IIT BOMBAY

Lab Assignment #11
Points: 5

Main TAs: *Barah Fazili, Tejpalsingh Baljeetsingh Siledar, Ashwin Ramachandran*

Course: *CS 335* – Instructor: *Preethi Jyothi*
Due date: *October 30, 2023*

General Instructions

1. Download lab11.tgz(`https://drive.google.com/file/d/12ps-Z6T60d6ckUAYqSbpw9m_pmtMcae7/view?usp=sharing`) from Moodle, and extract the file to get `lab11`.
2. For your final submission, you need to submit two `.py` files with all the TODOs implemented: `lab11_q1.py` and `lab11_q2.py`. Submit these two files together as a single .tgz file named `[rollnumber].tgz`.
3. Your final submission is due on Moodle on or before **11.59 pm on Oct 30, 2023**.
4. For Q1, you will get 3 points if your mean error across 50 runs matches the error from the solution code ($\approx$ 244). For Q2, you will get 1.5 points for computing the correct error values e using k-means ($e \approx 0.0725$) and GMM ($e = 0$), respectively, and 0.5 points for the plot in `fig.png`.

Q1. *k*-means and *k*-means++

For this problem, you will implement the k-means algorithm from scratch. Given a training set $\mathcal{D} = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$, $\mathbf{x}_i \in \mathbb{R}^d$ and a number of clusters k, the k-means algorithm aims to find k clusters that minimizes the following objective:

$$\min_{\substack{\mu_1,\ldots,\mu_n \\ C_1,\ldots,C_n}} \sum_{i=1}^{k} \sum_{\mathbf{x}\in C_i} ||\mathbf{x} - \mu_i||^2$$

where $\mu_1, \ldots, \mu_n$ are cluster centers and $C_1, \ldots, C_n$ are assignments i.e., C_k consists of all $\mathbf{x}_i$ s.t. $k = \arg\min_j ||\mathbf{x}_i - \mu_j||^2$.
The k-means algorithm is as follows:

```
1: Randomly initialize $\mu_1, \ldots, \mu_k$, $\mu_i \in \mathbb{R}^d$
2: begin
3:     while New cluster assignments are possible do
4:         Compute cluster assignments $C_1, \ldots, C_k$ for each $\mathbf{x}_i$    ▷ Assignment step
5:         That is, $C_k = \{\mathbf{x}_i \mid ||\mathbf{x}_i - \mu_k||^2 \le ||\mathbf{x}_i - \mu_j||^2\} \ \forall j \in \{1, \ldots, k\}$
6:         Recompute the cluster means: $\mu_k = \frac{1}{|C_k|} \sum_{i \in C_k} \mathbf{x}_i$    ▷ Update step
7:     end while
8: end
```

Implement the following two functions in `q1.py`:

1. `k_means`: Implements the k-means algorithm. Returns both cluster centroids and the nearest cluster index assignment for each datapoint.
2. `compute_clustering_error`: Computes the sum of squared distances for each datapoint from its nearest centroid.

Initialization. In `initialize_centroids`, when `default` is set to `True` the cluster centroids are randomly initialized. By setting `default` to `False`, one can invoke the *k-means++ algorithm* that aims to spread apart the k initial cluster centers using the following algorithm:

```
1: Choose the first center $\mu_1$ uniformly at random from among the instances in $\mathcal{D}$.
2: $\mathcal{D} \leftarrow \mathcal{D} - \{\mu_1\}$
3: while $k$ cluster centers have not been chosen do
4:     Compute $\rho(\mathbf{x})$ for each $\mathbf{x}$ in $\mathcal{D}$    ▷ $\rho(\mathbf{x})$ is the shortest distance from $\mathbf{x}$ to the closest center among the ones chosen so far
5:     Choose the next center $\mu$ by selecting an $\mathbf{x}' \in \mathcal{D}$ with probability $\frac{\rho(\mathbf{x}')^2}{\sum_{\mathbf{x}\in\mathcal{D}} \rho(\mathbf{x})^2}$
6:     $\mathcal{D} \leftarrow \mathcal{D} - \{\mu\}$
7: end while
```

Implement this part in `initialize_centroids`.

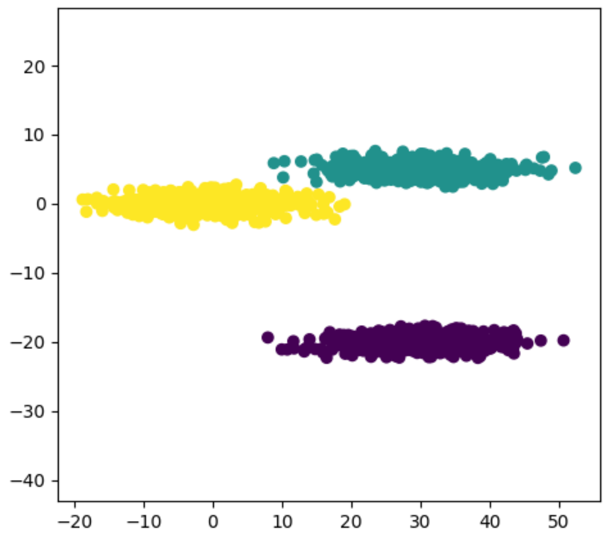

Figure 1: Toy dataset generated using three multivariate Gaussians with diagonal covariance matrices.

Q2. Gaussian Mixture Models (GMMs)

The k-means algorithm tends to identify *spherical* clusters which are not desirable for many real data sets. As we saw in class, a more general clustering with soft assignments can be derived by fitting a Gaussian mixture model (GMM) to the data with the number of mixture components set as the number of clusters.

In this problem, you will work with the toy dataset shown in Figure 1. You will use both k-means and GMM algorithms to find cluster indices for the datapoints in this toy dataset. You can use `sklearn`'s `KMeans` (`https://scikit-learn.org/stable/modules/generated/sklearn.cluster.KMeans.html`) and `GaussianMixture` (`https://scikit-learn.org/stable/modules/generated/sklearn.mixture.GaussianMixture.html`) classes.

Complete the following TODOs outlined in the source file `q2.py`:

1. Predict cluster labels for each datapoint in `X` using both k-means with $k = 3$ and a GMM with three mixture components.
2. Print the clustering error using both k-means and GMM. We define a clustering error metric that measures the purity of each cluster based on ground-truth labels. `true_labels` in `q2.py` contains the ground-truth indices for each datapoint. For each cluster $\mathbb{C}$ discovered by the algorithm, compute the entropy of the label distribution across the points in $\mathbb{C}$. The final error will be the average entropy across all three clusters. Note that for a perfect clustering, this error will evaluate to 0.
3. Plot the three clusters identified using k-means and GMMs. Save the plot in `fig.png`.