CS-331: INTRODUCTION TO ARTIFICIAL INTELLIGENCE

ASSIGNMENT 4:MDPs and Decision Trees

Instructor: Mian Muhammad Awais

TAs: Habiba Farrukh, Aiza Anjum Chaudhry, Fatima Tariq, Hamza Mahmood and Maryam Khalid

Submission Deadline: Sunday 13th December, 2015

As before, this assignment can be done in groups of 2 (same groups as last time). The evaluation of the assignment will be on individual basis. Members of the same group will get marks depending on the working of tasks they have implemented and their performance in viva. So make sure you contribute equally in the assignment.

Also, each member will be required to know what the other has done. One of the aims of these assignments is to ensure that you get the concepts as they are practically applied. With groups, you can divide tasks but in the end you need to know all that has been asked in the assignment.

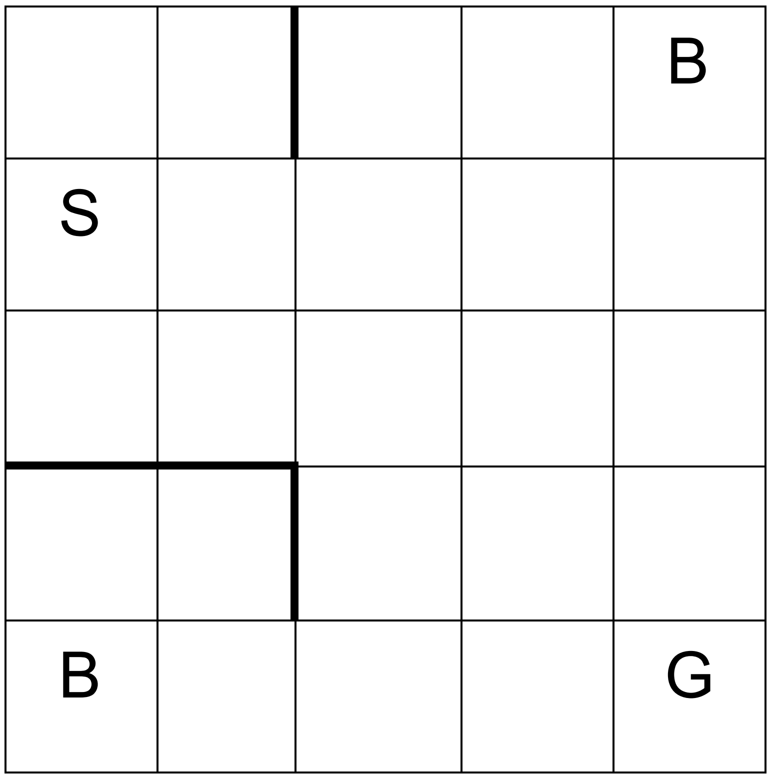
All the submitted codes will be tested against each other and the solutions present online for plagiarism and in case of slightest doubt the concerned case will be forwarded to the Disciplinary Committee.

As always please reach out to the course staff for any help regarding the assignment and its configuration.

PART-1: MDPs
70 POINTS

Submit all your working in files named "Part1-<roll-number1>-<roll-number2>.m" and "Part1-<roll-number1>-<roll-number2>.txt" e.g. "Part1-17100888-17100999.m" and "Part1-17100888-17100999.txt".

Markov decision processes (MDPs) offer an elegant mathematical framework for representing planning and decision problems in the presence of uncertainty.
In this part of the assignment, you have to implement the value iteration algorithm for the following grid world.



An agent starting in the start state S must reach the goal state G. At each time step, the agent can go up, down, left or right. However, the agent's movements are a bit noisy since it goes in the intended direction with a high probability a and in one of the two lateral directions with a low probability b. For instance, when executing the action up, the agent will indeed go up by one square with probability a, but may go left with probability b and right with probability b (here a + b + b= 1). Similarly, when executing the action left, the agent will indeed go left with probability a, but may go up with probability b and down with probability b, When an action takes the agent out of the grid world, the agent simply bounces off the wall and stays in its current location. For example, when the agent executes left in the start state it stays in the start state with probability a, it goes up with probability b and down with probability b. Similarly, when the agent executes up from the start state, it goes up with probability a, right with probability b and stays in the start state with probability b. Finally, when the agent is in the goal state, the task is over and the agent transitions to a special end state with probability 1 (for any action). This end state is absorbing, meaning that the agent cannot get out of the end state (i.e., it stays in the end state with probability 1 for every action). The agent receives a reward of 100 when it reaches the goal state, 70 for the bad state and 1 for every other state, except the end state, which has a 0 reward. The agent's task is to find a policy to reach the goal state as quickly as possible, while avoiding the bad state. In case you are not certain about the transition and reward model, attached is a file (gridWorld.m) with a precise description (in Matlab) of the transition and reward models. Feel free to directly use this.

Tip: The implementation of value iteration will be very short (i.e., a few lines of code) Compute the optimal policy by implementing the value iteration algorithm.
Use a discount factor of 0.99 and run value iteration until the difference between two successive value functions is at most 0.01 (i.e., $|Vt+1(s) - Vt(s)| < 0.01\ \forall s$). Run value iteration once with a = 0.9, b = 0.05 and a second time with a = 0.8, b = 0.1.

Submit a .txt file on LMS containing answers to the following questions.

1. What are the optimal policies and optimal value functions found for a = 0.9, b = 0.05 and for a = 0.8, b = 0.1.
2. What are the differences found in the optimal policies and value functions for different combination of a and b.
3. Explain briefly how a and b impact the optimal policy.

PART-2: DECISION TREES
30 POINTS

For this part, please show all your working. Marks will be deducted for incomplete work.

Submit all your working for this part in a file named “Part2-<roll-number1>-<roll-number2>.doc” e.g. “Part2-17100888-17100999.doc”. Handwritten assignments will not be accepted.

Humidity	Windy	Play
103	False	No
102	False	Yes
101	True	No
88	True	No
89	True	No
103	True	Yes
103	True	Yes
101	True	No
88	False	Yes

a. Calculate the GINI index for the attribute “Windy”. [1]

b. Calculate the GINI index for all possible splits of the continuous attribute “Humidity”. For this, first sort the table for this attribute, then identify all possible binary splits and compute the GINI index for these splits. (Refer to slide 41 of the lecture on Decision Trees for more help.) [4]

c. Identify the best split among “Windy” and all splits of “Humidity” using the GINI index computed in the above parts and draw the partial Decision Tree that includes the first best split. [2]

d. Repeat these steps and draw the final Decision Tree. Show all your working and draw the intermediate Decision Trees obtained at each step. [14]

e. Calculate the entropy of this training set. [1]

f. Calculate the entropy and information gain for the attribute “Windy”. [1]

g. Calculate the classification error rate of this training set. [1]

h. Calculate the classification error rate for the attribute “Windy”. [1]

i. Calculate the classification error rate for all possible splits of the attribute “Humidity”. [4]

j. Identify the best split among “Windy” and all splits of “Humidity” based on the classification error rate. [1]