



Лекция 11: композиции классификаторов

Евгений Борисов

четверг, 29 ноября 2018 г.

КОМПОЗИЦИИ классификаторов

X - объекты y - метки

$a(x)=C(b(x))$ - классификатор

$b : X \rightarrow R$ - базовый алгоритм

$C : R \rightarrow y$ - решающее правило

R - множество оценок (существенно шире чем y)

КОМПОЗИЦИИ КЛАССИФИКАТОРОВ

примеры оценок и решающих правил

$a(x) = \text{sign}(b(x))$ - классификация на 2 класса

$y = \{-1, 0, 1\}$ - метки

$b : X \rightarrow \mathbb{R}$ - базовый алгоритм

$C(b) = \text{sign}(b)$ - решающее правило

КОМПОЗИЦИИ КЛАССИФИКАТОРОВ

примеры оценок и решающих правил

$a(x) = \text{sign}(b(x))$ - **классификация на 2 класса**

$y = \{-1, 0, 1\}$ - метки

$b : X \rightarrow \mathbb{R}$ - базовый алгоритм

$C(b) = \text{sign}(b)$ - решающее правило

$a(x) = \arg\max b_t(x)$ - **классификация на m классов**

$y = \{1, \dots, m\}$ - метки

$R = \mathbb{R}^m$ - оценки

$b : X \rightarrow \mathbb{R}^m$ - базовый алгоритм

$C(b_1, \dots, b_m) = \arg\max b_t$ - оценки

КОМПОЗИЦИИ КЛАССИФИКАТОРОВ

примеры оценок и решающих правил

$a(x) = \text{sign}(b(x))$ - **классификация на 2 класса**

$y = \{-1, 0, 1\}$ - метки

$b : X \rightarrow \mathbb{R}$ - базовый алгоритм

$C(b) = \text{sign}(b)$ - решающее правило

$a(x) = \arg\max b_t(x)$ - **классификация на m классов**

$y = \{1, \dots, m\}$ - метки

$R = \mathbb{R}^m$ - оценки

$b : X \rightarrow \mathbb{R}^m$ - базовый алгоритм

$C(b_1, \dots, b_m) = \arg\max b_t$ - оценки

$a(x) = b(x)$ - **регрессия**

$y = R = \mathbb{R}$ - метки / оценки

$b : X \rightarrow \mathbb{R}$ - базовый алгоритм

$C(b) = b$ - решающее правило (вырождено)

КОМПОЗИЦИИ КЛАССИФИКАТОРОВ

X - объекты y - метки

$b_t: X \rightarrow R$ - базовый алгоритм

$C: R \rightarrow y$ - решающее правило

R - множество оценок

композиция базовых алгоритмов $b_t(x)$

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

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

КОМПОЗИЦИИ КЛАССИФИКАТОРОВ

$a(x) = C\left(F\left(b_1(x), \dots, b_T(x)\right)\right)$ композиция из классификаторов b_t

примеры корректирующих операций

$$F\left(b_1(x), \dots, b_T(x)\right) = \frac{1}{T} \sum_{i=1}^T b_t(x) \quad \text{простое голосование}$$

КОМПОЗИЦИИ КЛАССИФИКАТОРОВ

$a(x) = C\left(F\left(b_1(x), \dots, b_T(x)\right)\right)$ композиция из классификаторов b_t

примеры корректирующих операций

$$F\left(b_1(x), \dots, b_T(x)\right) = \frac{1}{T} \sum_{i=1}^T b_i(x)$$
 простое голосование

$$F\left(b_1(x), \dots, b_T(x)\right) = \sum_{i=1}^T a_i \cdot b_i(x); a_i \in \mathbb{R}$$
 взвешенное голосование

КОМПОЗИЦИИ КЛАССИФИКАТОРОВ

$a(x) = C\left(F\left(b_1(x), \dots, b_T(x)\right)\right)$ композиция из классификаторов b_t

примеры корректирующих операций

$$F\left(b_1(x), \dots, b_T(x)\right) = \frac{1}{T} \sum_{i=1}^T b_t(x)$$
 простое голосование

$$F\left(b_1(x), \dots, b_T(x)\right) = \sum_{i=1}^T a_t \cdot b_t(x); a_t \in \mathbb{R}$$
 взвешенное голосование

$$F\left(b_1(x), \dots, b_T(x)\right) = \sum_{i=1}^T g_t(x) \cdot b_t(x); g_t: X \rightarrow \mathbb{R}$$
 смесь алгоритмов

КОМПОЗИЦИИ КЛАССИФИКАТОРОВ

бустинг

$X \subset \mathbb{R}^n$ - объекты ; $y = \{-1; +1\}$ - метки

$b_t(x): X \rightarrow \{-1, 0, +1\}$ классификатор с отказами

взвешенное голосование

$$a(x) = \text{sign} \left(\sum_{i=1}^T a_i \cdot b_i(x) \right)$$

функционал качества - количество ошибок

$$Q_T = \sum_{i=1} \left[\left(y_i \cdot \sum_{t=1}^T a_t \cdot b_t(x_i) \right) < 0 \right]$$

последовательно добавляем компоненты $a_t b_t(x)$

при этом фиксируем параметры предыдущих компонент

КОМПОЗИЦИИ классификаторов

AdaBoost

взвешенное голосование

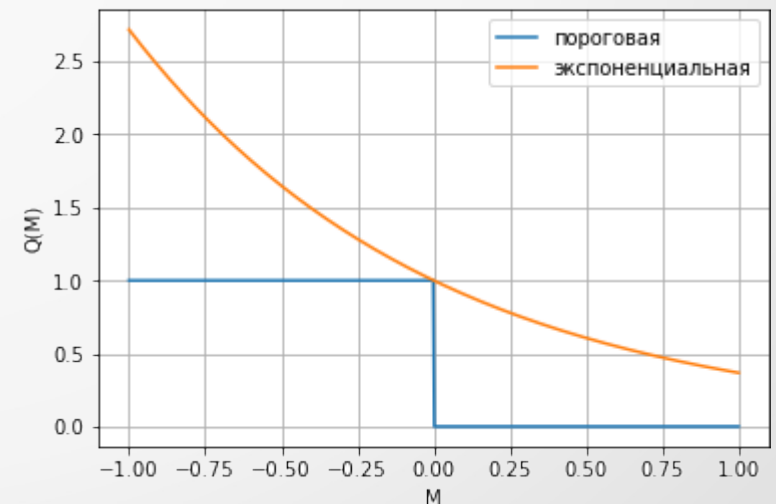
$$a(x) = \text{sign} \left(\sum_{i=1}^T a_i \cdot b_i(x) \right)$$

функционал качества - количество ошибок

$$Q_T = \sum_{i=1} \left[\left(y_i \cdot \sum_{i=1}^T a_i \cdot b_i(x_i) \right) < 0 \right]$$

функционал качества - пороговая функция
оптимизировать не удобно
заменим его на гладкую аппроксимацию

$$Q_T \leq \tilde{Q}_T = \sum_i \left[\exp \left(- y_i \cdot \sum_{i=1}^T a_i \cdot b_i(x_i) \right) \right]$$



КОМПОЗИЦИИ классификаторов

AdaBoost обучение

взвешенное голосование

$$a(x) = \text{sign} \left(\sum_{i=1}^T a_i \cdot b_i(x) \right)$$

функционал качества

$$Q_T = \sum_i \left[\exp \left(- y_i \cdot \sum_{i=1}^T a_i \cdot b_i(x_i) \right) \right]$$

для каждого учебного примера X введём вес w

КОМПОЗИЦИИ классификаторов

AdaBoost обучение

взвешенное голосование

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

функционал качества

$$Q_T = \sum_i \left[\exp \left(- y_i \cdot \sum_{t=1}^T a_t \cdot b_t(x_i) \right) \right]$$

для каждого учебного примера X введём вес w

сумма весов w примеров x , классифицированных b с ошибкой

$$N(b, W) = \sum_i w_i [b(x_i) \neq y_i]$$

сумма весов w примеров x , классифицированных b верно

$$P(b, W) = \sum_i w_i [b(x_i) = y_i]$$

КОМПОЗИЦИИ классификаторов

AdaBoost обучение

Теорема Freund, Schapire (1996)

пусть для вектора весов W существует классификатор b , который классифицирует выборку X лучше чем наугад ($P > N$) тогда минимум функционала Q_T достигается при следующих параметрах.

$$a_t = \frac{1}{2} \ln \left(\frac{P(b_t, W)}{N(b_t, W)} \right)$$

$$b_T = \underset{b}{\operatorname{argmax}} \sqrt{P(b, W)} - \sqrt{N(b, W)}$$

КОМПОЗИЦИИ классификаторов

AdaBoost обучение

$w_i = \frac{1}{n}$ начальные значения весов примеров,
n - количество примеров

последовательно обучаем и добавляем компоненты композиции
с учётом весов примеров w

КОМПОЗИЦИИ классификаторов

AdaBoost обучение

$w_i = \frac{1}{n}$ начальные значения весов примеров,
n - количество примеров

последовательно обучаем и добавляем компоненты композиции
с учётом весов примеров w

вес классификатора
в композиции

$$a_t = \frac{1}{2} \ln \left(\frac{P(b_t, W)}{N(b_t, W)} \right)$$

КОМПОЗИЦИИ классификаторов

AdaBoost обучение

$w_i = \frac{1}{n}$ начальные значения весов примеров,
n - количество примеров

последовательно обучаем и добавляем компоненты композиции
с учётом весов примеров w

вес классификатора
в композиции

$$a_t = \frac{1}{2} \ln \left(\frac{P(b_t, W)}{N(b_t, W)} \right)$$

изменение весов примеров

при добавлении классификатора b_t

$$w_i := w_i \cdot \exp(-y_i \cdot a_t \cdot b_t(x_i))$$

$$w_i := \frac{w_i}{\sum_i w_i} \quad \text{нормируем веса после коррекции}$$

$$Q_T = \sum_i \left[\underbrace{\exp \left(-y_i \cdot \sum_{t=1}^{T-1} a_t \cdot b_t(x_i) \right)}_{w_i} \cdot \exp(-y_i \cdot a_T \cdot b_T(x_i)) \right]$$

КОМПОЗИЦИИ классификаторов

AdaBoost обучение

$w_i = \frac{1}{n}$ начальные значения весов примеров,
n - количество примеров

последовательно обучаем и добавляем компоненты композиции
с учётом весов примеров w

вес классификатора
в композиции

$$a_t = \frac{1}{2} \ln \left(\frac{P(b_t, W)}{N(b_t, W)} \right)$$

изменение весов примеров

при добавлении классификатора b_t

$$w_i := w_i \cdot \exp(-y_i \cdot a_t \cdot b_t(x_i))$$

$$w_i := \frac{w_i}{\sum_i w_i} \quad \text{нормируем веса после коррекции}$$

$$Q_T = \sum_i \left[\underbrace{\exp\left(-y_i \cdot \sum_{t=1}^{T-1} a_t \cdot b_t(x_i)\right)}_{w_i} \cdot \exp(-y_i \cdot a_T \cdot b_T(x_i)) \right]$$

«трудные» примеры получают больший вес

КОМПОЗИЦИИ классификаторов

AdaBoost

инициализировать
веса примеров w

обучаем классификатор $b(x)$
на взвешенных примерах

считаем ошибки $N(b, w)$

$N > 0.5$

вычисляем вес α
классификатора $b(x)$
добавляем его в композицию

вычисляем потерю Q

$Q < \text{порога}$

корректируем
веса примеров w

конец
работы

КОМПОЗИЦИИ классификаторов

примеры базовых классификаторов для AdaBoost

решающие деревья

пороговый классификатор

КОМПОЗИЦИИ КЛАССИФИКАТОРОВ

примеры базовых классификаторов для AdaBoost

решающие деревья

пороговый классификатор

AdaBoost строит длинные композиции из простых классификаторов

для коротких композиции из сложных классификаторов (SVM) результаты AdaBoost сильно хуже

КОМПОЗИЦИИ КЛАССИФИКАТОРОВ

другие методы построения композиций

bagging

обучение по случайным подвыборкам набора примеров,
подвыборки могут пересекаться,
применяем на больших наборах

композиции классификаторов

другие методы построения композиций

bagging

обучение по случайным подвыборкам набора примеров,
подвыборки могут пересекаться,
применяем на больших наборах

rsm (random subspace method)

обучение на случайном подмножестве признаков,
применяем если много признаков

КОМПОЗИЦИИ КЛАССИФИКАТОРОВ

другие методы построения композиций

bagging

обучение по случайным подвыборкам набора примеров,
подвыборки могут пересекаться,
применяем на больших наборах

rsm (random subspace method)

обучение на случайном подмножестве признаков,
применяем если много признаков

bagging и rsm можно комбинировать

КОМПОЗИЦИИ классификаторов

схема построения композиции bagging/rsm

выделяем случайное подмножество примеров/признаков

обучаем классификатор

если результат классификатора хороший
то добавляем в композицию

если ошибка композиции уменьшилась
то повторяем
иначе завершение работы

$$a(x) = \text{sign} \left(\sum_{i=1}^T b_i(x) \right) \quad \text{композиция - простое голосование}$$

$b_t(x): X \rightarrow \{-1, 0, +1\}$ классификатор с отказами

КОМПОЗИЦИИ классификаторов

метод RandomForest (случайный лес)

bagging над решающими деревьями без pruning (без оптимизации)

признак в каждой вершине выбираем из случайного подмножества размера k всех признаков учебного набора размера n

$$k = \sqrt{n} \quad \text{для задач классификации}$$

подбираем количество деревьев T по критерию out-of-bag

$$\text{out-of-bag}(a) = \sum_{i=1}^{\ell} \left[\text{sign} \left(\sum_{t=1}^T [x_i \notin U_t] b_t(x_i) \right) \neq y_i \right] \rightarrow \min$$

т.е. проверяем количество ошибок
на учебных наборах других деревьев

КОМПОЗИЦИИ классификаторов: литература

git clone https://github.com/mechanoid5/ml_lectorium.git

К.В. Воронцов Композиция классификаторов. - курс
"Машинное обучение" ШАД Яндекс 2014

Е.С.Борисов Бустинг - композиции классификаторов
<http://mechanoid.kiev.ua/ml-adaboost.html>

<http://www.machinelearning.ru>

КОМПОЗИЦИИ классификаторов



Вопросы ?

КОМПОЗИЦИИ КЛАССИФИКАТОРОВ: практика

источники данных для экспериментов



sklearn.datasets
UCI Repository
kaggle



задание

- разделить на train/test/valid
- посчитать метрики качества AdaBoost
- реализовать bagging/rsm