Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

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

АТЕСТАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти другий (магістерський)

ДОСЛІДЖЕННЯ ТА РЕАЛІЗАЦІЯ МЕТОДУ ВІДНОВЛЕННЯ ДАНИХ У ТАБЛИЦЯХ «ОБ'ЄКТ-ВЛАСТИВІСТЬ»
(тема)

Виконала:
студентка 2 курсу, групи ІНФм-19-2
Петухова К.С
(прізвище, ініціали)

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

Освітня програма Інформатика
(повна назва освітньої програми)

Керівник доц. Руденко Д.О.
(посада, прізвище, ініціали)

Допускається до захисту

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

2020 р.

Харківський національний університет радіоелектроніки

Факультет Інформаційно-аналітичних технологій та менеджменту
(повна назва)

Кафедра Інформатики
(повна назва)

Рівень вищої освіти другий (магістерський)

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

Освітня програма Інформатика
(повна назва освітньої програми)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
«____»_____ 20 ___ р.

ЗАВДАННЯ
НА АТЕСТАЦІЙНУ РОБОТУ

студентові Петуховій Катерині Сергіївні
(прізвище, ім'я, по батькові)

1. Тема роботи Дослідження та реалізація методу відновлення даних в таблицях «об'єкт-властивість»

затверджена наказом по університету від «23» жовтня 2020 року №1428Ст.

2. Термін подання студентом роботи до екзаменаційної комісії 23 листопада 2020 р.

3. Вихідні дані до роботи *Теоретичні відомості про методи відновлення даних у таблицях, математичні моделі методів відновлення даних у таблицях, перелік використовуваних програмних засобів:C#, .NET, MS Visual Studio*

4. Перелік питань, що потрібно опрацювати в роботі
1. Огляд методів кластеризації даних
2. Аналіз проблеми пропущених даних
3. Аналіз методів відновлення даних у таблицях «об'єкт-властивість»
4. Аналіз предметної області
5. Математична модель методів відновлення даних
6. Програмна реалізація методів відновлення даних та кластеризації

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (слайдів) *Актуальність проблеми пропущених даних в таблицях «об'єкт-властивість», постановка задачі, математична модель методу відновлення даних, архітектура проекту, програмна реалізація.*

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

Найменування розділу	Консультант (посада, прізвище, ім'я, по батькові)	Позначка консультанта про виконання розділу	
		підпис	дата

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів роботи	Терміни виконання етапів роботи	Примітка
1	Отримання завдання на атестаційну роботу	23.10.2020	
2	Аналіз завдання, підбір літератури	24.10.20-25.10.20	
3	Аналіз літератури з досліджуваної проблеми	26.10.20-27.10.20	
4	Аналіз технічних засобів	28.10.20-29.10.20	
5	Розробка методу	30.10.20-05.11.20	
6	Програмна реалізація	06.11.20-16.11.20	
7	Оформлення пояснювальної записки	17.11.20-25.11.20	
8	Перевірка на плагіат	26.11.20	
9	Рецензування	27.11.20	
10	Підготовка презентації та доповіді	28.11.20	
11	Занесення роботи в електронний архів	29.11.20	
12	Попередній захист атестаційної роботи	30.11.20	

Дата видачі завдання 23 жовтня 2020 р.

Студент _____
(підпис)

Керівник роботи _____ доц. Руденко Д.О.
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ/ABSTRACT

Пояснювальна записка до атестаційної роботи: 79 с., 1 табл., 39 рис., 48 джерел.

ВЕЛИКІ ДАНІ, АНАЛІЗ ДАНИХ, ТАБЛИЦЯ «ОБ'ЄКТ-ВЛАСТИВІСТЬ», ПРОПУСКИ ДАНИХ, МЕТОДИ ВІДНОВЛЕННЯ ДАНИХ, КЛАСТЕРИЗАЦІЯ, МЕТОД РЕГРЕСІЙНОГО АНАЛІЗУ, МЕТОД БАРЛЕТА, EM-АЛГОРИТМ, ZET-АЛГОРИТМ, БАГАТОВИМІРНА НЕЧІТКА ЕКСТРАПОЛЯЦІЯ, FCM, C#.

Метою дослідження є розробка методів FCM та багатовимірної нечіткої екстраполяції для кластеризації даних з пропусками.

Об'єктом дослідження є набір даних з маркетинговими дослідженнями покупців інтернет-магазинів, отриманий шляхом соціального опитування на краудсорсингових платформах.

Проаналізовано актуальність проблеми пропущених даних для вирішення задачі аналізу даних та їх кластеризації. Проведено дослідження методів відновлення даних у таблицях вигляду «об'єкт-властивість». Побудовано математичну модель методів відновлення даних. Реалізовано метод багатовимірної нечіткої екстраполяції та FCM.

У результаті роботи здійснена програмна реалізація системи для відновлення пропущених даних та кластеризації набору даних з маркетинговими дослідженнями.

BIG DATA, DATA MINING, TABLE OBJECT-PROPERTY, DATA WITH GAPS, DATA RECOVERY, CLUSTERING, REGRESSION ANALYSIS, BARLET METHOD, EM-ALGORITHM, ZET-ALGOITHM, MULTIWARE FUZZY EXTRAPOLATION, FCM, C#.

The aim of the research is to develop FCM and multiware fuzzy extrapolation algorithms for clustering data with gaps.

The object of the research is the marketing research dataset with information about buyers at online stores, obtained through social research on crowdsourcing platforms.

The relevance of the problem of data with gaps for data analysis and clustering is analyzed. A research of data recovery methods in tables of the form «object-property». A mathematical model of data recovery methods is constructed. The multiware fuzzy extrapolation and FCM algorithms are implemented.

As a result of the work the software implementation of the system for recovery of the missed data and clustering of a data set with marketing researches is carried out.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

АД – аналіз даних

БД – база даних

ГБ – гігабайт

ЕБ – ексабайт

ЕМ – expectation maximization

ЗБ – зетабайт

ОС – операційна система

ПК – персональний комп’ютер

CLI – common language infrastructure

CLR – common language runtime

FCM – fuzzy classifier means

IDE – integrated development environment

IL – intermidiate language

IT – informational technologies

MAR – missing at random

MCAR – missing completely at random

MS VS – microsoft visual studio

NMAR – missing not at random

OLAP – online analytical processing

RTB – real time bidding

VVV – volume, velocity, variety

ВСТУП

Зі швидким розвитком комп’ютерних технологій і наук задачі, що постають перед науковцями змінюються, так раніше обчислювальні машини, а, разом з ними і комп’ютерні науки, розвивалися досить повільно і основний інтерес становив розвиток саме апаратної бази: збільшення пам’яті, як оперативної так і жорстких дисків, загальна швидкість обчислень та інше. Розвиток програмного забезпечення був обмежений саме апаратними характеристиками, необхідно було розв’язувати проблеми збільшення працездатності програми та вирішувати задачі скорочення ресурсів, а саме зменшення програмного коду. На сьогодні проблема з пам’яттю або обчислювальними ресурсами не стоїть так гостро, апаратна база стрімко розвивається і більше не становить основний інтерес.

Ще однією рушійною силою є те, що комп’ютери стали все більш доступними і немалий вклад в розвиток технологій роблять пересічні користувачі. Враховуючи це, збільшується кількість інформації, що потрапляє в комп’ютер і безпосередньо в Інтернет.

Зі збільшенням об’ємів даних стали виникати нові задачі, все більший інтерес становить робота з даними і розв’язання проблем, що пов’язані з їх обробкою і подальшим аналізом. Оскільки можливості створення нового контенту мають всі бажаючи – обсяги даних непомірно зростають, а їх впорядкованість слабка. З’являється необхідність пошуку даних, в тому числі зображень, їх обробки, для маркетингових і статистичних досліджень, використанню у інших сферах, наприклад повсякденного життя пересічного користувача.

Найпопулярнішими напрямками досліджень частіше стають: Big Data, Data Mining, Machine Learning, постає питання пошуку даних їх глобальний та інтелектуальний аналіз. Серед таких актуальних завдань знаходять своє місце і поняття класифікації та кластеризації даних.

Класифікація встановлює закономірності для розбиття даних на заздалегідь визначені підмножини (класи). Кластеризація є процес розбиття заданої вибірки об’єктів (спостережень) на підмножини (як правило, непересічні), які називаються кластерами, так, щоб кожен кластер складався з схожих об’єктів, а об’єкти різних кластерів істотно відрізнялися. Задачі подібні, але основна відмінність полягає у тому, що кластеризація передбачає розбиття за умови початкової невизначеності щодо конкретних груп, вона може мати критерії щодо кількості кінцевих кластерів, але не їх зміст, тобто, можна сказати, що це навчання без вчителя. Виділяють наступні основні завдання кластерного аналізу:

– розробка типології або класифікації;

– дослідження корисних концептуальних схем групування об’єктів;

– породження гіпотез на основі дослідження даних;

– перевірка гіпотез для визначення, чи дійсно типи (групи) виділені тим чи іншим способом, присутні в наявних даних.

Отже, кластеризація становить інтерес, як спосіб попередньої обробки даних, для більш зручного подальшого аналізу. Отримавши необхідні групи, а також їх центроїди можна продовжувати роботу вже з конкретними представниками, а не з усім набором даних, що особливо актуально в умовах безкінечно зростаючого об’єму інформації.

Даний підхід дозволяє краще зрозуміти дані, шляхом використання для кожного кластеру найбільш оптимального алгоритму аналізу; провести стиснення, виділивши найбільш типових представників, за умов збитковості даних; виявлення новизни, шляхом виділення об’єктів, що не потрапили до жодного з кластерів. Сфера застосування може бути доволі широка, використання в сегментації зображень, аналізу відео, прогнозування, аналіз текстів, оптимізація, машинне навчання, інтелектуальному аналізі даних.

Таким чином вивчення та використання методів кластеризації для вирішення багатьох важливих питань є досить цікавою задачею, але вона має низку своїх недоліків і проблем, що потребують вирішення. Так, наприклад,

однією з проблем кластеризації можна виділити роботу з пропусками даних. Існує багато методів, але вони не передбачають відсутності якоїсь кількості інформації. Але, як тільки виходимо за рамки тестових даних і переходимо до обробки реальних – стикаємося з цією проблемою, адже в дійсності ідеальних даних не існує і всі вони містять шуми (некорисну інформацію, яка може зашкодити результату), пропуски, невідповідні формати та інше.

Для реалізації та аналізу методів відновлення даних у таблицях типу «об’єкт-властивість» було обрано предметну область маркетингових досліджень. Це зумовлено тим, що дана сфера має велику актуальність з розвитком реклами у соціальних мережах, особливо на території України. Для реалізації маркетингових досліджень проводяться анкетування та соціальні опитування, які частіш за все і мають пропуски у даних або некоректні дані.

Таким чином, доцільним є розгляд варіантів рішення проблеми кластеризації даних з пропусками та огляд методів відновлення пропусків у таблицях типу «об’єкт-властивість».

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ОГЛЯД МЕТОДІВ ВІДНОВЛЕННЯ ДАНИХ

1.1 Аналіз великих даних

Соціально-економічний феномен «великих даних» охопив усі галузі і бізнес-функції. Самі «великі дані» стали важливим фактором виробництва поряд з працею і капіталом. Дослідники виділяють такі основні способи використання великих даних для створення цінності:

– великі дані можуть зробити інформацію доступнішою, використовуючи її на більш високій частоті;

– оскільки організації створюють і зберігають більше транзакційних даних в цифровий формі, вони можуть збирати більш точну інформацію про продуктивність на всіх етапах від кадастрів продуктів до днів хвороби і, отже, виявляти мінливість і підвищувати продуктивність;

– великі дані дозволяють сегментувати споживачів, що в подальшому полегшує можливість робити клієнтам персоналізовані пропозиції;

– аналітика дозволяє оптимізувати процес прийняття рішень.

Тобто, основний напрям роботи з великим даним – це їхній аналіз.

1.1.1 Визначення великих даних

Результатом інтелектуальної діяльності людей є лавиноподібне зростання інформації в світі. За даними компанії IDC Digital Universe, до середини 2015 року загальна кількість даних перевищила 6,5 зеттабайт (ЗБ). (1 ЗБ = 1024 ексабайта (ЕБ), 1 ЕБ = 1 млрд. гігабайтів). За прогнозами до кінця 2021-го року світовий обсяг інформації досягне 40-44 зеттабайт, тобто на кожного жителя Землі припадатиме по 5200 Гб.

Значущими джерелами великих даних є сховища бізнес-інформації, показання різних датчиків і пристроїв, інтернет (соціальні мережі, інформаційні сайти, файлообмінники), мобільні пристрої і т.д.

Все більш актуальною стає проблема підвищення ефективності в використанні великих даних чи допоможуть вони збільшити продажі або скоротити витрати бізнесу.

Традиційні технології зберігання, аналізу та управління типовими базами даних (БД) далеко не завжди ефективні при роботі з великими даними. Це пов’язано з рядом особливостей:

– великі дані знаходяться в різних сховищах, що не дозволяє використовувати звичні інструменти для встановлення корисних взаємозв’язків між ними;

– великі дані не мають структурованого формату як це характерно для традиційних БД;

– дані безперервно оновлюються.

До галузевих джерел, що генерує величезні потоки даних, можна віднести медичні технології (різні відео, моніторинг в реальному часі, зображення), телефонію, електростанції, комунальні служби. У світі з’являються технології «smart grid», що дозволяють комунальним службам щохвилини і навіть щомиті вимірювати десятки тисяч параметрів, пов’язаних зі споживанням електроенергії домогосподарствами.

Зростає використання великих даних державними і комерційними секторами з накопиченими обсягами даних в сотні терабайт або петабайт. Моніторинг поведінки користувачів на сайтах, покупців інтернет – магазинів, завантаження інформації на сайтах, обмін відео, наприклад, через YouTube, генерують щодня величезні масиви інформації.

Дані класифікуються як Extremely Big Data, якщо терміни їх зберігання обчислюються роками. Термін big data (великі дані) вперше з’явився в спеціальному випуску журналу Nature, присвяченому вибухового зростання

світових обсягів інформації в 2008 році. Big Data є сьогодні найважливішим трендом інтернет-маркетингу і IT-індустрії.

Великі дані – новий суттєвий елемент в інформаційній структурі сучасного суспільства. Саме тому, технології та методи їх аналізу настільки значущі. Залежно від цілей, які ставляться перед аналізом великих даних, можна виділити наступні його основні різновиди:

– інформаційно-пошуковий і візуальний аналіз. Чи не набуваючи нових знань про предмет, ми отримуємо можливість розглянути його з різних точок зору і по частинах. Це досягається видачею конкретного запиту до реляційної бази даних. Такий вид аналізу малоприйнятний до великих даними, так як розкид варіацій для конкретного запиту буде в результаті занадто великий;

– оперативно-аналітичний аналіз або OLAP (OnLine Analytical Processing). В OLAP дані агрегуються до будь-якого ступеня узагальнення з будь-якого розрізу. В даному випадку вже є можливість виявити закономірності і тенденції в даних, які інакше були б не видно. Практично OLAP вводить нас в сферу узагальнених даних;

– інтелектуальний аналіз або Data Mining. Цей тип аналізу спрямований на виявлення прихованих закономірностей в даних (повторюваних шаблонів або кластерів даних), тобто, моделей, що лежать в основі структури даних і дають можливість краще розуміти дані і передбачати їхню поведінку. Саме Data Mining є безпосереднє виявлення знань.

Великі дані розраховані на виявлення залежностей, які можуть бути засновані на непрямих взаємозв’язках.

1.1.2 Основні принципи роботи з великими даними

Великі дані – це сукупність технологій, які покликані здійснювати три операції:

– обробляти великі в порівнянні з «стандартними» сценаріями обсяги даних;

– вміти працювати з даними, що швидко надходять в дуже великих обсягах. Тобто даних не просто багато, а їх постійно стає все більше і більше;

– вміти працювати зі структурованими і слабо структурованими даними паралельно і в різних аспектах.

Вважається, що ці «вміння» дозволяють виявити приховані закономірності, що ховаються від обмеженого людського сприйняття. Це дає безпрецедентні можливості оптимізації багатьох сфер нашого життя: державного управління, медицини, телекомунікацій, фінансів, транспорту, виробництва тощо.

Визначальними характеристиками для великих даних є, крім їх фізичного обсягу, і інші, що підкреслюють складність завдання обробки і аналізу цих даних. Набір ознак VVV (volume, velocity, variety – фізичний обсяг, швидкість приросту даних і необхідності їх швидкої обробки, можливість одночасно обробляти дані різних типів) був вироблений компанією Meta Group в 2001 році з метою вказати на рівну значимість управління даними по всім трьом аспектам.

Надалі з’явилися інтерпретації з чотирма V (додавалася veracity – достовірність), п’ятьма V (viability – життєздатність і value – цінність), сім’ю V (variability – мінливість і visualization – візуалізація). Але компанія IDC, наприклад, інтерпретує саме четвертий V як value (цінність), підкреслюючи економічну доцільність обробки великих обсягів даних у відповідних умовах.

Виходячи з вищенаведених визначень, основні принципи роботи з великими даними такі:

– горизонтальна масштабованість. Це – базовий принцип обробки великих даних. Як вже говорилося, великих даних з кожним днем стає все більше. Відповідно, необхідно збільшувати кількість обчислювальних вузлів,

за якими розподіляються ці дані, причому обробка повинна відбуватися без погіршення продуктивності;

– відмовостійкість. Цей принцип випливає з попереднього. Оскільки обчислювальних вузлів в кластері може бути багато (іноді десятки тисяч) і їх кількість, не виключено, буде збільшуватися, зростає і ймовірність виходу машин з ладу. Методи роботи з великими даними повинні враховувати можливість таких ситуацій і передбачати превентивні заходи;

– локальність даних. Так як дані розподілені по великій кількості обчислювальних вузлів, то, якщо вони фізично знаходяться на одному сервері, а обробляються на іншому, витрати на передачу даних можуть стати невиправдано великими. Тому обробку даних бажано проводити на тій же машині, на якій вони зберігаються.

Ці принципи відрізняються від тих, які характерні для традиційних, централізованих, вертикальних моделей зберігання добре структурованих даних. Відповідно, для роботи з великими даними розробляють нові підходи і технології.

1.1.3 Аналіз даних

Аналіз даних – це область математики та інформатики, що займається побудовою і дослідженням найбільш загальних математичних методів і обчислювальних алгоритмів вилучення знань з експериментальних (в широкому сенсі) даних; процес дослідження, фільтрації, перетворення і моделювання даних з метою отримання корисної інформації та прийняття рішень.

Аналіз даних – сукупність дій, здійснюваних дослідником в процесі вивчення отриманих тим чи іншим чином даних з метою формування певних уявлень про характер явища, що описується цими даними. Дослідник намагається дані «згорнути», скоротити їх кількість, прагнучи при цьому не

втратити корисну інформацію, потенційно в них закладену. Робиться це звичайно з допомогою математичних методів.

Також, аналіз даних – це процес вивчення статистичних даних за допомогою математичних методів, що не припускають ймовірнісної моделі досліджуваного явища. Протистоїть імовірнісного. підходу до обробки даних, що спирається на їх вірогідну інтерпретацію (як випадкової вибірки з генеральної сукупності) і використання імовірнісних моделей для побудови і вибору найкращих методів обробки. Одержані за допомогою імовірнісного підходу висновки спираються на строго доведені математичні положення. Зокрема, цей підхід забезпечує коректний перенесення результатів з вибірки на генеральну сукупність. У методах АД подібні можливості не закладені. Ці методи не задовольняють строгим математичним вимогам.

Вибір найкращого методу тут майже завжди спирається на формалізації евристичні міркування. Тому проблема обґрунтування одержуваних висновків вимагає особливої уваги. Особливо гострою стає необхідність виділення «точок дотику» змісту завдання і математичного формалізму, реалізації в процесі людино-машинного діалогу.

До методів АД відносять і імовірнісні методи в тих випадках, коли невдається перевірити адекватність реальності ймовірнісної моделі, передбачуваної методом.

Виділення методів АД обумовлено потребами ряду наук (в т.ч. соціології), в яких велика потреба пошуку статистичних закономірностей. Однак припущення, що лежать в основі ймовірнісних методів, розроблених спеціально для вирішення таких завдань, часто не виконуються.

Існує думка, що оскільки методи АД з точки зору суворої математики не є достатньо обґрунтованими, то має сенс використовувати їх лише на попередньому етапі аналізу для уточнення уявлень дослідника про досліджуваному явищі, коригування понятійного апарату, формулювання гіпотез і т.д. Однак методи АД можуть служити і засобом отримання фундаментального знання, виявлення невідомих раніше закономірностей,

якщо перейти на новий рівень розуміння самого математичного формалізму: вважати, що адекватним розв'язуваної задачі є не окремий метод, а сукупність методів, що застосовуються відповідно до визначених методологічними принципами.

Аналіз даних – термін, що ототожнюється з поняттям «прикладна статистика», яка розуміється як науч. дисципліна, що розробляє і систематизує поняття, прийоми, математичні методи і моделі, призначені для організації збору, стандартного запису, систематизації і обробки статистичних даних з метою їх зручного уявлення, інтерпретації та отримання наукових і практичних висновків.

Також, АД – це процедури пошуку статистичних закономірностей («згортки» інформації), що не зводяться до застосування формальних алгоритмів. В основі лежить комплексне використання математико-статистичних методів і методів АД з опорою на кілька методологічних принципів, розглянутих нижче.

Перший принцип – варіація передумов, що лежать в основі обираних методів (будь-який метод спирається на певну модель досліджуваного явища, тобто певну систему передумов і постулатів): зміна таких передумов, розгляд наслідків цієї зміни, порівняння використання різних передумов і т.д. Актуальність реалізації цього принципу пояснюється тим, що для більшості методів перевірка спроможності закладених в них моделей в соціологічних задачах є досить проблематичною.

Другий принцип – системний підхід. В процесі АД вишукуються різні прийоми для повного використання і ендогенної інформації (тобто даних, що описують досліджуваний об'єкт), і екзогенної (тобто даних, що описують «середовище проживання» об'єкта). Системний підхід пред'являє до дослідника підвищені вимоги, оскільки носить принципово міждисциплінарний характер.

Третій принцип – відмова від тієї точки зору, що будь-яке дослідження має початок і кінець. АД – спосіб існування даних. Готовність до постійного

повернення до одних і тих самих даних. У безперервному процесі АД передбачаються розриви, що дозволяють витягати накопичену інформацію і приймати рішення, пов’язані з управлінням обробкою даних, з вибором подальших кроків АД. Формальні операції перемежовуються з неформальними процедурами прийняття рішення. З появою нових даних виникають нові ідеї, підходи, методи, уточнюється розуміння процесів, що відбуваються і т.д. У соціології реалізація цього принципу актуальна, тому що соціолог зазвичай не має тієї апріорної моделі досліджуваного явища, яка є необхідною і для вибору формального апарату аналізу даних і взагалі для проведення дослідження, починаючи з формулювання гіпотез і розробки способу збору.

1.1.4 Методи і техніки аналізу великих даних

Міжнародна консалтингова компанія McKinsey, що спеціалізується на вирішенні завдань, пов’язаних зі стратегічним управлінням, виділяє 11 методів і технік аналізу, які можна застосувати до великих даних:

– Data Mining (видобуток даних, інтелектуальний аналіз даних, глибинний аналіз даних) – сукупність методів виявлення в даних раніше невідомих, нетривіальних, практично корисних знань, необхідних для прийняття рішень. До таких методів, зокрема, відносяться навчання асоціативним правилами (association rule learning), класифікація (розбиття на категорії), кластерний аналіз, регресійний аналіз, виявлення та аналіз відхилень і інші;

– краудсорсінг – класифікація і збагачення даних силами широкого, невизначеного кола осіб, які виконують цю роботу без вступу в трудові відносини;

– змішування і інтеграція даних (data fusion and integration) – набір технік, що дозволяють інтегрувати різнорідні дані з різноманітних джерел з

метою проведення глибинного аналізу (наприклад, цифрова обробка сигналів, обробка природної мови, включаючи тональний аналіз, та ін.);

– машинне навчання, включаючи навчання з учителем і без вчителя – використання моделей, побудованих на базі статистичного аналізу або машинного навчання для отримання комплексних прогнозів на основі базових моделей;

– штучні нейронні мережі, мережевий аналіз, оптимізація, в тому числі генетичні алгоритми (genetic algorithm – евристичні алгоритми пошуку, які використовуються для вирішення завдань оптимізації та моделювання шляхом випадкового підбору, комбінування і варіації шуканих параметрів з використанням механізмів, аналогічних природному відбору в природі);

– розпізнавання образів;

– прогнозна аналітика;

– імітаційне моделювання (simulation) – метод, що дозволяє будувати моделі, що описують процеси так, як вони проходили б у дійсності. Імітаційне моделювання можна розглядати як різновид експериментальних випробувань;

– просторовий аналіз (spatial analysis) – клас методів, які використовують топологічну, геометричну і географічну інформацію, видобуту з даних;

– статистичний аналіз – аналіз часових рядів, A / B-тестування (A / B testing, split testing – метод маркетингового дослідження; при його використанні контрольна група елементів порівнюється з набором тестових груп, в яких один або кілька показників були змінені, для того щоб з’ясувати, які з змін покращують цільовий показник);

– візуалізація аналітичних даних – подання інформації у вигляді малюнків, діаграм, з використанням інтерактивних можливостей та анімації як для отримання результатів, так і для використання в якості вихідних даних для подальшого аналізу. Дуже важливий етап аналізу великих даних, що

дозволяє представити найважливіші результати аналізу в найбільш зручному для сприйняття вигляді.

1.1.5 Системи аналізу великих даних

Великі дані (з різних джерел і різних форматів) зберігаються в необробленому вигляді в так званих «озерах» даних (Data lake). Переваги Data lake:

- дані аналізуються в початковому вигляді;
- «озера» менш затратні, ніж сховища структурованих даних;
- використовувати дані зі сховищ можуть одночасно кілька осіб.

Big Data (кілька терабайт, петабайт) можуть бути збережені і систематизовані в так званих розподілених файлових системах.

Для управління розподіленої файлової системою використовуються програмні засоби і стандартне технічне забезпечення.

Система Hadoop, наприклад, призначена для зберігання і управління сховищами даних в діапазоні від декількох терабайт до петабайт. Інформація зберігається на жорстких дисках (їх кількість може досягати тисячі), на стандартних комп'ютерах. Кожну порцію інформації, для збільшення надійності, зберігають декілька разів. Спеціальна карта map відстежує місце зберігання конкретної інформації. Наприклад, про угоди в мережі магазинів.

Ідея, так званого, алгоритму MapReduce: вхідні дані розподіляються на робочі вузли, де вони попередньо обробляються, а потім вони об'єднуються.

Компанія Google вперше почала використовувати MapReduce. Це дозволило розподіляти велику задачу по сотням або тисячам серверів і потім збирати багато їх відповідей в один. Завдяки зростанню попиту Google значно збільшила свій дохід (не менше, ніж на 30 відсотків).

Hadoop – відкрита версія MapReduce – була активно використана іншими компаніями, такими як Amazon, Facebook, Google і NSA і т.д., що

значно знизило вартість обробки даних. Підсумком стала поява платної послуги, що отримала назву хмарні обчислення. Соціальні мережі ефективно існують також завдяки MapReduce і Hadoop. Компанії мають прибуток тільки від показів реклами. Для мільярдів же користувачів Facebook, наприклад, безкоштовний. Методи аналізу, обробки оцифрованої інформації, її подання, а також проектування баз даних зв'язуються з Data science (наукою про дані). В останні роки спостерігається комерціалізація Data science і, як наслідок, поява в світі високооплачуваною професії data scientist.

Великі дані, завдяки високій швидкості їх обробки роблять аналіз самий корінь на відміну від традиційного описового бізнес – аналізу, що важливо при розробці бізнес – стратегій. Крім цього, технології великих даних дозволяють аналізувати значно більшу кількість швидко одержуваних і мінливих відомостей різних типів. Це дає можливість проводити більш глибокі дослідження.

Технології, а також методи виявлення у вихідних даних раніше невідомих закономірностей (data mining), знаходяться на стику штучного інтелекту, статистики та баз даних.

Методи і алгоритми Data Mining призначені для аналізу неструктурованих даних великого обсягу і розмірності. Кореляції і зв'язку встановлюються в процесі використання сучасних методів розпізнавання образів, інших аналітичних технологій (дерева прийняття рішень і класифікації, кластеризацію, нейронномережеві методи і т.д.) [1].

Сфера застосування Data Mining досить широка і включає в себе маркетинг, бізнес, інтернет, промисловість, геологія, медицина, телекомунікації, фармацевтика і т.д.

1.2 Проблема відсутніх даних у таблицях «об'єкт-властивість»

Найчастіше при проведенні соціально-економічних і соціологічних досліджень доводиться стикатися з проблемою обробки пропусків у масивах даних. Традиційними причинами, що призводять до появи пропусків, є неможливість отримання або обробки, спотворення або приховування інформації.

1.2.1 Визначення таблиці «об'єкт-властивість»

Нехай маємо таблицю типу «об'єкт-властивість», яка зображена на рисунку 1.1 [2]. Таблиця містить інформацію про N об'єктів, кожен із яких описується $(1 \times n)$ – рядковими векторами характеристик

$$\underline{x_i} = (x_{i1}, \ldots, x_{ip}, \ldots, x_{ij}, \ldots, x_{in}). \quad (1.1)$$

Припустимо, що N_G. рядки можуть мати одне або кілька відсутніх значень, а $N - N_G$. заповнені повністю. Це не виключає ситуації, коли $N = N_G$., тобто всі вектори містять відсутні значення, кількість яких у кожному рядку, $<n_i,\ i = 1,2,\ldots,N$.

Під час обробки таблиця повинна бути заповнена у відсутніх значеннях, щоб відновлені елементи були в якомусь сенсі найбільш «подібними» або «найближчими» до апріорно невідомих закономірностей, прихованих у цій таблиці.

	1	...	p	...	j	...	n
1	x_{11}	...	x_{1p}	...	x_{1j}	...	x_{1n}
...	...	...	...	...	...	...	...
i	x_{i1}	...	x_{ip}	...	x_{ij}	...	x_{in}
...	...	...	...	...	...	...	...
k	x_{k1}	...	x_{kp}	...	x_{kj}	...	x_{kn}
...	...	...	...	...	...	...	...
N	x_{N1}	...	x_{Np}	...	x_{Nj}	...	x_{Nn}

Рисунок 1.1 – Таблиця «об’єкт-властивість»

1.2.2 Механізми формування пропусків та їх класифікація

Рано чи пізно практично всі дослідники змушені зіткнутися з проблемою пропуску значень в досліджуваних масивах даних. Як досліджуваних масивів даних може виступати найрізноманітніша інформація, наприклад, це можуть бути результати промислових або лабораторних експериментів, показання будь-яких приладів або засобів вимірювань, дані отримані вході опитувань або медичних досліджень.

Пропуски в таких масивах називають загубленими або пропущеними, так як їм відповідають реальні значення, які могли б бути отримані, якби не виникло ситуації, при якій виник пропуск. Виникнення пропусків може носити технічний характер, наприклад, обладнання або засіб вимірювання, з використанням якого отримують дані, може мати низьку якість виконання, дані можуть бути зашумлені або спотворені перешкодами або зовсім можуть бути не отримані через проблеми з лінією зв’язку, по якій здійснюється передача.

У соціальній та економічній сферах, дані можуть бути не отримані в результаті помилок респондентів при проведенні опитувань. В медичній сфері пропуски в даних можуть бути відсутні з найрізноманітніших причин. Описані вище ситуації ведуть до того, що, за місце повноцінного масиву даних, буде отриманий масив, в якому є в наявності зміщення, неповнота і недостовірність даних. В результаті, з таким набором даних неможливо працювати, він непридатний для виконання будь-яких операцій.

Для подальшої обробки та аналізу, потрібно привести в порядок самі дані – необхідно виключити зашумлені дані і відновити втрачені значення, якщо такі є. Внаслідок цього, виникають питання, як враховувати пропуски даних, як їх відновити і заповнити, і як з ними працювати, адже відновлення пропущених значень в масивах даних веде до появі різних колізій, наприклад, зміщення відновлених значень щодо вибірки, зростання помилки обчислення значення пропуску, і т.д. [3].

Внаслідок того, що причини виникнення пропусків можуть бути самими різноманітними, необхідно знати механізм, за допомогою якого був сформований пропуск. Знання механізму формування пропуску дозволить зрозуміти ступінь важливості втрачених даних, що важливо, так як неповні і відновлені дані несуть в собі видозмінену від «оригіналу», нову інформацію.

Виділимо три типи пропусків – повністю випадкові, випадкові і проігноровані пропуски, що відображено на рисунку 1.2.

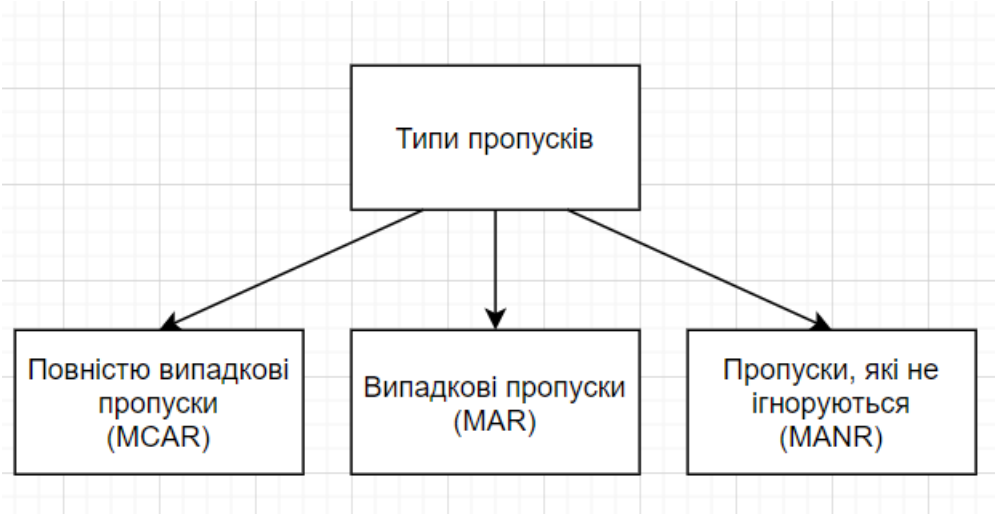


Рисунок 1.2 – Класифікація типів пропусків

Відповідно до представленого вище малюнку, розглянемо класифікацію пропусків в даних докладніше.

Виділяють три типи пропусків, які розрізняються по причин їх виникнення [4]:

1. Повністю випадкові пропуски або MCAR виникають в результаті подій, які призводять до несистематичний відсутності будь-якого конкретного елемента в масиві даних, вони незалежні від спостережуваних параметрів і змінних і відбуваються абсолютно випадково, тобто ймовірність пропуску для кожного елемента в масиві однакова пропуски повинні бути розподілені по всьому масиву даних випадково [5]. На етапі відновлення пропуску елемента в масиві даних, допускається ігнорувати і зраджувати значення пропущеного елемента, так як це не веде до спотворення результату.

2. Випадкові пропуски або MAR виникають, коли пропуски елементів в масивах даних не є випадковими – дані розподілені всередині визначених підгруп змінних, пропуски виникають не випадково, а з-за деяких закономірностей. Імовірність пропуску елемента може бути визначена на основ інший, наявної в наборі інформації, що не містить пропуски. Виняток або заміна пропущеного елемента на деяке значення не веде до істотного спотворення результатів.

3. MNAR, так само даний тип пропусків називають неприйнятним відмовою або відсутністю випадковості. Виникає даний тип пропусків, коли елементи в масиві даних відсутні в залежності від невідомих чинників, пропуски даних систематично пов’язані з їх передбачуваними значеннями [6]. Імовірність пропуску елемента може бути описана на основі інших атрибутів, але інформація по цим атрибутам в наборі даних відсутній, внаслідок цього, ймовірність пропуску неможливо виразити на основі

інформації, що міститься в наборі даних. З математичної точки зору, ці методи виглядають наступним чином:

– MCAR – ймовірність виникнення пропуску не залежить ні від спостережуваних, ні від значень пропущених даних

$$P\left(V_{imissed} \mid X_{missed}, X_{other}\right) = const; \quad (1.2)$$

– MAR – ймовірність пропуску залежить від значень спостережуваних, але не від значень пропущених даних.

$$P\left(V_{imissed} \mid X_{missed}, X_{other}\right) = f(X_{other}); \quad (1.3)$$

– NMAR – ймовірність пропуску залежить від значень спостережуваних і від значень пропущених даних.

$$P\left(V_{imissed} \mid X_{missed}, X_{other}\right) = f(X_{missed}, X_{other}). \quad (1.4)$$

Наявність MCAR можна перевірити, статистично, використовуючи *t*-критерій Стьюдента або критерій x_i-квадрат. Якщо в результаті проведення тесту є незначний рівень критерію, то мають місце повністю випадкові пропуски [7].

Наявність MAR неможливо перевірити статистично і слід покладатися на матеріальну обґрунтованість. За частотою зустрічей, такий розподіл пропусків трапляється частіше, ніж перший тип. Отже, на основі вищесказаного, важливо розуміти яким чином, і з якого джерела були отримані дані.

1.3 Класифікація та огляд методів відновлення даних

Існує багато способів заповнення пропусків (ремонту вибірки) вже після етапу збору даних: заповнення середнім значенням, пропорційне розміщення спостережень з пропущеними даними за тим самим які градаціях шкали, розрахунок можливого значення за допомогою регресійної моделі і так далі. Заповнення пропусків дозволяє не тільки отримати додаткову інформацію (передбачені значення), але і зберегти вже наявну, часто дуже важливу і отриману ціною значних зусиль інформацію, за рахунок збереження спостережень спочатку містили пропуски [8]. Крім очевидних переваг, заповнення, як спосіб вирішення проблеми недостатньої інформації має кілька недоліків, які не можна не враховувати:

– використання для передбачення пропусків наявних повних даних спотворює структуру результуючих даних (після заповнення), яка зміщується в сторону структури тільки повних спостережень;

– штучна підстановка пропусків вносить в масив певну частку штучних даних, які в свою чергу призводять до зміщення значущості одержуваних на їх основі результатів.

Можливо, що, моделі, побудовані по заповненим даними, будуть менш точними в порівнянні з ідеальною моделлю, побудованої тільки на повних спостереженнях. Втрати в їх точності залежатимуть від якості передбачення відсутніх значень. Але втративши в точності, можна виграти в репрезентативності результатів.

При виборі конкретного методу відновлення пропущених значень слід враховувати, що, так як алгоритми заповнення пропусків не універсальні, можливості застосування того чи іншого способу заповнення пропусків залежать від методу аналізу даних, який планується використовувати в подальшому. Найбільш популярні з існуючих методів заповнення, в найбільш повній і докладної класифікації Р. Літла [9], відображені на наступній схемі.

Охарактеризуємо кожну групу методів, зображену на рисунку 1.3.

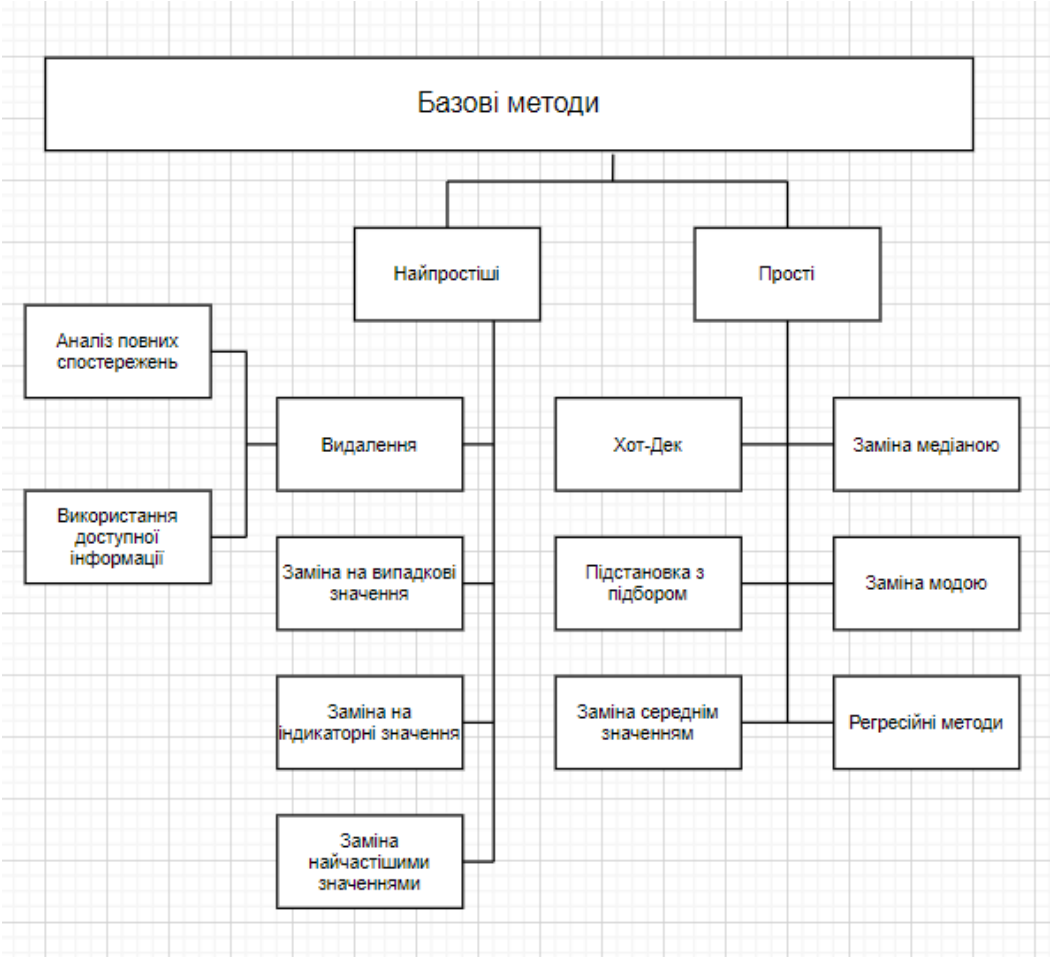


Рисунок 1.3 – Схема класифікації базових методів відновлення пропусків в масивах даних

Найпростіші алгоритми – неітеративноі алгоритми, засновані на простих арифметичних операціях, відстанях між об’єктами, регресійному моделюванні.

До них відноситься заповнення пропусків середнім арифметичних, регресійні моделювання пропусків, метод HotDeck і підбір в групі [10]. Складні алгоритми – ітеративні алгоритми, які передбачають оптимізацію деякого функціоналу, що відображає точність розрахунку підставляється на місце пропуску значень. Їх можна розділити на глобальні та локальні.

Прості алгоритми – в оцінюванні (прогнозі) кожного пропущеного значення беруть участь всі об’єкти даної сукупності: метод Бартлета, EM – оцінювання та Resampling. Локальні алгоритми – в оцінюванні (прогнозі) кожного пропущеного значення беруть участь повні спостереження, що знаходяться в деякій околиці пророкує об’єкта. До даної групи належать

алгоритми Zet і Zet Braid Коротко розглянемо окремі методи, заповнення, що входять до складу перерахованих груп. Заповнення середнім і підбір всередині груп [11].

Метод заповнення середнім значенням, передбачає, що всі пропущені значення замінюються середнім значенням цього показника, розрахованим за наявними даними. Підбір всередині груп передбачає, що вся сукупність об’єктів розбивається на групи за певною ознакою, всередині кожної групи для заповнення пропусків використовуються тільки присутнє в ній значення [12].

Для боротьби з цими двома видами пропусків застосовують вісім основних класів методів [13]:

– аналіз повних спостережень (listwise deletion);

– методи, що використовують доступну інформацію (pairwise deletion);

– підстановка середнього по вибірці (mean substitution);

– метод хот-дек (hot deck);

– регресійний аналіз (regression);

– оцінка за допомогою максимізації правдоподібності (maximum likelihood estimation);

– підстановка за допомогою факторного аналізу (factor analysis substitution);

– модель множинного відновлення даних (multiple imputations method).

Два перших методи широко поширені в дослідницькій практиці.

1.4 Аналіз предметної області

Для реалізації та аналізу методів відновлення даних у таблицях типу «об’єкт-властивість» було обрано предметну область маркетингових

досліджень. Це зумовлено тим, що дана сфера має велику актуальність з розвитком реклами у соціальних мережах, особливо на просторі України. Для реалізації маркетингових досліджень проводяться анкетування та соціальні опитування, які частіш за все і мають пропуски у даних або некоректні дані.

1.4.1 Великі дані у маркетингових дослідженнях

Аналіз великих даних знаходить своє застосування в маркетингу для прогнозування попиту, виявлення особливостей поведінки споживачів, їх сегментування, розробки маркетингових і, зокрема, комунікативних стратегій. Зіставляючи такі фактори як сезонність, географію запитів на конкретний продукт можна спрогнозувати сезонний попит і розробити план по розподілу бюджету реклами на конкретний період в конкретному регіоні [14]. При розробці рекламної кампанії необхідно визначити цільову аудиторію і розробити контент рекламного звернення.

Для виявлення цільової аудиторії можна спиратися як на власні критерії, так і на дані рекламодавця про відвідувачів своїх сайтів (промо-сайту, відповідних сторінок основного сайту). Для пошуку схожою аудиторії використовуються Інтелектуальне алгоритми по навчається вибірці, так званий Look-a-like Яндекс запустив цей особливий вид таргетингу Look-a-like ще в 2013 році [15]. Look-a-like націлює рекламу тієї аудиторії, яка за своїми характеристиками і поведінкою схожа на необхідну, цільову аудиторію.

Для розуміння «схожості» потрібно організувати велику вибірку, щоб більш точно задати характеристики про поведінку своїх клієнтів. Збирається інформація про тих клієнтів, які зробили певну дію: скачали прайс-лист або каталог, перейшли по посиланню, замовили і оплатили товар і так далі [16]. При невеликих обсягах власної вибірки додаткову інформацію можна купити. Наприклад, платформи Visual DNA, Weborama надають різні дані в

тому числі, про записах користувачів в соціальних мережах, їх останніх покупках. Щоб підвищити ефективність Look-a-like корисно мати не менше десятків тисяч власних записів і мати постачальників додаткової інформації, які реально готові цією інформацією ділитися. Ефект Look-a-like в соціальних мережах, як свідчать практики, буде набагато менше в зв'язку з невеликим охопленням аудиторії [17]. Для націлювання змісту рекламного звернення на конкретного користувача використовується персоналізація email реклами, товарні рекомендації на сайті, динамічний ретаргетінг.

Суть динамічного ретаргетінга полягає в тому, що реклама дається тільки тим учасникам спільноти, які вступили туди недавно. Це пов'язано з особливістю індивідуальних інтересів учасників, які і враховує реклама. якщо учасник покине співтовариство, то продовження реклами втратить сенс. Застосування Big Data, розвиток поведінкового таргетування привели до появі технології RTB-аукціону – Real Time Bidding. Відповідно до цієї технологією продаж і купівля рекламних показів відбувається на основі аукціону. Сайт рекламодавця прив'язаний до певного RTB-агентству. Користувач вводить в рядку браузера цікавить його запит [18]. Сайт відправляє запит на перегляд рекламного оголошення і дані користувача для таргетування (стать, вік, інтереси) в RTB агентство. Запит класифікується за рядом параметрів (дані про майданчику, на яку здійснений вхід, часу входу, даних про клієнта і т.д). Інформація про клієнта, якому показується реклама, визначається на основі даних cookie і його інтернет історії [19].

Великі дані не замінюють дослідження, але вони є інформаційної основою для проведення більш глибоких і коротких опитувань. Соціальні мережі є джерелами великої кількості маркетингової інформації і, перш за все, про користувачів, включаючи як їх ідентифікаційні дані, так і різні додаткові дані (інтересах, захоплення, спосіб проведення дозвілля, рівень освіти, друзях, покупках і т.д. Розвиток технології великих даних створили нові можливості для інтегрування і збагачення різних відомостей про користувачів соціальних мереж.

1.4.2 Проблеми аналізу великих даних у маркетингових дослідженнях

По-перше, це інтеграція даних, зібраних раніше. Для маркетолога важливо мати доступ до даних по клієнтах, проведеним кампаніям, маркетингових досліджень минулих періодів [20]. Без них, часто, неможливо вибудувати тренд, зрозуміти специфіку поведінки споживача на ринку.

По-друге системи зберігання таких даних ніколи не призначалися для використання їх в режимі реального часу.

Всі фахівці в області Big Data сходяться на думці, що технології Big Data безглузді, якщо серверні системи не можуть підтримувати транзакції в реальному часі [21].

По-третє, системи збору і обробки даних в сучасних великих компаніях нагадують справжній зоопарк. Організації збирали дані для різних цілей, різними способами, не часто інтегрували системи збору один з одним. Ніхто не уявляв собі, що коли-небудь потрібно буде взаємодіяти з абсолютно непов’язаними системами і сховищами даних, одночасно всередині і за межами підприємства, і для аналізу, і для візуалізації.

Навіть коли технологія може забезпечити рішення для інтеграції і взаємодії, власники бізнесу неохоче відмовляються від контролю або вимагають від IT персонал виставляти пріоритети по проектам на основі поточних інтересів бізнесу [22].

Також, для якісної реалізації технологій Big Data необхідно постійно пам’ятати, що деякі види даних, включаючи фінансові та медичні записи, підлягають захисту і значного регулювання, яке може варіюватися в залежності від географії та юрисдикції. Ці правила можуть утруднити чи зробити неможливим використання деяких даних [23]. Усі ці проблеми провокують появу пропусків у таблицях за даними, що ускладнюють процес їх аналізу.

1.5 Постановка задачі дослідження

Таким чином, відновлення даних у таблицях «об’єкт-властивість» є актуальним завданням у сфері аналізу даних маркетингових досліджень. Тому ставиться завдання розробки методів відновлення даних у таких таблицях, їх огляд та порівняння.

Об’єктом дослідження є набір даних з маркетинговими дослідженнями покупців інтернет-магазинів, отриманий шляхом соціального опитування на краудсорсингових платформах.

Метою дослідження є розробка методів FCM та багатовимірної нечіткої екстраполяції для кластеризації даних з пропусками.

Для цього необхідно вирішити такі завдання:

– провести аналіз причин появлення пропусків у таблицях;

– провести аналіз існуючих методів відновлення даних;

– розробити опитування для маркетингових досліджень з метою отримання учбового набору даних;

– проаналізувати та обрати середовище розробки та мову програмування;

– реалізувати методи відновлення даних;

– основними методами для порівняння є методи FCM та MFE;

– порівняти результати роботи розроблених методів.

2 МАТЕМАТИЧНІ МОДЕЛІ МЕТОДІВ ВІДНОВЛЕННЯ ДАНИХ

2.1 Метод регресійного аналізу

Заповнення пропусків з використанням регресії (вставка умовного середнього).

Термін «регресія» з’явився в кінці 19-го століття, подальшої розвиток механізм отримав в роботі [24]. У літературі присвяченій статистикою регресію ділять на 3 види – по розмірності, по лінійності і з параметричного ознакою. Згідно, застосування регресійних моделей в задачах відновлення пропусків ділять на два етапи: побудова регресійної моделі за повними спостереженнями і оцінка коефіцієнтів в рівнянні; підстановка в отримане рівняння відомих значень [25]. Методи регресійного аналізу використовують для дослідження впливу однієї або декількох незалежних змінних на одну залежну. Причому, для використання методів даної групи потрібно складання регресійних рівнянь, причому якщо має місце велика кількість пропусків, потрібно складати унікальні рівняння. У загальному випадку, відбувається заміна пропуску на прогнозовані оцінки з рівняння регресії. Через кореляції змінних, відновлюване значення має бути спрогнозовано з використанням всієї інформації спостережуваних даних [26]. До методів відновлення пропусків на основі регресії відносять метод заповнення умовним середнім, або метод Бака.

Метод більш перспективний по порівняно з розглянутими раніше методами. Механізм роботи методу полягає в оцінці середнього і коваріаційної матриці по повним спостереженнями, після чого, отримані оцінки використовуються для обчислення лінійної регресії відсутніх змінних за наявними змінним для кожного спостереження [27]. Підстановка спостережуваних оцінок до відповідних рівняння регресії генерує передбачені значення для неповних змінних, і ці передбачені оцінки підставляються на місце пропуску, таким чином, роблячи неповний набір

даних повним набором. Застосування оператора згортки дозволяє частково автоматизувати процес складання унікальних рівнянь регресії. Заповнення пропусків методом Бака забезпечує розумні оцінки середніх.

Метод багатовимірної регресії схожий за своєю суттю з методом простої регресії, і лише поширює її принципи на *n*-мірна, що трохи складніше в реалізації. Отримання регресійних коефіцієнтів для кожного з передбачуваних елементів відбувається шляхом застосування методу найменших квадратів до вибірці з повними даними [28]. Прогноз пропущеного показника буде отримано при підстановці отриманого значення в регресійне рівняння. Проблеми методу полягають в тому, що в деяких ситуаціях пропуски можуть бути як у змінній, яку потрібно відновити, так і в змінної, дані якої беруться для відновлення, через це пророцтво на основі коефіцієнтів регресійного рівняння не працюватиме – воно буде неможливо. Крім цього, змінні повинні корелювати з поточною «робочої» змінної і відмінно пояснювати її варіацію, а у вибірці, що використовується для передбачень, може просто не вистачити елементів .

Величина зміщення в відхиленнях і коваріації є передбачуваною, і відповідно до джерел, є коригувальні поправки для цих параметрів. Метод багатовимірної регресії дозволяє отримати правдоподібно заповнені пропуски в даних, проте реальні дані мають певний розкид в своїх значеннях, який при заповненні методами на основі лінійної регресії не враховується – варіація значень характеристики стає менше, а кореляція між характеристиками штучно посилюється.

Отже, чим вище варіація значень характеристики і чим більше пропусків необхідно заповнити – тим більше погіршується результат. Проблему, що виникає при застосуванні багатовимірної регресії, дозволяє вирішити використання методу стохастичною регресії. Метод також використовує рівняння регресії і заснований на заміні пропуску значенням, підставляє під час заповнення по регресії, в сумі з залишком, що відображає невизначеність пророкує значення. Додавання залишків до вставляються

значенням відновлює втрачену мінливість даних і ефективно усуває зміщення, пов’язані зі стандартними схемами [29].

2.2 Метод Барлетта

Метод, запропонований Бартлеттом для вирішення даної проблеми (1937 р.) [30], полягає в підстановці початкових значень замість пропусків і проведенні коваріаційного аналізу з супутньої змінної пропусків для кожного пропущеного значення. Припустимо, що кожен пропуск y_i заповнюється початковим значенням, щоб вектор значень Y був повний. Позначимо початкові значення $\tilde{y}_i$, $i = 1, \ldots, m_0$. Нехай Z – $n \times m$ – матриця m_0 супутніх змінних пропусків [31]. За визначенням *i*-та супутня змінна пропусків – це індикатор *i*-го пропущеного значення, тобто завжди 0, за винятком випадку, коли пропущено *i*-те значення, тоді вона дорівнює 1. Перший рядок $Z, z1,$ дорівнює (1,0, ..., 0), ..., рядок m_0 дорівнює (0, ..., 0,1), а z_i при $i > m_0$ рівні (0, ..., 0), так як вони відповідають присутнім y_i. При коваріаційного аналізу використовується і X, і Z для передбачення Y. Припустимо, що для вихідної змінної $Y = (y_1, \ldots, y_n)^T$ вірна лінійна модель:

$$Y = X_\beta + Z_\gamma + e, \tag{2.1}$$

де γ – вектор-стовпець з m_0 коефіцієнтів регресії для супутніх змінних пропусків,

$$e = (e_1, \ldots e_n)^T, \tag{2.2}$$

де e_i – незалежно і однаково розподілені з нульовим середнім і однаковою дисперсією σ^2;

β – оцінюваний параметр – вектор довжини p.

Класична оцінка найменших квадратів β дорівнює:

$$\hat{\beta} = (X^T X)^{-1} X^T Y, \quad (2.3)$$

якщо $(X^T X)$ має повний ранг. А якщо $(X^T X)$ невирождени, то $\hat{\beta}$ – незміщенна оцінка β з мінімальною дисперсією. Якщо e_i розподілені нормально, то $\hat{\beta}$ – оцінка максимальної правдоподібності, розподілена нормально з середнім β і дисперсією $\sigma^2 (X^T X)^{-1}$.

Метод має наступні переваги:

– він неітеративному, і, отже, знімає питання про збіжність. Якщо структура пропусків має виродження (наприклад, в тому випадку, коли не можна оцінити деякі параметри, як при відсутності всіх значень для якоїсь обробки), цей метод «попереджає» дослідника, тоді як ітеративні методи призводять до відповіді, можливо, неприпустимого;

– метод дає правильні оцінки і залишкові суми квадратів, а також вірні стандартні помилки, суми квадратів і *F*-критерії. Хоча цей метод привабливий в певних відносинах, його часто можна реалізувати безпосередньо, тому що спеціалізовані програми дисперсійного аналізу можуть не мати можливість вести обробку при багатьох супутніх змінних [31].

2.3 EM-алгоритм

Метод максимізації очікувань (expectation maximization), в деяких джерелах так само званий EM-оцінюванням, дозволяє не тільки відновлювати пропущені значення з використанням двоетапного ітеративного алгоритму, а й оцінювати середні значення, коваріаційні і кореляційні матриці для кількісних змінних. EM-алгоритм, в

найзагальнішому сенсі являє собою ітераційну процедуру, призначену для вирішення завдань оптимізації деякого функціоналу, через аналітичний пошук екстремуму цільової функції [32].

В основі ідеї ЕМ-алгоритму лежить припущення, що досліджувана множина даних може бути змодельоване за допомогою лінійної комбінації багатовимірних нормальних розподілів, а метою є оцінка параметрів розподілу, які максимізують логарифмічну функцію правдоподібності, використовувану в якості запобіжного якості моделі. Іншими словами, передбачається, що дані в кожному кластері підкоряються певним законом розподілу, а саме, нормальному розподілу. З урахуванням цього припущення можна визначити параметри – математичне сподівання і дисперсію, які відповідають закону розподілу елементів в кластері, найкращим чином «невластивому» до спостережуваних даними.

Цей алгоритм реалізується в 2 етапи. Перші букви назв, яких утворюють загальну абревіатуру алгоритму [33].

Таким чином, ми припускаємо, що будь-який спостереження належить до всіх кластерів, але з різною ймовірністю. Тоді завдання полягатиме в «підгонці» розподілів суміші до даних, а потім у визначенні ймовірностей приналежності спостереження до кожного кластеру. Очевидно, що спостереження повинно бути віднесено до того кластеру, для якого ця можливість вище.

Серед переваг ЕМ-алгоритму можна виділити наступні:

- потужна статистична основа;
- лінійне збільшення складності при зростанні обсягу даних;
- стійкість до шумів і перепустками в даних;
- можливість побудови бажаного числа кластерів;
- швидка збіжність при вдалій ініціалізації.

Однак алгоритм має і ряд недоліків. По-перше, припущення про нормальність всіх вимірювань даних не завжди виконується. По-друге, при невдалій ініціалізації збіжність алгоритму може виявитися досить повільним.

Крім цього, алгоритм може зупинитися в локальному мінімумі і дати квазіоптимальне рішення [34].

Алгоритм EM заснований на обчисленні відстаней. Він може розглядатися як узагальнення кластеризації на основі аналізу суміші імовірнісних розподілів. У процесі роботи алгоритму відбувається ітеративне поліпшення рішення, а зупинка здійснюється в момент, коли досягається необхідний рівень точності моделі. Мірою в даному випадку є монотонно збільшується статистична величина, яка називається логарифмічним правдоподібністю. Метою алгоритму є оцінка середніх значень *C*, коваріаційний *R* і ваг суміші *W* для функції розподілу ймовірності [35].

2.4 ZET алгоритм

Основні етапи алгоритму:

– розбиваємо багатовимірні дані на двомірні так, щоб між ними була залежність;

– визначаємо таксони на двомірних даних;

– визначаємо стійкий таксон на основі запропонованої емпіричної гіпотези: в середньому об’єкти (дані) з більшою ймовірністю появи в досліджуваному Таксоні в різні досліджувані інтервали несуть більшу кількість інформації;

– визначаємо, до якого таксону належить елемент, що стоїть на місці пропуску, на основі аналізу інших факторів. Пропуск будемо відносити до того таксону, до якого відносяться елементи, що потрапили в стійкий таксон у відновлюваному факторі [36].

– проводимо відновлення пропуску за допомогою лінійної регресії $y_i(x) = b + a * x$ за значеннями, що потрапили в стійкий таксон по іншому фактору:

$$a = \frac{\sum_{i=1}^{n} x_i \sum_{i=1}^{n} y_i - n \sum_{i=1}^{n} x_i y_i}{\left(\sum_{i=1}^{n} x_x\right)^2 - n \sum_{i=1}^{n} x_i^2}, \quad (2.4)$$

$$b = \frac{1}{n} \left(\sum_{i=1}^{n} y_i - a \sum_{i=1}^{n} x_i\right); \quad (2.5)$$

– визначаємо ступінь впливу однієї ознаки на інший (k) на основі побудованого нечіткого графа ієрархії з кореневими вершинами на даному полі даних;

– відновлюємо пропуск на основі врахування ступеня впливу кожної ознаки, що потрапив в стійкий таксон:

$$– \quad y(x) = k_1 y_1(x) + k_2 y_2(x) + \ldots + k_i y_i(x). \quad (2.6)$$

2.5 Багатовимірна нечітка екстраполяція

Представимо таблицю з рисунку 1.1 у вигляді $(N \times n)$ – матриця X, в якій відсутній в найпростішому випадку один елемент x_{ip} або в загальному випадку $\sum_{i=1}^{N} n_i$ елементів [37]. Усі дані заздалегідь відцентровані та нормалізовані за ознаками, тому вони усі належать $[-1,1]^n$. Ці дані формують масив:

$$\tilde{X} = \{\widetilde{x_1}, \ldots, \widetilde{x_k}, \ldots, \widetilde{x_N}\} \subset R^n, \quad (2.7)$$

$$\widetilde{x_k} = (\widetilde{x_{k1}}, \ldots, \widetilde{x_{kl}}, \ldots, \widetilde{x_{kn}})^T, \quad (2.8)$$

де $-1 \le \widetilde{x_{kl}} \le 1,\ 1 < m < N,\ 1 \le q \le m,\ 1 \le i \le n,\ 1 \le k \le N.$

Для кожного рядку $\underline{x_i}$, що містить пропуск, необхідно розрахувати відстань між самим пропуском та іншими рядками за формулою:

$$D_P^2\left(\underline{x_i},\underline{x_k}\right)=\frac{n}{n_i+n_k-n_{ik}}\sum_{j=1}^{n}\left(x_{ij}-x_{kj}\right)^2\delta_j\,, \tag{2.9}$$

де $\delta_j=\begin{cases}0, \text{якщо } \underline{x_i} \text{ або } \underline{x_k} \text{ в позиції } j \text{ містять пропущені значення}\\ 1, \text{в іншому випадку}\end{cases}$,

n_{ik} – загальна кількість пропущенних значень і однакових позиціях $\underline{x_i}$ і $\underline{x_k}$.

У цьому випадку не розглядається $\underline{x_k}$ для якого $\sum_{j=1}^{n}\delta_j=0$. Упорядкуємо $\tilde{N}\le N-1$ розраховані відстані так, що:

$$0\le D_p^{2[\min]}=D_p^{2[1]}<D_p^{2[2]}<\dots<D_p^{2[N]}\le 4n\,, \tag{2.10}$$

де індекс у квадратних дужках означає ранг.

Залишимо лише ті значення, що задовільняють

$$\frac{D_p^{2[I]}}{4n}\le\varepsilon, I=1,\dots,N\,, \tag{2.11}$$

де $0<\varepsilon<1$.

Використовуючи концепцію стандартного методу FCM [38] розрахуємо

$$U_i(i)=\frac{D_p^{2[I]}}{\sum_{q=1}^{N}D_p^{2[q]}}, I=1,\dots N\,. \tag{2.12}$$

Якщо $D_P^{2[1]}=0$, то ми вважаємо, що $U_1(i)=1$. Тоді кожний вектор $\underline{x_i}$ може бути наближен за формулою

$$\underline{x_i}=\sum_{I=1}^{N}U_I(i)\underline{x_I}. \tag{2.13}$$

Зазначимо також, що даний метод використовується як наближення типу, але замість рівнів членства $U_I(i)$ використовуються ваги, отримані при

вирішенні задачі оптимізації, яка не завжди є розв'язною. І нарешті, останній етап – заповнення відсутніх значень. Неважко помітити, що оцінка відсутнього елемента x_{ip} може бути записана як

$$x_{ip} = \sum_{I=1}^{N} U_I(i) x_{Ip}. \quad (2.14)$$

2.6 FCM

Алгоритм нечіткої кластеризації називають FCM-алгоритмом (Fuzzy Classifier Means, Fuzzy C-Means). Метою FCM-алгоритму кластеризації є автоматична класифікація безлічі об'єктів, які задаються векторами ознак в просторі ознак. Іншими словами, такий алгоритм визначає кластери і відповідно класифікує об'єкти. Кластери є нечіткими множинами, і, крім того, кордони між кластерами також є нечіткими.

FCM-алгоритм кластеризації передбачає, що об'єкти належать всім кластерам з певною вірогідністю. Ступінь приналежності визначається відстанню від об'єкта до відповідних кластерних центрів. Даний алгоритм ітераційно обчислює центри кластерів і нові ступені приналежності об'єктів [38].

Для заданої множини K вхідних векторів x_k і N виділених кластерів c_j передбачається, що будь-який x_k належить будь-якому c_j з приналежністю μ_{jk} інтервалу [0,1], де j – номер кластера, а k – номер вхідного вектора.

Беруться до уваги наступні умови нормування для μ_{jk}:

$$\sum_{j=1}^{N} \mu_{jk} = 1;\ \forall k = 1, \dots, K;\ 0 < \sum_{j=1}^{N} \mu_{jk} \le K;\ \forall j = 1, \dots, N. \quad (2.15)$$

Мета алгоритму – мінімізація суми всіх зважених відстаней $||x_k - c_j||$:

$$\sum_{j=1}^{N}\sum_{k=1}^{K}(\mu_{jk})^{q}\,||x_k - c_j|| \to min, \tag{2.16}$$

де q – фіксований параметр, що задається перед ітераціями.

Для досягнення вищевказаної мети необхідно вирішити наступну систему рівнянь:

$$\begin{cases} {}^{d}/_{d\mu_{jk}}\left(\sum_{j=1}^{N}\sum_{k=1}^{K}(\mu_{jk})^{q}\,||x_k - c_j||\right) = 0, \\ {}^{d}/_{dc_j}\left(\sum_{j=1}^{N}\sum_{k=1}^{K}(\mu_{jk})^{q}\,||x_k - c_j||\right) = 0. \end{cases} \tag{2.17}$$

Спільно з умовами нормування μ_{jk} дана система диференційних рівнянь має наступне рішення:

$$c_j = \frac{\sum_{j=1}^{N}(\mu_{jk})^{q}x_k}{\sum_{k=1}^{K}(\mu_{jk})^{q}}, \tag{2.18}$$

зважений центр гравітації і

$$\mu_{jk} = \frac{{}^{1}/_{||x_k - c_j||^{1/(q-1)}}}{\sum_{j=1}^{N}\left({}^{1}/_{||x_k - c_j||^{1/(q-1)}}\right)}. \tag{2.19}$$

Алгоритм нечіткої кластеризації виконується крок за кроком.

Крок 1 – ініціалізація. Обираються наступні параметри:

– необхідну кількість кластерів N, $2 < N < K$;

– міра відстаней, як Евклідова відстань;

– фіксований параметр q;

– початкова (на нульовій ітерації) матриця приналежності об’єктів x_k з урахуванням заданих початкових центрів кластерів c_j.

Крок 2 – регулювання позицій центрів кластерів. На ітерації t при відомій матриці обчислюється відповідно до викладеного вище рішенням системи диференціальних рівнянь.

Крок 3 – коригування значень приналежності μ_{jk}. З огляду на відомі, обчислюються, якщо, в іншому випадку:

$$\mu_{jk}^{(t+1)} = \begin{cases} 1, l = j, \\ 0, \text{в інших випадках.} \end{cases} \quad (2.20)$$

Крок 4 – зупинка алгоритму. Алгоритм нечіткої кластеризації зупиняється при виконанні наступної умови:

$$||U^{(t+1)} - U^{(t)}|| \leq \varepsilon, \quad (2.21)$$

де $||U^{(t+1)} - U^{(t)}||$ – матрична норма (наприклад, Евклідова норма), рівень точності задається заздалегідь [39].

3 ПРОГРАМНА РЕАЛІЗАЦІЯ МЕТОДІВ ВІДНОВЛЕННЯ ДАНИХ

3.1 Обґрунтування вибору середовища програмної реалізації

У рамках атестаційної роботи були розроблені метод відновлення даних у таблицях типу «об’єкт-властивість» та метод кластеризації, а саме метод багатовимірної нечіткої екстраполяції та FCM. Для реалізації були обрані мова програмування C# та середовище розробки Microsoft Visual Studio 2019.

3.1.1 Мова програмування C#

Мова C#, розроблена компанією Майкрософт, одна з найпопулярніших сучасних мов програмування. Вона має популярність на ринку розробки в різних країнах. C# застосовують при роботі з програмами для ПК, створення складних веб-сервісів або мобільних додатків. З’явилась як мова для власних потреб платформи Microsoft .NET. Поступово ця мова стала дуже популярною.

Розробка мови почалася в 1998 році, а перша версія побачила світ в 2001 році. Групою розробників керував відомий в професійних колах фахівець Андерс Хейлсберг. Нові версії C# виходять порівняно часто, а поточні доопрацювання, виправлення помилок і розширення бібліотек ведеться практично на постійній основі [40].

В результаті мова вийшла вкрай гнучкою, потужною і універсальною. На ній пишуть практично все, що завгодно, від невеликих веб-додатків до потужних програмних систем, які об’єднують в собі веб-структури, додатки для десктопів і мобільних пристроїв. Все це стало можливим завдяки зручному C-подібному синтаксису, суворому структуруванню, величезній кількості фреймворків і бібліотек (їх число досягає декількох сотень) [41].

Довгий час платформа .NET поставлялася з закритим ядром, що створювало певні труднощі в розробці і знижувало популярність C# в професійному середовищі. Але в листопаді 2014 року Майкрософт радикально змінила підхід і стала видавати безкоштовні ліцензії для Visual Studio вже з відкритим вихідним кодом для всіх наборів інструментів.

C# – дійсно цікавий інструмент, гідний уваги. Він впевнено займає високі позиції в рейтингах затребуваних мов програмування на ринку праці.

Компанія Microsoft приділяє значну увагу підтримці мови розробки, а тому регулярно з'являються оновлення та доповнення, виправляються виявлені помилки в компіляторі, розширюються бібліотеки. Розробники зацікавлені в популяризації інструменту і докладають до цього масу зусиль.

Також, розробники надають докладну і розгорнуту документацію на своїх офіційних ресурсах. Крім того, відповіді практично на будь-які питання, пов'язані з роботою в C#, можна знайти в мережі Інтернет. Популярність мови привела до появи безлічі професійних співтовариств, присвячених C#. Існує безліч підручників, курсів для новачків і середніх розробників, відео добірок та інших навчальних матеріалів [42].

Інструментарій C# дозволяє вирішувати широке коло завдань, мова дійсно дуже потужна і універсальна. На ній розробляють:

- додатки для WEB;
- різні ігрові програми;
- додатки платформ Андроїд або iOS;
- програми для Windows.

Перелік можливостей розробки практично не має обмежень завдяки найширшому набору інструментів і засобів. Звичайно, все це можна реалізувати за допомогою інших мов, але деякі з них вузькоспеціалізовані, в інших доведеться використовувати додаткові інструменти сторонніх розробників. У C# рішення широкого кола завдань можлива швидше, простіше і з меншими витратами часу і ресурсів.

Мова дозволяє в автоматичному режимі очистити пам’ять від об’єктів, які не використовуються, або знищених додатків.

За допомогою цього інструменту можна легко виявляти і обробляти помилки в коді. Спосіб є структурованим з широким набором функцій. При цьому важливо не зловживати можливостями роботи з винятками, так як при неправильному використанні з’являється ризик появи багів.

У мові прийнята загальна система роботи з типами, починаючи від примітивів і закінчуючи складними, в тому числі, призначеними для користувача наборами. Застосовується єдиний набір операцій для обробки і зберігання значень типізації. Також можна використовувати посилальні типи користувача, що дозволить динамічно виділити пам’ять під об’єкт або зберігати спрощену структуру в мережі.

Мова програмування забороняє звернення до змінних, що не були ініційовані, що виключає можливість виконання безконтрольного приведення типів або виходу за межі певного масиву даних.

Дуже цікава особливість C# – контроль версій. Суть в тому, що багато мов не приділяють належної уваги цьому питанню, і програми нерідко перестають коректно працювати при переході на нову версію продукту. У C# це було виправлено [43].

Для використання архітектури платформи на C # необхідно встановити і налаштувати платформу .NET Framework. Вона поставляється повністю безкоштовно, застосовується вкрай широко, а тому проблем з користувацькими пристроями зазвичай не виникає. Платформа вбудована в інсталяційний пакет Windows, при необхідності її також можна скачати і поставити окремо. Існують версії для Linux і MAC.

В рамках платформи до обробки виконуваного коду підключається середу CLR – єдиний об’єднаний набір бібліотек і класів, який був розроблений Майкрософт і є реалізацією світового стандарту Common Language Infrastructure (CLI) [44].

Після роботи компілятора текст програми перекладається в проміжний мову IL, який «розуміє» CLI. Працює це так. IL і всі необхідні ресурси, включаючи рядки і малюнки формату bmp, зберігаються на жорсткий диск у вигляді виконуваного файлу dll або exe. З таких файлів з проміжним кодом формується збірка додатки, яка включає в себе опис з повною інформацією про всі важливі параметри роботи.

Безпосередньо при виконанні програми CLR звертається до збірки і виробляє дії в залежності від отриманих відомостей. Якщо код написаний правильно і проходить перевірку безпеки системи, проводиться компіляція з IL в інструкції в машинні команди [45]. Середа CLR попутно виконує ще багато побічних функцій:

- видалення програмного сміття;
- робота з винятками;
- розподіл ресурсів;
- контроль типізації;
- управління версіями;
- типізація;
- управління версіями.

В результаті код C # вважається керованим, тобто він компілюється в двійковий вид на призначеному для користувача пристрої з урахуванням особливостей встановленої системи.

C# протягом довгого часу впевнено лідирує в рейтингу кращих і найбільш затребуваних на ринку розробки мов. Спочатку їм зацікавилися тільки розробники, які пишуть програми під Windows. Але в процесі розвитку C# навчився працювати на Mac, Linux, IoS і Android. А після того, як код платформи відкрили для всіх бажаючих, були зняті практично всі можливі обмеження в застосуванні C#. В результаті мова активно розвивається, застосовується все ширше.

3.1.2 Середовище розробки MS Visual Studio

Microsoft Visual Studio – це програмна среда розробки додатків для ОС Windows, як консольних, так і з графічним інтерфейсом.

У комплект входять наступні основні компоненти:

– Visual Basic.NET – для розробки додатків на VisualBasic;

– Visual C ++ – на традиційній мові C ++;

– Visual C # – наязике C # (Microsoft);

– Visual F # – на F # (Microsoft Developer Division).

Функціональна структура середовища включає в себе:

– редактор вихідного коду, який включає безліч додаткових функцій, як автодоповнення IntelliSense, рефракторінг коду і т. д.;

– відладчик коду;

– редактор форм, призначений для спрощеного конструювання графічних інтерфейсів;

– веб-редактор;

– дизайнер класів;

– дізайнерсхем баз даних.

Visual Studio також дозволяє створювати і підключати сторонні додатки (плагіни) для розширення функціональності практично на кожному рівні, включаючи додавання підтримки систем контролю версій вихідного коду (Subversion і VisualSourceSafe), додавання нових наборів інструментів (для редагування і візуального проектування коду на предметно-орієнтованих мовах програмування або інструментів для інших аспектів процесу розробки програмного забезпечення) [46].

Переваги й недоліки описані нижче.

Інтегроване середовище розробки (Integrated Development Environment – IDE) Visual Studio пропонує ряд високорівневих функціональних можливостей, які виходять за рамки базового управління кодом.

Вбудований Web-сервер. Для обслуговування Web-додатку ASP.NET необхідний Web-сервер, який буде очікувати Web-запити і обробляти відповідні сторінки. Наявність в Visual Studio інтегрованого Web-сервера дозволяє запускати Web-сайт прямо з середовища проектування, а також підвищує безпеку, виключаючи ймовірність отримання доступу до тестового Web-сайту з якого-небудь зовнішнього комп'ютера, оскільки тестовий сервер може приймати з'єднання лише з локального комп'ютера [47].

Підтримка безлічі мов при розробці. Visual Studio дозволяє писати код своєю рідною мовою чи будь-якими іншими бажаними мовами, використовуючи весь час один і той же інтерфейс (IDE). Більш того, Visual Studio також ще дозволяє створювати Web-сторінки на різних мовах, але поміщати їх все в один і той же Web-додаток. Єдиним обмеженням є те, що в кожної Web-сторінці можна використовувати тільки якусь одну мову (очевидно, що в іншому випадку проблем при компіляції було б просто не уникнути).

Менше коду для написання. Для створення більшості додатків потрібно пристойну кількість стандартного стереотипного коду, і Web-сторінки ASP. NET тому не виключення. Наприклад, додавання Web-елемента управління, приєднання обробників подій і коригування форматування вимагає установки в розмітці сторінки ряду деталей. У Visual Studio такі деталі встановлюються автоматично.

Інтуїтивний стиль кодування. За замовчуванням Visual Studio форматує код у міру його введення, автоматично вставляючи необхідні відступи і застосовуючи колірне кодування для виділення елементів типу коментарів. Такі незначні відмінності роблять код більш зручним для читання і менш схильним до помилок. Застосовувані Visual Studio автоматично параметри

форматування можна навіть налаштовувати, що дуже зручно у випадках, коли розробник вважає за краще інший стиль розміщення дужок.

Більш висока швидкість розробки. Багато з функціональних можливостей Visual Studio спрямовані на те, щоб допомагати розробнику робити свою роботу якомога швидше. Зручні функції, на зразок функції IntelliSense (яка вміє перехоплювати помилки і пропонувати правильні варіанти), функції пошуку і заміни (яка дозволяє відшукувати ключові слова як в одному файлі, так і в усьому проекті) і функції автоматичного додавання і видалення коментарів (яка може тимчасово приховувати блоки коду), дозволяють розробнику працювати швидко і ефективно.

Можливості налагодження. Пропоновані в Visual Studio інструменти налагодження є найкращим засобом для відстеження загадкових помилок і діагностування дивної поведінки. Розробник може виконувати свій код по рядку за раз, встановлювати інтелектуальні точки переривання, при бажанні зберігаючи їх для використання в майбутньому, і в будь-який час переглядати поточну інформацію з пам’яті.

Visual Studio також має і безліч інших функцій: можливість управління проектом. Вбудована функція управління вихідним кодом, можливість рефакторізації коду, потужна модель розширюваності. Більш того, в разі використання Visual Studio 2008 Team System розробник отримує розширені можливості для модульного тестування, спільної роботи і управління версіями коду (що значно більше того, що пропонується в більш простих інструментах на кшталт Visual SourceSafe).

Як недолік можна відзначити неможливість відладчика (Microsoft Visual Studio Debugger) відстежувати в коді режиму ядра. Налагодження в Windows в режимі ядра в загальному випадку виконується при використанні WinDbg, KD або SoftICE.

3.2 Програмна реалізація

За допомогою мови програмування C# та середовища розробки Microsoft Visual Studio 2019 розроблено програму, що кластеризує повні дані за допомогою алгоритму FCM та заповню дані з пропусками й проводить кластеризацію методом багатовимірної нечіткої екстраполяції.

Розглянемо структуру проекту, зображену на рисунку 3.1.

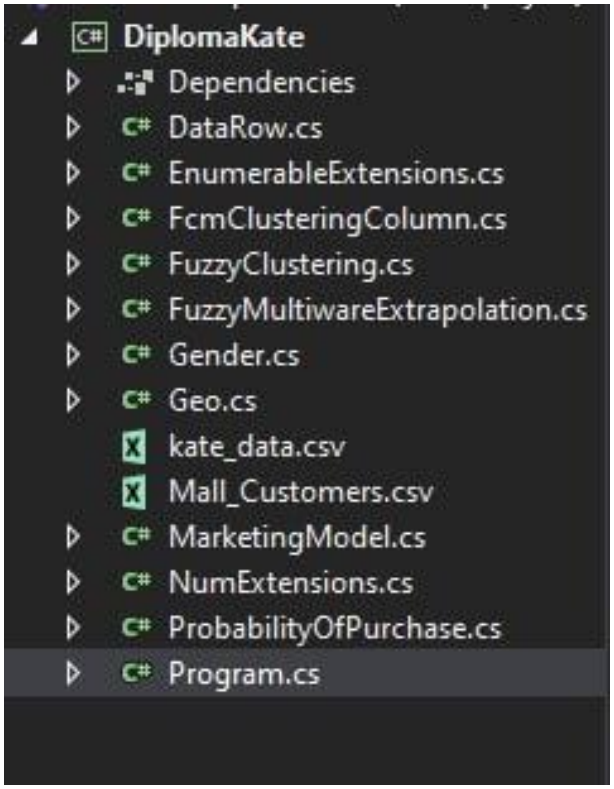


Рисунок 3.1 – Структура проекту

3.2.1 Алгоритм розробки FCM

Загальний алгоритм реалізації алгоритму Fuzzy c-means зображено на рисунку 3.2.

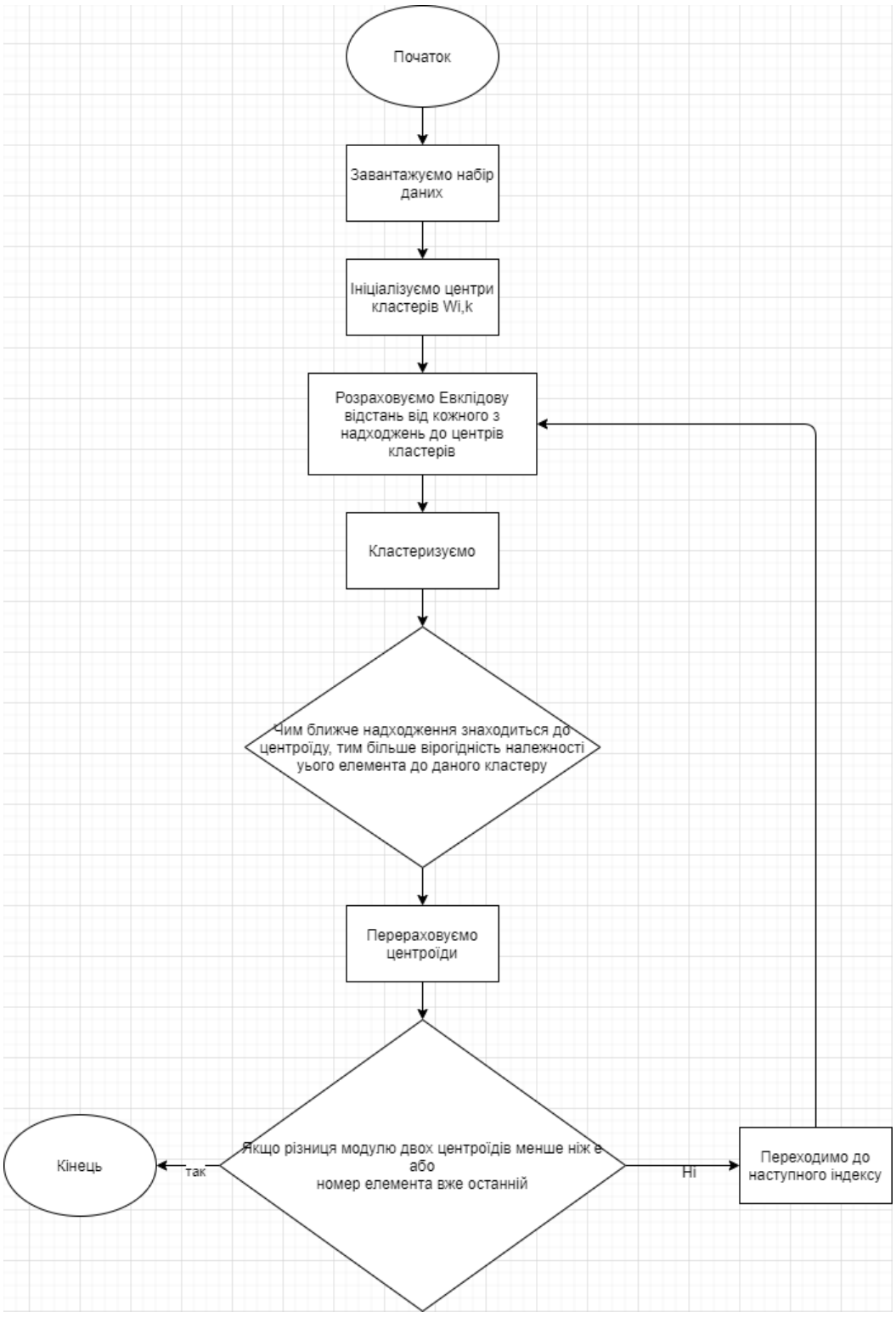
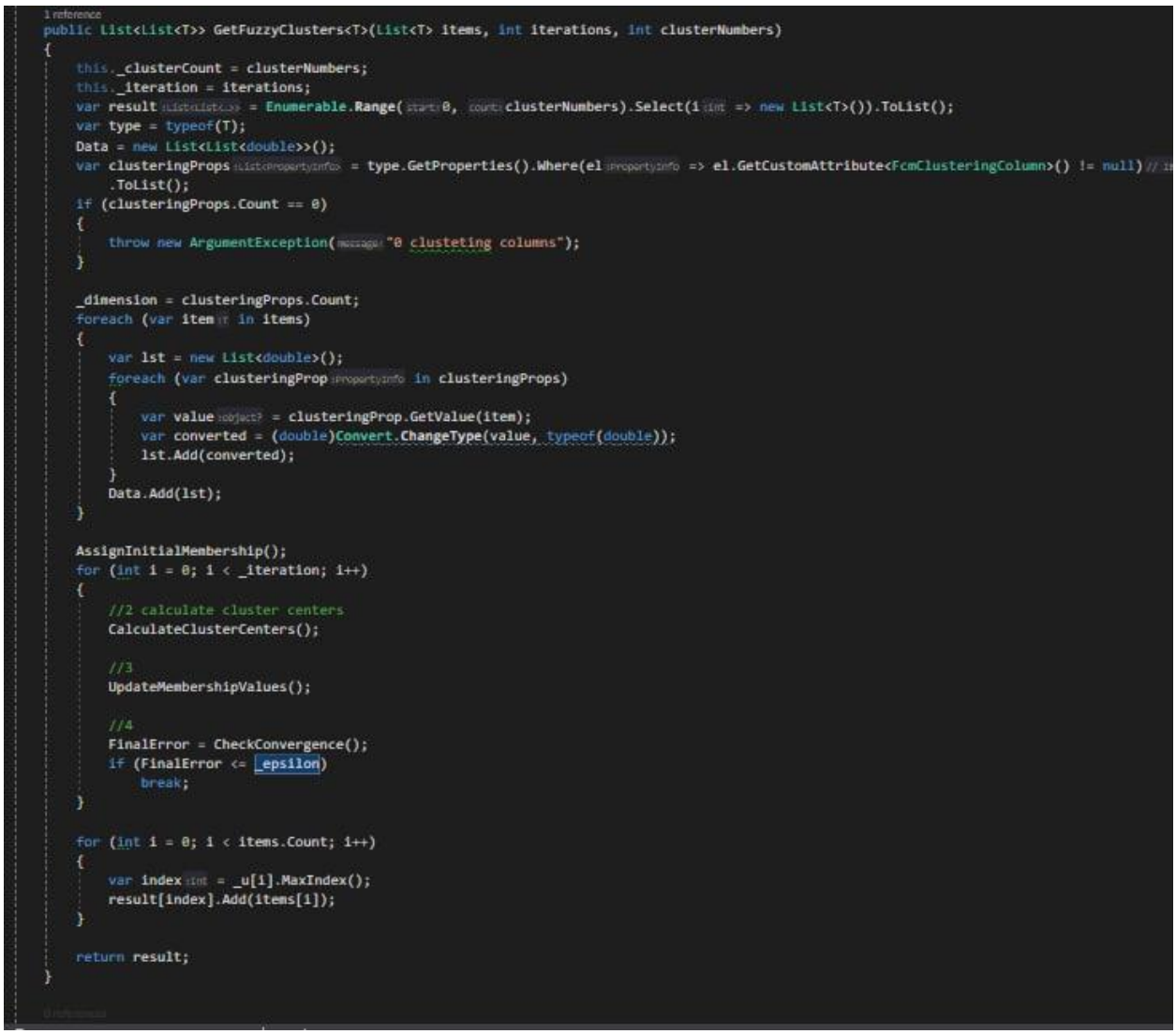


Рисунок 3.2 – Алгоритм реалізації FCM

Реалізація методу зображена на рисунку 3.3.



```
1 reference
public List<List<T>> GetFuzzyClusters<T>(List<T> items, int iterations, int clusterNumbers)
{
    this._clusterCount = clusterNumbers;
    this._iteration = iterations;
    var result = Enumerable.Range(start: 0, count: clusterNumbers).Select(i => new List<T>()).ToList();
    var type = typeof(T);
    Data = new List<List<double>>();
    var clusteringProps = type.GetProperties().Where(el => el.GetCustomAttribute<FcmClusteringColumn>() != null)
        .ToList();
    if (clusteringProps.Count == 0)
    {
        throw new ArgumentException(message: "0 clusteting columns");
    }

    _dimension = clusteringProps.Count;
    foreach (var item in items)
    {
        var lst = new List<double>();
        foreach (var clusteringProp in clusteringProps)
        {
            var value = clusteringProp.GetValue(item);
            var converted = (double)Convert.ChangeType(value, typeof(double));
            lst.Add(converted);
        }
        Data.Add(lst);
    }

    AssignInitialMembership();
    for (int i = 0; i < _iteration; i++)
    {
        //2 calculate cluster centers
        CalculateClusterCenters();

        //3
        UpdateMembershipValues();

        //4
        FinalError = CheckConvergence();
        if (FinalError <= _epsilon)
            break;
    }

    for (int i = 0; i < items.Count; i++)
    {
        var index = _u[i].MaxIndex();
        result[index].Add(items[i]);
    }

    return result;
}
```

Рисунок 3.3 – Реалізація методу FCM

На першому кроці завантажуємо дані. Далі назначаємо початкове значення приналежності. Розраховуємо центроїди кластерів, як зазначено на рисунку 3.4.

```
/**
 * in this function we calculate value of each cluster
 */
2 references
private void CalculateClusterCenters()
{
    _clusterCenters.Clear();
    for (int i = 0; i < _clusterCount; i++)
    {
        List<double> tmp = new List<double>();
        for (int j = 0; j < _dimension; j++)
        {
            double clusterIj;
            double sum1 = 0;
            double sum2 = 0;
            for (int k = 0; k < Data.Count; k++)
            {
                double tt = Math.Pow(_u[k][i], _fuzziness);
                sum1 += tt * Data[k][j];
                sum2 += tt;
            }
            clusterIj = sum1 / sum2;
            tmp.Add(clusterIj);
        }
        _clusterCenters.Add(tmp);
    }
}
```

Рисунок 3.5 – Розрахунок центроїдів

Найважливіший крок – підрахунок відстаней від надходжень до центроїдів. У даному алгоритмі використовано Евклідову відстань (рис. 3.6).

```
private double Distance(List<double> p1, List<double> p2)
{
    double sum = 0;
    for (int i = 0; i < p1.Count; i++)
    {
        sum += Math.Pow(p1[i] - p2[i], 2);
    }
    sum = (float)Math.Sqrt(sum);
    return sum;
}
```

Рисунок 3.6 – Евклідова відстань

Останнім кроком перераховуємо значення належності до кластерів (рис. 3.7).

```
2 references
private void UpdateMembershipValues()
{
    for (int i = 0; i < Data.Count; i++)
    {
        for (int j = 0; j < _clusterCount; j++)
        {
            _uPre[i][j] = _u[i][j];
            double sum = 0;
            double upper = Distance(Data[i], _clusterCenters[j]
            for (int k = 0; k < _clusterCount; k++)
            {
                double lower = Distance(Data[i], _clusterCenter
                sum += Math.Pow((upper / lower), 2 / (_fuzzines
            }
            _u[i][j] = 1 / sum;
        }
    }
}
```

Рисунок 3.7 – Перерахунок значень належності

3.2.2 Алгоритм розробки методу багатовимірної нечіткої екстраполяції

Алгоритм багатовимірної нечіткої екстраполяції можливо зручно зобразити схемою, як показано на рисунку 3.8.

Рисунок 3.8 – Алгоритм багаторівневої нечіткої екстраполяції

Реалізацію алгоритму багатовимірної нечіткої екстраполяції для заповнення пропусків та подальшої кластеризації зображено на рисунку 3.9.

```
1 reference
public List<List<T>> GetClustersWithFilledData(List<T> items)
{
    _iteration = 0;
    _source = items;
    FillM0(items);
    GetRandomCentroids();
    FillMatrixes();
    FindDistances();
    FindU();
    var suc = GetNewCentroids();
    while (suc)
    {
        _iteration++;
        FillMatrixes();
        FindDistances();
        FindU();
        var err = CheckConvergence();
        if (err < 0.0000001)
        {
            break;
        }
        suc = GetNewCentroids();
    }

    return GetClusters();
}
```

Рисунок 3.9 – Алгоритм багатовимірної нечіткої екстраполяції

На першому кроці ми обираємо випадкові центроїди (рис. 3.10) та заповнюємо пропуски у надходженнях відповідними значеннями центроїдів (рис. 3.11).

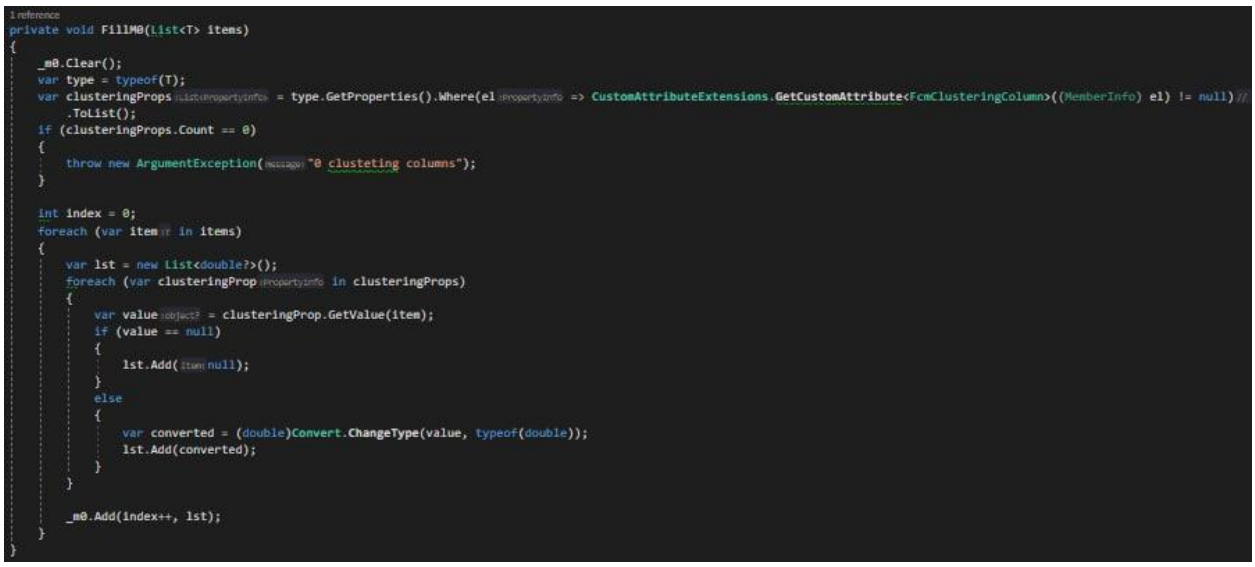
```
private void GetRandomCentroids()
{
    _centroids.Clear();
    _currentCentroidsIndexes.Clear();
    var notNullsList = _m0.Where(el => el.Value.All(d => d.HasValue)).ToDictionary(el => el.Key, el => el.Value.Select(el => el.Value).ToList(
    if (notNullsList.Count < _clustersCount)
    {
        throw new ArgumentException("clusters count > filled rows");
    }

    var keysIndexes = new HashSet<int>();
    var centroidsKeys = new List<int>();
    var rnd = new Random();
    var keys = notNullsList.Keys.ToList();

    for (int i = 0; i < _clustersCount;)
    {
        var kIdx = rnd.Next(0, notNullsList.Keys.Count);
        var k = keys[kIdx];
        if (keysIndexes.Add(kIdx))
        {
            centroidsKeys.Add(k);
            _currentCentroidsIndexes.Add(kIdx);
            i++;
        }
    }

    foreach (var centroidsKey in centroidsKeys)
    {
        _centroids.Add(notNullsList[centroidsKey]);
    }
}
```

Рисунок 3.10 – Обираємо центроїди



```
1 reference
private void FillM0(List<T> items)
{
    _m0.Clear();
    var type = typeof(T);
    var clusteringProps = type.GetProperties().Where(el => CustomAttributeExtensions.GetCustomAttribute<FcmClusteringColumn>((MemberInfo) el) != null)//
        .ToList();
    if (clusteringProps.Count == 0)
    {
        throw new ArgumentException("0 clusteting columns");
    }

    int index = 0;
    foreach (var item in items)
    {
        var lst = new List<double?>();
        foreach (var clusteringProp in clusteringProps)
        {
            var value = clusteringProp.GetValue(item);
            if (value == null)
            {
                lst.Add(null);
            }
            else
            {
                var converted = (double)Convert.ChangeType(value, typeof(double));
                lst.Add(converted);
            }
        }

        _m0.Add(index++, lst);
    }
}
```

Рисунок 3.11 – Заповнення пропусків

Наступним етапом розраховуємо часткову відстань та параметр належності (рис. 3.12).

```
private void FindU()
{
    _uPre = _u.Select(el => el.ToDictionary(keySelector: el => el.Key, elementSelector: el => el.Value)).ToList();
    _u.Clear();
    for (int i = 0; i < _clustersCount; i++)
    {
        var u = new Dictionary<int, double>();
        var dist = _distances[i];
        var distancesSum = dist.Select(el => el.Value).Sum();
        foreach (var el in dist)
        {
            u[el.Key] = el.Value / distancesSum;
        }
        _u.Add(u);
    }
}
```

Рисунок 3.12 – Визначення параметра належності до кластера

Відповідно до отриманих результатів необхідно перерахувати центроїди та повторити всі етапи ітеративно до тих пір, доки центроїди не будуть змінюватись.

3.3 Тестування розробленої програми

Наступним кроком після реалізації методів проводимо тестування програми. Необхідно обрати та підготувати набір даних, застосувати до них алгоритми та проаналізувати отримані результати.

3.3.1 Набір даних для тестування

Для тестування програмної реалізації методів обрано набір даних з результатами маркетингових досліджень. Дані містять інформацію про покупців інтернет-магазинів, отриманий шляхом соціального опитування на краудсорсингових платформах. Початковий вигляд даних зображено на рисунку 3.13.

file_descriprion | kate_data.csv

```
CustomerID,Gender,Age,AnnualIncome,SpendingScore,IsCustomer,Geo,EngagementRate,ProbabilityOfPurchase
1,Male,22,9603,0,False,Kyiv,66,Medium
2,Female,50,397,97,True,Lviv,76,Hight
3,Male,20,2697,0,False,Odessa,78,Medium
4,Male,26,6659,0,False,Lviv,83,Medium
5,Male,51,3770,48,True,Dnipro,95,Hight
6,Male,33,7192,0,False,Odessa,95,Medium
7,Male,51,893,26,True,Kyiv,73,Medium
8,Female,19,466,10,True,Odessa,58,Hight
9,Female,30,9253,0,False,Kharkiv,47,Medium
10,Female,58,8074,25,True,Kharkiv,64,Medium
11,Male,55,4557,78,True,Dnipro,82,Hight
12,Male,25,8600,64,True,Kharkiv,31,Medium
13,Female,15,4591,0,False,Other,86,Medium
14,Female,43,9165,50,True,Other,88,Hight
15,Female,23,8840,75,True,Odessa,40,Medium
16,Male,59,1335,22,True,Dnipro,75,Medium
17,Female,52,1131,25,True,Kyiv,37,Low
18,Male,36,2114,43,True,Dnipro,59,Medium
19,Female,13,8240,25,True,Odessa,66,Hight
20,Male,45,5270,11,True,Dnipro,72,Medium
21,Male,25,8214,0,False,Odessa,43,Low
22,Male,18,9722,20,True,Dnipro,47,Low
23,Male,43,5097,0,False,Kharkiv,35,Low
24,Male,30,7789,0,False,Odessa,55,Low
25,Female,25,5339,0,False,Kyiv,74,Medium
26,Female,18,2968,0,False,Lviv,64,Medium
27,Male,47,514,0,False,Kyiv,67,Low
28,Female,23,1312,0,False,Dnipro,82,Medium
29,Male,42,271,0,False,Dnipro,31,Low
```

Рисунок 3.13 – Початковий вигляд даних

Для полегшення аналізу змінемо формат файлу та відобразимо їх у вигляді таблиці, зображеної на рисунку 3.14

	A	B	C	D	E	F	G	H	I
1	CustomerID	Gender	Age	AnnualIncome	SpendingScore	IsCustomer	Geo	EngagementRate	ProbabilityOfPurchase
2	1	Male	22	9603	0	FALSE	Kyiv	66	Medium
3	2	Female	50	397	97	TRUE	Lviv	76	Hight
4	3	Male	20	2697	0	FALSE	Odessa	78	Medium
5	4	Male	26	6659	0	FALSE	Lviv	83	Medium
6	5	Male	51	3770	48	TRUE	Dnipro	95	Hight
7	6	Male	33	7192	0	FALSE	Odessa	95	Medium
8	7	Male	51	893	26	TRUE	Kyiv	73	Medium
9	8	Female	19	466	10	TRUE	Odessa	58	Hight
10	9	Female	30	9253	0	FALSE	Kharkiv	47	Medium
11	10	Female	58	8074	25	TRUE	Kharkiv	64	Medium
12	11	Male	55	4557	78	TRUE	Dnipro	82	Hight
13	12	Male	25	8600	64	TRUE	Kharkiv	31	Medium
14	13	Female	15	4591	0	FALSE	Other	86	Medium
15	14	Female	43	9165	50	TRUE	Other	88	Hight
16	15	Female	23	8840	75	TRUE	Odessa	40	Medium
17	16	Male	59	1335	22	TRUE	Dnipro	75	Medium
18	17	Female	52	1131	25	TRUE	Kyiv	37	Low
19	18	Male	36	2114	43	TRUE	Dnipro	59	Medium
20	19	Female	13	8240	25	TRUE	Odessa	66	Hight
21	20	Male	45	5270	11	TRUE	Dnipro	72	Medium
22	21	Male	25	8214	0	FALSE	Odessa	43	Low
23	22	Male	18	9722	20	TRUE	Dnipro	47	Low
24	23	Male	43	5097	0	FALSE	Kharkiv	35	Low
25	24	Male	30	7789	0	FALSE	Odessa	55	Low
26	25	Female	25	5339	0	FALSE	Kyiv	74	Medium
27	26	Female	18	2968	0	FALSE	Lviv	64	Medium
28	27	Male	47	514	0	FALSE	Kyiv	67	Low

Рисунок 3.14 – Таблиця з набором даних для тестування

Набір даних має 9 параметрів (рис. 3.15):

– Ідентифікаційний номер;
– Стать;
– Вік;
– Річний дохід;
– Оцінка витрат;
– Чи є клієнтом онлайн-магазину;
– Геоположення;
– Рівень залученості;
– Вірогідність покупки.

No.	Name
1	CustomerID
2	Gender
3	Age
4	AnnualIncome
5	SpendingScore
6	IsCustomer
7	Geo
8	EngagementRate
9	ProbabilityOfPurchase

Рисунок 3.15 – Параметри набору даних

Параметр ідентифікаційного номеру має назву CustomerID, є унікальним та приймає значення від 1 до 10000 (рис. 3.16). Дане значення не буде використано як параметр в досліджуваних методах.

Name: CustomerID Type: Numeric
Missing: 0 (0%) Distinct: 10000 Unique: 10000 (100%)

Statistic	Value
Minimum	1
Maximum	10000

Рисунок 3.16 – CustomerID

Параметр визначення статі має назву Gender. Він може приймати 2 значення Male або Female. Кількість кожного зі значень зображено на рисунку 3.17.

Name: Gender Type: Nominal
Missing: 0 (0%) Distinct: 2 Unique: 0 (0%)

No.	Label	Count	Weight
1	Male	4941	4941.0
2	Female	5059	5059.0

Рисунок 3.17 – Gender

Параметр віку має назву Age та приймає значення від 13 до 65 (рис. 3.18). Показує вік людей, які приймали участь в соціальному опитуванні.

Name: Age Type: Numeric
Missing: 0 (0%) Distinct: 53 Unique: 0 (0%)

Statistic	Value
Minimum	13
Maximum	65
Mean	39.148

Рисунок 3.18 – Age

Параметр, що показує річний дохід учасників дослідження має назву AnnualIncome (рис. 3.19).

Name: AnnualIncome Type: Numeric
Missing: 0 (0%) Distinct: 6327 Unique: 3725 (37%)

Statistic	Value
Minimum	0
Maximum	9999
Mean	5018.001

Рисунок 3.19 – AnnualIncome

Параметр оцінки витрат (рис. 3.20) має назву SpendingScore та приймає значення від 0 до 100. Його основна мета показати на скільки багато людина витрачає в онлайн-магазині. Можемо побачити на рисунку 3.20, що середнім значенням є 24.737, тобто рівень витрат не є досить високим.

Name: SpendingScore Type: Numeric
Missing: 0 (0%) Distinct: 100 Unique: 0 (0%)

Statistic	Value
Minimum	0
Maximum	99
Mean	24.737

Рисунок 3.20 – SpeningScore

Параметр із назвою IsCustomer вказує чи є людина покупцем онлайн-магазину чи ні. Даний параметр може приймати одне з двох значень – False або True. Так як більшість людей на момент опитування ще ні разу не

купляли товар в даному онлайн-магазині, то бачимо (рис. 3.21), що кількість значення False перевищую кількість значень True.

Name: IsCustomer Type: Nominal
Missing: 0 (0%) Distinct: 2 Unique: 0 (0%)

No.	Label	Count	Weight
1	False	5032	5032.0
2	True	4968	4968.0

Рисунок 3.21 – IsCustomer

Параметр геоположення з назвою Geo приймає одне з 6 значень:

– Київ;
– Львів;
– Одеса;
– Дніпро;
– Харків;
– Інше.

Кількість входжень за даними значеннями параметру зображено на рисунку 3.22.

Name: Geo Type: Nominal
Missing: 0 (0%) Distinct: 6 Unique: 0 (0%)

No.	Label	Count	Weight
1	Kyiv	1678	1678.0
2	Lviv	1745	1745.0
3	Odessa	1685	1685.0
4	Dnipro	1635	1635.0
5	Kharkiv	1679	1679.0
6	Other	1578	1578.0

Рисунок 3.22 – Geo

Параметр рівня залученості має назву EngagementRate. Даний параметр показує відсоток залученості людини до магазину, тобто показує як часто вона відвідує сторінку, продивляється товари, зберігає чи порівнює товари, чи людина ставить лайки або пише коментарі. На рисунку 3.23 бачимо, що

значення нижче 20 відсутні. Це означає, що уся аудиторія магазину залучена хоча б частково, навіть якщо вони ще не є покупцями.

Name: EngagementRate Type: Numeric
Missing: 0 (0%) Distinct: 80 Unique: 0 (0%)

Statistic	Value
Minimum	20
Maximum	99
Mean	58.889

Рисунок 3.23 – EngagementRate

Параметр вірогідності покупки (рис. 3.24) є саме тим полем, який необхідно поставити у відповідність кластерам. Він має ProbabilityOfPurchase та приймає 3 значення – Low, Medium, High. На даний параметр впливають усі значення із набору даних окрім ідентифікаційного номера. Дане поле не буде розглядатися в алгоритмах, адже саме з ним необхідно провести порівняння результатів кластеризації.

Name: ProbabilityOfPurchase Type: Nominal
Missing: 0 (0%) Distinct: 3 Unique: 0 (0%)

No.	Label	Count	Weight
1	Medium	4245	4245.0
2	Hight	2754	2754.0
3	Low	3001	3001.0

Рисунок 3.24 – ProbabilityOfPurchase

З розглянутих параметрів одразу можливо зробити висновок, що найбільша кількість ProbabilityOfPurchase приймає значення Medium через те, що вся аудиторія залучена до активності в магазині, навіть якщо вони ще не були покупцями товару, а рівень річного доходу розподілен рівномірно. Під параметр Low підпадають ті, хто не є покупцем і має найнижчі значення параметрів SpendingScore, AnnualIncome та EngagementRate.

3.3.2 Підготовка даних

Для коректності роботи методів усі значення повинні приймати числові значення. Тому набір даних було відредаговано та усі номінальні значення були замінені на чисельні. Відповідність початкових та змінених значень відображена у таблиці 3.1

Таблиця 3.1 – Заміна номінальних значень чисельними

Параметр	Номінальне значення	Чисельне значення
Gender	Male	0
	Female	1
Geo	Kharkiv	0
	Kyiv	1
	Odessa	2
	Lviv	3
	Dnipro	4
	Other	5
isCustomer	False	0
	True	1

Результати заміни зображено на рисунку 3.25.

	A	B	C	D	E	F	G	H	I
1	CustomerID	Gender	Age	AnnualIncome	SpendingScore	IsCustomer	Geo	EngagementRate	ProbabilityOfPurchase
2	1	0	22	9603	0	0	1	66	1
3	2	1	50	397	97	1	3	76	2
4	3	0	20	2697	0	0	2	78	1
5	4	0	26	6659	0	0	3	83	1
6	5	0	51	3770	48	1	4	95	2
7	6	0	33	7192	0	0	2	95	1
8	7	0	51	893	26	1	1	73	1
9	8	1	19	466	10	1	2	58	2
10	9	1	30	9253	0	0	0	47	1
11	10	1	58	8074	25	1	0	64	1
12	11	0	55	4557	78	1	4	82	2
13	12	0	25	8600	64	1	0	31	1
14	13	1	15	4591	0	0	5	86	1
15	14	1	43	9165	50	1	5	88	2
16	15	1	23	8840	75	1	2	40	1
17	16	0	59	1335	22	1	4	75	1
18	17	1	52	1131	25	1	1	37	0
19	18	0	36	2114	43	1	4	59	1
20	19	1	13	8240	25	1	2	66	2
21	20	0	45	5270	11	1	4	72	1
22	21	0	25	8214	0	0	2	43	0
23	22	0	18	9722	20	1	4	47	0
24	23	0	43	5097	0	0	0	35	0
25	24	0	30	7789	0	0	2	55	0
26	25	1	25	5339	0	0	1	74	1
27	26	1	18	2968	0	0	3	64	1
28	27	0	47	514	0	0	1	67	0

Рисунок 3.25 – Підготовлені значення набору даних для проведення експерименту

Наступним етапом необхідно згенерувати набір даних з пропусками. Для цього реалізуємо функції видаленні випадкових значень (рис. 3.26).

```
1 reference
private static void GenerateRandomNullsVals(string[] args)
{
    var filePath = args[1];
    var outputPath = args[2];
    var probability = double.Parse(args[3]);
    probability = probability > 1 ? 1 : probability;
    var maxRowsCount = int.Parse(args[4]);
    var dir = Path.GetDirectoryName(outputPath);
    if (!Directory.Exists(dir))
    {
        Directory.CreateDirectory(dir);
    }

    var csvReader = new CsvReader(new StreamReader(File.Open(filePath, FileMode.Open, FileAccess.Read, FileShare.Read)), CultureInfo.InvariantCulture);
    var records = csvReader.GetRecords<MarketingModel>().ToList();
    var clusteringColumns = typeof(MarketingModel).GetProperties()
        .Where(el => el.GetCustomAttribute<FcmClusteringColumn>() != null).ToList();
    using (var fs = File.Open(outputPath, FileMode.Create, FileAccess.Write, FileShare.Write))
    {
        using var streamWriter = new StreamWriter(fs);
        using var writer = new CsvWriter(streamWriter, CultureInfo.InvariantCulture);
        writer.WriteHeader<MarketingModel>();
        writer.NextRecord();
        var rnd = new Random();
        foreach (var el in records)
        {
            if (rnd.NextDouble() <= probability)
            {
                var props = clusteringColumns.ToList();
                var nullColumnsCount = rnd.Next(1, maxRowsCount + 1);
                for (int i = 0; i < nullColumnsCount; i++)
                {
                    if (props.Count == 0)
                    {
                        break;
                    }
                    var index = rnd.Next(0, props.Count);
                    var p = props[index];
                    p.SetValue(el, null);
                    props.RemoveAt(index);
                }
            }

            writer.WriteRecord(el);
            writer.NextRecord();
        }
    }
}
```

Рисунок 3.26 – Функція генерації випадкових пропусків

Результат роботи методу зображено на рисунку 3.27

	A	B	C	D	E	F	G	H	I
1	CustomerID	Gender	Age	AnnualIncome	SpendingScore	IsCustomer	Geo	EngagementRate	ProbabilityOfPurchase
2	1	Male	22	9603			Kyiv	66	Medium
3	2	Female	50	397		TRUE	Lviv	76	Hight
4	3			2697	0		Odessa	78	Medium
5	4	Male	26		0	FALSE	Kharkiv	83	Medium
6	5	Male	51	3770		TRUE	Dnipro	95	Hight
7	6	Male	33		0	FALSE	Kharkiv	95	Medium
8	7	Male	51	893	26		Kyiv	73	Medium
9	8	Female	19			TRUE	Kharkiv	58	Hight
10	9	Female	30		0		Kharkiv	47	Medium
11	10			8074	25	TRUE	Kharkiv	64	Medium
12	11	Male	55		78		Kharkiv	82	Hight
13	12	Male		8600	64	TRUE	Kharkiv	31	Medium
14	13	Female		4591	0	FALSE	Other	86	Medium
15	14	Female	43	9165			Other	88	Hight
16	15	Female	23	8840	75		Odessa	0	Medium
17	16	Male		1335	22	TRUE	Kharkiv	75	Medium
18	17	Female	52		25		Kyiv	37	Low
19	18	Male	36	2114	43	TRUE	Kharkiv	59	Medium
20	19	Female		8240	25	TRUE	Odessa	66	Hight
21	20		45	5270	11	TRUE	Kharkiv	72	Medium
22	21		25	8214	0	FALSE	Kharkiv	43	Low
23	22	Male		9722	20		Dnipro	47	Low
24	23	Male	43	5097		FALSE	Kharkiv	35	Low
25	24		30		0	FALSE	Odessa	55	Low
26	25		25	5339	0		Kharkiv	74	Medium
27	26	Female	18		0	FALSE	Lviv	64	Medium
28	27	Male	47	514	0		Kyiv	0	Low

Рисунок 3.27 – Набір даних з пропусками

Кількість заповнених значень та відсотки пропусків відповідно до різних параметрів відображено на рисунках 3.28 – 3.31.

Name: Gender Type: Nominal
Missing: 2891 (29%) Distinct: 2 Unique: 0 (0%)

No.	Label	Count	Weight
1	Male	3519	3519.0
2	Female	3590	3590.0

Рисунок 3.28 – Пропуски у полі Gender

Name: Age Type: Numeric
Missing: 2920 (29%) Distinct: 53 Unique: 0 (0%)

Statistic	Value
Minimum	13
Maximum	65
Mean	39.058

Рисунок 3.29 – Пропуски у полі Age

Name: AnnualIncome Type: Numeric
Missing: 2817 (28%) Distinct: 5107 Unique: 3481 (35%)

Statistic	Value
Minimum	5
Maximum	9999
Mean	5002.388

Рисунок 3.30 – Пропуски у полі AnnualIncome

Name: SpendingScore Type: Numeric
Missing: 2868 (29%) Distinct: 100 Unique: 0 (0%)

Statistic	Value
Minimum	0
Maximum	99
Mean	25.288

Рисунок 3.31 – Пропуски у полі SpendigScore

Наступним кроком нові дані також необхідно змінити для коректної роботи методів. Відповідність числових та номінальних значень так само беремо із таблиці 3.1. Результат перетворення набору даних зображено на рисунку 3.32

	A	B	C	D	E	F	G	H	I
1	CustomerID	Gender	Age	AnnualIncome	SpendingScore	IsCustomer	Geo	EngagementRate	ProbabilityOfPurchase
2	1	0	22	9603			1	66	1
3	2	1	50	397		1	Lviv	76	2
4	3			2697	0		2	78	1
5	4	0	26		0	0	0	83	1
6	5	0	51	3770		1	4	95	2
7	6	0	33		0	0	0	95	1
8	7	0	51	893	26		1	73	1
9	8	1	19			1	0	58	2
10	9	1	30		0		0	47	1
11	10			8074	25	1	0	64	1
12	11	0	55		78		0	82	2
13	12	0		8600	64	1	0	31	1
14	13	1		4591	0	0	5	86	1
15	14	1	43	9165			5	88	2
16	15	1	23	8840	75		2	0	1
17	16	0		1335	22	1	0	75	1
18	17	1	52		25		1	37	0
19	18	0	36	2114	43	1	0	59	1
20	19	1		8240	25	1	2	66	2
21	20		45	5270	11	1	0	72	1
22	21		25	8214	0	0	0	43	0
23	22	0		9722	20		4	47	0
24	23	0	43	5097		0	0	35	0
25	24		30		0	0	2	55	0
26	25		25	5339	0		0	74	1
27	26	1	18		0	0	3	64	1
28	27	0	47	514	0		1	0	0

Рисунок 3.32 – Набір даних з пропусками із числовими значеннями

Для використання даних у методах створюємо програмну модель даних. Для цього реалізуємо клас MarketingModel (рис. 3.33).

```
public class MarketingModel
{
    [Name("CustomerID")]
    [DataMember(Name = "CustomerID")]
    1 reference
    public int CustomerId { get; set; }
    [FcmClusteringColumn]
    [Name("AnnualIncome")]
    [DataMember(Name = "AnnualIncome")]
    3 references
    public int? AnnualIncome { get; set; }
    [FcmClusteringColumn]
    [Name("SpendingScore")]
    [DataMember(Name = "SpendingScore")]
    3 references
    public int? SpendingScore { get; set; }
    [FcmClusteringColumn]
    [Name("EngagementRate")]
    [DataMember(Name = "EngagementRate")]
    3 references
    public int EngagementRate { get; set; }
    [FcmClusteringColumn]
    [Name("Gender")]
    [DataMember(Name = "Gender")]
    5 references
    public Gender? Gender { get; set; }
    [FcmClusteringColumn]
    [Name("Age")]
    [DataMember(Name = "Age")]
    3 references
    public int? Age { get; set; }
    [FcmClusteringColumn]
    [Name("IsCustomer")]
    [DataMember(Name = "IsCustomer")]
    1 reference
    public bool? IsCustomer { get; set; }
    [FcmClusteringColumn]
    [Name("Geo")]
    [DataMember(Name = "Geo")]
    1 reference
    public Geo Geo { get; set; }
    [Name("ProbabilityOfPurchase")]
    [DataMember(Name = "ProbabilityOfPurchase")]
    7 references
    public ProbabilityOfPurchase? ProbabilityOfPurchase { get; set; }
}
```

Рисунок 3.33 – Клас MarketingModel

3.3.3 Тестування

На етапі тестування необхідно перевірити роботу алгоритму FCM на повних даних та алгоритму багатовимірної нечіткої на даних з пропусками та порівняти результати.

Вказуємо програмно, що кількість кластерів повинна дорівнювати 3 та вказуємо шлях до файлу з набором даних для тестування.

Після запуску програми отримуємо результати, зображені на рисунках 3.34 та 3.35.

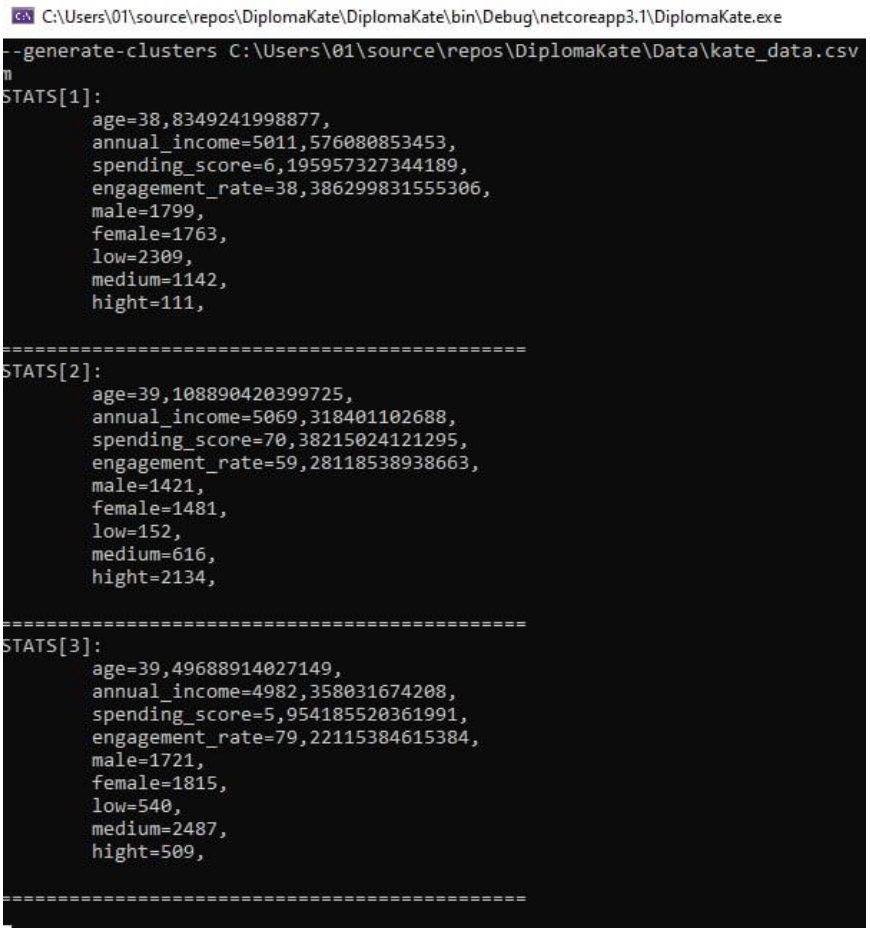


Рисунок 3.34 – Результат роботи алгоритму FCM на повних даних

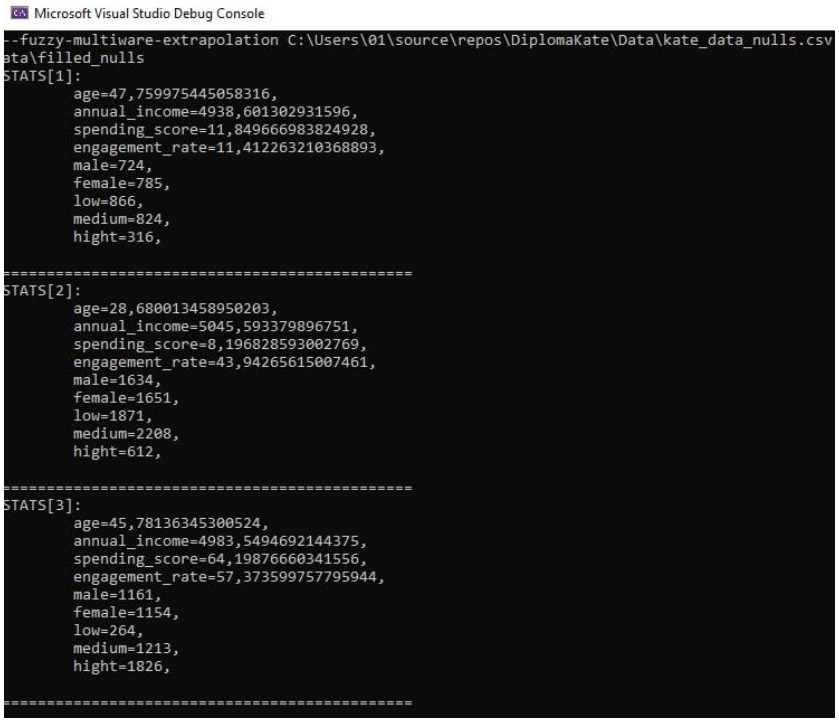


Рисунок 3.35 – Результат роботи алгоритму багатовимірної нечіткої екстраполяції на неповних даних

Реальний розподіл входжень по класам зображено на рисунку 3.36.

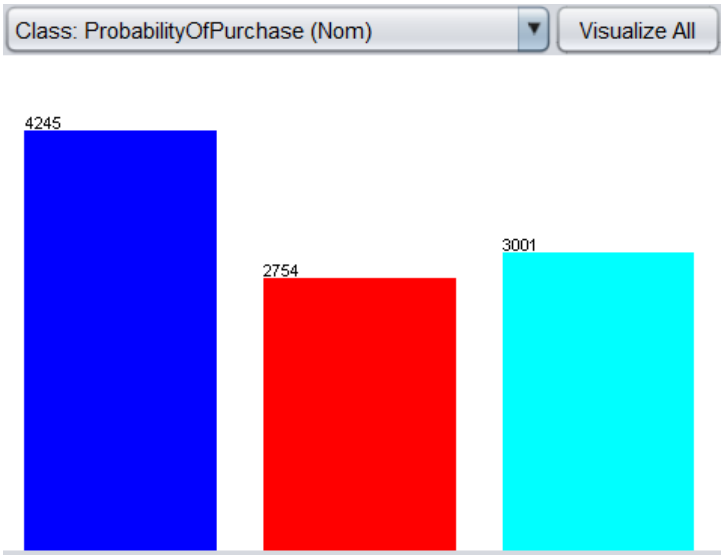


Рисунок 3.36 – Реальна кількість входжень зі значеннями PobabilityOfPurchase

Для оцінки якості кластеризації підрахуємо індекс Жаккара, який розраховується за формулою:

$$J = \frac{TP}{TP+TN+FP}, \tag{3.1}$$

де *TP* – елементи, що належать одному класу і одному кластеру;

TN – елементи, що належать одному кластеру, але різним класам;

FP – елементи, що належать різним кластерам, але одному класу.

Після підрахунку отримали результат, що для FCM *J*=0,51, а для алгоритму багатовимірної нечіткої екстраполяції *J*=0,57.

Даний результат показує, що кількість правильно кластеризованих даних вище 50%. Метод багатовимірної нечіткої кластеризації показав результат вище ніж FCM. Далі необхідно провести додаткове тестування з іншими наборами даних. У випадку, якщо результат кластеризації буде низьким, то необхідно покращувати алгоритми, які було досліджено у даній роботі.

ВИСНОВКИ

У рамках даного дослідження було розглянуто та проаналізовано методи відновлення даних у таблицях типу «об’єкт-властивість». За допомогою мови програмування C# реалізовано методи кластеризації FCM та метод відновлення даних з подальшою кластеризацією, що має назву багатовимірна нечітка екстраполяція.

Предметною областю для роботи було обрано сфера маркетингових досліджень. Об’єктом дослідження є набір даних з інформацією про покупців інтернет-магазинів, отриманий шляхом соціального опитування на краудсорсингових платформах. Для проведення аналізу методу багатовимірної ектраполяції дані були оброблені з метою створення випадкових пропусків.

Перед початком роботи проведено аналіз предметної області. Розглянуто, що пропуски в даних можуть виникати в будь-яких сферах діяльності людини, а, отже, питання попередньої обробки «сирих» даних має місце бути практично завжди, будь то дані економічних, медичних опитувань або технічних експериментів.

У роботі розглянуті механізми формування пропусків, представлений огляд основних базових методів відновлення пропусків у масивах даних.

Також, проаналізовано методи Бартлета, EM-алгоритм, ZET алгоритм, регрессії, багаторівневої нечіткої екстраполяції та Fuzzy *c*-means. Для даних методів були побудовані математичні моделі.

Практично всі розглянуті в статті методи мають одні і ті ж недоліки після процедури відновлення пропусків виникає залежність між спостереженнями, реальне і отримане розподілу після відновлення пропусків мають відмінності, виникає зміщення статистичних характеристик відновленої вибірки.

Незважаючи на наявність в деяких методах коригувальних процедур, вони все одно вважаються малоефективними, в порівнянні просунутими

методами відновлення пропусків. Після проведення тестування виявлено, що обидва методи мають оцінку якості кластеризації нижче середнього, зо може вказувати на неякісні дані для тестування.

Для подальшого дослідження рекомендовано провести додаткові тестування на інших наборах даних. Також, необхідно реалізувати й інші з розглянутих методів з метою більш глибокого аналізу та порівняння роботи алгоритмів.

Результати даної роботи апробовані на II Міжнародній науково-практичній конференції «World Science: Problems Prospects and Innovations» у місті Торонто, Канада [48].

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Руденко, Д. А., & Филатов, В. А. (2013). Формальный подход к описанию свойств данных в информационных системах. Вісник Херсонського національного технічного університету, (1), 146-149.

2. Путятін, Є. П., Гороховатський, В. О., & Матат, О. О. (2006). Методи та алгоритми комп'ютерного зору: навч. посіб. Харків: ТОВ «Компанія СМІТ».

3. Танянский, С. С., Руденко, Д. А., & Яковлева, Е. С. (1999). Об одном вопросе построения системы поддержки принятия решений на основе распределенных систем обработки данных. Радиоэлектроника и информатика, (4 (9)).

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

5. Вэй Тан, Брайан Блейк, Иман Салех. Аналитика Больших Данных и социальные сети // Открытые системы.СУБД. — 2013. — № 8. — С. 37–41. Режим доступа: http://www.osp.ru/os/2013/08/13037856

6. Бодянский, Е. В., Шафроненко, А. Ю., & Патлань, Е. В. НЕЧЕТКАЯ КЛАСТЕРИЗАЦИЯ МАССИВОВ ДАННЫХ НА ОСНОВЕ ЭВОЛЮЦИОННОГО МЕТОДА ОПТИМИЗАЦИИ КОШАЧЬИХ СТАЙ. ИНФОРМАЦИЯ, ЯЗЫК, ИНТЕЛЛЕКТ, 3.

7. Романова, Т. С. (2018). Непараметрический алгоритм восстановление пропусков «входных-выходных» переменных процесса (Doctoral dissertation, Сибирский федеральный университет).

8. Shafronenko, A., Dolotov, A., Bodyanskiy, Y., & Setlak, G. (2018, August). Fuzzy clustering of distorted observations based on optimal expansion using partial distances. In 2018 IEEE Second International Conference on Data Stream Mining & Processing (DSMP) (pp. 327-330). IEEE.

9. Шамрик, Д. Л. (2018). Базовые методы восстановления пропусков в массивах данных. In Информационные технологии в науке и производстве, материалы V Всероссийской молодежной научно-технической конференции.–ОмГТУ.

10. Бодянский, Е. В., & Шафроненко, А. Ю. (2018). Рандомизированная модификация метода оптимизации на основе кошачьих стай. Системи обробки інформації, (1), 142-147.

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

12. Shafronenko, A., Bodyanskiy, Ye., Rudenko, D.: Neuro-fuzzy clustering of Distorted Data Using Cat Swarm Optimization. Saarbrücken, LAP LAMBERT Academic Publishing (2020).

13. Shafronenko, A., Bodyanskiy, Y., Pliss, I., & Popov, S. (2020, September). Evolving Neo-Fuzzy System for Distorted Data Online Processing. In 2020 10th International Conference on Advanced Computer Information Technologies (ACIT) (pp. 352-355). IEEE.

14. Бодянский, Е. В., Кучеренко, В. Е., Кучеренко, Е. И., Михалёв, А. И., & Филатов, В. А. (2008). Гибридные нейро-фаззи модели и мультиагентные технологии в сложных системах.

15. Shafronenko, A. Yu, Bodyanskiy, Ye. V., Pliss I.P.: The Fast Modification of Evolutionary Bioinspired Cat Swarm Optimization Method. In: 2019 IEEE 8th International Conference on Advanced Optoelectronics and Lasers (CAOL), Sozopol, Bulgaria, 6-8 Sept. 2019, pp. 548-552. (2019) doi: 10.1109/CAOL46282.2019.9019583

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

17. Шафроненко А.Ю., Волкова В.В., Бодянский Е.В. Адаптивная кластеризация данных с пропущенными значениями. Radio Electronics, Computer, Science, Control. 2011. №2 (25). P.115-119

18. Shafronenko A., Pliss I., Bodyanskiy Ye. The evolving adaptive neural network for data processing with missing observations. Radio Electronics, Computer, Science, Control. 2013. №2 (29). P.119-125**.** DOI: http://dx.doi.org/10.15588/1607-3274-2013-2-19

19. Bodyanskiy, Y., Shafronenko, A., Rudenko, D. Online neuro fuzzy clustering of data with omissions and outliers based on completion strategy. CEUR Jornal. 2019. Vol.2353. P. 18-27.

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

21. D'urso, P., & Massari, R. (2019). Fuzzy clustering of mixed data. Information Sciences, 505, 513-534.

22. V. Lyashenko, O. Kobylin, A. Shafronenko. Wavelet Analysis and Decomposition Into Color Spaces in Researching of Human Fluorescently Labeled Images Tissues. Conference Proceedings «2019 IEEE 8th International Conference on Advanced Opto-electronics and Lasers (CAOL *2019)», Sozopol, Bulgaria, 06-08 September 2019, P.618-621 DOI: https://doi.org/10.1109/CAOL46282.2019.9019575

23. Bodyanskiy, Y., Shafronenko, A., & Volkova, V. (2012). Adaptive clustering of incomplete data using neuro-fuzzy Kohonen network. Artificial Intelligence Methods and Techniques for Business and Engineering Applications–Rzeszow-Sofia: ITHEA, 287-296.

24. Bodyanskiy, Y., Shafronenko, A., & Volkova, V. (2012). Adaptive fuzzy probabilistic clustering of incomplete data. INFORMATION MODELS & ANALYSES, 112.

25. Гороховатський, В. О., Руденко, Д. О., & Сірик, Т. О. (2019). Дослідження системи ієрархічних ознак при блочному поданні опису у складі множини ключових точок зображення.

26. Bodyanskiy, Y., & Shafronenko, A. ROBUST ADAPTIVE FUZZY CLUSTERING FOR DATA WITH MISSING VALUES. Transformation, 1, 1.

27. Bodyanskiy, Y. V., & Shafronenko, A. Y. (2014). Tables of data with gaps restoration using multivariate fuzzy extrapolation. Системні технології, (6), 11-17.

28. Рахимова, М. А. О СЕЛЕКЦИИ ПРИЗНАКОВ ПРИ НАЛИЧИИ ДАННЫХ С ПРОПУСКАМИ. СОВРЕМЕННЫЕ ПРОБЛЕМЫ МАТЕМАТИКИ, 228.

29. Zhu, X., Zhang, S., Zhu, Y., Zheng, W., & Yang, Y. (2020). Self-weighted multi-view fuzzy clustering. ACM transactions on knowledge discovery from data (TKDD), 14(4), 1-17.

30. Zhou, J., Lai, Z., Miao, D., Gao, C., & Yue, X. (2020). Multigranulation rough-fuzzy clustering based on shadowed sets. Information Sciences, 507, 553-573.

31. Мухамедиева, Д. Т. (2013). Алгоритм кластеризации правил систем нечеткого вывода. Естественные и технические науки, (2), 248-251.

32. Aliahmadipour, L., Torra, V., & Eslami, E. (2017). On hesitant fuzzy clustering and clustering of hesitant fuzzy data. In Fuzzy sets, rough sets, multisets and clustering (pp. 157-168). Springer, Cham.

33. Шафроненко, А. Ю. (2014). Методи динамічного інтелектуального аналізу даних з пропусками.

34. Nikfalazar, S., Yeh, C. H., Bedingfield, S., & Khorshidi, H. A. (2017, July). A new iterative fuzzy clustering algorithm for multiple imputation of missing data. In 2017 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE) (pp. 1-6). IEEE.

35. Sullivant, S. (2005). Small contingency tables with large gaps. SIAM Journal on Discrete Mathematics, 18(4), 787-793.

36. Nikfalazar, S., Yeh, C. H., Bedingfield, S., & Khorshidi, H. A. (2020). Missing data imputation using decision trees and fuzzy clustering with iterative learning. Knowledge and Information Systems, 62(6), 2419-2437.

37. Bodyanskiy, Y. V., & Shafronenko, A. Y. (2014). Tables of data with gaps restoration using multivariate fuzzy extrapolation. Системні технології, (6), 11-17.

38. Bodyanskiy Ye. V., Shafronenko A. Yu., Rudenko D. O., Klymova I. M. Online recurrent method of credibilistic fuzzy clustering. Topical issues of the development of modern science. 5th International scientific and practical conference. Publishing House «ACCENT». Sofia, Bulgaria. 2020. P. 37-40.

39. Фленов, М. Е. (2012). Библия C# (2-е издание). БХВ-Петербург.

40. Агуров, П. В. (2007). *C#. Сборник рецептов*. БХВ-Петербург.

41. Албахари, Д., & Албахари, Б. (2014). C# 5.0 Справочник. Полное описание языка.

42. Альфред, В. (2015). Ахо Компиляторы. Принципы, технологии и инструментарий/Альфред В. Ахо и др. *М.: Вильямс*, *689*.

43. Вагнер, Б. С. (2013). Эффективное программирование/Билл Вагнер. *М.: ЛОРИ*, *320*.

44. Троелсен, Э., & Джепикс, Ф. (2019). *Язык программирования C# 7 и платформы. NET и. NET Core*. Litres.

45. Deitel, P., & Deitel, H. (2016). *Visual C# how to program*. Pearson.

46. Sharp, J. (2018). *Microsoft visual C# step by step*. Microsoft Press.

47. Almeida, F. (2018). Visual C# .NET: Console Applications and Windows Forms.

48. Петухова К.С. (2020). Огляд методів відновлення даних у таблицях типу «об’єкт-властивість».