Лекция 10

Ансамбли, бустинг

Зачем нам много моделей?

Идея с прошлого занятия

01

Много простых деревьев лучше, чем одно сложное?

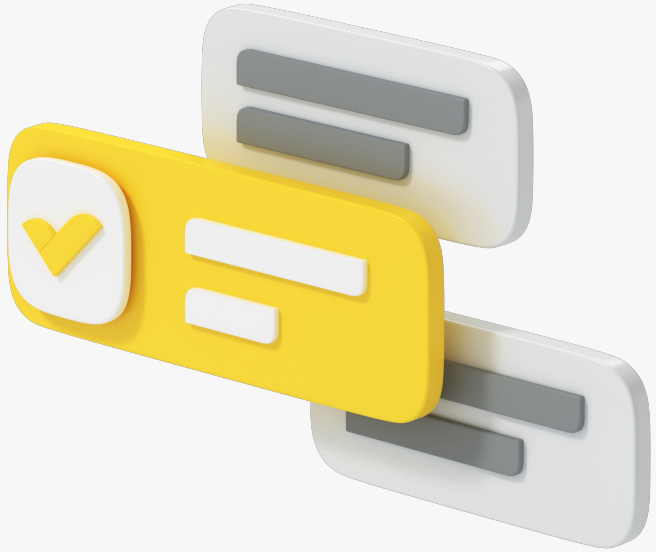
02

Можно ли то же самое сказать про другие алгоритмы?

03

Как получать ответ, когда много моделей?

Какие есть варианты?



01

Блендинг

02

Стекинг

03

Бэггинг

04

Бустинг

Блендинг & Стекинг

Базово похожие идеи

1	2	3	4	5
Возьмём разные модели	Обучим на одинаковых данных	Возьмем одну дополнительную модель	Обучим на предсказаниях слабых моделей	Профит

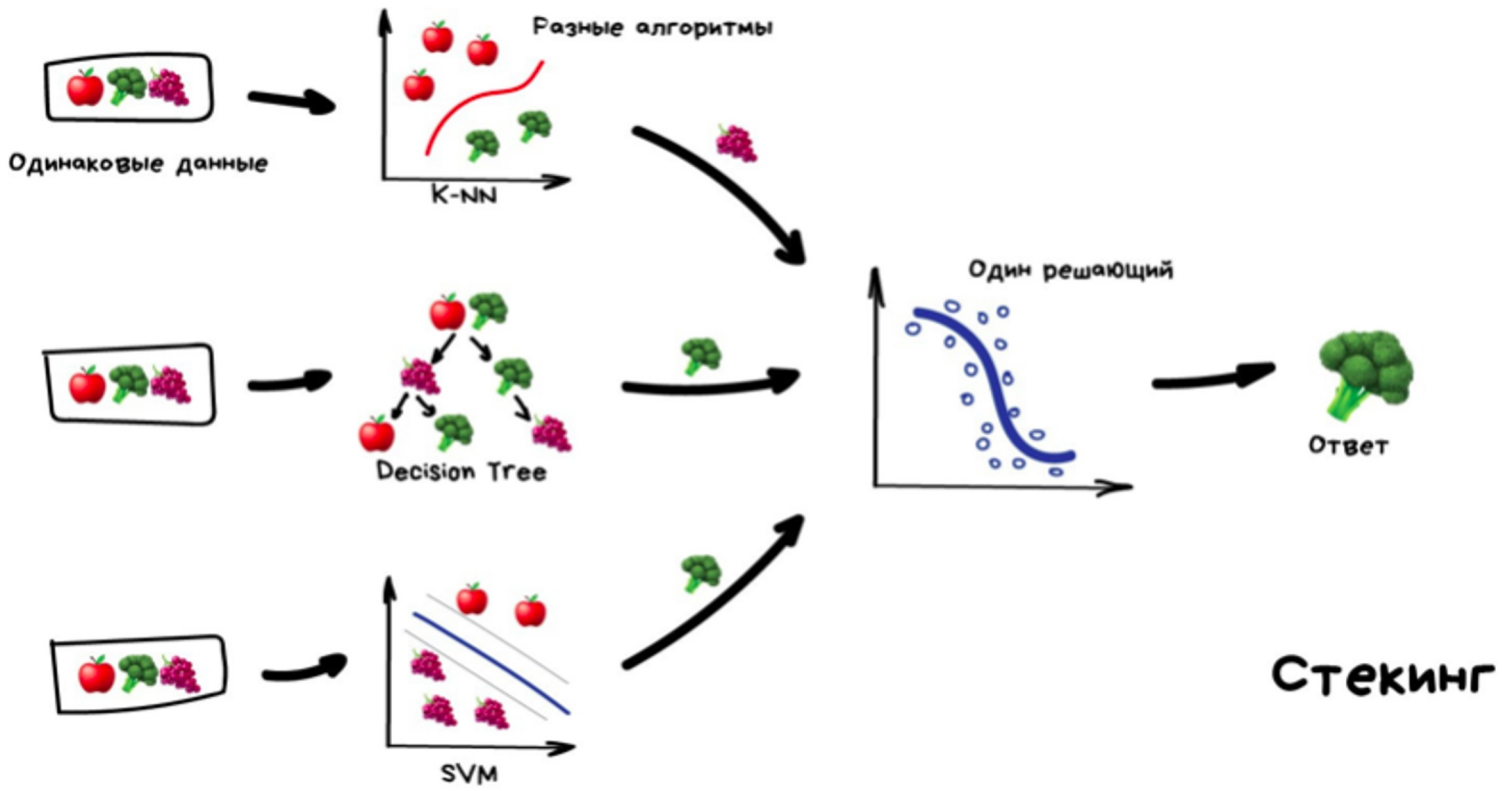
- Английский термин weak-learners
- По-русски можно назвать базовыми моделями

- Её также называют метамоделью

Как учат метамодель?

- Пусть у нас есть N объектов с M признаками
- Обучаем на такой выборке K базовых моделей
- Каждой базовой моделью делаем предсказание
- Получаем новую выборку из N объектов с K признаками
- На этой выборке обучаем метамодель

Стекинг


Разные алгоритмы
Одинаковые данные
K-NN
Один решающий
Ответ
Decision Tree
SVM
Стекинг

В чём разница?

Различия в подходе к обучению метамодели

В случае блендинга

- Разбиваем на train, valid & test
- Базовые модели обучаем на train
- Предсказываем valid
- Метамодель учим на valid
- Проверяем всю цепочку на test

В случае стекинга

- Разбиваем на train & test
- Базовые модели обучаем на train
- Предсказываем train
- Метамодель учим также на train
- Проверяем всю цепочку на test

Бэггинг

Сокращение от bootstrap aggregating

Что такое bootstrap или bootstrapping?

Создание случайных подвыборок из исходной

Что нам это дает?

Модели смотрят на разные части данных
Их усреднение будет более устойчивым к шумам

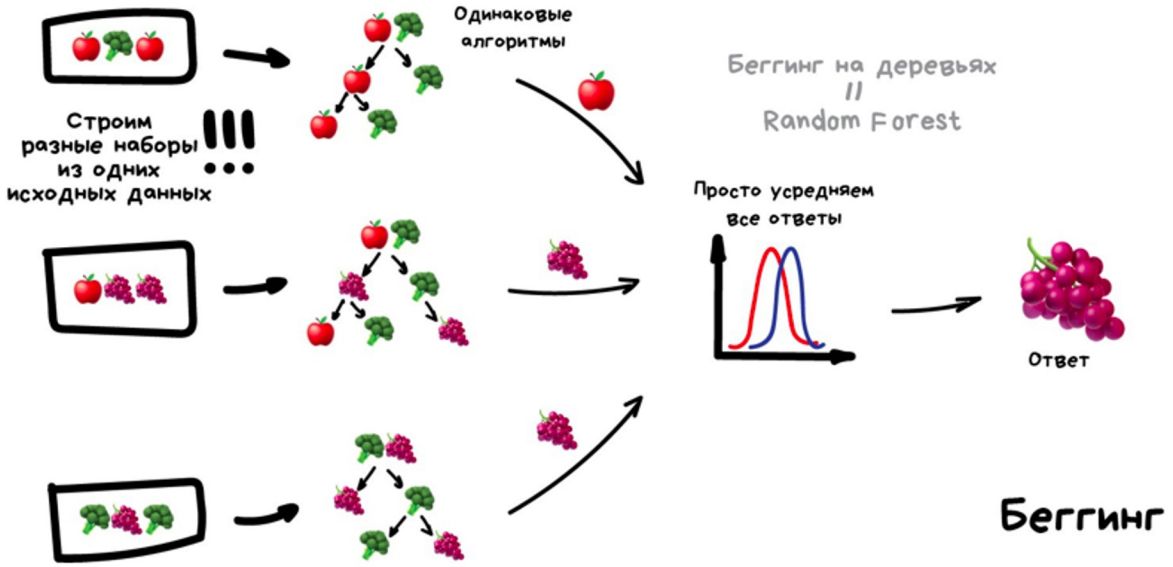
Бэггинг

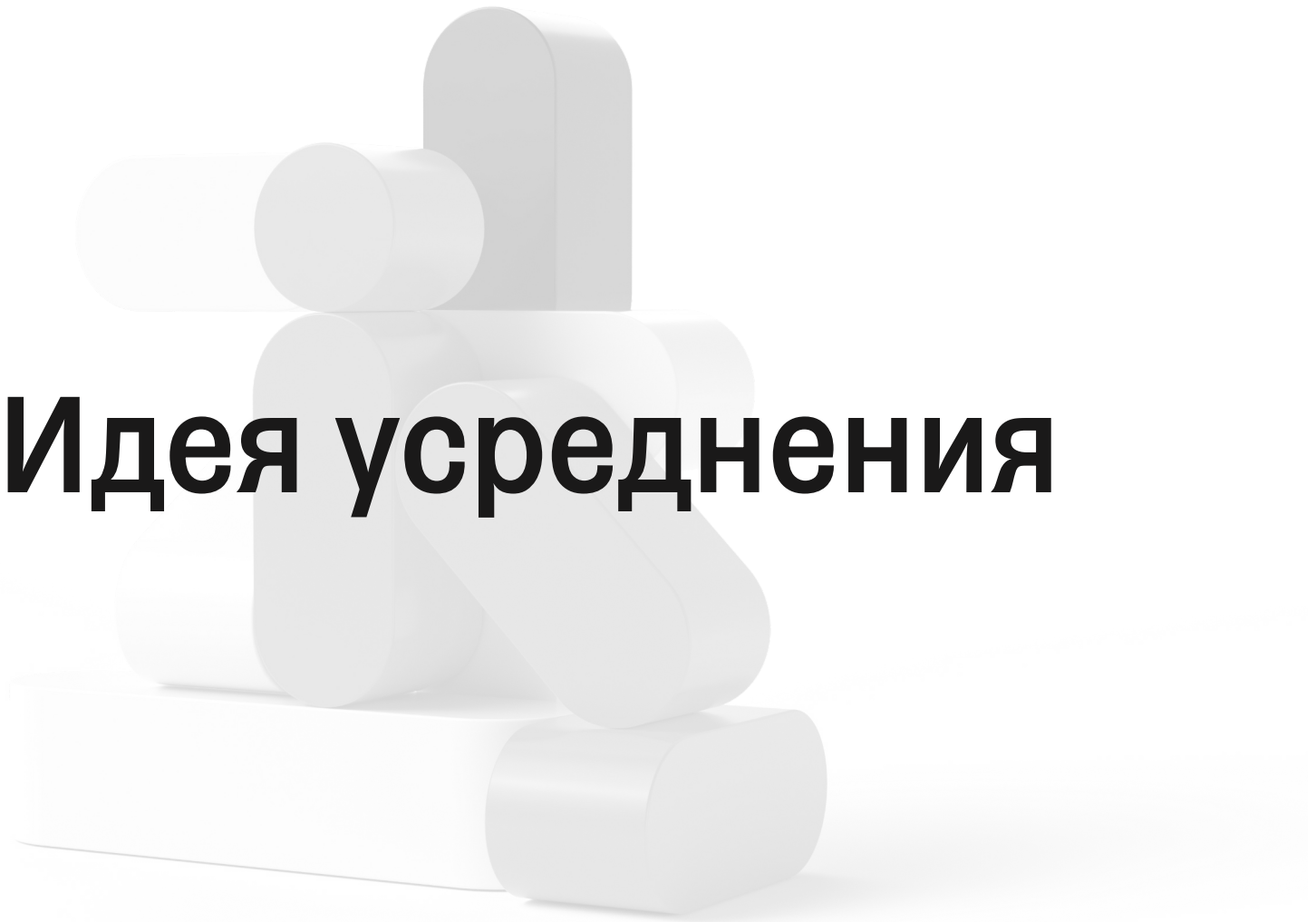
- В качестве базовых моделей можно брать любой алгоритм
- Bootstrapping можно делать еще и по признакам

Часто берут деревья

- Деревья склонны переобучаться на шуме
- Если сделать их устойчивыми, получится сильная модель

Такая модель называется случайным лесом





Идея усреднения

Усреднение

Почему работают стекинг и бэггинг?

Сформулируем теорему:

- Пусть у нас есть k **независимых** случайных величин $\{X_1, X_2, \dots, X_k\}$
- Усредним эти случайные величины, получим $\bar{X}$
- Тогда дисперсия $\bar{X}$ будет уменьшаться с ростом k
- Доказательство

Ключевые моменты

- Случайным величинам достаточно быть некоррелированными
- Пусть наши данные распределены нормально с $\mu = 0, \sigma = 1$
- Так как матожидание линейно, то оно не изменится
- Это удобно, т.к. в ML всё крутится вокруг нормальности данных

Выводы

Стекинг, блендинг и бэггинг имеют схожую природу

Что хотим?

- Сделать много случайных величин
- Усреднить
- Уменьшить итоговую дисперсию

Аналогия в реальном мире

- Диверсификация инвест-портфеля

Бустинг

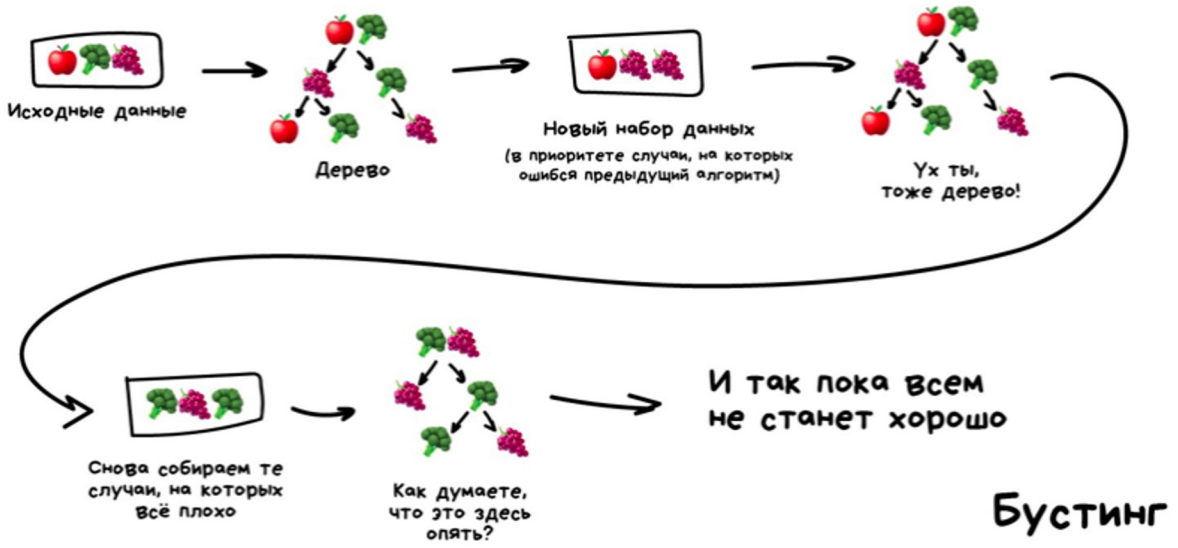
Чуть иная идея

1	2	3	4
Сделаем устойчивую базовую модель	Будем добавлять дополнительные модели	Ответы моделей сложим	Повторяем, пока модель не переобучилась

Бустинг

Разберём на примере линейной регрессии

- Базовая модель – $avg(y)$
- Как улучшить ответ?
- Выучим разницу между средним и ответами
- Предсказания не идеальны
- Добавляем новые модели
- Останавливаемся, если переобучились



Градиентный бустинг

Какой критерий качества при обучении линейной регрессии?

- Часто берем квадрат разности
- В чем преимущество MSE перед MAE?
- В чем недостаток?

Корректируем модель в несколько итераций

- В бустинге вместо коррекции весов добавляем новую модель
- Учим эту модель на остатках

Здесь остатки – производная от MSE по ответам

Хотим изменить ответы модели так, чтобы loss уменьшился

Градиентный бустинг

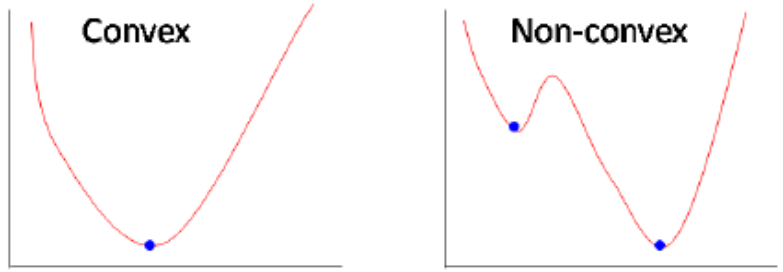
С задачей классификации есть трудности

Как заставить линейную модель выдать вероятности?

Что будет, если оптимизировать вероятности при помощи MSE?

Функция потерь должна соответствовать модели

Иначе модель застрянет в локальном минимуме или разойдется



Градиентный бустинг

Вспоминаем логистическую регрессию

- **Изначально** имеем $f(x) = x_1 a_1 + \cdots + x_m a_m$
- **Затем** применяем логистическую функцию $\tilde{y} = \frac{1}{1+e^{-f(x)}}$
- **Теперь** можем оптимизировать при помощи logistic loss: $-\mathcal{L} = y \times \log \tilde{y} + (1-y) \times \log(1-\tilde{y})$
  - Как выглядит производная от такой функции потерь?

Градиентный бустинг

Вспоминаем логистическую регрессию

- **Изначально** имеем $f(x) = x_1 a_1 + \cdots + x_m a_m$
- **Затем** применяем логистическую функцию $\tilde{y} = \frac{1}{1+e^{-f(x)}}$
- **Теперь** можем оптимизировать при помощи logistic loss: $-\mathcal{L} = y \times \log \tilde{y} + (1-y) \times \log(1-\tilde{y})$
  - Как выглядит производная от такой функции потерь?

$$-\frac{\partial \mathcal{L}}{\partial \tilde{y}} = y - \tilde{y} = y - \frac{1}{1+e^{-f(x)}}$$

Градиентный бустинг

Важная идея

- Каждый раз будем обучать регрессионную модель приближать остаток
- Просуммировали ответы регрессионных моделей
- Применили к сумме ответов сигмоиду
  - Таким образом получили вероятность
- Остатки считаем при помощи производной от логистической функции потерь

? Что делать, если это **не бинарная** классификация?

ТИНЬКОФФ

Вопросы?