Занятие 9. Ансамбли моделей. Случайный лес

Колмагоров Евгений
ml.hse.dpo@yandex.ru

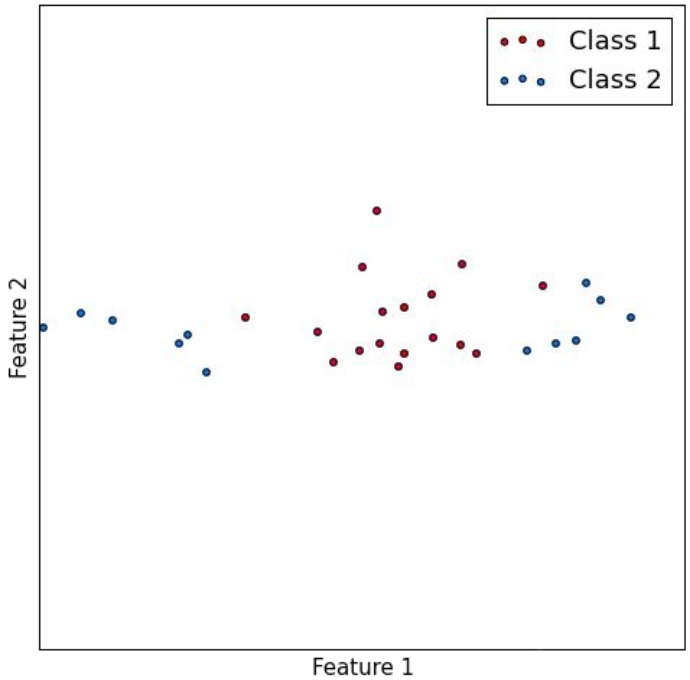
4 декабря 2024

План лекции

1. Ещё раз про недообучение/переобучение
2. Bias-Variance tradeoff
3. Ансамбли моделей
4. Случайный лес
5. Стекинг

Мотивация

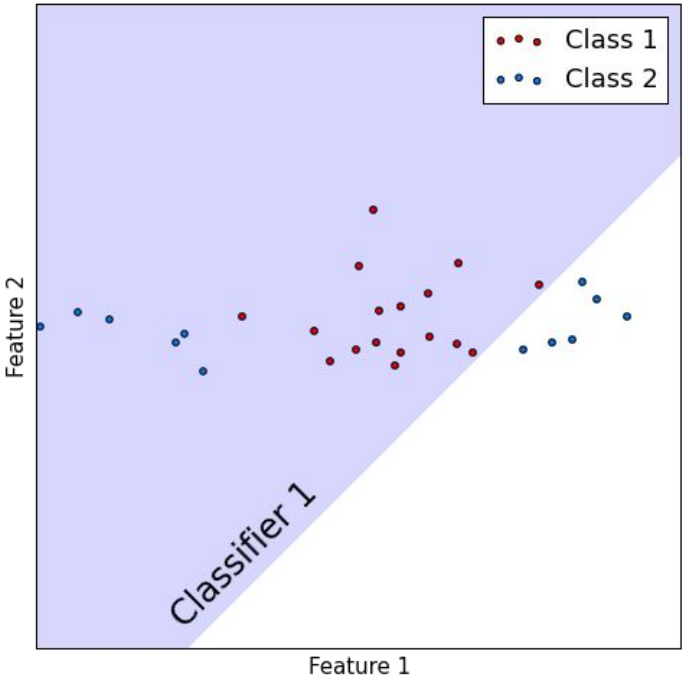
При построении алгоритмов машинного обучения, возникает проблема с тем, что природа данных зачастую оказывается сложнее, чем способность модели находить зависимости в данных

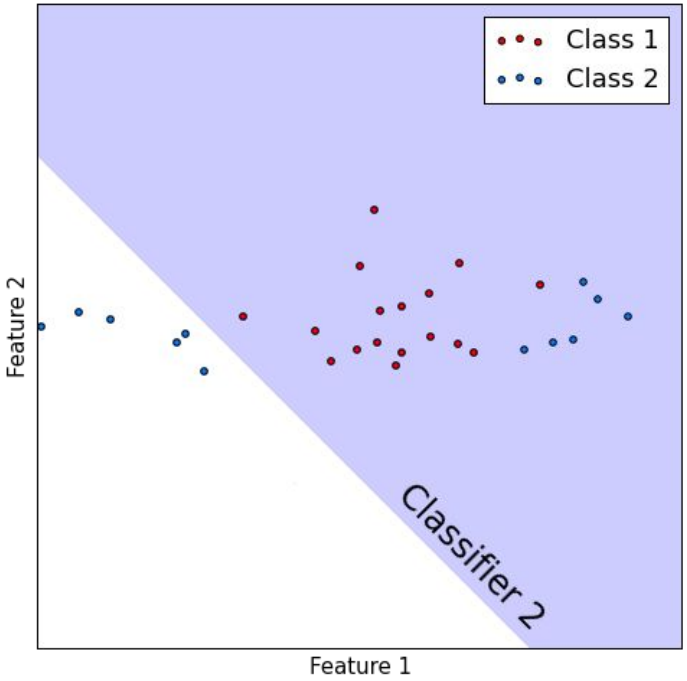


Попробуем решить задачу классификации объектов в представленной выборке линейной моделью

Пример недобучение

В результате обучения будут получаться следующие модели

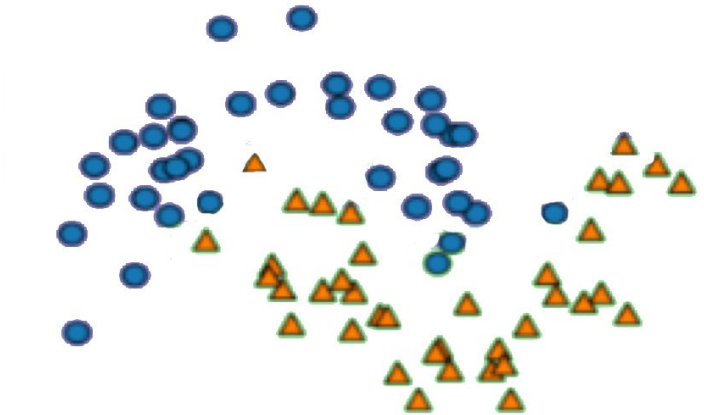




Пример переобучение

В предыдущем примере за счёт композиции алгоритмов удалось построить из простых подходов **более сложную** модель.

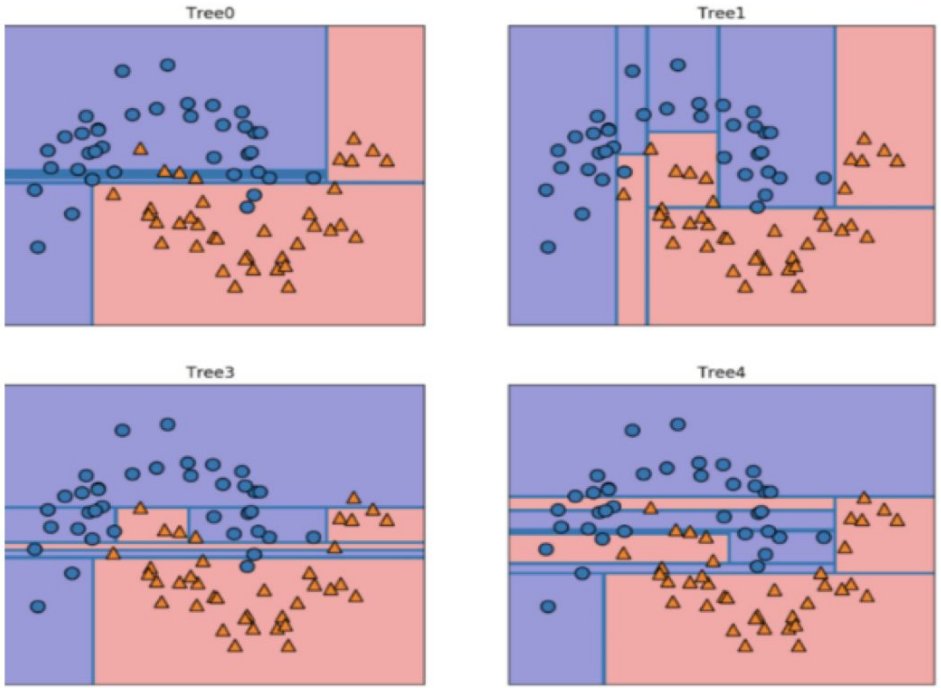
Но также с помощью композиции можно наоборот проводить **упрощение** моделей



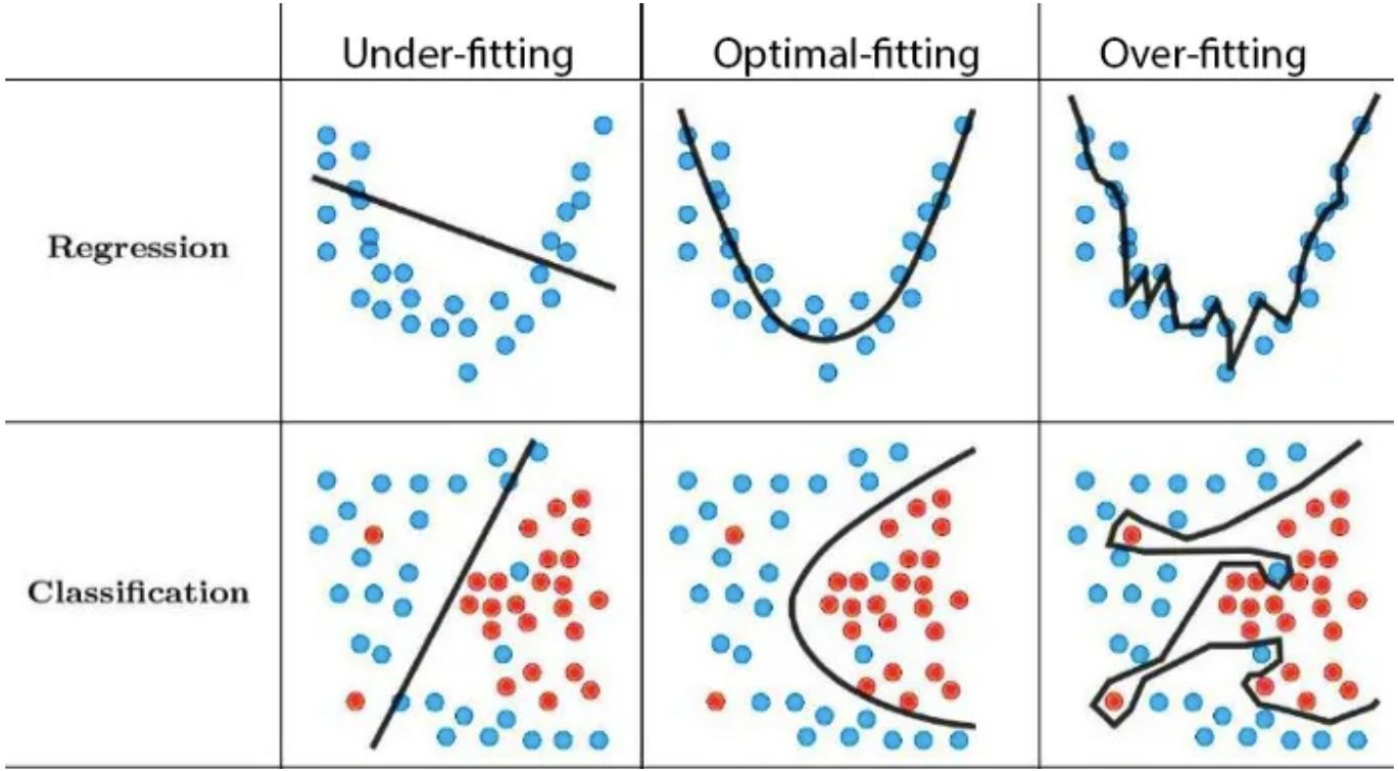
Попробуем решить задачу классификации объектов в представленной выборке деревом решений

Пример переобучение

В результате обучения получим следующие деревья решений



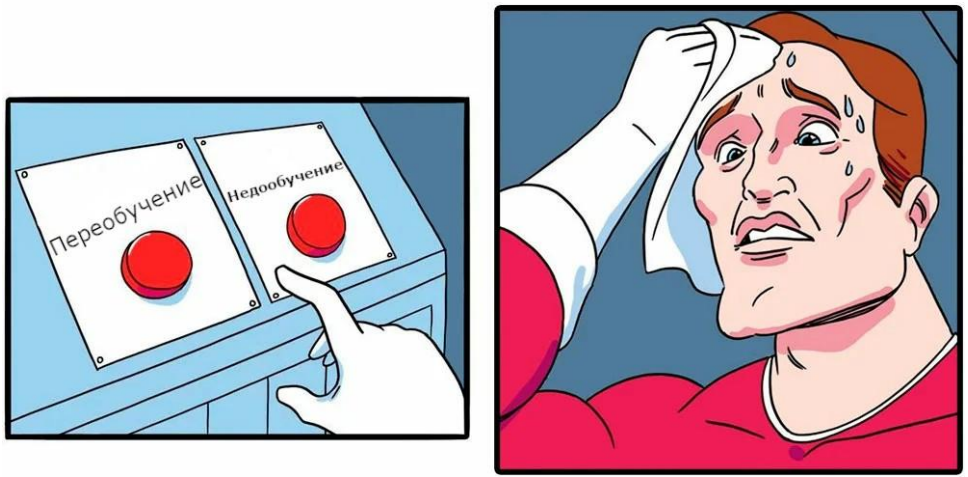
Напоминание переобучение и недообучение



Недообучение vs переобучение

Каждый из алгоритмов имеет тендецию либо к переобучению либо к недообучению.

Как правило слишком **простые** модели с малым числом признаков **недообучены**, а слишком **сложные** модели **переобучены**



Декомпозиция ошибки (Bias-Variance tradeoff)

Ошибку любого семейства алгоритма можно представить в виде специальной декомпозиции.

Пусть целевая переменная y выражается через x как:

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

Где f – некоторая детерминированная функция, а ε – случайный шум, с нулевым мат. ожиданием и некоторой дисперсией:

$$\mathbb{E}\varepsilon = 0$$

$$Var\ \varepsilon = \mathbb{E}\varepsilon^2 = \sigma^2$$

Декомпозиция ошибки (Разложение Bias-Variance)

Тогда для любого алгоритма машинного обучения a(x, X) справедливо следующее равенство:

$$\mathbb{E}_{X,\varepsilon}[f(x) + \varepsilon - a(x, X)] =$$
$$= (f(x) - \mathbb{E}_X[a(x, X)])^2 +$$
$$+\mathbb{E}_X[(a(x, X) - \mathbb{E}_X[a(x, X)])^2] + \sigma^2$$

Полный математический вывод можно посмотреть в этом источнике

Смещение (bias)

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

Данная величина носит название “**смещение**” и характеризует среднюю ошибку алгоритма по всем возможным наборам обучающим выборкам

- Показывает насколько хорошо с помощью данного алгоритма *a(x)* можно приблизить целевую зависимость *f(x)*
- Маленькое смещение – хорошее предсказание целевой переменной в **среднем**
- Большое смещение – предсказания далеки от истинной переменной

Разброс (Variance)

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

Данная величина носит название “**разброс**” и характеризует чувствительность алгоритма к изменениям в обучающей выборке

- Показывает дисперсию предсказаний алгоритма в зависимости от обучающей выборки
- Маленький разброс – устойчивая к изменениям в данных модель
- Большой разброс – сильно переобученная чувствительная модель

Случайный шум

$$\sigma^2 = \mathbb{E}\varepsilon^2$$

Случайный шум обусловлен природой самих данных. Причины, по которым он возникает в данных

- Данные на самом деле имеют случайный характер
- Измерительный прибор не может зафиксировать целевую переменную абсолютно точно
- Имеющихся признаков не достаточно, чтобы исчерпывающим образом описать связь между целевой переменной и признаками объекта x

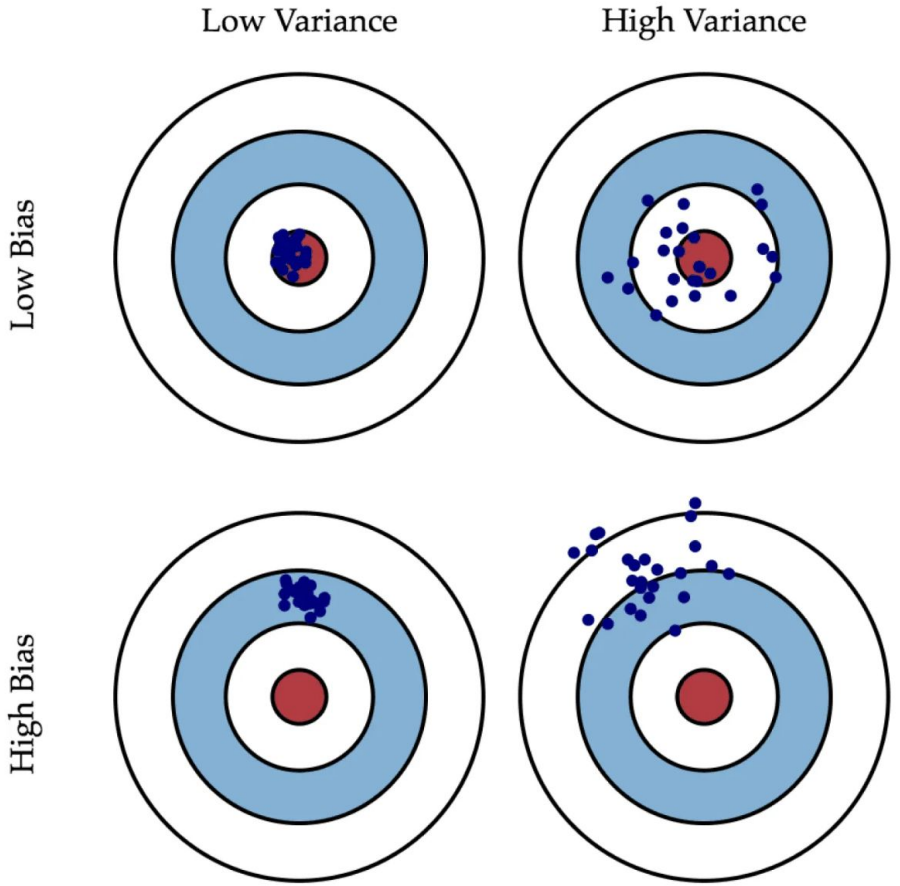
Краткая форма записи

С учётом введённых обозначений среднюю ошибку алгоритма можно представить как:

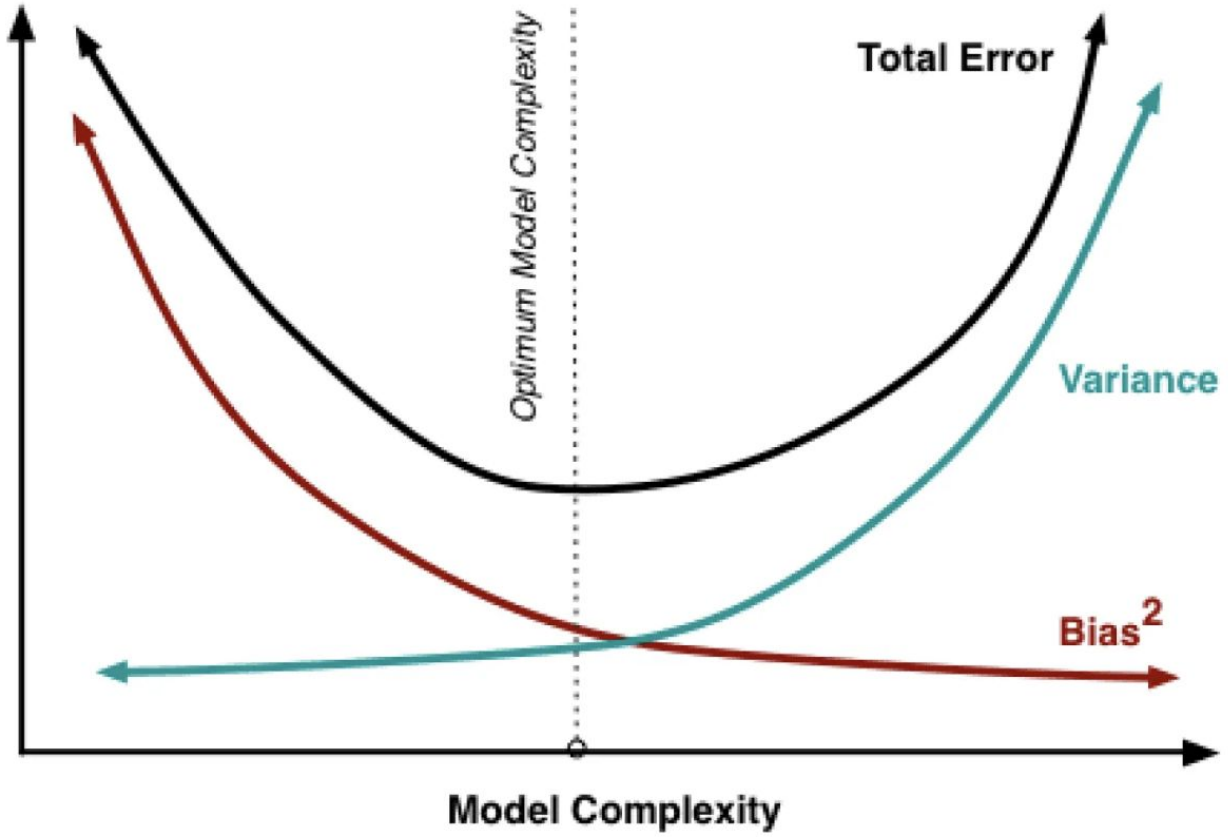
$$\mathbb{E}_X[Err] = bias_X^2(a(x, X)) + Var_X[a(x, X)] + \sigma^2$$

Смещение и разброс визуализация

- Каждая синяя точка – модель обученная на некоторой обучающей выборке
- Красный круг в центре– ближайшая окрестность целевого решения



Bias-Variance Tradeoff



Есть ли из этого выход?

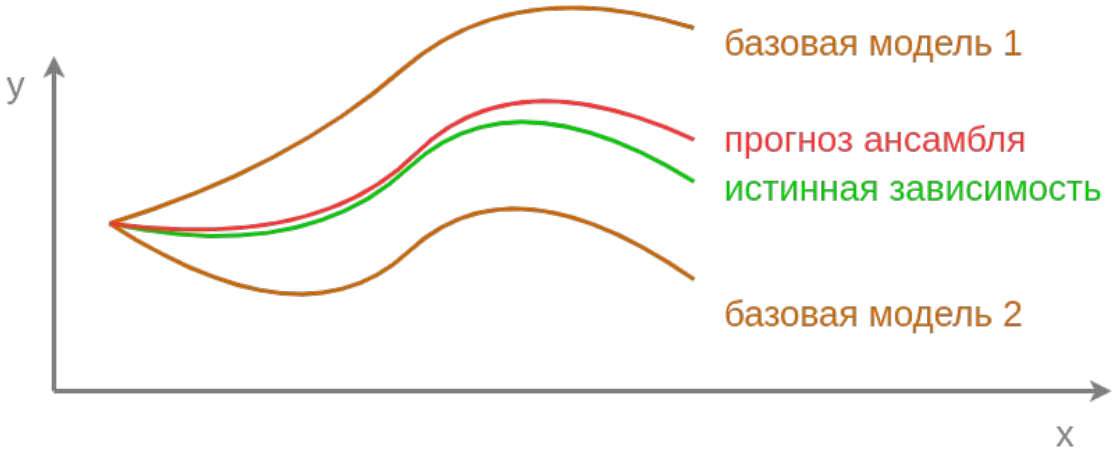
Может возникнуть вопрос, как научиться бороться с проблемой большого смещения у слишком простых моделей и большого разброса у слишком сложных моделей?

Существует способ уменьшить ту или иную компоненту ошибки через использование **композиции** (ансамбля) нескольких моделей

Ансамбль моделей

Ансамблем моделей называется подход, который строит свой прогноз, используя не одну модель $f(x)$, а совокупность **базовых** моделей $f_1(x), \ldots, f_M(x)$ и **агрегирующую мета-модель** $G(.)$, которая учитывает прогнозы всех базовых моделей:

$$\hat{y}(x) = G(f_1(x), \ldots, f_M(x))$$

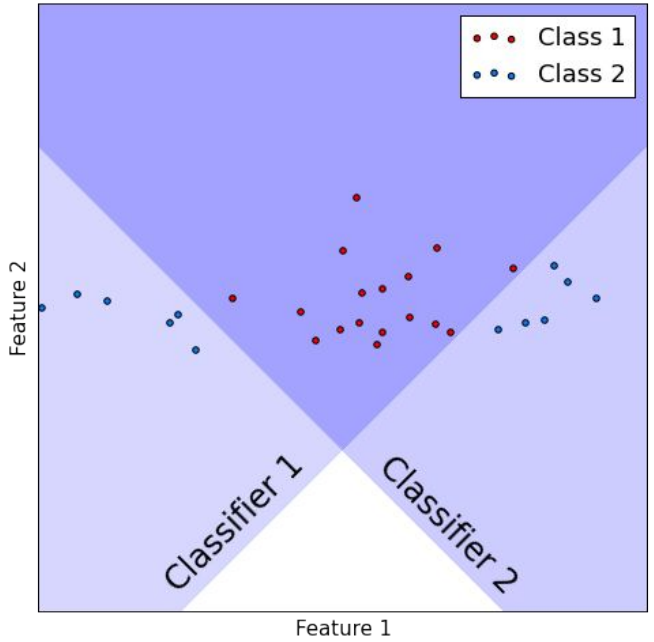


Пример ансамблирования простых моделей

А что если попробовать взять логическую операцию И над ответами полученных моделей…

$$a(x) = a_1(x) \wedge a_2(x)$$

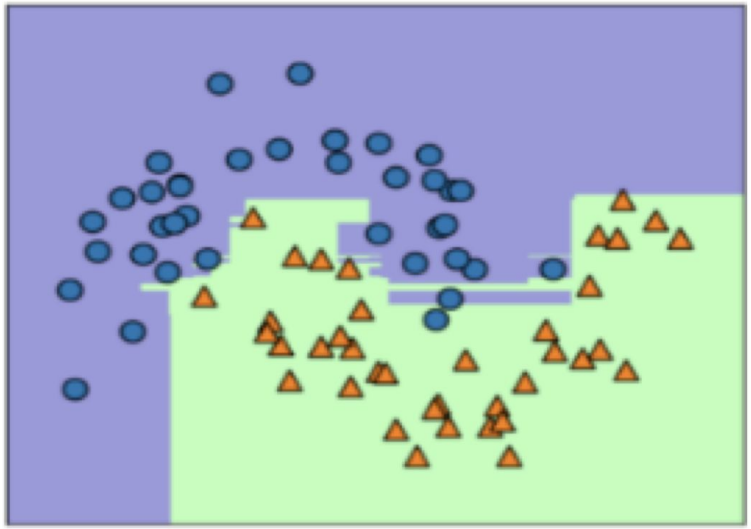
То полученная модель будет иметь нулевую ошибку!



Пример ансамблирования сложных моделей

Попробуем усреднить ответы нескольких деревьев используя голосование классификаторов

$$a(x) = mode(a_1(x), \ldots, a_4(x))$$



И опять за счёт объединения нескольких моделей удалось улучшить качество решения!

Виды ансамблирования

Существует множество различных способов, как произвести агрегацию ответов базовых алгоритмов $\{b_1, ..., b_M\}$:

- Брать среднее $a(x) = mean(b_1(x), ..., b_M(x))$
- Брать медиану $a(x) = median(b_1(x), ..., b_M(x))$
- Брать взвешенное среднее $a(x) = (w_1 b_1(x) + ... + w_M b_M(x))$

Также существуют различные способы голосования (для классификации):

- Голосование по большинству $a(x) = mode(b_1(x), ..., b_M(x))$
- Комитет единогласия $a(x) = min(b_1(x), ..., b_M(x))$

Теоретическое обоснование

Пусть ответы модели – независимые случайные величины $\{a_1, \ldots, a_m\}$ с одинаковым мат. ожиданием и дисперсией.

Тогда справедливо

- $a = \frac{1}{m}(a_1 + \ldots + a_m)$
- $\mathbb{E}a = \frac{1}{m}(\mathbb{E}a_1 + \ldots + \mathbb{E}a_m) = \{\mathbb{E}a_i = \mathbb{E}a_j,\ \forall i, j\} = \mathbb{E}a_1$
- $Da = \frac{1}{m^2}(Da_1 + \ldots + Da_m) = \{Da_i = Da_j, \forall i, j\} = \frac{Da_1}{m}$

Более общий вывод для случая, когда есть корреляция между предсказаниями можно посмотреть здесь

Построение независимых моделей

Важное допущение, которое было сделано в предыдущем выводе, состоит в том, что все предсказания были получены от независимых моделей.

Вопрос: Возможно ли получить такие модели, если все они были обучены на одном и том же множестве с одним и теми же параметрами обучения?

Построение независимых моделей

Важное допущение, которое было сделано в предыдущем выводе, состоит в том, что все предсказания были получены от независимых моделей.

Вопрос: Возможно ли получить такие модели, если все они были обучены на одном и том же множестве с одним и теми же параметрами обучения?

Ответ: Нет, невозможно, но можно попробовать сделать более независмимыми друг от друга.

Построение независимых моделей

Существует несколько способов построения, как можно более независимых моделей:

- Использовать в ансамбле модели разных классов, например, использовать в
- Использовать различные гиперпараметры при обучении
- Использовать разную начальную инициализацию
- Использовать различные подмножества обучающей выборки
- Использовать разные функции потерь

Чем более независимые модели удастся построить тем более сильный прирост даст ансамблирование

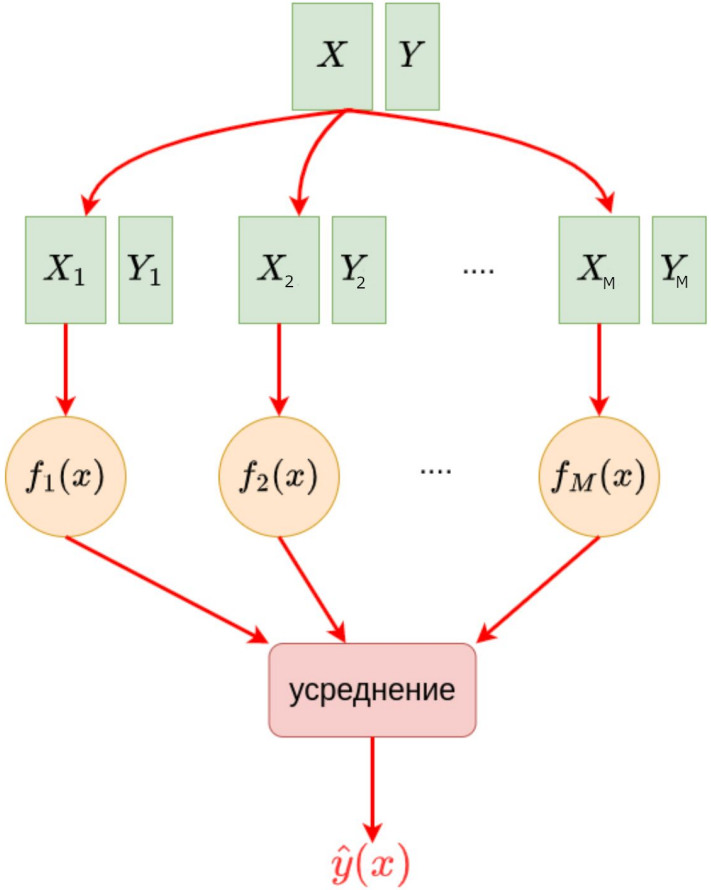
Настройка на разных наборах обучающих данных

Сгенерируем из исходной выборки (X, Y)
M фрагметов

$(X, Y) \rightarrow (X_1, Y_1), (X_2, Y_2), \ldots, (X_{M,} Y_M)$

Называемыми далее псевдовыборками, и настроим на каждой псевдовыборке одну и ту же модель a(x)

Процесс ансамблирования по псевдовыборкам

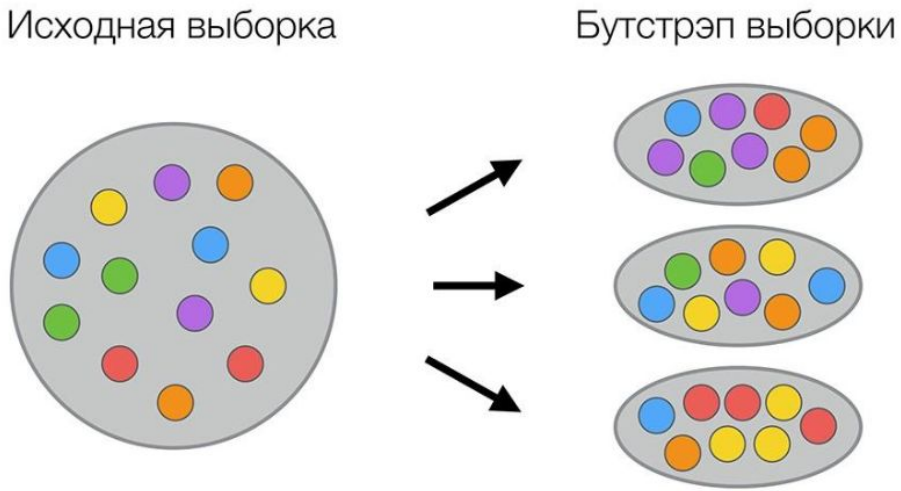


Бутстреп

Существует различные способы получения псеводовыбок, но наибольшее распространение получил подход бутстреп.

Суть метода состоит в том, чтобы

- выбрать из исходной выборки X равномерно M объектов с возвращением
- повторить данную процедуру K раз



Пересечение объектов

Так как происходит возврат объектов, то могут быть пересечение обучающих данных в пределах одной подвыборки:

- Вероятность не выбрать объект: $1 - \frac{1}{N}$
- Вероятность не выбрать объект N раз: $(1 - \frac{1}{N})^N \overset{N \to \infty}{\to} \frac{1}{e}$
- Вероятность встретить объект в выборке хотя бы 1 раз: $1 - \frac{1}{e} \approx 0.63$

Таким образом каждая выборка будет в среднем содержать 63% уникальных объектов

Бэггинг

Бэггинг – (bagging – bootstrap aggregating) – строим алгоритм, обучая каждый базовый алгоритм на $b_j(x)$ на своей псевдовыбороке (X_j, Y_j) и усредняя их предсказания:

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

- Исходя из предыдущих теоретических результатов, бэггинг не ухудшает смещение базовой модели, но при этом способен минимизировать её разброс

Другие способы построения псевдовыборок

Помимо бэггинга существуют другие способы построения псевдовыборок:

- Кросс-валидация – получаем K подвыборок с пересечением (1 - 1/K)
- Пейстинг – псевдобыборка генерируется из исходной через семплирование объектов без возвращения
- Метод случайных подпространств – берутся все объекты, но множество признаков берётся случайным без возвращение
- Метод случайных фрагментов – комбинируются сэмплирование объектов и признаков без возвращение

Случайный лес (Random forest)

Если в качестве базового алгоритма $b_j(x)$ использовать дерево решений, обученное на своей псевдослучайной выборке (X_j, Y_j), и после чего усреднить их ответы, то получим алгоритм, который называется **случайным лесом**

- В каждой вершине дерева ищем разбиение не по всем признакам, а по их подмножеству
- Дерево строится до тех пор, пока в листе не окажется n_{min} объектов

Случайный лес (Random forest)

Алгоритм 3.1. Random Forest

1: **для** $n = 1, \ldots, N$

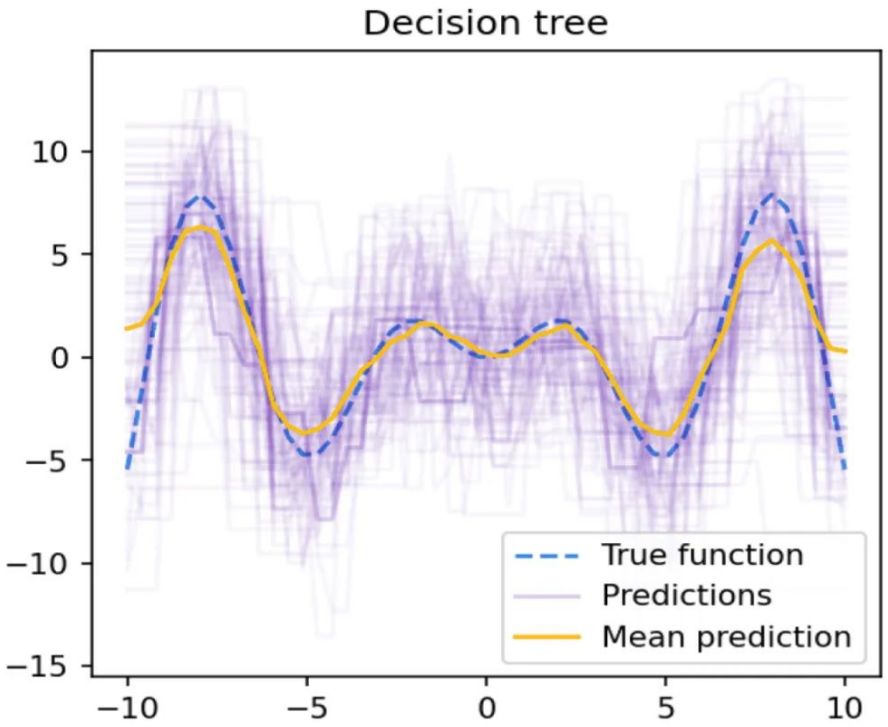
2: Сгенерировать выборку $\tilde{X}_n$ с помощью бутстрэпа

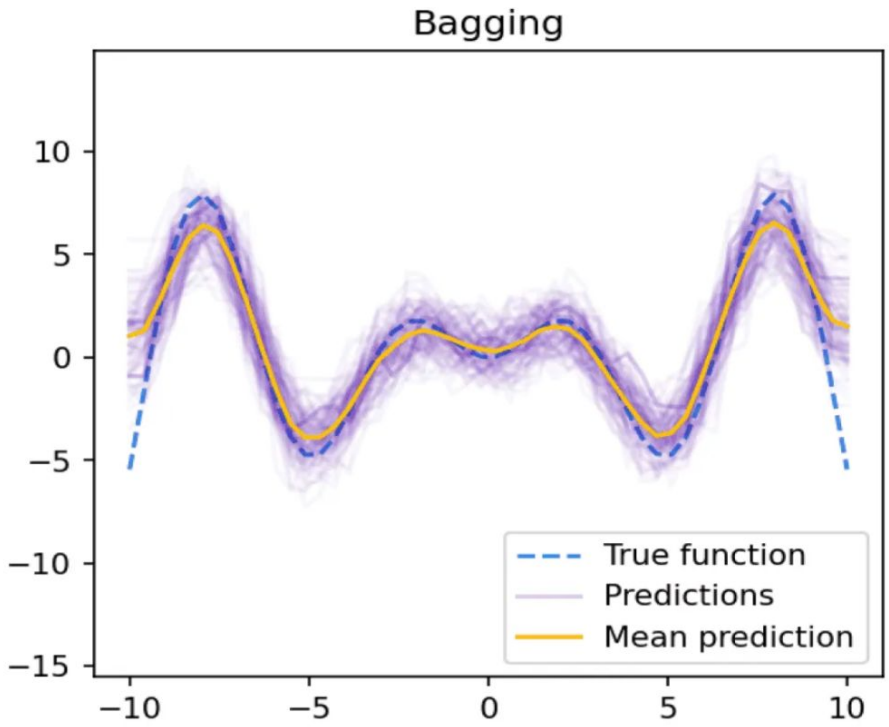
3: Построить решающее дерево $b_n(x)$ по выборке $\tilde{X}_n$:

- дерево строится, пока в каждом листе не окажется не более $n_{\min}$ объектов
- при каждом разбиении сначала выбирается m случайных признаков из p, и оптимальное разделение ищется только среди них

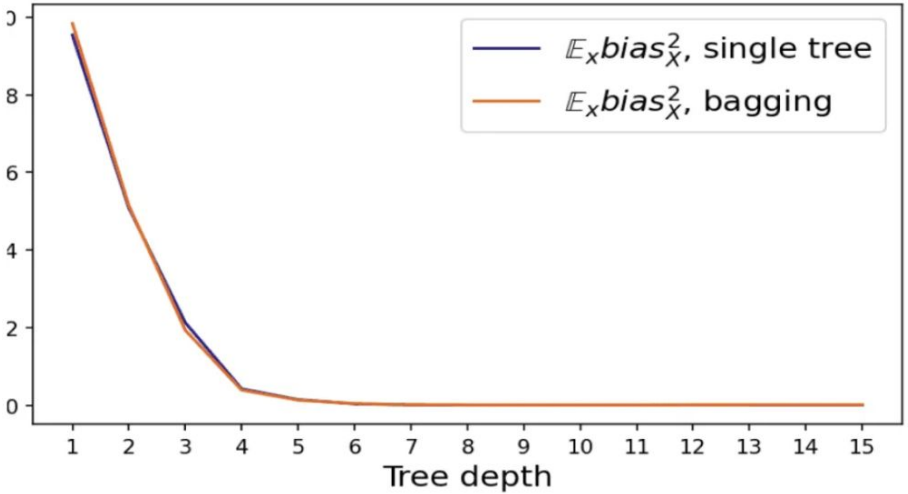
4: Вернуть композицию $a_N(x) = \frac{1}{N} \sum_{n=1}^{N} b_n(x)$

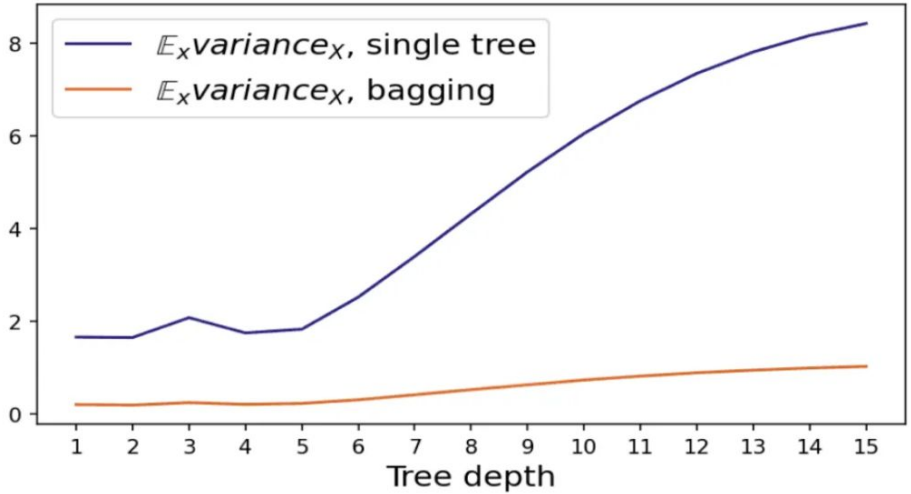
Практический эксперимент





Смещение и разброс у случайного леса





Полученные результаты согласуются с теоретическим выводом:

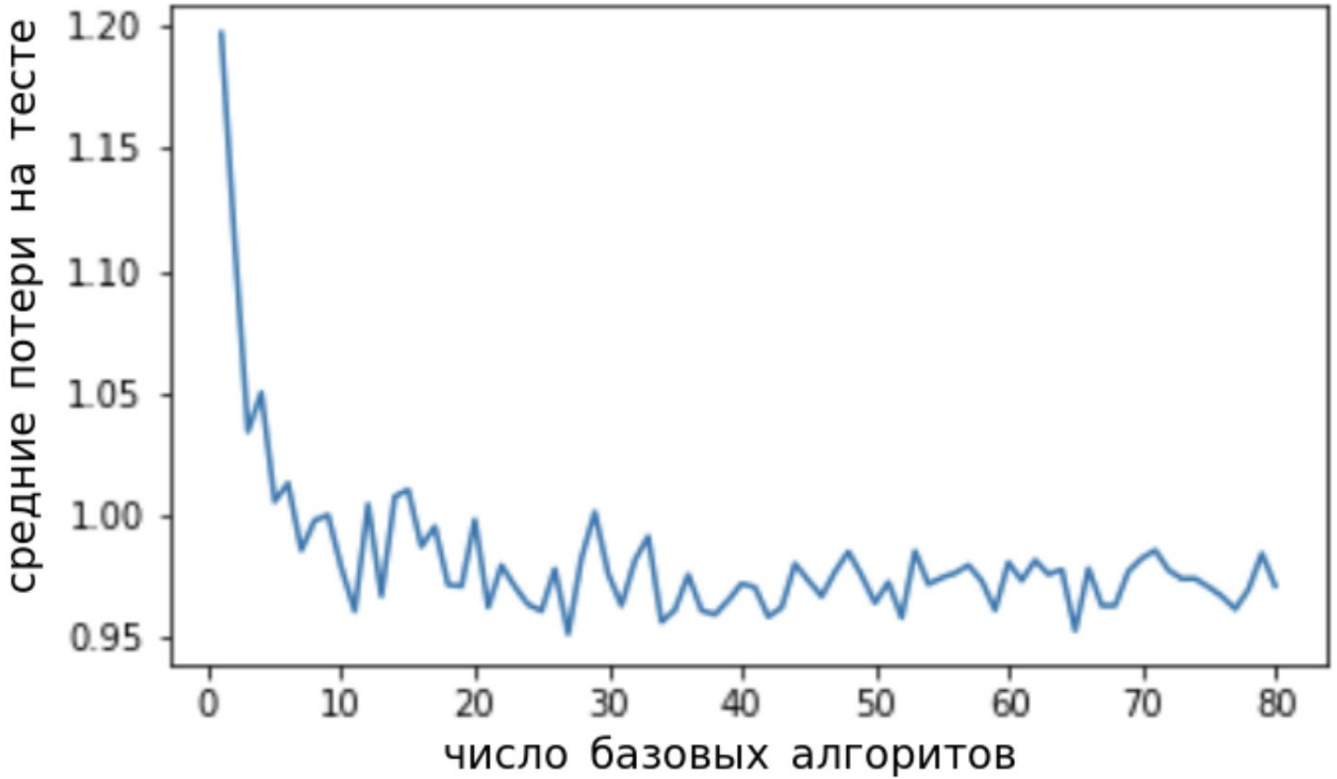
- Смещение не меняется при усреднении
- Значительно снизился разброс

Глубина базовых алгоритмов

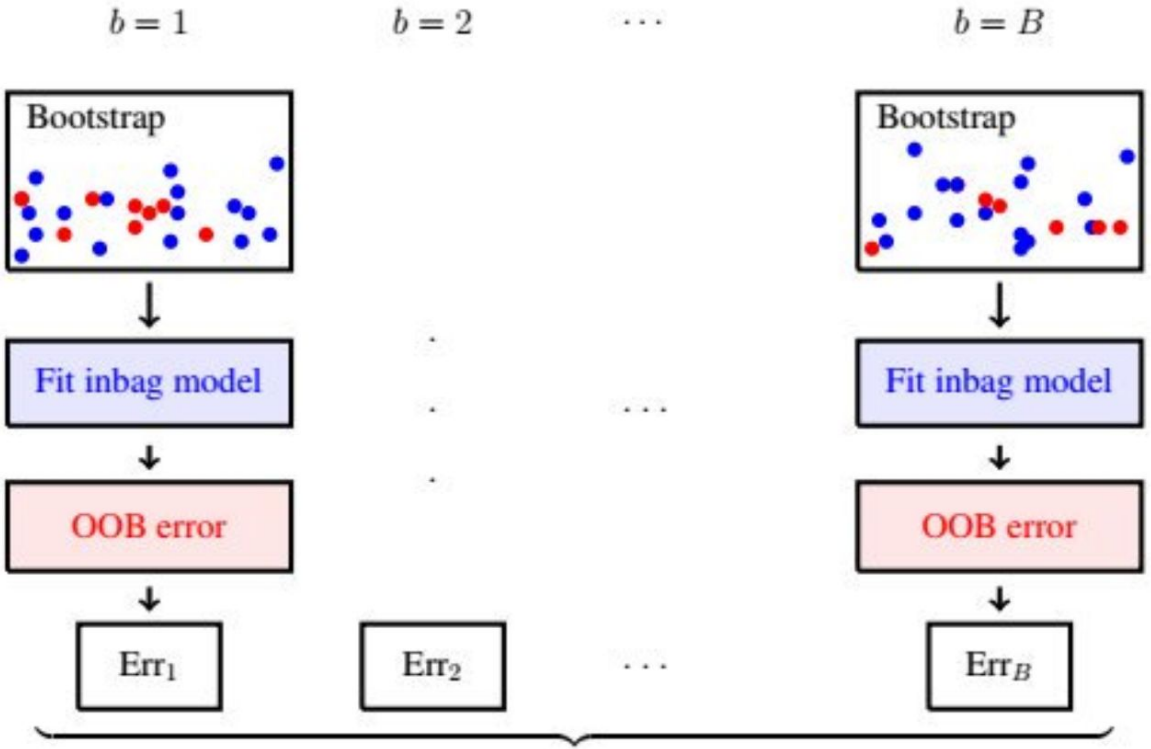
- Неглубокие деревья имеют малое число параметров, поэтому плохо учитывают закономерности в данных и имеют **большое смещение**
- Глубокие деревья наоборот слишком сильно запоминают выборку и имеют слишком большой разброс

Вывод: Так как усреднение деревьев не способно изменить смещение базового алгоритма, но способно уменьшить его разброс, то имеет смысл использовать **глубокие деревья**

Сколько деревьев использовать



Out-of-bag ошибка



$$\mathrm{Err}_{\mathrm{oob}} = \frac{\mathrm{Err}_1 + \cdots + \mathrm{Err}_B}{B} = \frac{1}{B}\sum_{b=1}^{B} \mathrm{Err}_b$$

Out-of-bag ошибка

- Каждое дерево в случайном лесе обучается по некоторому подмножеству объектов
- Значит для каждого объекта x_n есть деревья, которые на этом объекте не обучались

Пусть *I(n)* – множество псевдовыборок, куда объект x_n не попал.

Тогда out-of-bag ошибка:

$$OOB = \sum_{n=1}^{N} L(y_n, \frac{1}{|I(x_n)|} \sum_{i \in I(n)} b_i(x_n))$$

Стекинг

В предыдущих примерах при агрегации базовых моделей, использовались достаточно простые методы агрегации: среднее (для регрессии) и голосование (для классификации).

Но можно иметь мета-алгоритм G(.), который сам будет некоторой **обучаемой моделью** со своими параметрами.

$$\hat{y}(\mathbf{x}) = G(f_1(\mathbf{x}), f_2(\mathbf{x}), ...f_M(\mathbf{x}))$$

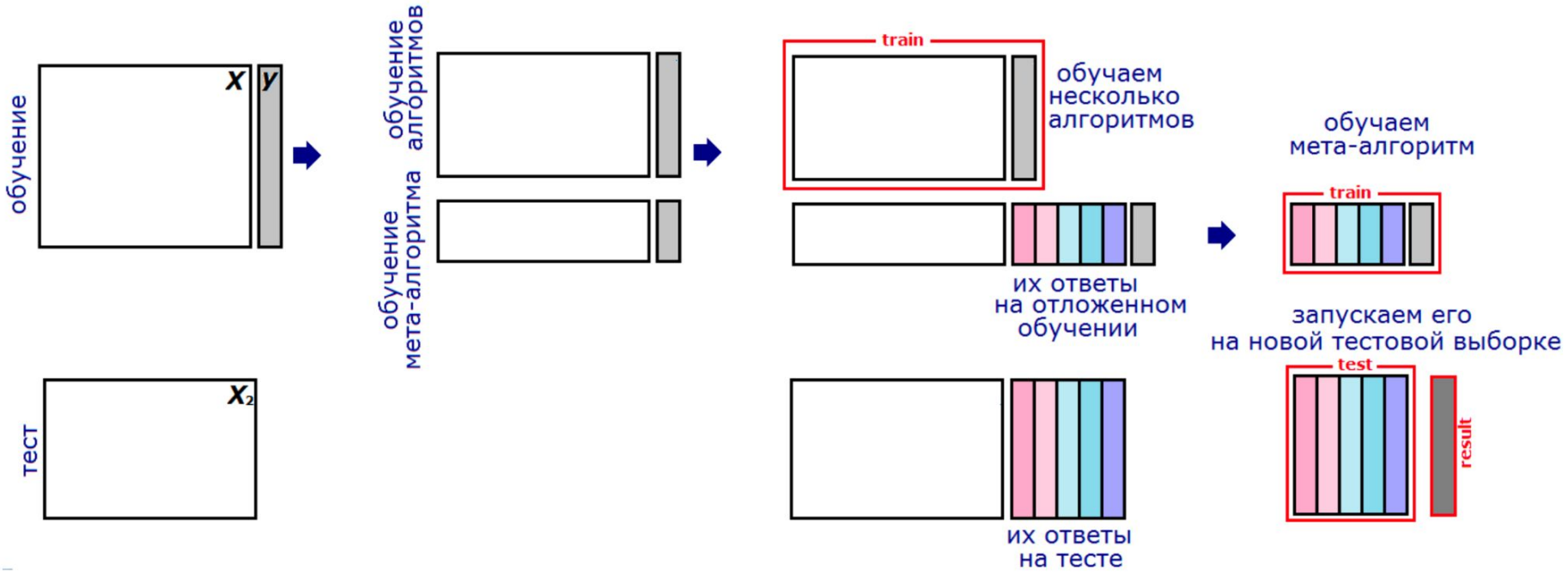
Линейный стекинг

Самым простым видом мета-алгоритма является линейная модель, которая работа на ответах базовых алгоритмов:

$$\hat{y}(\mathbf{x}) = w_0 + w_1 f_1(\mathbf{x}) + w_2 f_2(\mathbf{x}) + ... + w_M f_M(\mathbf{x})$$

- Вес w_0 нужен если у базовых алгоритмов есть систематическое смещение и они недообучены, если такой проблемы нет, то w_0
- При настройке базовых алгоритмов f_1, .., f_M и мета-алгоритма G(.) нельзя использовать одну и ту же выборку иначе будет происходит переобучение

Стекинг. Схема обучения



Выводы

- Позволяют простыми методами серьёзно улучшить качество работы базового алгоритма
- В зависимости от склонности к недообучению или переобучению базового метода необходимо подбирать соответствующий мета-алгоритм G
- Так как происходит увеличение количество параметров и гиперпараметров в исходной модели, то нужны обучающие выборки большего размера
- Чтобы получить хороший эффект от ансамблирования, необходимо строить как можно более независимые друг от друга базовые модели