CSC412 – Assignment #3

Due: March8, 10am at the **START** of tutorial
Worth: 10%
Late assignments not accepted.

1 Learning Mixtures of Gaussians

In this assignment (which has only one question!) you will implement the EM algorithm for training a mixture of full covariance Gaussians:

$$p(\mathbf{x}) = \sum_k p(z=k)p(\mathbf{x}|z=k) = \sum_k \alpha_k |2\pi\Sigma_k|^{-1/2} \exp\left\{-\frac{1}{2}(\mathbf{x}-\mu_k)^\top \Sigma_k^{-1}(\mathbf{x}-\mu_k)\right\}$$

1.1 What to do

- Using the data provided in `a3geyser.mat`, train mixtures with 1,2,3 and 6 components, starting at 1000 random initializations and training until convergence.

  In case you are interested, the data represents intervals between erruptions and durations of erruptions of the Old Faithful Geyser at Yellowstone National Park, USA.

1.2 What to hand in

- The equation you are using to update the covariance matrices in the M-step (see below).
- A plot of the best fit (highest likelihood) that you found for each number of components, showing the training data as points and the one-standard deviation ellipses of the Gaussians. (4 plots total, on a single page.)
- A histogram, for each number of components, of the final log likelihoods of the training data and of the number of iterations of EM it took for training to converge. Don't use too few or too many bins in your histograms. (8 histograms total, arranged in two columns of 4, on one page.)
- Your MATLAB code, properly commented (5 pages maximum).

1.3 Some hints

- You might find the the function `plotGauss.m` helpful in making the one-sigma Gaussian ellipse plots.
- You can assess convergence of EM by monitoring the log likelihood and stopping when the fractional change `(newlik-oldlik)/newlik` is less than one part in a million. You log likelihood should never decrease!
- Try to get the EM algorithm working *without* the covariance updates at first (ie just update the means). Then when that is working, and your likelihood is always going up, try adding in the covariance updates.
- Initialize the model parameters by setting the means randomly in some sensible range, for example the covariance of the entire data set. Set the covariances to some small but not tiny values. Make sure there is enough randomness in your initialization that you fall into different local maxima.

- You might find it interesting to plot the parameters after every iteration of EM, so you can see how things are going. If your code issues a `drawnow` command in MATLAB after plotting the data and all the Gaussian ellipses this will update the plots on the screen. (If these ellipses don't fit the data at all, you may have an error somewhere in your code or in your covariance update equation!)
- Every once in a while, if you are unlucky, a Gaussian cluster will get trapped on a single datapoint and its variance will get very small. This can cause numerical problems, so watch out for it. If this happens, you can just restart at a different random initialization.

1.4 Reminders

Amongst other things, your code will need to:

- Compute the log probability of every datapoint under each mixture component:

$$\ell_{nk} = \log p(\mathbf{x}^n, z = k | \mu_k, \Sigma_k)$$

- Compute the log probability of every datapoint:

$$\ell_n = \log p(\mathbf{x}^n) = \log \sum_k p(\mathbf{x}^n, z = k | \mu_k, \Sigma_k)$$

NB: Make sure to do this in a numerically stable way. In particular, *do not* use $\ell_n = \log \sum_k \exp(\ell_{nk})$.
Hint: You may find the function `logsum.m` helpful.

- Compute the E-step "responsibilities":

$$r_{nk} = p(z = k | \mathbf{x}^n) = \frac{p(\mathbf{x}^n, z = k)}{p(\mathbf{x}^n)} = \exp\left(\ell_{nk} - \ell_n\right)$$

(You probably don't want to use $\exp(\ell_{nk}) / \exp(\ell_n)$. Think about why.)

- Update the mixing proportions in the M-step:

$$\alpha_k^{new} = \frac{\sum_n r_{nk}}{N}$$

- Update the means in the M-step:

$$\mu_k^{new} = \frac{\sum_n r_{nk} \mathbf{x}^n}{\sum_n r_{nk}}$$

- Update the covariance matrices Σ_k in the M-step.
(I've left this equation for you to write down from the class notes and the book.)
Use full (not diagonal or sphereical) covariances.

$$\Sigma_k^{new} = ???$$

- At each iteration, you should also compute the total log probability of the training set:

$$\ell = \sum_n \log p(\mathbf{x}^n) = \sum_n \ell_n$$