Міністерство освіти і науки України
Рівненський державний гуманітарний університет
Кафедра інформаційних технологій та моделювання

Кваліфікаційна робота

за освітнім ступенем «бакалавр»

на тему:

АДАПТАЦІЯ АЛГОРИТМІВ МАШИННОГО НАВЧАННЯ БЕЗ УЧИТЕЛЯ ДЛЯ КЛАСТЕРИЗАЦІЇ НЕЧІТКО ВИЗНАЧЕНИХ ОБРАЗІВ

Виконав:
здобувач IV курсу
групи КН-41
спеціальності 122 «Комп'ютерні науки»
Банацький-Шуманський Максим
Валентинович

Науковий керівник:
к.т.н., доцент Сяський В.А.

Рівне – 2025

ЗМІСТ

ВСТУП

При розробці систем штучного інтелекту знання про конкретну предметну область рідко бувають повними й абсолютно достовірними. Навіть кількісні дані, отримані шляхом досить точних експериментів, мають статистичні оцінки вірогідності, надійності, значимості і т.д. Поряд із кількісними характеристиками в базах знань інтелектуальних систем повинні зберігатися якісні показники, евристичні правила, текстові знання тощо. При обробці знань із застосуванням строгих (чітких) механізмів формальної логіки виникає протиріччя між нечіткими знаннями і чіткими методами логічного виведення.

Це дослідження спрямоване на подолання обмежень традиційних методів кластеризації при роботі з нечіткими даними, що дозволить покращити якість інтелектуального аналізу, забезпечивши більш точне моделювання реальних явищ з неоднозначними або розмитими межами.

Актуальність дослідження обумовлена необхідністю розробки методів, які можуть адаптуватися до невизначеності та варіативності інформації, що є ключовим фактором для забезпечення надійності роботи інтелектуальних систем. Одним із перспективних підходів для роботи з такими даними є використання нечітких множин та алгоритмів кластеризації, зокрема алгоритмів на основі нейронних мереж Кохонена.

Такі методи особливо актуальні в умовах, коли традиційні підходи на основі чітких даних виявляються малоефективними. Зокрема, це стосується сфер, де інформація є неповною, неоднозначною або отриманою на основі експертних оцінок. До таких сфер належать: медична діагностика, прогнозування ризиків, аналіз споживчих уподобань, управління складними технічними системами, інтелектуальний аналіз даних, робототехніка, а також системи підтримки прийняття рішень. Використання нечітких множин у поєднанні з нейронними мережами дозволяє моделювати складні залежності та забезпечувати адаптивність моделей до нових або змінних умов.

Мета дослідження є модифікація алгоритму кластеризації на основі нейронних мереж Кохонена для вирішення проблеми групування нечітких даних.

Завдання дослідження:

- проаналізувати сучасні методи кластеризації даних, зокрема методи, що враховують нечіткість та ймовірнісну приналежність даних до кластерів;
- дослідити особливості нейронних мереж Кохонена та можливості їх адаптації для роботи з нечіткими даними;
- розробити модифікований алгоритм кластеризації на основі нейронної мережі Кохонена, що враховує ймовірності приналежності до кластерів для дискретного та неперервного просторів ознак;
- реалізувати запропонований алгоритм кластеризації в програмному середовищі та провести тестування на вибраних наборах нечітких даних.

Розроблення інформаційної системи інтелектуального аналізу даних проведено на основі дедуктивного підходу, що передбачає розвиток процесу дослідження від *загального* до *конкретного*: задача *кластеризації* як завдання *інтелектуального аналізу даних* → методи *машинного навчання* як технологія *кластеризації* → *нейронні мережі Кохонена* як метод *машинного навчання* → модифікація *нейронної мережі Кохонена* для роботи з *нечіткими даними*.

Об’єктом дослідження є інтелектуальний аналіз даних, кластеризація як одна із задач інтелектуального аналізу даних.

Предметом дослідження є методи машинного навчання для кластеризації образів з нечіткими або розмитими межами.

Методи дослідження: *теоретичні* — аналіз літератури та наукових джерел: вивчення існуючих підходів до кластеризації нечітких даних та їхніх застосувань у штучному інтелекті; *емпіричні* — адаптація існуючих методів кластеризації, реалізація алгоритму в програмному середовищі та проведення тестування з використанням вибраних наборів нечітких даних.

Практичне значення дослідження полягає в тому, що розроблений модифікований алгоритм кластеризації на основі нейронних мереж Кохонена може бути корисним у різних прикладних задачах, де дані мають нечітку природу.

Зокрема, алгоритм може використовуватись у таких сферах, як:

- Медична діагностика — для групування пацієнтів за симптомами або результатами обстежень, коли симптоматика не завжди однозначна;
- Фінансовий аналіз — для виявлення груп ризику серед клієнтів банку за нечіткими поведінковими або фінансовими показниками;
- Розпізнавання образів — у випадках, коли зображення мають шуми або нечіткі межі об’єктів;
- Інтелектуальні системи підтримки прийняття рішень — у промисловості, енергетиці, транспорті, де часто доводиться працювати з неточними або експертними даними.

Апробація і впровадження результатів дослідження:

Основні теоретичні та практичні результати дослідження доповідалися та обговорювалися на:

- XVII Всеукраїнській науково-практичній конференції «Інформаційні технології в професійній діяльності» (м. Рівне, 5 листопада 2024 р.) [1].
- IV Всеукраїнській науково-практичній конференції «Підготовка педагогів до професійної діяльності в умовах змішаного навчання» (м. Рівне, 14 травня 2025 р.) [2].

Структура кваліфікаційної роботи складається зі вступу, чотирьох розділів, висновків, списку використаних джерел із 15 найменувань. Кількість сторінок основної частини роботи — 43. Кількість рисунків — 16.

РОЗДІЛ 1
ЗАДАЧА КЛАСТЕРИЗАЦІЇ В ІНТЕЛЕКТУАЛЬНОМУ АНАЛІЗІ ДАНИХ

1.1. Вступ до інтелектуального аналізу даних

Інтелектуальний аналіз даних — це процес автоматичного виявлення корисної інформації у великих сховищах даних. Методи інтелектуального аналізу даних застосовуються для пошуку у великих базах даних нових і корисних закономірностей, які інакше могли б залишитися невідомими.

Інтелектуальним аналізом даних (*Data Mining*) зазвичай займаються спеціалісти з обробки даних. Але це також можуть виконувати бізнес-аналітики, керівники та працівники, які функціонують як спеціалісти з даних громадян в організації.

Його основні елементи включають машинне навчання та статистичний аналіз, а також завдання керування даними, які виконуються для підготовки даних для аналізу. Використання алгоритмів машинного навчання та інструментів штучного інтелекту автоматизувало більшу частину процесу та полегшило видобуток масивних наборів даних, таких як бази даних клієнтів, записи транзакцій і файли журналів із веб-серверів, мобільних додатків і датчиків.

Data Mining є важливим компонентом успішних аналітичних ініціатив в організаціях. Інформацію, яку він генерує, можна використовувати в програмах бізнес-аналітики (BI) і розширених аналітичних програмах, які включають аналіз історичних даних, а також у аналітичних програмах у реальному часі, які перевіряють потокові дані під час їх створення або збору.

Методи *Data Mining* надають можливість передбачити результат майбутнього спостереження, наприклад, передбачити, чи витратить новоприбулий покупець більше 100 доларів в універмазі [12].

Ефективний інтелектуальний аналіз даних допомагає в різних аспектах планування бізнес-стратегії та управління операціями. Сюди входять такі функції, що стосуються клієнтів, як-от маркетинг, реклама, продажі та підтримка клієнтів,

а також виробництво, управління ланцюгом постачання, фінанси та кадри. Інтелектуальний аналіз даних підтримує виявлення шахрайства, управління ризиками, планування кібербезпеки та багато інших критичних бізнес-випадків використання. Він також відіграє важливу роль в охороні здоров'я, уряді, наукових дослідженнях, математиці, спорті тощо.

Процес аналізу даних (*Data Mining*) можна розбити на чотири основні етапи:

- **збір даних.** Релевантні дані для аналітичної програми визначаються та збираються. Дані можуть розташовуватися в різних вихідних системах, сховищі даних або озері даних, що стає все більш поширеним у середовищах великих даних, які містять суміш структурованих і неструктурованих даних. Також можна використовувати зовнішні джерела даних. Звідки б не надходили дані, фахівець із обробки даних часто переміщує їх до озера даних для решти етапів процесу;
- **підготовка даних.** Цей етап включає набір кроків для підготовки даних до видобутку. Він починається з дослідження даних, профілювання та попередньої обробки, після чого йде робота з очищення даних для виправлення помилок та інших проблем із якістю даних;
- **видобуток даних.** Після того, як дані підготовлені, фахівець з даних вибирає відповідну техніку інтелектуального аналізу даних (*Data Mining*), а потім реалізує один або кілька алгоритмів для інтелектуального аналізу. У програмах машинного навчання алгоритми, як правило, потрібно випробувати на вибіркових наборах даних, щоб шукати необхідну інформацію, перш ніж їх перевірити на повному наборі даних;
- **аналіз та інтерпретація даних.** Результати аналізу даних використовуються для створення аналітичних моделей, які можуть допомогти в ухваленні рішень та інших бізнес-діях. Спеціаліст із обробки даних або інший член групи з вивчення даних також має донести результати до керівників компаній і користувачів, часто за допомогою візуалізації даних і використання методів оприлюднення даних.

Більшість організацій переходять на цифрові технології. У результаті вони отримують у своє розпорядження величезні масиви даних, які за належного аналізу дадуть змогу підвищити цінність основних продуктів і послуг.

Data Mining забезпечує конкурентну перевагу, допомагаючи витягти цінні знання з даних про цифрові операції. Проаналізувавши поведінку клієнтів, компанії можуть використовувати результати для створення нових продуктів, послуг або методів просування.

Завдання бізнес-аналізу формулюються по-різному, але розв'язання більшості з них зводиться до тієї чи іншої задачі *Data Mining* або до їхньої комбінації. Наприклад, оцінка ризиків — це розв'язання задачі регресії або класифікації, сегментація ринку — кластеризація, стимулювання попиту — асоціативні правила. Фактично завдання *Data Mining* є елементами, з яких можна «зібрати» рішення більшості реальних бізнес-завдань.

Для розв'язання вищеописаних завдань використовуються різні методи та алгоритми *Data Mining*. З огляду на те, що *Data Mining* розвивався і розвивається на стику таких дисциплін, як математична статистика, теорія інформації, машинне навчання і бази даних, цілком закономірно, що більшість алгоритмів і методів *Data Mining* було розроблено на основі різних методів із цих дисциплін. Наприклад, алгоритм кластеризації *k-means* був запозичений зі статистики.

У *Data Mining* великої популярності набули такі методи: нейронні мережі, дерева рішень, алгоритми кластеризації, зокрема й масштабовані, алгоритми виявлення асоціативних зв'язків між подіями тощо.

Засновником і одним з ідеологів *Data Mining* вважається П'ятецький-Шапіро. Уперше термін було введено 1989 року на одному із семінарів, присвячених технологіям пошуку знань у базах даних, що проводилися в рамках Міжнародної конференції зі штучного інтелекту (*International Joint Conference on Artificial Intelligence) IJCAI-89.*

Хмарні обчислення суттєво вплинули на поширення інтелектуального аналізу даних. Незважаючи на виклики, пов’язані з безпекою, хмарні технології ідеально підходять для швидкої обробки великих обсягів даних, які можуть бути як частково структурованими, так і неструктурованими, і які накопичують численні організації [15]. Оскільки хмара дозволяє зберігати дані в різноманітних форматах, виникає потреба в розширеному наборі інструментів для збору та перетворення цих

даних у цінну інформацію. До того ж, сучасні методи інтелектуального аналізу, такі як штучний інтелект і машинне навчання, тепер доступні у хмарі у вигляді сервісів.

Очікується, що подальший розвиток хмарних технологій сприятиме появі більш ефективних інструментів для збору даних. Оскільки штучний інтелект і машинне навчання активно розвиваються, а обсяги інформації збільшуються, хмарні платформи дедалі частіше застосовуються для зберігання й обробки даних у бізнесі. Ймовірно, саме можливості хмари визначатимуть вибір методів інтелектуального аналізу даних у майбутньому.

1.2. Типи методів Data Mining

Для аналізу даних для різних програм обробки даних можна використовувати різні методи. Розпізнавання шаблонів — це звичайний випадок використання інтелектуального аналізу даних (*Data Mining*), який складається з декількох методів, та включає в себе в тому числі й виявлення аномалій, метою чого є визначення викидних значень у наборах даних.

Однак деякі методи інтелектуального аналізу даних є гнучкими: залежно від сфери використання фахівці можуть застосовувати їх у прогнозному та описовому контекстах. Ці гнучкі методи можна розділити на окремі розділи.

Прогнозне моделювання

Прогнозний дата-майнінг — це аналіз поточних та історичних даних для прогнозування майбутніх подій. Це особливо корисно в ситуаціях, коли важливо зрозуміти тенденції, закономірності та можливі результати. Наприклад, в галузі охорони здоров'я прогнозний дата-майнінг може бути використаний для аналізу даних пацієнтів і медичних записів для прогнозування майбутніх епідемій, виявлення факторів ризику конкретних захворювань і поліпшення догляду за пацієнтами за допомогою персоналізованих планів лікування.

Прогнозний дата-майнінг може бути розбитий на кілька ключових методів:

- Класифікація
- Регресія

- Аналіз часових послідовностей

Класифікація - це сортування даних на заздалегідь обрані категорії. Цей процес досліджує атрибути даних, щоб визначити, до якого класу належить кожен елемент даних. Ідентифікувавши ключові характеристики даних, можна систематично групувати або класифікувати відповідні дані.

Наприклад, авіакомпанія може класифікувати клієнтів на підставі частоти польотів і патернів витрат. Вона може ідентифікувати частих бізнес-мандрівників, які купують преміальні послуги, і відпочивальників, які віддають перевагу лоукостерам. Потім авіакомпанія може пропонувати програми лояльності та робити персоналізовані пропозиції, щоб підвищити зручність і лояльність клієнтів.

Регресія використовується для виявлення та аналізу взаємовідносин між різними змінними в даних. Основне завдання регресії — створення моделі, здатної обчислювати значення однієї змінної (залежна змінна) на підставі зміни інших змінних (незалежні змінні) [11].

Наприклад, мережа готелів може використовувати регресію для аналізу минулих аналізів бронювання і стратегій ціноутворення для прогнозування доходу в різні сезони.

Аналіз часових послідовностей — це спеціалізована методика аналізу та інтерпретації даних, зібраних через регулярні проміжки часу. Ця методика особливо корисна під час виявлення трендів, сезонних патернів і циклічних поведінок. На відміну від інших методик дата-майнінгу, що мають справу зі статичною інформацією, аналіз часових послідовностей вивчає дані, що змінюються з часом.

Авіакомпанії часто використовують аналіз часових послідовностей для прогнозування попиту пасажирів. Вивчаючи історичні дані купівлі та скасування купівлі авіаквитків, кількості пасажирів, авіакомпанія може визначити пікові періоди польотів, сезонні коливання і тренди попиту на довгу перспективу.

Дескриптивне моделювання

Дескриптивний дата-майнінг робить акцент на створенні зведень і розумінні характеристик історичних даних. Він намагається виявити патерни,

взаємовідносини і структури в наявних даних, що допомагає зрозуміти внутрішню поведінку даних.

Методики дескриптивного дата-майнінгу:

- Кластеризація
- Узагальнення
- Асоціативні правила

Кластеризація групує різні приклади даних на підставі їхньої схожості, формуючи кластери, члени яких мають більше спільного, ніж ті, що знаходяться в інших кластерах. На відміну від класифікації, за якої дані сортуються в заздалегідь встановлені категорії на підставі відомих атрибутів, кластеризація — це дослідницьке групування даних без готових міток.

Наприклад, бізнес з організації круїзів може застосовувати кластеризацію для сегментації клієнтів з метою більш ефективного маркетингу. Вивчаючи такі дані, як історія подорожей, витрати на борту і демографічний склад, круїзні компанії можуть виявляти природні групи серед своїх клієнтів. Один кластер може складатися з сімей, що віддають перевагу зручним для дітей активностям, а інший — з пар пенсіонерів, які прагнуть вишуканих задоволень.

Узагальнення (*Summarization*) — це стиснення великих датасетів у більш зручну і зрозумілу форму без втрати важливої інформації. Цей процес передбачає вилучення ключових ознак даних, що дають змогу швидко переглядати і розуміти їхні основні характеристики.

Візьмемо для прикладу велику мережу готелів з безліччю відділень по всьому світу. Узагальнення можна використовувати для консолідації та презентації таких ключових операційних даних, як коефіцієнт заповнення номерів, середня вартість номерів і демографія відвідувачів. Також це може включати в себе створення короткого звіту або дешборда для швидкого оцінювання показників.

Асоціативні правила — це методика дескриптивного моделювання даних, націлена на виявлення цікавих взаємозв'язків і асоціацій між різними змінними у великих датасетах. На відміну від узагальнення, що конденсує дані, і класифікації/кластеризації, що групують схожі елементи, асоціативні правила

виявляють патерни, зв'язки та спільну появу елементів у даних. Ця методика особливо цінна при виявленні патернів, які можуть бути неочевидні на перший погляд.

У контексті готелів асоціативні правила можуть допомогти у виявленні взаємозв'язків між сервісами, що використовуються відвідувачами. Наприклад, аналіз може показати, що ті, хто подорожує наодинці, часто віддають перевагу номерам, вікна яких не виходять на басейн (і готові платити за них більше). Цей патерн може бути показником того, що ці відвідувачі (можливо, ті, хто подорожує з діловими цілями) віддають перевагу тихішим місцям, віддаленим від потенційних джерел шуму.

Аналогічно, може з'ясуватися, що сім'ї з дітьми часто просять сусідні номери та з великою ймовірністю харчуватимуться в зручному для сімей ресторані готелю.

1.3. Задача кластеризації як завдання інтелектуального аналізу даних

Вирішення завдань інтелектуального аналізу даних часто вимагає проведення класифікації та кластеризації образів. Кластеризація або кластерний аналіз — це статистична процедура, задача якої полягає в розбитті вибірки об'єктів на підмножини, що не перетинаються і називаються кластерами. Кластеризація відноситься до класу задач машинного навчання без вчителя [4], які реалізуються штучними нейронними мережами. Хороші результати демонструють мережі конкурентного навчання, зокрема мережі Кохонена [9].

Кластеризація, або кластерний аналіз групує об'єкти таким чином, що більш схожі один на одного (за деякими ознаками) об'єкти опиняться в одному кластері, а різні — у різних. Кластеризація застосовується в тих випадках, коли ми не знаємо про те, як організовані дані.

Існують різні методи кластеризації, і їх використання призводить до різних результатів. Розглянемо два методи, які часто використовують у реальних завданнях.

K-Means

Для використання цього алгоритму потрібно визначитися з параметром k — кількістю кластерів, яку ми очікуємо отримати.

1. Обираємо k — кількість кластерів, яку ми очікуємо отримати.
2. Випадковим чином вибираємо k точок-об'єктів з усіх наявних даних. Ми називаємо їх центроїдами, бо на цьому кроці ми вважаємо, що це - потенційні центри кластерів.
3. Для кожної точки-об'єкта даних визначаємо, який центроїд до неї найближчий. На цьому етапі вважаємо, що всі точки, найближчі до одного й того самого центроїда, утворюють кластер.
4. У кожному такому кластері знаходимо середню координату - тепер це новий центроїд.
5. Повторюємо пункти 3-4: розраховуємо відстані між кожною точкою-об'єктом і новими центроїдами, визначаємо найближчі, рахуємо нові центроїди нових кластерів. Цей процес повторюється доти, доки центроїди не перестануть змінюватися.

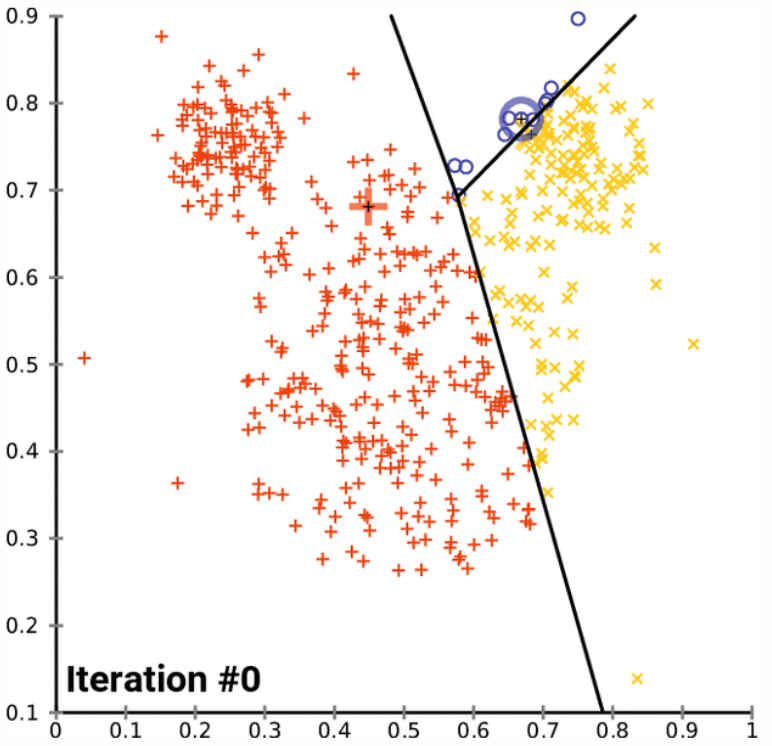

Рисунок 1.1. Початкова множина точок

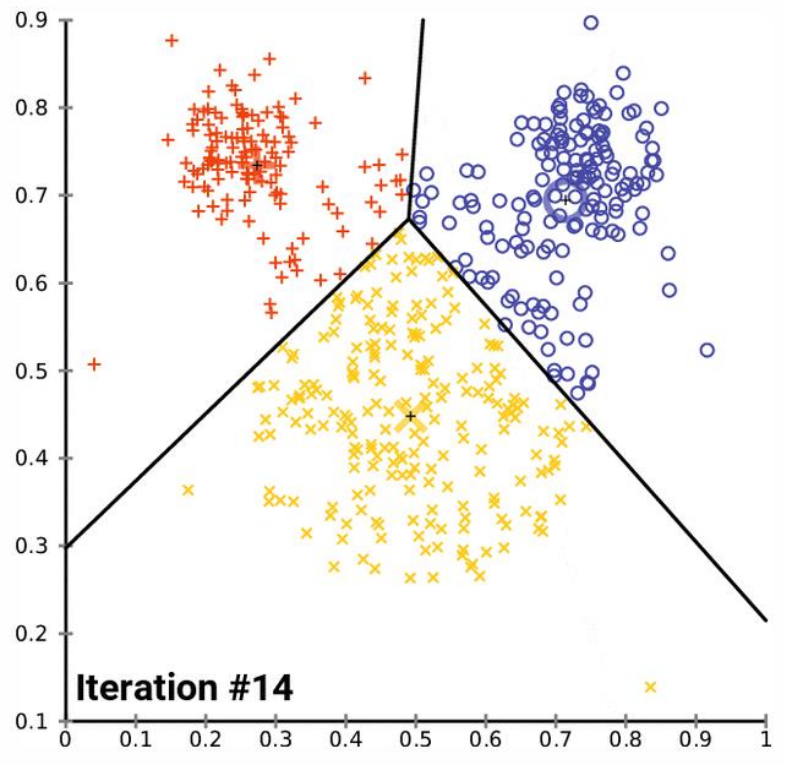

Рисунок 1.2. Результат кластеризації

DBSCAN

У цьому алгоритмі не потрібно вказувати кількість кластерів: вона визначається автоматично. Замість цього потрібно визначити радіус пошуку точок і мінімальну кількість точок у кластері.

1. Вибираємо випадкову точку-об'єкт вибірки і шукаємо точки навколо неї в обраному нами радіусі.
2. Якщо на попередньому етапі набралося менше точок, ніж мінімальна потрібна нам кількість, позначаємо всі ці точки як викиди: вважатимемо, що вони не належать жодному кластеру.
3. Якщо набралася достатня кількість точок, для кожної з них також шукаємо нові точки у визначеному нами радіусі. Таким чином, усі точки, які розташовані одна від одної на відстані меншій або рівній заданому радіусу, утворюватимуть один кластер.

Для різних завдань підходять різні методи, а застосування кількох методів на одному наборі даних може виявити різні його особливості. Наприклад, якщо серед даних є об'єкти, що не належать до жодної групи, і кластери можуть мати складну

форму, то краще використовувати *DBSCAN*. *K-Means* краще використовувати, якщо заздалегідь відома кількість груп, які мають виявитися в даних.

Важлива особливість даних, яку потрібно враховувати під час вибору алгоритму - лінійна роздільність. Якщо вибірку можна розділити лініями таким чином, що вийдуть окремі кластери, з цим завданням краще впорається *K-Means*, а якщо це неможливо — *DBSCAN*. На рисунку 1.3 перші два набори даних лінійно нероздільні: видно, що в цих випадках *DBSCAN* визначає кластери більш коректно, ніж *K-Means*.

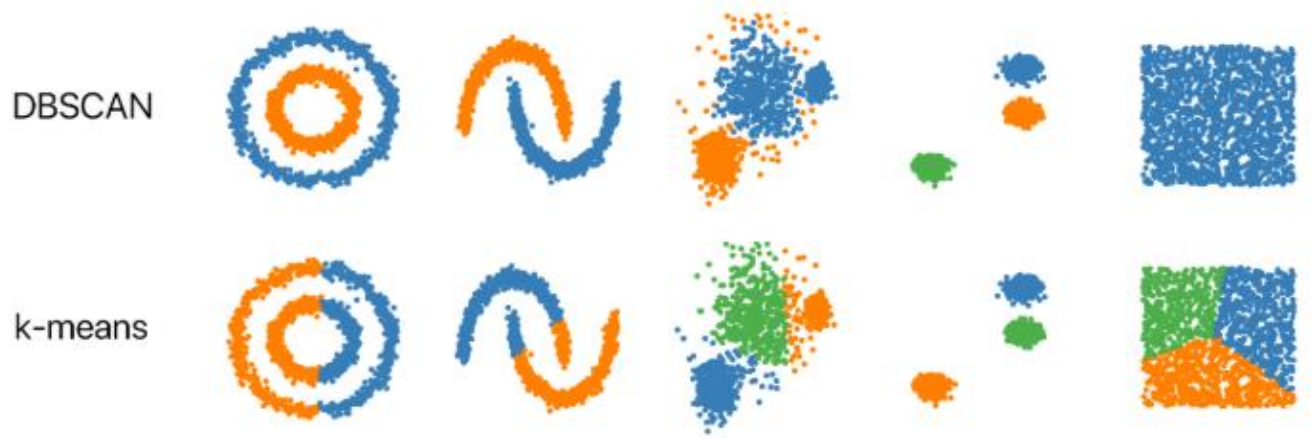

Рисунок 1.3. Кластеризація образів різними методами

Кластеризацію широко застосовують у всіх галузях науки про дані. У маркетингових завданнях кластерний аналіз дає змогу визначити категорії клієнтів (які ми заздалегідь не знали) і на основі цієї інформації пропонувати різні товари різним групам.

Кластеризація користувачів за геоданими дає змогу згрупувати близькі (у просторі та часі) точки в повноцінні локації: дім, місце роботи або улюблений магазин — і, таким чином, зробити про користувача додаткові висновки.

Також кластеризація дає змогу розв'язати, наприклад, задачу розпізнавання облич у натовпі: обличчя, що належать одній і тій самій людині, за достатньої кількості даних, збиратимуться в один кластер. Такий підхід дає змогу уникнути ручної розмітки зображень.

РОЗДІЛ 2
МЕТОДИ МАШИННОГО НАВЧАННЯ ЯК ТЕХНОЛОГІЇ ВИРІШЕННЯ ЗАДАЧІ КЛАСТЕРИЗАЦІЇ

2.1. Алгоритми машинного навчання, що використовуються для кластеризації

2.1.1. Навчання без учителя як основа кластеризації

Навчання без учителя — це підхід, застосовуваний для виявлення базової структури даних. Алгоритми навчання без вчителя не вимагають відображення входів і виходів і, відповідно, участі людини. Зазвичай їх використовують для виявлення наявних закономірностей у даних, тож екземпляри групуються без необхідності в мітках. Передбачається, що екземпляри, які потрапляють в одну групу, мають схожі характеристики.

Цей метод найчастіше використовується в таких сценаріях, як сегментація клієнтів, виявлення аномалій, аналіз споживчого кошика, кластеризація документів, аналіз соціальних мереж, стиснення зображень.

Найпоширеніші алгоритми та методи навчання без учителя:

- Кластеризація методом *k*-середніх (*k-means*). Популярний і широко використовуваний алгоритм, який розбиває дані на так звані *k*-кластери. Кожну точку даних присвоюють найближчому кластерному центру, а центри кластерів перераховують ітеративно. Часто використовується для кластеризації документів, стиснення зображень і сегментації ринку.
- Ієрархічна кластеризація. Алгоритм будує ієрархію кластерів двома способами: агломеративним (підхід від низу до верху) і розділовим (низхідний підхід). Використовується для організації документів і аналізу соціальних мереж.
- *DBSCAN*. Алгоритм групує точки даних, які щільно запаковані, і відзначає ті, які лежать окремо, як викиди. *DBSCAN* припускає, що кластери - це щільні області в просторі, розділені областями з меншою щільністю. На відміну від

k-середніх, *DBSCAN* виводить кількість кластерів на основі даних і може виявляти кластери довільної форми. Використовується для просторового аналізу даних і фільтрації шуму.

- Аналіз головних компонентів. Перетворює дані на набір некорельованих компонентів, які максимізують дисперсію. Цей процес знижує розмірність даних. Метод використовується в аналізі експресії генів, стисненні зображень і розвідувальному аналізі даних.
- Ізоляційні ліси. Алгоритм створює набір дерев, випадковим чином вибираючи ознаку і розділяючи дані. Потім алгоритм виявляє аномалії, шукаючи точки, які потребують менше поділів для ізоляції. Метод використовується для забезпечення мережевої безпеки та виявлення шахрайства.
- Однокласовий *SVM*. Цей метод вивчає межу, яка відокремлює нормальні точки даних від викидів. Використовується для багатовимірних даних і в задачах виявлення аномалій, таких як виявлення виробничих дефектів або шахрайство з кредитними картками.

Переваги:

- не потрібні розмічені набори даних;
- виявляються приховані закономірності;
- скорочуються розмірності;
- виявляються аномалії та викиди в представлених даних;
- підвищується економічна ефективність.

Недоліки:

- труднощі в інтерпретації результатів через відсутність міток;
- немає чітких метрик;
- ресурсоємність;
- проблеми перенавчання;
- залежність від якості використовуваних ознак.

2.1.2. Використання PCA для зменшення розмірності перед кластеризацією

Метод головних компонент (*PCA*) — це один із найбільш зрозумілих і широко застосовуваних способів зменшення розмірності даних шляхом їх проєкції на ортогональний підпростір ознак.

В загальному випадку можна уявити, що всі наші дані розташовані у вигляді еліпсоїда в просторі початкових ознак, а новий базис, на який ми їх проєктуємо, співпадає з осями цього еліпсоїда. Таке припущення дозволяє усунути сильні кореляції між ознаками, оскільки вектори нового базису будуть взаємно перпендикулярними.

Хоча розмірність еліпсоїда зазвичай дорівнює розмірності вихідного простору, припускаючи, що дані фактично розташовані у підпросторі меншої розмірності, можна відкинути непотрібні напрямки — ті, уздовж яких еліпсоїд має найменшу протяжність. Процес вибору нового базису відбувається поетапно: кожного разу обирається вісь еліпсоїда з максимальною дисперсією серед залишених, що дозволяє «жадібно» побудувати підпростір з найбільшою інформацією [7].

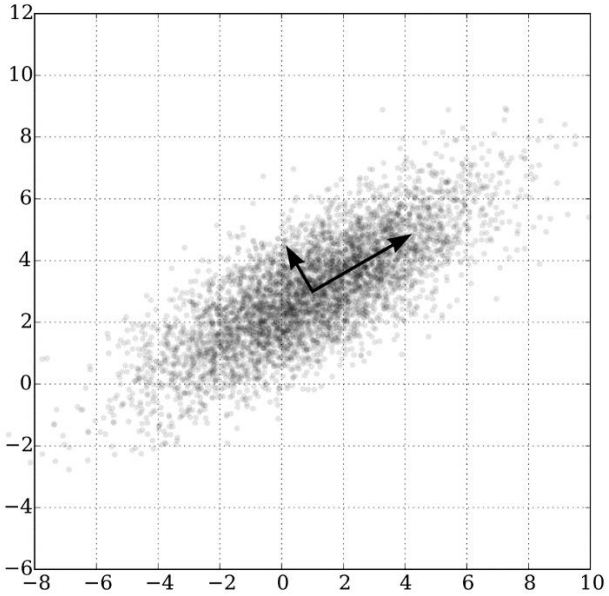

Рисунок 2.1. Зображення еліпсоїда

Для того щоб знизити розмірність даних з n в k, $k \leq n$, нам потрібно вибрати топ-k осей такого еліпсоїда, відсортовані по спаданню по дисперсії вздовж осей.

Почнемо з того, що порахуємо дисперсії та коваріації вихідних ознак. Це робиться за допомогою матриці коваріації. По визначенню коваріації, для двох ознак X_i та X_j їх коваріація визначається за формулою:

$cov(X_i,\ X_j) = E[(X_i - \mu_i)(X_j - \mu_j)] = E[X_i X_j] - \mu_i \mu_j$, де μ_i – математичне сподівання і-тої ознаки.

При цьому зазначимо, що коваріація симетрична і коваріація вектора із самим собою дорівнюватиме його дисперсії.

Таким чином матриця коваріації являє собою симетричну матрицю, де на діагоналі лежать дисперсії відповідних ознак, а поза діагоналлю - коваріації відповідних пар ознак. У матричному вигляді, де X – це матриця спостережень, матриця коваріації матиме такий вигляд:

$\Sigma = E[(\boldsymbol{X} - \boldsymbol{E}[\boldsymbol{X}])(\boldsymbol{X} - \boldsymbol{E}[\boldsymbol{X}])^T]$

Матриці як лінійні оператори мають власні значення та власні вектори (eigenvalues та eigenvectors). Вони визначають підпростір, який під час дії цією матрицею як лінійним оператором, залишається на місці або «переходить у себе».

Коли матриця діє на відповідний лінійний простір, власні вектори залишаються на місці і лише множаться на відповідні їм власні значення. Власний вектор w_i з власним значенням λ_i для матриці M обчислюється як:

$Mw_i = \lambda_i w_i$

Зі співвідношення Релея випливає, що максимальна варіація набору даних буде досягатися вздовж власного вектора цієї матриці, що відповідає максимальному власному значенню. Отже, головні компоненти, на які потрібно проектувати дані, є власними векторами відповідних топ-k штук власних значень цієї матриці.

Далі потрібно просто помножити матрицю даних на ці компоненти і ми отримаємо проекцію наших даних в ортогональному базисі цих компонент.

Тепер якщо ми транспонуємо нашу матрицю даних і матрицю векторів головних компонент, ми відновимо вихідну вибірку в тому просторі, з якого ми робили проекцію на компоненти.

Для прикладу створимо синтетичний набір даних з 20 компонент, з яких:

- 5 інформативних (*n_informative = 5*) — мають реальне значення для кластеризації;
- 5 надлишкових (редундантних) (*n_redundant=5*) — це комбінації інформативних ознак;
- 10 залишкових (шумових) — не несуть значущої інформації (створюються автоматично як випадкові).

Таким чином, із 20 ознак лише частина дійсно корисна, що імітує реальні задачі, де важливо виділити головне серед зайвого.

Код програми для генерації та порівняння обох випадків

```
import numpy as np
import matplotlib.pyplot as plt
from sklearn.datasets import make_classification
from sklearn.cluster import KMeans
from sklearn.decomposition import PCA
from sklearn.metrics import silhouette_score

X, y_true = make_classification(
        n_samples=1000, n_features=20, n_informative=5,
        n_redundant=5, n_clusters_per_class=1, n_classes=4,
        random_state=42)

# --- Без PCA ---
kmeans_no_pca = KMeans(n_clusters=4, random_state=42)
labels_no_pca = kmeans_no_pca.fit_predict(X)
score_no_pca = silhouette_score(X, labels_no_pca)
```

```
# --- 3 PCA (2 компоненти) ---
pca = PCA(n_components=2)
X_pca = pca.fit_transform(X)

kmeans_pca = KMeans(n_clusters=4, random_state=42)
labels_pca = kmeans_pca.fit_predict(X_pca)
score_pca = silhouette_score(X_pca, labels_pca)
```

Для порівняння якості кластеризації використовується метрика *Silhouette Score* (коефіцієнт силуету) — метрика, яка показує, наскільки добре об’єкти в кластеризації відокремлені один від одного та наскільки щільно згруповані всередині своїх кластерів.

На рисунку 2.2 наведено приклад кластеризації без *PCA* та з *PCA* для згенерованого набору даних. Можна зробити висновок, що без *PCA* кластери накладаються, бо *K-Means* намагається працювати у просторі з великою кількістю ознак, де багато з них можуть бути шумовими або надлишковими. А з використанням *PCA* ми виділяємо головні компоненти, які містять найбільше інформації. Завдяки цьому *K-Means* знаходить значно чіткіші межі між кластерами.

Метод головних компонент допоміг покращити якість кластеризації в понад 2 рази, що особливо помітно на високовимірних даних.

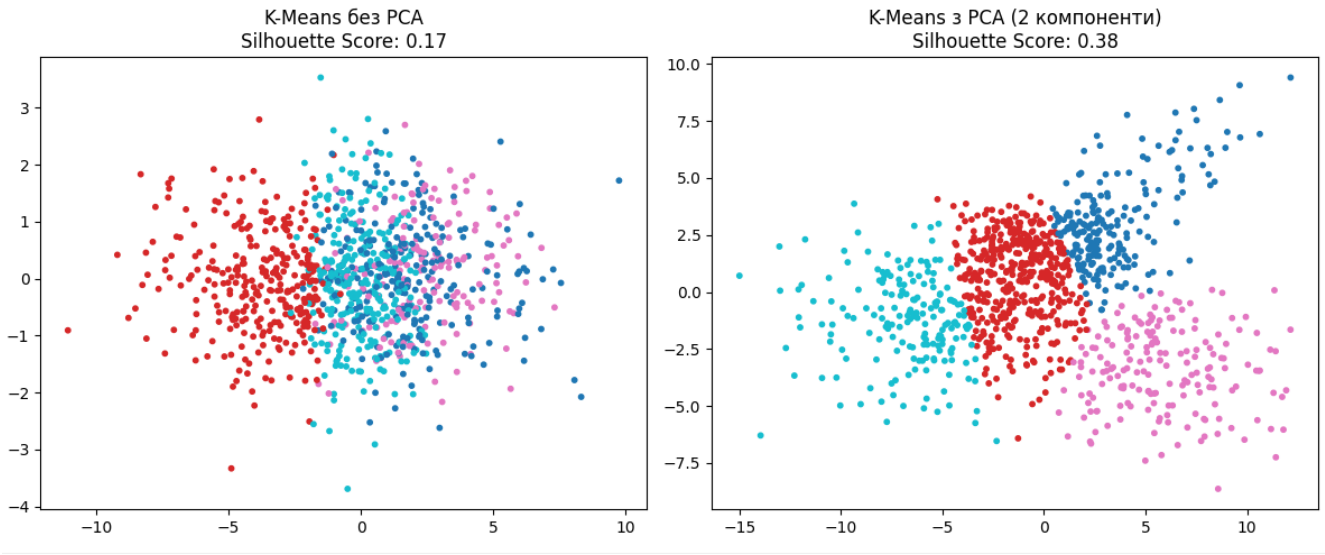

Рисунок 2.2. Кластеризація набору даних до і після зменшення розмірності

2.2. Нейронні мережі Кохонена як інструмент для кластеризації

2.2.1. Загальний опис нейронних мереж

Для вирішення задач класифікації та кластеризації поряд з іншими методами широко використовуються штучні нейронні мережі [3]. Хороші результати демонструють мережі зустрічного поширення (*Counter Propagation*), у яких поєднуються різні нейропарадигми — два шари нейронів різного типу (Кохонена і Ґросберга). Мережі Кохонена відповідають за кластеризацію, виявляючи структури даних у навчальному наборі, тоді як нейрони Ґросберга проводять класифікацію на основі відповідей кластерів.

Іншим ефективним підходом є використання нейромереж квантування навчального вектора (*Learning Vector Quantization, LVQ*). Цей метод класифікації базується на змаганні між нейронами за належність до конкретного класу і дозволяє побудувати просту модель на основі навчальних прикладів, при цьому зберігаючи високу точність у випадках, де необхідно ідентифікувати образи або передбачити категорії.

Загалом було розроблено і впроваджено в практику численні методи кластеризації, і більшість з них дозволяють отримати якісні або оптимальні результати кластеризації в галузі інформатики, науки про дані, статистики, розпізнавання образів, штучного інтелекту і машинного навчання.

Також важливим напрямом є самоорганізовуючі карти Кохонена (*Self-Organizing Maps, SOM*), які дозволяють вирішувати задачі кластеризації з використанням принципів самоорганізації. Цей алгоритм утворює карту простору ознак, де близькі за властивостями об’єкти автоматично розміщуються у сусідніх зонах карти. Це сприяє інтуїтивно зрозумілому відображенню кластерів даних, що корисно для виявлення природних групувань.

У мережі Кохонена (*SOM*) використовується впорядкована топологія розміщення нейронів, яка зазвичай представлена у вигляді одно- або двовимірної сітки. Кожен нейрон в такій сітці представлений n-вимірним вектором-стовпцем $W = (w_1, \dots, w_n)^T$, де n відповідає розмірності простору вхідних даних.

Використання одно- та двовимірних сіток обумовлене складністю візуалізації та інтерпретації структур вищих розмірностей.

Нейрони, як правило, розміщуються у вузлах двовимірної решітки, яка може мати прямокутну або шестикутну форму. Між нейронами існують взаємозв’язки, інтенсивність яких залежить від відстані між ними на карті.

Крім цього, для нечітких даних можуть використовуватися нечіткі нейронні мережі, які враховують невизначеність у даних та дозволяють моделювати складні системи з неповною інформацією. Прикладом є нечіткі самоорганізовуючі карти, де нейрони характеризуються нечіткими правилами або ймовірностями приналежності до різних кластерів.

Таким чином, використання нейронних мереж у задачах класифікації та кластеризації дозволяє підвищити якість та точність інтелектуального аналізу даних. Вибір конкретної архітектури та алгоритму залежить від специфіки даних, рівня невизначеності та необхідної точності прогнозування або виявлення кластерів.

2.2.2. Нейронні мережі Кохонена

Нейронні мережі Кохонена — клас нейронних мереж, що використовують навчання без учителя, основним принципом роботи яких є введення у правило навчання нейрона інформації про його розташування. Мережа Кохонена являє собою спеціальний тип нейронної мережі для розв'язання задачі кластеризації. Вона складається всього з двох шарів - вхідного (розподільчого) і вихідного, який також називають шаром Кохонена.

У мережі Кохонена кожен нейрон вхідного шару пов'язаний з усіма нейронами вихідного, а всередині шарів зв'язків немає. На нейрони вхідного шару подаються вектори ознак об'єктів, що кластеризуються.

В основу алгоритму навчання прошарку Кохонена покладена властивість скалярного добутку векторів фіксованої довжини зростати до свого максимального значення при прямуванні кута між ними до нуля.

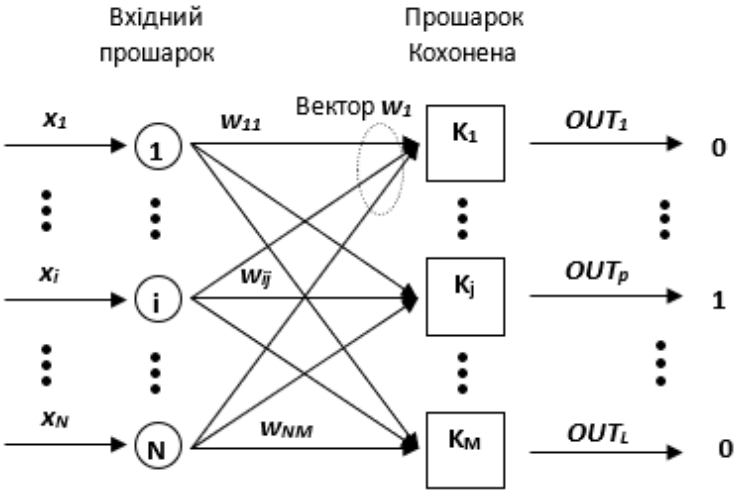

Рисунок 2.3. Структура мережі Кохонена

У мережі Кохонена кожен елемент вхідного образу $\boldsymbol{x_1, x_2,\ldots, x_N}$ подається на всі нейрони прошарку Кохонена. Ваги зв'язків $\boldsymbol{w_{ij}}$ утворюють матрицю **W**. При цьому вектор-стовпець $\boldsymbol{W_j}$ матриці **W** утворює вектор ваг нейрона Кохонена з номером **j.** Нейрони на виході дають сигнал, рівний 0 (пасивний стан) або 1 (збуджений стан).

Як і у звичайній нейронній мережі, вхідні нейрони не беруть участі в процесі навчання й оброблення даних, а просто розподіляють вхідний сигнал по нейронах наступного шару. Число вхідних нейронів дорівнює розмірності вектора ознак (тобто числу ознак об'єкта).

Основним класифікатором у таких мережах виступає шар Кохонена. Він функціонує за правилом «переможець отримує все»: активується лише один нейрон-переможець $\boldsymbol{j}$, для якого вектор ваг $\vec{W}^j$ є найближчим до вхідного образа $\vec{X}$ серед векторів ваг усіх нейронів Кохонена. При цьому всі інші нейрони перебувають у пасивному стані. Звичайно, так само близьким має бути вектор ваг $\vec{W}^j$ до інших образів, подібних до $\vec{X}$.

Цей визначальний принцип функціонування дозволяє шару Кохонена розподіляти вхідні вектори в групи подібних між собою – кластери. Якщо деякий вхідний образ активує якийсь нейрон-переможець, то всі подібні до нього образи даватимуть аналогічний результат. Очевидно, що відмінні образи вже не

даватимуть такого самого результату — швидше за все вони активуватимуть іншого переможця.

У двовимірному випадку подібні вектори заповнюють деякий сектор з гострим кутом при вершині. Для прикладу на рисунку 2.4 зображено три групи подібних векторів, які утворюють три сектори зі спільною вершиною у початку координат.

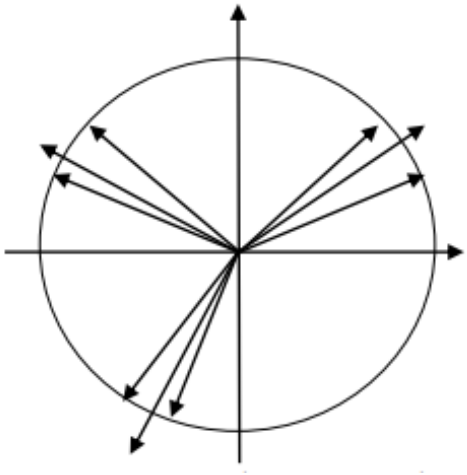

Рисунок 2.4. Три групи подібних векторів-образів

У тривимірному випадку подібні вектори охоплюються деяким криволінійним (не обов’язково круговим) конусом. Аналогічно у N-вимірному випадку подібні між собою вектори розміщуватимуться в деякому N-вимірному криволінійному гіперконусі. Навчальна множина може містити багато подібних між собою вхідних векторів, і мережа має бути навченою активувати один і той самий нейрон Кохонена для кожного з них. Для кожної групи подібних образів вектор $\overset{\rho}{W}j$ відповідного нейрона-«переможця» має зайняти деяке серединне положення в своєму гіперконусі. Це положення має бути таким, щоб мінімізувати відстань вектора ваг до найбільш віддалених із образів. Якщо знову звернутися до двовимірного випадку, то вектор ваг має бути близьким до бісектриси сектора, заповненого вхідними образами. Таке розміщення вектора $\overset{\rho}{W}j$ забезпечить максимальні значення суматора нейрона-«переможця» Кохонена для всіх подібних образів.

Положення вектора ваг також залежатиме від довжин векторів, що утворюють кластер. Щоб нейрон-«переможець» приблизно однаково

«відгукувався» на всі подібні образи, потрібно, щоб його вектор вагових коефіцієнтів був ближчим до коротших із таких векторів.

У випадку чітких даних класичний алгоритм навчання шару Кохонена передбачає локалізацію векторів ваг його нейронів серед сукупностей векторів образів, що утворюють спільні кластери. Якщо ж дані, що є компонентами образів деякої предметної області, визначені нечітко, то вектор вагових коефіцієнтів нейронів Кохонена стане багатозначним із різним ступенем приналежності в межах від 0 до 1.

Навчання мережі Кохонена, як і звичайної нейронної мережі, полягає в підлаштуванні ваг зв'язків між нейронами, але проводиться з використанням технології конкурентного навчання.

2.3. Історія нечіткої логіки та визначення нечіткої кластеризації

Нечітку логіку вперше запропонував Лотфі Заде в 1965 році в статті для журналу Information and Control. У своїй статті під назвою "Нечіткі множини" Заде спробував відобразити тип даних, що використовуються в обробці інформації, і вивів елементарні логічні правила для такого типу множин.

"Найчастіше класи об'єктів, що зустрічаються в реальному фізичному світі, не мають чітко визначених критеріїв належності, — пояснює Заде. "Проте факт залишається фактом: такі нечітко визначені "класи" відіграють важливу роль у людському мисленні, особливо у сферах розпізнавання образів, передачі інформації та абстрагування".

Відтоді нечітка логіка успішно застосовується в системах керування машинами, обробці зображень, штучному інтелекті та інших сферах, які покладаються на сигнали з неоднозначною інтерпретацією.

Термін «нечіткий» відноситься до речей, які не є чіткими або розпливчастими. У реальному світі ми часто стикаємося з ситуацією, коли не можемо визначити, чи є стан істинним або хибним, їх нечітка логіка забезпечує

дуже цінну гнучкість для міркувань. Таким чином, ми можемо врахувати неточності та невизначеності будь-якої ситуації.

Нечітка логіка — це форма багатозначної логіки, в якій значеннями істинності змінних можуть бути будь-які дійсні числа від 0 до 1, а не лише традиційні значення «істина» або «хибність». Вона використовується для роботи з неточною або невизначеною інформацією і є математичним методом для представлення нечіткості та невизначеності в процесі прийняття рішень.

Нечітка логіка ґрунтується на ідеї, що в багатьох випадках поняття істини або хибності є занадто обмежувальним, і що між ними існує багато відтінків сірого. Вона допускає часткові істини, коли твердження може бути частково істинним або хибним, а не повністю істинним або хибним.

Фундаментальним поняттям нечіткої логіки є функція належності, яка визначає ступінь приналежності вхідного значення до певної множини або категорії. Функція належності є відображенням вхідного значення на ступінь належності від 0 до 1, де 0 означає неналежність, а 1 - повну належність.

Нечітка логіка реалізується за допомогою нечітких правил, які є твердженнями типу «якщо-тоді», що виражають зв'язок між вхідними змінними та вихідними змінними у нечіткий спосіб. Результатом роботи системи нечіткої логіки є нечітка множина, яка є набором ступенів приналежності для кожного можливого вихідного значення.

Таким чином, нечітка логіка — це математичний метод для представлення нечіткості та невизначеності в процесі прийняття рішень, який допускає часткову істинність і використовується в широкому діапазоні застосувань. Він базується на концепції функції належності, а його реалізація здійснюється за допомогою нечітких правил.

У булевій системі значення істинності 1.0 означає абсолютну істину, а 0.0 — абсолютну брехню. Але в нечіткій системі немає логіки для абсолютної істини і абсолютної брехні. Але в нечіткій логіці також присутнє проміжне значення, яке є частково істинним і частково хибним.

Традиційні методи кластеризації, такі як K-середні, відносять кожну точку даних до одного кластера, створюючи чітко визначені межі. Однак у багатьох реальних сценаріях точки даних не належать строго до одного кластера, а скоріше демонструють характеристики декількох кластерів одночасно.

Саме тут на допомогу приходить нечітка кластеризація, заснована на нечіткій логіці, що робить її більш придатною для роботи з невизначеністю та розподілом даних, що перетинаються.

Нечітка кластеризація — це тип алгоритму кластеризації в машинному навчанні, який дозволяє точці даних належати до декількох кластерів з різним ступенем приналежності. На відміну від традиційної кластеризації (наприклад, *K-Means*), де кожна точка даних належить лише одному кластеру, нечітка кластеризація дозволяє точці даних належати до декількох кластерів з різними рівнями приналежності.

Для прикладу можна привести гостей на вечірці. Гості невимушено об'єднуються в групи на основі спільних інтересів, наприклад, любителі музики, гурмани та спортивні фанати. Деякі люди чітко вписуються в одну групу, скажімо, гітарист, який говорить тільки про музику. Але інші можуть належати до кількох груп. Людина, яка любить і музику, і їжу, може бути частково в обох групах, замість того, щоб бути примушеною належати лише до однієї.

Нечітка кластеризація або м'яка кластеризація — це навчання без учителя, яке групує вибірки даних у кластери з різною вагою або ймовірністю. На відміну від методів жорсткої кластеризації, таких як k-середні, які відносять кожну вибірку даних до одного кластера, нечітка кластеризація дозволяє вибіркам даних належати до більш ніж одного кластера. Таким чином, нечітка кластеризація вносить невизначеність у сегментацію даних.

Найвідомішою технікою нечіткої кластеризації є метод нечітких *C*-середніх (*Fuzzy C-means, FCM*) [14], в якій вибірки даних розподіляються по різних кластерах на основі певного ступеня приналежності за допомогою ітеративного процесу, спрямованого на мінімізацію цільової функції. У стандартному *FCM* точка даних матиме високий ступінь приналежності до кластера, якщо вона лежить

ближче до центру цього кластера, і низький ступінь приналежності, якщо вибірка даних лежить далеко від центру кластера. Одним з обмежень стандартного *FCM* є те, що цільова функція *FCM* не враховує просторову залежність між пікселями зображення [8].

Можливі галузі використання нечіткої кластеризації:

- **Сегментація зображень:** Нечітку кластеризацію можна використовувати для сегментації зображень, групуючи пікселі зі схожими властивостями, такими як колір або текстура [6].
- **Розпізнавання образів:** Нечітку кластеризацію можна використовувати для виявлення закономірностей у великих наборах даних, групуючи схожі точки даних разом.
- **Маркетинг:** Нечітку кластеризацію можна використовувати для сегментування клієнтів на основі їхніх уподобань і купівельної поведінки, що дозволяє проводити більш цілеспрямовані маркетингові кампанії.
- **Медична діагностика:** Нечітку кластеризацію можна використовувати для діагностики захворювань, групуючи пацієнтів зі схожими симптомами.
- **Моніторинг навколишнього середовища:** Нечітка кластеризація може бути використана для виявлення зон, що викликають занепокоєння з точки зору екології, шляхом групування зон зі схожими рівнями забруднення або іншими екологічними показниками.
- **Аналіз транспортних потоків:** Нечітка кластеризація може бути використана для аналізу транспортних потоків шляхом групування схожих транспортних потоків, що дозволяє покращити управління та планування дорожнього руху.
- **Оцінка ризиків:** Нечітка кластеризація може бути використана для виявлення та кількісної оцінки ризиків у різних сферах, таких як фінанси, страхування та інженерія.

РОЗДІЛ 3
КЛАСТЕРИЗАЦІЯ НЕЧІТКИХ ОБРАЗІВ У ДИСКРЕТНИХ ПРОСТОРАХ ОЗНАК

3.1. Підхід до проблеми кластеризації нечітких образів

Для вирішення проблеми кластеризації нечітких образів пропонується спочатку розглянути дискретний випадок. У випадку чітких даних кожен образ $\overset{\rho}{X}^t = \left(x_1^t, x_2^t, \text{K}, x_N^t\right)$ фактично є точкою у N-вимірному просторі ознак. У випадку нечітких дискретних даних в якості кожного образа пропонується розглядати множину векторів із збільшеною на один кількістю компонент виду

$$\overset{\rho}{X}^t = \left\{\left(x_1^{q_t}, x_2^{q_t}, \text{K}, x_N^{q_t}, p_{q_t}\right)\right\}_{q_t=1}^{K_t}, \quad t = 1, 2, \text{K}, D.$$

Тут позначено: D – загальна кількість усіх образів; K_t – кількість можливих дискретних значень образа $\overset{\rho}{X}^t$ у множині даних; $\left(x_1^{q_t}, x_2^{q_t}, \text{K}, x_N^{q_t}\right)$ – одне із можливих дискретних значень образа $\overset{\rho}{X}^t$ у множині даних; p_{q_t} – ймовірність перебування образа $\overset{\rho}{X}^t$ у точці $\left(x_1^{q_t}, x_2^{q_t}, \text{K}, x_N^{q_t}\right)$ простору ознак. Для різних образів $\overset{\rho}{X}^t$ число K_t можливих дискретних значень може бути різним, головне щоб сума їх ймовірностей давала одиницю, тобто $\sum_{q_t=1}^{K_t} p_{q_t} = 1$.

Такий спосіб структурування нечітких даних дозволяє адаптувати алгоритм навчання нейронів Кохонена. Модифікація алгоритму передбачає наступне:

- нечіткі образи утворюють двовимірну структуру даних, подібну до «порізаних»/«рваних» масивів (jagged arrays) у мові програмування C#: зовнішній масив – це набір усіх образів $\overset{\rho}{X}^t$, а кожен внутрішній масив – це одне із можливих дискретних значень образа $\overset{\rho}{X}^t$;

- на вхід нейронів Кохонена по черзі подаються можливі дискретні значення образів $\left(x_1^{q_t}, x_2^{q_t}, \text{K}, x_N^{q_t}\right)$ і за класичним алгоритмом визначається нейрон-переможець. Для різних можливих значень образів можуть бути різні переможці, тому всі нейрони-переможці зберігаються у додатковому списку;
- корекція ваг переможців проводиться лише після опрацювання усіх можливих дискретних значень образів із врахуванням ймовірності p_{q_t} перебування образа у відповідній точці простору ознак, яка виконує функцію коефіцієнта швидкості навчання. Таким чином усі можливі дискретні значення одного образа можуть корегувати одночасно декілька нейронів Кохонена.

Запропоновану методику кластеризації нечітких образів можна застосовувати і у випадку, коли нечіткі дані задаються неперервним розподілом ймовірності перебування образа у деякій області простору ознак. У цьому випадку потрібно використовувати інтегральні міри близькості векторів ваг нейронів Кохонена та неперервно розподілених образів у відповідних областях простору ознак.

Кожна точка (образ) в множині нечітких даних — це сукупність можливих значень цього образа з певними ймовірностями. Тобто початкове завдання – це вибрати центр образа (його координати) і вже після цього навколо центра можна згенерувати вибірку можливих значень цієї точки.

При генерації значень важливо враховувати, що сумарна ймовірність всіх значень повинна дорівнювати 1. Тому для того, щоб згенерувати рівномірну сітку значень кожного образа можна вибрати ймовірності в діапазоні [0,01; 0,05]. Мова програмної реалізації – Python.

chunk_prob = round(np.random.uniform(0.01, 0.05), 2)

sum_prob += chunk_prob

Всі ймовірності генеруються в циклі *while*, поки *sum_prob* < 1. Коли цикл завершується, остання ймовірність коригується, для того щоб сума чітко дорівнювала 1.

Таким чином, отримуємо структуру формату [*point_center*, [*point_chunk*, *chunk_prob*]], де *point_center* – координати самого образу, *point_chunk* – координати певного дискретного значення образу, *chunk_prob* – ймовірність образу знаходитися на координатах *point_chunk*.

Всі ці образи групуються в одному масиві, де кожен *chunk* (дискретне значення образа) має посилання на свій оригінальний образ (*point_center*), що дасть змогу віднести кінцевий образ до різних кластерів, якщо різні точки цього образа активують різні нейрони.

Розглянутим вище методом досліджено залежність результату кластеризації від початкової локалізації та від розподілу ймовірностей перебування образа у деякій області простору ознак. Не зменшуючи загальності, розглянуто випадок двовимірного простору ознак.

На рисунку 3.1 зображено початкові згенеровані образи. Збоку зліва, справа та знизу згенеровані рівномірно розподілені образи з приблизною кількістю точок 35-40 на один образ. По 4 образи з кожної сторони. Розмір точки вказує на її ймовірність, чим більша точка, тим більша її ймовірність.

Наступні образи — це еліптичні образи, які були початкові згенеровані так, щоб найбільші ймовірності мали точки, які знаходяться на краю еліпса. Приклад генерації еліптичних точок наведений на рисунку 4.3.

Кожен образ має свій центр, в даному випадку це центр еліпса. Поточний еліпс має чотири крайні точки з більшими ймовірностями і решту точок з меншими ймовірностями.

У процесі кластеризації особлива увага приділяється тому, як саме ймовірнісні характеристики образу впливають на кінцеве рішення алгоритму. Зокрема, якщо дискретні точки одного і того ж образу мають високу ймовірність та виявляються просторово рознесеними у різні області ознакового простору, це може призвести до ситуації, коли один образ буде частково віднесений до кількох різних кластерів. Такий підхід дозволяє моделювати неоднорідність образів та їхню внутрішню структуру, що є важливим при роботі зі складними або багатокомпонентними даними.

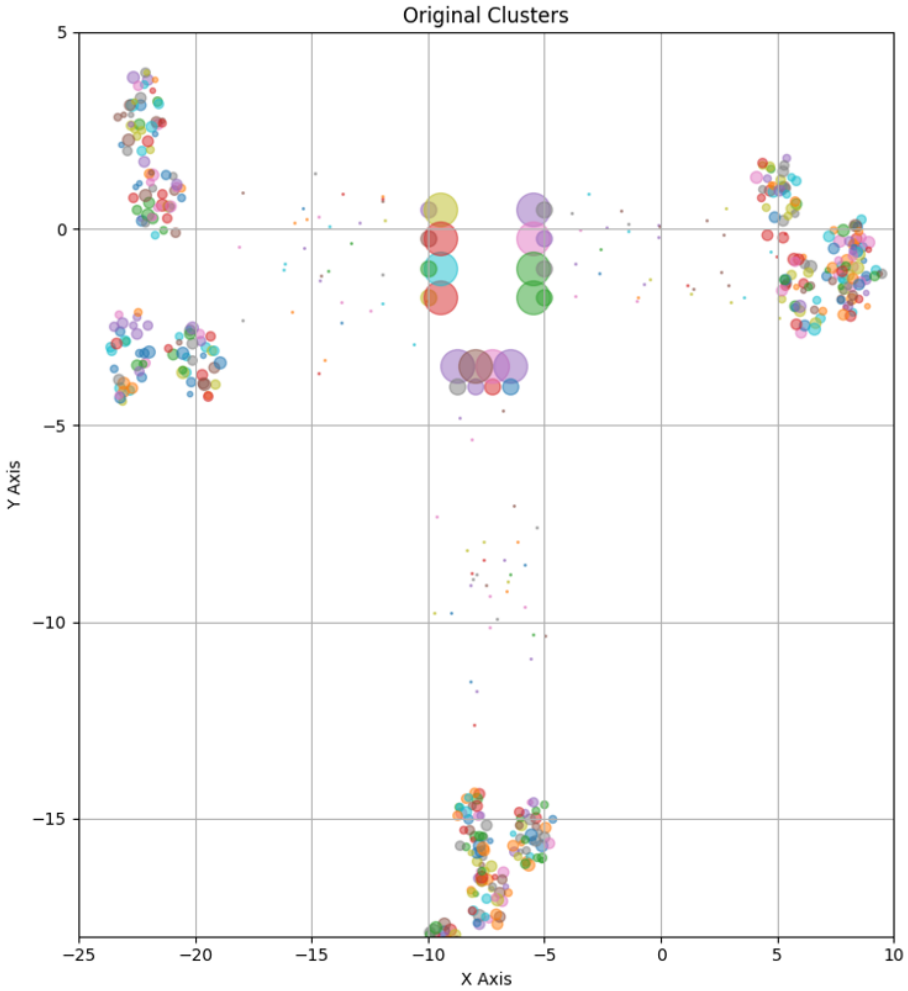

Рисунок 3.1. Множина згенерованих нечітких образів

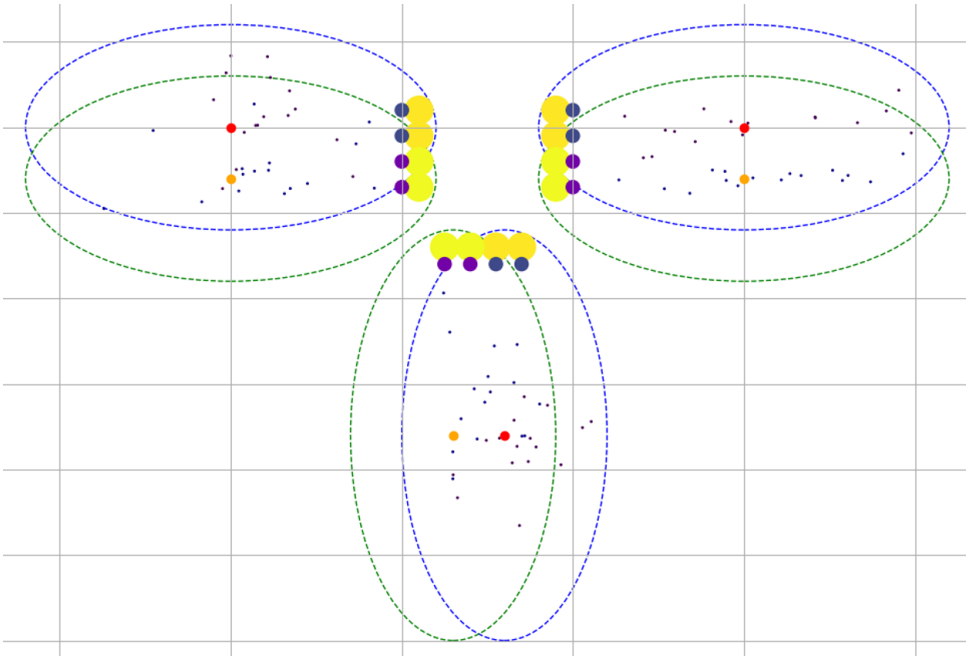

Рисунок 3.2. Згенеровані еліптичні образи

3.2. Групування множини та розділення її на окремі кластери

Згрупувавши множину в одне ціле, можна провести процес кластеризації. Результат кластеризації наведений на рисунку 3.3.

Як видно на рисунку, образи з більшими ймовірностями утворили свій власний кластер 6, забравши ще декілька близьких до себе точок. Водночас рівномірні розподілені образи утворили також власні кластери 0, 1, 2, розмістивши вагові вектори приблизно по центру кластерів. Решта точок з меншими ймовірностями утворили окремі кластери 3, 4, 5.

Отже, можна з впевненістю заявити, що мережа розділяє окремі точки з різними ймовірностями, «притягуючи» вагові вектори до центрів локалізації точок зі схожими ймовірностями. Що і є початковим завданням кластеризації — об’єднати схожі між собою точки, попередньо не знаючи (навчання без учителя) до яких класів чи кластерів вони належали.

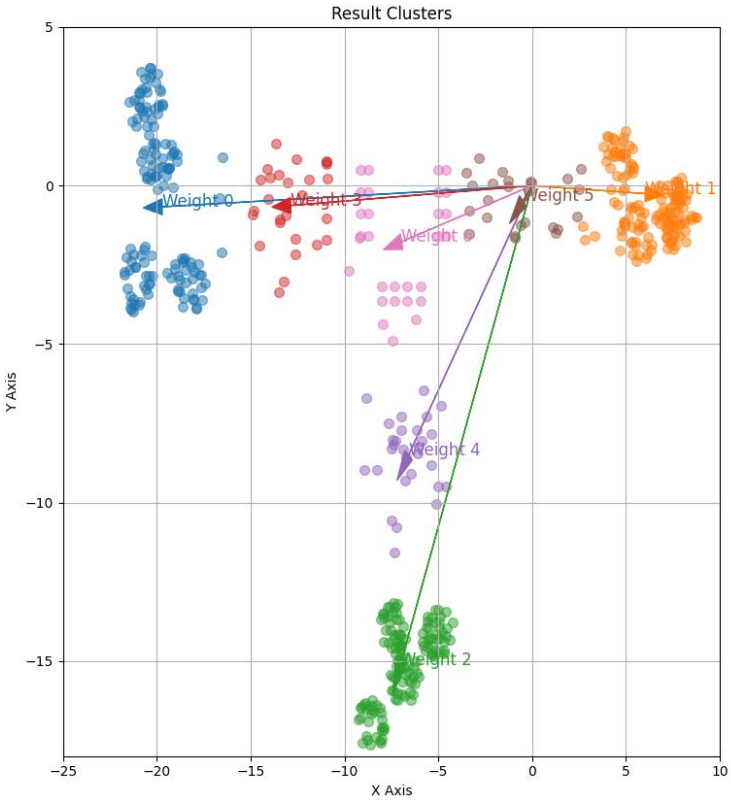

Рисунок 3.3. Результат кластеризації

Тепер можна розглянути результат по початково згенерованим точкам, вказавши з якою ймовірністю та чи інша точка належить певному кластеру. Як було

сказано вище, точки — це центри еліпсів, навколо яких генерувалася множина вхідних образів.

Всього кластерів: 7

Точка (-21.52, 0.61) належить до кластерів:

- Кластер 0: ймовірність 100.00

Точка (-22.36, 2.89) належить до кластерів:

- Кластер 0: ймовірність 100.00

Точка (-22.89, -3.14) належить до кластерів:

- Кластер 0: ймовірність 100.00

Точка (-20.00, -3.38) належить до кластерів:

- Кластер 0: ймовірність 100.00

Точка (-15, 0) належить до кластерів:

- Кластер 0: ймовірність 10.00
- Кластер 3: ймовірність 70.00
- Кластер 6: ймовірність 20.00

Точка (-15, -1.5) належить до кластерів:

- Кластер 0: ймовірність 5.00
- Кластер 3: ймовірність 65.00
- Кластер 6: ймовірність 30.00

Точка (8.55, -1.32) належить до кластерів:

- Кластер 1: ймовірність 100.00

Точка (4.92, 0.83) належить до кластерів:

- Кластер 1: ймовірність 100.00

Точка (6.10, -1.72) належить до кластерів:

- Кластер 1: ймовірність 100.00

Точка (0, -1.5) належить до кластерів:

- Кластер 1: ймовірність 30.00
- Кластер 5: ймовірність 50.00
- Кластер 6: ймовірність 20.00

Точка (8.03, -0.66) належить до кластерів:

- Кластер 1: ймовірність 100.00

Точка (0, 0) належить до кластерів:

- Кластер 1: ймовірність 5.00
- Кластер 5: ймовірність 75.00
- Кластер 6: ймовірність 20.00

Точка (-9.28, -18.43) належить до кластерів:

- Кластер 2: ймовірність 100.00

Точка (-7.44, -16.95) належить до кластерів:

- Кластер 2: ймовірність 100.00

Точка (-7.99, -15.17) належить до кластерів:

- Кластер 2: ймовірність 100.00

Точка (-5.50, -15.36) належить до кластерів:

- Кластер 2: ймовірність 100.00

Точка (-7, -9) належить до кластерів:

- Кластер 4: ймовірність 70.00
- Кластер 6: ймовірність 30.00

Точка (-8.5, -9) належить до кластерів:

- Кластер 4: ймовірність 75.00
- Кластер 6: ймовірність 25.00

Одержані результати дозволяють зробити висновок, що використання модифікованого алгоритму навчання мережі Кохонена дозволяє не лише якісно сегментувати вхідні еліптичні образи на основі їх внутрішньої структури, але й досягти суттєвого зменшення часу навчання мережі. Це особливо важливо у випадках обробки великої кількості вхідних даних або при роботі в умовах обмежених обчислювальних ресурсів.

Додатково, результати кластеризації підтвердили здатність мережі адаптуватися до нерівномірного розподілу даних, що свідчить про її гнучкість та стійкість до варіацій структури вхідних образів.

РОЗДІЛ 4
КЛАСТЕРИЗАЦІЯ НЕЧІТКИХ ОБРАЗІВ У НЕПЕРЕРВНИХ ПРОСТОРАХ ОЗНАК

4.1. Загальні положення та архітектура неперервної моделі

4.1.1. Особливості модифікації алгоритму кластеризації

У випадку кластеризації в неперервних просторах ознак кожен нечіткий образ $\boldsymbol{X^t}$ розуміється як область $\boldsymbol{S^t}$ (однозв'язна або багатозв'язна, замкнута або відкрита), в якій відома густина ймовірності $\rho^{\boldsymbol{t}}(\boldsymbol{x_1}, \boldsymbol{x_2}, \text{K}, \boldsymbol{x_N})$ перебування цього образу у відповідній точці простору $\boldsymbol{X} = (\boldsymbol{x_1}, \boldsymbol{x_2}, \text{K}, \boldsymbol{x_N}) \in \boldsymbol{S^t}$. При цьому інтегральна сума густини ймовірності по відповідній області для кожного образу має давати одиницю.

Алгоритм кластеризації нейронними мережами Кохонена, адаптований для випадку нечітких дискретних образів, слід модифікувати з врахуванням неперервності розподілу густини ймовірності для кожного образу:

1) початково в межах області розподілу образів випадковим чином генеруються нейрони Кохонена;
2) проводиться розділення області розподілу образів прямими (площинами), які є серединними перпендикулярами між кожною парою найближчих нейронів — діаграма Вороного [5]. У результаті формуються багатокутні (багатогранні) області активації відповідного нейрона Кохонена. В одну область активації можуть потрапити фрагменти різних образів. Вони утворюватимуть спільний кластер (Рис. 1);
3) встановлюються координати центрів мас кожного кластера з врахуванням розподілів густин ймовірностей образів, що туди потрапили;
4) вектори вагових коефіцієнтів нейронів Кохонена корегуються в напрямку центрів мас відповідних кластерів;
5) кроки 2), 3), 4) повторюється зі зменшенням коефіцієнту швидкості навчання

на кожній епосі до тих пір, поки положення нейронів Кохонена практично перестане змінюватися.

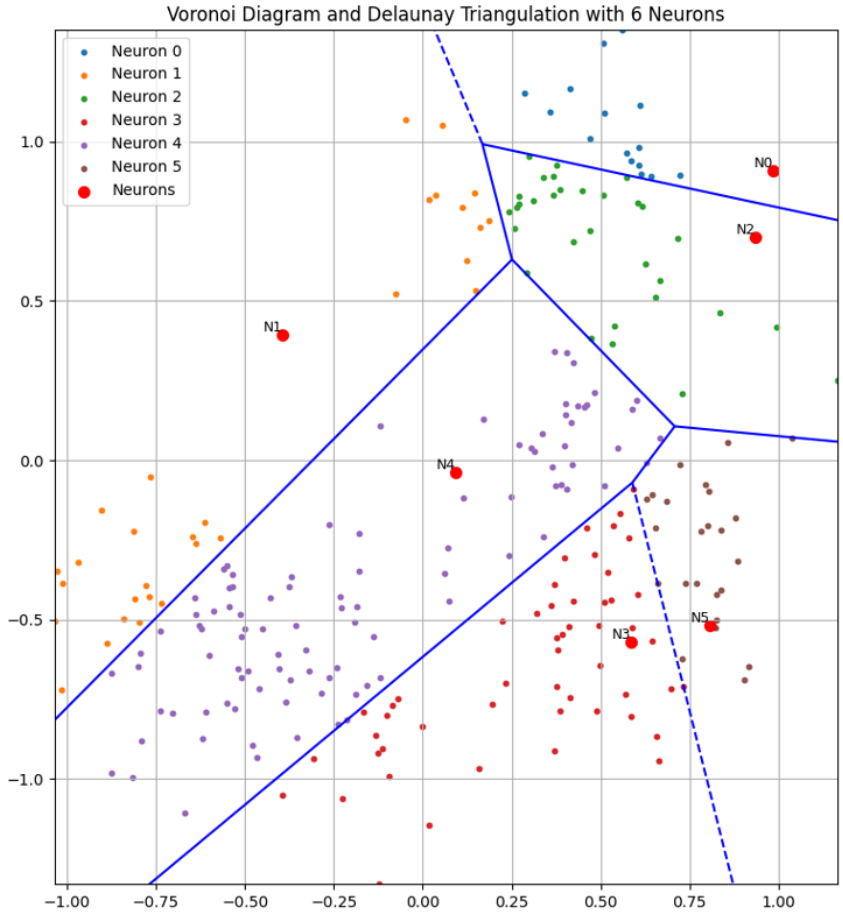

Рисунок 4.1. Початкове розділення області розподілу образів

Визначення координат центрів мас кожного кластера, що є багатозв'язною областю з кусково-неперервною функцією густини ймовірності є проблемою обчислювального характеру. Тому для інтегрування використовуються квадратурні формули по особливій системі вузлів, які початково в межах області розподілу образів генеруються випадковим чином. Після цього будується тріангуляція Делоне для сформованої системи вузлів та знаходяться точки перетину ребер тріангуляції з ребрами діаграми Вороного. Точки перетину — це фактично новозгенеровані точки, в яких густини обчислюються за формулою

$$\rho = \frac{l_1 \cdot \rho_2 + l_2 \cdot \rho_1}{l_1 + l_2},$$

де l_1 та l_2 — це відстані від точки перетину до відповідних вершин ребра тріангуляції, а ρ_1 та ρ_2 — це густини ймовірностей у вершинах ребра тріангуляції.

4.1.2. Тріангуляція Делоне та її особливості

Тріангуляція Делоне для множини точок P на площині — це така побудова трикутників, при якій жодна точка з цієї множини не розташовується всередині кола, що описане навколо будь-якого з трикутників, утворених у процесі тріангуляції. Такий підхід мінімізує появу малих кутів у трикутниках. Метод був розроблений Борисом Делоне у 1934 році.

У контексті цього визначення, **порожнє коло** — це коло, що проходить через три точки з початкової множини і не містить жодної іншої точки з цієї множини у своєму внутрішньому просторі (присутність на межі дозволена).

Критерій Делоне передбачає, що сітка трикутників є тріангуляцією Делоне тоді й тільки тоді, коли всі описані навколо трикутників кола не мають у своєму внутрішньому просторі інших точок із множини. Це є основним формулюванням для двовимірного простору, яке також має узагальнення на тривимірний випадок — там замість кіл розглядаються описані сфери.

Якщо всі точки лежать на одній прямій, побудувати тріангуляцію Делоне неможливо, адже в такому випадку тріангуляція як така не визначається. У ситуації, коли чотири точки розташовані на одному колі (наприклад, у формі прямокутника), існує два допустимі варіанти тріангуляції Делоне — обидва відповідають її умовам, оскільки розбиття чотирикутника може відбуватись двома різними діагоналями.

У ситуації, коли чотири точки розташовані на одному колі (наприклад, у формі прямокутника), існує два допустимі варіанти тріангуляції Делоне — обидва відповідають її умовам, оскільки розбиття чотирикутника може відбуватись двома різними діагоналями. Така ситуація називається невизначеною або неунікальною тріангуляцією Делоне, оскільки кілька тріангуляцій можуть одночасно задовольняти умову пустого описаного кола.

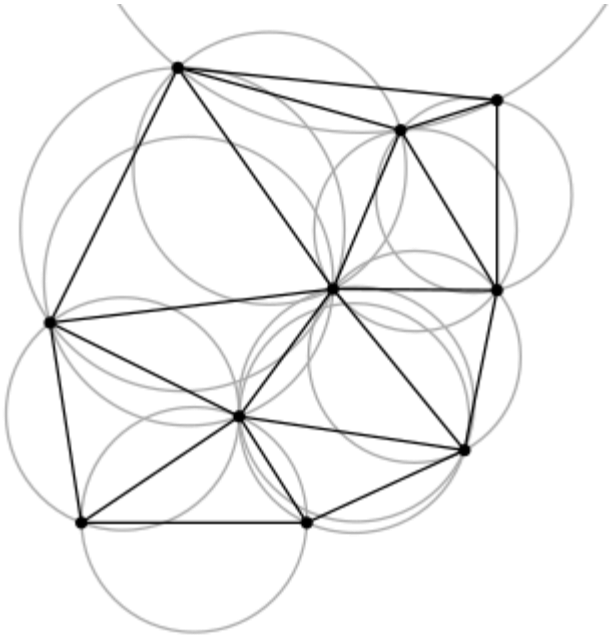

Рисунок 4.2. Тріангуляція Делоне і описані навколо трикутників кола

Властивості

Нехай n — кількість точок, а d — розмірність простору. Тоді для тріангуляції Делоне справедливі наступні твердження:

- Усі симплекси (тобто трикутники, тетраедри тощо залежно від розмірності) разом утворюють опуклу оболонку початкової множини точок.
- Кількість симплексів у тріангуляції Делоне не перевищує порядку $O(n^{d/2})$.
- У випадку площини (тобто при $d = 2$), якщо b точок утворюють межу опуклої оболонки, то загальна кількість трикутників у тріангуляції не перевищує $2n - 2 - b$. Крім того, присутня одна зовнішня "грань".
- На площині кожна точка в середньому бере участь у шести трикутниках.
- Тріангуляція Делоне на площині максимізує найменший з кутів серед усіх можливих тріангуляцій, тобто жодна інша тріангуляція не матиме більшого мінімального кута. Водночас вона не гарантує мінімального значення найбільшого кута.
- Коло описане навколо довільного трикутника тріангуляції не містить всередині жодних інших вхідних вершин.

- Якщо коло що проходить через дві вхідні точки не містить всередині жодних інших, тоді сегмент що з'єднує ці дві точки є ребром тріангуляції Делоне цих точок.
- Тріангуляція Делоне множини точок в *d*-вимірному просторі є проєкцією точок опуклої оболонки на (*d + 1*)-мірний параболоїд.
- Найближчий сусід *b* довільної точки *p* лежить на ребрі *bp* тріангуляції Делоне бо граф найближчих сусідів — підграф тріангуляції Делоне.

4.1.3. Діаграма Вороного

Діаграма Вороного — це тип поділу метричного простору на області, які формуються на основі відстаней до певного набору розташованих окремо точок. Ця концепція названа на честь українського математика Георгія Вороного.

Діаграми Вороного є дуже корисними структурами, вони відображають відношення відстаней та явища росту. Тому не дивно, що діаграми Вороного використовують для моделювання росту кристалів або великих об'єктів Всесвіту, а також для спостереження їх у природних об'єктах, таких як панцир черепахи або шия жирафа. Діаграми Вороного також є структурами даних, що розв'язують багато задач, таких як пошук найближчого сусіда або планування руху. Вивчення діаграм Вороного, їх математичних властивостей, обчислення та узагальнення було і залишається основним предметом обчислювальної геометрії [10].

Припустимо, що є *n* точок, розкиданих на площині, діаграма Вороного розбиває площину рівно на *n* клітинок, що охоплюють частину площини, яка є найближчою до кожної точки. Таким чином створюється теселяція, яка повністю покриває площину. Для ілюстрації на рисунку 4.3 зображено 100 випадкових точок і відповідну їм діаграму Вороного. Як можна побачити, кожна точка укладена в комірку, межі якої є точно рівновіддаленими між двома або більше точками. Іншими словами, вся область, укладена в комірку, знаходиться ближче до точки в комірці, ніж до будь-якої іншої точки [13].

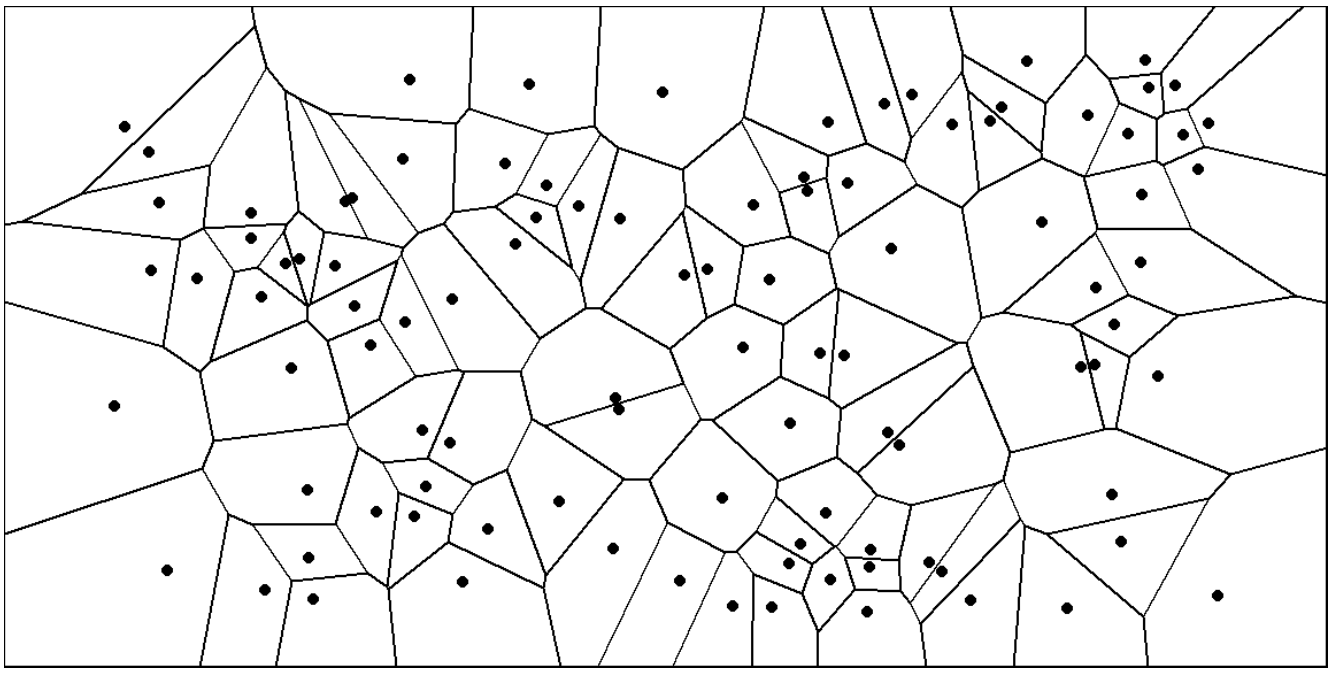

Рисунок 4.3. Діаграма Вороного для набору точок

Математичне означення

Нехай задано скінченну множину точок $S = \{p_1, p_2, \ldots, p_n\}$, яку називають сайтом або генераторами.

Тоді діаграма Вороного для цієї множини — це множина областей $V(p_i)$, де кожна область визначається так:

$$V(p_i) = \{x \in R^2 \mid \forall_j \neq i, \left||x - p_i|\right| \leq \left||x - p_j|\right|\},$$

де $\|\cdot\|$ — евклідова норма (відстань між точками)

Геометричні властивості

- Межі між комірками Вороного — це відрізки або промені, які лежать на перпендикулярних бісектрисах між точками.
- Всі точки на межі між $V(p_i)$ та $V(p_j)$ однаково віддалені від p_i та p_j.
- Якщо точка лежить у внутрішності $V(p_i)$, то вона ближча до p_i, ніж до будь-якої іншої точки $p_j \in S \setminus \{p_i\}$.

У R^d аналогічне означення теж діє: простір розбивається на області навколо точок, причому кожна точка належить області найближчого сайту.

Діаграми Вороного можна будувати за неевклідовими метриками, ваговими функціями (зважені діаграми Вороного) і навіть для кривих або областей.

4.2. Особливості програмної реалізації

У моделі генерується *N* нейронів, які ініціалізуються в просторі ознак. Кожен нейрон представлений ваговим вектором, що має ту саму розмірність, що й вхідні дані.

Діаграма Вороного будується для візуалізації поділу простору між нейронами. Кожна область Вороного відповідає одній нейронній точці (ваговому вектору) та містить усі точки простору, що ближчі до цього нейрона, ніж до інших.

Тріангуляція Делоне використовується для аналізу взаємного розташування нейронів у просторі та створення мережевої структури зв’язків між ними.

Перетини між тріангуляцією Делоне та діаграмою Вороного дозволяють знайти додаткові ключові точки в просторі. Ці точки створюють нові можливості для деталізації поділу та уточнення структури простору. Отримані точки перетину додаються до загального набору точок, розширюючи існуючу геометричну структуру. Це дозволяє уточнити просторову модель та виділити додаткові регіони.

Для кожної області Вороного формується окремий набір точок, що включає ваговий вектор нейрона, граничні точки області та додаткові точки, отримані з перетинів з тріангуляцією Делоне. Тріангуляція всередині кожної області виконується за допомогою алгоритму Делоне з бібліотеки *scipy.spatial.Delaunay*.

Далі для обчислення центру мас кожної області використовується тріангуляція Делоне. Це дозволяє поділити полігони на менші трикутники, для яких легко обчислити геометричні характеристики.

Програмна реалізація полягає в асинхронній моделі навчання, яка базується на оновленні даних в реальному часі, підлаштовуючи кожен раз центри мас кожної області та нейрони Кохонена.

Поточна модель мережі Кохонена використовує початкове розділення множини точок на кластери за допомогою діаграми Вороного. Програма генерує нейрони випадковим чином і розділяє область за допомогою серединних перпендикулярів між нейронами. Розділення відбувається за принципом найближчої точки для кожного з нейронів.

Обчислення центру мас кожної області на основі площ та центрів мас трикутників забезпечує точніше визначення середнього положення даних, що потрапляють у цей регіон. У процесі навчання нейрон Кохонена оновлює свій ваговий вектор у напрямку до цього центру мас, забезпечуючи поступову адаптацію до розподілу даних у своїй області впливу.

На рисунку 4.4 наведено результат тріангуляції Делоне для різних областей активації, де нейрони Кохонена вже фактично збігаються з центрами мас (останні ітерації).

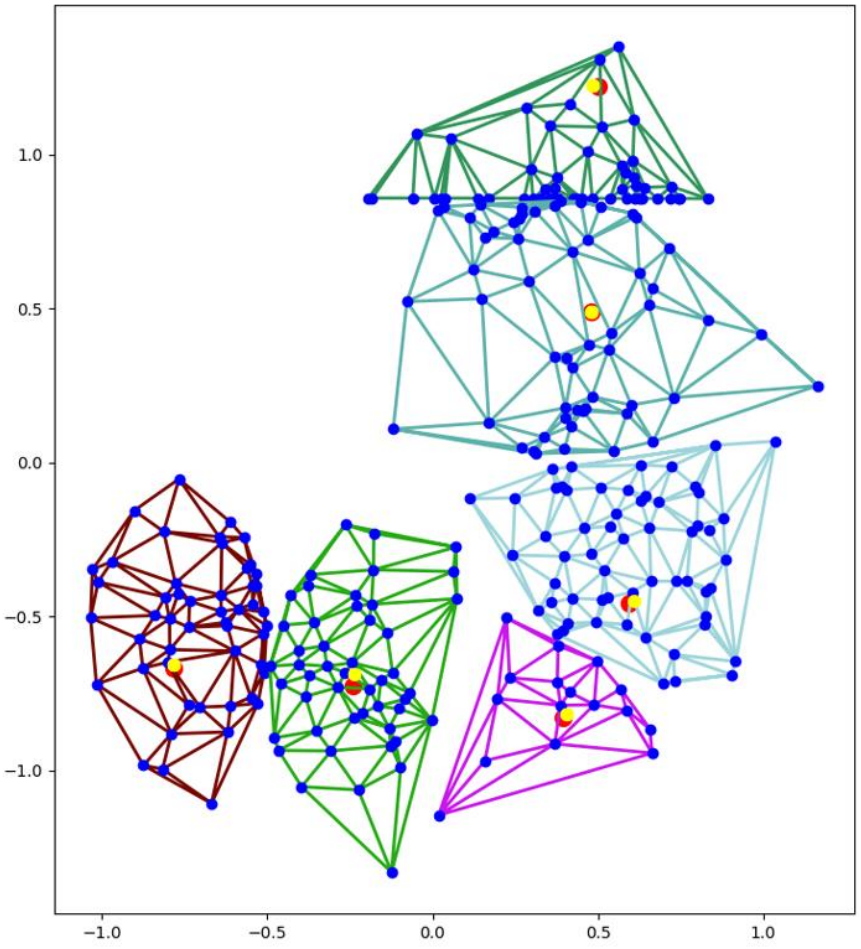

Рисунок 4.4. Тріангуляція Делоне для областей активації

На рисунку 4.5 наведено фрагмент коду програмної реалізації для розрахунку густин кожної з вершин трикутників та площ цих трикутників для розрахунку сумарного інтегралу, який використовується в формулі центрів мас.

```
for i, simplex in enumerate(tri.simplices):
    # Triangle vertices
    p1, p2, p3 = cluster_points[simplex]

    # Densities for each vertex
    d1 = get_density_for_vertex(p1, data)
    d2 = get_density_for_vertex(p2, data)
    d3 = get_density_for_vertex(p3, data)

    # Area of the triangle
    area = 0.5 * abs(np.linalg.det(np.array([
        [p1[0], p1[1], 1],
        [p2[0], p2[1], 1],
        [p3[0], p3[1], 1]
    ])))

    # Density mean in the triangle
    avg_density = (d1 + d2 + d3) / 3

    # Triangle center
    x_c = (p1[0] + p2[0] + p3[0]) / 3
    y_c = (p1[1] + p2[1] + p3[1]) / 3

    # Integral for the triangle
    integral = area * avg_density
    integrals.append(integral)

    # Weighted contributions for center of mass
    x_weighted += integral * x_c
    y_weighted += integral * y_c

# All triangles sum
total_integral = sum(integrals)

# Center of mass
x_cm = x_weighted / total_integral
y_cm = y_weighted / total_integral

print(f"Neuron {neuron_idx}: Total Integral = {total_integral:.4f}, Center of Mass = ({x_cm:.4f}, {y_cm:.4f})")
```

Рисунок 4.5. Фрагмент коду програми

Опис алгоритму

1. Для кожного трикутника беруться координати його вершин: *p1*, *p2*, *p3*.
2. Аналогічно з бази даних отримуються густини для кожної із вершин трикутника *d1*, *d2*, *d3*.
3. Обчислюється площа трикутника та середнє значення густини.
4. Рахуються координати центра трикутника як середнє арифметичне.
5. Обчислюється локальний внесок в масу трикутника (інтеграл).
6. Обчислюється зважений внесок центра мас трикутника. Сума всіх таких внесків дає чисельник для глобального центра мас.

7. Для того, щоб отримати центр мас, сумарний зважений внесок центрів мас усіх трикутників ділиться на загальну масу (суму всіх локальних мас).

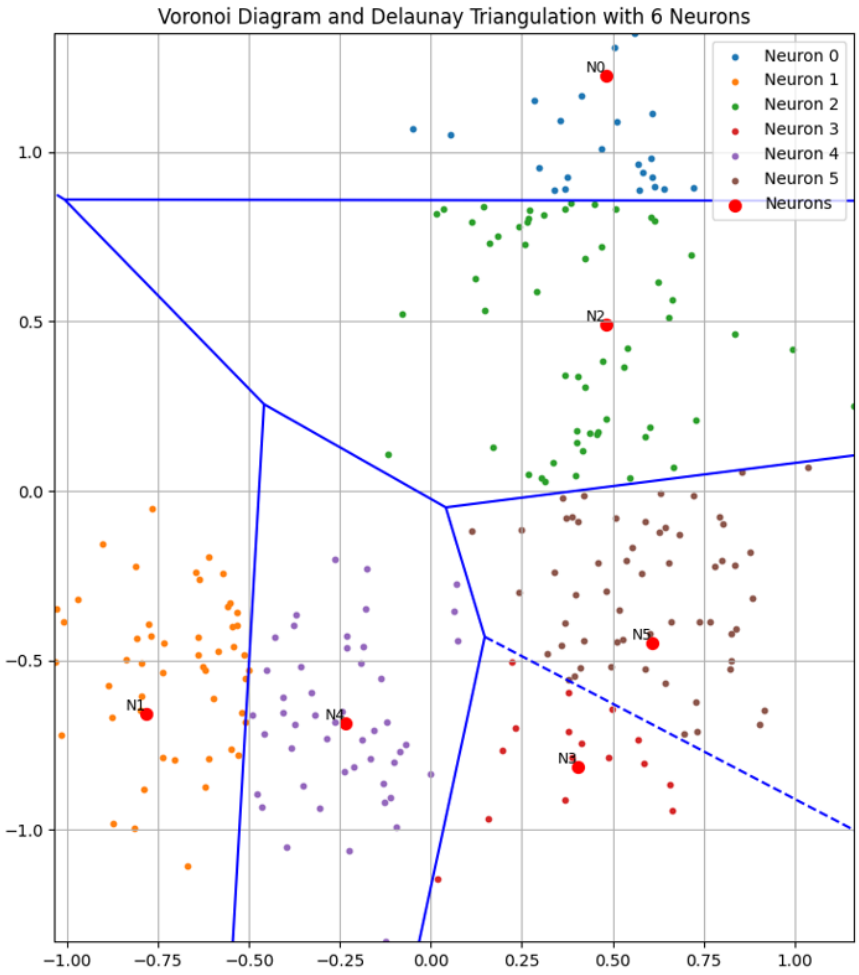

Рисунок 4.6. Кінцевий результат кластеризації (25 епох)

ВИСНОВКИ

Проведене дослідження продемонструвало, що модифікований підхід до кластеризації нечітких даних на основі нейронних мереж Кохонена є ефективним інструментом для роботи в умовах високої невизначеності. Метод спрацював як у дискретному, так і в неперервному випадках, що дозволило зробити висновок про його адаптивність та практичну гнучкість.

У процесі дослідження було використано спільну методику для обох типів вхідних даних, із залученням додаткових геометричних рішень, зокрема, побудови тріангуляції Делоне для просторової організації даних. Такий підхід дозволив досягти високої якості кластеризації без необхідності розділення методів залежно від типу даних.

Результати експериментів засвідчили, що запропонована методика є універсальною та може бути ефективно використана в різноманітних прикладних задачах, де важлива здатність алгоритму працювати з нечіткими, варіативними або частково невизначеними даними. Запропонований підхід відкриває нові можливості для розв’язання задач, у яких традиційні методи кластеризації не забезпечують належної точності або стійкості.

Здобуті під час роботи результати акцентують увагу на важливості поєднання нейромережевих алгоритмів із методами геометричної обробки даних, що створює умови для побудови більш гнучких та адаптивних моделей кластеризації. Така інтеграція дозволила не лише забезпечити стійкість алгоритму до варіативності вхідних параметрів, а й суттєво зменшити вплив шуму та нечіткості у даних.

Отримані результати свідчать про потенціал подальшого розвитку даного підходу, зокрема в напрямі динамічної кластеризації, адаптивного навчання та розширення на багатовимірні простори.

Таким чином, розроблена методика може стати основою для створення інтелектуальних систем аналізу інформації, які потребують високої адаптивності до змінних умов та нечіткої природи вхідної інформації.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Банацький-Шуманський М. В., Сяський В. А. Адаптація нейронних мереж Кохонена для кластеризації нечітко визначених образів. *Інформаційні технології в професійній діяльності* : матеріали XVII Всеукраїнської науково-практичної конференції (Рівне, 5 листопада 2024 р.). Рівне : РВВ РДГУ. 2024. С. 125–129.

2. Банацький-Шуманський М. В., Сяський В. А. Кластеризація нечітких образів у неперервних просторах ознак конкурентними нейронними мережами Кохонена. *Підготовка педагогів до професійної діяльності в умовах змішаного навчання* : матеріали IV Всеукраїнської науково-практичної конференції (Рівне, 14 травня 2025 р.). Рівне : РВВ РДГУ. 2025. С. XXX.

3. Лещинський О. Л., Іщенко А. О. Використання нейромереж у процесі інтелектуального (кластерного) аналізу даних. *Економіка і суспільство*. 2017. Вип. № 11. С. 578–581.

4. Марченко О. О., Россада Т. В. Актуальні проблеми Data Mining: Навч. посіб. для студентів факультету комп'ютерних наук та кібернетики. Київ : КНУ ім. Т. Шевченка, 2017. 150 с.

5. Aurenhammer F., Klein R., Lee D.-T. *Voronoi Diagrams and Delaunay Triangulations*. World Scientific Publishing Company, 2013. 348 с.

6. Bezdek J. C., Keller J., Krisnapuram R., Pal N. R. Fuzzy Models and Algorithms for Pattern Recognition and Image Processing. — Boston: Springer, 2005. 776 с.

7. Jolliffe I. T., ред. Principal Component Analysis. — 2-е вид. — Springer, 2002. 487 с.

8. Kaufman L., Rousseeuw P. J. Finding Groups in Data: An Introduction to Cluster Analysis. — Hoboken: Wiley, 2005. 342 с.

9. Kohonen T. Self-Organizing Maps. 3rd ed. Berlin; Heidelberg; New York; Barcelona; Hong Kong; London; Milan; Paris; Singapore; Tokyo : Springer, 2001. 501 с.

10. Okabe A., Boots B., Sugihara K., Chiu S. N., ред. Spatial Tessellations: Concepts and Applications of Voronoi Diagrams. — 2-е вид. — Wiley, 2000. 671 с.

11. Provost F., Fawcett T., ред. Data Science and Its Relationship to Big Data and Data-

Driven Decision Making. — Big Data, 2013. — Т. 1, № 1. С. 51–59.

12. Tan P.-N., Steinbach M., Kumar V., ред. Introduction to Data Mining. — 2-е вид. — Pearson, 2018. С. 5-6.

13. The fascinating world of Voronoi diagrams. URL: https://towardsdatascience.com/the-fascinating-world-of-voronoi-diagrams-da8fc700fa1b/

14. Valente de Oliveira J., Pedrycz W., ред. Advances in Fuzzy Clustering and its Applications. — Чічестер: Wiley, 2007. 320 с.

15. Witten I. H., Frank E., Hall M. A., Pal C. J., ред. Data Mining: Practical Machine Learning Tools and Techniques. — 4-е вид. — Morgan Kaufmann, 2016. 640 с.