Final Proposal

Matas LAUZADIS[1]

[1] Undergraduate Student, Department of Geography and Geographic Information Science, University of Illinois at Urbana-Champaign, IL 61801; email: matasl2@illinois.edu

Divvy is a bicycle sharing system located in the City of Chicago, operated by Lyft for the Chicago Department of Transportation. The idea of introducing a bicycle sharing system to Chicago first began in 2007, when Mayor Richard M. Daley visited Paris, France and tested their Vélib' system. Immediately after returning from his trip, he requested companies to submit proposals to satisfy this need. However, the submitted plans were too expensive at the time, and a system was not implemented.

A few years later, on June 28th, 2013, Divvy set out 750 bikes at 75 stations in downtown Chicago near the Loop. Their expansion was further fueled by Rahm Emanuel in 2019, when he proposed a 9-year contract in which Lyft would have exclusive rights to operate in the City of Chicago. This contract requires Lyft to invest $50 million, adding 175 stations and 10,500 bikes in the process. By 2021, they are slated to expand to all 50 wards of the city, and they are currently beginning processes to deploy bikes and stations in the South Side of the city. After this newest installation, they are expected to have 16,500 bikes and 800 stations in total.

Divvy publicly provides their bicycle sharing system's data in CSV files which are released monthly and hosted on an Amazon Web Services (AWS) bucket [1]. Prior to April, 2020, the data was released quarterly. In order to use this data, we must accept Divvy's data license agreement, which prohibits actions such as selling the data and attempting to deanonymize the users [2]. The most important component of the data is the trips. Each trip is anonymized and includes the start day and time, the end day and time, the start location, the end location, and the rider type. The data has been processed by Divvy to remove trips that are taken by Divvy staff, as well as any trips that were below sixty seconds in length, which are considered false-starts.

In addition to hosting historical data in CSV format, Divvy also provides access to a General Bikeshare Feed Specification (GBFS) JavaScript Object Notation (JSON) feed [3]. GBFS is a data specification created by the North American Bike Share Association (NABSA) [4]. This specification provides a common format that all bike share system provider members are obligated to follow. Specific details can be read in the specification document, which is publicly hosted on NABSA's GitHub organization [5]. JSON is a popular data storage format consisting of key-value pairs and arrays organized in a human-readable text file. Since its inception, it has become the de facto standard data structure, and is used by almost every web service.

My hypothesis is that Divvy rides will be more prevalent and shorter within the Loop, and less common, but longer, outside of the Loop. To evaluate this, we focused solely on the historical data. All of the available data was downloaded from Divvy's System Data site in the format of .zip files. This was then extracted using the command line, resulting in 3.4 GB of data. This may not seem large in the context of modern picture, music and video file sizes, but this data is in solely text format -- there is quite a lot of it.

A Jupyter notebook was created to perform initial import and analysis of the data [6]. Jupyter Notebook is a project hosted by Project Jupyter. Their goal is to "develop open-source software, open-standards, and services for interactive computing across dozens of programming languages." These notebooks are commonly used in the industry for quick prototype designs and for sharing the work with peers and colleagues. Within the notebook, code is structured in "cells". Variables and memory persist across these cells, so you only need to run the bits of code that you need, and not the whole file, like in compiled languages such as C++ or Java.

To store the data in a more efficient manner, MongoDB was employed. Known as "The most popular database for modern apps", MongoDB is used by top companies such as Google, Adobe, and Cisco. It is a NoSQL database, which means that data can be stored in an unstructured format, typically in JSON. The unstructured nature makes it easier to get started, allowing the developers to store data in the same way they use it in their application code. For this project, I hosted a MongoDB server on my personal desktop, but they also provide a free cloud storage solution for up to 512 MB of data, though that would have not been enough for the Divvy data.

After creating the notebook and starting the database, we began importing the data. A database connection was made using the Python module pymongo. This allows the developer to create a connection to a MongoDB database, making lookups, insertions, and deletions. The CSV files were selected using a Python module called glob. For this initial prototype, only data from January 2020 onwards was selected. After getting all of the filenames in a list from glob, we used pandas to open each file. Pandas is the most popular data science library in Python. It allows the user to work with tabular data such as CSV or Excel files, and perform rapid analysis. After opening the file, we parsed each line, which corresponds to a single ride, into JSON format. This is a simple process, because the dictionary data type in Python is analogous to JSON. This JSON data structure was then inserted into the database. Any errors were logged and reported to standard output. A final count of the inserted rides was reported, totalling 3,916,426 rides.

The data was then split into two portions, rides beginning in the Loop and ending in the Loop, and rides beginning outside of the Loop and ending outside of the Loop. Any rides that cross the Loop border were not considered. We needed to create a bounding box of the Loop in order to see which points were inside, and which were outside. The following coordinates were selected: -87.638588,41.875803,-87.62464,41.887082, corresponding to the left, bottom, right, and top of the box.

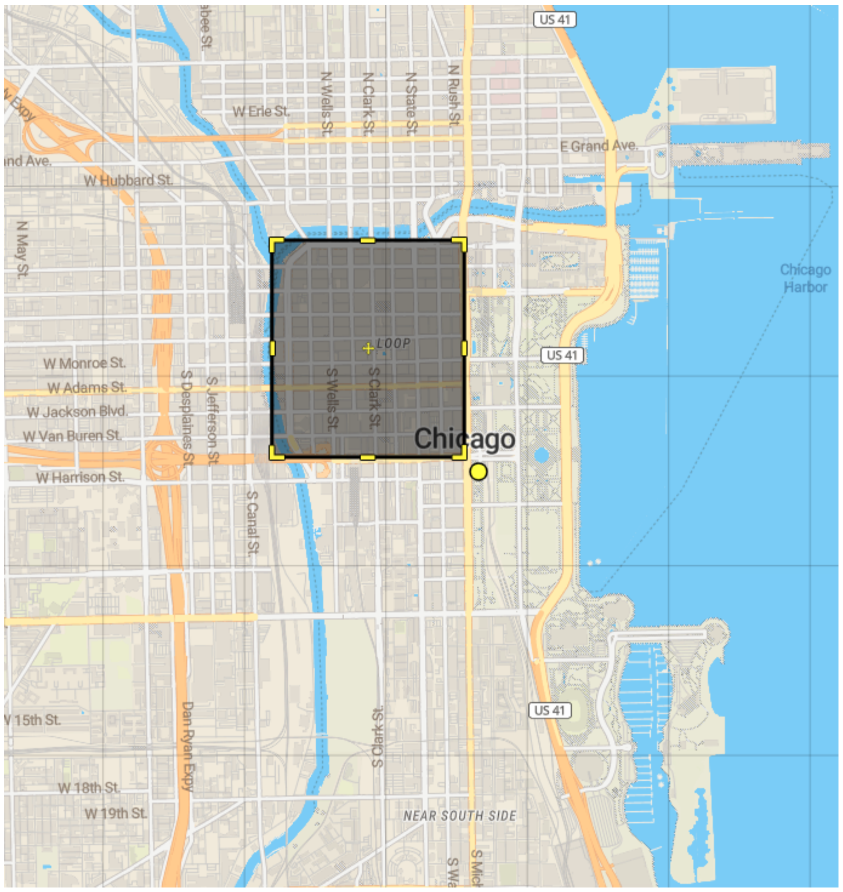

Figure 1: The Loop bounding box

After making the bounding box, we could use the coordinates to conditionally select the rides within the Loop and outside of the Loop. The following query string was used to select the bike rides inside the Loop:

```
inside_loop = client.te360.divvy.find({
    "$and": [
        {
            "start_lat": { "$gte": 41.875803, "$lte": 41.918306 },
            "start_lon": { "$gte": -87.638588, "$lte": -87.62464 },
            "end_lat": { "$gte": 41.875803, "$lte": 41.918306 },
            "end_lon": { "$gte": -87.638588, "$lte": -87.62464 }
        }
    ]
})
```

The query specifies that we are looking for rides which began and ended within the boundaries of the bounding box. After executing this query, the variable "inside_loop" will be a PyMongo cursor, which we can iterate over to find each ride. Looping over each ride, we convert the start time and end time from a string to datetime object. Then, these times are subtracted from each other, and added to a total count. Finally, the average elapsed time is calculated by dividing the number of total seconds traveled by the number of rides.

In conclusion, we calculated the average ride time within the Loop and outside of the Loop. Inside the Loop, the average ride time was 1380 seconds, or 23 minutes, with 333,232 total rides. Outside of the Loop, the average ride time was 1209 seconds, or about 20 minutes, with 870,017 total rides. So, my hypothesis was wrong. There are less rides occurring inside of the Loop, and they are longer in length on average. One reason this may be the case is that the Loop is fairly small. There is simply more area in other parts of Chicago where people can ride bikes, such as the lakefront. Additionally, Divvy does not place that many bike racks inside the

Loop, there are more bikes and racks outside.

References

[1]: “Divvy System Data.” Divvy Bikes, 2021, divvybikes.com/system-data.
[2]: “Data License Agreement.” Divvy Bikes, 2021, divvybikes.com/data-license-agreement.
[3]: “GBFS Data Source.” Divvy Bikes, 2021, gbfs.divvybikes.com/gbfs/gbfs.json.
[4]: North American Bikeshare Association. “NABSA Homepage.” North American Bikeshare Association, 10 Mar. 2021, nabsa.net.
[5]: Nabsa. “NABSA/Gbfs.” GitHub, 2021, github.com/NABSA/gbfs.
[6]: Matas Lauzadis. “mataslauzadis/te360.” GitHub, 2021, github.com/mataslauzadis/te360.