
Домашнее задание №3.

Проксимальные стохастические методы.

Практическое задание

Дедлайн: 24 апреля 2022 года, 23:59 по Москве

Действует система¹ мягких дедлайнов!

Это практическое домашнее задание. Для его выполнения нужно будет реализовать нужные методы, а затем поставить эксперименты и сделать выводы. Отчет об экспериментах принимается в формате Jupyter notebook. Также нужно прислать код реализованных вами методов (файл `algorithms.py`). Код с заготовками методов и пояснениями нужно скачать из [репозитория](#). Просьба присылать задания на почту Горбунова Эдуарда: ed-gorbunov@yandex.ru. Кроме того, просьба указывать следующую тему письма: «Методы оптимизации в ML, весна 2022. Домашнее задание 3».

1 Постановка задачи

Задача логистической регрессии² формально определяется следующим образом. Пусть $A \in \mathbb{R}^{m \times n}$ — матрица признаков, $y \in \{-1, 1\}^m$ — вектор ответов. Здесь m — число объектов в датасете, n — число признаков. Функцию потерь в логистической регрессии можно записать в следующем виде:

$$Loss_{logreg}(x) = \frac{1}{m} \sum_{i=1}^m \log(1 + \exp(-y_i \cdot (Ax)_i)), \quad (1)$$

где $x \in \mathbb{R}^n$ — вектор параметров, которые мы и хотим подобрать, $(Ax)_i$ — i -я компонента вектора Ax . Как правило, на практике задачу минимизации функции (1) не решают в чистом виде³, а решают регуляризованную задачу

$$Loss_{logreg}(x) + \hat{R}(x) \rightarrow \min_{x \in \mathbb{R}^n}, \quad (2)$$

¹Указанная дата – мягкий дедлайн, т.е. работы можно присылать некоторое время после дедлайна. Но количество дней просрочки влияет на финальное количество набранных баллов: **если прислать задание на k -й день после дедлайна, то заработанные баллы будут домножены на коэффициент $\max\{0, 1 - 0.1 \cdot k\}$** . Таким образом, если прислать задание на первый день после дедлайна, то баллы будут домножены на 0.9, если на второй день – на 0.8 и так далее. Отсюда следует, что на 10-й день и позднее после дедлайна за задание можно набрать только 0 баллов. Задание можно присылать частями в разные дни. Тогда баллы за каждую часть, присланную после мягкого дедлайна, будут домножены на свой коэффициент.

²Вспомнить/познакомиться с тем, что такое логистическая регрессия, можно, например, на [хабре](#) или в [википедии](#), но для выполнения задания это не обязательно.

³Кстати говоря, неплохое разминочное упражнение – подумайте, почему эту задачу не рекомендуется решать в чистом виде. При всех ли A данная задача имеет решение?

где функция $\hat{R}(x)$ — выпуклая, замкнутая и правильная функция. В частности, очень популярна ℓ_2/ℓ_1 -регуляризация $\hat{R}(x) = \frac{l_2}{2}\|x\|_2^2 + l_1\|x\|_1$. Для простоты в численных методах оптимизации мы будем рассматривать $\frac{l_2}{2}\|x\|_2^2$ не как регуляризатор, а как часть исходной функции. Таким образом, задачу, которую мы будем решать в этом задании, можно переписать в следующем виде:

$$F(x) = \frac{1}{m} \sum_{i=1}^m \underbrace{\left(\log(1 + \exp(-y_i \cdot (Ax)_i)) + \frac{l_2}{2}\|x\|_2^2 \right)}_{f_i(x)} + \underbrace{l_1\|x\|_1}_{R(x)} \rightarrow \min_{x \in \mathbb{R}^n}, \quad (3)$$

где $f(x) = \frac{1}{m} \sum_{i=1}^m f_i(x)$.

2 Свойства задачи

Из лекции 5 мы знаем как вычислять $\text{prox}_R(x)$:

$$\text{prox}_R(x) = [|x| - l_1 \mathbf{1}]_+ \odot \text{sign}(x),$$

где $\mathbf{1} \stackrel{\text{def}}{=} (1, \dots, 1)^\top \in \mathbb{R}^n$, модуль $|x|$, срезка⁴ $[|x| - l_1 \mathbf{1}]_+$ и сигнум (знак) $\text{sign}(x)$ применяются к векторам покомпонентно и $y \odot z \stackrel{\text{def}}{=} (y_1 z_1, \dots, y_n z_n)^\top$ обозначает произведение Адамара двух векторов (покомпонентное произведение).

Разберёмся теперь, какими свойствами обладает функция $f(x)$. Для этого вычислим градиенты и матрицы Гессе функций $f_i(x)$. Для удобства запишем A в следующем виде:

$$A = \begin{pmatrix} a_1^\top \\ \vdots \\ a_m^\top \end{pmatrix}, \quad a_1, \dots, a_m \in \mathbb{R}^n.$$

Тогда

$$\begin{aligned} \nabla f_i(x) &= \nabla \left(\log(1 + \exp(-y_i \cdot (Ax)_i)) + \frac{l_2}{2}\|x\|_2^2 \right) \\ &= \nabla \left(\log(1 + \exp(-y_i \cdot a_i^\top x)) + \frac{l_2}{2}\|x\|_2^2 \right) \\ &= -\frac{y_i \exp(-y_i a_i^\top x)}{1 + \exp(-y_i a_i^\top x)} a_i + l_2 x = -\frac{y_i}{1 + \exp(y_i a_i^\top x)} a_i + l_2 x, \end{aligned}$$

откуда

$$\begin{aligned} \nabla f(x) &= \frac{1}{m} \sum_{i=1}^m \nabla f_i(x) = -\frac{1}{m} \sum_{i=1}^m \frac{y_i}{1 + \exp(y_i a_i^\top x)} a_i + l_2 x \\ &= -\frac{1}{m} \cdot \frac{A^\top y}{1 + \exp(y \odot Ax)} + l_2 x, \end{aligned}$$

⁴По определению $[y]_+ \stackrel{\text{def}}{=} \max\{y, 0\}$

где деление вектора на вектор и вычисление экспоненты от вектора производятся покомпонентно. Таким образом, используя матрично-векторные операции можно вычислять градиент функции $f(x)$. На самом деле точно так же можно вычислять стох. градиенты по любым батчам. Действительно, пусть $S = \{i_1, i_2, \dots, i_k\}$ — некоторый набор индексов из множества $\{1, 2, \dots, m\}$ (возможно, с повторениями) и

$$A_S = \begin{pmatrix} a_{i_1}^\top \\ a_{i_2}^\top \\ \vdots \\ a_{i_k}^\top \end{pmatrix}, \quad y_S = \begin{pmatrix} y_{i_1} \\ y_{i_2} \\ \vdots \\ y_{i_k} \end{pmatrix}.$$

Тогда стох. градиент по батчу S равен

$$\begin{aligned} \frac{1}{k} \sum_{j=1}^k \nabla f_{i_j}(x) &= -\frac{1}{k} \sum_{j=1}^k \frac{a_{i_j} y_{i_j}}{1 + \exp(y_{i_j} a_{i_j}^\top x)} + l_2 x \\ &= -\frac{1}{k} \cdot \frac{A_S^\top y_S}{1 + \exp(y_S \odot A_S x)} + l_2 x, \end{aligned}$$

то есть и для подсчёта стох. градиента достаточно выполнять матрично-векторные операции.

Чтобы оценить параметр сильной выпуклости μ и константу Липшица градиента L функций f_i , вычислим их матрицы Гессе:

$$\begin{aligned} \nabla^2 f_i(x) &= \frac{y_i^2 \exp(y_i a_i^\top x)}{(1 + \exp(y_i a_i^\top x))^2} a_i a_i^\top + l_2 I \\ &= \frac{1}{(\exp(-\frac{1}{2} y_i a_i^\top x) + \exp(\frac{1}{2} y_i a_i^\top x))^2} a_i a_i^\top + l_2 I, \end{aligned}$$

где I — единичная матрица. Заметим, что матрица $a_i a_i^\top$ неотрицательно определена и

$$\frac{1}{(\exp(-\frac{1}{2} y_i a_i^\top x) + \exp(\frac{1}{2} y_i a_i^\top x))^2} \geq 0$$

для всех $x \in \mathbb{R}^n$. Следовательно,⁵

$$\nabla f_i(x) \succeq l_2 I \quad \forall i \in \{1, \dots, m\},$$

а значит, $\nabla f(x) \succeq l_2 I$, то есть функции $f_1(x), f_2(x), \dots, f_m(x), f(x)$ являются сильно выпуклыми с константой сильной выпуклости $\mu = l_2$ (см. Теорему 2.1.11 из [книги Нестерова](#)). Оценим теперь константы гладкости функций $f_i(x)$ и $f(x)$. Введём обозначение $z_i = \exp(-\frac{1}{2} y_i a_i^\top x)$. Тогда

$$\frac{1}{(\exp(-\frac{1}{2} y_i a_i^\top x) + \exp(\frac{1}{2} y_i a_i^\top x))^2} = \frac{1}{\left(z_i + \frac{1}{z_i}\right)^2} \leq \frac{1}{4},$$

⁵Обозначение $B \succeq C$, определённое для квадратных матриц B и C , означает, что матрица $B - C$ является положительно полуопределённой.

где мы воспользовались тем, что для любого положительного числа z выполняется неравенство $z + \frac{1}{z} \geq 2$ (это неравенство следует, например, из неравенства между средним арифметическим и средним геометрическим, применённым для чисел z и $\frac{1}{z}$). Отсюда следует, что

$$\nabla^2 f_i(x) \preceq \frac{a_i a_i^\top}{4} + l_2 I$$

и

$$\nabla^2 f(x) \preceq \frac{1}{4m} \sum_{i=1}^m a_i a_i^\top + l_2 I = \frac{1}{4m} A^\top A + l_2 I.$$

Максимальное собственное число матрицы $a_i a_i^\top$ равняется

$$\begin{aligned} 4\hat{L}_i &\stackrel{\text{def}}{=} \lambda_{\max}(a_i a_i^\top) = \max_{\|x\|_2=1} x^\top a_i a_i^\top x = \max_{\|x\|_2=1} (a_i^\top x)^2 \\ &= \begin{cases} \left(a_i^\top \frac{a_i}{\|a_i\|_2}\right)^2, & \text{если } a_i \neq 0, \\ 0, & \text{иначе} \end{cases} = \|a_i\|_2^2. \end{aligned}$$

Тогда константа гладкости функции f_i равняется

$$L_i = \hat{L}_i + l_2 = \frac{\|a_i\|_2^2}{4} + l_2.$$

Отсюда видно, что для разных $f_i(x)$ у нас своя константа гладкости, а в лекциях мы предполагали, что эти константы одинаковы. В качестве общей константы гладкости для всех f_i нам придётся взять «худшую», т.е. максимальную, из них:

$$L_{\text{worst}} = \max_{i \in \{1, \dots, m\}} L_i = \max_{i \in \{1, \dots, m\}} \frac{\|a_i\|_2^2}{4} + l_2.$$

Заметим, что мы автоматически показали, что функция $f(x)$ является L_{worst} -гладкой, но для функции $f(x)$ эта оценка может оказаться очень грубой. Во-первых, из имеющихся фактов мы на самом деле можем утверждать, что $f(x)$ является L_{average} -гладкой, где

$$L_{\text{average}} = \frac{1}{m} \sum_{i=1}^m L_i = \frac{1}{4m} \sum_{i=1}^m \|a_i\|_2^2 + l_2.$$

Константа L_{average} может оказаться существенно меньше, чем L_{worst} , что в свою очередь означает, что для таких методов, как **prox-GD** и **FISTA**, мы можем использовать шаги больших размеров, что улучшает скорость сходимости как в теории, так и на практике. Во-вторых, мы можем уточнить оценку для константы гладкости f , если воспользуемся уже доказанным неравенством $\nabla^2 f(x) \preceq \frac{A^\top A}{4m} + l_2 I$. Действительно, отсюда следует, что мы можем взять в качестве константы гладкости f следующую оценку:

$$L = \frac{1}{4m} \lambda_{\max}(A^\top A) + l_2 = \frac{1}{4m} \sigma_{\max}^2(A) + l_2,$$

где σ_{\max} — максимальное сингулярное⁶ число матрицы A .

⁶Если вдруг кто-то забыл, что это такое, то почитайте статью в Википедии про сингулярное разложение матрицы (SVD).

Как вы скоро в этом убедитесь, константа L_{worst} может быть заметно больше L . Однако в методах (и даже стохастических) часто (но не всегда) можно использовать константу L вместо L_{worst} , поэтому далее в этом задании используйте именно константу L для выбора размера шага в методах.

3 Задания (20 баллов)

Вам предлагается выполнить несколько заданий, связанных с имплементацией изученных методов и тестированием их на задаче логистической регрессии. Все детали и тесты корректности работы (для каждого метода обязательно нужно, чтобы он проходил тест корректности) можно найти в jupyter notebook'е. Если в задании вас просят объяснить полученные результаты, то нужно прокомментировать, как они соотносятся с теорией (с указанием теоремы или слайда с лекции, где этот вопрос обсуждался) и почему те или иные методы работают быстрее/медленнее и с чем это на ваш взгляд связано. При сравнении методов нужно выбирать одну и ту же точку старта. **Если в задании требуется сравнить работу разных методов или работу одного метода, но с разным выбором размера шага/размера батча, то необходимо строить кривые сходимости для каждого метода на одном графике.**

1. (1 балл) Как мы уже убедились, чтобы посчитать градиент функции $f(x)$ необходимо умножить матрицу A^T на вектор и матрицу A на вектор (аналогично в случае, когда нужно посчитать стох. градиент по батчу). Скорость вычисления этих произведений иногда очень существенно зависит от того, в каком формате хранится матрица A (детали см. в jupyter notebook'е). Проанализируйте для 5-ти различных датасетов (обязательно рассмотрите `a9a`, `gisette` и `australian`, остальные 2 датасета выберите сами), в каком формате лучше хранить A , чтобы итерации методов, которые вы будете имплементировать, работали быстрее (рассмотрите батчи размера 1, 10, 100 и случай, когда батч равен размеру датасета). Кроме того, оцените константы гладкости L_{worst} , L_{average} и L из формул выше (при $l_2 = 0$) для выбранных датасетов. Везде далее (если не оговорено обратное) будем считать, что L — это константа гладкости функции $f(x)$ при $l_2 = 0$.
2. (1 балл) Имплементируйте функцию, вычисляющую $\text{prox}_R(x)$ для $R(x) = l_1 x$.
3. (1 балл) Имплементируйте `prox-SVRG` с мини-батчингом (см. Алгоритм 1).

Algorithm 1 prox-SVRG с мини-батчингом

Require: размер шага $\gamma > 0$, стартовая точка $x^0 \in \mathbb{R}^d$, количество эпох S , длина внутреннего цикла M , размер батча r

```

1: Пусть  $w = x^0$ 
2: for  $s = 0, 1, \dots, S - 1$  do
3:   Вычислить  $\nabla f(w)$ 
4:   for  $k = 0, 1, \dots, M - 1$  do
5:     Случайно равномерно и независимо друг от друга выбрать индексы  $J = \{j_1, j_2, \dots, j_r\}$ 
      (возможно, с повторениями) из множества  $\{1, 2, \dots, m\}$ 
6:      $g^k = \frac{1}{r} \sum_{l=1}^r \nabla f_{j_l}(x^k) - \frac{1}{r} \sum_{l=1}^r \nabla f_{j_l}(w) + \nabla f(w)$ 
7:      $x^{k+1} = \text{prox}_{\gamma R}(x^k - \gamma g^k)$ 
8:   end for
9:    $w = x^0 := x^M$ 
10: end for

```

Используйте заготовку, которая есть в файле `algorithms.py`. Обратите внимание, что есть возможность передать массив индексов `indices`, который содержит случайно сгенерированную выборку из равномерного распределения на множестве $\{1, 2, \dots, m\}$ и которую можно использовать для контроля правильности работы вашей имплементации. Запустите вашу имплементацию для $\gamma = \frac{1}{6(L+l_2)}$ на датасете `a9a`. Возьмите $l_2 = \frac{L}{10000}$ и $l_1 = \frac{L}{1000}$. Если взять $S = 50$, $M = \frac{2*m}{r}$, $r = 10$ и $l_1 = 0$, то значение функции $F(x)$ в найденной точке должно быть примерно таким же, как и значение, которые находит стандартный солвер из `scipy`. Обратите внимание, как зависит число ненулевых элементов у точек, генерируемых `prox-SVRG`, от l_1 . Многие знают, что ℓ_1 -регуляризацию применяют в частности для того, чтобы производить отбор признаков. Таким образом добиваются того, что точка минимума функции $F(x)$ имеет разреженную структуру (это контролируется выбором l_1). Так вот оказывается, что проксимальные методы для такой задачи генерируют последовательности точек, которые тоже обладают разреженной структурой, начиная с некоторого момента. Для этого сравните точку, которую выдаёт `prox-SVRG`, с точкой, которую находит стандартный солвер. Для хотя бы одного датасета рассмотрите 2 случая: $l_1 = 0$ и такое l_1 , что процент ненулевых координат в решении лежит между 10% и 30%. Для всех остальных датасетов достаточно рассматривать только второй случай. Решение можно находить приближённо, используя `prox-SVRG`: запустите метод с размером батча 10 или 100 с таким S , чтобы метод работал примерно 10-20 минут ($S \approx 1000$ для `a9a` и $r = 10$).

Сравните работу `prox-SVRG` для батчей размера $r = 1, 10, 100$ (при этом выбирайте $M = \frac{2m}{r}$) для датасета `a9a` (параметр `indices` выставьте равным `None`). При этом не нужно запускать метод на слишком большое число эпох, достаточно, чтобы он достигал $\frac{\|x^k - x^*\|_2^2}{\|x^0 - x^*\|_2^2}$ порядка 10^{-8} . Здесь и везде далее значение l_2 выбирайте равным $l_2 = \frac{L}{10000}$.

Для корректного сравнения, Вам потребуется проделать действия ниже и ответить на вопросы.

- Для каждого значения l_1 постройте графики сравнения работы метода по числу

итераций, по числу эпох (число подсчётов градиентов складываемых, делённое на размер датасета) и по времени. В итоге у Вас должно получиться для каждого значения ℓ_1 по 3 графика, на каждом графике – 3 траектории метода (каждая траектория соответствует конкретному размеру батча).

- Для какого размера батча метод сходится быстрее по числу итераций? По числу эпох? По затраченному времени? Почему так получается и как это связано (и связано ли) с теоретическими гарантиями рассмотренными в лекции 6?

4. (4 балла) Имплементируйте **prox-SGD** с мини-батчингом и постоянным шагом. Имплементируйте **prox-SGD** с мини-батчингом и периодически уменьшающимся шагом.

Algorithm 2 prox-SGD с мини-батчингом и постоянным шагом

Require: размер шага $\gamma > 0$, стартовая точка $x^0 \in \mathbb{R}^d$, количество эпох S , размер батча r

```

1: for  $k = 0, 1, \dots, S \cdot m - 1$  do
2:   Случайно равномерно и независимо друг от друга выбрать индексы  $J = \{j_1, j_2, \dots, j_r\}$ 
   (возможно, с повторениями) из множества  $\{1, 2, \dots, m\}$ 
3:    $g^k = \frac{1}{r} \sum_{l=1}^r \nabla f_{j_l}(x^k)$ 
4:    $x^{k+1} = \text{prox}_{\gamma R}(x^k - \gamma g^k)$ 
5: end for
```

Algorithm 3 prox-SGD с мини-батчингом и уменьшающимся шагом

Require: стартовый размер шага $\gamma > 0$, стартовая точка $x^0 \in \mathbb{R}^d$, количество эпох S , размер батча r , период уменьшения шага (в эпохах) T , коэффициент уменьшения шага β

```

1:  $c = 1$ 
2:  $d = 0$ 
3: for  $k = 0, 1, \dots, S \cdot m - 1$  do
4:   if  $d \geq T \cdot c$  then
5:      $\gamma := \gamma \cdot \beta$ 
6:      $c := c + 1$ 
7:   end if
8:   Случайно равномерно и независимо друг от друга выбрать индексы  $J = \{j_1, j_2, \dots, j_r\}$ 
   (возможно, с повторениями) из множества  $\{1, 2, \dots, m\}$ 
9:    $d := d + \frac{r}{m}$ 
10:   $g^k = \frac{1}{r} \sum_{l=1}^r \nabla f_{j_l}(x^k)$ 
11:   $x^{k+1} = \text{prox}_{\gamma R}(x^k - \gamma g^k)$ 
12: end for
```

Проверьте правильность имплементации, используя подготовленные тесты. Сравните (постройте графики) **prox-SGD** с постоянным шагом при разных γ (удобно выбирать $\gamma = \alpha \cdot \frac{1}{L+l_2}$ и пробовать разные значения α ; нужно попробовать α разного порядка; достаточно взять $\alpha = 1, 0.1, 0.01$) и **prox-SGD** с уменьшающимся шагом при разных политиках выбора T и β . Рассмотрите размеры батчей $r = 1, 10, 100$. Не обязательно строить

все графики в одной картинке, важно, чтобы построенные графики для разных методов можно было легко отличить. Объясните полученные результаты по аналогии с предыдущим пунктом.

Для корректного сравнения, Вам потребуется проделать действия ниже и ответить на вопросы.

- Сначала рассмотрите **prox-SGD** с мини-батчингом и постоянным шагом. Для каждого размера батча $r = 1, 10, 100$ запустите **prox-SGD** с размером шага $\gamma = \alpha \cdot \frac{1}{L+l_2}$ с $\alpha = 1, 0.1, 0.01$. Для каждого значения r постройте графики сходимости **prox-SGD** с разными α (по числу итераций, эпох и времени работы). Какое α лучше выбирать? Зависит ли ответ от желаемой точности? Зависит ли ответ от r ?
 - Теперь рассмотрите **prox-SGD** с мини-батчингом и уменьшающимся шагом. Для каждого размера батча $r = 1, 10, 100$ запустите **prox-SGD** со стартовым размером шага $\gamma = \frac{1}{L+l_2}$. Попробуйте $T = 1, 2$ и $\beta = 0.5, 0.9$ (4 варианта). Для каждого значения r постройте графики сходимости **prox-SGD** с разными политиками выбора T, β . Какая политика выбора шага лучше? Зависит ли ответ от желаемой точности? Зависит ли ответ от r ?
 - Теперь для обоих вариантов **prox-SGD** для каждого r выберите лучшие параметры (α – в первом случае, T, β – во втором случае) для достижения точности $\varepsilon = 10^{-3}$. Они понадобятся для предпоследнего пункта.
5. (2 балла) Имплементируйте методы **prox-GD**, **FISTA** (вариант для сильно выпуклого случая, см. лекцию 4), а также обычный субградиентный спуск (**GD**)⁷. Для **prox-GD** и **FISTA** параметры выбирайте согласно лекции 4, для **GD** размер шага попробуйте подбирать как $\gamma = \alpha \cdot \frac{1}{L+l_2}$ с $\alpha = 1, 0.1, 0.01$. Постройте графики сходимости и объясните полученные результаты по аналогии с предыдущими пунктами.
6. (2 балла) Выберите для каждого метода его лучшие параметры (в случае **prox-SGD** – лучшие параметры для достижения точности $\varepsilon = 10^{-3}$ для каждого r), которые вы для него тестировали, и постройте на одной картинке 9 траекторий сходимости рассмотренных методов. Сравните методы по времени работы и по числу проходов по датасету. Объясните полученные результаты, опираясь на теоретические результаты рассмотренные в лекциях.
7. (9 баллов) Повторите эксперименты из предыдущих пунктов для ещё одного датасета на Ваш выбор. Вызывать тесты корректности работы не нужно.

⁷Контрольный вопрос: в чём разница между **prox-GD** и **GD** для рассматриваемой задачи?