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?*

It does make to the destination since we have set 'enforce_deadline' parameter to 'False'. However, sometimes it takes a longer time to reach the destination as the moves are random thereby introducing uncertainty in ending up at the destination.

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?*

I have chosen to implicitly define a state with the help of some input variables. The variables that contribute to each state are:

1. Traffic on the left, right and oncoming lanes.
2. Traffic light color
3. Next waypoint

This is because, the agent should also consider whether there is any traffic from other lanes occupying the intersection and whether it is allowed to move in the direction depending upon the traffic light color before taking action. It should also have an idea about the direction in which it should move. Although, deadline is another variable that can be included in the state of the agent. However, it is not a good idea since it increases the number of states in the state space. If the deadline is not included, the number of states = 2 * 2 * 2 * 2 * 3 = 48. However, if the deadline is included, the number of states will be = 2 * 2 * 2 * 2 * 3 * 20 = 960 (as the number of values of deadline is 5 times the max allowed distance, i.e., 4).

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?*

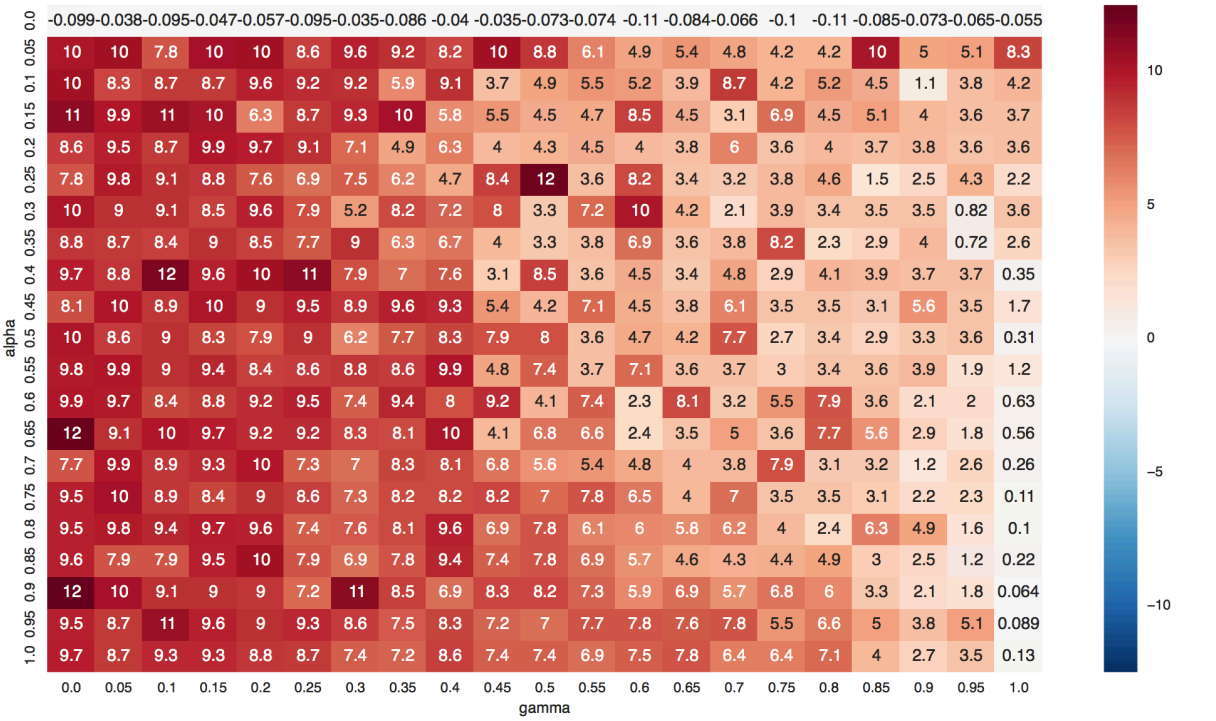
In basic driving agent, the moves were random and thus it was quite unlikely to end up at the destination. After implementing Q learning, the agent took random moves with a high probability to explore the state space in initial set of trials and in later trials it exploited the explored state space with a high probability thereby choosing the best moves of what it had discovered until then. For the optimal set of parameters that I derived from a set of (gamma, alpha) values, it eventually learns to follow the best next_waypoint.

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?*

I varied the epsilon value based on the trail number with the formula (1/trial). This helps achieving a good balance of exploration v/s exploitation wherein the agent initially explores the state space and in later stages exploits the experience it has obtained till then. I have tuned the Q-Learning algorithm for values of alpha and gamma from 0.0 to 1 in steps of 0.05 for each of them. This resulted in the following heatmap wherein the value of performance metric is given by the formula:

Performance = ([Average correct moves * 4]/ ([Average Incorrect Moves] + [Average Invalid Moves * 2])) * [Success count / 100] * [1 / Average Steps]

This is because I want the performance to be directly proportional to correct moves made by the agent and success count. Also, performance should be inversely proportional to average incorrect moves, average invalid moves and steps taken. The multipliers in types of moves indicate the ratio of the magnitude of rewards. Performance is inversely proportional to steps taken because I want my agent to reach the destination faster if there exists a tie in multiple solutions.



Thus the optimal set of parameters in pairs of (alpha, gamma) are:

- 0.25, 0.5
- 0.4, 0.1
- 0.65, 0.0
- 0.9, 0.0

For gamma = 0, the agent does not consider previous state knowledge and thus it needs to be discarded. In order to choose from remaining two solutions, I ran the simulations again and discovered that for alpha=0.4 and gamma=0.1, the number of incorrect + invalid moves is much smaller than the other one. Thus, I preferred the parameters, alpha =0.4 and gamma = 0.1

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?*

The 100 trials are not sufficient enough to obtain a policy similar to optimal one. However, I ran the simulation for 1000 trials with the above set of parameters and in latter half of the simulation, it rarely took an invalid or incorrect move (even if the invalid move occurred, it must be because of the random number being less than epsilon value which results in choosing a random action).

The optimal policy would be the one where the agent follows the traffic rules while taking an action that aligns with next_waypoint to reach the destination.