Машинное обучение, ФКН ВШЭ
Семинар №10

1 Bias-variance decomposition

На лекции была выведена формула, демонстрирующая, как можно представить среднеквадратичную ошибку алгоритма регрессии в виде суммы трёх компонент:

$$L(\mu) = \mathbb{E}_{x,y}\big[\mathbb{E}_X\big[(y - \mu(X)(x))^2\big]\big] =$$
$$\underbrace{\mathbb{E}_{x,y}\big[(y - \mathbb{E}[y|x])^2\big]}_{\text{шум}} + \underbrace{\mathbb{E}_x\big[(\mathbb{E}_X[\mu(X)(x)] - \mathbb{E}[y|x])^2\big]}_{\text{смещение}} + \underbrace{\mathbb{E}_x\big[\mathbb{E}_X\big[(\mu(X)(x) - \mathbb{E}_X[\mu(X)(x)])^2\big]\big]}_{\text{разброс}}$$

- $\mu(X)$ — это алгоритм, обученный на конкретной выборке $X = \{(x_1, y_1), \ldots, (x_\ell, y_\ell)\}$
- $\mu(X)(x)$ — это предсказание алгоритма $\mu(X)$ для конкретного объекта x
- $\mathbb{E}_X$ — математическое ожидание по всем возможным выборкам X
- $\mathbb{E}_X[\mu(X)(x)]$ — это «средний» ответ алгоритма, обученного по всем возможным выборкам X, на объекте x

Шум — это неустранимая часть ошибки, он возникает из-за факторов, которые модель не может учесть. К ним относятся случайный характер имеющихся данных, погрешность измерений или нехватка признаков для точного описания объекта из выборки. Стоит понимать, что какой бы мощной ни была модель, от шума избавиться не удастся.

Смещение позволяет оценить, насколько хорошо с помощью алгоритма $\mu(X)$ можно приблизить истинную зависимость f. «Жёсткие» модели имеют высокое смещение, а «гибкие» — низкое. Например, модель линейной регрессии не справится с задачей приближения синусоиды: сколько бы данных не использовалось для обучения, алгоритм не сможет описать истинную зависимость. Глубокие деревья благодаря гибкости могут подстроиться под сложные зависимости и потому обладают низким смещением.

Разброс, в свою очередь, показывает, насколько сильно предсказания модели зависят от обучающей выборки. «Жёсткие» модели имеют низкий разброс, а «гибкие» — высокий: линейные модели ограничены классом функций, поэтому результаты разных обучений почти не будут отличаться. Глубокие деревья, как мы обсуждали на предыдущем семинаре, напротив, сильно зависят от того, какие объекты попали в обучающую выборку, и незначительное её изменение приведёт к новой структуре дерева. Следовательно, предсказания будут сильно колебаться, а, значит, глубокие деревья имеют высокий разброс.

§1.1 Пример расчёта оценок смещения и разброса

Рассмотрим задачу разложения на смещение и разброс на конкретном примере. Пусть истинная зависимость имеет вид $f(x) = 3\sin x$, а предсказания модели зашумлены, при этом шум имеет нормальное распределение с нулевым средним и дисперсией $\sigma^2 = 4$. Сгенерированные данные имеют такой вид, как на рис. 1.

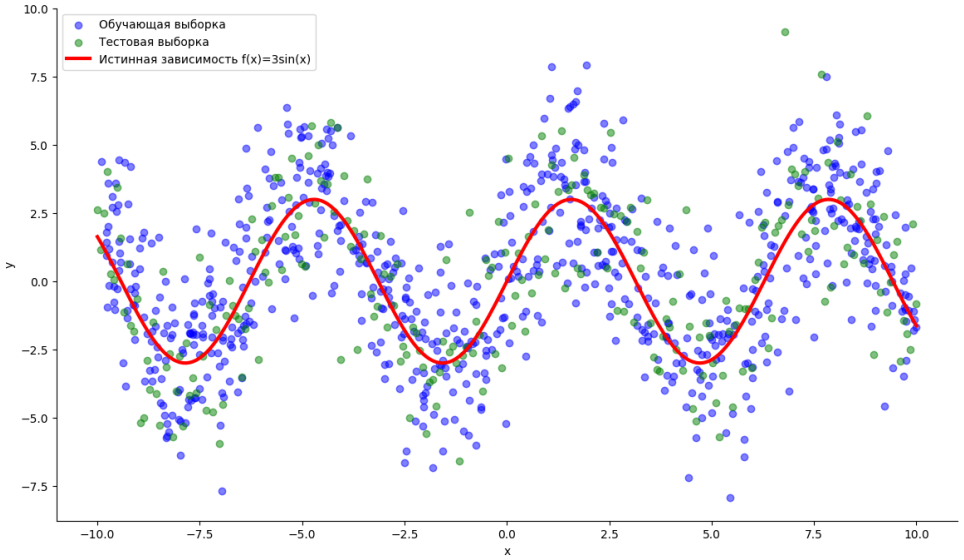

Рис. 1. Сгенерированные данные

Построим графики для понимания того, как предсказания решающих деревьев зависят от обучающей выборки и установленного значения максимальной глубины. На рис. 2 представлены результаты обучения деревьев на независимых подвыборках.

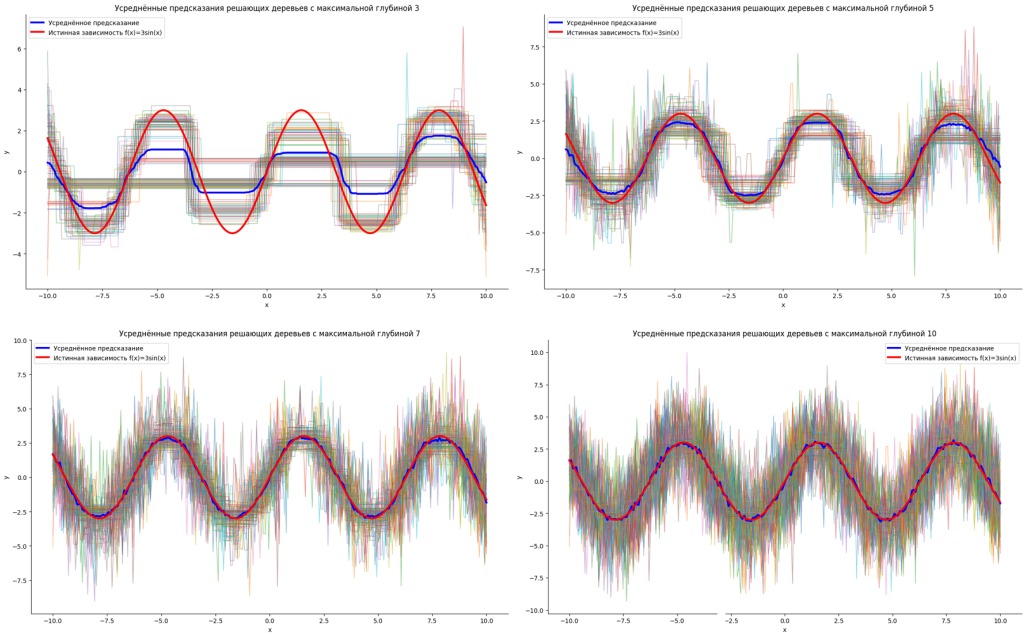

Рис. 2. Зависимость предсказаний от максимально допустимой глубины дерева

Полученные результаты позволяют предположить, что с увеличением максимально допустимой глубины решающего дерева смещение падает, а разброс — растёт. Для проверки выдвинутой гипотезы найдём компоненты разложения для решающих деревьев различной глубины (рис. 3). Код для обучения решающих деревьев и построения графиков можно найти по ссылке.

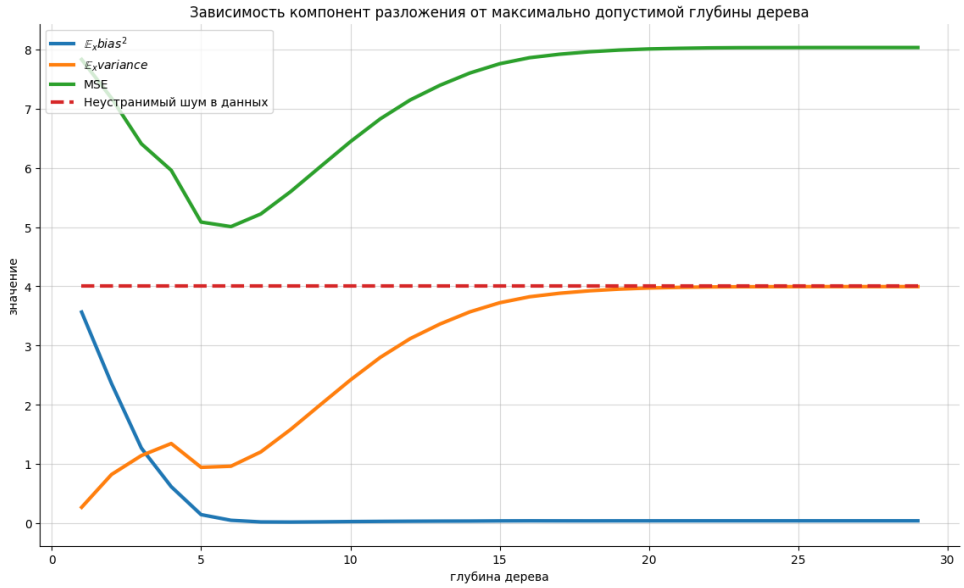

Рис. 3. Зависимость компонент разложения от максимально допустимой глубины дерева

Итак, выдвинутая гипотеза частично подтверждается: сначала значение смещения резко уменьшается, а разброса — растёт, то есть обученные на разных выборках модели действительно значительно отличаются друг от друга, однако усреднение значений практически идеально приближает корректный результат. Однако после того, как значение максимально допустимой глубины дерева достигает 20, разброс модели приближается к величине, обусловленной шумовой компонентой. Таким образом, слишком глубокие деревья идеально подстраиваются под зашумлённую выборку и теряют способность к обобщению.

§1.2 Разложение для линейной модели

Теперь рассмотрим задачу одномерной линейной регрессии, когда зависимость целевого признака y от объекта x моделируется с помощью примитивной линейной функции $y = kx$. В таком случае оптимальный параметр k находится путём минимизации $\text{MSE} = \sum_{i=1}^{\ell}(y_i - kx_i)^2$, в результате получается алгоритм $\mu(X) = k(X)x$, где $k(X) = \frac{\sum_i x_i y_i}{\sum_i x_i^2}$.

Будем считать, что объекты x генерируются из нормального распределения $x \sim p(x) = \mathcal{N}(0, \sigma_1^2)$. Правильный ответ на объекте x определяется зашумлённой функцией $f(x)$: $y = f(x) + \varepsilon$, $\varepsilon \sim p(\varepsilon) = \mathcal{N}(0, \sigma_2^2)$. Выборка X составляется из ℓ независимых пар (x_i, y_i).

Рассмотрим два простейших случая:

- $f(x) = ax$

- $f(x)$ — чётная функция, то есть $f(-x) = f(x)$.

Задача 1.1. *Найдите шумовую компоненту для одномерной линейной регрессии.*

Решение. Так как распределение $p(y|x) = \mathcal{N}(f(x), \sigma_2^2)$ нормальное, для него легко вычислить математическое ожидание $\mathbb{E}[y|x] = f(x)$, тогда

$$\mathbb{E}_{x,y}\big[(y - \mathbb{E}[y|x])^2\big] = \mathbb{E}_{x,\varepsilon}\big[(f(x) + \varepsilon - f(x))^2\big] = \mathbb{E}_\varepsilon \varepsilon^2 = \mathbb{D}\varepsilon + \big(\mathbb{E}\varepsilon\big)^2 = \sigma_2^2 + 0 = \sigma_2^2.$$

■

Задача 1.2. *Найдите смещение алгоритма одномерной линейной регрессии для $f(x) = ax$ и для произвольной чётной $f(x)$.*

Решение. Для начала найдем «средний» по всем выборкам ответ алгоритма на объекте x:

$$\mathbb{E}_X[\mu(X)(x)] = \mathbb{E}_X[k(X)]\, x.$$

Теперь необходимо найти «среднее» по всем выборкам значение k:

$$\mathbb{E}_X[k(X)] = \int \frac{\sum_i x_i(f(x_i) + \varepsilon_i)}{\sum_i x_i^2} \prod_i \big(p(x_i)p(\varepsilon_i)\big)\, dx_1 \dots dx_\ell\, d\varepsilon_1 \dots d\varepsilon_\ell$$

Итак, мы имеем дело с несобственным интегралом, в котором каждая переменная принимает значения от $-\infty$ до $+\infty$, его значение определяется функцией $f(x)$ и не всегда может быть получено аналитически. В том случае, когда истинная зависимость в данных линейная, получим:

$$\mathbb{E}_X[k(X)] = \int \frac{\sum_i x_i(a\, x_i + \varepsilon_i)}{\sum_i x_i^2} p(\bar{x})p(\bar{\varepsilon})d\bar{x}d\bar{\varepsilon} =$$

$$= a \int \frac{\sum_i x_i^2}{\sum_i x_i^2} p(\bar{x})p(\bar{\varepsilon})d\bar{x}d\bar{\varepsilon} + \int \frac{\sum_i x_i\varepsilon_i}{\sum_i x_i^2} p(\bar{x})p(\bar{\varepsilon})d\bar{x}d\bar{\varepsilon}.$$

Интеграл, входящий в состав первого слагаемого, равен a (интеграл по всему пространству от плотности распределения). Второй интеграл берётся по симметричным относительно нуля интервалам от нечётной по x_i и по ε_i функции, а потому равен 0. Значит, «средний» коэффициент k равен a.

Найдем смещение:

$$\mathbb{E}_x\big[(\mathbb{E}_X[\mu(X)(x)] - \mathbb{E}[y|x])^2\big] = \mathbb{E}_x\big[(a\, x - a\, x)^2\big] = 0.$$

Это согласуется со здравым смыслом: перебрав все возможные выборки размера ℓ и усреднив по ним значение k, мы обязательно найдём истинное значение коэффициента.

Теперь по аналогии найдём «среднее» k для произвольной чётной $f(x)$:

$$\mathbb{E}_X[k(X)] = \int \frac{\sum_i (x_i f(x_i))}{\sum_i x_i^2} p(\bar{x})d\bar{x} + \int \frac{\sum_i x_i\varepsilon_i}{\sum_i x_i^2} p(\bar{x})p(\bar{\varepsilon})d\bar{x}d\bar{\varepsilon}.$$

Мы уже знаем, что интеграл, входящий в состав второго слагаемого, равен 0. В данном случае первый интеграл тоже равен 0, так как подынтегральное выражение является нечётной по всем x_i функцией. Значит, «среднее» значение коэффициента тоже равно нулю.

Найдём смещение:

$$\mathbb{E}_x\big[(\mathbb{E}_X[\mu(X)(x)] - \mathbb{E}[y|x])^2\big] = \mathbb{E}_x\big[(0 - f(x))^2\big] = \mathbb{E}_x f^2(x).$$

■

Задача 1.3. *Найдите разброс алгоритма одномерной линейной регрессии для $f(x) = ax$ и для произвольной чётной $f(x)$.*

Решение. Сначала рассмотрим случай $f(x) = ax$:

$$\mathbb{E}_x\big[\mathbb{E}_X\big[(\mu(X)(x) - \mathbb{E}_X[\mu(X)(x)])^2\big]\big] = \mathbb{E}_x\big[\mathbb{E}_X\big[(ax + \frac{\sum_i x_i\varepsilon_i}{\sum_i x_i^2}x - ax)^2\big]\big] =$$

$$= \Big(\mathbb{E}_x x^2\Big)\Big(\mathbb{E}_X\Big(\frac{\sum_i x_i\varepsilon_i}{\sum_i x_i^2}\Big)^2\Big) = \sigma_1^2\mathbb{E}_X\Big(\frac{\sum_i x_i\varepsilon_i}{\sum_i x_i^2}\Big)^2.$$

Математическое ожидание можно немного упростить, раскрыв квадрат суммы в числителе и внеся внутрь суммы математическое ожидание по $\bar{\varepsilon} = (\varepsilon_1, \ldots, \varepsilon_\ell)$:

$$\mathbb{E}_X\Big(\frac{\sum_i x_i\varepsilon_i}{\sum_i x_i^2}\Big)^2 = \mathbb{E}_{\bar{x}}\Big[\frac{\sum_{i\neq j} x_i x_j \mathbb{E}_{\bar{\varepsilon}}[\varepsilon_i\varepsilon_j] + \sum_i x_i^2\mathbb{E}_{\bar{\varepsilon}}\varepsilon_i^2}{(\sum_i x_i^2)^2}\Big].$$

Так как ε_i и ε_j независимы, то $\mathbb{E}[\varepsilon_i\varepsilon_j] = 0$, а $\mathbb{E}_{\bar{\varepsilon}}\varepsilon_i^2 = \sigma_2^2$. Тогда

$$\mathbb{E}_X\Big(\frac{\sum_i x_i\varepsilon_i}{\sum_i x_i^2}\Big)^2 = \mathbb{E}_{\bar{x}}\Big[\frac{\sum_i x_i^2\sigma_2^2}{(\sum_i x_i^2)^2}\Big] = \sigma_2^2\mathbb{E}_{\bar{x}}\Big[\frac{1}{\sum_i x_i^2}\Big],$$

а разброс

$$\mathbb{E}_x\big[\mathbb{E}_X\big[(\mu(X)(x) - \mathbb{E}_X[\mu(X)(x)])^2\big]\big] = \sigma_1^2\sigma_2^2\mathbb{E}_{\bar{x}}\Big[\frac{1}{\sum_i x_i^2}\Big].$$

Последнее математическое ожидание можно рассчитать, но мы этого делать не будем. Итак, если шум в ответах небольшой, то и разброс модели будет небольшой.

Теперь рассмотрим чётную $f(x)$:

$$\mathbb{E}_x\big[\mathbb{E}_X\big[(\mu(X)(x) - \mathbb{E}_X[\mu(X)(x)])^2\big]\big] = \mathbb{E}_x\big[\mathbb{E}_X\big[(0 - \frac{\sum_i x_i(f(x_i) + \varepsilon_i)}{\sum_i x_i^2}x)^2\big]\big] =$$

$$= \Big(\mathbb{E}_x x^2\Big)\Big(\mathbb{E}_X\Big(\frac{\sum_i x_i(f(x_i) + \varepsilon_i)}{\sum_i x_i^2}\Big)^2\Big) = \sigma_1^2\mathbb{E}_X\Big[\frac{\sum_i x_i(f(x_i) + \varepsilon_i)}{\sum_i x_i^2}\Big]^2.$$

■

§1.3 Приближенное вычисление интегралов

Bias-variance decomposition — это теоретическая конструкция, которая позволяет понять, из-за чего алгоритмы переобучаются и недообучаются. Чаще всего при анализе алгоритма термины «смещение» и «разброс» используют для оценки их величины, но точные значения для большинства алгоритмов вычислить аналитически невозможно.

Если необходимость это сделать всё же появляется, можно использовать техники приближенного вычисления интегралов с помощью семплирования. Так, например, если нам нужно оценить математическое ожидание $\mathbb{E}_{x\sim p(x)} f(x) = \int f(x)p(x)dx$, то можно просемплировать выборку $\{x_1, \ldots, x_n\}$ из распределения $p(x)$ и приближенно вычислить интеграл:

$$\mathbb{E}_{x\sim p(x)} f(x) \approx \frac{1}{n}\sum_{i=1}^{n} f(x_i).$$

Из областей с большим значением плотности в выборку попадёт больше точек, и они внесут больший вклад в значение интеграла. Несложно показать, что данная оценка является несмещённой.

Для вычисления математического ожидания по случайным выборкам при этом нужно сгенерировать несколько выборок:

$$\mathbb{E}_X[\mu(X)(x)] \approx \frac{1}{n}\sum_{i=1}^{n} \mu(X_i)(x), \quad X_i = \{(x_{i,1}, y_{i,1}), \ldots, (x_{i,\ell}, y_{i,\ell})\},\ x_{i,j}, y_{i,j} \sim p(x, y)$$

§1.4 Bias-variance trade-off

В литературе часто можно встретить график, на котором по горизонтали отложена условная «сложность» обученной модели, а по вертикали — ошибка на тестовой выборке. Для большинства методов этот график представляет собой U-образную кривую, как на рис. 4: недостаточно сложная модель недообучена, то есть обладает высоким смещением, а слишком сложная — напротив, переобучена, то есть обладает высоким разбросом. При этом между этими двумя состояниями существует оптимальная сложность, позволяющая минимизировать ошибку на тестовой выборке.

Такое поведение характерно для многих методов, однако оно не является аксиомой: например, в работах по глубоким нейронным сетям авторы отмечают, что по мере увеличения числа параметров ошибка на тестовой выборке может сначала падать, затем резко возрастать в окрестности точки, в которой модель может идеально запомнить обучающую выборку, а потом снова уменьшаться. Такое явление получило название double descent.

Также стоит понимать, что из разложения, которое остаётся корректным всегда, не следует однозначно, что при уменьшении смещения обязательно должен возрастать разброс (и наоборот). На практике возможны сценарии, когда обе компоненты одновременно уменьшаются при увеличении мощности модели.

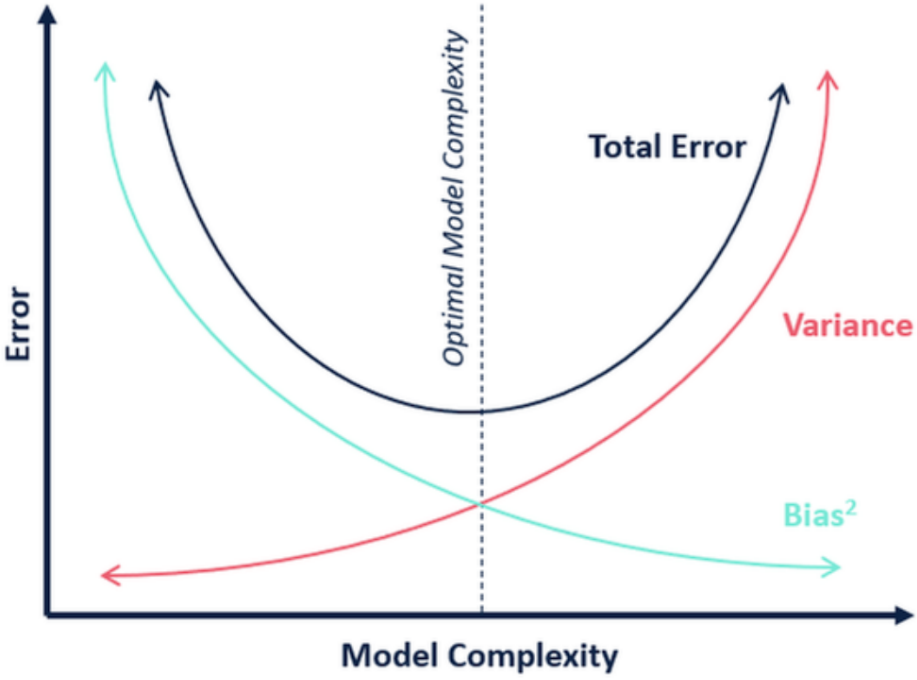

Рис. 4. Bias-variance trade-off

2 Композиционные алгоритмы

Существуют способы уменьшить смещение и разброс алгоритма. Одним из наиболее простых и популярных подходов является усреднение предсказаний нескольких моделей. Его идея состоит в том, что отдельная модель может допускать случайные ошибки из-за особенностей обучающей выборки, однако если обучить несколько моделей и объединить их предсказания, то случайные отклонения будут компенсироваться, а итоговый результат окажется более стабильным и точным.

§2.1 Простое голосование

Рассмотрим задачу классификации: для неё идею усреднения предсказаний напрямую применить невозможно, однако можно использовать принцип большинства. Для этого достаточно построить несколько классификаторов, предсказывающих класс для объекта x, и относить его к тому классу, который чаще всего предсказывали эти алгоритмы. Проверим, действительно ли такой подход даёт результат.

Задача 2.1. *Рассмотрим три бинарных классификатора, каждый из которых ошибается с вероятностью p. С какой вероятностью будет ошибаться классификатор, построенный с помощью простого голосования? При каких значениях p эта вероятность будет меньше p?*

Решение. Простое голосование выдаст правильный ответ, если не более чем один алгоритм ошибётся. Вероятность того, что все три алгоритма дадут корректный ответ, равна $(1-p)^3$, а вероятность того, что его дадут ровно два алгоритма из трёх —

$3p(1-p)^2$. Тогда итоговая вероятность ошибки:

$$1-(1-p)^3-3p(1-p)^2 = 1-1+3p-3p^2+p^3-3p+6p^2-3p^3 = -2p^3+3p^2 = p^2(3-2p)$$

$-2p^3 + 3p^2 < p$;

$p(-2p^2 + 3p - 1) < 0$;

$-2p(p-1)(p-\frac{1}{2}) < 0$.

На отрезке $p \in [0,1]$ решением этого неравенства является полуинтервал $p \in (0, \frac{1}{2})$. Таким образом, если каждый из алгоритмов хотя бы немного лучше, чем случайный, композиция даст лучший результат, чем каждый алгоритм по отдельности.

Очевидно, что при большем нечётном числе классификаторов выигрыш от голосования будет только возрастать, если $p < 0.5$. Если же $p > 0.5$, композиция результат будет делать только хуже.

■

§2.2 Бэггинг

Самый известный метод, использующий описанный выше подход, — бэггинг, который заключается в генерации нескольких новых выборок одинакового размера $X_1, \ldots, X_m$ на основе имеющейся, обучении алгоритма на каждой из сгенерированных выборок и усреднении ответов всех алгоритмов на новом объекте. Такой подход на практике позволяет уменьшить разброс рассматриваемого алгоритма.

Задача 2.2. *При бэггинге новую выборку $\widetilde{X}$ составляют, генерируя элементы из X с возвращением. При этом объекты в $\widetilde{X}$ могут повторяться. Будем считать, что число объектов в $\widetilde{X}$ и в X одинаковое и равно ℓ. Найдите вероятность того, что конкретный объект попадёт в выборку.*

Решение. Вероятность того, что объект попадёт в выборку при одном вытаскивании, равна $\frac{1}{\ell}$, где ℓ — число объектов выборки. Вероятность того, что он не попадёт в выборку — $1 - \frac{1}{\ell}$, а вероятность того, что не попадёт ни при одном вытаскивании, соответственно, $\left(1 - \frac{1}{\ell}\right)^\ell$. Наконец, искомая вероятность равна

$$1 - \left(1 - \frac{1}{\ell}\right)^\ell \to 1 - \frac{1}{e} \text{ при } \ell \to \infty.$$

■

Таким образом, примерно 63% объектов попадает в выборку, а 37% остаётся вне. Такая случайность обеспечивает достаточное разнообразие обучающих выборок, что в итоге и делает усреднение предсказаний эффективным.

На лекции было показано, что усреднение с помощью бэггинга уменьшает разброс алгоритма примерно в m раз, если базовые алгоритмы мало коррелированы. Такой эффект, к слову, наблюдается при усреднении предсказаний любых алгоритмов регрессии, необязательно полученных бэггингом, если эти алгоритмы выдают слабо коррелированные ответы.

Вспомним задачу, которую мы решали в начале семинара, и вновь рассмотрим случай, когда истинная зависимость имеет вид $f(x) = 3\sin x$, а предсказания модели зашумлены с теми же параметрами, что и ранее. Обратим внимание на то, как меняется картинка при использовании бэггинга над решающими деревьями (рис. 5).

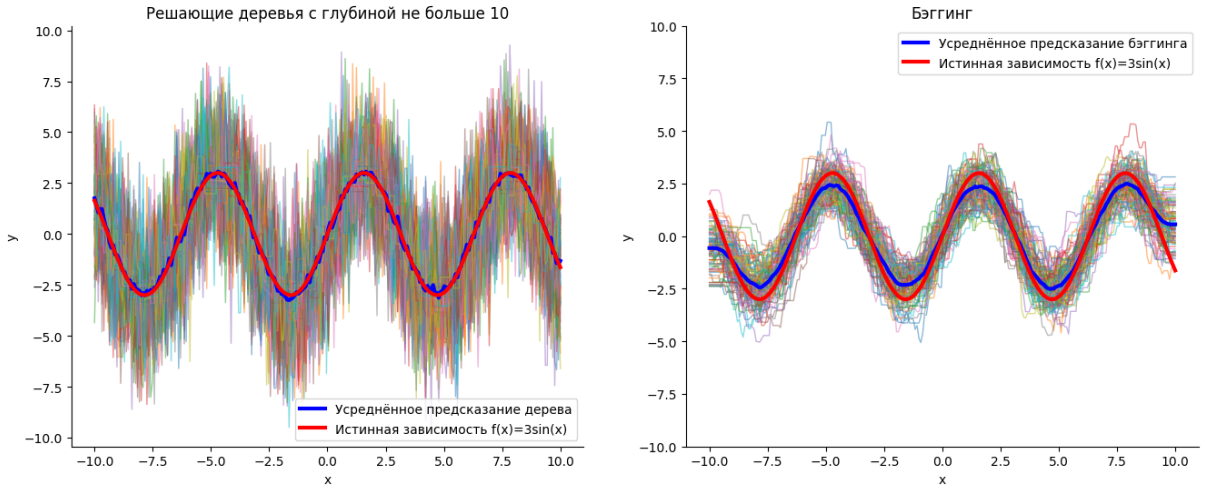

Рис. 5. Сравнение классических решающих деревьев с бэггингом

Как и ожидалось, использование бэггинга позволило значительно уменьшить разброс, при этом значение смещения осталось невысоким. Для подтверждения этого наблюдения также построим график, на котором отобразим значения компонент разложения для обоих методов (рис. 6).

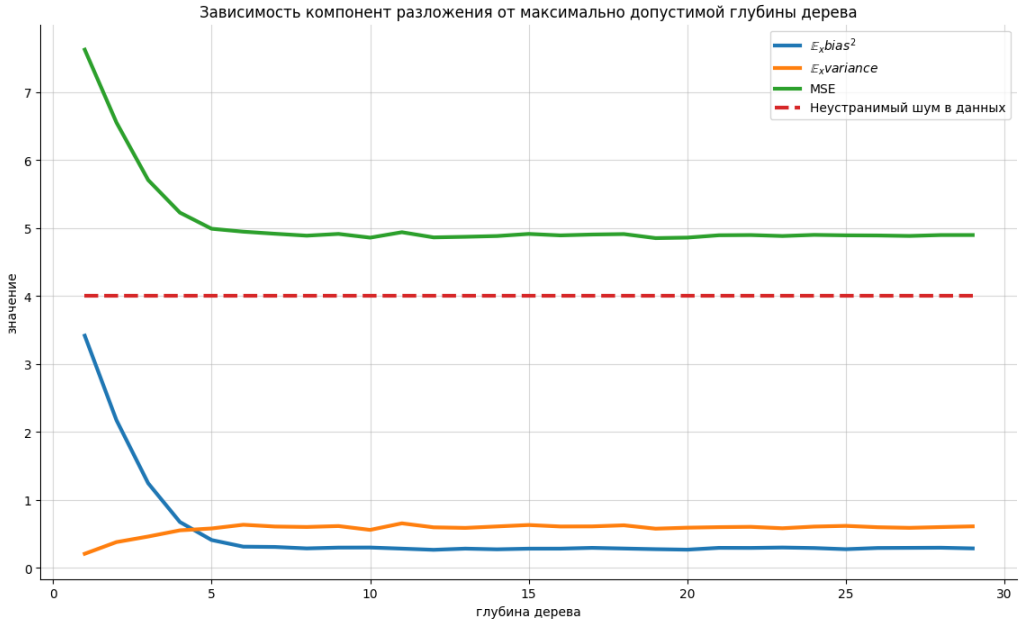

Рис. 6. Разложение на компоненты для бэггинга

Итак, с помощью бэггинга значение разброса удалось уменьшить практически в 10 раз.

§2.3 Случайные леса

В реальной жизни добиться отсутствия корреляции между алгоритмами практически невозможно, однако это и не нужно: на практике достаточно, чтобы они были лишь в некоторой степени не похожи друг на друга. На этой идее основан другой алгоритм — случайный лес. Главное его отличие от обычного бэггинга заключается в том, что он вносит дополнительную случайность в процесс построения деревьев за счёт двух приёмов:

- каждое дерево обучается на отдельной подвыборке объектов (аналогично бэггингу)
- в каждой вершине при поиске оптимального разбиения рассматривается лишь случайное подмножество признаков

Благодаря этому отдельные деревья становятся менее похожими друг на друга, а усреднение их предсказаний оказывается более эффективным. Итоговая модель благодаря этому даёт более высокое качество даже при использовании простейших базовых моделей (решающих деревьев без сильных ограничений на максимально допустимую глубину).

Интересно, что случайные леса являются одним из самых надёжных и универсальных методов машинного обучения и успешно применяются для решения задач классификации и регрессии, а также зачастую выступают в роли бейзлайна.

§2.4 Бустинг

Итак, мы разобрали метод, позволяющий уменьшить разброс модели, однако зачастую требуется понизить смещение, а с этой задачей бэггинг не справляется. Для её решения рассмотрим другой подход — бустинг, о котором подробнее будет рассказано на следующих лекциях.

Главная идея этого метода заключается в последовательном (а не параллельном, в отличие от бэггинга) обучении нескольких базовых моделей. Таким образом, каждая следующая модель старается исправить ошибки, которые допустили предыдущие. Итоговое предсказание — это взвешенная комбинация предсказаний всех обученных моделей.

Поскольку каждый следующий базовый алгоритм в бустинге обучается так, чтобы уменьшить общую ошибку всех своих предшественников, итоговая композиция будет иметь меньшее смещение, что и требовалось. В качестве базовых обычно выбирают алгоритмы с высоким смещением и небольшим разбросом, например, деревья небольшой глубины.