COMS W4701: Artificial Intelligence, Spring 2022

Homework 5

Instructions: Compile all solutions to the written problems on this assignment in a single PDF file. **Show your work by writing down relevant equations or expressions, and/or by explaining any logic that you are using to bypass known equations**. When ready, follow the submission instructions to submit all files to Gradescope. Please be mindful of the deadline, as late submissions are not accepted, as well as our course policies on academic honesty.

Problem 1: Descendants of Effects (20 points)

We will investigate the absence of conditional independence guarantees between two random variables when an arbitrary descendant of a common effect is observed. We will consider the simple case of a causal chain of descendants:

$$\begin{tikzcd} A \arrow[dr] & & & & \\ & D_0 \arrow[r] & D_1 \arrow[r] & \cdots \arrow[r] & D_n \\ B \arrow[ur] & & & & \end{tikzcd}$$

Suppose that all random variables are binary. The marginal distributions of A and B are both uniform $(0.5, 0.5)$, and the CPTs of the common effect D_0 and its descendants are as follows:

A	B	$\Pr(+d_0 \mid A, B)$
$+a$	$+b$	1.0
$+a$	$-b$	0.5
$-a$	$+b$	0.5
$-a$	$-b$	0.0

D_{i-1}	$\Pr(+d_i \mid D_{i-1})$
$+d_{i-1}$	1.0
$-d_{i-1}$	0.0

(a) Give an analytical expression for the joint distribution $\Pr(D_0, D_1, \cdots, D_n)$. Your expression should only contain CPTs from the Bayes net parameters. What is the size of the full joint distribution, and how many entries are nonzero?

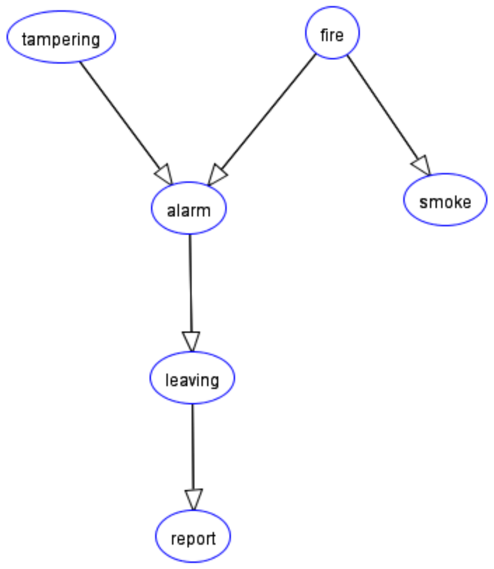
(b) Suppose we observe $D_n = +d_n$. Numerically compute the CPT $\Pr(+d_n|D_0)$. Please show how you can solve for it using the joint distribution in (a), even if you do not actually use it.

(c) Let's turn our attention to A and B. Give a **minimal** analytical expression for $\Pr(A, B, D_0, +d_n)$. Your expression should only contain CPTs from the Bayes net parameters or the CPT you found in part (b) above.

(d) Lastly, compute $\Pr(A, B \mid +d_n)$. Show that A and B are not independent conditioned on D_n.

Problem 2: Bayes Net 1 (20 points)

The following Bayes net is the "Fire Alarm Belief Network" from the Sample Problems of the Belief and Decision Networks tool on AIspace. All variables are binary.



(a) Which pair(s) of nodes are guaranteed to be independent given no observations in the Bayes net? Now suppose Alarm is observed. Identify and briefly explain the nodes whose conditional independence guarantees, given Alarm, are **different** from their independence guarantees, given no observations.

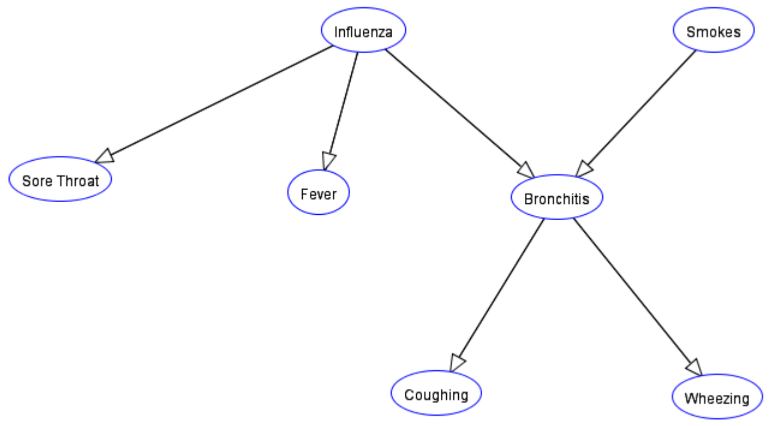
(b) We are interested in computing the conditional distribution Pr(Smoke | report). Give an analytical expression in terms of the Bayes net CPTs that computes this distribution (or its unnormalized version). What is the maximum size of the resultant table if all marginalization is done at the end?

(c) We employ variable elimination to solve for the query above. Identify a variable ordering that i) yields the greatest number of operations possible, and ii) yields the fewest number of operations possible. Also give the max table sizes in each case.

(d) Following your second variable ordering above, numerically solve for Pr(Smoke | report) using the default parameters in the applet example. You may check your answer using the applet, but you should work it out yourself and show your work.

Problem 3: Bayes Net 2 (20 points)

The following Bayes net is the "Simple Diagnostic Example" from the Sample Problems of the Belief and Decision Networks tool on AIspace. All variables are binary.



(a) We can describe all guaranteed independences in the Bayes net by defining two or more *subsets* of nodes S_i, such that all nodes in S_i are independent of all nodes in S_j for $i \neq j$. For example, we can define $S_1 = \{\text{Smokes}\}$ and $S_2 = \{\text{Influenza, Sore Throat, Fever}\}$ given no observations. Do the same for *conditionally independent* nodes i) given Influenza, ii) given Bronchitis, and iii) given both Influenza and Bronchitis. Make sure your answers capture **all** guaranteed independences.

(b) Consider using likelihood weighting to solve two queries, one in which Influenza and Smokes are observed, and one in which Coughing and Wheezing are observed. Explain how the two cases differ in the distribution of the resulting samples, as well as the weights that are applied to the samples.

(c) We perform Gibbs sampling and would like to resample the Influenza variable conditioned on the current sample $(+s, +st, -f, -b, +c, -w)$. Give a **minimal** analytical expression for the sampling distribution $\Pr(\text{Influenza} \mid \textit{sample})$ (or its unnormalized form). What is the maximum size of the table that has to be constructed?

(d) Numerically solve for the sampling distribution $\Pr(\text{Influenza} \mid \textit{sample})$ using the default parameters in the applet example. You may check your answer using the applet, but you should work it out yourself and show your work.

Problem 4: Bayes Net Experiments

Many commonly used Bayesian network benchmarks, many of which were proposed in actual research papers, can be found in the bnlearn repository. pgmpy is a Python library that contains all of these networks and also provides implementations of inference and learning algorithms for general Bayes nets and other probabilistic models.

Most of the code that you will write for this problem will be scripting code rather than function implementations. While you will still turn in all code that you do write, we recommend that you use Jupyter for flexibility in running different experiments and generating plots. An autograder will not be used to grade your code.

4.1: Exact Inference (10 points)

You will perform one inference query per tuple in the following list. The first tuple entry is the model name, and the second tuple entry is the variable whose distribution you are solving for. There are no evidence variables in any of these queries.

```
[('earthquake',JohnCalls'),('survey','T'),('asia','dysp'),('sachs','Akt'),
('child','LowerBodyO2'),('insurance','PropCost'),('alarm','BP'),('hailfinder','R5Fcst')]
```

The Bayes nets above range from 5 nodes to 56 nodes. You are highly encouraged to look at the structure of each on the bnlearn repository. For each of the tasks below, you should following the examples here, load each example network, create a `VariableElimination` instance, and then run the `query` command with the appropriate inputs. You will also be **timing** each of the queries, for example by calling `default_timer()` from the timeit library before and after each query.

(a) Perform inference on each of the Bayes net / variable pairs above using the default options. Record the time taken for each one and produce two scatter plots: i) **log** of seconds taken vs number of nodes, and ii) **log** of seconds taken vs number of Bayes net parameters. Briefly explain how these plots show that exact inference is generally exponential in Bayes net size.

(b) Repeat the above experiments but using the option `elimination_order='MinNeighbors'`. Produce the same two plots as described above. Briefly contrast your observations here with those of part (a), and explain the effect of this elimination ordering.

4.2: Sampling (20 points)

You will now investigate the efficacy of using sampling to perform approximate inference on the `'insurance'` Bayes net. First, recompute (if necessary) the distribution Pr(PropCost) from exact inference. You can simply store the probabilities in a 1D NumPy array (but do note the ordering of the PropCost domain values). Then study the documentation and example on this page, and create an `ApproxInference` instance.

(a) Solve for Pr(PropCost) using n samples, for values of n ranging from 100 to 10000 in increments of 100. For each solution, note the time taken, and also compute the **error** between the returned distribution and the exact one as the Euclidean norm of the difference between the two arrays. **Please be aware that you may have to reorder the returned distribution to match the one computed from exact inference.** Produce two plots, one showing time taken per number of samples, the other showing error per number of samples.

(b) Briefly comment on how computational complexity scales with number of samples (there may be outliers in your plot, which you can ignore). Also comment on the efficacy of using sampling vs exact inference; after about how many samples does the error stop improving, and how close do we get to the exact inference result?

(c) Next, you will perform a comparison of rejection sampling and likelihood weighting, both described here. We will be sampling conditioning on subsets of the following evidence:

```
[State('Age','Adult'), State('DrivHist','Zero'), State('VehicleYear','Current'),
State('ThisCarDam','None'), State('MakeModel','Luxury'), State('HomeBase','City'),
State('Cushioning','Good'), State('Airbag','True'), State('AntiTheft','True')]
```

Iteratively perform sampling on increasing subsets of the evidence variables above, each time returning 100 samples. That is, generate 100 samples conditioned on `'Age'` only, then 100 samples conditioned on both `'Age'` and `'DrivHist'`, and so on. You would thus have 9 sampling queries for one experiment. Do this for both rejection sampling and likelihood weighting and record the time taken for each query, as well as **average sample weight** for the latter method. Produce three plots: time taken vs number of evidence variables for rejection sampling, time taken vs number of evidence variables for likelihood weighting, and average sample weight vs number of evidence variables (for likelihood weighting).

(d) Briefly comment on how time needed to perform sampling varies with the number of evidence variables in each method. Explain the trends that you see (e.g., why do times increase or stay constant?). Also comment on how sample weights vary with the number of evidence variables.

4.3: Parameter Learning (10 points)

In this last part, you will perform parameter learning for the "Fire Alarm Belief Network" from Problem 2. We provide two data sets. The full one contains 1000 samples of all variables in the network; the partial one has the same data but without the `‘tampering’` and `'fire'` columns. You will use maximum likelihood estimation on the former and expectation maximization on the latter.

First import both data sets using `pandas.read_csv()`. Follow the tutorial here to define two model structures (the only difference is that one of them explicitly labels the two latent variables). Then run the parameter learning algorithms, both with default settings, to find the best fitting CPDs for both models.

When finished, you can iterate over the learned CPDs and print them out. Print or screenshot the results in your PDF. Unless you were extremely lucky, you should see that the two methods yield very different distributions in at least some of the variables. For example, a distribution may be very lopsided toward one value from MLE, but it may then be closer to uniform or more lopsided toward the other value from EM. Briefly explain why the two methods can produce such drastically different results when the data is identical apart from two hidden variables.

Submission

You should have one PDF document containing all solutions, responses, and plots. You should also have a Python file or notebook containing all code that you wrote for problem 4. Submit the document and code file to the respective assignment bins on Gradescope. For full credit, you must tag your pages for each given problem on the former.