НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ»

ННК "Інститут прикладного системного аналізу"
(повна назва інституту/факультету)

Кафедра Системного проектування
(повна назва кафедри)

«До захисту допущено»
Завідувач кафедри _____
А.І.Петренко
(підпис) (ініціали, прізвище)
"___" _____2016 р.

ДИПЛОМНА РОБОТА

першого (бакалаврського) **рівня вищої освіти**
(першого (бакалаврського), другого (магістерського))

зі спеціальності 7.050102, 8.050102 Інформаційні технології проектування
7.050103, 8.050103 Системне проектування
(код та назва спеціальності)

на тему: «Інтелектуальний аналіз геоданих»
Виконав: студент IV курсу, групи групи ДА-22
(шифр групи)

Магас Валентин Васильович _____
(прізвище, ім'я, по батькові) (підпис)

Керівник ст.в., к.т.н. Булах Б.В. _____
(посада, науковий ступінь, вчене звання, прізвище та ініціали) (підпис)

Консультант _____ _____ _____
(назва розділу) (посада, вчене звання, науковий ступінь, прізвище, ініціали) (підпис)

Рецензент _____ _____
(посада, науковий ступінь, вчене звання, науковий ступінь, прізвище та ініціали) (підпис)

Нормоконтроль ст.. викладач Бритов О.А. _____

Засвідчую, що у цій дипломній роботі немає запозичень з праць інших авторів без відповідних посилань. Студент

_____
(підпис)

Київ – 2016 року

Національний технічний університет України
«Київський політехнічний інститут»

Факультет (інститут) ННК "Інститут прикладного системного аналізу"
(повна назва)

Кафедра Системного проектування
(повна назва)

Рівень вищої освіти Перший(Бакалаврський)
(перший (бакалаврський), другий (магістерський) або спеціаліста)

Спеціальність 7.050102, 8.050102 Інформаційні технології проектування

7.050103, 8.050103 Системне проектування
(код і назва)

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ А.І.Петренко
(підпис) (ініціали, прізвище)

«___»_____2016 р.

ЗАВДАННЯ НА ДИПЛОМНИЙ ПРОЕКТ (РОБОТУ) СТУДЕНТУ

Магасу Валентину Васильовичу
(прізвище, ім'я, по батькові)

1. Тема проекту (роботи)) «Інтелектуальний аналіз геоданих»
керівник проекту (роботи)) Булах Б.В., к.т.н., ст.в.,
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом по університету від 12 травня 2016 р. № 50-ст

2. Строк подання студентом проекту (роботи) 08.06.2016

3. Вихідні дані до проекту (роботи)
Дослідити алгоритми алгоритми кластеризації на просторових даних. Для виконання дослідження використати мову програмування Python з необхідними розширеннями та безкоштовні засоби аналізу даних, наприклад Orange. Вхідними даними є база даних SEER . Результати роботи мають включати графіки з результатами кластерного аналізу.

4. Зміст розрахунково-пояснювальної записки (перелік завдань, які потрібно розробити)

1. Обрати методи та аглоритми для аналізу.
2. Обрати програмні засоби.
3. Створити програмну реалізацію.
4. Проаналізувати результати проведеного дослідження, зробити висновки.

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслеників, плакатів тощо)

1. Структурні елементи процесу ІАПД– плакат.
2. Результати кластеризації – плакат
3. Результати порівняння алгоритмів – плакат.

6. Консультанти розділів проекту (роботи)

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Економічний	Семенченко Н.В., професор		

7. Дата видачі завдання 01.02.2016

Календарний план

№ з/п	Назва етапів виконання дипломного проекту (роботи)	Строк виконання етапів проекту (роботи)	Примітка
1	Отримання завдання	01.02.2016	
2	Збір інформації	15.02.2016	
3	Дослідження існуючих алгоритмів ІАД та вибір варіанту для розробки	28.02.2016	
4	Розробка алгоритму та структури системи	10.03.2016	
5	Розробка плану тестування	15.03.2016	
6	Розробка програмної моделі	25.03.2016	
7	Робота над описом системи	25.04.2016	
8	Тестування додатку та отримання результатів	30.04.2016	
9	Оформлення дипломної роботи	31.05.2016	
10	Отримання допуску до захисту та подача роботи в ДЕК	08.06.2016	

Студент _____ В.В. Магас
(підпис) (ініціали, прізвище)

Керівник проекту (роботи) _____ Б.В. Булах
(підпис) (ініціали, прізвище)

АНОТАЦІЯ

бакалаврської дипломної роботи Магаса Валентина Васильовича

на тему «Інтелектуальний аналіз геоданих»

Величезні розміри прихованої інформації у великих об’ємах даних зумовили наростаючий інтерес до сфери інтелектуального аналізу даних.

Метою даної роботи є застосування засобів інтелектуального аналізу даних для виявлення прихованих закономірностей та зв’язків у просторових баз даних онкохворих пацієнтів. Інтелектуальний аналіз просторових даних включає виявлення цікавих та потенційно корисних шаблонів з просторових баз даних шляхом групування об’єктів у кластери. Дана робота зосереджується як на дискретних так і неперервних просторових медичних базах даних, до яких застосовуються кращі методи кластеризації для визначення максимально корисних кластерів.

За результатами виконання бакалаврської роботи було подано тези на 18-ту міжнародну конференцію SAIT 2016. Тези було надруковано в збірнику тез відповідної конференції.

Загальний обсяг роботи 104с., 33 рис., 9 табл., 15 джерел.

Ключові слова: k-means, інтелектуальний аналіз просторових даних, алгоритми кластеризації, дендограма, просторовий розподіл

АННОТАЦИЯ

бакалаврской дипломной работы Магаса Валентина Васильевича

на тему «Интеллектуальный анализ геоданных»

Огромные размеры скрытой информации в больших объемах данных обусловили растущий интерес к сфере интеллектуального анализа данных.

Целью данной работы является применение средств интеллектуального анализа данных для выявления скрытых закономерностей и связей в пространственных базах данных онкобольных пациентов. Интеллектуальный анализ пространственных данных включает выявление интересных и потенциально полезных шаблонов из пространственных базах данных путем группирования объектов в кластеры. Данная работа направлена на использование как дискретных так и непрерывных пространственных медицинских баз данных, на которых применяются лучшие методы кластеризации для определения максимально полезных кластеров.

По результатам выполнения бакалаврской работы были представлены тезисы на 18-ю международную конференцию SAIT 2016. Тезисы были напечатаны в сборнике тезисов соответствующей конференции.

Общий обьем роботы 104с., 33 рис., 9 табл., 15 джерел.

Ключевые слова: k-means, интеллектуальный анализ пространственных данных, алгоритмы кластеризации, дендограма, пространственное распределение.

ANNOTATION

On Valentyn Mahas bachelor's degree

thesis: "Spatial data mining"

The vast amount of hidden information in large volumes of data led to growing interest to the sphere of data mining. The aim of this research is the application of data mining for detecting hidden patterns and relationships in spatial cancer databases. Spatial data mining includes discovery of interesting and useful patterns from spatial databases by grouping the objects into clusters. This study focuses on discrete and continuous spatial medical databases on which clustering techniques are applied and the efficient clusters were formed.

As the results of this work, abstracts were submitted for the 18th International Conference SAIT 2015. Abstracts were published in collection of abstracts of relevant conference.

Bachelor's work size 104p., 33 pic., 9 tables, 15 sources.

Keywords: k-means, spatial data mining, clustering algorithm, dendogram, spatial distribution

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ

ІАД – Інтелектуальний аналіз даних

ООП – Об'єктно-орієнтоване програмування

ІАПД – Інтелектуальний аналіз просторових даних

BIRCH – Balanced Iterative Reducing and Clustering using Hierarchies

DBSCAN – Density-based spatial clustering of applications with noise

OLAP – Online analytical processing

PAM – Partitioning around Medoids

CLARA – Clustering LARge Application

OPTICS – Ordering points to identify the clustering structure

SADT – Structured Analysis and Design Technique

GTK – The GIMP ToolKit

ARI – Adjusted Rand Index

AMI – Adjusted mutal information

ВСТУП

Сучасний період розвитку суспільства характеризується значним впливом на нього інформаційних технологій. Не стала винятком і медицина, яка сьогодні набула абсолютно нових рис. Поміж всієї ширини застосування новітніх технологій та засобів у даній сфері, чільне місце посідає інтелектуальний аналіз даних.

Інтелектуальний аналіз даних - збірна назва, що використовується для позначення сукупності методів виявлення в даних раніше невідомих, нетривіальних, практично корисних і доступних інтерпретації знань, необхідних для прийняття рішень в різних сферах людської діяльності.

Актуальність.

У зв’язку бурхливим накопиченням медичних даних, особливо гостро постає проблема застосування аналізу даних для їх опрацювання та подальшого прогнозування, на їх основі. В свою чергу необхідність визначення прихованих тенденцій поширення певного захворювання на тій чи іншій території, виявлення неочевидних зв’язків між факторами що його спричиняють та подальшого прогнозування можливого розвитку подій з кожним роком набуває все чіткіших рис. Дуже актуальним виглядає дослідження просторової бази даних онкохворих, адже дана хвороба відзначається значною смертністю серед хворих, як результат, кожного року помирає понад 15 мільйонів людей у цілому світі.

Мета.

Метою даної роботи є застосування засобів інтелектуального аналізу даних для виявлення прихованих закономірностей та зв’язків у просторових баз даних онкохворих пацієнтів. Інтелектуальний аналіз просторових даних включає виявлення цікавих та потенційно корисних шаблонів з просторових баз даних шляхом групування об’єктів у кластери. Дана робота зосереджується як на дискретних так і неперервних просторових медичних базах даних, до яких

застосовуються кращі методи кластеризації для визначення максимально корисних кластерів.

Основні завдання.

До основних завдань, що виносяться на дипломну роботу відносяться:

- З’ясування специфічних для сфери ІАПД вимог.
- Дослідження існуючих рішень і підходів.
- Вивчення підходів до аналізу просторових даних з використанням з використання алгоритмів кластеризації.
- Постановка прикладної задачі, яка б надала можливість для детального дослідження.
- Безпосереднє використання засобів дослідження до поставленої задачі.
- Аналіз результатів, оцінка використаних та досліджених підходів, визначення переваг і недоліків створеного рішення.

Об’єкт дослідження – застосування технологій ІАПД у медицині.

Предмет дослідження – кластерний аналіз просторових баз даних онкохворих пацієнтів.

1. ВИБІР МЕТОДІВ ТА АЛГОРИТМІВ

1.1 Огляд предметної області

1.1.1 Передмова

Data Mining – мультидисциплінарна область, що виникла і розвивається на базі таких наук як прикладна статистика, розпізнавання образів, штучний інтелект, теорія баз даних і ін.

Технологія Data Mining - це процес виявлення в «сирих» даних раніше невідомих, нетривіальних, практично корисних і доступних інтерпретації знань, необхідних для прийняття рішень в різних сферах людської діяльності.

Технологія Data Mining призначена для пошуку у великих обсягах даних неочевидних, об'єктивних і корисних на практиці закономірностей.

Неочевидних - це значить, що знайдені закономірності не виявляються стандартними методами обробки інформації або експертним шляхом.

Об'єктивних - це значить, що виявлені закономірності будуть повністю відповідати дійсності, на відміну від експертної думки, яка завжди є суб'єктивною.

Практично корисних - це значить, що висновки мають конкретне значення, якому можна знайти практичне застосування.

В основу технології Data Mining покладена концепція шаблонів (patterns), які являють собою закономірності, властиві підвибіркам даних, які можуть бути виражені в формі, зрозумілій людині.

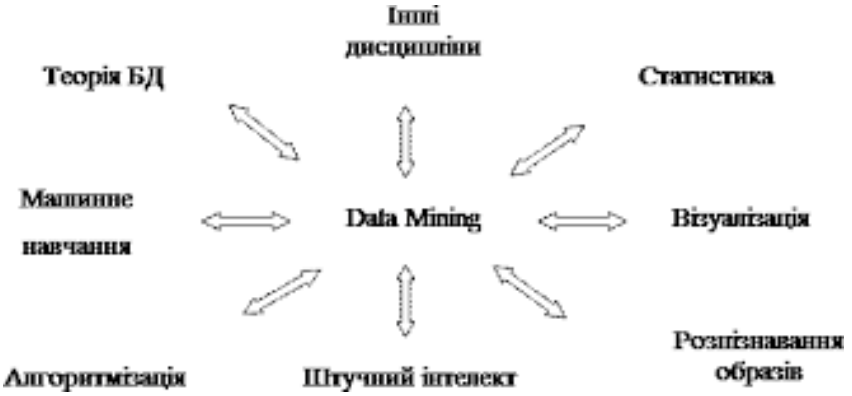

Рисунок 1.1 – Data Mining як мультидисциплінарна область[1]

На рисунку 1.1 наведено складові інтелектуального аналізу даних.

Метою пошуку закономірностей є подання даних у вигляді, що відображає шукані процеси. Побудова моделей прогнозування також є метою пошуку закономірностей.

Засоби Data Mining, на відміну від статистичних, не вимагають наявності строго певної кількості ретроспективних даних. Ця особливість може стати причиною виявлення недостовірних, хибних моделей і, як результат, прийняття на їх основі неправильних рішень. Необхідно здійснювати контроль статистичної значущості виявлених знань.

Інтелектуальний аналіз просторових даних(ІАПД) - це процес виявлення раніше невідомих але потенційно корисних та цікавих закономірностей з великих наборів просторових даних.

Умовно структуру інтелектуального аналізу просторових даних можна поділити на три шари, як показано на Рисунку 1.2. Перший шар, тобто, інтерфейс користувача використовується для завдань вводу і виводу. Дані беруться зі сховища з використанням інтерфейсу бази даних, що дозволяє оптимізувати запити. Другий шар переважно використовується для управління даними, вибору алгоритмів обробки та зберігання добутих знань. Для прикладу, він може вирішити чи деякі атрибути відповідають поставленим завданням, або вибрати об’єкти чиє використання обіцяє хороші результати. Третій і останній шар включає базу просторових даних і базу даних предметної області. Вхідні дані інтелектуального аналізу просторових даних значно складніші ніж класичного, адже вони включають розширені об’єкти, такі як лінії, багатокутники та інші.

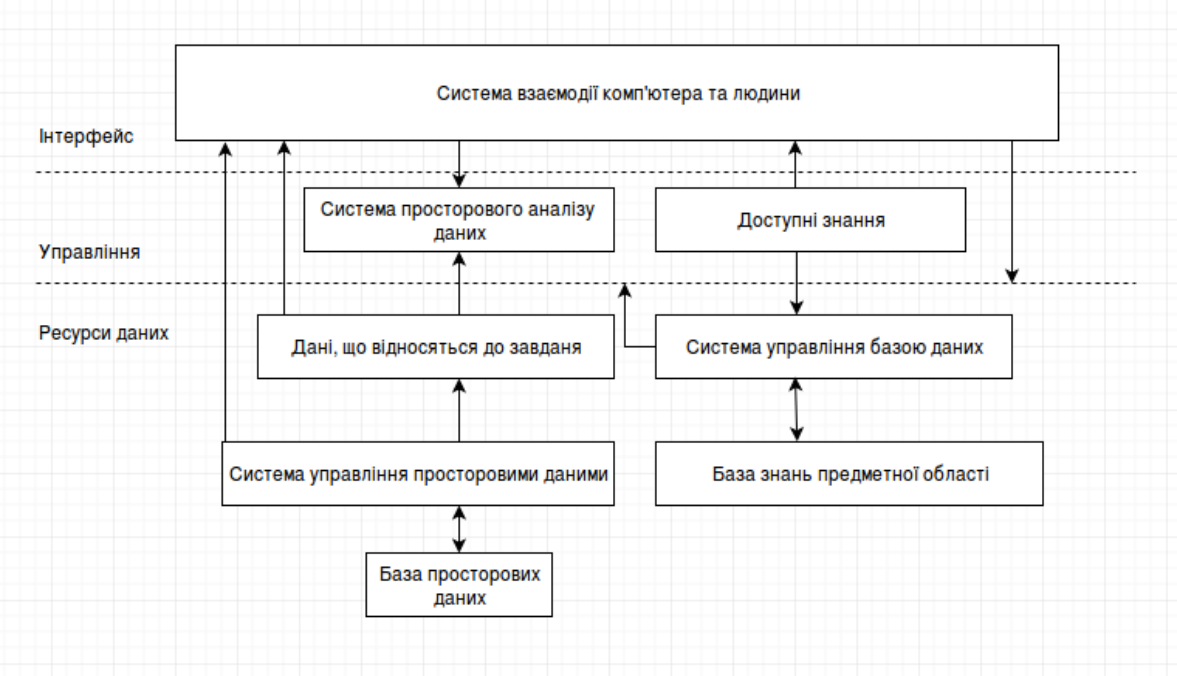

Рисунок 1.2 – Структура ІАПД

Інтелектуальний аналіз просторових даних пропонує значні вигоди для прикладного прийняття рішень на основі геоінформаційної системи. Останнім часом, завдання інтеграції цих двох технологій стало вирішальним фактором, особливо в різних організаціях державного і приватного секторів, що володіють величезними базами географічно пов’язаних даних, адже вони усвідомили величезний потенціал, що міститься у цій інформації. До них можна віднести:

- організації, що потребують аналізу чи поширення статистичних геоданих
- організації, які займаються сферою охорони здоров’я і шукають закономірності поширення певної хвороби
- природоохоронні установи, які оцінюють вплив зміни характеру землекористування в області зміни клімату
- геомаркетингові компанії, що займаються сегментацією клієнтів, грунтуючись на їхньому просторовому розташуванні

1.1.2 Представлення даних

У широкому розумінні дані являють собою факти, текст, графіки, картинки, звуки, аналогові або цифрові відео-сегменти. Дані можуть бути отримані в результаті вимірів, експериментів, арифметичних і логічних операцій.

Дані повинні бути представлені у формі, придатній для зберігання, передачі і обробки. Іншими словами, дані - це необроблений матеріал, що надається постачальниками даних і використовується споживачами для формування інформації на основі даних.

Найбільш часто зустрічаються дані , що складаються із записів (record data). Це табличні дані, матричні дані, документальні дані, транзакційні або операційні. Табличні дані складаються з записів, кожна з яких складається з фіксованого набору атрибутів. Транзакційні дані являють собою особливий тип даних, де кожен запис, що є транзакцією, включає набір значень.

Іншою важливою категорією даних є графічні дані, вони включають різного роду www-дані, графи(Рисунок 1.3), карти та інші.

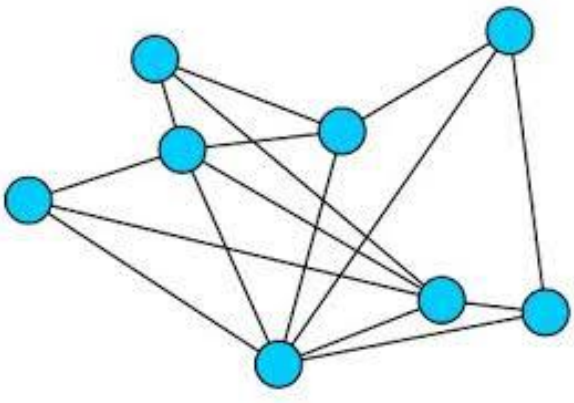

Рисунок 1.3 – Графове представлення даних[5]

За допомогою карт, наприклад, можна відстежити зміни об'єктів в часі і просторі, визначити характер їх розподілу на площині або в просторі.

Перевагою графічного представлення даних є велика простота їх сприйняття, ніж, наприклад, табличних даних. Приклад карти Кохонена (модель нейронних мереж), представлений на рисунку 1.4.

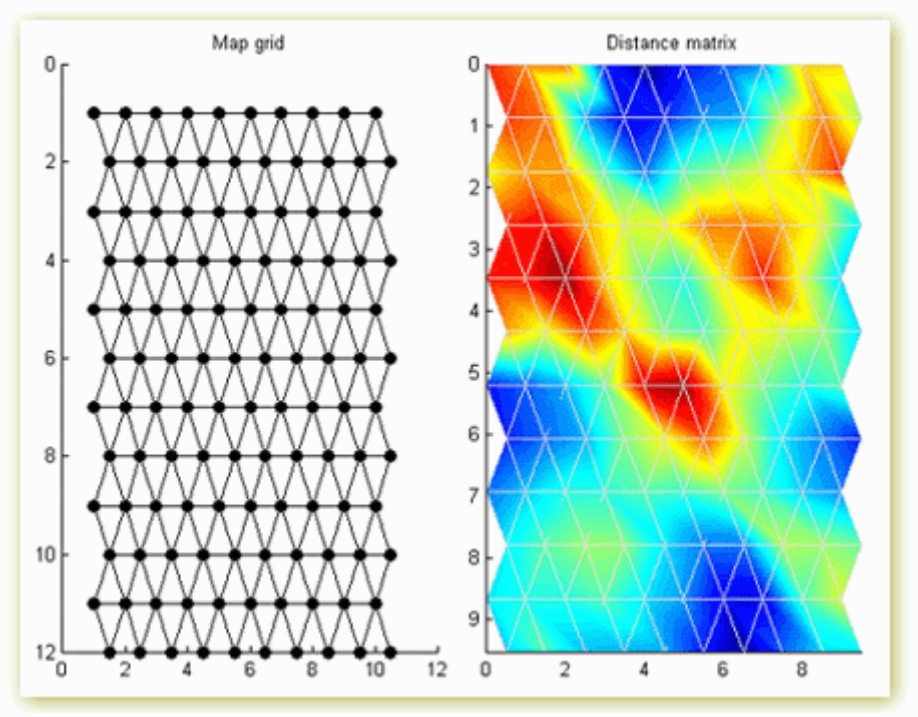

Рисунок 1.4 – Приклад даних типу “Карта Кохонена”[3]

Класифікація видів даних.

Реляційні дані - це дані з реляційних баз – таблиць (Логічна модель даних).

Багатовимірні дані - це дані, представлені в кубах OLAP (аналітична обробка в реальному часі).

Вимірювання (dimension) або вісь - в багатовимірних даних - це зібрання даних одного і того ж типу, що дозволяє структурувати багатовимірну базу даних.

За критерієм сталості своїх значень в ході вирішення поставленого завдання дані можуть бути:

1) змінними - дані, які змінюють свої значення в процесі виконання завдання;

2) постійними - дані, які зберігають свої значення в процесі виконання завдання (математичні константи, координати нерухомих об'єктів) і не залежать від зовнішніх чинників;

3) умовно-постійними - дані, які можуть іноді змінювати свої значення, але ці зміни не залежать від процесу вирішення задачі, а визначаються зовнішніми факторами.

Дані, в залежності від тих функцій, які вони виконують, можуть бути: *довідковими, оперативними, архівними*.

Слід розрізняти *дані за період* і *точкові дані*. Ці відмінності важливі при проектуванні системи збору інформації, а також в процесі вимірювань.

Дані за період характеризують деякий період часу.

Точкові дані представляють значення деякої змінної в конкретний момент часу.

Також, характерним э той факт, що дані бувають первинними і вторинними. Вторинні дані - це дані, які є результатом певних обчислень, застосованих до первинних даних. Вторинні дані, як правило, призводять до прискореного отримання відповіді на запит користувача за рахунок збільшення об'єму інформації. Первинні дані отримують безпосереднім виміром (наглядом).

Метадані (Metadata) - це дані про дані.

До складу метаданих можуть входити: каталоги, довідники, реєстри. Метадані містять відомості про склад даних, зміст, статус, походження, місцезнаходження, як, форматах і формах уявлення, умови доступу, придбання і використання, авторських, майнових і суміжних з ними правах на дані та ін. Метадані, що застосовуються при управлінні сховищем, містять інформацію, необхідну для його налаштування і використання.

Прийнято виділяти бізнес-метадані та оперативні метадані.

Бізнес-метадані містять бізнес-терміни і визначення, належність даних і правила оплати послуг сховища.

Оперативні метадані - це інформація, зібрана під час роботи сховища даних, вона включає:

- походження перенесених і перетворених даних;
- статус використання даних (активні, заархівовані або віддалені);
- дані моніторингу, такі як статистика використання, повідомлення про помилки і т.д.

Метадані сховища зазвичай розміщуються в репозиторії (зазвичай дані в репозиторії зберігаються у вигляді файлів, доступних для подальшого поширення по мережі). Це дозволяє використовувати метадані спільно з різними інструментами і процесам при проектуванні, установці, експлуатації та адмініструванні сховища.

1.1.3 Етапи процесу ІАД

Процес Data Mining складається з певних етапів, що включають елементи порівняння, типізації, класифікації, узагальнення, абстрагування, повторення. Він нерозривно пов'язаний з процесом прийняття рішень.

Традиційний процес Data Mining включає наступні етапи:

- аналіз предметної області;
- постановка задачі;
- підготовка даних;
- побудова моделей;
- перевірка та оцінка моделей;
- вибір моделі;
- застосування моделі;
- корекція і оновлення моделі.

Етап 1. Аналіз предметної області.

Дослідження - це процес пізнання певної предметної області, об'єкта чи явища з певною метою. Процес дослідження полягає в спостереженні властивостей об'єктів з метою виявлення і оцінки важливих, з точки зору суб'єкта-дослідника, закономірних відносин між показниками даних властивостей.

Предметна область - це подумки обмежена область реальної дійсності, що підлягає опису або моделювання та дослідження. Вона складається з об'єктів, що розрізняються за властивостями і знаходяться в певних відносинах між собою або взаємодіючих яким-небудь чином.

Предметна область - це частина реального світу, вона містить як істотні, так і не значущі дані, з точки зору проведеного дослідження. Істотність даних залежить від вибору предметної області.

У процесі вивчення предметної області повинна бути створена її модель. Знання з різних джерел повинні бути формалізовані за допомогою будь-яких засобів. Це можуть бути текстові описи предметної області або спеціалізовані графічні нотації. Існує велика кількість методик опису предметної області: наприклад, методика структурного аналізу SADT і заснована на ньому IDEF0, діаграми потоків(Рисунок 1.5) даних Гейне-Сарсона, методика об'єктно-орієнтованого аналізу UML та інші.

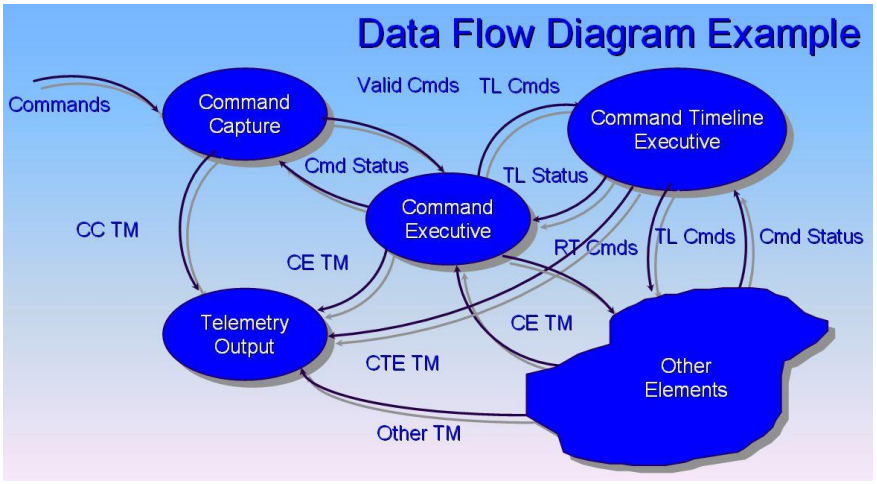

Рисунок 1.5 – Приклад діаграми потоку даних[7]

Модель предметної області описує процеси, що відбуваються в предметній області, і дані, які в цих процесах використовуються. Від того, наскільки вірно змодельована предметна область, залежить успіх подальшої розробки програми Data Mining.

Етап 2. Постановка завдання.

Постановка завдання Data Mining включає наступні кроки:

- формулювання завдання;
- формалізація завдання.

Постановка завдання включає також опис статичної і динамічного поведінки досліджуваних об'єктів. Опис статики передбачає опис об'єктів і їх властивостей. При описі динаміки описується поведінка об'єктів і ті причини, які впливають на їх поведінку. Динаміка поведінки об'єктів часто описується разом зі статикою. Іноді етапи аналізу предметної області і постановки завдання об'єднують в один етап.

Етап 3. Підготовка даних.

Метою даного етапу являється належна розробка розробка бази даних для подальшого аналізу даних. Підготовка даних є найважливішим етапом, від якості виконання якого залежить можливість отримання якісних результатів усього процесу Data Mining. Розглянемо сновні кроки цього етапу.

1. Визначення та аналіз вимог до даних.

На цьому кроці здійснюється так зване моделювання даних, тобто визначення та аналіз вимог до даних, які необхідні для здійснення Data Mining. При цьому вивчаються питання розподілу даних (географічне, організаційне, функціональне); питання доступу до даних, які необхідні для аналізу, необхідність у зовнішніх і / або внутрішніх джерелах даних; а також аналітичні характеристики системи (вимірювання даних, основні види вихідних документів, послідовність перетворення інформації та ін.).

2. Збір даних.

Наявність в організації сховища даних робить аналіз простішим і ефективнішим, його використання, з точки зору вкладень, обходиться дешевше, ніж використання окремих баз даних. Однак далеко не завжди є сховища даних. У цьому випадку джерелом для вихідних даних є оперативні, довідкові та архівні БД, тобто дані з існуючих інформаційних систем. Також для Data Mining може знадобитися інформація з інформаційних систем зовнішніх джерел, паперових носіїв, а також знання експертів або результати опитувань.

На цьому кроці здійснюється кодування деяких даних. При визначенні необхідної кількості даних слід враховувати, чи є дані впорядкованими чи ні. Якщо дані впорядковані і мають справу з тимчасовими рядами, бажано знати, чи включає такий набір даних сезонну / циклічну компоненту. У разі присутності в наборі даних сезонної / циклової компоненти, необхідно мати дані як мінімум за один сезон / цикл.

Якщо дані не впорядковані, тобто події з набору даних не пов'язані за часом, в ході збору даних слід дотримуватися таких правил.

- Недостатня кількість записів в наборі даних може стати причиною побудови некоректною моделі. З точки зору статистики, точність моделі збільшується зі збільшенням кількості досліджуваних даних.
- Можливо, деякі дані є застарілими або описують якусь нетипову ситуацію, і їх потрібно виключити з бази даних.
- Алгоритми, що використовуються для побудови моделей на надвеликих базах даних, повинні бути масштабованими.
- При використанні багатьох алгоритмів необхідна певне співвідношення вхідних змінних і кількості спостережень. Кількість записів (прикладів) в наборі даних має бути значно більше кількості чинників (змінних).
- Набір даних повинен бути репрезентативним і представлятиме якомога більше можливих ситуацій. Пропорції представлення різних прикладів в наборі даних повинні відповідати реальній ситуації.

3. Попередня обробка даних.

Аналізувати можна як якісні, так і неякісні дані. Результат буде досягнутий і в одному, і в іншому випадку. Для забезпечення якісного аналізу необхідне проведення попередньої обробки даних, яка є необхідним етапом процесу Data Mining. Дані, отримані в результаті збору, повинні відповідати певним критеріям якості. Таким чином, можна виділити важливий підетап процесу Data Mining - оцінювання якості даних.

Якість даних (Data quality) - це критерій, який визначає повноту, точність, своєчасність і можливість інтерпретації даних.

Дані можуть бути високої якості і низької якості, останні - це так звані брудні або "погані" дані. Дані високої якості - це повні, точні, своєчасні дані, які піддаються інтерпретації. Такі дані забезпечують отримання якісного результату: знань, які зможуть підтримувати процес прийняття рішень.

Дані низької якості, або брудні дані - це відсутні, неточні або непотрібні дані з точки зору практичного застосування (наприклад, представлені в невірному форматі, який не відповідає стандарту).

Найбільш поширені види брудних даних:

- пропущені значення;
- дублікати даних;
- шуми і викиди.

Деякі значення даних можуть бути пропущені у зв'язку з тим, що:

- дані взагалі не були зібрані;
- деякі атрибути можуть бути незастосовні для деяких об'єктів.

Для вирішення проблеми з пропущеними даними потрібно провести попердню їх обробку, а саме:

1. Виключити об'єкти з пропущеними значеннями з аналізу.
2. Розрахувати нові значення для пропущених даних.

3. Ігнорувати пропущені значення в процесі аналізу.
4. Замінити пропущені значення на можливі значення.

Набір даних може включати продубльовані дані, тобто дублікати. Дублікатами називаються записи з однаковими значеннями всіх атрибутів. Наявність дублікатів в наборі даних може бути способом підвищення значущості деяких записів. Така необхідність іноді виникає для особливого виділення певних записів з набору даних. Однак в більшості випадків, продубльовані дані є результатом помилок при підготовці даних.

Існує два варіанти обробки дублікатів. При першому варіанті видаляється вся група записів, що містить дублікати. Цей варіант використовується в тому випадку, якщо наявність дублікатів викликає недовіру до інформації, повністю її знецінює. Другий варіант полягає в заміні групи дублікатів на один унікальний запис.

Викиди – об’єкти, що помітно відрізняються в певному наборі даних. Шуми і викиди є досить загальною проблемою в аналізі даних. Викиди можуть як являти собою окремі спостереження, так і бути об'єднаними в якісь групи. Завдання аналітика - не тільки їх виявити, але і оцінити ступінь їх впливу на результати подальшого аналізу. Якщо викиди є інформативною частиною аналізованого набору даних, використовують робастні методи і процедури.

Досить поширена практика проведення двоетапного аналізу - З викидами і з без них - і порівняння отриманих результатів.

Різні методи Data Mining мають різну чутливість до викидів, цей факт необхідно враховувати при виборі методу аналізу даних. Також деякі інструменти Data Mining мають вбудовані процедури очищення від шумів і викидів. Візуалізація даних дозволяє представити дані, в тому числі і викиди, в графічному вигляді.

Очевидно, що результати Data Mining на основі брудних даних не можуть вважатися надійними і корисними, очевидно необхідне очищення даних.

Очищення даних (data cleaning, data cleansing або scrubbing) займається виявленням і видаленням помилок і невідповідностей в даних з метою поліпшення якості даних.

Етап 4. Побудова моделей.

У широкому сенсі слова *моделювання* - це наукова дисципліна, мета якої - вивчення методів побудови і використання моделей для пізнання реального світу. Існує величезна кількість ситуацій, коли експериментувати в реальному житті не представляються можливими. У цих випадках якраз і застосовується моделювання.

Моделювання як процес являє собою побудову моделі і вивчення її властивостей, які подібні до найбільш важливим, з точки зору аналітика, властивостям досліджуваних об'єктів.

Таким чином, за допомогою моделювання вивчаються властивості об'єктів шляхом дослідження відповідних властивостей побудованих моделей.

Моделювання широко застосовується при використанні методів Data Mining. Шляхом використання моделей Data Mining здійснюється аналіз даних. За допомогою моделей Data Mining виявляється корисна, раніше невідома, доступна інтерпретації інформація, яка використовується для прийняття рішень.

Модель являє собою спрощене уявлення про реальний об'єкт, процес або явище. Створення і використання Data Mining моделі є ключовим моментом для початку розуміння, осмислення і прогнозування тенденцій аналізованого об'єкта. Побудова моделей Data Mining здійснюється з метою дослідження або вивчення модельованого об'єкта, процесу, явища і отримання нових знань, необхідних для прийняття рішень. Використання моделей Data Mining дозволяє визначити оптимальне рішення в конкретній ситуації. Аналітик створює модель як подобу досліджуваного об'єкта. Моделі можуть бути записані у вигляді різних зображень, схем, математичних формул і т.д. Схематичний приклад моделі було розглянуто в лекції, присвяченій задачі класифікації, в першому розділі курсу.

Перевагою використання моделей при дослідженнях є простота моделі в порівнянні з досліджуваним об'єктом. При цьому моделі дозволяють виділити в об'єкті найбільш істотні фактори з точки зору мети дослідження, і не відволікатися на незначні деталі.

Етап 5. Перевірка і оцінка моделей.

Перевірка моделі передбачає перевірку її достовірності або адекватності. Ця перевірка полягає у визначенні ступеня відповідності моделі реальності. Адекватність моделі перевіряється шляхом тестування.

Адекватність моделі (adequacy of a model) - відповідність моделі модельованого об'єкту або процесу.

Поняття достовірності та адекватності є умовними, оскільки ми не можемо розраховувати на повну відповідність моделі реальному об'єкту, інакше це був би сам об'єкт, а не модель. У процесі перевірки моделі необхідно встановити включення в модель всіх істотних чинників. Складність вирішення цієї проблеми залежить від складності розв'язуваної задачі.

Перевірка моделі також має на меті визначення того рівня, в якій вона дійсно допомагає менеджеру при прийнятті рішень. Оцінка моделі передбачає перевірку її правильності. Оцінка побудованої моделі здійснюється шляхом її тестування.

Тестування моделі включає в себе проведення безлічі експериментів. На вхід моделі можуть подаватися вибірки різного об'єму. З точки зору статистики, точність моделі збільшується зі збільшенням кількості досліджуваних даних. Алгоритми, що є основою для побудови моделей на надвеликих базах даних, повинні мати властивість масштабування.

Якщо модель досить складна, а значить, потрібно багато часу на її навчання і подальшу оцінку, то іноді буває можна побудувати і протестувати модель на невеликій частині вибірки. Однак цей варіант підходить тільки для однорідних даних, в іншому випадку необхідно використовувати всі доступні дані. Побудовані моделі рекомендується тестувати на різних вибірках для визначення їх узагальнюючих здібностей. В ході експериментів можна

варіювати обсяг вибірки (кількість записів), набір вхідних і вихідних змінних, використовувати вибірки різної складності.

Виявлені співвідношення і закономірності повинні бути проаналізовані експертом в предметній області - він допоможе визначити, як є з'ясовані закономірності (можливо, занадто загальними або вузькими і специфічними). Для оцінки результатів отриманих моделей слід використовувати знання фахівців предметної області. Якщо результати отриманої моделі експерт вважає незадовільними, слід повернутися на один з попередніх кроків процесу Data Mining, а саме: підготовка даних, побудова моделі, вибір моделі. Якщо ж результати моделювання експерт вважає прийнятними, її можна застосовувати для вирішення реальних завдань.

Етап 6. Вибір моделі.

Якщо в результаті моделювання нами було побудовано кілька різних моделей, то на підставі їх оцінки ми можемо здійснити вибір кращої з них. В ході перевірки і оцінки різних моделей на підставі їх характеристик, а також з урахуванням думки експертів, слід обрати найкращу. Досить часто це виявляється непростим завданням.

Основні характеристики моделі, які визначають її вибір, - це точність моделі і ефективність роботи алгоритму .

У деяких програмних продуктах реалізований ряд методів, розроблених для вибору моделі. Багато з них засновані на так званій "конкурентної оцінці моделей", яка полягає в застосуванні різних моделей до одного і того ж набору даних і наступному порівнянні їх характеристик.

Етап 7. Застосування моделі.

Після тестування, оцінки та вибору моделі наступає етап застосування моделі. На цьому етапі обрана модель застосовується до нових даних з метою вирішення завдань, поставлених на початку процесу Data Mining. Для класифікаційних і прогнозуючих моделей на цьому етапі прогнозується цільовий (вихідний) атрибут (target attribute).

Етап 8. Корекція і оновлення моделі.

Після певного встановленого проміжку часу з моменту початку використання моделі Data Mining слід проаналізувати отримані результати, визначити, чи справді вона "успішна" або ж виникли проблеми і складності в її використанні.

Однак навіть якщо модель з успіхом використовується, її не слід вважати абсолютно вірною. Справа в тому, що необхідно періодично оцінювати адекватність моделі набору даних, а також поточну ситуацію (слід враховувати можливість зміни зовнішніх факторів). Навіть найточніша модель з часом перестає бути такою. Для того щоб побудована модель виконувала свою функцію, слід працювати над її корекцією (поліпшенням). При появі нових даних потрібно повторне навчання моделі. Цей процес називають оновленням моделі. Роботи, що проводяться з моделлю на цьому етапі, також називають контролем і супроводом моделі.

Існує багато причин, що вимагають навчити модель заново, тобто оновити її, щоб відобразити певні зміни.

Основними причинами є наступні:

- змінилися вхідні дані або їх поведінка;
- з'явилися додаткові дані для навчання;
- змінилися вимоги до форми і кількості вихідних даних;
- змінилися цілі, які вплинули на критерії прийняття рішень;
- змінилося зовнішнє оточення або середовище

1.1.4 Основні задачі ІАД

В основу технології Data Mining покладена концепція шаблонів, що являють собою закономірності. В результаті виявлення цих, прихованих від неозброєного ока закономірностей вирішуються завдання Data Mining. Різним типам закономірностей, які можуть бути виражені в формі, зрозумілою людині, відповідають певні завдання Data Mining.

Зазвичай виділяють наступні задачі: класифікація, кластеризація, прогнозування, асоціація, візуалізація, аналіз і виявлення відхилень, оцінювання, аналіз зв'язків, підведення підсумків.

Класифікація (Classification). Найбільш проста і поширена задача Data Mining. В результаті рішення задачі класифікації виявляються ознаки, які характеризують групи об'єктів досліджуваного набору даних - класи; за цими ознаками новий об'єкт можна віднести до того чи іншого класу. Для вирішення завдання класифікації можуть використовуватися методи: найближчого сусіда (Nearest Neighbor); k- найближчого сусіда (k-Nearest Neighbor); байєсовські мережі (Bayesian Networks); індукція дерев рішень; нейронні мережі (neural networks).

Кластеризація (Clustering). Кластеризація є логічним продовженням ідеї класифікації. Її завдання більш складне, особливість кластеризації полягає в тому, що класи об'єктів спочатку не визначені. Результатом кластеризації є розбиття об'єктів на групи. Приклад методу розв'язання задачі кластеризації: навчання "без вчителя" особливого виду нейронних мереж - самоорганізованих карт Кохонена

Асоціація (Associations). В результаті виконання завдання пошуку асоціативних правил відшукуються закономірності між пов'язаними подіями в наборі даних. Відмінність асоціації від двох попередніх задач Data Mining пошук закономірностей здійснюється не на основі властивостей аналізованого об'єкта, а між кількома подіями, які відбуваються одночасно. Найбільш відомий алгоритм рішення задачі пошуку асоціативних правил - алгоритм Apriori.

Послідовність (Sequence), або послідовна асоціація (sequential association). Послідовність дозволяє знайти тимчасові закономірності між транзакціями. Завдання послідовності подібна асоціації, але її метою є встановлення закономірностей не між одночасно наступаючими подіями, а між подіями, пов'язаними в часі (тобто відбуваються з деяким певним інтервалом у часі). Іншими словами, послідовність визначається високою ймовірністю ланцюжка пов'язаних у часі подій. Фактично, асоціація є окремим випадком

послідовності з тимчасовим лагом, рівним нулю. Це завдання Data Mining також називають завданням знаходження послідовних шаблонів (sequential pattern). Правило послідовності: після події X через певний час відбудеться подія Y.

Прогнозування (Forecasting). В результаті рішення задачі прогнозування на основі особливостей історичних даних оцінюються пропущені або ж майбутні значення цільових чисельних показників. Для вирішення таких завдань широко застосовуються методи математичної статистики, нейронні мережі та ін.

Визначення відхилень або викидів (Deviation Detection), аналіз відхилень або викидів. Мета рішення даного завдання - виявлення та аналіз даних, найбільш відрізняються від загальної множини даних, виявлення так званих нехарактерних шаблонів.

Оцінювання (Estimation). Завдання оцінювання зводиться до передбачення неперервних значень ознаки.

Аналіз зв'язків (Link Analysis) - задача знаходження залежностей в наборі даних.

Візуалізація (Visualization, Graph Mining). В результаті візуалізації створюється графічний образ аналізованих даних. Для вирішення завдання візуалізації використовуються графічні методи, що показують наявність закономірностей в даних. Приклад методів візуалізації - представлення даних в 2-D і 3-D вимірах.

Підведення підсумків (Summarization) - задача, мета якої - опис конкретних груп об'єктів з аналізованого набору даних.

Відповідно до класифікації за стратегіями, завдання Data Mining підрозділяються на наступні групи:

- навчання з учителем;
- навчання без вчителя;
- інші.

Категорія навчання з учителем представлена наступними завданнями Data Mining: класифікація, оцінка, прогнозування. Категорія навчання без вчителя представлена завданням кластеризації. У категорію «інші» входять завдання, не включені в попередні дві стратегії.

Відповідно до цієї класифікації, завдання Data Mining представлені групами описових і прогнозтичних завдань.

В результаті рішення описових (descriptive) завдань аналітик отримує шаблони, що описують дані, які піддаються інтерпретації. Ці завдання описують загальну концепцію аналізованих даних, визначають інформативні, підсумкові, відмінні риси даних. Концепція описових завдань має на увазі характеристику і порівняння наборів даних. Характеристика набору даних забезпечує короткий і стислий опис деякого набору даних. Порівняння забезпечує порівняльне опис двох або більше наборів даних.

Прогнозуючі (predictive) ґрунтуються на аналізі даних, створення моделі, передбаченні тенденцій або властивостей нових або невідомих даних.

Головна цінність Data Mining - це практична спрямованість даної технології, шлях від сирих даних до конкретного знання, від постановки завдання до готового додатку, за підтримки якого можна

Варто сказати, що у нашій роботі ми вирішуватимо задачу кластеризації даних з якою ми ознайомимося більш детально в наступному підрозділі.

1.2. Вибір алгоритмів кластерного аналізу даних

1.2.1 Кластерний аналіз

Кластеризація (або кластерний аналіз) - це задача розбиття множини об'єктів на групи, які називаються кластерами. Усередині кожної групи повинні виявитися «схожі» об'єкти, а об'єкти різних групи повинні бути якомога більш відмінні. Головна відмінність кластеризації від класифікації полягає в тому, що перелік груп чітко не заданий і визначається в процесі роботи алгоритму.

Кластер має наступні математичні характеристики: центр, радіус, середньоквадратичне відхилення, розмір кластера.

Центр кластера - це середнє геометричне місце точок в просторі змінних.

Радіус кластера - максимальна відстань точок від центру кластера. Кластери можуть бути перекриваються. Така ситуація виникає, коли виявляється перекриття кластерів. У цьому випадку неможливо за допомогою математичних процедур однозначно віднести об'єкт до одного з двох кластерів. Такі об'єкти називають спірними.

Спірний об'єкт - це об'єкт, який у міру подібності може бути віднесений до кількох кластерів.

Розмір кластера може бути визначений або по радіусу кластера, або по середньоквадратичному відхиленню об'єктів для цього кластера. Об'єкт відноситься до кластеру, якщо відстань від об'єкта до центру кластера менше радіуса кластера. Якщо ця умова виконується для двох і більше кластерів, об'єкт є спірним. Неоднозначність даного завдання може бути усунена експертом або аналітиком.

Робота кластерного аналізу опирається на два припущення. Перше припущення - що розглядувані ознаки об'єкта, в принципі, допускають бажане розбиття пулу (сукупності) об'єктів на кластери. Друге припущення - правильність вибору масштабу або одиниць вимірювання ознак.

Застосування кластерного аналізу в загальному вигляді зводиться до наступних етапів:

- Відбір вибірки об'єктів для кластеризації.
- Визначення безлічі змінних, за якими будуть оцінюватися об'єкти у вибірці. При необхідності - нормалізація значень змінних.
- Обчислення значень міри схожості між об'єктами.Застосування методу кластерного аналізу для створення груп схожих об'єктів (кластерів).
- Представлення результатів аналізу.

Після отримання та аналізу результатів можливе корегування обраної метрики і методу кластеризації до отримання оптимального результату.

Отже, як же визначати «схожість» об'єктів? Для початку потрібно скласти вектор характеристик для кожного об'єкта - як правило, це набір числових значень, наприклад, зростання-вага людини. Однак існують також алгоритми, що працюють з якісними (т.зв. категорійними) характеристиками.Після того, як ми визначили вектор характеристик, можна провести нормалізацію, щоб всі компоненти давали однаковий внесок при розрахунку «відстані». У процесі нормалізації все значення приводяться до деякого діапазону, наприклад, [1, -1] або [0, 1]. Нарешті, для кожної пари об'єктів вимірюється «відстань» між ними - ступінь схожості. Існує безліч метрик, ось лише основні з них:

Евклідова відстань. Найбільш поширена функція відстані. Являє собою геометричну відстань в багатовимірному просторі:

$$\rho(x, x') = \sqrt{\sum_{i}^{n} (x_i - x_i')^2} \quad (1)$$

Квадрат евклідової відстані. Застосовується для додання більшої ваги більш віддаленим один від одного об'єктів. Це відстань обчислюється таким чином:

$$\rho(x, x') = \sum_{i}^{n} (x_i - x_i')^2 \quad (2)$$

Відстань міських кварталів (Манхетенська відстань). Це відстань є усередненим значенням різниць по координатах. У більшості випадків ця міра відстані приводить до таких же результатів, як і для звичайної відстані Евкліда. Однак для цієї міри вплив окремих великих різниць (викидів) зменшується (тому що вони не зводяться в квадрат). Формула для розрахунку манхетенської відстані:

$$\rho(x, x') = \sum_{i}^{n} | x_i - x_i' | \quad (3)$$

Відстань Чебишева. Ця відстань може виявитися корисною, коли потрібно визначити два об'єкти як «різні», якщо вони різняться якоюсь однією координатою.

Степенева відстань. Застосовується в разі, коли необхідно збільшити або зменшити вагу, що відноситься до розмірності, для якої відповідні об'єкти сильно відрізняються. Степенева відстань обчислюється по такій формулі:

$$\rho(x,x') = \sqrt[r]{\sum_{i}^{n}(x_i - x_i')^p} \quad (4)$$

де r і p - параметри, що визначаються користувачем. Параметр p відповідальний за поступове зважування різниць за окремими координатами, параметр r відповідальний за прогресивне зважування великих відстаней між об'єктами. Якщо обидва параметри - r і p - дорівнюють двом, то це відстань збігається з відстанню Евкліда.

Характерно, що вибір метрики повністю залежить від дослідника, оскільки результати кластеризації можуть істотно відрізнятися при використанні різних методів.

Алгоритми кластерного аналізу. У наступних пункта ми розглянемо найпоширеніші методи кластеризації даних з розрахунку на те, щоб вони працювали належним чином для великих об’ємів просторових даних. Всі методи можна умовно поділити на 3 основні категорії, а саме:

- Ітеративні
- Ієрархічні
- Щільнісні(Density-based)

Тож, розглянемо кожен із них детальніше.

1.2.2 Ітеративні методи

Ітеративні методи виявляють більш високу стійкість по відношенню до шумів і викидів, некоректного вибору метрики, включенню незначущих змінних в набір, який бере участь в кластеризації. Ціною, яку доводиться платити за ці переваги методу, є слово "апріорі". Аналітик повинен заздалегідь визначити кількість кластерів, кількість ітерацій або правило зупинки, а також деякі інші параметри кластеризації. Це особливо складно початківцям фахівцям.

Якщо немає припущень щодо числа кластерів, рекомендують використовувати ієрархічні алгоритми. Однак якщо обсяг вибірки не дозволяє це зробити, можливий шлях - проведення ряду експериментів з різною кількістю кластерів, наприклад, почати розбиття сукупності даних з двох груп і, поступово збільшуючи їх кількість, порівнювати результати. За рахунок такого "варіювання" результатів досягається досить велика гнучкість кластеризації.

Алгоритм k-середніх(k-means).

Найбільш поширений серед ітеративних методів алгоритм k- середніх, також званий швидким кластерним аналізом. На відміну від ієрархічних методів, які не вимагають попередніх припущень щодо числа кластерів, для можливості використання цього методу необхідно мати гіпотезу про найбільш ймовірну кількість кластерів. Алгоритм k-середніх будує k кластерів, розташованих на великих відстанях один від одного. Основний тип задач, які вирішує алгоритм k-середніх, - наявність припущень (гіпотез) щодо числа кластерів, при цьому вони повинні бути різні настільки, наскільки це можливо. Вибір числа k може базуватися на результатах попередніх досліджень, теоретичних міркуваннях або інтуїції.

Метод базується на мінімізації суми квадратів відстаней між кожним спостереженням та центром його кластера, тобто функції.

$$\sum_{i=1}^{k} \sum_{k \in D_i} \|x_i - c_i\|^2 \quad (5)$$

де D_i- набір векторів що належать до і-го кластеру, а c_i - середнє значення цих векторів.

$$c_i = \frac{\sum_{k=1}^{N_i} x_k}{N_i}, x_k \epsilon D_i \quad (6)$$

Основна ідея полягає в тому, що на кожній ітерації заново вираховується центр мас, для кожного кластера, потім вектори розбиваються на нові класи, відповідно до того який з отриманих центрів виявився ближчим за метрикою(приклад алгоритму на рисунку 1.6).

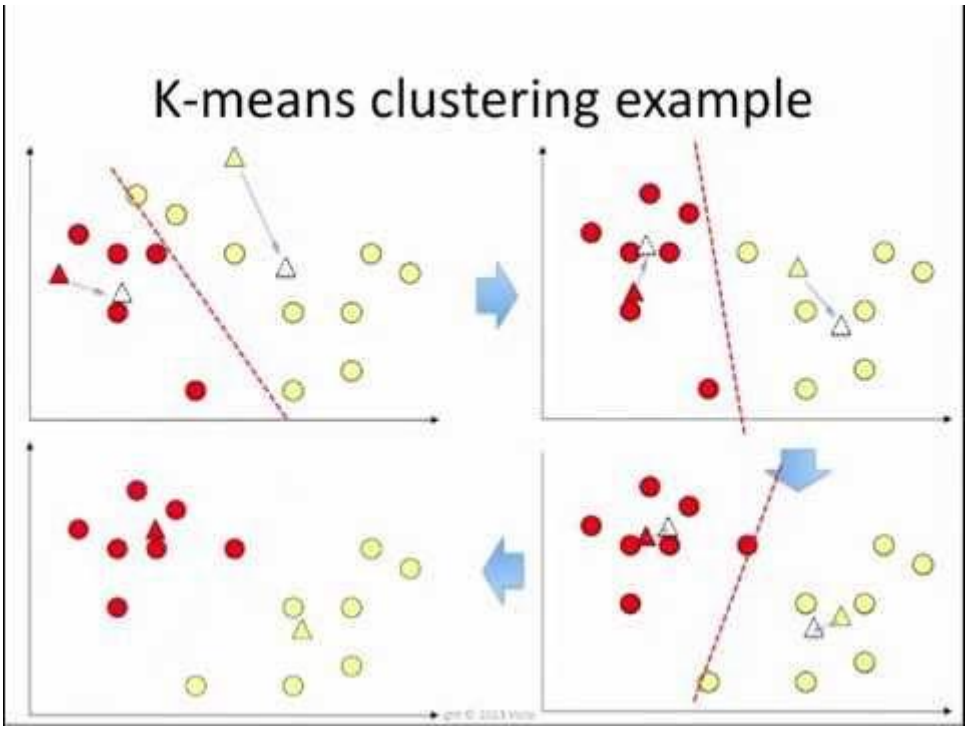

Рисунок 1.6 — Алгоритм K-means[1]

Переваги алгоритму k-середніх:

- простота використання;
- швидкість використання;
- зрозумілість і прозорість алгоритму.

Недоліки алгоритму k-середніх:

- алгоритм занадто чутливий до викидів, які можуть спотворювати середнє. Можливим вирішенням цієї проблеми є використання модифікації алгоритму - алгоритм k-медіани;
- алгоритм може повільно працювати на великих базах даних. Можливим вирішенням цієї проблеми є використання вибірки даних.

Алгоритм EM(Expectation–maximization)

EM-алгоритм — алгоритм, що використовується для знаходження оцінок максимальної схожості параметрів ймовірних моделей, у випадку, коли модель залежить від деяких прихованих змінних. Кожна ітерація алгоритму складається з двох кроків. На E-кроці (expectation) вираховується очікуване значення функції правдоподібності, при цьому приховані змінні розглядаються як спостережувані. На M-кроці (maximization) вираховується оцінка максимальної схожості, таким чином збільшується очікувана схожість, вирахувана на E-кроці. Потім це значення використовується для E-кроку на наступній ітерації. Алгоритм виконується до збіжності.

За певних обставин зручно розглядати EM-алгоритм як два чергуються кроку максимізації. Розглянемо функцію:

$$F(q,\theta) = \mathrm{E}_q[\log L(\theta; x, Z)] + H(q) = -D_{\mathrm{KL}}\left(q \middle\| p_{Z|X}(\cdot|x;\theta)\right) + \log L(\theta; x) \qquad (7)$$

де q — розподіл ймовірностей неспостережуваних змінних Z;

pZ|X(· |x;θ) — умовний розподіл неспостережуваних змінних при фіксованих спостережуваних x і параметрах розподілення ймовірностей неспостережуваних змінних θ; H — ентропія і DKL — відстань Кульбака — Лейблера.

Тоді кроки EM-алгоритму можна показати як:

E(xpectation) крок: Вибираємо q, щоб максимізувати F:

$$q^{(t)} = \arg\max_{q} \; F(q, \theta^{(t)}) \tag{8}$$

M(aximization) крок: Вибираємо θ, щоб максимізувати F:

$$\theta^{(t+1)} = \arg\max_{\theta} \; F(q^{(t)}, \theta) \tag{9}$$

Алгоритм EM простий і легкий в реалізації, не чутливий до ізольованим об'єктах і швидко сходиться при вдалій ініціалізації. Однак він вимагає для ініціалізації вказівки кількості кластерів k, що має на увазі наявність апріорних знань про дані. Крім того, при невдалій ініціалізації збіжність алгоритму може виявитися повільної або може бути отриманий неякісний результат.

Алгоритм PAM(partitioning around Medoids)

PAM є модифікацією алгоритму k-середніх, алгоритмом k- медіани (k-medoids). Алгоритм менш чутливий до шумів і викидів даних, ніж алгоритм k-means, оскільки медіана менше піддається впливам викидів. PAM ефективний для невеликих баз даних, але його не слід використовувати для великих наборів даних.

Більш зрозумілі і прозорі результати кластеризації можуть бути отримані, якщо замість безлічі вихідних змінних використовувати якісь

узагальнені змінні або критерії, які містять в стислому вигляді інформацію про зв'язки між змінними. Тобто виникає задача зниження розмірності даних. Вона може вирішуватися за допомогою різних методів; один з найбільш поширених - факторний аналіз.

Факторний аналіз переслідує дві мети:

- скорочення числа змінних;
- класифікацію змінних - визначення структури взаємозв'язків між змінними.

На першому кроці факторного аналізу здійснюється стандартизація значень змінних. Факторний аналіз спирається на гіпотезу про те, що аналізовані змінні є непрямими проявами порівняно невеликого числа якихось прихованих чинників.

Факторний аналіз - це сукупність методів, орієнтованих на виявлення та аналіз прихованих залежностей між спостережуваними змінними. Приховані залежності також називають латентними.

Один з методів факторного аналізу - метод головних компонент - заснований на припущенні про незалежність факторів один від одного

1.2.3 Ієрархічні методи методи

Загальна ідея методів даної групи полягає в послідовній ієрархічній декомпозиції множини об'єктів. У разі агломеративного методу (від низу до верху) процес декомпозиції начитається з того, що кожен об'єкт являє собою самостійний кластер. Потім на кожній ітерації пари прилеглих кластерів послідовно об'єднуються в загальний кластер. Ітерації тривають до тих пір, поки всі об'єкти не будуть об'єднані в один кластер або поки не виконається деяка умова зупинки. Низхідний метод (зверху вниз) навпаки, грунтується на тому, що на початковому етапі всі об'єкти об'єднані в єдиний кластер. На кожній ітерації він розділяється на більш дрібні до тих пір, поки кожен об'єкт не виявиться в окремому кластері або не буде виконано умову зупинки. Як

умова зупинки можна використовувати граничне число кластерів, яке необхідно отримати, проте зазвичай використовується порогове значення відстані між кластерами.

Основна проблема ієрархічних методів полягає в складності визначення умови зупинки таким чином, щоб виділити «природні» кластери і в той же час не допустити їх розбиття. Ще одна проблема ієрархічних методів кластеризації полягає у виборі точки поділу або злиття кластерів. Цей вибір критичний, оскільки після поділу або злиття кластерів на кожному наступному кроці метод буде оперувати тільки новоствореними кластерами, тому невірний вибір точки злиття або поділу на будь-якому етапі може привести до неякісної кластеризації. Крім того, ієрархічні методи не можуть бути застосовані до великих наборів даних, тому як рішення про поділ або злиття кластерів вимагає аналізу великої кількості об'єктів і кластерів, що веде до великої обчислювальної складності методу.

Алгоритм BIRCH

Завдяки узагальненому вигляду кластерів, швидкість кластеризації збільшується, алгоритм BIRCH (Balanced Iterative Reducing and Clustering using Hierarchies) при цьому володіє великим масштабуванням.

У цьому алгоритмі реалізований двоетапний процес кластеризації.

В ході першого етапу формується попередній набір кластерів. На другому етапі до виявлених кластерів застосовуються інші алгоритми кластеризації - придатні для роботи в оперативній пам'яті.

Аналогія, що описує цей алгоритм. Якщо кожен елемент даних варто уявити собі як намистину, що лежить на поверхні столу, то кластери намистин можна "замінити" тенісними кульками і перейти до більш детального вивчення кластерів тенісних кульок. Число намистин може виявитися досить велике, проте діаметр тенісних кульок можна підібрати таким чином, щоб на другому етапі можна було, застосувавши традиційні алгоритми кластеризації, визначити дійсну складну форму кластерів.

Алгоритм WaveCluster

WaveCluster є алгоритмом кластеризації заснованим на основі хвильових перетворень. На початку роботи алгоритму дані узагальнюються шляхом накладення на простір даних багатовимірної решітки. На подальші кроки алгоритму аналізуються не окремі точки, а узагальнені характеристики точок, які потрапили в одну клітинку решітки. В результаті такого узагальнення необхідна інформація вміщується в оперативній пам'яті. На наступних кроках для визначення кластерів алгоритм застосовує хвильове перетворення до узагальнених даних.

Головні особливості WaveCluster:

- складність реалізації;
- алгоритм може виявляти кластери довільних форм;
- алгоритм не чутливий до шумів;
- алгоритм може бути застосований тільки до даних низької розмірності.

Алгоритм CLARA

Алгоритм CLARA(Clustering LARge Applications) витягує безліч зразків з бази даних. Кластеризація застосовується до кожного із зразків, на виході алгоритму пропонується найкраща кластеризація.

Для великих баз даних цей алгоритм ефективніше, ніж алгоритм PAM. Ефективність алгоритму залежить від обраного в якості зразка набору даних. Хороша кластеризація на обраному наборі може не дати хорошу кластеризацію на всій множині даних.

1.2.4 Щільнісні методи

Дана категорія методів розглядає кластери, як регіони простору даних з високою щільністю об'єктів, які розділені регіонами з низькою щільністю об'єктів. Щільнісні методи часто застосовуються для фільтрації шуму та виявлення кластерів довільної форми.

Алгоритм DBSCAN

Алгоритм DBSCAN - один з перших алгоритмів кластеризації щільнісним методом. В основі цього алгоритму лежить кілька визначень:

- ε-околицею об'єкта називається околиця радіуса ε деякого об'єкту.
- Кореневим об'єктом називається об'єкт, ε-околиця якого містить не менше деякого мінімального числа MinPts об'єктів.
- Об'єкт p безпосередньо щільно-досяжний з об'єкта q якщо p знаходиться в ε-околиці q і q є кореневим об'єктом.
- Об'єкт p щільно-досяжний з об'єкта q при заданих ε і MinPts, якщо існує послідовність об'єктів p1, ..., pn, де p1 = q і pn = p, така що pi + 1 безпосередньо щільно досяжний з pi, $1 \leq i \leq n$.
- Об'єкт p щільно-з'єднаний з об'єктом q при заданих ε і MinPts, якщо існує об'єкт o такий, що p і q щільно-досяжні з o.

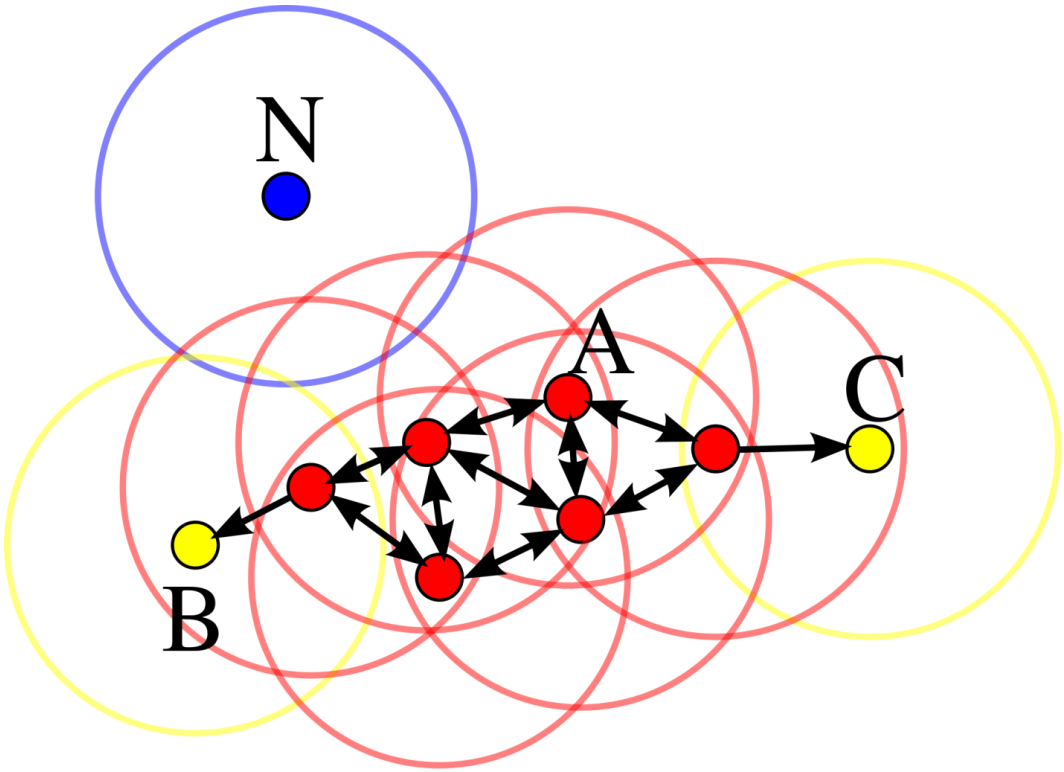

Рисунок 1.7 — Алгоритм DBSCAN[3]

Для пошуку кластерів алгоритм DBSCAN перевіряє ε-околиця кожного об'єкта(Рисунок 1.7). Якщо ε-околиця об'єкта p містить більше точок ніж MinPts, то створюється новий кластер з кореневим об'єктом p. Потім DBSCAN ітеративно збирає об'єкти безпосередньо щільно-досяжні з кореневих об'єктів, які можуть привести до об'єднання кількох щільно-досяжних кластерів. Процес завершується, коли ні до одного кластеру не може бути додано жодного нового об'єкта. Хоча, на відміну від методів розбиття, DBSCAN не вимагає заздалегідь вказувати число одержуваних кластерів, виникне потреба у вказівках значень параметрів ε і MinPts, які безпосередньо впливають на результат кластеризації. Оптимальні значення цих параметрів складно визначити, особливо для багатовимірних просторів даних.

Алгоритм OPTICS

OPTICS (Ordering points to identify the clustering structure)- це алгоритм знаходження щільності на основі кластерів у просторових данихЙого основна ідея схожа на DBSCAN, але він вирішує одну з основних слабостей DBSCAN: проблему визначення значущих кластерів в наборах даних різної щільності. Для цього об'єкти бази даних повинні бути впорядковані(за лінійний час) так, що об'єкти, які просторово близькі, будуть сусідами в упорядкуванні. Крім того, особлива відстань зберігається для кожної точки, яка являє собою щільність, яка повинна бути прийнятна для кластера, щоб мати обидві сусідні точки належали до тієї ж групи. Для кращого розуміння наведемо графік досяжності OPTICS на рисунку 1.8.

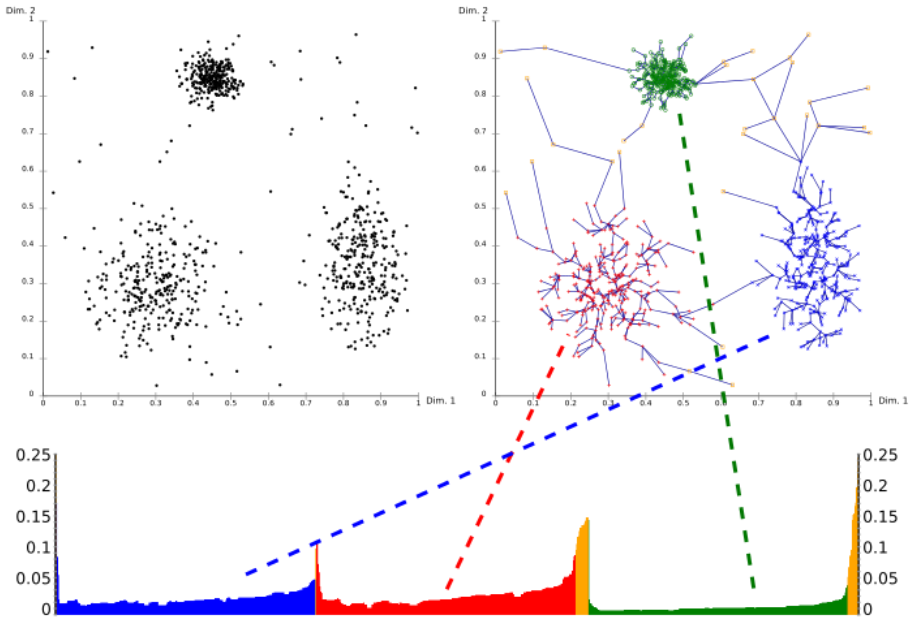

Рисунок 1.8 — Графік досяжності OPTICS[5]

Використовуючи графік досяжності , легко отримати ієрархічну структуру кластерів. Це 2D графік, з упорядкуванням точок, оброблених OPTICS на осі Ox, а відстань досяжності на осі Oy. Так, як точки, що належать до одного кластера мають низьку досяжність для найближчих сусідів, кластери показують ділянки досяжності. Чим глибша ділянка, тим щільніший кластер.

Зображення вище ілюструє цю концепцію. У верхній лівій частині, показано штучний набір даних. Верхня права частина візуалізує сполучне дерево, створене за допомогою OPTICS, і нижня частина показує ділянку досяжності, обчислену OPTICS. Виявлення кластерів з допомогою цього графіку може бути зроблене вручну, вибравши діапазон на осі Ox після візуального огляду, вибравши поріг на осі Oy або за допомогою різних алгоритмів, які намагаються виявити ділянки, використовуючи крутизну, виявлення коліна або локальні максимуми. Кластери, отримані таким чином можуть мати ієрархічну структуру, чого не може бути досягнуто за допомогою DBSCAN.

1.3.Висновок

У даному розділі було детально досліджено предметну область та визначено методи та алгоритми дослідження. Перш за все, варто сказати, що для аналізу даних та пошуку цікавих закономірностей було обрано саме кластерний аналіз, зважаючи на його призначення та поставлені нами завдання. В свою чергу, з цілого різноманіття методів та алгоритмів кластеризації ми практично застосуємо по одному з кожної групи. З ітеративних алгоритмів буде доречно застосувати k-means, з ієрархічних зупинимося на алгоритмі BIRCH, а з щільнісних оберемо DBSCAN. Такий підбір дозволить нам максимально якісно опрацювати дані та провести їх аналіз.

2.ВИБІР ПРОГРАМНИХ ЗАСОБІВ

2.1 Python

Добре відомо, що Python є одним із лідерів в сенсі застосування мов програмування в області data science. Тож, давайте з’ясуємо у чому ж все-таки сильні сторони мови Python і чому її доречно використовувати для інтелектуального аналізу даних.

Python - інтерпретована об'єктно-орієнтована мова програмування високого рівня з динамічною семантикою. Розроблена в 1990 році Гвідо ван Россумом. Структури даних високого рівня разом з динамічною семантикою і динамічним зв'язуванням роблять її привабливою для швидкої розробки програм, а також як засіб поєднання існуючих компонентів.

Об’єктно-орієнтованість.

Python є об'єктно-орієнтованою мовою програмування. Його об'єктна модель підтримує такі поняття, як поліморфізм, перевантаження операторів і множинне спадкування, однак, враховуючи простоту синтаксису і типізації Python, ООП не викликає складнощів в застосуванні. Якщо ці терміни вам незрозумілі, пізніше ви виявите, що вивчати Python набагато легше, ніж інші об'єктно-орієнтовані мови програмування.

Об'єктно-орієнтована природа Python, є могутнім засобом структурування програмного коду багаторазового користування, крім того, робить цю мову ідеальним інструментом підтримки сценаріїв для об'єктно-орієнтованих мов, таких як C ++ і Java. Наприклад, при наявності відповідного сполучного програмного коду, програми на мовою Python можуть використовувати механізм успадкування від класів реалізованих на C ++, Java і C #.

Як би там не було, але ООП не є обов'язковим в Python; ви зможете стати досвідченим програмістом і при цьому не бути фахівцем з ООП.

Як і C ++, Python підтримує обидва стилю програмування – процедурний

і об'єктно-орієнтований. Об'єктно-орієнтовані механізми можуть використовуватися в міру необхідності. Це особливо зручно при вирішенні тактичних завдань, коли відсутня фаза проектування.

Доступність.

Python може використовуватися і розповсюджуватися абсолютно безкоштовно. Як і у випадку з іншими відкритими програмними продуктами, такими як Tel, Perl, Linux і Apache, ви зможете отримати в Інтернеті повні вихідні тексти реалізації Python. Немає ніяких обмежень на його копіювання, вбудовування в свої системи або поширення в складі ваших продуктів. Фактично ви зможете навіть продавати вихідні тексти Python, якщо з'явиться таке бажання. Але «доступний» не означає, що він не підтримується. Навпаки, співтовариство прихильників Python в Інтернеті відповідає на запитання користувачів зі швидкістю, якої могли б позаздрити більшість розробників комерційних продуктів. Крім того, вільне поширення вихідних текстів Python сприяє розширенню команди експертів з реалізації. І хоча надається можливість вивчати або змінювати реалізацію мови програмування не у всіх викликає захоплення, проте, наявність останньої інстанції у вигляді вихідних текстів переконує. Ви вже не залежите від примх комерційного виробника - у вашому розпорядженні знаходиться повний комплект вичерпної документації.

Як уже згадувалося вище, розробка Python ведеться співтовариством, зусилля якого координуються в основному через Інтернеті. До складу співтовариства входить творець Python - Гвідо ван Россум (Guido van Rossum), який отримав офіційне звання Довічного Великодушного Диктатора (Benevolent Dictator for Life, BDFL) Python, плюс тисячі інших розробників.

Зміни в мові вживаються тільки після проходження формальної процедури (Відомої як «програма вдосконалення продукту», або PEP) і ретельно перевіряються формальною системою тестування і самим Довічним Диктатором. Це забезпечує більшу ступінь консерватизму Python щодо змін, в порівнянні з деякими іншими мовами програмування.

Переносимість.

Стандартна реалізація мови Python написана на переносимому ANSI 3, завдяки чому він компілюється і працює практично на всіх основних платформах. Наприклад, програми на мові Python можуть виконуватися на найширшому спектрі пристроїв, починаючи від наладонних комп'ютерів (PDA) і закінчуючи суперкомп'ютерами. Нижче наводиться далеко неповний список операційних систем і пристроїв, де можна використовувати Python:

- Операційні системи Linux і UNIX
- Microsoft Windows і DOS (всі сучасні версії)
- Mac OS (обидва різновиди: OS X і Classic)
- BeOS, OS / 2, VMS і QNX
- Системи реального часу, такі як VxWorks
- Суперкомп'ютери Cray і EOM виробництва компанії IBM
- наладонних комп'ютери, що працюють під управлінням PalmOS, PocketPC
- або Linux
- Стільникові телефони, що працюють під управлінням операційних систем
- Symbian і Windows Mobile
- Ігрові консолі і iPod

Крім самого інтерпретатора мови в складі Python поширюється стандартна бібліотека модулів, яка також реалізована належним шляхом. Крім того, програми на мові Python компілюються в байт-код, який однаково добре працює на будь-яких платформах, де встановлена сумісна версія Python .

Все це означає, що програми на мові Python, що використовують основні можливості мови і стандартні бібліотеки, будуть працювати однаково і в Linux, і в Windows, і в будь-яких інших операційних системах, де встановлений інтерпретатор Python. У більшості реалізацій Python під певні операційні

системи є також підтримка специфічних механізмів цих систем (наприклад, підтримка COM в Windows), але ядро мови Python і бібліотеки працюють абсолютно однаково в будь-якій системі. як уже говорилося вище, Python включає в себе кошти створення графічного інтерфейсу Tk GUI під назвою tkinter (Tkinter в Python 2.6), що дозволяє програмами на мові Python створювати графічний інтерфейс, сумісний з усіма основними графічними платформами без індивідуального програмного налаштування.

Потужність.

З точки зору функціональних можливостей Python можна назвати гібридом. Його інструментальні засоби вкладаються в діапазон між традиційними мовами сценаріїв (такими як Tel, Scheme і Perl) і мовами розробки програмних систем (такими як C, C ++ і Java). Python забезпечує простоту і невимушеність мови сценаріїв і міць, яку зазвичай можна знайти в компільованих мовами. Перевищуючи можливості інших мов сценаріїв, така комбінація робить Python зручним засобом розробки великомасштабних проектів. Для попереднього ознайомлення нижче наводиться список основних можливостей, які є в арсеналі Python.

Динамічна типізація. Python сам стежить за типами об'єктів, що використовуються в програмі, завдяки чому не потрібно писати довгі і складні оголошення в програмному коді. Насправді, в мові Python взагалі відсутні поняття типу і необхідність оголошення змінних. Так як програмний код на мові Python не обмежений рамками типів даних, він автоматично може обробляти цілий діапазон об'єктів.

Автоматичне управління пам'яттю. Python автоматично розподіляє пам'ять під об'єкти і звільняє її, коли об'єкти стають непотрібними. Більшість об'єктів можуть збільшувати і зменшувати обсяг пам’яті яку вони обіймають по мірі необхідності. Як відомо, Python сам виконує всі низькорівневі операції з пам'яттю, тому вам не доведеться турбуватися про це.

Модульне програмування. Для створення великих систем Python надає такі можливості, як модулі, класи і виключення. Вони дозволяють розбити

систему на складові, застосовувати ООП для створення програмного коду багаторазового користування і елегантно обробляти події та помилки, що виникають.

Вбудовані типи об'єктів. Python надає найбільш типові структури даних, такі як списки, словники і рядки, у вигляді особливостей, властивих самій мові програмування. Як ви побачите пізніше, ці типи відрізняються високою гнучкістю і зручністю. Наприклад, вбудовані об'єкти можуть розширюватися і стискатися в міру необхідності, можуть комбінуватися один з одним для представлення даних зі складною структурою і багато іншого.

Вбудовані інструменти. Для роботи з усіма цими типами об'єктів в складі Python є потужні і стандартні засоби, включаючи такі операції, як конкатенація (об'єднання колекцій), отримання зрізів (витяг частини колекції), сортування, відображення і багато іншого.

Бібліотеки утиліт. Для виконання більш вузьких завдань до складу Python також входить велика колекція бібліотечних інструментів, які підтримують практично все, що тільки може знадобитися, - від пошуку з використанням регулярних виразів до роботи в мережі. Бібліотечні інструменти мови Python - це те місце, де виконується велика частина операцій.

Утиліти сторонніх розробників. Python - це відкритий програмний продукт і тому розробники можуть створювати свої попередньо скомпільовані інструменти підтримки завдань. У Мережі ви знайдете вільну реалізацію підтримки COM, засобів для роботи з зображеннями, розподілених об'єктів CORBA, XML, механізмів доступу до баз даних і багато іншого. Незважаючи на широкі можливості, Python має надзвичайно простий синтаксис і архітектуру. В результаті ми маємо потужний інструмент програмування, що володіє простотою і зручністю, властивими мовам сценаріїв.

Зручність.

Щоб запустити програму на мові Python, досить просто ввести її ім'я. Не потрібно виконувати проміжну компіляцію і зв'язування, як це робиться в мовах програмування, подібних C або C ++. Інтерпретатор Python негайно

виконує програму, що дозволяє програмувати в інтерактивному режимі і отримувати результати відразу ж після внесення змін - у більшості випадків ви зможете спостерігати ефект зміни програми з тією швидкістю, з якою ви вводите зміни з клавіатури.

Безумовно, швидкість розробки - це лише один з аспектів зручності Python. Крім того, він забезпечує надзвичайно простий синтаксис і набір потужних вбудованих інструментів. Тому деякі навіть називають Python «Виконуваним псевдокодом». Оскільки велика частина складнощів ліквідується іншими інструментами, програми на мові Python простіше, менше і гнучкіше еквівалентних їм програм, написаних на таких мовах, як C, C ++ і Java!

Легкість у вивченні.

Це дуже важливий аспект даної мови: в порівнянні з іншими мовами програмування, базова мова Python дуже легко запам'ятовується. Ви насправді ви зможете писати на мові Python більш-менш значущі програми вже через кілька днів. Це чудова новина для розробників, які прагнуть вивчити мову для застосування її в своїй роботі, а також для кінцевих користувачів, які застосовують Python для налаштування або управління програмним продуктом.

Сьогодні багато систем виходять з того, що кінцеві користувачі можуть швидко вивчити Python в достатній мірі, щоб самостійно створити свій власний програмний код. І хоча в Python є складні інструменти програмування, основа мови як і раніше залишається простою для вивчення як початківцями, так і досвідченими програмістами.

Розробники мови Python є прихильниками певної філософії програмування, яку називають «The Zen of Python» («Дзен Пайтона») . Її текст можна отримати в інтерпретаторі Python за допомогою команди import this (один раз за сесію). Автором цієї філософії вважається Тім Пейтерс. Текст філософії:

- Гарне краще, ніж потворне.

- Явна краще, ніж неявне
- Просте краще, ніж складне.
- Складне краще, ніж заплутане.
- Плоске краще, ніж вкладене.
- Розріджене краще, ніж щільне.
- Легкість читання має значення.
- Особливі випадки не настільки особливі, щоб порушувати правила.
- При цьому практичність важливіше бездоганності.
- Помилки ніколи не повинні замовчуватися.
- Якщо не замовчуються явно.
- Зустрівши двозначність, відкинь спокуса вгадати.
- Повинен існувати один - і, бажано, тільки один - очевидний спосіб 36 зробити це.
- Хоча спочатку він може бути і не очевидним, якщо ви не голландець .
- Зараз краще, ніж ніколи.
- Хоча ніколи, як правило, краще, ніж просто зараз.
- Якщо реалізацію важко пояснити - ідея погана.
- Якщо реалізацію легко пояснити - ідея, можливо, хороша.
- Простір імен - чудова річ! Будемо робити їх побільше!

2.2 Numpy

Numpy — розширення мови Python, що додає підтримку великих багатовимірних масивів і матриць, разом з великою бібліотекою високорівневих математичних функцій для операцій з цими масивами. Попередник Numpy, Numeric, був спочатку створений Jim Hugunin. Numpy — відкрите програмне забезпечення і має багато розробників. Оскільки Python — інтерпретована мова, математичні алгоритми, часто працюють в ньому набагато

повільніше ніж у компільованих мовах, таких як C або навіть Java. NumPy намагається вирішити цю проблему для великої кількості обчислювальних алгоритмів забезпечуючи підтримку багатовимірних масивів і безліч функцій і операторів для роботи з ними. Таким чином будь-який алгоритм який може бути виражений в основному як послідовність операцій над масивами і матрицями працює також швидко як еквівалентний код написаний на C.

NumPy можна розглядати як гарну вільну альтернативу MATLAB, оскільки мова програмування MATLAB зовні нагадує NumPy: обидві вони інтерпретовані, і обидві дозволяють користувачам писати швидкі програми поки більшість операцій проводяться над масивами або матрицями, а не над скалярами. Перевага MATLAB у великій кількості доступних додаткових тулбоксів, включаючи такі як пакет Simulink.

2.3 SciPy

SciPy — відкрита бібліотека високоякісних наукових інструментів для мови програмування Python. SciPy містить модулі для оптимізації, інтегрування, спеціальних функцій, обробки сигналів, обробки зображень, генетичних алгоритмів, розв'язування звичайних диференціальних рівнянь та інших задач, які розв'язуються в науці і при інженерній розробці. Бібліотека розробляється для тієї ж аудиторії, що iMATLAB та Scilab. Для візуалізації при використанні SciPy часто застосовують бібліотеку Matplotlib, яка є аналогом засобів виводу графіки MATLAB. В даний час SciPy поширюється під ліцензією BSD і його розробники спонсоруються Enthought. Основною структурою даних в SciPy є багатовимірний масив, реалізований модулем NumPy .

До основних пакетів даної бібліотеки відносять:

- **constants**: Фізичні константи і коефіцієнти перерахунку **cluster:** Векторне квантування

- **fftpack:** Дискретні алгоритми перетворення Фур'є
- **integrate:** Інструменти для інтегрування
- **interpolate:** Інструменти для інтерполяції
- **io:** Введення-виведення даних
- **lib:** Враппери Python для зовнішніх бібліотек
- **linalg:** Лінійна алгебра
- **misc:** Різні утиліти
- **ndimage:** функції для обробки зображень
- **optimize:** Засоби оптимізації
- **sandbox:** Експериментальний код
- **signal:** Обробка сигналів
- **sparse:** Підтримка розріджених матриць
- **spatial:** К-вимірні дерева, метод найближчих k-сусідів, метрики.
- **special:** Спеціальні функції
- **stats:** Статистичні функції
- **weave:** Дозволяє включати код C/C++ всередину коду Python

Додаткова функціональність:

- **Графіка.** На даний момент рекомендованим пакетом для рисування двомірної графіки є Matplotlib, однак існує велика кількість інших, наприклад, HippoDraw, Chaco, і Biggles. Також популярними є Python Imaging Library і MayaVi (для 3D візуалізації).
- **Оптимізація.** Хоча SciPy має свій пакет для оптимізації, OpenOpt має доступ до більшої кількості оптимізаційних пакетів і розв'язувачів.
- **Розширений аналіз даних.** За допомогою RPy, SciPy забезпечує інтерфейс до статистичному пакету R, призначеному для складного аналізу даних.

- **База даних.** SciPy може взаємодіяти з PyTables, ієрархічною базою даних, розробленою для ефективного керування великими об'ємами даних, що зберігаються у файлах формату HDF5.
- **Інтерактивна оболонка.** IPython це інтерактивне середовище, яке забезпечує зневадження і створення коду в стилі, близькому до MATLAB.
- **Символьна математика.** Існує декілька бібліотек для Python, таких як PyDSTool, Symbolic і SymPy, які дозволяють працювати із символьною математикою.

2.4 Pandas

Pandas — вільна бібліотека Python, створена для проведення зручних маніпуляцій з даними та аналізу. Зокрема, дана бібліотека пропонує можливості роботи з числовими таблицями та часовими рядами. Назву бібліотека отримала від терміну зеконометрії «panel data» — багатовимірного струкрутованого набору данних.

Даний пакет робить Python потужним інструментом для аналізу даних. Пакет дає можливість будувати зведені таблиці, виконувати угруповання, надає зручний доступ до табличних даних, а при наявності пакету matplotlib дає можливість малювати графіки на отриманих наборах даних.

Wes McKinney почав працювати над Pandas в 2008, коли у AQR Capital Management виникла потреба у високопродуктивному, гнучкому інструменту, для здійснення кількісного аналізу фінансових даних. Перед тим, як покинути AQR він зміг переконати керівництво дозволити йому віддати власні напрацювання у відкритий доступ.

Інший співробітник AQR Chang She приєднався в 2012 як другий основний контриб'ютор бібліотеки. Приблизно в цей час бібліотека стала популярною в Python спільноті, та збільшилась кількість контриб'юторів

проекту. На даний момент проект вважається одним з найбільш важливих і активних бібліотек аналізу даних на Python.

До можливостей бібліотеки відносять:

- DataFrame — тип данних з інтегрованою індексацією, створений для маніпуляцій над данними
- Інструменти для зчитування та запису данних таких розширень, як: CSV, Excel, JSON, SAS, FWF.
- Обробка відсутніх даних
- Підтримка операцій типу Group by
- Вставка та видалення строк/стовпців
- Злиття та приєднання датасетів
- Ієрархічна індексація осей для роботи з високорозмірними даними в низькорозмірних структурах даних
- Функціональність для роботи з часовими рядами

2.5 Matplotlib

Matplotlib — бібліотека на мові програмування Python для візуалізації даних двовимірною 2D графікою (3D графіка також підтримується). Отримувані зображення можуть бути використані в якості ілюстрацій в публікаціях.

Matplotlib є гнучким, легко конфігурованим пакетом, який разом з NumPy, SciPy і IPython надає можливості, подібні до MATLAB. В даний час пакет працює з декількома графічними бібліотеками, включаючи wxWindows iPyGTK.

Пакет підтримує багато видів графіків і діаграм:

- Графіки (line plot)
- Діаграми розсіювання (scatter plot)

- Стовпчасті діаграми (bar chart) і гістограми (histogram)
- Секторні діаграми (pie chart)
- Діаграми «Стовбур-листя» (stem plot)
- Контурні графіки (contour plot)
- Поля градієнтів (quiver)
- Спектральні діаграми (spectrogram)

Користувач може вказати осі координат, сітку, додати підписи і пояснення, використовувати логарифмічну шкалу або полярні координати.

Нескладні тривимірні графіки можна будувати з допомогою набору інструментів (toolkit) mplot3d. Існують і інші набори інструментів: для картографії, для роботи з Excel, утиліти для GTK та інші.

2.6 Scikit-learn

Бібліотека scikit-learn надає реалізацію цілого ряду алгоритмів для навчання з учителем (Supervised Learning) і навчання без вчителя (Unsupervised Learning) через інтерфейс для мови програмування Python. Дана бібліотека поширюється під ліцензією "Simplified BSD License" і має дистрибутиви для безлічі різних версій Linux, заохочуючи тим самим академічне і комерційне використання scikit-learn.

Scikit-learn побудована поверх SciPy (Scientific Python), який повинен бути встановлений перед використанням scikit-learn. Доповнення і модулі SciPy умовно іменуються SciKits. Саме тому модуль, який надає алгоритми навчання, називається scikit-learn. Одна з основних концепцій бібліотеки scikit-learn - бібліотека з рівнем надійності і підтримки, який необхідний для продакшн-систем, а це означає, що велика увага приділяється питанням зручності використання, якості коду, документації та оптимізації швидкості роботи бібліотеки.

Незважаючи на те що весь інтерфейс бібліотеки представлений на Python, але використання бібліотек, написаних на C у внутрішній реалізації деяких частин scikit-learn, дозволяє значно підвищити швидкість роботи, наприклад, використання NumPy для роботи з масивами і для операцій з матрицями, використання LAPACK і LibSVM і акуратне використання Cython.

Бібліотека scikit-learn орієнтована в першу чергу на моделювання даних, а не на завантаження, маніпуляцію і узагальнення даних. Для таких цілей, як раз-таки, краще використовувати NumPy і Pandas. Ось кілька популярних функціональних областей, в яких scikit-learn допомагає вирішувати поставлені завдання:

- **Кластеризація (Clustering):** для угрупування нерозмічених даних, наприклад, метод k-середніх (k-means)
- **Перехресна перевірка (Cross Validation):** для оцінки ефективності роботи моделі на незалежних даних
- **Набори даних (Datasets):** для тестових наборів даних і для генерації наборів даних з певними властивостями для дослідження поведінкових властивостей моделі
- **Скорочення розмірності (Dimensionality Reduction):** для зменшення кількості атрибутів для візуалізації та відбору ознак (Feature Selection), наприклад, метод головних компонент (Principal Component Analysis)
- **Алгоритмічні композиції (Ensemble Methods):** для комбінування прогнозів декількох моделей
- **Витяг ознак (Feature Extraction):** визначення атрибутів в зображеннях і текстових даних
- **Відбір ознак (Feature Selection):** для виявлення значущих атрибутів на основі яких буде побудована модель
- **Оптимізація параметрів алгоритму (Parameter Tuning):** для отримання максимально ефективної віддачі від моделі

- **Множинне навчання (Manifold Learning):** для нелінійного скорочення розмірності даних
- **Алгоритми навчання з учителем (Supervised Models):** величезний набір методів не обмежується узагальненими лінійними моделями (Generalized Linear Models), дискримінантним аналізом (Discriminate Analysis), наївним Байєсовим класифікатором (Naive Bayes), нейронними мережами (Neural Networks), методом опорних векторів (Support Vector Machines) і деревами прийняття рішень (Decision Trees).

2.7 Orange

Orange є безкоштовним програмним забезпеченням для машинного навчання та інтелектуального аналізу даних (написане на Python). Дане програмне забезпечення має чудову фрон-енд частину для візуалізації результатів аналізу, а також може бути використане в якості бібліотеки Python. Програма підтримується і розробляється в лабораторії біоінформатики факультету обчислювальної техніки та інформатики в Університеті Любляни. Компоненти, з яких складається Orange , називаються відметами і кожен з них надає різні можливості, від візуалізації даних до емпіричної оцінки алгоритмів та прогностичного моделювання.

Візуальне програмування здійснюється через інтерфейс, в якому робочі процеси створюються шляхом об'єднання стандартних або призначених для користувача наборів віджетів, в той час, як досвідчені користувачі можуть використовувати Orange в якості бібліотеки Python для безпосереднього маніпулювання даними. Orange складається з інтерфейсу – робочого полотна, на який користувач поміщає віджети і створює робочий процес аналізу даних. Віджети пропонують основні функції, такі як читання даних, побудова таблиць, вибір функцій, навчання, порівняння алгоритмів, візуалізація елементів даних і т.д. Користувач може в інтерактивному режимі займатися процесом

інтелектуального аналізу даних. Робочий інтерфейс Orange наведено на рисунку 2.1.

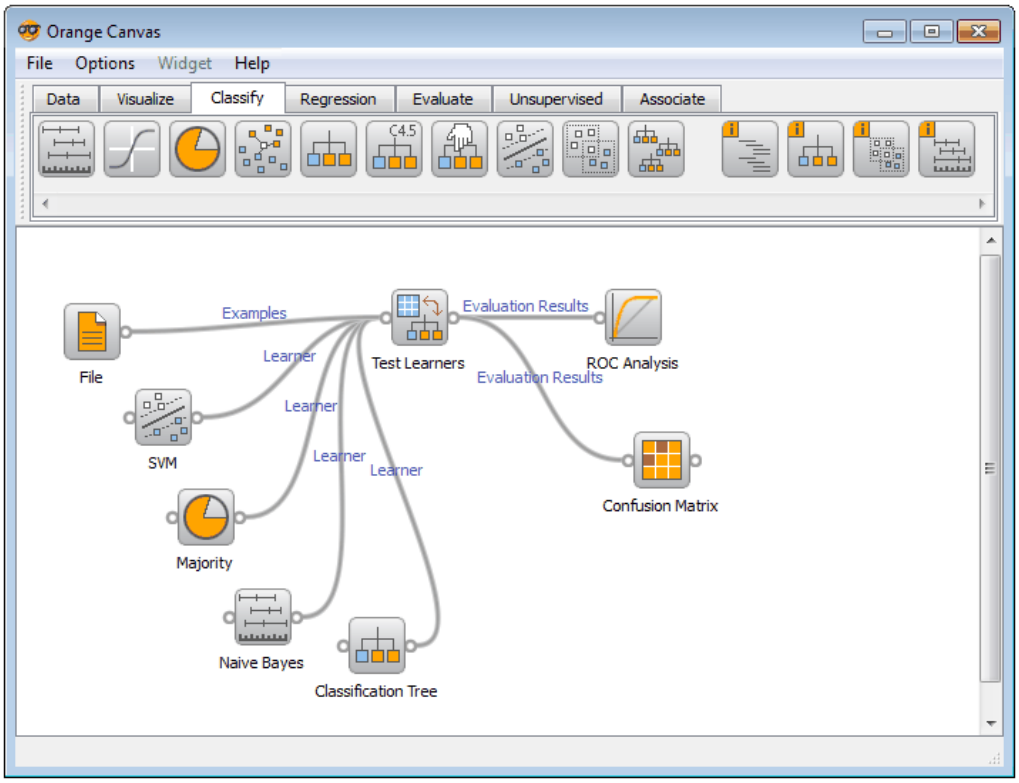

Рисунок 2.1— Робочий інтерфейс Orange[6]

2.8 Weka

Weka (Waikato Environment for Knowledge Analysis) — вільне програмне забезпечення для аналізу даних та машинного навчання, написане на Java в університеті Уайкато, розповсюджується за ліцензією GNU GPL.

Weka — це набір засобів візуалізації та алгоритмів для аналізу даних і вирішення задач прогнозування, разом з графічною оболонкою для доступу до них. Основним інтерфейсом користувача є Explorer, хоча ті ж функціональні можливості підтримуються з командного рядка та інтерфейсу Knowledge Flow. Для систематичного порівняння різних алгоритмів машинного навчання використовується інтерфейс Experimenter. Він дозволяє порівнювати результати не лише різних алгоритмів на одному наборі даних, а й одного алгоритму на різних наборах даних.

Інтерфейс Explorer містить наступні панелі:

- Панель попереднього опрацювання уможливлює імпорт даних з бази даних, текстових файлів у форматі CSV, а також попереднє опрацювання цих даних за допомогою різноманітних алгоритмів (фільтрів). Ці фільтри використовуються для трансформування даних, а також для видалення певних атрибутів.
- Панель класифікації надає можливість застосувати алгоритми класифікації та регресійного аналізу до обраного набору даних, візуалізувати та оцінити результати, відобразити ROC криві тощо.
- Панель асоціації надає доступ до методів, які дозволяють оцінити взаємозв'язки між атрибутами.
- Панель кластеризації містить різноманітні методи кластеризації, наприклад метод кластеризації методом k-середніх, EM-алгоритм тощо.
- Панель вибору атрибутів дозволяє ідентифікувати атрибути, які найбільш впливають на якість прогнозування.
- Панель візуалізації відображає точкові діаграми.

2.9 Висновок

У даному розділі було проведено опис інструментарію для дослідження, який був безпосередньо використаний при виконанні дипломної роботи. Головним чином мовою програмування Python було реалізовано весь цикл дослідження, в той час, як бібліотеки Numpy, SciPy та Pandas були дуже корисними у підготовці даних. В свою чергу scikit-learn ми застосували для власне аналізу даних. Зручну ж візуалізацію нам забезпечили бібліотека matplotlib та програма Orange .

3.ПОСТАНОВКА ЗАВДАННЯ, РЕАЛІЗАЦІЯ ТА АНАЛІЗ РЕЗУЛЬТАТІВ

3.1.Постановка завдання

В якості просторової бази онкохворих пацієнтів використаємо SEER(The Surveillance, Epidemiology, and End Results) Database , яка визначається значною кількістю записів, що являється ключовим фактором для успішного аналізу та подальшого прогнозування а його основі, а саме, у ній мітяться дані про онкохворих жителів США, віком від 15 до 65 років, що збиралися протягом понад 40 років. Характерним є той факт, що дані були представлені у незручному форматі й потребували попередньої обробки. Вони були поділені на дві групи просторові і непросторові. Набір просторових даних містить координати розташування, зображення дистанційного зондування та інша географічна інформація. До основних непросторових даних відносились :

- стать
- вік
- сімейний стан
- висота
- вага
- расова приналежність
- соціальне становище

Досліджувались основні різновиди захворювання такі як: рак легень, рак нирок, рак горла і т.д. Основний інтерес у вирішенні даної задачі полягає у безпосередньому застосування методів ІАПД у медицині. Таким чином, основний акцент буде зроблено на географічну прив’язку пацієнтів, їх розташування та закономірності розподілу по території. Дана особливість і буде головним чином відрізняти наше дослідження від стандартного процесу інтелектуального аналізу даних.

3.2 Інтелектуальний аналіз даних засобами Weka

В даному пункті дослідимо наші дані за допомогою дуже зручного засобу для аналізу даних Weka. У цьому випадку дещо обмежимо нашу вибірку даних, тобто дослідимо закономірності та проведемо кластеризацію набору даних по штату Нью-Йорк. База даних, що досліджувалася містить 15241 запис про пацієнтів та такі атрибути, як geo-id, географічні координати та інші. Для дослідження оберемо три методи, а саме: ітеративний, ієрархічний та щільнісний.

3.2.1 Кластеризація методом k-means.

В результаті кластеризації, всі дані були поділені на два кластери, перший з яких містив 70% даних, інший 30%. Для отримання кінцевого результату нам знадобилось 14 ітерацій. Сума квадратів помилок сягнула 31905. З вихідними даними можна знайомитись на рисунку 3.1.

```
Test mode:    evaluate on training data

=== Model and evaluation on training set ===


kMeans
======

Number of iterations: 14
Within cluster sum of squared errors: 31905.13571638936
Missing values globally replaced with mean/mode

Cluster centroids:
                                            Cluster#
Attribute                 Full Data                 0                 1
                            (15241)           (10740)           (4501)
=============================================================================
GEOID10             360605533612.4042 360664459792.7392 360464927707.7612
DISPLAY_ID               007\012101\1      007\012101\1      007\012102\3
DOHREGION              36061DOH0015      36061DOH0015      36045DOH0005
POP_TOT                  1271.4456         1308.4019         1183.2628
INTPTLAT                   41.5406           40.9414           42.9704
INTPTLONG                 -74.7028          -73.8166          -76.8172


Clustered Instances

0      10740 ( 70%)
1       4501 ( 30%)
```

Рисунок 3.1 — Вихідні дані кластеризації методом k-means

В результаті роботи ми отримали два графіки. Перший це закономірність поширення пацієнтів по штату Нью-Йорк. Другий графік залежність geo-id від географічної довготи. Дані графіки зображені на рисунках 3.2 та 3.3 відповідно.

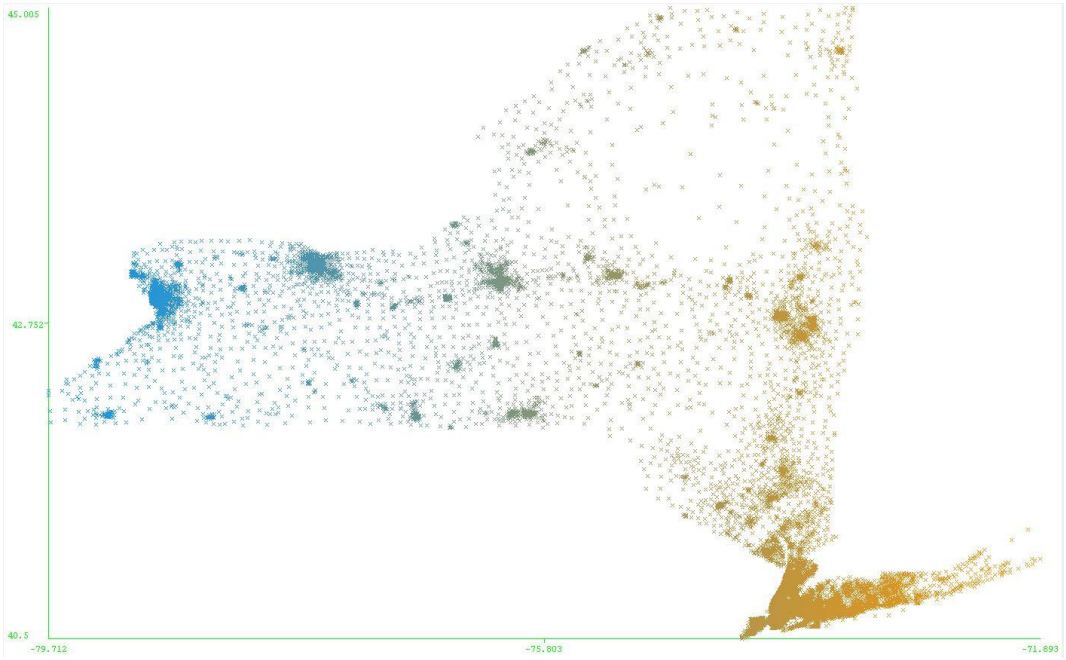

Рисунок 3.2 — Графік поширення пацієнтів по штату

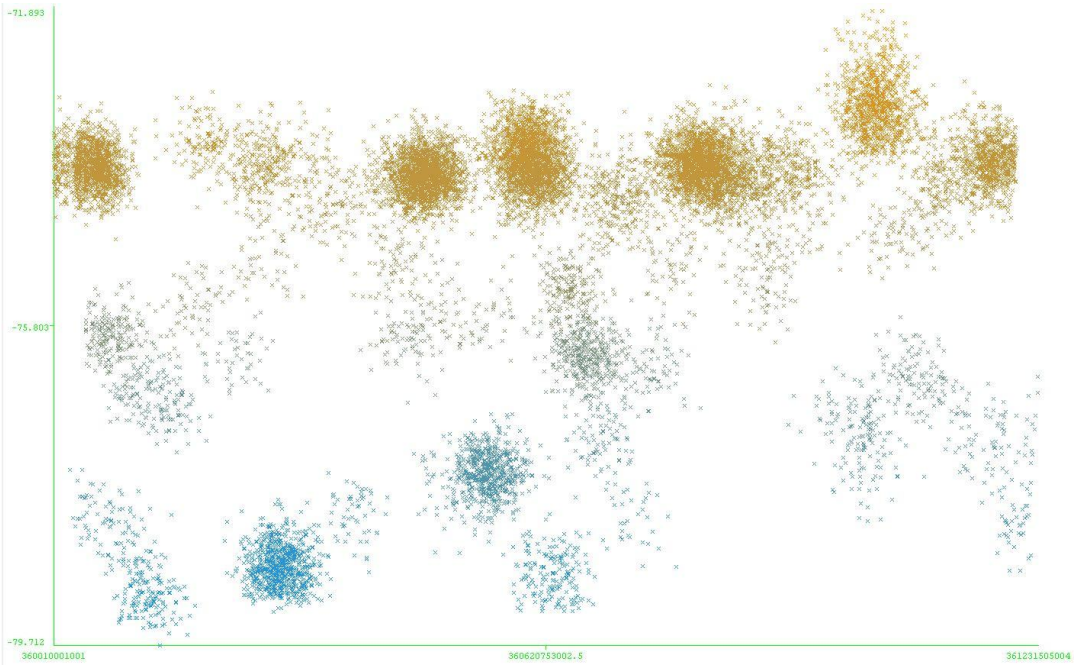

Рисунок 3.3 — Графік залежності geo-id від географічної довготи

3.2.2 Кластеризація методом BIRCH

У випадку ієрархічної кластаризації наша вибірка була поділена на два кластери,як і в попередньому випадку. Як результат, ми отримали два графіки. Перший це закономірність поширення пацієнтів по штату Нью-Йорк. Другий графік залежність geo-id від географічної широти. Дані графіки можна глянути на рисунках 3.4 та 3.5 відповідно.

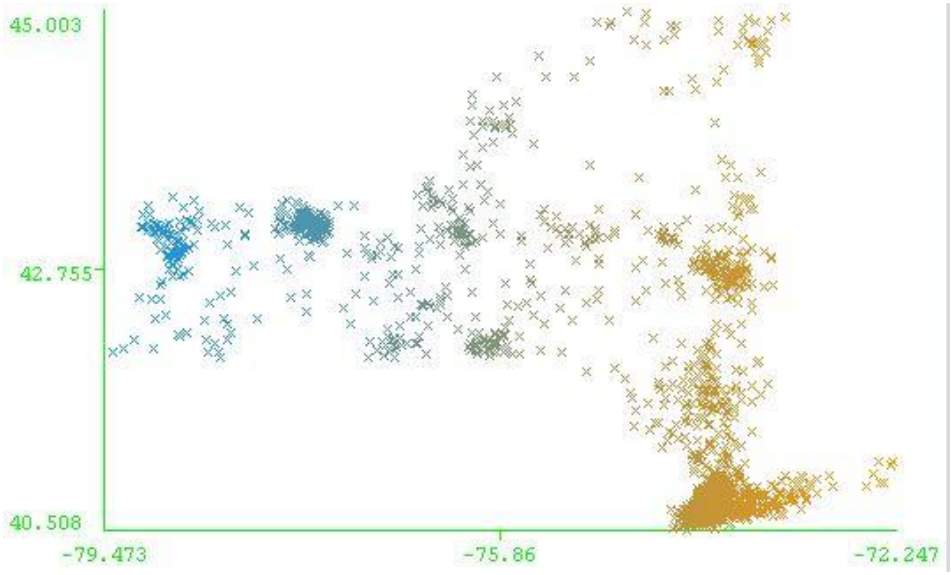

Рисунок 3.4 — Графік поширення пацієнтів по штату

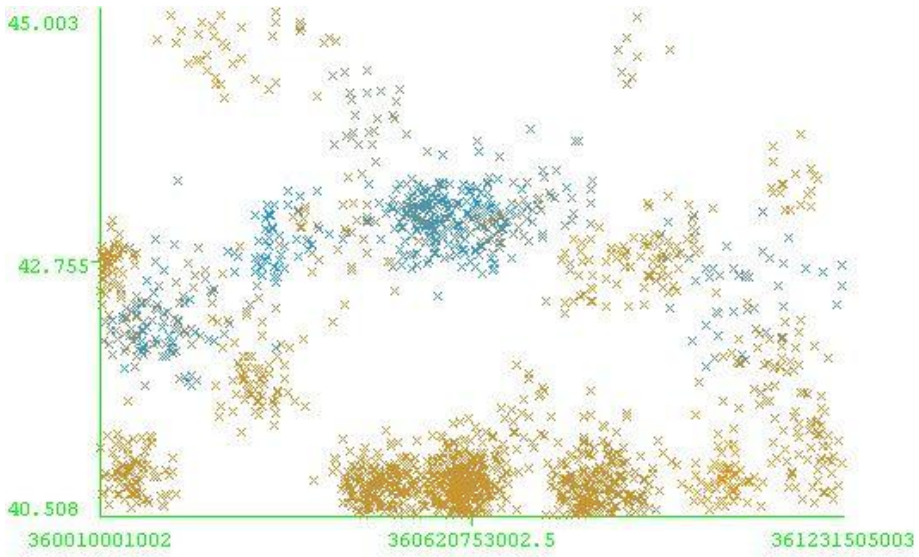

Рисунок 3.5 — Графік залежності geo-id від географічної довготи

Крім того, було побудовано дендограму(рисунок 3.6), що дає можливість краще оцінити результати роботи.

Під дендрограмою зазвичай розуміється дерево, тобто граф без циклів побудований за матриці заходів близькості. Дендрограма дозволяє зобразити взаємні зв'язки між об'єктами з заданого переліку. Для створення дендрограми потрібно матриця подібності (чи відмінності), яка визначає рівень спорідненості між парами об'єктів. Частіше використовуються агломеративні методи.

Далі необхідно вибрати метод побудови дендрограми, який визначає метод візуалізації матриці подібності (чи відмінності) після об'єднання (або поділу) чергових двох об'єктів в кластер.

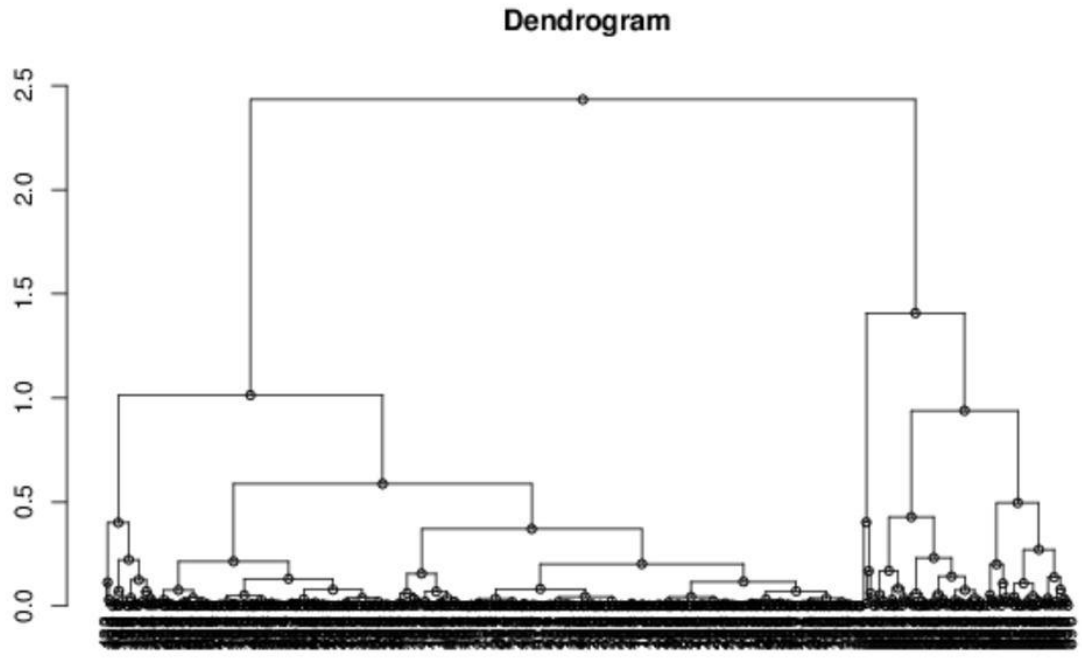

Рисунок 3.6 — Дендограма

3.2.3 Кластеризація методом DBSCAN

Для щільнісного методу аналізу ми обрали 6 атрибутів. Як і у попередніх випадках, результатом роботи алгоритму стало розбиття на два кластери з

відсотковим відношенням: перший 67% та другий 33%. Судячи зі звіту, початкова ймовірність для першого кластера була на рівні 0.7047, для другого ж 0.2953. Розгорнутий результат аналізу можна глянути на рисунку 3.7, який зображений нижче.

```
Initial starting points (random):

Cluster 0: 360470894004,'047\\089400\\4',36047DOH0240,801,40.659298,-73.915158
Cluster 1: 360930216004,'093\\021600\\4',3.60930216004E11,1207,42.792032,-73.941345

Missing values globally replaced with mean/mode

Final cluster centroids:
                                          Cluster#
Attribute                Full Data              0              1
                         (15241.0)        (10740.0)       (4501.0)
=============================================================================
GEOID10             360605533612.4042 360664459792.7392 360464927707.7612
DISPLAY_ID               007\012101\1      007\012101\1      007\012102\3
DOHREGION                36061DOH0015      36061DOH0015      36045DOH0005
POP_TOT                     1271.4456         1308.4019         1183.2628
INTPTLAT                      41.5406           40.9414           42.9704
INTPTLONG                    -74.7028          -73.8166          -76.8172

Fitted estimators (with ML estimates of variance):

Cluster: 0 Prior probability: 0.7047

Attribute: GEOID10
Normal Distribution. Mean = 360664459792.7392 StdDev = 310576897.284
Attribute: DISPLAY_ID
Discrete Estimator. Counts =  2 1 1 1 1 1 1 1 2 2 2 2 2 2 2 1 1 1 1 1 1 1 2 2
Attribute: DOHREGION
Discrete Estimator. Counts =  2 1 1 1 1 1 1 1 2 2 2 2 2 2 2 1 1 1 1 1 1 1 2 3
Attribute: POP_TOT
Normal Distribution. Mean = 1308.4019 StdDev = 566.2982
Attribute: INTPTLAT
Normal Distribution. Mean = 40.9414 StdDev = 0.5192
Attribute: INTPTLONG
Normal Distribution. Mean = -73.8166 StdDev = 0.3322

Cluster: 1 Prior probability: 0.2953

Attribute: GEOID10
Normal Distribution. Mean = 360464927707.7612 StdDev = 296968442.6271
Attribute: DISPLAY_ID
Discrete Estimator. Counts =  1 2 2 2 2 2 2 2 1 1 1 1 1 1 1 2 2 2 2 2 2 2 1 1
Attribute: DOHREGION
Discrete Estimator. Counts =  1 2 2 3 3 2 2 3 1 1 1 1 1 1 1 2 2 2 2 2 2 2 1 1
Attribute: POP_TOT
Normal Distribution. Mean = 1183.2628 StdDev = 567.1749
Attribute: INTPTLAT
Normal Distribution. Mean = 42.9704 StdDev = 0.5734
Attribute: INTPTLONG
Normal Distribution. Mean = -76.8172 StdDev = 1.6688


Time taken to build model (full training data) : 0.58 seconds

=== Model and evaluation on training set ===

Clustered Instances

0      10266 ( 67%)
1       4975 ( 33%)
```

Рисунок 3.7 — Статистика кластеризації методом DBSCAN

У представленні візуальних результатів кластеризації зупинимося на варіантах графіків залежності geo-id від регіонів проживання пацієнтів(рисунок 3.8) та графіку закономірності поширення пацієнтів по штату(рисунок 3.9). У першому випадку буде змога оцінити нові результати, у той час другий варіант дозволить порівняти останній графік з графіками одержаними у попередніх двох дослідженнях.

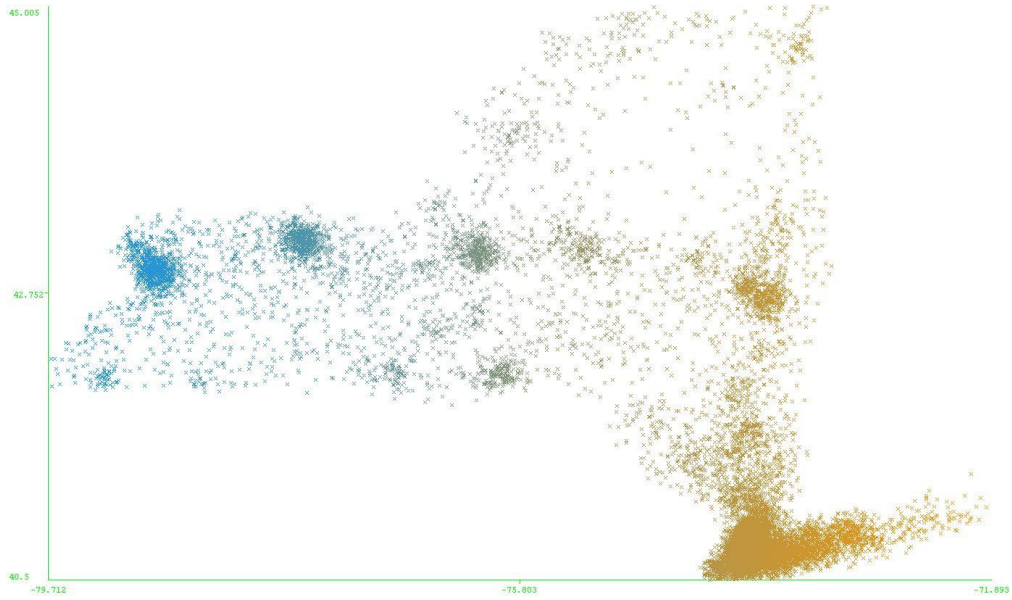

Рисунок 3.8 — Графік поширення пацієнтів по штату

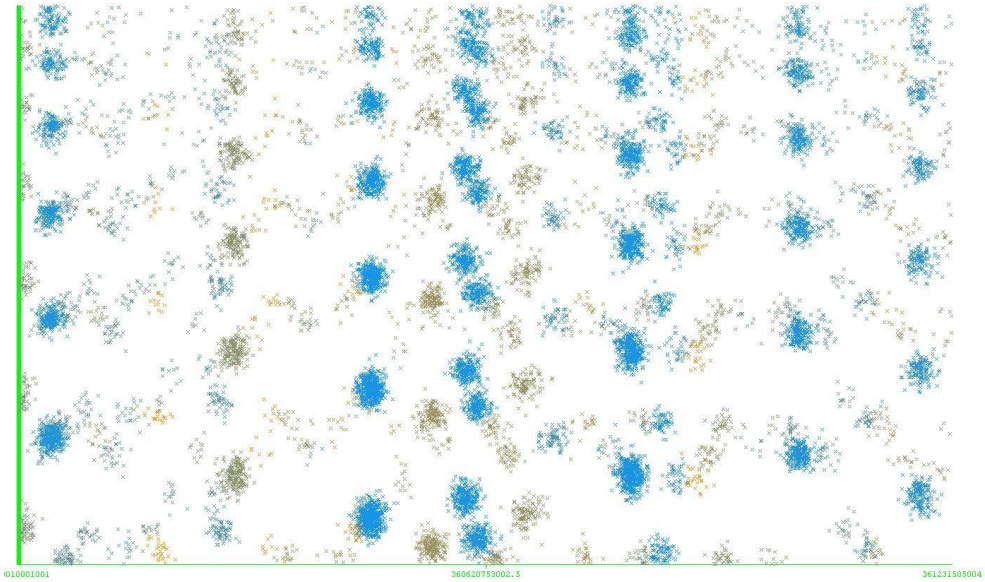

Рисунок 3.9 — Графік залежності geo-id від регіонів проживання пацієнтів

3.3 Інтелектуальний аналіз даних засобами Orange

Застосування такого засобу для інтелектуального аналізу даних як Orange дає значні можливості для ретельного дослідження набору даних. Особливістю даної програми є значна кількість можливостей для зручного представлення даних у вигляді графіків, діаграм, таблиць. У зв'язку з цим, ми скористаємося цією перевагою Orange, для кластеризації набору даних з іформацією про онкохворих пацієнтів, що зареєстровані в США.

Перш за все, оцінимо власне графік з результатами кластеризації. На першому рисунку 3.10 зображено залежність між географічнми координатами пацієнтів. На другому 3.11 ж влесне залежність між віком пацієнта, та розміром пухлини.

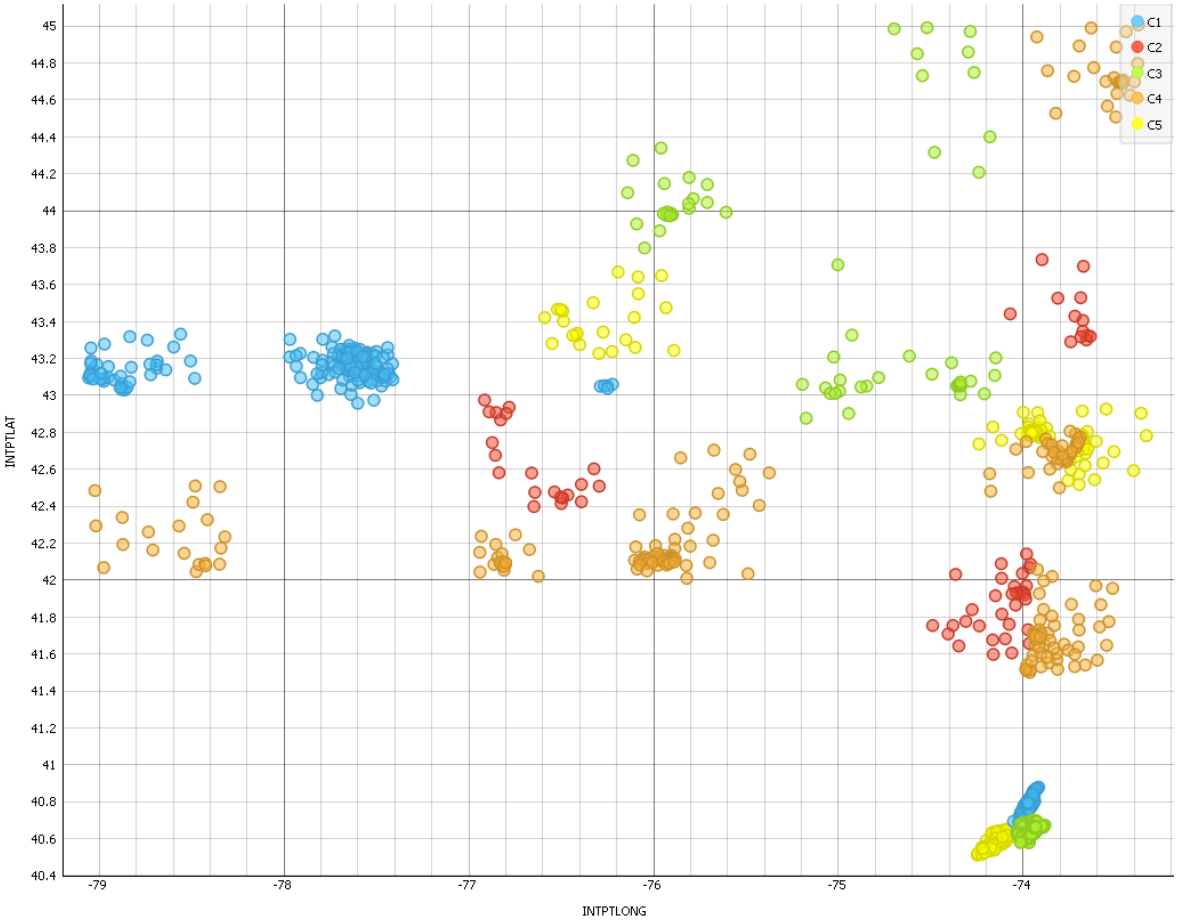

Рисунок 3.10 — Графік залежності між географічно довготою та широтою

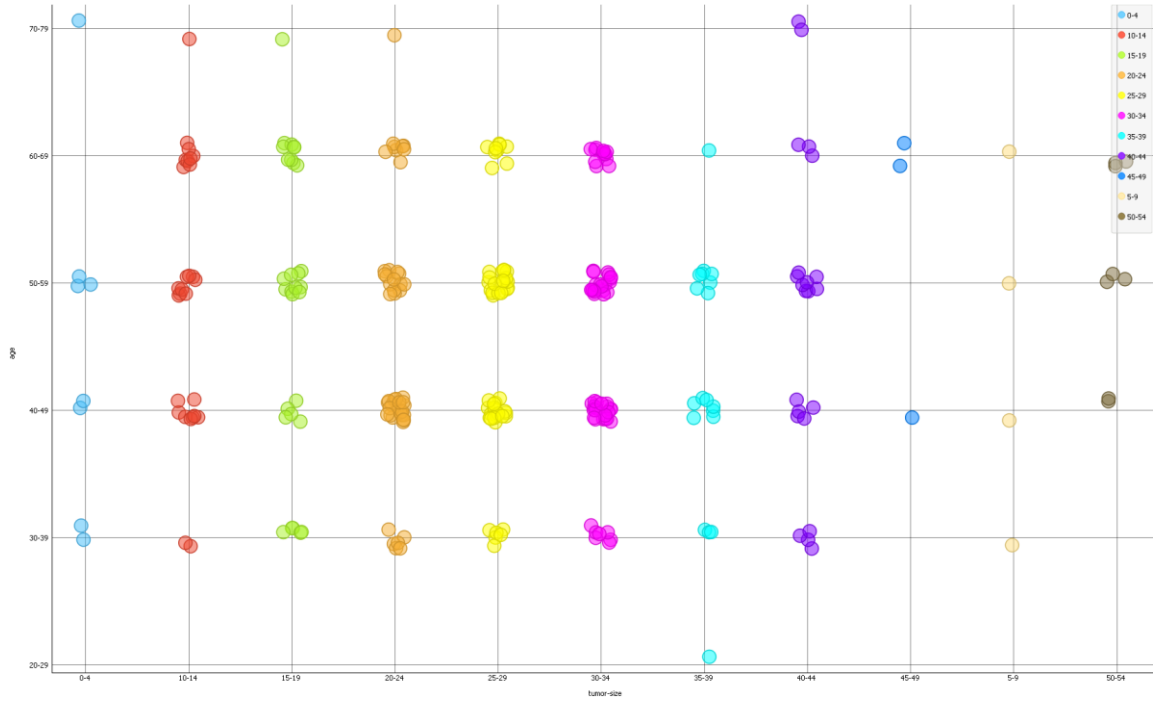

Рисунок 3.11 — Графік залежності віку від розміру пухлини пацієнта

Наступною технікою дослідження даних у нас буде багатошарове шкалювання(рисунок 3.12). Тож, давайте з'ясуємо суть даного підходу.

Багатовимірне шкалювання – ряд пов'язаних між собою статистичних технік, що часто використовують в інформаційній візуалізації для дослідження схожості та відмінності у даних. БШ є особливим видом розміщення. БШ будується як матриця подібних елементів, після чого підписується розміщення кожного елементу у N-вимірному просторі, де через N позначають пріоритетність. Для достатньо малих N результат розміщень може бути представлений як графік чи візуалізований у 3D. БШ потрапляє в таксономію залежно від значення вхідних матриць.

Виділяють наступні типи шкалювання:

*Класичне багатовимірне шкалювання.*Також відоме як Аналіз Нормальних Координат, масштабування Торгенса або Торгенса-Гувера. Вхідна матриця містить значення відмінностей між парами елементів і видає координовану матрицю, конфігурація якої мінімізує функцію втрат.

Метричне багатовимірне шкалювання. Множина класичних БШ, що узагальнює оптимізаційну процедуру до ряду втрачених функцій та вхідних матриць з відомими відстанями та вагами і т. і. Корисна втрачена функція називається стресовою, яку часто мінімізують використовуючи процедуру, що називається стрес мажорування.

Неметричне багатовимірне шкалювання. На відміну від метричного багатовимірного шкалювання Неметричне багатовимірне шкалювання здійснює пошук як непараметричних монотонних зв'язків між відмінностями в поелементній матриці та відмінності в евклідових відстанях між елементами так і розміщення кожного елементу у маловимірному просторі. Зв'язки зазвичай встановлюють використовуючи ізотонічну регресію. Дослідження Найменшого вимру Луїза Гатмена є прикладом неметричної БШ процедури.

Узагальнене БШ. Розширене метричне БШ, в якому цільовий вимір є випадково згладженим неевклідовим виміром. У випадку, якщо відмінними є відстані на поверхні, цільовим простором є інша поверхня. Узагальнене БШ дозволяє знайти мінімально-спотворене проникнення однієї поверхні в іншу.

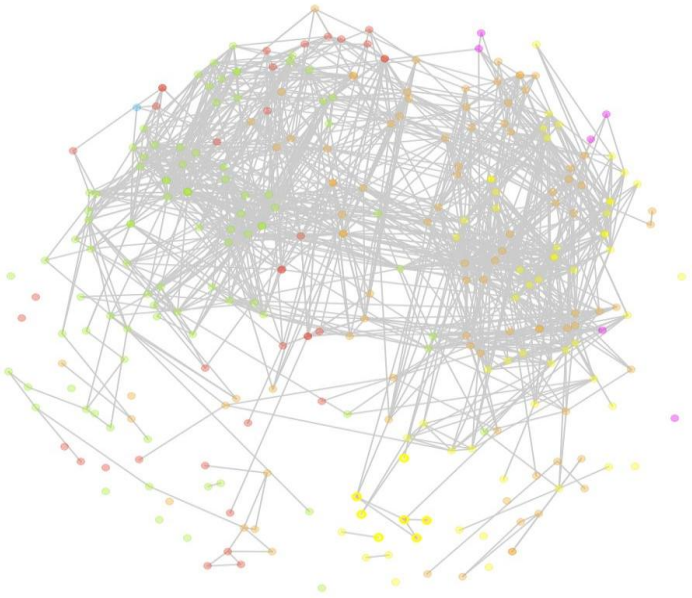

Рисунок 3.12 — Графік результатів багатовимірного шкалювання

Для отримання повної картини результатів, необхідно навести також графік лінійного перетворення методом головних компонент (рисунок 3.13).

Метод головних компонент — один з найбільш розповсюджених методів факторного аналізу. МГК широко представлений у літературних джерелах, звернувшись до яких можна одержати відомості про метод головних компонент з різним ступенем деталізації й математичної строгості.

Серед інших подібних методів, що дозволяють узагальнювати значення елементарних ознак, МГК виділяється простою логічною конструкцією, й у той же час на його прикладі стають зрозумілими загальна ідея й цілі численних методів факторного аналізу.

Метод головних компонент дає можливість по m — числу вихідних ознак виділити m головних компонент, або узагальнених ознак. Простір головних компонент ортогональний. Математична модель методу головних компонент базується на логічному припущенні, що значення множини взаємозалежних ознак породжують деякий загальний результат.

Розв'язування завдання методом головних компонент зводиться до поетапного перетворення матриці вихідних даних X.

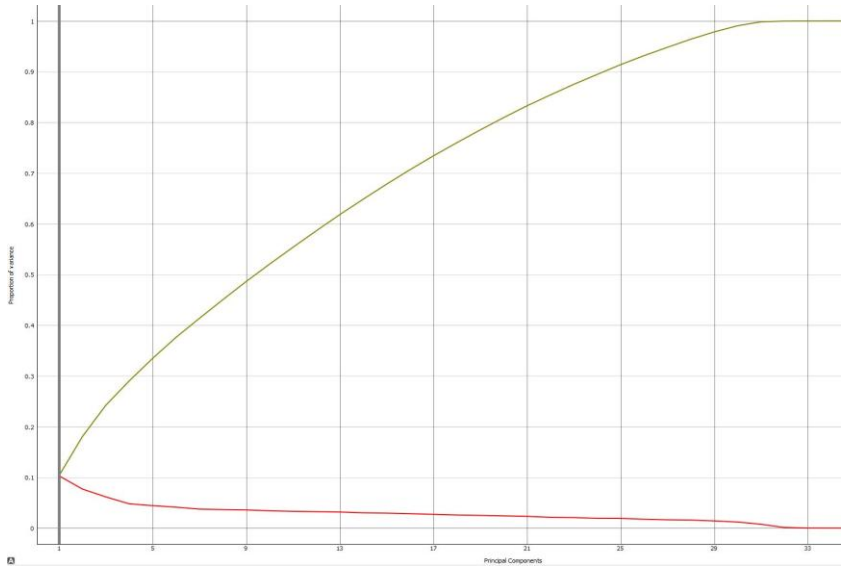

Рисунок 3.13 — Графік лінійного перетворення МГК

Одним з кращих способів інтерпретації і перевірки узгодженості в кластер них даних можна назвати **Silhouette метод** . Методика забезпечує стисле графічне представлення, наскільки добре кожен об'єкт знаходиться в межах його кластера. Результат дослідження зображено на рисунку 3.14.

Значення силуету є мірою того, наскільки схожий об'єкт знаходиться в своєму власному кластері (когезії) в порівнянні з іншими кластерами (поділ). Силует варіюється в діапазоні від -1 до 1, де високе значення вказує, що об'єкт добре узгоджується з його власним кластером і погано узгоджені з сусідніми кластерами. Якщо більшість об'єктів мають високе значення, то конфігурація кластеризації підходить. Якщо багато точок мають низьку або негативну величину, то конфігурація кластеризації може мати занадто багато або занадто мало кластерів.

Силует може бути обчислений з будь-якою метрикою відстані, такою як Евклідова або Манхетенська відстань.

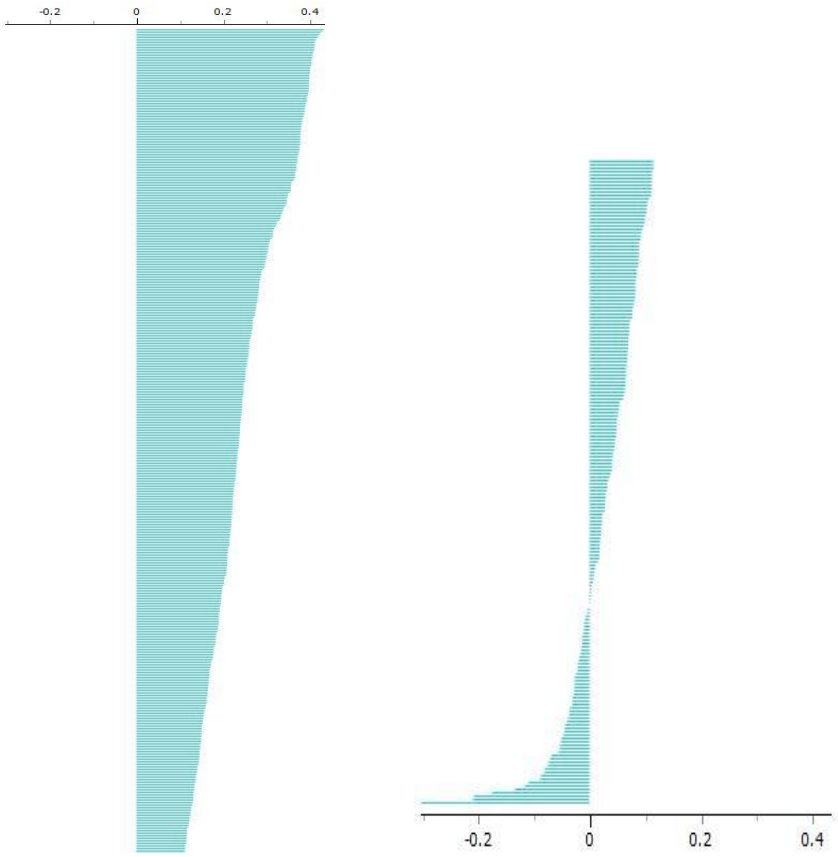

Рисунок 3.14 — Графік перевірки узгодженості даних після кластеризації

Закінчимо огляд результатів кластеризації засобами Orange Sieve діаграмою(рисунок 3.15). Даний різновид діаграми є графічним методом для візуалізації частот в двосторонню спряжену таблицю, при цьому порівнюючи їх з очікуваними частотами. Sieve діаграма була запропонована в 1983 році. На даній діаграмі площа кожного прямокутника пропорційна відношенню очікуваної частоти до аналізованої. Різниця між одержаною і очікуваною частотою зображується як щільність штрихування, котре в свою чергу може бути синього або червоного кольору в залежності від відхилення.

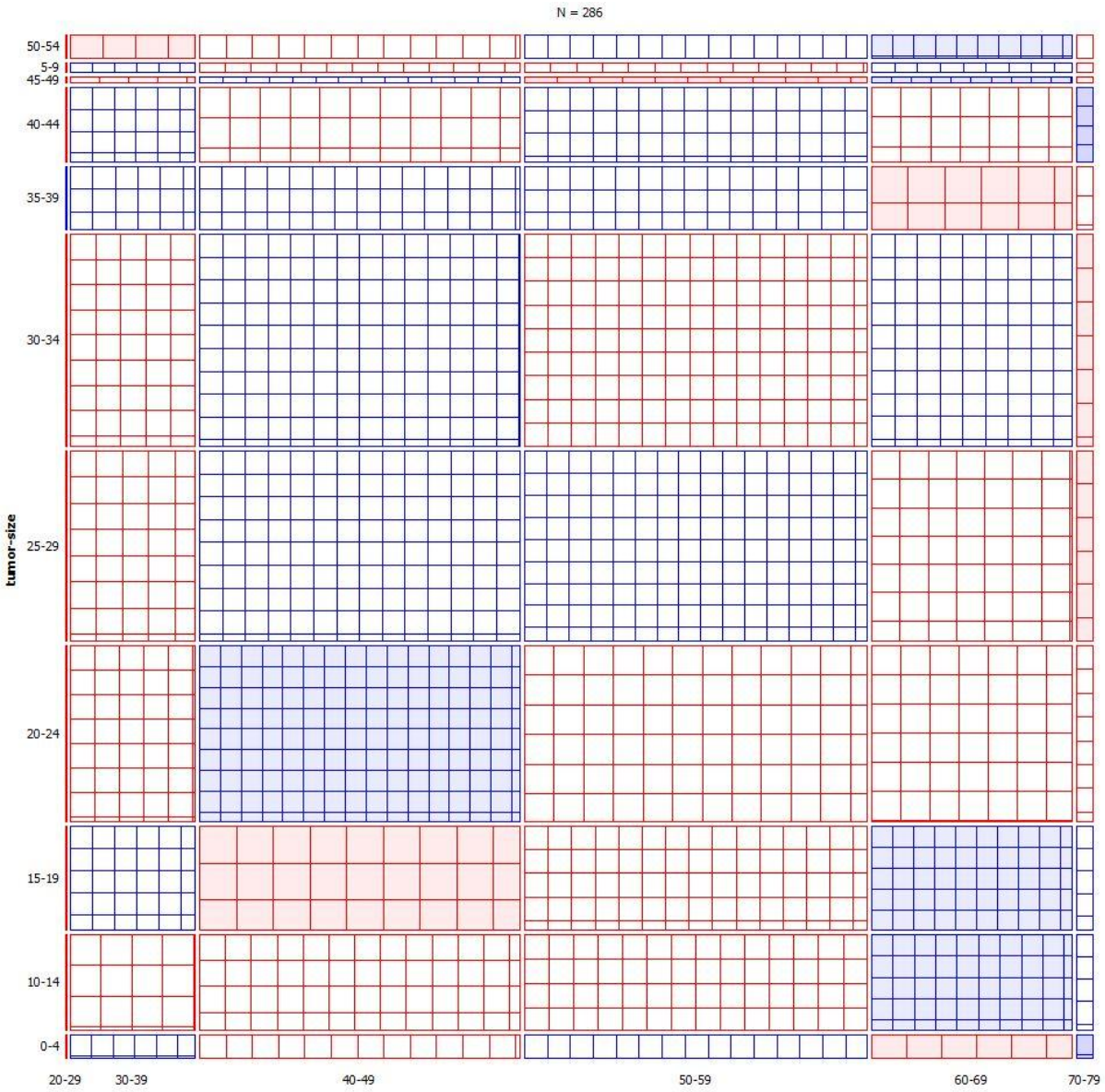

Рисунок 3.15 — Sieve діаграма

3.4 Програмна реалізація алгоритмів кластеризації та результати виконання

Для реалізації алгоритмів кластеризації були застосовано мову програмування Python з цілою низкою доповнень та бібліотек, зокрема : Pandas, Numpy, SciPy, Scikit-learn, Matplotlib. Перші три бібліотеки застосовувалися переважно для зручного представлення вхідних даних. Scikit-learn став нам у нагоді для безпосереднього аналізу даних. Остання з бібліотек стала незамінною для візуалізації результатів, а саме, при побудові діаграм та графіків. Як було з'ясовано у першому розділі, ми реалізували три алгоритми кластеризації. Це надасть нам змогу повніше проаналізувати дані, та порівняти ефективність кожного в поданій ситуації. Результати аналізу кожного з них наведено на рисунках 3.16 – 3.18.

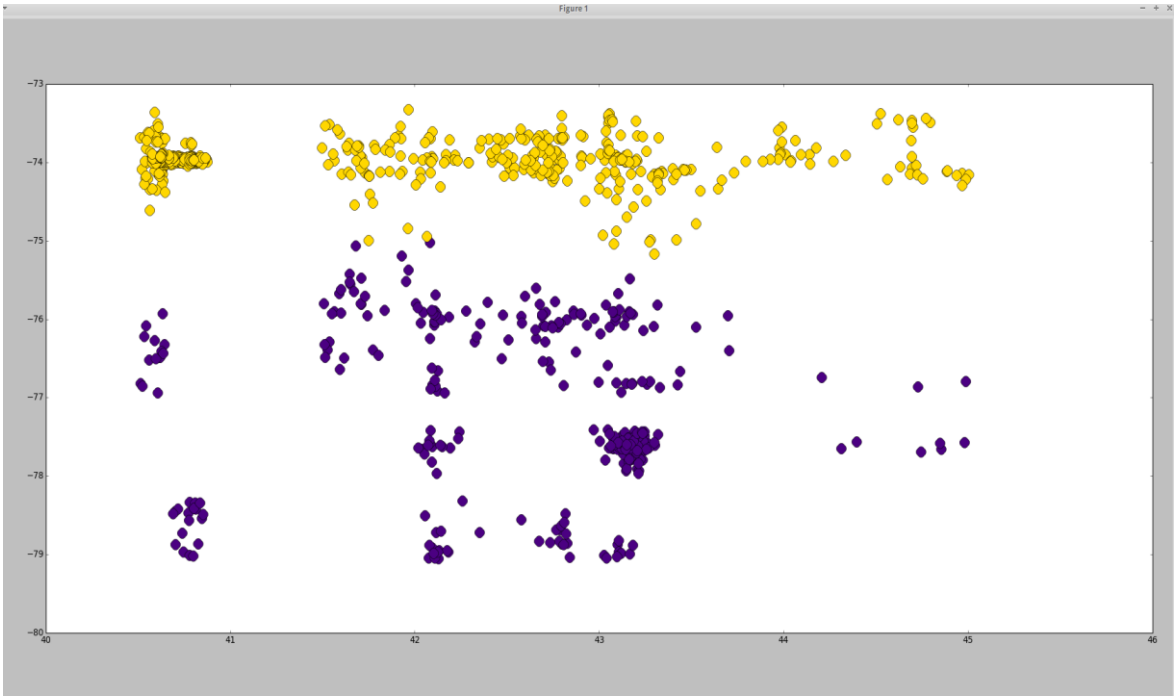

Рисунок 3.16— Результати кластеризації методом k-means

Як можна бачити на малюнку вище , методом k-means весь набір даних був поділений на 2 кластери. У свою чергу, методом DBSCAN було створено 3

кластери. Алгоритм ієрархічної кластеризації виділив 4 кластери на тому ж наборі вхідних даних.

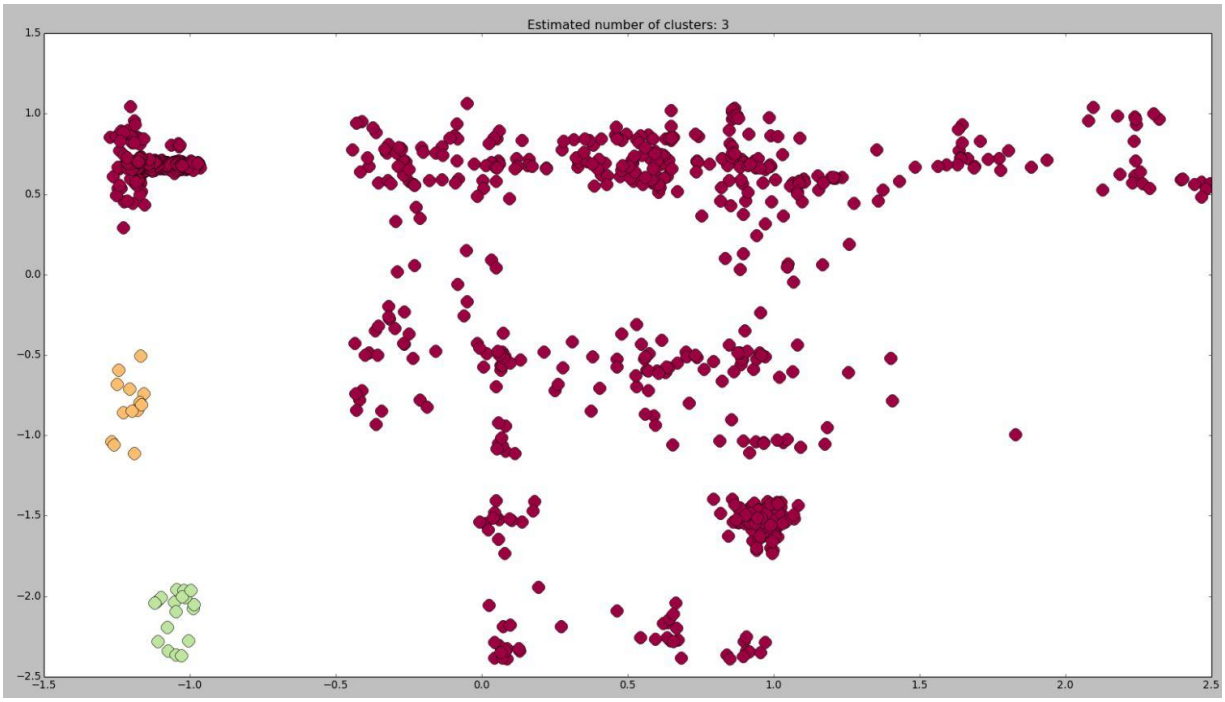

Рисунок 3.17 — Результати кластеризації методом DBSCAN

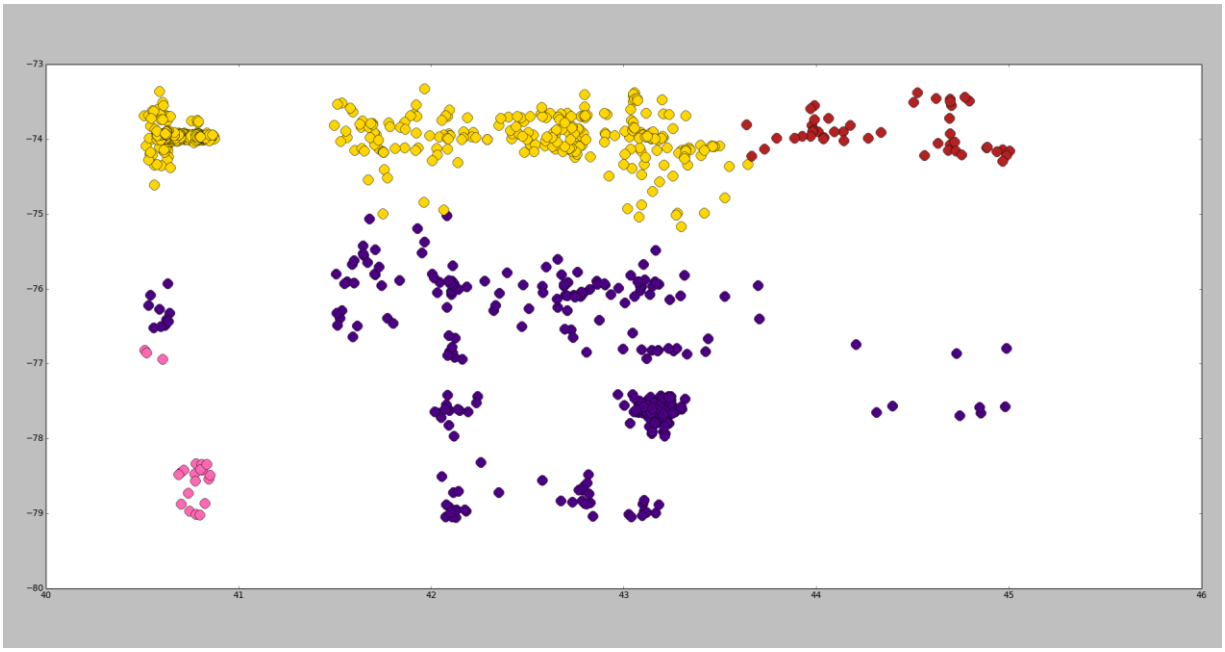

Рисунок 3.18 — Результати кластеризації методом BIRCH

Для оцінки роботи алгоритмів застосуємо кілька основних критеріїв порівняння. Тож, для кращого розуміння спочатку з'ясуємо, для чого власне вони потрібні і що характеризують.

Однорідність (homogeneity) – характеристика, що показує в якій мірі кожен кластер містить члени єдиного класу. Визначається за наступною формулою.

$$h = 1 - \frac{H(C|K)}{H(C)} \tag{10}$$

Повнота (completeness) - характеристика, що показує в якій мірі всі члени даного класу присвоєні одному кластеру. Формула для знаходження наведена нижче.

$$c = 1 - \frac{H(K|C)}{H(K)} \tag{11}$$

V-міра (V-Measure) – міра заснована на ентропії, яка в явному вигляді визначає наскільки критерії однорідності та повноти були задоволені. Дана характеристика обчислюється, як середнє гармонійне різниці однорідності й повноти. Дана міра обчислюєть по наступній формулі:

$$v = 2 \cdot \frac{h \cdot c}{h + c} \tag{12}$$

Скоригований індекс (Adjusted Rand Index) – міра подібності між двома кластерами даних. З математичної точки зору скоригований індекс пов'язаний с точністю і має місце навіть тоді, коли мітки класу не використовуються. Граничні межі даного індексу знаходяться в проміжку [-1,1]. Від'ємні

значення є вельми небажані, в той час коли 1.0 являється ідеальним результатом.

$$AdjustedIndex = \frac{Index - ExpectedIndex}{MaxIndex - ExpectedIndex} \tag{13}$$

$$ARI = \frac{\sum_{ij}\binom{n_{ij}}{2} - [\sum_i \binom{a_i}{2} \sum_j \binom{b_j}{2}]/\binom{n}{2}}{\frac{1}{2}[\sum_i \binom{a_i}{2} + \sum_j \binom{b_j}{2}] - [\sum_i \binom{a_i}{2} \sum_j \binom{b_j}{2}]/\binom{n}{2}} \tag{14}$$

Де, n_{ij}, a_i, b_j значення з таблиці спряженості.

Взаємна інформація (Mutal information) - це функція, що вимірює узгодженість кількох варіантів ігноруючи перестановки. Тобто, скоригована міра взаємної інформації тісно пов'язана зі зміною даних в процесі кластеризації. Формульне представлення даної міри наведено нижче.

$$AMI(U,V) = \frac{MI(U,V) - E\{MI(U,V)\}}{\max\{H(U), H(V)\} - E\{MI(U,V)\}} \tag{15}$$

Граничні межі даної міри знаходяться в проміжку [0,1]. Значення близькі до нуля значну несхожість даних, в той час як близькі до одиниці свідчать про чудову узгодженість. Часто дана характеристика застосовується, як будівельний блок для індексу контексту, який у свою чергу використовується для вибору моделі кластеризації.

Коефіцієнт силуету (Silhouette Coefficient) – міра створена для оцінки моделі з чіткіше визначеним набором кластерів. Даний коефіцієнт визначається для кожного зразка і складається з двох оцінок :

- Середня відстань між зразком і всіма іншими точками в тому ж класі.

- Середня відстань між зразком і всіма іншими точками в наступному найближчому кластері.

Значення варіюються в проміжку [-1, 1], де від'ємні значення свідчать про некоректну кластеризацію а близькі до одиниці – про якісну. Формула для обчислення коефіцієнта для окремого випадку.

$$s = \frac{b - a}{max(a, b)} \tag{16}$$

Таблиця 3.1 — Результати порівняння алгоритмів

Характеристика	K-means	DBSCAN	BIRCH
Однорідність	0.917	0.953	0.960
Повнота	0.766	0.883	0.812
V-міра	0.881	0.917	0.855
Скоригований індекс	0.966	0.952	0.914
Скоригована взаємна інформація	0.855	0.883	0.896
Коефіцієнт силуету	0.722	0.626	0.691

Отже, проаналізувавши отримані результати роботи(таблиця 3.1) алгоритмів можна сказати, що щільнісний алгоритм DBSCAN проявив себе на більшості дослідів дещо краще ніж ітеративний та ієрархічний.

В якості результатів наведемо і карту поширеності онкозахворювань серед населення США , щоб була змога оцінити масштаби. Як видно з рисунка 3.19, основна маса пацієнтів знаходиться на сході та південному сході держави, а з просуванням на захід їх кількість значно зменшується.

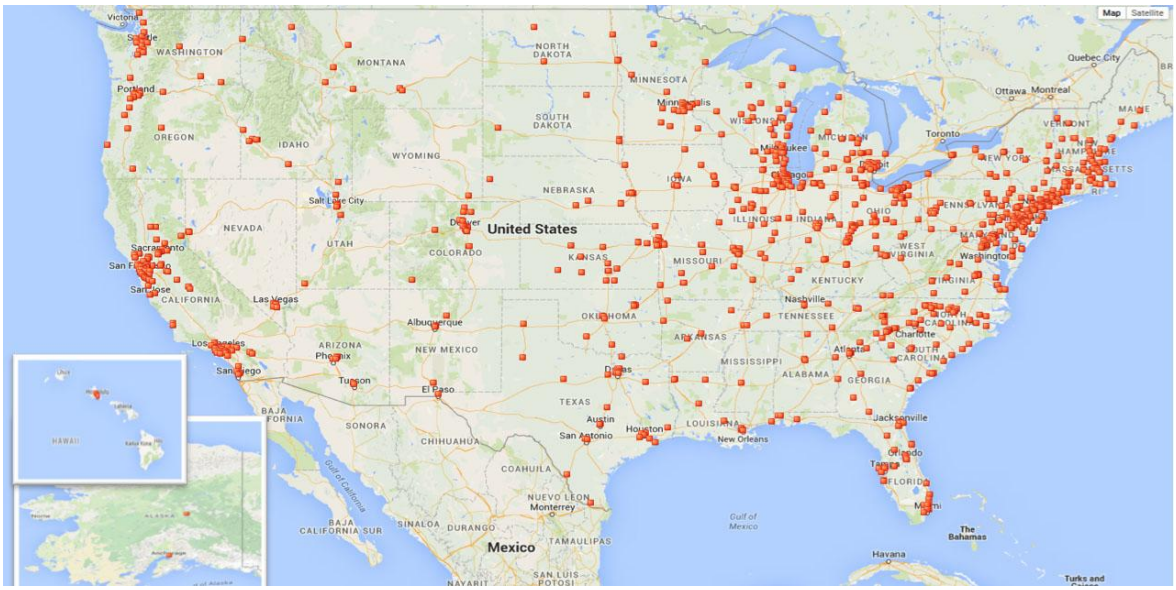

Рисунок 3.19 — Розподіл пацієнтів по території США

Що стосується інших результатів дослідження, ми зведемо їх у таблицях 3.2-3.3, що надасть змогу ретельно оцінити вихідні дані.

Таблиця 3.2 — Статистика діагностування та смертності від онкозахворювань

Вік в якому діагностували	Стать		Вік у якому померли	Стать	
	Чоловіки	Жінки		Чоловіки	Жінки
<1	23.3	23.7	<1	1.8	1.6
1-4	22.7	20.2	1-4	2.3	2.0
5-9	13.8	11.7	5-9	2.4	2.0
10-14	14.2	13.5	10-14	2.3	2.1
15-19	22.6	21.1	15-19	3.4	2.4
20-24	34.6	37.0	20-24	5.1	3.5
25-29	48.8	67.2	25-29	7.0	6.0
30-34	66.1	113.4	30-34	10.2	11.9
35-39	89.5	179.4	35-39	17.1	22.3
40-44	145.4	288.1	40-44	32.2	42.4
45-49	270.9	432.5	45-49	72.1	79.1
50-54	528.2	590.4	50-54	150.1	136.0
55-59	891.2	755.8	55-59	268.0	205.7
60-64	1,376.7	1,003.1	60-64	422.8	309.9
65-69	2,036.1	1,347.4	65-69	640.0	460.7
70-74	2,471.3	1,596.7	70-74	929.3	649.0
75-79	2,785.2	1,826.9	75-79	1,292.9	859.0
80-84	2,935.1	1,919.2	80-84	1,733.5	1,082.2
85+	2,894.8	1,800.1	85+	2,348.8	1,347.2

Таблиця 3.3 — Статистика діагностування, смертності та виживання для кожного виду онкозахворювання

Уражений орган	Діагностували		Померли		Вижили(%)	
	Чоловіки	Жінки	Чоловіки	Жінки	Чоловіки	Жінки
Губи	1.2	0.3	0.0	0.0	89.6	88.0
Язик	5.0	1.8	0.9	0.4	65.2	63.9
Слинна залоза	1.9	1.0	0.4	0.1	64.5	83.3
Порожнина рота	1.8	1.3	0.4	0.3	55.6	63.5
Носоглотка	0.9	0.4	0.3	0.1	58.9	65.0
Мигдалина	3.6	0.7	0.4	0.1	73.1	69.7
Глотка	0.7	0.2	0.4	0.1	44.0	36.6
Стравохід	7.4	1.7	7.4	1.5	18.3	18.5
Шлунок	10.0	5.3	4.5	2.4	28.4	33.2
Тонка кишка	2.7	1.6	0.4	0.3	66.3	67.7
Пряма кишка	32.1	26.8	-	-	64.5	64.3
Печінка	13.0	4.5	9.8	3.6	17.1	18.5
Підшлункова залоза	14.5	11.0	12.5	9.5	7.6	7.8
Ніс	0.9	0.5	0.2	0.1	57.2	53.9
Гортань	5.6	1.1	1.9	0.4	61.5	57.4
Легені та бронхи	67.9	49.4	57.8	37.0	14.9	20.8
Трахеї	0.3	0.1	0.1	0.0	53.3	46.3
Кістки та суглоби	1.1	0.8	0.5	0.3	65.4	70.1
М'які тканини	4.0	2.9	1.9	1.2	64.2	65.7
Шкіра	31.4	18.3	5.6	2.1	88.9	93.5
Груди	1.2	125.0	0.3	21.5	83.6	89.7
Сечовий міхур	35.3	8.6	7.7	2.2	78.9	73.0
Нирки	21.3	10.7	5.7	2.5	73.0	75.0
Очі	1.0	0.7	0.1	0.1	81.2	82.3
Щитовидна залоза	6.9	20.6	0.5	0.5	95.7	98.7
Тимус	0.8	0.7	0.3	0.3	65.8	64.2
Лімфома	26.6	18.4	8.2	5.0	71.5	74.4
Лейкоз	17.6	10.5	9.3	5.2	60.7	58.3

Підсумовуючи зведені у таблицях результати, варто сказати, що дані про пацієнтів були взяті за останні п’ять років. Що стосується випадків

діагностування та смертності, вони наводяться з розрахунком на кожних 100 000 людей.

3.5 Висновок

У даному розділі було детально описано проведені дослідження, результати роботи було подано у вигляді графіків, діаграм і таблиць. Характерно, що крім ретельного дослідження вхідного набору даних, було проаналізовано ефективність застосування кожного з алгоритмів кластеризації за цілою низкою параметрів. Як ми з’ясували, з поставленими завданнями дещо краще справився щільнісний метод DBSCAN.

Найголовнішим є те, що поставлена задача була виконана, і ми одержали відповідні результати.

Найскладнішим у процесі інтелектуального аналізу просторових даних, можна виділити попередню підготовку даних та подання їх у зручному, для дослідження, форматі.

4. ФУНКЦІОНАЛЬНО-ВАРТІСНИЙ АНАЛІЗ ПРОГРАМНОГО ПРОДУКТУ

4.1 Вступ

У даному розділі проводиться оцінка основних характеристик програмного продукту, SDM Tool. Вся логіка та реалізація алгоритмів кластеризації були написані мовою Python у середовищі розробки PyCharm. Крім того для належної візуалізації використовувалась бібліотека Matplotlib.

Програмний продукт призначено для використання на персональних комп’ютерах під управлінням будь-якої операційної системи.

Нижче наведено аналіз різних варіантів реалізації модулю з метою вибору оптимальної, з огляду при цьому як на економічні фактори, так і на характеристики продукту, що впливають на продуктивність роботи і на його сумісність з апаратним забезпеченням. Для цього було використано апарат функціонально-вартісного аналізу.

Функціонально-вартісний аналіз (ФВА) – це технологія, яка дозволяє оцінити реальну вартість продукту або послуги незалежно від організаційної структури компанії. Як прямі, так і побічні витрати розподіляються по продуктам та послугам у залежності від потрібних на кожному етапі виробництва обсягів ресурсів. Виконані на цих етапах дії у контексті метода ФВА називаються функціями.

Мета ФВА полягає у забезпеченні правильного розподілу ресурсів, виділених на виробництво продукції або надання послуг, на прямі та непрямі витрати. У даному випадку – аналізу функцій програмного продукту й виявлення усіх витрат на реалізацію цих функцій.

Фактично цей метод працює за таким алгоритмом:

– визначається послідовність функцій, необхідних для виробництва продукту. Спочатку – всі можливі, потім вони розподіляються по двом групам:

ті, що впливають на вартість продукту і ті, що не впливають. На цьому ж етапі оптимізується сама послідовність скороченням кроків, що не впливають на цінність і відповідно витрат.

– для кожної функції визначаються повні річні витрати й кількість робочих часів.

– для кожної функції наоснові оцінок попереднього пункту визначається кількісна характеристика джерел витрат.

– після того, як для кожної функції будуть визначені їх джерела витрат, проводиться кінцевий розрахунок витрат на виробництво продукту.

4.2 Постановка задачі

У роботі застосовується метод ФВА для проведення техніко-економічний аналізу розробки.

Відповідно цьому варто обирати і систему показників якості програмного продукту.

Технічні вимоги до продукту наступні:

– програмний продукт повинен функціонувати на персональних комп’ютерах із стандартним набором компонент;

– забезпечувати високу швидкість обробки великих об’ємів даних у реальному часі;

– забезпечувати зручність і простоту взаємодії з користувачем або з розробником програмного забезпечення у випадку використовування його як модуля;

– передбачати мінімальні витрати на впровадження програмного продукту.

4.2.1 Обґрунтування функцій програмного продукту

Виходячи з конкретної мети, можна виділити наступні основні функції ПП:

F_1 – вибір мови програмування;

F_2 – вибір оптимальної СКБД;

F_3 – інтерфейс користувача.

Кожна з основних функцій може мати декілька варіантів реалізації.

Функція F_1:

а) мова програмування Python;

б) мова програмування R;

Функція F_2:

а) MongoDB;

б) SQLite.

Функція F_3:

а) інтерфейс користувача, створений за технологією Django;

б) інтерфейс користувача, створений за технологією Kivy.

4.2.2 Варіанти реалізації основних функцій

Варіанти реалізації основних функцій наведені у морфологічній карті системи (рис. 4.1). На основі цієї карти побудовано позитивно-негативну матрицю варіантів основних функцій (таблиця 4.1).

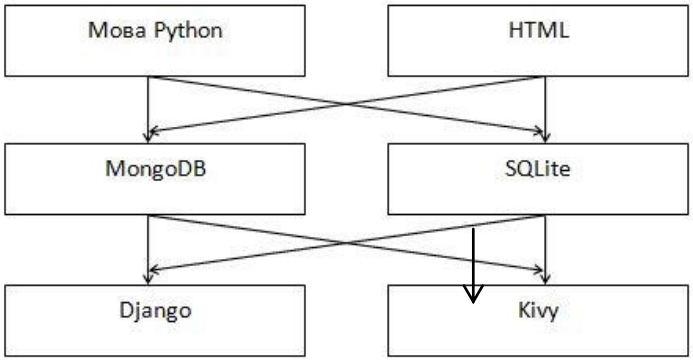

Рисунок 4.1— Морфологічна карта

Морфологічна карта відображує всі можливі комбінації варіантів реалізації функцій, які складають повну множину варіантів ПП.

Таблиця 4.1 – Позитивно-негативна матриця

Основні функції	**Варіанти реалізації**	**Переваги**	**Недоліки**
F1	*А*	Кросплатформений	Низька швидкодія
	Б	Займає менше часу при написанні коду	Більший час на виконання операцій
F2	А	Безкоштовність	Відсутність вкладених запитів
	Б	Надійність, внесення змін без перезапуску, безкоштовність	Необхідність додаткової інсталяції, низький рівень користувацької підтримки
F3	*А*	Простота створення	Відсутність кросплатформеності
	Б	Простота створення,	Кросплатформеність

На основі аналізу позитивно-негативної матриці робимо висновок, що при розробці програмного продукту деякі варіанти реалізації функцій варто відкинути, тому, що вони не відповідають поставленим перед програмним продуктом задачам. Ці варіанти відзначені у морфологічній карті.

Функція *F1:*

Оскільки розрахунки проводяться з великими об’ємами вхідних даних, то час виконання програмного коду є дуже необхідним, тому варіант а) має бути відкинутий.

Функція *F2:*

Вибір СКБД не відіграє велику роль у даному програмному продукту, тому вважаємо варіанти а) та б) гідними розгляду.

Функція *F3:*

Оскільки, програмний продукт реалізується на мові Python, використовуємо варіант Б як єдиний можливий.

Таким чином, будемо розглядати такі варіанти реалізації ПП:

1. F1б – F2а – F3б
2. F1б – F2б – F3б

Для оцінювання якості розглянутих функцій обрана система параметрів, описана нижче.

4.3 Обґрунтування системи параметрів ПП

4.3.1 Опис параметрів

На підставі даних про основні функції, що повинен реалізувати програмний продукт, вимог до нього, визначаються основні параметри виробу, що будуть використані для розрахунку коефіцієнта технічного рівня.

Для того, щоб охарактеризувати програмний продукт, будемо використовувати наступні параметри:

– *X1* – швидкодія мови програмування;

– *X2* – об’єм пам’яті для збереження даних;

– *X3* – час обробки даних;

– *X4* – потенційний об’єм програмного коду.

X1: Відображає швидкодію операцій залежно від обраної мови програмування.

X2: Відображає об’єм пам’яті в оперативній пам’яті персонального комп’ютера, необхідний для збереження та обробки даних під час виконання програми.

X3: Відображає час, який витрачається на дії.

X4: Показує розмір програмного коду який необхідно створити безпосередньо розробнику.

4.3.2 Кількісна оцінка параметрів

Гірші, середні і кращі значення параметрів вибираються на основі вимог замовника й умов, що характеризують експлуатацію ПП як показано у табл. 4.2.

Таблиця 4.2 – Основні параметри ПП

Назва Параметра	**Умовні позначення**	**Одиниці виміру**	**Значення параметра**		
			гірші	**середні**	**Кращі**
Швидкодія мови програмування	X1	Оп/мс	21000	11000	2000
Об’єм пам’яті для збереження даних	X2	Мб	64	32	8
Час обробки запитів користувача	X3	мс	5000	200	60
Потенційний об’єм програмного коду	X4	кількість строк коду	1800	1400	500

За даними таблиці 4.2 будуються графічні характеристики параметрів – рис. 4.2 – рис. 4.5.

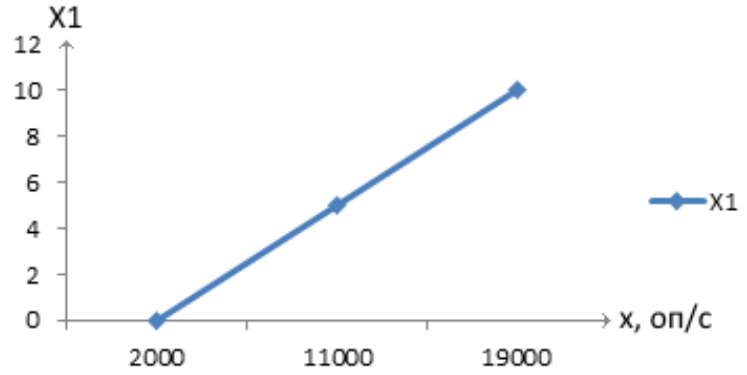

Рисунок 4.2 – X1, швидкодія мови програмування

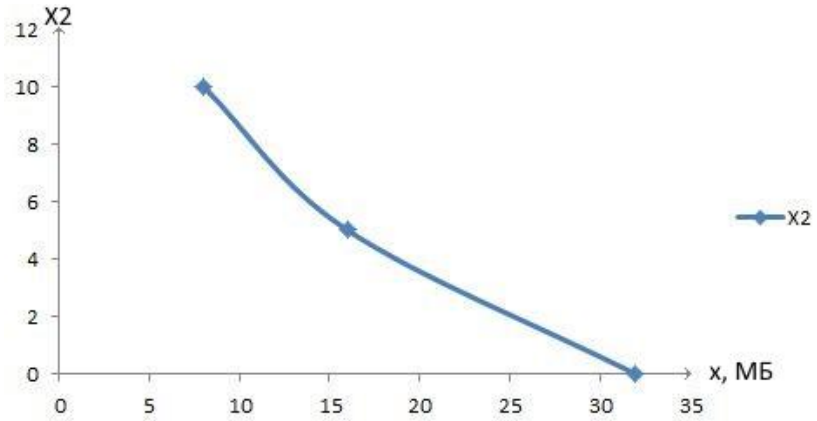

Рисунок 4.3 – X2, об’єм пам’яті для збереження даних

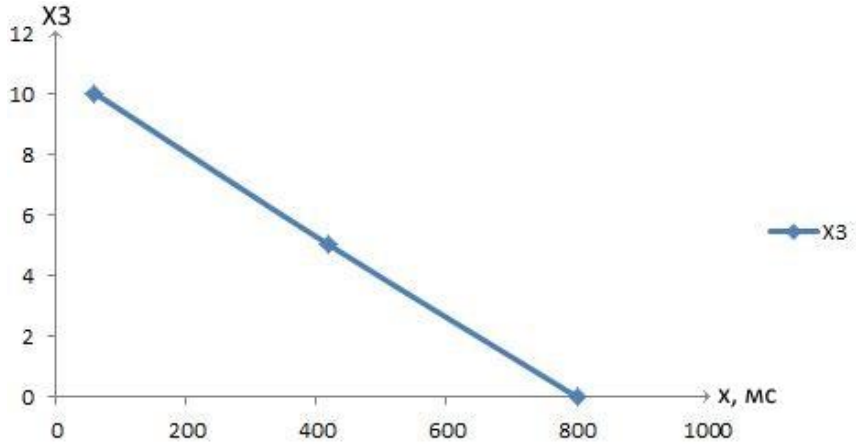

Рисунок 4.4 – X3, час виконання запитів користувача

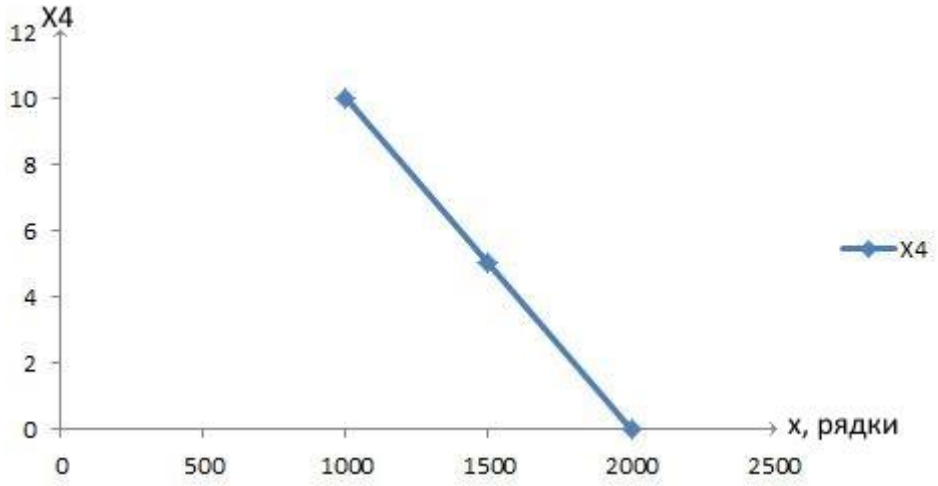

Рисунок 4.5 – X4, потенційний об’єм програмного коду

4.3.3 Аналіз експертного оцінювання параметрів

Після детального обговорення й аналізу кожний експерт оцінює ступінь важливості кожного параметру для конкретно поставленої цілі – розробка програмного продукту, який дає найбільш точні результати при знаходженні параметрів моделей адаптивного прогнозування і обчислення прогнозних значень.

Значимість кожного параметра визначається методом попарного порівняння. Оцінку проводить експертна комісія із 7 людей. Визначення коефіцієнтів значимості передбачає:

– визначення рівня значимості параметра шляхом присвоєння різних рангів;

– перевірку придатності експертних оцінок для подальшого використання;

– визначення оцінки попарного пріоритету параметрів;

– обробку результатів та визначення коефіцієнту значимості.

Результати експертного ранжування наведені у таблиці 4.3.

Таблиця 4.3 – Результати ранжування параметрів

Позначення параметра	Назва параметра	Одиниці виміру	Ранг параметра за оцінкою експерта							Сума рангів R_i	Відхилення Δ_i	Δ_i^2
			1	2	3	4	5	6	7			
X1	Швидкодія мови програмування	Оп/мс	4	3	4	4	4	4	4	27	0,6	0,36
X2	Об’єм пам’яті для збереження даних	Мб	4	4	4	3	4	4	4	27	-1,3	1,69
X3	Час обробки запитів користувача	Мс	2	2	2	2	1	2	2	13	-15,6	243,36
X4	Потенційний об’єм програмного коду	кількість строк коду	5	6	6	6	6	6	6	41	13,8	190,44
	Разом		15	15	15	15	15	15	15	105	0	435,85

Для перевірки степені достовірності експертних оцінок, визначимо наступні параметри:

а) сума рангів кожного з параметрів і загальна сума рангів:

$$R_i = \sum_{j=1}^{N} r_{ij} R_{ij} = \frac{Nn(n+1)}{2} = 108 \qquad (17)$$

де N – число експертів, n – кількість параметрів;

б) середня сума рангів:

$$T = \frac{1}{n} R_{ij} = 27. \quad (18)$$

в) відхилення суми рангів кожного параметра від середньої суми рангів:

$$\Delta_i = R_i - T \quad (19)$$

Сума відхилень по всім параметрам повинна дорівнювати 0;

г)загальна сума квадратів відхилення:

$$S = \sum_{i=1}^{N} \Delta_i^2 = 435{,}85. \quad (20)$$

Порахуємо коефіцієнт узгодженості:

$$W = \frac{12S}{N^2(n^3 - n)} = \frac{12 \cdot 435{,}85}{7^2(5^3 - 5)} = 1{,}17 > W_k = 0{,}67 \quad (21)$$

Ранжування можна вважати достовірним, тому що знайдений коефіцієнт узгодженості перевищує нормативний, котрий дорівнює 0,67.

Скориставшись результатами ранжирування, проведемо попарне порівняння всіх параметрів і результати занесемо у таблицю4.4.

Числове значення, що визначає ступінь переваги *i*–го параметра над *j*–тим, a_{ij} визначається по формулі:

$$a_{ij} = \begin{cases} 1{,}5 \text{ при } X_i > X_j \\ 1.0 \text{ при } X_i = X_j \\ 0.5 \text{ при } X_i < X_j \end{cases}$$

З отриманих числових оцінок переваги складемо матрицю А=$\| a_{ij} \|$.

Таблиця4.4 – Попарне порівняння параметрів

Параметри	Експерти							Кінцева оцінка	Числове значення
	1	2	3	4	5	6	7		
X1 і X2	=	>	=	<	=	<	<	<	0,5
X1 і X3	<	<	<	<	<	<	<	<	0,5
X1 і X4	>	>	>	>	>	>	>	>	1,5
X2 і X3	<	<	<	<	<	<	<	<	0,5
X2 і X4	>	>	>	>	>	>	>	>	1,5
X3 і X4	>	>	>	>	>	>	>	>	1,5

Для кожного параметра зробимо розрахунок вагомості $K_{ві}$ за наступними формулами:

$$K_{\text{ві}} = \frac{b_i}{\sum_{i=1}^{n} b_i}, \text{де } b_i = \sum_{i=1}^{N} a_{ij}. \quad (22)$$

Таблиця 4.5 – Розрахунок вагомості параметрів

Параметриx_i	Параметриx_j				Перша ітер.		Друга ітер.		Третя ітер	
	X1	X2	X3	X4	b_i	$K_{\text{ві}}$	b_i^1	$K_{\text{ві}}^1$	b_i^2	$K_{\text{ві}}^2$
X1	1,0	0,5	0,5	1,5	3,5	0,219	22,25	0,216	100	0,215
X2	1,5	1,0	0,5	1,5	4,5	0,281	27,25	0,282	124,25	0,283
X3	1,5	1,5	1,0	1,5	5,5	0,344	34,25	0,347	156	0,348
X4	0,5	0,5	0,5	1,0	2,5	0,156	14,25	0,155	64,75	0,154
Всього:					16	1	98	1	445	1

Відносні оцінки розраховуються декілька разів доти, поки наступні значення не будуть незначно відрізнятися від попередніх (менше 2%).На другому і наступних кроках відносні оцінки розраховуються за наступними формулами:

$$K_{\text{ві}} = \frac{b_i'}{\sum_{i=1}^{n} b_i'} \text{,де } b_i' = \sum_{i=1}^{N} a_{ij} b_j \tag{23}$$

Як видно з таблиці 4.5, різниця значень коефіцієнтів вагомості не перевищує 2%, тому більшої кількості ітерацій не потрібно.

4.4 Аналіз рівня якості варіантів реалізації функцій

Визначаємо рівень якості кожного варіанту виконання основних функцій окремо.

Абсолютні значення параметрів *X2*(об’єм пам’яті для збереження даних) та *X1* (швидкодія мови програмування) відповідають технічним вимогам умов функціонування даного ПП.

Абсолютне значення параметра *X3 (*час обробки даних) обрано не найгіршим (не максимальним), тобто це значення відповідає або варіанту а) 1000 мс або варіанту б) 80мс.

Коефіцієнт технічного рівня для кожного варіанта реалізації ПП розраховується так (таблиця 4.6):

$$K_K(j) = \sum_{i=1}^{n} K_{\text{в}i,j} B_{i,j} \text{ ,} \tag{24}$$

де n – кількість параметрів;$K_{вi}$– коефіцієнт вагомості i–го параметра;B_i – оцінка i–го параметра в балах.

За даними з таблиці 4.6 за формулою 25:

$$K_K = K_{\text{ТУ}}[F_{1k}] + K_{\text{ТУ}}[F_{2k}] + \ldots + K_{\text{ТУ}}[F_{zk}], \quad (25)$$

Таблиця 4.6 – Розрахунок показників рівня якості варіантів реалізації основних функцій ПП

Основні функції	Варіант реалізації функції	Абсолютне значення параметра	Бальна оцінка параметра	Коефіцієнт вагомості параметра	Коефіцієнт рівня якості
F1(X1)	А	11000	3,6	0,215	0,770
F2(X2)	А	16	3,4	0,253	0,962
F3(X3,X4)	А	800	2,4	0,378	0,974
	Б	80	1	0,154	0,154

визначаємо рівень якості кожного з варіантів:

K_{K1} = 0,770 + 0,962 + 0,978 = 2,71

K_{K2} = 0,770 + 0,962 + 0,154 = 1,65

Як видно з розрахунків, кращим є перший варіант, для якого коефіцієнт технічного рівня має найбільше значення.

4.5 Економічний аналіз варіантів розробки ПП

Для визначення вартості розробки ПП спочатку проведемо розрахунок трудомісткості.

Всі варіанти включають в себе два окремих завдання:

1. Розробка проекту програмного продукту;

2. Розробка програмної оболонки;

Завдання 1 за ступенем новизни відноситься до групи А, завдання 2 – до групи Б. За складністю алгоритми, які використовуються в завданні 1 належать до групи 1; а в завданні 2 – до групи 3.

Для реалізації завдання 1 використовується довідкова інформація, а завдання 2 використовує інформацію у вигляді даних.

Проведемо розрахунок норм часу на розробку та програмування для кожного з завдань.

Проведемо розрахунок норм часу на розробку та програмування для кожного з завдань. Загальна трудомісткість обчислюється як

$$T_O = T_P \cdot K_П \cdot K_{СК} \cdot K_M \cdot K_{СТ} \cdot K_{СТ.М} \quad (26)$$

де T_P – трудомісткість розробки ПП;$K_П$ – поправочний коефіцієнт;$K_{СК}$ – коефіцієнт на складність вхідної інформації; K_M – коефіцієнт рівня мови програмування;$K_{СТ}$ – коефіцієнт використання стандартних модулів і прикладних програм; $K_{СТ.М}$ – коефіцієнт стандартного математичного забезпечення.

Для першого завдання, виходячи із норм часу для завдань розрахункового характеру степеню новизни А та групи складності алгоритму 1, трудомісткість дорівнює: T_P =90 людино-днів. Поправочний коефіцієнт, який враховує вид нормативно-довідкової інформації для першого завдання: $K_П$ = 1.7. Поправочний коефіцієнт, який враховує складність контролю вхідної та вихідної інформації для всіх семи завдань рівний 1: $K_{СК}$ = 1. Оскільки при розробці першого завдання використовуються стандартні модулі, врахуємо це за допомогою коефіцієнта $K_{СТ}$ = 0.8. Тоді, за формулою 5.1, загальна трудомісткість програмування першого завдання дорівнює:

$T_1 = 90 \cdot 1.7 \cdot 0.8 = 122.4$ людино-днів.

Проведемо аналогічні розрахунки для подальших завдань.

Для другого завдання (використовується алгоритм третьої групи складності, степінь новизни Б), тобто $Т_Р$ =27 людино-днів, $К_П$ =0.9,$К_{СК}$ = 1,$К_{СТ}$ =0.8:

$Т_2 = 27 \cdot 0.9 \cdot 0.8 = 19.44$ людино-днів.

Складаємо трудомісткість відповідних завдань для кожного з обраних варіантів реалізації програми, щоб отримати їх трудомісткість:

$Т_I = (122.4 + 19.44 + 4.8 + 19.44) \cdot 8 = 1328{,}64$ людино-годин;

$Т_{II} = (122.4 + 19.44 + 6.91 + 19.44) \cdot 8 = 1345.52$ людино-годин;

Найбільш високу трудомісткість має варіант II.

В розробці беруть участь три програмісти з окладом 20000 грн., Визначимо зарплату за годину за формулою:

$$С_Ч = \frac{М}{T_m \cdot t} \text{ грн.}, \quad (27)$$

де М – місячний оклад працівників;T_m – кількість робочих днів тиждень;t – кількість робочих годин в день.

$$С_Ч = \frac{20000 + 20000 + 20000}{3 \cdot 21 \cdot 8} = 119{,}02 \text{ грн.}$$

Тоді, розрахуємо заробітну плату за формулою

$$С_{ЗП} = С_Ч \cdot Т_i \cdot К_Д \quad (28)$$

де $С_Ч$– величина погодинної оплати праці програміста;$Т_i$ – трудомісткість відповідного завдання; $К_Д$ – норматив, який враховує додаткову заробітну плату.

Зарплата розробників за варіантами становить:

I. $С_{ЗП} = 119{,}02 \cdot 1328.64 \cdot 1.2 = 189805{,}71$ грн.

II. $С_{ЗП} = 119{,}02 \cdot 1345.52 \cdot 1.2 = 192172{,}54$ грн.

Відрахування на єдиний соціальний внесок в залежності від групи професійного ризику (II клас) становить 38%:

I. $С_{ВІД} = С_{ЗП} \cdot 0.38 = 189805,71*0.22 = 72125.9$ грн.

II. $С_{ВІД} = С_{ЗП} \cdot 0.38 = 75285,04*0.22 = 73025.36$ грн.

Тепер визначимо витрати на оплату однієї машино-години. ($С_М$)

Так як одна ЕОМ обслуговує одного програміста з окладом 20000 грн., з коефіцієнтом зайнятості 0,2 то для однієї машини отримаємо:

$С_Г = 12 \cdot М \cdot К_З = 12 \cdot 20000 \cdot 0,2 = 48000$ грн.

З урахуванням додаткової заробітної плати:

$С_{ЗП} = С_Г \cdot (1 + К_З) = 48000 \cdot (1 + 0.2) = 57600$ грн.

Відрахування на єдиний соціальний внесок:

$С_{ВІД} = С_{ЗП} \cdot 0.22 = 57600 * 0,22 = 12672$ грн.

Амортизаційні відрахування розраховуємо при амортизації 25% та вартості ЕОМ – 50000 грн.

$С_А = К_{ТМ} \cdot К_А \cdot Ц_{ПР} = 1.15 \cdot 0.25 \cdot 50000 = 14375$ грн.,

де $К_{ТМ}$– коефіцієнт, який враховує витрати на транспортування та монтаж приладу у користувача; $К_А$– річна норма амортизації; $Ц_{ПР}$– договірна ціна приладу.

Витрати на ремонт та профілактику розраховуємо як:

$С_Р = К_{ТМ} \cdot Ц_{ПР} \cdot К_Р = 1.15 \cdot 50000 \cdot 0.05 = 2875$ грн.,

де $К_Р$– відсоток витрат на поточні ремонти.

Ефективний годинний фонд часу ПК за рік розраховуємо за формулою:

$Т_{ЕФ} = (Д_К – Д_В – Д_С – Д_Р) \cdot t_З \cdot К_В = (365 – 104 – 8 – 16) \cdot 8 \cdot 0.9 = 1706.4$ годин,

де $Д_К$ – календарна кількість днів у році; $Д_В$, $Д_С$ – відповідно кількість вихідних та святкових днів; $Д_Р$ – кількість днів планових ремонтів устаткування; t –кількість робочих годин в день; $К_В$– коефіцієнт використання приладу у часі протягом зміни.

Витрати на оплату електроенергії розраховуємо за формулою:

$С_{ЕЛ} = Т_{ЕФ} \cdot N_С \cdot К_З \cdot Ц_{ЕН} = 1706,4 \cdot 0,156 \cdot 0,9733 \cdot 2,0218 = 523.83$ грн.,

де $N_С$ – середньо-споживча потужність приладу; $К_З$– коефіцієнтом зайнятості приладу; $Ц_{ЕН}$ – тариф за 1 КВт-годин електроенергії.

Накладні витрати розраховуємо за формулою:

$С_Н = Ц_{ПР} \cdot 0.67 = 50000 \cdot 0,67 = 33500$ грн.

Тоді, річні експлуатаційні витрати будуть:

$С_{ЕКС} = С_{ЗП} + С_{ВІД} + С_А + С_Р + С_{ЕЛ} + С_Н$

$С_{ЕКС} = 57600 + 12672 + 14375 + 2875 + 523.83 + 33500 = 176345$ грн.

Собівартість однієї машино-години ЕОМ дорівнюватиме:

$С_{М\text{-}Г} = С_{ЕКС} / Т_{ЕФ} = 35269.03 / 1706,4 = 86.03$ грн/час.

Оскільки в даному випадку всі роботи, які пов‘язані з розробкою програмного продукту ведуться на ЕОМ, витрати на оплату машинного часу, в залежності від обраного варіанта реалізації, складає:

$С_М = С_{М\text{-}Г} \cdot Т$

I. $С_М = 86.03 * 1328,64 = 137310$ грн.;

II. $С_М = 86.03 * 1345.52 = 139055$ грн.;

Накладні витрати складають 67% від заробітної плати:

$С_Н = С_{ЗП} \cdot 0,67$

I. $С_Н = 371700.08 * 0,67 = 249039,18$ грн.;

II. $С_Н = 381700.08 * 0,67 = 258754,98$ грн.;

Отже, вартість розробки ПП за варіантами становить:

$С_{ПП} = С_{ЗП} + С_{ВІД} + С_М + С_Н$

I. $С_{ПП} = 189805,71 + 12672 + 137310 + 249039,18 = 289654.17$ грн.;

II. $С_{ПП} = 192172,54 + 12672 + 139055 + 258754,98 = 302156.83$ грн.;

4.6 Вибір кращого варіанта ПП техніко-економічного рівня

Розрахуємо коефіцієнт техніко-економічного рівня за формулою:

$К_{ТЕРj} = К_{Кj} / С_{Фj}$,

$К_{ТЕР1}$ = 2,57 / 289654.17 = 0,89·10^{-6};

$К_{ТЕР2}$ = 1,89 / 302156.83 = 0,62·10^{-6};

Як бачимо, найбільш ефективним є перший варіант реалізації програми з коефіцієнтом техніко-економічного рівня $К_{ТЕР1}$= 0,89· 10^{-6}.

4.7 Висновки до розділу 4

В даному розділі проведено повний функціонально-вартісний аналіз ПП, який було розроблено в рамках дипломного проекту. Процес аналізу можна умовно розділити на дві частини.

В першій з них проведено дослідження ПП з технічної точки зору: було визначено основні функції ПП та сформовано множину варіантів їх реалізації; на основі обчислених значень параметрів, а також експертних оцінок їх важливості було обчислено коефіцієнт технічного рівня, який і дав змогу визначити оптимальну з технічної точки зору альтернативу реалізації функцій ПП.

Другу частину ФВА присвячено вибору із альтернативних варіантів реалізації найбільш економічно обґрунтованого. Порівняння запропонованих варіантів реалізації в рамках даної частини виконувалось за коефіцієнтом ефективності, для обчислення якого були обчислені такі допоміжні параметри, як трудомісткість, витрати на заробітну плату, накладні витрати.

Після виконання функціонально-вартісного аналізу програмного комплексу що розроблюється, можна зробити висновок, що з альтернатив, що залишились після першого відбору двох варіантів виконання програмного комплексу оптимальним є перший варіант реалізації програмного продукту. У нього виявився найкращий показник техніко-економічного рівня якості $К_{ТЕР1}$= 0,89· 10^{-6}.

Цей варіант реалізації програмного продукту має такі параметри:

– мова програмування – Python;

– СКБД MongoDB;

– за візуалізацію відповідатиме Matplotlib.

Отже, даний варіант виконання програмного комплексу дає хороші показники швидкодії, чудову візуалізацію та високу точність результатів.

ВИСНОВКИ

В ході дипломного проектування було досліджено шляхи застосування інтелектуального аналізу просторових даних у медицині. Перш за все, ми з'ясували специфічні для сфери ІАПД вимоги, після чого приступили до дослідження існуючих рішень та підходів, які найчастіше застосовують для досягнення поставлених цілей. Наступним нашим кроком було обрання підходів до аналізу просторових даних. Характерно, що ми зупинилися на кластерному аналізі, що дало нам змогу максимально якісно провести дослідження. Кластерний аналі ми проводили у кілька етапів, для того, щоб використати його сильні сторони найкращим чином. Головним чином було пройдено наступні етапи:

- Відбір вибірки для кластеризації.
- Визначення множини характеристик, по яких будуть оцінюватися об'єкти у вибірці.
- Обчислення значень тієї чи іншої міри схожості між об'єктами.
- Застосування одного з методів кластерного аналізу для створення груп схожих об'єктів.
- Перевірка достовірності результатів кластеризації.

Що стосується алгоритмів кластеризації, то ми зупинилися на трьох варіантах, а саме: k-means, BIRCH та DBSCAN. Такий вибір дозволив нам максимально використати сильні сторони кожного з різновидів методів кластерного аналізу. А все тому, що кожен з алгоритмів принципово відрізняється. Алгоритм k-means відноситься до категорії ітеративних і був обраний по причині своєї доступності для розуміння та простоти реалізації. Наступним ми обрали алгоритм BIRCH, який є ієрархічним алгоритмом кластеризації, за своєю суттю. Його головними чеснотами являються швидкість виконання та можливість якісно працювати на обмеженому наборі даних. Останнім алгоритмом, що привернув нашу увагу був щільнісний алгоритм кластеризації DBSCAN. Відмінною рисою цього чудового алгоритму є його

можливість віднаходження кластерів довільної форми. Крім того він чудово працює на наборі даних зі значними показниками шуму.

Інструментарій для виконання роботи було обрано з огляду на його функціональність та відповідність нашим вимогам для даного дослідження. Головним чином мовою програмування Python було реалізовано весь цикл дослідження, в той час, як бібліотеки Numpy, SciPy та Pandas були дуже корисними у підготовці даних. В свою чергу scikit-learn ми застосували для власне аналізу даних. Зручну ж візуалізацію нам забезпечили бібліотека matplotlib та програма Orange . Що стосується програми Weka, то вона дала нам змогу дослідити обрані алгоритми на величезних наборах даних.

Отримані результати були розгорнуто наведені у третьому розділі. Для кращої оцінки вихідних даних, вони були представлені у вигляді графіків, діаграм та таблиць. Крім дослідження власне вхідного набору даних, ми провели оцінку ефективності роботи кожного з алгоритмів кластеризації. Порівнювали наші алгоритми за такими характеристиками, як : однорідність, повнота, скоригований індекс, скоригована взаємна інформація, коефіцієнт силуету та v-міра. Зазначимо, що для використаного набору даних дещо кращу ефективність продемонстрував щільнісний метод DBSCAN.

В останньому розділі роботи проводиться повний функціонально-вартісний аналіз ПП. Після виконання якого можна зробити висновок, що з альтернатив, що залишилися після першого відбору варіантів виконання програмного комплексу оптимальним є перший варіант реалізації. У нього виявився найкращий показник техніко-економічного рівня якості $К_{ТЕР1}= 0,89 \cdot 10^{-6}$. Цей варіант реалізації програмного продукту має такі параметри:

- мова програмування – Python;
- СКБД MongoDB;
- за візуалізацію відповідатиме Matplotlib.

ПЕРЕЛІК ПОСИЛАНЬ

1. Паклин Н.Б. Бизнес-аналитика: от данных к знаниям./ Паклин Н.Б., Орешков В. И. – СПб.: Питер, 2009. — 45-69 с.

2. Барсегян А.А. Методы и модели анализа данных: OLAP и Data Mining./ Барсегян А.А., Куприянов М.С., Степаненко В.В., Холод И.И. – СПб.: БХВ – Петербург, 2008. — 13-22 с.

3. Кацко И.А. Практикум по анализу данных на компьютере. / Кацко И.А., Н.Б. Паклин. – К.: КолосС, 2009. — 125-128с.

4. Дюк В.А. Data Mining: учебный курс. / Дюк В.А., Самойленко А.П. – СПб.: Питер, 2001. — 55-87с.

5. Хайкин С. Нейронные сети: полный курс. 2-е. изд. / Хайкин С. – К.: Издательский дом «Вильямс», 2006. — 18-22с.

6. Ханк Д.Э. Бизнес-прогнозирование. / Ханк Д.Э., Уичерн Д.У., Райте А.Дж. – К.: Издательский дом «Вильямс», 2003. —78-93с.

7. Дубров А. М. Многомерные статистические методы: Учебник. / Дубров А. М., Мхитарян В. С., Трошин Л. И. – К.: Финансы и статистика, 2000. — 8-16с.

8. Андрейчиков А.В. Интеллектуальные информационные системы. / Андрейчиков А.В., Андрейчикова О.Н. – К.: ФиС, 2004. — 22-45с.

9. Мандель И. Д. Кластерный анализ. / Мандель И. Д. — К.: Финансы и статистика, 1988. — 10 с.

10.Жамбю М. Иерархический кластер-анализ и соответствия. / Жамбю М. — К.: Финансы и статистика, 1988. — 345 с.

11.Дюран Б. Кластерный анализ. / Дюран Б., Оделл П. — К.: Статистика, 1999. — 128-84с.

12.Gorban A.N. Principal Graphs and Manifolds, Ch. 2 in: Handbook of Research on Machine Learning Applications and Trends: Algorithms, Methods, and Techniques. / Gorban A.N., Zinovyev A.Y. — MA: MIT Press, 2010 — pp. 125—140.

13.P. S. Bradley. "Clustering via Concave Minimization, " in Advances in Neural Information Processing Systems, vol. 9. / P. S. Bradley., O. L. Mangasarian, W. N. Street. MA: MIT Press, 1997— pp. 368—374.

14.A. K. Jain. Algorithms for Clustering Data. / A. K. Jain., R. C. Dubes — Prentice Hall, IGI Global, 2012 — pp. 43-62.

15.L. Kaufman. Finding Groups in Data: An Introduction to Cluster Analysis. / L. Kaufman, P. J. Rousseeuw. — MA: MIT Press, 2007 — pp. 215—266.