Cyclistic Bike-Share Analysis: Achieving Speedy Success

A Google Data Analytics Professional Certification Capstone Project

Document Version 1.0
May 2023

This is a working document. I may update certain portions of this documentation as soon as relevant information becomes available. Additionally, the level and depth of analysis provided herein reflect my level of competence as a data analyst at the time of publication. I will endeavor to make corrections or improvements in this document as my competence in working with the analytic tools used develops over time.

For questions, concerns, or suggestions, you may reach me through my email address at info.ea001@gmail.com

Contents

Introduction and Company Overview

This paper is a documentation of the steps that were taken to analyze user-data for Cyclistic – a fictional bike-sharing company operating within the vicinity of Chicago. This activity is part of the completion requirements for Google's Data Analytics Professional Certification. In this exercise, I assumed the role of a junior data analyst working under the marketing analytics team of Cyclistic.

The company began its operations in 2016, and since then, it has grown to a fleet of about 5,800 geo-tracked bikes within a network of almost 700 docking stations across Chicago. Cyclistic's approach to marketing has conventionally relied on building general awareness by appealing to a broad segment of consumers. The company established its place in the market by offering a flexibility scheme in its pricing plan. Customers have the option to buy (1) single ride passes, (2) full-day passes, or (3) annual memberships, depending on their specific mobility needs at the time. Cognizant of nuanced demands within the customer segment that they currently serve, Cyclistic also offers assistive options such as reclining bikes, hand tricycles, and cargo bikes, in addition to traditional bikes (classic or electric) to their clients.

Supported by the analysis of the Finance analytics team, the director of marketing believes that the way to bolster and secure growth in the future lies in maximizing the number of annual customers. To achieve this, the director of marketing wants to explore the option of growing annual memberships internally, i.e., converting current casual riders into annual members, rather than acquiring customers that are currently not part of the Cyclistic ecosystem.

Project Details and Objectives

Business Task

The goal of the project is clear: to help the marketing analytics team understand the differences between annual and casual members by analyzing Cyclistic's historical bike trip data to identify relevant trends and insights. This project will also examine the reasons why casual riders might want to buy annual memberships and how the company's marketing tactics may be affected by digital media.

Stakeholders

An integral component of Google's framework for data analysis is that the data analyst must have a clear understanding of his / her project's audience. The audience, and by extension, the stakeholders, determine the tenor of the analysis and of the presentation. With this, both analysis and presentation are approached uniquely based on who the identified stakeholders for any given project are. In the case of the project at hand, three (3) key stakeholders are of note:

- *Lily Moreno* – The Director of Marketing (DOM) and my manager for this project. She is the point person assigned for the development of campaigns and initiatives that seek to promote Cyclistic's bike-share program, including email, social media, and other pertinent channels.

- *The Marketing Analytics Team* – A team of data analysts tasked with the collection, analysis, and reporting of data that helps inform Cyclistic's marketing strategy.

- *The Executive Team* – A notoriously detail-oriented team of top-level executives that has the deciding say on whether or not to approve the recommended marketing program.

Data Preparation

Data Source

The data for this project was provided by Motivate International Inc. under this license. The scope of the analysis was limited to the past 12 months (May 2022 – April 2023). The dataset was downloaded through this link as individual zip files. After extraction, the filenames were retained for all 12 csv files, and were stored within a main directory in my local device for future reference.

Data Limitations

This project makes use of regularly updated datasets collected and made publicly available by the partner company. Owing to prevailing laws and statutes on data privacy, personally-identifiable details are intentionally blotted out by the data source. As a consequence, the analysis only answers questions pertaining to general consumer behavior accounted for by historical data. It cannot provide specific insights on the consumer base since the dataset does not have information on the current number of unique customers, the rate at which each segment is growing, or specific statistics that can identify trends relating to changes in mobility needs of the customer base. As useful as it might have been, the dataset also cannot provide information on whether or not users are located within the vicinity of Cyclistic's service area.

Data Integrity

Google's approach to checking data integrity follows the ROCCC framework. This test was adopted in this project to assess the integrity of our dataset and flag issues early on, if need be. The following items were considered:

- Reliable: While I can conclude that the individual csv files downloaded for this project are consistently structured and organized, the same cannot be said regarding the veracity of the information they hold. Since the project makes use of publicly-available data, accuracy, in this instance, is a presumption that inheres upon good faith, i.e., that the main data source provider has taken relevant steps, not only in making the datasets accessible, but also in trying to protect and maintain its integrity.

- Original: The dataset for this project falls under the *secondary data* category. The data is collected and made public by the partner company for the use of data analysts and researchers at large. The data source can also be easily validated and confirmed through the license agreement provided.

- Comprehensive: In all, each csv file contains 13 columns that can sufficiently answer the business tasks established for this project. The dataset is as comprehensive as it can get without transgressing prevailing laws on data privacy.

- Current: The main repository for the datasets in this project updates its contents on a monthly basis. For this reason, the relevance or currency of the data can be confirmed.

- Cited: The partner organization (Motive International Inc.) is named as the provider of the datasets for this project. Per the license agreement presented, the information supplied in our dataset is part of the Divvy system data owned by the City of Chicago's bicycle sharing service. The consent is granted specifically to Lyft Bikes and Scooters, LLC, which operates the City's bicycle sharing service.

Data Pre-Processing

Tools Used in the Project

This project made use of three programs, namely: MS Excel, R, and MS PowerPoint. The preliminary screening of the datasets was carried out using Excel. However, owing to Excel's limitations on the volume of data it can accommodate, majority of the processing and analysis were achieved using R. Finally, a presentation to the stakeholders of the insights and trends revealed throughout the project was created using MS PowerPoint.

Overview of Dataset

As a preliminary inspection of the dataset for this project, the csv file for May 2022 was loaded to MS Excel. The spreadsheet had 13 columns and had a total of 634,858 rows. Subject to further confirmatory tests, assumptions were held to describe the contents of each column. The May 2022 csv file also served as the benchmark for consistency tests applied to the rest of the dataset later in the project.

Table 1. Dataset Column Names and Descriptions

Column	*Description*	*Column*	*Description*
ride_id	String entries that identify every ride session accounted for by the system.	end_station_name	Station names and corresponding station IDs where session ended.
rideable_type	Classifies the kind of bicycle used for the session: classic, electric, or docked	end_station_id	
started_at	Datetime entries that accounts for when a session starts and when a session ends. Values are stored in the following format: *yyyy-mm-dd hh:mm:ss*	start_lat start_lng	Geographic coordinates where session began.

ended_at		end_lat
		end_lng
			Geographic coordinates where session ended.

start_station_name	Station names and corresponding station IDs where session began.	member_casual	Identifies whether session was traceable to an annual member or a casual user.
start_station_id			

Data Cleaning & Data Wrangling

In order to deliver on the business tasks of the project, a data frame that consolidates all 12 csv files had to be created. To achieve this, certain procedures were undertaken to ensure the consistency, cleanliness, and usability of the csv files. To begin, libraries that were deemed to be useful in the analysis were loaded into the R session.

```r
library(tidyverse)  #Data Analysis
library(dplyr)  #Data Manipulation
library(janitor)  #Data Cleaning
library(skimr)  #Data Summary
library(here)  #File Referencing
library(lubridate)  #Datetime Analysis
library(tmap)  #plotting coordinate points
library(leaflet)  #plotting coordinate points
library(geosphere)  #Geo-spatial data
library(ggplot2)  #DataViz
library(data.table)  #Computational-efficiency for data processing
library(tibble)  #Data frame simplification
library(hms)  #Manipulation of time-related variables
library(knitr)  #For attaching figures and illustrations
library(scales)  #Scale Functions for DataViz
library(formatR)  #Proper Formatting of R Code
```

With the libraries now loaded into R, the working directory was then assigned into the environment.

```r
setwd("D:/Data Analytics/Portflio/01 - Cyclistic/Cyclistic Data")
```

After which, the 12 csv files representing customer data for May 2022 through April 2023 were loaded using the *read.csv* function. The *month_year* format was the adopted naming convention for this project.

```
may_2022 <- read.csv("202205-divvy-tripdata.csv")
jun_2022 <- read.csv("202206-divvy-tripdata.csv")
jul_2022 <- read.csv("202207-divvy-tripdata.csv")
aug_2022 <- read.csv("202208-divvy-tripdata.csv")
sep_2022 <- read.csv("202209-divvy-tripdata.csv")
oct_2022 <- read.csv("202210-divvy-tripdata.csv")
nov_2022 <- read.csv("202211-divvy-tripdata.csv")
dec_2022 <- read.csv("202212-divvy-tripdata.csv")
jan_2023 <- read.csv("202301-divvy-tripdata.csv")
feb_2023 <- read.csv("202302-divvy-tripdata.csv")
mar_2023 <- read.csv("202303-divvy-tripdata.csv")
apr_2023 <- read.csv("202304-divvy-tripdata.csv")
```

The next step is to create a data frame that consolidates all of the loaded csv files via data merging. To do that, however, the files had to be examined first for consistency, particularly in the naming of all 13 columns and in the data types stored in each column of each csv file. This data integrity test was paramount in order to avoid further complications down the line.

```
# We start by creating a list of the loaded csv files

csv_files <- list(may_2022, jun_2022, jul_2022, aug_2022, sep_2022,
    oct_2022, nov_2022, dec_2022, jan_2023, feb_2023, mar_2023,
    apr_2023)

# We will use the column names of the first csv file (may_2022) as our benchmark

first_file_col <- colnames(csv_files[[1]])

# After which, we will now check if the column names are consistent

all_same_col <- all(sapply(csv_files[-1], function(file) identical(colnames(file),
    first_file_col)))

# Lastly, we want to see the result of the foregoing test. We will ask R to print the
result.

if (all_same_col) {
    print("The column names for all 12 csv files are IDENTICAL")
} else {
    print("The column names for all 12 csv files are NOT IDENTICAL ")
}

## [1] "The column names for all 12 csv files are IDENTICAL"
```

The code confirms that the column names for the loaded csv files are identical. The same test was performed to check for consistency in stored data type.

```
# We start by loading a list of the csv files

csv_files <- list(may_2022, jun_2022, jul_2022, aug_2022, sep_2022,
    oct_2022, nov_2022, dec_2022, jan_2023, feb_2023, mar_2023,
    apr_2023)

# The first csv file (may_2022) is the benchmark
```

```
first_file_dtypes <- sapply(csv_files[[1]], class)



# check for consistency for other csv files

all_same_dtypes <- all(sapply(csv_files[-1], function(file) identical(sapply(file,
    class), first_file_dtypes)))

# Checking the result of the foregoing test

if (all_same_dtypes) {
    print("Data type for all columns of all csv files are IDENTICAL")
} else {
    print("Data type for all columns of all csv files are NOT IDENTICAL")
}

## [1] "Data type for all columns of all csv files are IDENTICAL"
```

As with the column names, the code confirmed that the data types are also consistent across all columns for all csv files. To investigate further, the following code was executed to check for the specific data types for each of the 13 columns.

```
# To check for the individual data types of each column

data_types <- tibble(Column = names(full_year), DataType = sapply(full_year, class))

print(data_types, row.names = FALSE)

## # A tibble: 13 × 2
##    Column             DataType
##    <chr>              <chr>
##  1 ride_id            character
##  2 rideable_type      character
##  3 started_at         character
##  4 ended_at           character
##  5 start_station_name character
##  6 start_station_id   character
##  7 end_station_name   character
##  8 end_station_id     character
##  9 start_lat          numeric
## 10 start_lng          numeric
## 11 end_lat            numeric
## 12 end_lng            numeric
## 13 member_casual      character
```

Moving along with the steps for data cleaning and wrangling, the individual csv files can now be merged into one main data frame: *full_ year*.

```
# We will combine all files using the bind function

full_year <- bind_rows(csv_files)
```

The data cleaning procedure for this project involved checking for missing values, duplicates, and the unique identifiers for select columns that can better inform our analysis. Additionally, appropriate treatment was also applied to the identified issues. To start, the total number of observations in the main *full_ year* data frame was checked using the *n.row()* function.

```
# To check for total number of rows in data frame

n.row(full_year)
```

```
## [1] 5,859,061
```

In order to check for missing values within the data frame, the *col.Sums* and *is.na* functions were used to count the total number of null values within our *full_year* data frame. This approach identified the total number of missing values in each of the 13 columns. Of note is the difference between missing values and empty observations. Each of these issues had its appropriate treatment in this project.

```
# Check for missing values

missing_values <- colSums(is.na(full_year))
print(missing_values)

##            ride_id      rideable_type         started_at           ended_at
##                  0                  0                  0                  0
## start_station_name   start_station_id   end_station_name     end_station_id
##                  0                  0                  0                  0
##          start_lat          start_lng            end_lat            end_lng
##                  0                  0               5973               5973
##      member_casual
##                  0
```

The code revealed that there are 5,973 missing values for both *end_lat* and *end_lng* columns. These columns contain information about geographic coordinates of the end locations corresponding for each ride id. In terms of proportional significance, the missing values only account for *0.001%* of the entire data frame. As such, the rows with null values will be dropped.

```
# To remove rows with null values

full_year <- full_year[complete.cases(full_year$end_lat, full_year$end_lng),]

# Re-check for missing values in updated data frame

missing_values <- colSums(is.na(full_year))
print(missing_values)

##            ride_id      rideable_type         started_at           ended_at
##                  0                  0                  0                  0
## start_station_name   start_station_id   end_station_name     end_station_id
##                  0                  0                  0                  0
##          start_lat          start_lng            end_lat            end_lng
##                  0                  0                  0                  0
##      member_casual         month_year          ride_time          week_date
##                  0                  0                  0                  0

# To check for new total number of rows in data frame

n.row(full_year)

## [1] 5,858,088
```

Following the elected treatment for 5,973 rows with null values, the *full_year* data frame is left with 5,858,088 observations. At this point, the data frame was examined for duplicates. One important consideration in executing this task was computational efficiency. With this in mind, the data.table package was installed and loaded in order to convert the data frame to a data.table. This conversion is important to do away with the standard *duplicated()* function – a computationally demanding process that can take its toll on processing time.

```
# Convert the data frame to a data.table
setDT(full_year)
```

```
# Check for duplicates
duplicates <- full_year[duplicated(full_year), ]


# Display duplicated rows
if (nrow(duplicates) > 0) {
    print("Duplicate rows found:")
    print(duplicates)
} else {
    print("No duplicate rows found.")
}

## [1] "No duplicate rows found."
```

While the code has confirmed the nonexistence of duplicate rows in the data frame, further validation is needed, particularly in determining the number of unique identifiers in certain columns of interest. The same duplicate check was performed for the following columns: *ride_id, start_station_id,* and *end_station_id.*

```
# Check for duplicates in the three identified columns

ride_id_duplicates <- any(duplicated(full_year$ride_id))
start_stationid_duplicates <- any(duplicated(full_year$start_station_id))
end_stationid_duplicates <- any(duplicated(full_year$end_station_id))

# Print Results
print(paste("Duplicate values found on ride_id:", ride_id_duplicates))

## [1] "Duplicate values found on ride_id: FALSE"

print(paste("Duplicate values found on start_station_id:", start_stationid_duplicates)
)

## [1] "Duplicate values found on start_station_id: TRUE"

print(paste("Duplicate values found on end_station_id:", end_stationid_duplicates))

## [1] "Duplicate values found on end_station_id: TRUE"
```

Two insights were gleaned from the result of the executed code. First, the *ride_id* column indicates that each entry corresponds to a unique ride session triggered by a Cyclistic client within the bike-share network. Second, the presence of duplicate values in the start and end station id columns suggests that docking stations within the network have preset identification characters that may be useful in the analysis. To confirm this, a code was executed to count the total number of unique start and end station IDs.

```
# To count the number of unique start_station_ids

unique_start_stations <- unique(full_year$start_station_id)
total_unique_ss <- length(unique_start_stations)

# To count the number of unique end_station_ids

unique_end_stations <- unique(full_year$end_station_id)
total_unique_es <- length(unique_end_stations)

# Print results

cat("Number of Unique Start Station IDs:", total_unique_ss, "\n")

## Number of Unique Start Station IDs: 1320

cat("Number of Unique End Station IDs:", total_unique_es, "\n")
```

```
## Number of Unique End Station IDs: 1325
```

A total of 1,320 unique start station IDs and 1,325 unique end station IDs were identified. This means that while the dataset had in excess of 5.8 million total observations, each ride session started and ended in any of over 1,300 docking stations situated within Cyclistic's locale of operation. This final step completes the data cleaning and inspection process.

At this point, it is important to approach data wrangling with the identified business tasks for the project in mind. As such, three (3) additional columns were created to extend the current data frame. All of the values in these additional columns were extracted from the datetime columns *started_ at* and *ended_ at* of the *full_ year* data frame. The first additional column was named *month_ year*. This column extracted month and year values from the two datetime columns to assign a month and year value to each observation. This is important for analyzing historical data over the past 12 months in the consolidated data frame.

```
# To convert started_at, ended_at columns to datetime

full_year$started_at <- ymd_hms(full_year$started_at)
full_year$ended_at <- ymd_hms(full_year$ended_at)

# To create and append column in data frame
full_year$month_year <- format(full_year$started_at, "%B %Y")
```

On the one hand, the second column was named *ride_ duration*. This column extracted the difference between the time values stored in the two datetime columns. The output value is expressed in minutes, and this helped in adding more granularity to the data being studied.

```
# To add ride_time column
full_year$ride_time <- difftime(full_year$ended_at, full_year$started_at)

# To convert ride_time column to numeric for calculations
full_year$ride_time <- as.numeric(as.character(full_year$ride_time))

# To check
class(full_year$ride_time)

## [1] "numeric"
```

Lastly, a column named *week_ date* was added, which specified the day of the week corresponding to each ride_id entry. This afforded the analysis with more specific observation points for comparing casual riders from annual members.

```
# We start by converting datetime columns to POSIXlt
full_year$week_date <- weekdays(as.Date(full_year$started_at))

# Convert week_date to factor for categorical
# representation
full_year$week_date <- factor(full_year$week_date, levels = c("Sunday",
    "Monday", "Tuesday", "Wednesday", "Thursday", "Friday", "Saturday"))
```

At this point, the data frame has been adequately prepared for analysis.

Analysis

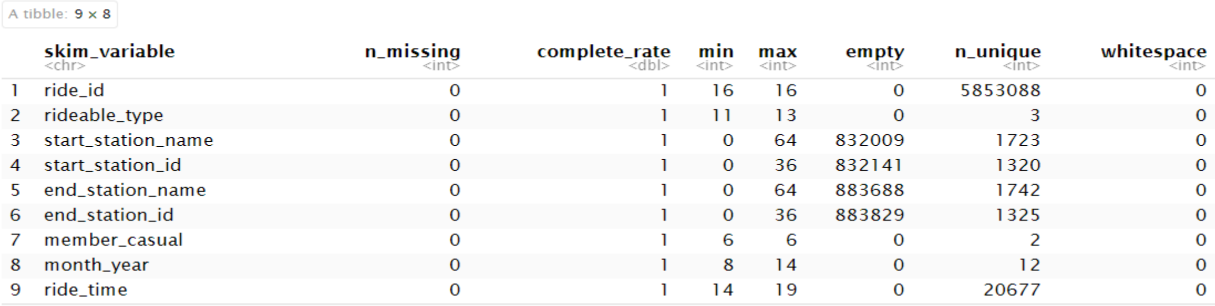
To recall, the identified business tasks for the project informed the direction of the analysis, namely:

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?

The analysis section focused on questions that were answerable by quantifiable data. Those that were not covered in the analysis portion were discussed in the insights and recommendations.

Descriptive Statistics

The skim () function provides users with a detailed, high-level overview of the dataset. In this project, I have elected to only include the most relevant portions of the code output to keep the presentation organized. The figure that follows reveals basic analyses on the *full_ year* data frame.



A tibble: 9 x 8

	skim_variable <chr>	n_missing <int>	complete_rate <dbl>	min <int>	max <int>	empty <int>	n_unique <int>	whitespace <int>
1	ride_id	0	1	16	16	0	5853088	0
2	rideable_type	0	1	11	13	0	3	0
3	start_station_name	0	1	0	64	832009	1723	0
4	start_station_id	0	1	0	36	832141	1320	0
5	end_station_name	0	1	0	64	883688	1742	0
6	end_station_id	0	1	0	36	883829	1325	0
7	member_casual	0	1	6	6	0	2	0
8	month_year	0	1	8	14	0	12	0
9	ride_time	0	1	14	19	0	20677	0

The number of unique entries in the *ride_ id* column reflects the current number of observations in our data frame, following the treatments applied during the data cleaning and wrangling phases. An interesting observation to note is the sheer number of empty values in the start and end station variables. Since these empty cells account for ~15% of the entire data frame, the affected rows will not be dropped.

A deeper study of the dataset and of the client company reveals further insights into the empty observations. For one, the fact that a session was recorded via the unique *ride_ id* entry for rows with empty start / end station cells suggests that while the bicycle was used within Cyclistic's network, the session might not have started at a preset or designated start station, hence the empty values. Consequently, for one reason or another, some users might not have returned or ended their sessions at the appropriate end station, hence the empty values in those cells, too. On the other hand, the lack of start / end station values may also be attributed to a faulty geo-tracker or a geo-tracker running out of power, which in both cases will result in its failure to record where the station started or ended. For a bike sharing company like Cyclistic, these instances are mundane and expected. For the purposes of this project, these considerations were factored into the analysis whenever applicable.

Distribution of Rides per User Type

The preliminary scan of the dataset reveals that Cyclistic's customer base fall in either of two categories. Users may be considered as casual riders (purchases single-day or full-day pass) or members (purchases annual membership). In any case, the initial hypothesis postulated was that the way forward for the company is to convert casual riders into annual members. To confirm this, the distribution of Cyclistic's membership base was quantified based on the count of membership type for all unique ride observations. An important consideration to note in this portion of the analysis is that the dataset did not permit the identification of unique members, i.e., one or more unique ride id observations may be attributable to one member. In other words, one user may use Cyclistic's services several times throughout the day, with each engagement recorded as a unique ride id session.

```r
# Distribution of member types
member_type_dist <- table(full_year$member_casual)

# Calculate the percentages
percentages <- round(prop.table(member_type_dist) * 100, 2)

# Create a Pie Chart
pie(percentages, labels = paste(names(percentages), percentages,
    "%"), main = "Member Type Distribution")
```

Ride Distribution per Member Type

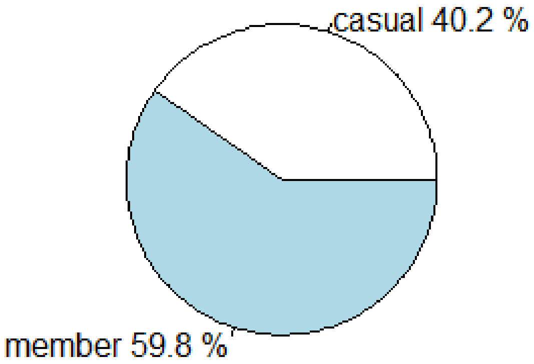


Figure 1. *Cyclistic's Membership Distribution*

Because of the limitations in the dataset, the main conclusion that may be derived from the code output is that users who pay for annual memberships use Cyclistic's services by approximately 20% more than casual riders. To reiterate, this does not mean that are almost 20% more annual members than casual riders in Cyclistic's user database since privacy laws prevent the access of data that leads to identifying how many unique users are currently in the network. What the code measures is the frequency of unique ride sessions attributable to either casual user or annual member.

Average Ride Duration

In order to further validate the initial hypothesis, it is important to further look into other attributes that can help the team to understand how casual riders differ from annual members. To start, I considered how the two member-types differed in terms of average ride duration. The expected output was achieved using the aggregate function and looking into the mean of the *ride_ time* column created during the data wrangling phase. Note that the outputs are expressed in minutes.

```
# Average ride time for casual and member users
aggregate(full_year$ride_time/60 ~ full_year$member_casual, FUN = mean)

##   full_year$member_casual full_year$ride_time/60
## 1                  casual               21.22373
## 2                  member               12.19585
```

Based on the initial results, ride sessions with casual users take longer than that of annual members. To further validate the accuracy of this statistic, I checked for possible outliers that could have skewed the results using the minimum and maximum values of the *ride_ time* column.

```
aggregate(full_year$ride_time/60 ~ full_year$member_casual, FUN = max)

##   full_year$member_casual full_year$ride_time/60
## 1                  casual              32035.450
## 2                  member               1499.933

aggregate(full_year$ride_time/60 ~ full_year$member_casual, FUN = min)

##   full_year$member_casual full_year$ride_time/60
## 1                  casual              -137.4167
## 2                  member            -10353.3500
```

There were anomalies identified in both extremes. In the case of the max value, the result suggests that the longest session lasted for about 32,000 minutes. For one reason or another, this may be caused by either the user or the equipment during which the ride time continued to count even if the actual ride session had already ended. These issues were normalized by capping the ride time to a maximum of 86,400 seconds.

```
max_ride_time <- 86400  # Maximum ride time limit in seconds (24 hours)

# Set maximum ride time for values exceeding the limit
full_year$ride_time <- ifelse(full_year$ride_time > max_ride_time,
    max_ride_time, full_year$ride_time)
```

Similarly, negative ride times were also accounted for in the dataset. To determine the appropriate treatment for the values involved, I had to determine how many rows in particular had negative *ride_ time* values.

```
# Create a subset for affected rows

negative_rows <- full_year[full_year$ride_time < 0, ]

# View results
View(negative_rows)
```

There appears to be 103 observations affected by negative *ride_ time* values. These entries are a result of an anomaly in the datetime values, wherein the recorded *ended_ at* session is earlier than the *started_ at* session. The problematic entries appear at random throughout the dataset, particularly in the months of May – November 2022, and February and April 2023. Since the affected observations are proportionally insignificant relative to the entire data frame, these values were dropped altogether.

```
full_year <- full_year[full_year$ride_time >= 0, ]
```

Following these treatments, the average ride duration was checked again:

```
aggregate(full_year$ride_time/60 ~ full_year$member_casual, FUN = mean)

##   full_year$member_casual full_year$ride_time/60
## 1                  casual               21.19070
## 2                  member               12.19891
```

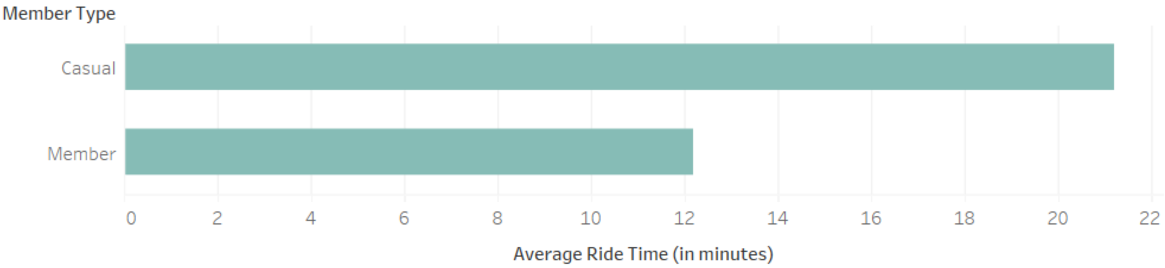


Figure 2. Average Ride Time per Member Type

The erring values did not have a significant effect on the average value. This means that casual riders really did take longer ride times on average than their annual member counterparts. That said, the analysis ought to confirm another inquiry on whether or not casual riders use Cyclistic's services for leisure. In addition to the average ride time, the difference between average distance covered per member type was also examined.

Average Distance

This part of the analysis required the creation of another column based on data extracted from the start and end coordinates of each ride session. The data was extracted as follows:

```
# Compute the distance travelled for each ride in km
distances <- distHaversine(matrix(c(full_year$start_lng, full_year$start_lat),
    ncol = 2), matrix(c(full_year$end_lng, full_year$end_lat),
    ncol = 2))/1000  # Convert to km

# Add the computed distances as a new column in the data frame
full_year$distance_km <- distances
```

Using the aggregate function, the average distance per member type was computed as follows:

```
aggregate(full_year$distance_km ~ full_year$member_casual, FUN = mean)

##   full_year$member_casual full_year$distance_km
## 1                  casual              2.148763
## 2                  member              2.099811
```

As was performed in previously, the results were validated by looking into possible outliers in the dataset. To do this, the maximum and minimum values were evaluated.

```
aggregate(full_year$distance_km ~ full_year$member_casual, FUN = max)

##   full_year$member_casual full_year$distance_km
## 1                  casual              9825.063
## 2                  member              9824.371

aggregate(full_year$distance_km ~ full_year$member_casual, FUN = min)

##   full_year$member_casual full_year$distance_km
## 1                  casual                     0
## 2                  member                     0
```

As it appears, the outlier values seem to be concentrated on the upper extreme. Considering Cyclistic's locale of operations, 50km was established as the upper threshold to determine the extent of the outlier values.

```
sum(full_year$distance_km > 50)

## [1] 8
```

There are only 8 observations with distances that exceed 50 km. Before rendering the appropriate treatment, a tibble was used to examine the affected rows in more detail.

```
above_50_tibble <- full_year %>%
    filter(distance_km > 50)
```

The affected rows are neither statistically nor proportionally significant. As such, these rows were dropped from the data frame. To maintain the integrity of the dataset, a new filtered data frame was created for this purpose.

```
full_year_filtered <- subset(full_year, distance_km <= 50)
```

At this point, the average distances were rechecked again.

```
aggregate(full_year_filtered$distance_km ~ full_year_filtered$member_casual,
    FUN = mean)

##   full_year_filtered$member_casual full_year_filtered$distance_km
## 1                           casual                       2.140414
## 2                           member                       2.082974
```

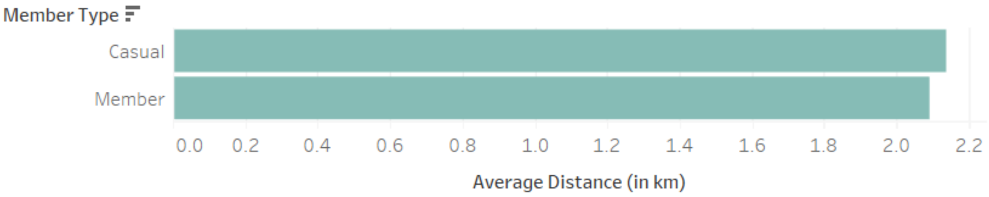


Figure 3. Average Distance per Member Type

Based on the results, there is still not that much difference between the average distance covered by casual users from annual members. Another way to answer the foregoing question is to look into the count of riders throughout the hours of the day.

Number of Riders per Hour / Day

To examine statistic, a new *hour* column is extracted using the lubridate package out of the *started_ at* datetime column.

```
# Create an hour column
full_year$hour <- lubridate::hour(full_year$started_at)
```

After which, the results were visualized within R.

```
# Create the line chart
ggplot(full_year, aes(x = hour, group = member_casual, color = member_casual)) +
    geom_line(stat = "count") + labs(x = "Hour of the Day", y = "Number of Riders") +
    scale_x_continuous(breaks = 0:23) + scale_color_manual(values = c(member = "blue",
    casual = "red")) + theme_minimal() + scale_y_continuous(labels = comma)
```

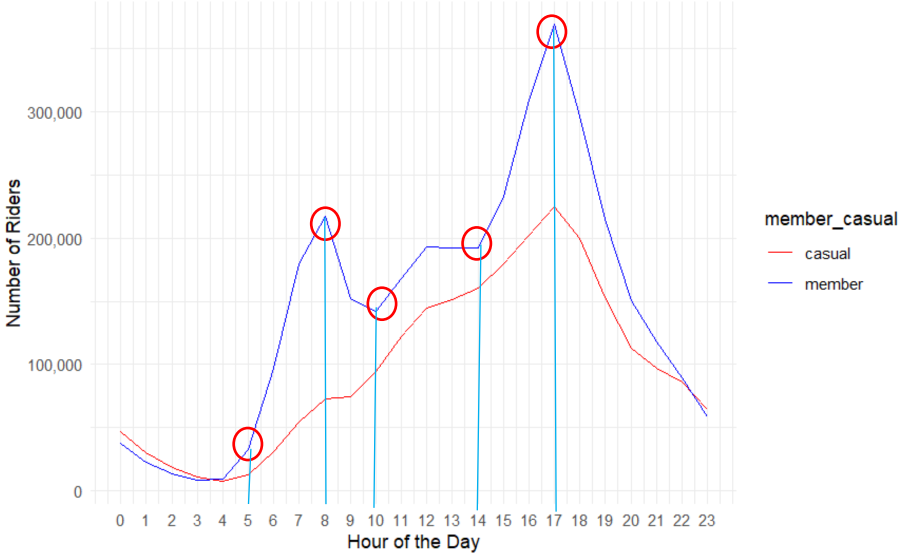


Figure 4. *Number of Users per Hour*

What is immediately apparent is how annual members outnumber casual riders at particular hours of the day. If we are allowed to infer, it appears as if the peaks and troughs in the chart can be explained by the typical schedule of corporate America. For instance, the number of riders begin to increase at around 5:00 AM and peaks at 8:00 AM when office work for most companies start. The count decreases until around 10:00 AM and peaks again at 12:00 NN, in time for lunch. The count remains more or less consistent from then until it starts to increase again by 2:00 PM, ultimately peaking at 5:00 PM, the end of the work day. As this statistic represents the aggregated figure for the entire dataset, I considered this as suggestive of the hypothesis that annual members use Cyclistic's services for their daily commute. To further validate this, I examined the distribution of the number of rides for each day of the week per member type using the following:

```
# Create the bar chart
ggplot(full_year, aes(x = week_date, fill = member_casual)) +
    geom_bar(stat = "count", position = "dodge") + labs(x = "Day of the Week",
    y = "Number of Rides") + scale_fill_manual(values = c(member = "blue",
    casual = "red")) + theme_minimal() + scale_y_continuous(labels = comma)
```

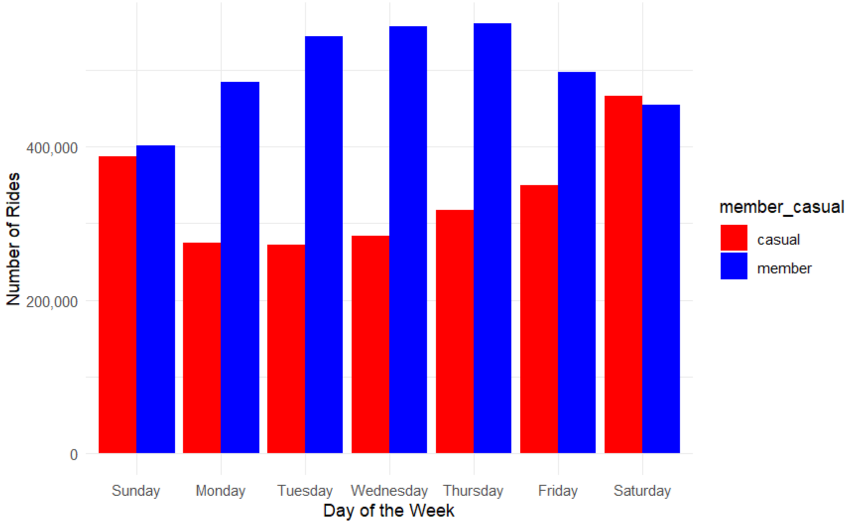


Figure 5. User count per day of the week

This visualization further supports the foregoing postulate about annual members using the bikes to commute to and from their respective workplaces. As shown, the number of rides attributable to annual members consistently outranked those attributable to casual riders throughout the week, except for Saturday. Of note is the fact that while the difference between ride count is marginal on the weekends, the variance becomes more profound during the workdays, which seem to further confirm the hypothesis.

Number of Rides per Month

With the hourly and weekly data appearing to be on the affirmative side of the hypothesis, one final dimension that must be explored is on whether or not the same trend stands throughout the scope of the analysis, i.e., all 12 months covered. To check this, months corresponding to each observation were extracted and aggregated to visualize the trend.

```r
# Convert started_at to date-time format
full_year$started_at <- as.POSIXct(full_year$started_at, format = "%Y-%m-%d %H:%M:%S")

# Extract the month from the started_at column
full_year$month <- format(full_year$started_at, "%B")

# Create a vector to express output in proper order
month_order <- month.name

# Convert month variable to a factor following desired order

ride_counts$month <- factor(ride_counts$month, levels = month_order)

# Create the bar chart
ggplot(ride_counts, aes(x = month, y = count, fill = member_casual)) +
    geom_col(position = "dodge") + labs(x = "Month", y = "Ride Count",
    fill = "Member Type") + scale_y_continuous(labels = comma) +
    theme_minimal() + guides(fill = guide_legend(title = "Member Type")) +
    theme(axis.text.x = element_text(angle = 45, hjust = 1))
```

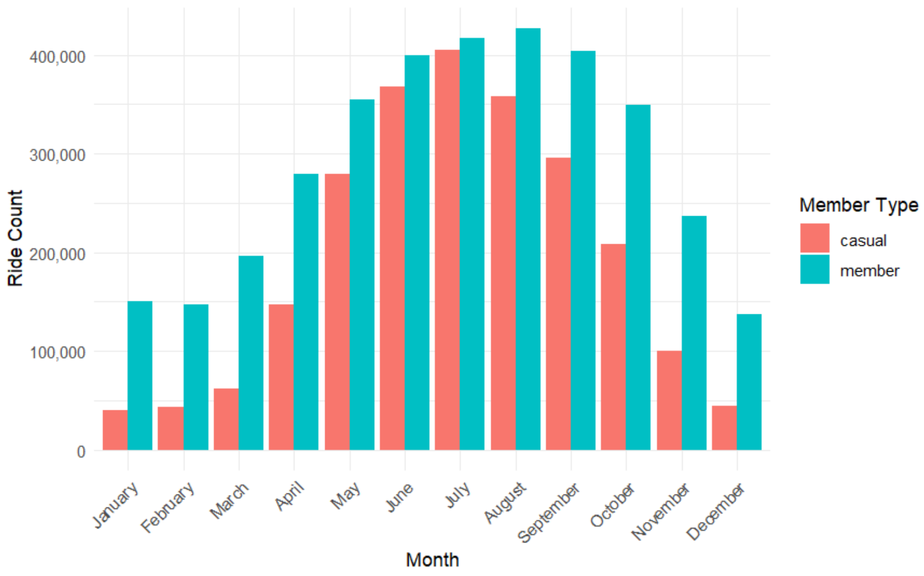


Figure 6. Rides per Month per User type

Much can be gleaned from the code output. Chief among all is that the apparent seasonality trend in this graph suggests that annual members do use Cyclistic's services for work. The main basis for this assertion is the comparatively higher count of member-rides even during the winter season relative to casual rides. This is because during winter months, those who use the bikes for leisure are effectively deterred by the weather from doing so. Conversely, those who use it for work still have to use the bikes as they report to their workplaces. The same trend endures throughout the year. As it appears, the ride count begins to pick up again around spring, peaks during the summer months, and then begins to descend again in the fall, leading up to winter before the cycle repeats again.

Preferred Ride type

Now that the major insights have been thoroughly fleshed out, we finally return to issue areas that can help in designing the marketing strategy for Cyclistic. We begin with looking into the preferred ride type of each member type.

```
# Count of rideable types per member type
rideable_count <- full_year %>%
    group_by(member_casual, rideable_type) %>%
    summarise(count = n()) %>%
    ungroup()

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

# Express relative percentages
percentage_counts <- rideable_count %>%
    group_by(member_casual) %>%
    mutate(percentage = count/sum(count) * 100)

# Rank of rideable types
ranked_counts <- percentage_counts %>%
    group_by(member_casual) %>%
    mutate(rank = rank(desc(count)))
```

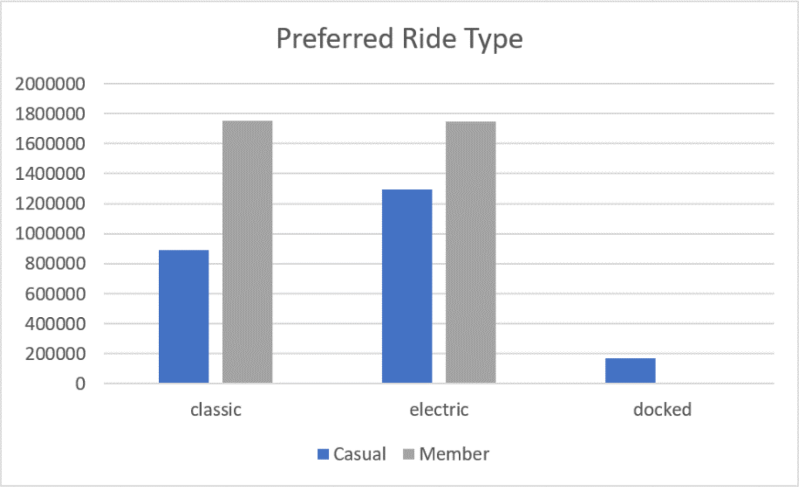


Figure 7. Preferred Ride Type of Each User Category

As provided by the code output, annual members do not diverge significantly in terms of preferred ride type, particularly between classic and electric bikes. Conversely, casual riders do seem to have more preference over electric bikes than classic bikes. If the goal of the project is to supply actionable insights to the marketing team, the ride type preference of casual riders, whom we have targeted to convert to annual members, must be noted.

Top Start and End Stations

Lastly, to get a hint of the start and end stations where users usually congregate, the analysis looked into the top 5 start and end stations for both user types. However, it must be noted that at least 5% of all ride observations in our dataset have no known start and end locations. As mentioned earlier, this may be a consequence of a user picking up / leaving a bike at places within the network other than the designated stations, or it could also be caused by a faulty battery in the bicycle's geo-tracker.

```r
# Calculate top start and end stations for casual users
top_start_casual <- full_year %>%
    filter(member_casual == "casual") %>%
    count(start_station_name) %>%
    top_n(6, wt = n) %>%
    arrange(desc(n))

top_end_casual <- full_year %>%
    filter(member_casual == "casual") %>%
    count(end_station_name) %>%
    top_n(6, wt = n) %>%
    arrange(desc(n))

# Calculate top start and end stations for members
top_start_member <- full_year %>%
    filter(member_casual == "member") %>%
    count(start_station_name) %>%
    top_n(6, wt = n) %>%
    arrange(desc(n))

top_end_member <- full_year %>%
    filter(member_casual == "member") %>%
    count(end_station_name) %>%
    top_n(6, wt = n) %>%
    arrange(desc(n))
```

```
# Create the output tibble
output <- tibble(`Top Start Station for Casual` = top_start_casual$start_station_name,
    `Count for Casual (Start)` = top_start_casual$n, `Top End Station for Casual` = to
p_end_casual$end_station_name,
    `Count for Casual (End)` = top_end_casual$n, `Top Start Station for Member` = top_
start_member$start_station_name,
    `Count for Member (Start)` = top_start_member$n, `Top End Station for Member` = to
p_end_member$end_station_name,
    `Count for Member (End)` = top_end_member$n)
```

Table 2. Top 5 Start and End Stations for Casual Riders

Top 5 Start Stations	Count	Top 5 End Stations	Count
Streeter Dr & Grand Ave	57,189	Streeter Dr & Grand Ave	59,543
DuSable Lake Shore Dr & Monroe St	31,926	DuSable Lake Shore Dr & Monroe St	29,328
Michigan Ave & Oak Street	25,341	Michigan Ave & Oak Street	26,638
Millennium Park	25,150	Millennium Park	26,597
DuSable Lake Shore & North Blvd	23,630	DuSable Lake Shore & North Blvd	26,216

Table 3. Top 5 Start and End Stations for Annual Members

Top 5 Start Stations	Count	Top 5 End Stations	Count
Kingsbury St & Kinzie St	25,423	Kingsbury St & Kinzie St	25,331
Clark St & Elm St	23,320	Clark St & Elm St	23,583
University Ave & 57th St	22,160	Wells St & Concord Ln	22,830
Wells St & Concord Ln	22,079	Clinton St & Washington Blvd	22,687
Clinton St & Washington Blvd	21,680	University Ave & 57th St	22,571

Analyzing the top 5 start and end stations for both user types offer further insights into how the two user categories differ in terms of their primary purpose in engaging Cyclistic's services. For one, the top stations for casual users are all located within the vicinity of the coast and docking pier, which suggests that they may be using the company's services for leisure or sightseeing. On the contrary, the top stations for annual members are more inland towards the city, which seem to confirm the hypothesis that annual members use Cyclistic's bikes for their primary mobility needs such as doing errands or going to their places of work.

There is also consistency in the start and end stations for casual users, which means that for those who do, casual users end their ride sessions at the exact place where they started. This only holds true in the top 2 stations for annual members. In any case, these observations still seem to affirm the hypothesis held regarding what customer use Cyclistic's services for.

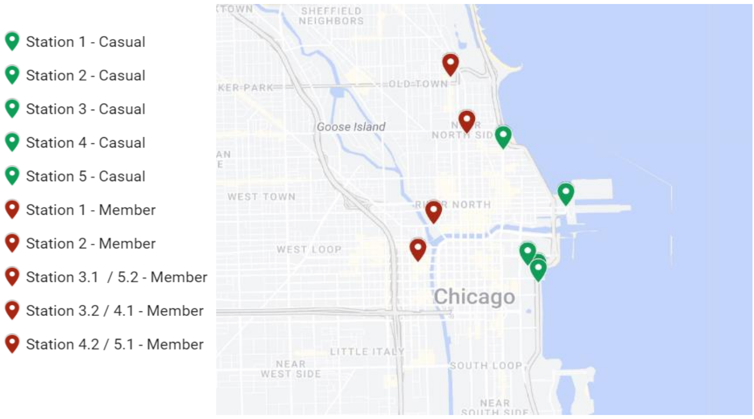


Figure 8. Location of Top Stations per User Type

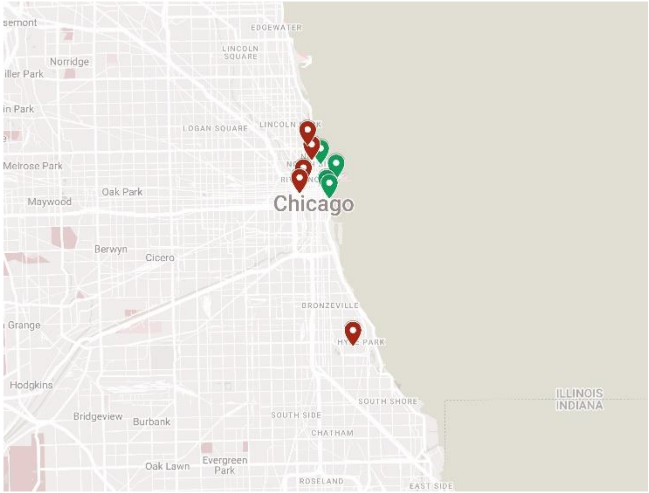


Figure 9. A bird's eye view of the map to include the last member station

Insights and Recommendations

To recall, the overarching objective for the analysis is to explore the hypothesis that the key towards Cyclistic's financial success moving forward lies in converting casual riders to annual users, rather than acquiring new customers. As such, this project was also prompted by the need to respond to three key questions:

1. How do annual members differ from casual riders?
2. Why would casual riders want to buy annual memberships?
3. How can Cyclistic use digital media to influence casual riders to become annual members?

In general, I submit two options for the team to move forward. Either the team presses on with the marketing campaign based on the insights of this analysis, or a secondary data gathering and analysis be implemented to arrive at a more nuanced picture of how casual riders differ from annual members and how casual riders may be nudged to purchase annual memberships.

Act on Current Results

If the first option is pursued, the team has to be comfortable with making assumptions to compensate for the limitations of our current dataset. In its present form, the scope of the analysis was only able to cover 8 areas of interest, and from the results thereof, the behavioral differences between the two user types were ascertained. Because of the identified gaps, however, it is reasonable to consider the results as indicative, rather than conclusive, of the behavioral differences we wanted to inspect.

The key observation that I wish to underscore in this analysis is that if the team's goal is to convert casual users to annual members, there has to be a deeper understanding first of the current pain points experienced by casual users. Once this is identified, the team could focus on incentivizing the shift from casual ridership to annual membership. As far as the results are concerned, it seems as if the major difference between the two user types go back to their specific purposes for engaging with Cyclistic's services. While the results are indeed not conclusive, there is evidence that points to casual users using our services for leisure / recreational activities, whereas annual members acquire their membership because they regularly need Cyclistic's bikes for their mobility needs. In other words, the solution is actually in redefining Cyclistic's concept of who an annual member is.

If the goal is to incentivize casual riders into making the switch, the team could look into payment plans and packages that incentivizes the current behavior of casual riders. For instance, a weekend membership pass can be offered since the data showed that casual rides spike on weekends. In the same breath, a Spring – Summer membership may also be offered since ride users for both types usually increase during months with weather conditions favorable to biking. Additionally, a rewards system may also be offered in order to incentivize riders to clock in greater distances per session. Initiatives like this can help provide reasons for casual riders to want to buy annual memberships, or any other version of a membership that makes them switch from being at most, full-day users. In any case, it would be helpful to confirm the financial viability of these options with the finance analytics team to work out the most profit-maximizing route.

As far as the use of digital media is concerned, the marketing campaign can use influencers or advertise in locations where most casual rides take place as pointed out in the data. The identified locations are different from the stations where most annual members congregate, and so the marketing could be more targeted in this sense. In terms of content, the team may also want to consider advertising more on electric bikes since this ride type appears to be that which casual riders prefer.

Iterate and Expand the Study s Scope

If the project is not impeded by time and financial constraints, the team may want to consider iterating the study and to collect more datapoints that can help address the main case question. Since the objective is in trying to convert casual riders to annual members, it would be to the company's advantage to invest in gathering data that can help us understand who are our casual riders are at a deeper level. This may be achieved through survey questionnaires, interviews, or some other form of data collection that seeks to collect direct data from casual riders.

A major limitation of our current dataset is that it is a generic record that compares the most basic data points between ride types. If the casual riders are sampled appropriately, then the team may be able to glean and gather helpful insights that can provide answers into how we ought to approach the user conversion strategy. In any event, this is an executive decision that must be addressed by the appropriate authorities. Nevertheless, as far as the data analysis is concerned, the foregoing insights are those that I was able to extract from our dataset for the past 12 months.