Aaron Iglesias
Udacity
Machine Learning Engineer Nanodegree
Project 4 - Train a Smartcab to Drive
09/24/2016

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?

The smartcab sometimes makes it to the destination although not efficiently because of random actions. Note that it reaches the destination sometimes (as opposed to eventually) because the Environment object sets a hard time limit to avoid deadlocks, even if 'enforce_deadline' is set to False.

Another interesting observation is that not only does the smartcab receive negative rewards more frequently than positive rewards as time approaches infinity, but its net reward approaches negative infinity as time approaches infinity. This means that the positive rewards the smartcab occasionally receives with random actions are not large enough to balance out the frequent negative rewards it receives.

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?

These states are appropriate for modeling the smartcab and environment:

1. Green light and learning agent's next waypoint is 'forward'.
2. Green light and learning agent's next waypoint is 'right'.
3. Green light, learning agent's next waypoint is 'left', and oncoming's next waypoint is 'forward' or 'left'.
4. Green light, learning agent's next waypoint is 'left', and oncoming's next waypoint is not 'forward' or 'left'.
5. Red light, learning agent's next waypoint is 'right', and left's next waypoint is 'forward'.
6. Red light, learning agent's next waypoint is 'right', and left's next waypoint is not 'forward'.
7. Red light and learning agent's next waypoint is not 'right'.

I believe each of these states are appropriate for this problem because a positive reward occurs if and only if a move is okay and the learning agent follows its next waypoint. A move is okay if any of the four conditions are true: (1) the learning agent moves forward on a green light, (2) the learning agent moves left on a green light when oncoming traffic is not moving forward or right, (3) the learning agent moves right on a green light or left traffic is not moving forward, or (4) the learning agent does not move. Because reward depends upon these four conditions, I believe the states described in the previous paragraph are all appropriate for this problem.

Note that the deadline variable is excluded. This is because not only are the traffic rules in the previous paragraph independent of the deadline, but also including the deadline would increase the state-space by a factor of the largest possible deadline. This would make the state-space too large and the learning agent's state too specific, which requires a very long exploration

phase that is not feasible to perform within 100 trials. This means including the deadline would negatively impact the learning agent's performance by making it too difficult to explore the entire state-space. It is better to exclude it and keep the more generalized list of 7 states mentioned above. Also, including it would not increase performance even if all of the states are explored as the deadline is irrelevant to learning traffic rules.

The explore the 7 appropriate states described in the beginning of this section, the learning agent performs random actions for its first 90 trials.

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?

The learning agent eventually reaches its destination consistently with Q-Learning, whereas it reached its destination inconsistently with random actions. This behavior is occurring because each time it executes action A in state S, its Q-value that corresponds to S and A are updated so it executes the optimal action next time it is in S by choosing the action with the highest Q-value.

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?

The table below displays the performance of the top 5 combinations of learning rate and discount factor values. Note that the performance is the average number of times (from 100 sets of 100 trials) the learning agent completes its final 10 trials after being trained for 90 trials. Negative Penalty % is the percentage of negative penalties the learning agent's incurs out of its total number of actions during its final 10 trials after being trained for 90 trials.

Learning Rate	Discount Factor	Performance	Negative Penalty %
0.2	0.2	10	0.00587563244655
0.1	0.1	9.99	0.0175061932287
0.1	0.3	9.99	0.0106548793259
0.2	0	9.99	0.0189755939361
0.2	0.4	9.99	0.00741726657244

The learning agent performs best at a learning rate of 0.2 and discount factor 0.2. Out of 100 sets of 100 trials, the final driving agent averages 10 successful trials while incurring a negative penalty 0.00587% of the time after being trained for 90 trials.

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?

Yes, the agent gets close to finding an optimal policy. Out of 100 sets of 100 trials, the final driving agent averages 10 successful trials while incurring a negative penalty 0.00587% of the time after being trained for 90 trials.

I would describe an optimal policy for this problem as one that reaches the destination as quickly as possible without incurring any penalties. This means the learning agent learns to follow its waypoint under the correct traffic conditions (+2 reward), or does not move otherwise (+0 reward). Not following the waypoint yields a negative reward, which is unfavorable.

Specifically, this is the optimal policy for my identified states:

1. Learning agent's moves forward if green light and learning agent's next waypoint is 'forward'.
2. Learning agent moves right if green light and learning agent's next waypoint is 'right'.
3. Learning agent moves does not move if green light, learning agent's next waypoint is 'left', and oncoming's next waypoint is 'forward' or 'left'.
4. Learning agent moves left if green light, learning agent's next waypoint is 'left', and oncoming's next waypoint is not 'forward' or 'left'.
5. Learning agent does not move if red light, learning agent's next waypoint is 'right', and left's next waypoint is 'forward'.
6. Learning agent moves right if red light, learning agent's next waypoint is 'right', and left's next waypoint is not 'forward'.
7. Learning agent does not move if red light and learning agent's next waypoint is not 'right'.

These are the parameter-tuned learning agent's Q-values for all state-action pairs after 100 trials:

	None	'Forward'	'Left'	'Right'
1	0.5135232607199794	5.081385244290079,	-0.5755493329053348	-0.566051777616929
2	0.4507135435922501	-0.08679284555457617	-0.3595549262069097	2.0641658532459406
3	0	-0.01904107574368733	-0.2	-0.010947169287909
4	0.6333055428232575	-0.45727598172024403	1.985610732540703	-0.553157699196905
5	0	-0.2	-0.2	-0.2
6	0.4513946770459477	-0.5470674702893057,	-0.5526014062187082,	2.0135638361691655
7	3.548512787488607e-12	-0.9999999272236618,	-0.9999999414819009,	-0.420949091488245

Note that the highest Q-value for each state corresponds to the action that agrees with the optimal policy described above.