Advanced topics in data modelling Assignment1: Graphical models

Mariuta Nicolae (rqt629) May 5, 2015

Question 1

Ancestral sampling

Ancestral sampling is a sampling technique which allows obtaining samples directly from the graphical model for a joint distribution.

The algorithm takes as input a joint distribution of the form $p(x_1, x_2, \ldots x_k)$, and if we represent the probability distribution as a Bayesian network it is necessary to store in an order that from one node there is no link to a previously considered node. This can be done, as told in Bishop's book, by numbering the nodes in that when we take the nodes in the numbering order, then at each node where sampling is done, we already have the sample values for their parents already calculated.

If the input satisfies the requirement previously described, the next step is to take all nodes in order and at each node i calculate $p(x_i|pa_i)$ which is the function for the conditional distribution of node $x_i$where $pa_i$are the parents of the node for which the sample value was already calculated. The technique is to previously generate a set of random numbers within an interval(for example, between 0 and 1) and split it into a number of intervals equal to the number of possible values that the sample may have and size proportional with the probability that the variable to have each value(obtained from the conditional distribution function). The sample value for a node is assigned corresponding to the interval from which the randomly generated number is part of.

After the last node is reached, the process is repeated by taking the next set of randomly generated numbers and calculating the corresponding sample value for each node. If we want to calculate an accurate approximation for any marginal probabilities then it is good to take a many samples as possible.

The goal of this sampling method is to find samples from the joint distribution. This is useful because, after a large number of samples areobtained, we can approximate values for any marginal or conditional probabilities by looking at the number of apparitions in the samples for any combination of values.

Markov blanket sampling

This sampling can also be done by using the property that each node is dependent only on: parent, its children and the parents of children.

The goal of this technique is to provide a faster method of obtaining a sample for only some nodes in the Bayesian network when we are not interested in the rest of the graph. This can be useful when we want to sample only some nodes in a huge network and we are only interested to obtain samples for some part of this network.

To achieve this, we have to obtain the Markov blanket which consists of the parents, children and parents of children for the node that is observed. We have to make sure that all those nodes in the blanket have been observed (calculate their sample value also based on the parents of the nodes in the blanket if they are also conditional dependent of some other nodes that were not observed). After the Markov blanket is obtained and all the nodes in the blanket have been observed, then we can calculate directly the sample value for the node that is currently being observed.

This procedure is repeated for all the nodes in the network that we want to analyze and by doing this we only have to observe the nodes that are required to be able to calculate the samples for that part of the Bayesian network. That is why the goal is achieved and the time for the computation is reduced in the case of a large and complex network.

Question 2

No, it is not possible to calculate the sample value when some unobserved nodes have unobserved parents because, as I described at the first question, at each node we have to calculate the value of conditional probability which is a function that takes as input the values calculated for the parent nodes. If the parent nodes are unobserved, then it is impossible to calculate the value of conditional distribution for that specific node.

This is the reason why we have to make sure that we calculate the sample values for each node in an order that for each node that is observed, its parent nodes have already been observed and this can be done by numbering the nodes or sorting them into a list so this issue will not be encountered.

Question 3

For the implementation I have used Matlab version 2014 and I have not used any 3rd party libraries because I understood quite well how to implement the algorithm and, even if my code is not parameterized for any structure of Bayesian network and contains much redundant code, I considered that is easier to make the implementation of my own with basic usage of Maltab built-in functions for making operations with matrices since the Bayesian network is not so complex. All of the code is included in the script with the name *ancestralSampling.m* and you simply have to run the code from that file into Matlab to obtain the results I will describe.

The pseudocode for my implementation is the following:

```
Procedure AncestralSampling(nrSamples, bayesianNetwork)
Input: nrSamples the number of samples that have to be obtained
bayesianNetwork = {x1, x2,...xn} the nodes of the bayesianNetwork
ordered so there is no link to a previous node
1 initialize array Samples with size nrSamples
2 for k = 1 to nrSamples
3         generate n random numbers and store them into array Randoms
4         for each xi from bayesianNetwork
5             Probability = SigmoidFunction([s1,s2,...sample values from
parent nodes of xi])
6             if Randoms(i) < Probability
7                 store sample 0 at Samples(k,i)
8             else store sample 1 at Samples(k,i)
9 calculate desired marginal probabilities using Samples
```

Now I will describe more exactly how I implemented the algorithm. I have the parameter `nrSamples` that determine how many samples I extract for each variable. I have noticed that after value of 10000 that I have set, the values of marginal probabilities are no longer changing much between the runs and at higher values the computation time is quite long. After deciding the number of samples to calculate, I have created the matrix *samples* with `nrSamples`rows and 7 columns (the number of nodes in the graph).

After this, at each of the `nrSamples`iterations I generated a set of 7 random numbers (to calculate sample for each variable). In the next step I calculated the conditional probability distribution for each of the 7 nodes by using the sigmoid function (as described in the assignment text) and stored them in different variables. According to the dependencies, I have calculated the following probabilities for each variable:

$$P(x_1 = 0)$$

$$P(x_2 = 0)$$

$$P(x_3 = 0)$$

$$P(x_4 = 0|x_1, x_2, x_3)$$

$$P(x_5 = 0|x_1, x_3)$$

$$P(x_6 = 0|x_4)$$

$$P(x_7 = 0|x_4, x_5)$$

In order to decide which the value of sample is, I assigned 0 if the value of the random number is smaller than $P(x_i = 0\ |Pa(x_i))$ and value 1 if the value of the random number is greater or equal than the probability that the variable to be 0. After calculating the sample for each node I stored them into the matrix.

After obtaining all the desired samples I calculated the approximate of marginalprobability for each node by counting the number of apparitions for sample value 0 and 1 at each node and dividing to the total number of samples. I have obtained the table:

	x1	x2	x3	x4	x5	x6	x7
0	0.7285	0.8798	0.9505	0.9232	0.9777	0.9959	0.998
1	0.2715	0.1202	0.0495	0.0768	0.0223	0.0041	0.002

After this I also calculated the marginal probabilities for all 8 possible values of node triplet $(x1, x4, x7)$ by calculating the number of apparitions for each of the possible value and, again, I divided to the total number of samples. I have obtained the following table:

x1	x4	x7	P(x1,x4,x7)
0	0	0	0.7014
1	0	0	0.221
0	1	0	0.0004
1	1	0	0.0008
0	0	1	0.7014
1	0	1	0.221
0	1	1	0.0004
1	1	1	0.0008

By looking at the results in the table it visible that node x3 and all nodes depending on that one have quite higher probability to be 0 because the value of exponent in the sigmoid function is very small(because of power -3 and larger) and then the result of division is getting closer and closer to 1. Because after x3 all nodes depend on the result of observing node x3, there is very small chance to have sample value 1 at all of the nodes.

As a conclusion, the implementation of Ancestral Sampling is quite straightforward and simple, just observe the nodes in the right order and calculate sample for each according to the dependencies. In my implementation is not quite optimal that I calculated probability distribution for each node separately and the node cannot be used for other network and this way will be impossible to make Ancestral Sampling for a large network. Since the given network is not very complex and I am not familiar with how to implement tree structure in Matlab, I have chosen this more basic approach. However, it is visible that for calculating with good approximation the marginal probability for a set of nodes it is needed a very large number of samples and this method will need large computation time for a very large Bayesian network and there it would be more useful to use the Markov blanket.