

Модификации градиентного бустинга

Кантонистова Е.О.

ВШЭ, 2019

The background is a light green gradient. In the corners, there are decorative circuit-like patterns. The top-left and bottom-left corners feature dark blue lines, while the top-right and bottom-right corners feature light blue lines. These lines form various geometric shapes, including circles and rectangles, resembling a stylized circuit board.

РАБОТА С ПРИЗНАКАМИ

КОДИРОВАНИЕ КАТЕГОРИАЛЬНЫХ ПРИЗНАКОВ: ONE-HOT ENCODING

- Предположим, категориальный признак $f_j(x)$ принимает t различных значений: C_1, C_2, \dots, C_t .

Пример: еда может быть *горькой, сладкой, солёной или кислой* (4 возможных значения признака).

КОДИРОВАНИЕ КАТЕГОРИАЛЬНЫХ ПРИЗНАКОВ: ONE-HOT ENCODING

- Предположим, категориальный признак $f_j(x)$ принимает t различных значений: C_1, C_2, \dots, C_t .

Пример: еда может быть *горькой, сладкой, солёной или кислой* (4 возможных значения признака).

- Заменяем категориальный признак на t бинарных признаков: $b_i(x) = [f_j(x) = C_i]$ (индикатор события).

Тогда One-Hot кодировка для нашего примера будет следующей:

горький = (1,0,0,0), *сладкий* = (0,1,0,0),

солёный = (0,0,1,0), *кислый* = (0,0,0,1).

ХЭШИРОВАНИЕ ПРИЗНАКОВ

- Возьмем некоторую функцию (hash-функция), которая переводит значения категориального признака в числа от 1 до B : $h: U \rightarrow \{1, 2, \dots, B\}$.
- То есть для каждого объекта:

$$g_j(x) = [h(f(x)) = j], j = 1, \dots, B$$

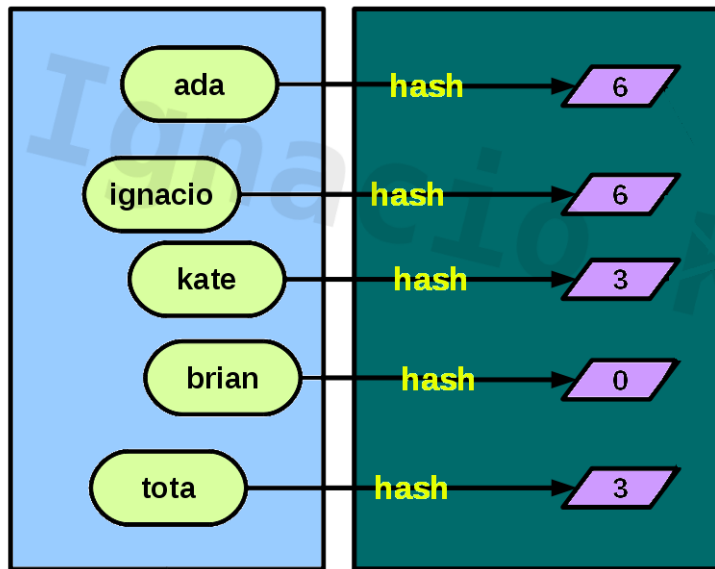
ХЭШИРОВАНИЕ ПРИЗНАКОВ

- Возьмем некоторую функцию (hash-функция), которая переводит значения категориального признака в числа от 1 до B : $h: U \rightarrow \{1, 2, \dots, B\}$.
- То есть для каждого объекта:

$$g_j(x) = [h(f(x)) = j], j = 1, \dots, B$$

Идея: хэширование группирует значения признака. Так как часто встречающихся значений немного, они редко попадают в одну группу при группировке.

ХЭШИРОВАНИЕ ПРИЗНАКОВ



elements

hash function

0	1	2	3	4	5	6
0	0	0	0	0	0	1
0	0	0	0	0	0	1
0	0	0	1	0	0	0
1	0	0	0	0	0	0
0	0	0	1	0	0	0

ХЭШИРОВАНИЕ ПРИЗНАКОВ

- Возьмем некоторую функцию (hash-функция), которая переводит значения категориального признака в числа от 1 до B : $h: U \rightarrow \{1, 2, \dots, B\}$.
- То есть для каждого объекта:

$$g_j(x) = [h(f(x)) = j], j = 1, \dots, B$$

+ позволяет закодировать любое значение категориального признака (в том числе, то, которого не было в тренировочной выборке)

ХЭШИРОВАНИЕ

- Хороший способ работать с категориальными данными, принимающими множество различных значений
- Хорошие результаты на практике
- Позволяет понизить размерность пространства признаков с незначительным снижением качества

Статья про хэширование:

<https://arxiv.org/abs/1509.05472>

СЧЁТЧИКИ

- Пусть целевая переменная y принимает значения от 1 до K .
- Закодируем категориальную переменную $f(x)$ следующим способом:

$$counts(u, X) = \sum_{(x,y) \in X} [f(x) = u]$$

$$successes_k(u, X) = \sum_{(x,y) \in X} [f(x) = u][y = k], k = 1, \dots, K$$

СЧЁТЧИКИ: ПРИМЕР

city	target	0	1	2
Moscow	1	$1/4$	$1/2$	$1/4$
London	0	$1/2$	0	$1/2$
London	2	$1/2$	0	$1/2$
Kiev	1	$1/2$	$1/2$	0
Moscow	1	$1/4$	$1/2$	$1/4$
Moscow	0	$1/4$	$1/2$	$1/4$
Kiev	0	$1/2$	$1/2$	0
Moscow	2	$1/4$	$1/2$	$1/4$

СЧЁТЧИКИ

- Пусть целевая переменная y принимает значения от 1 до K .
- Закодируем категориальную переменную $f(x)$ следующим способом:

$$counts(u, X) = \sum_{(x,y) \in X} [f(x) = u]$$

$$successes_k(u, X) = \sum_{(x,y) \in X} [f(x) = u][y = k], k = 1, \dots, K$$

Тогда кодировка:

$$g_k(x, X) = \frac{successes_k(f(x), X) + c_k}{counts(f(x), X) + \sum_{i=1}^K c_i} \approx p(y = k | f(x)),$$

c_i - чтобы не было деления на 0.

СЧЁТЧИКИ: ОПАСНОСТЬ ПЕРЕОБУЧЕНИЯ

Вычисляя счётчики, мы закладываем в признаки информацию о целевой переменной y , тем самым, переобучаемся!

СЧЁТЧИКИ: КАК ВЫЧИСЛЯТЬ

- Можно вычислять счётчики так:

city	target	
Moscow	1	Вычисляем счетчики по этой части
London	0	
London	2	
Kiev	1	
Moscow	1	Кодируем признак вычисленными счётчиками и обучаемся по этой части
Moscow	0	
Kiev	0	
Moscow	2	

СЧЁТЧИКИ: КАК ВЫЧИСЛЯТЬ

Более продвинутый способ (по кросс-валидации):

1) Разбиваем выборку

на m частей X_1, \dots, X_m

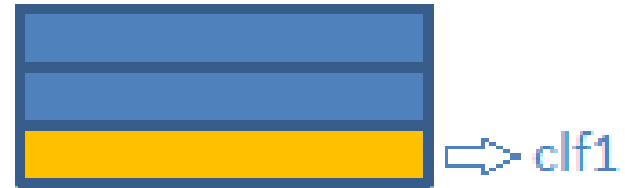
2) На каждой части X_i

значения признаков

вычисляются по

оставшимся частям:

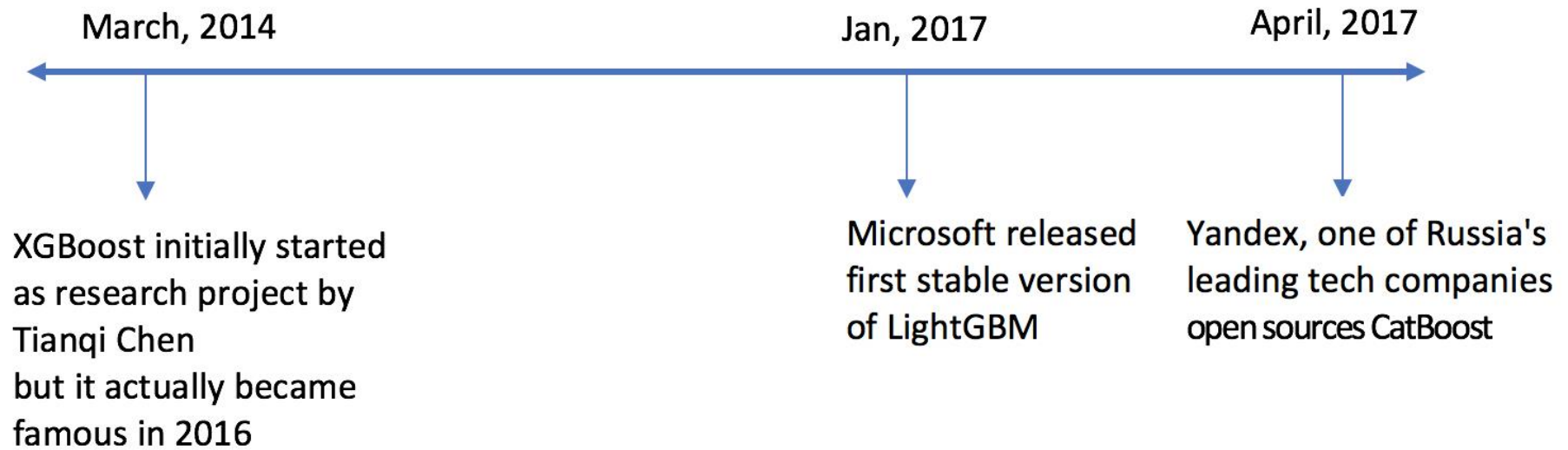
$$x \in X_i \Rightarrow g_k(x) = g_k(x, X \setminus X_i)$$



РЕАЛИЗАЦИИ ГРАДИЕНТНОГО БУСТИНГА

- Xgboost
- CatBoost
- LightGBM

XGBOOST, LIGHTGBM, CATBOOST



- <https://github.com/dmlc/xgboost>
- <https://github.com/Microsoft/LightGBM>
- <https://towardsdatascience.com/catboost-vs-light-gbm-vs-xgboost-5f93620723db>

XGBOOST (EXTREME GRADIENT BOOSTING)

- На каждом шаге градиентного бустинга решается задача

$$\sum_{i=1}^l (b(x_i) - s_i)^2 \rightarrow \min_b$$

$$\Leftrightarrow \sum_{i=1}^l \left(-s_i b(x_i) + \frac{1}{2} b^2(x_i) \right)^2 \rightarrow \min_b$$

- На каждом шаге xgboost решается задача

$$\sum_{i=1}^l \left(-s_i b(x_i) + \frac{1}{2} h_i b^2(x_i) \right) + \gamma J + \frac{\lambda}{2} \sum_{j=1}^J b_j^2 \rightarrow \min_b, \quad (*)$$

$$h_i = \frac{\partial^2 L}{\partial z^2} \Big|_{a_{N-1}(x_i)}$$

XGBOOST

$$\sum_{i=1}^l \left(-s_i b(x_i) + \frac{1}{2} h_i b^2(x_i) \right) + \gamma J + \frac{\lambda}{2} \sum_{j=1}^J b_j^2 \rightarrow \min_b$$

Основные особенности xgboost:

- базовый алгоритм приближает направление, посчитанное с учетом второй производной функции потерь
- функционал регуляризуется – добавляются штрафы за количество листьев и за норму коэффициентов
- при построении дерева используется критерий информативности, зависящий от оптимального вектора сдвига
- критерий останова при обучении дерева также зависит от оптимального сдвига

CATBOOST

CatBoost – алгоритм, разработанный в Яндексе. Он является оптимизацией Xgboost и в отличие от Xgboost умеет обрабатывать категориальные признаки.

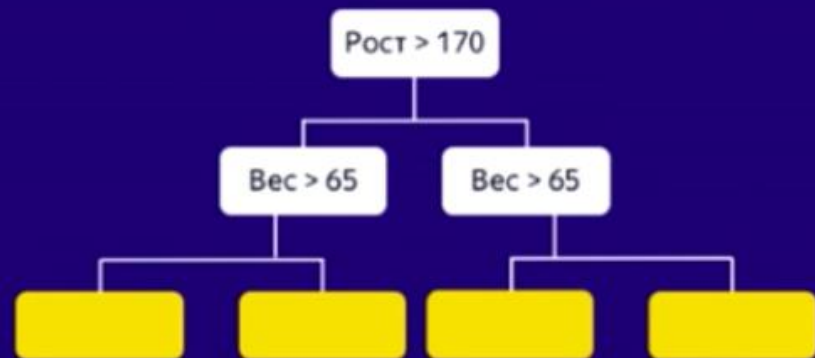
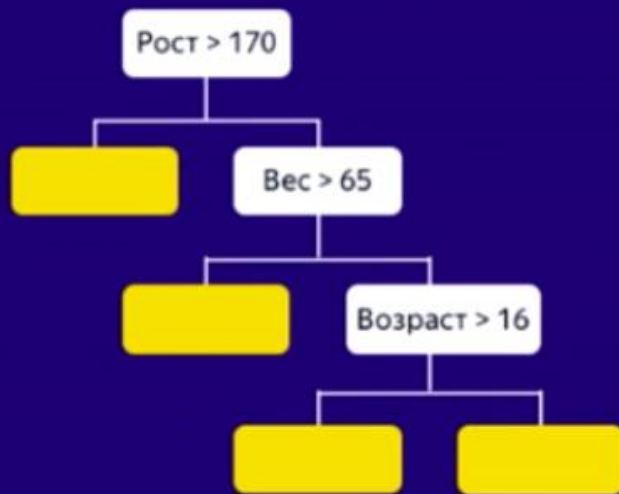
<https://github.com/catboost/catboost>

CATBOOST

Особенности catboost:

- используются симметричные деревья решений

Симметричные деревья



CATBOOST

Особенности catboost:

- Для кодирования категориальных признаков используется набор методов (one-hot encoding, счётчики, комбинации признаков и др.)

Статистики по категориальным факторам

- › One-hot кодирование
- › Статистики без использования таргета
- › Статистики по случайным перестановкам
- › Комбинации факторов

прошлое		SDE		1
		SDE		1
		SDE		0
		PR		
i		SDE		1
		PR		

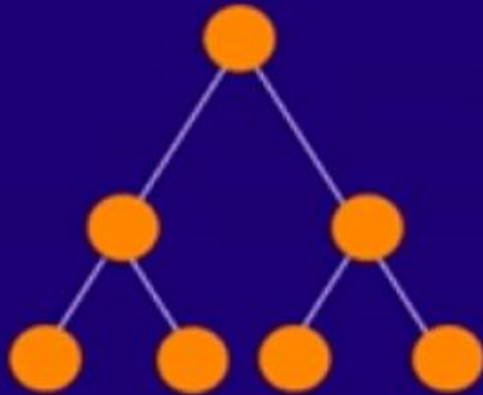
$$i \rightarrow \frac{1+1+0}{3}$$

CATBOOST

Особенности catboost:

- динамический бустинг

Динамический бустинг



$$\text{leafValue}(\text{doc}) = \sum_{i=1}^{\text{doc}} \frac{g(\text{approx}(i), \text{target}(i))}{\text{docs in the past}}$$

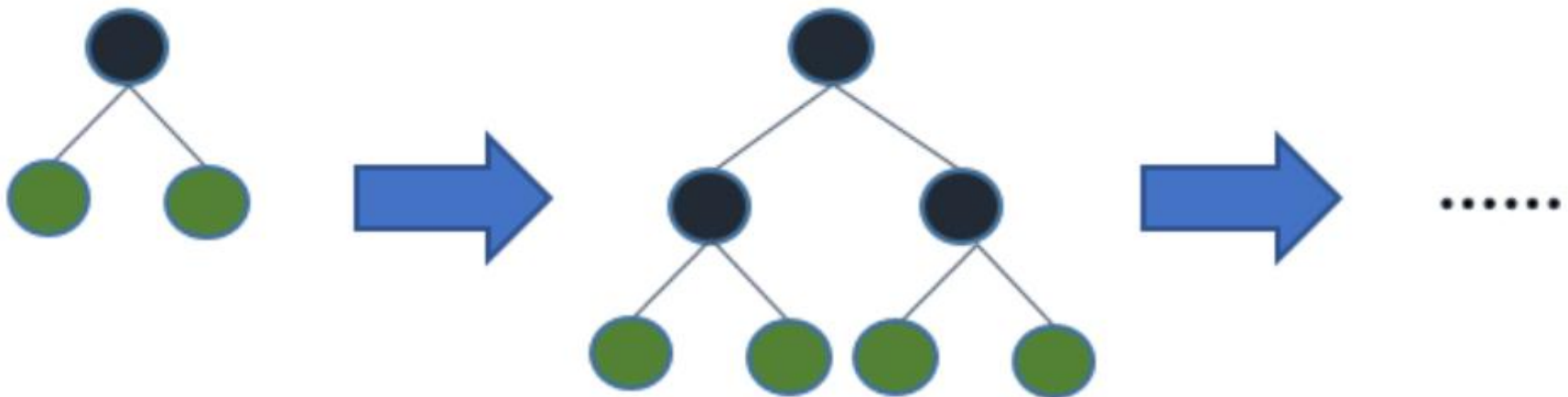
CATBOOST

Бонусы реализации:

- Поддержка пропусков в данных
- Обучается быстрее, чем xgboost
- Показывает хороший результат даже без подбора параметров
- Удобные методы: проверка на переобученность, вычисление значений метрик, удобная кросс-валидация и др.

LIGHTGBM

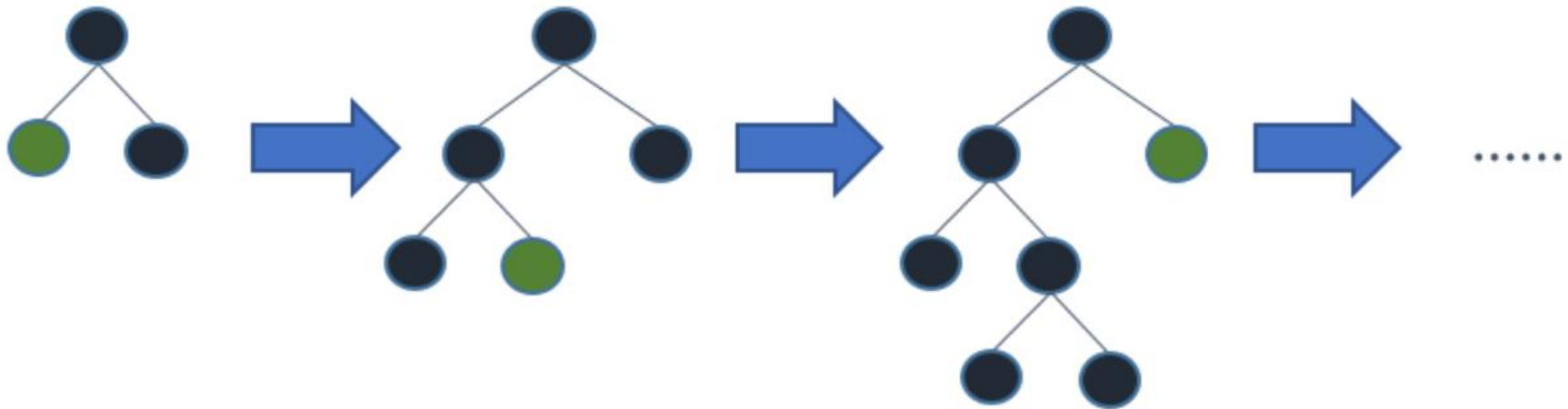
В других реализациях градиентного бустинга деревья строятся по уровням:



Level-wise tree growth

LIGHTGBM

LightGBM строит деревья, добавляя на каждом шаге один лист:



Leaf-wise tree growth

Такой подход позволяет добиться более высокой точности решения задачи оптимизации.

LIGHTGBM

Кодирование категориальных признаков.

- LightGBM разбивает значения категориального признака на два подмножества в каждой вершине дерева, находя при этом наилучшее разбиение
- Если категориальный признак имеет k различных значений, то возможных разбиений $2^{k-1} - 1$. В LightGBM реализован способ поиска оптимального разбиения за $O(k \log k)$ операций.

LIGHTGBM

Ускорение построения деревьев за счёт бинаризации признаков:

2	3	5	9	11	12	16
---	---	---	---	----	----	----



1	1	1	1	2	2	2
---	---	---	---	---	---	---

split

An example of how binning can reduce the number of splits to explore. The features must be sorted in advance for this method to be effective.