ITE4005 Data Science course

Assignment 3

2016026080 한다진

**Experiment environments**

OS Window 10

Language python3

Tools Jupyter notebook

Pure python script

**DBSCAN**

DBSCAN is clustering method using density of dataset. There are several keywords like Eps, MinPts, core point, boder point and noise point.

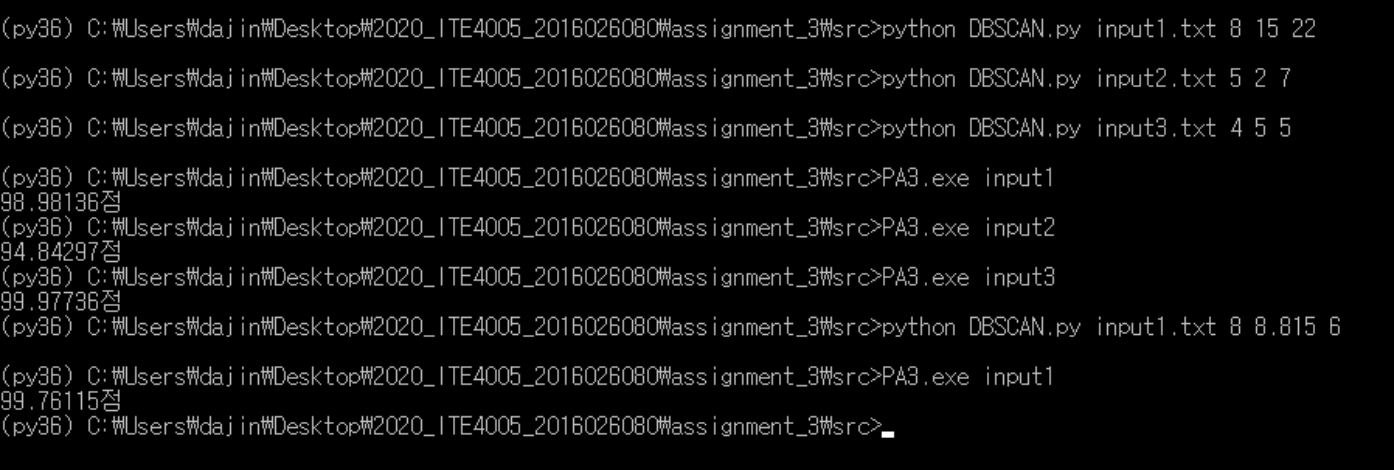
DBSCAN clusters specific data by considering the density of dataset. For every data in dataset, DBSCAN find the adjacent datas within a certain distance. If the number of adjacent datas is bigger than MinPts, set current data to core point and form a cluster of nearby datas around the current data. If there are another core points in adjacent datas, merge those clusters. In each clusters, points are divided to core point and non-core point, which means border point. And the rest of them are called noise point, which means the outlier.

I make a DBSCAN class using union-find algorithm.

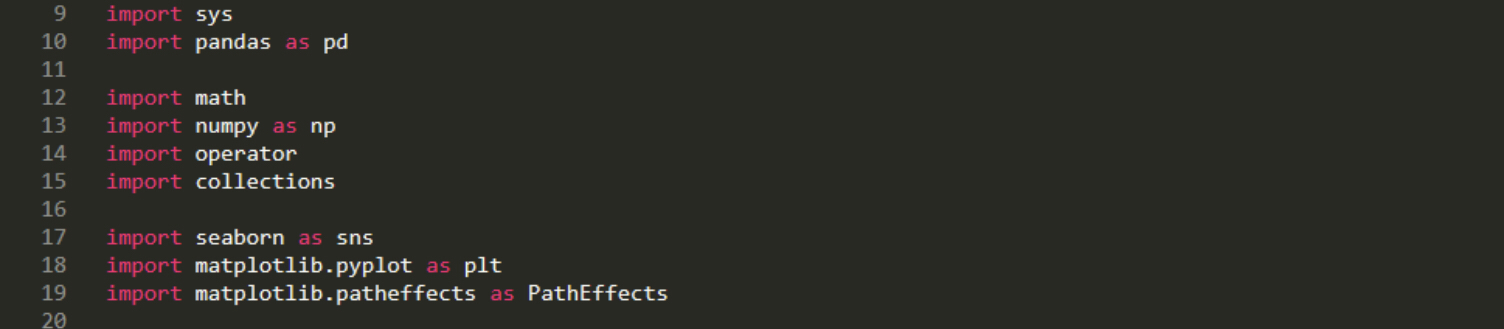
1. First, sort the dataset by x coordinates.
2. And then make a list of adjacent points. Adjacent points of point in condition of ‘id=24’ is adjs[24] = [first adjacent point, second adjacent point, … , last adjacent point]. I considered the spots of which diferrences of x coordinates is smaller than Eps, which means the radius of cluster.
3. Last, merge the clusters considering the adjacent datas. Using root[] list and recursive get\_root() function, I get the label of cluster and merge them.

**Quick Start**

* + Run DBSCAN.py

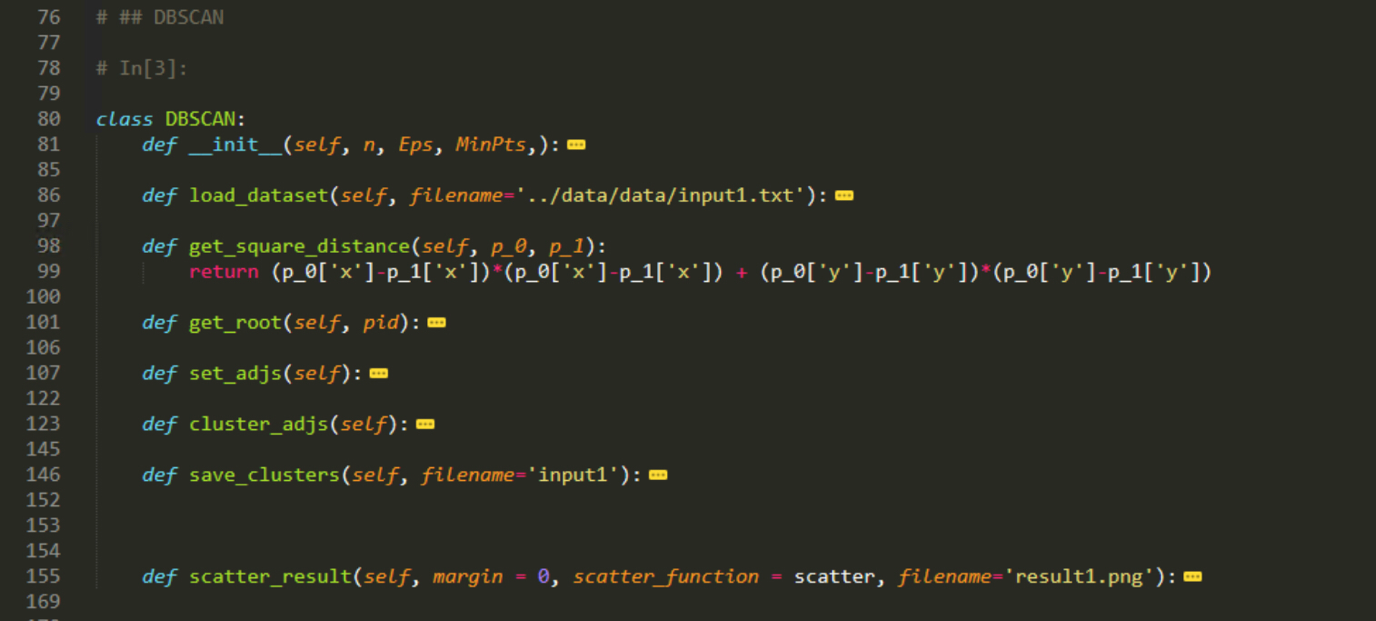


**Code explanation**

* + ****Import Libraries

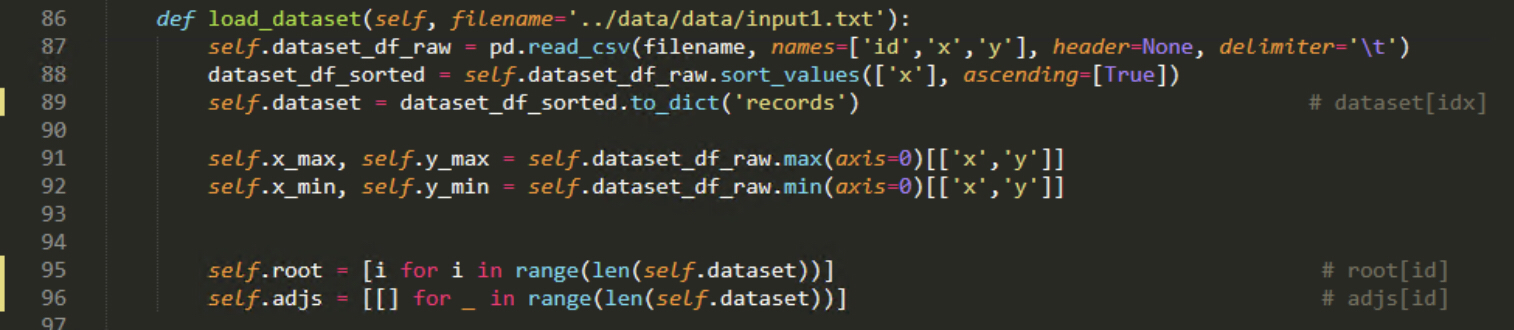
Import *pandas* libraries to use DataFrame.

Seaborn, matplot libraries are used only in .ipynb file to scatter the dateset.

* + **DBSCAN Class**

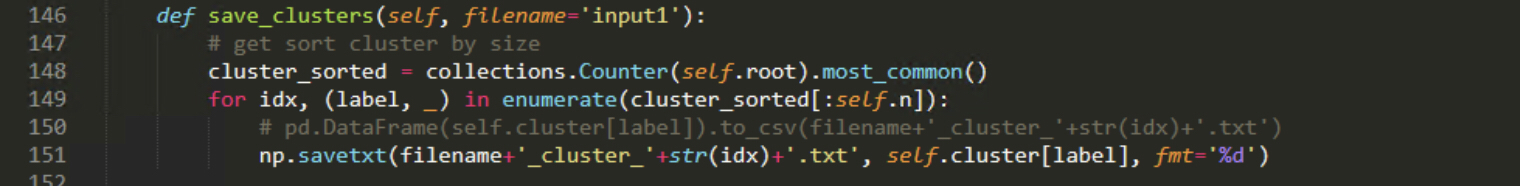
Make DecisionTree class and use the internal function in decision class. The structure of DecisionTree class is recursive. The internal function and variables work with recursive pattern, like subtree. Detail of each function is below.

* + Load input file



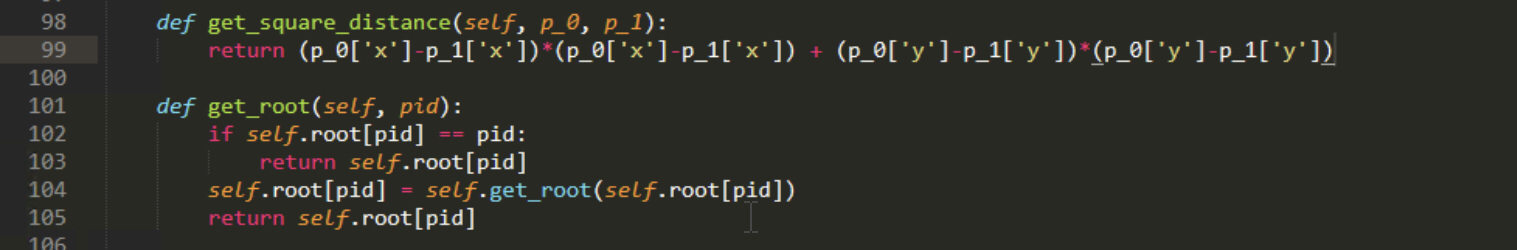
Load input dataset in DataFrame format and convert it to dictionary list. And set root list for union find, and adjs llist for saving adjacent points around center point.

* + Save output and get score



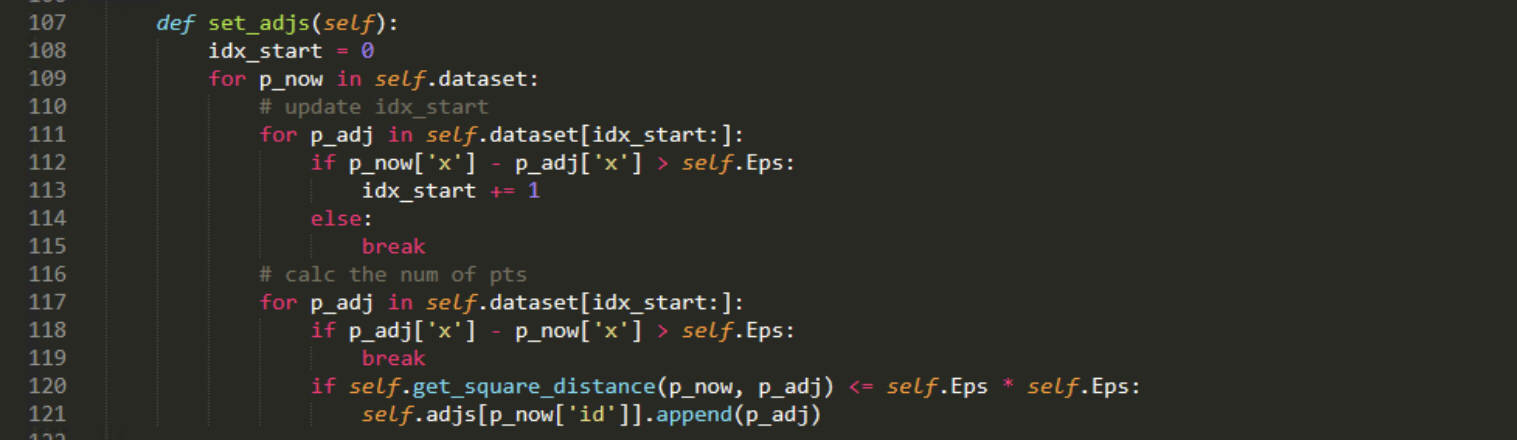
Save the largest n clusters.

* + Functions for union-find algorithm



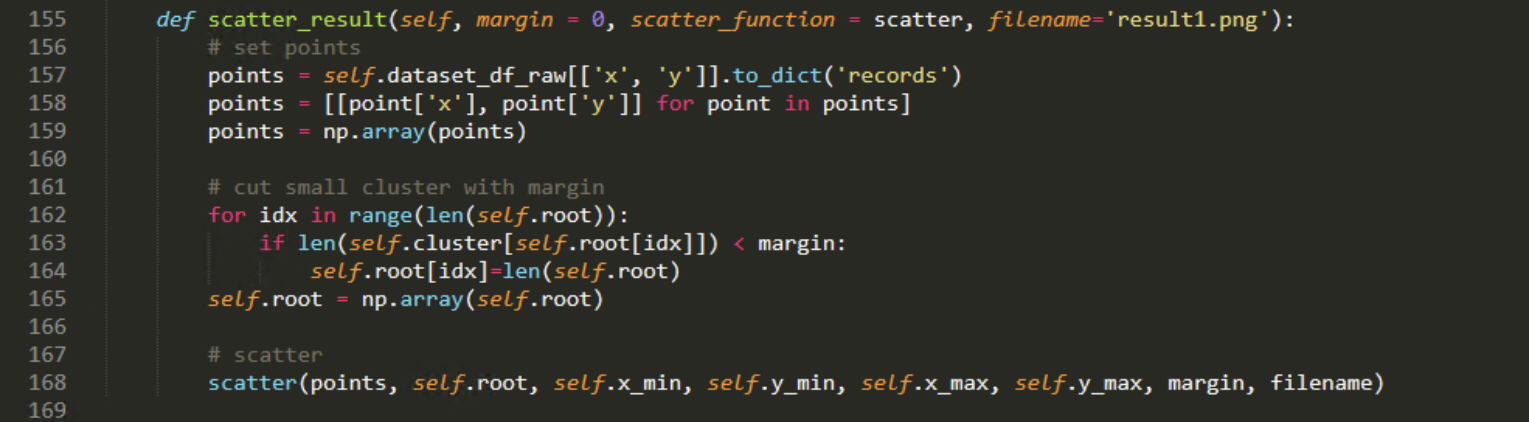
get\_square\_distance is used to check the adjacent spots. And get\_root function is used to get the cluster\_id of current point.

* + Get adjacent points and form clusters

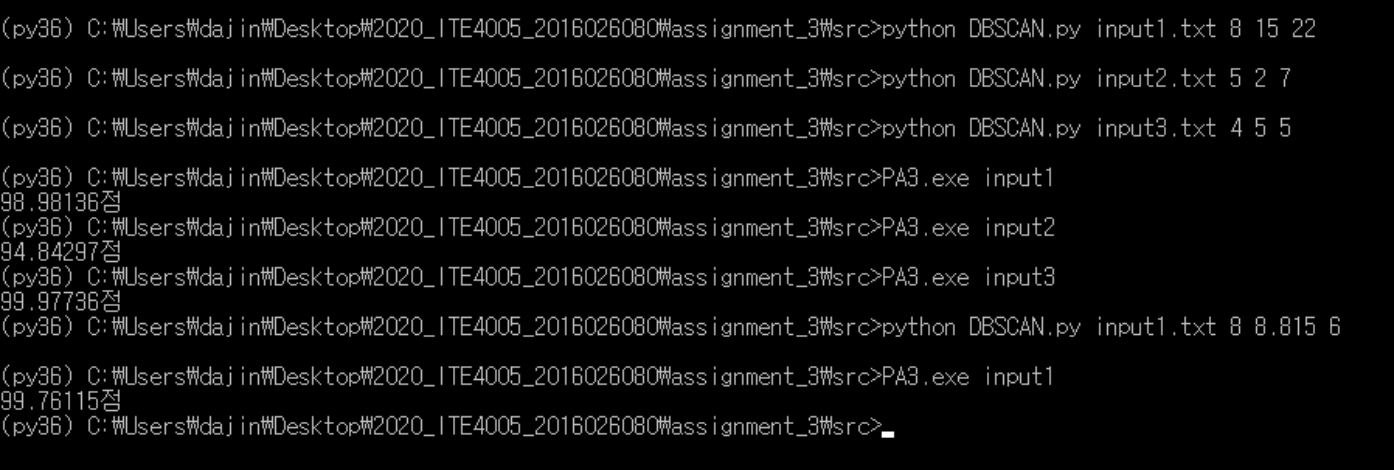
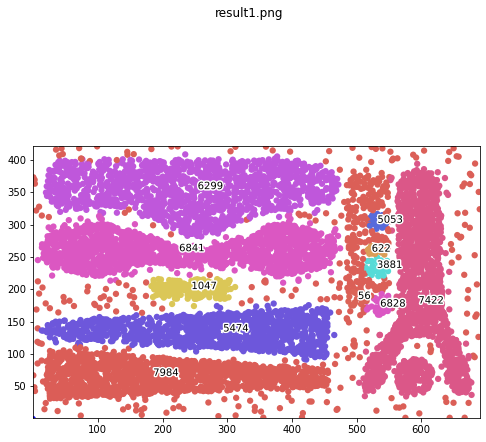
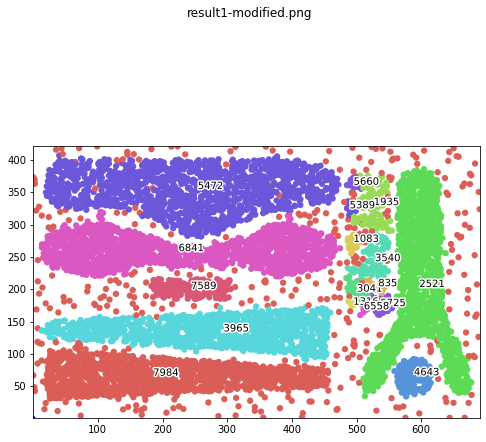
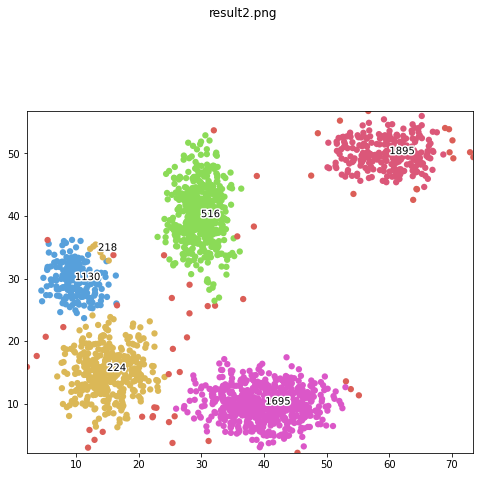
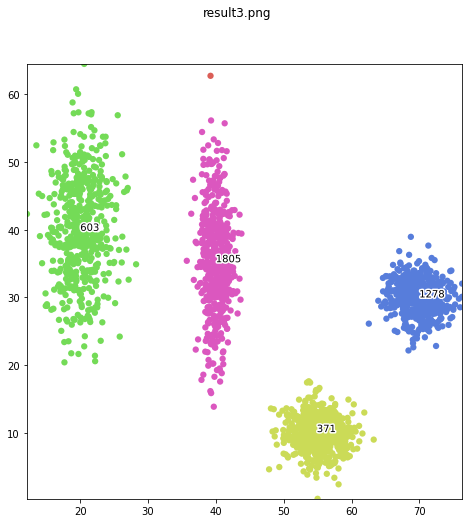




For every points, get all adjacent around current points in set\_adjs function. After that, cluster\_adjs form a clusters using the result of set\_adjs. In a sequence of spots ordered by x coordinates, merge core points in same adjacent point list.

* + **Scatter dataset**  
    I make a scatter function to check the result of DBSCAN clusterer. And save it to result.png.

**Experiment**



When I check the result of input1.txt, two clusters are built incorrectly. And checking the result.png, I change the parameter of input1.txt. and compare the score. Socre rises from 98.98 to 99.76.