**Train a Smartcab to Drive**

This report is addressing the questions associated with an Udacity’s project, in which we formulate reinforcement learning algorithms to train an agent (Smartcab) to drive around while learning the fastest way to reach its destination with minimum violation of traffic rules. First, we will implement a basic driving agent which randomly drives around. Then, we will inform the agent about the rules to update states. Next, we implement Q-learning. Finally, we optimize the behavior of the agent to reach the destination. Detailed answers to each question are *provided in blue*.

**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?*

*The agent which takes a random action from the set of actions (None, ‘forward’, ‘left’, ‘right’) at each intersection eventually makes it to the destination. However, the random agent’s approach to reach the destination is poor in performance. It sometimes takes no action even when it is not necessary (no oncoming traffic or red light). It sometimes hits the other cars or disobeys the traffic signals. The random agent is always trying new action without any knowledge of the past rewarding (or punishing) actions. Most of the time, it does not reach its destination in time.*

**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?*

*The states should include information about the intersection (environment) that is necessary for choosing the correct action.*

1. *The state of the traffic light light (‘red’ or ‘green’)*
2. *The direction of the car in the oncoming lane (None, ‘forward’, ‘left’, ‘right’)*
3. *The direction of the car in the right lane (None, ‘forward’, ‘left’, ‘right’)*
4. *The direction of the car in the left lane (None, ‘forward’, ‘left’, ‘right’)*
5. *The direction suggested by the route planner-next\_waypoint (None, ‘forward’, ‘left’, ‘right’).*

*I choose to model the two state variables- next\_waypoint and traffic light – because combining the two variables will be very useful in performance and help in training the cab to perform legal moves.*

***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?*

*If there are no rules, there are 2\*4\*4\*4\*4=512 states in total. However, some states will never be visited. If the traffic light is red, the car on the oncoming lane will always stop. In Q-Learning, we do not need to create the entire matrix, instead, we can fill the values iteratively as the agent drives around.*

**Implement a Q-Learning Driving Agent**

With your driving agent being capable of interpreting the input information and having a mapping of environmental states, your next task is to implement the Q-Learning algorithm for your driving agent to choose the *best* action at each time step, based on the Q-values for the current state and action. Each action taken by the **smartcab** will produce a reward which depends on the state of the environment. The Q-Learning driving agent will need to consider these rewards when updating the Q-values. Once implemented, set the simulation deadline enforcement enforce\_deadline to True. Run the simulation and observe how the **smartcab** moves about the environment in each trial.

***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?*

*After the implementation of Q-Learning, the agent started to obey the traffic signals and follow the directions provided by the route planner. It is because the agent takes the learned Q-values into account during its exploration.*

**Improve the Q-Learning Driving Agent**

Your final task for this project is to enhance your driving agent so that, after sufficient training, the **smartcab** is able to reach the destination within the allotted time safely and efficiently. Parameters in the Q-Learning algorithm, such as the learning rate (alpha), the discount factor (gamma) and the exploration rate (epsilon) all contribute to the driving agent’s ability to learn the best action for each state. To improve on the success of your **smartcab**:

* Set the number of trials, n\_trials, in the simulation to 100.
* Run the simulation with the deadline enforcement enforce\_deadline set to True (you will need to reduce the update delay update\_delay and set the display to False).
* Observe the driving agent’s learning and **smartcab’s** success rate, particularly during the later trials.
* Adjust one or several of the above parameters and iterate this process.

This task is complete once you have arrived at what you determine is the best combination of parameters required for your driving agent to learn successfully.

***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?*

*ti*

***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?*