Машинное обучение, ФКН ВШЭ
Семинар №9

15 ноября 2017 г.

1 Градиентный бустинг

На двух прошедших лекциях обсуждался градиентный бустинг, один из способ построения композиций алгоритмов. Вспомним его основные принципы. Будем искать алгоритм, оптимизирующий некоторую дифференцируемую функцию потерь $L(y, z)$, в виде взвешенной суммы базовых алгоритмов:

$$a_N(x) = \sum_{n=0}^{N} \gamma_n b_n(x).$$

Идея бустинга заключается в последовательном построении алгоритмов, каждый из которых учитывает ошибки построенной до сих пор композиции:

$$\sum_{i=1}^{\ell} L(y_i, a_{N-1}(x_i) + \gamma_N b_N(x_i)) \to \min_{b_N, \gamma_N}.$$

После выбора каких-нибудь простых γ_0 и $b_0(x)$ (например, для задачи регрессии можно положить $\gamma_0 = 1$ и $b_0(x) = \frac{1}{\ell}\sum_{i=1}^{l} y_i$) все последующие базовые алгоритмы стараются приблизить антиградиент суммы функций потерь, взятый в точках $z = a_{N-1}(x_i)$:

$$s_i = -\left.\frac{\partial L}{\partial z}\right|_{z=a_{N-1}(x_i)}.$$

При этом приближается антиградиент с точки зрения квадратичной функции потерь:

$$b_N(x) = \arg\min_{b \in \mathcal{A}} \sum_{i=1}^{\ell} (b(x_i) - s_i)^2.$$

Подбор коэффициентов производится по аналогии с методом наискорейшего спуска:

$$\gamma_N = \arg\min_{\gamma \in \mathbb{R}} \sum_{i=1}^{\ell} L(y_i, a_{N-1}(x_i) + \gamma b_N(x_i)).$$

Мы уже знаем, что для квадратичной функции потерь $L(y,z) = (y-z)^2$ сдвиги s_i^* выражаются как $s_i^* = y_i - a_N(x_i)$. Посмотрим, чему равны сдвиги для других функций потерь.

Задача 1.1. *Найдите сдвиги для функции потерь* $L(y,z) = |y-z|$.

Решение.

$$s_i = -\left.\frac{\partial |y_i - z|}{\partial z}\right|_{z=a_{N-1}(x_i)} = \operatorname{sign}(y_i - z)\Big|_{z=a_{N-1}(x_i)} = \operatorname{sign}(y_i - a_{N-1}(x_i))$$

■

Задача 1.2. *Найдите сдвиги для логистической функции потерь*
$L(y,z) = \log(1+\exp(-yz))$.

Решение. Вспомним, что логистическая функция потерь выражается через сигмоиду $\sigma(x) = \frac{1}{1+\exp(-x)}$ следующим образом:

$$L(y,z) = \log\left(\frac{1}{\sigma(yz)}\right) = -\log\sigma(yz)$$

Далее, пользуясь формулой для производной сигмоиды $\sigma'(x) = \sigma(x)(1-\sigma(x))$, получаем

$$s_i = \left.\frac{\partial \log\sigma(y_i z)}{\partial z}\right|_{z=a_{N-1}(x_i)} = \left.\frac{1}{\sigma(y_i z)}\sigma(y_i z)(1-\sigma(y_i z))y_i\right|_{z=a_{N-1}(x_i)} =$$

$$= (\sigma(y_i a_{N-1}(x_i)) - 1)y_i = \frac{y_i}{1+\exp(y_i a_{N-1}(x_i))}.$$

■

Теперь посмотрим, как выглядят оптимальные коэффициенты для квадратичной функции потерь:

Задача 1.3. *Предположим, что на очередном шаге бустинга сдвиги для обучения получились равны s_i, и на этих сдвигах был обучен алгоритм $b_N(x)$. Найдите оптимальное значение веса алгоритма γ_N для квадратичной функции потерь.*

Решение.

$$\gamma_N = \arg\min_{\gamma\in\mathbb{R}} \sum_{i=1}^{\ell} L(y_i, a_{N-1}(x_i) + \gamma b_N(x_i)) = \arg\min_{\gamma\in\mathbb{R}} \sum_{i=1}^{\ell} (y_i - (a_{N-1}(x_i) + \gamma b_N(x_i)))^2 =$$

$$= \arg\min_{\gamma\in\mathbb{R}} \sum_{i=1}^{\ell} (s_i - \gamma b_N(x_i)))^2.$$

Заметим, что функционал является выпуклой функцией относительно γ, поэтому для нахождения минимума продифференцируем функцию по γ:

$$\frac{\partial\left(\sum_{i=1}^{\ell}(s_i-\gamma b_N(x_i)))^2\right)}{\partial\gamma}=\sum_{i=1}^{\ell}\frac{\partial(s_i-\gamma b_N(x_i)))^2}{\partial\gamma}=\sum_{i=1}^{\ell}(s_i-\gamma b_N(x_i))(-b_N(x_i))=$$

$$=\gamma\sum_{i=1}^{\ell}b_N(x_i)^2-\sum_{i=1}^{\ell}s_ib_N(x_i).$$

Приравнивая результат к нулю, получаем

$$\gamma=\frac{\sum_{i=1}^{\ell}s_ib_N(x_i)}{\sum_{i=1}^{\ell}b_N(x_i)^2}.$$

■

2 Градиентный бустинг над деревьями

С точки зрения разложения на смещение и разброс в ходе обучения каждого нового алгоритма уменьшается смещение композиции, а разброс же остаётся неизменным или увеличивается. Поэтому на практике чаще всего в качестве базовых алгоритмов используются деревья небольшой грубины, обладающие большим смещением, но не склонные к переобучению. Мы знаем, что решающее дерево разбивает все пространство на непересекающиеся области, в каждой из которых его ответ равен константе:

$$b_n(x)=\sum_{j=1}^{J_n}b_{nj}[x\in R_j],$$

где $j=1,\ldots,J_n$ — индексы листьев, R_j — соответствующие области разбиения, b_{nj} — значения в листьях.

Значит, на N-й итерации бустинга композиция обновляется как

$$a_N(x)=a_{N-1}(x)+\gamma_N\sum_{j=1}^{J_N}b_{Nj}[x\in R_j].$$

Видно, что добавление в композицию одного дерева с J_N листьями равносильно добавлению J_N базовых алгоритмов, представляющих собой предикаты. Мы можем улучшить качество композиции, подобрав свой коэффициент при каждом из предикатов:

$$\sum_{i=1}^{\ell}L\left(y_i,a_{N-1}(x_i)+\sum_{j=1}^{J_N}\gamma_{Nj}[x\in R_j]\right)\to\min_{\{\gamma_{Nj}\}_{j=1}^{J_N}}.$$

Поскольку области разбиения R_j не пересекаются, данная задача распадается на J_N независимых подзадач:

$$\gamma_{Nj} = \arg\min_{\gamma} \sum_{x_i \in R_j} L(y_i, a_{N-1}(x_i) + \gamma), \qquad j = 1, \dots, J_N.$$

Для простых данных подберём вручную оптимальные коэффициенты:

Задача 2.1. *Дана следующая выборка:*

$f_1(x)$	$f_2(x)$	y
1	1	2
4	4	2
1	5	8
1	2	3
5	2	6

Допустим, мы оптимизируем $L(y, z) = |y - z|$. В качестве начального алгоритма мы выбрали предсказание медианой: $a_0(x) = median\{y_i\}_{i=1}^{5} = 3$, на первом шаге построили дерево с четырьмя листьями: $R_1 = \{(x_3, y_3)\}, R_2 = \{(x_5, y_5)\}, R_3 = \{(x_2, y_2)\}, R_4 = \{(x_1, y_1), (x_4, y_4)\}$. Подберите оптимальные коэффициенты γ_{1j} для каждого из листов.

Решение.

Как было сказано, чтобы найти коэффициенты в листах, достаточно решить независимо 4 задачи оптимизации. Распишем задачу для первого листа:

$$\gamma_{11} = \arg\min_{\gamma \in \mathbb{R}} \sum_{x_i \in R_1} L(y_i, a_0(x_i) + \gamma) = \arg\min_{\gamma \in \mathbb{R}} |8 - (3 + \gamma)| = 5.$$

Аналогично находятся $\gamma_{12} = 3$ и $\gamma_{13} = 1$. Для последнего листа:

$$\gamma_{14} = \arg\min_{\gamma \in \mathbb{R}} \sum_{x_i \in R_4} L(y_i, a_0(x_i) + \gamma) = \arg\min_{\gamma \in \mathbb{R}} |3 - (2 + \gamma)| + |3 - (3 + \gamma)|.$$

Поэтому в качестве γ_{14} может служить любое число из отрезка $[0, 1]$.

■

Далее на семинаре мы будем сравнивать градиентный бустинг и случайные леса. Для того, чтобы сравнивать сложность построения композиций, получаемых этими методами, нам потребуется решить следующую задачу:

Задача 2.2. *Дана выборка из ℓ объектов, описываемых d признаками. Приведите временную асимптотику обучения и построения прогнозов для композиции вида $a_N(x) = \sum_{n=0}^{N} b_n(x)$ над решающими деревьями b_n глубины не более, чем D.*

Решение. На стадии обучения при построении очередного решающего дерева глубины D нам необходимо выбрать предикаты в не более чем $2^D - 1$ внутренних вершинах этого дерева. Каждый порог выбирается путем перебора $\leq \ell - 1$ значений для каждого из d признаков. После выбора предиктов необходимо вычислить прогноз дерева в каждом из листов за линейное время от количества объектов обучающей выборки, попавших в лист. Для вычисления прогнозов во всех листах одновременно нам достаточно один раз пройтись по всем объектам. Отсюда асимптотика построения одного решающего дерева — $O(2^D \ell d + \ell) = O(2^D \ell d)$.

На стадии построения прогноза для объекта x он «пропускается» через дерево от корня к листьям, тем самым проходя путь из не более чем D внутренних вершин, в каждой из которых происходит проверка предиката за константное время. Отсюда имеем асимптотику для построения прогноза композиции — $O(ND)$.

■

Таким образом, вычислительно менее сложным оказывается процесс обучения большого количества неглубоких деревьев, а потому градиентный бустинг является более выгодным с точки зрения времени обучения алгоритмом по сравнению со случайным лесом.

Заметим также, что, поскольку обучение градиентного бустинга является «направленным», то ему требуется меньшее по сравнению со случайным лесом количество базовых алгоритмов для достижения того же качества композиции.