Bike Ride Full Analysis

Egwolo Akpere

2022-08-06

Abstract

The work reported in this study explores and discuss extensively on a public dataset gotten from Divvy Tripdata. The study was preceded by an elaborate and up-to-date analysis using the R programming language.

Introduction

In this case study, i will perform many real-world tasks of a junior data analyst. I will work for a fictional company, Cyclistic, and meet different characters and team members. In order to answer the key business questions, i will follow the steps of the data analysis process: ask, prepare, process, analyze, share, and act.

About the Company

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. Riders who have an annual subscription are called members while riders who are **"single-ride"** or **"full-day"** pass users are considered casual riders. Cyclistic users are more likely to ride for leisure, but about 30% use them to commute to work each day.

Task

I am a junior data analyst working in the marketing analyst team at Cyclistic, a bike-share company in Chicago. The director of marketing believes the company's future success depends on maximizing the number of annual memberships. Therefore, my team wants to understand how casual riders and annual members use Cyclistic bikes differently. From these insights, my team will design a new marketing strategy to convert casual riders into annual members. But first, Cyclistic executives must approve my recommendations, so they must be backed up with compelling data insights and professional data visualizations.

Solution

The following data analysis steps will be followed:

- Ask
- Prepare
- Process
- Analyze
- Share
- Act.

STEP 1 (ASK)

- **What is the problem you are trying to solve?**

  At the end of this study, we will answer these three questions:

1. How do annual members and casual riders use Cyclistic bikes differently?
2. Why would casual riders buy Cyclistic annual memberships?
3. How can Cyclistic use digital media to influence casual riders to become members?

- **How can your insights drive business decisions?**

  If the aforementioned questions are answered correctly, insights from this analysis will inform marketing strategies aimed at converting casual riders into annual members which will be satisfactory to our Stakeholders Objectives.

- **Who are the key stakeholders?**

1. The primary stakeholders are: Cyclistic executive team and its Director of Marketing.
2. Secondary stakeholders are Cyclistic marketing analytics team.

STEP 2 (PREPARE)

For this step, we will be providing solutions to some questions.

- **Where is your data located?**

  I will be using the public dataset located here.

- **How is the data organized?**

  The Data set was downloaded locally. It was next unpacked from the Archives and arranged into in a single folder as a CSV file. All CSV files will be uploaded on R and then be appended in a single Data frame

- **Are there issues with bias or credibility in this data? Does your data ROCCC?**

  The dataset follows the ROCCC Analysis as described below:

1. Reliable - yes, dataset is not biased.

2. Original - yes, we can locate the original public data here
3. Comprehensive - yes, dataset does not miss any important information
4. Current - yes, dataset is updated monthly
5. Cited - yes, dataset is cited.

- **How are you addressing licensing, privacy, security, and accessibility?**

  The data has been made available by Motivate International Inc. under this license. It is a secured public data that is accessible.

- **How did you verify the data's integrity?**

  The data is consistent, accurate but it has few columns which are not required, hence cannot cause damage to our analysis.

- **How does it help you answer your question?**

  All information to enable us carryout this analysis accurately and provide accurate results are available, hence this will enable us answer our questions.

- **Are there any problems with the data?** There are no problems that can cause damage to our analysis.

Download data and store it appropriately

Previous 12 months data will be downloaded here and stored as .CSV using appropriate file-naming conventions.

Installing necessary packages to be used for our analysis

All necessary packages to enable us carefully explores and analyze extensively our data will be installed using the codes below;

```
#install.packages("Tidyverse")
#install.packages("lubridate")
#install.packages("ggplot2")
#install.packages("dplyr")
#install.packages("skimr")
#install.packages("readr")
#install.packages("janitor")
```

- Tidyverse package is for data importation and wrangling
- lubridate Package is for date functions
- ggplot2 package is for visualization
- dplyr package comprises many functions that perform mostly used data manipulation operations such as applying filter, selecting specific columns, sorting data, adding or deleting columns and aggregating data.
- Skimr is designed to provide summary statistics about variables in data frames, tibbles, data tables and vectors

- readr package helps to deal with reading in large flat files quickly
- janitor package has simple functions for examining and cleaning dirty data

Loading necessary packages to be used for our analysis

Immediately after installing the above packages, we load them to enable us access their built-in functions using the codes below;

```
#library(tidyverse)
#library(lubridate)
#library(ggplot2)
#library(dplyr)
#library(skimr)
#library(readr)
#library(janitor)
```

Importing data into Rstudio

Before importing data into Rstudio, it is important to know the working directory. Having your data in a wrong working directory will given an error feedback during data importation. We use the code below to display our working directory.

```
getwd()
```

```
## [1] "C:/Users/HP PC/Documents"
```

We can as well set our working directory to simplify calls to data using the code below;

```
setwd("C:/Users/HP PC/Documents")
```

After our working directory has been set correctly, we commence our data importation of all 12 .CSV files using the follow codes;

```
#Jan <- read.csv("Tripdata_2021_Aug_13th_11_16_49_Pm.csv")
#Feb <- read.csv("Tripdata_2021_Dec_8th_12_19_04_Pm.csv")
#Mar <- read.csv("Tripdata_2021_Nov_4th_12_58_36_Pm.csv")
#Apr <- read.csv("Tripdata_2021_Oct_4th_10_21_39_Am.csv")
#May <- read.csv("Tripdata_2021_Sep_8th_11_10_46_Am.csv")
#Jun <- read.csv("Tripdata_2022_Apr_6th_10_07_41_Am.csv")
#Jul <- read.csv("Tripdata_2022_Feb_2nd_08_55_22_Am.csv")
#Aug <- read.csv("Tripdata_2022_Jan_6th_08_18_45_Am.csv")
#Sep <- read.csv("Tripdata_2022_Jul_15th_08_27_59_Am.csv")
#Oct <- read.csv("Tripdata_2022_Jun_3rd_07_08_02_Pm.csv")
#Nov <- read.csv("Tripdata_2022_May_2nd_12_22_47_Pm.csv")
#Dec <- read.csv("Tripdata_2022_May_3rd_09_33_19_Am.csv")
```

Jan, Feb, Mar, …Dec as displayed above are labels chosen to name the imported files. These names could be anything desirable.

Check/compare Column names and View full Data

Next, we ensure column names of all data imported match perfectly so we do not get any error when joining them into one file. Afterwards we view to ensure that all data imported are accurate. This can be done using the following code;

```
#colnames(Jan)
#colnames(Feb)
#colnames(Mar)
#colnames(Apr)
#colnames(May)
#colnames(Jun)
#colnames(Jul)
#colnames(Aug)
#colnames(Sep)
#colnames(Oct)
#colnames(Nov)
#colnames(Dec)
```

The above codes are used to Check/compare column names of all data files.

```
#View(Jan)
#View(Feb)
#View(Mar)
#View(Apr)
#View(May)
#View(Jun)
#View(Jul)
#View(Aug)
#View(Sep)
#View(Oct)
#View(Nov)
#View(Dec)
```

The above codes are used to View the data frame of each data file imported.

Merging all 12 Data Files into a single Data Frame

We merge all 12 Data Files using the code below;

```
#General_tripdata <- rbind(Jan, Feb, Mar, Apr, May, Jun, Jul, Aug, Sep, Oc
t, Nov, Dec)
```

STEP 3 (PROCESS)

This step involves Cleaning Data for Analysis. Before we commence cleaning of our data, we inspect our data using the following;

```
#colnames(General_tripdata)
#nrow(General_tripdata)
#ncol(General_tripdata)
#table(General_tripdata)
#dim(General_tripdata)
#head(General_tripdata)
#tail(General_tripdata)
#str(General_tripdata)
#glimpse(General_tripdata)
#summary(General_tripdata)
#skim_without_charts(General_tripdata)
```

The above commands are used to view the content of our data frame.

- colnames(General_tripdata): DIsplays the list of column names.
- nrow(General_tripdata): Displays the number of rows in the data frame
- ncol(General_tripdata): Displays the number of rows in the data frame
- table(General_tripdata): Displays the categorical representation of data with variable name and the frequency in the form of a table
- dim(General_tripdata): Displays the dimensions of the data frame
- head(General_tripdata): Displays the first 6 rows of the data frame.
- tail(General_tripdata): Displays the last 6 rows of the data frame.
- str(General_tripdata): Displays the list of columns and data types(numeric,character,etc)
- glimpse(General_tripdata): DIsplays every column in the data frame. It's a little like str() applied to a data frame but it tries to show as much data as possible. summary(General_tripdata): Displays Statistical summary of data. Mainly for numerics.
- skim_without_charts(General_tripdata)is an alternative to summary(), quickly providing a broad overview of a data frame. It handles data of all types, dispatching a different set of summary functions based on the types of columns in the data frame.

Note; All the above command may not be used at once.

First, we check for duplicate rows and remove any if found. this can be done using the code below;

```
#nrow(distinct(General_tripdata)) == nrow(General_tripdata)
```

Next we check for cells with NA, afterwards, we remove NA if any and recheck to confirm. This can be done using the following codes below;

```
#sum(is.na(General_tripdata))
#General_tripdata <- na.omit(General_tripdata)
#sum(is.na(General_tripdata))
```

- sum(is.na(General_tripdata)): check and sum the number of NA in our Data Frame.
- General_tripdata <- na.omit(General_tripdata): Remove all NA in our Data Frame.

Next we Rename columns to make them consistent and easy to call. We do that using the following codes;

```
#General_tripdata <- General_tripdata %>%
#  rename(bike_id = "ride_id") %>%
#  rename(bike_type = "rideable_type") %>%
#  rename(user_type = "member_casual") %>%
#  rename(start_time = "started_at") %>%
#  rename(end_time = "ended_at")
```

Next we view our Data Frame to ensure the above has been effected correctly. This can be done using the following;

```
#View(General_tripdata)
```

Next we add columns that list the date, month, day, and year of each ride. This will allow us to aggregate ride data for each month, day, or year.

```
#General_tripdata$date <- as.Date(General_tripdata$start_time)
#General_tripdata$month <- format(General_tripdata$date, "%m")
#General_tripdata$day <- format(General_tripdata$date, "%d")
#General_tripdata$year <- format(General_tripdata$date, "%Y")
#General_tripdata$day_of_week <- format(General_tripdata$date, "%A")
```

It is interesting to note that days of the week may not be arranged in the appropriate form, i.e in ascending or descending order, Hence we use the following codes to arrange to our desired order.

```
#General_tripdata$day_of_week <- factor(General_tripdata$day_of_week, leve
ls = c
#                                       ("Sunday", "Monday", "Tuesday","W
ednesday","Thursday",
#                                         "Friday", "Saturday"))
```

Next we add Ride Length to our Data Frame, this can be done using the following;

```
#General_tripdata$ride_length <- difftime(General_tripdata$end_time,
#                                         General_tripdata$start_time, uni
ts = "mins")
```

Next we convert **"ride_length"** from Factor to numeric so we can run calculations on the data.Note that after conversion, values displays seconds. We do this using the following codes;

```
#is.factor(General_tripdata$ride_length)
#General_tripdata$ride_length <- as.numeric(as.character(General_tripdata$
ride_length))
#is.numeric(General_tripdata$ride_length)
```

Next we check for **"bad data"**. The dataframe includes a few hundred entries when bikes were taken out of docks and checked for quality or ride_length was negative. We use the following codes for this;

```
#table(General_tripdata$ride_length<=0)
#select(General_tripdata, ride_length, end_time, start_time, start_station
_name) %>%
#  filter(ride_length <= 0)
```

Next we delete our **"bad data"** using the following codes;

```
#updated_tripdata <- General_tripdata[!(General_tripdata$start_station_nam
e
#                                       == "HQ QR" | General_tripdata$ride
_length<=0),]
```

Next we confirm and view our data frame to ensure all bad data have been deleted. We do this using the following;

```
#table(updated_tripdata$ride_length<=0)
#View(updated_tripdata)
```

STEP 4 (ANALYZE)

This step involves conducting descriptive analysis. For this step, we will be carrying out the following analysis.

- Summary of Ride Length
- Percentage Ride Distribution
- Average Ride Length Vs Users
- Number of Rides Per Year
- Number of Rides Per Month
- Number of Rides Per Day
- Most Used Bike
- Most Used Bike By Users
- Top 20 Most Popular Start Station By Casual Users
- Top 20 Most Popular Start Station By Member Users
- Top 20 Most Popular End Station By Casual Users
- Top 20 Most Popular End Station By Member Users

Summary of Ride Length

Our first analysis will be to calculate the minimum, maximum, and mean ride length. This can be done using the following;

```
#summary_ride_length <- updated_tripdata %>%
#  summarise(min_ride_length = min(ride_length), max_ride_length = max(rid
e_length),
#            mean_ride_length = mean(ride_length), units= "mins")
#View(summary_ride_length)
```

From the above calculations we observe that;

- The min ride length is 0.01666667mins which is approximately 1sec.
- The max ride length is 49107.15mins which is approximately 34days.
- The mean ride length is 18.51667mins which is approximately 19mins.

Percentage Ride Distribution

Next we calculate the percentage ride distribution, we use the following codes;

```
#Percent_dist <- updated_tripdata %>%
#  group_by(user_type)%>%
#  summarise(n=n())%>%
#  mutate(percent = n*100/sum(n))
#View(Percent_dist)
```

From the above calculations we observe that;

- Percentage ride of Casual Users is 43% which is 2.6million rides.
- Percentage ride of Member Users is 57% which is 3.3million rides.

Average Ride Length Vs Users

Next we calculate the Average Ride Length Vs Users, we use the following codes;

```
#rides_per_length<-updated_tripdata %>%
#  group_by(user_type) %>%
#  summarise(number_of_ride = n(), avg_ride_length = mean(ride_length), .g
roups = "drop")
#View(rides_per_length)
```

From the above calculations we observe that;

- The average ride length of Casual Users is 26mins while the average ride length of member users is 13mins.
- The average ride length of casual users is a little more than twice the size of the average length of member users.

Number of Rides Per Year

Next we calculate the number of rides per year, we use the following codes;

```
#rides_per_year<-updated_tripdata %>%
#  group_by(user_type, year) %>%
#  summarise(number_of_ride = n(), avg_ride_length = mean(ride_length), .g
roups = "drop")
#View(rides_per_year)
```

From the above calculations we observe that;

- In 2021, there was a total of 1.7million rides by casual users with an average ride length of 27mins.
- In 2021, there was a total of 1.97million rides by member users with an average ride length of 13mins.
- In 2022, there was a total of 903thousand rides by casual users with an average ride length of 24mins
- In 2022, there was a total of 1.4million rides by member users with an average ride length of 13mins.

Number of Rides Per Month

Next we calculate the number of rides per month, we use the following codes;

```
#rides_per_month<-updated_tripdata %>%
#  group_by(user_type, month) %>%
#  summarise(number_of_ride = n(), avg_ride_length = mean(ride_length), .g
roups = "drop")
#View(rides_per_month)
```

From the above calculations we observe that;

- Casual users had the least number of ride in the month of January and highest in the month of July, while Member users had the least number of ride in the month of January and highest in the month of June.
- Both Casual users and Member users take more rides around May, June, July, August and September.
- Both Casual users and Member users take less rides around December, January, and February.

Number of Rides Per Day

Next we calculate the number of rides per day, we use the following codes;

```
#rides_per_day<-updated_tripdata %>%
#  group_by(user_type, day_of_week) %>%
#  summarise(number_of_ride = n(), avg_ride_length = mean(ride_length), .g
roups = "drop")
#View(rides_per_day)
```

From the above calculations we observe that;

- Casual users take more ride during weekends with a relatively fair number of rides during weekday, while Member users take more rides during weekdays with a relatively fair number of rides during weekend.

Most Used Bike

Next we calculate the most used bike, we use the following codes;

```
#most_used_bike<-updated_tripdata %>%
#  group_by(bike_type) %>%
#  summarise(number_of_ride = n(), avg_ride_length = mean(ride_length), .g
roups = "drop")
#View(most_used_bike)
```

From the above calculations we observe that;

- Classic bike is the most used bike with a total number of 3.2million rides and an average ride length of 18mins
- Electric bike is the 2nd most used bike with a total number of 2.5million rides and an average ride length of 15mins

- Docked bike is the least used bike with a total number of 252thousand rides and an average ride length of 63mins

Most Used Bike By Users

Next we calculate the number of rides per year, we use the following codes;

```
#most_used_user_bike<-updated_tripdata %>%
#  group_by(user_type, bike_type) %>%
# summarise(number_of_ride = n(), avg_ride_length = mean(ride_length), .gr
oups = "drop")
#View(most_used_user_bike)
```

From the above calculations we observe that;

- Member users with 1.97million rides and average ride length of 13mins use Classic bike more than Casual users with 1.2million rides and average ride length of 25mins.
- Member users with 1.4million rides and average ride length of 12mins use Electric bike more than Casual users with 1.1million rides and average ride length of 18mins.
- Docked bike is usually used by Casual users with 252 thousand rides and an average ride length of 63mins

Top 20 Most Popular Start Station By Casual Users

Next we calculate the top 20 most popular start station by Casual users, we use the following codes;

```
#casual_start_station<-updated_tripdata %>%
#  group_by(user_type, start_station_name) %>%
#  filter(user_type == "casual") %>%
#  summarise(number_of_ride = n(), .groups = "drop") %>%
#  arrange(desc(number_of_ride)) %>%
#  head(20)
#View(casual_start_station)
```

In the top 20 chart, the most popular start station by Casual users is Streeter Dr & Grand Ave and the least is Broadway & Barry Ave

Top 20 Most Popular Start Station By Member Users

Next we calculate the top 20 most popular start station by Member users, we use the following codes;

```
#member_start_station<-updated_tripdata %>%
#  group_by(user_type, start_station_name) %>%
#  filter(user_type == "member") %>%
#  summarise(number_of_ride = n(), .groups = "drop") %>%
#  arrange(desc(number_of_ride)) %>%
```

```
#  head(20)
#View(member_start_station)
```

In the top 20 chart, the most popular start station by Member users is Kingsbury St & Kinzie St and the least is Wilton Ave & Belmont Ave

Top 20 Most Popular End Station By Casual Users

Next we calculate the top 20 most popular end station by Casual users, we use the following codes;

```
#casual_end_station<-updated_tripdata %>%
#  group_by(user_type, end_station_name) %>%
#  filter(user_type == "casual") %>%
#  summarise(number_of_ride = n(), .groups = "drop") %>%
#  arrange(desc(number_of_ride)) %>%
#  head(20)
#View(casual_end_station)
```

In the top 20 chart, the most popular end station by Casual users is Streeter Dr & Grand Ave and the least is Clark St & Newport St

Top 20 Most Popular End Station By Member Users

Next we calculate the top 20 most popular end station by Member users, we use the following codes;

```
#member_end_station<- updated_tripdata %>%
#  group_by(user_type, end_station_name) %>%
#  filter(user_type == "member") %>%
#  summarise(number_of_ride = n(), .groups = "drop") %>%
#  arrange(desc(number_of_ride)) %>%
#  head(20)
#View(member_end_station)
```

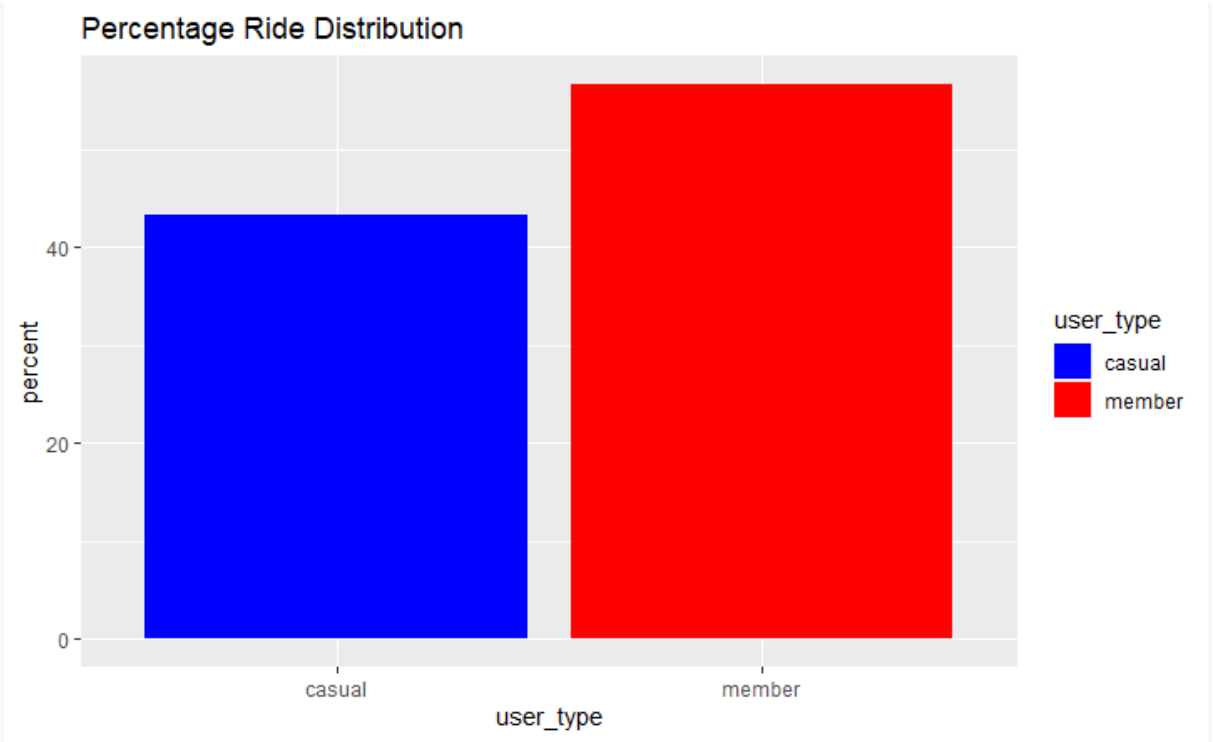
In the top 20 chart, the most popular end station by Member users is Kingsbury St & Kinzie St and the least is Sheffield Ave & Fullerton Ave

STEP 5 (SHARE)

Now it is time to share our findings through the art of visualization. To do this, we will plot some charts using ggplot.

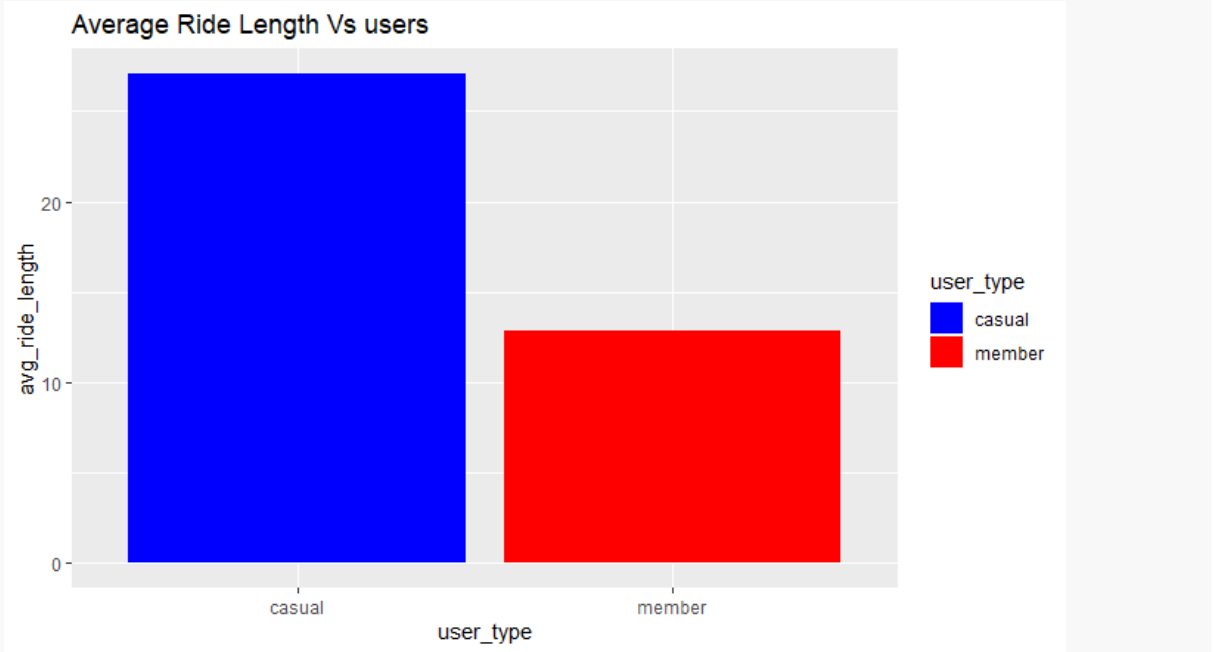
Percentage Ride Distribution

```
#ggplot(data=Percent_dist, aes(x=user_type, y= percent, fill = user_type))
+
#  geom_col(position = "dodge") +
#  scale_fill_manual(values = c("blue", "red"))+
# labs(title="Percentage Ride Distribution")
```



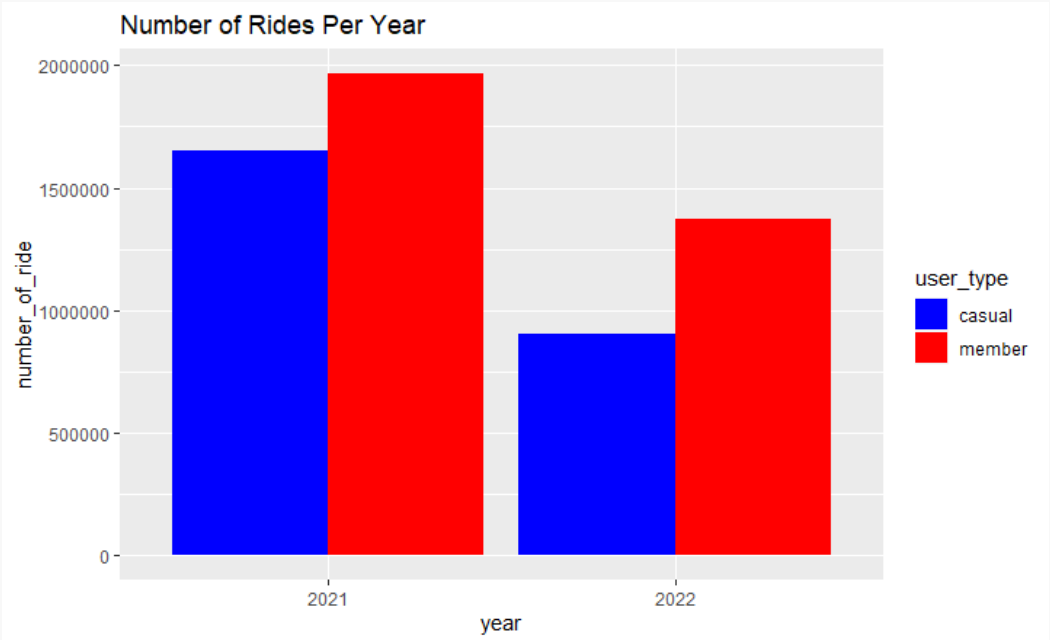
Average Ride Length Vs Users

```
#ggplot(data=rides_per_year, aes(x=user_type, y=avg_ride_length, fill = us
er_type)) +
#  geom_col(position = "dodge") +
#  scale_fill_manual(values = c("blue", "red"))+
#  labs(title="Average Ride Length Vs users")
```



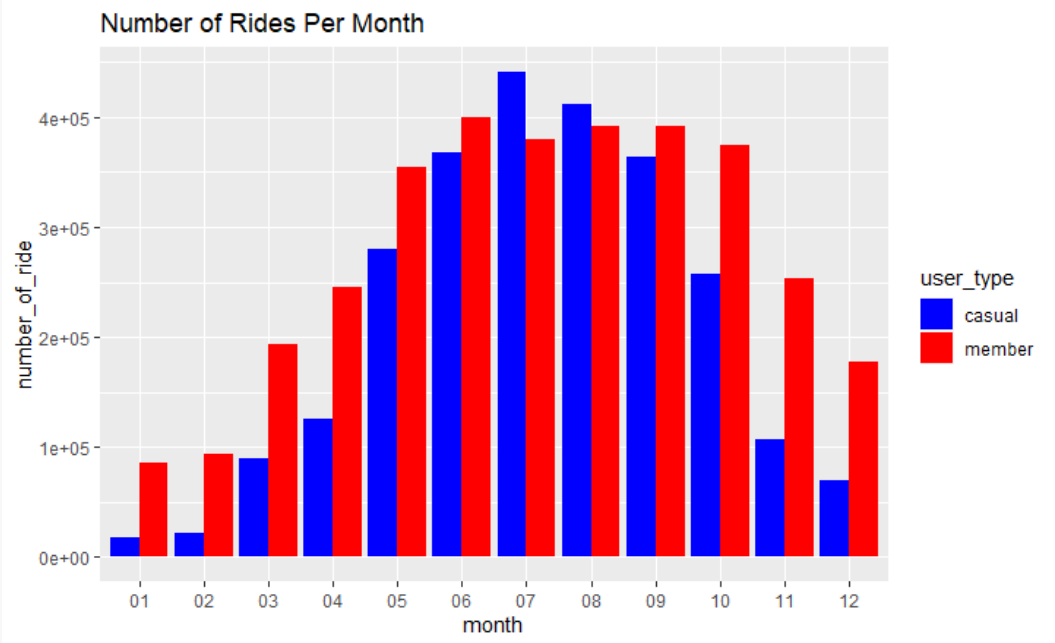
Number of Rides Per Year

```
#ggplot(data=rides_per_year, aes(x=year, y= number_of_ride, fill = user_ty
pe)) +
#  geom_col(position = "dodge") +
#  scale_fill_manual(values = c("blue", "red"))+
#  labs(title="Number of Rides Per Year")
```



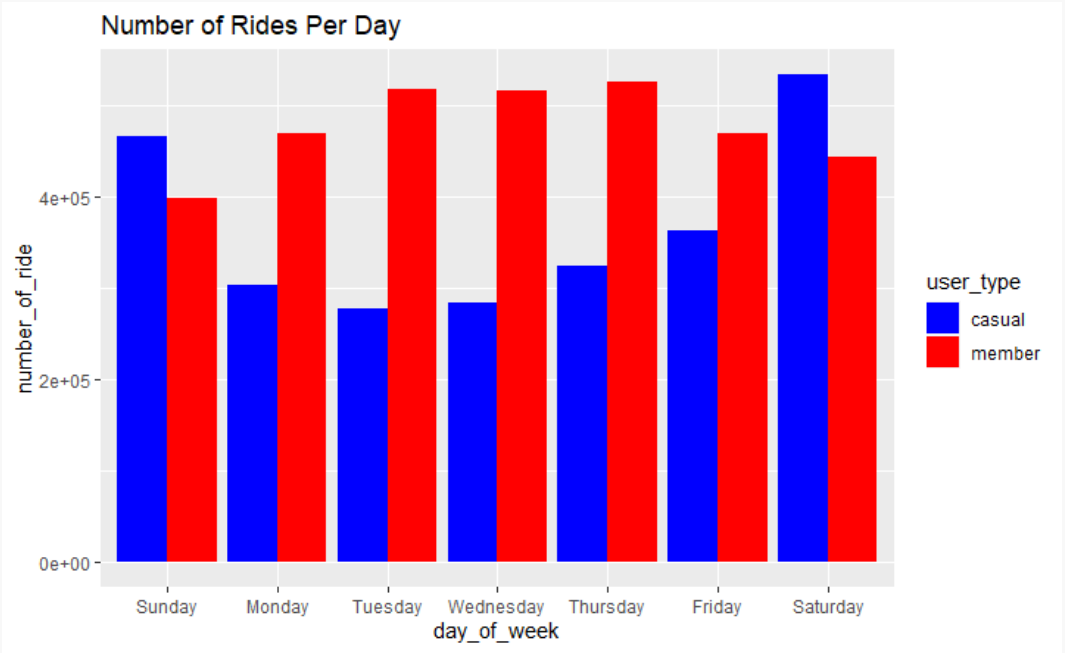
Number of Rides Per Month

```
#ggplot(data=rides_per_month, aes(x=month, y= number_of_ride, fill = user_
type)) +
#  geom_col(position = "dodge") +
#  scale_fill_manual(values = c("blue", "red"))+
#  labs(title="Number of Rides Per Month")
```



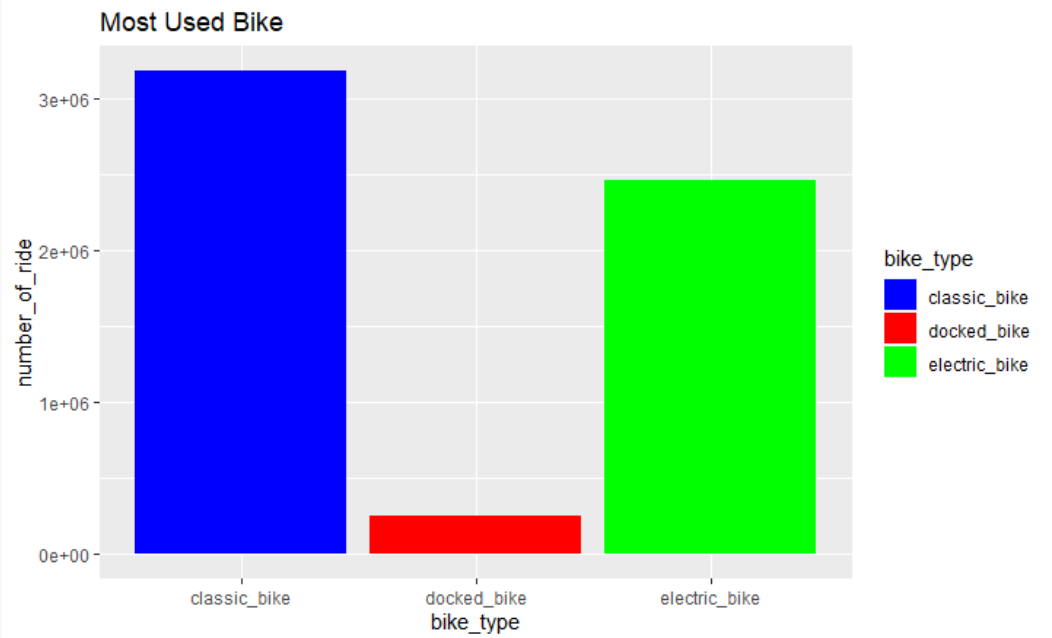
Number of Rides Per Day

```
#ggplot(data=rides_per_day, aes(x=day_of_week, y= number_of_ride, fill = u
ser_type)) +
#  geom_col(position = "dodge") +
#  scale_fill_manual(values = c("blue", "red"))+
#  labs(title="Number of Rides Per Day")
```



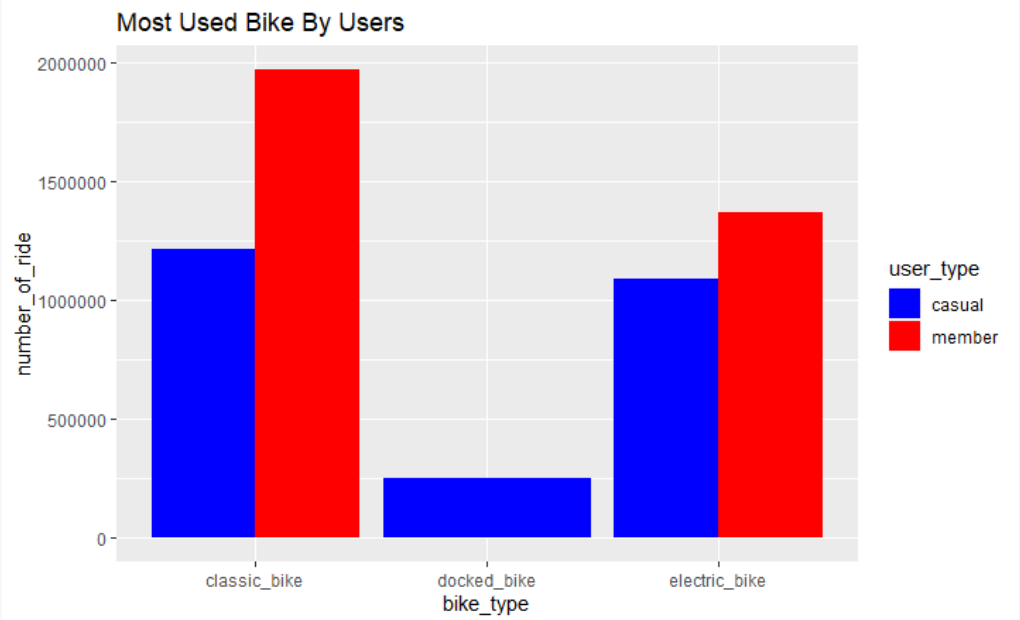
Most Used Bike

```
#ggplot(data=most_used_bike, aes(x=bike_type, y= number_of_ride, fill = bi
ke_type)) +
#  geom_col(position = "dodge")+
#  scale_fill_manual(values = c("blue", "red", "Green"))+
#  labs(title="Most Used Bike")
```



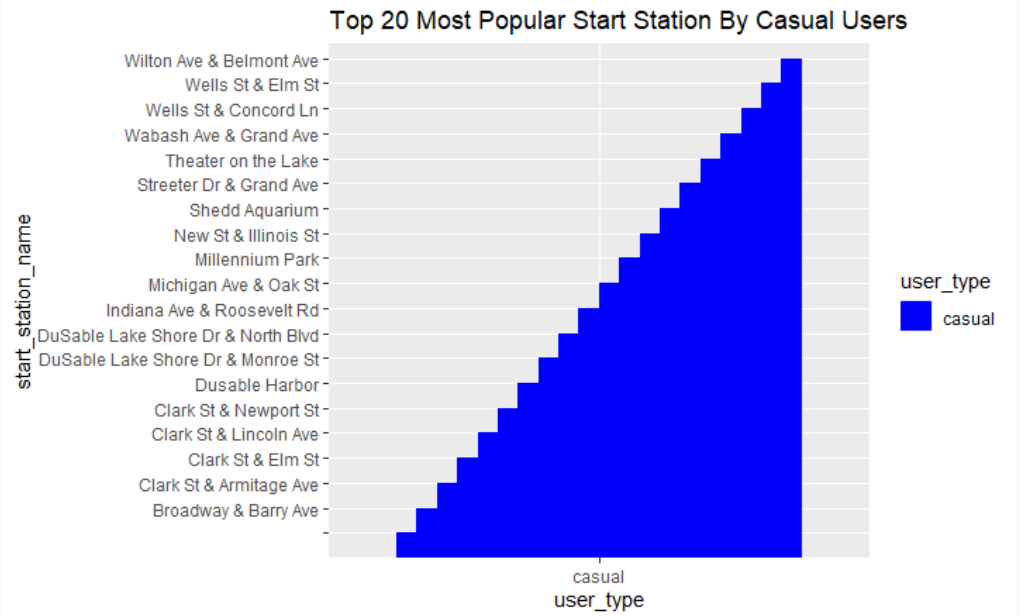
Most Used Bike By Users

```
#ggplot(data=most_used_user_bike, aes(x=bike_type, y= number_of_ride, fill
= user_type)) +
#  geom_col(position = "dodge")+
#  scale_fill_manual(values = c("blue", "red"))+
#  labs(title="Most Used Bike By Users")
```



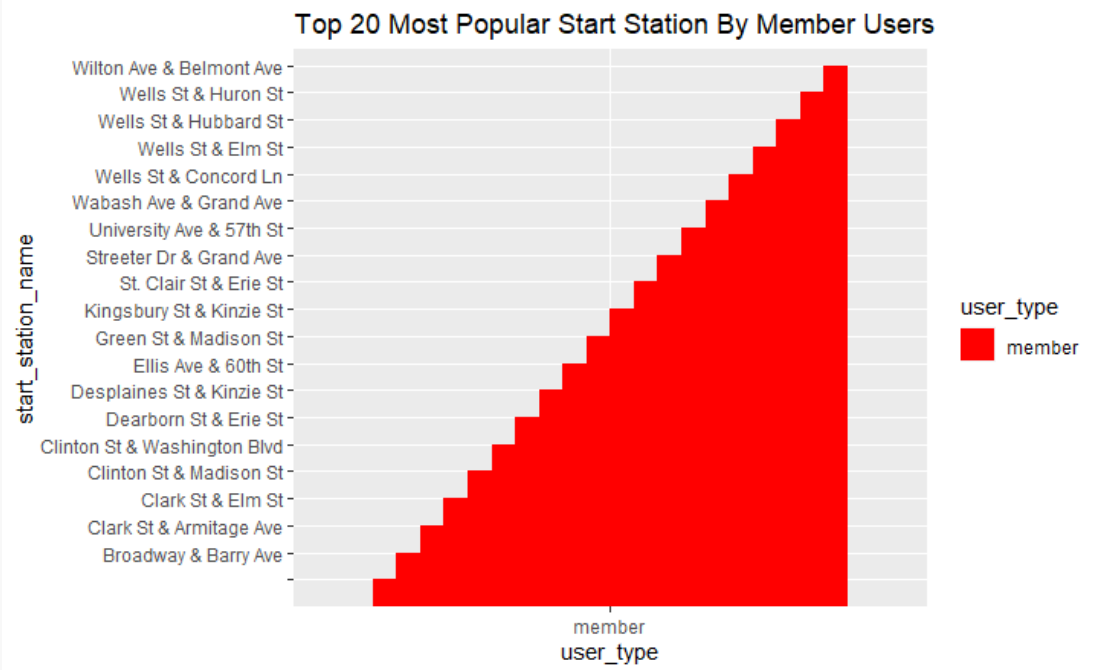
Top 20 Most Popular Start Station By Casual Users

```
#ggplot(data=casual_start_station,aes(x=user_type, y= start_station_name,
fill = user_type)) +
#  geom_col(position = "dodge")+
#  scale_fill_manual(values = c("blue"))+
#  labs(title="Top 20 Most Popular Start Station By Casual Users")
```



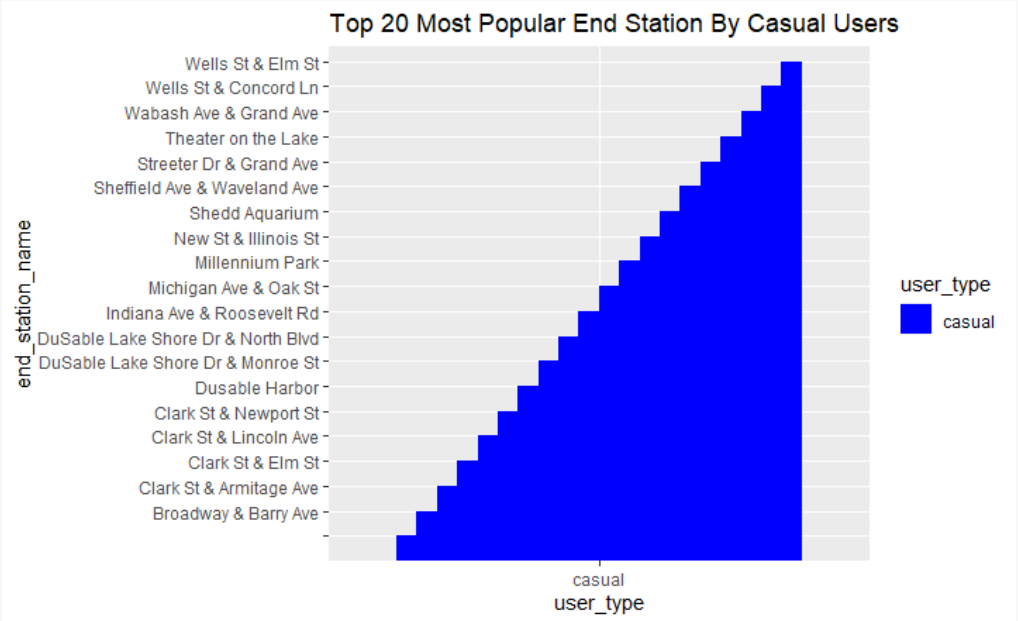
Top 20 Most Popular Start Station By Member Users

```
#ggplot(data=member_start_station, aes(x=user_type, y= start_station_name,
fill = user_type,))+
#  geom_col(position = "dodge")+
#  scale_fill_manual(values = c("red"))+
# labs(title="Top 20 Most Popular Start Station By Member Users")
```



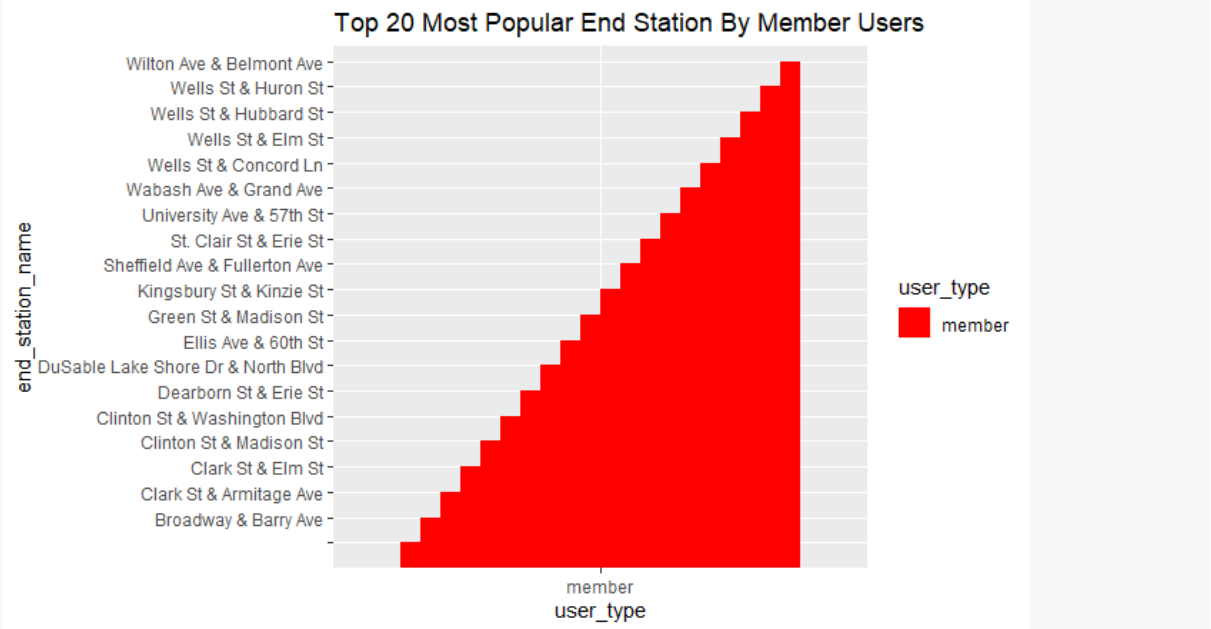
Top 20 Most Popular End Station By Casual Users

```
#ggplot(data=casual_end_station, aes(x=user_type, y= end_station_name, fil
l = user_type)) +
#  geom_col(position = "dodge")+
#  scale_fill_manual(values = c("blue"))+
#  labs(title="Top 20 Most Popular End Station By Casual Users")
```



Top 20 Most Popular End Station By Member Users

```
#ggplot(data=member_end_station, aes(x=user_type, y= end_station_name, fil
l = user_type,)) +
#  geom_col(position = "dodge")+
#  scale_fill_manual(values = c("red"))+
#  labs(title="Top 20 Most Popular End Station By Member Users")
```



The above charts are graphical visualization of the descriptive analysis found in Step 4.

Summary

- Percentage ride of Casual Users is 43.3% which is 2.6million rides while the Percentage ride of Member Users is 56.7% which is 3.3million rides. This clearly state that Member Users take more rides than Casual users.
- The average ride length of Casual Users is 26mins while the average ride length of member users is 13mins. The average ride length of casual users is a little above twice the size of the average length of member users. This clearly states that Casual users ride longer than Member users.
- Casual users take more ride during weekends with a relatively fair number of rides during weekday, while Member users take more rides during weekdays with a relatively fair number of rides during weekend.

STEP 6 (ACT)

Recommendations

- Since the average ride length of Casual users is relatively high, i.e. they spend more time riding, I suggest most, if not all casual users should subscribe to the member user package, this will enable them save more, as annual membership comes with discount and some other special packages.

- Digital media such as online ads and electronic billboards should be placed on Saturday and Sundays at Start and End stations with the highest casual users to influence casual riders to become member Users.