CSC412 – Assignment #2

Due: Feb.8 2006, 2pm at the **START** of class
Worth: 15%
Late assignments not accepted.

1 Handwritten Digit Classification

For this question you will build three classifiers to label images of handwritten digits collected by the United States Post Office. The images are 8 by 8 pixels in size, which we will represent as a vector $\mathbf{x}$ of dimension 64 by listing all the pixel values in "raster scan" order. The labels y are $1, 2, \ldots, 9, 10$ corresponding to which character was written in the image. The label 10 is used for the digit "0". There are 700 training cases and 400 test cases for each digit; they can be found in the file `a2digits.mat` or `a2digits.zip`.

- As a warm up question, load the data and plot a few examples. Decide if the second dimension (component) of the input vector is a pixel in the second row or the second column of the image, and write that somewhere on your answer to this question.

1.1 Conditional Gaussian Classifier Training

- Using maximum likelihood, fit a set of 10 class-conditional Gaussians with a separate, full covariance matrix for each class. In particular, fit the model below to maximize the average of $\log p(\mathbf{x}, y)$ on the training set (where D is the dimension of $\mathbf{x}$ (64 in our case) and $|M|$ is the determinant of the matrix M).

$$
\begin{aligned}
p(y = k) &= \alpha_k \\
p(\mathbf{x}|y = k) &= (2\pi)^{-D/2}|\Sigma_k|^{-1/2} \exp\left\{-\frac{1}{2}(\mathbf{x} - \mu_k)^\top \Sigma_k^{-1}(\mathbf{x} - \mu_k)\right\}
\end{aligned}
$$

- You should get parameters μ_k and Σ_k for $k \in (0 \ldots 9)$. (The ML estimates for α_k=1/10 since all classes have the same number of observations in this training set.)
- After fitting the Gaussians, regularize each class' covariance matrix by adding a small amount of the identity matrix. (For this assignment, add $0.01I$ to the sample covariance matrix.)
- Hand in plot showing an 8 by 8 image of each mean μ_k, and below the mean another image showing the log of the diagonal elements of the covariance matrix Σ_k. Plot all ten classes side by side, and use the same grayscale for all 10 means so that no pixels are saturated. Use another grayscale for all 10 log variances. Indicate in some way the grayscales you used, and also indicate for each if white means larger values or if black means larger values.

1.2 Naive Bayes Classifier Training

- Convert the real-valued features $\mathbf{x}$ into binary features $\mathbf{b}$ by thresholding: b_n=1 if $x_n > 0.5$ otherwise $b_n = 0$.

- Using these new binary features $\mathbf{b}$ and the class labels, train a Naive Bayes classifier on the training set. In particular, fit the model below to maximize the average of $\log p(\mathbf{b}, y)$ on the training set.

$$
\begin{aligned}
p(y=k) &= \alpha_k \\
p(b_n=1|y=k) &= \eta_{kn} \\
p(\mathbf{b}|y=k,\eta) &= \prod_n (\eta_{nk})^{b_n}(1-\eta_{nk})^{(1-b_n)}
\end{aligned}
$$

- You should get parameters $\eta_{kn} \equiv p(b_n = 1|y = k)$ for $k \in (0 \ldots 9), n \in (1 \ldots 64)$.
(Again, all class priors α_k are equal since all classes have the same number of observations.)

- Hand in plot showing an 8 by 8 image of each vector $\log(\eta_{k\cdot}./(1-\eta_k))$, all ten side by side sharing a single grayscale. Indicate the grayscale as above.

1.3 Logistic Regression Classifier Training

- Using maximum conditional likelihood, fit a logistic regression classifier to the raw pixel data, after augmenting the vector $\mathbf{x}$ with a constant (bias) term $x_0 = 1$. In particular, fit the model below to maximize the average of $\log p(y|\mathbf{x})$ on the training set:

$$
p(y=k|\mathbf{x},\theta) = \frac{e^{\theta_k^\top \mathbf{x}}}{\sum_j e^{\theta_j^\top \mathbf{x}}}
$$

- You should get parameters θ_{kn} for for $k \in (0 \ldots 9), n \in (0 \ldots 64)$ (n=0 is the constant or bias term).

- Hand in plot showing an 8 by 8 image of each vector θ_k (excluding θ_{k0}), all ten side by side sharing a single grayscale. Indicate the grayscale as above. (You don't need to plot the bias terms.)

- Remember that you cannot find the model parameters *directly*; instead you need to compute the gradient of the average conditional log likelihood with respect to θ and use that gradient to iteratively maximize the objective. You are free to do this in any way you choose, but the following adaptive gradient procedure, starting with random parameters, should work pretty nicely:

  1. Measure the current objective and gradient.
  2. Adjust the parameters by subtracting Δ times the gradient.
  3. Measure the new objective and new gradient.
  4. If the new objective is better, set $\Delta = 1.1\Delta$
     else if the new objective is worse, go back to the old parameters, objective and gradient, set $\Delta = .5\Delta$.
  5. If $\Delta > \Delta_{min}$, go to (2).

1.4 Performance Evaluation

- DO NOT HAND IN ANY CODE.

- Using the parameters you fit on the training set compute $\log p(y|\mathbf{x})$ for each of the training and test cases under all three of the Gaussian-conditionals, logistic regression, and Naive Bayes models.

- Select the most likely posterior class for each training and test case under each classifier. If this matches the label, the classifier is correct. If not, the classifier has made an error. Hand in a 3 column by 22 row table showing how many errors (out of 400) each classifier makes on each of the 10 training and test sets and what the overall training and testing error rate (in %) is.

- Compute the average conditional log likelihood achieved by each classifier on the training set and test set. Average both over data cases and digit classes.

1.5 MATLAB Tips

If you are using MATLAB , here are some tips:

- You might find the commands `caxis` and `colorbar` helpful in setting and indicating your grayscale.
- The `imagesc` function can be used to display vectors as images. In particular, try the line:
  `imagesc(reshape(xx,8,8)'); axis equal; axis off; colormap gray;`
  to display the vector `xx`.
- The `repmat` command in conjunction with `sum` and the operators .* and ./ are helpful in renormalizing arrays so that the rows or columns sum to one.
- The expression `(M > a)` for a matrix `M` and a scalar `a` performs the comparison at every element and evaluates to a binary matrix the same size as `M`.