Деревья решений и случайный лес

Горюнов Егор

yegor_goryunov@vk.com

17.10.2023

Дерево решений

Пусть задано *бинарное дерево*, в котором:

- каждой внутренней вершине v прописан предикат $B_v : \mathbb{X} \to \{0, 1\}$;
- каждой листовой вершине v приписан прогноз $c_v \in \mathbb{Y}$, где $\mathbb{Y}$ — область значений целевой переменной (в случае классификации листу может быть также приписан вектор вероятностей классов).

В ходе предсказания осуществляется проход по этому дереву к некоторому листу.

Как правило, в качестве предиката B_v используют сравнение с порогом $t \in \mathbb{R}$ по какому-то j-му признаку:

$$B_v(x, j, t) = [x_j \leq t]$$

Жадный алгоритм построения решающего дерева

Пусть X — исходное множество объектов, а X_m — множество объектов, попавших в текущую вершину. Тогда жадный алгоритм можно описать следующим образом:

1. Создаём вершину v.
2. Если выполнен критерий остановки, объявляем вершину листом, иначе продолжаем.
3. Находим предикат $B_{j,t}$, который соответствует наилучшему разбиению X_m на две подвыборки X_ℓ и X_r.
4. Рекурсивно повторяем процедуру для X_ℓ и X_r, максимизируя *критерий ветвления*.

Критерий ветвления для классификации

На множестве объектов X_m можем получить частотные оценки вероятностей классов $P(y|X_m)$:

$$p_y = \frac{1}{|X_m|} \sum_{x_i \in X_m} [y_i = y]$$

Пусть $H(X_m)$ — мера непределённости (impurity) распределения p_y на множестве X_m. Будем искать такое разбиение, чтобы снизить неопределённость в узлах:

$$H(X_m) = \sum_{y \in Y} p_y L(p_y) \to \min$$

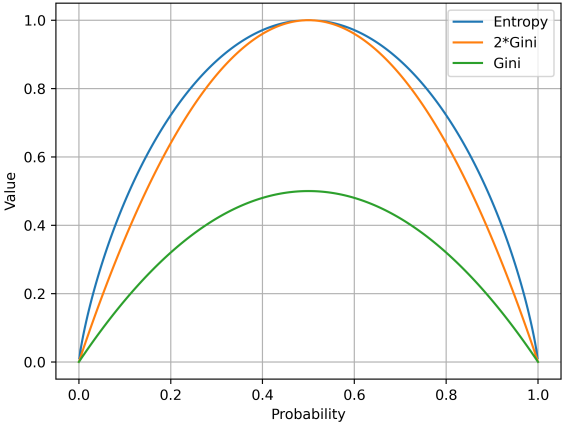
Тогда выигрыш от разбиения в вершине v предикатом B_v определяется, как:

$$\text{Gain}(B_v, X_m) = H(X_m) - \frac{|X_l|}{|X_m|} H(X_l) - \frac{|X_r|}{|X_m|} H(X_r) \to \max$$

Энтропия и критерий Джини

В качестве impurity для классификации можно взять:

- Если $L(p) = -\log_2 p$, то $H(X_m) = -\sum p_y \log_2 p_y$ — энтропия выборки.
- Если $L(p) = 1 - p$, то $H(X_m) = \sum p_y(1 - p_y)$ — неопределённость Джини (Gini impurity).



Особенности жадной стратегии

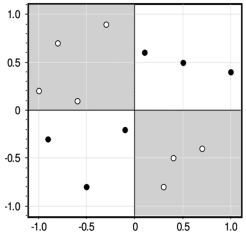
Достоинства:

- Интерпретируемость модели и простота классифкации
- Допустимы разнотипные данные и данные с пропусками
- Эффективность алгоритма по по времени

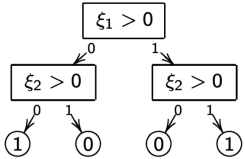
Недостатки:

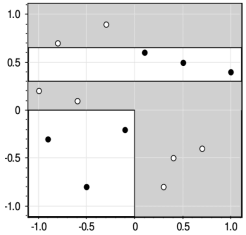
- Переусложнение структуры дерева и переобучение
- Высокая чувствительность к шуму, составу выборки

Проблема XOR

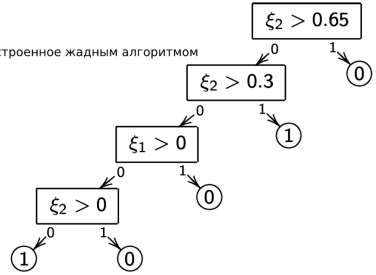


Оптимальное дерево





Дерево,построенное жадным алгоритмом



Регуляризация решающих деревьев

Деревья легко переобучаются, поэтому процесс ветвления необходимо вовремя останавливать. Для этого используют критерии:

- ограничение по максимальной глубине дерева;
- ограничение на минимальное количество объектов в листе;
- ограничение на максимальное количество листьев в дереве;
- требование, чтобы функционал качества при делении подвыборки на две улучшался не менее чем на s процентов.

Это можно делать на разных этапах работы алгоритма:

- проверять критерии прямо во время построения дерева (**pre-pruning**);
- построить дерево жадно без ограничений, а затем провести **стрижку** (**pruning**) ;

Пример. Деревья решений с разной глубиной

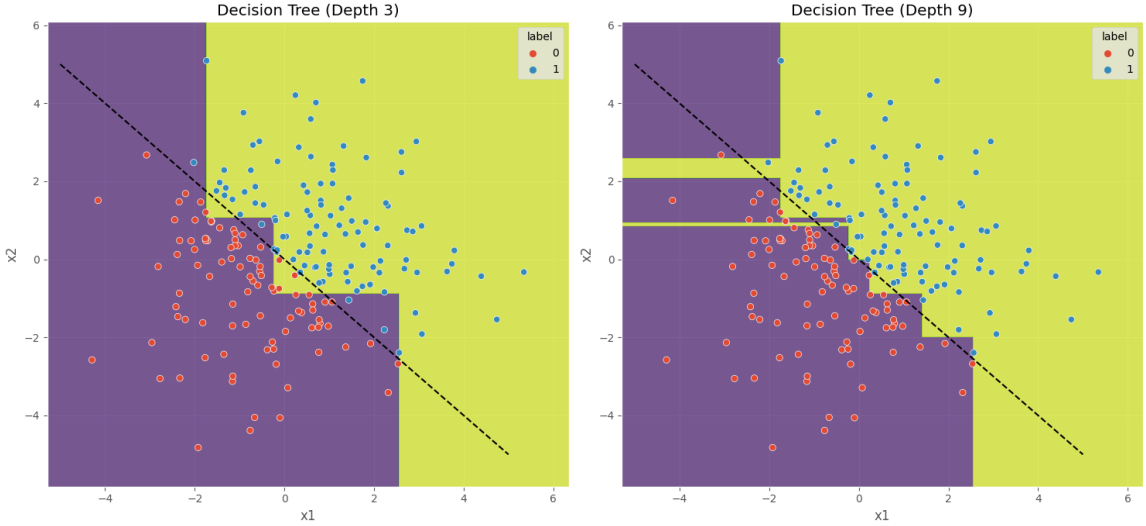


Рис.: Проблема разделения по наклонной

Решающее дерево для регрессии

Обобщение на случай регрессии: $Y = \mathbb{R}$, $y_v \in \mathbb{R}$. Неопределённость в вершине — среднеквадратическая ошибка:

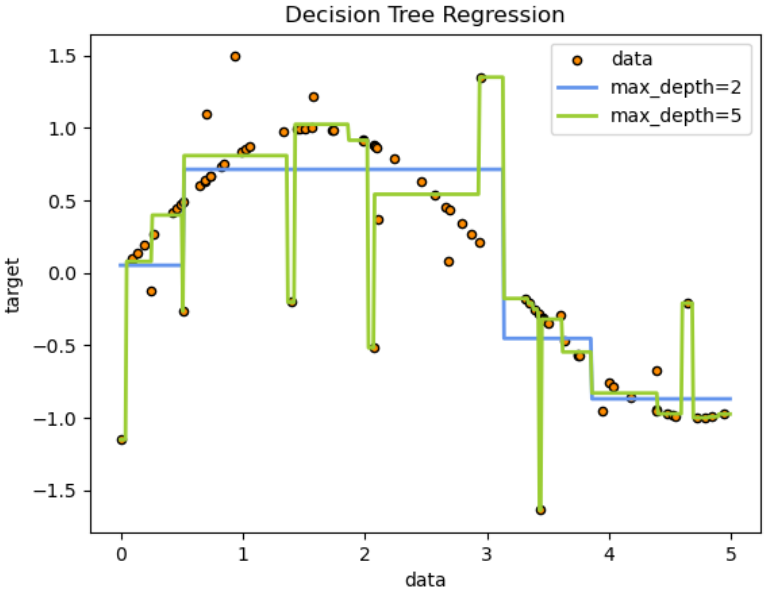
$$H(X_m) = \frac{1}{|X_m|} \min_{c \in Y} \sum_{x_i \in X_m} (y_i - c)^2$$

Тогда значение y_v в вершине v равно среднему подвыборки (по МНК):

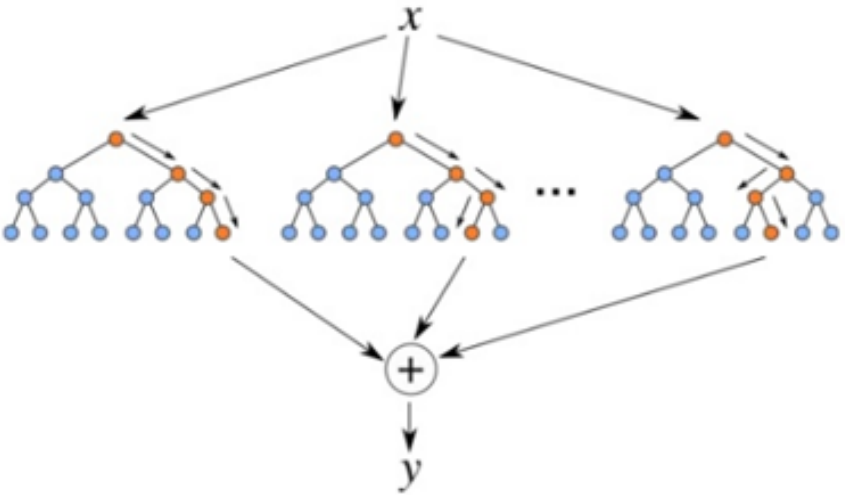
$$y_v = \frac{1}{|X_m|} \sum_{x_i \in X_m} y_i$$

Следовательно, дерево решений $a(x)$ — это кусочно-постоянная функция.

Пример. Зависимость дерева регрессии от глубины



Случайный лес



Бэггинг

Метод **бэггинга** (*bagging, bootstrap aggregation*) состоит в следующем: из исходной обучающей выборки длины N формируются различные обучающие подвыборки той же длины N с помощью *бутстрепа* — случайного выбора с возвращениями. Базовые алгоритмы, обученные по подвыборкам, объединяются в композицию с помощью простого голосования.

Оценим долю выбранных объектов начальной выборки после бутстрапирования.
p — вероятность попасть в подвыборку,
q — вероятность не быть выбраным.
$p_1 = \frac{1}{N}$ — вероятность выбора объекта на каждом шаге.

$$p = 1 - q = 1 - (1 - p_1)^N = 1 - \left(1 - \frac{1}{N}\right)^N$$

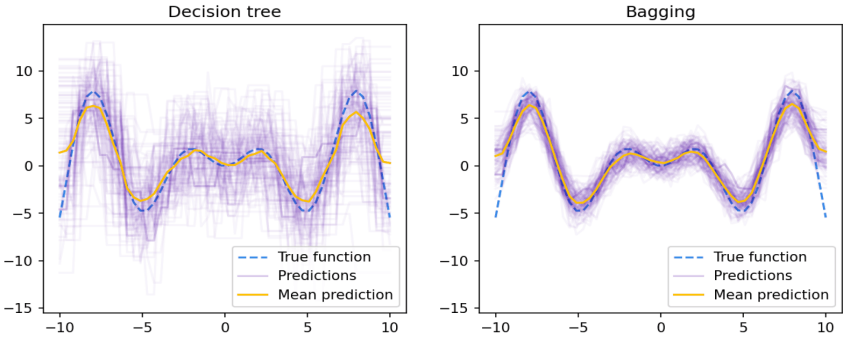
$$\left(1 - \frac{1}{N}\right)^N \xrightarrow{N\to\infty} \frac{1}{e} \implies p \to 1 - \frac{1}{e} \approx 0,632$$

Пример. Бэггинг над решающими деревьями

$b_i(x)$ — базовые модели, обученные на подвыборках.
$a(x)$ — модель бэггинга над моделями $b_i(x)$.

$$a(x) = \frac{1}{T}(b_1(x) + \ldots + b_T(x))$$

Сравним предсказания решающих деревьев глубины 7 и бэггинга над такими деревьями в зависимости от обучающей выборки.

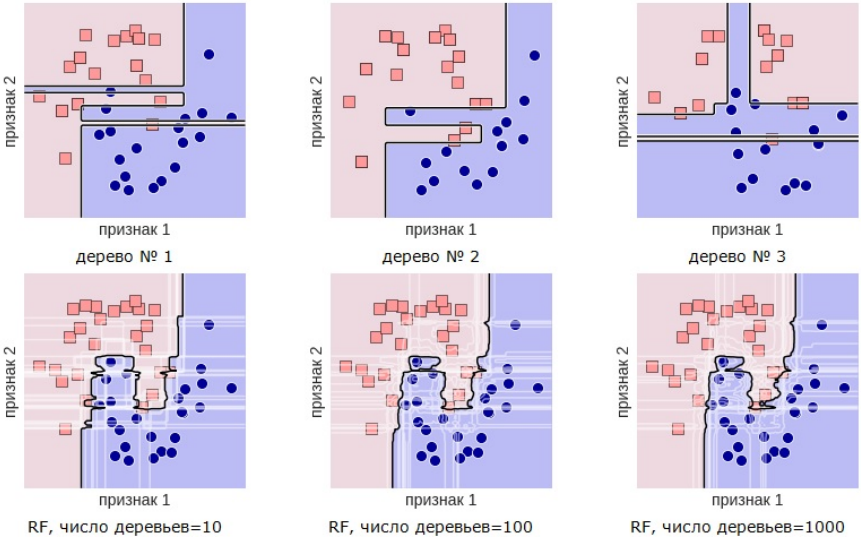


Random forest

Случайный лес (*Random forest*) — это ансамблевая модель над деревьями, сочетающая в себе *бэггинг* и *метод случайных подпространств*.
Метод случайных подпространств: В процессе обучения каждого дерева **в каждой вершине** случайно выбираются $k < N$ признаков для поиска оптимального разбиения. Это позволяет управлять степенью скоррелированности алгоритмов. По умолчанию $k = \lfloor N/3 \rfloor$ для регрессии, $k = \lfloor \sqrt{N} \rfloor$ для классификации.
Гиперпараметры:

- Число T деревьев
- Число k случайно выбираемых признаков
- Максимальная глубина деревьев
- Минимальное число объектов в расщепляемой выборке
- Минимальное число объектов в листьях
- Критерий расщепления.

Пример. Сглаживание разделяющей поверхности



Случайный лес: преимущества и недостатки

Преимущества:

- Простая реализация и простое распараллеливание
- Возможность оценивания важности признаков
- Хорошо работает "из коробки"
- Редко переобучается. Добавление деревьев часто только улучшает модель.
- Хорошо работает с пропущенными данными
- Имеет высокую точность предсказания. RF — один из лучших универсальных методов в ML.

Недостатки:

- Требуется очень много базовых алгоритмов
- Результаты сложнее интерпретировать
- Не обладает возможностью экстраполяции

Обсудим?