", width = 0.6, color = "white") +
  theme_classic() +
  coord_cartesian(ylim = c(1, 35)) +
  scale_y_continuous(name = "Mean of Ride Duration") +
  ggtitle("Ride Duration Mean per Bike Type") +
  theme(plot.title = element_text(hjust = 0.5, vjust = 0.5, face = "bold", size = 20))
+
  labs(x = "Bike Type") +
  geom_text(aes(label = mean_duration), vjust = -1, color = "black", size = 5) +
  theme(legend.position = "none")
```

### Ride Duration Mean per Bike Type



## Levene test for variance equality

```
#levene_test_result <- leveneTest(ride_duration ~ bike_type, data = bike_cleaned_2)
#levene_test_result
```

The result shows that the variances are not equal between bike types.

## ANOVA analysis for ride duration by bike type

```
duration_aov <- aov(ride_duration~bike_type, bike_cleaned_2)
summary(duration_aov)
```

```
##              Df      Sum Sq  Mean Sq F value Pr(>F)
## bike_type      2 7.687e+07 38434108   74048 <2e-16 ***
## Residuals 4343047 2.254e+09      519
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The p-value associated with the “bike\_type” variable is remarkably low ( $p < 0.001$ ), indicating a significant impact of bike type on ride duration. This finding suggests that docked bikes are the most popular choice. Further analysis using the TukeyHSD function reveals a significant difference between electric and classic bikes, leading to the conclusion that the classic bike stands as the second most favored option.

## Post-hoc test of ANOVA analysis for ride duration by bike type

```
tukey_aov<-TukeyHSD(duration_aov)
tukey_aov
```

```
##      Tukey multiple comparisons of means
##      95% family-wise confidence level
##
## Fit: aov(formula = ride_duration ~ bike_type, data = bike_cleaned_2)
##
## $bike_type
##              diff          lwr          upr p adj
## docked_bike-classic_bike  11.015812  10.943002  11.088623    0
## electric_bike-classic_bike -1.817289  -1.881273  -1.753306    0
## electric_bike-docked_bike -12.833102 -12.918625 -12.747578    0
```

The result shows that there are significant difference between pairwise bike types.

## Median of ride duration per bike type

```
aggregate(bike_cleaned_2, ride_duration~bike_type, FUN = median)
```

```
##      bike_type ride_duration
## 1  classic_bike          12.42
## 2  docked_bike          17.95
## 3  electric_bike          11.12
```

The results reveal that docked bikes have the highest median ride duration, while electric bikes have the lowest median ride duration.

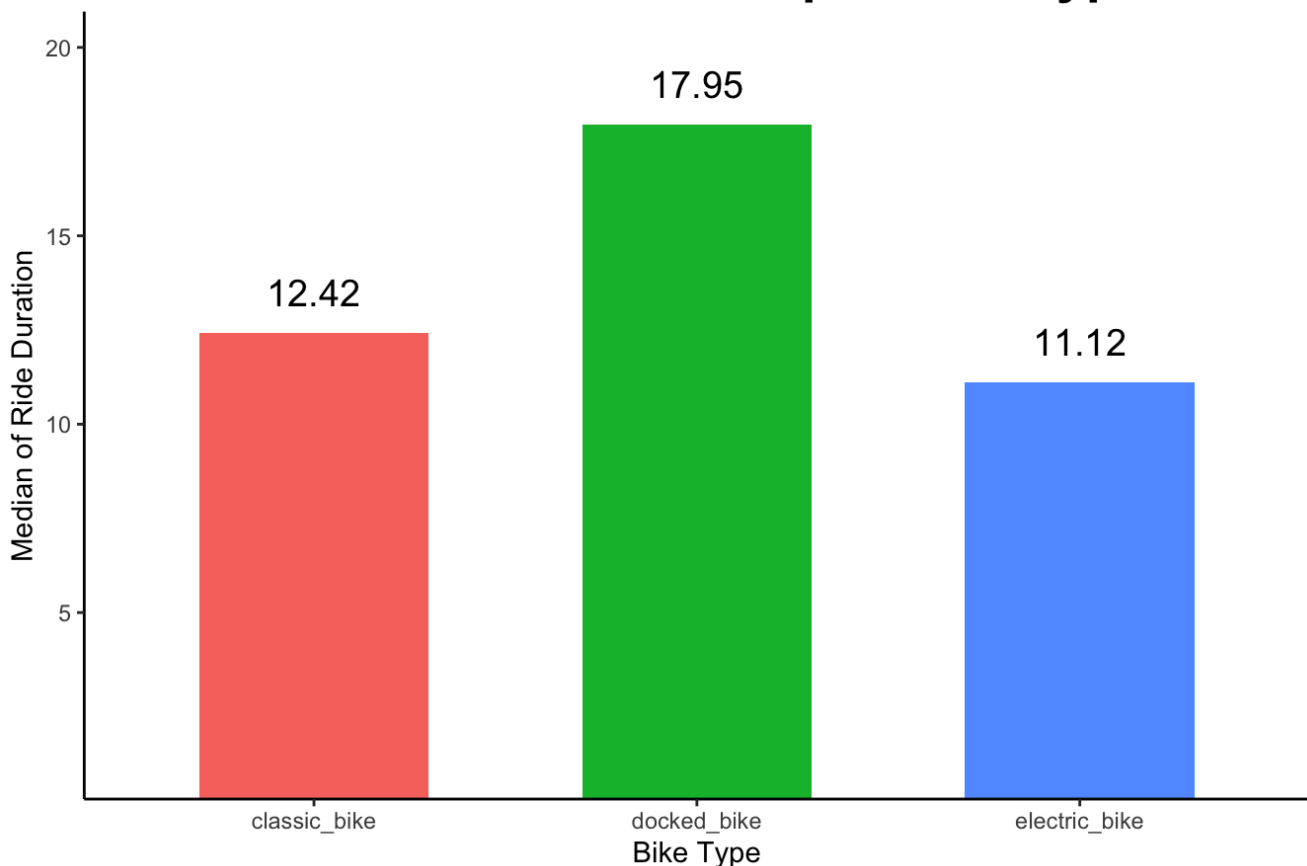
## Bar chart of ride duration median per bike type

```

bike_cleaned_2 %>%
  group_by(bike_type) %>%
    summarise(median_duration_b = median(ride_duration)) %>%
    ggplot(aes(x = bike_type, y = median_duration_b, fill = bike_type, label = median_dur
ation_b)) +
    geom_col(width = 0.6, position = "dodge") +
    labs(y = "Median of Ride Duration", x = "Bike Type") +
    coord_cartesian(ylim = c(1, 20)) +
    theme_classic() +
    ggtitle("Ride Duration Median per Bike Type") +
    theme(plot.title = element_text(hjust = 0.5, vjust = 0.5, face = "bold", size = 20),
    legend.position = "none") +
    geom_text(vjust = -1, color = "black", size = 5)

```

## Ride Duration Median per Bike Type



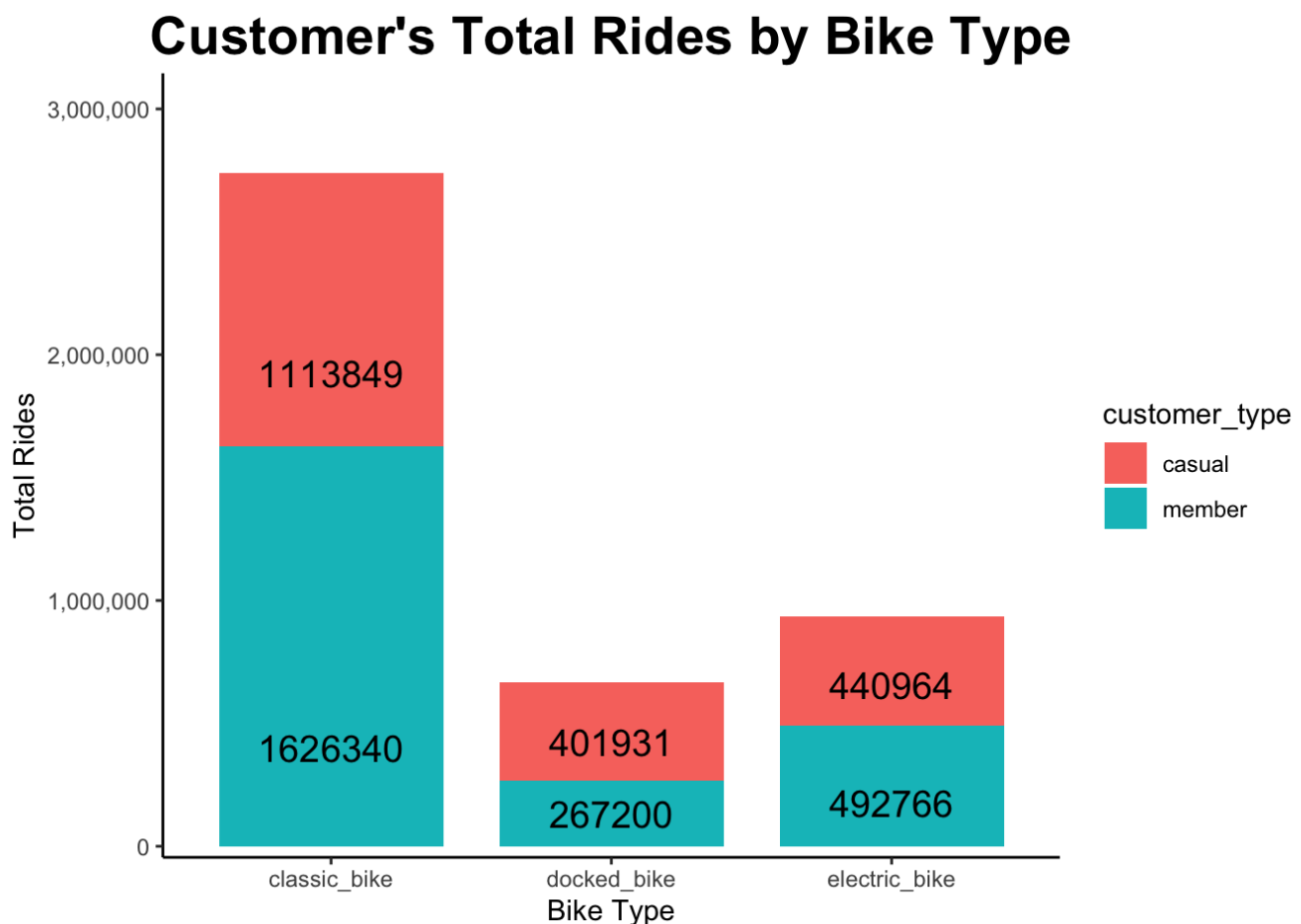
### H3 Customer's rides difference on bike type

H3: Member and casuals are significantly different in bike type choice.

Total rides comparison between customer types based on bike type

```
bike_cleaned_2 %>%
  group_by(bike_type, customer_type) %>%
  summarise(total_rides = n()) %>%
  ggplot(aes(fill = customer_type, x = bike_type, y = total_rides)) +
  geom_col(width = 0.8, position = "stack") +
  theme_classic() +
  scale_y_continuous(name = "Total Rides", labels = function(x) format(x, big.mark =
",", scientific = F)) +
  coord_cartesian(ylim = c(100000, 3000000)) +
  ggtitle("Customer's Total Rides by Bike Type") +
  theme(plot.title = element_text(hjust = 0.5, vjust = 0.5, face = "bold", size = 20))
+
  labs(x = "Bike Type") +
  geom_text(aes(label = total_rides, group = customer_type), position = position_stack
(vjust = 0.2, reverse = F), vjust = -0.2, size = 5)
```

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



The results suggest that member users exhibit the highest ride frequency with classic bikes compared to all other segments, while also demonstrating the lowest ride frequency on docked bikes among all segments. In contrast, casual users show a higher ride frequency with classic bikes, although it remains lower than that of member users with classic bikes. Additionally, casual users have the lowest ride frequency with docked bikes, yet this frequency is higher than that of member users with docked bikes.

### Chi-square test for customer rides

```
chisq.test(table(bike_cleaned_2$customer_type, bike_cleaned_2$bike_type))
```

```
##
## Pearson's Chi-squared test
##
## data:  table(bike_cleaned_2$customer_type, bike_cleaned_2$bike_type)
## X-squared = 84189, df = 2, p-value < 2.2e-16
```

The results of the chi-square test indicate that the p-value is less than 0.05, signifying a significant distinction between bike types preferred by members and casual users. Specifically, member riders tend to favor classic and electric bikes more than casual users. Additionally, it's noteworthy that casual users show a preference for docked bikes compared to members. This evidence supports the hypothesis **H3**, which suggests that there are significant differences in bike type choices between members and casuals.

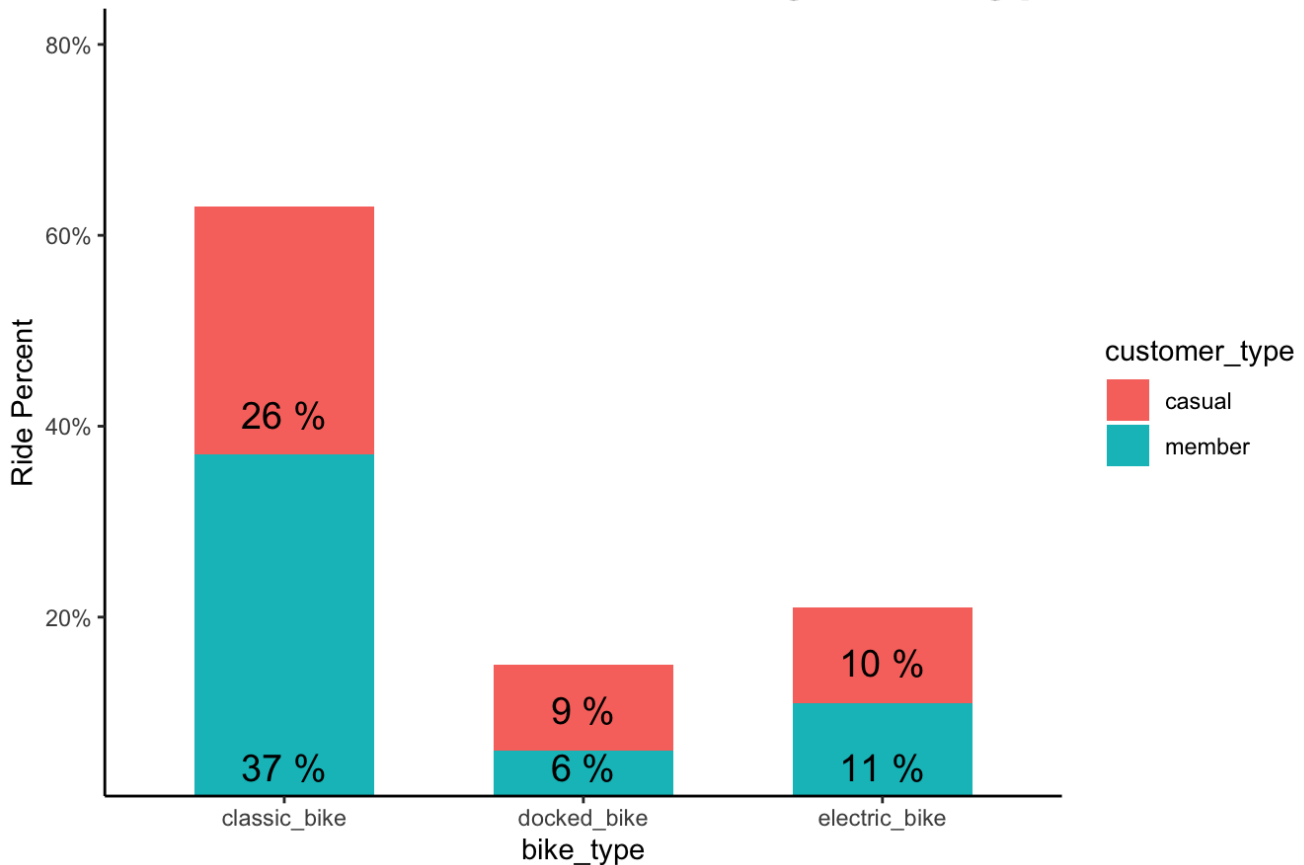
## Customers' rides percent by bike type

```
bike_cleaned_2 %>%
  group_by(bike_type, customer_type) %>%
  summarise(ride_percent = round(n()/nrow(bike_cleaned_2), 2), "%") %>%
  ggplot(aes(x = bike_type, y = ride_percent, fill = customer_type)) +
  geom_col(width = 0.6, position = "stack") +
  scale_y_continuous(name = "Ride Percent", labels = scales::percent) +
  coord_cartesian(ylim = c(0.05, 0.8)) +
  theme_classic() +
  ggtitle("Customer's Rides Percent by Bike Type") +
  theme(plot.title = element_text(hjust = 0.5, vjust = 0.5, face = "bold", size = 20))
+
  geom_text(aes(label = paste(ride_percent*100, "%"), group = customer_type), position
= position_stack(vjust = 0, reverse = F), vjust = -1, color = "black", size = 5)
```

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



# Customer's Rides Percent by Bike Type



## Ride duration mean for customer types based on bike types

```
aggregate(bike_cleaned_2, ride_duration~bike_type + customer_type, FUN = mean)
```

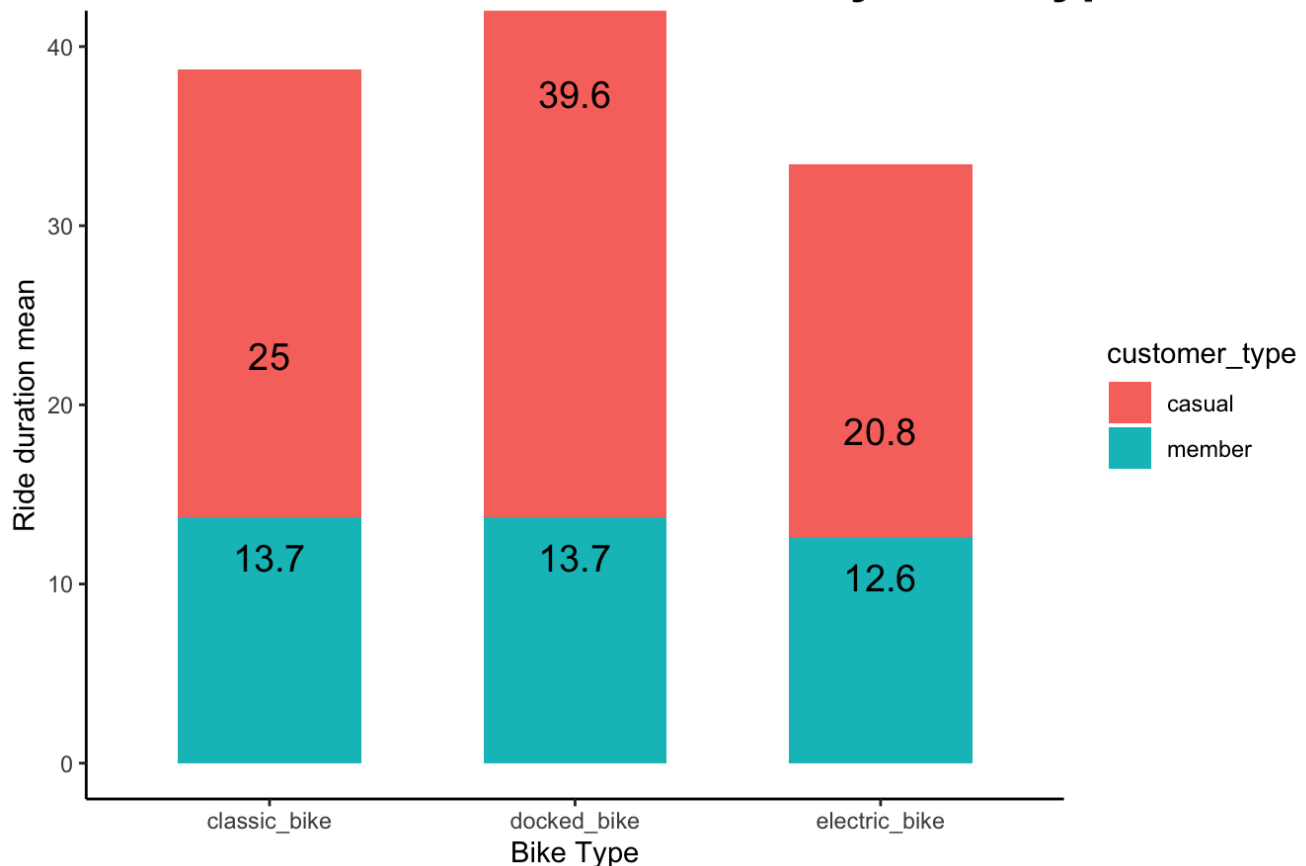
```
##      bike_type customer_type ride_duration
## 1 classic_bike      casual      24.96029
## 2 docked_bike      casual      39.61202
## 3 electric_bike     casual      20.75367
## 4 classic_bike      member      13.65749
## 5 docked_bike      member      13.70753
## 6 electric_bike     member      12.56964
```

## Bar chart of ride duration mean for customer types based on bike types

```
bike_cleaned_2 %>%
  group_by(bike_type, customer_type) %>%
  summarise(mean_duration_b = round(mean(ride_duration), 1)) %>%
  ggplot(aes(x = bike_type, y = mean_duration_b, fill = customer_type)) +
  geom_col(width = 0.6, position = "stack") +
  scale_y_continuous(name = "Ride duration mean") +
  coord_cartesian(ylim = c(0, 40)) +
  labs(x = "Bike Type") +
  theme_classic() +
  ggtitle("Customer's Ride Duration Mean by Bike Type") +
  theme(plot.title = element_text(hjust = 0.5, vjust = 0.5, face="bold", size = 20)) +
  geom_text(aes(label = mean_duration_b), vjust = 2, size = 5)
```

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

## Customer's Ride Duration Mean by Bike Type



The findings reveal that members exhibit comparable average ride durations for classic and docked bikes, with slightly shorter durations for electric bikes. Furthermore, the mean ride durations for all bike types are consistently lower among members in comparison to casual users.

### T test for ride duration by customer type based on bike types

```
mult_test <- bike_cleaned_2 %>%
  group_by(bike_type) %>%
  group_modify(
    ~ t.test(ride_duration ~ customer_type, data = .x, var.equal = TRUE) %>%
      broom::tidy()
  )
mult_test
```

```
## # A tibble: 3 × 11
## # Groups:   bike_type [3]
##   bike_type      estimate estimate1 estimate2 statistic p.value parameter conf.low
##   <chr>          <dbl>     <dbl>     <dbl>     <dbl>   <dbl>     <dbl>     <dbl>
## 1 classic_bike    11.3        25.0        13.7      466.     0    2740187    11.3
## 2 docked_bike     25.9        39.6        13.7      323.     0     669129    25.7
## 3 electric_bi...   8.18        20.8        12.6      225.     0     933728     8.11
## # i 3 more variables: conf.high <dbl>, method <chr>, alternative <chr>
```

I use another method to compare the mean difference between member and casual users.

```
bike_cleaned_2 %>%
  group_by(bike_type) %>%
  rstatix::t_test(ride_duration ~ customer_type, var.equal = F)
```

```
## # A tibble: 3 × 9
##   bike_type      .y.      group1 group2      n1      n2 statistic      df      p
## * <chr>      <chr>      <chr> <chr>    <int> <int>    <dbl> <dbl> <dbl>
## 1 classic_bike ride_duration casual member 1.11e6 1.63e6    411. 1.42e6      0
## 2 docked_bike  ride_duration casual member 4.02e5 2.67e5    382. 5.09e5      0
## 3 electric_bike ride_duration casual member 4.41e5 4.93e5    219. 6.89e5      0
```

The results of the t-test reveal a noteworthy disparity in means between members and casuals for each bike type. This observation suggests that, irrespective of the specific bike type, casual users generally exhibit longer ride duration compared to members. Given this compelling finding, Cyclistic could benefit from revisiting their service fee policy. Implementing a service fee structure based on ride duration might be a strategic approach to enhance revenue generation. Such a policy adjustment aligns with the observed usage patterns leading to increased revenue for the company.

## Changing levels of day of week

As my data is from October of 2020 to September to 2021, so I arrange months levels from October as a start.

```
bike_cleaned_2$started_month <- factor(bike_cleaned_2$started_month, levels = c("Oct",
"Nov", "Dec", "Jan", "Feb", "Mar", "Apr", "May", "Jun", "Jul", "Aug", "Sep"))
```

## Calculating year-month column

```
bike_cleaned_2$started_y_month <- format(bike_cleaned_2$started_at, "%Y-%m")
```

# H4 & H5 Rides difference due to seasons

**H4: Due to the weather in Chicago, both members and casuals prefer to use bikes in summer than winter.**

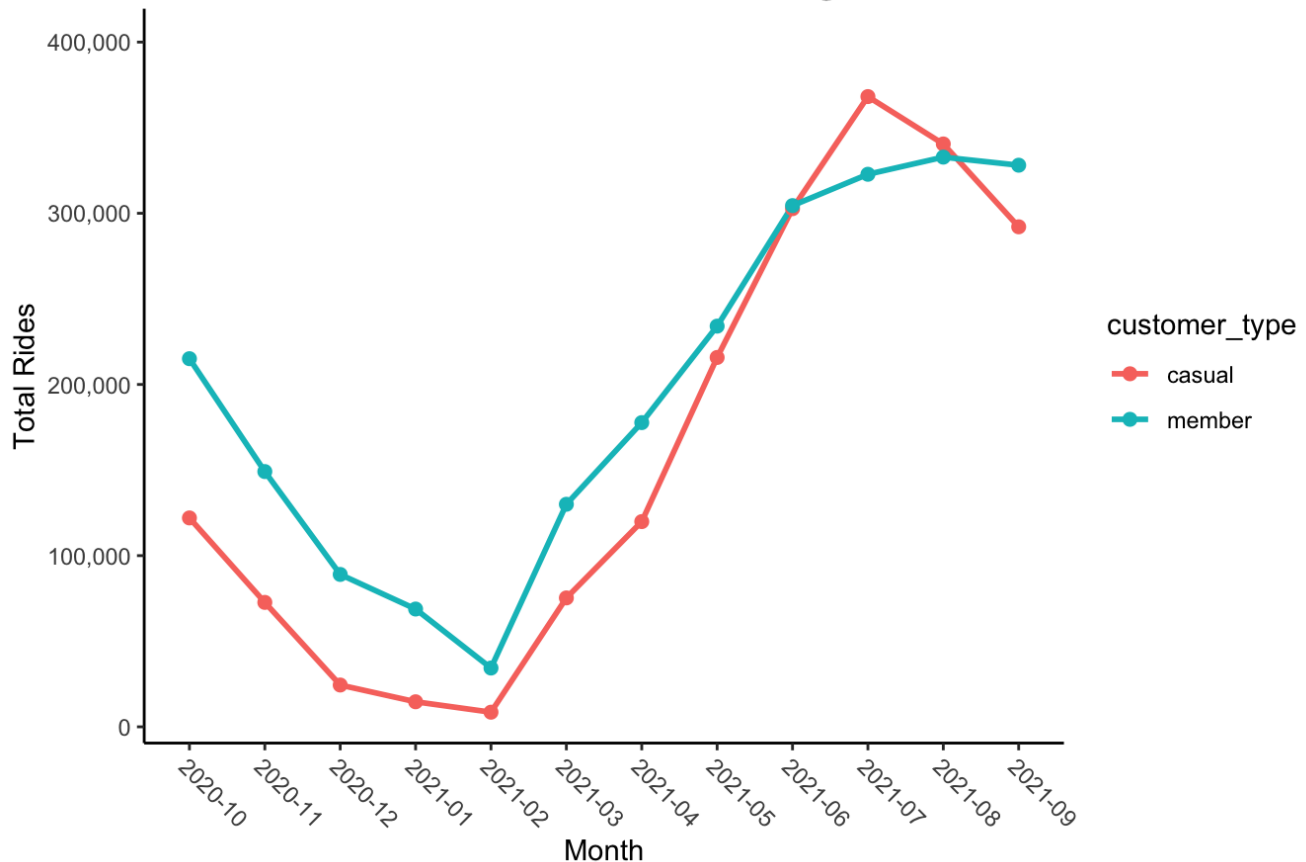
**H5: Members' total rides per month are higher than casuals.**

## Line chart of monthly total rides by customer types

```
bike_cleaned_2 %>%
  group_by(started_y_month, customer_type) %>%
  summarise(count = n()) %>%
  ggplot(aes(x = started_y_month, y = count, color = customer_type, group = customer_type)) +
  geom_line(linewidth = 1) +
  geom_point(size = 2) +
  scale_y_continuous(name = "Total Rides", labels = function(x) format(x, big.mark =
",", scientific = F)) +
  coord_cartesian(ylim = c(10000, 400000)) +
  labs(x = "Month") +
  theme_classic() +
  ggtitle("Customer's Total Rides by Month") +
  theme(plot.title = element_text(hjust = 0.5, vjust = 0.5, face = "bold", size = 20),
axis.text.x = element_text(angle = -45, hjust = 0))
```

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

## Customer's Total Rides by Month



## Chi-square test for monthly total rides

```
chisq.test(table(bike_cleaned_2$started_y_month, bike_cleaned_2$customer_type))
```

```
##
##  Pearson's Chi-squared test
##
## data:  table(bike_cleaned_2$started_y_month, bike_cleaned_2$customer_type)
## X-squared = 130000, df = 11, p-value < 2.2e-16
```

Chi-square test result show that p-value is less than 0.05, means that there is a significant difference between members and casuals rides. There was evidence that at the 5% level, months is a factor which can impact total rides significantly.

The data suggests a preference for biking during the months of May through September, with lower activity observed in January and February. Casual riders demonstrated their peak usage in July, while members reached their highest point in August. It's worth noting that member ridership consistently exceeded that of casual riders except in July and August. The fourth hypothesis, H4: **due to the weather in Chicago, both members and casuals prefer to use bikes in summer rather than winter**, finds support in the data. Consequently, launching marketing campaigns in spring to encourage bike usage would be advantageous for Cyclistic Company.

However, the fifth hypothesis, H5: **Members' total rides per month are higher than casuals**, is not supported by the data. While the overall trend shows casual users surpassing members in July and August, the differences in May, June, and September are relatively small, suggests that Cyclistic Company should not solely focus on

member users. Implementing campaigns to encourage both casual and member ridership could potentially yield greater benefits for the company.

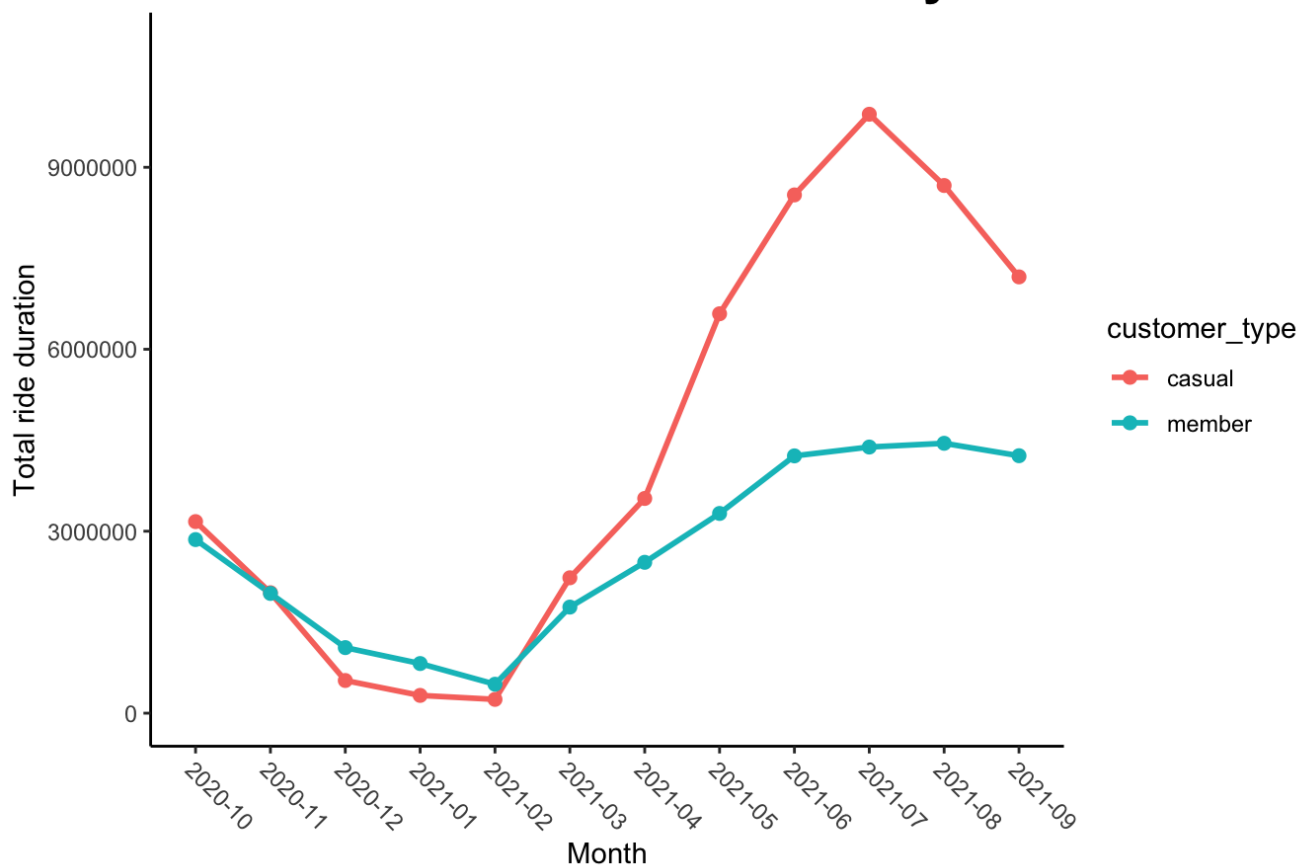
## H6 Monthly ride duration comparison of customer types

**H6: Members' total ride duration per month is higher than casuals.**

```
bike_cleaned_2 %>%
  group_by(started_y_month, customer_type) %>%
  summarise(total_duration_m = sum(ride_duration)) %>%
  ggplot(aes(x = started_y_month, y = total_duration_m, color = customer_type, group =
customer_type)) +
  geom_line(linewidth = 1) +
  geom_point(size = 2) +
  scale_y_continuous(name = "Total ride duration", labels = function(x) format(x, bigm
ark = ",", scientific = F)) +
  coord_cartesian(ylim = c(5000, 11000000)) +
  labs(x = "Month") +
  theme_classic()+
  ggtitle("Customer's Total Ride Duration by Month") +
  theme(plot.title = element_text(hjust = 0.5, vjust = 0.5, face = "bold", size = 20),
axis.text.x = element_text(angle = -45, hjust = 0))
```

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

### Customer's Total Ride Duration by Month



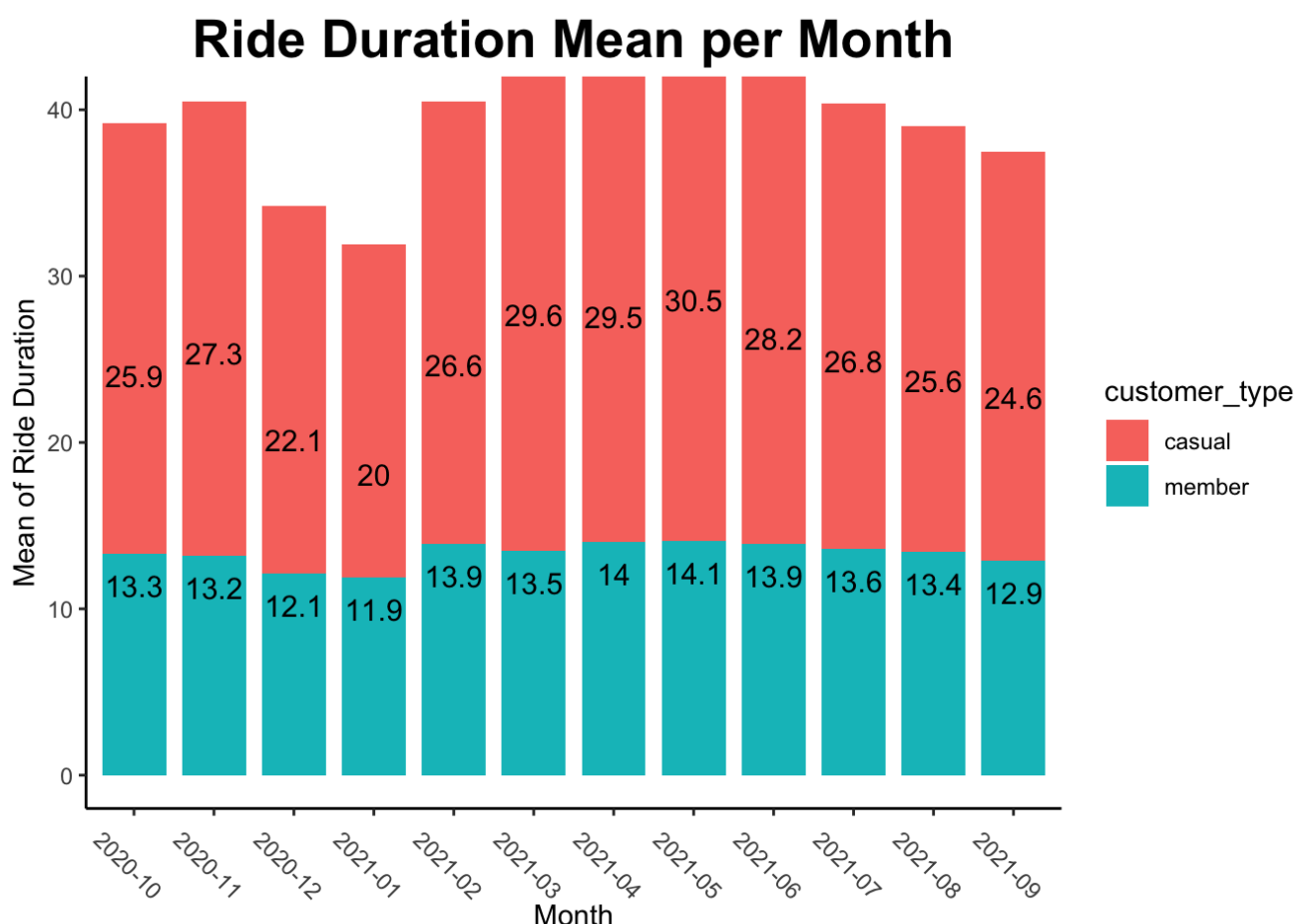
By the chart above, it's evident that casual users have higher total ride duration throughout the year, with the exception of November, December, January, and February. This pattern suggests that, apart from the winter months, casual riders maintain longer ride duration compared to members during other seasons. The sixth

hypothesis, which posited that members' total ride duration per month is greater than that of casuals, did not find support in the data. This reinforces the need for Capital Bikesharing aimed at expanding the base of casual users.

## Monthly ride duration mean across customer types

```
bike_cleaned_2 %>%
  group_by(started_y_month, customer_type) %>%
  summarise(mean_duration_m = round(mean(ride_duration), 1)) %>%
  ggplot(aes(x = started_y_month, y = mean_duration_m, fill = customer_type)) +
  geom_col(width = 0.8) +
  labs(x = "Month", y = "Mean of Ride Duration") +
  coord_cartesian(ylim = c(0, 40)) +
  theme_classic() +
  ggtitle("Ride Duration Mean per Month") +
  theme(plot.title = element_text(hjust = 0.5, vjust = 0.5, face = "bold", size = 20),
        axis.text.x = element_text(angle = -45)) +
  geom_text(aes(label = mean_duration_m), vjust = 2, size = 4)
```

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



The chart above clearly illustrates that casual users consistently have a longer mean ride duration compared to members throughout the entire year.

## Monthly ride duration T test accross customer type

```
bike_cleaned_2 %>%
  group_by(started_y_month) %>%
  rstatix::t_test(ride_duration ~ customer_type)
```

```
## # A tibble: 12 × 9
##   started_y_month .y.   group1 group2    n1    n2 statistic    df      p
##   * <chr>         <chr> <chr> <chr>   <int> <int>   <dbl> <dbl> <dbl>
## 1 2020-10        ride_... casual member 122030 215059    146.  1.47e5  0
## 2 2020-11        ride_... casual member  72685 149122    121.  8.41e4  0
## 3 2020-12        ride_... casual member  24417  89023    58.9  2.68e4  0
## 4 2021-01        ride_... casual member  14632  68804    40.4  1.60e4  0
## 5 2021-02        ride_... casual member   8546  34343    36.8  9.45e3 1.73e-277
## 6 2021-03        ride_... casual member  75307 130005    137.  8.87e4  0
## 7 2021-04        ride_... casual member 119880 177713    160.  1.48e5  0
## 8 2021-05        ride_... casual member 215773 234071    219.  2.75e5  0
## 9 2021-06        ride_... casual member 302704 304485    233.  3.97e5  0
## 10 2021-07        ride_... casual member 368172 322789    244.  4.99e5  0
## 11 2021-08        ride_... casual member 340498 332806    227.  4.63e5  0
## 12 2021-09        ride_... casual member 292100 328086    210.  3.85e5  0
```

The results of the t-test reveal a noteworthy disparity in means between members and casuals for each month. This observation suggests that, irrespective of the specific month, casual users generally exhibit longer ride duration compared to members. This indicates that casual riders tend to use bikes for longer periods than members across all of the year. Members exhibit greater ride duration stability throughout the year with minimal fluctuations. Casual users experience their shortest ride duration on January (20 minutes) and their longest on March (29.6 minutes).

## H7 Day of week rides comparison of customer types

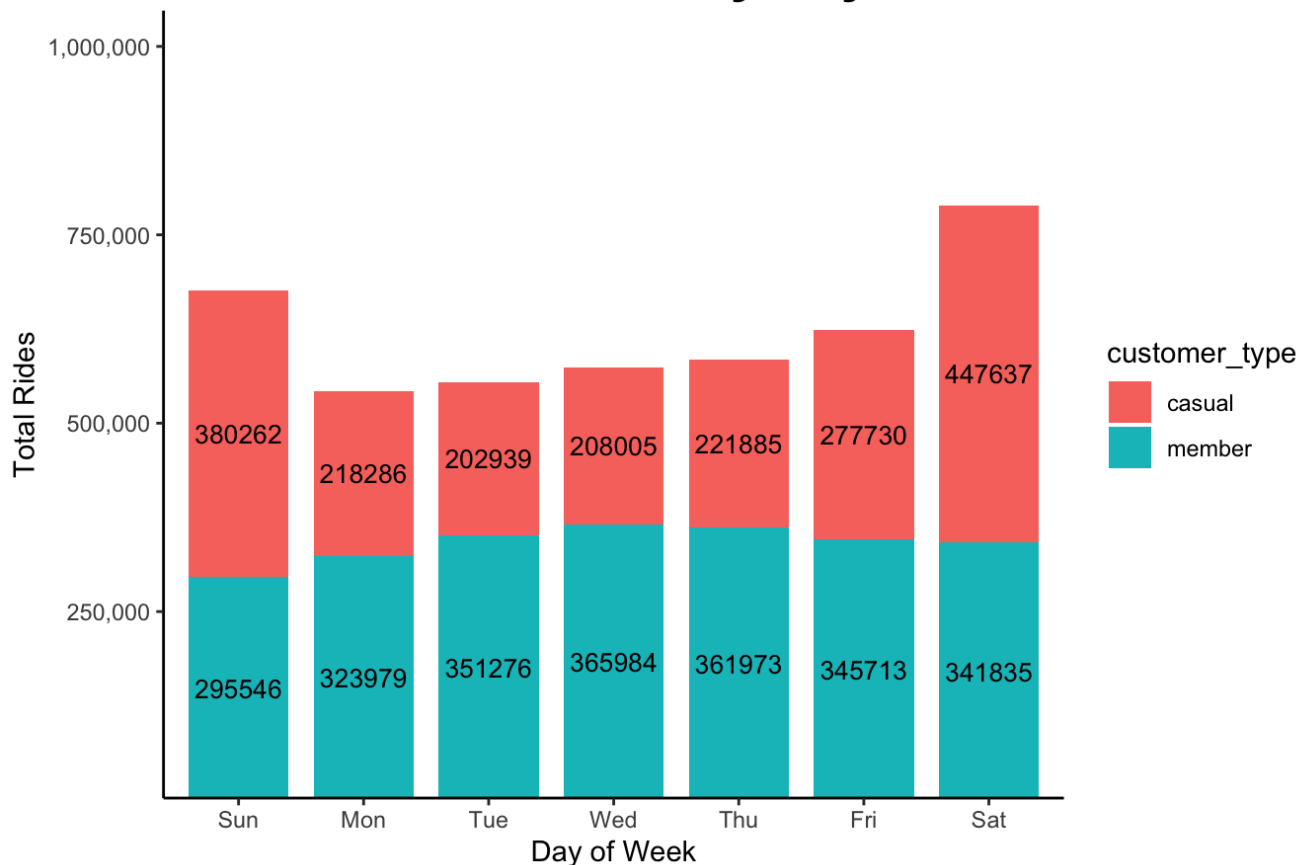
**H7: Members are more active on weekdays rather than weekends, casual riders are highly active on weekends.**

### Day of week rides bar chart

```
bike_cleaned_2 %>%
  group_by(day_of_week, customer_type) %>%
  summarise(count = n()) %>%
  ggplot(aes(x = day_of_week, y = count, fill = customer_type)) +
  geom_col(width = 0.8) +
  scale_y_continuous(name = "Total Rides", labels = function(x) format(x, big.mark =
",", scientific = F)) +
  coord_cartesian(ylim = c(50000, 1000000)) +
  labs(x = "Day of Week") +
  theme_classic() +
  ggtitle("Customer Total Rides by Day of Week") +
  theme(plot.title = element_text(hjust = 0.5, vjust = 0.5, face = "bold", size = 20)) +
  geom_text(aes(label = count), position = position_stack(vjust = 0.5), size = 3.5)
```

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

# Customer Total Rides by Day of Week



## Chi-square test of day of week rides

```
chisq.test(table(bike_cleaned_2$day_of_week, bike_cleaned_2$customer_type))
```

```
##  
##  Pearson's Chi-squared test  
##  
## data:  table(bike_cleaned_2$day_of_week, bike_cleaned_2$customer_type)  
## X-squared = 128377, df = 6, p-value < 2.2e-16
```

Chi-square test result show that p-value is less than 0.05, means that there is a significant difference between members and casuals rides by the day of week. There was evidence that at the 5% level, day of week is a factor which can impact total rides significantly.

The data suggests that casual users prefer to use bike on weekend, while members love to use bike during weekdays. On weekdays, the total rides of members are higher than casual users. And, on weekend, the total rides of casual users are higher than members. Casual riders demonstrated their peak usage Saturday, while members reached their highest point on Wednesday. It's worth noting that member ridership consistently exceeded that of casual riders except Sunday and Saturday. The seventh hypothesis, **H7: members are more active on weekdays rather than weekends, casual riders are highly active on weekends**, finds support in the data. Consequently, launching marketing campaigns on weekend for casual users and during weekdays for members would be advantageous for Cyclistic Company.

## H8 Day of week ride duration comparision

**H8: Members' total ride duration is higher than casuals on weekdays while lower than casuals on weekends.**

Bar chart of day of week ride duration



```

bike_cleaned_2 %>%
  group_by(day_of_week, customer_type) %>%
  summarise(total_duration_d = round(sum(ride_duration), 1)) %>%
  ggplot(aes(x = day_of_week, y = total_duration_d, fill = customer_type)) +
  geom_col(width = 0.95) +
  scale_y_continuous(name = "Total ride duration", labels = function(x) format(x, big.mark = ",", scientific = F)) +
  coord_cartesian(ylim = c(50000, 20000000)) +
  scale_x_discrete(expand = c(0.1, 0.1)) +
  labs(x = "Day of Week") +
  theme_classic() +
  ggtitle("Customers' Total Ride Duration per Day of Week") +
  theme(plot.title = element_text(hjust = 0.5, vjust = 0.8, face = "bold", size = 20),
        legend.position = "none") +
  geom_text(aes(label = total_duration_d), position = position_stack(vjust = 0.5), vjust = 1, size = 4)

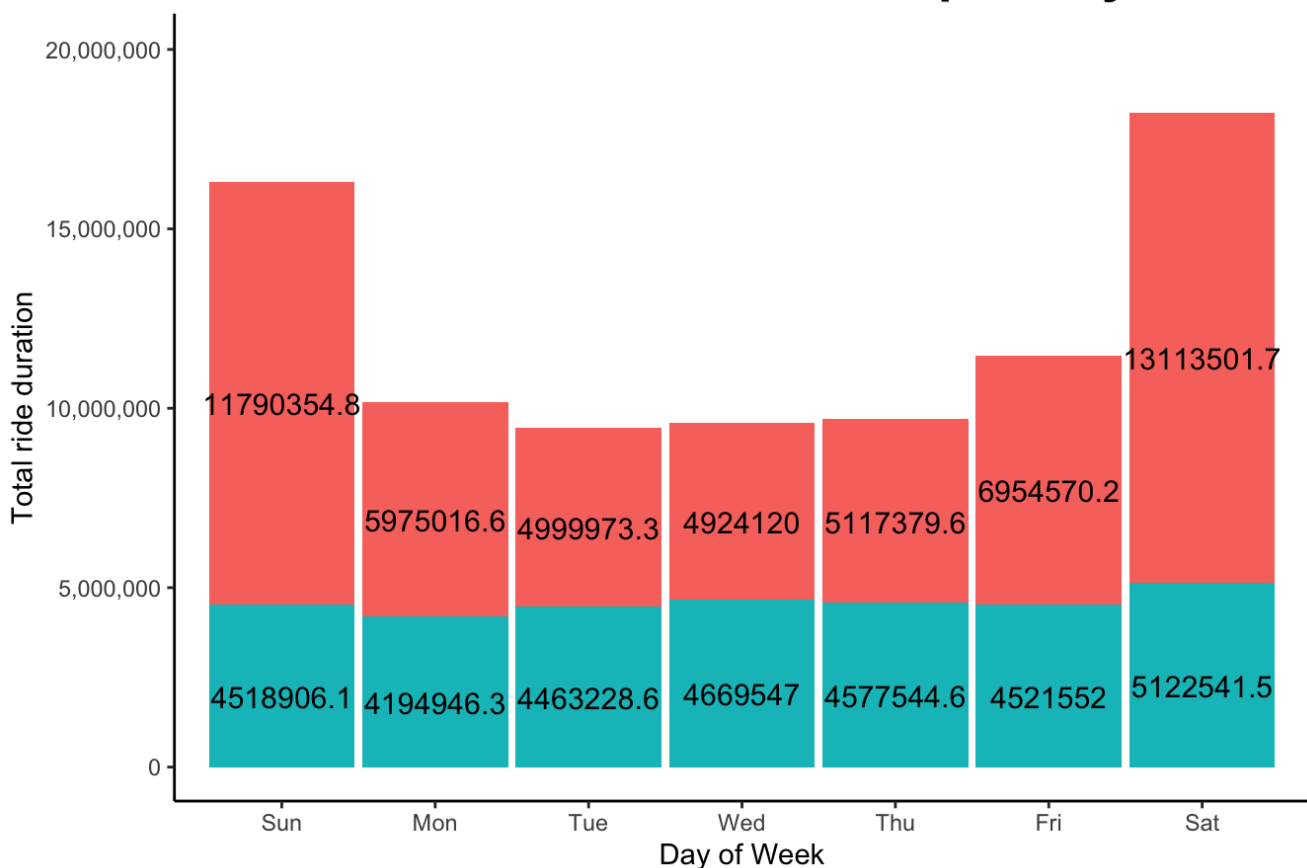
```

```

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

```

## Customers' Total Ride Duration per Day of Week



By the chart above, it's evident that casual users have higher total ride duration throughout all days of week. This pattern suggests that casual riders maintain longer ride duration compared to members during every day of week. The eighth hypothesis **H8**, which posited that **members' total ride duration is higher than casuals on weekdays while lower than casuals on weekends**, did not find support in the data. This reinforces the need for Cyclistic to formulate strategies aimed at casual users as they spend more time on Cyclistic's product.

# H9 Day of week ride duration mean

H9: Members' average ride duration is higher than casuals on weekdays while lower than casuals on weekends.

## Day of week ride duration mean

```
aggregate(bike_cleaned_2, ride_duration~ day_of_week + customer_type, FUN = mean)
```

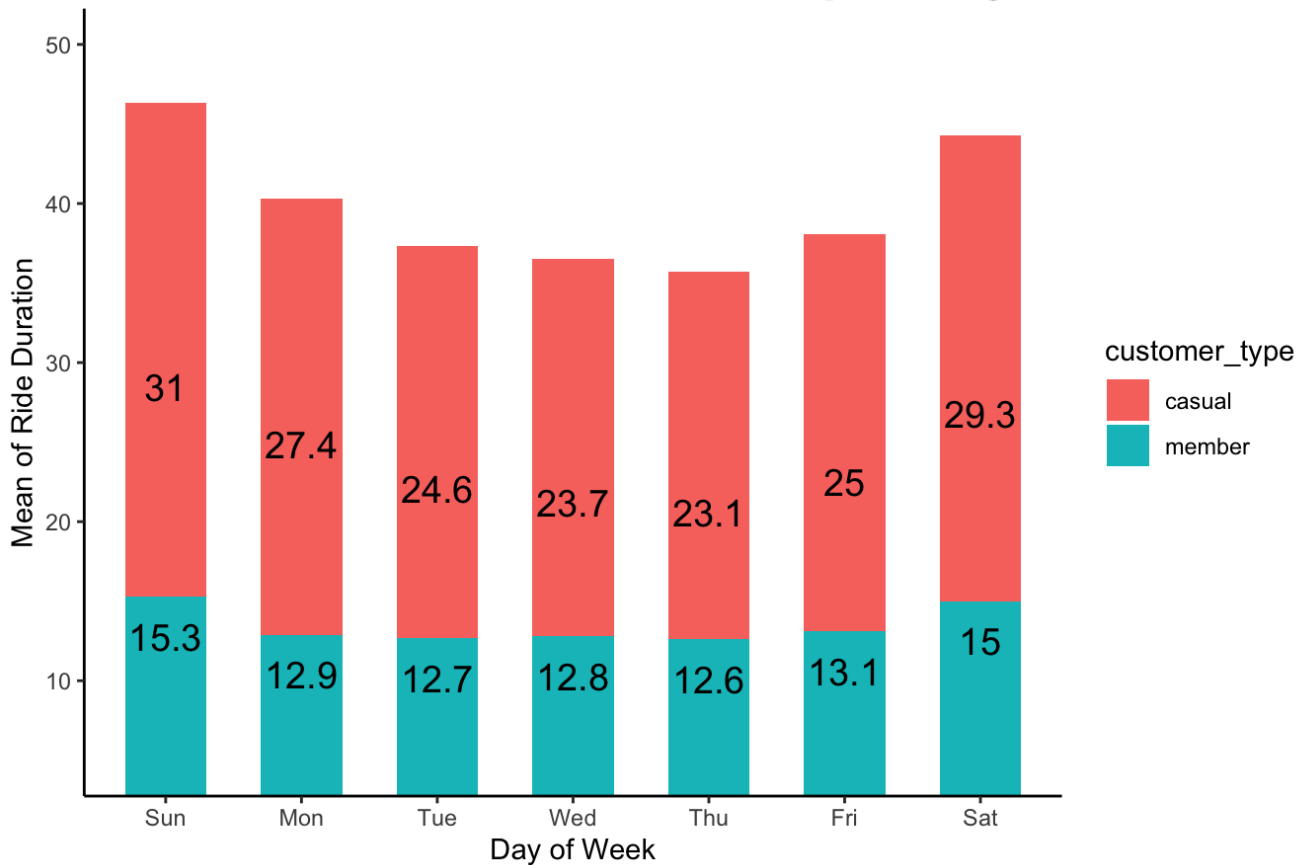
```
##      day_of_week customer_type ride_duration
## 1             Sun          casual    31.00587
## 2             Mon          casual    27.37242
## 3             Tue          casual    24.63781
## 4             Wed          casual    23.67308
## 5             Thu          casual    23.06321
## 6             Fri          casual    25.04076
## 7             Sat          casual    29.29495
## 8             Sun          member    15.29003
## 9             Mon          member    12.94820
## 10            Tue          member    12.70576
## 11            Wed          member    12.75888
## 12            Thu          member    12.64609
## 13            Fri          member    13.07892
## 14            Sat          member    14.98542
```

## Bar chart of day of week ride duration mean

```
bike_cleaned_2 %>%
  group_by(day_of_week, customer_type) %>%
  summarise(mean_duration_d = round(mean(ride_duration), 1)) %>%
  arrange(day_of_week) %>%
  ggplot(aes(x = day_of_week, y = mean_duration_d, fill = customer_type)) +
  geom_col(width = 0.6, position = "stack") +
  scale_y_continuous(name = "Mean of Ride Duration") +
  coord_cartesian(ylim = c(5, 50)) +
  labs(x = "Day of Week") +
  theme_classic() +
  ggtitle("Customer's Ride Duration Mean per Day of Week") +
  theme(plot.title = element_text(hjust = 0.3, vjust = 0.5, face = "bold", size = 20))
+
  geom_text(aes(label = mean_duration_d), vjust = 2, size = 5)
```

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

# Customer's Ride Duration Mean per Day of Week



```
bike_cleaned_2$combined_type <- interaction(bike_cleaned_2$bike_type, bike_cleaned_2$customer_type, drop = TRUE)
#leveneTest(ride_duration ~ combined_type, data = bike_cleaned_2)
```

The variance between customer type and bike type is not equal.

## T test for ride duration mean of day of week

```
bike_cleaned_2 %>%
  group_by(day_of_week) %>%
  rstatix::t_test(ride_duration ~ customer_type)
```

```
## # A tibble: 7 × 9
##   day_of_week .y.      group1 group2    n1    n2 statistic      df      p
## * <fct>      <chr>      <chr> <chr>   <int> <int>   <dbl>   <dbl> <dbl>
## 1 Sun        ride_duration casual member 380262 295546 264. 542431. 0
## 2 Mon        ride_duration casual member 218286 323979 212. 263087. 0
## 3 Tue        ride_duration casual member 202939 351276 182. 241738. 0
## 4 Wed        ride_duration casual member 208005 365984 175. 249956. 0
## 5 Thu        ride_duration casual member 221885 361973 174. 270498. 0
## 6 Fri        ride_duration casual member 277730 345713 209. 358024. 0
## 7 Sat        ride_duration casual member 447637 341835 272. 645028. 0
```

The results of the t-test reveal a significant difference in ride duration means between members and casuals for each day of the week. This finding indicates that, regardless of the specific day of the week, casual users generally have longer ride durations compared to members. This suggests that casual riders tend to use bikes for extended periods throughout the entire week, while members exhibit greater ride duration consistency during weekdays with minimal fluctuations. Casual users experience their shortest ride duration on Thursday (23.1 minutes) and their longest on Sunday (31 minutes).

Notably, both member and casual users demonstrate extended average ride duration over the weekend.

Contrary to my hypothesis (**H9**) that suggested a **higher peak at 8 AM for members on weekdays and lower on weekends compared to casuals**, the data does not support this hypothesis. This discrepancy is another indicator that Cyclistic company should pay more attention to casual users. They may need to recheck their financial data to determine whether member users are the most profitable segment.

## H10

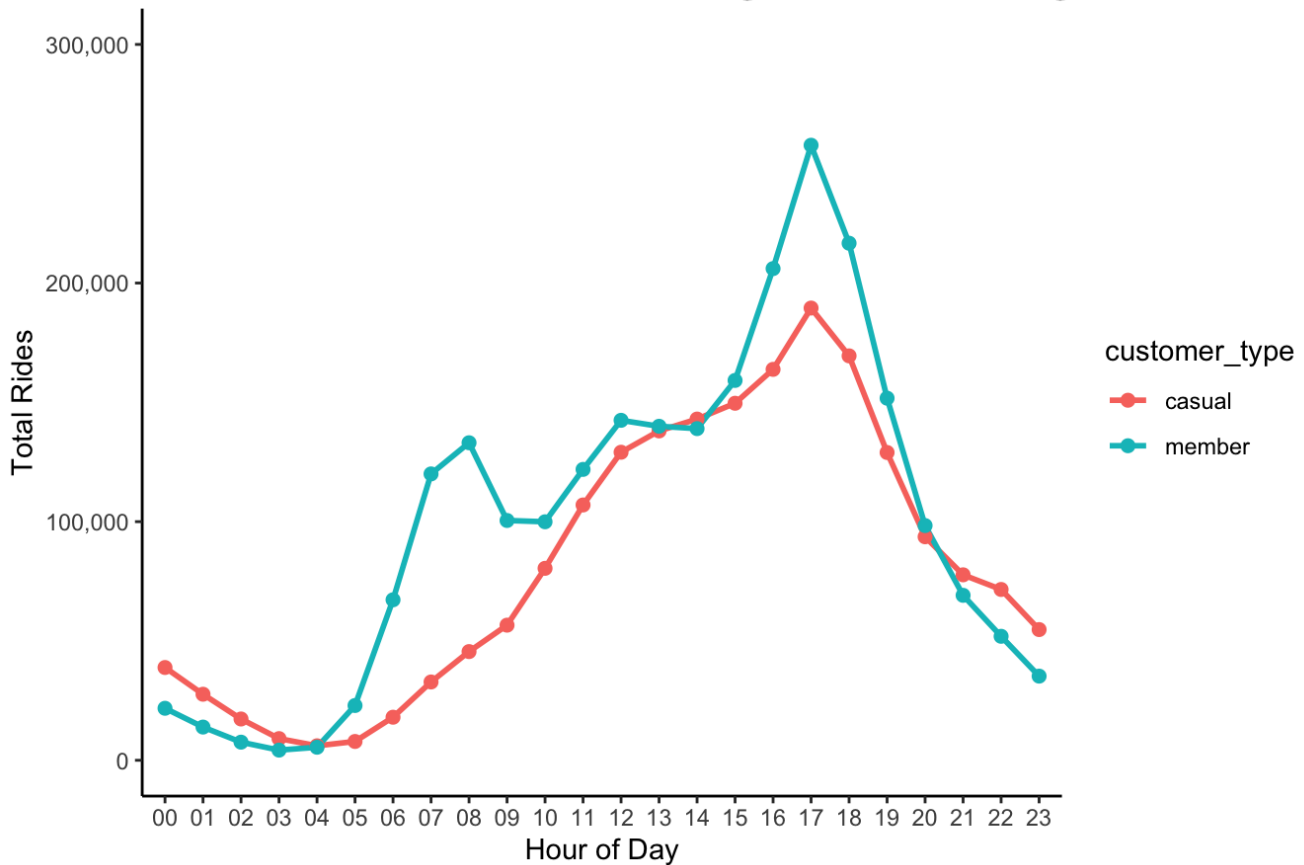
**H10: Members have a ride peak at 8 mornings and at 17 afternoons during weekdays while casual users tend to ride more in the afternoon during weekends.**

### Customer rides line chart by hour of day

```
bike_cleaned_2 %>%
  group_by(started_hour, customer_type) %>%
  summarise(count_h = n()) %>%
  ggplot(aes(x = started_hour, y = count_h, color = customer_type, group = customer_type)) +
  geom_line(linewidth = 1) +
  geom_point(size = 2) +
  scale_y_continuous(name = "Total Rides", labels = function(x) format(x, big.mark =
",", scientific = F)) +
  coord_cartesian(ylim = c(0, 300000)) +
  labs(x = "Hour of Day") +
  theme_classic() +
  ggtitle("Customers Total Rides by Hour of Day") +
  theme(plot.title = element_text(hjust = 0.5, vjust = 0.5, face = "bold", size = 20))
```

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

# Customers Total Rides by Hour of Day



Based on the chart provided, a clear pattern emerges: member users exhibit peak ride activity during both mornings at 8 AM and afternoons at 5 PM, while casual users predominantly show a peak ride time exclusively afternoons at 5 PM. This observation lends support to the tenth hypothesis, **H10: Members have a ride peak at 8 mornings and at 17 afternoons during weekdays while casual users tend to ride more in the afternoon during weekends.**

## H11: Members' total rides per hour of the day are higher than casuals.

**H11: Members' total rides per hour of the day are higher than casuals.**

```
chisq.test(table(bike_cleaned_2$customer_type, bike_cleaned_2$started_hour))
```

```
##
##  Pearson's Chi-squared test
##
## data:  table(bike_cleaned_2$customer_type, bike_cleaned_2$started_hour)
## X-squared = 149434, df = 23, p-value < 2.2e-16
```

The results of the chi-square test indicate a p-value of less than 0.05, indicating a statistically significant difference in the preferred ride hours between members and casual users. Specifically, member riders have two distinct preferred periods: the first period is from 6 AM to 9 AM, and the second period is from 3 PM to 7 PM. In contrast, casual users exhibit a single preferred period, which is from 3 PM to 7 PM. Additionally, during the time period from 9 PM to 3 AM, casual users have higher ride frequencies than members. As a result, the eleventh hypothesis, **H11: Members' overall rides per hour of the day are higher than those of casuals**, is not supported.

## H12: Members' average ride duration per hour of the day is higher than casuals.

H12: Members' average ride duration per hour of the day is higher than casuals.

Ride duration mean by hour of day

```
aggregate(bike_cleaned_2, ride_duration~customer_type + started_hour, FUN = mean)
```

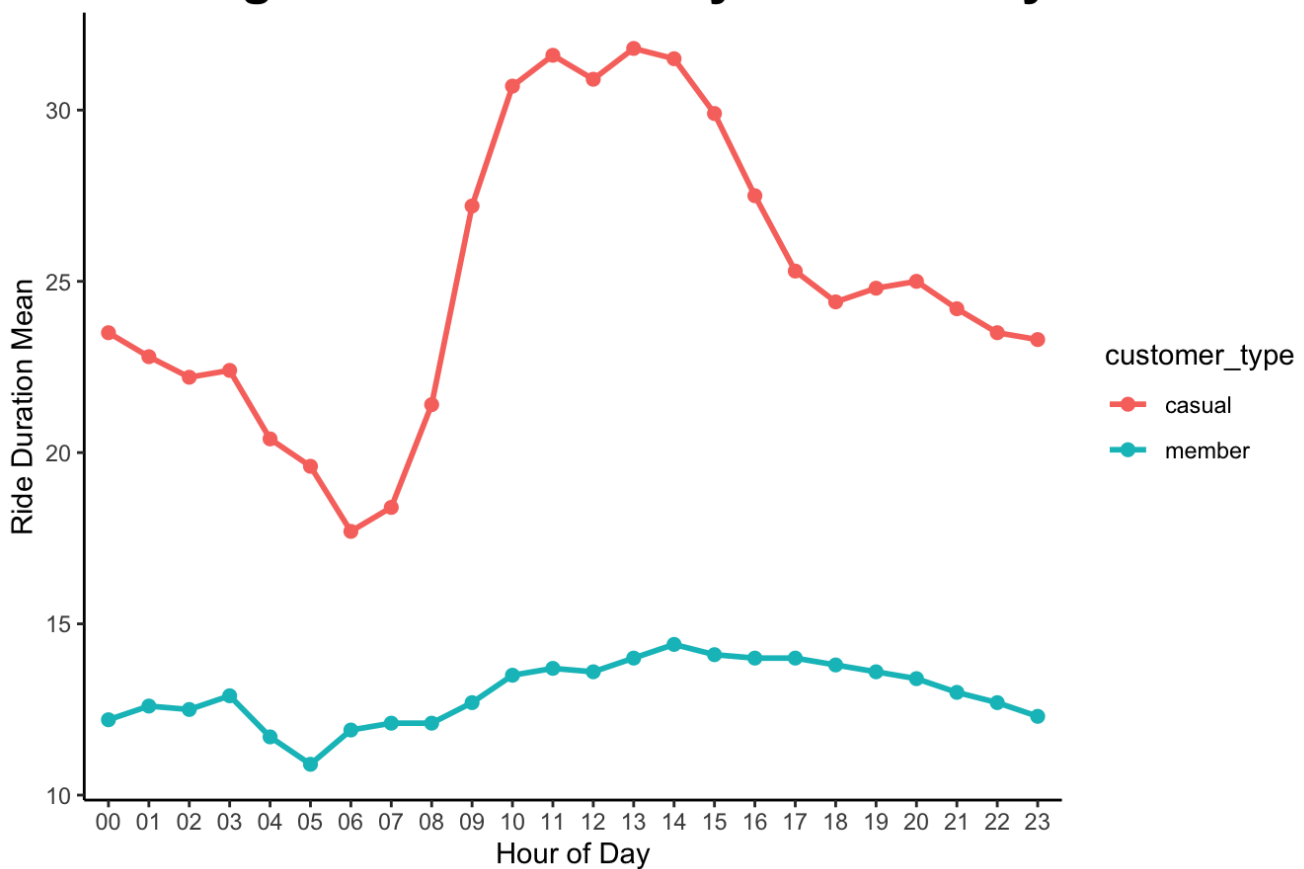
##	customer_type	started_hour	ride_duration
## 1	casual	00	23.52395
## 2	member	00	12.19040
## 3	casual	01	22.78001
## 4	member	01	12.57688
## 5	casual	02	22.22165
## 6	member	02	12.54819
## 7	casual	03	22.42704
## 8	member	03	12.85106
## 9	casual	04	20.44658
## 10	member	04	11.65212
## 11	casual	05	19.58184
## 12	member	05	10.94873
## 13	casual	06	17.70604
## 14	member	06	11.85674
## 15	casual	07	18.44755
## 16	member	07	12.08027
## 17	casual	08	21.41048
## 18	member	08	12.08692
## 19	casual	09	27.19392
## 20	member	09	12.65344
## 21	casual	10	30.69946
## 22	member	10	13.48381
## 23	casual	11	31.59941
## 24	member	11	13.74236
## 25	casual	12	30.88798
## 26	member	12	13.59812
## 27	casual	13	31.77114
## 28	member	13	13.95049
## 29	casual	14	31.46794
## 30	member	14	14.36625
## 31	casual	15	29.86641
## 32	member	15	14.11615
## 33	casual	16	27.53829
## 34	member	16	14.02153
## 35	casual	17	25.34928
## 36	member	17	14.03994
## 37	casual	18	24.42931
## 38	member	18	13.83907
## 39	casual	19	24.77987
## 40	member	19	13.60101
## 41	casual	20	25.03570
## 42	member	20	13.36551
## 43	casual	21	24.16309
## 44	member	21	12.99250
## 45	casual	22	23.46371
## 46	member	22	12.67809
## 47	casual	23	23.27230
## 48	member	23	12.34701

Ride duration mean line chart by hour of day

```
bike_cleaned_2 %>%
  group_by(started_hour, customer_type) %>%
  summarise(mean_duration_h = round(mean(ride_duration), 1)) %>%
  ggplot(aes(x = started_hour, y = mean_duration_h, color = customer_type, group = customer_type)) +
  geom_line(linewidth = 1) +
  geom_point(size = 2) +
  labs(x = "Hour of Day", y = "Ride Duration Mean") +
  theme_classic() +
  ggtitle("Average Ride Duration by Hour of Day") +
  theme(plot.title = element_text(hjust = 0.5, vjust = 0.5, face = "bold", size = 20))
```

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

## Average Ride Duration by Hour of Day



The chart provided reveals a distinct pattern: the average ride duration of members is lower than that of casual users across different hours of the day.

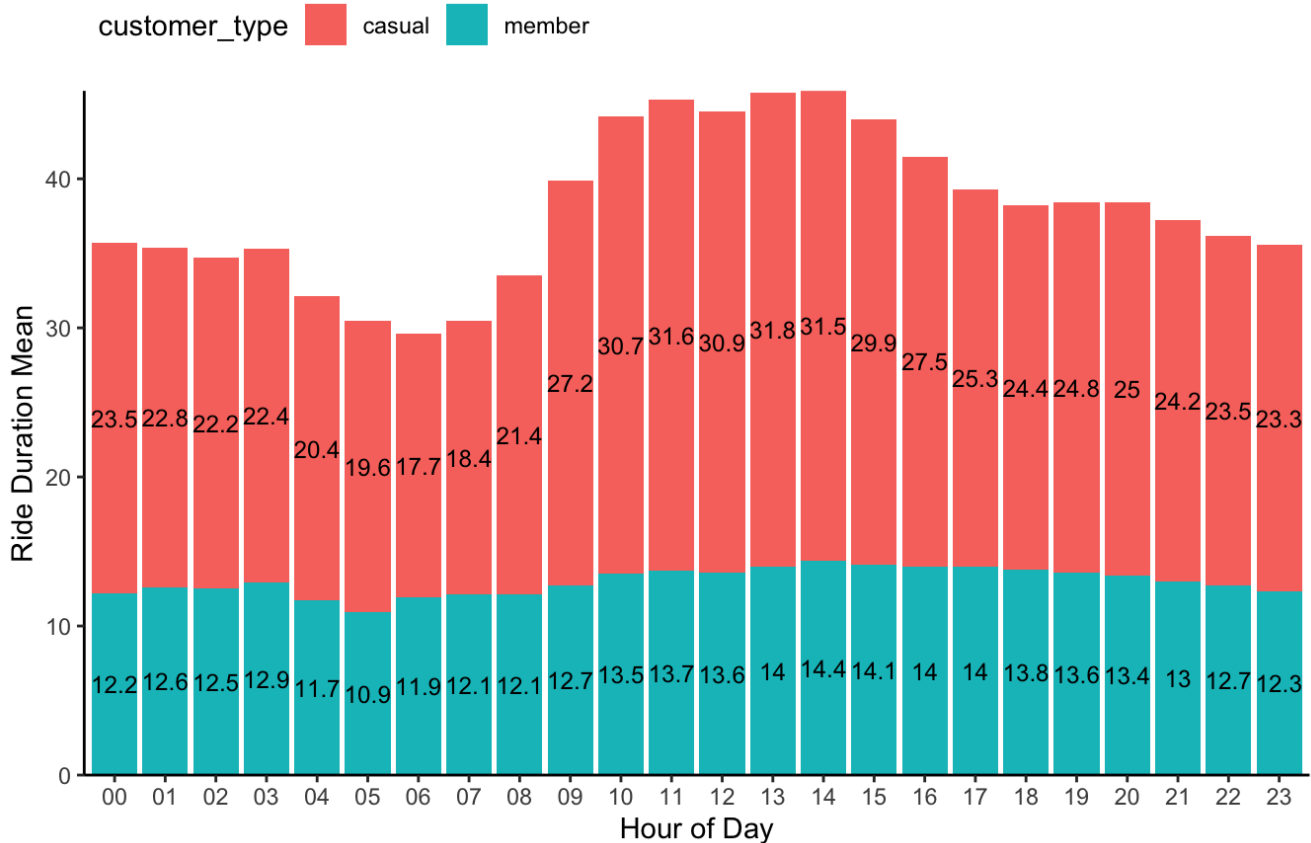
### Ride duration mean bar chart by hour of day



```
bike_cleaned_2 %>%
  group_by(started_hour, customer_type) %>%
  summarise(mean_duration_hh = round(mean(ride_duration), 1)) %>%
  ggplot(aes(x = started_hour, y = mean_duration_hh, fill = customer_type)) +
  geom_col(width = 0.9) +
  scale_y_continuous(expand = c(0, 0)) +
  labs(x = "Hour of Day", y = "Ride Duration Mean") +
  theme_classic() +
  ggtitle("Average Ride Duration by Hour of Day") +
  theme(plot.title = element_text(hjust = 0.5, vjust = 0.5, face = "bold", size = 20),
        legend.position = "top", legend.justification = "left") +
  geom_text(aes(label = mean_duration_hh), position = position_stack(vjust = 0.5), vjust = 0.5, size = 3.2)
```

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

## Average Ride Duration by Hour of Day



### T test for ride duration mean of hour of day

```
bike_cleaned_2 %>%
  group_by(started_hour) %>%
  rstatix::t_test(ride_duration ~ customer_type)
```

```
## # A tibble: 24 × 9
##   started_hour .y.      group1 group2    n1    n2 statistic    df      p
## * <chr>      <chr>      <chr> <chr> <int> <int>    <dbl> <dbl>    <dbl>
## 1 00          ride_dura... casual member 38885 21793    74.1 57095. 0
## 2 01          ride_dura... casual member 27676 13921    56.6 41339. 0
## 3 02          ride_dura... casual member 17307  7597    42.2 24749. 0
## 4 03          ride_dura... casual member  9092  4260    27.9 13057. 7.13e-166
## 5 04          ride_dura... casual member  6074  5472    23.9  8706. 9.99e-123
## 6 05          ride_dura... casual member  7888 22955    27.6  8601. 4.64e-161
## 7 06          ride_dura... casual member 18052 67255    31.2 19835. 1.28e-208
## 8 07          ride_dura... casual member 32818 120013    47.2 36012. 0
## 9 08          ride_dura... casual member 45557 133069    70.2 50303. 0
## 10 09         ride_dura... casual member 56625 100498   102.  65386. 0
## # i 14 more rows
```

The results of the t-test highlight a significant disparity in ride duration means between members and casual users across every hour of the day. This finding underscores the fact that, irrespective of the specific time, casual users tend to have longer ride durations compared to members. This observation implies that casual riders frequently utilize bikes for extended periods throughout the entire day, while members exhibit a more consistent ride duration with minimal fluctuations. Notably, casual users experience their briefest ride duration at 6 am (17.7 minutes) and their lengthiest at 11 am (31.6 minutes).

In contrast to my hypothesis (**H12**), which posited that **members' average ride duration per hour of the day is higher than that of casuals**, the data fails to support this conjecture. This discrepancy serves as a clear indicator that Cyclistic should allocate more attention to casual users. It may be necessary to review their financial data to ascertain whether member users indeed constitute the most profitable segment.

## H13: Members' starting and ending locations are clustered in the downtown area, while casual customers' locations are more dispersed and along the beach.

**H13: Members' starting and ending locations are clustered in the downtown area, while casual customers' locations are more dispersed and along the beach.**

### Calculating total rides for every start station

#### Calculating total rides of start stations for member users

```
start_station_map_member <- bike_cleaned_2 %>%
  select(start_station_name,
         start_lat,
         start_lng,
         customer_type) %>%
  filter(customer_type == "member") %>%
  group_by(start_station_name) %>%
  mutate(num_rides = n()) %>%
  distinct(start_station_name, .keep_all = TRUE)
arrange(start_station_map_member, desc(num_rides))
```

```
## # A tibble: 756 × 5
## # Groups:   start_station_name [756]
##   start_station_name      start_lat start_lng customer_type num_rides
##   <chr>                <dbl>    <dbl> <chr>          <int>
## 1 Clark St & Elm St      41.9     -87.6 member        23161
## 2 Wells St & Concord Ln  41.9     -87.6 member        21260
## 3 Kingsbury St & Kinzie St 41.9     -87.6 member        20289
## 4 Wells St & Elm St      41.9     -87.6 member        19130
## 5 Dearborn St & Erie St   41.9     -87.6 member        17801
## 6 St. Clair St & Erie St  41.9     -87.6 member        17517
## 7 Wells St & Huron St    41.9     -87.6 member        17336
## 8 Broadway & Barry Ave   41.9     -87.6 member        17128
## 9 Clark St & Armitage Ave 41.9     -87.6 member        15934
## 10 Theater on the Lake   41.9     -87.6 member        15933
## # i 746 more rows
```

```
sum(start_station_map_member$num_rides)
```

```
## [1] 2386306
```

## Calculating total rides of start stations for casual users

```
start_station_map_casual <- bike_cleaned_2 %>%
  select( start_station_name,
          start_lat,
          start_lng,
          customer_type) %>%
  filter(customer_type == "casual") %>%
  group_by(start_station_name) %>%
  mutate(num_rides = n()) %>%
  distinct(start_station_name, .keep_all = TRUE)
arrange(start_station_map_casual, desc(num_rides))
```

```
## # A tibble: 776 × 5
## # Groups:   start_station_name [776]
##   start_station_name      start_lat start_lng customer_type num_rides
##   <chr>                <dbl>    <dbl> <chr>          <int>
## 1 Streeter Dr & Grand Ave  41.9     -87.6 casual        60125
## 2 Millennium Park        41.9     -87.6 casual        30646
## 3 Michigan Ave & Oak St   41.9     -87.6 casual        28114
## 4 Lake Shore Dr & Monroe St 41.9     -87.6 casual        22743
## 5 Shedd Aquarium          41.9     -87.6 casual        21231
## 6 Theater on the Lake     41.9     -87.6 casual        20955
## 7 Wells St & Concord Ln    41.9     -87.6 casual        17746
## 8 Lake Shore Dr & North Blvd 41.9     -87.6 casual        16005
## 9 Indiana Ave & Roosevelt Rd 41.9     -87.6 casual        15954
## 10 Clark St & Lincoln Ave   41.9     -87.6 casual        15774
## # i 766 more rows
```

```
sum(start_station_map_casual$num_rides)
```

```
## [1] 1956744
```

## Calculating total rides of start stations for all users

```
start_station_map <- start_station_map_casual %>%  
  union(start_station_map_member)  
tail(start_station_map)
```

```
## # A tibble: 6 × 5  
## # Groups:   start_station_name [6]  
##   start_station_name      start_lat start_lng customer_type num_rides  
##   <chr>                <dbl>    <dbl> <chr>          <int>  
## 1 Narragansett & McLean      41.9     -87.8 member           1  
## 2 WEST CHI-WATSON           41.9     -87.7 member           1  
## 3 Kildare Ave & 26th St     41.8     -87.7 member           1  
## 4 Panama Ave & Grace St     42.0     -87.8 member           1  
## 5 Doty Ave & 111th St       41.7     -87.6 member           1  
## 6 Central Park Ave & Douglas Blvd 41.9     -87.7 member           2
```

```
sum(subset(start_station_map, customer_type == "member")$num_rides)
```

```
## [1] 2386306
```

```
sum(subset(start_station_map, customer_type == "casual")$num_rides)
```

```
## [1] 1956744
```

## Creating bins start\_station\_map

```
start_bins <- seq(0, 70000, by = 1000)
```

## Assigning color to show ride density per station.

```
density_color <- colorBin(palette = "magma", domain = start_station_map$num_rides, na.c  
olor = "transparent", bins = start_bins, reverse = T)
```

## Setting text for interactive tooltip

```
start_map_text <- paste("station name: ", start_station_map$start_station_name, "<br/  
>", "number of rides: ", start_station_map$num_rides, "<br/>", "customer type: ", start  
_station_map$customer_type, sep = "") %>%  
  lapply(htmltools::HTML)
```

## Setting map title for all users

```

tag.map.title <- htmltools::tags$style(htmltools::HTML("
  .leaflet-control.map-title {
    transform: translate(-50%,20%);
    position: fixed !important;
    left: 25%;
    text-align: center;
    padding-left: 10px;
    padding-right: 10px;
    background: rgba(255,255,255,0.75);
    font-weight: bold;
    font-size: 18px;
    color: blue
  }
"))

title <- htmltools::tags$div(
  tag.map.title, htmltools::HTML("Popular Start Stations")
)

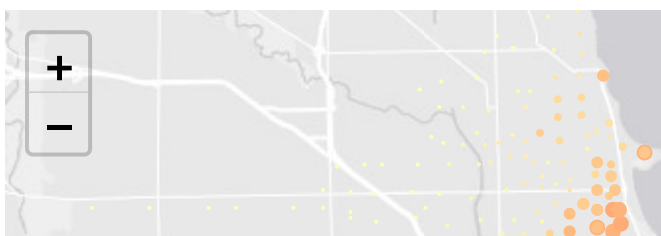
```

## Creating interactive html leaflet widget to show RIDE density for all users

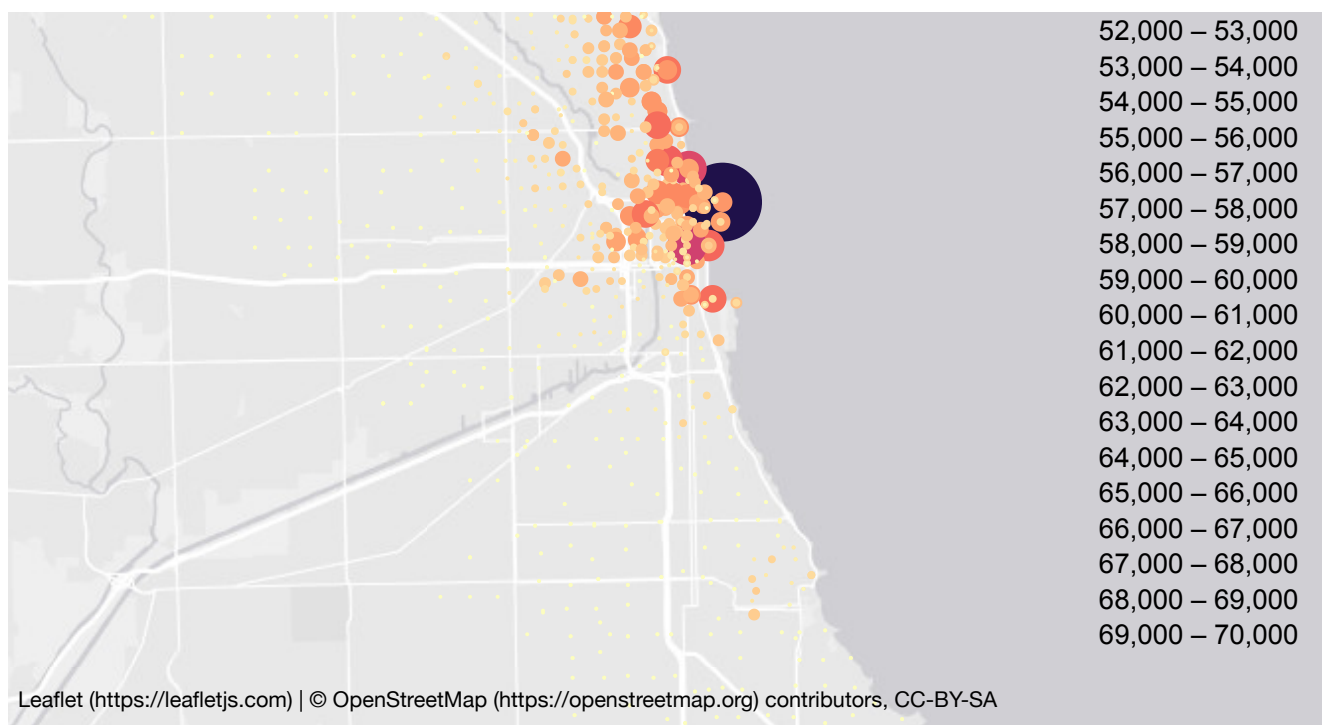
```

rides_per_station <- leaflet(start_station_map) %>%
  leaflet::addTiles() %>%
  leaflet::setView(lng = -87.6298, lat = 41.8781, zoom = 10.5) %>% # Chicago coordinates
  leaflet::addProviderTiles("Esri.WorldGrayCanvas") %>% # set map style
  leaflet::addCircleMarkers(~ start_lng, ~ start_lat, # add circle markers for each station, set fill color, add tooltip.
    fillColor = ~ density_color(num_rides),
    fillOpacity = 1,
    color = "white",
    # radius = ~ start_station_map$num_rides/1500, #this is an option for variable marker size based on number of rides.
    radius = start_station_map$num_rides/3000,
    stroke = FALSE,
    label = start_map_text,
    labelOptions = labelOptions(style = list("font-weight" = "normal", padding = "3px 8px"),
    textsize = "13px",
    direction = "auto")) %>%
  #add legend.
  leaflet::addLegend(
    pal = density_color,
    values = ~ num_rides,
    opacity = 0.8,
    title = "Number of Rides",
    position = "bottomright") %>%
  leaflet::addControl(title, position = "topleft", className="map-title")
rides_per_station

```



46,000 – 47,000
47,000 – 48,000
48,000 – 49,000
49,000 – 50,000
50,000 – 51,000
51,000 – 52,000



The station that stands out as the most popular is Streeter Dr & Grand Ave. This particular station boasts proximity to the lakeside train connector, along with Jane Addams Memorial Park and Ohio Beach Street. It is highly likely to be frequented by tourists and weekend visitors. Among the top ten stations, six are conveniently located next to public parks, three are situated in front of sizable residential buildings, and two are in close proximity to fitness centers.

## Assigning color to show ride density per station.

```
density_color_member <- colorBin(palette = "magma", domain = start_station_map_member$num_rides, na.color = "transparent", bins = start_bins, reverse = T)
```

```
density_color_casual <- colorBin(palette = "magma", domain = start_station_map_casual$num_rides, na.color = "transparent", bins = start_bins, reverse = T)
```

## Setting text for interactive tooltip

```
start_map_text_member <- paste("station name: ", start_station_map_member$start_station_name, "<br/>", "number of rides: ", start_station_map_member$num_rides, sep = "") %>%  
  lapply(htmltools::HTML)
```

```
start_map_text_casual <- paste("station name: ", start_station_map_casual$start_station_name, "<br/>", "number of rides: ", start_station_map_casual$num_rides, sep = "") %>%  
  lapply(htmltools::HTML)
```

## Setting map title for member users

```

tag.map.title <- htmltools::tags$style(htmltools::HTML("
  .leaflet-control.map-title {
    transform: translate(-50%,20%);
    position: fixed !important;
    left: 25%;
    text-align: center;
    padding-left: 10px;
    padding-right: 10px;
    background: rgba(255,255,255,0.75);
    font-weight: bold;
    font-size: 18px;
    color: blue
  }
"))

member_map_title <- htmltools::tags$div(
  tag.map.title, htmltools::HTML("Popular Stations for Member User")
)

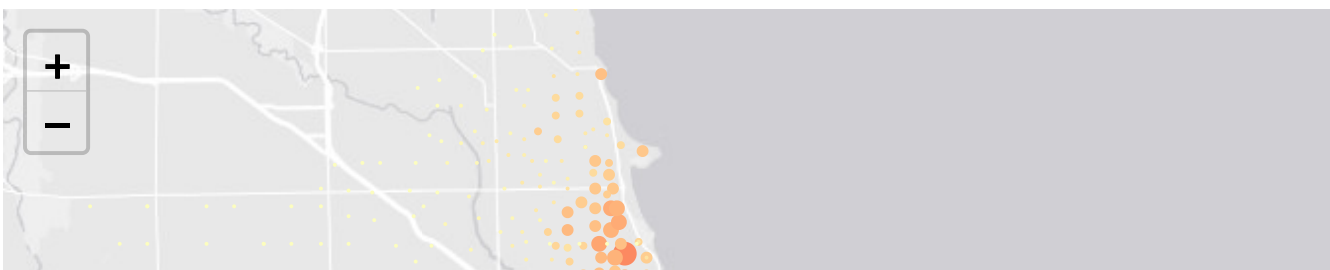
```

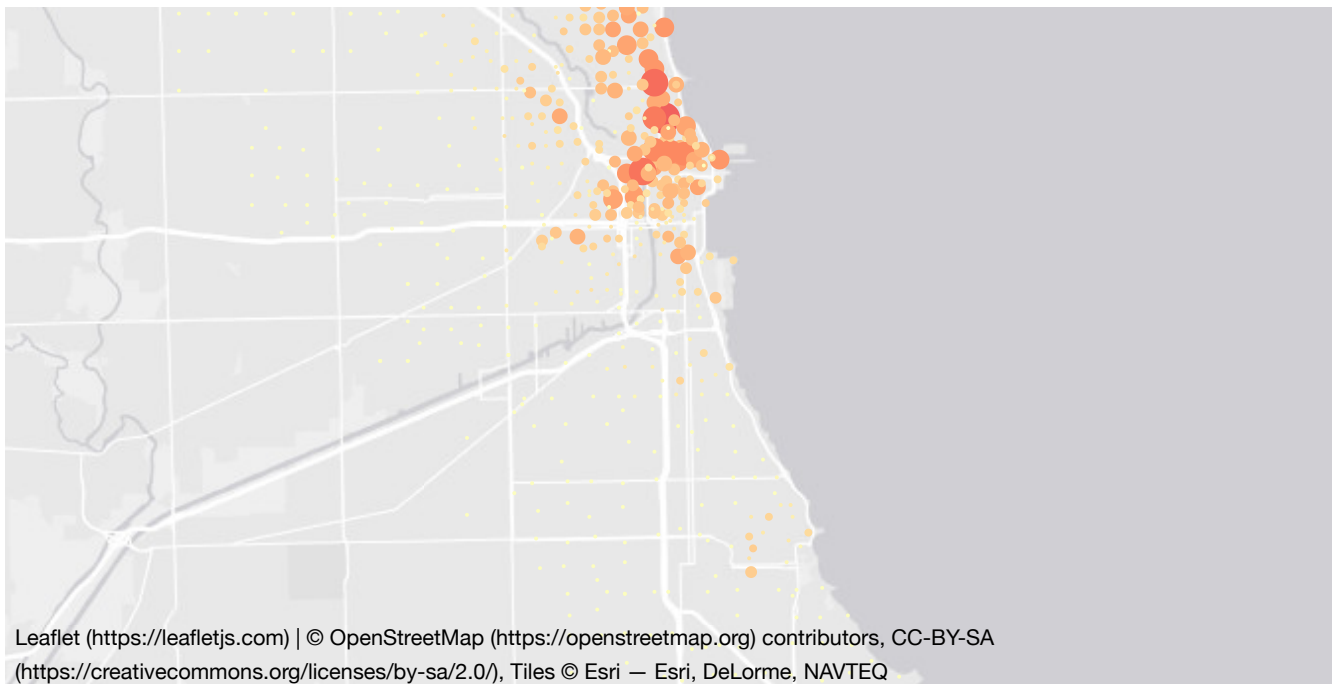
## Creating interactive html leaflet widget to show RIDE density for member users

```

rides_per_station_member <- leaflet(start_station_map_member) %>%
  leaflet::addTiles() %>%
  leaflet::setView(lng = -87.6298, lat = 41.8781, zoom = 10.5) %>% # Chicago coordinates
  leaflet::addProviderTiles("Esri.WorldGrayCanvas") %>% # set map style
  leaflet::addCircleMarkers(~ start_lng, ~ start_lat, # add circle markers for each station, set fill color, add tooltip.
    fillColor = ~ density_color_member(num_rides),
    fillOpacity = 1,
    color = "white",
    # radius = ~ start_station_map$num_rides/1500, #this is an option for variable marker size based on number of rides.
    radius = start_station_map_member$num_rides/3000,
    stroke = FALSE,
    label = start_map_text_member,
    labelOptions = labelOptions(style = list("font-weight" = "normal", padding = "3px 8px"),
    textsize = "13px",
    direction = "auto"))
  #add legend.
  #addLegend(
    #pal = density_color_member,
    # values = ~ num_rides,
    #opacity = 0.8,
    #title = "Number of Rides",
    #position = "bottomright")
rides_per_station_member

```





## Setting map title for casual users

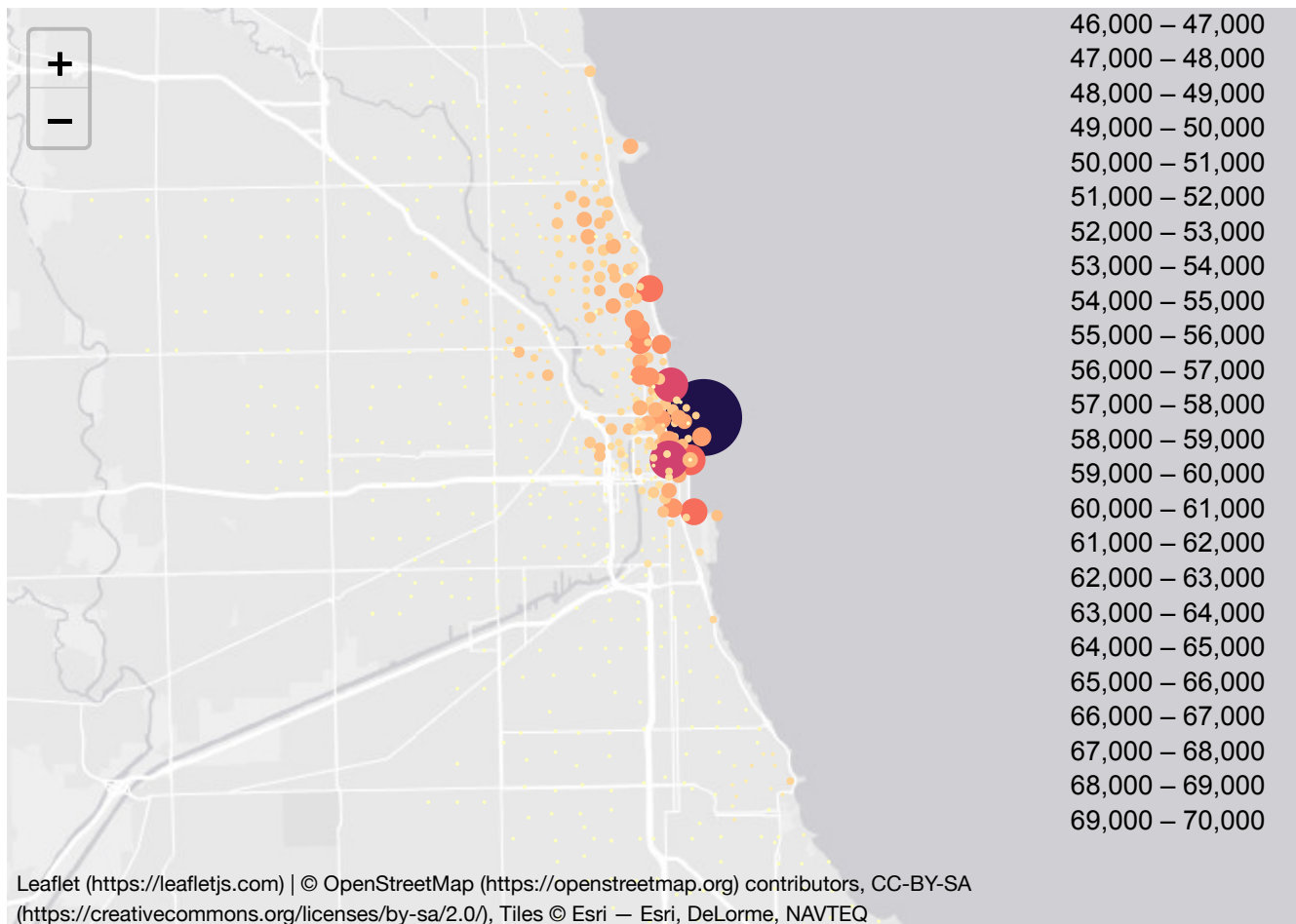
```
tag.casual.title <- htmltools::tags$style(htmltools::HTML("
  .leaflet-control-map-title {
    transform: translate(-50%,20%);
    position: fixed !important;
    left: 65%;
    text-align: center;
    padding-left: 10px;
    padding-right: 10px;
    background: rgba(255,255,255,0.75);
    font-weight: bold;
    font-size: 18px;
    color: blue
  }
"))

casual_map_title <- htmltools::tags$div(
  tag.casual.title, htmltools::HTML("Casual Users")
)
```

## Creating interactive html leaflet widget to show RIDE density for casual users

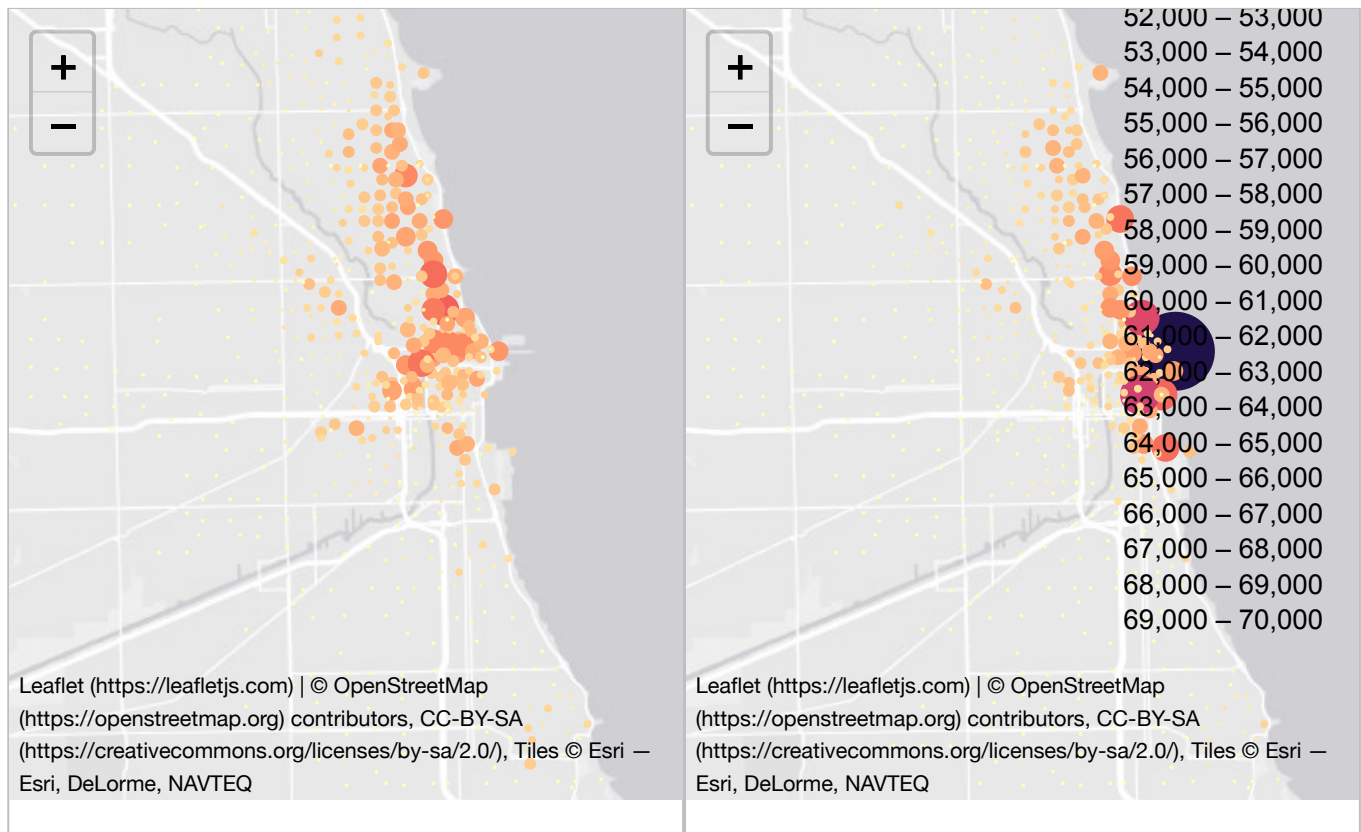


```
rides_per_station_casual <- leaflet(start_station_map_casual) %>%
  leaflet::addTiles() %>%
  leaflet::setView(lng = -87.6298, lat = 41.8781, zoom = 10.5) %>% # Chicago coordinates
  leaflet::addProviderTiles("Esri.WorldGrayCanvas") %>% # set map style
  leaflet::addCircleMarkers(~ start_lng, ~ start_lat, # add circle markers for each station, set fill color, add tooltip.
    fillColor = ~ density_color_casual(num_rides),
    fillOpacity = 1,
    color = "white",
    # radius = ~ start_station_map$num_rides/1500, #this is an option for variable marker size based on number of rides.
    radius = start_station_map_casual$num_rides/3000,
    stroke = FALSE,
    label = start_map_text_casual,
    labelOptions = labelOptions(style = list("font-weight" = "normal", padding = "3px 8px"),
    textsize = "13px",
    direction = "auto")) %>%
  #add legend.
  leaflet::addLegend(
    pal = density_color_casual,
    values = ~ num_rides,
    opacity = 0.8,
    title = "Number of Rides",
    position = "bottomright") %>%
  leaflet::addControl(casual_map_title, position = "topleft", className="map-title")
rides_per_station_casual
```



## Comparing maps for member and casual users

```
sync_maps <- sync(rides_per_station_member, rides_per_station_casual)
sync_maps
```



Through the maps above, we can clearly see that for casual riders the areas of their interest are located around the city center and along the beach where the most cultural & leisure points are.

In contrast, members' bike usage during the week is less dense in tourist areas but instead, it's quite heavy around Chicago's downtown area which could also prove our hypothesis that most of the annual members commute daily to work.

For the starting stations, Casual riders often started from aquariums, the vicinity of museums, parks, beaches, and harbor points. In contrast, members often started from stations close to universities, residential areas, restaurants, hospitals, grocery stores, etc.

## Calculating total rides of end stations for member users

```
end_station_map_member <- bike_cleaned_2 %>%
  select(end_station_name,
         end_lat,
         end_lng,
         customer_type) %>%
  filter(customer_type == "member") %>%
  group_by(end_station_name) %>%
  mutate(num_rides = n()) %>%
  distinct(end_station_name, .keep_all = TRUE)
  arrange(end_station_map_member, desc(num_rides))
```

```
## # A tibble: 754 × 5
## # Groups:   end_station_name [754]
##   end_station_name      end_lat end_lng customer_type num_rides
##   <chr>                <dbl>   <dbl> <chr>          <int>
## 1 Clark St & Elm St      41.9    -87.6 member         23484
## 2 Wells St & Concord Ln  41.9    -87.6 member         21886
## 3 Kingsbury St & Kinzie St 41.9    -87.6 member         20723
## 4 Wells St & Elm St      41.9    -87.6 member         19564
## 5 Dearborn St & Erie St   41.9    -87.6 member         18425
## 6 St. Clair St & Erie St  41.9    -87.6 member         17724
## 7 Broadway & Barry Ave   41.9    -87.6 member         17436
## 8 Wells St & Huron St     41.9    -87.6 member         16724
## 9 Clark St & Armitage Ave  41.9    -87.6 member         15304
## 10 Clark St & Lincoln Ave 41.9    -87.6 member         15017
## # i 744 more rows
```

```
sum(end_station_map_member$num_rides)
```

```
## [1] 2386306
```

## Calculating total rides of end stations for casual users

```
end_station_map_casual <- bike_cleaned_2 %>%
  select(end_station_name,
         end_lat,
         end_lng,
         customer_type) %>%
  filter(customer_type == "casual") %>%
  group_by(end_station_name) %>%
  mutate(num_rides = n()) %>%
  distinct(end_station_name, .keep_all = TRUE)
arrange(end_station_map_casual, desc(num_rides))
```

```
## # A tibble: 774 × 5
## # Groups:   end_station_name [774]
##   end_station_name      end_lat end_lng customer_type num_rides
##   <chr>                <dbl>   <dbl> <chr>          <int>
## 1 Streeter Dr & Grand Ave  41.9    -87.6 casual         62556
## 2 Millennium Park         41.9    -87.6 casual         32285
## 3 Michigan Ave & Oak St    41.9    -87.6 casual         29689
## 4 Theater on the Lake      41.9    -87.6 casual         22753
## 5 Lake Shore Dr & Monroe St 41.9    -87.6 casual         21606
## 6 Shedd Aquarium          41.9    -87.6 casual         19624
## 7 Lake Shore Dr & North Blvd 41.9    -87.6 casual         19032
## 8 Wells St & Concord Ln     41.9    -87.6 casual         17656
## 9 Clark St & Lincoln Ave    41.9    -87.6 casual         16284
## 10 Wabash Ave & Grand Ave   41.9    -87.6 casual         16068
## # i 764 more rows
```

```
sum(end_station_map_casual$num_rides)
```

```
## [1] 1956744
```

## Calculating total rides of end stations for all users

```
end_station_map <- end_station_map_casual %>%  
  union(end_station_map_member)  
arrange(end_station_map, desc(num_rides))
```

```
## # A tibble: 1,528 × 5  
## # Groups:   end_station_name [776]  
##   end_station_name      end_lat end_lng customer_type num_rides  
##   <chr>              <dbl>   <dbl> <chr>          <int>  
## 1 Streeter Dr & Grand Ave    41.9   -87.6 casual        62556  
## 2 Millennium Park           41.9   -87.6 casual        32285  
## 3 Michigan Ave & Oak St      41.9   -87.6 casual        29689  
## 4 Clark St & Elm St          41.9   -87.6 member        23484  
## 5 Theater on the Lake        41.9   -87.6 casual        22753  
## 6 Wells St & Concord Ln       41.9   -87.6 member        21886  
## 7 Lake Shore Dr & Monroe St   41.9   -87.6 casual        21606  
## 8 Kingsbury St & Kinzie St    41.9   -87.6 member        20723  
## 9 Shedd Aquarium            41.9   -87.6 casual        19624  
## 10 Wells St & Elm St          41.9   -87.6 member        19564  
## # i 1,518 more rows
```

```
sum(subset(end_station_map, customer_type == "member")$num_rides)
```

```
## [1] 2386306
```

```
sum(subset(end_station_map, customer_type == "casual")$num_rides)
```

```
## [1] 1956744
```

Among the top ten stations, six are conveniently located next to public parks, three are situated in front of sizable residential buildings, and two are in close proximity to fitness centers.

## Creating bins start\_station\_map

```
end_bins <- seq(0, 63000, by = 1000)
```

## Assigning color to show ride density per station.

```
density_color_end <- colorBin(palette = "magma", domain = end_station_map$num_rides, n  
a.color = "transparent", bins = end_bins, reverse = T)
```

## Setting text for interactive tooltip

```
end_map_text <- paste("station name: ", end_station_map$end_station_name, "<br/>", "num  
ber of rides: ", end_station_map$num_rides, "<br/>", "customer type: ", end_station_map  
$customer_type, sep = "") %>%  
  lapply(htmltools::HTML)
```

```

tag.map.title_end <- htmltools::tags$style(htmltools::HTML("
  .leaflet-control-map-title {
    transform: translate(-50%,20%);
    position: fixed !important;
    left: 25%;
    text-align: center;
    padding-left: 10px;
    padding-right: 10px;
    background: rgba(255,255,255,0.75);
    font-weight: bold;
    font-size: 18px;
    color: blue
  }
"))

title_end <- htmltools::tags$div(
  tag.map.title, htmltools::HTML("Popular End Stations")
)

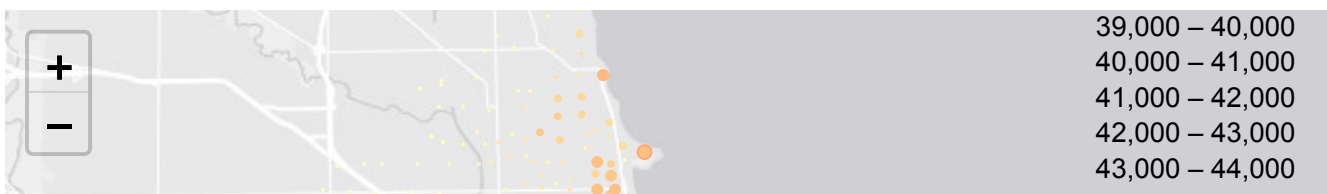
```

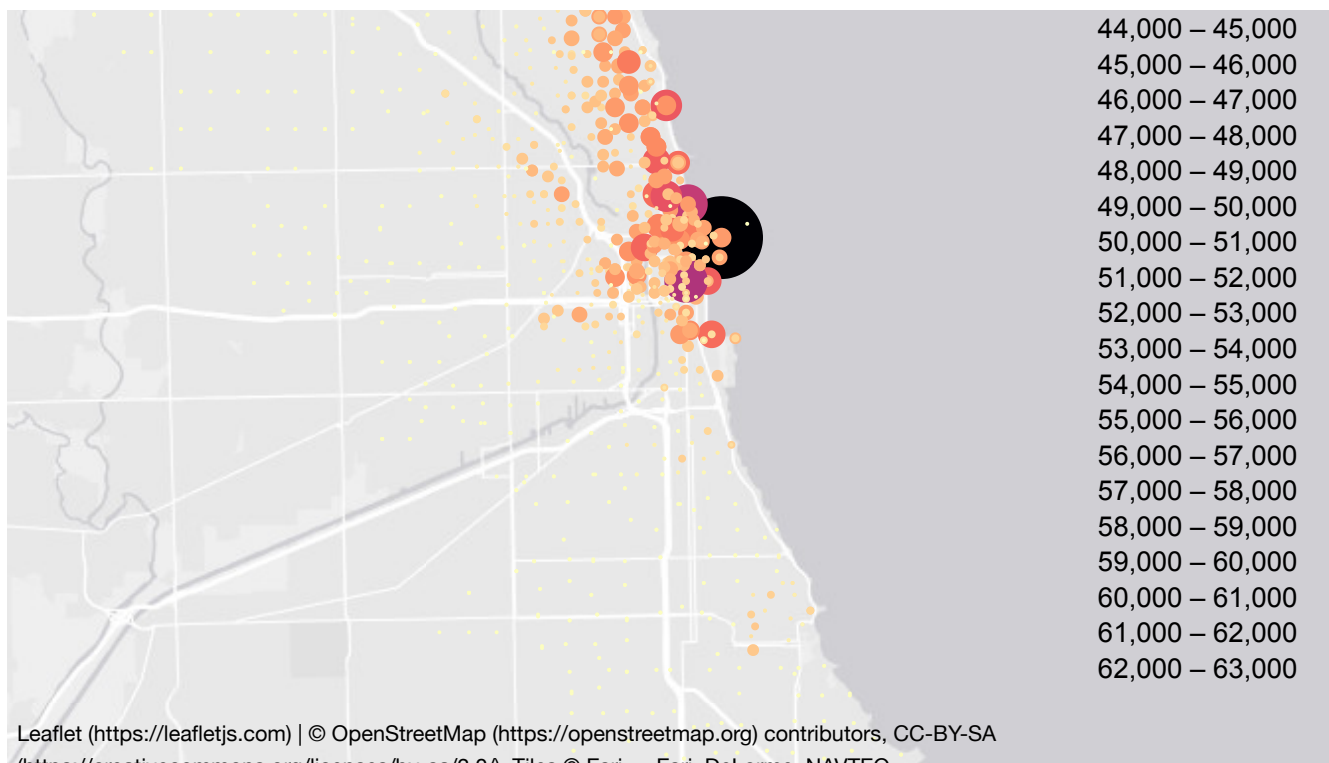
## Creating interactive html leaflet widget to show end station RIDE density for all users

```

rides_end_station <- leaflet(end_station_map) %>%
  leaflet::addTiles() %>%
  leaflet::setView(lng = -87.6298, lat = 41.8781, zoom = 10.5) %>% # Chicago coordinates
  leaflet::addProviderTiles("Esri.WorldGrayCanvas") %>% # set map style
  leaflet::addCircleMarkers(~ end_lng, ~ end_lat, # add circle markers for each station, set fill color, add tooltip.
    fillColor = ~ density_color_end(num_rides),
    fillOpacity = 1,
    color = "white",
    # radius = ~ start_station_map$num_rides/1500, #this is an option for variable marker size based on number of rides.
    radius = end_station_map$num_rides/3000,
    stroke = FALSE,
    label = end_map_text,
    labelOptions = labelOptions(style = list("font-weight" = "normal", padding = "3px 8px"),
    textsize = "13px",
    direction = "auto")) %>%
  #add legend.
  leaflet::addLegend(
    pal = density_color_end,
    values = ~ num_rides,
    opacity = 0.8,
    title = "Number of Rides",
    position = "bottomright") %>%
  leaflet::addControl(title_end, position = "topleft", className="map-title")
rides_end_station

```





The end station that stands out as the most popular is also Streeter Dr & Grand Ave.

## Assigning color to show ride density per station.

```
density_color_end_member <- colorBin(palette = "magma", domain = end_station_map_member$
$num_rides, na.color = "transparent", bins = end_bins, reverse = T)
```

```
density_color_end_casual <- colorBin(palette = "magma", domain = end_station_map_casual$
$num_rides, na.color = "transparent", bins = end_bins, reverse = T)
```

## Setting text for interactive tooltip

```
end_map_text_member <- paste("station name: ", end_station_map_member$end_station_name,
"<br/>", "number of rides: ", end_station_map_member$num_rides, sep = "") %>%
  lapply(htmltools::HTML)
```

```
end_map_text_casual <- paste("station name: ", end_station_map_casual$end_station_name,
"<br/>", "number of rides: ", end_station_map_casual$num_rides, sep = "") %>%
  lapply(htmltools::HTML)
```

## Setting map title for member users

```

tag.member_end.title <- htmltools::tags$style(htmltools::HTML("
  .leaflet-control.map-title {
    transform: translate(-50%,20%);
    position: fixed !important;
    left: 25%;
    text-align: center;
    padding-left: 10px;
    padding-right: 10px;
    background: rgba(255,255,255,0.75);
    font-weight: bold;
    font-size: 18px;
    color: blue
  }
"))

member_end_map_title <- htmltools::tags$div(
  tag.member_end.title, htmltools::HTML("Popular End Stations (member)")
)

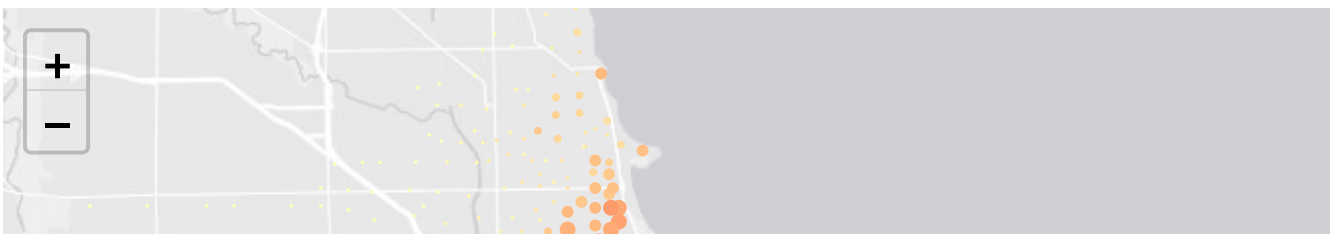
```

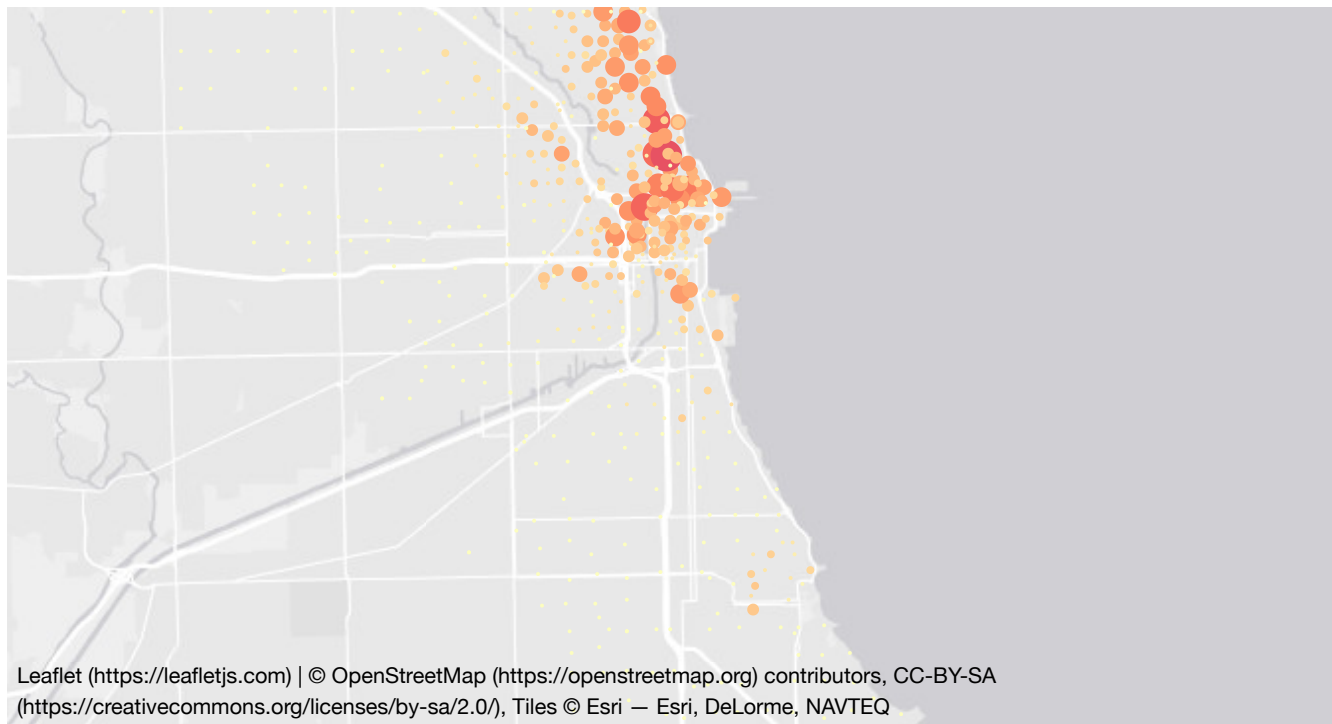
## Creating interactive html leaflet widget to show RIDE density for member users

```

rides_end_station_member <- leaflet(end_station_map_member) %>%
  leaflet::addTiles() %>%
  leaflet::setView(lng = -87.6298, lat = 41.8781, zoom = 10.5) %>% # Chicago coordinates
  leaflet::addProviderTiles("Esri.WorldGrayCanvas") %>% # set map style
  leaflet::addCircleMarkers(~ end_lng, ~ end_lat, # add circle markers for each station, set fill color, add tooltip.
    fillColor = ~ density_color_end_member(num_rides),
    fillOpacity = 1,
    color = "white",
    # radius = ~ start_station_map$num_rides/1500, #this is an option for variable marker size based on number of rides.
    radius = end_station_map_member$num_rides/3000,
    stroke = FALSE,
    label = end_map_text_member,
    labelOptions = labelOptions(style = list("font-weight" = "normal", padding = "3px 8px"),
    textsize = "13px",
    direction = "auto"))
#addControl(member_end_map_title, position = "topleft", className="map-title")
#add legend.
#addLegend(
  # pal = density_color_end_member,
  # values = ~ num_rides,
  # opacity = 0.8,
  # title = "Number of Rides",
  # position = "bottomright")
rides_end_station_member

```





## Setting map title for casual users

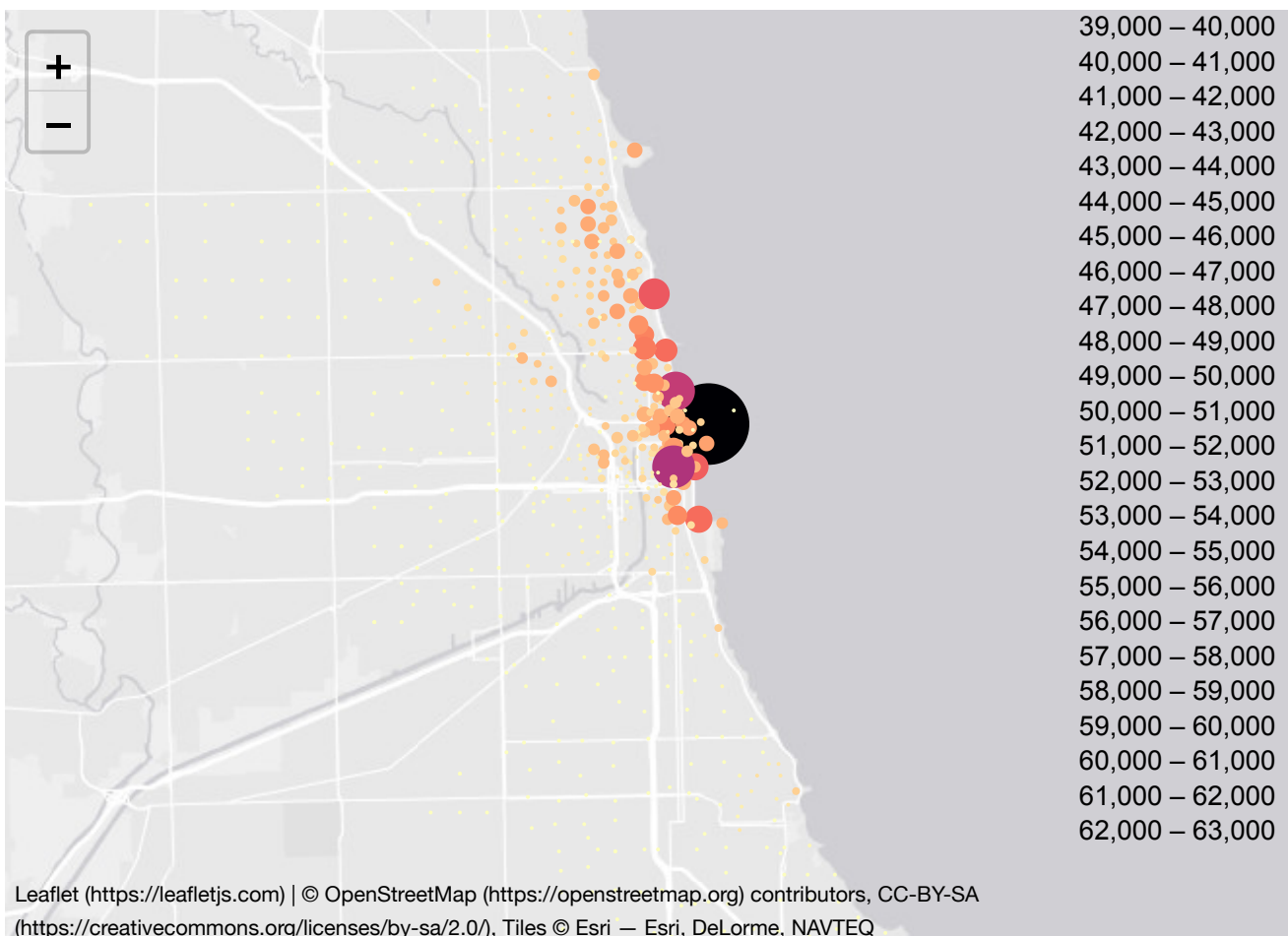
```
tag.casual_end.title <- htmltools::tags$style(htmltools::HTML("
  .leaflet-control-map-title {
    transform: translate(-50%,20%);
    position: fixed !important;
    left: 60%;
    text-align: center;
    padding-left: 10px;
    padding-right: 10px;
    background: rgba(255,255,255,0.75);
    font-weight: bold;
    font-size: 18px;
    color: blue
  }
"))

casual_end_map_title <- htmltools::tags$div(
  tag.casual_end.title, htmltools::HTML("Casual End Stations")
)
```

## Creating interactive html leaflet widget to show RIDE density for casual users

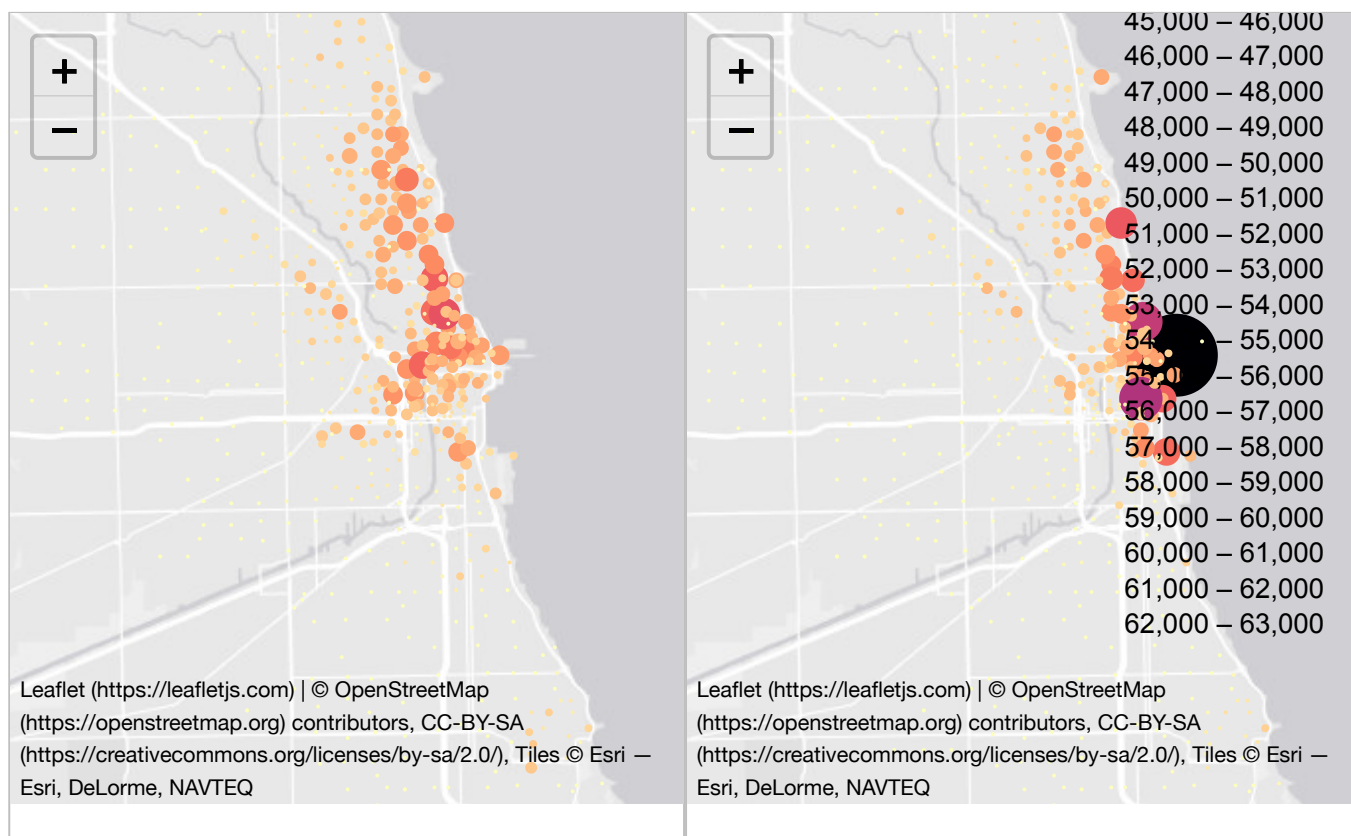


```
rides_end_station_casual <- leaflet(end_station_map_casual) %>%
  leaflet::addTiles() %>%
  leaflet::setView(lng = -87.6298, lat = 41.8781, zoom = 10.5) %>% # Chicago coordinates
  leaflet::addProviderTiles("Esri.WorldGrayCanvas") %>% # set map style
  leaflet::addCircleMarkers(~ end_lng, ~ end_lat, # add circle markers for each station, set fill color, add tooltip.
    fillColor = ~ density_color_end_casual(num_rides),
    fillOpacity = 1,
    color = "white",
    # radius = ~ start_station_map$num_rides/1500, #this is an option for variable marker size based on number of rides.
    radius = end_station_map_casual$num_rides/3000,
    stroke = FALSE,
    label = end_map_text_casual,
    labelOptions = labelOptions(style = list("font-weight" = "normal", padding = "3px 8px"),
    textsize = "13px",
    direction = "auto")) %>%
  #add legend.
  leaflet::addLegend(
    pal = density_color_end_casual,
    values = ~ num_rides,
    opacity = 0.8,
    title = "Number of Rides",
    position = "bottomright") %>%
  leaflet::addControl(casual_end_map_title, position = "topleft", className="map-title")
rides_end_station_casual
```



```
sync_maps_end <- sync(rides_end_station_member, rides_end_station_casual)
sync_maps_end
```

Rides End Stations



For the ended stations, we have similar findings. Casual riders tend to end their trip near museums and other attraction sites while members often end their journey close to universities, and residential and commercial areas.

The thirteen hypothesis, **H13: members' starting and ending locations are clustered in the downtown area, while casual customers' locations are more dispersed and along the beach**, was supported.

## Finding Summary

Through the data analysis, we can confidently say that members and casuals are two different customer groups.

Firstly, it's important to recognize that members and casual customers utilize Cyclistic bikes for distinct purposes. Members use the bikes for their daily commutes, whereas casual customers use them primarily for sightseeing around Chicago. Consequently, members exhibit higher bike usage frequency, with ride durations roughly half that of casual riders.

Furthermore, there are notable differences in the start and end locations of their journeys. Members typically initiate and conclude their trips in proximity to universities, residential areas, and commercial districts. On the other hand, casual customers tend to start and finish their rides near parks, museums, and coastal areas.

These variations in bike usage patterns, trip durations, and preferred locations should be taken into account while devising strategies to target and engage both member and casual customer segments effectively. It seems that converting casual customers to members will prove to be the most challenging task as both groups have totally different preferences. Although Cyclistic can attempt to develop a campaign to convert casual customers into members, the likelihood of achieving success seems rather low.

## Phase 5: Sharing

# Use Frequency

Based on the analysis, it is evident that casual riders predominantly use bike-sharing for leisure and tourism, with heightened activity during weekends. On the other hand, members primarily use bike-sharing for their daily work commute, showing increased activity on weekdays. Considering these findings, Cyclistic should explore offering new membership options tailored to weekend riders, family memberships (as families often spend weekends together), or partnerships with museums, theaters, and other popular locations frequently visited by casual riders.

# Usage Time

The data indicates that casual riders take significantly longer trips compared to members. To capitalize on this insight, Cyclistic can consider introducing bonuses or rewards for riders who take longer trips, encouraging extended usage.

# Seasonality

Bike usage peaks during June to August, coinciding with the summer season. With this in mind, Cyclistic should initiate marketing campaigns in spring, offering early-bird discounts for the new membership types. These campaigns should continue during the peak summer months to attract and retain customers.

# Phase 6: Acting

“Despite the challenges, all hope is not lost for Cyclistic, as the analysis results provide a foundation for several revenue-boosting actions. To address the initial questions more succinctly, we can rephrase them as, ‘How can Cyclistic increase its revenue based on the available data?’ Additionally, let’s not forget the last original question: ‘How can the marketing team leverage digital media to maximize the number of members?’

# Suggestion 1: Casual Users have longer ride durations.

Create targeted social media campaigns aimed at casual customers on weekends, for example, offering trial benefits to casual users with an average monthly ride duration of 25 minutes. Target users near the top 25 stations and run the campaign from mid-spring to mid-autumn, with special promotions for weekend trips and classic bike riders. Regarding digital media utilization, Cyclistic can craft more focused and effective campaigns, for instance, a ‘Weekend Experience with Cyclistic’ social media campaign could incentivize them to share their positive experiences for future discounts or lower membership fees. Encouraging resharing would expand reach and attract potential customers within a limited timeframe.

Furthermore, expansion opportunities could be explored by analyzing customers’ home addresses, identifying tourist hotspots. By establishing branches in these areas, Cyclistic can attract both local residents and tourists as members, enhancing daily commute and sightseeing experiences.

A more flexible pricing structure tailored to weekend users could also entice more customers to become members. A membership plan accommodating weekend-only bike usage would appeal to those seeking occasional rides.

## Suggestion 2: Members use Cyclistic for commuting to work.

Cyclistic could offer promotions to casual users who use Cyclistic for work commutes or create a corporate package for companies near the top 25 stations, providing Cyclistic rides as an employee benefit. The campaign should run from mid-spring to mid-autumn, targeting users near the top 25 stations, with special promotions for weekdays and electric and classic bike riders.

## Suggestion3: Round Trips

Recognize that members often take round trips, and many casual users may exhibit similar behavior but have not yet become members. Develop a campaign for users who use the service twice a day, offering a special promotion for becoming members. Target users near the top 25 stations and run the campaign from mid-spring to mid-autumn, focusing on weekdays and classic bike riders.

By implementing these strategies, Cyclistic can leverage the data insights to attract more members, enhance customer engagement, and ultimately increase revenue. Additionally, providing stakeholders with additional information such as user IDs, age, gender, and price could further refine the analysis and lead to even more targeted marketing efforts.

In conclusion, the Cyclistic case study demonstrates that casual customers and members constitute distinct customer groups, making conversion challenging. However, leveraging the available data, Cyclistic can implement targeted actions to maximize revenue and cater to both segments effectively.

Working on this capstone project has been an enriching experience, providing valuable insights into real-world data analysis, including data preparation and cleaning. Ensuring data reliability and integrity emerged as vital aspects before conducting the analysis. Additionally, visualizing diverse data metrics led to fascinating discoveries, informing sound business decisions.

Ultimately, the fulfillment lies in using data to guide decisions that enhance people's lives. I look forward to future opportunities to delve into further analytical pursuits. Until then, happy analyzing!"