Vibhav Vasudevan

vibhav.vasudevan@gmail.com

REPORT

Test 1 - Data science for public transit; VISSIM modeler for parking

Indian Institute of Science (IISc), Bengaluru Summer Internship: Round 1

Question 1

1. Here are the values that were required to be calculated:

```
a) Maximum duration of the trip: 518 minutes
b) Minimum duration of the trip: 1 minutes
c) Total trips that had duration as minimum duration: 89
   trips
d) Percentage of circular trips: 2.4776425744025805 %
```

The function was run both on Google Colab and Jupyter Notebook and here were the observations that followed for the run time:

a) Google Colab:

```
Runtime of the function:  0.6592812538146973 seconds
```

b) Jupyter Notebook:

```
Runtime of the function:  1.880969762802124 seconds
```

Assumptions:

i) All trips start and finish on the same day.
ii) The times are in the 24-hour format in all rows.

Findings:

i) A total of 48 trips started and ended in the same minute, leading us to believe that these are faulty/inaccurate.

ii) Approximately 2.5% of all trips were circular trips leading to the conclusion that many users returned their bikes to the original depot from which it was originally taken.

2. The required values to be calculated:

a. `The number of feasible pairs are:  45174`
b. `Runtime of the function:  151.0345494747162 seconds (Google Colab)`

Assumptions:

i) The dataset is sorted appropriately based on the 'started_at' column. With this assumption, for the inner for loop, we only have to iterate from the i+1th element until the end.

ii) The times are in 24-hour format.

Findings:

i) The number of feasible pairs was a large amount (45174) which mean this depot system is working well and the bikes are being used efficiently.

ii) 4874 trips were completed in the span from 6:00 am to 6:00 pm indicating this is a very popular timespan for travel (71.16% of all trips)

3. Findings:

i. The first hundred records were retrieved from the table however due to compatibility and time issues the osmnx module usage was limited and hence incomplete.

ii. Dijkstra would be appropriate to implement since the edge lengths would be positive regardless of negative coordinates.

Out[28]:

	trip_id	started_at	ended_at	start_lat	start_lng	end_lat	end_lng
0	1	01-02-2023 00:03	01-02-2023 00:13	38.905470	-77.002130	38.906299	-76.983221
1	2	01-02-2023 00:06	01-02-2023 00:19	38.907693	-77.071512	38.909858	-77.038566
2	3	01-02-2023 00:07	01-02-2023 00:14	38.915544	-77.038252	38.916137	-77.022003
3	4	01-02-2023 00:07	01-02-2023 00:26	38.909701	-77.085646	38.935839	-77.106913
4	5	01-02-2023 00:10	01-02-2023 00:20	38.905509	-77.065246	38.898302	-77.046913
...	...	...	...	...	...	...	...
95	96	01-02-2023 05:45	01-02-2023 05:49	38.920000	-77.050000	38.900000	-77.050000
96	97	01-02-2023 05:45	01-02-2023 05:55	38.873755	-77.089233	38.887010	-77.095257
97	98	01-02-2023 05:45	01-02-2023 05:55	38.893410	-77.076348	38.905711	-77.047318
98	99	01-02-2023 05:46	01-02-2023 05:47	38.876300	-77.003700	38.875010	-77.002400
99	100	01-02-2023 05:46	01-02-2023 06:15	38.907444	-76.986813	38.892441	-77.048947

Question 2:

1) Assumptions:

i) The trajectory_id and individual_id is in order for the same user that is all ids for the same individual and trajectory are one after the other.

ii) The distances are on a sphere so the Haversine formula can be used

Findings:

Here was the output obtained for each individual user presented in a dictionary:

```
{1.0: 51.173037422073385,
2.0: 5.8025627163784055,
3.0: 16.15471488425598,
4.0: 51.173037422073385,
5.0: 1.7681346389773294,
6.0: 4.933166858318395,
7.0: 1.9374428932558183,
8.0: 5.511713297383309,
9.0: 29.815636815485266,
10.0: 21.572423083412065,
11.0: 132.85130896193604,
```

```
12.0: 10.903093068267129,
13.0: 6.450491193157014,
14.0: 0.608463817152366,
15.0: 2.326599199941408,
16.0: 15.570487190647356,
17.0: 3.963051257161561,
18.0: 39.13130388133462,
19.0: 2.5920579690345047,
20.0: 12.949462819761612,
21.0: 1.0794990959577775,
22.0: 11.631945050526932,
23.0: 32.82356483009358,
24.0: 3.562206399056561,
25.0: 2.2373412109814668,
26.0: 1.9099755862551462,
27.0: 29.01673860454691,
28.0: 2.9139131144340444,
29.0: 29.34720652835843,
30.0: 17.504654314461785,
31.0: 3.96485191842587,
32.0: 0.9733539383947729,
33.0: 45.560046342221945,
34.0: 111.52571509400109,
35.0: 4.000814653824774,
36.0: 17.17687795688196,
}
```

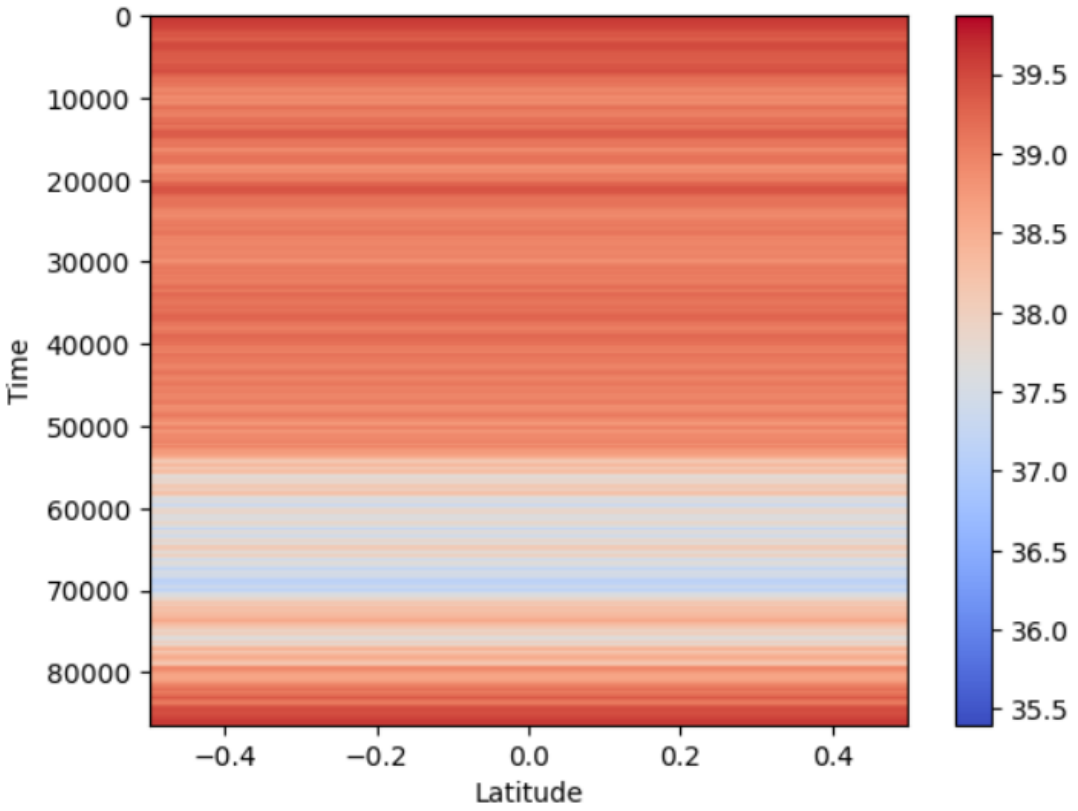
2) Assumptions:

i) The spatial range assumed is the coordinate range [39.985,40.0].

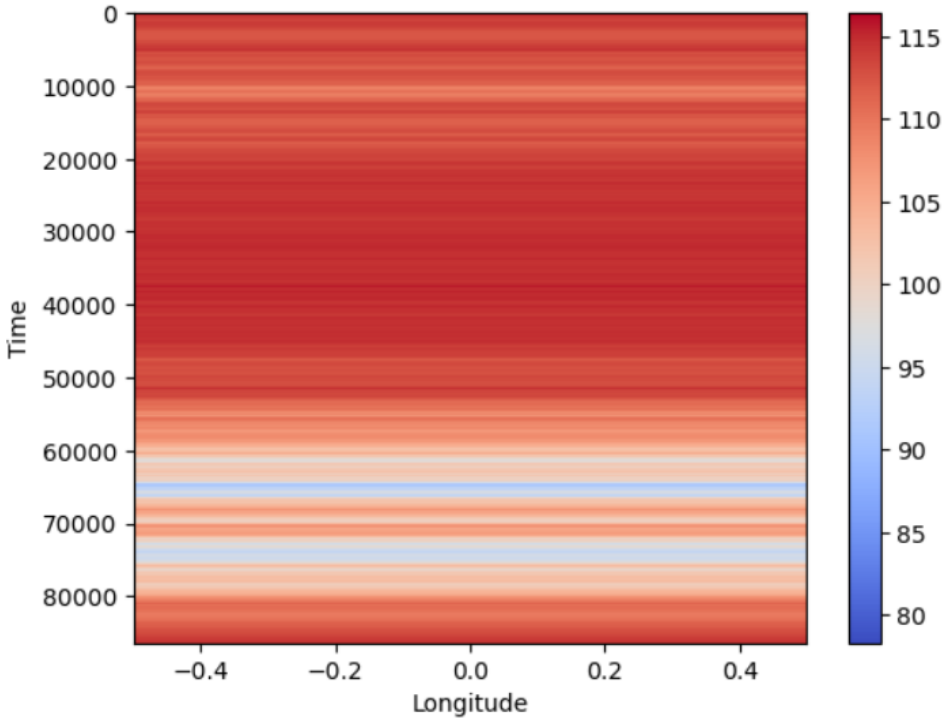
ii) The temporal range assumed is between 4 and 6 pm.

Findings:

i) Latitude vs Time heatmap



ii) Longitude vs Time Heatmap



3) Here is the proposed problem and my suggested solution:

Access to such a large, valuable dataset will be extremely useful in solving numerous modern day problems.

One problem for example would be to reduce congestion and overcrowding in areas that are seen to be frequently visited from the dataset, also noting the timings when high concentration is seen in these particular areas. This can be done with the help of heatmaps and various other data visualization techniques.

The population statistics can be studied at different points of the day and public transit can be adjusted in order to be made more frequent at these high-density timings.

In addition to this, the routes that are not taken as frequently can also be made more viable and accessible by developing methods of commute through these regions as well.

In traffic-intense cities like Bangalore especially, this could come to prove extremely useful as an alternate route could be much quicker, considering the fact that many roads are often under maintenance and hence leading to large delays.

One interesting business idea that can evolve from this is the idea of carpooling, which I believe can be an extremely profitable and beneficial in India. Using the GPS dataset, certain hubs can be selected which can be made pickup locations. This solves multiple issues, like avoiding the need to travel in a crowded bus, reduction in cost as this would be much cheaper than travelling by a normal cab.

This would also be environmentally friendly as the number of vehicles on the road will reduce. This idea can easily be implemented by existing commute apps like Uber, Ola etc. by using data to find individuals travelling to a nearby destination every day and setting up hubs in popular locations.