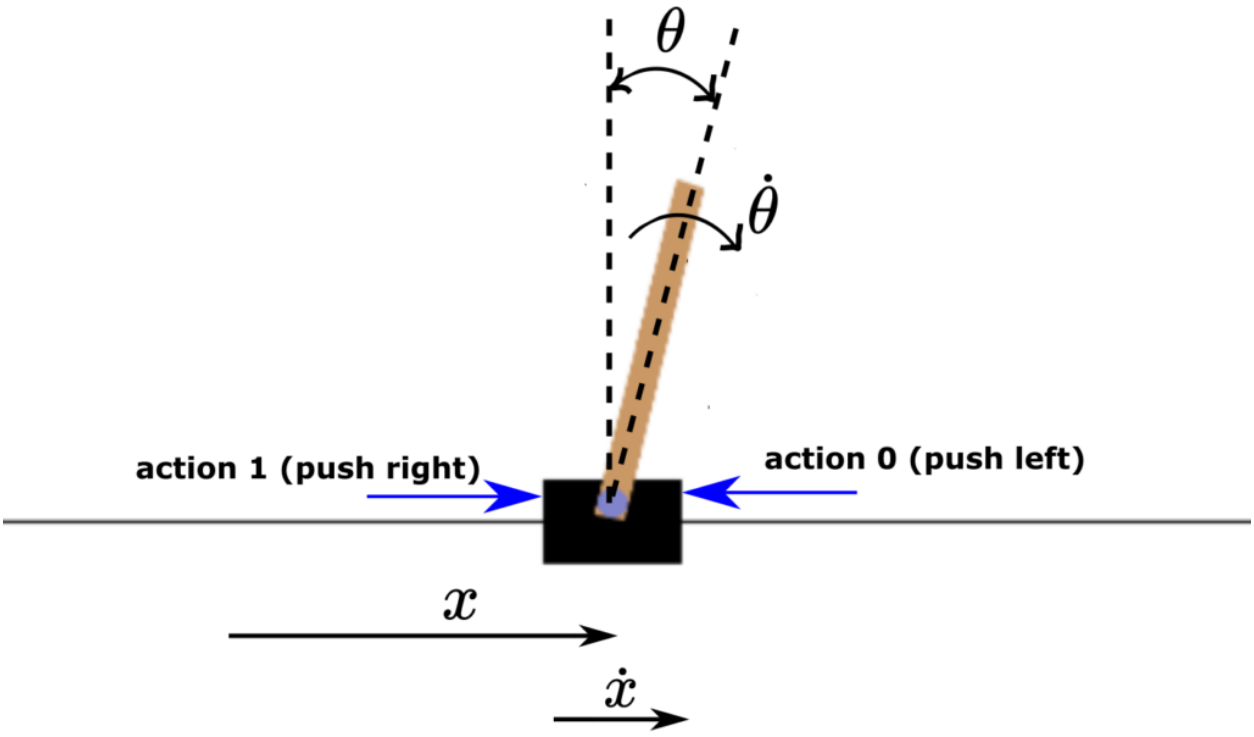
REINFORCEMENT LEARNING

ASSIGNMENT 2

ERP: 25371

INTRODUCTION

In this assignment, we are solving a cart pole task. The Cart Pole task is a classic problem in the field of reinforcement learning. It involves balancing a pole on a cart that can move left or right on a frictionless track. The goal is to keep the pole upright by moving the cart left or right as needed. The environment is characterized by a state, action, and reward.



Here

X = Position of the cart,

X' = Velocity of the cart,

Theta = Angle of rotation of pole

Theta` = Angular velocity.

OBJECTIVE

In this task, we have a cart which contains a pole in the middle. Our goal is to balance the pole and move the cart left and right. The control objective is to keep the pole vertical.

Model IMPLEMENTED

- Q-Learning
- SARSA temporal learning.

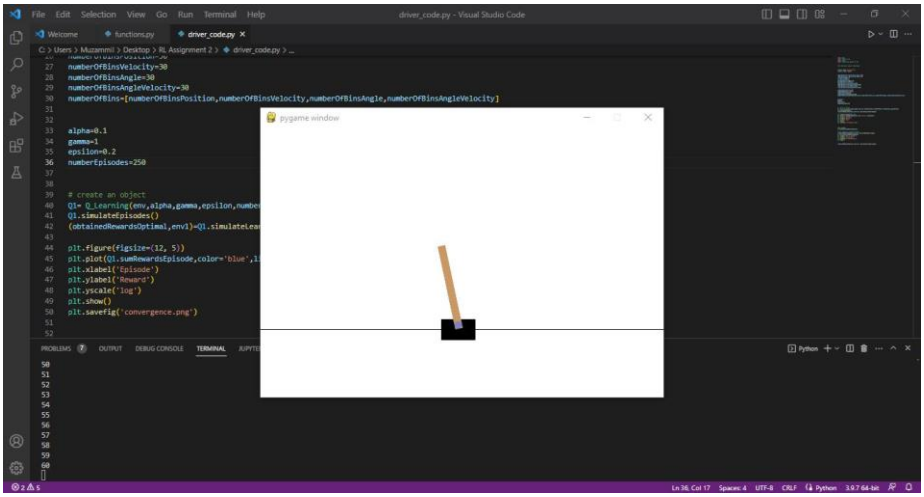
In this assignment, we are using these two learning algorithms and comparing the performance of both in terms of rewards, computation etc.

Q – Learning

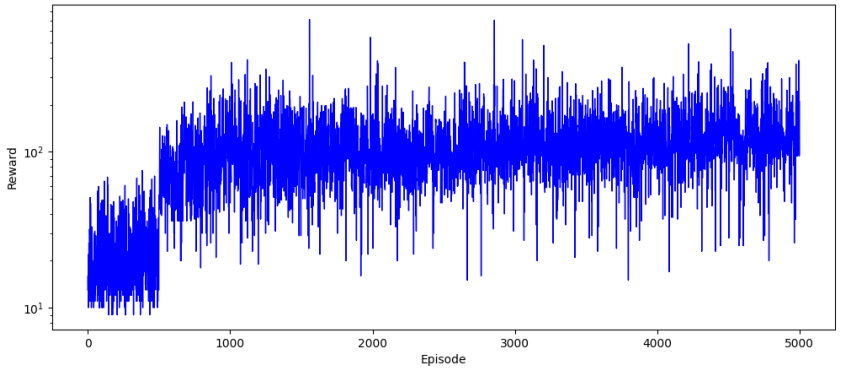
We are implementing the Q-learning algorithm using two different approaches.

1- Q Learning Approach 1

In the first approach, we are initializing the environment with the initial state. Then for every timestep of single episode until the terminal state is reached selecting an action A using epsilon greedy algorithm and then applying that action on the environment and observing the reward and the next state. If the next state is not the terminal state, then update the action value functions then repeat the process until the terminal state is reached.

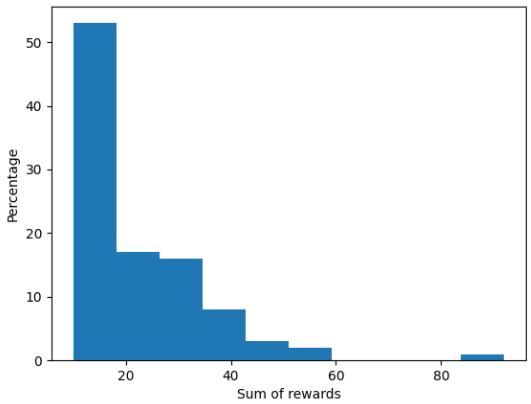


Here is a snapshot of pole balancing after running the code of Q-learning.



We can observe from the graph that the sum of rewards is increasing with the number of episodes. We have run the algorithm on 5000 Episodes and epsilon 0.2.

Here with this implementation, we are getting a reward of more than 300.



2- Q Learning Approach 2:

In the second implementation of Q learning, we are creating environment and setting limits of environments like max and min angle of rotation, velocity of the movement of the cart (-alpha, alpha), angular rotation (-24,24) and the position of the cart (-4.8,4.8). Once the environment is started our system will start with the random initial state. After that we are setting a hyperparameters such as number of episodes, learning rate etc. and running the training on the specified number of episodes and at every episode the Q value corresponding to the agents pole position and pole velocity was updated.

```
#Hyperparamters
EPISODES = 10000
DISCOUNT = 0.95
EPISODE_DISPLAY = 5

0
LEARNING_RATE = 0.25
EPSILON = 0.2
```

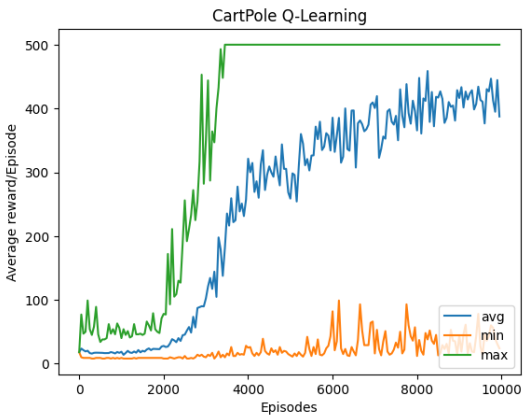
SARSA Algorithm

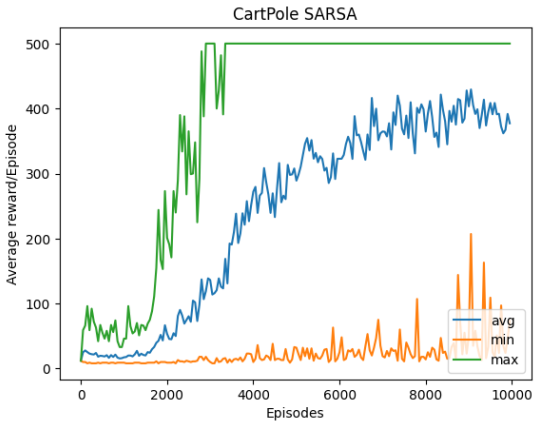
SARSA is an on-policy temporal difference learning algorithm, which is a model free learning and performs learning by bootstrapping from the current function. In SARSA, initialization of hyperparameters is like Q learning but the main difference comes in the selection of the action and updating of the Q-table. In SARSA, at each when the agent is in a particular state, an action is taken and the corresponding reward and state that is reached (next state) are recorded.

```
#Hyperparamters
EPISODES = 10000
DISCOUNT = 0.95
EPISODE_DISPLAY = 50
LEARNING_RATE = 0.25
EPSILON = 0.2
```

Parameters are also same for both SARSA and Q learning models, so that we can differentiate between the algorithms.

RESULTS: Q-LEARNING VS SARSA





We can observe from the results of both models that Q learning is getting higher maximum reward per episode as compared to the SARSA algorithm.

The maximum reward is achieved at around 2500 episodes by the agent for Q learning SARSA.

The maximum reward is converged at 500 for around 2500 episodes training and average reward is 350 plus for both algorithms after 5000 episodes.

If we run for around 15000 episodes then maybe average reward for Q learning will converge with the maximum reward as we can see the trend of average reward, but SARSA one is declining as one point.

In the end, we conclude that Q Learning is working better than the SARSA Algorithm in our experiment based on the following results.