Artificial Intelligence (CMSC 671) Assignment 4

-By Ajay Pal, Rohan Gujarathi, Kushal Samir Mehta, Rishabh Sachdeva.

PART I. LEARNING IN THE WILD (8 PTS.)

Assignment: Consider the problem faced by a robot trying to figure out which of the objects that it can see are manipulable (small enough to to be picked up and handled). (Maximum: 300 words)

1. Explain how this problem fits into the general learning model.

2. Describe the percepts (sensor inputs) and actions of the robot.

3. Describe the types of learning the robot must do.

4. Describe the sub functions the robot is trying to learn in terms of inputs, outputs, and available training data.

Ans: The general learning model is comprised of four fundamental components. The Performance element, Critic, Learning element and the Problem generator. In this model, the agent is expected to improve its performance on the task at hand rather than just learn how to perform the task.
In our case, the agent is a robot that must figure out which objects are manipulable. Before assuming the action, the agent must perform, defining the environment, the sensors and the actuators of the agent is important. The environment of the agent must be a static, discrete and a deterministic one. The agent must first be able to visualize the object. This can be done with the help of a camera that has a decent field of view. The agent must also know its distance from the object. Two camera lenses can be used to sense depth. The actuators are mechanical arms that can interact with the objects. The actions that it can perform are to pick up the object with mechanical arms.
The input consists of a video and its depth. The agent must perform object recognition and also determine the size of the object in each frame of the video. Type of learning the robot does in this case is unsupervised learning. The performance element chooses to either carry the object or look around to find another object that it can lift. Recognition on the image, lifting the object and moving around and scanning for other objects are the sub-functions. The data to train are the frames extracted from the video. The performance standard defines the maximum size of the object that is manipulable and the critic will send its feedback to the learning element to make changes to the actions that the performance element decides. This will help the agent to make better decisions on its environment

PART II. DECISION TREE LEARNING (30 PTS.)

Consider the training examples shown in the following table of data instances for a binary classification problem with three attributes.

5. Entropy (S) = $-p_1 \log p_1 - p_2 \log p_2$

$= -0.44 \log_2 0.44 - 0.55 \log_2 0.55$
$= -0.44(-1.18) - 0.55 (-0.86)$
$= 0.52 + 0.47$
$= 0.99$

6. What are the information gains of a1 and a2 relative to these training examples?

IG(S, a1) = Entropy(S) $-\sum_{v \in T,F} |Sv|/|S| * \text{Entropy}(Sv)$

Entropy(S) = 0.99

Entropy(S:True) = -(3/4)* log(3/4)-(1/4)log(1/4) = (-0.75)*(-0.41) - (0.25)(-2) =0.31+0.5=0.81

Entropy(S:False)= -(1/5)log(1/5)-(4/5)log(4/5) = (-0.20)*(-2.32) – (0.8)(-0.32)=0.46+ 0.26= 0.72

IG(S, a1) = 0.99 – ((4/9)*0.81+ (5/9) * 0.72) = 0.99 – 0.36 – 0.4 = 0.23

IG(S, a2) = Entropy(S) $-\sum_{v \in T,F} |Sv|/|S| * \text{Entropy}(Sv)$

Entropy(S) = 0.99

Entropy(S:True) = -(3/5)* log(3/5)-(2/5)log(2/5) =0.44 + 0.53 = 0.97

Entropy(S:False)= -(2/4)log(2/4)-(2/4)log(2/4) = 1

IG(S, a2) = 0.99 – (5/9 * 0.97) – (4/9 * 1)= 0.99 – 0.54 – 0.44 = 0.005

7. For a3 (which is continuous), compute the information gain for every possible split.

Sorted a2 values	**Split values**
1.0	0.5
3.0	2
4.0	3.5
5.0	4.5
6.0	5.5
7.0	6.5
8.0	7.5
	8.5

Split (0.5):
Case1: a3 <= 0.5
Entropy(S) = 0
Case2: a3 > 0.5
Entropy(S) = $-p_1 \log p_1 - p_2 \log p_2$
= $-0.44 \log_2 0.44 - 0.55 \log_2 0.55$
= -0.44(-1.18) –0.55 (-0.86)
= 0.52 + 0.47
= 0.99
IG(S,a3) = 0.99 - .99 = 0

Split(2) :
Case1: a3 <= 2
Entropy(S) = 0
Case2: a3 > 2
Entropy(S) = $-p_1 \log p_1 - p_2 \log p_2$
= $-3/8 \log_2 3/8 - 5/8 \log_2 5/8$
= -0.375(-1.415) –0.625 (-0.678)
= 0.53 + 0.42
= 0.95
Average = 1/9*0 + 8/9 *0.95 = 0.84
IG(S,a3) = 0.99 - 0.84 = 0.15

Split(3.5) :
Case1: a3 <= 3.5
Entropy(S) = $-p_1 \log p_1 - p_2 \log p_2$
= $-1/2 \log_2 1/2 - 1/2 \log_2 1/2$
= 1
Case2: a3 > 3.5
Entropy(S) = $-p_1 \log p_1 - p_2 \log p_2$
= $-3/7 \log_2 3/7 - 4/7 \log_2 4/7$
= 0.98
Average = 2/9*1 + 7/9 *0.98 = 0.988
IG(S,a3) = 0.99 - 0.988 = 0.002

Split(4.5) :
Case1: a3 <= 4.5
Entropy(S) = $-p_1 \log p_1 - p_2 \log p_2$
= $-2/3 \log_2 2/3 - 1/3 \log_2 1/3$
= 0.92
Case2: a3 > 4.5
Entropy(S) = $-p_1 \log p_1 - p_2 \log p_2$
= $-2/6 \log_2 2/6 - 4/6 \log_2 4/6$
= 0.92
Average = 3/9*0.92 + 6/9 *0.92 = 0.92
IG(S,a3) = 0.99 - 0.92 = 0.07

Split(5.5) :

Case1: a3 <= 5.5

Entropy(S) = $-p_1 \log p_1 - p_2 \log p_2$

= $-2/5 \log_2 2/5 - 3/5 \log_2 3/5$

= 0.97

Case2: a3 > 5.5

Entropy(S) = $-p_1 \log p_1 - p_2 \log p_2$

= $-2/4\log_2 2/4 - 2/4 \log_2 2/4$

= 1

Average = 5/9*0.97 + 4/9 *1 = 0.983

IG(S,a3) = 0.99 - 0.983 = 0.007

Split(6.5) :

Case1: a3 <= 6.5

Entropy(S) = $-p_1 \log p_1 - p_2 \log p_2$

= $-3/6 \log_2 3/6 - 3/6 \log_2 3/6$

= 1

Case2: a3 > 6.5

Entropy(S) = $-p_1 \log p_1 - p_2 \log p_2$

= $-1/3\log_2 1/3 - 2/3 \log_2 2/3$

= 0.918

Average = 6/9*1 + 3/9 *0.918 = 0.972

IG(S,a3) = 0.99 - 0.972 = 0.019

Split (7.5) :

Case1: a3 <= 7.5

Entropy(S) = $-p_1 \log p_1 - p_2 \log p_2$

= $-4/8 \log_2 4/8 - 4/8 \log_2 4/8$

= 1

Case2: a3 > 7.5

Entropy(S) = $-p_1 \log p_1 - p_2 \log p_2$

= $- 1/1 \log_2 1/1$

= 0

Average = 8/9*1 + 1/9 *0 = 0.889

IG(S, a3) = 0.99 - 0.88 = 0.109

Split (8.5) :

Case1: a3 <= 8.5

Entropy(S) = $-p_1 \log p_1 - p_2 \log p_2$

= $-4/9 \log_2 4/9 - 5/9 \log_2 5/9$

= 0.99

Case2: a3 > 8.5

Entropy(S) = $-p_1 \log p_1 - p_2 \log p_2$

= 0

Average = 0.9

IG (S, a3) = 0.99 - 0.88 = 0.109

8. What is the best split (among a1, a2, and a3) according to the information gain?

Ans: Best Split is on attribute **a2**, because it has maximum Information again i.e. 0.23

9. What is the best split (between a1 and a2) according to the classification error rate?

Ans: The error rate without partitioning on any attribute is

E = 1 – max(4/9, 5/9) = 4/9

The error rate at E=a1 is

1) E a1 = T = 1 – max (3/4, ¼) = 1/4

2) E a1 = F = 1 – max (1/5, 4/5) = 1/5

The information gain at a1 will be

∆a1 = E – 4/9* E a1 = T – 5/9* E a1 = F = 4/9 – 1/9 – 1/9 = 2/9

The error rate at E=a2 is

1) E a2 = T = 1 – max (2/5, 3/5) = 2/5

2) E a2 = F = 1 – max (2/4, 2/4) = 2/4

The information gain at a2 will be

∆a2 = E – 5/9* E a2 = T – 4/9* E a2 = F = 4/9 – 2/9 – 2/9 = 0

a1 has the highest gain and hence the best split according to the classification error rate will be at a1

PART III. RESOLUTION AND FORMAL LOGIC (30 PTS.)

10. Represent the following knowledge base in first-order logic. (8 pts)

a) Everything that is hardy, watered, and fertilized will be well-rooted.

Ans: ∀x (hardy(x) ∧ watered(x) ∧ fertilized(x)) → rooted(x)

b) Everything that is well-rooted will survive a planting if it is in season.

Ans: ∀x ∀t rooted(x) ∧ in-season(x) → survive (x, t)

c) A plant survives a planting if and only if they don’t die.

Ans: ∀x ∀t survive (x, t) ↔ die (x, t)

d) Every native plant is hardy.

Ans: ∀x native(x) → hardy(x)

e) If a planting isn't in season, every plant will die in the planting.

Ans: ∀x (∃t in-season(x) ⟶ die (x, t))

f) My inkberry is a native plant.

Ans: inkberry → native(inkberry) (using inkberry for my inkberry)

g) My holly survived the fall planting.

Ans: holly → survive (holly, fall-planting) (using holly for my_holly and fall-planting for fall planting)

h) I fertilize my inkberry.

Ans: inkberry → fertilized(inkberry)

11. Convert the KB to conjunctive normal form. (Hint: you will need to define three constants, in addition to the predicates above and variables x and t.) (8 points)

Ans: a) ∀x (hardy(x) ∧ watered(x) ∧ fertilized(x) →rooted(x))

=∀x (**¬**(hardy(x) ∧ watered(x) ∧ fertilized(x)) ∨ rooted(x)) …...using P → Q = ¬P ∨ Q

=∀x (**¬**hardy(x) ∨ **¬**watered(x) ∨ **¬**fertilized(x) ∨ rooted(x))

= **¬**hardy(x) ∨ **¬**watered(x) ∨ **¬**fertilized(x) ∨ rooted(x)……dropping quantifier

b) ∀x ∀t(rooted(x) ∧ in-season(x) → survive (x, t))

= ∀x ∀t (**¬**(rooted(x) ∧ in-season(x)) ∨ survive (x, t)) …...using P → Q = ¬P ∨ Q

= ∀x ∀t (**¬**rooted(x) ∨ **¬** in-season(x) ∨ survive (x, t))

= ¬rooted(x) ∨ **¬** in-season(x) ∨ survive (x, t) ……dropping quantifier

c) ∀x ∀t (survive (x, t) ⟷ die (x, t))

= ∀x ∀t (survive (x, t) ⟶ die (x, t) ∧ die (x, t) ⟶ survive (x, t)) ……using P↔Q = P→Q ∧ Q→P

=∀x ∀t ((**¬**survive (x, t) ∨ die (x, t)) ∧ (**¬**die (x, t) ∨ survive (x, t))) …...using P → Q = ¬P ∨ Q

= (**¬**survive (x, t) ∨ die (x, t)) ∧ (**¬**die (x, t) ∨ survive (x, t)) ……dropping quantifier

d) ∀x native(x) → hardy(x)

=∀x (¬native(x) ∨ hardy(x)) …...using P → Q = ¬P ∨ Q

= ¬native(x) ∨ hardy(x) ……dropping quantifier

e) ∀x (∃t (in-season(x) ⟶ die (x, t)))

= ∀x (∃t ¬in-season(x) ∨ die (x, t)) …...using P → Q = ¬P ∨ Q

= ∀x (¬in-season(x) ∨ die (x, F(x))

= ¬in-season(x) ∨ die (x, F(x)) ……dropping quantifier

f) inkberry →native(inkberry) (creating variable inkberry for my inkberry)

=¬inkberry ∨ native(inkberry) …...using P → Q = ¬P ∨ Q

g) holly → survive (holly, fall-planting) (creating variable holly and fall-planting)

=¬ holly ∨ survive (inkberry) …...using P → Q = ¬P ∨ Q

h) inkberry →fertilized (my inkberry)

=¬inkberry ∨ fertilized (inkberry) …...using P → Q = ¬P ∨ Q

12. Express the negation of this statement: (inkberry) → survive (inkberry, fall-planting) in conjunctive normal form. (2 pts)

Ans: (inkberry) → survive (inkberry, fall-planting)

Converting to CNF:

=¬inkberry ∨ survive (inkberry, fall-planting)

Using Negation:

=¬ (¬inkberry ∨ survive (inkberry, fall-planting))

=inkberry ∧ ¬ survive (inkberry, fall-planting)

13. Prove that (inkberry) → survive (inkberry, fall-planting).

Adding the negated goal to the KB and using resolution refutation to prove that it is true. You may show your proof as a series of sentences to be added to the KB or as a proof tree. In either case, you must clearly show which sentences are resolved to produce each new sentence, and what the unifier is for each resolution step.

Ans:

1. Negation of above statement is as follows: ¬ ((inkberry) → survive (inkberry, fall-planting))

Which states that inkberry will not survive fall planting, need to

2. From KB[f]: inkberry → native(inkberry) and KB[d]: ∀x native(x) → hardy(x)

we can add following statement to our KB as follows: **KB[i] = inkberry →hardy(inkberry)**

3. From KB[h]: inkberry → fertilized(inkberry) , KB[i]: inkberry →hardy(inkberry)

and KB[a] ∀x (hardy(x) ∧ watered(x) ∧ fertilized(x)) → rooted(x)

we can add following statement to our KB as follows: **KB[j] = inkberry → rooted(inkberry)**

4. From KB[b]: ∀x ∀t rooted(x) ∧ in-season(x) → survive (x, t) , KB[j]: inkberry → rooted(inkberry) and its given that its fall planting therefore we can add following to our KB.

KB[k] = (inkberry) → survive (inkberry, fall-planting)

5.using KB[k] we can prove that ¬ ((inkberry) → survive (inkberry, fall-planting)) is False. Hence proved.

Note: we have assumed that inkberry is watered.

PART IV. FORGING A PATH (30 PTS.)

14. What heuristic did you use for choosing what path to explore next?

Ans: The heuristic that we chose is that the agent always chooses the less explored path first rather than going on to the same path. For this we are maintaining a dictionary of explored coordinates with number of times the agent has visited that tile. Another factor we included in our heuristic is geographical location (sand, path and mountain). Major weightage is given to number of times agent visited a tile, but location cost comes into picture when number of visits are equal.

Also, if any path is a dead end, we don't visit that again.

15. What size beams did you experiment with? What beam size did you use, and why?

Ans: We are using beam of unit size. We preferred only considering one node at a time, because the world ahead is not predictable for the agent. Considering several nodes at a time may cause backtracking, so we tried to avoid it.

16. What was the easiest thing about coding this up? The hardest?

Ans: We already had a skeleton prepared because of previous assignment. So, it was easier to start implementing logic. Also, problem statement was an extension to previous assignment, so it was easy to understand the agent's world. The hardest part was to come up with effective heuristic. Some heuristics performed well with some environments, others with different ones. Coming up with the one which performed good enough with all possible environments was a difficult job.

17. How many times did your group meet in solving this? How well did it go?

Ans: We met 4 times. We started with problem discussion and possible solutions. On second meeting we developed basic program with zero-heuristic. Further, we came up with different possible heuristics and tried testing our agents with different worlds. It went very well; each member came with different approach and we went through some healthy and productive discussions.