Addis Ababa University(AAIT)

Probabilistic Graphical Models

Bayesian network HW-II report

Prepared by – Yared Terefe

Id – GSE/9895/16

Submitted Date - Aug 31, 2024

Submitted to - Mr Beakal Gizachew (PhD)

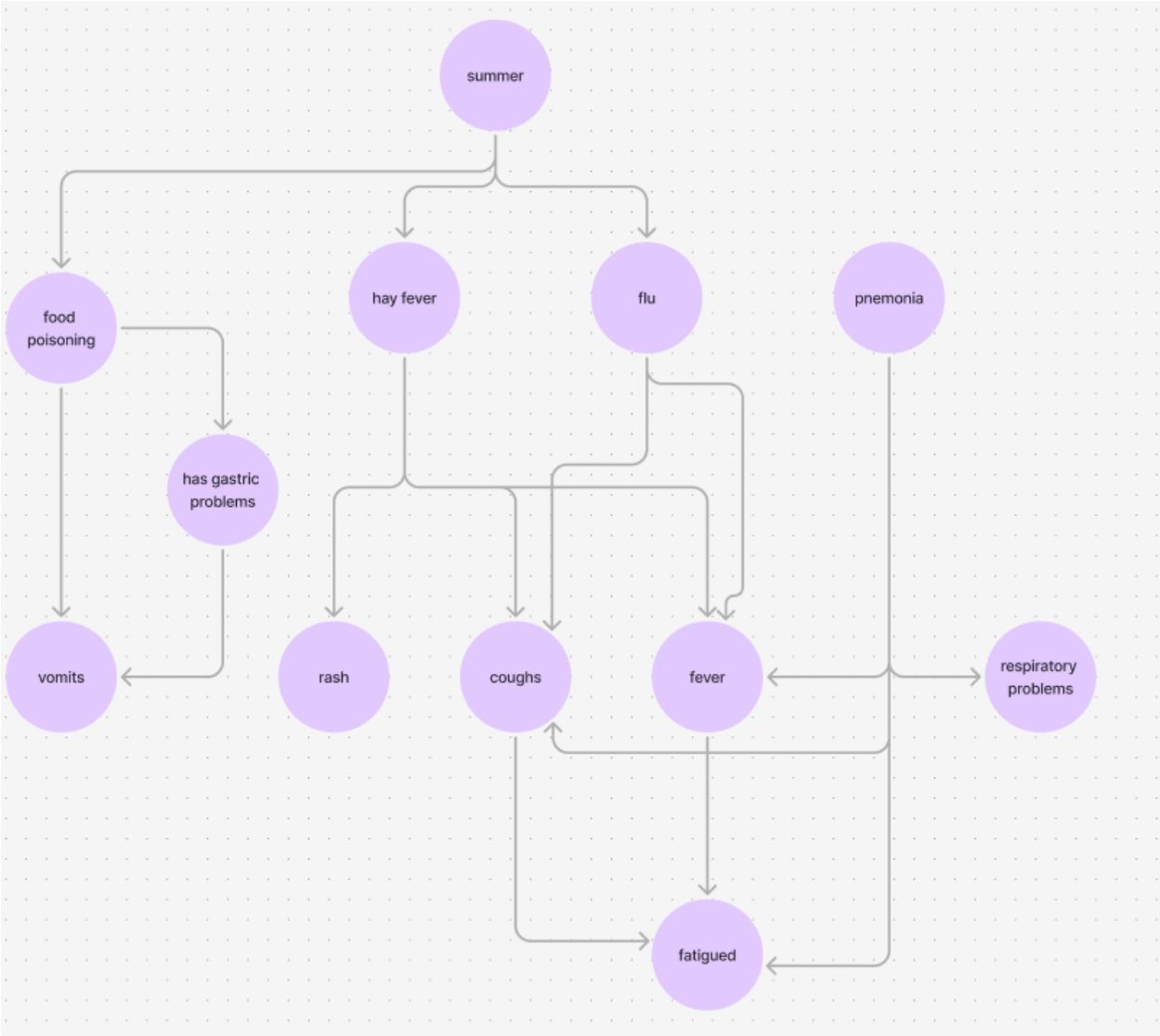
Report

Setup

I used python and the pgmpy library which is specifically designed for bayesian networks.

Then I defined the structure of the bayesian networks based on what i thought was sensible. I call it “Expert” knowledge.

Here is the bayesian network



Then I assigned random conditional probability distributions for each variable of my network. ***In total there are 84 parameters.*** After that I was able to make a full assignment and get the probability of that assignment based on my network.

This concludes the initial setup. Now we can get to real parameter calculation.

Parameter calculation

Here initially I only counted positive occurences of my target variables based on the parents if they have any, but that was complicated to get the probabilities given the parents of many of my variables which depended on parents. So the final decesion was to directly count positive(1) and negative(0) occurences for each of my variables.

Steps

1. Define count variables for all of my nodes in my network.
    - If the node has no parent then use one variable for 0 and 1 counts
      Eg - cpd_IsSummerTotal = [0]
      cpd_IsNotSummerTotal = [0]
    - If the node has a single parent then 2 variables for counting negative and positive each
    - If 2 parents then 8 total
2. Now loop through the JPD and get the binary representation of each values
3. Counting logic
    - If the node has no parents directly count it's occurrence
    - If the node has one parent, then use this reference table where to add count
      [both parent and child false, parent true child false]
      [parent false and child true, both true]
    - Use similar logic for more parents
4. Now that we have the counts, we can find the actual probability distributions.
    - For no parent nodes it's easy. The cpd will be
      [negative count/ total count, positive count/ total count]
    - For single parent the CPD should be like this

      [(child false| parent false), (child false| parent true)]
      [(child true|parent false), (child true| parent true)]

    - The counts we did initially are more like an "and" count of each, so to get the first value of the cpd for example
      Probability of child false and parent false/ probability of parent false
    - A clearer description in the comments of code can be found

Then sample of the conditional probability distributions found.

IsSummer(0)	0.0528075
IsSummer(1)	0.947192

IsSummer	IsSummer(0)	IsSummer(1)
HasFoodPoisoning(0)	0.8080205449261898	0.6263598920607868
HasFoodPoisoning(1)	0.19197945507381023	0.3736401079392132

IsSummer	IsSummer(0)	IsSummer(1)
HasHayFever(0)	0.8218350546483424	0.6489845192444255
HasHayFever(1)	0.17816494535165767	0.3510154807555745

Please check inside notebook code for all of the CPDS generated.

Validation

So for this in general the plan is to generate jpd from our model and then compare them with the L1 distance metrics

Steps

1. Make function to calculate l1 distance
2. Generate jpd for calculated model, and the random model
3. Compare amongst all the 3 jpds.

In the output of the code seems like the random model is closer to the true joint probability distribution with a lower value of 1.91

Rechecked the logic of my calculation many times to check for errors but I was unsuccesful in finding any. But since the code is hacked together with ideas that came to me as developing it's very error prone and I suspect a better way of generating the CPD could reduce the likelihood of the errors. Or maybe the structure I dreamed up was just a dream and not close to the actual probability distribution. This last theory is further strengthened in the results of the queries shown later.

Querying

Defining the querying function

The function accepts a joint probability distribution, evidence variables in a form of key-value pair dictionary, and an array of the variables to be estimated.
Steps

1. Firstly get the evidence variables and their values, then filter out the jpd values with the ones that only correspond to the evidence. After this operation we get a variable evidence_jpd which contains the part of the jpd that corresponds to the evidence, meaning other values are ignored.
2. Loop through all the query variables, and sum over the probabilities inside the evidence_jpd where our query variable is true, and then divide it by the length of the evidence_jpd (it's already filtered out to include only values where evidence matches)
3. Now we have the marginal probability.

We can see results of our query variables in the notebook output. Some probabilities seem to match up while the others are fairly different.

These are the interesting ones I found

vomits and gastric problems when has food poisoning-
{'Vomits': 0.9721520756412543, 'HasGastricProblems': 1.0}
true vomits and gastric problems when has food poisoning-
{'Vomits': 0.09782844465389545, 'HasGastricProblems': 0.24}

Seems like with my model vomitting is all but assured when the person has food poisoning but in the true probability distribution it seems vomiting is less than 10% of a chance, while having gastric problems is around 1/4th

This shows that my model structure may have been misled by my incomplete knowledge of the domain, because vomiting when having food poisoning has at least more than 50% chance in my head, which is far from the truth.

Another interesting example is
chance of pneumonia when he's fatigued- {'HasPneumonia': 0.19}
true chance of pneumonia when he's fatigued- {'HasPneumonia': 0.13}
he's fatigued but also has fever and coughs- {'HasPneumonia': 0.20}
true he's fatigued but also has fever and coughs- {'HasPneumonia': 0.20}

Here the probabilities aren't too far off from eachother but in my model, when the person is fatigued, but also has fever and coughs it should have meant the probability of having pneumonia should have decreased, because having fever and cough should *Explain away* some of the chance of the problem beng pneumonia.