CSCI 3202 Introduction to
Spring 2019 Artificial Intelligence

Assignment 5

BayesNets

Aim: Develop methods to Implement and Query a Bayes Net

Issued: March 22nd

Due: April 8, 4pm

Submission: Test using the autograder provided. Submit final solution to Inginious.

Policy: Work must be done individually, refer to collaboration policy in syllabus for more details.

Introduction

In this homework, you will implement and query your own BayesNet

As in previous projects, this project includes an autograder for you to grade your solutions on your machine. Unlike in previous homeworks, the autograder is just one script, you cannot run per question. You can however comment out portions of the autograder in lines 171-181

```
python autograder.py
```

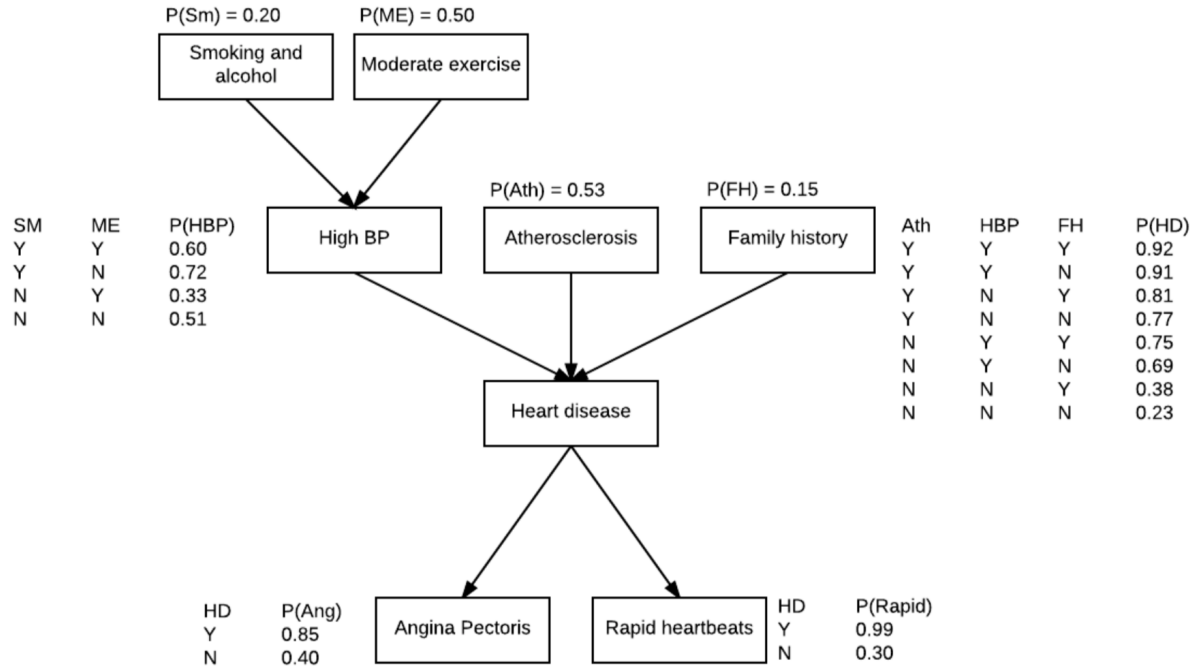
The skeleton code is available on Moodle. You will only need to edit one of the files.

A `BayesNode` class is already defined for you. As well as a Probability function `P`. You will need to use the both BayesNode Class and be compatible with the probability function in order to get full points. Each instance of the BayesNode class will have the following properties:

- variable - String name of variable
- parents - list of variable names of the parents for a node
- cpt - Conditional probability table
- values - list of possible values

We will use the following Bayesian network in order to test your code, but it should work on any network.

This network is based loosely on a study that examined heart disease risk factors in 167 elderly individuals in South Carolina. Note that this figure uses Y and N to represent Yes and No, these can be trivially converted to booleans. This is a pretty famous study and now commonly used in AI classes because of the interpretable network we can make. This is a great example of how Bayes Nets can make something complex and serious easier to understand.



Q1. [5 pts] Bayes Net Class

Use the BayesNode class provided to create a BayesNet Class.

Your class should satisfy the following constraints:

- the nodes are represented using the BayesNode class and can work with the P function for probabilities,
- your BayesNet structure keeps track of which nodes are in the Bayes net, as well as which nodes are the parents/children of which other nodes.

Skeleton codes for a class structure is provided. The methods to implement are in view of the fact that we will need to calculate some probabilities, which is going to require us to find_nodes and find_values that nodes can take on.

Q2. [8 pts] Get Probabilities

Craft a function `get_prob(Q, e, bn)` to return the **normalized** probability distribution of variable Q in Bayes net bn, given the evidence e. That is, return $P(X \mid e)$. The arguments are:

- Q is some representation is the BayesNode that you are trying to Query. This will be an object of the BayesNode class
- e is a dictionary of evidence in the format {`"NodeName":True`} is some representation of the evidence your probability is conditioned on. When given an empty argument (or None) for e, get_prob should return the marginal distribution $P(X)$.
- bn is your BayesNet object.

You may do this using the enumeration algorithm from class (pseudocode is in the slides as well as the book), or by brute force (i.e., use a few for loops). The BayesNet object has all the information that you need about nodes and relationships between nodes so your $\text{get}_{p}robfunctionknowsthesethings$.

You should return a list of probabilities the order should be relative to the order of `Q.values`, e.g. `dist[0]` should be the probability for `Q.values[0]`.

Hint: Remember that ordering is important! You'll need to make sure you are doing everything in the right order!

Q3. [4 pts] Make Prediction

Craft a function `make_prediction(Q, e, bn)` to return the the most likely value for a given query node Q in Bayes net bn, given the evidence e. The arguments are in the same format as in Question 2. You will likely want to call your `get_prob`, so make sure that function is correct first.

You should return the **value** of the most probable item.

Q4. [8 pts] Prior Sample

Make a function `prior_sample_n(bn, n)` that produces n samples of the Bayes Net bn, using the prior sampling approach discussed in class.

You should return a list of n dictionaries where each dictionary is a sample. The dictionaries should be in the format {`Node.variable: Value, Node2.variable: Value2, ...`}.

Hint: A good check of this function is to estimate a probability using your samples and compare that to what your `get_prob` function returns. With enough samples it should be pretty close.

Hint: Remember that ordering is important! You'll need to make sure you are doing everything in the right order!