Data Processing at Scale (CSE 511) Portfolio Report

Anurag Banerjee
School of Computing and Augmented Intelligence, Ira A. Fulton Schools of Engineering
Arizona State University
Tempe, United States
abaner40@asu.edu

***Abstract*—This document contains the solution, results and lessons learned for Project 1 and 2 of the course Data Processing at Scale. Project 1 was to query a NoSQL database and find businesses from a collection based on given set of conditions like city or maximum distance from a location. Project 2 was to do hotspot analysis on the NYC taxi trip dataset and find the hotness of zones and cells.**

Keywords—NoSQL, haversine, Python, Hot Spot Analysis, Spatial, Spatiotemporal, Getis-ord, Scala, Apache Spark

I. Introduction

This document is a portfolio report of the projects done for the Data Processing at Scale (CSE 511) course taken in Fall 2022. We had to complete 2 projects in total for this course.

For the 1st project, 2 functions needed to be implemented in python to fetch data from a given NoSQL database based on some prerequisites. The job of the first function was to find businesses based on a given city. The function would take as input the name of the city, the name of the collection from NoSQL database that contained all the businesses and the location to which the results needed to be written. The work done by the function was to find all the businesses in the given collection which were in the provided city name and then write the results to a file in a specific format into the location that was provided. The job of the second function was to find businesses based on location. The function would take as input the coordinates of a location as a string in a comma separated form, a list of categories of businesses among which to search from, a maximum distance in which businesses were to be searched, the name of the collection from NoSQL database that contained all the businesses and the location to which the results needed to be written. The work done by the function was to find all the businesses within the given maximum distance of the provided location coordinated that belonged to any of the categories provided as input and save the results in the location provided. The distance was calculated using the haversine formula [1].

For the 2nd project, we needed to do hot zone analysis and hot cell analysis. In hot zone analysis, a join needed to be performed on a rectangle dataset (the zones in New York city) and a point dataset (pickup points for New York taxi trips). The rectangular dataset was provided in the form of the coordinates of the diagonals of the rectangle. The result of this join operation would give the number of points within each rectangular zone. The greater the number of points in a zone the hotter the zone was which meant that a greater number of taxi trips started from that zone. The task was to calculate the hotness of all the zones that were provided in the rectangular dataset. In hot cell analysis, we were required to find the hotspots in the provided dataset, but we also had to take time into account. The cell unit sizes were provided in terms of latitude and longitude degrees and 1 day was to be used as the time step size. The hotness of each cell was calculated using the getis-ord statistic [2] and the results were saved accordingly.

II. Description of Solution

A. *Project 1 – Function 1*

For this part, a function needed to be written in python which would be able to fetch all the businesses from a NoSQL collection and filter the businesses that belonged to a particular city and write the results back to a provided location. In order to perform this, I fetched all the businesses that were present in the collection that was provided to us and filtered those that belonged to the city that was provided to us. The results were stored in a list. Then, a single traversal of the list, that was generated in the step before, was done and the details of each of the businesses present in the array was written in a new line in the location provided in the format Name$FullAddress$City$State.

B. *Project 1 – Function 2*

Next, a function needed to be written in python which found out the businesses that were within a given maximum distance within a provided location and belonged to one of the categories that were provided. In order to perform this, I traversed through all the business that were present in the collection that was provided and in turn traversed through all the categories of each of the businesses. If any category of a business was present in the list of categories that was provided the given business was added to a list. Then this list was traversed and for every business in the list it was checked whether the distance of this business was less than or equal to the maximum distance from the location that was provided to us. If true, the name of the business was written in the location provided as input to the function. The formula for calculating the distance between two geographic coordinates was provided to us as the haversine formula [1]. I used this formula in a helper function to calculate the distance between two coordinates. The helper function would take in the latitudes and longitudes of both coordinates and return the distance between the coordinates according to the haversine formula. The formula used was as follows [1]:

Haversine formula:

$$a = \sin^2(\Delta\varphi/2) + \cos \varphi 1 \cdot \cos \varphi 2 \cdot \sin^2(\Delta\lambda/2)$$

$$c = 2 \cdot \text{atan2}(\sqrt{a}, \sqrt{(1-a)})$$

$$d = R \cdot c$$

where φ is latitude, λ is longitude, R is earth's radius (mean radius = 6,371km)

note that angles need to be in radians to pass to trigonometric

functions.

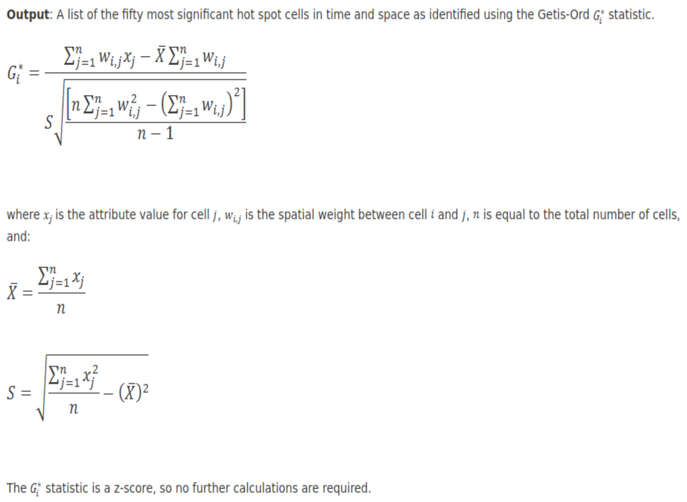
C. *Project 2 – Hot Zone Analysis*

The task that was needed to be done in hot zone analysis was to calculate the hotness of each rectangular region that was provided in the rectangular dataset (the zone dataset). The rectangles were provided in the form of the 2 coordinates of a diagonal. The starting point of each taxi trip was provided in the taxi trip dataset. In order to complete the task a helper function was written which was used to determine whether a point was inside a rectangle or not. This helper function took the coordinates of the rectangle and the point and returned true if the point was inside the rectangle. This function basically compared the x coordinate of the point with the minimum and maximum x coordinate of the diagonals and checked whether the x coordinate was in between both of them. Similarly, the checks were performed for the y coordinates. If the coordinates of the point passed all the checks, then the helper function returned true, else the helper function returned false. The join was performed between the zone dataset and the taxi trip dataset. The rows from the two datasets were joined only if the helper function described above returned true. After performing the join, the rows were grouped based on the rectangle and then the count function was used to count the number of rows per rectangle. The rows were then arranged according to the coordinates of the rectangle. The final results which contained the hotness of the zones were written into a file.

D. *Project 2 – Hot Cell Analysis*

The task that was needed to be done in hot cell analysis was to calculate the hotness of each cell that was defined in space and time by the given set of parameters (0.01 latitude and longitude degrees, 1 day time step). In order to do this some pre-processing was done on the provided dataset. Then, I filtered the dataset according to the maximum x, y, and z coordinates provided to us (z being the time of the taxi trip) and stored it in a dataframe named filteredData. During this operation, I grouped the data by their x, y, and z coordinates and the total number of trips for a particular coordinate was calculated using the sum function which was stored as $x_{j.}$, Next, the sum of all x_j and $(x_j)^2$ obtained in the previous step was calculated. Next, a helper function named isNeighbour was defined in scala to check whether a cell was a neighbour of another cell or not. In this helper function, it was checked whether the x, y, and z coordinates were within 1 unit above or below another cell. Another utility function named getNeighboursCount was defined to get the count of the number of possible neighbours for a particular cell. In this function, it was checked that out of all possible neighbours of a cell which were between the maximum x, y, and z coordinates provided to us. Next, I took a self-join of the filteredData and the condition was that cell defined by the rows should be a neighbours (this condition was checked using the helper function isNeighbour). Then the total number of possible neighbours was found out for a particular cell using the getNeighboursCount function. This was $\Sigma w_{i,j}$.

The sum of count of the neighbour trips gave us $\Sigma w_{i,j} x_{j.}$ All these values were plugged into the getis-ord [2] statistic formula shown in Fig.1.


Output: A list of the fifty most significant hot spot cells in time and space as identified using the Getis-Ord G_i^* statistic.

$$G_i^* = \frac{\sum_{j=1}^{n} w_{i,j} x_j - \bar{X} \sum_{j=1}^{n} w_{i,j}}{S \sqrt{\frac{\left[n \sum_{j=1}^{n} w_{i,j}^2 - \left(\sum_{j=1}^{n} w_{i,j}\right)^2\right]}{n-1}}}$$

where x_j is the attribute value for cell j, $w_{i,j}$ is the spatial weight between cell i and j, n is equal to the total number of cells, and:

$$\bar{X} = \frac{\sum_{j=1}^{n} x_j}{n}$$

$$S = \sqrt{\frac{\sum_{j=1}^{n} x_j^2}{n} - (\bar{X})^2}$$

The G_i^* statistic is a z-score, so no further calculations are required.

Fig. 1.

After obtaining the z scores from the getis-ord statistic formula shown above the cells were arranged in descending order by their z scores. This gave us the spatio temporal hotness of the cells.

III. Results

A. *Project 1 – Function 1*

The function for finding business based on city wrote the results to a file. The results are shown in Fig. 2 below. These results are based on the NoSQL collection that was provided to us.

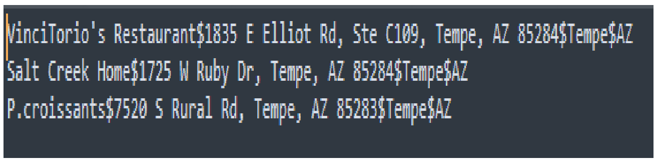

```
VinciTorio's Restaurant$1835 E Elliot Rd, Ste C109, Tempe, AZ 85284$Tempe$AZ
Salt Creek Home$1725 W Ruby Dr, Tempe, AZ 85284$Tempe$AZ
P.croissants$7520 S Rural Rd, Tempe, AZ 85283$Tempe$AZ
```

Fig. 2

B. *Project 1 – Function 2*

The function for finding businesses based on maximum distance from a location and based on categories also wrote the results to a file. The results are shown below in Fig. 3. These results were based on the NoSQL collection provided to us.

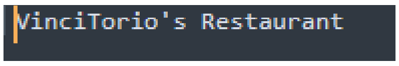

```
VinciTorio's Restaurant
```

Fig. 3

C. *Project 2 – Hot Zone Analysis*

The output for the first few lines for calculating hottest zones are shown in Fig. 4 below.

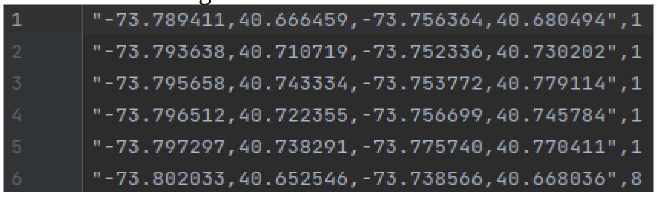
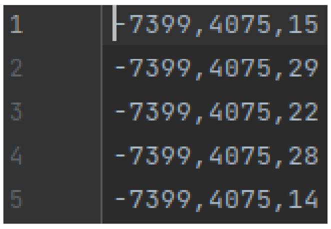

```
"-73.789411,40.666459,-73.756364,40.680494",1
"-73.793638,40.710719,-73.752336,40.730202",1
"-73.795658,40.743334,-73.753772,40.779114",1
"-73.796512,40.722355,-73.756699,40.745784",1
"-73.797297,40.738291,-73.775740,40.770411",1
"-73.802033,40.652546,-73.738566,40.668036",8
```

Fig. 4

D. *Project 2 – Hot Cell Analysis*

The output for the first few lines of the function to calculate the hottest cells is shown in Fig. 5 below.



```
-7399,4075,15
-7399,4075,29
-7399,4075,22
-7399,4075,28
-7399,4075,14
```

Fig. 5

E. *Interesting Findings:*

Some of the places where I think the results can be used for are as follows:

- The function to find the particular business in city can be used to find any business, hotel, school etc in a given city when we have collection which has all the businesses, hotels, schools etc. Suppose we need to build a website for hotel booking, we can take as input from the customer the city which they are going to and return the list of hotels available to them.
- The function to find businesses based on the maximum distance from a location and within certain categories can be used to provide the users with a more fine-grained search than the above example. Suppose we have an hotel booking application and we know that the user has arrived at an airport then we can suggest hotels based on the maximum distance the user provides. The categories in that function can be used in a similar way to provide search options such as 4-star hotels, boutique hotels, resort type hotels etc.
- The taxi zone hotness can be used by cab operating companies to assign cabs to the zones in proportion to their hotness. The hot zone analysis function can be used analogously by governments for health care and hospitals. Suppose we have previous years data of diseases and zones where those diseases spread. The government can build hospitals dedicated to a particular disease in the hottest zones.
- The cell hotness can be used by cab operating companies by anticipating the demands by checking the data of previous years and their hotness and moving cabs to anticipated hot cells at a particular time. The hot cell analysis can be used analogously to shift resources like doctors or beds between hospitals. When we have the data about diseases and their timings from previous years we can anticipate the demands for hospitals, beds, doctors, and medical resources in a zone at a particular time of the year and can shuffle the total resources accordingly so that there is no shortage of resources.

IV. Lessons Learned

A. *Project 1*

After completing the 1st project, I learned the following lessons:

- I learned how data is stored in NoSQL databases in the form of collections that have documents that are in the form of JSON.
- I got to know that this format is used when there is no strict need for structure / schema in our databases.
- I learned how to traverse through a dataset that is given to us in the form of a collection from a NoSQL database.
- I learned different data manipulation techniques to filter the data according to the criteria of the query from the provided collection. The criteria could be like name of a city, or categories to which a business may belong to etc.
- I learned how to write data to files in Python [3].
- I learned about the haversine formula [1] which is used to calculate the distance between two points in a sphere.
- I learned how the haversine formula could be used to calculate the distance between two points on the earth's surface whose latitude and longitude were provided to us.

B. *Project 2*

After completing the 2nd project, I learned the following lessons:

- I learned about how to setup and work with Apache Spark [4] and Scala [5] on my local machine.
- I learned the basic Scala syntax like variable declaration and conditional statements etc.
- I learned how to work with spark dataframes.
- I learned how to do join of two dataframes using Spark SQL.
- I got to know how to read data from csv files in Scala.
- I learned how to calculate the hot zones from given rectangular and point datasets.
- I learned about the getis-ord statistic and how it could be used a z-score to find out the hotness of cells based on space and time.

References

[1] Calculate distance, bearing and more between Latitude/Longitude points for haversine formula. Available at: http://www.movable-type.co.uk/scripts/latlong.html

[2] ACM SIGSPATIAL Cup 2016 for getis-ord statistic. Available at: http://sigspatial2016.sigspatial.org/giscup2016/problem

[3] Python 2 documentation. Available at: https://docs.python.org/2/

[4] Apache Spark documentation: Available at: https://spark.apache.org/docs/latest/

[5] Scala documentation: Available at: https://docs.scala-lang.org/