Expectation Maximization Clustering

Density Estimation using Gaussian Mixture Models -

- In density estimation, we try to represent the data compactly using a density from a parametric family, e.g., a Gaussian or Beta distribution. For example, we look for the mean and variance of a dataset in order to represent the data compactly using a Gaussian distribution.
- The mean and variance can be found using : maximum likelihood or maximum a posteriori estimation. We can then use the mean and variance of this Gaussian to represent the distribution underlying the data, i.e., we think of the dataset to be a typical realization from this distribution if we were to sample from it.
- A Gaussian mixture model is a density model where we combine a finite number of K Gaussian distributions N [x | μ_k , Σ_k] so that p(x | θ) = $\sum_{k=1}^{K} \pi_k N$ [x | μ_k , Σ_k] where $0 \leq \pi k \leq 1 \quad \sum_{k=1}^{K} \pi k = 1$, where θ := {$\mu_k$, Σ_k , π_k : k = 1, . . . , K} as the collection of all parameters of the model.

Parameter Learning via Maximum Likelihood

- Assume we are given a dataset X = {x1 , . . . , xN }, where x_n { n = 1, . . . , N } are drawn from an unknown distribution p(x). Our objective is to find a good approximation/representation of this unknown distribution p(x) by means of a GMM with K mixture components.
- The parameters of the GMM are the K means μ_k , the covariances Σ_k , and mixture weight π_k .

Expectation Maximization

- The expectation maximization algorithm (EM algorithm) was proposed by Dempster et al. (1977) and is a general iterative scheme for learning parameters (maximum likelihood or MAP) in mixture models and, more generally, latent-variable models where essentially what we want to do is estimate the parameters of K Gaussians from which our data might have generated.
- As the initial step of EM Algorithm we Initialize μ_k , Σ_k , π_k .
- Where μk is the mean for the K^{th} Gaussian distribution

- Σ_k is the covariance (matrix) for Kth Gaussian distribution and π_k is the weight of Kth Gaussian or the prior probability of occurrence of Kth Gaussian.
- Initialization of the parameters μ_k , Σ_k is done in various ways
  - All the three parameters can be initialized randomly but due to this many times the algorithm can get stuck at local optima and fails to converge.
  - Initially k-means is run on the data set and then the k centroids for the k clusters are assigned as means of the k gaussians and covariance across the whole dataset is assigned to Σ_k. (for all k)
- E-step: Evaluate responsibilities or Posterior probabilities r_{nk} for every data point x_n using current parameters π_k , μ_k , Σ_k for each k { k = 1,2, …. N } Gaussian.
- M-step: In this step we re estimate parameters π_k , μ_k , Σ_k using the current responsibilities r_{nk} (from E-step) as given below:
  - $\mu_k = \frac{1}{N_k} \sum_{n=1}^{N} r_{nk} x_n$
  - $\sum k = \frac{1}{N_k} \sum_{n=1}^{N} r_{nk} (x_n - \mu_n) \ (x_n - \mu_n)^T$
  - $\pi_k = \frac{N_k}{N}$
- We iterate over E-step and M-step until convergence i.e until the change observed in Log Likelihood is less than some threshold (meaning if the change in Log Likelihood is very very small).
- Where likelihood is the predictive distribution of the training data given the parameters (μ_k, Σ_k) and log-likelihood is given by
  - $\sum_{n=1}^{N} log \sum_{k=1}^{K} \Pi_k N \, [x_n \mid \mu_k, \Sigma_k]$ which we want to maximize over each iteration.

Implementation Details

- This algorithm is implemented in C++ language and IRIS Dataset has been used to perform the clustering task considering all 4 attributes in the dataset.
- An external library “Eigen” was used to simplify the various matrix operations involved in this algorithm.
- Here as the Iris Dataset consists of 3 classes K (no of gaussians/clusters) is chosen to be 3.
- Very Initial task is proper initialization of the K Gaussians.
- K Gaussians can be initialized either randomly or by considering some facts about the data.
- Both methods of initialization of K gaussians are implemented.
- Informed initialization of the Gaussians was done as follows:

    - There are 3 classes in the Iris dataset so classwise mean was calculated and assigned as the initial mean of the Gaussians.
        - Mean Class 1 = μ_1 , Mean Class 2 = μ_2 , Mean Class 3 = μ_3
    - Covariance for all the 150 data points in the dataset was calculated and assigned as the initial covariance for the Gaussians.
    - Initial weights for the Gaussians i.e π_k (priors) were considered as equal.
        - $\pi_1 = \pi_2 = \pi_3 = ⅓$
- Random initialization of the Gaussians was done as follows:
    - Initially the data points in the dataset are arranged class-wise i.e. examples of class 1 first then examples of class 2 and then of class 3.
    - The 150 points in the dataset were shuffled randomly.
    - Three disjoint sets containing 50 data points each were considered and their mean was calculated which was assigned as the initial mean of the k Gaussians respectively.
        - Mean of Set 1 = μ_1 , Mean of Set 2 = μ_2 , Mean of Set 3 = μ_3
    - Covariance for all the 150 data points in the dataset was calculated and assigned as the initial covariance for the Gaussians.
    - Initial weights for the Gaussians i.e π_k (priors) were considered as equal.
        - $\pi_1 = \pi_2 = \pi_3 = ⅓$
- E - step : Using these three parameters, responsibilities or Posterior Probabilities for each of the 150 data points were calculated with respect to the 3 Gaussians.
- Then Log-Likelihood was calculated with the formula given above
- M - step :Then using the calculated responsibilities, the mean, covariance and priors for the 3 Gaussians were re-estimated by the formulae mentioned above.
- Then again the “E - step” was repeated and new value of Log-Likelihood was calculated, if the absolute difference between the two Log Likelihoods was found to be less than some threshold value then the last “M - step” is executed and the algorithm is ended otherwise the algorithm keeps iterating over the “E - step” and “M - step” until the stopping condition is reached.
- After the algorithm reached the stopping condition the posterior probability for each point with respect to the 3 gaussians was compared and the data point was allotted that cluster which corresponded to the largest posterior probability for that point.