Линейные классификаторы и градиентные методы обучения

Сорокин Олег, 317

каф. ММП, ВМК МГУ
14.11.2022

Содержание

1 Введение

В качестве первого приближения решения задач бинарной классификации часто рассматриваются линейные классификаторы. Этот класс методов основан на предположении о выполнении в данном датасете гипотезы линейной сепарабельности. Линейные классификаторы часто рекомендуются к применению в случае разреженности входной матрицы объектов-признаков. Далее рассмотрим одну из таких задач, а именно - определение токсичности комментария на основании матрицы встречаемости слов.

Рассматривая датасет соответствующий этой задаче:

- Реализуем на языке Python с помощью библиотек numpy и scipy линейный классификатор, а также градиентный спуск (в т.ч. стохастический) с подсчётом необходимых величин (градиента, функционалов в точке и т.д.).
- Исследуем сходимость градиентного спуска и влияние этого процесса на качество модели в зависимости от гиперпараметров обучения.
- Проанализируем классические подходы к предобработке для увеличения качества текстовых моделей.
- Реализуем некоторые дополнительные методы, потенциально способные повысить качество модели.

2 Пояснения к задаче

При проведении экспериментов, в том числе при реализации метода градиентного спуска, будем пользоваться теоретическими выводами, описанными в этом разделе.

Для начала дадим краткое описание линейной модели бинарной классификации, которую будем использовать. Пусть дана обучающая выборка $X = (x_i, y_i)_{i=1}^l$. Предположив, что множество объектов каждого из классов $y \in \{+1, -1\}$ отделено от другого некоторой гиперплоскостью, определим модель классификации как

$$a_w(x) = sign(\langle w, x\rangle + w_0) = sign(\langle \hat{w}, \hat{x}\rangle),$$

где $\hat{w}$ - вектор параметров размерности на 1 большей, чем w, $\hat{x}$ - признаковое описание объектов с введённым константным ненулевым признаком.

Критерием для обучения модели считаем минимизацию эмпирического риска, добиваться этого будем с помощью градиентного спуска. Получим явное выражение для градиента, чтобы ускорить процесс обучения и не прибегать к численным методам. Функционал качества с учётом $L2$-регуляризации с параметром силы регуляризации $\frac{\lambda}{2}$ запишем в виде

$$Q(X, w) = \frac{1}{l}\sum_{i=1}^{l}\mathcal{L}(M_i(w)) + \frac{\lambda}{2}||w||_2^2 \tag{1}$$

где

$$M_i(w) = y_i * \langle w, x_i\rangle,$$
$$\mathcal{L}(M) = \log\left(1 + \exp\left(-M\right)\right)$$

Найдём дифференциал этого функционала:

$$d_w Q(X, w)[dw] = \frac{1}{l}\sum_{i=1}^{l} d\mathcal{L}(M_i) + \frac{\lambda}{2} d(||w||_2^2) =$$

$$= \frac{1}{l}\sum_{i=1}^{l}\frac{d(1 + \exp\left(-M_i\right))}{1 + \exp\left(-M_i\right)} + \frac{\lambda}{2} * 2||w||_2^2 * d(||w||_2) =$$

$$= \frac{1}{l}\sum_{i=1}^{l}\frac{-\exp\left(-M_i\right) * d(M_i)}{1 + \exp\left(-M_i\right)} + \lambda||w||_2 * d(\langle w, w\rangle^{1/2}) =$$

$$= \frac{1}{l}\sum_{i=1}^{l}\frac{-\exp\left(-M_i\right) * y_i\langle x_i, dw\rangle}{1 + \exp\left(-M_i\right)} + \lambda||w||_2 * \frac{\langle w, dw\rangle}{||w||_2} =$$

$$= \langle \frac{1}{l} \sum_{i=1}^{l} (\frac{\exp(-M_i)}{1+\exp(-M_i)} * (-y_i)) * x_i, dw \rangle + \langle \lambda w, dw \rangle =$$

$$= \langle \lambda w - \frac{1}{l} \sum_{i=1}^{l} (\frac{1}{1+\exp(M_i)} * y_i) * x_i, dw \rangle$$

То есть

$$\nabla_w Q = \lambda w - \frac{1}{l} \sum_{i=1}^{l} (\frac{1}{1+\exp(M_i)} * y_i) * x_i \tag{2}$$

Теперь обобщим задачу на несколько классов. Пусть в той же постановке даны K классов. Тогда вероятность каждого выразим как

$$P(y=j|x) = \frac{\exp(\langle w_j, x \rangle)}{\sum_{k=1}^{K} \exp(\langle w_k, x \rangle)}$$

Тем самым, задача сводится к методу максимального правдоподобия, то есть минимизации по всем весам для выражения

$$Q(X,w) = -\frac{1}{l} \sum_{i=1}^{l} logP(y_i|x_i) + \frac{\lambda}{2} \sum_{k=1}^{K} ||w_k||_2^2$$

Для простоты в следующих выкладках положим $\lambda = 0$, то есть уберём регуляризацию. Рассмотрим один объект $x \in X$ класса $k \in K$ и по аналогии с бинарным случаем вычислим дифференциал, считая все веса, кроме некоторого w_n фиксированными:

$$d_{w_n} Q(x,w) = \frac{\sum_{i=1}^{K} \exp(\langle w_i, x \rangle)}{\exp(\langle w_k, x \rangle)} * \frac{\exp(\langle w_n, x \rangle + \langle w_k, x \rangle) * d(\langle w_n, x \rangle) - [\sum_{i=1}^{K} \exp(\langle w_i, x \rangle)] * d\exp(\langle w_k, x \rangle)}{(\sum_{i=1}^{K} \exp(\langle w_i, x \rangle))^2} =$$

$$= \frac{\exp(\langle w_n, x \rangle) * d(\langle w_n, x \rangle) - [\sum_{i=1}^{K} \exp(\langle w_i, x \rangle)] * d(\langle w_k, x \rangle)}{\sum_{i=1}^{K} \exp(\langle w_i, x \rangle)} =$$

$$= \langle \frac{\exp(\langle w_n, x \rangle) * x - [\sum_{i=1}^{K} \exp(\langle w_i, x \rangle)] * [n=k] * x}{\sum_{i=1}^{K} \exp(\langle w_i, x \rangle)}, dw_n \rangle =$$

$$= \langle (\frac{\exp(\langle w_n, x \rangle)}{\sum_{i=1}^{K} \exp(\langle w_i, x \rangle)} - [n=k]) * x, dw_n \rangle =$$

$$\nabla_{w_n} Q(x,w) = (\frac{\exp(\langle w_n, x \rangle)}{\sum_{i=1}^{K} \exp(\langle w_i, x \rangle)} - [n=k]) * x \tag{3}$$

Суммируя по всем объектам выборки и учитывая слагаемое регуляризации, полученное в бинарном случае, выпишем искомое выражение

$$\nabla_{w_n} Q(X,w) = \lambda w_n - \frac{1}{l} \sum_{i=1}^{l} (\frac{\exp(\langle w_n, x \rangle)}{\sum_{i=1}^{K} \exp(\langle w_i, x \rangle)} - [n=y_i]) * x_i \tag{4}$$

Теперь покажем, что (2) вытекает из (4) при рассмотрении двух классов $\{-1, +1\}$. Для этого без ограничения общности в (3) положим $n = +1$, рассматривая один объект класса y:

$$(\frac{\exp(\langle w_y, x \rangle)}{\exp(\langle w_{+1}, x \rangle) + \exp(\langle w_{-1}, x \rangle)} - [y=+1]) * x$$

Рассмотрим случай $y = +1$, остальные случаи рассматриваются аналогично:

$$(\frac{\exp(\langle w_{+1}, x \rangle)}{\exp(\langle w_{+1}, x \rangle) + \exp(\langle w_{-1}, x \rangle)} - 1) * x =$$

$$= -\frac{\exp(\langle w_{-1}, x \rangle)}{\exp(\langle w_{+1}, x \rangle) + \exp(\langle w_{-1}, x \rangle)} * x =$$

$$= -\frac{1}{1+\exp(\langle w_{+1} - w_{-1}, x \rangle)} * x$$

Обозначим разность $(w_{+1} - w{-}1)$ как новый вектор весов w. Рассмотрев все случаи и просуммировав по выборке, приходим к обобщённой записи (1).

3 Эксперименты

3.1 Описание экспериментальных данных и их предобработка

Данные взяты с соревнования Kaggle "Toxic Comment Classification Challenge"и представляют собой размеченные по признаку токсичности комментарии из английской Википедии.

Перед проведением экспериментов во всех текстах:

- все символы приведены к нижнему регистру, убраны ведущие и конечные пробелы;
- символы, не являющиеся буквами и цифрами, заменены на пробелы.

После этого для датасета была построена разреженная матрица объектов-признаков (Bag of Words), причём для первых экспериментов были исключены языковые единицы, встречающиеся менее чем в 25 документах (комментариях).

3.2 Анализ и сравнение методов градиентного спуска

3.2.1 Программная реализации методов

На основании формул (1) и выведенной формулы для градиента бинарного линейного классификатора (2) были реализованы векторизованные методы подсчёта функции и её градиента. Проверка работы функций сверялась с численным подсчётом градиента произвольного функционала (правая производная с шагом $1e^{-8}$).

Также реализован градиентный спуск, удовлетворяющий заданному набору тестов и стохастический (minibatch) градиентный спуск. Для тестирования последнего была взята выборка из 100.000 точек на плоскости, разделимая по классам прямой $y = 0$. При этом точность на валидационной выборке (20% от обучающей) на каждом из 5 измерений достигала 0.95 и выше.

3.2.2 Анализ влияния гиперпараметров на градиентный спуск

Итерационный процесс сдвига в направлении антиградиента $Q(X, w)$ (см. (1)) осуществляется по правилу

$$w^{(k+1)} = w^k - \eta_k * \nabla_w Q(X, w),$$

где

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

Начальное приближение в экспериментах, не связанных с его влиянием на результат, определяется случайным равномерно распределённым на отрезке $[0, 1]$ вектором. Чтобы это приближение не влияло на исход при пересоздании объекта классификатора, каждый переинициализируем зерно генерации случайных чисел.

Зафиксируем $\beta = 0$ для рассмотрения влияния параметра шага α (тем самым отключив "угасание"градиента с каждой последующей итерацией). На рис. 1 показаны результаты для величины функции ошибки. Видно, что при $\alpha <= 0.01$ сходимость очень медленная, функция практически не изменяется. При $\alpha \in \{0.1, 1.0\}$ сходимость значительно быстрее. Шаг $\alpha = 10.0$ оказался слишком большим для попадания в точку минимума при заданной инициализации, что привело к расхождению метода. Эти замечания справедливы и для точности на валидационной выборке (рис. 2). При $\alpha = 1.0$ на этом графике можем заметить колебания, что говорит о нестабильности процесса обучения.

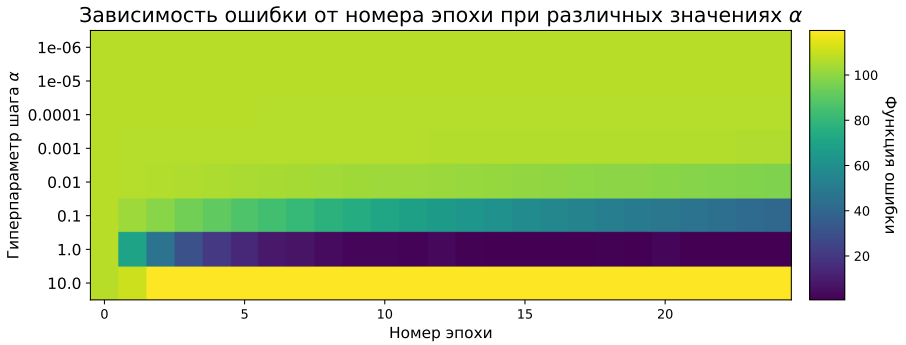

Рис. 1: Градиентный спуск: влияние α на ошибку в зависимости от итерации

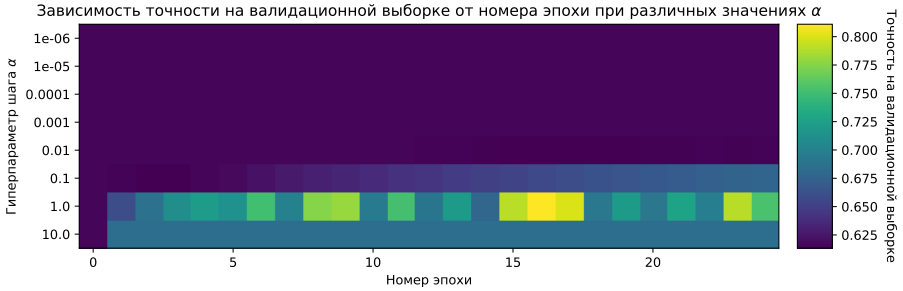

Рис. 2: Градиентный спуск: влияние α на точность на валидационной выборке в зависимости от итерации

Для решения проблемы нестабильности обучения проанализируем влияние параметра "угасания"градиента β, что потенциально способствует остановке алгоритма в точке локального минимума и предотвращению выхода из соответствующей зоны выпуклости графика. Зафиксируем $\alpha = 1.0$, при котором наблюдались осцилляции. На рис. 3 и 4 видим общий характер зависимостей ошибки и точности на валидации соответственно:

- при $\beta < 0.022$ наблюдаем колебания по точности, сравнимые с $\beta = 0$, то есть влияние параметра незначительно;
- при больших β (на графике такое поведение отражено последней строкой) градиент по норме близок к нулю, после некоторой итерации эффекта от обучения практически нет;
- остальные β сглаживают процесс обучения. В аппендиксе приведено более подробное рассмотрение этого диапазона с целью поиска оптимального значения для последующих экспериментов.

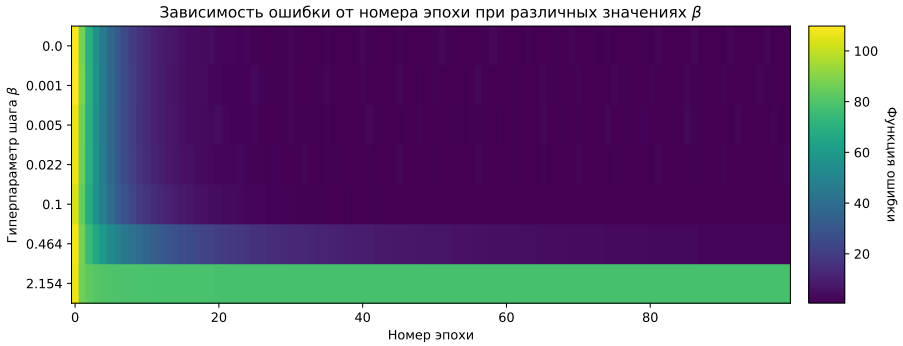

Рис. 3: Градиентный спуск: влияние β на ошибку в зависимости от итерации

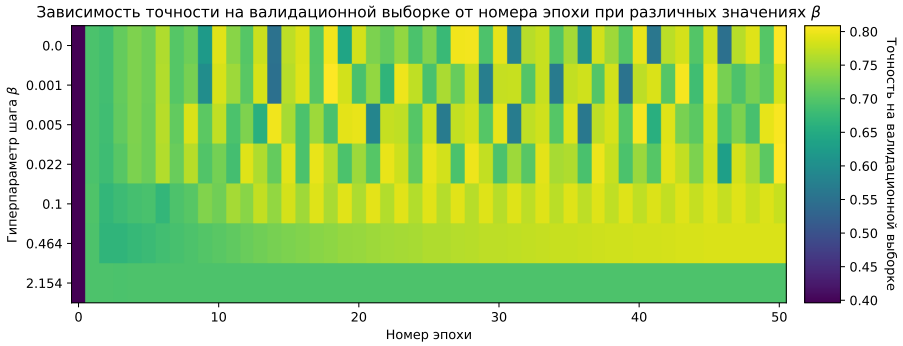

Рис. 4: Градиентный спуск: влияние β на точность на валидационной выборке в зависимости от итерации

Наконец, при фиксированных оптимальных параметрах $\alpha = 1.0, \beta = 0.3$ (см. аппендикс, рис. 11, 12) рассмотрим, как следующие стратегии инициализации влияют на исход:

- нулевые веса;
- единичные веса;
- равномерные веса из отрезка $[0, 1]$;
- равномерные веса из отрезка $[-5, 5]$;
- нормально распределённые веса.

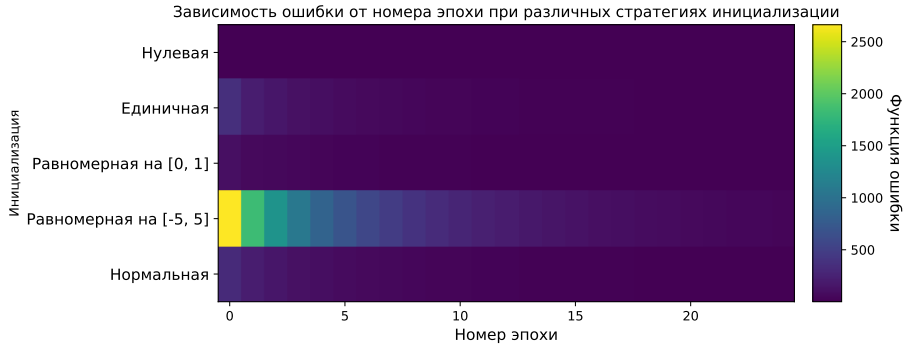

Рис. 5: Градиентный спуск: влияние инициализации на ошибку в зависимости от итерации

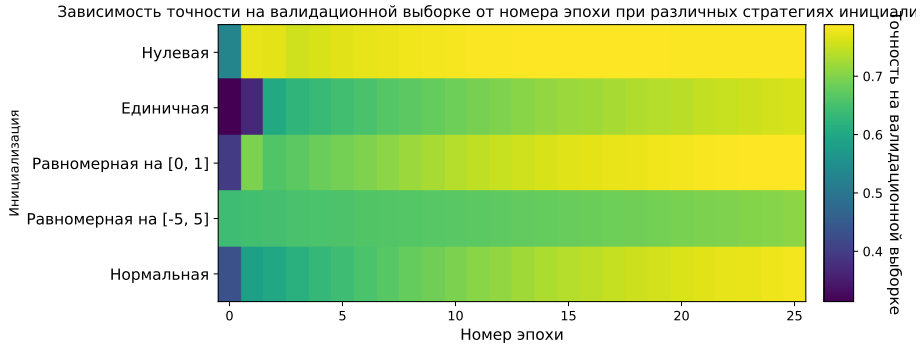

Рис. 6: Градиентный спуск: влияние инициализации на точность на валидационной выборке в зависимости от итерации

На рис. 5, 6 отражена динамика ошибки и точности на валидации соответственно.

При нулевой инициализации после первой же итерации значения практически перестают меняться. Это связано с тем, что уже изначально многие бесполезные признаки были обнулены, а после первой итерации незначительно изменились наиболее полезные (рис. 7).

Для единичной и равномерной на [-5, 5] инициализаций характерен как наибольший пик на первой итерации, так и в целом наибольшие среди остальных изменения весов (рис. 7). Это может означать, что соответствующие точки находятся дальше всего от искомой точки минимума.

Оставшиеся две инициализации асимптотически совпадают с нулевой (как по ошибке, так и по точности на валидации).

В дальнейших экспериментах будем использовать стандартную нормальную инициализацию.

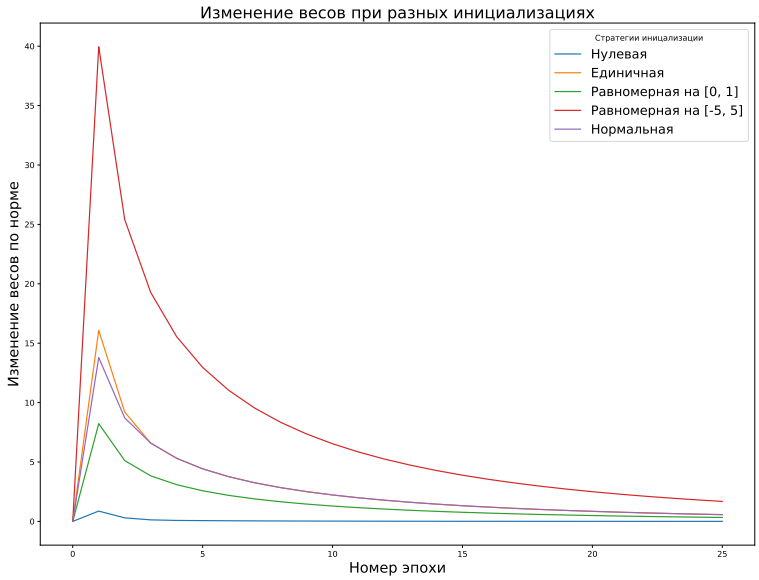

Рис. 7: Градиентный спуск: влияние инициализации на норму изменения весов на каждой итерации

3.2.3 Анализ влияния гиперпараметров на стохастический градиентный спуск, сравнение с градиентным спуском

Зафиксировав те же значения $\alpha = 1.0, \beta = 0.3$, рассмотрим поведение алгоритма при изменении размера батча (включая обычный градиентный спуск).

Отметим, что некорректно сравнивать итерации при разных размерах батчей. Можно сравнивать целые эпохи, но чтобы собрать репрезентативную выборку о большом количестве проходов через весь датасет потребуется много времени. Вместо этого будем сравнивать алгоритмы по времени.

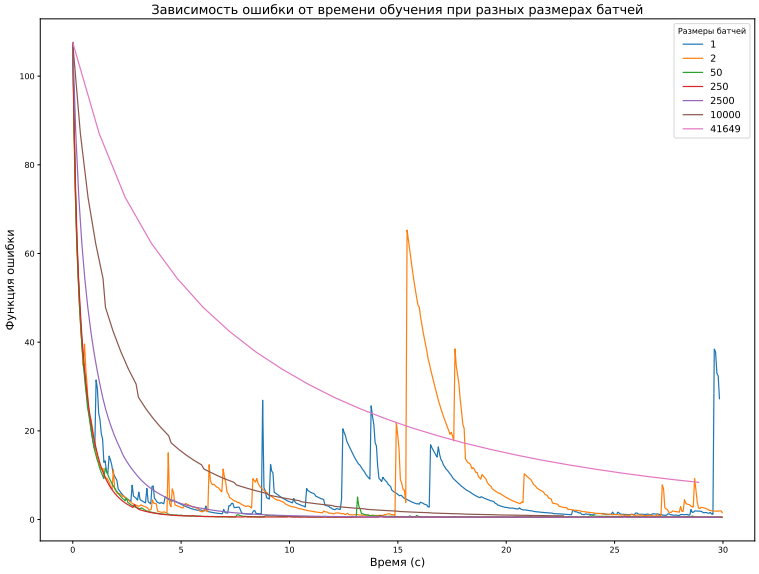

Рис. 8: Стохастический градиентный спуск: зависимость ошибки от времени в зависимости от размера батча

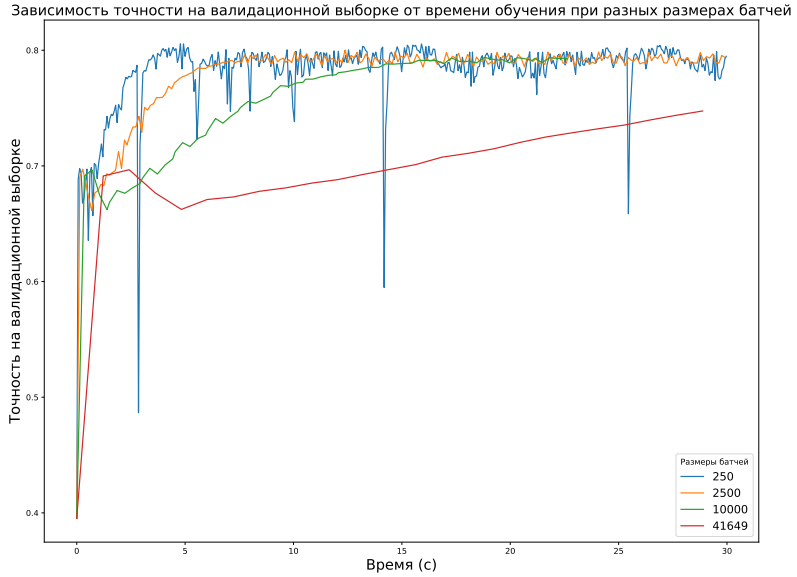

Рис. 9: Стохастический градиентный спуск: влияние размера батча на точность на валидационной выборке в зависимости от итерации

На рис. 8 показано изменение ошибки по ходу обучения. Видно, что при градиентном спуске кривая обучения гладкая, но обучение при этом происходит очень медленно. С уменьшением размера батча графики становятся более ступенчатыми. Это связано с тем, что при итерации по батчам мы строим оценку градиента по всей выборке, которая может сильно отличаться (см. пики на рис. 8) от правильного направления к оптимуму функции.

Рис. 9 отражает тот же смысл: оценка градиента может быть очень приблизительна, но скорость сходимости (имеется в виду сходимость в среднем) значительно возрастает. Некоторые графики могут быть очень шумными. Для данного рисунка соответствующие размеры батчей не учитывались.

В ходе перезапуска аналогичных экспериментов с параметрами α, β для разных размеров батчей видны те же тенденции по эпохам, поэтому оптимальные параметры остаются теми же. Для примера ниже приведён график ошибки для батча размера 2500 (рис. 10). В сравнении с рис. 1 вновь обнаруживаем, что при слишком малых значениях параметра обучение происходит слишком медленно, при больших значениях может возникнуть расходимость как на рис. 1 или осцилляция вокруг точки минимума, как на рис. 10.

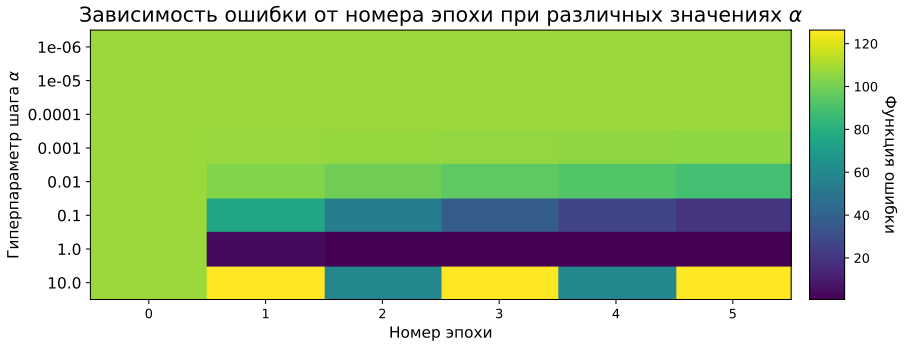

Рис. 10: Стохастический градиентный спуск: влияние параметра α на ошибку в зависимости от эпохи

Таким образом, градиентный спуск и стохастический градиентный спуск имеют схожий смысл гиперпараметров (в том смысле, что динамика в обучении при их изменении сохраняет-

ся). Стохастический градиентный спуск по сравнению с обычной версией даёт значительный прирост по скорости работы алгоритма: оценивая градиент небольшим числом объектов можно быстро прийти в минимум. Эта оценка состоятельная, но при малой выборке может быть очень неточна и привести к движению в неправильном направлении.

3.3 Увеличение точности классификации, поиск наилучшего в этом смысле алгоритма

Зафиксируем те же параметры α, β, w_0, используем батчи по 2500 объектов. Обучим алгоритм на 100 итерациях и посчитаем точность на тестовой выборке.

Далее применим к обучающей выборке алгоритм лемматизации, то есть приведём словоформы к начальной форме. Такое преобразование позволит уменьшить размерность признакового пространства. Наконец, удалим из предложений стоп-слова. После применения каждого из этих алгоритмов пересчитаем точность.

В таблицах ниже приведены результаты по применению алгоритмов и влиянию этого на точность на тестовой выборке, а также на время обучения алгоритма.

CountVectorizer:

	Точность на тесте	Размерность	Время(с)
Без изменений	0.738	6239	17.18
Лемматизация	0.754	5167	12.98
Лемматизация + удаление стоп-слов	0.759	5042	12.31

TfidfVectorizer (?!):

	Точность на тесте	Размерность	Время(с)
Без изменений	0.698	6239	16.61
Лемматизация	0.757	5167	13.07
Лемматизация + удаление стоп-слов	0.742	5042	11.89

Так как наилучшую точность показал CountVectorizer (?!) с лемматизацией и удалёнными стоп-словами, далее работаем с этим представлением.

Рассмотрим влияние параметра min_df из конструктора CountVectorizer на точность на тестовой выборке, размерность признакового пространства и время работы:

min_df	Точность на тесте	Размерность	Время(с)
3	0.737	21812	37.16
10	0.735	9128	12.98
25	0.757	5042	12.31
50	0.736	3170	8.68

То есть изначально выбранный параметр min_df = 25 близок к оптимальному.

Из всех параметров модели остался неисследованным только коэффициент регуляризации. Изменение точности на тестовой выборке отражено в таблице:

L2 коэффициент	Точность на тесте
0.001	0.718
0.01	0.729
0.1	0.759
1.0	0.743
10.0	0.343

То есть используемый ранее коэффициент регуляризации близок к оптимальному значению.

Таким образом, лучший алгоритм в смысле точности на тестовой выборке описывается набором параметров:

- $\alpha = 1.0, \beta = 0.3$
- $\lambda = 0.1$
- batch_size = 2500

Алгоритм чаще всего допускает ошибки в текстах, где токсичность проявляется в связи слов между собой. Например, если слова "europe "hell "genocide"по отдельности могут употребляться в повествовательном, нейтральном контексте, то вместе они могут давать токсичный политико-направленный эффект. Модель уловить эту связь не может (это сделать по набору слов невозможно и для человека) и часто помечает такие тексты как нетоксичные.

4 Дополнительные методы увеличения точности

4.1 n_grams для CountVectorizer

Поскольку CountVectorizer показал лучший результат (?!), n_grams будем использовать применительно к нему. В таблице отражено влияние добавлений n_gram в признаковое пространство. Начиная с $n = 4$ наблюдается ухудшение качества, то есть становится слишком много неинформативных признаков.

Макс. gram	Качество	Время(с)
1	0.757	12.31
2	0.763	31.34
3	0.765	40.64
4	0.761	48.54

4.2 PCA

-

5 Выводы

Модель линейного классификатора может быть полезна в задачах анализа текстов. Основная трудность в таких задачах - подбор парамеров предобработки текста для упрощения работы классификатора и гиперпараметров самой модели. Наиболее важными из таких гиперпараметров являются влияющие на шаг и "угасание"градиента параметры, а также параметр регуляризации. В зависимости от их значения алгоритм может как сойтись за несколько итераций, так и бесконечно осциллировать около точки минимума.

Градиентный спуск позволяет подобрать веса для описанного классификатора с помощью сложновычислимого итерационного процесса. Стохастический градиентный спуск упрощает этот процесс: не нужно хранить огромные матрицы при итерации по батчам. Недостатком стохастического подхода является неточность оценки градиента.

Модели такого типа плохо работают при больших размерностях признакового пространства. Существуют специальные методы уменьшения размерности матричных текстовых представлений (лемматизация, удаление стоп-слов), но могут оказаться полезными и общие методы (такие как PCA из utils).

В случае анализа связанных текстов есть смысл добавить некоторые информативные признаки, иллюстрирующие эту связь. Примерами таких признаков являются n_grams.

6 Аппендикс

6.1 Графики для более точного подбора параметра β

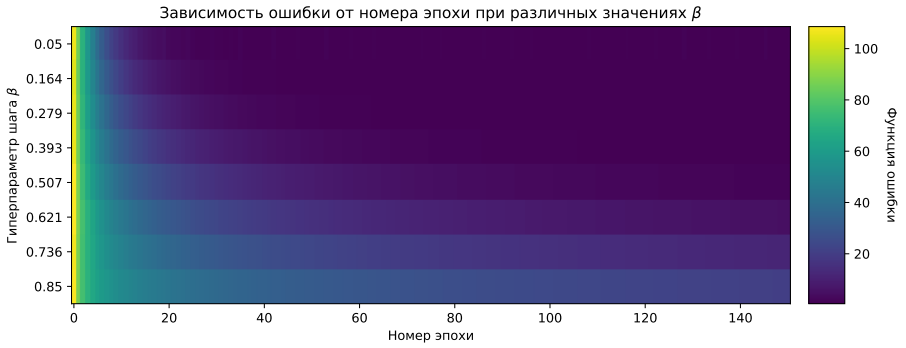

Рис. 11: Градиентный спуск: влияние β на ошибку в зависимости от итерации

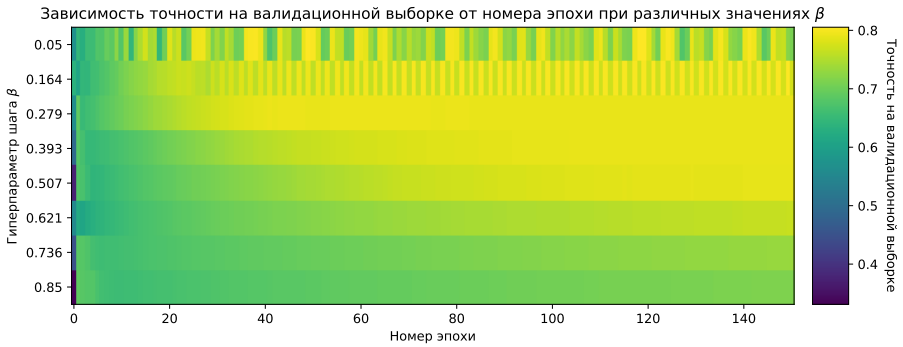

Рис. 12: Градиентный спуск: влияние β на точность на валидационной выборке в зависимости от итерации