Машинное обучение. Online learning

Алексей Колесов

Белорусский государственный университет

29 ноября 2017 г.

- раньше: купили m драников, все попробовали, стали экспертами
- теперь:
  - купили драник
  - предположили вкусный он или нет
  - попробовали
  - повторили

Содержание

Бинарная классификация

- обучение происходит по раундам
- на раунде t алгоритм получает $x_t \in X$ и выдаёт метку p_t
- $y_t \in \{0, 1\}$ сообщается алгоритму
- цель алгоритма — сделать как можно меньше ошибок

Замечания об объектах

- если нет связи между прошлым и будущим обучение невозможно
- в PAC-модели предполагали, что тренировочная выборка из распределения D
- в online-learning не делаем предположения о распределениях
- какие-то предположения нужны

Гипотеза реализуемости

- гипотеза: пусть $\forall t \; y_t = h^*(x_t)$ для $h^* : X \to Y$
- $h^* \in H$, где H — известно алгоритму
- задача алгоритма — сделать как можно меньше ошибок, предполагая, что объекты и h^* могут подбираться намеренно
- $M_A(H)$ — максимальное количество ошибок, которое сделает алгоритм A на последовательности объектов, размеченных $h^* \in H$

Оценка на количество ошибок

Mistake-bound, online learnability

Пусть H — класс гипотез, A — online learning алгоритм.
Рассмотрим последовательность
$S = ((x_1, h^*(x_1)), \ldots, (x_T, h^*(x_T)),\ T \in \mathbb{N}_+,\ h^* \in H$
Обозначим $M_A(S)$ — количество ошибок, которое A допускает на S
$M_A(H) = \sup\limits_S M_A(S)$
Оценка вида $M_A(H) \leqslant B < \infty$ называется **mistake-bound**
Класс H называется online learnable, если существует A с конечной mistake-bound

- какие классы online learnable?
- у каких алгоритмов хорошие mistake-bounds?

ERM

- в PAC: если класс изучаемый, то он изучаемый с помощью ERM
- исследуем, что в online learning
- предположим, что H конечен

Consistent algorithm

Алгоритм 1 Consistent

Вход: конечный H

1: $V_1 = H$
2: **for** $t = 1, 2, \ldots$ **do**
3: получить x_t
4: выбрать любой $h \in V_t$
5: предсказать $p_t = h(x_t)$
6: получить $y_t = h^*(x_t)$
7: $V_{t+1} = \{h \in V_t : h(x_t) = y_t\}$
8: **end for**

Анализ

- когда `Consistent` делает ошибку, как минимум одна гипотеза удаляется
- $1 \leqslant |V_t| \leqslant |H| - M$
- для конечного H верно, что $M_{\texttt{Consistent}}(H) \leqslant |H| - 1$
- можно составить пример, когда $M = |H| - 1$
- можно гораздо лучше: halving algorithm

Halving algorithm

Алгоритм 2 Halving

Вход: конечный H

$V_1 = H$
for $t = 1, 2, \ldots$ **do**
  получить x_t
  предсказать $p_t = \operatorname{argmax}_{r \in \{0,1\}} |\{h \in V_t : h(x_t) = r\}|$
  получить $y_t = h^*(x_t)$
  $V_{t+1} = \{h \in V_t : h(x_t) = y_t\}$
end for

Mistake bound для halving

Mistake bound для halving

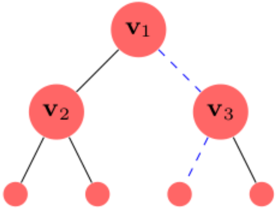
Пусть H — конечный класс гипотез. Тогда $M_{\mathtt{Halving}}(H) \leqslant \log_2(|H|)$

- оценка для `Halving` гораздо лучше, чем для `Consistent`
- в отличие от PAC, ERM-гипотеза не даёт гарантий на эффективность

Online learnability

- хотим знать, какой лучший алгоритм для фиксированного H.
- Nick Littlestone предложил характеризацию классов (Ldim), рассмотрев следующую формулировку:
  - online learning — игра алгоритма и природы
  - природа выбирает x_t, получает ответ p_t, выдаёт y_t
  - задача природы — заставить алгоритм ошибиться на первых T раундах

Стратегия природы



	h_1	h_2	h_3	h_4
$\mathbf{v}_1$	0	0	1	1
$\mathbf{v}_2$	0	1	$*$	$*$
$\mathbf{v}_3$	$*$	$*$	0	1

- при $y_t = 0$ — идём влево, иначе вправо
- номер вершины на t-м раунде: $i_t = 2^{t-1} + \sum_{j=1}^{t-1} y_j 2^{t-1-j}$

Разукрашиваемое дерево

Разукрашиваемое дерево

Последовательность объектов $v_1, \ldots, v_{2^d-1}$ называется **разукрашиваемым деревом** (shattered tree), если для любого вектора $(y_1, \ldots, y_d) \in \{0,1\}^d$ существует $h \in H$, такая что $\forall t \in [d]$ выполняется $h(v_{i_t}) = y_t$, где $i_t = 2^{t-1} + \sum_{j=1}^{t-1} y_j 2^{t-1-j}$

Размерность Littlestone-a (Ldim)

Размерность Littlestone-a (Ldim)

$\mathrm{Ldim}(H)$ — максимальное целое T, что существует разукрашиваемое дерево высоты T

Mistake-bound Ldim

Не существует алгоритма, которые имеет mistake bound меньше, чем $\mathrm{Ldim}(H)$

Примеры

- для конечного класса гипотез $\mathrm{Ldim}(H) \leqslant \log_2(|H|)$
- пусть $X = \{1, \dots, d\}$, $H = \{h_1, \dots, h_d\}$, где $h_j(x) = 1_{[x=j]}$; тогда $\mathrm{Ldim}(H) = ?$

Примеры

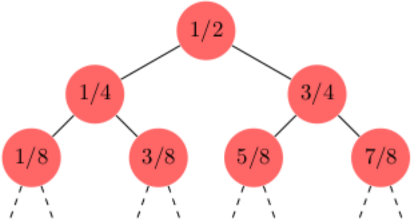
- для конечного класса гипотез $\mathrm{Ldim}(H) \leqslant \log_2(|H|)$
- пусть $X = \{1, \ldots, d\}$, $H = \{h_1, \ldots, h_d\}$, где $h_j(x) = 1_{[x=j]}$; тогда $\mathrm{Ldim}(H) = 1$

Ещё пример

Пусть $X = [0,1]$ и $H = \{x \mapsto 1_{[x<\alpha]} : \alpha \in [0,1]\}$
$\mathrm{Ldim}(H) =?$

Ещё пример

Пусть $X = [0, 1]$ и $H = \{x \mapsto 1_{[x<\alpha]} : \alpha \in [0, 1]\}$
$\mathrm{Ldim}(H) = \infty$



SOA

- $\mathrm{Ldim}(H)$ — оценка снизу на $M_A(H)$
- есть простой алгоритм, который делает ошибок не больше, чем $\mathrm{Ldim}(H)$
- идея как в `Halving`, только вместо большого класса, выбираем больший Ldim

Standard Optimal Algorithm (SOA)

Алгоритм 3 Standard Optimal Algorithm (SOA)

Вход: H

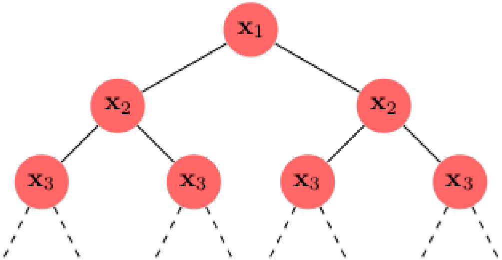
$V_1 = H$
for $t = 1, 2, \ldots$ **do**
    получить x_t
    для $r \in \{0, 1\}$ обозначим $V_t^{(r)} = \{h \in V_t : h(x_t) = r\}$
    предсказать $p_t = \operatorname*{argmax}_{r \in \{0,1\}} \mathrm{Ldim}(V_t^{(r)})$
    получить $y_t = h^*(x_t)$
    $V_{t+1} = \{h \in V_t : h(x_t) = y_t\}$
end for

SOA

- для SOA выполняется $M_{\mathtt{SOA}} \leqslant \mathrm{Ldim}(H)$
- никакой алгоритм не может иметь $M_A < \mathrm{Ldim}(H)$
- таким образом, $M_{\mathtt{SOA}} = \mathrm{Ldim}(H)$

Связь с VCdim

- VCdim(H), максимальное количество объектов, которое можно разукрасить всеми способами с помощью H
- VCdim(H) $\leqslant$ Ldim(H) (см. рисунок)
- разрыв может быть сколько угодно большой



Содержание

Нереализуемый сценарий

- как и в PAC-модели, будем просить $A(S)$ быть сравнимым с лучшей гипотезой из H
- будем оптимизировать Regret_A:

$$\mathrm{Regret}_A(h, T) = \sup_{(x_1,y_1),\ldots,(x_T,y_T)} \left[\sum_{t=1}^{T} |p_t - y_t| - \sum_{t=1}^{T} |h(x_t) - y_t| \right]$$
$$\mathrm{Regret}_A(H, T) = \sup_{h \in H} \mathrm{Regret}_A(h, T)$$

Невозможность

- существует ли A, $\text{Regret}_A(H, T) = o(T)$?
- ответ: нет, даже если $|H| = 2$:
  - природа может заставить любой алгоритм ошибиться T раз
  - лучшая из двух гипотез сделает не больше $T/2$ ошибок
  - $\text{Regret}_A(H, T) \geqslant T - T/2 \geqslant T/2$
- нужно помочь алгоритмам

Стохастичность алгоритмов

- разрешим алгоритмам быть стохастичными
- одна стохастичность не помогает
- положим, что природа может знать алгоритм, значения скрытых переменных на предыдущем раунде, но не на текущем
- будем анализировать ожидаемое число ошибок алгоритма
- алгоритм выдаёт вероятность того, что метка 1
- $p_t \in [0, 1]$, ошибка алгоритма: $\mathbb{P}[\hat{y}_t = y_t] = |p_t - y_t|$

Regret-bound

Regret-bound

Для любого класса H найдётся алгоритм для онлайн классификации, который выдаёт $p \in [0,1]$ такой что:

$$\forall h \in H,\ \sum_{t=1}^{T} |p_t - y_t| - \sum_{t=1}^{T} |h(x_t) - y_t| \leqslant \sqrt{2\min\{\log(|H|), \mathrm{Ldim}(H)\log(eT)\}\,T}$$

Никакой алгоритм не может достичь оценку лучше, чем $\Omega\left(\sqrt{\mathrm{Ldim}(H)\,T}\right)$

Взвешенное большинство

- `Weighted-majority` — алгоритм, который на каждом шаге выбирает ответ одного из экспертов
- алгоритм определяет распределение $w^{(t)}$ над d экспертами, выбирает эксперт из него
- после предсказания получает вектор $v_t \in [0, 1]^d$, насколько плох ответ i-го эксперта
- функция потерь $\langle w^{(t)}, v_t \rangle$

Weighted-majority

Алгоритм 4 Weighted-majority

Вход: d — количество экспертов, T — количество раундов

1: $\eta = \sqrt{2\log(d)/T}$
2: $\tilde{w}^{(1)} = (1, \ldots, 1)$
3: **for** $t = 1, 2, \ldots$ **do**
4: $\quad w^{(t)} = \tilde{w}^{(t)} / \sum_i \tilde{w}^{(t)}$
5: $\quad$ выбрать эксперта i из распределения $\mathbb{P}[i] = w_i^{(t)}$
6: $\quad$ получить $v_t \in [0,1]^d$
7: $\quad$ заплатить $\langle w^{(t)}, v_t \rangle$
8: $\quad \forall i,\ \tilde{w}_i^{(t+1)} = \tilde{w}_i^{(t)} e^{-\eta v_{t,i}}$
9: **end for**

Оценка Weighted-majority

Оценка Weighted-majority

Если $T > 2\log(d)$, то для `Weighted-majority` верно:

$$\sum_{t=1}^{T} \langle w^{(t)}, v_t \rangle - \min_{i \in [d]} \sum_{t=1}^{T} v_{t,i} \leqslant \sqrt{2\log(d)T}$$

- если класс конечный, то доказали Regret-bound (каждый эксперт — гипотеза)
- если бесконечный, то надо выбрать конечное число экспертов

Expert$(i_1, \ldots, i_L)$

Алгоритм 5 Expert$(i_1, \ldots, i_L)$

Вход: H, индексы $i_1 < \ldots < i_L$

```
V_1 = H
for t = 1, 2, ..., T do
    получить x_t
    для r ∈ {0, 1} обозначим V_t^(r) = {h ∈ V_t : h(x_t) = r}
    ỹ_t = argmax_r Ldim(V_t^(r))
    if t ∈ (i_1, ..., i_L) then
        ответить ŷ_t = 1 − ỹ_t
    else
        ответить ŷ_t = ỹ_t
    end if
    V_{t+1} = V_t^(ŷ_t)
end for
```

- $V_1 = H$
- **for** $t = 1, 2, \ldots, T$ **do**
  - получить x_t
  - для $r \in \{0, 1\}$ обозначим $V_t^{(r)} = \{h \in V_t : h(x_t) = r\}$
  - $\tilde{y}_t = \operatorname{argmax}_r \operatorname{Ldim}\left(V_t^{(r)}\right)$
  - **if** $t \in (i_1, \ldots, i_L)$ **then**
    - ответить $\hat{y}_t = 1 - \tilde{y}_t$
  - **else**
    - ответить $\hat{y}_t = \tilde{y}_t$
  - **end if**
  - $V_{t+1} = V_t^{(\hat{y}_t)}$
- **end for**

Анализ

- количество экспертов: $d = \sum_{L=0}^{\mathrm{Ldim}(H)} C_T^L$
- $d \leqslant (eT/\,\mathrm{Ldim}(H))^{\mathrm{Ldim}(H)}$, если $T \geqslant \mathrm{Ldim}(H) + 2$
- для любой последовательности $x_1, \ldots, x_T$ и $h \in H$ найдётся $i_1, \ldots, i_L$, такие что $\mathrm{Expert}(i_1, \ldots, i_L)$ отвечает так же, как и h

Содержание

Online выпуклая оптимизация

- ранее показывали, что выпуклые задачи машинного обучения решаемы
- в online-learning есть похожие результаты

Online выпуклая оптимизация

Алгоритм 6 Online Convex Optimization

Вход: H, домен Z, функция потерь $l : H \times Z \to \mathbb{R}$
Вход: H — выпуклый, $l(\cdot, z)$ выпукла $\forall z$

1: **for** $t = 1, 2, \ldots$ **do**
2: алгоритм выдаёт $w^{(t)} \in H$
3: природа отдаёт $z_t \in Z$
4: алгоритм платит $l(w^{(t)}, z_t)$
5: **end for**

Хотим минимизировать:

$$\text{Regret}_A(w^*, T) = \sum_{t=1}^{T} l(w^{(t)}, z_t) - \sum_{t=1}^{T} l(w^*, z_t)$$
$$\text{Regret}_A(H, T) = \sup_{w^* \in H} \text{Regret}_A(w^*, T)$$

Online Gradient Descent

Алгоритм 7 Online Gradient Descent

Вход: $\eta > 0$

1: $w^{(1)} = 0$
2: **for** $t = 1, 2, \ldots$ **do**
3: выдать $w^{(t)} \in H$
4: получить z_t и обозначить $f_t(\cdot) = l(\cdot, z_t)$
5: выбрать $v_t \in \partial f_t(w^{(t)})$
6: $w^{(t+\frac{1}{2})} = w^{(t)} - \eta v_t$
7: $w^{(t+1)} = \mathrm{argmin}_{w \in H} ||w - w^{(t+\frac{1}{2})}||$
8: **end for**

Теорема об Online Gradient Descent

Теорема об Online Gradient Descent

Для Online Gradient Descent верно:

$$\mathrm{Regret}_A(w^*, T) \leqslant \frac{||w^*||^2}{2\eta} + \frac{\eta}{2}\sum_{t=1}^{T} ||v_t||^2$$

Если кроме того f_t является ρ-липшицевой $\forall t$, то с $\eta = 1/\sqrt{T}$ получаем:

$$\mathrm{Regret}_A(w^*, T) \leqslant \tfrac{1}{2}(||w^*||^2 + \rho^2)\sqrt{T}$$

Если кроме того H является B-ограниченным, то с $\eta = \dfrac{B}{\rho\sqrt{T}}$ имеем:

$$\mathrm{Regret}_A(w^*, T) \leqslant B\rho\sqrt{T}$$

Online Perceptron

- пусть $X = \mathbb{R}^d$, $Y = \{-1, 1\}$
- на каждом раунде
  - поддерживаем $w^{(t)} \in \mathbb{R}^d$
  - получаем $x_t \in \mathbb{R}^d$
  - предсказываем $p_t = \mathrm{sign}(\langle w^{(t)}, x_t \rangle)$
  - получаем y_t и платим $1_{[p_t \neq y_t]}$
- хотим получить маленькое число ошибок
- если $d \geqslant 2$, то $\mathrm{Ldim}(H) = \infty$
- будем использовать суррогатные функции потерь!

Суррогатные функции потерь

- $l(w, (x, y)) = 1_{[y\langle w, x\rangle < 0]}$
- можем использовать разные функции для разных раундов!
- если ошиблись, то $f_t^{(-)}(w) = \max\{0, 1 - y_t\langle w, x_t\rangle\}$
- если нет, то $f_t^{(+)}(w) = 0$
- $\partial f_t^{(-)}(w^{(t)}) = -y_t x_t$
- $\partial f_t^{(+)}(w^{(t)}) = 0$

$$w^{(t+1)} = \begin{cases} w^{(t)} & \text{если } y_t\langle w^{(t)}, x_t\rangle > 0 \\ w^{(t)} + \eta y_t x_t & \text{иначе} \end{cases}$$

Online Perceptron

Алгоритм 8 Online Perceptron

$w^{(1)} = 0$
for $t = 1, 2, \ldots$ **do**
    получить x_t
    выдать $p_t = \operatorname{sign}(\langle w^{(t)}, x_t \rangle)$
    if $y_t \langle w^{(t)}, x_t \rangle \leqslant 0$ **then**
        $w^{(t+1)} = w^{(t)} + y_t x_t$
    else
        $w^{(t+1)} = w^{(t)}$
    end if
end for

Batch Perceptron

Алгоритм 9 Batch perceptron

Вход: Разделимая тренировочная выборка $S = \{(x_1, y_1), \ldots, (x_m, y_m)\}$

Выход: w, такой что $y_i\langle w, x_i\rangle > 0\ \forall i = 1, \ldots, m$

1: $w^{(1)} = (0, \ldots, 0)$
2: **for** $t = 1, 2, \ldots$ **do**
3:     **if** $\exists i$, т.ч. $y_i\langle w^{(t)}, x_i\rangle \leqslant 0$ **then**
4:         $w^{(t+1)} = w^{(t)} + y_i x_i$
5:     **else**
6:         **return** $w^{(t)}$
7:     **end if**
8: **end for**

Анализ

- по теореме об Online Gradient Descent имеем, что:
$$\sum_{t=1}^{T} f_t(w^{(t)}) - \sum_{t=1}^{T} f_t(w^*) \leqslant \frac{1}{2\eta}||w^*||_2^2 + \frac{\eta}{2}||v_t||_2^2$$
- пусть $\mathcal{M}$ — объекты, на которых алгоритм допустил ошибку
- $\sum_{t=1}^{T} f_t(w^{(t)}) \geqslant |\mathcal{M}|$
- обозначив $R = \max_t ||x_t||$ и $\eta = \dfrac{||w^*||}{R\sqrt{|\mathcal{M}|}}$, получим:
$$|\mathcal{M}| - R||w^*||\sqrt{|\mathcal{M}|} - \sum_{t=1}^{T} f_t(w^*) \leqslant 0$$

Теорема о Online Perceptron

Теорема о Online Perceptron

Пусть мы запустили Online Perceptron на последовательности $(x_1, y_1), \ldots, (x_T, y_T)$ и $R = \max_t ||x_t||$. Пусть $\mathcal{M}$ — раунды, на которых алгоритм ошибся и $f_t(w) = 1_{[t \in \mathcal{M}]}[1 - y_t \langle w, x_t \rangle]_+$. Тогда для любого w^*:

$$|\mathcal{M}| \leqslant \sum_t f_t(w^*) + R||w^*||\sqrt{\sum_t f_t(w^*)} + R^2||w^*||^2$$

Если существует w^*, что $y_t \langle w^*, x_t \rangle \geqslant 1$ для всех t, то:

$$|\mathcal{M}| \leqslant R^2||w^*||^2$$

Содержание

Итоги

- изучили модель online обучения
- ввели понятие Ldim, описывающее сложность класса гипотез в online-сценарии
- привели базовые алгоритмы обучения в online-случае

Литература

- Shai Shalev-Shwartz and Shai Ben-David — Understanding Machine Learning: From theory to algorithms (глава 21)