## 1 Постановка задачи

Рассмотрим задачу классификации:

Пусть имеется два датасета: $D_0$ и $D_1$, объекты в которых можно относятся к двум классам: $C_0$ и $C_1$. Пусть

$$p(x \in C_0 | x \in D_0) \equiv \alpha$$

$$p(x \in C_0 | x \in D_1) \equiv \beta$$

($x$ – равномерно распределённая случайная величина со значениями в множестве всех объектов), и $\alpha$ и $\beta$, вообще говоря, неизвестны; но известно, что $\alpha < \beta$. Пусть $D_0$ и $D_1$ ничем более не отличаются. Условно назовём объекты, принадлежащие $C_0$, *сигналами*, а объекты, принадлежащие $C_1$, *шумами*

Хотим решать задачу классификации выборки объектов $x_1, \ldots, x_n$ на принадлежность её к $D_0$ или $D_1$ (причём лейблы $x$-ов, указывающие на принадлежность к $C_0$ или $C_1$, неизвестны), т.е. построить аппроксимацию распределения $p(x \in D_1 | x)$.

Эта задача важна потому, что при разумном допущении разделимости классов $C_0$ и $C_1$ из решения этой задачи можно получить решение задачи классификации $C_0$ против $C_1$ (напомним, что лейблы $C_0$ и $C_1$ у объектов неизвестны).

## 2 Переход от задачи $D_0$ против $D_1$ к задаче $C_0$ против $C_1$

Заметим следующее:

$$p(x \in D_1 | x) = p(x \in D_1 \cap C_0 | x) + p(x \in D_1 \cap C_1 | x) =$$

$$= p(x \in D_1 | x \in C_0, x) p(x \in C_0 | x) + p(x \in D_1 | x \in C_1, x) p(x \in C_1 | x)$$

В предположении о том, что

- $p(x \in D_0 | x \in C_0, x) = p(x \in D_0 | x \in C_0)$,
- $p(x \in D_0 | x \in C_1, x) = p(x \in D_0 | x \in C_1)$,
- $p(x \in D_1 | x \in C_0, x) = p(x \in D_1 | x \in C_0)$,
- $p(x \in D_1 | x \in C_1, x) = p(x \in D_1 | x \in C_1)$

Преобразуем это выражение:

$$= p(x \in D_1 | x \in C_0) p(x \in C_0 | x) + p(x \in D_1 | x \in C_1) p(x \in C_1 | x) =$$

$$= \frac{p(x \in C_0 | x \in D_1) p(x \in D_1)}{p(x \in C_0 | x \in D_1) p(x \in D_1) + p(x \in C_0 | x \in D_0)(x \in D_0)} p(x \in C_0 | x) +$$
$$+ \frac{p(x \in C_1 | x \in D_1) p(x \in D_1)}{p(x \in C_1 | x \in D_1) p(x \in D_1) + p(x \in C_1 | x \in D_0)(x \in D_0)} p(x \in C_1 | x) =$$
$$= \frac{\frac{1}{2}\beta}{\frac{1}{2}\beta + \frac{1}{2}\alpha} p(x \in C_0 | x) + \frac{\frac{1}{2}(1-\beta)}{\frac{1}{2}(1-\beta) + \frac{1}{2}(1-\alpha)} (x \in C_1 | x) =$$
$$= \frac{\beta}{\alpha + \beta} p(x \in C_0 | x) + \frac{1-\beta}{2-\alpha-\beta} p(x \in C_1 | x) =$$
$$= \frac{\beta}{\alpha + \beta} (1 - p(x \in C_1 | x)) + \frac{1-\beta}{2-\alpha-\beta} p(x \in C_1 | x) =$$
$$= (\frac{1-\beta}{2-\alpha-\beta} - \frac{\beta}{\alpha+\beta}) p(x \in C_1 | x) + \frac{\beta}{\alpha+\beta}$$

Т.е. зная доли класса $C_0$ в $D_0$ и $D_1$ и имея классификатор $p(x \in D_1 | x)$ можно построить классификатор $p(x \in C_1 | x)$ преобразованием:

$$p(x \in C_1 | x) = (p(x \in D_1 | x) - \frac{\beta}{\alpha+\beta})(\frac{1-\beta}{2-\alpha-\beta} - \frac{\beta}{\alpha+\beta})^{-1}$$

Ну а $\alpha$ и $\beta$ можно примерно узнать, пристально взглянув на ROC - кривую, или же разметив вручную часть объектов.

## 3 Процесс обучения

Классический подход предлагает нам выбрать семейство моделей $f_\theta : \Theta \times X \to [0, 1]$ и минимизировать кроссэнтропию

$$\mathcal{L} = -\mathop{\mathbb{E}}_{x,t}(p(x \in D_1 | x) f_\theta(x) + (1 - p(x \in D_1 | x))(1 - f_\theta(x)))$$

на обучающей выборке.

Заметим, что в случае сложной структуры функции $f_\theta$ (например, когда это - свёрточная нейросеть)не приходится надеяться на успех в аналитическом поиске минимума данной функции, и необходимо пользоваться приближёнными методами, например, стохастическим градиентным спуском.

При использовании подобных методов без каких-либо улучшений возникает проблема, заключающаяся в низкой скорости работы данного метода: так как классы $D_0$ и $D_1$ существенно различаются только долями классов $C_0$, а

доли эти в прикладных задачах близки к нулю, то обучение может длиться долго: члены $\mathcal{L}$, отвечающие двум сходным объектам класса $C_1$, принадлежащим $D_0$ и $D_1$ соответственно, имеют различные знаки градиента; причём число таких объектов в обучающей выборке преобладает, при всей их бесполезности для процесса обучения.

## 4 Метод борьбы номер 1. Вспомогательный классификатор

Идея этого метода в том, чтобы начать не сразу с обучения классификатора $f$ (сложный мощный классификатор (нейросеть)), а обучить сначала более простой классификатор $f_0$ (линейная регрессия), после чего изменить лейблы объектов для нашей задачи следующим образом: если у объета $x_i$ был лейбл $q_i$ (0 или 1), то новый лейбл будет равен $f_0(x_i)$.

Почему это хорошо? Рассмотрим, как происходит процесс обучения на первых итерациях без привлечения $f_0$. Пусть в батче, поданном на вход SGD, встретился объект из $D_0$ $C_1$. Градиент лосса на нём будет большим (по норме), что куда-то сдвинет наши параметры. Хотя мы точно знаем, что где-то неподалёку от этого объекта (в пространстве признаков) существует ровно такой же (ну или близкий) объект, но уже из $D_1$ $C_1$, градиент лосса на котором будет иметь другой знак (и тоже будет большим). В итоге, два этих движения в пространстве параметров частично взаимосократятся (в итоге приведя значение классификатора к примерно $\frac{1}{2}$), но это может занять много времени (пока не встретится объект - пара из иного $D_i$). Но времени к этому моменту скорее всего пройдёт много (обучающая выборка большая и разнообразная).

Заметим, что этой проблемы можно было бы избежать, если бы значение лейблов таких объектов (из $C_1$) будет равно (или будет близко) $\frac{1}{2}$. Тогда модель будет обучаться тому, что значение классификатора на таких объектах должно быть равно $\frac{1}{2}$ уже в рамках одного батча, так что произойдёт это быстро, т.е. всё обучение займёт меньше времени.

Именно для этого и обучается лёгковесная модель $f_0$: чтобы быстро научиться определять объекты из $C_1$. Проставив объектам новые лейблы, можно приступать к обучению основного классификатора.