Лекция 7

Ансамбли и бустинг

Машинное обучение · 2026

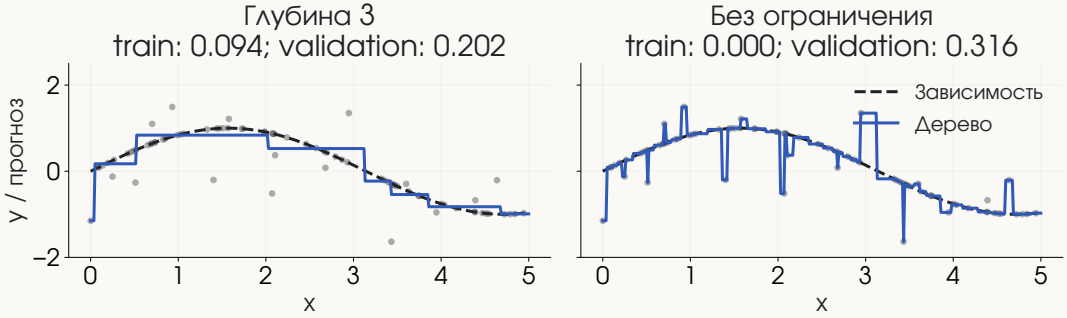
Лектор

Елизавета Власова

Практики

Кирилл Захаров
Михаил Подсытник
Игорь Толстокулаков

Переобучение дерева



Глубокое дерево подгоняет отдельные шумные наблюдения.

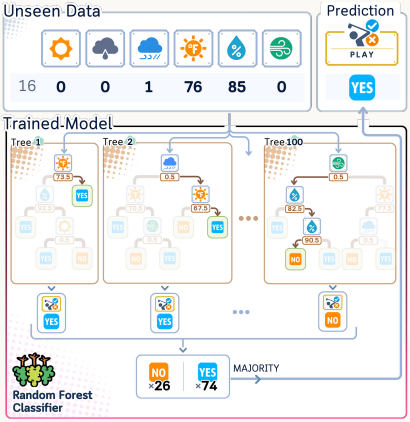
Можно ли сохранить гибкость дерева и сделать прогноз устойчивее?

Бэггинг и случайный лес

Разные деревья и общий прогноз

1

Композиция базовых алгоритмов



Берём базовые алгоритмы $a_1, \ldots, a_M$ и объединяем их прогнозы:

$$F(x) = C\big(a_1(x), \ldots, a_M(x)\big).$$

C — правило агрегации.

На схеме: **74 Yes**, **26 No**. Голосование даёт **Yes**.

Отдельные деревья могут отвечать по-разному.

Bootstrap-выборка

Исходная train-выборка содержит строки $1, 2, 3, 4, 5, 6$.

Для каждого дерева выбираем **6 строк с возвращением**:

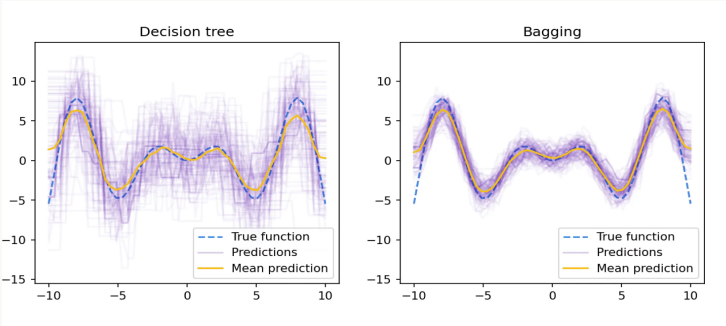
Дерево	Выбранные строки	Не попали
1	$1, 1, 2, 4, 4, 6$	$3, 5$
2	$2, 2, 3, 5, 5, 6$	$1, 4$
3	$1, 3, 3, 4, 6, 6$	$2, 5$

Одни строки повторяются; другие не попадают в выборку.

Размер сохраняется, но **уникальных строк меньше**.

Бэггинг деревьев

Bagging = bootstrap aggregating.



Слева — отдельное дерево, справа — бэггинг; каждый раз обучаемся на новой выборке.

В регрессии усредняем числа; в классификации — голоса или вероятности.

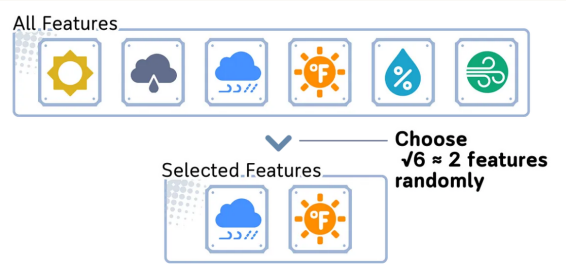
Случайный лес

Random Forest добавляет к бэггингу случайный выбор признаков.

1. Для каждого дерева создаём bootstrap-выборку строк.
2. **В каждом узле** выбираем случайное подмножество признаков.
3. Ищем лучший допустимый сплит среди выбранных признаков.
4. Агрегируем прогнозы деревьев.

Дополнительная случайность уменьшает сходство деревьев и корреляцию их ошибок.

Случайные признаки в узле



На схеме из 6 признаков случайно выбраны 2 кандидата для сплита.

В следующем узле выбирается **новое подмножество**.

Если разрешить все признаки, получим обычный бэггинг деревьев. Это допустимый вариант.

Свойства случайного леса

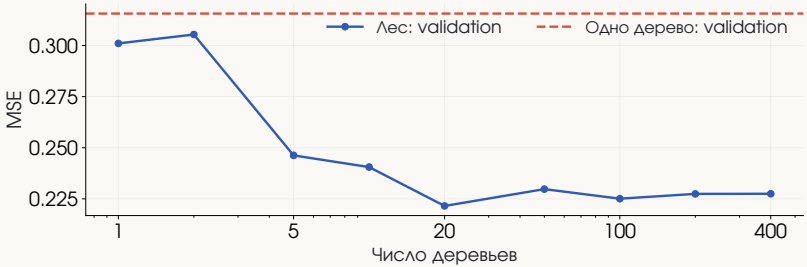
Свойство	Что получаем
Нелинейности	Деревья сохраняют нелинейности и взаимодействия признаков.
Масштабы признаков	Стандартизация обычно не нужна.
Устойчивость	Чувствительность к train-выборке меньше; переобучение всё ещё возможно.
Экстраполяция	Обычный регрессионный лес не продолжает тренд за пределы обучающих значений.
Интерпретация	Лес труднее прочитать как правила. Для важности признаков используем MDI или PI.

Параметры случайного леса

Параметр	Что меняет
`n_estimators`	Число деревьев: стабильность среднего, время и память.
`max_features`	Число кандидатов в каждом узле: разнообразие деревьев и качество отдельного сплита.
`max_depth`	Максимальная глубина каждого дерева.
`min_samples_leaf`	Минимальный размер листа: сглаживание прогнозов.
`max_samples`	Размер bootstrap-выборки для каждого дерева.

Меньше признаков — сильнее различаются деревья, но отдельное дерево может стать хуже. Выбираем по validation или CV.

Число деревьев в лесу



Тот же пример регрессии: 80 объектов на train, 2000 на validation.

Здесь один признак: график показывает эффект bootstrap и усреднения. С ростом числа деревьев выигрыш насыщается; время и память продолжают расти.

OOB-оценка качества

Out-of-bag: для строки используем деревья, которые не видели её при обучении.

Дерево	Bootstrap-строки	OOB-строки
1	$1, 1, 2, 4, 4, 6$	$3, 5$
2	$2, 2, 3, 5, 5, 6$	$1, 4$
3	$1, 3, 3, 4, 6, 6$	$2, 5$

Для строки 5 берём деревья 1 и 3: $\widehat{y}_5^{\mathrm{OOB}} = (8 + 12)/2 = 10$.

Считаем метрику по OOB-прогнозам. При больших n около 37% строк не попадают в bootstrap одного дерева.

Для временных и групповых данных нужна проверка с учётом структуры. Финальный test оставляем отдельно.

Бустинг и другие ансамбли

Последовательные поправки к прогнозу

2

Лес и бустинг

Случайный лес

Деревья учатся отдельно предсказывать исходный y.

Часто деревья глубокие: каждое гибкое, но нестабильное.

Итог — среднее прогнозов.

Бустинг

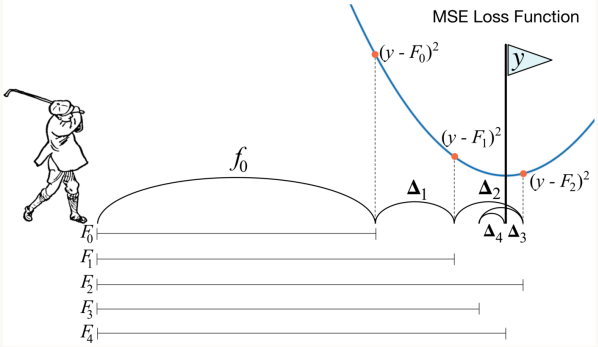
Каждое следующее дерево уточняет текущий общий прогноз.

Часто деревья неглубокие: каждое даёт простую поправку.

Итог — сумма поправок.

Что именно должно предсказывать следующее дерево?

Последовательные поправки



$$F_m(x) = F_{m-1}(x) + \eta h_m(x).$$

Суммарная модель:

$$F_M(x) = F_0(x) + \eta \sum_{m=1}^{M} h_m(x).$$

Старые деревья остаются в композиции; новое дерево добавляет поправку.

Начальный прогноз и остатки

Четыре объекта; минимизируем квадратичную ошибку.

$$F_0(x) = \bar{y} = 6, \qquad r_i^{(1)} = y_i - F_0(x_i).$$

x	1	2	3	4
y	2	4	8	10
$F_0(x)$	6	6	6	6
Остатки $r^{(1)}$	-4	-2	2	4

$$\mathrm{MSE}(F_0) = 10.$$

Новое дерево учим предсказывать остатки **по исходным признакам** x.

Первое дерево-поправка

Пень делит по $x \le 2.5$. В каждом листе — среднее остатков:

$$h_1(x) = \begin{cases} (-4-2)/2 = -3, & x \le 2.5, \\ (2+4)/2 = 3, & x > 2.5. \end{cases}$$

Берём половину поправки: learning rate $\eta = 0.5$.

$$F_1(x) = F_0(x) + 0.5 h_1(x).$$

x	1	2	3	4
$F_1(x)$	4.5	4.5	7.5	7.5

$$\text{MSE}: \quad 10 \longrightarrow 3.25.$$

Следующий шаг бустинга

Текущий прогноз: $F_1 = (4.5, 4.5, 7.5, 7.5)$; $\eta = 0.5$.

Новое дерево снова делит по $x \leq 2.5$.

x	1	2	3	4
y	2	4	8	10
$r^{(2)} = y - F_1$	-2.5	-0.5	0.5	2.5

Чему равны значения в двух листьях h_2?

Какой прогноз $F_2(4)$ получим после обновления?

Второе дерево и новая композиция

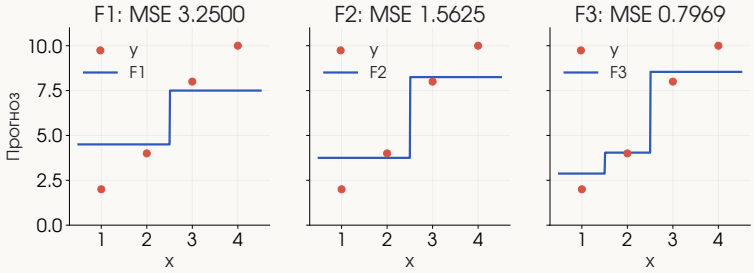
$$h_2(x) = \begin{cases} -1.5, & x \le 2.5, \\ 1.5, & x > 2.5. \end{cases} \qquad F_2 = F_1 + 0.5 h_2.$$

x	1	2	3	4
y	2	4	8	10
$F_2(x)$	3.75	3.75	8.25	8.25
$y - F_2(x)$	-1.75	0.25	-0.25	1.75

$$\mathrm{MSE}(F_2) = 1.5625.$$

Теперь третье дерево будем обучать уже на **новых остатках**.

Сумма простых деревьев



На третьем шаге лучший порог меняется на $x \leq 1.5$.

Сумма деревьев постепенно уточняет форму прогноза, хотя каждое дерево имеет только два листа.

От остатков к градиенту

Для квадратичной потери:

$$L(y, F) = \tfrac{1}{2}(y - F)^2, \qquad -\frac{\partial L(y, F)}{\partial F} = y - F.$$

Остатки — **антиградиент потери по прогнозу**.

Для другой дифференцируемой потери используем псевдоостатки:

$$r_i^{(m)} = -\left.\frac{\partial L(y_i, F)}{\partial F}\right|_{F=F_{m-1}(x_i)}.$$

Новое дерево приближает направление, в котором нужно изменить текущие прогнозы.

Бустинг для классификации

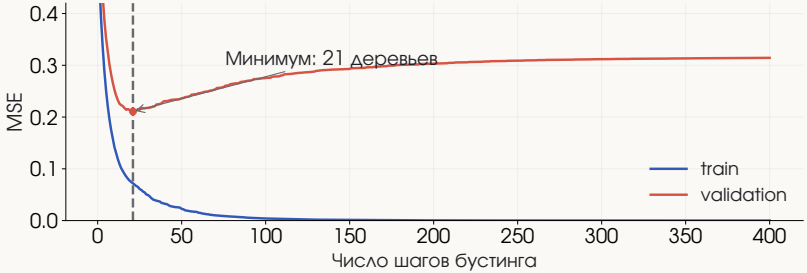
Деревья складываются в числовой score $F(x)$. Для бинарной log loss:

$$p(x) = \sigma(F(x)) = \frac{1}{1 + e^{-F(x)}}, \qquad r_i = y_i - p_i.$$

y	p	$r = y - p$	Поправка к score
1	0.2	$+0.8$	Сильно увеличить
1	0.9	$+0.1$	Немного увеличить
0	0.8	-0.8	Уменьшить

Дерево приближает антиградиент; значения поправок в листьях выбираются с учётом функции потерь.

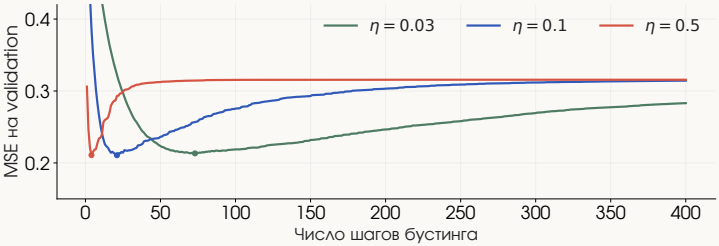
Переобучение бустинга



Тот же пример регрессии: глубина деревьев 3, $\eta = 0.1$.

Train-ошибка продолжает уменьшаться. После минимума validation-ошибка растёт: композиция всё точнее подгоняет шум.

Learning rate и число шагов



Меньший η даёт меньшую поправку на каждом шаге и обычно требует больше деревьев.

Настраиваем η и число шагов M совместно.

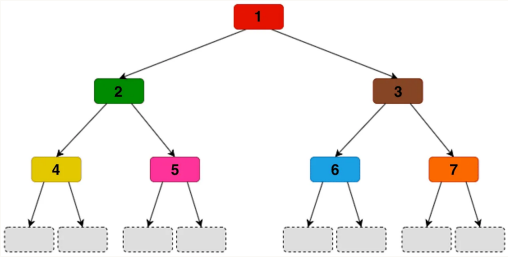
Early stopping: выбираем лучшую стадию по validation.

Регуляризация бустинга

Настройка	Как ограничивает подгонку
Learning rate η	Уменьшает вклад нового дерева.
Число шагов M	Ограничивает число поправок.
Глубина / число листьев	Ограничивает сложность одной поправки.
Минимальный размер листа	Ограничивает подгонку под отдельные объекты.
Подвыборки строк и признаков	Добавляют случайность в обучение.
Штраф на значения листьев	Сжимает слишком большие поправки.

Параметры подбираем совместно по validation или CV.

XGBoost



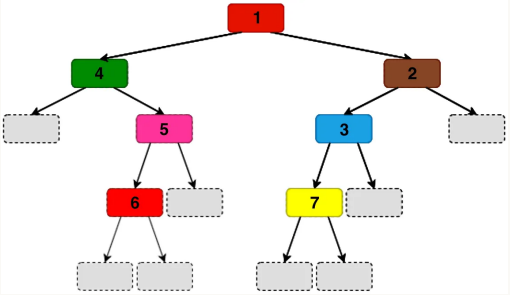
Depthwise: сначала делим узлы, ближайшие к корню.

На схеме порядок разбиений: $1 \to 2, 3 \to 4, 5, 6, 7$.

Ограничиваем глубину и вводим штрафы за сложность деревьев.

Есть и `grow_policy=lossguide`: выбор листа с наибольшим выигрышем.

LightGBM



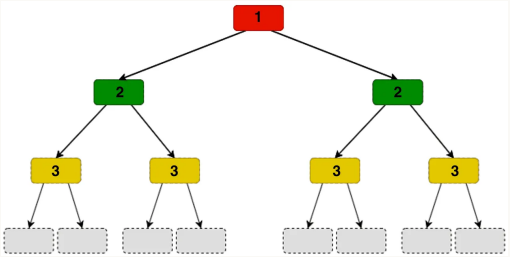
Leaf-wise: выбираем лист, чьё разбиение даёт наибольший выигрыш.

Дерево может расти неравномерно: одна ветвь заметно глубже других.

`num_leaves` ограничивает число **листьев**; `max_depth` — глубину.

Для числовых признаков используются гистограммы значений.

CatBoost



Симметричное дерево: на одном уровне все узлы используют одинаковый предикат.

Для полной глубины d получается 2^d листьев.

CatBoost умеет обрабатывать категориальные признаки; передаём их явно.

Симметричные деревья — стандартная политика; доступны также Depthwise и Lossguide.

CatBoost: кодирование категорий

Ordered target statistics: кодируем категорию по таргетам только **предыдущих строк** случайной перестановки train.

Порядок	Категория	y	Код
1	A	1	0.50
2	B	0	0.50
3	**A**	**0**	**0.75**
4	A	1	0.50
5	B	1	0.25

Для строки 3 раньше у категории A был только $y_1 = 1$.

$$z_3 = \frac{1 + 0.5}{1 + 1} = 0.75.$$

Текущий $y_3 = 0$ **не входит в код**: иначе возникает утечка таргета.

В примере бинарный y, сглаживание: prior 0.5 с весом 1.

Validation/test кодируем по статистикам train. При небольшом числе категорий также используется one-hot.

Стекинг без утечки

В voting усредняем прогнозы; в стекинге учим **мета-модель**:

$$z(x) = (a_1(x), \dots, a_M(x)), \qquad F(x) = g(z(x)).$$

Train делим на фолды A, B, C. Собираем **OOF-прогнозы**:

Базовые модели обучаем на	Прогнозируем для
$B \cup C$	A
$A \cup C$	B
$A \cup B$	C

Каждая строка получает прогноз модели, которая её не видела.

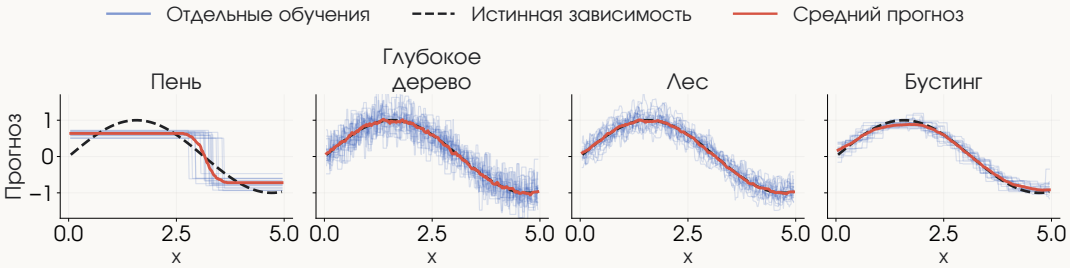
После обучения g базовые модели переобучаем на всём train. Новые прогнозы передаём в g; всю композицию оцениваем отдельно.

Смещение и разброс

Bias–variance tradeoff: объясняем поведение моделей

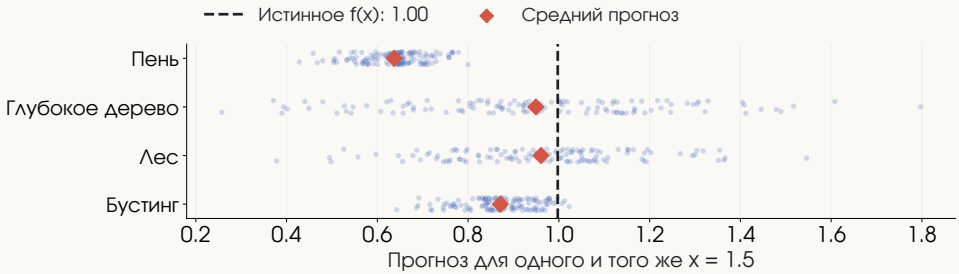
3

Прогнозы на разных выборках



Одна зависимость $y = \sin x + \varepsilon$, $\varepsilon \sim \mathcal{N}(0, 0.3^2)$. Для каждой модели обучаем 120 вариантов на новых выборках по 80 объектов.

Bias и variance в одной точке



- **Bias**: отличие среднего прогноза от истинного $f(x)$.
- **Variance**: разброс прогнозов при смене обучающей выборки.

Разложение ошибки прогноза

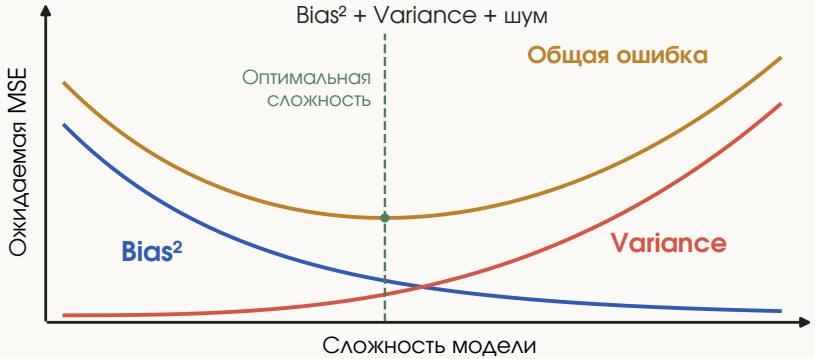
Для фиксированного x: $y = f^*(x) + \varepsilon$, $\mathbb{E}[\varepsilon \mid x] = 0$. Обучающая выборка D и новый шум независимы.

$$\mathbb{E}_{D,\varepsilon}\left[(y - F_D(x))^2\right] = \underbrace{\left(\mathbb{E}_D F_D(x) - f^*(x)\right)^2}_{\text{Bias}^2} + \underbrace{\mathrm{Var}_D(F_D(x))}_{\text{Variance}} + \underbrace{\sigma^2(x)}_{\text{шум}}.$$

- **Смещение**: систематическая ошибка среднего прогноза.
- **Разброс**: чувствительность к обучающим данным.
- **Шум**: вариативность ответа, не объяснимая известными признаками.

Разложение относится к **квадратичной ошибке**; это не разложение accuracy.

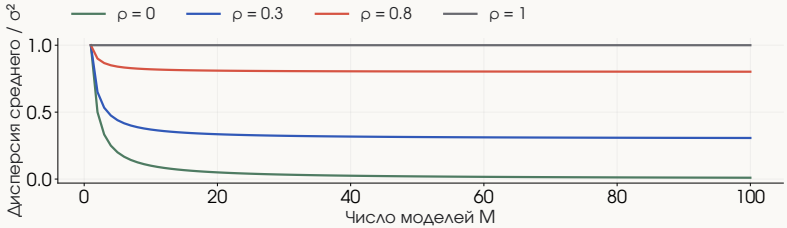
Bias–variance tradeoff



Выбираем минимум общей ошибки на новых данных.

Слишком простая модель недообучается; слишком гибкая может переобучаться.

Корреляция ошибок и усреднение

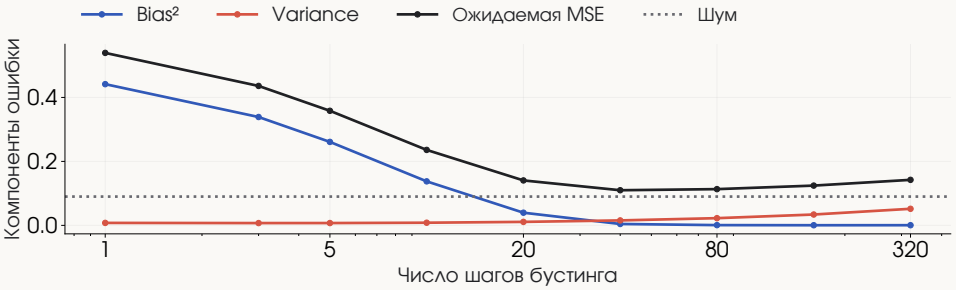


Если дисперсия каждого прогноза σ_a^2, а общая попарная корреляция ρ:

$$\mathrm{Var}\left(\frac{1}{M}\sum_{m=1}^{M} a_m\right) = \rho\sigma_a^2 + \frac{1-\rho}{M}\sigma_a^2.$$

Одинаковые прогнозы усреднять бесполезно. Bootstrap и случайные признаки помогают получить разные деревья.

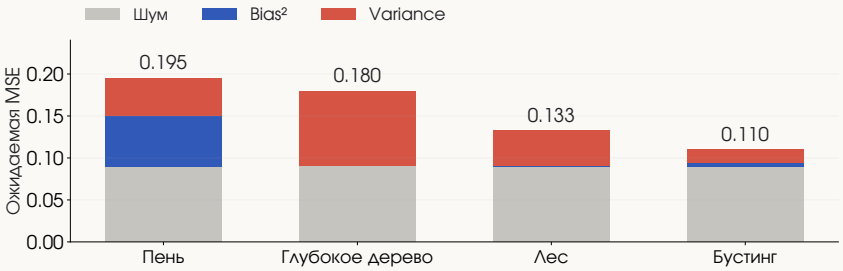
Bias–variance в бустинге



В начале новые поправки уменьшают недообучение. Позже усиливается подгонка особенностей train.

Бустинг может менять **и Bias, и Variance**: число шагов выбираем по validation.

Компоненты ошибки разных моделей



Те же данные и повторные обучения: пень, глубокое дерево, лес из 64 деревьев и бустинг из 40 деревьев глубины 2.

Какая часть ошибки мешает каждому из этих вариантов?

Теоретический минимум

- **Типы ансамблей**: voting и усреднение, бэггинг, бустинг, стекинг.
- **Bootstrap. Случайный лес. Out-of-bag (OOB).**
- **Градиентный бустинг**: остатки, антиградиент функции потерь.
- **Регуляризация ансамблей**: learning rate, сложность деревьев, early stopping.
- **Политики роста деревьев**: depth-wise, leaf-wise, симметричные деревья.
- **Кодирование категорий**: one-hot, ordered target statistics.
- **Bias, variance, неустранимый шум. Bias–variance tradeoff.**

Ко(т)нец презентации

Вопросы?

