Градиентные методы обучения линейных моделей. Отчет.

Пацация Александр, 317 группа

9 ноября 2021 г.

1 Введение

В данном отчете описываются эксперименты и их результаты применительно к алгоритму логистической регрессии. Эксперименты проводятся над набором данных, представляющим из себя комментарии из англоязычной Википедии. По этим данным требуется научится определять является ли комментарий токсичным или нет.

2 Теоретическая часть

В текущем разделе выводится формула градиента функционала потерь на всей выборке для задач бинарной и многоклассовой логистической регрессии.

2.1 Градиент функционала потерь для задачи бинарной логистической регрессии

$X = (x_i, y_i)_{i=1}^l$ – обучающая выборка, где $x_i \in \mathbb{R}^d$, $y_i \in \{-1, 1\}$.
$\mathcal{L}(z) = \log(1 + e^{-z})$ – логарифмическая функция потерь.
$a(x) = sign(\langle w, x\rangle)$ – линейный классификатор, где $w \in \mathbb{R}^d$ – вектор коэффициентов (считается, что признак из единиц уже добавлен в выборку).
$M_i(w) = y_i\langle w, x_i\rangle$ - отступ объекта x_i относительно класса y_i.
$Q(X, w) = \frac{1}{l}\sum\limits_{i=1}^{l}\mathcal{L}(M_i(w)) + \frac{\lambda}{2}\|w\|_2^2$ - функционал потерь на всей выборке X.
Градиент для функции потерь $\mathcal{L}(M(w))$:

$$d_w\mathcal{L} = d(\log(1 + e^{-M})) = \frac{-e^{-M}}{(1 + e^{-M})}dM = \frac{-e^{-y\langle w,x\rangle}}{(1 + e^{-y\langle w,x\rangle})}d(y\langle w, x\rangle) =$$
$$= \frac{-e^{-y\langle w,x\rangle}}{(1 + e^{-y\langle w,x\rangle})}y\langle dw, x\rangle = \frac{-e^{-y\langle w,x\rangle}}{(1 + e^{-y\langle w,x\rangle})}y\langle x, dw\rangle = \frac{-1}{(1 + e^{y\langle w,x\rangle})}y\langle x, dw\rangle$$

$\Rightarrow \nabla_w\mathcal{L}(M(w)) = \sigma(-y\langle w, x\rangle)yx$ – градиент функции потерь $\mathcal{L}(M(w)))$, где $\sigma(z) = \frac{1}{1+e^{-z}}$ – сигма функция.
Градиент для функционала потерь на всей выборке $Q(X, w)$:

$$\nabla_w Q(X, w) = -\frac{1}{l}\sum_{i=1}^{l}\nabla_w\mathcal{L}(M_i(w)) + \frac{\lambda}{2}\nabla_w(\|w\|_2^2) = -\frac{1}{l}\sum_{i=1}^{l}\frac{1}{(1 + e^{y_i\langle w,x_i\rangle})}y_ix_i + \lambda w$$

$\Rightarrow \nabla_w Q(X, w) = -\frac{1}{l}\sum\limits_{i=1}^{l}\sigma(-y_i\langle w, x_i\rangle)y_ix_i + \lambda w$ – градиент функционала потерь на всей выборке $Q(X, w)$.

2.2 Градиент функционала потерь для задачи многоклассовой логисчтической регрессии

$X = (x_i, y_i)_{i=1}^l$ – обучающая выборка, где $x_i \in \mathbb{R}^d$, $y_i \in \{1, \dots C\}$.
$P(y = j|x) = \frac{e^{\langle w_j,x\rangle}}{\sum\limits_{k=1}^{C}e^{\langle w_k,x\rangle}}$ – вероятность j-го класса при условии объекта x.
$a(x) = arg\max\limits_{k\in\{1,\dots C\}} sign(\langle w_k, x\rangle)$ – линейный классификатор, где $w_k \in \mathbb{R}^d$ – вектор коэффициентов k-го класса (по-прежнему считается, что признак из единиц уже добавлен в выборку).

$M_i(w) = y_i\langle w_i, x_i\rangle - \max\limits_{j\neq i} y_j\langle w_j, x_i\rangle$ - отступ объекта x_i относительно класса y_i.

$Q(X,w) = -\frac{1}{l}\sum\limits_{i=1}^{l} log(P(y_i|x_i)) + \frac{\lambda}{2}\sum\limits_{k=1}^{C}\|w_k\|_2^2$ - функционал потерь на всей выборке X.

$$d_{w_j}(log(P(y_i|x_i))) = d\left(log\frac{e^{\langle w_i,x_i\rangle}}{\sum\limits_{k=1}^{C} e^{\langle w_k,x_i\rangle}}\right) = d\left(\langle w_i,x\rangle - log(\sum_{k=1}^{C} e^{\langle w_k,x\rangle})\right) =$$

$$[y_i = j]x_i^T dw_j - \frac{e^{\langle w_j,x_i\rangle}x_i^T dw_j}{\sum\limits_{k=1}^{C} e^{\langle w_k,x\rangle}} = \langle [y_i = j]x_i - \frac{e^{\langle w_j,x_i\rangle}x_i}{\sum\limits_{k=1}^{C} e^{\langle w_k,x\rangle}}, dw_j\rangle$$

$\Rightarrow \nabla_{w_j} log(P(y_i|x_i)) = [y_i = j]x_i - \frac{e^{\langle w_j,x_i\rangle}x_i}{\sum\limits_{k=1}^{C} e^{\langle w_k,x\rangle}}$ – градиент по w_j функции потерь на объекте x_i.

$$d_{w_j}\left(\frac{\lambda}{2}\sum_{k=1}^{C}\|w_k\|_2^2\right) = d(\frac{\lambda}{2}\|w_j\|_2^2) = \lambda w_j^t dw_j$$

$\Rightarrow \nabla_{w_j}\left(\frac{\lambda}{2}\sum\limits_{k=1}^{C}\|w_k\|_2^2\right) = \lambda w_j$ – градиент члена регуляризации по w_j. $\Rightarrow$

$$\Rightarrow \nabla_{w_j} Q(X,w) = -\frac{1}{l}\sum_{i=1}^{l}\nabla_{w_j} log(P(y_i|x_i)) + \nabla_{w_j}\frac{\lambda}{2}\sum_{k=1}^{C}\|w_k\|_2^2 = -\frac{1}{l}\sum_{i=1}^{l}\left([y_i = j]x_i - \frac{e^{\langle w_j,x_i\rangle}x_i}{\sum\limits_{k=1}^{C} e^{\langle w_k,x\rangle}}\right) + \lambda w_j$$

$\Rightarrow \nabla_w Q(X,w) = (\nabla_{w_j} Q(X,w))_{j=1}^{d}$

2.3 Сведение задачи многоклассовой регрессии к бинарной при количестве классов равном двум

$X = (x_i, y_i)^l$ – обучающая выборка, где $x_i \in \mathbb{R}^d$, $y_i \in \{-1, 1\}$.
$a(x) = arg\max\limits_{k\in\{-1,1\}} sign(\langle w_k, x\rangle)$ – многоклассовый линейный классификатор при двух классах.

$$a(x) = arg\max_{k\in\{-1,1\}} sign(\langle w_k,x\rangle) = sign(\langle w_1,x\rangle - \langle w_{-1},x\rangle) = \{w = w_1 - w_{-1}\} = sign(\langle w,x\rangle)$$

$\Rightarrow$ задача многоклассовой классификации сведена к бинарной.

3 Эксперименты

3.0 Предварительные замечания

Формула шага градиентного спуска:

$$w^{i+1} = w^i - \eta_i\nabla Q(X,w^i), \eta_i = \frac{\alpha}{i^\beta}$$

Формула шага стохастического градиентного спуска:

$$w^{i+1} = w^i - \eta_i\nabla\varepsilon_i, \eta_i = \frac{\alpha}{i^\beta}$$

где $\varepsilon_i = \frac{1}{p}\sum\limits_{k=1}^{p}\mathcal{L}(M(x_k,w_i))$ – значение функционала ошибки на батче данных, p – размер батча , α, β – гиперпараметры. В приведенных ниже экспериментах в выборку не добавляется единичный признак (так как это не было указано в задании). Параметр регуляризации по умолчанию равен $\lambda = 0.01$. В качестве начального приближения w_0 по умолчанию выбирается нулевой вектор. Точность считается по метрике **accuracy**. Исходная обучающая выборка делится на обучающую и валидационную в соотношении $7:3$. Точность представленная на графике считается для валидационной выборки. Для сокращения записи, в

некоторых случаях алгоритм градиентного спуска будет называться GD (Gradient descent), а алгоритм стохастического градиентного спуска будет называться SGD (Stochastic Gradient Descent). Далее, под сходимостью будет пониматься достижение критерия остановки:

$$|Q_{k+1} - Q_k| < tolerance \tag{1}$$

где k – номер итерации для GD и эпохи для SGD. По умолчанию рассматривается $tolerance = 10^{-5}$.

3.1 Первый эксперимент

Первым делом, данные приводятся к унифицированному виду: все символы переводятся в нижний регистр и удаляются все символы отличные от букв, цифр или пробелов. Перевод в нижний регистр требуется для того, чтобы одинаковые слова не считались за разные признаки. Символы, отличные от цифр и букв, не несут никакой смысловой нагрузки и не представляют пользы.

3.2 Второй эксперимент

В данном эксперименте, датасет приводится к разреженной матрице в соответствии с моделью **Bag of words**. Значение в позиции (i, j) равняется количеству слов j в документе i. Также, на этом этапе понижается размерность признакового пространства путем отбора слов по частоте встречаемости во всех документах (параметр **min_df** в конструкторе класса **CountVectorizer**). Итоговая размерность признакового пространства – **3735**

3.3 Третий эксперимент

В данном эксперименте анализируется поведение метода градиентного спуска при различных значениях гиперпараметров.

Рис. 1: Поведение $Q(X, w)$ при различных значениях параметра α ($\beta = 0$, $w_0 = 0$)для GD.

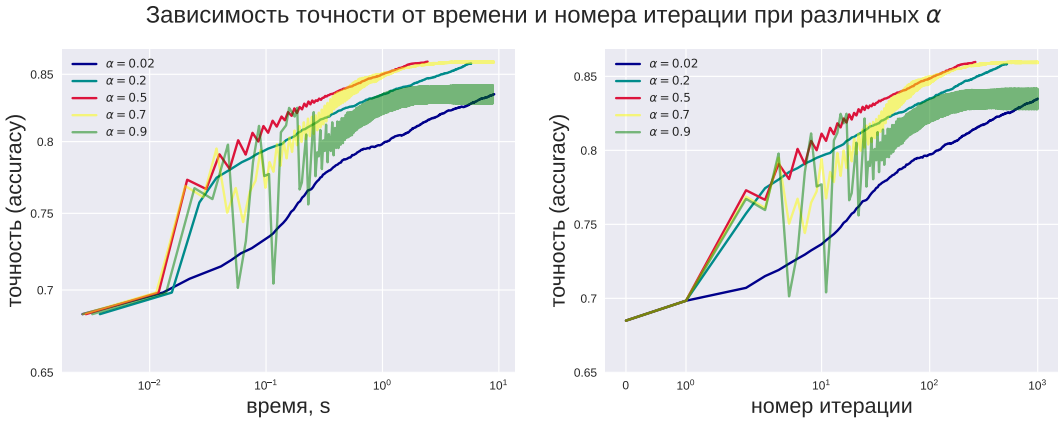

Рис. 2: Точность при различных значениях параметра α ($\beta = 0$, $w_0 = 0$) для GD.

На рис. 1, 2 представлена зависимость функционала $Q(X, w)$ и точности от значений α, при нулевом β. При увеличении значения α от 0.02 до 0.5 наблюдается увеличении скорости сходимости алгоритма (и в терминах итераций, и в терминах времени), а также небольшое улучшение точности. Начиная с $\alpha = 0.7$ алгоритм перестает сходится. Наблюдаются скачки в окрестности оптимума. Можно сделать вывод о том, что алгоритм градиентного спуска способен сходится к оптимуму при постоянном значении шага ($\beta = 0$), но только при α меньше определенного значения.

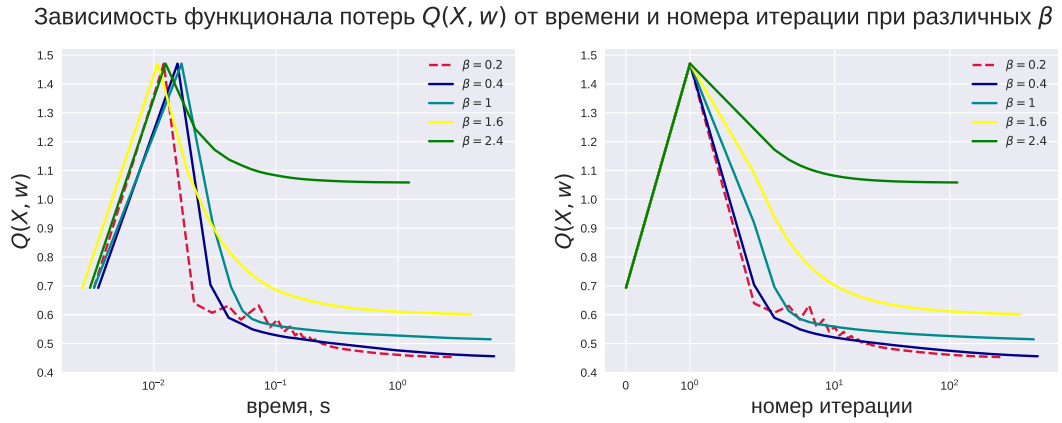

Рис. 3: Поведение $Q(X, w)$ при различных значениях параметра β ($\alpha = 1$, $w_0 = 0$) для GD.

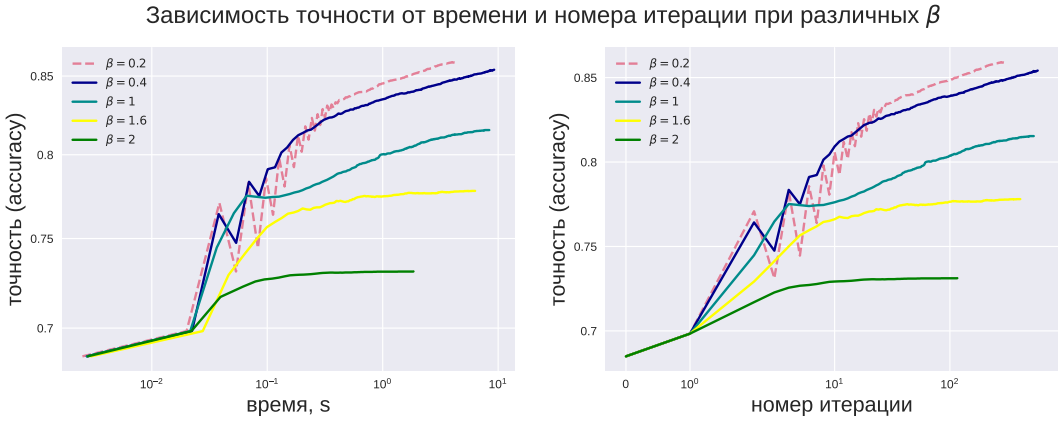

Рис. 4: Точность при различных значениях параметра β ($\alpha = 1$, $w_0 = 0$) для GD.

На рис. 3, 4 изображено поведение $Q(X, w)$ и точности при различных значениях β и фиксированном значении α. При маленьких значениях β (0.2, 0.4) наблюдаются скачки на первых итерациях. Это связано с тем, что на в начале величины k^{β} недостаточно, чтобы скомпенсировать α, поэтому происходят скачки в противоположных полуокрестностях оптимума. Далее, при увеличении значения β уменьшается время сходимости и количество итерации алгоритма, также ухудшается точность, так как при больших значениях β шаг слишком быстро стремится к нулю, и алгоритм сходится в большей окрестности минимума. Алгоритм показывает лучший результат при $\beta = 0.2$.

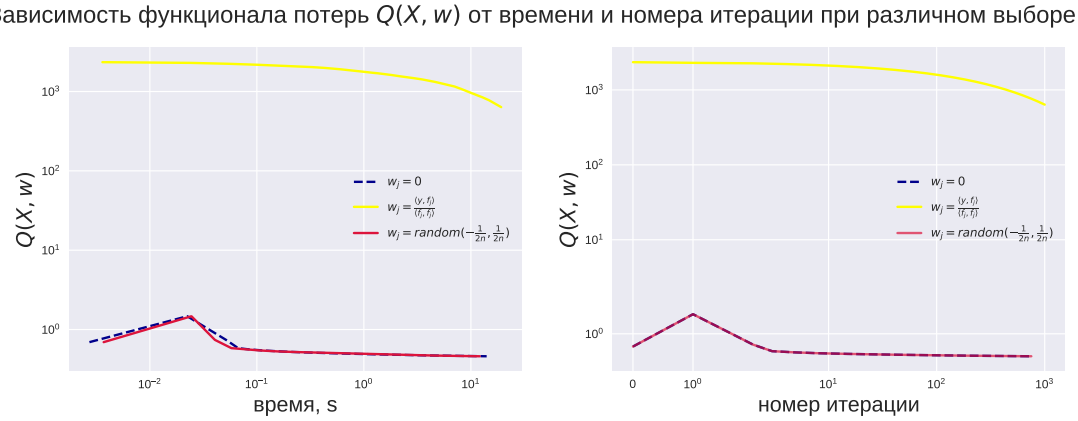

Рис. 5: Поведение $Q(X, w)$ при различном выборе начального приближения w_0 ($\alpha = 1$, $\beta = 0.5$) для GD.

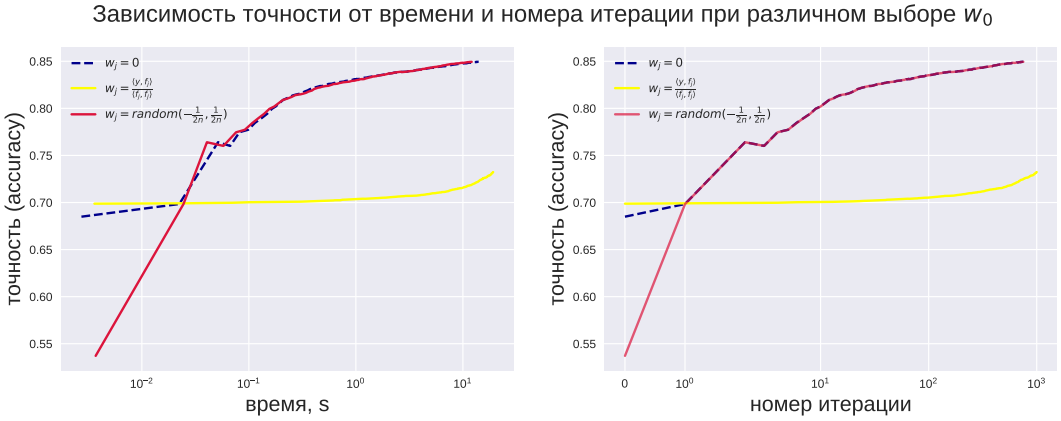

Рис. 6: Точность при различном выборе начального приближения w_0 ($\alpha = 1$, $\beta = 0.5$) для GD.

На рис. 5, 6 представлено влияние начального приближения w_0 на точность и значение $Q(X, w)$. Рассматриваются следующие начальные приближения:

- $w_0 = 0$ – нулевое значение вектора признаков (по умолчанию).
- $w_0^j = \frac{\langle y, f_j \rangle}{\langle f_j, f_j \rangle}$, где $f_j = (f_j(x_i))_{i=1}^l$ – вектор значений признака на обучающей выборке. Данная оценка является оптимальной, если признаки некоррелированы.
- $w_0^j = random(-1/2n, 1/2n)$, где n – размерность признакового пространства.

Анализируя график, можно сделать предположение о том, что признаки сильно коррелируют, так как при $w_0^j = \frac{\langle y, f_j \rangle}{\langle f_j, f_j \rangle}$ функционал $Q(X, w)$ достигает очень большого значения, и алгоритм сходится лишь в большой окрестность оптимума. При остальных значениях начального приближения значение Q(X, w) близко к минимуму, и алгоритм показывает практический идентичные результаты. Для следующих экспериментов будут использоваться следующие гиперпараметры алгоритма градиентного спуска:

- $\alpha = 0.3$.

- $\beta = 0.2$.
- $w_0 = 0$.

3.4 Четвертый эксперимент

В данном эксперименте, аналогично предыдущему, анализируется поведение метода стохастического градиентного спуска при различных значениях гиперпараметров.

Рис. 7: Поведение $Q(X, w)$ при различных α ($\beta = 0$, $w_0 = 0$) для SGD.

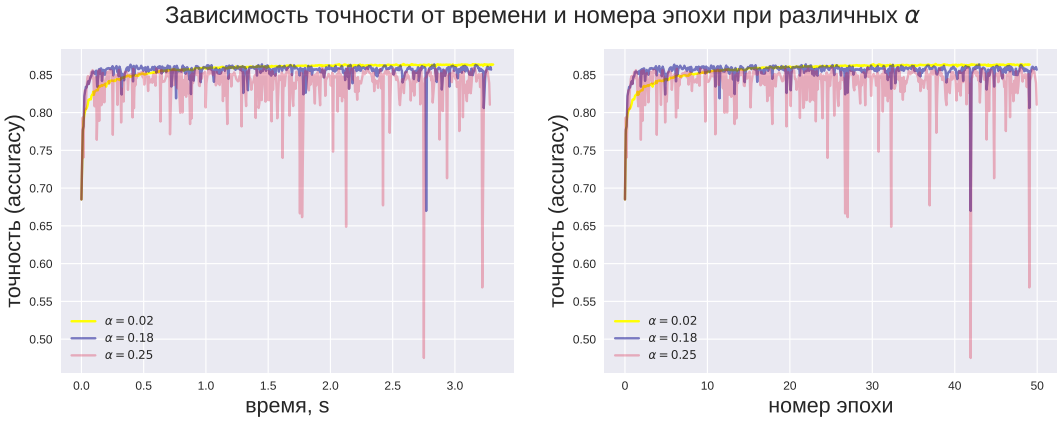

Рис. 8: Точность при различных α ($\beta = 0$, $w_0 = 0$) для SGD.

При значениях α, достаточно малых, чтобы сходился GD при $\beta = 0$, алгоритм SGD совершает скачки в достаточно большой окрестности оптимума (рис. 7, 8). Даже при $\alpha = 0.02$ SGD не способен достичь критерия остановки. Но можно отметить, что с уменьшением α амплитуда скачков уменьшается.

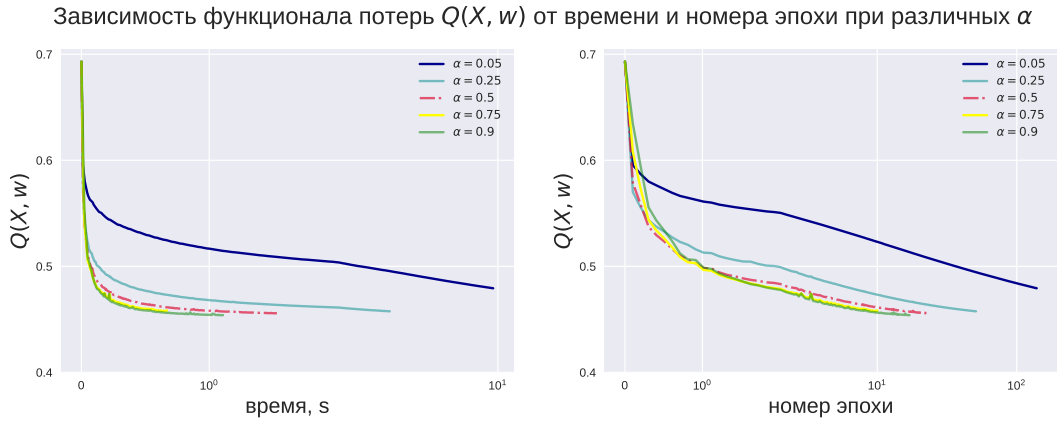

Рис. 9: Поведение $Q(X, w)$ при различных α ($\beta = 0.5$, $w_0 = 0$) для SGD.

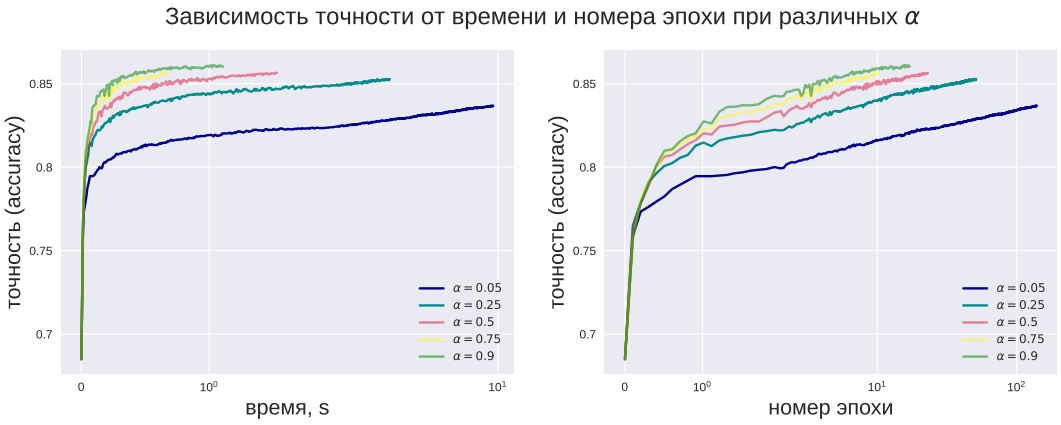

Рис. 10: Точность при различных α ($\beta = 0.5$, $w_0 = 0$) для SGD.

На рис. 9, 10 изображена зависимость точности и $Q(X, w)$ от параметра α при $\beta = 0.5$. Из графиков видно, что при увеличении α уменьшается время рабаты алгоритма и количество эпох, при этом увеличивается точность, аналогично тому как ведет себя алгоритм градиентного спуска, при небольших значениях α. Лучший результат по времени наблюдается при $\alpha = 0.75$, лучший результат по точности - при $\alpha = 0.9$,

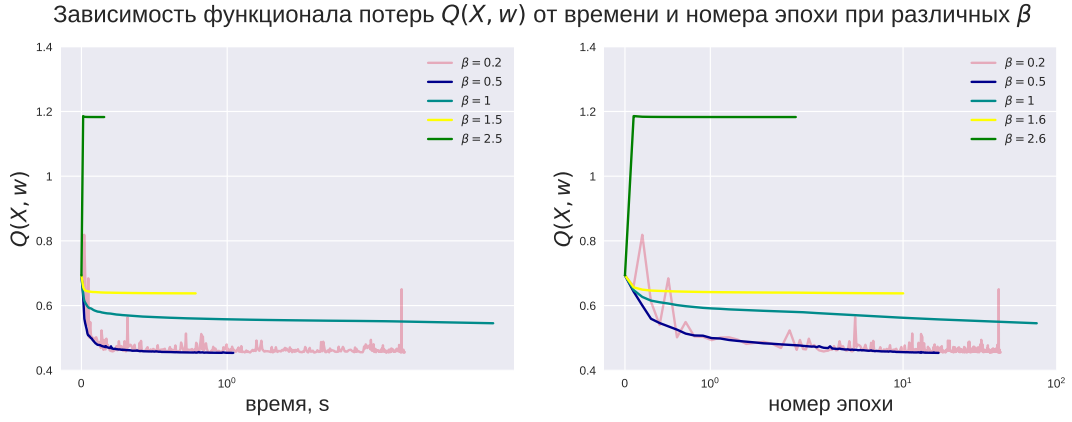

Рис. 11: Поведение $Q(X, w)$ при различных β ($\alpha = 1$, $w_0 = 0$) для SGD.

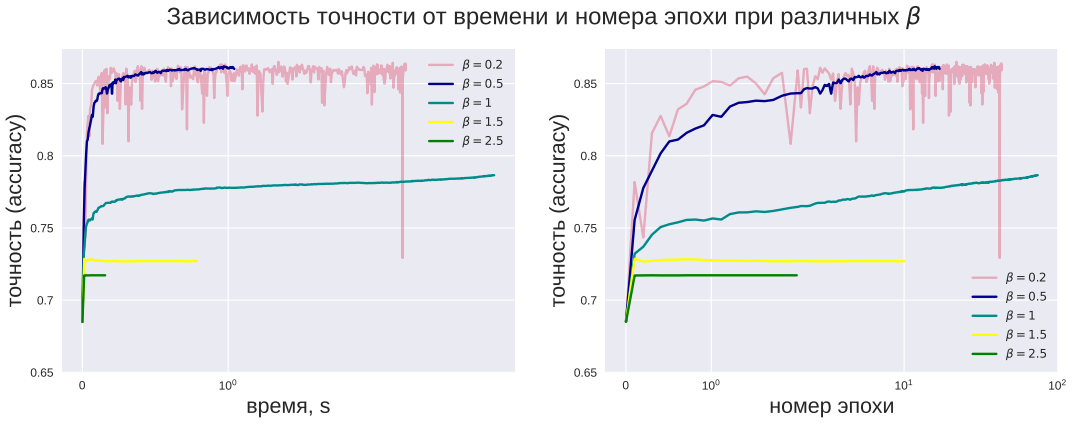

Рис. 12: Точность при различных β ($\alpha = 1$, $w_0 = 0$).

Из рис. 11, 12 на которых изображена зависимость $Q(X, w)$ и точности при различных значениях β видно, что уже при малых β может отсутствовать сходимость алгоритма. При увеличении β уменьшается скорость сходимости. Лучшее качество достигается при $\beta = 0.5$. При больших значениях качество алгоритма существенно падает, так как шаг спуска слишком быстро стремится к нулю.

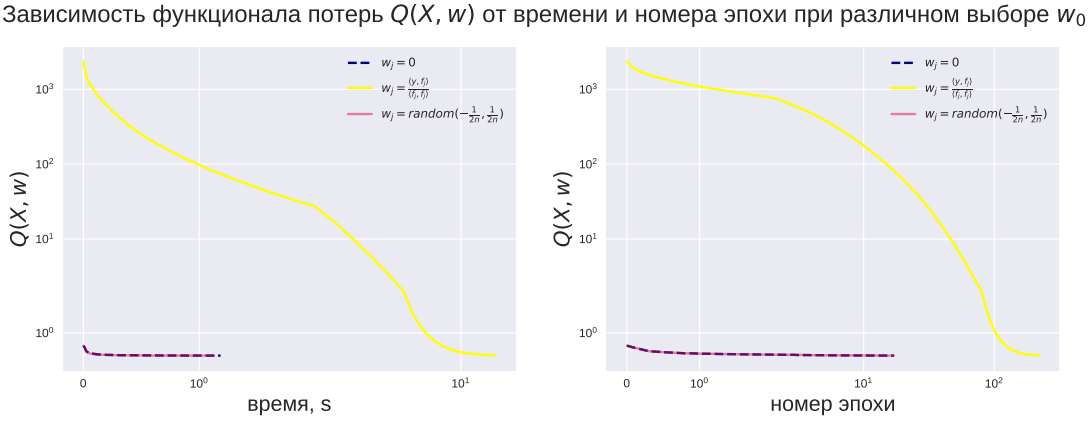

Рис. 13: Поведение $Q(X, w)$ при различном выборе начального приближения w_0 ($\alpha = 1$, $\beta = 0.5$ для SGD).

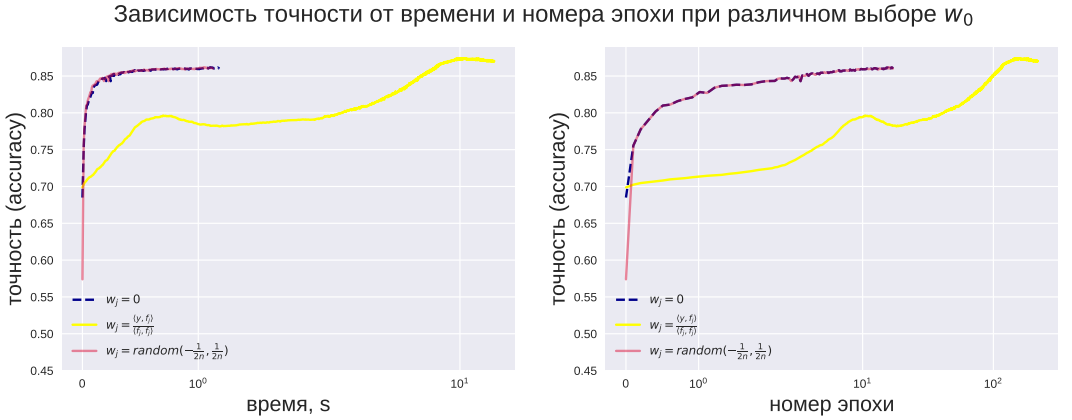

Рис. 14: Точность при различном выборе начального приближения w_0 ($\alpha = 1$, $\beta = 0.5$) для SGD.

На рис. 13, 14 рассматриваются такие же варианты начального приближения, как и в прошлом эксперименте. Результат схож, однако при $w_0^j = \frac{\langle y, f_j \rangle}{\langle f_j, f_j \rangle}$ SGD успевает сойтись к неплохому значению точности.

Рис. 15: Поведение $Q(X, w)$ при различном выборе размера батча ($\alpha = 1$, $\beta = 0.5$ для SGD).

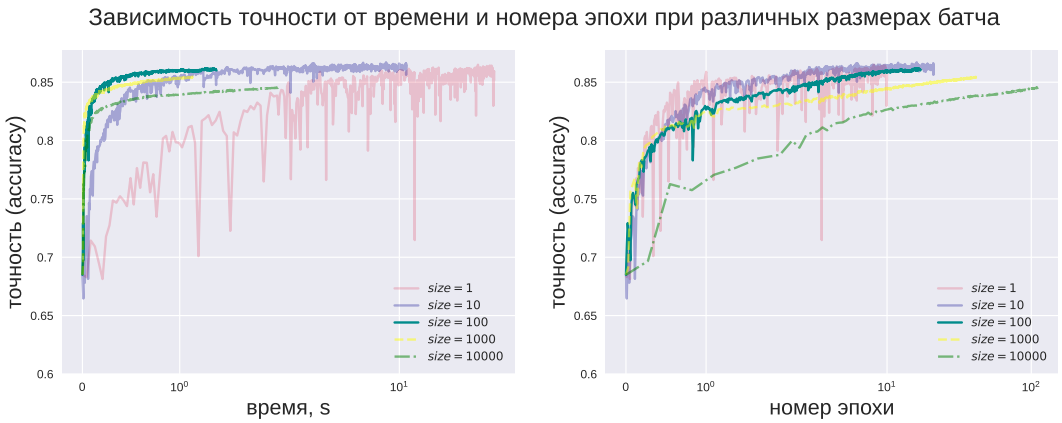

Рис. 16: Точность при различном выборе размера батча ($\alpha = 1$, $\beta = 0.5$) для SGD.

Дольше всего SGD работает при размере батча в 1 элемент (рис. 15, 16).Также, при таком размере оценка градиента функционала градиентом функции потерь на одном объекте достаточно зашумленная, поэтому наблюдаются частые скачки точности. То же самое можно сказать и про случай с размером батча в 10 элементов. При увеличении размера вплоть до 1000 уменьшается время сходимости. При размере 10000 время работы снова увеличивается. Лучший результат показывает алгоритм при размере батча в 100 объектов. Лучший результат наблюдается при размере батча в 100 объектов.

В последующих экспериментах будет использоваться следующий набор гиперпараметров для SGD:

- $\beta = 0.5$
- $\alpha = 0.9$
- $w_0 = 0$

3.5 Пятый эксперимент

В данном эксперименте сравнивается поведение GD и SGD при гиперпараметрах, отобранных в предыдущих экспериментах.

Из рис. 17, 18 видно, что SGD работает существенно быстрее чем GD (и в терминах времени, и в терминах эпох). Также, при SGD достигается лучшая точность.

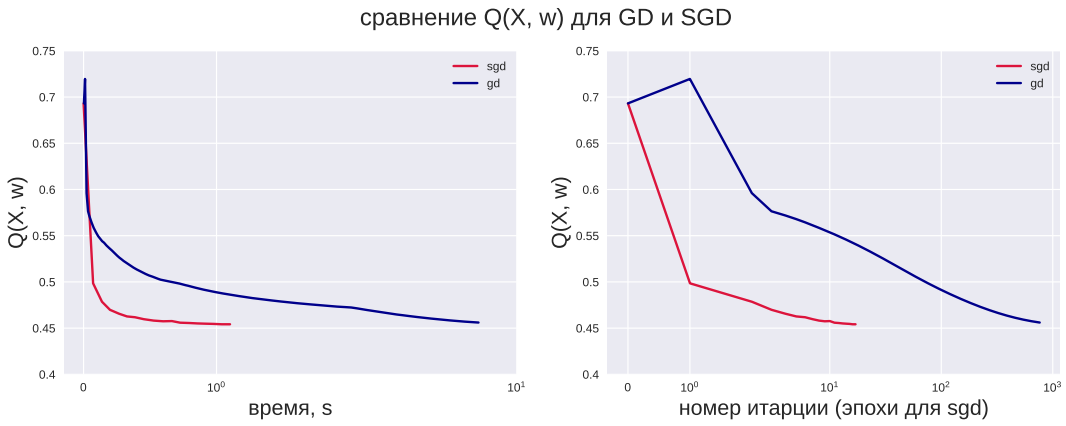

Рис. 17: Сравнение GD и SGD по значениям функционала Q(X, w).

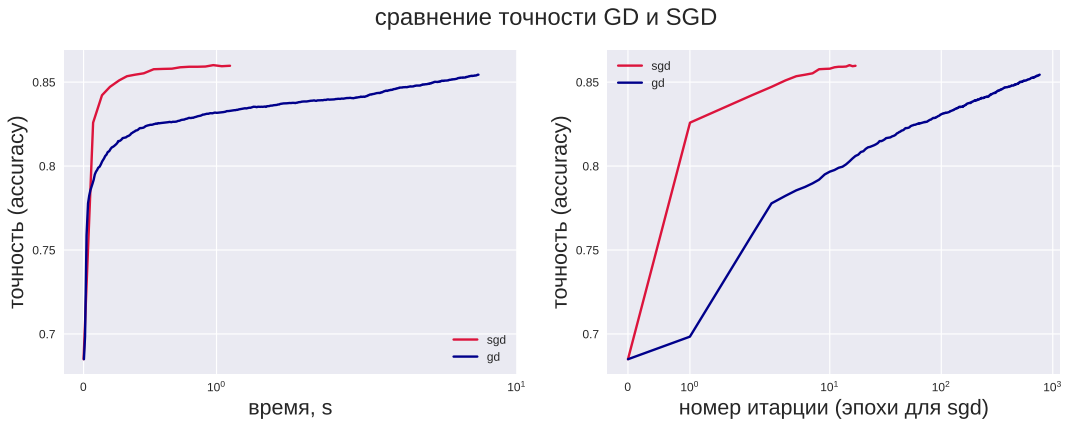

Рис. 18: Сравнение точности GD и SGD.

Меньшее количество проходов по всей выборке для SGD, объясняется тем, что в выборке часто могу находится объекты, не несущие полезной информации, а только увеличивающие время подсчета градиента. После перемешивания все объектов и смещении на градиент функционала на некоторой подвыборке, можно достичь сходимости на меньшем числе эпох. Сложность вычисления градиента на одной итерации: $O(p)$ для SGD, где p – размер батча, $O(l)$ для GD, где l – объем выборки. Поэтому SGD требуется меньше времени на сходимость при меньшем числе эпох. Однако, как было рассмотрено в предыдущих экспериментах, в отличие от GS, SGD не способен сходится при константном выборе шага ($\beta = 0$). Требуется динамическое уменьшение шага, либо ослабления критерия остановки.

3.6 Шестой эксперимент

В данном эксперименте к исходным данным применяется лемматизация (привидение слов в именительный падеж). Удаляются стоп-слова, которые встречаются с равной частотой как в токсичных комментариях, так и в обычных. Для лемматизации используется WordNetLemmatizer из библиотеки nltk. Для удаления стоп-слов используется их список также из библиотеки nltk. При неизменном параметре **min_df=0.001** признаковое пространство уменьшается с **3735** до **3023**. Результаты преобразования представлены на рис. 19

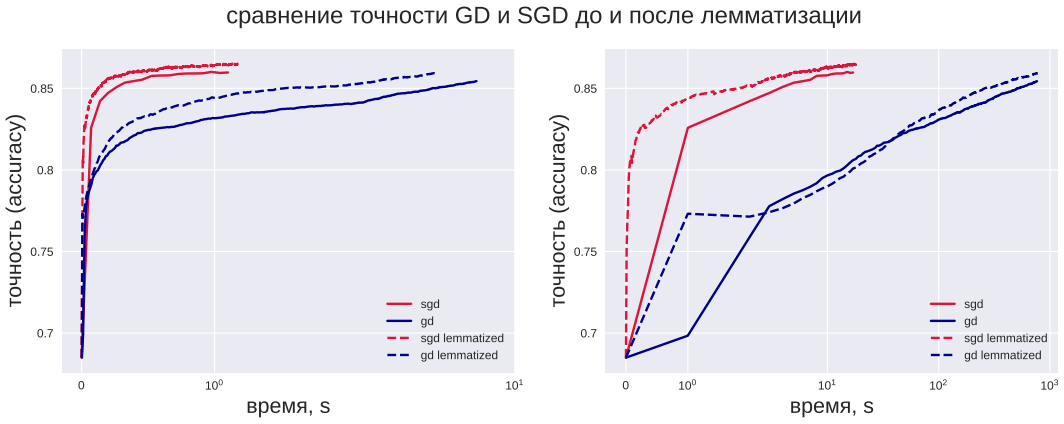

Рис. 19: Сравнение GD и SGD после лемматизации и удаления стоп-слов.

Несмотря на уменьшение количества признаков, увеличивается точность на валидационной выборке. Также, уменьшается время работы GD, при неизменной количестве итераций. Время работы SGD, напротив, незначительно увеличивается.

3.7 Седьмой эксперимент

В данном эксперименте сравниваются две модели векторного представления текстов:

- **Bag of words** (использовалась в предыдущих экспериментах) – j-ый признак текста равен числу вхождений в него j-го слова.
- **TF-IDF** – j-ый признак текста равен числу вхождений в него j-го слова, умноженному на обратную частоту этого слова по всем текстам выборки.

Также будет рассмотрена влияние параметров **min_df** и **max_df**, отвечающих за исключение слова встречаемость которых ниже или, соответственно, выше заданного порога.

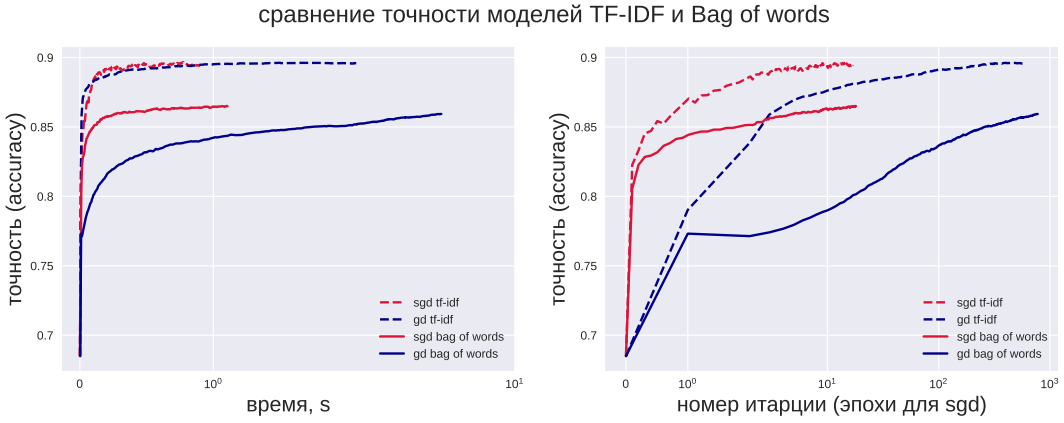

Рис. 20: Сравнение моделей **TF-IDF** и **Bag of words**.

Модель **TF-IDF** работает лучше (в терминах точности) и быстрее (в терминах времени), при практически неизменном числе итераций для GD и эпох для SGD (рис. 20).

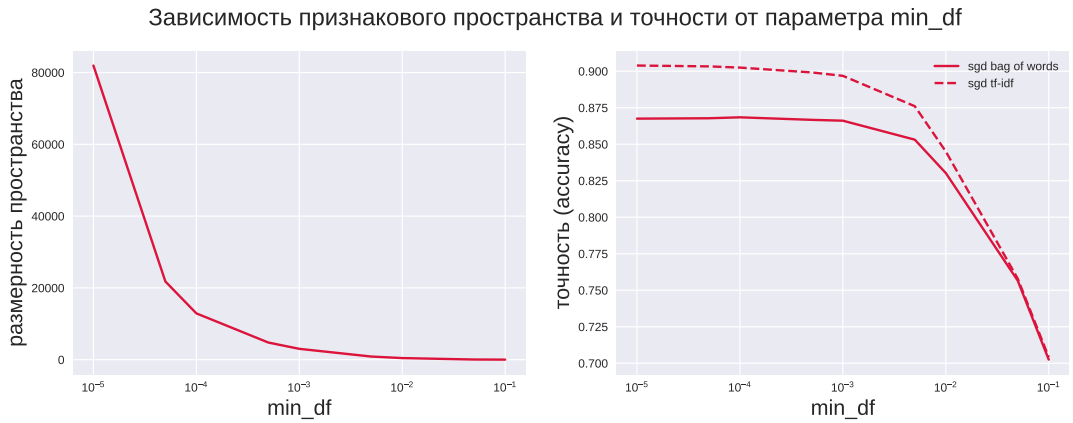

Рис. 21: Зависимость точности и размерности признакового пространства при различных значениях **min_df**.

При исследовании влияния различных значений параметров **min_df** и **max_df** (рис. 21, 22) используется алгоритм SGD после лемматизации выборки и удаления стоп-слов. Точность считается на валидационной выборке.

Для обеих моделей при увеличении параметра **min_df** наблюдается незначительная потеря качества при увеличении **min_df** с 10^{-5} до 10^{-3}. Однако Многократно снижается размерность признакового пространства (с 81945 до 3023). Это может говорить о том, что большинство очень редко встречающихся слов в документах не несут в себе полезной информации.

При уменьшении параметра **max_df** происходит заметное ухудшение точности вместе с уменьшением размерности признакового пространства. После удаления стоп-слов, большинство оставшихся признаков несут существенную смысловую нагрузку (как слова в документе).

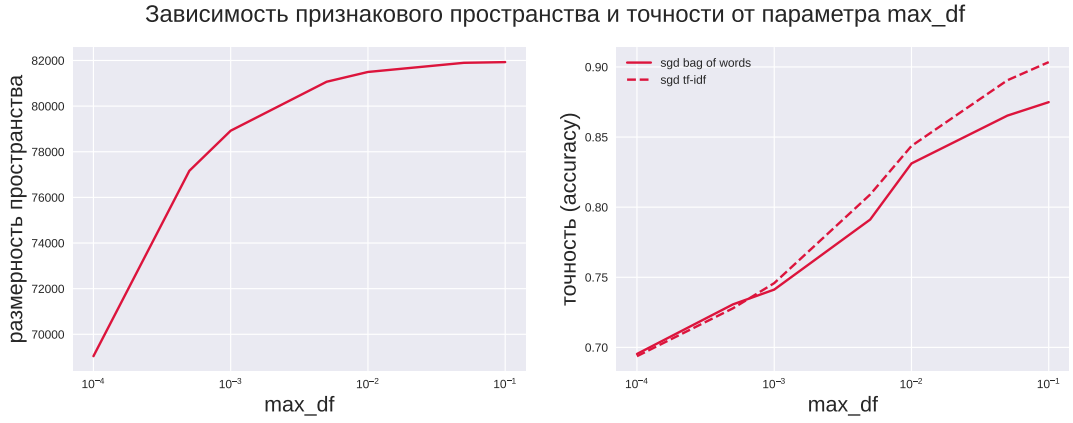

Рис. 22: Зависимость точности и размерности признакового пространства при различных значениях **min_df**.

3.8 Восьмой эксперимент

В рамках данного эксперимента исследуется качество лучшего алгоритма на тестовой выборке по результатам предыдущих экспериментов. Анализируются конкретные примеры документов на которых была допущена ошибка.

Используется алгоритм SGD с следующими гиперпараметрами:

- $\alpha = 0.9$
- $\beta = 0.5$
- $w_0 = 0$

Обучение происходит на данных после лемматизации и удаления стоп-слов. Используется модель **TF-IDF**. **min_df** $= 10^{-4}$, **max_df** $= 10^{-1}$. Полученная точность: **0.8707**. Ошибки классификации были допущены на **2143** объектах принадлежащих классу нетоксичных комментариев (FP – False Positive) и на **632** объектах принадлежащих к классу токсичных комментариев (FN – False Negative). В качестве примеров FP можно рассмотреть следующие комментарии:

1. “12121212121212 jews jews jews jews jews jews jews jews jews jews jews jews jews jews jews jews jews”
2. “I WILL BURN YOU TO HELL IF YOU REVOKE MY TALK PAGE ACCESS!!!!!!!!!!!!”
3. “== black mamba == It.is ponious snake of the word and but it not kills many people but king cobra kills many people in India”

В документе (1) содержится слово “jews”, которое хоть и не является токсичным само по себе, но часто фигурирует в токсичных комментариях. Аналогично со словом “kills” в документе (3). Документ (2) содержит такие слова как “BURN” и “HELL”, которые тоже часто фигурируют в токсичных комментариях. Стоит отметить, что отсутствие токсичности в комментарии (2) может быть поставлено под вопрос. Примеры FN:

1. “Y una mierda.Tu puta madre””
2. “How dare you vandalize that page about the HMS Beagle! Don’t vandalize again, demon”
3. “Please, someone fix this godawful article.”

Документ (2) написан на испанском языке, поэтому слова встречающиеся в нем очень редки среди всех документов. Можно предположить, что они отбрасываются параметром **min_df**. В документах (2) и (3) не употребляются слова, часто встречающиеся в токсичных комментариях.

3.9 Первое бонусное задание

В данном пункте исследуется влияние длин максимальных n-грамм (последовательности из n слов) на качество и время работы алгоритма. Используется параметр **ngramm_range** конструктора класса **TfidfVectorizer**. Точность исследуется на валидационной выборке используемой в эксперимента 3-7.

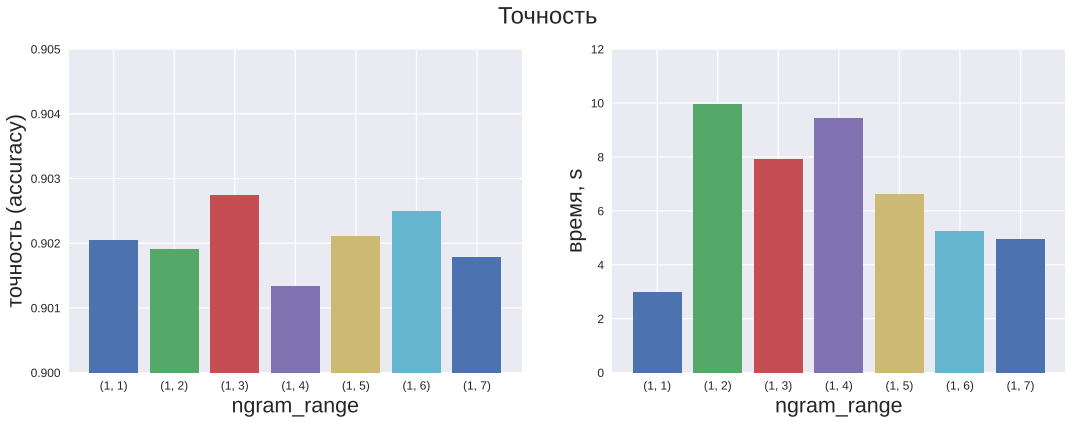

Рис. 23: Зависимость точности и размерности признакового пространства при различных значениях **min_df**.

Добавление n-грамм дает некоторый прирост точности, однако время работы алгоритма заметно увеличивается (рис. 23). Лучший результат точности показывает алгоритм при **ngram_range=(1,3)**. На тестовой выборке качество немного увеличивается: при ngram_range=(1,3) достигается точность **0.8736**.

3.10 Второе бонусное задание

В рамках данного задания реализуется режим работы SGD, при котором батчи считываются с диска и выборка не хранится в оперативной памяти. Сравнивается время работы одной эпохи и точность SGD с модифицированным SGD. Оценивается экономия памяти. Преобразованная выборка хранится в формате **npy**.

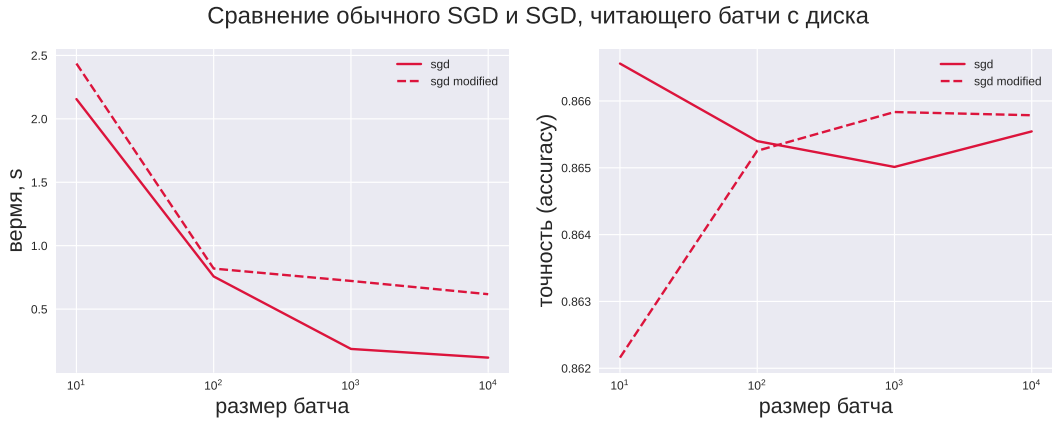

Рис. 24: Сравнение двух режимов работы SGD.

На рис. 24 сравниваются время и точность двух режимом работы SGD по результатам 10 эпох при различных размерах батча. При считывании батчей с файла алгоритм работает немного медленнее. Точность сравнивается на тестовой выборке. При всех размерах батча кроме 10, обычный режим SGD работает точнее. Экономия памяти составляет примерно **21МБ**. Для подсчета расхода памяти используется **tracemalloc**.

3.11 Третье бонусное задание

В предыдущих экспериментах в признаковом пространстве не находилось константного признака. При алгоритме SGD ($\alpha = 0.9$, $\beta = 0.5$, $w_0 = 0$, модель TF-IDF, min_df=10^{-4}) добавление константного признака дает существенный прирост в точности на тестовой выборке: точность – **0.8885**.

4 Выводы

SGD показывает большую скорость сходимости чем GD. Модель **TF-IDF** опережает по точности модель **Bag of words** в данной задаче. Лучшие результаты показывает алгоритм SGD после лемматизации и удаления стоп-слов, применения модели **TF-IDF** со следующими гиперпараметрами:

- $\alpha = 0.9$
- $\beta = 0.5$
- $w_0 = 0$
- min_df $= 10^{-3}$
- max_df $= 10^{-1}$
- ngram_range $= (1, 3)$

Итоговая точность на тестовой выборке с добавлением константного признака – **0.8885**.