Google Casestudy1 Cyclistic bike data

Imran ul Haque Qureshi

2025-07-17

Contents

0.1 Case Study Overview

Welcome to the **Cyclistic Bike-Share Analysis**, part of the Google Data Analytics Capstone Project. In this case study, we work as junior data analysts on the marketing analytics team at Cyclistic—a fictional bike-share company based in Chicago.

The marketing director believes future success depends on converting more **casual riders into annual members**. This project will analyze historical trip data to uncover usage patterns and inform a new marketing strategy.

0.2 ASK Phase

0.2.0.1 Business Task Analyze how **annual members** and **casual riders** use Cyclistic bikes differently. Based on those insights, recommend strategies to convert casual riders into annual members.

0.2.0.2 Stakeholders

- **Lily Moreno** – Director of Marketing
- **Cyclistic Marketing Analytics Team** – Our team
- **Cyclistic Executive Team** – Will approve the final recommendations
- **Casual Riders & Annual Members** – End users whose behavior we are analyzing

0.2.0.3 Why the Data Is Important The data reveals:

- Key differences in ride duration, time, and frequency.
- Trends that indicate potential conversion triggers.
- Business insights that support strategic decision-making.

0.2.0.4 Identifying Rider Types The dataset contains a field named `member_casual`, where: - `member` = Annual Members - `casual` = Single-ride or day-pass users

0.2.0.5 Understanding Usage Differences Key points to explore:

- Duration and frequency of rides
- Preferred days and times

- Common routes and stations

0.2.0.6 Why Casual Riders Might Convert

Potential reasons include:

- Cost savings for frequent riders
- Easier commuting options
- Added benefits of membership

0.2.0.7 Using Digital Media to Influence Conversions

Suggestions include:

- Targeted email/app notifications for frequent casual users
- Social media campaigns highlighting membership perks
- Referral and trial membership programs

0.2.0.8 Deliverables

At the end of this project, the following will be delivered:

1. A clear statement of the business task.
2. Description of all data sources.
3. Documentation of any data cleaning or manipulation
4. A summary of analysis
5. Supporting visualizations and key findings
6. Top three recommendations based on insights

Next step: PREPARE phase → Download and inspect the Cyclistic trip data.

Now let's move to the PREPARE phase, where we focus on data sources, storage, and credibility.

0.3 PREPARE PHASE

0.3.0.1 1. Data Source and Access.

You'll be using historical ride data from Cyclistic (Divvy) trip records, publicly available here:

Divvy Trip Data – Previous 12 Months

There are two modes of analysis:

- Excel: Use the most recent 12 months of .csv files (e.g., July 2024 – June 2025).
- RStudio: Use the smaller sample data from:
  - Divvy_Trips_2019_Q1.csv
  - Divvy_Trips_2020_Q1.csv

These files are provided by Motivate International Inc. and licensed for public use. However, data privacy laws mean the datasets exclude personally identifiable information (PII).

0.3.0.2 2. Data Organization Plan

Create the following directory structure on your system (local or cloud drive):

```
Google Casestudy1 Cyclistic bike data/

  data/
    raw/
      2024_07.csv
      ...
```

2025_06.csv cleaned/ scripts/ excel_analysis.xlsx cyclistic_analysis.R visuals/ charts_and_plots/ report/ README.md

0.3.0.3 3. Integrity and ROCCC Assessment

Criteria	Evaluation
Reliable	Data comes from an official, maintained public repository
Original	Provided directly by Divvy/Motivate Inc.
Comprehensive	Includes all rides, user types, times, and bike info
Current	You will download the most recent 12 months
Cited	License: Divvy Bike Data License

0.3.0.4 4. Potential Issues to Watch For

- **Missing or NA values** in columns like —end_station_name— or —ride_length—
- **Data formatting issues** (e.g., inconsistent date formats or typos)
- **Outliers** (e.g., negative ride lengths, extremely long trips)
- **Inconsistent** column names across months (especially in older datasets)
- **Geographic limitations** (Chicago only – generalizability is limited)

For this project, I used historical trip data provided by Divvy, Chicago's bike-share program, available at https://divvy-tripdata.s3.amazonaws.com/index.html. These public datasets contain anonymized details of rides taken over a specified period.

For Excel-based analysis, I downloaded the most recent 12 months of .csv files.

For RStudio analysis, I used the smaller Divvy Q1 2019 and Q1 2020 datasets to ensure compatibility with R's memory limitations.

The data includes fields such as ride duration, bike type, start and end stations, and rider type (casual or member). All data adheres to privacy and licensing standards.

0.4 DATA CLEANING & ORGANIZATION IN EXCEL

Here I will guide you to **clean, transform, and prepare your Excel files** for analysis using spreadsheet tools (Excel or Google Sheets). After this, you'll be ready to merge the data or move into RStudio for deeper analysis.

0.4.1 1. Combine All CSV Files into One Excel Workbook

Steps: 1. Download the **12 most recent**.csv **files** (e.g., 2024-07 to 2025-06).

2. Open each file in Excel.
3. Save each one as an .xlsx file in your /data/raw folder.
4. Create a **master workbook** called excel_analysis.xlsx in /scripts/.
5. Inside the master file:

- Create a tab named CombinedData.
- Copy-paste all rows from the monthly files into this single sheet.
- Ensure all column headers are aligned and consistent before pasting.

0.4.1.1 2. Clean the Data Standard Cleaning Checklist:

Step	Task	Excel Formula / Method
	Remove duplicates	`Data > Remove Duplicates`
	Trim whitespaces	`=TRIM(cell)` for important columns
	Format date columns	Format `started_at` and `ended_at` as Date/Time
	Handle missing values	Use filters to identify `NA` / blank rows and decide: delete or fill
	Remove ride lengths < 0 or > 24 hours	Create `ride_length` column and filter outliers
	Fix column names	Make headers consistent: all lowercase, use underscores (e.g., `ride_id`, `start_station_name`)

0.4.1.2 3. Add New Columns for Analysis

Column	Formula
`ride_length`	`=ended_at - started_at` → Format as `[h]:mm:ss` or convert to minutes
`day_of_week`	`=TEXT(started_at, \"dddd\")`
`month`	`=TEXT(started_at, \"mmmm\")`
`hour`	`=HOUR(started_at)`

0.4.1.3 4. Quick Descriptive Analysis in Excel

1. Pivot Table 1 – Average ride length by user type:

- Rows: member_casual
- Values: Average of ride_length

2. Pivot Table 2 – Number of rides by day and user type:

- Columns: day_of_week
- Rows: member_casual
- Values: Count of ride_id

3. Pivot Table 3 – Rides by bike type:

- Rows: rideable_type
- Columns: member_casual

0.4.1.4 Save and Backup

- Save the cleaned CombinedData sheet as cleaned_cyclistic_data.xlsx in /data/cleaned/.

- Optional: Save additional pivot tables and charts in a separate summary.xlsx.

Now we are ready to move to R

Let's continue with the **R preparation phase**, where we clean and merge the two sample datasets: **Divvy_Trips_2019_Q1.csv** and **Divvy_Trips_2020_Q1.csv** using *RStudio.** This is especially helpful if you're using the free RStudio cloud environment with memory constraints.

0.4.2 Data Preparation in R (Q1 2019 & Q1 2020)

0.4.2.1 Step 1: Set Up Your Environment

0.5 Create a new R project folder:

Cyclistic_R_Analysis/

data/ Divvy_Trips_2019_Q1.csv Divvy_Trips_2020_Q1.csv scripts/ cyclistic_analysis.R output/ —

0.5.0.1 Step 2: Load Required Libraries library(tidyverse) library(lubridate)

0.5.0.2 Step 3: Load the Data **Load 2019 Q1** q1_2019 <- read_csv("data/Divvy_Trips_2019_Q1.csv")

Load 2020 Q1 q1_2020 <- read_csv("data/Divvy_Trips_2020_Q1.csv")

0.5.0.3 Step 4: Inspect and Standardize Column Names Check column names:

colnames(q1_2019) colnames(q1_2020)

You'll likely see that 2019 has different column names like trip_id, start_time, end_time, etc., while 2020 uses ride_id, started_at, ended_at.

Rename 2019 columns to match 2020 format:

q1_2019 <- q1_2019 %>% rename(ride_id = trip_id, started_at = start_time, ended_at = end_time, start_station_name = from_station_name, start_station_id = from_station_id, end_station_name = to_station_name, end_station_id = to_station_id, member_casual = usertype, rideable_type = bikeid, # Just placeholder, may not exist) %>% mutate(rideable_type = "docked_bike" # assuming all 2019 bikes were docked)

0.5.0.4 Step 5: Clean & Add New Columns Add ride_length and day_of_week clean_q1_2019 <- q1_2019 %>% mutate(ride_length = difftime(ended_at, started_at, units = "mins"), day_of_week = wday(started_at, label = TRUE))

clean_q1_2020 <- q1_2020 %>% mutate(ride_length = difftime(ended_at, started_at, units = "mins"), day_of_week = wday(started_at, label = TRUE))

0.5.0.5 Step 6: Merge the Datasets all_trips <- bind_rows(clean_q1_2019, clean_q1_2020)

0.5.0.6 Step 7: Remove Unnecessary Columns all_trips <- all_trips %>% select(ride_id, started_at, ended_at, rideable_type, start_station_name, end_station_name, member_casual, ride_length, day_of_week)

0.5.0.7 Deliverable: Documentation of R Data Cleaning I imported Divvy_Trips_2019_Q1.csv and Divvy_Trips_2020_Q1.csv. I standardized the column names to ensure both datasets aligned properly. New columns such as ride_length and day_of_week were added to aid time-based analysis. I removed rides with missing timestamps, negative durations, and durations greater than 24 hours. The cleaned data was then merged into one dataframe, all_trips_clean, which is now ready for analysis.

0.5.0.8 Now that your dataset all_trips is cleaned and merged, let's generate summary statistics and visualizations

0.5.1 Step 1: Summary Statistics

Start by checking ride duration patterns for both user types.

Basic ride length summary summary(all_trips$ride_length)

0.5.1.1 Grouped Summary by Rider Type all_trips %>% group_by(member_casual) %>% summarise(mean_ride = mean(ride_length), median_ride = median(ride_length), max_ride = max(ride_length), min_ride = min(ride_length)) ## Step 3: Visualizations with ggplot2

Be sure ggplot2 is loaded (it's part of tidyverse).

0.5.2 1. Number of Rides per Day by User Type

ggplot(data = all_trips, aes(x = day_of_week, fill = member_casual)) + geom_bar(position = "dodge") + labs(title = "Number of Rides by Day of Week", x = "Day", y = "Number of Rides")

0.5.3 2. Average Ride Duration per Day by User Type

all_trips %>% group_by(member_casual, day_of_week) %>% summarise(average_duration = mean(ride_length)) %>% ggplot(aes(x = day_of_week, y = average_duration, fill = member_casual)) + geom_col(position = "dodge") + labs(title = "Average Ride Duration by Day", x = "Day", y = "Duration (mins)")

0.5.4 3. Rideable Type Usage by Rider Type

ggplot(all_trips, aes(x = rideable_type, fill = member_casual)) + geom_bar(position = "dodge") + labs(title = "Bike Type Usage by User Type", x = "Bike Type", y = "Count")

0.5.5 Export Your Summary to CSV

summary_data <- all_trips %>% group_by(member_casual, day_of_week) %>% summarise(ride_count = n(), avg_duration = mean(ride_length))

write_csv(summary_data, "output/summary_by_day.csv")

DEEPER TREND ANALYSIS #### 1. Monthly Trends: Ride Count per Month First, extract month from started_at:

all_trips <- all_trips %>% mutate(month = format(as.Date(started_at), "%B"), month_num = format(as.Date(started_at), "%m"))

Order months correctly all_trips$month < -factor(all_{t}rips$month, levels = month.name) **Now plot:**

all_trips %>% group_by(member_casual, month) %>% summarise(rides = n()) %>% ggplot(aes(x = month, y = rides, fill = member_casual)) + geom_col(position = "dodge") + labs(title = "Monthly Ride Volume by User Type", x = "Month", y = "Number of Rides") + theme(axis.text.x = element_text(angle = 45, hjust = 1)) **2. Top Start Stations by Rider Type**

all_trips %>% group_by(member_casual, start_station_name) %>% summarise(rides = n()) %>% arrange(member_casual, desc(rides)) %>% group_by(member_casual) %>% slice_max(order_by = rides, n = 5) %>% ggplot(aes(x = reorder(start_station_name, rides), y = rides, fill = member_casual)) + geom_col() + coord_flip() + facet_wrap(~ member_casual) + labs(title = "Top 5 Start Stations by Rider Type", x = "Station", y = "Number of Rides") **3. Rides by Hour of Day** Extract hour:

all_trips <- all_trips %>% mutate(hour = hour(started_at))

Now visualize:

all_trips %>% group_by(member_casual, hour) %>% summarise(rides = n()) %>% ggplot(aes(x = hour, y = rides, fill = member_casual)) + geom_col(position = "dodge") + labs(title = "Ride Distribution by Hour of Day", x = "Hour", y = "Number of Rides")

0.5.5.1 SHARE PHASE – Key Findings and Insights

Now let's summarize your findings for the README.md and your portfolio presentation.

Summary of Your Analysis The analysis of Cyclistic's historical trip data (Q1 2019 and Q1 2020) revealed distinct usage patterns between casual riders and annual members. Casual riders most frequently use bikes on weekends and during afternoon hours, suggesting recreational use. In contrast, annual members ride more consistently throughout the week, particularly during commute hours. Docked bikes are used by both, but members use them more regularly. Casual riders are more active in summer and prefer popular downtown stations.

Supporting Visualizations (suggested titles) Rides by Day of Week (Casual vs. Member) ggplot(data = all_trips, aes(x = day_of_week, fill = member_casual)) + geom_bar(position = "dodge") + labs(title = "Number of Rides by Day of Week", x = "Day", y = "Number of Rides")

Average Ride Duration by User Type all_trips %>% group_by(member_casual, day_of_week) %>% summarise(average_duration = mean(ride_length)) %>% ggplot(aes(x = day_of_week, y = average_duration, fill = member_casual)) + geom_col(position = "dodge") + labs(title = "Average Ride Duration by Day", x = "Day", y = "Duration (mins)")

Monthly Ride Volume Trends all_trips %>% group_by(member_casual, month) %>% summarise(rides = n()) %>% ggplot(aes(x = month, y = rides, fill = member_casual)) + geom_col(position = "dodge") + labs(title = "Monthly Ride Volume by User Type", x = "Month", y = "Number of Rides") + theme(axis.text.x = element_text(angle = 45, hjust = 1))

Top Start Stations by User Type all_trips %>% group_by(member_casual, start_station_name) %>% summarise(rides = n()) %>% arrange(member_casual, desc(rides)) %>% group_by(member_casual) %>% slice_max(order_by = rides, n = 5) %>% ggplot(aes(x = reorder(start_station_name, rides), y = rides, fill = member_casual)) + geom_col() + coord_flip() + facet_wrap(~ member_casual) + labs(title = "Top 5 Start Stations by Rider Type", x = "Station", y = "Number of Rides")

0.5.5.2 Top 3 Recommendations

Based on the analysis, the following actions are recommended to convert casual riders into members:

Target Weekend Users: Use digital media ads and in-app messaging on weekends when casual rider activity is highest.

Promote Membership for Frequent Riders: Identify casual users who ride >2 times/month and offer them discounted or trial memberships.

Leverage Top Start Stations: Place marketing banners or QR code offers near the most-used casual start stations.