Assignment 3

Gunjan Kumar - 2019CS10353 and Nishant Kumar - 2019CS50586

Part A: Computing Policies

1. Formulation of the taxi domain as an MDP:

a. **State Space:** There are 29*25 possible states. There are 25 locations of the passenger if the passenger has not been picked up. For each location of the passenger, we have a state at all the cells of the maze. If the passenger has been picked up, then there are 4 possible destinations, and corresponding to each destination we have a policy/state at each of the cells of the maze.
b. **Action Space:** There are six possible actions at each state - U, D, L, R, pickup, and pick down. The first four are stochastic and the last two are deterministic.
c. **Transition Model:** Pickup and Pick down occur with a probability of 1 and for the rest four actions, action occurs in the intended direction with a probability of 0.85 and other random directions with a probability of 0.05.
d. **Reward Model:** +20 rewarded if dropped the passenger at the intended location. -10 if wrong drop and pickup are attempted. For the rest of the actions, -1 is rewarded.

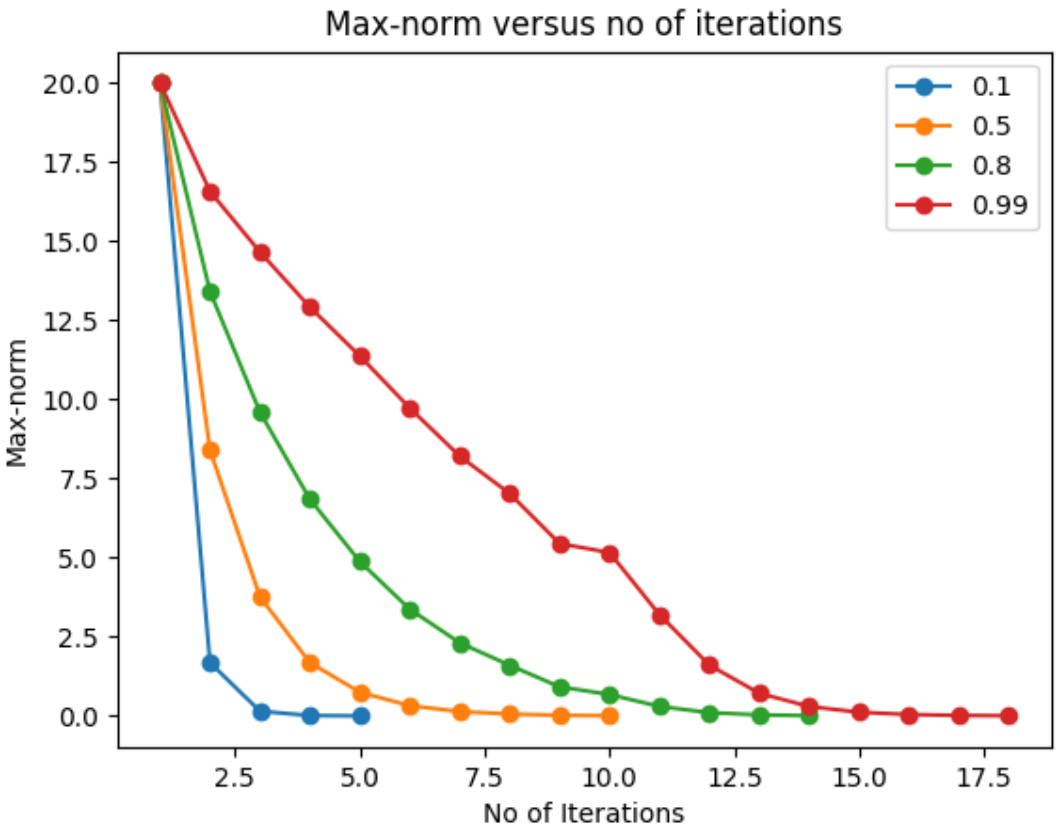
2. Value Iteration for the taxi domain:

a. Epsilon: We have used 0.01 as the value of epsilon. As we will increase the value of epsilon, it will terminate faster but it will also comprise the accuracy of the optimal value obtained.

No of iterations required for convergence: In this case, it takes 16 iterations to convergence to the optimal value.

b. Connection between the discount factor and the rate of convergence

Discount Factor	No of Iterations
0.1	5
0.5	10
0.8	14
0.99	18



As clear from the graph, the rate of convergence is faster if we have lower values of the discount factor. This is because if we have a lower value of the discount factor, then the future rewards will have very less effect on the current state. But, if the discount value is too low, then the MDP may not learn the optimal policy as the

reward obtained at a particular location will not be propagated further to a longer distance.

c. **Simulate the policy obtained for discount factors as 0.1 and 0.99**

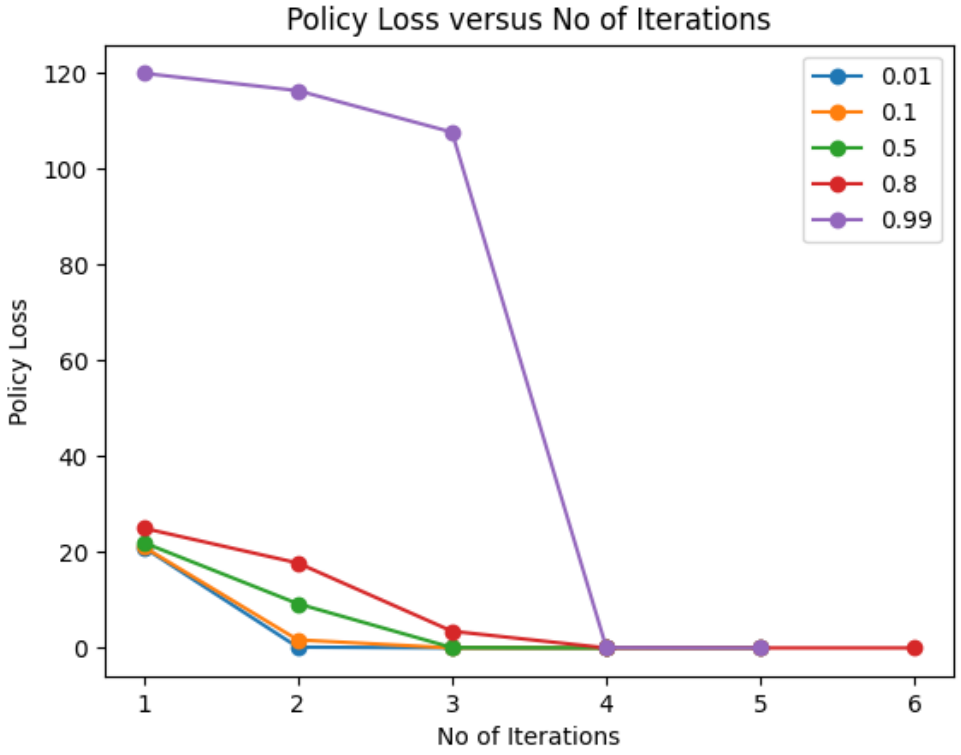
Initial Passenger Location is (0, 0). The final Passenger Location is (4,4) and the initial Taxi Location is (0,4). We have run the value iteration for two discount factors and obtained observation is: The policy obtained in the case of discount factor = 0.99 is optimal. Hence, the taxi is able to pick up the person and drop him at the desired location within 20 steps. Whereas in the case of discount factor = 0.1, the obtained policy is not optimal and the taxi is not able to perform the desired action of picking up and dropping the passenger at the right location.

3. Implement Policy Iteration for the problem

a. The linear Algebra method of Policy evaluation is suited only for the problem having a small space state. If the number of states is very high, then, the matrix inversion will be computationally feasible. In this case, we should use the iterative method of policy evaluation which uses dynamic programming to evaluate the utility of states.

b. For lower values (0.01, 0.1) of the discount factor, the policy does not converge to the optimal policy as it does not care much about the future rewards whereas, in the higher values (0.5, 0.8, 0.99) of the discount factor, the policy converges to the optimal policy.

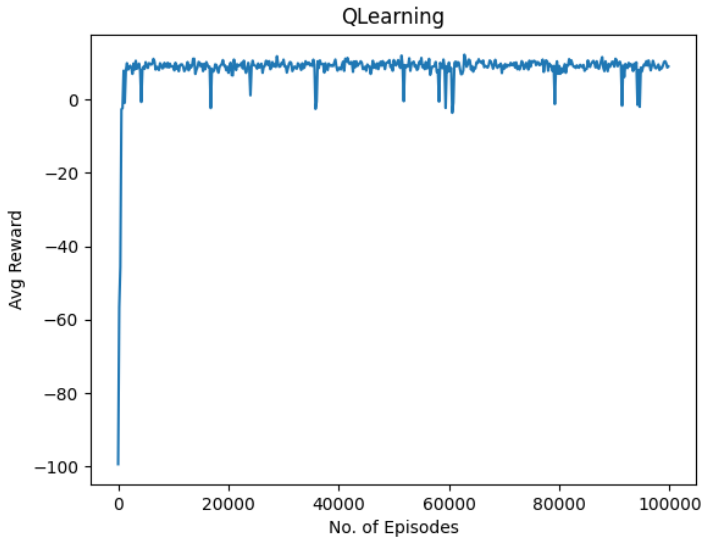
Graph of Policy Loss vs No of iterations is:

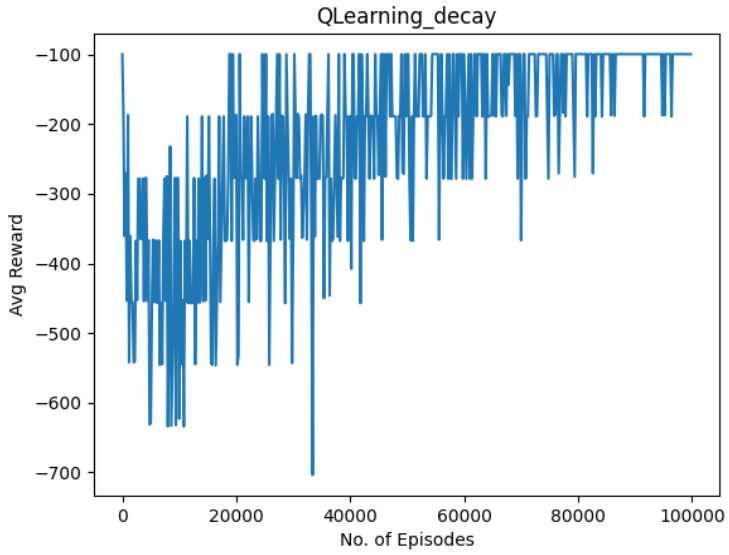


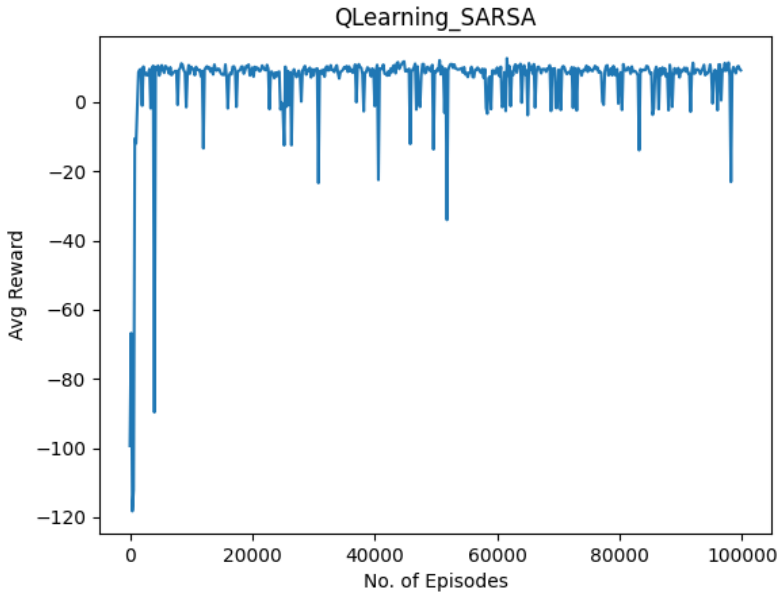
Part B: Incorporating Learning

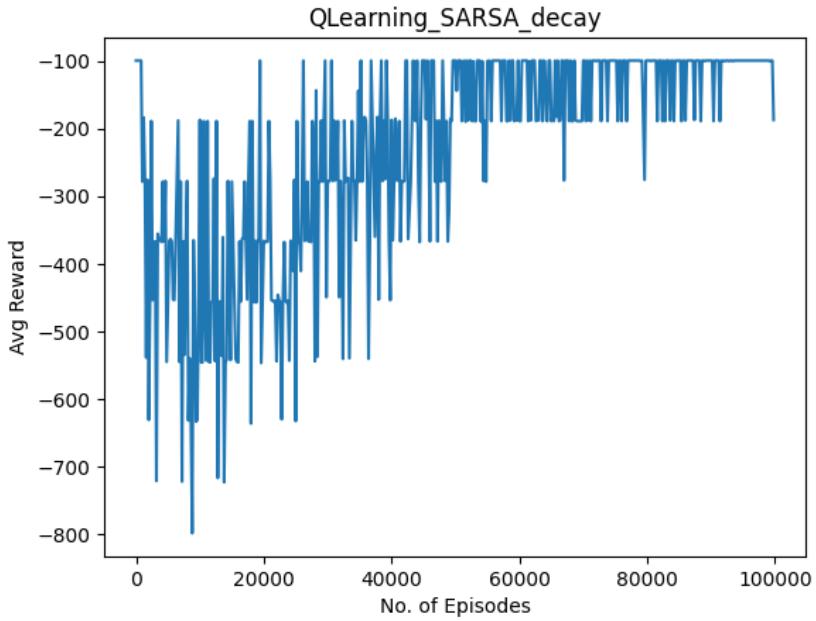
1. **Implementation of the different approaches to learn optimal policy**
    a. Q Learning with epsilon greedy exploration
    b. Q Learning with exponential decay exploration
    c. SARSA Learning with epsilon greedy exploration
    d. SARSA Learning with exponential decay exploration

2. **Sum of discounted rewards of an episode versus the number of training episodes**





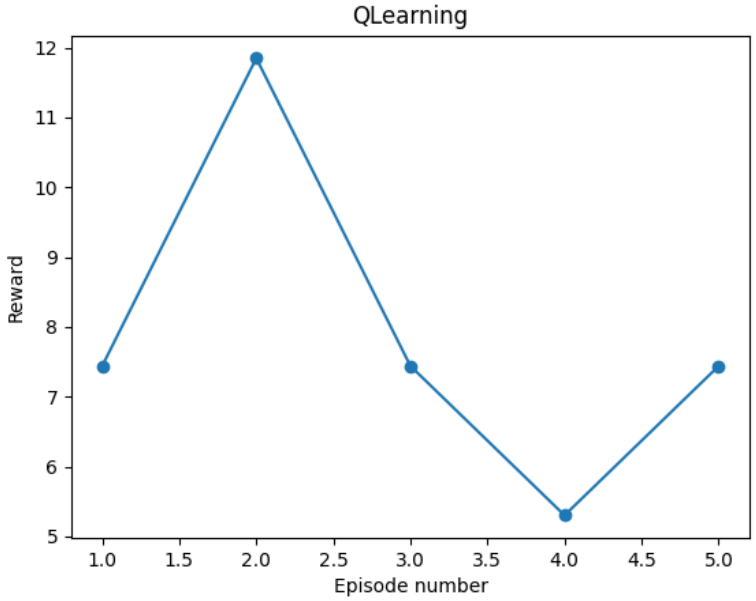




As clear from the above graphs, Q Learning with decaying exploration and SARSA Learning with decaying exploration are not able to learn the optimal policy. This is because these algorithms are exploring less. Hence, they are not exploring even the actions which will fetch the best reward. Further among Q Learning and SARSA Learning, we can infer that Q Learning performs better than SARSA Learning as it is converging faster and shoeing fewer fluctuations in the graph

3. Execution of the policy obtained from the Q Learning:

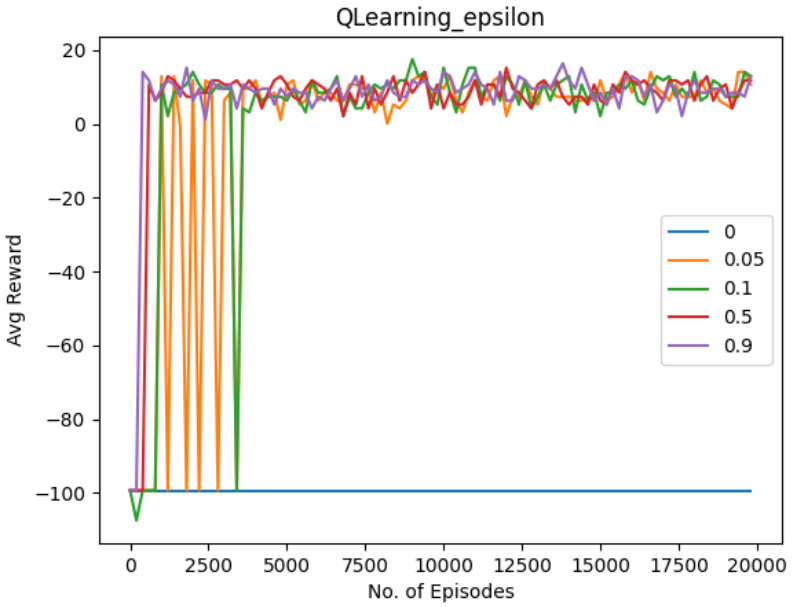
We have executed the optimal policy obtained from the Q Learning for five different instances. The plot of reward versus five episodes is:



Since the destination is within 20 steps. For each action, we get -1, and at the successful drop at the destination, we get +20, so the finally accumulated reward is positive for all five episodes. This also shows that the final policy obtained by Q Learning is optimal. If the taxi has to travel larger destination, then the final accumulated reward is low and vice-versa.

4. Effect of alpha and epsilon:

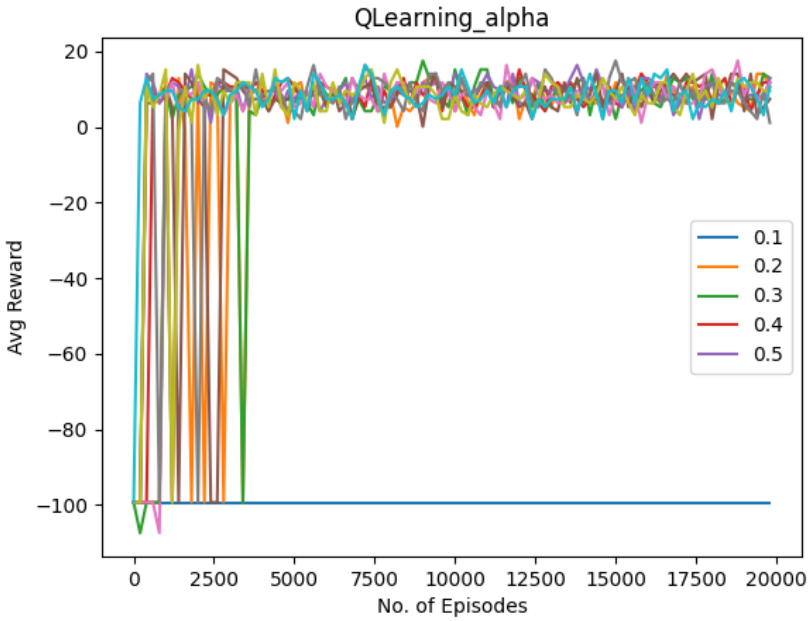
a. Sum of discounted rewards of an episode versus Number of training episodes by varying the exploration rate



As clear from the above plot, if the exploration rate is zero, then the agent is not able to learn the optimal policy thus leading to the constant negative reward of -100. Also, for a lower value of 0.05, it takes more episodes to learn the optimal policy in comparison to the higher exploration rate.

b. Sum of discounted rewards of an episode versus Number of training episodes by varying the learning rate

If the learning rate is low i.e. 0.1, then the agent will take more time to learn the optimal policy. This is because it trusts less on the newly obtained value of the sample.



5. Reinforcement Learning on the Big Maze

The optimal reward obtained is 0.81. Graph of reward vs no of episodes is plotted below: