DATA MINING - CS685A

ASSIGNMENT 1

Report for Assignment 1

Student Name: Boppana Tej Kiran

December 28, 2020

The assignment 1 for data mining course was solved and the programs needed to produce the outputs as asked for each question in assignment were written in python3. The python programs, script files needed to execute the programs all kept into the folder 20111070-assign1 and is submitted as zip file.

In this report we will discuss about the pre-processing done on data set followed by the processiong done on data as asked in assignment and finally discuss the effect of lock-down in controlling the spread of Corona virus in India using the results obtained.

Preprocessing done on the INPUT data:

The input files used for solving are :
1) data-all.json : This json file contains district wise COVID 19 cases information and is downloaded from https://api.covid19india.org/ website

2) neighbor-dictricts.json: This file is provided with the assignment and it contains information of neighbor districts for districts in India

The naming format of districts present in neighbor-district.json is different from the naming format of districts present in data-all.json file. Both are not matching.

Adjusting the names in neighbor-districts.json according to naming format followed in COVID portal data:

Multiple dissimilarities are observed between districts present in neighbor-districts.json and data-all.json. Those dissimilarities observed and how they were rectified to match the district names of neighbor-districts.json according to their corresponding names in data-all.json is explained below.

Case:1

Format followed for name string in neighbor-districts.json:

The sub strings in name string of district name present in neighbor-districts.json and separated by " _ " symbol and all letters present in name string are lower case.

Format followed for name string in data-all.json: The sub strings in name string of district name present in neighbor-districts.json and separated by " " space and starting letter of each word is classified.

Example: kanpur_nagar (in neighbor-districts.json)
Kanpur Nagar (in data-all.json)

This issue was addressed in the following manner:
All the district names in neighbor districts.json following the naming format as mentioned above, " _ " was replaced by " ' 'space and each word is capitalized to

match with naming format of data-all.json

Case:2

The name of same district present in neighbor-districts.json and data-all.json are differing by few letters.

Example: Maldah (in neighbor-districts.json)
Malda (data-all.json)

This issue was handled by, sequence matcher function from the difflib package of python is used to compare names of districts which are similar. We have set the parameter to match two strings if they are 90% similar. In this way, the names of districts in neighbor-districts.json are modified to match their corresponding names in data-all.json

Case 3:

There are states (Telangana, Sikkim, Assam, Goa, Delhi, Manipur) for which district level data is not available in data-all.json file. But the districts present in these states are present in neighbor-districts.json file.

This issue was handled by, modifying the neighbor-districts data such that, all the districts present in these states were deleted and new districts with same name as the above state names were added. The neighbors districts for these states were added.

Case 4:

There are some districts whose name are completely changed to different names and neighbor-districts.json file has older names and data-all.json has newer names for these districts.

This issue was addressed by modifying the names in neighbor-districts.json data by hard coding into the program as per the names present in data-all.json file.

A new .json file was generated after adjusting our districts data as per above four cases (prepossessing) which was names as neighbor-districts-modified.json

Thus the preprocessing of district names in neighbor-districts.json was done such that they match the names of districts present in data-all.json. So, that we can use the data from data-all.json file downloaded from COVID 19 portal for assignment

Solving of assignment questions using the preprocessed data:

a.k.a Data Processing

The data from **data-all.json** file and **neighbor-districts-modified.json** are used to answer the questions asked in assignment from **date: 15-03-2020 to date: 05-09-2020**

A total of eight python3 programs are written to generate the output files as asked in each question. The inputs needed and outputs generated by each of these programs was shown in **README.txt file.**

Here we shall discuss the functionality of each program:

(1) questn1.py
Functionality of the program:
This program is **solution to question1** asked in the assignment. This program reads the **data-all.json** and **neighbor-districts.json** input files and updates the neighbor districts information as per districts information available in **data-all.json**. It outputs the modified neighboring districts information to **neighbor-districts-modified.json** file.

It also outputs the districts present in each state as a python dictionary which is stored as a state-information.json file used by **questn5.py program**

Script file which executes questn1.py is: **neighbor-districts-modified.json**

(2) questn2.py
Functionality of the program:
This program is the **solution to question 2** asked in the assignment. This program reads the **data-all.json** and **neighbor-districts-modified.json** as input files. It calculates the weekly monthly and overall cases for each district between and including the dates 15-03-2020 and 05-09-2020 and outputs the results to the files **cases-week.csv, cases-month.csv, cases-overall.csv** respectively.

It also outputs **day-wise-cases.json** file which contains information about date-wise number of confirmed cases for each district in neighbor-districts-modified.json file. It is an intermediate file generated to be **used by questn4.py, questn5.py, questn6.py, questn7.py programs to reduce their execution time**.

Script file which executes questn2.py is: **case-generator.sh**

(3) questn3.py
Functionality of the program:
This program is s**solution to question 3** asked in the assignment. This program reads the **neighbor-districts-modified.json** file as input and creates a undirected graph of districts where each district is connected to its neighbors by an edge and outputs the list of edges present in the graph as **edge-graph.csv**

Script file which executes questn3.py is: **edge-generator.sh**

(4) questn4.py
Functionality of the program:
This program is **solution to question 4** asked in the assignment. This program reads the input from the **neighbor-districts-modified.json** and
day-wise-cases.json files. It calculates the mean and standard deviation of the neighbors of each district for every week, month and overall. It stores the calculated mean and standard deviation values for every week, month, overall into

neighbor-week.csv, neighbor-month.csv, neighbor-overall.csv files.

This program also outputs **neighmeth_weekly_cases.json, neighmeth_monthly_cases.json neighmeth_overall_cases.json** as intermediate files. These files contain the number of cases of neighbors of each district stored as list for every week, month, overall respectively. The data in above files is **used by questn6.py, questn7.py programs for quicker execution.**

Script file which executes questn4.py is: **neighbor-generator.sh**

(5) questn5.py
Functionality of the program:
This program is **solution to question 5** asked in the assignment. This program reads the **day-wise-cases.json, neighbor-districts-modified.json, state-information.json files** as input files. For every district, it calculates the mean and standard deviation of confirmed cases of all other districts present in its state for every week, month and overall and outputs the data as **state-week.csv, state-month.csv and state-overall.csv files**

This program also generates the **statemeth_weekly_cases.json, statemeth_monthly_cases.json, statemeth_overall_cases.json** files as intermediate .json files. They contain, for every district, weekly, monthly and overall number of confirmed cases of all other districts in its state. This data is used by **questn6.py, questn7.py programs for quicker execution.**

Script file which executes questn5.py is: **state-generator.sh**

(6) questn6.py
Functionality of the program:
This program is **solution to question 6** asked in the assignment. This program reads the following files as input:
1) neighbor-districts-modified.json
2) day-wise-cases.json
3) neighmeth_weekly_cases.json
4) neighmeth_monthly_cases.json
5) neighmeth_overall_cases.json
6) statemeth_weekly_cases.json
7) statemeth_monthly_cases.json
8) statemeth_overall_cases.json
It calculates the z scores for each district and outputs the neighborhoodzscore and statezscore for every week, month, overall to **zscore-week.csv, zscre-month.csv and zscore-overall.csv files**

Script file which executes questn6.py is: **zscore-generator.sh**

(7) questn7.py
Functionality of the program:
This program is **solution to question 7** asked in the assignment. This program reads the following files as input:

1) neighbor-districts-modified.json
2) day-wise-cases.json
3) neighmeth_weekly_cases.json
4) neighmeth_monthly_cases.json
5) neighmeth_overall_cases.json
6) statemeth_weekly_cases.json
7) statemeth_monthly_cases.json
8) statemeth_overall_cases.json
For every district it decides if it is a hot spot or cold spot based on the mean and standard deviation values of other districts calculated in two methods (neighborhood/state). Every district is decided if it is a hot spot or cold spot for every week, month and over all. The program outputs the above hot spot and cold spot data to **method-spot-week.csv, method-spot-month.csv, method-spot-overall.csv files**

Script file which executes questn7.py is: **method-spot-generator.sh**

(8) questn8.py
Functionality of the program:
This program is **solution to question 8** asked in the assignment. This program reads the following files as input:
1) zscore-week.csv
2) zscore-month.csv
3) zscore-overall.csv
4) neighbor-districts-modified.json
Based on the zscores calculated for every district in both methods (neighborhood/state methods) **by the questn6.py program**, It ranks the zscores and finds the top five districts as hot and cold spots corresponding to each neighborhood/state methods for every week, month and overall. It outputs the top five hot spots and cold-spots as output into **top-week.csv, top-month.csv and top-overall.csv files.**

Script file which executes questn8.py is: **top-generator.sh**

Data Summarization:

After generating the outputs as per the assignment, the results analysed to summarize the output.

The goal of the analysis performed is to verify wheather the Lockdown implemented in India helped to reduce the growth of COVID cases.

In India, complete lockdown was implemented from 25-03-2020 and relaxing of lockdown started from 08-06-2020.

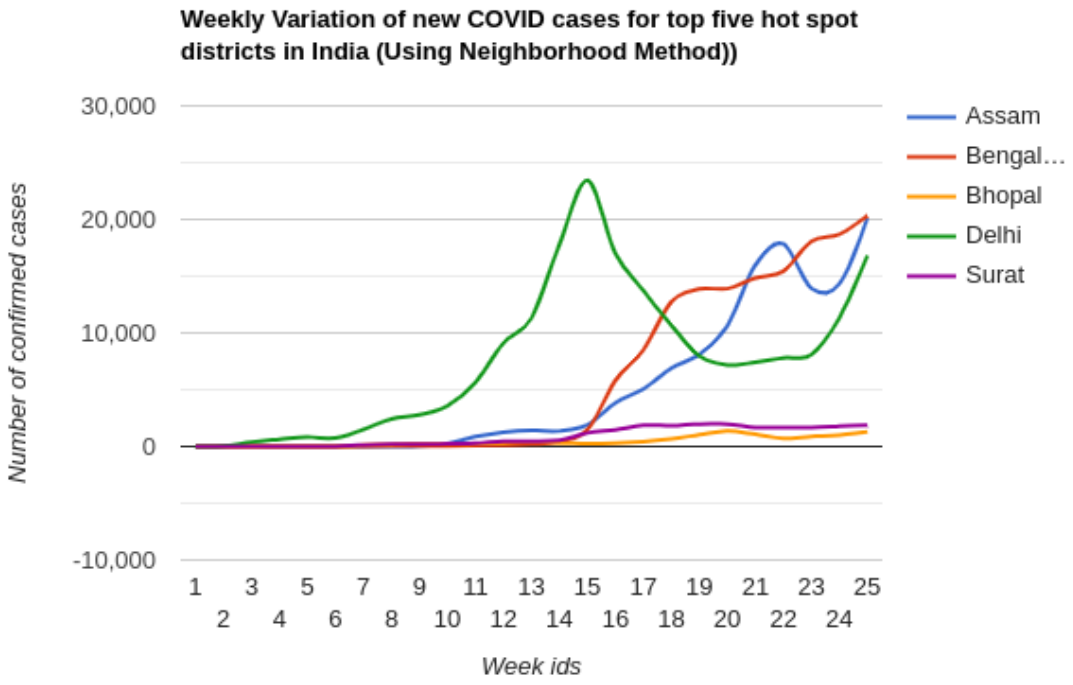
We have performed our analysis on COVID data from 15-03-2020 (i.e Week id-1) to 05-09-2020 (Week id-25).

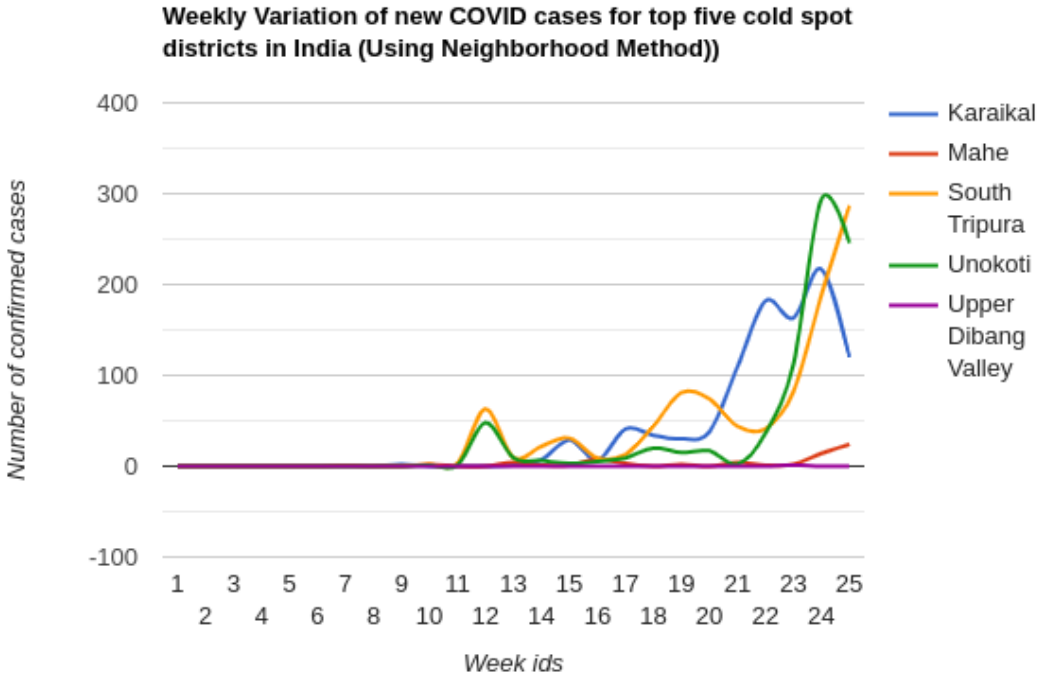
So, the locked down started from Week id -2 (25-03-2020) and was relaxed from

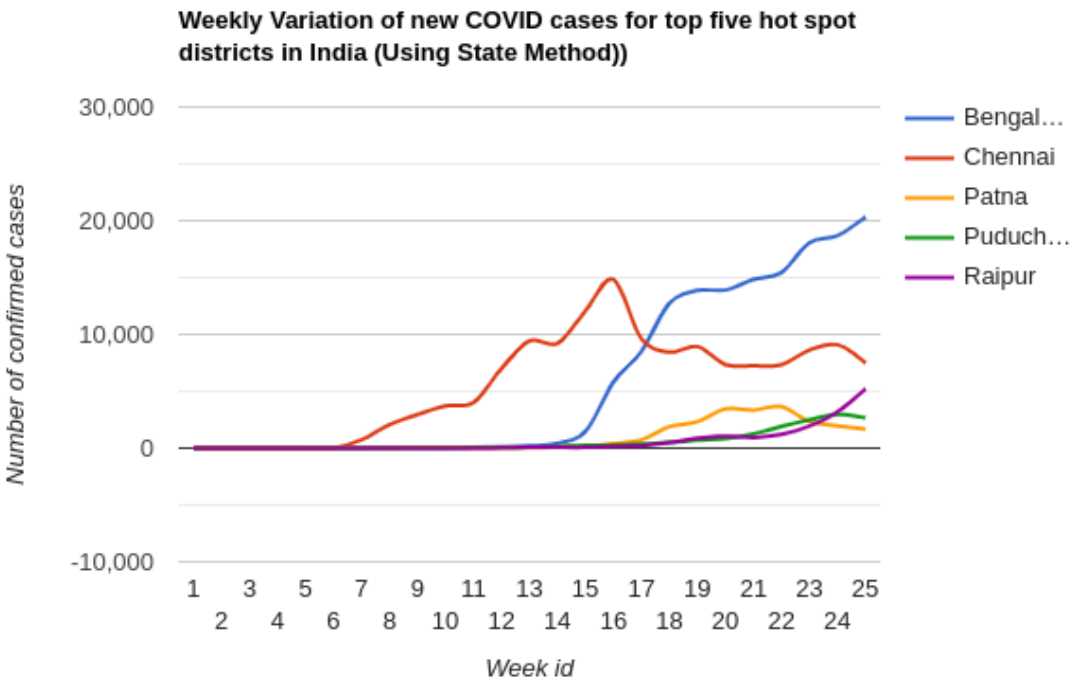
Week id – 13 (08-06-2020)

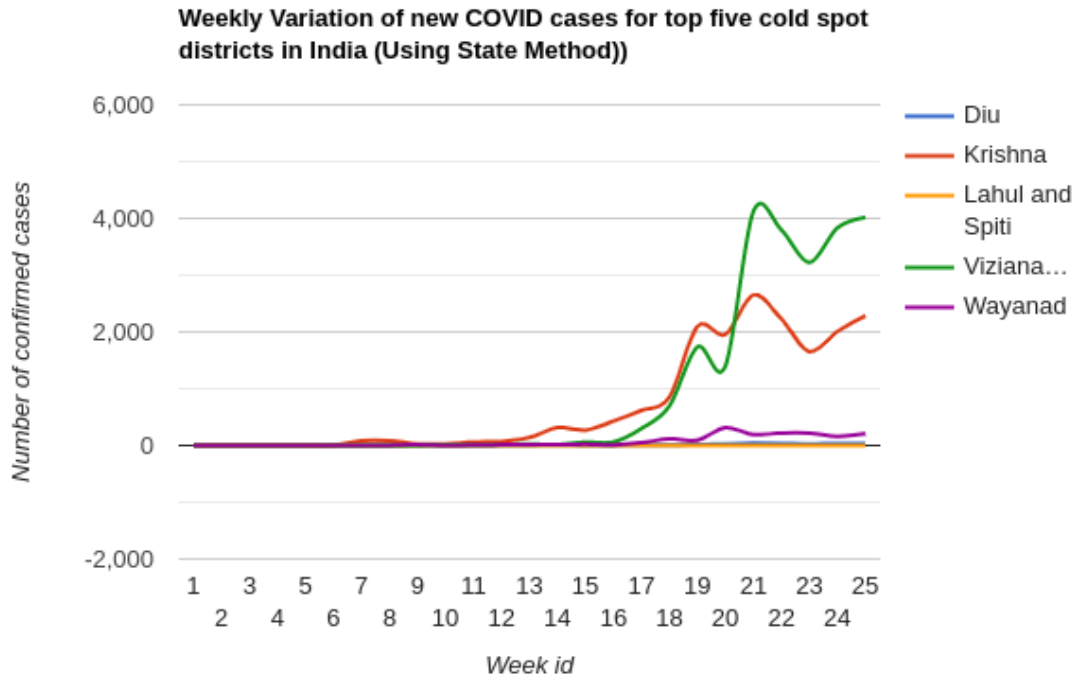
So, based on the above information, graphs are plotted to visialize the number of new cases confirmed for every week (from week id-1 to week-id-25).

The graphs were plotted for the overall top five hot spots and top five cold spot districts across the India obtained by using both neighborhood method and state method. These graphs are shown below:




Weekly Variation of new COVID cases for top five cold spot districts in India (Using Neighborhood Method))
Number of confirmed cases
400
300
200
100
0
-100
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25
Week ids
Karaikal
Mahe
South Tripura
Unokoti
Upper Dibang Valley


Weekly Variation of new COVID cases for top five hot spot districts in India (Using State Method))
Number of confirmed cases
30,000
20,000
10,000
0
-10,000
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25
Week id
Bengal…
Chennai
Patna
Puduch…
Raipur



Conclusion: From the above graphs, we can observe that for both hot spot and cold spot districts calculated by using neighborhood method and state methods we can observe that the rate of increase of new COVID cases for every week is very low till Week id 13 (The week in which the lock down was relaxed). After Week 13 (The week in which the lock down was relaxed), theredata summarization is high increment in growth rate of new COVID cases for most of the top hot spots and cold spots evaluated by both the methods (neighborhood/state). Thus our analysis showed that implemeting lockdown across helped reduce the growth of number of new COVID cases