Листок 6

Ещё одна попытка ликбеза

Центральным объектом для понимания как теории вероятностей в целом, так и (особенно) теории вероятностей в приложении к теории алгоритмов является понятие случайной величины.

Случайная величина

– не является величиной – это функция,

– не является случайной – это детерминированная функция.

Вероятностное пространство

Случайная величина придумана для того, чтоб не думать о вероятностном пространстве. А вероятностное пространство придумано, чтоб обосновать введение случайной величины.

Вероятностное пространство – это тройка $(\Omega, \mathcal{B}, \mathsf{P})$.

Ω – это просто множество, абстрактное множество "исходов". Часто, но далеко не всегда в алгоритмах хватает конечного множества.

$\mathcal{B}$ – это некоторая совокупность подмножеств множества Ω. Конкретно, это σ-алгебра. В случае конечного Ω можно взять множество всех подмножеств, в случае бесконечного множества это не всегда разумно.

P – это вероятностная мера. Это функция, которая каждому элементу $\mathcal{B}$ сопоставляет число. Необходимы всякие условия типа конечности и счётной аддитивности. В случае конечного Ω можно сопоставить числа всем одноэлементым подмножествам (фактически, элементам), а на все остальные подмножества мера распространится сама.

Вероятностное пространство формализует абстракцию "элементарных исходов", "результатов бросков" и тому подобного. Элементы Ω – это элементарные события, элементы $\mathcal{B}$ – это "хорошие" подмножества, которые называются событиями (а прочие подмножества мы не рассматриваем). P говорит нам, какое событие имеет какую вероятность.

Вся эта кухня позволяет нам делать вид, что мы работает с понятным и строго описанным математическим объектом. Проблемы начинаются, когда мы захотим что-то посчитать. Пусть дана обычная монетка – как она падает "в среднем"? Если вероятности орла и решки равны друг другу и равны 1/2, то в среднем монетка падает на ребро? Не падает? Падает "средне"? Может, вопрос не имеет смысла?

Чтобы что-то считать, что-то вычислять и использовать мощные аппараты анализа и теории чисел, нам нужны, собственно, числа. Случайная величина делает нам числа.

Морская свинка

Случайная величина ξ (обычно используют самые сложно рисуемые из греческих букв, либо заглавные буквы из конца латинского алфавита) это функция

$$\xi : \Omega \to \mathbb{R}$$

Рассматривают также случайные величины, действующие не в $\mathbb{R}$, но в этом курсе нам они не понадобятся. Кроме того, не каждая функция подойдёт, технически нам нужны измеримые функции, при этом на $\mathbb{R}$ предполагается борелевская σ-алгебра, что бы это ни значило.

Пусть $\Omega = \{\text{орёл}, \text{решка}\}$, пусть $\xi(\text{орёл}) = 0$, $\xi(\text{решка}) = 1$. Теперь имеет смысл спрашивать не только вероятности выпадения орла. Имеет смысл спрашивать о вероятности выпадения нуля. А именно, случайная величина "протолкнула" меру, заданную на абстрактном пространстве на вещественные числа. Имеет также смысл спросить, чему равна вероятность "выпадения" -1 или же "выпадения" какого-то числа из промежутка $[2, 14]$ (обе они равны нулю).

Формально введение случайной величины задало распределение вероятностей на $\mathbb{R}$.

В случае дискретного множества Ω (особенно в случае конечного) случайная величина ξ принимает счётный набор значений y_k. Тогда можно спросить, чему равна вероятность того, что $\xi = y_k$. По определению это $p_k = \mathsf{P}(\xi = y_k) = \mathsf{P}(\omega \in \Omega \,|\, \xi(\omega) = y_k)$.

В случае непрерывного множества Ω буквально такой же подход затруднителен, поэтому используют более общее определение: вероятность ищут не для каждого y_k, а для каждого "хорошего" (а именно, борелевского) подмножества $S \subset \mathbb{R}$

$$\mathsf{P}(\xi \in S) = \mathsf{P}(\omega \in \Omega \,|\, \xi(\omega) \in S)$$

Для дискретных Ω можно также использовать этот подход, а для непрерывных он просто необходим.

Этот набор вероятностей (конечный для конечного множества значений и бесконечный в более общем случае) и есть распределение случайной величины ξ. После того, как распределение задано, "костыль" в виде вероятностного пространства более не нужен. Вся необходимая информация о вероятностях хранится в распределении. Вернуться к вероятностному пространству полезно при определении зависимости или независимости двух различных случайных величин.

Примеры

1) ξ_1 принимает два значения: 0 и 1. При этом вероятность единицы есть p, а вероятность нуля, соответственно, $1 - p$ для некоторого $p \in [0, 1]$.

2) ξ_2 может принимать любое натуральное значение $\{1, 2, \ldots\}$. Вероятность принять значение k есть 2^{-k}.

3) ξ_3 может принимать любое вещественное значение. Вероятность принять любое конкретное значение при этом равно нулю, вероятность того, что ξ_3 находится в промежутке (в том числе бесконечном) (a, b) есть $\frac{1}{\sqrt{2\pi}} \int\limits_a^b e^{-\frac{x^2}{2}}\, dx$

После введения распределения, как уже было сказано, вероятностное пространство особого значения не имеет.

В примерах выше говорят, что ξ_1 имеет распределение Бернулли или является бернуллиевской случайной величиной с параметром p. ξ_2 имеет геометрическое распределение с параметром $1/2$, а ξ_3 распределена нормально (имеет нормальное распределение) с параметрами $(0, 1)$.

Если дано несколько случайных величин, они могут быть зависимыми или независимыми. В

алгоритмических приложениях нас чаще всего интересуют независимые величины (моделирующиеся независимыми запусками генератора случайных чисел).

Случайные величины $\xi_1, \ldots, \xi_n$ независимы в совокупности (другие понятия не потребуются), если

$$\forall y_1, \ldots, y_n \; \mathsf{P}(\xi_1 = y_1, \ldots, \xi_n = y_n) = \mathsf{P}(\xi_1 = y_1) \cdot \ldots \cdot \mathsf{P}(\xi_n = y_n)$$

Математическое ожидание

Матожидание случайной величины ξ – это число $\mathsf{E}\xi$, которое равно тому, чему равна случайная величина "в среднем".

Для дискретного распределения матожидание равно взвешенному среднему каждого из возможных исходов. Вес каждого исхода, что логично, – его вероятность.

Получаем $\mathsf{E}\xi = \dfrac{\sum\limits_k p_k y_k}{\sum\limits_k p_k}$. Поскольку знаменатель дроби равен единице, матожидание равно

$$\mathsf{E}\xi = \sum_k p_k y_k$$

В случае дискретного вероятностного пространства матожидание можно также записать в виде

$$\mathsf{E}\xi = \sum_k p_k y_k = \sum_{\omega \in \Omega} \xi(\omega) \mathsf{P}(\omega)$$

Для непрерывного распределения придётся брать интеграл Лебега ¯_(ツ)_/¯

В примерах выше $\mathsf{E}\xi_1 = p$, $\mathsf{E}\xi_2 = 2$, $\mathsf{E}\xi_3 = 0$.

Матожидание (поскольку это сумма или интеграл) – это **линейный** функционал (то есть линейный оператор из функций в числа):

$$\mathsf{E}(a\xi + b\eta) = a\mathsf{E}\xi + b\mathsf{E}\eta$$

Равенство верно для любых чисел и любых случайных величин (в том числе и зависимых).

Неравенства и закон больших чисел

Неравенство Маркова

Пусть ξ принимает только неотрицательные значения, а число $a > 0$. Тогда

$$\mathsf{P}(\xi \geqslant a) \leqslant \frac{\mathsf{E}\xi}{a}$$

$$\mathsf{P}(\xi \geqslant a\mathsf{E}\xi) \leqslant \frac{1}{a}$$

То есть, случайная величина сильно отклоняется от своего матожидания с **полиномиально** малой вероятностью.

Неравенство Чебышёва

Дисперсией случайной величины называется число $\mathsf{D}\xi = \mathsf{E}\left((\xi - \mathsf{E}\xi)^2\right) = \mathsf{E}\xi^2 - (\mathsf{E}\xi)^2$

Пусть ξ случайная величина с конечным матожиданием, а число $b > 0$. Тогда

$$\mathsf{P}(|\xi - \mathsf{E}\xi| \geqslant b) \leqslant \frac{\mathsf{D}\xi}{b^2}$$

$$\mathsf{P}(|\xi - \mathsf{E}\xi| \geqslant b\sqrt{\mathsf{D}\xi}) \leqslant \frac{1}{b^2}$$

Отсюда, в том числе, берётся “правило трёх сигм”.

Граница Чернова

Пусть $\xi_1, \ldots, \xi_n$ набор независимых одинаково распределённых случайных величин. Каждая ξ_i имеет бернуллиевское распределение с параметром p. Матожидание каждой из ξ_i есть p.

Пусть $\zeta = \xi_1 + \ldots + \xi_n$, тогда $\mathsf{E}\zeta = pn$ по линейности.

Тогда выполняется следующее

$$\mathsf{P}\left(\left|\frac{\zeta}{n} - p\right| \geqslant \delta\right) \leqslant 2e^{-2\delta^2 n}$$

То есть, среднее арифметическое большого количества независимых бернуллиевских случайных величин сильно отклоняется от своего матожидания с **экспоненциально** малой вероятностью.

Задача 6.1:

Вы бросаете правильную монету (вероятности орла и решки одинаковы) до тех пор, пока не выполнится определённое условие. Для каждого условия ниже найдите математическое ожидание числа бросков монеты.

1) Пока не выпадет решка.

2) Пока не выпадет две решки.

3) Пока не выпадет и орёл, и решка.

4) Пока в двух последовательных бросках не выпадут две решки подряд.

5) Пока в двух последовательных бросках не выпадут орёл, затем решка.

6) Пока в четырёх последовательных бросках не выпадут последовательно решка-орёл-решка-орёл.

7) Пока в четырёх последовательных бросках не выпадут последовательно решка-решка-орёл-орёл.

8) Пока суммарно число выпавших орлов не превысит число выпавших решек.

9) Пока суммарно число выпавших орлов не будет равно числу выпавших решек.

10) Пока в n последовательных бросках не выпадут n решек подряд.

Задача 6.2 (не так-то просто перемешать числа):

Рассмотрим алгоритм получения случайной перестановки. Необходимо получать любую из $n!$ перестановок независимо и равновероятно.

Следующий алгоритм использует функцию RANDOM(n), возвращающую независимо и равновероятно случайное натуральное число от 1 до n включительно.

```
1: procedure RandomPermutation(n)
2:     for i = 1 ... n do
3:         π[i] := i
4:     end for
5:     for i = 1 ... n do
6:         поменять местами π[i] и π[Random(n)]
7:     end for
8:     return π
9: end procedure
```

Докажите, что этот алгоритм работает **некорректно**.

Задача 6.3 (перемешаем числа хорошим способом):

Рассмотрим следующий алгоритм равномерного перемешивания колоды из n карт (это значит, что вероятность получить любое конкретное перемешивание одинакова):

1) возьмём колоду и пронумеруем карты сверху вниз от 1 до n;

2) далее работаем по шагам, на каждом шаге:

2.1) берём карту с вершины стопки,

2.2) в оставшейся колоде из $n-1$ карты равновероятно выбираем случайное место для карты (n способов – на верх колоды или под какую-то из карт колоды),

2.3) вставляем взятую карту в выбранное случайное место (карт в колоде снова n);

3) алгоритм заканчивает свою работу после того, как цикл сработал на карте номер $n-1$, т.е. карта номер $n-1$ была взята сверху и вставлена в случайное место.

а) Доказать, что представленный алгоритм работает корректно, т.е. перемешивает колоду равномерно.

б) Найти математическое ожидание количества шагов при выполнении алгоритма, т.е. числа вставок карты в колоду.

Задача 6.4 (монеты, и это нечестно!):

Вам дана нечестная монета – вы не знаете, насколько она нечестна, т.е. вы не знаете вероятности выпадения орла и решки при броске, вы знаете лишь, что обе эти вероятности отличны от нуля.

1) Придумать алгоритм, симулирующий бросок честной монеты – равновероятное получение единицы или нуля. В качестве единственного источника случайных значений вы можете использовать нечестную монету.

2) Вычислить математическое ожидание числа «бросков» нечестной монеты для вашего алгоритма.

Формально, вы моделируете бернуллиевскую случайную величину с параметром $1/2$ при помощи бернуллиевской случайной величины с неизвестным параметром $p \in (0,1)$.

Задача 6.5 (снова нечестная монета):

В предыдущей задаче "уровень нечестности" монеты не был вам известен. Но пусть вам известно, что при броске с вероятностью $p = 1/3$ выпадает решка, а с вероятностью $2/3$ – орёл.

Как и раньше необходимо придумать алгоритм, симулирующий бросок честной монеты – равновероятное получение единицы или нуля. Математическое ожидание числа бросков должно быть меньше, чем в предыдущей задаче.

Задача 6.6 (монеты, теперь честнее!):

Вам дана честная монета – вероятности выпадения орла и решки при броске одинаковы.

1) Придумать алгоритм, симулирующий бросок нечестной монеты с заданной вероятностью выпадения орла и решки. В качестве единственного источника случайных значений вы можете использовать честную монету.

2) Вычислить математическое ожидание числа «бросков» честной монеты для вашего алгоритма.

Формально, вы моделируете произвольную бернуллиевскую случайную величину с параметром $p \in (0,1)$ при помощи бернуллиевской случайной величины с параметром $1/2$.

Задача 6.7 (а вы когда-нибудь пробовали собрать все наклейки от жвачек?):

Вы собираете некие коллекционные объекты – карточки с картинками, наклейки, фишки, билеты, марки – что угодно. Существует n разновидностей этих объектов, ваша цель – собрать их все.

Сбор происходит следующим образом: вы покупаете один объект в закрытом конверте, распаковываете. Конечно же, может оказаться, что то, что вы распаковали, у вас уже есть – тогда покупка была неудачной. Предположим, что закрытые конверты абсолютно равнозначны – вероятность того,

что в конверте будет объект номер i, не зависит от i, не зависит от конверта и всегда равна $1/n$.

Может получиться, что вам повезёт и n покупок хватит для наполнения коллекции n различными объектами. Может получиться, что вам катастрофически не везёт и первые миллион, миллиард и т.д. конвертов содержат один и тот же объект (мы считаем, что объектов каждой разновидности в мире существует бесконечное количество). Найти математическое ожидание числа покупок конвертов, требуемое для получения хотя бы одного объекта каждой разновидности в вашей коллекции.

Задача 6.8 (когда монеты не хватает):

Вам дана игральная кость – вы не знаете, насколько она нечестна, т.е. вы не знаете вероятности выпадения каждого из чисел от 1 до 6 при броске, вы знаете лишь, что все эти вероятности отличны от нуля.

1) Придумать алгоритм, симулирующий бросок честной игральной кости – равновероятное получение каждого из исходов от 1 до 6. В качестве единственного источника случайных значений вы можете использовать нечестную игральную кость.

2) Вычислить математическое ожидание числа «бросков» нечестной игральной кости для вашего алгоритма.

Задача 6.9 (зависимость и независимость):

Вы с другом договорились о числе $p \in [0, 1]$. После этого ваш друг выбирает случайное число – с вероятностью p он выбирает 1, с вероятностью $1 - p$ он выбирает 0.

Вы хотите угадать это число и для этого вы выбираете число q и пишете программу, которая

1) с вероятностью q выбирает 1, с вероятностью $1 - q$ выбирает 0,

2) проверяет, равно ли выбранное число загаданному.

Если равно, число найдено, если не равно, алгоритм повторяет шаги 1 и 2.

Рассмотрим две постановки задачи

а) после неудачной попытки угадать ваш друг снова выбирает случайное число с вероятностью p;

б) после неудачной попытки число, загаданное другом, не изменяется.

Какое q вам нужно выбрать, чтоб минимизировать число попыток вашего алгоритма в среднем?

Задача 6.10 (вырвать слово из контекста):

На вход алгоритма подаётся поток данных. Каждый элемент потока – это некоторый объект заранее оговорённого вида (это может быть ip-адрес ботов, которые стучатся на сервер, или пакет данных, прошедших сквозь роутер). Необходимо получить случайный объект из входного потока неизвестной длины. При этом выбор объекта должен быть равновероятен.

Рассмотрим алгоритм

```
1: procedure GetSample(входной поток S)
2:     l := 0
3:     while поток не исчерпан do
4:         x := новый объект из потока
5:         l := l + 1
6:         if Random(l) = 1 then
7:             s := x
```

```
 8:             return s
 9:         end if
10:     end while
11: end procedure
```

Здесь RANDOM(n), как и в задаче 4.2, возвращает независимо и равновероятно случайное натуральное число от 1 до n включительно. По окончанию алгоритма в переменной l хранится длина входного потока (заранее неизвестная).

Как и большая часть онлайн-алгоритмов этот алгоритм возвращает некоторое значение на каждом шаге (после каждого объекта, пришедшего из потока), так что вывод алгоритма содержит последовательность из n объектов.

1) Доказать, что GETSAMPLE(S) на каждом шаге возвращает равновероятно случайный элемент из S (т.е. вероятность получить любой из элементов входного куска S длины k на шаге k есть $1/k$)

2) Найти математическое ожидание числа выполнений строки номер 7 при запуске алгоритма на потоке длины n.

3) Найти математическое ожидание значения переменной l, после того, как строка номер 7 выполнится первый раз.

4) Найти математическое ожидание значения переменной l, после того, как строка номер 7 выполнится второй раз.

5) Найти математическое ожидание значения переменной l, после того, как строка номер 7 выполнится последний раз.