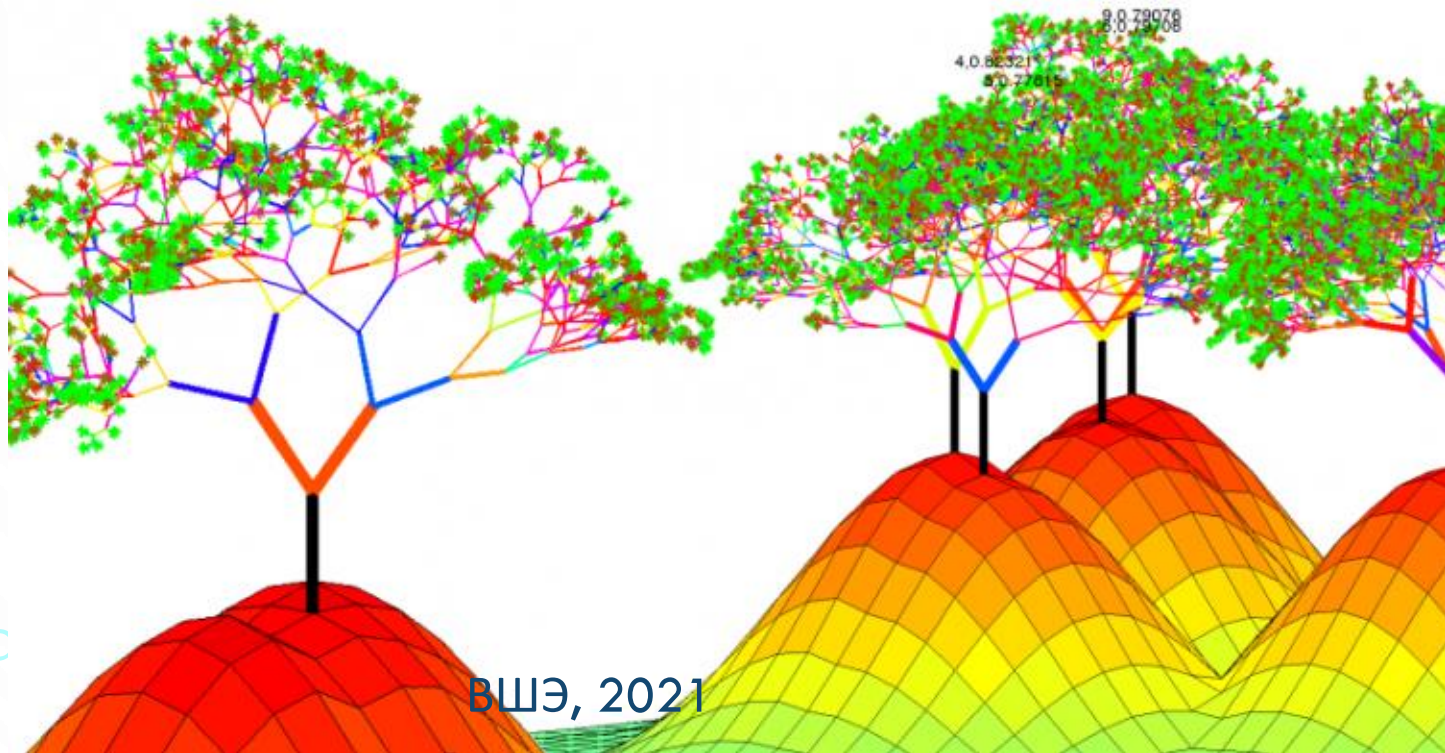


Занятие 11

Бустинг.

Елена Кантонистова

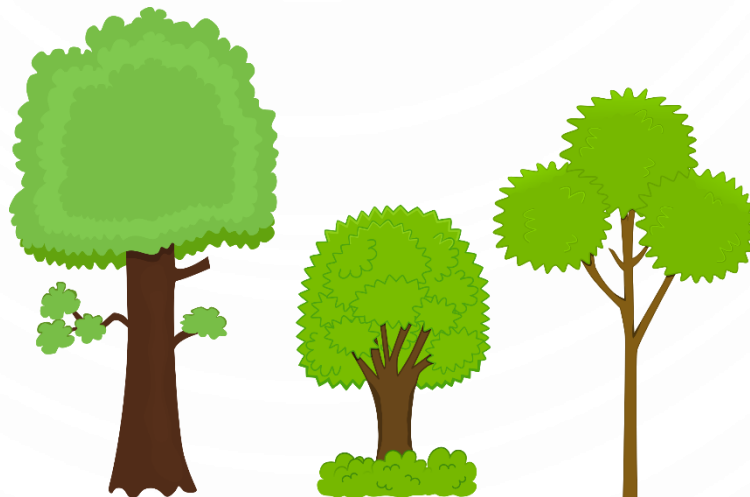
elena.kantonistova@yandex.ru



ВШЭ, 2021

СЛУЧАЙНЫЙ ЛЕС (RANDOM FOREST)

- Возьмем в качестве базовых алгоритмов для бэггинга **решающие деревья**, т.е. каждое случайное дерево $b_i(x)$ построено по своей бутстрепной подвыборке X_i .
- В каждой вершине дерева будем искать **разбиение не по всем признакам, а по подмножеству признаков**.
- Итоговая композиция имеет вид $a(x) = \frac{1}{N} \sum_{j=1}^N b_j(x)$.



СМЕЩЕНИЕ И РАЗБРОС У БЭГГИНГА

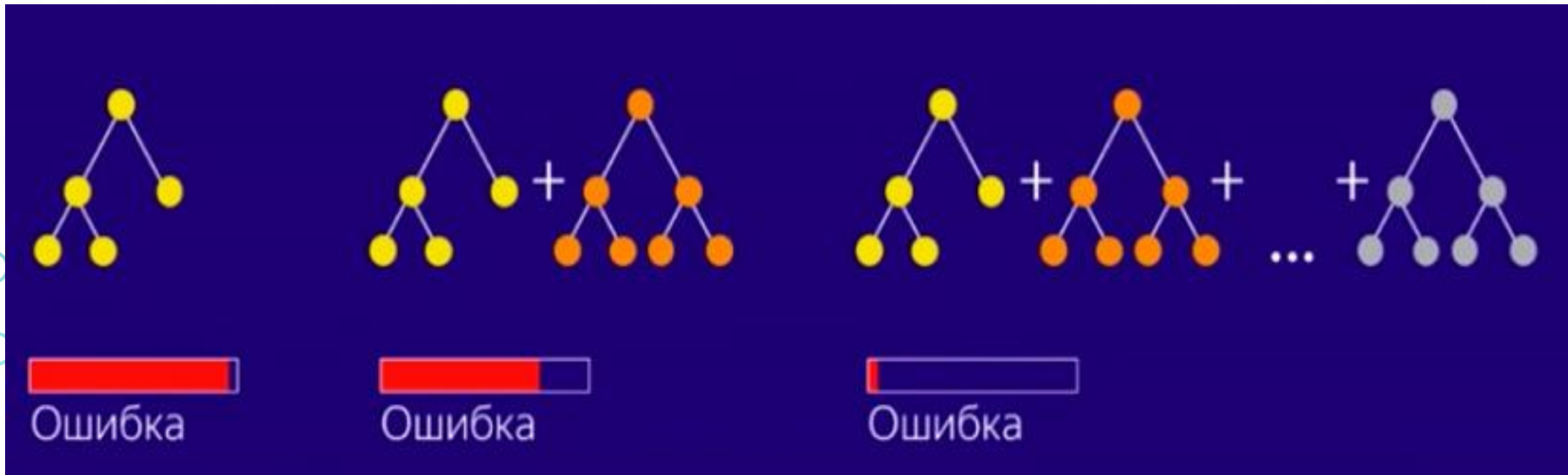
Утверждение.

1) **Бэггинг не ухудшает смещенность модели, т.е. смещение $a_N(x)$ равно смещению одного базового алгоритма.**

2) **Если базовые алгоритмы некоррелированы, то дисперсия бэггинга $a_N(x)$ в N раз меньше дисперсии отдельных базовых алгоритмов.**

БУСТИНГ

Идея: строим набор алгоритмов, каждый из которых исправляет ошибку предыдущих.



ГРАДИЕНТНЫЙ БУСТИНГ

Пусть $L(y, z)$ – произвольная дифференцируемая функция потерь.
Строим алгоритм $a_N(x)$ вида

$$a_L(x) = \sum_{n=1}^L \gamma_n b_n(x),$$

где на N -м шаге

$$b_N(x) = \operatorname{argmin}_{b \in A} \sum_{i=1}^l \left(b(x_i) - s_i^{(N)} \right)^2,$$

$$s_i^{(N)} = -\frac{\partial L}{\partial z}$$

Коэффициент γ_N должен минимизировать ошибку:

$$\gamma_N = \min_{\gamma \in \mathbb{R}} \sum_{i=1}^l L(y_i, a_{N-1}(x_i) + \gamma_N b_N(x_i))$$

СМЕЩЕНИЕ И РАЗБРОС БУСТИНГА

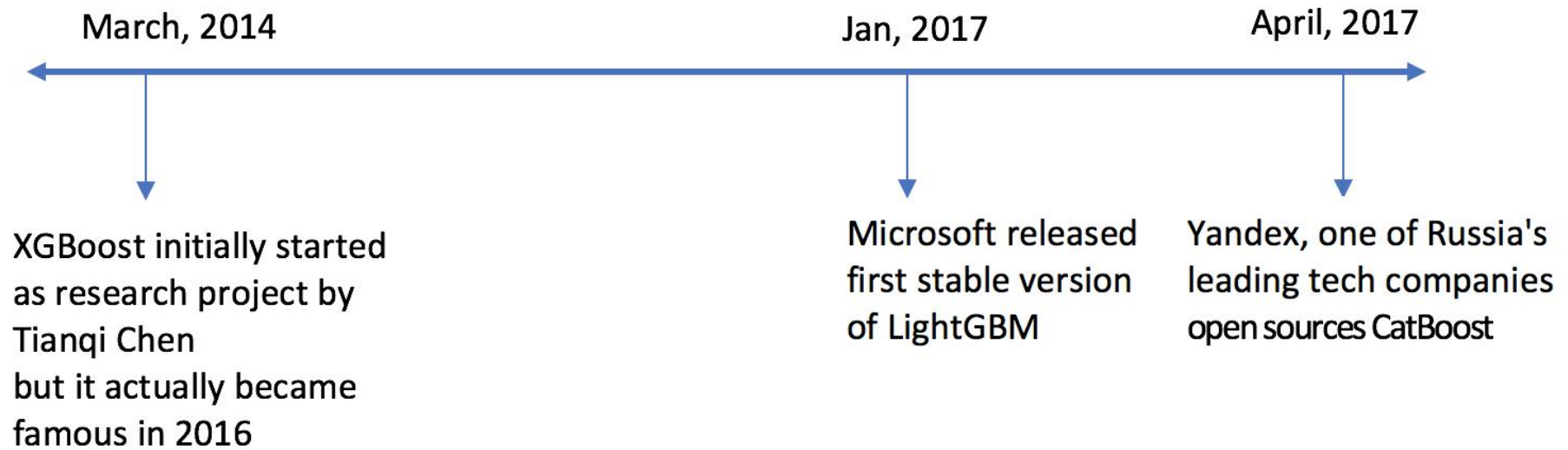
- Бустинг целенаправленно уменьшает ошибку, т.е. смещение у него маленькое.
- Алгоритм получается сложным, поэтому разброс может быть большим.

Значит, чтобы не переобучиться, в качестве базовых алгоритмов надо брать неглубокие деревья (глубины 3-6).

ИМПЛЕМЕНТАЦИИ 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

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

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.

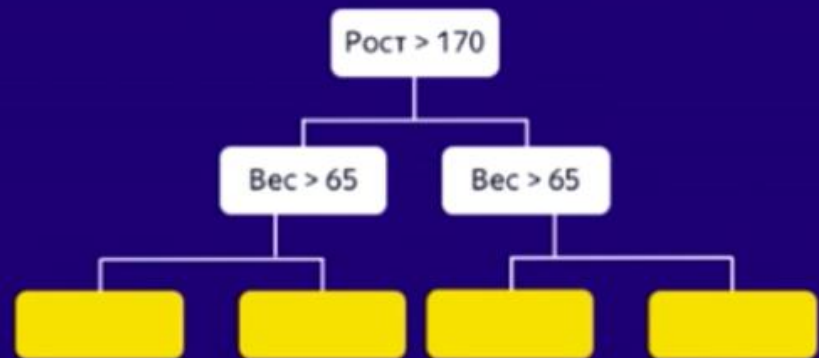
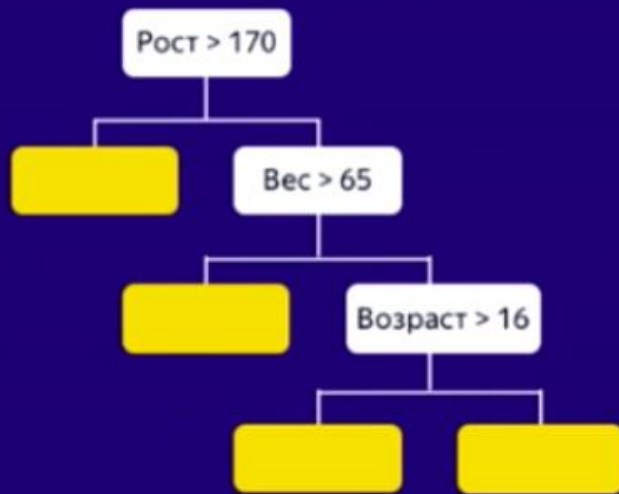
<https://github.com/catboost/catboost>

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		

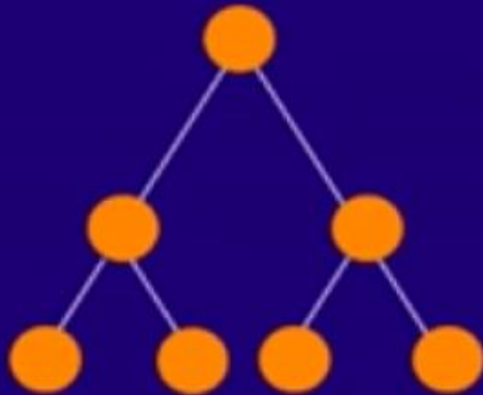
$$i \rightarrow \frac{1+1+0}{3}$$

CATBOOST

Особенности catboost:

- динамический бустинг

Динамический бустинг



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

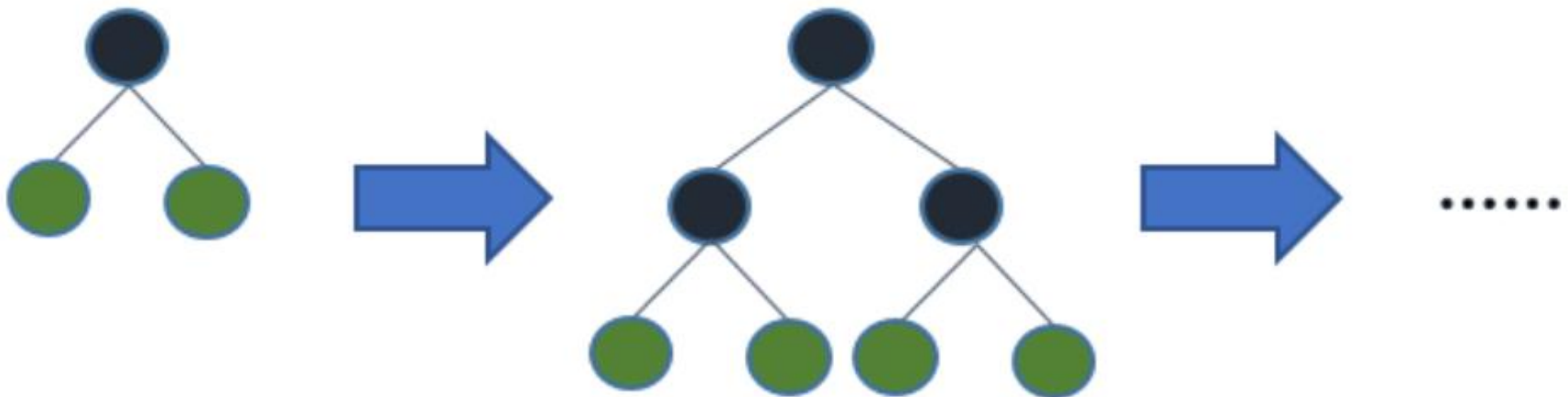
CATBOOST

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

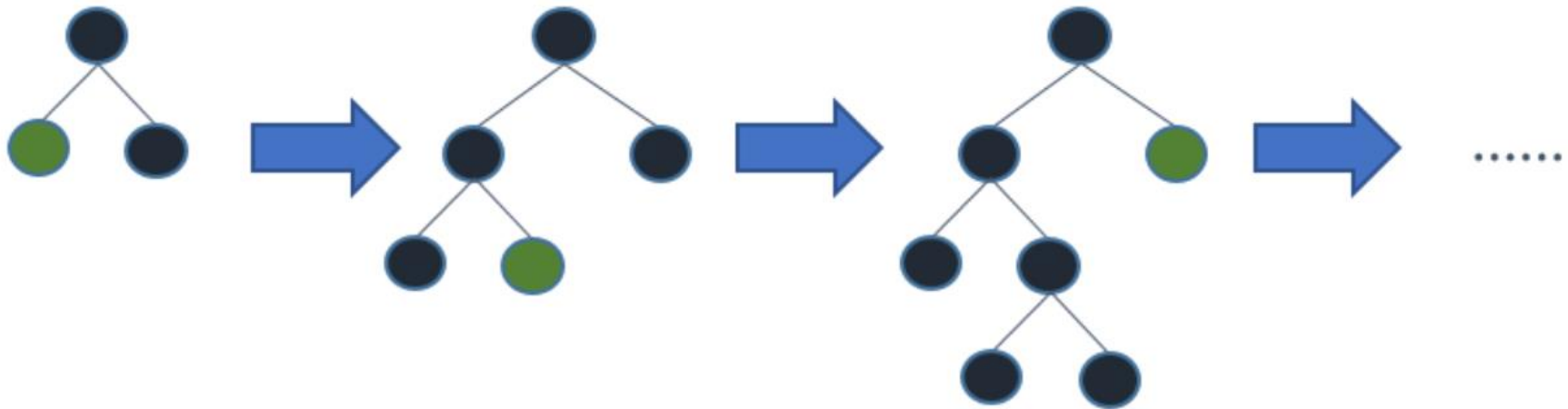
В других реализациях градиентного бустинга деревья строятся по уровням:



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.