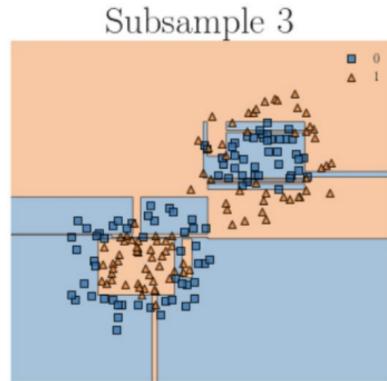
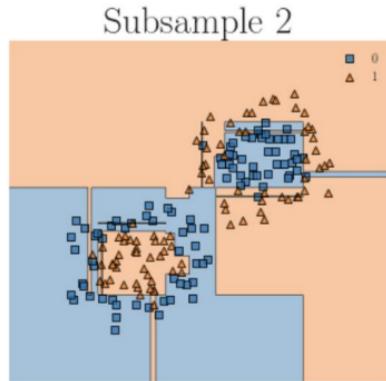
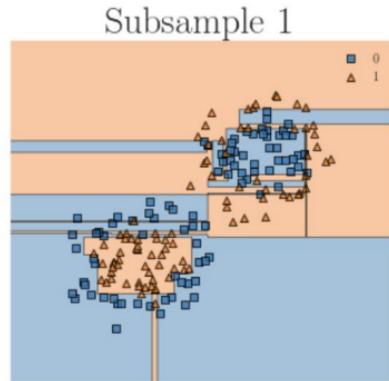
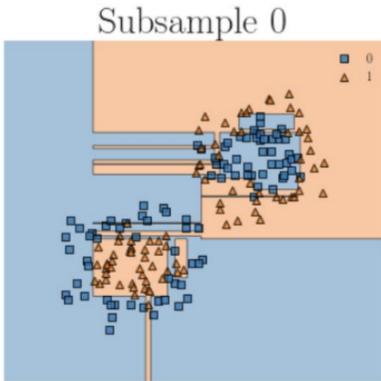


# Композиции моделей и Градиентный Бустинг

Лекция 10

# Мотивация

- Бокоме усвоимбом (the other way)
  - салынан ↓
    - бокоме жүзбөрө => baggy
  - Depicto => тұжанды  $\Rightarrow$  (3 ≥)
    - бокоме жүзбөрө
  - (6 наложе nose & oral)



- Deepbo ⇒ ve riyade ⇒ • weyne ↑ => Boost  
 (2-3)  
 (→ ←)

Bagging limits part of variance  
Boosting limits average bias

# Разложение Ошибки на Смещение и Разброс

$$L(\mu) = \mathbb{E}_{x,y} \left[ (y - \mathbb{E}[y|x])^2 \right] + \mathbb{E}_x \left[ (\mathbb{E}_X[\mu(X)] - \mathbb{E}[y|x])^2 \right] + \mathbb{E}_x \left[ \mathbb{E}_X[(\mu(X) - \mathbb{E}_X[\mu(X)])^2] \right]$$

шум

смещение

разброс "variance"

алгоритм  $\rho(x)\rho(y|x)$

наблюдаем

не можем снизить эту часть

формула для биаса.

// биасом.

алгоритм биасом подборка

идеальный ответ про  $y$  но  $x$

"bias"

Boosting

алгоритм  $A_1$  bias variance 1

алгоритм  $A_2$  bias variance 1

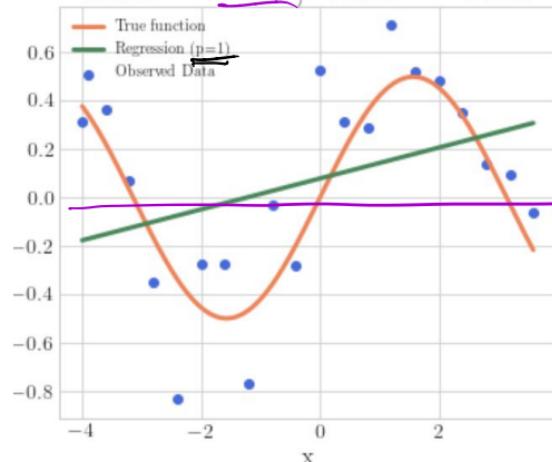
Bagging

$$a(x) = \underbrace{w_0 + w_1 x^1}_{x^0} + w_2 x^2 + \dots + w_p x^p$$

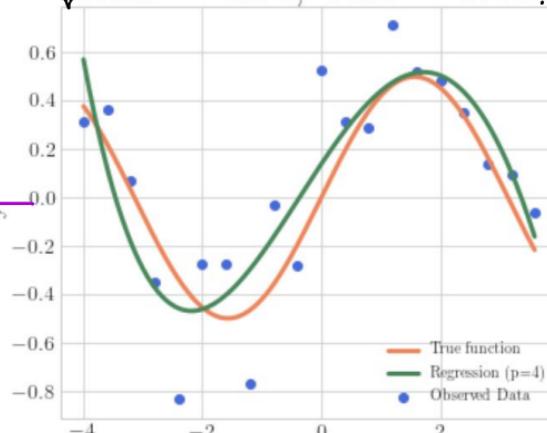
## Смещение и Разброс для Линейной Регрессии

$$p=0 \quad a(x) = w_0$$

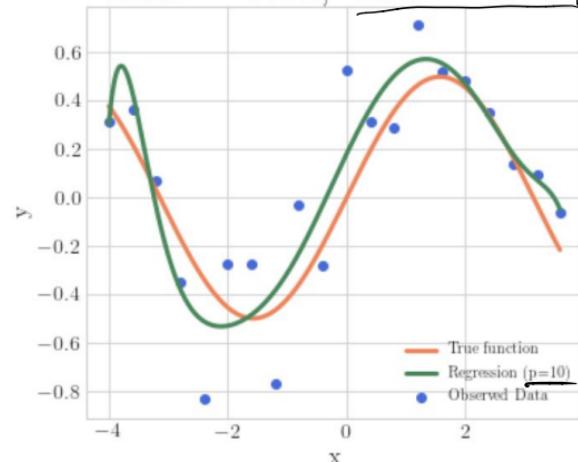
$$\text{Bias} = 0.12, \text{Var.} = 0.01$$



$$\downarrow \text{Bias} = 0.04, \text{Var.} = 0.01?$$



$$\text{Bias} = 0.04, \text{Var.} = 4.73$$

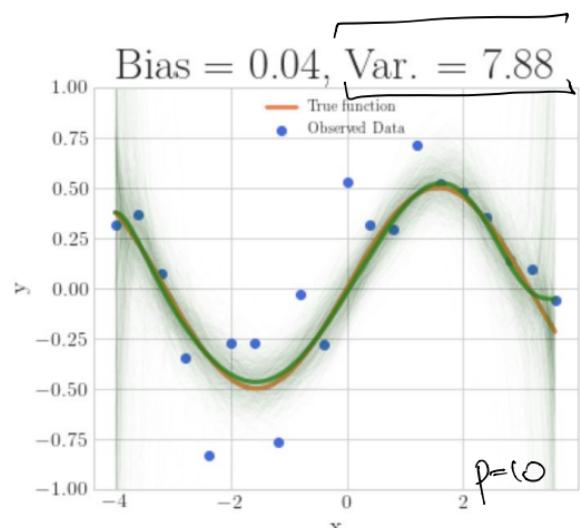
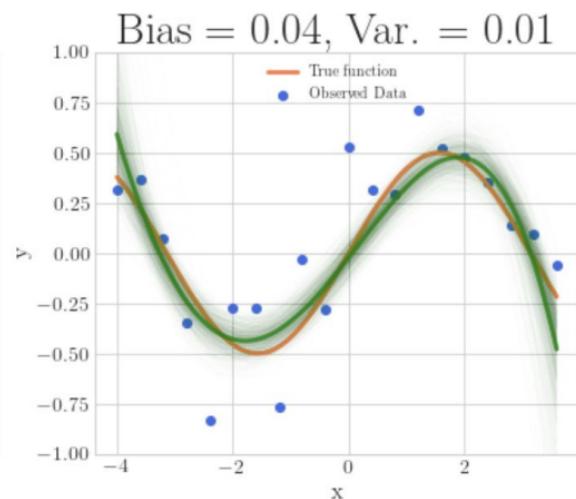
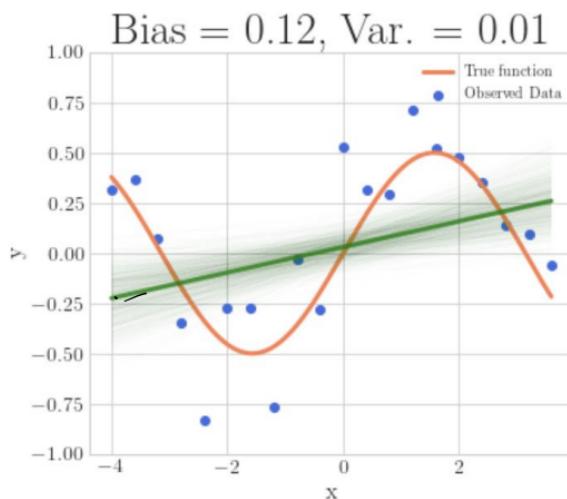


$\text{Bias} \uparrow > 0.12$

$$\begin{aligned} & \Rightarrow (w_0 + w_1 x^1) \\ & + (\tilde{w}_0 + \tilde{w}_1 x^1) \\ & = (w_0 + \tilde{w}_0) + (w_1 + \tilde{w}_1)x_1 \end{aligned}$$

малое смещение не будет.  
на извёрнутых кривых так и даёт)

# Смещение и Разброс для Линейной Регрессии



# Композиции Моделей

- Bagging

- Boosting

- Stacking

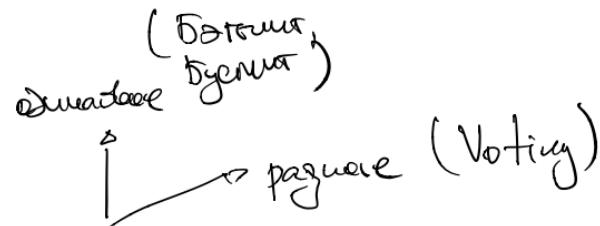
- Blending

- ...

# Композиции Моделей

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

# Композиции Моделей



- Давайте обучим M базовых алгоритмов

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

- И объединим их в композицию

как объединять?

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

- 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?

последовательно || параллельно  
↑

- Базовые алгоритмы учатся независимо?

≈ Boosting ✓

- Как делать композицию?

≈ Stacking

Скорость?

≈ Boosting ✗

✗

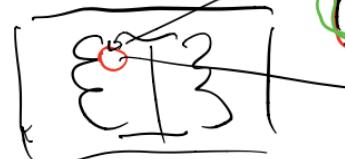
$$a_2 \{ a_1(x) \}$$

уравнение

✓ АМ, ГМ  
• классификация  
регрессия  
(все на свое время)

$$\left[ \prod_{n=1}^N a_n(x) \right]^{\frac{1}{N}} \Rightarrow$$

сегментация



# Бэггинг (Bagging)

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

# Баггинг (Bagging)

Баггинг (базова с повторениями, подвергает чисте нейтрал.)  $\Rightarrow$  здруж се къмъждане.

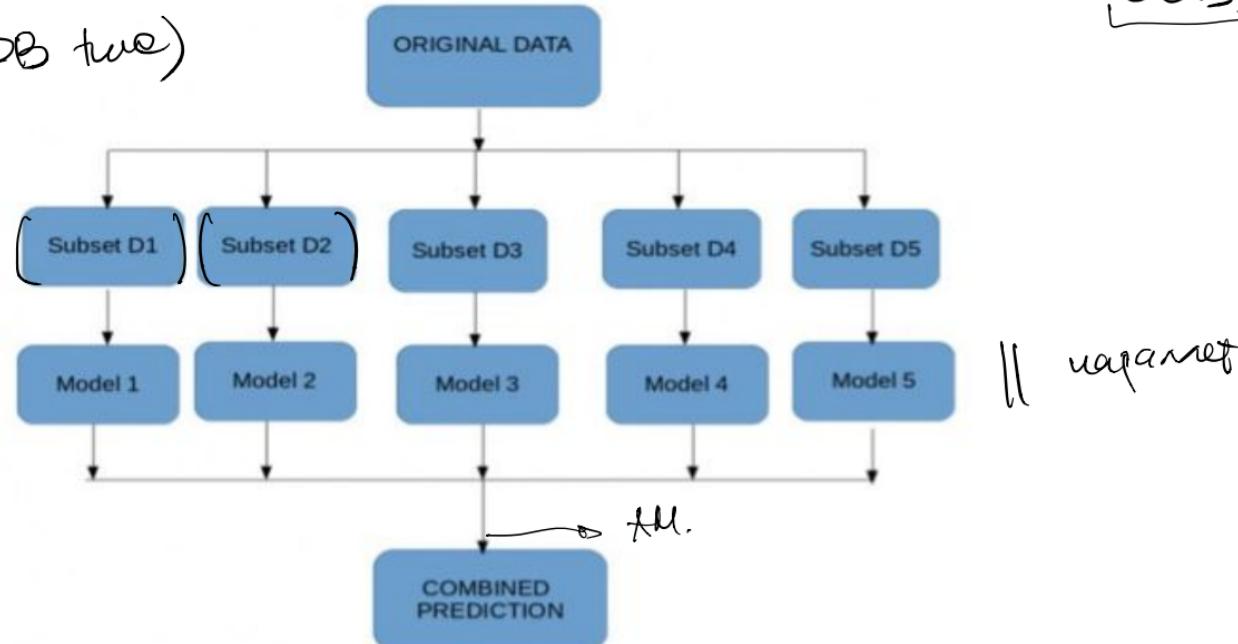
↓  
↓

OOB

- sklearn (OOB true)

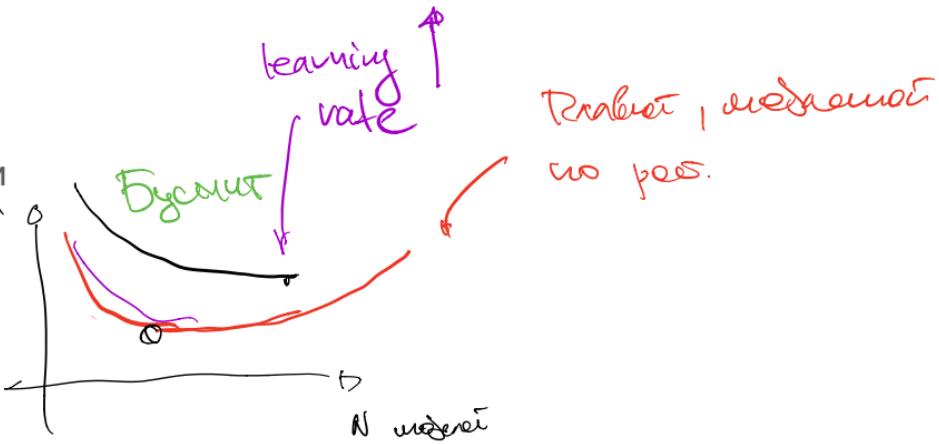
bootstrap true

- Если не bootstrap,  
то предсказание.



# Бустинг (Boosting)

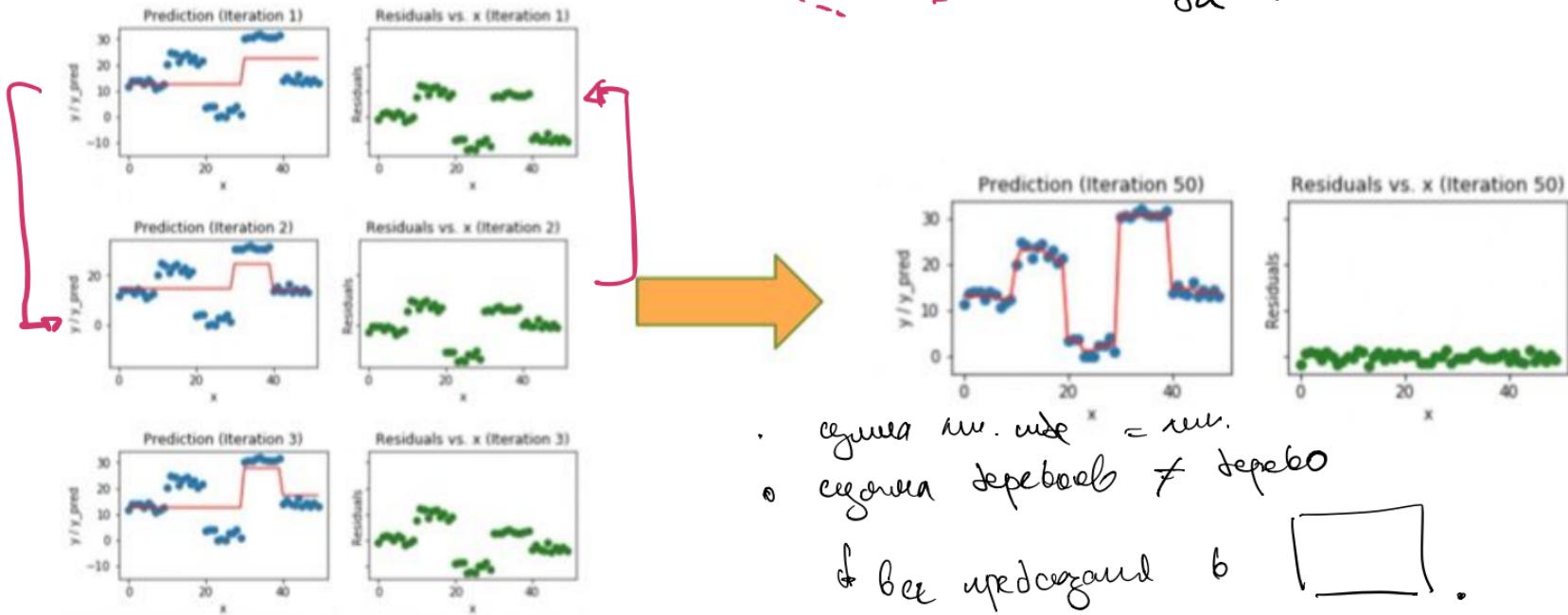
- Все базовые алгоритмы из одного семейства?
  - Да. Часто также использую подвыборку (и то ~~объемом~~, и то ~~изделием~~).
- Базовые алгоритмы учатся независимо? ( последовательно)
  - Нет, каждый следующий исправляет ошибки предыдущего
- Как делать композицию?
  - Усреднение для регрессии
  - Majority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



# Бустинг (Boosting)

$$\text{MSE} \Rightarrow (y_n - a_i(x_i))^2; L \Rightarrow L(y_n, a(x_i))$$

→ linear  $\frac{\partial L}{\partial a}$



# Вотинг (Voting)

голосование  
фазы  
Mixed of Experts

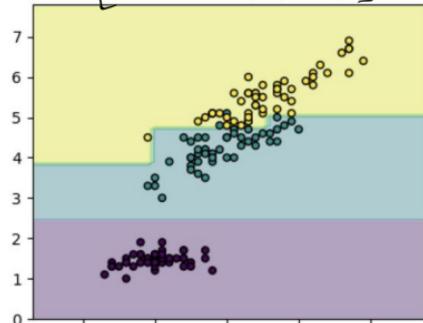
- Все базовые алгоритмы из одного семейства?
  - Нет, можно использовать разные алгоритмы
- Базовые алгоритмы учатся независимо?
  - Да, можно учить параллельно
- Как делать композицию?
  - Усреднение для регрессии
  - Majority Voting для классификации

# Вотинг (Voting)

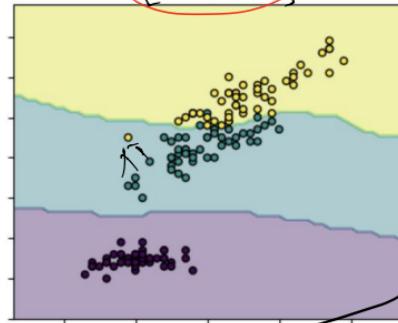
- можно на разных методах
- на разных структурах
- ...

самые  
модели дают хорошие  
результаты

Decision Tree (depth=4)



KNN (k=7)

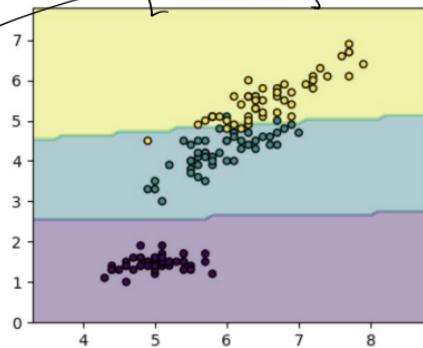


алгебраиче-

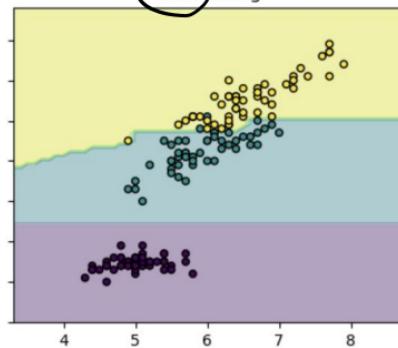
Soft (Hard  
probability) voted

Kernel SVM

poly



Soft Voting



# Стекинг (Stacking)

исследовательский  
направленный на обобщение

$$a(x) = w_0 + w_1 b^1(x) + w_2 b^2(x)$$

да • (бустинг), RF.

- Все базовые алгоритмы из одного семейства?

- можно много Нет, можно использовать разные алгоритмы  
- очень не нужно брать.

- Базовые алгоритмы учатся независимо?

- Да, можно учить параллельно

## Как делать композицию?

- Делаем предсказание на валидации
- Обучаем мета-модель на этих предсказаниях

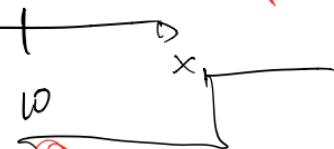
1. Учим к "экспертам" (базовым алгоритмам)  $\Rightarrow$  получаем их ответы

2. Учим модель на ошибках  $y - \hat{y}$  шаг 1.

[2]  $\rightarrow b_1(x), b_2(x), b_3(x)$  |  $y$   
мета-модель.

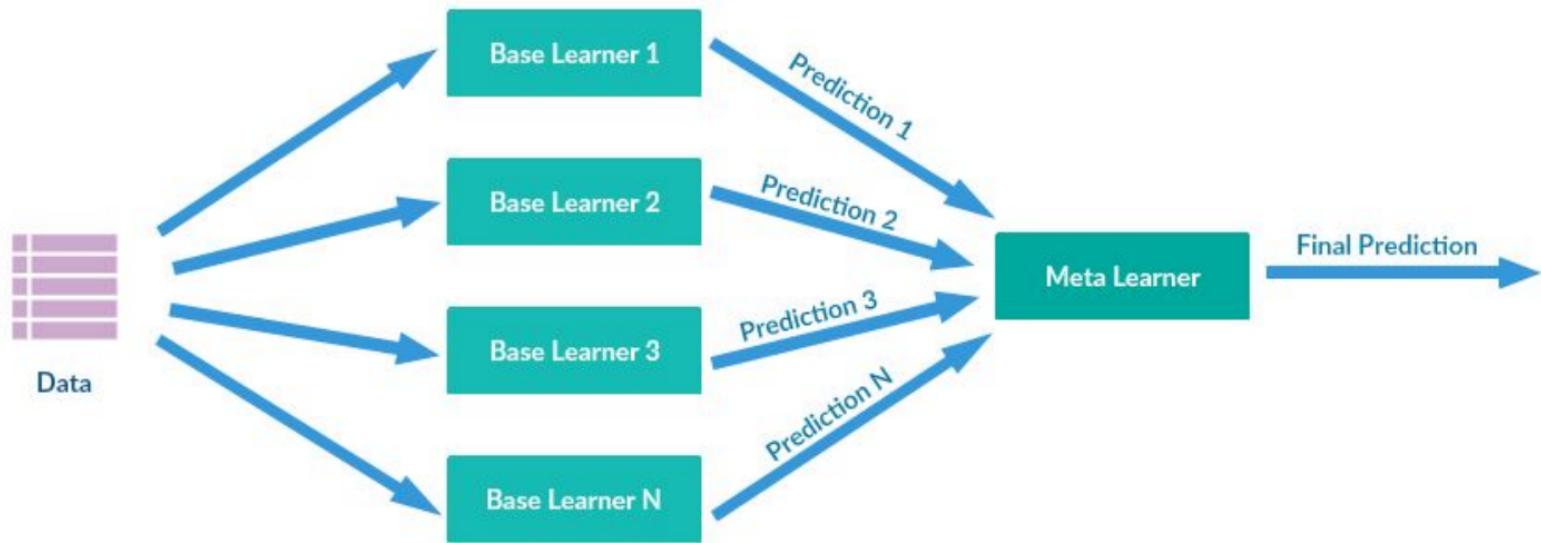
$$a(x) = w_0 + \underline{w_1}(x) b^1(x) + \underline{w_2}(x) b^2(x)$$

Boosting



4. Итоговая схема:  
короткие пути:  $f(W_i)$   
1. учимся в толще  
2. получив в ответ  
одинаковую модель.

# 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 следующий алгоритм приближет антиградиент функционала ошибки

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

регуляризации и гиперпараметров  
автор Борис Буслаев

RF (базы)

## Вариации бустинга

### XGBoost (чуть менее удобно)

- ✓ XGBoost
  - Стандарт до конца 2016 года.
  - Оптимизированность построения деревьев
  - Различные регуляризации модели

+ сложнее из-за избыточности

(избыточна)  
чтобы сократить

### LightGBM (чуть проще)

- LightGBM
  - Быстрота построения композиции.
  - ✓ Быстрота обучения

(даже шаг, характеристики.)

одинаково XGBoost  
LightGBM → упрощение.

### CatBoost → обходит много проблем

- CatBoost
  - Библиотека от компании Яндекс.
  - Позволяет автоматически обрабатывать категориальные признаки
  - Менее чувствительным к выбору конкретных гиперпараметров

mean embedding

ODT

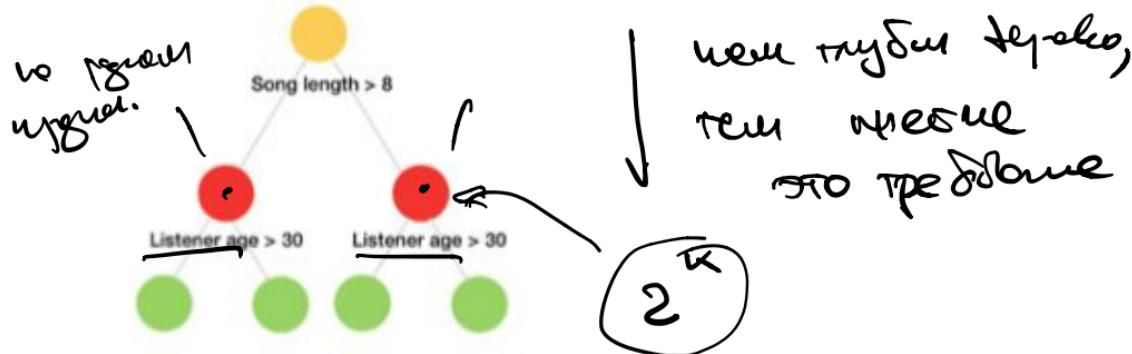
дерево решений  
decision tree

Базовый алгоритм в библиотеке catboost (баз),  
намного от баз. алг.  
меньше ошибок

- Oblivious decision trees

"забывчив" → регуляризация дерева.

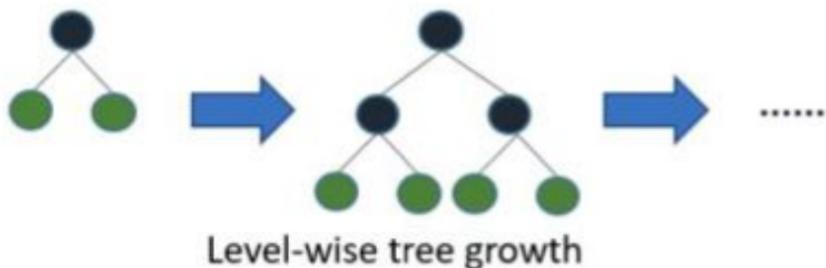
- Ограничение: на одном уровне дерева используется один и тот же предикат



<https://catboost.ai/>

# Способ построения дерева

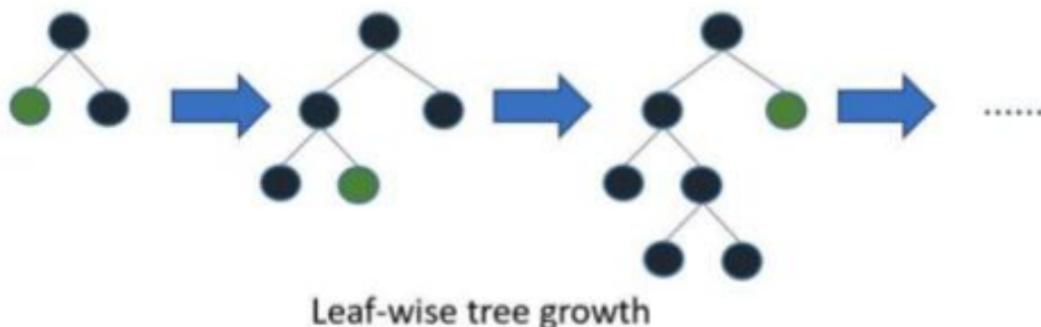
- Level-wise: дерево строится рекурсивно до тех пор, пока не достигнута максимальная глубина



<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

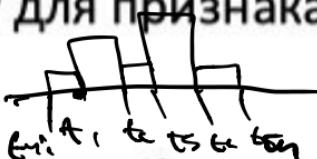


# Выбор лучшего порога для предиката

- $[x_j < t]$  — как выбрать  $t$ ?

~~Все значения~~  
✗ 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

$\Rightarrow$  малые  $\downarrow$ , много  $\uparrow$

$\Rightarrow$  малое  $\downarrow$   
у границ классов

$$(y - \hat{y}_w)^T (y - \hat{y}_w) + \lambda \|w\|_2^2$$

## + Регуляризация деревьев

- Базовая регуляризация: введение длины шага и количество выбираемых признаков

- Штрафы за число листьев в дереве  $\Rightarrow$  штраф за тупицу.
- Штрафы за величину прогнозов в листьях дерева

сокращает сжатия

1) `tree-10` работает

2) те acción хороши  $\rightarrow$  не это  
не хороший эффект  $\rightarrow$  не это  
нужно умирать

# XGBoost

- Распределенное обучение
- Регуляризация

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

- Прунинг

## MEAN EMBEDDING

