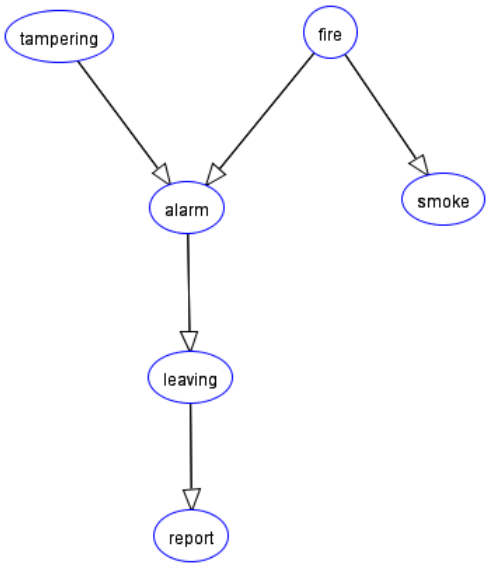
COMS W4701: Artificial Intelligence, Summer 2024

Homework 5a

Instructions: Compile all written solutions for this assignment in a single, typed PDF file. If you use Python or any other computational tool, append the code to your PDF file. Turn in your submission on Gradescope, and **tag all pages**. Please be mindful of the deadline and late policy, as well as our policies on citations and academic honesty.

Problem 1: Fire Alarm Bayes Net (16 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.



1. Consider using likelihood weighting to solve for a posterior distribution conditioned on the following sets of evidence: i) Tampering and Fire, ii) Smoke and Report. For each case, explain whether the generated samples reflect the prior or posterior distribution. Also explain the impact of the likelihood weights in helping us to estimate the posterior in each case.

2. Inspect the individual CPTs of this Bayes net in the applet. Identify all unique evidence events that are impossible to occur. (You don't need to list any events that contain those you already listed as subsets. For example, if you identify $\{+t, +f\}$ to be such an event, you do not need to write $\{+t, +f, +a\}$, $\{+t, +f, +s\}$, etc.) What would happen if we perform rejection sampling or likelihood weighting with these events as observed?

3. We perform Gibbs sampling and want to resample the Alarm variable conditioned on the current sample $(+t, -f, -s, -l, -r)$. Find a minimal analytical expression for Pr(Alarm |

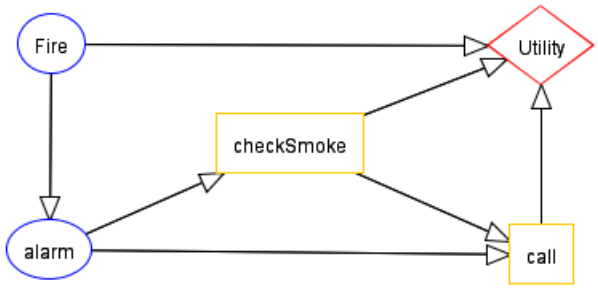
sample) (or its unnormalized version) in terms of the Bayes net CPTs, and then numerically compute it using the applet parameters.

4. Repeat the above, but now suppose that we have lost the value of Leaving in the sample. In other words, compute the distribution $\Pr(\text{Alarm} \mid +t, -f, -s, -r)$. Why does conditioning on less information lead to potentially *greater* computation?

Problem 2: Fire Alarm Decision Network (16 points)

The following Bayes net is a simplified version of the "Fire Alarm Decision Problem" from the Sample Problems of the Belief and Decision Networks tool on AIspace. All variables are binary, and the chance node CPTs as well as utility values are shown below.

While you are not required to do so, we recommend that you use `pandas` functionality to carry out the operations in this problem. It is also acceptable if you prefer to perform all calculations manually. Please include your work in either case.



Fire	CheckSmoke	Call	Utility
True	True	True	−220
True	True	False	−5020
True	False	True	−200
True	False	False	−5000
False	True	True	−220
False	True	False	−20
False	False	True	−200
False	False	False	0

Fire	Pr(Fire)
T	0.01
F	0.99

Fire	Alarm	Pr(Alarm | Fire)
True	True	0.95
True	False	0.05
False	True	0.01
False	False	0.99

1. Compute the joint factor required to sum out the chance nodes that are not parents of any decision nodes. Show the resultant factor after performing the elimination.

2. Determine the optimal decision function and expected utilities for Call. Show the table containing this information.

3. Briefly explain whether the optimal decision function for Call actually depends on both of its parent nodes. Would the result change if we had determined the optimal decision function for CheckSmoke first, followed by Call?

4. Determine the optimal decision function and expected utilities for CheckSmoke. Show the table containing this information. Then compute the maximum expected utility.

We will be working with the following data set for Problems 3 and 4. There are two trinary features x_1 and x_2, and a class variable y with values 0 and 1.

Sample	x_1	x_2	y
1	$+1$	-1	0
2	$+1$	$+1$	0
3	0	-1	0
4	0	$+1$	0
5	$+1$	$+1$	1
6	0	0	1
7	0	$+1$	1
8	-1	$+1$	1

Problem 3: Naïve Bayes (16 points)

1. The maximum likelihood class CPT is $\Pr(Y) = (0.5, 0.5)$. Estimate the feature CPTs $\Pr(X_1|Y)$ and $\Pr(X_2|Y)$ without Laplace smoothing.

2. Which value(s) of x_1 give a direct prediction of y regardless of the value of x_2, and what are the predictions of y for each? Give the same analysis for values of x_2. Are there any feature combinations (x_1, x_2) for which a prediction is not well defined?

3. Suppose that we find out that the class values y in the data set may not be correct. We can use the estimated probabilities in Part 1 as a starting point for expectation-maximization. Compute the expected counts $\Pr(Y|x_1, x_2)$ for each sample (x_1, x_2) in the data set.

4. Perform the maximization step and find the new CPTs $\Pr(Y)$, $\Pr(X_1|Y)$, and $\Pr(X_2|Y)$.

Problem 4: Linear Classifiers (16 points)

1. Suppose we drop samples 4 and 5, leaving us with six data points. We want to learn a linear classifier $f_{\mathbf{w}}$ such that we predict 0 when $f_{\mathbf{w}} \leq 0$ and 1 otherwise. Starting with the weight vector $\mathbf{w} = (1, -2, 0)$, use the perceptron learning rule and learning rate $\alpha = 1$ to process the six data points once in order. Show the result of each sample classification and any weight updates that occur.

2. Plot the data in x_1-x_2 space (you can either hand-draw or use Python). Use different indicators for each class. Then overlay the decision boundaries from the original weight vector $\mathbf{w} = (1, -2, 0)$ and the new weight vector $\mathbf{w}'$ that you found in Part 1. Did perceptron find a model that performs perfect classification? Would you say that the model is a good one in terms of separability?

3. Now use sklearn to fit a logistic model to the original data set. Show a DecisionBoundaryDisplay() plot overlaid on a scatter plot of the data. What are the learned weights and classification accuracy on the data set?

4. Compute the mean negative log likelihood of the data given the learned weights. (You can either apply the formula directly or use a function like `predict_log_proba()`.) Compute the gradient of the negative log likelihood, evaluated at the current weight vector.