CSE 4/546: Reinforcement Learning

G. Laxmi Devi (50400730)

PART 1 – Defining RL Environments

1.1 Objective

The environment depicts "A rabbit reaching a bunch of carrots avoiding/passing through hurdles and traps and collecting carrots in the grid environment". The rabbit on its way if it falls into a trap /hurdle gets a negative reward and similarly, if it collects a carrot on its way gets a positive reward.
The main objective is that the rabbit should reach the bunch of carrots for its family avoiding the traps and hurdles and gaining high cumulative reward.

1.2 Actions

A ϵ {up, down, right, left}

1.3 States

The environment is a **4*4** grid that has **16** states. The rabbit starts from the position [0,0] and the goal (a bunch of carrots) is at [3,3].

1.4 Rewards

There are **5** unique rewards in the environment.

R ϵ {0, -0.5, -1, +1, +2}

- Goal (a bunch of carrots) at [3,3] = +2
- Carrots at [1,3] and [2,0] = +1 each
- Hurdle at [3,2] = -0.5
- Trap at [1,1] = -1
- Other positions = 0

1.5 Environment

In the grid environment if the rabbit(agent) chooses an action it can be implemented two ways:

- Deterministic (choice = 1)
- Stochastic (choice = 0)

What is the difference between Deterministic and Stochastic Environments?

Deterministic:

In an environment, if the agent's current state completely determines the next state of the agent the environment is said to be Deterministic which can be represented by $\pi(s) = a$.
Example: Chess
In the defined environment the probability of the chosen action is always 1 (deterministic).

Stochastic:

In an environment where the agent's current cannot completely determine the next action (random in nature) is said to be Stochastic which can be represented by $\pi(a|s) = P_\pi[A = a|S = s]$.
Example: Self-driving cars
The probability of the agent going to the chosen action is not always 1, there is a probability that the agent can go in the wrong direction.

How did you define the Stochastic Environment?

In the defined environment when the action is chosen a decimal number is randomly picked between (0,1). If the decimal number (probability) generated is greater than or equal to **0.8,** random action is selected other than the chosen action.

1.6 Visualizations

Figure 1: Rabbit at the initial position.
Figure 2: Rabbit collected carrot1.
Figure 3: Rabbit caught in a trap.
Figure 4: Rabbit caught in a hurdle.
Figure 5: Rabbit collected carrot2.
Figure 6: Rabbit reached the goal.

Figure 1

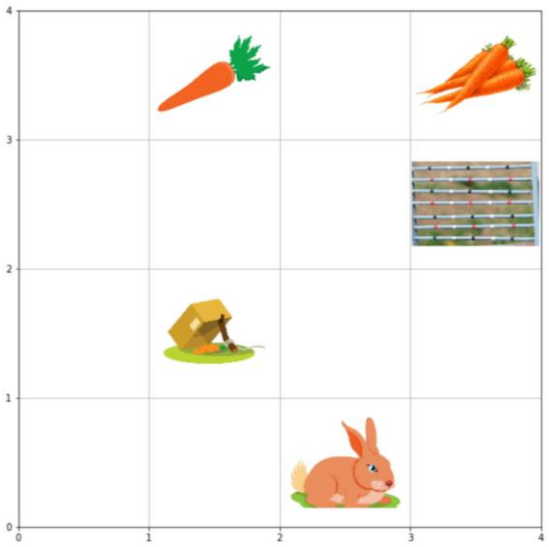


Figure 2

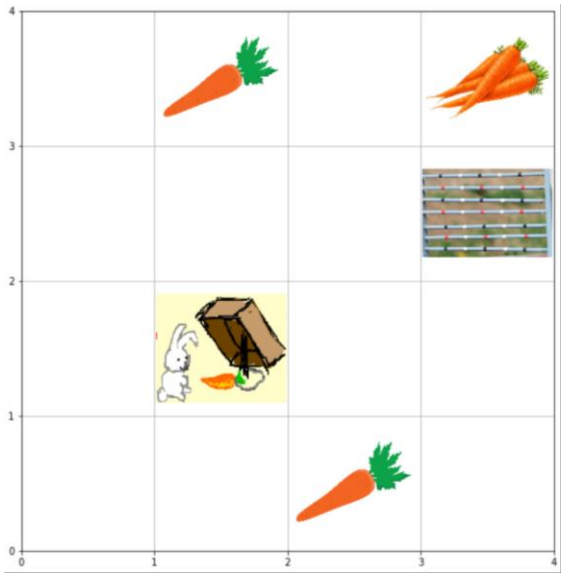


Figure 3

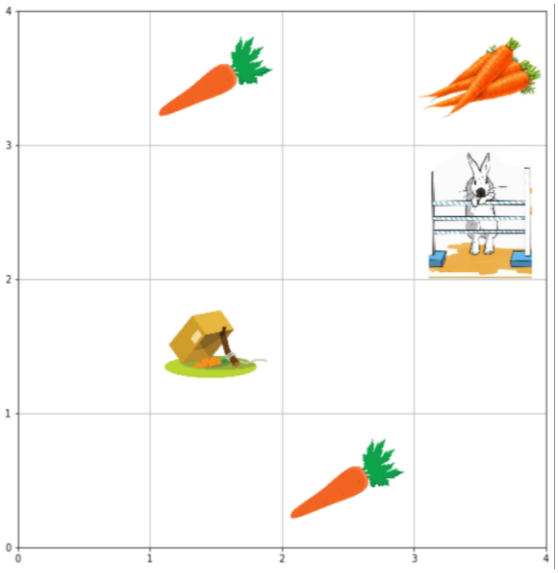


Figure 4

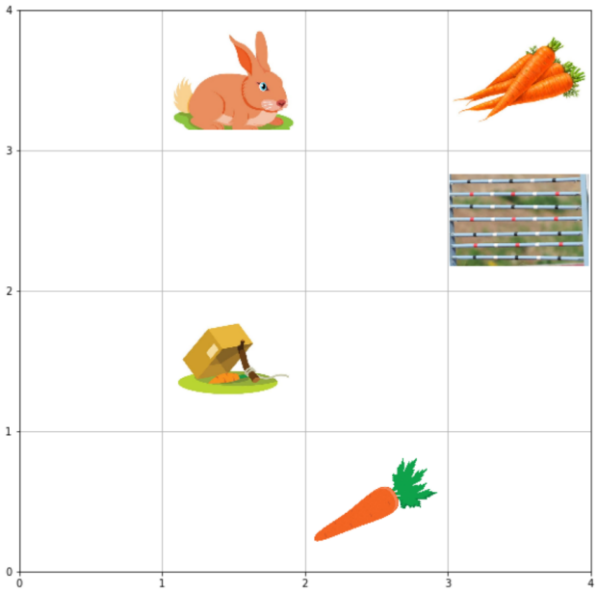


Figure 5

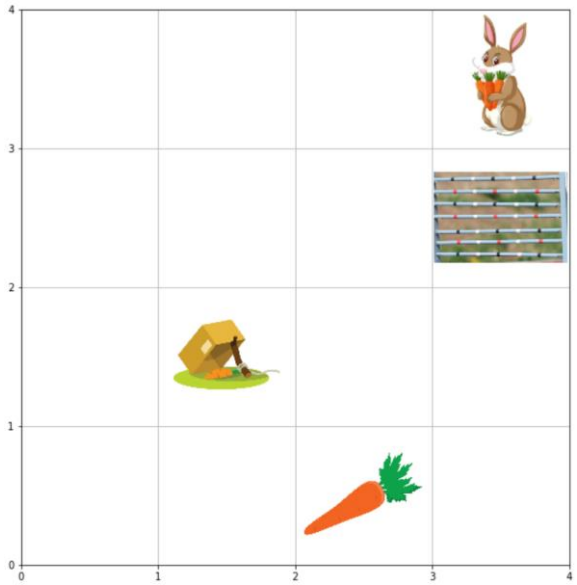


Figure 6

1.7 Safety in AI

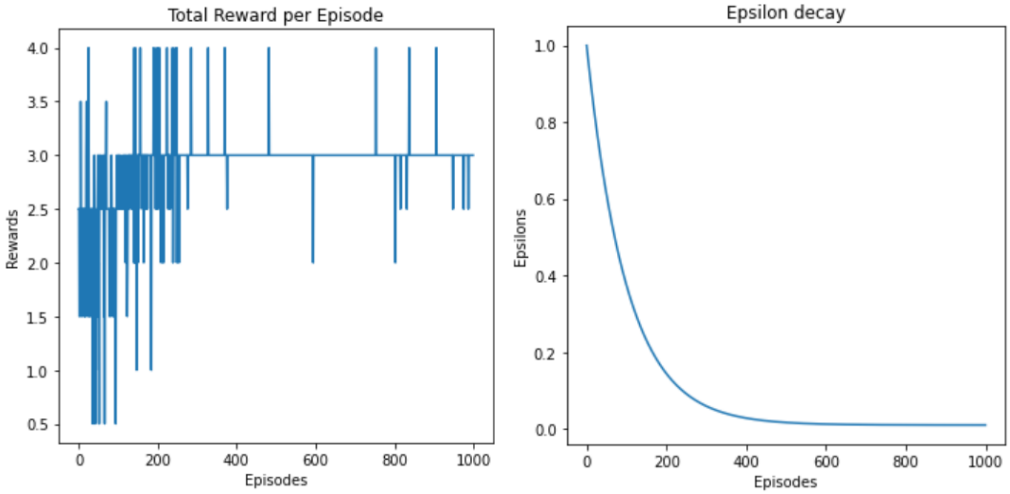
In the defined environment if the action chosen takes the agent out of the grid (4*4 environment), the agent has been restricted to stay in the same position without going out of the grid using the clip() function. This ensures that the agent stays within the environment.

There are hindrances in the environment like a trap and a hurdle for the rabbit (agent) which give negative rewards to the agent. Having these negative rewards will make the agent learn and adapt to the environment such that it will avoid these hindrances and reach the goal.

PART 2 – Applying Tabular Methods

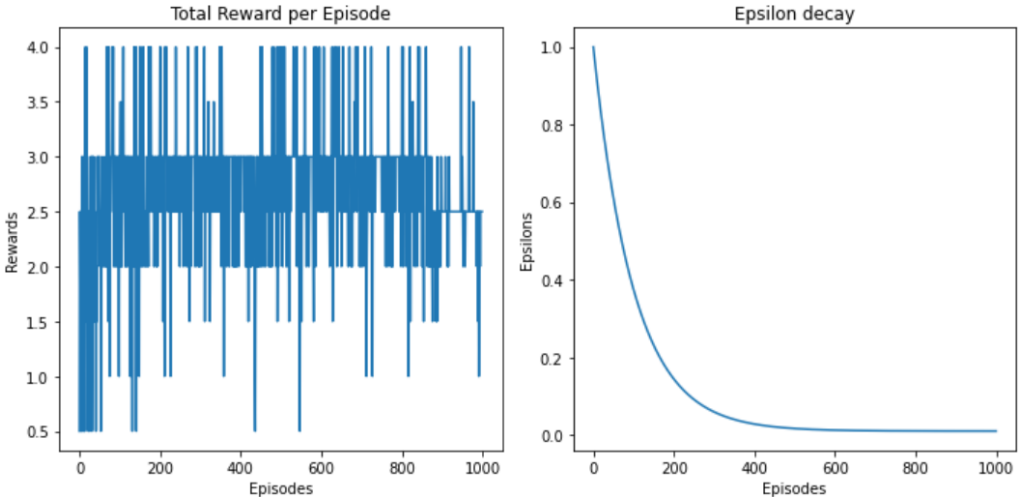
2.1 Results

2.1.1 Q-learning (Deterministic)



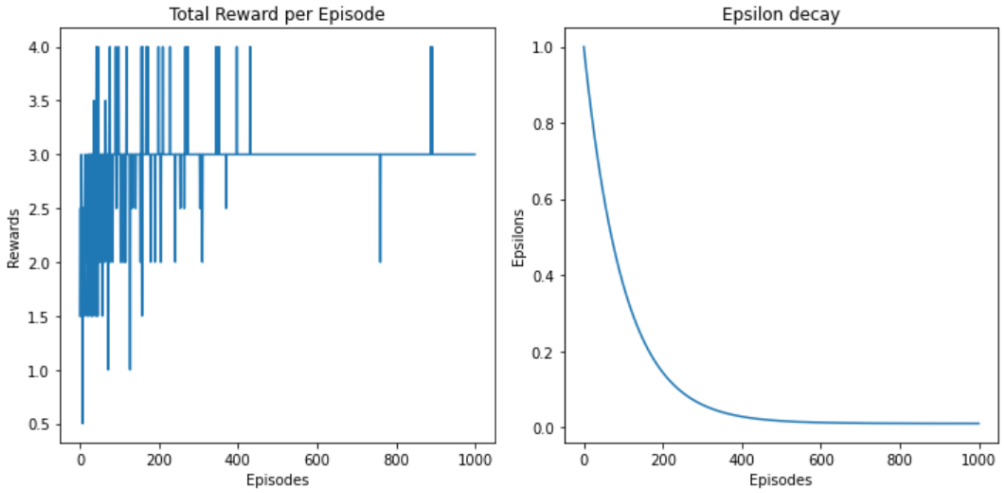
- Epsilon (parameter used for the epsilon-greedy action selection) with a decay factor (0.01) keeps exploring the environment till it reaches the minimum value (0.01). Having a high decay factor will not allow the agent to explore the environment and cannot find the optimal path.
- In this environment the agent explores for around 400 episodes as shown in the graph above.
- The Total Rewards graph shows that during the exploration phase the rewards are very low. The agent starts collecting rewards and avoiding the penalties and gradually the cumulative reward per episode increases (collecting the rewards).

2.1.2 Q-learning (Stochastic)



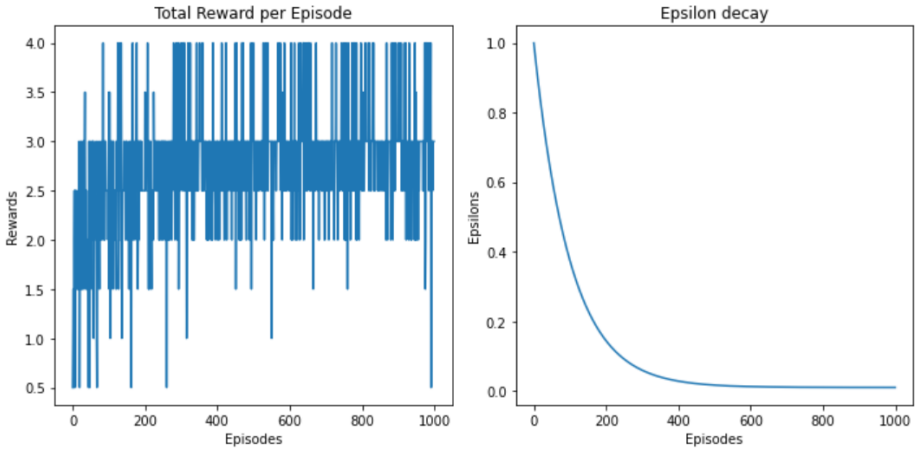
- Epsilon (parameter used for the epsilon-greedy action selection) with a decay factor (0.01) keeps exploring the environment till it reaches the minimum value (0.01). Having a high decay factor will not allow the agent to explore the environment and cannot find the optimal path.
- In this environment the agent explores for around 400 episodes as shown in the graph above.
- Due to the stochasticity in the environment the rewards fluctuate though during the exploration phase the rewards are very low gradually the cumulative reward increases, and the agent collects the maximum rewards in more episodes than the deterministic environment.

2.1.3 SARSA (Deterministic)



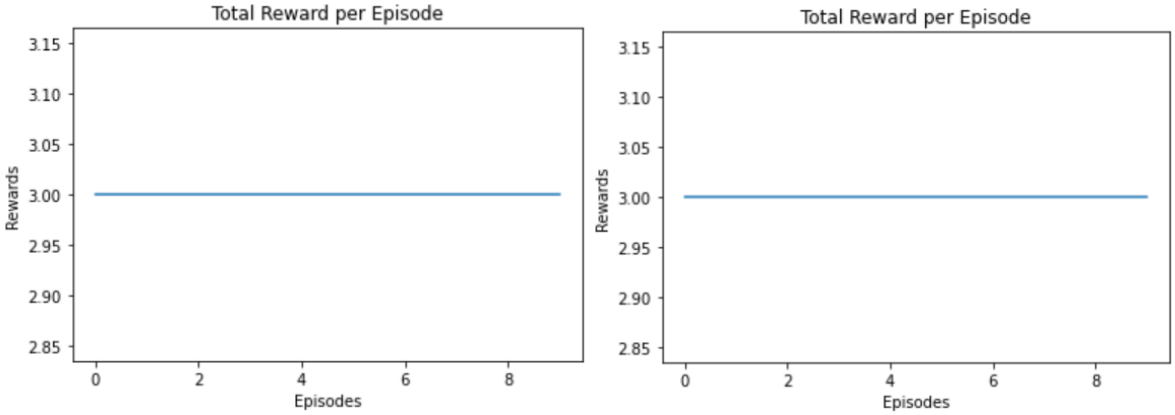
- Epsilon (parameter used for the epsilon-greedy action selection) with a decay factor (0.01) keeps exploring the environment till it reaches the minimum value (0.01). Having a high decay factor will not allow the agent to explore the environment and cannot find the optimal path.
- In this environment the agent explores for around 300 episodes as shown in the graph above.
- The Total Rewards graph shows that during the exploration phase the rewards are very low. The agent starts collecting rewards and avoiding the penalties and gradually the cumulative reward per episode increases (collecting the rewards). The cumulative reward is almost static after the exploration.

2.1.4 SARSA (Stochastic)



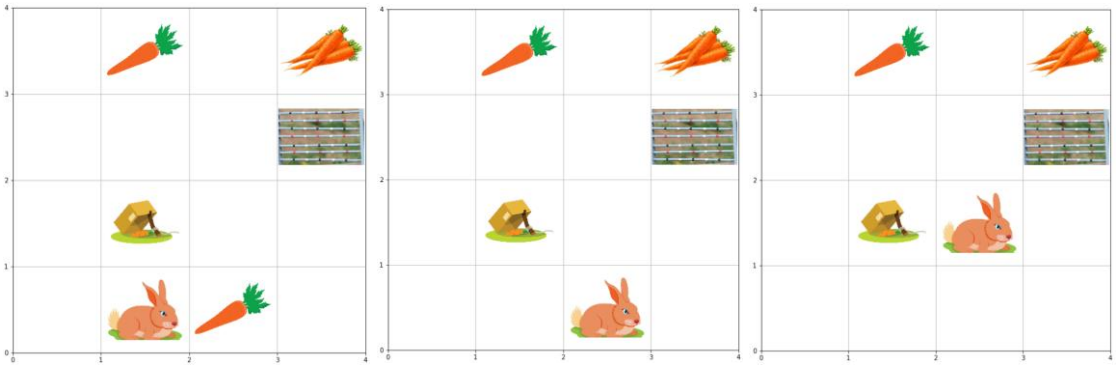
- Epsilon (parameter used for the epsilon-greedy action selection) with a decay factor (0.01) keeps exploring the environment till it reaches the minimum value (0.01). Having a high decay factor will not allow the agent to explore the environment and cannot find the optimal path.
- In this environment the agent explores for around 300 episodes as shown in the graph above.
- Due to the stochasticity in the environment the rewards fluctuate though during the exploration phase the rewards are very low gradually the cumulative reward increases, and the agent collects the maximum rewards in more episodes than the deterministic environment.

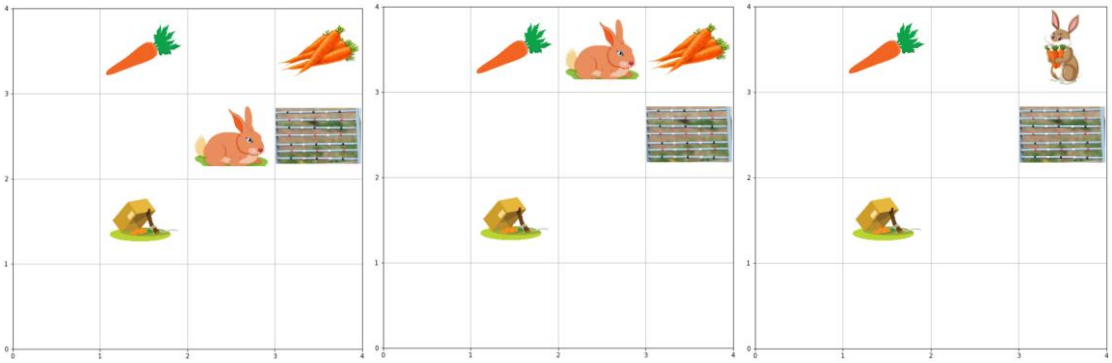
2.1.5 Q-learning and SARSA (Deterministic) - Evaluation



Q-Learning SARSA

Visualisation for 1 Episode (Q-learning):





Visualisation for 1 Episode (SARSA):

Q-learning and SARSA (Stochastic) - Evaluation

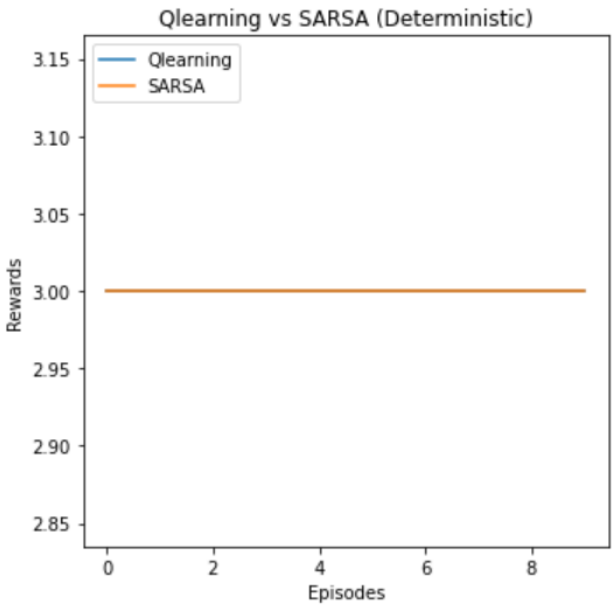
Q-Learning

SARSA

Visualisation for 1 Episode (Q-learning):

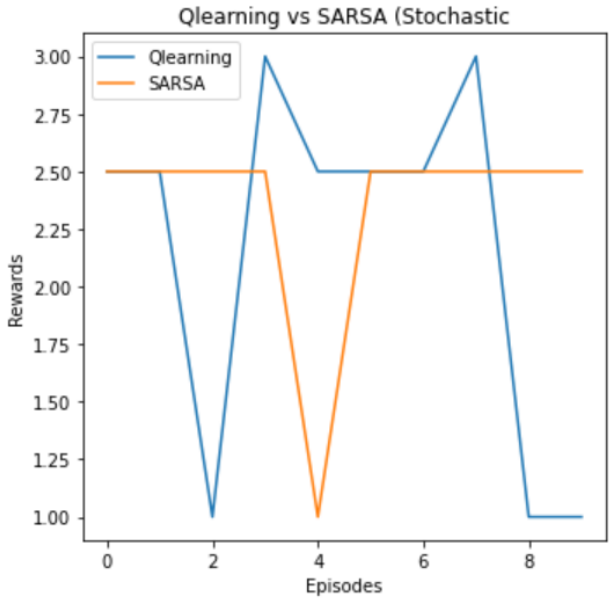
Visualisation for 1 Episode (SARSA):

2.2 Q-learning VS SARSA (Deterministic)

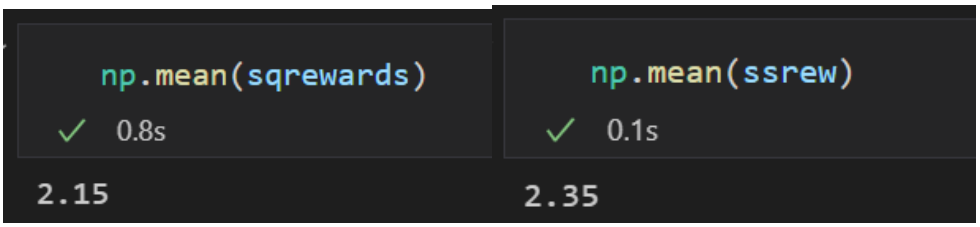


- Since it is a deterministic environment the cumulative reward per episode will be constant, using the optimal policy that the agent has learned.
- Due to the environment the optimal policy in Q-learning and SARSA is the same because that is the only optimal way to reach the goal by gaining rewards though the agent doesn't collect all the rewards.
- Hence the average cumulative reward is same for both the algorithms. There is no visible change in the performance.

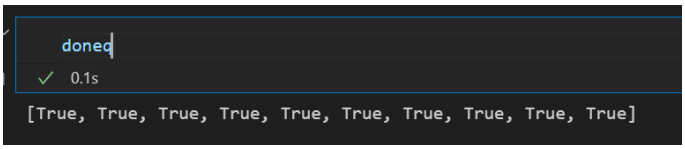
2.3 Q-learning VS SARSA (Stochastic)

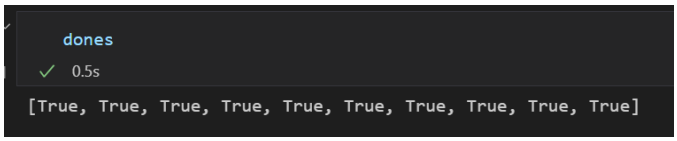


- In the case of a Stochastic environment, the cumulative reward per episode is not constant and keeps fluctuating as shown in the above plot.
- In case of Q-learning the average cumulative reward is **2.15** and for SARSA the average cumulative reward is **2.35**.



- From which we can say that SARSA is better than Q-learning in a Stochastic Environment.
- The higher rewards say that in SARSA agent has not got more penalties than in Q-learning.
- Also, In both the algorithms the agent has reached the goal in a Stochastic Environment. Though it didn't reach the goal in 6 steps(optimal). The agent reached the goal in 10 steps which means the agent has learned the policy.
- 30% Stochasticity – 6 + (0.3) *6





2.4 Tabular Methods

- **Q-Learning:**

1. Q-learning is a model-free, off-policy reinforcement learning that will find the best course of action, given the current state of the agent. Depending on where the agent is in the environment, it will decide the next action to be taken.
2. The objective of the model is to find the best course of action given its current state. To do this, it may come up with rules of its own or it may operate outside the policy given to it to follow. This means that there is no actual need for a policy, hence we call it off-policy.
3. Model-free means that the agent uses predictions of the environment's expected response to move forward. It does not use the reward system to learn, but rather, trial and error.
4. An example of Q-learning is an Advertisement recommendation system.

Update Function:

$$\boldsymbol{Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha\left(R_{t+1} + \gamma \max_{A'} Q\,(S_{t+1}, A') - Q(S_t, A_t)\right)}$$

- **SARSA:**

    1. SARSA stands for **State Action Reward State Action** which symbolizes the tuple (s, a, r, s', a').
    2. SARSA is an on-policy algorithm which means that while learning the optimal policy it uses the current estimate of the optimal policy to generate the behavior.
    3. SARSA converges to an optimal policy if all state-action pairs are visited an infinite number of times and the policy converges in the limit to the greedy policy ($\epsilon = 1/t$).

    Update Function:

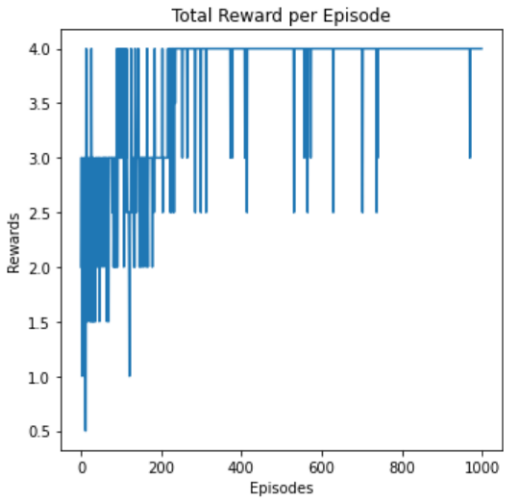
$$\boldsymbol{Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha[R_{t+1} + \gamma Q(S_{t+1}, A_{t+1}) - Q(S_t, A_t)]}$$

3. Extra Points

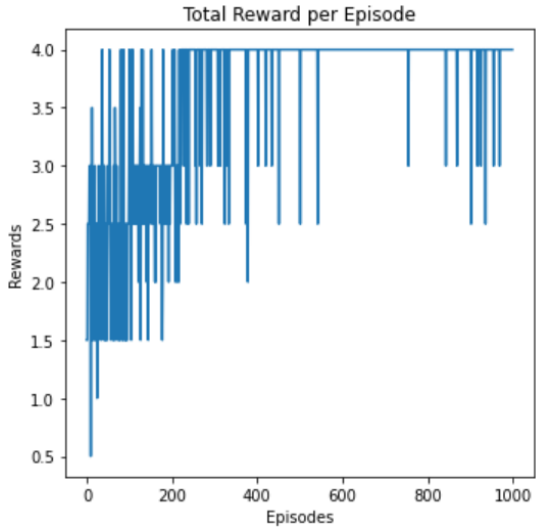
Hyper Parameter Tuning

- Discount Factor γ

    When $\gamma = 0.8$,



    When $\gamma = 0.9$,

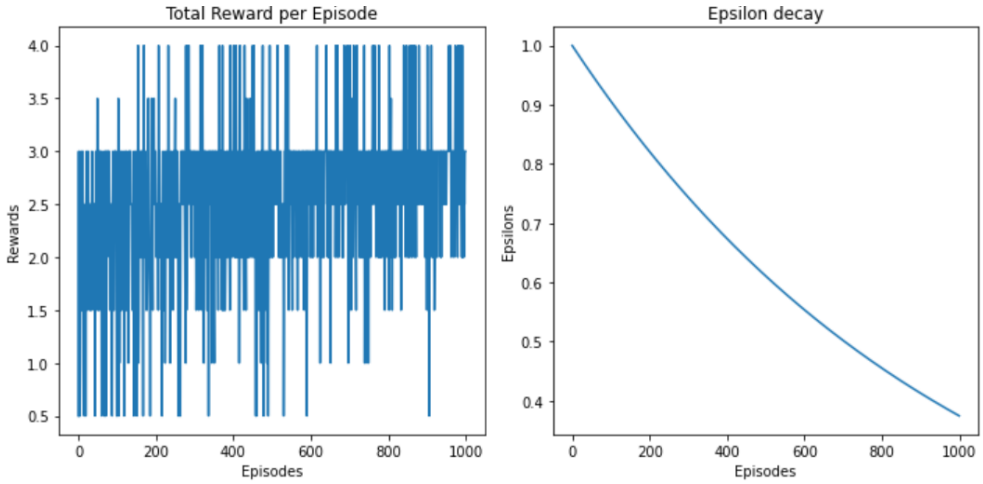


When $\gamma = 0.6$,

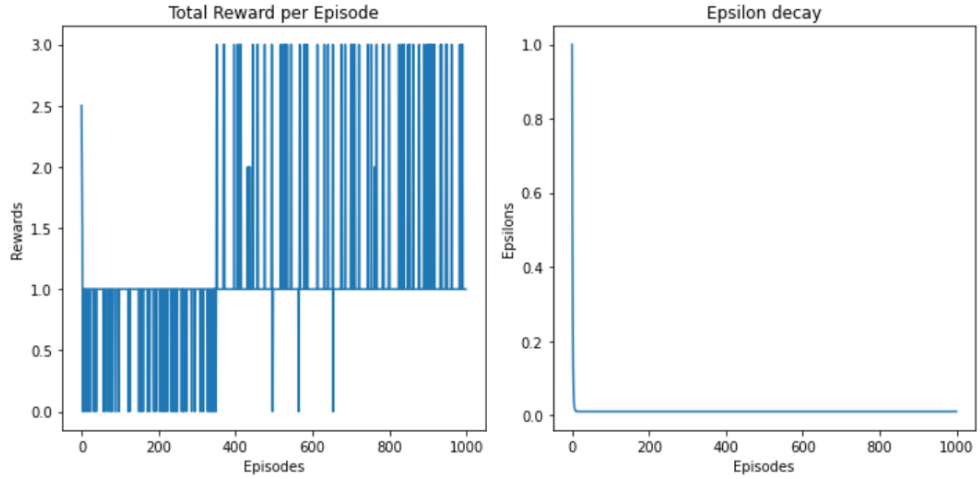
When the discount factor is greater than 0.8 the agent looks for the immediate rewards collecting all the rewards in the environment without learning the optimal policy. From the above plots we can see that when discount factor is greater than 0.8 the average cumulative reward is 4 and when it is 0.6 the reward is 3. Though the reward is less in the latter case the agent reaches the goal using optimal policy rather traversing the environment collecting the immediate rewards. Hence, for this environment 0.6 is a good discount factor.

- Decay Factor

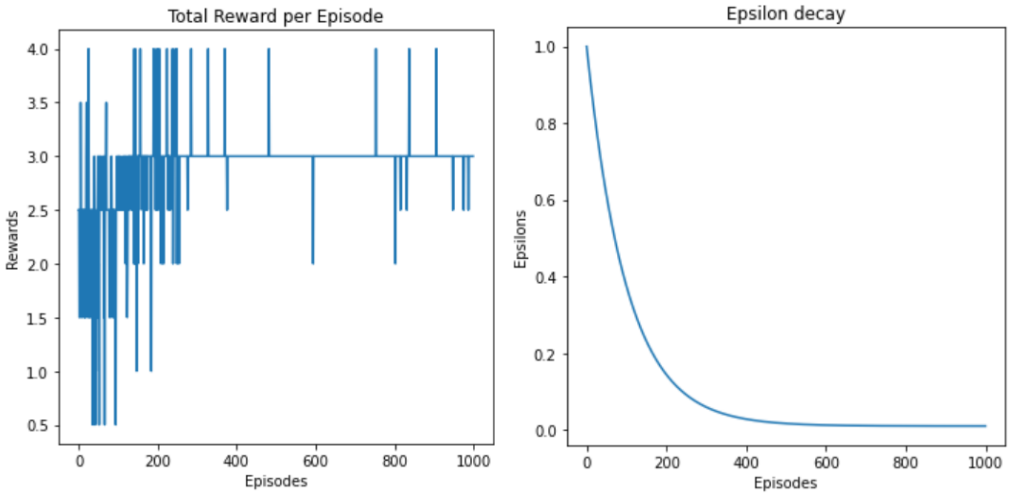
  When decay factor =0.001,



  When decay factor =0.99,



When decay factor =0.01,



When the decay factor is too low (0.001) or too high (0.99) the agent is stuck in a local optima which it keeps traversing between two states because the agent has not properly explored the environment and Q table is not updated. So, it doesn't move forward in the environment. After further tuning of the environment decay factor between (0.01-0.1) allowed the agent to explore the environment and update the Q table to take the greedy action.