【推荐算法】推荐系统中的EE问题及传统Bandit算法

原创 奔跑的老白菜 后来遇见AI 2020-12-10

写在前面：笔者前面写的文章大都是基于机器学习和深度学习的推荐算法模型，这里介绍一个推荐系统中的经典问题：EE问题。

1 推荐系统中的EE问题

Exploration and Exploitation（EE问题，探索与开发）是计算广告和推荐系统里常见的一个问题，为什么会有EE问题？是为了平衡推荐系统的准确性和多样性，在进行物品推荐时，不仅要投其所好，还要进行适当的长尾物品挖掘。

- Exploitation：对于用户已经确定的兴趣当然要迎合投放；
- Exploration：除了推荐已知的用户感兴趣的内容，还需要不断探索用户其他兴趣。

2 MAB 问题

赌场的老虎机有一个绰号叫单臂强盗（single-armed bandit），因为它即使只有一只胳膊，也会把你的钱拿走。而多臂老虎机（Multi-armed bandit，简称MAB或多臂强盗）就从这个绰号引申而来。假设你进入一个赌场，面对一排老虎机（所以有多个臂），**由于不同老虎机的期望收益和期望损失不同，你采取什么老虎机选择策略来保证你的总收益最高呢？这就是经典的多臂老虎机问题。**

这个经典问题集中体现了推荐系统中**一个核心的权衡问题**：我们是应该**探索（exploration）**，去尝试挖掘用户新的兴趣，还是应该**守成（exploitation）**，**坚持**目前已知的用户兴趣？在多臂老虎机问题中，探索意味着去玩还没玩过的老虎机，但这有可能使你花太多时间和金钱在收益不好的机器上；而守成意味着只玩目前为止给你收益最好的机器，但这又可能使你失去找到更好机器的机会。而类似抉择在日常生活中随处可见：去一个餐厅，你是不是也纠结于是点熟悉的菜品，还是点个新菜？去一个地方，是走熟知的老路还是选一条新路？

MAB 中的每个摇臂都是一个选项，所以它其实是一个选择问题，如果想要获得最大化的累积奖赏，最好的办法就是试一试，但是不能盲目的去试，而是有策略的试一试，这些策略就是bandit算法。

3 Bandit算法

Bandit算法是在线学习的一种，一切通过数据收集进行的概率预估任务，都可以通过Bandit系列算法来进行在线优化。这里的“在线”，并不是指互联网的线上，而是指算法模型参数根据观

察数据不断变化。

如何将Bandit算法、MAB问题、推荐系统中的EE问题三者联系起来呢？

假设我们已经经过一些试验，得到了当前每个老虎机的吐钱的概率预估值，如果想要获得最大的收益，我们会一直摇哪个吐钱概率预估值最高的老虎机，这就是Exploitation。但是，当前获得的信息并不是老虎机吐钱的真实概率，可能还有吐钱概率更高的老虎机没有被我们试验出来，因此还需要进一步探索，这就是Exploration问题。

Bandit 算法中有几个关键元素：臂，回报，环境。

- 臂：指每次选择的候选项，有几个选项就有几个臂。
- 回报：就是选择一个臂之后得到的奖励，比如选择赌博机后吐出来的硬币。
- 环境：就是决定每个臂不同的那些因素，统称为环境。

将以上的关键元素对应到推荐系统中。

- 臂：指每次推荐的候选项，可以是具体物品，也可以是物品类别，也可以是推荐策略和算法。
- 回报：指用户对推荐的结果是否满意。
- 环境：指给当前用户推荐时的所有周边环境

如何衡量Bandit算法的优劣？

Bandit算法需要量化一个核心问题：错误的选择到底有多大的遗憾？能不能遗憾少一些？所以我们便有了衡量Bandit算法的一个指标：**累积遗憾**。

$$R_A(T) = E(\sum_{t=1}^{T} r_{t,a_t^*}) - E(\sum_{t=1}^{T} r_{t,a_t})$$

这里t表示当前选择的轮数；T表示总共选择的轮数；$R_A(T)$表示经过T次选择后的累计遗憾；r_{t,a_t^*}表示在第t次选择时选择了最好的臂所获得的收益；r_{t,a_t}表示在第t次选择时实际所选的臂所带来的收益。公式右边的第一项表示第t轮的期望最大收益，而右边的第二项表示第t轮实际选择的臂获取的收益，把每次差距累加起来就是总的遗憾。

Bandit 算法的套路就是：小心翼翼地试，越确定某个选择好，就多选择它，越确定某个选择差，就越来越少选择它。如果某个选择实验次数较少，导致不确定好坏，那么就多给一些被选择机会，直到确定了它是金子还是石头。简单说就是，**把选择的机会给“确定好的”和“还不确定的”**。

有了衡量算法优劣的指标就来看一看具体的Bandit算法吧。

3.1 朴素Bandit算法

核心思想：先随机试若干次，计算每个臂的平均收益，一直选均值最大那个臂。

这个算法是我们普通人实际生活中最常采用的，不可否认，它还是比随机乱猜要好。

3.2 Epsilon-Greedy算法

先确定一个$(0,1)$之间较小的数$epsilon$，每轮以概率$epsilon$在所有臂中随机选一个臂，以$1-epsilon$的概率选择截止当前平均收益最大的那个臂。根据选择臂的回报值来对回报期望进行更新。

简单清晰地以$epsilon$的值控制对exploit和explore的偏好程度，每次决策以概率$epsilon$去勘探（Exploration），$1-epsilon$的概率来开发（Exploitation），基于选择的item及回报，更新item的回报期望。

优点：能够应对变化，即如果item的回报发生变化，能及时改变策略，避免卡在次优状态。同时$epsilon$的值可以控制对Exploit和Explore的偏好程度。

缺点：策略运行一段时间后，我们已经对各item有了一定程度了解，但没有充分利用这些信息，仍然不做任何区分地随机Exploration，这是Epsilon-Greedy算法的缺点。

3.3 Thompson sampling算法

该方法基于$Beta$分布，假设每个老虎机都有一个吐钱的概率p，同时该概率p的概率分布符合$Beta(wins, lose)$分布，每个臂都维护一个$Beta$分布的参数，即$wins, lose$。每次试验后，选中一个臂，摇一下，有收益则该臂的$wins$增加1，否则该臂的$lose$增加1。

每次选择臂的方式是：用每个臂现有的$Beta$分布产生一个随机数b，选择所有臂产生的随机数中最大的那个臂去摇。

3.4 UCB算法

前面提到了，Epsilon-Greedy算法在探索的时候，所有的老虎机都有同样的概率被选中，这其实没有充分利用历史信息，比如每个老虎机之前探索的次数，每个老虎机之前的探索中吐钱的频率。

那我们怎么能够充分利用历史信息呢？首先，根据当前老虎机已经探索的次数，以及吐钱的次数，我们可以计算出当前每个老虎机吐钱的观测概率$\tilde{p}$。同时，由于观测次数有限，因此观测概率和真实概率p之间总会有一定的差值Δ，即$\tilde{p}-\Delta <= p <= \tilde{p}+\Delta$。

基于上面的讨论，我们得到了另一种常用的Bandit算法：UCB(Upper Confidence Bound)算法。**该算法在每次推荐时，总是乐观的认为每个老虎机能够得到的收益是$\tilde{p}+\Delta$。**

好了，接下来的问题就是观测概率和真实概率之间的差值Δ如何计算了，我们首先有两个直观的理解：

- 对于被选中的老虎机来说，每多被选择一次会使对应地Δ变小，当被选择无穷多次时，Δ趋近于0，最终会小于其他被选中次数较少的老虎机的Δ。
- 对于没有被选中的老虎机，Δ会随着轮数的增大而增加，最终会大于其他被选中的老虎机。

因此，当进行了一定的轮数的时候，每个老虎机都有机会得到探索的机会。

如何计算Δ?首先了解Chernoff-Hoeffding Bound。

[Chernoff-Hoeffding Bound] 假设$x_1, x_2, \ldots, x_n$是在[0, 1]之间取值的独立同分布随机变量，用$\tilde{p}=\frac{\sum_{i=1}^{n} x_i}{n}$表示样本均值，用$p$表示分布的均值，那么，

$$P(|\tilde{p}-p| \leq \Delta) \geq 1-2e^{-2n\Delta^2}$$

当Δ取值$\sqrt{\frac{2lnT}{n}}$（其中T是目前的试验次数，n是该老虎机臂被选择的次数）时，

$$P(|\tilde{p}-p| \leq \sqrt{\frac{2lnT}{n}}) \geq 1-\frac{2}{T^4}$$

也就是说，$\tilde{p}-\sqrt{\frac{2lnT}{n}} <= p <= \tilde{p}+\sqrt{\frac{2lnT}{n}}$是以大于等于$1-\frac{2}{T^4}$的概率成立的。

- $T=2$时，成立的概率为0.875
- $T=3$时，成立的概率为0.975
- $T=4$时，成立的概率为0.992

可以看出，$\Delta=\sqrt{\frac{2lnT}{n}}$是一个挺靠谱的选择。

有了Δ，那么UCB算法中每个老虎机对应的$\tilde{p}+\Delta$的计算公式也就确定了，

$$\tilde{p}+\sqrt{\frac{2lnT}{n}}$$

其中前面是这个老虎机到目前的收益均值；后面的叫做bonus，本质上是均值的标准差，反映了置信区间，可以简单地理解为不确定性的程度，区间越宽，越不确定。这个公式反映了如下思想：均值越大，标准差越小，被选中的概率会越来越大，起到了exploit的作用；同时如果某个老虎机置信区间很宽（被选次数很少，标准差很大）也会得到试验机会，起到了explore的作用。

UCB算法的具体步骤：先对每一个臂都试一遍之后，每次选择$\tilde{p}+\sqrt{\frac{2lnT}{n}}$值最大的那个臂。

UCB是一种乐观的算法，选择置信区间上界排序，如果使用悲观保守的做法，是选择置信区间下界排序。

4 小结

Exploration and Exploitation（EE问题，探索与开发）是计算广告和推荐系统里常见的一个问题，在给用户进行物品推荐时，不仅要投其所好，还要进行适当的长尾物品挖掘。

在权衡**Exploration and Exploitation**时，不妨有策略的试一试，**把选择的机会给“确定好的”和“还不确定的”**。

本文介绍了朴素Bandit算法、Epsilon-Greedy算法、Thompson sampling算法以及UCB算法等4个传统Bandit算法的核心思想和实现步骤。

实际上，Bandit算法是一种不太常用的推荐系统算法，究其原因，是**它能同时处理的物品数量不能太多**。但是，在冷启动和处理EE问题时，Bandit 算法简单好用，值得一试。