Image Analysis with Gaussian Mixture Model

with PCA for image dimensionality reduction prior to EM algorithm implementation

Summary Report of Analysis

Margaret Catherman

October 1, 2023

Data: The MNIST handwritten digits dataset. For this question, we reduce the dataset to be only two cases, of digits "2" and "6" only. Thus, you will fit GMM with C = 2. The matrix images is of size 784-by-1990, i.e., there are 1990 images in total, and each column of the matrix corresponds to one image of size 28-by-28 pixels (the image is vectorized; the original image can be recovered by mapping the vector into a matrix).

As we are using Gaussian mixture model in this analysis, we use the EM algorithm to find the (local) maximum likelihood estimates of parameters, and not MLE. Maximum likelihood estimation (MLE) is appropriate when all variables are known (Supervised learning). GMM (Gaussian mixture model) is a generic method used for cluster analysis and other instances entailing latent, unknown or hidden variables, that is, the full shape of the data's distribution function may not be known, such as semiparametric modes. In these situations, MLE is not appropriate to directly estimate GMM.

Instead, the expectation–maximization (EM) algorithm is the appropriate way to estimate the GMM model for the MNIST data. The EM algorithm is an iterative method to find (local) maximum likelihood estimates of parameters, where the model depends on unobserved latent variables. The EM steps to learn the parameters that maximize the log−likelihood function for GMMs are:

1) Expectation step (E-step): take expectation over posterior distribution conditioning on data: it can be shown this forms a lower bound (in the t-th iteration); & compute the expectation.
2) Maximization step: Maximize $f\ (\theta)$.

Iterate these two steps till convergence. (Xie, Y., OMSA_Week_7_Slides.pdf)

For the EM algorithm for a Gaussian mixture model (GMM), this is how to use the Bayes rule to drive t_i^k in a closed-form expression:

Bayes rule

$$\underbrace{P(z|x)}_{\text{posterior}} = \frac{\overset{\text{likelihood}}{P(x|z)}\overset{\text{prior}}{P(z)}}{\underbrace{P(x)}_{\text{normalization constant}}} = \frac{P(x,z)}{\sum_z' P(x,z')}$$

Prior: $p(z) = \pi_z$

Likelihood: $p(x|z) = N(x|\mu_z, \sum_z)$

Posterior: $p(z|x) = \frac{\pi_z N(X|\mu_z, \sum_z)}{\sum_{z'} \pi_{z'} N(X|\mu_{z'}, \sum_z{}')})$

E-step: find the posterior distribution

$q(z^1, z^2, \ldots, z^m)$: posterior distribution of the latent variables in t-th iteration

$$q(z^1, z^2, \ldots, z^m) = \prod_{i=1}^{m} p(z^i|x^i, \theta^t)$$

For each data point x^i , compute $p(z^i = k|x^i)$ for each k

$$t_k^i = p(z^i = k|x^i, \theta^t) = \frac{p(x^i|z^i = k)p(z^i = k)}{\sum_{k'-1..k} p(z^i = k', x^i)}$$

$$= \frac{\pi_k N(x^i|\mu_k, \sum_k)}{\sum_{k'=1..k} \pi_{k'} N(x^i|\mu_{k'}, \sum_{k'})}$$

(Xie, Y., OMSA_Week_7_Slides.pdf)

Implementing the EM algorithm for fitting a Gaussian mixture model (GMM)

Implement the EM algorithm for fitting a Gaussian mixture model (GMM**)** for the MNIST handwritten digits dataset. For this question, we reduce the dataset to be only two cases, of digits "2" and "6" only. Thus, you will fit GMM with C = 2.
Use the data file data.mat or data.dat. The matrix images is of size 784-by-1990, i.e., there are 1990 images in total, and each column of the matrix corresponds to one image of size 28-by-28 pixels (the image is vectorized; the original image can be recovered by mapping the vector into a matrix).

First, we use PCA to reduce the dimensionality of the data before applying to EM.

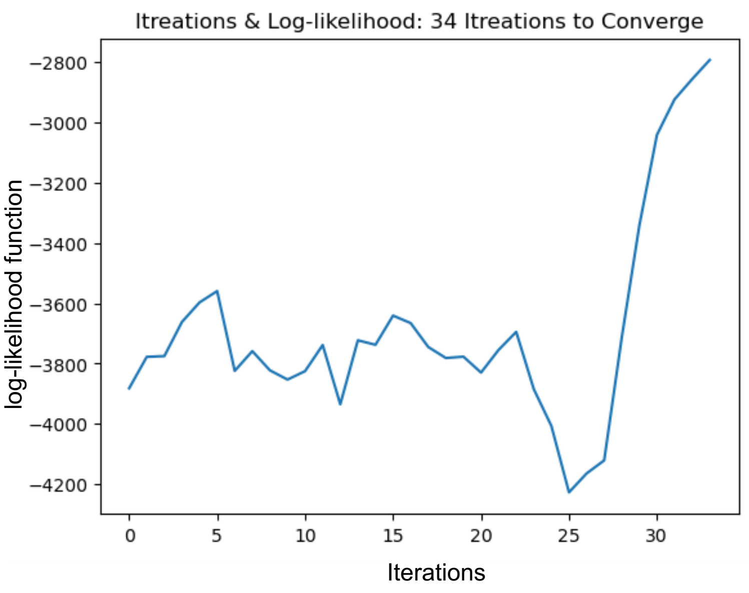
PCA of MNIST handwritten digits dataset. We will put all "6" and "2" digits together, to project the original data into 4-dimensional vectors*. (See code, attached separately.)*

Now implement EM algorithm for the projected data (with 4-dimensions). (In this question, we use the same set of data from the provided data files for training and testing)

(a) Implement EM algorithm yourself. Use the following initialization • initialization for mean: random Gaussian vector with zero mean • initialization for covariance: generate two Gaussian random matrix of size n-byn: S1 and S2, and initialize the covariance matrix for the two components are:

$$\Sigma_1 = S_1 S_1^T + I_n and \Sigma_2 = S_2 S_2^T + I_n$$

where I_n is an identity matrix of size n-by-n. Plot the log-likelihood function versus the number of iterations to show your algorithm is converging, *see right.*



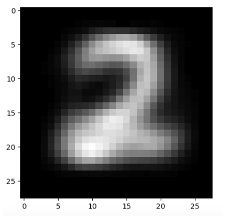
(b) Report, the fitted GMM model when EM has terminated in your algorithms as follows. Report the weights for each component, and the mean of each component, by mapping them back to the original space and reformat the vector to make them into 28-by-28 matrices and show images. Ideally, you should be able to see these means corresponds to some kind of "average" images. You can report the two 4-by-4 covariance matrices by visualizing their intensities (e.g., using a gray scaled image or heat map), *see below, right.*

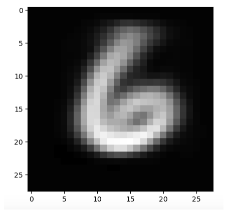
c) Use the τik to infer the labels of the images, and compare with the true labels. Report the mis-classification rate for digits "2" and "6" respectively, *see bottom right.*

Perform K-means clustering with K = 2 (you may call a package or use the code from your previous homework). Find out the mis-classification rate for digits "2" and "6" respectively, and compare with GMM. Which one achieves the better performance?

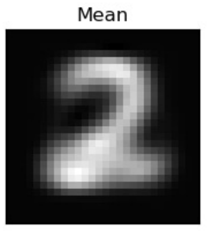
Based on the assessments used, the accuracy of GMM is higher than that for the k-means, see findings, *right.* While the metrics used for accuracy assessment are slightly different for each approach, superior scores for both metrics are closer to 1, & inferior scores are closer to 0.

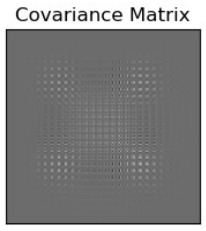
K-means clustering

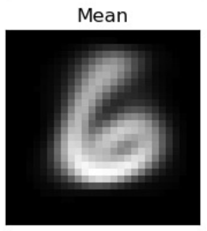


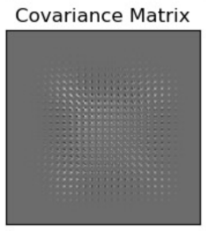


GMM









Purity score,
Manhattan
Distance,
both centroid: 0.253

```
Accuracy 2: 0.795
Accuracy 6: 0.998
```