



Тверской
государственный
технический
университет

Интеллектуальные информационные системы

Ансамблевые методы

Материалы курса доступны по ссылке:

<https://github.com/AndreyShpigar/ML-course>

2024 г.

Идея ансамблирования – как из множества по отдельности плохих алгоритмов построить один хороший?

Models in Ensemble algorithms:



Apes together strong.

Bias-Variance decomposition

Рассмотрим на примере: *функционал оценки качества работы алгоритма а при использовании квадратичной функции потерь:*

$$Q(a) = \mathbb{E}_x \mathbb{E}_{X, \epsilon} [y(x, \epsilon) - a(x, X)]^2$$

X – обучающая выборка

x – объект из тестовой выборки

$y = f(x) + \epsilon$ – целевая зависимость, измеренная с точностью до случайного шума ϵ

$a(x, X)$ – значение алгоритма на объекте x

\mathbb{E}_x - математическое ожидание по всем объектам тестовой выборки

$\mathbb{E}_{X, \epsilon}$ - математическое ожидание по всем обучающим выборка X и случайному шуму ϵ

Представим этот же функционал в виде трех составляющих: шум, смещение (bias) и разброс (variance)

$$Q(a) = \mathbb{E}_x \text{bias}_X^2 a(x, X) + \mathbb{E}_X \mathbb{V}_X[a(x, X)] + \sigma^2$$

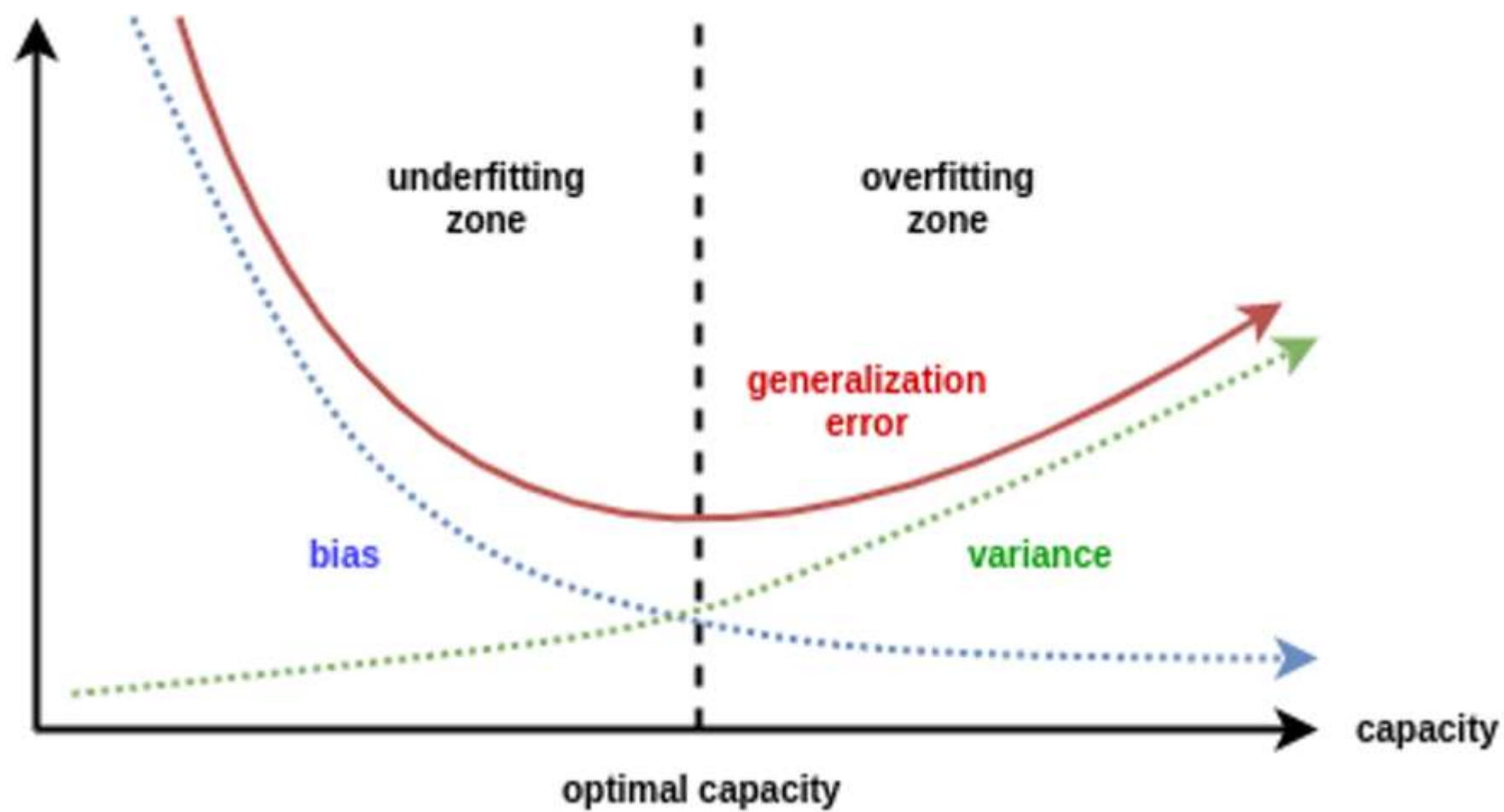
$$\text{bias}_X a(x, X) = f(x) - \mathbb{E}_X[a(x, X)] - \text{смещение}$$

предсказания алгоритма на объекте x , усредненного по всем возможным обучающим выборкам, относительно истинной зависимости f

$$\mathbb{V}_X[a(x, X)] = \mathbb{E}_X[a(x, X) - \mathbb{E}_X[a(x, X)]]^2 - \text{разброс}$$

(дисперсия) предсказаний алгоритма в зависимости от обучающей выборки X

σ^2 - шум в данных

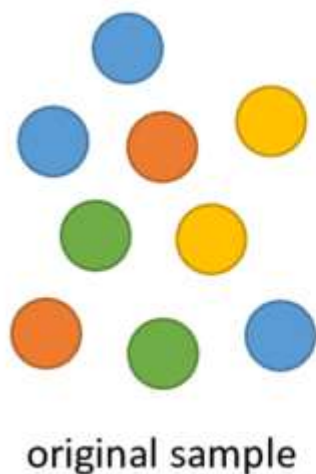


Гипотеза: возможно ли уменьшить одну из компонент ошибки, не увеличивая другую?

Bootstrap

$X = (x_i, y_i)$ – конечная выборка размером ℓ

Сгенерируем N подвыборок X_1, \dots, X_N размером ℓ с помощью бутстрапа – выбираем ℓ объектов равновероятно с возвращением.



На каждой из подвыборок обучим модель, например, линейной регрессии, получив базовые алгоритмы $b_1(x), \dots, b_N(x)$.

Существуют истинная функция ответа для всех объектов $y(x)$ и задано распределение на объектах $p(x)$. Тогда ошибка каждой функции регрессии:

$$\varepsilon_j(x) = b_j(x) - y(x), \quad j = 1, \dots, N$$

Математическое ожидание среднеквадратичной ошибки:

$$\mathbb{E}_X(b_j(x) - y(x))^2 = \mathbb{E}_x \varepsilon_j^2(x)$$

Тогда средняя ошибка построенных функций регрессии:

$$E_1 = \frac{1}{N} \sum_{j=1}^N \mathbb{E}_x \varepsilon_j^2(x)$$

Предположим, что ошибки несмещены и некоррелированы:

$$\mathbb{E}_x \varepsilon_j(x) = 0 \text{ и } \mathbb{E}_x \varepsilon_i(x) \varepsilon_j(x) = 0, i \neq j$$

Тогда новая функция регрессии, которая будет усреднять ответы построенных моделей:

$$a(x) = \frac{1}{N} \sum_{j=1}^N b_j(x)$$

Выведем среднеквадратичную ошибку этой функции:

$$\begin{aligned} E_N &= \mathbb{E}_x \left(\frac{1}{N} \sum_{j=1}^N b_j(x) - y(x) \right)^2 = \mathbb{E}_x \left(\frac{1}{N} \sum_{j=1}^N \varepsilon_j(x) \right)^2 \\ &= \frac{1}{N^2} \mathbb{E}_x \left(\sum_{j=1}^N \varepsilon_j^2(x) + \sum_{i \neq j} \varepsilon_i(x) \varepsilon_j(x) \right) = \boxed{\frac{1}{N} E_1} \end{aligned}$$

↑
=0

Ошибка уменьшилась
в N раз !

Предположим, что ошибки ~~несмещены и некоррелированы~~:

$$\mathbb{E}_x \varepsilon_j(x) = 0 \text{ и } \mathbb{E}_x \varepsilon_i(x) \varepsilon_j(x) = 0, i \neq j$$

Тогда новая функция регрессии, которая будет усреднять ответы построенных моделей:

$$a(x) = \frac{1}{N} \sum_{j=1}^N b_j(x)$$

Выведем среднеквадратичную ошибку этой функции:

$$\begin{aligned} E_N &= \mathbb{E}_x \left(\frac{1}{N} \sum_{j=1}^N b_j(x) - y(x) \right)^2 = \mathbb{E}_x \left(\frac{1}{N} \sum_{j=1}^N \varepsilon_j(x) \right)^2 \\ &= \frac{1}{N^2} \mathbb{E}_x \left(\sum_{j=1}^N \varepsilon_j^2(x) + \sum_{i \neq j} \varepsilon_i(x) \varepsilon_j(x) \right) = \boxed{\frac{1}{N} E_1} \end{aligned}$$

↑
=0

Ошибка уменьшилась
в ~~N~~ раз !

Bagging = bootstrap aggregating

Bagging – простое (не взвешенное) голосование

Обучаем некоторое число базовых алгоритмов $b_n(x)$ с помощью бутстрапирования выборки и строим итоговую композицию как среднее всех базовых алгоритмов (моделей):

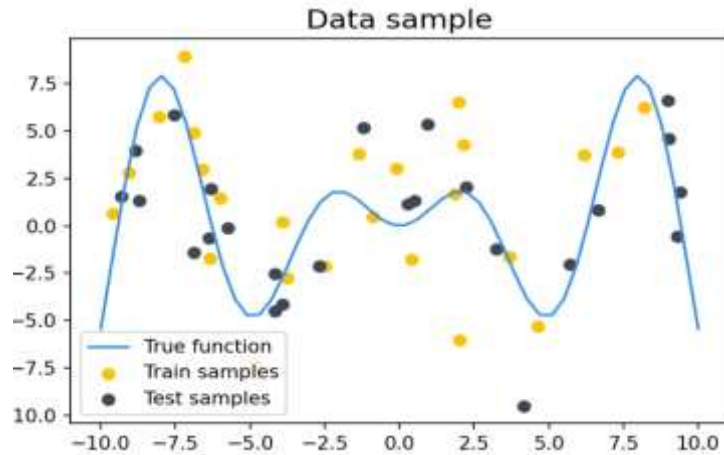
$$a_N(x) = \frac{1}{N} \sum_{n=1}^N b_n(x)$$

Оценим смещение и разброс ансамбля базовых алгоритмов (матожидание оцениваем по всем возможным подвыборкам, получаемым с помощью бутстрапа):

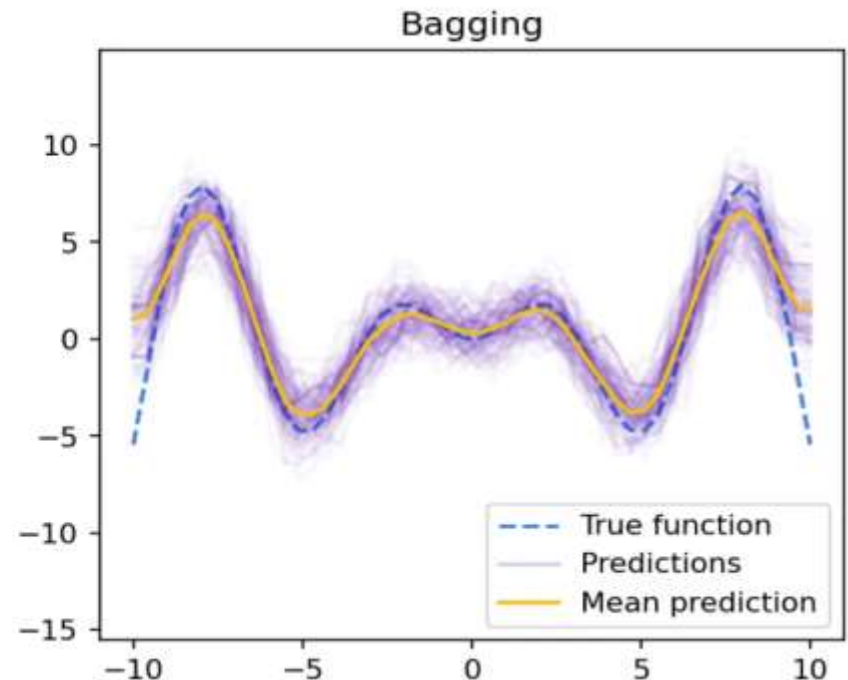
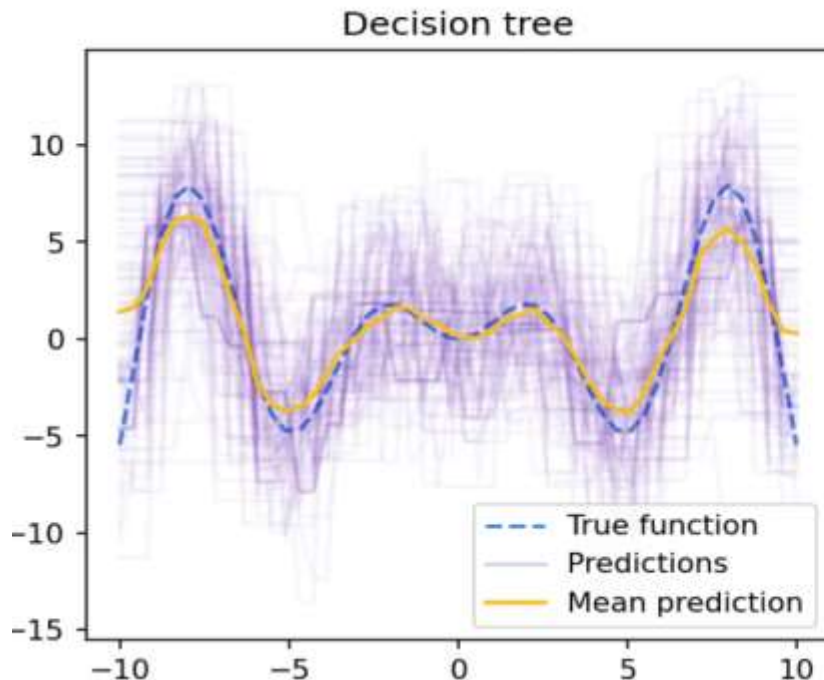
$bias_X a(x, X) = bias_X b(x, X)$ - смещение не изменилось по сравнению со средним смещением отдельных моделей (смещение ансамбля равно смещению одного алгоритма)

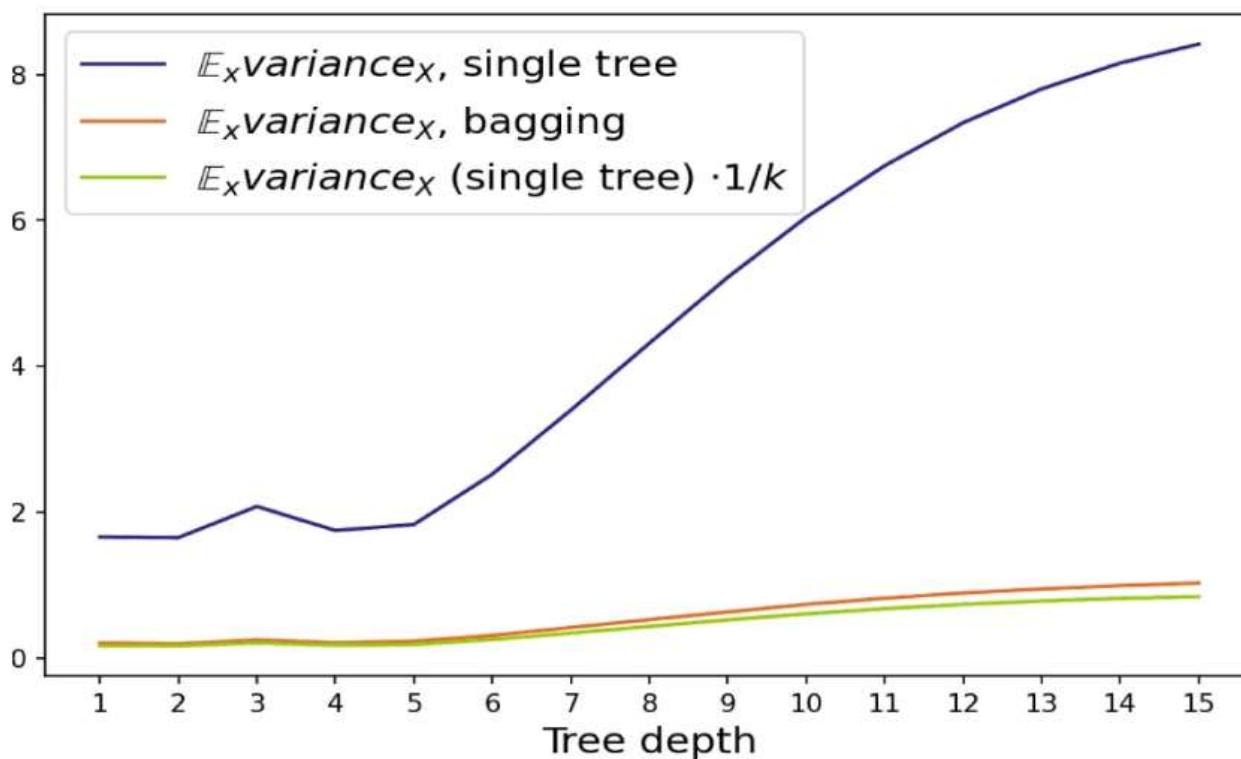
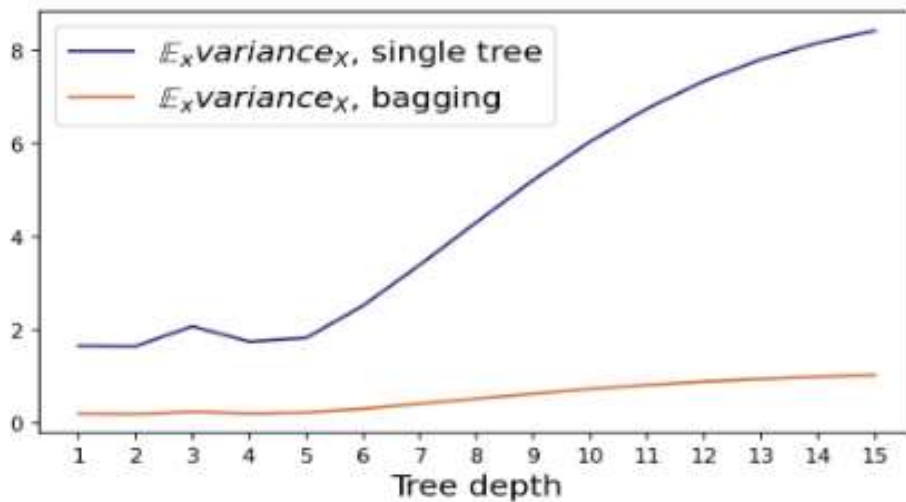
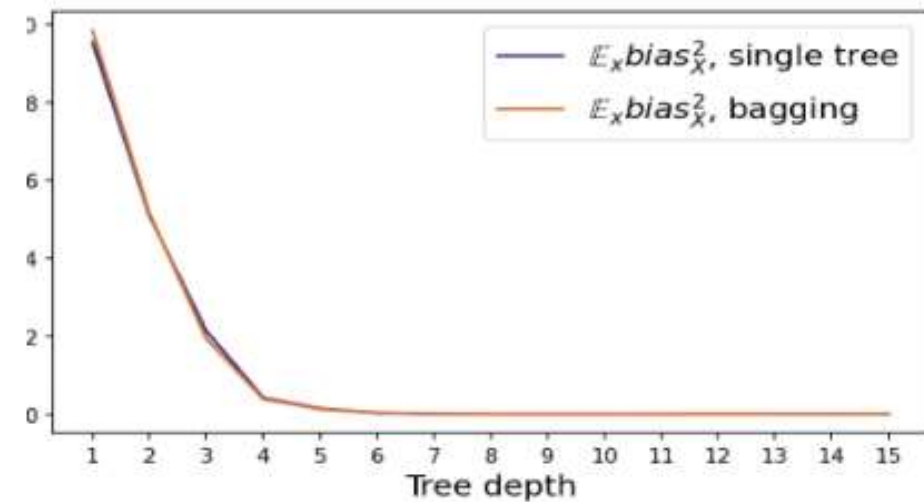
$\mathbb{V}_X[a(x, X)] = \frac{1}{k} \mathbb{V}_X b(x, X)$ – если базовые алгоритмы некоррелированы, то дисперсия ансамбля уменьшается в k раз

Бэггинг над решающими деревьями



Общая дисперсия предсказаний в зависимости от обучающего множества у бэггинга значительно ниже, чем у отдельных деревьев, а в среднем их предсказания не отличаются

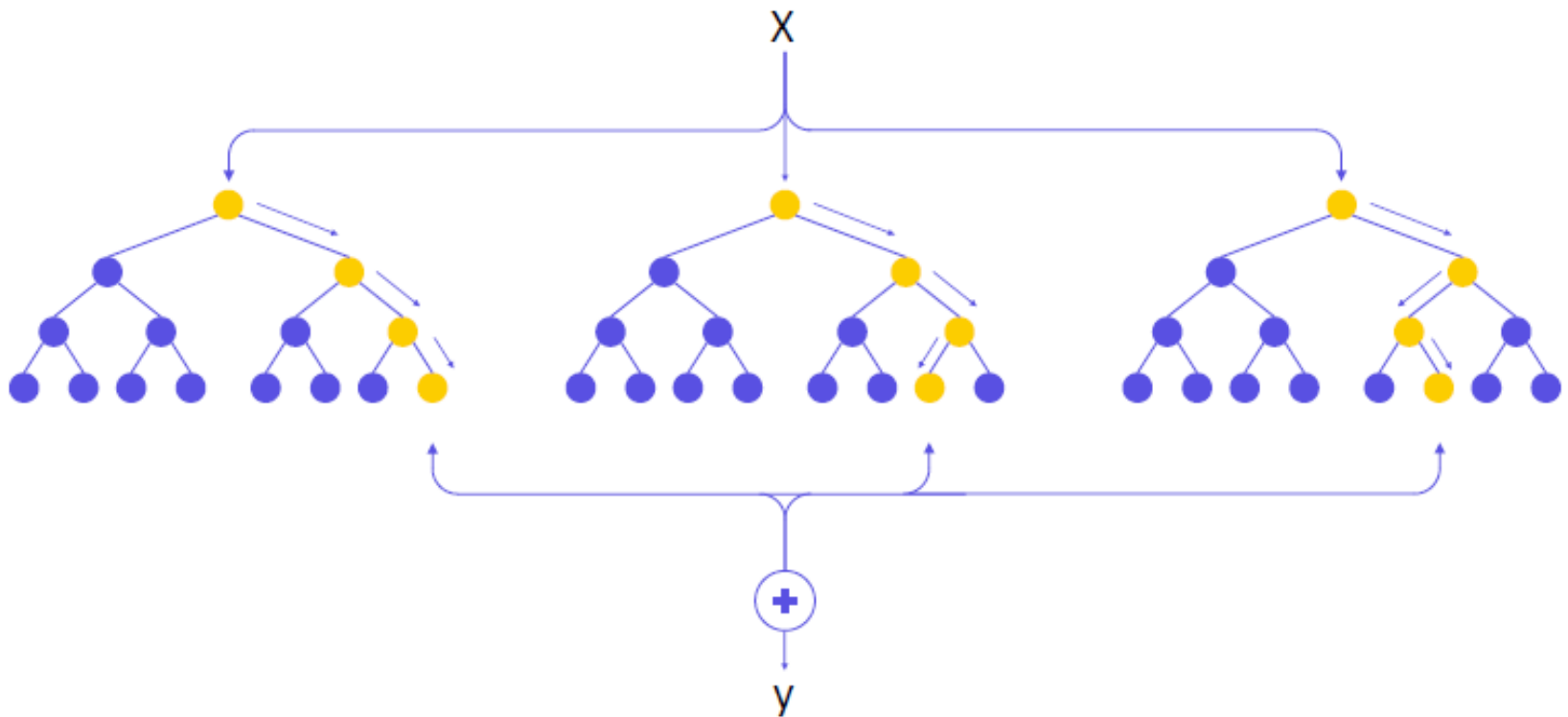




Random forest

Метод случайных лесов – основан на бэггинге над решающими деревьями и методе случайных подпространств (random subspace method)

Bagging + RSM = Random Forest



1. Для построения i -го дерева:

- из обучающей выборки X выбирается с возвращением случайная подвыборка X^i того же размера, что и X (бэггинг)

- в процессе обучения каждого дерева в каждой вершине случайно выбираются $n < N$ признаков, где N – полное число признаков (метод случайных подпространств), и среди них ищется оптимальный сплит

2. Получение предсказания ансамбля на тестовом объекте: для регрессии – усредняем отдельные ответы деревьев, для классификации – выбираем самый популярный класс

Out-of-Bag

Объекты, которые не вошли в бустрапированную выборку X_n дерева b_n являются контрольными для данного дерева. Для каждого объекта x_i можно выбрать деревья, которые были обучены без него и вычислить по их ответам несмещенную out-of-bag-ошибку:

$$OOB = \sum_{i=1}^l L \left(y_i, \frac{1}{\sum_{n=1}^N [x_i \notin X_n]} \sum_{n=1}^N [x_i \notin X_n] b_n(x_i) \right)$$

По мере увеличения числа деревьев N данная оценка стремится к leave-one-out оценке, но существенно проще для вычисления. По OOB оценке можно настраивать гиперпараметры Random Forest.

Особенности построения Random Forest

- Строим глубокие деревья, потому что у них низкое смещение, а разброс уменьшается за счет бэггинга.
- Чем больше признаков, тем больше корреляция между деревьями. По умолчанию – для регрессии $[n/3]$ признаков, для классификации \sqrt{n}
- Количество деревьев в случайном лесе:
 - выбираем исходя из зависимости уменьшения ошибки ансамбля от количества деревьев (строим график и смотрим когда ошибка перестает значимо уменьшаться)
 - ограничиваем исходя из требований к времени работы модели
- Минимальное число объектов в расщепляемой подвыборке
- Минимальное число объектов в листьях
- Критерий расщепления (информативности)

Обобщение ансамблевых методов

Способы повышения разнообразия базовых алгоритмов:

- Обучение по случайным подвыборкам
- Обучение по случайным наборам признаков
- Обучение из разных параметрических моделей
- Обучение с использованием рандомизации:
 1. *Bagging* – в каждую выборку попадает $1 - \left(1 - \frac{1}{l}\right)^l \rightarrow 1 - \frac{1}{e} \approx 63.2\%$ объектов при $l \rightarrow \infty$
 2. *Pasting* – случайные обучающие подвыборки
 3. *Random subspaces* – случайные подмножества (без возвращения)
 4. *Random patches* – случайные подмножества как объектов так и признаков
 5. *Cross-validated committees* – выборка разбивается на k фолдов и делается k обучений без одного фолда

Преимущества и недостатки стохастических методов ансамблирования

- ✓ Ансамблирование это *метод обертка* над базовым методом обучения (базовые алгоритмы обучаются готовыми методами)
- ✓ Универсальность – подходит для классификации, регрессии, поиска аномалий и других задач
- ✓ Простая реализация, легкое распараллеливание, т.к. все базовые алгоритмы строятся независимо
- ✓ Возможность получения несмещенных оценок OOB
- ✓ Random Forest – один из лучших универсальных методов, часто используется как baseline-решение с хорошим качеством, которое легко реализовать «из коробки»

- × Требуется большое количество базовых алгоритмов
- × Базовые алгоритмы должны быть разнообразными – необходимо найти компромисс качество/различность
- × Random forest плохо работает с разреженными признаками