Homework 1: Expectation Maximization

Robert Grant
2009-09-06

1 Expectation Maximization

The Expectation Maximization algorithm itself is an iterative method of finding the maximum likelihood estimate of parameters of a supposed model given a set of data. This algorithm consists of two main steps: the Expectation step or "E" step, and the Maximization step or "M" step. After some initialization, in the E step one computes the posterior probabilities of the datapoints given that they come from a distribution with one's estimated parameters. Then, in the M step, those parameters are recomputed after reclassifying the data according to the probabilities computed in the E step. The E step and M step are repeated until the algorithm is deemed to converge; this can be when the change in the log-likelihood function (or in the estimated parameters) across iterations has fallen below a threshold.

The purpose of this homework is to explore the properties of Expectation Maximization on data modelled as a mixture of two gaussians. Thus, we model the data as having a PDF of the form

$$p(\mathbf{x}) = \sum_{k=1}^{2} \pi_k N(\mathbf{x}|\mu_\mathbf{k}, \mathbf{\Sigma_k}), \tag{1}$$

where μ and Σ are the mean vector and covariance matrix for each component gaussian, and π is mixing coefficient (the prior probability) for each component gaussian. Since this distribution is completely determined by the parameters π, μ, and Σ, these are the parameters we are trying to estimate with Expectation Maximization.

2 Implementation

My implementation is contained in three files: `projectFile.m`, `em.m`, and `classify_gaussian.m`. The Expectation Maximization algorithm itself is implemented in the function file `em.m`. In the function file `classify_gaussian.m` I have implemented a couple different ways of classifying the points after Expectation Maximization. Finally, the script file `projectFile.m` generates data, calls the two function files, and generates plots. Each file is commented extensively, and usage of the two function files is accessible from the Matlab interactive prompt though the usual `help function_name` syntax.

3 Results

For all experiments, I generated 2000 data points from two, 2-d gaussians. To determine convergence, I set a threshold for how much the mean estimate changed between runs. Specifically, I was satisfied with convergence when

$$\|m_1 - m_{1old}\| < threshold, \tag{2}$$

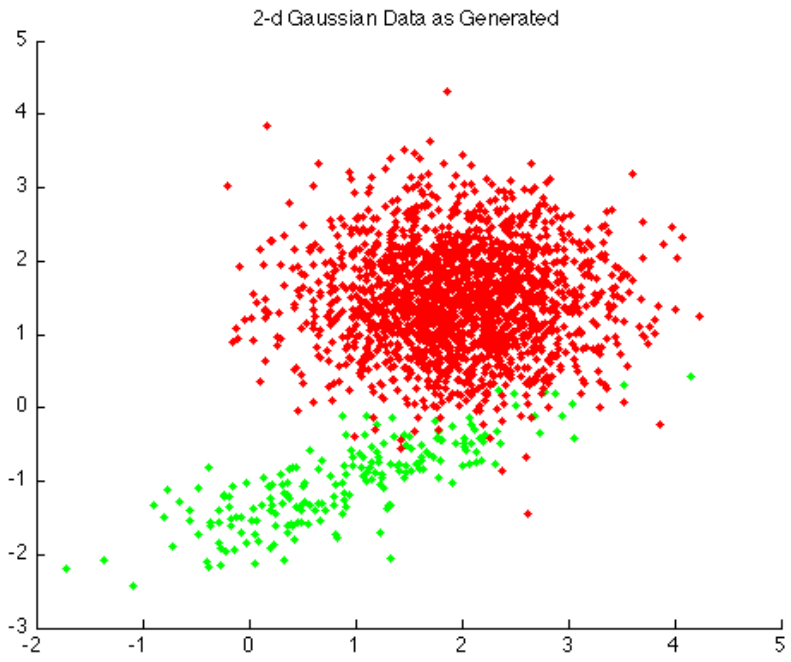

Figure 1: Data with classification from prior knowledge.

and

$$\|m_2 - m_{2old}\| < threshold. \tag{3}$$

As one of my experiments, I generated data from a mixture of two gaussians with the following parameters:

$$\pi_1 = 0.1, \qquad \pi_2 = 0.9 \tag{4}$$
$$\mu_1 = [-1, 1], \qquad \mu_2 = [2, 1.5] \tag{5}$$
$$\Sigma_1 = \begin{bmatrix} 0.9 & 0.4 \\ 0.4 & 0.3 \end{bmatrix}, \qquad \Sigma_2 = \begin{bmatrix} 0.5 & 0.0 \\ 0.0 & 0.5 \end{bmatrix} \tag{6}$$

Figure 1 shows the generated data, classified with prior knowledge. Figures 2 and 3 show the result of expectation maximization, with the results of two different methods of classification. In Figure 2, after EM the data were classified according to which gaussian they were more likely to have come from. In Figure 3, the data were randomly classified, with the classification weighted based on likelihood. I refer to these methods as "hard" and "soft" classification, and I believe "soft" classification is the method recommended from class. As you can see in the figures, under "hard" classification there is a clearly recognizable boundary where points are classified into one gaussian or the other, whereas with "soft" classification there is a region where points may be classified as belonging to either distribution.

The EM algorithm seems to have done very well. The results shown were generated after 28 iterations.

The difference in the classification types can be even more clearly seen when the means are close together.

Figure 2: Data with ‘hard‘ classification.

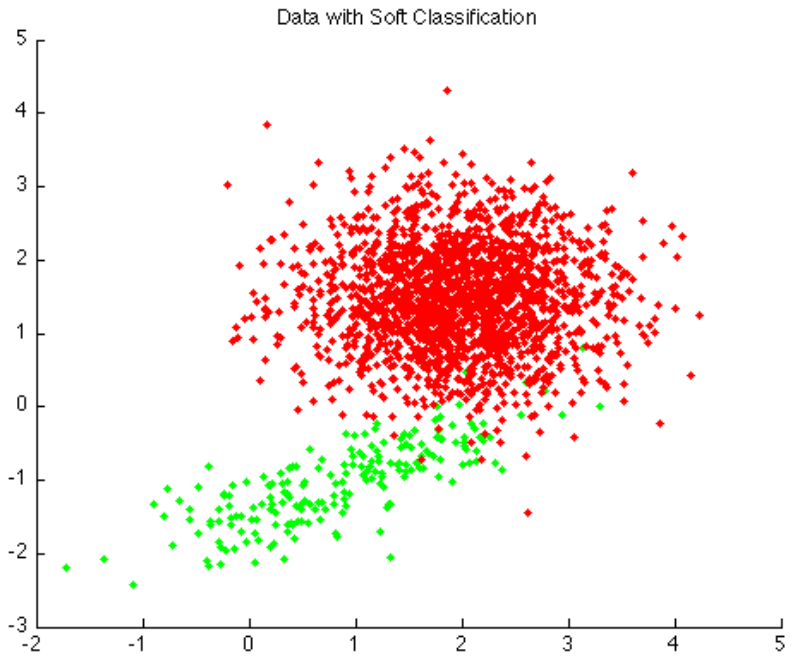

Figure 3: Data with ‘soft‘ classification.

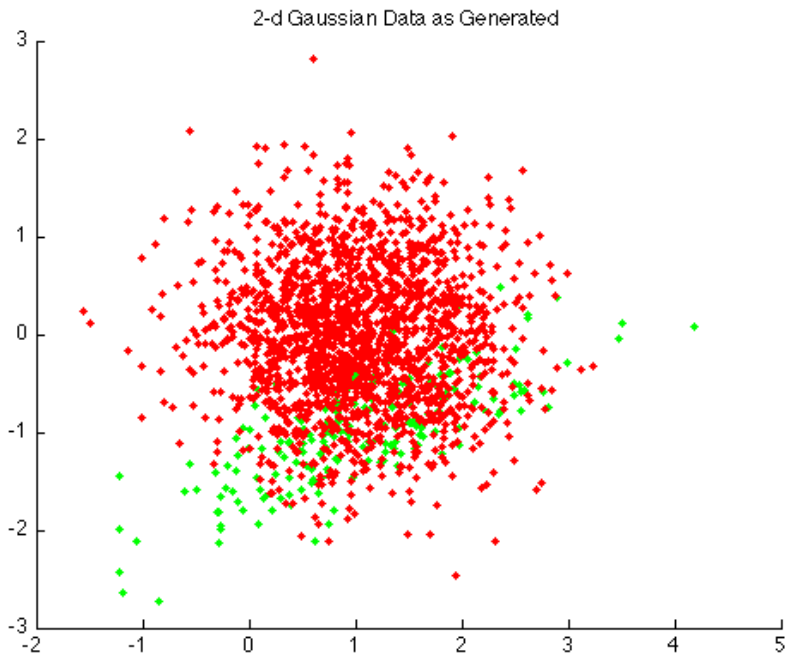

Figure 4: Data with classification from prior knowledge.

For example, Figures 4, 5, and 6 show the above experiment repeated with means much closer together ($\mu_1 = [1, -1]$ and $\mu_2 = [1, 0]$). The "soft" classification method, though still not great, more accurately represents the distribution of the data than the "hard" classification method.

4 Other Observations

Since this report is already running long, I will briefly summarize two other observations. From testing different distributions, it is clear that expectation maximization has more difficulty being accurate the "closer" distributions are to each other, where "close" is a combination of small distance between the means and large values for variance.

Also, when using a fixed threshold to test for convergence as I have, one must be very aware of the scale of one's parameters. Using too large a threshold will allow the algorithm to terminate before being effective, and using too small a threshold will cause the algorithm to run too long. There is probably a better way to threshold based on percentage of a parameter's size, so it scales itself automatically to the scale of the problem.

References

[1] Christopher M. Bishop, *Pattern Recognition and Machine Learning*, Springer, 2007.

Figure 5: Data with ‘hard‘ classification.

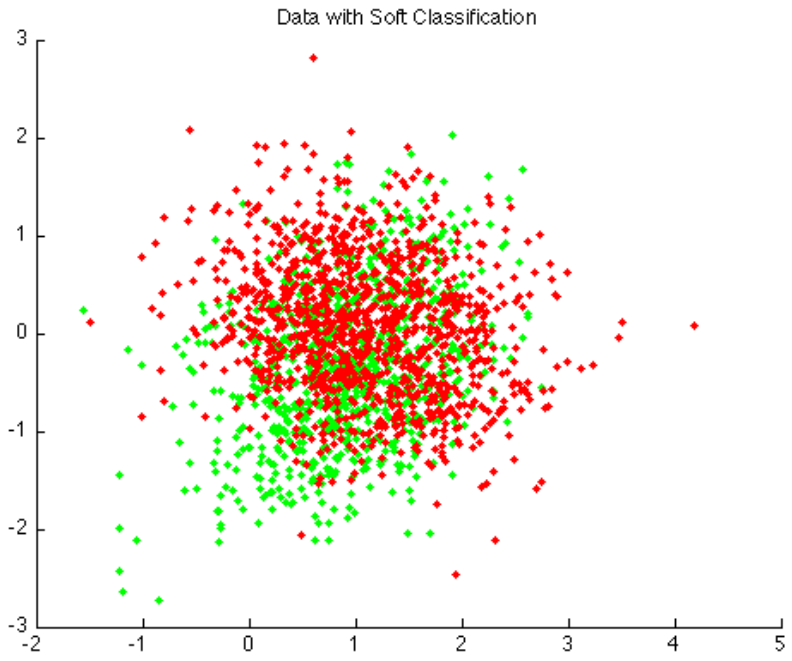

Figure 6: Data with ‘soft‘ classification.