



ML Professional

Gradient boosting



Проверить, идет ли запись

Меня хорошо видно && слышно?



Ставим "+", если все хорошо "-",
если есть проблемы



Тема вебинара

ML Professional

Gradient boosting



Игорь Стурейко

Руководитель курсов: Reinforcement Learning, ML Professional, ML Basic

**Teamlead, главный инженер проекта,
Физический факультет МГУ, PhD теоретическая физика**

Опыт:

Более 15 лет занимался прикладной математикой и мат моделированием (Data Scientist) (Python, C++) в НИИ ПАО Газпром

Анализ временных рядов, эволюционные модели, финансовые модели

@stureiko (TG)

LinkedIn: [igor-stureiko](#)

Правила вебинара



Активно
участвуем



Off-topic обсуждаем
в учебной группе
#ML-2024-02



Задаем вопрос
в чат



Вопросы вижу в чате,
могу ответить не сразу

Условные обозначения



Индивидуально



Время, необходимое
на активность



Пишем в чат



Говорим голосом

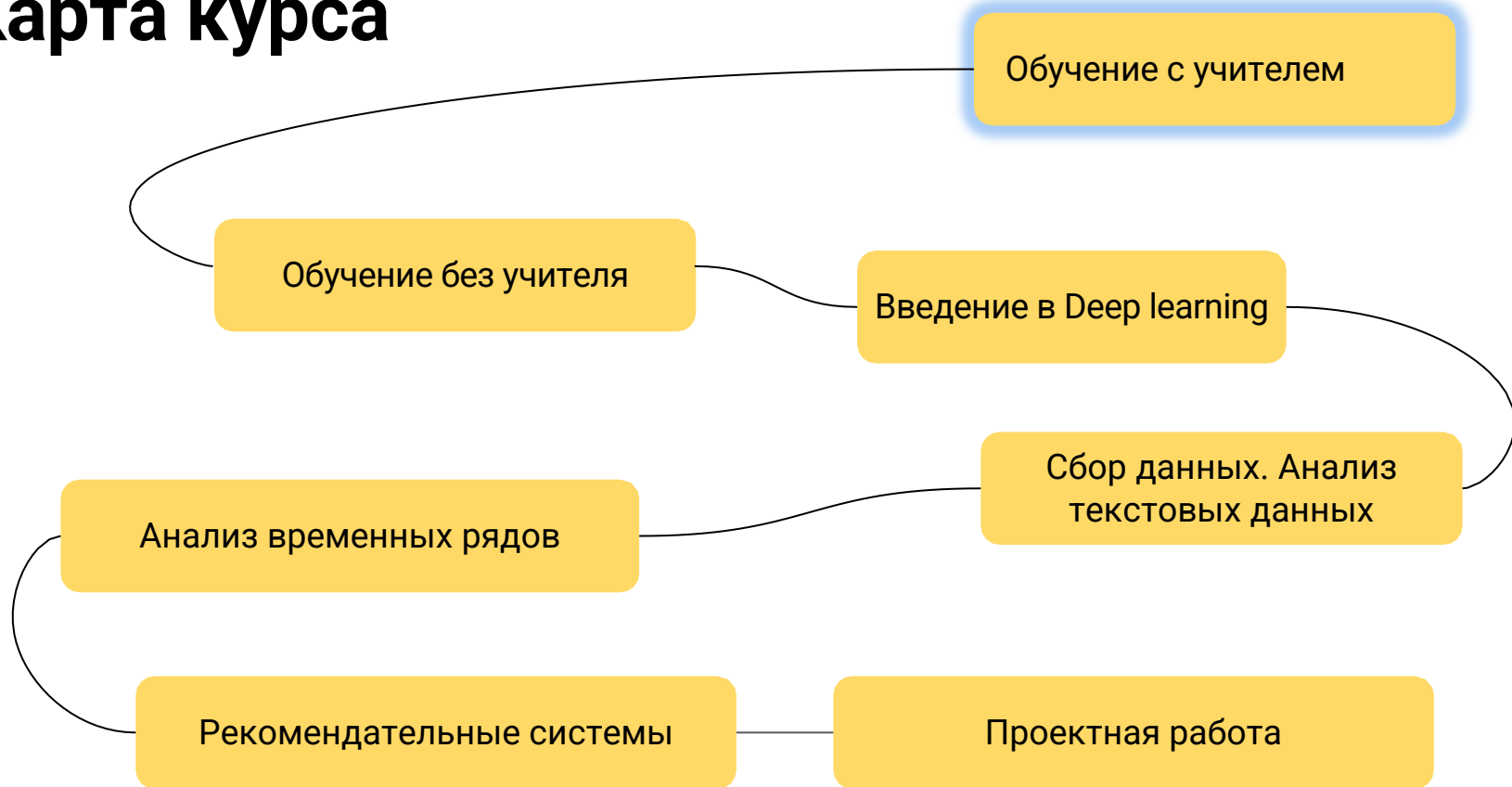


Документ



Ответьте себе или
задайте вопрос








Карта курса



Программа курса



Обучение с учителем

	29.02 -	Вводный урок
	07.03 -	Метод градиентного спуска
	11.03 -	EDA, cross-validation, метрики качества
	14.03 -	Деревья решений
	18.03 -	Ансамбли моделей
	21.03 -	Градиентный бустинг
	25.03 -	Метод опорных векторов



HomeTask

Маршрут вебинара

Идея

Последовательное улучшение

Градиентный бустинг над деревьями

Гиперпараметры

LightGBM | XGBoost | CatBoost

Практика применения



Цели занятия

Что вы сможете

1. Рассмотреть метод градиентного бустинга
2. Узнать, как применять современные библиотеки

Смысл

Зачем вам это уметь

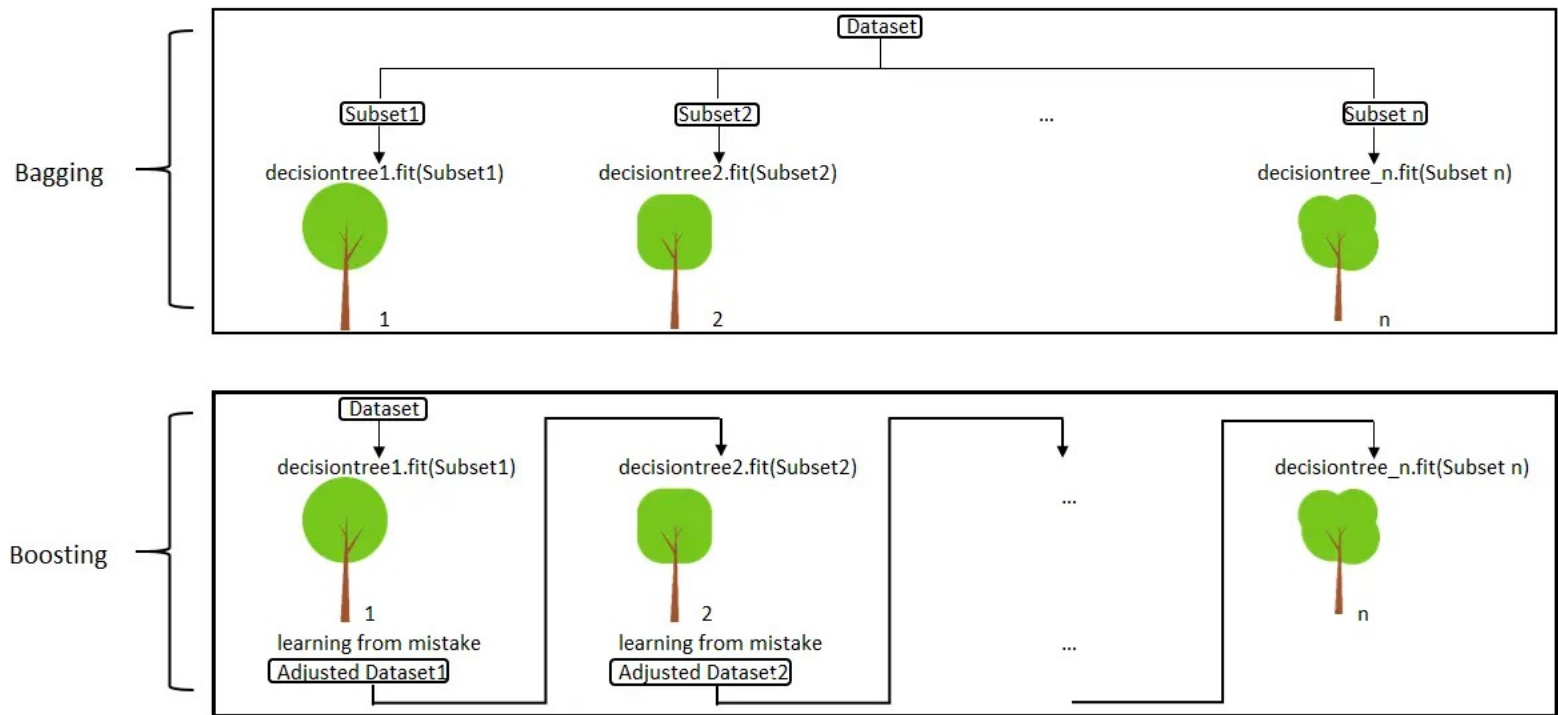
1. Понимать как применять методы градиентного бустинга
2. Знать особенности применения

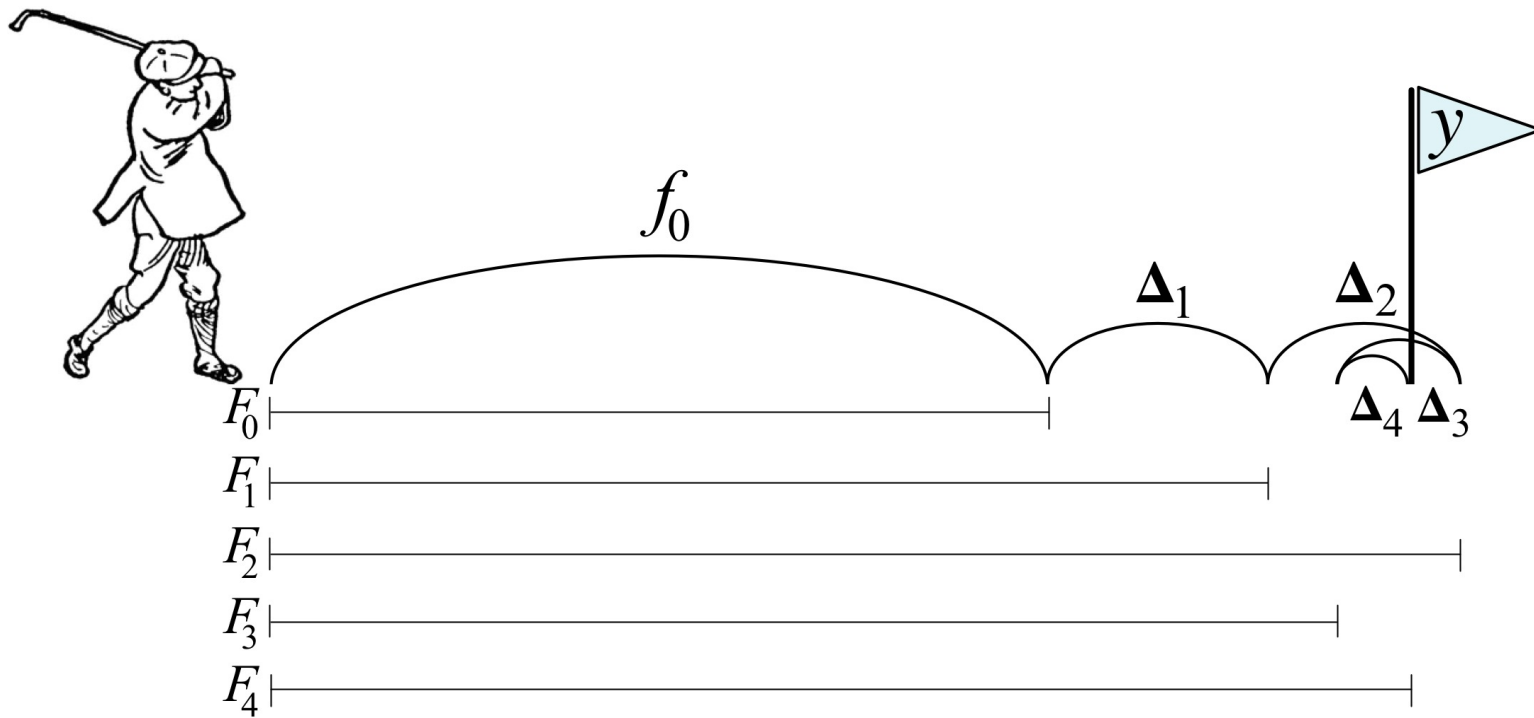


Gradient boosting

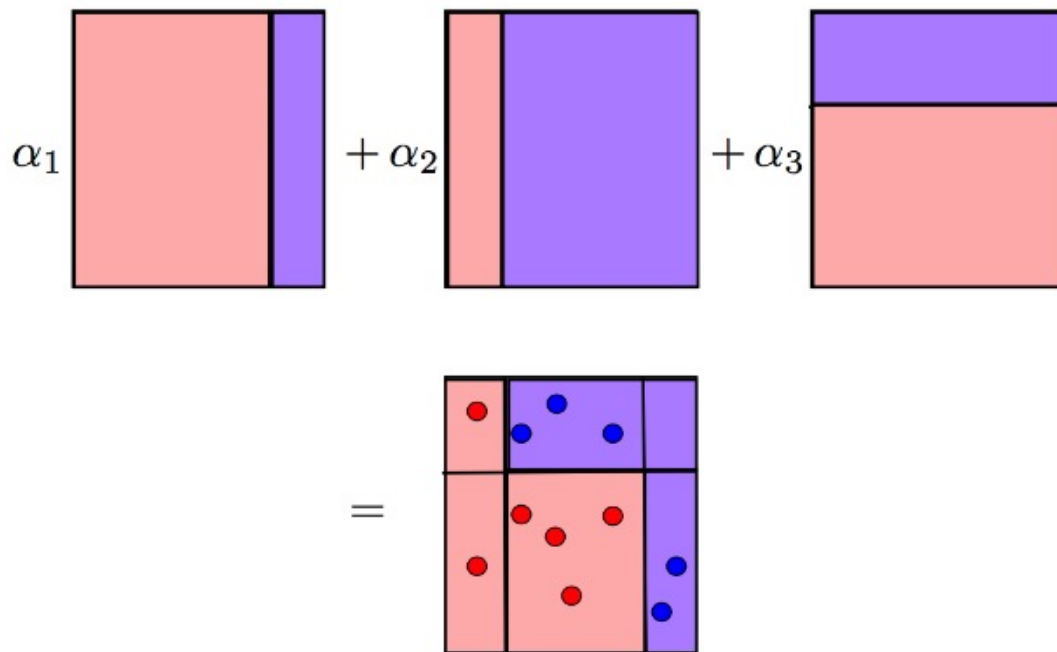
идея

Bagging vs Boosting





AdaBoost – Последовательное улучшение



Алгоритм - AdaBoost

- $a(x) = b_1(x) + b_2(x) + \dots + b_k(x)$ – последовательное приближение решения, $\mathcal{L}(y, x)$ - функция потерь
- Построим **первое** приближение: $b_1(x) = \operatorname{argmin}_{b \in \mathcal{B}} \mathcal{L}(y, b(x))$ на исходных данных и вычислим ошибки: $s_i^1 = y_i - \lambda b_1(x);$
- Построим **второе** приближение на ошибках первого: $b_2(x) = \operatorname{argmin}_{b \in \mathcal{B}} \mathcal{L}(s^1, b(x))$ и вычислим его ошибки: $s_i^2 = s_i^1 - \lambda b_2(x);$
- ...
- Приближение n на ошибках $n - 1$: $b_n(x) = \operatorname{argmin}_{b \in \mathcal{B}} \mathcal{L}(s^{n-1}, b(x)),$
 $s \leftarrow s^{n-1} - \lambda b_n;$

Особенности бустинга

Бустинг - итерационный алгоритм, реализующий «сильный» классификатор, который позволяет добиться произвольно малой ошибки обучения на основе композиции «слабых» классификаторов, каждый из которых немного лучше, чем просто угадывание.

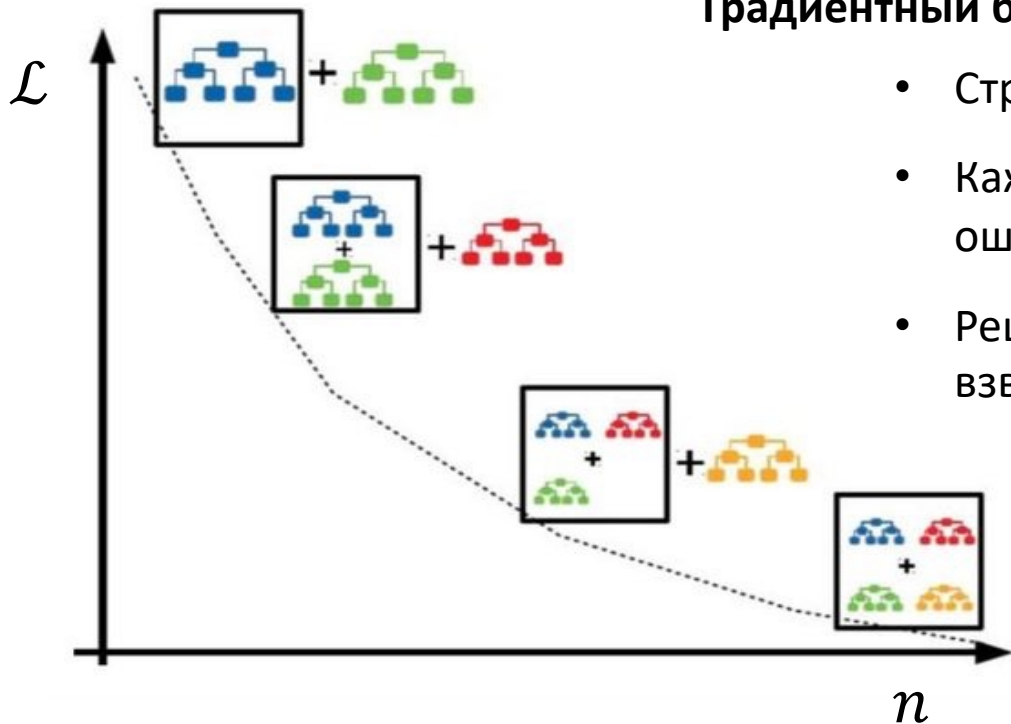
Базовые классификаторы должны быть слабыми, из сильных хорошую композицию не построить («бритва Оккама»)

- сильный классификатор, давая нулевую ошибку на обучающих данных, не адаптируется и композиция будет состоять из одного классификатора
- один, даже сильный, классификатор может дать «плохое» предсказание на данных тестирования, давая «хорошие» результаты на обучающих данных.

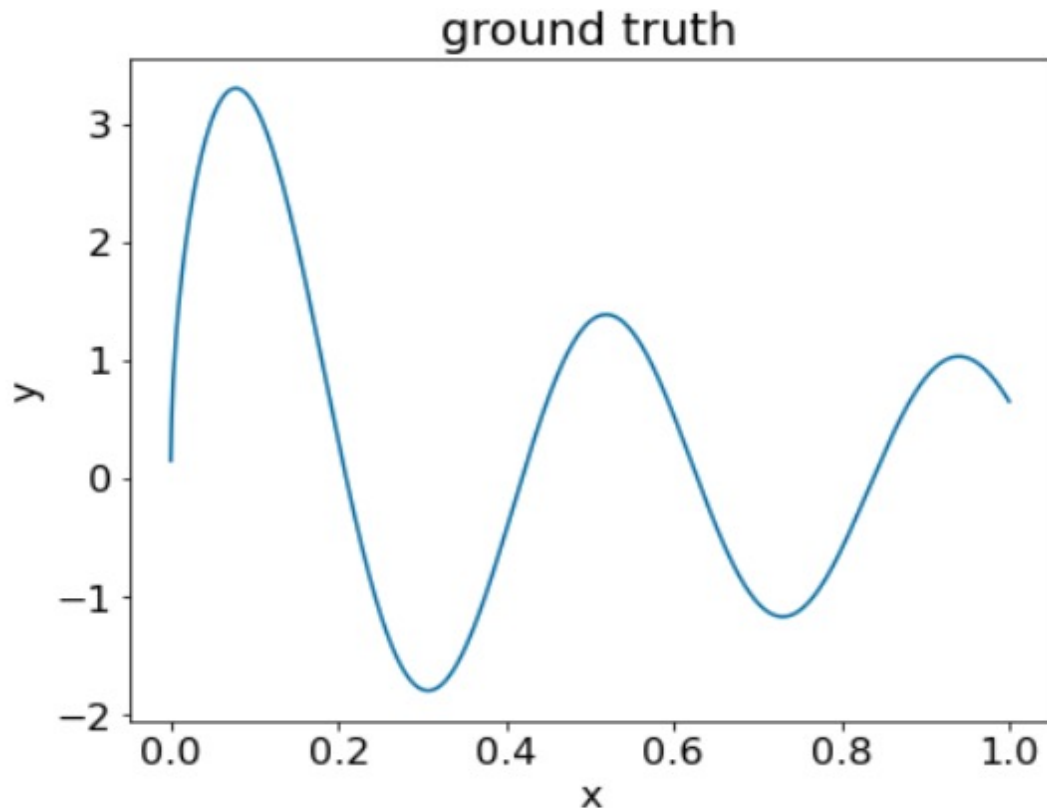
Gradient boosting

Градиентный бустинг над решающими деревьями

- Строим алгоритмы последовательно
- Каждый следующий строится на ошибках предыдущего
- Решение принимается методом взвешенного голосования

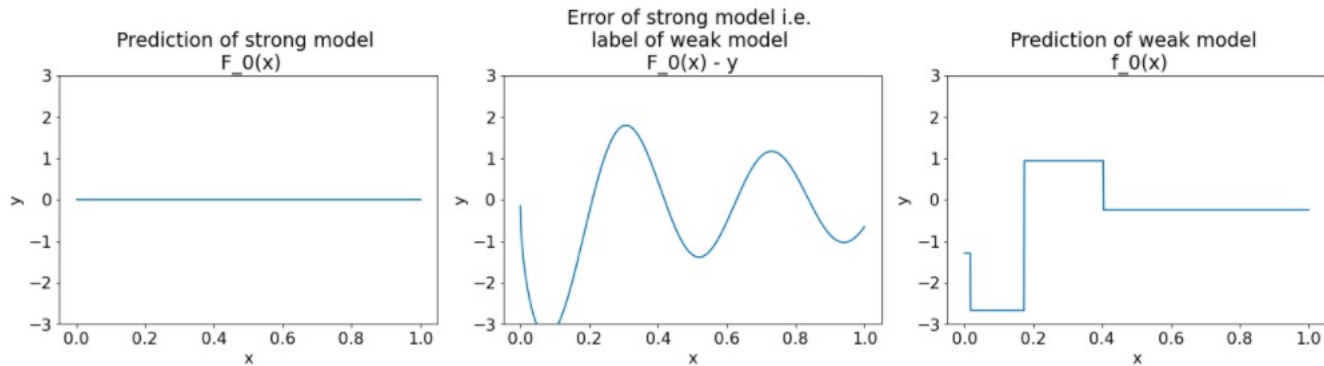


Пример работы градиентного бустинга

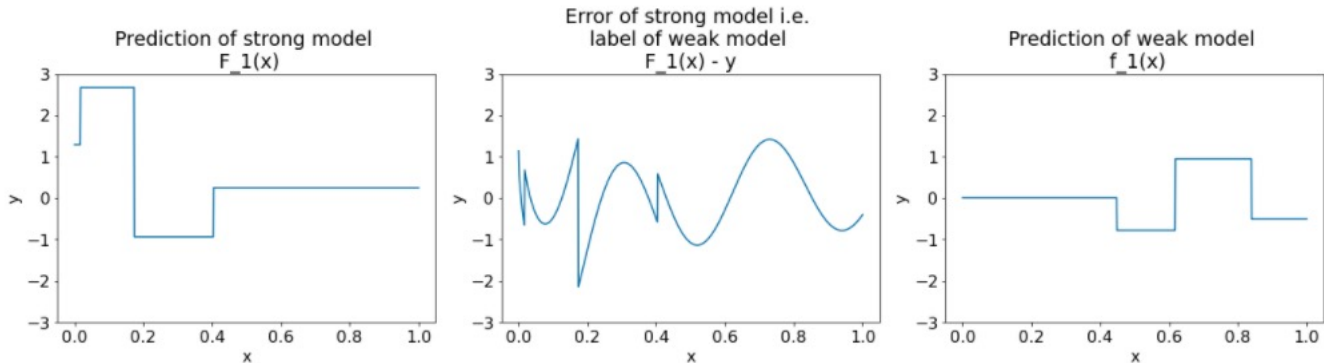


Пример работы градиентного бустинга

Первый шаг

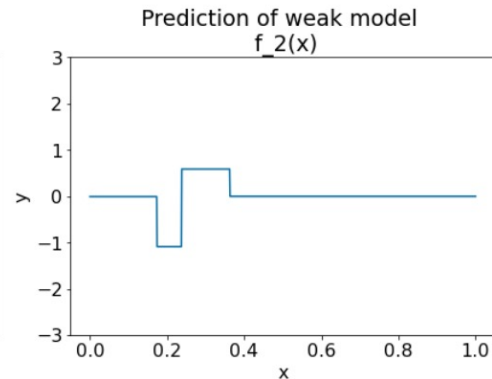
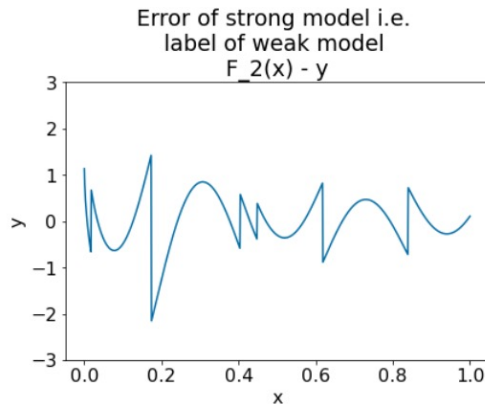
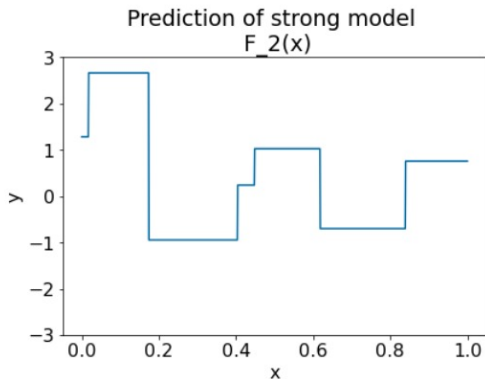


Второй шаг



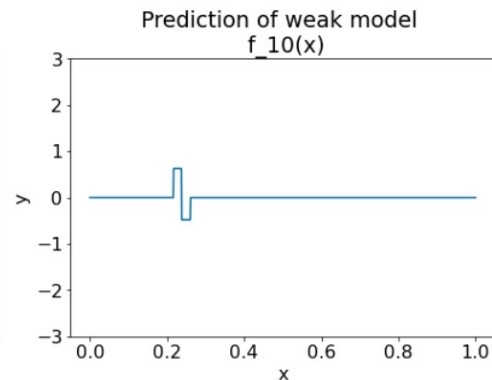
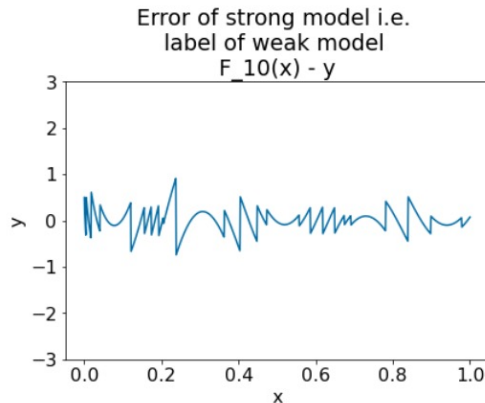
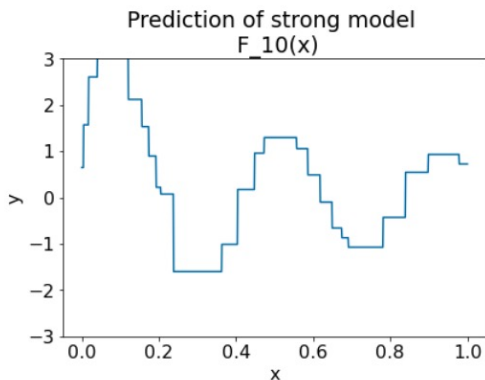
Пример работы градиентного бустинга

Третий шаг



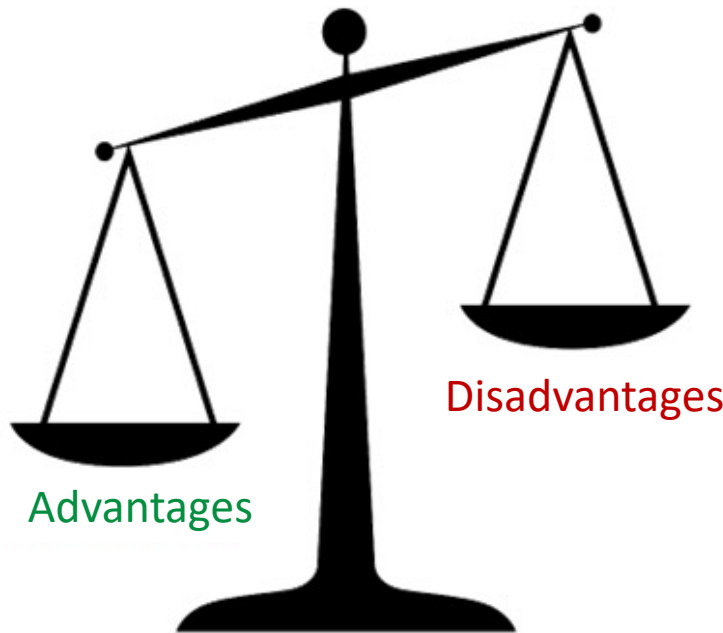
...

11-й шаг



AdaBoost – достоинства и недостатки

- Эффективный с вычислительной точки зрения
- Позволяет решать сложные задачи, которые плохо решаются отдельными алгоритмами
- Простой с точки зрения программирования Только один параметр настройки - число итераций
- Не требует априорной информации о слабом классификаторе
- Обеспечивает во многих случаях высокую точность прогнозирования
- Прост для модификаций



- Слишком эффективные или сложные классификаторы могут привести к переобучению
 - Чрезмерная чувствительность к выбросам
 - Громоздкие композиции из сотен алгоритмов не интерпретируемы
 - Требуются достаточно большие обучающие выборки
- Не удастся строить короткие композиции из «сильных» алгоритмов типа SVM (только длинные из слабых)
- Слишком “слабые” слабые классификаторы плохо работают

Вопросы?



Ставим “+”,
если вопросы есть



Ставим “-”,
если вопросов нет

Современные реализации

Реализации бустинга



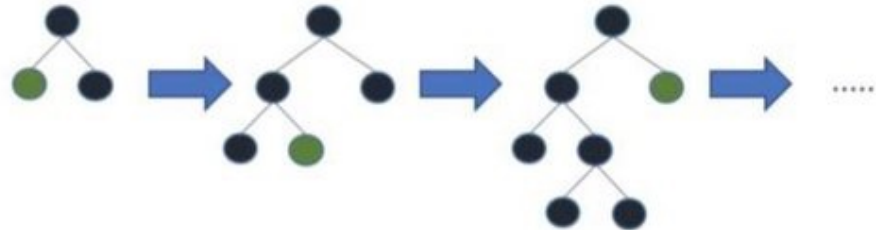
CatBoost

XGBoost & LightGBM

dmlc
XGBoost



 **LightGBM**



Практика

Вопросы?



Ставим “+”,
если вопросы есть



Ставим “-”,
если вопросов нет

Рефлексия

**Заполните, пожалуйста,
опрос о занятии
по ссылке в чате**

Спасибо за внимание!

Приходите на следующие вебинары

25.03 – Метод опорных векторов

Игорь Стурейко

Руководитель курсов: Reinforcement Learning, ML Professional, ML Basic

Teamlead, главный инженер проекта,
Физический факультет МГУ, PhD теоретическая физика

Опыт:

Более 15 лет занимался прикладной математикой и мат моделированием (Data Scientist) (Python, C++) в НИИ ПАО Газпром

Анализ временных рядов, эволюционные модели, финансовые модели

@stureiko (TG)

LinkedIn: [igor-stureiko](#)

