Artificial Intelligence

Programming Assignment - 4

Kaustav Vats (2016048)

Solving 2D Maze using **Q-Learning** in Reinforcement Learning with given Maze Environment. Maze Size **5x5.**

$$\underbrace{NewQ(s,a)}_{\text{New Q value for that state and that action}} = \underbrace{Q(s,a)}_{\text{Current Q value}} + \underset{\text{Learning Rate}}{\alpha}[\underbrace{R(s,a)}_{\text{Reward for taking that action at that state}} + \underset{\text{Discount rate}}{\gamma}\, \underbrace{max\, Q'(s',a')}_{\text{Maximum expected future reward given the new s' and all possible actions at that new state}} - Q(s,a)]$$

Above is the QTable updation formula. Parameters explained:-

- NewQ(s, a): Denotes the new Q value for that state s after taking action a
- Q(s, a): This is the old Q value of that state, it's actually representation of quality of the action taken from a particular state. Larger Q value means better chances of getting higher rewards.
- α: Learning Rate for the Algorithm. For higher alpha value agent consider reward and the future reward that can lead to convergence and it rejects former Q values.
- R(s, a): Reward we get after taking action a when on the particular state s.
- γ: Discount Rate denotes how much importance need to be given to the future reward. If Discount Factor value closer to 1, then captures long term reward. If 0 then only consider the immediate reward.

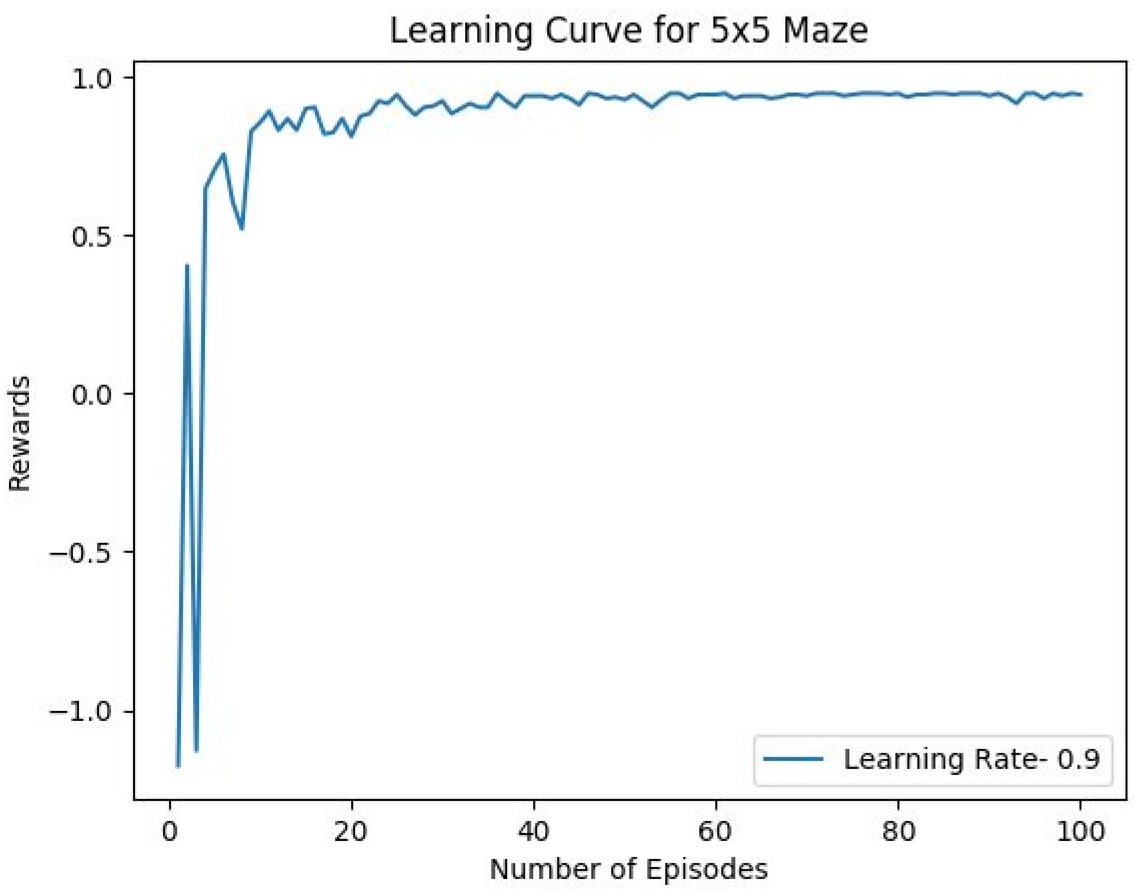
How I implemented Q-Learning-

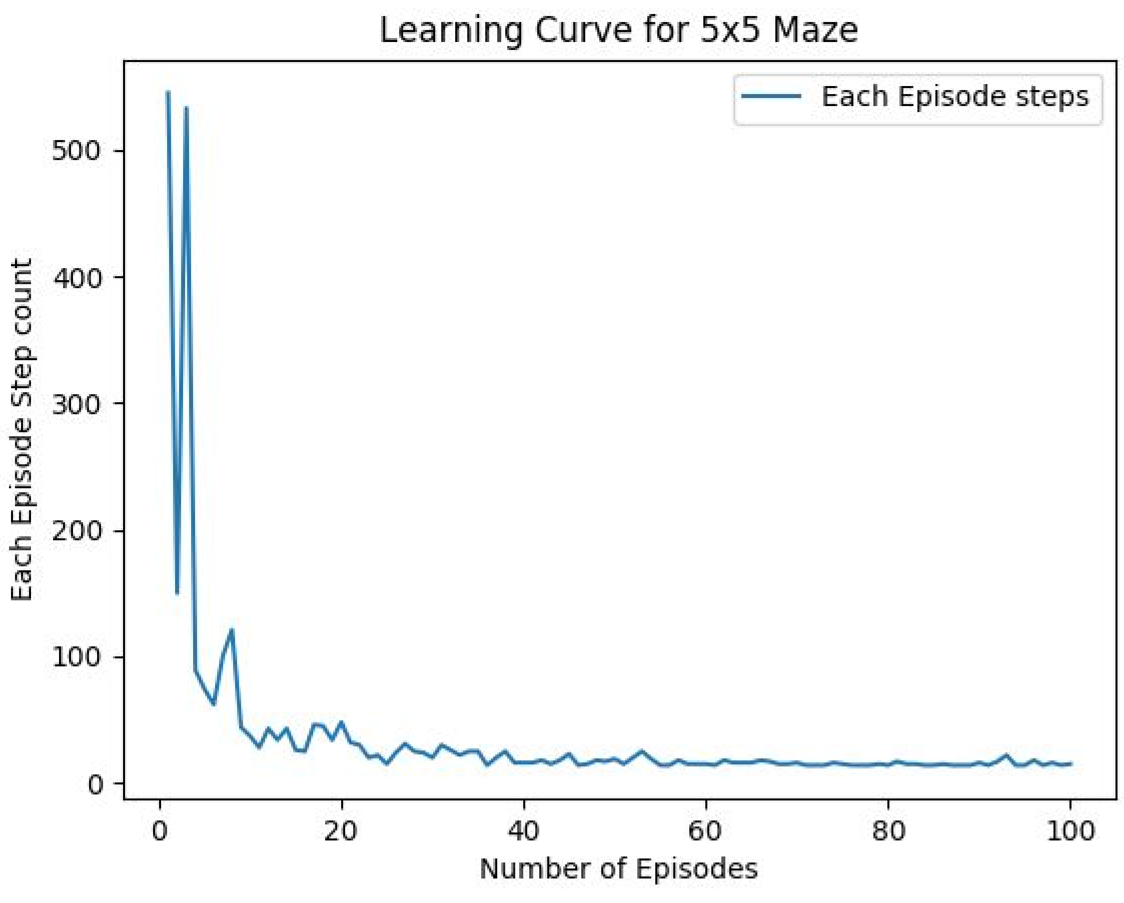
- Initialized Q-Table with zero value.
- If exploration has more probability then select any random action, else select action with maximum Q Value for a particular state(Exploration probability decreases as the learning proceeds).
- Go to the next state after choosing action from the above state.
- Update Q table
- Repeat above steps unitil Agent learn and always take the best possible path.

Exploration Rate is reduced by the 30% after each episode. Initially the Exploration rate is high to Explore the Environment. This is to provide Exploration of the environment to the agent initially and as Learning proceeds Agent Exploration Rate decreases and reach a minimum value of 0.1. When Exploration Rate is closer to Zero then Agent will mostly take action based on the Q value for each Action.

Learning Curve

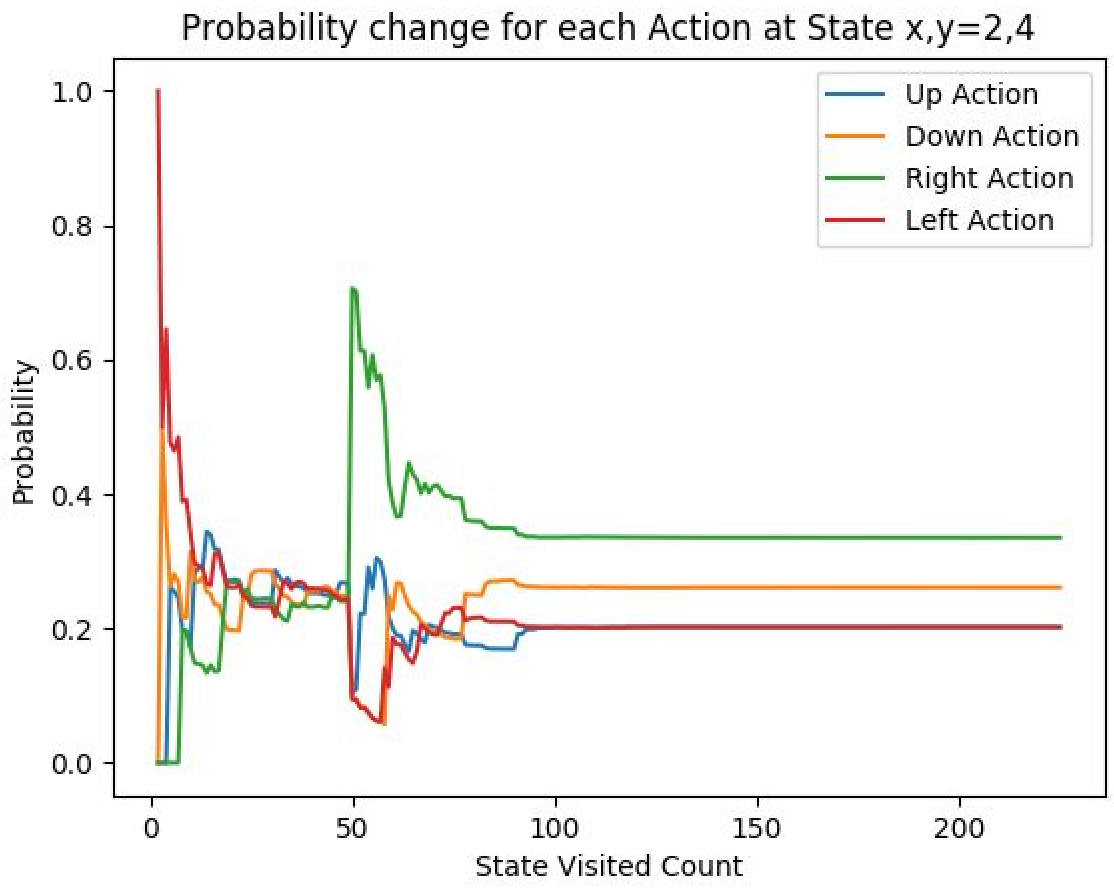
- **Figure 1**, shows the learning curve based on the total reward received each Episode. As the Agent proceeds learning, Total Reward keep increasing. Maximum Reward is around 0.94
- **Figure 2**, shows the learning curve based on the Total steps taken to reach goal as the Agent learning proceeds, Total steps keep decreasing. Best step count is 14


Learning Curve for 5x5 Maze
Rewards
1.0
0.5
0.0
−0.5
−1.0
Learning Rate- 0.9
0
20
40
60
80
100
Number of Episodes


Learning Curve for 5x5 Maze
Each Episode Step count
500
400
300
200
100
0
Each Episode steps
0
20
40
60
80
100
Number of Episodes

Probability occurrence of each Action

- **Figure 3**, shows the probability of choosing an action. Probability of an action is calculated as (Q Value of the that Action)/(Sum of all actions Q values on a particular state). As the learning proceeds, Probability of an action that leads to best path, increases based on its Q value. If there's no change in the probability of an action, then Probability of each action remains same.



Impact of various Hyperparameters of Q-Learning

Learning Rate	Discount Factor	Exploration Rate	Total step Count
0.7	0.8	0.4	30.85
0.3	0.8	0.4	31.70
0.7	0.4	0.8	31.61
0.3	0.4	0.8	36.80
0.7	0.8	0.8	30.50

For low exploration rate number of steps taken by the agent for 100 episodes are less as compared to other factors.
For high Discount Factor, Agent give more weightage to new Q Value more and discard former value that were calculated by the agent. When Exploration Rate is high, Agent explore the environment more.

Best result is marked with Green, Learning rate, Exploration rate and Discount Factor is also high which helps the agent to explore the environment and give less weightage to former values.

Result for Agent trained using Q-Learning

Total Episodes: 100
Average steps per episode: 14.0
Average penalties per episode: 13.0
Average Reward per Episode: 0.9479999999999998

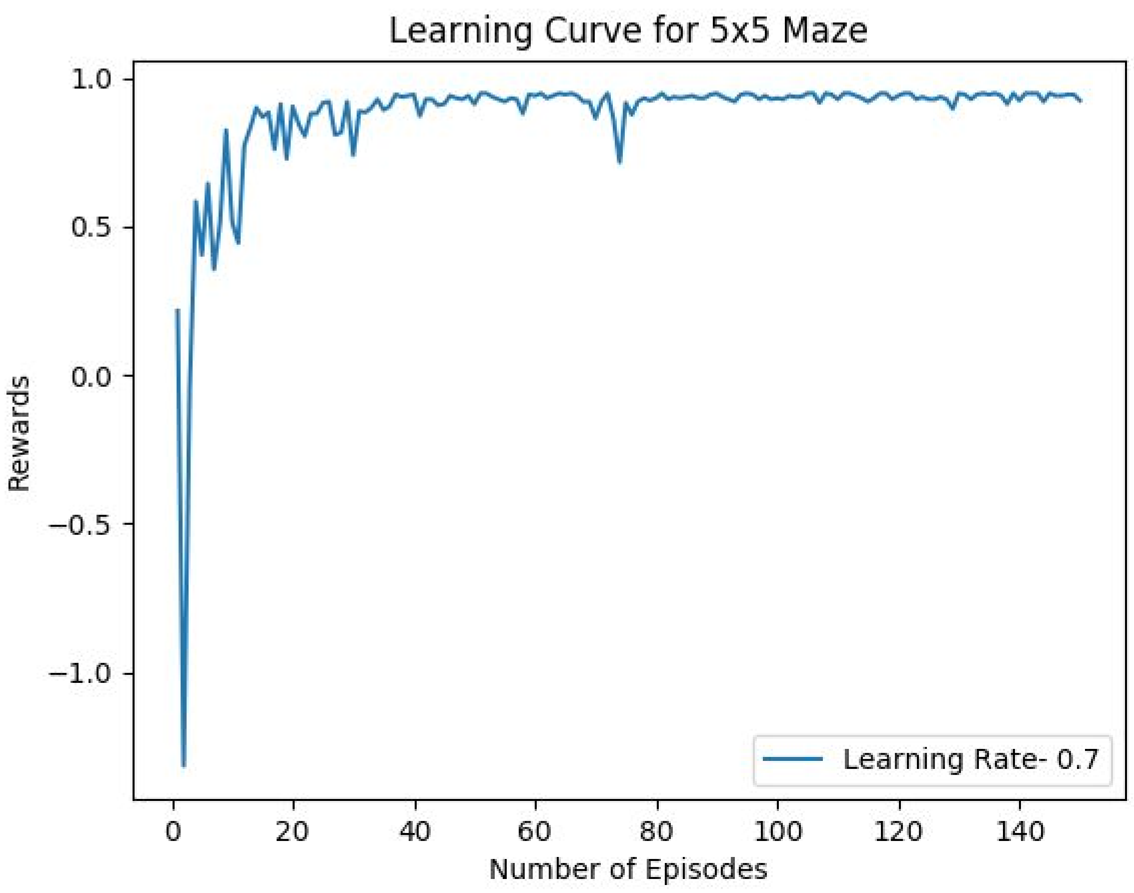
Here we can see that Average Reward per episode is very close to 1. Agent is taking 14 steps to complete the maze. Best step count to complete the maze is 14, and my trained agent is able to achieve this.
Average Penalty is number of time the agent has received negative reward for 100 episodes. *(A Negative reward doesn't always mean the agent has taken the wrong step, Negative rewards can also be used for any RL problem)*
Problem is that Agent is receiving Negative reward even after taking best step in a state. Reward should be given based quality of the action. I observed that the rewards are not defined properly in the environment. Documentation of the environment is also not provided. To solve this problem Agent should receive some positive reward for all best action and it should be higher than the other actions.

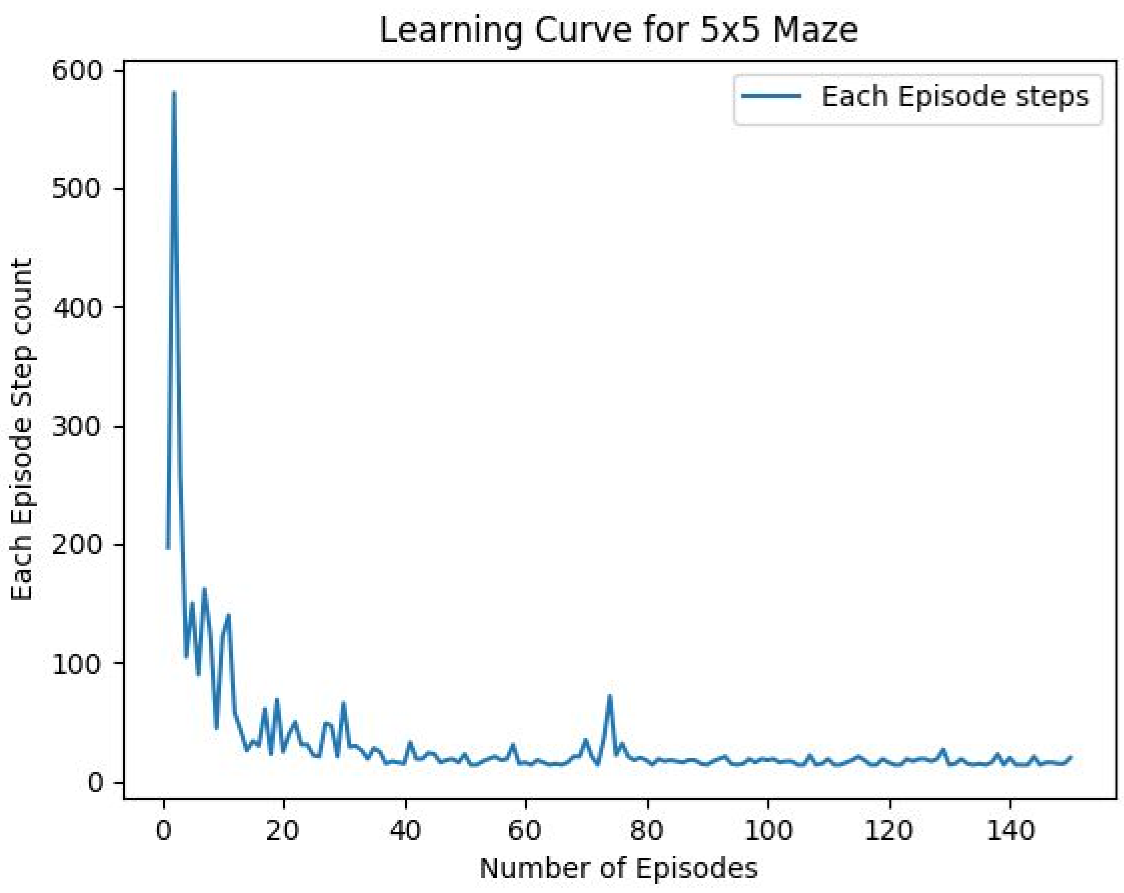
Bonus
predict = QTable[CurrentState + (Action,)]
target = RewardAction + DiscountFactor*QTable[NewState + (Action2,)]
QTable[CurrentState + (Action,)] = QTable[CurrentState + (Action,)] + LearningRate * (target - predict)
CurrentState = NewState
Action = Action2

Above how action is updated with new action, In SARSA we don't calculate action based on the maximum Q value. We update action with new action that was taken to update Q value with new Q value.

Learning Curve

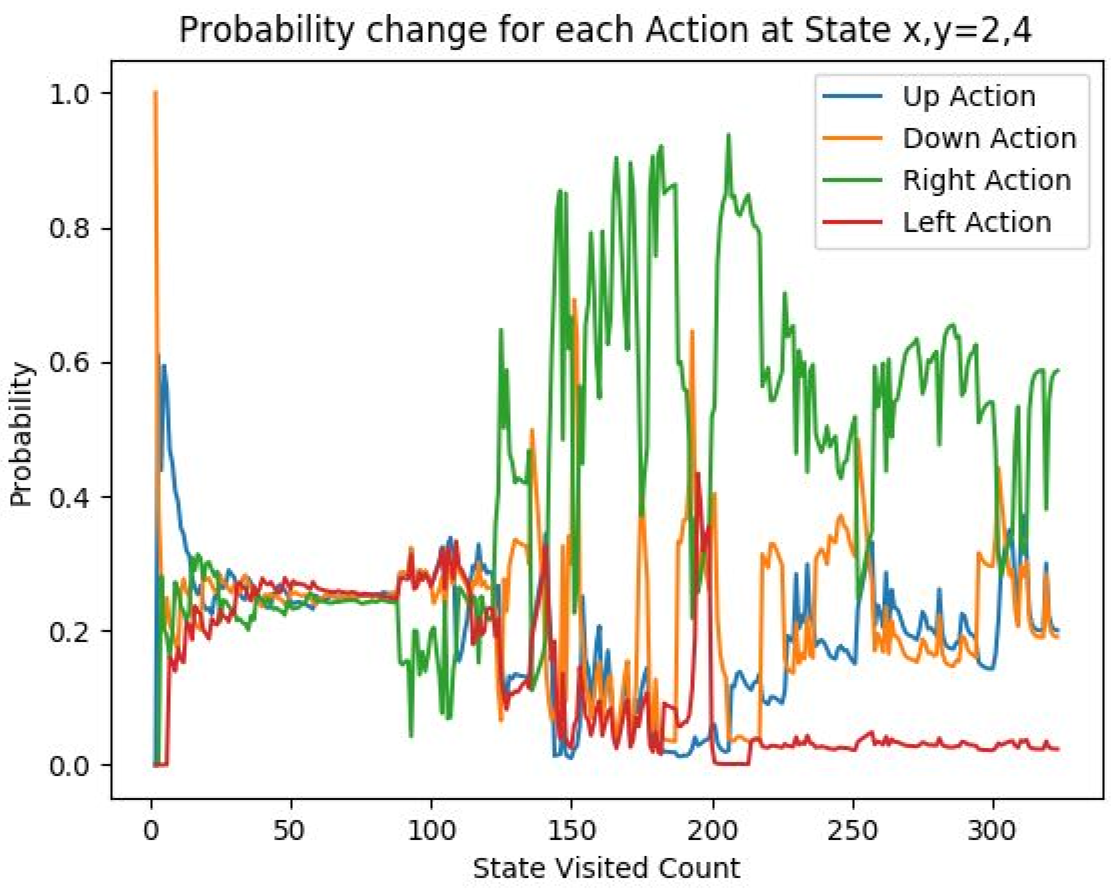
- **Figure 1**, shows the learning curve based on the total reward received each Episode. As the Agent proceeds learning, Total Reward keep increasing. Maximum Reward is around 0.94
- **Figure 2**, shows the learning curve based on the Total steps taken to reach goal as the Agent learning proceeds, Total steps keep decreasing. Best step count is 14


Learning Curve for 5x5 Maze
Rewards
Number of Episodes
Learning Rate- 0.7
1.0
0.5
0.0
−0.5
−1.0
0
20
40
60
80
100
120
140


Learning Curve for 5x5 Maze
Each Episode Step count
Number of Episodes
Each Episode steps
600
500
400
300
200
100
0
0
20
40
60
80
100
120
140

Probability occurrence of each Action

- **Figure 3**, shows the probability of choosing an action. Probability of an action is calculated as (Q Value of the that Action)/(Sum of all actions Q values on a particular state). As the learning proceeds, Probability of an action that leads to best path, increases based on its Q value. If there's no change in the probability of an action, then Probability of each action remains same.



References-

1. https://medium.freecodecamp.org/how-to-build-an-ai-game-bot-using-openai-gym-and-universe-f2eb9bfbb40a
2. https://medium.freecodecamp.org/diving-deeper-into-reinforcement-learning-with-q-learning-c18d0db58efe
3. https://www.learndatasci.com/tutorials/reinforcement-q-learning-scratch-python-openai-gym/
4. https://medium.com/swlh/introduction-to-reinforcement-learning-coding-sarsa-part-4-2d64d6e37617
5. https://studywolf.wordpress.com/2013/07/01/reinforcement-learning-sarsa-vs-q-learning/