

Ансамблі моделей

План уроку

- + Побудова ансамблів: основні принципи та приклади побудови
- + Базові техніки (Max Voting, Averaging, Weighted Averaging)
- + Продвинуті техніки (Stacking, Bagging, Boosting)
- + Алгоритми, засновані на Bagging (Random Forest) та Boosting (AdaBoost, xgboost)

Ансамблеве навчання / Ансамблеві методи

Ансамблеві методи — це підхід до створення алгоритмів машинного навчання, де кілька моделей (часто називаних "слабкими учнями") навчаються для вирішення однієї й тієї ж проблеми та об'єднуються для отримання кращих результатів.

Основна гіпотеза полягає в тому, що при правильному поєднанні слабких моделей ми можемо отримати більш точні та/або надійні моделі.

Мотивація

Уявимо, що нам потрібно прийняти рішення

- + Переглянути певний обраний фільм? (бінарна класифікація)
- + Який фільм переглянути? (мультикласова класифікація)

Мотивація

Ми можемо

- попросити кого-небудь з друзів оцінити фільм за вас і зробити вибір, спираючись на його оцінку
- попросити 5 колег оцінити фільм
- попросити 50 людей оцінити фільм

Мотивація

Ми можемо

- попросити кого-небудь з друзів оцінити фільм за вас і зробити вибір, спираючись на його оцінку
- попросити 5 колег оцінити фільм
- попросити 50 людей оцінити фільм

Багато відповідей від людей з різними думками, і результат, ймовірно, буде найкращим

Попросити кілька різних експертів висловити свою думку (надати рішення задачі) та агрегувати результат — це ідея побудови ансамблів методів.

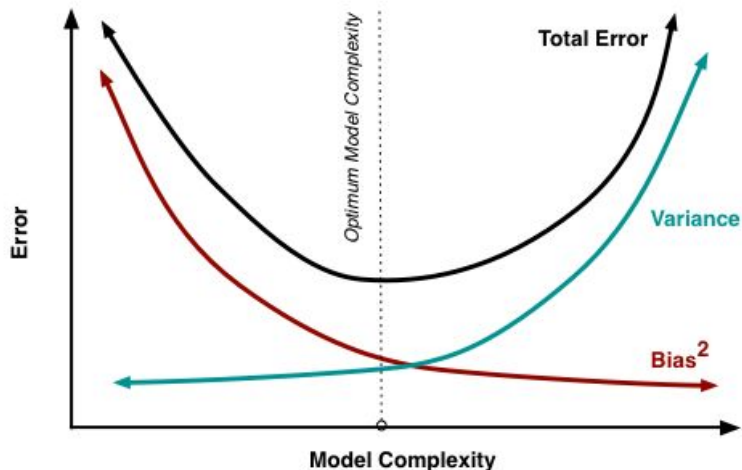
Слабкий та сильний учні

Слабкий учень — базова модель, яка працює добре у деяких випадках.

Слабких учнів можна використовувати як будівельні блоки для проєктування більш складних моделей шляхом об'єднання кількох з них.

У більшості випадків ці базові моделі працюють самостійно не так добре. Це пов'язано з тим, що вони мають високий bias (наприклад, моделі з низьким ступенем свободи) або мають занадто великий variance, щоб бути стійкими (наприклад, моделі з високим ступенем свободи).

Тоді ідея ансамблних методів полягає в тому, щоб спробувати зменшити bias і/або variance таких слабких учнів, об'єднуючи кілька з них разом, щоб створити сильного учня (або модель ансамблю), який досягає кращих результатів.



Однорідні та неоднорідні ансамблеві моделі

При побудові однорідної ансамблевої моделі використовується єдиний (той самий) базовий алгоритм навчання, так що у нас є **однорідні** слабкі учні, які навчаються по-різному (різна ініціація ваг/на різних підвибірках).

Неоднорідна ансамблева модель об'єднує в собі кілька **різних** методів.

Прості техніки ансамблювання

Max Voting / Голосування

Averaging / Усереднення

Weighted Averaging /

Взважене усереднення



```
from sklearn.ensemble import VotingClassifier
```

Max voting / Голосування

Зазвичай використовується в задачах класифікації, коли в результаті модель видає деяке дискретне число.

Уявімо, що наші колеги проголосували за фільм:

Colleague 1	Colleague 2	Colleague 3	Colleague 4	Colleague 5	Final rating
5	4	5	4	4	4

Кінцева оцінка фільму буде **модою** з оцінок колег (у загальному випадку, - експертів).

Averaging / Усереднення

Для передбачень у завданнях регресії або під час обчислення ймовірностей для завдань класифікації використовуємо усереднення:

$$\text{i.e. } (5+4+5+4+4)/5 = 4.4$$

Colleague 1	Colleague 2	Colleague 3	Colleague 4	Colleague 5	Final rating
5	4	5	4	4	4.4

Weighted Average / Взважене усереднення

Якщо для нас важлива думка деяких експертів більше, ніж інших (можливо, ми більше довіряємо їхній думці), ми можемо дати ваги думкам експертів.

The result is calculated as $[(5 \cdot 0.23) + (4 \cdot 0.23) + (5 \cdot 0.18) + (4 \cdot 0.18) + (4 \cdot 0.18)] = 4.41$.

	Colleague 1	Colleague 2	Colleague 3	Colleague 4	Colleague 5	Final rating
weight	0.23	0.23	0.18	0.18	0.18	
rating	5	4	5	4	4	4.41

Вдосконалені методи побудови ансамблів

Різноманітні методи побудови ансамблів для різних цілей

Вибір слабких учнів має бути узгоджений з тим, як ми агрегуємо ці моделі.

Якщо ми обираємо слабких учнів з низьким bias , але високим variance - ансамбль маємо будувати за допомогою методу агрегування, який має тенденцію зменшувати variance .

Якщо ми обираємо слабких учнів з низьким variance , але з високим bias , ансамблем має бути метод агрегування, який має тенденцію зменшувати bias .

Мета-алгоритми об'єднання слабких учнів

- + **Stacking.** У цьому випадку часто враховують *різномірних* слабких учнів, навчають їх паралельно і об'єднують їх, навчаючи метамодель для виведення прогнозу, заснованого на передбаченнях різних слабких моделей.
- + **Bagging.** У цьому випадку часто беруть *однорідних* слабких учнів, навчають їх *паралельно* і незалежно один від одного, а потім об'єднують їх, слідуючи певному детермінованому процесу усереднення.
- + **Boosting.** У цьому випадку часто беруть *однорідних* слабких учнів, навчають їх *послідовно* адаптивним способом (кожен наступний слабкий учень враховує помилки попередніх) і об'єднують їх, слідуючи детермінованій стратегії.
- +

Мета-алгоритми об'єднання слабких учнів

Загалом, ми можемо сказати, що bagging дозволить отримати модель з меншим variance, ніж її компоненти, тоді як boosting і stacking в основному будуть намагатися створити потужні моделі з меншим bias, ніж їх компоненти.

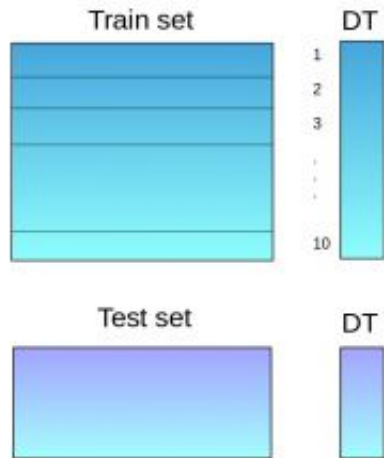
Stacking

Stacking використовує прогнози з декількох моделей (наприклад, дерево рішень, kNN та логістична регресія) для побудови нової моделі. Завершальна модель використовується для прогнозів на тестовій вибірці.

```
sklearn.ensemble.StackingClassifier
```

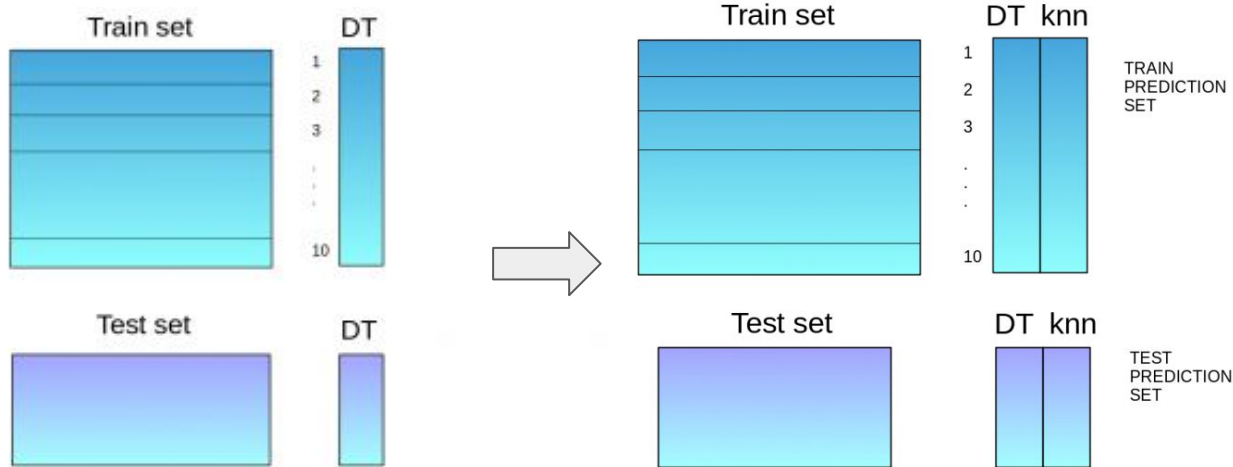
Принцип роботи

1. Навчальний набір даних розділений на 10 частин. Базова модель (допустимо, дерево прийняття рішень) складається з 9 частин, а для 10-ї частини робляться прогнози. Це робиться для кожної частини трен. даних. Потім базова модель (у цьому випадку - дерево рішень) пристосовується до всього трен. набору даних. Використовуючи цю модель, на тестовій вибірці робляться прогнози.



Stacking

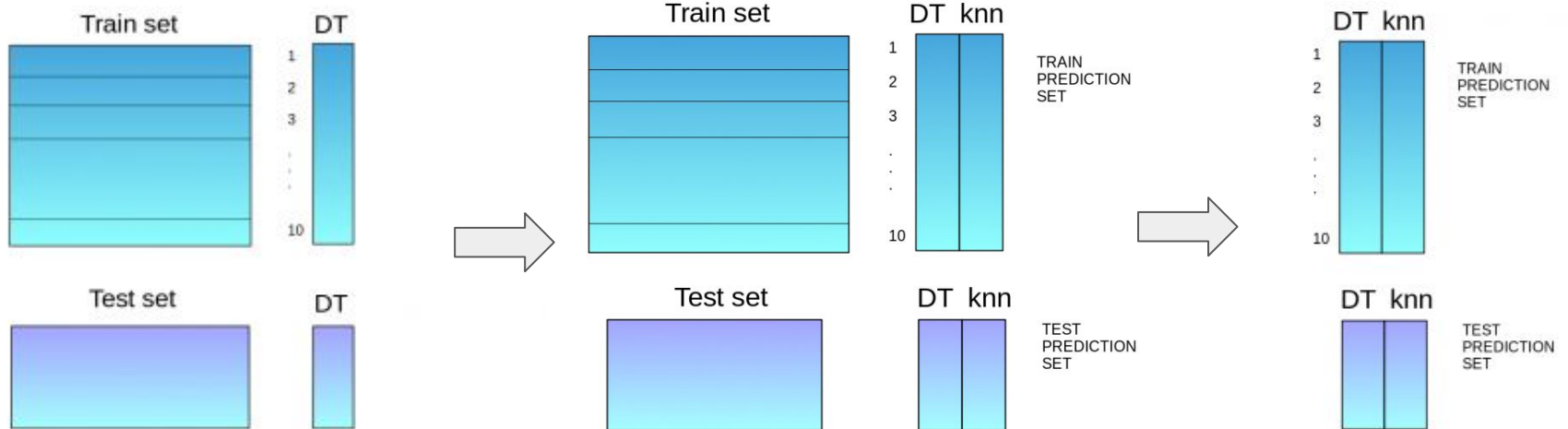
2. Виконуємо те ж саме, використовуючи іншу модель класифікації (скажімо, kNN). Ми можемо використовувати ту ж модель, але з іншими гіперпараметрами. Отримали ще один набір прогнозів.



Stacking

3. Прогнози попередніх простих моделей на тренувальному наборі використовуються як ознаки для побудови нової моделі.

Ця модель використовується для того, щоб зробити остаточні прогнози для набору тестових даних.



Bagging (Bootstrap Aggregating)

Ідея Bagging полягає в об'єднанні результатів декількох моделей (наприклад, всіх дерев рішень) для отримання узагальненого результату. Метою методу є створення ансамблевої моделі, яка є **надійнішою**, ніж окремі моделі, її складові.

Bootstrapping

Цей статистичний метод полягає у генерації вибірок розміру B (так званих bootstrap вибірок) з початкового набору даних розміру N шляхом випадкового вибору елементів з повтореннями в кожному з облікових записів B .



Властивості вибірок методу bootstrap

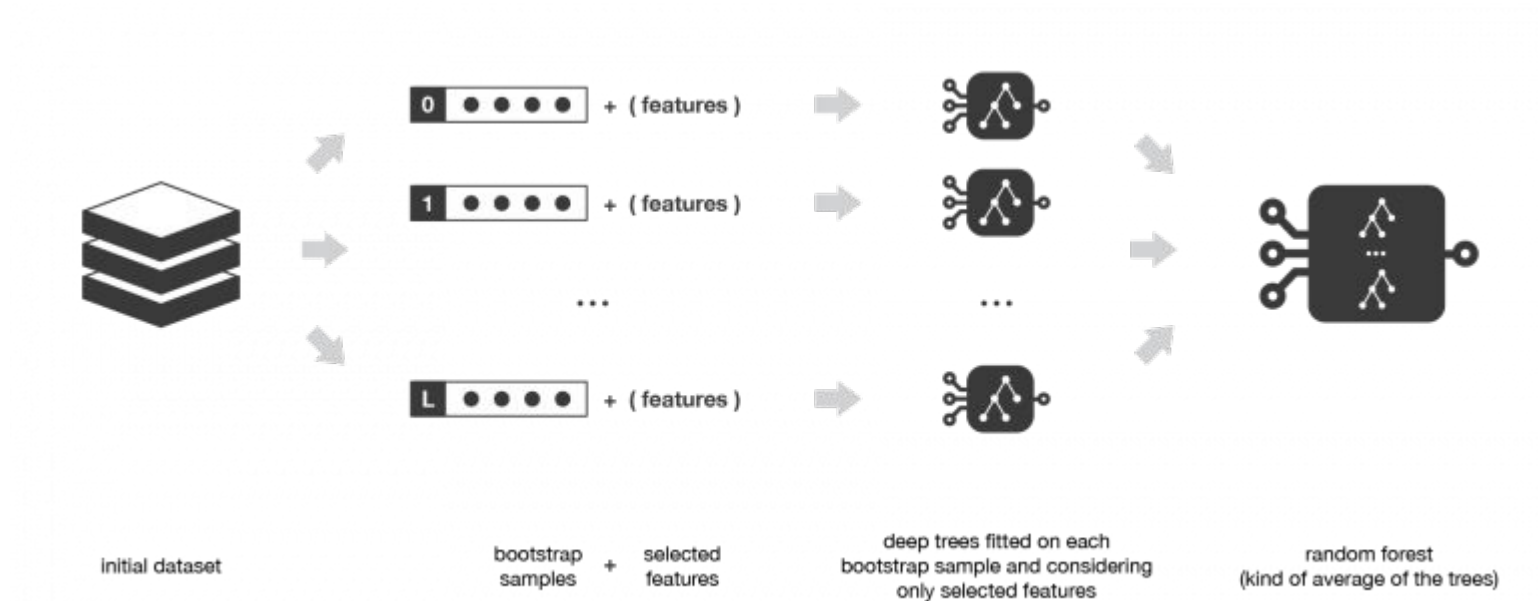
З деякими припущеннями ці вибірки мають досить хороші статистичні властивості, оскільки їх можна розглядати як взяті безпосередньо зі справжнього базового (і часто невідомого) розподілу даних, а також незалежно одне від одного. Це означає, що **їх можна розглядати як репрезентативні та незалежні вибірки справжнього розподілу даних** (майже ідентичні вибірки).

Звісно, для цього вибірки повинні **охоплювати всю складність справжнього набору** даних та не бути занадто корельованими (тобто справжній набір повинен бути досить великим).

Принцип роботи алгоритму

- Декілька підмножин створюються з початкового набору даних, вибираючи спостереження з заміною.
- Базова модель (слабкий учень) навчається на кожній з цих підмножин (по одній моделі на множину).
- Моделі (однорідні) навчаються паралельно і не залежать одна від одної.
- Остаточні прогнози визначаються шляхом об'єднання прогнозів всіх моделей.

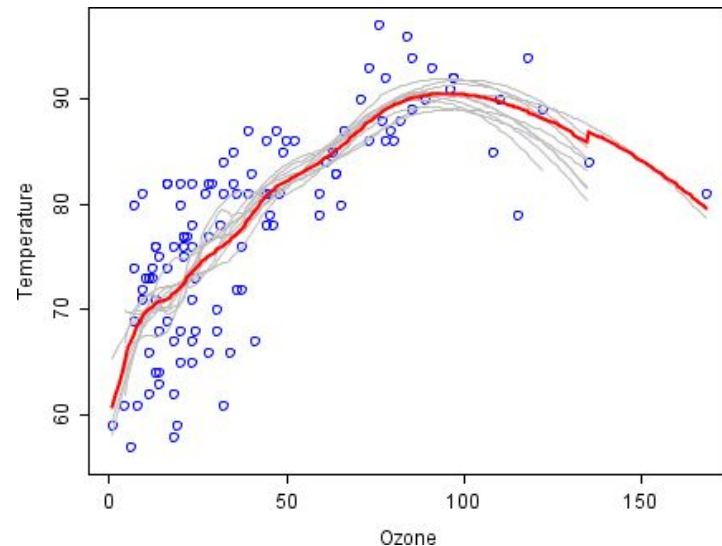
Принцип роботи алгоритму



Random Forest / Випадковий ліс

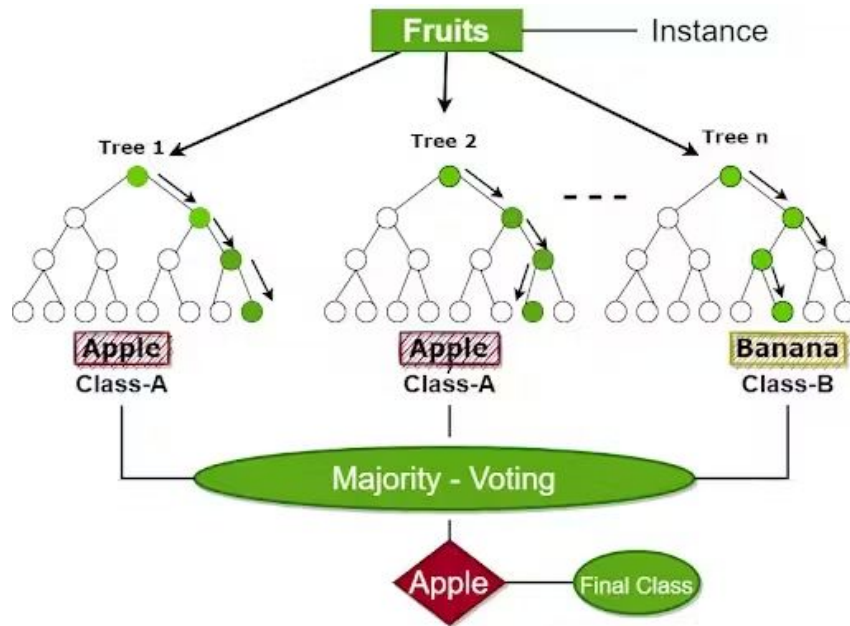
RF (випадковий ліс) - це множина дерев рішень. У задачі регресії їх відповіді усереднюються, у задачі класифікації рішення приймається голосуванням за більшістю.

Ансамбль, побудований за допомогою беггінгу дерев рішень — це випадковий ліс. Random Forest виконує беггінг всередині. Замість того, щоб розглядати всі ознаки при поділі вузла, алгоритм випадкового лісу вибирає найкращу ознаку з підмножини усіх ознак (“feature bagging”). Це призводить до більшої похибки при меншій дисперсії, що дає набагато кращу модель.



Візуалізація роботи алгоритму bagging

Random Forest Example (спрощено)



Принцип роботи

Усі дерева будуються незалежно за такою схемою:

- Вибирається підвибірка навчальної вибірки розміром *samplesize* (можливо з поверненням) - на її основі будується дерево (для кожного дерева — своя підвибірка).
- Для побудови кожного розгалуження в дереві переглядаємо *max_features* випадкових ознак (для кожного нового розгалуження — свої випадкові ознаки).
- Вибираємо кращу ознаку і розгалуження за нею (за заздалегідь заданим критерієм). Дерево, як правило, будується до вичерпання вибірки (поки в листках не залишаться представники лише одного класу). У сучасних реалізаціях є параметри, які обмежують висоту дерева, кількість об'єктів у листках і кількість об'єктів у підвибірці, при якому проводиться розгалуження.

Різниця між деревом рішень та випадковим лісом

Випадковий ліс - це метод об'єднання **кількох дерев прийняття рішень** в один великий класифікатор з використанням ще більшої рандомізації (вибір випадкових вибірок з поверненням для навчання кожного дерева плюс випадковий вибір ознак, які дерево може використовувати для виконання розділення).

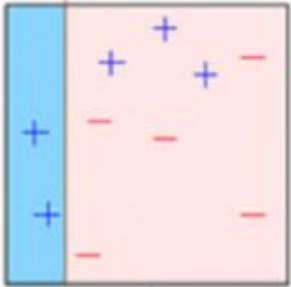
RF - це приклад bagging ансамблю.

Boosting

Бустинг - це послідовний процес, в якому кожна наступна модель намагається **виправити помилки попередньої моделі**. Наступні моделі залежать від попередньої моделі. Отже, тут навчання моделей відбувається НЕ паралельно.

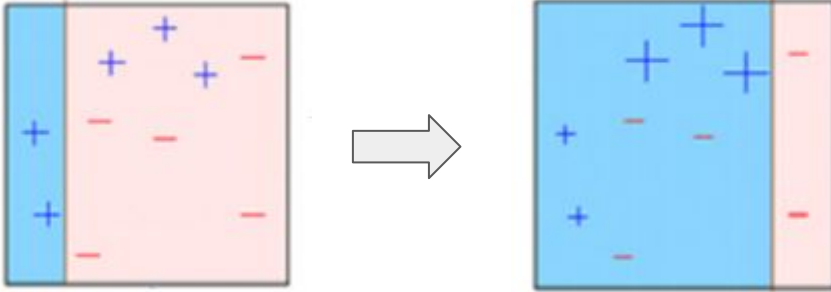
Принцип роботи. 1 - Початок

Ми навчили модель і зафіксували її помилки.



2 - Покращення

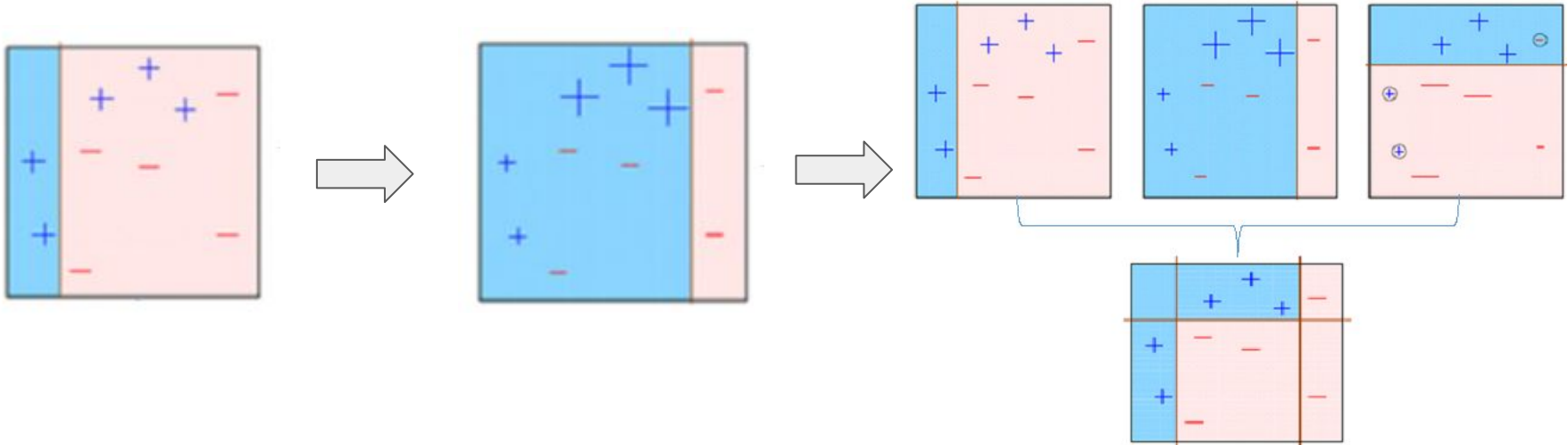
Під час навчання ми збільшуємо вагу екземплярів, на яких попередня модель припустилася помилки, зосереджуючи увагу другої моделі на цих екземплярах.



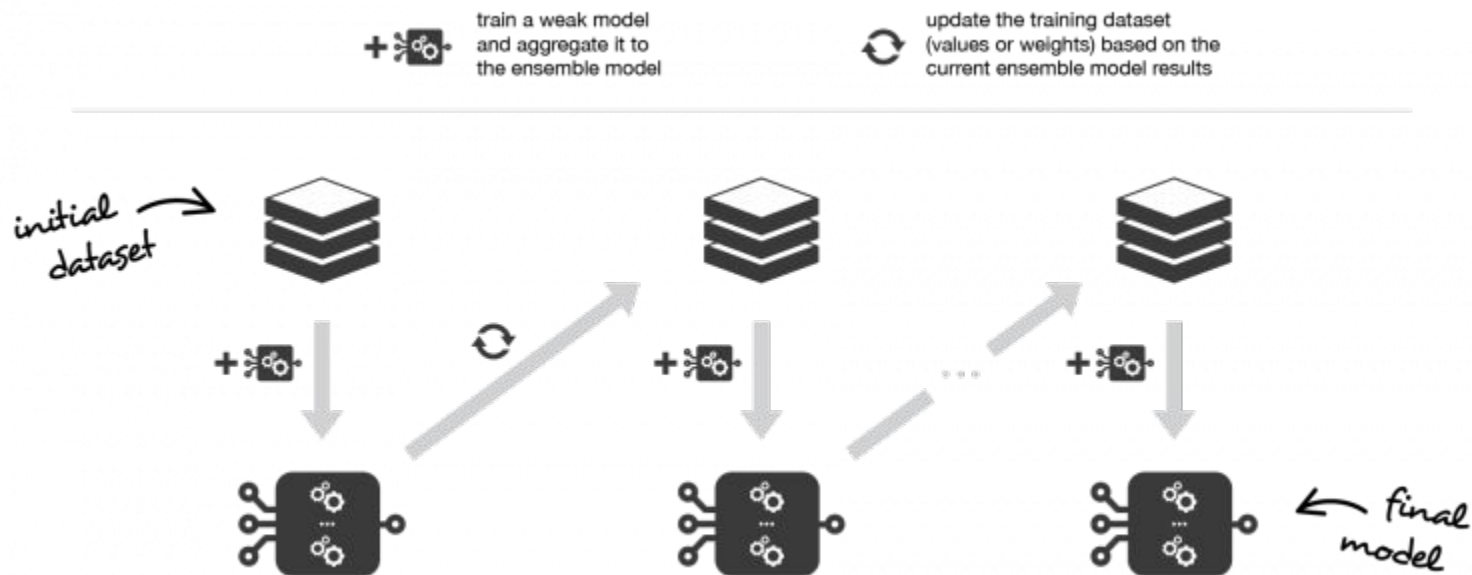
3 - Boosting

І так далі...

Кожна нова модель фокусує свої зусилля на найскладніших об'єктах вибірки при навчанні попередніх моделей, щоб ми отримали в кінці процесу сильного учня з меншим bias (навіть якщо виявиться, що бустінг при цьому зменшує variance). **Бустінг, як і беггінг, може використовуватись як для задач регресії, так і для класифікації.**



Принцип роботи алгоритму



Слабкі учні для бустингу

- Ті, які не потребують великих обчислювальних витрат

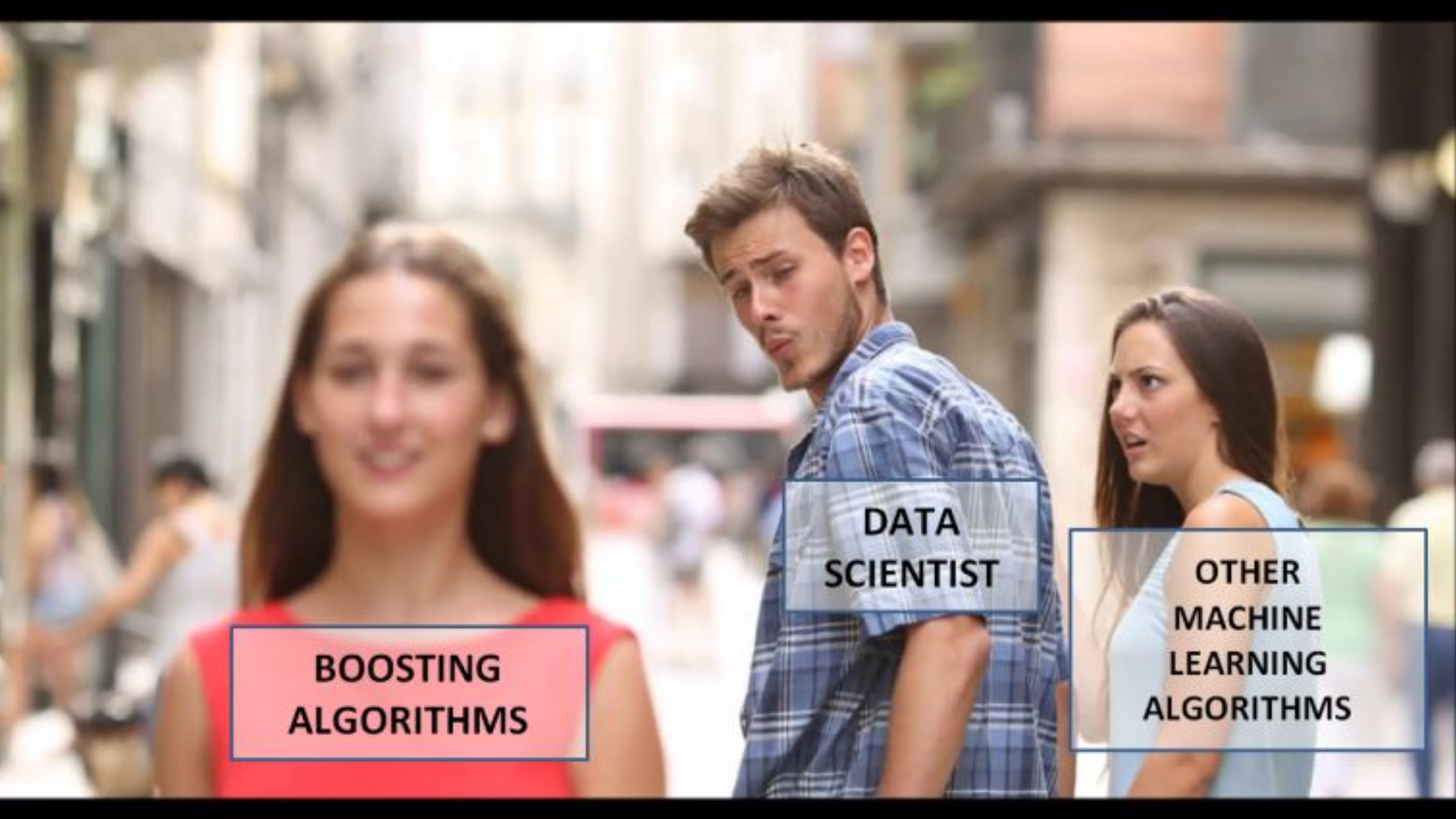
Приклад: неглибокі дерева рішень

Як вибрати спосіб підгону та агрегації слабких учнів

Це описують два метаалгоритми: **adaboost** (адаптивний бустинг) і **градієнтний бустинг**.

Ці два метаалгоритми відрізняються тим, як вони створюють і об'єднують слабких учнів під час послідовного процесу.

Адаптивний бустинг оновлює *ваги*, прикріплені до кожного з об'єктів навчального датасету, тоді як **градієнтний бустинг** оновлює *значення* цих об'єктів. Ця різниця впливає з того, що обидва методи намагаються вирішити задачу оптимізації, що полягає в пошуку найкращої моделі, яка може бути записана у вигляді зваженої суми слабких учасників.

A man in a blue plaid shirt is walking on a city street, looking back over his shoulder at a woman in a red dress who is smiling at him. Another woman in a light blue dress is walking behind him, looking at him with a concerned expression. The background is a blurred city street with other pedestrians.

**BOOSTING
ALGORITHMS**

**DATA
SCIENTIST**

**OTHER
MACHINE
LEARNING
ALGORITHMS**

Додаткова стаття по темі

<https://www.analyticsvidhya.com/blog/2018/06/comprehensive-guide-for-ensemble-models/>