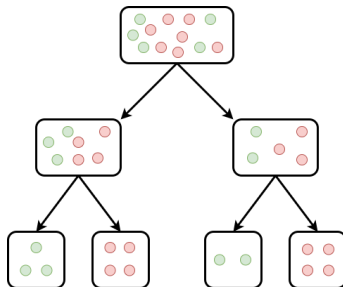


Решающие деревья

Виктор Китов

victorkitov.github.io



Курс поддержан
фондом
'Интеллект'



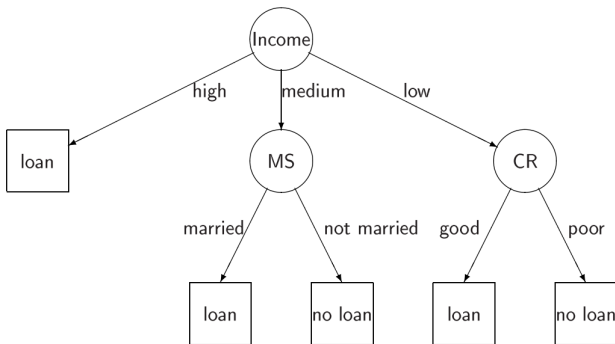
Победитель
конкурса VK среди
курсов по IT



Содержание

- 1 Понятие решающего дерева
- 2 Правила спуска
- 3 Выбор критерия ветвления
- 4 Назначение прогнозов листьям
- 5 Критерий остановки
- 6 Анализ решающих деревьев

Пример решающего дерева



Определение решающего дерева

- Прогнозы строятся деревом T .
- Для каждого внутреннего узла t задана функция ветвления $Q_t(x)$.
- Для каждого ребра $1, \dots, K_t$ ассоциирован набор множеств $S_t(1), \dots, S_t(K_t)$.
 - $Q_t(x) \in S_t(i) \Rightarrow$ спуститься в узел i .
 - $\bigcup_k S_t(k) = \text{range}[Q_t(\cdot)]$
 - $S_t(i) \cap S_t(j) = \emptyset \ \forall i \neq j$

Построение прогноза

- множество вершин разделяется на:
 - внутренние вершины $int(T)$, каждая имеет ≥ 2 потомков
 - терминальные вершины $terminal(T)$, которые не имеют дочерних, а ассоциированы с прогнозами.

Построение прогноза

- множество вершин разделяется на:
 - внутренние вершины $int(T)$, каждая имеет ≥ 2 потомков
 - терминальные вершины $terminal(T)$, которые не имеют дочерних, а ассоциированы с прогнозами.
- Прогноз для дерева T :
 - $t = root(T)$
 - пока t - не терминальная вершина:
 - рассчитать $Q_t(x)$
 - определить j такой, что $Q_t(x) \in S_t(j)$
 - спуститься в j -ую дочернюю вершину $t := t_j$
 - вернуть прогноз, ассоциированный с листом t .

Спецификация решающего дерева

Спецификация решающего дерева:

- функции ветвления $Q_t(x) \forall t \in \text{IntNodes}$
- в каждом внутреннем узле t :
 - число дочерних вершин: K_t
 - разбиение: $S_t(1), \dots, S_t(K_t)$
- прогноз в каждом листе дерева

Спецификация решающего дерева

Спецификация решающего дерева:

- функции ветвления $Q_t(x) \forall t \in \text{IntNodes}$
- в каждом внутреннем узле t :
 - число дочерних вершин: K_t
 - разбиение: $S_t(1), \dots, S_t(K_t)$
- прогноз в каждом листе дерева

Спецификация обучения:

- критерий остановки
 - когда узел становится терминальным при построении top-down

Содержание

- 1 Понятие решающего дерева
- 2 Правила спуска**
- 3 Выбор критерия ветвления
- 4 Назначение прогнозов листьям
- 5 Критерий остановки
- 6 Анализ решающих деревьев

Возможные правила спуска (предикаты)

- $S_t(1) = \{x^{i(t)} \leq h_t\}$, $S_t(2) = \{x^{i(t)} > h_t\}$
- $Q_t(x) = x^{i(t)}$, где $S_t(j) = v_j$, $v_1, \dots, v_K = \text{unique}(x^{i(t)})$.
- $S_t(j) = \{h_j < x^{i(t)} \leq h_{j+1}\}$ для набора порогов $h_1, h_2, \dots, h_{K_t+1}$.
- $S_t(1) = \{x : \langle x, w \rangle \leq h\}$, $S_t(2) = \{x : \langle x, w \rangle > h\}$
- $S_t(1) = \{x : \|x\| \leq h\}$, $S_t(2) = \{x : \|x\| > h\}$
- и т.д.

Популярные алгоритмы решающих деревьев

- CART (classification and regression trees)
 - реализован в scikit-learn
- C4.5

Правила спуска для CART

- рассматривается единственный признак:

$$Q_t(x) = x^{i(t)}$$

- бинарные разбиения:

$$K_t = 2$$

- спуск основан на предикатах=сравнении с порогом h_t :

$$S_1 = \{x^{i(t)} \leq h_t\}, S_2 = \{x^{i(t)} > h_t\}$$

- достаточно выбрать порог из уникальных значений признака $x^{i(t)}$
 - применимо для вещественных, порядковых и бинарных признаков
 - категориальные признаки:

Правила спуска для CART

- рассматривается единственный признак:

$$Q_t(x) = x^{i(t)}$$

- бинарные разбиения:

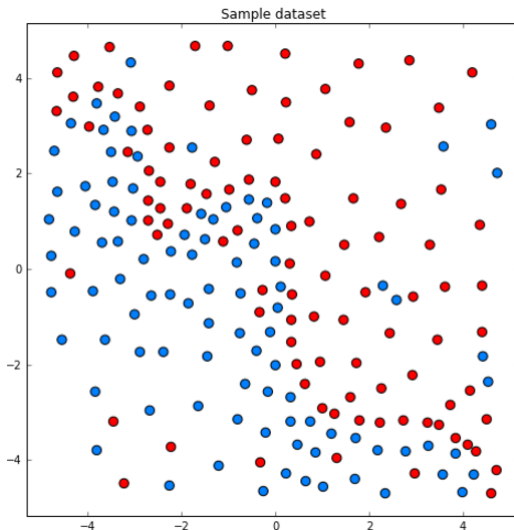
$$K_t = 2$$

- спуск основан на предикатах=сравнении с порогом h_t :

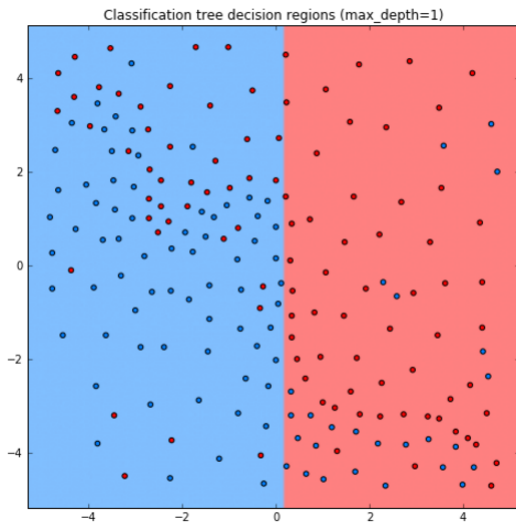
$$S_1 = \{x^{i(t)} \leq h_t\}, S_2 = \{x^{i(t)} > h_t\}$$

- достаточно выбрать порог из уникальных значений признака $x^{i(t)}$
 - применимо для вещественных, порядковых и бинарных признаков
 - категориальные признаки: one-hot, кодирование средним, частотное кодирование.

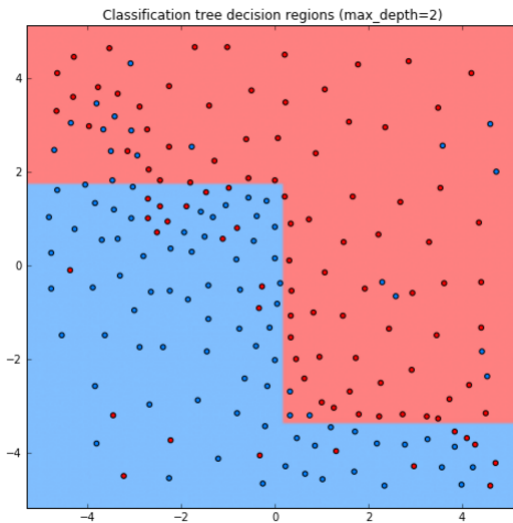
Пример обучающей выборки



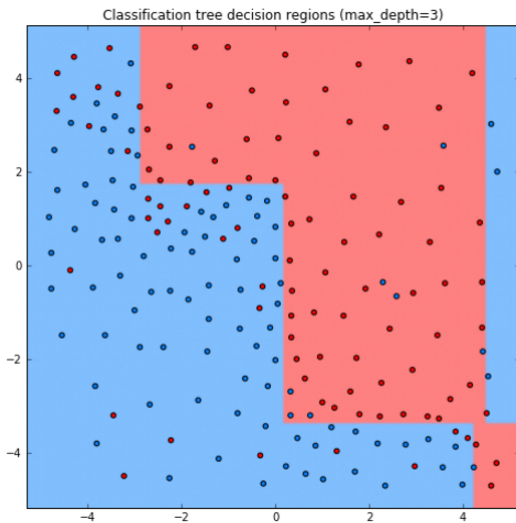
Разбиение на классы (глубина=1)



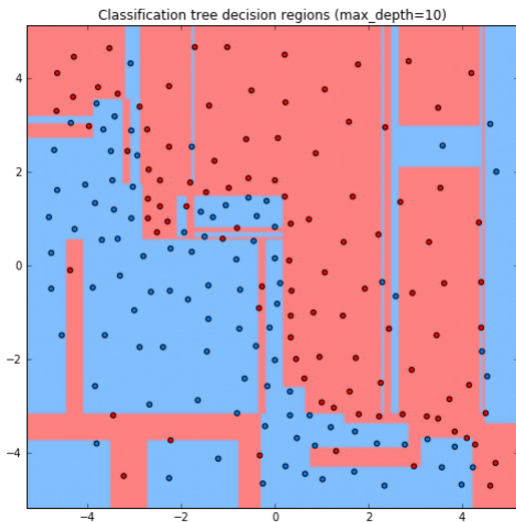
Разбиение на классы (глубина=2)



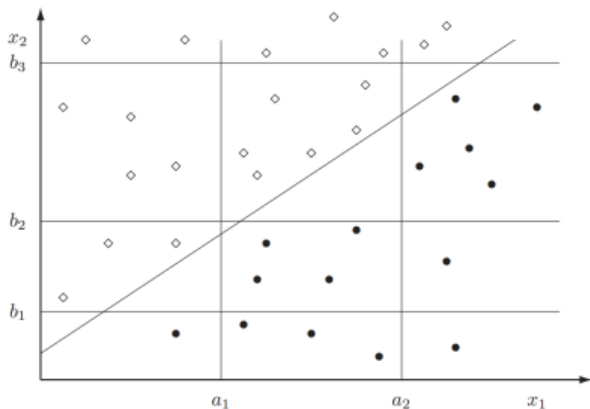
Разбиение на классы (глубина=3)



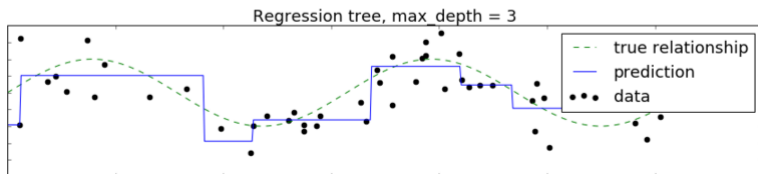
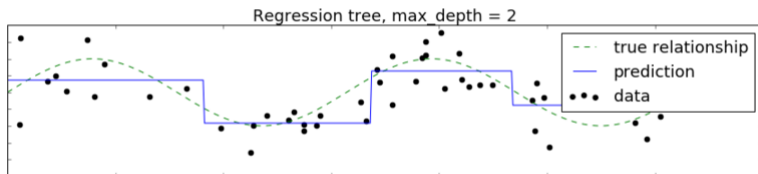
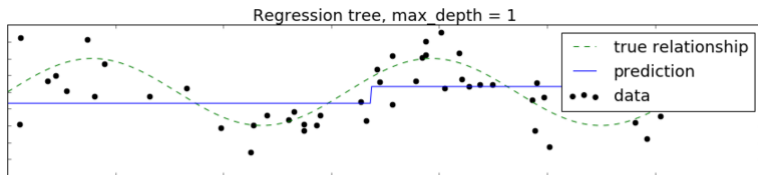
Разбиение на классы (глубина=10)



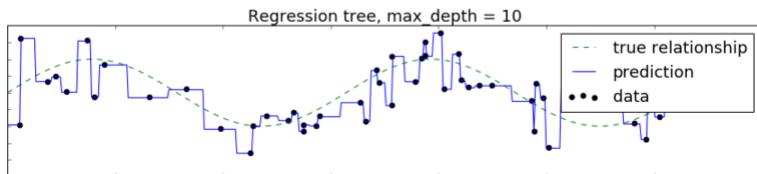
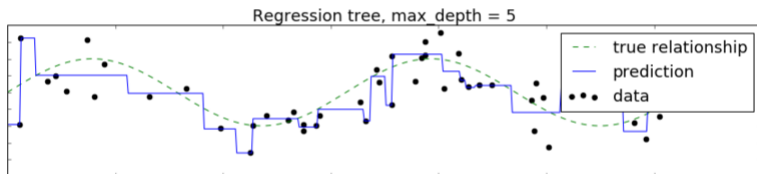
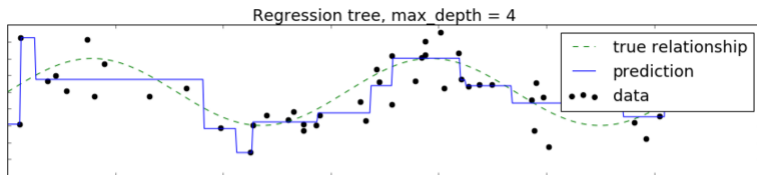
Проблемы с аппроксимацией наклонных границ



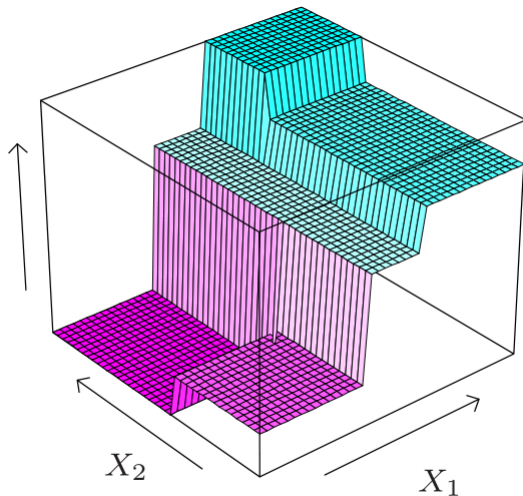
CART для задачи регрессии (малая глубина)



CART для задачи регрессии (большая глубина)



Кусочно-постоянные прогнозы CART



Анализ критерия ветвления CART

Преимущества:

- интерпретируемость (визуализация)
- вычислительная простота прогнозирования
- отбор признаков
- работает для признаков разной природы
 - обрабатывает вещественные, упорядоченные и бинарные признаки
 - прогноз инвариантен к монотонным преобразованиям признака для $Q_t(x) = x^{i(t)}$:

$$x^{i(t)} \leq h \Leftrightarrow f\left(x^{i(t)}\right) \leq f(h) \quad \forall \uparrow f(\cdot)$$

Недостатки:

- константные прогнозы в листьях
 - можно в листах ассоциировать $\hat{y} = f_t(x)$, а не константу \hat{y}_t .
- много вершин - для описания наклонных границ к осям

Содержание

- 1 Понятие решающего дерева
- 2 Правила спуска
- 3 Выбор критерия ветвления**
- 4 Назначение прогнозов листьям
- 5 Критерий остановки
- 6 Анализ решающих деревьев

Определение критерия ветвления

- Из узла t перейти в:
$$\begin{cases} \text{левого потомка } t_L, & \text{если } x^{\hat{I}_t} \leq \hat{h}_t \\ \text{правого потомка } t_R, & \text{если } x^{\hat{I}_t} > \hat{h}_t \end{cases}$$

Определение критерия ветвления

- Из узла t перейти в:
$$\begin{cases} \text{левого потомка } t_L, & \text{если } x^{\hat{I}_t} \leq \hat{h}_t \\ \text{правого потомка } t_R, & \text{если } x^{\hat{I}_t} > \hat{h}_t \end{cases}$$
- Определим $\phi(t)$ - ф-цию неопределенности (критерий информативности, impurity function)
 - измеряет степень неоднозначности u для объектов в узле t .

Определение критерия ветвления

- Из узла t перейти в:
$$\begin{cases} \text{левого потомка } t_L, & \text{если } x^{\hat{I}_t} \leq \hat{h}_t \\ \text{правого потомка } t_R, & \text{если } x^{\hat{I}_t} > \hat{h}_t \end{cases}$$
- Определим $\phi(t)$ - ф-цию неопределенности (критерий информативности, impurity function)
 - измеряет степень неоднозначности y для объектов в узле t .
- Качество разбиения t :

$$\Delta\phi(t) = \phi(t) - \phi(t_L)\frac{N(t_L)}{N(t)} - \phi(t_R)\frac{N(t_R)}{N(t)}$$

Определение критерия ветвления

- Из узла t перейти в:
$$\begin{cases} \text{левого потомка } t_L, & \text{если } x^{\hat{i}_t} \leq \hat{h}_t \\ \text{правого потомка } t_R, & \text{если } x^{\hat{i}_t} > \hat{h}_t \end{cases}$$
- Определим $\phi(t)$ - ф-цию неопределенности (критерий информативности, impurity function)
 - измеряет степень неоднозначности y для объектов в узле t .
- Качество разбиения t :

$$\Delta\phi(t) = \phi(t) - \phi(t_L)\frac{N(t_L)}{N(t)} - \phi(t_R)\frac{N(t_R)}{N(t)}$$

- Оптимизация CART (регрессия, классификация): выбрать признак x_i и порог h , максимизирующие $\Delta\phi(t)$:

$$\hat{i}_t, \hat{h}_t = \arg \max_{i,h} \Delta\phi(t)$$

Функция информативности для регрессии

- пусть $I_t = \{i_1, \dots, i_K\}$ - множество индексов объектов узла t .
Можно определить $\phi(t)$ как

$$\phi(t) = \min_{\hat{y}} \frac{1}{|I_t|} \sum_{i \in I_t} (y_i - \hat{y})^2$$

$$\phi(t) = \min_{\hat{y}} \frac{1}{|I_t|} \sum_{i \in I_t} |y_i - \hat{y}|$$

Функция информативности для регрессии

- пусть $I_t = \{i_1, \dots, i_K\}$ - множество индексов объектов узла t .
Можно определить $\phi(t)$ как

$$\phi(t) = \min_{\hat{y}} \frac{1}{|I_t|} \sum_{i \in I_t} (y_i - \hat{y})^2 = \frac{1}{|I_t|} \sum_{i \in I_t} (y_i - \text{mean}_{i \in I_t}(y_i))^2$$

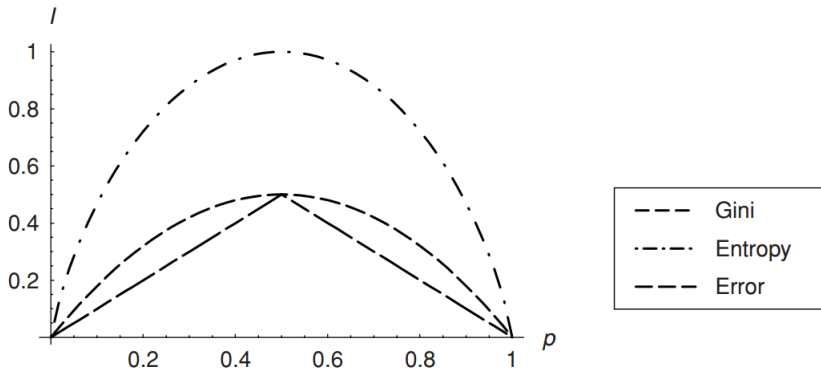
$$\phi(t) = \min_{\hat{y}} \frac{1}{|I_t|} \sum_{i \in I_t} |y_i - \hat{y}| = \frac{1}{|I_t|} \sum_{i \in I_t} |y_i - \text{median}_{i \in I_t}(y_i)|,$$

Функции информативности для классификации

- Классификация:
 - p_1, \dots, p_C - вероятности классов в узле t .
- Функция информативности $\phi(t) = \phi(p_1, p_2, \dots, p_C)$ должна удовлетворять:
 - ϕ определена для $p_j \geq 0$ и $\sum_j p_j = 1$.
 - ϕ достигает максимума при $p_j = 1/C$, $k = 1, 2, \dots, C$.
 - ϕ достигает минимума, когда $\exists j : p_j = 1, p_i = 0 \forall i \neq j$.
 - ϕ симметрична относительно p_1, p_2, \dots, p_C .

Визуализация основных функции информативности

Функции информативности для $y \in \{+1, -1\}$ с $p(y = +1|x) = p$ и $p(y = -1|x) = 1 - p$.



Классификационная ошибка

- **Классификационная ошибка:** как часто ошибаемся при константном прогнозе?

$$\begin{aligned}\phi(t) &= \min_{\hat{y}} \frac{1}{|I_t|} \sum_{i \in I_t} \mathbb{I}[y_i \neq \hat{y}] \\ &= \frac{1}{|I_t|} \sum_{i \in I_t} \mathbb{I}[y_i \neq y_{most.common}] \\ &= 1 - \hat{p}_{max}\end{aligned}$$

Критерий Джини

- Критерий Джини: оценка Бриера¹

$$\begin{aligned}
 \phi(t) &= \min_{p: \sum_c p_c = 1} \frac{1}{|I_t|} \sum_{i \in I_t} \|p - p_i^{true}\|^2 = \\
 &= \min_{p: \sum_c p_c = 1} \frac{1}{|I_t|} \sum_{i \in I_t} \sum_{c=1}^C (p_c - \mathbb{I}[y_i = c])^2 = \\
 &= \sum_{i=1}^C \hat{p}_i (1 - \hat{p}_i) = 1 - \sum_{i=1}^C \hat{p}_i^2
 \end{aligned}$$

- Это вероятность ошибки при случайном угадывании с вероятностями $p(\hat{y} = 1) = \hat{p}_1, \dots, p(\hat{y} = C) = \hat{p}_C$

¹ Докажите оптимальность выборочных оценок вероятностей классов и финальный вид критерия.

Энтропия

- **Энтропия:** -логарифм правдоподобия оптимальных вероятностей классов²

$$\begin{aligned}\phi(t) &= \min_{p: \sum_c p_c = 1} -\frac{1}{|I_t|} \left(\sum_{i \in I_t} \sum_{c=1}^C \ln p_c^{\mathbb{I}[y_i=c]} \right) = \\ &= \min_{p: \sum_c p_c = 1} -\frac{1}{|I_t|} \left(\sum_{i \in I_t} \sum_{c=1}^C \mathbb{I}[y_i = c] \ln p_c \right) = -\sum_{i=1}^C \hat{p}_i \ln \hat{p}_i\end{aligned}$$

- Это среднее количество информации $= -\ln p_y$, которое получаем, узнав класс y .

² Докажите оптимальность выборочных оценок вероятностей классов и финальный вид критерия.

Комментарии

- Логичнее брать в качестве $\phi(t)$ пользовательскую ф-цию потерь (но может не существовать аналит. решения).
- Алгоритм $\hat{i}_t, \hat{h}_t = \arg \max_{i,h} \Delta\phi(t)$ применяется рекурсивно при построении дерева сверху вниз.
 - жадный алгоритм, см. только на 1 шаг вперед (глобально неоптимальный, зато быстрый)
 - можно заглядывать на 2 шага вперед
- Сложность подбора $\phi(t) \leq h$: $O(N^2 D)$
 - D признаков, $O(N)$ значений порога, $\Delta\phi(t)$: $O(N)$.
- Как её можно сократить?

Комментарии

- Логичнее брать в качестве $\phi(t)$ пользовательскую ф-цию потерь (но может не существовать аналит. решения).
- Алгоритм $\hat{i}_t, \hat{h}_t = \arg \max_{i,h} \Delta\phi(t)$ применяется рекурсивно при построении дерева сверху вниз.
 - жадный алгоритм, см. только на 1 шаг вперед (глобально неоптимальный, зато быстрый)
 - можно заглядывать на 2 шага вперед
- Сложность подбора $\phi(t) \leq h$: $O(N^2 D)$
 - D признаков, $O(N)$ значений порога, $\Delta\phi(t)$: $O(N)$.
- Как её можно сократить?
 - экономный пересчет $\phi(t)$
 - при смещении порога 1 объект меняет вершину
 - дискретизация признака
 - квантили: 0.1, 0.2, ... 0.9

Содержание

- 1 Понятие решающего дерева
- 2 Правила спуска
- 3 Выбор критерия ветвления
- 4 Назначение прогнозов листьям**
- 5 Критерий остановки
- 6 Анализ решающих деревьев

Оптимальный прогноз в листьях: регрессия

- Регрессия:

$$\hat{y} = \arg \min_f \sum_{i: x_i \in t} \mathcal{L}(f - y_i)$$

- Например³

$$\mathcal{L}(u) = u^2 : \hat{y} = \text{mean}_{i: x_i \in t} \{y_i\}$$

$$\mathcal{L}(u) = |u| : \hat{y} = \text{median}_{i: x_i \in t} \{y_i\}$$

³ Докажите оптимальность среднего и медианы для соответствующих ф-ций потерь.

Оптимальный прогноз в листьях: классификация

В практических задачах классификации типы ошибок приводят к разным штрафам.

- например, при определении болен пациент или здоров.

Определим матрицу штрафов⁴ $\Lambda \in \mathbb{R}^{C \times C}$, где

$\lambda_{ij} = \text{cost}(\hat{y} = j | y = i)$:

		прогноз		
		$\hat{y} = 1$	\dots	$\hat{y} = C$
факт	$y = 1$	λ_{11}	\dots	λ_{1C}
	\dots	\dots	\dots	\dots
	$y = C$	λ_{C1}	\dots	λ_{CC}

⁴Как эта матрица будет выглядеть в случае единичных потерь за любой тип ошибки?

Оптимальный прогноз в листьях: классификация

В случае общих потерь $\lambda_{ij} = \text{cost}(\hat{y} = j | y = i)$

$$\hat{y} = \arg \min_j \sum_{i \in I_t} \lambda_{y(i),j} = \arg \min_j \sum_{c=1}^C (N_t p_c) \lambda_{cj}$$

В случае $\lambda_{cj} = \lambda \mathbb{I}[c \neq j]$:

$$\begin{aligned} \hat{y} &= \arg \min_j N_t \sum_{c=1}^C p_c \lambda \mathbb{I}[c \neq j] = \arg \min_j \sum_{c \neq j} p_c \\ &= \arg \min_j (1 - p_j) = \arg \max_j p_j \end{aligned}$$

Использование CART для регрессии

```
...  
from sklearn.tree import DecisionTreeRegressor  
from sklearn.metrics import mean_absolute_error  
  
X_train, X_test, Y_train, Y_test =  
    get_demo_regression_data()  
# инициализация, criterion - функция неопределённости:  
model = \  
    DecisionTreeRegressor(criterion='absolute_error')  
model.fit(X_train, Y_train) # обучение модели  
Y_hat = model.predict(X_test) # построение прогнозов  
print(f'Средний модуль ошибки (MAE): \  
{mean_absolute_error(Y_test, Y_hat):.2f}')
```

Больше информации. Полный код.

Использование CART для классификации

```
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score
from sklearn.metrics import brier_score_loss

X_train, X_test, Y_train, Y_test =
    get_demo_classification_data()
# инициализация, criterion - функция неопределённости:
model = DecisionTreeClassifier(criterion='gini')
model.fit(X_train, Y_train) # обучение модели
Y_hat = model.predict(X_test) # построение прогнозов
print(f'Точность прогнозов: \
{100*accuracy_score(Y_test, Y_hat):.1f}%')

P_hat = model.predict_proba(X_test) # вероятности классов
loss = brier_score_loss(Y_test, P_hat[:,1])
print(f'Мера Бриеера ошибки вероятностей: {loss:.2f}')
```

Больше информации. Полный код.

Содержание

- 1 Понятие решающего дерева
- 2 Правила спуска
- 3 Выбор критерия ветвления
- 4 Назначение прогнозов листьям
- 5 Критерий остановки**
 - Остановка, основанная на правилах
 - Алгоритм обрезки в CART
- 6 Анализ решающих деревьев

Критерий остановки

- Сложность модели должна соответствовать сложности данных:
 - слишком глубокие деревья -> переобучение
 - в крайнем случае: 1 лист содержит 1 объект, нет обобщающей способности.
 - слишком мелкие деревья -> недообучение
- Необходимо выбрать оптимальную глубину при построении дерева.
- Подходы к остановке построения:
 - основанные на правилах
 - обрезка деревьев (pruning)

5 Критерий остановки

- Остановка, основанная на правилах
- Алгоритм обрезки в CART

Остановка, основанная на правилах

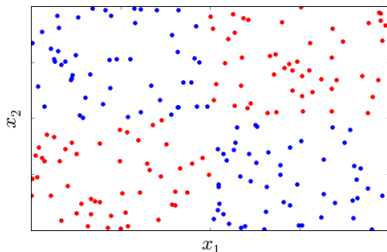
- Остановка, когда критерий больше порога.
- Варианты критерия:
 - глубина дерева
 - #объектов в узле
 - минимальное #объектов в дочерних узлах после разбиения
 - значение информативности $\phi(t)$
 - изменение информативности $\Delta\phi(t)$ после разбиения

Анализ подхода

Преимущества: простота, интерпретируемость.

Недостатки:

- нужно подбирать порог
- изменение информативности $\Delta\phi(t)$ неоптимально: последующие разбиения могут привести к большему $\Delta\phi(t)$:



5 Критерий остановки

- Остановка, основанная на правилах
- Алгоритм обрезки в CART

Обрезка деревьев

Поскольку ранняя остановка по $\Delta\phi(t)$ может давать неоптимальный результат, часто:

- 1 Дерево строится до самого низа.
- 2 Потом лишние ветви обрезаются (tree pruning)

Простой подход - обрезать всевозможные деревья по валидации.

- нужно перебирать все поддеревья
- эффективнее использовать направленную обрезку с помощью алгоритма *минимального штрафа за сложность* (minimal cost-complexity pruning)

Алгоритм минимального штрафа за сложность

Алгоритм минимального штрафа за сложность:

- Определим:

- A_t поддерево с корнем t , \tilde{A}_t - его листья
- $R(A_t)$ - мера ошибок дерева A_t
 - #ошибок классификации либо сумма квадратов ошибок
- $R_\alpha(T)$ - со штрафом за сложность ($+\alpha$ за каждый лист).

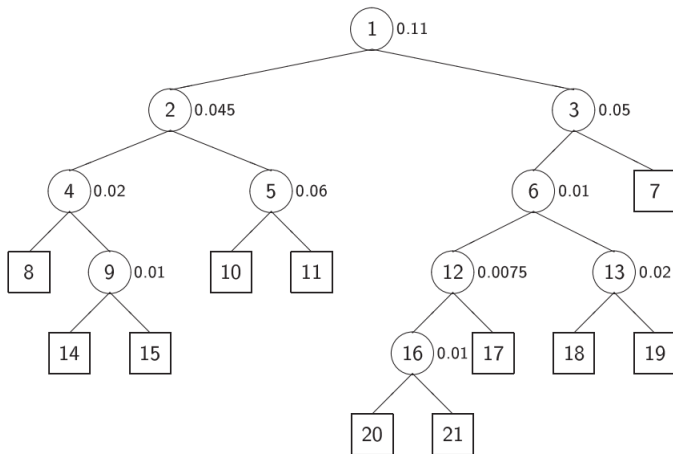
потери за ошибки : $R(A_t) = \sum_{\tau \in \tilde{A}_t} R(\tau)$

ошибки и сложность: $R_\alpha(A_t) = R(A_t) + \alpha |\tilde{T}|$

- Целесообразность построения A_t вместо t определим по равновесному α из $R_\alpha(A_t) = R_\alpha(t)$:

$$R(A_t) + \alpha |\tilde{A}_t| = R(t) + \alpha \Rightarrow \alpha_t = \frac{R(t) - R(A_t)}{|\tilde{A}_t| - 1}$$

Пример вычисления α_t



Алгоритм обрезки

- 1 Строим дерево T до самого низа (пока в листьях не останутся объекты с одинаковым откликом).
- 2 Строим систему вложенных поддеревьев $T = T_0 \supset T_1 \supset \dots \supset T_{|T|}$ содержащих $|T|, |T| - 1, \dots, 1$ узлов, повторяя процедуру:
 - заменить T_t с самым малым α_t её корнем t
 - пересчитать α_t для всех предков t .
- 3 Выберем T_i , дающее минимальные потери на валидационной выборке.

Обработка пропущенных значений

Если проверяемый признак отсутствует:

- заполнить пропуски:
 - вещественные: средним, медианой
 - категориальные: модой или значением "пропущено"
- Можно предсказывать пропущенные значения по др. признакам (использовано в CART)
- C4.5: спускаемся из t по всем дочерним вершинам t_1, \dots, t_S , потом усредняем прогнозы с весами:

$$N(t_1)/N(t), N(t_2)/N(t), \dots N(t_S)/N(t)$$

Важность признаков: mean decrease in impurity

- Важность признаков по изменению критерия информативности (mean decrease in impurity, MDI).

Важность признаков: mean decrease in impurity

- Важность признаков по изменению критерия информативности (mean decrease in impurity, MDI).
 - рассмотрим признак f
 - пусть $T(f)$ -множество всех вершин, использующих f в функции ветвления
 - эффективность разбиения в t :

$$\Delta\phi(t) = \phi(t) - \sum_{c \in \text{children}(t)} \frac{N(c)}{N(t)} \phi(c)$$

- значимость f :

$$\frac{1}{N} \sum_{t \in T(f)} N(t) \Delta\phi(t)$$

- Поощряет признаки с большим количеством уникальных значений.

Важность признаков: mean decrease in impurity

В sklearn:

- важность содержится с *model.feature_importances_*
- недостатки:
 - вычисляется на обучающей выборке
 - если модель переобучается на признаке, важность высока, но вклад в точность прогнозов мал.

Содержание

- 1 Понятие решающего дерева
- 2 Правила спуска
- 3 Выбор критерия ветвления
- 4 Назначение прогнозов листьям
- 5 Критерий остановки
- 6 Анализ решающих деревьев**

Преимущества решающих деревьев

- нелинейная модель с гибкой настройкой сложности
- вычислительная эффективность прогнозов
- интерпретируемость (для неглубоких деревьев)
- встроенный отбор признаков
- расчет важности признаков
- инвариантны к монотонным преобразованиям признаков
- работают с признаками разной природы
 - бинарные, вещественные, порядковые
 - категориальные \rightarrow бинарные (one-hot) или вещественные (mean-value encoding)
 - категориальные \rightarrow порядковые, упорядочив категории по \bar{y} при условии категории в рамках узла

Недостатки решающих деревьев

- нет динамической подстройки под потоковые данные
- сравнительно невысокая точность:
 - границы перпендикулярно осям признаков
 - "жадная" настройка сверху вниз глобально неоптимальна
 - точность повышается композицией решающих деревьев