

Бустинг

Градиентный бустинг над решающими деревьями

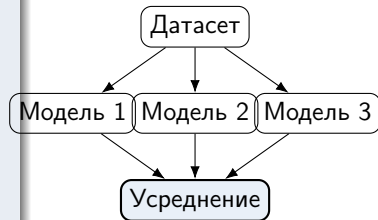
Лазара В. И. Козлова Е. Р.

Лекция по машинному обучению

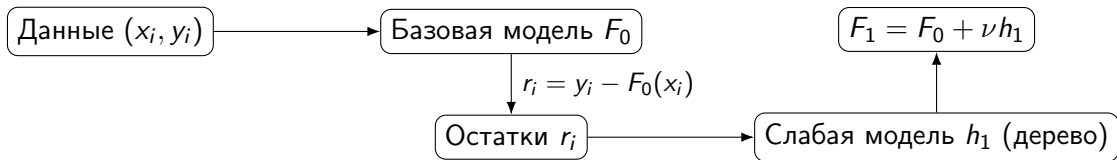
17 октября 2025 г.

Коротко

- **Бэггинг** (Bootstrap Aggregating): параллельные независимые модели на бутстрап-подвыборках, усреднение/голосование \Rightarrow снижение дисперсии.
- **Стэкинг**: мета-модель учится комбинировать ответы базовых моделей.
- **Бустинг**: *последовательно* добавляем слабые модели, каждая исправляет ошибки предыдущих \Rightarrow снижение смещения.



- Модель $F_0(x)$ грубо приближает зависимость.
- Вычисляем **ошибки/остатки** и обучаем следующую слабую модель $h_1(x)$ предсказывать эти ошибки.
- Новая модель: $F_1(x) = F_0(x) + \nu h_1(x)$, где $\nu \in (0, 1]$ — *скорость обучения*.
- Повторяем M раз: $F_M(x) = F_{M-1}(x) + \nu h_M(x)$.



Градиентный бустинг: оптимизация в пространстве функций

Задача

Дана выборка $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^n$, хотим минимизировать

$$\mathcal{L}(F) = \sum_{i=1}^n \ell(y_i, F(x_i)),$$

где F — искомая функция, а ℓ — выбранная функция потерь.

Идея

Выполняем **градиентный спуск по F** : на шаге m подбираем слабую модель h_m , хорошо аппроксимирующую *антиградиент* по значениям F на обучающих объектах:

$$g_{im} = \left. \frac{\partial \ell(y_i, z)}{\partial z} \right|_{z=F_{m-1}(x_i)}, \quad \text{учим } h_m \approx -g_{im} \text{ по } x_i.$$

Затем обновляем $F_m(x) = F_{m-1}(x) + \nu \gamma_m h_m(x)$

Бустинг над деревьями: шаг обучения

1. Дано F_{m-1} . Считаем **псевдо-остатки** (антиградиенты):

$$r_{im} = - \left. \frac{\partial \ell(y_i, z)}{\partial z} \right|_{z=F_{m-1}(x_i)}.$$

2. Обучаем регрессионное дерево $h_m(x)$ по парам (x_i, r_{im}) .
3. Находим оптимальные константы по листам: для каждого листа R_{jm}

$$\gamma_{jm} = \arg \min_{\gamma} \sum_{x_i \in R_{jm}} \ell(y_i, F_{m-1}(x_i) + \gamma).$$

4. Обновляем модель:

$$F_m(x) = F_{m-1}(x) + \nu \sum_j \gamma_{jm} 1\{x \in R_{jm}\}.$$

Градиентный бустинг над деревьями: псевдокод

Вход

$\mathcal{D} = \{(x_i, y_i)\}_{i=1}^n$, функция потерь $\ell(y, z)$, число итераций M , глубина дерева d , скорость обучения ν .

Алгоритм

- Инициализация: $F_0(x) = \arg \min_c \sum_{i=1}^n \ell(y_i, c)$ (например, среднее для MSE, логит-квантиль для логлосса).
- Для $m = 1, \dots, M$:
 1. Вычислить $r_{im} = -\partial \ell(y_i, z) / \partial z \big|_{z=F_{m-1}(x_i)}$.
 2. Обучить регрессионное дерево h_m глубины $\leq d$ на $\{(x_i, r_{im})\}$.
 3. Для каждого листа R_{jm} найти $\gamma_{jm} = \arg \min_{\gamma} \sum_{x_i \in R_{jm}} \ell(y_i, F_{m-1}(x_i) + \gamma)$.
 4. Обновить $F_m(x) = F_{m-1}(x) + \nu \sum_j \gamma_{jm} \mathbf{1}\{x \in R_{jm}\}$.

Регрессия

- Квадратичная: $\ell(y, z) = \frac{1}{2}(y - z)^2 \Rightarrow r_{im} = y_i - F_{m-1}(x_i)$ (классические «остатки»).
- Абсолютная: $\ell(y, z) = |y - z| \Rightarrow r_{im} = \text{sign}(y_i - F_{m-1}(x_i))$ (в слабом смысле).

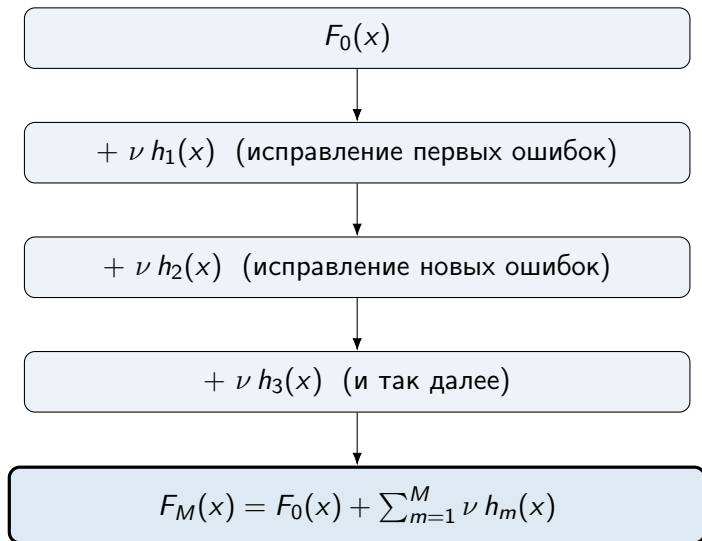
Классификация (бинарная)

- Логистическая потеря: $y \in \{0, 1\}$, логиты $F(x)$, $p(x) = \sigma(F(x))$.

$$\ell(y, z) = -(y \log \sigma(z) + (1-y) \log(1-\sigma(z)))$$

- Псевдо-остатки:
 $r_{im} = y_i - \sigma(F_{m-1}(x_i))$.
- Итог: решение выдаём как $\mathbb{I}\{F_M(x) \geq 0\}$ или по $p(x)$.

Иллюстрация: «слоёный пирог» из деревьев



Каждое дерево — слабое (мелкое, с малым числом листьев), но сумма даёт сильную модель 9 / 15

- **Скорость обучения** (*shrinkage*) $\nu \in (0, 1]$: меньше \Rightarrow устойчивее, но нужно больше деревьев.
- **Глубина дерева / число листьев**: неглубокие деревья (3–8 уровней) \Rightarrow слабые базовые модели.
- **Субсемплинг объектов** (*subsample*): обучаем h_m на случайной доле данных (напр., 0.5–0.9).
- **Субсемплинг признаков**: случайный поднабор признаков на сплите/уровне/дереве.
- **Минимальный размер листа, L2-штраф на веса листов, макс. число узлов.**
- **Ранняя остановка по валидации**: мониторим метрику и прекращаем рост M .

Идея

На каждом шаге используем случайную подвыборку объектов (и, опционально, признаков).

- Снижает коррелированность базовых деревьев, улучшая обобщающую способность.
- Даёт ускорение и повышает устойчивость к шуму.
- Хорошо комбинируется с малым ν и ранней остановкой.

- **XGBoost**: точный/приближённый поиск сплитов, регуляризация (L1/L2), колонки по блокам, эффективная параллелизация.
- **LightGBM**: лист-ориентированный рост (*leaf-wise*) с ограничением глубины, гистограммные сплиты, *Gradient-based One-Side Sampling*.
- **CatBoost**: обработка категориальных признаков порядковыми статистиками, упор на устойчивость к *target leakage*.

Замечание

Хотя детали реализации различаются, базовая идея — тот же градиентный бустинг над деревьями.

- Разрыв между train и valid метриками растёт со временем \Rightarrow остановиться раньше.
- Локальные всплески ошибки при слишком глубоком дереве.
- Слишком малый *subsample* может добавить шум (слишком большой — повысит корреляцию базовых моделей).

Бустинг

- Последовательный, корректирует смещение.
- Высокая предсказательная сила «из коробки».
- Чувствителен к шуму/выбросам (важна регуляризация).

Бэггинг/Стэкинг

- Бэггинг — параллельный, снижает дисперсию.
- Стэкинг — мета-комбинация, требует аккуратной валидации.
- Часто менее чувствительны к отдельным выбросам.

- Бустинг — последовательное уменьшение смещения путём добавления слабых моделей.
- Градиентный бустинг — градиентный спуск в пространстве функций по выбранной потере.
- Деревья — удобные слабые модели: быстрые, интерпретируемые на уровне сплитов.
- Ключ — **регуляризация**: шринкаж, ранняя остановка, контроль сложности деревьев и стохастичность.