Problem Set 4

MAS 622J/1.126J: Pattern Recognition and Analysis

Due: Wednesday, October 15th, 2008
Second submission: Friday, October 17th, 2008

[Note: All instructions to plot data or write a program should be carried out using either Python accompanied by the `matplotlib` package or Matlab. Feel free to use either or both, but in order to maintain a reasonable level of consistency and simplicity we ask that you do not use other software tools. Please provide a printed copy of any code used for this problem set.]

Problem 1: Fisher Linear Discriminant

In this problem, we will do some practice on Fisher linear discriminant, trying to obtain a better understanding of component analysis. We consider the problem of projecting Gaussian distributed data from 2 dimensions onto a line. All data samples are collected from two classes ω_1 and ω_2. The projection seeks the direction that is most efficient for discrimination.

As a reminder, the Fisher projection direction is found by $\mathbf{w} = S_W^{-1}(\mathbf{m}_1 - \mathbf{m}_2)$, where $S_W = S_1 + S_2$ and $S_i = \sum_{\mathbf{x}\in\mathcal{D}_i}(\mathbf{x} - \mathbf{m}_i)(\mathbf{x} - \mathbf{m}_i)^t$. Note that the scale of $\mathbf{w}$ does NOT matter at all. Multiplying $\mathbf{w}$ with a scalar doesn't have any impact on the discriminability. However, it's a good practice to normalize it after you obtain it.

a. Let's do a Matlab exercise. Let $p_{\mathbf{x}}(\mathbf{x}|w_i)(i = 1, 2)$ be both Gaussian distributed in a 2D space. $p_{\mathbf{x}}(\mathbf{x}|w_1) = \mathcal{N}([4, 3]^t; \text{diag}([2, 1]))$. For $p_{\mathbf{x}}(\mathbf{x}|w_2)$, we fix Σ_2 as $\text{diag}([1, 1])$, and let $\boldsymbol{\mu_2}$ be $[2, 3]^t$, $[3, 4.7321]^t$, $[2.2679, 4]^t$ in turn. For each of the three cases, you are asked to generate 500 random samples for both classes using the 'randn' Matlab function. Based on your generated data, find out the first PCA component and the Fisher projection direction. Plot these directions on the same graph, label them clearly, and comment on how they vary when we move the mean. Can you explain from the graph the key difference between PCA component and Fisher projection direction? Don't use any built-in Matlab function to find out these directions.

b. This time we fix the mean $\boldsymbol{\mu_2}$ as $[2, 5]^t$, and change the covariance matrix Σ_2 each time to be diag([1, 1]), diag([2, 2]), and diag([3, 3]). Repeat the problem in (a), and comment on how the directions change with respect to the variance change.

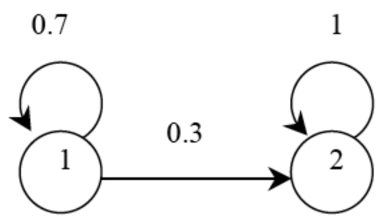
c. Let's consider a more general case first for linear discrimination. Let $p_{\mathbf{x}}(\mathbf{x}|w_1) = \mathcal{N}([\mu_{1x}, \mu_{1y}]^t; \text{diag}([\sigma_{1x}^2, \sigma_{1y}^2]))$, $p_{\mathbf{x}}(\mathbf{x}|w_2) = \mathcal{N}([\mu_{2x}, \mu_{2y}]^t; \text{diag}([\sigma_{2x}^2, \sigma_{2y}^2]))$. After the linear projection $\mathbf{y} = \mathbf{w}^t\mathbf{x}$ for any direction $\mathbf{w}$, what is the decision rule on $\mathbf{y}$ that minimizes the probability of misclassification? (Hint: for a Gaussian random vector $\mathbf{x}$, any of its linear transformation including $\mathbf{w}^t\mathbf{x}$ is still Gaussian).

d. Let $p_{\mathbf{x}}(\mathbf{x}|w_1) = \mathcal{N}([4, 2]^t; \text{diag}([9, 1]))$, $p_{\mathbf{x}}(\mathbf{x}|w_2) = \mathcal{N}([2, 5]^t; \text{diag}([2, 3]))$. Again, generate 500 samples for each class and then project all samples to the following four directions: (i) Fisher linear discriminant direction (ii) The eigenvector of the all-sample covariance matrix corresponding to the larger eigenvalue (iii) The eigenvector of the all-sample covariance matrix corresponding to the smaller eigenvalue (iv) the line connecting the two means. For each projection, compute the probability of error when the minimum probability of misclassification decision rule is applied (results in (c) can be useful here). Which direction of projection gives the smallest probability of error?

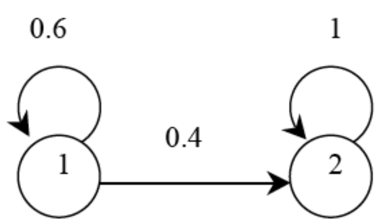
Problem 2: Hidden Markov Models

We have two 2-state Hidden Markov Models, where both states have two possible output symbols A and B.

Model 1



Model 2



The output probabilities are given by:

$$\begin{array}{lll} \text{Model1} & : & b_1(A) = 0.8 \quad b_1(B) = 0.2 \quad b_2(A) = 0.3 \quad b_2(B) = 0.7 \\ \text{Model2} & : & b_1(A) = 0.3 \quad b_1(B) = 0.7 \quad b_2(A) = 0.1 \quad b_2(B) = 0.9 \end{array}$$

The initial probabilities are given by:

$$\begin{array}{lll} \text{Model1} & : & \pi_1 = 0.75 \quad \pi_2 = 0.25 \\ \text{Model2} & : & \pi_1 = 0.4 \quad \pi_2 = 0.6 \end{array}$$

a. For Model 1, what is the probability of an observation sequence $\{BAB\}$ is generated?

b. For Model 1, given that this HMM produced an observation sequence $\{BAB\}$, what is the most likely sequence of hidden states that led to those observations?

c. Which model is more likely to produce the observation sequence $\{BAB\}$?

Problem 3: Baum-Welch Algorithm and Discrete HMMs

Download the datasets from the course webpage. The datasets consist of training and testing sequences belonging to two classes. We assume the two HMMs for the two classes have the same configuration, i.e. the same number of states, zero transition probabilities and the number of output states.

Implement the Baum-Welch algorithm for training a discrete HMM. Train HMMs with one, three, and five states with transition probabilities in a strictly left-to-right configuration (see the figure below for a two-state HMM in left-to-right configuration). The visible output has four possible states 0, 1, 2 or 3. and

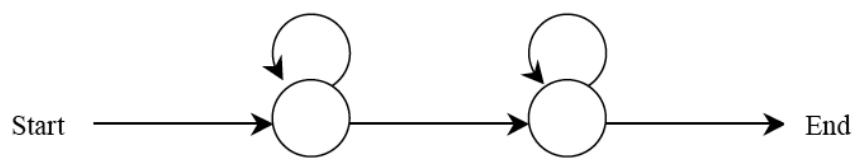


Figure 1: HMM in a left-to-right configuration

Repeat the following steps for each of the three HMM configurations with one, three, and five states:

a. Train two HMMs, one for each class of data. State very clearly the threshold you are using and the maximum number of iterations. List the output probabilities and state transition probabilities of each HMM.

b. Implement the Viterbi algorithm to decode each test sequence using both HMMs. Show the log probability of each test sequence using each HMM.

c. Compute the recognition accurary on the entire test set.

Include a complete listing of your source code.