Отчет по заданию 2

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

курс «Практикум на ЭВМ»
кафедра ММП ВМК МГУ

Михеев Борис, 317 группа

5 ноября 2021 г.

Формулировка задания

В данном задании требуется реализовать линейный классификатор на языке `Python`, реализовать логистическую регрессию с использованием метода градиентного спуска, провести бинарную классификацию комментариев из обсуждений английской Википедии: является ли комментарий токсичным или нет. Помимо этого требуется провести ряд экспериментов для исследования зависимости качества классификации от используемых методов и их параметров, от видов преобразований текстовых данных.

Теоретическая часть

Общая постановка

В задании рассматривается задача бинарной классификации. Пусть $X = \{x_i, y_i\}_{i=1}^{l}$ — обучающая выборка, $x_i \in \mathbb{R}^d$ — векторы признаковых описаний объектов, $y_i \in \{-1, 1\}$ - метки классов объектов. Для удобства считаем, что в множестве признаков имеется константный. Тогда в используемой линейной модели a ответ для объекта x может быть получен как $a(x, w) = sign\,\langle x, w\rangle$, где $w \in \mathbb{R}^d$ - вектор весов линейной модели a. Для оценки качества классификации объекта используют т. н. *отступ объекта* x $M(x)$ как меру его «погруженности» в свой класс: $M(x) = \langle x, w\rangle\, y$, где y - ответ для объекта x.

Логистическая регрессия

В задаче логистической регрессии используется следующая функция потерь:

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

Требуется найти w, минимизирующее эмпирический риск. В таком случае имеем задачу:

$$Q(X,w) = \frac{1}{l}\sum_{i=1}^{l}\mathcal{L}\left(M_i(w)\right) = \frac{1}{l}\sum_{i=1}^{l}\log\left(1+e^{-\langle x_i,w\rangle y_i}\right) \to \min_w.$$

Можно также применить регуляризацию для борьбы с переобучением, например, L2-регуляризацию. Тогда задача примет вид:

$$Q(X,w) = \frac{1}{l}\sum_{i=1}^{l}\log\left(1+e^{-\langle x_i,w\rangle y_i}\right) + \frac{\lambda}{2}\|w\|^2 \to \min_w,$$

где λ - коэффициент регуляризации.

Вычислим градиент данной функции потерь согласно правилам векторного и матричного дифференцирования:

$dQ(X,w) = \frac{1}{l}\sum_{i=1}^{l}(d\log\left(1+e^{-\langle x_i,w\rangle y_i}\right)) + \frac{\lambda}{2}d\|w\|^2 = \frac{1}{l}\sum_{i=1}^{l}(\frac{1}{1+e^{-\langle x_i,w\rangle y_i}}d(1+e^{-\langle x_i,w\rangle y_i})) +$
$+ \frac{\lambda}{2}dw^Tw = -\frac{1}{l}\sum_{i=1}^{l}(\frac{e^{-\langle x_i,w\rangle y_i}}{1+e^{-\langle x_i,w\rangle y_i}}d(\langle x_i,w\rangle\, y_i)) + \lambda w^Tdw = -\frac{1}{l}\sum_{i=1}^{l}(\frac{y_ix_i^T}{1+e^{\langle x_i,w\rangle y_i}}dw) + \lambda w^Tdw =$
$= \left\langle -\frac{1}{l}\sum_{i=1}^{l}(\frac{y_ix_i}{1+e^{\langle x_i,w\rangle y_i}}) + \lambda w, dw\right\rangle \Rightarrow$

$$\Rightarrow \nabla Q(X,w) = -\frac{1}{l}\sum_{i=1}^{l}\frac{y_ix_i}{1+e^{\langle x_i,w\rangle y_i}} + \lambda w$$

В данной задаче можно рассматривать следующую вероятностную модель. Пусть предполагается, что $y_i \in \{0,1\}$, и в качестве ответа модели на объекте x берется вероятность его принадлежности к классу 1:

$$a(x) = \mathbb{P}(y=1|x,w) = \frac{1}{1+e^{-\langle w,x\rangle}}$$

Оптимальный вектор весов w можно искать с помощью метода максимального правдоподобия:

$$-L(w) = -\log\prod_{i=1}^{l}\mathbb{P}(y_i|x_i,w) = \sum_{i=1}^{l}(-y_i\log a(x_i) - (1-y_i)\log(1-a(x_i))) \to \min_w$$

Подставив в данную задачу выражение для $a(x_i)$, получим задачу, эквивалентную задаче с функционалом Q(X,w) и переобозначенными метками классов. Таким образом данный вероятностный подход эквивалентен исходной задаче с минимизацией эмпирического риска.

Мультиномиальная регрессия

Перейдем к задаче мультиномиальной логистической регрессии - обобщению логистической регрессии на случай k классов. Строятся k линейных моделей $a_1(x)$ ... $a_k(x)$, $a_i(x) = sign\,\langle x, w_i\rangle$, каждая из которых выдает оценку принадлежности объекта x классу i. Пусть вероятность того, что объект x принадлежит классу c, $c \in \{1...k\}$, то есть что ответ y для него равен c, может быть найдена как:

$$\mathbb{P}(y=c|x) = \frac{e^{\langle w_c,x\rangle}}{\sum_{i=1}^{k}e^{\langle w_i,x\rangle}}$$

Тогда оценки моделей $a_i(x)$ можно перевести в вероятностное распределение при помощи функции `softmax`:

$$softmax(x_1, ..., x_k) = (\frac{e^{x_1}}{\sum_{i=1}^{k} e^{x_i}}, ..., \frac{e^{x_k}}{\sum_{i=1}^{k} e^{x_i}})$$

Используя метод максимального правдоподобия и L2-регуляризацию, получим задачу минимизации по w следующего функционала:

$$Q(X, w) = -\frac{1}{l}\sum_{i=1}^{l} \log \mathbb{P}(y_i|x_i) + \frac{\lambda}{2}\sum_{i=1}^{k} \|w_i\|^2 \to \min_{w_1,...,w_k} \Leftrightarrow$$

$$\Leftrightarrow Q(X, w) = -\frac{1}{l}\sum_{i=1}^{l} \log \frac{e^{\langle w_{y_i}, x_i\rangle}}{\sum_{s=1}^{k} e^{\langle w_s, x_i\rangle}} + \frac{\lambda}{2}\sum_{i=1}^{k} \|w_i\|^2 \to \min_{w_1,...,w_k}$$

Вычислим градиент по произвольному вектору весов w_t:

$\frac{dQ(X,w)}{dw_t} = -\frac{1}{l}\sum_{i=1}^{l}(\frac{d}{dw_t}(\langle w_{y_i}, x_i\rangle - \log\sum_{s=1}^{k} e^{\langle w_s, x_i\rangle}) + \frac{\lambda}{2}\sum_{i=1}^{k}\frac{d}{dw_t}(w_i^T w_i) =$
$= -\frac{1}{l}\sum_{i=1}^{l}([y_i = t]x_i^T dw_t - \frac{e^{\langle w_t, x_i\rangle} x_i^T dw_t}{\sum_{s=1}^{k} e^{\langle w_s, x_i\rangle}}) + \lambda w_t^T dw_t \Rightarrow$

$$\Rightarrow \nabla_{w_t} Q(X, w) = -\frac{1}{l}\sum_{i=1}^{l}([y_i = t] - \frac{e^{\langle w_t, x_i\rangle}}{\sum_{s=1}^{k} e^{\langle w_s, x_i\rangle}})x_i + \lambda w_t$$

Полный градиент функции в данном случае будет являться матрицей. Таким образом, объединив столбцы $\nabla_{w_i} Q(X, w)$, $i \in \{1...\text{k}\}$ в матрицу, получим полный градиент функции потерь.

Покажем, что при $k = 2$ классах задача мультиномиальной логистической регрессии сводится к бинарной логистической регрессии. Найдем явно вероятности принадлежности к классам 1 и 2:

$$\mathbb{P}(y = 1|x) = \frac{e^{\langle w_1, x\rangle}}{e^{\langle w_1, x\rangle} + e^{\langle w_2, x\rangle}}$$

$$\mathbb{P}(y = 2|x) = \frac{e^{\langle w_2, x\rangle}}{e^{\langle w_1, x\rangle} + e^{\langle w_2, x\rangle}}$$

Заметим, что $\mathbb{P}(y = 1|x) + \mathbb{P}(y = 2|x) = 1$, $\mathbb{P}(y = 1|x) = \frac{1}{1+e^{\langle w_2 - w_1, x\rangle}}$, $\mathbb{P}(y = 2|x) =$
$= \frac{1}{1+e^{\langle w_1 - w_2, x\rangle}}$. Таким образом, задача переходит в бинарную логистическую регрессию, ее вектор весов: $w_1 - w_2$. Переобозначив классы как -1 и $+1$ и положив, что модель a_1 в задаче мультиномиальной регрессии дает оценку принадлежности классу $+1$, и придав ей вектор весов w_1, получим задачу бинарной логистической регресии с тем же вектором весов $w_1 - w_2$. При этом при использовании L2 регуляризации задачи не будут эквивалентны в силу наличия слагаемого регуляризации.

Эксперименты

В задании необходимо реализовать логистическую регрессию и решить задачу бинарной классификации комментариев, используя при этом классический и стохастический градиентный спуск. Также требуется исследовать зависимость качества классификации в зависимости от параметров алгоритмов и способов предобработки данных. Для оценки качества классификации используется метрика `accuracy` (доля правильных ответов). В ходе экспериментов работа алгоритмов оценивается на отложенной выборке, полученной разбиением случайно перемешанной исходной обучающей выборки в соотвношении 7/3 на обучение и валидацию соответственно. Итоговое качество измеряется на исходной тестовой выборке.

Предоставленные для работы тексты - комментарии из обсуждения англоязычной Википедии, принадлежащие двум классам - токсичный и нетоксичный комментарий. Класс 1 соответствует токсичному комментарию, -1 - нетоксичному. Стоит отметить несбалансированность классов в задаче, токсичных комментариев примерно треть.

Предварительная обработка

Тексты обучающей и тестовой выборок изначально приводятся к нижнему регистру, символы, отличные от букв и цифр, заменяются на пробелы. Далее текст разбивается на слова, преобразуется с помощью `sklearn.feature_extraction.text.CountVectorizer` в разреженную матрицу, в которой в строке i и столбце j находится число раз, которое слово с номером j встретилось в документе i. Значение параметра `min_df` было выбрано равным 0.0001. Данный параметр отвечает за удаление из получаемого словаря редких слов. В данном случае не будут рассмтариваться слова, встречающиеся менее чем в 0.0001 доле от всех документов,
т. е. менее чем в 5 документах в случае имеющейся обучающей выборки.

Исследование поведения градиентного спуска

Зависимость от параметров темпа обучения

В решаемой задаче для классического и стохастического градиентного спуска используется следующий темп обучения (`learning rate`):

$$\eta_k = \frac{\alpha}{k^\beta},$$

где α, β - гиперпараметры, k - номер итерации алгоритма (эпохи а случае стохастического градиентного спуска). Для исследования зависимости работы алгоритмов от данных параметров проведем перебор различных комбинаций их значений: α рассматривается из логарифмической шкалы по основанию 10 от 10^{-3} до 1 длиной 10, β берется из равномерной сетки от 0 до 2 с шагом 0.5. Ниже приведены графики функции потерь и точности для различных α при фиксированных β для классического градиентного спуска. Используются параметры `tolerance`=10^{-5}, `max_iter`=2000.

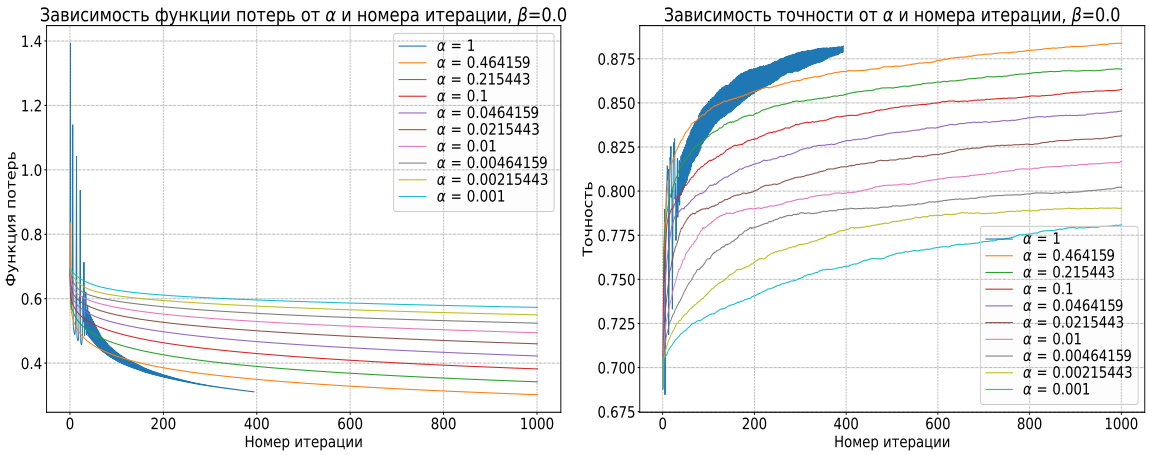

Рис. 1: Функция потерь и точность для градиентного спуска, $\beta = 0$

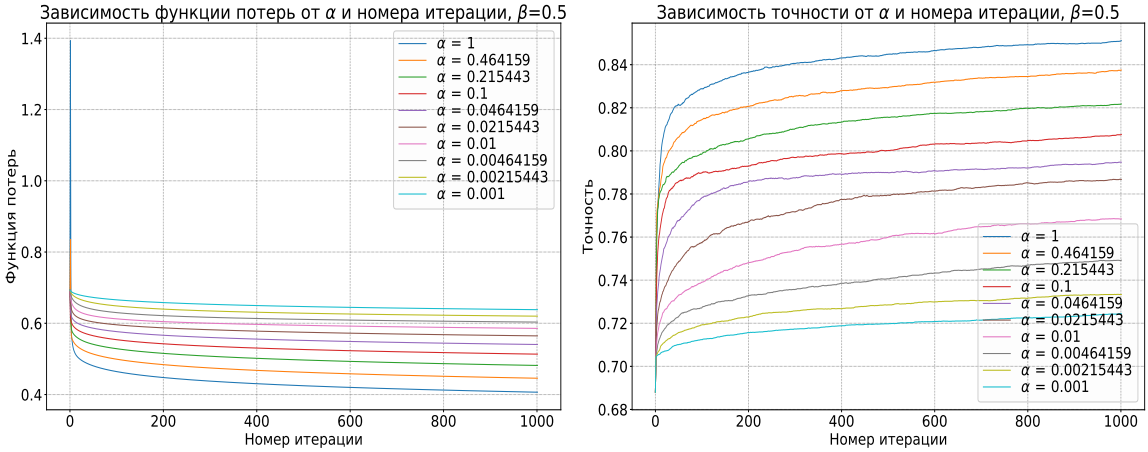

Рис. 2: Функция потерь и точность для градиентного спуска, $\beta = 0.5$

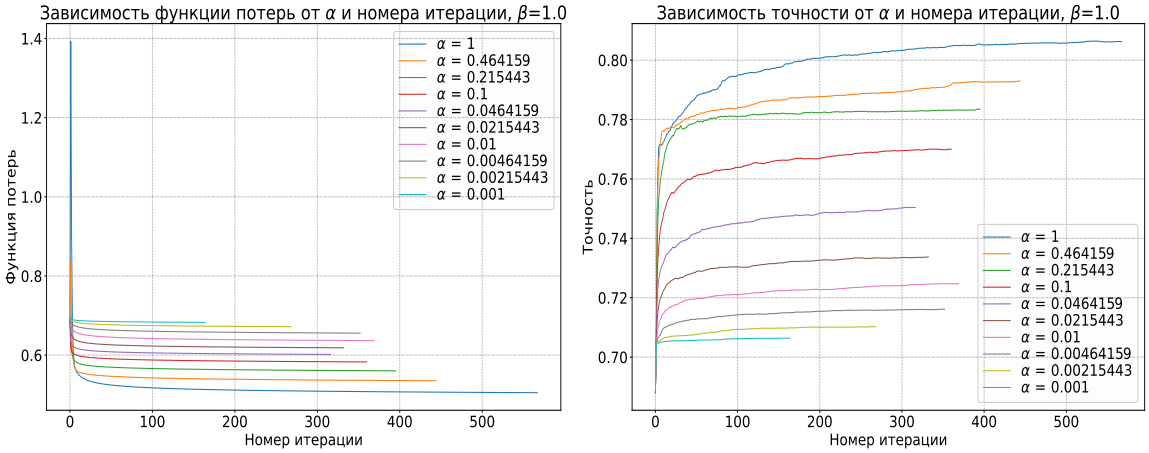

Рис. 3: Функция потерь и точность для градиентного спуска, $\beta = 1$

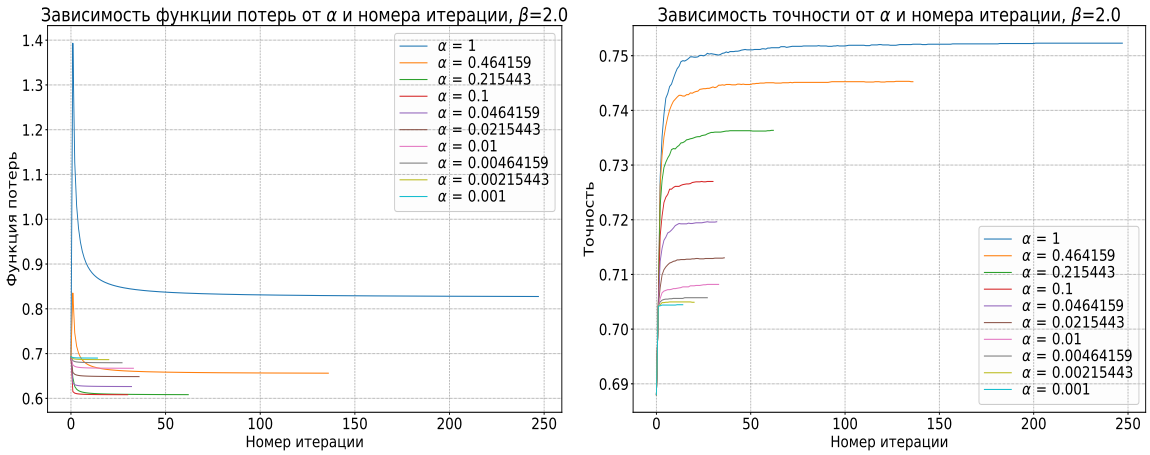

Рис. 4: Функция потерь и точность для градиентного спуска, $\beta = 2$

При $\beta = 0$ темп обучения постоянный, при остальных значениях - динамический, убывает с ростом номера итерации k. Для постоянного η наилучшие результаты достигаются для $\alpha = 1$, в том числе и по скорости сходимости. Однако графики функции ошибки и точности при этом сильно осциллируют, но с ростом числа итераций достигаются лучшие значения. При меньших значениях α значение функции потерь больше, точность меньше, поведение графиков более стабильное, скорость сходимости меньше. При ненулевом β графики в целом более гладкие и меньше осциллируют, быстрее «изламываются», выходят на асимптоту. Наблюдается прямая зависимость точности от α и обратная зависимость функции потерь от α, в целом монотонная. Исключение составляет случай $\beta = 2$ и $\alpha = 1$, при нем в сравнении с остальными α достигается наибольшее значение функции потерь, но и точность также максимальна. При больших β алгоритм начинает сходиться значительно быстрее, но при этом падает точность с ростом β. Это связано с тем, что алгоритм начинает слишком быстро сходиться при использовании критерия останова по разности значений функции потерь на соседних итерациях, что можно наблюдать на графиках. Коэффициент перед градиентом в итерационном процессе будет в таком случае быстрее стремиться к нулю, и алгоритм не успеет сойтись к оптимуму. В целом наилучшие результаты достигались при $\alpha = 1$ и $\beta = 0$, т. е. константном темпе обучения. Однако при этом графики сильно осциллировали, что может привести к получению нестабильных весов модели. В целом для получения схожих результатов и преодоления сопутствующих проблем можно брать значения β около 0, а α взять немного больше 1. Тогда графики должны сгладиться, точность, функция потерь и скорость сходимости останутся примерно теми же, а получаемое оптимальное w будет более стабильно. Можно и увеличивать α, но это может привести к большей нестабильности весов и не скажется сильно на точности.

Проведем те же эксперименты для стохастического градиентного спуска. Стоит отметить, что в приведенных графиках удобнее откладывать по оси абсцесс не номер итерации, а приближенный номер эпохи, т. е. отношение числа обработанных алгоритмом объектов к размеру исходной выборки. Так как на каждой итерации используется не вся выборка, а лишь ее часть, то понятие итерации имеет разный смысл для классического и стохастического методов. Используются параметры `batch_size`=1000, `tolerance`=10^{-5}, `max_iter`=2000.

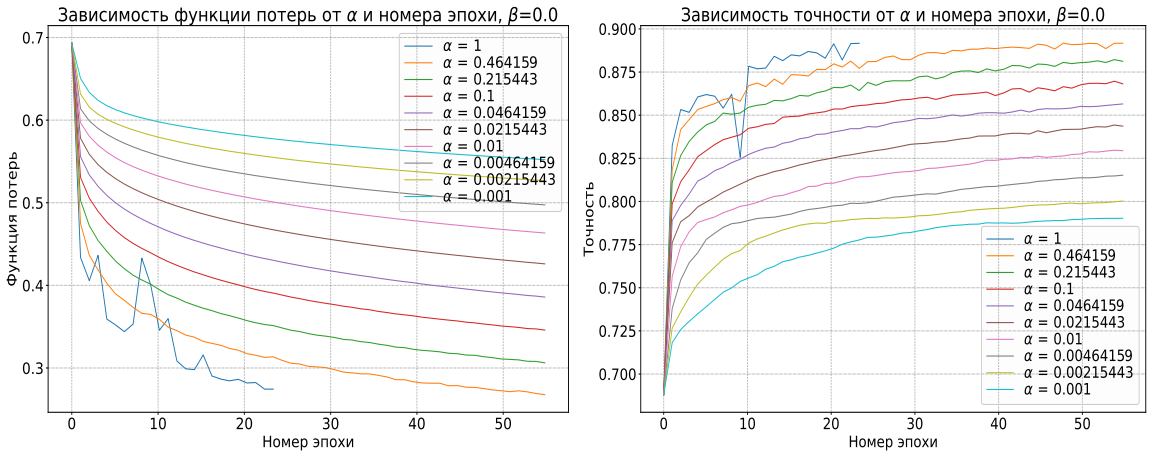

Рис. 5: Функция потерь и точность для стохастического градиентного спуска, $\beta = 0$

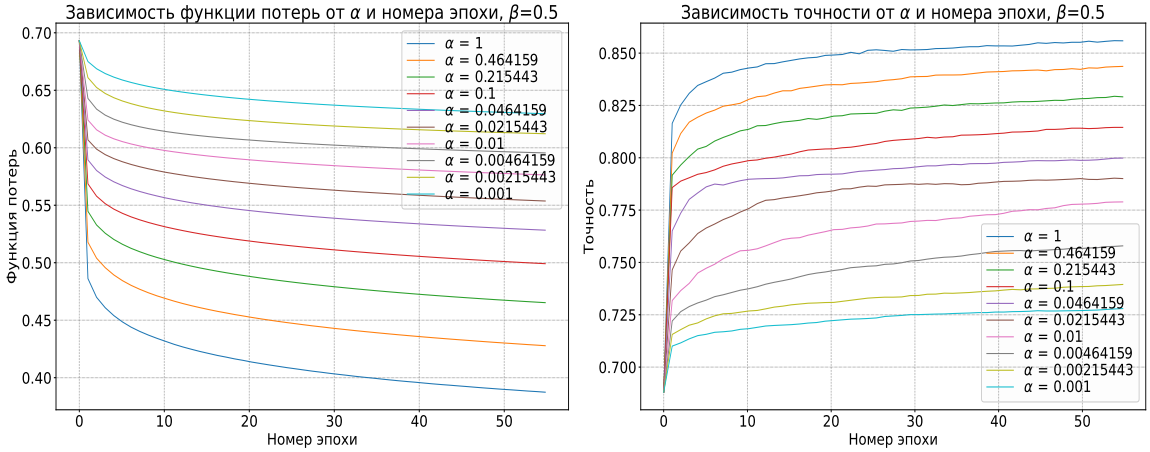

Рис. 6: Функция потерь и точность для стохастического градиентного спуска, $\beta = 0.5$

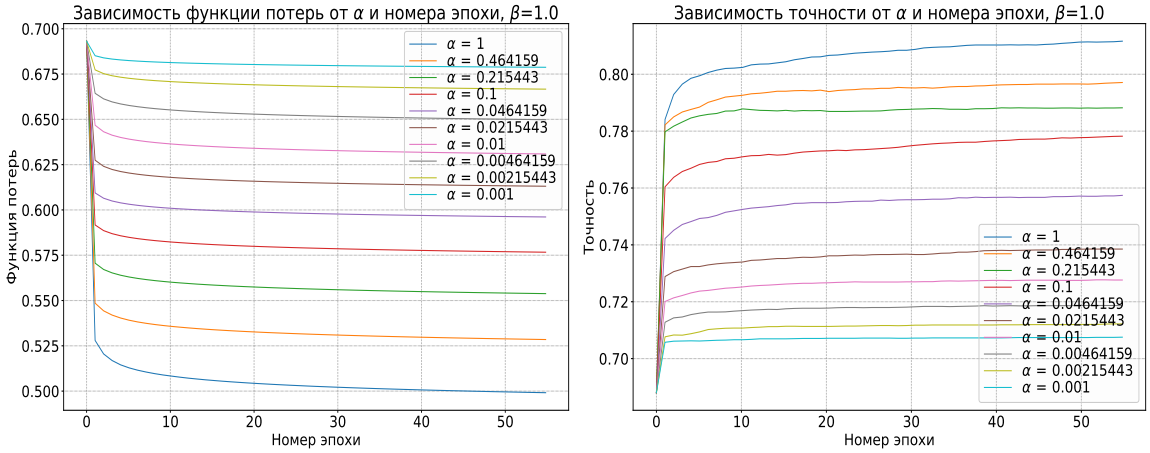

Рис. 7: Функция потерь и точность для стохастического градиентного спуска, $\beta = 1$

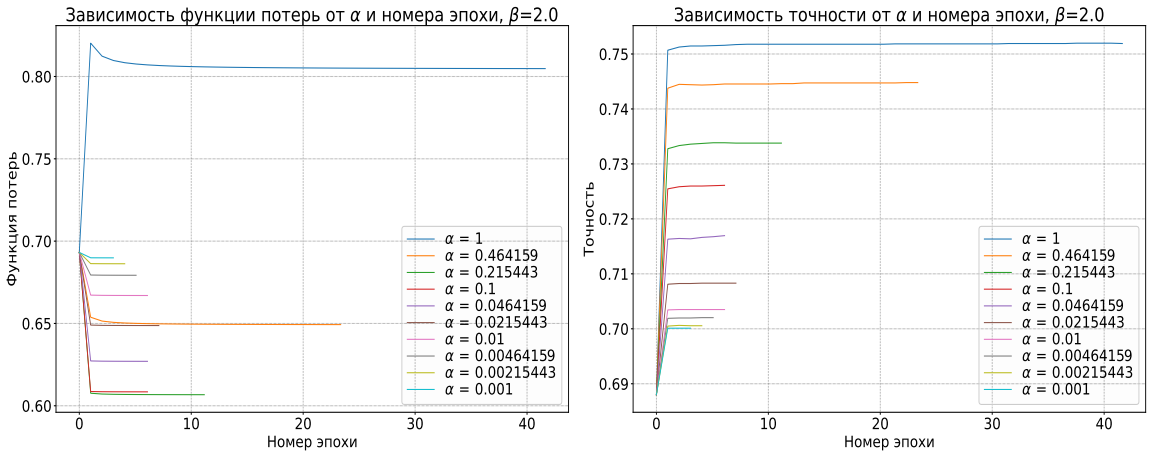

Рис. 8: Функция потерь и точность для стохастического градиентного спуска, $\beta = 2$

Результаты схожи с классическим алгоритмом. С ростом α достигается большая точность и меньшее значение функции потерь. Однако при $\beta = 2$ и $\alpha = 1$ значение функции ошибок будет заметно больше, чем при других α. При $\beta < 1$ при больших α алгоритм сходится быстрее, при $\beta >= 1$ сходится быстрее при меньших α. В целом графики меньше осциллируют, и так же становятся более стабильными с ростом β. Это может быть связано с тем, что темп обучения, если он динамический, уменьшается не на всей выборке, а на каждом батче, и алгоритм лучше будет идти к минимуму. При увеличении β сходимость быстрее, но точность падает. Для сравнительно небольших β алгоритм долго не будет сходиться при приближении к асимптоте по точности. В целом большое число итераций допустимо, так как алгоритм обучается на фрагментах выборки, и для просмотра большего их числа потребуется большее число шагов. Лучшие результаты по точности, сходимости и функции потерь также достигаются при $\alpha = 1$ и $\beta = 0$, и графики при этом заметно меньше осциллируют, что говорит о получении более стабильных весов модели. Можно использовать идеи, аналогичные классическому алгоритму, для получения наилучших результатов и стабильного оптимального вектора весов. Также при константном η, т. е. при $\beta = 0$, качество классификации несколько лучше, чем для классического алгоритма. Это может быть следствием того, что в стохастическом алгоритме вектор весов меняется на сравнительно небольшие значения, т. к. на каждой итерации рассматривается лишь один батч. В обычном алгоритме используется вся выборка, и градиент может оказаться большим по модулю.

Зависимость от начального приближения вектора весов

Рассмотрим зависимость результата работы алгоритма от выбора начального приближения вектора весов. Будем рассматривать наиболее используемые способы инициализации весов: нулевой вектор θ, сэмплированные из равномерного распределения на отрезках $[-\frac{1}{d}, \frac{1}{d}]$ и $[-\frac{1}{\sqrt{d}}, \frac{1}{\sqrt{d}}]$, где d - размерность признакового пространства, а также w: $w_i = \frac{\langle y, f_i \rangle}{\langle f_i, f_i \rangle}$, где y - вектор ответов для обучающей выборки, f_i - столбец значений i-ого признака в обучающей выборке. Рассмотрим результаты для обычного градиентного спуска. Отметим, что в предыдущих экспериментах использовалось нулевое начальное приближение.

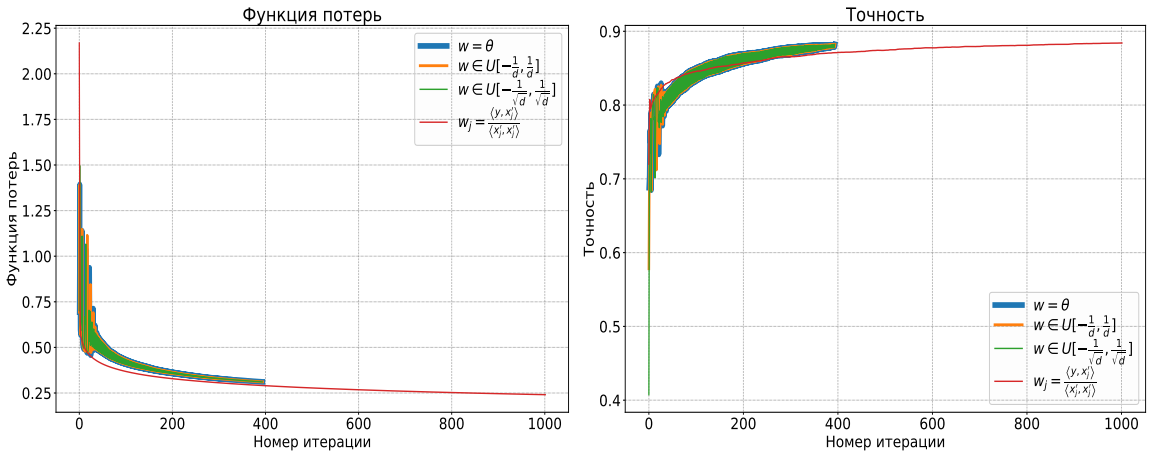

Рис. 9: Функция потерь и точность для градиентного спуска в зависимости от различных начальных приближений вектора весов

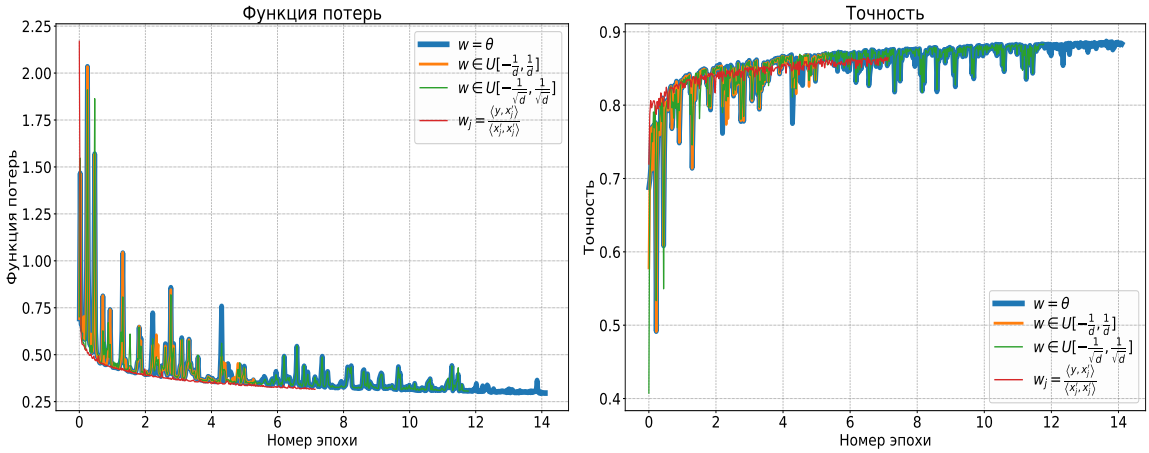

Рис. 10: Функция потерь и точность для стохастического градиентного спуска в зависимости от различных начальных приближений вектора весов

Для нулевого вектора и векторов из равномерных распределений результаты примерно одинаковы, графики сильно наслаиваются друг на друга. Поначалу они сильно осциллируют. Для вектора с $w_i = \frac{\langle y, f_i \rangle}{\langle f_i, f_i \rangle}$ значения чуть лучше, график более стабильный, но алгоритм сходится дольше.

Для стохастического градиентного спуска графики менее стабильны, результаты для нулевого вектора и сэмплированных из равномерных распределений по прежнему схожи, но графики значительно больше осциллируют. Для $w_i = \frac{\langle y, f_i \rangle}{\langle f_i, f_i \rangle}$ результаты также чуть лучше остальных, но график более стабильный, и алгоритм сходится быстрее.

В целом данные приближения дают очень близкие приемлемые результаты, сложно выделить явно лучший среди них. Также стоит отметить, что на первых итерациях обоих алгоритмов значения функции ошибок еще достаточно велики. Это может быть связано с тем, что выбранное начальное приближение находилось далеко от оптимальной точки, и далее проходились точки с возрастающим значением ошибки, что привело к осцилляциям местами. Таким образом, истинное распределение весов далеко от равномерного. Учитывая несбалансированность классов в решаемой задаче, это вполне закономерно, и большинство слов в текстах будут нетоксичными и будут иметь веса с малыми модулями.

Зависимость от размера подвыборки (батча)

Переберем размеры подвыборки от 1 до 10000 по логарифмической шкале по основанию 10 и рассмотрим результаты работы стохастического градиентного спуска.

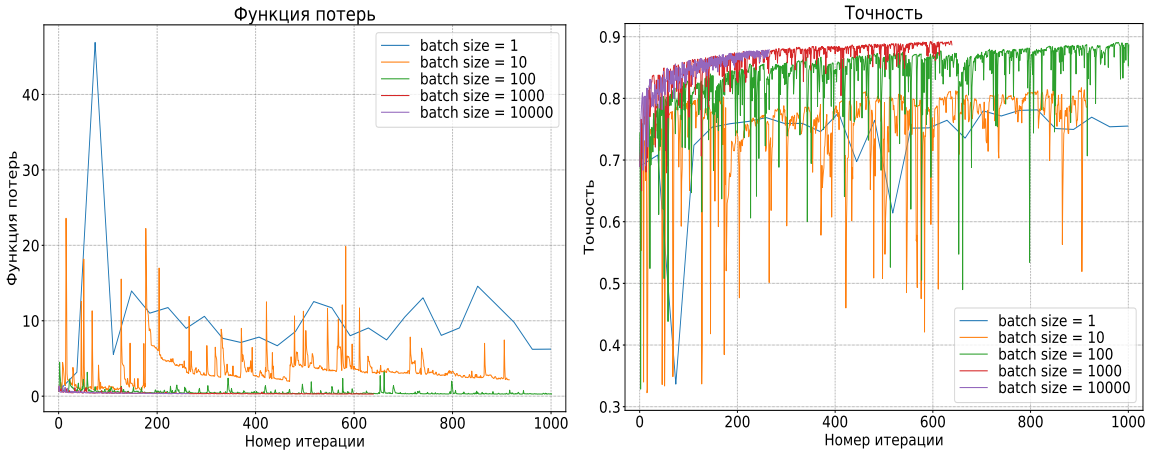

Рис. 11: Функция потерь и точность для стохастического градиентного спуска в зависимости от размера батча, $\alpha = 1$, $\beta = 0$

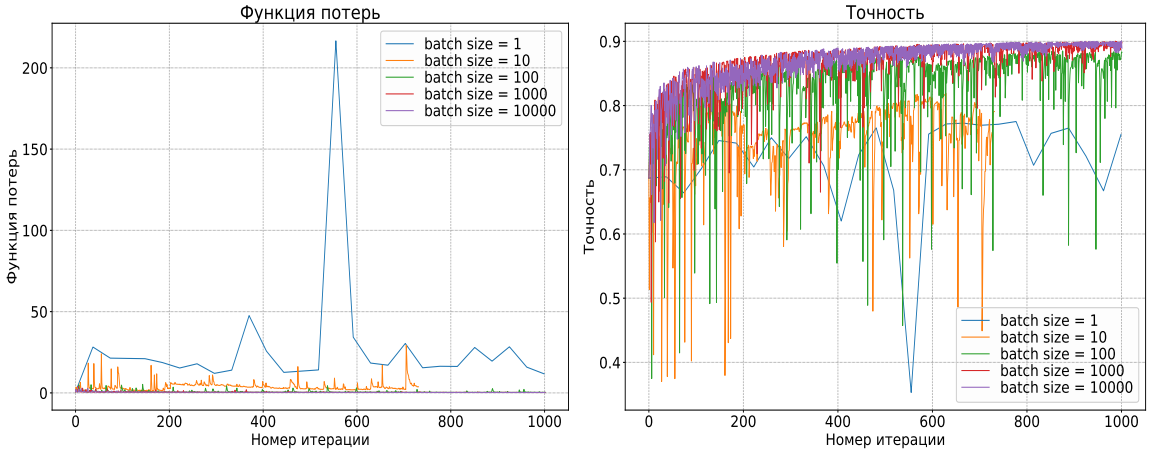

Рис. 12: Функция потерь и точность для стохастического градиентного спуска в зависимости от размера батча, $\alpha = 1.75$, $\beta = 0.025$

При малом размере батча алгоритм совершает большое число итераций, графики сильно осциллируют, хотя и функция потерь в целом уменьшается. Скачки графиков можно объяснить тем, что в экспериментах статистика о работе алгоритма заносилась в словарь примерно раз в одну эпоху, и при малом размере батча обновления происходили крайне редко, а изменения успевали накопиться. При β отличных от 0, например, при $\alpha = 1.75$, $\beta = 0.025$, т. е. при динамическом темпе обучения, графики также нестабильны, алгоритмы дольше сходятся. Лучшие результаты достигаются при больших размерах подвыборки. При размерах батча, близких к размеру всей выборки, алгоритм станет по сути стремиться к своей классической версии, так как каждый раз берется почти вся выборка.

Зависимость от времени работы

Приведем графики зависимости функции потерь и точности от времени для обычного и стохастического градиентного спуска для различных значений α и β. По сути они отличаются от графиков из соответствующего пункта экспериментов лишь разметкой оси абсцисс. Для наглядности приведем некоторые из них, для константного и динамического темпа обучения.

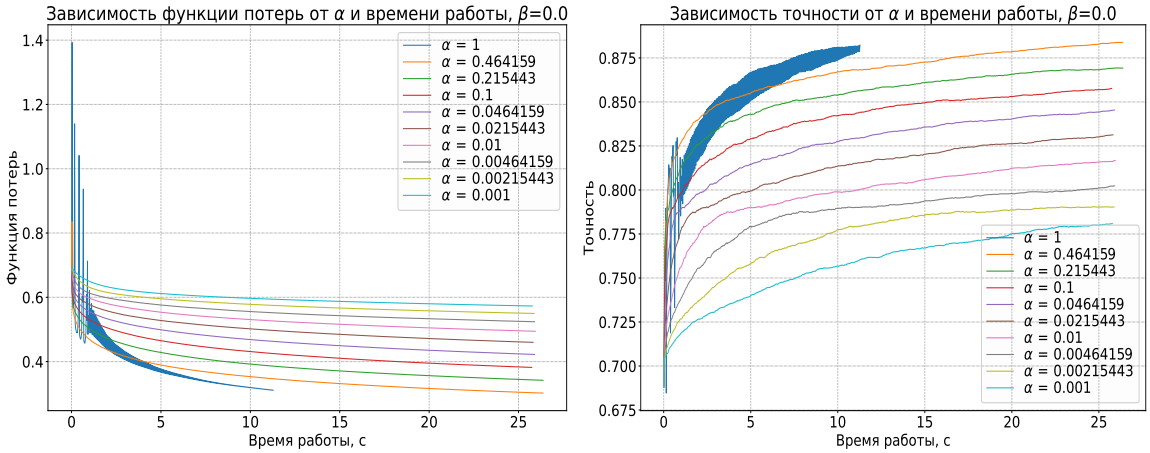

Рис. 13: Функция потерь и точность для градиентного спуска в зависимости от времени работы, $\beta = 0$

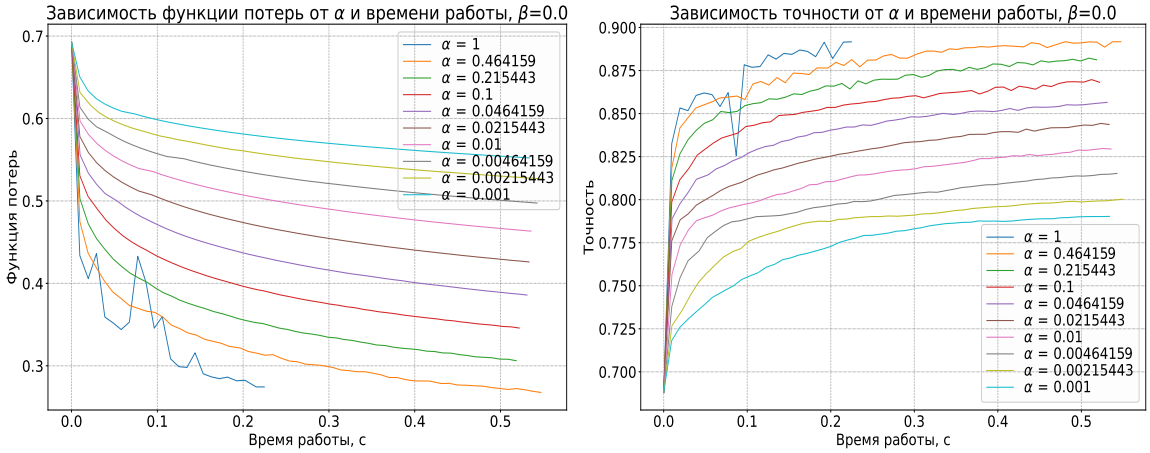

Рис. 14: Функция потерь и точность для стохастического градиентного спуска в зависимости от времени работы, $\beta = 0$

В целом со временем алгоритмы приближаются к асимптотам по точности и функции потерь, даже в отстутствии сходимости. Стохастический вариант работает быстрее классического. Также из предыдущих пунктов и соответствующих построенных ранее графиков можно заключить, что использование малых батчей в стохастическом градиентном спуске ведет к очень долгой работе алгоритма. С ростом размера батча время работы сокращается. Быстрее всего стохастический алгоритм сходится при больших значениях β в силу убывания темпа обучения.

Исследование влияния регуляризации

Рассмотрим, как влияет использование регуляризации на качество классификации. Отметим, что до этого во всех экспериментах регуляризация не использовалась. В рассматриваемой задаче предлагается использовать L2 регуляризацию. На основании выкладок, приведенных в теоретической части работы, можно заметить, что в градиент функции потерь вклад регуляризации войдет в виде линейного слагаемого λw, соответственно, в итерационном процессе обновления вектора весов это выльется в добавочное слагаемое $-\eta_k \lambda w_k$, остальные слагаемые останутся прежними. Также выражение $\frac{\lambda}{2}\|w\|^2$ будет учтено при подсчете функции потерь и разности ее значений на каждой итерации. На основании этого можно исследовать влияние регуляризации на работу алгоритма при фиксированных α и β, так как их отношение войдет в добавочное слагаемое в обновлении вектора весов в виде множителя. Рассмотрим влияние регуляризации на работу классического и стохастического градиентного спуска при следующих параметрах, отобранных как наилучшие в результате подбора: $\alpha = 1.75$, $\beta = 0.025$, `tolerance`$=10^{-7}$, максимальное число итераций - 2000, размер батча - 1000. Коэффициент регуляризации λ будем перебирать по логарифмической шкале от 10^{-1} до 10^{-7}. Брать большие значения λ нецелесообразно, так как это сильно упростит модель, ибо она будет стремиться брать близкие к нулю веса.

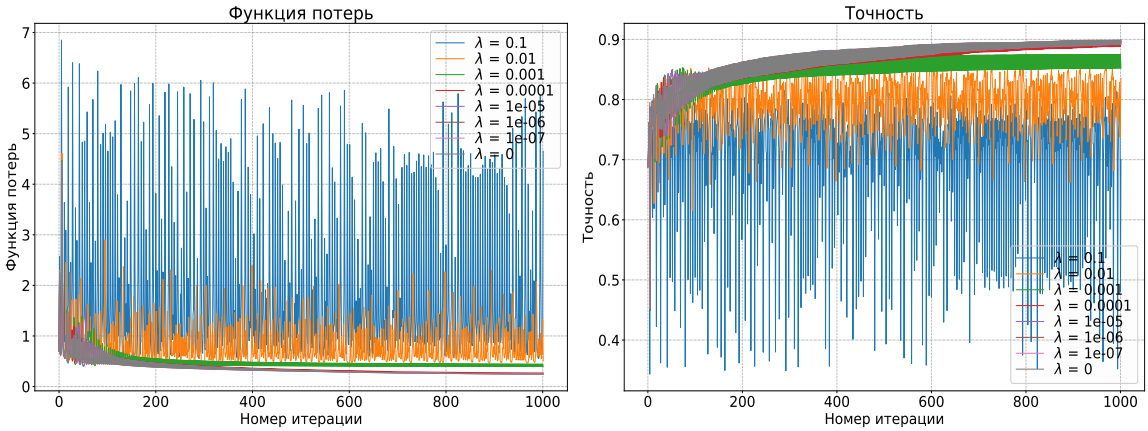

Рис. 15: Функция потерь и точность для градиентного спуска в зависимости от коэффициента регуляризации λ

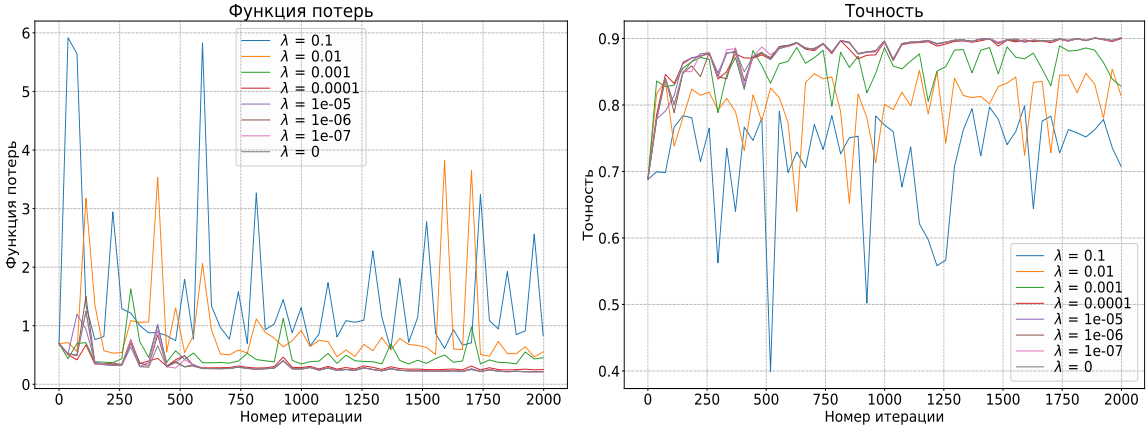

Рис. 16: Функция потерь и точность для стохастического градиентного спуска в зависимости от коэффициента регуляризации λ

Таким образом, при больших значениях λ графики сильно осциллируют, точность падает с ростом λ, а значения функции потерь возрастают и сильно колеблются. Это может быть связано с тем, что слагаемые λw и $-\eta_k \lambda w_k$ входят непосредственно в градиент и градиентный шаг, из за-чего значения весов и функции потерь сильно меняются от итерации к итерации, а разность весов войдет в разность функций потерь. Для классического градиентного спуска осцилляции графиков заметно сильнее, чем для стохастического, так как на каждой итерации градиент берется по всей выборке, а не по батчу, и значение градиента может оказаться большим по модулю. В целом лучшие значения с использованием регуляризации практически не отличаются от таковых для алгоритмов без нее, что можно заметить и по графику.

Сравнение классического и стохастического алгоритма градиентного спуска

Стохастический градиентный спуск по сути является модернизацией, обобщением обычного градиентного спуска. Варьирование размера батча и его случайный выбор из всей выборки позволяют стохастическому градиентному спуску быть более мощной и гибкой моделью. Может достигаться лучшая сходимость в сравнении со стандартным вариантом, так как разница функций ошибки и темп обучения контролируются на каждом батче, а не на всех данных, можно тщательнее контролировать сходимость. Это может дать прирост в скорости работы. Также в стохастическом алгоритме не нужно хранить всю выборку в памяти, а лишь ее часть, что полезно в работе с данными большого объема. Таким образом, выгоднее использовать стохастический градиентный спуск в сравнении с обычным.

Выбор наилучшей модели

Таким образом, на основании полученых результатов экспериментов можно сказать, что лучшие результаты по точности, сходимости и функции потерь дает стохастический градиентный спуск. В результате подбора параметров и на основе результатов эксперимето в качестве лучшей модели выберем стохастический градиентный спуск со следующими параметрами: $\alpha = 1.75$, $\beta = 0.025$, `tolerance`=10^{-7}, максимальное число итераций - 2000, размер батча - 1000, $\lambda = 0$

Применение лемматизации и удаление стоп-слов

Применим преобразование лемматизации к текстам - преобразование, при котором слова приводятся к своей т. н. нормальной (словарной) форме - лемме. Также можно провести удаление стоп-слов. Это часто встречаемые слова, в основном предлоги, союзы, междометия и другие часто встречаемые части речи, в основном служебные, не несущие смысла.

Применим соответствующие преобразования к исходным данным с помощью `WordNetLemmatizer` из библиотеки `nltk`. Далее вновь применим `CountVectorizer` с тем же параметром `min_df`, равным 0.0001. Применим также и удаление стоп-слов.

	исходный текст	лемматизация	удаление стоп-слов	лемматизация и удаление стоп-слов
число признаков	16050	14433	15907	14294
точность	0.85955	0.86182	0.86680	0.86796
время работы, с	0.65621	0.65090	0.52606	0.51419

Таблица 1: Значения точности, времени работы и размерности признакового пространтсва при различных преобразованиях текстовых данных

Сокращение числа признаков связано с «объединением» слов, являющихся по сути разными формами одного и того же слова.

Таким образом, можно заметить, что лемматизация значительно сокращает число признаков, качество при этом улучшается. Удаление стоп-слов также ведет к снижению числа признаков и улучшению точности. В совокупности оба преобразования позволяют неплохо улучшить точность и сократить время работы алгоритма.

Исследование представлений текста и их параметров

Рассмотрим две модели представления текста - `Tfidf` и `BagOfWords` и их влияние на работу алгоритмов. Использованная нами до этого модель представления текста — `BagOfWords`. В ней все множество документов представляется матрицей, где в строке i и столбце j находится число раз, которое слово с номером j встретилось в документе i.

В модели `Tfidf` учитывается популярность и распространенность слов, то есть общеупотребительные и служебные слова вроде местоимений, предлогов и т. д. не рассматриваются как несущие смысловую нагрузку и влияющие на классификацию текста.

Сравним модели при фиксированном `min_df`=0.0001 с применением лемматизации и без нее.

		число признаков	точность	время работы, с
без лемматизации	`BagOfWords`	16050	0.85954	0.66816
	`Tifidf`	16050	0.84905	0.47489
с лемматизацией	`BagOfWords`	14433	0.86182	0.63054
	`Tifidf`	14433	0.85369	0.46853

Таблица 2: Значения точности, времени работы и размерности признакового пространтсва при использовании различных текстовых представлений и лемматизации

Таким образом, лемматизация снижает число признаков и время работы, положительно влияет на точность для обоих моделей, `Tfidf` работает немного хуже, но быстрее чем `BagOfWords`

Рассмотрим, как качество, время работы и размер признакового пространства зависят от моделей представления текстов и параметров `min_df` и `max_df`. `max_df` отвечает за то, насколько часто встречаемые слова не попадут в словарь: если `max_df`=n, то не рассматриваются слова, присутствующие в доле документов, превышающей n. Рассмотрим результаты работы алгоритмов в зависимости от этих параметров.

`min_df`	**число признаков**	**BagOfWords**		**Tfidf**	
		точность	**время работы, с**	**точность**	**время работы, с**
10^{-6}	89368	0.86191	0.75631	0.84547	0.61254
10^{-5}	89368	0.86191	0.79944	0.84547	0.61731
10^{-4}	16050	0.85954	0.61978	0.84905	0.47975
10^{-3}	3736	0.85843	0.47326	0.85388	0.33348
10^{-2}	568	0.75411	0.39028	0.83671	0.29113

Таблица 3: Точность, время работы и размерность признакового пространтсва при различных значениях `min_df` для моделей `BagOfWords` и `Tfidf`

`max_df`	**число признаков**	**BagOfWords**		**Tfidf**	
		точность	**время работы, с**	**точность**	**время работы, с**
1.0	89368	0.86191	0.80564	0.84547	0.62469
0.75	89368	0.86191	0.81756	0.84547	0.62121
0.6	89367	0.86394	0.78062	0.84827	0.64315
0.5	89365	0.86699	0.75195	0.85127	0.63940
0.3	89357	0.86704	0.72531	0.85558	0.59678
0.1	89313	0.86593	0.67067	0.86510	0.55411
0.05	89260	0.86689	0.65445	0.87047	0.52490

Таблица 4: Точность, время работы и размерность признакового пространтсва при различных значениях `max_df` для моделей `BagOfWords` и `Tfidf`

Таким, образом, можно сказать, что целесообразно исключать слишком редкие и слишком частые слова, иначе модель может обнаруживать нелогичные и случайные зависимости. Также при определенном наборе параметров можно добиться значительного сокращения числа признаков и порой улучшения качества. Стоит помнить о несбалансированности классов в задаче, и поэтому модели может не хватать имеющейся выборки для достижения лучшей точности. В основном `Tfidf` уступает в точности `BagOfWords`, однако при `min_df`=0.01 она показывает значительно лучший результат. С ростом `min_df` падает число признаков в силу удаления редких слов, точность немного падает. Уменьшение `max_df` почти не сокращает число признаков, так как удаляет наиболее часто встречающиеся слова, на точность влияет слабо, местами есть частичная тенденция к увеличению.

Применение лучшей модели и анализ ошибок

Рассмотрим наилучшую из полученных моделей и применим ее к тестовой выборке. В качестве лучшей модели, как было принято ранее, используем стохастический градиентный спуск со следующими параметрами: $\alpha = 1.75$, $\beta = 0.025$, `tolerance`=10^{-7}, максимальное число итераций - 2000, размер батча - 1000. Для повышения качества применяем лемматизацию без удаления стоп-слов из текстов, рассматриваем модель `BagOfWords` с `max_df`=0.3, `min_df`=10^{-5}. Эти значения и методы были выбраны по результатам экспериментов и проведенному подбору параметров. Получим точность 0.86854 на тестовой выборке. Для более детальной оценки можем рассмотреть матрицу ошибок:

	$a(x_i) = 1$	$a(x_i) = -1$
$y_i = 1$	5760	483
$y_i = -1$	2247	12186

Таблица 5: Матрица ошибок для лучшей модели на тестовой выборке

Таким образом, построенная модель достаточно хорошо справляется с поставленной задачей классификации.

Рассмотрим комментарии, на которых алгоритм допустил ошибку.

- «Dear god this site is horrible.»

Данный комментарий отчасти можно считать токсичным, но недовольство в нем скорее всего направлено на качество работы сайта или иные проблемы, нежели на конкретного человека или группу лиц.

- «Please, someone fix this godawful article.»

В этом случае опять же подразумеваются проблемы со статьей, возможно ее оформлением или плохо составленным содержанием, т. е. врядли этот комментарий можно расценить как оскорбление кого-либо.

- «This page has one sentence about the basic definition of the word, and a huge amount about the slang/profane uses. Perhaps the former should be extended; is there no information about female dogs available beyond their name? This is an encyclopaedia, not a dictionary. i feel that whoever is looking this definition up is very appropiate and should be deleted from wikipedia...IMMEDIATLY. this word is used very often and is also a very mean word. i belive that is majorly true. very much so. okay so, the good meaning is a female dog. BITCH !!!!!!!!!It also stands for the name Brittany Fellows—Preceding unsigned comment added by • ==etymology== The word bitch is from the Old Norse Bikkjuna meaning female of the dog of unknown origin, Grimm derives the Old Norse words from Lapp Pittja, But OED notes that the converse is equally possible. The adj. Bitchy was first seen in 1925. The verb meaning to complain in 1930. Slang Bitc»

В данном случае дается информации о этимологии конкретного бранного слова, и из-за его присутствия алгоритм выделил комментарий как токсичный.

- «and lewd sex in China»

В данном случае фраза выглядит обрывочной, и понять полный контекст представляется сложным, но все же врядли она является частью токсичного высказывания.

- «black mamba == It.is ponious snake of the word and but it not kills many people but king cobra kills many people in India»

Данный текст является скорее просто констатацией факта, и модель расценила его как токсичный комментарий из-за наличия слова «kills»

Использование n-грамм

Добавим к текстам n-граммы и рассмотрим, как при этом изменится поведение модели. n-граммы — непрерывные последовательности из n элементов (слов) в тексте, как правило в предложении. Их использование позволяет приблизить вероятность некоторго слова с учетом предшествующих ему в последовательности слов. Добавим n-граммы к текстам и применим лучшую модель, рассмотрим результаты.

размер максимальных n-грамм	**число признаков**	**точность**	**время работы, с**
1	89357	0.85558	0.51211
2	1012240	0.83570	1.57163
3	2996706	0.81611	3.35401
4	5461337	0.80557	5.37456
5	8026525	0.79933	6.84682
6	10581455	0.79609	8.26328
7	13102075	0.79294	10.42232
8	15583598	0.79198	11.32184
9	18025483	0.79120	12.85903

Таблица 6: Зависимость числа признаков, точности и времени работы алгоритма в зависимости от размера максимальных добавленных n-грамм

Таким образом, добавление n-грамм привело к снижению точности, причем с ростом размера максимальных n-грамм точность падает сильнее. Также заметно возрастает размерность признакового пространства и время работы алгоритма. Полученные результаты могут быть причиной того, что выражения и конструкции, определяющие токсичный комментарий (оскорбления, нецензурная лексика и т. д.) в основном состоят из одного слова и могут относительно свободно располагаться в предложении, поэтому рассмотрение последовательности слов не оказалось полезным в данном случае.

Выводы

В работе было проведено исследование логистической регрессии, методов классического и стохастического градиентного спуска, оценено качество и скорость их работы в зависимости от гиперпараметров (темп обучения, размер подвыборки в стохастической реализации, начальное приближение вектора весов и т. д.). Было проведено сравнение стандартного и стохастического методов градиентного спуска, в результате которого стохастический оказался предпочтительнее с точки зрения точности классификации, значений функции потерь, скорости сходимости и затрат памяти. Были также рассмотрены основы работы с текстовыми данными, различные методы, приемы и модели текстовых представлений. Были рассмотрены и сравнены модели `Tfidf`, `BagOfWords`, были рассмотрены идеи лемматизации, удаления стоп-слов, их влияеие на качество классификации. В целом применение данных подходов положительно влияет на точность алгоритмов. Также были рассмотрены некотрые теоретические аспекты, связанные с логистической регрессией.

Список литературы

[1] К. В. Воронцов. Машинное обучение, курс лекций. `http://www.machinelearning.ru/wiki/index.php?title=%D0%9C%D0%B0%D1%88%D0%B8%D0%BD%D0%BD%D0%BE%D0%B5_%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)`