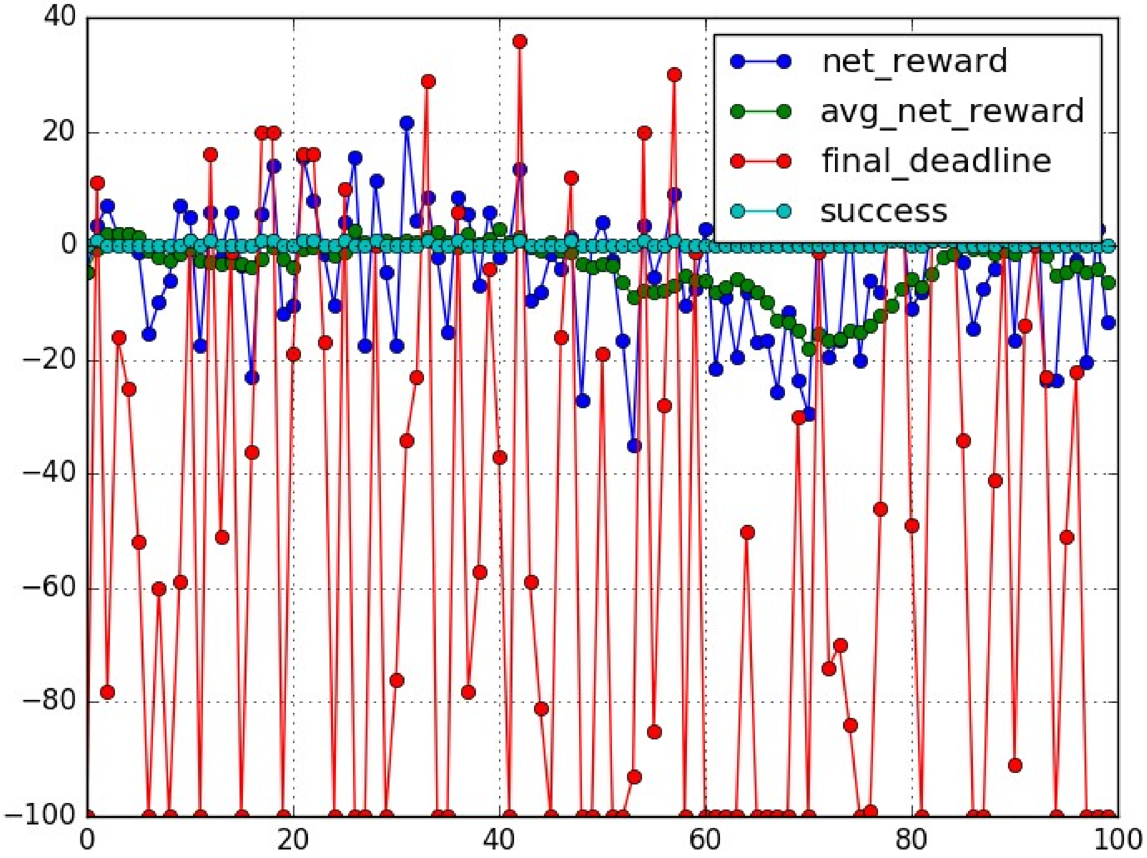
Project 4 Report:
Train a Smartcab to Drive

by: Nicolás Alvarez

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?*

Choosing random actions, the smartcab eventually makes it to the destination. Anyway, the probability of reaching the destination decreases as the number of trials, n_trials, (or deadline if activated) decreases. The reason why the smartcab reaches the destination is because its random driving, each node of the grid (even the destination) has a probability to be reached. If the trials are enough, the destination eventually will be reached.



The route that the smartcab makes to reach the destination (supposing n_trials and deadline enough big) has a very small probability to be an optimal one (or even a suboptimal).

With deadline disable, the simulation finishes the trip when deadline=-100. From the graph above we can see that:

- Only few trips are successful, those that are finished with deadline >= 0.
- Lot of times the smartcab reaches the destination, but with negative deadline.
- Several times the smartcab doesn't reach the destination because the simulator finish the trip when deadline=-100.

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 driving agent is given the following information at each intersection:

- The next waypoint location relative to its current location and heading.
- The state of the traffic light at the intersection and the presence of oncoming vehicles from other directions.
- The current time left from the allotted deadline.

Because the goal is that the agent learn the traffic rules and reach the objective as soon as possible, the states are conformed by the following variables:

- The “inputs” (“light”, “oncoming”, “right” and “left”) will be useful to the smartcab to learn the traffic rules.
- The next_waypoint state will help the smartcab to learn following the route designed by the planner. This will allow to the smartcat to reach the destination following the optimal route.

The deadline does not help in learning the traffic rules. On the other hand, we want the the smartcab to reach the destination as soon as possible, so it neither cares the deadline for learning the optimal route.

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?*

The following table shows the possible values that each state variable could take:

State variable	**Possible values to take**	**No. of possible values**
“light”	red, green	2
“oncoming”	None, 'forward', 'left', 'right'	4
“right”	None, 'forward', 'left', 'right'	4
“left”	None, 'forward', 'left', 'right'	4
“next_waypoint”	forward, right, left	3
	Total states:	**384**

From the table above, there are 384 possible states for the smartcab in this environment. For this simple scenario, it looks a significant number, but still reasonable. Specially because the low traffic (the probability that “oncoming”, “right” and “left” take values different than None is very low) the smartcab must make lot of trips for learning the best action to take in every state.

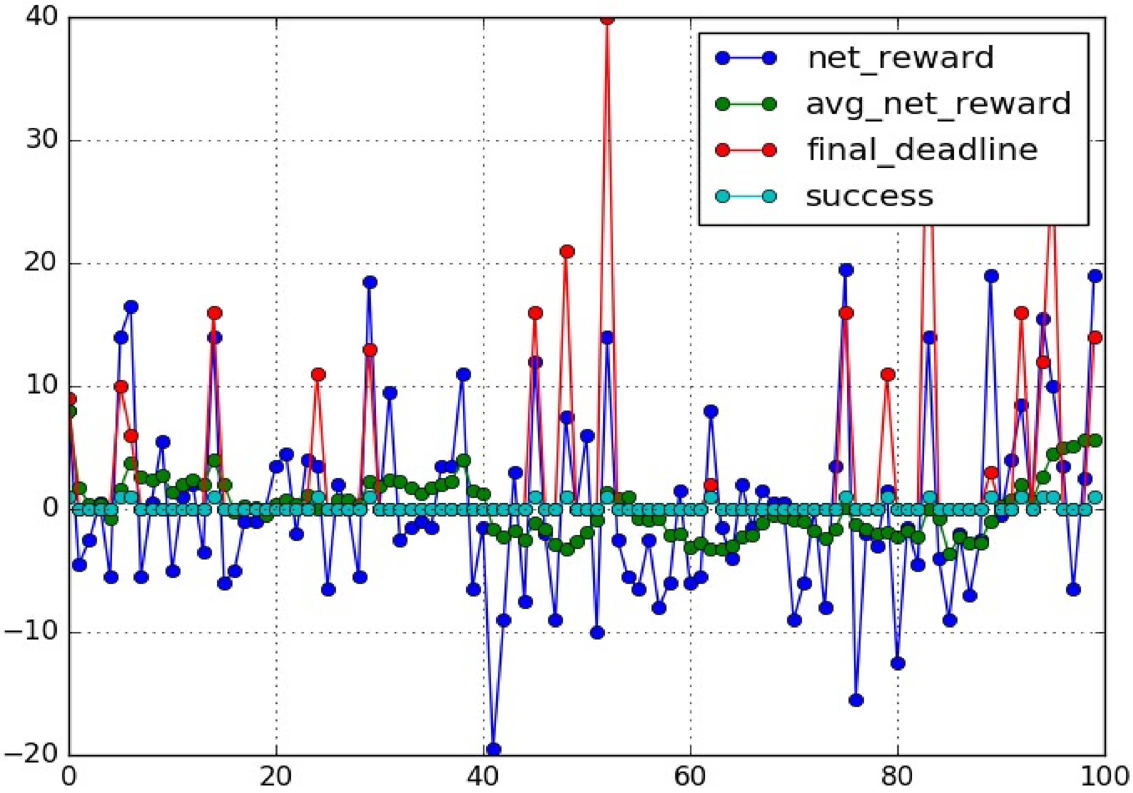
Implement a Q-Learning Driving Agent

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?*

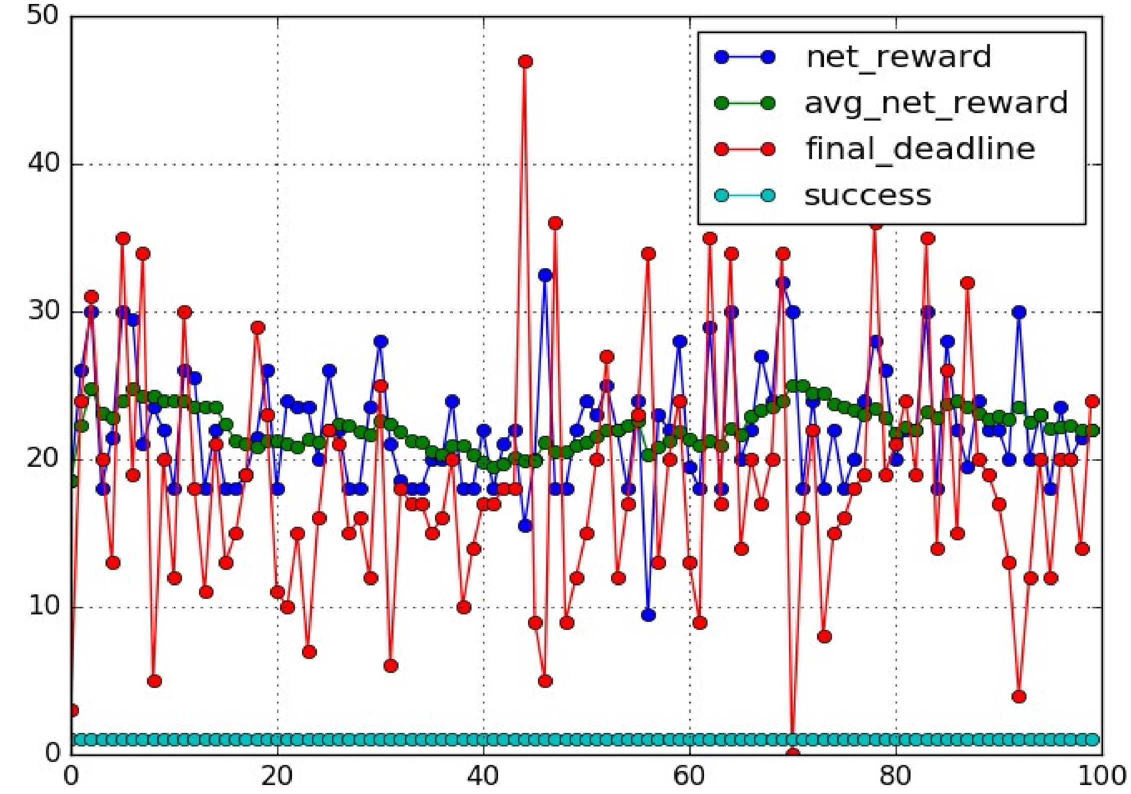
At the beginning, it acts very similar (even equal) to the basic agent, the one who takes only random actions. This is because the Q table is empty, which means that the agent has not learned anything yet.

With the increase of the iterations and trials, the Q-Learning algorithm starts to make its work. The agent learns, given the rewards granted for every state it has visited, what is the best action to perform. The smartcab first learns how to reach the destination following the route given by the planner, but it makes some traffic infractions. This occurs because the traffic density is low, so there are few cases where the smartcat meets some other car in a intersection. With enough trials, the smartcab also learns the traffic rules.

The following two graphs shows the differences between the agent that only makes random decisions and the one who learns using Q-learning algorithm.



With random behavior



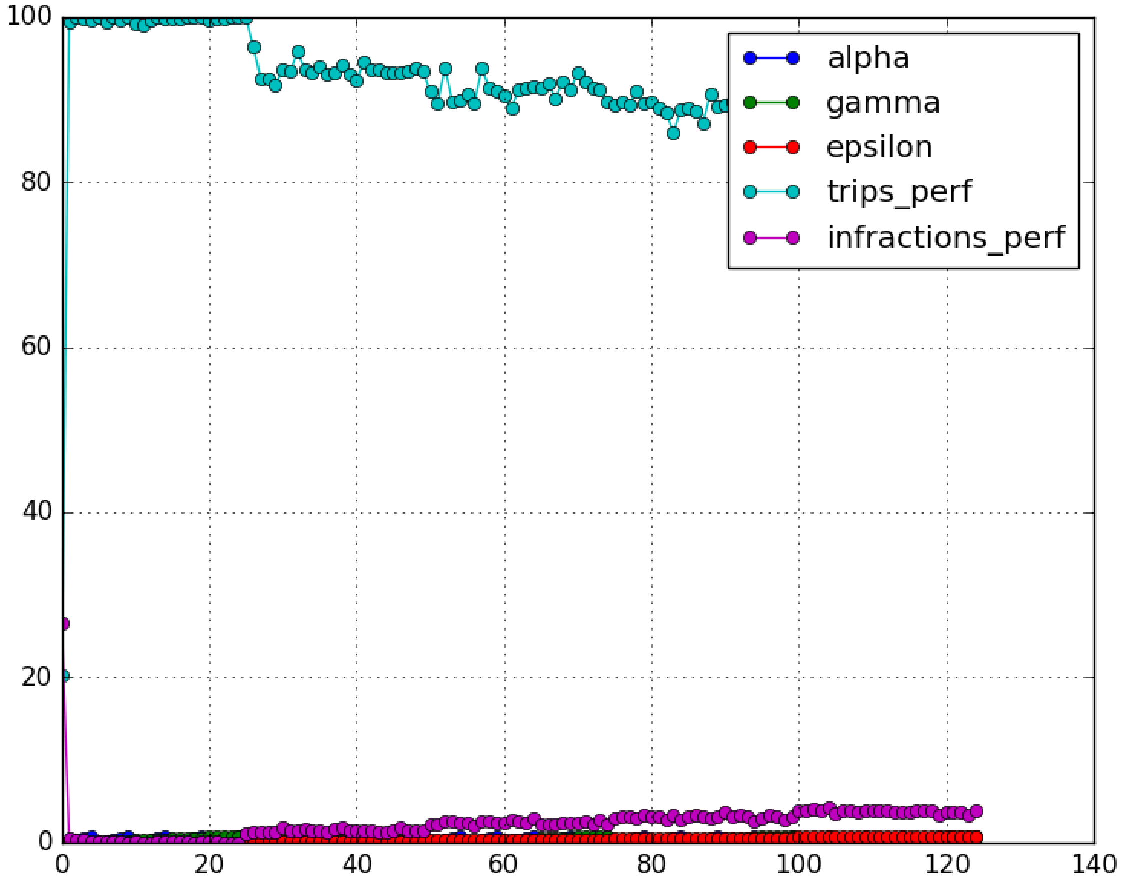
With Q-learning

Improve the Q-Learning Driving Agent

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?*

Varying the parameters alpha, gamma and epsilon from 0 to 1 with a step of 0.2 with 5 averaged simulations, the best configuration set obtained is:

Parameter	***Best configuration for trips performance***	***Best configuration for traffic rules performance***
alpha	*0.4*	*0.4*
gamma	*0.0*	*0.4*
epsilon	*0.0*	*0.0*
trips_perf	*100%*	*99.6%*
infractions_perf	*0.315%*	*0.015%*



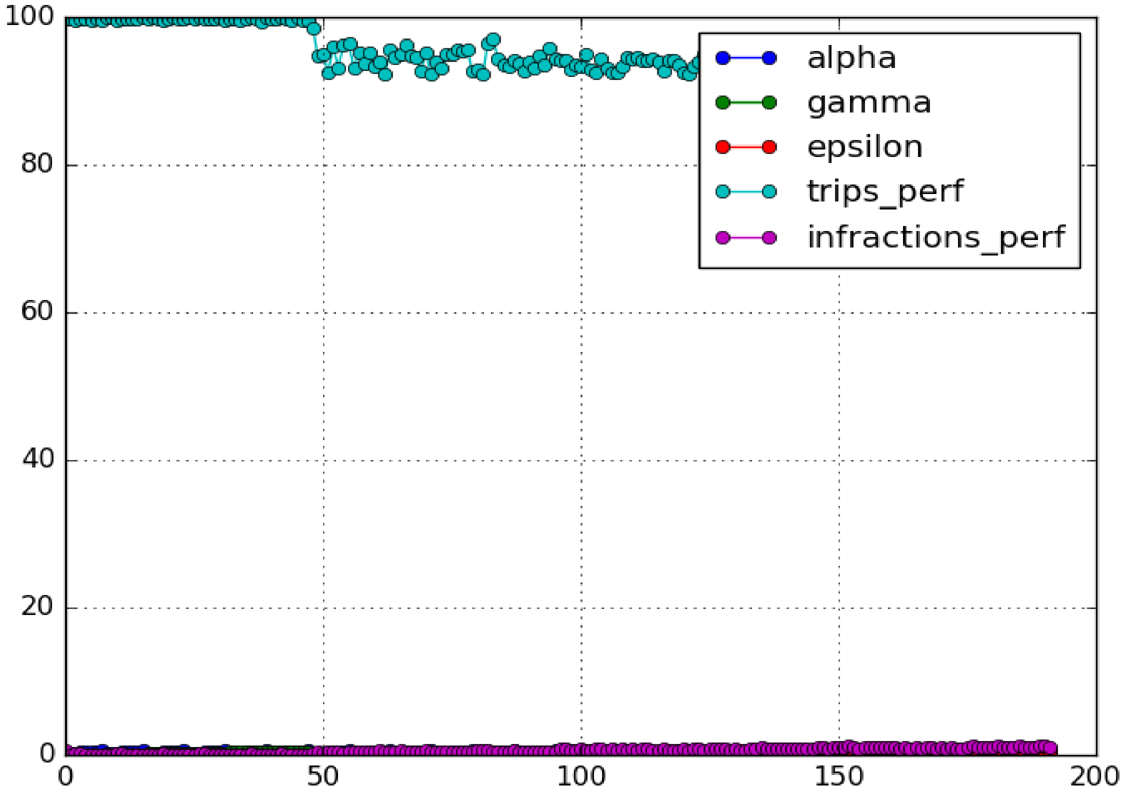
Both parameter sets are very good. The one of the left has better successful trips performance, but makes 0.3% of infractions. The other set has 99,6% of successful trips, very close to 100%, but it makes much less infractions, 0.015% of the movements made.

A new execution was made with the following parameters:

- epsilon, between 0 and 0.2 with a step of 0.05
- gamma, between 0 and 0.6 with a step of 0.1
- alpha, between 0.2 and 0.6 with a step of 0.05

In this case, the best configuration set obtained is:

Parameter	***Best configuration for trips performance***	***Best configuration for traffic rules performance***
alpha	*0.2*	*0.4*
gamma	*0.1*	*0.1*
epsilon	*0.0*	*0.0*
trips_perf	*100*	*99.8*
infractions_perf	*0.078*	*0*



If it is more important to avoid infractions rather than to reach the destination on time, the best set of parameters is (epsilon=0, gamma=0.1, alpha=0.4). Otherwise, the best parameter set is (epsilon=0, gamma=0.1, alpha=0.2).

NOTE: In the annex at the end of the report there is a table with all the results of the second execution.

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 optimal policy should allow the smartcab to reach to the destination in the less possible time following the planner instructions (next_waypoint) and obeying the traffic rules. A smartcab that follows the optimal policy will always reach the destination without making any infraction.

At the beginning, the smartcab makes many mistakes during the learning process. But, after several interactions, it start to perform very well, using a policy very close to the optimal one.

Annex: Performance for different set of parameters

idx	alpha	gamma	epsilon	trips_perf	infractions_perf
0	0.2	0	0	99.8	0.712503
1	0.25	0	0	99.8	0.094308
2	0.3	0	0	99.6	0.139062
3	0.35	0	0	99.8	0.138894
4	0.4	0	0	99.8	0.015613
5	0.45	0	0	99.6	0.015432
6	0.5	0	0	99.8	0.090427
7	0.55	0	0	99.6	0.076784
8	0.2	0.1	0	100	0.078434
9	0.25	0.1	0	100	0.063931
10	0.3	0.1	0	99.6	0.142018
11	0.35	0.1	0	99.8	0.156595
12	0.4	0.1	0	99.8	0
13	0.45	0.1	0	99.8	0.0806
14	0.5	0.1	0	99.8	0.078373
15	0.55	0.1	0	100	0.030688
16	0.2	0.2	0	99.8	0.065096
17	0.25	0.2	0	100	0.031472
18	0.3	0.2	0	99.8	0.03291
19	0.35	0.2	0	99.6	0.123465
20	0.4	0.2	0	99.8	0.015186
21	0.45	0.2	0	100	0.079569
22	0.5	0.2	0	99.8	0.015094
23	0.55	0.2	0	99.8	0.030935
24	0.2	0.3	0	100	0.061145
25	0.25	0.3	0	99.8	0.079565
26	0.3	0.3	0	100	0.108647
27	0.35	0.3	0	99.8	0.109238
28	0.4	0.3	0	99.8	0.07455
29	0.45	0.3	0	99.8	0.044836
…	…	…	…	…	…
162	0.3	0.2	0.15	90.8	0.886972
163	0.35	0.2	0.15	90.8	1.035428
164	0.4	0.2	0.15	93.6	0.783015
165	0.45	0.2	0.15	94	1.058435
166	0.5	0.2	0.15	94.4	0.890644
167	0.55	0.2	0.15	92.8	1.122137
168	0.2	0.3	0.15	92.6	1.048325
169	0.25	0.3	0.15	92.6	0.899177
170	0.3	0.3	0.15	93.6	1.062233
171	0.35	0.3	0.15	93.8	0.862873
172	0.4	0.3	0.15	93.6	0.978501
173	0.45	0.3	0.15	94.6	0.854057

174	0.5	0.3	0.15	94.8	0.905266
175	0.55	0.3	0.15	93.2	1.013696
176	0.2	0.4	0.15	94	1.151247
177	0.25	0.4	0.15	93.4	1.039849
178	0.3	0.4	0.15	94.2	0.979816
179	0.35	0.4	0.15	91.6	1.000104
180	0.4	0.4	0.15	92.4	1.111578
181	0.45	0.4	0.15	94.4	1.207673
182	0.5	0.4	0.15	95	1.070305
183	0.55	0.4	0.15	92.6	1.143915
184	0.2	0.5	0.15	92	1.120941
185	0.25	0.5	0.15	93.6	1.231792
186	0.3	0.5	0.15	94.6	0.969564
187	0.35	0.5	0.15	93.2	0.992482
188	0.4	0.5	0.15	92.8	0.942129
189	0.45	0.5	0.15	92.6	1.350002
190	0.5	0.5	0.15	94	1.241555
191	0.55	0.5	0.15	93	0.999005