Линейные методы классификации и регрессии: метод стохастического градиента

Воронцов Константин Вячеславович
vokov@forecsys.ru
http://www.MachineLearning.ru/wiki?title=User:Vokov

Этот курс доступен на странице вики-ресурса
http://www.MachineLearning.ru/wiki
«Машинное обучение (курс лекций, К.В.Воронцов)»

Видеолекции: http://shad.yandex.ru/lectures

ШАД Яндекс • 9 марта 2015

Содержание

Обучение регрессии — это оптимизация

Обучающая выборка: $X^\ell = (x_i, y_i)_{i=1}^\ell, \quad x_i \in \mathbb{R}^n, \quad y_i \in \mathbb{R}$

1. Модель регрессии — *линейная*:
$$a(x, w) = \langle x, w \rangle = \sum_{j=1}^n f_j(x) w_j, \qquad w \in \mathbb{R}^n$$
2. Функция потерь — *квадратичная*:
$$\mathscr{L}(a, y) = (a - y)^2$$
3. Метод обучения — *метод наименьших квадратов*:
$$Q(w) = \sum_{i=1}^\ell \big(a(x_i, w) - y_i\big)^2 \to \min_w$$
4. Проверка по тестовой выборке $X^k = (\tilde{x}_i, \tilde{y}_i)_{i=1}^k$:
$$\bar{Q}(w) = \frac{1}{k} \sum_{i=1}^k \big(a(\tilde{x}_i, w) - \tilde{y}_i\big)^2$$

Обучение классификации — тоже оптимизация

Обучающая выборка: $X^\ell = (x_i, y_i)_{i=1}^\ell, \quad x_i \in \mathbb{R}^n, \quad y_i \in \{-1, +1\}$

1. Модель классификации — *линейная*:
$$a(x, w) = \text{sign}\langle x, w\rangle$$
2. Функция потерь — бинарная или её аппроксимация:
$$\mathscr{L}(a, y) = \big[\langle x_i, w\rangle y_i < 0\big] \leqslant \mathscr{L}\big(\langle x_i, w\rangle y_i\big)$$
3. Метод обучения — *минимизация эмпирического риска*:
$$Q(w) = \sum_{i=1}^{\ell} \big[a(x_i, w) y_i < 0\big] \leqslant \sum_{i=1}^{\ell} \mathscr{L}\big(\langle x_i, w\rangle y_i\big) \to \min_w$$
4. Проверка по тестовой выборке $X^k = (\tilde{x}_i, \tilde{y}_i)_{i=1}^k$:
$$\bar{Q}(w) = \frac{1}{k} \sum_{i=1}^{k} \big[\langle \tilde{x}_i, w\rangle \tilde{y}_i < 0\big]$$

Понятие отступа для разделяющих классификаторов

Задача классификации с двумя классами: $y_i \in \{-1, +1\}$

Разделяющий классификатор: $a(x, w) = \operatorname{sign} g(x, w)$,
$g(x, w)$ — *разделяющая (дискриминантная) функция*,
w — вектор параметров,
$g(x, w) = 0$ — разделяющая поверхность

Определение

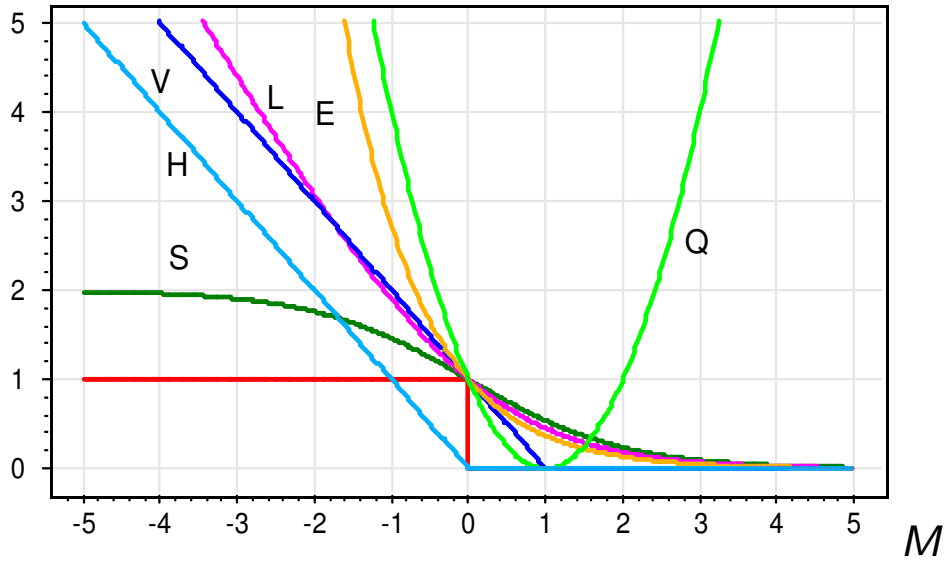
$M_i(w) = g(x_i, w)y_i$ — *отступ* (margin) объекта x_i

$M_i(w) < 0 \iff$ алгоритм $a(x, w)$ ошибается на x_i

Линейный классификатор: $a(x, w) = \operatorname{sign}\langle x, w\rangle$:
$\langle x, w\rangle = 0$ — разделяющая гиперплоскость,
$M_i(w) = \langle w, x_i\rangle y_i$ — отступ объекта x_i.

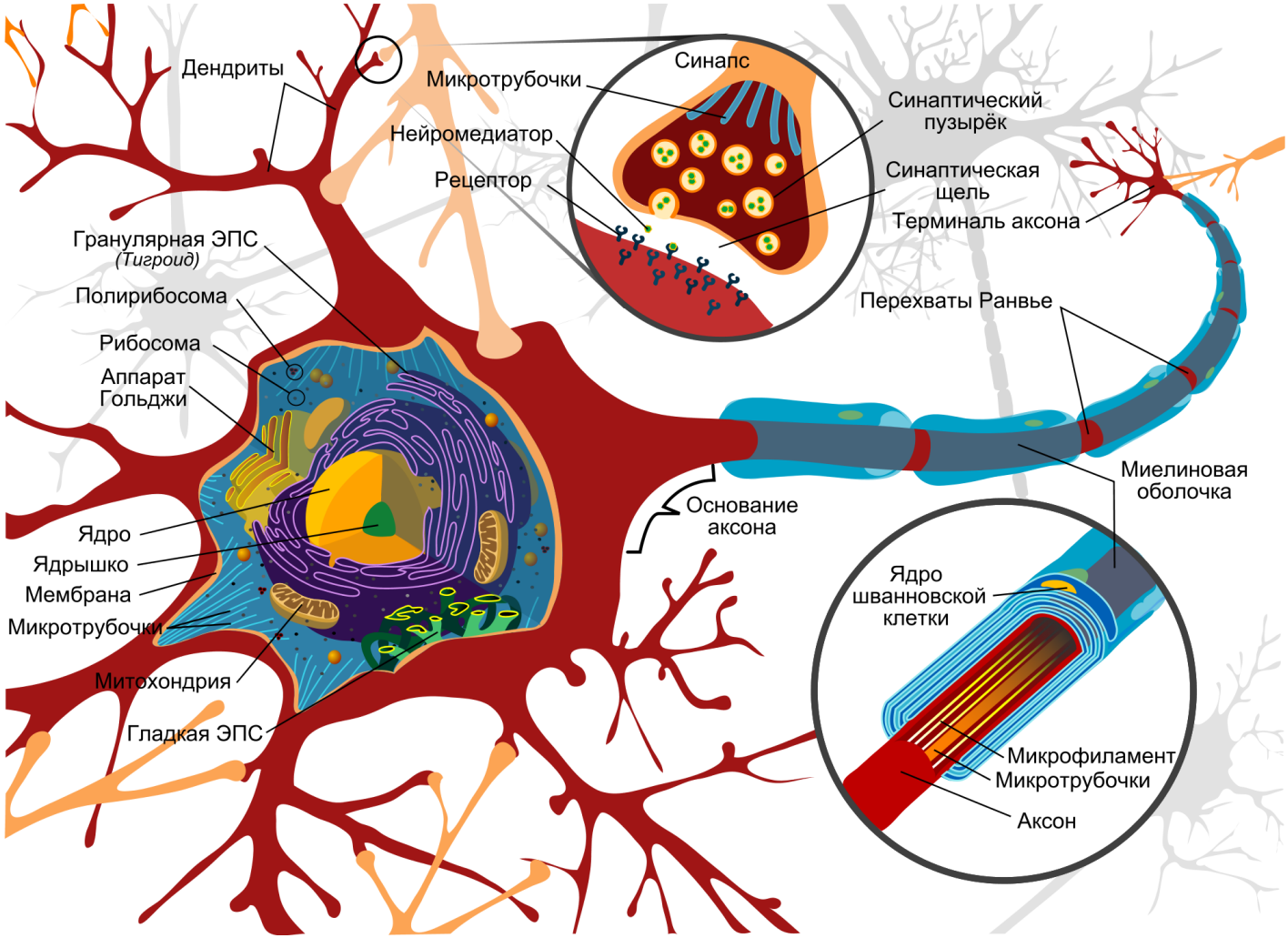
Непрерывные аппроксимации пороговой функции потерь

Часто используемые непрерывные функции потерь $\mathscr{L}(M)$:



$V(M) = (1 - M)_+$ — кусочно-линейная (SVM);
$H(M) = (-M)_+$ — кусочно-линейная (Hebb's rule);
$L(M) = \log_2(1 + e^{-M})$ — логарифмическая (LR);
$Q(M) = (1 - M)^2$ — квадратичная (FLD);
$S(M) = 2(1 + e^{M})^{-1}$ — сигмоидная (ANN);
$E(M) = e^{-M}$ — экспоненциальная (AdaBoost);
$[M < 0]$ — пороговая функция потерь.

Линейный классификатор — математическая модель нейрона

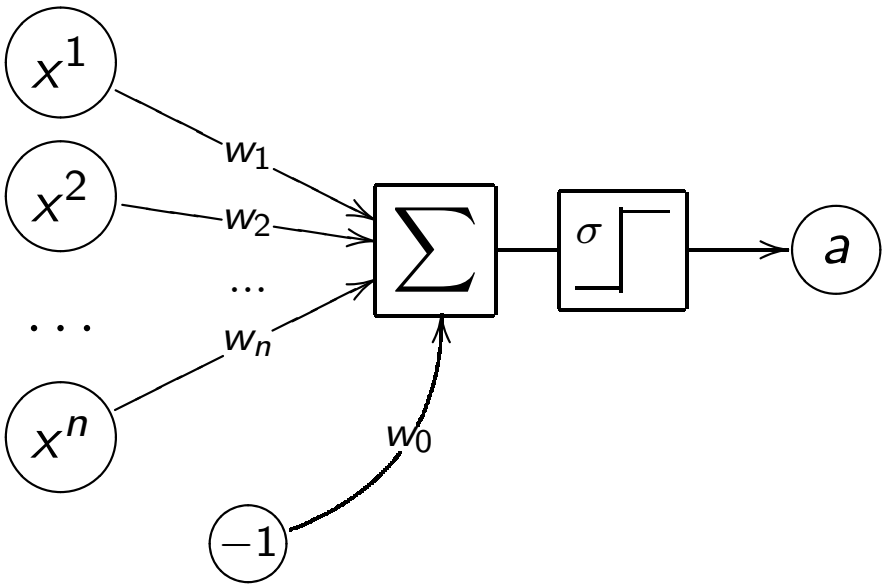


Линейный классификатор — математическая модель нейрона

Линейная модель нейрона МакКаллока-Питтса [1943]:

$$a(x, w) = \sigma\big(\langle w, x\rangle\big) = \sigma\left(\sum_{j=1}^{n} w_j f_j(x) - w_0\right),$$

$\sigma(z)$ — функция активации (например, sign),
w_0 — порог активации,
$w, x \in \mathbb{R}^{n+1}$, если ввести константный признак $f_0(x) \equiv -1$



Персептрон Розенблатта [1957]

Нейрофизиология: эффект тренировки синаптической связи: при синхронном возбуждении двух связанных нервных клеток синаптическая связь между ними усиливается.

Задача классификации: $x_i \in \{0,1\}^n$, $y_i \in \{0,1\}$,

$$a(x, w) = \big[\langle w, x\rangle > 0\big].$$

Формализация нейрофизиологического принципа обучения:

- $a(x_i, w) = y_i \implies w$ менять не нужно;
- $a(x_i, w) = 0,\ y_i = 1 \implies w_j := w_j + h \cdot f_j(x_i)$;
- $a(x_i, w) = 1,\ y_i = 0 \implies w_j := w_j - h \cdot f_j(x_i)$;

Объединим это в одну рекуррентную формулу коррекции весов:

$$w := w - h\big(a(x_i, w) - y_i\big)x_i$$

Градиентный метод численной минимизации

Минимизация эмпирического риска:

$$Q(w) = \sum_{i=1}^{\ell} \mathscr{L}\big(g(w, x_i), y_i\big) = \sum_{i=1}^{\ell} \mathscr{L}_i(w) \to \min_w .$$

Численная минимизация методом *градиентного спуска*:

$w^{(0)} :=$ начальное приближение;

$$w^{(t+1)} := w^{(t)} - h \cdot \nabla Q(w^{(t)}), \qquad \nabla Q(w) = \left(\frac{\partial Q(w)}{\partial w_j}\right)_{j=0}^{n},$$

где h — *градиентный шаг*, называемый также *темпом обучения*.

$$w^{(t+1)} := w^{(t)} - h \sum_{i=1}^{\ell} \nabla \mathscr{L}_i\big(w^{(t)}\big).$$

Идея ускорения сходимости:
брать (x_i, y_i) по одному и сразу обновлять вектор весов.

Алгоритм SG (Stochastic Gradient)

Вход: выборка X^ℓ, темп обучения h, темп забывания λ
Выход: вектор весов w

инициализировать веса w_j, $j = 0, \ldots, n$;
инициализировать оценку функционала: $\bar{Q} := \frac{1}{\ell} \sum_{i=1}^{\ell} \mathscr{L}_i(w)$;
повторять
  выбрать объект x_i из X^ℓ случайным образом;
  вычислить потерю: $\varepsilon_i := \mathscr{L}_i(w)$;
  сделать градиентный шаг: $w := w - h\nabla\mathscr{L}_i(w)$;
  оценить функционал: $\bar{Q} := (1-\lambda)\bar{Q} + \lambda\varepsilon_i$;
пока значение $\bar{Q}$ и/или веса w не сойдутся;

Robbins, H., Monro S. A stochastic approximation method // Annals of Mathematical Statistics, 1951, 22 (3), p. 400–407.

Откуда взялась такая оценка функционала?

Проблема: после каждого шаг w по одному объекту x_i, не хотелось бы оценивать Q по всей выборке $x_1, \ldots, x_\ell$.

Решение: использовать рекуррентную формулу.

Среднее арифметическое $\bar{Q}_m = \frac{1}{m}\sum_{i=1}^{m} \varepsilon_i$:

$$\bar{Q}_m = (1 - \tfrac{1}{m})\bar{Q}_{m-1} + \tfrac{1}{m}\varepsilon_m.$$

Экспоненциальное скользящее среднее

$$\bar{Q}_m = \lambda\varepsilon_m + \lambda(1-\lambda)\varepsilon_{m-1} + \lambda(1-\lambda)^2\varepsilon_{m-2} + \lambda(1-\lambda)^3\varepsilon_{m-3} + \ldots$$

$$\bar{Q}_m := (1 - \lambda)\bar{Q}_{m-1} + \lambda\varepsilon_m.$$

Чем больше λ, тем быстрее забывается предыстория ряда.
Параметр λ называется *темпом забывания*.

Алгоритм SAG (Stochastic Average Gradient)

Вход: выборка X^ℓ, темп обучения h, темп забывания λ
Выход: вектор весов w

инициализировать веса w_j, $j = 0, \ldots, n$;
инициализировать градиенты: $G_i := \nabla \mathscr{L}_i(w)$, $i = 1, \ldots, \ell$;
инициализировать оценку функционала: $\bar{Q} := \frac{1}{\ell} \sum_{i=1}^{\ell} \mathscr{L}_i(w)$;
повторять
    выбрать объект x_i из X^ℓ случайным образом;
    вычислить потерю: $\varepsilon_i := \mathscr{L}_i(w)$;
    вычислить градиент: $G_i := \nabla \mathscr{L}_i(w)$;
    сделать градиентный шаг: $w := w - h \sum_{i=1}^{\ell} G_i$;
    оценить функционал: $\bar{Q} := (1 - \lambda)\bar{Q} + \lambda \ell \varepsilon_i$;
пока значение $\bar{Q}$ и/или веса w не сойдутся;

Schmidt M., Le Roux N., Bach F. Minimizing finite sums with the stochastic average gradient // arXiv.org, 2013.

Частный случай №1: дельта-правило ADALINE

Задача регрессии: $x_i \in \mathbb{R}^{n+1}$, $y_i \in \mathbb{R}$.

Адаптивный линейный элемент ADALINE [Видроу, Хофф 1960]:

$$a(x, w) = \langle w, x \rangle, \qquad \mathscr{L}_i(w) = (\langle w, x_i \rangle - y_i)^2.$$

Градиентный шаг SG — дельта-правило (delta-rule):

$$w := w - h\underbrace{\big(\langle w, x_i \rangle - y_i\big)}_{\Delta_i} x_i,$$

Δ_i — ошибка алгоритма $a(x, w)$ на объекте x_i.

Формально совпадает с правилом персептрона Розенблатта!

Частный случай №2: правило Хэбба и персептрон Розенблатта

Задача классификации: $x_i \in \mathbb{R}^{n+1}$, $y_i \in \{-1, +1\}$,

$$a(x, w) = \text{sign}\langle w, x\rangle, \qquad \mathscr{L}_i(w) = \big(-\langle w, x_i\rangle y_i\big)_+.$$

Градиентный шаг SG — правило Хэбба [1949]:

$$\textbf{если } \langle w, x_i\rangle y_i < 0 \textbf{ то } w := w + h x_i y_i,$$

То же самое для случая $y_i \in \{0, 1\}$,

$$a(x, w) = \big[\langle w, x\rangle > 0\big], \qquad \mathscr{L}_i(w) = \big(a(x_i, w) - y_i\big)\langle w, x_i\rangle,$$

Градиентный шаг SG — персептрон Розенблатта [1957]:

$$w := w - h\big(a(x_i, w) - y_i\big)x_i.$$

Обоснование Алгоритма SG с правилом Хэбба

Задача классификации: $x_i \in \mathbb{R}^{n+1}$, $y_i \in \{-1, +1\}$.

Теорема (Новиков, 1962)

Пусть выборка X^ℓ линейно разделима:

$$\exists \tilde{w}, \ \exists \delta > 0\colon \langle \tilde{w}, x_i \rangle y_i > \delta \ \text{ для всех } \ i = 1, \ldots, \ell.$$

Тогда Алгоритм SG с правилом Хэбба находит вектор весов w,

- разделяющий обучающую выборку без ошибок;
- при любом начальном положении $w^{(0)}$;
- при любом темпе обучения $h > 0$;
- независимо от порядка предъявления объектов x_i;
- за конечное число исправлений вектора w;
- если $w^{(0)} = 0$, то число исправлений $t_{\max} \leqslant \frac{1}{\delta^2} \max \|x_i\|^2$.

Доказательство теоремы Новикова

Рассмотрим $\cos(\widehat{\tilde{w}, w^t}) = \dfrac{\langle \tilde{w}, w^t \rangle}{\|w^t\|}$ после t-го исправления w^t, при $\|\tilde{w}\| = 1$.

При t-м исправлении $\langle x_i, w^{t-1} \rangle y_i < 0$. В силу линейной разделимости

$$\langle \tilde{w}, w^t \rangle = \langle \tilde{w}, w^{t-1} \rangle + h\langle \tilde{w}, x_i \rangle y_i > \langle \tilde{w}, w^{t-1} \rangle + h\delta > \langle \tilde{w}, w^0 \rangle + th\delta.$$

В силу ограниченности выборки, $\|x_i\| < D$:

$$\|w^t\|^2 = \|w^{t-1}\|^2 + h^2\|x_i\|^2 + 2h\langle w^{t-1}, x_i \rangle y_i < \|w^{t-1}\|^2 + h^2 D^2 < \|w^0\|^2 + th^2 D^2.$$

Подставим эти соотношения в выражение для косинуса:

$$\cos(\widehat{\tilde{w}, w^t}) > \frac{\langle \tilde{w}, w^0 \rangle + th\delta}{\sqrt{\|w^0\|^2 + th^2 D^2}} \to \infty \text{ при } t \to \infty.$$

$\cos \leqslant 1$, значит при некотором t не найдётся ни одного $x_i \in X^\ell$ такого, что $\langle w^t, x_i \rangle y_i < 0$, то есть выборка окажется поделенной безошибочно.

Если $w^0 = 0$, то из условия $\cos = \dfrac{\sqrt{t}\delta}{D} \leqslant 1$ находим $t_{\max} = \left(\dfrac{D}{\delta}\right)^2$.

Варианты инициализации весов

1. $w_j := 0$ для всех $j = 0, \ldots, n$;
2. небольшие случайные значения:
   $w_j := \text{random}\left(-\frac{1}{2n}, \frac{1}{2n}\right)$;
3. $w_j := \frac{\langle y, f_j \rangle}{\langle f_j, f_j \rangle}$, $f_j = \left(f_j(x_i)\right)_{i=1}^{\ell}$ — вектор значений признака.

   Упражнение: доказать, что оценка w оптимальна, если
   1) функция потерь квадратична и
   2) признаки некоррелированы, $\langle f_j, f_k \rangle = 0$, $j \neq k$.
4. $w_j := \ln \frac{\sum_i [y_i = +1] f_j(x_i)}{\sum_i [y_i = -1] f_j(x_i)}$ — для классификации, $Y = \{-1, +1\}$
5. обучение по небольшой случайной подвыборке объектов;
6. мультистарт: многократные запуски из разных случайных начальных приближений и выбор лучшего решения.

Варианты порядка предъявления объектов

Возможны варианты:

1. *перетасовка объектов* (shuffling):
   попеременно брать объекты из разных классов;
2. чаще брать те объекты, на которых была допущена бóльшая ошибка
   (чем меньше M_i, тем больше вероятность взять объект)
   (чем меньше $|M_i|$, тем больше вероятность взять объект);
3. вообще не брать «хорошие» объекты, у которых $M_i > \mu_+$
   (при этом немного ускоряется сходимость);
4. вообще не брать объекты-«выбросы», у которых $M_i < \mu_-$
   (при этом может улучшиться качество классификации);

Параметры μ_+, μ_- придётся подбирать.

Варианты выбора градиентного шага

1. сходимость гарантируется (для выпуклых функций) при
$$h_t \to 0, \quad \sum_{t=1}^{\infty} h_t = \infty, \quad \sum_{t=1}^{\infty} h_t^2 < \infty,$$
в частности можно положить $h_t = 1/t$;
2. *метод скорейшего градиентного спуска*:
$$\mathscr{L}_i\big(w - h\nabla\mathscr{L}_i(w)\big) \to \min_h,$$
позволяет найти *адаптивный шаг* h^*;

   Упражнение: доказать, что при квадратичной функции потерь $h^* = \|x_i\|^{-2}$.
3. пробные случайные шаги
   — для «выбивания» из локальных минимумов;
4. метод Левенберга-Марквардта (второго порядка)

Диагональный метод Левенберга-Марквардта

Метод Ньютона-Рафсона, $\mathscr{L}_i(w) \equiv \mathscr{L}(\langle w, x_i \rangle y_i)$:

$$w := w - h\big(\mathscr{L}_i''(w)\big)^{-1} \nabla \mathscr{L}_i(w),$$

где $\mathscr{L}_i''(w) = \left(\frac{\partial^2 \mathscr{L}_i(w)}{\partial w_j \partial w_{j'}}\right)$ — гессиан, $n \times n$-матрица

Эвристика. Считаем, что гессиан диагонален:

$$w_j := w_j - h\left(\frac{\partial^2 \mathscr{L}_i(w)}{\partial w_j^2} + \mu\right)^{-1} \frac{\partial \mathscr{L}_i(w)}{\partial w_j},$$

h — темп обучения, можно полагать $h = 1$
μ — параметр, предотвращающий обнуление знаменателя.

Отношение h/μ есть темп обучения на ровных участках функционала $\mathscr{L}_i(w)$, где вторая производная обнуляется.

SG: Достоинства и недостатки

Достоинства:

1. легко реализуется;
2. легко обобщается на любые $g(x, w)$, $\mathscr{L}(a, y)$;
3. возможно динамическое (потоковое) обучение;
4. на сверхбольших выборках можно получить неплохое решение, даже не обработав все (x_i, y_i);
5. всё чаще применяется для Big Data

Недостатки:

1. возможна расходимость или медленная сходимость;
2. застревание в локальных минимумах;
3. подбор комплекса эвристик является искусством;
4. проблема переобучения;

Проблема переобучения

Возможные причины переобучения:

1. слишком мало объектов; слишком много признаков;
2. линейная зависимость (мультиколлинеарность) признаков:
   пусть построен классификатор: $a(x, w) = \text{sign}\langle w, x\rangle$;
   мультиколлинеарность: $\exists u \in \mathbb{R}^{n+1}$: $\forall x \;\; \langle u, x\rangle \equiv 0$;
   тогда $\forall \gamma \in \mathbb{R} \quad a(x, w) = \text{sign}\langle w + \gamma u, x\rangle$

Симптоматика:

1. слишком большие веса $|w_j|$ разных знаков;
2. неустойчивость $a(x, w)$;
3. $Q(X^\ell) \ll Q(X^k)$;

Терапия:

1. регуляризация (сокращение весов, weight decay);
2. ранний останов (early stopping);

Регуляризация (сокращение весов)

Штраф за увеличение нормы вектора весов:

$$\widetilde{\mathscr{L}}_i(w) = \mathscr{L}_i(w) + \frac{\tau}{2}\|w\|^2 = \mathscr{L}_i(w) + \frac{\tau}{2}\sum_{j=1}^{n} w_j^2 \to \min_w .$$

Градиент:

$$\nabla\widetilde{\mathscr{L}}_i(w) = \nabla\mathscr{L}_i(w) + \tau w.$$

Модификация градиентного шага:

$$w := w(1 - h\tau) - h\nabla\mathscr{L}_i(w).$$

Подбор параметра регуляризации τ:

1. скользящий контроль;
2. стохастическая адаптация;
3. байесовский вывод второго уровня;

Функции потерь, зависящие от штрафов за ошибку

Задача классификации на два класса, $y_i \in \{-1, +1\}$.
Модель классификации: $a(x; w, w_0) = \text{sign}\big(g(x, w) - w_0\big)$.
Чем меньше w_0, тем больше x_i: $a(x_i) = +1$.

Пусть λ_y — штраф за ошибку на объекте класса y.
Функция потерь теперь зависит от штрафов:

$$\mathscr{L}(a, y) = \lambda_{y_i}\big[a(x_i; w, w_0) \neq y_i\big] = \lambda_{y_i}\big[(g(x_i, w) - w_0)y_i < 0\big].$$

Проблема

На практике штрафы $\{\lambda_y\}$ могут пересматриваться

- Нужен удобный способ выбора w_0 в зависимости от $\{\lambda_y\}$, не требующий построения w заново.
- Нужна характеристика качества модели $g(x, w)$, не зависящая от штрафов $\{\lambda_y\}$ и численности классов.

Определение ROC-кривой

ROC — «receiver operating characteristic».

- Каждая точка кривой соответствует некоторому $a(x; w, w_0)$.
- по оси X: доля *ошибочных положительных классификаций* (FPR — false positive rate):

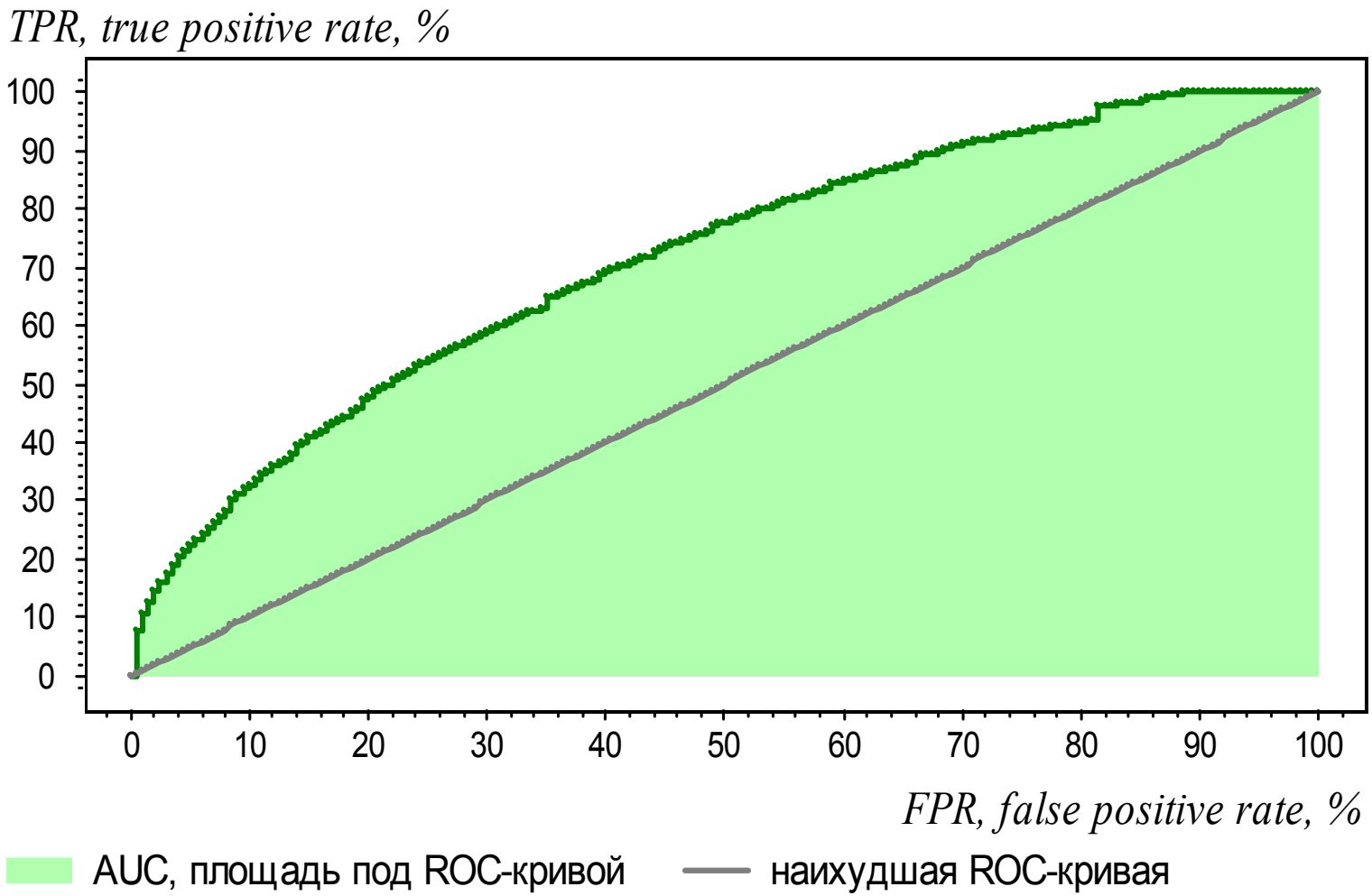
$$\text{FPR}(a, X^\ell) = \frac{\sum_{i=1}^{\ell} [y_i = -1][a(x_i; w, w_0) = +1]}{\sum_{i=1}^{\ell} [y_i = -1]};$$

  $1 - \text{FPR}(a)$ называется *специфичностью* алгоритма a.
- по оси Y: доля *правильных положительных классификаций* (TPR — true positive rate):

$$\text{TPR}(a, X^\ell) = \frac{\sum_{i=1}^{\ell} [y_i = +1][a(x_i; w, w_0) = +1]}{\sum_{i=1}^{\ell} [y_i = +1]};$$

  $\text{TPR}(a)$ называется также *чувствительностью* алгоритма a.

Пример ROC-кривой



Алгоритм эффективного построения ROC-кривой

Вход: выборка X^ℓ; дискриминантная функция $g(x, w)$;

Выход: $\{(\mathrm{FPR}_i, \mathrm{TPR}_i)\}_{i=0}^{\ell}$, AUC — площадь под ROC-кривой.

$\ell_y := \sum_{i=1}^{\ell} [y_i = y]$, для всех $y \in Y$;

упорядочить выборку X^ℓ по убыванию значений $g(x_i, w)$;

поставить первую точку в начало координат:
$(\mathrm{FPR}_0, \mathrm{TPR}_0) := (0, 0)$; $\mathrm{AUC} := 0$;

для $i := 1, \ldots, \ell$

  если $y_i = -1$ **то** сместиться на один шаг вправо:

    $\mathrm{FPR}_i := \mathrm{FPR}_{i-1} + \frac{1}{\ell_-}$; $\mathrm{TPR}_i := \mathrm{TPR}_{i-1}$;

    $\mathrm{AUC} := \mathrm{AUC} + \frac{1}{\ell_-}\mathrm{TPR}_i$;

  иначе сместиться на один шаг вверх:

    $\mathrm{FPR}_i := \mathrm{FPR}_{i-1}$; $\mathrm{TPR}_i := \mathrm{TPR}_{i-1} + \frac{1}{\ell_+}$;

Градиентная максимизация AUC

Модель: $a(x_i, w, w_0) = \text{sign}\big(g(x_i, w) - w_0\big)$.

AUC — это доля правильно упорядоченных пар (x_i, x_j):

$$\text{AUC}(w) = \frac{1}{\ell_-}\sum_{i=1}^{\ell}\big[y_i = -1\big]\text{TPR}_i =$$

$$= \frac{1}{\ell_-\ell_+}\sum_{i=1}^{\ell}\sum_{j=1}^{\ell}\big[y_i < y_j\big]\big[g(x_i, w) < g(x_j, w)\big] \to \max_w.$$

Явная максимизация аппроксимированного AUC:

$$\text{AUC}(w) \leqslant Q(w) = \sum_{i,j:\ y_i<y_j} \mathscr{L}\big(\underbrace{g(x_j, w) - g(x_i, w)}_{M_{ij}(w)}\big) \to \min_w,$$

где $\mathscr{L}(M)$ — гладкая убывающая функция отступа,
$M_{ij}(w)$ — новое понятие отступа для пар объектов.

Алгоритм SG (Stochastic Gradient) для AUC

Возьмём для простоты линейный классификатор:

$$g(x,w) = \langle x, w\rangle, \qquad M_{ij}(w) = \langle x_j - x_i, w\rangle.$$

Вход: выборка X^ℓ, темп обучения h, темп забывания λ

Выход: вектор весов w

инициализировать веса w_j, $j = 0,\ldots,n$;
инициализировать оценку: $\bar{Q} := \frac{1}{\ell_+\ell_-}\sum_{i,j:\, y_i<y_j} \mathscr{L}(M_{ij}(w))$;
повторять
  выбрать пару объектов (i,j): $y_i < y_j$, случайным образом;
  вычислить потерю: $\varepsilon_{ij} := \mathscr{L}(M_{ij}(w))$;
  сделать градиентный шаг: $w := w - h\mathscr{L}'(M_{ij}(w))(x_j - x_i)$;
  оценить функционал: $\bar{Q} := (1-\lambda)\bar{Q} + \lambda\varepsilon_{ij}$;
пока значение $\bar{Q}$ и/или веса w не сойдутся;

Резюме в конце лекции

- Метод стохастического градиента (SG, SAG) подходит для любых моделей, функций потерь и больших данных
- *Аппроксимация пороговой функции потерь* $\mathscr{L}(M)$ позволяет использовать градиентную оптимизацию и повышает качество классификации за счёт увеличения зазора между классами.
- *Регуляризация* решает проблему мультиколлинеарности и также снижает переобучение.
- *AUC* — мера качества классификации, не зависящая от соотношения штрафов и численности классов. *Явная максимизация AUC* используется в задачах классификации и ранжирования