Predicting Shared Dockless Vehicle Time to Activation

Elisabeth Ericson

MUSA Capstone 2022

Why dockless micromobility?

- New transportation mode that has been both popular and controversial
- Human-scaled electric transportation suited to urban environments
- Potential to address first- and last-mile problem for neighborhoods poorly served by transit and traditional bikeshare
- Research suggests it provides more equitable access than traditional, dock-based bikeshare
- It’s a fun and convenient way to get around

The problem

{ will be familiar to anyone who's ever arrived at their favorite neighborhood Indego dock only to find it empty }

Because anyone can take a vehicle at any time, there's no guarantee it'll still be there when you need it

What are the chances that a bike or scooter listed as “available” in an app will be taken before I can get to it?

How can I model how long an inactive bike or scooter will remain idle between trips?

Bike & scooter location data

- Clean datasets not publicly available
- Real-time API publishes coordinates for all inactive vehicles
- Standardized format: General Bikeshare Feed Specification (GBFS)

Micromobility in Washington, D.C.

- D.C. Department of Transportation (DDOT) requires all dockless micromobility operators to publish real-time vehicle data
- Includes city bikeshare system (Capital Bikeshare, operated by Lyft) and five private companies (Bird, Helbiz, Lime, Lyft, Spin)
- Initially scraped data from Capital Bikeshare only; later expanded to all six providers

From last time:

Two issues with the bikeshare data

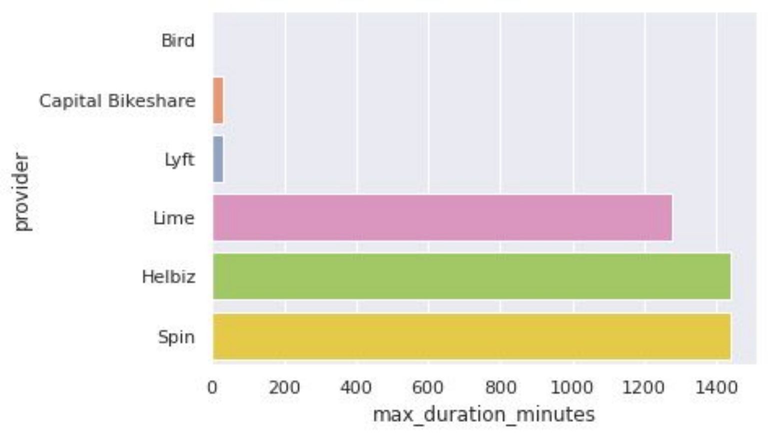
- ➔ **Same bike, different IDs:** Bicycle IDs reset every 30 minutes
- ➔ **Same ID, different instance:** One ID can represent more than one period of inactivity (if someone activates a bike, rides it, and deactivates it again)

bike_id	is_reserved	is_disabled	type	lon	lat	timestamp
002604d3123025e6e2fa8384ee72d2a6	0	0	electric_bike	-76.974229	38.932343	09:30:07
002604d3123025e6e2fa8384ee72d2a6	0	0	electric_bike	-76.974219	38.932373	09:31:09
002604d3123025e6e2fa8384ee72d2a6	0	0	electric_bike	-76.974228	38.932403	09:32:11
002604d3123025e6e2fa8384ee72d2a6	0	0	electric_bike	-76.974228	38.932403	09:33:13
002604d3123025e6e2fa8384ee72d2a6	0	0	electric_bike	-76.974240	38.932374	09:34:14

Can I finally get out from under my data issues?

`even in an ostensibly standardized format, data varies substantially between providers`

	provider	max_duration	max_duration_minutes
0	Bird	0 days 00:01:00	1
1	Capital Bikeshare	0 days 00:29:00	29
2	Lyft	0 days 00:30:00	30
3	Lime	0 days 21:16:00	1276
4	Helbiz	0 days 23:59:00	1439
5	Spin	0 days 23:59:00	1439

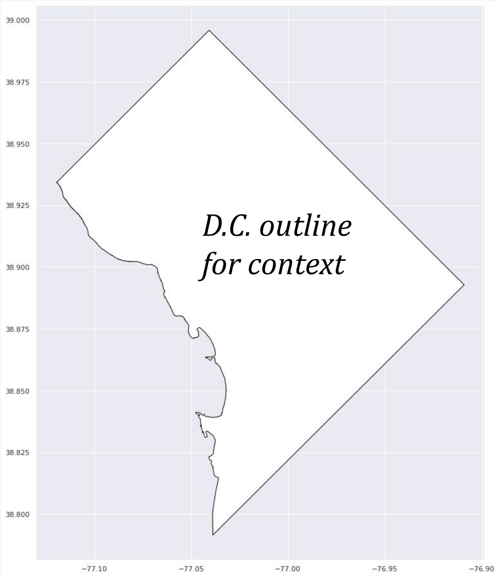


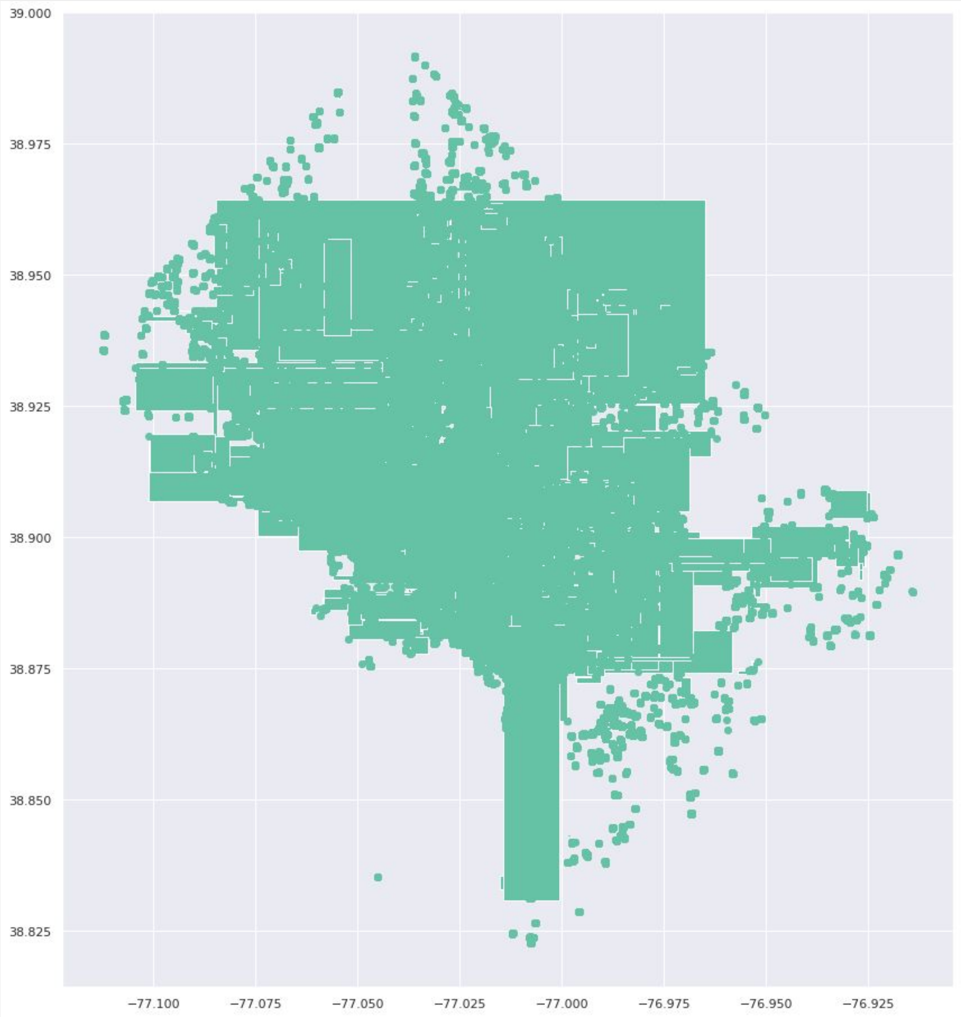
Time each ID is present in the data, by provider

(This has implications for which is easiest to model)

Still, nothing is ever easy

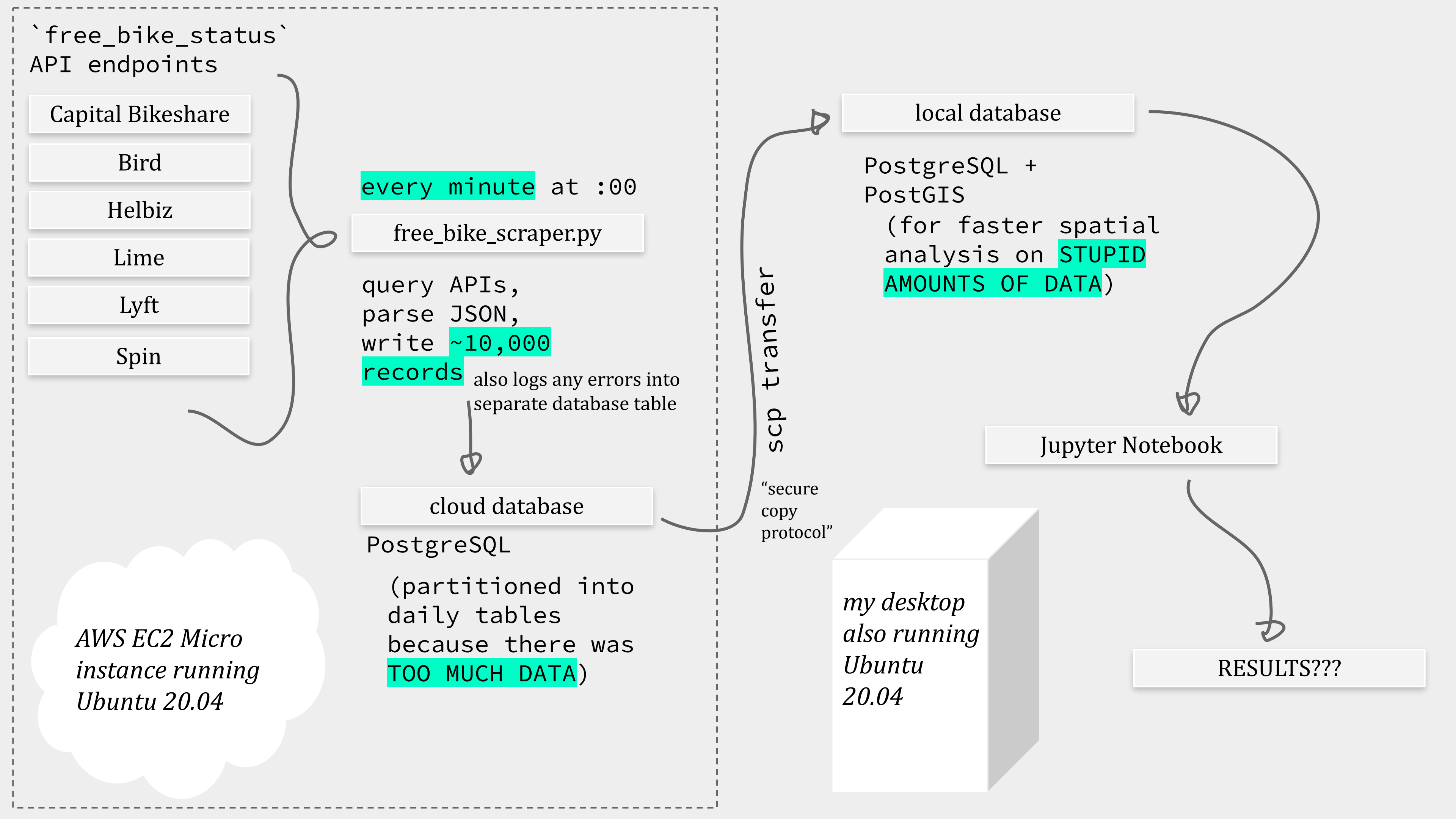
(spatial bounding boxes for each Lime scooter ID)





Data pipeline and infrastructure

this turned into most of the project, to be honest


`free_bike_status`
API endpoints
Capital Bikeshare
Bird
Helbiz
Lime
Lyft
Spin
every minute at :00
free_bike_scraper.py
query APIs,
parse JSON,
write ~10,000
records
also logs any errors into
separate database table
cloud database
PostgreSQL
(partitioned into
daily tables
because there was
TOO MUCH DATA)
AWS EC2 Micro
instance running
Ubuntu 20.04
scp transfer
“secure
copy
protocol”
local database
PostgreSQL +
PostGIS
(for faster spatial
analysis on STUPID
AMOUNTS OF DATA)
Jupyter Notebook
RESULTS???
my desktop
also running
Ubuntu
20.04

things I had never done before this project

- scraped anything on a recurring basis
- written a stand-alone Python script
- used a scheduler to run something repeatedly
- worked with data in the millions of records, never mind tens of millions
- used SQL really at all
- set up a Postgres database (or several)
- partitioned a database table
- written Python error handling
- used AWS or any other cloud instance
- used PostGIS for spatial analysis
- accidentally deleted my entire database (with 60 million irreplaceable records) because I made a typo in the terminal
- …and many more things I don't remember

```
"""Scrapes D.C. dockless vehicle locations every minute.
Scrapes General Bikeshare Feed Specification (GBFS)
free_bike_status API endpoints for all current Washington, D.C.
dockless vehicle providers as of 2022-04-16. Saves data to a
PostgreSQL database. Logs any errors to a separate table.
"""


import traceback
import time


import requests
import psycopg
import schedule


from datetime import datetime


def main():
    """Schedules scraper to run once every minute."""

    schedule.every().minute.at(':00').do(scrape_all)

    while True:
```

Questions?

```
        try:
            scrape_dockless_vehicles(provider,
time_scraped=time_scraped)


        except:
            time_failed =
datetime.now().astimezone().isoformat(timespec='seconds', sep='
')
            traceback_text = traceback.format_exc()


            with psycopg.connect("dbname=capstone-aws
user=ubuntu") as conn:


                with conn.cursor() as cur:
                    cur.execute("""
                        INSERT INTO errors (time_scraped,
provider, time_failed, traceback)
                        VALUES (%s, %s, %s, %s)
                        """, (time_scraped, provider, time_failed,
traceback_text))


                    conn.commit()
```