Project Milestone Meeting

Crew: Nathan Ryan, Roy Kim, Sharon Obiefuna, Jiayang Liu

Our Project

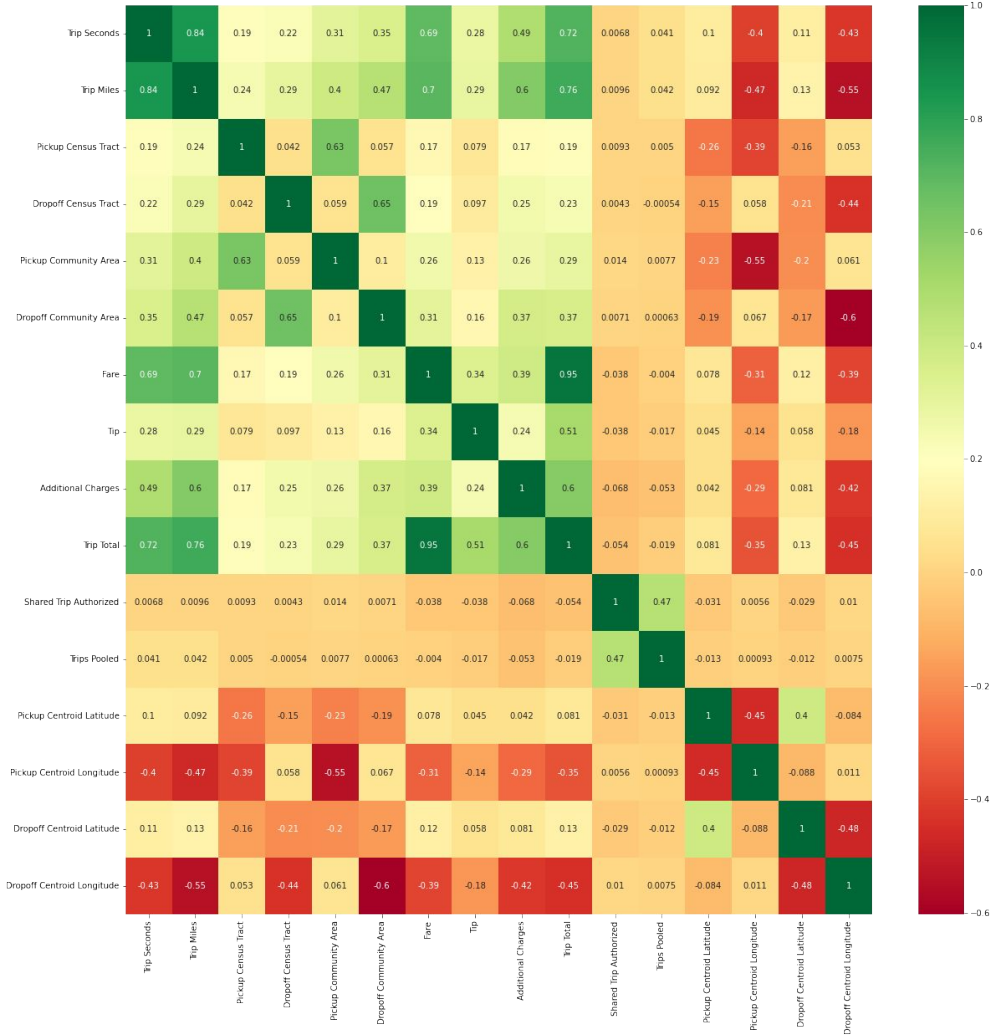
- Examine the dynamics of trip costs on “transportation network providers” like Uber and Lyft

- Discover how the cost of trips and/or the amount of tips depends on other factors like trip length, duration, pickup and dropoff location, and more.

Exploratory Data Analysis

Potential Features:

Fare, Total, Time (in seconds), Trip miles, Pickup Centroid Location, Dropoff Centroid Location. Based on correlation heatmap



Difficulties So Far

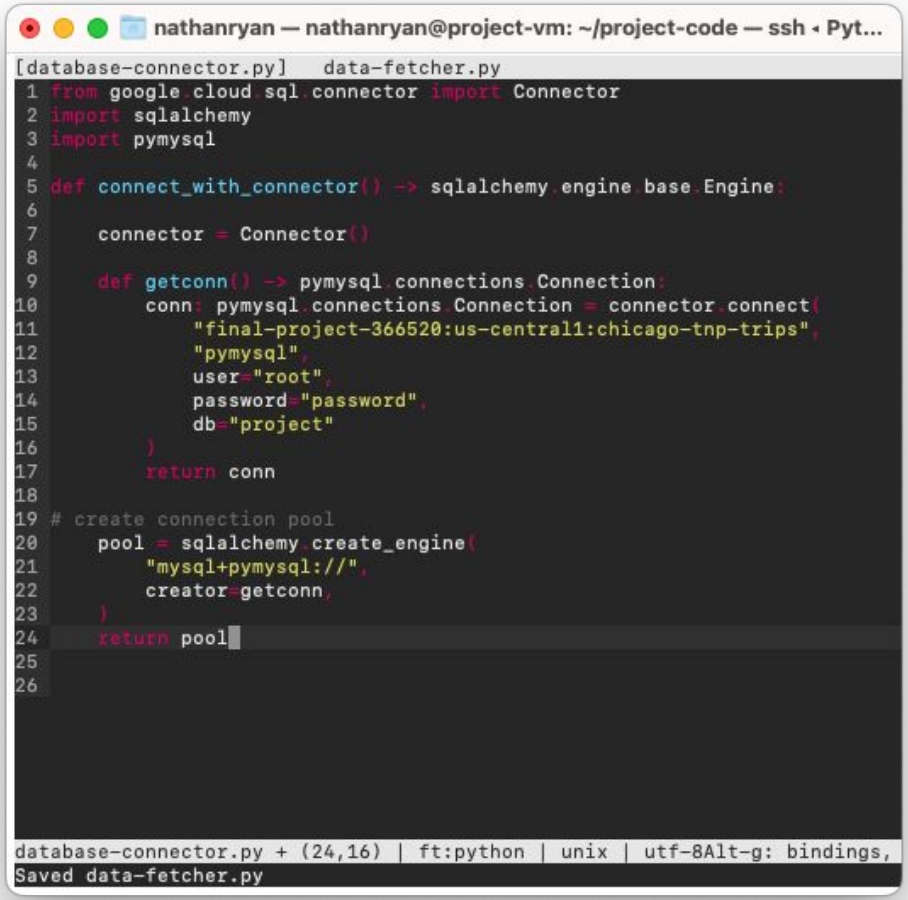
- We have had a lot of difficulty working with the size of our dataset so far, as the full dataset is over 50 GB
- Now, after acquiring Google Cloud credits, we can move our data retrieval and processing workload to remote machines, allowing us to connect to these machines for analysis using Tableau and other software
- As we continue, if the Google Cloud resources do not help, we were hoping to learn more by analyzing the data temporally by having to break them up

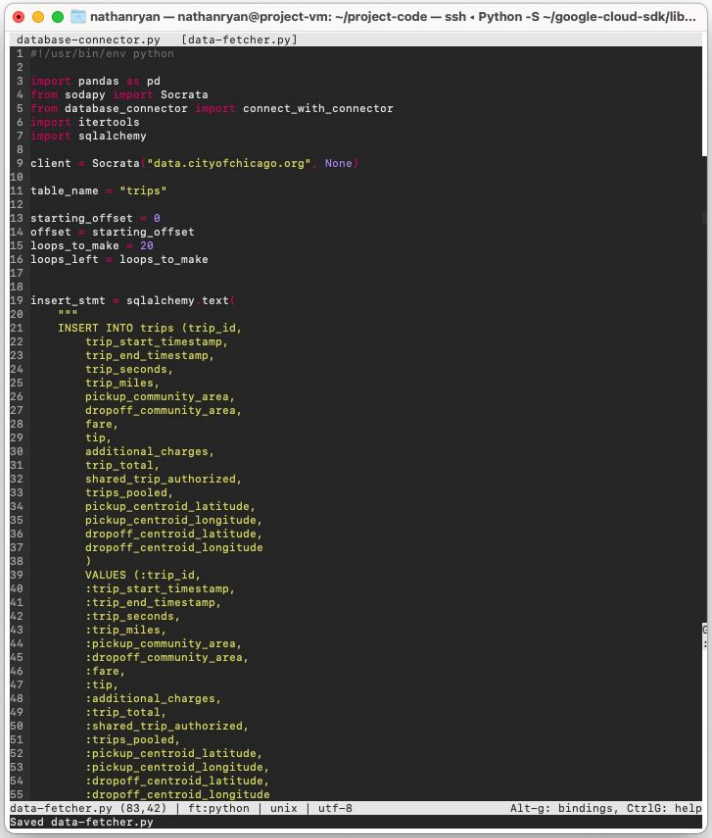
ML Models

1. Geopandas
- Features: Enables the use and manipulation of geospatial data in Python
- Geospatial Visualization.
2. K Means clustering
- Utilize elbow method to derive a value of k and see what centroids develop.
- We likely have to scale down the dimensions.
- See potential conclusions that may not immediately be obvious
3. Geospatial Regression
- Need to account for the spatial dependencies as part of your dependent variable
4. Logistic Regression between cost/tip and other fields
- Examine which factors have a higher correlation with cost/tip.

Progress

- Acquired Google Cloud credit
- Set up MySQL database to contain dataset from City of Chicago
- Set up Google Cloud virtual machine to capture and process dataset
- Wrote Python scripts to connect to database and insert trip data into database table
- Preliminary research on desired statistical and ML models to use


nathanryan — nathanryan@project-vm: ~/project-code — ssh ◂ Pyt...
[database-connector.py] data-fetcher.py
1 from google.cloud.sql.connector import Connector
2 import sqlalchemy
3 import pymysql
4
5 def connect_with_connector() -> sqlalchemy.engine.base.Engine:
6
7 connector = Connector()
8
9 def getconn() -> pymysql.connections.Connection:
10 conn: pymysql.connections.Connection = connector.connect(
11 "final-project-366520:us-central1:chicago-tnp-trips",
12 "pymysql",
13 user="root",
14 password="password",
15 db="project"
16)
17 return conn
18
19 # create connection pool
20 pool = sqlalchemy.create_engine(
21 "mysql+pymysql://",
22 creator=getconn,
23)
24 return pool
25
26
database-connector.py + (24,16) | ft:python | unix | utf-8Alt-g: bindings,
Saved data-fetcher.py


nathanryan — nathanryan@project-vm: ~/project-code — ssh ◂ Python -S ~/google-cloud-sdk/lib...
database-connector.py [data-fetcher.py]
1 #!/usr/bin/env python
2
3 import pandas as pd
4 from sodapy import Socrata
5 from database_connector import connect_with_connector
6 import itertools
7 import sqlalchemy
8
9 client = Socrata("data.cityofchicago.org", None)
10
11 table_name = "trips"
12
13 starting_offset = 0
14 offset = starting_offset
15 loops_to_make = 20
16 loops_left = loops_to_make
17
18
19 insert_stmt = sqlalchemy.text(
20 """
21 INSERT INTO trips (trip_id,
22 trip_start_timestamp,
23 trip_end_timestamp,
24 trip_seconds,
25 trip_miles,
26 pickup_community_area,
27 dropoff_community_area,
28 fare,
29 tip,
30 additional_charges,
31 trip_total,
32 shared_trip_authorized,
33 trips_pooled,
34 pickup_centroid_latitude,
35 pickup_centroid_longitude,
36 dropoff_centroid_latitude,
37 dropoff_centroid_longitude
38)
39 VALUES (:trip_id,
40 :trip_start_timestamp,
41 :trip_end_timestamp,
42 :trip_seconds,
43 :trip_miles,
44 :pickup_community_area,
45 :dropoff_community_area,
46 :fare,
47 :tip,
48 :additional_charges,
49 :trip_total,
50 :shared_trip_authorized,
51 :trips_pooled,
52 :pickup_centroid_latitude,
53 :pickup_centroid_longitude,
54 :dropoff_centroid_latitude,
55 :dropoff_centroid_longitude
data-fetcher.py (83,42) | ft:python | unix | utf-8 Alt-g: bindings, Ctrl-G: help
Saved data-fetcher.py

Next Steps

- Continue trying to use entire dataset; if not feasible, constrain to just post-COVID trip records
- Come to a consensus on the ML approach that is most appropriate for our dataset