Project Report

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: The smartcab does not seem to reach the destination in the trials I observed. It just moves randomly, and doesn’t converge to any position.

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?*

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: The states I identified are a function of the following variables:

- Input of the environment, which gives, the current states of **light**, **oncoming**, **left** and **right** traffic. This information is important for the smartcab in order to gain knowledge of the surroundings and learn when it breaks traffic rules or not
- **Next waypoint**, given by the planner, is an important variable because it indicates the best route the smartcab can take, which is one of the factors of an optimal decision and provides the best rewards
- Deadline, with 2 values, **0 if deadline bigger than 12**, and **1 if deadline was equal or lower than 12**. The objective was for the smartcab to learn if deadline is getting close to zero, it might be beneficial to take some penalties in order to reach the destination.

Therefore, the total states count is:

- **Light**: [red, green] *(2 states)*
- **Oncoming**: [None, forward, right, left] *(4 states)*
- **Right**: [None, forward, right, left] *(4 states)*
- **Left**: [None, forward, right, left] *(4 states)*
- **Next Waypoint**: [forward, right, left] *(3 states)*
- **Deadline:** [0, 1] *(2 states)*

Total states = 2 x 4 x 4 x 4 x 3 x 2 = 768 states

The number seems reasonable, without being too big. There are some states that occur much more frequently than others. These states are the ones with no cars in the intersection, because, in this environment, there are 4 cars for 48 positions, which makes it rather unlikely for multiple cars in the same intersection. The number of these high frequency states are 16.

Hence, the smartcab has the potential to pass through the most important states in a short number of trials. To pass through all the 768 states, it will take a high number of trials.

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?*

ANSWER: The basic implementation was a Q-Learning with epsilon-greedy, done with the following values for alpha, gamma and epsilon:

- $\alpha = 0.5$
- $\gamma = 0.5$
- $\varepsilon = \frac{1}{\left(\frac{k}{2}\right)+1}$, *where k is the trial number, and k/2 is the integer division*

Epsilon was chosen with this formula, in order to not decay too much fast and for the probability of a random action to still be somewhat significant in the latter trials.

The Q-Learning table was initialized with zero for all values. In the occasion there was more than one maximum Q-value, it was randomly chosen one of the actions that had this Q-value. Therefore, at the beginning, the smartcab was exploring the environment with random actions until it begun to learn rewards.

The changes of the agent behavior were noticeable after 10 to 20 trials, where the smartcab learned to take optimal decisions and, therefore, follow traffic rules at the same time as it followed the next waypoint.

This behavior occurred because the smartcab was learning rewards, and updating Q-values on the Q-table, as it executed random actions on the environment (exploration), until it visited enough states to begin exploiting. After 10 to 20 trials, it visited enough of the most common states (following waypoint with no surrounding traffic) to soon begin to take these behaviors.

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?*

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: The tuned parameters in basic Q-Leaning algorithm were the following:

- $\alpha \in \{0.25, 0.50, 0.75, \}$ (learning rate)
- $\gamma \in \{0.25, 0.50, 0.75, \}$ (discount factor)

The epsilon function remained fixed:

- $\varepsilon = \frac{1}{\left(\frac{k}{2}\right)+1}$

The score variables were calculated for the last quartile (25) trials of each 100 trials run. The later trials should reflect the Q-Learning performance, as enough states were learned by the smartcab to have some convergence. The variables calculated were:

- **Success Rate (in %)**: the percentage of times the smartcab reached the destination
- **Average Time Taken (per trial)**: calculated as the average of the time taken for the smartcab to reach the destination. The time taken was calculated as the starting deadline minus the final deadline, when smartcab reached the destination or the deadline was over.
- **Average Mistakes (per trial)**: calculated as the average count of mistakes per trial. Mistakes are counted when the reward for an action is negative.

The performance of the Q-Learning algorithm was evaluated by the following order of priority:

1) Each variable was pre-processed with a Min-Max Normalization
2) The objective was to give more score to Success Rate of 100%, than to take into account Average Time Taken and Average Mistakes. The following score formula was applied to each trial *k*:

    a. **Success Rate Score** $= 1 + \frac{\text{Succ(k)} - \text{Min(Succ)}}{\text{Max(Succ)} - \text{Min(Succ)}}$

    b. **Time Taken Score** $= -\frac{\text{Time(k)} - \text{Min(Time)}}{\text{Max(Time)} - \text{Min(Time)}}$

    c. **Mistakes Score** $= -\frac{\text{Mistakes(k)} - \text{Min(Mistakes)}}{\text{Max(Mistakes))} - \text{Min(Mistakes)}}$

    d. **Total Score** = Success Rate Score + Time Taken Score + Mistakes Score

The transformations to the Min-Max formulas above were applied in order to have a coherent total scoring system, where the best Total Score was the one with the smallest value.

With the above definitions, the **optimal policy** can be defined as the policy which minimizes the Total Score, by having 100% Success Rate, zero Mistakes and the least Time Taken, which should be close to 7 (maximum distance of smartcab and any destination and shortest path taken while encountering no traffic).

The results of iterating each alpha and gamma values were:

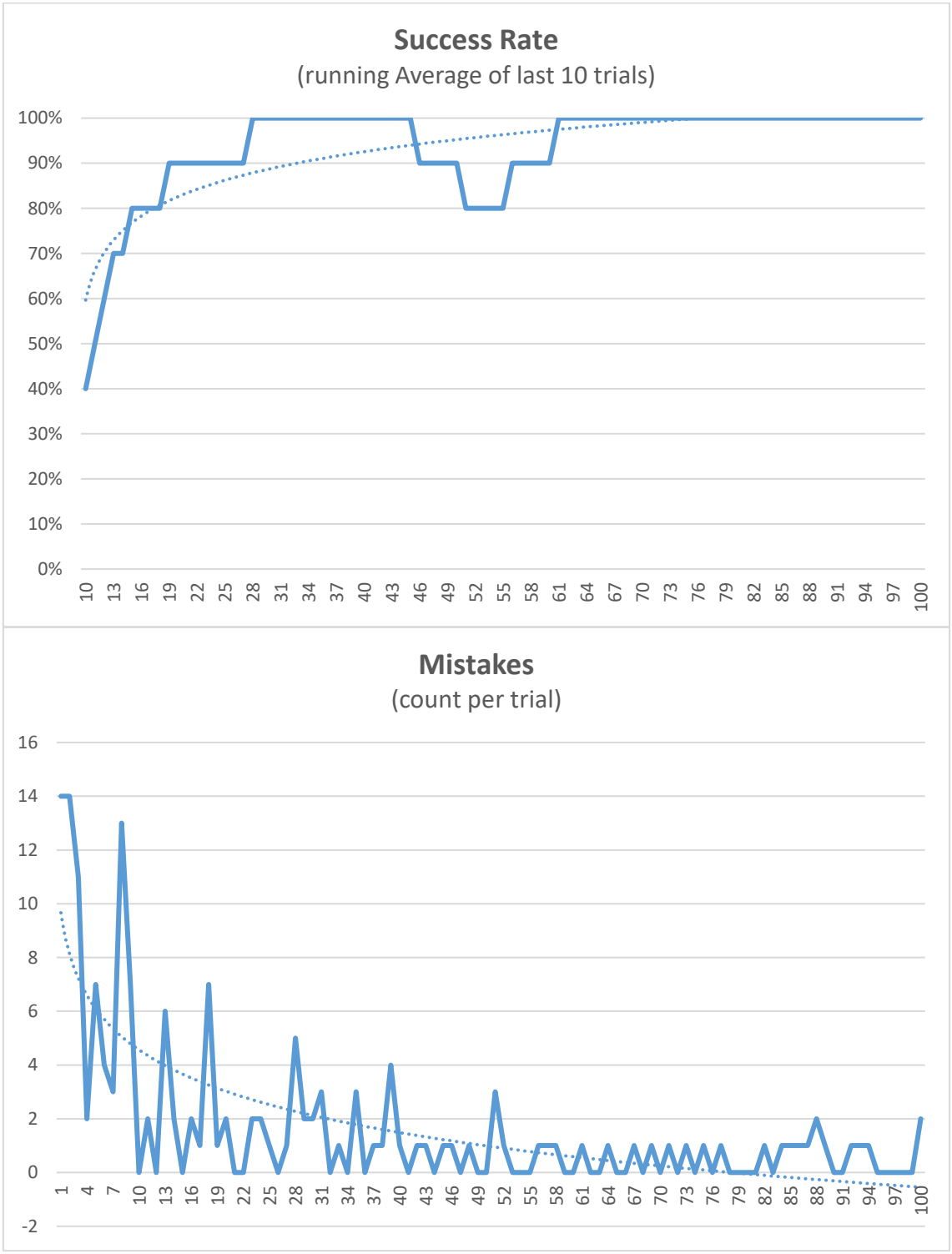
alpha	gamma	Success Rate	Avg. Time	Avg. Mistakes	T_Score	M_Score	S_Score	Total Score
0,25	0,25	96%	13,8	0,520	0,436	0,040	0,500	0,976
0,25	0,50	100%	12,5	0,560	0,242	0,048	0,000	0,290
0,25	0,75	92%	14,1	0,320	0,485	0,000	1,000	1,485
0,50	0,25	100%	12,3	0,440	0,206	0,024	0,000	0,230
0,50	0,50	100%	13,8	1,400	0,436	0,214	0,000	0,651
0,50	0,75	96%	14,8	2,760	0,594	0,484	0,500	1,578
0,75	**0,25**	100%	10,9	0,560	0,000	0,048	0,000	0,048
0,75	0,50	100%	14,0	0,880	0,467	0,111	0,000	0,578
0,75	0,75	92%	17,5	5,360	1,000	1,000	1,000	3,000

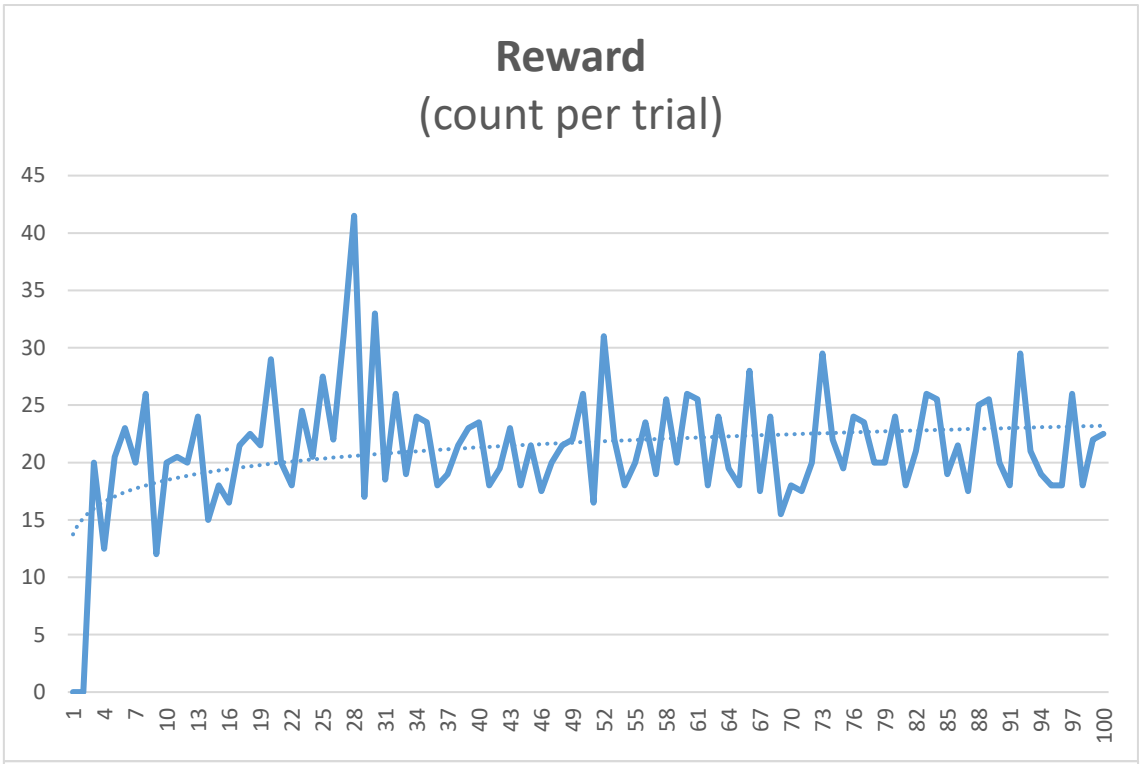
The best values for Alpha and Gamma found, were:

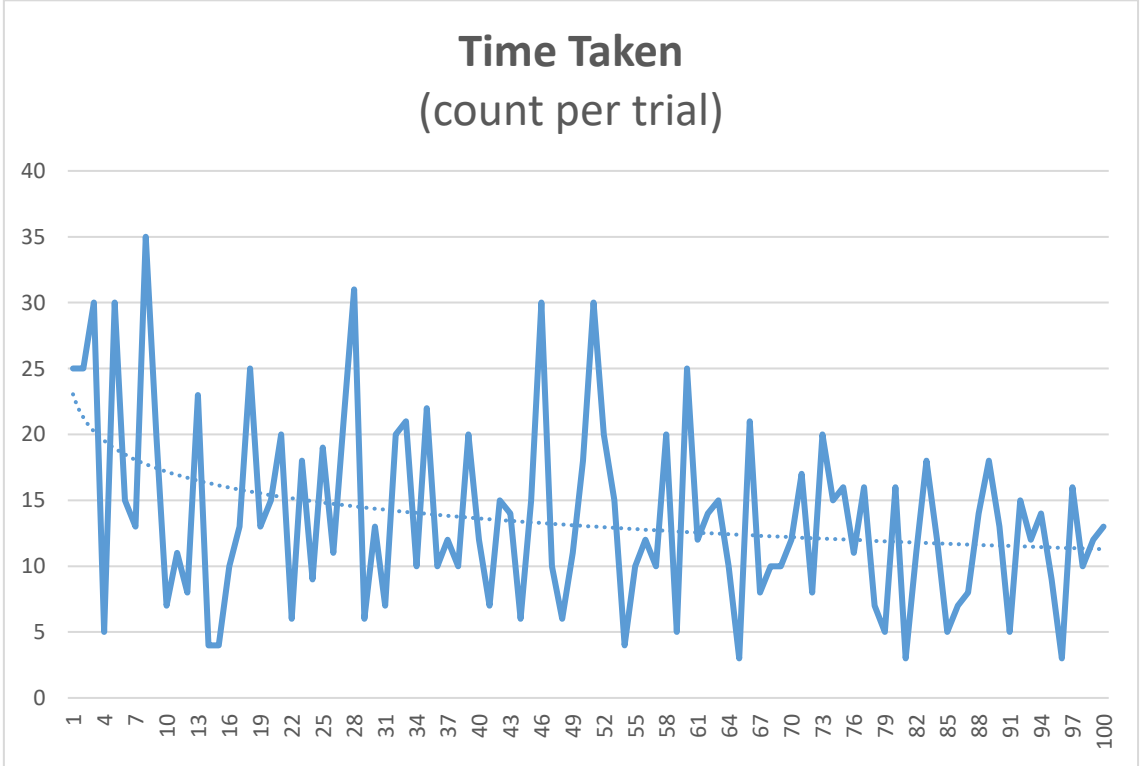
- $\alpha = 0.75$
- $\gamma = 0.25$

The performance of the smartcab with these values was very close to the optimal policy, with 100% Success Rate, less than 1 mistake per trial, and 10.9 steps taken per trail, to reach the destination, which is close to the minimum of 7 steps, only possible in optimum conditions.

This best smartcab's evolution of the different performance indicators can be depicted below:




Reward
(count per trial)
45
40
35
30
25
20
15
10
5
0
1 4 7 10 13 16 19 22 25 28 31 34 37 40 43 46 49 52 55 58 61 64 67 70 73 76 79 82 85 88 91 94 97 100


Time Taken
(count per trial)
40
35
30
25
20
15
10
5
0
1 4 7 10 13 16 19 22 25 28 31 34 37 40 43 46 49 52 55 58 61 64 67 70 73 76 79 82 85 88 91 94 97 100