Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет Інформаційно-аналітичних технологій та менеджменту
(повна назва)
Кафедра Інформатики
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти другий (магістерський)

ДОСЛІДЖЕННЯ МЕТОДУ ПРАВДОПОДІБНОЇ КЛАСТЕРИЗАЦІЇ ДАНИХ НА ОСНОВІ ЕВОЛЮЦІЙНОГО МЕТОДУ БОЖЕВІЛЬНИХ КОТІВ
(тема)

Виконав:
студент 2 курсу, групи ІНФм-20-1
Москаленко В.В.
(прізвище, ініціали)

Спеціальності 122 Комп'ютерні науки
(код і повна назва спеціальності)

Тип програми освітньо-професійна

Освітня програма Інформатика
(повна назва освітньої програми)

Керівник доц. Шафроненко А.Ю.
(посада, прізвище, ініціали)

Допускається до захисту

Зав. кафедри _____ Кобилін О.А.
(підпис) (прізвище, ініціали)

2021 р.

Харківський національний університет радіоелектроніки

Факультет Інформаційно-аналітичних технологій та менеджменту
(повна назва)

Кафедра Інформатики
(повна назва)

Рівень вищої освіти другий (магістерський)

Спеціальність 122 Комп'ютерні науки
(код і повна назва)

Тип програми освітньо-професійна

Освітня програма Інформатика
(повна назва освітньої програми)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
«____»_____ 2021 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

студентові Москаленко Валерії Володимирівні
(прізвище, ім'я, по батькові)

1. Тема роботи Дослідження методу правдоподібної кластеризації даних на основі еволюційного методу божевільних котів

затверджена наказом по університету від 22» жовтня 2021 року № 1574Ст.
2. Термін подання студентом роботи до екзаменаційної комісії ___ _____ 2021 р.
3. Вихідні дані до роботи математичні моделі кластеризації даних, перелік використовуваних програмних засобів: теоретичні відомості про методи правдоподібної кластеризації даних.

4. Перелік питань, що потрібно опрацювати в роботі
1. Огляд підходів нечіткої кластеризації даних
2. Огляд методів правдоподібного підходу нечіткої кластеризації даних
3. Математична модель еволюційного методу божевільних котів
4. Практична оцінка методу

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (п.5 включається до завдання за рішенням випускової кафедри) актуальність проблеми потокового режиму в нечіткій кластеризації даних, постановка задачі, тестові дані.

6. Консультанти розділів роботи (п.6 включається до завдання за наявності консультантів згідно з наказом, зазначеним у п.1)

Найменування розділу	Консультант (посада, прізвище, ім'я, по батькові)	Позначка консультанта про виконання розділу	
		підпис	дата
Консультант з дотримання діючих стандартів та норм	Доцент Бєлова Н.В.		

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів роботи	Терміни виконання етапів роботи	Примітка
1	Отримання завдання на кваліфікаційну роботу	22.10.2021	
2	Аналіз завдання, підбір літератури	22.10.21-24.10.21	
3	Аналіз літератури з досліджуваної проблеми	25.10.21-27.10.21	
4	Аналіз еволюційних методів кластеризації	28.10.21-30.10.21	
5	Аналіз методу котячої зграї	31.10.21-03.11.21	
6	Вибір алгоритму і середовища програмної реалізації	04.11.21-06.11.21	
7	Оформлення пояснювальної записки	07.11.21-19.11.21	
8	Перевірка на плагіат	14.12.2021	
9	Рецензування	14.12.2021	
10	Підготовка презентації та доповіді	14.12.2021	
11	Занесення роботи в електронний архів	15.12.2021	
12	Попередній захист кваліфікаційної роботи	15.12.2021	

Дата видачі завдання _____ _____ 2021 р.

Студент _____
(підпис)

Керівник роботи _____ _____
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ/ABSTRACT

Пояснювальна записка до кваліфікаційної роботи: 57 с., 2 табл., 11 рис., 41 джерело.

НЕЧІТКА КЛАСТЕРИЗАЦІЯ, ТЕОРІЯ ПРАВДОПОДІБНОСТІ, ЕВОЛЮЦІЙНІ МЕТОДИ ОПТИМІЗАЦІЇ, КОТЯЧА ЗГРАЯ, РЕЖИМ ТРАСУВАННЯ, РЕЖИМ ПОШУКУ, РІВЕНЬ НАЛЕЖНОСТІ.

Об’єктом дослідження є правдоподібна кластеризація даних.

Метою дослідження є огляд та аналіз правдоподібних методів нечіткої кластеризації даних, що може швидко знаходити екстемуми (центри) кластерів, незалежно від кількості даних, що надходять.

Проведено дослідження методів правдоподібної нечіткої кластеризації. Досліджено методи, засновані на еволюційному методі божевільних котів.

У результаті роботи здійснена програмна реалізація кластеризації за допомогою методу божевільних котів.

FUZZY CLUSTERING, CREDIBILITY THEORY, EVOLUTIONARY METHODS OF OPTIMIZATION, CATS SWARM, TRACING MODE, SEEKING MODE, MEMBERSHIP LEVEL.

The object of the research is the credibilistic clustering.

The aim of the research is to review and analyze credibilistic methods which can quickly find the extremes (centers) of clusters, regardless of the amount of data received.

Methods of credibilistic fuzzy clustering are reviewed. Methods based on the evolutionary method of crazy cats have been studied.

As a result of implemented software implementation of the clustering using the method of crazy cats.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

CSO – Оптимізація котячої зграї (Cat Swarm Optimization)

CSOC – Кластеризація за допомогою методу оптимізації котячої зграї (Cat Swarm Optimization Clustering)

DM – Добування даних, також глибинний аналіз даних (Data Mining)

FCM – методу нечітких с-середніх

KHM –Алгоритм k-гармонійних середніх (K-Harmonic Means algorithm)

KCSOC – Алгоритм k-гармонійних середніх за допомогою методу оптимізації котячої зграї (K-harmonic means Cat Swarm Optimization Clustering)

SM – режим пошуку локальних екстремумів

TM – режим погоні

ВСТУП

Обробка великих об’ємів даних має застосування всюди. Задача кластеризації, на відміну від класифікації, має значно більше застосування – обробка даних із метою визначення рис схожості нових невідомих раніше даних із відомими. В решті решт схожа задача була вирішена Д.М. Менделеєвим, що надало можливість великого ривку розвитку хімії.

Інтелектуальний аналіз даних, вилучення цінної інформації, такої як тенденції, особливості або закономірності, із прихованих прогнозних даних – це область, яка знаходиться на перетині статистики, машинного навчання, розпізнавання шаблонів та візуалізації даних, щоб забезпечити новий погляд на аналіз даних. Проблема кластеризації визначається як проблема класифікації сукупності об’єктів у набір природних кластерів без будь-яких апріорних знань. Кластерний аналіз, метод класифікації без нагляду, став важливою технікою в аналізі даних.

Повідомлялося про багато підходів до кластеризації, які можна розділити на дві категорії: ієрархічні та роздільні [1]. Серед них популярний алгоритм k-середніх (k-means), типовий ітераційний метод підйому на гору. Однак основними недоліками k-середніх є те, що воно часто застряє на локальних мінімумах, і його результат значною мірою залежить від вибору початкових центрів кластерів. Алгоритм k-гармонійних середніх (KHM), інший метод кластеризації на основі центру, запропонований Чжаном і модифікований Хаммерлі та Елканом для вирішення проблеми ініціалізації k-середніх. Показано, що KHM по суті нечутливий до ініціалізації кластерних центрів. Однак у деяких випадках він має тенденцію зближуватися до локального оптимуму. Щоб подолати недоліки k-середніх, дослідники розробили багато вдосконалених методів кластеризації [1]. Останнім часом для вирішення проблеми кластеризації використовуються деякі метаевристичні алгоритми, такі як генетичні алгоритми, імітований відпал,

оптимізація рою частинок і пошук табу. Щоб ефективно застосувати метаевристичні алгоритми до кластерного аналізу, покращити кластеризаційне рішення та прискорити швидкість зходження, дослідники поєднали метаевристичні алгоритми з алгоритмом k-середніми.

Оптимізація котячої зграї (CSO), нещодавня методика, про яку вперше повідомили Чу і Цай [2], моделює поведінку кішок для вирішення проблеми оптимізації. У цій статті наша мета – представити CSO для вирішення проблеми кластеризації та вивчити застосовність CSO до векторів даних кластеризації. Наскільки нам відомо, це перше зареєстроване дослідження, яке розглядає використання CSO в кластерному аналізі. Оскільки KHM перевершує k-середніх та привабливий з точки зору обчислень, ми робимо спробу поєднати його з CSO та прийняти однокроковий алгоритм KHM як операцію покращення, щоб підвищити продуктивність алгоритму кластеризації. В результаті ми розробляємо два методи для вирішення проблеми кластеризації: кластеризація за допомогою методу оптимізації котячої зграї (CSOC) і алгоритм k-гармонійних середніх за допомогою методу оптимізації котячої зграї (KCSOC). Експериментальні результати на реальних наборах даних наведено для ілюстрації ефективності двох запропонованих алгоритмів.

Актуальність дослідження полягає у вирішенні проблеми відшукання глобального екстремуму цільової функції правдоподібної нечіткої кластеризації введено модифікацію ройового алгоритму зграй божевільних котів, що об'єднує в собі переваги еволюційних алгоритмів та глобального випадкового пошуку.

1 АНАЛІЗ ОСНОВНИХ МЕТОДІВ КЛАСТЕРИЗАЦІЇ ДАНИХ

1.1 Основні поняття та постановка задачі кластеризації

Кластеризація – це одна з найбільш важливих проблем навчання без вчителя. Як і кожна проблема такого типу, вона має справу з виявленням прихованої структури у сукупності даних.

Об’єднання в кластери – це процес організації об’єктів у групи, члени яких подібні до певної міри, а кластер – це сукупність об’єктів, близьких між собою у смислі деякої міри схожості. У просторі змінних кластери являють собою скупчення точок (об’єктів) різної форми.

Введемо кілька основних понять.

Метрика відстані – множливо, найважливіше поняття, що використовуються в кластеризації. Саме за допомогою відстані між вхідними векторами визначається їх схожість аботрізниця. Є багато способів визначення відстані: евклідова, Хеммінга, відстань Махаланобіса та ін. Вибір способу підрахунку відстані залежить від природи обєктів що використовуються і беспосередньо впливає на результат.

Цільова функція – це функція, мінімізація якої дає розвязок задачі кластеризації. Послідовність дій, що реалізує пошук мінімума цільової функції, є алгоритмом кластеризації.

Матриця розбиття – основний результат неієрархічнох кластеризації. Матриця розбиття представляє собою таблицю, де кожна ячейка містить значення функції належності даного вектора заданому кластеру. На основі цієї матриці отримується результуюче розбиття. В більшості алгоритмів крім матриці належності в якості результату породжується множина центрів кластерів. Центр кластеру – це вектор, степінь належності якого заданому кластеру максимальна. Як правило, центрів кластерів немає у вхідній множині.

Набір обмежень пов'язаний з умовами, що накладаються на значення елементів матриця належності. Ці обмеження визначаються алгоритмом, що використовується для кластеризації.

Кластерний аналіз займає одне з центральних місць серед методів аналізу даних і являє собою сукупність підходів, методів і алгоритмів, призначених для знаходження деякого розбиття досліджуваної сукупності об'єктів на підмножини щодо подібних, схожих між собою об'єктів. При цьому для виділення таких підмножин, які отримали спеціальну назву «кластер», які так само іноді називають «таксонами» або просто класами, служить лише неформальне припущення про те, що об'єкти, які належать до одного кластеру, повинні мати більшу схожість між собою, ніж з об'єктами з інших кластерів.

Задача кластеризації відрізняється від задачі класифікації відсутність попередніх відомостей щодо належності даних до підмножин.

Результат кластеризації суттєво залежить від даних, їх характеру, характеру подання даних та ін., що визначає спеціалізованість методів. Слід зауважити, що для одних і тих самих даних результат кластеризації може змінюватись не тільки при зміні методу кластеризації та кількості кластерів, а й при виборі початкових центрів.

На вхід подаються дані у вигляді багатовимірних векторів даних:

$$X = \{x(1), \dots, x(k), \dots, x(N)\}, \tag{1.1}$$

де k – у загальному випадку відповідає номеру спостереження, кортежу, об’єкту вихідного масиву даних.

Результатом кластеризації є розподіл даного масиву на m кластерів Cl_j, заданий центроідом $Cl_j \in R^n, j = 1,2, \dots, m$ та рівнем належності $0 \leq U_j(k) \leq 1$ кожного об’єкту $x(k)$ до кожного кластера Cl_j. Якщо кластери неперетинні, то це чітка кластеризація, а якщо такого обмеження немає, то мова йде про нечітку кластеризацію.

В загальному вигляді кластеризація відбувається в кілька етапів:

Етап 1. Отримання, відбір даних для кластеризації.

Етап 2. Підготовка даних. Визначення множини змінних, нормалізація значень, вилучення чи заповнення відсутніх значень та ін. в разі необхідності.

Етап 3. Обчислення міри подібності між об’єктами.

Етап 4. Застосування методу кластеризації для створення кластерів – груп схожих об’єктів.

Етап 5. Надання результату аналізу.

Після отримання результату зазвичай йде коригування методу обчислення міри подібності та кластеризації та подальше повторення етапів 3-4 до отримання оптимального результату.

Для об’єднання об’єктів у групи необхідно ввести поняття «схожості», «відстані», або міри подібності об’єктів. Міра подібності обчислюється для пари векторів даних – значень змінних чи атрибутів, що описують об’єкт – для двох конкретних об’єктів вибірки.

Існують різні метрики, за якими можна обрахувати міру подібності, але більшість із них працює із чисельними значеннями, або навіть із даними, зведеними до певного діапазону (зазвичай $[0,1]$ або $[-1,1]$), що дозволяє врівноважити внесок атрибутів. Необхідність в нормалізації виникає з-за розбіжності в характері даних – це можуть бути значення не тільки різних діапазонів (наприклад, кількість років та зріст), а й різних шкал (зріст та стать, або дата народження).

Найбільш поширеними серед них є:

– евклідова відстань

Відома із школи геометрична відстань в багатовимірному просторі, де розмірність простору – кількість атрибутів):

$$\rho(x, x') = \sqrt{\sum_i^n (x_i - x_i')^2}. \quad (1.2)$$

– квадрат евклідової відстані

Використовується для збільшення вираженості віддаленості об’єктів один від одного:

$$\rho(x, x') = \sum_i^n (x_i - x_i')^2. \quad (1.3)$$

– манхетенська відстань, або відстань міських квадратів

Обчислюється як середнє різниць по координатах. Зазвичай надає схожі результаті із евклідовою відстанню, але зменшується значимість викидів, бо немає підведення до квадрату. Обраховується як:

$$\rho(x, x') = \sum_i^n |x_i - x_i'|. \quad (1.4)$$

– відстань Чебишева

Відображає найбільшу відмінність між об’єктами за координатами:

$$\rho(x, x') = \max_i |x_i - x_i'|. \quad (1.5)$$

– ступенева відстань

Гнучка до налаштування чутливості відстань, загальний випадок Евклідової відстані. Обчислюється за формулою:

$$\rho(x, x') = \sqrt[r]{\sum_i^n (x_i - x_i')^p}, \quad (1.6)$$

де r – параметр відповідальний за прогресивне зважування великих відстаней між об’єктами;

p – параметр відповідальний за поступове зважування різниць за окремими координатами.

Дані параметри задаються вручну. Якщо $r = p = 2$, то маємо Евклідову відстань.

– відстань Мінковського

Відстань $D\big(x(k), w_j\big)$ в метриці L^p зазвичай обраховується як:

$$D^p\big(x(k), w_j\big) = \big\|x_i(k) - w_{ji}\big\|_{L_p}^p = \Big(\sum_{i=1}^{n} \big|x_i(k) - w_{ji}\big|^p\Big)^{\frac{1}{p}},\ p \geq 1, \quad (1.7)$$

де $x_i(k)$ – i-та компонента вектора $x(k)$;

w_{ji} – i-та компонента вектора w_j.

1.2 Класифікація методів кластеризації

Оскільки поняття «кластеру» не може бути точно визначено, то це є однією з причин чому існує так багато різних методів кластеризації[4]. Але є і спільна риса — це об'єднання схожих об'єктів у групи. Однак, різні дослідники використовують різні моделі кластерів і для кожної з цих моделей можуть бути застосовані різні алгоритми. Поняття кластера, які отримуються у різних алгоритмах, різняться властивостями. Розуміння цих «кластерних моделей» є ключовим для розуміння відмінностей між різними алгоритмами. Типовими кластерними моделями є:

– моделі зв'язності. Наприклад, ієрархічна кластеризація або таксономія будуються на основі відстані між вузлами;

– центроїдні моделі. Наприклад, метод k-середніх (k -means) представляє кожен кластер єдиним усередненим вектором;

– статистичні моделі. Кластери будуються ґрунтуючись на статистичних розподілах. Таких як багатовимірний нормальний розподіл з допомогою EM-алгоритму;

– моделі засновані на щільності. Наприклад, в DBSCAN і в OPTICS кластери визначаються як зв'язані області відповідної щільності у просторі даних;

– групові моделі. Деякі алгоритми не забезпечують вдосконалену модель для своїх результатів, а просто описують групування об'єктів;

– графові моделі. Поняття кліки (така підмножина вершин, в якій кожна пара вершин з'єднана ребром) у графі слугує прототипом кластеру. Пом'якшення вимоги до повної зв'язності (тобто, частина ребер може бути відсутня) призводить до поняття відомого як квазі-кліка. Вони будуються алгоритмом HCS;

– нейронні моделі. Найбільш відомою моделлю нейронної мережі з навчанням без учителя є нейронна мережа Кохонена. Ці моделі, як правило, можна охарактеризувати як схожі на одну або подібні якійсь з наведених вище моделей, включаючи моделі у підпросторах, коли нейронні мережі реалізують метод головних компонент або аналіз незалежних компонент.

«Кластеризацією» зазвичай вважають такий набір кластерів, які містять усі об'єкти набору даних. Додатково, можна розглянути відношення між кластерами. Наприклад, ієрархію вкладеності кластерів один у одного. Грубо можна виділити такі кластеризації:

– жорстка кластеризація. Кожен об'єкт або належить кластеру або ні;

– м'яка кластеризація (також нечітка кластеризація). Кожен об'єкт належить кожному кластеру до певної міри. Наприклад, це ймовірність належності кластеру.

Серед них виділяють декілька доладних:

– жорстке розбиття на кластери. Кожен об'єкт належить рівно одному кластеру;

– жорстке розбиття на кластери з викидами. Об'єкт може не належати жодному кластеру і розглядається як викид;

– кластери з перетином. Об'єкт може належати більш ніж одному кластеру;

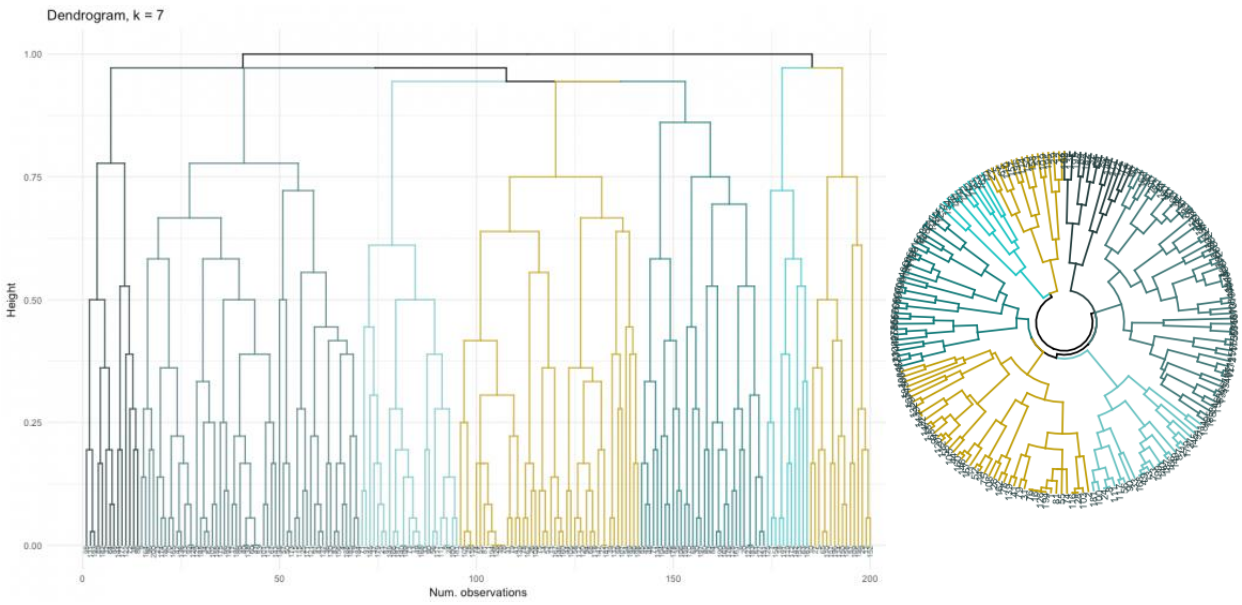
– ієрархічна кластеризація. Якщо об'єкт належить нащадку, то він також належить і предку;

– підпросторова кластеризація. Хоч кластери і можуть перетинатись, проте в межах визначеного підпростору кластери не перетинаються. Для прикладу дивись SUBCLU.

Чіткої та однозначної класифікації методів немає, але є кілька властивостей та особливостей алгоритмів, що можуть бути для цього використані.

Ієрархічні та плоскі алгоритми.

Ієрархічні алгоритми, або алгоритми таксономії відрізняються системою вкладеного розбиття вибірки на непересічні кластери. Результатом таких методів є дерево кластерів із всією сукупністю даних в якості кореня та листям як найбільш дрібними кластерами. Відображає розбиття так звана дендрограма (рис. 1.1).



а) б)

Рисунок 1.1 – Приклад дендрограми у звичайному (а) та круговому (б) варіанті

Розрізняють агломеративні та дивизимні ієрархічні методи.

Ієрархічні агломеративні методи (Agglomerative Nesting, AGNES). Ця група методів характеризується послідовним об’єднанням вихідних елементів і відповідним зменшенням числа кластерів. На початку роботи алгоритму всі об’єкти є окремими кластерами. На першому кроці найбільш схожі об’єкти поєднуються в кластер. На наступних кроках об’єднання триває доти, поки всі об’єкти не будуть становити один кластер.

Ієрархічні дивизимні (ділені) методи (Divisive Analysis, DIANA). Ці методи є логічною протилежністю агломеративним методам. На початку роботи алгоритму всі об’єкти належать одному кластеру, який на наступних кроках ділиться на менші кластери, у результаті утворюється послідовність груп, що розщеплюються.

Плоскі, або неієрархічні алгоритми будують одне розбиття на певну кількість кластерів.

Серед неієрархічних методів кластеризації особливої уваги заслуговують ітеративні методи. Вони працюють за наступним алгоритмом:

Крок 1. Обираються початкові центри кластерів (цей крок може сильно вплинути на час виконання та результат).

Крок 2. Об’єкти розподіляються між кластерами (наприклад, до кластеру із найближчим центром до об’єкту).

Крок 3. Обчислюються нові центри кластерів як центри тяжіння.

Крок 4. Кроки 2 і 3 повторюються доти, поки не перестануть змінюватись кластери, або виконанні інших умов зупинки алгоритму (наприклад, за кількістю ітерацій чи зміна центра кластера менша за певний поріг).

Чіткі та нечіткі методи.

Чіткі методи в результаті надають сукупність неперетинних кластерів, тобто один об’єкт кластеру належить тільки одному кластеру. Нечіткі методи такого обмеження не мають і кожному об’єкту ставлять у відповідність інформацію про ступінь відповідність до кластерів. Для групи методів це значення за сенсом схоже із ймовірність належності об’єкта кластеру.

Відомим прикладом Чітких методів є метод k-середніх, або k-means. Це ітераційний метод, який розділяє набір даних на k (неперетинних) кластерів. Задача методу виділити кластери так, щоб об’єкти всередині кластера були якомога схожі, а об’єкти різних кластерів якомога різними (далекими). Для цього об’єкти кластеру призначаються таким чином, щоб сума квадратів відстані між об’єктами і центром кластера (середнє арифметичне всіх об’єктів, які належать до цього кластера) була мінімальною. Чим менше ми маємо варіацій всередині кластерів, тим однорідніші (подібніші) об’єкти в одному кластері. Алгоритм даного методу:

Крок 1. Вибрати випадковим чином k об’єктів як початкові центри кластерів.

Крок 2. Обчислити суму квадратів відстані між об’єктами та центрами кластерів.

Крок 3. Призначити кожен об’єкт належним до найближчого кластеру (кластеру із найближчим центром).

Крок 4. Обчислити нові центри кластерів як середнє значення всіх об’єктів кластеру.

Крок 5. Повторювати кроки 2-4 до виконання умов зупинки (виходу).

Умов зупинки зазвичай декілька. Це не тільки визначеність кластерів (яка може визначатися як незмінність розподілу об’єктів на кластери), так й обмеження за кількістю ітерацій, що робиться для запобігання зациклення. Зупинка за другим випадком зазвичай означає необхідність змін, наприклад початкових центів кластерів.

Оскільки алгоритми кластеризації, включаючи k-середніх, використовують вимірювання на основі відстані для визначення схожості між точками даних, рекомендується стандартизувати дані, щоб вони мали середнє значення нуль і стандартне відхилення одиниці, оскільки майже завжди ознаки в будь-якому наборі даних будуть мати різні одиниці виміру. наприклад, вік проти доходу.

Враховуючи ітераційний характер k-середніх і випадкову ініціалізацію центрів на початку алгоритму, різні ініціалізації можуть призвести до різних кластерів, оскільки алгоритм k-середніх може застрягти в локальному оптимумі і не сходитися до глобального оптимуму. Тому рекомендується запустити алгоритм, використовуючи різні ініціалізації центрів і вибрати результати запуску, які дали меншу суму квадратів відстані.

Також існує нечітка модифікація даного методу, відома як fuzzy c-means, або FCM.

Параметри алгоритму: K – кількість об'єктів, x_k – k-й об'єкт, N – кількість кластерів ($2 < N < K$), j – номер кластеру, μ_{jk} – міра належності об'єкту x_k j-го до кластеру та приймає значення в інтервалі $[0,1]$, d – відстань (евклідова зазвичай), q – фіксований параметр ($\sim 1,5$), ε – параметр зупинки. Початкова матриця приналежності об'єктів до кластерів генерується випадково.

Алгоритм виглядає наступним чином:

Крок 1. Обчислити центри кластерів за формулою:

$$c_j = \frac{\sum_{j=1}^{N} (\mu_{jk})_1^q \cdot x_k}{\sum_{j=1}^{N} (\mu_{jk})_1^q}. \tag{1.8}$$

Крок 2. Обчислити відстані між об'єктами та центрами кластерів:

$$d_{jk} = \sqrt{\left\| x_k - C_j \right\|^2}. \tag{1.9}$$

Крок 3. Перерахувати матрицю приналежності:

$$\mu_{jk} = \begin{cases} \frac{1}{\left(d_{jk}^2 \cdot \sum_{j=1}^{N} \frac{1}{d}\right)^{\frac{1}{q-1}}}, \text{якщо } d_{jk} < 0 \\ 1, \text{якщо } d_{jk} > 0 \text{ та } j = k \\ 0, \text{якщо } d_{jk} > 0 \text{ та } j \neq k \end{cases} . \quad (1.10)$$

Крок 4. Повторювати кроки 1-3 доки не виконається умова виходу:

$$\left\| U^{(t)} - U^{(t-1)} \right\|^2 < \varepsilon. \quad (1.11)$$

1.3 Оцінка кількості кластерів

Конкретне оптимальне значення кількості кластерів k дізнатися дуже складно. Однак, якщо виміряти ефективність кластеризації для різних значень k, можна в певній мірі вгадати найкраще значення k. Розглянемо вплив росту відповідності кластера на ріст його радіуса або діаметра, при якому різкий ріст цих значень сигналізує про досягнення оптимальної кількість кластерів.

Якщо коректне значення k невідоме, можна знайти гарне значення через кількість операцій кластеризації, що зростає логарифмічно, доки залишається коректним. Почнемо з запуску алгоритму для $k = 1, 2, 4, 8, \ldots$. Колись будуть досягнуті величини v та $2v$ між якими є невелике зменшення діаметру, або іншої міри відповідності кластера. Ми можемо завершити тим, що значення k задається на проміжку між $v/2$ та v. Якщо ви використаєте бінарний пошук на цьому проміжку, ви можете знайти найкраще значення k за $\log 2v$ операцій кластеризації, тобто усього $2\log 2v$ за операції. Оскільки реальне значення k щонайменше $v/2$, маємо кількість операцій, логарифмічну до k. Графічне представлення такого аналізу наведена на рисунку 1.2.

Рисунок 1.2 – Середній діаметр або інша величина починає різко зростати із досягненням нижньої межі кількості адекватних кластерів

Оскільки визначення «не сильно змінюється» є неточним, не можна сказати точно, коли воно зміниться сильно. Однак бінарний пошук може бути використаний, якщо припустити, що термін «не сильно змінюється» уточнений формулою. Відомо, що існує велика зміна у проміжку між $v/2$ та v, інакше кластеризація для $2v$ кластерів не мала б сенсу. Припустимо, що у якийсь момент ми звузили проміжок k до проміжку між x та y. Нехай $z = (x + y)/2$. Запустимо кластеризацію з z вихідними кластерами. Якщо не існує значної різниці між z та y, тоді істинне значення k лежить між x та z. Тоді можна рекурсивно зменшувати проміжок, щоб знайти коректне значення k. З іншого боку, якщо існує відчутна різниця між z та y, тоді слід використати бінарний пошук у проміжку z та y.

Для ієрархічних методів широко розповсюджений метод Ліктя, він дозволяє оцінити оптимальну кількість сегментів. Метод Ліктя – це графічне відображення, рекомендація по кількості кластерів. Він відібражає зміну суми квадратів відстаней між об’єктами та центами їх кластерів в залежності від кількості кластерів. Ідеальним варіантом є явний «ліктьовий згин» для значення, що надає лише малі покращення суми квадратів відстаней.

Для наведеного нижче прикладу можна відзначити кількість кластерів 7, що відображено на рисунку 1.3.

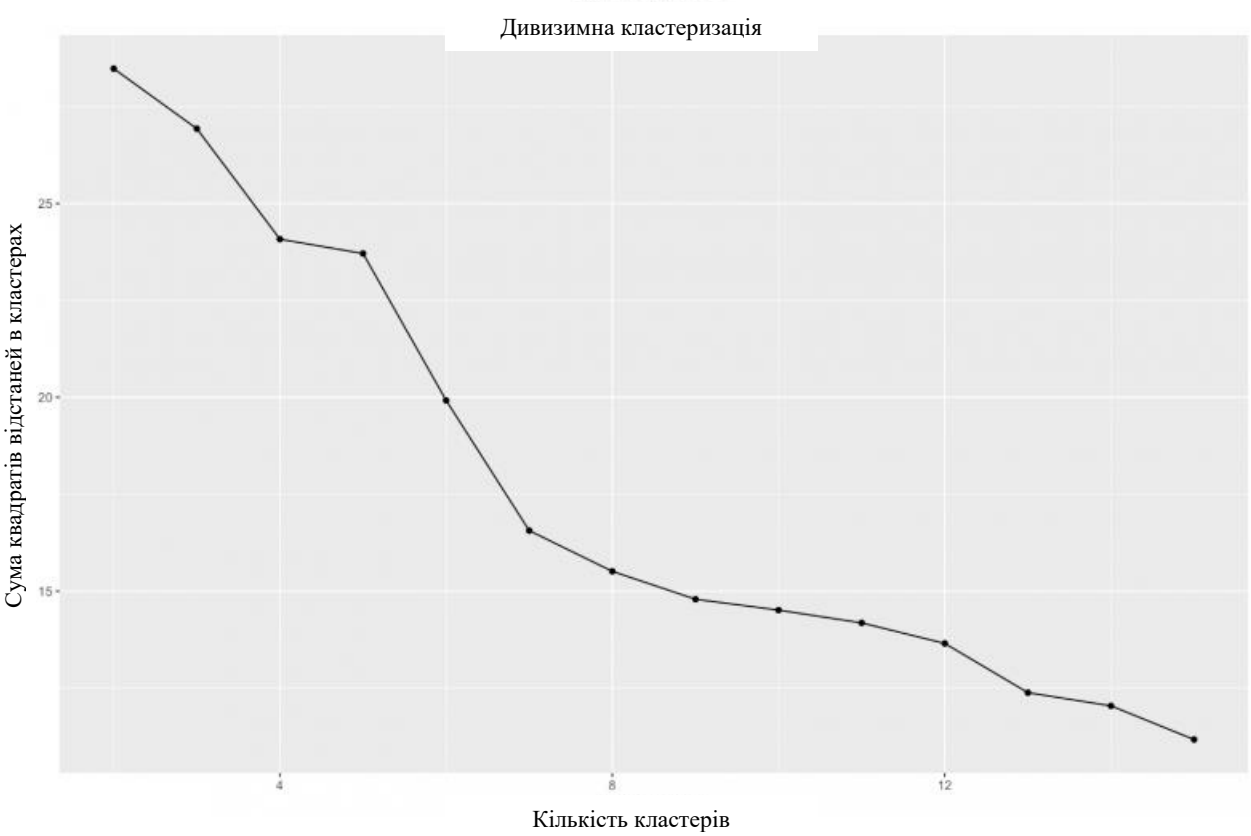


Рисунок 1.3 – Приклад ліктьового графіку для дивизийнного підходу

Слід зазначити що для агломеративного та дивизийнного підходу ліктьовий графік не повинен співпадати, що можна побачити на рисунку 1.4:

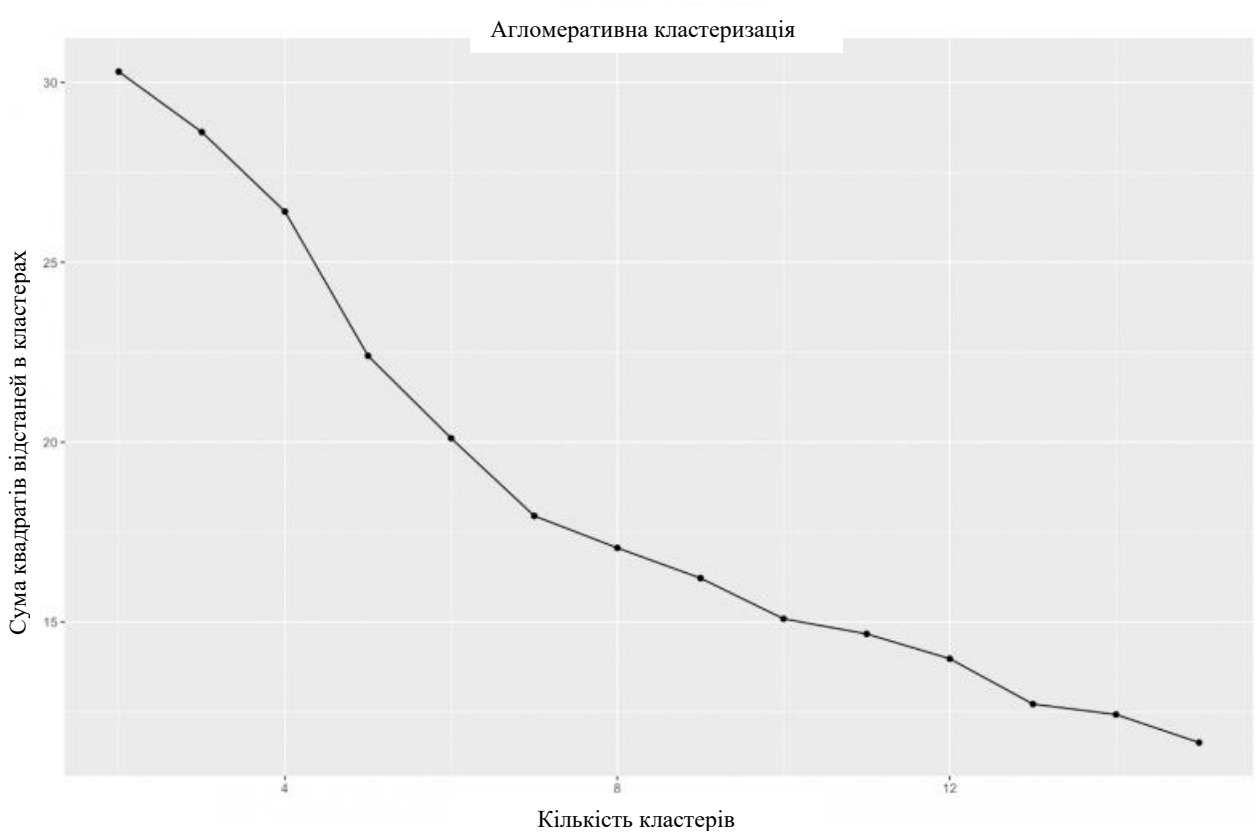


Рисунок 1.4 – Приклад ліктьового графіку для агломеративного підходу

Можна побачити, що для агломеративного підходу графік виглядає значно плавніше, а згини не так наочно виражені, що означає більшу вільність при виборі кількості кластерів та меншу залежність результату кластеризації від вибору кількості кластерів.

Інший метод оцінки кількості кластерів називається методом оцінки силуетів. В цьому випадку слід обирати таку кількість, що надає максимальний коефіцієнт силуету, оскільки нам необхідні кластери із якомога більшою різницею між собою. Приклад візуалізації такого методу оцінки представлено на рисунку 1.5.

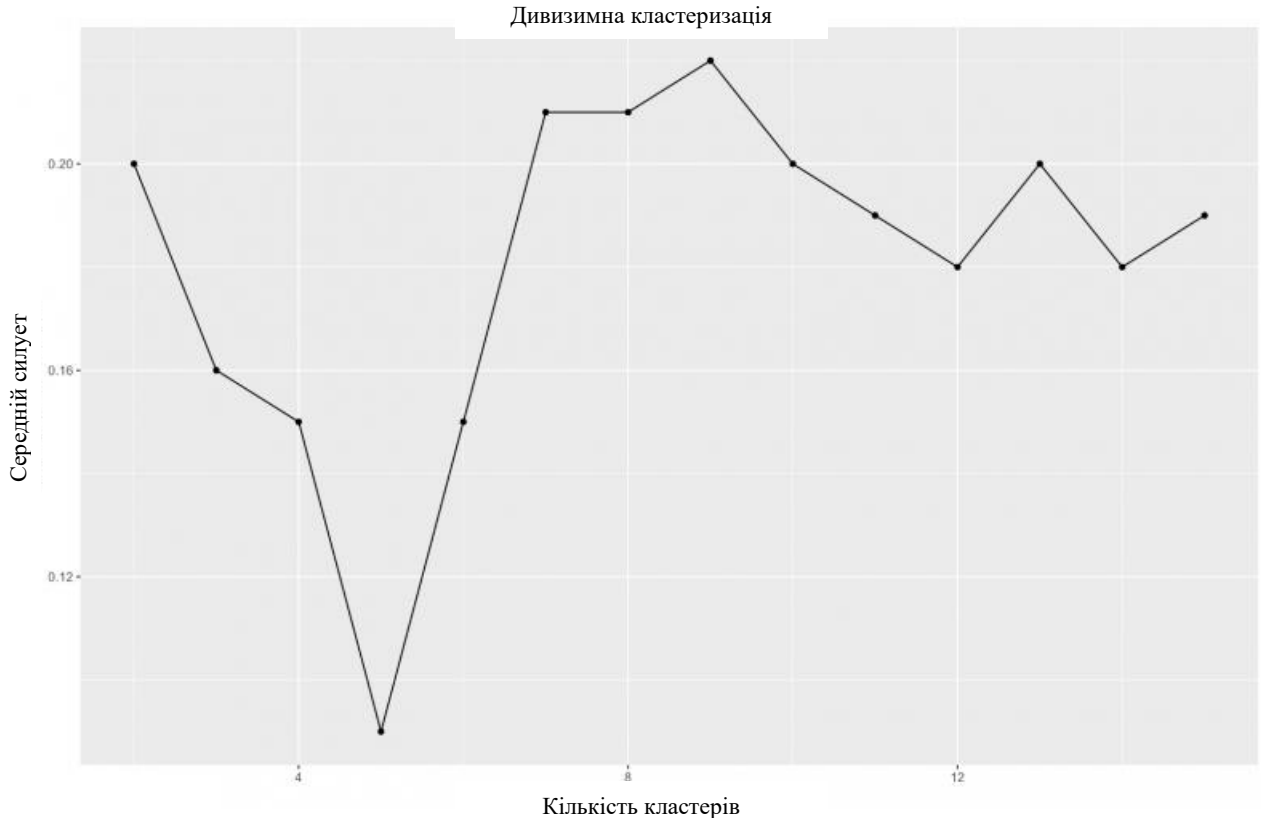


Рисунок 1.5 – Приклад графіку методу оцінки силуетів

Як можна побачити, методів оцінки тільки кількості кластерів багато і вони надають не тільки різні результати, а й орієнтуються на різні аспекти. Правильність вибору кількості кластерів завжди є важливим питанням, адже для ієрархічних методів ще можна швидко перейти від розбиття на одну кількість кластерів до іншої, а для неієрархічних методів це означає початок розрахунків із самого початку.

1.4 Постановка задачі дослідження

Таким чином, находження глобального екстремуму цільової функції є актуальним завданням для правдоподібної нечіткої кластеризації. Тому ставиться завдання реалізації ройового алгоритму зграй божевільних котів, що об'єднує в собі переваги еволюційних алгоритмів та глобального випадкового пошуку.

Об’єктом дослідження є великі набори даних.

Метою дослідження є реалізація ройового алгоритму зграй божевільних котів, що об'єднує в собі переваги еволюційних алгоритмів та глобального випадкового пошуку.

Для досягнення мети необхідно вирішити такі завдання:

– провести аналіз існуючих еволюційних методів кластеризації даних;

– розробити алгоритм кластеризації за допомогою методу оптимізації котячої зграї (CSOC);

– реалізувати алгоритм і алгоритм k-гармонійних середніх за допомогою методу оптимізації котячої зграї (KCSOC).;

– реалізувати комп’ютерну модель для перевірки дії алгоритму.

2 ДОСЛІДЖЕННЯ МЕТОДІВ КЛАСТЕРИЗАЦІЇ ДАНИХ

2.1 Підходи до методів нечіткої кластеризації

Особливість методів нечіткої кластеризації полягає в перетині кластерів і існуванні для кожного об’єкту вектору рівней належності (ймовірності) до кожного з кластерів. На даний момент має місце декілька підходів до нечіткої кластеризації, кожен з яких має свої переваги та недоліки.

Найбільш широке поширення має ймовірнісний підхід до нечіткої кластеризації. Суть даного відходу полягає в мірі (ймовірності) належності кожного об’єкту до кожного з кластерів. Кластер визначається своїм центом та за міру подібності береться відстань, а належність обернено пропорційна до відстані об’єкту до центру кластера, що зводить значення до інтервалу [0,1]. Має місце обмеження – сума мір належності об’єкту завжди є 1.

Даний підхід полягає в оптимізації цільової функції:

$$E\left(u_j(k), w_j\right) = \sum_{k=1}^{N} \sum_{j=1}^{m} u_j^{\beta}(k) D^2\left(x(k), w_j\right), \tag{2.1}$$

за обмежень:

$$\begin{cases} \sum_{j=1}^{m} u_j(k) = 1 \\ 0 \le \sum_{k=1}^{N} u_j(k) \le N \end{cases}, \tag{2.2}$$

де $u_j(k) \in [0,1]$ – рівень належності об’єкту $x(k)$ до j-го кластеру;

w_j – центр j-го кластеру;

β – фаззіфікатор, що визначає міру розмитості меж кластерів;

$D\big(x(k), w_j\big)$ – відстань між об'єктом $x(k)$ та центром w_j в певній метриці.

Результатом буде матриця нечіткого розбиття розміром $N \times m$:

$$W = \{u_j(k)\}. \tag{2.3}$$

Легко помітити, що елементи матриці W можна розглядати як ймовірності гіпотези належності $x(k)$ до j-го кластеру, звідкіля й пішла назва ймовірнісного підходу.

Відстань $D\big(x(k), w_j\big)$ зазвичай обраховується як відстань Мінковського в L^p метриці:

$$D^p\big(x(k), w_j\big) = \big\|x_i(k) - w_{ji}\big\|_{L_p}^p = \left(\sum_{i=1}^{n} \big|x_i(k) - w_{ji}\big|^p\right)^{\frac{1}{p}},\ p \geq 1, \tag{2.4}$$

де $x_i(k)$ – i-та компонента вектора $x(k)$;

w_{ji} – i-та компонента вектора w_j.

Функція Лагранжа:

$$L\left(u_j(k), w_j, \lambda(k)\right) = \sum_{k=1}^{N} \sum_{j=1}^{m} u_j^{\beta}(k) D^2\big(x(k), w_j\big) +$$

$$\sum_{k=1}^{N} \lambda(k)\big(\textstyle\sum_{j=1}^{m} u(k) - 1\big) = \displaystyle\sum_{k=1}^{N} \left(\sum_{j=1}^{m} u_j^{\beta}(k) D^2\big(x(k), w_j\big) + \right.$$

$$\left. \lambda(k)\big(\textstyle\sum_{j=1}^{m} u(k) - 1\big)\right), \tag{2.5}$$

де $\lambda(k)$ – невизначений множник Лагранжа, який забезпечує виконання умов (2.2).

За системою рівнянь Куна-Таккера розв'язком є:

$$\begin{cases} u_j^{pr}(k) = \dfrac{\left(D^2\left(x(k),w_j\right)\right)^{\frac{1}{1-\beta}}}{\sum_{i=1}^{m}\left(D^2(x(k),w_i)\right)^{\frac{1}{1-\beta}}} \\ \lambda(k) = -\left(\sum_{i=1}^{m}\left(\beta\left(D^2\left(x(k),w_j\right)\right)^{\frac{1}{1-\beta}}\right)\right)^{1-\beta} \\ w_j^{pr} = \dfrac{\sum_{k=1}^{N} u_j^{\beta}(k)x(k)}{\sum_{k=1}^{N} u_j^{\beta}(k)} \end{cases}. \quad (2.6)$$

Слід зазначити, що при $\beta = p = 2$, що відповідає евклідовому простору, маємо відомий та поширений метод нечітких c-середніх (FCM) Бездека:

$$D^E\left(x(k),w_j\right) = \left\|x(k) - w_j\right\| = \sqrt{\left(x(k) - w_j\right)^T\left(x(k) - w_j\right)}$$

$$\begin{cases} u_j^{pr}(k) = \dfrac{\|x(k)-w_j\|^{-2}}{\sum_{i=1}^{m}\|x(k)-w_i\|^{-2}} \\ \lambda(k) = -\left(\sum_{i=1}^{m}\dfrac{\|x(k)-w_i\|^{-2}}{2}\right)^{-1} \\ w_j^{pr} = \dfrac{\sum_{k=1}^{N} u_j^{2}(k)x(k)}{\sum_{k=1}^{N} u_j^{2}(k)} \end{cases}. \quad (2.7)$$

Однак недоліком даного підходу є ймовірнісні обмеження на рівні належності, що може призвести до віднесення викидів до різних кластерів при близьких рівнях належності. Наприклад навіть для найпростішої ситуації із двома кластерами ($m = 2$) легко навести приклад із рівноналежним до обох кластерів об’єктом та викидом, рівновіддаленим від центрів, але їх вектора належності виглядатимуть однаково.

У зв’язку з цим було запропоновано можливісний підхід до нечіткої кластеризації (PCM), в якому усунуто обмеження до суми мір належностей об’єкту. В цьому випадку цільова функція, яку потрібно мінімізувати, має вигляд:

$$E\left(u_j(k), w_j\right) = \sum_{k=1}^{N} \sum_{j=1}^{m} u_j^{\beta}(k) D^2\left(x(k), w_j\right) + \sum_{k=1}^{N} \mu_j \sum_{j=1}^{m} \left(1 - u_j(k)\right)^{\beta}, \quad (2.8)$$

де параметр $\mu_j > 0$ – відстань, на якій рівень належності становить 0.5, тобто $D^2\left(x(k), w_j\right) = \mu_j \rightarrow u_j(k) = 0.5$.

Мінімізація невизначеного множника Лагранжа $\lambda(k)$ (2.7) за рівнем належності $u_j(k)$, центром кластеру w_j та відстанню μ_j надає систему рівнянь:

$$\begin{cases} \frac{\partial E\left(u_j(k), w_j\right)}{\partial u_j(k)} = 0 \\ \frac{\partial E\left(u_j(k), w_j\right)}{\partial \mu(k)} = 0 \\ \nabla_{w_j} E\left(u_j(k), w_j\right) = \vec{0} \end{cases}. \quad (2.9)$$

Розв’язки перших двох рівнянь відомі, а третє розглянемо для евклідової норми (2.7):

$$\begin{cases} u_j^{pos}(k) = \left(1 + \left(\frac{D^2\left(x(k), w_j\right)}{\mu_j}\right)^{\frac{1}{\beta-1}}\right)^{-1} \\ \mu_j = \frac{\sum_{k=1}^{N} u_j^{\beta}(k) D^2\left(x(k), w_j\right)}{\sum_{k=1}^{N} u_j^{\beta}(k)} \\ w_j^{pos} = \frac{\sum_{k=1}^{N} u_j^{\beta}(k) x(k)}{\sum_{k=1}^{N} u_j^{\beta}(k)} j' \end{cases}. \quad (2.10)$$

Хоча можливісний підхід більш стійкий до описаної вище ситуації з викидами, з’являється проблема злиття кластерів в один, відома як проблема співпадіння, що призводить до невірної кількості кластерів.

Також для обох, ймовірнісного і можливісного, підходів властива неможливість роботи в реальному часі, тобто в потоковому режимі.

Необхідність вирішення описаних вище проблем призвела до виникнення ще одного підходу. Даний підхід полягає в тому, що під час розрахунків визначається рівні нечіткої залежності, а також рівні довіри, засновані на мірі належності спеціального вигляду.

2.2 Достовірна нечітка кластеризація з функціями належності спеціального типу

З математичної точки зору завдання правдоподібної нечіткої кластеризації пов'язане з мінімізацією цільової функції

$$E\left(Cr_q(k), w_q\right) = \sum_{k=1}^{N} \sum_{q=1}^{m} Cr_q^{\beta}(k) D^2\left(x(k), w_q\right), \tag{2.11}$$

за обмежень:

$$\begin{cases} 0 \le Cr_q(k) \le 1 \forall q, k, \\ sup\, C\, r_q(k) \ge 0{,}5 \forall k, \\ Cr_q(k) + sup\, C\, r_l(k) = 1, l = 1{,}2 \ldots, m;\ l \ne q \end{cases}, \tag{2.12}$$

де $\beta > 1$ – фаззіфікатор;

$D^2\left(x_k, w_q\right) = \left\|x_k - w_q\right\|^2$ – звичайна Евклідова відстань;

для всіх q і k, в яких $Cr_q(k) \ge 0{,}5$.

Треба відмітити те, що дана цільова функція (2.11) практично співпадає з цільовою функцією ймовірнісної нечіткої кластеризації. Але тут замість

рівнів належності $U_q(k)$ використовуються рівні довіри $Cr_q(k)$, що дозволяє відмовитись від ймовірнісних обмежень вигляду:

$$\sum_{q=1}^{m} U_q(k) = 1. \quad (2.13)$$

Треба відмітити, що при $\beta = 2$ (2.11) є близькою до цільової функції популярного методу нечітких с-середніх (FCM) Дж. Бездека.

Оцінка рівнів належності в правдоподібному підході визначається за функцією належності виду:

$$U_q(k) = \varphi_q\left(D\left(x_k, w_q\right)\right), \quad (2.14)$$

яка відповідає мірі подібності, що використовує оцінку відстані $D\left(x_k, w_q\right)$.

2.2.1 Кластеризація в пакетному режимі

Для кластеризації в пакетному режимі необхідне формування пакету всього масиву даних.

В [10,11] в якості такої функції запропоновано використовувати вираз:

$$U_q(k) = \frac{1}{1+D^2\left(x_k, w_q\right)}, \quad (2.15)$$

який є дзвонуватою функцією щільності розподілу Коші з одиничним параметром ширини. Тут можна показати, що використання оцінок ймовірнісної нечіткої належності:

$$U_q(k) = \frac{\left(D^2(x_k, w_q)\right)^{\frac{1}{1-\beta}}}{\sum_{l=1}^{m}\left(D^2(x_k, w_l)\right)^{\frac{1}{1-\beta}}} = \frac{\left(D^2(x_k, w_q)\right)^{\frac{1}{1-\beta}}}{\left(D^2(x_k, w_q)\right)^{\frac{1}{1-\beta}} + \sum_{\substack{l=1 \\ l\neq q}}^{m}\left(D^2(x_k, w_l)\right)^{\frac{1}{1-\beta}}} =$$

$$= \frac{1}{1+\frac{\sum_{\substack{l=1 \\ l\neq q}}^{m}\left(D^2(x_k,w_l)\right)^{\frac{1}{1-\beta}}}{D^2(x_k,w_q)^{\frac{1}{1-\beta}}}}. \qquad (2.16)$$

для $\beta = 2$ дозволяє отримати розподіл Коші:

$$U_q(k) = \frac{1}{1+\frac{\|x_k - w_q\|^2}{\sigma_q^2}}, \qquad (2.17)$$

параметр ширини якого:

$$\sigma_q^2 = \frac{1}{\sum_{\substack{l=1 \\ l\neq q}}^{m}\|x_k - w_l\|^{-2}}, \qquad (2.18)$$

що є узагальненням (2.16) та співпадає з ним при $\sigma_q^2 = 1$. Таким чином, рівні належности в правдоподібному підході співпадають з ймовірнісними рівнями належності, які використовуються в FCM.

Таким чином процедура правдоподібної кластеризації може бути модифікована та записанна у вигляді:

$$\begin{cases} U_q(k) = \frac{1}{1+\frac{\|x_k - w_q\|^2}{\sigma_q^2}} \\ U_q^*(k) = \frac{U_q(k)}{sup\, U_l(k)} \\ Cr_q(k) = \frac{1}{2}\left(U_q(k) + 1 - \underset{l \neq q}{sup} U_l^*(k) \right), \\ w_q = \frac{\sum_{k=1}^{N} \left(Cr_q(k)\right)^{\beta} x_k}{\sum_{k=1}^{N} \left(Cr_q(k)\right)^{\beta}} \end{cases} \quad (2.19)$$

де σ_q^2 визначається виразом (2.81).

2.2.2 Кластеризація в потоковому режимі

Якщо ж дані надходять на обробку послідовно в онлайн режимі можна скористатися адапивним правдоподібним методом кластеризації у вигляді:

$$\begin{cases} \sigma_q^2(k+1) = \dfrac{1}{\sum\limits_{\substack{l=1 \\ l \neq q}}^{m} \left\| x_{k+1} - w_l(k) \right\|^{-2}} \\ U_q(k+1) = \dfrac{1}{1 + \dfrac{\left\| x_{k+1} - w_q(k) \right\|^2}{\sigma_q^2(k+1)}} \\ U_q^*(k+1) = \dfrac{U_q(k+1)}{\sup U_l(k+1)} \\ Cr_q(k+1) = \dfrac{1}{2}\left(U_q^*(k+1) + 1 - \sup\limits_{l \neq q} U_l^*(k+1) \right) \\ w_q(k+1) = w_q(k) + \eta(k+1) Cr_q^{\beta}(k+1)\left(x_{k+1} - w_q(k) \right) \end{cases}, \quad (2.20)$$

де $\eta(k+1)$ – параметр кроку навчання.

Важливо зазначити, що модифікована процедура правдоподібної нечіткої кластеризації (2.19) є результатом пакетної оптимізації цільової функції (2.14), а адапивний правдоподібний метод кластеризації (2.20) – градієнтним алгоритмом пошуку екстремуму цієї ж функції, тобто гарантується відшукання тільки локального екстремуму.

Також було показано, що задача умовної оптимізації цільової функції ймовірнісної нечіткої кластеризації може бути представлена у вигляді задачі безумовної оптимізації функції:

$$E(w_q) = \sum_{k=1}^{N} \left(\sum_{q=1}^{m} \left\| x_k - w_q \right\|^{2(1-\beta)} \right)^{1-\beta}, \quad (2.21)$$

або при $\beta = 2$:

$$E(w_q) = \sum_{k=1}^{N} \left(\sum_{q=1}^{m} \left\| x_k - w_q \right\|^{-2} \right)^{-1}, \quad (2.22)$$

що має безліч локальних екстремумів. Отже мова йде про відшукання лише одного з них, що буде найближчим до початкової точки пошуку.

Таким чином виявляється ситуація, де гарно показують себе ройові алгоритми оптимізації, які повинні бути відповідним чином адаптовані до розглянутої задачі і відповідати вимогам високої швидкодії.

2.3 Еволюційнй методи

У випадку багатоекстремальної функції є ризик потрапити у локальний мінімум що ускладнює або робить інколи неможливим використання градієнтних методів.

Тому, доцільно використовувати методи еволюційного пошуку, які є стохастичними методами, заснованими на аналогії з природними еволюційними процесами, не висувають додаткових вимог до виду цільової

функції, на кожній ітерації працюють з множиною рішень, що дозволяє в багатьох випадках більш детально в порівнянні з градієнтними методами багатовимірної нелінійної безумовної оптимізації аналізувати простір пошуку.

До еволюційних методів відносять: генетичні методи, еволюційні стратегії, генетичне програмування та еволюційне програмування.

При вирішенні багатьох задач у техніці актуальною є проблема знаходження глобального оптимуму цільової функції в багатовимірному просторі керованих змінних. Традиційні методи багатовимірної оптимізації є методами локального пошуку, сильно залежать від вибору початкової точки пошуку та накладають додаткові обмеження на властивості цільової функції оптимізації.

Тому для вирішення оптимізаційних задач доцільним є використання генетичних методів, як є методами еволюційного пошуку й поєднують комп'ютерні методи моделювання генетичних процесів у природних і штучних системах.

Для роботи генетичних методів, як інформація про функцію, що оптимізується, використовуються її значення в даних точках простору пошуку і не потрібно обчислень похідних або інших характеристик. Тому дані методи можуть бути застосовані до широкого класу функцій, зокрема до тих, що не мають аналітичного опису. Таким чином, методи генетичного пошуку є достатньо гнучкими і можуть бути застосовані до широкого кола задач, в тому числі до задач, для розв'язування яких не існує загальновідомих методів.

Еволюційні стратегії орієнтовані на оптимізацію неперервних функцій з використанням операторів схрещування.

В той час, як генетичний метод моделює еволюцію на рівні геномів (хромосом), еволюційні стратегії спрямовані на еволюцію фенотипів. Оскільки еволюційні стратегії створювались спеціально для чисельної оптимізації, в них фенотипи подаються дійсними векторами.

Оригінальна (у первинному варіанті) еволюційна стратегія була двоелементною схемою, що складалася з батька і нащадка. У базовому методі

батько, мутуючи, створює нащадка, і одна з двох особин з кращим значенням фітнес-функції переходить в популяцію наступного покоління. Цей метод був узагальнений пізніше на так звані $(\mu + \lambda)$ і $(\mu,\ \lambda)$ еволюційні стратегії. Параметри μ і λ означають число батьків і нащадків, відповідно.

У еволюційній $(\mu + \lambda)$ стратегії μ батьків породжують λ нащадків. При цьому батьки і нащадки беруть участь у загальному селективному відборі, внаслідок чого μ кращих особин переходять в наступну популяцію.

Еволюційна $(\mu,\ \lambda)$ стратегія передбачає, що μ батьків, породивши λ нащадків, гинуть. У наступну популяцію переходять μ особин з множини нащадків, причому в даному способі повинна виконуватися умова λ > μ. Такий підхід може бути застосовний до задач з оптимумом, що змінюється, і з зашумленими даними.

Коефіцієнт $\eta = \lambda/\mu$ зазвичай більше або дорівнює семи. Чим він більше, тим більше шансів, що кожен батько згенерує, щонайменше, одного нащадка, кращого, ніж він сам. Проте відмінності між $(\mu + \lambda)$ та $(\mu,\ \lambda)$ еволюційними стратегіями стають меншими, якщо η є достатньо великим.

Канонічна еволюційна стратегія може бути подана такою послідовністю кроків.

Крок 1. Встановити: $t = 0$.

Крок 2. Ініціалізувати популяцію P_t μ випадковими індивідами.

Крок 3. Випадковим чином відібрати особини з P_t для генерації λ нащадків.

Крок 4. Виконати оператор мутації над нащадками.

Крок 5. Обчислити фітнес-функції нащадків.

Крок 6. Вибрати кращі μ особин, базуючись на значеннях фітнесфункції, і створити популяцію P_{t+1}.

Крок 7. Встановити: $t = t + 1$.

Крок 8. Виконати перевірку критеріїв зупинення пошуку. В разі їх невиконання, перейти до кроку 3.

Крок 9. Зупинитися.

Порівняльний аналіз еволюційних стратегій і генетичних методів показує, що головна їх відмінність в тому, що тоді як в генетичному методі вибираються особини для рекомбінації пропорційно їх фітнес-функції і замінюються особини з попередньої популяції, в еволюційних стратегіях діють навпаки. В цьому випадку особини для репродукції вибираються з рівними ймовірностями, а формування наступної популяції базується на значеннях фітнес-функцій.

Генетичне програмування застосовує еволюційний пошук для оптимізації комп'ютерних програм.

Як правило, методи генетичного програмування використовуються для пошуку функціональної залежності, що найповніше відображає досліджуваний об'єкт, процес або явище. При цьому як фітнес-функція може бути використаний критерій, що характеризує ефективність застосування одержаної функціональної залежності або моделі.

Еволюційне програмування орієнтоване на оптимізацію неперервних функцій без використання оператора схрещування. Інший напрям еволюційного програмування пов'язаний з пошуком залежності цільових параметрів від інших змінних у формі функцій якогось певного вигляду. Наприклад, в одному з найбільш вдалих методів цього типу, методі групового врахування аргументів, залежність шукають у формі поліномів. Від генетичних методів еволюційне програмування відрізняється не лише відсутністю схрещування, але і поданням особин популяції. Оскільки тут немає необхідності у використанні хромосом, то особина часто є реальним рішенням без додаткового кодування. Наприклад, в еволюційному програмуванні в якості особин дуже часто використовуються скінченні автомати. Відповідно оператори відбору і мутації застосовуються безпосередньо до них. При оптимізації функцій в якості особин популяції використовуються безпосередньо вектори дійсних чисел.

Узагальнена послідовність виконання еволюційного програмування включає такі кроки.

Крок 1. Виконати ініціалізацію параметрів методу. (При використанні в якості особин скінчених автоматів необхідно сформувати вхідний словник, множину вхідних і вихідних станів, набір можливих станів, умови переходів зі стану в стан, фітнес-функцію для характеристики моделей, що генеруються.)

Крок 2. Випадковим чином згенерувати початкову популяцію особин.

Крок 3. Виконати тестування особин початкової популяції шляхом розв’язання поставленої задачі (на вхід моделі подається задана вибірка даних) і оцінювання отриманих результатів на основі обраної фітнес-функції.

Крок 4. Відібрати рішення з кращими значеннями фітнес-функції.

Крок 5. На основі випадкового застосування оператора мутації до особин-батьків згенерувати нащадків, які перейдуть в нову популяцію.

Крок 6. Виконати тестування нащадків шляхом розв’язання поставленої задачі і оцінювання отриманих результатів.

Крок 7. Відібрати найбільш перспективних нащадків.

Крок 8. Перевірити умови закінчення процесу еволюції, якими можуть бути: досягнення оптимального значення фітнес-функції або досягнення граничних значень, що обмежують тривалість процесу. Якщо умови закінчення пошуку задовільнено, тоді виконати перехід до кроку 9, в іншому випадку виконати повернення до кроку 5, де особини останньої згенерованої популяції, виступають в якості батьків.

Крок 9. Зупинитися.

2.4 Алгоритм глобальної оптимізації божевільної котячої зграї в задачі нечіткої кластеризації

Для вирішення питання пошуку глобального екстремуму цільової функції нечіткої кластеризації пропонується використовувати модифікований

метод оптимізації божевільної котячої зграї, синтезованого на основі оптимізаційного підходу котячої зграї і методів глобального випадкового пошуку.

Оптимізація на основі еволюційного алгоритму котячої зграї полягає в тому, що формується група-зграя "котів", кожен з яких рухається у напрямку або локального, або глобального екстремуму прийнятої цільової функції $E(w_q)$. При цьому ця зграя складається з Q осіб $cat_p, p = 1,2, \dots, Q$, кожна з яких може перебувати в одному з двох можливих станів: режим пошуку (SM) локальних екстремумів і режим погоні (або режим трасування) (TM), що ставить собі за мету відшукання глобального екстремуму. SM, як правило, реалізується на основі градієнтного пошуку з малим параметром навчання і таким чином сканує локальний окіл кожного з котів, що знаходиться в цьому режимі. TM характеризується випадковими стрибками з великою амплітудою і ставить собі за мету "витягти" кота cat_p з локального екстремуму в разі його потрапляння туди.

Використання як локального сканування, так й швидкої зміні поточного стану дозволяє збільшити ймовірність знаходження глобального екстремуму, в порівнянні з традиційними методами багатоекстремальної оптимізації.

В загальному випадку стандартний алгоритм котячої зграї може бути представлений у вигляді послідовності ітерацій:

Крок 1. Випадковим чином створюється зграя з Q котів cat_p, кожен з яких є за суттю n -вимірним вектором $w_p(0)$ в області визначення функції, що підлягає оптимізації, та оцінці значення функції в цій точці $E\left(w_p(0)\right)$.

Крок 2. Вводиться параметр стану, що приймає значення 0 або 1, за допомогою якого вихідна зграя розбивається випадковим чином на дві групи: якщо параметр стану є 1, то кіт знаходиться в режимі пошуку, якщо ж параметр стану є 0, то відповідий кіт знаходиться в режимі погоні.

Крок 3. Коти з параметром стану 1 починають пошук локального екстремуму, а коти з параметром стану 0 запускаються в режим погоні.

Крок 4. Оцінюються значення цільової функції для всіх котів і зберігаються всі $w_p(1)$ з найменшими значеннями цієї функції, коти з найбільшим значенням $E\left(w_p(1)\right)$ можуть бути видалені із зграї

Крок 5. Повернення до першого кроку з новими значеннями, тобто починається новий етап з оновленою популяцією.

У загальному випадку обидва режими SM і TM реалізуються паралельно, при цьому SM фактично базується на основі покоординатного спуску, тобто в кожен конкретний момент може змінюватися тільки одна координата n -вимірного простору пошуку, що суттєво знижує швидкодію процедури. У режимі погоні швидкості руху по кожній координаті також оцінюються незалежно одне від одної, що знову-таки знижує швидкодію.

Для вирішення цих проблем було запропоновано рандомізований алгоритм оптимізації на основі котячих зграй, що забезпечує підвищену швидкодію в порівнянні з відомою процедурою-прототипом. При цьому рух кота в режимі пошуку може бути описано за допомогою рекурентної процедури:

$$w_p(\tau+1) = w_p(\tau) - \eta_{SM}\hat{\nabla}E\left(w_p(\tau)\right), \qquad (2.23)$$

де $\hat{\nabla}E\left(w_p(\tau)\right)$ – оцінка градієнта функції, що оптимізується в точці $w_p(\tau)$, одержувана або на основі пошуку з центральною пробою, або на основі випадкових проб (статистичний градієнт);

η_{SM} – малий крок пошуку в просторі R^n.

Рух кота в режимі погоні описується алгоритмом, що є "гібридом" популярного методу оптимізації "важкої кульки" і випадкового пошуку:

$$w_p(\tau+1) = w_p(\tau) - \alpha\left(w_p(\tau) - w_p(\tau-1)\right) - -\eta_{TM}\hat{\nabla}E\left(w_p(\tau)\right) + \Xi(\tau), \qquad (2.24)$$

де $0 < \alpha < 1$ – параметр інерції режиму погоні;

$\Xi(\tau)$ – випадкове збурення, що вводить додаткове сканування простору пошуку.

Для поліпшення процесу пошуку глобального екстремуму в режимі погоні в алгоритм руху кожної кішки додатково було введено "фактор божевільності", який описується набором випадкових параметрів і дозволяє здійснювати раптові стрибки, що змінюють траєкторію руху, шляхом варіювання характеристик сигналу збурення $\Xi(\tau)$.

Для керування сигналом $\Xi(\tau)$ доцільно скористатися ідеєю "блукаючого" глобального пошуку, який довів свою ефективність при вирішенні багатоекстремальних задач.

При цьому характеристики випадкового збурення Л. Растригін запропонував змінювати відповідно до виразу:

$$\Xi(\tau) = \gamma\Xi(\tau - 1) - \delta\left(E\left(w_p(\tau)\right) - E\left(w_p(\tau - 1)\right)\right) + \sigma^2 H(k), \quad (2.25)$$

де γ – параметр корекції характеристик збурення;

$0 \le \delta \le 1$ – параметр швидкості самонавчання типу параметра інерції α у (2.14);

$\sigma^2\,\sigma^2$ – дисперсія білого шуму $H(\tau)$.

Таким чином весь процес оптимізації за допомогою підходу "божевільних котів" може бути описаний за допомогою рекурентних співвідношень (2.24), (2.25), при цьому при $\alpha = \gamma = \delta = 0$ режим погоні автоматично переходить в режим пошуку (рух по антіградієнту).

3 КОМП'ЮТЕРНА МОДЕЛЬ

3.1 Обґрунтування вибору середовища програмної реалізації

У рамках кваліфікаційної роботи був розроблений алгоритм дослідження методу правдоподібної кластеризації даних на основі еволюційного методу божевільних котів. Для реалізації було обране середовище Python 3.0.

Python – об'єктно-орієнтована мова загального призначення, який розроблений з метою підвищення продуктивності програміста. Із плюсів можна виділити:

– низький поріг входження. Синтаксис Python більш зрозумілий для новачка;

– логічний, лаконічний і зрозумілий. У порівнянні з багатьма іншими мовами Python має легкочитаємий синтаксис. Кроссплатформовий: підходить для різних платформ: і Linux, і Windows;

– широке застосування. Використовується для розробки веб-додатків, ігор, зручний для автоматизації, математичних обчислень, машинного навчання, в області інтернету речей. Існує реалізація під назвою Micro Python, оптимізована для запуску на мікроконтролерах (можна писати інструкції, логіку взаємодії пристроїв, організовувати зв'язок, реалізовувати розумний будинок);

– у світі для Python багато якісних бібліотек, які й будуть використовуватись для реалізації алгоритмів, так що не потрібно винаходити велосипед, якщо треба терміново вирішити якесь завдання;

– Python відрізняється суворою вимогою до написання коду (вимагає відступи), що є перевагою, за моїми спостереженнями. Спочатку мова сприяє писати код організовано і красиво.

– мова Python виправдовує звання мови, написаної програмістами – вона позбавлена великої кількості непотрібних елементів таких як позначення початку чи кінця тіла, наприклад, циклу, що зазвичай робиться за допомогою фігурних дужок чи слів, а потім гарним тоном вважається виокремлення даного блоку тексту відступом, тому в Python залишили сам відступ, що допомагає підвищити читабельність коду та позбавились зайвих символів та рядків для них.

Звичайно, у сторони дві медалі, і якщо говорити про мінуси, то Python – мова із динамічною типізацією. З одного боку код простіше і швидше писати, але продуктивність поступається таким компільованим мовам, як C ++ і Golang. Але для більшості завдань: для веб-розробки, для скриптів, прототипирования, машинного навчання та роботи з великими даними, – один з кращих мов. Хоча в деяких випадках програмування наприклад на C++ може надавати збільшення показників швидкодії в рази.

3.2 Навчальні вибірки

Для перевірки ефективності, а також оцінки працездатності та якості кластеризації великих обсягів даних правдоподібної нечіткої кластеризації даних на основі еволюційного методу божевільних котів було проведено дослідження на декількох різних вибірках даних, наведених в Таблиці 3.1.

Таблиця 3.1 – Характеристики навчальних вибірок

Навчальна вибірка	Кількість кластерів	Кількість атрибутів	Кількість спостережень
Іриси Фішера	3	4	150
Cancer	2	9	699
Вина	3	13	178
Glass	6	8	214

Вибірка даних: Iris , або Іриси Фішера.

Вона містить 150 записів (по 50 кожного з 3 класів).

Міститься 4 чисельних атрибута:

– sepal length in cm – довжина чашолистику в см;
– sepal width in cm – ширина чашолистику в см;
– petal length in cm – довжина пелюстки в см;
– petal width in cm – ширина пелюстки в см.

Дані поділені на 3 класи, що відповідають сортам ірисів:

– Iris Setosa
– Iris Versicolour
– Iris Virginica

Пропущених даних немає.

Вибірка даних: Cancer.

Містить 699 записів (вміст класів 458 та 241).

Міститься 9 чисельних атрибутів, зведених до значень у діапазоні [1,10]:

– Clump Thickness – товщина згустку.
– Uniformity of Cell Size – рівномірність розміру клітини.
– Uniformity of Cell Shape – рівномірність форми клітини.
– Marginal Adhesion – гранична адгезія.
– Single Epithelial Cell Size – розмір окремої епітеліальної клітини
– Bare Nuclei – голі ядра
– Bland Chromatin – слабкий хроматин
– Normal Nucleoli – нормальні ядерця
– Mitoses – мітози

Дані поділені на 2 класи, що відповідають характеру пухлин:

– Benign – доброякісні, містить 458 об’єктів.
– Malignant – злоякісні, містить 241 об’єкт.

В даних є 16 пропущених значень атрибутів.

Вибірка даних: Wines.

Містить 178 записів (вміст класів 59, 71, 48).

Міститься 13 чисельних атрибутів:

– Alcohol.
– Malic acid.
– Ash.
– Alcalinity of ash.
– Magnesium.
– Total phenols.
– Flavanoids.
– Nonflavanoid phenols.
– Proanthocyanins
– Color intensity
– Hue
– OD280/OD315 of diluted wines
– Proline

Дані поділені на 3 класи:

– містить 59 об’єктів.
– містить 71 об’єкт.
– містить 48 об’єктів.

Пропущених даних немає.

Вибірка даних: Glass.

Містить 214 записів (вміст класів 59, 71, 48).

Міститься 13 чисельних атрибутів:

– Alcohol.
– Malic acid.
– Ash.
– Alcalinity of ash.
– Magnesium.

– Total phenols.
– Flavanoids.
– Nonflavanoid phenols.
– Proanthocyanins
– Color intensity
– Hue
– OD280/OD315 of diluted wines
– Proline

Дані поділені на класи:

– 163 об’єкти. Віконне скло (вікна будівлі та транспортні засоби)
– 87 м’яко обромлених
– 70 вікон будівельних
– 17 вікон для автівок
– 76 нем’яких вікон (всі для будівель)
– 51 невіконного скла
– 13 для контейнерів
– 9 столових приладів
– 29 фар

Пропущених даних немає.

3.3 Програмна реалізація

3.3.1 Попередня обробка даних

Приклад попередньої обробки даних розглянемо на вибірці Iris.

За описом використані дані не містять пропущених або некоректних даних. Також всі чисельні.

Імпорт даних є важливою складовою процесу попередньої підготовки даних до кластеризації. Отримувати дані ми будемо із інтернет-репозиторію (рис. 3.1).

```
def import_data():
    data_path = 'https://archive.ics.uci.edu/ml/machine-learning-
databases/iris/iris.data'
    data_file = pd.read_csv(data_path, header=None)
    data_file.columns = ['sepal_length', 'sepal_width',
                         'petal_length ', 'petal_width', 'class']
    return data_file
```

Рисунок 3.1 – Функція імпорту даних

В репозиторії дані зберігаються у файлах з форматом CSV та мають заголовок. В разі необхідності завжди можна перевірити інформацію щодо набору даних у супровідному інформаційному файлі, що зазвичай наявні в серйозних репозиторіях.

Далі проведемо нормалізацію даних за допомогою кодування на гіперкуб (рис 3.2).

```
def coding_to_hypercube(numbers):
    old_numbers = np.array(numbers)
    new_numbers = []
    min_number = np.min(old_numbers)
    max_number = np.max(old_numbers)
    for number in old_numbers:
        new_numbers.append(2 * (number - min_number) / (max_number -
min_number) - 1)
    return new_numbers
```

Рисунок 3.2 – Функція кодування на гіперкуб

Для використання чисельних методів, необхідно позбутися класу в даних, який є рядком та не є атрибутом для кластеризації. Тому створюємо список класів та додаткову матрицю для зберігання факту належності елементу до класу (рис 3.3).

```
class_amount = 3
classes = ['Iris-setosa', 'Iris-versicolor', 'Iris-virginica']
……
Y = [[-1] * len(data_T_num) for i in range(class_amount)]
for train_index in range(len(data_T_num)):
    for class_index in range(len(classes)):
        if data_T_num[train_index][-1] == classes[class_index]:
            Y[class_index][train_index] = 1

check = np.array(data_T_num).T
check = check[:len(check)-1:]
check = check.T.tolist()
check = [[float(check[i][j]) for j in range(len(check[i]))] for i in
range(len(check))]
```

Рисунок 3.3 – Вилучення даних про клас

Таким чином дані зведені до вигляду, показаному на рисунку 3.4.

```
44 :    [-0.5555555555555558, 0.49999999999999956, -0.6949152542372882, -0.75]     [ 1 -1 -1]
45 :    [-0.7222222222222223, -0.16666666666666674, -0.864406779661017, -0.8333333333333334]    [ 1 -1 -1]
46 :    [-0.5555555555555558, 0.49999999999999956, -0.7966101694915254, -0.9166666666666666]    [ 1 -1 -1]
47 :    [-0.8333333333333335, 0.0, -0.864406779661017, -0.9166666666666666]     [ 1 -1 -1]
48 :    [-0.44444444444444453, 0.4166666666666665, -0.8305084745762712, -0.9166666666666666]    [ 1 -1 -1]
49 :    [-0.611111111111111, 0.08333333333333304, -0.864406779661017, -0.9166666666666666]     [ 1 -1 -1]
50 :    [0.4999999999999998, 0.0, 0.2542372881355932, 0.08333333333333326]      [-1  1 -1]
51 :    [0.16666666666666674, 0.0, 0.18644067796610164, 0.16666666666666674]    [-1  1 -1]
52 :    [0.4444444444444444, -0.08333333333333337, 0.3220338983050848, 0.16666666666666674]     [-1  1 -1]
53 :    [-0.33333333333333337, -0.7500000000000002, 0.016949152542372836, 0.0]      [-1  1 -1]
54 :    [0.2222222222222221, -0.3333333333333336, 0.22033898305084731, 0.16666666666666674]     [-1  1 -1]
55 :    [-0.2222222222222221, -0.3333333333333336, 0.18644067796610164, 0.0]    [-1  1 -1]
56 :    [0.11111111111111094, 0.08333333333333304, 0.2542372881355932, 0.25]    [-1  1 -1]
57 :    [-0.6666666666666664, -0.6666666666666667, -0.22033898305084754, -0.25]     [-1  1 -1]
58 :    [0.27777777777777746, -0.2500000000000002, 0.22033898305084731, 0.0]    [-1  1 -1]
```

Рисунок 3.4 – Скріншот коду виводу check та Y

3.3.2 Реалізація правдоподібної кластеризації на основі еволюційного методу божевільних котів

Найважливішою частиною є рандомізований алгоритм оптимізації на основі котячих зграй, що забезпечує підвищену швидкодію в порівнянні з відомою процедурою - прототипом. При цьому рух кота в режимі пошуку може бути описано за допомогою рекурентної процедури.

Рух кота в режимі погоні описується алгоритмом, що є "гібридом" популярного методу оптимізації "важкої кульки" і випадкового пошуку (рис 3.5).

```
for it in range(3):

    for j in range(k):
        denoSum = sum(np.power(weight[:, j], 2))

        sumMM = 0
        for i in range(n):
            mm = np.multiply(np.power(weight[i, j], p), X[i, :])
            sumMM += mm
        cc = sumMM / denoSum
        C[j] = np.reshape(cc, d+1)

    for i in range(n):
        denoSumNext = 0
        for j in range(k):
            denoSumNext += np.power(1 / distance.euclidean(C[j, 0:d], X[i,
0:d]), 1 / (p-1))
        for j in range(k):
            w = np.power((1 / distance.euclidean(C[j, 0:d], X[i, 0:d])), 1 /
(p-1)) / denoSumNext
            weight[i, j] = w
```

Рисунок 3.5 – Скріншот коду реалізації

3.4 Перевірка результату кластеризації

Для перевірки поглянемо на те, скільки до якого класу:

– правильно позначено, як належний (TP);

– неправильно позначено, як належний (FP);

– неправильно позначено, як неналежний (FN);

– правильно позначено, як неналежний (TN).

А також виведемо загальну кількість перевірених значень за кожною класифікацією щоб переконатись, що їх 75 (об’єм test), що показано на рисунках 3.6 та 3.7).

```
test_check_title = ['TP', 'FP', 'FN', 'TN.']
test_check = [[0] * len(test_check_title) for i in range(class_amount)]

for class_index in range(class_amount):
    for k in range(len(test)):
        if Y_predicted[class_index][k] == 1 and classes[class_index] ==
test[k][-1]:
            test_check[class_index][0] += 1
        elif Y_predicted[class_index][k] == 1 and classes[class_index] !=
test[k][-1]:
            test_check[class_index][1] += 1
        elif Y_predicted[class_index][k] == -1 and classes[class_index] ==
test[k][-1]:
            test_check[class_index][2] += 1
        elif Y_predicted[class_index][k] == -1 and classes[class_index] !=
test[k][-1]:
            test_check[class_index][3] += 1

print('\ncheck on test')
print('\t', test_check_title)
for i in range(class_amount):
    print(i, ':\t', test_check[i], '\t', np.sum(test_check[i]))
```

Рисунок 3.6 – Скріншот коду перевірки роботи методу

```
check on test
     ['TP', 'FP', 'FN', 'TN.']
0 :  [25, 3, 0, 47]      75
1 :  [5, 18, 20, 32]     75
2 :  [24, 13, 1, 37]     75
```

Рисунок 3.7 – Скріншот роботи перевірки роботи методу

3.5 Результати досліджень

Для проведення порівняння із іншими методами кластеризації та доведення переваги над аналогами було обрано метод рою частинок (Particle Swarm Optimization, PSO) та оптимізація котячої зграї (CSO, Cat Swarm Optimization). Проведено дослідження на вказаних вибірках даних, результати яких наведено в Таблиці 3.2.

Таблиця 3.2 – Порівняльні результати показників ефективності таких алгоритмів, як PSO, CSO

Нав-чальна вибірка	MSE	PSO	CSO	Правдоподібна нечітка кластеризація даних на основі еволюційного методу божевільних котів
Іриси Фішера	Найкраще	4×10^{-7}	9×10^{-10}	8.4×10^{-11}
	Середнє	7×10^{-6}	7.3×10^{-9}	5.2×10^{-10}
	Найгірше	1.2×10^{-5}	9.3×10^{-9}	9.6×10^{-10}
Cancer	Найкраще	1.3×10^{-7}	8×10^{-10}	8.5×10^{-11}
	Середнє	7×10^{-6}	4.4×10^{-9}	7.6×10^{-11}
	Найгірше	2.02×10^{-5}	6.8×10^{-9}	7.8×10^{-10}
Вина	Найкраще	1.4×10^{-6}	7×10^{-10}	9.5×10^{-11}
	Середнє	5×10^{-5}	4×10^{-9}	6.6×10^{-11}
	Найгірше	2×10^{-5}	6×10^{-9}	6.8×10^{-10}
Glass	Найкраще	2.5×10^{-7}	7.9×10^{-10}	8.7×10^{-11}
	Середнє	6.9×10^{-6}	5×10^{-9}	7.7×10^{-11}
	Найгірше	3×10^{-5}	5.9×10^{-9}	7.7×10^{-10}

Як можна побачити, різниця між ройовим методом та котячими значна, але і не в його користь. Котячі ж методи показують близькі результати між собою, але перевага на боці правдоподібної нечіткої кластеризації даних на основі еволюційного методу божевільних котів.

В такому розподілі результатів є сенс, адже оптимізація котячої зграї була запропонована та введена із значним прогресом обчислень та результатів, а еволюційний метод божевільних котів заснований саме на ньому, як продовження покращення результату та котячої тематики.

Аналізуючи та оцінюючи отримані результати експериментальних досліджень, можна зробити висновки, що запропонований метод правдоподібної нечіткої кластеризації даних на основі еволюційного методу божевільних котів забезпечує достатньо якісні результати кластеризації, що підтверджується експериментально.

ВИСНОВКИ

У рамках кваліфікаційної роботи було проведено огляд відомих підходів до нечіткої кластеризації даних, а також еволюційних методів. Аналіз розглянутих методів показує перевагу правдоподібного підходу над ймовірнісним і можливісним за рахунок врахування рівня довіри, що дозволяє визначати легше викиди так запобігати злиттю кластерів.

Для експериментальних досліджень було обрано еволюційний метод котячих зграй. Експериментальні дослідження було проведено за допомогою мови програмування Python.

Під час проведення експериментальних досліджень виявилося, що правдоподібний підхід до нечіткої кластеризації даних вирішує недоліки ймовірнісного та можливісного подходів, а еволюційний метод божевільних котів є простим в обчислювальній реалізації і забезпечує високу швидкодію при вирішенні задач нечіткої кластеризації.

Результати роботи було апробовано на конференції «Science, innovations and education: problems and prospects», Токіо, Японія, 2021.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Omran, M. G., Engelbrecht, A. P., & Salman, A. (2007). An overview of clustering methods. Intelligent Data Analysis, 11(6), 583-605

2. Chu, S. C., & Tsai, P. W. (2007). Computational intelligence based on the behavior of cats. International Journal of Innovative Computing, Information and Control, 3(1), 163-173

3. Ezerski, J. C., & Cheung, M. S. (2018). CATS: a tool for clustering the ensemble of intrinsically disordered peptides on a flat energy landscape. The Journal of Physical Chemistry B, 122(49), 11807-11816

4. Xu, R., Wunsch, D.C. (2009), Clustering. Hoboken, N.J. John Wiley & Sons, Inc.

5. Aggarwal, C. C. (2015). Data mining: the textbook. Springer

6. Kohonen, T. (1995). Self-Organizing Maps. Berlin: Springer-Verlag

7. Bezdek, J. C. (1981). Objective function clustering. In Pattern recognition with fuzzy objective function algorithms (pp. 43-93). Springer, Boston, MA

8. Zhou, J., Wang, Q., Hung, C. C., & Yi, X. (2015). Credibilistic clustering: the model and algorithms. International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems, 23(04), 545-564

9. Zhou, J., Wang, Q., Hung, C. C., & Yang, F. (2017). Credibilistic clustering algorithms via alternating cluster estimation. Journal of Intelligent Manufacturing, 28(3), 727-738

10. Young, F. W., Hamer, R. M. (1994). Theory and Applications of Multidimensional Scaling-Hillsdale, N.J.: Erlbaum

11. Zhou, J., & Hung, C. C. (2007). A generalized approach to possibilistic clustering algorithms. International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems, 15(supp02), 117-138.

12. Hu, Z., Bodyanskiy, Y. V., Tyshchenko, O. K., & Shafronenko, A. (2019, July). Fuzzy clustering of incomplete data by means of similarity measures. In 2019 IEEE 2nd Ukraine Conference on Electrical and Computer Engineering (UKRCON) (pp. 957-960). IEEE

13. Grosan, C., Abraham, A., & Chis, M. (2006). Swarm intelligence in data mining. In Swarm Intelligence in Data Mining (pp. 1-20). Springer, Berlin, Heidelberg

14. Chu, S. C., Tsai, P. W., & Pan, J. S. (2006, August). Cat swarm optimization. In Pacific Rim international conference on artificial intelligence (pp. 854-858). Springer, Berlin, Heidelberg

15. Shafronenko, A., Bodyanskiy, Y., Pliss, I., & Patlan, K. (2019, June). Fuzzy Clusterization of Distorted by Missing Observations Data Sets Using Evolutionary Optimization. In 2019 9th International Conference on Advanced Computer Information Technologies (ACIT) (pp. 217-220). IEEE

16. Hu, Z., Bodyanskiy, Y. V., Tyshchenko, O. K., & Shafronenko, A. (2019, July). Fuzzy clustering of incomplete data by means of similarity measures. In 2019 IEEE 2nd Ukraine Conference on Electrical and Computer Engineering (UKRCON) (pp. 957-960). IEEE

17. Shafronenko, A. Y., Bodyanskiy, Y. V., & Pliss, I. P. (2019, September). The Fast Modification of Evolutionary Bioinspired Cat Swarm Optimization Method. In 2019 IEEE 8th International Conference on Advanced Optoelectronics and Lasers (CAOL) (pp. 548-552). IEEE

18. Bodyanskiy, Y., Shafronenko, A., & Pliss, I. (2021). Правдоподібна нечітка кластеризація даних на основі еволюційного методу божевільних котів. System research and information technologies, (3), 110-119

19. Bodyanskiy, Y. V., Shafronenko, A. Y., & Klymova, I. N. (2021). ОНЛАЙН МЕТОД МОЖЛИВІСНОЇ КЛАСТЕРИЗАЦІЇ ДАНИХ НА ОСНОВІ ЕВОЛЮЦІЙНОЇ ОПТИМІЗАЦІЇ КОТЯЧИХ ЗГРАЙ. Radio Electronics, Computer Science, Control, (2), 65-70

20. Shafronenko, A., & Bodyanskiy, Y. V. (2020). Adaptive fuzzy clustering approach based on evolutionary cat swarm optimization. In CMIS (pp. 832-842)

21. Bodyanskiy, Y. V., Shafronenko, A. Y., & Klymova, I. N. (2021). ONLINE FUZZY CLUSTERING OF INCOMPLETE DATA USING CREDIBILISTIC APPROACH AND SIMILARITY MEASURE OF SPECIAL TYPE. Radio Electronics, Computer Science, Control, 1(1), 97-104

22. Бодянський, Є. В., Шафроненко, А. Ю., & Климова, І. М. (2020). Рекурентна достовірна нечітка кластеризація великих даних з використанням функції належності спеціального типу. Біоніка інтелекту, 2(95), 77-81

23. Shafronenko, A., Bodyanskiy, Y., Pliss, I., & Irina, K. (2021, September). Online Credibilistic Fuzzy Clustering Method Based on Cauchy Density Distribution Function. In 2021 11th International Conference on Advanced Computer Information Technologies (ACIT) (pp. 704-707). IEEE

24. Shafronenko, A., Bodyanskiy, Y., Pliss, I., & Popov, S. (2020, September). Evolving neo-fuzzy system for distorted data online processing. In 2020 10th International Conference on Advanced Computer Information Technologies (ACIT) (pp. 352-355). IEEE

25. Shafronenko, A., Bodyanskiy, Y., & Rudenko, D. (2020). Neuro-fuzzy clustering of Distorted Data Using Cat Swarm Optimization. LAP LAMBERT Academic Publishing

26. Shafronenko, A. Y., Bodyanskiy, Y. V., & Rudenko, D. A. (2020). Evolutionary Algorithms Based on Cat Swarm Optomozation in Fuzzy Clustering Tasks

27. Shafronenko, A., & Bodyanskiy, Y. V. (2020). Adaptive fuzzy clustering approach based on evolutionary cat swarm optimization. In CMIS (pp. 832-842)

28. Shafronenko, A., Bodyanskiy, Y., Pliss, I., & Patlan, K. (2019, June). Fuzzy Clusterization of Distorted by Missing Observations Data Sets Using

Evolutionary Optimization. In 2019 9th International Conference on Advanced Computer Information Technologies (ACIT) (pp. 217-220). IEEE

29. Bodyanskiy, Y., Shafronenko, A., & Mashtalir, S. (2019, May). Online Robust Fuzzy Clustering of Data with Omissions Using Similarity Measure of Special Type. In International Scientific Conference “Intellectual Systems of Decision Making and Problem of Computational Intelligence” (pp. 637-646). Springer, Cham

30. Shafronenko, A., Bodyanskiy, Y. V., Klymova, I., & Holovin, O. (2020, May). Online credibilistic fuzzy clustering of data using membership functions of special type. In CMIS (pp. 744-753)

31. Sarangi, A., Sarangi, S. K., Mukherjee, M., & Panigrahi, S. P. (2015, December). System identification by Crazy-cat swarm optimization. In 2015 International Conference on Microwave, Optical and Communication Engineering (ICMOCE) (pp. 439-442). IEEE

32. Ezerski, J. C., & Cheung, M. S. (2018). CATS: a tool for clustering the ensemble of intrinsically disordered peptides on a flat energy landscape. The Journal of Physical Chemistry B, 122(49), 11807-11816

33. Жернова, П. Є. (2019). Нечітка кластеризація потоків даних за умов невідомої кількості кластерів

34. Bodyanskiy, Y., Perova, I., & Zhernova, P. (2019). Online fuzzy clustering of high-dimensional data based on ensembles in data stream mining tasks. The current state of research and technology in industry, 1(7), 16-24

35. Zhernova, P. Y., Deineko, A. O., Bodyanskiy, Y. V., & Riepin, V. O. (2018, August). Adaptive kernel data streams clustering based on neural networks ensembles in conditions of uncertainty about amount and shapes of clusters. In 2018 IEEE Second International Conference on Data Stream Mining & Processing (DSMP) (pp. 7-12). IEEE

36. БОДЯНСЬКИЙ, Є., ДЕЙНЕКО, А., ЖЕРНОВА, П., & РЄПІН, В. (2017). ONLINE MODIFICATION OF THE METHOD OF X-MEDIUM ON THE

BASIS OF ANSAMBLY OF SELORGANIZED MAP T. KOHONEN. Transport development, (1 (1)), 96-107

37. Bahrami, M., Bozorg-Haddad, O., & Chu, X. (2018). Cat swarm optimization (CSO) algorithm. In Advanced optimization by nature-inspired algorithms (pp. 9-18). Springer, Singapore

38. Kumar, Y., & Singh, P. K. (2018). Improved cat swarm optimization algorithm for solving global optimization problems and its application to clustering. Applied Intelligence, 48(9), 2681-2697

39. Bahrami, M., Bozorg-Haddad, O., & Chu, X. (2018). Cat swarm optimization (CSO) algorithm. In Advanced optimization by nature-inspired algorithms (pp. 9-18). Springer, Singapore

40. Ji, X. F., Pan, J. S., Chu, S. C., Hu, P., Chai, Q. W., & Zhang, P. (2020). Adaptive cat swarm optimization algorithm and its applications in vehicle routing problems. Mathematical Problems in Engineering, 2020

41. Шафроненко А. Ю, Москаленко В. В. ПРАВДОПОДІБНА НЕЧІТКА КЛАСТЕРИЗАЦІЯ ДАНИХ НА ОСНОВІ ЕВОЛЮЦІЙНИХ ПРОЦЕДУР. Proceedings of the 5th International scientific and practical conference. CPN Publishing Group. Tokyo, Japan. 2021. Pp. 383-385