CSCI 1950-F Homework 8: PCA & EM Algorithm

Brown University, Spring 2011

Homework due at 11:59pm on April 29, 2011

We begin by revisiting the Bayesian linear regression model from homework 4. As before, given M basis functions $\phi_j(x)$, $j = 1, \ldots, M$, we model the dependence of response variables y_i on input covariates x_i as follows:

$$p(y_i \mid x_i, w, \beta) = \mathcal{N}(y_i \mid w^T \phi(x_i), \beta^{-1}) \qquad p(w \mid \alpha) = \mathcal{N}(w \mid 0, \alpha^{-1} I_M)$$

Rather than searching over a discrete grid of potential values for the hyperparameters α and β, we will instead estimate them via the EM algorithm. In the E-step, we compute the expected value of certain statistics of the M-dimensional vector of regression coefficients w. In the M-step, we use these to produce new estimates of α and β. The previously distributed solutions for homework 4 may also be useful.

Question 1:

a) For the normal distributions assumed above, derive the form of the expected "complete-data" log likelihood, $\mathbb{E}[\log p(y, w \mid x, \alpha, \beta)]$, given N observations $y = (y_1, y_2, \ldots, y_N)$ of inputs $x = (x_1, x_2, \ldots, x_N)$. It may be helpful to examine the EM derivation for the factor analysis model. What particular statistics do we need to determine the expectations of in the E-step, in order to concretely evaluate this expression?

b) Take the derivative of the expression in part (a) with respect to α, set it to zero, and determine the M-step update of α.

c) Take the derivative of the expression in part (a) with respect to β, set it to zero, and determine the M-step update of β.

d) Using the equations from the preceding parts, implement the EM algorithm for this model. Consider two different families of basis functions, the polynomial and radial basis functions from homework 4, both with order $M = 100$. For each family, initialize $\alpha^{(0)} = 0.01$, $\beta^{(0)} = 0.0025$, and run EM until changes in the likelihood fall below 10^{-6}. Plot the resulting sequences of parameters $\alpha^{(t)}$ and $\beta^{(t)}$, as well as the corresponding likelihoods $p(y \mid \alpha^{(t)}, \beta^{(t)})$ (see the formula from homework 4, problem 3(a)). What is the test accuracy of the resulting models, using the squared-error loss from homework 4?

The next question asks you to devise an EM algorithm for what is sometimes called *co-clustering*. Suppose our observations are sequences of pairs $D = \{(x_1, y_1), \ldots, (x_N, y_N)\}$, where $x_i \in \mathcal{X}$ and $y_i \in \mathcal{Y}$ for $i = 1, \ldots, N$. Here, $\mathcal{X}$ and $\mathcal{Y}$ are (possibly different) discrete spaces. Our goal is to jointly cluster the x and y variables based on D. For example, in a computational linguistics application D might consist of *(Adjective, Noun)* pairs that appear adjacent to each other in a corpus: *strong coffee*, *important people*, *sublime music*, etc. Our goal is to cluster all of the adjectives that co-occur with similar nouns together, and to cluster all of the nouns that co-ocurr with similar adjectives together.

In general, we'll model the data by assuming that each (x_i, y_i) pair is generated by a pair of hidden classes (u_i, v_i), where $u_i \in \mathcal{U} = \{1, \ldots, L\}$ is the class (or cluster) that generates x_i, and $v_i \in \mathcal{V} = \{1, \ldots, M\}$ is the class that generates y_i. In particular, for a pair (x_i, y_i) generated from (u_i, v_i), we assume the following probabilistic model:

$$p(u_i, v_i, x_i, y_i) = p(u_i, v_i)p(x_i \mid u_i)p(y_i \mid v_i)$$

We parameterize our model as follows:

$$p(u, v) = \pi_{uv} \qquad p(x \mid u) = \sigma_{x|u} \qquad p(y \mid v) = \varphi_{y|v}$$

Here, π_{uv} is a categorical distribution on $\mathcal{U} \times \mathcal{V}$, $\sigma_{x|u}$ is a categorical distribution on $\mathcal{X}$, and $\varphi_{y|v}$ is a categorical distribution on $\mathcal{Y}$. To simplify the notation, we'll use $\boldsymbol{\theta} = (\boldsymbol{\pi}, \boldsymbol{\sigma}, \boldsymbol{\varphi})$ to refer to the parameters of these distributions.

Question 2

To develop our EM algorithm, we'll first develop an estimator for completely observed data $D' = \{(u_1, v_1, x_1, y_1), \ldots, (u_N, v_N, x_N, y_N)\} = \{u, v, x, y\}$, i.e., a dataset where the cluster assignments for each data pair are directly observed.

a) *Give an expression for the complete data log-likelihood $\log p(u, v, x, y \mid \boldsymbol{\theta})$. Derive maximum likelihood estimators for $\boldsymbol{\theta} = (\boldsymbol{\pi}, \boldsymbol{\sigma}, \boldsymbol{\varphi})$ given D'. Use compact notation for counts, for example n_{uv} for the number of times that (u, v) co-occur in D'. Be sure to define all terms that you use.*

b) *Now suppose that the class labels are unobserved, so that $D = \{(x_1, y_1), \ldots, (x_N, y_N)\}$. Give an expression for the expected complete-data log-likelihood $E[\log p(u, v, x, y \mid \boldsymbol{\theta})]$, where the expectation is with respect to some distribution on the unobserved class labels u and v. Express the M-step of the EM algorithm for estimating $\boldsymbol{\theta} = (\boldsymbol{\pi}, \boldsymbol{\sigma}, \boldsymbol{\varphi})$ in terms of the expected values of the sufficient statistics you defined in part (a). You'll develop formulae for calculating these expectations (the E-step of EM) in the next subsections.*

c) *Give the formulae for the conditional probability distribution $p(u_i, v_i \mid x, y, \boldsymbol{\theta})$ of the pair of class labels (u_i, v_i), as well as the marginals $p(u_i \mid x, y, \boldsymbol{\theta})$ and $p(v_i \mid x, y, \boldsymbol{\theta})$, given observations x, y and model parameters $\boldsymbol{\theta} = (\boldsymbol{\pi}, \boldsymbol{\sigma}, \boldsymbol{\varphi})$.*

d) *Give the formulae for each of the expectations you used in the M-step of the EM algorithm (part (b)) in terms of $p(u_i, v_i \mid x, y, \boldsymbol{\theta})$, and the data $D = \{(x_1, y_1), \ldots, (x_n, y_n)\}$.*

In the final problem, we explore the use of principal components analysis (PCA) for face recognition (a so-called *eigenface* model). Download the Matlab data file `faces.mat`, which contains cropped, normalized 51×43 images of 34 individuals:

`/course/cs195f/asgn/faces`

Each individual has been photographed with two facial expressions, "neutral" and "smiling". The pixels of each persons image have been stored as columns of the `neutralFaces` and `smileFaces` matrices. To convert back and forth between image and vector representations, use the reshape command:

```
faceImage  = reshape(smileFaces(:,index), 51, 43);
faceVector = reshape(faceImage, 51*43, 1);
```

In the following question, we explore whether the dimensionality reduction provided by PCA can effectively reduce the computational cost of nearest neighbor classifiers.

Question 3

a) To (approximately) account for illumination variations, independently normalize each of the face vectors so that averaging across all pixels, it has zero mean and unit variance.

b) Assume that we have a database of neutral face images, and want to determine the identity of the smiling individuals. For each of the 34 smiling faces, measure the L_2 norm of the difference between their normalized appearance vector and each of the 34 neutral faces. Classify each smiling face according to its nearest neighbor. What percentage of the smiling faces are correctly recognized?

c) As discussed in Sec. 20.3.4.3 of Murphy's text, a PCA decomposition can be efficiently computed via a singular value decomposition (SVD). Using the Matlab `svd` *command, determine the principal components of the set of normalized neutral faces (do* not *include the smiling faces). Be sure to subtract the mean face before performing your PCA. Plot the mean face and the first three principal components (the eigenfaces).* Hint: To avoid excessive memory usage when calculating the SVD, use Matlab's "economy size" option.

d) Determine the number of principal components required to model 90% of the total variance in the neutral face set. Project each of the neutral and smiling faces onto the corresponding eigenfaces. Use the coefficients of these projections to classify each smiling face according to its nearest neighbor in the PCA space. Compute the percentage of correctly recognized faces, and compare to part (b).

e) Repeat part (d) using the numbers of principal components required to model 80%, 70%, 60%, and 50% of the total neutral face variance. Plot the corresponding recognition rates.