Problem Set 5

MAS 622J/1.126J: Pattern Recognition and Analysis

Due Monday, 8 November 2010. Resubmission due, 10 November 2010

Note: All instructions to plot data or write a program should be carried out using Matlab. In order to maintain a reasonable level of consistency and simplicity we ask that you do not use other software tools.

If you collaborated with other members of the class, please write their names at the end of the assignment. Moreover, you will need to write and sign the following statement: "In preparing my solutions, I did not look at any old homeworks, copy anybody's answers or let them copy mine."

Problem 1: Mixture of Gaussians [35 points]

Implement the EM algorithm for estimating the parameters of a mixture of Gaussians with isotropic covariances $\Sigma_j = \sigma_j I$. Note: Download the data file from the course website. There are two datasets each of which is two-dimensional.

To solve this problem, and just this problem, you can write your own code or use any MATLAB toolboxes available for the purpose. In particular, there is a MATLAB mixture of Gaussians algorithm available for download here that would be good to explore (Note: If you use this you will need to make some small adjustments to the code. Specifically to handle the requirement for isotropic covariances.):

`http://www.lx.it.pt/~mtf/mixturecode2.zip`

Also see the accompanying paper "Unsupervised Learning of Finite Mixture Models", M. Figueiredo and A.K. Jain.

a. Experiment with the number of mixtures and comment on the tradeoff between the number of mixtures and goodness of fit (i.e. loglikelihood) of the data. Suggestion: Plot the loglikelihood as a function of the number of components of a mixture of Gaussians to support your argument.

b. Find a fixed number of Gaussians that works well for each data set.

c. Plot the estimated Gaussians as one-sigma countours of each mixing component on top of the training data.

d. List mean, covariance and mixing weights of each mixture component.

e. Include your source code.

Problem 2: k-Means [30 points]

a. Prove that k-Means algorithm is guaranteed to converge to a solution. Use $C(i)$ as the cluster associated to the data point x_i, where $i = 1 \ldots n$.

b. Does the algorithm always converge to the same solution? Prove this or provide a counter example.

Problem 3: k-Nearest-Neighbor vs Generalized Linear Discriminant [35 points]

Compare the performance of Generalized Linear Discriminant (GLD) and kNN to detect cases of breast cancer [1]. For each of the methods you will need to find their optimal parameter. That is, k for kNN and the polynomial degree p for GLD (i.e. $y^T = [1 \; x \; x^2 \; \ldots x^p]$). For the latter case, let the margin vector be $\mathbf{b} = \mathbf{1}$.

These undetermined parameters will be estimated using leave-one-out validation as follows:

- Break the training data set into two parts, A and B, where B contains only a single sample.
- Train your algorithms using A only.
- Determine if B (the sample left out) is classified correctly.
- Repeat this process for every possible choice of B. Define $cv(h)$ as the leave-one-out accuracy of one algorithm for the parameter h.

a. Use the training data to generate $cv(i)$ for both algorithms. You should constrain the parameter search to $i = [1 : 2 : 20]$ for kNN and $i = [1 : 7]$ for GLD. Plot your cv's and indicate the best parameters. [You can use the helper function "*[Xtrain Xtest Ytrain Ytest] = loadData;*"]

b. Use the previous parameters and all your training data to re-train your algorithms. What are the accuracies for the testing data set?

[1] This is an altered version of the Breast Cancer Wisconsin Data Set located at the UCI Machine Learning repository (http://archive.ics.uci.edu/ml/datasets.html)