DBSCAN Report

2014021358

노민수

1. Test Envioronment

Language : java

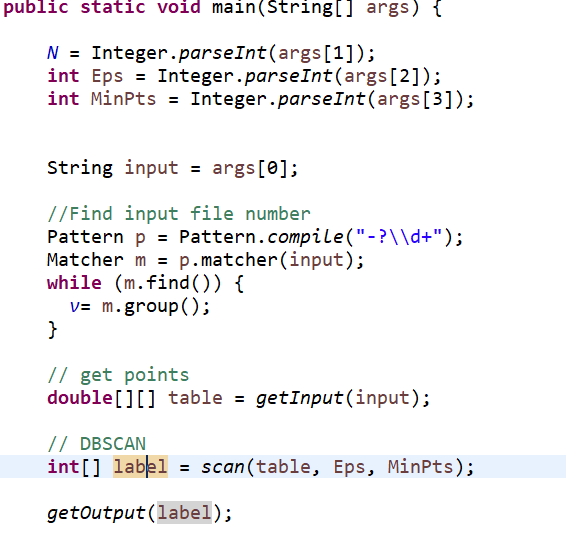
2. How to Execute

Run Configuration -> Argument : input1.txt 8(Number ClusterAllowed) 5(Eps) 22(MinPts)

3. How to Implement

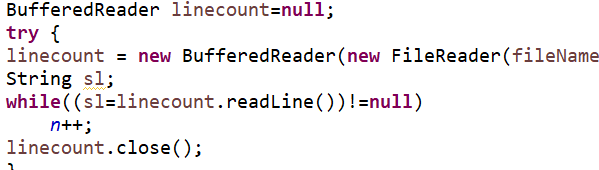
1) main

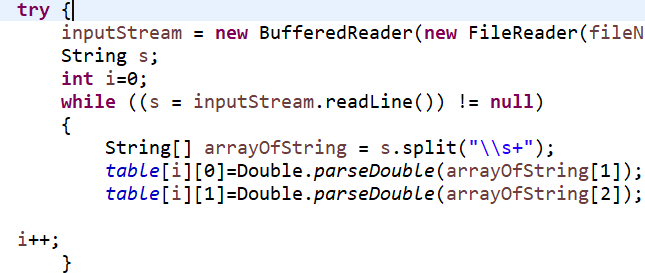
Get the arguments, especially input file’s number by using pattern and matcher, go getInput(), dbscan that input then getOutput().



2) getInput

Count all the points by proceeding readLine till it finds EOF



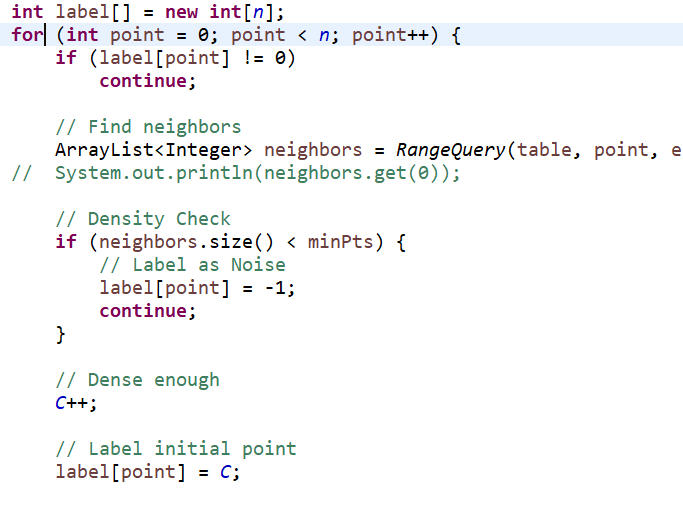
Row of Table is a point’s index, two column represent x,y coordinates, parse each line into two double, till it reaches EOF

3) scan

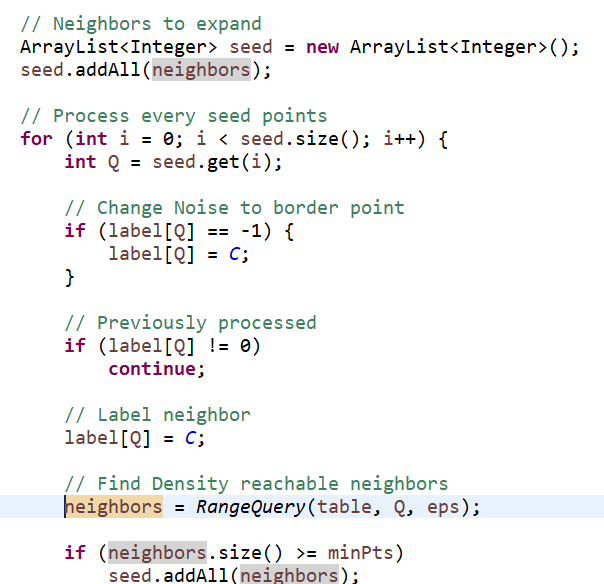
If point is already visited, continue to other point.

Find current point’s neighbors and do density check

If it’s not dense, mark as -1 (noise). Otherwise start clustering and assign point to that cluster

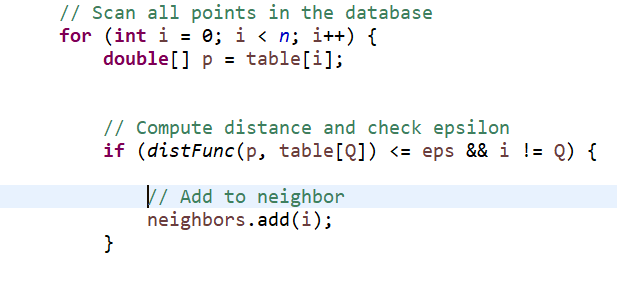


With all neighbors of the current point, do RangeQuery to find Density-reachable points unless its not a noise or previously visited.



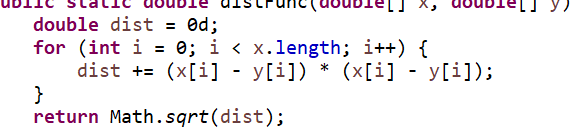
4) RangeQuery

For each row of table(index of points), measure distance(reachable) to decide whether It is neighbor of Q(current point) or not.



5) disfFunc

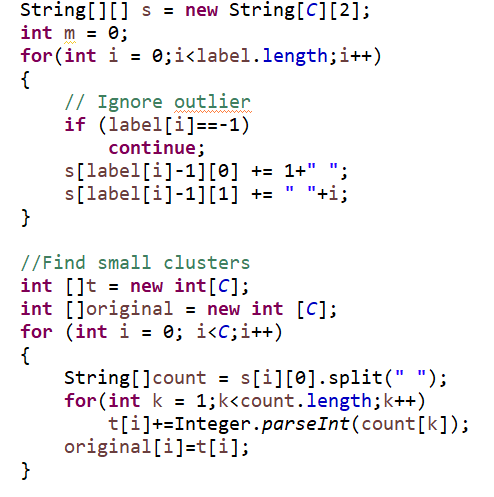
Get distance of two points by calculating Euclidean distance;sqrt(dx^2 + dy^2)



6) getOutput

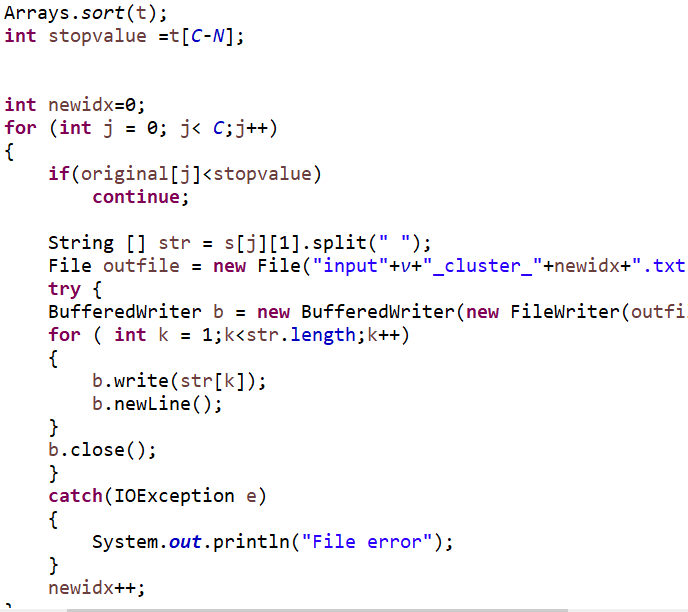
Concatenate the idx of points in same cluster in s[row][1]

To handle N, concatenate all the incident(’1’) that hit the each cluster. Count the number of each cluster hits in t[i] and copy original order of cluster since t[i]’s going to be sorted later.



Sort t[i] to find the hits of each cluster in ascending order.

Find the stop value and if the current cluster has smaller hits than stop value, skip making that cluster as file. Put new index to each cluster file so that the name of the cluster file would be written in ascending order



4. Result

Input1.txt : 98

Input2.txt : 94

Input3.txt : 99

