Implementing a basic driving agent.

When implementing the basic driving agent by initially defining action as:

```
action = random.choice((None, 'forward', 'left', 'right'))
```

The agent would sometimes find its destination and occasionally not (within an extended period of time before terminating the program) find its destination. Furthermore, using simple random choices, the agent would not find its destination within the deadline. The action definition was updated as suggested in a number of forum posts:

```
action = self.next_waypoint
```

With this definition, the agent always finds its destination. The agent seems to find the destination within the deadline and almost always with a positive reward at the end of each trial (this is based on observations of 30 trials.)

Definition of state

The state is defined as the inputs. The state consists of 'light', 'oncoming', 'right', and 'left'. These are the necessary inputs for the agent to make a decision on what action to take. Location was not included as the actual location is not relevant to what action to take since the actions are solely governed by what color the light is, whether there is oncoming traffic, and if it is clear to turn left or right. Finally, deadline is not included in state since the deadline is not *necessarily* required for the agent to take an action.

Implement Q-Learning

When q-learning was initially implemented, the agent took an extremely long time to reach its destination. The agent would repeatedly get stuck in suboptimal actions since it was looking for the max of similar states.

Additional Resources:
Reinforcement Learning 3 - Q Learning
https://www.youtube.com/watch?v=1XRahNzA5bE

Reinforcement Learning 4 - Q Learning Parameters
https://www.youtube.com/watch?v=XrxgdpduWOU
Path Finding Q-Learning Tutorial
http://mnemstudio.org/path-finding-q-learning-tutorial.htm