# Тема 3. Класичні алгоритми для навчання з вчителем

Класичне машинне навчання або Classical Machine Learning, будується на класичних статистичних алгоритмах і вирішує питання, що пов'язані з прийняттям рішень на основі даних. Перші алгоритми навчання розроблено ще в 1950-х. Вони вирішували формальні завдання - шукали закономірності в числах, оцінювали близькість точок в просторі і вираховували напрямки.

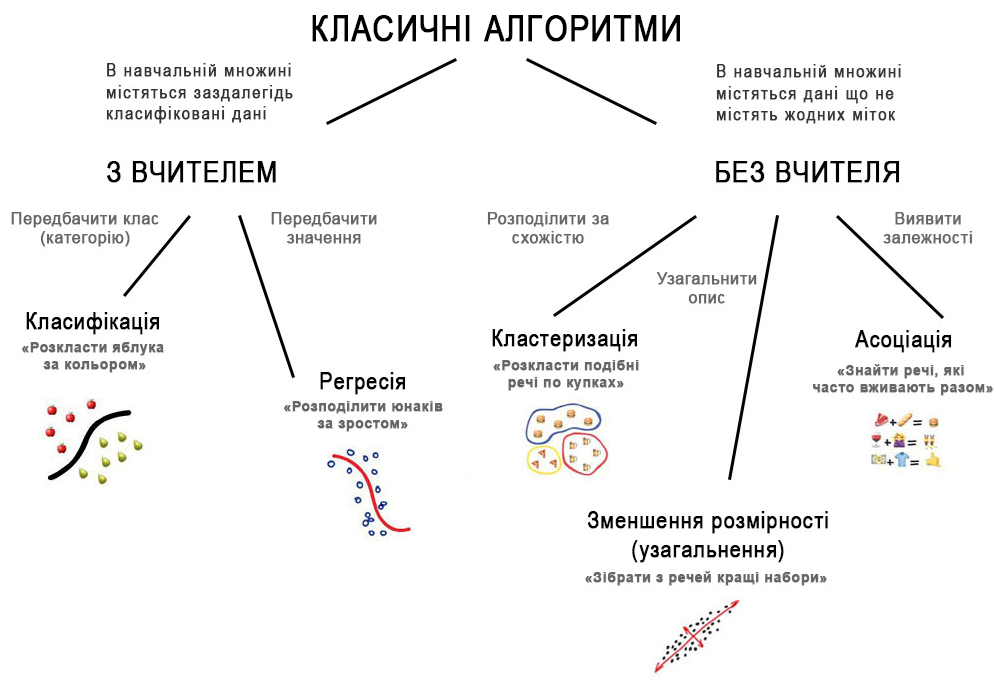


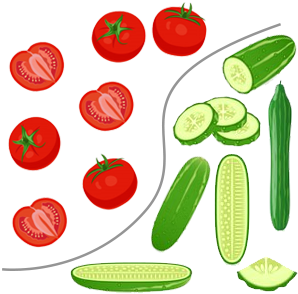
Рис. 3.1. Класифікація завдань, що вирішуються класичними алгоритмами

Сьогодні на класичних алгоритмах тримається добра половина Інтернету. Коли зустрічається блок «Рекомендовані статті» на сайті, або банк блокує гроші на картці після першої покупки кави за кордоном - це майже завжди результат виконання одного з цих алгоритмів.

Класичні алгоритми є популярними і простими для пояснення. Сьогодні вони як основи арифметики – використовуються постійно, але іноді про них забувають. Ці алгоритми часто застосовуються в Data Mining і їх можна розглядати як частину Data Science.

## 3.1. Класифікація

Розділення об'єктів за заздалегідь відомою ознакою: олівці за кольорами, документи за мовами, музику за жанрами. Використовується для:

* Спам-фільтрів.
* Визначення мови.
* Пошуку схожих документів.
* Аналіз тональності.
* Розпізнавання рукописних букв і цифр.
* Визначення підозрілих транзакцій.

**Популярні алгоритми:** [Наївний Байєс](https://ru.wikipedia.org/wiki/Наивный_байесовский_классификатор), [Дерева Рішень](https://logic.pdmi.ras.ru/~sergey/teaching/mlcsclub/02-dectrees.pdf), [Логістична Регресія](https://ru.wikipedia.org/wiki/Логистическая_регрессия), [K-найближчих сусідів](https://ru.wikipedia.org/wiki/Метод_k-ближайших_соседей), [Метод Опорних Векторів](https://ru.wikipedia.org/wiki/Метод_опорных_векторов).

Класифікація є самим популярним завданням для машинного навчання. Для класифікації завжди потрібен вчитель - розмічені дані з ознаками і категоріями, на яких машина буде вчитися розрізняти ознаки. Класифікувати можна що завгодно: користувачів за інтересами - так роблять алгоритмічні рядки, статті за мовами і тематикою - важливо для пошукових систем, музику за жанрами - плейлисти на музичних сервісах, листи у поштовій скриньці.

У класифікації є корисна зворотна сторона - пошук аномалій. Якщо якась ознака об'єкта сильно не вписується у визначені класи, вона яскраво підсвічується на екрані. Такий підхід є популярним в медицині: комп'ютер підсвічує всі підозрілі області МРТ або виділяє відхилення в аналізах. На біржах таким чином визначають нестандартних гравців, які швидше за все є інсайдерами. Навчивши комп'ютер «як правильно», автоматично отримується і зворотний класифікатор - як неправильно.

Наївний байєсів алгоритм складається з простої арифметики та простих обчислень: множення і ділення. Незважаючи на простоту, може бути доволі точним. Може бути успішно застосований для фільтрації спаму(рис. 3.2).

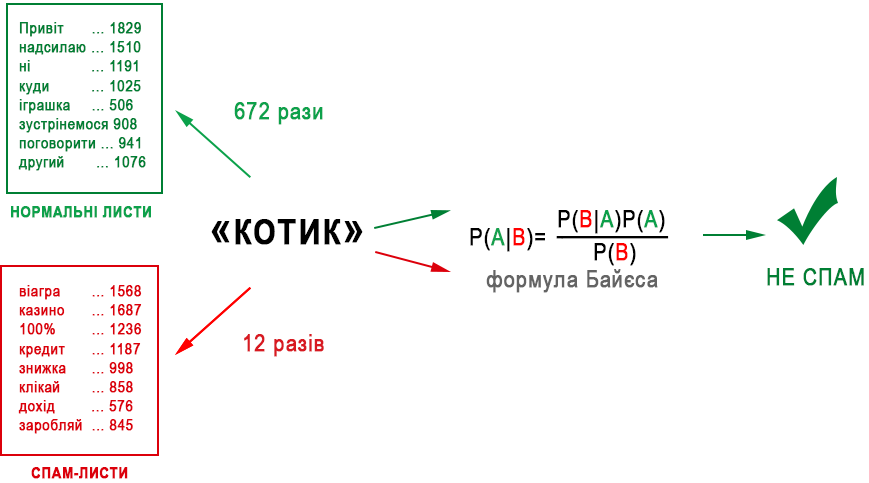


Рис. 3.2. Наочний приклад застосування фільтра Байєса для визначення спаму.

Раніше всі спам-фільтри працювали на алгоритмі Наївного Байєса. Машина враховувала скільки разів слово «казино» зустрічається в спамі, а скільки разів в нормальних листах. Перемножувала ці дві ймовірності за формулою Байєса, складала результати всіх слів і видавала як машинне навчання

Пізніше спамери навчилися обходити фільтр Байєса, просто вставляючи в кінець листа багато слів з «хорошими» рейтингами. Метод отримав іронічну назву «Отруєння Байєса», а спам стали фільтрувати за іншими алгоритмами. Але метод назавжди залишився в підручниках як найпростіший, красивий і один з перших практично корисних.

**Застосування наївного байєсівського алгоритму**

* **Класифікація в режимі реального часу.** НБА дуже швидко навчається, тому його можна використовувати для обробки даних в режимі реального часу.
* **Багатокласова класифікація.** НБА забезпечує можливість багатокласової класифікації. Це дозволяє прогнозувати ймовірність для багатьох значень цільової змінної.
* **Класифікація текстів, фільтрація спаму, аналіз тональності тексту.** При вирішенні завдань, пов'язаних з класифікацією текстів, НБА перевершує багато інших алгоритмів. Завдяки цьому, даний алгоритм знаходить широке застосування в області фільтрації спаму (ідентифікація спаму в електронних листах) і аналізу тональності тексту (аналіз соціальних медіа, ідентифікація позитивних та негативних думок клієнтів).
* **Рекомендаційні системи.** Наївний байєсівський класифікатор у поєднанні з колаборативною фільтрацією (collaborative filtering) дозволяє реалізувати рекомендаційну систему.

На практиці використовується в наступних областях машинного навчання:

* Визначення спаму, що приходить на електронну пошту.
* Автоматична прив'язка новинних статей до тематичних рубрик.
* Виявлення емоційного забарвлення тексту.
* Розпізнавання осіб чи інших об’єктів на зображеннях.

### Дерева Рішень

Дерево рішень (Decision Trees) - це графічна модель, що використовується в машинному навчанні для прийняття рішень. Воно є одним із найбільш популярних та інтерпретованих методів навчання з учителем для вирішення завдань класифікації та прогнозування. Дерева рішень дозволяють візуально і аналітично оцінити результати вибору різних рішень і використовуються в галузі статистики та аналізу даних для прогнозних моделей.

Дерева рішень використовують, коли потрібно прийняти рішення в умовах невизначеності, де кожне рішення залежить від результату попередніх результатів або деяких заданих умов, що з'являються з певною ймовірністю.

Дерево рішень є ієрархічною структурою, що складається з вузлів і листя. Вузли є рішення, а листя - класи чи прогнози.

Кореневий вузол

Вузол

Вузол

Листя

Листя

Листя

Вузол

Листя

Листя

Батьківський Вузол

Дочірні Вузли  
(нащадки)

Рис. 3.3. Структура дерева рішень

Дерево рішень будується з врахуванням навчального набору даних. На кожному кроці вибирається найкраща ознака (фактор), який поділяє дані на найбільш однорідні підгрупи.

Алгоритми побудови визначають суттєві фактори. На кожному вузлу ієрархії використовується критерій, який вирішує найбільшу невизначеність. Суттєві фактори розташовуються на найближчому рівні від кореня ніж інші.

У найбільш простому вигляді дерево рішень - це спосіб представлення правил «Якщо, тоді» в ієрархічній, послідовної структурі. Основою такої структури є відповіді "Так" або "Ні" на ряд питань.

Алгоритм автоматично розділяє всі дані відповідно від питання. Вирішується завдання бінарної класифікації, для вибору є лише дві відповіді на поставлене питання ("так" і "ні"). Бінарні дерева є найпростішим, окремим випадком дерев рішень. В інших випадках, відповідей і, відповідно, гілок дерева, що виходять з його внутрішнього вузла, може бути більше за двох.

Побудова дерева може бути зупинена, коли виконано одну з умов, таких як досягнення максимальної глибини дерева, недостатня кількість елементів у вузлі або інші критерії.

Коли дерево готове, воно може використовуватися для прогнозування класів (у задачах класифікації) або числових значень (у задачах регресії) для нових даних.

Наочним прикладом для класифікації за алгоритмом дерев рішень є надання кредиту в банку. У банка є тисячі профілів інших людей, які вже брали кредит. Там вказано їх вік, освіту, посаду, рівень зарплати та головне - хто з них повернув кредит, а з ким виникли проблеми.

Ці дані можна використовувати як навчальну множину, на якій алгоритм навчають передбачати результат. Питання можуть бути не зовсім адекватними з точки зору людини, наприклад «зарплата позичальника більше, ніж 25000 гривень?», але машина придумує їх так, щоб на кожному кроці розділення було найточнішим. Так виходить дерево питань. Чим вище рівень, тим більш узагальнене питання (рис.3.4).



Рис.3.4. Наочний приклад дерева рішень

Припустимо, існує вибірка з тисяч записів, де кожний запис – це опис характеристик клієнта та відомості щодо повернення кредиту. При навчанні дерева використано фактори:

|  |  |
| --- | --- |
| * № паспорту * Прізвище, ім’я, по батькові * Адреса   Ці поля визначено як несуттєві | * Розмір кредиту * Термін кредиту * Мета кредиту * Місячний дохід * Місячні витрати * Основні витрати * Наявність власного житла |

Цільовим полем буде поле *Видати кредит*, що приймає значення Так / Ні.

Після побудови дерева отримується модель оцінювання кредитоспроможності клієнтів певного банку у вигляді ієрархічної структури правил.

Певні фактори можуть бути замінені одним узагальненим фактором. Наприклад, *Розмір кредиту, Термін кредиту, Середньомісячний дохід, Місячні витрати* – можуть не використовуватися, бо існує фактор *Кредит під заставу*, що є їх вдалим узагальненням.

Правила, за яким визначається належність клієнта до певної групи, записуються природньою мовою:

Якщо «Кредит під заставу» - Так і

«Термін проживання в цьому місті» > 19 років і

«Наявність нерухомості» - Так і

«Наявність банківського розрахунку» - Так

Тоді «Надати кредит» - Так

Достовірність = 98%

Правильно побудоване дерево має властивості до узагальнення, тобто, якщо виникає нова ситуація (новий клієнт), то ймовірно такі ситуації вже були і клієнт буде поводитися аналогічно, як клієнти з подібними характеристиками.

Але, проблема в тому, що банк не може повністю довіряти відповіді машини, без пояснень: може статися збій, втручання хакерів або ненавмисна зміна певного скрипту адміністратором. Тому, алгоритми дерев рішень, зазвичай можна використовувати як інструмент для прийняття рішень людиною.

Метод дерев рішень часто називають "наївним" підходом. Але завдяки цілому ряду переваг, даний метод є одним з найбільш популярних для вирішення задач класифікації.

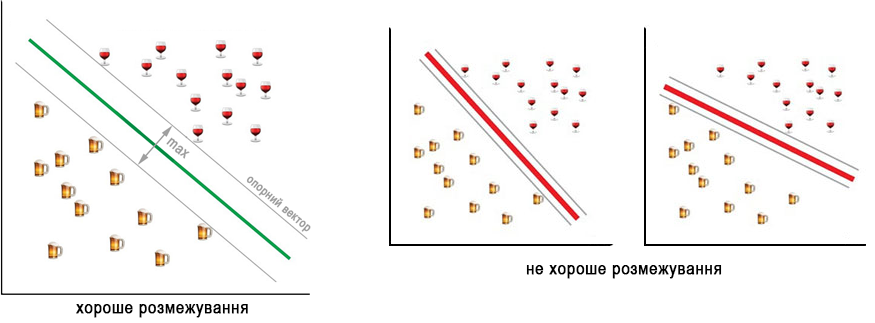
Дерева знайшли свою нішу в областях з високою відповідальністю: діагностиці, медицині, фінансах. У чистому вигляді дерева сьогодні використовують рідко, але їх комбінації лежать в основі складних систем і часто обробляють навіть результати від нейромережі. Наприклад, коли задається питання до пошукової системи, ранжуванням результатів займаються саме дерева рішень.

### Метод опорних векторів

Метод Опорних Векторів (Support Vector Machines, SVM) є популярним методом класичної класифікації та регресії.

Метод опорних векторів (Support Vector Machine, SVM) - це потужний алгоритм машинного навчання, який використовується для класифікації задач і регресії. Основна ідея SVM полягає у знаходженні гіперплощини (у багатовимірному просторі), яка максимально розділяє класи даних. Даний алгоритм має широке застосування на практиці і може вирішувати як лінійні так і нелінійні задачі.

Суть методу Опорних Векторів проста: SVM будує гіперплощину, яка максимально віддалена від найближчих точок навчальної вибірки різних класів. Ця гіперплощина називається максимальною роздільною гіперплощиною.



Опорними векторами називають точки даних, які знаходяться ближче до максимальної роздільної гіперплощини. Вони відіграють ключову роль у побудові гіперплощини та визначенні її становища. Гіперплощина - це n-1 мірна площина в n-вимірному Евклідовому просторі, яка поділяє простір на окремі частини.

Наприклад, складний набір даних про пацієнта клініки. Кожен пацієнт може бути описаний різними параметрами, такими як пульс, рівень холестерину, тиск тощо. Кожен з цих параметрів є виміром. SVM відображає ці параметри у багатовимірному просторі вищого виміру, а потім знаходить гіперплощину, щоб розділити класи.

Алгоритм SVM шукає опорні вектори, які розташовані безпосередньо до лінії розділення найближче. Обчислюється відстань між опорними векторами і роздільною лінією, так званий зазор. Для пошуку максимальної роздільної гіперплощини SVM вирішує задачу квадратичної оптимізації. Кращою вважається така лінія, для якої цей зазор є максимально великим (рис.3.6).

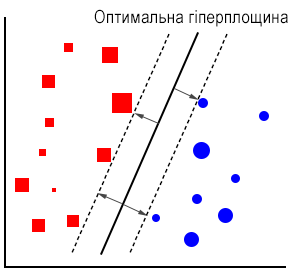


Рис.3.6. Визначення оптимальної роздільної лінії

Перевагами SVM є хороша узагальнююча здатність, здатність працювати з високорозмірними даними та здатність обробляти як бінарні, так і багатокласові завдання класифікації. Вони також добре працюють за умов обмеженої кількості даних.

До недоліків SVM можна віднести чутливість до шумів і викидів даних, а також складність вибору методу перетворення даних на більш високорозмірний простір (лінійне, поліноміальне та радіально-базисне функціональне).

SVM широко використовуються в різних галузях, включаючи обробку зображень, класифікацію тексту, спам-фільтри, біоінформатику та фінансовий аналіз завдяки їх здатності обробки складних завдань класифікації та регресії.

## 3.2. Регресія

Регресією називають апроксимацію даних з врахуванням їх статистичних параметрів. Таке завдання постає при обробці даних, отриманих в результаті вимірювань процесів або фізичних явищ. Завданням регресійного аналізу є підбір математичних формул, які найкращим чином можуть описати заданий набір.

Сьогодні використовують для:

* Прогнозу вартості цінних паперів.
* Аналізу попиту, обсягу продажів.
* Медичних діагнозів.
* Будь-які залежності числа від часу.

Популярні алгоритми: [Лінійна або Поліноміальна Регресія](http://www.machinelearning.ru/wiki/index.php?title=Линейная_регрессия_%28пример%29)

Регресія - та ж класифікація, тільки замість категорії передбачається число. Вартість автомобіля залежно від пробігу, кількість корків за часом доби, обсяг попиту на товар від зростання компанії тощо. Регресією ідеально вирішуються завдання, де є залежність від часу.

Математична постановка задачі регресії полягає в наступному. Залежність величини певної властивості об’єкту Y від іншої змінної властивості або параметра Х зареєстровано на множині точок множиною значень. В кожній точці зареєстровані значення відображено з випадковою похибкою. За сукупністю значень потрібно підібрати таку функцію, яка б з мінімальною похибкою відображала зареєстровані дані.

Види регресії називаються за типом апроксимуючих функцій: лінійна, поліноміальна, експоненціальна, логарифмічна. Якщо регресія формує пряму лінію - її називають лінійною, якщо криву – поліноміальною (рис. 3.7).

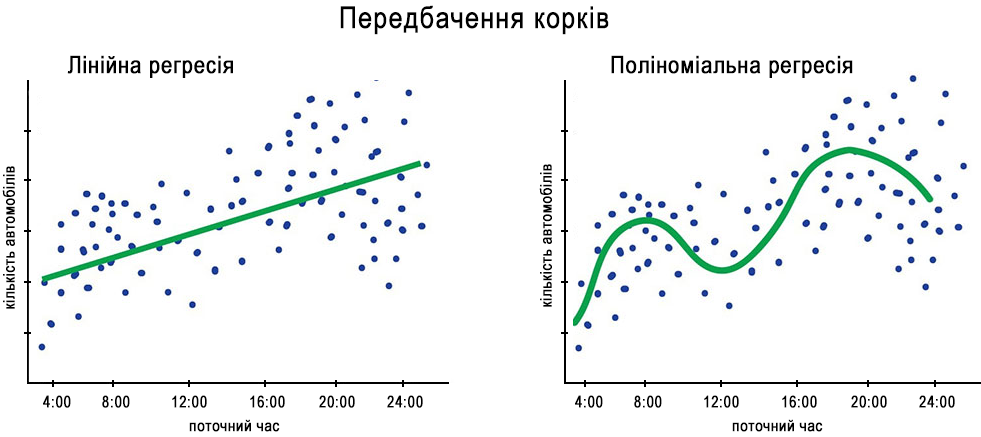


Рис. 3.7. Наочний приклад регресії

Вибірку даних, найчастіше, представляють у вигляді масиву, що складається з пар чисел (xi, yi). Тому, виникає завдання апроксимації дискретної залежності y(x) безперервною функцією f(x). Функція f(x), залежно від специфіки завдання, може відповідати різним вимогам:

* + f(x) повинна проходити через точки (xi, yi) , тобто f (хi) = уi , i = 1 ... n. В цьому випадку говорять про **інтерполяцію** даних функцією f (х) між точками хi, або **екстраполяцію** за межами інтервалу, що містить всі хi.
  + f(х) повинна певним чином (наприклад, у вигляді певної аналітичної залежності) наближати y (xi), не обов'язково проходячи через точки (xi, yi). Таку постановку завдання регресії в багатьох випадках можна назвати **згладжуванням** функції.
  + f(х) повинна наближати експериментальну залежність y (xi), враховуючи, що дані (xi, yi) отримано з деякою погрішністю, що виражає шумову компоненту вимірювань. При цьому функція f(х), за допомогою того чи іншого алгоритму, зменшує похибку, що присутня в даних (xi, yi). Такого типу задачі називають **фільтрацією**.

На рисунку проілюстровано різні види побудови апроксимуючої залежності f (х). Тут, вихідні дані позначено точками, інтерполяція - пунктиром, лінійна регресія (згладжування) - похилою прямою лінією, а фільтрація – грубою гладкою кривою (рис.3.8).

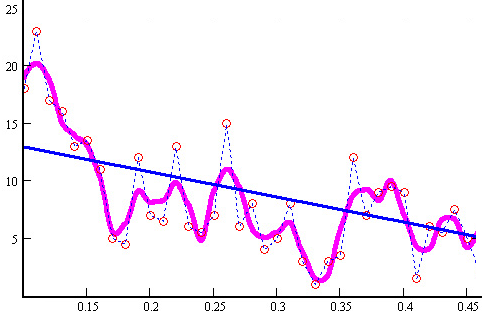


Рис.3.8. Різні види побудови апроксимуючої залежності f (х)

Для реалізації лінійної регресії часто використовують метод найменших квадратів. Тут вимірюється відстань по вертикалі від кожної точки до лінії. Необхідної лінією буде та конструкція, де сума відстаней буде мінімальною . Іншими словами, крива проводиться через точки, що мають нормально розподілене відхилення від істинного значення (рис.3.9).

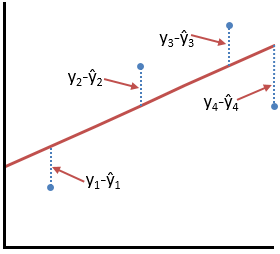


Рис.3.9. Наочний приклад методу найменших квадратів

Якщо лінійна функція може бути застосована для підбору даних, то метод найменших квадратів відноситься до типів метрики помилок, яка мінімізує похибки.

## 3.3. Ансамблі

Ансамблі (комбінації моделей) та нейромережі – головними лідерами на сьогодні, вони надають найточніші результати і використовуються всіма великими компаніями.

Ансамблеві методи - це парадигма машинного навчання, де кілька базових моделей («слабкі учні») можна використовувати як будівельні блоки для проектування більш складних моделей шляхом їх об'єднання. Вибір базових моделей може залежати від багатьох змінних задачі: кількості даних, розмірності простору, гіпотези розподілу.

Ідея ансамблевих методів полягає в тому, щоб спробувати зменшити похибки таких слабких учнів, об'єднуючи кілька з них разом, щоб створити сильного учня (або модель ансамблю), який досягає кращих результатів.

При формуванні ансамблю моделей необхідно вирішити три завдання:

* Вибрати базові моделі.
* Визначити підхід до використання навчальної множини.
* Вибрати метод комбінування результатів.

Оскільки ансамбль - це складна модель, що складається з окремих базових моделей, то при його формуванні можливі два випадки:

* Ансамбль складається з базових моделей одного типу, наприклад, тільки з дерев рішень, тільки з нейронних мереж тощо.
* Ансамбль складається з моделей різного типу - дерев рішень, нейронних мереж, регресійних моделей тощо.

При побудові ансамблю використовується навчальна множина, для використання якої існують два підходи:

* Підвибірки, тобто з існуючої навчальної множини витягується кілька підмножин, кожна з яких використовується для навчання однієї з моделей ансамблю.
* Використання одної навчальної множини для навчання всіх моделей ансамблю.

Для комбінування результатів, виданих окремими моделями, використовують три способи:

* Голосування - вибирається той результат, який був виданий простою більшістю моделей ансамблю.
* Зважене голосування - для моделей ансамблю встановлюються ваги, з врахуванням яких виноситься результат.
* Усереднення (зважене або незважене) - вихід всього ансамблю визначається як просте середнє значення виходів всіх моделей, при зваженому усередненні виходи всіх моделей множаться на відповідні ваги.

#### Основні ансамблеві методи

* **Беггінг (Bootstrap AGGregatING)**. Однорідні слабкі учні навчаються паралельно і незалежно один від одного, далі вони об'єднуються за певним процесом усереднення.
* **Бустінг (Boosting)**. Однорідні слабкі учні навчаються послідовно адаптивним способом (слабкий учень залежить від попередніх), далі вони об'єднуються відповідно до певної стратегії.
* **Стекінг (Stacking)**. Різнорідні слабкі учні навчаються паралельно і об'єднуються, навчаючи метамодель для виведення результату, заснованого на результатах різних слабких моделей.

### Беггінг

Основна ідея Bagging полягає в тому, щоб створити багато різних копій моделі навчання, навчити кожну з них на різних підмножинах даних і об'єднати (агрегувати) їх результати, щоб отримати більш стійке і точне передбачення.

Обирається базова модель навчання, наприклад, дерево рішень або метод опорних векторів, і створюється кілька копій цієї моделі. Для кожної копії моделі випадково вибирається підмножина даних з навчальної вибірки. Це означає, що кожна копія бачить лише частину даних.

Кожна копія моделі навчається на власній випадково обраній підмножині даних. Це робить їх різними. Після навчання всіх копій моделей їх результати комбінуються в різний спосіб: шляхом голосування (у завданнях класифікації) або усереднення (у завданнях регресії). Таким чином отримується остаточне передбачення.

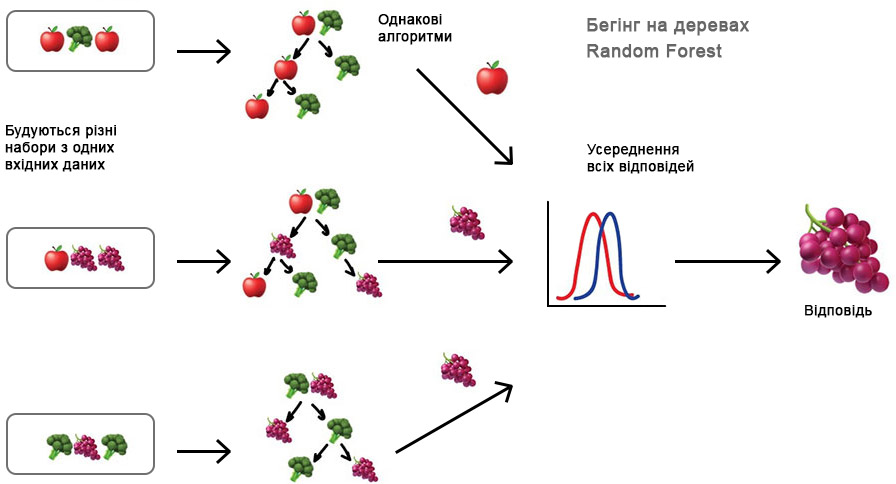


Рис.3.10 Наочне представлення беггінгу

Основна ідея Bagging полягає в тому, що моделі не виправляють помилки одна одної, а компенсують їх при формуванні кінцевого результату. Так вони покращують узагальнюючу здатність ансамблю. Це дозволяє знизити вплив викидів та шуму в даних та створити більш стабільне передбачення.

### Бустінг

Основна ідея бустінгу у машинному навчанні полягає в тому, щоб об'єднати декілька слабких моделей навчання для створення сильної та точної моделі. Це процедура послідовної побудови композиції алгоритмів машинного навчання, коли кожен наступний алгоритм прагне компенсувати недоліки композиції всіх попередніх алгоритмів. Алгоритми виконуються послідовно, кожен наступний приділяє особливу увагу тим випадкам, на яких помилився попередній.

Бустінг відноситься до ітераційних алгоритмів. Він вчиться розпізнавати приклади на межах класів. Кожному запису даних на кожній ітерації алгоритму привласнюється вага.

Перший класифікатор навчається на всіх прикладах з рівними вагами. На кожній наступній ітерації ваги розставляються відповідно до класифікованих прикладів, тобто ваги правильно класифікованих прикладів зменшуються, а неправильно класифікованих - збільшуються.

Вага даних оновлюється після кожної ітерації на основі того, наскільки добре модель передбачає їх. Це робить дані, які були неправильно класифіковані на попередніх ітераціях, важливішими на наступних ітераціях.

Отже, пріоритетними для наступного класифікатора стануть неправильно розпізнані приклади, навчаючись на яких новий класифікатор буде виправляти помилки класифікатора минулої ітерації.

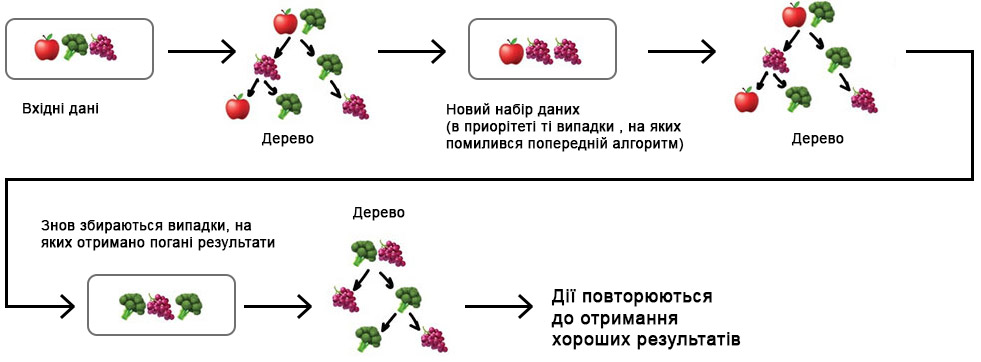


Рис.3.11. Наочне представлення бустінгу

Кінцевий результат ансамблю формується шляхом комбінування результатів всіх слабких моделей з врахуванням їх ваг.

У порівнянні з беггінгом бустінг є більш складною процедурою, але в багатьох випадках працює ефективніше. Значним плюсом є висока точність результатів, але мінусом є не паралельний процес, хоча цей алгоритм працює швидше ніж нейромережі.

### Стекінг

Основна ідея Stacking полягає в тому, щоб використовувати виходи кількох базових моделей як входи для більш високорівневої моделі, яка робить остаточне передбачення.

Обирається кілька різних базових моделей машинного навчання, таких як дерева рішень, метод опорних векторів, випадковий ліс тощо. Кожна з цих моделей навчається на вихідних даних. Кожна з базових моделей використовується для створення прогнозів на навчальній або тестовій вибірці. Ці прогнози стають новими ознаками даних.

Утворюється нова матриця даних, де рядки є зразками, а стовпці - прогнози, зроблені базовими моделями. Ця матриця даних використовується як вхід для метамоделі (стекінг моделі), яка навчається для виконання остаточного прогнозу. Метамодель може бути будь-якою моделлю машинного навчання, наприклад лінійною регресією, деревом рішень або випадковим лісом.

Метамодель навчається на прогнозах базових моделей та вихідних цільових змінних для створення фінального прогнозу. Це дозволяє врахувати думку всіх базових моделей та зробити більш точне передбачення.

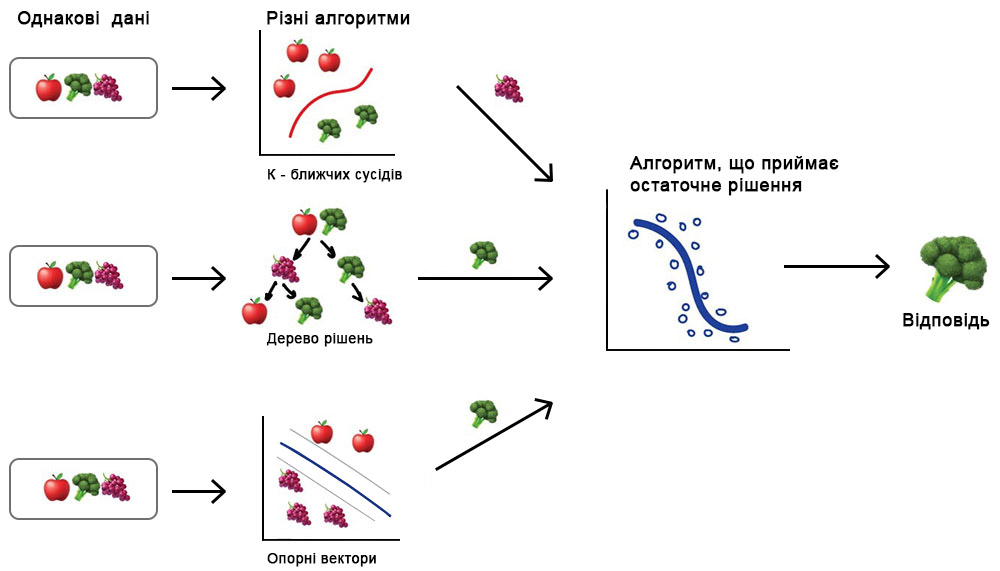


Рис.3.12. Наочне представлення стекінгу

Stacking дозволяє зменшити ризик перенавчання, оскільки моделі навчаються на різних частинах даних, і враховує результати кількох моделей, що призводить до більш точних результатів. Цей метод широко використовується у змаганнях з машинного навчання та для вирішення складних завдань прогнозування.

### Переваги ансамблевих методів

Ансамблеві методи в машинному навчанні мають ряд переваг, які роблять їх популярними та потужними:

* За рахунок об'єднання передбачень кількох моделей ансамблі часто досягають вищої точності, ніж окремі моделі.
* Ансамблеві методи зазвичай менш схильні до перенавчання, ніж окремі моделі. Різні моделі можуть робити різні помилки, і ансамбль їх усереднює.
* Ансамблеві методи можуть успішно працювати з різними видами даних, включаючи категоріальні, числові та текстові дані. Це робить їх універсальними для багатьох завдань.
* Ансамблі можуть справлятися з проблемами незбалансованих класів даних, що робить їх корисними для завдань класифікації, де один клас може бути менш представлений.
* Ансамбль моделей може надати більш інтерпретовані результати порівняно з окремими моделями. Наприклад, важливість ознак може бути більш явною, коли вона обчислюється на основі ансамблю.
* Ансамблі можуть використовуватись з різними базовими моделями та різними алгоритмами навчання. Це дозволяє вибирати найкращий набір моделей для конкретного завдання.
* Ансамблі зазвичай більш стійкі до змін даних і випадкових варіацій, таких як викиди.

Застосування ансамблевих методів може значно покращити результати у багатьох завданнях машинного навчання, і тому вони широко використовуються у практиці.

## Висновки

На ефективність алгоритмів впливає багато факторів і зокрема розмір та структура набору даних. З цієї причини доводиться пробувати багато алгоритмів, перевіряючи ефективність кожного на тестовому наборі даних, і потім вибирати кращий варіант. Зрозуміло, що потрібно вибирати серед алгоритмів, що відповідають поточній задачі.

Алгоритми машинного навчання можна описати як навчання цільової функції f, яка найкращим чином співвідносить вхідні змінні X і вихідну змінну Y: Y = f (X). Функція f є невідомою, тому для її знаходження використовують різні алгоритми.

Найбільш поширеним завданням в машинному навчанні є передбачення значень Y для нових значень X. Це називається прогностичним моделюванням, і метою є зробити якомога точне передбачення. На вибір алгоритму впливає багато факторів:

* Розмір, якість і характер даних.
* Доступний час для обчислень.
* Терміновість завдання.
* Кінцева мета обробки даних.

Навіть досвідчений фахівець не скаже, який алгоритм буде працювати краще, перш ніж спробує кілька варіантів. Існує багато класичних алгоритмів машинного навчання, але наведені в рамках цієї лекції – є найбільш популярними.

## Контрольні питання

1. Для яких типів завдань використовується навчання з вчителем?
2. Перелічіть алгоритми, які є популярними для вирішення задачі класифікації.
3. В чому полягає популярність використання наївного байєсіва класифікатора?
4. Назвіть основні терміни алгоритму дерева рішень.
5. Яка кінцева мета алгоритму опорних векторів?
6. Перелічіть типи регресій та види побудови апроксимуючої залежності.
7. Зазначте основну ідею методу ансамблів.
8. Які завдання потрібно вирішити при формуванні ансамблю моделей?
9. Яка модель для ансамблів є самою популярною?
10. В чому полягає різниця між беггінгом та бустінгом?

## Використані джерела

1. Введення в машинне навчання - <https://habr.com/ru/post/448892/>
2. Машинне навчання для людей <https://vas3k.ru/blog/machine_learning/>
3. Байєсів класифікатор - <http://crypto.pp.ua/2011/04/bajesovskij-klassifikator/>
4. Топ-10 Data Mining-алгоритмів - <https://habr.com/ru/company/iticapital/blog/262155/>
5. Огляд популярних алгоритмів машинного навчання - <https://tproger.ru/translations/top-machine-learning-algorithms/>
6. Ансамблеві методи: беггінг, бустінг і стекінг <https://neurohive.io/ru/osnovy-data-science/ansamblevye-metody-begging-busting-i-steking/>
7. Як працюють ансамблі <https://evergreens.com.ua/ru/articles/ensembles.html>
8. Ансамблевий метод машинного навчання, що базується на рекомендації класифікаторів <http://intsysjournal.ru/articles/is1904/02_kashnickiy.pdf>
9. Композиції: беггінг, випадковий ліс <https://habr.com/ru/company/ods/blog/324402/>
10. Історія розвитку ансамблевих методів класифікації в машинному навчанні <https://www.researchgate.net/publication/278019662_Istoria_razvitia_ansamblevyh_metodov_klassifikacii_v_masinnom_obucenii>