上海交通大学安泰经济与管理学院

BUSS3620 人工智能导论

第一学期, 2023-24

Project #3. 广告投放

刘佳璐 助理教授

广告投放

本次项目将会制作一个能够自动选择出最优广告的 AI，旨在展示网络公司如何运用抽样技术去决定广告最佳投放策略，从而实现利润最大化。

```
$ python Advertising.py
The best version is:
AB Testing Result: 6
Thompson Sampling Result: 6
```

背景介绍

在数字时代，建立网站或营销活动只是营销过程的第一步。活动准备就绪后，公司应该在各种目标受众或感兴趣的人群中进行测试，从而判断出哪个广告系列效果最好。

A/B 测试是解决这个问题的一种被广泛使用的策略。这种方法对比两个或多个版本的广告、活动、网页等，对比不同版本对客户的影响。公司随机给每个客户分配一个版本的广告，然后观察客户是否购买了公司的产品。经过一段时间的观察后，公司可以计算不同广告版本的转化率（即购买人数除以看到广告的人数），继而选择转化率最高的广告投放给未来的客户。

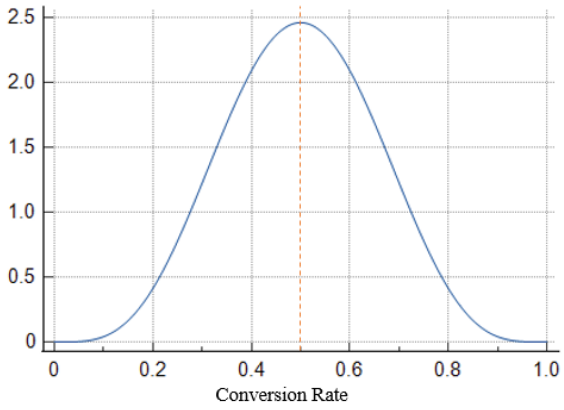
但是，A/B 测试的成本很高。在测试期间，更多的测试客户接触到的是效果不佳的广告版本，这让公司失去许多潜在客户，这些潜在客户可能会在看到优质广告时购买公司的商品，但是由于他们看到的是效果不好的广告，他们可能不会购买公司产品。此外，公司可能需要更长的时间才能选出最佳的广告版本。

有一种 Thompson 抽样的人工智能算法（有时也称为贝叶斯老虎机 Multiarmed Bandits）可以潜在地避免 A/B 测试的缺点。与 A/B 测试不同之处在于，Thompson 抽样算法动态地将更多客户分配给表现良好的广告，而将较少客户分配给表现不佳的广告。

Thompson 抽样的基本逻辑如下。一开始，AI 没有关于各个版本广告的效果的任何信息。所以 AI 简单假设每个版本广告的转化率是相同的。然后 AI 随机选择一个版本的广告并将其投放给客户。如果客户在看到这个版本的广告后购买了产品，那么 AI 会逐渐更多的去投放这个版本的广告给下一个客户。如果客户在看到广告后没有购买商品，那么 AI 会降低给下个客户展示该版本广告的概率。

为了实现这个逻辑，AI 需要在观察客户的购买结果后，根据已有数据，更新他对每个版本的广告的转化率的估计。为此，AI 假设每个广告的转化率服从 Beta 分布$(\alpha+1, \beta+1)$，其中α是购买产品的客户数量，β是不购买产品的客户数量 (这里+1 因为 beta 分布的参数必须大于零)。例如，假设有 4 个客户看到一个版本的广告后购买了公司产品，4 个客户没有购买该产品，那么 AI 会假设这个版本广告的转化率服从 Beta(5,5)分布，他的概率密度函数如下图所示。从图中我们可以观察到，这个版本广告的转化率是 0.5 的可能性最大（橙色虚线）。因此，AI 可以从 $Beta(\alpha+1, \beta+1)$中产生一个随机数，这个随机数是 AI 基于现有证据对该版本广告的转化率的一个预估。

图片. Beta(5,5)的概率密度函数



综上，Thompson 抽样的步骤如下。

对每位顾客，重复：

步骤 1：对于每个版本的广告，AI 从 `Beta(α+1, β+1)`中产生一个随机数，当做此版本广告的`预估转化率`，其中`α`是看到此版本广告的客户中购买产品的人数，`β`是看到这个版本广告但未购买产品的客户数量。

步骤 2：AI 向客户投放`预估转换率`最高的那个版本的广告。

步骤 3：如果客户购买了产品，则该版本广告的`α`加 1。 如果顾客没购买产品，则将此版本的广告的`β`增加 1。

在这次项目中，我们希望能够制作一个广告投放 AI，他的主要功能是用 A/B 测试或 Thompson 采样的方式，选择最佳的广告版本。

开始

- 从课程中心平台 Canvas 上下载 `week6 不确定性`单元中的 `week6_project.zip` 并且解压缩
- 当处于本项目文件所在的工作目录中时，在终端上运行 `pip3 install -r requirements.txt` 用来安装这次项目需要的 Python 包。

理解项目的相关文件

这个项目主要包含两个文件：`customer.txt` 和 `Advertising.py`。

首先打开 `customer.txt`。这个文件内容是顾客看到不同版本广告后，是否会购买产品。文件中每行代表一个客户，每列代表一个广告版本。`1` 表示如果客户被投放了这个版本的广告，他会购买该产品，而 `0` 表示客户不会购买该产品。 例如，我们观察到第 `1` 行第 `7` 列是 1，这意味着如果向第 1 个客户展示第 7 个版本的广告 ，那么他将会购买该产品。 现实中，公司无法向同一位客户展示所有版本的广告。 所以这个文件只是模拟，假设客户被展示给任一版本的广告，他会不会购买公司的产品。

其次，打开 `Advertising.py`。在这个代码中，我们定义了一个类 `AB_testing`，是我们将要制作的广告投放 AI，可以用 A/B 测试的方式找到最优的广告版本。在这个类中，`__init__`函数，以及 `draw` 函数已经为你写好了，无需变动。同时，我们还定义了一个继承自 `AB_testing` 的类 `Thompson_sampling`，他是用 Thompson 采抽样的方式找到最优广告版本。

`__init__`函数将客户的购买结果数据存储在 `self.result`，其中这个字典的键(key)代表不同广告版本，`buy` 代表看了这个版本广告后购买产品的人数（`α`），`not_buy` 代表看了这个广告后没购买产品的人数（`β`）。`self.cvr` 记录

了每当观察到一个新的客户数据后，现有的各个版本广告的转换率，是为了之后画图表示顾客数量与两种方法的表现时更方便呈现。`__init__`函数还读取了`customer.txt`文件，记录了测试客户的数量以及一共有几个广告版本。

`draw`函数根据已测试的客户数量绘制九个版本策略的预估转化率。 通过对比 A/B 测试和 Thompson 抽样算法的结果图，我们可以注意到 Thompson 抽样算法可以在大约 500 名客户后找出最佳广告版本，比 A/B 测试（大约 1000 名客户）要早得多。

剩余的三个函数，`sample`, `update_result`和`best`函数，将留给同学们你来完成。

`week3_project.zip`中还包含`autograde`文件夹，里面包含测试代码的相关文件。

要求

类`AB_testing`中的`sample`函数

- 输入：无
- 功能：随机抽取将要展示给顾客看的广告版本，每个广告版本被选择的概率是相等的。
- 输出：一个从`0`到`8`的数字，每个数字对应一个广告版本

`update_result`函数

- 输入：无
- 功能：根据观察到的顾客购买情况，更新`self.result`和`self.cvr`。
  - 用`sample`函数，为每一个顾客选择一个广告版本，观察顾客是否会购买公司的产品。
  - 根据观察到的购买情况，更新`self.result`中对应的广告版本的`'buy'`或`'not_buy'`数据。
  - 根据观察到的购买情况，计算每次收到新数据后的每个版本的转换率，将其添加在`self.cvr`中对应的广告版本的列表内。
    - 转换率为购买的顾客数量除以看到这个版本广告的顾客总数量
    - 如果一个版本的广告没有投放给任何顾客，那么当前这个版本广告的转换率为 0
    - 假如已经投放给`N`个顾客，那么`self.cvr`中每个版本对应的转化率列表都需要有`N`个转换率
      - 假设顾客被投放的是版本`0`，在知道这个顾客的购买情况后，你的函数也需要计算其他广告版本`1-8`的转化率，并将他们添加到`self.cvr`中。
- 输出：无

`best`函数

- 输入：无
- 功能：在观察到所有顾客的购买情况后，选择最优的广告策略
- 输出：一个从`0`到`8`的数字，每个数字对应一个广告版本

类`Thompson_sampling`中的`sample`函数输出将要展示给顾客看的广告版本。

- 输入：无
- 功能：按照 Thompson 抽样的方法抽取将要展示给顾客看的广告版本
- 输出：一个从`0`到`8`的数字，每个数字对应一个广告版本

你不应该修改 `Advertising.py` 中除了 `sample`, `update_result` 和 `best` 函数之外的已经写好的其他部分，但是你可以添加新的函数。如果你熟悉 `numpy` 或者 `pandas`，你也可以使用这两个包，但是你不可以使用其他的第三方包。

提示

你可以查看 Python 官网的 random 包，里面的方程有助于产生随机数，无论是等概率随机数，还是服从 Beta 分布的随机数。

测试代码

- 你可以使用代码 `pytest autograde/autograde.py --tb=no` 自行测试自己的代码是否满足要求。您需要安装 `requirements.txt` 中的 `pytest` 包。
- 请先确保你的程序能够成功运行并输出结果。请确保你的工作目录中包含 `Advertising.py`。