Reinforcement Learning Lab

Lesson 2: Multi-Armed Bandit

Gabriele Roncolato and Alberto Castellini

University of Verona
email: gabriele.roncolato@univr.it

Academic Year 2024-25

Environment Setup

The first step for the setup of the laboratory environment is to update the repository and load the miniconda environment.

Safe Procedure

Always back up the previous lessons' solutions before executing the repository update.

- Update the repository of the lab:

```
cd RL-Lab
git stash
git pull
git stash pop
```

- Activate the *miniconda* environment:

```
conda activate rl-lab
```

Today Assignment

In today's lesson, we will implement the Multi-Armed Bandit Environment and the Simple Bandit Algorithm algorithm to solve it. In particular, the file to complete is:

```
RL-Lab/lessons/lesson_2_code.py
```

Inside the file, a python class and a function are partially implemented. The objective of this lesson is to complete it.

- **class MultiArmedBandit()**
- **def banditAlgorithm()**

Expected results can be found in:

```
RL-Lab/results/lesson_2_results.txt
```

Environment: Multi-Armed Testbed

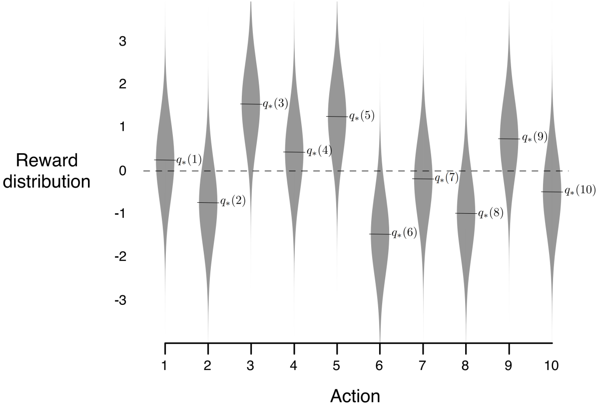

Figure: Visual explanation of the Multi-Armed Testbed environment, from the Sutton and Barto book *Reinforcement Learning: An Introduction*

- The *Multi-Armed Testbed* environment consists of a set of N possible actions, from 1 to N. A mean value ($q^*(a)$) has been assigned to each action, sampled from a normal distribution with $\mu = 0$ and $\sigma^2 = 1$.
- For a given action a, the environment should return a reward sampled from a normal distribution with $\mu = q^*(a)$ and $\sigma^2 = 1$.

Algorithm: Simple Bandit

A simple bandit algorithm

Initialize, for $a = 1$ to k:

$Q(a) \leftarrow 0$

$N(a) \leftarrow 0$

Loop forever:

$$A \leftarrow \begin{cases} \arg\max_a Q(a) & \text{with probability } 1-\varepsilon \quad \text{(breaking ties randomly)} \\ \text{a random action} & \text{with probability } \varepsilon \end{cases}$$

$R \leftarrow bandit(A)$

$N(A) \leftarrow N(A) + 1$

$Q(A) \leftarrow Q(A) + \frac{1}{N(A)}\left[R - Q(A)\right]$

Figure: Pseudocode for Simple Bandit Algorithm, from the Sutton and Barto book *Reinforcement Learning: An Introduction*

Simple Bandit Algorithm applied to 10-Armed Testbed

The suggested solution exploits a NumPy function to sample from a normal distribution, numpy.random.normal(). More details can be found on the official website (here).

Seeding

Given the (particularly) high stochasticity of the method and the environment, for this lesson, we fixed a random seed equal to 6.

Hint (Expected results)

The plot on the right is the expected result. Notice that the best results have been obtained with eps=0.1, while the worst one with eps=0 (i.e., no exploration).