CS4725/CS6705 - Fall 2017

Assignment # 4

SAMPLE SOLUTIONS

---

1. **(14 marks) Markov Decision Processes**

   (Based in part on Russell & Norvig, Exercise 17.10)

   Consider an MDP with three states (*A*,*B*,*C*), with rewards $R(A) = 1$, $R(B) = 2$ and $R(C) = 3$. State C is a terminal state. In states A and B, there are two possible actions: x and y. The transition model is:

   - In state A, action x moves the agent to state B with probability 0.8 and makes the agent stay in A with probability 0.2.
   - In state B, action x moves the agent to state A with probability 0.8 and makes the agent stay in B with probability 0.2.
   - In either state A or state B, action y moves the agent to state C with probability 0.1 and makes the agent stay in its current state with probability 0.9.

   (a) (10 marks) Using a discount factor of $\gamma = 0.5$, run the value iteration algorithm presented in class to calculate utilities for states A, B and C. Start by initializing $U_0(A) = 0, U_0(B) = 0, U_0(C) = 0$.

   You can stop when all utility values have changed by less than 0.03 from their values in the previous step.

   Suggestion: Show your calculations, but for each iteration, draw yourself a summary table that looks something like this:

State s	$R(s)$	Exp. util. of action x $= \sum_{s'} P(s'\|s,x)U_i(s')$	Exp. util. of action y $= \sum_{s'} P(s'\|s,y)U_i(s')$	$U_{i+1}(s) = R(s) +$ $\gamma \cdot \max(EU(x), EU(y))$
A				
B				
C				

   Note: For state C, there are no actions available, so $U_{i+1}(C)$ will always just equal $R(C)$.

   [Hint: If your total number of iterations is more than 10, then something is going wrong.]

Here is a trace of the steps that would be followed.

State s	$R(s)$	Exp. util. of action x $= \sum_{s'} P(s'\|s,x)U_i(s')$	Exp. util. of action y $= \sum_{s'} P(s'\|s,y)U_i(s')$	$U_{i+1}(s) = R(s) + \gamma \cdot \max(EU(x), EU(y))$
A	1	0.8(0) + 0.2(0) = 0	0.9(0) + 0.1(0) = 0	1 + 0.5(0) = 1
B	2	0.8(0) + 0.9(0) = 0	0.9(0) + 0.1(0) = 0	2 + 0.5(0) = 2
C	3			3

State s	$R(s)$	Exp. util. of action x $= \sum_{s'} P(s'\|s,x)U_i(s')$	Exp. util. of action y $= \sum_{s'} P(s'\|s,y)U_i(s')$	$U_{i+1}(s) = R(s) + \gamma \cdot \max(EU(x), EU(y))$
A	1	0.8(2) + 0.2(1) = 1.8	0.9(1) + 0.1(3) = 1.2	1 + 0.5(1.8) = 1.9
B	2	0.8(1) + 0.2(2) = 1.2	0.9(2) + 0.1(3) = 2.1	2 + 0.5(2.1) = 3.05
C	3			3

State s	$R(s)$	Exp. util. of action x $= \sum_{s'} P(s'\|s,x)U_i(s')$	Exp. util. of action y $= \sum_{s'} P(s'\|s,y)U_i(s')$	$U_{i+1}(s) = R(s) + \gamma \cdot \max(EU(x), EU(y))$
A	1	0.8(3.05) + 0.2(1.9) = 2.82	0.9(1.9) + 0.1(3) = 2.01	1 + 0.5(2.82) = 2.41
B	2	0.8(1.9) + 0.2(3.05) = 2.13	0.9(3.05) + 0.1(3) = 3.045	2 + 0.5(3.045) = 3.5225
C	3			3

State s	$R(s)$	Exp. util. of action x $= \sum_{s'} P(s'\|s,x)U_i(s')$	Exp. util. of action y $= \sum_{s'} P(s'\|s,y)U_i(s')$	$U_{i+1}(s) = R(s) + \gamma \cdot \max(EU(x), EU(y))$
A	1	0.8(3.5225) + 0.2(2.41) = 3.3	0.9(2.41) + 0.1(3) = 2.469	1 + 0.5(3.3) = 2.65
B	2	0.8(2.41) + 0.2(3.5225) = 2.6325	0.9(3.5225) + 0.1(3) = 3.4703	2 + 0.5(3.4703) = 3.7351
C	3			3

State s	$R(s)$	Exp. util. of action x $= \sum_{s'} P(s'\|s,x)U_i(s')$	Exp. util. of action y $= \sum_{s'} P(s'\|s,y)U_i(s')$	$U_{i+1}(s) = R(s) + \gamma \cdot \max(EU(x), EU(y))$
A	1	0.8(3.7351) + 0.2(2.65) = 3.5181	0.9(2.65) + 0.1(3) = 2.685	1 + 0.5(3.5181) = 2.7591
B	2	0.8(2.65) + 0.2(3.7351) = 2.8670	0.9(3.7351) + 0.1(3) = 3.6616	2 + 0.5(3.6616) = 3.8308
C	3			3

State s	$R(s)$	Exp. util. of action x $= \sum_{s'} P(s'\|s,x)U_i(s')$	Exp. util. of action y $= \sum_{s'} P(s'\|s,y)U_i(s')$	$U_{i+1}(s) = R(s) + \gamma \cdot \max(EU(x), EU(y))$
A	1	0.8(3.8308) + 0.2(2.7591) = 3.6165	0.9(2.7591) + 0.1(3) = 2.7832	1 + 0.5(3.6165) = 2.8083
B	2	0.8(2.7591) + 0.2(3.8308) = 2.9734	0.9(3.8308) + 0.1(3) = 3.7477	2 + 0.5(3.7477) = 3.8739
C	3			3

State s	$R(s)$	Exp. util. of action x $= \sum_{s'} P(s'\|s,x)U_i(s')$	Exp. util. of action y $= \sum_{s'} P(s'\|s,y)U_i(s')$	$U_{i+1}(s) = R(s) + \gamma \cdot \max(EU(x), EU(y))$
A	1	0.8(3.8739) + 0.2(2.8083) = 3.6608	0.9(2.8083) + 0.1(3) = 2.8275	1 + 0.5(3.6608) = 2.8304
B	2	0.8(2.8083) + 0.2(3.8739) = 3.0214	0.9(3.8739) + 0.1(3) = 3.7865	2 + 0.5(3.7865) = 3.8933
C	3			3

At this point, all utility values have changed by less than 0.03 from the values in the previous step, and so we will stop. The final utilities are $u(A) = 2.8304$, $u(B) = 3.8933$ and $u(C) = 3$.

(b) (4 marks) What is the optimal policy for the agent? (Based on your final utilities from part (a), what is the optimal action in state A and what is the optimal action in state B?)

In state A, the optimal action is x because:

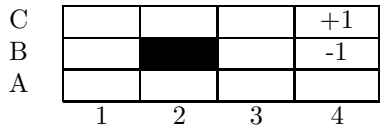
- The expected utility of x is $0.8(3.8933) + 0.2(2.8304) = 3.6807$
- The expected utility of y is $0.9(2.8304) + 0.1(3) = 2.8474$

In state B, the optimal action is y because:

- The expected utility of x is $0.8(2.8304) + 0.2(3.8933) = 3.0430$
- The expected utility of y is $0.9(3.8933) + 0.1(3) = 3.8040$

2. **(6 marks) Partially-Observable MDPs** (POMDPs)

In the POMDP example we covered in class, we considered the grid world problem (pictured below). We did not know what state we were in at any time, but we did have the added information that there was a sound coming from location $4C$ that can be heard if and only if we are within two squares (Manhattan distance) of $4C$.



We did not know our initial location, but since we could not hear the sound, we determined that we must be in state 1A, 2A, 3A, 1B or 1C, and we assumed equal probability for each state (0.2).

After attempting to move to the right, and observing that we **still** could not hear the sound, we used the belief state update algorithm to recalculate the probability that we were in each cell. Our new probabilities were:

State	Probability
1A	0.061
2A	0.303
3A	0.273
1B	0.303
1C	0.061

Starting from the belief state in the table immediately above this sentence, suppose that we try a **second** time to move to the right. After this action, we are now able to hear the sound. Use this information to recalculate our belief state.

Additional notes:

- Assume that we are following the same transition model that we discussed in class. When we attempt to take an action, there is a probability of 0.8 that we will actually go in the intended direction, and a probability of 0.1 that we will go in each of the two directions that are 90 degrees off from the intended direction. If an action causes us to hit a wall, then we simply stay in our current state.
- Assume that the sound (or lack of sound) is the only feedback we get. We are not even aware of whether we hit a wall after making a move.

Given that we can now hear the sound, the only possible states that we could be in are 4A, 3B, 4B, 2C, 3C, 4C.
However, given our beliefs about where we could have been before making the move, we can eliminate 4B, 3C and 4C. (There is no way that we could have reached those states from our possible previous states in one move.)
The probabilities for the remaining possible states (4A, 3B, 2C) are calculated as follows:

$$\begin{aligned} b'(4A) &= \alpha P(\texttt{sound}|4A) \sum_s P(4A|s, \texttt{right}) b(s) \\ &= \alpha(1)[P(4A|3A, \texttt{right}) b(3A)] \\ &= \alpha(1)[(0.8)(0.273)] \\ &= 0.2184\alpha \end{aligned}$$

$$\begin{aligned} b'(3B) &= \alpha P(\texttt{sound}|3B) \sum_s P(3B|s, \texttt{right}) b(s) \\ &= \alpha(1)[P(3B|3A, \texttt{right}) b(3A)] \\ &= \alpha(1)[(0.1)(0.273)] \\ &= 0.0273\alpha \end{aligned}$$

$$\begin{aligned} b'(2C) &= \alpha P(\texttt{sound}|2C) \sum_s P(2C|s, \texttt{right}) b(s) \\ &= \alpha(1)[P(2C|1C, \texttt{right}) b(1C)] \\ &= \alpha(1)[(0.8)(0.061)] \\ &= 0.0488\alpha \end{aligned}$$

Since these probabilities must add up to 1, we know $0.2184\alpha + 0.0273\alpha + 0.0488\alpha = 1$, and so $\alpha = \frac{1}{0.2945}$, which means:

$b'(4A) = \frac{0.2184}{0.2945} = 0.7416$

$b'(3B) = \frac{0.0273}{0.2945} = 0.0927$

$b'(2C) = \frac{0.0488}{0.2945} = 0.1657$

3. **(12 marks) Santa Claus is coming to town...**

You'd better watch out
You'd better not cry
You'd better not pout
I'm telling you why
Santa Claus is coming to town

He's making a list
And checking it twice
He's gonna find out who's naughty and nice
Santa Claus is coming to town

Santa Claus is gonna find out who's naughty and nice. And he's going to use decision tree learning to do it! Consider the following set of training examples:

Person	WatchedOut?	Cried?	Pouted?	Naughty or Nice?
Quinton	F	F	T	Naughty
Robbie	T	F	F	Nice
Steve	T	F	T	Nice
Trevor	T	T	F	Naughty
Ursula	F	F	F	Nice
Violet	T	F	F	Nice
Wanda	T	T	T	Naughty
Xavier	F	T	T	Naughty
Yolanda	F	T	F	Naughty
Zoe	T	F	T	Nice

Note that there are 5 positive (nice) examples and 5 negative (naughty) examples, and so our total entropy initially is $B(\frac{5}{10}) = -\frac{5}{10}\log_2\frac{5}{10} - \frac{5}{10}\log_2\frac{5}{10} = 1$.

(a) For each attribute A (WatchedOut?, Cried? and Pouted?), use the formulas discussed in the book and in class to calculate $Remainder(A)$ and $Gain(A)$. Show your work.

Choose the attribute with the largest gain and use that attribute as the splitting attribute at the root of your decision tree.

Some possibly helpful information:

- To use a calculator to calculate $\log_2(x)$ for some value x, compute $\frac{\ln(x)}{\ln(2)}$ or $\frac{\log_{10}(x)}{\log_{10}(2)}$.
- Note that $B(0) = B(1) = 0$. If you use this rule, you will not run into the problem of trying to take the log of 0.

(b) Repeat part (a) as needed, adding the most useful remaining attribute test along each branch of the tree until you have a complete decision tree. Note that, when applying the formulas at each node in the tree, the values of p and n are the numbers of positive and negative examples remaining at that node. You should compute the entropy at that node before performing the tests to determine which splitting attribute to use next.

(a) When considering possible splitting attributes for the root of the tree,

$$\begin{aligned}
Gain(WatchedOut) &= B(\tfrac{5}{10}) - Remainder(WatchedOut) \\
&= 1 - (\tfrac{6}{10}B(\tfrac{4}{6}) + \tfrac{4}{10}B(\tfrac{1}{4})) \\
&= 1 - (0.6(-\tfrac{4}{6}\log_2(\tfrac{4}{6}) - \tfrac{2}{6}\log_2(\tfrac{2}{6})) + 0.4(-\tfrac{1}{4}\log_2(\tfrac{1}{4}) - \tfrac{3}{4}\log_2(\tfrac{3}{4}))) \\
&= 1 - (0.6(0.3900 + 0.5283) + 0.4(0.5 + 0.3113)) \\
&= 1 - (0.5510 + 0.3245) \\
&= 0.1245
\end{aligned}$$

$$\begin{aligned}
Gain(Cried) &= B(\tfrac{5}{10}) - Remainder(Cried) \\
&= 1 - (\tfrac{4}{10}B(\tfrac{0}{4}) + \tfrac{6}{10}B(\tfrac{5}{6})) \\
&= 1 - (0.4(0) + 0.6(-\tfrac{5}{6}\log_2(\tfrac{5}{6}) - \tfrac{1}{6}\log_2(\tfrac{1}{6}))) \\
&= 1 - (0.4(0) + 0.6(0.2192 + 0.4308)) \\
&= 1 - (0 + 0.39) \\
&= 0.61
\end{aligned}$$

$$\begin{aligned}
Gain(Pouted) &= B(\tfrac{5}{10}) - Remainder(Pouted) \\
&= 1 - (\tfrac{5}{10}B(\tfrac{2}{5}) + \tfrac{5}{10}B(\tfrac{3}{5})) \\
&= 1 - (0.5(-\tfrac{2}{5}\log_2(\tfrac{2}{5}) - \tfrac{3}{5}\log_2(\tfrac{3}{5})) + 0.5(-\tfrac{3}{5}\log_2(\tfrac{3}{5}) - \tfrac{2}{5}\log_2(\tfrac{2}{5}))) \\
&= 1 - (0.5(0.5288 + 0.4422) + 0.5(0.4422 + 0.5288)) \\
&= 1 - (0.4855 + 0.4855) \\
&= 0.029
\end{aligned}$$

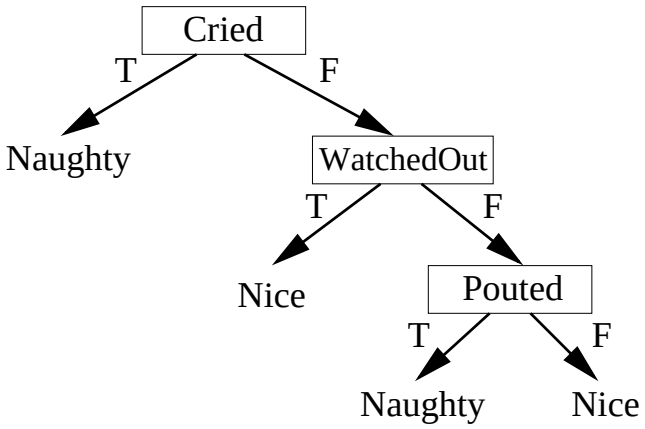
Note that $Cried$ has the highest information gain and so we should split according to that attribute. Note that if $Cried = true$, we can classify any example as $Naughty$ and so there is no need to further subdivide that branch. However, if $Cried = false$, we are left with 5 $Nice$ examples and 1 $Naughty$ example, and so we must consider whether to split according to the $WatchedOut$ or $Pouted$ attribute.

(b) Since there are now 5 positive examples and 1 negative example, the remaining entropy is now $B(\frac{5}{6}) = -\frac{5}{6}\log_2\frac{5}{6} - \frac{1}{6}\log_2\frac{1}{6} = 0.65$. For the next split,

$$\begin{aligned} Gain(WatchedOut) &= B(\tfrac{5}{6}) - Remainder(WatchedOut) \\ &= 0.65 - (\tfrac{4}{6}B(\tfrac{4}{4}) + \tfrac{2}{6}B(\tfrac{1}{2})) \\ &= 0.65 - (\tfrac{4}{6}(0) + \tfrac{2}{6}(-\tfrac{1}{2}\log_2(\tfrac{1}{2}) - \tfrac{1}{2}\log_2(\tfrac{1}{2}))) \\ &= 0.65 - (0 + \tfrac{2}{6}(1)) \\ &= 0.65 - 0.3333 \\ &= 0.3167 \end{aligned}$$

$$\begin{aligned} Gain(Pouted) &= B(\tfrac{5}{6}) - Remainder(Pouted) \\ &= 0.65 - (\tfrac{3}{6}B(\tfrac{2}{3}) + \tfrac{3}{6}B(\tfrac{3}{3})) \\ &= 0.65 - (0.5(-\tfrac{2}{3}\log_2(\tfrac{2}{3}) - \tfrac{1}{3}\log_2(\tfrac{1}{3})) + 0.5(0)) \\ &= 0.65 - (0.5(0.3900 + 0.5283) + 0.5(0)) \\ &= 0.65 - (0.4592) \\ &= 0.1908 \end{aligned}$$

Note that $WatchedOut$ has the higher information gain and so we should split according to that attribute. Note that if $WatchedOut = true$, we can classify any example as $Nice$ and so there is no need to further subdivide that branch. However, if $WatchedOut = false$, we are left with 1 $Nice$ example and 1 $Naughty$ example, and so we must split one more time according to the only remaining attribute, $Pouted$. The resulting decision tree is pictured below.



4. **(10 marks) Neural Networks**

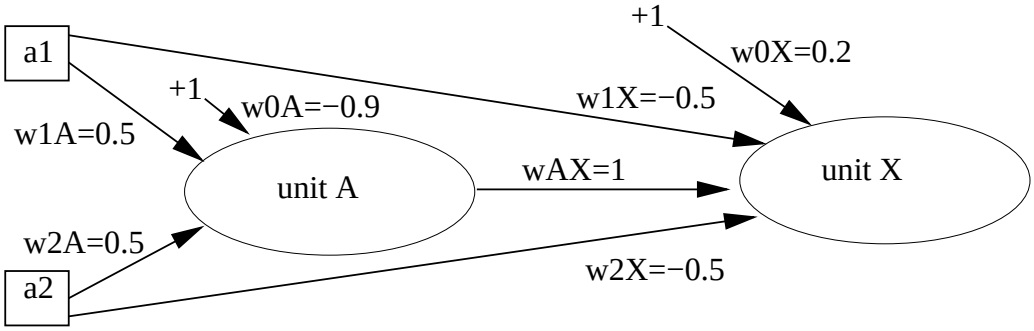
(a) Construct a small neural network that computes the if-and-only-if function, $a_1 \leftrightarrow a_2$, of two boolean inputs. Be sure to explain clearly what the activation function is in each of your neural network units.

a_1	a_2	$a_1 \leftrightarrow a_2$
0	0	1
0	1	0
1	0	0
1	1	1

Note: Since if-and-only-if is not linearly separable, you will need a hidden layer. It can be done with just one hidden node, but correct solutions using more than one hidden node will be accepted as long as they are not overly complicated.

One possible neural network is drawn below, where the activation function for each unit is a threshold function that returns 1 if the weighted sum of the inputs is nonnegative and 0 if the weighted sum of the inputs is negative.

The structure of this network is somewhat different from the ones that we discussed in class, in that the input units connect to the hidden unit, but also directly to the output unit. Note that a different possible solution is provided on the next page.



(b) For each of the four possible combinations of inputs, show the calculation of the weighted sum of inputs and the calculation of the output for each node in your network.

When $a_1 = 0$, $a_2 = 0$, the weighted sum at unit A is $(-0.9)(1) + 0.5(0) + 0.5(0) = -0.9$, which is negative and so, according to the activation function described above, it would output 0. The weighted sum at unit X would then be $0.2(1) + (-0.5)(0) + (-0.5)(0) + 1(0) = 0.2$, which is positive and so it would correctly output 1.

$a_1 = 0$, $a_2 = 1$:
Weighted sum at unit A: $(-0.9)(1) + 0.5(0) + 0.5(1) = -0.4$ (neg.), output 0.
Weighted sum at unit X: $0.2(1) + (-0.5)(0) + (-0.5)(1) + 1(0) = -0.3$ (neg.), output 0.

$a_1 = 1$, $a_2 = 0$:
Weighted sum at unit A: $(-0.9)(1) + 0.5(1) + 0.5(0) = -0.4$ (neg.), output 0.
Weighted sum at unit X: $0.2(1) + (-0.5)(1) + (-0.5)(0) + 1(0) = -0.3$ (neg.), output 0.

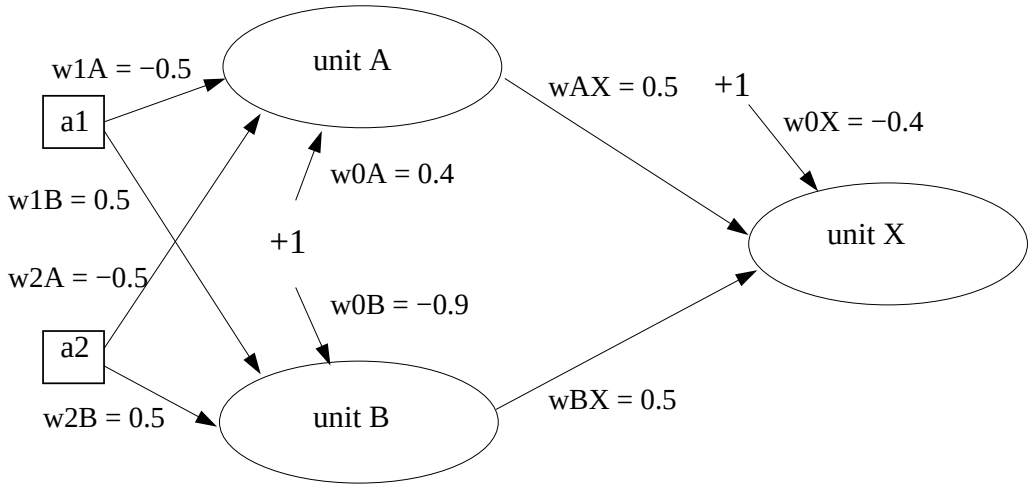
$a_1 = 1$, $a_2 = 1$:
Weighted sum at unit A: $(-0.9)(1) + 0.5(1) + 0.5(1) = 0.1$ (pos.), output 1.
Weighted sum at unit X: $0.2(1) + (-0.5)(1) + (-0.5)(1) + 1(1) = 0.2$ (pos.), output 1.

Another possible neural network is drawn below, where the activation function for each unit is a threshold function that returns 1 if the weighted sum of the inputs is nonnegative and 0 if the weighted sum of the inputs is negative.

Unit A basically implements a NOR gate: it outputs 1 if and only if both a_1 and a_2 are 0.

Unit B implements an AND gate: it outputs 1 if and only if both a_1 and a_2 are 1.

Unit X then returns an OR of unit A and unit B, so it will output 1 if and only if a_1 and a_2 have the same value.



When $a_1 = 0$, $a_2 = 0$, the weighted sum at unit A is $0.4(1) + (-0.5)(0) + (-0.5)(0) = 0.4$, which is positive and so, according to the activation function described above, it would output 1. The weighted sum at unit B would be $(-0.9)(1) + 0.5(0) + 0.5(0) = -0.9$, which is negative and so it would output 0. The weighted sum at unit X would then be $(-0.4)(1) + 0.5(1) + 0.5(0) = 0.1$, which is positive and so it would correctly output 1.

$a_1 = 0$, $a_2 = 1$:
Weighted sum at unit A: $0.4(1) + (-0.5)(0) + (-0.5)(1) = -0.1$ (neg.), output 0.
Weighted sum at unit B: $(-0.9)(1) + 0.5(0) + 0.5(1) = -0.4$ (neg.), output 0.
Weighted sum at unit X: $(-0.4)(1) + 0.5(0) + 0.5(0) = -0.4$ (neg.), output 0.

$a_1 = 1$, $a_2 = 0$:
Weighted sum at unit A: $0.4(1) + (-0.5)(1) + (-0.5)(0) = -0.1$ (neg.), output 0.
Weighted sum at unit B: $(-0.9)(1) + 0.5(1) + 0.5(0) = -0.4$ (neg.), output 0.
Weighted sum at unit X: $(-0.4)(1) + 0.5(0) + 0.5(0) = -0.4$ (neg.), output 0.

$a_1 = 1$, $a_2 = 1$:
Weighted sum at unit A: $0.4(1) + (-0.5)(1) + (-0.5)(1) = -0.6$ (neg.), output 0.
Weighted sum at unit B: $(-0.9)(1) + 0.5(1) + 0.5(1) = 0.1$ (pos.), output 1.
Weighted sum at unit X: $(-0.4)(1) + 0.5(0) + 0.5(1) = 0.1$ (pos.), output 1.