Методы машинного обучения
Линейные модели классификации

Воронцов Константин Вячеславович
www.MachineLearning.ru/wiki?title=User:Vokov
вопросы к лектору: k.vorontsov@iai.msu.ru

материалы курса:
github.com/MSU-ML-COURSE/ML-COURSE-24-25
орг.вопросы по курсу: ml.cmc@mail.ru

ВМК МГУ • 24 сентября 2024

Содержание

Оптимизационная задача обучения регрессии

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

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

Оптимизационная задача обучения бинарной классификации

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

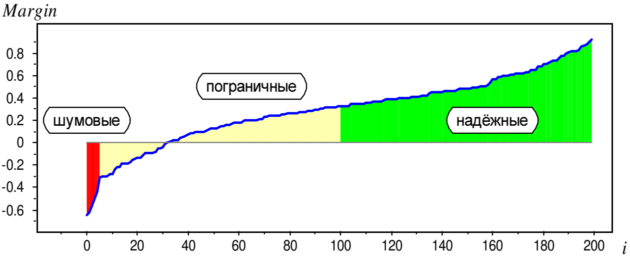
1. Модель классификации — *линейная* с параметром $w \in \mathbb{R}^n$:
$$a(x, w) = \operatorname{sign}\langle x, w\rangle = \operatorname{sign} \sum_{j=1}^n w_j f_j(x)$$
2. Функция потерь — бинарная, заменяем её аппроксимацией:
$$\mathscr{L}(w, x) = \big[a(x, w)y(x) < 0\big] = \big[\langle x, w\rangle y(x) < 0\big] \leqslant L\big(\langle x, w\rangle y(x)\big)$$
3. Метод обучения — *минимизация эмпирического риска*:
$$Q(w) = \sum_{i=1}^\ell \mathscr{L}(w, x_i) = \sum_{i=1}^\ell \big[\langle x_i, w\rangle y_i < 0\big] \leqslant \sum_{i=1}^\ell L\big(\langle x_i, w\rangle y_i\big) \to \min_w$$
4. Проверка по тестовой выборке $X^k = (\tilde{x}_i, \tilde{y}_i)_{i=1}^k$:
$$\tilde{Q}(w) = \frac{1}{k} \sum_{i=1}^k \big[\langle \tilde{x}_i, w\rangle \tilde{y}_i < 0\big]$$

Бинарный разделяющий классификатор (margin-based classifier)

Бинарный классификатор: $a(x, w) = \mathrm{sign}\, g(x, w)$, $\quad Y = \{-1, +1\}$
$g(x, w)$ — *разделяющая (дискриминантная) функция*
$x\colon g(x, w) = 0$ — разделяющая поверхность между классами

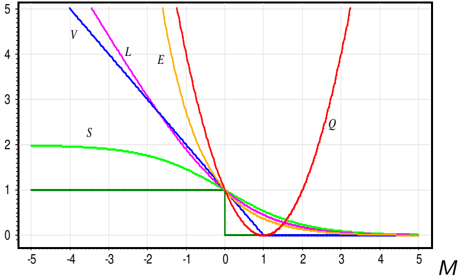
$M_i(w) = g(x_i, w) y_i$ — *отступ* (margin) объекта x_i
$M_i(w) < 0 \iff$ алгоритм $a(x, w)$ ошибается на x_i

Ранжирование объектов по возрастанию отступов $M_i(w)$:



Непрерывные аппроксимации пороговой функции потерь

Часто используемые непрерывные функции потерь $L(M)$:



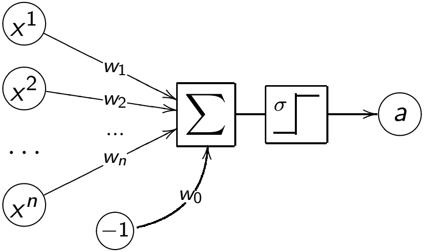
$V(M) = (1 - M)_+$	— кусочно-линейная (SVM);
$H(M) = (-M)_+$	— кусочно-линейная (Hebb's rule);
$L(M) = \log_2(1 + e^{-M})$	— логарифмическая (LR);
$Q(M) = (1 - M)^2$	— квадратичная (FLD);
$S(M) = 2(1 + e^{M})^{-1}$	— сигмоидная (ANN);
$E(M) = e^{-M}$	— экспоненциальная (AdaBoost);
$[M < 0]$	— пороговая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

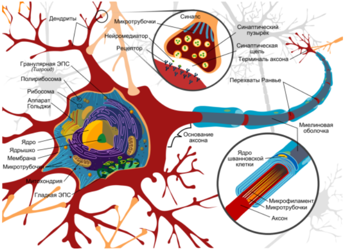
Линейный классификатор — математическая модель нейрона

Линейная модель нейрона МакКаллока-Питтса [1943]:

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

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





Задача многоклассовой классификации (multiclass classification)

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

1. Модель классификации — *линейная*, $w = \bigl(w_y \colon y \in Y\bigr)$:

$$a(x, w) = \arg\max_{y \in Y} \langle x, w_y \rangle$$

2. Функция потерь — бинарная, заменяем её аппроксимацией:

$$\mathscr{L}(w, x) = \sum_{z \neq y(x)} \bigl[\langle x, w_{y(x)} \rangle < \langle x, w_z \rangle\bigr] \leqslant \sum_{z \neq y(x)} L\bigl(\langle x, w_{y(x)} - w_z \rangle\bigr)$$

3. Метод обучения — *минимизация эмпирического риска*:

$$Q(w) = \sum_{i=1}^{\ell} \sum_{z \neq y_i} L\bigl(\langle x_i, w_{y_i} - w_z \rangle\bigr) \to \min_w$$

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

Многоклассовый разделяющий классификатор

Многоклассовый классификатор: $a(x, w) = \arg\max_{y \in Y} g_y(x, w_y)$

$g_y(x, w_y)$ — *дискриминантная функция* класса $y \in Y$

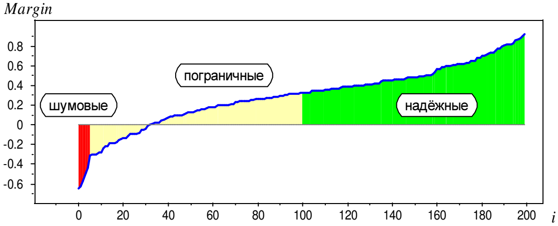
$x\colon g_y(x, w_y) = g_z(x, w_z)$ — разделяющая поверхность между y, z

$M_{iy}(w) = g_{y_i}(x_i, w_{y_i}) - g_y(x_i, w_y)$ — *отступ* объекта x_i по классу y

$M_i(w) = \min_{y \neq y_i} M_{iy}(w)$ — *отступ* (margin) объекта x_i

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

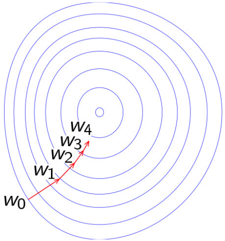
Ранжирование объектов по возрастанию отступов $M_i(w)$:



Градиентный метод оптимизации

Минимизация эмпирического риска:

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



Метод *градиентного спуска*:

$$w^{(0)} := \text{начальное приближение;}$$

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

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

$$w^{(t+1)} := w^{(t)} - h \sum_{i=1}^{\ell} \nabla \mathscr{L}\bigl(w^{(t)}, x_i\bigr)$$

Идея ускорения сходимости:

брать объекты x_i по одному и сразу обновлять вектор весов

Алгоритм SG (Stochastic Gradient)

$$Q = \sum_{i=1}^{\ell} \mathscr{L}(w, x_i) \to \min_w$$ — минимизация эмпирического риска

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

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

H. Robbins, S. Monro A stochastic approximation method. 1951.

Откуда взялась рекуррентная оценка функционала?

Проблема: вычисление оценки Q по всей выборке $x_1, \dots, x_\ell$ намного дольше градиентного шага по одному объекту x_i.

Решение: использовать приближённую рекуррентную формулу.

Среднее арифметическое:

$$\bar{Q}_m = \frac{1}{m}\varepsilon_m + \frac{1}{m}\varepsilon_{m-1} + \frac{1}{m}\varepsilon_{m-2} + \dots$$

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

Экспоненциальное скользящее среднее:

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

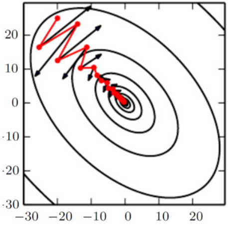
$$\bar{Q}_m = \lambda\varepsilon_m + (1-\lambda)\bar{Q}_{m-1}$$

Параметр λ (порядка $\frac{1}{m}$) — *темп забывания* предыстории ряда.

Метод накопления инерции (momentum)

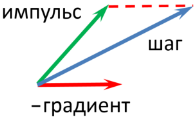
Momentum — экспоненциальное скользящее среднее градиента по последним $\approx \frac{1}{1-\gamma}$ итерациям [Б.Т.Поляк, 1964]:

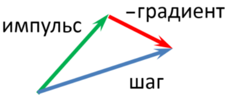
$$
\begin{aligned}
v &:= {\color{red}\gamma} v + {\color{red}(1-\gamma)} \nabla \mathscr{L}(w, x_i) \\
w &:= w - hv
\end{aligned}
$$



NAG (Nesterov's accelerated gradient) — стохастический градиент с инерцией [Ю.Е.Нестеров, 1983]:

$$
\begin{aligned}
v &:= \gamma v + (1-\gamma) \nabla \mathscr{L}(w {\color{red}\,-\, h\gamma v}, x_i) \\
w &:= w - hv
\end{aligned}
$$





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. $w_j := \frac{\langle y, f_j \rangle}{\langle f_j, f_j \rangle}$, $\;f_j = \bigl(f_j(x_i)\bigr)_{i=1}^{\ell}$ — вектор значений признака.

   Эта оценка w оптимальна, если
   1) функция потерь квадратична и
   2) признаки некоррелированы, $\langle f_j, f_k \rangle = 0,\; j \neq k$.
4. обучение по небольшой случайной подвыборке объектов;
5. 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 объекты, на которых ошибка больше:
чем меньше M_i, тем больше вероятность взять объект;
3. чаще брать объекты, на которых уверенность меньше:
чем меньше $|M_i|$, тем больше вероятность взять объект;
4. вообще не брать «хорошие» объекты, у которых $M_i > \mu_+$
(при этом немного ускоряется сходимость);
5. 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* основан на поиске оптимального *адаптивного шага* h^*:
$$\mathscr{L}\bigl(w - h\nabla\mathscr{L}(w, x_i), x_i\bigr) \to \min_h$$
При квадратичной функции потерь $h^* = \|x_i\|^{-2}$
3. пробные случайные шаги для «выбивания» итерационного процесса из локальных экстремумов
4. метод Левенберга-Марквардта (сходимость второго порядка)

Диагональный метод Левенберга-Марквардта

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

$$w := w - h\bigl(\mathscr{L}''(w, x_i)\bigr)^{-1} \nabla \mathscr{L}(w, x_i),$$

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

Эвристика. Считаем, что гессиан диагонален:

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

$h > 0$ — темп обучения, можно полагать $h = 1$,
$\mu > 0$ — параметр, предотвращающий обнуление знаменателя.

Вблизи минимума сходимость второго порядка с темпом h
Вдали от минимума сходимость первого порядка с темпом $\frac{h}{\mu}$

Мультиколлинеарность и переобучение в линейных моделях

Причины — те же, что и в линейных моделях регрессии:

- объектов меньше, чем признаков, либо
- признаки линейно зависимы (мультиколлинеарны): если $\exists u \in \mathbb{R}^n$: $\forall x_i \in X^\ell$ $\langle u, x_i \rangle = 0$, то решение не единственно и не устойчиво: $\forall \gamma \in \mathbb{R}$ $\langle w + \gamma u, x_i \rangle = \langle w, x_i \rangle$

Проявления мультиколлинеарности:

- слишком большие веса $|w_j|$ разных знаков
- вес w_j не интерпретируется как важность признака f_j
- переобучение: $Q(w^*, X^\ell) \ll Q(w^*, X^k)$

Способы устранения мультиколлинеарности и переобучения:

- регуляризация: $\|w\| \to \min$;
- отбор признаков: $f_1, \dots, f_n \to f_{j_1}, \dots, f_{j_m}$, $m \ll n$.
- преобразование признаков: $f_1, \dots, f_n \to g_1, \dots, g_m$, $m \ll n$;

L_2-регуляризация (сокращение весов, weight decay)

Штраф за увеличение нормы вектора весов:

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

Градиент:

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

Модификация градиентного шага:

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

Методы подбора коэффициента регуляризации τ:

- hold-out или скользящий контроль
- стохастическая адаптация

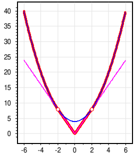
Негладкие регуляризаторы для отбора и группировки признаков

Общий вид регуляризаторов (μ — параметр селективности):

$$\sum_{i=1}^{\ell} \mathscr{L}(w, x_i) + \sum_{j=1}^{n} R_\mu(\alpha_j) \;\to\; \min_{\alpha}.$$

Регуляризаторы с эффектами отбора и группировки признаков:

LASSO (L_1): $R_\mu(\alpha) = \mu|\alpha|$



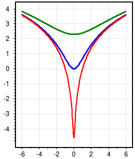
Elastic Net: $R_\mu(\alpha) = \mu|\alpha| + \tau\alpha^2$

Support Feature Machine (SFM):

$$R_\mu(\alpha) = \begin{cases} 2\mu|\alpha|, & |\alpha| \leqslant \mu; \\ \mu^2 + \alpha^2, & |\alpha| \geqslant \mu; \end{cases}$$

Relevance Feature Machine (RFM):

$$R_\mu(\alpha) = \ln\bigl(\mu\alpha^2 + 1\bigr)$$



Резюме в конце лекции

- Метод стохастического градиента (SG)
  — подходит для любых моделей и функций потерь
  — подходит для обучения по большим данным
- *Аппроксимация пороговой функции потерь* $L(M)$ позволяет использовать градиентную оптимизацию
- Функции $L(M)$, штрафующие за приближение к границе классов, увеличивают зазор между классами, благодаря чему повышается надёжность классификации
- *Регуляризация* снижает переобучение, возникающее в линейных моделях из-за мультиколлинеарности
- *Недостаток*: подбор эвристик является искусством (не забыть про сходимость, застревание, переобучение,...)