

# Композиции моделей и Градиентный Бустинг

Лекция 10

# Мотивация

хgb, catb, boost  
устойчивость

RF

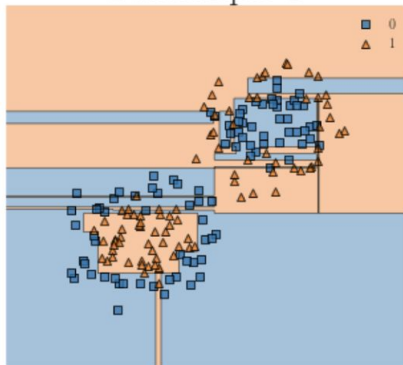
boost  
DT  $\nearrow$  strength  $\uparrow$  bias  $\downarrow$

DDT strength  $\downarrow$  bias  $\uparrow$   
bagging

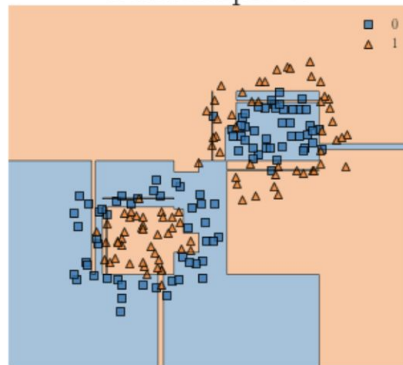
Subsample 0



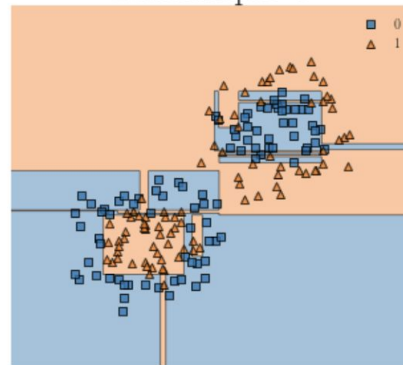
Subsample 1



Subsample 2



Subsample 3

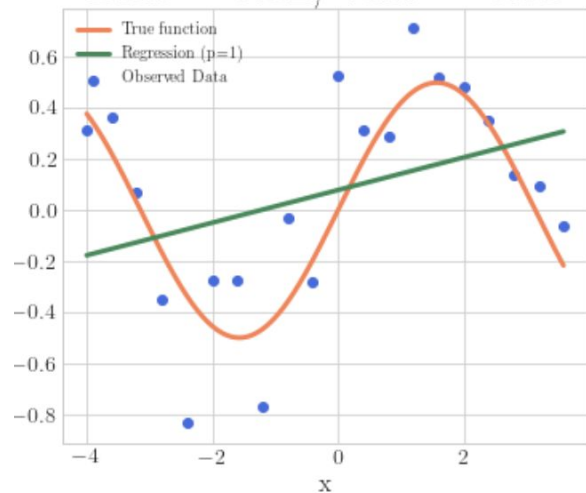


# Разложение Ошибки на Смещение и Разброс

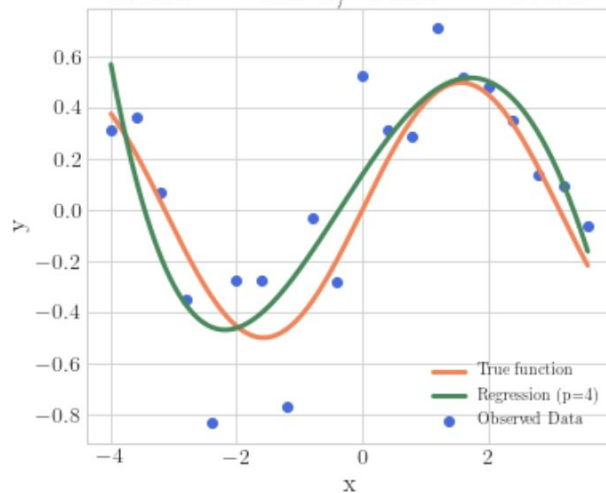
$$\begin{aligned} L(\mu) = & \underbrace{\mathbb{E}_{x,y} \left[ (y - \mathbb{E}[y | x])^2 \right]}_{\text{шум}} + \\ & + \underbrace{\mathbb{E}_x \left[ (\mathbb{E}_X [\mu(X)] - \mathbb{E}[y | x])^2 \right]}_{\text{смещение}} + \underbrace{\mathbb{E}_x \left[ \mathbb{E}_X \left[ (\mu(X) - \mathbb{E}_X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

# Смещение и Разброс для Линейной Регрессии

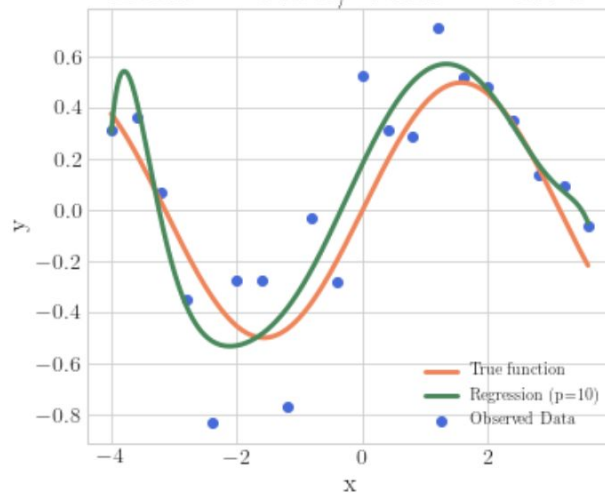
Bias = 0.12, Var. = 0.01



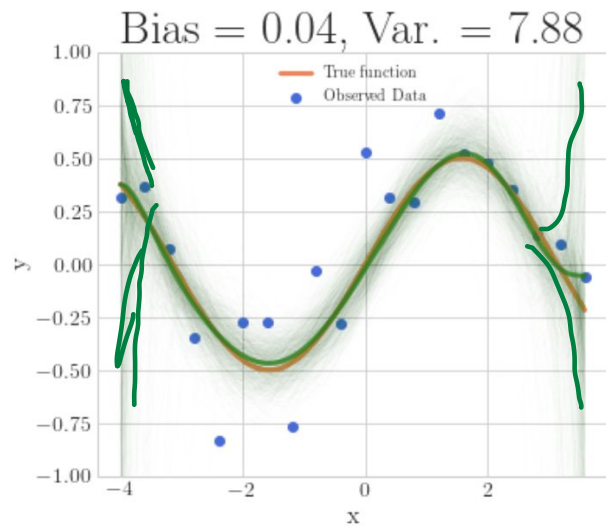
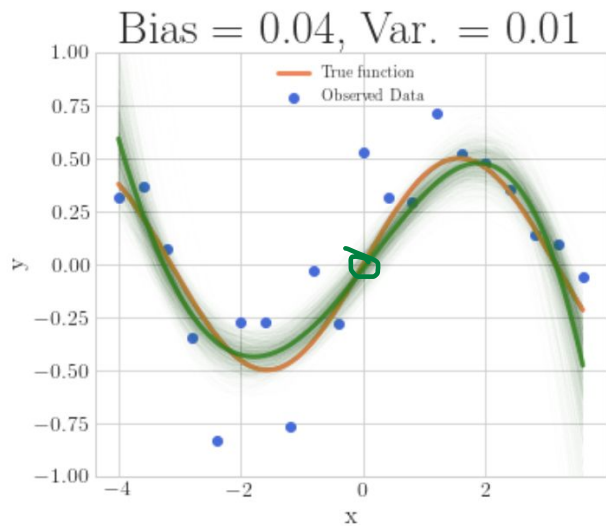
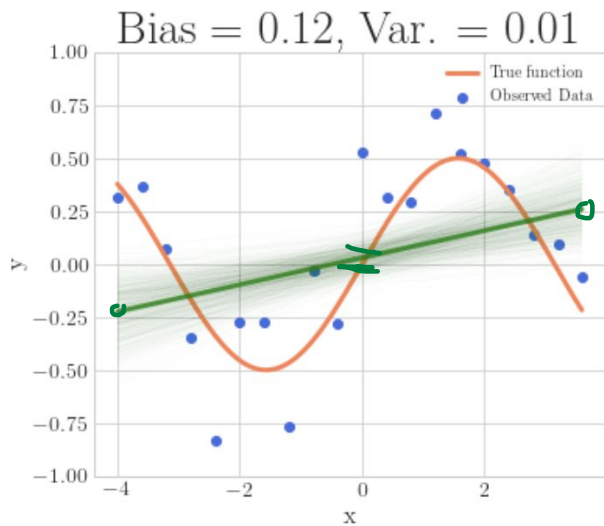
Bias = 0.04, Var. = 0.01



Bias = 0.04, Var. = 4.73



# Смещение и Разброс для Линейной Регрессии



# Композиции Моделей

- Bagging

- Boosting

- Stacking

- Blending

- ...



# Композиции Моделей

- Voting
- Bagging
- Boosting
- Stacking
- Blending
- ...

# Композиции Моделей

same (bag boost)  
→ random (Voting, Stacking)

- Давайте обучим  $M$  базовых алгоритмов

$$b_1(x), \dots, b_M(x)$$

- И объединим их в композицию

$$a_M(x) = f(b_1(x), \dots, b_M(x))$$

- Например, усреднив их предсказания

$$a_M(x) = \frac{1}{M} \sum_{m=1}^M b_m(x)$$

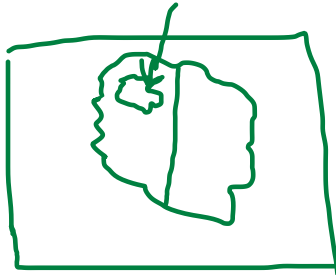


# Разнообразие Ансамблей

- Все базовые алгоритмы из одного семейства?
- Базовые алгоритмы учатся независимо?
- Как делать композицию?

усреднения

(AM) GM





Bagging V

Boosting X

Voting V

Stacking X

# Беггинг (Bagging)

- Все базовые алгоритмы из одного семейства?
  - Да. Обучаем базовые алгоритмы на подвыборках, полученных бутстрапом
- Базовые алгоритмы учатся независимо?
  - Да, можно учить параллельно
- Как делать композицию?
  - Усреднение для регрессии 
  - Majority Voting для классификации 

# Беггинг (Bagging)

Bootstrap

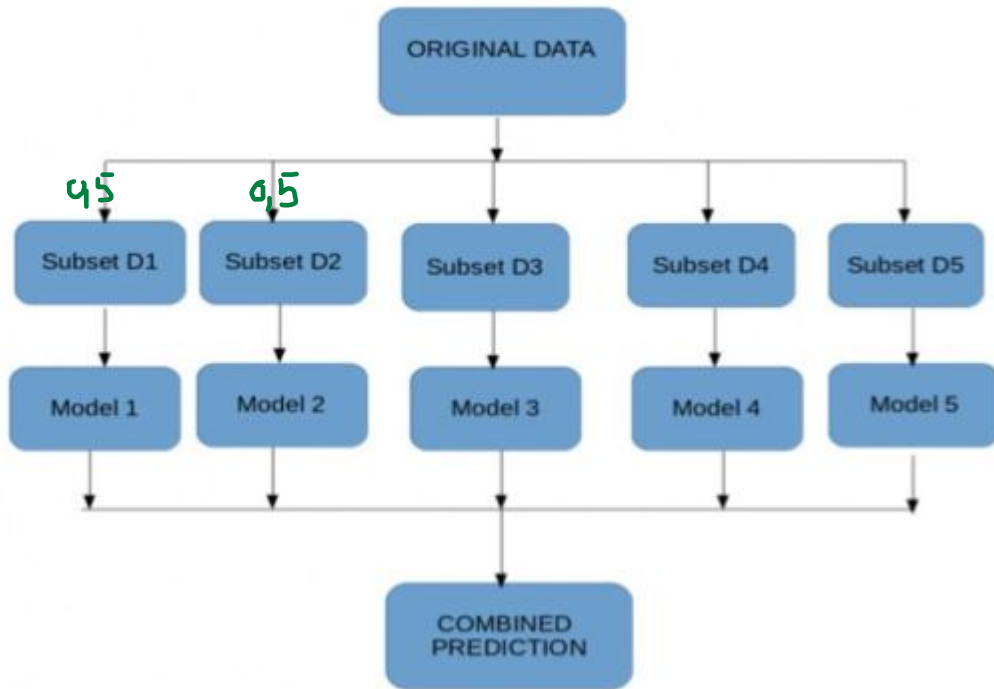
33%



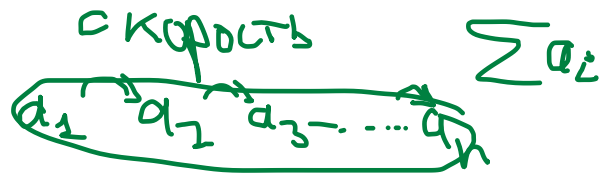
Bootstrap True

sub True

batch

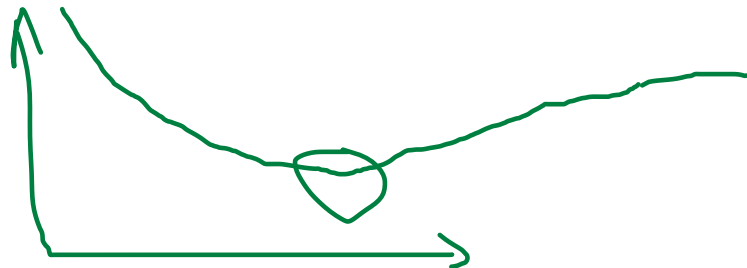
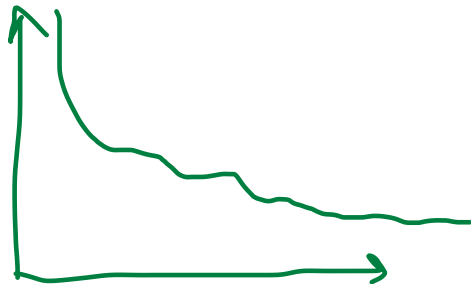


# Бустинг (Boosting)



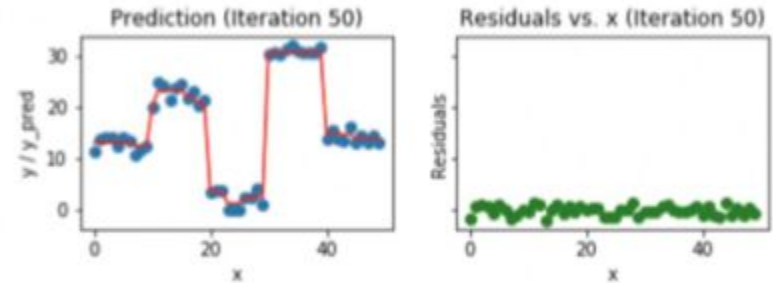
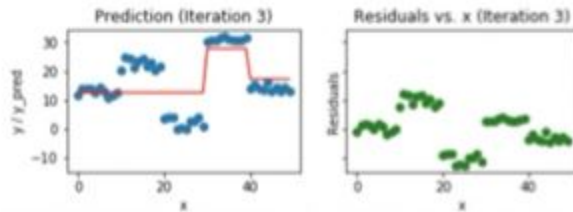
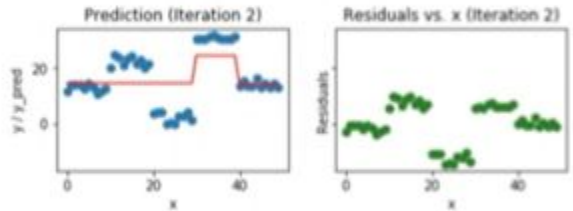
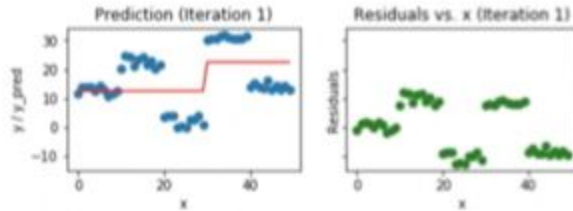
- Все базовые алгоритмы из одного семейства?
  - Да. Часто также использую подвыборку
- Базовые алгоритмы учатся независимо?
  - Нет, каждый следующий исправляет ошибки предыдущего
- Как делать композицию?
  - Усреднение для регрессии
  - Majority Voting для классификации

$f_n$



# Бустинг (Boosting)

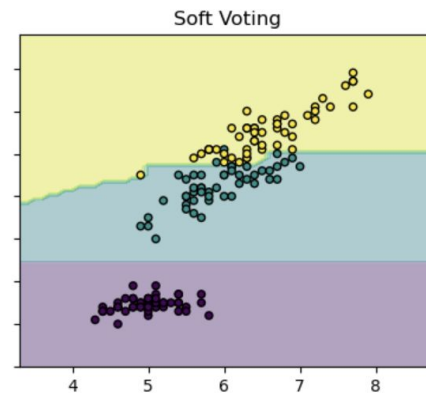
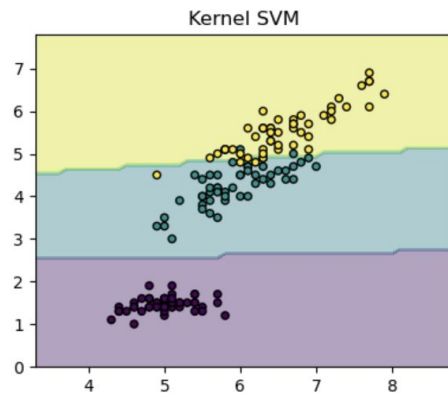
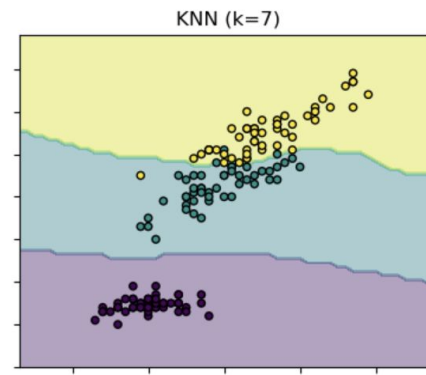
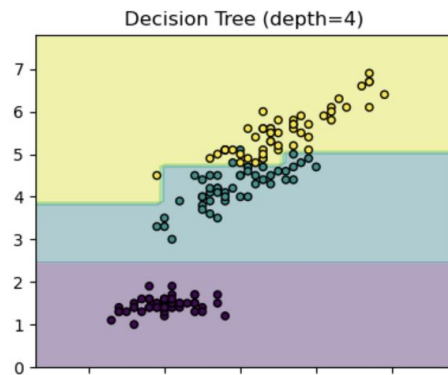
~~$\frac{1}{2}$~~   $\rightarrow (y - a(x))$   
 $(y - \sigma_1 x)$   
 $(a_1 + a_2)$



# Вотинг (Voting)

- Все базовые алгоритмы из одного семейства?
  - Нет, можно использовать разные алгоритмы
- Базовые алгоритмы учатся независимо?
  - Да, можно учить параллельно
- Как делать композицию?
  - Усреднение для регрессии
  - Majority Voting для классификации

# Вотинг (Voting)



$$z = w_1 \underline{b_1(x)} + w_2 \underline{b_2(x)}$$

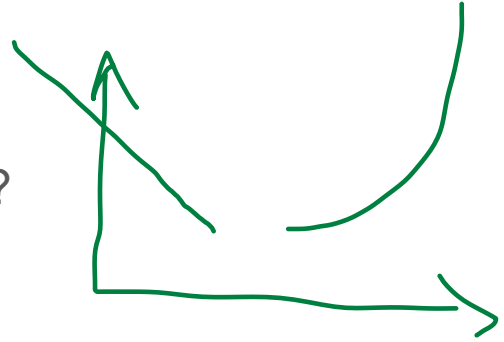
$$w_1(x)$$

$$w_2(x)$$

# Стекинг (Stacking)

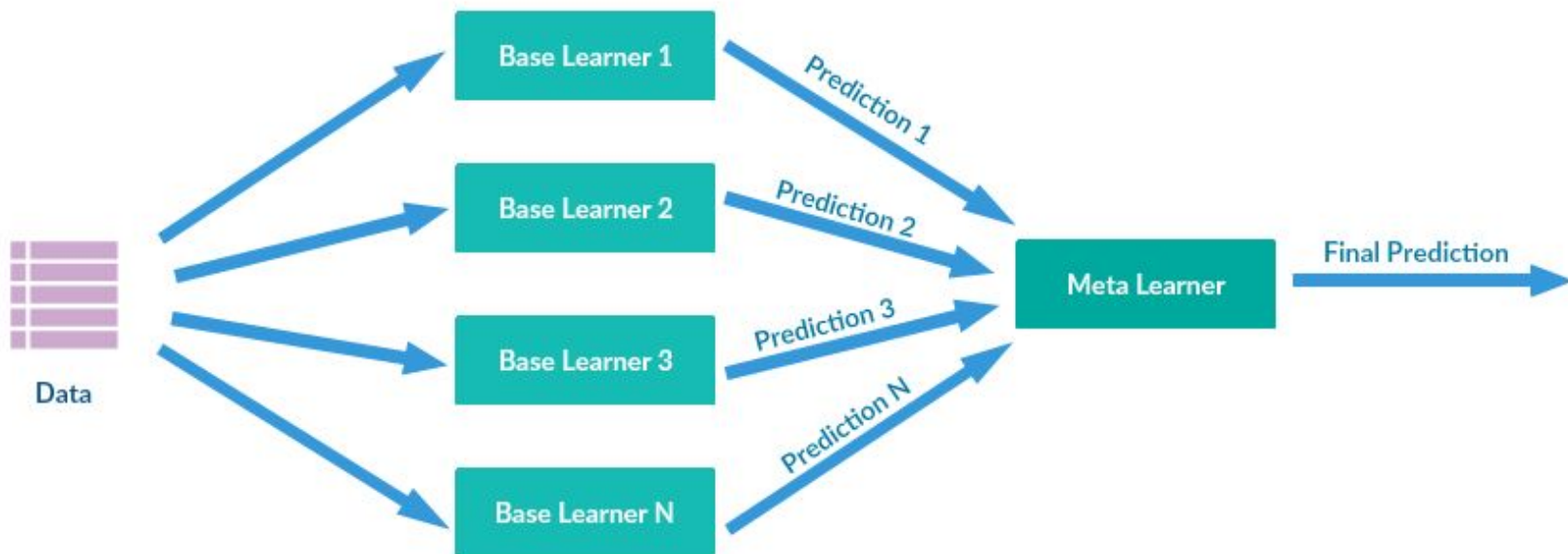
- Все базовые алгоритмы из одного семейства?
  - Нет, можно использовать разные алгоритмы
- Базовые алгоритмы учатся независимо?
  - Да, можно учить параллельно
- Как делать композицию?
  - Делаем предсказание на валидации
  - Обучаем мета-модель на этих предсказаниях

- К Экспертам
- учим мета-модель на ответах





# Стекинг (Stacking)



# Градиентный бустинг

$$a_N(x) = \sum_{n=0}^N \gamma_n b_n(x).$$

- Учим базовые алгоритмы последовательно, исправляя ошибки предыдущих

$$\sum_{i=1}^{\ell} L(y_i, a_{N-1}(x_i) + \gamma_N b_N(x_i)) \rightarrow \min_{b_N, \gamma_N}$$

- Т.е. Каждый следующие алгоритм приближет антиградиент функционала ошибки

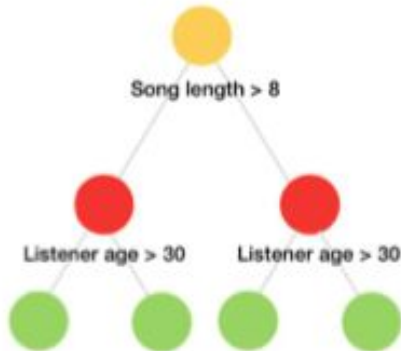
$$s_i = - \left. \frac{\partial L(y_i, z)}{\partial z} \right|_{z=a_{N-1}(x_i)}$$

# Вариации бустинга

- XGBoost
  - Стандарт до конца 2016 года.
  - Оптимизированность построения деревьев
  - Различные регуляризации модели
- LightGBM
  - Быстрота построения композиции.
  - Быстрота обучения
- CatBoost
  - Библиотека от компании Яндекс.
  - Позволяет автоматически обрабатывать категориальные признаки
  - Менее чувствительным к выбору конкретных гиперпараметров

# ODT

- Oblivious decision trees
- Ограничение: на одном уровне дерева используется один и тот же предикат

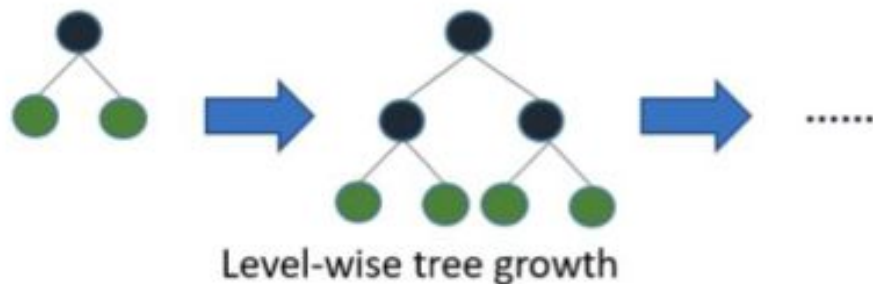


$$2^{K-1}$$

<https://catboost.ai/>

# Способ построения дерева

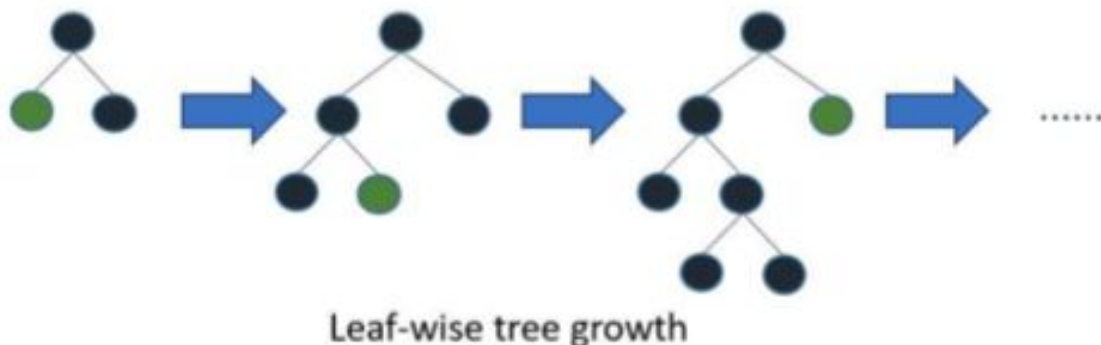
- Level-wise: дерево строится рекурсивно до тех пор, пока не достигнута максимальная глубина



<https://lightgbm.readthedocs.io/>

# Способ построения дерева

- Level-wise: дерево строится рекурсивно до тех пор, пока не достигнута максимальная глубина
- Leaf-wise: среди текущих листьев выбирается тот, чьё разбиение сильнее всего уменьшает ошибку




# Выбор лучшего порога для предиката

- $[x_j < t]$  — как выбрать  $t$ ?
- Вариант 1: перебрать все известные значения признака
- Вариант 2: построить гистограмму для признака и искать пороги среди границ на гистограмме
- Вариант 3: случайно выбрать объекты с близкими к нулю значениями производной функции потерь



# Регуляризация деревьев

- Базовая регуляризация: введение длины шага и количества выбираемых признаков
- Штрафы за число листьев в дереве 
- Штрафы за величину прогнозов в листьях дерева



# XGBoost

- Распределенное обучение
- Регуляризация

$$\Omega(f) = \gamma T + \frac{1}{2} \lambda \sum_{j=1}^T w_j^2$$

- Прунинг