STA 414/2104, Spring 2014 — Assignment #2

Due at the start of class on March 20. Please hand it in on 8 1/2 by 11 inch paper, stapled in the upper left, with no other packaging.

This assignment is to be done by each student individually. You may discuss it in general terms with other students, but the work you hand in should be your own. In particular, you should not leave any discussion with someone else with any written notes (either on paper or in electronic form).

In this assignment, you will classify handwritten digits with mixture models fitted by maximum penalized likelihood using the EM algorithm.

The data you will use consists of 800 training images and 1000 test images of handwritten digits (from US zip codes). I derived these images from the well-known MNIST dataset, by randomly selecting images from the total 60000 training cases provided, reducing the resolution of the images from 28×28 to 14×14 by averaging 2×2 blocks of pixel values, and then thresholding the pixel values to get binary values. A data file with 800 lines each containing 196 pixel values (either 0 or 1) is provided on the course webpage. Another file containing the labels for these 800 digits (0 to 9) is also provided. Similarly, there is a file with 1000 test images, and another file with the labels for these 1000 test images. You should look at the test labels only at the very end, too see how well the methods do.

In this assignment, you should try to classify these images of digits using a generative model, from which you can derive the probabilities of the 10 possible classes given the observed image of a digit. You should guess that the class for a test digit is the one with highest probability (ie, we will use a loss function in which all errors are equally bad).

The generative model we will use estimates the class probabilities by their frequencies in the training set (which will be close to, but not exactly, uniform over the 10 digits) and estimates the probability distributions of images within each class by mixture models with K components, with each component modeling the 196 pixel values as being independent. It will be convenient to combine all 10 of these mixture models into a single mixture model with $10K$ components, which model both the pixel values and the class label. The probabilities for class labels in the components will be fixed, however, so that K components give probability 1 to digit 0, K components give probability 1 to digit 1, K components give probability 1 to digit 2, etc.

The model for the distribution of the label, y_i, and pixel values $x_{i,1}, ..., x_{i,196}$, for digit i is therefore as follows:

$$P(y_i, x_i) \;=\; \sum_{k=1}^{10K} \pi_k \, q_{k,y_i} \prod_{j=1}^{196} \theta_{k,j}^{x_{i,j}} (1-\theta_{k,j})^{1-x_{i,j}}$$

The data items, (y_i, x_i), are assumed to be independent for different cases i. The parameters of this model are the mixing proportions, $\pi_1, \ldots, \pi_{10K}$, and the probabilities of pixels being 1 for each component, $\theta_{k,j}$ for $k = 1, \ldots, 10K$ and $j = 1, \ldots, 196$. The probabilities of class labels for each component are fixed, as

$$q_{k,y} \;=\; \begin{cases} 1 & \text{if } k \in \{Ky+1, \ldots, Ky+K\} \\ 0 & \text{otherwise} \end{cases}$$

for $k = 1, \ldots, 10K$ and $y = 0, \ldots, 9$.

You should write an R function to try to find the parameter values that maximize the log likelihood from the training data plus a "penalty". (Note that with this "penalty" higher values are better.) The EM algorithm can easily be adapted to find maximum penalized likelihood estimates rather than maximum likelihood estimates — referring to the general version of the algorithm that is presented in the lecture slides, the E step remains the same, but the M step will now maximize $E_Q[\log P(x, z|\theta) + G(\theta)]$, where $G(\theta)$ is the "penalty".

The "penalty" to use is designed to avoid estimates for pixel probabilities that are zero or close to zero, which could cause problems when classifying test cases (for example, zero pixel probabilities could result in a test case having zero probability for every possible digit that it might be). The penalty to add to the log likelihood should be

$$G(\theta) \;=\; \alpha \sum_{k=1}^{10K} \sum_{j=1}^{196} \Big[\log(\theta_{k,j}) + \log(1-\theta_{k,j}) \Big]$$

Here, α controls the magnitude of the penalty. For this assignment, you should fix α to 0.05, though in a real application you would probably need to set it by some method such as cross validation.

You may write your R function by modifying the example EM function on the course web page, although since it would need considerable modifications, you might instead decide to just start anew. The re-estimation of the mixing proportions, π, in the M step will be similar to the example, but to re-estimate the θ parameters, you must take account of the penalty. The resulting formula for the update in the M step is

$$\hat{\theta}_{k,j} \;=\; \frac{\alpha + \sum_{i=1}^{n} r_{i,k} x_{i,j}}{2\alpha + \sum_{i=1}^{n} r_{i,k}}$$

where $r_{i,k}$ is the probability that case i came from component k, estimated in the E step. You should hand in a derivation of this formula from the general form of the EM algorithm presented in the lecture slides (modified as above to include a penalty term).

Your function implementing the EM algorithm should take as arguments the images in the training set, the labels for these training cases, the number of mixture components for each digit class (K), the penalty magnitude (α), and the number of iterations of EM to do. It should return a list with the parameter estimates (π and θ) and responsibilities (r). You will need to start with some initial values for the responsibilities (and then start with an M step). The responsibility of component k for item i should be zero if component k has $q_{k,y_i} = 0$. Otherwise, you should randomly set $r_{i,k}$ from the uniform distribution between 1 and 2 and then rescale these values so that for each i, the sum over k of $r_{i,k}$ is one.

After each iteration, your EM function should print the value of the log likelihood and the value of the log likelihood plus the penalty function. The latter should never go down — if it does, you have a bug in your EM function. You should use enough iterations that these values have almost stabilized by the last iteration.

You will also need to write an R function that takes the fitted parameter values from running EM and uses them to predict the class of a test image. This function should use Bayes' Rule to find the probability that the image came from each of the $10K$ mixture components, and then add up the probabilities for the K components associated with each digit, to obtain the probabilities of the image being of each digit from 0 to 9. It should return these probabilities, which can then be used to guess what the digit is, by finding the digit with the highest probability.

You should first run your program EM and prediction functions for $K = 1$, which should produce the same results as the naive Bayes method would. (Note that EM should converge immediately with $K = 1$.) You should then do ten runs with $K = 5$ using different random number seeds, and see what the predictive accuracy is for each run. Finally, for each test case, you should average the class probabilities obtained from each of the ten runs, and then use these averaged probabilities to classify the test cases. You should compare the accuracy of these "ensemble" predictions with the accuracy obtained using the individual runs that were averaged.

You should hand in your derivation of the update formula for $\hat{\theta}$ above, a listing of the R functions you wrote for fitting by EM and predicting digit labels, the R scripts you used to apply these functions to the data provided, the output of these scripts, including the classification error rates on the test set you obtained (with $K = 1$, with $K = 5$ for each of ten initializations, and with the ensemble of ten fits with $K = 5$), and a discussion of the results. Your discussion should consider how naive Bayes ($K = 1$) compares to using a mixture (with $K = 5$), and how the ensemble predictions compare with predicting using a single run of EM, or using the best run of EM according to the log likelihood (with or without the penalty).