Лекция 7. Композиции алгоритмов

Основы интеллектуального анализа данных

Полузёров Т. Д.

БГУ ФПМИ

Функионал риска

Рассмотрим задачу регрессии с квадратичной функцией потерь. Качество алгоритма a:

$$Q(a) = \mathbb{E}_x \, \mathbb{E}_{X,\varepsilon}[y(x,\varepsilon) - a(x,X)]^2$$

- X — обучающая выборка
- $y = f(x) + \varepsilon$ — целевая зависимость, наблюдаемая с точностью до шума ε
- $a(x, X)$ — ответ алгоритма в точке x, обученного по выборке X
- $\mathbb{E}_x$ — среднее по всем тестовым точкам,
- $\mathbb{E}_{X,\varepsilon}$ — среднее по всем обучающим выборкам и случайному шуму

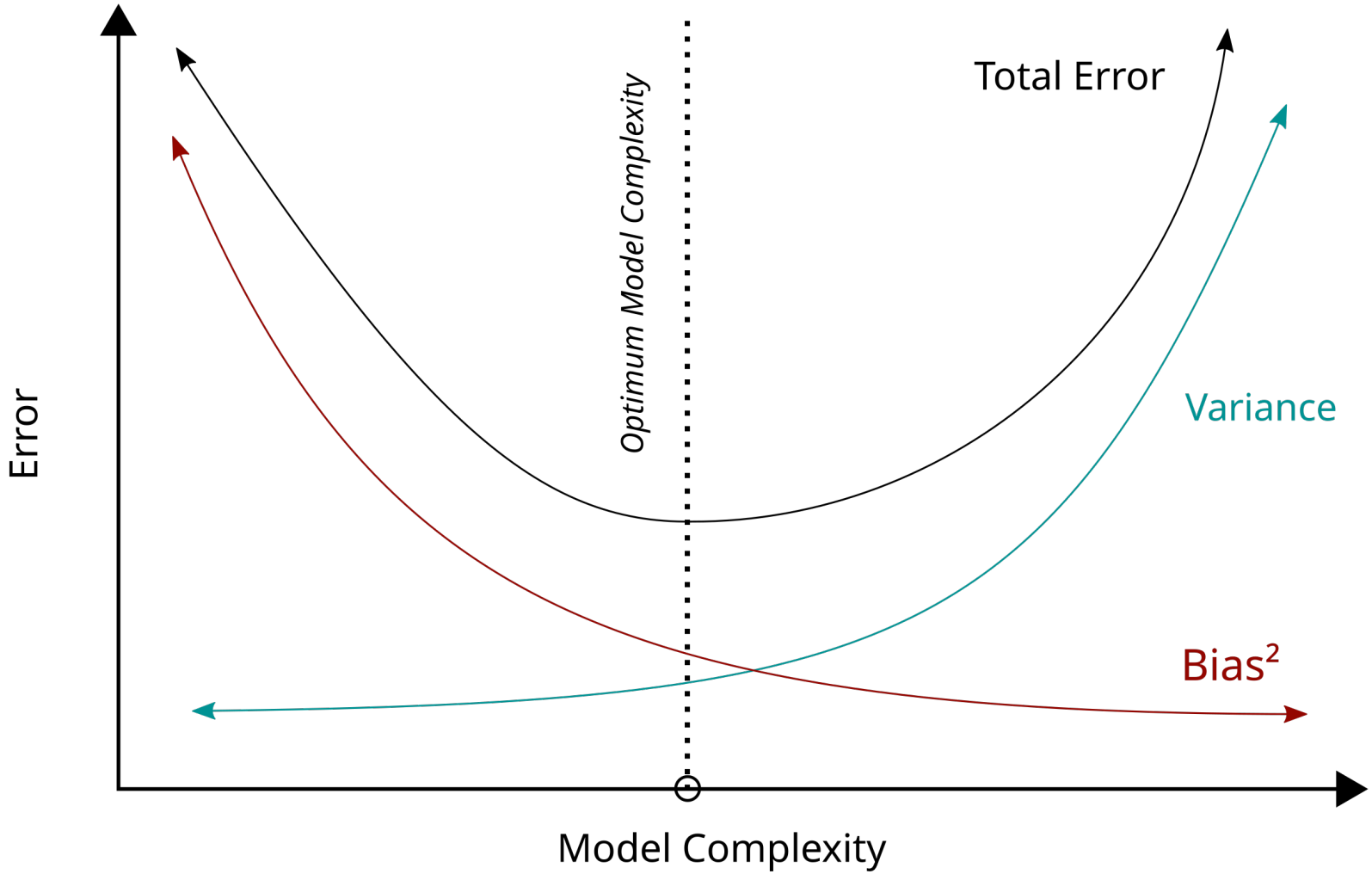
Смещение и разброс

Ошибку можно предствавить в виде трех слагаемых:

$$Q(a) = \mathbb{E}_x\, bias_X^2(a) + \mathbb{E}_x\, variance_X(a) + \sigma^2$$

где

- $bias_X(a) = f(x) - \mathbb{E}_x[a(x, X)]$ — смещение
- $variance(a) = \mathbb{E}_X[a(x, X) - \mathbb{E}_x[a(x, X)]]^2$ — разброс
- $\sigma^2 = \mathbb{E}_x\, \mathbb{E}_\varepsilon[y(x, \varepsilon) - f(x)]^2$ — неустранимый шум


Error
Optimum Model Complexity
Total Error
Variance
Bias^2
Model Complexity

Бутстреп

Пусть имеется выборка X объема ℓ.

С помощью **выбора с возвращением** сформируем новую «выборку» X^1 тоже объема ℓ.
Проделаем эту операцию k раз и получим $\{X^1, \ldots, X^k\}$ **псевдовыборок**.

Такой метод получения псевдовыборок называется **бутстрепом** (bootstrap).

Применяется в статистике для проверки гипотез, построения доверительных интервалов ...

Бэггинг

Сгенерируем k псевдовыборок $\{X^1, \ldots, X^k\}$ и обучим на каждой из них базовую модель b.
Получим k обученных алгоритмов $b_i(x) = b_i(x, X^i), i = 1, \ldots, k$.
Итговый алгоритм есть голосование базовых

$$a(x) = \frac{1}{k}\left(b_1(x) + \cdots + b_k(x)\right)$$

Такой ансамбль алгоритмов — **бэггинг** (bagging, bootstrap aggregation)

Смещение бэггинга

$$bias_X(a) = f(x) - \mathbb{E}_X[a(x, X)] =$$
$$= f(x) - \mathbb{E}_X \left[\sum_{i=1}^{k} \frac{1}{k} b(x, X^i) \right] =$$
$$= f(x) - \mathbb{E}_X \, b(x, X) =$$
$$= bias_X(b)$$

Смещение ансамбля определяется смещением базового алгоритма.

Разброс бэггинга

$$variance_X(a) = \mathbb{E}_X[a(x,X) - \mathbb{E}_x[a(x,X)]]^2 =$$

$$= \mathbb{E}_X \left[\frac{1}{k}\sum_{i=1}^{k} b_i - \mathbb{E}_X \left[\frac{1}{k}\sum_{i=1}^{k} b_i\right]\right]^2 =$$

$$= \frac{1}{k^2}\,\mathbb{E}_X \left[\sum_{i=1}^{k} (b_i - \mathbb{E}_X\, b_i)\right]^2 =$$

$$= \frac{1}{k^2}\sum_{i=1}^{k} variance_X(b_i) + \frac{1}{k^2}\sum_{i\neq j} cov(b_i, b_j)$$

В случае если базовые алгоритмы некоррелированны, то

$$variace_X(a) = \frac{1}{k^2}\sum_{i=1}^{k} variance_X(b_i) =$$
$$= \frac{1}{k^2}\sum_{i=1}^{k} variance_X(b) =$$
$$= \frac{1}{k} variance_X(b)$$

Используя некоррелированную композицию алгоритмов, можно добиться уменьшения разброса в k раз!

Случайный лес

На практике строгое выполнение требования некоррелированности — необяязательно, достаточно, чтобы базовые алгоритмы были непохожи друг на друга.

Посмтроим бэггинг над решающими деревьями

1. Построение i-го дерева
   - сэмплируется псевдовыборка X^i
   - в процессе построения дерева, в каждой вершине выбирается $n^{'} < n$ признаков и по ним ищется сплит (метод случайных подпространств)
2. итоговый ответ ансамбля: среднее для задач регрессии, наиболее популярный класс для классификации

Случайный лес

Бэггинг над решающими дереьями — случайный лес (random forest).
Непохожесть алгоритмов получается за счет обучения на разных псевдовыборках и использовании метода случайных подпространств.

Остаются вопросы

- Какой глубины h строить деревья?
- Сколько признаков n' использовать для обучения?
- Сколько деревьев k использовать в композиции?

Глубина деревьев

Ошибка состоит из смещения и разброса. Разброс снижается за счет композиции, а смещение определяется смещением базового дерева.
Поэтому необходимо использовать **деревья с низким смещением**.

- Неглубокие деревья имеют малое число параметров и поэтому строят только верхнеуровневые зависимости. При разных обучающих подвыборках алгоритмы не будут значительно отличатся (низкий разброс, высокое смещение).
- Глубокие деревья наоборот чересчур сильно подстраиваются под данные, поэтому имеют сильный разброс, но низкое смещение.

Используем **глубокие деревья**.

Число признаков

Большое число признаков обеспечивает слабое различие деревьев, поэтому эффект от бэггинга — слабый.
С другой стороны, используя малое число признаков, базовые деревья будут слабыми.

Практическая рекомендация:

- Для задач регрессии — $n^{'} = n/3$
- Для задач классификации — $n^{'} = \sqrt{n}$

Размер ансамбля

С ростом числа деревьев снижается разброс, но число признаков и вариантов подвыборок — ограничены. Поэтому бесконечно уменьшать разброс не получится.

Имеет смысл построить график зависимости ошибки от числа деревьев и остановиться в тот момент, когда ошибка перестанет значимо уменьшатся

Так же ограничением может выступать время работы ансамбля. Большее число деревьев — большее время принятия решения. Однако, алгоритмы стоятся независимо друг от друга и это позволяет распараллеливать построение отдельных деревьев.

Бустинг

Задана функция потерь $L(a) = (y - a(x))^2$
Будем строить последовательно линейную композицию

$$a(x) = b_1(x) + \cdots + b_k(x)$$

исходя из следующих соображений.
Допустим, на шаге $t-1$ имеем композицию a_{t-1}. Она допускает ошибки $s_{t-1} = y - a_{t-1}$.
Если дополнить композицию $a_t(x) = a_{t-1} + s_{t-1}$ — такой алгоритм решит задачу точно.
Попробуем строить следующий базовый алгоритм $b_t(x)$ который «испрвялет» ошибки, допущенные предыдущей композицией

Требуем от добавления алгоритма к композиции

$$Q(a_k) < Q(a_{k-1})$$

Более подробно

$$Q(a_k) = \sum_{i=1}^{\ell} \mathcal{L}\left(\sum_{j=1}^{k-1} b_j(x_i) + b_k(x_i), y_i\right) \to \min_b$$

a_0 — начальное приближение

$$a_1 = a_0 + b_1$$
$$\dots$$
$$a_t = a_{t-1} + b_t$$

Очень похоже на градиентный спуск

$$b_t = -\nabla\mathcal{L}(a_{t-1})$$

Будем строить b_t, приближающий антиградиент $\mathcal{L}$ в точке a_{t-1}.

$$b_t(x) = \arg\min_b \sum_{i=1}^{\ell} (b(x_i) + \nabla \mathcal{L}(a_{t-1}))^2$$

В градиентных метод оптимизации строится релаксационная последовательность весов, а в градиентном бустинге — последовательность композиций.

Градиентный шаг

Бывает полезным ввести параметр — **темп обучения**
А именно делать градиентный шаг

$$a_t(x) = a_{t-1}(x) + \eta_t b_t(x)$$

величиной η_t.
Это позволит делать более точную подстройку.

На t-м шаге вычислять η_t из решения задачи

$$\eta_t = \arg\min_{\eta>0} \sum_{i=1}^{\ell} \mathcal{L}(a_{t-1} + \eta b_t(x_i), y_i)$$