CS6700 - Reinforcement Learning

Assignment - 1

Bagavath Shri Ram S B - ME20B042

K Kannikeshwwar - ME20B094

28th February 2024

Contents

1 Environment Description

The Grid world is defined with two start locations and 3 Goal locations

• Goal state: The goal is to reach one of these states. There are 3 goal states in total.

• Obstructed state: These are walls that prevent entry to the respective cells. Transition to these states will not result in any change.

• Bad state: Entry into these states will incur a higher penalty than a normal state.

• Restart state: Entry into these states will incur a very high penalty and will cause agent to teleport to the start state without the episode ending. Once the restart state is reached, no matter what action is chosen, it goes to the start state at the next step.

• Normal state: None of the above. Entry into these states will incur a small penalty.

Rewards: -1 for normal states, -100 for restart states, -6 for bad states, +10 for goal states.

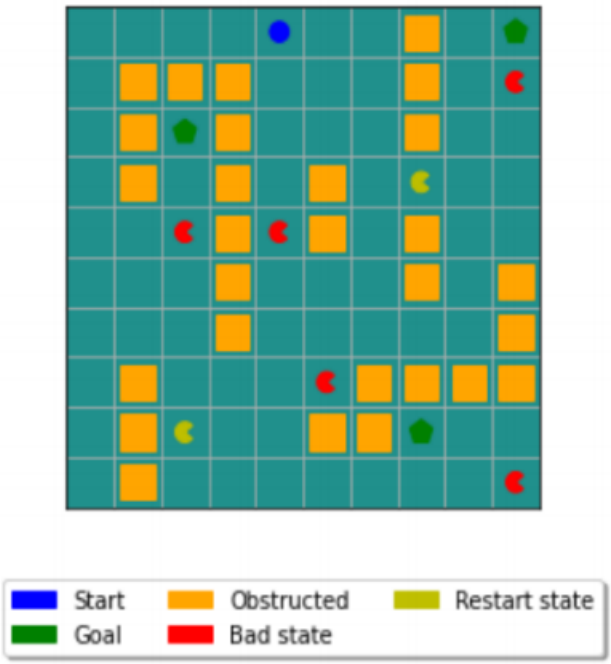


Figure 2: An example grid with start point at (0, 4)

We also define a transition probability $p \in [0, 1]$ and a bias $b \in [0, 1]$, and the total transition probability for the agent is given by:

$$P_{\text{west}} = (1 - p) \cdot b$$

$$P_{\text{east}} = (1 - p) \cdot (1 - b)$$

where P_{west} and P_{east} are the probabilities that the agent transitions to the West and East of the given action respectively where the given action is taken as north. Bias, b, is set at 0.5, and we will vary the transition probability p among $\{1, 0.7\}$.

Additionally, we also define a probability $P_{\text{wind}} = 0.4$ that determines whether the agent moves one additional cell to the right when wind is set to True.

2 Tasks

- We will be implementing SARSA and Q-Learning algorithms.
- For each algorithm, we run the experiment either starting at $(0, 4)$ or $(3, 6)$.
- We will also have three variants of stochasticity for each start state:
    1. Wind = False, $p = 1.0$ (deterministic)
    2. Wind = False, $p = 0.7$ (stochastic)
    3. Wind = True, $p = 1.0$ (deterministic)
- In total we will be conducting **12 different experiments**.
- For each experiment, the best set of hyperparameters $\{\tau, \epsilon, \alpha, \gamma\}$ are determined using multiple grid searches on different ranges of each value in a greedy manner.
- Finally, after determining the best set of hyperparameters, the agent is trained on 10000 episodes over 5 runs and 4 sets of plots for each experiment:
    1. Reward curves (RC) and number of steps to goal against episodes
    2. Heatmap of grid with state visit counts (SVC)
    3. Heatmap of grid with Q values and optimal actions for the best policy.

3 SARSA and Q-Learning Algorithms

3.1 SARSA

SARSA is an on-policy temporal difference (TD) control method. It estimates the action-value function, denoted as $Q_\pi(s, a)$, for the current policy π and for all state-action pairs (s, a).

The SARSA update rule is:

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

This update is performed after every non-terminal state transition $(S_t, A_t, R_{t+1}, S_{t+1})$. When a terminal state S_{t+1} is reached, $Q(S_{t+1}, A_{t+1})$ is set to zero.

3.2 Q-Learning

Q-Learning is an off-policy TD control method. It estimates the optimal action-value function, denoted as $q^*(s, a)$, which is independent of the current policy π.

The Q-Learning update rule is:

$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha[R_{t+1} + \gamma \max_a Q(S_{t+1}, a) - Q(S_t, A_t)] \tag{2}$$

This update is also performed after every non-terminal state transition. When a terminal state S_{t+1} is reached, $Q(S_{t+1}, A_{t+1})$ is set to zero.

4 Hyperparameter Tuning

The hyperparameters for each experiment are:

1. Learning Rate (α)
2. Discount Factor (γ)
3. Temperature Parameter in Softmax (τ)
4. Epsilon in ϵ-greedy (ϵ)

Based on multiple tuning runs and intuition, we choose a certain range (or grid) of values for each parameter. We then find the best hyperparameter combination greedily using grid search. The search is performed over the following grids:

1. $\alpha \in \{0.01, 0.05, 0.1, 0.2\}$
2. $\gamma \in \{0.8, 0.9, 1.0\}$
3. $\tau \in \{1.0, 2.0, 4.0, 10\}$
4. $\epsilon \in \{0.01, 0.05, 0.1\}$

Grid search provides the best set of hyperparameters for each experiment, resulting in the optimal mean episodic rewards. One such observation is presented below. The marked one is the set of optimal parameters for softmax and epsilon greedy.

	Epsilon	Alpha	Gamma	Reward
1	0.01	0.01	0.8	-28.516
2	0.01	0.01	0.9	-24.9694
3	0.01	0.01	1.0	-23.278
4	0.01	0.05	0.8	-11.2938
5	0.01	0.05	0.9	-10.219
6	0.01	0.05	1.0	-9.8304
7	0.01	0.1	0.8	-8.968
8	0.01	0.1	0.9	-8.4098
9	0.01	0.1	1.0	-8.154
10	0.01	0.2	0.8	-7.8524
11	0.01	0.2	0.9	-7.5528
12	0.01	0.2	1.0	-7.3432
13	0.05	0.01	0.8	-32.4196
14	0.05	0.01	0.9	-27.9358
15	0.05	0.01	1.0	-25.373
16	0.05	0.05	0.8	-12.6002
17	0.05	0.05	0.9	-11.4596
18	0.05	0.05	1.0	-10.9058
19	0.05	0.1	0.8	-9.8402
20	0.05	0.1	0.9	-9.3408
21	0.05	0.1	1.0	-9.1108
22	0.05	0.2	0.8	-8.9118
23	0.05	0.2	0.9	-8.3096
24	0.05	0.2	1.0	-8.2672
25	0.1	0.01	0.8	-34.0824
26	0.1	0.01	0.9	-30.2638
27	0.1	0.01	1.0	-27.3464
28	0.1	0.05	0.8	-13.9894
29	0.1	0.05	0.9	-12.5898
30	0.1	0.05	1.0	-12.12
31	0.1	0.1	0.8	-11.082
32	0.1	0.1	0.9	-10.576
33	0.1	0.1	1.0	-10.0794
34	0.1	0.2	0.8	-11.509
35	0.1	0.2	0.9	-9.7736
36	0.1	0.2	1.0	-9.4552

Figure 1: Grid-Search on ϵ, α, γ

	Tau	Alpha	Gamma	Reward
1	1.0	0.01	0.8	-89.4014
2	1.0	0.01	0.9	-57.1042
3	1.0	0.01	1.0	-39.6796
4	1.0	0.05	0.8	-72.4492
5	1.0	0.05	0.9	-24.5676
6	1.0	0.05	1.0	-13.7856
7	1.0	0.1	0.8	-71.4318
8	1.0	0.1	0.9	-21.9352
9	1.0	0.1	1.0	-11.5958
10	1.0	0.2	0.8	-71.1458
11	1.0	0.2	0.9	-20.9806
12	1.0	0.2	1.0	-12.456
13	2.0	0.01	0.8	-97.4102
14	2.0	0.01	0.9	-82.8672
15	2.0	0.01	1.0	-52.7338
16	2.0	0.05	0.8	-88.0322
17	2.0	0.05	0.9	-57.6048
18	2.0	0.05	1.0	-23.4324
19	2.0	0.1	0.8	-87.2786
20	2.0	0.1	0.9	-55.4866
21	2.0	0.1	1.0	-20.0442
22	2.0	0.2	0.8	-85.4488
23	2.0	0.2	0.9	-53.4756
24	2.0	0.2	1.0	-18.035
25	4.0	0.01	0.8	-102.441
26	4.0	0.01	0.9	-98.9826
27	4.0	0.01	1.0	-79.0988
28	4.0	0.05	0.8	-96.2376
29	4.0	0.05	0.9	-87.7098
30	4.0	0.05	1.0	-39.2036
31	4.0	0.1	0.8	-94.8196
32	4.0	0.1	0.9	-85.1716
33	4.0	0.1	1.0	-34.1508
34	4.0	0.2	0.8	-94.484
35	4.0	0.2	0.9	-84.7502
36	4.0	0.2	1.0	-31.1678
37	10.0	0.01	0.8	-110.1598
38	10.0	0.01	0.9	-109.8952
39	10.0	0.01	1.0	-105.9074
40	10.0	0.05	0.8	-104.1528
41	10.0	0.05	0.9	-102.0328
42	10.0	0.05	1.0	-68.6506
43	10.0	0.1	0.8	-103.022
44	10.0	0.1	0.9	-100.0306
45	10.0	0.1	1.0	-60.3094
46	10.0	0.2	0.8	-102.5716
47	10.0	0.2	0.9	-99.8626
48	10.0	0.2	1.0	-56.0932

Figure 2: Grid-Search on τ, α, γ

5 Experiments

The experiments are ordered based on start states, wind, p, and algorithm in that order. Each experiment will show both softmax and ϵ-greedy policy runs and the optimal hyperparameters obtained for each from tuning. The ordering of the experiment (indices) can be a bit inconsistent, but it is in the same order as we used in our code.

The dimensions of the grid are 10×10. The coordinates for "special" cells are given as below:

Obstruction cells are:

```
[[0, 7],[1, 1],[1, 2],[1, 3],
 [1, 7],[2, 1],[2, 3],[2, 7],
 [3, 1],[3, 3],[3, 5],[4, 3],
 [4, 5],[4, 7],[5, 3],[5, 7],
 [5, 9],[6, 3],[6, 9],[7, 1],
 [7, 6],[7, 7],[7, 8],[7, 9],
 [8, 1],[8, 5],[8, 6],[9, 1]]
```

Bad cells are:

```
[[1, 9],[4, 2],[4, 4],[7, 5],[9, 9]]
```

Restart state cells are:

```
[[3, 7], [8, 2]]
```

Goal state cells are:

```
[[0, 9],[2, 2],[8, 7]]
```

5.1 Start state = $(0, 4)$, wind = False, $p = 1.0$

Experiment 1 - SARSA

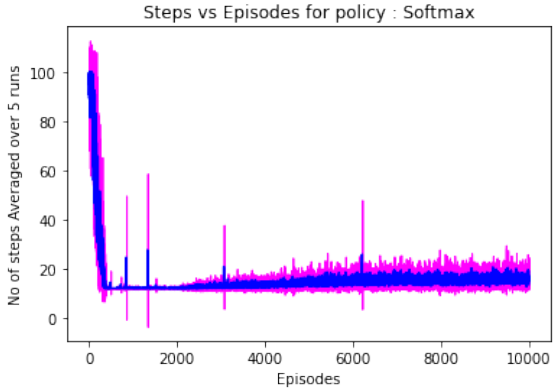
- Algorithm: SARSA
- Wind: False
- $p = 1.0$ (deterministic step)

Choosing the best policy between softmax and ϵ-greedy.

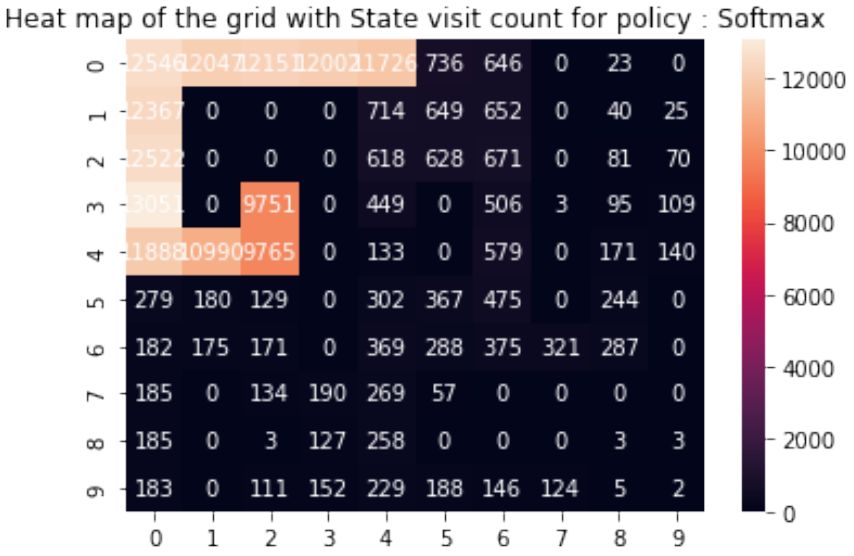
softmax

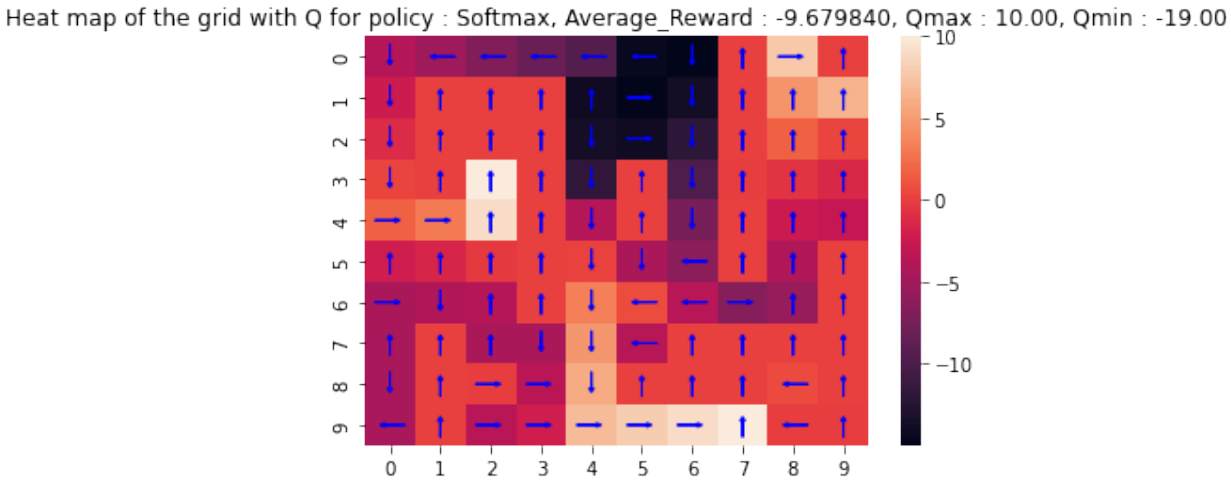
Optimal Hyperparameters: $\{\tau : 1.0, \alpha : 0.1, \gamma : 1.0\}$

(a) Rewards vs Episodes



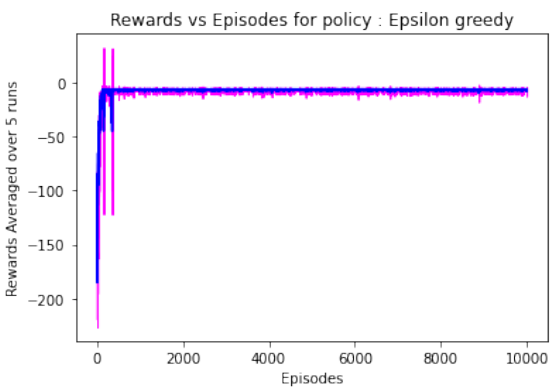
(b) Number of steps vs Episodes



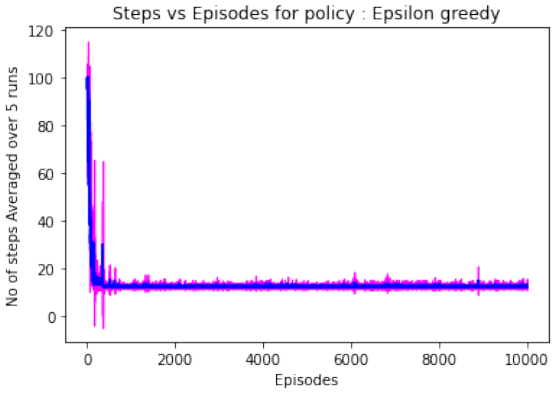


ϵ-greedy

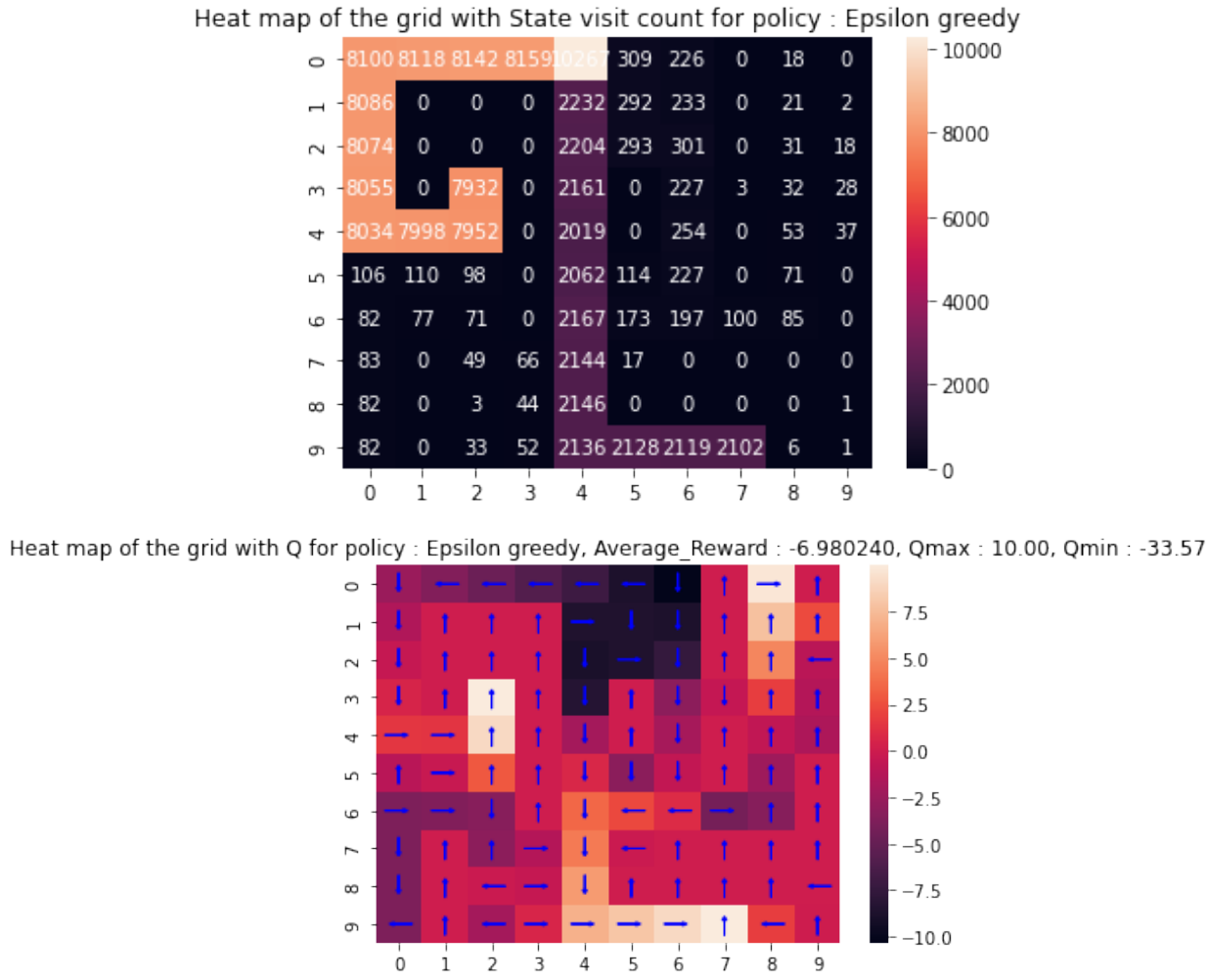
Optimal Hyperparameters: $\{\epsilon : 0.01, \alpha : 0.2, \gamma : 1.0\}$



(a) Rewards vs Episodes



(b) Number of steps vs Episodes



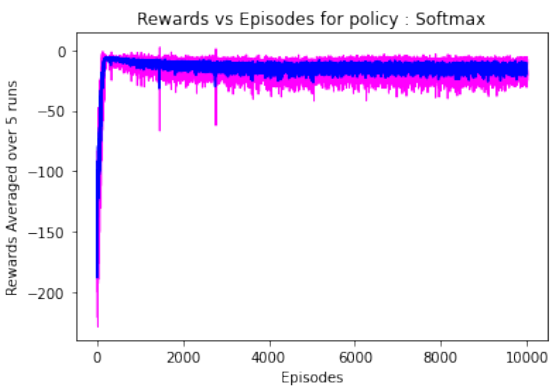
Experiment 7 - Q-Learning

- Algorithm: Q-Learning
- Wind: False
- $p = 1.0$ (deterministic step)

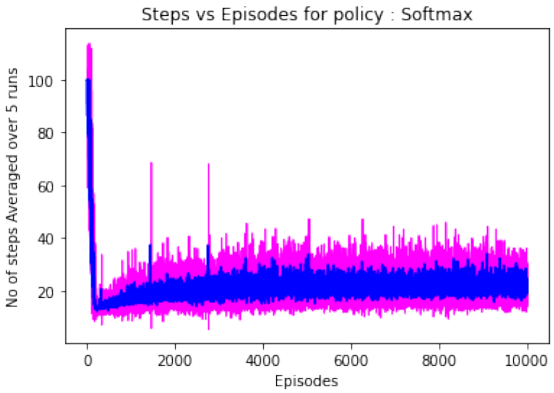
Choosing the best policy between softmax and ϵ-greedy.

softmax

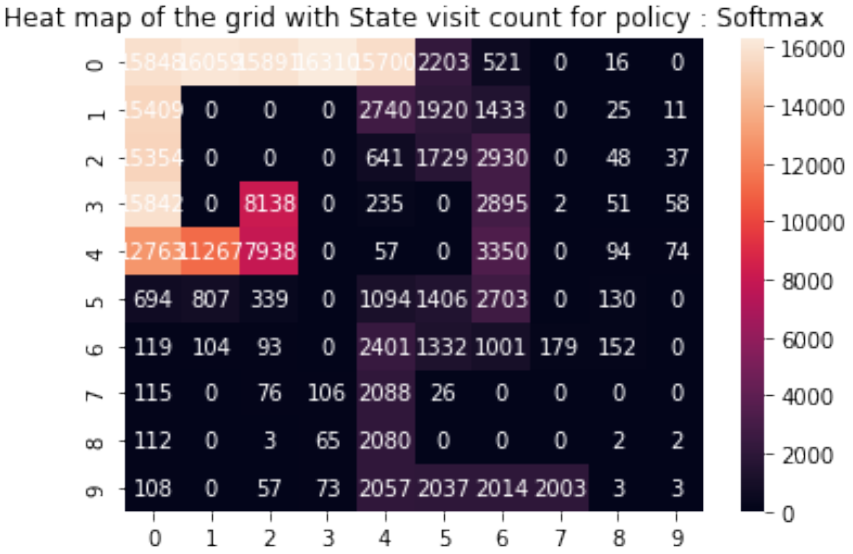
Optimal Hyperparameters: $\{\tau : 1.0, \alpha : 0.2, \gamma : 1.0\}$

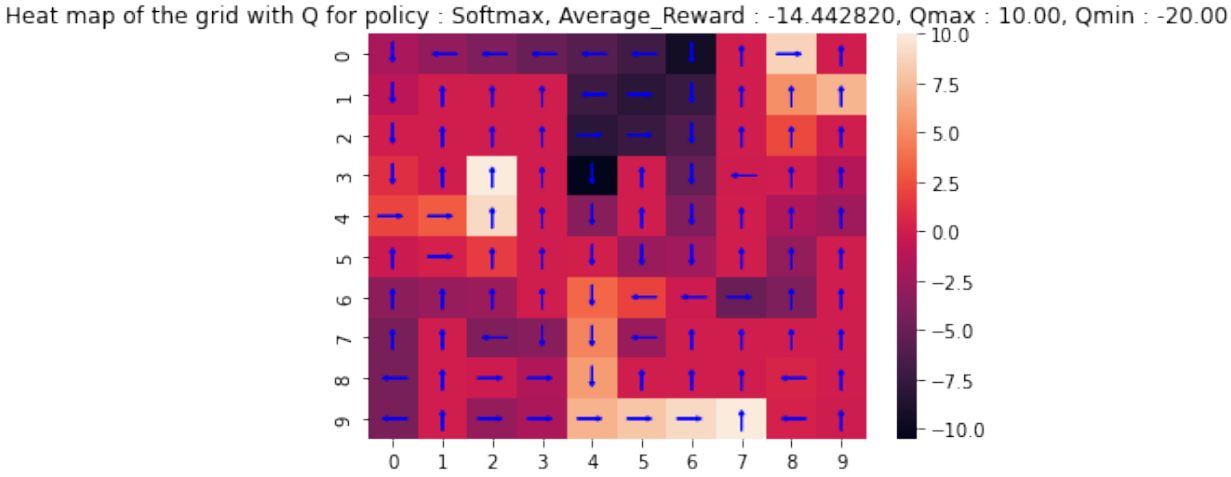


(a) Rewards vs Episodes



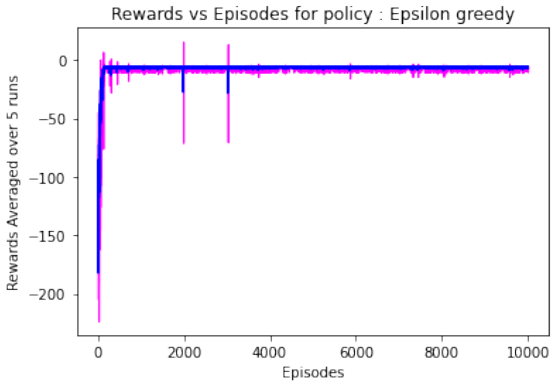
(b) Number of steps vs Episodes



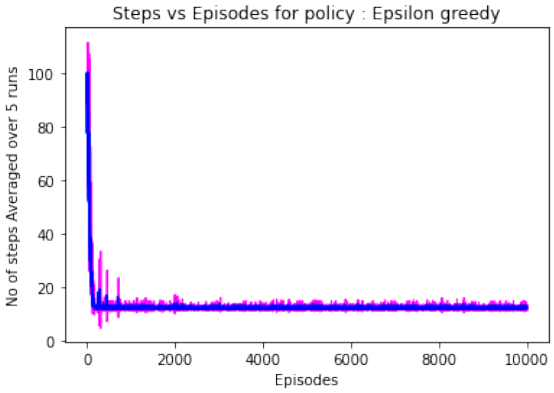


ϵ-greedy

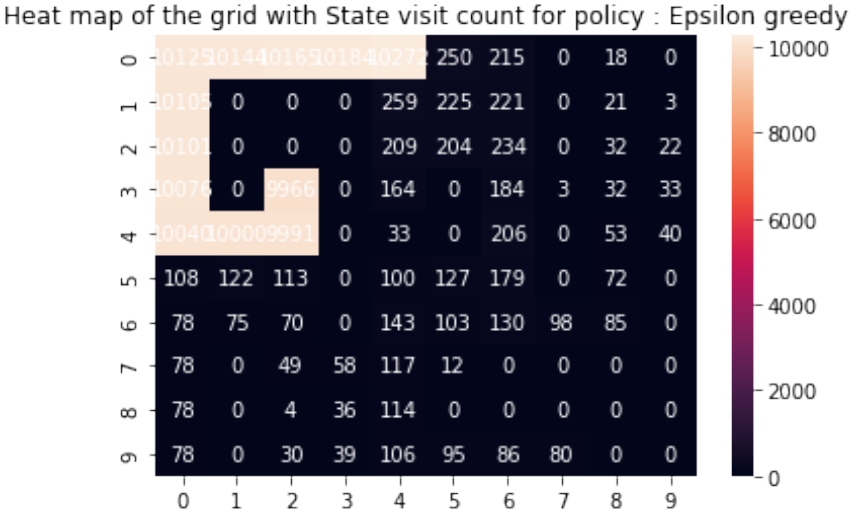
Optimal Hyperparameters: $\{\epsilon : 0.01, \alpha : 0.2, \gamma : 1.0\}$

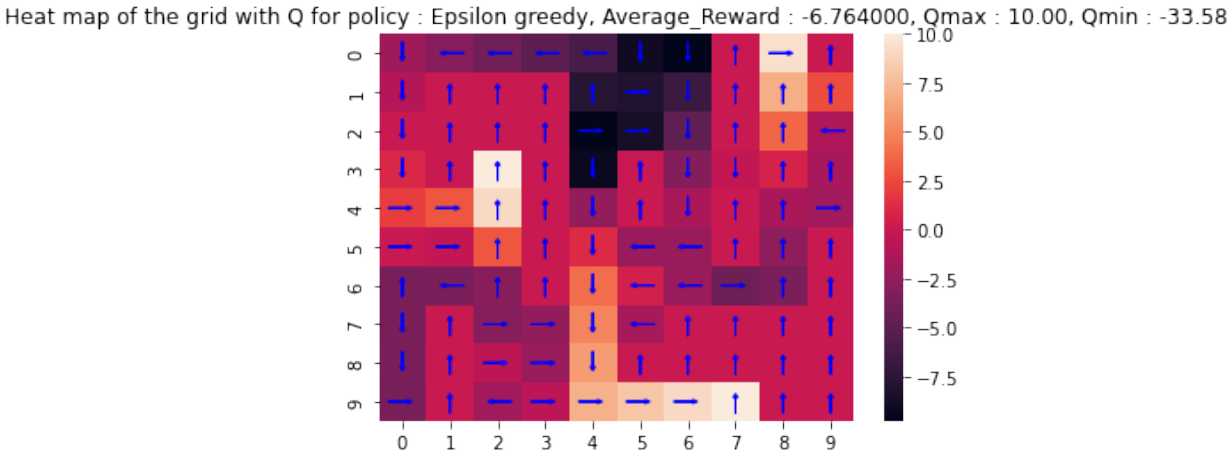


(a) Rewards vs Episodes



(b) Number of steps vs Episodes





Inferences for 5.1

Similarities

- Epsilon greedy gives higher average reward when compared to Softmax.
- Both the algorithms are giving more expected return towards goal (8, 7).

Differences

- In SARSA, Epsilon greedy is exploring two goal states more number of times when compared to softmax where one of the goal state is visited majority of the time.
- In QLEARNING, Softmax is exploring two goal states more number of times when compared to Epsilon greedy where one of the goal state is visited majority of the time.

Justification of path

- Both the policies learnt by SARSA and Q Learning are leading to the goal state (2, 2). SARSA gives a safer path whereas Q Learning gives an optimal path. Since travelling to all three goal states gives nearly the same reward, it doesn't matter which path it takes in the case of Q Learning.
- Along the path to (0, 9) there is one restart state and one bad state, and along the path to (8, 7) there are many bad states, whereas along the path to (2, 2) there is only one bad state, but in the deterministic case it doesn't matter which path it takes in the case of SARSA.

5.2 Start state = $(0, 4)$, wind = False, $p = 0.7$

Experiment 2 - SARSA

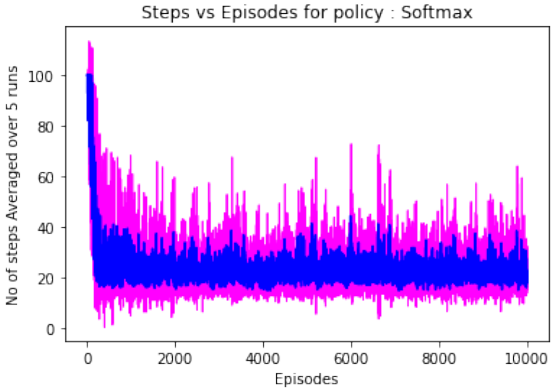
- Algorithm: SARSA
- Wind: False
- $p = 0.7$ (stochastic step)

Choosing the best policy between softmax and ϵ-greedy.

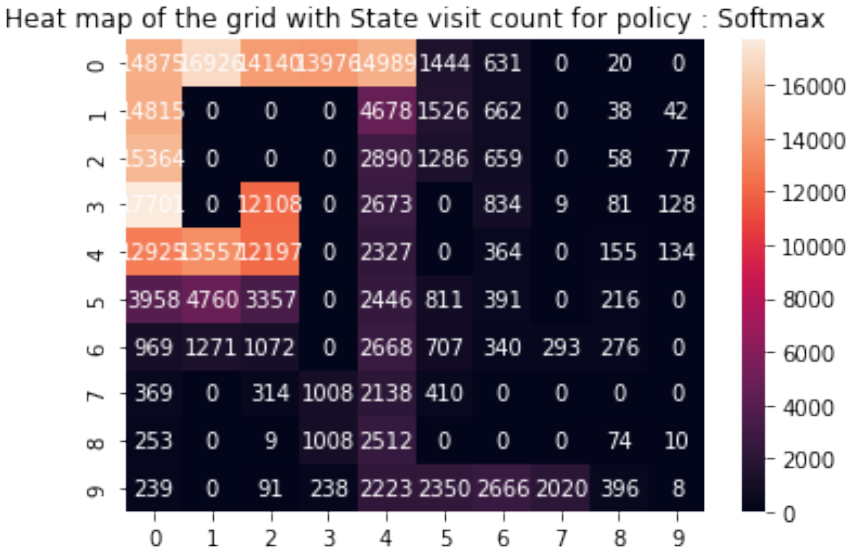
softmax

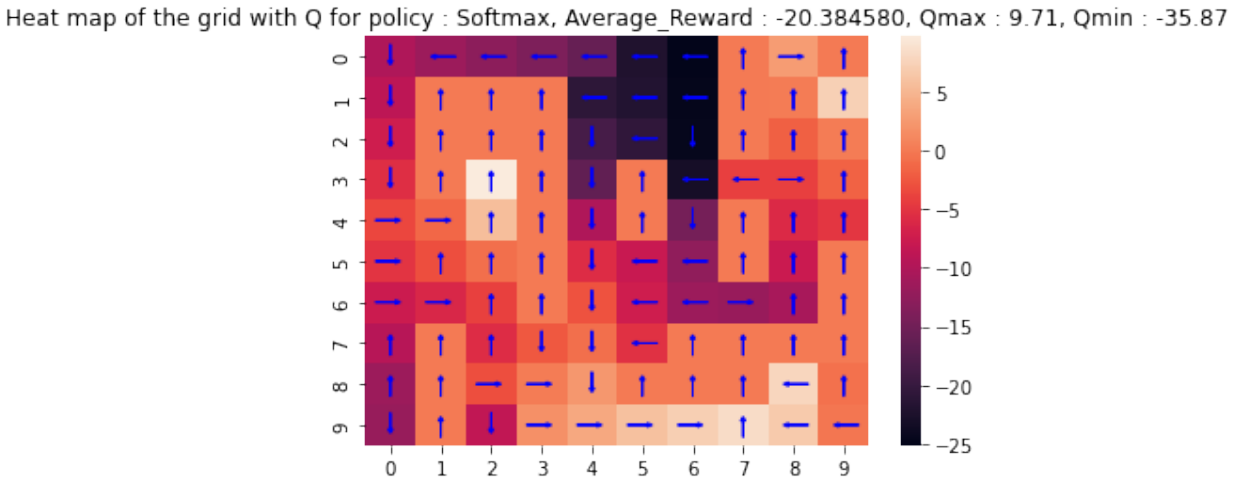
Optimal Hyperparameters: $\{\tau : 1.0, \alpha : 0.2, \gamma : 1.0\}$

(a) Rewards vs Episodes



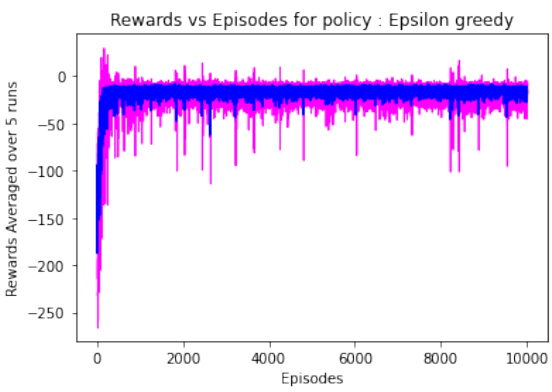
(b) Number of steps vs Episodes



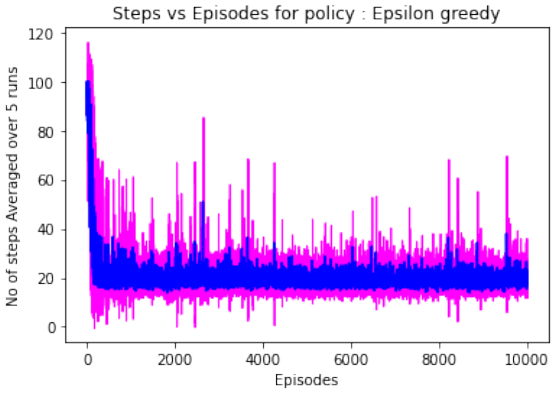


ϵ-greedy

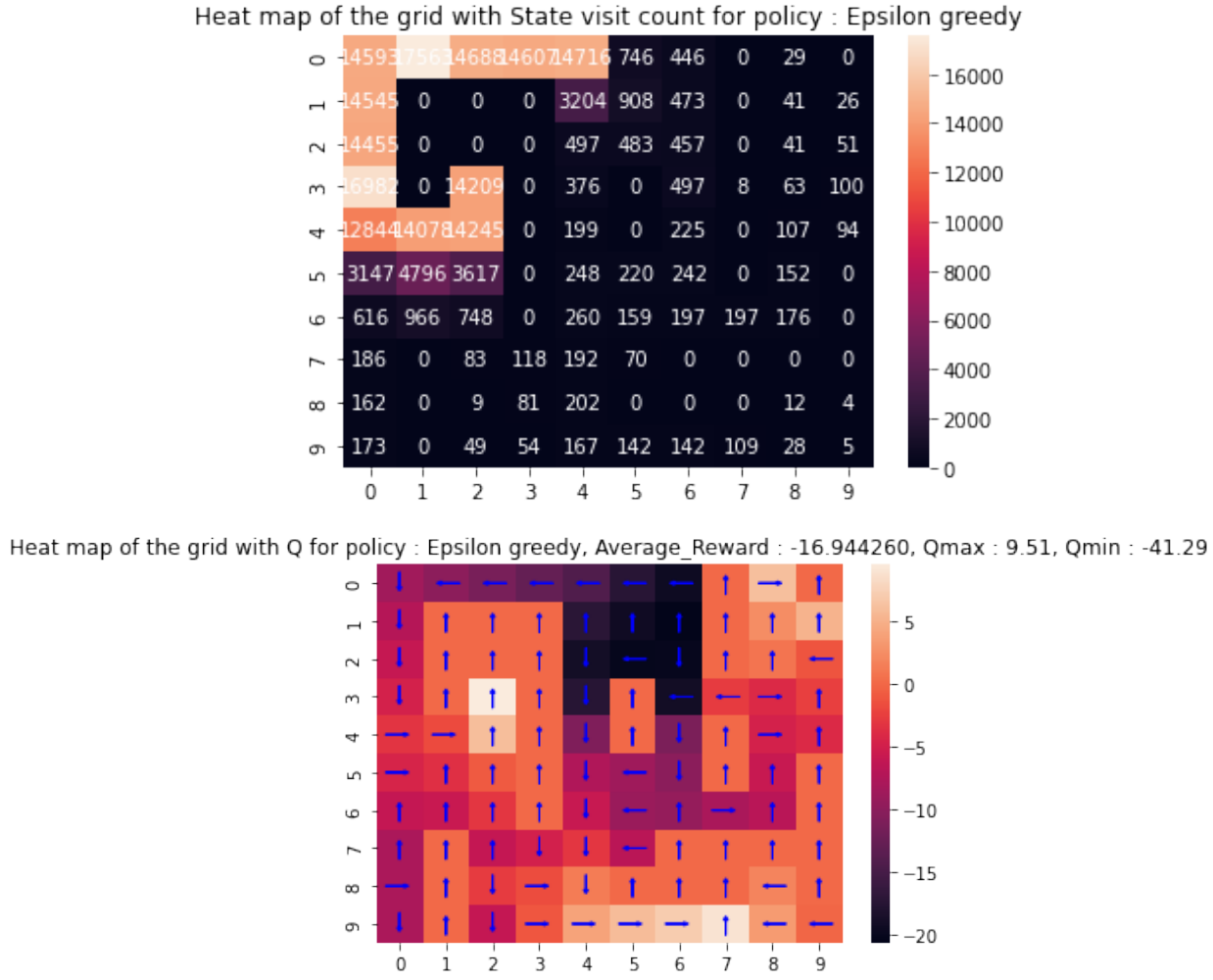
Optimal Hyperparameters: $\{\epsilon : 0.01, \alpha : 0.2, \gamma : 1.0\}$



(a) Rewards vs Episodes



(b) Number of steps vs Episodes



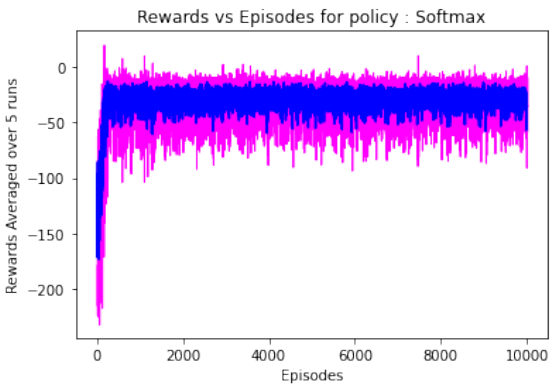
Experiment 8 - Q-Learning

- Algorithm: Q-Learning
- Wind: False
- $p = 0.7$ (stochastic step)

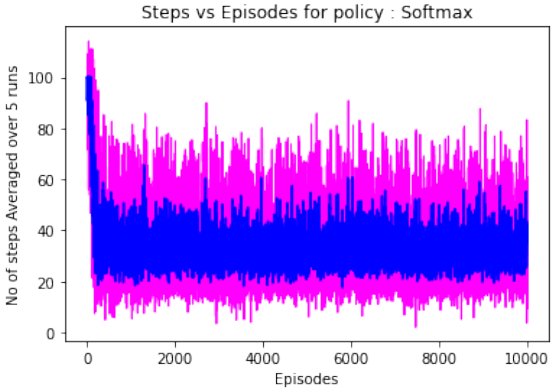
Choosing the best policy between softmax and ϵ-greedy.

softmax

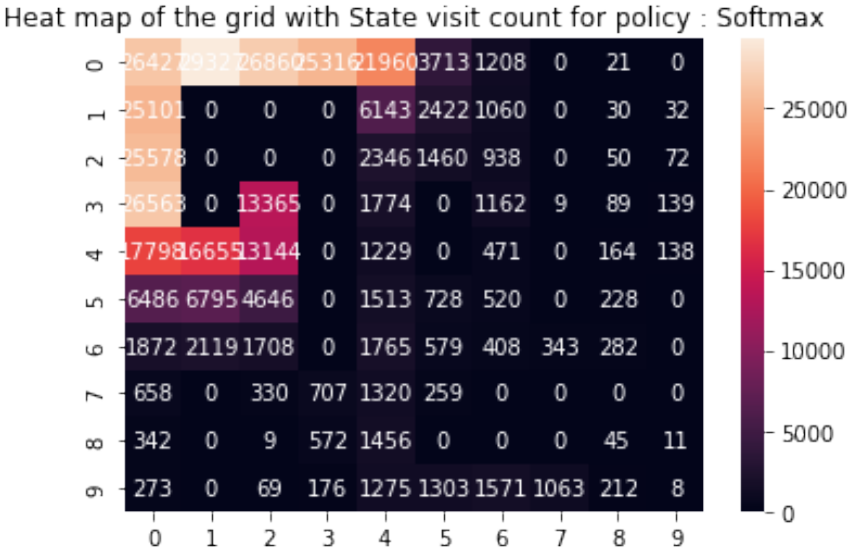
Optimal Hyperparameters: $\{\tau : 1.0, \alpha : 0.2, \gamma : 1.0\}$

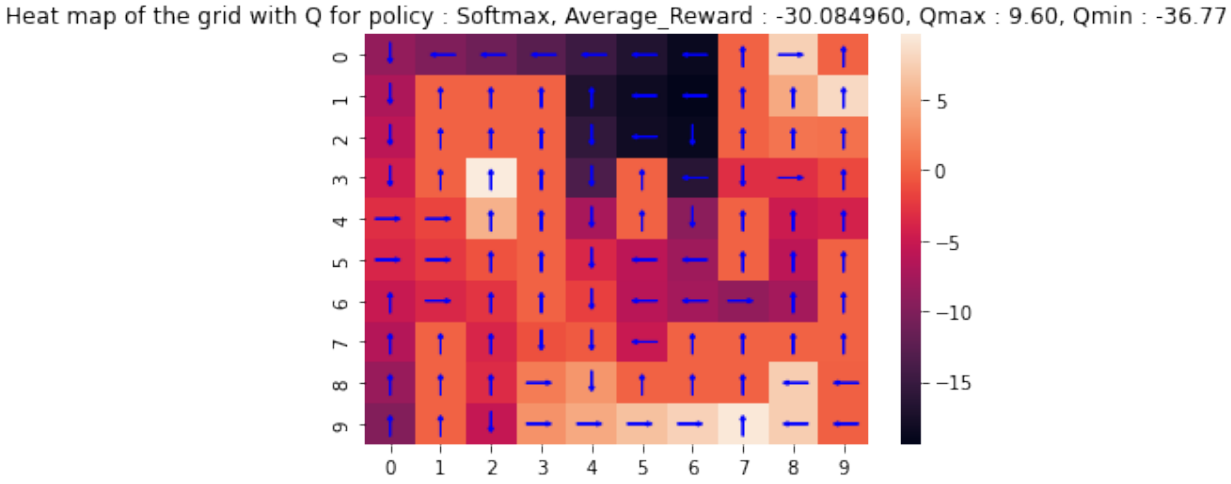


(a) Rewards vs Episodes



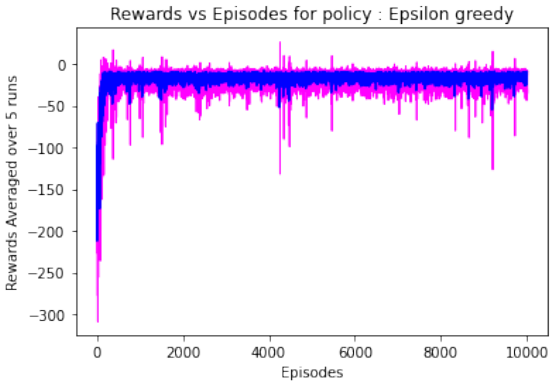
(b) Number of steps vs Episodes



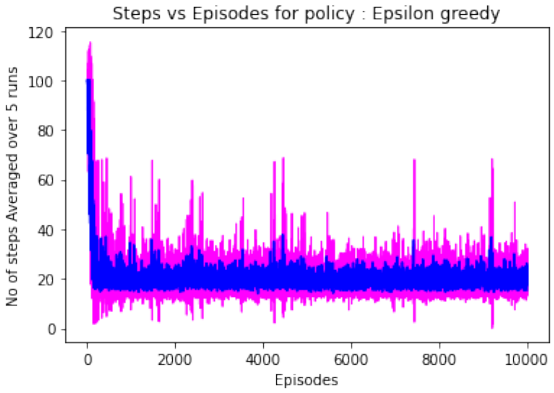


ϵ-greedy

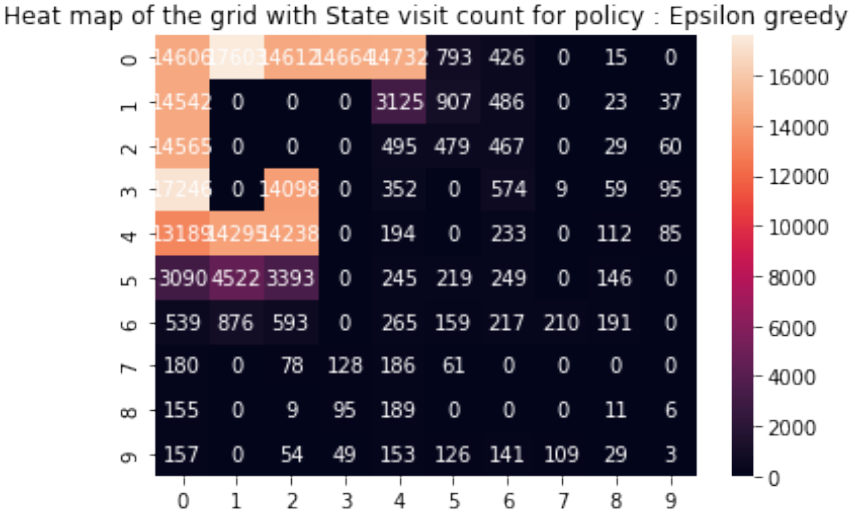
Optimal Hyperparameters: $\{\epsilon : 0.01, \alpha : 0.2, \gamma : 1.0\}$

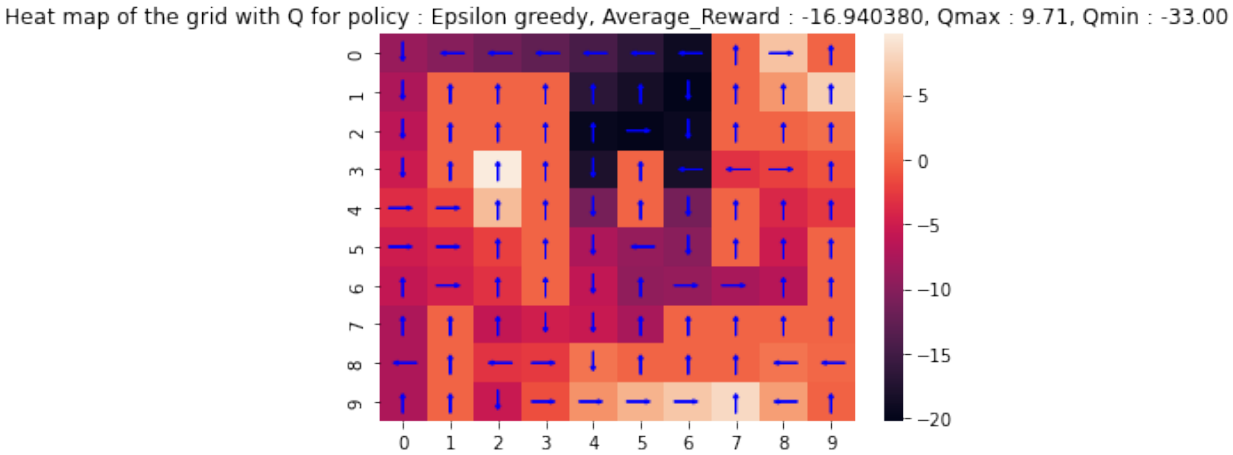


(a) Rewards vs Episodes



(b) Number of steps vs Episodes





Inferences for 5.2

Similarities

- Epsilon greedy gives higher average reward when compared to Softmax.
- Both the algorithms are giving more expected return around goal (8, 7).

Differences

- In SARSA Softmax is exploring two goal states (2,2) and (8,7) more number of times when compared to Epsilon greedy where one of the goal state(2,2) is visited majority of the time.
- In QLEARNING both Softmax and Epsilon greedy one goal state(2,2) are exploring more number of times than the other goal states.

Justification of path

- Both the policies learnt by SARSA and Q Learning are leading to the goal state (2, 2). SARSA gives a safe path and Q Learning gives an optimal path. Since travelling to all three goal states gives nearly the same reward, Q Learning is showing the path to (2, 2).
- Whereas along the path to (0, 9) there is one restart state and one bad state, and along the path to (8, 7) there are many bad states, whereas along the path to (2, 2) there is only one bad state, hence SARSA gives a path to reach (2, 2) which is a safer path.

5.3 Start state $= (0, 4)$, wind $=$ True, $p = 1.0$

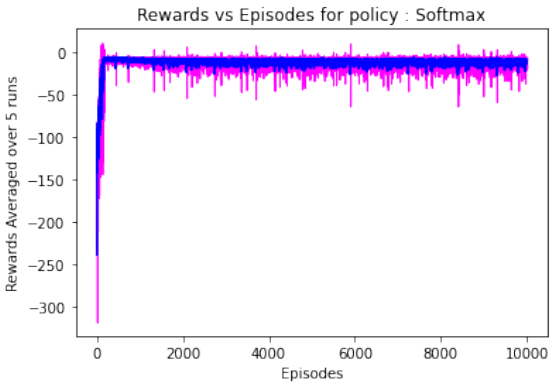
Experiment 3 - SARSA

- Algorithm: SARSA
- Wind: True
- $p = 1.0$ (deterministic step)

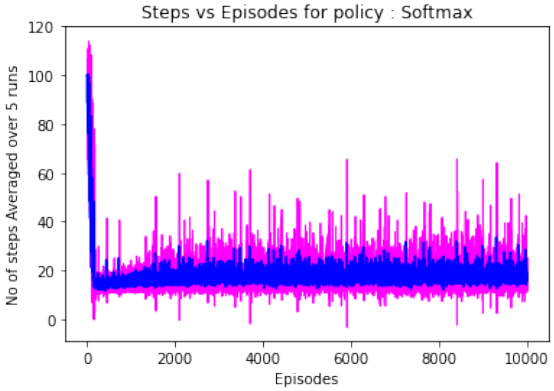
Choosing the best policy between softmax and ϵ-greedy.

softmax

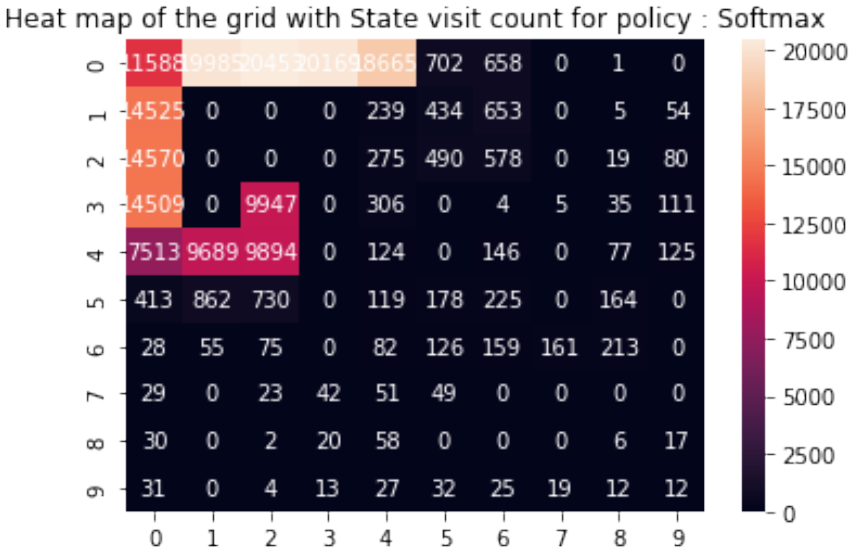
Optimal Hyperparameters: $\{\tau : 1.0, \alpha : 0.2, \gamma : 1.0\}$

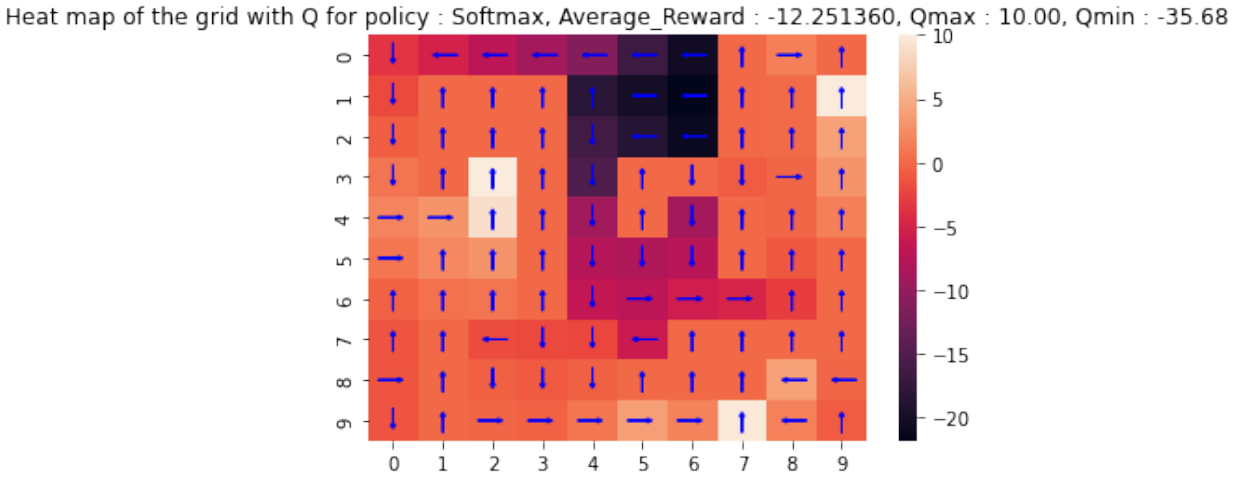


(a) Rewards vs Episodes



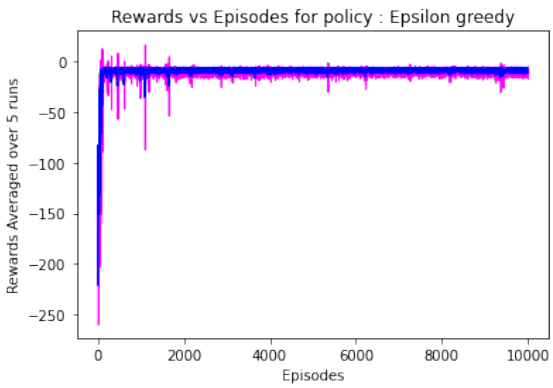
(b) Number of steps vs Episodes



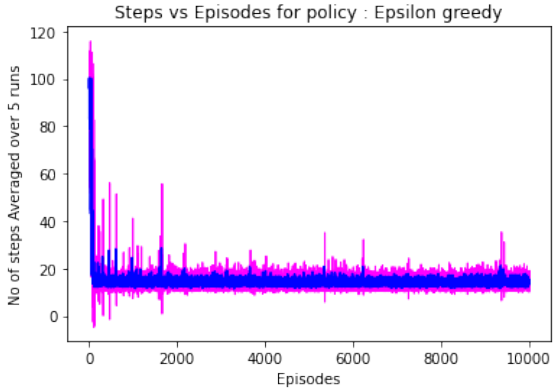


ϵ-greedy

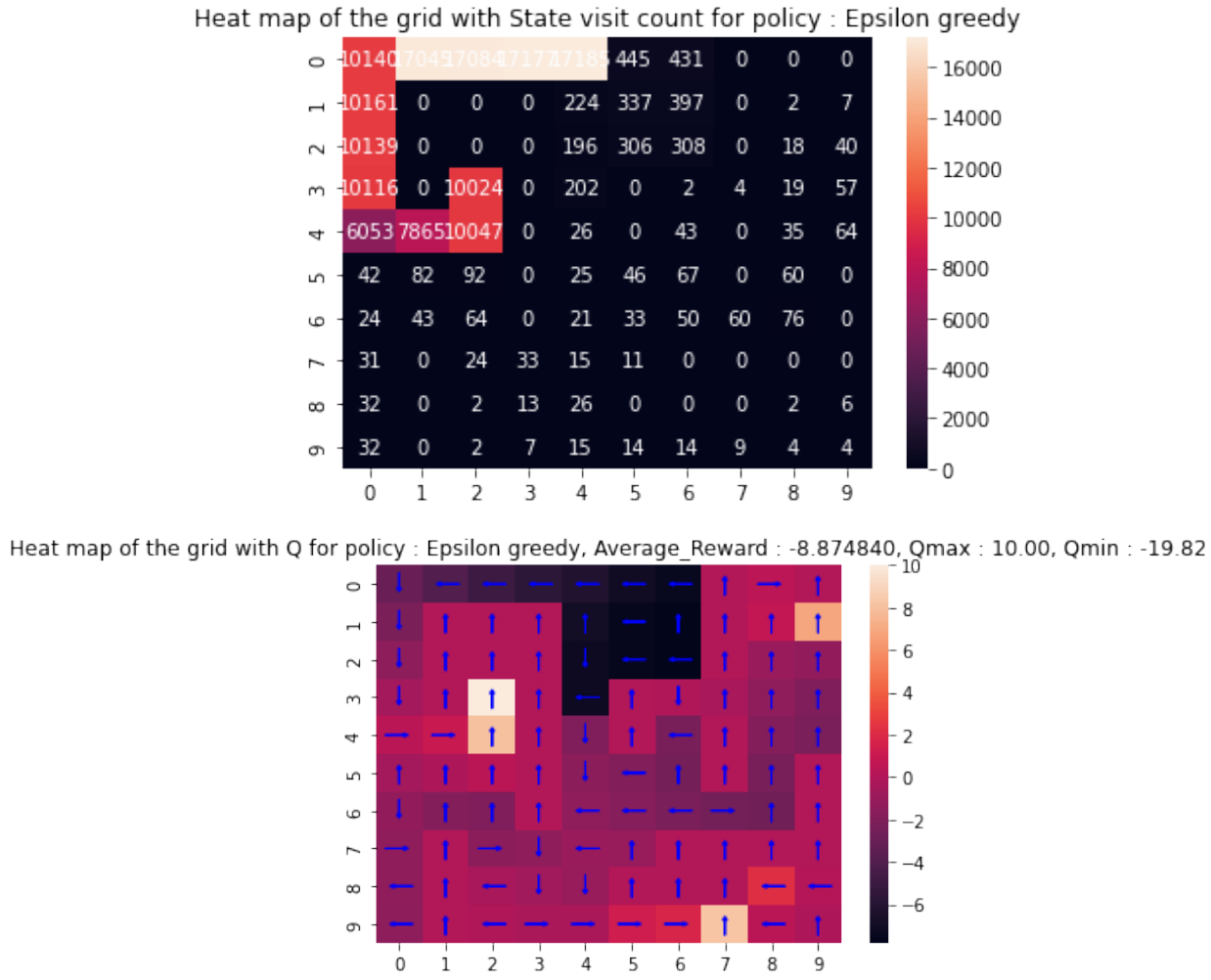
Optimal Hyperparameters: $\{\epsilon : 0.01, \alpha : 0.2, \gamma : 0.9\}$



(a) Rewards vs Episodes



(b) Number of steps vs Episodes



Experiment 9 - Q-Learning

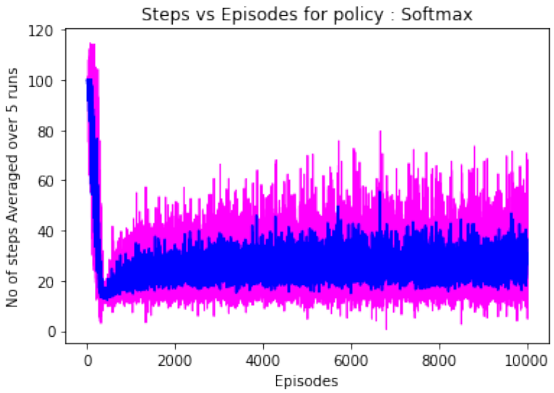
- Algorithm: Q-Learning
- Wind: True
- $p = 1.0$ (deterministic step)

Choosing the best policy between softmax and ϵ-greedy.

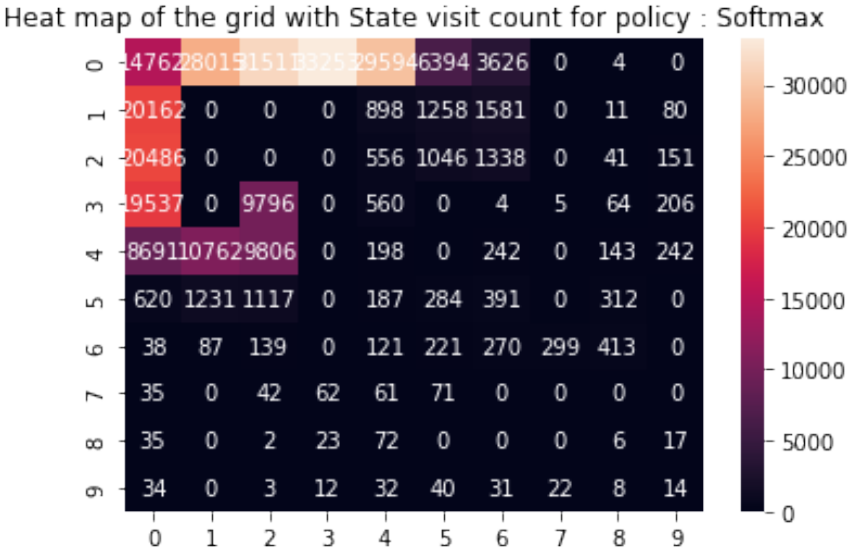
softmax

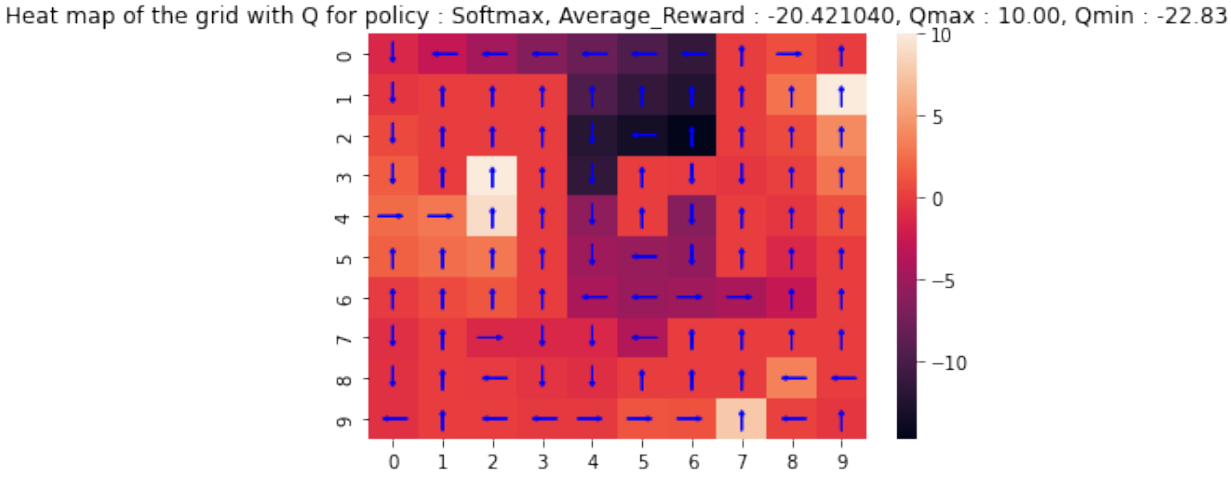
Optimal Hyperparameters: $\{\tau : 1.0, \alpha : 0.1, \gamma : 1.0\}$

(a) Rewards vs Episodes



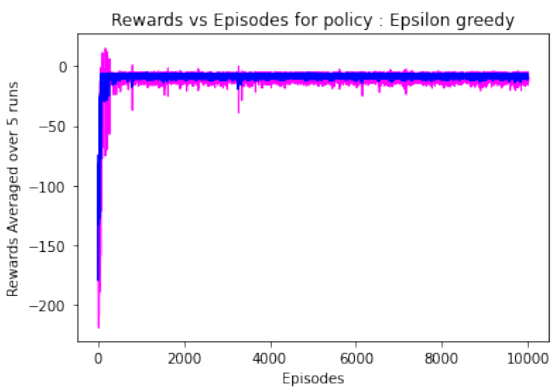
(b) Number of steps vs Episodes



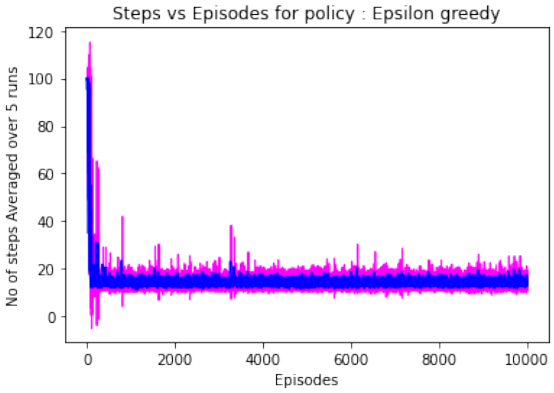


ϵ-greedy

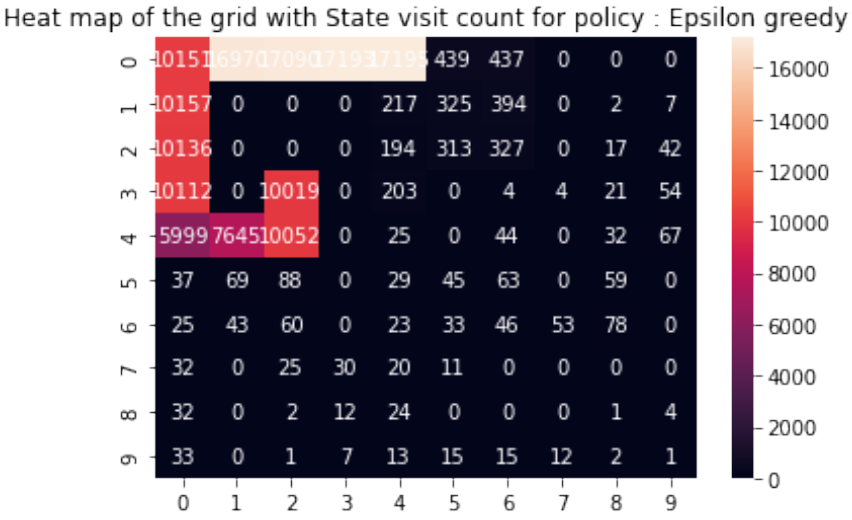
Optimal Hyperparameters: $\{\epsilon : 0.01, \alpha : 0.2, \gamma : 0.9\}$

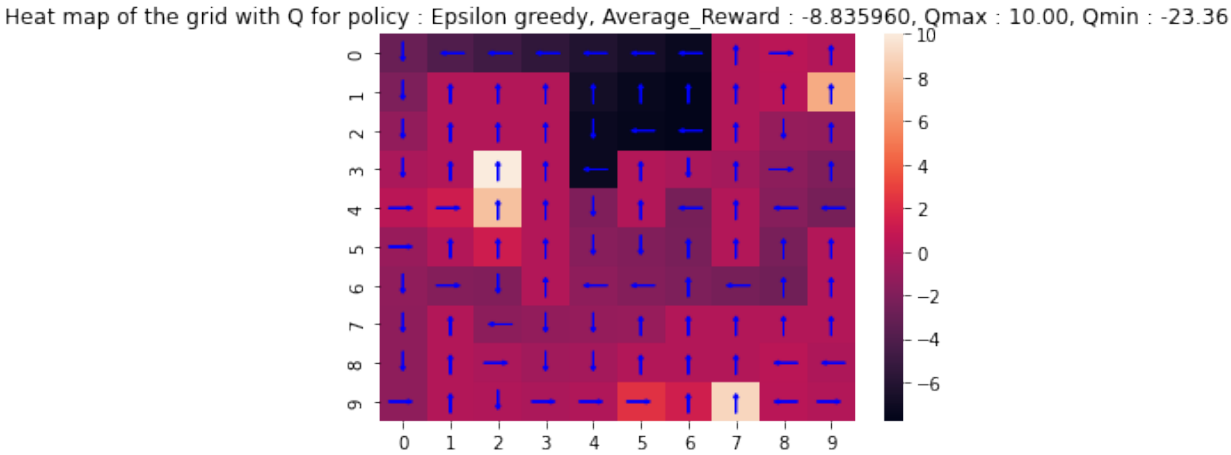


(a) Rewards vs Episodes



(b) Number of steps vs Episodes





Inferences for 5.3

Similarities

- Epsilon greedy gives higher average reward when compared to Softmax.
- Both the algorithms are exploring (2, 2) more number of times when compared to other goal states.

Differences

- In SARSA When softmax policy is used it gives less negative reward.
- In QLEARNING When Softmax policy is used it gives more negative reward.

Justification of path

- Both the policies learnt by SARSA and Q Learning are leading to the goal state (2, 2). SARSA gives a safe path and Q Learning gives an optimal path. Since travelling to all three goal states gives nearly the same reward, Q Learning is showing the path to (2, 2).
- Whereas along the path to (0, 9) there is one restart state and one bad state, and along the path to (8, 7) there are many bad states, whereas along the path to (2, 2) there is only one bad state, hence SARSA gives a path to reach (2, 2) which is a safer path.

5.4 Start state = $(3, 6)$, wind = False, $p = 1.0$

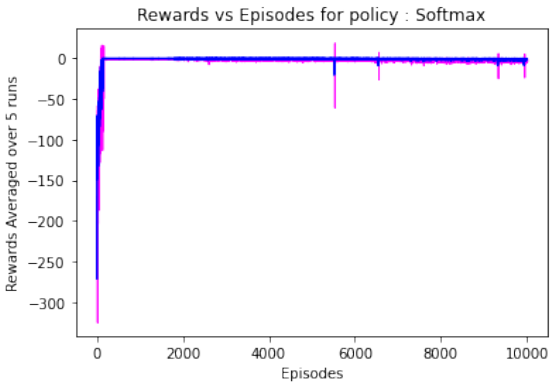
Experiment 4 - SARSA

- Algorithm: SARSA
- Wind: False
- $p = 1.0$ (deterministic step)

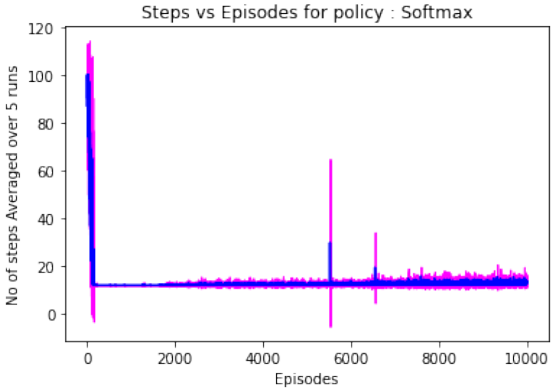
Choosing the best policy between softmax and ϵ-greedy.

softmax

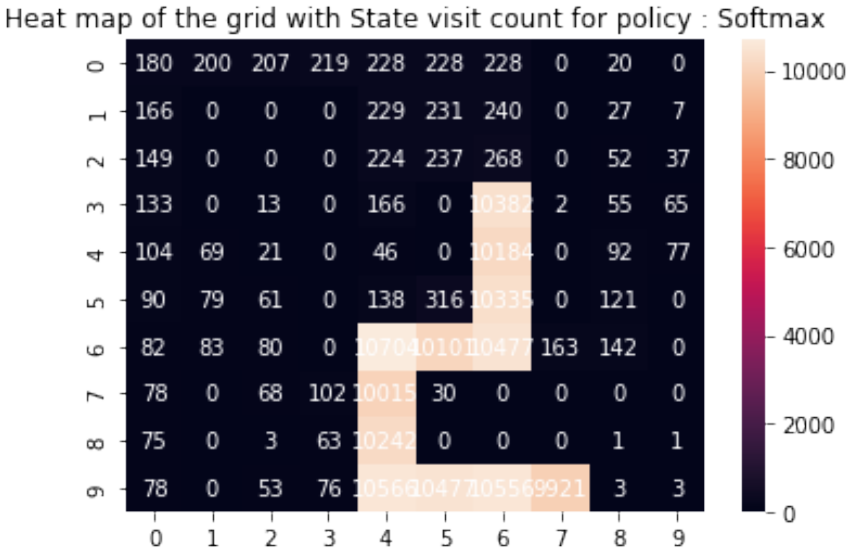
Optimal Hyperparameters: $\{\tau : 1.0, \alpha : 0.2, \gamma : 1.0\}$

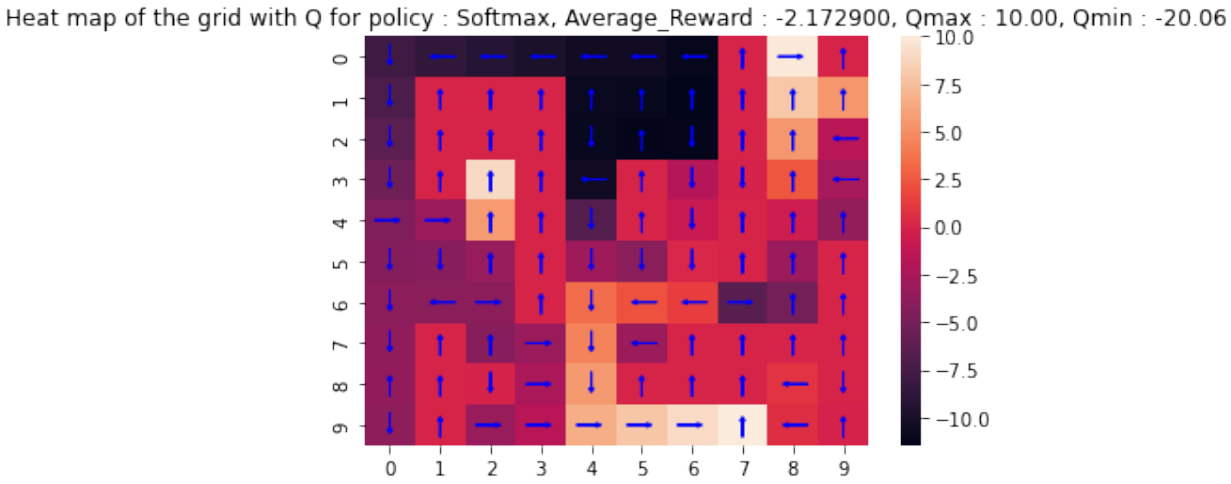


(a) Rewards vs Episodes



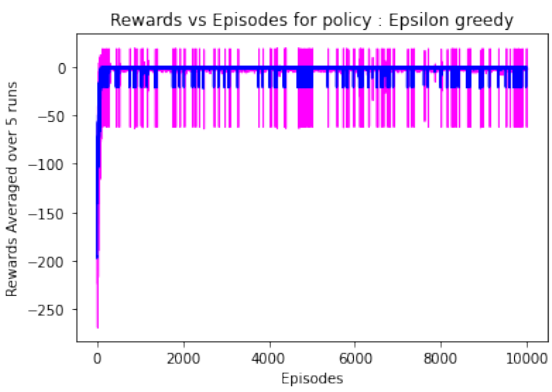
(b) Number of steps vs Episodes



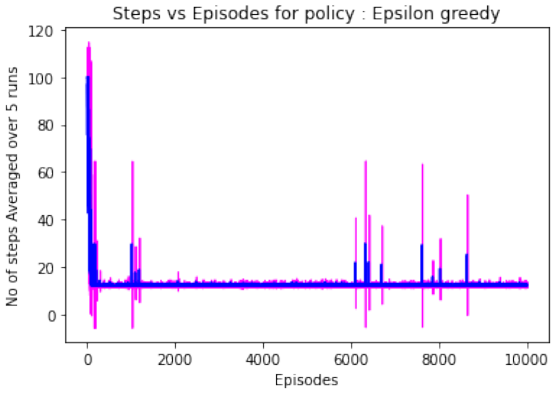


ϵ-greedy

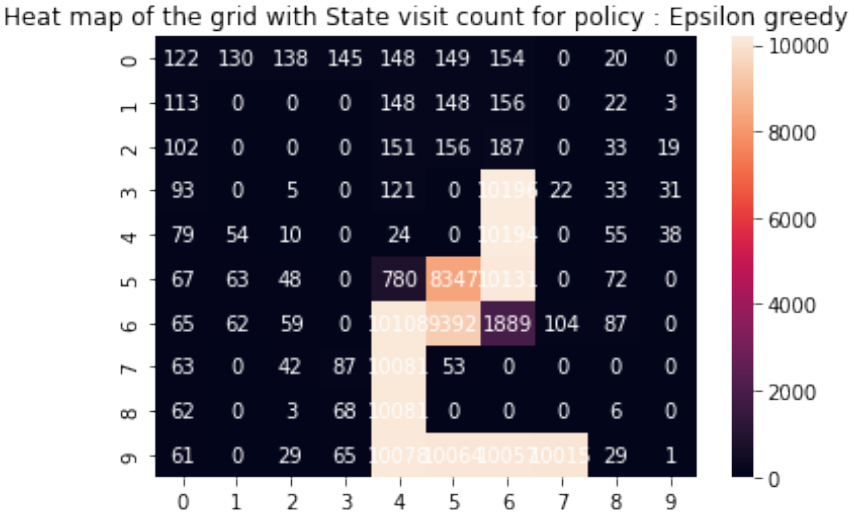
Optimal Hyperparameters: $\{\epsilon : 0.01, \alpha : 0.2, \gamma : 1.0\}$

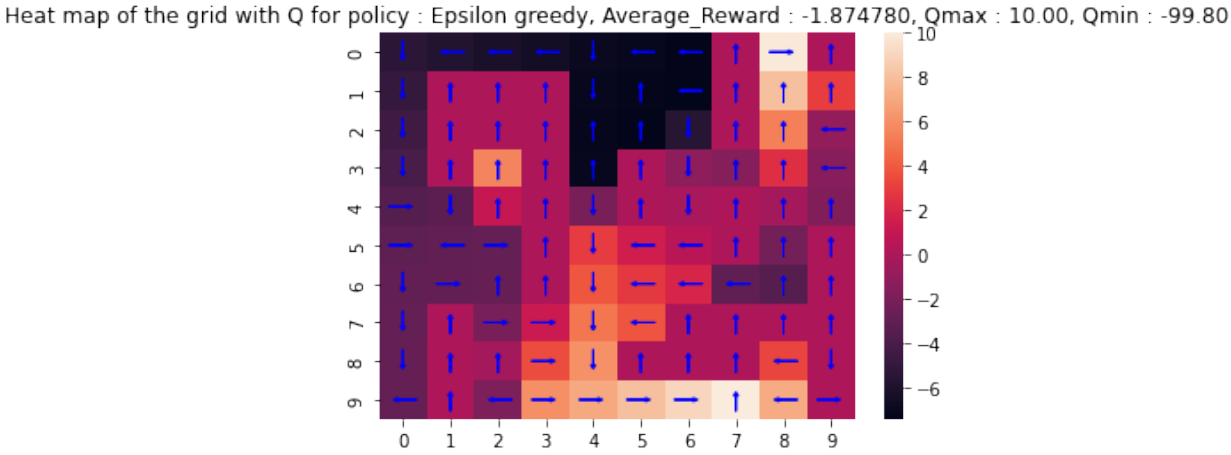


(a) Rewards vs Episodes



(b) Number of steps vs Episodes





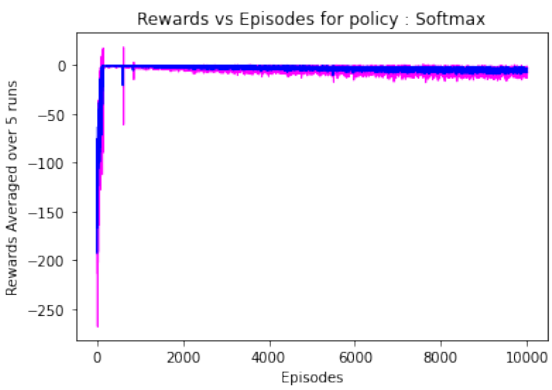
Experiment 10 - Q-Learning

- Algorithm: Q-Learning
- Wind: False
- $p = 1.0$ (deterministic step)

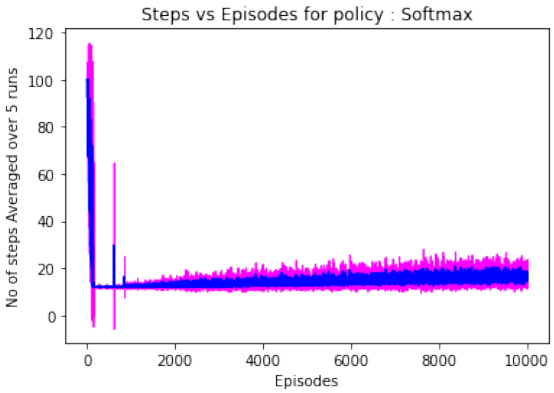
Choosing the best policy between softmax and ϵ-greedy.

softmax

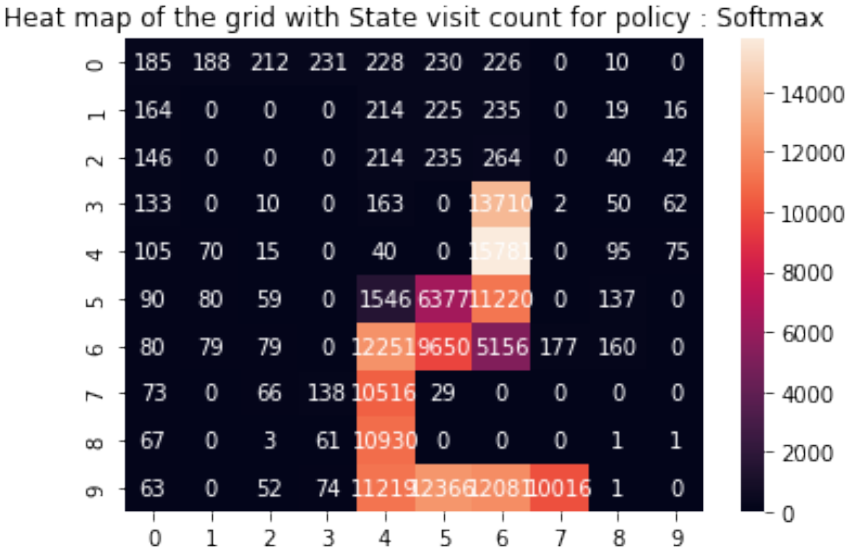
Optimal Hyperparameters: $\{\tau : 1.0, \alpha : 0.2, \gamma : 1.0\}$

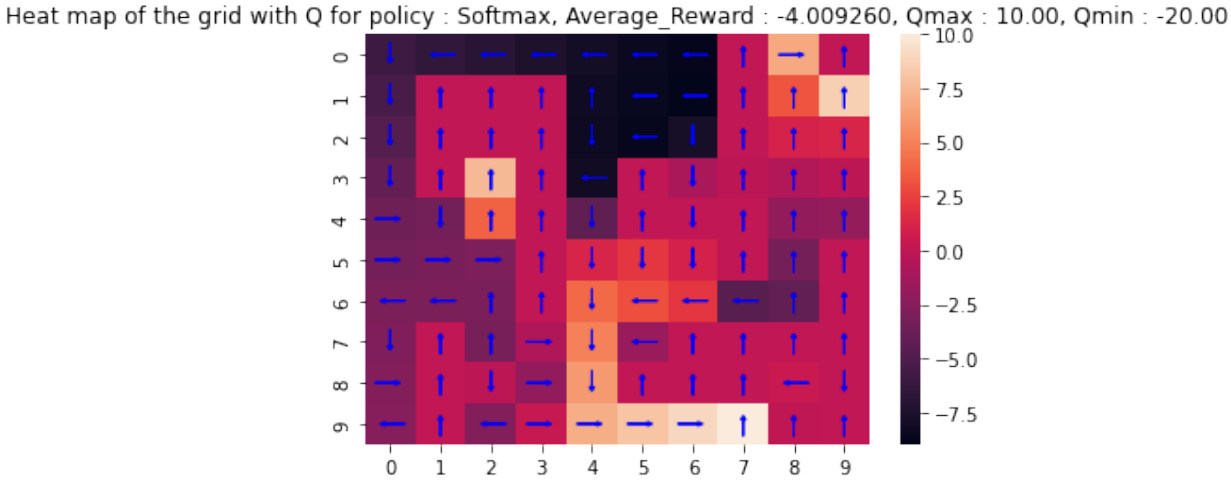


(a) Rewards vs Episodes



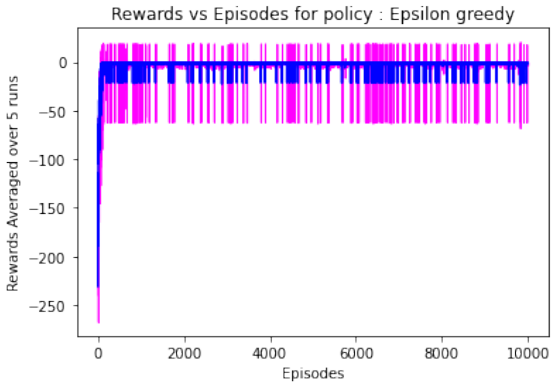
(b) Number of steps vs Episodes



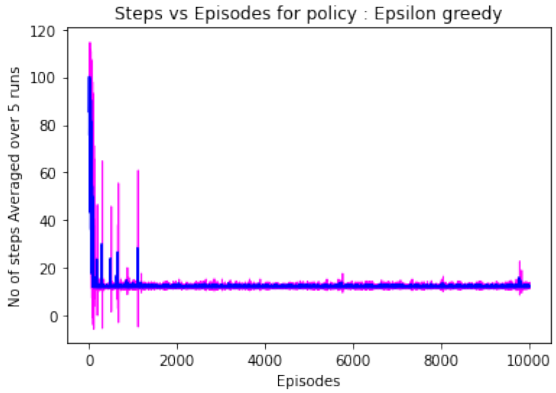


ϵ-greedy

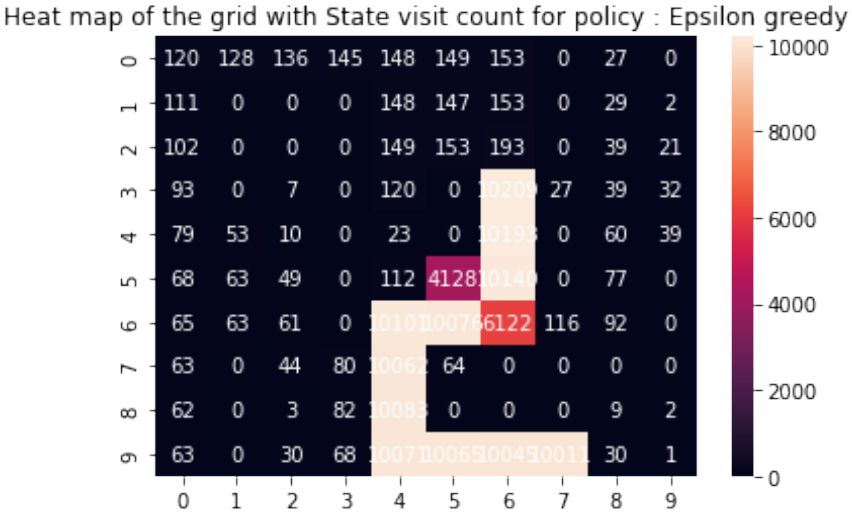
Optimal Hyperparameters: $\{\epsilon : 0.01, \alpha : 0.2, \gamma : 1.0\}$

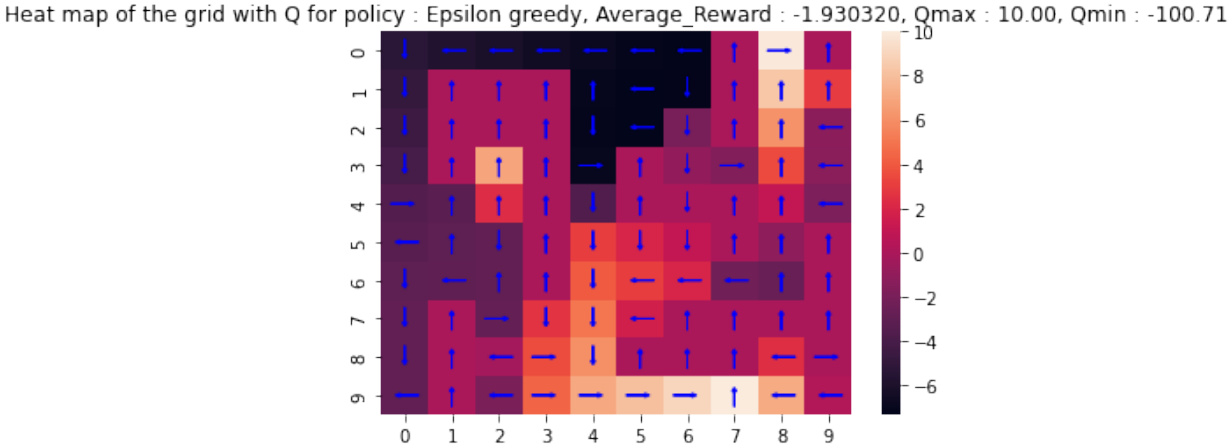


(a) Rewards vs Episodes



(b) Number of steps vs Episodes





Inferences for 5.4

Similarities

- Both the algorithms are exploring more towards the goal (8, 7) when compared to (2, 2) and (0, 9). This may be due to the fact that while moving towards (0, 9) there is a restart state, so it takes longer time to find a path to reach (0, 9) and it takes more number of steps to reach (2, 2).

Differences

- Softmax gives slightly better reward for sarsa than Q learning.

Justification of path

- Both the policies learnt by SARSA and Q Learning are leading to the goal state (8, 7). SARSA gives a safe path and Q Learning gives an optimal path. Since travelling to all three goal states gives nearly the same reward, it doesn't matter which path it takes.
- Along the path to (0, 9) there is one restart state and one bad state, and along the path to (8, 7) there are many bad states, whereas along the path to (2, 2) there is only one bad state, but in the deterministic case it doesn't matter which path it takes.

5.5 Start state $= (3, 6)$, wind $=$ False, $p = 0.7$

Experiment 5 - SARSA

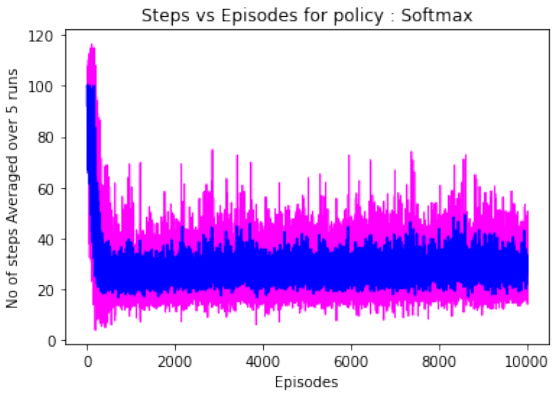
- Algorithm: SARSA
- Wind: False
- $p = 0.7$ (stochastic step)

Choosing the best policy between softmax and ϵ-greedy.

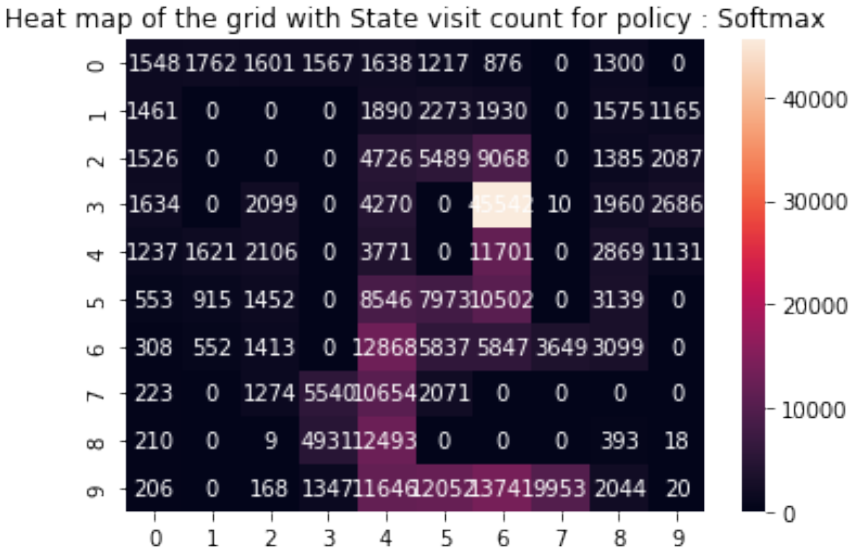
softmax

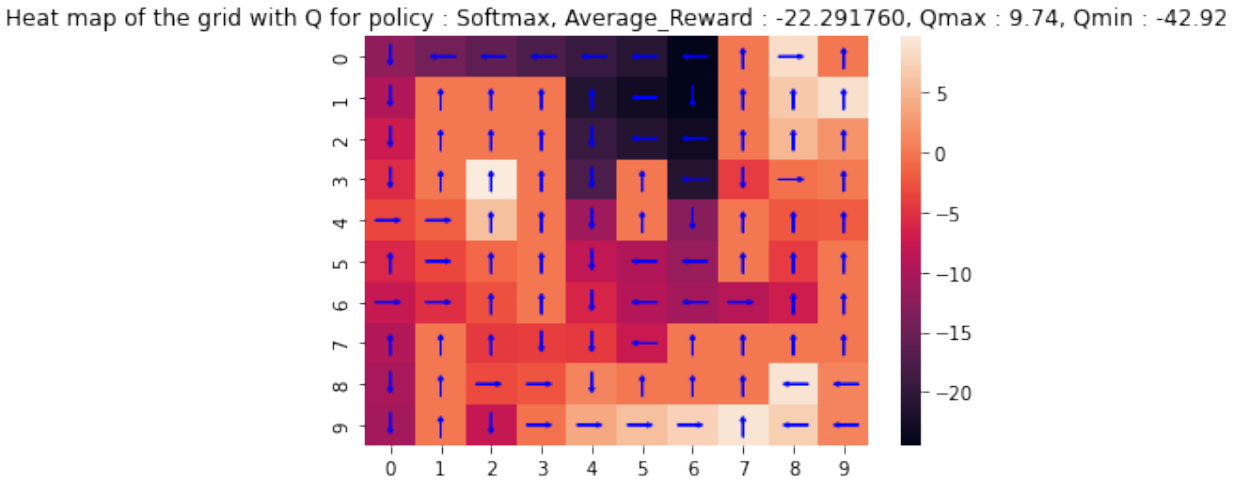
Optimal Hyperparameters: $\{\tau : 1.0, \alpha : 0.2, \gamma : 1.0\}$

(a) Rewards vs Episodes



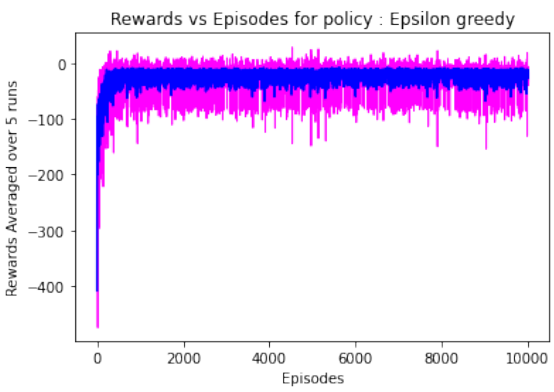
(b) Number of steps vs Episodes



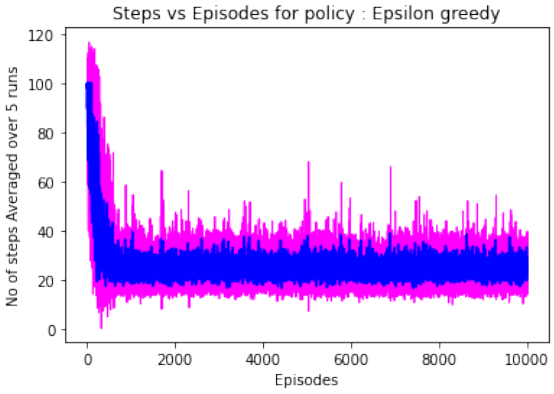


ϵ-greedy

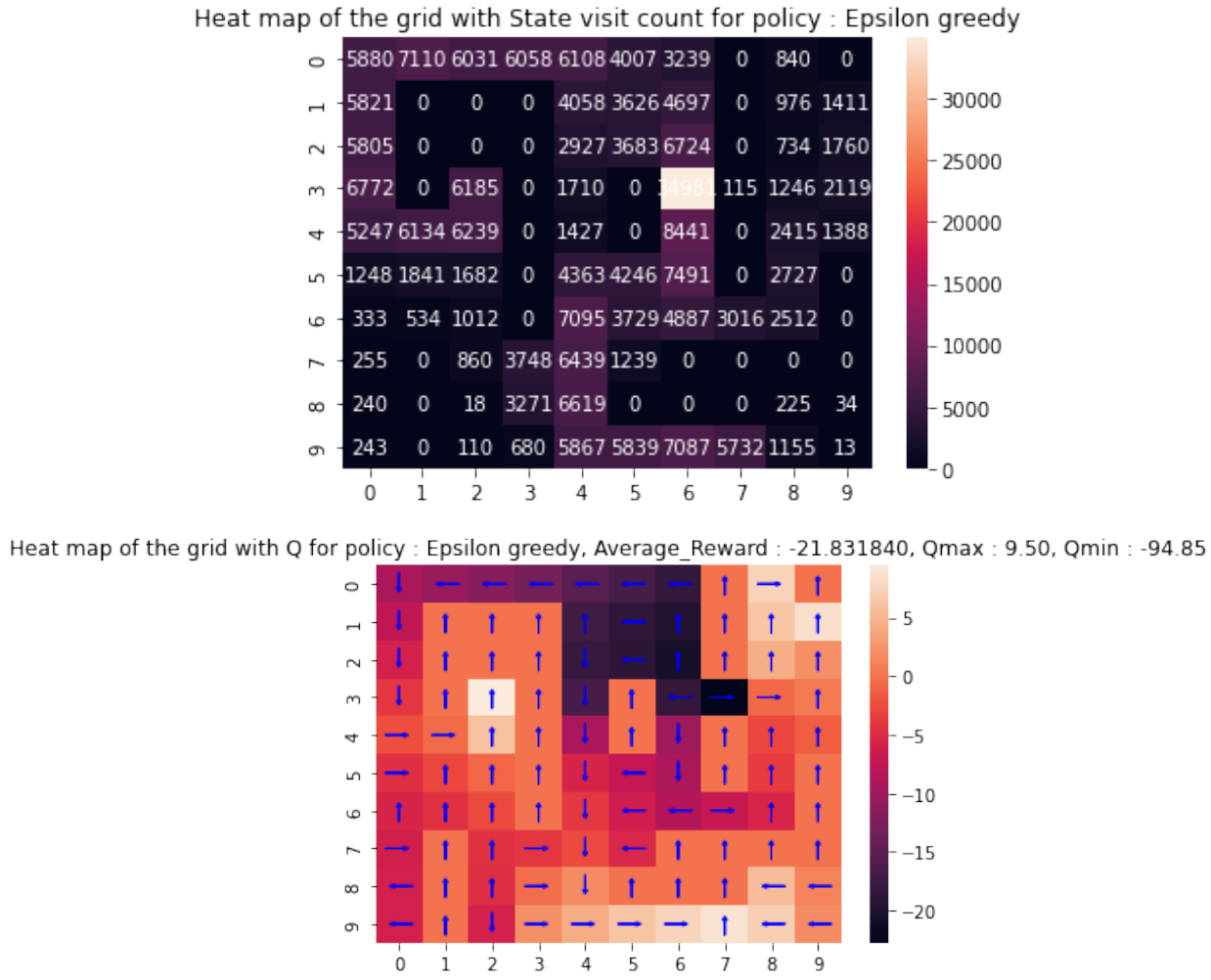
Optimal Hyperparameters: $\{\epsilon : 0.01, \alpha : 0.1, \gamma : 1.0\}$



(a) Rewards vs Episodes



(b) Number of steps vs Episodes



Experiment 11 - Q-Learning

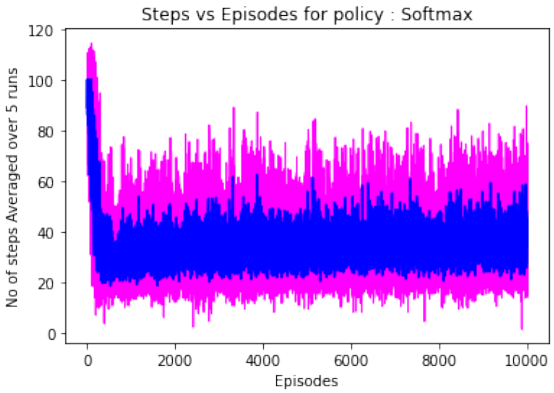
- Algorithm: Q-Learning
- Wind: False
- $p = 0.7$ (stochastic step)

Choosing the best policy between softmax and ϵ-greedy.

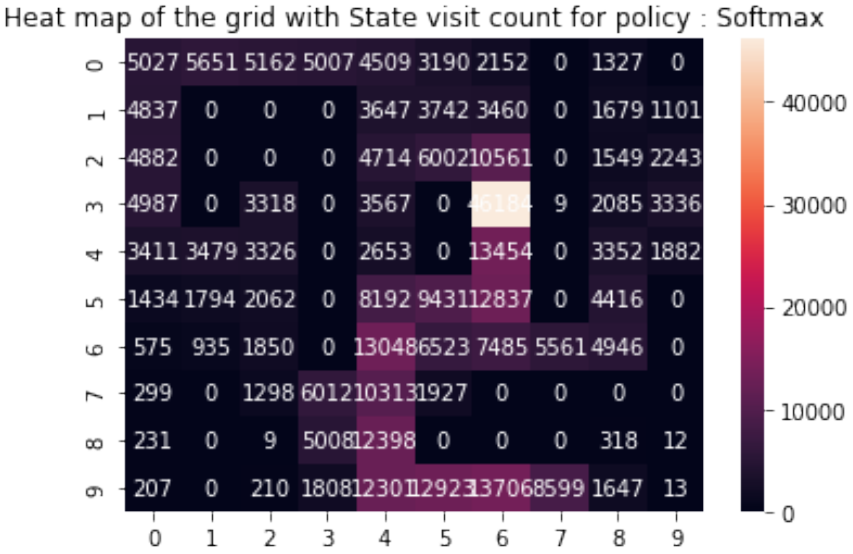
softmax

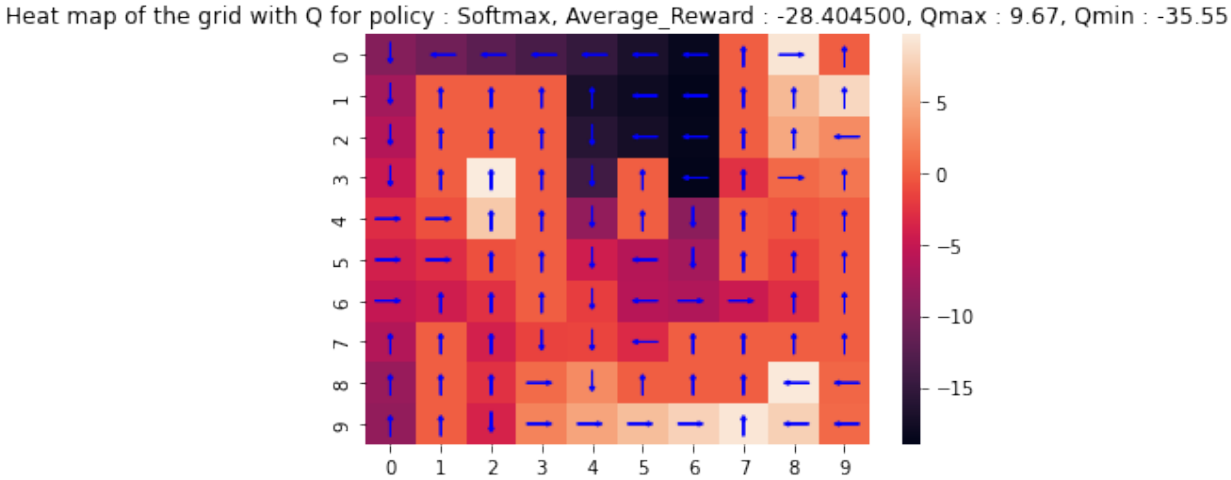
Optimal Hyperparameters: $\{\tau : 1.0, \alpha : 0.2, \gamma : 1.0\}$

(a) Rewards vs Episodes



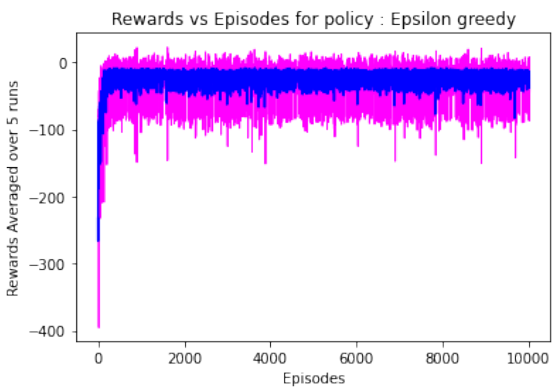
(b) Number of steps vs Episodes



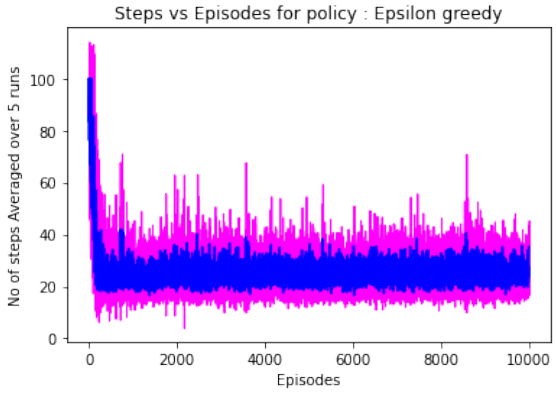


ϵ-greedy

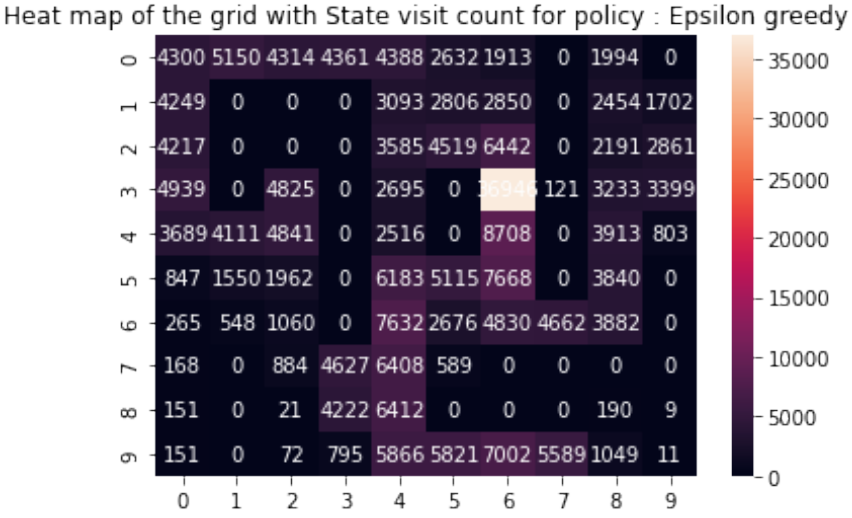
Optimal Hyperparameters: $\{\epsilon : 0.01, \alpha : 0.2, \gamma : 1.0\}$

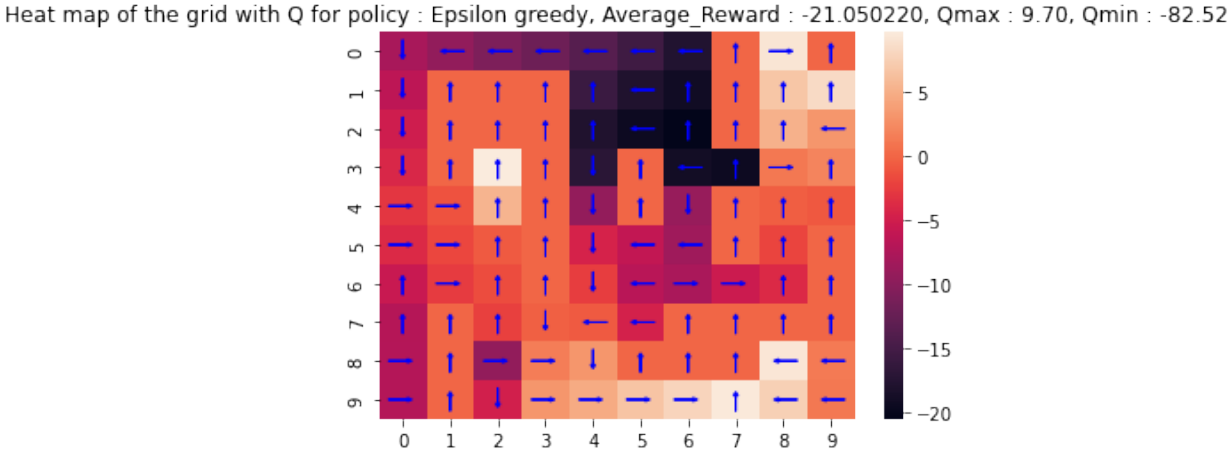


(a) Rewards vs Episodes



(b) Number of steps vs Episodes





Inferences for 5.5

Similarities

- Both the algorithms are exploring all three goal states nearly equal number of times.

Differences

- Softmax gives slightly better reward for sarsa than Q learning.

Justification of path

- Both the policies learnt by SARSA and Q Learning are leading to all three goal state almost equally, but the path for reaching (8, 7) has slightly more expected return. SARSA gives a safe path and Q Learning gives an optimal path.
- Sarsa gives the more safer path more expected return. Q Learning gives more optimal path, that is to reach (8, 7).

5.6 Start state $= (3, 6)$, wind $=$ True, $p = 1.0$

Experiment 6 - SARSA

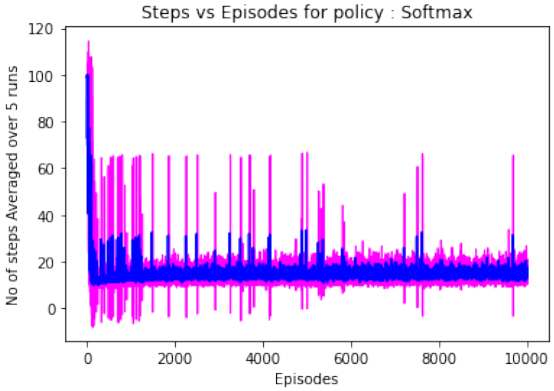
- Algorithm: SARSA
- Wind: True
- $p = 1.0$ (deterministic step)

Choosing the best policy between softmax and ϵ-greedy.

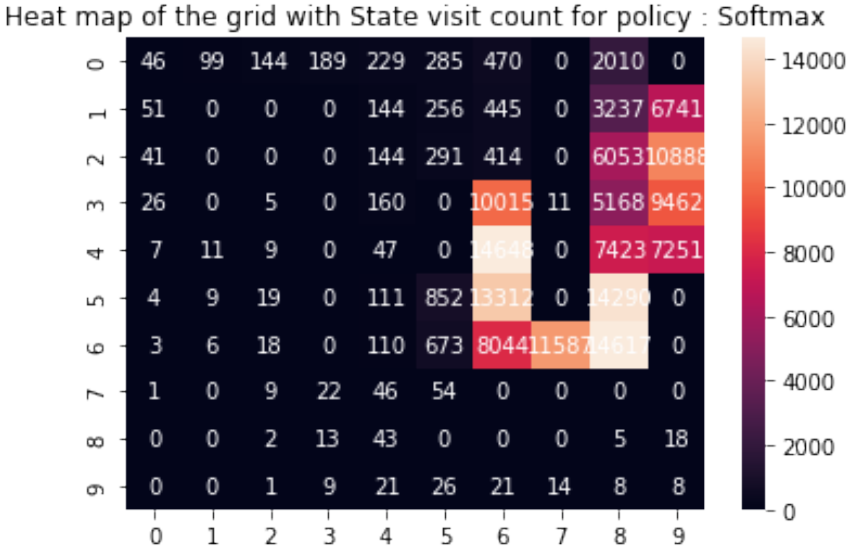
softmax

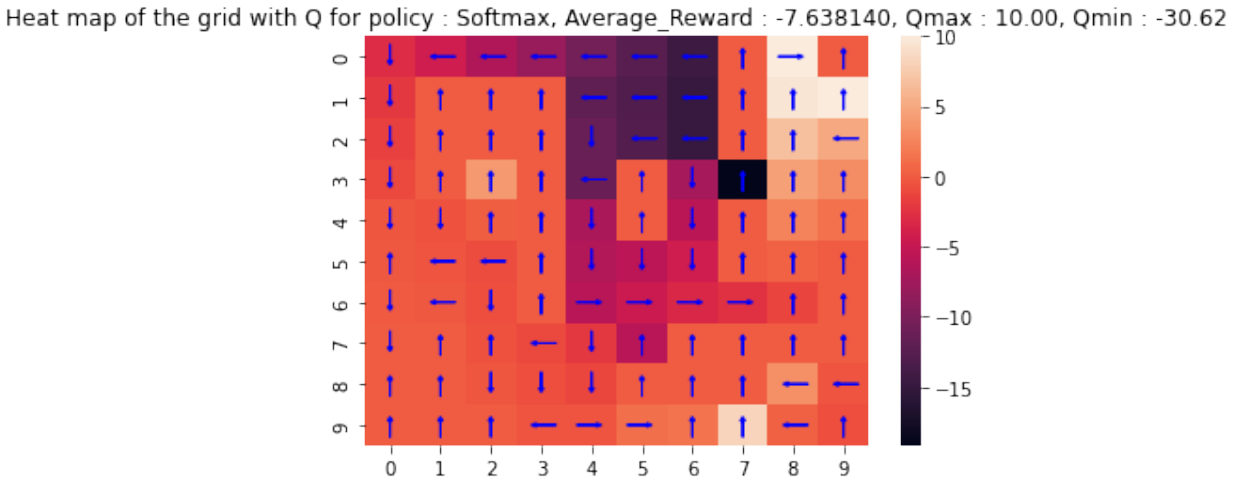
Optimal Hyperparameters: $\{\tau : 1.0, \alpha : 0.2, \gamma : 1.0\}$

(a) Rewards vs Episodes



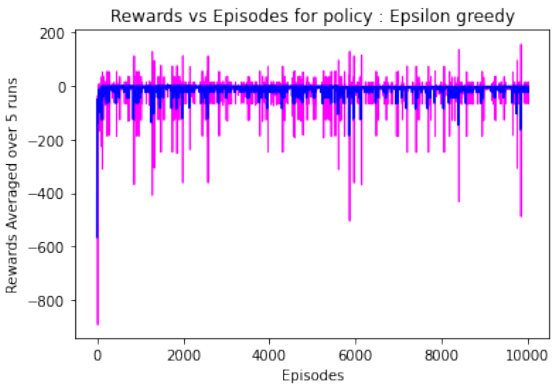
(b) Number of steps vs Episodes



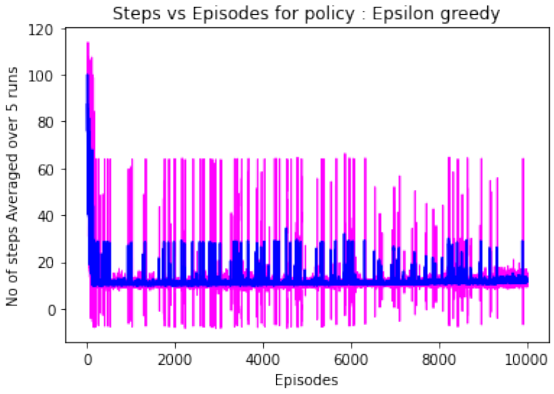


ϵ-greedy

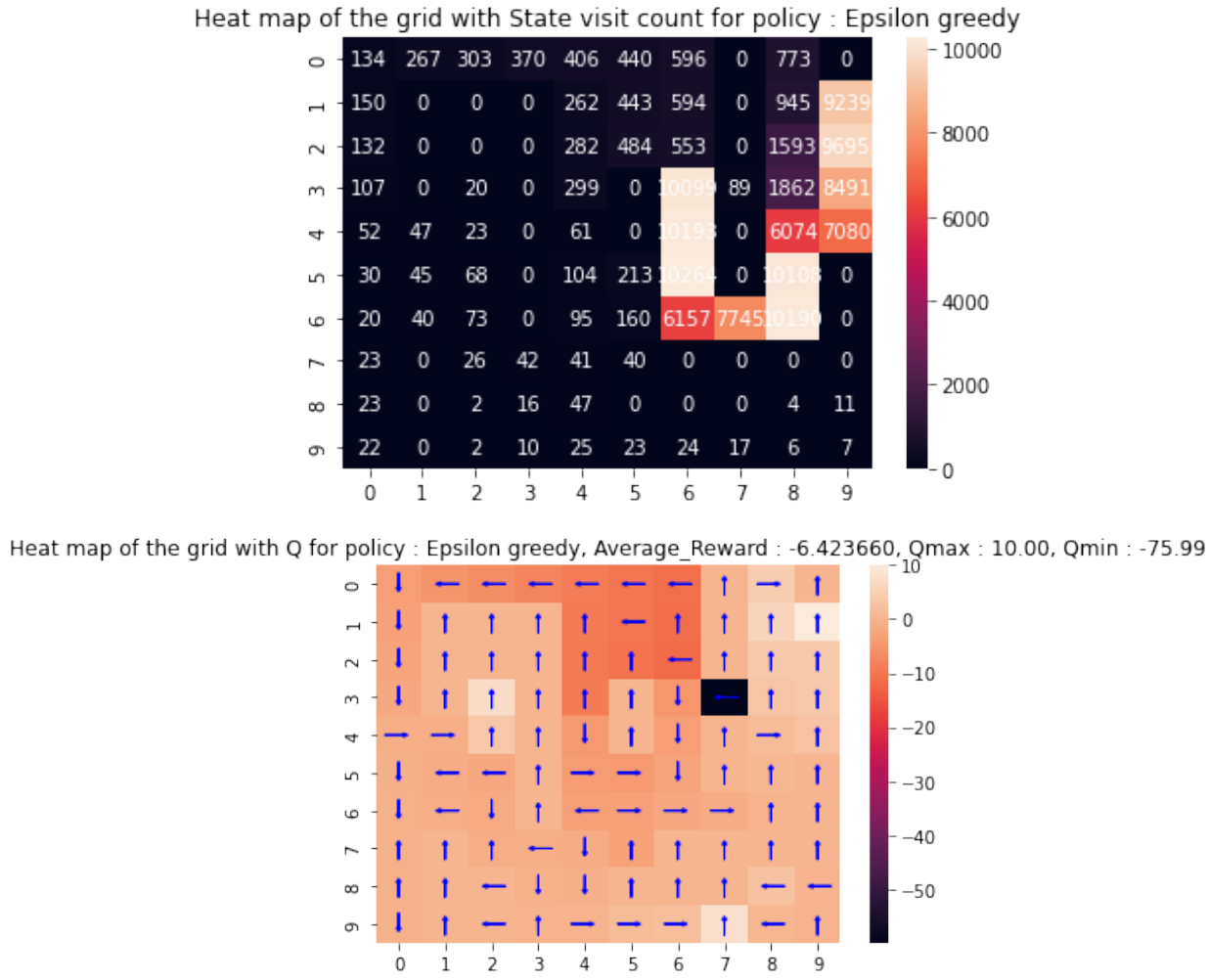
Optimal Hyperparameters: $\{\epsilon : 0.01, \alpha : 0.1, \gamma : 1.0\}$



(a) Rewards vs Episodes



(b) Number of steps vs Episodes



Experiment 12 - Q-Learning

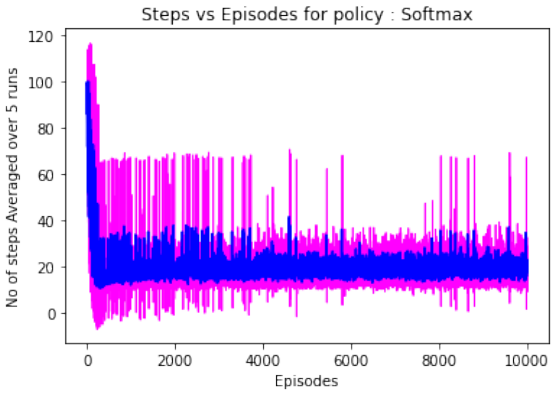
- Algorithm: Q-Learning
- Wind: True
- $p = 1.0$ (deterministic step)

Choosing the best policy between softmax and ϵ-greedy.

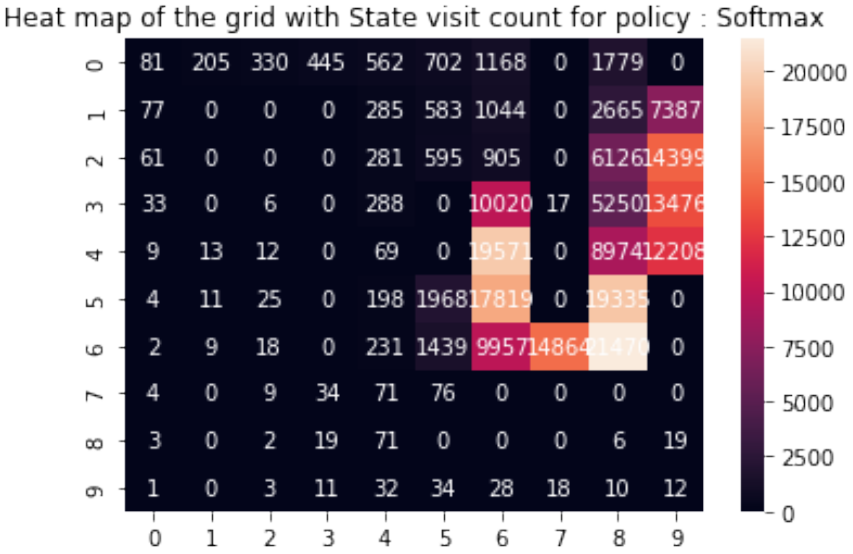
softmax

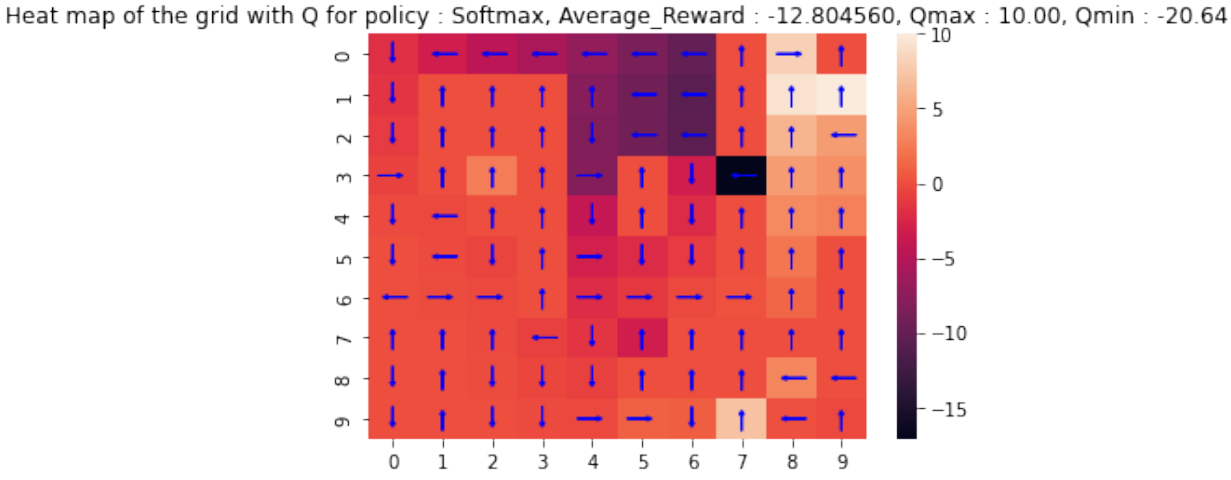
Optimal Hyperparameters: $\{\tau : 1.0, \alpha : 0.1, \gamma : 1.0\}$

(a) Rewards vs Episodes



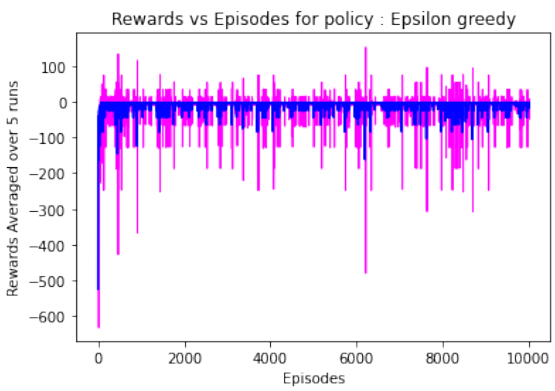
(b) Number of steps vs Episodes



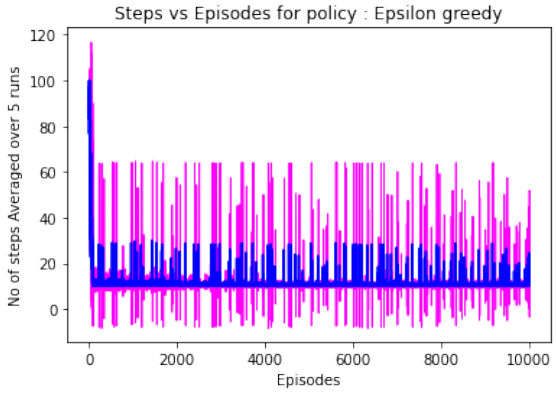


ϵ-greedy

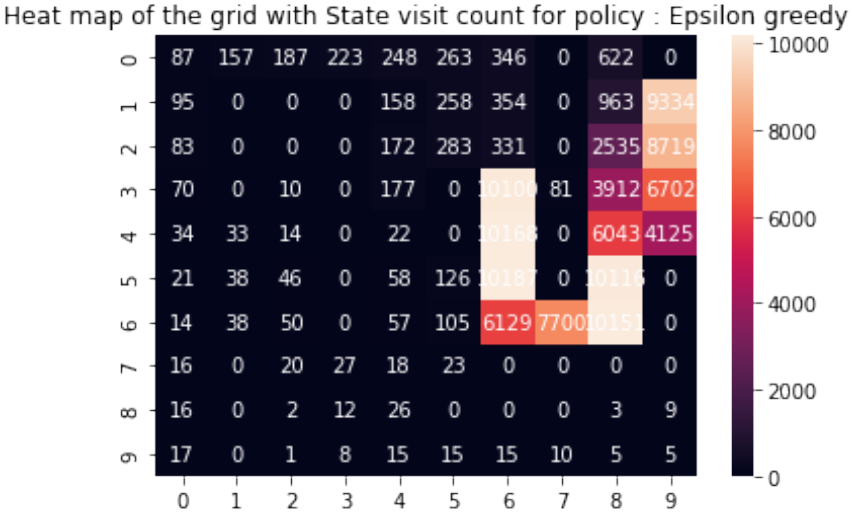
Optimal Hyperparameters: $\{\epsilon : 0.01, \alpha : 0.2, \gamma : 0.9\}$

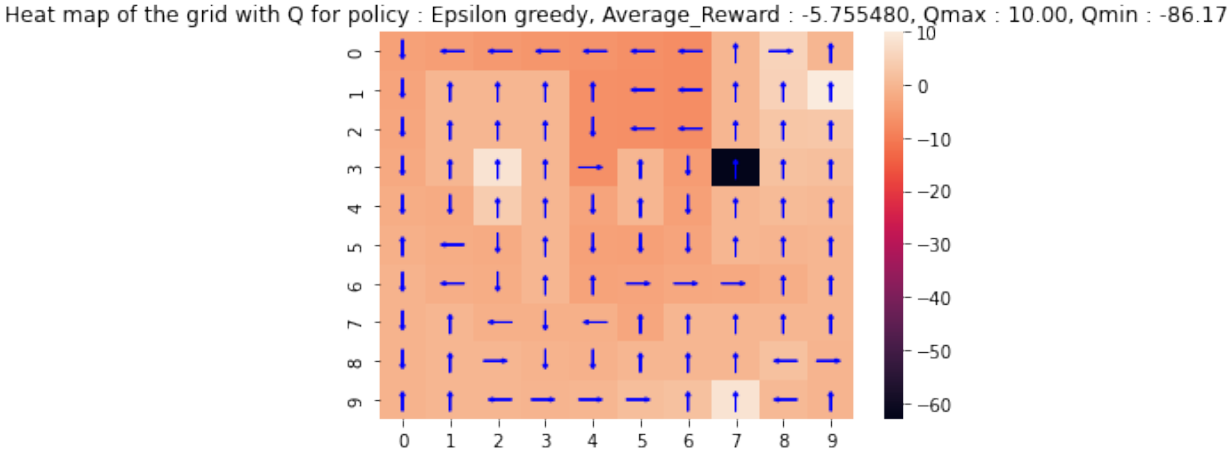


(a) Rewards vs Episodes



(b) Number of steps vs Episodes





Inferences for 5.6

Similarities

- Both the algorithms are exploring the goal state (0, 9) more number of times.

Differences

- Softmax gives slightly better reward for sarsa than Q learning.

Justification of path

- Both the policies learnt by SARSA and Q Learning are leading to (0, 9). Sarsa gives the safer path. In this case wind pushes the agent towards right with some probability. But in the given path even if the wind pushes it doesn't reach the restart state, hence it is the safer path.
- Q Learning gives the path because it is optimal.