# CENG 414
## Introduction to Data Mining
Fall 2019-2020
THE 3

Murat Ozturk
mozturk@ceng.metu.edu.tr
Due date: December 16, 2019, 23:59

## 1 Overview

In this THE, you are going to implement **Agglomerative Hiearchical Clustering** algorithm in Python language. For this purpose, you will be provided a template file and a driver file: In the template file, you are expected to implement the methods which have missing definitions(bodies). The driver file is provided to you to test your code. After you implement the methods in the template file, you will be able to test your implementation with various parameters and given dataset using the driver file. The pseudocode of the **Agglomerative Hiearchical Clustering** algorithm that you are expected to implement is as follows:

**Algorithm 1:** Agglomerative Hierachical Clustering Pseudocode

```
Data: 2-D data points
Result: Clusters of 2-D data points
Initialize each data point as a cluster
Calculate the initial proximity matrix between all data points
repeat
    Merge the two closest clusters considering the linkage
    Update the proximity matrix to reflect the proximity between the new cluster and the
     original clusters
until The desired number of clusters obtained;
```

## 2 Tasks

In this THE, sample driver file ("the3_main.py") implementation and Agglomerative Hierarchical Clustering template file ("the3_agglomerative_clustering.py") are given to you. If you check "the3_agglomerative_clustering.py" file, you will see sections like "# TODO: Implement here". Your job is to complete code sections that needs implementation.

### 2.1 Reading Input File (20 Pts.)

In this task, you are expected to implement the code that reads the dataset from a given file which will be used as input to **Agglomerative Hiearchical Clustering** algorithm. The dataset in

the given file consists of 2-D data points where every data point is represented as a (x,y) tuple which are x-coordinate and y-coordinate values, respectively. You are expected to implement the **"read_input_file()"** function inside "the3_agglomerative_clustering.py" file. The necessary information on what this function returns is provided in the template file. This function is used to read data and feed it to the Agglomerative Hiearchical Clustering model.

The format of the file will be as follows:

(point1_x, point1_y)
(point2_x, point2_y)
(point3_x, point3_y)
.
.
.

In the input file, the (x,y) coordinate values will be provided as floating point values with 2 decimal digits.

### 2.2 Initialize Clustering (5 Pts.)

In this task, you are expected to implement **initialize_clustering()** method in "the3_agglomerative_clustering.py". This method represents the first step of the algorithm given in Algorithm1:

1 Initialize each data point as a cluster

The necessary information on what this function returns is provided in the template file.

### 2.3 Assign data points to clusters (75 Pts.)

In this task, you are expected to implement the **"fit_predict()"** method inside "the3_agglomerative_clustering.py". This method takes desired number of clusters and linkage function as parameters and returns the resultant clusters. This method represents the main loop of the algorithm given in Algorithm1:

2 Calculate the initial proximity matrix between all data points
3 **repeat**
4 | Merge the 2 closest clusters considering the linkage
5 | Update the proximity matrix to reflect the proximity between the new cluster and the original clusters
6 **until** *The desired number of clusters obtained*;

#### 2.3.1 Calculating initial proximity matrix

In the above algorithm, step 2 defines the initial proximity matrix calculation. Initially, each data point will be assigned to an independent cluster as stated in step 1 of the algorithm. Step 2 of the algorithm states that the initial proximity matrix' values are the pairwise distances between data points. In this THE, Euclidean distance function should be used to calculate pairwise distances between data points. In order to clarify the issue, assume the following 2-D points are given as in the following table:

| Point | x Coordinate | y Coordinate |
|---|---|---|
| **p1** | 0.40 | 0.53 |
| **p2** | 0.22 | 0.38 |
| **p3** | 0.35 | 0.32 |
| **p4** | 0.86 | 0.19 |
| **p5** | 0.98 | 0.41 |
| **p6** | 0.45 | 0.23 |

Considering the given data points, the corresponding initial proximity matrix will be formed as follows:

| | **p1** | **p2** | **p3** | **p4** | **p5** | **p6** |
|---|---|---|---|---|---|---|
| **p1** | 0.00 | 0.23 | 0.22 | 0.57 | 0.59 | 0.30 |
| **p2** | 0.23 | 0.00 | 0.14 | 0.67 | 0.76 | 0.27 |
| **p3** | 0.22 | 0.14 | 0.00 | 0.53 | 0.64 | 0.13 |
| **p4** | 0.57 | 0.67 | 0.53 | 0.00 | 0.25 | 0.41 |
| **p5** | 0.59 | 0.76 | 0.64 | 0.25 | 0.00 | 0.56 |
| **p6** | 0.30 | 0.27 | 0.13 | 0.41 | 0.56 | 0.00 |

A sample calculation is given below which calculates the distance between $p1$ and $p2$:

$$\sqrt{(0.40-0.22)^2+(0.53-0.38)^2} = \sqrt{0.0324+0.0225} = \sqrt{0.0559} = 0.23 \tag{1}$$

When doing distance calculations, you are expected to round the result to 2 decimal digits.

### 2.3.2 Linkage functions

In the above algorithm, the linkage determines the methodology to merge the clusters. In this THE, **"fit_predict()"** may be invoked with the following 3 linkage functions: "MIN, MAX, GROUP AVERAGE". You are expected to implement all the linkage functions to get full points. The description of these linkage functions are given below.

**MIN LINKAGE:** In the MIN version (a.k.a single link) of hierarchical clustering, the proximity of two clusters is defined as the minimum of the distance between any two points in the two different clusters. Mathematically:

$$Proximity(C1, C2) = min(dist(P_i, P_j)) \text{ such that } P_i \in C_1, Pj \in C2 \tag{2}$$

**Example:** Assume the 2-D data points given in the previous section. Assume after 1 iteration of the algorithm, the clusters form as follows: [p1],[p2],[p3,p6],[p4],[p5]. Here, each list (shown as "[a,b,..]" where a,b are elements of the list) corresponds to an independent cluster and the elements inside each list is a member of that corresponding cluster. The proximity between clusters [p2] and [p3,p6] are given by:

$$\begin{aligned} &= min(dist(p2, p3), dist(p2, p6)) \\ &= min(0.14, 0.27) \\ &= 0.14 \end{aligned} \tag{3}$$

**MAX LINKAGE:** In the MAX version (a.k.a complete link) of hierarchical clustering, the proximity of two clusters is defined as the maximum of the distance between any two points in the two different clusters. Mathematically:

$$Proximity(C1, C2) = max(dist(P_i, P_j)) \text{ such that } P_i \in C_1, Pj \in C2 \tag{4}$$

**Example:** Assume the 2-D data points given in the previous section. Assume after 1 iteration of the algorithm, the clusters form as follows: [p1],[p2],[p3,p6],[p4],[p5]. Here, each list (shown as "[a,b,..]" where a,b are elements of the list) corresponds to an independent cluster and the elements inside each list is a member of that corresponding cluster. The proximity between clusters [p2] and [p3,p6] are given by:

$$\begin{aligned} &= max(dist(p2, p3), dist(p2, p6) \\ &= max(0.14, 0.27) \\ &= 0.27 \end{aligned} \tag{5}$$

**GROUP AVERAGE LINKAGE:** In the GROUP AVERAGE version of hierarchical clustering, the proximity of two clusters is defined as the average pairwise proximity among all pairs of points in the different clusters. Mathematically:

$$Proximity(C1, C2) = \frac{\sum dist(P_i, P_j)}{m_1 * m_2} \text{ such that } P_i \in C_1, Pj \in C2 \tag{6}$$

where $m_1, m_2$ denote the number of elements in clusters $C_1, C_2$, respectively.

**Example:** Assume the 2-D data points given in the previous section. Assume after 1 iteration of the algorithm, the clusters form as follows: [p1],[p2],[p3,p6],[p4],[p5]. Here, each list (shown as "[a,b,..]" where a,b are elements of the list) corresponds to an independent cluster and the elements inside each list is a member of that corresponding cluster. The proximity between clusters [p2] and [p3,p6] are given by:

$$\begin{aligned} &= \frac{dist(p2, p3) + dist(p2, p6)}{1 * 2} \\ &= \frac{0.14 + 0.27}{2} \\ &= 0.21 \end{aligned} \tag{7}$$

### 2.3.3 Merging the two closest clusters and updating the proximity matrix

This part of the algorithm represents the loop including the steps between 3-6. In order to merge the two closest clusters, the smallest pairwise proximity value is found in the proximity matrix: The corresponding clusters are then merged. As an example, assume the clusters after 1 iteration of the algorithm are formed as follows: [p1],[p2],[p3,p6],[p4],[p5]. Assume MIN is used as the linkage function. After calculating all the pairwise cluster distances, the resultant proximity matrix will be formed as follows:

| | p1 | p2 | p3,p6 | p4 | p5 |
|---|---|---|---|---|---|
| **p1** | 0.00 | 0.23 | 0.22 | 0.57 | 0.59 |
| **p2** | 0.23 | 0.00 | 0.14 | 0.67 | 0.76 |
| **p3,p6** | 0.22 | 0.14 | 0.00 | 0.41 | 0.56 |
| **p4** | 0.57 | 0.67 | 0.41 | 0.00 | 0.25 |
| **p5** | 0.59 | 0.76 | 0.56 | 0.25 | 0.00 |

The smallest value in the above matrix is 0.14 which is the proximity between [p2] and [p3,p6] clusters. These two clusters are merged. After merging these 2 clusters, the proximity matrix is updated as follows:

| | p1 | p2,p3,p6 | p4 | p5 |
|---|---|---|---|---|
| **p1** | 0.00 | 0.22 | 0.57 | 0.59 |
| **p2,p3,p6** | 0.22 | 0.00 | 0.41 | 0.56 |
| **p4** | 0.57 | 0.41 | 0.00 | 0.25 |
| **p5** | 0.59 | 0.56 | 0.25 | 0.00 |

The algorithm should stop whenever the desired number of clusters are obtained. The desired number of clusters will be provided as input to **"fit_predict()"** function. As an example, assume the desired number of clusters=2 is provided as input. Then, the algorithm should eventually stop when the number of clusters is 2. Continuing with the above example, the algorithm will eventually stop and will return the following clusters: [p1,p2,p3,p6], [p4,p5]. The necessary information on what this method returns is provided in the template file.

# 3 Rules, Suggestions and Tips

- Make sure you keep the template file ("the3_agglomerative_clustering.py") structure that is given to you. You can add method(s)/function(s)/member variable(s)/lines of code to the template file, but can not remove any function/method/member variable. Also, you MUST have following lines defined in your code:
  **self.data=input_data**
  **self.clusters_current=**[ ]

- You are expected to implement the desired method(s)/function(s) bodies after "# TODO: Implement here" comment line. You are expected to write return statements inside function(s)/method(s) considering their usages in the driver file ("the3_main.py").

- You are NOT allowed to import any libraries other than the given libraries in the template file. If this rule is violated, your submission will not be evaluated and you will get 0 points for this THE.

- Sample input file will be provided to you. The output file considering the code in the driver file will also be provided. The function(s)/method(s) that you are going to implement inside the template file should stick with their usage inside the driver file.

- After you complete your implementation, check sure your outputs are the same as the outputs inside the sample output file to get credits.

- You can test your code with different parameters and input file(s) by modifying the driver file. You will not submit the driver file and it will not be used for evaluation.

- The sample input and output files are provided you as "sample_input.txt" and "sample_output.txt"
- In order to run your code, use the following syntax:
  python the3_main.py < input_file_name >
  **Ex:** python the3_main.py sample_input.txt

# 4 Submission and Regulations

1. Do not put your binary files,text files, IDE related files etc into your submission. Zip pnly "the3_agglomerative_clustering.py" file which includes your implementation, rename the zip file as <ID> _ <FullNameSurname> and submit it through odtuclass. For example: e1234567_MuratOzturk.zip
2. Copying from others is strictly forbidden and is subject to discplinary action.
3. Black box method will be used for evaluation.
4. Late submissions will not be accepted.