Яндекс

CatBoost

[Cat]Boosting

Иван Лыжин,
разработчик CatBoost

23.11.2021

Градиентный бустинг на решающих деревьях (GBDT)

Что такое бустинг?

Это сумма базовых моделей: $\hat{y}_T(x) = \sum_{t=1}^{T} b_t(x)$

Почему градиентный?

Потому что используем градиент функции потерь для обучения следующей модели $b_{t+1} = F(X, \frac{\delta L(y,\hat{y}_t)}{\delta \hat{y}_t})$

Почему бы не учиться на остатках?

Остатки не всегда отражают функцию потерь.
Иногда таргета нет, а функция потерь есть.
Иногда таргетов несколько, хотя предикт один.

Почему на решающих деревьях?

Мощная и гибкая модель.
Не требует предобработки данных.

Популярные реализации

CatBoost

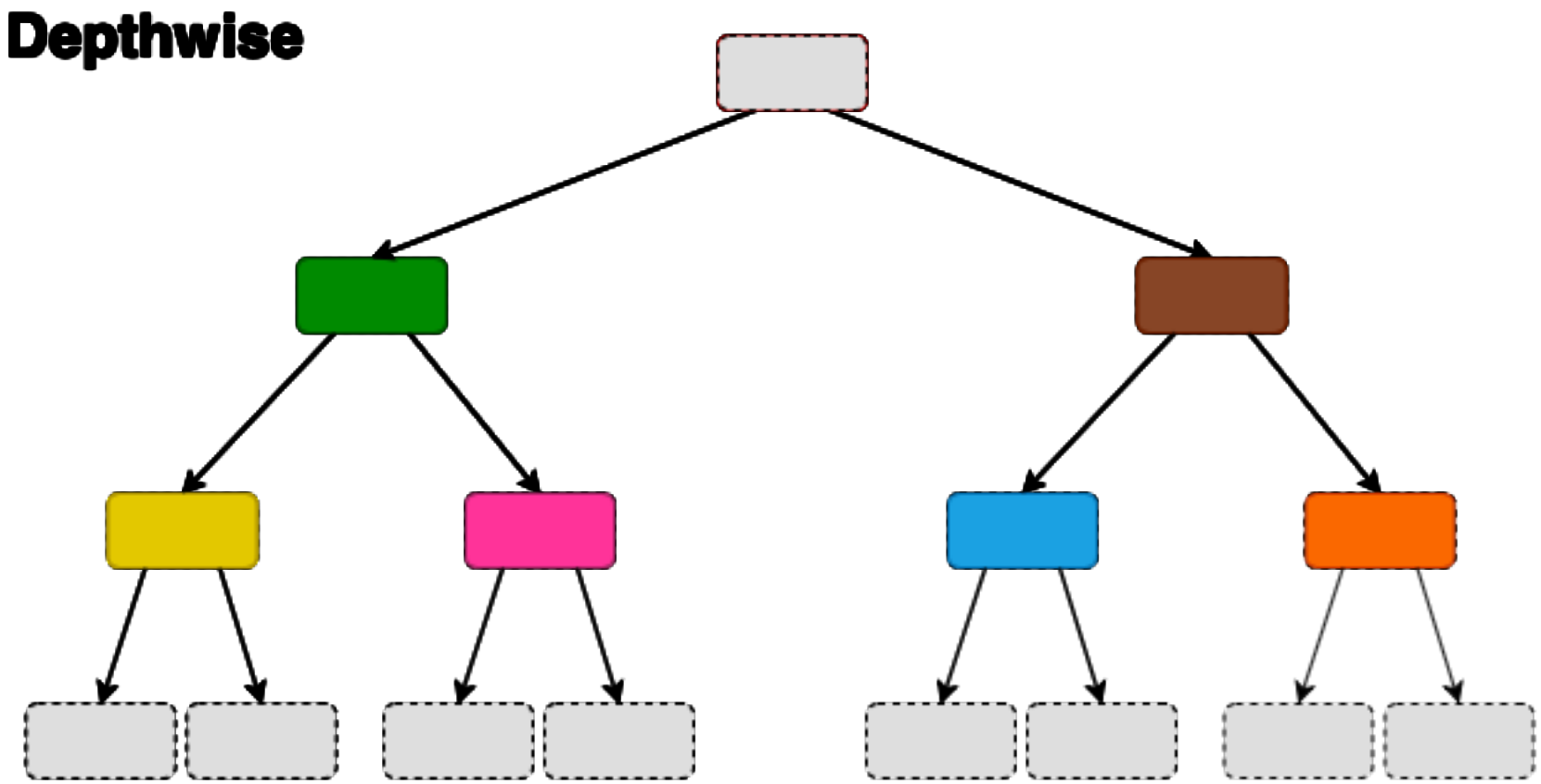
XGBoost

LightGBM

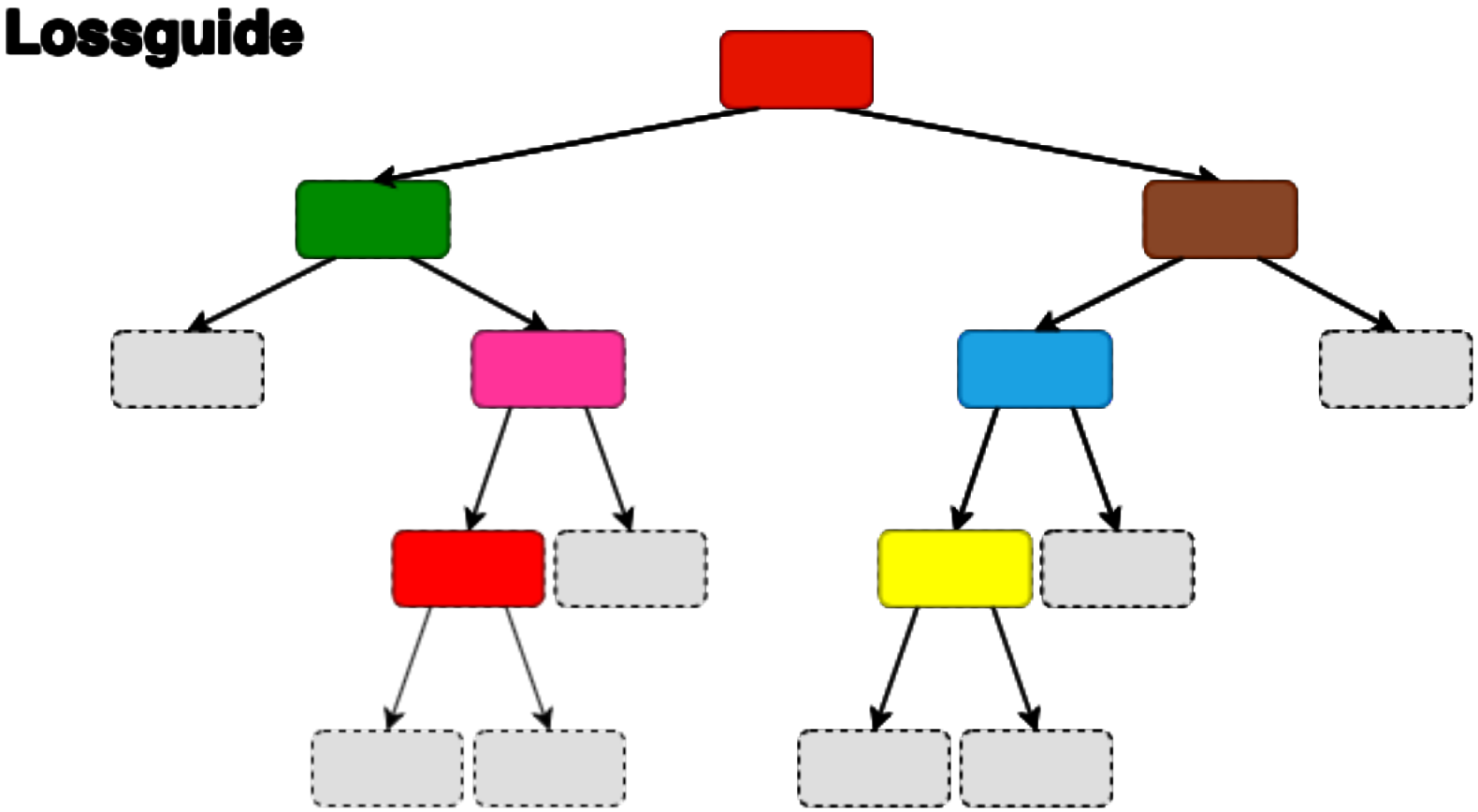
Как строится очередное дерево

- Вход:
  - X - признаки
  - $\frac{\delta L(y,\hat{y}_t)}{\delta \hat{y}_t}$ - градиент функции потерь
- Выход:
  - набор вершин со сплитами $(featureIdx, thresholdValue)$
  - значения в листьях v_j
- Как выбирать сплиты?
- В каком порядке разбивать вершины?

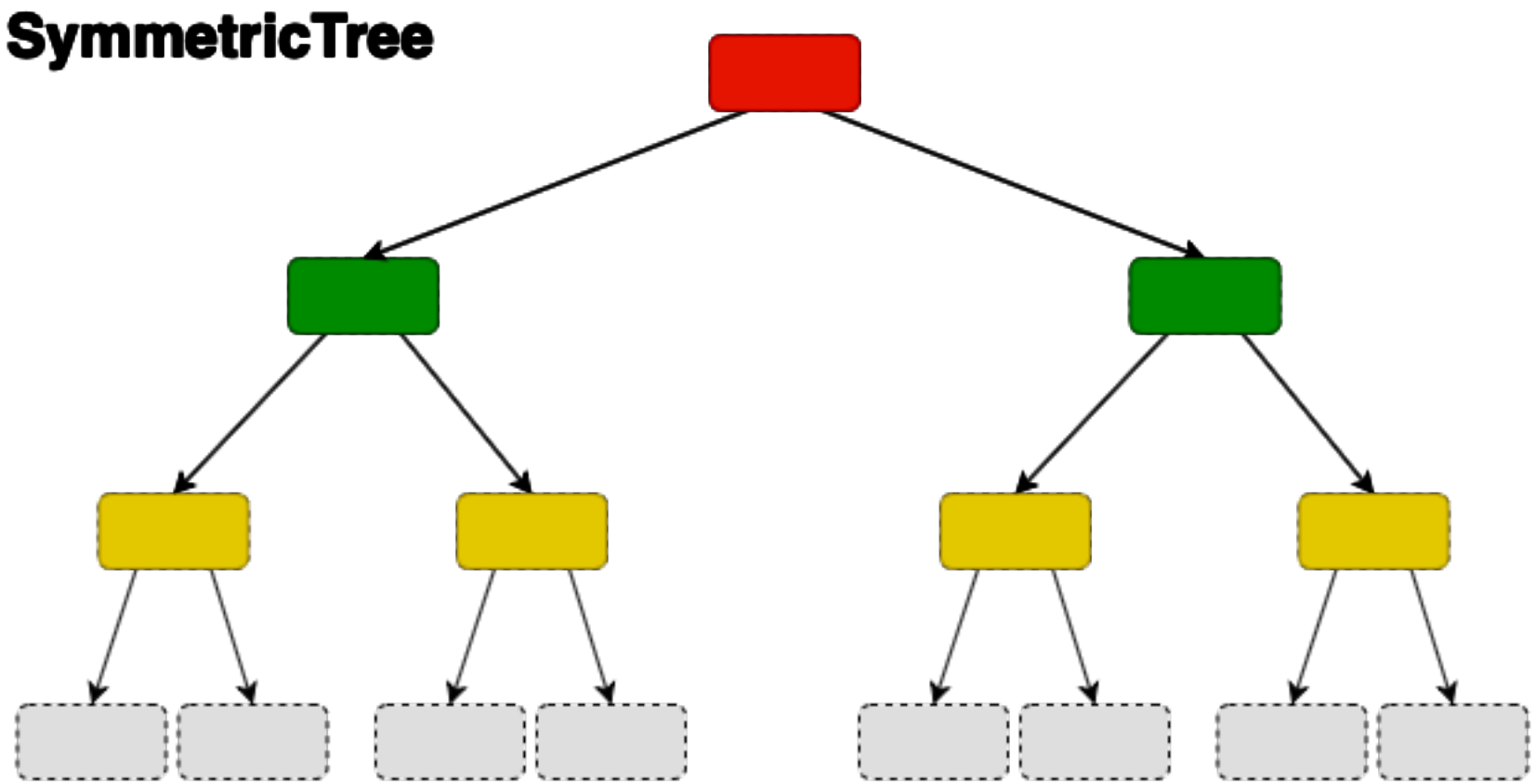
Форма дерева – XGBoost – Depthwise



Форма дерева – LightGBM – Lossguide



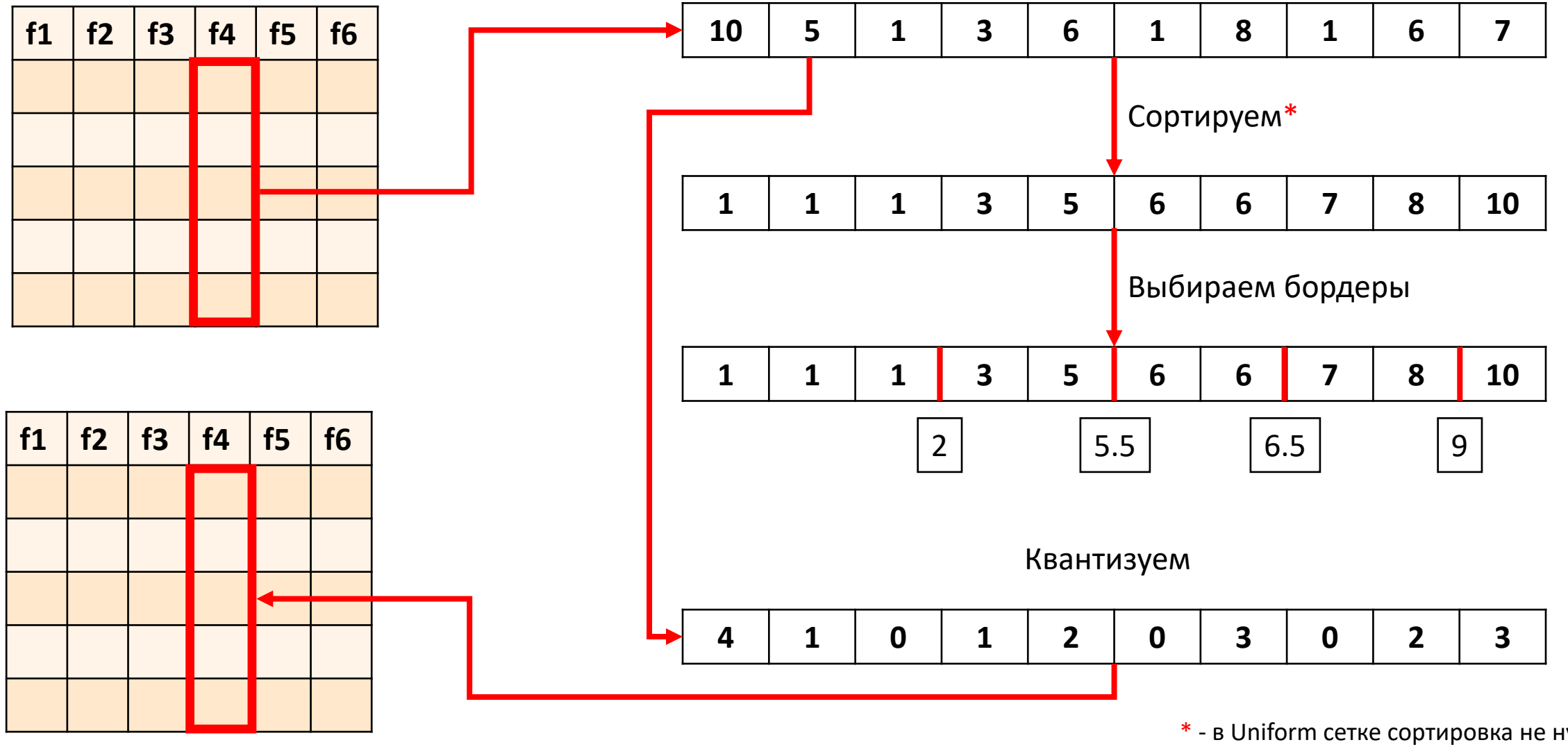
Форма дерева – CatBoost – SymmetricTree



Выбор сплита

- Для каждой фичи для каждого возможного разбиения посчитать score и по нему выбрать лучший сплит
- Score function
  - $L2 = -\sum\left(v_i - (-g_i)\right)^2 \to max$
  - $Cosine = -\frac{\sum v_i g_i}{\sqrt{\sum {v_i}^2}\sqrt{\sum g_i^2}} \to max$
- В рамках одной фичи score между последовательными разбиениями пересчитывается за O(1)
- Сложность: **O(D*N*logN)**

Ускоряем выбор сплита - квантизация

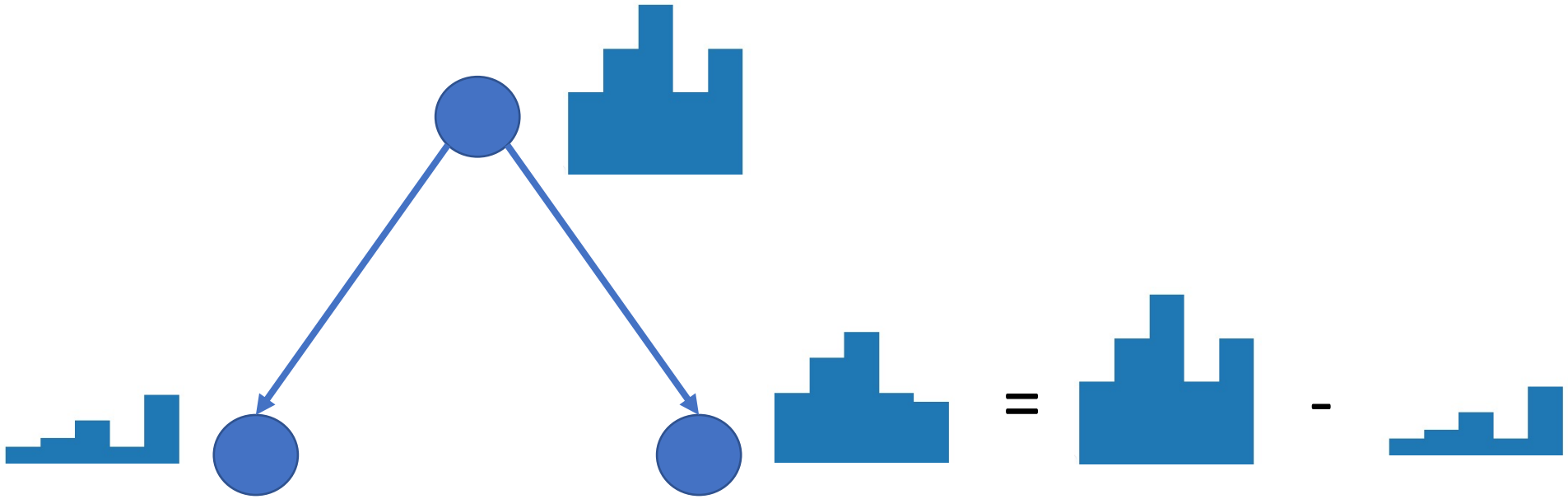


* - в Uniform сетке сортировка не нужна

Ускоряем выбор сплита - квантизация

- **N** объектов, **D** фичей, **B** бордеров
- Сложность выбора сплита раньше:
  - Для каждой фичи **O(D)**:
    - Отсортировать значения фичи **O(N*logN)**
    - Пройтись по отсортированному списку, поддерживая статистики и лучший сплит **O(N)**
  - Итого: **O(D*N*logN)**
- Сложность выбора сплита теперь:
  - Для каждой фичи **O(D)**:
    - Пройти по квантизованным значениям фичи, собрав статистики по всем значениям **O(N)**
    - Пройти по собранным статистикам и выбрать лучший сплит **O(B)**
  - Итого: **O(D*N)**, так как обычно **B << N**

Ускоряем выбор сплита – subtract trick



Даже в случае равномерных сплитов ускорение в 2 раза.

Значения в листьях - Gradient

$$\left(v - (-g)\right)^2 \to min$$

$$v_j = value\ in\ leaf_j = -\frac{\sum_{i \in leaf_j} g_i}{|leaf_j|}$$

Значения в листьях - Newton

$$L = \sum_{i=1}^{N} l(a_i + v_{leaf(i)}) \to min \quad \Rightarrow \quad L = \sum_{j=1}^{|Leaves|} \sum_{i \in leaf_j} l(a_i + v_j) \to min$$

$$\sum_{i \in leaf_j} l(a_i + v_j) \to min$$

$$\sum_{i \in leaf_j} l(a_i + v_j) \approx \sum_{i \in leaf_j} l(a_i) + v_j l'(a_i) + \frac{1}{2} v_j^2 l''(a_i) \to min$$

$$v_j = -\frac{\sum_{i \in leaf_j} g_i}{\sum_{i \in leaf_j} h_i}$$

Значения в листьях - Exact

- Какая проблема с MAE и MAPE?
  - Время обучения пропорционально масштабу целевой переменной

$MAE = \lvert y_i - a_i \rvert$	$MAE' = sign(a_i - y_i)$	$MAE'' = 0$
$MAPE = \frac{\lvert y_i - a_i \rvert}{\lvert y_i \rvert}$	$MAPE' = \frac{sign(a_i - y_i)}{\lvert y_i \rvert}$	$MAPE'' = 0$

$$v_j = median(\{\dots, y_i - a_i, \dots\})$$

Значения в листьях - Backtracking

- Выбор сплитов – самая тяжелая операция
- После долгого построения дерева делаем маленький шаг по антиградиенту
- Идея: после построения дерева во время вычисления значений в листьях будем делать несколько шагов, пересчитывая градиент

```
leaf_estimation_iterations
leaf_estimation_backtracking
```

Категориальные признаки – label encoding

- Каждой возможной категории сопоставляем число
  - sklearn.preprocessing.LabelEncoder — алфавитный порядок.
  - pandas.factorize — в порядке встречаемости значения.
  - Минусы:
    - Линейные модели плохо работают с такими признаками.
    - Деревья могут работать, но потребуется глубокое дерево.

Категориальные признаки – one-hot-encoding

- Создаем $K - 1$ новых признаков, K — кол-во категорий.
- Каждый i-ый признак — индикатор i-ой категории.
- Если у фичи много уникальных значений, то добавим много новых признаков.
- **Никогда не делайте one-hot-encoding вручную!!!**
- CatBoost может сделать его гораздо быстрее и эффективнее
- Параметр *one-hot-maxsize* отвечает за то, какие фичи пройдут через one-hot-encoding

Категориальные фичи – mean encoding

- У нас это называют ***счетчиками***
- Упрощенно – средний таргет при заданной категории

$$ctr_{f\,j=c} = \frac{\sum_{x_{ij}=c} y_i}{\sum_{x_{ij}=c} 1}$$

- На самом деле, есть разные типы счетчиков
- Решается проблема с большой глубиной дерева
- Проблемы:
  - Если количество объектов в категории мало, то оценка статистики будет очень шумной.
  - В счетчике используется значение таргета

Mean encoding - сглаживание

- Добавляем сглаживающие слагаемые, которые помогут при малом количестве объектов в категории

$$ctr_{f\,j=c} = \frac{A + \sum_{x_{ij}=c} y_i}{B + \sum_{x_{ij}=c} 1}$$

Категориальные фичи – kfold счетчики

- Разбиваем данные на k фолдов
- Для получения статистики для k-ого фолда используем таргеты всех фолдов, кроме k-ого
- Для подсчета статистики для теста используется весь train

Категориальные фичи – expanding mean

- Зафиксируем некоторый порядок объектов
- От порядка зависят значения счетчиков
- Для расчета счетчика на i-ом объекте используем только объекты, стоящие в перестановке до него

Expanding mean в CatBoost

- Генерируем 3 + 1 случайных перестановки объектов
- На каждой итерации одну из трех перестановок используем для выбора сплитов
- Четвертая перестановка нужна для расчета значений в листьях и итоговых счетчиков, сохраняемых в модель

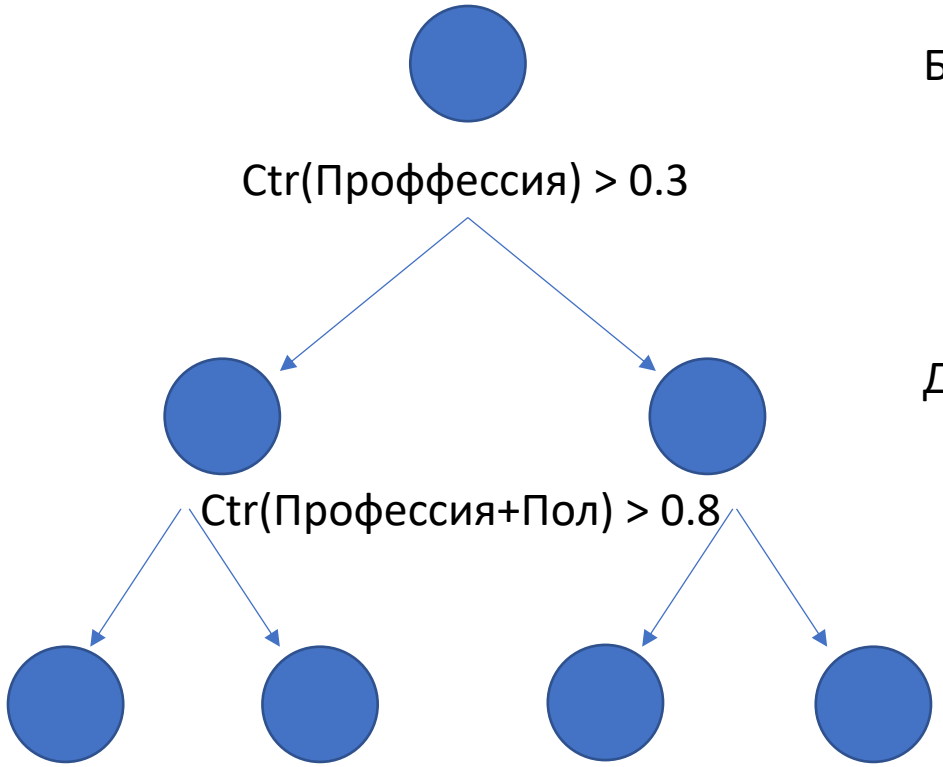
Категориальные фичи - комбинации

Пол	Профессия	Город	Y
М	Водитель	Москва	0
Ж	Продавец	Челябинск	1
М	Продавец	Москва	0
М	Программист	Самара	0
Ж	Программист	Москва	0
М	Водитель	Челябинск	1
Ж	Продавец	Самара	1

Пол + Профессия	Профессия + Город	Город + Пол	Y
М+Водитель	Водитель+Москва	Москва+М	0
Ж+Продавец	Продавец+Челябинск	Челябинск+Ж	1
М+Продавец	Продавец+Москва	Москва+М	0
М+Программист	Программист+Самара	Самара+М	0
Ж+Програмист	Программист+Москва	Москва+Ж	0
М+Водитель	Водитель+Челябинск	Челябинск+М	1
Ж+Продавец	Продавец+Самара	Самара+Ж	1

Проблема: комбинаций очень много – замедляется скорость обучения и увеличивается шанс переобучения.

Категориальные фичи - комбинации



Без комбинаций

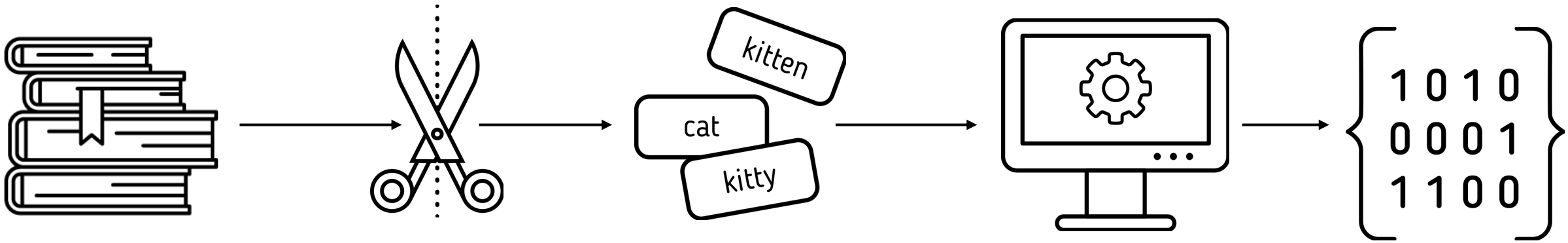
Добавляем комбинации вида «Профессия + X»

Добавляем комбинации вида «Профессия + Пол + X»

Решение: добавляем комбинации динамически только с теми фичами, которые уже были выбраны выше.

Размер комбинации ограничиваем параметром ***max_ctr_complexity***.

Текстовые фичи



Input text

Tokenizing

Creating dictionary

Applying feature

These computed estimators features are passed to training procedure

Текстовые фичи

- Bag of Words
  - Default: униграммы и биграммы
- Naïve Bayes
  - $P(Class \mid Text) = P(Class) * \prod P(word_i \mid Class)$
- BM25
  - $score(D, Q) = \sum IDF(q_i) * \frac{f(q_i, D) * (k+1)}{f(q_i, D) + k * (1 - b + b * \frac{|D|}{avgdl})}$

Монотонные ограничения

- Хотим, чтобы предсказание модели монотонно зависело от значения признака
  - Положительное монотонное ограничение

$$x_1 \leq x_1' \Rightarrow f(x_1, x_2) \leq f(x_1', x_2)$$

  - Отрицательное монотонное ограничение

$$x_1 \leq x_1' \Rightarrow f(x_1, x_2) \geq f(x_1', x_2)$$

- Априорное знание или бизнес-требование

Монотонные ограничения

- Чтобы ансамбль был монотонным, достаточно составлять его из монотонных деревьев
- Это сильно ограничивает, но искать комбинацию немонотонных деревьев намного сложнее
- Алгоритм в LightGBM и XGBoost:
  - В корень кладем тривиальное ограничение на значение (-inf, +inf)
  - При вычислении значения в листе будем заменять его на ближайшее разрешенное
  - При сплите по монотонной фиче делим отрезок в точке, равной среднему значению в новых листьях

Монотонные ограничения

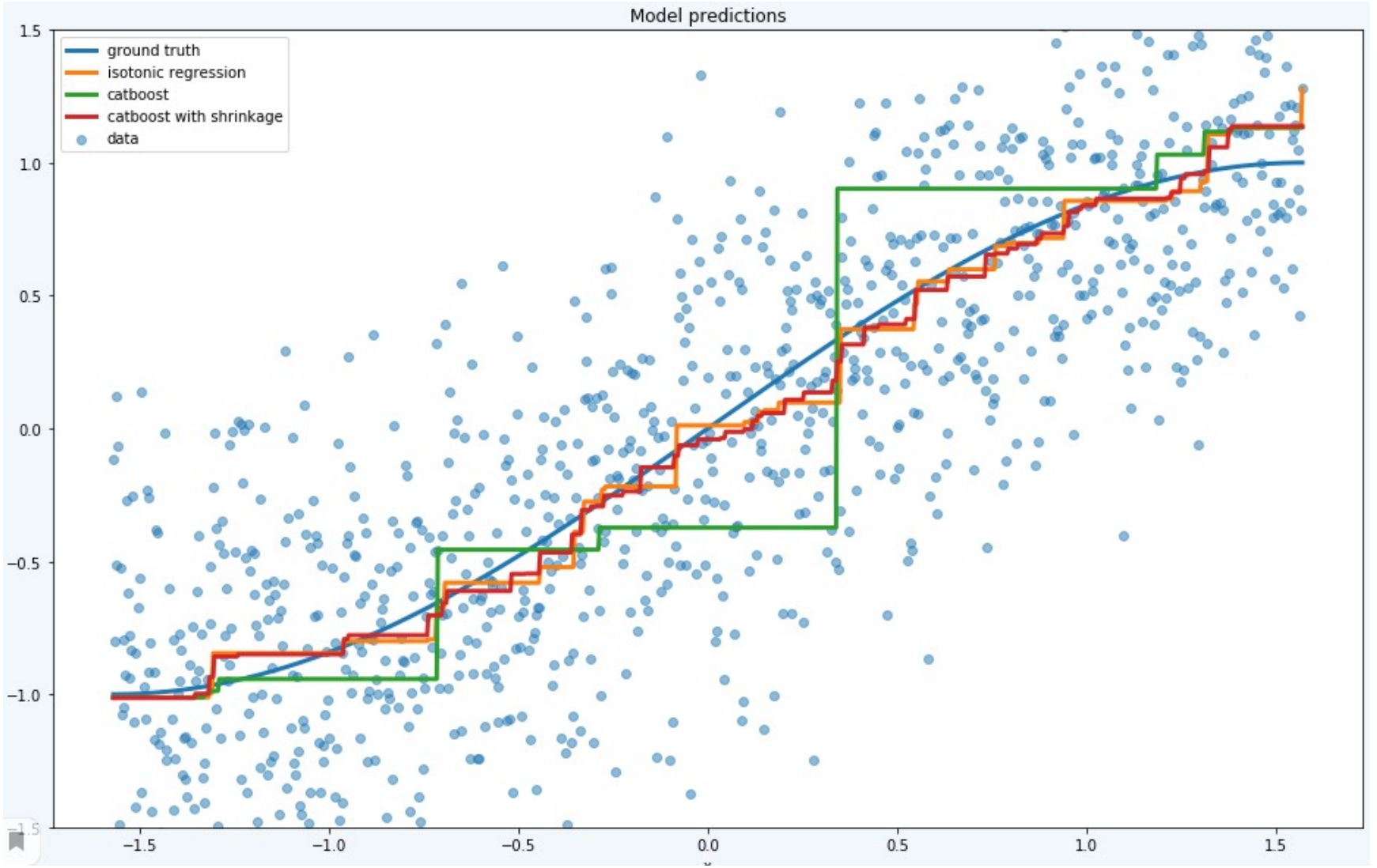
- Алгоритм CatBoost:
  - Так как сплит выбирается единый на весь уровень, можем подбирать оптимальные значения сразу для всех текущих листьев
  - Максимизацию скора можно переписать в виде:

$$\sum_{l \in Leaves} w_l (v_l - v_l')^2 \to min$$

$$v_1' \leq v_2' \leq \cdots \leq v_L'$$

  - Получаем одномерную изотоническую регрессию – для нее есть линейный алгоритм

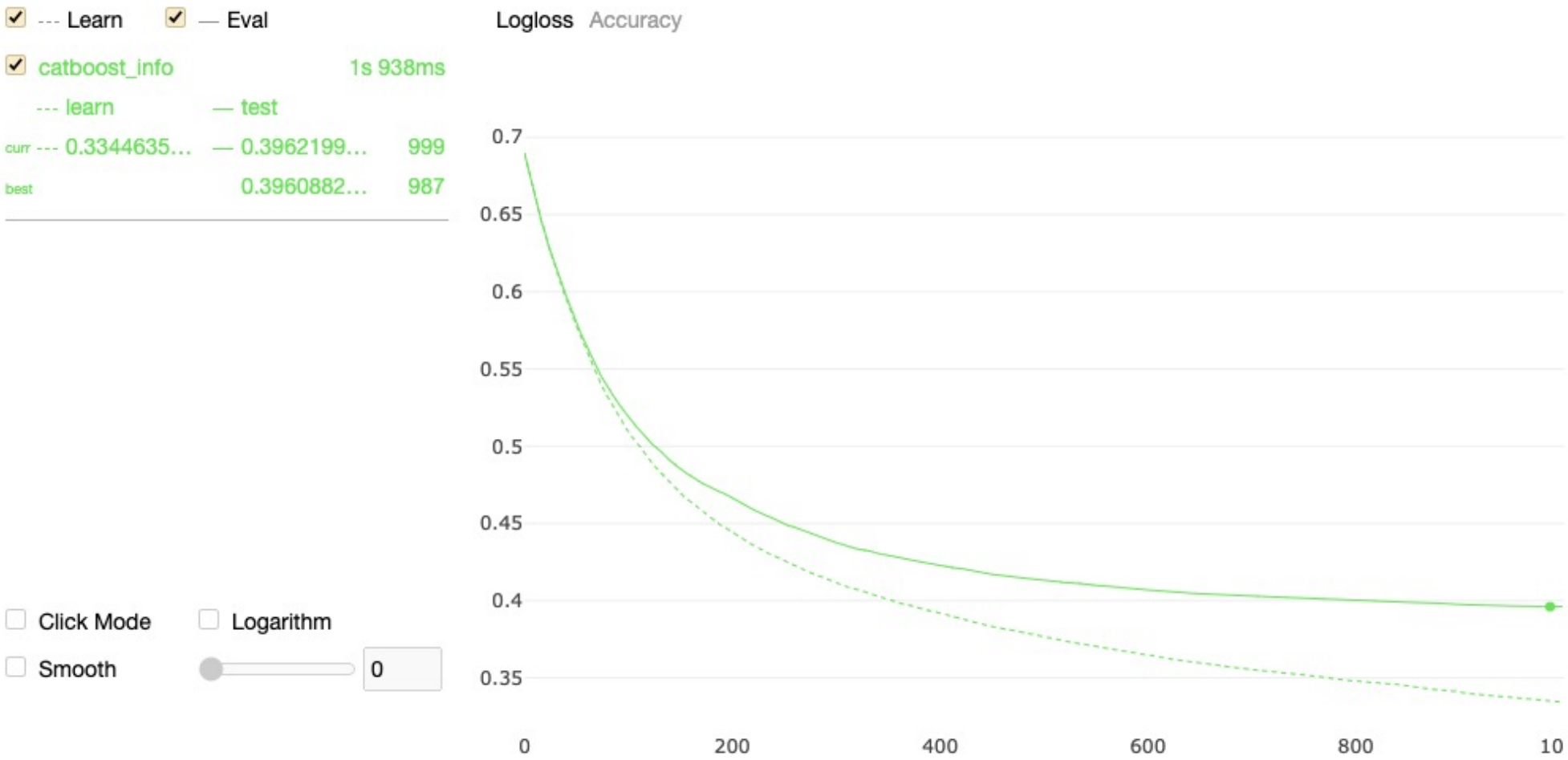
Монотонные ограничения – сжатие модели



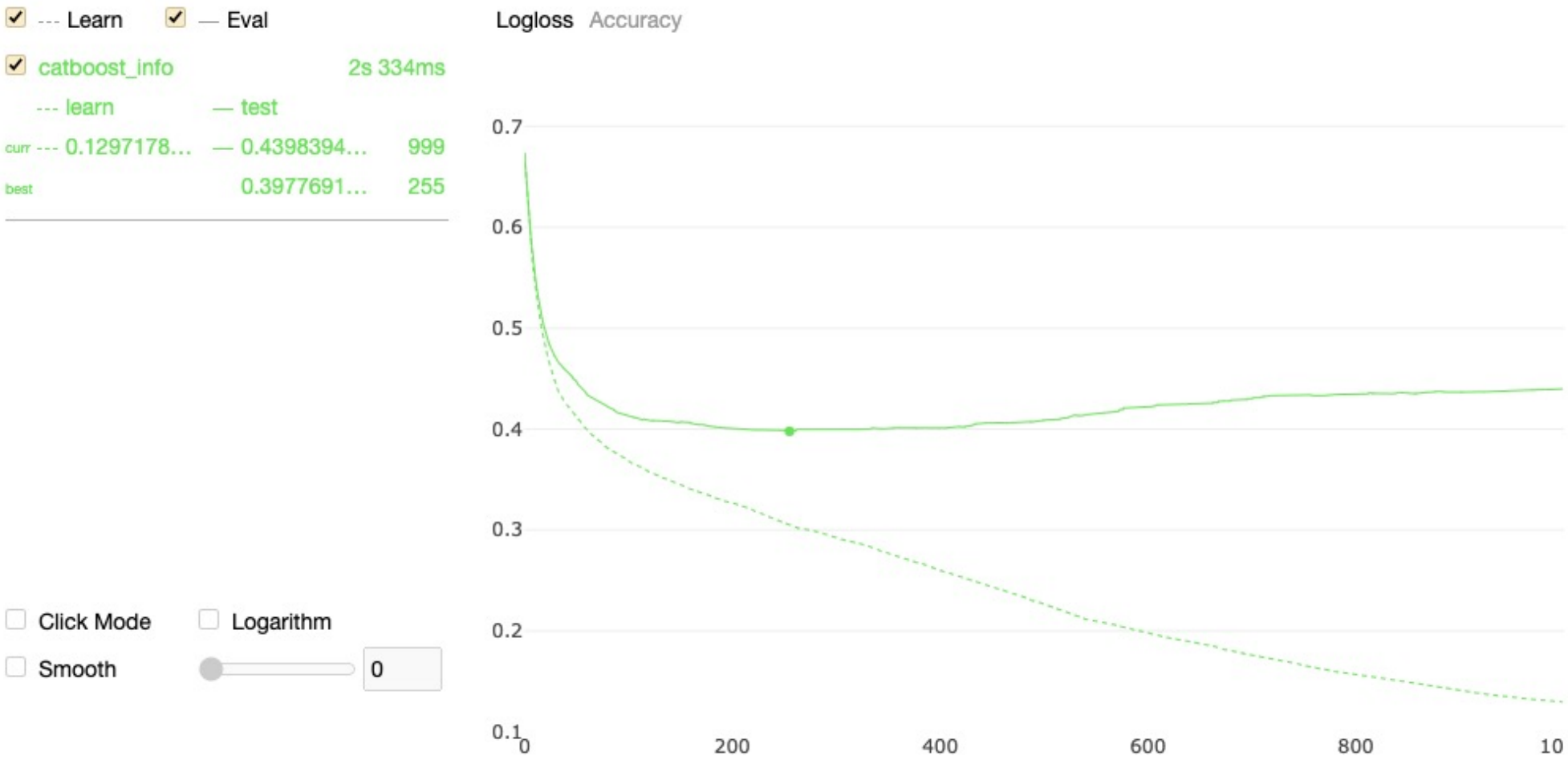
Поддерживаемые режимы

- Классификация
- Мультиклассификация
- MultiLabel классификация
- Регрессия
- Мультирегрессия
- RMSEWithUncertainty
- Ранжирование
- Попарное ранжирование

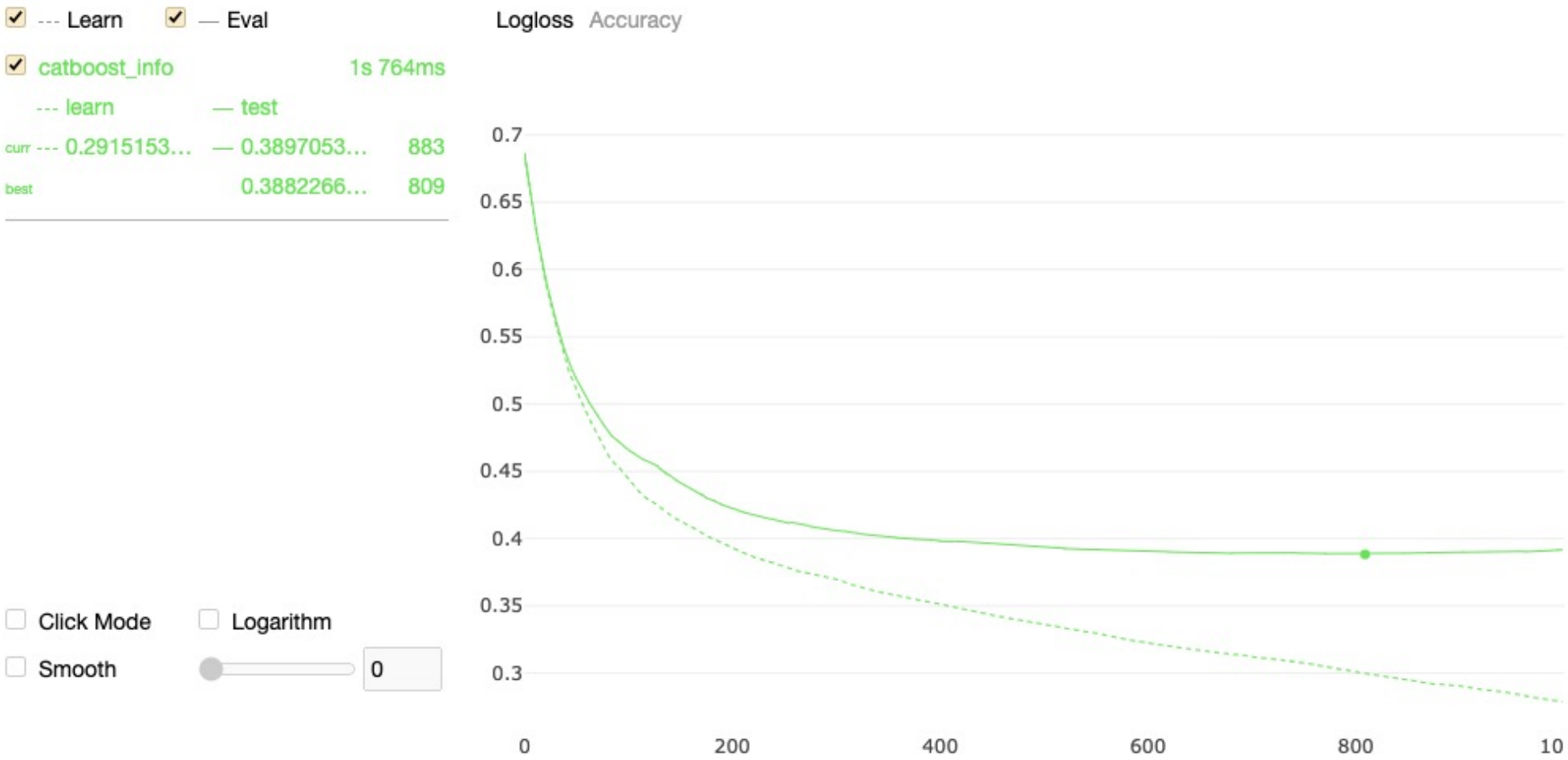
Learning-rate vs iterations – underfitting



Learning-rate vs iterations - overfitting



Learning-rate vs iterations



eval_set и eval_metric

- В обучение можно передать датасет для валидации и целевую метрику
- Это открывает возможность использования опций:
  - use_best_model – обрезает модель по лучшей итерации на валидации
  - early_stopping_rounds – позволяет отследить переобучения и остановиться

Snapshots

- Задает опциями *snapshot_file* и *snapshot_interval*
- Позволяет сохранять прогресс обучения
- Можно продолжить обучение с другим *learning_rate*

Подбор гиперпараметров

- iterations / learning-rate
- depth – глубина деревьев
- l2-leaf-reg – параметр регуляризации
- max-ctr-complexity – сложность комбинаций кат.фичей
- random-strength – элемент случайности, иногда полезно выключить
- border-count – размер сетки квантизации
- grow-policy – форма дерева

Анализ модели и признаков в CatBoost

- Feature importance
- Shap Values
- Partial Dependence Plot
- Feature Statistics
- Object importance
- Monoforest
- …

Feature Importance

Prediction values change

- Как сильно признак влияет на предсказания модели
- Очень быстро считается
- Не зависит от датасета
- Дефолт для классификации и регрессии

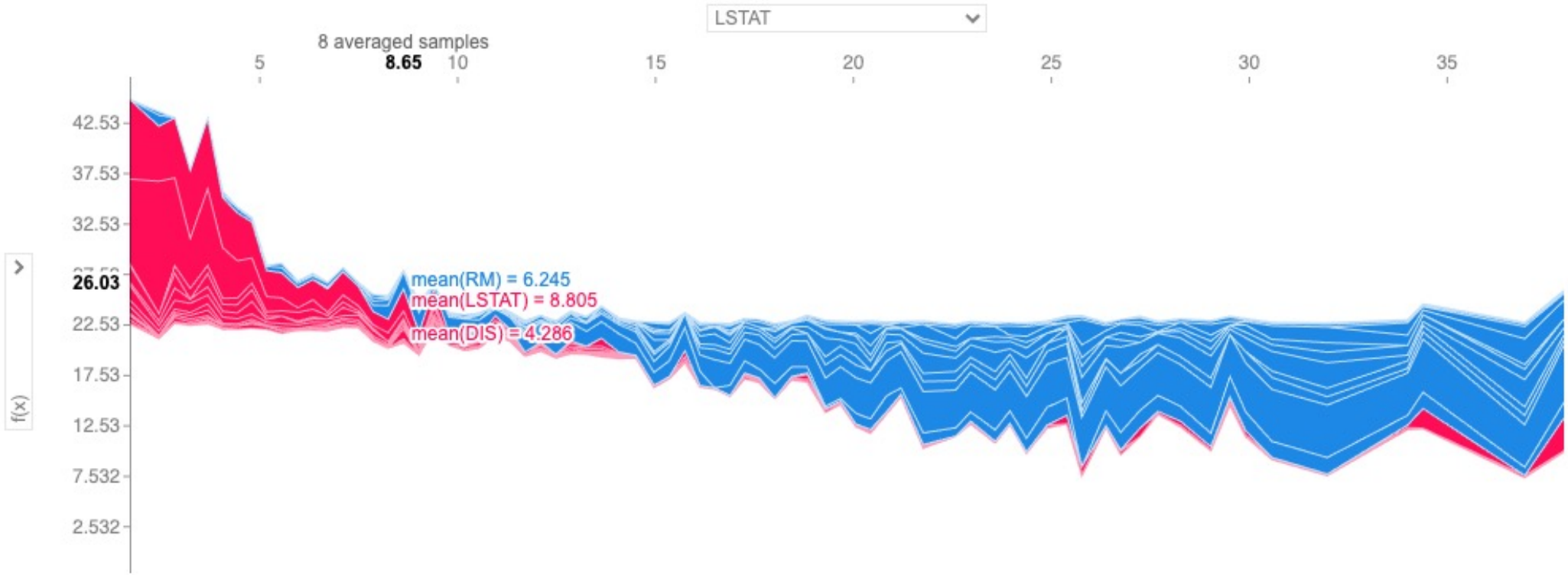
Loss function change

- Как сильно признак влияет на значение функции потерь
- Считается относительно медленно
- Зависит от датасета
- Дефолт для ранжирования

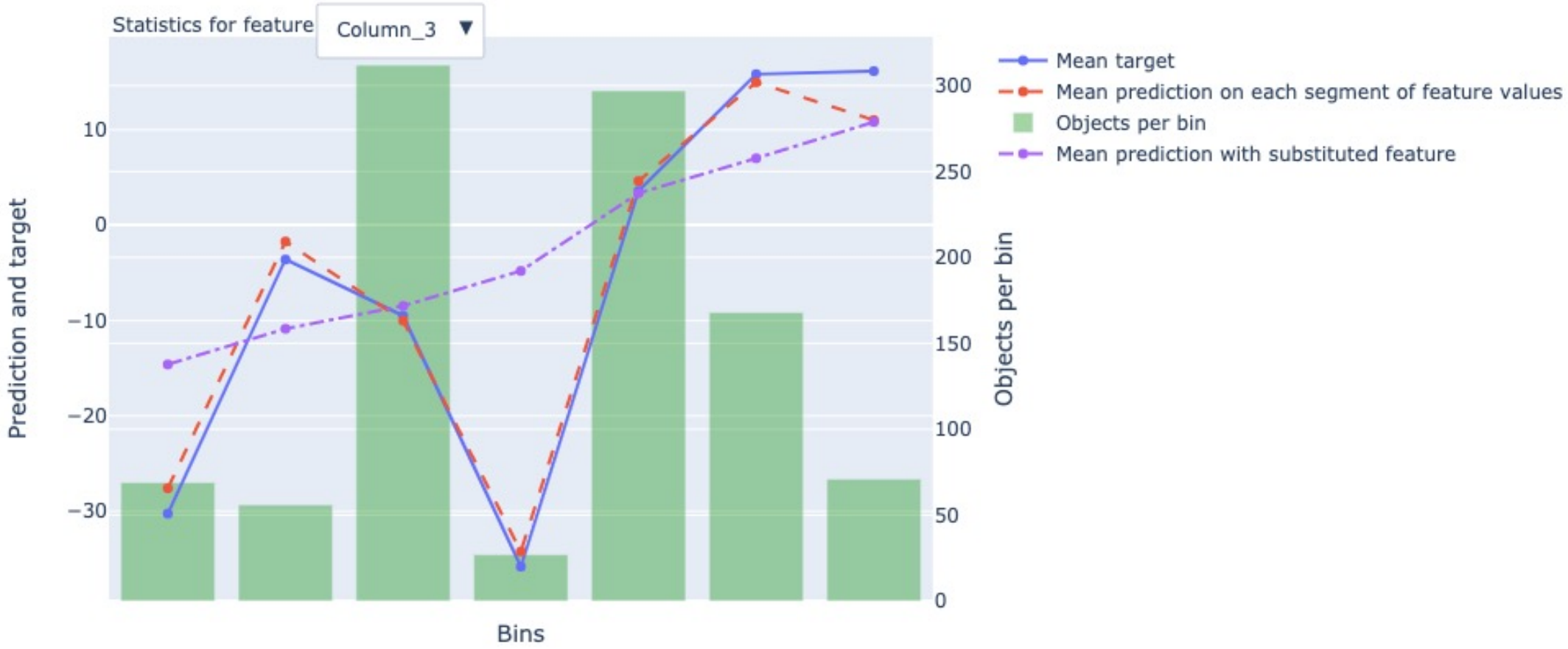
Shap values


higher ⇄ lower
base value
f(x)
16.53
18.53
20.53
22.53
24.21
26.53
28.53
INDUS = 2.31
PTRATIO = 15.3
LSTAT = 4.98
RAD = 1
RM = 6.575
CRIM = 0.00632
TAX = 296

Shap values


LSTAT
8 averaged samples
8.65
5
10
15
20
25
30
35
42.53
37.53
32.53
26.03
22.53
17.53
12.53
7.532
2.532
f(x)
mean(RM) = 6.245
mean(LSTAT) = 8.805
mean(DIS) = 4.286

Feature statistics



Custom loss and custom metric

- **pip install numba**
- JIT-компиляция python кода
- Ускоряет процесс обучения

CatBoost

Яндекс

catboost_en

catboost_ru

https://catboost.ai/

https://github.com/catboost/catboost

Иван Лыжин,

разработчик CatBoost

ilyzhin