Introduction to Machine Learning, Fall 2013

Problem Set 6: Hidden Markov Models and PCA

Due: Thursday, December 5, 2013 by 11am (Problem 2 requires an electronic submission to Chen-Chien. You can either submit Problem 1 in class or as part of your electronic submission)

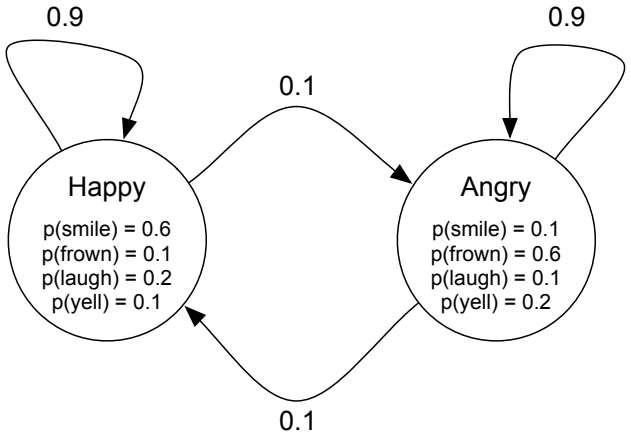
Important: *See problem set policy on the course web site.* You must show **all** of your work and be rigorous in your writeups to obtain full credit.

---

1. Amy lives a simple life. Some days she is Angry and some days she is Happy. But she hides her emotional state, and so all we can observe is whether she smiles, frowns, laughs, or yells. Amy's best friend is utterly confused about whether Amy is actually happy or angry and decides to model her emotional state using a hidden Markov model.

   Let $X_d \in \{\text{Happy, Angry}\}$ denote Amy's emotional state on day d, and let $Y_d \in \{\text{smile, frown, laugh, yell}\}$ denote the observation made about Amy on day d. **Assume that on day 1 Amy is in the Happy state**, i.e. $X_1 = \text{Happy}$. Furthermore, assume that Amy transitions between states exactly once per day (staying in the same state is an option) according to the following distribution: $p(X_{d+1} = \text{Happy} \mid X_d = \text{Angry}) = 0.1$, $p(X_{d+1} = \text{Angry} \mid X_d = \text{Happy}) = 0.1$, $p(X_{d+1} = \text{Angry} \mid X_d = \text{Angry}) = 0.9$, and $p(X_{d+1} = \text{Happy} \mid X_d = \text{Happy}) = 0.9$.

   The observation distribution for Amy's Happy state is given by $p(Y_d = \text{smile} \mid X_d = \text{Happy}) = 0.6, p(Y_d = \text{frown} \mid X_d = \text{Happy}) = 0.1, p(Y_d = \text{laugh} \mid X_d = \text{Happy}) = 0.2$, and $p(Y_d = \text{yell} \mid X_d = \text{Happy}) = 0.1$. The observation distribution for Amy's Angry state is $p(Y_d = \text{smile} \mid X_d = \text{Angry}) = 0.1, p(Y_d = \text{frown} \mid X_d = \text{Angry}) = 0.6, p(Y_d = \text{laugh} \mid X_d = \text{Angry}) = 0.1$, and $p(Y_d = \text{yell} \mid X_d = \text{Angry}) = 0.2$. **All of this is summarized in the following figure**:



   Each question below is worth 5 points. Be sure to show all of your work!

   (a) What is $p(X_2 = \text{Happy})$?

   (b) What is $p(Y_2 = \text{frown})$?

   (c) What is $p(X_2 = \text{Happy} \mid Y_2 = \text{frown})$?

   (d) What is $p(Y_{100} = \text{yell})$?

   (e) Assume that $Y_1 = Y_2 = Y_3 = Y_4 = Y_5 = \text{frown}$. What is the most likely sequence of the states? That is, compute the MAP assignment $\arg\max_{x_1,\ldots,x_5} p(X_1 = x_1, \ldots, X_5 = x_5 \mid Y_1 = Y_2 = Y_3 = Y_4 = Y_5 = \text{frown})$.

2. In this section, we will explore some properties of Principal Components Analysis (PCA). You may use the programming language of your choice, but we strongly encourage you to use Matlab/Octave or Python.

   Face Data: We use the Olivetti face dataset.[1] The data contains 400 face images of size 64×64. In `faces.csv`, each line represents a face image. The first 64 values represent the first column of the image, and the next 64 values represent the second column and so on.

   We provide some sample code written in MATLAB/Octave and Python (using the SciPy and Matplotlib packages). You can find the sample code in `ps6.py`, `ps6.m`, and `visualize.m`. `visualize.m` is the function for visualizing principal components in Matlab; a similar function for Python is in `ps6.py`. The sample code shows how to load the data and display the first image.

   You can find the line "Your code starts from here" in the sample code, below which you may start writing your code. Please include all your figures in the write up, and send your code to `ccw352@nyu.edu`.

   (a) Display a face image chosen randomly from the 400 images.

   (b) Compute and display the mean of the faces.

   (c) Subtract the mean from the face images and get the centralized data matrix X, which is of dimension $m = 400$ (number of images) by $p = 4096$ (number of pixels).

   (d) Compute the singular value decomposition (SVD) of the centralized face data such that $X = USV^T$. The columns of V are the principal components that define the new basis. Letting $W = US$, we notice that $W = USV^TV = XV$, where we used the fact that $V^TV = I$ since V is an orthonormal matrix. Thus, each row of W consists of the dimensionality-reduced data, i.e. the coefficients expressing the data using the principal component coordinate system.

   (e) Display images of the first 10 principal components, i.e. the first 10 columns in V. You can do this by reshaping the principal components to 64×64 matrices, and plot them by `imagesc` in Matlab or `matplotlib.pyplot.imshow` in Python.

   (f) Let's now use the first two principal components to project the data into $\mathbb{R}^2$ to try to better visualize it. The i'th face will be located at (W_{i1}, W_{i2}) in the new coordinate system. Randomly choose 30 faces and visualize the data. We provide the plotting functions `visualize` in Python and `visualize.m` in Matlab for this purpose.

   (g) Recall that the principal component can also be found by decomposing the sample covariance matrix, that is, $\Lambda = \frac{1}{m}X^TX = \frac{1}{m}VSU^TUSV^T = V\frac{S^2}{m}V^T$. The relation between the singular values σ_i given along the diagonal of S (found using SVD on X) and the eigenvalues λ_i (of the sample covariance matrix Λ) is thus given by $\sigma_i^2/m = \lambda_i$.

   The total variance in a data set is defined as the sum of the variances of the individual components. Assuming that the data has aleady had its mean subtracted, we have that the total variance of the data in the original basis is given by

$$\text{TotalVar} = \sum_{j=1}^{p} \text{Var}(X_{\cdot j}) = \sum_{j=1}^{p} \frac{1}{m} \sum_{i=1}^{m} X_{ij}^2 = \frac{1}{m} \sum_{i,j} X_{ij}^2 = \sum_j \lambda_j,$$

   where in the last step we used the fact that[2]

$$\sum_{i,j} X_{ij}^2 = \text{trace}(X^TX) = \text{trace}(VS^2V^T) = \text{trace}(V^TVS^2) = \text{trace}(S^2) = m\sum_j \lambda_j.$$

[1] The original data set can be downloaded at `http://www.cs.nyu.edu/~roweis/data.html`.

[2] $\text{trace}(M) := \sum_i M_{ii}$ is the sum of the diagonal entries of a square matrix, which has the property that $\text{trace}(AB) = \text{trace}(BA)$. $(X^TX)_{jj} = \sum_i (X^T)_{ji}(X)_{ij} = \sum_i X_{ij}^2$, and thus $\text{trace}(X^TX) = \sum_j (X^TX)_{jj} = \sum_{ij} X_{ij}^2$.

We next notice that the variance of the projected data given by the k'th principal component v_k (i.e. the k'th column of V) is given by the corresponding eigenvalue λ_k:

$$\frac{1}{m}(Xv_k)^T(Xv_k) = \frac{1}{m}v_k^T X^T X v_k = \frac{1}{m}v_k^T(X^T X v_k) = \frac{1}{m}(v_k^T m\lambda_k v_k) = \lambda_k.$$

Thus, we can calculate the proportion of variance explained by the i'th principal component as $\lambda_i / \sum_j \lambda_j = \sigma_i^2 / \sum_j \sigma_j^2$.
Plot the proportion of variance explained by the first 10 principal components.

(h) Randomly choose a face, reconstruct it using 5, 10, 25, 50, 100, 200, 300, 399 principal components, and show the reconstructed images.

(i) ***Extra credit.*** Reconstruct all the faces using $k = 5, 10, 25, 50, 100, 200, 300, 399$ principal components. You can reconstruct all the faces at once using the formula

$$R_k = W_k V_k^T + \text{mean face},$$

where k is the number of principal components to use and the notation M_k refers to the submatrix of M obtained by using the first k columns of M. The reconstruction error using the first k principal components is defined as $\sum_{i,j}(\tilde{X}_{ij} - [R_k]_{ij})^2$, where $\tilde{X}$ is the uncentralized data matrix.
For $k = 1, 2, \ldots, p$ compute the sum of the last $p - k$ squares of singular values, i.e. $E_k = \sum_{j=k+1}^{p} \sigma_j^2$.
Plot (I) the reconstruction error as a function of k and (II) E_k as a function of k.