CS7290 Causal Modeling in Machine Learning: Homework 1

For this assignment, you get to build a generative model with `bnlearn` and with `pyro`. Check out the *bnlearn docs* and the *pyro docs* if you have questions about these packages.

Submission guidelines

Use a Jupyter notebook and/or R Markdown file to combine code and text answers. Compile your solution to a static PDF document(s). Submit both the compiled PDF and source files. If you use Google Collab, send the link as well as downloaded PDF and source files.

Background

Recall the survey data discussed in the lecture.

- **Age (A):** It is recorded as *young* (**young**) for individuals below 30 years, *adult* (**adult**) for individuals between 30 and 60 years old, and *old* (**old**) for people older than 60.
- **Sex (S):** The biological sex of individual, recorded as *male* (**M**) or *female* (**F**).
- **Education (E):** The highest level of education or training completed by the individual, recorded either *high school* (**high**) or *university degree* (**uni**).
- **Occupation (O):** It is recorded as an *employee* (**emp**) or a *self employed* (**self**) worker.
- **Residence (R):** The size of the city the individual lives in, recorded as *small* (**small**) or *big* (**big**).
- **Travel (T):** The means of transport favoured by the individual, recorded as *car* (**car**), *train* (**train**) or *other* (**other**)

Travel is the *target* of the survey, the quantity of interest whose behavior is under investigation.

We use the following directed acyclic graph (DAG) as our basis for building a model of the process that generated this data.
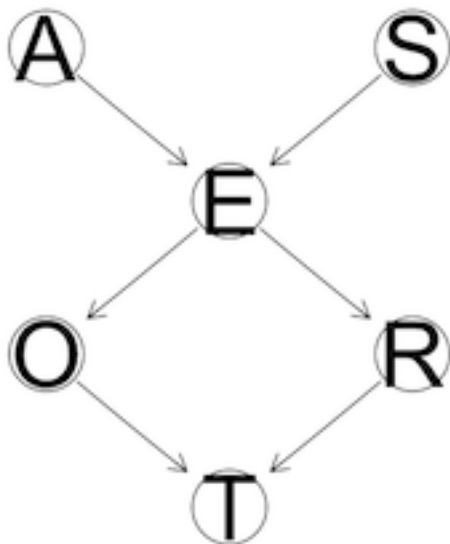

Figure 1: DAG

Question 1: Building a DAG (5 points)

A DAG maps to a factorization of the joint distribution (e.g., $P(A, B, C) == P(A)P(B|A)P(C|B)$). In *bnlearn*, you can use the function `modelstring(dag)` to convert a DAG into a string representation of a factorization of the joint probability distribution. We can go from a string representation to a DAG using the function `model2network(string)`.

1.1

Write out the factorization of the joint distribution implied by the DAG using mathematical notation. (1 point)

1.2

Rewrite the above factorization in *bnlearn*'s string representation. (1 point)

1.3

Use this to create a DAG in *bnlearn*. (1 point)

1.4

Print the class of the DAG object. (1 point)

1.5

Use `graphviz.plot` to plot the DAG. (1 point)

Question 2: Experimenting with graph utilities (5 points)

2.1

Extract and print the nodes and arcs of the DAG you created in previous questions. (1 point)

2.2

Extract and print the parents and the children of each node using `parents` and `children` functions. (1 point)

2.3

Use the `mb` function to extract the Markov blanket of A, E, and T. (1 point)

2.4

How do you identify the Markov blanket from the DAG? (1 point)

2.5

Describe, in terms of coniditional independence (NOT in terms of the DAG) the definition of a Markov blanket. (1 point)

Question 3: Conditional probability distribution (CPD) parameter estimation (5 points)

Bayesian network = DAG + CPD with specified parameters

3.1

Fit the parameters of the DAG from the data stored in survey2.txt using Bayesian estimation, and save the result into an object of class bn.fit. (2 points)

3.2

Play with the Bayesian prior parameter **iss** and report the changes in the parameters learned from Bayesian network. Explain the changes. (3 points)

Question 4: Graph manipulation (4 points)

4.1

Create a copy of the DAG (e.g. `dag2 <- dag`). Remove the arc from Education to Occupation, and plot the result with `graphviz.plot`. (2 points)

4.2

Fit the parameters of the modified network. Which local distributions change, and how? (2 points)

Question 5: Markov equivalence (4 points)

5.1

Compute and plot the PDAG of the DAG for the survey data using the `cpdag` function. Call this PDAG P1 and the original DAG D1. How does P1 and D1 compare? Explain any similarities or differences. (1 point)

5.2

Create a DAG D2 that is the same as D1 except that it has a new arc from Occupation to Residence. This makes sense because surely somebody's job determines where they live (or is it the other way around?). Note that this is a fine example of applying domain knowledge about the data generative process in causal model development. Plot the result with `graphviz.plot`. Now recompute a PDAG P2 from D2. What, if anything, is different between P1 and P2 and what explains these differences or lack of differences? (1 point)

5.3

Create a third DAG D3 that is different from the second DAG (with the O->R edge) but is in the same Markov equivalence class. Do this by reasoning about P2 – in other words look at P2 and create another DAG D3, such that `cpdag(D3)` will also produce P2. Plot D3. (1 point)

5.4

Calculate the log-likelihood of the data given D2 and the log-likelihood of the data given D3. These values should be the same, explain why. You can use the `score` function with the argument `type = 'loglik`, or you can simply se the `logLik` function, which is just a wrapper for `score`. You don't need to provide paramter values for the CPDs of the DAG, `score` will estimate them for you. (1 point)

Question 6: Modeling and Inference using Pyro (18 points)

If you are new to tensor-based frameworks, make sure you give yourself plenty of time for this question. It takes time to get used to debugging. One common source of bugs is integers, *pyro* prefers you use floats (e.g., `torch.tensor(1.0)` instead of `torch.tensor(1)`). If you hit a bug and solve it, why not share with your classmates on Piazza?

6.1 Modeling

Use *pyro* to reimplement the Bayesian network with parameter values you fitted in question 3.Use default *iss* values and round parameter estimates to 2 decimal places. When coding conditional probability table using tensor, make sure its meaning is clear for the purpose of easy debugging and readability. You can use comments or alias as shown here. In this example, it's clear that P('on')=0.8, and P('off')=0.2.

```
X_alias = ['off', 'on']
prob_X = tensor([0.2, 0.8])
```

When there are multiple variables conditioned on, make sure the order they are coded match the order they are sampled in the model. For example, assuming $Z \in \{z1, z2\}$ is conditioned on $X \in \{x1, x2, x3\}$ and $Y \in \{y1, y2\}$, the Conditional Probability Table (CPT) of Z can be coded in two ways (*a* and *b* here), each corresponds to a different order of the dimensions when generating sample of Z using pyro.sample. It is suggested that you write comments on what probability it is (e.g.P(Z=z1|X=x1,Y=y1)) for each entry in the CPT. (6 points)

```
prob_X = tensor([P(X=x1), P(X=x2)])
prob_Y = tensor([P(Y=y1), P(Y=y2)])


a.
prob_Z = tensor([[[P(Z=z1|X=x1,Y=y1), P(Z=z2|X=x1,Y=y1)],
                  [P(Z=z1|X=x1,Y=y2), P(Z=z2|X=x1,Y=y2)]],

                 [P(Z=z1|X=x2,Y=y1), P(Z=z2|X=x2,Y=y1)],
                 [P(Z=z1|X=x2,Y=y2), P(Z=z2|X=x2,Y=y2)]],

                 [P(Z=z1|X=x3,Y=y1), P(Z=z2|X=x3,Y=y1)],
                 [P(Z=z1|X=x3,Y=y2), P(Z=z2|X=x3,Y=y2)]]])

X = pyro.sample("Z", dist.Categorical(probs=prob_X))
Y = pyro.sample("Z", dist.Categorical(probs=prob_Y))
```

```
Z = pyro.sample("Z", dist.Categorical(probs=prob_Z[X][Y]))

b.
prob_Z = tensor([[[P(Z=z1|X=x1,Y=y1), P(Z=z2|X=x1,Y=y1)],
                  [P(Z=z1|X=x2,Y=y1), P(Z=z2|X=x2,Y=y1)],
                  [P(Z=z1|X=x3,Y=y1), P(Z=z2|X=x3,Y=y1)]],

                 [[P(Z=z1|X=x1,Y=y2), P(Z=z2|X=x1,Y=y2)],
                  [P(Z=z1|X=x2,Y=y2), P(Z=z2|X=x2,Y=y2)],
                  [P(Z=z1|X=x3,Y=y2), P(Z=z2|X=x3,Y=y2)]]])

X = pyro.sample("Z", dist.Categorical(probs=prob_X))
Y = pyro.sample("Z", dist.Categorical(probs=prob_Y))
Z = pyro.sample("Z", dist.Categorical(probs=prob_Z[Y][X]))
```

6.2 Inference

There are two broad categories of causal inference questions, one is to predict effects from causes, referred to as forward causal inference, and the other is to predict causes from effects, referred to as reverse causal inference. Forward causal inference is straightforward to do through forward sampling, without the need of any inference algorithm. Forward sampling doesn't work for reverse causal inference, since there is no guarantee the sample generated matches the evidence we observed. An easy solution is rejection sampling, which rejects all the samples that do not match the evidence. But it can be very inefficient when the probability of the observed evidence is very small. An improvement is likelihood weighting, a special case of importance sampling, which samples unobserved variables like in forward sampling, but sets the observed variables to the same value as evidence, and weights each sample by its likelihood. An caveat of importance sampling is its accuracy depends on how close the proposal distribution is to the target distribution. In the case of Bayesian network, the proposal distribution is often the CPTs, and the target distribution is the posterior distribution we want to estimate. (For details of the theory of different sampling methods, see "Probabilistic Graphical Models" Chapter 12)

Pyro has implemented many inference algorithms including importance sampling, and we encourage you to experiment with them. Detailed understanding about how these methods work in Pyro is not a requirement. For the purpose of the homework, it's sufficient to have a high level understanding of the general idea and be able to use at least one of the inference algorithms in Pyro in your model.

Most of the inference algorithms implemented in Pyro are approximate inference algorithms either through sampling or variational inference. Exact inference using variable elimination is NP hard. It becomes intractable when the model is large. However, most models in homeworks are fairly small. Although inference can be done manually without much efforts in small networks, inference using Pyro is required in this question.

6.2.a

Implement forward causal inference using `pyro.condition`. Assuming you observed a person with a university degree. What is your prediction of this person's means of travel? Provide a histogram of the marginal distribution on the variable "T". (6 points)

6.2.b

Implement reverse causal inference using `pyro.condition` and an inference algorithm in Pyro. Assuming you observed a self-employed person who lives in a big city. What is your prediction of this person's age? Provide a histogram of the marginal on the variable "A". (6 points)