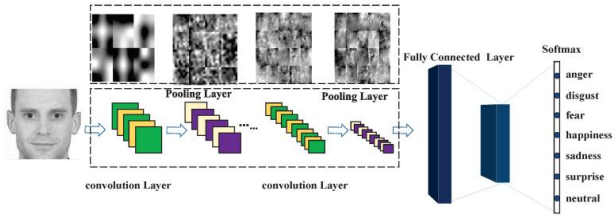
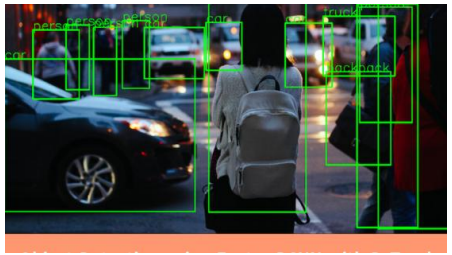
Введение в Машинное обучение











Лекция

Ансамбли моделей на примере решающих деревьев

к.ф.-м.н., доцент УИ
Корнаева Е.П.

Ансамбли моделей на примере решающих деревьев

Понятие смещения (bias) и разброса (variance) модели

Другими словами:

- **Смещение (bias)** оценивает насколько в среднем модель хорошо предсказывает целевую переменную.
- **Разброс (variance)** оценивает устойчивость модели к изменениям в обучающей выборке.

Как улучшить качество модели?

- либо повышать её сложность (уменьшать смещение);
- либо повышать устойчивость (понижать разброс).

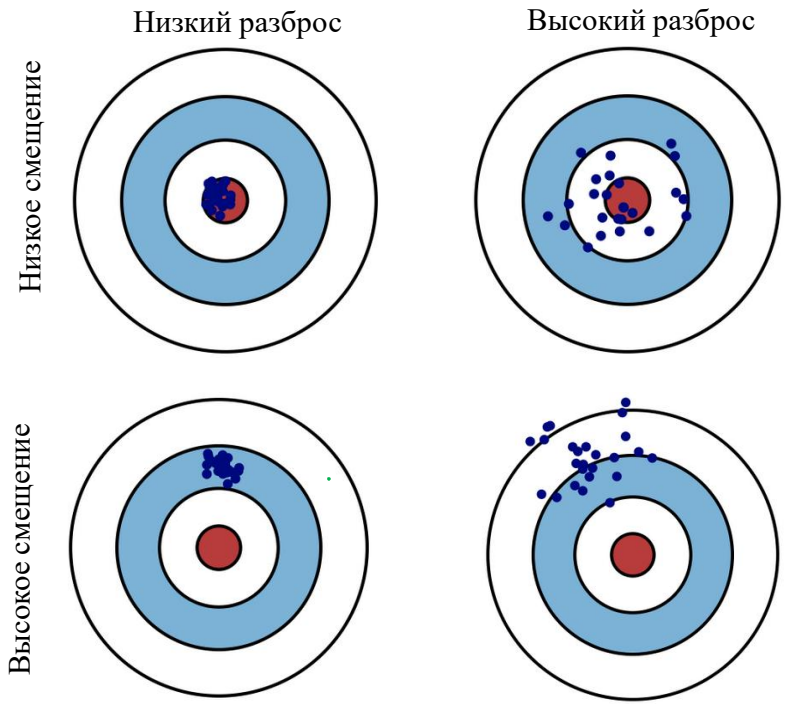


Рис.1. Смещение и разброс [2]

[2] https://habr.com/ru

Обучение с учителем: Решающие деревья

Понятие смещения (bias) и разброса (variance)

Например, для регрессии:

$$y(x) = f(x) + \varepsilon$$

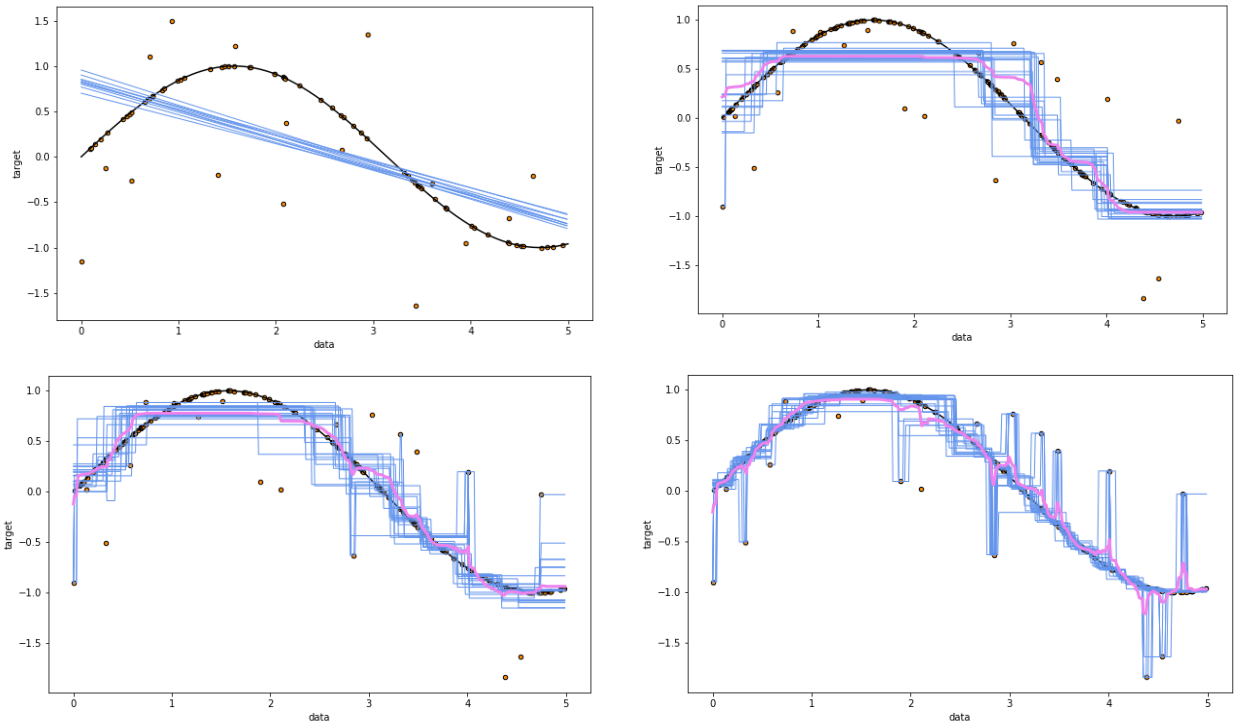


Рис.2. Смещение и разброс на примере регрессии [3]

[3] https://base.mnmc.hse.ru

Ансамбли моделей на примере решающих деревьев

Понятие смещения (bias) и разброса (variance) модели

Смещение (bias) - это систематическая ошибка модели, которая показывает насколько сильно усредненные предсказания модели отличаются от реального (истинного) значения.
Разброс (variance) – это ошибка, возникающая из-за чувствительности к малым отклонениям в обучающей выборке.

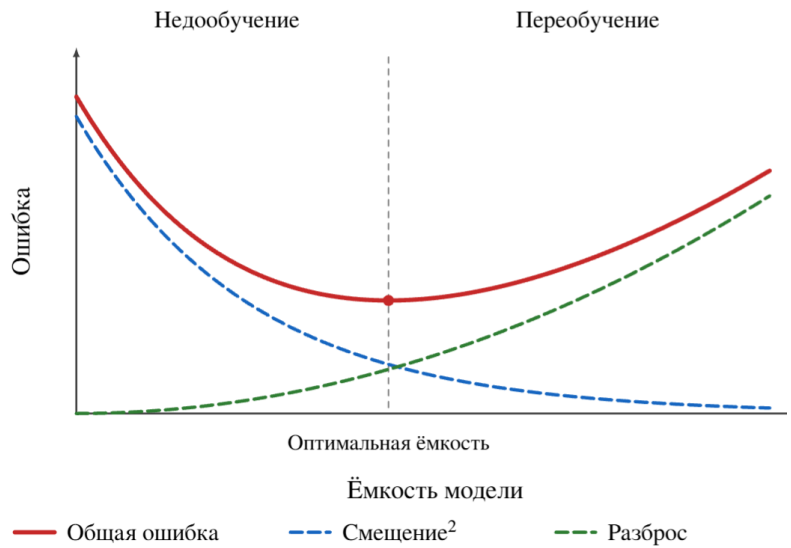


Рис.3. Ошибка модели схематично [1]

[1] Гудфеллоу Я., Бенджио И., Курвилль А. Глубокое обучение / пер. с анг. А. А. Слинкина. – 2-е изд., испр. – М.: ДМК Пресс, 2018. – 652 с.

Ансамбли моделей на примере решающих деревьев

Понятие смещения (bias) и разброса (variance) модели

$M_D[(y-\tilde{y}(x))^2]$ - математическое ожидание квадрата ошибки (по всевозможным выборкам D)

$y = f(x) + \varepsilon$ – реальное значение отклика

$f(x)$ – идеальная функция, $\varepsilon \sim N(0, \sigma^2)$ – разброс в данных (шум), причем $\mathrm{r}(\varepsilon_k, \varepsilon_j) = 0$

$\tilde{y}(x)$ – прогноз модели, $\bar{y}(x) = M_D[\tilde{y}(x)]$ – среднее значение прогноза по всем выборкам D

$$y - \tilde{y}(x) = y - \bar{y}(x) + \bar{y}(x) - \tilde{y}(x) = f(x) + \varepsilon - \bar{y}(x) + \bar{y}(x) - \tilde{y}(x) =$$

$$= \underbrace{\left(f(x) - \bar{y}(x)\right)}_{\text{const}} + \underbrace{\left(\bar{y}(x) - \tilde{y}(x)\right)}_{\text{случ.вел.}} + \underbrace{\varepsilon}_{\text{шум}}$$

a B C, const

$$\mathrm{M}[(y-\tilde{y}(x))^2] = M\left(a^2 + B^2 + C^2 + 2aB + 2aC + 2BC\right) = M(a^2) + M(B^2) + M(C^2) + 2M(aB) + 2M(aC) + 2M(BC) =$$

где $M(a^2) = M\left((f(x) - \bar{y}(x))^2\right) = bias^2$

$M(B^2) = M\left((\tilde{y}(x) - \bar{y}(x))^2\right) = var(\tilde{y}(x))$

$M(C^2) = var(\varepsilon) = \sigma^2$ (т.к. $var(\varepsilon) \overset{\text{св-во дисп.}}{=} M(\varepsilon^2) - M^2(\varepsilon) = M(\varepsilon^2)$, $M(\varepsilon) = 0$)

$M(aB) \overset{\text{const}}{=} aM(B) = aM\left(\bar{y}(x) - \tilde{y}(x)\right) = a\left(M(\bar{y}(x)) - M(\tilde{y}(x))\right) = a\left(\bar{y}(x) - \bar{y}(x)\right) = 0$

$M(aC) = aM(\varepsilon) = 0$

$M(BC) \overset{*}{=} M(B)M(C) = 0$

* B и C – независимые сл. вел., т.к. $\mathrm{r}(\varepsilon_k, \varepsilon_j) = 0$ (т.е. ошибки в одном опыте не влияют на ошибки в другом опыте, т.е. ошибки (шум) в обуч. выборке не коррелированы с ошибками в тестовой выборке. $\tilde{y}(x)$ - постр. по обуч. выборке, а ошибка модели считается на тестовой)

$$= bias^2 + var + \sigma^2$$

Ансамбли моделей на примере решающих деревьев

Понятие смещения (bias) и разброса (variance) модели

$M_D[(y - \tilde{y}(x))^2]$ - математическое ожидание квадрата ошибки (по всевозможным выборкам D)

$y = f(x) + \varepsilon$ – реальное значение отклика

$f(x)$ – идеальная функция, $\varepsilon \sim N(0, \sigma^2)$ – разброс в данных (шум), причем $r(\varepsilon_k, \varepsilon_j) = 0$

$\tilde{y}(x)$ – прогноз модели, $\bar{y}(x) = M_D[\tilde{y}(x)]$ – среднее значение прогноза по всем выборкам D

$$M[(y - \tilde{y}(x))^2] = bias^2 + var(\tilde{y}(x)) + \sigma^2$$

данные; оценка модели; смещение; разброс; шум

Как улучшить качество модели?

- либо повышать её сложность (уменьшать смещение);
- либо повышать устойчивость (понижать разброс).

Ансамбли моделей на примере решающих деревьев

Мотивация: отдельному дереву свойственно переобучение, т.е. модель на основе дерева – неустойчивая.

Что будет, если деревья обучать на разных подвыборках?

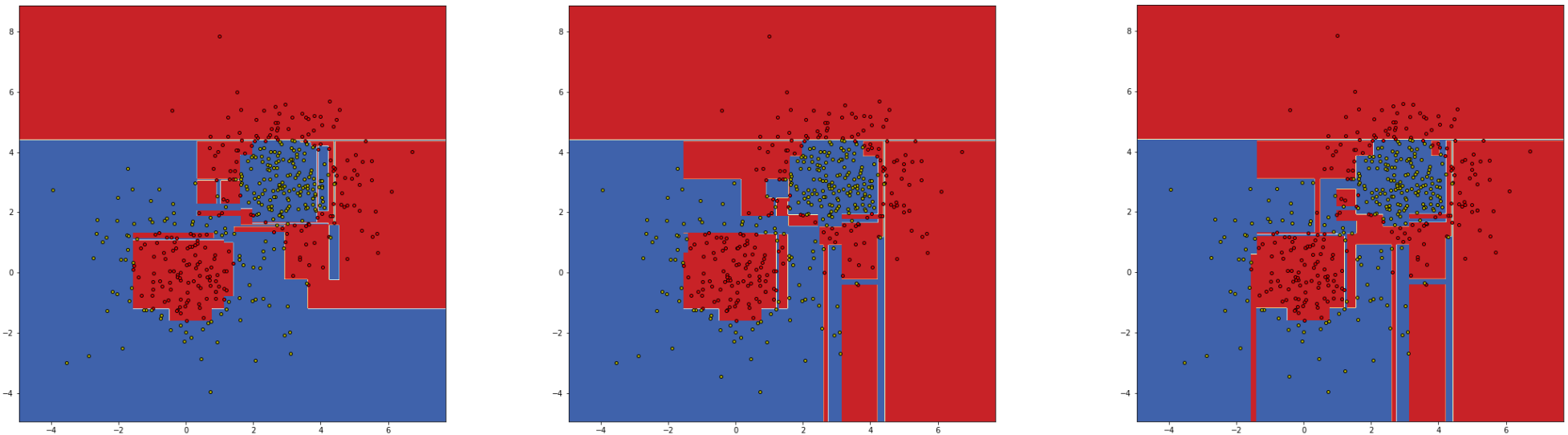


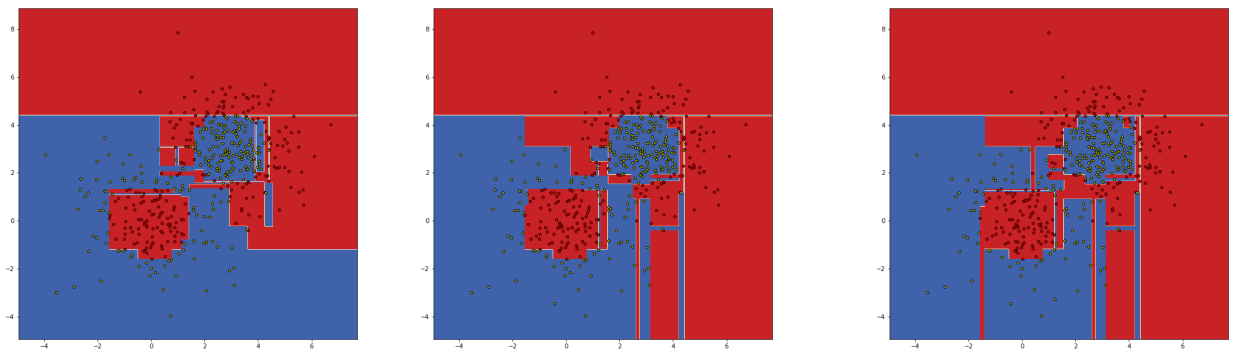
Рис.4. Чувствительность решающих деревьев к изменениям в исходных данных [1]

[1] https://base.mnmc.hse.ru

Ансамбли моделей на примере решающих деревьев

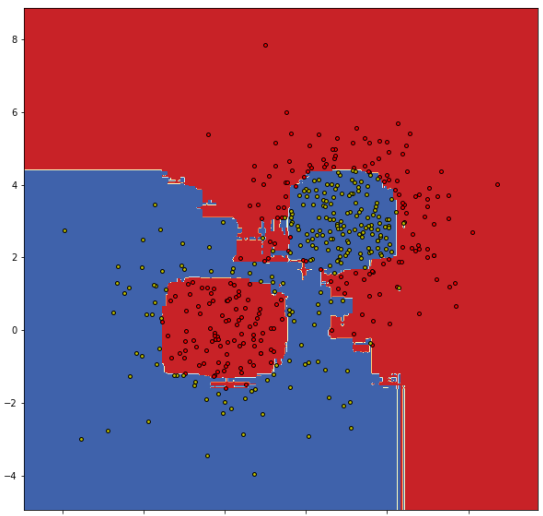
Мотивация: отдельному дереву свойственно переобучение, т.е. модель на основе дерева – неустойчивая.

Что будет, если деревья обучать на разных подвыборках?



$\tilde{y}_1(x)$ $\tilde{y}_2(x)$ ... $\tilde{y}_K(x)$

$\Rightarrow$



$\tilde{y} \quad (x)$

Рис.5. Композиция моделей [3]

[3] https://base.mnmc.hse.ru

Ансамбли моделей на примере решающих деревьев

Как по одной выборке построить K различных моделей?

- **Обучать модели на различных подвыборках** (бэггинг, случайный лес);
  - Обучать модели последовательно для корректирования ошибки (градиентный бустинг).

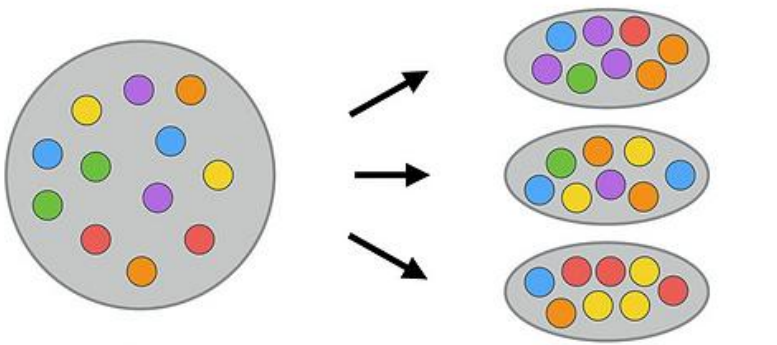
- **Идея**: Обучать модели на различных подвыборках: строится K базовых моделей, а прогноз строить по K различным моделям:
  - голосованием по большинству (классификация): $\tilde{y}(x) = \arg\max_{y \in \mathbb{Y}} \sum_{k=1}^{K} [\tilde{y}_k(x) = y]$
  - усреднением (регрессия): $\tilde{y}(x) = \frac{1}{K} \sum_{k=1}^{K} \tilde{y}_k(x)$

где $\tilde{y}_k(x)$ - решающие деревья, построенные на подвыборках

Ансамбли моделей на примере решающих деревьев

Бэггинг (Bagging от Bootstrap aggregating) - это метод машинного обучения, который строит ансамбль из независимых моделей, обучая каждую из них на своей случайной подвыборке исходных данных, а затем усредняет их прогнозы.

Процедура бутстрэпа



n – объем выборки n –объем каждой подвыборки

Рис.6. Бутстрап [4]

Замечание. Вхождение объекта несколько раз в выборку как бы повышает его вес.

Примерно 37% примеров остаются вне выборки бутстрэпа и не используются при построении k-го дерева

$p_i = \frac{1}{n}$	- вероятность извлечь один объект в одном испытании
$q_i = 1 - p_i$	- вероятность того, что i-й объект не будет выбран в одном испытании
$Q_i = \left(1 - \frac{1}{n}\right)^n$	- вероятность того, что i-й объект не выбран в каждом из n независимых испытаний (объект i не попал в бутстрап)
$Mx = n\left(1 - \frac{1}{n}\right)^n$	математическое ожидание количества «пропущенных» объектов
$\left(1 - \frac{1}{n}\right)^n$	Доля не отобранных объектов

для больших выборок: $\lim_{n\to\infty}\left(1 - \frac{1}{n}\right)^n = \frac{1}{e} \approx 0{,}367$

Ансамбли моделей на примере решающих деревьев

Бэггинг (**Bagging от Bootstrap aggregating**) **-** это метод машинного обучения, который строит ансамбль из независимых моделей, обучая каждую из них на своей случайной подвыборке исходных данных, а затем усредняет их прогнозы.

Пусть $\tilde{y}_k(x)$ - k[ая] модель в композиции $\tilde{y}(x)$ из K моделей

Чему равно смещение композиции $\tilde{y}(x)$?

$$var\left(\frac{1}{K}\sum_{k=1}^{K}\tilde{y}_k(x)\right) = \frac{1}{K^2}\left(\sum_{k=1}^{K} var\left(\tilde{y}_k(x)\right) + \sum_{\substack{k,q=1 \\ k\neq q}}^{K} cov\left(\tilde{y}_k(x), \tilde{y}_q(x)\right)\right)$$

Ансамбли моделей на примере решающих деревьев

Бэггинг (**Bagging от Bootstrap aggregating**) **-** это метод машинного обучения, который строит ансамбль из независимых моделей, обучая каждую из них на своей случайной подвыборке исходных данных, а затем усредняет их прогнозы.

Пусть $\tilde{y}_k(x)$ - k[ая] модель в композиции $\tilde{y}(x)$ из K моделей

Чему равно смещение композиции $\tilde{y}(x)$?

$$var\left(\frac{1}{K}\sum_{k=1}^{K}\tilde{y}_k(x)\right) = \frac{1}{K^2}\left(\sum_{k=1}^{K} var\big(\tilde{y}_k(x)\big) + \sum_{\substack{k,q=1 \\ k\neq q}}^{K} cov\left(\tilde{y}_k(x), \tilde{y}_q(x)\right)\right)$$

Допущения: модели $\tilde{y}_k(x)$ идентично распределены (обучены одним алгоритмом *на данных из одного распределения*)

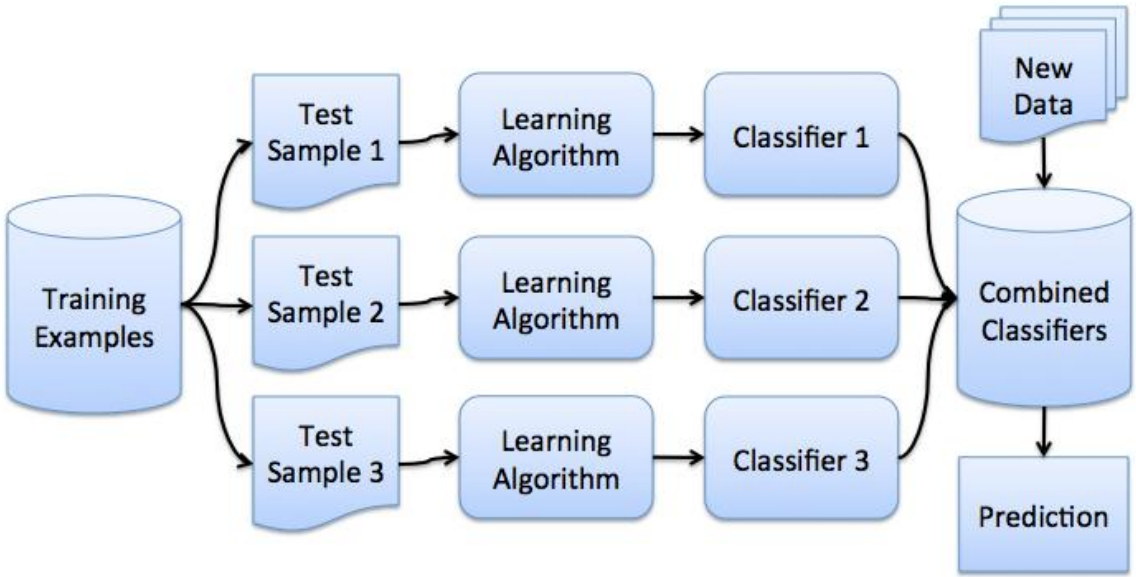
$$var\left(\frac{1}{K}\sum_{k=1}^{K}\tilde{y}_k(x)\right) = \frac{1}{K^2}\left(K\sigma^2(\tilde{y}_k(x)) + K(K-1)cov(\tilde{y}_k(x), \tilde{y}_q(x))\right) = \frac{1}{K}\sigma^2 + \frac{(K-1)}{K}cov$$

Таким образом, если базовые модели $\tilde{y}_k(x)$ независимы, то разброс композиции уменьшается в K раз!

* Для классификации дисперсия ансамбля получается аналогично, если и использовать “soft voting”: $\tilde{p} = \frac{1}{K}\sum_{k=1}^{K}\tilde{p}_k(x)$

Ансамбли моделей на примере решающих деревьев

Бэггинг (Bagging от Bootstrap aggregating) - это метод машинного обучения, который строит ансамбль из независимых моделей, обучая каждую из них на своей случайной подвыборке исходных данных, а затем усредняет их прогнозы.



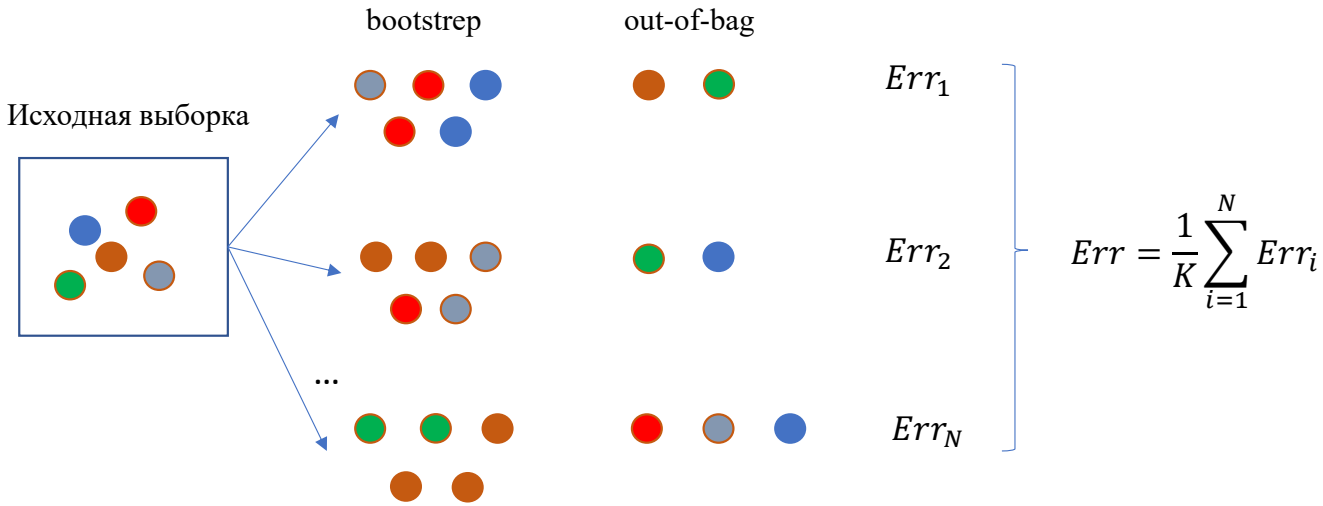
Замечание. Композиция моделей на основе Бэггинга:

- понижает разброс, но не меняет смещение модели;
- однако, чем более независимы базовые модели, тем сильнее понижается разброс модели;
- для эффективного понижения разброса надо строить независимые базовые модели композиции.

Ансамбли моделей на примере решающих деревьев

Бэггинг (Bagging от Bootstrap aggregating) - это метод машинного обучения, который строит ансамбль из независимых моделей, обучая каждую из них на своей случайной подвыборке исходных данных, а затем усредняет их прогнозы.

Каждое дерево обучается примерно на 63% данных, что можно сделать с оставшимися объектами?



Out-of-bag оценка (OOB оценка) - это усредненная оценка базовых алгоритмов на тех ~37% данных, на которых они не обучались

Ансамбли моделей на примере решающих деревьев

Метод случайных подпространств - ансамбль моделей, использующих метод случайного подпространства

Проблема бэггинга: Что если в данных есть один-два признака, которые лучше всего разделяют классы?
В этом случае почти все деревья начнут свои первые (и самые важные) разбиения именно с этих сильных признаков. В результате все деревья в ансамбле получатся очень похожими друг на друга.

Можно ли обучать базовые модели композиции на подпространстве признаков?

Подпространство признаков $\hat{X} \subseteq X$

	X_1	X_2	X_3	...	X_l	X_m	Y
1	x_{11}	x_{12}	x_{13}		x_{1l}	x_{1m}	y_1
2	x_{21}	x_{22}	x_{23}		x_{2l}	x_{2m}	y_2
...							...
n	x_{n1}	x_{n2}	x_{n3}		x_{nl}	x_{nm}	y_n

Ансамбли моделей на примере решающих деревьев

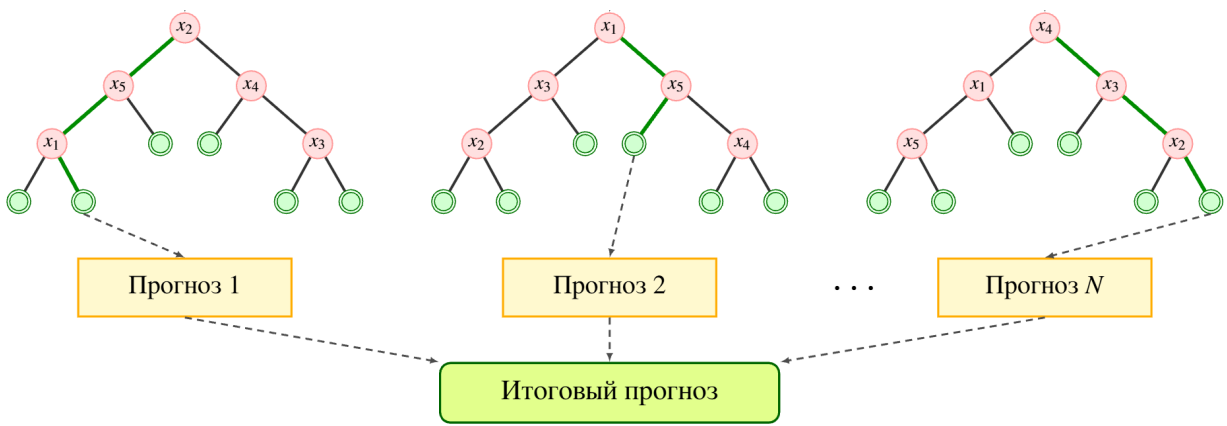
Метод случайных подпространств - ансамбль моделей, использующих метод случайного подпространства

Подпространство признаков $\hat{X} \subseteq X$

	X_1	X_2	X_3	...	X_l	X_m	Y
1	x_{11}	x_{12}	x_{13}		x_{1l}	x_{1m}	y_1
2	x_{21}	x_{22}	x_{23}		x_{2l}	x_{2m}	y_2
...							...
n	x_{n1}	x_{n2}	x_{n3}		x_{nl}	x_{nm}	y_n

m – мерность признакового пространства;

m' – мерность признакового подпространства для разбиения в узлах каждого дерева.



Выбор m' :

- для классификации $m' = \sqrt{m}$
- для регрессии $m' = \frac{m}{3}$

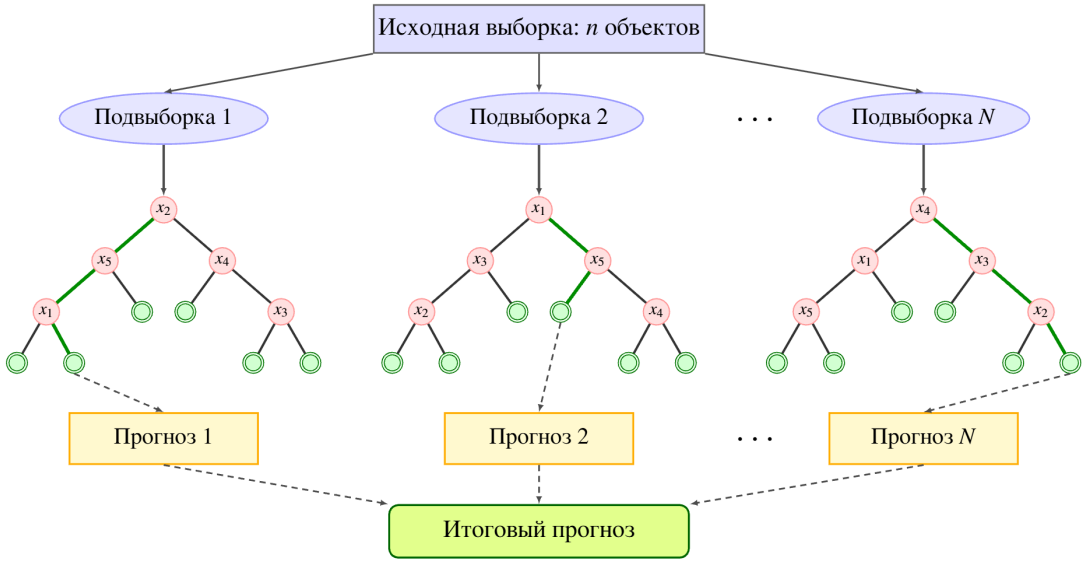
Ансамбли моделей на примере решающих деревьев

Случайный лес (random forest) - ансамбль моделей, использующих бэггинг и метод случайного подпространства [Лео Брейман, 2001]

Подпространство признаков $\hat{X} \subseteq X$

случайная подвыборка

	X_1	X_2	X_3	...	X_l	X_m	Y
1	x_{11}	x_{12}	x_{13}		x_{1l}	x_{1m}	y_1
2	x_{21}	x_{22}	x_{23}		x_{2l}	x_{2m}	y_2
...							...
n	x_{n1}	x_{n2}	x_{n3}		x_{nl}	x_{nm}	y_n



Ансамбли моделей на примере решающих деревьев

Как по одной выборке построить K различных моделей?

- Обучать модели на различных подвыборках (бэггинг, случайный лес);
- **Обучать модели последовательно для корректирования ошибки** (бустинг).

Проблемы бэггинга:
- если базовая модель окажется смещённой, то и композиция будет смещенной;
- базовые модели долго обучать и применять, дорого хранить.

Идея бустинга:
- ✓ композиция моделей строится последовательно и жадно, начиная с простой базовой модели;
- ✓ каждая следующая модель строиться так, чтобы максимально корректировать ошибки предыдущих моделей.

Бустинг - это метод машинного обучения, который строит ансамбль из «слабых» моделей (чаще всего неглубоких деревьев решений) **последовательно**, где каждая следующая модель пытается исправить ошибки, допущенные предыдущими.

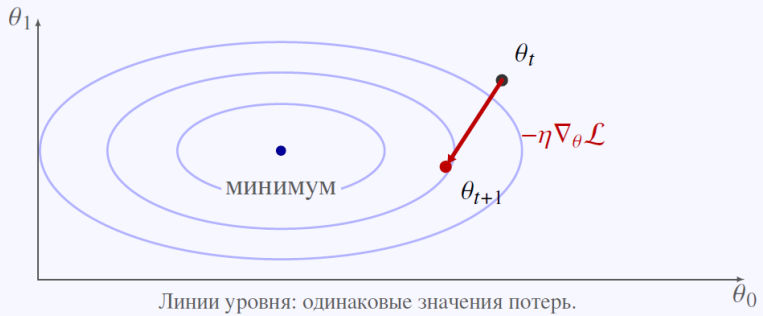
Ансамбли моделей на примере решающих деревьев

Градиентный бустинг строит ансамбль из слабых моделей последовательно, где каждая новая модель обучается предсказывать *антиградиент функции потерь* текущего ансамбля.

Простыми словами: это градиентный спуск, но не в пространстве чисел (весов), а в пространстве функций.

Пространство параметров

Например, прямая: $f_\theta(x) = \theta_0 + \theta_1 x$



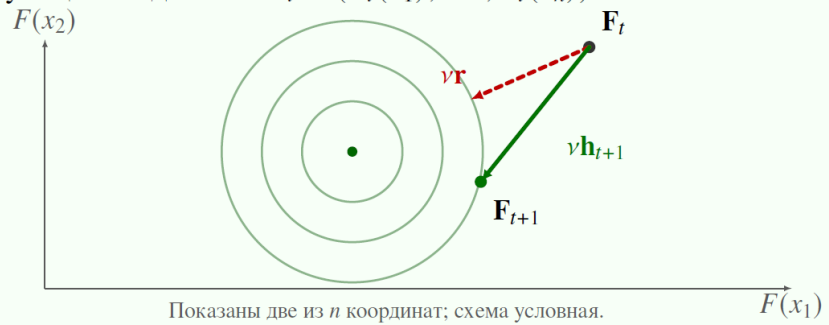
Линии уровня: одинаковые значения потерь.

$$\theta_{t+1} = \theta_t - \eta\nabla_\theta\mathcal{L}(\theta_t)$$

Меняем коэффициенты θ_0, θ_1.
Семейство моделей остаётся тем же.

Пространство функций

Функция на данных: $\mathbf{F}_t = (F_t(x_1), \ldots, F_t(x_n))$



Показаны две из n координат; схема условная.

$$F_{t+1}(x) = F_t(x) + \nu h_{t+1}(x)$$

Дерево учится на $r_i = y_i - F_t(x_i)$:
$h_{t+1}(x_i) \approx r_i = -\partial\mathcal{L}/\partial F(x_i)$.

Главное отличие: в бустинге дерево приближает антиградиент и задаёт поправку для любого x.
Пунктир – желаемый шаг; зелёная стрелка – доступная поправка деревом, обычно не точный антиградиент.

Ансамбли моделей на примере решающих деревьев

Градиентный бустинг на примере задачи регрессии $\{X_i, y_i\}$

$$L(F(x), y) = \frac{1}{2}\sum_{i=1}^{n}\left(y_i - F(x_i)\right)^2 \to min$$

$F_0(x)$ - базовая модель (например, константная модель $F_0(x) = \bar{y}$)

Концептуально: следующая модель строится двигаясь в пространстве функций в направлении антиградиента:

$$F_1(x) = F_0(x) - \nu\nabla_F L(F(x), y)|_{F_0}$$

Величину антиградиента $-\nabla_F L(F(x))|_{F_t} = \left[\frac{\partial L}{\partial F(x_i)}\right]|_{F_t}$ называют **псевдо-остатками**

Что такое псевдо-остатки для MSE?

Ансамбли моделей на примере решающих деревьев

Градиентный бустинг на примере задачи регрессии $\{X_i, y_i\}$

$$L(F(x), y) = \frac{1}{2}\sum_{i=1}^{n}\left(y_i - F(x_i)\right)^2 \to min$$

$F_0(x)$ - базовая модель (например, константная модель $F_0(x) = \bar{y}$)

Концептуально: следующая модель строится двигаясь в пространстве функций в направлении антиградиента:

$$F_{t+1}(x) = F_t(x) - \nu\nabla_F L(F(x), y)\,|_{F_t}$$

Величину антиградиента $-\nabla_F L(F(x))|_{F_t} = \left[\frac{\partial L}{\partial F(x_i)}\right]|_{F_t}$ называют **псевдо-остатками**

Псевдо-остатки для MSE на i-ом объекте для модели $F_t(x)$:

$$-\nabla_F L(F(x), y)|_{F_t(x_i)} = y_i - F_t(x_i) = r_{ti}$$

Замечание. Для другого лосса псевдо-остатки будут отличатся от «остатков» в обычном смысле.

Ансамбли моделей на примере решающих деревьев

Градиентный бустинг на примере задачи регрессии $\{X_i, y_i\}$

$$L(F(x), y) = \frac{1}{2}\sum_{i=1}^{n}\left(y_i - F(x_i)\right)^2 \to min$$

Концептуально: следующая модель строится двигаясь в пространстве функций в направлении антиградиента:

$$F_{t+1}(x) = F_t(x) - \nu\nabla_F L(F(x), y)\,|_{F_t} \qquad (*)$$

Величину антиградиента $-\nabla_F L(F(x), y)|_{F_t} = \left[\frac{\partial L}{\partial F(x_i)}\right]|_{F_t}$ называют **псевдо-остатками**

Псевдо-остатки для MSE на i-ом объекте для модели $F_t(x)$: $-\nabla_F L(F(x), y)|_{F_t(x_i)} = y_i - F_t(x_i) = r_{ti}$

Таким образом, чтобы следующая модель исправляла базовый прогноз $F_0(x)$ нужно построить новую модель*:

дерево $h_1(x)$: $h_1(x) = \underset{h}{\operatorname{argmin}} \sum_{i=1}^{n}\left(r_{1i} - h(x_i)\right)^2$

Тогда ф. $(*)$ на втором шаге примет вид:

$$F_1(x) = F_0(x) + \nu h_1(x)$$

* модель $h_t(x)$ аппроксимирует антиградиент лосса (в случае MSE - настоящие остатки модели)

Ансамбли моделей на примере решающих деревьев

Градиентный бустинг на примере задачи регрессии $\{X_i, y_i\}$

$$F_{t+1}(x) = F_0(x) + \nu \sum_{i=1}^{t+1} h_i(x)$$

$h_{t+1}(x) = \underset{h}{\operatorname{argmin}} \sum_{i=1}^{n} \left(r_{ti} - h(x_i)\right)^2$ - деревья, аппроксимирующие псевдо-остатки r_{ti} модели $F_t(x)$;

ν – шаг обучения;

$r_{ti} = y_i - F_t(x_i)$ - псевдо-остатки для текущей модели.

Ансамбли моделей на примере решающих деревьев

Градиентный бустинг на примере задачи регрессии $\{X_i, y_i\}$

Пошаговый алгоритм (упрощенно):

Инициализация: Ансамбль начинает с простого константного прогноза $F_0(x) = \bar{y}$

1. Цикл (для каждого нового дерева):

Шаг А: Для каждого объекта вычисляются **псевдо-остатки** для каждого объекта.

Шаг В: Обучается новое дерево $h_t(x)$ предсказывать эти псевдо-остатки.

Шаг С: Обновляется ансамбль: $F_{t+1}(x) = F_t(x) + \nu h_t(x)$, где ν — это **скорость обучения (learning rate)**. Она не дает новому дереву слишком резко менять прогноз, заставляя алгоритм учиться медленно и осторожно.

2. Повтор: Шаги повторяются много раз, пока ошибка не перестанет уменьшаться.

3. Финал: Предсказание для нового объекта x_{new}: $y(x_{new}) = F_0 + \nu \sum_{t=1}^{T} h_t(x_{new})$

GradientBoostingRegressor()

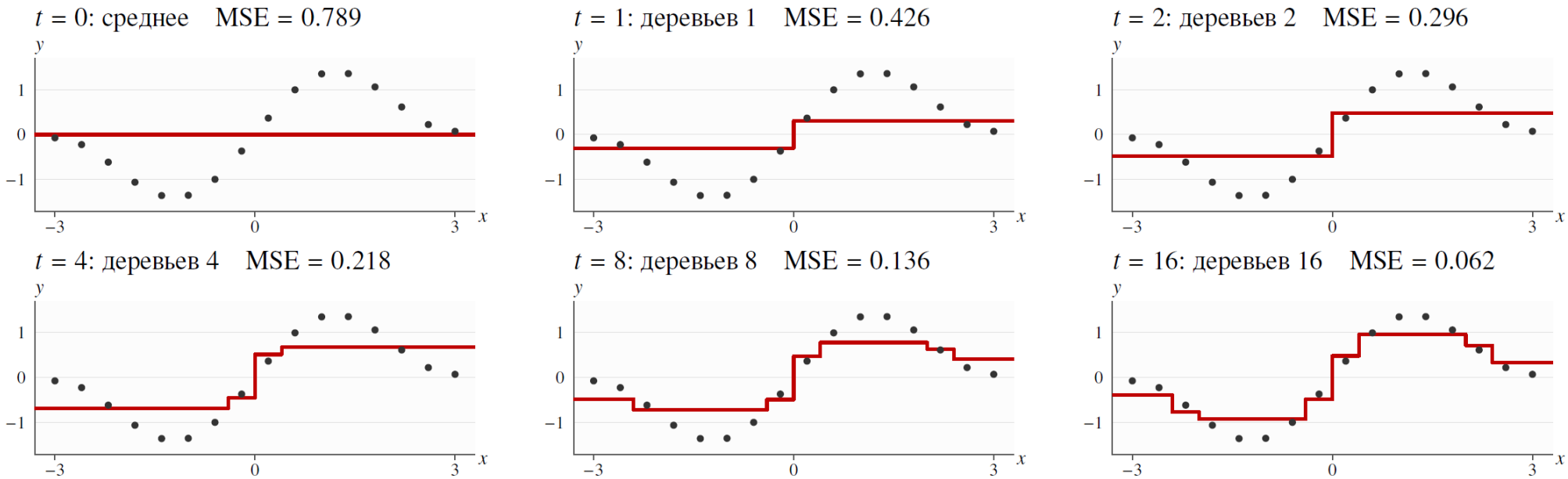
Ансамбли моделей на примере решающих деревьев

Градиентный бустинг на примере задачи регрессии $\{X_i, y_i\}$

Учебные данные: 16 точек, $y_i = \sin(1.5x_i) + 0.35x_i, x_i \in [-3, 3]$.
Деревья-пни (два листа), $\nu = 0.4$; все графики в одном масштабе.
MSE считаем на этих же обучающих точках.

- Обучающие точки (x_i, y_i) — Прогноз ансамбля $F_t(x)$

$F_t = F_0 + 0.4 \sum_{s=1}^{t} h_s, \quad F_0 = \bar{y} = 0$



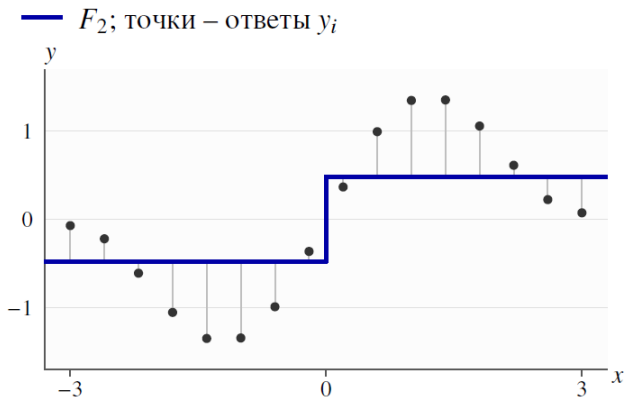
Каждый шаг добавляет простую ступеньку. Сумма деревьев всё точнее описывает обучающие данные.

Ансамбли моделей на примере решающих деревьев

Градиентный бустинг на примере задачи регрессии $\{X_i, y_i\}$

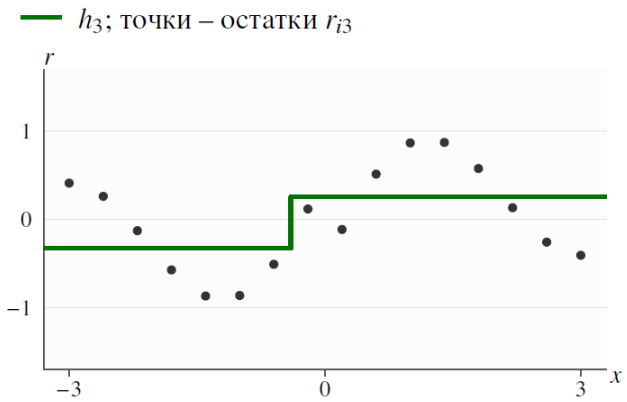
Те же данные и $\nu = 0.4$.
Берём текущий прогноз F_2, считаем остатки $r_{3_i} = y_i - F_2(x_i)$,
обучаем дерево h_3 на этих остатках.
В ансамбль добавляем $0.4h_3$, а не всё дерево целиком.

1. Текущий ансамбль



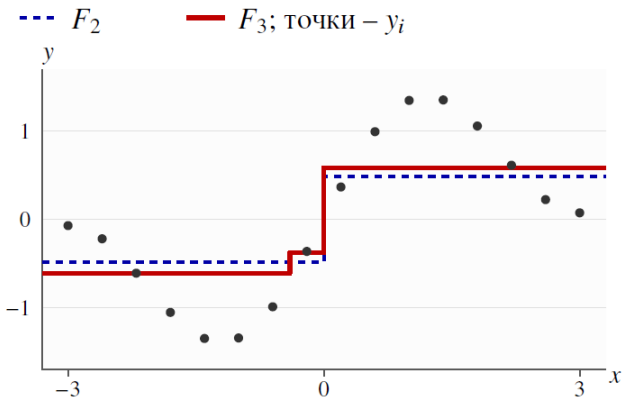
$r_{i3} = y_i - F_2(x_i)$
Серые отрезки показывают ошибки текущего ансамбля.

2. Дерево на остатках



$h_3(x) = -0.326$ при $x \le -0.4$;
$h_3(x) = 0.254$ при $x > -0.4$.
Это средние остатки в листьях.

3. Обновлённый прогноз



$F_3 = F_2 + 0.4h_3$
MSE: 0.296 $\longrightarrow$ 0.243.
Затем остатки пересчитываем.

Числа округлены. Уменьшение ошибки на обучении ещё не гарантирует улучшения на новых данных.

Ансамбли моделей на примере решающих деревьев

Градиентный бустинг на примере задачи бинарной классификации $y \in \{0,1\}$

$$p = sigm(F(x)) = \frac{1}{1-\exp(-F(x))}, \qquad L(p(F), y) = -(yln(p) + (1-y)\ln(p)) \to min$$

$$y_{pred} = [p > 0.5]$$

Пошаговый алгоритм (упрощенно):

Инициализация: Ансамбль начинает с простого константного прогноза $F_0(x) = ln\frac{\bar{y}}{1-\bar{y}}$

1. Цикл (для каждого нового дерева):

Шаг A: Для каждого объекта вычисляются **вероятности**, используя текущую модель: $p_i = sigm(F(x_i))$

Шаг B: вычисляются **псевдо-остатки** для каждого объекта: $r_i = y_i - p_i$

Шаг C: Обучается новое дерево $h_t(x)$ предсказывать эти псевдо-остатки.

Шаг D: Обновляется ансамбль: $F_t(x) = F_{t-1}(x) + \nu h_t(x)$, где ν — это **скорость обучения (learning rate)**.

Она не дает новому дереву слишком резко менять прогноз, заставляя алгоритм учиться медленно и осторожно.

2. Повтор: Шаги повторяются много раз, пока ошибка не перестанет уменьшаться.

3. Финал: Предсказание для нового объекта x_{new}:

1. $F_T(x_{new}) = F_0 + \nu \sum_{t=1}^{T} h_t(x_{new})$
2. $p(x_{new}) = sigm(F_T(x_{new}))$
3. $y_{pred} = [p(x_{new}) > 0.5]$

scikit learn GradientBoostingClassifier()

Ансамбли моделей на примере решающих деревьев

Сравнение

Характеристика	Бэггинг (Random Forest)	Бустинг (Gradient Boosting)
Как строятся модели	**Параллельно** и независимо друг от друга.	**Последовательно**, каждая следующая зависит от предыдущей.
Какие модели внутри	Глубокие, сложные деревья (сильные ученики).	Очень мелкие, простые деревья (слабые ученики, пни).
Что снижает	Снижает **Разброс (Variance)**.	Снижает **Смещение (Bias)**.
Устойчивость к шуму	Высокая (шум усредняется).	Низкая (бустинг пытается выучить даже шум, если не остановить вовремя, применяется регуляризация).

Ансамбли моделей на примере решающих деревьев

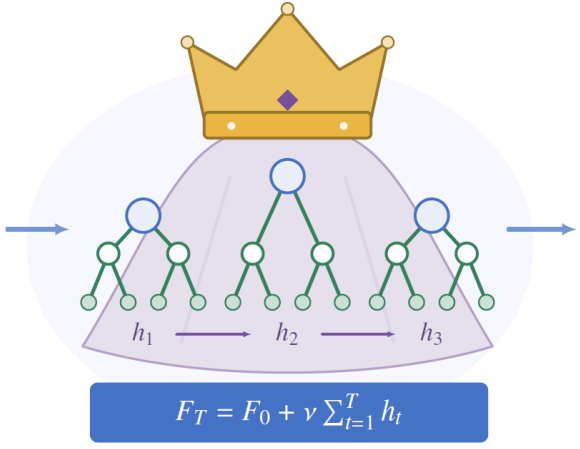
Градиентный бустинг – король табличных данных

Таблица признаков превращается в прогноз через последовательность деревьев-поправок

Табличные данные

x_1	x_2	x_3	y
2	0	7	1
5	1	3	0
1	1	8	1
4	0	2	0

Признаки X и ответы y для обучения



Прогноз

Число или
вероятность класса

Корону подтверждает качество на отложенных данных.

Сегодня индустрия использует три основных фреймворка, которые реализуют градиентный бустинг с разными инженерными оптимизациями:

- **XGBoost** (eXtreme Gradient Boosting) - классика, с которой начался бум.
- **LightGBM** (от Microsoft) - очень быстрый, использует гистограммы.
- **CatBoost** (от Яндекса) - лучше всех работает с категориальными признаками и требует меньше настройки.