Author：Liu Jian

Time：2020-07-23

机器学习14-强化学习

1 基本概念

强化学习 (Reinforcement Learning) 任务常用马尔可夫决策过程 (Markov Decision Process, MDP) 来描述，MDP 对应了四元组 $E=\langle \mathbb{X},\mathbb{A},P,R\rangle$，其中，

1. $\mathbb{X}$ 为状态空间；
2. $\mathbb{A}$ 为动作空间；
3. P 为状态转移函数：$\mathbb{X}\times\mathbb{A}\times\mathbb{X}\to\mathbb{R}$，即在 x 状态下执行动作 a 后状态转移为 x' 的概率 $P^a_{x\leftarrow x'}=Pr(X_{t+1}=x'|X_t=x,A_t=a)$；
4. R 为奖赏函数：$\mathbb{X}\times\mathbb{A}\times\mathbb{X}\to\mathbb{R}$，$R^a_{x\leftarrow x'}$ 表示 x 状态下执行动作 a 且状态转移为 x' 的情况下产生的奖赏。可以看到，这里的奖赏是确定性的，而白板推导中的奖赏更宽泛，是随机性的，服从某个概率分布 $Pr(R^a_{x\leftarrow x'}|X_t=x,A_t=a,X_{t+1}=x')$。白板推导中，条件概率 $Pr(X_{t+1}=x',R_{t+1}=r|X_t=x,A_t=a)$ 称为动态特性，给出了状态转移概率和奖赏概率。

策略 π：

1. 确定性策略 $\pi(x)$，给出了状态 x 下要执行的动作，即 $\pi(x)\in\mathbb{A}$；
2. 随机性策略 $\pi(x,a)$，给出了状态 x 下执行动作 a 的概率，即 $\pi(x,a)=Pr(A_t=a|X_t=x)\in[0,1]$。对于确定性策略，我们也可以使用 $\pi(x,a)$ 来描述状态 x 下执行动作 a 的概率，显然，这个概率值要么为 0 要么为 1。

在强化学习任务中，学习的目的就是要找到能使价值函数最大化的策略。价值函数有多种计算方式，常用的有：

1. 策略 π 下的 T 步价值函数：$V^\pi_T(x)=\mathbb{E}^\pi\left[\frac{1}{T}\sum_{t=1}^T R_t|X_0=x\right]$；
2. 策略 π 下的 γ 折扣价值函数：$V^\pi_\gamma(x)=\mathbb{E}^\pi\left[\sum_{t=0}^{+\infty}\gamma^t R_{t+1}|X_0=x\right]$。

强化学习中没有监督学习中的有标记样本，即没有人告诉机器在什么状态做什么动作，只有等到最终结果揭晓之后，才能“反思”之前动作是否正确来进行学习，因此，强化学习在某种意义上可以看成“延迟标记信息”的监督学习问题。

下文要介绍的内容：

1. K-摇臂赌博机：单步强化学习任务的理论模型，不用考虑状态的转移，是对 MDP 的进一步简化；模型信息未知，我们需要边探索模型信息，边最大化单步奖赏，存在两种算法：ϵ -贪心算法和 Softmax 算法。
2. 有模型学习：model-based learning，在 MDP 模型信息已知的情况下，通过最大化价值函数，学习最优的策略，包括策略迭代和价值迭代两种方法，其中策略迭代的每一次迭代又分策略评估和策略改进两步。
3. 免模型学习：model-free learning，也称无模型学习，即 MDP 模型信息 (转移概率、奖赏函数等) 未知情况下的学习，方法有：
    1. 蒙特卡罗强化学习，包括：同策略蒙特卡罗强化学习算法和异策略蒙特卡罗强化学习算法；
    2. 时序差分学习，也存在同策略与异策略之分，同策略算法有 Sarsa 算法，异策略算法有 Q-learning 算法。
4. 价值函数近似：从离散到连续，讨论了连续状态空间的处理方式；
5. 模仿学习：imitation learning，也称学徒学习 (apprenticeship learning)，示范学习 (learning from demonstration)，观察学习 (learning by watching)，研究如何在强化学习中引入或者说利用人类专家给出的信息 (范例数据)，包括：
    1. 直接模仿学习：人类专家会给出一批状态-决策轨迹数据，其中每个"状态-决策对"就可以看做一个样本点 (相当于有标记样本，状态是示例，动作是标记)，由此我们可以使用监督学习，基于这些"状态-动作对"学习出初始策略 (离散动作对应分类算法，连续动作对应回归算法)，再通过强化学习方法对初始策略进行改进，从而获得更好的策略；
    2. 逆强化学习 (inverse reinforcement learning)：除了学习策略外，考虑到一般任务中的奖赏函数是人们直接假定的，因此我们还可以基于人类专家提供的范例数据对奖赏函数进行学习；可见，逆强化学习的目标有两个，即奖赏函数+策略。逆强化学习的背后逻辑是：欲使机器做出与人类给出的范例数据一致的行为，等价于寻找某种奖赏函数和相应的最优策略，使产生的轨迹与范例数据一致；换而言之，我们需要寻找某种奖赏函数使得范例数据是最优的，然后即可使用这个奖赏函数来训练强化学习的策略。西瓜书上给出了奖赏函数为线性模型的情况，不再赘述。

2 K-摇臂赌博机

模型描述：有 K 个摇臂，每个摇臂对应一个概率分布，每次选择按下其中一个摇臂，则这个摇臂会以一定的概率吐出一些硬币，但这个概率分布并不知道，即模型信息未知；赌徒的目标是，通过一定的策略最大化自己的奖赏，即获得最多的硬币。

显然，因为模型信息未知，所以我们一方面要对这 K 个摇臂的吐币概率进行探索，一方面还要使我们的得到的奖赏尽可能大。这两个方面是矛盾的，在摇臂次数一定的情况下，为了对摇臂的吐币概率进行探索，我们需要对每个摇臂都进行尝试；而为了使我们得到的奖赏尽可能大，我们会偏向于把摇臂指标分配给当前平均奖赏高的摇臂，这称为探索-利用窘境 (Exploration-Exploitation dilemma)。可以看到，存在两种极端策略：每次摇臂时，每个摇臂被摇到的机会均等，这称为仅探索 (exploration-only) 法；每次摇臂时，选择当前平均奖赏最大的摇臂 (若有多个摇臂同为最优，则随机选取一个)，这称为仅利用 (exploitation-only) 法。显然，仅探索法对平均奖赏较高的摇臂考虑不足，因此无法最大化我们的奖赏；而仅利用法基于贪心策略，对模型的信息了解得不够充分，很可能经常选不到真实最优摇臂。我们可以采用如下方法对探索和利用这两个方面进行平衡：

1. ϵ -贪心算法：每次摇臂，我们以 ϵ 的概率进行探索，即以均匀概率随机选择一个摇臂；以 $1-\epsilon$ 的概率进行利用，即选择当前平均奖赏最高的摇臂。需要说明的是，若摇臂次数非常大，在一段时间后，每个摇臂的平均奖赏都能很好地近似出来，不再需要探索，这种情况下可让 ϵ 随着尝试次数的增加而逐渐减小。
2. Softmax 算法则没有明显的探索和利用两个步骤，每次摇臂的选择服从如下的 Boltzmann 分布：

$$Pr(k)=\frac{\exp\{Q(k)/\tau\}}{\sum_{i=1}^{K}\exp\{Q(i)/\tau\}}$$

其中，$Q(k)$ 表示第 k 个摇臂当前的平均奖赏；$\tau>0$ 称为温度，其值越小，平均奖赏高的摇臂被选取的概率越高，$\tau\to 0$ 时，趋于仅利用，$\tau\to\infty$ 时，趋于仅探索。作为对比，ϵ -贪心算法中当前最优摇臂被选中的概率为 $1-\epsilon+\frac{\epsilon}{K}$（假设当前最优摇臂只有一个），而其他每个摇臂被选中的概率为 $\frac{\epsilon}{K}$ 。

3 有模型学习

有模型学习对应 MDP 四元组 $E=\langle\mathbb{X},\mathbb{A},P,R\rangle$ 均为已知的情况。有两种方法：策略价值迭代、价值迭代。

3.1 策略迭代 (policy iteration)

策略迭代的思路是在已有策略的基础上进行迭代优化。显然，在能对策略进行评估的基础上我们才能对策略进行改进，因此策略的一个更新步分策略评估和策略改进两个阶段。

3.1.1 策略评估

前面我们给出了状态价值函数：

$$\begin{cases}V_T^\pi(x)=\mathbb{E}^\pi\left[\frac{1}{T}\sum_{t=1}^T R_t|X_0=x\right]\\ V_\gamma^\pi(x)=\mathbb{E}^\pi\left[\sum_{t=0}^{+\infty}\gamma^t R_{t+1}|X_0=x\right]\end{cases}$$

可以看到，$V_T^\pi(x)$ 是奖赏的算术平均，而 $V_\gamma^\pi(x)$ 是奖赏的加权平均，总权重及配分状况为 $\frac{1}{1-\gamma}=1+\gamma+\gamma^2+\gamma^3+\cdots$ 。

由 MPD 的马尔可夫性，可推得如下的递推关系 (Bellman 等式)：

$$\begin{cases}V_T^\pi(x)=\sum_a\pi(x,a)\sum_{x'}P_{x\to x'}^a\left(\frac{1}{T}R_{x\to x'}^a+\frac{T-1}{T}V_{T-1}^\pi(x')\right)\\ V_\gamma^\pi(x)=\sum_a\pi(x,a)\sum_{x'}P_{x\to x'}^a\left(R_{x\to x'}^a+\gamma V_\gamma^\pi(x')\right)\end{cases}$$

根据上述递归式使用动态规划算法就可以计算策略的状态价值函数，而状态价值函数也就是策略的评估指标。

类似地，我们定义状态-动作价值函数：

$$\begin{cases}Q_T^\pi(x,a)=\mathbb{E}^\pi\left[\frac{1}{T}\sum_{t=1}^T R_t|X_0=x,A_0=a\right]\\ Q_\gamma^\pi(x,a)=\mathbb{E}^\pi\left[\sum_{t=0}^{+\infty}\gamma^t R_{t+1}|X_0=x,A_0=a\right]\end{cases}$$

有了状态价值函数，就能直接计算出状态-动作价值函数，它们之间存在如下的关系：

$$\begin{cases}Q_T^\pi(x,a)=\sum_{x'}P_{x\to x'}^a\left(\frac{1}{T}R_{x\to x'}^a+\frac{T-1}{T}V_{T-1}^\pi(x')\right)\\ Q_\gamma^\pi(x,a)=\sum_{x'}P_{x\to x'}^a\left(R_{x\to x'}^a+\gamma V_\gamma^\pi(x')\right)\end{cases}$$

引入状态-动作价值函数是因为策略改进需要用到。

3.1.2 策略改进

可以证明，若某个策略 π 能使如下的最优 Bellman 等式成立，则该策略就是一个最优策略：

$$\begin{cases}V_T^\pi(x)=\max_a\sum_{x'}P_{x\to x'}^a\left(\frac{1}{T}R_{x\to x'}^a+\frac{T-1}{T}V_{T-1}^\pi(x')\right)=\max_a Q_T^\pi(x,a)\\ V_\gamma^\pi(x)=\max_a\sum_{x'}P_{x\to x'}^a\left(R_{x\to x'}^a+\gamma V_\gamma^\pi(x')\right)=\max_a Q_\gamma^\pi(x,a)\end{cases}$$

价值函数的最优解是唯一的，但最优策略可能有多个，记最优价值函数为 $V_T^*(x),Q_T^*(x,a)$（或 $V_\gamma^*(x),Q_\gamma^*(x,a)$ ）。为了便于描述，下面我们忽略下标 T,γ 。

记改进前的策略为 π ，改进后的策略为 π' ，非最优策略的改进公式如下：

$$\pi'(x) = \arg\max_a Q^{\pi}(x,a),\ \ \forall x \in \mathbb{X}$$

可以看到，改进后的策略 π' 是确定性策略。此外，我们可以证明迭代是收敛的，且像这样改进后的策略 π' 一定优于改进前的策略 π，即 $\forall x \in \mathbb{X},\ V^{\pi'}(x) \geqslant V^{\pi}(x)$。迭代过程中，当 $\forall x \in \mathbb{X},\ V^{\pi'}(x) = V^{\pi}(x)$ 时，我们可以证明，策略 π' 满足最优 Bellman 等式，即此时价值函数达到了最优 $\forall x \in \mathbb{X},\ V^{\pi'}(x) = V^{\pi}(x) = V^{*}(x)$，策略 π', π 均为最优策略。

3.2 价值迭代 (value iteration)

策略迭代先计算当前策略的价值函数，再基于价值函数更新策略，这一过程交替进行直到价值函数不再增加为止。价值迭代则只对价值函数进行迭代更新，在价值函数的更新过程中不再考虑和计算策略的更新，而只是在求得最优价值函数后再根据最优价值函数求最优策略。记更新前的价值函数为 V，更新后的价值函数为 V'，我们不加证明地给出价值迭代的计算公式：

$$\begin{cases} V'_T(x) = \max_a \sum_{x'} P^a_{x\to x'} \left(\frac{1}{T} R^a_{x\to x'} + \frac{T-1}{T} V_{T-1}(x')\right) = \max_a Q^{\pi}_T(x,a) \\ V'_{\gamma}(x) = \max_a \sum_{x'} P^a_{x\to x'} \left(R^a_{x\to x'} + \gamma V_{\gamma}(x')\right) = \max_a Q^{\pi}_{\gamma}(x,a) \end{cases}$$

迭代得到价值函数 V^* 后，最优策略 π^*：

$$\pi^*(x) = \arg\max_a Q^*(x,a)$$

可以看到，价值迭代可看视为最优 Bellman 等式的不动点解法 (迭代解法)，此外，价值迭代也可由策略迭代推得。

4 免模型学习

在现实的强化学习任务中，MDP 的转移概率、奖赏函数往往很难得知，甚至很难知道一共有多少状态，这种情况下的强化学习就是免模型学习，也称无模型学习。

4.1 蒙特卡罗强化学习

蒙特卡罗强化学习就是使用采样的方法评估价值函数，再根据价值函数学习策略。因采样次数有限，所以这种方法更适用于 T 步价值函数。此外，前面策略迭代算法估计的是状态价值函数 V，而最终的策略是通过状态-动作价值函数 Q 来获得，当模型已知时，从 V 到 Q 很容易转换，而当模型未知时，这一转换也很困难，因此，我们直接估计状态-动作价值函数 Q。

前面提到的 T 步状态-动作价值函数实际上是对随后的 T 步奖赏求和之后取算术平均，因此对不同的 T，我们可以不加区分，统一认为是对 $Q(x,a)$ 的估计。从起始状态出发，使用某种策略 π 进行采样，执行该策略 T 步得到轨迹：

$$\langle x_0, a_0, r_1, x_1, a_1, r_2, \cdots, x_{T-1}, a_{T-1}, r_T, x_T \rangle$$

对轨迹中出现的每一对状态-动作，记录其后的奖赏之和，再除以其后的步数，得到相应的 $Q_t(X,A)$，对 (X,A) 值相同的项 (比如 $(X,A) = (x,a)$) 取平均，即得 $Q(x,a)$；若存在多条轨迹，则作进一步平均即可。策略的改进规则为：

$$\pi'(x) = \arg\max_a Q(x,a)$$

$\pi'(x)$ 是一个确定性策略。

此外，确定性策略对于某个状态只会输出一个动作，因此基于确定性策略进行采样，只能得到多条相同的轨迹，这就会导致类似于 K-摇臂赌博机中的仅利用问题。受 K-摇臂赌博机的 ε -贪心算法的启发，我们可以对确定性策略 π 引入随机性，得到 ε -贪心策略 π^{ϵ}：

$$\pi^{\epsilon}(x) = \begin{cases} \pi(x), & \text{以概率} 1-\epsilon \\ \mathbb{A}\text{中以均匀概率选取动作}, & \text{以概率} \epsilon \end{cases}$$

我们使用 π^ϵ 来进行采样，计算 $Q(x,a)$，更新策略。

总地来说，同策略 (on-policy) 蒙特卡罗强化学习算法描述如下：

1. 对确定性策略 π，构造其 ε -贪心策略 π^ϵ，执行 π^ϵ 产生轨迹 $\langle x_0,a_0,r_1,x_1,a_1,r_2,\cdots,x_{T-1},a_{T-1},r_T,x_T\rangle$；
2. 根据这条新轨迹，更新状态-动作价值函数 $Q(x,a)$ (注意是更新，而不是重新计算，也就是说前期迭代时，前期策略产生的轨迹在计算 $Q(x,a)$ 时也会被使用)；
3. 根据当前 $Q(x,a)$，计算最优确定性策略 π'，将 π' 赋给 π，进入下一轮迭代。

下面将要介绍的异策略 (off-policy) 蒙特卡罗强化学习算法则只借助 ε -贪心策略 π^ϵ 进行采样，而评估和改进的是确定性策略 π。将一种策略的采样结果用于另一种策略的价值评估则类似于重要性采样，在评估时考虑相应权重即可。

> 重要性采样：
> $\mathbb{E}_p[f(x)]=\int p(x)f(x)=\int q(x)\frac{p(x)}{q(x)}f(x)=\mathbb{E}_q[\frac{p(x)}{q(x)}f(x)]\approx\frac{1}{N}\sum_{i=1}^N\frac{p(x_i)}{q(x_i)}f(x_i)$，其中 $x_i\sim q(x)$，$\frac{p(x_i)}{q(x_i)}$ 为权重。

给定轨迹 $\langle x_0,a_0,r_1,x_1,a_1,r_2,\cdots,x_{T-1},a_{T-1},r_T,x_T\rangle$，策略 π_1 和策略 π_2 产生该轨迹的概率分别为 $w=\prod_{i=0}^{T-1}\pi_1(x_i,a_i)P_{x_i\to x_{i+1}}^{a_i}$ 和 $\prod_{i=0}^{T-1}\pi_2(x_i,a_i)P_{x_i\to x_{i+1}}^{a_i}$，因此权重就为 $\prod_{i=0}^{T-1}\pi_1(x_i,a_i)/\pi_2(x_i,a_i)\triangleq\prod_{i=0}^{T-1}\frac{1}{p_i}$。当 π_1 取确定性策略 π，而 π_2 取 π 的 ε -贪心策略 π^ϵ 时，西瓜书上给出的结果如下：

$$p_i=\begin{cases}1-\epsilon+\frac{\epsilon}{|\mathbb{A}|}, & a_i=\pi(x_i)\\ \frac{\epsilon}{|\mathbb{A}|}, & a_i\neq\pi(x_i)\end{cases}$$

注意，对于确定性策略 π，$\pi(x)$ 不表示概率，而是某个动作，$\pi(x_i,a_i)$ 表示在策略 π 下给定状态 x_i 选择动作 a_i 的概率，显然，这一概率要么为 1 要么为 0。当 $a_i=\pi(x_i)$ 时，$\pi(x_i,a_i)=1$，根据 π^ϵ 的定义，策略 π^ϵ 在状态 x_i 下选择动作 a_i 的概率为 $1-\epsilon+\frac{\epsilon}{|\mathbb{A}|}$，因此 $p_i=\pi^\epsilon(x_i,a_i)/\pi(x_i,a_i)=1-\epsilon+\frac{\epsilon}{|\mathbb{A}|}$；若 $a_i\neq\pi(x_i)$，则根据 π^ϵ 的定义，策略 π^ϵ 在状态 x_i 下选择动作 a_i 的概率为 $\frac{\epsilon}{|\mathbb{A}|}$，但此时，$\pi(x_i,a_i)=0$，所以 $p_i=\infty$，至于为何此时西瓜书上取 $p_i=\frac{\epsilon}{|\mathbb{A}|}$，书上并没有说明，个人对此存疑。

有了权重的计算公式后，异策略蒙特卡罗强化学习算法和同策略蒙特卡罗强化学习算法类似，只用在计算 $Q(x,a)$ 时乘以相应权重即可。

4.2 时序差分 (time difference, TD) 学习

蒙特卡罗强化学习算法在一个完整的采样轨迹完成后再对状态-动作价值函数进行更新，这样效率会比较低。时序差分算法则将动态规划算法与蒙特卡罗强化学习算法结合，能做到更高效的免模型学习。西瓜书上本节的公式及推导并不是很详细，因此我们这里只介绍 TD 的基本思路，具体细节可参考其他资料。

相比蒙特卡罗强化学习算法在一个完整的采样轨迹完成后再对状态-动作价值函数进行更新的“批处理式”方法，时序差分算法则是一种“增量式”方法，采样过程中状态每转移一次 (每执行一步策略) 就对 $Q(x,a)$ 进行更新。同策略的 TD 算法有 Sarsa 算法，异策略的 TD 算法有 Q-learning 算法。

5 价值函数近似

本小节也只是介绍基本思路，具体细节可参考其他资料。

前面我们考察的都是离散状态 x ，对于离散状态，我们只需计算价值函数在有限个 ($|\mathbb{X}|$ 个) 状态下的值即可，计算目标只是一个向量。但若状态是高维连续的变量 $\boldsymbol{x} \in \mathbb{X}$ ，我们就无法计算价值函数在每一点的值，只能像机器学习所做的那样，假设一个价值函数模型来逼近真实的价值函数，比如可取线性模型：

$$V_\theta(\boldsymbol{x}) = \theta^T \boldsymbol{x}$$

其中， θ 为待定参数 (对状态-动作价值函数 $Q(\boldsymbol{x}, a)$ 也可以作类似的模型假设)。和机器学习一样，我们使由模型 $V_\theta(\boldsymbol{x})$ 输出的值与借助时序差分学习得到的价值函数估计值的经验风险最小化，从而估计参数 θ ，优化算法可采用梯度下降法。对参数 θ 的学习 (也就是对价值函数的学习) 可以糅合到对策略的学习中，由此可得线性价值函数近似 Sarsa 算法，线性价值函数近似 Q-learning 算法。此外，我们还可以引入核技巧实现非线性价值函数近似。

6 模仿学习

见第一章的总结。