1. Introduction

This project involves analyzing data generated by a bikesharing application, one of the recommended datasets for the final project of the Google Data Analytics course. Based on a dataset with a straightforward structure, it serves as the initial step in learning concepts of data analysis.

2. Project Goal

The objective is to discern differences in behavior between two user types and to try to use these insights to potentially convert occasional users into subscribers through, for example, targeted marketing campaigns based on their particular style of using the service.

3. Platforms and Technologies Used

Programming languages: Python, SQL

Data processing: Pandas, BigQuery

Data visualization: Matplotlib, Seaborn, Folium

Machine learning: Scikit-learn

Cloud platforms: Google Cloud Platform

Data storage: Google Cloud Storage (GCS), BigQuery Storage

4. Code

5. Main Tasks

Due to the simple structure of the data and the lack of any more advanced system architectures in this project, the focus was on pre-processing and exploratory data analysis processes.

1) Data processing:

    - Data loading: The data was imported into BigQuery and then merged from twelve CSV files into one file.

    - Data cleaning: A number of checks were performed to fix errors such as duplicates or missing values and to detect outliers.

    - Data consistency analysis: An analysis was performed to ensure that the data is consistent and does not contain logically invalid values.

- Feature engineering: A new feature of ride length expressed in minutes was added, based on columns containing data about starting and ending time of each bike trip.

2) Data analysis:

- Statistical analysis: A review of the basic statistics of the dataset was conducted using standard Pandas commands. However, due to the simplicity of this dataset, standard statistical calculations could only be performed on one column - the ride length, which is the only quantitative variable present.

- Creating Visualizations: Python libraries were used to create different types of visualizations, including column charts, pie charts, and maps, in order to better understand the data.

3) Machine learning:

- Using the Scikit-learn library to train models to fill in missing data in two of the columns.

6. Data Overview

a) Data source

The dataset is sourced from a bikesharing application called Divvy, operated by Lyft on behalf of the Chicago Department of Transportation.
The Chicago Department of Transportation owns Divvy, and Lyft manages the application on their behalf. The data is shared under an open license.

The data used covers one year, from January to December of 2022 and comes in twelve CSV files, and each of them contains data for each month of the year.
Files in their original source can be found here: link.

b) Data format

The dataset is non-relational and comes in the form of twelve CSV files, each one of them containing data for each of the months of the year 2022.

c) Data structure

The dataset consists of the following columns:

ride_id - Identifier of the ride in alphanumeric format with a length of 16 characters.

rideable_type - The type of bike rented. Categorical variable, with possible values of:

- electric_bike
- classic_bike
- docked_bike

started_at, ended_at - The start and end date and time of each ride, formatted in datetime format and UTC time zone.

start_station_name, start_station_id, end_station_name, end_station_id - Names and identifiers of start and end stations, represented as categorical variables. Station names are represented as their literal names, while station identifiers are a set of alphanumeric, categorical values that do not have a standardized format across all data rows.

start_lat, start_lng, end_lat, end_lng - The geographic coordinates of the starting and ending points of each trip are provided with precision of up to 9 to 10 digits.

member_casual - A categorical variable indicating whether the bicycle was rented by an occasional user or a subscriber to the service. Possible values are:

- member
- casual

d) Size of the data

The dataset consists of 5.667 million rows in total and takes 950MB of disk space.

e) Data quality

There are no duplicate rows in the dataset, and there are no data consistency issues.

The missing values occur in significant amounts in the columns containing names of final and end stations, and there are around 830 and 890 thousand of them, respectively. In addition to this, there are also missing values for a small number of rows in the columns containing longitude and latitude information of the locations of the ends of the trips, but these are small amounts, as there are only nearly 6,000 of them.

f) Example of the data

Row	rideable_type	started_at	ended_at	start_station_name	end_station_name	start_lat	start_lng	end_lat
1	electric_bike	2022-09-16 05:10:35 UTC	2022-09-16 05:20:11 UTC	*null*	*null*	42.01	-87.67	42.0
2	electric_bike	2022-05-06 14:13:24 UTC	2022-05-06 14:16:18 UTC	*null*	*null*	42.01	-87.69	42.0
3	electric_bike	2022-08-12 16:58:45 UTC	2022-08-12 17:04:10 UTC	*null*	*null*	42.01	-87.66	42.0
4	electric_bike	2022-09-30 12:58:55 UTC	2022-09-30 13:30:37 UTC	Winchester (Ravenswood) Ave ...	*null*	41.9797739...	-87.6775726...	42.0
5	electric_bike	2022-09-16 18:41:41 UTC	2022-09-16 19:16:14 UTC	State St & Pearson St	*null*	41.8974249...	-87.6287593...	42.0
6	electric_bike	2022-06-17 10:00:19 UTC	2022-06-17 10:22:02 UTC	Broadway & Argyle St	*null*	41.9737001...	-87.6597398...	42.0
7	electric_bike	2022-09-23 10:06:38 UTC	2022-09-23 10:09:18 UTC	Broadway & Thorndale Ave	*null*	41.9897654...	-87.6601600...	42.0
8	electric_bike	2022-05-20 12:20:35 UTC	2022-05-20 12:28:16 UTC	Clark St & Winnemac Ave	*null*	41.9733316...	-87.667843	42.0
9	electric_bike	2022-04-01 13:45:12 UTC	2022-04-01 13:49:15 UTC	*null*	*null*	42.0	-87.71	42.0
10	electric_bike	2022-11-11 19:36:09 UTC	2022-11-11 19:39:21 UTC	*null*	*null*	42.0	-87.69	42.0

7. Project structure

The initial stages of the project were carried out in BigQuery, where 12 CSV files corresponding to the twelve months of data for the year 2022 were loaded. After the preliminary data processing, I consolidated them into a single file in a Google Cloud Storage bucket. I then connected to this bucket in a Google Colab notebook, loading the data into a variable for further use in the code.

8. Key statistics and conclusions

Nearly 170,000 outliers were removed from the collection.

The average ride time for the casual user is longer, at about 18.83 minutes compared to 12.1218 for the member user.

The highest number of rides overall occurs on Saturdays, with roughly equal, lower levels for the rest of the days of the week (88.3 thousand compared to about 72-78 thousand). A certain trend can be seen in the number of rides by day of the week but by type of user. Here it can be noted that users who are subscribers to the service have a higher number of rides during the week, Monday through Friday, while occasional users are more likely to choose weekends. It can be assumed that subscribers opt for the service as a method of travel to work, so they purchase access for a certain period, while occasional users are presumably those who want to use the service on the weekend, for example, for entertainment, sightseeing around the city.

It also appears that the longer average travel time for occasional users may be due, for example, to the fact that they choose to take a more leisurely, touristic approach and thus potentially a slightly longer one than the familiar, defined route taken to work by member users.

60% of all rides in the dataset are those of member users, while the remaining 40 are occasional users. The most common type of bicycle is the electric bike, for both casual users (22% of all rides in the dataset) and member users (28.87% of all rides in the dataset). The most popular station is Streeter Dr & Grand Ave.

Since the data contained coordinates written with precision to a high number of decimal places, the KNN model achieved 95-96% precision on the test set, for filling in missing station names.

9. Potential improvements

Section with machine learning models is a simplistic one as of now and some updates are considered, like testing more types of algorithms with hyperparameter tuning.