

Problem Set 3

MAS 622J/1.126J: Pattern Recognition and Analysis

Due Monday, 6 October 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. Unless otherwise specified, please copy and paste all your codes in your solution document.]

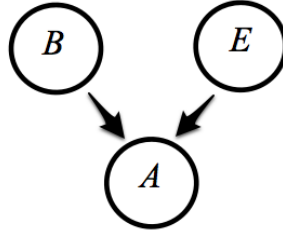
Problem 1: Discriminability and ROC

Please download the datasets (`dataset1.txt`, `dataset2.txt`, `dataset3.txt`, `dataset4.txt`) from the course website. Each dataset includes 1-D data samples from two classes w_1 and w_2 . The first column and the second column of each dataset specify 1000 samples from class w_1 and 1000 samples from class w_2 , respectively.

- For each dataset, compute the discriminability $d' = \frac{|\mu_2 - \mu_1|}{\sqrt{\sigma_1^2 + \sigma_2^2}}$ where μ_1 and σ_1 are the mean and standard deviation of the distribution of class w_1 , and μ_2 and σ_2 are the mean and standard deviation of the distribution of class w_2 .
- Now we compute the ROC curve for each dataset. Please plot four ROC curves in the same figure. To do this, we approximate $\mathbf{P}_{\mathbf{TP}} = \mathbf{P}(\mathbf{x} > \mathbf{x}^* | \mathbf{x} \in w_2)$ by $\mathbf{N}(\mathbf{x} > \mathbf{x}^* | \mathbf{x} \in w_2)/1000$, and $\mathbf{P}_{\mathbf{FP}} = \mathbf{P}(\mathbf{x} > \mathbf{x}^* | \mathbf{x} \in w_1)$ by $\mathbf{N}(\mathbf{x} > \mathbf{x}^* | \mathbf{x} \in w_1)/1000$. Here, for $i = 1$ or 2 , $\mathbf{N}(\mathbf{x} > \mathbf{x}^* | \mathbf{x} \in w_i)$ is denotes the number of samples in class w_i whose value is greater than \mathbf{x}^* . Note that \mathbf{N} doesn't denote a normal distribution!
- For each dataset, plot the two *approximated* probability density functions. Note that the probability density function is the derivative of the cumulative density function. (Do NOT just approximate distributions by Gaussians and draw those approximated Gaussians in this problem.) *Hint*: Use the same method we use in (b) to get the cumulative distribution. This is called *Monte Carlo* method. Also, note that the derivative (i.e., probability density function $p(x)$) relates to the increment of the cumulative density function $P_X(x)$. That is, $p(x) = \Delta P_X(x) / \Delta x$.
- How does the discriminability relate to the ROC curve?

Problem 2: Conditional Dependence

Consider the following Bayesian Network where the three events are B (Burglary), E (Earthquake) and A (Alarm). Assume that the three nodes are binary nodes that can take on the value f (false) or t (true).



Here, B and E are marginally independent. That is, $P(B, E) = P(B)P(E)$ (Note that this relation implies $P(B|E) = P(B)$ or $P(E|B) = P(E)$.) Now we want to show B and E are conditionally dependent given A . In other words, $P(B, E|A) \neq P(B|A)P(E|A)$.

- Prove that the relation $P(B, E|A) \neq P(B|A)P(E|A)$ implies $P(B|E, A) \neq P(B|A)$. Note that by the symmetry between B and E , this also means $P(E|B, A) \neq P(E|A)$.
- First, we consider a very simple case. Suppose $A = B + E$ where the $+$ sign means “Logical OR.” This means that B and E are independent deterministic causes of A . Construct the CPT (conditional probability table) for $P(A | B, E)$.

B, E	$P(A = f B, E), P(A = t B, E)$
f, f	
t, f	
f, t	
t, t	

- For the obtained CPT above, prove that $P(B = t|A = t) = P(B = t)/P(A = t)$. Which is greater between $P(B = t|A = t)$ and $P(B = t)$? What is the meaning of this?
- For the obtained CPT above, prove that $P(A = t) = P(B = t) + P(E = t) - P(B = t)P(E = t)$. If $P(B = t)$ and $P(E = t)$ are small, what happens to $P(B = t|A = t)$, compared with $P(B = t)$?
- For the obtained CPT above, prove that $P(B = t|E = t, A = t) = P(B = t)$. Which is greater between $P(B = t|E = t, A = t)$ and $P(B = t|A = t)$? What is the meaning of this? *Hint:* Note that the observation of Earthquake ($E = t$) alone is enough to explain the cause of Alarm ($A = t$). This occurrence is called “Explaining Away.”
- We assume that $P(B = t) = 0.1$ and $P(E = t) = 0.01$. Calculate $P(A = t)$, $P(B = t|A = t)$, $P(B = t|E = t, A = t)$, $P(E = t|A = t)$, $P(E = t|B = t, A = t)$ by hand.

- g. Now install Kevin Murphy’s Bayes Net Toolbox for Matlab. You can download the toolbox and find the install and usage instructions here:

<http://bnt.sourceforge.net/>

We compute the above $P(A = t), P(B = t|A = t), P(B = t|E = t, A = t), P(E = t|A = t), P(E = t|B = t, A = t)$ using this toolbox. To help you use the toolbox, we provide you with the basic Matlab code for this problem.

Download the Matlab code “**conditional_dep.m**” from the course website. Fill in the core part in the Matlab code and compute $P(A = t), P(B = t|A = t), P(B = t|E = t, A = t), P(E = t|A = t), P(E = t|B = t, A = t)$. Are the computed values the same to what you’ve obtained by hand?

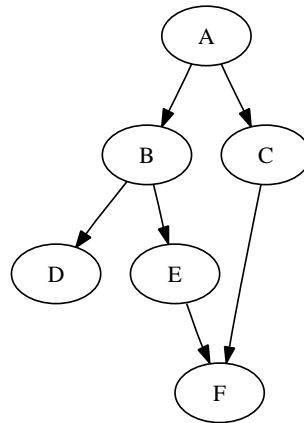
- h. Second, we consider a more complex case. Assume that $P(B = t) = 0.1, P(E = t) = 0.01$ and $P(A | B, E)$ is described by the following CPT.

B, E	$P(A = f B, E), P(A = t B, E)$	
f, f	0.99	0.01
t, f	0.05	0.95
f, t	0.01	0.99
t, t	0.001	0.999

Compute $P(A = t), P(B = t|A = t), P(B = t|E = t, A = t), P(E = t|A = t), P(E = t|B = t, A = t)$ using the toolbox.

Problem 3: Bayes Nets, Computational Complexity

Use the following Bayes belief net over categorical variables when answering the parts of this problem:



- a. Take advantage of the independence assumptions in the Bayes belief net in order to factor and simplify the following expression for $P(B = b)$:

$$P(B = b) = \sum_{a \in A} \sum_{c \in C} \sum_{d \in D} \sum_{e \in E} \sum_{f \in F} P(a, b, c, d, e, f)$$

- b. For this part assume that you have a computer that can perform multiplication and addition operations in one time step with memory operations taking no time. Assume also that the number of categories for each variable ($A, B, C, D, E,$ and F) is k .

We now want to consider the computational complexity “big oh” expression for the order of the original *unsimplified* expression for $P(B = b)$ and your *simplified* expression for $P(B = b)$ in part (a), in terms of the number of addition and multiplication operations required to compute the answer.

Note that the “big oh” notation (so called because it uses the symbol O) describes the limiting behavior of a function for very small or very large arguments, usually in terms of simpler functions. Find out more information about this notation from

http://en.wikipedia.org/wiki/Big_O_notation or DHS book’s appendix.

For example, in the original un-factored formula for $P(B = b)$, there are $k^5 - 1$ additions, $O(k^5)$, while there are no multiplies, $O(1)$. Also, the total amount of space required to store the entire joint distribution, $P(A, B, C, D, E, F)$, naively is $k^6 - 1$, $O(k^6)$.

Now, using the same computer and category number assumptions, give the “big oh” expression for the order of your *simplified* expression for $P(B = b)$, again, in terms of the number of addition and multiplication operations required to compute the answer. Give the numbers of additions and multiplications that are necessary. Also, give the amount of space required to store the distributions.

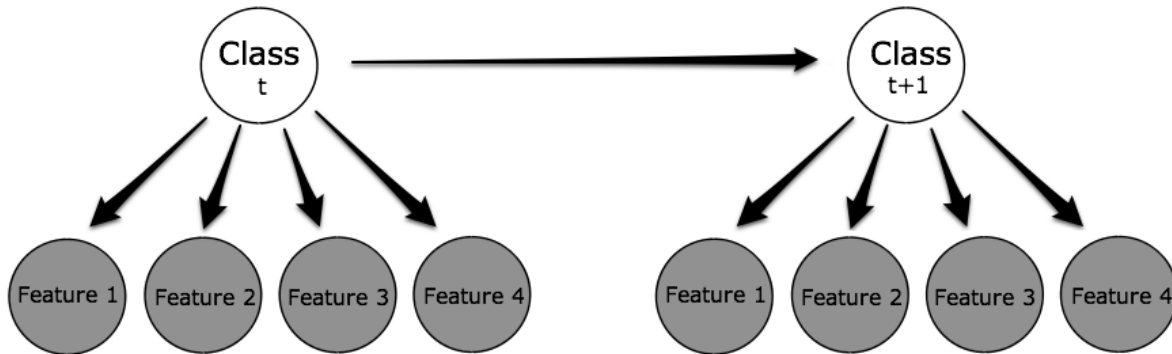
- c. Explain the complexity order relationship between the unsimplified expression of $P(B = b)$ and the simplified expression of $P(B = b)$. Why are they the same or different?
- d. Now, in addition to the above Bayes belief net, you are told that there are only two categories for each variable, 0 (false) and 1 (true). You are also given these specific values for the conditional relationships:

$$\begin{aligned}
 P(A = 1) &= 1/2 \\
 P(B = 1|A) &= \begin{cases} 1/2 & \text{if } A=1 \\ 3/4 & \text{if } A=0 \end{cases} \\
 P(C = 1|A) &= \begin{cases} 1/4 & \text{if } A=1 \\ 1/2 & \text{if } A=0 \end{cases} \\
 P(D = 1|B) &= \begin{cases} 2/5 & \text{if } B=1 \\ 1/4 & \text{if } B=0 \end{cases} \\
 P(E = 1|B) &= \begin{cases} 2/3 & \text{if } B=1 \\ 4/5 & \text{if } B=0 \end{cases} \\
 P(F = 1|C, E) &= \begin{cases} 1/5 & \text{if } C=1 \text{ and } E=1 \\ 2/5 & \text{if } C=1 \text{ and } E=0 \\ 2/5 & \text{if } C=0 \text{ and } E=1 \\ 1/5 & \text{if } C=0 \text{ and } E=0 \end{cases}
 \end{aligned}$$

Using these specific values for the conditional distributions, calculate a numerical value for $P(B = 1)$ by hand.

- e. Using the same specific values for the conditional distributions in part (e), you are told that $F = 1$. Calculate $P(B = 1|F = 1)$ by hand.
- f. Compute $P(B = 1)$ and $P(B = 1|F = 1)$ by the Bayes Net Toolbox for Matlab.

Problem 4: Dynamic Bayesian Networks



Now you are accustomed to modeling static Bayes nets. In this problem you play with a Dynamic Bayes Net (DBN) shown in the figure. This DBN has one discrete hidden (class) node and four discrete feature (observed) nodes per time slice. Again we use the Bayes Net Toolbox for Matlab. Please refer to the following tutorial page:

http://www.cs.ubc.ca/~murphyk/Bayes/usage_dbn.html

To help you solve this problem, we provide you with our Matlab example codes for the sample DBN (with one binary class node and two binary feature nodes) that we used in the lecture. Please download two Matlab files (“**generate_samples_dbn.m**”, “**learning_dbn.m**”, “**inference_dbn.m**” from the course website and run the codes to see how they works. Also, modify these codes to solve this problem. This will be easy, right?

The problem is decomposed into three phases. First, you model the DBN shown in the figure, and generate a toy dataset (1000 samples) from your modeled DBN (*Data Generation Phase*). Second, you divide the dataset into two parts (first 75% for training data and remaining 25% for testing data), and learn another new DBN based on the “training data” (*Learning (or Training) Phase*). Third, you test this learned DBN with the “testing data” (*Inference (or Testing) Phase*).

Although in this problem you deal with a toy dataset you generate in the data generation phase, you can use a similar approach (learning and inference) to train and test DBNs for target datasets in your research field. For your practical (research) purpose, the data generation phase in this problem could be replaced with your own research experiments that gather your target datasets.

- a. Define the DBN shown in the figure: one discrete hidden (class) node and four discrete feature (observed) nodes per time slice. Assume that the hidden class node and all feature nodes are binary (1 or 2). Then, generate 1000 samples (of two time slices) from the modeled DBN. Use the following probabilities for modeling the DBN, and modify “**generate_samples_dbn.m**”. (The given sample code models a DBN with

one class node and two discrete feature nodes, so you have to change the code.) Write the printed CPT values. Also, explain the structures of the two variables in the code, a DBN sample instance “ev” and the corresponding evidence “cases{i}”.

C = Class node in time slice t ,

C' = Class node in time slice $t + 1$,

$F1$ = Feature 1 node, $F2$ = Feature 2 node,

$F3$ = Feature 3 node, $F4$ = Feature 4 node

Prior probabilities: $P(C = 1) = 0.5, P(C = 2) = 0.5$

Conditional probabilities: (*Warning: all the features take 1 or 2, not 0 or 1. This is also the convention for binary nodes in the toolbox.*)

$P(F1 = 1|C = 1) = 0.8, P(F1 = 1|C = 2) = 0.3$

$P(F2 = 1|C = 1) = 0.5, P(F2 = 1|C = 2) = 0.7$

$P(F3 = 1|C = 1) = 0.2, P(F3 = 1|C = 2) = 0.8$

$P(F4 = 1|C = 1) = 0.8, P(F4 = 1|C = 2) = 0.1$

State transition probabilities:

$P(C' = 1|C = 1) = 0.7, P(C' = 1|C = 2) = 0.2$

- b. Learn the parameters of a new DBN based on the training data (first 750 samples) generated in the previous phase. Modify “**learning_dbn.m**”. (Note that the given sample code uses all the generated samples (1000 samples) for learning the parameters, so you have to change the code.) Run many times varying the seed value of random number generation, and try to maximize the log likelihood in the EM steps. Write your seed value, the printed EM steps (with log likelihood values) and the obtained parameters (printed CPT values). Compare this estimated CPT values with the original CPT values. Also, explain the relationship among the number of iterations, the log likelihood and the accuracy of the obtained values.
- c. Now you use the portion of the data (remaining 250 samples) set aside for testing to test the obtained classifier in the previous phase. Modify “**inference_dbn.m**”. (Note that the given sample code uses all the generated samples (1000 samples) for testing the classifier, so you have to change the code.) Show the confusion matrix of the results using a DBN junction tree inference engine. What is the test accuracy?
- d. Using the same generated data, do the learning and testing phases again. But this time use the first 250 samples for testing and the last 750 samples for learning. Show your seed value, the estimated CPT values, the confusion matrix, and the accuracy.

Thank you!

Prediction is very difficult, especially about the future. - Niels Bohr