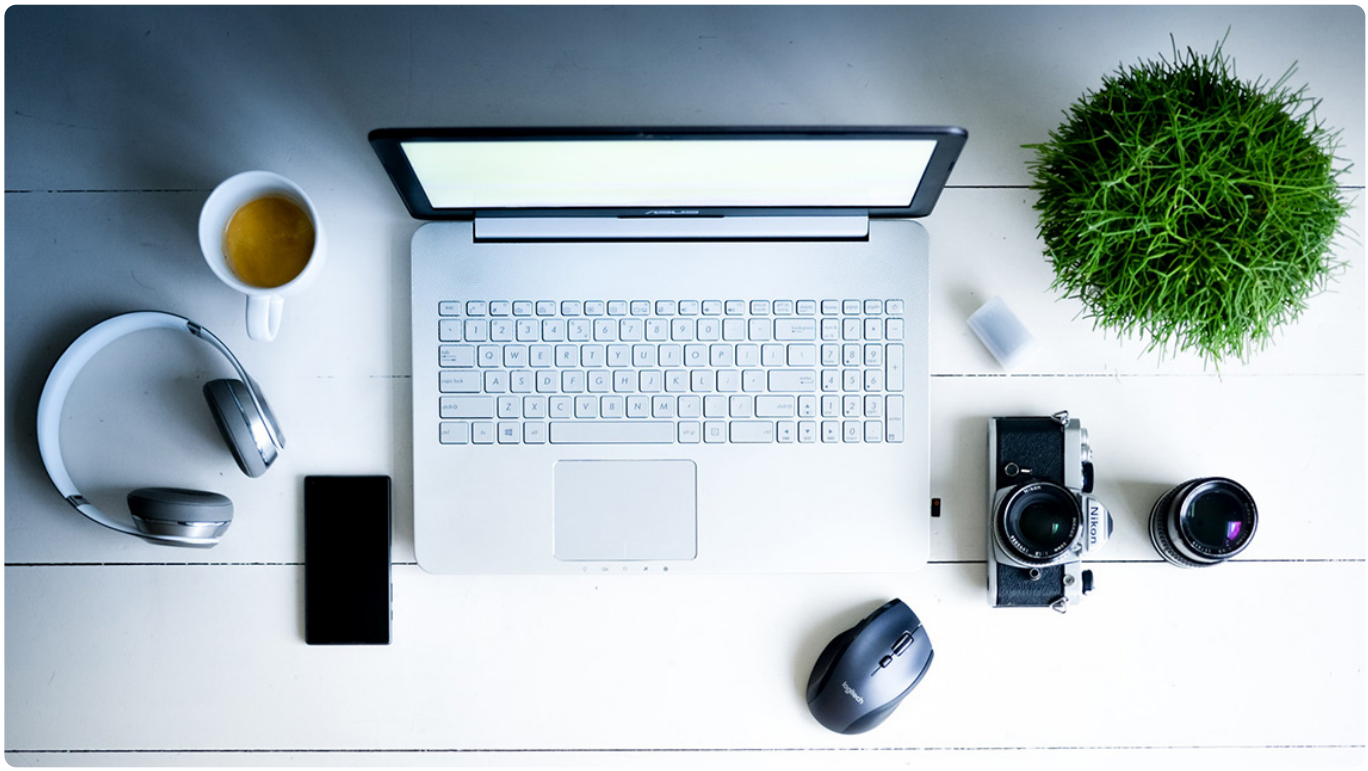
【MAB问题】结合上下文信息的Bandit算法

2018-04-11 刑无刀

推荐系统三十六式 进入课程 >



讲述：黄洲君
时长 10:37 大小 4.87M

上一篇文章我说到，Bandit 算法用的是一种走一步看一步的思路，这一点看上去非常佛系，似乎一点都不如机器学习深度学习那样厚德载物，但是且慢下结论，先看看我在前面介绍的那几个 Bandit 算法。

UCB 回顾

这些 Bandit 算法，都有一个特点：完全没有使用候选臂的特征信息。特征可是机器学习的核心要素，也是机器学习泛化推广的依赖要素。

没有使用特征信息的 Bandit 算法，问题就在于只能对当前已有的这些候选臂进行选择，对于新加入的候选只能从 0 开始积累数据，而不能借助已有的候选泛化作用。

举个例子，假如有一个用户是鹿晗的粉丝，通过 Bandit 算法有两个鹿晗的广告得到展示，并得到了较好的收益。

那么对于一个新的广告，如果具有鹿晗这个特征，直觉上前两个鹿晗广告的收益信息可以泛化到当前新广告上，新广告就不是完全从 0 开始积累数据，而是有了一定的基础，这样的收敛会更快。

UCB 和汤普森采样这两个 Bandit 算法在实际中表现很好。于是，前辈们就决定送 UCB 去深造一下，让它能够从候选臂的特征信息中学到一些知识。

UCB 就是置信上边界的简称，所以 UCB 这个名字就反映了它的全部思想。置信区间可以简单直观地理解为不确定性的程度，区间越宽，越不确定，反之就很确定。

1. 每个候选的回报均值都有个置信区间，随着试验次数增加，置信区间会变窄，相当于逐渐确定了到底回报丰厚还是可怜。
2. 每次选择前，都根据已经试验的结果重新估计每个候选的均值及置信区间。
3. 选择置信区间上界最大的那个候选。

“选择置信区间上界最大的那个候选”，这句话反映了几个意思：

1. 如果候选的收益置信区间很宽，相当于被选次数很少，还不确定，那么它会倾向于被多次选择，这个是算法冒风险的部分；
2. 如果候选的置信区间很窄，相当于被选次数很多，比较确定其好坏了，那么均值大的倾向于被多次选择，这个是算法保守稳妥的部分；
3. UCB 是一种乐观冒险的算法，它每次选择前根据置信区间上界排序，反之如果是悲观保守的做法，可以选择置信区间下界排序。

LinUCB

“Yahoo!”的科学家们在 2010 年基于 UCB 提出了 LinUCB 算法，它和传统的 UCB 算法相比，最大的改进就是加入了特征信息，每次估算每个候选的置信区间，不再仅仅是根据实验，而是根据特征信息来估算，这一点就非常的“机器学习”了。

在广告推荐领域，每一个选择的样本，由用户和物品一起构成，用户特征，物品特征，其他上下文特征共同表示出这个选择，把这些特征用来估计这个选择的预期收益和预期收益的置信区间，就是 LinUCB 要做的事情。

LinUCB 算法做了一个假设：一个物品被选择后推送给一个用户，其收益和特征之间呈线性关系。在具体原理上，LinUCB 有一个简单版本以及一个高级版本。简单版本其实就是让每一个候选臂之间完全互相无关，参数不共享。高级版本就是候选臂之间共享一部分参数。

先从简单版本讲起。

还是举个例子，假设现在两个用户，用户有一个特征就是性别，性别特征有两个维度，男，女。现在有四个商品要推荐给这两个用户，示意如下。

用户	性别	特征
u1	男	$x_1 = [1, 0]$
u2	女	$x_2 = [0, 1]$

两个用户就是 Bandit 算法要面对的上下文，表示成特征就是下面的样子。

参数	候选臂（商品）
$\theta_1 = [0.1, 0.5]$	华歌尔内衣
$\theta_2 = [0.2, 0.6]$	香奈儿口红
$\theta_3 = [0.9, 0.1]$	吉列剃须刀
$\theta_4 = [0.5, 0.6]$	苹果笔记本

每一次推荐时，用特征和每一个候选臂的参数去预估它的预期收益和置信区间。

$x_i \times \theta_j$，这就是给男性用户推荐剃须刀，给女性用户推荐口红，即使是新用户，也可以作出比随机猜测好的推荐，再观察用户是否会点击，用点击信息去更新那个被推荐了的候选臂的参数。

这里的例子还简化了一个地方，就是没有计算置信区间，这是 UCB 的精髓。下面来补上。

假如 D 是候选臂是候选臂在 m 次被选择中积累的特征，相当于就是 m 条样本，特征维度是 d，所以 D 是一个矩阵，维度是 m x d。

这 m 次被选择，每次得到用户的点击或者没点击，把这个反馈信息记录为一个 m x 1 的向量，叫做 C。所以这个候选臂对应的参数就是 d x 1 的向量，d 就是特征维度数，记录为一个戴帽子的西塔，$\hat{\theta}$。

按照 LinUCB 认为，参数和特征之间线性相乘就应该得到收益：

$$D_{m\times d} \times \hat{\theta_{d\times 1}} = C_{m\times 1}$$

你看 D 也知道，C 也知道，要求 θ ，这就很简单了。

$$\hat{\theta}_{d\times 1} = (D^T_{m\times d})^{-1} C_{m\times 1}$$

但是由于数据稀疏，实际上求参数西塔时不会这样简单粗暴，而是采用岭回归的方法，给原始特征矩阵加上一个单位对角矩阵后再参与计算：

$$\hat{\theta}_{d\times 1} = (D^T_{m\times d} D_{m\times d} + I_{d\times d})^{-1} D^T_{m\times d} C_{m\times 1}$$

每一个候选臂都像这样去更新它的参数，同时，得到参数后，在真正做选择时，用面对上下文的特征和候选臂的参数一起。

除了估算期望收益，还要计算置信区间的上边界，如果 x 是上下文特征，则期望收益和置信上边界的计算方法分别是下面的样子。

期望收益：

$$\hat{r} = x_{d\times 1}^T \hat{\theta}_{d\times 1}$$

置信区间上边界：

$$\hat{b} = \alpha \sqrt{x_{d\times 1}^T (D_{m\times d}^T D_{m\times d} + I_{d\times d})^{-1} x_{d\times 1}}$$

这两个计算结果都是标量数值。置信区间计算公式虽然看起来复杂，实际上反应的思想也很直观，随着被选择次数的增加，也就是 m 增加，这个置信上边界是越来越小的。

每一次选择时给每一个候选臂都计算这两个值，相加之后选择最大那个候选臂输出，就是 LinUCB 了。

刚才说到了岭回归（ridge regression），这里多说一句，岭回归主要用于当样本数小于特征数时，对回归参数进行修正。对于加了特征的 Bandit 问题，正好符合这个特点：试验次数（样本）少于特征数。

信息量有点大，我在这里再一次列出 LinUCB 的重点。

1. LinUCB 不再是上下文无关地，像盲人摸象一样从候选臂中去选择了，而是要考虑上下文因素，比如是用户特征、物品特征和场景特征一起考虑。
2. 每一个候选臂针对这些特征各自维护一个参数向量，各自更新，互不干扰。
3. 每次选择时用各自的参数去计算期望收益和置信区间，然后按照置信区间上边界最大的输出结果。
4. 观察用户的反馈，简单说就是“是否点击”，将观察的结果返回，结合对应的特征，按照刚才给出的公式，去重新计算这个候选臂的参数。

当 LinUCB 的特征向量始终取 1，每个候选臂的参数是收益均值的时候，LinUCB 就是 UCB。

说完简单版的 LinUCB，再看看高级版的 LinUCB。与简单版的相比，就是认为有一部分特征对应的参数是在所有候选臂之间共享的，所谓共享，也就是无论是哪个候选臂被选中，都会去更新这部分参数。

构建特征

LinUCB 算法有一个很重要的步骤，就是给用户和物品构建特征，也就是刻画上下文。

在“Yahoo！”的应用中，物品是文章。它对特征做了一些工程化的处理，这里稍微讲一下，可供实际应用时参考借鉴。

首先，原始用户特征有下面几个。

1. 人口统计学：性别特征（2 类），年龄特征（离散成 10 个区间）。
2. 地域信息：遍布全球的大都市，美国各个州。
3. 行为类别：代表用户历史行为的 1000 个类别取值。

其次，原始文章特征有：

1. URL 类别：根据文章来源分成了几十个类别。
2. 编辑打标签：编辑人工给内容从几十个话题标签中挑选出来的。

原始特征向量先经过归一化，变成单位向量。

再对原始用户特征做第一次降维，降维的方法就是利用用户特征和物品特征以及用户的点击行为去拟合一个矩阵 W。

$$\phi_u^T W \phi_a^T$$

就用逻辑回归拟合用户对文章的点击历史，得到的 W 直觉上理解就是：能够把用户特征映射到物品特征上，相当于对用户特征降维了，映射方法是下面这样。

$$\psi_u = \phi_u^T W$$

这一步可以将原始的 1000 多维用户特征投射到文章的 80 多维的特征空间。

然后，用投射后的 80 多维特征对用户聚类，得到 5 个类，文章页同样聚类成 5 个类，再加上常数 1，用户和文章各自被表示成 6 维向量。

接下来就应用前面的 LinUCB 算法就是了，特征工程依然还是很有效的。

总结

今天我和你分享了一种上下文有关的 Bandit 算法，叫做 LinUCB，它有这么几个优点：

1. 由于加入了特征，所以收敛比 UCB 更快，也就是比 UCB 更快见效；
2. 各个候选臂之间参数是独立的，可以互相不影响地更新参数；
3. 由于参与计算的是特征，所以可以处理动态的推荐候选池，编辑可以增删文章；

当然，LinUCB 以及所有的 Bandit 算法都有个缺点：同时处理的候选臂数量不能太多，不超过几百个最佳。因为每一次要计算每一个候选臂的期望收益和置信区间，一旦候选太多，计算代价将不可接受。

LinUCB 只是一个推荐框架，可以将这个框架应用在很多地方，比如投放广告，为用户选择兴趣标签，你还可以发挥聪明才智，看看它还能用来解决什么问题，欢迎留言一起交流。

上一篇 【MAB问题】简单却有效的Bandit算法

下一篇 【MAB问题】如何将Bandit算法与协同过滤结合使用

精选留言 (5)

写留言

cyrillian... 4
2018-07-09

真是越能看到后面的人越少。老师讲得不错，赞一个

展开

作者回复: 你真相了。

范深 3
2018-06-06

Bandit 的精髓之一就是在没有特征，没有状态的情况下进行选择。如果LinUCB引进来了，说明U-I的特征也有了，这时候能用的算法就很多了，包括DQN等。我理解的对吗?

展开

林彦 2
2018-04-11

简单搜索了下，除了典型的商品和内容推荐，也有人在非工业领域用于搜索更好的游戏策略(蒙特卡罗模拟)，更好的短信交互策略提高用户互动目标的完成度。如果更幻想一些，理论上有足够多的数据也可以协助更好地诊断疾病标签和推荐可能更有效的治疗方案(如果医疗机构有意愿积极协作)。

星 1
2018-08-06

老师好。看完您的视频，就能找份推荐的工作了吗？谢谢

展开

shangqiu86
2019-05-05

讲的很好，就是后面的linUCB感觉是个大工程，需要不断优化不断累积修正特征