Бустинг в задаче регрессии

Рассмотрим задачу минимизации квадратичного функционала:

$$\frac{1}{2}\sum_{i=1}^{l}(a(x_i)-y_i)^2 \to \min_a$$

Будем искать итоговый алгоритм в виде суммы базовых моделей $b_n(x)$:

$$a_N(x) = \sum_{n=1}^{N} b_n(x),$$

где базовые алгоритмы $b_n \in \mathbf{A}$.

Первый базовый алгоритм: $b_1(x) := \underset{b\in\mathbf{A}}{argmin}\frac{1}{2}\sum_{i=1}^{l}(b(x_i)-y_i)^2$.

Остатки на каждом объекте: $s_i^{(1)} = y_i - b_i(x)$

$b_2(x) := \underset{b\in\mathbf{A}}{argmin}\frac{1}{2}\sum_{i=1}^{l}(b(x_i)-s^{(1)})^2$

Бустинг в задаче регрессии

Таким образом, каждый следующий алгоритм тоже будем настраивать на остатки предыдущих:

$$s_i^{(N)} = y_i - \sum_{n=1}^{N-1} b_n(x_i) = y_i - a_{N-1}(x_i), \quad i = 1, \ldots, l$$

$$b_N(x) := \underset{b \in \mathbf{A}}{argmin} \frac{1}{2} \sum_{i=1}^{l} (b(x_i) - s_i^{(N)})^2$$

Также, остатки могут быть найдены как антиградиент функции потерь по ответу модели, посчитанный в точке ответа уже построенной композиции:

$$s_i^{(N)} = y_i - a_{N-1}(x_i) = -\frac{\partial}{\partial z} \frac{1}{2}(z - y_i)^2 \Big|_{z=a_{N-1}(x_i)}$$

Градиентный бустинг

Пусть дана некоторая дифференцируемая функция потерь $L(y, z)$.
Будем строить взвешенную сумму базовых алгоритмов:

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

Примеры выбора алгоритма $b_0(x)$:

1. Нулевой: $b_0(x) = 0$.
2. Возвращающий самый популярный класс (в задачах классификации):
   $b_0(x) = \underset{y \in \mathbb{Y}}{argmax} \sum_{i=1}^{l} [y_i = y]$
3. Возвращающий средний ответ (в задачах регрессии):
   $b_0(x) = \frac{1}{l} \sum_{i=1} l y_i$

Градиентный бустинг (продолжение)

Допустим, мы построили композицию $a_{N-1}(x)$ из $N-1$ алгоритма, и хотим выбрать следующий абзовый алгоритм $b_N(x)$ так, чтобы как можно сильнее уменьшить ошибку:

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

Какие числа $s_1, \ldots, s_l$ надо выбрать для решения следующей задачи:

$$\sum_{i=1}^{l} L\left(y_i, a_{N-1}(x_i) + s_i\right) \to \min_{s_1, \ldots, s_l}$$

Функция алгоритма

- $s_i = y_i - a_{N-1}(x_i)$?
- $s_i = -\frac{\partial L}{\partial z}\Big|_{z=a_{N-1}(x_i)}$

В этом случае сдвиг s_i будет противоположен производной функции потерь в точке $z = a_{N-1}(x_i)$

По данным значениям в конечном числе точек необходимо построить функцию, заданную на всем пространстве объектов.

$b_N(x) = \underset{b \in \mathbf{A}}{argmin} \sum_{i=1}^{l} (b(x_i) - s_i)^2$ — среднеквадратичная ошибка

$\gamma_N = \underset{\gamma \in \mathbb{R}}{argmin} \sum_{i=1}^{l} L(y_i, a_{N-1}(x_i) + \gamma b_N(x_i))$ — подбор коэффициента

Проблемы градиентного бустинга

- Если базовые алгоритмы очень простые, то они плохо приближают вектор антиградиента. Соответственно, градиентный бустинг может свестись к случайному блужданию в пространстве.
- Если базовые алгоритмы сложные, то они способны за несколько шагов бустинга идеально подогнаться под обучающую выборку.
  Решение:
  - Сокращение шага
    $a_N(x) = a_{N-1}(x) + \eta\gamma_N b_N(x)$, где $\eta \in (0, 1]$ — темп обучения.
  - Стохастический градиентный бустинг

Функции потерь

- Регрессия
  - Квадратичная $\sum_{i=1}^{l}(a(x_i) - y_i)^2 \to \min\limits_{a}$
  - Модуль отклонения $L(y, z) = |y - z|$, для которого антиградиент вычисляется по формуле $s_i^{(N)} = -sign(a_{N-1}(x_i) - y_i)$
- Классификация

  $L(y, z) = log(1 + e^{-yz})$

  Задача поиска базового алгоритма с ней принимает вид:

  $b_N = \underset{b \in \mathbf{A}}{argmin} \sum_{i=1}^{l}(b(x_i) - \frac{y_i}{1+exp(y_i a_{N-1}(x_i))})^2$

Особенность логистической функции

Ошибка на N-ой итерации:

$$Q(a_N) = \sum_{i=1}^{l} log(1 + exp(-y_i a_{N-1}(x_i))exp(-y_i \gamma_n b_N(x_i)))$$

Таким образом, величина $w_i^{(N)} = exp(-y_i a_{N-1} x(i))$ может случить мерой важности объекта x_i на N-й итерации градиентного бустинга.

Градиентный бустинг над деревьями

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

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

В N-й итерации бустинга композиция обновляется как

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

Можно улучшить качество композиции:

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

Градиентный бустинг над деревьями

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

$$\gamma_{Nj} = \underset{\gamma}{argmin} \sum_{x_i \in R_j} L(y_i, a_{N-1}(x_i) + \gamma), \qquad j = 1, \ldots, J_N$$

Логистическая функция потерь.

$$F_j^{(N)}(\gamma) = \sum_{x_i \in R_j} log(1 + exp(-y_i(a_{N-1}(x_i) + \gamma))) \to \underset{\gamma}{min}.$$

Смещение и разброс

В случайных лесах:

- Используются глубокие деревья, поскольку от базовых алгоритмов требуется низкое смещение.
- Разброс устраняется за счёт усреднения ответов различных деревьев.

В бустинге:

- Каждый следующий алгоритм понижает ошибку композиции.
- Переобучение при большом количестве базовых моделей.
- Можно понизить смещение моделей, а разброс либо останется таким же, либо увеличится.
- Используются неглубокие решающие деревья.

Сравнение случайного леса и бустинга

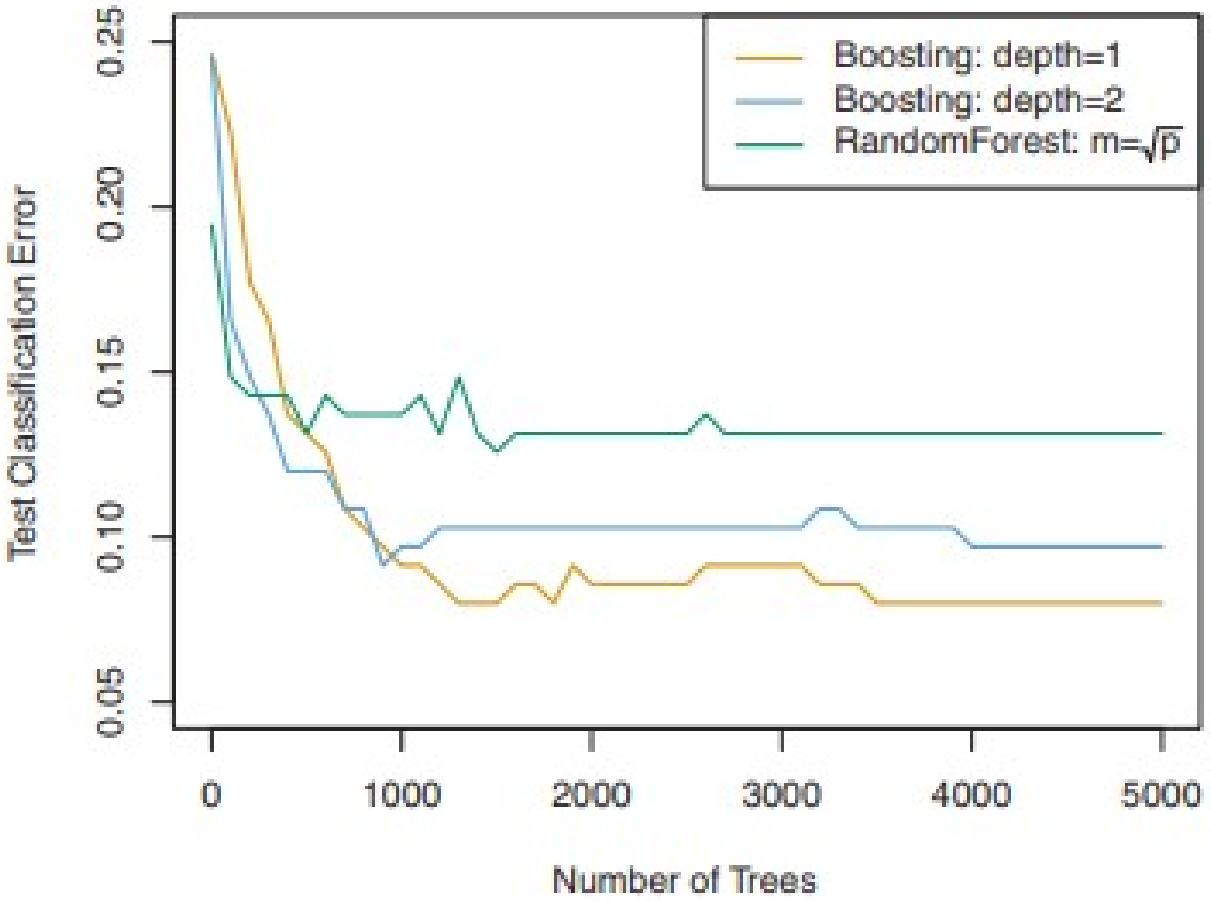

Рис.: График изменения ошибки моделей

Взвешивание объектов

AdaBoost: $L(y, z) = exp(-yz)$

$$L(a, X) = \sum_{i=1}^{l} exp\left(-y_i \sum_{n=1}^{N} \gamma_n b_n(x_i)\right)$$

Компоненты ее антиградиента после $N-1$ итерации:

$$s_i = -L(y_i, z)z|_z = a_{N-1}(x_i) = y_i \underbrace{exp\left(-y_i \sum_{n=1}^{N-1} \gamma_n b_n(x_i)\right)}_{w_i}$$

Влияние шума на обучение

Рассмотрим теперь логистическую функцию потерь, которая также может использоваться в задачах классификации:
$L(a, X^l) = \sum_{i=1}^{l} log(1 + exp(-y_i a(x_i)))$

Ее антиградиент после $N - 1$ шага:
$$s_i = y_i \underbrace{\frac{1}{1 + exp(y_i a_{N-1}(x_i))}}_{w_i^{(N)}}$$