COMP 2010 Computing III (Fall 2016) Instructor: Anna Rumshisky

Project #1 (Part 2 – EXTRA CREDIT). Agglomerative Clustering with Group-Average Distance.

Handed out: 11/22/16	Due date: 12/14/16 (one day after the final)

Submission instructions

Inside that the homework subdirectory, create a directory for project #1, part 2, and call it `project1_2`. When you finish the assignment, go to the homework directory and submit it as follows:

```
$ submit arum project1_2 project1_2
```

Problem

In the Project #1 (Part 1), you implemented an agglomerative clustering algorithm for real-valued vectors in multidimensional space. In this project, your goal is to implement a different version of the same algorithm that uses a different method to compute distance between clusters.

In Part 1, we used Euclidean distance to compute the distance between the objects being clustered, and the distance between cluster centroids to compute the distance between clusters.

For this project, the distance computation between objects remains the same, but the distance between clusters will be computed as the average of all pairwise distances between the elements of the two clusters.

For example, if you are computing the distance between two clusters A = $<a_1, a_2>$ and

B = $<b_1, b_2, b_3>$, the distance between A and B should be computed as follows:

$$d(A, B) = (d(a_1,b_1) + d(a_1,b_2) + d(a_1,b_3) + d(a_2,b_1) + d(a_2,b_2) + d(a_2,b_3)) / 6$$

Requirements:

Your algorithm needs to be efficient and not recompute the distances between the clusters for which you have computed the distance previously. You will be evaluated on efficiency of your solution.

Specification

The specification is similar to the specification for the agglomerative clustering homework, but please note the formatting changes in the example below.

Additionally, you need to print out the history of merges that your set of objects goes through during clustering, i.e. which clusters are merged at every step.

Your program should assume the following usage:

```
$ cat input.txt | project2 [number-of-clusters] > output.txt
```

Example

For example, given the following invocation and input, your program should produce the following output:

```
$ cat input.txt | project2 2 > output.txt
```

Input file `input.txt`:

```
10.1 23.1 4.9 56.84
1.2 23.4 4.2 5.35
2.5 67.4 5.6 34.9
11.32 2.1 4.12 15.3
```

Output file `output.txt`:

```
Merging: cluster: [[11.32 2.1 4.12 15.3]]
         cluster: [[1.2 23.4 4.2 5.35]]
Merging: cluster: [[1.2 23.4 4.2 5.35], [11.32 2.1 4.12 15.3]]
         cluster: [[10.1 23.1 4.9 56.84]]

Resulting clusters:
cluster: [[2.5 67.4 5.6 34.9]]
cluster: [[10.1 23.1 4.9 56.84], [1.2 23.4 4.2 5.35], [11.32 2.1 4.12 15.3]]
```