CS 475 Machine Learning: Homework 6
Structured Prediction
Due: Wednesday December 7, 2016, 11:59pm

Michael Mow
mmow1

1 Analytical (25 points)

1. (12 points) One of the uses of PCA is to create better data representations for classification. Consider two different classes in a binary classification task. The first class has points generated using a Gaussian with $\mu_1 = \{1, -1\}$ and covariance

$$\mathbf{\Sigma}_1 = \begin{pmatrix} 1 & 0 \\ 0 & .001 \end{pmatrix}$$

The second class has points generated using a Gaussian with $\mu_2 = \{1, 1\}$ and covariance

$$\mathbf{\Sigma}_2 = \begin{pmatrix} 2 & 0 \\ 0 & .002 \end{pmatrix}$$

a. How would a linear classifier perform when trained to classify between these two classes in the given representation? Why?

b. Suppose PCA is run on this two dimensional data to produce a one dimensional representation. Describe the principal component that PCA would select.

c. How would a linear classifier perform when trained to classify between these two classes in the one dimensional PCA representation? Why?

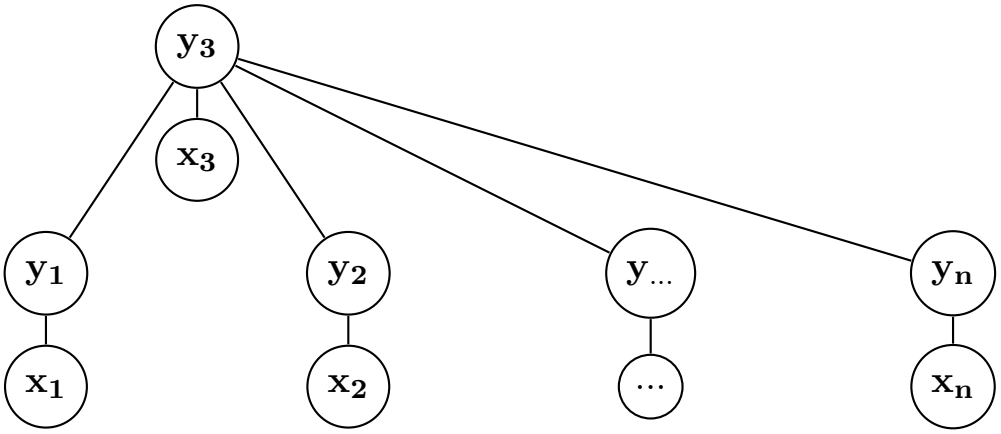
d. Suppose instead you train a neural network on this dataset. The neural network has a single hidden layer with a single hidden node, and a single output node corresponding to the label. Compare the performance of this neural network to a linear classifier trained on a representation of this data learned by PCA.

a. A linear classifier would perform very well since the data can easily be separated by a linear classifier through the x-z plane. Since the means of the two classes are separated by 2 units in the y direction with very small y-variances (0.001 and 0.002), the data will be separated well.

b. PCA would choose the component based on the covariance matrix with the highest variance. This would be the x component reducing the first class to a mean of 1 and variance of 2 and the second class to a mean of 1 and a variance of 2.

c. A linear classifier would perform very poorly on this one dimensional PCA representation. This is because the two classes have the same mean with only 1 having a higher variance. Thus a linear classifier will not even be able to separate the means of the two classes.

d. This neural network would perform much better than the linear classifier trained on the 1D data learned from PCA. This is because the neural network will learn that the important information is in the second dimension and will thus make the weights from the input dimensions to the hidden node appropriate so that the second dimension becomes the main determinant of the output label.



2. (13 points) Consider the graphical model shown above. In this model, $\mathbf{x}$ is a sequence of observations for which we want to output a prediction $\mathbf{y}$, which itself is a sequence, where the size of $\mathbf{y}$ is the same as $\mathbf{x}$. Unlike sequence models we discussed in class, this model has a tree structure over the hidden nodes. Assume that the potential functions have a log-linear form: $\psi(Z) = \exp\{\sum_i \theta_i f_i(Z)\}$, where Z is the set of nodes that are arguments to the potential function (i.e. some combination of nodes in $\mathbf{x}$ and $\mathbf{y}$,) θ are the parameters of the potential functions and f_i is a feature function.

a. Write the log likelihood for this model of a single instance $\mathbf{x}$: $p(\mathbf{y}, \mathbf{x})$.

b. Write the conditional log likelihood for this model of a single instance $\mathbf{x}$: $p(\mathbf{y}|\mathbf{x})$.

c. Assume that each variable y_i can take one of k possible states, and variable x_i can take one of k' possible states, where k' is very large. Describe the computational challenges of modeling $p(\mathbf{y}, \mathbf{x})$ vs $p(\mathbf{y}|\mathbf{x})$.

d. Propose an efficient algorithm for making a prediction for $\mathbf{y}$ given $\mathbf{x}$ and θ.

a. $log(p(y,x)) = \sum_C log(\psi(C)) - log(\sum_{x,y} \Pi_C \psi(C))$ where C is the set of all the 2-cliques.

b. $log(p(y|x)) = \sum_C log(\psi(C)) - log(\sum_y \Pi_C \psi(C))$ where C is the set of all the 2-cliques.

c. Modeling $p(y|x)$ will be much easier computationally than $p(y,x)$ since $p(y,x)$ includes a sum over all x and x has k' states where k' is very large. $p(y|x)$ does not include this sum over x

d. An efficient algorithm for making a prediction for $\mathbf{y}$ given $\mathbf{x}$ and θ would be the max sum algorithm since we are trying to find the highest global probability configuration.