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 smartcab doesn't reach its destination most of the time (almost all the time in my testing). It takes random action choices and the cab moves slowly away from the start point in a random direction.

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 found combination of inputs be appropriate for capturing state information as shown below:
self.state = (self.next_waypoint, inputs['light'], inputs['oncoming'], inputs['left'], inputs['right'])
Using above state combination, we can retrieve best action based on rewards captured for (state, action) pairs.
self.next_waypoint – captures the best route planner action
inputs['light'] – capture the traffic light at the intersection
inputs['right','left','incoming'] – capture the oncoming traffic at the intersection

Rewards are based on the best possible taken by smartcab at the intersection. We need this state information so it can learn the rewards it gets from the environment based on previous actions taken and continue to take best possible action through Q-Learning.

The other possible variable that can be included in state is self.env.get_deadline(self), by including this variable, learning takes lot longer since the number of possible states increases, and Q-learning takes longer.

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?

Total possible states for below state:
```
self.state = (self.next_waypoint, inputs['light'], inputs['oncoming'], inputs['left'], inputs['right'])
```
Total possible = 4*2*4*4*4

This is appropriate for current learning because of the short grid size and small number of deadline iterations that we take. It takes longer to train with the increase in 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?

The smartcab moves towards destination because of Q-Learning, it chooses the best action based on the rewards it receives for each of its actions. The smartcab tries to choose next step by evaluating Q value for all of its immediate (state,action) pairs and it will eventually reaches its destination. The smartcab reached destination for most of the time with Q-Learning. The smartcab initially chooses its action randomly as it receives feedback from the environment and as it learns the environment, it seems to take right steps towards the destination. The initial steps seem random and doesn't seem to get stuck in local minima once it learns the environment.

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?

Following are the parameters tuned:
self.gamma(discount factor) and self.alpa(learning factor)

For 1000 trails:
(gamma =0.4 and alpa = 1) Total steps = 21861; total rewards =17618; total negative rewards = -5694
(gamma =0.5 and alpa = 1) Total steps = 21311; total rewards =17493; total negative rewards = -5609
(gamma =0.9 and alpa = 1) Total steps = 21929; total rewards =17183; total negative rewards = -5770
(gamma =1 and alpa = 1) Total steps = 21546 ; total rewards =17139 ; total negative rewards = -5716
(gamma =0.5 and alpa = 0.5)Total steps = 21599 ; total rewards =17132 ; total negative rewards = -5716
(gamma =0.5 and alpa = 0.9)Total steps = 21779; total rewards =17254; total negative rewards = -5842

As per Wikipedia: Learning factor (alpa) "A factor of 0 will make the agent not learn anything, while a factor of 1 would make the agent consider only the most recent information. In fully deterministic environments, a learning rate of 1 is optimal."

I consider smartcab environment to be fully deterministic and alpa of 1 seems to be right choice.

Discount Factor(gamma): “The discount factor γ\gamma determines the importance of future rewards. A factor of 0 will make the agent "myopic" (or short-sighted) by only considering current rewards, while a factor approaching 1 will make it strive for a long-term high reward.”

Gamma of 0.5 seems to generate best performance by reducing number of steps and increasing rewards on average for same number of trails.

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?

I would define optimal policy by measuring the number of steps, total rewards and total negative rewards that it receives as it moves along to the destination. Lower number of steps and higher rewards indicate optimal policy.
The Q-Learning method receives lower total negative rewards as it learns the reward system while moving to the destination.