CS57300: Assignment 5

Due date: Wednesday December 2, 11:59pm (submit via BrightSpace)

Clustering

In this programming assignment you will implement the K-means algorithm and compare it to agglomerative clustering on the MNIST digit dataset. The MNIST data consists of 20,000 examples of 28×28 images of digits (i.e., numbers from 0-9). There are two files for this assignment.
`digits-raw.csv` contains the pixel information for each image (first column: image id, second column: class label, remaining 784 columns: pixel features).
`digits-embedding.csv` contains a precomputed 2-dimensional embedding of each image using t-Distributed Stochastic Neighbor Embedding (tSNE) (first column: image id, second column: class label, third and fourth columns: image embedding features).

You should implement your solution using Python. You can use supporting libraries like numpy, scipy as before, but DO NOT use any publicly available code including but not limited to libraries such as **sklearn**. As before, you should submit your typed assignment report as a pdf along with your source code file.

In the following sections, we specify a number of steps you are asked to complete for this assignment. **Note that all results in sample outputs are fictitious and for representation only.**

Note: Include the line `np.random.seed(0)` at the beginning of your code so that it is easier to compare your answers.

For all the experiments, the definition of Silhoutte Coefficient (SC) is as follows:
For an individual point i:

- A = average distance of i to points in same cluster
- B = average distance of i to points in other clusters
- S_i = (B-A) / max(A,B)

SC of clustering = average of S_i values for all points i in the dataset.

And, the definition of NMI to be used is: $NMI(C, G) = \frac{I(C,G)}{H(C)+H(G)}$

For all metrics that require distance calculation, you should use the Euclidean distance.

1 Exploration (5 pts)

Please put your code for this question in a file called **exploration.py**, and plots in the report.

1. Randomly pick one digit from each class in `digits-raw.csv` and visualize its image as a 28×28 grayscale matrix.
2. Visualize 1000 randomly selected examples in 2d from the file `digits-embedding.csv`, coloring the points to show their corresponding class labels. Select the examples randomly using the command `np.random.randint(0, N, size=1000)`, where N is the total number of examples.

2 K-means Clustering

Consider the following setup for questions on K-means Clustering.

- Data/features: Use the `digits-embedding.csv` data, with the two continuous embedding features.
- Distance measure: Euclidean distance.
- Starting cluster centers: If there are N points in the dataset, then randomly select K points, as the starting cluster centroids, using the command `np.random.randint(0, N, size=K)`.
- Stopping criteria: Set maximum number of iterations to 50 (In the last iteration, you break out of the loop after the assignment of points to the cluster and before recomputing the new centroids). You can also stop when the centroids are not updated.
- Evaluation: You will evaluate the results objectively with (1) within-cluster sum of squared distances, (2) silhouette coefficient, and (3) normalized mutual information gain (based on image labels).

2.1 Code (10 pts)

Please put your code for this question in a file called **kmeans.py**, and include results in the report. Your python script should take two arguments as input.

1. *dataFilename*: corresponds to a subset of the embedding data (in the same format as `digits-embedding.csv`) that should be used as the input data to your algorithm.
2. *K*: an integer to specify the number of clusters to use.

Your code should read in the data, conduct the k-means algorithm on the data, and print to standard output the within cluster sum of squared distances, silhouette coefficient, and the normalized mutual information gain for the clustering results. The sample inputs and outputs we expect to see are as follows (the numbers are fictitious):

```
$python kmeans.py dataFilename 10
WC-SSD: 1000.524
SC: 0.290
NMI: 0.338
```

2.2 Analysis (20 pts)

Consider three versions of the data for each of the questions below:
(i) Dataset 1: use the full dataset `digits-embedding.csv`;
(ii) Dataset 2: use only the subset of the data consisting of the digits 2, 4, 6 and 7; and
(iii) Dataset 3: use only the subset of the data consisting of the digits 6 and 7.

For this part, please put your code in a file called **kmeans_analysis.py**.

1. Cluster the data with different values of $K \in [2, 4, 8, 16, 32]$ and construct two plots for each dataset showing the within-cluster sum of squared distances (WC SSD) and silhouette coefficient (SC) as a function of K.

2. Using the results from Step 1, choose an appropriate K for each dataset and argue why your choice of K is the best. Discuss how the results compare across the two scores and the three versions of the data.

3. Repeat Step 1 ten times using 10 different random seeds (**Note:** set the seed using `np.random.seed(.)`). Measure the average and standard deviation (for WC SSD and SC) for the different values of K and plot them (standard deviations as error bars over the curves of means, like we did in the previous assignment). Discuss what the results show about k-means sensitivity to initial starting conditions.

4. For the value of K chosen in Step 2, cluster the data again (a single time) and evaluate the resulting clusters using normalized mutual information gain (NMI). Calculate and report NMI with respect to the image class labels. Visualize 1000 randomly selected examples in 2d, coloring the points to show their corresponding **cluster** labels. Discuss how both the NMI and visualization results compare across the three versions of the data.

3 Hierarchical Clustering (15 pts)

Consider the following setup for questions on Hierarchical Clustering.

- Data/features: Use the `digits-embedding.csv` data, with the two continuous embedding features.
- Distance measure: Euclidean distance.

1. Put your code for this section in **hierarchical.py**.

2. Create sub-samples for Dataset 1 in Section 2 by sampling 10 images at random from each digit group (i.e., 100 images in total). Use default random seed of 0. This sample will be used for all the analysis steps of Hierarchical Clustering. Use the `scipy` agglomerative clustering method to cluster the data using single linkage. Plot the dendrogram.

3. Cluster the data again, but this time using (i) complete linkage, and (ii) average linkage. Plot the associated dendrograms.

4. Consider cutting each of the dendrograms at successive levels of the hierarchy to produce partitions of different sizes ($K \in [2, 4, 8, 16, 32]$). Construct a plot showing the within-cluster sum of squared distances (WC SSD) and silhouette coefficient (SC) as a function of K.

5. Discuss what value you would choose for K (for each of single, complete, and average linkage) and whether the results differ from your choice of K using k-means for Dataset 1 in Section 2.

6. For your choice of K (for each of single, complete, and average linkage), compute and compare the NMI with respect to the image class labels. Discuss how the NMI obtained compared to the results from k-means on Dataset 1 in Section 2.

Submission Instructions:

Submit through Brightspace. Please submit the report file and the source code files separately. You should submit one pdf file and one zip file through Brightspace.

1. Include in your report which version of Python you are using.
2. Make sure you include in your report all the output and results you get from running your code for all sub-questions. You may include screen shots to show them.
3. Make a directory named $yourFirstName\_yourLastName\_HW4$ and copy all of your files to this directory.
4. **DO NOT** put the datasets into your directory.
5. Make sure you compress your directory into a **zip folder** with the same name as described above, and then upload your zip folder to BrightSpace.
6. Make sure to use any extension days used in your report.
7. We should not need to make any edits to be able to run your codes (including file paths, assume everything will reside in the root folder where your code files are).
8. Do not make any changes in your dataset, you are not submitting your dataset. We will be using the original version of the dataset to run your codes.
9. Stick to the command line arguments required to run each python file.
10. Your README file should include guides how to run the code, and any specifics regarding running your code, if you made any necessary changes mention them there.

Your submission should include the following files:

1. The source code in python.
2. Your evaluation & analysis in .pdf format. Note that your analysis should include visualization plots as well as a discussion of results, as described in details in the questions above.
3. A README file containing your name, instructions to run your code and anything you would like us to know about your program (like errors, special conditions, etc).