experiments

November 7, 2023

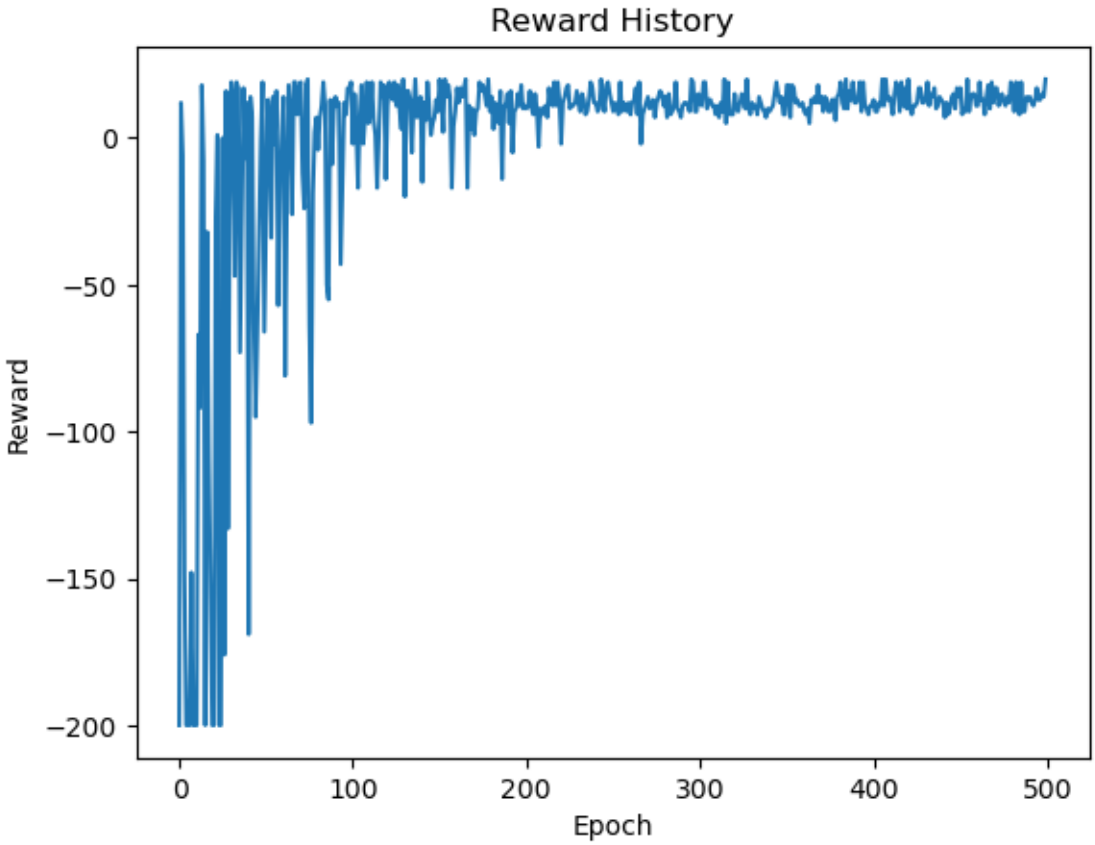
[1]:
```
import matplotlib.pyplot as plt

from sarsaLearner import sarsaLearner
from qLearner import qLearner
```

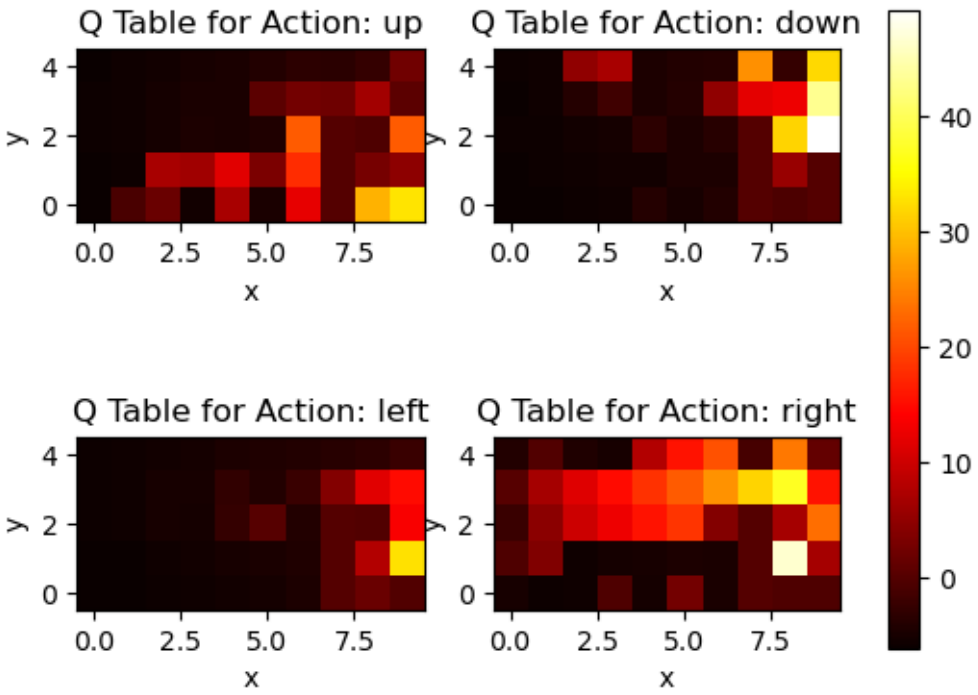
[2]:
```
# Common Parameters
epochs = 500 # number of episodes
time_steps = 200 # max time steps per episode
alpha = 0.25 # learning rate
gamma = 0.9 # discount factor
epsilon = 0.9 # exploration rate
epsilon_decay = 0.99 # exploration decay rate
```

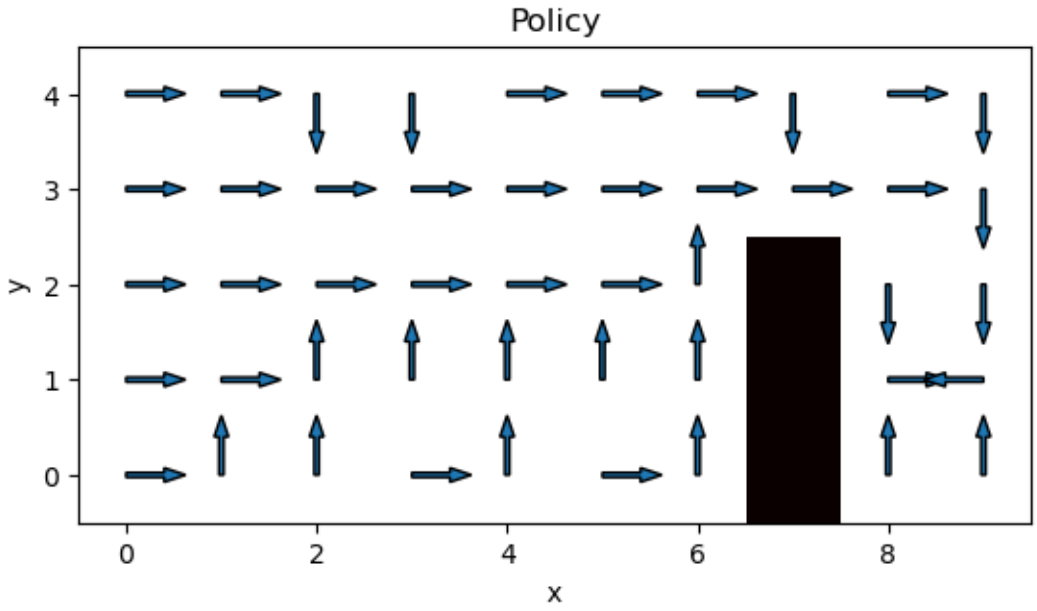
[9]:
```
## Experiments for SARSA learning

rng_door = False # random door

# Initialize learner
sarsaAgent = sarsaLearner(epochs, time_steps, alpha, gamma, epsilon,␣
 ↪epsilon_decay, rng_door)

# Run experiment
sarsaAgent.run(verbose=True)

# Generate plots
sarsaAgent.plot_reward_history()
sarsaAgent.plot_Q_table()
sarsaAgent.plot_policy()

# Show plots
plt.show()
```

```
Epoch: 1/500 | Reward for epoch: -200
Epoch: 101/500 | Reward for epoch: -2
Epoch: 201/500 | Reward for epoch: 10
Epoch: 301/500 | Reward for epoch: 12
Epoch: 401/500 | Reward for epoch: 13
Epoch: 500/500 | Reward for epoch: 20
```

```
Q Values for Door: {'up': 4.643981201075436, 'down': 0, 'left': 32.545240753983,
'right': 6.62335587469129, 'stay': 0}
```



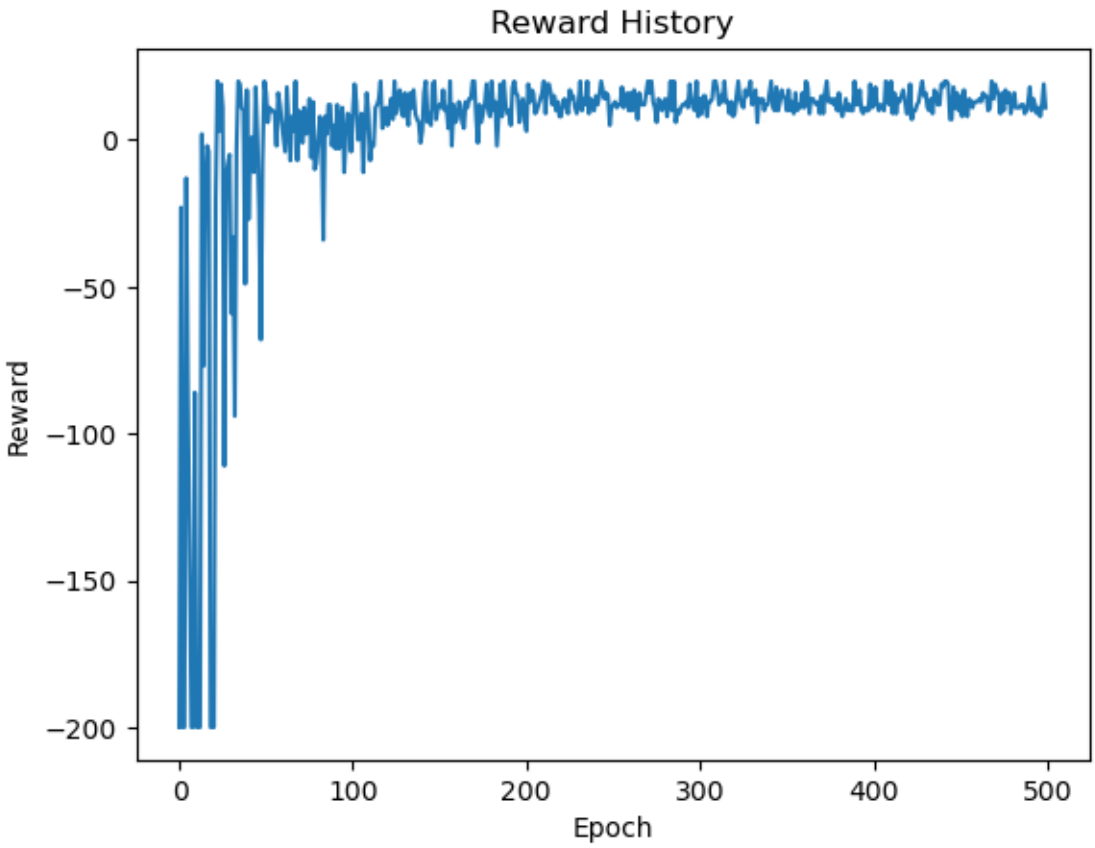


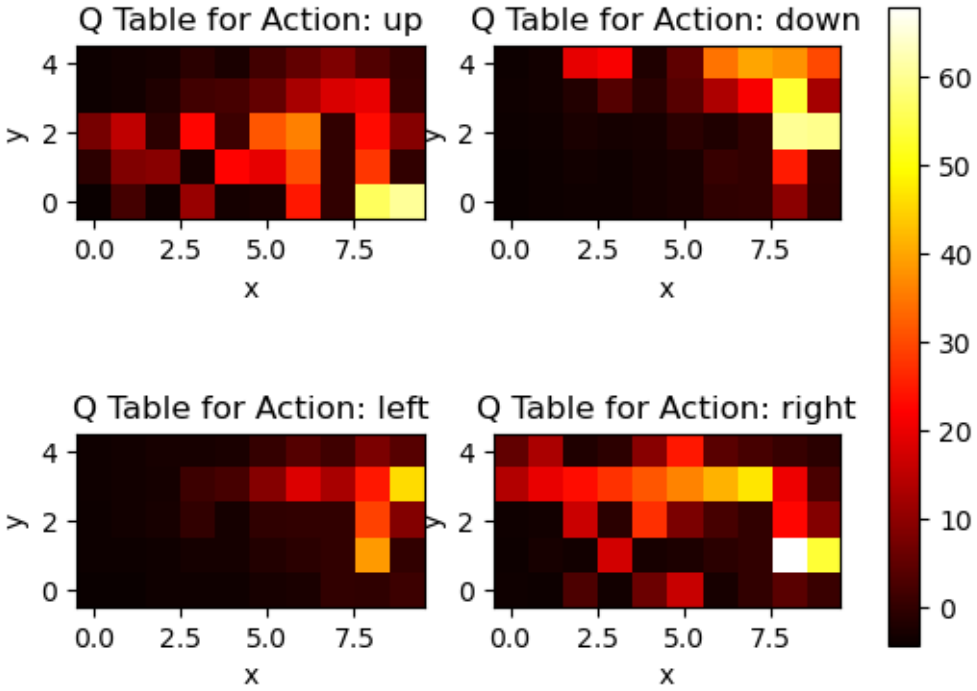


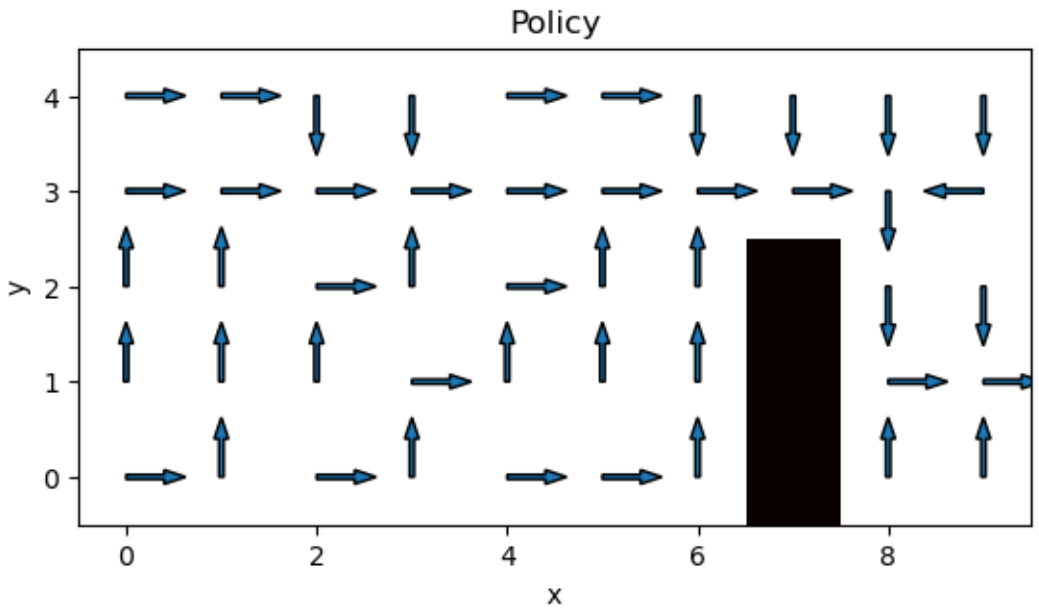
[12]:
```
## Experiments for Q-learning

rng_door = False # random door
```

```
# Initialize learner
qAgent = qLearner(epochs, time_steps, alpha, gamma, epsilon, epsilon_decay,␣
 ↪rng_door)

# Run experiment
qAgent.run(verbose=True)

# Generate plots
qAgent.plot_reward_history()
qAgent.plot_Q_table()
qAgent.plot_policy()

# Show plots
plt.show()
```

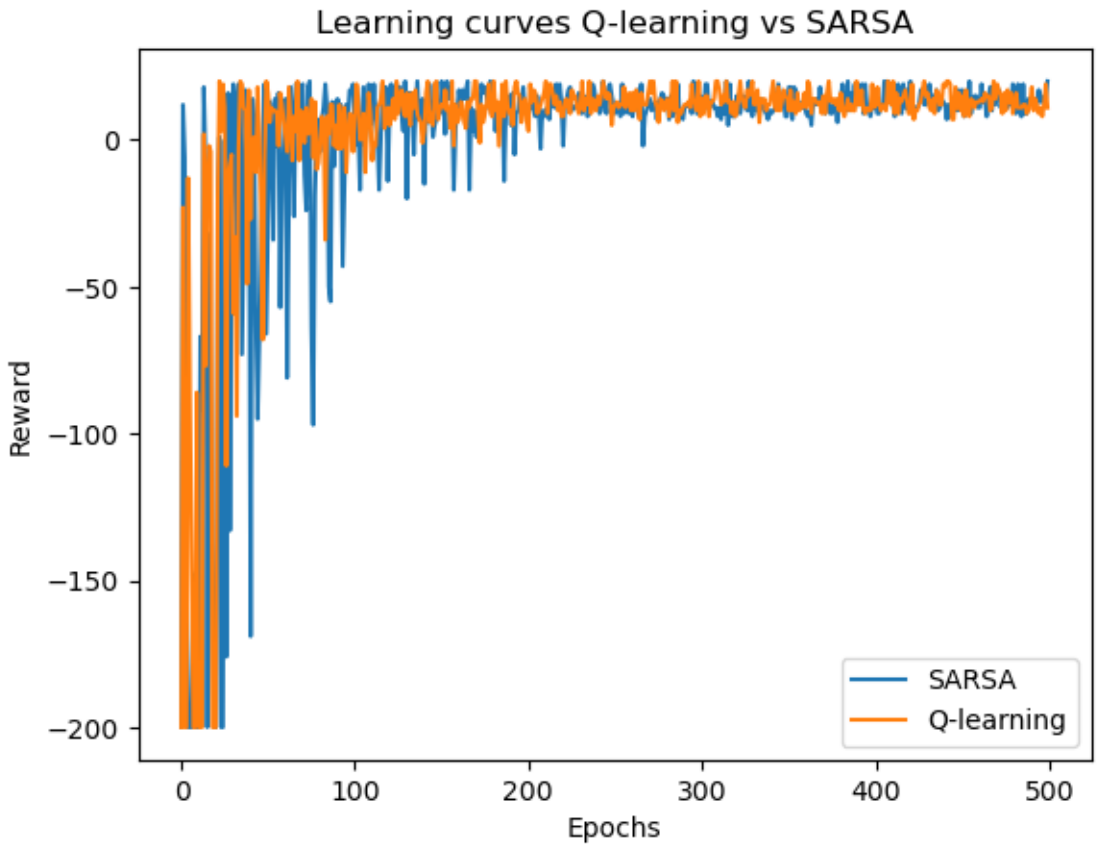
```
Epoch 1/500 | Reward for epoch: -200
Epoch 101/500 | Reward for epoch: 9
Epoch 201/500 | Reward for epoch: 3
Epoch 301/500 | Reward for epoch: 14
Epoch 401/500 | Reward for epoch: 12
Epoch 500/500 | Reward for epoch: 11
Q Values for Door: {'up': 0, 'down': 0, 'left': 0, 'right': 53.25186738754666,
'stay': 5.0}
```


Reward History
Reward
Epoch
0
−50
−100
−150
−200
0
100
200
300
400
500


Q Table for Action: up
Q Table for Action: down
Q Table for Action: left
Q Table for Action: right
x
y
0.0
2.5
5.0
7.5
0
2
4
0
10
20
30
40
50
60



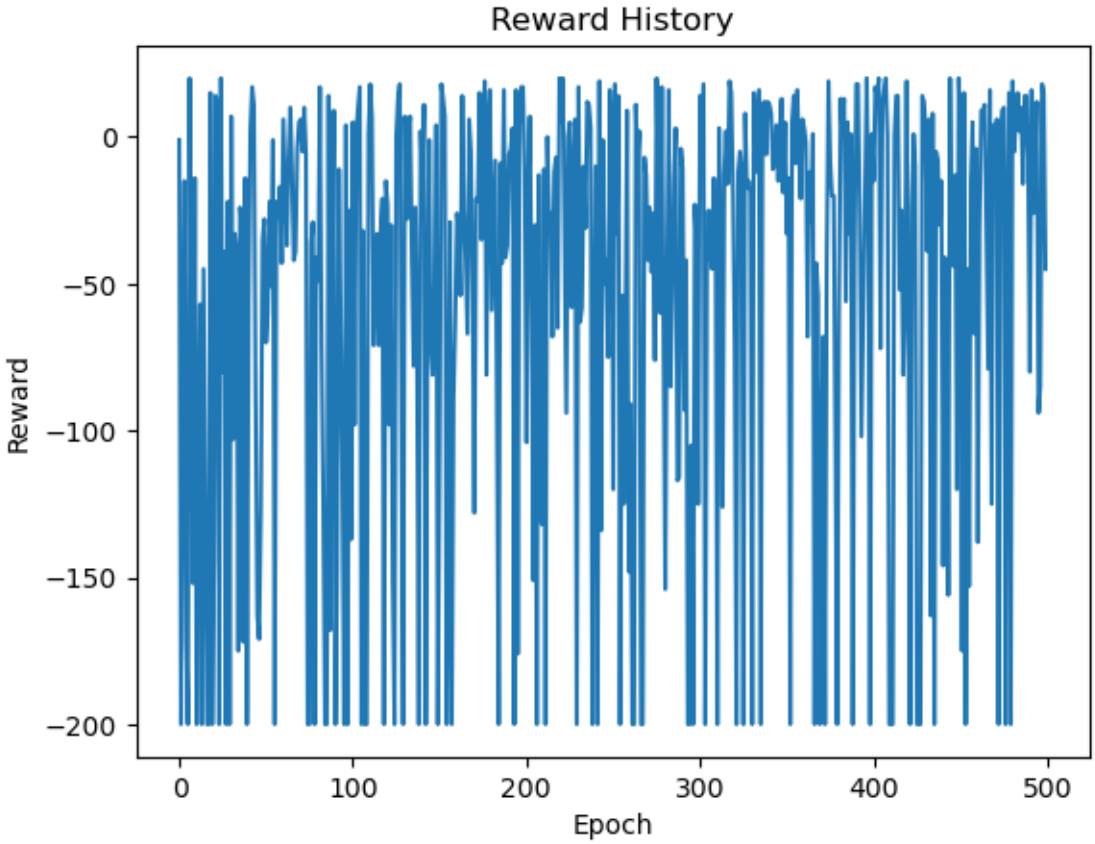
[13]:
```
# Overlay learning curves
plt.figure("Learning curves")
plt.plot(sarsaAgent.reward_history, label="SARSA")
plt.plot(qAgent.reward_history, label="Q-learning")
plt.xlabel("Epochs")
plt.ylabel("Reward")
plt.title("Learning curves Q-learning vs SARSA")
plt.legend()
plt.show()
```

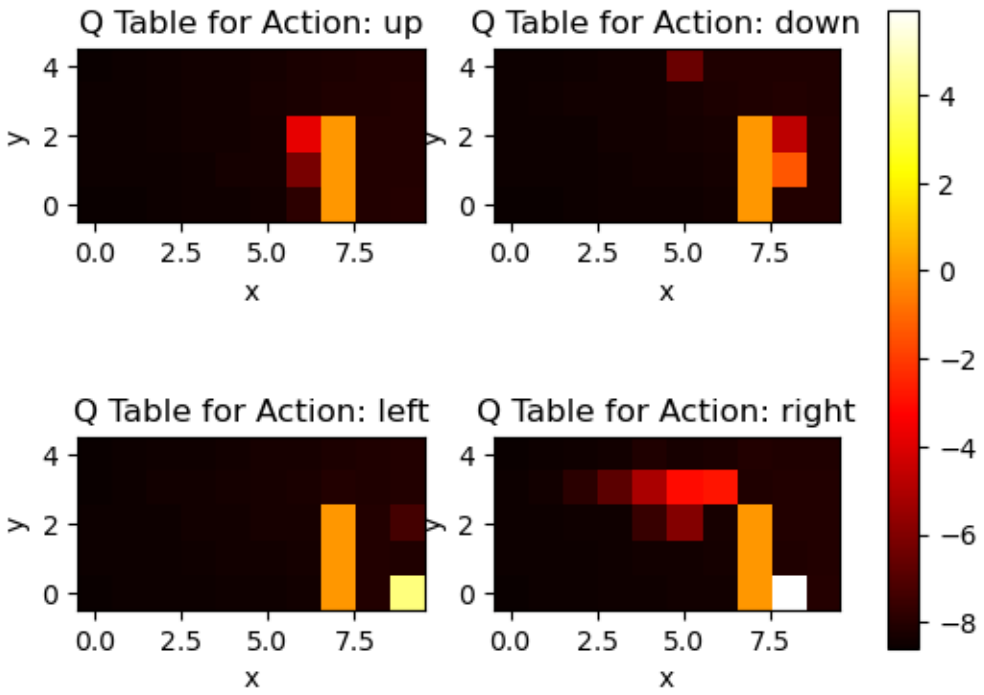


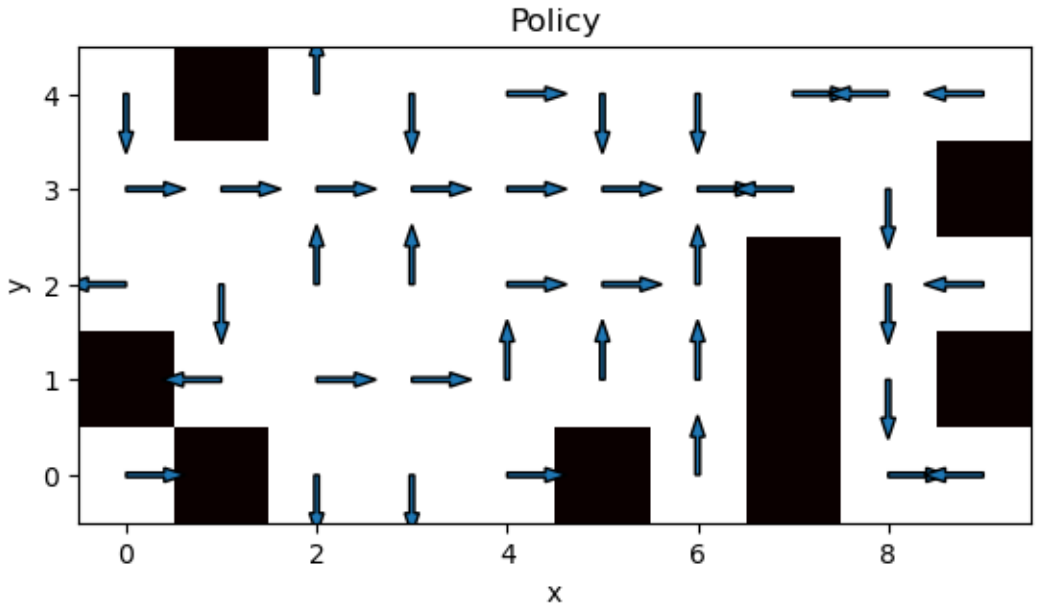
[22]:
```
## Experiments for SARSA learning with random door

rng_door = True # random door

# Initialize learner
sarsaAgent_rng_door = sarsaLearner(epochs, time_steps, alpha, gamma, epsilon,␣
 ↪epsilon_decay, rng_door)

# Run experiment
sarsaAgent_rng_door.run(verbose=True)

# Generate plots
sarsaAgent_rng_door.plot_reward_history()
sarsaAgent_rng_door.plot_Q_table()
sarsaAgent_rng_door.plot_policy()

# Show plots
plt.show()
```

```
Epoch: 1/500 | Reward for epoch: -1
```

```
Epoch: 101/500 | Reward for epoch: 5
Epoch: 201/500 | Reward for epoch: -104
Epoch: 301/500 | Reward for epoch: 14
Epoch: 401/500 | Reward for epoch: -15
Epoch: 500/500 | Reward for epoch: -45
```



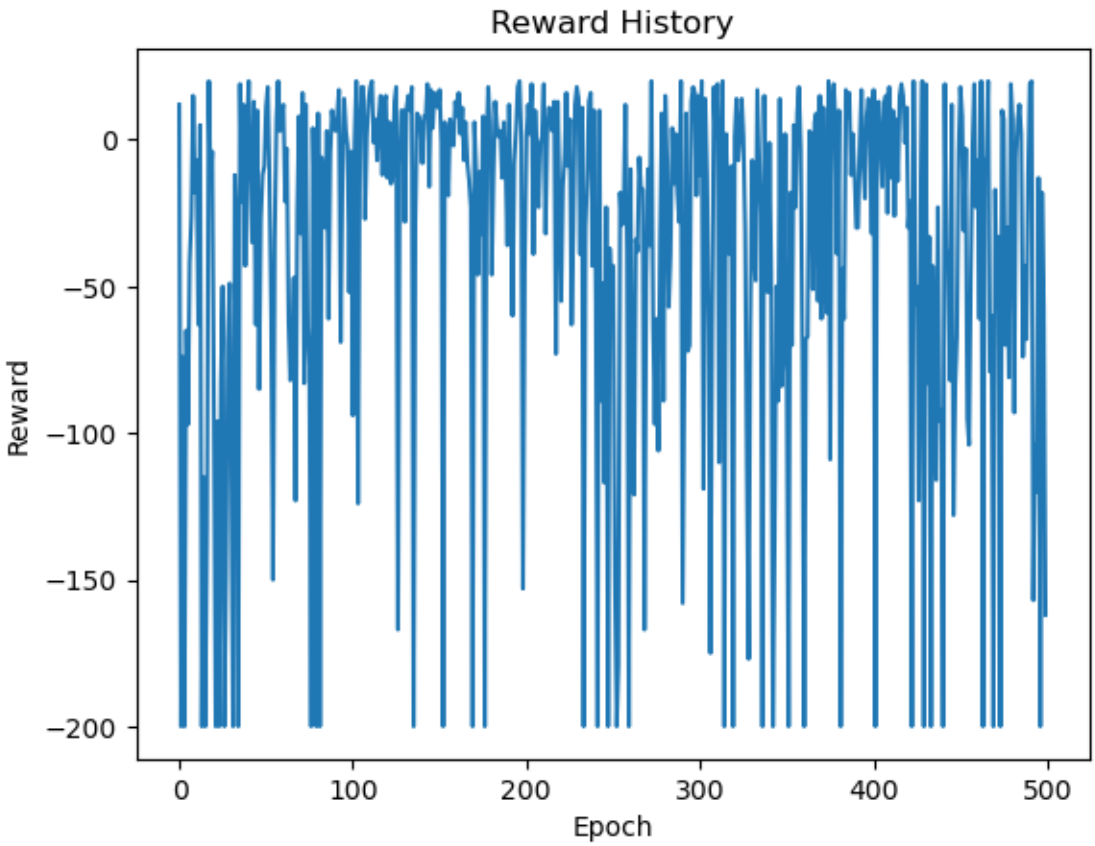


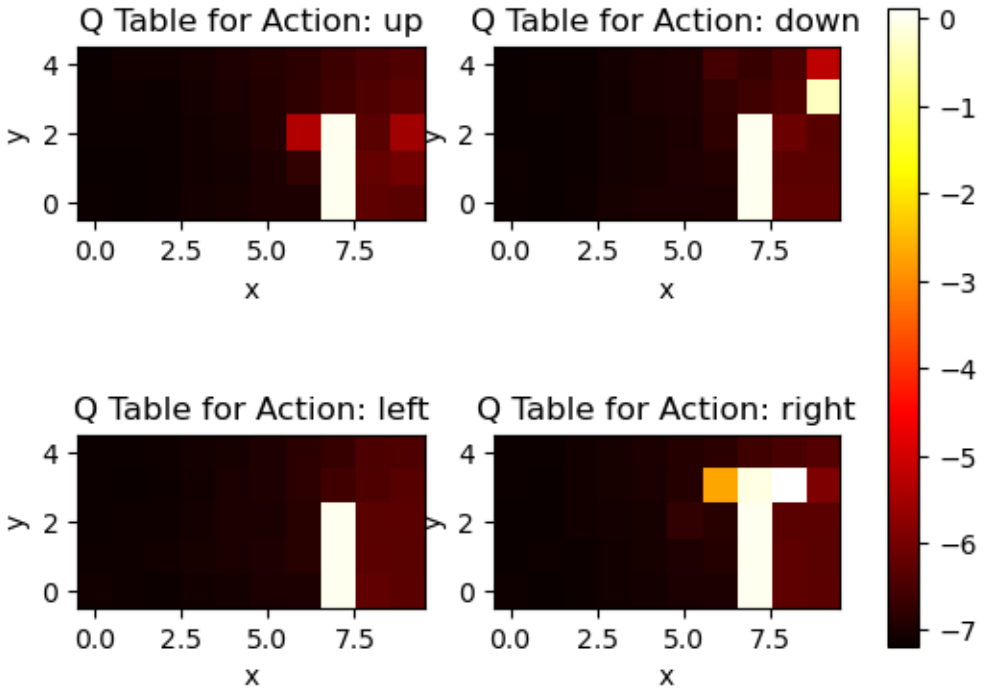


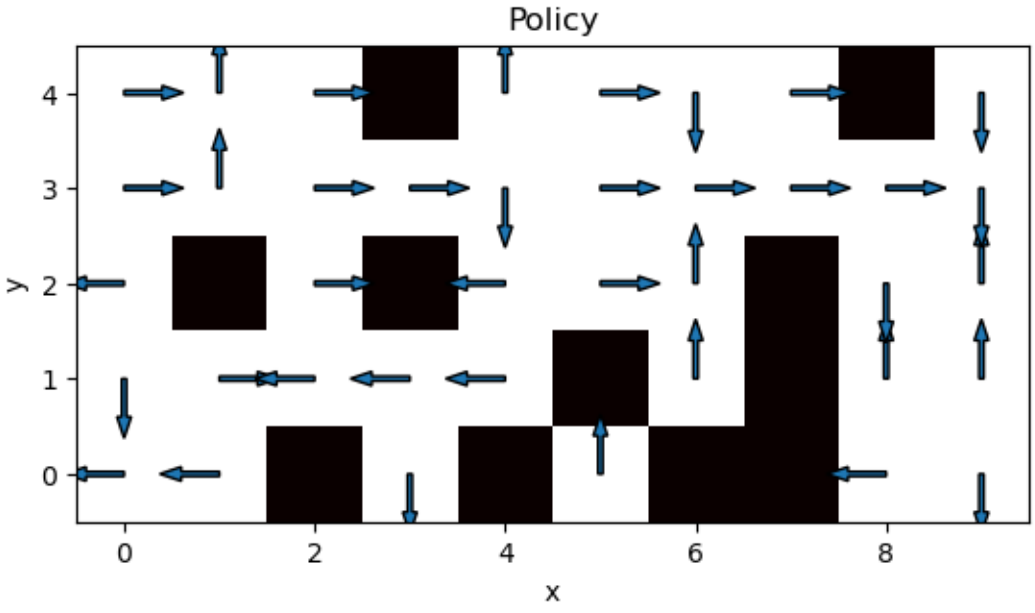
[23]:
```
## Experiments for Q-learning with random door

rng_door = True # random door
```

```
# Initialize learner
qAgent_rng_door = qLearner(epochs, time_steps, alpha, gamma, epsilon,␣
 ↪epsilon_decay, rng_door)

# Run experiment
qAgent_rng_door.run(verbose=True)

# Generate plots
qAgent_rng_door.plot_reward_history()
qAgent_rng_door.plot_Q_table()
qAgent_rng_door.plot_policy()

# Show plots
plt.show()
```

```
Epoch 1/500 | Reward for epoch: 12
Epoch 101/500 | Reward for epoch: -94
Epoch 201/500 | Reward for epoch: 1
Epoch 301/500 | Reward for epoch: -12
Epoch 401/500 | Reward for epoch: 17
Epoch 500/500 | Reward for epoch: -162
```




Q Table for Action: up
Q Table for Action: down
Q Table for Action: left
Q Table for Action: right
y
x
0
2
4
0.0
2.5
5.0
7.5
0
−1
−2
−3
−4
−5
−6
−7


Policy
y
x
0
1
2
3
4
0
2
4
6
8

[24]:
```
# Overlay learning curves
plt.figure("Learning curves")
plt.plot(sarsaAgent_rng_door.reward_history, label="SARSA")
plt.plot(qAgent_rng_door.reward_history, label="Q-learning")
plt.xlabel("Epochs")
plt.ylabel("Reward")
plt.title("Learning curves Q-learning vs SARSA with random door")
plt.legend()
plt.show()
```