Artificial Intelligence - Methods and Applications - 5DV181

Reinforcement Learning (RL)

Shinorina Shahrin Shaon (mai21ssn)

Kenneth Fednand Mukasa (mai21kma)

Multi-armed bandit

1. We worked on MyBandit.py file. Here we added epsilon_min and epsilon_decay with values. We also added code for descresing the epsilon value. Added epsilon_min for comparing with epsilon value. Added epsilon_decay for decresing epsilon value.

```
self.epsilon_min = 0.05

self.epsilon_decay = 0.95


 if self.epsilon > self.epsilon_min:
    self.epsilon *= self.epsilon_decay
```

2. We got result 15. Initially the result was 0. After doing the above mentioned step we got 15.

```
bandit\test_bandits.py:48: AssertionError
========================================================================== warnings summary ==========================================================================
C:\Users\User\anaconda3\lib\site-packages\pyreadline\py3k_compat.py:8
  C:\Users\User\anaconda3\lib\site-packages\pyreadline\py3k_compat.py:8: DeprecationWarning: Using or importing the ABCs from 'collections' instead of from 'collectio
ns.abc' is deprecated since Python 3.3, and in 3.9 it will stop working
    return isinstance(x, collections.Callable)

-- Docs: https://docs.pytest.org/en/stable/warnings.html
====================================================================== short test summary info =======================================================================
FAILED bandit/test_bandits.py::test_performance - assert 15 > 15
================================================================== 1 failed, 1 warning in 0.81s ==================================================================
PS G:\Umea\AI-Model and Application\RL\5dv181ht21\RL>
```

3. In future we can improve this by using another device other than my present device. The result will improve more by simplifying the run(). If we work more and give more time on run() than the result will improve.

Pong

1. We worked on Agent.py file. Here we discretized the observation. For this we took help. We changed the values of epsilon, alpha. We determined the action probabilities on determine_action_probabilities(). Here we returned the q-table. We added agents' action using random function. Actions depeneded on epsilon value.
   On our worked we did not use min_epsilon and epsilon_decay.

   a. epsilon=0.1,

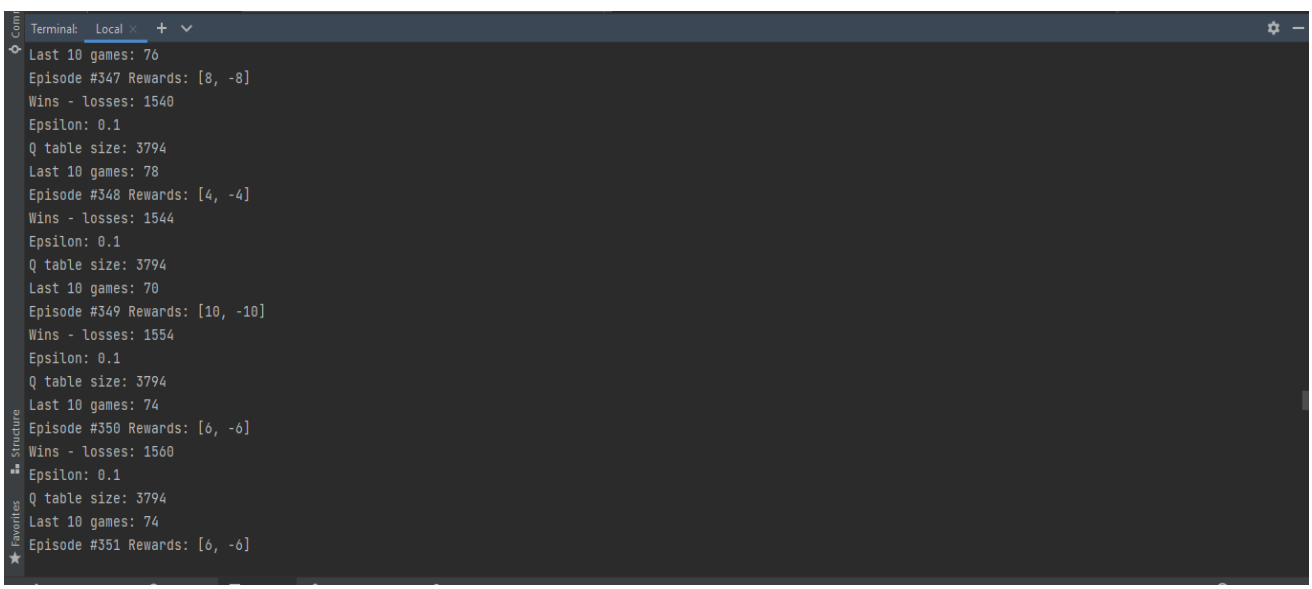
   b. def determine_action_probabilities(self, observation):

      ```
      best_action = reshape_obs(observation)
      return self.q[best_action]
      ```

   c. def act(self, observation):

      ```
      if random.random() < self.epsilon:
          return random.randint(0,2)
      else:
          return numpy.argmax(self.determine_action_probabilities(observation))
      ```

2. At the point of Episode 350 we got more than 1000. The value was 1560 where the epsilon was 0.1



3. We can improve the result by using min_epsilon and epsilon_decay. We can improve more by comparing min_epsilon with epsilon and decrease the epsilon value by using epsilon_decay. If we use observation step and put numerical value for observation then the result will more better.

References:

1. https://github.com/abdulqadirs/atari-pong-reinforcement-learning/tree/master/pong
2. https://www.geeksforgeeks.org/q-learning-in-python/
3. https://prutor.ai/q-learning-in-python/
4. https://github.com/tomkimsour/Reinforcement_Learning/search?l=Python