Урок 2

Линейные модели

2.1. Линейные модели в задачах регрессии

Данный урок будет посвящен линейным моделям. Речь пойдет о задачах классификации и регрессии, как их обучать, и с какими проблемами можно столкнуться при использовании этих моделей. В этом блоке мы обсудим, как выглядят линейные модели в задачах регрессии.

2.1.1. Повторение обозначений из прошлого урока

Для начала необходимо напомнить некоторые обозначения, которые были введены на прошлом уроке.

- $\mathbb{X}$ — пространство объектов
- $\mathbb{Y}$ — пространство ответов
- $x = (x^1, ..., x^d)$ — признаковое описание объекта
- $X = (x_i, y_i)_{i=1}^{\ell}$ — обучающая выборка
- $a(x)$ — алгоритм, модель
- $Q(a, X)$ — функционал ошибки алгоритма a на выборке X
- Обучение: $a(x) = \text{argmin}_{a \in \mathbb{A}} Q(a, X)$

Напомним, что в задаче регрессии пространство ответов $\mathbb{Y} = \mathbb{R}$. Чтобы научиться решать задачу регрессии, необходимо задать:

- **Функционал ошибки** Q: способ измерения того, хорошо или плохо работает алгоритм на конкретной выборке.
- **Семейство алгоритмов** $\mathbb{A}$: как выглядит множество алгоритмов, из которых выбирается лучший.
- **Метод обучения**: как именно выбирается лучший алгоритм из семейства алгоритмов.

2.1.2. Пример задачи регрессии: предсказание прибыли магазина

Пусть известен один признак — прибыль магазина в прошлом месяце, а предсказать необходимо прибыль магазина в следующем. Поскольку прибыль — вещественная переменная, здесь идет речь о задаче регрессии.

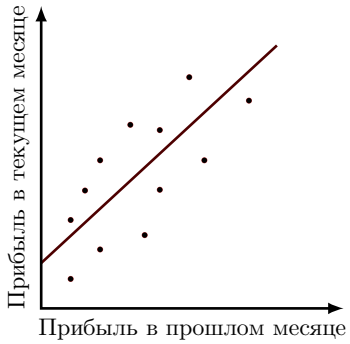

Рис. 2.1: Точки обучающей выборки.

По этому графику можно сделать вывод о существовании зависимости между прибылью в следующем и прошлом месяцах. Если предположить, что зависимость приблизительно линейная, ее можно представить в виде прямой на этом графике. По этой прямой и можно будет предсказывать прибыль в следующем месяце, если известна прибыль в прошлом.

В целом такая модель угадывает тенденцию, то есть описывает зависимость между ответом и признаком. При этом, разумеется, она делает это не идеально, с некоторой ошибкой. Истинный ответ на каждом объекте несколько отклоняется от прогноза.

Один признак — это не очень серьезно. Гораздо сложнее и интереснее работать с многомерными выборками, которые описываются большим количеством признаков. В этом случае нарисовать выборку и понять, подходит или нет линейная модель, нельзя. Можно лишь оценить ее качество и по нему уже понять, подходит ли эта модель.

Следует отметить, что вообще нельзя придумать модель, которая идеально описывает ваши данные, то есть идеально описывает, как порождается ответ по признакам.

2.1.3. Описание линейной модели

Далее обсудим, как выглядит семейство алгоритмов в случае с линейными моделями. Линейный алгоритм в задачах регрессии выглядит следующим образом:

$$a(x) = w_0 + \sum_{j=1}^{d} w_j x^j,$$

где w_0 — свободный коэффициент, x^j — признаки, а w_j — их веса.

Если добавить $(d+1)$-й признак, который на каждом объекте принимает значение 1, линейный алгоритм можно будет записать в более компактной форме

$$a(x) = \sum_{j=1}^{d+1} w_j x^j = \langle w, x \rangle,$$

где используется обозначение $\langle w, x \rangle$ для скалярного произведения двух векторов.

В качестве меры ошибки не может быть выбрано отклонение от прогноза $Q(a, y) = a(x) - y$, так как в этом случае минимум функционала не будет достигаться при правильном ответе $a(x) = y$. Самый простой способ — считать модуль отклонения:

$$|a(x) - y|.$$

Но функция модуля не является гладкой функцией, и для оптимизации такого функционала неудобно использовать градиентные методы. Поэтому в качестве меры ошибки часто выбирается квадрат отклонения:

$$(a(x) - y)^2.$$

Функционал ошибки, именуемый среднеквадратичной ошибкой алгоритма, задается следующим образом:

$$Q(a, x) = \frac{1}{\ell} \sum_{i=1}^{\ell} (a(x_i) - y_i)^2$$

В случае линейной модели его можно переписать в виде функции (поскольку теперь Q зависит от вектора, а не от функции) ошибок:

$$Q(w, x) = \frac{1}{\ell} \sum_{i=1}^{\ell} (\langle w, x_i \rangle - y_i)^2 .$$

2.2. Обучение модели линейной регрессии

В этом блоке речь пойдет о том, как обучать модель линейной регрессии, то есть как настраивать ее параметры. В прошлый раз было введено следующее выражение для качества линейной модели на обучающей выборке:

$$Q(w, x) = \frac{1}{\ell} \sum_{i=1}^{\ell} (\langle w, x_i \rangle - y_i)^2 \to \min_w .$$

Следует напомнить, что в число признаков входит также постоянный признак, равный 1 для всех объектов, что позволяет исключить постоянную составляющую в последнем соотношении.

2.2.1. Переход к матричной форме записи

Прежде, чем будет рассмотрена задача оптимизации этой функции, имеет смысл переписать используемые соотношения в матричной форме. Матрица «объекты–признаки» X составлена из признаковых описаний всех объектов из обучающей выборки:

$$X = \begin{pmatrix} x_{11} & \dots & x_{1d} \\ \dots & \dots & \dots \\ x_{\ell 1} & \dots & x_{\ell d} \end{pmatrix}$$

Таким образом, в ij элементе матрицы X записано значение j-го признака на i объекте обучающей выборки. Также понадобится вектор ответов y, который составлен из истинных ответов для всех объектов:

$$y = \begin{pmatrix} y_1 \\ \dots \\ y_\ell \end{pmatrix}.$$

В этом случае среднеквадратичная ошибка может быть переписана в матричном виде:

$$Q(w, X) = \frac{1}{\ell}\|Xw - y\|^2 \to \min_w.$$

Эта формула пригодится, в частности, при реализации линейной регрессии на компьютере.

2.2.2. Аналитический метод решения

Можно найти аналитическое решение задачи минимизации:

$$w_* = (X^T X)^{-1} X^T y.$$

Основные сложности при нахождении решения таким способом:

- Для нахождения решения необходимо вычислять обратную матрицу. Операция обращения матрицы требует, в случае d признаков, выполнение порядка d^3 операции, и является вычислительно сложной уже в задачах с десятком признаков.
- Численный способ нахождения обратной матрицы не может быть применен в некоторых случаях (когда матрица плохо обусловлена).

2.2.3. Оптимизационный подход к решению

Другой, несколько более удобный, способ найти решение — использовать численные методы оптимизации.

Несложно показать, что среднеквадратическая ошибка — это выпуклая и гладкая функция. Выпуклость гарантирует существование лишь одного минимума, а гладкость — существование вектора градиента в каждой точке. Это позволяет использовать метод градиентного спуска.

При использовании метода градиентного спуска необходимо указать начальное приближение. Есть много подходов к тому, как это сделать, в том числе инициализировать случайными числами (не очень большими). Самый простой способ это сделать — инициализировать значения всех весов равными нулю:

$$w^0 = 0.$$

На каждой следующей итерации, $t = 1, 2, 3, ...$, из приближения, полученного в предыдущей итерации w^{t-1}, вычитается вектор градиента в соответствующей точке w^{t-1}, умноженный на некоторый коэффициент η_t, называемый шагом:

$$w^t = w^{t-1} - \eta_t \nabla Q(w^{t-1}, X).$$

Остановить итерации следует, когда наступает сходимость. Сходимость можно определять по-разному. В данном случае разумно определить сходимость следующим образом: итерации следует завершить, если разница двух последовательных приближений не слишком велика:

$$\|w^t - w^{t-1}\| < \varepsilon.$$

Подробно метод градиентного спуска обсуждался в прошлом курсе этой специализации.

2.3. Градиентный спуск для линейной регрессии

2.3.1. Случай парной регрессии

В случае парной регрессии признак всего один, а линейная модель выглядит следующим образом:

$$a(x) = w_1 x + w_0,$$

где w_1 и w_0 — два параметра.

Среднеквадратичная ошибка принимает вид:

$$Q(w_0, w_1, X) = \frac{1}{\ell} \sum_{i=1}^{\ell} (w_1 x_i + w_0 - y_i)^2.$$

Для нахождения оптимальных параметров будет применяться метод градиентного спуска, про который уже было сказано ранее. Чтобы это сделать, необходимо сначала вычислить частные производные функции ошибки:

$$\frac{\partial Q}{\partial w_1} = \frac{2}{\ell} \sum_{i=1}^{\ell} (w_1 x_i + w_0 - y_i) x_i, \qquad \frac{\partial Q}{\partial w_0} = \frac{2}{\ell} \sum_{i=1}^{\ell} (w_1 x_i + w_0 - y_i).$$

2.3.2. Демонстрация градиентного спуска в случае парной регрессии

Следующие два графика демонстрируют применение метода градиентного спуска в случае парной регрессии. Справа изображены точки выборки, а слева — пространство параметров. Точка в этом пространстве обозначает конретную модель.

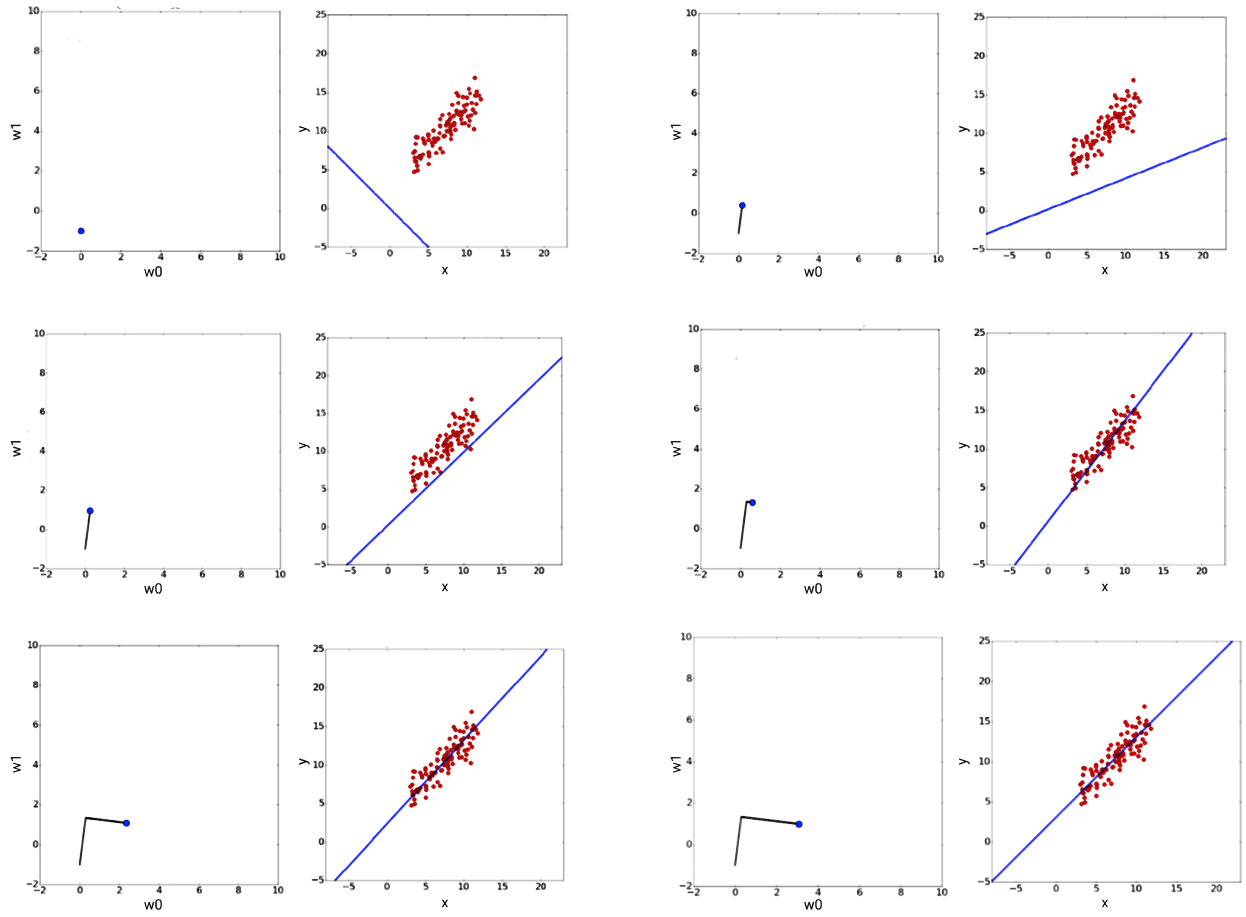

Рис. 2.2: Демонстрация метода градиентного спуска.

График зависимости функции ошибки от числа произведенных операции выглядит следующим образом:

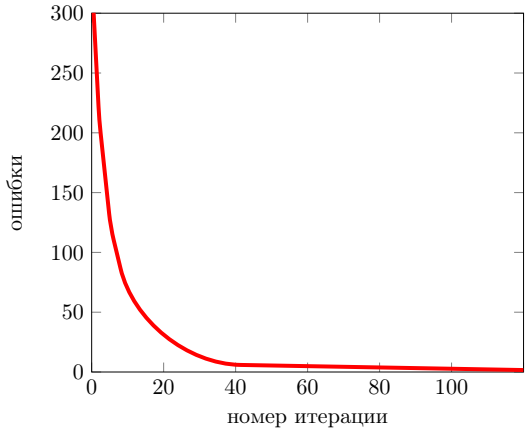

Рис. 2.3: Ошибка в зависимости от номера итерации

2.3.3. Выбор размера шага в методе градиентного спуска

Очень важно при использовании метода градиентного спуска правильно подбирать шаг. Каких-либо конкретных правил подбора шага не существует, выбор шага — это искусство, но существует несколько полезных закономерностей.

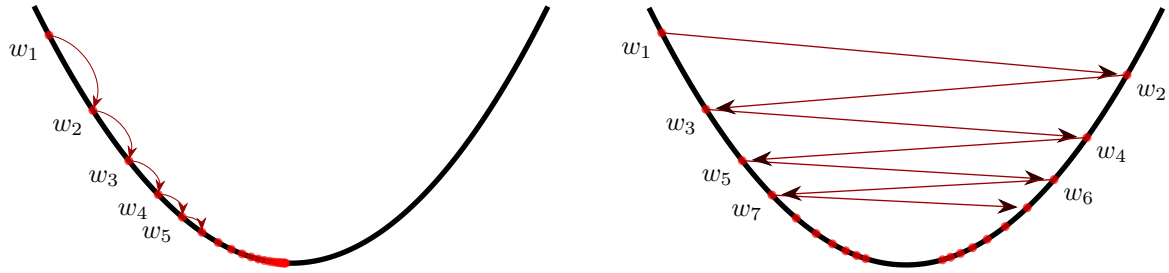

Рис. 2.4: Случаи маленького и большого шага

Если длина шага слишком мала, то метод будет неспеша, но верно шагать в сторону минимума. Если же взять размер шага очень большим, появляется риск, что метод будет перепрыгивать через минимум. Более того, есть риск того, что градиентный спуск не сойдется.

Имеет смысл использовать переменный размер шага: сначала, когда точка минимума находится еще далеко, двигаться быстро, а позже, спустя некоторое количество итерации — делать более аккуратные шаги. Один из способов задать размер шага следующий:

$$\eta_t = \frac{k}{t},$$

где k — константа, которую необходимо подобрать, а t — номер шага.

2.3.4. Случай многомерной линейной регрессии

В случае многомерной линейной регрессии используется тот же самый подход — необходимо решать задачу минимизации:

$$Q(w, X) = \frac{1}{\ell}\|Xw - y\|^2 \to \min_w,$$

где $\|x\|$ — норма вектора x. Формула для вычисления градиента принимает следующий вид:

$$\nabla_w Q(w, X) = \frac{2}{\ell} X^T (Xw - y)$$

Стоит отметить, что вектор $Xw - y$, который присутствует в данном выражении, представляет собой вектор ошибок.

2.4. Стохастический градиентный спуск

В этом блоке речь пойдет о стохастическом градиентном спуске, который особенно хорошо подходит для обучения линейных моделей.

2.4.1. Недостатки обычного метода градиентного спуска

В обычном методе градиентного спуска на каждом шаге итерации следующее приближение получается из предыдущего вычитанием вектора градиента, умноженного на шаг η_t:

$$w^t = w^{t-1} - \eta_t \nabla Q(w^{t-1}, X).$$

При этом выражение для градиента в матричной форме имеет вид:

$$\nabla_w Q(w, X) = \frac{2}{\ell} X^T (Xw - y)$$

Выражение для j-ой компоненты градиента, таким образом, содержит суммирование по всем объектам обучающей выборки:

$$\frac{\partial Q}{\partial w_j} = \frac{2}{\ell} \sum_{i=1}^{\ell} x_i^j (\langle w, x_i \rangle - y_i).$$

В этом и состоит основной недостаток метода градиентного спуска — в случае большой выборки даже одна итерация метода градиентного спуска будет производиться долго.

2.4.2. Стохастический градиентный спуск

Идея стохастического градиентного спуска основана на том, что в сумме в выражении для j-компоненты градиента i-ое слагаемое указывает то, как нужно поменять вес w_j, чтобы качество увеличилось для i-го объекта выборки. Вся сумма при этом задает, как нужно изменить этот вес, чтобы повысить качество для всех объектов выборки. В стохастическом методе градиентного спуска градиент функции качества вычисляется только на одном случайно выбранном объекте обучающей выборки. Это позволяет обойти вышеупомянутый недостаток обычного градиентного спуска.

Таким образом, алгоритм стохастического градиентного спуска следующий. Сначала выбирается начальное приближение:

$$w^0 = 0$$

Далее последовательно вычисляются итерации w^t: сначала случайным образом выбирается объект x_i из обучающей выборки X и вычисляется вектор градиента функции качества на этом объекте, а следующее приближение получается из предыдущего вычитанием умноженного на шаг η_t полученного вектора:

$$w^t = w^{t-1} - \eta_t \nabla Q(w^{t-1}, \{x_i\}).$$

Итерации прекращаются при достижении определенного условия, например:

$$\|w^t - w^{t-1}\| < \varepsilon.$$

2.4.3. Сходимость стохастического градиентного спуска

Показательно посмотреть на графики сходимости градиентного спуска и стохастического градиентного спуска. В обычном градиентном спуске на каждом шаге уменьшается суммарная ошибка на всех элементах обучающей выборки. График в таком случае обычно получается монотонным.

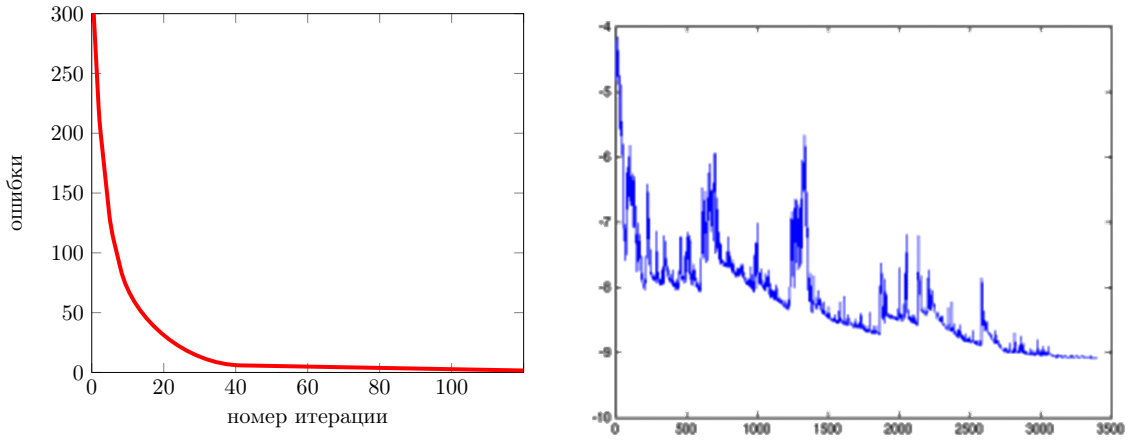

Рис. 2.5: Ошибка в зависимости от номера итерации

Напротив, в стохастическом методе весовые коэффициенты меняются таким образом, чтобы максимально уменьшить ошибку для одного случайно выбранного объекта. Это приводит к тому, что график выглядит пилообразным, то есть на каждой конкретной итерации полная ошибка может как увеличиваться, так и уменьшаться. Но в итоге с ростом номера итерации значение функции уменьшается.

2.4.4. Особенности стохастического градиентного спуска

Стохастический градиентный спуск (SGD) обладает целым рядом преймуществ. Во-первых, каждый шаг выполняется существенно быстрее шага обычного градиентного метода, а также не требуется постоянно хранить всю обучающую выборку в памяти. Это позволяет использовать для обучения выборки настолько большие, что они не помещаются в память компьютера. Стохастический градиентный спуск также можно использовать для онлайн-обучения, то есть в ситуации, когда на каждом шаге алгоритм получает только один объект и должен учесть его для коррекции модели.

2.5. Линейная классификация

2.5.1. Задача бинарной классификации

В этом разделе рассматривается задача линейной классификации, то есть применение линейных моделей к задаче классификации. Речь пойдет о самом простом виде классификации — бинарной классификации. В случае бинарной классификации множество возможных значений ответов состоит из двух элементов:

$$\mathbb{Y} = \{-1, +1\}.$$

Как уже было сказано, чтобы работать с той или иной моделью нужно:

- Выбрать функционал (функцию) ошибки, то есть задать способ определения качества работы того или иного алгоритма на обучающей выборке.
- Построить семейство алгоритмов, то есть множество алгоритмов, из которого потом будет выбираться наилучший с точки зрения определенного функционала ошибки.
- Ввести метод обучения, то есть определить способ выбора лучшего алгоритма из семейства.

2.5.2. Линейный классификатор

Ранее была рассмотрена задача линейной регрессии. В этом случае алгоритм представлял собой линейную комбинацию признаков с некоторыми весами и свободным коэффициентом.

Линейные классификаторы устроены похожим образом, но они должны возвращать бинарные значения, а следовательно требуется также брать знак от получившегося выражения:

$$a(x) = \text{sign}\left(w_0 + \sum_{j=1}^{d} w_j x^j\right).$$

Как и раньше, добавлением еще одного постоянного для всех объектов признака можно привести формулу к более однородному виду:

$$a(x) = \text{sign} \sum_{j=1}^{d+1} w_j x^j = \text{sign}\langle w, x\rangle$$

2.5.3. Геометрический смысл линейного классификатора

Выражение $\langle w, x\rangle = 0$ является уравнением некоторой плоскости в пространстве признаков.

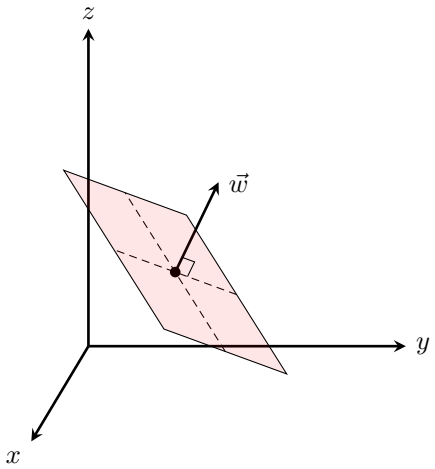

Рис. 2.6: Геометрический смысл линейного классификатора

При этом для точек по одну сторону от этой плоскости скалярное произведение $\langle w, x\rangle$ будет положительным, а с другой — отрицательным.

Таким образом, линейный классификатор проводит плоскость в пространстве признаков и относит объекты по разные стороны от нее к разным классам.

Согласно геометрическому смыслу скалярного произведения, расстояние от конкретного объекта, который имеет признаковое описание x, до гиперплоскости $\langle w, x\rangle = 0$ равно $\frac{|\langle w,x\rangle|}{\|w\|}$. С этим связано такое важное понятие в задачах линейной классификации как понятие отступа:

$$M_i = y_i\langle w, x_i\rangle.$$

Отступ является величиной, определяющей корректность ответа. Если оступ больше нуля $M_i > 0$, то классификатор дает верный ответ для i-го объекта, в ином случае — ошибается.

Причем чем дальше отступ от нуля, тем больше уверенность как в правильном ответе, так и в том, что алгоритм ошибается. Если отступ для некоторого объекта отрицательный и большой по модулю, это значит, что алгоритм неправильно описывает данные: либо этот объект является выбросом, либо алгоритм не пригоден для решения данной задачи.

2.6. Функции потерь в задачах классификации

2.6.1. Пороговая функция потерь

В случае линейной классификации естественный способ определить качество того или иного алгоритма — вычислить для объектов обучающей выборки долю неправильных ответов:

$$Q(a, x) = \frac{1}{\ell} \sum_{i=1}^{\ell} [a(x_i) \neq y_i]$$

С помощью введенного ранее понятия отступа можно переписать это выражение для случая линейной классификации в следующем виде:

$$Q(a,x) = \frac{1}{\ell}\sum_{i=1}^{\ell}\big[y_i\langle w, x_i\rangle < 0\big] = \frac{1}{\ell}\sum_{i=1}^{\ell}\big[M_i < 0\big]$$

Функция, стоящая под знаком суммы, называется функцией потерь. В данном случае это пороговая функция потерь, график которой в зависимости от отступа выглядит следующим образом:

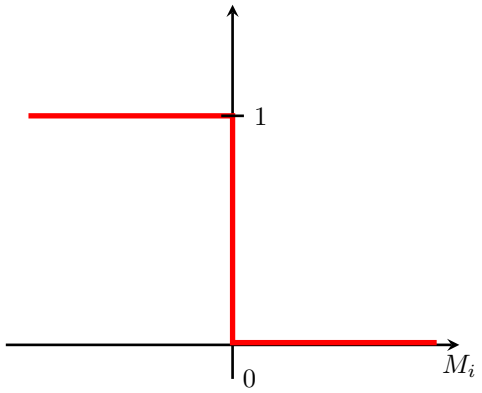

Рис. 2.7: График пороговой функции потерь

Такая функция является разрывной в точке 0, что делает невозможным применение метода градиентного спуска. Можно, конечно, использовать методы негладкой оптимизации, о которых шла речь в прошлом курсе, но они сложны в реализации.

2.6.2. Оценка функции потерь

Используя любую гладкую оценку пороговой функции:

$$\big[M_i < 0\big] \leq \tilde{L}(M_i)$$

можно построить оценку $\tilde{Q}(a,X)$ для функционала ошибки $Q(a,X)$:

$$Q(a,X) \leq \tilde{Q}(a,X) = \frac{1}{\ell}\sum_{i=1}^{\ell}\tilde{L}(M_i).$$

В этом случае минимизировать нужно будет не долю неправильных ответов, а некоторую другую функцию, которая является оценкой сверху:

$$Q(a,X) \leq \tilde{Q}(a,X) = \frac{1}{\ell}\sum_{i=1}^{\ell}\tilde{L}(M_i) \to \min_a.$$

Здесь используется предположение, что в точке минимума этой верхней оценки число ошибок также будет минимально. Строго говоря, это не всегда так.

2.6.3. Примеры оценок функции потерь

Примерами таких оценок функции потерь являются:

- Логистическая функция потерь (используется в логистической регрессии, о которой пойдет речь позже в данном курсе):
$$\tilde{L}(M) = \log_2\big(\exp(-M)\big)$$
- Экспоненциальная функция потерь:
$$\tilde{L}(M) = \exp(-M)$$

- Кусочно-линейная функция потерь (используется в методе опорных векторов):

$$\tilde{L}(M) = \max(0, 1 - M)$$

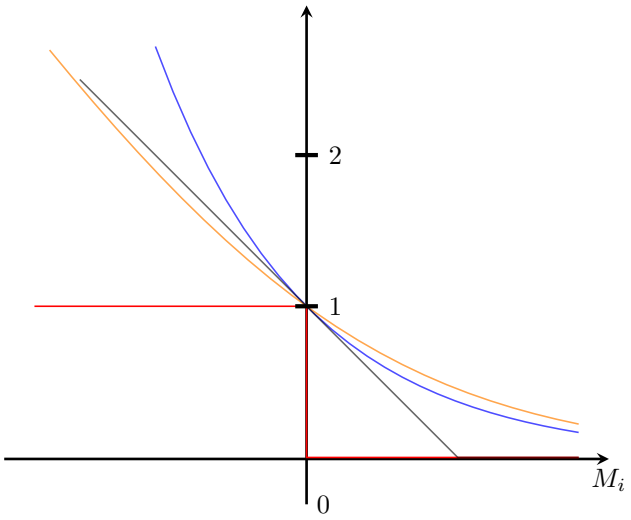

Рис. 2.8: Графики различных функций потерь: пороговая (красная линия), экспоненциальная (синяя), логистическая (оранжевая) и кусочно-линейная (серая).

2.6.4. Логистическая функция потерь

В случае логистической функции потерь функционал ошибки имеет вид:

$$\tilde{Q}(a, X) = \frac{1}{\ell} \sum_{i=1}^{\ell} \ln\big(\exp(-M_i)\big) = \frac{1}{\ell} \sum_{i=1}^{\ell} \ln\big(\exp(-y_i \langle w, x_i \rangle)\big).$$

Получившееся выражение является гладким, а, следовательно, можно использовать, например, метод градиентного спуска.

Следует обратить внимание, что в случае, если число ошибок стало равно нулю, все равно в ходе обучения алгоритма линейной классификации будут увеличиваться отступы, то есть будет увеличиваться уверенность в полученных результатах.