CS 412 Machine Problem 4

1 Question Answering

Q1. Please describe in your own words the K-means algorithm. (10 points)

Q2. Suppose you want to cluster five 2D points in the table below (each column is a 2D point). You have got 2 initial centroids (0, 0) and (-1, 0) for the K-means algorithm with K=2. Please manually perform one iteration of this K-means algorithm. What you need to calculate is (1) the cluster assignment for each point, and (2) the updated centroids.

x_1	0	1	0	0	-1
x_2	1	0	0	-1	0

Q3. Please describe in your own words the Expectation-Maximization (EM) algorithm for learning a Gaussian Mixture Model (GMM). No need to write equations. (10 points)

Q4. In what scenarios do you think a GMM is more suitable than K-means for clustering? (10 points)

Q5. Given 3 clusters of 1D points: {1, 2, 3}, {4, 5}, {7}, please perform one iteration of agglomerative clustering using each of the following four cluster similarity metrics: average distance between points, maximum distance between points, minimum distance between points, and distance between means. For each of the four metrics, please include in your answer the clusters of 1D points after this iteration, and justification. Do different cluster similarity metrics give you the same result? (10 points)

2 Programming

The goal of this programming assignment is to let you explore different clustering methods. It is mandatory to use Python 3. Unless otherwise stated, only the numpy package can be used to implement model learning and prediction.

P1. Please follow the steps below to explore the K-means algorithm. (25 points)

1. Use the code snippet below to generate your training data. You will get 200 points in the 2D space. Be careful on the indentation when copying the code.

```
np.random.seed(0)
X1 = np.random.multivariate_normal([2,1], np.diag([0.4, 0.04]), 100)
X2 = np.random.multivariate_normal([1,2], np.diag([0.4, 0.04]), 100)
X = np.concatenate((X1,X2), axis=0)
```

2. Perform K-means to cluster the training data into 2 groups. You can use the K-means code in our code tutorial as a template (available in Blackboard) or build your own code from scratch. Draw in a single figure (1) all training points whose colors reflect their respective cluster assignments, and (2) the two centroids in two colors.

3. Calculate the value of the K-means objective function (the reconstruction error on Page 2 of L14_KMeans.pdf) via the final cluster assignments and centroids you obtained from the K-means output. Print the objective function value.
4. At the end of each of the first 5 iterations, draw in a figure (1) all training points whose colors reflect their current cluster assignments, and (2) the two centroids in two colors. There will be 5 figures in total, each corresponding to one iteration. You will be able to see the change of cluster assignments and centroids during these K-means iterations.
5. At the end of each of the first 5 iterations, calculate and print the value of the K-means objective function. There will be 5 values in total, each corresponding to one iteration. You will be able to see the change of objective function values during these K-means iterations.

P2. Please follow the steps below to explore a Gaussian Mixture Model (GMM). (25 points)

1. Study how to use the GMM implemented in the scikit-learn package on this website[1]. Pay special attention to the example provided on the website, which is also shown below.

```
>>> import numpy as np
>>> from sklearn.mixture import GaussianMixture
>>> X = np.array([[1, 2], [1, 4], [1, 0], [10, 2], [10, 4], [10, 0]])
>>> gm = GaussianMixture(n_components=2, random_state=0).fit(X)
>>> gm.means_
array([[10.,  2.],
       [ 1.,  2.]])
>>> gm.predict([[0, 0], [12, 3]])
array([1, 0])
```

2. Use the GMM implementation in scikit-learn to cluster the training data in P1 into 2 groups. Draw in a single figure (1) all training points whose colors reflect their respective cluster assignments, and (2) the two mean vectors in two colors.
3. Print all learned parameters of this GMM. Hint: What are the parameters in a GMM? Check the description of *Attributes* on the website provided in step 1.
4. Use the learned parameters obtained in the previous step to calculate the responsibility values (check L16_GaussianMixtureModel.pdf) of a new point (1.5, 1.5). Print the two responsibility values corresponding to the two components. Directly calling a Gaussian function from any packages is not allowed. You need to implement your own Gaussian function based on its definition, and then implement the calculation of the responsibility values.
5. Based on the calculated responsibility values, which cluster or Gaussian component should the new point (1.5, 1.5) be assigned to? Is this result the same as the predicted cluster assignment obtained via the *gm.predict* function in scikit-learn? Please print the cluster assignments obtained via these two approaches.

4 Submission

Please follow the instructions below for submission.

- You need to upload two files to Blackboard: a PDF file and a .py file[2]. Do not compress them into a single ZIP file.

[1] https://scikit-learn.org/stable/modules/generated/sklearn.mixture.GaussianMixture.html
[2] Using Jupyter Notebook and submitting a .ipynb file instead of a .py file are fine.

- The PDF file contains all your solutions to this homework. For Question Answering, you can either type answers or handwrite them and take a photo. For Programming, you need to print the results or draw figures, and take screenshots.
- The .py file contains all your code for the programming problems.