Разбор летучки

Лекция 13

Композиции алгоритмов

Екатерина Тузова

Мотивация

Где уже видели композиции?

Постановка задачи

$X^l = (x_i, y_i)_{i=1}^l$ – обучающая выборка
$b : X \to R$ — базовый алгоритм
$C : R \to Y$ — решающее правило
R — пространство оценок.

Искомый алгоритм: $a(x) = C(b(x))$

Примеры

1. Классификация (2 класса): $a(x) = \operatorname{sign}(b(x))$
   $b : X \to \mathbb{R}$ $\quad C(b) = \operatorname{sign}(b)$
2. Классификация (M классов): $a(x) = \arg\max_{y \in Y}(b_y(x))$
   $b : X \to \mathbb{R}^M$ $\quad C(b_1, \dots, b_M) = \arg\max_{y \in Y}(b_y)$
3. Регрессия
   $C(b) = b$

Определение композиции

Композиция базовых алгоритмов $b_1, \ldots, b_T$

$a(x) = C(F(b_1(x), \ldots, b_T(x)))$
$F : R^T \to R$ – корректирующая операция

Примеры

1. Простое голосование

$$F(b_1(x), \ldots, b_T(x)) = \frac{1}{T} \sum_{t=1}^{T} b_t(x)$$

2. Взвешенное голосование

$$F(b_1(x), \ldots, b_T(x)) = \sum_{t=1}^{T} \alpha_t b_t(x)$$

3. Смесь алгоритмов

$$F(b_1(x), \ldots, b_T(x)) = \sum_{t=1}^{T} g_t(x) b_t(x)$$

Простое голосование

Теорема Кондорсе "о жюри присяжных"

Если каждый член жюри присяжных имеет независимое мнение, и если вероятность правильного решения члена жюри больше 0.5, то тогда вероятность правильного решения присяжных в целом возрастает с увеличением количества членов жюри и стремится к единице.

Простое голосование

$$F(b_1(x), \dots, b_T(x)) = \frac{1}{T} \sum_{t=1}^{T} b_t(x)$$

Простое голосование

$$F(b_1(x), \dots, b_T(x)) = \frac{1}{T} \sum_{t=1}^{T} b_t(x)$$

$$a(x) = \operatorname{sign} \left(\sum_{t=1}^{T} b_t(x) \right)$$

Простое голосование

$$F(b_1(x), \ldots, b_T(x)) = \frac{1}{T} \sum_{t=1}^{T} b_t(x)$$

$$a(x) = \text{sign}\left(\sum_{t=1}^{T} b_t(x)\right)$$

Если каждый из b_t лучше случайного гадания и $b_1, \ldots, b_T$ достаточно различны, то композиция может работать лучше.

Достаточно различны?

- Настройка по случайным подвыборкам
- Обучение по случайным подмножествам признаков
- Использование различных начальных приближений
- Использование различных моделей

Бэггинг

Идея: обучим b_t независимо по случайным подвыборкам длины l с повторениями.
Доля объектов, которые попадут в выборку ≈ 0.63

Бэггинг

Бэггинг позволяет снизить дисперсию (variance) обучаемого классификатора.

Бэггинг

Бэггинг позволяет снизить дисперсию (variance) обучаемого классификатора.

Бэггинг эффективен на малых выборках, когда исключение даже малой части обучающих объектов приводит к построению существенно различных базовых классификаторов.

Метод случайных подпространств

Идея: обучим b_t независимо по случайным подмножествам n' признаков.

Алгоритм

function BAGGING_RSM(X^l, T, l', n', ε_1, ε_2)

  for $t = 1, \dots, T$ **do**

    U_t – случайное подмножество X^l длины l'

    F_t – случайное подмножество признаков длины n'

    $b_t = \mu(F_t, U_t)$

    if $Q(b_t, U_t) > \varepsilon_1$ или $Q(b_t, X^l \setminus U_t) > \varepsilon_2$ **then**

      не включать b_t в композицию

Случайный лес

Бэггинг над решающими деревьями.

Голосование деревьев классификации, $Y = \{-1, +1\}$
$a(t) = Majority(b_t(x))$

– Каждое дерево $b_t(x)$ обучается по случайной выборке с повторениями
– В каждой вершине предикат выбирается из случайного подмножества n предикатов

Взвешенное голосование

Бустинг

$Y = \{\pm 1\}, \qquad b_t : X \to \{-1, 0, +1\}, \qquad C(b) = \mathrm{sign}(b)$
$b_t(x) = 0$ – отказ от классификации

Бустинг

$$Y = \{\pm 1\}, \qquad b_t : X \to \{-1, 0, +1\}, \qquad C(b) = \mathrm{sign}(b)$$

$b_t(x) = 0$ – отказ от классификации

$$a(x) = \mathrm{sign}\left(\sum_{t=1}^{T} \alpha_t b_t(x)\right)$$

Бустинг

Идея: Фиксируем $\alpha_1 b_1(x) \dots \alpha_{t-1} b_{t-1}(x)$ при добавлении b_t

Бустинг

Идея: Фиксируем $\alpha_1 b_1(x) \dots \alpha_{t-1} b_{t-1}(x)$ при добавлении b_t

$$
\begin{aligned}
&b_1 = \arg\min_{b} Q(b, X^l) \\
&b_2 = \arg\min_{b,F} Q(F(b_1, b), X^l) \\
&\dots \\
&b_t = \arg\min_{b,F} Q(F(b_1, \dots, b_{t-1}, b), X^l)
\end{aligned}
$$

Функционал качества

Функционал качества композиции:

$$Q_T = \sum_{i=1}^{l} \mathcal{L}(\sum_{t=1}^{T} \alpha_t b_t(x_i), y_i) = \sum_{i=1}^{l} \left[y_i \sum_{t=1}^{T} \alpha_t b_t(x_i) < 0 \right] \to \min_{\alpha, b}$$

Функционал качества

Функционал качества композиции:

$$Q_T = \sum_{i=1}^{l} \mathcal{L}(\sum_{t=1}^{T} \alpha_t b_t(x_i), y_i) = \sum_{i=1}^{l} \left[y_i \sum_{t=1}^{T} \alpha_t b_t(x_i) < 0 \right] \to \min_{\alpha, b}$$

Оценка функционала сверху:

$$Q_T \leq \hat{Q}_T = \sum_{i=1}^{l} \underbrace{\exp\left(-y_i \sum_{t=1}^{T-1} \alpha_t b_t(x_i) \right)}_{w_i} \exp(-y_i \alpha_T b_T(x_i)) \to \min_{\alpha, b}$$

Нормируем веса:

$$u_i = w_i / \sum_{j=1}^{l} w_j$$

Бустинг

Взвешенное число ошибочных классификаций:

$$N(b, U^l) = \sum_{i=1}^{l} u_i[b(x_i) = -y_i]$$

Взвешенное число правильных классификаций:

$$P(b, U^l) = \sum_{i=1}^{l} u_i[b(x_i) = y_i]$$

Adaboost

B – семейство базовых алгоритмов.

Пусть для любого нормированного вектора весов U^l существует алгоритм $b \in B$, классифицирующий выборку немного лучше, чем наугад $P(b, U^l) > N(b, U^l)$.
Минимум функционала Q_T достигается при:

$$b_T = \arg\max_{b \in B} \sqrt{P(b, U^l)} - \sqrt{N(b, U^l)}$$

$$\alpha_T = \frac{1}{2} \ln \frac{P(b_T, U^l)}{N(b_T, U^l)}$$

Алгоритм

function $\textsc{AdaBoost}(X^l, T)$

  Инициализировать $w_i = 1/l, \qquad i = 1, \dots, l$

  for $t = 1, \dots, T$ **do**

    $b_t = \arg\max_{b \in B} \sqrt{P(b, U^l)} - \sqrt{N(b, U^l)}$

    $\alpha_t = \frac{1}{2} \ln \frac{P(b_T, U^l)}{N(b_T, U^l)}$

    $w_i = w_i \exp(-\alpha_t y_i b_t(x_i)) \qquad i = 1, \dots, l$

    $u_i = w_i / \sum_{j=1}^{l} w_j$

Рекомендации

- Чаще всего в качестве базовых классификаторов используются решающие деревья
- Для SVM бустинг не эффективен

Наблюдения

\+ Хорошая обобщающая способность

Наблюдения

- \+ Хорошая обобщающая способность
- \+ Простота реализации

Наблюдения

+ Хорошая обобщающая способность
+ Простота реализации
+ Накладные расходы на построение не велики

Наблюдения

+ Хорошая обобщающая способность
+ Простота реализации
+ Накладные расходы на построение не велики

– Склонен к переобучению при наличии большого количества шума в данных

Наблюдения

+ Хорошая обобщающая способность
+ Простота реализации
+ Накладные расходы на построение не велики

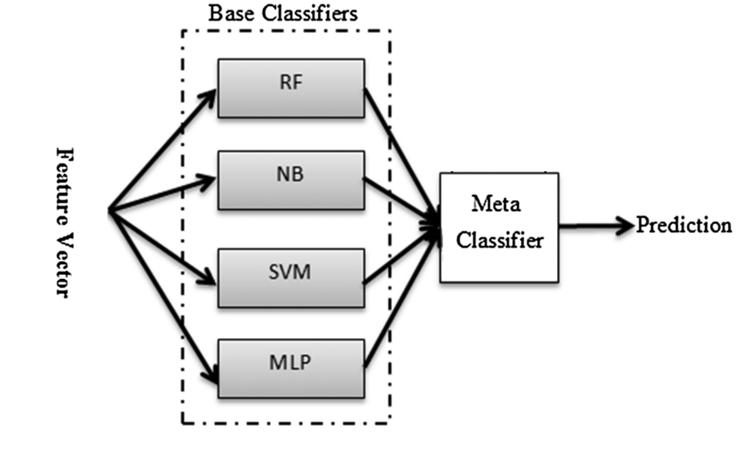
– Склонен к переобучению при наличии большого количества шума в данных
– Требует большой обучающей выборки

Наблюдения

+ Хорошая обобщающая способность
+ Простота реализации
+ Накладные расходы на построение не велики

– Склонен к переобучению при наличии большого количества шума в данных
– Требует большой обучающей выборки
– Жадность приводит к неоптимальности

Stacking


Base Classifiers
Feature Vector
RF
NB
SVM
MLP
Meta
Classifier
Prediction

Почему эти подходы работают

1. Бэггинг уменьшает разброс
2. Бустинг уменьшает разброс и смещение
3. Чем сильнее коррелируют базовые алгоритмы, тем менее эффективны композиции

Вопросы?