Теория и практика машинного обучения
• Лекция 3 •
Линейные классификаторы и бустинг

Воронцов Константин Вячеславович
МФТИ • МГУ • ВШЭ • ВЦ РАН • Яндекс • FORECSYS

Комбинаторика и алгоритмы
для школьников

• Летняя школа — 2015 •
21 августа 2015

Содержание

Восстановление зависимости по эмпирическим данным

Задача восстановления зависимости $y = y^*(x)$
по точкам *обучающей выборки* (x_i, y_i), $i = 1, \dots, \ell$.

Дано: векторы $x_i = (x_i^1, \dots, x_i^n)$ — объекты обучающей выборки, $y_i = y^*(x_i)$ — правильные ответы, $i = 1, \dots, \ell$:

$$\begin{pmatrix} x_1^1 & \dots & x_1^n \\ \dots & \dots & \dots \\ x_\ell^1 & \dots & x_\ell^n \end{pmatrix} \xrightarrow{y^*} \begin{pmatrix} y_1 \\ \dots \\ y_\ell \end{pmatrix}$$

Найти: функцию $a(x)$, способную давать правильные ответы на *тестовых объектах* $\tilde{x}_i = (\tilde{x}_i^1, \dots, \tilde{x}_i^n)$, $i = 1, \dots, k$:

$$\begin{pmatrix} \tilde{x}_1^1 & \dots & \tilde{x}_1^n \\ \dots & \dots & \dots \\ \tilde{x}_k^1 & \dots & \tilde{x}_k^n \end{pmatrix} \xrightarrow{a?} \begin{pmatrix} a(\tilde{x}_1) \\ \dots \\ a(\tilde{x}_k) \end{pmatrix}$$

Восстановление регрессии — это оптимизация

Задача восстановления регрессионной зависимости, $y_i \in \mathbb{R}$

1. Выбираем *модель регрессии*, например, линейную:
$$a(x, w) = \langle x, w \rangle = \sum_{j=1}^{n} x^j w_j, \qquad x, w \in \mathbb{R}^n$$
2. Выбираем функцию потерь, например, квадратичную:
$$\mathscr{L}(a, y) = (a - y)^2$$
3. Минимизируем потери *методом наименьших квадратов*:
$$Q(w) = \frac{1}{\ell} \sum_{i=1}^{\ell} \bigl(a(x_i, w) - y_i\bigr)^2 \to \min_w$$
4. Проверяем прогностическую (обобщающую) способность:
$$\tilde{Q}(w) = \frac{1}{k} \sum_{i=1}^{k} \bigl(a(\tilde{x}_i, w) - \tilde{y}_i\bigr)^2$$

Обучение классификации — тоже оптимизация

Задача классификации, $y_i \in \{-1, +1\}$

1. Выбираем *модель классификации*, например, линейную:
$$a(x, w) = \mathrm{sign}\langle x, w\rangle$$
2. Выбираем функцию потерь, например, *бинарную*:
$$\mathscr{L}(a, y) = \bigl[a(x_i, w)y_i < 0\bigr]$$
3. Минимизируем *частоту ошибок* на обучающей выборке:
$$Q(w) = \frac{1}{\ell}\sum_{i=1}^{\ell}\bigl[a(x_i, w)y_i < 0\bigr] \to \min_w$$
4. Проверяем прогностическую (обобщающую) способность:
$$\tilde{Q}(w) = \frac{1}{k}\sum_{i=1}^{k}\bigl[a(\tilde{x}_i, w)\tilde{y}_i < 0\bigr]$$

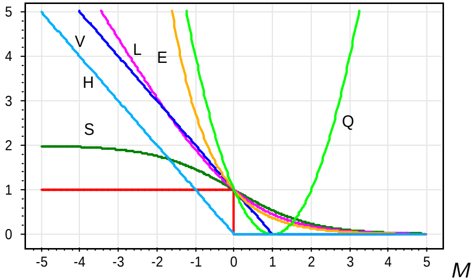
Обучение классификации — сглаживание функции потерь

Задача классификации, $y_i \in \{-1, +1\}$

1. Выбираем *модель классификации*, например, линейную:
$$a(x, w) = \mathrm{sign}\langle x, w\rangle$$
2. Мажорируем пороговую функцию потерь непрерывной:
$$\bigl[M_i < 0\bigr] \leqslant \mathscr{L}\bigl(M_i\bigr), \qquad M_i = \langle x_i, w\rangle y_i \; \text{ — отступ (margin)}$$
3. Минимизируем *сглаженную частоту ошибок*:
$$Q(w) = \frac{1}{\ell}\sum_{i=1}^{\ell} \mathscr{L}\bigl(\langle x_i, w\rangle y_i\bigr) \to \min_w$$
4. Проверяем прогностическую (обобщающую) способность:
$$\tilde{Q}(w) = \frac{1}{k}\sum_{i=1}^{k} \bigl[\langle \tilde{x}_i, w\rangle \tilde{y}_i < 0\bigr]$$

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, Logistic Regression)
$Q(M) = (1-M)^2$ — квадратичная (Fisher's Linear Discriminant)
$S(M) = 2(1+e^{M})^{-1}$ — сигмоидная (ANN, Artificial Neural Network)
$E(M) = e^{-M}$ — экспоненциальная (AdaBoost)
$[M<0]$ — пороговая функция потерь.

Общие подходы к решению оптимизационных задач

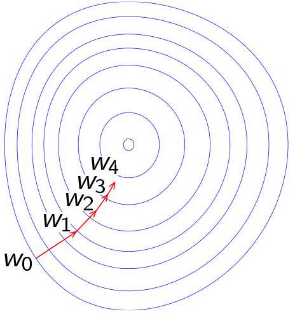
Аналитический подход (напр. метод наименьших квадратов):
Если w — точка минимума *гладкой* функции $Q(w)$, то

$$\frac{\partial Q(w)}{\partial w_j} = 0, \quad j = 1, \dots, n.$$

Это система n уравнений с n неизвестными.

Численный метод — градиентный спуск:

начальное приближение w^0, $t := 0$;
повторять
  $w_j^{t+1} := w_j^t - h^t \cdot \frac{\partial Q(w^t)}{\partial w_j}$, $j = 1, \dots, n$;
  $t := t + 1$;
пока процесс не сойдётся;



Аналитический подход. Одномерный частный случай

Два класса: $y_i \in \{-1, +1\}$, один признак: $x_i \in \{-1, 0, +1\}$.
Линейный классификатор: $a(x) = \mathrm{sign}(wx + w_0)$, $w_0 = \mathrm{const}$.

Функция потерь: $\mathscr{L}(M_i) = e^{-M_i}$, отступ $M_i = (wx_i + w_0)y_i$.

$$Q(w) = \sum_{i=1}^{\ell} \exp\bigl(-wx_iy_i - w_0y_i\bigr) \to \min_w;$$

$$Q(w) = \sum_{i=1}^{\ell} \underbrace{e^{-w_0y_i}}_{\gamma_i} \Bigl(e^{-w}\bigl[x_iy_i = 1\bigr] + e^{w}\bigl[x_iy_i = -1\bigr] + \bigl[x_i = 0\bigr]\Bigr) =$$

$$= e^{-w} \underbrace{\sum_{i=1}^{\ell} \gamma_i \bigl[x_iy_i = 1\bigr]}_{P} + e^{w} \underbrace{\sum_{i=1}^{\ell} \gamma_i \bigl[x_iy_i = -1\bigr]}_{N} + \sum_{i=1}^{\ell} \gamma_i \bigl[x_i = 0\bigr].$$

$$Q'(w) = -e^{-w}P + e^{w}N = 0 \quad \Rightarrow \quad \boxed{w = \frac{1}{2}\ln\frac{P}{N}}$$

Интерпретация полученного решения

Если $w_0 = 0$, то $\gamma_i = e^{-w_0 y_i} = 1$,
значения P и N показывают, насколько хорошо признак x предсказывает класс на объектах обучающей выборки:

$$P = \sum_{i=1}^{\ell} \gamma_i \bigl[x_i y_i = +1\bigr] \text{ — число позитивных примеров,}$$

$$N = \sum_{i=1}^{\ell} \gamma_i \bigl[x_i y_i = -1\bigr] \text{ — число негативных примеров.}$$

Весовой коэффициент $w = \frac{1}{2} \ln \frac{P}{N}$ (или $w = \frac{1}{2} \ln \frac{P+\delta}{N+\delta}$)
тем больше, чем чаще признак x_i угадывает ответ y_i.

Едем дальше: теперь признаков будет много.

Процесс жадного добавления признаков

Два класса: $y_i \in \{-1, +1\}$, n признаков: $x_i \in \{-1, 0, +1\}^n$.

Пользуясь предыдущим решением, будем жадно добавлять признаки в линейный классификатор:

$$a(x) = \mathrm{sign}\Bigl(\sum_{j=1}^{t} w_j x^j\Bigr) = \mathrm{sign}\Bigl(w_t x^t + \underbrace{\sum_{j=1}^{t-1} w_j x^j}_{w_0}\Bigr)$$

После добавления признака x^t изменяются веса объектов γ_i^t:

$$\gamma_i^{t+1} = \exp(-w_0 y_i) = \exp\Bigl(-y_i \sum_{j=1}^{t} w_j x_i^j\Bigr),$$

их удобно обновлять по рекуррентной формуле:

$$\gamma_i^{t+1} = \gamma_i^t \exp\bigl(-y_i w_t x_i^t\bigr), \qquad \gamma_i^0 = 1.$$

Какой признак добавлять следующим?

Эвристика 1: добавлять признаки независимо от предыдущих, каждый раз полагая $w_0 = 0$ и вычисляя $w_t = \frac{1}{2}\ln\frac{P_t}{N_t}$.
И это неплохо работает в задачах классификации текстов!

Эвристика 2: подставим решение e^w в функционал $Q(w)$,

$$Q(w) = e^{-w}P + e^{w}N + \sum_i \gamma_i - P - N \to \min;$$
$$e^{-w} = \sqrt{\frac{N}{P}} \quad e^{w} = \sqrt{\frac{P}{N}}$$
$$\sqrt{NP} + \sqrt{PN} - P - N = -\bigl(\sqrt{P} - \sqrt{N}\bigr)^2 \to \min;$$
$$\sqrt{P} - \sqrt{N} \to \max.$$

Таким образом, надо искать признак x^j, для которого

$$\boxed{\sqrt{P_j} - \sqrt{N_j} \to \max_{j=1\dots n}}$$

Алгоритм бустинга AdaBoost (Фройнд и Шапир, 1995)

Вход: обучающая выборка $(x_i, y_i)_{i=1}^{\ell}$; параметры T, δ;

Выход: веса признаков w_j, $j = 1 \dots n$;

1 инициализировать: $w_j := 0$, $j = 1 \dots n$; $\gamma_i := 1/\ell$, $i = 1 \dots \ell$;

2 **для всех** $t = 1 \dots T$

3 найти признак x^j с достаточно большим $\sqrt{P_j} - \sqrt{N_j}$, где

$$P_j = \sum_{i=1}^{\ell} \gamma_i \bigl[x_i^j y_i = +1\bigr];$$

$$N_j = \sum_{i=1}^{\ell} \gamma_i \bigl[x_i^j y_i = -1\bigr];$$

4 вычислить вес этого признака: $w_j := \dfrac{1}{2} \ln \dfrac{P_j + \delta}{N_j + \delta}$;

5 обновить веса объектов: $\gamma_i := \gamma_i \exp\bigl(-w_j x_i^j y_i\bigr)$, $i = 1 \dots \ell$;

6 нормировать веса объектов: $\gamma_i := \gamma_i \,/\, \sum_{s=1}^{\ell} \gamma_s$, $i = 1 \dots \ell$;

Обобщение на случай большего числа классов

Линейный классификатор на n признаках $x_i \in \{-1, 0, +1\}^n$:

$$a(x) = \arg\max_{y\in Y} \langle w_y, x\rangle = \arg\max_{y\in Y} \Bigl(w_{ty} x^t + \underbrace{\textstyle\sum_{j=1}^{t-1} w_{jy} x^j}_{w_{0y}} \Bigr)$$

Положим $\sigma_{yi} = 1$, если $y = y_i$, и $\sigma_{yi} = -1$, если $y \neq y_i$.

$$\sum_{i=1}^{\ell} \sum_{y\in Y} \exp\bigl(-\langle x_i, w_y\rangle \sigma_{yi}\bigr) = \sum_{i=1}^{\ell} \sum_{y\in Y} \underbrace{e^{-w_{0y}\sigma_{yi}}}_{\gamma_{yi}} \exp\bigl(-w_{ty} x_i^t \sigma_{yi}\bigr) =$$

$$= \sum_{i=1}^{\ell} \sum_{y\in Y} \gamma_{yi} \Bigl(e^{-w_{ty}} \bigl[x_i^t \sigma_{yi} = 1\bigr] + e^{w_{ty}} \bigl[x_i^t \sigma_{yi} = -1\bigr] + \bigl[x_i^t = 0\bigr] \Bigr) \to \min_w;$$

$$\sum_{y\in Y} e^{-w_{ty}} \underbrace{\sum_{i=1}^{\ell} \gamma_{yi} \bigl[x_i^t \sigma_{yi} = 1\bigr]}_{P_{ty}} + e^{w_{ty}} \underbrace{\sum_{i=1}^{\ell} \gamma_{yi} \bigl[x_i^t \sigma_{yi} = -1\bigr]}_{N_{ty}} \to \min_w;$$

Обобщение на случай большего числа классов

…и завершаем выкладки:

$$\frac{\partial}{\partial w_{ty}}\bigl(-e^{-w_{ty}}P_{ty} + e^{w_{ty}}N_{ty}\bigr) = 0 \quad\Rightarrow\quad \boxed{w_{ty} = \frac{1}{2}\ln\frac{P_{ty}}{N_{ty}}}$$

Значения P_{ty} и N_{ty} показывают, насколько хорошо признак x^t предсказывает класс y на объектах обучающей выборки:

$$P_{ty} = \sum_{i=1}^{\ell}\gamma_{yi}\bigl[x_i^t\sigma_{yi}=+1\bigr] \text{ — число позитивных примеров,}$$

$$N_{ty} = \sum_{i=1}^{\ell}\gamma_{yi}\bigl[x_i^t\sigma_{yi}=-1\bigr] \text{ — число негативных примеров.}$$

Весовой коэффициент $w_{ty} = \frac{1}{2}\ln\frac{P_{ty}}{N_{ty}}$ (или $w_{ty} = \frac{1}{2}\ln\frac{P_{ty}+\delta}{N_{ty}+\delta}$) тем больше, чем чаще признак x_i^t угадывает ответ y_i.

Многоклассовый AdaBoost

Вход: обучающая выборка $(x_i, y_i)_{i=1}^{\ell}$; параметры T, δ;

Выход: веса признаков w_{jy}, $j = 1 \ldots n$;

инициализировать: $w_{jy} := 0$, $j = 1 \ldots n$; $\gamma_{yi} := 1/\ell$, $i = 1 \ldots \ell$;

для всех $t = 1 \ldots T$

найти признак x^j с достаточно большим $\sum\limits_y \Bigl(\sqrt{P_{jy}} - \sqrt{N_{jy}}\Bigr)$,

$$P_{jy} = \sum_{i=1}^{\ell} \gamma_{yi} \bigl[x_i^j \sigma_{yi} = +1\bigr];$$

$$N_{jy} = \sum_{i=1}^{\ell} \gamma_{yi} \bigl[x_i^j \sigma_{yi} = -1\bigr];$$

вычислить вес этого признака: $w_{jy} := \dfrac{1}{2} \ln \dfrac{P_{jy} + \delta}{N_{jy} + \delta}$;

обновить веса объектов: $\gamma_{yi} := \gamma_{yi} \exp\bigl(-w_{jy} x_i^j \sigma_{yi}\bigr)$, $i = 1 \ldots \ell$;

нормировать веса объектов: $\gamma_{yi} := \gamma_{yi} \bigm/ \sum_{s=1}^{\ell} \gamma_{ys}$, $i = 1 \ldots \ell$;

Резюме

- Бустинг — один из лучших методов машинного обучения
- Это линейный классификатор с жадным добавлением признаков и возможностью синтеза признаков
- Альтернативный взгляд: бустинг — это сильная композиция слабых классификаторов
- Современное и наиболее успешное обобщение — *градиентый бустинг*, допускает любые функции потерь и не требует дискретности признаков
- Yandex MatrixNet — это именно градиентный бустинг

Метод стохастического градиента (SG, Stochastic Gradient)

Задача классификации: $y_i \in \{-1, +1\}$, $a(x, w) = \operatorname{sign}\langle w, x\rangle$.
Минимизация сглаженной частоты ошибок:

$$Q(w) = \sum_{i=1}^{\ell} \mathscr{L}\bigl(\langle w, x_i\rangle y_i\bigr) \to \min_w .$$

Один шаг *градиентного спуска*:

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

Идея ускорения сходимости: брать (x_i, y_i) по одному в случайном порядке и сразу обновлять вектор весов (стохастическая аппроксимация Роббинса–Монро, 1951):

$$w^{t+1} := w^t - h^t \mathscr{L}'\bigl(\langle w^t, x_i\rangle y_i\bigr)\, x_i\, y_i .$$

Алгоритм SG (Stochastic Gradient)

Вход: выборка $(x_i, y_i)_{i=1}^{\ell}$;
Выход: веса $w_1, \dots, w_n$;

1 инициализировать веса $w_j, \;\; j = 1, \dots, n$;
2 **повторять**
3 выбрать случайный объект (x_i, y_i) из обучающей выборки;
4 выбрать величину градиентного шага h;
5 выполнить градиентный шаг:
$w_j := w_j - h\mathscr{L}'\bigl(\langle w, x_i\rangle y_i\bigr)x_i^j y_i$ для всех $j = 1, \dots, n$;
6 **пока** процесс не сойдётся куда-нибудь;

Преимущества и недостатки:

⊕ можно брать какие угодно модели и функции потерь $\mathscr{L}$
⊕ хорошо работает на больших и растущих выборках
⊖ возможно застревание в локальных экстремумах

Эвристики

- Выбор начального приближения, например, так:

$$w_j^0 := \frac{\langle y, f_j \rangle}{\langle f_j, f_j \rangle} \quad \text{(из одномерной линейной регрессии)}$$

  $f_j = (x_i^j)_{i=1}^{\ell}$ — вектор значений j-го признака,
  $y = (y_i)_{i=1}^{\ell}$ — вектор ответов.
- Выбор темпа обучения (градиентного шага) h^t:
  сходимость гарантируется для выпуклых $Q(w)$ 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 $h^t = \frac{1}{t}$;
- Выбор порядка предъявления объектов:
  — случайно, но попеременно из разных классов;
  — чаще брать пограничные объекты с малым $|M_i|$;

Причины переобучения линейных моделей

1. слишком мало объектов; слишком много признаков;
2. линейная зависимость (мультиколлинеарность) признаков:

   пусть построен классификатор: $a(x, w) = \operatorname{sign}\langle x, w\rangle$;
   мультиколлинеарность: $\exists v \in \mathbb{R}^n$: $\forall x \quad \langle x, v\rangle \approx 0$;
   тогда $\forall \gamma \in \mathbb{R} \quad a(x, w) \approx \operatorname{sign}\langle x, w + \gamma v\rangle$

Последствия:

- решение неединственно и неустойчиво;
- веса w_j становятся разных знаков, увеличиваются $|w_j|$;
- $Q(w)$ на обучении много меньше, чем $\tilde{Q}(w)$ на контроле;

Спасает *регуляризация* — введение дополнительного критерия:

$$\|w\|^2 = \sum_{j=1}^{n} w_j^2 \;\to\; \min .$$

Метод сокращения весов (weight decay)

Штраф за увеличение нормы вектора весов:

$$Q_\tau(w) = Q(w) + \frac{\tau}{2}\|w\|^2 \to \min_w .$$

Градиент:

$$\frac{\partial}{\partial w_j} Q_\tau(w) = \frac{\partial}{\partial w_j} Q(w) + \tau w_j.$$

Модификация градиентного шага:

$$w_j^{t+1} := w_j^t (1 - h^t \tau) - h^t \frac{\partial}{\partial w_j} Q(w^t).$$

Параметр регуляризации τ подбирается экспериментально, по качеству на контрольной выборке.

Резюме

- Обучение — это оптимизации (в большинстве методов)
- Лучшие методы классификации основаны на сглаживании пороговой функции потерь
- Два мощных метода линейной классификации — бустинг и стохастическая аппроксимация
- Оба метода подходят для решения задач Big Data
- Оба метода подходят для решения нашего контеста ;)
- Переобучение — серьёзная проблема для линейных методов, решается с помощью регуляризации

Воронцов Константин Вячеславович
voron@forecsys.ru

www.MachineLearning.ru • Участник:Vokov

Если что-то было не понятно,
не стесняйтесь подходить и спрашивать :)