

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

$b : X \rightarrow R$ — базовый алгоритм

$C : R \rightarrow Y$ — решающее правило

R — пространство оценок.

Искомый алгоритм: $a(x) = C(b(x))$

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

Композиция базовых алгоритмов b_1, \dots, b_T

$$a(x) = C(F(b_1(x), \dots, b_T(x)))$$

$F : R^T \rightarrow R$ – корректирующая операция

1. Простое голосование

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

2. Взвешенное голосование

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

3. Смесь алгоритмов

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

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.

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

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

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

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

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

Если каждый из b_t лучше случайного гадания и b_1, \dots, 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

Идея: обучим b_t независимо по случайным подвыборкам длины l с повторениями.

Доля объектов, которые попадут в выборку ≈ 0.63

Бэггинг позволяет снизить дисперсию (variance) обучаемого классификатора.

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.

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

```
1 function BAGGING_RSM( $X^l, T, l', n', \varepsilon_1, \varepsilon_2$ )
2   for  $t = 1, \dots, T$  do
3      $U_t$  – случайное подмножество  $X^l$  длины  $l'$ 
4      $F_t$  – случайное подмножество признаков длины  $n'$ 
5      $b_t = \mu(F_t, U_t)$ 
6     if  $Q(b_t, U_t) > \varepsilon_1$  или  $Q(b_t, X^l \setminus U_t) > \varepsilon_2$  then
7       не включать  $b_t$  в композицию
```

Бэггинг над решающими деревьями.

Голосование деревьев классификации, $Y = \{-1, +1\}$

$$a(t) = \text{Majority}(b_t(x))$$

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

Взвешенное голосование

$$Y = \{\pm 1\}, \quad b_t : X \rightarrow \{-1, 0, +1\}, \quad C(b) = \text{sign}(b)$$

$b_t(x) = 0$ – отказ от классификации

$$Y = \{\pm 1\}, \quad b_t : X \rightarrow \{-1, 0, +1\}, \quad C(b) = \text{sign}(b)$$

$b_t(x) = 0$ – отказ от классификации

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

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

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

$$b_1 = \arg \min_b Q(b, X^l)$$

$$b_2 = \arg \min_{b, F} Q(F(b_1, b), X^l)$$

...

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

Функционал качества композиции:

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

Функционал качества композиции:

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

Оценка функционала сверху:

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

Нормируем веса:

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

Взвешенное число ошибочных классификаций:

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

Взвешенное число правильных классификаций:

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

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:

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

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

```
1 function ADABOOST( $X^l, T$ )
2   Инициализировать  $w_i = 1/l, \quad i = 1, \dots, l$ 
3   for  $t = 1, \dots, T$  do
4      $b_t = \arg \max_{b \in B} \sqrt{P(b, U^t)} - \sqrt{N(b, U^t)}$ 
5      $\alpha_t = \frac{1}{2} \ln \frac{P(b_t, U^t)}{N(b_t, U^t)}$ 
6      $w_i = w_i \exp(-\alpha_t y_i b_t(x_i)) \quad i = 1, \dots, l$ 
7      $u_i = w_i / \sum_{j=1}^l w_j$ 
```

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

+ Хорошая обобщающая способность

- + Хорошая обобщающая способность
- + Простота реализации

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

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

- Склонен к переобучению при наличии большого количества шума в данных

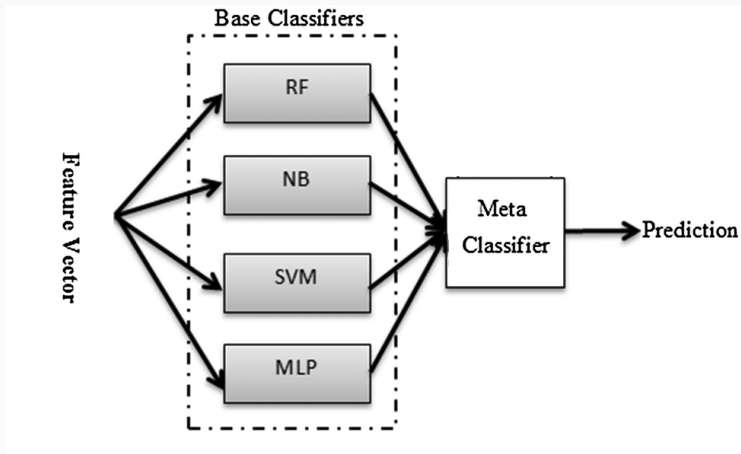
- + Хорошая обобщающая способность
- + Простота реализации
- + Накладные расходы на построение не велики

- Склонен к переобучению при наличии большого количества шума в данных
- Требуется большой обучающей выборки

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

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

Stacking



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?