

Линейные методы классификации

К. В. Воронцов, А. В. Зухба

`vokov@forecsys.ru`

`a__l@mail.ru`

март 2016

Содержание

- 1 **Метод стохастического градиента**
 - Минимизация эмпирического риска
 - Линейный классификатор
 - Метод стохастического градиента
- 2 **Эвристики для метода SG**
 - Инициализация весов и порядок объектов
 - Выбор величины градиентного шага
 - Проблема переобучения, метод сокращения весов
- 3 **Балансировка ошибок и ROC-кривая**
 - Постановка задачи
 - Определение ROC-кривой
 - Эффективное построение ROC-кривой
 - Градиентная максимизация AUC

Обучение регрессии — это оптимизация

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

- ❶ Модель регрессии — *линейная*:

$$a(x, w) = \langle x, w \rangle = \sum_{j=1}^n f_j(x) w_j, \quad w \in \mathbb{R}^n$$

- ❷ Функция потерь — *квадратичная*:

$$\mathcal{L}(a, y) = (a - y)^2$$

- ❸ Метод обучения — *метод наименьших квадратов*:

$$Q(w) = \sum_{i=1}^{\ell} (a(x_i, w) - y_i)^2 \rightarrow \min_w$$

- ❹ Проверка по тестовой выборке $X^k = (\tilde{x}_i, \tilde{y}_i)_{i=1}^k$:

$$\bar{Q}(w) = \frac{1}{k} \sum_{i=1}^k (a(\tilde{x}_i, w) - \tilde{y}_i)^2$$

Обучение классификации — тоже оптимизация

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

- ❶ Модель классификации — *линейная*:

$$a(x, w) = \text{sign}\langle x, w \rangle$$

- ❷ Функция потерь — бинарная или *её аппроксимация*:

$$\mathcal{L}(a, y) = [\langle x_i, w \rangle y_i < 0] \leq \mathcal{L}(\langle x_i, w \rangle y_i)$$

- ❸ Метод обучения — *минимизация эмпирического риска*:

$$Q(w) = \sum_{i=1}^{\ell} [a(x_i, w) y_i < 0] \leq \sum_{i=1}^{\ell} \mathcal{L}(\langle x_i, w \rangle y_i) \rightarrow \min_w$$

- ❹ Проверка по тестовой выборке $X^k = (\tilde{x}_i, \tilde{y}_i)_{i=1}^k$:

$$\bar{Q}(w) = \frac{1}{k} \sum_{i=1}^k [\langle \tilde{x}_i, w \rangle \tilde{y}_i < 0]$$

Понятие отступа для разделяющих классификаторов

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

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

Определение

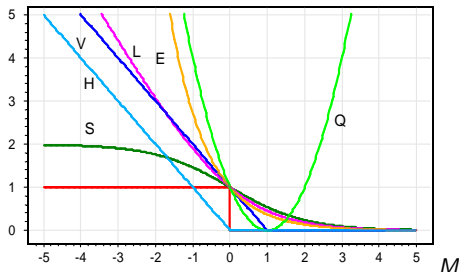
$M_i(w) = g(x_i, w)y_i$ — отступ (margin) объекта x_i

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

Линейный классификатор: $a(x, w) = \text{sign} \langle x, w \rangle$:
 $\langle x, w \rangle = 0$ — 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 $\mathcal{L}(M)$:



$$V(M) = (1 - M)_+$$

— 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);

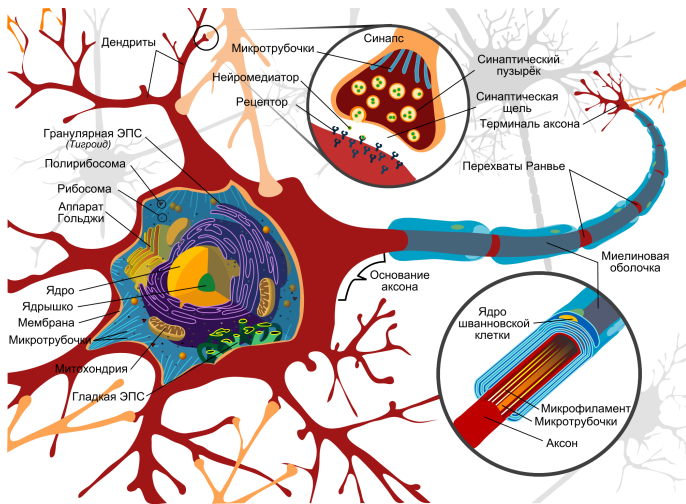
$$E(M) = e^{-M}$$

— экспоненциальная (AdaBoost);

$$[M < 0]$$

— пороговая функция потерь.

Линейный классификатор — математическая модель нейрона

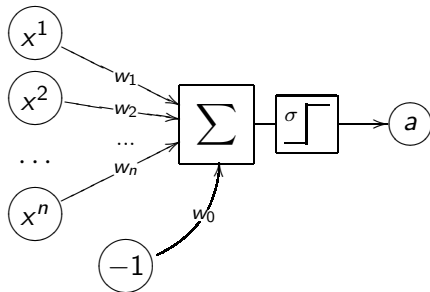


Математическая модель нейрона

Линейная модель нейрона МакКаллока-Питтса [1943]:

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

где $\sigma(s)$ — функция активации (в частности, sign).



Градиентный метод численной минимизации

Минимизация аппроксимированного эмпирического риска:

$$Q(w; X^\ell) = \sum_{i=1}^{\ell} \mathcal{L}(\langle w, x_i \rangle y_i) \rightarrow \min_w.$$

Численная минимизация методом *градиентного спуска*:

$w^{(0)}$:= начальное приближение;

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

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

$$w^{(t+1)} := w^{(t)} - \eta \sum_{i=1}^{\ell} \mathcal{L}'(\langle w^{(t)}, x_i \rangle y_i) x_i y_i.$$

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 ;

- 1 инициализировать веса w_j , $j = 0, \dots, n$;
- 2 инициализировать оценку функционала: $\bar{Q} := \frac{1}{\ell} \sum_{i=1}^{\ell} \mathcal{L}_i(w)$;
- 3 **повторять**
 - 4 | выбрать объект x_i из X^ℓ случайным образом;
 - 5 | вычислить потерю: $\varepsilon_i := \mathcal{L}_i(w)$;
 - 6 | сделать градиентный шаг: $w := w - h \nabla \mathcal{L}_i(w)$;
 - 7 | оценить функционал: $\bar{Q} := (1 - \lambda) \bar{Q} + \lambda \varepsilon_i$;
- 8 **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 шага w по одному объекту x_i , не хотелось бы оценивать Q по всей выборке x_1, \dots, x_ℓ .

Решение: использовать рекуррентную формулу.

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

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

Экспоненциальное скользящее среднее

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

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

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 ;

- 1 инициализировать веса w_j , $j = 0, \dots, n$;
- 2 инициализировать градиенты: $G_i := \nabla \mathcal{L}_i(w)$, $i = 1, \dots, \ell$;
- 3 инициализировать оценку функционала: $\bar{Q} := \frac{1}{\ell} \sum_{i=1}^{\ell} \mathcal{L}_i(w)$;
- 4 **повторять**
 - 5 | выбрать объект x_i из X^ℓ случайным образом;
 - 6 | вычислить потерю: $\varepsilon_i := \mathcal{L}_i(w)$;
 - 7 | **вычислить градиент:** $G_i := \nabla \mathcal{L}_i(w)$;
 - 8 | сделать градиентный шаг: $w := w - h \sum_{i=1}^{\ell} G_i$;
 - 9 | оценить функционал: $\bar{Q} := (1 - \lambda)\bar{Q} + \lambda \ell \varepsilon_i$;
- 10 **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: $X = \mathbb{R}^{n+1}$, $Y \subseteq \mathbb{R}$,

$$\mathcal{L}(a, y) = (a - y)^2.$$

Адаптивный линейный элемент ADALINE
[Видроу и Хофф, 1960]:

$$a(x, w) = \langle w, x \rangle$$

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

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

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

Частный случай №2: правило Хэбба

Задача классификации: $X = \mathbb{R}^{n+1}$, $Y = \{-1, +1\}$,

$$\mathcal{L}(a, y) = (-\langle w, x \rangle y)_+.$$

Линейный классификатор:

$$a(x, w) = \text{sign} \langle w, x \rangle.$$

Градиентный шаг — **правило Хэбба** [1949]:

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

Если $X = \{0, 1\}^n$, $Y = \{0, +1\}$, то правило Хэбба переходит в правило **перцептрона Розенблатта** [1957]:

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

Обоснование Алгоритма SG с правилом Хэбба

Задача классификации: $X = \mathbb{R}^{n+1}$, $Y = \{-1, 1\}$.

Теорема (Новиков, 1962)

Пусть выборка X^ℓ линейно разделима:

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

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 $\eta > 0$;
- независимо от порядка предъявления объектов x_i ;
- за конечное число исправлений вектора w ;
- если $w^{(0)} = 0$, то число исправлений $t_{\max} \leq \frac{1}{\delta^2} \max \|x_i\|$.

Инициализация весов

Возможны варианты:

- 1 $w_j := 0$ для всех $j = 0, \dots, n$;
- 2 небольшие случайные значения:
 $w_j := \text{random} \left(-\frac{1}{2n}, \frac{1}{2n} \right)$;
- 3 $w_j := \frac{\langle y, f_j \rangle}{\langle f_j, f_j \rangle}$, $f_j = (f_j(x_i))_{i=1}^\ell$ — вектор значений признака.
- 4 обучение по небольшой случайной подвыборке объектов;
- 5 многократные запуски из разных случайных начальных приближений и выбор лучшего решения.

Порядок предъявления объектов

Возможны варианты:

- 1 перетасовка объектов (shuffling):
попеременно брать объекты из разных классов;
- 2 чаще брать те объекты, на которых была допущена
большая 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.

Выбор величины градиентного шага

Возможны варианты:

- ❶ сходимость гарантируется (для выпуклых функций) при

$$\eta_t \rightarrow 0, \quad \sum_{t=1}^{\infty} \eta_t = \infty, \quad \sum_{t=1}^{\infty} \eta_t^2 < \infty,$$

в частности можно положить $\eta_t = 1/t$;

- ❷ *метод скорейшего градиентного спуска:*

$$Q(w - \eta \nabla Q(w)) \rightarrow \min_{\eta},$$

позволяет найти *адаптивный шаг* η^* ;

- ❸ пробные случайные шаги

— для «выбивания» из локальных минимумов;

Диагональный метод Левенберга-Марквардта

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

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

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

Эвристика. Считаем, что гессиан диагонален:

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

h — темп обучения, можно полагать $h = 1$

μ — параметр, предотвращающий обнуление знаменателя.

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

SG: Достоинства и недостатки

Достоинства:

- 1 легко реализуется;
- 2 легко обобщается на любые f , \mathcal{L} ;
- 3 возможно динамическое (потокковое) обучение;
- 4 на сверхбольших выборках не обязательно брать все x_i ;

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:

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

Симптоматика:

- ❶ слишком большие веса $\|w\|$;
- ❷ неустойчивость $a(x, w)$;
- ❸ $Q(X^\ell) \ll Q(X^k)$;

Терапия:

- ❶ сокращение весов (weight decay);
- ❷ ранний останов (early stopping);

Сокращение весов

Штраф за увеличение нормы вектора весов:

$$Q_{\tau}(w; X^{\ell}) = Q(w; X^{\ell}) + \frac{\tau}{2} \|w\|^2 \rightarrow \min_w.$$

Градиент:

$$\nabla Q_{\tau}(w) = \nabla Q(w) + \tau w.$$

Модификация градиентного шага:

$$w := w(1 - \eta\tau) - \eta \nabla Q(w).$$

Подбор параметра регуляризации τ :

- 1 скользящий контроль;
- 2 стохастическая адаптация;

Регуляризация в линейных классификаторах

- В случае мультиколлинеарности
 - решение $Q(w) \rightarrow \min_w$ неединственно или неустойчиво;
 - классификатор $a(x; w)$ неустойчив;
 - переобучение: $Q(X^\ell) \ll Q(X^k)$.
- Регуляризация — это выбор наиболее устойчивого решения
 - Гаусс — без отбора признаков;
 - Лаплас — с отбором признаков;
 - возможны комбинации (ElasticNet) и другие варианты...
- Выбор параметра регуляризации τ :
 - с помощью скользящего контроля;
 - с помощью оценок обобщающей способности;
 - стохастическая адаптация;

Зоопарк методов

- Вид разделяющей поверхности $f(x, w)$:
 - линейная $f(x, w) = \langle x, w \rangle$;
 - нелинейная;
- Вид непрерывной аппроксимации функции потерь $\mathcal{L}(M)$:
 - логарифмическая $\mathcal{L}(M) = \log(1 + e^{-M})$... LR;
 - кусочно-линейная $\mathcal{L}(M) = (1 - M)_+$... SVM;
 - экспоненциальная $\mathcal{L}(M) = e^{-M}$... AdaBoost;
- Вид регуляризатора — $\log p(w; \gamma)$:
 - равномерный ... перцептроны, LR;
 - гауссовский с равными дисперсиями ... SVM, RLR;
 - гауссовский с неравными дисперсиями ... RVM;
 - лапласовский ... приводит к отбору признаков;
- Вид численного метода оптимизации $Q(w) \rightarrow \min$.

Балансировка ошибок I и II рода. Постановка задачи

Задача классификации на два класса, $Y = \{-1, +1\}$;

λ_y — штраф за ошибку на объекте класса y ;

модель классификации: $a(x, w, w_0) = \text{sign}(f(x, w) - w_0)$.

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

Постановка задачи

- Нужен удобный способ выбора w_0 в зависимости от $\{\lambda_y\}$, не требующий построения w заново.
- Нужна характеристика качества классификатора, инвариантная относительно выбора $\{\lambda_y\}$.

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 .

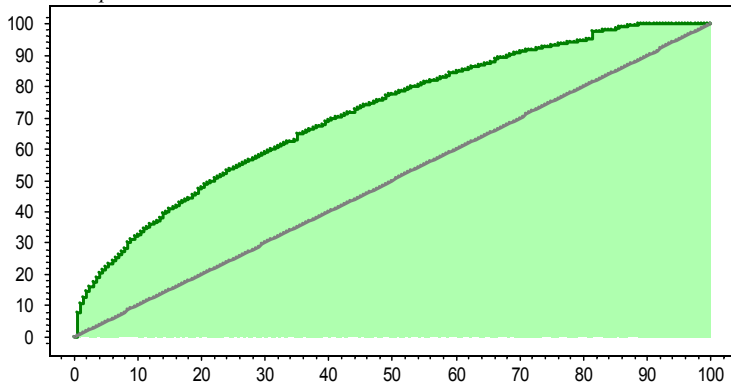
- по оси Y : доля правильных положительных классификаций (TPR — true positive rate):

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

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

Пример ROC-кривой

TPR, true positive rate, %



FPR, false positive rate, %

AUC, площадь под ROC-кривой

— наихудшая ROC-кривая

Алгоритм эффективного построения ROC-кривой

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

Выход: $\{(FPR_i, TPR_i)\}_{i=0}^\ell$, AUC — площадь под ROC-кривой.

- 1 $\ell_y := \sum_{i=1}^\ell [y_i = y]$, для всех $y \in Y$;
- 2 упорядочить выборку X^ℓ по убыванию значений $f(x_i, w)$;
- 3 $(FPR_0, TPR_0) := (0, 0)$; AUC := 0;
- 4 **для** $i := 1, \dots, \ell$
 - 5 **если** $y_i = -1$ **то**
 - 6 $FPR_i := FPR_{i-1} + \frac{1}{\ell_-}$; $TPR_i := TPR_{i-1}$;
 - 7 $AUC := AUC + \frac{1}{\ell_-} TPR_i$;
 - 8 **иначе**
 - 9 $FPR_i := FPR_{i-1}$; $TPR_i := TPR_{i-1} + \frac{1}{\ell_+}$;

Градиентная максимизация AUC

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

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

$$\begin{aligned} \text{AUC} &= \frac{1}{\ell_-} \sum_{i=1}^{\ell} [y_i = -1] \text{TPR}_i = \\ &= \frac{1}{\ell_- \ell_+} \sum_{i=1}^{\ell} \sum_{j=1}^{\ell} [y_i < y_j] [f(x_i, w) < f(x_j, w)] \rightarrow \max_w. \end{aligned}$$

Явная максимизация аппроксимированного AUC:

$$Q(w) = \sum_{i,j: y_i < y_j} \underbrace{\mathcal{L}(f(x_j, w) - f(x_i, w))}_{M_{ij}(w)} \rightarrow \min_w,$$

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

Алгоритм SG для максимизации AUC

Возьмём для простоты линейный классификатор:

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

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

Выход: вектор весов w ;

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

Резюме в конце лекции

- Методы обучения линейных классификаторов отличаются
 - видом функции потерь;
 - видом регуляризатора;
 - численным методом оптимизации.
- *Аппроксимация пороговой функции потерь* гладкой убывающей функцией отступа $\mathcal{L}(M)$ повышает качество классификации (за счёт увеличения зазора) и облегчает оптимизацию.
- *Регуляризация* решает проблему мультиколлинеарности и также снижает переобучение.
- *Максимизация AUC* не зависит от соотношения штрафов за ошибки I и II рода.