ε-greedy 算法实验报告

人工智能 91  卢佳源  2191121196

一、实验目的：

a) 理解ε-贪心算法的原理和实现过程；
b) 编写ε-贪心算法程序；
c) 改变超参数ε的大小，比较算法性能的不同；

二、实验环境：

a) IDE：VSCode，Python-3.9.7
b) 编程语言：Python；
c) 文件路径：C:\Users\jiayuan lu\OneDrive - MSRA\桌面\大三下\RL\作业 1 ε_greedy\ε.py

三、实验原理和思路：

a) 自定义给出 10 个动作（10 臂赌博机），每个动作对应的奖励 reward 和概率，迭代次数 time，将ε取 0，0.01，和 0.1 到 0.9 的等间距变化；
b) 初始化每个动作对应的价值函数 Q，以及每个动作被选择执行的次数 N；
c) 对每一个ε，进行 time 次迭代，计算每次迭代的平均累积奖励：
   i. 以ε的概率随机选择一个动作 A（试探），以 1-ε的概率选择之前计算的动作价值函数 Q 的最大值对应的动作赋给 A（利用）；
   ii. 利用 bandit 算法计算 i 中选择的动作 A 对应的奖励 R，并计算累积奖励的平均值；
   iii. 将动作 A 的执行次数 N 加上 1；
   iv. 将动作 A 的动作价值函数 Q 按照如下公式更新：

$$\mathrm{Q(A)} = \mathrm{Q(A)} + \frac{1}{N(A)}[\mathrm{R} - \mathrm{Q(A)}]$$

d) 对每一个ε，画出平均累积奖励和迭代次数的曲线，比较曲线之间的差异。

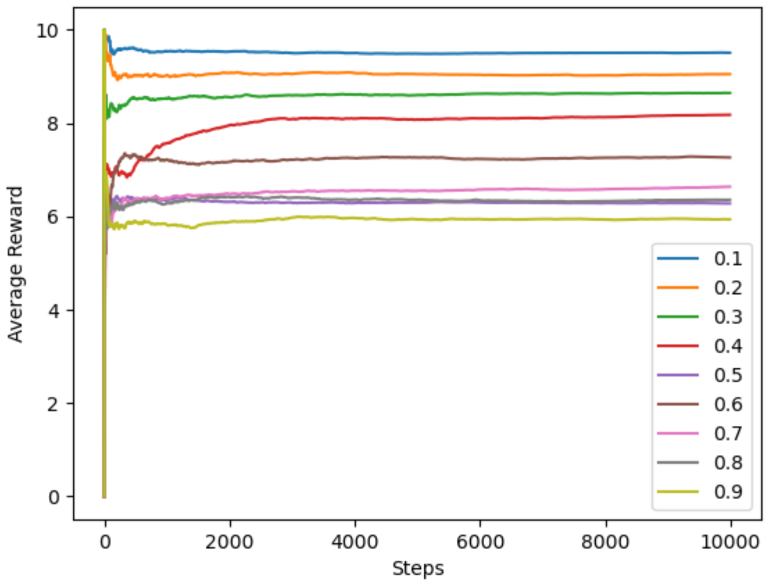
四、实验代码：

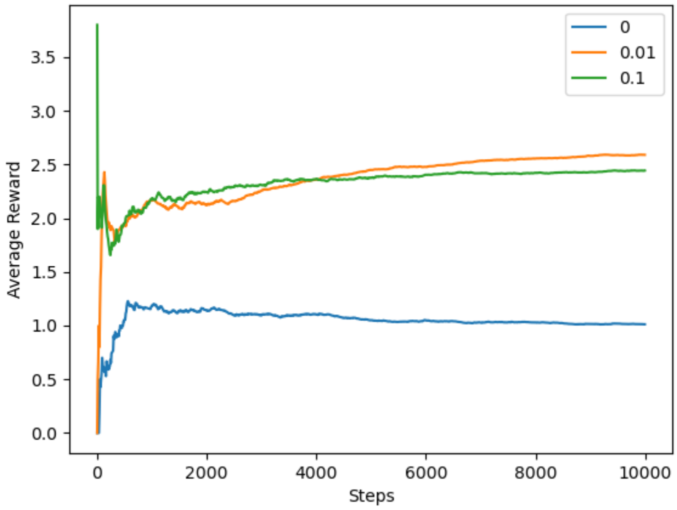
```python
import numpy as np
import matplotlib.pyplot as plt

action=[1,2,3,4,5,6,7,8,9,10]
reward={}
prob={}
reward={1:10,2:1,3:4,4:6,5:8,6:7,7:3,8:2,9:9,10:5}
prob={1:0.1,2:0.8,3:0.6,4:0.5,5:0.3,6:0.35,7:0.7,8:0.75,9:0.15,10:0.55}
epsilon=[0.1,0.2,0.3,0.4,0.5,0.6,0.7,0.8,0.9]
# epsilon=[0,0.01,0.1]
times=10000
num={}
x=[]
y=[]
x=[[0]*len(epsilon) for i in range(int(times/10))]
y=[[0]*len(epsilon) for i in range(int(times/10))]
Q={}
```

```
for e in range(len(epsilon)):
    R=0
    avg_R=0
    for i in reward.keys():
        Q[i]=0
    for i in reward.keys():
        num[i]=0
    for i in range(times):
        if np.random.random()<epsilon[e]:
            A=np.random.choice(action)
        else:
            A=max(Q,key=Q.get)
        R=R+(reward[A]-R)/(i+1)
        num[A]+=1
        Q[A]+=(R-Q[A])/num[A]
        if (i%10)==0:
            x[e].append(i)
            y[e].append(R)
    plt.plot(x[e],y[e])
plt.xlabel("Steps")
plt.ylabel("Average Reward")
plt.legend(['0.1','0.2','0.3','0.4','0.5','0.6','0.7','0.8','0.9'])
# plt.legend(['0','0.01','0.1'])
plt.show()
```

五、实验结论:

a) 实验结果:






b) 实验结果分析:

i. ε代表选择试探（随机选择动作 A）的概率，因此ε越大，该贪心算法越敢尝试新的动作，不固守在已经执行过并得到了动作价值函数的动作上，使得算法更有机会得到更高的回报，但是也冒着更多的风险——新探索的动作的价值回报可能没有已经尝试过的动作的价值回报高，使得算法进行了一定量的无用功；

ii. 从上图可以看出:

1. ε=0，即单纯的贪心算法，得到的最终期望收益是最小的，因为它没有探索的过程；
2. 随着ε的增大，期望收益先增大后减小，并且图中的曲线在刚开始的阶段出现了峰值，分析其原因，可能是因为前几步算法都是以探索新动作为主，每个动作基本上都探索到了，并且刚开始的时候每个动作的执行次数较少，因此期望收益在开始阶段出现峰值，随着时间的增加，期望收益会下降并趋于平稳；
3. ε并不是越大越有利于算法性能的提升，要考虑利用和探索的折衷，从上图可以看出，ε取 0.1 时，算法性能最优，得到最高的最终期望收益。

六、实验反思:

a) ε对贪心算法性能的影响主要有两个：峰值地出现和最终期望收益；

b) ε对于上述两个方面的影响，我认为是这样的:

i. 收敛速度：ε越大，越快地探索到所有动作，峰值出现地越早，也越高；

ii. 最终期望收益：ε取折衷值时，即同时考虑到利用和探索带来地回报，会得到更高地最终期望收益，但是都比单纯地贪心算法（ε=0）时得到地最终期望收益要高，因为ε不为 0 时，可以有机会去尝试新的动作，也就有机会得到更高地回报。