Introduction to Reinforcement Learning

多臂老虎机

何新卫

信息学院,
华中农业大学

2024 年 3 月 4 日

Part I

Chapter2: K 臂老虎机问题（Multi-armed Bandits）

案例

图. 抓娃娃机

喜欢哪个娃娃，投个币就有机会得到它。

案例

图. 两台娃娃机

设想一下，有两台娃娃机，投入币，娃娃机即可以一定概率输出娃娃。

- 假如我告诉你左边的机器以概率为 0.3 的概率输出娃娃，右边机器以 0.4 的概率输出娃娃，你会怎么玩呢？
- 假如不告诉你奖励概率，但是给你 100 枚游戏币，你会怎么玩呢？完全随机的，还是逮着其中一台机器玩 10 次？有更好的策略吗？

概述

- 多臂老虎机问题类似暗黑版夹娃娃器，它可以被看作简化版的强化学习问题。
- 我们借 K 臂老虎机问题问题介绍了一些基本学习方法，后续章节中将扩展应用于完整的强化学习问题。

K 臂老虎机问题

- 通用问题：K 个不同选项或动作之间的选择, 每个选择对应固定概率分布中选择的数值奖励，目标：在某个时间段内最大化期望总奖励，例如，在 1000 次动作选择或时间步骤中
- K 臂老虎机问题: K 个手杆，每次拉动老虎机手杆之一，奖励就是击中大奖的支付。通过重复的行动选择，您要通过将行动集中在最佳手杆上来最大化您的赢利。
- 医生在为一系列重病患者选择实验性治疗方法之间做出选择。每个行动就是选择一种治疗方法，每个奖励是患者的存活或健康状况
- K 个股票选择问题
- 其它产品推荐等等。

在 k 臂老虎机问题中，每个动作都有一个 expected 或 mean reward，即在选择该动作时的 value。我们将时间步骤 t 上选择的行动表示为 A_t，相应的奖励表示为 R_t。然后，任意动作 a 的值，标记为 $q_*(a)$，是给定选择 a 时的期望奖励:

$$q_*(a) \doteq \mathbb{E}[R_t | A_t = a]$$

如果我们知道每个动作的 value，那么多臂老虎机的最优策略：总是选择 value 最高的动作即可

多臂老虎机中的探索与利用（exploration vs. exploitation）问题一直以来都是一个特别经典的问题，理解它能够帮助我们学习强化学习。

问题定义

多臂老虎机（multi-armed bandit，MAB）问题（见下图）：有一个拥有 K 根拉杆的老虎机，拉动每一根拉杆都对应一个关于奖励的概率分布 R。我们每次拉动其中一根拉杆，就可以从该拉杆对应的奖励概率分布中获得一个奖励 r。我们在**各根拉杆的奖励概率分布未知**的情况下，从头开始尝试，目标是在操作 T 次拉杆后获得尽可能高的累积奖励。
由于奖励的概率分布是未知的，因此我们需要在“探索拉杆的获奖概率”和“根据经验选择获奖最多的拉杆”中进行权衡。“采用怎样的操作策略才能使获得的累积奖励最高”便是多臂老虎机问题。如果是你，会怎么做呢？

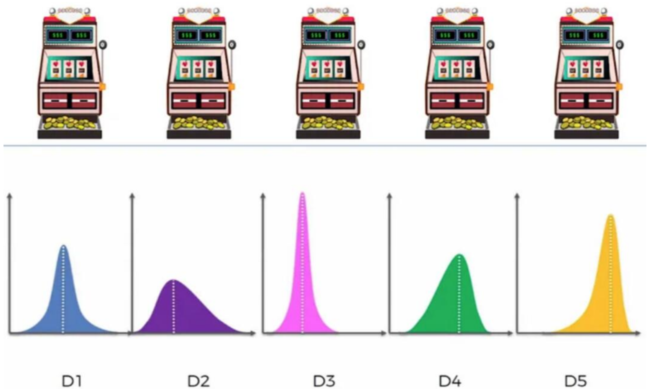

图. 多臂老虎机（multi-armed bandit，MAB）问题。各根拉杆的奖励概率分布未知

K 臂老虎机形式化描述

- 智能体：?
- 环境：?
- 动作：?
- 状态：?
- 动作空间大小：?
- 状态空间：?
- 奖励：?
- 学习的目标

K 臂老虎机形式化描述

- 智能体：我们的算法
- 环境：k 臂老虎机
- 动作：选择哪一个老虎机臂。由于只有一个状态，我们做出的选择并不会对以后发生的事情造成影响。
- 状态：做出动作后，老虎机在那里不会自己发生改变（所有老虎机奖励的概率分布都是确定的，不会随时间发生改变，不会随我们做出的选择 action 发生改变。
- 动作空间大小？$A = \{a_1, a_2, ..., a_K\}$
- 状态空间大小？1，只有一个状态。老虎机在那里不会自己发生改变（所有老虎机奖励的概率分布都是确定的，不会随时间发生改变，不会随我们做出的选择 action 发生改变
- 奖励：老虎机的返回奖励。注意在多臂老虎机问题中是没有延迟奖励问题的，行动得到的奖励是即时的，且由于只有一个状态，得到的奖励也不会随 action 发生改变。
- 学习的目标：在操作 T 次拉杆后获得尽可能高的累积奖励:

$$max \sum_{t=1}^{T} r_t, r_t \sim R(\cdot|a_t)$$

其中 a_t 表示在第 t 时间步拉动某一拉杆的动作，r_t 表示动作获得的奖励

评估动作价值

我们并不知道每个动作的 value, 那么只能进行估计:

- 对于动作 a, 我们记其真实价值为 $q_*(a)$, 第 t 次动作时，估计的价值为 $Q_t(a)$
- 通过对实际收到的奖励进行平均进行估计。假设 t 次动作前，动作 a 被选择了 k_a 次，分别获得奖励为 $r_1, r_2, ..., r_{k_a}$, 则 $Q_t(a)$ 的估计价值为:

$$Q_t(a) \doteq = \frac{\sum_{i=1}^{t-1} R_i \cdot \mathbb{1}_{A_i=a}}{\sum_{i=1}^{t-1} \mathbb{1}_{A_i=a}}$$

其中，如果 $k_a = 0$，则我们定义 $Q_t(a)$ 为 0；随着 $k_a \xrightarrow{\infty}$, 根据大数定理，$Q_t(a)$ 将收敛到 $q_*(a)$。

- 选择动作的贪心策略：我们选择具有最大 estimated value 的动作:

$$A_t \doteq \mathrm{argmax}_a Q_t(a)$$

- 贪婪的动作选择总是利用当前知识来最大化即时奖励；它完全不花时间去采样开起来较差的动作，以查看它们是否可能真的更好。一个简单的替代方法是大部分时间都贪婪地动作，但偶尔以较小的概率 ϵ 随机从动作空间中等概率的随机选择一个动作。该方法被称为 $\epsilon - greedy$ 算法。

作业 2

问题: $\epsilon - greedy$ 算法，当前最佳动作被选择的概率有多大?

答案

$\left(1-\epsilon+\epsilon/|A|\right)$，其中 A 为动作空间。

作业 2

问题：假设动作空间有两个动作 $\{a_1, a_2\}$, 使用 $\epsilon = 0.5$ 的贪心贪心策略。若某个状态，动作 a_1 是最佳动作 (即最大的 Q 值)，择其选中的概率有多大？

答案

答案：a_2 被选中的概率有多大？只有 $0.5/2 = 0.25$, 因此 a_1 被选中的概率是 $1 - 0.5 + 0.5/2 = 0.75$。

计算动作平均奖励的公式

记 R_i 为第 i 次选择动作 a 时获取的奖励，Q_n 代表已经选择动作 a 已经执行了 $n-1$ 次的奖励，则容易计算第 n 次操作时要选择动作 a 的奖励估计奖励均值为：

$$Q_n \doteq \frac{R_1 + R_2 + ... + R_{n-1}}{n-1} \tag{1}$$

显而易见的实现方式是保留所有奖励记录，然后在需要估计值时执行这个计算。
然而，如果这样做，随着看到更多奖励，内存和计算需求将随时间增长。每个额外的奖励都需要额外的内存来存储它，并需要额外的计算来计算分子中的总和。

练习：计算动作平均奖励公式

考虑一个 4 臂老虎机问题，有 $K = 4$ 个动作，分别标记为 1，2，3，4。我们使用 ϵ-greedy 算法进行动作选择。每个动作，我们初始花 Q 值为 0，即 $Q_1(a) = 0$, for all a。假设我们获得了一个动作和奖励的序列，如下：

$$A_1 = 1, R_1 = -1, A_2 = 2, R_2 = 1, A_3 = 2, R_3 = -2, A_4 = 2, R_4 = 2, A_5 = 3, R_5 = 0$$

计算交互过程中动作平均奖励？

练习：计算动作平均奖励公式

考虑一个 4 臂老虎机问题，有 $K = 4$ 个动作，分别标记为 1，2，3，4。我们使用 ϵ-greedy 算法进行动作选择。每个动作，我们初始花 Q 值为 0，即 $Q_1(a) = 0$, for all a。假设我们获得了一个动作和奖励的序列，如下：

$$A_1 = 1, R_1 = -1, A_2 = 2, R_2 = 1, A_3 = 2, R_3 = -2, A_4 = 2, R_4 = 2, A_5 = 3, R_5 = 0$$

计算交互过程中动作平均奖励？

- t=1： $Q_1(1) = 0, Q_1(2) = 0, Q_1(3) = 0, Q_1(4) = 0$
- t=2: $Q_2(1) = -1, Q_2(2) = 0, Q_2(3) = 0, Q_2(4) = 0$
- t=3: $Q_3(1) = -1, Q_3(2) = 1, Q_3(3) = 0, Q_3(4) = 0$
- t=4: $Q_4(1) = -1, Q_4(2) = -1/2, Q_4(3) = 0, Q_4(4) = 0$
- t=5: $Q_5(1) = -1, Q_5(2) = 1/3, Q_5(3) = 0, Q_5(4) = 0$
- t=6: $Q_5(1) = -1, Q_5(2) = 1/3, Q_5(3) = 0, Q_5(4) = 0$

增量实现 (Incremental Implementation)

给定 Q_n 和第 n 次获得的奖励 R_n, 则所有 n 次奖励的平均奖励通过一下方式计算得到：

$$
\begin{aligned}
Q_{n+1} &= \frac{1}{n}\sum_{i=1}^{n} R_i \\
&= \frac{n-1}{n} Q_n + \frac{1}{n} R_n \\
&= Q_n + \frac{1}{n}[R_n - Q_n]
\end{aligned}
\tag{2}
$$

这个公式适用于任意的 Q_1，即使对于 $n = 1$，也可以得到 $Q_2 = R_1$。这种实现只需要保存 Qn 和 n 的内存，并且对于每个新的奖励只需要进行简单的计算。

上述更新公式可以简化成如下形式：

NewEstimate $\leftarrow$ OldEstimate $+$ StepSize $\cdot$ (Target $-$ OldEstimate)

注意：

- 表达式 $[Target - OldEstimate]$ 是估计误差，通过朝着 “Target” 迈出一步来减少这个误差。
- 需要注意的是，增量方法中使用的步长参数（StepSize）会随着时间步长的变化而变化。在处理动作 a 的第 n 个奖励时，该方法使用步长参数 $1/n$。在本书中，我们用符号 α 或更一般地用 $\alpha_t(a)$ 来表示步长参数。

多臂老虎机增量实现伪代码

完整的使用增量计算样本平均值和 ϵ-贪心动作选择的强盗算法的伪代码如下所示。其中假设函数 bandit(a) 接受一个动作并返回相应的奖励。

A simple bandit algorithm

Initialize, for $a = 1$ to k:

$Q(a) \leftarrow 0$

$N(a) \leftarrow 0$

Loop forever:

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

$R \leftarrow bandit(A)$

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

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

图. Caption

代码实现解读

代码解读和课堂演示环节。