**University of Missouri-Columbia / College of Engineering**

**CS 8750: Artificial Intelligence II**

****

**Programming Assignment #2**

Chanmann Lim  
Adil A. Al-Azzawi  
Fernando Torre

# Introduction

In our previous assignment, we implemented two discrete Bayesian networks to tackle the problem of determining a person’s state of drunkenness from already established conditional probabilities tables (CPTs) however in this task those CTPs have to be learned from the measured data. In dealing with real value data we cannot assign the probability for each data-point not only that the probabilities for each measurement is too small but also listing the probabilities for all real value data-points is simply impossible. In this regard, two proposed solutions appeal to be reasonable: 1) Discretization to group the real value data into ranges (for Bayesian network 1 and 2), and 2) to assume that each independent observation comes from Gaussian (for Bayesian network 3 and 4) then construct the CPTs or parameters of the Gaussian distribution from the given datasets. We also found that the conditional probability differed appreciably from the prior probability for all networks, suggesting an imperfect model.

In this assignment, we will modify and analyze our discrete networks to obtain an exact measurement of the accuracy. We will then create a continuous version of our networks, which we will similarly analyze and for accuracy. We conclude with a comparison of these accuracies.

# BN#1

Our first network, from our previous assignment, is a discrete naïve Bayes network where breathing rate, heart rate, and skin temperature are given a single cause: the state of drunkenness. This network takes the continuous values measured and discretizes them into High, Medium, or Low using the mechanism described in the next section.

## Learning program

### Formulas of Maximum Likelihood

### Program code

**BN1.m**

%% Learn parameters of the CPDs

P\_Pd = [sum(Pd == 1)/length(Pd); sum(Pd == 0)/length(Pd)];

P\_Xb\_given\_Pd = BN1.CPT(Xb, Pd);

P\_Xh\_given\_Pd = BN1.CPT(Xh, Pd);

P\_Xt\_given\_Pd = BN1.CPT(Xt, Pd);

**BN1.CPT.m**

function [ cpt ] = CPT ( x , Pd )

% BN1.CPT - construct CPT for a random variable in BN 1

% learn CPT parameters using Maximum Likelihood estimator

%

% Input:

% x - discrete random variable. Domain = {H, M, L}

% Pd - evidence variable 'drink'

%

% Output:

% cpt - conditional probability table of the random variable given the

% evidence drink.

% eg.

% Xb

% Pd | H | M | L |

% 1 | | | |

% 0 | | | |

% Note:

% The elements/rows of evidence must correspond to x's

%

domain = ['H', 'M', 'L'];

cpt = zeros(2, length(domain));

Pd\_is\_1 = sum(Pd == 1);

Pd\_is\_0 = sum(Pd == 0);

% count(x = {H, M, L} , pd = {1, 0})

% likelihood = -----------------------------------

% count(pd = {1, 0})

for i = 1:length(domain)

x\_i = Pd(x == domain(i));

cpt(1, i) = sum(x\_i == 1) / Pd\_is\_1;

cpt(2, i) = sum(x\_i == 0) / Pd\_is\_0;

end

end

### Parameters learned

#### Dataset 1001

*P(Xb|Pd)*

|  |  |  |  |
| --- | --- | --- | --- |
| Drink | H | M | L |
| 1 | 0 | 0.8824 | 0.1176 |
| 0 | 0.16 | 0.6933 | 0.1467 |

#### Dataset 1004

*P(Xb|Pd)*

|  |  |  |  |
| --- | --- | --- | --- |
| Drink | H | M | L |
| 1 | 0.0588 | 0.8235 | 0.1176 |
| 0 | 0.1067 | 0.8933 | 0 |

## Prediction program

### Formulas

#### Dataset 1001

#### Dataset 1004

### Program code

**BN1.m**

evidence = [Xb Xh Xt];

data\_size = size(evidence, 1);

prediction = zeros(data\_size, 1);

% prediction

model = BN1.model(P\_Pd, P\_Xb\_given\_Pd, P\_Xh\_given\_Pd, P\_Xt\_given\_Pd);

for i = 1:data\_size

prediction(i, 1) = model.predict(1, evidence(i, :));

end

**BN1.model.m**

classdef model

% MODEL - BN1 prediction

% Compute probability of 'Pd' given the evidences

properties

P\_Pd;

P\_Xb\_given\_Pd;

P\_Xh\_given\_Pd;

P\_Xt\_given\_Pd;

end

methods

%

% Initializer

%

function obj = model( ...

P\_Pd, P\_Xb\_given\_Pd, P\_Xh\_given\_Pd, P\_Xt\_given\_Pd)

obj.P\_Pd = P\_Pd;

% Add total of probability as the last column

obj.P\_Xb\_given\_Pd = [P\_Xb\_given\_Pd ones(2,1)];

obj.P\_Xh\_given\_Pd = [P\_Xh\_given\_Pd ones(2,1)];

obj.P\_Xt\_given\_Pd = [P\_Xt\_given\_Pd ones(2,1)];

end

%

% Compute P(pd|e)

%

function P = predict(obj, pd, e)

p\_pd\_is\_1 = obj.p\_pd(1);

p\_xb\_given\_pd\_is\_1 = obj.p\_x\_given\_pd(e(1), 1, obj.P\_Xb\_given\_Pd);

p\_xh\_given\_pd\_is\_1 = obj.p\_x\_given\_pd(e(2), 1, obj.P\_Xh\_given\_Pd);

p\_xt\_given\_pd\_is\_1 = obj.p\_x\_given\_pd(e(3), 1, obj.P\_Xt\_given\_Pd);

p\_pd\_is\_0 = obj.p\_pd(0);

p\_xb\_given\_pd\_is\_0 = obj.p\_x\_given\_pd(e(1), 0, obj.P\_Xb\_given\_Pd);

p\_xh\_given\_pd\_is\_0 = obj.p\_x\_given\_pd(e(2), 0, obj.P\_Xh\_given\_Pd);

p\_xt\_given\_pd\_is\_0 = obj.p\_x\_given\_pd(e(3), 0, obj.P\_Xt\_given\_Pd);

P\_normalized = [ ...

p\_pd\_is\_1 \* p\_xb\_given\_pd\_is\_1 \* p\_xh\_given\_pd\_is\_1 \* p\_xt\_given\_pd\_is\_1 ...

p\_pd\_is\_0 \* p\_xb\_given\_pd\_is\_0 \* p\_xh\_given\_pd\_is\_0 \* p\_xt\_given\_pd\_is\_0

];

P = P\_normalized(obj.pd\_row(pd)) / sum(P\_normalized);

end

end

%

% private functions

%

methods(Access = private)

%

% Lookup P(x|pd) probability from CPT

%

function P = p\_x\_given\_pd(obj, x, pd, cpt)

P = cpt(obj.pd\_row(pd), obj.x\_column(x));

end

%

% Lookup P(pd) from CPT

%

function P = p\_pd(obj, pd)

P = obj.P\_Pd(obj.pd\_row(pd));

end

%

% Map Pd values {1 0} to row [1 2] in the CPT

%

function r = pd\_row(~, pd)

pds = [1 0];

rows = [1 2];

r = rows(pds == pd);

end

%

% Map the values {H, M, L , -} to column [1 2 3 4]

%

function c = x\_column(~, x)

xs = ['H' 'M' 'L' '-'];

columns = [1 2 3 4];

c = columns(xs == x);

end

end

end

## Confusion Matrices

### Dataset 1001

|  |  |
| --- | --- |
| 2 | 0 |
| 15 | 75 |

Prediction accuracy = 83.7%

### Dataset 1004

|  |  |
| --- | --- |
| 7 | 4 |
| 8 | 53 |

Prediction accuracy = 83.3%

# BN#2

Our second network, from our previous assignment, is a discrete complex Bayes network where breathing rate, heart rate, and skin temperature are given two single cause: the state of drunkenness and the state of ambulation. This network takes the continuous values measured and discretizes them into High, Medium, or Low using the mechanism described in section 1.1.

## Learning program

### Formulas of Maximum Likelihood

### Program code

### Parameters learned

#### Dataset 1001

#### Dataset 1004

## Prediction program

### Formulas

#### Dataset 1001

#### Dataset 1004

### Program code

#### Dataset 1001

#### Dataset 1004

## Confusion Matrices

### Dataset 1001

### Dataset 1004

# BN#3

## Learning program

### Formulas of Maximum Likelihood

### Program code

### Parameters learned

#### Dataset 1001

#### Dataset 1004

## Prediction program

### Formulas

#### Dataset 1001

#### Dataset 1004

### Program code

#### Dataset 1001

#### Dataset 1004

## Confusion Matrices

### Dataset 1001

### Dataset 1004

# BN#4

## Learning program

### Formulas of Maximum Likelihood

### Program code

### Parameters learned

#### Dataset 1001

#### Dataset 1004

## Prediction program

### Formulas

#### Dataset 1001

#### Dataset 1004

### Program code

#### Dataset 1001

#### Dataset 1004

## Confusion Matrices

### Dataset 1001

### Dataset 1004

# Conclusions

Four Bayesian networks were implemented: two discrete, two continuous. Of each, the first was a naïve Bayes network and the latter was a more complex two-cause model. The conditional probabilities were calculated for all four networks, of which 4 were found to be equal to zero.

When comparing the accuracies, we found that… We suspect this is due to the zero-probabilities.