**AIHW6 – 2017**

**Berat Barakat**

1) We are given the following independence statements about random variables

A, B, C, D, E, F, G, H:

1 I({A,B,C},{D,E}|{})

2 I({A,B,C},{F,G,H}|{})

3 I({D,E},{F,G,H}|{})

4 I({A},{C}|{B})

5 I({F},{H}|{})

6 not I({A},{C}|{})

7 not I({D},{E}|{})

8 not I({F},{H}|{G})

a) Construct a Bayes network that is consistent with the

above set of independence statements.

ANSWER:

1,2,3 and the fact that there is no other node than them, implies that there are no edges between abc and de and fgh. They are 3 separate components.

4 implies that there is no edge between a and c

6 implies that there is a path between a and c. This path should be through the only remaining node in this group which is b. And b must block the path and it is in evidence, so b is a passthrough or diverging node in the path between a and c

5 implies that there is no edge between f and h

8 implies that there is a path between f and h. This path should be through the only remaining node in this group which is g. And g mustn’t block the path and it is not in evidence so g is a converging node in the path between f and h

7 implies that there is a path between d and e. Since there is no other node in this group, this path is a direct edge between d and e, and can be in either direction.

b) Is the answer above unique? If not, what is the set of possible BNs

that is consistent with all the statements?

ANSWER:

As explained in the above there are multiple possible directions for the edges. fgh has one possibility in which g is a converging node between f and h. acb has three possibilities, in two of them b is a passthrough and in one of them b is a diverging node. de has two possibilities, a direct edge in either direction between them.

2) Consider the following variables and their immediate causes (parents):

variable parents

-------------------------------

A none

B none

C A B

D A

E B

F D

G D E

1. Is this network a poly-tree?

ANSWER:

It is not a polytree. Counter example the undirected cycle: acbegda

1. Is this network (directed-path) singly connected?

ANSWER:

No, there is no two directed paths between any two nodes.

c) Determine the truth of the following independence

statements (using d-separation):

ANSWER:

1. I({D}, {E} | {})

Yes, g blocks dge and c blocks dacbe

1. I({D}, {E} | {A})

Yes, a blocks dacbe and g blocks dge

1. I({D}, {E} | {A, G})

No, dge is not blocked because g is in evidence

1. I({B}, {F} | {C})

No, bcadf is not blocked because c is in evidence and a,d not in evidence

1. I({B}, {F} | {D})

Yes, d blocks bcadf and g blocks begdf

1. I({B}, {F} | {A, G})

No, begdf is not blocked because g is in evidence and e,d not in evidence

d) Assume that the nodes are binary-valued.

Suppose that P(A=true) = 0.2, P(B=true)= 0.3

P(C=true|A=true,B=true) = 0.9, P(C=true|A=false,B=true) = 0.2

P(C=true|A=true,B=false) = 0.5, P(C=true|A=false,B=false) = 0.1

P(E=true|B=true) = 0.8, P(E=true|B=false) = 0.1

P(G=true|D=true, E=true) = 0

ANSWER:

Preprocessing: Drop barren node f, g and c, then d becomes barren, drop d.

In the new topology:

e and a are d-seperated then P(e,a|b)=P(e|b).P(a|b)

a and d are d-seperated then P(a|b)=P(a)

P(e|b).P(a|b)=P(e|b)P(a)=0.8 x 0.2=0.16

3) For the blocks world as defined in class, show how POP constructs the plan

(that is, show all partial plans from the initial plan until a correct

plan is reached) for initial state:

E

D

A

C B

------------

and the goal state containing: On(D,A) and On(A,C)

ANSWER:

Initial state has two dummy actions:

Start: Percond:null

Effect: on(c,t), on(b,t), on(a,b), on(d,a), on(e,d) c(c), c(e)

End: Precond: on(d,a), on(a,c)

Effect: null

Links: start--on(d,a)-->end

Constraints: start>end

Explanation: This state is not a solution, take an unsatisfied precondition, say on(a,c), it is not satisfied in current state, add an action.

Step 1: put(a,c)

Precond: c(c), c(a), on(a,x){x/b}

Effect: ~c(c), on(a,c), c(x){x/b}, ~on(a,x){x/b}

Links: step1--on(a,c)-->end

Start--c(c),on(a,b)-->step1

Constraints: start>step1>end

Start>put(a,c)>end

Explanation: This state is not a solution, take an unsatisfied condition say c(a), add an action to satify it.

Step 2: put(d,t)

Precond: c(d), on(d,x){x/a}

Effect: c(x){x/a}, on(d,t), ~on(d,x){x/a}

Links: Start--on(d,a)-->step2

step2--c(a)-->step1

Constraints: start>step2>step1>end

Start>put(d,t)>put(a,c)>end

Explanation: step2 clobbers with end because on(d,a). step2 can not be after end. So we will need a new action to satify on(d,a) after step2 before end.

This state is not a solution, take an unsatisfied condition say c(d), add an action to satify it.

Step 3: put(e,t)

Precond: c(e), on(e,x){x/d}

Effect: c(x){d}, on(e,t), ~on(e,x){x/d}

Links: Start--on(e,d)-->step3

Start--c(e)-->step3

Step3--c(d)-->step2

Constraints: start>step3>step2>step1>end

Start>put(e,t)>put(d,t)>put(a,c)>end

Explanation: This state is not a solution, take the unsatisfied condition on(d,a) because it clobbers with action put(d,t), add an action after put(d,t) to satify it.

Step 4: put(d,a)

Precond: c(d), c(a), on(d,x){x/t}

Effect: on(d,a), ~c(a)

Links: step3--c(d)-->step4

Step2--c(a), on(d,t)-->step4

Step4--on(d,a)-->end

Constraints: start>step3>step2>step1>step4>end

Start>put(e,t)>put(d,t)>put(a,c)>put(d,a)>end

Explanation: This state is a solution. Step 4 must be after step 1 because it clobbers on c(a) effect of step2.

4)

a) What is your optimal policy, in order to maximize your sum of gains for the first 3 years (i.e. this year, next year, and the year after the next year)?

Considering the following assumptions:

* All gain is, what is gained from factory which is 1000M/year

A: Don’t meet Lim and don’t advertise for 3 years.

U(A)=1000+1000+1000=3000

B: Meet Lim and advertise for 3 years

U(B) =+1000-300-100+0.5\*500-0.001(1000)

+1000-100-100+0.5\*500-0.001(1000+20000)

+1000-100-100+0.5\*500-0.001(1000+20000)

=3000-500-300+750-43

=2907

Optimal policy is A, don’t meet Lim and don’t advertise for 3 years.

b) You can now also purchase an additional market survey, that will give you perfect information on whether the advertizing campain will work, for 50 MMU. Does your optimal policy differ?

U(B) =+1000-300-50+0.5(-100+500)-0.001(1000)

+1000-100-50+0.5(-100+500)-0.001(1000+20000)

+1000-100-50+0.5(-100+500)-0.001(1000+20000)

=3000-500-150+600-43

=2907

No, the optimal policy does not change.

c) Repeat a and b above for sum of gains for 3 years under a discounted reward factor of 0.1, that is, next year is only worth 10% of what this year is worth, etc.

U(A) =3000

U(B) =+1000-300-100+0.5\*500-0.001(1000)

+1000-100-100+0.5\*500-0.1\*0.001(1000+20000)

+1000-100-100+0.5\*500-0.01\*0.001(1000+20000)

=3000-500-300+750-(1+2.1+0.21)

=2946.69

Optimal policy is still A, don’t meet Lim and don’t advertise for 3 years.

With the survey:

U(B) =+1000-300-50+0.5(-100+500)-0.001(1000)

+1000-100-50+0.5(-100+500)-0.1\*0.001(1000+20000)

+1000-100-50+0.5(-100+500)-0.01\*0.001(1000+20000)

=3000-500-150+600-(1+2.1+0.21)

=2946.69

No, the optimal policy does not change.

5) Consider the following Harassed Citizen problem instance.

The (undirected) graph has 7 vertices, as follows:

(S, I1, I2, V1, V2, V3, G)

The edges are as follows:

(S, V1), weight 10.

(V1, V2), weight 1.

(V2, V3), weight 1.

(V3, G), weight 1.

(S, G), weight 100.

(S, I1), weight 1.

(S, I2), weight 2.

(I1, V2), weight 1000.

(I2, V3), weight 1000.

S is the start vertex, V2 may contain a key with a probability of 0.5,

V3 may contain a lock of the same type with probablity 0.8, and G is a goal vertex.

No other locks or keys are possible in this graph.

b1) Formalize the above problem as a (belief-state) MDP, i.e.

what are the states, the transition probability

function, and the reward function? (He the cost of traversing an edge is equal to it weight).

ANSWER:

State variable is defined as (a,k,l,c) where

a: agent location with domain={s,v1,v2,v3,g,i1,i2}

k: key existence with domain={t,f,u}

l: lock existence with domain={t,f,u}

c: the fact of carrying key with domain={t,f,u}

Transition probabilities:

(x,u,y,z)🡪(x,t,y,z)=0.5

(x,u,y,z)🡪(x,f,y,z)=0.5

(x,y,u,z)🡪(x,y,t,z)=0.8

(x,y,u,z)🡪(x,y,f,z)=0.2

(x,u,u,z)🡪(x,t,t,z), (x,t,f,z), (x,f,t,z), (x,f,f,z) are the proper multiplications of the above ones but there can’t be such case in the given graph.

All other allowed transitions are with probability 1.

Action is traversing an edge to a node say ‘a’, then the reward for traversing from a state to a node ‘a’ is defined as

R(s,a)=R((x,y,z,t),a)=-weight(edge(x,a))

b2) Find the optimal policy. You may assume that the agent starts

at vertex S in order to save some computation.

ANSWER:

We can set the following utilities:

U(g,x,y,z)=0

U(i1,t,u,f)=-14

U(i1,f,u,f)= need to calculate

U(i1,t,t,f)=-14

U(i1,f,t,f)=-101

U(i2,u,f,f)=-15

U(i2,u,t,f)= need to calculate

U(i2,f,f,f)=-15

U(i2,f,t,f)=-102

U(s,f,u,f)= need to calculate

U(s,u,t,f)= need to calculate

U(v1,t,u,f)=-3

U(v1,f,u,f)= need to calculate

U(v1,t,t,f)=-3

U(v1,f,t,f)=-110

U(v2,f,f,f)=-2

U(v2,f,t,f)=-111

U(v1,f,u,f) =-1+0.2\*U(v2,f,f,f)+0.8\*U(v2,f,t,f)

=-1+0.2\*-2+0.8\*-111

=-90.2

U(s,f,u,f) =max[U(v1,f,u,f),-2+0.2\*U(i2,f,f,f)+0.8\*U(i2,f,t,f)]

=max[-90.2, -2+0.2\*-15+0.8\*-102]

=max[-90.2, -86.6]

=-86.6

U(i1,f,u,f) =-1+U(s,f,u,f)

=-1-86.6

=-87.6

U(s,u,t,f) =max[(-10+0.5\*u(v1,t,t,f)+0.5\*u(v1,f,t,f)),

(-1+0.5\*u(i1,t,t,f)+0.5\*u(i1,f,t,f)]

=max[(-10+0.5\*-3+0.5\*-110), (-1+0.5\*-14+0.5\*-101)]

=max[-66.5, -58.5]

=-58.5

U(i2,u,t,f) =-2+u(s,u,t,f)

=-2-58.5

=-60.5

Starting in s node agent need to decide between i1,i2,v1 and g.

So we need to calculate the following:

U(s,u,u,f) =max[(-1+0.5\*u(i1,t,u,f)+0.5\*u(i1,f,u,f)),

(-2+0.8\*u(i2,u,t,f)+0.2\*u(i2,u,f,f)),

(-10+0.5\*u(v1,t,u,f)+0.5u(v1,f,u,f)),

(-100+0)]

=max[(-1+0.5\*-14+0.5\*-87.6)),

(-2+0.8\*-60.5+0.2\*-15),

(-10+0.5\*-3+0.5\*-90.2),

(-100+0)]

=max[51.8, 53.4, 56.6, 100]

=51.8 which corresponds to i1.

Accordingly the optimal policy is

Go to i1,

if there is a key go to s,v1,v2,v3,g

if there is no key go to s,i2

if there is no lock go to s,v1,v2,v3,g

if there is a lock go to s,g

6)

a) Construct by hand (using the greedy search using the best information

gain heuristic) a decision tree as close to consistent as possible for the following table:

input variables decision

A B C D

--------------------------------------------

2 F L 0 N

2 F H 1 Y

2 T L 0 N

3 F H 0 Y

3 T H 2 Y

3 T H 2 Y

3 T H 2 N

3 T L 0 Y

3 T H 1 Y

Search the best attribute for top level:

A=2 1Y,2N

A=3 5Y,1N

B=T 4Y,2N

B=F 2Y,1N

C=L 1Y,2N

C=H 5Y,1N

D=0 2Y,2N

D=1 2Y

D=2 2Y,1N

* A and C has the same entropy. => A=C
* B has (2Y,1N) and A has (1Y,2N) which are same. Comparing A’s (5Y,1N) with B’s (4Y,2N); both has the same number of samples but A has less entropy. So A is preferable. => A=C>B
* D has (2Y,1N) and A has (1Y,2N) which are same. D has zero entropy in (2Y). Comparing A’s (5Y,1N) with D’s (2Y,2N)needs calculation.

E(5Y,1N)=-(log(1/6)+5\*log(5/6))=2.58+1.31=3.89

E(2Y,2N)=-(2\*log(2/4)+2\*log(2/4))=4

So A is preferable. => A=C>D

* Select A.

A=2 B=T N

B=F 1Y,1N

C=L 2N

C=H Y

D=0 2N

D=1 Y

D=2 N (default value for A=2)

* Clearly C is the best attribute. Select C. And this branch is ended.

**A=2:**

**C=L: N**

**C=H: Y**

* Now we look for other branch.

A=3 B=T 4Y,1N

B=F 1Y

C=L 1Y

C=H 4Y,1N

D=0 2Y

D=1 Y

D=2 2Y,1N

* Clearly B=C. D’s (2Y,1N) seems preferable to B’s (4Y,1N), because its less number of samples but calculated for sure.
  + E(2Y,1N)=-(2log(2/3)+log(1/3))=1.17+1.58=2.75
  + E(4Y,1N)=-(4log(4/5)+log(1/5))=1.28+2.32=3.6

So D is preferable

* Select D. D=0 and D=1 branches are ended.

**A=3:**

**D=0: Y**

**D=1: Y**

* Now we look for D=2 branch.

A=3

D=2

B=T 2Y,1N

B=F 1Y (Default for A=3,D=2)

C=L 1Y (Default for A=3,D=2)

C=H 2Y,1N

* Clearly B and C same. Select B. Then we are done for B=F.

**A=3:**

**D=2:**

**B=F: Y**

* Now look for B=T branch

A=3

D=2

B=T

C=L 1Y (Default for A=3,D=2,B=T)

C=H 2Y,1N

* Now we are done for C=L

**A=3:**

**D=2:**

**B=T:**

**C=L: Y**

* Now for C=H we are not done, and there is no more attribute. This is because there is noise in the dataset. Same samples are given with different labels, see the following:

3 T H 2 **Y**

3 T H 2 **Y**

3 T H 2 **N**

* But we label it with the majority vote.

**A=3:**

**D=2:**

**B=T:**

**C=H: Y**

b) Is a more compact (fewer internal nodes) decision tree possible for the above table?

This algorithm does not gurantee the smallest tree due to its greedy nature in selecting the nodes with least possible number of branches. However in this problem we have 6 internal nodes and there is no smaller tree. This is because we have the last 3 internal nodes due to noisy data, and no way to don’t have them. And the rest three top nodes are the least possible number of nodes because we tried all attributes for the root node and none of them gave less than two branches.

7) You need to construct a 7-input neural network that has 1 output: the output should be 1 if the number of input units that are 1 is odd, and 0 otherwise.

a) Can this be done without hidden units?

No, because this problem is not linearly seperable. We need a hidden layer.

b) Show a network using threshold elements that performs

the required computation (specify the weights, too!)

Using 3 hidden units and one output unit.

All units get all inputs with a weight of 1.

Output unit additionally gets the outputs of 3 hidden units.

Thresholds:

Output unit: Threshold 0.5 so it seperates between the odd and even in the interval given by hidden layer.

First hidden unit: Threshold +1.5 and feeds into output unit with a weight of -2. So if the sum is less than 1.5, this is inactive and the output unit seperates between the sum 0 and the sum 1.

Second hidden unit: Threshold +3.5 and feeds into output unit with a weight of -2. So if the sum is less than 3.5, this is inactive and first hidden unit is active, and the output unit seperates between the sum 2 and the sum 3.

Third hidden unit: Threshold +5.5 and feeds into output unit with a weight of -2. So if the sum is less than 5.5, this is inactive and first and second hidden units are active, and the output unit seperates between the sum 4 and the sum 5.

Lastly if the sum is 6 or 7 then all the hidden units are inactive and the output unit seperates between 6 and 7.