10-701 Machine Learning, Spring 2011: Homework 5 Solution

April 25, 2011

Instructions There are three questions on this assignment. Please submit your writeup as two separate sets of pages according to questions, with your name and userid on each set.

1 Hidden Markov Models [Xi Chen, 30 points]

Andrew lives a simple life. Some days he is Angry and some days he is Happy. But he hides his emotional state, and so all you can observe is whether he smiles, frowns, laughs, or yells. We start on day 1 in the Happy state and there is one transition per day.

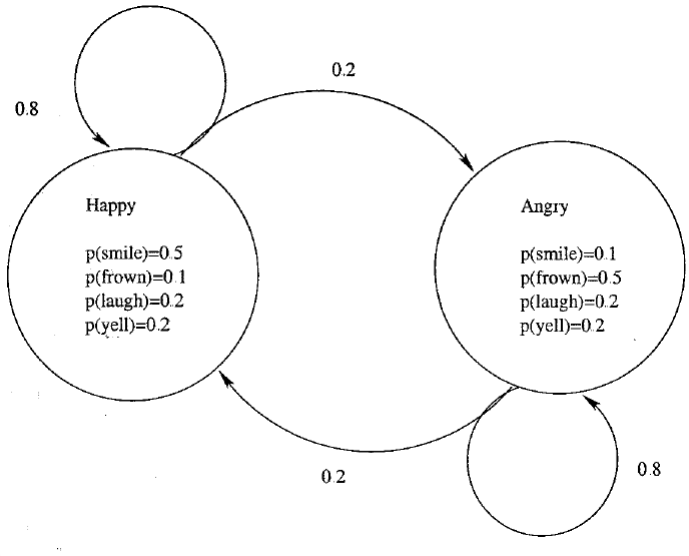

Figure 1: Transition Model for Andrew

We define;

- q_t: state on day t
- O_t: observation on day t

(a) What is $\Pr(q_2 = Happy)$? [5pt]

(b) What is $\Pr(O_2 = frown)$? [5pt]

(c) What is $\Pr(q_2 = Happy | O_2 = frown)$? [5pt]

(d) What is $\Pr(O_{100} = yell)$? [5pt]

(e) Assume that $O_1 = O_2 = O_3 = O_4 = O_5 = frown$. What is the most likely sequence of the states. [10pt]

★ **SOLUTION:**

(a)

$$\Pr(q_2 = Happy) = 0.8$$

(b)

$$\Pr(O_2 = frown) = \frac{8}{10}\frac{1}{10} + \frac{2}{10}\frac{1}{2} = \frac{18}{100} = 0.18$$

(c)

$$\Pr(q_2 = Happy|O_2 = frown) = \frac{\Pr(O_2 = frown|q_2 = Happy)\Pr(q_2 = Happy)}{\Pr(O_2 = frown)} = \frac{\frac{1}{10}\frac{8}{10}}{\frac{18}{100}} = \frac{4}{9}$$

(d)

$$\begin{aligned}\Pr(O_{100} = yell) &= \Pr(O_{100} = yell|q_{100} = Happy)\Pr(q_{100} = Happy) + \Pr(O_{100} = yell|q_{100} = Angry)\Pr(q_{100} = Angry \\ &= 0.2\end{aligned}$$

(e) Happy, Angry, Angry, Angry, Angry

2 Dimension Reduction [Yi Zhang, 35 points]

2.1 Principal components analysis vs. Fisher's linear discriminant

Principal components analysis (PCA) reduces the dimensionality of the data by finding projection direction(s) that *minimizes the squared errors in reconstructing the original data* or equivalently *maximizes the variance of the projected data*. On the other hand, Fisher's linear discriminant is a supervised dimension reduction method, which, given labels of the data, finds the projection direction that *maximizes the between-class variance relative to the within-class variance of the projected data*.

[10 points] In the following Figure 2, **draw** the first principal component direction in the left figure, and the first Fisher's linear discriminant direction in the right figure. Note: for PCA, ignore the fact that points are labeled (as round, diamond or square) since PCA does not use label information. For linear discriminant, consider round points as the positive class, and both diamond and square points as the negative class (since in the course lecture we only discuss the two-class case).

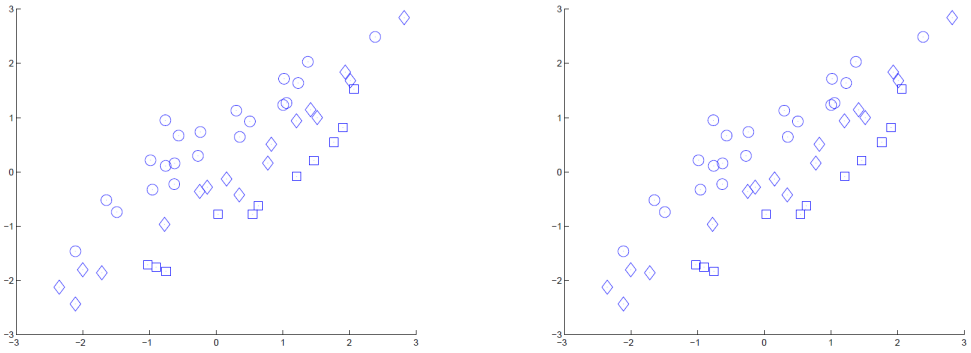

Figure 2: Draw the first principal component and linear discriminant component, respectively

★ **SOLUTION:** The PCA and LDA directions are shown in the following figure.

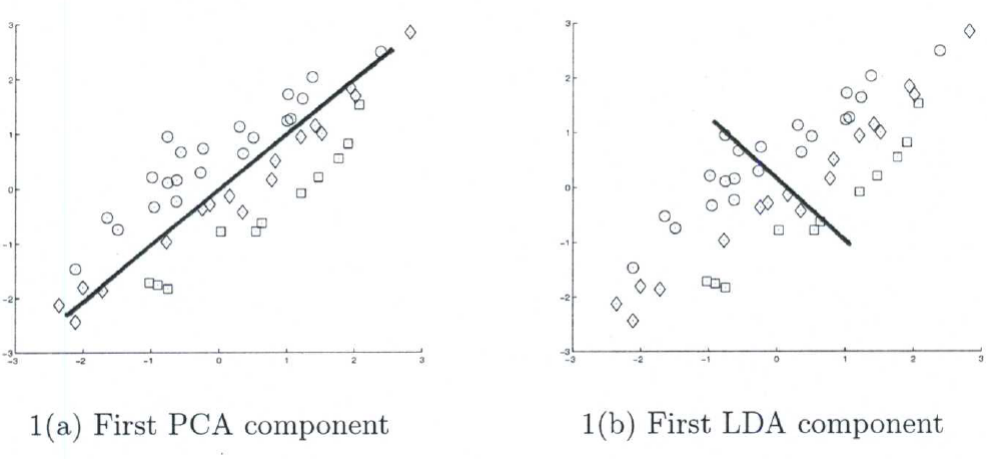

Figure 3: The first principal component and linear discriminant component, respectively

2.2 Canonical correlation analysis

Canonical correlation analysis (CCA) handles the situation that each data point (i.e., each object) has two representations (i.e., two sets of features), e.g., a web page can be represented by the text on that page, and can also be represented by other pages linked to that page. Now suppose each data point has two representations $\mathbf{x}$ and $\mathbf{y}$, each of which is a 2-dimensional feature vector (i.e., $\mathbf{x} = [x_1, x_2]^T$ and $\mathbf{y} = [y_1, y_2]^T$). Given a set of data points, CCA finds a pair of projection directions $(\mathbf{u}, \mathbf{v})$ to maximize the sample correlation $\hat{corr}(\mathbf{u}^T\mathbf{x})(\mathbf{v}^T\mathbf{y})$ along the directions $\mathbf{u}$ and $\mathbf{v}$. In other words, after we project one representation of data points onto $\mathbf{u}$ and the other representation of data points onto $\mathbf{v}$, the two *projected* representations $\mathbf{u}^T\mathbf{x}$ and $\mathbf{v}^T\mathbf{y}$ should be maximally correlated (intuitively, data points with large values in one projected direction should also have large values in the other projected direction).

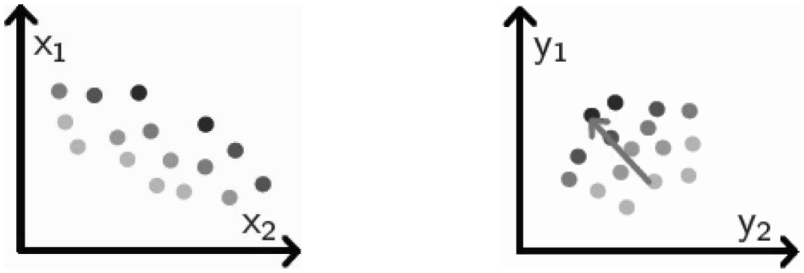

Figure 4: Draw the CCA projection direction in the left figure

[6 points] Now we can see data points shown in the Figure 4, where each data point has two representations $\mathbf{x} = [x_1, x_2]^T$ and $\mathbf{y} = [y_1, y_2]^T$. Note that data are paired: each point in the left figure corresponds to a specific point in the right figure and vice versa, because these two points are two representations of the same object. Different objects are shown in different gray scales in the two figures (so you should be able to approximately figure out how points are paired). In the right figure we've given one CCA projection direction $\mathbf{v}$, **draw** the other CCA projection direction $\mathbf{u}$ in the left figure.

★ **SOLUTION:** The CCA projection direction is shown in the following figure (on the left).

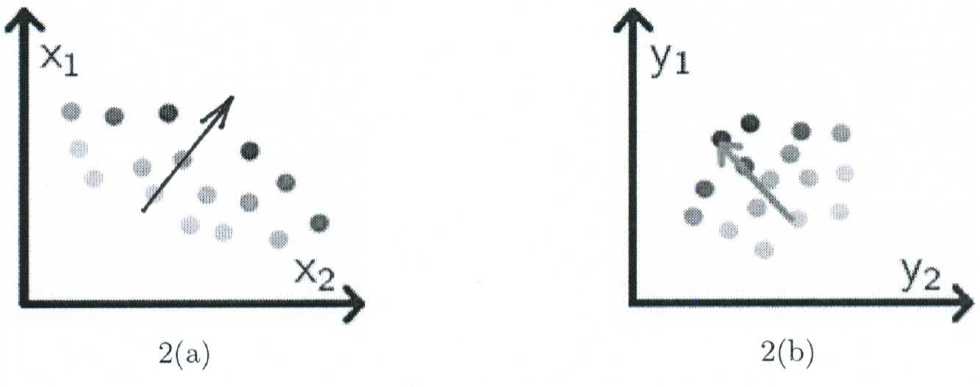

Figure 5: The first principal component and linear discriminant component, respectively

2.3 More Principal Components Analysis

Consider 3 data points in the 2-d space: (-1, -1), (0,0), (1,1).

[6 points] What is the first principal component (write down the actual vector)?

★ **SOLUTION:** The first principal component is $\mathbf{v} = [\frac{\sqrt{2}}{2}, \frac{\sqrt{2}}{2}]^T$ (you shouldn't really need to solve any SVD or eigenproblem to see this). Note that the principal component should be normalized to have unit length. (The negation $\mathbf{v} = [-\frac{\sqrt{2}}{2}, -\frac{\sqrt{2}}{2}]^T$ is also correct.)

[7 points] If we project the original data points into the 1-d subspace by the principal component you choose, what are their coordinates in the 1-d subspace? And what is the variance of the projected data?

★ **SOLUTION:** The coordinates of three points after projection should be $z_1 = \mathbf{x}_1^T\mathbf{v} = [-1, -1][\frac{\sqrt{2}}{2}, \frac{\sqrt{2}}{2}]^T = -\sqrt{2},\ z_2 = \mathbf{x}_2^T\mathbf{v} = 0,\ z_3 = \mathbf{x}_3^T\mathbf{v} = \sqrt{2}$. Note that the sample mean is 0, and thus the variance is $\frac{1}{3}\sum_{i=1}^{3}(z_i - 0)^2 = \frac{4}{3}$ (or you can also choose to use the unbiased estimation $\frac{1}{3-1}\sum_{i=1}^{3}(z_i - 0)^2 = 2$).

[6 points] For the projected data you just obtained above, now if we represent them in the original 2-d space and consider them as the reconstruction of the original data points, what is the reconstruction error?

★ **SOLUTION:** The reconstruction error is 0, since all three points are perfectly located on the direction of the first principal component. Or, you can actually calculate the reconstruction: $\hat{\mathbf{x}}_1 = z_1 \cdot \mathbf{v} = -\sqrt{2} \cdot [\frac{\sqrt{2}}{2}, \frac{\sqrt{2}}{2}]^T = [-1, -1]^T,\ \hat{\mathbf{x}}_2 = [0, 0]^T,\ \hat{\mathbf{x}}_3 = [1, 1]^T$, which are exactly $\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3$.

3 Neural Nets [Carl, 35 points]

The neural networks shown in class used logistic units: that is, for a given unit U, if A is the vector of activations of units that send their output to U, and W is the weight vector corresponding to these outputs, then the activation of U will be $(1 + exp(W^T A))^{-1}$. However, activation functions could be anything. In this exercise we will explore some others. Consider the following neural network, consisting of two input units, a single hidden layer containing two units, and one output unit:

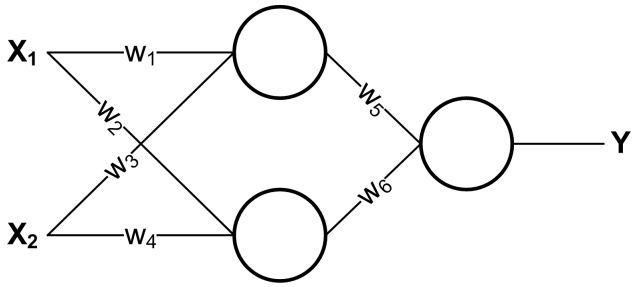

Figure 6: Neural network for question 3

1. **[9 points]** Say that the network is using linear units: that is, defining W and and A as above, the output of a unit is $C * W^T A$ for some fixed constant C. Let the weight values w_i be fixed. Re-design the neural network to compute the same function without using any hidden units. Express the new weights in terms of the old weights and the constant C.

   ★ **SOLUTION:** Connect the input for X_1 to the output unit with a weight $C * (w_5 * w_1 + w_6 * w_2)$, and connect the input for X_2 to the output unit with weight $C(w_5 * w_3 + w_6 * w_4)$. Then the output unit can use the same activation function it used originally.

2. **[4 points]** Is it always possible to express a neural network made up of only linear units without a hidden layer? Give a one-sentence justification.

   ★ **SOLUTION:** This is true. Each layer can be thought of as performing a matrix multiply to find its representation given the representation on the layer that it receives input from. Thus the entire netowrk just performs a chain of matrix multiplies, and therefore we can simply multiply the matrices together to find the weights to represent the function with a single layer.

3. **[9 points]** Another common activation function is a theshold, where the activation is $t(W^T A)$ where $t(x)$ is 1 if $x > 0$ and 0 otherwise. Let the hidden units use sigmoid activation functions and let the output unit use a threshold activation function. Find weights which cause this network to compute the XOR of X_1 and X_2 for binary-valued X_1 and X_2. Keep in mind that there is no bias term for these units.

   ★ **SOLUTION:** One solution: $w_1 = w_3 = -10$,$w_2 = w_4 = -1$,$w_5 = 5$, and $w_6 = -6$. The intuition here is that we can decompose `A XOR B` into `(A OR B) AND NOT (A AND B)`. We make the upper hidden unit behave like an OR by making it saturate when either of the input units are 1. It isn't possible to make a hidden unit that behaves exactly like AND, but we can at least make the lower hidden unit continue to increase in activation after the upper one has saturated.

4. **[4 points]** Why are threshold activation functions generally inconvenient for training? Explain in one sentence.

★ **SOLUTION:** Although many people wrote that the non-differentiability of the threshold function makes computing gradients impossible, this isn't exactly correct. In fact, it is quite common to perform gradient descent on the absolute value function for the sake of finding sparse representations (google the lasso if you're interested). A bit harder to deal with is the fact that the threshold function has a discontinuity; there are optimization techniques to handle some forms of discontinuity, but I don't know of one that would work well with a threshold. Probably the worst problem is that the gradient is zero almost everywhere; therefore, all of the weight updates computed by backpropagation would be zero.

5. [**9 points**] Using the same architecture as in figure 6, choose an activation function for each unit in the network which will cause this network to learn the same function that logistic regression would learn. Each unit must use a logistic, linear, or threshold activation function, with no constraints on the weights. You may assume either gradient-descent learning, or you may assume that there is an oracle which can set the weights optimally in terms of squared-error.

★ **SOLUTION:** Most of the class correctly assumed what was implicitly assumed in the problem: that the goal was to create a network that simulated logistic regression with no bias term. A handful of people tried to create a network with a bias, for example, using a threshold unit, but it isn't possible to do this without allowing the network to represent nonlinear decision surfaces. However, I tried not to take off points for this if your argument was well-reasoned.

The intended solution was to create a hidden layer with linear units and a logistic output unit. As long as the weight matrix between the first and second layers is non-singular (i.e. it is possible to reconstruct the input layer from the hidden layers), it will always be possible for the final layer to learn a pair of weights which will achieve the optimal squared-error, since the entire optimization is linear up until the final sigmoid function is applied on the last hidden layer.

Due to a technicality, though, this isn't quite correct. The objective functions for logistic regression and neural networks are different: neural networks optimize squared error, whereas logistic regression optimizes probabilities. In general, the optima of these two objective functions will not be exactly the same, although they will usually be quite similar.

Despite its flaws, nearly the entire class understood the spirit of this question. Thus I still graded it, though I tried to be forgiving of misinterpretations, and therefore rarely took off points.