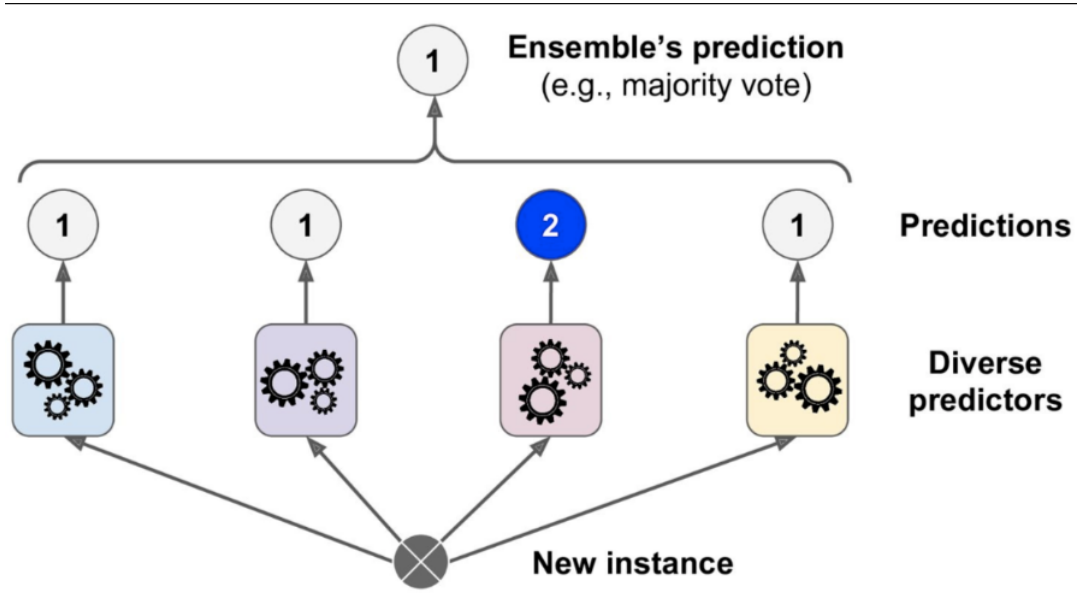
Ансамблирование моделей • Ensembling

Сегодня

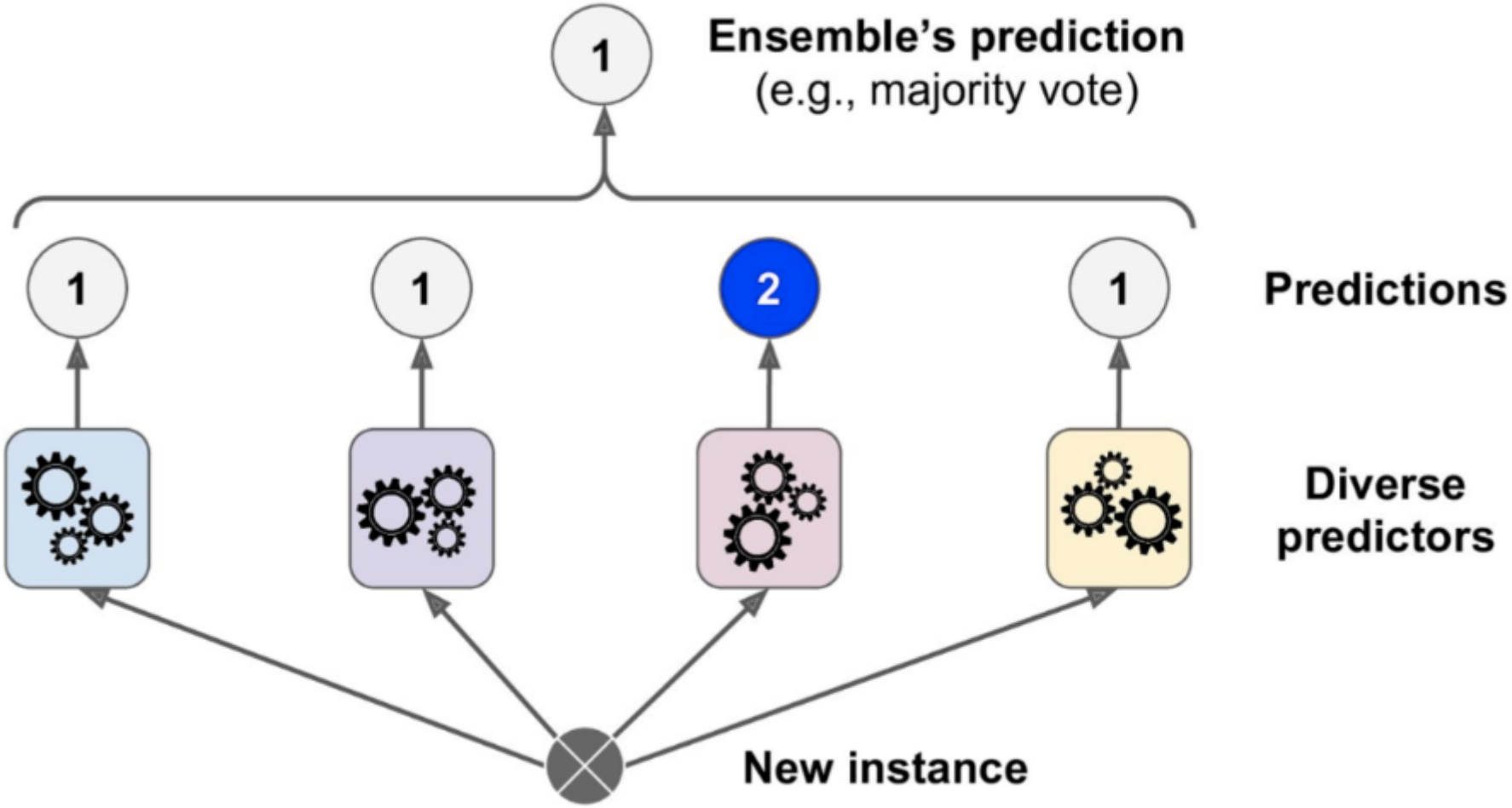
- ответим на вопрос "как можно соедининть несколько моделей?"
- разберем наиболее популярные варианты
- установим и запустим самые мощные библиотеки для работы с ансамблями моделей

Голосование / Voting



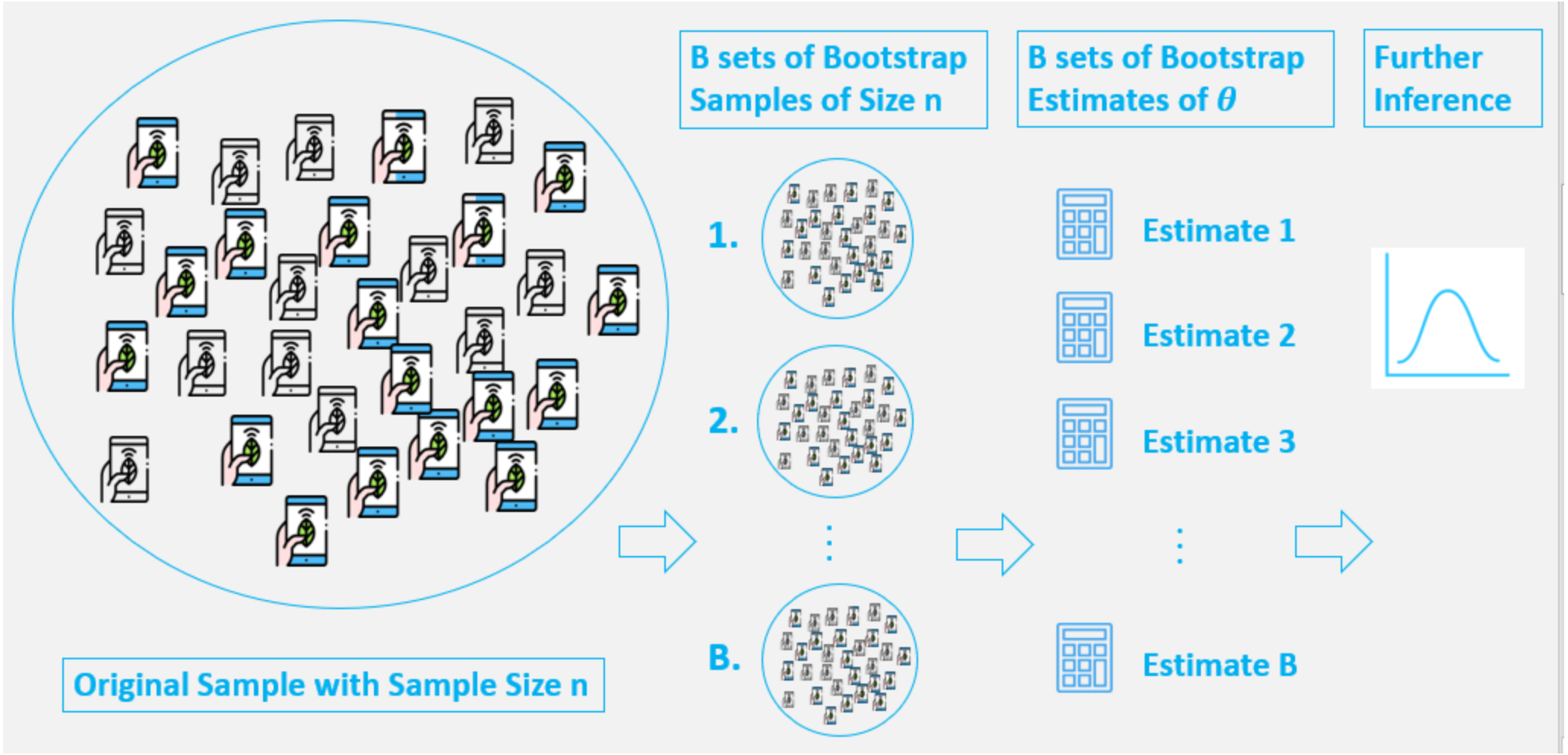
- **Hard voting**: классификаторы предсказывают классы - выбираем наиболее частотный
- **Soft voting**: классификаторы предсказывают вероятности - усредняем вероятности по классам

Голосование / Voting



> Что делать с задачей регрессии?

Бутстреп / Bootstrap



Бэггинг / Bagging

1. Основная идея: формируем *boostrap* выборки
    - Обучаем одну простую модель на каждой выборке
2. Агрегируем предсказания всех моделей

Случайный лес / Random Forest

1. Формируем N boostrap-выборок из исходного датасета
2. Обучаем **одно** дерево на каждой из N выборок
3. Для предсказания агрегируем результаты каждого дерева: наиболее частотный класс в случае классификации, усредненное значение в случае регрессии:

- Классификация - самый частотный класс
- Регрессия - усредняем предсказания

Параметры

- число деревьев: `n_estimators`
- + все те же, что для деревьев

`sklearn.ensemble` : scikit-learn.org

Стекинг / Stacking

1. Разделить обучающую выборку на 2 части: `train_1` и `train_2`
2. Выбрать L базовых моделей, одну мета-модель
3. Обучить L базовых моделей на `train_1`
4. Сделать L прогнозов для объектов `train_2`
5. Обучить мета-модель на результатах предсказаний базовых моделей

Стекинг / Stacking

A				
X0	x1	x2	xn	y
0.17	0.25	0.93	0.79	1
0.35	0.61	0.93	0.57	0
0.44	0.59	0.56	0.46	0
0.37	0.43	0.74	0.28	1
0.96	0.07	0.57	0.01	1

B				
X0	x1	x2	xn	y
0.89	0.72	0.50	0.66	0
0.58	0.71	0.92	0.27	1
0.10	0.35	0.27	0.37	0
0.47	0.68	0.30	0.98	0
0.39	0.53	0.59	0.18	1

C				
X0	x1	x2	xn	y
0.29	0.77	0.05	0.09	?
0.38	0.66	0.42	0.91	?
0.72	0.66	0.92	0.11	?
0.70	0.37	0.91	0.17	?
0.59	0.98	0.93	0.65	?

Train algorithm **0** on A and make predictions for B and C and save to **B1**, **C1**
Train algorithm **1** on A and make predictions for B and C and save to **B1**, **C1**
Train algorithm **2** on A and make predictions for B and C and save to **B1**, **C1**

B1			
pred0	pred1	pred2	y
0.24	0.72	0.70	0
0.95	0.25	0.22	1
0.64	0.80	0.96	0
0.89	0.58	0.52	0
0.11	0.20	0.93	1

C1				
pred0	pred1	pred2	y	Preds3
0.50	0.50	0.39	?	0.45
0.62	0.59	0.46	?	0.23
0.22	0.31	0.54	?	0.99
0.90	0.47	0.09	?	0.34
0.20	0.09	0.61	?	0.05

Train algorithm **3** on B1 and make predictions for C1

Стекинг / Stacking

- Нужно много данных
- Нужно много мощностей
- Нужно много времени
- Зато оригинально

`sklearn` : scikit-learn.org

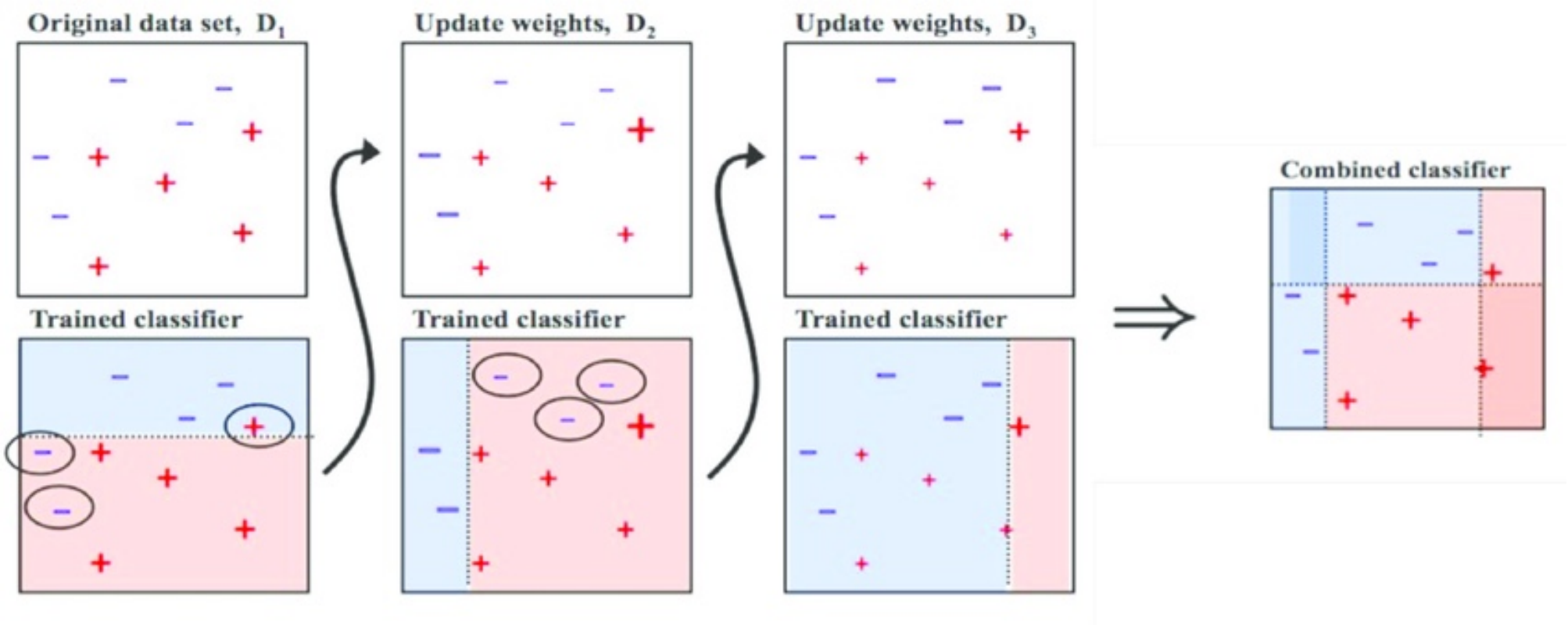
📝Про настройку гиперпараметров ансамблей моделей машинного обучения

Бустинг / Boosting

- Строим алгоритмы последовательно
- Каждый следующий исправляет суммарную ошибку предыдущих
- Решение принимаем взвешенным голосованием

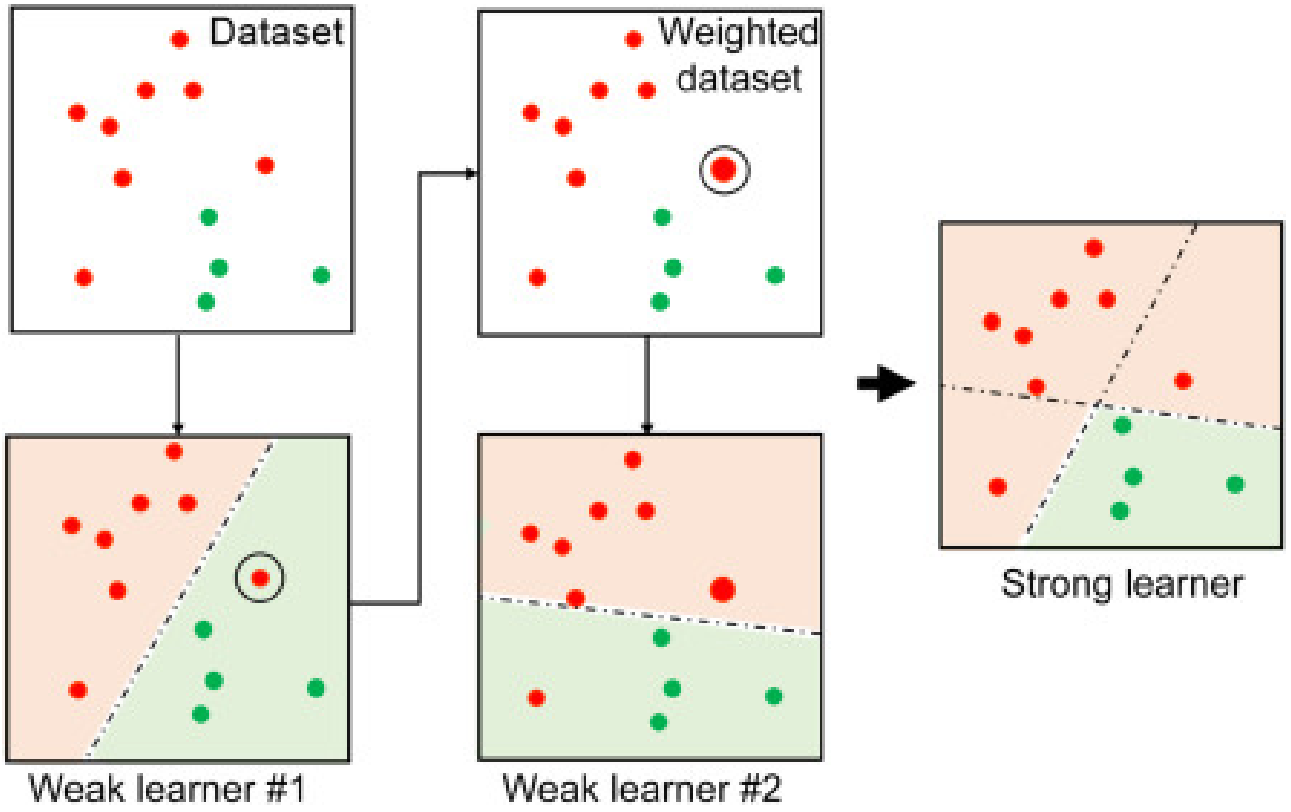
Адаптивный бустинг / AdaBoost

Бустинг на решающих деревьях



Адаптивный бустинг / AdaBoost

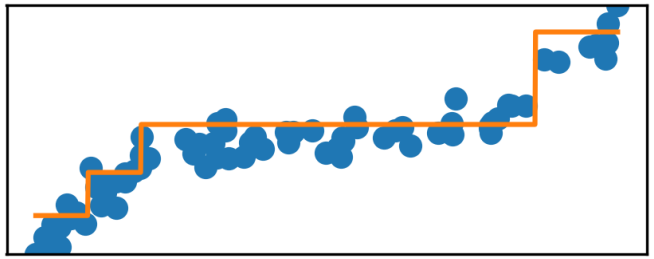
Бустинг на линейных моделях



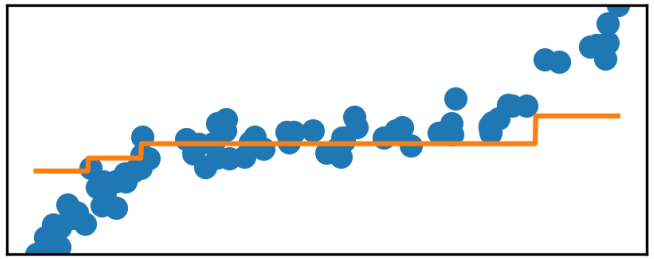
Градиентный бустинг / Gradient Boosting

- x^i – множество объектов обучающей выборки объемом l
- y^i – истинные ответы обучающей выборки
- $a_i(x)$ - i-ый алгоритм
- $Q(y, a(x)) = \frac{1}{l} \sum_{i=1}^{l} \mathcal{L}(y^i, a(x^i))$ – функция потерь

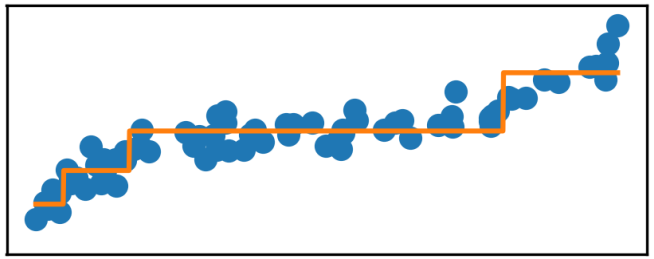
Residual prediction step 1



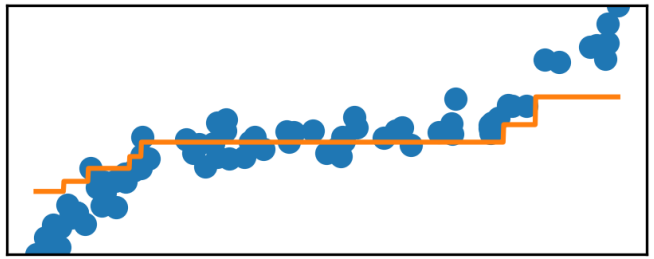
Total prediction step 1



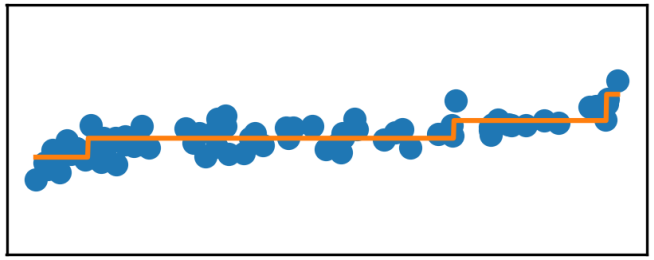
Residual prediction step 2



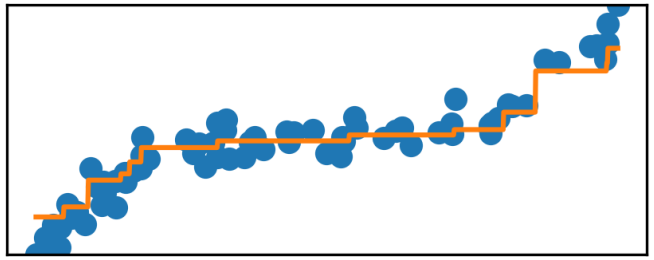
Total prediction step 2



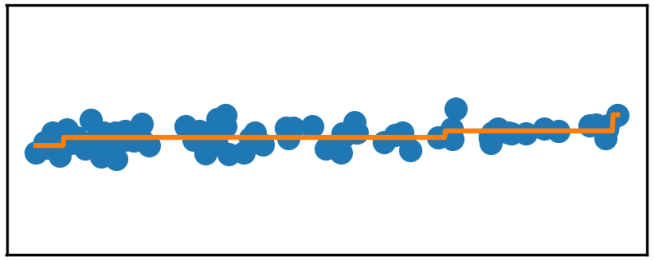
Residual prediction step 5



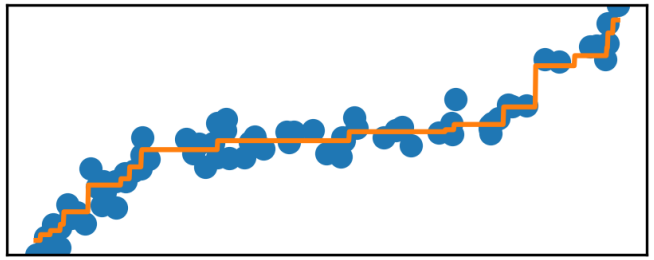
Total prediction step 5



Residual prediction step 9



Total prediction step 9

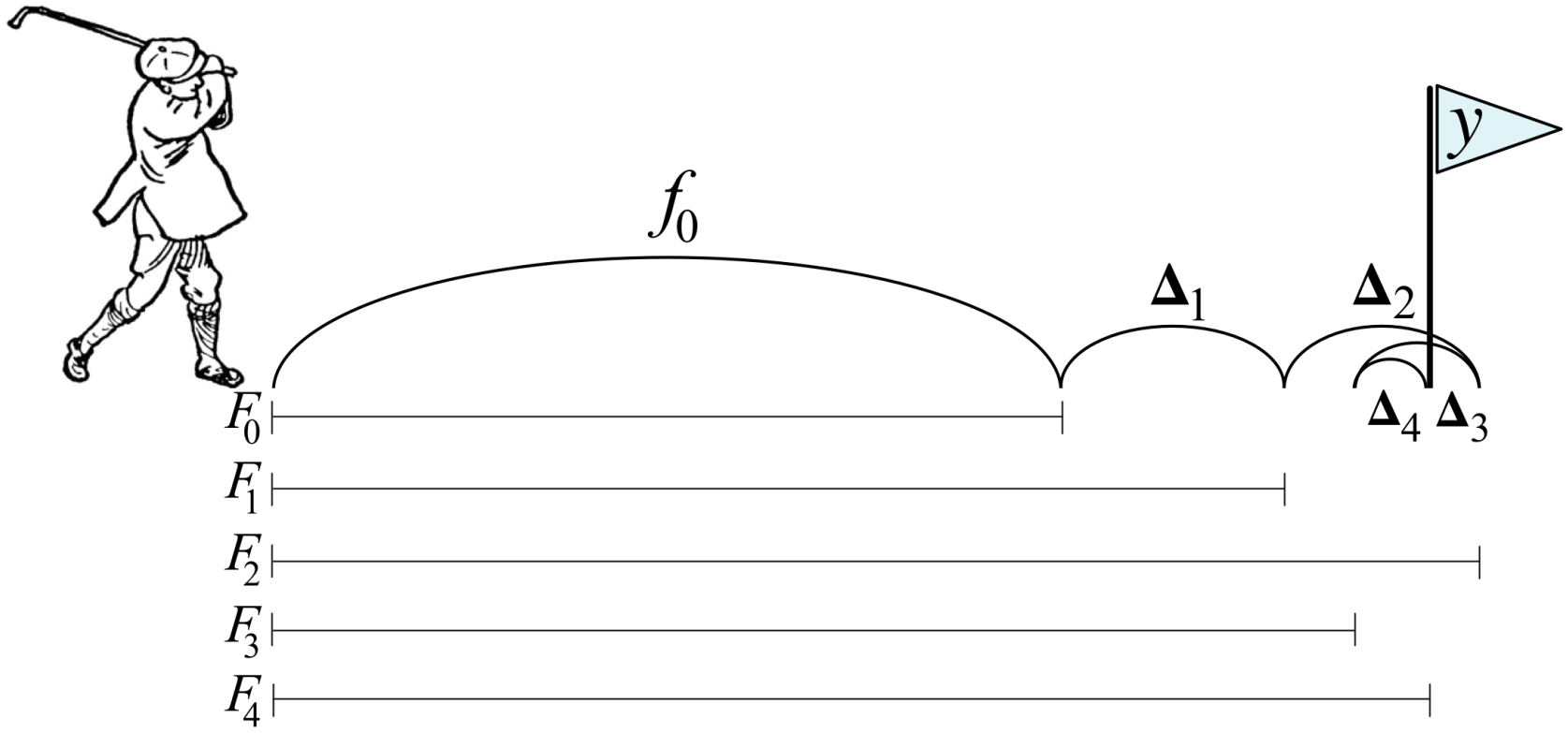


Градиентный бустинг / Gradient Boosting

- Например, построены k алгоритмов, нужно построить a_{k+1} алгоритм с c_{k+1} весом
- Функция потерь для одного i-го объекта:

$$\mathcal{L}(y^i, a(x^i)) = \mathcal{L}(y^i, c_1 a_1(x^i) + \cdots + c_k a_k(x^i) + c_{k+1} a_{k+1}(x^i))$$

Градиентный бустинг / Gradient Boosting



📝 Gradient boosting: Distance to target

Градиентный бустинг / Gradient Boosting

- Базовые модели могут быть любыми, но лучше деревьями
- Решает задачи классификации, регрессии, ранжирования (не очень быстро)
- Нужна большая выборка
- Реализации:
  - `sklearn.ensemble`: scikit-learn.org
  - Extreme Gradient Boosting / `xgboost`: xgboost.readthedocs.io
  - Light Gradient Boosting Machine / `lighgbm`: lightgbm.readthedocs.io
  - Categorical Boosting / `catboost`: catboost.ai
  - Про сравнения и детали реализаций можно почитать тут:
    - CatBoost vs. Light GBM vs. XGBoost
    - When to Choose CatBoost Over XGBoost or LightGBM [Practical Guide]

Итоги

- **Voting**: берем произвольные модели, а результаты агрегируем произвольным образом
- **Бэггинг**: строим бустреп подвыборки и обучаем простые модели на каждой из выборок
  - пример: случайный лес
- **Стекинг**: используем предсказания моделей в качестве признаков для обучения мета-модели
- **Бустинг**: последовательно строим модели, каждая следующая исправляет ошибку предыдущих
  - Бустинг пока еще лучшее решение для табличных данных