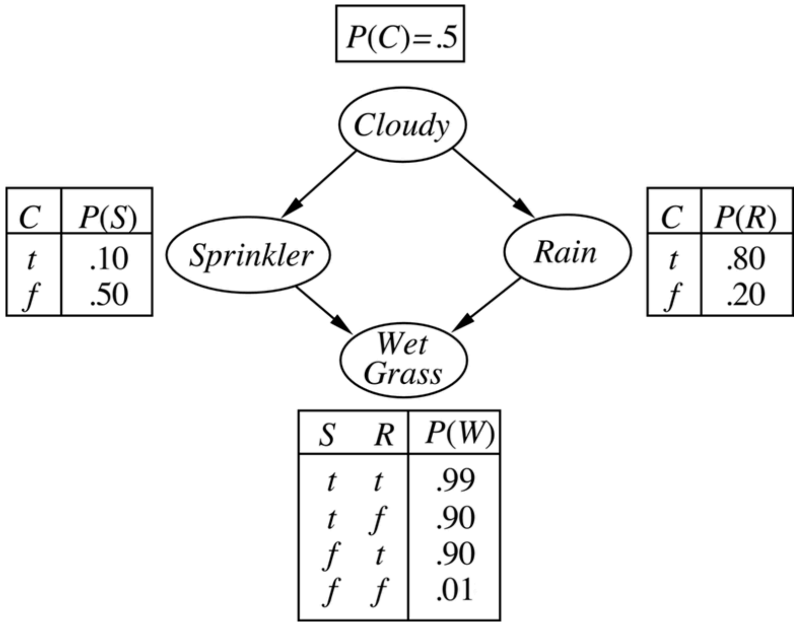
Advanced Artificial Intelligence, 2425

Workshop Task: Quiz on (Approximate) Inference in Bayes Nets

Question 1

30 points



Using the Sprinkler Bayes net discussed in this week's lecture, shown above, answer the following probabilistic queries using exact inference. Your answers should include the first 4 decimals without rounding.

P(S=true|C=true)=[Blank 1]

P(S=false|C=true)=[Blank 2]

P(R=true|C=true)=[Blank 3]

P(R=false|C=true)=[Blank 4]

P(W=true|S=false,R=true)=[Blank 5]

P(W=false|S=false,R=true)=[Blank 6]

Here an example for the first query: **`python BayesNetInference.py InferenceByEnumeration .\config\config-sprinkler.txt "P(W|S=false,R=true)"`**

Now answer the probabilistic query *P(W|S=false,R=true)* but this time using Rejection Sampling with varying amounts of samples and multiple times due to randomness in the outputs. This part requires you to fill a spreadsheet such as the one below to record your results when running **`python BayesNetInference.py RejectionSampling .\config\config-sprinkler.txt "P(W|S=false,R=true)" $YourNumberOfSamples`**. Your workshop materials of this week have an Excel template for your convenience under the doc folder: doc/ SamplingExperiment-sprinkler-net.xlsb

SRINKLER network								
	N=100		N=1000		N=10000		N=100000	
Run	true	false	true	false	true	false	true	false
1								
2								
3								
4								
5								
AVG	#DIV/0!	#DIV/0!	#DIV/0!	#DIV/0!	#DIV/0!	#DIV/0!	#DIV/0!	#DIV/0!
STD	#DIV/0!	#DIV/0!	#DIV/0!	#DIV/0!	#DIV/0!	#DIV/0!	#DIV/0!	#DIV/0!

Is the standard deviation of N=100 lower than 0.1 and compatible samples >0? Answer (Y/N)=[Blank 7]
Is the standard deviation of N=1000 lower than 0.1 and compatible samples >0? Answer (Y/N)=[Blank 8]
Is the standard deviation of N=1000 lower than 0.1 and compatible samples >0? Answer (Y/N)=[Blank 9]
Is the standard deviation of N=10000 lower than 0.1 and compatible samples >0? Answer (Y/N)=[Blank 10]

Now repeat the sampling experiment but this time using Gibbs Sampling with bnlearn code. Fill the corresponding spreadsheet according to the outputs of the following -- varying the number of samples in each run (or running each value of N but 5 times): `python bnlearn_GibbsSampling-sprinkler.py`

Is the standard deviation of N=100 lower than 0.1 and compatible samples >0? Answer (Y/N)=[Blank 11]
Is the standard deviation of N=1000 lower than 0.1 and compatible samples >0? Answer (Y/N)=[Blank 12]
Is the standard deviation of N=1000 lower than 0.1 and compatible samples >0? Answer (Y/N)=[Blank 13]
Is the standard deviation of N=10000 lower than 0.1 and compatible samples >0? Answer (Y/N)=[Blank 14]

Answers to the questions above can help to determine an appropriate number of samples (N) for this scenario, which may not apply to others.

Blank 1

Blank 2

Blank 3

Blank 4

Blank 5

Blank 6

Blank 7

Blank 8

Blank 9

Blank 10

Blank 11

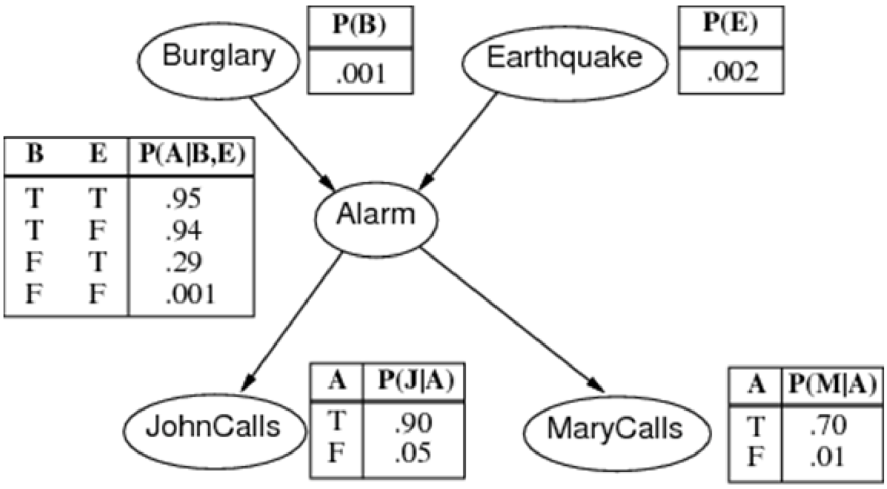
Blank 12

Blank 13

Blank 14

Question 2

35 points



Using the Burglary Bayes net discussed in week 2, shown above, answer the following probabilistic queries using exact inference. Your answers should include the first 4 decimals without rounding.
P(B=true|J=true,M=true)=[Blank 1]
P(B=false|J=true,M=true)=[Blank 2]
P(E=true|J=true,M=true)=[Blank 3]
P(E=false|J=true,M=true)=[Blank 4]

Here an example for the first query: **`python BayesNetInference.py InferenceByEnumeration .\config\config-alarm.txt "P(B|J=true,M=true)"`**

Then answer the probabilistic query *P(B|J=true,M=true)* using Rejection Sampling with varying amounts of samples and multiple times due to randomness in the outputs. This requires you to fill a spreadsheet such as the one below to record your results when running **`python BayesNetInference.py RejectionSampling .\config\config-alarm.txt "P(B|J=true,M=true)" $YourNumberOfSamples`**. Your workshop materials have an Excel template for your convenience: doc/SamplingExperiment-alarm-net.xlsb

ALARM network								
	N=100		N=1000		N=10000		N=100000	
Run	true	false	true	false	true	false	true	false
1								
2								
3								
4								
5								
AVG	#DIV/0!	#DIV/0!	#DIV/0!	#DIV/0!	#DIV/0!	#DIV/0!	#DIV/0!	#DIV/0!
STD	#DIV/0!	#DIV/0!	#DIV/0!	#DIV/0!	#DIV/0!	#DIV/0!	#DIV/0!	#DIV/0!

Is the standard deviation of N=100 lower than 0.1 and compatible samples >0? Answer (Y/N)=[Blank 5]
Is the standard deviation of N=1000 lower than 0.1 and compatible samples >0? Answer (Y/N)=[Blank 6]
Is the standard deviation of N=1000 lower than 0.1 and compatible samples >0? Answer (Y/N)=[Blank 7]
Is the standard deviation of N=10000 lower than 0.1 and compatible samples >0? Answer (Y/N)=[Blank 8]

Now repeat the sampling experiment but this time using Gibbs Sampling with bnlearn code. In contrast to the previous exercise (using the Sprinkler network) that imports an already existing Bayes net, this exercise will create the Burglary network from scratch. This suggests to inspect the Python code either during the workshop or in your own time. Fill the corresponding spreadsheet according to the outputs of the following program -- varying the number of samples in each run (or running each value of N but 5 times):
`python bnlearn_GibbsSampling-alarm.py`

Is the standard deviation of N=100 lower than 0.1 and compatible samples >0? Answer (Y/N)=[Blank 9]
Is the standard deviation of N=1000 lower than 0.1 and compatible samples >0? Answer (Y/N)=[Blank 10]
Is the standard deviation of N=1000 lower than 0.1 and compatible samples >0? Answer (Y/N)=[Blank 11]
Is the standard deviation of N=10000 lower than 0.1 and compatible samples >0? Answer (Y/N)=[Blank 12]

Answers to the questions above can help to determine an appropriate number of samples (N) for this scenario, which may not be the same as the previous question. Using what you consider as an appropriate value for N, answer the query *P(E|J=true,M=true)* using both Rejection and Gibbs sampling. Your results should have been close enough to those with exact inference.

Blank 1

Blank 2

Blank 3

Blank 4

Blank 5

Blank 6

Blank 7

Blank 8

Blank 9

Blank 10

Blank 11

Blank 12

Question 3

35 points

This exercise requires you to use a dataset of cardiovascular data containing discrete random variables with large ranges in their domain values, which is more or less the case of continuous data. But to be able to use the methods discussed so far in the module, such a dataset must be discretised. The CPTs become too large otherwise and with very sparse conditional probabilities, if possible to fit in memory. The following steps are followed the the program below:

- Data discretisation of 5 random variables (only those with a large range of values even if not continuous) . The other random variables remain with their original values due to being discrete and with small domain sizes. Please note that original data is only used for discretisation.
- Parameter learning using Maximum Likelihood Estimation and the discretised data.
- Probabilistic inference of test examples/instances using their original values and corresponding discretised values.

Run the following: **`python bnlearn_DataDiscretisation.py`**

Consider the following evidence sets:

age	gender	height	weight	ap_hi	ap_lo	cholesterol	gluc	smoke	alco	active
17623	2	169	82	150	100	1	1	0	0	1
17482	1	154	68	100	70	1	1	0	0	0

Answer the following probabilistic queries using the first 4 decimals with rounding
What is *P(target=0|evidence_with_age_17623)*? Answer=[Blank 1]
What is *P(target=1|evidence_with_age_17623)*? Answer=[Blank 2]
What is *P(target=0|evidence_with_age_17482)*? Answer=[Blank 3]
What is *P(target=1|evidence_with_age_17482)*? Answer=[Blank 4]

Did the probabilistic predictions agree with ground truth of the test examples? Answer (Y/N)=[Blank 5]

The program above only uses a small portion of the whole training data. Try loading the whole training dataset. Were you able to train a Bayes net using all training data? Answer (Y/N)=[Blank 6]

Blank 1

Blank 2

Blank 3

Blank 4

Blank 5

Blank 6