Московский государственный университет имени М. В. Ломоносова

Факультет Вычислительной Математики и Кибернетики
Кафедра Математических Методов Прогнозирования

ОТЧЁТ ПО ЗАДАНИЮ №2

«Градиентные методы обучения линейных моделей. Применение линейных моделей для определения токсичности комментария»

Выполнил:
студент 3 курса 317 группы
Суглобов Кирилл Алексеевич

Москва, 2021

Содержание

1 Теория

Ниже представлены решения теоретических задач [1]:

1.1 Задача №1: Бинарная логистическая регрессия

В стандартной постановке задачи бинарной классификации рассматривается обучающая выборка

$$X = (x_i, y_i)_{i=1}^{l}, \quad x_i \in \mathbb{R}^d \text{ — объекты}, \quad y_i \in \{-1, +1\} \text{ — метки классов.}$$

Линейная модель классификации определяется следующим образом:

$$a(x) = sign\langle w,\ x_i + w_0\rangle, \quad w \in \mathbb{R}^d \text{ — вектор весов}, \quad w_0 \text{ — сдвиг}$$

Далее будем считать, что среди признаков есть константа. Тогда классификатор задаётся как:

$$a(x) = sign\langle x,\ w\rangle$$

Вместо пороговой фунции потерь $\mathcal{L}_{\mathbb{I}}(M)$ мы используем гладкую, монотонно невозрастающую мажоранту $\mathcal{L}(M)$:

$$\mathcal{L}_{\mathbb{I}}(y, a) = [a \cdot y < 0] = [\big\langle x, w\big\rangle\, y < 0] \leqslant \mathcal{L}(M)$$

Процесс обучения заключается в минимизации эмпирического риска $Q(X, w)$ путём настройки вектора весов w:

$$Q(X, w) = \frac{1}{l}\sum_{i=1}^{l}[\,\underbrace{\big\langle x_i, w\big\rangle\, y_i}_{M_i(w)} < 0\,] \leqslant \frac{1}{l}\sum_{i=1}^{l}\mathcal{L}\left(M_i(w)\right) \ \to\ \min_{w}$$

Эту задачу оптимизации можно решать градиентными методами.

Выведем **формулу градиента функции потерь**. Рассмотрим логистическую функцию потерь (это потери отдельно взятого объекта из выборки):

$$\mathcal{L}(M) = \log\left(1 + e^{-M}\right) = \log\left(1 + e^{-\langle x, w\rangle y}\right),$$

тогда градиент функции потерь на одном объекте:

$$\frac{\partial \mathcal{L}}{\partial w} = \frac{\partial}{\partial w} \log\left(1 + e^{-\langle x, w\rangle y}\right) = \frac{1}{1 + e^{-\langle x, w\rangle y}} \cdot \frac{\partial}{\partial w}\left(1 + e^{-\langle x, w\rangle y}\right)$$
$$= \frac{e^{-\langle x, w\rangle y}}{1 + e^{-\langle x, w\rangle y}} \cdot \frac{\partial}{\partial w}\left(-\left\langle x,\, w\right\rangle y\right) = \frac{1}{1 + e^{\langle x, w\rangle y}} \cdot (-yx) = -y\sigma\left(-\left\langle \vec{x}, \vec{w}\right\rangle y\right) \cdot \vec{x},$$

где σ– сигмоида:

$$\sigma(s) = \frac{1}{1 + e^{-s}}.$$

Значит, **градиент функции потерь на всей выборке равен**:

$$\frac{\partial Q}{\partial w} = -\frac{1}{l} \sum_{i=1}^{l} y_i \sigma\left(-\left\langle x_i,\, w\right\rangle y_i\right) \cdot x_i$$

1.2 Задача №2: Мультиномиальная логистическая регрессия

Пусть имеется m классов: $1, \ldots, m$, тогда:

$$\mathbb{P}(y = k \mid x, w_1, \ldots, w_m) = \mathrm{Softmax}_k\left(\left\langle x\, w_1\right\rangle, \ldots, \left\langle x,\, w_m\right\rangle\right) \text{ (k-я компонента)},$$

где Softmax:

$$\mathrm{Softmax}(s) = \left(\frac{e^{s_1}}{\sum_{t=1}^{m} e^{s_t}}, \ldots, \frac{e^{s_m}}{\sum_{t=1}^{m} e^{s_t}}\right) : \mathbb{R}^m \to (0, 1)^m$$

Достаточно иметь вероятности для $m - 1$ класса, так как для $m -$ класса вероятность найдётся из нормировки. Без ограничения общности положим $w_m = 0$

Пусть:

$$y_{ik} = [y_i = k], \quad p_{ik} = \frac{e^{\langle x_i, w_k\rangle}}{\sum_{t=1}^{m} e^{\langle x_i, w_t\rangle}}, \quad \text{где } i \in \{1, \ldots, l\},\ k \in \{1, \ldots, m\} \quad (w_m = 0)$$

Обучение ведётся с помощью метода максимального правдоподобия, выведем функцию потерь через правдоподобие (максимум правдоподобия эквивалентно минимуму эмпирического риска):

$$-l(w) = -\log \prod_{i=1}^{l} \mathbb{P}(y_i|x_i, w_1, \ldots, w_{m-1}) = -\log \prod_{i=1}^{l} \prod_{k=1}^{m} p_{ik}^{y_{ik}} = -\sum_{i=1}^{l} \sum_{k=1}^{m} y_{ik} \log p_{ik} = l \cdot Q(X, w)$$

$\Rightarrow Q(X, w) = -\dfrac{1}{l} \displaystyle\sum_{i=1}^{l} \sum_{k=1}^{m} y_{ik} \log p_{ik}$. Значит, потери на одном объекте:

$$\mathcal{L}_i = -\sum_{k=1}^{m} y_{ik} \log p_{ik} = -\sum_{k=1}^{m} y_{ik} \log \frac{e^{\langle x_i, w_k \rangle}}{\sum_{t=1}^{m} e^{\langle x_i, w_t \rangle}} = -\sum_{k=1}^{m} y_{ik} \langle x_i, w_k \rangle + \log \sum_{t=1}^{m} e^{\langle x_i, w_t \rangle}$$

Тогда градиент потери на одном объекте по вектору весов w_j:

$$\frac{\partial \mathcal{L}_i}{\partial w_j} = \frac{1}{1 + \sum_{k=1}^{m-1} e^{\langle x_i, w_k \rangle}} \cdot e^{\langle x_i, w_j \rangle} \cdot x_i - y_{ik} x_i = \left(p_{ij} - y_{ij}\right) x_i.$$

$\Rightarrow$ градиент функции потерь на всей выборке по вектору весов w_j равен:

$$\frac{\partial Q}{\partial w_j} = \frac{1}{l} \sum_{i=1}^{l} \left(p_{ij} - y_{ij}\right) \cdot x_i \text{ – столбцы градиента } \frac{\partial Q}{\partial w} \in \mathbb{R}^{d \times (m-1)}$$

Конкатенируем столбцы и получаем матрицу – градиент полной функции потерь матрице весов.

1.3 Задача №3: Мультиномиальная модель для двух классов

В многоклассовом случае в векторе вероятностей принадлежности объекта классам достаточно иметь вероятности для $m - 1$ класса, так как для $m -$ класса вероятность найдётся из нормировки. Без ограничения общности положим $\mathbf{w_m} = \mathbf{0}$

Рассмотрим случай $m = 2$ с точки зрения мультиномиальной логистической регрессии. Положим, как говорилось выше, $w_m = w_2 = 0$. Тогда вектор вероятностей принадлежности объекта x классам:

$$\left(\frac{e^{\langle x, w_1 \rangle}}{e^{\langle x, w_1 \rangle} + 1}, \frac{1}{e^{\langle x, w_1 \rangle} + 1}\right) = \left(\sigma\left(\langle x,\ w_1 \rangle\right), \sigma\left(-\langle x,\ w_1 \rangle\right)\right) = \{\text{замена } w := -w_1\} =$$
$$= \left(1 - \sigma\left(\langle x, w \rangle\right), \sigma\left(\langle x,\ w \rangle\right)\right)$$

А это в точности вероятности принадлежности объекта x классам в бинарной логистической регрессии.
Показали, что при количестве классов = 2, задача мультиномиальной логистической регрессии сводится к бинарной логистической регрессии.

2 Эксперименты

Эксперименты этого задания проводятся на датасете комментариев из раздела обсуждений английской Википедии, который был преобразован для решения задачи бинарной классификации: является ли данный комментарий токсичным или нет [2].

Рассматривается бинарная логистическая регрессия с оптимизаторами в виде градиентного спуска (GD) и стохастического градиентного спуска (SGD). Исследуется влияние множества гиперпараметров на зависимости полной функции потерь (усреднённая логистическая функция потерь с L_2-регуляризацией) и качества модели (доля правильных ответов, то есть *accuracy*) от времени, итераций или эпох при конкретных гиперпараметрах.

Также данные предобрабатываются и преобразуются различными способами: в поисках улучшения качества модели.

Изначально имеем обучающую и тестовую выборку, делаем на них предобработку, подготавливаем тексты. Далее в ходе экспериментов мы будем смотреть качество модели на валидационной (отложенной выборке), которую выделим из обучающей (соответственно, уменьшив обучающую выборку при этом).

Выборка	**Обучающая**	**Отложенная**	**Тестовая**
Число объектов	40000	12061	20676

Таблица 1: Разбиение датасета по выборкам

2.1 Эксперименты №1 и №2: предварительная обработка текста и преобразование выборки

В этих экспериментах требовалось:

1. **Сделать предобработку текста**: привести все тексты (комментарии) к нижнему регистру и заменить в них все символы, не являющиеся буквами и цифрами, на пробелы – датасет был считан с помощью библиотеки `pandas` и обработан с помощью библиотеки регулярных выражений `re`.

2. **Сделать преобразование выборки в разреженную матрицу** `scipy.sparse.csr_matrix`, где значение `x` в позиции `(i,j)` означает, что в документе `i` слово `j` встретилось `x` раз – было сделано при помощи конструктора `sklearn.feature_extraction.text.CountVectorizer` с параметром `min_df = 5` (остаются только слова, которые встречаются в 5 и более документах).

После чего из копии обучающей выборки была выделена валидационная выборка.

Далее, если не уточняется, классификаторы имеют такие параметры по умолчанию: $\alpha = 1$, $\beta = 0$, параметр регуляризации $\lambda = 0$, `tolerance = `10^{-5}, у SGD `batch_size = 500`.

2.2 Эксперименты №3, №4 и №5: градиентный спуск (GD), стохастический градиентый спуск (SGD) и их сравнение

2.2.1 Исследование моделей при разных α и β

В задании используется такая скорость обучения:

$$\eta_k = \frac{\alpha}{k^\beta}, \quad \text{где } k \text{ – номер итерации (эпохи)}, \ \alpha, \beta \text{ – численные константы.}$$

Скорость обучения (2.2.1) зависит от обеих констант, поэтому будем подбирать α и β вместе. На графиках будет зафиксирован β и будет варьироваться α.

Градиентный спуск
В ходе экспериментов были построены графики зависимости функции потерь и качества модели от номера итерации и от времени.

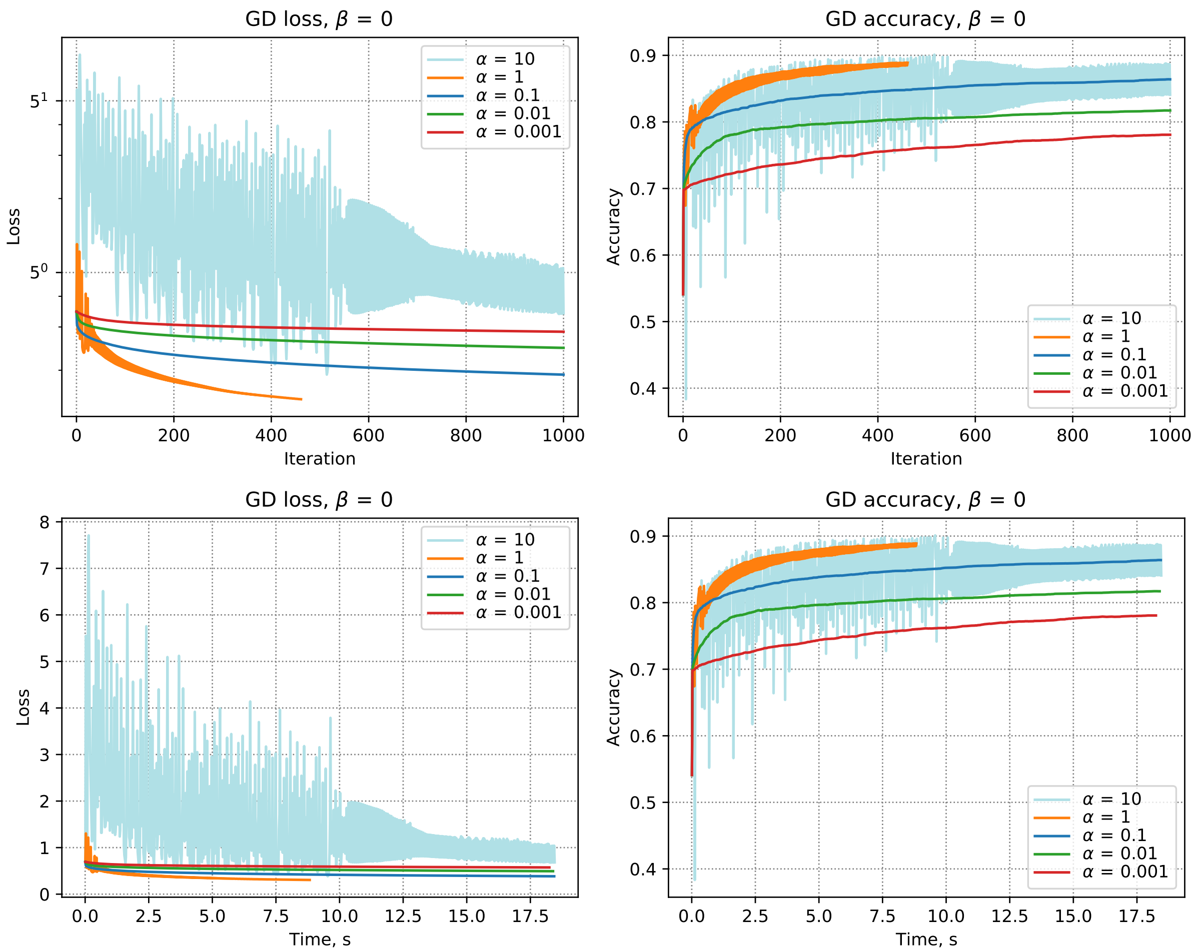

Рис. 1: Градиентный спуск, $\beta = 0$

При $\beta = 0$ (рис. 1) скорость обучения – константа. При больших значениях α имеется зигзагообразность и у графика функции потерь, и у доли правильных ответов (сильнее всего выражено на первых итерациях).Скорость обучения с максимальным $\alpha = 10$ не некоторых итерациях показывает улучшение качетства до самого лучшего, а после ухудшение. А на других α поведение достаточно стабильное и с уменьшением α, а следом и скорости обучения алгоритм сходится медленнее.

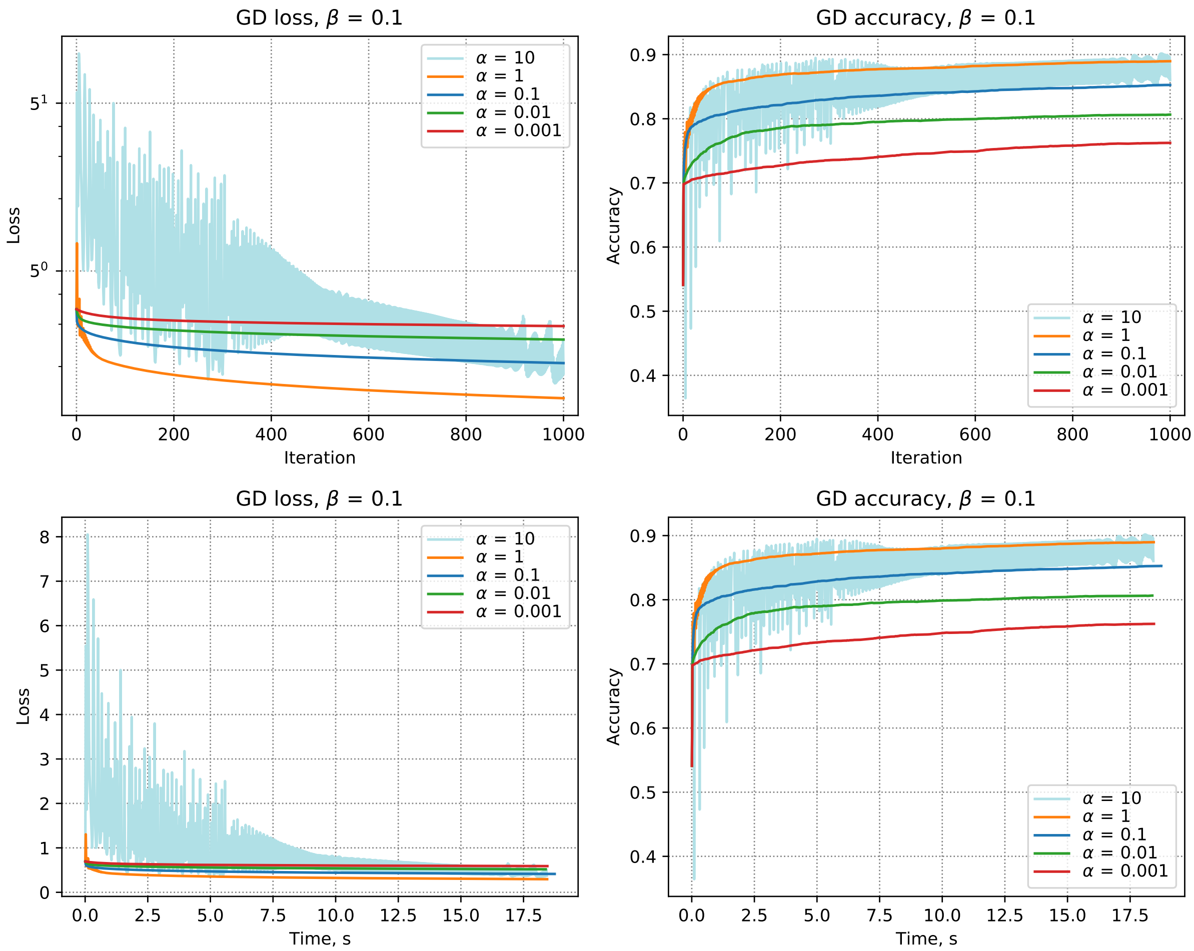

Рис. 2: Градиентный спуск, $\beta = 0.1$

При $\beta = 0.1$ (рис. 2) графики более гладкие и менее «разбросанные» с увеличением итераций. При $\alpha = 10$ функция потерь сходится быстрее, а при остальных α функция потерь и качество похожи на асимптоты.

При $\beta = 0.5$ (рис. 3) видно, что чем больше α, тем меньше потери, и больше точность: с ростом α сходимость улучшается, несмотря на то, что 1000 итераций недостаточно для сходимости при этом β.

При $\beta = 1$ (рис. 4), градиентный спуск сходится достаточно быстро (сходимость видна как отсутствие особых изменнеий или колебаний на соседних итерациях). Также из-за быстрой сходимости, как видно по качеству, веса не достигают оптимальных значений.

При максимальном $\beta = 2$ (рис. 5) сходимость ещё быстрее, чем при $\beta = 1$. В результате качество самое низкое. Также по-прежнему сохраняется монотонная зависимость графиков (убывание потерь и возрастание качества при возрастании α) от α.

Итого, большие α дают наилучшее качество, но слишком много колебаний, веса нестабильны. А если взять большое β, то функции потерь выйдут на асимптоту,

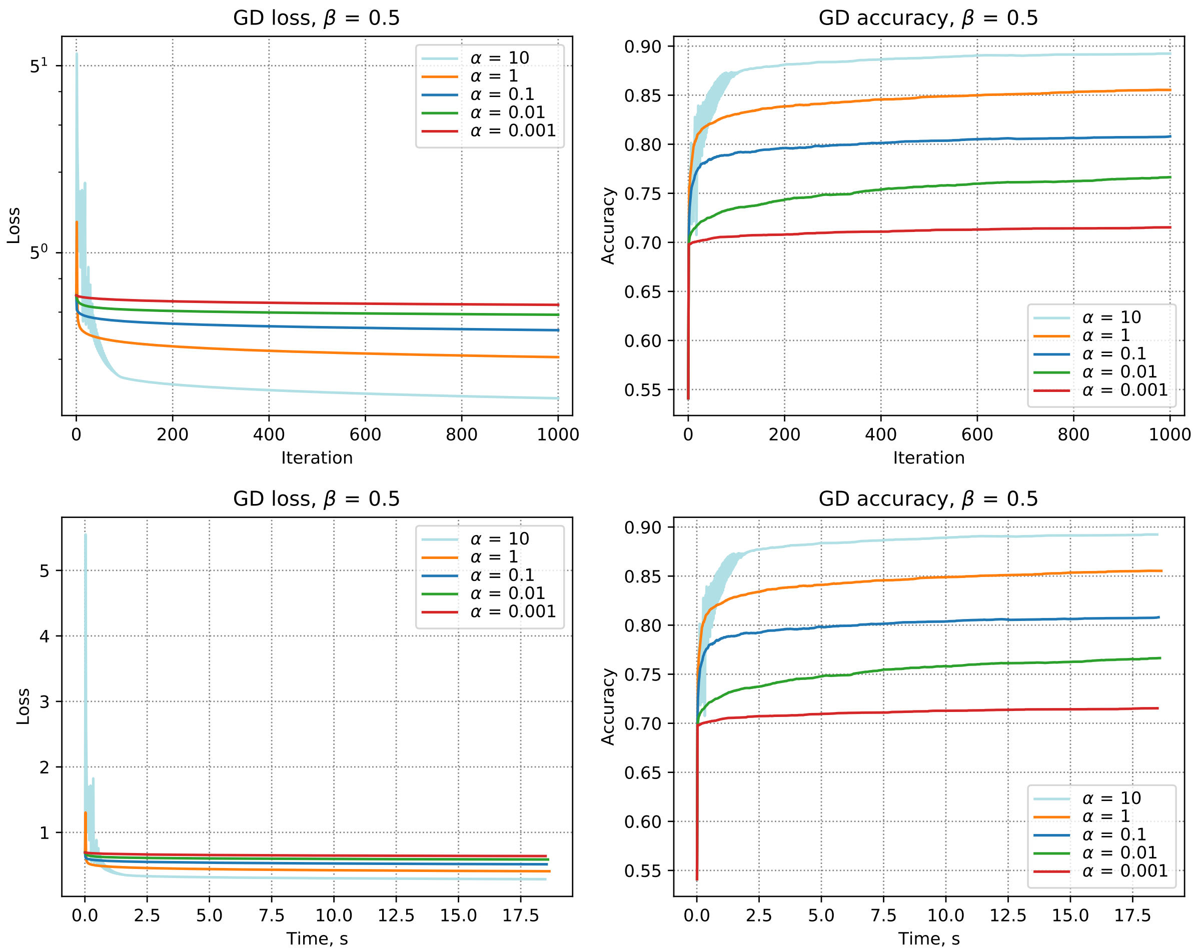

Рис. 3: Градиентный спуск, $\beta = 0.5$

и веса не успеют сойтись к значениям, отвечающим высокому качеству. Таким образом, судя по графикам, надо брать либо $\beta = 0$ и $\alpha \approx 1$(т.к. при больших α не будет стабильности, а при малых — будет медленная сходимость и низкое качество), либо малое $\beta > 0$ (здесь 0.1 – 0.5) и большое $\alpha \geqslant 1$.

Стохастический градиентный спуск

Аналогично, посмотрим как α и β влияют на поведение стохастического градиентного спуска. В отличие от GD, на каждом шаге SGD берется не один случайный объект, а подмножество (mini-batch). Соответственно, в данном случае вместо номера итерации используется приближенный номер эпохи, то есть доля объектов, на которых уже считался градиент, от размера выборки.

Также, функция потерь и качетво обновляются, когда разница между текущим и предыдущим приближенным номером эпохи > 1 (значение по умолчанию) – интуитивно близко к итерации GD.

На рис. 6 – рис. 10 приведены графики функции потерь и качества модели на тех же наборах α и β. Аналогично, на больших α имеем большее качество и большие β стабилизируют графики.

Выраженное отличие в том, что при константном шаге (рис. 6) и небольшом α точность выходит на асимтоту, но алгоритм продолжает работать, так как кон-

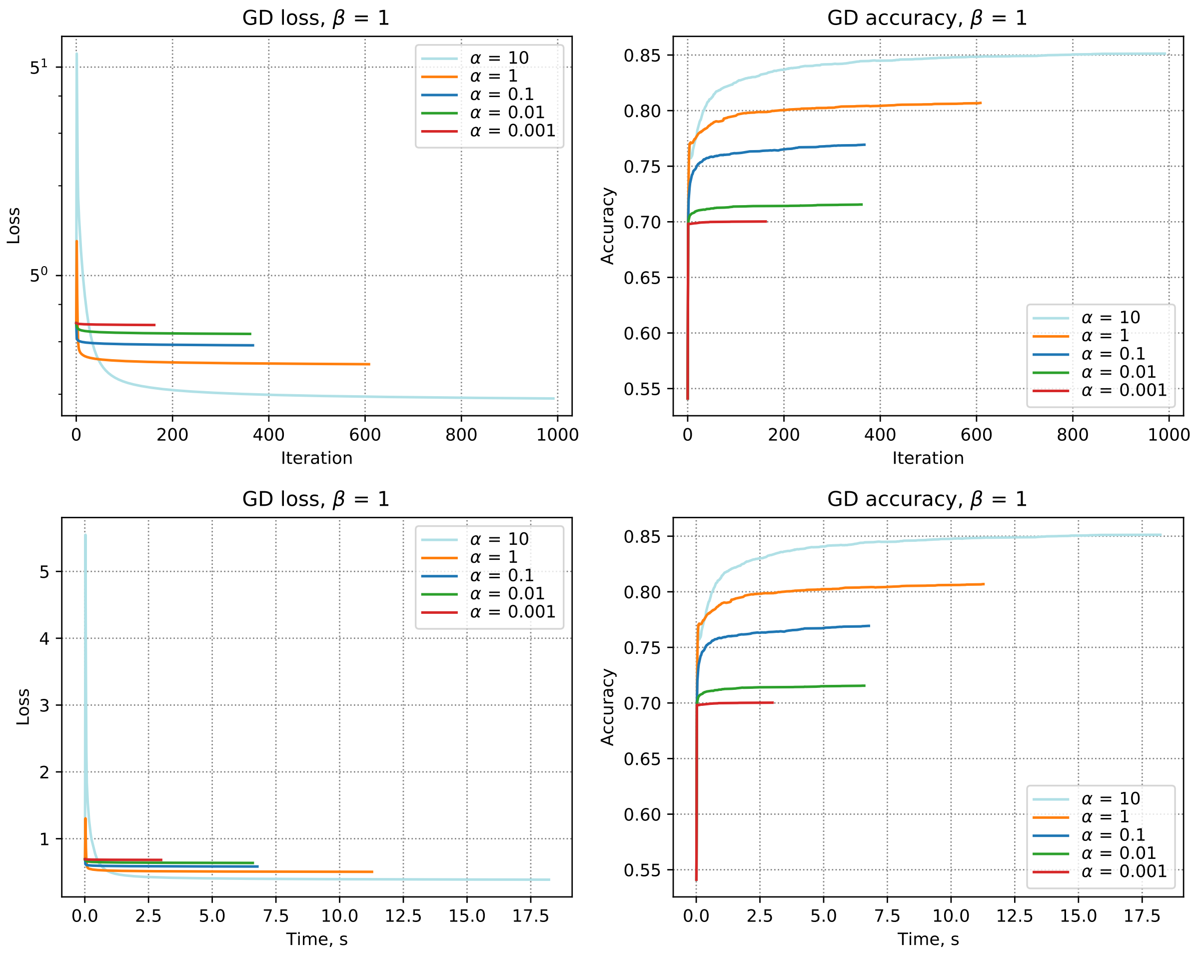

Рис. 4: Градиентный спуск, $\beta = 1$

стантная скорость обучения не гарантируют сходимость для SGD (тем не менее при константном шаге нормальное поведение с $\alpha = 1$). При $\beta = 0.5$ и больше алгоритм работает более предсказуемо: точность и потери монотонно зависят от α и большие β конмпенсируют и позволяют брать большие α. Но при ($\beta = 2$, рис. 10) снова очень сильное затухание, алгоритм не супевает сойтись к оптимальным весам.

Итого, выбор α и β для SGD похож на выбор при GD. Можно брать $\beta = 0$ и $\alpha \approx 1$, но не увеличивать α. Или можно взять $\beta < 2$: при большом α лучше $\beta = 1$, а при малом – $\beta = 0.1$.

2.2.2 Исследование моделей при разных w_0

В этом эксперименте нужно было варьировать начальное приближение весов. Рассмотрим начальные приближения для вектора весов w, идеи инициализации взяты из [3] и из [4]:

1. $w_i = 0$,
2. $w_i = 1$,
3. $w_i \sim U[-\frac{1}{2d}, \frac{1}{2d}]$,

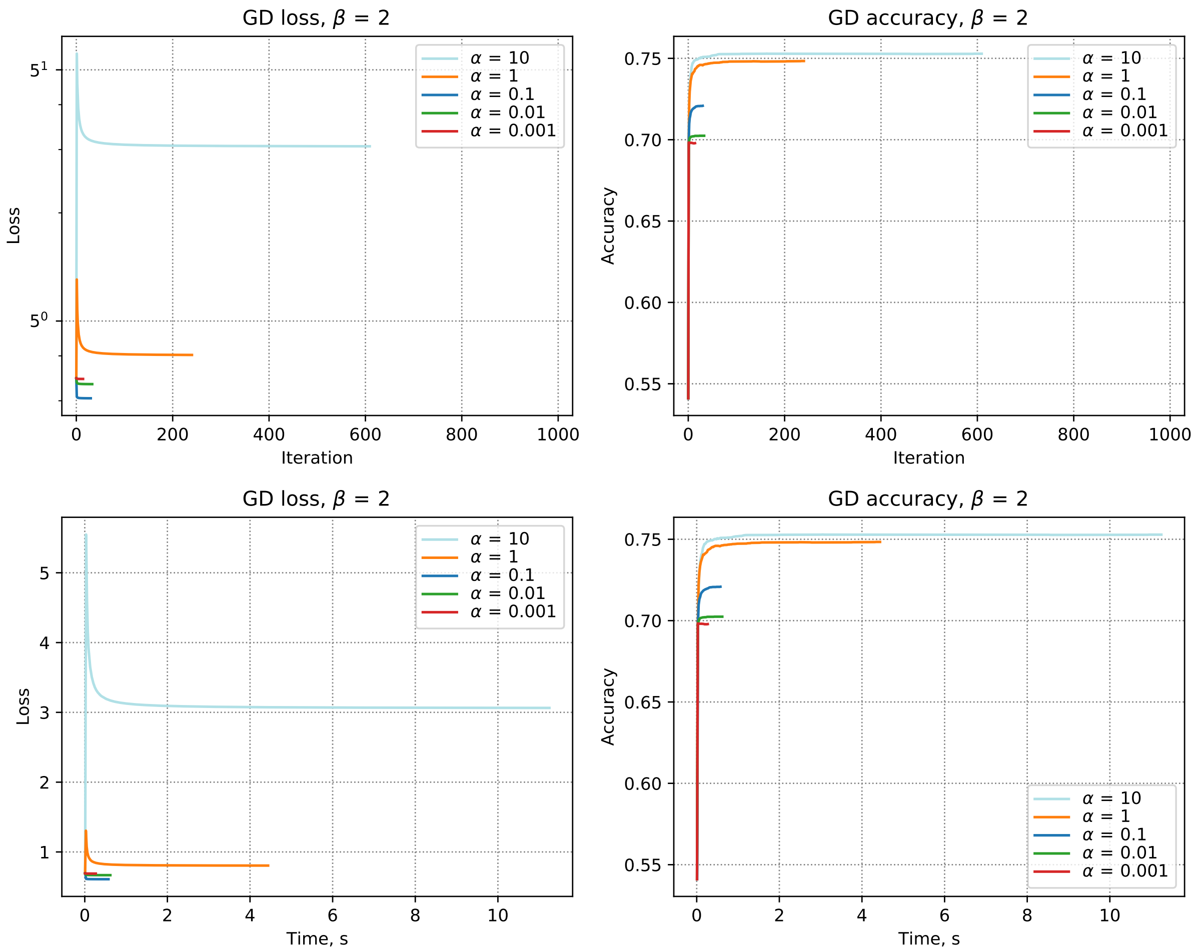

Рис. 5: Градиентный спуск, $\beta = 2$

4. $w_i \sim U[-\frac{1}{\sqrt{d}}, \frac{1}{\sqrt{d}}]$,

5. $w_i = \frac{\langle y, x_i' \rangle}{\langle x_i', x_i' \rangle}$, где y — вектор меток, x_i' — вектор значений i-го признака.

На рис. 11 и на рис. 12 видно, что инициализация случайными числами в окрестности 0 или именно нулями влечё за собой примерно одинаковое поведение. Также все графики ведут себя похожим образом зигзагообразно в случае с SGD.

Инициализация вектором дает наихудшие результаты, так как это ведёт к предсказанию класса 1 с самого начала, поэтому начальная точность очень низкая, ниже 0.65, и функция потерь тоже велика, особенно с начала.

5-е приближение: в градиентном спуске (рис. 11) оно сделало алгоритм более стабильным, чем при инициализации около нуля. А в случае стохастического градиентного спуска (рис. 12) эта инициализация дала высокую точность.

Таким образом, подходит любое начальное приближение, кроме единиц. Либо 5-е приближение, либо в окретности нуля / непосредственно ноль.

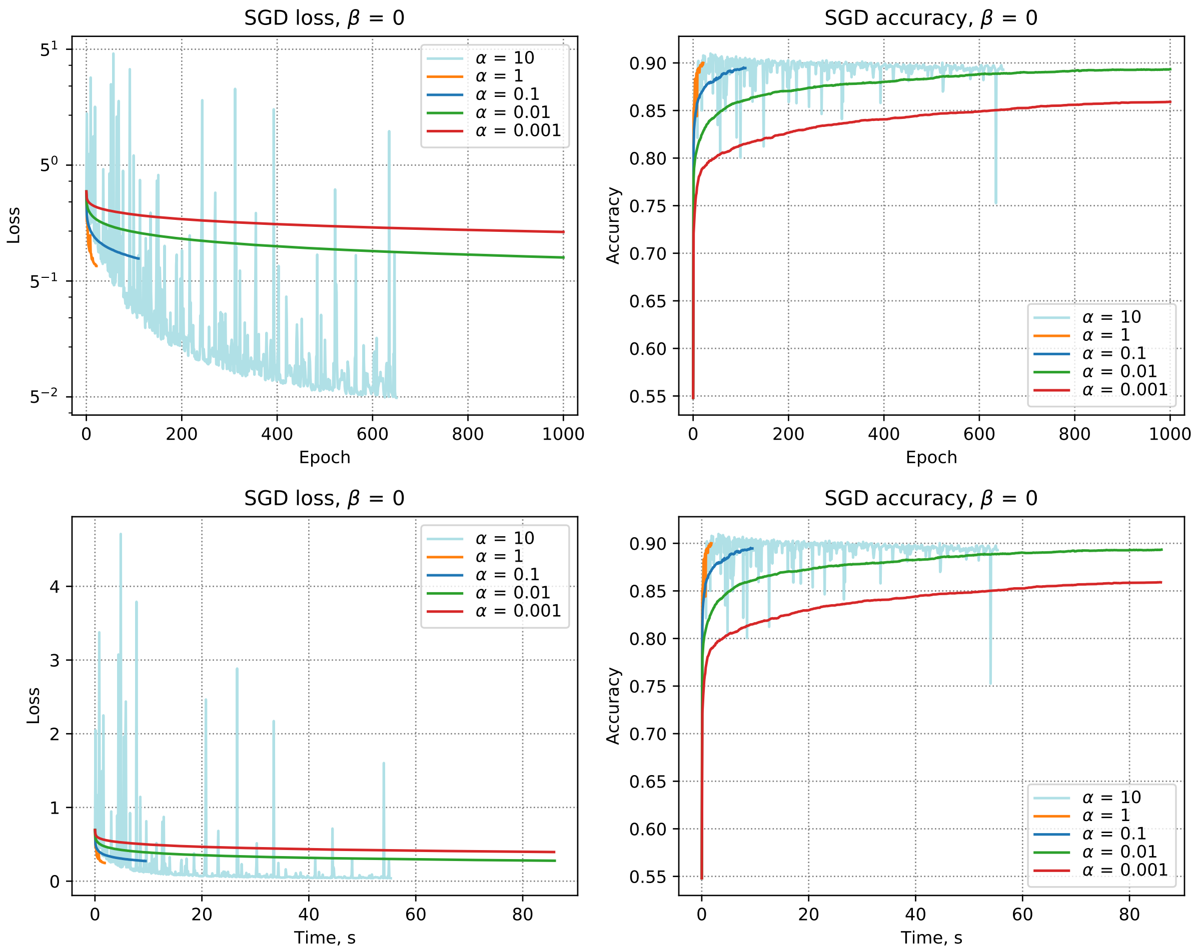

Рис. 6: Стохастический градиентный спуск, $\beta = 0$

2.2.3 Исследование модели SGD при разных batch_size

В этом эксперименте нужно было варьировать размер подвыборки, то есть `batch_size`, на которой пересчитывается градиент, и исследовать влияние `batch_size` на функцию потерь и на качество модели с SGD.

Уменьшение размера батча влечёт за собой увеличение числа шаговов. При константном шаге и маленьком (1 – 10 объектов) батче алгоритм не сходится. То есть функция потерь «в целом» стремится к нулю, но очень медленно, нестабильными движениями. А на графике качества на рис. 13 видно, что оно падает со временем, что точность даже падает.

Как видно по рис. 14, β сглаживает функции, и сходимость есть даже при маленьком батче.

А если взять батч, примерно равный размеру выборки (около 10000 объектов), то алгоритм становится похож на обыкновенный градиентный спуск: рис. 1 и рис. 3.

Лучшее качество имеем при размере батча в 100 – 1000 объектов: алгоритм сходится через несколько эпох, стабильно и с большой точностью.

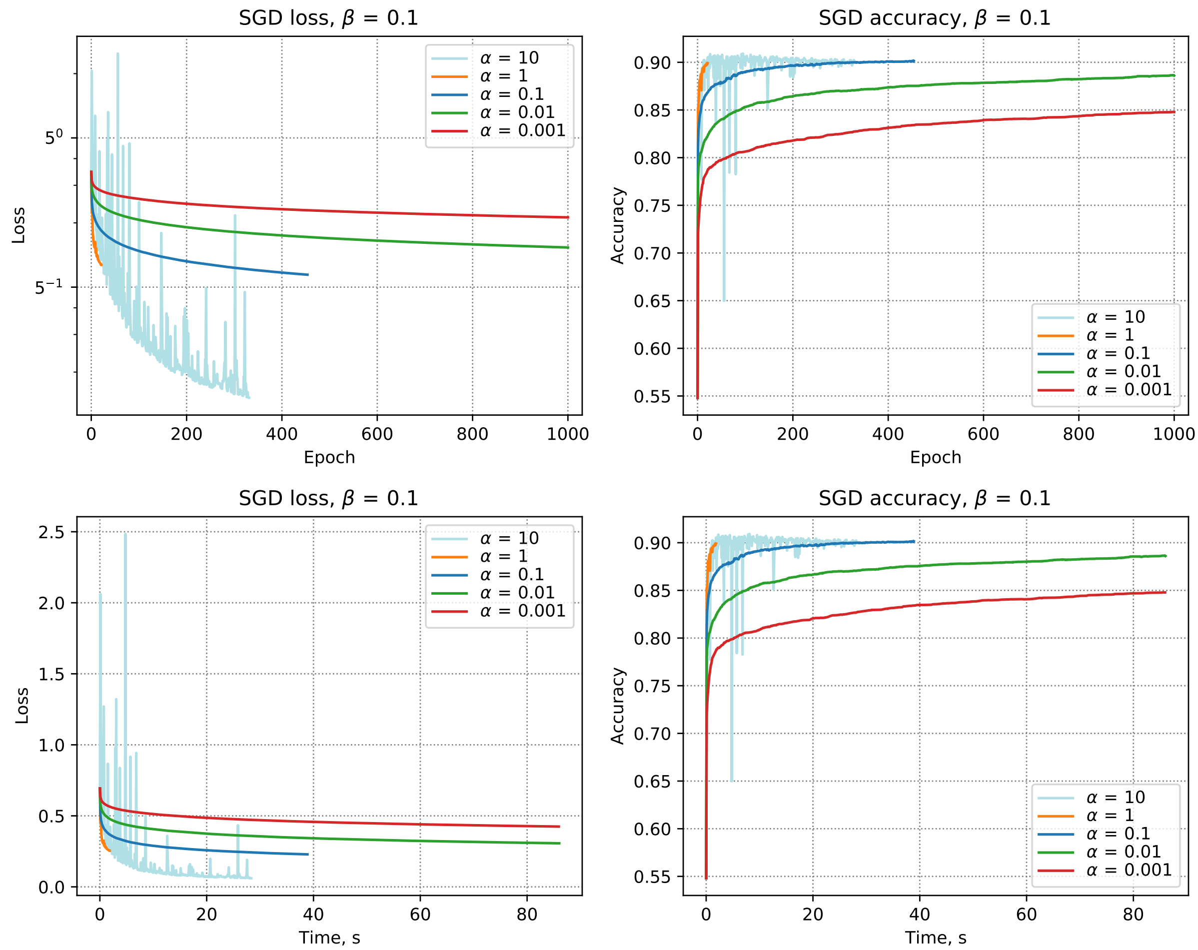

Рис. 7: Стохастический градиентный спуск, $\beta = 0.1$

2.3 Baseline

Был проведён анализ гиперпараметров градиентного спуска и стохастического градиентного спуска. Получили, что, как минимум, для нашей задачи второй метод оптимизации, то есть SGD лучше. На валидационной выборке была выбрана лучшая комбинация гиперпараметров и сам алгоритм (ещё раз):
$SGD, \alpha = 0.1, \beta = 0.1, w_0 \sim U[-\frac{1}{2d}, \frac{1}{2d}]$. Такая модель, обученная на полной обучающей выборке, даёт на тестовой качество **0.8671**.

2.4 Эксперимент №6: лемматизация, удаление стоп-слов

В этом эксперименте требовалось сверху базововй предобработки из первых двух экспериментов применить лемматизацию и удалить стоп-слова. Результаты в табл. 2.

Из результатов эксперимента видно, что:

- Лемматизация хорошо уменьшает признаковое пространство и неплохо увеличивает качество
- Удаление стоп-слов в некоторых случая может растянуть работу алгоритма.

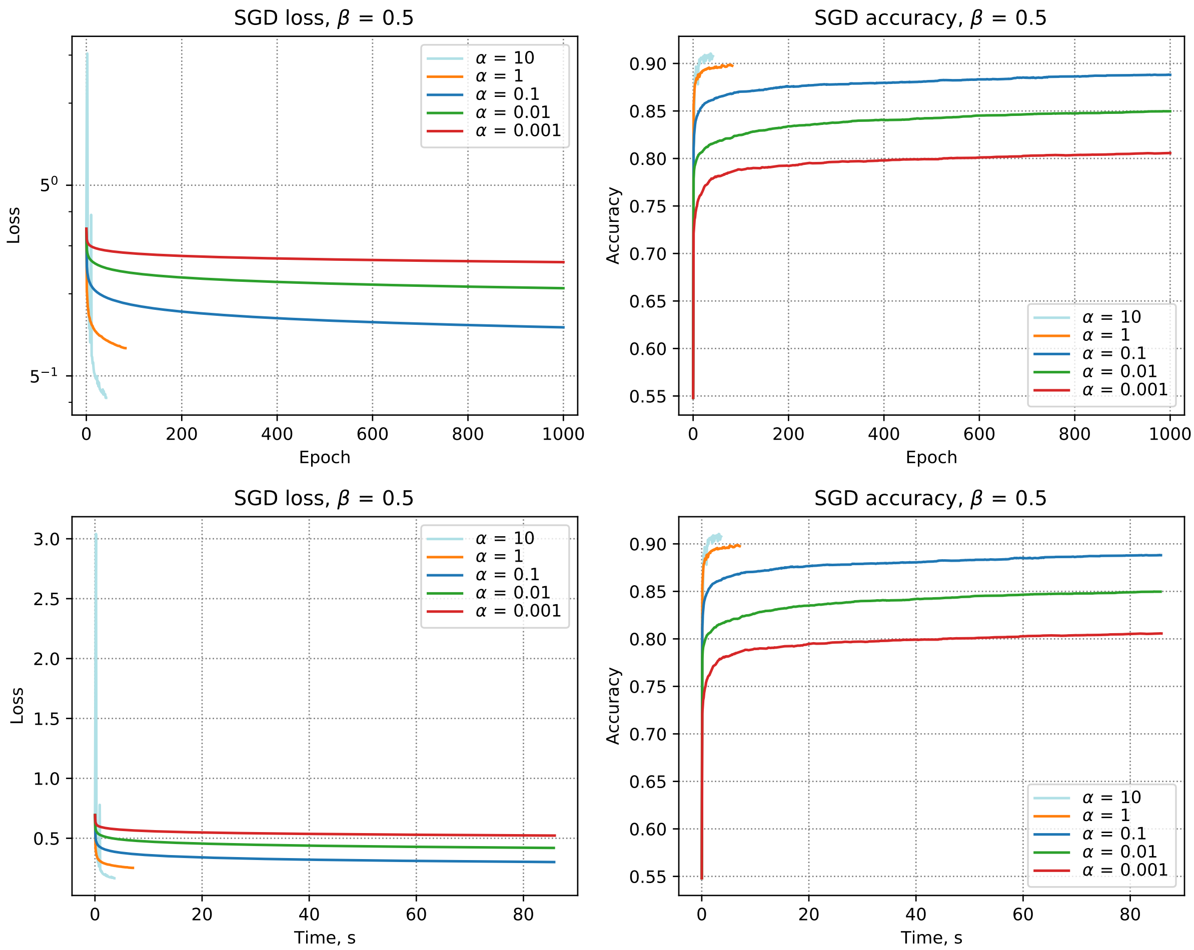

Рис. 8: Стохастический градиентный спуск, $\beta = 0.5$

Лемматизация		✓		✓
Удаление стоп-слов			✓	✓
число признаков	18253	16374	18109	16234
время работы, с	33	14	107	6
точность	0.8671	0.8716	0.8694	0.8684

Таблица 2: Предобработка корпуса текстов

Также есть небольшой прирост в качестве модели, но он не такой, когда к корпусу текстов применялась только лемматизация

Значит, далее рассматриваем курпусы текстов с лемматизацией.

2.5 Эксперимент №7: влияние BagOfWords и Tfidf, min_df и max_df

В этом эксперименте требовалось выяснить, как влияют на качество, скорость работы и размерность признакового пространства параметры Bag of words и TFIDF.

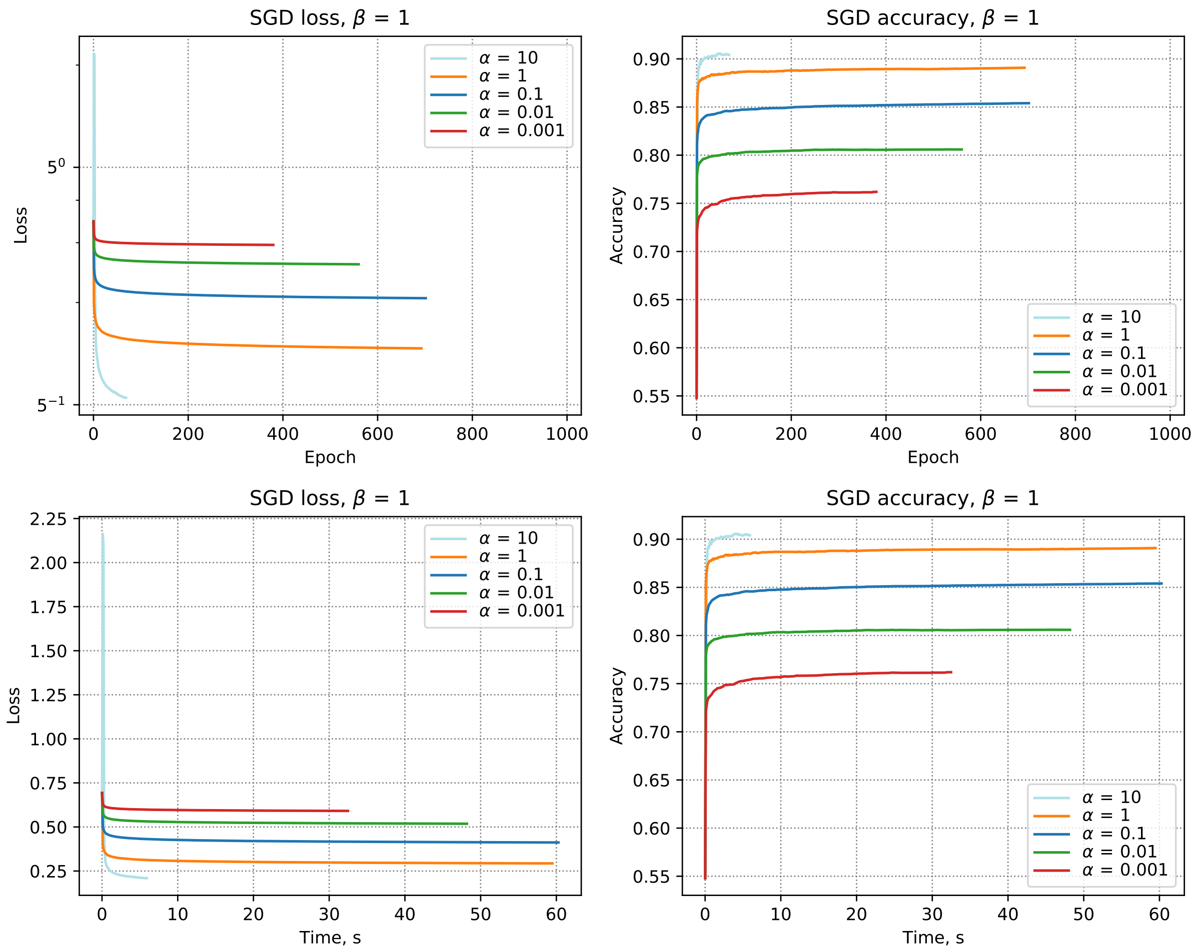

Рис. 9: Стохастический градиентный спуск, $\beta = 1$

min_df	1	3	5	10	20	30	40	45
число признаков	82991	23925	16374	10210	6646	5163	4298	3982
время BoW (с)	35	26	24	38	18	18	24	19
время TFIDF (с)	149	110	97	92	61	60	59	60
точность BoW	0.8665	0.8665	0.8666	**0.8696**	0.8684	0.8685	0.8684	0.8657
точность TFIDF	0.8610	0.8624	0.8636	0.8639	0.8646	0.8648	**0.8661**	0.8659

Таблица 3: Варьирование `min_df`

Сначала подберём `min_df`, дающий максимальное качество. Результаты перебора значений приведены в табл. 3. Ясно, что число признаков очень быстро падает с ростом `min_df`. Точность растёт на bag of words до значения на `min_df = 10`, а потом падает. Аналогично на Tfidf, переломная точка: `min_df = 40`. На BoW больше, поэтому возьмем `min_df=10` и будем теперь выбрасывать самые частые слова – параметр `max_df`.

В случае `max_df` качество выше всего на Tfidf при `max_df=0.05` (качество растёт до этой точки) Итого, лучше всего `max_df = 40` на Tfidf при условии, что

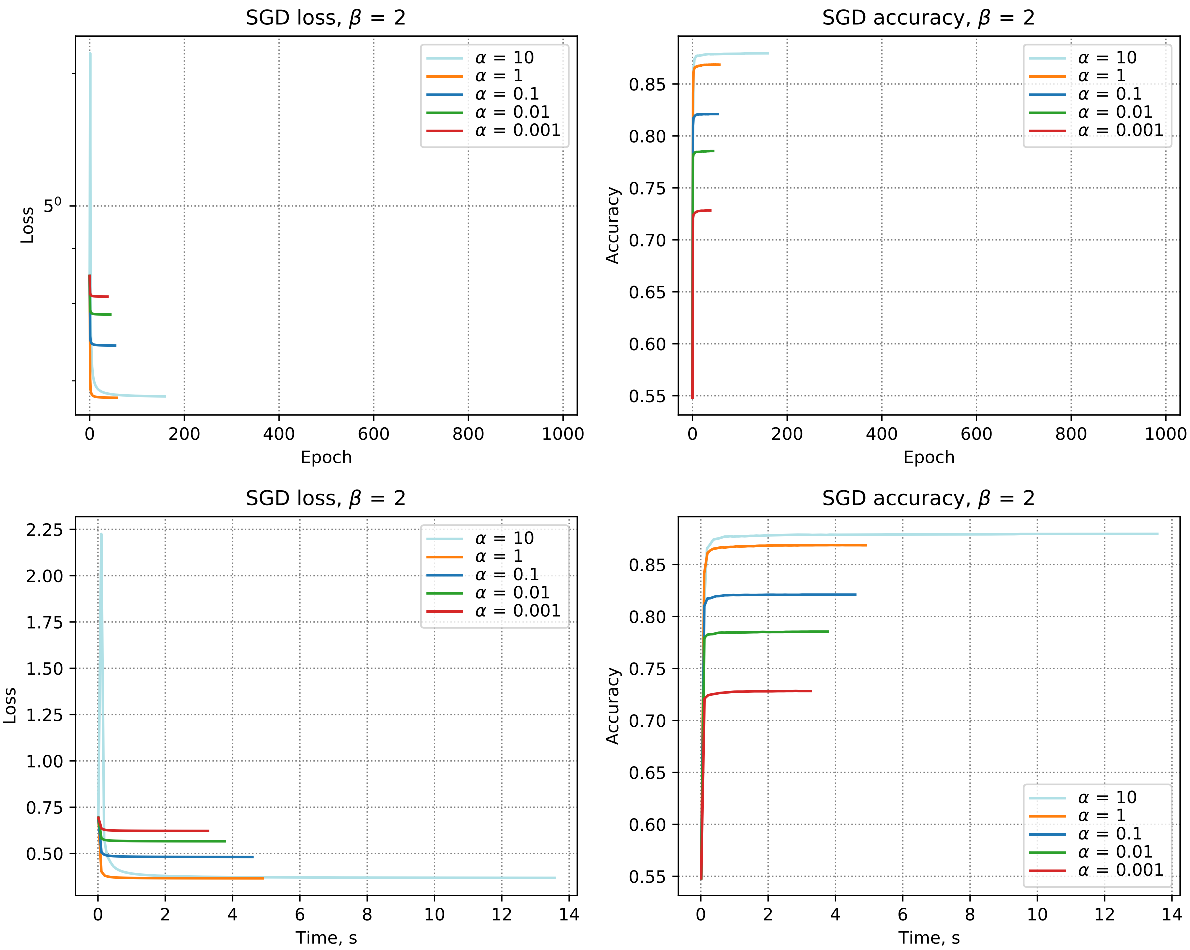

Рис. 10: Стохастический градиентный спуск, $\beta = 2$

max_df	1	0.5	0.3	0.2	0.15	0.1	0.05
число признаков	10210	10207	10199	10190	10174	10156	10099
время BoW (с)	43	79	111	108	106	107	101
время TFIDF (с)	105	96	101	95	93	94	95
точность BoW	0.8696	0.8683	0.8692	0.8690	0.8694	**0.8703**	**0.8703**
точность TFIDF	0.8639	0.8654	0.8707	0.8728	0.8754	0.8789	**0.8812**

Таблица 4: Подбор `max_df`

`min_df = 10` на BoW.

2.6 Эксперимент №8: лучший алгоритм

Итого, наилучшая модель: α = 0.1, β = 0.1, `min_df=10` на BagOfWords, `max_df=0.05` на Tfidf, другие параметры —по умолчанию.

Эта модель дает на тестовой выборке качество **0.8812**.

Ошибки у модели могут быть на объектах, которые представляют собой:

- Фразы, вырванные из контекста (ложное отнесение объекта к классу токсич-

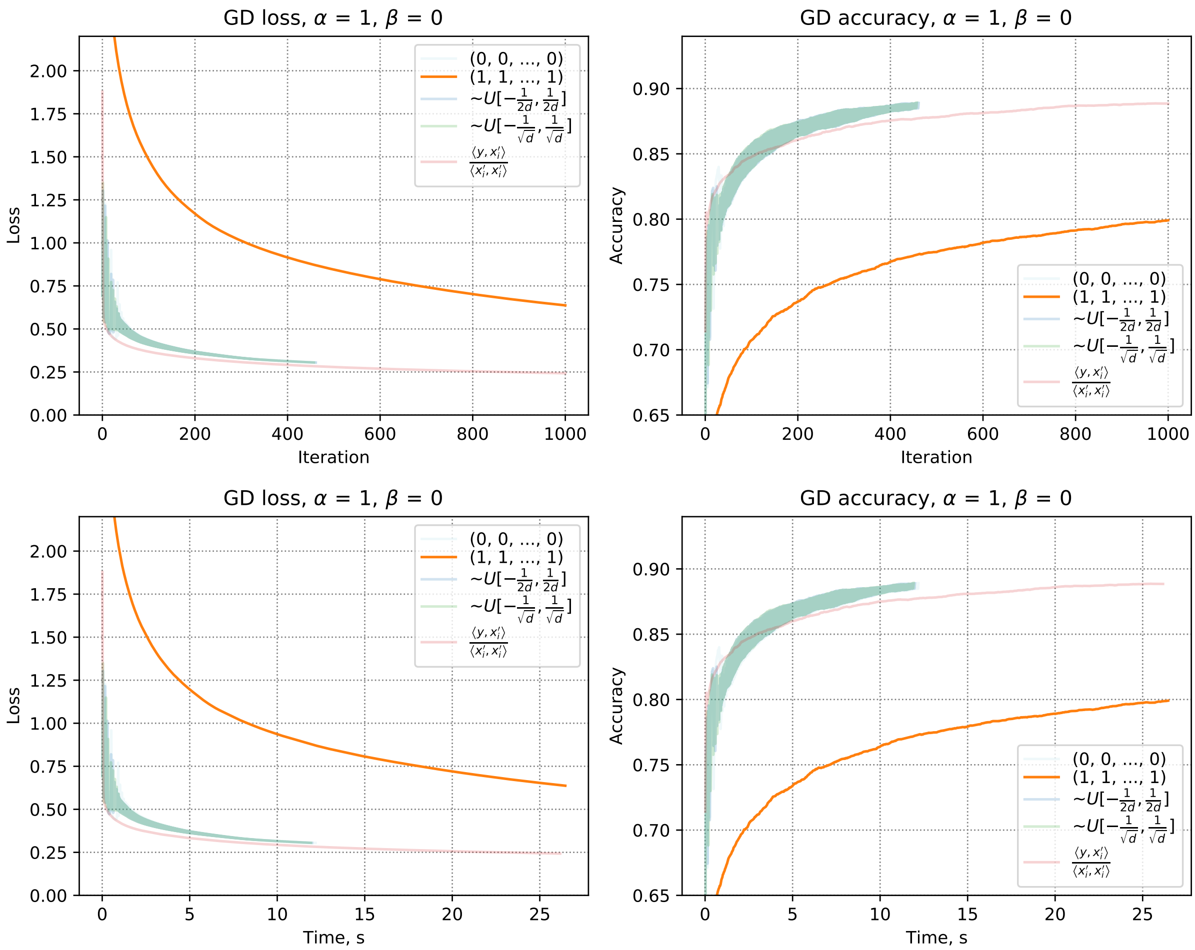

Рис. 11: Градиентный спуск, инициализация весов

ных комментариев)

- Тексты, в которых есть слова, обычно не характерные для токсичных комментариев (ложное отнесение объекта к классу нетоксичных)

2.7 Бонус №1: n-граммы

В ходе эксперименты были добавлены n-граммы. Результаты экспериментов представлены в таблице табл. 5. Из неё видно, что n-граммы не улучшают лучшее качество, а время работы делают только больше. табл. 5.

ngram_range	(1, 1)	(1, 2)	(1, 3)	(1, 4)	(1, 5)
Число признаков	10099	41860	59389	65986	70410
Время работы, с	91	148	167	169	171
Accuracy	**0.8812**	0.8748	0.8734	0.8730	0.8728

Таблица 5: Добавление n-грамм

В нашей задаче n-граммы не нужны.

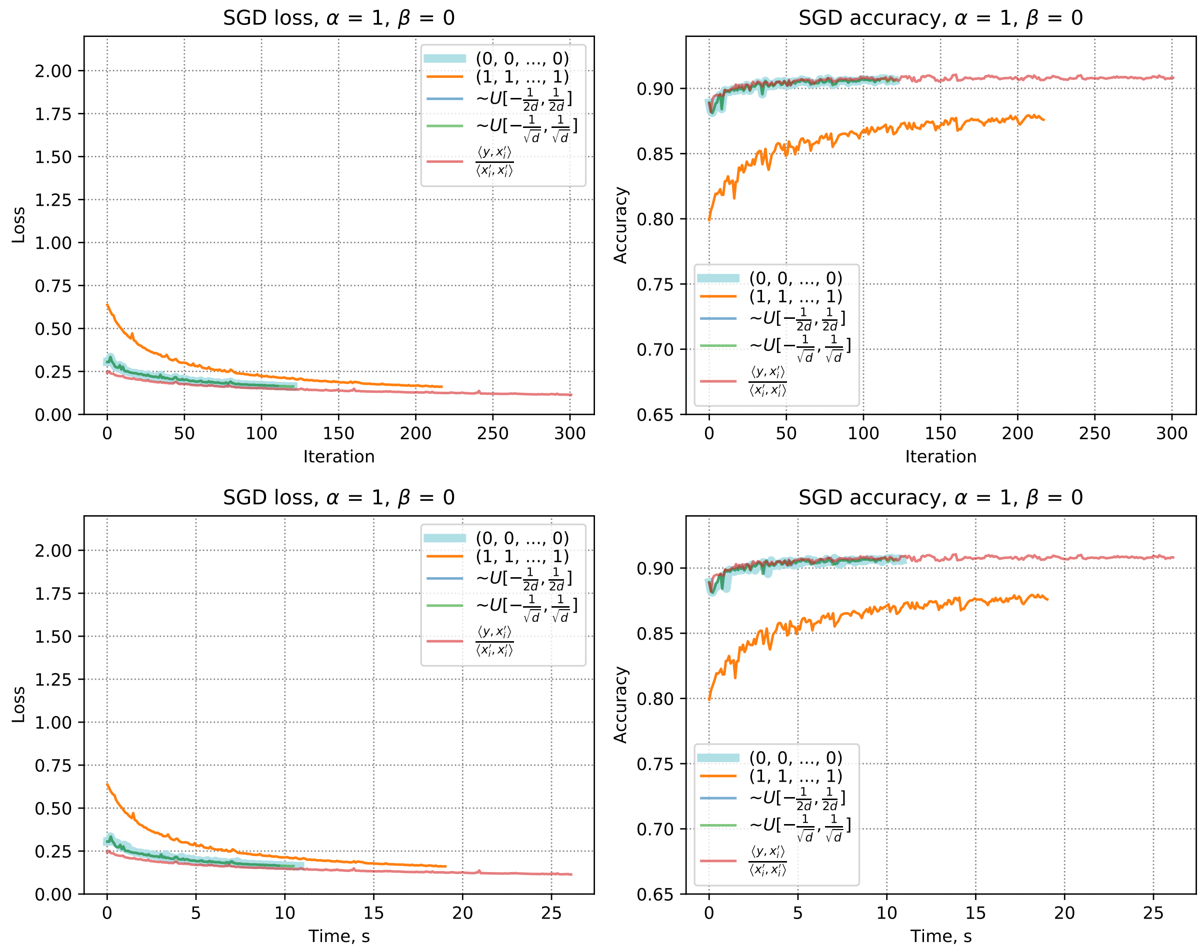

Рис. 12: Стохастический градиентный спуск, инициализация весов

References

[1] *Summary of the lecture ¡¡Linear models for classification¿¿*. URL: https://github.com/mmp-practicum-team/mmp_practicum_fall_2021/blob/main/Seminars/Seminar%2008.%20Text%20processing/%D0%9B%D0%B8%D0%BD%D0%B5%D0%B9%D0%BD%D1%8B%D0%B5%20%D0%BC%D0%BE%D0%B4%D0%B5%D0%BB%D0%B8%20%D0%B4%D0%BB%D1%8F%20%D0%BA%D0%BB%D0%B0%D1%81%D1%81%D0%B8%D1%84%D0%B8%D0%BA%D0%B0%D1%86%D0%B8%D0%B8.pdf. assessed: 11 November 2021.

[2] *Toxic Comment Classification Challenge*. URL: https://www.kaggle.com/c/jigsaw-toxic-comment-classification-challenge. accessed: 11 November 2021.

[3] *Stochastic gradient descent*. URL: https://neerc.ifmo.ru/wiki/index.php?title=%D0%A1%D1%82%D0%BE%D1%85%D0%B0%D1%81%D1%82%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B8%D0%B9_%D0%B3%D1%80%D0%B0%D0%B4%D0%B8%D0%B5%D0%BD%D1%82%D0%BD%D1%8B%D0%B9_%D1%81%D0%BF%D1%83%D1%81%D0%BA. accessed: 11 November 2021.

[4] *Machine learning (lecture course)*. URL: machinelearning.ru/wiki/index.php?title=%D0%9C%D0%B0%D1%88%D0%B8%D0%BD%D0%BD%D0%BE%D0%B5_

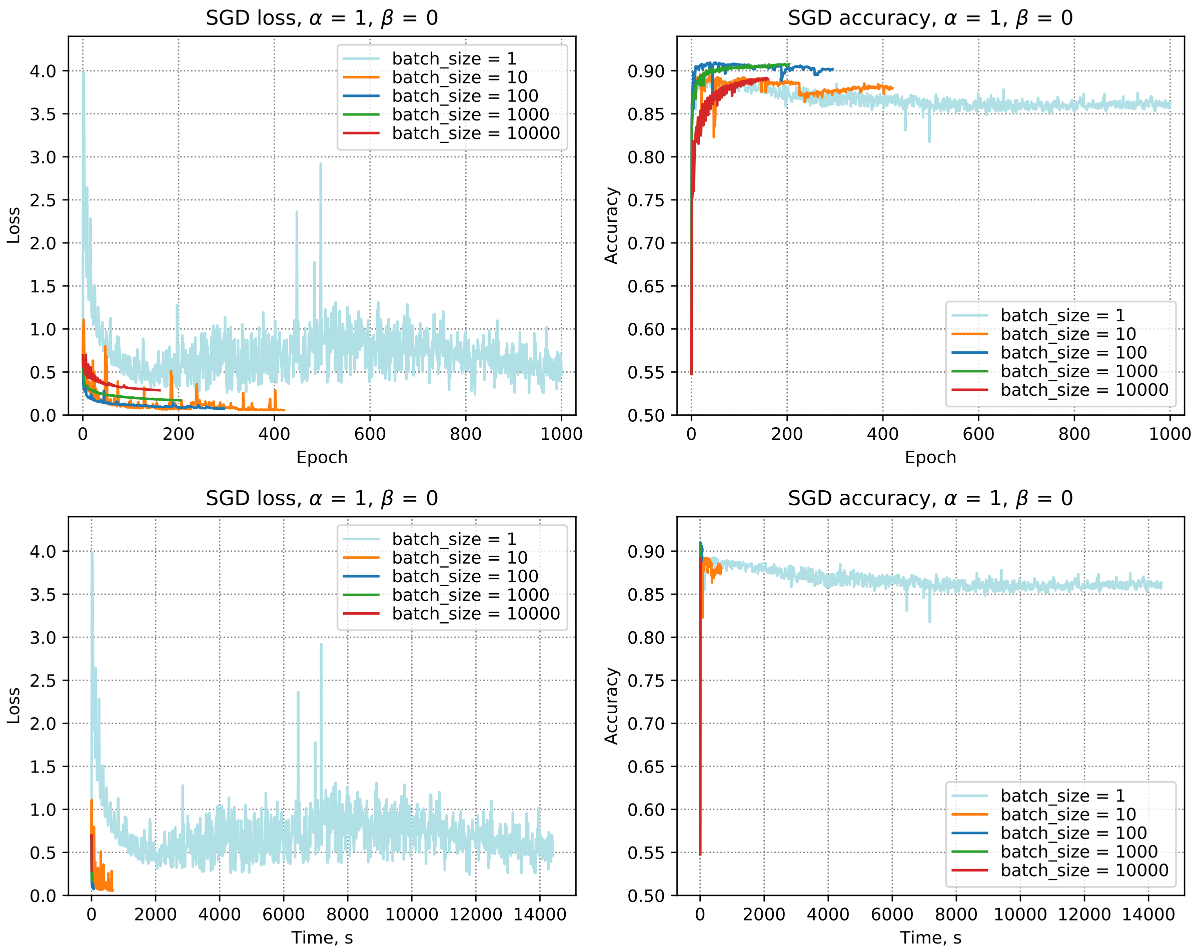

Рис. 13: Стохастический градиентный спуск, $\beta = 0$

%D0%BE%D0%B1%D1%83%D1%87%D0%B5%D0%BD%D0%B8%D0%B5_(%D0%BA%D1%83%D1%80%D1%81_%D0%BB%D0%B5%D0%BA%D1%86%D0%B8%D0%B9,_%D0%9A.%D0%92.%D0%92%D0%BE%D1%80%D0%BE%D0%BD%D1%86%D0%BE%D0%B2).

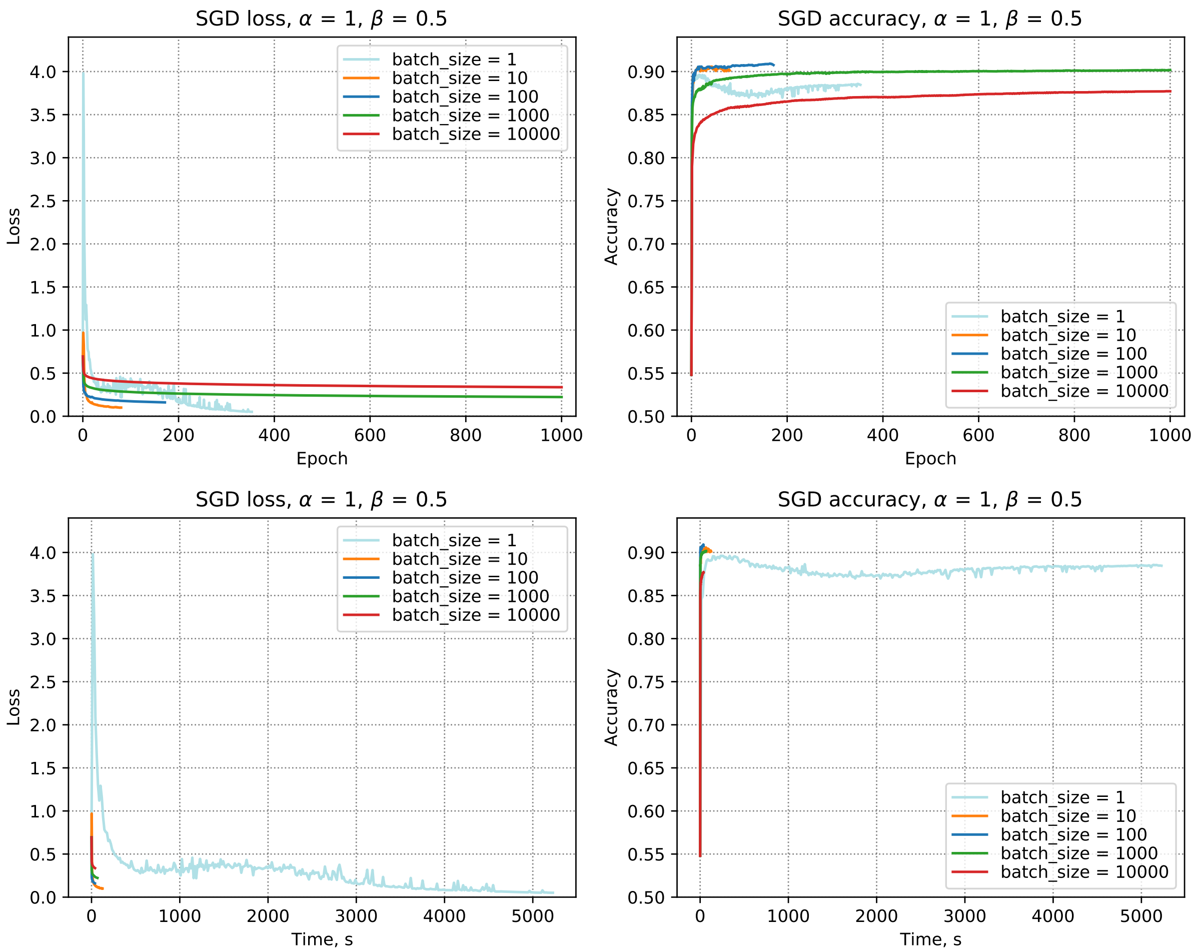

Рис. 14: Стохастический градиент, β = 0.5