概率 & 期望入门

black_trees

2023 年 4 月 1 日

Chengdu Foreign Languages School

前言

当你学习概率论时，不要相信你的直觉。

基本概念
样本点 & 样本空间

- 样本点 ω (sample point): 一次随机试验 E 中的每一个可能出现的试验结果。可以是抽出的卡牌编号，也可以是扔出的骰子点数。
- 样本空间 Ω (sample space): 所有样本点构成的集合。

- 随机变量 X (random variable)：一个由样本空间到实数域的**单射**。

因为样本点可能是用文字表示的，比如硬币正面朝上或是反面朝上，所以为了研究方便，我们建立一个映射，将样本点转化为实数。
注意，随机变量是一个函数，它不对应样本点，它的取值才对应样本点。
上面的抛硬币问题的随机变量 X 的取值为 $\{0,1\}$，所以 X 是一个映射：$\Omega \to \{0,1\}$。
通常我们研究的是离散型随机变量，也就是取值可数的随机变量。

基本概念

随机事件

- 随机事件 S (random event)：样本空间 Ω 的一个子集，表示一系列结果。
- 和事件 $A \cup B$ (sum event)：任意两个事件的和事件 $A \cup B$ 如果发生，当且仅当 A, B 中有一个事件发生。可以理解为并集。
- 积事件 $A \cap B$ (product event)：任意两个事件的积事件 $A \cap B$ 如果发生，当且仅当 A, B 同时发生。可以理解为交集。
- 互斥事件：若对于任意两个事件 A, B，它们的积事件发生的概率为 0，则称这两个事件互斥 (可以表达为 $P(A \cap B) = 0$)

和事件以及积事件都可以扩展到多个事件的情况，不再赘述。

古典定义：
如果一个试验 E 满足两条性质：

- 试验的样本空间是**可数**的（即，样本空间只包含有限个样本点）。
- 试验的每个样本点等可能出现。

这样的试验便是古典试验。
对于古典试验中的事件 A，它的概率定义为 $P(A)=\frac{m}{n}$，其中 n 表示该试验中所有可能出现的基本结果的总数目，m 表示事件 A 包含的试验基本结果数。

公理化定义:
设样本空间为 Ω，若对于 Ω 中的每一个事件 A，都存在定义在实数域上，值域为实数的函数 $P(A)$，满足:

- $P(A) \geq 0$
- $P(\Omega) = 1$
- 对于若干个**两两互斥**事件 $A_1, A_2, \ldots$，有 $\sum P(A_i) = P(\bigcup A_i)$

则称 $P(A)$ 为随机事件 A 发生的概率 (probability)。
其实概率还有一种统计定义，不过目前没啥用，我提一下就行了。

- 条件概率：对于任意两个事件 A, B，在已知事件 A 发生的条件下，发生 B 事件的概率称作条件概率，记作 $P(B|A)$。
- 计算公式：$P(B|A) = \dfrac{P(A \cap B)}{P(A)}, P(A \neq 0)$。

- 事件的独立性：若事件 A 是否发生对 B 没有影响，即是 $P(B|A) = P(B)$，则称事件 A, B 相互独立。

概率

相关计算

Sum Rule：
定义里已经提到，事件可以看作一个集合，所以我们也可以用集合的思想来考虑概率问题。
由容斥原理可以得到：

$$P(A \cup B) = P(A) + P(B) - P(A \cap B)$$

还记得我们提到的互斥事件吗？它们的积事件发生的概率为 0，我们带入容斥原理：

$$P(A \cup B) = P(A) + P(B)$$

换句话说，如果两个事件互斥，则它们的和事件发生的概率为它们分别发生的概率之和。
推广：若事件 $A_1, A_2, \ldots, A_n$ 两两互斥，则

$$P(\bigcup A_i) = \sum P(A_i)$$

可以发现这就是概率的公理化定义。

Multiplication Rule:
还记得条件概率的公式吗?

$$P(B|A) = \frac{P(A \cap B)}{P(A)}$$

我们想想，如果 A, B 相互独立，那么这个公式会长成什么样呢?

$$P(B|A) = P(B) \iff P(A) \cdot P(B) = P(A \cap B)$$

换句话说，如果两个事件相互独立，则它们的积事件发生的概率为它们分别发生的概率之积。
推广：若事件 $A_1, A_2, \ldots, A_n$ 两两独立，则

$$P(\bigcap A_i) = \prod P(A_i)$$

Bayes Rule:
若 A, B 相互独立，则有：

$$P(B|A) = \frac{P(B)P(A|B)}{P(A)}$$

推导是显然的，代入 Multiplication Rule 即可。

Total Probability Theorem:

若事件 $A_1, A_2, \ldots, A_n$ 构成一组完备的事件且都有正概率，即 $\forall i, j, A_i \cap A_j = \varnothing$ 且 $\sum_{i=1}^{n} A_i = 1$，则有:

$$P(B) = \sum_{i=1}^{n} P(A_i)P(B|A_i)$$

证明也比较容易，拆开之后代入条件概率公式即可。

- 数学期望/均值 E (mean)：若离散型随机变量 X 的取值为 $\{x_1, x_2, x_3, \ldots\}$，其中随机事件 $X = x_i$ 发生的概率为 $P(X = x_i) = p_i$，那么，离散型随机变量 X 的数学期望 $E(X)$ 为 $\sum x_i p_i$。

换句话说，一个离散型随机变量的期望，等于其取值和对应概率之积的和。
很多时候，我们会将数学期望看作一种“加权平均数”，这样更加易于理解。

期望
性质

- 线性性：$E(aX+bY)=aE(X)+bE(Y)$。
- 常用形式：$E(X+Y)=E(X)+E(Y)$。
- 与概率的相互转化：这个其实不常用，省去。

期望的线性性在求解期望 dp 的时候非常有用，通常，我们可以利用它，省去直接按定义展开式子的繁琐步骤。

就是设计一个状态，求一些事件发生的概率。
大致分三步：

1. 找到样本点，样本空间，事件。
2. 设计合适的状态。
3. 考虑利用 Sum/Multiplication Rule 来转移。

Talk is cheap, show me the problem.

Zelda 有 n 张牌，每张牌上有一个数字 $a_i, 1 \le a_i \le n$。
Zelda 每次会从牌堆里抽一张牌，抽完之后不放回。
假设当前抽到 x，上一张抽出的是 y。

- 如果 $x = y$，win。
- 如果 $x < y$，lose。
- 如果 $x > y$，continue。

问 Zelda 最后获胜的概率是多少，对 998244353 取模，$1 \le n \le 5000$[1]。

[1] Source: CF1156F. Card Bag

发现要处理的要素有点多:

- 抽完不放回去
- 每次可能有三种游戏状态：win,lose,continue。
- 每种牌的数量

第一个直接记录当前抽了几次就行了。
第二个也比较好做，lose 就直接退出了，对于最后的答案不会有贡献，continue 需要设计状态表示，发现 win 只需要考虑最后两个牌相同的情况就行了，所以也不用设计状态，最后统一计算贡献即可。
那么，我们只需要处理 continue 的情况。
最麻烦的部分是处理每种牌剩余的数量，感觉上来说需要多开一维状态来记录，但这样状态承受不了，也不好转移。
但其实不用，因为有一个性质。

性质：
发现如果游戏能继续下去，当且仅当取出来的数按照时间戳排序之后是**严格单调递增**的。
所以，每种牌是只能取一张的，也就是说，抽到一张牌 j 的时候，一定是第一次，并且是最后一次抽到它。

所以可以设计出 DP ：设 $dp(i,j)$ 表示考虑第 i 次，抽到 j 且继续下去的概率。
因为要保证一直继续，所以转移的时候不能从上一个取大于 j 的状态转移过来，又要保证需要继续下去，所以也不能从上一个取 j 的转移。
那么可以写出方程：

$$dp(i,j)=\sum_{k=1}^{j-1} dp(i-1,k)+\frac{cnt(j)}{n-i+1}$$

前缀和优化一下，能够做到 $O(n^2)$。

在求出 dp 数组之后，我们考虑如何计算答案。
发现如果 $cnt(j) > 1$，那么状态 $dp(i, j)$ 对于答案的贡献就是 $\frac{cnt(j) - 1}{n - i}$。
因为我们之前取到了一个 j，所以我们需要在 $dp(i, j)$ 的基础上乘一个再次取到 j 的概率。
又因为所有胜利的情况是互斥（互不影响）的，所以最后的答案直接求个 $\sum$ 即可。

概率 dp 的话，其实只需要想清楚事件是和事件还是积事件，设计好状态，用定义转移就行了，没啥难度。

期望 dp 和概率 dp 的区别就在于，期望 dp 还需要考虑取值问题。所以一般情况下，期望 dp 的处理都比概率 dp 稍要复杂一些，不过其实也不算太难。

期望 DP
例题 1

有一个 n 滴血的怪物。每一次攻击你有 $P\%$ 的概率让它失去 2 滴血，有 $(100-P)\%$ 的概率让它失去 1 滴血。
如果攻击过后怪物的血量 ≤ 0，它就死了。你需要一直攻击怪物直到它死亡。输出攻击次数的期望对 998244353 取模的值[2]。
$1 \leq n \leq 2 \times 10^5, 0 \leq P \leq 100$。

[2] Source: ABC280E. Critical Hit

期望 DP
例题 1

设 $dp(i)$ 表示打死一只血量为 i 的怪物的步数期望。
因为血量为 1 的时候怎么打都是 gg，所以 $dp(1)=1$。
方程比较显然，如果当前打掉了 1 滴血，那么应当从 $dp(i-1)$ 转移过来，概率为 $\dfrac{100-P}{100}$，攻击次数加一。
否则从 $dp(i-2)$ 转移过来，概率为 $\dfrac{P}{100}$。
可以得到方程：

$$dp(i)=\frac{100-P}{100}dp(i-1)+\frac{P}{100}dp(i-2)+1$$

我们这里省去了直接代入期望的定义的过程，只需要乘法分配律一下就可以知道式子的正确性了。
你可以把上一个状态**看作**这个状态的一个取值，不过其本质还是推式子。

给出一个有向无环的连通图，起点为 1，终点为 N，每条边都有一个长度。
数据保证从起点出发能够到达图中所有的点，图中所有的点也都能够到达终点。
Yuyuko 从起点出发，走向终点。
到达每一个顶点时，如果有 K 条离开该点的道路，Yuyuko 可以选择任意一条道路离开该点，并且走向每条路的概率为 $\frac{1}{K}$。
现在 Yuyuko 想知道，从起点走到终点所经过的路径总长度的期望是多少？[3]
$1 \le n \le 10^5, 1 \le m \le 2 \times 10^5$。

[3]Source: Acwing217. 绿豆蛙的归宿

期望 DP

例题 2

设 $dp(u)$ 表示从 $1 \to u$ 的路径长度的期望。
我们根据定义尝试推一下式子，假设 u 是从 $v_1 \ldots v_r$ 这些节点可以到达的。
假设 $x(v_i)_j$ 表示从 $1 \to v_i$ 的路径的某一种可能长度，$p(v_i)_j$ 表示出现 $x(v_i)_j$ 这种情况的概率。
那么就有：

$$
\begin{aligned}
dp(u) &= \sum_i \sum_j (p(v_i)_j \times \frac{1}{deg(v_i)} \times (x(v_i)_j + w(u, v_i))) \\
&= \sum_i (\frac{1}{deg(v_i)} \times \sum_j (p(v_i)_j \times x(v_i)_j + p(v_i)_j \times w(u, v_i))) \\
&= \sum_i (\frac{1}{deg(v_i)} \times (dp(v_i) + w(u, v_i) \times \sum_j p(v_i)_j))
\end{aligned}
$$

注意到 $\sum\limits_j (p(v_i)_j)$ 就是从 1 出发，走到 v_i 的概率。
所以我们转移 dp 的时候顺带着处理一个数组 $P(v_i)$ 表示从 $1 \to v_i$ 的概率就行了。
初始化 $dp(1) = 0$，终态 $dp(n)$。

其实这种“正推”的方法比较麻烦，其原因在于需要处理 P 这个数组，因为从 $1 \to v_i$ 的概率是不能直接确定的。
注意到我们最后一定会走到 n，也就是说 $\forall u \neq n, P(u \to n) = 1$，那么，如果我们考虑倒推会怎样呢？
设 $dp(u)$ 表示从 $u \to n$ 的路径长度期望，并且 u 可以到达 $v_1 \dots v_r$，那么可以有：

期望 DP

例题 2

$$dp(u) = \frac{1}{deg(u)} \sum_i (dp(v_i) + w(u, v_i) \times \sum_j p(v_i)_j)$$

是不是基本一样？并不，注意到这里的 $\sum_j p(v_i)_j$ 表示的是从 v_i 出发走到 n 的概率，这个概率一定是 1，所以状态转移变为：

$$dp(u) = \frac{1}{deg(v_i)} \sum_i (dp(v_i) + w(u, v_i))$$

然后转移就不用额外维护信息了。

- 求解期望 DP 时，可以将上一个状态**当作**下一个状态的随机变量的取值，进而简化转移。
- 其本质类似：$\sum(x_i p_i \times P) = P \times \sum(x_i p_i)$。
- 当然，也可以直接暴力用定义展开求解，例题 2 就展示了这样的过程，不过一般不推荐使用这种做法。
- 很多时候，正推需要额外处理概率，在确定终态的情况下，我们可以考虑倒推，借此来省去处理概率的繁琐步骤。

当然，期望 DP 不有止这些技巧，习题当中还有一个更有意思的 trick。

习题

- CF148D. Bag of mice
- CF518D. Ilya and Escalator
- CF1778D. Flexible String Revisit
- ABC295E. Kth Number
- CF24D. Broken Robot （需要高斯消元）
- HNOI2013. 游走（需要高斯消元）

后两题推出方程后会有后效性，但是利用高斯消元，将转移方程当作几个线性方程组来求解，可以消除后效性。
这两题不做要求，感兴趣可以自己研究一下，不懂可以问我。
以上题目会放在 Vjudge 专题/洛谷团队上，链接：link

答疑

尽管问！

后记

Reference

- Mathematics for Computer Science
- Introduction to Algorithms
- 算法竞赛进阶指南
- 我曾经的高中数学老师 Cy 的讲义
- 我的博客：hylwxqwq.github.io (这算引用吗 (逃))
- 感谢 Zy 同学和 Wkm 学长，他们提出了许多宝贵意见。

后记
吐槽

- 写这玩意儿挺占时间的。
- 感觉很多书讲定义都没有很清晰的逻辑体系。
- 所以这套逻辑是我自己搞出来的，应该在其他地方见不到！（是的，这种感觉很好）
- 如有谬误，请联系：QQ: 1020061231.
- 如需转载，请注明原作者。
- SCOI2023 RP++
- 更正发布地址：here