# Тема 4. Класичні алгоритми для навчання без вчителя

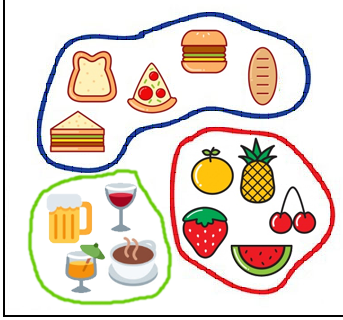
Машинне навчання без вчителя (Unsupervised Machine Learning) - це розділ машинного навчання, у якому моделі навчаються на даних, які не мають розмічених (навчальних) відповідей чи цілей. Основна мета машинного навчання без вчителя - виявлення закономірностей, структур і шаблонів даних без заздалегідь заданих правильних відповідей.

Типовими завдання для застосування навчання є

* Кластеризація схожих об’єктів за різними критеріями близькості
* Зменшення розмірності даних шляхом виявлення найважливіших ознак чи створення нових ознак.
* Пошук асоціативних правил даних, щоб визначити зв'язки між різними елементами.
* Генеративні моделі, які можуть генерувати нові дані, схожі на навчальні дані.

Машинне навчання без вчителя відіграє важливу роль в аналізі даних та витягування цінної інформації з великих обсягів даних. Цей вид навчання часто використовується на ранніх етапах дослідження даних для отримання розуміння про структуру і характеристики даних.

## 4.1. Кластеризація

 Кластеризація - це завдання машинного навчання без вчителя, що полягає у групуванні схожих об'єктів у кластери чи групи. Основна мета кластеризації - знайти структуру в даних, шляхом поділу їх на однорідніші групи, що може спростити аналіз даних і виділити внутрішні закономірності.

Приклади використання

* Сегментація ринку (типів покупців, лояльності) на основі їхньої купівельної поведінки.
* Об'єднання близьких точок на карті.
* Стиснення зображень.
* Аналіз і розмітка нових даних.
* Детектори аномальної поведінки.
* Кластеризація статей, новин, книжок відповідно до тематики

Основні алгоритми кластеризації: K-Means, DBSCAN, Ієрархічна кластеризація, Mean Shift, Spectral Clustering, Agglomerative Clustering, OPTICS. Кожен із цих алгоритмів має свої переваги та недоліки, і вибір алгоритму залежить від природи даних та завдання кластеризації.

Кластеризація - це класифікація, але без заздалегідь відомих класів. Машина сама шукає схожі об'єкти та об'єднує їх у кластери. Кількість кластерів можна задати заздалегідь або довірити це машині. Схожість об'єктів машина визначає за тими ознаками, які в них виявлено - в кого багато схожих характеристик, тих в один кластер.

Кластерний аналіз - це сімейство алгоритмів, розроблених для формування груп таким чином, щоб члени групи були найбільш схожими один на одного і не схожими на елементи, що не виходять в групу. Кластер і група - це синоніми в світі кластерного аналізу.

### Метод k-середніх

Популярним алгоритмом кластеризації є Метод К-середніх (K-Means). Метод к-середніх створює к-груп з набору об'єктів таким чином, щоб члени групи були найбільш однорідними. Це популярна техніка кластерного аналізу для дослідження набору даних.

Випадковим чином у скупченні даних зазначається певна кількість базових точок - центроїдів. Всі інші точки будуть відноситься до найближчого від них центроїду - виходять як би сузір'я. Далі центр ваги зсувається в центр свого сузір'я і дії повторюються, поки центроїди не перестануть пересуватися. Кластери виявлені, стабільні і їх рівно стільки, скільки було зазначено центроїдів (рис.4.1).

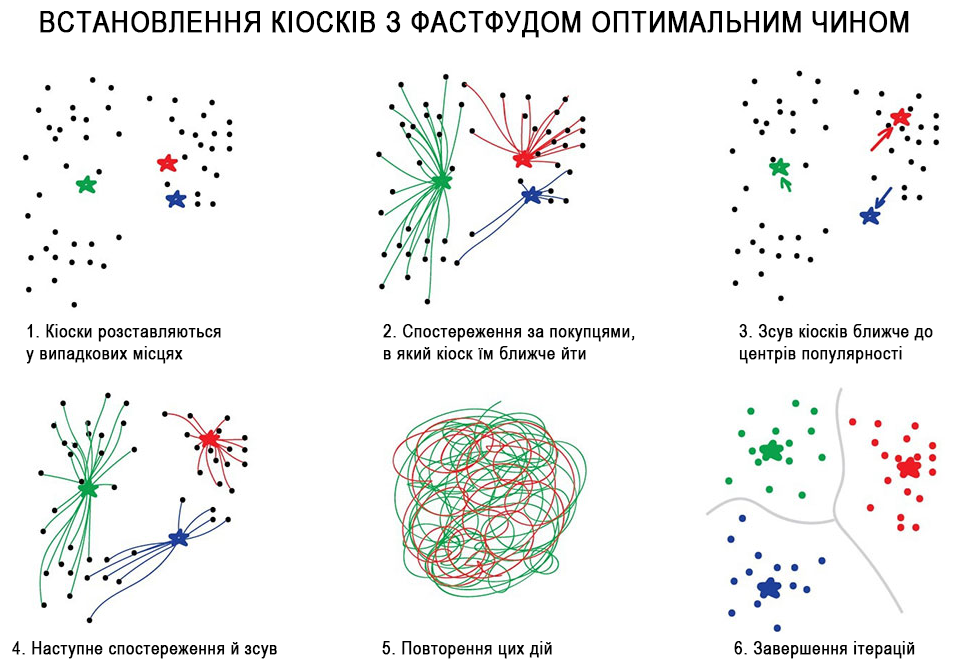


Рис. 4.1. Наочний приклад методу k-середніх

Як правило, метод к-середніх є самонавчальним. Замість того, щоб уточнювати кількість кластерів, метод к-середніх «вивчає» кластери самостійно, не вимагаючи інформації про те, до якого кластеру відносяться дані спостереження. Іноді, метод к-середніх може застосовувати етап попереднього навчання.

Основною перевагою алгоритму є його простота, висока швидкість виконання і ефективність у порівнянні з іншими алгоритмами, особливо при роботі з великими наборами даних.

Метод к-середніх може використовуватися для попереднього розділення на групи великого набору даних, після чого проводиться більш потужний кластерний аналіз підкластера. Метод к-середніх може використовуватися, щоб приблизно визначити кількість кластерів і перевірити наявність неврахованих даних і зв'язків в наборах.

До основних недоліків методу к-середніх відносять чутливість до «викидів» та початковий вибір центроїдів. Метод к-середніх створений для роботи з безперервними значеннями, тому доведеться виконати вдосконалення, щоб змусити алгоритм працювати з дискретними даними.

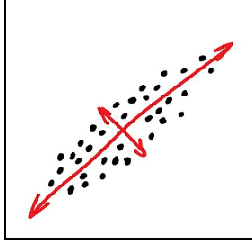
### Алгоритм просторової кластеризації DBSCAN

DBSCAN (Density-Based Spatial Clustering Of Applications With Noise, щільний алгоритм просторової кластеризації з присутністю шуму)**.**

Алгоритм самостійно знаходить скупчення точок і будує навколо кластери. Його легко зрозуміти, якщо уявити, що точки - це люди на площі. Нехай 3 людини, що знаходяться поруч беруться за руки. Далі вони починають брати за руку тих, до кого можуть дотягнутися. Так за ланцюжком, поки ніхто більше не зможе взяти когось за руку - це і буде перший кластер. Дії повторюються, поки це скупчення не поділиться на групи. Ті, кому взагалі нікого брати за руку - це викиди, аномалії.

Алгоритм оперує щільністю даних. На вхід подається матриця близькості і два параметри - радіус околу і кількість сусідів.

## 4.2. Зменшення Розмірності (Узагальнення)

Зменшення розмірності даних – це процес зменшення кількості ознак (вимірювань) у наборах даних. Цей процес може бути корисним для покращення продуктивності моделей машинного навчання, зменшення обчислювальної складності та боротьби з прокляттям розмірності..

Використовуються для:

* Рекомендаційні Системи.
* Красиві візуалізації.
* Визначення тематики та пошук схожих документів.
* [Аналіз фейкових зображень](https://vas3k.ru/blog/390/)
* Ризик-менеджмент.

#### Популярні методи зменшення розмірності даних

**Метод головних компонентів (Principal Component Analysis, PCA)** є одним із найпопулярніших методів зменшення розмірності. Він знаходить нові ознаки (головні компоненти), які є лінійними комбінаціями вихідних ознак і зберігають при цьому найбільшу дисперсію даних. Головні компоненти впорядковані за часткою поясненої дисперсії, і можна вибрати, скільки залишити.

Найбільша дисперсія даних - це міра розкиду чи змінності значень набору даних. Обчислення дисперсії дозволяє визначити, наскільки дані розкидані навколо середнього значення. Якщо дисперсія велика, це означає, що значення даних дуже різноманітні і можуть відрізнятися від середнього значення. Якщо дисперсія мала, дані мають менший розкид і ближче до середнього значення.

Дисперсія є важливою статистичною метрикою, яка використовується у багатьох аспектах аналізу даних, включаючи машинне навчання. Вона допоможе визначити, наскільки добре дані описують розподіл значень і які закономірності можна витягнути з даних.

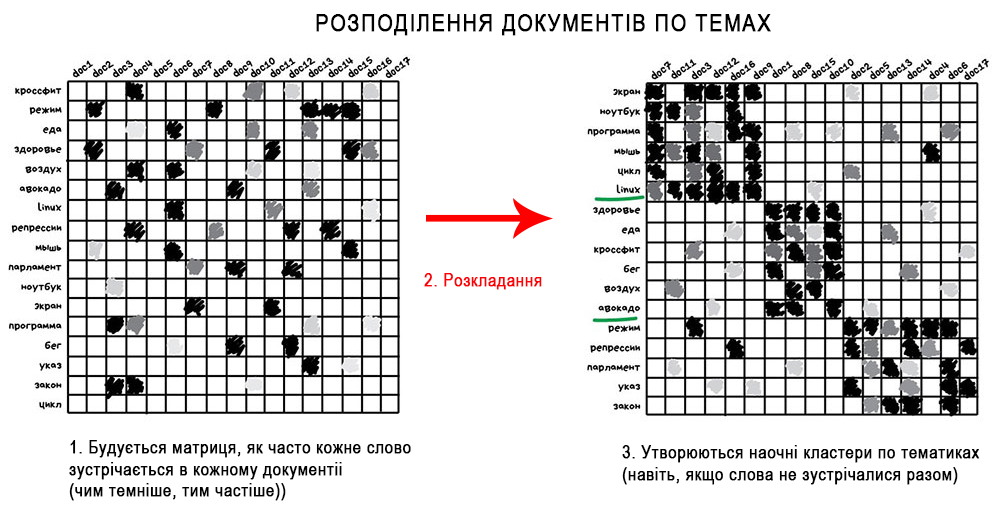


Рис. 4.5. Наочний приклад методу головних компонентів

**t-SNE (t-distributed Stochastic Neighbor Embedding)** - це метод візуалізації та зменшення розмірності, який дозволяє проектувати дані у простір меншої розмірності, зберігаючи при цьому відносні відстані між точками. Він часто використовується для візуалізації даних високої розмірності.

**LLE (Locally Linear Embedding)** - це спосіб зменшення розмірності, який прагне зберегти локальні структури даних. Він знаходить лінійні відносини між сусідніми точками та використовує їх для створення нового подання даних.

**Інкрементне зменшення розмірності** будує модель машинного навчання і поступово зменшує розмірність даних, видаляючи найменш інформативні ознаки. Це може бути корисним для пошуку найкращої розмірності в залежності від задачі.

**LDA (Linear Discriminant Analysis)** – це метод зменшення розмірності, який використовується у задачах класифікації. Він знаходить нові ознаки, які максимізують поділ класів даних.

**ICA (Independent Component Analysis)** – це метод, який прагне знайти незалежні компоненти даних. Він може бути корисним, якщо дані містять змішані сигнали або джерела.

**Sparse Coding: Sparse Coding** - це метод, який знаходить розріджені представлення даних у новому просторі ознак. Це може допомогти виділити найінформативніші ознаки.

Вибір методу зменшення розмірності залежить від конкретної задачі та структури даних. Кожен із цих методів має свої власні особливості та призначений для певних сценаріїв використання.

Популярне застосування методів зменшення розмірності знайшли в рекомендаційних системах. Виявилося, якщо абстрагувати ним оцінки фільмів, виходить непогана система рекомендацій кіно, музики, ігор тощо.

Отримані абстракції було важко зрозуміти, але коли дослідники почали пильно розглядати нові ознаки, вони виявили, що деякі з них явно корелюють з віком користувача (діти частіше грали в майнкрафт і дивилися мультфільми), інші з певними жанрами кіно, а треті взагалі з синдромом пошуку глибокого сенсу. Машина, яка не знала нічого окрім оцінок користувачів, змогла дістатися до таких високих матерій, навіть не розуміючи їх.

## 4.3. Пошук асоціативних правил

Метод аналізу даних, який використовується виявлення структурних зв'язків між різними елементами чи об'єктами в наборі даних. Основна мета пошуку асоціативних правил - виявити цікаві і найпоширеніші патерни або асоціації в даних..

Сьогодні використовують для:

* Прогноз акцій і розпродажів.
* Аналіз товарів, що купуються разом.
* Розміщення товарів на полицях.
* Аналіз шаблонів поведінки на веб-сайтах.

Цей метод найчастіше застосовується у галузі аналізу даних, інтелектуального аналізу даних та маркетингового дослідження.

Першим застосуванням асоціативних правил було знаходження типових шаблонів покупок (здебільшого у супермаркетах) для аналізу купівельного кошику (Market Basket Analysis). Алгоритми виявлення асоціацій знаходять правила про окремі предмети, що з'являються разом в одній економічній операції, наприклад в одній покупці.

#### Основні поняття та терміни, пов'язані з пошуком асоціативних правил:

* **Транзакція** - це один запис або набір даних, який є список елементів або об'єктів. Наприклад, у роздрібній торгівлі транзакцією може бути чек із покупками.
* **Підмножина елементів (предметів)** - кожна транзакція містить певні елементи (наприклад, продукти в чеку).
* **Асоціативне правило** - це вираз, який пов'язує деякі елементи даних і передбачає, що якщо деякі елементи, що присутні в транзакції, то з ймовірністю буде присутній і інший елемент.
* **Асоціація** записується як **«А->Б»**, де А називається лівою частиною або передумовою, Б - правою частиною або наслідком. Приклад асоціативного правила: "Якщо покупець купив хліб та молоко, то з великою ймовірністю він також купить яйця."

#### Асоціативні правила мають числові характеристики

1. ***Підтримка правила (Support)*** – відсоткове відношення кількості транзакцій, що містять всі товари, які є у правилі до загальної кількості транзакцій



Низький рівень підтримки (менше за одну тисячну відсотка) свідчить, що така асоціація не істотною.

2. ***Достовірність правила (Confidence)*** – це відсоткове відношення кількості транзакцій, які містять всі товари правила до кількості транзакцій, які містять товари, що є в умові.



Ця величина показує як часто з появою А з'являється Б и розраховується як відношення частоти покупок А і Б разом до покупки А. Тобто якщо достовірність А до Б дорівнює 20%, то це означає, що при покупці товару А в кожному п'ятому випадку придбають і товар Б.

Достовірність А до Б не дорівнює достовірності Б к А. Справді, покупка комп'ютера частіше призводить до покупки мишки, ніж покупка мишки до покупки комп'ютера.

Алгоритми пошуку асоціативних правил призначені для знаходження всіх правил А→Б, причому підтримка і достовірність таких правил має бути вищою за певних визначених порогів, що відповідно називають *мінімальною підтримкою* та *мінімальною достовірністю*.

Ці значення обираються, щоб обмежити кількість знайдених правил. Якщо підтримка має велике значення, то алгоритми будуть знаходити правила, які є добре відомі аналітикам або є настільки очевидними, що немає сенсу проводити аналіз. Мале значення підтримки призводить до генерації величезної кількості правил, що потребує суттєвих обчислювальних ресурсів.

Зменшення мінімальної підтримки призводить до збільшення кількості потенційно цікавих правил. Але занадто мала підтримка правила робить його статистично необґрунтованим.

3. ***Потужність асоціації (Power)****.* Чим більше потужність, тим сильніше вплив, який поява А робить на появу Б. Потужність розраховується за формулою:



Алгоритми пошуку асоціацій можуть створювати свої бази даних підтримки, достовірності і потужності, до яких можна звертатися при запиті. Наприклад: "*Знайти всі асоціації, в яких для товару Х достовірність більш 50% і підтримка не менш 2,5%*"

Але, в такому аналізі є слабкі місця: випадково може виникнути ситуація, коли товари в супермаркеті будуть згруповані за допомогою знайдених моделей, але замість очікуваного прибутку спостерігається зворотний ефект. Це може відбутись через те, що клієнт швидко здійснить покупку і не буде блукати магазином, щоб прикупити ще щось.

За допомогою алгоритмів пошуку асоціативних правил вирішують широкий спектр практичних задач.

Розглянемо задачу, яка має за мету стимулювання продажів, що базується на знаннях про типову поведінку клієнта при оформленні замовлень. Застосування асоціативних правил надає можливість передбачити, що скоріше придбає клієнт і запропонувати йому саме цей товар.

Існує фірма «Гуртівня». Кількість позицій товарів є 1000. Клієнтами фірми є дрібно гуртові фірми, або компанії, що торгують в роздріб. Типовий процес закупки товарів у фірмі «Гуртівня»:

* Приходить клієнт з сумою 10 000 грн.
* Йому надають прас-лист.
* Клієнт замовляє визначені товари. Менеджер «Гуртівні» вводить обрані товари в електронну накладну.
* На складі цього товару є лише на 8 000 грн. 2 000 грн. клієнт також може витратити у «Гуртівні».

Оскільки менеджер зацікавлений у тому, щоб 2 000 грн. було витрачено , клієнтові потрібно запропонувати на вибір інший товар. Оскільки, кількість позицій є великою, то скоріше за все менеджер запропонує купити товари, що є лідерами продажів, або ті товари, що розташовані поруч у прас-листі.

Менеджер не завжди може запропонувати власне той товар, що клієнт (свідомо чи несвідомо) хоче купити. І переважно, товар, який можна було б продати лишається на складі.

Задачею менеджера «Гуртівні» є передбачити бажання клієнта та підказати клієнту про товари, що можуть гідно замінити ті товари, яких зараз немає на складі. Така задача є складною і для досвідченого менеджера, оскільки позицій товару є багато і певні закономірності не зовсім очевидні.

На допомогу менеджеру стане система, яка визначить перелік товарів, які може купити клієнт, як доповнення до тих, що вже він купив в цій транзакції. Така система може бути складовою інформаційної системи оформлення замовлень. Обчислення правил відбувається у фоновому режимі, тобто менеджер цього не бачить. Однак, за запитом йому надаються обчислені правила, що надає менеджеру вагомі аргументи.

Обчислені правила можна використовувати як для оперативної підказки так і для аналізу. Вони мають бути зручними і зрозумілими для кінцевого користувача (менеджера).

Пошук асоціативних правил може використовуватись у різних галузях, включаючи роздрібний маркетинг (для рекомендацій товарів), медичний аналіз даних (для виявлення асоціацій між симптомами та захворюваннями) та інші.

#### Існує кілька варіантів відображення асоціативних правил

1. У вигляді звичайної таблиці.
2. У вигляді форматованого тексту.
3. У вигляді дерева.
4. У вигляді перехресної таблиці.

В кожному варіанті є переваги та недоліки, тому, конкретний вибір залежить від особливостей особистого сприйняття або від кінцевої мети.

### Пошук послідовних шаблонів

При аналізі асоціацій часто викликає інтерес послідовність подій, що відбуваються. Тут додається змінна часу, що дозволяє працювати із серією подій для знаходження послідовних асоціацій протягом деякого періоду часу. При виявленні закономірностей в таких послідовностях можна з деякою часткою ймовірності передбачати появу подій у майбутньому, що дозволяє приймати правильні рішення. Таке завдання є різновидом задачі пошуку асоціацій і називається пошуком послідовних шаблонів.

Пошук послідовних шаблонів (Sequential Pattern Mining) - це методи аналізу даних, які використовуються для виявлення послідовних патернів, що часто зустрічаються, в послідовних даних, таких як тимчасові ряди, логи дій користувачів, транзакції, події та інші дані, які мають тимчасову послідовність. Ці методи дозволяють виявити цікаві та інформативні послідовності, які можуть бути корисними для розуміння поведінки, прогнозування майбутніх подій та прийняття рішень.

**Популярні методи та алгоритми для пошуку послідовних шаблонів:** AprioriAll, GSP (Generalized Sequential Pattern), SPADE (Sequential PAttern Discovery using Equivalence classes), PrefixSpan.

#### Ключові аспекти та терміни, пов'язані з пошуком послідовних шаблонів:

**Послідовні дані (Sequential Data)** - це дані, в яких елементи або події, що впорядковані за часом або будь-якою іншою послідовністю. Прикладами можуть бути історія замовлень клієнтів, записи тимчасових рядів, наслідки після пересування об'єктів та інші.

**Послідовні шаблони (Sequential Patterns)** - це часті та інформативні послідовності подій чи елементів у послідовних даних. Наприклад, в аналізі веб-логів це можуть бути послідовності дій користувача на сайті (наприклад, "пошук -> перегляд товару -> додавання до кошика -> оформлення замовлення").

**Підтримка (Support)** - це міра, яка визначає, як часто конкретний послідовний шаблон зустрічається в даних. Вона вказує, скільки разів ця послідовність спостерігається і може використовуватися для фільтрації шаблонів.

**Довжина шаблону (Pattern Length)** - це кількість елементів або подій у послідовному шаблоні. Чим довше шаблон, тим складніші та інформативні послідовності можна виявити.

Методи пошуку послідовних шаблонів широко застосовується в різних областях, таких як аналіз поведінки користувача в інтернеті, виявлення аномалій, прогнозування часових рядів та інші, де важлива облік послідовності подій або даних.

## Висновки

Машинне навчання без вчителя (Unsupervised Learning) - це підрозділ машинного навчання, у якому алгоритми навчаються з неструктурованих даних без явної розмітки чи нагляду.

#### Переваги та перспективи машинного навчання без вчителя:

* Методи машинного навчання без вчителя можуть автоматично виявляти приховані шаблони та структури даних. Це дозволяє досліджувати невідомі залежності та відкриває нові шляхи для аналізу даних.
* Машинне навчання без вчителя дозволяє групувати дані в кластери на основі їхньої подібності. Це може бути корисним для сегментації користувачів, виявлення аномалій або класифікації даних.
* Методи зменшення розмірності дозволяють зменшити розмірність даних, зберігаючи у своїй важливі показники. Це може бути корисним для візуалізації даних, видалення шуму або прискорення обробки даних.
* Машинне навчання без вчителя може використовуватися для створення нових даних на основі існуючих прикладів. Це може бути корисним, наприклад, для створення синтетичних даних для навчання моделей або розширення наборів даних.
* Методи машинного навчання без вчителя можуть автоматично отримувати релевантні ознаки даних. Це спрощує процес підготовки даних та може покращити продуктивність моделей машинного навчання.

В перспективах машинного навчання без вчителя є розвиток нових алгоритмів і методів, покращення ефективності та масштабованості, а також використання комбінації методів навчання з вчителем та без вчителя для досягнення кращих результатів.

Однак, незважаючи на переваги та перспективи, машинне навчання без вчителя стикається з низкою викликів, таких як складність інтерпретації результатів, проблеми з визначенням оптимальної кількості кластерів або розмірності даних, а також проблеми з обробкою великих обсягів даних.

В цілому, машинне навчання без вчителя є потужним інструментом для виявлення структур та патернів у даних, і його розвиток та застосування залишаються активною областю досліджень та інновацій.

# Навчання з підкріпленням

Навчання з підкріпленням (Reinforcement Learning, RL) - один із способів машинного навчання, в ході якого певна система (агент) навчається приймати рішення, взаємодіючи з деяким середовищем, щоб максимізувати певну нагороду. Тут немає ні вчителів, ні міток, які вказують, які відповіді правильні, які патерни агенту потрібно вивчати, що є хорошим або поганим результатом.

Цей підхід натхненний вивченням того, як живі організми навчаються на основі досвіду та спроб і помилок.

#### Основні компоненти та поняття у навчанні з підкріпленням:

* **Агент (Agent)** - сутність, яка знаходиться в середовищі та приймає рішення. Агент має набір дій, які може виконати, і метою максимізувати деяку нагороду, визначену кожної послідовності дій.
* **Середовище (Environment)** - навколишній простір, у якому діє агент. Середовище може бути фізичним (наприклад, робот або ігрове поле) або віртуальним (наприклад, комп'ютерна гра або симуляція).
* **Дії (Actions)** - набір можливих кроків, які агент може зробити у стані середовища. Вибір дій важливий, оскільки впливає майбутнє стан середовища проживання і, отже, на майбутні нагороди.
* **Стан (State)** - опис поточного стану середовища, в якому знаходиться агент. Стан є вхідною інформацією, яку агент використовує для вибору дій.
* **Нагорода (Reward)** - числова оцінка, яка вказує, наскільки добре агент виконав послідовність дій. Мета агента – максимізувати кумулятивну нагороду за тривалий час.
* **Функція корисності (Value Function)** - функція, яка оцінює очікувану суму нагород, яку агент може отримати, починаючи з певного стану та застосовуючи певні дії. Функція корисності допомагає агенту вибирати найкращі дії задля досягнення мети.
* **Політика (Policy)** - стратегія чи набір правил, якими агент вибирає дії у певних станах з метою максимізації нагороди.

Навчання з підкріпленням передбачає динамічну взаємодію агента з середовищем багаторазову взаємодію агента з середовищем, накопичення досвіду і оновлення стратегії (політики) на основі цього досвіду з метою поліпшення продуктивності агента. Агент має виконувати деякі дії з певного набору. Середовище змінюється в результаті виконання цієї дії, і агент сприймає новий стан середовища. Один з елементів цього стану - деяке число, яке представляє собою винагороду. Ця винагорода свідчить про те, чи правильно агент виконує ті чи інші дії в цьому середовищі (рис.4.6).



Рис.4.6. Навчання з підкріпленням

* На кожному кроці агент може знаходитися в стані s ∈ S.
* На кожному кроці агент вибирає з наявного набору дій деяку дію a ∈ A.
* Середовище повідомляє агенту, яку нагороду r він за це отримав і в якому стані s’ після цього опинився.

Таке навчання є окремим випадком навчання з учителем, але учителем є середовище або її модель. Наочно різницю між навчанням з вчителем і навчанням з підкріпленням можна продемонструвати на рис. 4.7.

|  |  |
| --- | --- |
|  |  |
| Рис.4.7а. Навчання з вчителем | Рис.4.7б. Навчання з підкріпленням |

При навчанні з вчителем система навчається на різних правильних прикладах, і в подальшому застосовує придбаний досвід в реальному середовищі. При навчанні з підкріпленням процеси навчання і використання йдуть паралельно, тобто система постійно корегує свою поведінку відповідно до змінних станів середовища.

Навчання з підкріпленням є потужним інструментом для вирішення завдань, в яких необхідно приймати послідовні рішення з урахуванням впливу дій на майбутній стан середовища та отримання нагороди.

Цей процес може бути застосований до різноманітних завдань, включаючи ігри, керування роботами, автономну навігацію, керування ресурсами та інші.

Головною перевагою цього типу навчання є порівняна простота: спостерігаються дії агента і в залежності від результату його заохочують або штрафують, не пояснюючи, як саме потрібно діяти. Роль вчителя може грати зовнішнє середовище. В даному підході значної уваги приділяють заохоченню/штрафуванню не лише поточних дій, які безпосередньо призвели до позитивного/негативного результату, але і тих дій, які передували поточним.

Після визначення функції винагороди, потрібно визначити алгоритм, який буде використовуватися для знаходження стратегії, що забезпечує найкращий результат.

Наївний підхід до вирішення цього завдання має наступні кроки:

1. Випробувати всі можливі стратегії.
2. Вибрати стратегію з найбільш очікуваною винагородою.

Перша проблема такого підходу полягає в тому, що кількість доступних стратегій може бути дуже значною або нескінченною. Друга проблема виникає, якщо винагороди є стохастичними - щоб точно оцінити винагороду від кожної стратегії потрібно буде багато разів застосувати кожну з них. Цих проблем можна уникнути, якщо допустити деяку структуризацію і, можливо, дозволити результатам, отриманими від застосування однієї стратегії, впливати на оцінку для іншої. Двома основними підходами для реалізації цих ідей є оцінка функцій корисності і пряма оптимізація стратегій.

Метою агента є максимізація сумарної винагороди, яку він отримує в результаті досить довгої взаємодії в цьому середовищі. У навчанні з підкріпленням невеликий проміжок дії агента прийнято називати епізодом. Він може закінчуватися термінальним (проміжним) станом або цільовим (кінцевим) станом, коли агент досяг кінцевого результату.

До прикладу можна розглянути робота (агента), що пересувається в деяких приміщеннях: може їхати вперед, назад і відкривати двері в сусідні кімнати. Кінцевою метою є прибуття в певну кімнату, щоб взяти конкретну річ. Винагородою служить, наприклад, відстань до предмета в цій кімнаті: чим ближче робот є до предмету, тим більше йому надходить балів нагороди. Якщо він далеко, це 5 балів, якщо він близько, це 10 або 20.

Завдання агента - виконати такі дії, щоб якомога швидше заробити більше балів. Нагородою за успішно виконане завдання є набрані в процесі вирішення бали та можливість виконувати нове завдання. Чим ефективніше виконано завдання, тим більше нараховується балів. Винагороди можуть бути від’ємними, і це значить, що агента штрафують, якщо він, наприклад, просувається не в ту сторону.

Такі завдання сильно відрізняються від класичних задач машинного навчання, оскільки агенту явно не повідомляють, що він повинен робити правильно. Агент має знайти золоту середину між дослідженням середовища і максимально швидким зароблянням балів. Спочатку він нічого не знає про існуючі середовища - кімнати. Агент має накопичити пам'ять, а потім її використовувати, щоб здійснювати відповідні дії. Це одна з основних задач, яку потрібно подолати в навчанні з підкріпленням: якнайшвидше набрати досвіду, а потім його використовувати для вирішення завдання.

1. Перша мета робота в навчанні з підкріпленням - мінімізувати помилки. Машина вчиться аналізувати інформацію перед кожним наступним ходом. Наприклад, безпілотний автомобіль під час навчання вчиться вчасно реагувати на сигнал світлофора, зупинятися перед пішоходом на переході, пропустити автомобіль, що швидко рухається або спецтранспорт справа. Щоб досягти кращого результату, машина навчається в віртуальної моделі міста з випадковими пішоходами та іншими учасниками дорожнього руху.
2. Друга мета робота в навчанні з підкріпленням - отримати від виконання завдання максимальну вигоду. Сама вигода при цьому повинна бути запрограмована заздалегідь: максимально швидкий час проходження маршруту, оптимальне витрачання ресурсів підприємства, обслуговування якомога більшої кількості відвідувачів.

Один із найпопулярніших алгоритмів машинного навчання з підкріпленням - це Q-навчання (Q-learning). Q-навчання засноване на оцінці функції Q, яка представляє очікувану сумарну нагороду, яку агент отримує, вибираючи певну дію у певному стані. Алгоритм Q-навчання оновлює значення функції Q на основі отриманих нагород і дозволяє агенту поступово покращувати свою поведінку та вибирати оптимальні дії.

Існують інші підходи до машинного навчання з підкріпленням, такі як актор-критик методи, де актор відповідає за вибір дій, а критик оцінює ефективність обраних дій. Також існують алгоритми, що ґрунтуються на ідеях марківських процесів прийняття рішень (Markov Decision Processes) та глибокого навчання (Deep Reinforcement Learning), які застосовують нейронні мережі для навчання агента.

Машинне навчання з підкріпленням широко застосовується у різних галузях, включаючи ігрові програми, робототехніку, управління автономними системами та фінансовий аналіз. Воно дозволяє агентам вчитися адаптуватися до складних та непередбачуваних середовищ і знаходити оптимальні стратегії поведінки.

Області практичного застосування

* Постановка цілей.
* Планування.
* Управління ресурсами підприємств.
* Системи сприйняття.
* Боти для комп'ютерних ігор.
* Чат-боти, які навчаються від діалогу до діалогу.
* Самокеровані автомобілі.
* Роботи-порохотягі.

## Висновки

Навчання з підкріпленням застосовується там, де потрібно узгодити відтерміновану вигоду - мету - з ситуативним прийняттям рішення. Цей вид навчання вирішує складне завдання співвіднесення негайних дій з відстроченою віддачею, яку вони виробляють. Як і людям, алгоритмам підкріплення навчання іноді доводиться чекати, щоб побачити результати своїх рішень. В таких випадках часто складно зрозуміти, яка дія не призводить до жодних результатів.

Машина намагається вирішити задачу різними способами, помиляється, вчиться на своїх помилках, покращує показники.

Навчання з підкріпленням застосовується в тих випадках, коли потрібно вибрати кращий варіант серед багатьох або досягти складної мети за багато ходів. Там, де потрібно навчити машину виживати в реальному середовищі.

## Контрольні питання

1. Яка загальна ідея методу к-середніх та алгоритму просторової кластеризації?
2. Перелічіть сфери, де застосування асоціативних правил буде ефективним.
3. Що собою представляють характеристики асоціативних правил: підтримка, достовірність та потужність?
4. Чим пошук послідовності подій різниться від пошуку асоціативних правил?
5. Назвати основну ідею навчання з підкріпленням.
6. Для яких завдань можна застосовувати навчання з підкріпленням?
7. Назвати основні цілі, які ставляться до робота, що навчається з підкріпленням.
8. Пояснити різницю між навчанням з вчителем та навчанням з підкріпленням
9. За які дії робот отримує нагороду або штрафується?

## Використані джерела

1. Алгоритм кластеризації DBSCAN - <https://habr.com/ru/post/322034/>
2. Best Reinforcement Learning Tutorials, Examples, Projects, and Courses - <https://neptune.ai/blog/best-reinforcement-learning-tutorials-examples-projects-and-courses>
3. ConvNetJS Deep Q Learning Demo - <https://cs.stanford.edu/people/karpathy/convnetjs/demo/rldemo.html>
4. Навчання з підкріпленням для початківців - <https://proglib.io/p/reinforcement-learning/>
5. Навчання з підкріпленням в машинному навчанні - <https://evergreens.com.ua/ru/articles/reinforcement-learning.html>
6. Навчання нейромережі з вчителем, без вчителя, з підкріпленням - <https://neurohive.io/ru/osnovy-data-science/obuchenie-s-uchitelem-bez-uchitelja-s-podkrepleniem/>
7. 11 must-have алгоритмов машинного обучения для Data Scientist - <https://proglib.io/p/11-ml-algorithms/>