B551 Fall 2012 Homework 3

Due date: 10/11/2012

1) Build a one-node Bayes network to represent the probability of drawing candy flavors out of a bag manufactured by some candy company. The node should be named "Flavor" and have three possible values: "Lime", "Lemon", and "Cherry", with probabilities 0.3, 0.3, and 0.4 respectively.
   a) Set up a loop and draw N samples of the "Flavor" node using the DiscreteCPT.rand_result() method. Count the fraction of samples of each flavor. Repeat this with N=100, N=1,000, and N=10,000. What do you observe?
   b) For each value of N in part b, perform the same experiments 100 times using different random seeds to get estimates $\%Lime_N^k$, $\%Lemon_N^k$, and $\%Cherry_N^k$ for k=1,…,100. For each N, compute the average squared error $E_N = \Sigma_{k,Flavor} (\%Flavor_N^k - \%Flavor)^2$ between the estimated probabilities and the true probabilities. What kind of relationship do you observe between N and E_N?

2) Build a 2-node Bayes network to represent the probability of drawing candy flavors from one of 10 bins in a store. The parent node should be named "Bin" and have possible values "1" through "10". These values should have uniform probabilities (i.e., 10%). The child node should be named "Flavor" and have the following probability table:

Bin	Lemon Lime Cherry
1	0.0, 0.0, 1.0
2	0.0, 1.0, 0.0
3	1.0, 0.0, 0.0
4	0.0, 0.33, 0.67
5	0.0, 0.67, 0.33
6	0.33, 0.0, 0.67
7	0.67, 0.0, 0.33
8	0.33, 0.67, 0.0
9	0.67, 0.33, 0.0
10	0.33, 0.33, 0.33

Suppose a child comes to us with a lime-flavored candy that he or she got out of one of the bins. Show the calculation steps needed to determine the probability that the candy came from each bin? In other words, describe how to compute P(Bin|Flavor=Lime) from the information available in the Bayesian network. Use the network's enumerate_ask() method to verify that your calculations are correct.

3) For your two-node Bayesian network from question 2, consider now the *learning* problem in which you do not know the prior parameters of the network. This question will simulate a “real world” scenario in which you can come up with a reasonable Bayesian network structure, but want to estimate parameters from data.

   To generate a sample set, generate N=100 samples from the true Bayes net by repeatedly calling DiscreteBayesNet.monte_carlo_sample(). Then, train the parameters using DiscreteBayesNet.train_ml(). Finally, we will evaluate how well we’ve learned by using the trained network to compute P(Bin|Flavor=Lime). What do you observe? Repeat this experiment for N=1,000 and N=10,000.

4) In the files "poll_train.txt" and "poll_test.txt" there are samples of people's voting habits. Each file contains one polling record per line. Each record consists of 6 values representing, in order, the voter's gender (male/female, notated M and F), race or ethnicity (white/black/hispanic/asian/other, represented by first letter of value), age ('0' for 18-29, '1' for 30-45, '2' for 45-59, '3' for 60 and up), political ideology (progressive/moderate/conservative, by first letter), population of local area ('LC' for large city, 'SC' for small city, 'SB' for suburbs, 'ST' for small town, and 'RU' for rural), and vote (democrat/republican/independent, by first letter).

   In the file politics.py is a dict named politicsVars. It defines variable names for each of the parts in a voting record, and maps those to the possible values for those variables. In the same file are functions named load_samples_training() and load_samples_testing(). Each takes a filename as an argument and returns a list of samples.

   Each item in the list returned by load_samples_training() is a dict mapping all the variable names to the values of one polling record. This list can be passed directly into DiscreteBayesNet.train_ml(). Each item in the list returned by load_samples_testing() is a tuple. The first entry in the tuple is a dict like the entries from load_samples_training(), EXCEPT that the dict does not contain an entry for the Vote variable. You may use this dict as evidence in DiscreteBayesNet.enumerate_ask(). The second entry in the tuple is the value of the Vote variable corresponding to that entry, which can be held in reserve and used to test an estimate of the Vote value.

   To test a network designed to model this system, we would use load_samples_testing() to load some test cases from a file, then step through those testing samples. On each sample, we would compute P(Vote | evidence=sample[0]), and derive a guess for the Vote value based on that distribution. Then we would test our guess against the outcome given in sample[1]. After trying all the samples, we can divide the number of correct guesses by the number of samples and obtain our accuracy.

   To test the accuracy of your network, use testing samples from poll_test.txt in the questions below.

   a. Build a Naïve Bayes classifier with Vote as the class variable.
   b. What independence relations exist in this network?
   c. Use DiscreteBayesNet.train_ml() to train the network parameters using the samples in the file poll_train.txt.
   d. Explain the theoretical and practical differences between DiscreteBayesNet.enumerate_ask(), DiscreteBayesNet.variable_elimination_ask(), and the Monte-Carlo methods in DiscreteBayesNet.monte_carlo_estimate() and DiscreteBayesNet.likelihood_weighting_estimate(). Which method would you recommend for this network? Why?
   e. What is the accuracy of this classifier on the poll_train.txt? On poll_test.txt?
   f. It is strange to think of a vote as *causing* a person's demographic characteristics via the arrows in the Bayesian network. Why doesn't this seem to matter for the vote prediction task?

   For the remaining questions, you will build your own network to represent this problem

   g. Construct a Bayesian network representing the system. Write down the parents of each node in the network.
   h. What independence relations exist in your network?

i. Try to hand-tune your network's probability tables. What is the best accuracy you can achieve? What are the greatest difficulties you encounter? (Note: Superhuman efforts are not required here. Don't spend more time than you need to in order to get a decent grasp of the situation and answer the questions.)
j. Use DiscreteBayesNet.train_ml() to train your network on the file poll_train.txt. What accuracy do you achieve now? How does this process compare to hand-tuning?
k. Is it always a good idea to add additional edges to your network? Give at least one advantage with doing so. Give at least three disadvantages with doing so.

Submission Instructions

Place the code you use to answer each question in hw3.py, in functions p1() through p4(). Your functions should run and print out the results of your experiments when hw3.py is executed via the command ‘python hw3.py’. Turn in hw3.py and hw3_answers.txt on OnCourse.