***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:*** Smartcab does not reach the destination when the parameters are not set and tuned. Smartcab is making random actions even when the destination is in next intersection, instead of moving towards the destination the smartcab went in different direction. Smartcab does not seem to be aware of the destination and is not working towards moving in that direction. After observing for 20 trails, I did not see even once when the Smartcab reach its destination during the designated time. This would be the typical movement of a Cab without any plan to reach its destination. If the deadline is not enforced, smartcab would eventually reach its destination due to random actions.

***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***: *I have identified following states to be relevant for the smartcab environment, namely:* 'light', 'oncoming', 'left', 'right', 'next_waypoint', 'deadline' and 'action'.

State of the traffic light determines the direction of smartcab movement, smartcab has to make a decision either to wait the signal or make a turn, so traffic light would be critical in modeling.

'next_waypoint' provides the next waypoint to the destination, this is crucial indicator which points which direction the smartcab should be moving.

'deadline' denotes the current time left from the allotted time to reach the destination, this would be a continuous value from 0 to max value of 55, selected by the simulator.

'left' and 'right' indicates if there are traffic on the left and right direction.

'action' provides different actions that smartcab could take.

'deadline' and 'right' are states that smartcab could use, but these parameters though relevant for modeling increases the sizes of state/action space for the smartcab to learn and model, hence after experimenting with these variables I removed them from the final model.

Including 'deadline' would increase the complexity of the state space drastically from 1536(2*4*4*3*4*4) to 46080(2*4*4*4*3*4*30) states(assuming on average 30 as deadline), this would have a very high impact on smartcab's performance since all the values in the Q-table would not be populated due to inability of the smartcab to learn the value of all the action in the given trials and also that Q-learning would suffer from slow rate of convergence due to state space complexity.

Since the smartcab environment Right-of-way rule applies, we could safely remove the state 'right'.

Making the changes in the python code to include or remove the states in the namedtuple is easier, but for final project submission and result analysis I have removed these 2 states.

***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?*

***Answer:*** State: light has – green, red
oncoming – none, forward, right, left
left – none, forward, right, left
right – none, forward, right, left
next_waypoint – left, right, forward
deadline – 0 to max value of 55
action - none, forward, right, left
Total states: 2 * 4 * 4 * 4 * 3 * 4 * 55 = 84480 states
There are 84480 states, too many for this modeling problem, we could remove 2 states: 'deadline' and 'right' to reduce the number of states to 384 and make an informed decision.
On an average with a deadline of 30, in 100 trials we could expect the smartcab to learn about 3000 states. With the tuned learning rate, 100 trials and objective of reaching the destination, 384 states seems reasonable compared with 84480 states.

***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?*

***Answer:*** Smartcab is making an intelligent decision at every intersection and reaches the destination, though it missed the destination at the beginning of the trial, it reached the destination after it calculated the optimal action-selection policy.
Smartcab is able to perform better than random action because it works by learning an action-value function that ultimately gives the expected utility of taking a given action in a given state and following the optimal policy thereafter. The policy is the rule that the smartcab follows in selecting the action, given the state it is in. When such an action-value function is learned, the optimal policy can be constructed by simply selecting the action with the highest value in each state. In this project we store the values in a dictionary with the states as a named tuple key, this dictionary is referenced to take next action in a state, also the value is updated with the reward for the action taken.

***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:*** 3 parameters influence the Q-Learning algorithm, namely: Learning rate, Discount factor and Exploration rate.
Learning rate(alpha) determines to what extent the newly acquired information will override the old information. A value of 0 will make the agent not learn anything, and a value of 1 would make the agent consider only the most recent information.
Discount factor(gamma) determines the importance of future rewards, a value of 0 would make the agent short-sighted by considering only current rewards, while a factor approaching 1 will make it strive for a long-term high reward.
Exploration rate(epsilon) determines to what extent actions for each state need to be explored instead of using the best action for each state. Low value would make the smartcab take the best

actions for the state, high value would make the smartcab explore a random action for current state.

I explored a grid search for alpha between (0, 1) with step 0.05, gamma between range (0, 1) with step 0.05, and exploration rate between range (100, 400) with step 100. Following parameters provided the best performance with least number of failures, i.e. least number of times the smartcab did not reach the destination.

.5 <= alpha <= .35

.1 <= gamma <= .15

Exploration rate = 100

For alpha=.2, gamma=.15 and epsilon=100, it failed to reach the destination 7 times out of 100, i.e. success rate was greater than 90%.

Following table shows the results for different parameters that had low failures:

| alpha | gamma | epsilon | count | success | failure | penalty | net_rewards | time_remaining | size |
|---|---|---|---|---|---|---|---|---|---|
| 0.65 | 0.15 | 100 | 1375 | 99 | 1 | 467 | 2130.5 | 16 | 55 |
| 0.8 | 0 | 100 | 1430 | 99 | 1 | 467 | 2204 | 18 | 66 |
| 0.95 | 0.1 | 100 | 1525 | 98 | 1 | 511 | 2246.5 | 15 | 53 |
| 0.3 | 0.05 | 100 | 1509 | 98 | 2 | 518 | 2241.5 | 19 | 58 |
| 0.9 | 0.1 | 100 | 1396 | 96 | 2 | 439 | 2222 | 20 | 54 |
| 0.1 | 0.1 | 100 | 1522 | 94 | 3 | 534 | 2212 | 0 | 57 |
| 0.1 | 0 | 100 | 1347 | 95 | 3 | 426 | 2173 | 19 | 47 |
| 0.1 | 0.25 | 100 | 1510 | 94 | 3 | 461 | 2596 | 4 | 46 |
| 0.15 | 0.05 | 200 | 1416 | 97 | 3 | 489 | 2152 | 15 | 62 |
| 0.35 | 0 | 200 | 1394 | 96 | 3 | 462 | 2157.5 | 17 | 62 |
| 0.4 | 0.35 | 100 | 1555 | 93 | 3 | 484 | 2657.5 | 19 | 52 |
| 0.45 | 0.7 | 100 | 1472 | 96 | 3 | 424 | 2769 | 21 | 49 |
| 0.5 | 0.1 | 100 | 1595 | 94 | 3 | 544 | 2307 | 37 | 55 |
| 0.55 | 0.3 | 100 | 1496 | 95 | 3 | 482 | 2419 | 17 | 53 |
| 0.55 | 0.25 | 100 | 1448 | 94 | 3 | 474 | 2327 | 18 | 49 |
| 0.6 | 0.4 | 100 | 1514 | 97 | 3 | 481 | 2544.5 | 20 | 52 |
| 0.65 | 0.25 | 200 | 1492 | 95 | 3 | 480 | 2366 | 18 | 61 |
| 0.65 | 0.3 | 100 | 1493 | 97 | 3 | 482 | 2414 | 13 | 52 |
| 0.7 | 0.4 | 100 | 1478 | 97 | 3 | 467 | 2466 | 16 | 60 |
| 0.7 | 0 | 100 | 1576 | 95 | 3 | 537 | 2258 | 19 | 58 |
| 0.75 | 0.2 | 100 | 1385 | 97 | 3 | 448 | 2191 | 20 | 57 |
| 0.85 | 0.5 | 100 | 1506 | 97 | 3 | 466 | 2585.5 | 31 | 52 |

***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?*

***Answer:*** Yes, agent was able to determine the optimal policy for low alpha, gamma and epsilon values as shown above. Low learning rate(alpha), lower discount rate and take best action for each state is the optimal policy for this samartcab problem.
These parameters are also based on the grid environment, how noisy is the environment, number of vehicles on the road.
For high exploration rate(epsilon) smartcab explores a random action and incurs penalty, for smaller rate the penalty is less. When smartcab is not able to select the best action for a state, it selects random action and incurs a penalty, but this penalty goes down as the Q-table gets populated for all the states.

Including all the states except 'deadline' and 'right', I calculated the number of failures and penalty after 90 trials and 10 trials after that, though smartcab reached the destination, it still had penalty since all the states has not been explored.
count: 1363 success: 83 failure: 4 penalty: 457 net_reward 2053.5 time remaining: 25 size: 55
count: 1509 success: 93 failure: 4 penalty: 498 net_reward 2295.5 time remaining: 5 size: 59

Including 'deadline' increased the number of failures and penalty, due to increase in the size of the state/action that need to be maintained.
count: 1879 success: 61 failure: 31 penalty: 856 net_reward 1692.0 time remaining: 4 size: 766
count: 2075 success: 69 failure: 33 penalty: 934 net_reward 1908.5 time remaining: 26 size: 791

Optimal policy would be to have low alpha, gamma and epsilon values that includes states: 'light', 'next_waypoint', 'oncoming', 'left', and 'action'. Following grid shows the final Q-table, we could see that the agent has not adequately visited all the states to learn all the value of the action in the first 100 trials. Of the 384(2*4*4*3*4) possible states only 46 states have been visited, i.e. 12% of the states have values. Since the smartcab is not able to visit all the states, it would make mistakes, but with only 12% of the states learned, smartcab is able to reach its destination during the last 10 trials.

| | | | | **action** | | | |
|---|---|---|---|---|---|---|---|
| **light** | **oncoming** | **left** | **next_waypoint** | **none** | **left** | **right** | **forward** |
| green | forward | None | right | | | 0.442 | |
| green | left | None | forward | | 0.013 | -0.100 | |
| green | left | None | right | | | 0.546 | |
| green | None | forward | forward | 0.000 | | -0.073 | |
| green | None | forward | left | | | | -0.016 |
| green | None | forward | right | | | 1.589 | |
| green | None | left | forward | | 0.029 | -0.100 | |
| green | None | left | right | | | 0.869 | |
| green | None | None | forward | 0.175 | -0.099 | -0.258 | 4.512 |
| green | None | None | left | 0.265 | 4.931 | -0.100 | -0.180 |
| green | None | None | right | 0.172 | -0.030 | 2.475 | -0.121 |
| green | None | right | forward | | -0.100 | -0.100 | |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| green | right | None | forward | | -0.200 | | |
| green | right | None | right | | | | -0.029 |
| red | forward | None | forward | | | 0.045 | -0.200 |
| red | left | None | forward | -0.006 | -0.200 | | |
| red | left | None | forward | | | -0.067 | |
| red | left | None | right | | | 0.862 | |
| red | left | None | forward | | | | -0.360 |
| red | None | left | forward | | | 0.021 | |
| red | None | None | forward | 0.000 | -1.000 | -0.391 | -1.000 |
| red | None | None | left | 0.000 | -1.000 | -0.300 | -1.000 |
| red | None | None | right | 0.070 | | 2.429 | -0.321 |
| red | None | right | forward | | -0.200 | -0.084 | |