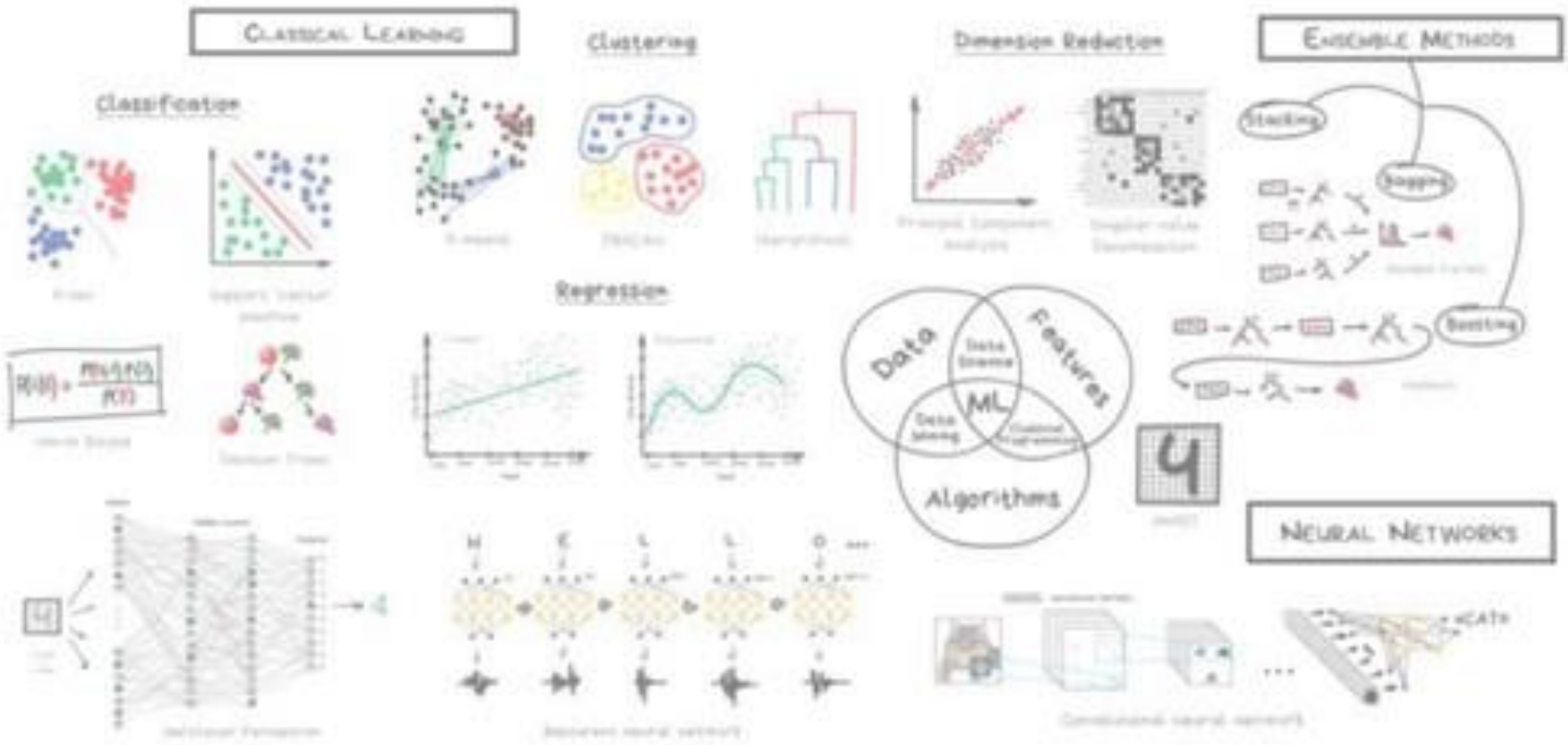
МАШИННЕ НАВЧАННЯ



Ансамблеві методи

Лекція № 13

Ансамблеві методи

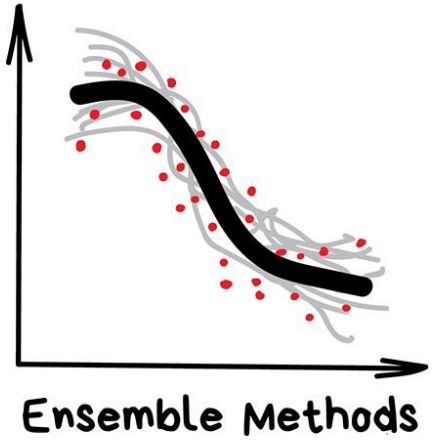
- *Питання:* «Чи модливо, маючи множину відносно слабких і простих моделей (алгоритмів), побудувати сильну модель ?»
- *Відомо:* помилка вимірювань зменшується відповідно корня квадратного з кількості вимірювань
- *Питання:* Чи модливо у машинному нвчанні аналогічним чином усереднювати роботу алгоритмів та надіятися на покращення якості.

Сьогодні використовують для:

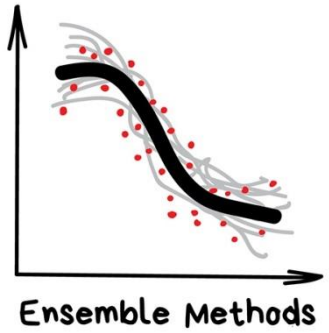
- Всього, де підходять класичні алгоритми (але працюють точніше)
- Пошукові системи
- Компьютерний зір
- Роспізнавання об'єктів

Три перевірених способи робити ансамблі:

- Стекінг = Stacking = Meta Ensembling,
- Беггінг = Bootstrap AGGregatING,
- Бустинг = Boosting



Ансамблеві методи



Ансамблі моделей машинного навчання

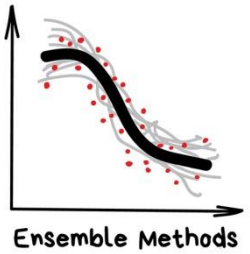
Ансамблі використовують коли потрібно вирішити складне обчислювальне завдання і жоден алгоритм не підходить ідеально. Ансамблі – поєднання одразу кількох алгоритмів, які навчаються одночасно та виправляють помилки один одного. На сьогоднішній день саме вони дають найточніші результати, тому саме їх найчастіше використовують усі великі компанії, для яких важлива швидка обробка великої кількості даних.

Як працюють ансамблі

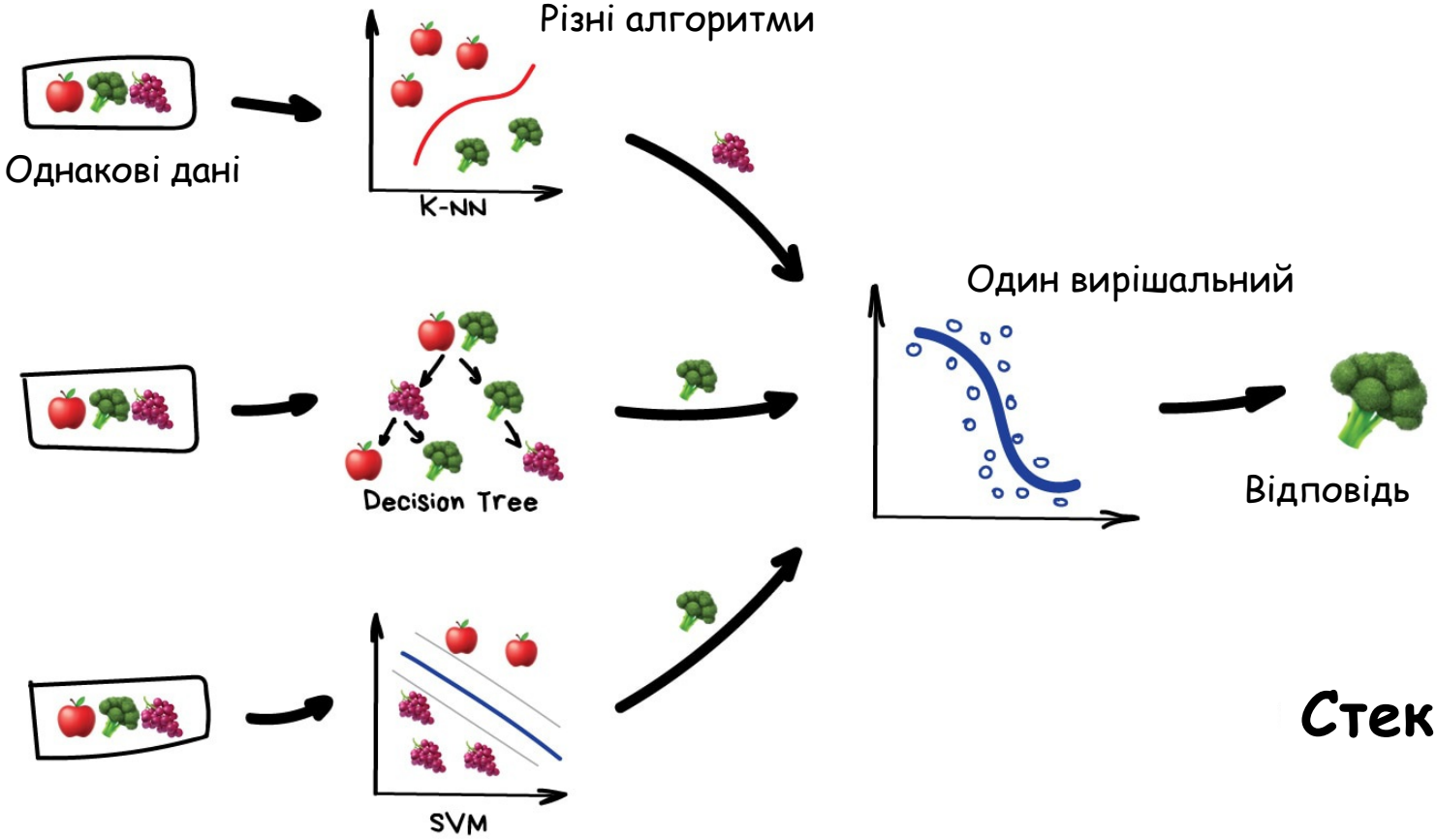
- Идея дуже проста – дати всім одну задачу та спостерігати як вони знайдуть десятки та білбше нових способів вирішити завдання.
- Найкращий результат отримується, коли алгоритми в ансамблях максимально різні. Наприклад, Регресія (Regression) та Дерева пошуку розв'язків (Decision Trees) поєднуються відмінно.
- При цьому поєднання Простого Байесу (Naive Bayes) та методу k-найближчих сусідів (k-nn method) в ансамблях не використовують – вони дуже стабільні і тому не можуть вийти за звичні рамки рішень.

Ансамблеві методи

Стекінг = Stacking = Meta Ensembling

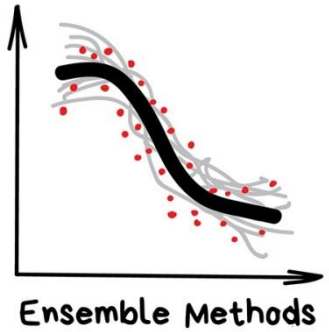


Спочатку навчають декілька **різних алгоритмів на однакових даних ,** а потім результати їх роботи демонсрують останньому (вирішальному) алгоритму. Саме він приймає остаточне рішення. Стекінг – хороший, проте найменш точний ансамбль серед інших методів ансабліювання.

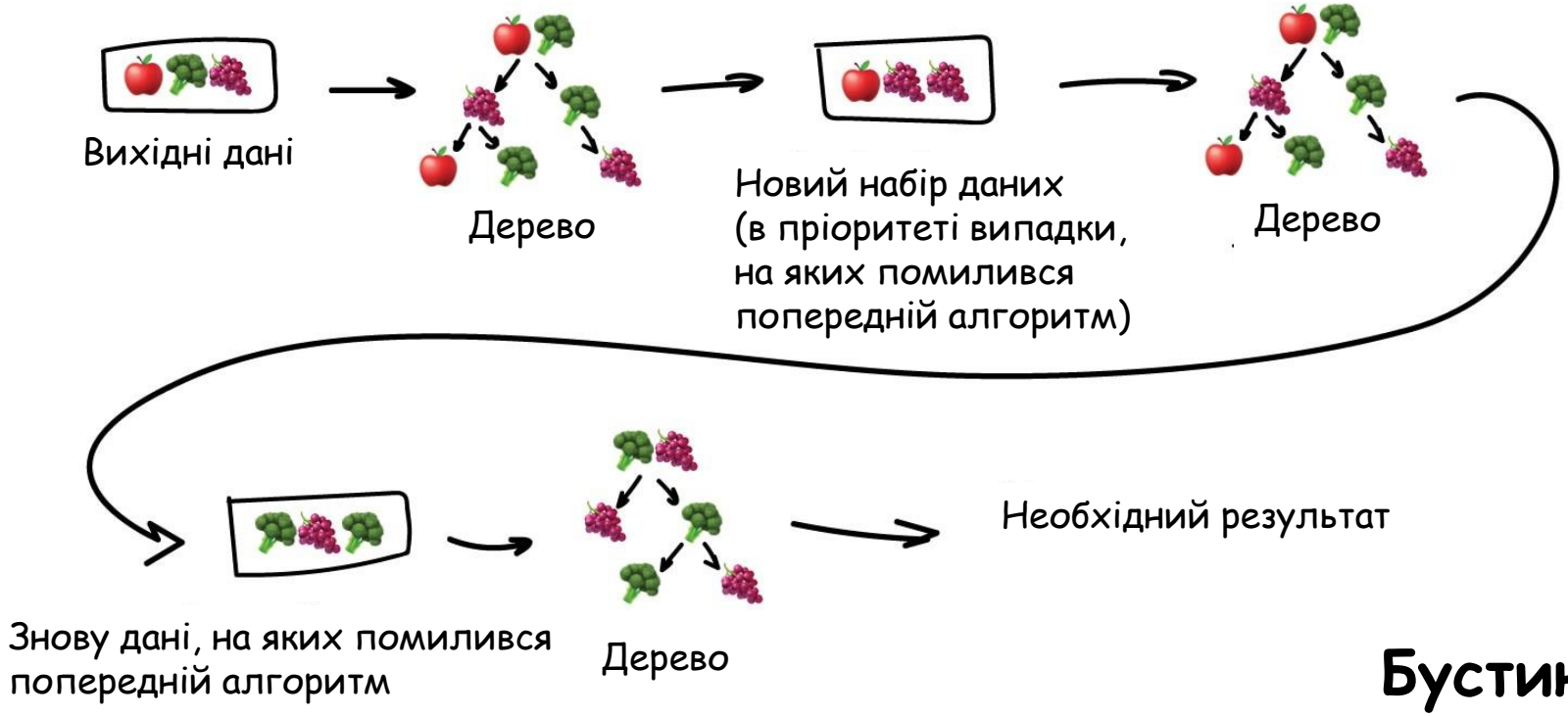


Ансамблевые методы

Бустинг = **Boosting**



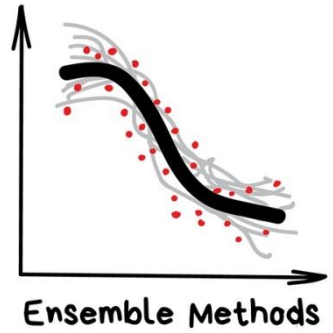
Этот способ включает последовательное обучение алгоритмов. То есть сперва обучаем первый и отмечаем места, где он ошибся. Затем обучаем второй, особое внимание уделяя местам, на которых ошибался первый. И так далее. До необходимого результата.



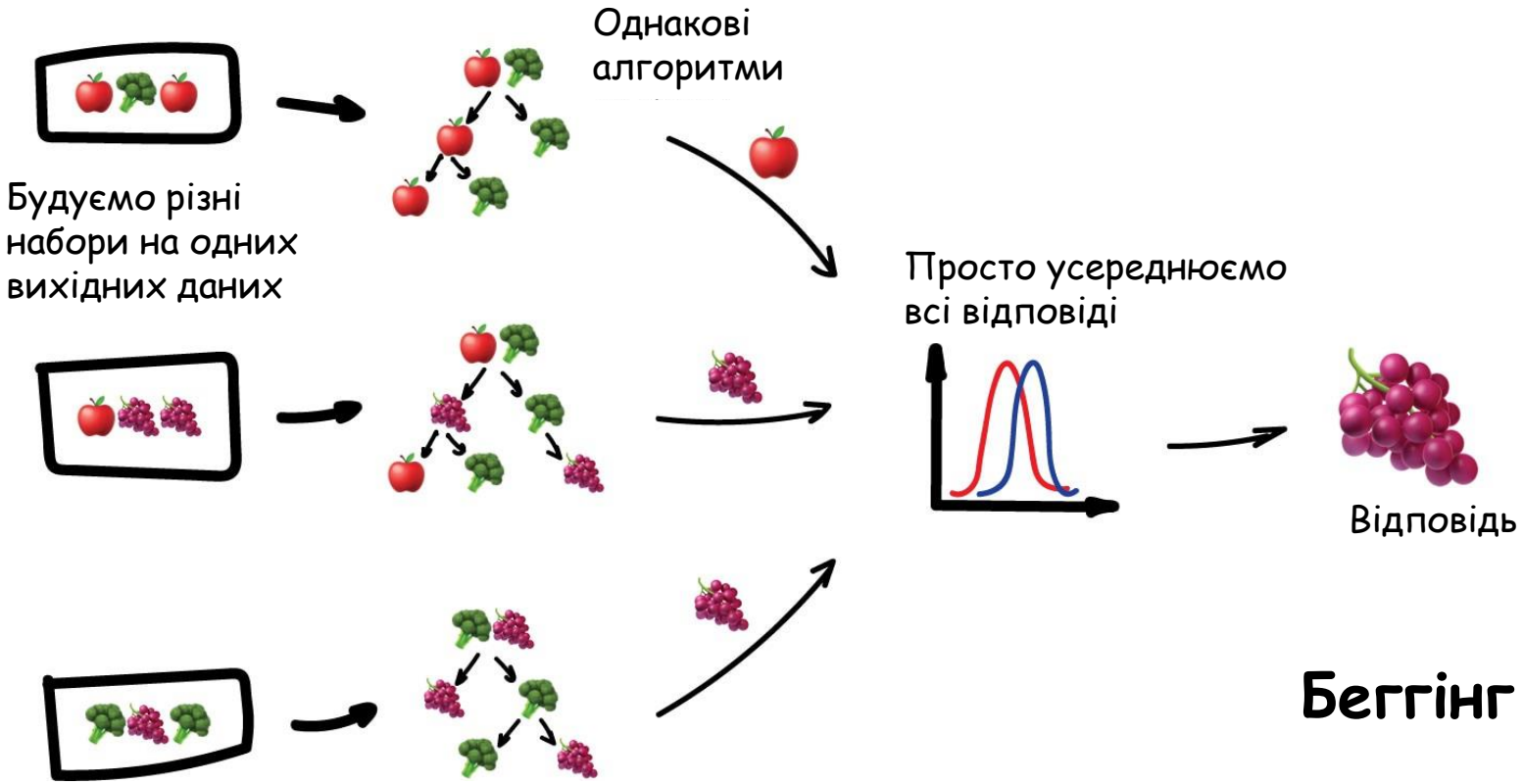
Бустинг

Ансамблеві методи

Беггінг = Bootstrap AGGregatING

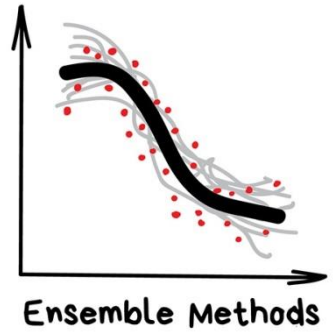


Цей тип навчання означає: **багато разів навчати один алгоритм на довільних вибірках вихідних даних**. І зрештою усереднити відповіді. Це виглядає як голосування за найпопулярнішу відповідь, де багато моделей працюють паралельно.

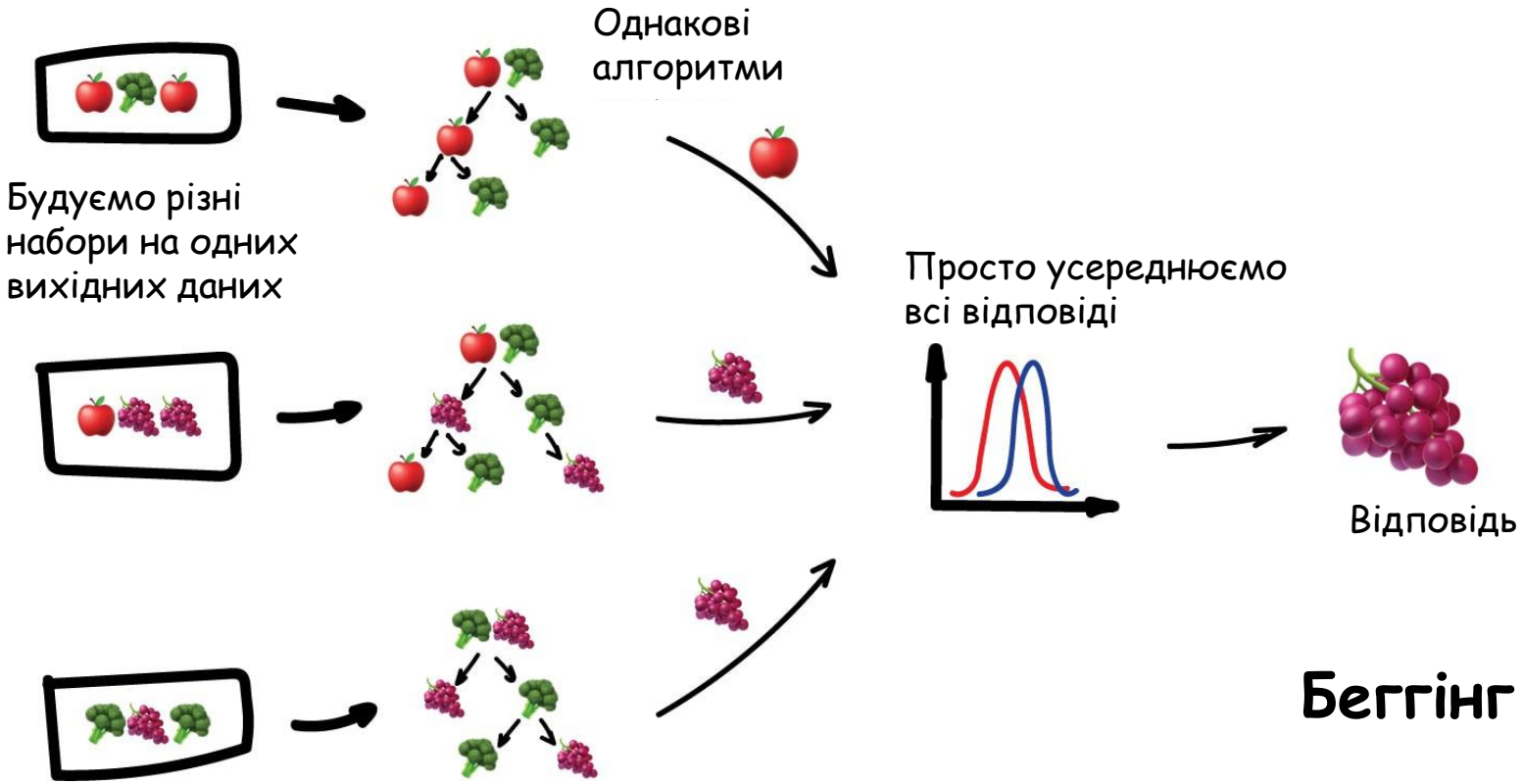


Ансамблеві методи

Беггінг = Bootstrap AGGregatING

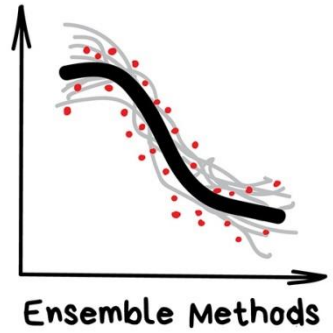


Дані у випадкових вибірках можуть повторюватися. Тобто з набору 1-2-3 ми можемо робити вибірки 2-2-3, 1-2-2, 3-1-2 і так поки не набридне. На них ми навчаємо той самий алгоритм кілька разів, а в кінці обчислюємо відповідь простим голосуванням.

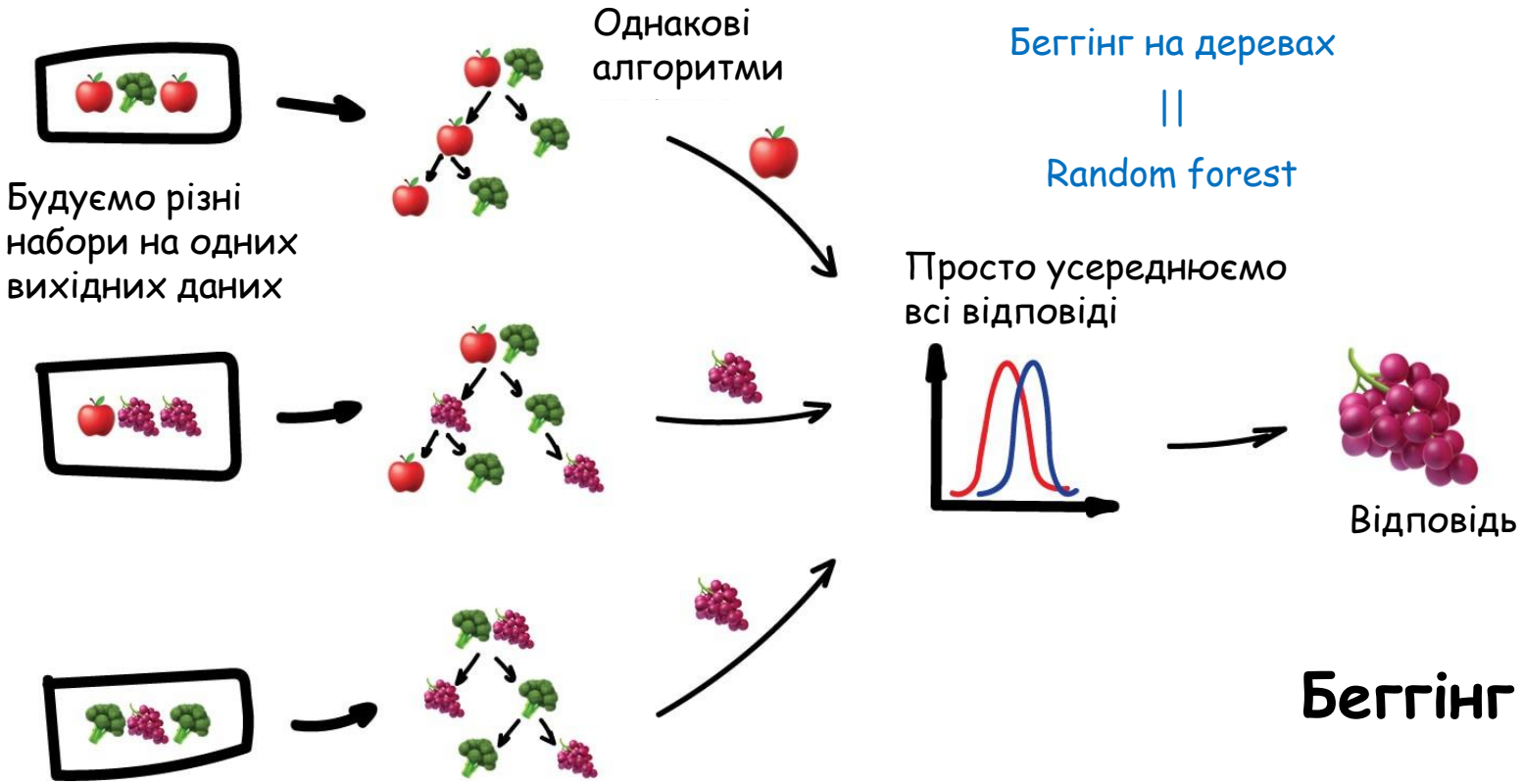


Ансамблеві методи

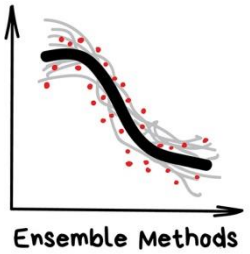
Беггінг = Bootstrap AGGregatING



Найпопулярніший приклад беггінгу - алгоритм **Random Forest**, бегінг на деревах, який і намальований на картинці. Коли ви відкриваєте камеру на телефоні і бачите, як вона окреслила обличчя людей у кадрі жовтими прямокутниками — це приклад їхньої роботи.



Визначення ансамблю



Вихідні дані:

$X^\ell = (x_i, y_i)_{i=1}^\ell \subset X \times Y$ – навчальна вибірка, $y_i = y^*(x_i)$

$a_t: X \to Y,\ t = 1, \dots, T$ – базові алгоритми, що навчаються

Ідея ансамблю: чи можливо з множини «поганих» алгоритмів a_t побудувати один «гарний» ?

Декомпозиція базових алгоритмів $a_t(x) = C(b_t(x))$:

$a_t: X \xrightarrow{b_t} R \xrightarrow{C} Y$, де R – найбільш зручний простір оцінок;

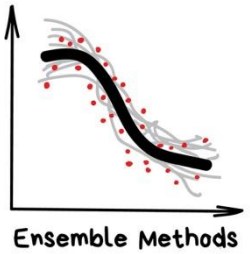
C – вирішальне правило, як правило досить простого вигляду

Ансамбль базових алгоритмів $b_1, \dots, b_T$:

$$a(x) = C\big(F(b_1(x), \dots, b_T(x), x)\big),$$

$F:\ R^T \times X \to R$ – агрегуюча функція або мета-алгоритм

Агрегуючі функції



Загальні вимоги до агрегуючих функцій:

- $F(b_1, \dots, b_T, x) \in [min_t b_t, max_t b_t]$ – середнє для всіх x
- $F(b_1, \dots, b_T, x)$ – монотонно не спадає по всім b_t

Приклади агрегуючих функцій

- Просте голосування (simple voting)

$$F(b_1, \dots, b_T) = \frac{1}{T}\sum_{t-1}^{T} b_t$$

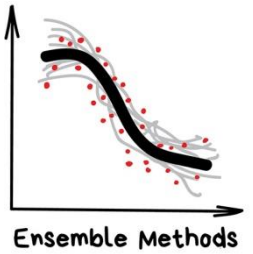
- Зважене голосування (weighted voting)

$$F(b_1, \dots, b_T) = \sum_{t=1}^{T} \alpha_t b_t\,, \qquad \sum_{t=1}^{T} \alpha_t = 1, \qquad \alpha_t \geq 0$$

- Суміш алгоритмів (mixture of experts) з функціями компетентності

$$F(b_1, \dots, b_T, x) = \sum_{t=1}^{T} g_t(x) b_t(x)$$

Проблема різновиду базових алгоритмів



Якщо розглядати значення базових алгоритмів b_t на об'єкті як **незалежні випадкові величини** ξ_t з однаковим мат очкуванням та однаковою дисперсією, то випадкова величина $\xi = \frac{1}{T}(\xi_1 + \cdots + \xi_T)$ має таке ж мат очкування, але меншу дисперсію:

- $E\xi = \frac{1}{T}(E\xi_1 + \cdots + E\xi_T) = E\xi_t$ – математичне очікування
- $D\xi = \frac{1}{T^2}(D\xi_1 + \cdots + D\xi_T) = \frac{1}{T}D\xi_t$ – дисперсія $\to 0$ при $T \to \infty$

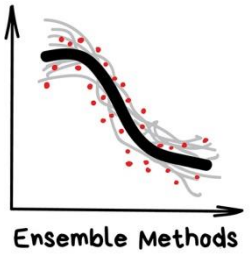
Базові алгоритми не є незалежними випадковими величинами:

- Вирішують одне й те ж завдання
- Налаштовуються на один цільовий вектор
- Зазвичай обираються з однієї ж і тієї моделі

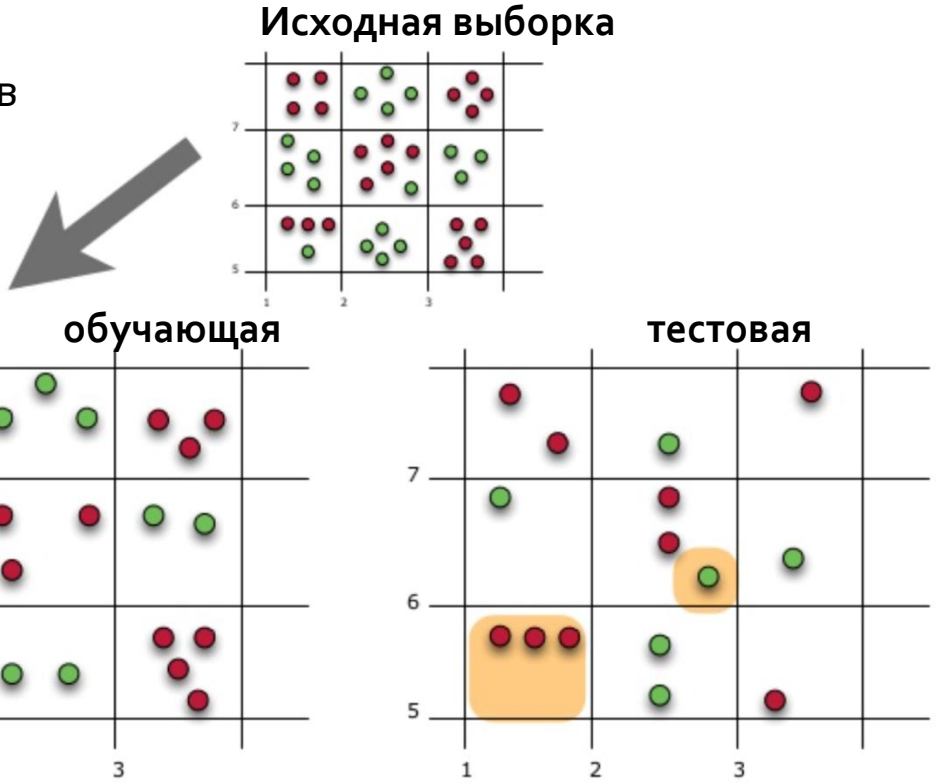
Способи підвищення різновиду базових алгоритмів

- Навчання на різних (випадкових) підвибірках
- Навчання на різних (випадкових) наборах ознак
- Навчання з різних параметричних моделей
- Навчання з використанням рандомізації
- (інколи) Навчання на зашумлених даних

Методи стохастичного ансамбліювання



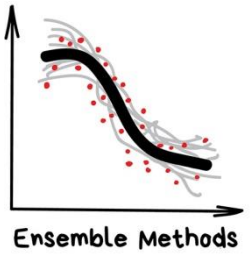
Кожне дерево будується з використанням різних вихідних даних. Приблизно 37% прикладів залишаються поза вибіркою і не використовуються при побудові k-го дерева.



Доведення

Нехай у вибірці ℓ об'єктів. На кожному кроці всі об'єкти потрапляють у вибірку з поверненням рівноймовірно, тобто окремий об’єкт — з імовірністю $1/\ell$. Імовірність того, що об’єкт **НЕ** попаде до підвибірки (тобто його не взяли ℓ разів): $(1-1/\ell)^{\ell}$. При $\ell \to \infty$ отримуємо одну з «чудових" границь $1/e$. Тоді, ймовірність попадання конкретного об’єкту до підвибірки $\approx 1-1/e \approx 63\%$.

Методи стохастичного ансамбліювання



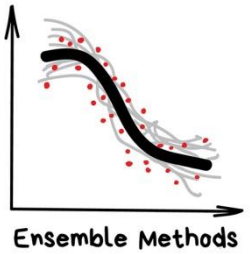
Способи підвищення різновиду з використанням рандомізації

- Bagging (bootstrap aggregating) – використання підвибірок навчальної вибірки з "поверненням", в кожну вибірку попадає $\left(1-\frac{1}{e}\right) \approx 63\%$ об'єктів
- Pasting – випадкові підвибірки для навчання
- Random subspaces – випадкові підмножини озеак
- Random patches – випадкові підмножини об'єктів та ознак
- Cross-validated committees – вся вибірка розбивається на k-блоків і проводиться навчання k-разів без одного блоку

У загальному вигляді задачу навчання кожного алгоритму на своїй підвибірці можна сформулювати таким чином:

$\mu: (G, U) \mapsto b_t$ – метод навчання на підвибірці $U \subseteq X^\ell$, який використовує лише ознаки з множини $G \subseteq F^n = \{f_1, \dots, f_n\}$

Методи стохастичного ансамбліювання (у псевдокоді)



Вхід: навчальна вибірка X^ℓ, параметри базових Т алгоритмів

ℓ' – розмір кожної навчальної підвибірки

n' – розмірність підпросторів ознак

ε_1 – поріг якості базових алгоритмів на навчанні

ε_2 – поріг якості базових алгоритмів на тестуванні

Вихід: базові алгоритми b_t, $t = 1, \dots, T$

1: **для всіх** $t = 1, \dots, T$

2: визначаємо випадкову підвибірку U_t розміром ℓ' з X^ℓ

3: визначаємо випадкову множину ознак G_t розмірності n' з F^n

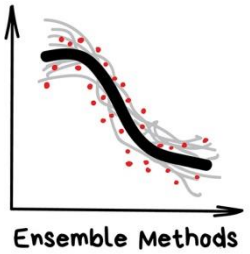
4: застосовуємо базовий алгоритм $b_t = \mu(G_t, U_t)$

5: якщо $\mathrm{Q}(G_t, U_t) > \varepsilon_1$ то b_t не включається в ансамбль

6: якщо $\mathrm{Q}(G_t, X^\ell / U_t) > \varepsilon_2$ то b_t не включається в ансамбль

Ансамбль – просте голосування $b(x) = \frac{1}{T}\sum_{t-1}^{T} b_t(x)$

Незміщена оцінка помилок



Out-of-bag – незміщена оцінка ансамблю на об'єкті

$$OOB(x_i) = \frac{1}{|T_i|}\sum_{t \in T_i} b_t(x_i)\,, \qquad T_i = \{t : x_i \notin U_t\}$$

Незміщена оцінка помилки ансамблю на навчальній вибірці

$$OOB\left(X^\ell\right) = \sum_{i=1}^{\ell} \mathcal{L}(OOB(x_i), y_i)\,,$$

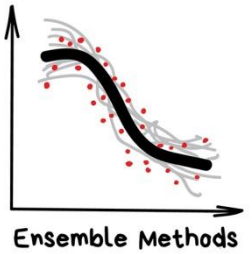
де $\mathcal{L}$ – значення функції втрат на об'єкті x_i

Оцінка важливості (importance) ознак $f_j, j = 1, \ldots, n$:

$$I_j = \frac{OOB^j\left(X^\ell\right) - OOB\left(X^\ell\right)}{OOB(X^\ell)} \times 100\%$$

При обчисленні $b_t(x_i)$ для OOB^j значення ознаки f_j випадковим чином перемішуються на всіх об'єктах $x_i \notin U_t$

Переваги та обмеження стохастичного ансамбліювання



Переваги:

- Метод-обгортка (envelope) над базовим методом навчання
- Підходить для класифікації, регресії й інших завдань
- Проста реалізація
- Просте розпаралелювання
- Можливість отримання незміщених оцінок OOB
- Можливість оцінювання важливості ознак
- Один із найкращих та універсальних методів – випадковий ліс

Обмеження:

Потребує дуже багато базових алгоритмів

Складно агрегувати стійкі базові методи навчання

Дякую за увагу