

1. Фруктовая метрика (5)

Задача предоставлена Научно-образовательным центром когнитивного моделирования МФТИ

Вы работаете в лаборатории, разрабатывающей алгоритм для автоматической сортировки плодов на **Спелые** и **Неспелые**. Алгоритм оценивает вероятность того, что плод является **Спелым**. Чтобы понять, насколько хорошо алгоритм справляется с задачей классификации, вам нужно рассчитать метрику **Average Precision (AP)**. Эта метрика поможет оценить точность предсказания модели для класса **Спелый** при различных порогах вероятности от 0 до 1 ($[0, 1)$ с шагом 0.1).

Precision (точность) для бинарной классификации — это доля правильных положительных предсказаний среди всех предсказанных положительных классов. Формально, точность для порога t можно выразить как:

$$Precision(t) = \frac{TP(t)}{TP(t) + FP(t)}$$

где $TP(t)$ — это количество истинных положительных предсказаний (плоды классифицированы как **Спелые** и действительно **Спелые**), а $FP(t)$ — это количество ложных положительных предсказаний (плоды классифицированы как **Спелые**, но на самом деле они **Неспелые**).

Average Precision (AP) рассчитывается как среднее значение точности для различных порогов вероятности:

$$AP = \frac{1}{n} \sum_{i=1}^n Precision(t_i)$$

где t_i — это порог вероятности, а $Precision(t_i)$ — точность для предсказаний, сделанных при данном пороге t_i .

Формат ввода

Формат ввода

Массив вероятностей и истинных меток представлен в таблице:

№ Фотографии	Вероятность предсказания	Истинная метка
1	0.85	Спелый
2	0.55	Спелый
3	0.65	Неспелый
4	0.40	Спелый
5	0.95	Спелый
6	0.75	Неспелый
7	0.50	Спелый
8	0.60	Спелый
9	0.30	Неспелый
10	0.80	Спелый

Формат вывода

Вам необходимо написать значение посчитанного **Average Precision**, записанное через запятую (или точку) и округленное до двух знаков после запятой (точки).

Примечания

При пороге, например, 0.1 предсказание с вероятностью 0.1 считается **Спелым**.

2. Разделяющий эллипс (5)

Задача предоставлена Научно-исследовательским институтом системных исследований РАН при поддержке Российской нейросетевой ассоциации

Пусть два класса объектов заданы нормальными распределениями $X_1 \sim \mathcal{N}(\mu_1, \Sigma_1)$ и $X_2 \sim \mathcal{N}(\mu_2, \Sigma_2)$ в пространстве \mathbb{R}^2 с известными μ_1, Σ_1 и μ_2, Σ_2 . Определите длину большой полуоси эллипса, описывающего геометрическое место точек, расположенных равновероятно относительно обоих классов, т.е. описываемых уравнением:

$$P_1(x) = P_2(x),$$

где P_1 и P_2 - нормальные распределения, заданные теми же значениями μ_1, Σ_1 и μ_2, Σ_2 .

Дано:

Вектора средних $\mu_1 = (-2, 3)$ и $\mu_2 = (1, 0)$.

Матрицы ковариации

$$\Sigma_1 = \begin{bmatrix} 2 & 0 \\ 0 & 1 \end{bmatrix}$$

и

$$\Sigma_2 = \begin{bmatrix} 3 & -1 \\ -1 & 4 \end{bmatrix}$$

.

Найти:

Длину большой полуоси эллипса.

Формат вывода

Одно число — длина большой полуоси разделяющего эллипса, округлите до ближайшего целого числа.

3. Двойное блуждание по числовой прямой (5)

Двое друзей договорились встретиться в метро, которое представляет собой бесконечную целочисленную числовую прямую. Но они забыли договориться на какой станции встретиться, и оказались на станциях с координатами 0 и 10. Каждую минуту они одновременно и независимо перемещаются в соседнюю станцию в случайном направлении (оба направления равновероятны). Найдите вероятность, что друзья встретятся за не более чем 100 минут, то есть окажутся на одной и той же станции в один и тот же момент времени.

Формат вывода

Введите одну десятичную дробь — значение вероятности, округлённое до 3 знаков после запятой (или точки). Пример: 0, 123 или 0.123.



4. Минимизация MSE (5)

Вы пытаетесь предсказать случайную величину $Y = X^3$, используя признаки $Y_1 = X$ и $Y_2 = X^2$. Используя линейную модель (без bias), какое минимальное значение может принять MSE (Mean Squared Error) получившейся модели, если значения X равномерно распределены по $[0, 1]$?

Формат вывода

Введите одну несократимую дробь — значение MSE. Пример: $2/5$

5. Комбинации для геймера (5)

Задача предоставлена лабораторией инноватики МФТИ

Алексей разрабатывает собственную компьютерную игру. У персонажа в компьютерной игре ровно 100 способностей. Алексей хочет, чтобы каждая способность k вызывалась уникальной последовательностью s_k кликов на левую (0) и правую (1) кнопки мыши. Чтобы игровой движок однозначно понимал, какую способность хочет использовать игрок, необходимо, чтобы ни одна последовательность, соответствующая некоторой способности, не была префиксом последовательности, соответствующей другой способности. Какое наименьшее суммарное количество символов понадобится Алексею, чтобы закодировать все 100 способностей с таким свойством?

Более формально, пусть $A = \{s_1, \dots, s_{100}\}$ — некоторое множество слов из нулей и единиц, причём для любых $k \neq m$ слово s_k не является префиксом слова s_m . Найдите минимальное возможное значение суммы длин s_k по всем $k = 1, \dots, 100$.

Формат вывода

В ответ напишите одно целое число — минимальное возможное значение суммы, которым может обойтись Алексей.

6. Квантизация распределения Лапласа (7)

Ограничение времени	1 секунда
Ограничение памяти	256Mb
Ввод	стандартный ввод или input.txt
Вывод	стандартный вывод или output.txt

Задача предоставлена Научно-исследовательским институтом системных исследований РАН при поддержке Российской нейросетевой ассоциации

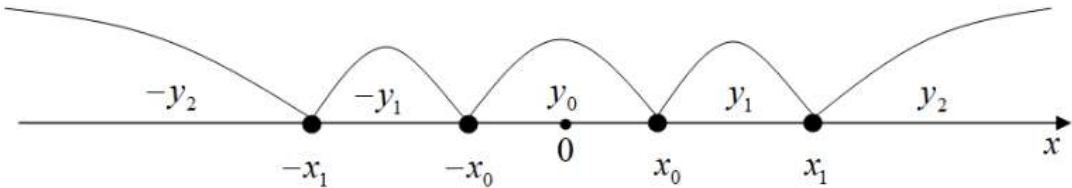
В последнее время большое внимание уделяется тому, как сделать большие нейронные сети менее требовательными к памяти и вычислительным ресурсам. Особенно это касается больших языковых моделей, объём которых занимает гигабайты. Популярным подходом для сжатия нейронных сетей является понижение разрядности весов, в том числе бинаризация или квантование весов. Суть квантования в том, чтобы заменить веса нейронных сетей из формата float32 на некоторое дискретное множество, например из пяти или трёх весов (к примеру, такой же приём с тремя весами использовался в недавней статье «The Era of 1-bit LLMs: All Large Language Models are in 1.58 Bits»). Задача состоит в следующем. Предположим, у нас есть некоторый слой нейронной сети, веса в котором имеют симметричное распределение Лапласа с нулевым средним:

$$p(x) = \frac{1}{2\lambda} e^{-\frac{|x|}{\lambda}}, \lambda > 0.$$

Для заданных порогов (концов отрезков квантования) x_0 и x_1 необходимо посчитать оптимальные значения квантованных весов y_0, y_1 и y_2 , такие, что при симметричном квантовании весов:

$$y = \begin{cases} y_2, & \text{if } x > x_1, \\ y_1, & \text{if } x_0 < x \leq x_1, \\ y_0, & \text{if } |x| \leq x_0, \\ -y_1, & \text{if } -x_1 \leq x < -x_0, \\ -y_2, & \text{if } x < -x_1, \end{cases}$$

корреляция (коэффициент корреляции Пирсона) между исходными весами x и финальными (квантованными) y была максимальная.



Формат ввода

Формат ввода

В первой строке значение $\lambda > 0$ - вещественное число типа float с 5 знаками после запятой.
Во второй строке значения $x_0 > 0$ и $x_1 > 0$, $x_0 < x_1$ через запятую - вещественные числа типа float с 5 знаками после запятой.

Формат вывода

Выведите две строки:
Во первой строке - значение максимальной корреляции Пирсона ρ_{max} - число в формате float с 5 знаками после запятой.
В второй строке - оптимальные значения $y_0, y_1, y_2 = \underset{y_0, y_1, y_2}{\operatorname{argmax}} \rho(x, y)$ через запятую - числа в формате float с 5 знаками после запятой. Нормируйте значения y_0, y_1, y_2 так, чтобы сумма их квадратов равнялась 1.

Пример 1

Ввод	Вывод
1.00000	0.77180
2.22489,4.25478	0.00000,0.48547,0.87425

Пример 2

Ввод	Вывод
1.00000	0.72613
2.50438,4.53978	0.00000,0.50002,0.86601

Пример 3

Ввод	Вывод
1.00000	0.74316
0.09746,8.73671	0.00000,0.11185,0.99373

7. Восстановить ядро свертки (7)

Ограничение времени	10 секунд
Ограничение памяти	256Mb
Ввод	стандартный ввод или input.txt
Вывод	стандартный вывод или output.txt

Задача предоставлена Научно-исследовательским институтом системных исследований РАН при поддержке Российской нейросетевой ассоциации

Пусть дана чёрно-белая картинка и результат операции свёртки над ней некоторым ядром. Нужно определить веса ядра свёртки.

Размер окна свёртки известен $h \times w$. Известно что свёртка производилась в режиме `mode='same'` - это значит что к картинке добавляется паддинг из нулей с каждой из сторон слева и справа шириной $(w - 1)/2$, а также сверху и снизу шириной $(h - 1)/2$. Считаем, что w и h - нечётные. Таким образом результат свёртки совпадает по размеру с размером исходной картинки. Значения пикселей в исходном изображении и в ядре свёртки целочисленные. Известно, что для приведённых в задаче тестов решение единственно.

Формат ввода

В первой строке написано два числа m, n , такие что $1 \leq m, n \leq 10^3$ — размер исходной картинки и результата свёртки.

В следующих m строках записаны элементы матрицы по n целых чисел, $-1000 \leq a_{ij} \leq 1000$ через запятую в каждой строке — значения пикселей исходной картинки.


В следующей строке написано два целых нечётных числа h, w , такие что $3 \leq h, w \leq 11$ — размер ядра свёртки, причём размер окна свёртки не больше размеров картинки $h \leq m$ и $w \leq n$


В следующих m строках записаны элементы матрицы по n целых чисел, $-100000 \leq b_{ij} \leq 100000$ через запятую в каждой строке — значения пикселей результата свёртки.

Формат вывода

Выведите h строк, в каждой строке по w целых чисел через запятую - значения весов свёртки.

Пример 1

Ввод 

Вывод 

Формат вывода

Выведите h строк, в каждой строке по w целых чисел через запятую - значения весов свёртки.

Пример 1

Ввод	Вывод
3,5	6,1,2
9,0,46,57,49	15,11,11
19,23,41,23,88	-15,12,-10
47,71,92,8,86	
3,3	
256,1096,1586,2501,1300	
1135,1014,1028,2221,1341	
1465,2149,1830,936,1860	

Пример 2

Ввод	Вывод
6,4	3,-2,-1
68,74,98,40	14,-5,-4
82,13,45,54	14,6,15
83,51,88,89	
91,71,7,40	
31,1,96,82	
22,97,63,17	
3,3	
571,757,-167,-745	
1203,3152,2777,994	
1004,2371,1858,115	
1692,2417,2833,1366	
1646,3077,2159,-546	
1448,2124,1274,1595	

8. Обходы графа (7)

Ограничение времени	1 секунда
Ограничение памяти	256Mb
Ввод	стандартный ввод или input.txt
Вывод	стандартный вывод или output.txt

Ваня хочет обойти ориентированный граф, начав в первой вершине и закончив в ней. При этом он хочет совершить переход ровно по k ребрам. Сколькими способами Ваня может обойти граф при таких условиях? Два способа считаются различными, если последовательности посещённых вершин не совпадают (например последовательности 1 1 2 1 и 1 2 1 1 - различны, а последовательности 1 2 1 1 и 1 2 1 1 - совпадают). Так как это число может быть достаточно большим, выведите его по модулю 1000000007.

Формат ввода


В первой строке находятся два числа n и k ($1 \leq n \leq 10$; $2 \leq k \leq 10^9$) - количество вершин и длина пути.

В следующих n строках находится по n чисел - матрица смежности графа. Формально если число в строке i и столбце j это 1, то есть ребро из вершины i в вершину j , иначе это число равно 0 и такого ребра нет.


Формат вывода

Выведите одно число - количество способов по модулю 1000000007.

Пример 1


Ввод 


2 3
1 1
1 0

Вывод 


3


Пример 2

Ввод 

Вывод 

Пример 2

Ввод 

Вывод 

```
4 100
1 0 1 0
1 1 1 1
0 1 1 0
1 1 0 1
```

538717459

Примечания

В первом примере возможны следующие варианты обхода:

- 1 → 1 → 1 → 1
- 1 → 2 → 1 → 1
- 1 → 1 → 2 → 1

9. Наклон переменной (7)

Ограничение времени	1 секунда
Ограничение памяти	256Mb
Ввод	стандартный ввод или input.txt
Вывод	стандартный вывод или output.txt

Задача предоставлена Научно-образовательным центром когнитивного моделирования МФТИ

Пусть прямой проход по абстрактной нейронной сети представлен следующим математическим выражением:

$$Y = \text{softmax}(Z)V,$$

где Z — тензор размерности $[1, m]$ с элементами z ;

$\text{softmax}(Z)$ — поэлементная операция $\frac{e^{z_{1i}}}{\sum_{j=1}^m e^{z_{1j}}}$ для $i = 1, 2, \dots, m$;

V — тензор размерности $[m, m]$ с элементами v ;

Y — тензор размерности $[1, m]$ с элементами y .

При условии, что $L = \sum_{i=1}^m y_{1i}$, найдите значения $\frac{\partial L}{\partial Z}$.

Формат ввода

В первой строке записано целое число $1 < m < 10$.

Во второй строке элементы тензора Z — по m вещественных чисел $\in [0, 1)$ через пробел в каждой строке.

В следующих m строках записаны элементы тензора V — по m вещественных чисел $\in [0, 1)$ через пробел в каждой строке.

Формат вывода

Элементы тензора $\frac{\partial L}{\partial Z}$ — m вещественных чисел через пробел. Ваш ответ будет засчитан, если погрешность не будет превышать 10^{-5} . Формально, если a - ваш ответ, а b - ответ жюри, то a будет засчитан, если $|a - b| \leq 10^{-5}$.

Пример

Ввод



Вывод



```
2
0.6732252240180969 0.7077440023422241
0.9007378220558167 0.8930410742759705
0.855198323726654 0.2551969289779663
```

```
0.1707950383424759 -0.1707950532436370
```



10. Generative Flow Networks (7)

Ограничение времени	2 секунды
Ограничение памяти	256Mb
Ввод	стандартный ввод или input.txt
Вывод	стандартный вывод или output.txt

Эта задача посвящена моделям **Generative Flow Networks (GFlowNets)**, которые являются одним из методов генерации дискретных объектов.

Формально задача генерации дискретных объектов формулируется так: у нас есть конечное множество объектов \mathcal{X} и неотрицательная reward функция $\mathcal{R}(x)$, определенная для всех $x \in \mathcal{X}$. Задача состоит в том, чтобы построить и обучить модель, которая генерирует объекты из \mathcal{X} из вероятностного распределения $\mathcal{P}(x) = \frac{1}{Z} \mathcal{R}(x)$ для $Z = \sum_{x \in \mathcal{X}} \mathcal{R}(x)$.

Генеративный процесс с помощью GFlowNet может быть рассмотрен как последовательность действий, при которой мы стартуем из некоторого пустого объекта и на каждом шаге добавляем некоторый компонент в него.

Определим процесс формально.

- Рассмотрим ориентированный ациклический граф (DAG) $G = (\mathcal{S}, \mathcal{E})$, где \mathcal{S} это множество состояний и $\mathcal{E} \subseteq \mathcal{S} \times \mathcal{S}$ это множество возможных переходов от одного состояния к следующему.
- В множестве состояний \mathcal{S} есть ровно одна вершина исток s_0 (стартовое состояние), в которую не входит ни одного ребра. Гарантируется, что все вершины графа G достижимы из s_0 по ребрам.
- В множестве состояний \mathcal{S} содержится множество финальных объектов $\mathcal{X} \subseteq \mathcal{S}$, причем это множество в точности совпадает с множеством стоков графа (вершин, из которых не выходят ребра).
- Для каждого состояния $s \in \mathcal{S}$ рассмотрим множество возможных действий \mathcal{A}_s , а именно множество возможных состояний, в которые можно попасть из s . Формально $\mathcal{A}_s = \{s' \in \mathcal{S} \mid (s, s') \in \mathcal{E}\}$
- Генеративная модель GFlowNet \mathcal{F} задается как множество распределений для действий из каждого состояния. Формально для всех состояний $s \in \mathcal{S}$ и следующих состояний $s' \in \mathcal{A}_s$ мы знаем неотрицательные числа $\mathcal{F}(s \rightarrow s')$, в соответствии с которыми будут генерироваться действия.

Определим процесс генерации финальных объектов из множества \mathcal{X} с помощью заданной модели GFlowNet \mathcal{F} :

- Стартуем из стартового состояния $s := s_0$.
- До тех пор пока $s \notin \mathcal{X}$ генерируем следующее состояние $s' \in \mathcal{A}_s$ из вероятностного распределения $\mathcal{P}(s'|s) = \frac{1}{Z_s} \mathcal{F}(s \rightarrow s')$, где $Z_s = \sum_{s' \in \mathcal{A}_s} \mathcal{F}(s \rightarrow s')$. После этого делаем переход в сгенерированную вершину $s := s'$.
- Полученное финальное состояние $s \in \mathcal{X}$ в конце процесса генерации будет являться сгенерированным объектом.

На практике GFlowNet может быть применена следующим образом. Обычно нам известно некоторое множество объектов \mathcal{X} и reward функция \mathcal{R} из реальной жизни (например, это могут быть геномные строки или молекулярные графы и reward функция взятая из их химических или физических свойств). Задача состоит в том, чтобы задать процесс конструирования объектов (граф \mathcal{G}) и построить для него модель \mathcal{F} так, что описанный генеративный процесс будет генерировать объекты из желаемого распределения, заданного \mathcal{R} . Обычно в реальности множество объектов \mathcal{X} очень большое и распределения \mathcal{F} параметризуются нейронной сетью.

Задача

Задача

Вам полностью дан граф генеративных переходов \mathcal{G} , а также GFlowNet модель \mathcal{F} . Для всех финальных объектов $x \in \mathcal{X}$ найдите вероятность, с которой они генерируются.

Формат ввода

В первой строке находится единственное целое число n ($1 \leq n \leq 2 \cdot 10^5$) — количество состояний, то есть $n = |\mathcal{S}|$.

Для простоты пусть множество состояний будет $S = \{1, 2, \dots, n\}$, а стартовое состояние $s_0 = 1$.

В каждой из следующих n строк задаются возможные переходы из каждого состояния и значения \mathcal{F} для них.

В строке с номером s сначала находится целое число k_s ($0 \leq k_s \leq n - 1$) — количество возможных переходов из s , то есть $k_s = |\mathcal{A}_s|$. Затем следует k_s пар целых чисел, i -я пара это $s'_i, \mathcal{F}(s \rightarrow s'_i)$ ($2 \leq s'_i \leq n, s'_i \neq s, 1 \leq \mathcal{F}(s \rightarrow s'_i) \leq 1000$). Множество переходов задается как $A_s = \{s'_1, s'_2, \dots, s'_{k_s}\}$. Гарантируется, что заданные s'_i различны.

Гарантируется, что заданный ориентированный граф \mathcal{G} является ациклическим, все вершины достижимы из вершины 1 по ребрам.

Множество финальных объектов \mathcal{X} задается как множество стоков графа \mathcal{G} (множество состояний s , для которых $|\mathcal{A}_s| = 0$).

Гарантируется, что $\sum_{s=1}^n k_s \leq 2 \cdot 10^5$ (общее количество переходов не превосходит $2 \cdot 10^5$).

Формат вывода

Пусть $\mathcal{X} = \{x_1, x_2, \dots, x_k\}$, где $x_1 < x_2 < \dots < x_k$.

Необходимо найти $\mathcal{P}(x_i)$ — вероятности, с которыми финальные объекты будут генерироваться с помощью описанного генеративного процесса с заданной моделью \mathcal{F} .

Для точной проверки вычисленных вероятностей (во избежание потерь вещественного типа данных) необходимо найти вероятности по простому модулю $P = 10^9 + 7$.

Формально: можно показать, что для всех $x_i \in \mathcal{X}$ вероятность $\mathcal{P}(x_i)$ может быть представлена в виде несократимой дроби $\mathcal{P}(x_i) = \frac{a_i}{b_i}$, где b_i не делится на P . Тогда число для вероятности, которое необходимо вычислить, будет $(a_i \cdot b_i^{-1}) \bmod P$.

Выведите k строк, в i -й строке два целых числа x_i и $\mathcal{P}(x_i)$ по модулю $10^9 + 7$.

Пример 1

Ввод



Вывод



```
5
2 4 1 3 1
0
2 2 2 4 1
2 2 1 5 1
0
```

```
2 666666672
5 333333336
```

Пример 1

Ввод



Вывод



5

2 666666672

2 4 1 3 1

5 333333336

0

2 2 2 4 1

2 2 1 5 1

0

Пример 2

Ввод



Вывод



7

5 111111112

3 2 1 3 1 4 1

6 222222224

3 3 1 4 1 5 1

7 666666672

2 4 1 6 1

1 7 42

0

0

0

Примечания

В первом тесте $\mathcal{P}(2) = \frac{2}{3}$, $\mathcal{P}(5) = \frac{1}{3}$.

Во втором тесте $\mathcal{P}(5) = \frac{1}{9}$, $\mathcal{P}(6) = \frac{2}{9}$, $\mathcal{P}(7) = \frac{2}{3}$.

11. Предсказание победителя в компьютерной игре (10)

Задача предоставлена Научно-образовательным центром когнитивного моделирования МФТИ

В ходе наблюдений за матчами в популярной игре Counter-Strike была собрана база данных. Задача заключается в том, чтобы разработать алгоритм, способный предсказать победителя в конкретном раунде киберспортивного матча на основе данных начала раунда.

Формат ввода

[Ссылка на данные](#)

Тренировочная выборка *train.csv* представляет собой csv-таблицу. Каждая строка представляет собой информацию о раунде.

Описание полей:

- **mapName** — название карты (например, `de_mirage`), представляющее собой строку, указывающую на название игровой карты.
- **roundNum** — номер раунда в игре (целое число), где каждый раунд имеет свой уникальный номер.
- **ctScore** — количество очков команды CT на момент данного раунда (целое число).
- **tScore** — количество очков команды T на момент данного раунда (целое число).
- **ctTeam** — название команды, играющей за сторону CT (строка), например, `00NATION` или `Fluxo`.
- **tTeam** — название команды, играющей за сторону T (строка), например, `Fluxo` или `00NATION`.
- **ctFreezeTimeEndEqVal** — стоимость снаряжения команды CT на конец фазы заморозки перед раундом (целое число), выраженная в игровой валюте.
- **ctRoundStartEqVal** — стоимость снаряжения команды CT в начале раунда (целое число), выраженная в игровой валюте.
- **ctBuyType** — тип покупок команды CT (строка), например, `Full Eco`, `Full Buy`, или `Semi Buy`, определяющий тип закупки снаряжения в начале раунда.
- **tFreezeTimeEndEqVal** — стоимость снаряжения команды T на конец фазы заморозки перед раундом (целое число), выраженная в игровой валюте.
- **tRoundStartEqVal** — стоимость снаряжения команды T в начале раунда (целое число), выраженная в игровой валюте.
- **tBuyType** — тип покупок команды T (строка), например, `Full Eco`, `Full Buy`, или `Semi Buy`, определяющий тип закупки снаряжения в начале раунда.
- **winnerSide** — сторона победителя раунда (целое число): 1, если победила команда CT, и 0, если победила команда T.

Тестовая выборка *test.csv* представляет собой csv-таблицу с аналогичными столбцами-признаками. Ваша задача — для каждой строки из тестовой выборки предсказать `winnerSide`.

- **ctRoundStartEqVal** — стоимость снаряжения команды СТ в начале раунда (целое число), выраженная в игровой валюте.
- **ctBuyType** — тип покупок команды СТ (строка), например, Full Eco, Full Buy, или Semi Buy, определяющий тип закупки снаряжения в начале раунда.
- **tFreezeTimeEndEqVal** — стоимость снаряжения команды Т на конец фазы заморозки перед раундом (целое число), выраженная в игровой валюте.
- **tRoundStartEqVal** — стоимость снаряжения команды Т в начале раунда (целое число), выраженная в игровой валюте.
- **tBuyType** — тип покупок команды Т (строка), например, Full Eco, Full Buy, или Semi Buy, определяющий тип закупки снаряжения в начале раунда.
- **winnerSide** — сторона победителя раунда (целое число): 1, если победила команда СТ, и 0, если победила команда Т.

Тестовая выборка *test.csv* представляет собой csv-таблицу с аналогичными столбцами-признаками. Ваша задача — для каждой строки из тестовой выборки предсказать *winnerSide*.

Формат вывода

В файле *SampleSubmission.csv* показан формат файла ответа, который нужно загружать в систему. В связи с особенностями работы скрипта подсчета метрики качества, необходимо сохранять порядок следования строк как в *test.csv*.

Примечания

Метрика, которая используется в задаче: Ассигасу по всем классам.

Баллы вычисляются по формуле

$$\max(0, 10 \cdot Accuracy)$$

12. Анализ энергопотребления (10)

Задача предоставлена Центром интеллектуальной электроэнергетики ИПУ РАН при поддержке Российской ассоциации искусственного интеллекта

В офисе есть несколько электроприборов, и руководитель хочет понимать, насколько часто используется каждый из них. К сожалению, все приборы подключены в одну сеть, и измерить можно только общее энергопотребление. Программисты предлагают решить эту задачу с помощью машинного обучения. У входа в офис установлен общий счетчик, который измеряет текущую активную мощность сети, реактивную мощность и напряжение. И активная, и реактивная мощности складываются из мощностей всех приборов, активных в данный момент. Для определения прибора используются переменные $G(t)$ и $B(t)$, полученные нормированием мощностей на квадрат напряжения. Задача состоит в том, чтобы по временным рядам $G(t)$ и $B(t)$ в каждый момент времени $t = 1, \dots, T$ определить, активен ли прибор $i = 1, \dots, N$. Модель для каждого момента t должна предсказать значения $y_i(t) \in \{0, 1\}$, где 0 - прибор выключен, 1 - прибор включен. Качество предсказания будет оцениваться по метрике

$$\text{Score} = \min\left(\frac{\sum_t \sum_i I[y_i(t) = \hat{y}_i(t)]}{NT} - 0.5, 0\right) \cdot 20,$$

где $y_i(t) \in \{0, 1\}$ - истинное состояние прибора, $\hat{y}_i(t) \in \{0, 1\}$ - предсказание модели.

Формат ввода

Данные доступны по ссылке: <https://disk.yandex.ru/d/45cYMZOQ5I3v8g>

Архив содержит следующие файлы:

train.csv – обучающие данные. Заголовок (строка с номером 0) содержит названия столбцов. Каждая следующая строка с номером $t = 1, \dots, T$ содержит показания счетчика в момент времени t :

\$G (mkS)\$ – активная мощность сети в момент времени \$t\$, нормированная на квадрат

\$B (mkS)\$ – реактивная мощность сети в момент времени \$t\$, нормированная на квадрат

class1, ..., classN – столбец \$i\$ принимает значение 1, если прибор \$i\$ был активен



test.csv – тестовые данные. Первая строка содержит названия столбцов. Каждая следующая строка t содержит показания счетчика в момент времени k .

random.csv - шаблон решения для загрузки. Количество строк совпадает с файлом test.csv

Формат вывода

Нужно загрузить файл submission.csv, заголовок (строка с номером 0) которого содержит названия классов, как в файле random.csv. Каждая следующая строка t содержит результаты классификации для моментов времени $t = 1, \dots, T$ из файла test.csv:

class1, ..., classN – столбец \$i\$ принимает значение 1, если прибор \$i\$ активен в момент



13. Оптимизация Хирша (10)

Задача предоставлена Институтом искусственного интеллекта AIRI

Для оценки научного вклада в научных организациях часто используется индекс Хирша. Индекс h определяется как наибольшее число h , для которого у учёного есть хотя бы h статей, каждая из которых цитировалась не менее h раз.

Василий, сотрудник исследовательской лаборатории, работает над улучшением показателей на известном академическом портале. Недавно он узнал, что может объединять статьи, и тогда количество их цитирований будет суммироваться. Это позволяет учитывать версии одной и той же статьи вместе. Василий понял, что это также может повысить суммарный индекс Хирша. Однако если объединить статьи, которые не связаны по содержанию, это может вызвать подозрения у научного сообщества. Статьи, объединённые в одну группу считаются как одна статья.

Василий знает, что для анализа статей будут использованы автоматизированные средства. Для каждой группы объединённых статей будет рассчитываться сумма токенных различий в их названиях. Название каждой статьи разбивается на множество токенов, после чего для каждого токена, который есть в группе проверяется, встречается ли он во всех других статьях группы. Если токен отсутствует хотя бы в одной статье группы, показатель общей "подозрительности" ученого увеличивается на 1. Для токенизации используется токенизатор GPT-4o. Известно, что допустимая подозрительность ученого определяется максимальной подозрительностью, представленной группой статей и составляет 42; при её превышении научное сообщество поймает хитреца. Ваша задача помочь Василию улучшить показатели Хирша.

Формат ввода

В файле `submit.json` находится множество со списками опубликованных статей.

Ссылка: <https://disk.yandex.ru/d/OkxwzTn40ZFZvw>

Формат вывода

Измените файл `submit.json`, установив одинаковые группы для статей, которые необходимо объединить: установите в поле "group" для таких статей одинаковое произвольное целое неотрицательное число.

Примечания

Оценка решений участников осуществляется по формуле

$$\text{score} = 10 \times \frac{\max(w, \min(b, s)) - w}{b - w}.$$

Здесь: s – результат полученный участником, b – результат лучшего решения жюри, $w = 24$ – нижний порог оценки.

14. Создаем Марковские процессы (10)

Задача предоставлена Институтом искусственного интеллекта AIRI

Преподаватель Василий дал студентам задачу написать код, который вычисляет оптимальную функцию полезности состояния-действия для Марковского процесса принятия решений (МППР). Одно из решений студентов вызвало у Василия подозрения, так как оно заметно отличалось от эталонного и выглядело так, будто его мог написать не человек. Чтобы проверить свою догадку, Василий решил протестировать это решение на нескольких МППР.

Ваша задача — помочь преподавателю, составив список отличающихся МППР, которые будет сложно решить с помощью алгоритма, код которого приведен ниже:

```
import random

def weighted_choice(choices, weights):
    # Рассчитать суммарный вес
    total = sum(weights)
    # Построить кумулятивные веса (накопленные суммы)
    cumulative_weights = [sum(weights[:i + 1]) for i in range(len(weights))]
    # Случайным образом выбрать число от 0 до total
    r = random.uniform(0, total)
    # Найти и вернуть соответствующий выбор
    for i, cw in enumerate(cumulative_weights):
        if r < cw:
            return choices[i]

def q_learning(mdp, gamma=0.9, alpha=0.1, epsilon=0.1, episodes=1000, max_steps=1000):
    # Извлечь вероятности переходов и функцию награды из модели MDP
    transition_probs = mdp['transition_probs']
    reward_function = mdp['reward_function']

    # Инициализация Q-таблицы
    q_table = {state: {action: 0 for action in actions} for state, actions in mdp['states']}
    # Список всех состояний
    states = list(transition_probs.keys())

    for episode in range(episodes):
        # Начальное состояние выбирается случайно
        state = random.choice(states)
        steps = 0

        while steps < max_steps:
            steps += 1

            # Выбор действия: либо случайное действие (exploration), либо действие с максимальной Q-оценкой (exploitation)
            if random.random() < epsilon:
                action = random.choice(list(transition_probs[state].keys()))
            else:
                action = max(q_table[state], key=q_table[state].get)

            # Определить следующее состояние с учетом вероятностей переходов
            next_states = list(transition_probs[state][action].keys())
```

```

# Выбор действия: либо случайное действие (exploration), либо действие
if random.random() < epsilon:
    action = random.choice(list(transition_probs[state].keys()))
else:
    action = max(q_table[state], key=q_table[state].get)

# Определить следующее состояние с учетом вероятностей переходов
next_states = list(transition_probs[state][action].keys())
probabilities = list(transition_probs[state][action].values())
next_state = weighted_choice(next_states, weights=probabilities)

# Получить награду за выбранное действие
reward = reward_function[state][action][next_state]

# Найти максимальное Q-значение для следующего состояния
max_next_q = max(q_table[next_state].values()) if next_state in q_table
# Обновить Q-значение текущего состояния и действия
q_table[state][action] += alpha * (reward + gamma * max_next_q - q_table[state][action])

# Перейти в следующее состояние
state = next_state

# Вернуть обновленную Q-таблицу
return q_table

```

МППР считается решенным неправильно, если хотя бы одна полезность состояния действия отличается больше чем на 10 от эталонной.

Размер пространства состояний не должен превышать 7, а размер пространства действий — 2. Количество исходов для каждого действия ограничено 2. Вознаграждения могут быть только целыми числами в диапазоне от -10 до 10, а вероятности переходов должны в сумме равняться 1. Каждый МППР будет протестирован 10 раз.

Формат ввода

В файле `submit.json` дан пример решения: 10 МППР, кодируемые двумя ключами `'transition_probs'` — вероятности переходов между состояниями и `'reward_function'` — вознаграждения за переходы.

В файле `example.py` пример расчета полезностей для списка МППР.

Ссылка: <https://disk.yandex.ru/d/jfDQc3YAQcLkwg>

Формат вывода

Создайте новый файл `submit.json`, в котором должны быть указаны МППР, которые будет сложно решить правильно алгоритму студента.

Примечания

За каждый из возможных 10-ти МППР, которые не получится решить правильно, будет начислен 1 балл.