Udacity MLND P4 -R1

Ryan Gao

change log

1. Add discusstion about involving `deadline` to the states
2. Add answer to the last question

Implement a Basic Driving Agent

QUESTION: *Observe what you see with the agent's behavior as it takes random actions. Does the smartcab eventually make it to the destination? Are there any other interesting observations to note?*

Answer
The agent will firstly go to the intersection which is in the row or column of the destination. Then it will take a turn to the destination. The smartcab will eventually make it to the destination. No matter which color the surrounding trafic lights are, it just go forward until the tracfic light in front of it turns green. Since we have not 'taught' it to obey trafic rules, it sometimes breaks them. Nearly 20 out of 100 reach the destination with a zero or negative net reward.

Inform the Driving Agent

QUESTION: *What states have you identified that are appropriate for modeling the smartcab and environment? Why do you believe each of these states to be appropriate for this problem?*

Answer

- The next way point `[None, 'forward', 'left', 'right']`
- The trafic lights at the intersection `['red', 'green']`
- Whether there is a vehicle at the left or oncoming direction of the intersection
  `oncoming : [None, 'forward', 'left', 'right']`
  `left: [None, 'forward', 'left', 'right']`

The next way point identifies the destination.

The trafic light signal could be sensed by the car agent and if we ignore it there will be penalties

The intersection state must be considered as a state of the learning agent. For we want the car to get reward when perform legally. And there is no need to involve the trafic to the right into the state, because it has no effect when the car execute the its action.

Note
I didn't take the `deadline` into account. There will be enormous state if `deadline` added to the state. So we need a lot more examples for Q-learning convergence. Our goal is to train the agent to reach passengers' destinations in the *allotted* time and not break the tracfic rules. The optimal policy may reward breaking the tracfic rules if the destination is near.

OPTIONAL: *How many states in total exist for the smartcab in this environment? Does this number seem reasonable given that the goal of Q-Learning is to learn and make informed decisions about each state? Why or why not?*

Implement a Q-Learning Driving Agent

QUESTION: *What changes do you notice in the agent's behavior when compared to the basic driving agent when random actions were always taken? Why is this behavior occurring?*

```
Q(s, a) = (1 - alpha) * Q(s, a) + alpha * (reward + gamma * max_Q)
```

The initial Q value of each state is 5.0. The epsilon is set to 0.1. So the agent has 10% chance to perform a random action.
After implement Q-Learing, the agent will not just go straight head to the destination. It learns(or randomly chooses) to take a turn based on the policy we set. For the first several trials it doesn't perform well and the net reward could be negative. As the trial increases, the car performs better, reaching the destination with higher net reward.

Improve the Q-Learning Driving Agent

QUESTION: *Report the different values for the parameters tuned in your basic implementation of Q-Learning. For which set of parameters does the agent perform best? How well does the final driving agent perform?*

Answer:

alpha	gamma	epsilon	success rate	avg net reward	avg deadline

Alpha - the learning rate. Setting it to 0 means that the Q-values are never updated, hence nothing is learned. Setting a high value such as 0.9 means that learning can occur quickly. I set it to 0.9 and 0.5 for comparison.

Gamma is the value of future reward. If it is equal to one, the agent values future reward JUST AS MUCH as current reward. So learning doesn't work at that well at high gamma values. So I set gamma to 0.1 and 0.5 for comparison.

Epsilon is the parameter that we can control the trade-off between exploration and exploitation. I set the Epsilon to 0.9 and 0.1 for comparison.

alpha	gamma	epsilon	success rate	avg net reward	avg deadline
0.9	0.5	0.9	16 / 100	4.6	3.4
0.9	0.5	0.1	95 / 100	22.5	15.6
0.9	0.1	0.9	29 / 100	4.81	3.64
0.9	0.1	0.1	97 / 100	21.06	14.94
0.5	0.5	0.9	29 / 100	3.77	4.16
0.5	0.5	0.1	89 / 100	23.57	15.14
0.5	0.1	0.9	36 / 100	5.23	5.4
0.5	0.1	0.1	98 / 100	21.70	15.1

The "avg net reward" in the form is the average of each trial's net reward, which indicates that after q-learning the agent tends to obey tracfic rules to get more reward.
Similarly, "avg deadline" is the average deadline remain in each trial. It shows that the agent tends to reach the destination as soon as possible.

When alpha = 0.5, gamma = 0.1, epsilon = 0.1, the agent performs best.

QUESTION: *Does your agent get close to finding an optimal policy, i.e. reach the destination in the minimum possible time, and not incur any penalties? How would you describe an optimal policy for this problem?*

direction	light	oncoming	left	forward	right	None	left

Answer

The optimal policy for the agent would be 1) obey the trafic so as to gain maximum reward 2) move to the destination as soon as possible

As discussed in previous question, the avgrage deadline remained in the case (alpha = 0.5, gamma = 0.1, epsilon = 0.1) is much greater than that in (alpha = 0.5, gamma = 0.1, epsilon = 0.9) which has poorly learning ability.

Here are some data extracted from the Q-table below when the `light` is `red` .
The 4 columns on the right is the Q values for each possible action.

direction	light	oncoming	left	forward	right	None	left
forward	red	forward	None	0.88	2.36	1.49	5.0
forward	red	left	None	0.88	5.0	5.0	5.0
forward	red	None	left	2.25	2.26	2.52	5.0
forward	red	None	forward	2.0	5.0	5.0	5.0
forward	red	None	right	0.56	2.26	5.0	5.0
forward	red	None	None	-0.83	-0.3	0.14	-0.95
left	red	None	right	2.1	5.0	5.0	5.0
left	red	None	left	2.01	5.0	5.0	5.0
left	red	None	None	-0.88	-0.24	0.12	-0.89
right	red	forward	None	0.88	3.51	1.63	2.11
right	red	None	None	-0.37	2.04	0.92	-0.39

Obviously, every q-value of `forward` action in the table is the smallest in all four q values, which shows that the agent is being punished when breaking the rules.
Let's look at a state `(forward, red, None, left)`. The agent will choose to go left since the q-value is the largest even though the direction to the target is to the forward. However, a lot of states' q value is still the initial one. More iteration or dummy agent is needed.

Appendix

In [59]:

```
# waypoint, light, oncoming , left
display(q25)
```

```
forward red     None    None    forward:-0.54   right:-0.37    None:
0.18    left:-0.39

forward green   right   None    forward: 3.6   right: 5.0    None:
 5.0    left: 5.0

right   red     forward None    forward:0.88   right:3.51    None:1.
63    left:2.11

left    green   None    None    forward:0.42   right:0.42    None:0.
51    left:2.09

forward green   None    right   forward:3.51   right: 5.0    None:
 5.0    left: 5.0

left    red     None    left    forward:2.01   right: 5.0    None:
 5.0    left: 5.0

right   green   None    None    forward:0.42   right:2.21    None:0.
93    left:0.08

forward red     None    left    forward:2.25   right: 5.0    None:
 5.0    left: 5.0

forward green   None    None    forward: 2.1   right:0.06    None:0.
47    left:0.06

left    red     None    None    forward:-0.73   right:-0.11    None:
0.14    left:-0.41

right   red     None    None    forward:0.88   right:2.19    None:1.
63    left:0.81

left    green   None    forward forward:2.36   right: 5.0    None:
 5.0    left: 5.0

forward green   None    left    forward:3.65   right:2.35    None:1.
63    left:2.35

forward red     None    right   forward:2.11   right:2.26    None:
 5.0    left: 5.0
```

In [61]:

```
display(q50)
```

forward green None right forward:3.51 right: 5.0 None:
 5.0 left: 5.0

forward red None None forward:-0.67 right:-0.31 None:
0.17 left:-0.39

left red None None forward:-0.76 right:-0.24 None:
0.12 left:-0.41

right green None forward forward:2.36 right: 5.0 None:
 5.0 left: 5.0

forward red None forward forward: 2.0 right: 5.0 None:
 5.0 left: 5.0

right red None None forward:0.88 right:2.06 None:1.
63 left:0.01

forward green None left forward:2.93 right:2.35 None:1.
63 left:2.35

left green None forward forward:2.36 right: 5.0 None:
 5.0 left: 5.0

right red forward None forward:0.88 right:3.51 None:1.
63 left:2.11

left red None right forward: 2.1 right: 5.0 None:
 5.0 left: 5.0

left green None None forward:0.07 right:0.07 None:0.
51 left:2.17

forward green left None forward:3.61 right: 5.0 None:
 5.0 left: 5.0

left red None left forward:2.01 right: 5.0 None:
 5.0 left: 5.0

right green None None forward:-0.1 right:2.22 None:0.
93 left:0.08

forward red None left forward:2.25 right: 5.0 None:
 5.0 left: 5.0

forward green None None forward:2.24 right:0.06 None:0.
47 left:-0.24

forward green right None forward: 3.6 right: 5.0 None:
 5.0 left: 5.0

forward red None right forward:2.11 right:2.26 None:
 5.0 left: 5.0

right green left None forward:2.36 right:3.61 None:
 5.0 left: 5.0

left green left None forward:2.36 right: 5.0 None:
 5.0 left: 5.0

In [62]:

```
display(q75)
```

right red forward None forward:0.88 right:3.51 None:1.
63 left:2.11

left green None None forward:0.07 right:0.07 None:0.
37 left:2.21

forward green None right forward:3.51 right: 5.0 None:
5.0 left: 5.0

right green None None forward:-0.1 right:2.11 None:0.
93 left:0.08

forward red None left forward:2.25 right:2.26 None:2.
52 left: 5.0

forward red left None forward:0.88 right: 5.0 None:
5.0 left: 5.0

right green left None forward:2.36 right:3.61 None:
5.0 left: 5.0

forward red None forward forward: 2.0 right: 5.0 None:
5.0 left: 5.0

left green None forward forward:2.36 right: 5.0 None:
5.0 left: 5.0

left red None right forward: 2.1 right: 5.0 None:
5.0 left: 5.0

forward red None right forward:2.11 right:2.26 None:
5.0 left: 5.0

left green left None forward:2.36 right: 5.0 None:
5.0 left: 5.0

forward red None None forward:-0.83 right:-0.31 None:
0.14 left:-0.84

forward green right None forward: 3.6 right: 5.0 None:
5.0 left: 5.0

forward red forward None forward:0.88 right:2.36 None:1.
49 left: 5.0

forward green left None forward:3.61 right:2.36 None:
5.0 left: 5.0

left red None left forward:2.01 right: 5.0 None:
5.0 left: 5.0

forward green None None forward:2.13 right:0.06 None:0.
28 left:-0.27

left red None None forward:-0.88 right:-0.24 None:
0.18 left:-0.89

right green None forward forward:2.36 right: 5.0 None:
5.0 left: 5.0

right red None None forward:0.05 right:2.13 None:0.

92   left:-0.39

forward green   None   left   forward:2.93   right:2.35   None:1.
63   left:2.35

right   green   None   left   forward:2.36   right: 5.0   None:
5.0   left: 5.0

In [55]:

```
display(q100)
```

```
right   red     forward None    forward:0.88   right:3.51    None:1.
63    left:2.11

left    green   None    None    forward:0.07   right:-0.1    None:0.
37    left:2.19

forward green   None    right   forward:3.51   right: 5.0    None:
 5.0    left: 5.0

right   green   None    None    forward:-0.1   right:2.26    None:0.
93    left:0.08

forward red     None    left    forward:2.25   right:2.26    None:2.
52    left: 5.0

forward red     left    None    forward:0.88   right: 5.0    None:
 5.0    left: 5.0

right   green   left    None    forward:2.36   right:3.61    None:1.
06    left:2.36

forward red     None    forward forward: 2.0   right: 5.0    None:
 5.0    left: 5.0

left    green   None    forward forward:2.36   right: 5.0    None:
 5.0    left: 5.0

left    red     None    right   forward: 2.1   right: 5.0    None:
 5.0    left: 5.0

forward red     None    right   forward:0.56   right:2.26    None:
 5.0    left: 5.0

left    green   left    None    forward:2.36   right: 5.0    None:
 5.0    left: 5.0

forward red     None    None    forward:-0.83   right:-0.3    None:
0.14    left:-0.95

forward green   right   None    forward: 3.6   right: 5.0    None:
 5.0    left: 5.0

forward red     forward None    forward:0.88   right:2.36    None:1.
49    left: 5.0

forward green   left    None    forward:3.61   right:2.36    None:
 5.0    left: 5.0

left    red     None    left    forward:2.01   right: 5.0    None:
 5.0    left: 5.0

forward green   None    None    forward:2.19   right:-0.37    None:
0.25    left:-0.27

left    red     None    None    forward:-0.88   right:-0.24    None:
0.12    left:-0.89

right   green   None    forward forward:2.36   right: 5.0    None:
 5.0    left: 5.0

right   red     None    None    forward:-0.37   right:2.04    None:
```

```
0.92    left:-0.39

forward green   None    left    forward:2.47   right:2.35    None:1.
63    left:2.35

right   green   None    left    forward:2.36   right: 5.0    None:
 5.0    left: 5.0
```