Постановка задачи выпуклой онлайн оптимизации. Базовые методы

Сформулируем модель задачи онлайн выпуклой оптимизации. Дано:

1. *Агенты*: игрок, противник (или природа).
2. *Пространство допустимых решений* $\mathcal{D}$ является выпуклым множеством евклидова пространства $\mathbb{R}^d$.
3. *Семейство функций потерь* $\mathcal{F}$, состоящее из выпуклых функций $f : \mathcal{D} \to \mathbb{R}$. Величины возможных потерь принадлежат некоторому ограниченному множеству.
4. *Горизонт игры* (количество раундов) T (может быть конечным или бесконечным).

Протокол задачи выпуклой онлайн оптимизации следующий:

1. Для каждого раунда $t \in [1, T]$ игрок принимает решение $x_t \in \mathcal{D}$.
2. После принятия решения игрок несет потери, которые неизвестны игроку заранее, т. е. игрок действует в условиях неопределенности. Выглядит это следующим образом: после принятия игроком решения противник выбирает выпуклую функцию $f_t \in \mathcal{F}$, а игрок соответственно несет потери $f_t(x_t)$.
3. Кроме количественного значения потери $f_t(x_t)$ игрок в качестве обратной связи получает градиент $\nabla f_t(x_t)$ и, возможно, матрицу Гессе функции f_t в точке $x = x_t$ или вообще только их аппроксимации. Игрок вправе использовать эту информацию, чтобы принять решение на следующем раунде.

Как и при предсказании с помощью экспертов, алгоритм онлайн выпуклой оптимизации $\mathcal{A}$ характеризуется *кумулятивным регретом* к моменту времени t, который определяется по формуле

$$\text{regret}_t(\mathcal{A}) = \sup_{\{f_1,\dots,f_t\}\subset\mathcal{F}} \left\{ \sum_{s=1}^{t} f_s(x_s) - \min_{x\in\mathcal{D}} \sum_{s=1}^{t} f_s(x) \right\}.$$

Разумеется, особый интерес представляют алгоритмы, для которых $\text{regret}_t(\mathcal{A}) = o(t)$, поскольку в этом случае с течением времени алгоритм в среднем не уступает лучшему фиксированному решению. Точная верхняя грань в определении кумулятивного регрета отражает возможность оказания противником противодействия. Вместе с тем при наличии равномерных оценок на параметры семейства функций $\mathcal{F}$ этот супремум можно опускать.

Рассмотрим примеры задач выпуклой онлайн оптимизации.

Упражнение 1. *Попробуйте с некоторыми оговорками переформулировать задачу принятия решений с помощью экспертов как задачу выпуклой онлайн оптимизации. Почему фреймворк выпуклой онлайн оптимизации пригоден для анализа задач предсказания с помощью экспертов?*

Фильтрация спама в режиме онлайн. Необходимо классифицировать поток поступающих электронных сообщений на два класса: спам и обычное сообщение. Каждому сообщению можно сопоставить вектор $z \in \mathbb{R}^d$ с помощью bag-of-words представления, т. е. компонента z_i вектора z будет флагом наличия в сообщении слова с индексом i из словаря слов, имеющего размерность d. Чтобы классифицировать входящее сообщение, необходим вектор $x \in \mathbb{R}^d$, называемый *фильтром*. Непосредственно классификация осуществляется по формуле

$$p = \operatorname{sgn}\langle x, z\rangle,$$

причем полученная метка отвечает одному из классов (например, $+1$ для обычного сообщения, а -1 для спама). В такой постановке в качестве множества допустимых решений $\mathcal{D}$ можно взять единичный шар, а для сообщения z_t, имеющего метку y_t, семейство $\mathcal{F}$ может состоять, например, из функций вида

$$f_t(x) = (y_t - \operatorname{sgn}\langle x, z_t\rangle)^2.$$

Рекомендательные системы. Набор данных о предпочтениях, образованных парой пользователь-предмет, является частично наблюдаемой матрицей. Строки в этой матрице отвечают пользователям, а столбцы — предметам. На пересечении стоит величина, характеризующая оценку пользователем соответствующего предмета, например лайк (1) или дизлайк (0). Также в матрице присутствуют незаполненные значения (`null`). Поэтому имеющуюся информацию о предпочтениях можно хранить в виде матрицы $M \in \{0, 1, \texttt{null}\}^{n\times m}$, где n — число пользователей, а m — число предметов. Цель рекомендательной системы правильно определить значения для незаполненных полей.

Это можно сделать разными способами. Один из них основан на предположении, что истинная матрица предпочтений обладает малым рангом. Если матрица X имеет ранг k, то

$$X = UV, \quad U \in \mathbb{R}^{n\times k}, \quad V \in \mathbb{R}^{k\times m}.$$

Это означает, что для объяснения предпочтений пользователя достаточно небольшого количества признаков.

Математически эту задачу можно сформулировать следующим образом: определить матрицу $X_\circ \in \mathbb{R}^{n\times m}$, которая удовлетворяет заданному ограничению на ранг, т. е. $\operatorname{rank}(X_\circ) \le k$, причем

$$\|X_\circ - M\|_{\text{набл}} = \min_{\substack{X\in\mathbb{R}^{n\times m}\\ \operatorname{rank}(X)\le k}} \|X - M\|_{\text{набл}},$$

где $\|\cdot\|_{\text{набл}}$ — евклидова норма, которая берется только по компонентам с заполненными предпочтениями. Поскольку ранг матрицы не является выпуклой функцией, осуществляют релаксацию, заменяя его на ядерную норму матрицы $\|\cdot\|_*$, являющуюся суммой сингулярных чисел матрицы.

Перейдем к онлайн постановке для рекомендательной системы:

- множеством допустимых решений являются всевозможные матрицы предпочтений;
- игрок принимает решение $X_t \in \mathbb{R}^{n\times m}$;
- противник выбирает пару (i_t, j_t) пользователь-предмет, а также истинное значение предпочтения пользователя y_t;

- игрок несет потери $f_t(X_t) = ([X_t]_{i_t,j_t} - y_t)^2$.

Вспомним некоторые методы выпуклой оптимизации и рассмотрим в первом приближении их онлайн варианты.

Градиентный спуск и его онлайн версия. Рассмотрим следующую задачу условной оптимизации: для заданной гладкой выпуклой функции $f(x)$, определенной на выпуклом множестве $\mathcal{D}$, найти точку $x_{\min} \in \mathcal{D}$, для которой

$$f(x_{\min}) = \min_{x \in \mathcal{D}} f(x). \tag{1}$$

Эту задачу можно итерационно решить с помощью *градиентного спуска с проектирование градиента*: для начальной точки $x_1 \in \mathcal{D}$ и последовательности шагов $\{\alpha_t\}$ строим последовательность приближений $\{x_t\}$ по формулам

$$\begin{aligned} y_{t+1} &= x_t - \alpha_t \nabla f(x_t), \\ x_{t+1} &= \Pi_{\mathcal{D}}(y_{t+1}), \end{aligned}$$

где $\Pi_{\mathcal{D}}$ — оператор проектирования на выпуклое множество $\mathcal{D}$, который формально определяется формулой

$$\Pi_{\mathcal{D}}(y) = \arg\min_{x \in \mathcal{D}} \|x - y\|_2.$$

Онлайн градиентный спуск устроен следующим образом: для начальной точки (базового решения) $x_1 \in \mathcal{D}$ и последовательности шагов $\{\alpha_t\}$

- игрок в момент $t \in [1, T]$ принимает решение x_t и получает потери $f_t(x_t)$;
- осуществляется сдвиг и проектирование:

$$\begin{aligned} y_{t+1} &= x_t - \alpha_t \nabla f_t(x_t), \\ x_{t+1} &= \Pi_{\mathcal{D}}(y_{t+1}). \end{aligned}$$

Метод Ньютона и его онлайн версия. Как мы помним, задачу (1) также можно решать и с помощью *метода Ньютона*: для начальной точки $x_1 \in \mathcal{D}$ строится последовательность приближений x_t по рекуррентной формуле

$$\begin{aligned} y_{t+1} &= x_t - \left[\nabla^2 f(x_t)\right]^{-1} \nabla f(x_t), \\ x_{t+1} &= \mathcal{P}_{\mathcal{D}}(y_{t+1}), \end{aligned}$$

где $\nabla^2 f$ — матрица Гессе для функции $f(x)$, а $\mathcal{P}_{\mathcal{D}}$ — некоторый оператор проектирования на множество $\mathcal{D}$.

Онлайн вариант метода Ньютона устроен следующим образом. Задается базовое решение $x_1 \in \mathcal{D}$, параметры $\gamma, \varepsilon > 0$, а также матрица $A_0 = \varepsilon I_{d \times d}$. Далее, на каждом раунде $t \in [1, T]$

- игрок принимает решение x_t и получает потери $f_t(x_t)$;
- выполняется одноранговое обновление матрицы по формуле

$$A_t = A_{t-1} + \nabla f_t(x_t) \nabla f_t(x_t)^\top;$$

- производится шаг по методу Ньютона и проектирование:

$$y_{t+1} = x_t - \frac{1}{\gamma} A_t^{-1} \nabla f_t(x_t),$$
$$x_{t+1} = \Pi_{\mathcal{D}, A_t}(y_{t+1}),$$

где $\Pi_{\mathcal{D},A_t}$ — оператор проектирования на множество $\mathcal{D}$, который определяется формулой

$$\Pi_{\mathcal{D}}(y) = \arg\min_{x \in \mathcal{D}} \|x - y\|_{A_t} = \arg\min_{x \in \mathcal{D}} \left\{ \sqrt{(x-y)^\top A_t (x-y)} \right\}.$$

Задача о формировании портфеля ценных бумаг. У игрока имеется начальное состояние, обозначаемое через W_1. В каждом раунде t игрок выбирает распределение x_t своего состояния между d активами, т. е. допустимое решение принадлежит симплексу

$$\Delta_d = \left\{ x \in \mathbb{R}^d_+ : \sum_{i=1}^d x_i = 1 \right\}.$$

Цель игрока — максимизировать свое состояние по проишествии T раундов.

Противник (природа) независимо выбирает рыночную доходность активов, т. е. вектор $r_t \in \mathbb{R}^d_+$, у которого i-ая компонента определяется формулой

$$[r_t]_i = \frac{\text{цена единицы } i\text{-го актива в момент } t+1}{\text{цена единицы } i\text{-го актива в момент } t}.$$

Пусть W_t обозначает состояние игрока в момент t. Тогда

$$W_{t+1} = W_t \cdot r_t^\top x_t,$$

а по истечении T раундов

$$W_T = W_1 \cdot \prod_{t=1}^T r_t^\top x_t.$$

Цель игрока эквивалентна максимизации отношения W_T/W_1, поэтому он добивается увеличения выражения

$$\log \frac{W_T}{W_1} = \sum_{t=1}^T \log r_t^\top x_t.$$

В этой постановке $g_t(x) = \log r_t^\top x$ — вогнутая функция, а регрет к раунду T определяется формулой

$$\text{regret}_T(\mathcal{A}) = \max_{x \in \Delta_d} \sum_{t=1}^T g_t(x) - \sum_{t=1}^T g_t(x_t)$$

и является величиной, которую игрок хочет минимизировать.

Сделав замену $f_t(x) = -g_t(x)$, приходим к выводу, что задача о формировании портфеля ценных бумаг является задачей выпуклой онлайн оптимизации.

Упражнение 2. *Выпишите выражения для градиента и матрицы Гессе функции $f_t(x)$ в задаче о формировании портфеля ценных бумаг.*