Машинное обучение, ФКН ВШЭ
Семинар №9

1 Бустинг

На лекции обсуждался метод градиентного бустинга, напомним, как он устроен.

Композиция:

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

Выбор сдвигов для обучения:

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

$$s_i = -\left.\frac{\partial L(y_i, z)}{\partial z}\right|_{z=a_{N-1}(x_i)}, \quad i = 1, \dots, \ell$$

Обучение:

$$b_N(x) = \arg\min_b \sum_{i=1}^{\ell} (b(x_i) - s_i)^2$$

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

Разберем по шагам алгоритм градиентного бустинга на простом примере. Будем считать, что обучающая выборка задана рис. 1a, где каждый объект имеет два признака f_1, f_2 и изображен как серый прямоугольник. Координаты центра прямоугольника задают значения его признаков, а написанное в нем число — значение целевой функции на объекте. Будем использовать квадратичную функцию потерь и решающие пни (деревья глубины 1) в качестве базового алгоритма. Для простоты в качестве начального алгоритма выберем тождественный ноль ($b_0 \equiv 0$), а также положим все веса равными единице: ($\gamma_i = 1$). Тогда начальное значение функционала качества равно $Q(b_0) = 3 \cdot 1^2 + 3 \cdot 0^2 = 3$.

При построении первого решающего пня b_1 сначала нужно выбрать оптимальное разбиение в корне. Так как мы используем квадратичную функцию потерь, оптимальное разбиение всего множества R_m на части R_l и R_r — это такое, которое максимизирует функционал качества

$$F(R_m, R_l, R_r) = H(R) - \frac{|R_l|}{|R_m|} H(R_l) - \frac{|R_r|}{|R_m|} H(R_r),$$

$$H(R) = \min_c \frac{1}{|R|} \sum_{i \in R} (y_i - c)^2.$$

Очевидно, что для разбиений, помещающих всю выборку с одной стороны, это значение равно 0. Для разбиений $[f_1(x) < 1]$ и $[f_1(x) < 2]$ это значение равно 0.125, а для $[f_2(x) < 1]$ оно равно ~ 0.028. Таким образом, оптимальными являются разбиения $[f_1(x) < 1]$ и $[f_1(x) < 2]$; будем считать, что было выбрано первое из них. Наименьшую квадратичную ошибку по подвыборке дает среднее значение на ней, поэтому слева от разделяющей прямой мы получим значение 1, а справа — значение $\frac{1}{4}$. Итого получаем $b_1(x) = 1 \cdot [f_1(x) < 1] + \frac{1}{4} \cdot [f_1(x) >= 1]$. Этот решающий пень изображен на рис. 1b. После вычисления сдвига получаем новую обучающую выборку на рис. 1c. Новое значение функционала качества равно $Q(b_0 + b_1) = 2 \cdot 0^2 + \left(\frac{3}{4}\right)^2 + 3\left(-\frac{1}{4}\right)^2 = \frac{3}{4}$.

Для второго алгоритма оптимальным является разбиение $[f_1(x) < 2]$, и, вычисляя среднее, получаем, что слева оно равно $\frac{1}{8}$, а справа равно $-\frac{1}{4}$. Итого получаем $b_2(x) = \frac{1}{8} \cdot [f_1(x) < 2] - \frac{1}{4} \cdot [f_1(x) >= 2]$. Этот решающий пень изображен на рис. 1d. После вычисления сдвига получаем новую обучающую выборку на рис. 1e. Новое значение функционала качества равно $Q(b_0 + b_1 + b_2) = 2\left(-\frac{1}{8}\right)^2 + \left(\frac{5}{8}\right)^2 + \left(-\frac{3}{8}\right)^2 + 2(0)^2 = \frac{9}{16} \approx 0.56$.

Для третьего алгоритма оптимальным является разбиение $[f_2(x) < 1]$, и, вычисляя среднее, получаем, что снизу оно равно $-\frac{1}{6}$, а сверху равно $\frac{1}{6}$. Итого получаем $b_3(x) = -\frac{1}{6} \cdot [f_2(x) < 1] + \frac{1}{6} \cdot [f_2(x) >= 1]$. Этот решающий пень изображен на рис. 1f. После вычисления сдвига получаем новую обучающую выборку на рис. 1g. Новое значение функционала качества равно $Q(b_0 + b_1 + b_2 + b_3) \approx 0.396$. Также на рис. 1h изображена вся композиция на исходной выборке.

Задача 1.1. *(дополнительная) Проделайте еще один шаг бустинга для примера выше, найдите новую композицию и значение функции потерь.*

Задача 1.2. *Ответьте на следующие вопросы*

1. *Правда ли, что бустинг аналогичен бэггингу, так как использует взвешенную сумму алгоритмов, но только в отличие от него подбирает веса в зависимости от качества алгоритма?*

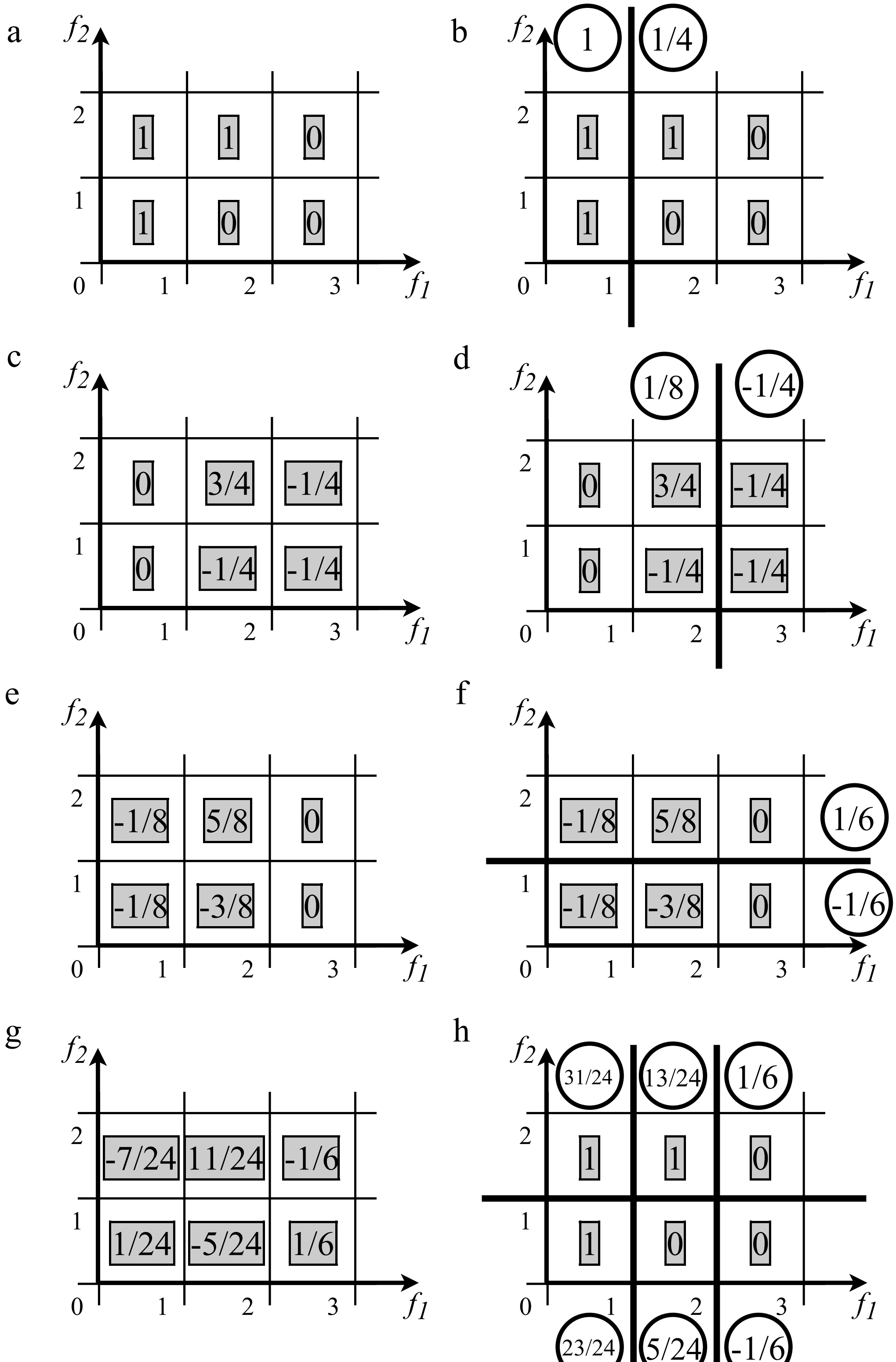

Рис. 1. Пример обучения бустинга

2. *Почему после вычисления градиента мы обучаемся на нем вместо того, чтобы просто сдвинуться в этом направлении, как мы обычно делали в градиентном спуске?*

3. *Почему нельзя оптимизировать функцию потерь сразу по параметрам нового алгоритма?*

4. *Зачем проводить еще одну минимизацию для веса алгоритма, если мы уже обучались на оптимальных сдвигах?*

Решение.

1. Нет, в бэггинге все алгоритмы обучаются на одно и то же распределение пар объект-ответ $p(x, y)$, а в бустинге — на разные распределения (в них целевая функция изменяется в зависимости от исходного распределения и текущего предсказания). Веса при алгоритмах скорее выражают не их значимость, а коэффициент их масштабирования.

2. Потому что градиент берется по объектам, а не по параметрам, то есть для итерации обучения нам нужно еще дополнительно понять, как менять параметры.

3. Некоторые алгоритмы могут быть недифференцируемыми по параметрам.

4. Новый алгоритм скорее всего не сможет предсказывать сдвиги точно, и сдвиг вдоль него не совпадет с желаемым направлением сдвига. Впрочем, даже и при идеальной настройке градиент задает оптимальное направление сдвига только в бесконечно малой окрестности, поэтому любой фиксированный сдвиг может быть слишком большим.

■

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.

Задача обучения веса ставится так:

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

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

Для нахождения минимума продифференцируем функцию по γ.

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

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

Приравнивая результат к нулю, получаем

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

■

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 1.4. *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 1.5. *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

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

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

■

Разберем на примере следующий вопрос: почему мы двигаемся вдоль градиента функции потерь вместо того, чтобы двигаться просто в направлении минимума? Действительно, ведь в отличие от обычного градиентного спуска мы точно знаем, на каком значении достигается минимум — для всех разумных функций потерь будет выполнено $L(z, z) = 0$, то есть минимум по $\gamma_N b_N(x_i)$ достигается в точности на значениях $y_i - a_N(x_i)$, как в случае с квадратичной функцией потерь. Ответ следующий: новый алгоритм скорее всего не сможет точно восстановить искомый сдвиг, поэтому мы сдвинемся в нужном направлении слабее, чем хотелось бы. Действительно, предположим, что мы рассматриваем функцию потерь $L(y, z) = |y - z|$ для двух объектов, находимся в точке $a_1 = 0, a_2 = 0$, и правильные ответы равны $y_1 = 1, y_2 = 3$. Мы можем сдвинуться в любом направлении на расстояние d, и при малых значениях d сдвиг вдоль градиента даст лучший результат. Это продемонстрировано на рис. 2, где по оси абсцисс изображены первые объекты, по оси ординат — вторые, также отмечены линии уровня функции потерь и два возможных направления сдвига. Видно, что сдвиг вдоль градиента пересечет больше линий уровня.

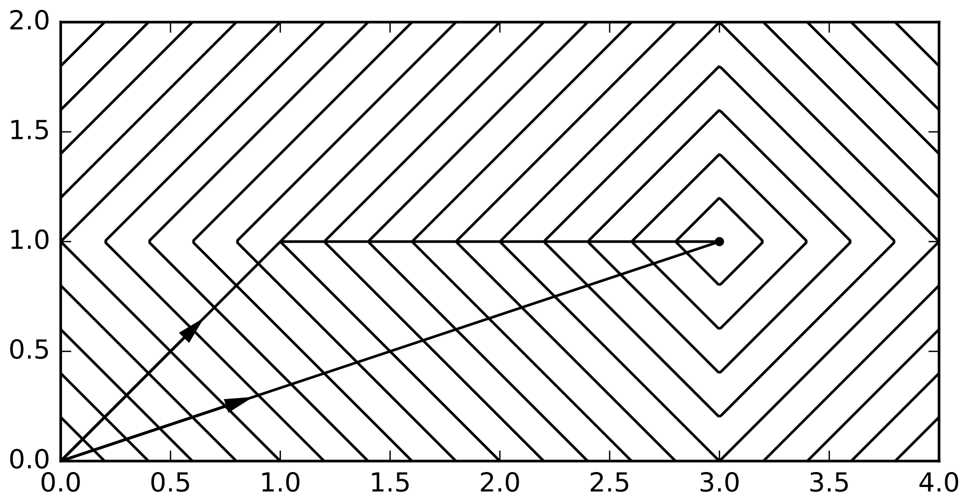

Рис. 2. Оптимизация вдоль функции потерь $L(y, z) = |y - z|$.