CS685: Data Mining

Assignment 1 (150 marks)

Due on: 25th September, 2020, 6:00pm

Explore the website `https://www.covid19india.org/`. The APIs needed to access the data are available at `https://api.covid19india.org/`. The entire data is available from `https://api.covid19india.org/v4/data-all.json`.

The district data of India is available as a JSON file from `neighbor-districts.json`. It is in an adjacency list format. The file is available at the resources tab of the course webpage as well as the resources section of the `hello.iitk.ac.in` portal.

Submit all the necessary components of your data as a single `zip` file named `rollno-assign1.zip` in the `hello.iitk.ac.in` portal within the deadline.

If you do not follow the naming conventions, marks for that question will be automatically 0 (zero).

While the programs/scripts should be named `.sh` files, you can invoke any program from within the shell file. The programs should run in the Linux operating system.

All the output files should be sorted by the first field.

1. (20 marks) Modify the neighbor districts data according to the districts found from the Covid-19 portal. If a district in the portal is a combination of multiple districts in the file, create a new district with that name and assign a random id (Q...) that is unique. The node identifiers or names in this graph should match the Covid-19 district names. Delete the smaller districts. A neighbor of the larger district is a combination of all the neighbors of its components.

   Output the new data as `neighbor-districts-modified.json`. Arrange all the districts in alphabetical order, and give them ids starting from 101 (yes, 101 and not 1).

2. (20 marks) For every district i, find the number of cases from the Covid-19 portal. Take the time-period of analysis from 15th March, 2020 to 5th September, 2020.

   Output the total number of cases per week for every district in the following manner: $districtid$, $timeid$, $cases$, where $timeid$ is the id of the time (week/month/overall) starting from 1. Call this output file `cases-time.csv` and the script/program to generate this `case-generator.sh` where `time` is `week`, `month`, and `overall`.

3. (10 marks) Construct an undirected graph of districts out of this new file. In the graph, every district is a node. A district node is connected by an edge to every adjacent district of it, and vice versa.

   Output the graph in an edge list format. If district i has an edge with district j, output i, j. Call this output file `edge-graph.csv` and the script/program to generate this `edge-generator.sh`.

4. (10 marks) For every district, find the average and standard deviation of the number of cases of all its neighboring districts. Denote these by μ_i^N and σ_i^N.

   1. Find these for every week. A week is from Sunday to Saturday.
   2. Find these for every month.

3. Find these overall.

The output format is: $districtid, timeid, neighbormean, neighborstdev$ where the mean and standard deviation values are *rounded* to 2 places of decimal digits. Call this output file `neighbor-time.csv` and the script/program to generate this `neighbor-generator.sh` where `time` is `week`, `month`, and `overall`.

5. (15 marks) For every district, find the average and standard deviation of the number of cases of all the other districts in the state. Denote these by μ_i^S and σ_i^S. Note that this requires state information which *you* need to find out.

   Repeat finding these for every week/month/overall as in the earlier question. If a district is the only one in its state, consider its standard deviation to be 0.

   The output format is: $districtid, timeid, statemean, statestdev$ where the mean and standard deviation values are *rounded* to 2 places of decimal digits. Call this output file `state-time.csv` and the script/program to generate this `state-generator.sh` where `time` is `week`, `month`, and `overall`.

6. (15 marks) Using the above data, for every district, find its z-score for every week, month, and overall, using separately the neighborhood and the state information.

   The output format is: $districtid, timeid, neighborhoodzscore, statezscore$ where the z-score values are *rounded* to 2 places of decimal digits. Call this output file `zscore-time.csv` and the script/program to generate this `zscore-generator.sh` where `time` is `week`, `month`, and `overall`.

7. (15 marks) Find the hotspot and coldspot districts per week, per month, and overall. A district is hotspot (respectively, coldspot) if x_i is greater than (respectively, less than) the mean plus (respectively, minus) the standard deviation of its compared set.

   The output format is: $timeid, method, spot, districtid$ where $method$ indicates the comparison method used (neighborhood/state) and *spot* indicates hot/cold. Each hotspot/coldspot district must be in a separate row. Call this output file `method-spot-time.csv` and the script/program to generate this `method-spot-generator.sh` where `time` is `week`, `month`, and `overall`.

8. (15 marks) For every month and overall, find the top-5 hotspot and top-5 coldspot districts using the z-score values according to both the neighborhood and state methods.

   The output format is: $timeid, method, spot, districtid1, \ldots, districtid5$ where $method$ indicates the comparison method used (neighborhood/state) and *spot* indicates hot/cold. Call this output file `top-time.csv` and the script/program to generate this `top-generator.sh` where `time` is `week`, `month`, and `overall`.

9. (10 marks) Write a manual that describes how to use your code. Include all the programs, their plugins, and dependencie needed to run the program. Include a top-level script `assign1.sh` that runs the entire assignment.

   Call this manual `README.txt`.

10. (20 marks) Include a small report in LaTeX where you analyze the results, draw overall conclusions, etc. In the analysis, please do not keep repeating the numbers or the derived values. The analysis should be brief and should be easily explainable to the general audience (not mathematically oriented computer scientists).

    Call this report `report.tex`. Include all the necessary accessory files needed for its successful compilation.