Multi-Armed Bandit Problems

Recitation: greedy and ϵ-greedy policy, UCB

Ayoub Ajarra

21 de noviembre de 2024

Reminder: RL setting

Key characteristics of an RL problem:

- Learning to take action in many situations.
- Delayed reward/credit assignment.
- Exploration/Exploitation trade-off.

Difference with bandit setting (immediate RL)

- Agent sees the same state all the time.
- Rewards are immediate.
- Exploration/Exploitation trade-off.

Types of feedback

- Instructive feedback:
  1. Instructs the right action a^*.
  2. Ignores the action taken.
  3. Used in supervised learning.
- Evaluative feedback:
  1. Evaluates the action taken A_t by giving some reward.
  2. Completely depends on the action taken.
  3. Used in RL.

Example: K-armed bandits

Assumption: Rewards are chosen from stationary probability distributions that depend on the action taken.

Goal: Maximize total reward over some period of time.

Value of an action

- Actual value of action a (Ground-truth):

$$q^*(a) = \mathbb{E}\{R_t | A_t = a\}$$

- Always pick action $a^* = \arg\max_{a \in \mathcal{A}} q^*(a)$

Value of an action

- Actual value of action a (Ground-truth):

$$q^*(a) = \mathbb{E}\{R_t | A_t = a\}$$

- Always pick action $a^* = \arg\max_{a \in \mathcal{A}} q^*(a)$
- $q^*(a)$ is unknown to the agent.

Value of an action

- Actual value of action a (Ground-truth):

$$q^*(a) = \mathbb{E}\{R_t | A_t = a\}$$

- Always pick action $a^* = \arg\max_{a \in \mathcal{A}} q^*(a)$
- $q^*(a)$ is unknown to the agent.
- What the agent can access: Q_t the estimate of the value function $q^*(a)$ at timestep t
- Find the best action as quickly as possible:

$$A_t = \arg\max_{a \in \mathcal{A}} Q_t(a)$$

Regret Vs Reward

Regret is the amount of reward the agent has lost because of the learning process (selected policy)

- If the optimal action was known:

$$\text{Regret} = \text{K}q^*(a^*)$$

Regret Vs Reward

Regret is the amount of reward the agent has lost because of the learning process (selected policy)

- If the optimal action was known:

$$\mathrm{Regret} = \mathrm{K}q^*(a^*)$$

- In reality, the optimal action is unknown:

$$\mathrm{Regret} = \mathrm{K}q^*(a^*) - \sum_{t=1}^{K} R_t$$

Action-value methods: Estimation of Q_t

$$Q_t(a) = \frac{\text{Sum of rewards when action a taken prior to time t}}{\text{Number of times action a taken prior to time t}}$$

$$= \frac{\sum_{i=1}^{t-1} R_i 1_{A_i=a}}{\sum_{i=1}^{t-1} 1_{A_i=a}}$$

How to solve the bandit problem ?

- Only exploit (Greedy):

$$A_t = \arg\max_{a \in \mathcal{A}} Q_t(a)$$

- Not enough for learning.
- Why?

centralelille
ÉCOLE CENTRALE DE LILLE

ϵ-greedy algorithm

A possible solution

centralelille
ÉCOLE CENTRALE DE LILLE

ϵ-greedy actions

General idea: Take greedy action, and once in a while take ϵ-greedy action.

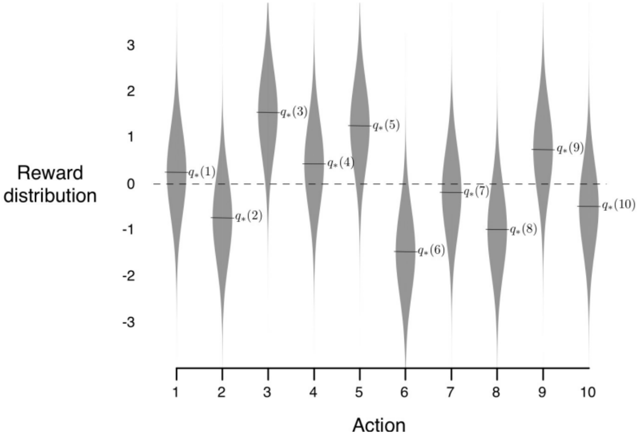
$$A_t = \begin{cases} \underset{a \in \mathcal{A}}{\operatorname{argmax}} Q_t(a) & \text{with probability } 1 - \epsilon, \\ \text{Random action from } \mathcal{A} & \text{with probability } \epsilon. \end{cases}$$

Advantage: In the limit, every action will be sampled an infinite number of times, thus ensuring that $Q_t(a)$ converges to $q^*(a)$

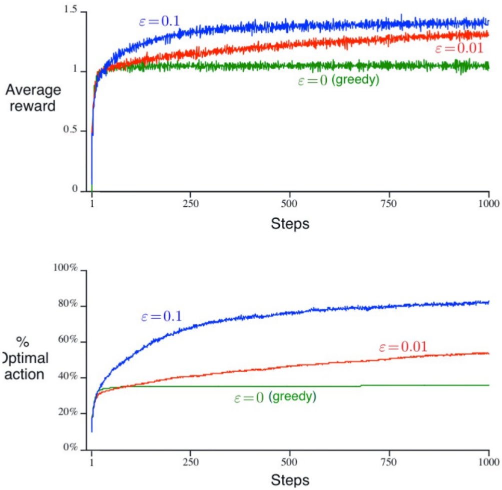
Example 10-armed bandits

2000 randomly generated K-armed bandits with K fixed to 10, q^* are selected according to a gaussian distribution with mean 0 and variance 1. When the action is taken the reward is sampled from a Gaussian distribution of mean $q^*(A_t)$ and variance 1.

Example 10-armed bandits



Comparaison between greedy and ϵ-greedy



Implementation of ϵ-greedy

- R_i denote the reward received after the i^{th} selection of this action.
- Let Q_n denote the estimate of its action value after it has been selected $n-1$ times.

$$Q_n = \frac{R_1 + R_2 + \cdots + R_{n-1}}{n-1}$$

- Naive implementation: store all rewards and compute the average every time.
- Memory constraints.

Implementation of ϵ-greedy

- R_i denote the reward received after the i^{th} selection of this action.
- Let Q_n denote the estimate of its action value after it has been selected $n-1$ times.

$$Q_n = \frac{R_1 + R_2 + \cdots + R_{n-1}}{n-1}$$

- Naive implementation: store all rewards and compute the average every time.
- Memory constraints.
- Can we do better?

Incremental implementation

$$
\begin{aligned}
Q_{n+1} &= \frac{1}{n}\sum_{i=1}^{n} R_i \\
&= \frac{1}{n}\left(R_n + \frac{n-1}{n-1}\sum_{i=1}^{n-1} R_i\right) \\
&= \frac{1}{n}\left(R_n + (n-1)Q_n\right) \\
&= Q_n + \frac{1}{n}\left(R_n - Q_n\right)
\end{aligned}
$$

$$
\text{New Estimate} = \text{Old Estimate} + \text{Step size}[\text{Target} - \text{Old estimate}]
$$

Pseudo-code

A simple bandit algorithm

Initialize, for $a = 1$ to k:

$Q(a) \leftarrow 0$

$N(a) \leftarrow 0$

Loop forever:

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

$R \leftarrow bandit(A)$

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

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

What about non-stationary rewards ?

- It makes sense to give more weight to recent rewards than long-past rewards.
- One easy way to do that is by having a constant step size parameter:

$$Q_{n+1} = Q_n + \alpha(R_n - Q_n)$$

What about non-stationary rewards ?

$$\begin{aligned} Q_{n+1} &= Q_n + \alpha(R_n - Q_n) \\ &= \alpha R_n + (1-\alpha)Q_n \\ &= \alpha R_n + (1-\alpha)(\alpha R_{n-1} + (1-\alpha)Q_{n-1}) \\ &= (1-\alpha)^n Q_1 + \sum_{i=1}^{n} \alpha(1-\alpha)^{n-i} R_i \end{aligned}$$

This is weighted average because:

$$(1-\alpha)^n + \sum_{i=1}^{n} \alpha(1-\alpha)^{n-i} = 1$$

What about non-stationary rewards ?

- Let $\alpha_n(a)$ denote stepsize parameter used to process the reward received after the n^{th} selection of action a.
- $\alpha_n(a) = \frac{1}{n}$ leads to sample average method.
- Note: Convergence to the values is not guaranteed for all choices of $\alpha_n(a)$
- Conditions required to assure convergence with probability 1:
  1. guarantees that the steps are large enough to eventually overcome any initial conditions or random fluctuations.

$$\sum_{i=1}^{\infty} \alpha_n(a) = \infty$$

  2. guarantees that the steps eventually become small enough to assure convergence.

$$\sum_{i=1}^{\infty} \alpha_n(a)^2 > \infty$$

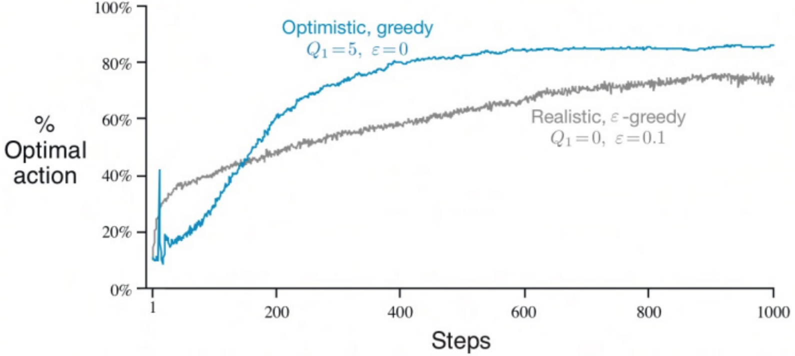
Stationary Vs Non-stationary settings

Both conditions are met for $\alpha_{\mathrm{n}}(\mathrm{a}) = \frac{1}{\mathrm{n}}$. But for $\alpha_{\mathrm{n}}(\mathrm{a}) = \alpha$, the second condition is not met.

Optimism

All these methods are dependent on the initial action - value estimates, $Q_1(a)$.
They are biased by their initial estimates.

Optimism



100%
80%
60%
40%
20%
0%
% Optimal action
Optimistic, greedy
$Q_1 = 5, \ \varepsilon = 0$
Realistic, ε-greedy
$Q_1 = 0, \ \varepsilon = 0.1$
1
200
400
600
800
1000
Steps

Hands-On Session: ϵ- greedy

Lets look at a demo.

Upper Confidence Bound Algorithm

A more plausible solution

Reminder

On board.

Hands-On Session: UCB

Lets look at a demo.

Questions ?