

Применение линейных моделей для определения токсичности  
комментария

Григорьев Илья, 317 группа

Ноябрь, 2020

# Введение

В данном практическом задании надо было написать свою реализацию логистической регрессии для бинарной классификации. Для обучения модели использовались метод градиентного спуска и стохастического градиентного спуска.

Модель надо было применить для бинарной классификации текстов на токсичность. Для предобработки текстов применялись разные методы: приведение к нижнему регистру, удаление символов с помощью регулярных выражений, лемматизация, удаление стоп слов. Для кодирования текстов использовались методы Bag of Words и Tf-idf.

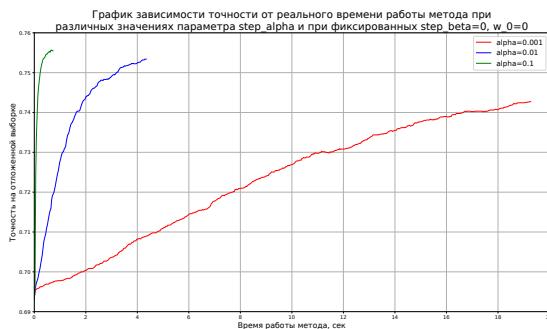
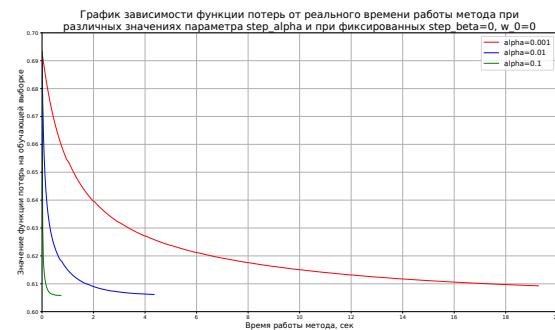
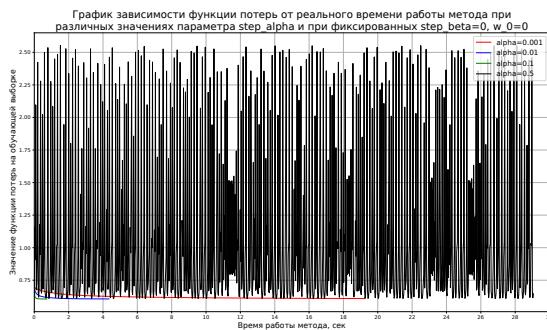
## Эксперименты 1 – 2

После приведения всех текстов к нижнему регистру, замены всех символов, не являющихся буквами и цифрами, на пробелы и кодирования Bag of Words, размер признакового пространства стал равен 18254. Параметры `min_df` и `max_df` конструктора `CountVectorizer` были установлены соответственно 5 и 1.0.

## Эксперимент 3

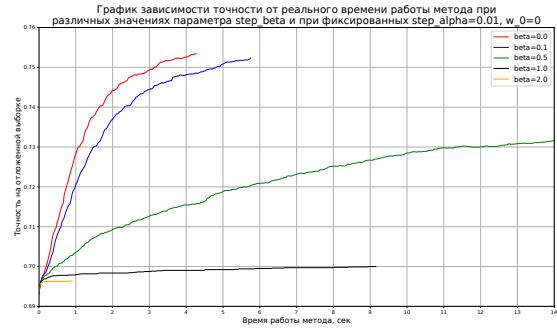
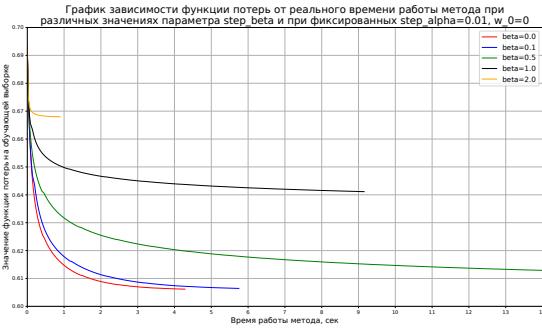
На каждой итерации градиентного спуска шаг метода выбирается по следующей формуле:  
 $learning\_rate = step\_alpha/k^{step\_beta}$ , где  $k$  это номер итерации. Исследуем поведение метода градиентного спуска для задачи логистической регрессии в зависимости от параметров: `step_alpha`, `step_beta` и начального приближения.

При фиксированных `step_beta = 0` и начальном приближении равном нулевому вектору, обучая метод с разными `step_alpha` получаются следующие графики.



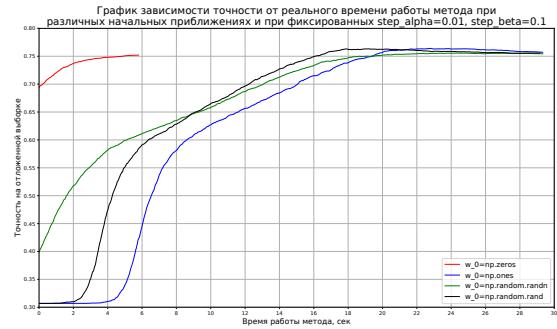
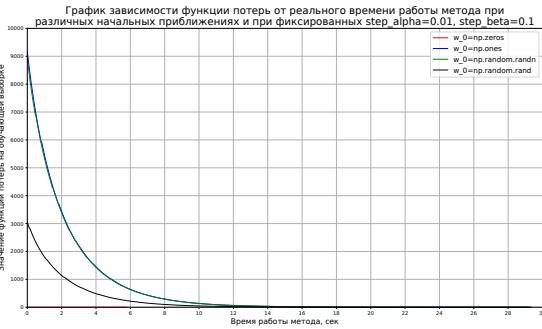
Так как `step_beta = 0`, `learning_rate` получается не зависящим от номера итерации и равен `step_alpha`. При `step_alpha = 0.5` метод не может удержаться в точке минимума и поэтому расходится, надо брать меньшие значения для `step_alpha`. При `step_alpha = 0.1, 0.01, 0.001` метод градиентного спуска сходится, причем чем больше значение этого параметра, тем быстрее наступает сходимость.

Рассмотрим параметр `step_beta` при фиксированных `step_alpha = 0.01` и начальном приближении равном нулевому вектору.



По графикам видно, что при больших  $step\_beta learning\_rate$  очень быстро стремится к нулю, и метод сходится далеко от минимума. Оптимальным значением является  $step\_beta = 0.1$ .

Рассмотрим различные начальные приближения при фиксированных  $step\_alpha = 0.01$  и  $step\_beta = 0.1$ .



По графикам видно, что начальное приближение, равное нулевому вектору, находится ближе всех к минимуму. Дальше всех от оптимума расположены начальные приближения равные вектору из единиц и вектору из равномерно распределенных на отрезке  $[0, 1]$  чисел.

## Эксперимент 4

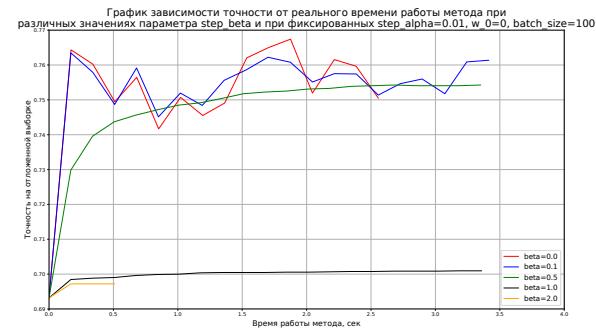
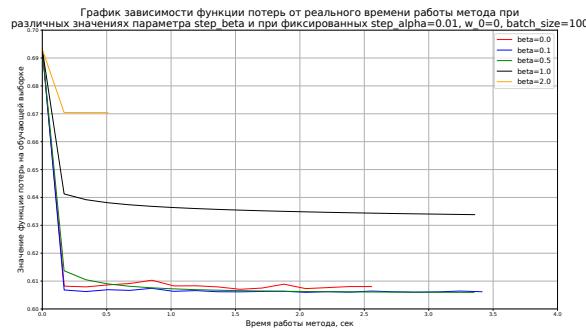
В этом эксперименте исследуем поведение метода стохастического градиентного спуска для задачи логистической регрессии в зависимости от параметров:  $step\_alpha$ ,  $step\_beta$ , начального приближения и  $batch\_size$ .

Рассмотрим параметр  $step\_alpha$ .



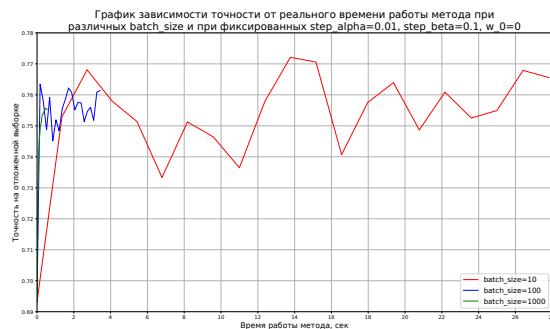
Для стохастического градиентного спуска нужен более короткий шаг, так как градиент пересчитывается не по всей обучающей выборке, а только по ее подмножеству, а потому направление антиградиента может быть неточным. SGD с параметром  $step\_alpha = 0.1$  не сопшелся за 20 эпох.

Рассмотрим параметр *step\_beta*.



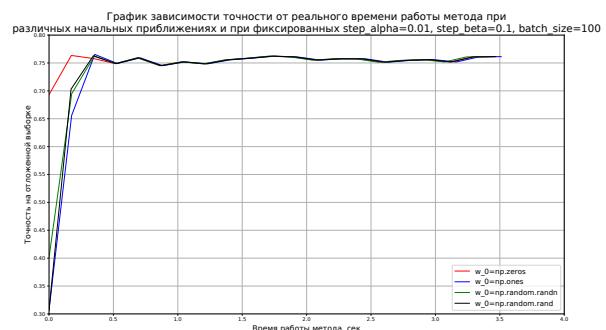
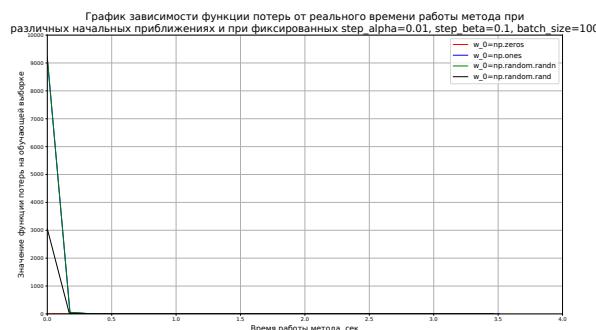
С этим параметром ситуация аналогична методу градиентного спуска. Большие значения *step\_beta* быстро уменьшают *learning\_rate* и метод сходится вдалеке от минимума.

Рассмотрим параметр *batch\_size*.



По графикам видно, что при значении параметра равным 1000 метод стохастического градиентного спуска сходится быстрее, а колебания функции потерь меньше, потому что градиент вычисляется более точно. С другой стороны, по графику зависимости точности на отложенной выборке от реального времени работы метода видно, что алгоритм с *batch\_size* = 10 достигает лучшего результата.

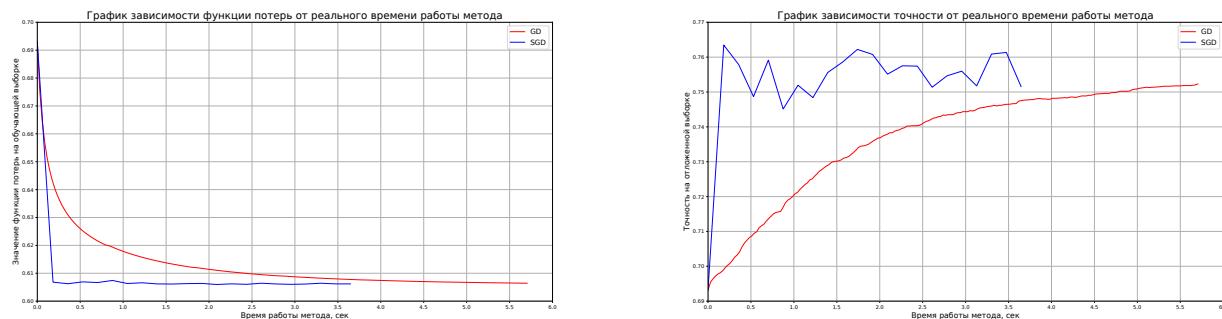
Рассмотрим как различные начальные приближения влияют на метод стохастического градиентного спуска.



Эти начальные приближения расположены относительно минимума функции потерь аналогично методу градиентного спуска. Отличие заключается в том, что SGD намного быстрее GD выходит на одинаковую точность на отложенной выборке из этих четырех начальных приближений.

## Эксперимент 5

Сравним поведение методов градиентного спуска и стохастического градиентного спуска между собой.



По графикам видно, что SGD намного быстрее обучается и достигает высокой точности на отложенной выборке. Если на небольшом датасете может быть выгоднее использовать GD, то при работе с огромным количеством объектов в обучающей выборке намного лучше использовать SGD. На небольшом датасете из эксперимента разница в скорости работы двух методов не сильно большая: 5.71 секунд против 3.65 секунд. Так же у метода стохастического градиентного спуска есть серьезное преимущество перед обычным методом градиентного спуска: если оптимизационная задача не выпуклая, то SGD может сойтись к глобальному минимуму, а GD сойдется к локальному минимуму.

## Эксперимент 6

В этом эксперименте на отложенной выборке был подобран порог для бинаризации вероятностей с целью повышения точности классификации. Точность на тестовой выборке составила 0.7707. Затем была проведена дополнительная предобработка текста: лемматизация с определением частей речи, удаление стоп слов. После такой предобработки точность на тестовой выборке выросла до 0.8063, при этом количество признаков уменьшилось с 18254 до 14559, и, разумеется, модель стала обучаться быстрее.

## Эксперимент 7

Здесь были исследованы качество, время работы алгоритма и размер признакового пространства в зависимости от параметров `min_df` и `max_df`, а так же от того используется `Bag of Words` или `Tf-idf` кодирование.

На данных из эксперимента качество выше с `Bag of Words`. Оптимальные значения параметров `min_df` и `max_df` равны соответственно 6 и 0.1. То-есть мы отбросили больше слишком редких и слишком частых слов среди документов. Количество признаков уменьшилось с 14559 до 12837, скорость обучения соответственно увеличилась, а точность на тестовой выборке выросла до 0.8289.

## Эксперимент 8

Выбрав лучший алгоритм и подобрав параметр регуляризации, удалось достичь точности 0.8816 на тестовой выборке.

Проанализируем ошибки алгоритма. Легко заметить общие черты объектов, на которых были сделаны неправильные предсказания. Ошибки типа `False Positive` возникают, когда используются резкие агрессивные слова, но либо не в отношении кого-то или без реальной злости, либо в устойчивых выражениях и словосочетаниях. Ошибки типа `False Negative` возникают, когда кого-то оскорбляют, но без использования ярких слов, не слишком агрессивно, поэтому алгоритму сложно отловить такие комментарии.

## Теория

$$\begin{aligned}
 f(w) &= \sum_{k=1}^N \mathbb{I}(w^T x_k \cdot y_k) + \lambda \langle w, w \rangle = \\
 &= \sum_{k=1}^N \ln(1 + e^{-w^T x_k y_k}) + \lambda \langle w, w \rangle \\
 df(w) &= \langle \nabla_w f(w), dw \rangle = \sum_{k=1}^N \frac{-e^{-w^T x_k y_k} \cdot d(w^T x_k y_k)}{1 + e^{-w^T x_k y_k}} + 2\lambda \langle w, dw \rangle \\
 &= \sum_{k=1}^N \frac{-e^{-w^T x_k y_k} \cdot y_k d\langle w, x_k \rangle}{1 + e^{-w^T x_k y_k}} + \langle 2\lambda w, dw \rangle = \\
 &= \sum_{k=1}^N \frac{-y_k e^{-y_k \langle w, x_k \rangle} \cdot \langle x_k, dw \rangle}{1 + e^{-y_k \langle w, x_k \rangle}} + \langle 2\lambda w, dw \rangle = \\
 &= \left\{ \begin{array}{l} -y_k \langle w, x_k \rangle \\ e^{-y_k \langle w, x_k \rangle} = z \end{array} \right\} = \sum_{k=1}^N \left\langle \frac{-y_k \cdot z \cdot x_k}{1+z}, dw \right\rangle + \langle 2\lambda w, dw \rangle \\
 \nabla_w f(w) &= \sum_{k=1}^N \frac{-y_k \cdot e^{-y_k \langle w, x_k \rangle} \cdot x_k}{1 + e^{-y_k \langle w, x_k \rangle}} + 2\lambda w
 \end{aligned}$$

$$\left\{ \begin{array}{l} \prod_{k=1}^N \frac{e^{-y_k \langle w, x_k \rangle}}{\sum_{i=1}^c e^{-y_i \langle w, x_k \rangle}} \\ w_c = 0 \end{array} \right. \rightarrow \max_{w_1, \dots, w_{c-1}}$$

$$\sum_{k=1}^N \ln \frac{e^{-y_k \langle w, x_k \rangle}}{\sum_{i=1}^c e^{-y_i \langle w, x_k \rangle}} \rightarrow \max_{w_1, \dots, w_{c-1}}$$

$$\begin{aligned}
 df &= \langle \nabla f, d\bar{w} \rangle = \\
 &= \sum_{k=1}^N \left( \langle x_k, dw_k \rangle - \frac{\sum_{i=1}^C e^{\langle w_i, x_k \rangle} \cdot \langle x_k, dw_i \rangle}{\sum_{i=1}^C e^{\langle w_i, x_k \rangle}} \right) = \\
 &= \sum_{k=1}^N \left( \langle x_k, dw_k \rangle - \sum_{i=1}^C \left\langle \frac{e^{\langle w_i, x_k \rangle} \cdot x_k}{\sum_{i=1}^C e^{\langle w_i, x_k \rangle}}, dw_i \right\rangle \right)
 \end{aligned}$$