Міністерство освіти і науки України

Національний технічний університет

«Дніпровська політехніка»

Інститут електроенергетики

(інститут)

Факультет інформаційних технологій

(факультет)

Кафедра Системного аналізу та управління

(повна назва)

ПОЯСНЮВАЛЬНА ЗАПИСКА

кваліфікаційної роботи ступеню бакалавра

(бакалавра, магістра)

студента Рижченко Дениса Віталійовича

(ПІБ)

академічної групи 124-19-2

(шифр)

спеціальності 124 Системний аналіз

(код і назва спеціальності)

на тему: «Пошук правил асоціацій та аналіз ринкового кошика для знання поведінки споживачів і моделі купівлі»

(назва за наказом ректора)

Керівники	Прізвище, ініціали	Оцінка за шкалою		Підпис
		рейтинговою	інституційною	
кваліфікаційної роботи	*Д.ф.-м.н., проф. Купенко О. П.*			
розділів:				
Інформаційно-аналітичний	*Д.ф.-м.н., проф. Купенко О. П*			
Спеціальний	*Д.ф.-м.н., проф. Купенко О. П.*			

Рецензент	*К.т.н., доц. Сергєєва К.Л.*			

Нормоконтролер	*К.ф.-м.н., доц. Хом'як Т.В.*			

Дніпро
2023

ЗАТВЕРДЖЕНО:

завідувач кафедри

Системного аналізу та управління
(повна назва)

_____ к.т.н., доц. Т.А. Желдак
(підпис) (прізвище, ініціали)

«_____»_____2023 року

ЗАВДАННЯ

на кваліфікаційну роботу

ступеня бакалавра
(бакалавра, магістра)

студенту Рижченко Денису Віталійовичу **академічної групи** 124-19-2
(прізвище та ініціали) (шифр)

спеціальності 124 Системний аналіз

спеціалізації _____

на тему «Пошук правил асоціацій та аналіз ринкового кошика для знання поведінки споживачів і моделі купівлі»
затверджену наказом ректора НТУ «Дніпровська політехніка» від 16.05.2023 № 350-с

Розділ	Зміст	Термін виконання
Інформаційно-аналітичний розділ	Визначено поняття аналізу ринкового кошику, методів пошуку асоціативних правил, розглянуті методи визначення зв'язків між товарами.	30.04.2023
Спеціальний розділ	Проведено аналіз за допомогою методів Апріорі і Табу-пошуку, порівняння методів та визначення більш ефективного в залежності від наданих умов.	31.05.2023

Завдання видано _____ Купенко О.П.
(підпис керівника) (прізвище, ініціали)

Дата видачі 18.03.2023 р.

Дата подання до екзаменаційної комісії 12.06.2023 р.

Прийнято до виконання _____ Рижченко Д.В.
(підпис студента) (прізвище, ініціали)

РЕФЕРАТ

Пояснювальна записка: 77 с., 25 рис., 4 табл., 3 додатків, 12 джерел.

Аналіз ринкового кошика — це основний метод для аналізу та виявлення зв’язків між певними продуктами, які оточують нас у будь-якій сфері. Основа алгоритму даного методу полягає у тому, щоб знаходити певні комбінації товарів, які найчастіше обирають споживачі.

Об’єкт дослідження: аналіз історії покупок для визначення повторюваних товарних наборів у мережі супермаркетів Варус.

Предмет дослідження: пошук асоціативних правил методами аналізу ринкового кошика.

Мета дослідження: отримання значущих асоціативних правил на основі наданих даних.

Методи дослідження та апаратура: спостереження, порівняння та пояснення методів аналізу ринкового кошика застосованих до пошуку зв’язків між товарами; мова програмування Python.

В *інформаційно-аналітичному розділі* надано теоретичний опис поняття аналізу ринкового кошику, методів пошуку асоціативних правил, розглянуті методи визначення зв’язків між товарами.

У *спеціальному розділі* формування вхідних даних, аналіз за допомогою методу Апріорі, Табу-пошук; було проведено порівняння методів та визначення більш ефективного в залежності від наданих умов.

Практична цінність отриманих у роботі результатів полягає в отриманні інформації для розуміння поведінки клієнтів на основі їх покупок для визначення майбутніх організаційних та маркетингових рішень, що у свою чергу має на мате оптимізувати та збільшити кількість продажів.

Ключові слова: АСОЦІАТИВНІ ПРАИВИЛА, АНАЛІЗ РИНКОВОГО КОШИКУ, ОПТИМІЗАЦІЯ.

ABSTRACT

Explanatory note: 77 p., 25 pictures, 4 tables, 3 appendixes, 12 sources.

Market basket analysis is a basic method for analyzing and identifying relationships between certain products that surround us in any field. The basis of the algorithm of this method is to find certain combinations of goods that are most often chosen by consumers.

Object of research: analysis of purchase history to identify repeated product sets in the “Varus” supermarket chain

Subject of research: search for associative rules by methods of market basket analysis.

The purpose of the research: obtaining significant associative rules based on the provided data.

Research methods and equipment: observation, comparison and explanation of market basket analysis methods applied to the search for relationships between goods; Python programming language.

The *information-analytical section* a theoretical description of the concept of market basket analysis, methods of finding associative rules, and methods of determining relationships between goods are considered.

In a *special section*, the formation of input data, analysis using the Apriori method, Tabu-search; a comparison of methods was carried out and a more effective one was determined depending on the given conditions.

The *practical value* of the results obtained in the work consists in obtaining information to understand the behavior of customers based on their purchases in order to determine future organizational and marketing decisions, which in turn aims to optimize and increase the number of sales.

Keywords: ASSOCIATIVE RULES, MARKET BASKET ANALYSIS, OPTIMIZATION.

ЗМІСТ

ВСТУП

Аналіз ринкового кошика - це метод, який використовують роздрібні продавці для аналізу поведінки своїх покупців в магазинах. Його результат може покращити прибутковість постачальника, якість обслуговування та задоволення клієнтів. До мережі супермаркетів «Varus» належить багато маркетів з різноманітним асортиментом.

Дана робота має на меті використання знеособлених даних про транзакції клієнтів для описового аналізу моделей їх покупок, товарів, які зазвичай купуються разом, та одиниць, які найчастіше купуються в магазинах. Це полегшує формування списку найпопулярніших товарів, а також допомагає підтримувати постійний інтерес клієнтів до мережі «Varus». Часто елементи можуть бути проаналізовані для визначення правил асоціації шляхом аналізу доступних даних. У даній роботі розглядаються декілька алгоритмів, що допомагають знайти правила асоціації та кореляції, а саме: Апріорі та Табу-пошук.

На основі кожного методу були розроблені моделі для дослідження підходів до використання правил асоціації в рекомендаційних системах. Мінімальні значення підтримки та впевненості, які використовуються для правил дата майнінгу, є параметрами першого порядку.

1 ІНФОРМАЦІЙНО-АНАЛІТИЧНИЙ РОЗДІЛ

1.1 Аналіз ринкового кошика

Аналіз ринкового кошика — це основний метод для аналізу та виявлення зв’язків між певними продуктами, які оточують нас у будь-якій сфері. Основа алгоритму даного методу полягає у тому, щоб знаходити певні комбінації товарів, які найчастіше обирають споживачі. Найбільшу вигоду з цього отримують саме ті, хто продає, або надає ці товари. У даній ситуації цими «товарами» можуть бути, наприклад, як продукти споживання, так і фільми або послуги. На ринку споживання продавці цих «товарів» можуть за допомогою сучасних інформаційних технологій збирати дані, перетворювати ці дані в корисну інформацію, аналізувати її та отримувати необхідні результати для прийняття рішень в умовах постійної зміни динаміки поведінки ринку. На основі отриманих результатів можна побачити які товари найчастіше купляють разом та створити оптимальні необхідні умови для споживача, щоб зацікавити його у наступній покупці.

Перше використання аналізу ринкового кошику було проведено інформатиками Ракешем Агравалом, Томашом Імієлінським та Аруном Свамі 1993 році. Вони, на основі великої кількості зібраних даних про споживчі транзакції, виявили правила асоціації між придбаними товарами. Після цього, даний метод почав набирати популярність, через що у 2005 році був швидко впроваджений, як стандартний метод для практичного використання у сфері маркетингу. Пізніше маркетологи отримали для себе можливість використовувати аналіз ринкового кошика для вирішення задач прийняття рішення, а також для побудов теоретичних моделей рішень, що стосуються купівлі товарів.

Також у 2005 році дослідники Ю.Л. Чен, Тан, Шен і Ху запропонували встановити індуктивні гіпотези, які вказують на те, що клієнти можуть мати ментальні шаблони серед кількох інших колекцій об’єктів, включати зв’язки, які доповнюють один одного з точки зору його діяльності та інтересів. З логічної точки зору, результати цього дослідження можуть бути використані для прийняття рішень, таких як демонстрація двох продуктів близько один до одного, наприклад, якщо клієнти, які купують чай, ймовірно, купуватимуть й печиво або солодощі [4].

У свій час Рамакрішн Шрікант та Ракеш Агравал запропонували визначити **метод Apriori** [1], як традиційний алгоритм для пошуку частих булевих правил. На основі цього алгоритму люди використовували дані для аналізу відсотку клієнтів, які купують певний продукт, і відсоток загального доходу від цього продукту. Результати цього аналізу давали можливість формувати правила асоціації, за якими можна легко дізнатися про лідируючі товари та їх частку продажів. Наступним етапом йде визначення провідних продуктів, адже, важливо використовувати ці дані для розміщення інших подібних продуктів поблизу. Таким чином, у результаті було проаналізовано процес генерації правил асоціації для продуктів. Основною метою було виявлення закономірностей шляхом вилучення зв’язків або одночасних випадків із транзакційної інформації, наданої магазином. Дані, що були отримані у результаті аналізу, можуть допомогти маркету розробити план розташування товарів таким чином, щоб клієнти могли швидко знайти необхідні товари, а також щоб зацікавити їх у ще більшої кількості товарів, адже на думку дослідників споживач хоче заповнити свій кошик усіма необхідними товари швидко й в одному магазині, що у свою чергу обов’язково змусить його частіше скуплятися саме в цьому супермаркеті.

Дані, що використовуються у даній кваліфікаційній роботі, були взяті у мережі супермаркетів «Варус». Історія транзакцій вмістить у собі товари, які були придбані протягом місяця по одній касі. Для проведення аналізу, на основі даних

за місяць по всім касам неможливо через те, що ця інформація не може бути надана у відкритому доступі [9].

1.2 Сучасний підхід до аналізу ринкового кошику

Ми живемо у світі сучасних технологій, які у свою чергу надають можливість зберігати величезну кількість даних. Через зростання загальної інформації збільшилися об’єми баз даних, що надають можливість не тільки накопичувати інформацію, а також аналізувати її.

Беручи до уваги сучасний продовольчий ринок, який тримається на великої кількості супермаркетів та продуктових магазинів. Кожному з них треба проводити низькі рекламні акції, формувати стратегії яким чином з мінімальними витратами отримати максимальний прибуток, тобто обрати оптимальний шлях розвитку. Однак слід пам’ятати, що для досягнення цього, необхідно мати достатньо надійні та актуальні дані, що були отримані з практичного досвіду. В цей момент й приходить на допомогу інтелектуальний аналіз даних.

Він проводить процес уточнення важливих даних із величезних баз даних, який включає величезний набір статистичних і обчислювальних методів, аналіз нейронної мережі, кластеризацію, класифікацію та підсумовування інформації. Формулювання правил асоціації, який є одним із методів інтелектуального аналізу даних, реалізованих Market Basket Research, є частиною цього проекту.

Market Basket Research визначає моделі купівлі клієнтів, знаходячи значний зв’язок між товарами, які вони вибирають у своїх кошиках для покупок. Аналіз ринкового кошика допомагає процедурі, а також розширює цільову стратегію в численних бізнес-асоціаціях [8].

Пошук правил асоціації є одним із найважливіших методів інтелектуального аналізу даних, який використовується в аналізі ринкового кошика. Усі фрукти сортуються в одному проході в супермаркеті, усі молочні продукти розміщуються разом в іншому проході. Таким чином, витрачання часу

та навмисне інвестування ресурсів для розміщення найнеобхідніших речей у організований спосіб не тільки скорочує час, який клієнти витрачають на покупки, але й допомагає клієнтам придбати найбільш підходящі товари, які відносяться до їх типових ринкових кошиків. Правило асоціації пов’язане з твердженням «що з чим поєднується». Історія покупок споживачів відображається в історіях транзакцій, на основі чого й проводиться аналіз. Величина асоціативного правила може бути отримана за наявності чотирьох параметрів, а саме підтримки (support), впевненості (confidence), підйому (lift) та левередж (leverage).

Набори товарів — це сукупність усіх товарів у ринковому кошику (1.1),

$$I = \{i_1, i_2, \dots i_n\}. \quad (1.1)$$

Транзакція - це група всіх транзакцій (1.2),

$$T = \{ t_1, t_2, \dots t_n\} \quad (1.2)$$

Кожна окрема транзакція позначається, як набір елементів з елемента *I*. Якщо в наборі елементів є *n* елементів, це називається набором n елементів. Наприклад, це називається як ***3-itemset*** {Молоко, Хліб, М’ясо}. Якщо набір елементів не містить жодного елемента, він називається набором нульовим (порожнім). Представлення набору елементів більш ніж одним елементом значно збільшує шанси на те, що правила будуть перелічені [6].

1.3 Ієрархія асоціативних правил

Ієрархічні асоціативні правила (Hierarchical Association Rules) є розширенням класичних асоціативних правил, які використовуються для аналізу зв'язків між елементами набору даних. У той час, як класичні асоціативні правила

визначають асоціації між наборами товарів, ієрархічні асоціативні правила враховують ієрархічну структуру елементів.

В ієрархічному аналізі асоціативних правил елементи даних організовані в ієрархічну структуру, де кожен елемент має своїх батьків та нащадків. Це може бути представлене у вигляді дерева чи ієрархічної структури.

Ієрархічні асоціативні (рис. 1.1) правила можуть використовуватися для виявлення зв'язків між елементами різних рівнях ієрархії. Наприклад, правило "Якщо покупець купує яблука, то він також схильний купувати груші" або "Якщо покупець купує шампунь, то він також схильний купувати губну помаду". Ці правила дозволяють досліджувати як асоціації між конкретними товарами, а й асоціації більш високих рівнях ієрархії, що може бути корисним керувати асортиментом, маркетингових стратегій тощо. Ієрархічні асоціативні правила зазвичай витягуються з великих наборів даних за допомогою різних алгоритмів аналізу даних [6].

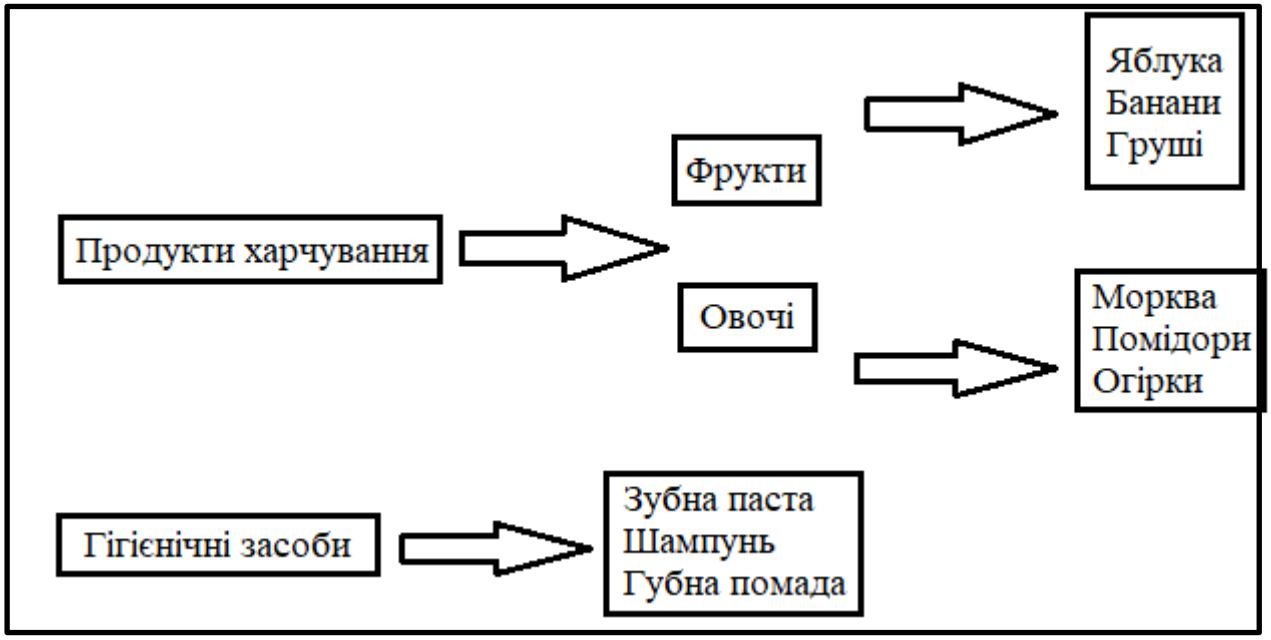


Рис. 1.1. Ієрархічні асоціативні правила

1.4 Параметри асоціативних правил

Розмір транзакції вимірюється кількістю елементів, що містяться в транзакції. Підрахунок підтримки, що стосується кількості транзакцій із певним набором елементів, є важливим компонентом набору елементів.

1.4.1 Support

Підтримка (Support) елемента або набору елементів виражається як частка транзакцій у певному наборі даних, яка містить кількість цього конкретного елемента продукту, до загальної кількості транзакцій. Підтримка дає можливість зрозуміти те, скільки разів набір елементів зустрічався в загальних транзакціях [2].

Наприклад, уявимо ситуацію в якій 100 людей відвідали супермаркет з метою придбати необхідні їм товари. Нехай лише 50 людей зі 100 придбали продукт *A*, 40 з них придбали продукт *B*, й лише 25 зі 100 придбали обидва продукту, тобто, продукт *A* й продукт *B*. Отже маємо, що підтримка продукту *A* становить 50%, підтримка продукту *B* становить 40% і Підтримка продуктів *A* і *B* разом становить 25%. Значення підтримки допомагає розглянути правила, які варті подальшого аналізу на співвідношення продуктів з іншими наявними продуктами в магазині. Наприклад, якщо ми хочемо відзначити набори елементів, які зустрічаються принаймні 50 разів у 10 000 транзакціях, тоді мінімальний рівень підтримки = 0,005. З низьким значенням підтримки ми не матимемо достатньо інформації про те, як продукти пов’язані один з одним, і, таким чином, це допоможе знайти «приховані» стосунки. Формула розрахунку «**Support**» наступна (1.3):

$$Support\ A\ = \frac{(Number\ of\ Transaction\ that\ Contains\ A)}{(\ Total\ Transaction)} \tag{1.3}$$

1.4.2 Confidence

Впевненість (Confidence) визначає ймовірності того, що клієнт, який купує продукт *A*, також купить продукт *B*. Таким чином, правило асоціації є зауваженням форми (набір елементів *A*) ⇒ (набір елементів *B*), де *A* являється прецедентом, а *B* являється наслідком [2]. Достовірність дає ймовірність виникнення наслідків у кошику з уже існуючими антецедентами. Параметр «впевненість» у правилах асоціації дуже часто має велике значення. Наприклад, нехай 50 клієнтів придбали продукт *A* та 25 з них придбали продукт *B*. Це відображає, що якщо хтось купить продукт *A*, то з імовірністю 50% він зможе купити продукт *B*. Формула розрахунку «**Confidence**» наступна (1.4):

$$Confidence(A => B) = P(A \mid B) = \frac{(Number\ of\ Transaction\ that\ Contains\ A\ and\ B)}{(Total\ Transaction\ that\ Contains\ A)} \quad (1.4)$$

1.4.3 Lift

У порівнянні з випадковим вибором транзакції, співвідношення показує, наскільки ефективним є правило для визначення наслідків. Загалом, підвищення або «**ліфтінг**» вище за одиницю означає, що правило має певну користь. Підйом більше одиниці вказує на те, що присутність товару *A* збільшила ймовірність наявності *B* у цій транзакції. Роль *A* зменшує ймовірність появи продукту *B*, у разі якщо значення підвищення становить менше одиниці. Формула розрахунку «**Lift**» наступна (1.5):

$$Lift\ (A => B) = \frac{(Confidence(A => B))}{(Support(B))} \quad (1.5)$$

1.4.4 Leverage

Леверидж вимірює кореляцію між наборами елементів шляхом порівняння підтримки наборів елементів за припущення незалежності та в наборі даних [5].

Зокрема, підйом використовує співвідношення, тоді як плече використовує різницю. Через різницю у формулі плече надає перевагу наборам предметів із вищою підтримкою, тоді як підйом може знайти сильні асоціації серед менш частих наборів предметів.

З точки зору інтерпретації, підтримка (AUB) — це підтримка, яка спостерігається в наборі даних, підтримка (A) х підтримка (B) — це очікувана підтримка за припущення незалежності (1.6).

$$Leverage\ of\ A \rightarrow B = Support\ (A \cup B) - Support(A) * Support(B) \qquad (1.6)$$

Таким чином,

Якщо плече = 0, A і B є незалежними (згідно з теорією ймовірності, *support(AUB) = support(A) x support(B)*, якщо A і B є незалежними).

Якщо плече > 0 , A і B позитивно корелюють . Зокрема, спостережувана підтримка вища за очікувану.

Якщо плече < 0 , A і B негативно корелюють . Використовуючи приклад супермаркету, клієнти, як правило, НЕ купують A і B разом.

1.4.5 Conviction

Переконання - це міра, яка допомагає судити про те, випадково з'явилося правило чи ні, тобто це можна назвати порівнянням ймовірності появи *if* без *then*, якщо вони залежали від фактичної частоти *if without then*. Формула розрахунку «**Conviction**» наступна (1.7):

$$conv(\{A \rightarrow B\}) = \frac{1-supp(\{B\})}{1-conf(\{A \rightarrow B\})} \quad (1.7)$$

1.5 Алгоритм Апріорі

Apriori – вважається основним алгоритмом для визначення набору частих елементів для булевих правил асоціації, що був запроваджений Агравалом та Стрикантом у 1994 році. Основний принцип даного алгоритму стверджує, що «у випадку, коли набір елементів є частим, то всі підмножини будуть являтися частими». Тобто коли підтримка набору елементів перевищує рівень підтримки, то набір елементів являється «частим». Даний алгоритм заснований на процесі передбачення елементів що регулярно переміщуються з попереднього етапу. В основі лежить термін «пріор», а сам алгоритм включає певний тип правил в інтелектуальному аналізі даних. Правило, що встановлює зв’язки між кількома атрибутами, називають аналізом спорідненості або аналізом ринкового кошика [9].

Для більш детального опису принципу алгортму Апріорі слід розглянути невеликий приклад:

Нехай, є набір елементів *{b, d, e}* із певного набору даних, який являється частим набором елементів, тобто його показник підтримки (0,3) більший за мінімальний показник підтримки (0,2). Виходячи з цього, отримуємо те, що тоді

усі його підмножини, такі як $b, d, e, \{b, d\}, \{b, e\}, \{d, e\},$ також будуть частими наборами елементів. Тобто, тому всі підтипи b, d, e мають бути регулярними, якщо $\{b, d, e\}$ є частими.

Й навпаки, якщо будь-які набори елементів, такі як $\{a, b\}$, будуть незвичайними, тоді усі супермножини мають бути навіть рідкісними. Весь сегмент, що містить надмножини $\{a, b\}$, можна відразу обрізати. Метод обрізки лінійного напрямку, що спирається на міру опори, називається обрізанням на основі опори. Ця характеристика також відома як антимонотонна властивість опорної міри.

Алгоритм **Apriori працює в два етапи**, а саме обрізати та приєднатися:

1. Створіть усі часті набори елементів;

Частий набір елементів – це набір елементів, який підтримує транзакції вище мінімальної.

2. Створюйте всі впевнені правила асоціації з частих наборів елементів.

Правило впевненого асоціювання – це правило з достовірністю, що перевищує мінімальну достовірність.

Щоб застосувати алгоритм Apriori до набору даних «Varus», застосовано клас Apriori, імпортований із бібліотеки Apyori.

• k-itemset — набір елементів, який містить: k номер елемента.

• Lk відноситься до частих наборів елементів з; k елементів.

• Ck відповідає частим наборам елементів-кандидатів з елементами; k.

Функція Apriori зменшує кількість елементів, які потрібно шукати, щоб знайти часті набори елементів. Цей алгоритм продовжує ідентифікацію 2-елементів за допомогою 1-елементів і 3-елементів за допомогою 2-елементів ітераційним способом. Це можна узагальнити таким чином; часті набори

предметів (k-1) елементів використовуються для пошуку частих кандидатів на k елементів [8].

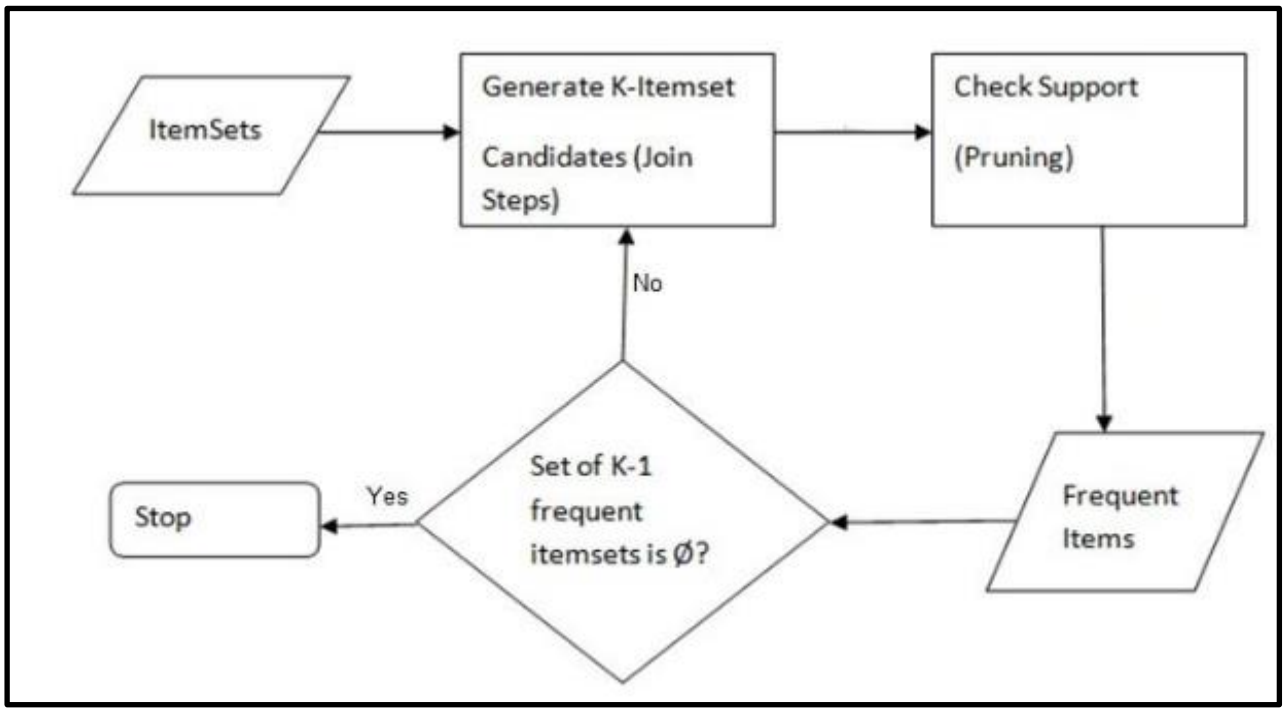


Рис. 1.2. Блок-схема алгоритму Апріорі

```
Apriori(T, ε)
    L1 ← {large 1 - itemsets}
    k ← 2
    while Lk-1 is not empty
        Ck ← Apriori_gen(Lk-1, k)
        for transactions t in T
            Dt ← {c in Ck : c ⊆ t}
            for candidates c in Dt
                count[c] ← count[c] + 1

        Lk ← {c in Ck : count[c] ≥ ε}
        k ← k + 1

    return Union(Lk)

Apriori_gen(L, k)
     result ← list()
     for all p ∈ L, q ∈ L where p1 = q1, p2 = q2, ..., pk-2 = qk-2 and pk-1 < qk-1
         c = p ∪ {qk-1}
         if u ∈ L for all u ⊆ c where |u| = k-1
             result.add(c)
      return result
```

Рис. 1.3. Псевдокод алгоритму Апріорі.

1.6 Алгоритм Табу-пошук

Табу-пошук (TS) — це метаевристичний метод пошуку, який можна використовувати для розв’язування задач комбінаторної оптимізації (задачі, де потрібне оптимальне впорядкування та вибір варіантів). Він був створений Фредом В. Гловером у 1986 році та офіційно оформлений у 1989 році.

Локальні (сусідні) пошуки шукають потенційне рішення проблеми та перевіряють її безпосередніх сусідів (тобто рішення, які схожі, за винятком дуже кількох незначних деталей) у надії знайти покращене рішення. Методи локального пошуку мають тенденцію застрягати в субоптимальних регіонах або на плато, де багато рішень однаково підходять.

Пошук Tabu покращує продуктивність локального пошуку, послаблюючи його основне правило. По-перше, на кожному кроці можуть бути прийняті погіршення ходів, якщо немає покращувальних ходів (наприклад, коли пошук застряг на строгому локальному мінімумі). Крім того, вводяться заборони (відтепер термін табу), щоб перешкодити пошуку повертатися до раніше відвіданих рішень.

Реалізація пошуку табу використовує структури пам’яті, які описують відвідані рішення або надані користувачем набори правил. Якщо потенційне рішення було попередньо відвідано протягом певного короткострокового періоду або якщо воно порушувало правило, воно позначається як «табу» (заборонено), щоб алгоритм не розглядав цю можливість повторно.

Поточні застосування TS охоплюють такі сфери: планування ресурсів, телекомунікації, проектування HBIC, фінансовий аналіз, планування, планування простору, розподіл енергії, молекулярна інженерія, логістика, класифікація

моделей, гнучке виробництво, управління відходами, розвідка корисних копалин, біомедичний аналіз, збереження навколишнього середовища та десятки інших.

Останніми роками журнали в різних галузях опублікували навчальні статті та обчислювальні дослідження, що документують успіхи табу-пошуку в розширенні межі проблем, які можна ефективно вирішувати — даючи рішення, якість яких часто значно перевершує якість, отриману за допомогою методів, які застосовувалися раніше.

Табу-пошук використовує процедуру локального або сусіднього пошуку для ітеративного переходу від одного потенційного рішення x до покращеного рішення x' в околицях x, доки не буде задоволено певний критерій зупинки (загалом, обмеження спроб або поріг оцінки). Процедури локального пошуку часто застрягають у районах із низьким рівнем оцінки або в районах, де показники плато. Щоб уникнути цих пасток і досліджувати регіони простору пошуку, які залишилися б недослідженими іншими процедурами локального пошуку, пошук *tabu* ретельно досліджує околиці кожного рішення під час пошуку. Розв'язки, допущені до нового оточення, $N^*(x)$, визначаються за допомогою використання структур пам'яті. Використовуючи ці структури пам'яті, пошук прогресує шляхом повторного переходу від поточного рішення x до покращеного рішення x' у $N^*(x)$.

Пошук табу має кілька подібностей із симуляцією відпалу, оскільки обидва включають можливі рухи вниз. Насправді симуляцію відпалу можна розглядати як особливу форму TS, за якої ми використовуємо «градуйований термін перебування», тобто хід стає табу із заданою ймовірністю.

Ці структури пам'яті утворюють те, що називається списком табу, набором правил і заборонених рішень, які використовуються для фільтрації того, які рішення будуть допущені до околиці $N^*(x)$, яку досліджуватиме пошук. У своїй найпростішій формі список табу — це короткочасний набір рішень, відвіданих у недавньому минулому (менше ніж n ітерацій тому, де n — кількість попередніх

рішень, які потрібно зберегти — також називається володінням табу). Зазвичай список табу складається з рішень, які змінилися в процесі переходу від одного рішення до іншого [3].

Algorithm 1: Tabu Search

Data: S - the search space, $maxIter$ - the maximal number of iterations, f - the objective function, the definition of neighborhoods, and the aspiration criteria.
Result: the best solution found
Choose the initial candidate solution $s \in S$
$s^* \leftarrow s$ `// Initialize the best-so-far solution.`
$k \leftarrow 1$
while $k \leq MaxIter$ **do**
  `/* Sample the allowed neighbors of` s `*/`
  Generate a sample $V(s,k)$ of the allowed solutions in $N(s,k)$
    // $s' \in V(s,k) \iff (s' \notin T) \vee (a(k,s') = true)$
  Set s to the best solution in $V(s,k)$
  `/* Update the best-so-far if necessary */`
  if $f(s) < f(s^*)$ **then**
    $s^* \leftarrow s$
  end
  Update T and a
  `/* Start another iteration */`
  $k \leftarrow k + 1$
end

Рис. 1.4. Псевдокод алгоритму Табу-Пошук

1.7 Висновки

Отже, як висновок можна сказати те, що у повсякденному життя будь-яка торгівельна компанія, будь-яка мережа супермаркетів, як наприклад «**Варус**», будь-яка торгівельна точка постійно потребує інформацію щодо переваг своїх клієнтів, своїх споживачів. Для отримання такого типу інформації був створений основний метод для аналізу та виявлення зв’язків між певними продуктами, який називається «аналіз ринкового кошика». В основі методу полягає аналіз даних, які

являють собою історію чеків, з метою виявлення певних комбінації товарів, які найчастіше обирають клієнти. Основними методами для проведення **Market Basket Research** являються **Апріорі** метод, а також метод **Табу-Пошук**.

Метод Apriori заснований на ідеї, що якщо деякий набір предметів є частим у наборі транзакцій, то будь-який його піднабір також буде частим. Іншими словами, якщо група товарів A та B часто купуються разом, то вони можуть бути пов'язані та утворювати асоціативне правило. Для визначення того, які набори є "частими", задається граничне значення підтримки. Якщо підтримка набору перевищує заданий поріг, він вважається частим. Після визначення всіх частих наборів алгоритм Apriori може використовуватися для визначення асоціативних правил між ними. Асоціативне правило свідчить про зв'язок між двома наборами.

В цілому метод Apriori є потужним інструментом для аналізу даних та виявлення прихованих взаємозв'язків між наборами даних.

Метод Tabu Search - це евристичний метод оптимізації, він заснований на принципі пошуку у просторі станів, де кожен стан є набір асоціативних правил.

Алгоритм Tabu Search починається із створення початкового набору асоціативних правил. Потім алгоритм проводить пошук у просторі станів, намагаючись знайти нові набори правил, які задовольняють заданим обмеженням, наприклад, мінімального значення підтримки і достовірність. У процесі пошуку алгоритм може використовувати механізм заборон (tabu), який запобігає повторенню вже пройдених станів. Таким чином, алгоритм не зациклюватиметься на вже досліджених наборах правил.

Алгоритм Tabu Search може бути ефективним методом для пошуку асоціативних правил у великих наборах даних. Він може використовувати різні метрики та обмеження для визначення найкращих правил та може застосовувати механізм заборон для прискорення пошуку. Однак, ефективність методу може залежати від вибраних метрик та параметрів алгоритму.

Зазвичай метод Apriori працює швидше, ніж метод Tabu Search, оскільки він використовує більш простий та прямолінійний алгоритм. У методі Apriori основний час витрачається створення кандидатів і підрахунок їх підтримки, тоді як у методі Tabu Search, потрібен складніший алгоритм оптимізації, що може уповільнити його швидкість роботи.

Обидва методи можуть бути ефективними у пошуку асоціативних правил, але ефективність залежить від розміру та характеристик набору даних, а також від вибраних метрик та параметрів алгоритмів. Наприклад, метод Apriori може працювати краще на невеликих наборах даних з низькою розмірністю, а метод Tabu Search може бути більш ефективним на великих наборах даних зі складними зв'язками між елементами.

Таким чином, вибір між методом Apriori та методом Tabu Search залежить від конкретного завдання та характеристик набору даних, а також від необхідної швидкості та ефективності роботи.

2 СПЕЦІАЛЬНИЙ РОЗДІЛ

2.1 Постановка задачі

Основною метою даної роботи являється вирішення наступною задачі:

Мережа супемаркетів «**Varus**» бажає провести оптимізацію загального асортименту товару, який виставляються на полиці, а також розробити новий план того, як і де повинні бути розташовані продукти різних типів між собою.

Для виконання даного завдання компанія надала певні дані, які у свою чергу являються чеками, які містять у собі інформацію про те, які товари купляли споживачі протягом місяця.

Вирішення даної задачі полягає у використанні певних методів пошуку асоціативних правил, а саме метод Апріорі та Табу-пошук, для того, щоб мати можливість провести аналіз вхідних даних, на основі якого можна передбачити наступне рішення клієнта, тобто визначити те, що він захоче придбати. У результаті цього можна повністю зрозуміти купівельну поведінку клієнта та запропонувати йому необхідні набори товарів наступним чином:

Сформувати необхідну рекламну акцію;

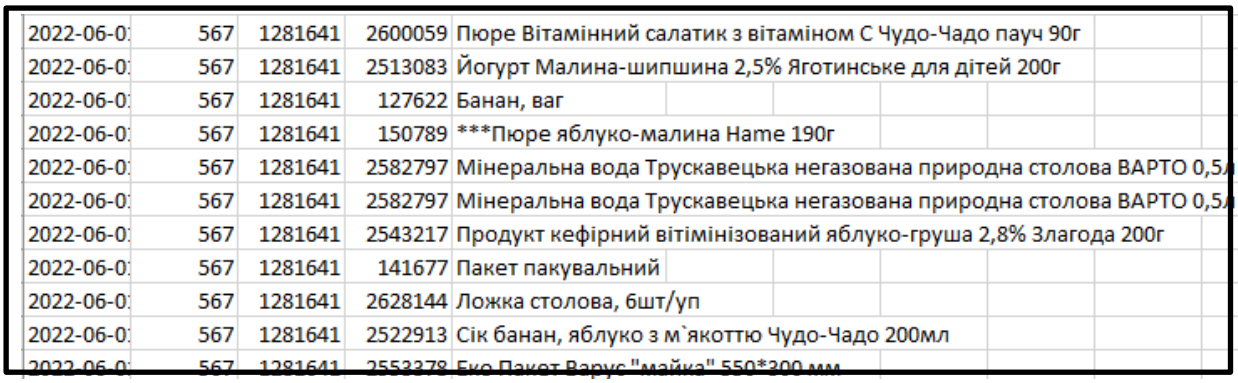
Правильно розташовувати товари;

Покращити планування та керування запасами товарів.

При правильному аналізі та виконанні описаних вимог супермаркет отримує можливість, не тільки «утримувати», а також закликати нових клієнтів, які будуть задоволені роботою компанії, що у результаті приведе до збільшення продажів.

2.2 Аналіз вхідних даних

Початкові дані надані у форматі *.csv*, наступним чином (рис. 2.1):



2022-06-0	567	1281641	2600059	Пюре Вітамінний салатик з вітаміном С Чудо-Чадо пауч 90г
2022-06-0	567	1281641	2513083	Йогурт Малина-шипшина 2,5% Яготинське для дітей 200г
2022-06-0	567	1281641	127622	Банан, ваг
2022-06-0	567	1281641	150789	***Пюре яблуко-малина Hame 190г
2022-06-0	567	1281641	2582797	Мінеральна вода Трускавецька негазована природна столова ВАРТО 0,5л
2022-06-0	567	1281641	2582797	Мінеральна вода Трускавецька негазована природна столова ВАРТО 0,5л
2022-06-0	567	1281641	2543217	Продукт кефірний вітімінізований яблуко-груша 2,8% Злагода 200г
2022-06-0	567	1281641	141677	Пакет пакувальний
2022-06-0	567	1281641	2628144	Ложка столова, 6шт/уп
2022-06-0	567	1281641	2522913	Сік банан, яблуко з м`якоттю Чудо-Чадо 200мл
2022-06-0	567	1281641	2553378	Еко Пакет Варус "майка" 550*300 мм

Рис. 2.1. Вхідні дані

Для початку роботи з аналізу даних та пошуку асоціативних правил перш за все необхідно провести очищення даних від непотрібної інформації, яка може негативно вплинути на подальшу роботу та результати. Також слід пам’ятати, що будь-яка інформація – це додаткове навантаження у плані обробки даних для комп’ютера, на якому проходить аналіз даних.

Для пошуку асоціативних правил необхідно бажано мати наступні дані: назва товару, порядковий номер чеку, а також порядковий номер товару. Отже, виходячи з цього можна прийти до висновку, що необхідно видалити наступні дані: порядковий номер каси та дату продажу товару.

Наступним етапом слід провести переформування даних, тобто ввести ієрархію продуктів. У кожній ієрархічній схемі будь-які товари мають безліч підвидів, однак у чеках інформація надаються лише про конкретний товар. Отже, наприклад слід об’єднати у себе однакові продукти, які відрізняються лише за назвою, в один товар, наприклад, «Макарони Вермішель Тая 450г» та «Макарони локшина довга Тая 450г» об’єднати в «Макарони». Приклад ієрархії зображений на рис (2.2).

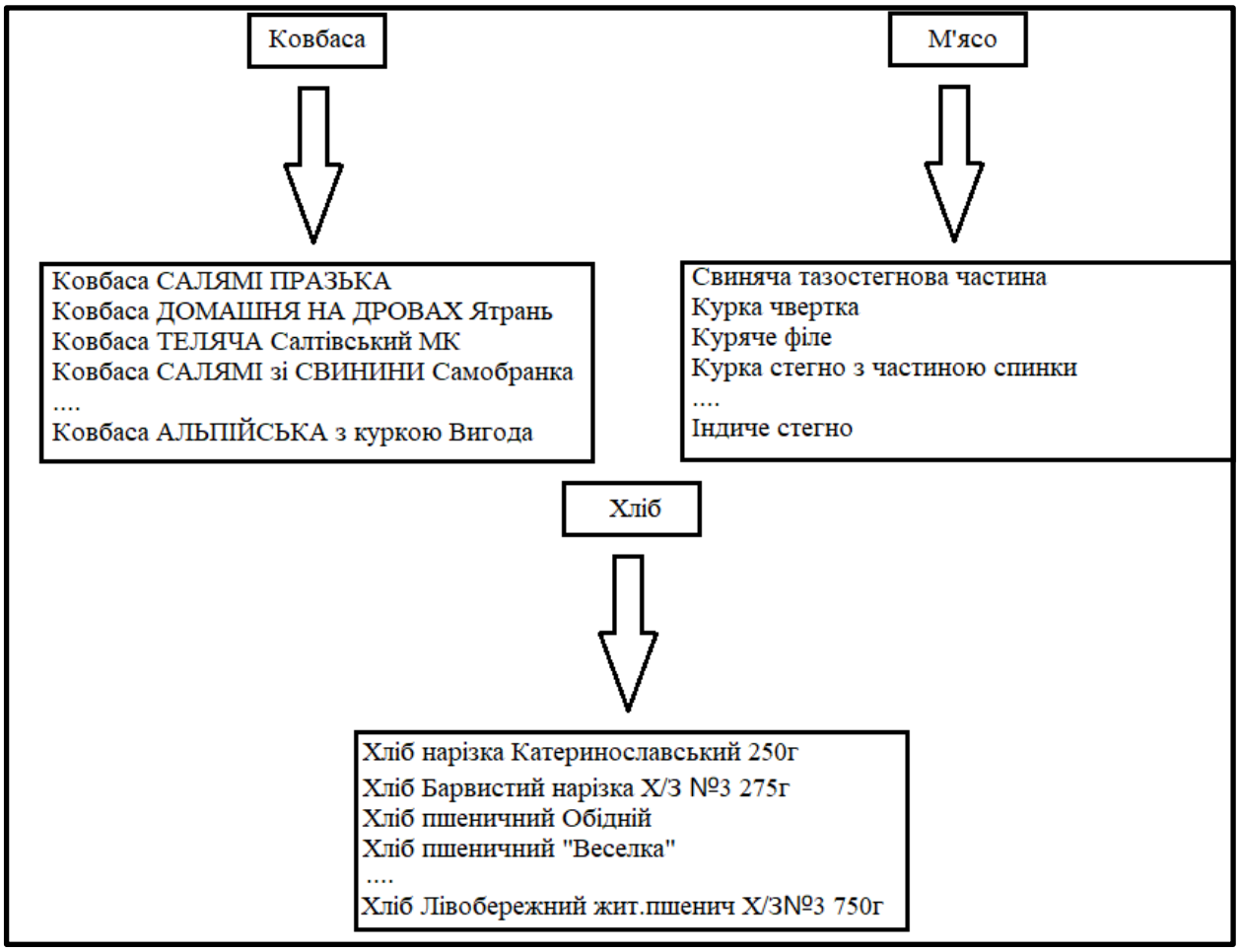


Рис. 2.2. Приклад ієрархії вхідних даних

Останній етап полягає у тому, щоб об’єднати кожний рядок, який несе у собі інформацію про порядковий номер чеку, порядковий номер товару та назву товару в одну клітинку, оскільки лише у такому вигляді дані підлягають аналізу для пошуку асоціативних правил (рис. 2.3).

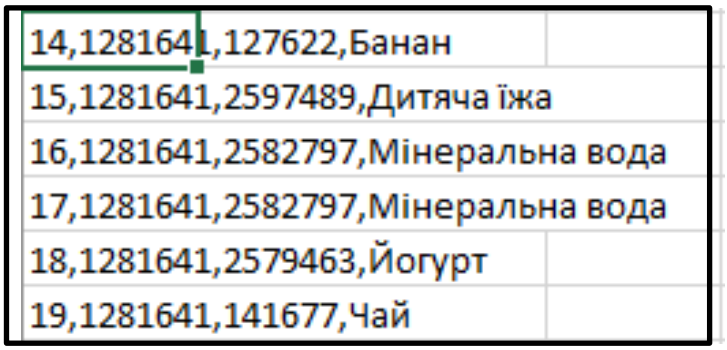


Рис. 2.3. Вхідні дані після обробки

2.3 Пошук асоціативних правил за методом Апріорі

Відштовхуючись від поставленої задачі, необхідно підібрати основні бібліотеки, які надають можливість керувати наданими даними, а також провести аналіз з метою отримання асоціативних правил на основі методу Апріорі.

1. Розглянемо початок коду, а саме частину, що відповідає за імпорт, тобто за під'єднання бібліотек:

```
import numpy as np
import pandas as pd
from itertools import permutations
from mlxtend.preprocessing import TransactionEncoder
from mlxtend.frequent_patterns import apriori,
association_rules
import matplotlib.pyplot as plt
import seaborn as sns
from pandas.plotting import parallel_coordinates
import networkx as nx
```

Використовуючи команду *import* імпортуємо наступні бібліотеки:

numpy (або скорочено *np*) – це бібліотека для чисельних обчислень на Python

pandas (або скорочено *pd*) – це бібліотека для обробки та аналізу даних у Python

itertools – це модуль, який надає різні функції для роботи з різними ітераторами, як наприклад ітераторами перестановки

TransactionEncoder від *mlxtend.preprocessing* – це клас для перетворення даних транзакції (наприклад, список списків, де кожен внутрішній список представляє транзакцію та містить придбані товари)

apriori та *association_rules* з *mlxtend.frequent_patterns* – це функції для генерації частих наборів елементів і правил асоціації з даних транзакцій

matplotlib.pyplot (або скорочено *plt*) – це бібліотека для створення статичних, інтерактивних та анімованих візуалізацій у Python

seaborn (скорочено *sns*) – це бібліотека для візуалізації статистичних даних на Python

parallel_coordinates від *pandas.plotting* – це функція для створення графіків паралельних координат

networkx (або скорочено *nx*) – це бібліотека для роботи з графіками та мережами на Python.

2. Після проведення імпорту бібліотек, починається наступний етап роботи коду, а саме - процес обробки вхідних даних, а також приведення їх до необхідного формату з метою подальшої роботи з ними.

```
df = pd.read_csv('C:/Users/User/Desktop/GoodsStore.csv')

df = df.drop(['key'],axis = 1) ,'date','cash_register'
df = df.drop_duplicates()

df.head()
```

Ця частина коду проводить збір даних у файлі CSV формату під назвою «GoodsStore.csv», що розташованому на робочому столі користувача, за допомогою функції *pd.read_csv()* із бібліотеки *pandas*.

Наступним етапом є видалення стовпця «key», «date», «cash_register» із фрейму даних за допомогою методу *.drop()* об'єкта *pandas.DataFrame*. Параметр осі має значення 1, щоб вказати, що стовпець видаляється. Через те, що початкові дані були імпортовані з загальної бази даних торгівельної мережі «Varus», вони можуть містити непотрібні, зайві дані, які не несуть інформаційної важливості.

Після видалення зайвих стовпців, переходимо до видалення за допомогою методу *.drop_duplicates()* будь-які повторювані рядки у фреймі даних. Видалення дублікатів необхідно для того, щоб була можливість проаналізувати дані без зайвих факторів, які можуть негативно вплинути на результат програми. Дана

процедура вважається обов’язковим етап обробки даних для використання методу *Apriori*.

Результат цієї частини коду можна побачити за допомогою методу *.head()*, який використовується для відображення кількох перших рядків результуючого кадру даних. З метою повного розуміння проведеної роботи слід провести порівняння формату вхідних даних із даними після першої обробки:

Дані до проведення форматування:

```
key, date, cach_register, check_id, product_id, product_name
1,2023-04-01,526,1281640,2628425,Хліб
2,2023-04-01,526,1281640,2628425,Хліб
3,2023-04-01,526,1281640,137820,М'ясо
4,2023-04-01,526,1281640,137820,М'ясо
5,2023-04-01,526,1281640,137820,М'ясо
6,2023-04-01,526,1281640,137820,М'ясо
7,2023-04-01,526,1281640,137820,М'ясо
8,2023-04-01,526,1281641,2597489,Дитяча їжа
9,2023-04-01,526,1281641,2579463,Йогурт
10,2023-04-01,526,1281641,2597489,Каша швидкого приготування
11,2023-04-01,526,1281641,2597489,Каша швидкого приготування
```

Дані після проведення форматування (рис 2.3):

	check_id	product_id	product_name
0	1281640	2628425	Хліб
2	1281640	137820	М'ясо
7	1281641	2597489	Дитяча їжа
8	1281641	2579463	Йогурт
9	1281641	2597489	Каша швидкого приготування

Рис. 2.4. Дані після проведення форматування

Отже, після проведення аналізу формату даних «до» та «після», можна прийти до висновку, що був скорочений загальний об’єм вхідних даних, що у свою чергу дає можливість не тільки провести більш детальний аналіз ринкового кошика, а також надати як можна більше унікальних даних для аналізу, оскільки максимально допустимий об’єм даних обмежено характеристиками персонального комп’ютера (ПК).

Важливо пам’ятати, що така велика торгівельна компанія, як «Варус», має потужний центр обробки даних (ЦОД), який у свою чергу складається з потужних серверів, основна мета яких полягає саме в обробці та зберігання терабайтів даних.

3. Наступним етапом роботи коду полягає в аналізі даних та визначення розподілу унікальних товарів у чеках

```
meandf = df.groupby(by=["check_id"],as_index=False).count()
meandf=meandf[meandf['product_id']>1]

plt.style.use('fivethirtyeight')
plt.figure(figsize=(8, 7))

for column in meandf.columns[1:]:
    plt.title("Розподіл унікальних товарів у чеках")
    sns.histplot(meandf[column], stat='density',
color='orange')
    sns.kdeplot(meandf[column], color='brown')
```

```
    plt.axvline(meandf[column].mean(), color='red',
linestyle='--', linewidth=0.8)
    min_ylim, max_ylim = plt.ylim()
    plt.text(meandf[column].mean()*1.05, max_ylim*0.96,
'Mean (μ): {:.2f}'.format(meandf[column].mean()))
    plt.xlabel("Кількість товарів", fontsize=12);
plt.ylabel("Щільність товарів за чеками", fontsize=12)
plt.show()

for check in df['check_id']:
    sum = 0
    for unic in meandf['check_id']:
        if check != unic:
            sum += 1

    if sum==meandf['check_id'].shape[0]:
        df = df[df['check_id']!=check]


print('Rows amount = {0}'.format(df.shape[0]))
print('Number of unique checks =
{0}'.format(df.groupby(["check_id"]).count().shape[0]))
print('Number of unique products =
{0}'.format(df.groupby(["product_id"]).count().shape[0]))
```

Дана частина коду відповідає за певний аналіз даних, а також фільтрує попередньо прочитаний *DataFrame df*.

Перші декілька рядків коду проводять групування *DataFrame* за *check_id*, а також проводять підрахунок кількості рядків у кожній групі за допомогою методу *count()*. Отриманий *DataFrame* призначається *meandf*.

Тоді лише ті рядки, де кількість *product_id* перевищує 1, зберігаються за допомогою *meandf=meandf[meandf['product_id']>1]*.

Наступним етапом роботи коду є побудова графік за допомогою *matplotlib* та *seaborn* для візуалізації розподілу кількості унікальних продуктів на перевірку. Цикл повторює кожен стовпець (тобто кожну перевірку) у *meandf* (починаючи з другого стовпця, оскільки перший стовпець містить ідентифікатори перевірок). Для кожного стовпця код створює гістограму значень у стовпці за допомогою

sns.histplot(), а також накладає графік оцінки щільності ядра (*KDE*) тих самих даних за допомогою *sns.kdeplot()* і додає вертикальну лінію в середньому значення стовпця за допомогою *plt.axvline()*. Середнє значення також відображається на графіку за допомогою *plt.text()*. Отриманий графік відображається за допомогою *plt.show()*.

Після графіка код фільтрує вихідний *DataFrame df*, щоб видалити перевірки, які містять унікальні продукти, яких немає в інших перевірках. Цей код проходить цикл по кожному *check_id* у *df* , а також підраховує кількість унікальних значень *check_id* у *meandf*, які не збігаються з поточним *check_id*.

Якщо ця кількість дорівнює числу унікальних значень *check_id* у *meandf,* це означає, що поточний *check_id* містить унікальні продукти, яких не було знайдено в інших перевірках. У цьому випадку відповідні рядки видаляються з *df* за допомогою *df = df[df['check_id']!=check]*. Нарешті, код виводить кількість рядків, унікальних перевірок і унікальних продуктів у результуючому *df*.

Отже, в результаті маємо наступний графік, що відображає результат розподілу унікальних товарів у чеках, а також надає коротку інформацію про надані дані (рис. 2.5):

Рис. 2.5. Графік розподілу унікальних товарів у чеках

Проаналізувавши отриманий результат, маємо наступні висновки:

У середньому в чеках від 4 до 5 унікальних товарів, при загальної кількості унікальних чеків у розмірі 549 одиниць, а також маємо загальну кількість унікальних товарів лише 80 одиниць.

Отриманий графік надає нам інформацію про додатню асиметрію, що є результатом того, що більшість чеків містять мало товарів, а деякі чеки містять дуже багато товарів. Такий розподіл називається правостороннім (або позитивно скошеним), оскільки "хвіст" розподілу знаходиться у правій частині графіка.

Це може бути пов'язано з тим, що деякі покупці роблять великі покупки, наприклад, для запасу на тривалий період часу або для проведення великого заходу, тоді як більшість покупців купують лише необхідну кількість товарів для своїх поточних потреб. Це може бути корисною інформацією для управління запасами та маркетингу, наприклад, для планування акцій на великі покупки або для створення зручніших умов для покупки великої кількості товарів [7].

4. Дана частина коду відповідає за побудови зведеної таблиці для подальшого аналізу даних

```
uid_sales =
df.pivot_table(values='product_id',columns=['product_name']
,index='check_id')#,aggfunc=np.sum)
uid_sales[np.isnan(uid_sales)] = 0
uid_sales[uid_sales>0] = 1
uid_sales.head()
```

Ця частина коду відповідає за побудову зведеної таблиці даних щодо продаж, де кожен рядок представляє собою унікальний ідентифікатор перевірки, а кожен стовпець представляє унікальну назву продукту. Значення в таблиці є підрахунком кожного продукту у відповідному чеку, але оскільки нас цікавить лише те, чи був продукт у чеку, кількість неважлива. Тому значення замінюються на 1, якщо продукт був присутній у чеку, і на 0 в іншому випадку.

Це створює двійкову таблицю, де кожна клітинка вказує, чи був певний продукт куплений за певним чеком. Цей тип таблиці корисний для створення правил асоціації між продуктами, які можуть допомогти визначити, які продукти часто купують разом, і запропонувати рекомендації щодо розміщення продукту, ціноутворення та рекламних акцій.

Розглянемо частину побудованої таблиці, результат роботи коду (рис 2.6):

product_name check_id	Ікра рибна	Авокадо	Банан	Батончик	Борошно	Бублики	Булка	Варення	Вафлі	Горілка
1281640	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
1281641	0.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
1281642	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
1281643	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
1281644	0.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

Рис. 2.6. Двійкова таблиця

5. Наступним етапом йде перша спроба визначена товарів, що частіше за всіх купують споживачі, а також розрахунок параметру «Підтримка (Support)»

```
freq_items = apriori(uid_sales, min_support=0.03,
use_colnames=True, max_len=2)
freq_items.head()
```

Дана частина коду генерує набір елементів за допомогою алгоритму Apriori. Дані *uid_sales* передаються як вхідні дані разом із мінімальним порогом підтримки 0,03 і параметром use_colnames встановлено значення *True*. Параметр *max_len* також має значення 2, що означає, що ми хочемо генерувати лише набори елементів довжиною до 2 (тобто пари продуктів).

Результатом цього коду є DataFrame, що містить список товарів, які купують частіше за інші, а також відповідні їм значення підтримки. Значення підтримки для кожного товару представляє частку транзакцій (тобто чеків), у яких з’являється набір елементів. Наприклад, якщо значення підтримки для набору елементів становить 0,05, це означає, що набір елементів з’являється в 5% усіх транзакцій.

DataFrame має два стовпці: «підтримка», який містить значення підтримки для кожного набору елементів, і «набір елементів», який містить набори елементів, які часто з’являються разом у транзакціях.

Стовпець «itemsets» є замороженим об’єктом, що означає, що елементи в кожному наборі елементів не впорядковані та не можуть бути змінені.

Результатом цієї частини коду є наступна таблиця (рис 2.7):

	support	itemsets
0	0.036430	(Ікра рибна)
1	0.100182	(Банан)
2	0.060109	(Батончик)
3	0.076503	(Булка)
4	0.036430	(Гриби)

Рис. 2.7. Результат роботи частини коду

6. Після цього йде визначення та формування частих наборів елементів та розрахунок параметру усіх необхідних параметрів

```
rules = association_rules(freq_items, metric =
"confidence", min_threshold = 0)
rules.sort_values(by = ["support","confidence"], ascending
= False)
```

Ця частина код генерує правила асоціації з частих наборів елементів, згенерованих раніше за допомогою алгоритму Apriori. *Freq_items* DataFrame передається як вхідні дані разом із параметром метрики, встановленим на «confidence», і параметром *min_threshold*, встановленим на 0.

Вихідним результатом є DataFrame, що містить правила асоціації та їхню відповідну підтримку, довіру, підйом та інші показники інтересу.

Метод *sort_values* використовується для сортування правил за зменшенням підтримки та впевненості, а метод *head* використовується для показу 3 найпопулярніших правил.

«Підтримка (Support)» правила — це частка транзакцій, які містять як попередній, так і наслідок правила. «Достовірність (Confidence)» правила — це частка транзакцій, що містять антецедент, які також містять результат. «Підйом (Lift)» правила – це відношення спостережуваної підтримки правила до очікуваної підтримки за припущення, що антецедент і наслідок є незалежними.

Встановлюючи для параметра *min_threshold* значення 0, ми повідомляємо функції *association_rules* повертати всі правила, незалежно від їх надійності (рис 2.8).

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction
94	(М'ясо)	(Хліб)	0.404372	0.468124	0.218579	0.540541	1.154696	0.029283	1.157613
95	(Хліб)	(М'ясо)	0.468124	0.404372	0.218579	0.466926	1.154696	0.029283	1.117347
62	(Ковбаса)	(Хліб)	0.431694	0.468124	0.211293	0.489451	1.045560	0.009207	1.041774
63	(Хліб)	(Ковбаса)	0.468124	0.431694	0.211293	0.451362	1.045560	0.009207	1.035849
37	(М'ясо)	(Ковбаса)	0.404372	0.431694	0.185792	0.459459	1.064317	0.011228	1.051366
...	...	...	...	...	...	...	...	...	...
135	(Сік)	(Чіпси)	0.162113	0.100182	0.030965	0.191011	1.906639	0.014725	1.112275
109	(Мінеральна вода)	(Морозиво)	0.183971	0.103825	0.030965	0.168317	1.621157	0.011865	1.077544
13	(М'ясо)	(Гриби)	0.404372	0.036430	0.030965	0.076577	2.102027	0.016234	1.043476
82	(М'ясо)	(Огірок)	0.404372	0.045537	0.030965	0.076577	1.681622	0.012551	1.033613
99	(М'ясо)	(Чіпси)	0.404372	0.100182	0.030965	0.076577	0.764373	-0.009545	0.974437

Рис. 2.8. Асоціативні правила

7. Далі необхідно провести графічний аналіз попередньої та наступної підтримки, що визначаються на основі отриманих даних

```
sns.set_context("talk")
sns.relplot(x='antecedent support', y='consequent support',
data=rules,
            size='lift', hue='confidence', height=6,
aspect=2)
plt.title("Antecedent Support v.s. Consequent Support",
fontsize=16, y=1.02)
plt.xlabel('Antecedent Support', fontsize=12)
plt.ylabel('Consequent Support', fontsize=12)
plt.show()
```

Даний графік показує зв’язок між попередньою підтримкою та наступною підтримкою правил зв’язування, створених алгоритмом Apriori. Кожна точка на графіку представляє правило, і розмір точки відповідає висоті правила, тоді як відтінок представляє достовірність правила.

Сюжет дозволяє нам визначити сильні правила з високою підтримкою та впевненістю, а також високим підйомом. Загалом, ми хотіли б бачити високі значення для всіх трьох показників, щоб мати сильні та діючі правила.

У цьому конкретному сюжеті ми бачимо, що існує кілька правил із високим підйомом, які вказують на те, що зв’язок між антецедентом і наслідком є сильнішим, ніж те, що ми очікували випадково. Крім того, ми бачимо, що багато правил мають високу достовірність, що вказує на те, що антецедент є хорошим провісником наслідків.

Загалом, цей графік надає корисний візуальний підсумок правил асоціації, згенерованих алгоритмом Apriori, що дозволяє нам швидко ідентифікувати цікаві та потенційно дієві шаблони в даних.

Результатом даної частини коду є наступний графік, який надає візуальний підсумок на основі існуючих даних правил асоціації (рис 2.9).

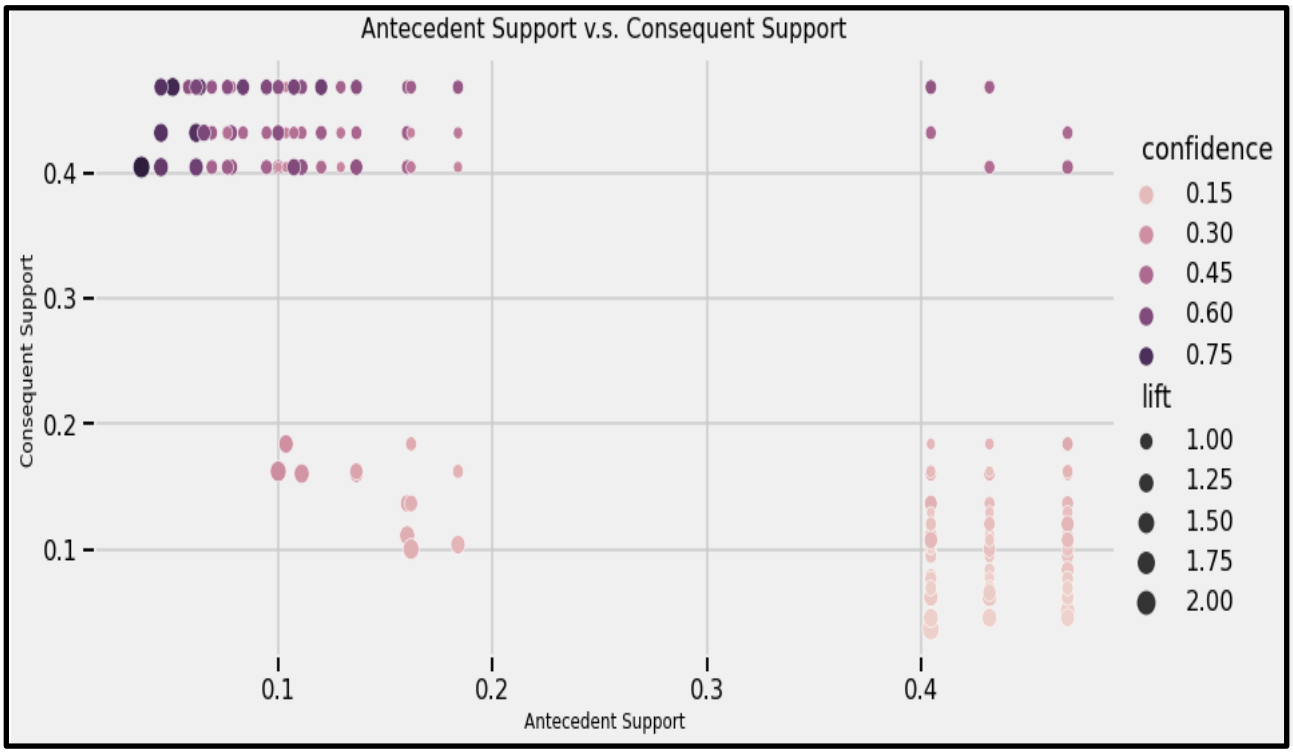


Рис. 2.9. Графік відношення попередньої підтримки до наступної

8. Наступним етапом йде розрахунок параметру Zhang для кожного визначеного правила асоціації

```
def zhangs_rule(rules):
    rule_support = rules['support'].copy()
    rule_ante = rules['antecedent support'].copy()
    rule_conseq = rules['consequent support'].copy()
    num = rule_support - (rule_ante * rule_conseq)
    denom = np.max((rule_support * (1 - rule_ante).values,
                    rule_ante * (rule_conseq -
rule_support).values), axis = 0)
    return num / denom

rules_zhangs_list = zhangs_rule(rules)
rules = rules.assign(zhang = rules_zhangs_list)
rules.sort_values(by = ["support","confidence"], ascending
= False)
```

Наведений вище код обчислює значення правила Zhang для кожного правила асоціації та додає його як новий стовпець під назвою «zhang» до кадру даних правил асоціації. Правило Zhang — це метрика, яка поєднує підтримку,

попередню підтримку та наступну підтримку для вимірювання цікавості правила асоціації.

Вищі значення правила Zhang вказують на сильніші та цікавіші правила. Потім код сортує правила за підтримкою та достовірністю в порядку спадання та спочатку показує 5 найвищих правила з найвищою підтримкою та достовірністю разом із відповідними значеннями правила Zhang, всього ми маємо 144 правила (рис 2.9).

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction	zhang
94	(М'ясо)	(Хліб)	0.404372	0.468124	0.218579	0.540541	1.154696	0.029283	1.157613	0.224924
95	(Хліб)	(М'ясо)	0.468124	0.404372	0.218579	0.466926	1.154696	0.029283	1.117347	0.251884
62	(Ковбаса)	(Хліб)	0.431694	0.468124	0.211293	0.489451	1.045560	0.009207	1.041774	0.076674
63	(Хліб)	(Ковбаса)	0.468124	0.431694	0.211293	0.451362	1.045560	0.009207	1.035849	0.081926
37	(М'ясо)	(Ковбаса)	0.404372	0.431694	0.185792	0.459459	1.064317	0.011228	1.051366	0.101457
...	...	...	...	...	...	...	...	...	...	...
135	(Сік)	(Чіпси)	0.162113	0.100182	0.030965	0.191011	1.906639	0.014725	1.112275	0.567519
109	(Мінеральна вода)	(Морозиво)	0.183971	0.103825	0.030965	0.168317	1.621157	0.011865	1.077544	0.469538
13	(М'ясо)	(Гриби)	0.404372	0.036430	0.030965	0.076577	2.102027	0.016234	1.043476	0.880194
82	(М'ясо)	(Огірок)	0.404372	0.045537	0.030965	0.076577	1.681622	0.012551	1.033613	0.680518
99	(М'ясо)	(Чіпси)	0.404372	0.100182	0.030965	0.076577	0.764373	-0.009545	0.974437	-0.341038

144 rows × 10 columns

Рис. 2.10. Асоціативні правила з параметром Zhang

9. Наступним й останнім етапом є побудова графіка, що надає можливість візуально проаналізувати який набір товарів купляються найчастіше.

```
rules['antecedents'] = rules['antecedents'].apply(lambda a:
', '.join(list(a)))
rules['consequents'] = rules['consequents'].apply(lambda a:
', '.join(list(a)))

pivot_support = rules.pivot(index='antecedents',
columns='consequents', values='support')

sns.set_context("talk")
plt.style.use('ggplot')
plt.subplots(figsize=(12, 16))
sns.set()
ax = sns.heatmap(data=pivot_support, annot=True, fmt='.2f',
cmap='YlGnBu', cbar=True)
plt.title("Items' Support Matrix", fontsize=16, y=1.02)
ax.set_xlabel("Consequents",fontsize=16)
ax.set_ylabel("Antecedents",fontsize=16)
plt.xticks(rotation=45)
plt.yticks(rotation=0)
plt.show()
```

На графіку ми можемо бачити матрицю підтримки товарів (Items' Support Matrix), яка показує, які товари найчастіше купують разом. Верхній ряд графіка показує імена товарів, які є результатом правила (consequents), а лівий стовпець показує імена товарів, що стоять як умова (antecedents) [7].

Чим темніший колір комірки, тим вища підтримка відповідних клей, а білий колір відповідає відсутності даних (рис 2.11).

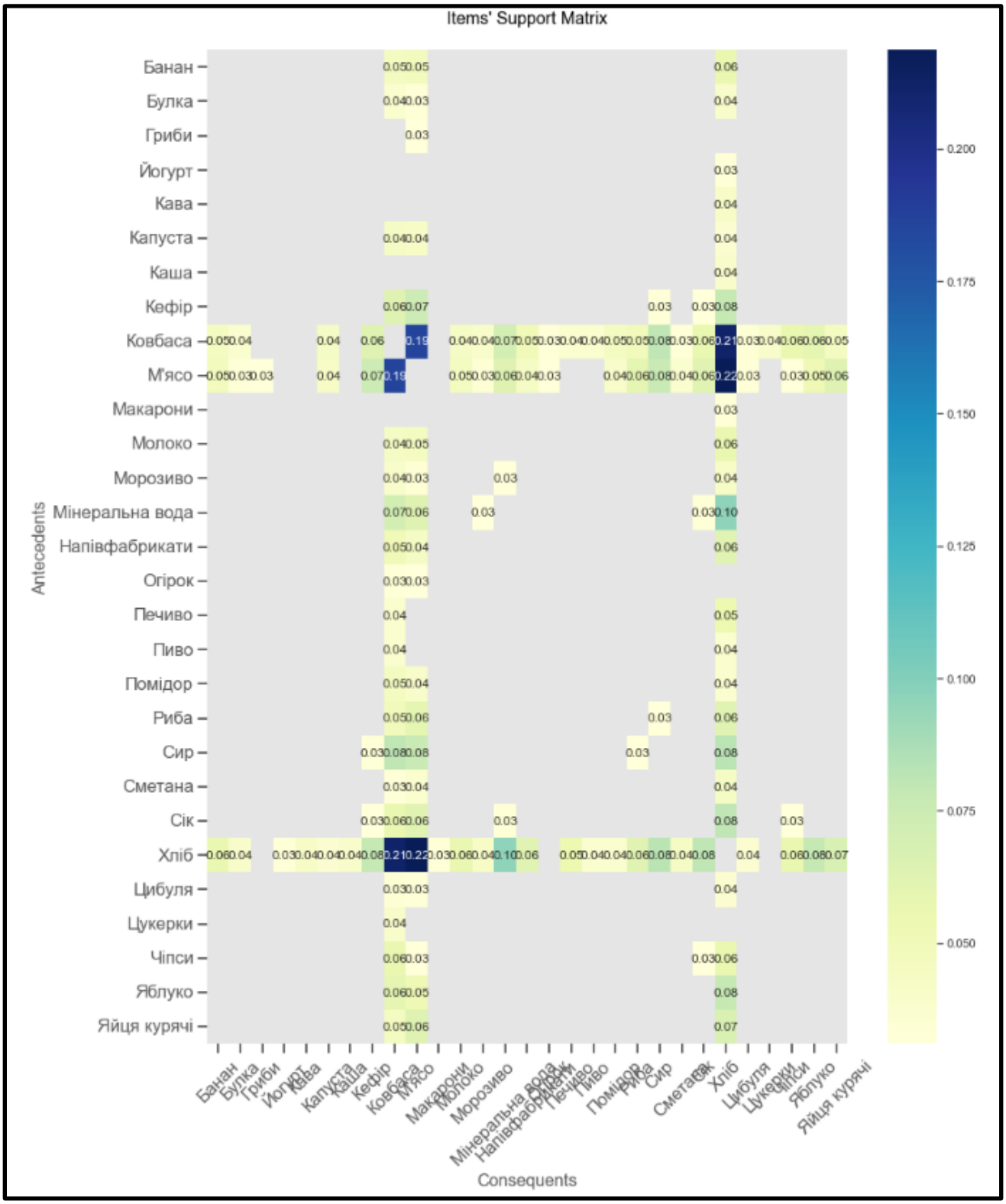


Рис. 2.11. Матриця «підтримки» товарів

10. Остання частина коду відповідає за побудову мережевої діаграми, що у свою чергу проводить процес візуалізації частих наборів товарів:

```
plt.figure(figsize=(10, 10))
bk_rules = rules
bk_network = bk_rules[['antecedents', 'consequents']]

bk_network_G = nx.from_pandas_edgelist(
    bk_network,
    source = 'antecedents',
    target = 'consequents',
    create_using = nx.DiGraph())

network_bt = nx.in_degree_centrality(bk_network_G)
bk_network_bt = pd.DataFrame(list(network_bt.items()),
columns = ['item', 'centrality'])
bk_network_bt = bk_network_bt.dropna(axis=0)

pos = nx.kamada_kawai_layout(bk_network_G)

sizes = [x[1]*100 for x in bk_network_G.degree()]

nx.draw_networkx(bk_network_G, pos,
    with_labels = True,
    node_size = sizes,
    width = 0.1, alpha = 0.8,
    arrowsize = 3, linewidths = 0)

plt.axis('off')
plt.show()
```

Цей код створює мережеву діаграму частих наборів елементів і правил асоціації за допомогою бібліотеки *NetworkX* у Python. Код починається з вилучення антецедентів і консеквентів із фрейму даних правил асоціації та створення спрямованого графіка за допомогою функції *from_pandas_edgelist*.

Далі він обчислює центральність кожного вузла за ступенем за допомогою функції *in_degree_centrality*, яка вимірює, скільки ребер спрямовано до вузла. Потім значення центральності зберігаються в новому фреймі даних під назвою

bk_network_bt, який використовується для встановлення розміру кожного вузла в мережевому графіку.

Нарешті, мережевий графік малюється за допомогою функції *nx.draw_networkx* із параметрами розміру вузла, ширини краю та розміру стрілки.

Розташування вузлів визначається за допомогою алгоритму Камада-Каваї, який намагається мінімізувати кількість перетинів ребер у графі. Потім отриманий графік відображається за допомогою *plt.show()* (рис. 2.12).

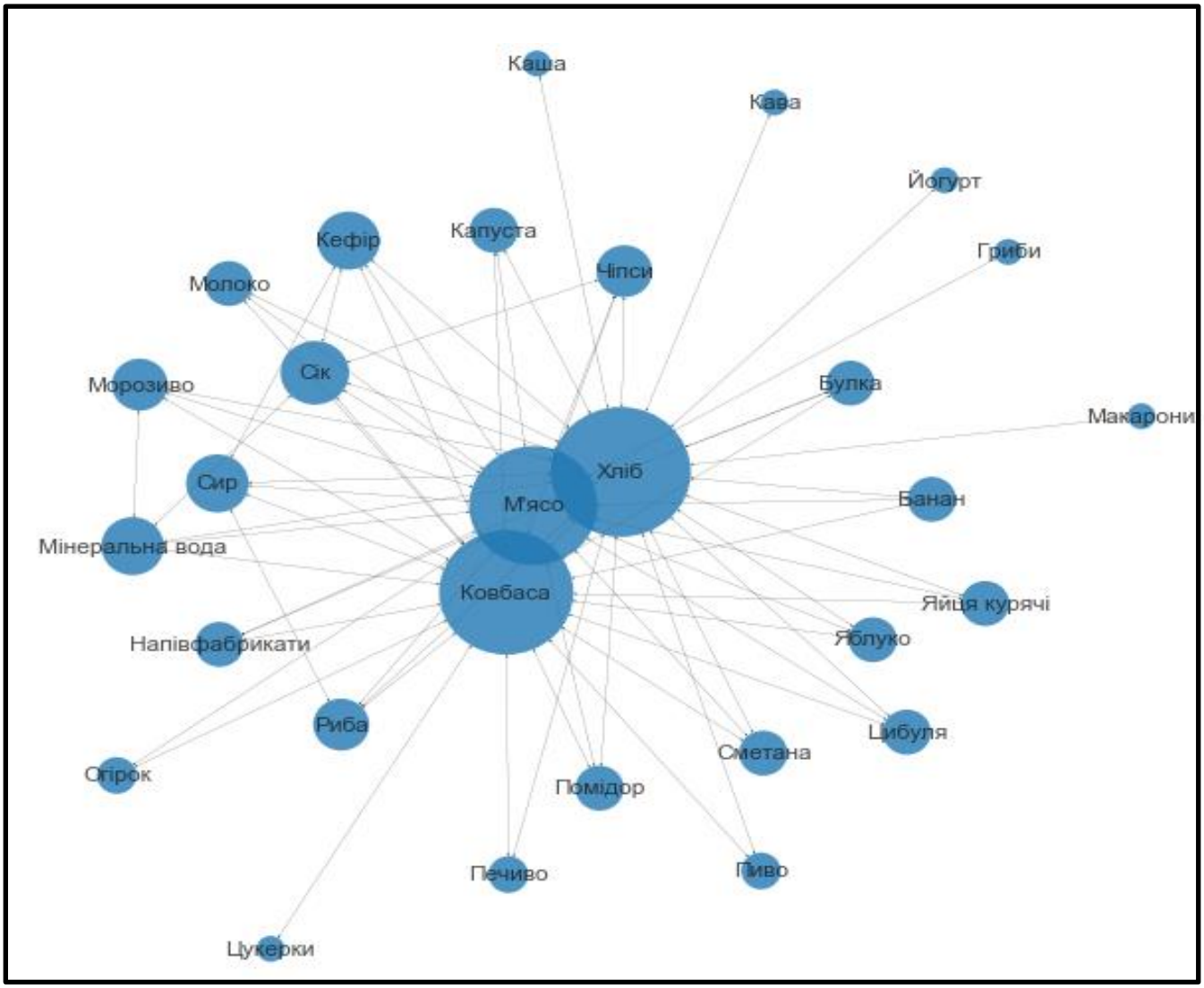


Рис. 2.12. Мережева діаграма частих наборів елементів

2.4 Пошук асоціативних правил за методом Табу-пошук

Відштовхуючись від поставленої задачі, необхідно підібрати основні бібліотеки, які надають можливість керувати наданими даними, а також провести аналіз з метою отримання асоціативних правил на основі методу Табу.

1. Розглянемо початок коду, а саме частину, що відповідає за імпорт, тобто за під'єднання бібліотек:

```
import pandas as pd
import numpy as np
from mlxtend.frequent_patterns import fpgrowth
from mlxtend.frequent_patterns import association_rules
```

Використовуючи команду *import* імпортуємо бібліотеки Python, які використовуються для аналізу та видобутку частих наборів елементів і правил асоціації з наборів даних:

pandas — це популярна бібліотека для обробки та аналізу даних, яка надає гнучкі структури даних та інструменти для роботи зі структурованими даними.

numpy — це бібліотека для чисельних обчислень на Python.

mlxtend — це пакет, який надає інструменти для машинного навчання, інтелектуального аналізу даних і дослідницького аналізу даних.

fpgrowth — це алгоритм для частого видобутку набору елементів і швидший за Apriori.

association_rules — це метод для створення правил асоціації з часто використовуваних наборів елементів.

2. Наступний етап полягає в імпорті вхідних даних з основного CSV файлу:

```
df = pd.read_csv('C:/Users/User/Desktop/GoodsStore.csv')

uid_sales =
df.pivot_table(values='product_id',columns=['product_name']
,index='check_id')
uid_sales[np.isnan(uid_sales)] = 0
uid_sales[uid_sales>0] = 1
```

Ця частина коду читає файл CSV під назвою «GoodsStore.csv» і зберігає дані у Pandas DataFrame під назвою «*df*». Потім він створює новий DataFrame під назвою «*uid_sales*», обертаючи дані в «*df*» таким чином, що кожен рядок представляє унікальний *check_id*, кожен стовпець представляє унікальну назву продукту, а значення в таблиці показують, чи було придбано певний продукт. у даному чеку, тобто 1 означає, що товар було придбано, а 0 означає, що він не був придбаний.

Код замінює будь-які значення *NaN* у DataFrame на 0, а потім перетворює будь-які ненульові значення на 1 для створення двійкової матриці.

3. Далі переходимо до перетворення даних до одноразового кодового формату:

```
df_onehot = uid_sales
```

4. Задаємо параметри пошуку асоціативних правил за допомогою бібліотеки *fpgrowth:*

```
frequent_itemsets = fpgrowth(df_onehot, min_support=0.05,
use_colnames=True)
```

Функція *fpgrowth* з бібліотеки *mlxtend* використовується для частого майнінгу *itemset* за допомогою алгоритму *FP-growth*. У цьому випадку ми передаємо фрейм даних з одноразовим кодуванням *df_onehot* і встановлюємо

мінімальний поріг підтримки на 0,05 за допомогою параметра *min_support*. Для параметра *use_colnames* встановлено значення *True*, щоб використовувати назви стовпців вихідного кадру даних у виводі.

Отриманий фрейм даних *repeat_itemsets* містить усі часті набори елементів, які відповідають мінімальному порогу підтримки, а також їхні відповідні значення підтримки.

5. Останній етап полягає у використанні методу Табу для пошуку асоціативних правил:

```
def tabu_search(frequent_itemsets, num_iterations,
tabu_length):
    tabu_list = []
    best_itemset = frequent_itemsets.copy()
    best_support = best_itemset.iloc[0]['support']
    for i in range(num_iterations):
        neighbors = []
        for itemset in best_itemset['itemsets']:
            for item in itemset:
                neighbor = itemset - frozenset([item])
                if neighbor not in tabu_list:
                    neighbors.append(neighbor)
        best_neighbor = None
        best_neighbor_support = 0
        for neighbor in neighbors:
            if
frequent_itemsets[frequent_itemsets['itemsets'] ==
neighbor].empty==False:
                neighbor_support =
frequent_itemsets[frequent_itemsets['itemsets'] ==
neighbor]['support'].values[0]
```

```
                if neighbor_support >
best_neighbor_support:
                    best_neighbor = neighbor
                    best_neighbor_support =
neighbor_support
        if best_neighbor_support > best_support:
            best_itemset =
frequent_itemsets[frequent_itemsets['itemsets'] ==
best_neighbor]
            best_support = best_neighbor_support
        tabu_list.append(best_itemset.iloc[0]['itemsets'])
        if len(tabu_list) > tabu_length:
            tabu_list.pop(0)
    return best_itemset


tabu_itemset = tabu_search(frequent_itemsets,
num_iterations=100, tabu_length=10)


rules = association_rules(tabu_itemset, metric="lift",
min_threshold=1)


print(rules.sort_values(by = ["support","confidence"],
ascending=False))
```

Функція *tabu_search* приймає три параметри:

pogost_itemsets — фрейм даних із частими наборами елементів, згенерований за допомогою алгоритму *fpgrowth*;

num_iterations — кількість ітерацій, які потрібно виконати;

tabu_length — довжина списку табу, який використовується в алгоритмі.

Потім функція ініціалізує список табу і встановлює найкращий набір елементів як той, що має найвищу підтримку в кадрі даних із частими наборами елементів [11]. Потім він переходить до виконання ітерацій *num_iterations*.

У кожній ітерації він генерує сусідів поточного найкращого набору елементів, видаляючи один елемент за раз, і перевіряє, чи є кожен сусід у списку табу.

Якщо ні, він обчислює підтримку сусіда та вибирає сусіда з найвищою підтримкою як новий найкращий набір елементів, якщо його підтримка вища за поточну найкращу підтримку. Потім він додає поточний найкращий набір елементів до списку табу та видаляє найстаріший елемент зі списку, якщо його довжина перевищує *tabu_length*.

Нарешті, функція повертає найкращий набір елементів, знайдений після всіх ітерацій.

Потім змінній tabu_itemset призначається найкращий набір елементів, знайдений за допомогою виклику *tabu_search* із *frequent_itemsets, num_iterations,* встановленою на 100, і *tabu_length*, встановленою на 10.

Нарешті, функція *association_rules* з бібліотеки *mlxtend* використовується для генерації правил асоціації з наборів елементів *tabu_itemset* із використанням мінімального порогового значення 1. Отриманий фрейм даних правил сортується за підтримкою та достовірністю в порядку спадання [3].

Результатом даного методу є виявлення 40 асоціативних правил (рис 2.13 та рис. 2.14):

	antecedents	consequents	antecedent support
1	(М'ясо)	(Хліб)	0.347305
0	(Хліб)	(М'ясо)	0.408683
17	(Ковбаса)	(Хліб)	0.383234
16	(Хліб)	(Ковбаса)	0.408683
2	(М'ясо)	(Ковбаса)	0.347305
3	(Ковбаса)	(М'ясо)	0.383234
6	(М'ясо, Ковбаса)	(Хліб)	0.152695
5	(Хліб, Ковбаса)	(М'ясо)	0.173653
4	(Хліб, М'ясо)	(Ковбаса)	0.179641
8	(М'ясо)	(Хліб, Ковбаса)	0.347305
9	(Ковбаса)	(Хліб, М'ясо)	0.383234
7	(Хліб)	(М'ясо, Ковбаса)	0.408683
11	(Мінеральна вода)	(Хліб)	0.188623
10	(Хліб)	(Мінеральна вода)	0.408683
22	(Сир)	(Хліб)	0.133234
23	(Хліб)	(Сир)	0.408683
24	(Сир)	(Ковбаса)	0.133234
13	(Сік)	(Хліб)	0.139222
25	(Ковбаса)	(Сир)	0.383234
12	(Хліб)	(Сік)	0.408683
19	(Яблуко)	(Хліб)	0.098802
37	(Кефір)	(Хліб)	0.113772
18	(Хліб)	(Яблуко)	0.408683
36	(Хліб)	(Кефір)	0.408683
26	(Сир)	(М'ясо)	0.133234
27	(М'ясо)	(Сир)	0.347305
39	(Кефір)	(М'ясо)	0.113772
38	(М'ясо)	(Кефір)	0.347305
31	(Яйця курячі)	(Хліб)	0.088323
30	(Хліб)	(Яйця курячі)	0.408683
33	(Яйця курячі)	(М'ясо)	0.088323
34	(Кефір)	(Ковбаса)	0.113772
32	(М'ясо)	(Яйця курячі)	0.347305
35	(Ковбаса)	(Кефір)	0.383234
21	(Риба)	(Хліб)	0.095808
29	(Напівфабрикати)	(Хліб)	0.107784
15	(Сік)	(М'ясо)	0.139222
14	(М'ясо)	(Сік)	0.347305
20	(Хліб)	(Риба)	0.408683
28	(Хліб)	(Напівфабрикати)	0.408683

Рис. 2.13. Асоціативні правила (1)

	consequent support	support	confidence	lift	leverage	conviction
1	0.408683	0.179641	0.517241	1.265631	0.037703	1.224872
0	0.347305	0.179641	0.439560	1.265631	0.037703	1.164612
17	0.408683	0.173653	0.453125	1.108745	0.017032	1.081266
16	0.383234	0.173653	0.424908	1.108745	0.017032	1.072467
2	0.383234	0.152695	0.439655	1.147225	0.019596	1.100691
3	0.347305	0.152695	0.398438	1.147225	0.019596	1.084999
6	0.408683	0.089820	0.588235	1.439345	0.027417	1.436056
5	0.347305	0.089820	0.517241	1.489298	0.029510	1.352010
4	0.383234	0.089820	0.500000	1.304688	0.020976	1.233533
8	0.173653	0.089820	0.258621	1.489298	0.029510	1.114608
9	0.179641	0.089820	0.234375	1.304688	0.020976	1.071490
7	0.152695	0.089820	0.219780	1.439345	0.027417	1.085983
11	0.408683	0.079341	0.420635	1.029246	0.002254	1.020630
10	0.188623	0.079341	0.194139	1.029246	0.002254	1.006845
22	0.408683	0.067365	0.505618	1.237190	0.012915	1.196074
23	0.133234	0.067365	0.164835	1.237190	0.012915	1.037839
24	0.383234	0.065868	0.494382	1.290028	0.014809	1.219827
13	0.408683	0.065868	0.473118	1.157667	0.008971	1.122296
25	0.133234	0.065868	0.171875	1.290028	0.014809	1.046661
12	0.139222	0.065868	0.161172	1.157667	0.008971	1.026168
19	0.408683	0.064371	0.651515	1.594184	0.023992	1.696824
37	0.408683	0.064371	0.565789	1.384423	0.017874	1.361822
18	0.098802	0.064371	0.157509	1.594184	0.023992	1.069682
36	0.113772	0.064371	0.157509	1.384423	0.017874	1.051914
26	0.347305	0.062874	0.471910	1.358776	0.016602	1.235954
27	0.133234	0.062874	0.181034	1.358776	0.016602	1.058367
39	0.347305	0.061377	0.539474	1.553312	0.021863	1.417280
38	0.113772	0.061377	0.176724	1.553312	0.021863	1.076465
31	0.408683	0.055389	0.627119	1.534488	0.019293	1.585806
30	0.088323	0.055389	0.135531	1.534488	0.019293	1.054609
33	0.347305	0.052395	0.593220	1.708065	0.021720	1.604541
34	0.383234	0.052395	0.460526	1.201686	0.008794	1.143274
32	0.088323	0.052395	0.150862	1.708065	0.021720	1.073650
35	0.113772	0.052395	0.136719	1.201686	0.008794	1.026580
21	0.408683	0.050898	0.531250	1.299908	0.011743	1.261477
29	0.408683	0.050898	0.472222	1.155474	0.006849	1.120391
15	0.347305	0.050898	0.365591	1.052651	0.002546	1.028824
14	0.139222	0.050898	0.146552	1.052651	0.002546	1.008589
20	0.095808	0.050898	0.124542	1.299908	0.011743	1.032821
28	0.107784	0.050898	0.124542	1.155474	0.006849	1.019142

Рис. 2.14. Асоціативні правила (2)

2.5 Порівняння результатів з додатковими даними

Після проведення аналізу даних за методом Апріорі було виявлено 144 правил асоціації, які були відсортовані за порядком спадання параметрів **«support»** та **«confidence»** (табл. 2.1):

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction	zhang
94	(М'ясо)	(Хліб)	0.404372	0.468124	0.218579	0.540541	1.154696	0.029283	1.157613	0.224924
95	(Хліб)	(М'ясо)	0.468124	0.404372	0.218579	0.466926	1.154696	0.029283	1.117347	0.251884
62	(Ковбаса)	(Хліб)	0.431694	0.468124	0.211293	0.489451	1.045560	0.009207	1.041774	0.076674
63	(Хліб)	(Ковбаса)	0.468124	0.431694	0.211293	0.451362	1.045560	0.009207	1.035849	0.081926
37	(М'ясо)	(Ковбаса)	0.404372	0.431694	0.185792	0.459459	1.064317	0.011228	1.051366	0.101457
36	(Ковбаса)	(М'ясо)	0.431694	0.404372	0.185792	0.430380	1.064317	0.011228	1.045659	0.106335
114	(Мінеральна вода)	(Хліб)	0.183971	0.468124	0.096539	0.524752	1.120969	0.010418	1.119156	0.132244
115	(Хліб)	(Мінеральна вода)	0.468124	0.183971	0.096539	0.206226	1.120969	0.010418	1.028037	0.202895
128	(Сир)	(Хліб)	0.160291	0.468124	0.081967	0.511364	1.092368	0.006931	1.088491	0.100699
129	(Хліб)	(Сир)	0.468124	0.160291	0.081967	0.175097	1.092368	0.006931	1.017949	0.158980
56	(Сир)	(Ковбаса)	0.160291	0.431694	0.080146	0.500000	1.158228	0.010949	1.136612	0.162690
132	(Сік)	(Хліб)	0.162113	0.468124	0.080146	0.494382	1.056092	0.004257	1.051933	0.063389
57	(Ковбаса)	(Сир)	0.431694	0.160291	0.080146	0.185654	1.158228	0.010949	1.031145	0.240385
133	(Хліб)	(Сік)	0.468124	0.162113	0.080146	0.171206	1.056092	0.004257	1.010972	0.099860
140	(Яблуко)	(Хліб)	0.120219	0.468124	0.078324	0.651515	1.391758	0.022047	1.526253	0.319948
34	(Кефір)	(Хліб)	0.136612	0.468124	0.078324	0.573333	1.224747	0.014373	1.246585	0.212540
35	(Хліб)	(Кефір)	0.468124	0.136612	0.078324	0.167315	1.224747	0.014373	1.036872	0.345014
141	(Хліб)	(Яблуко)	0.468124	0.120219	0.078324	0.167315	1.391758	0.022047	1.056560	0.529229
88	(Сир)	(М'ясо)	0.160291	0.404372	0.076503	0.477273	1.180283	0.011685	1.139463	0.181903
89	(М'ясо)	(Сир)	0.404372	0.160291	0.076503	0.189189	1.180283	0.011685	1.035641	0.256444
28	(Кефір)	(М'ясо)	0.136612	0.404372	0.074681	0.546667	1.351892	0.019439	1.313886	0.301482
29	(М'ясо)	(Кефір)	0.404372	0.136612	0.074681	0.184685	1.351892	0.019439	1.058962	0.437011
42	(Мінеральна вода)	(Ковбаса)	0.183971	0.431694	0.071038	0.386139	0.894473	-0.008381	0.925789	-0.126313
43	(Ковбаса)	(Мінеральна вода)	0.431694	0.183971	0.071038	0.164557	0.894473	-0.008381	0.976762	-0.171907
142	(Яйця курячі)	(Хліб)	0.107468	0.468124	0.067395	0.627119	1.339643	0.017087	1.426395	0.284060
143	(Хліб)	(Яйця курячі)	0.468124	0.107468	0.067395	0.143969	1.339643	0.017087	1.042640	0.476675
102	(Яйця курячі)	(М'ясо)	0.107468	0.404372	0.063752	0.593220	1.467018	0.020295	1.464253	0.356676
26	(Кефір)	(Ковбаса)	0.136612	0.431694	0.063752	0.466667	1.081013	0.004778	1.065574	0.086799

Таблиця 2.1. Асоціативні правила за методом Апріорі

Дана таблиця містить у собі не повний результат, оскільки нас цікавить лише перші шість рядків. Саме ці перші шість правил показуються найкращий результат, який залежить зазвичай від показників «support» та «confidence», які

передають інформацію стосовно того, скільки разів зустрічався той чи інший набір елементів у чеках, а також ймовірність того, що при покупці одного товару, придбають також й інший.

Після проведення аналізу даних за методом Табу-пошук було виявлено лише 40 правил асоціації, які також були відсортовані за порядком спадання параметрів «support» та «confidence» (табл. 2.2):

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction
1	(М'ясо)	(Хліб)	0.347305	0.408683	0.179641	0.517241	1.265631	0.037703	1.224872
0	(Хліб)	(М'ясо)	0.408683	0.347305	0.179641	0.439560	1.265631	0.037703	1.164612
17	(Ковбаса)	(Хліб)	0.383234	0.408683	0.173653	0.453125	1.108745	0.017032	1.081266
16	(Хліб)	(Ковбаса)	0.408683	0.383234	0.173653	0.424908	1.108745	0.017032	1.072467
2	(М'ясо)	(Ковбаса)	0.347305	0.383234	0.152695	0.439655	1.147225	0.019596	1.100691
3	(Ковбаса)	(М'ясо)	0.383234	0.347305	0.152695	0.398438	1.147225	0.019596	1.084999
6	(М'ясо, Ковбаса)	(Хліб)	0.152695	0.408683	0.089820	0.588235	1.439345	0.027417	1.436056
5	(Хліб, Ковбаса)	(М'ясо)	0.173653	0.347305	0.089820	0.517241	1.489298	0.029510	1.352010
4	(Хліб, М'ясо)	(Ковбаса)	0.179641	0.383234	0.089820	0.500000	1.304688	0.020976	1.233533
8	(М'ясо)	(Хліб, Ковбаса)	0.347305	0.173653	0.089820	0.258621	1.489298	0.029510	1.114608
9	(Ковбаса)	(Хліб, М'ясо)	0.383234	0.179641	0.089820	0.234375	1.304688	0.020976	1.071490
7	(Хліб)	(М'ясо, Ковбаса)	0.408683	0.152695	0.089820	0.219780	1.439345	0.027417	1.085983
11	(Мінеральна вода)	(Хліб)	0.188623	0.408683	0.079341	0.420635	1.029246	0.002254	1.020630
10	(Хліб)	(Мінеральна вода)	0.408683	0.188623	0.079341	0.194139	1.029246	0.002254	1.006845
22	(Сир)	(Хліб)	0.133234	0.408683	0.067365	0.505618	1.237190	0.012915	1.196074
23	(Хліб)	(Сир)	0.408683	0.133234	0.067365	0.164835	1.237190	0.012915	1.037839
24	(Сир)	(Ковбаса)	0.133234	0.383234	0.065868	0.494382	1.290028	0.014809	1.219827
13	(Сік)	(Хліб)	0.139222	0.408683	0.065868	0.473118	1.157667	0.008971	1.122296
25	(Ковбаса)	(Сир)	0.383234	0.133234	0.065868	0.171875	1.290028	0.014809	1.046661
12	(Хліб)	(Сік)	0.408683	0.139222	0.065868	0.161172	1.157667	0.008971	1.026168
18	(Яблуко)	(Хліб)	0.098802	0.408683	0.064371	0.651515	1.594184	0.023992	1.696824
36	(Кефір)	(Хліб)	0.113772	0.408683	0.064371	0.565789	1.384423	0.017874	1.361822
19	(Хліб)	(Яблуко)	0.408683	0.098802	0.064371	0.157509	1.594184	0.023992	1.069682
37	(Хліб)	(Кефір)	0.408683	0.113772	0.064371	0.157509	1.384423	0.017874	1.051914
26	(Сир)	(М'ясо)	0.133234	0.347305	0.062874	0.471910	1.358776	0.016602	1.235954
27	(М'ясо)	(Сир)	0.347305	0.133234	0.062874	0.181034	1.358776	0.016602	1.058367
38	(Кефір)	(М'ясо)	0.113772	0.347305	0.061377	0.539474	1.553312	0.021863	1.417280
39	(М'ясо)	(Кефір)	0.347305	0.113772	0.061377	0.176724	1.553312	0.021863	1.076465

Таблиця 2.2. Асоціативні правила за методом Табу-Пошук

Дана таблиця містить у собі не повний результат, оскільки нас цікавить лише перші шість рядків. Саме ці перші шість правил показуються найкращий результат.

Показники «підтримки» та «впевненості» відрізняються між собою, оскільки розмірність вхідних даних по-різному впливає на результат роботи методів. Апріорі швидше аналізує дані та формує асоціативні правила ніж Табу-Пошук, але його результат може залежати від великої кількості даних, що не є проблемою для Табу[12].

Тобто, якщо проаналізувати результати, то можна помітити, що перші шість результатів співпадають, але Табу винайшов ще шість правил на основі попередніх, чого не зміг зробити Апріорі. У цьому випадку більше ефективним є другий метод.

Асоціативні правила напряму залежать від кількості вхідних даних. Тому наступні результати були отримані за методом Апріорі на основі додаткових даних (табл. 2.3):

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction	zhang
103	(М'ясо)	(Хліб)	0.417637	0.464226	0.221298	0.529880	1.141427	0.027420	1.139654	0.212760
102	(Хліб)	(М'ясо)	0.464226	0.417637	0.221298	0.476703	1.141427	0.027420	1.112871	0.231261
67	(Ковбаса)	(Хліб)	0.432612	0.464226	0.212978	0.492308	1.060491	0.012148	1.055312	0.100532
66	(Хліб)	(Ковбаса)	0.464226	0.432612	0.212978	0.458781	1.060491	0.012148	1.048352	0.106464
41	(М'ясо)	(Ковбаса)	0.417637	0.432612	0.199667	0.478088	1.105118	0.018992	1.087132	0.163333
40	(Ковбаса)	(М'ясо)	0.432612	0.417637	0.199667	0.461538	1.105118	0.018992	1.081531	0.167644
123	(Мінеральна вода)	(Хліб)	0.176373	0.464226	0.093178	0.528302	1.138027	0.011301	1.135840	0.147258
137	(Сир)	(Хліб)	0.186356	0.464226	0.093178	0.500000	1.077061	0.006667	1.071547	0.087935
122	(Хліб)	(Мінеральна вода)	0.464226	0.176373	0.093178	0.200717	1.138027	0.011301	1.030457	0.226375
136	(Хліб)	(Сир)	0.464226	0.186356	0.093178	0.200717	1.077061	0.006667	1.017967	0.133540
60	(Сир)	(Ковбаса)	0.186356	0.432612	0.091514	0.491071	1.135130	0.010894	1.114867	0.146310
61	(Ковбаса)	(Сир)	0.432612	0.186356	0.091514	0.211538	1.135130	0.010894	1.031939	0.209811
96	(Сир)	(М'ясо)	0.186356	0.417637	0.089850	0.482143	1.154454	0.012021	1.124563	0.164432
97	(М'ясо)	(Сир)	0.417637	0.186356	0.089850	0.215139	1.154454	0.012021	1.036673	0.229735
141	(Сік)	(Хліб)	0.166389	0.464226	0.084859	0.510000	1.098602	0.007616	1.093416	0.107667
140	(Хліб)	(Сік)	0.464226	0.166389	0.084859	0.182796	1.098602	0.007616	1.020076	0.167519
149	(Яблуко)	(Хліб)	0.124792	0.464226	0.083195	0.666667	1.436081	0.025263	1.607321	0.346958
148	(Хліб)	(Яблуко)	0.464226	0.124792	0.083195	0.179211	1.436081	0.025263	1.066301	0.566770
94	(Риба)	(М'ясо)	0.131448	0.417637	0.074875	0.569620	1.363911	0.019978	1.353137	0.307195
95	(М'ясо)	(Риба)	0.417637	0.131448	0.074875	0.179283	1.363911	0.019978	1.058285	0.458159
110	(Яйця курячі)	(М'ясо)	0.116473	0.417637	0.073211	0.628571	1.505065	0.024568	1.567900	0.379815
134	(Риба)	(Хліб)	0.131448	0.464226	0.073211	0.556962	1.199764	0.012190	1.209318	0.191701
111	(М'ясо)	(Яйця курячі)	0.417637	0.116473	0.073211	0.175299	1.505065	0.024568	1.071330	0.576234
135	(Хліб)	(Риба)	0.464226	0.131448	0.073211	0.157706	1.199764	0.012190	1.031175	0.310771
151	(Яйця курячі)	(Хліб)	0.116473	0.464226	0.071547	0.614286	1.323246	0.017478	1.389043	0.276486
150	(Хліб)	(Яйця курячі)	0.464226	0.116473	0.071547	0.154122	1.323246	0.017478	1.044509	0.455944
32	(Кефір)	(М'ясо)	0.129784	0.417637	0.069884	0.538462	1.289304	0.015681	1.261786	0.257853

Таблиця 2.3. Асоціативні правила за методом Апріорі

У даному випадку було отримано 152 правила, але, як й попередньому випадку, лише шість перший правил показали найкращі результати, порівняно з іншими. Однак слід зазначити, що показники «support» та «confidence»

збільшились, що свідчить про те, що вони мають найвищу ймовірність того, що дані правила спрацюють у повсякденному житті.

Результати на основі методу Табу-пошук (табл. 2.4):

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction
0	(М'ясо)	(Хліб)	0.363510	0.409471	0.185237	0.509579	1.244481	0.036390	1.204126
1	(Хліб)	(М'ясо)	0.409471	0.363510	0.185237	0.452381	1.244481	0.036390	1.162287
26	(Ковбаса)	(Хліб)	0.387187	0.409471	0.178273	0.460432	1.124456	0.019731	1.094448
27	(Хліб)	(Ковбаса)	0.409471	0.387187	0.178273	0.435374	1.124456	0.019731	1.085344
2	(М'ясо)	(Ковбаса)	0.363510	0.387187	0.167131	0.459770	1.187464	0.026385	1.134357
3	(Ковбаса)	(М'ясо)	0.387187	0.363510	0.167131	0.431655	1.187464	0.026385	1.119901
4	(М'ясо, Ковбаса)	(Хліб)	0.167131	0.409471	0.097493	0.583333	1.424603	0.029058	1.417270
6	(Ковбаса, Хліб)	(М'ясо)	0.178273	0.363510	0.097493	0.546875	1.504430	0.032689	1.404668
5	(М'ясо, Хліб)	(Ковбаса)	0.185237	0.387187	0.097493	0.526316	1.359334	0.025772	1.293717
7	(М'ясо)	(Ковбаса, Хліб)	0.363510	0.178273	0.097493	0.268199	1.504430	0.032689	1.122884
8	(Ковбаса)	(М'ясо, Хліб)	0.387187	0.185237	0.097493	0.251799	1.359334	0.025772	1.088962
9	(Хліб)	(М'ясо, Ковбаса)	0.409471	0.167131	0.097493	0.238095	1.424603	0.029058	1.093141
32	(Сир)	(Хліб)	0.157382	0.409471	0.077994	0.495575	1.210282	0.013551	1.170698
16	(Мінеральна вода)	(Хліб)	0.182451	0.409471	0.077994	0.427481	1.043984	0.003286	1.031458
17	(Хліб)	(Мінеральна вода)	0.409471	0.182451	0.077994	0.190476	1.043984	0.003286	1.009913
33	(Хліб)	(Сир)	0.409471	0.157382	0.077994	0.190476	1.210282	0.013551	1.040882
34	(Сир)	(Ковбаса)	0.157382	0.387187	0.076602	0.486726	1.257083	0.015666	1.193929
35	(Ковбаса)	(Сир)	0.387187	0.157382	0.076602	0.197842	1.257083	0.015666	1.050439
36	(Сир)	(М'ясо)	0.157382	0.363510	0.075209	0.477876	1.314617	0.017999	1.219041
37	(М'ясо)	(Сир)	0.363510	0.157382	0.075209	0.206897	1.314617	0.017999	1.062432
18	(Сік)	(Хліб)	0.144847	0.409471	0.071031	0.490385	1.197606	0.011720	1.158774
19	(Хліб)	(Сік)	0.409471	0.144847	0.071031	0.173469	1.197606	0.011720	1.034630
29	(Яблуко)	(Хліб)	0.104457	0.409471	0.069638	0.666667	1.628118	0.026866	1.771588
28	(Хліб)	(Яблуко)	0.409471	0.104457	0.069638	0.170068	1.628118	0.026866	1.079056
11	(Риба)	(М'ясо)	0.114206	0.363510	0.062674	0.548780	1.509672	0.021159	1.410600
10	(М'ясо)	(Риба)	0.363510	0.114206	0.062674	0.172414	1.509672	0.021159	1.070334
39	(Яйця курячі)	(М'ясо)	0.097493	0.363510	0.061281	0.628571	1.729174	0.025842	1.713628
12	(Риба)	(Хліб)	0.114206	0.409471	0.061281	0.536585	1.310436	0.014517	1.274300
38	(М'ясо)	(Яйця курячі)	0.363510	0.097493	0.061281	0.168582	1.729174	0.025842	1.085504
13	(Хліб)	(Риба)	0.409471	0.114206	0.061281	0.149660	1.310436	0.014517	1.041694
40	(Яйця курячі)	(Хліб)	0.097493	0.409471	0.059889	0.614286	1.500194	0.019968	1.531002
41	(Хліб)	(Яйця курячі)	0.409471	0.097493	0.059889	0.146259	1.500194	0.019968	1.057120
43	(Кефір)	(М'ясо)	0.110028	0.363510	0.058496	0.531646	1.462535	0.018500	1.358993
44	(Кефір)	(Хліб)	0.110028	0.409471	0.058496	0.531646	1.298373	0.013443	1.260860
42	(М'ясо)	(Кефір)	0.363510	0.110028	0.058496	0.160920	1.462535	0.018500	1.060652
45	(Хліб)	(Кефір)	0.409471	0.110028	0.058496	0.142857	1.298373	0.013443	1.038301
22	(Напівфабрикати)	(Хліб)	0.121170	0.409471	0.055710	0.459770	1.122840	0.006095	1.093107
23	(Хліб)	(Напівфабрикати)	0.409471	0.121170	0.055710	0.136054	1.122840	0.006095	1.017229
46	(Чіпси)	(Ковбаса)	0.094708	0.387187	0.054318	0.573529	1.481274	0.017648	1.436942
21	(Сік)	(М'ясо)	0.144847	0.363510	0.054318	0.375000	1.031609	0.001664	1.018384
20	(М'ясо)	(Сік)	0.363510	0.144847	0.054318	0.149425	1.031609	0.001664	1.005383
47	(Ковбаса)	(Чіпси)	0.387187	0.094708	0.054318	0.140288	1.481274	0.017648	1.053018

Таблиця 2.4. Асоціативні правила за методом Табу-Пошук

У даному випадку було отримано 48 правил але, як й попередньому випадку, лише дванадцять перший правил показали найкращі результати, порівняно з іншими. Однак слід зазначити, що показники «support» та «confidence» збільшились, що свідчить про те, що вони мають найвищу ймовірність того, що дані правила спрацюють у повсякденному житті [10].

При додаткових даних також змінилися інші результати так показники, які можна проаналізувати за допомогою графіків.

За новими вхідними даними маємо у середньому 5 унікальних товарі в одному чеку, а також збільшилась кількість унікальних чеків, як й загальна кількість рядків без дублікатів (рис. 2.15):

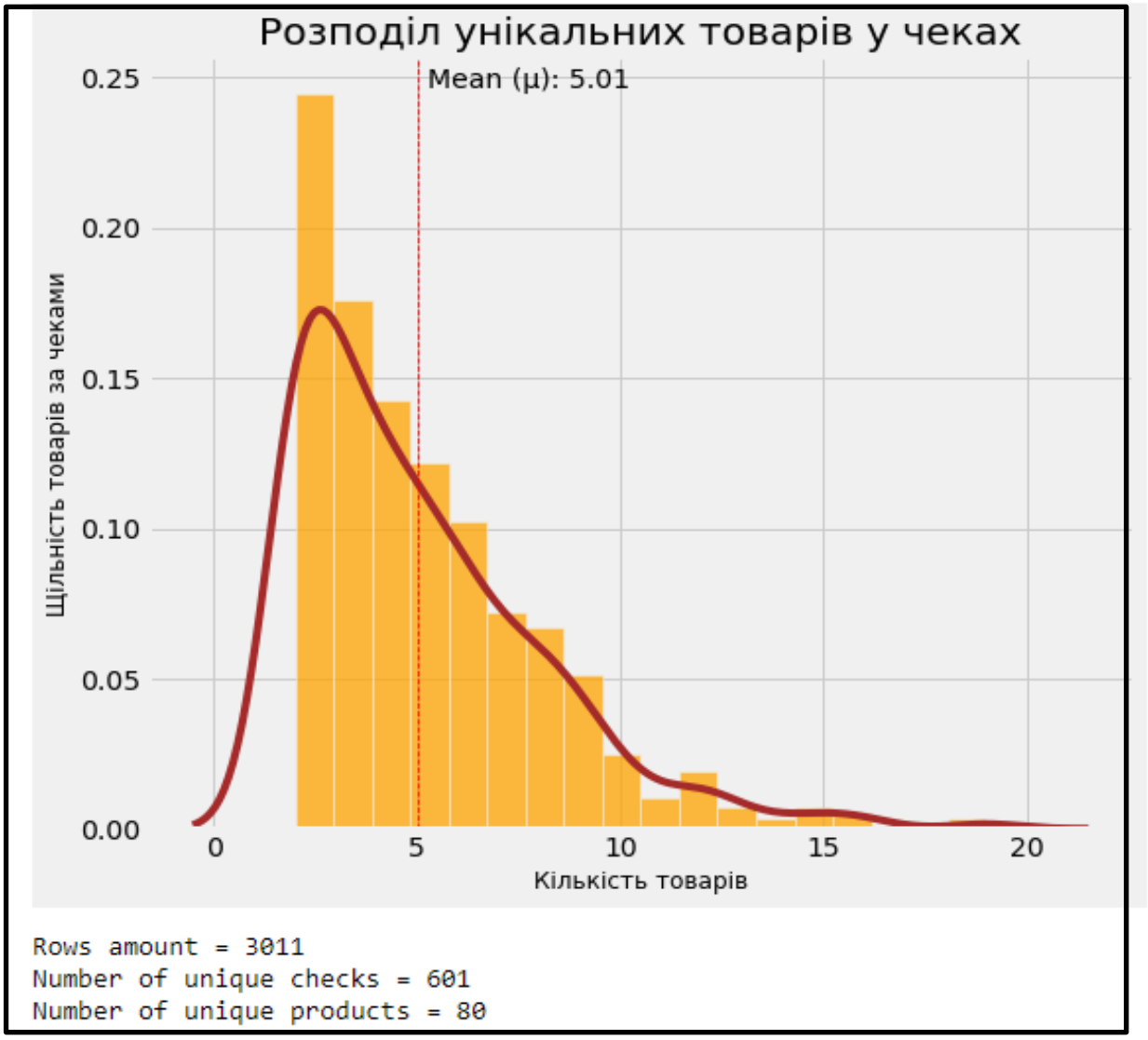


Рис. 2.15. Графік розподілу унікальних товарів у чеках

Додаткові дані дають можливість провести більш детальний аналіз для виявлення більш ймовірних асоціативних правил з достатнім рівнем показників «support» та «confidence». Проаналізувавши останній та перший графік розподілу унікальних товарів, можна побачити, що додаткові 52 чека, або 422 унікальних

рядків покращили результати, що є показником того, наскільки змінюються асоціативні правила при збільшенні вхідних даних.

Рис 2.16. Матриця «підтримки» товарів

Для більше детального розуміння слід проаналізувати нову матрицю «підтримки» товарів, яка надає можливість візуально оцінити сформовані асоціативні правила.

Однак, через невелику кількість додаткових даних, суттєво змінилися показники лише шести асоціативних правил. Дані зміни виділені на рис(2.16) та рис(2.17).

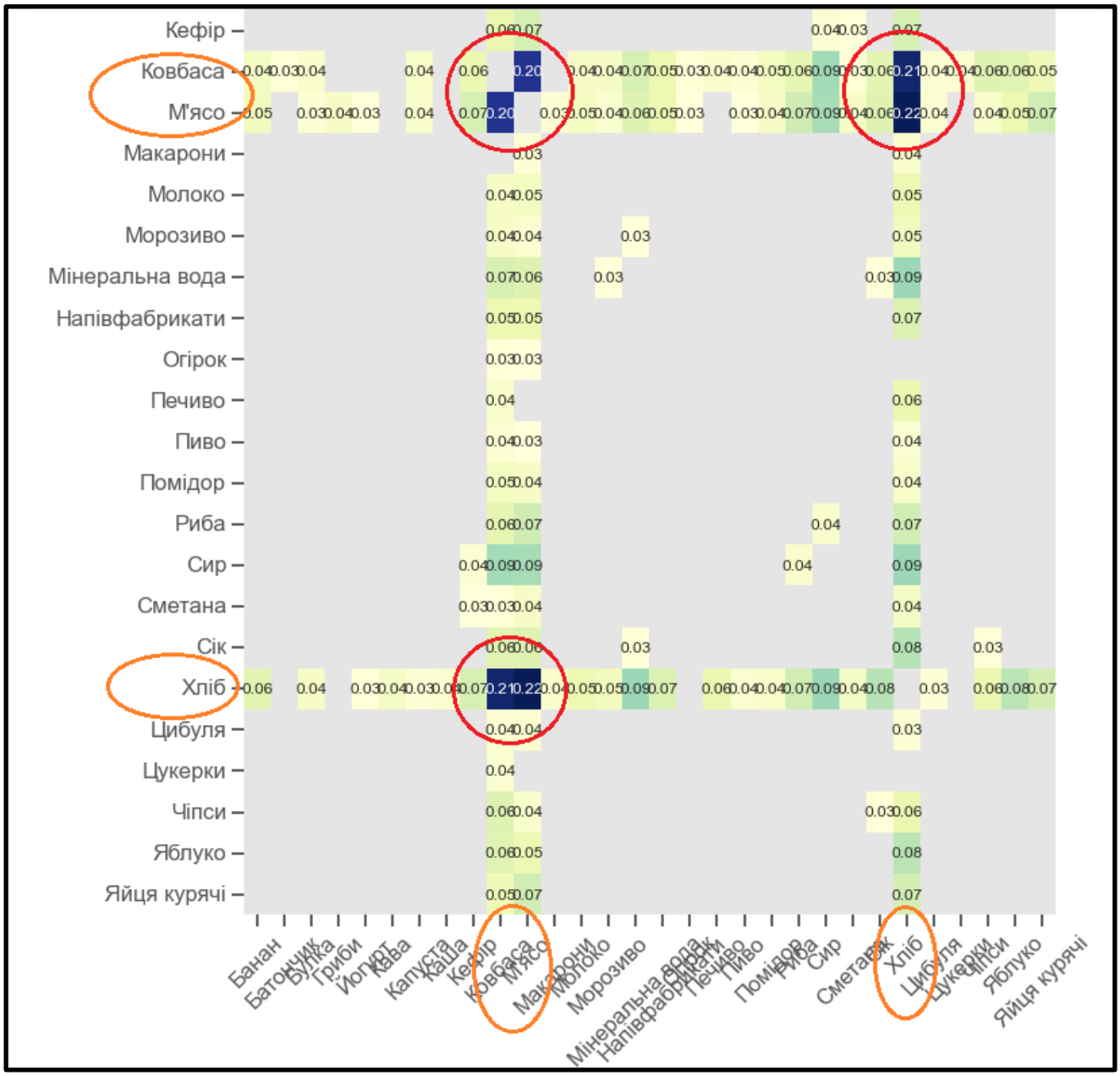


Рис. 2.17. Матриця «підтримки» товарів

Також слід проаналізувати матрицю «підтримки» товарів й за методом Табу-пошук на основі початкових даних та додаткових, що зображена на рис. 2.18 та рис. 2.19:

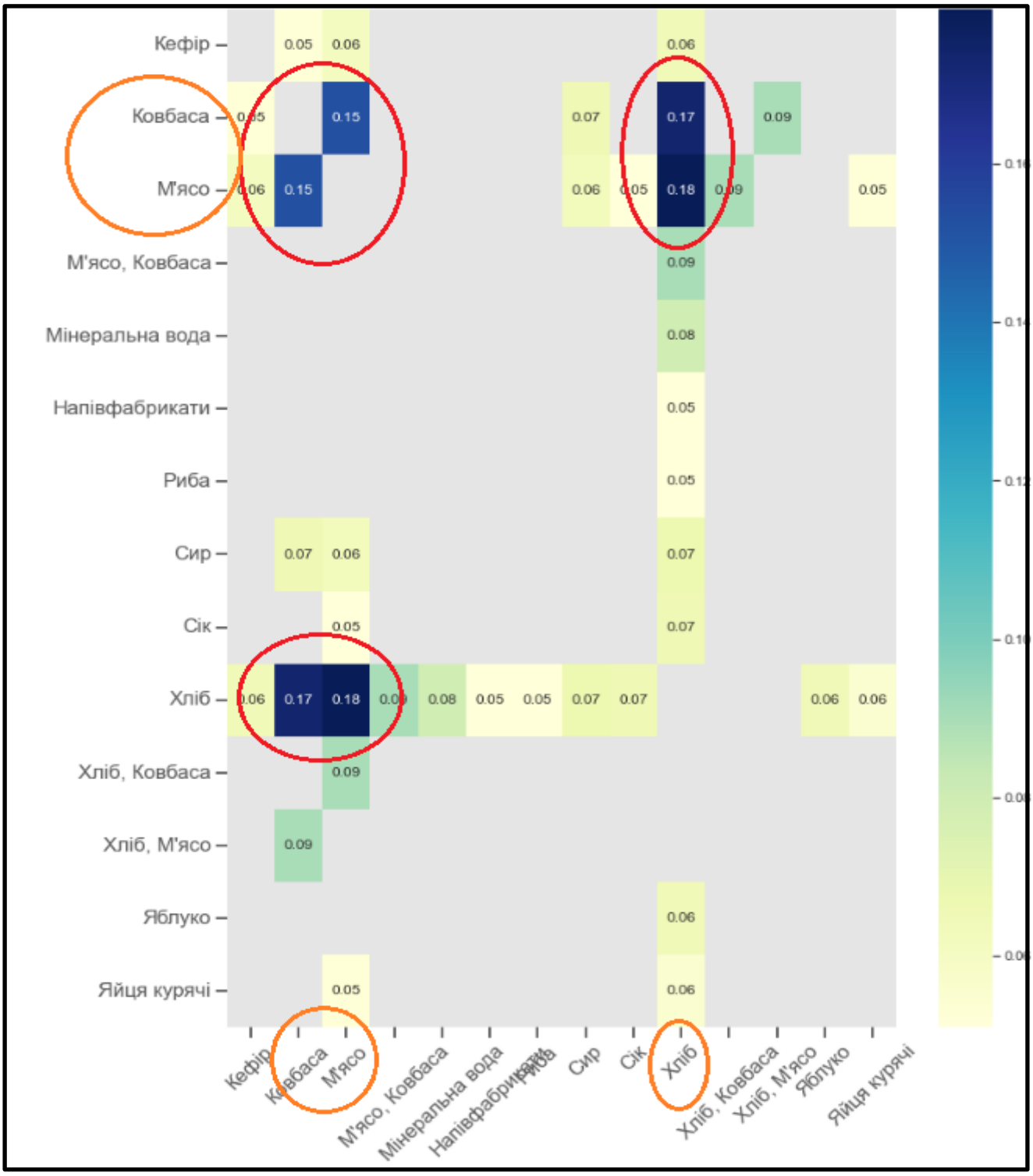


Рис. 2.18. Матриця «підтримки» товарів

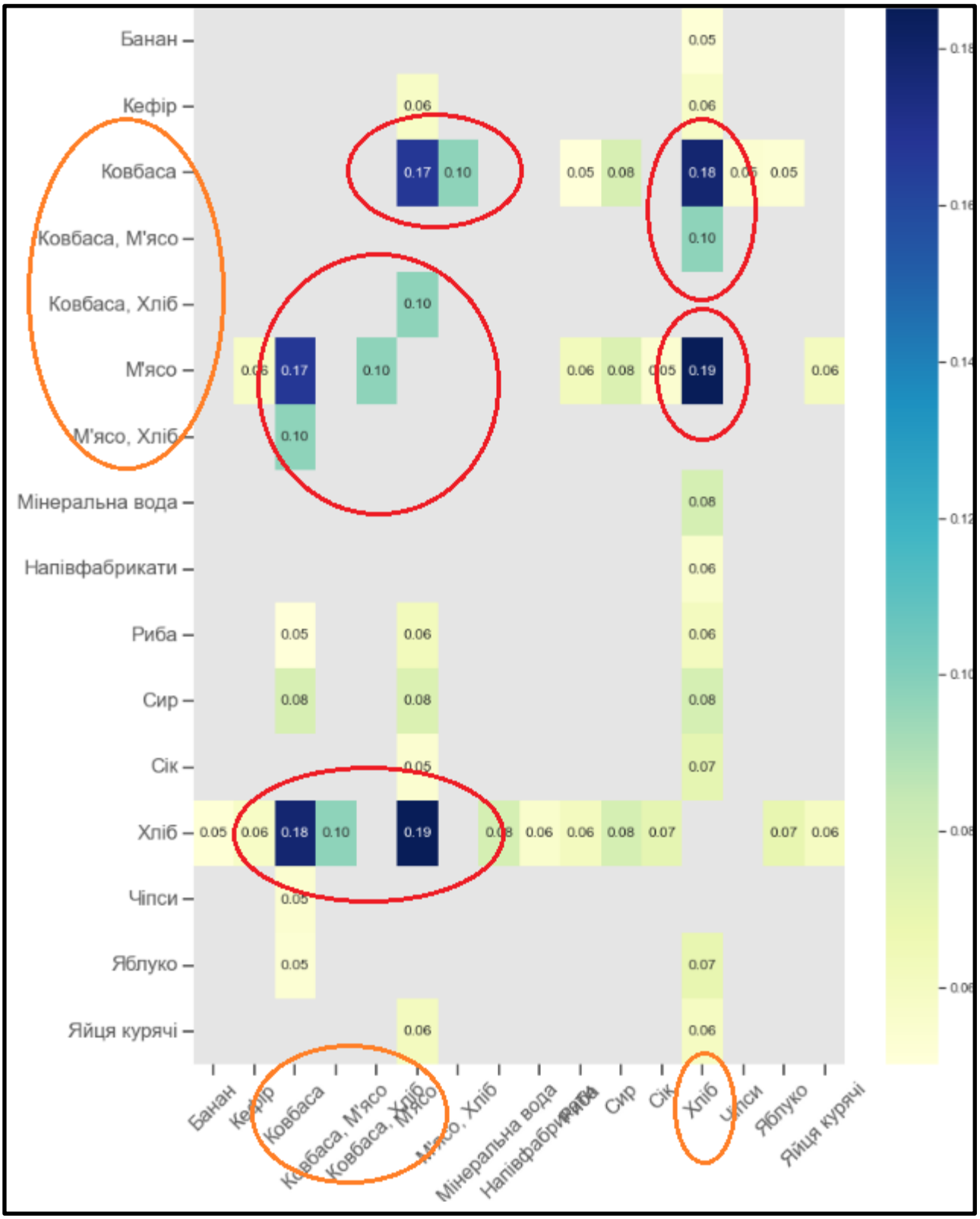


Рис. 2.19. Матриця «підтримки» товарів

Перш за все слід зазначити, що вплив додаткових даних відображається краще при аналізі даних за допомогою методу Табу-пошук, оскільки саме цей алгоритм дає можливість формувати асоціативні правила при великій кількості вхідних даних.

При аналізі нових вхідних даних за допомогою методу Апріорі та Табу-пошук були виявлені нові правила, а також збільшився параметр «Support» на 0.1 на основних шість правил асоціацій, однак алгоритм Табу надав кращі результати за рахунок того, що виявив більше правил та зв’язків між товарами рис (2.19).

2.6 Висновки

Алгоритм виявлення асоціативних правил був розроблений на мові Python, на основі методу Апріорі та Табу-пошук. Основною складовою написаної програми є правильний підбір бібліотек, які надають можливість провести оптимальний аналіз даних за двома методами. Програма базується на використанні наступних бібліотек:

numpy (або скорочено *np*) – це бібліотека для чисельних обчислень на Python

pandas (або скорочено *pd*) – це бібліотека для обробки та аналізу даних у Python

TransactionEncoder від *mlxtend.preprocessing* – це клас для перетворення даних транзакції (наприклад, список списків, де кожен внутрішній список представляє транзакцію та містить придбані товари)

apriori та *association_rules* з *mlxtend.frequent_patterns* – це функції для генерації частих наборів елементів і правил асоціації з даних транзакцій

mlxtend — це пакет, який надає інструменти для машинного навчання, інтелектуального аналізу даних і дослідницького аналізу даних.

fpgrowth — це алгоритм для частого видобутку набору елементів і швидший за Apriori.

Перед початком аналізу даних було проведено очищення та підготовка вхідних даних. Отже, були видалені стовпці, які не несли у собі важливої інформації, але могли негативно вплинути та результат. Наступним етапом було переформування загального виду даних, тобто вся інформація по товару занесена в єдину клітинку, а також була побудована ієрархія товарів.

Після цього було проведення імпорту даних та видалення дублікатів, оскільки це обов’язкова умова для побудови асоціативних правил. Аналіз даних за допомогою методу Апріорі було розпочато з первинного аналізу, результатом чого став графік «Розподілу унікальних товарів у чеках» рис(2.4) та рис(2.14), що відображає середню кількість товарів у чеку, загальний об’єм даних, загальну кількість унікальних товарів та чеків.

Далі, на основі попередньої інформації, була побудована двійкова таблиця табл() на основі якої були побудовані правила асоціації методом Апріорі. Наступна робота програму була зосереджена на візуальному відображенні отриманих результатів, а також на його більш детальному описі. Другий етап базується на аналізу даних за допомогою методу Табу-пошук.

Метод Апріорі виявив 144 правила, але слід зазначити, що лише шість перших правил показали найкращий результат, який залежить від показників «support» та «confidence». Наступні правила мають показник «підтримки» нижче ніж 0.1. У даному випадку це свідчить про те, що дані правила мають меншу ймовірність того, що спрацюють у реальному житті, однак вони також вважаються ефективними.

На основі отриманого результату було виявлено, що більшість людей купують наступні набори товарів (рис 2.20):

antecedents	consequents
(М'ясо)	(Хліб)
(Хліб)	(М'ясо)
(Ковбаса)	(Хліб)
(Хліб)	(Ковбаса)
(М'ясо)	(Ковбаса)
(Ковбаса)	(М'ясо)

Рис. 2.20. Основні набори товарів за методом Апріорі

Тобто більшість людей, купуючи, наприклад, м’ясо або ковбасу також придбають й хліб з ймовірністю 22%. Дане правило також показує те, що може людина, навпаки, придбає хліб та захоче придбати м’ясо або ковбасу. Отже, на основі цього можна стверджувати, що це три основні товари, що постійно обирали клієнти супермаркету «Варус». Якщо проаналізувати наступні правила, то можна прийти до висновку, що також постійно обирали наступні набори продуктів (рис. 2.21):

(Сир)	(Хліб)
(Хліб)	(Сир)
(Сир)	(Ковбаса)
(Ковбаса)	(Сир)
(Сир)	(М'ясо)
(М'ясо)	(Сир)
(Кефір)	(М'ясо)
(М'ясо)	(Кефір)
(Риба)	(М'ясо)

Рис. 2.21. Додаткові набори товарів за методом Апріорі

Метод Табу-пошук виявив 40 правил але слід зазначити, що лише шість перших правил показали найкращий результат, який залежить від показників «support» та «confidence». Наступні правила мають показник «підтримки» нижче ніж 0.1. У даному випадку це свідчить про те, що дані правила мають меншу ймовірність того, що спрацюють у реальному житті, однак вони також вважаються ефективними.

На основі отриманого результату було виявлено результати співпадають з асоціативними правилами на основі Апріорі методу, але показники «support» та «confidence» відрізняються від попередніх результатів, що свідчить про факт, що Табу-пошук більш ефективний при обробці великої кількості даних, ніж Апріорі. Прикладом цього також є те, що на основі попередніх правил були також виявлені набори, які свідчать, що з ймовірністю 9% люди, які придбали, наприклад, м'ясо та ковбасу також придбають й хліб. Однак, не зважаючи на даний результат, наступні правила не відрізняються від результату Апріорі методу.

Однак, слід визначити, що для більш детального аналізу слід збільшити кількість вхідних даних. Приклад такого впливу є результати, що були отримані після внесення додаткових даних, які не тільки покращили загальні показники асоціативних правил, але також надали змогу сформувати нові результати, як за методом Табу-пошук, так й за методом Апріорі.

ВИСНОВКИ

У даній кваліфікаційній роботі була розглянута задача провести оптимізацію загального асортименту товару, який виставляються на полиці, а також розробити новий план того, як й де повинні бути розташовані продукти різних типів між собою. Основою цієї задачі являються дані від мережі супермаркетів «Варус»

Для вирішення даної задачі було задіяно аналіз ринкового кошика (Market Basket Analysis), який надає змогу провести аналіз споживацької поведінки, що досліджує зв'язки та залежності між продуктами, які споживачі купують разом. В основі цього методу лежить ідея про те, що споживачі, як правило, вибирають не окремі товари, а набір товарів, які вони споживають разом.

Принцип роботи алгоритму полягає у використанні двох методі, дослідження вхідних даних, що у результаті надають змогу зрозуміти які товари обирають споживачі, а також отримати інформацію про зв’язки серед наборів різних товарів, тобто може допомогти виявити такі залежності, як "якщо клієнт купує товар A, то він часто купує товар B", або "якщо клієнт купує товар C, то він ймовірно купить товар D". Такі залежності називаються асоціативними правилами.

Основа даної роботи полягає у пошуку асоціативних за допомогою методу Апріорі та Табу-пошук. Метод Apriori заснований на ідеї, що якщо деякий набір предметів є частим у наборі транзакцій, то будь-який його піднабір також буде частим. Метод Tabu Search - це евристичний метод оптимізації, він заснований на принципі пошуку у просторі станів, де кожен стан є набір асоціативних правил.

Для оцінки результатів, а також виявлення більш ефективного методу аналізу ринкового кошику були також розраховані необхідні показники: підтримка (Support), достовірність (Confidence), леверидж (Leverage), ліфтінг

(Lift) та переконання (Conviction). Однак слід зазначити, що основні порівняння були зроблені за допомогою лише перших двох критеріїв.

Підтримка (support) відображає частоту, з якою конкретний набір товарів зустрічається в загальному обсязі продажів. Визначається як відношення кількості транзакцій, в яких зустрічається даний набір товарів, до загальної кількості транзакцій.

Впевненість (confidence) відображає ймовірність, з якою товари, що належать до одного набору, будуть куплені разом. Визначається як відношення кількості транзакцій, в яких купуються якісь конкретні товари разом, до кількості транзакцій, в яких купуються хоча б один з цих товарів.

Аналіз даних та формування асоціативних правил були проведені за допомогою розробленої програми, що являє собою автоматизацію у визначенні зв’язків та залежності між товарами за допомогою методу Апріорі та Табу-пошук. Основний алгоритм роботи коду базується на використання основних бібліотек, які відповідають за обробку, аналіз даних, а також за реалізацію необхідних методів.

Спочатку, результати за методом Апріорі та Табу-пошук надали 144 та 40 правил відповідно. При сортуванні визначених правил відносно показників «підтримки» та «впевненості» за спаданням було визначено шість основних правил, що вказували на те, що такі товари як: ковбаса, хліб та м’ясо мають найвищу ступінь залежності між собою порівняно з іншими товарами. На наступному етапі було прийнято рішення, що необхідно збільшити загальну кількість даних.

Результатом такого рішення стало збільшення визначених правил за методом Апріорі та Табу-пошук у кількості до 152 та 48 правил відповідно.

На основі отриманої інформації можна прийти до висновку, що протягом місяця клієнти супермаркету «Варус» купували м’ясо, ковбасу та хліб, тобто у

разі, якщо людина придбала, наприклад, м'ясо то з ймовірністю 22 – 23% придбає також хліб або ковбасу, а також споживач, який придбав, наприклад, ковбасу та хліб з ймовірністю 17 – 18% придбає м’ясо й навпаки.

Також слід зазначити, що результати аналізу виявили залежність між хлібом та сиром, соком, яблуком та рибою, що вказує на те, що коли клієнт купує, наприклад, хліб то з ймовірністю 8 – 9% придбає й сир або рибу, тощо. Існує також залежність з ймовірністю 3 – 4% між такими товарами, як м’ясо та гриби, сік та чіпси, сир та риба, тощо.

Дані результати були отримані на основі даних, що являються вершиною ієрархічного розподілу товарів (рис 2.2). Тобто отримані показники ймовірності теоретично розповсюджуються на будь-який товар, наприклад, людина, що придбала «Ковбасу САЛЯМІ ПРАЗЬКА» з ймовірністю 22 – 23% придбає «Хліб нарізка Катеринославський 250г» або, наприклад, придбає «Чіпси сметана і зелень Лейс 140г» та «Сік Яблучний Біола 0,5л», тощо. Для отримання точних результатів на практиці слід провести пошук асоціативних правил на основних вхідних даних.

На основі отриманих даних слід провести оптимізацію щодо розташування товарів по супермаркету, а також можна розробити нову рекламу, яка буде закликати до купівлі товарів, які показали нижчі показники, оскільки це не тільки зацікавить клієнтів, але й підвищить дохід та надасть можливість отримати нову інформацію для аналізу.

Дані результати були отримані відштовхуючись від того, що обидва методи виявили середню кількість унікальних товарів в одному чеку у розмірі лише 5 штук, а загальна кількість унікальних товарів становить 80 одиниць. Метод Апріорі виявив зв’язок між 30 унікальними товарами, а Табу-пошук лише між 16. Відштовхуючись від цього, можна прийти до висновку, що необхідно збільшувати загальну кількість даних, обирати більший часовий інтервал, а також уважно обирати метод для аналізу даних.

Перш за все, для задоволення даних вимог необхідно проводити аналіз на спеціально виділеному сервері, який забезпечить необхідною кількістю ресурсів для великої кількості даних. Також слід правильно обирати метод аналізу через те, що, наприклад, Апріорі має можливість визначити асоціативні правила, але дуже залежить від кількості даних, що впливає на швидкість та ефективність отриманих результатів. Але у випадку з методом Табу-пошук, все навпаки, тобто він має перевагу в обробці великої кількості даних. В даній роботі загальний об’єм вхідних даних становить 4515 рядків, що у свою чергу являється оптимальною кількість для обох методів, оскільки отримані асоціативні правила не суперечать один одному.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Researchgate [Електронний ресурс]. – Режим доступу https://www.researchgate.net/publication/242360346_Mining_Associations_Between_Sets_of_Items_in_Massive_Databases
2. Vldb [Електронний ресурс]. – Режим доступу https://www.vldb.org/conf/1994/P487.PDF
3. Wikipedia [Електронний ресурс]. – Режим доступу https://en.wikipedia.org/wiki/Tabu_search
4. Researchgate [Електронний ресурс]. – Режим доступу https://www.researchgate.net/publication/343484851_Market_Basket_Analysis_Recommendation_System_Using_Association_Rules
5. Kaggle [Електронний ресурс]. – Режим доступу https://www.kaggle.com/code/xvivancos/market-basket-analysis
6. Конспект лекцій з дисципліни «Інтелектуальний аналіз даних» [Текст]: конспект, 2019. – 86 с.
7. Datacamp [Електронний ресурс]. – Режим доступу https://www.datacamp.com/tutorial/association-rule-mining-python
8. KavithaVenkatachari and IssacDavanbuChandrasekaran, “MARKET BASKET ANALYSIS USING FP GROWTH AND APRIORI ALGORITHM: A CASE STUDY OF MUMBAI RETAIL STORE”, BVIMSR’s Journal of Management Research Vol. 8 Issue- 1: April 2016.
9. Loraine Charlet Annie M.C. and Ashok Kumar D,“MARKET BASKET ANALYSIS FOR A SUPERMARKET BASED ON FREQUENT ITEM SET MINING”, IJCSI International Journal of Computer Science Issues, Vol. 9, Issue 5, No 3, September 2012.
10. Anurag Sinha, "Implying Association Rule Mining and Market Basket Analysis for Knowing Consumer Behavior and Buying Pattern in Lockdown - A Data Mining Approach Department of information technology, Amity University Jharkhand, May 2021.

11. Theautomatic [Електронний ресурс]. – Режим доступу https://theautomatic.net/2019/08/21/python-basket-analysis-and-pymining/
12. A. Bertoni and T. Larsson, “ScienceDirect Data Mining in Product Service Systems Design: Literature Review and Research Questions,” Procedia CIRP, vol. 64, pp. 306–311, 2017.

ДОДАТКИ

ДОДАТОК А

Відомість матеріалів кваліфікаційної роботи

№ з/п	Позначення	Назва	Кількість	Примітки
1				
2		Документація		
3				
4	САУ.КР. 23.12.ПЗ	Пояснювальна записка	77	Формат А4
5				
6	САУ.КР. 23.12.ДМ	Демонстраційні матеріали	18	Презентація на CD-R
7				
8	САУ.КР. 23.12.КР	Копія роботи	1	Диск CD-R
9				
10				
11				
12				
13				
14				
15				
16				
17				
18				

					САУ.КР.23.12.ДА.ПЗ.
Змін	Аркуш	№ докум.	Підпис	Дата	
Розроб.		Рижченко			**Матеріали кваліфікаційної роботи**
К. розд.		Купенко			
Керівн.		Купенко			
Н.контр.		Хом’як			
Зав. каф.		Желдак			

Літ.	Аркуш	Аркушів

НТУ «ДП»,
12; 124-19-2

ДОДАТОК Б

Відгук

на кваліфікаційної роботу бакалавра за спеціальністю 124 Системний аналіз

студента групи 124-19-2

Рижченко Дениса Віталійовича

Тема кваліфікаційної роботи: «Пошук правил асоціацій та аналіз ринкового кошика для знання поведінки споживачів і моделі купівлі»

Обсяг кваліфікаційної роботи 77 стор.

Мета кваліфікаційної роботи: на основі даних, що представляють собою набір купівельних чеків мережі супермаркетів, отримати значущі асоціативні правила щодо наборів продуктів, які часто купуються разом.

Актуальність теми: виявлені значущі асоціативні правила у поведінці покупців слугують базисом для розробки аналітичних, організаційних та маркетингових рішень направлених на збільшення продажів супермаркету.

Тема кваліфікаційної роботи безпосередньо пов'язана з об'єктом діяльності бакалавра спеціальності 124 Системний аналіз, оскільки системний підхід до підготовки даних та виявлення асоціацій і патернів у даних є одним з найважливіших аспектів роботи інформаційних, рекомендаційних та аналітичних систем.

Виконані в дипломній роботі завдання відповідають вимогам освітньо-кваліфікаційного рівня бакалавра. Оригінальність наукових рішень полягає в системному погляді на область проблем, які пов’язані з обробкою даних для задачі виявлення асоціативних правил, у порівнянні та поєднанні різних підходів до пошуку вірогідних патернів купівельної поведінки.

Практичне значення результатів кваліфікаційної роботи полягає у підготовці програмного забезпечення для виявлення вірогідних асоціативних правил в залежності від розміру бази вхідних даних.

Висновки підтверджують можливість використання результатів роботи в аналітичних центрах магазинів та супермаркетів, інтернет магазинів та торгівельних мереж.

Оформлення пояснювальної записки та демонстраційного матеріалу до неї виконано згідно з вимогами. Роботу виконано самостійно, відповідно до завдання та у повному обсязі.

У роботі відзначено такі недоліки: не досліджено в повній мірі можливості пошуку ієрархічних асоціативних правил для виявлення більш тонких закономірностей купівельної поведінки.

Кваліфікаційна робота в цілому заслуговує оцінки: 95 – «відмінно».

З урахуванням висловлених зауважень автор заслуговує присвоєння кваліфікації «бакалавр з системного аналізу».

Керівник кваліфікаційної роботи бакалавра
Д. ф.-м. н., професор,
професор кафедри системного аналізу та управління _____/ Купенко О.П.

ДОДАТОК В

У даному додатку представлені лістинги програм, що використовувались у кваліфікаційній роботі.

В.1 Агоритм Апріорі

```
import numpy as np
import pandas as pd
from itertools import permutations
from mlxtend.preprocessing import TransactionEncoder
from mlxtend.frequent_patterns import apriori, association_rules
import matplotlib.pyplot as plt
import seaborn as sns
from pandas.plotting import parallel_coordinates
import networkx as nx
df = pd.read_csv('C:/Users/User/Desktop/GoodsStore307.csv')
df = df.drop(['key'],axis = 1) df = df.drop_duplicates()
df.head()
meandf = df.groupby(by=["check_id"],as_index=False).count()
meandf=meandf[meandf['product_id']>1]
print(meandf.info())
plt.style.use('fivethirtyeight')
plt.figure(figsize=(8, 7))
for column in meandf.columns[2:]:
   plt.title("Розподіл унікальних товарів у чеках")
   sns.histplot(meandf[column], stat='density', color='orange')
   sns.kdeplot(meandf[column], color='brown')
   plt.axvline(meandf[column].mean(), color='red', linestyle='--', linewidth=0.8)
   min_ylim, max_ylim = plt.ylim()
   plt.text(meandf[column].mean()*1.05, max_ylim*0.96, 'Mean (μ):
{:.2f}'.format(meandf[column].mean()))
   plt.xlabel("Кількість товарів", fontsize=12); plt.ylabel("Щільність товарів за чеками", fontsize=12)
plt.show()
for check in df['check_id']:
   sum = 0
   for unic in meandf['check_id']:
      if check != unic:
```

```
            sum += 1
    if sum==meandf['check_id'].shape[0]:
        df = df[df['check_id']!=check]
print('Rows amount = {0}'.format(df.shape[0]))
print('Number of unique checks = {0}'.format(df.groupby(["check_id"]).count().shape[0]))
print('Number of unique products = {0}'.format(df.groupby(["product_id"]).count().shape[0]))
uid_sales =
df.pivot_table(values='product_id',columns=['product_name'],index='check_id')#,aggfunc=np.sum)
uid_sales[np.isnan(uid_sales)] = 0
uid_sales[uid_sales>0] = 1
uid_sales.head()
freq_items = apriori(uid_sales, min_support=0.03, use_colnames=True, max_len=2)
freq_items.head()
rules = association_rules(freq_items, metric = "confidence", min_threshold = 0)
rules.sort_values(by = ["support","confidence"], ascending = False)
sns.set_context("talk")
sns.relplot(x='antecedent support', y='consequent support', data=rules,
            size='lift', hue='confidence', height=6, aspect=2)
plt.title("Antecedent Support v.s. Consequent Support", fontsize=16, y=1.02)
plt.xlabel('Antecedent Support', fontsize=12)
plt.ylabel('Consequent Support', fontsize=12)
plt.show()
# add a column for a Zhang's core
def zhangs_rule(rules):
    rule_support = rules['support'].copy()
    rule_ante = rules['antecedent support'].copy()
    rule_conseq = rules['consequent support'].copy()
    num = rule_support - (rule_ante * rule_conseq)
    denom = np.max((rule_support * (1 - rule_ante).values,
                    rule_ante * (rule_conseq - rule_support).values), axis = 0)
    return num / denom
rules_zhangs_list = zhangs_rule(rules)
rules = rules.assign(zhang = rules_zhangs_list)
rules.sort_values(by = ["support","confidence"], ascending = False)
rules['antecedents'] = rules['antecedents'].apply(lambda a: ', '.join(list(a)))
rules['consequents'] = rules['consequents'].apply(lambda a: ', '.join(list(a)))
pivot_support = rules.pivot(index='antecedents', columns='consequents', values='support')
sns.set_context("talk")
plt.style.use('ggplot')
plt.subplots(figsize=(12, 16))
sns.set()
ax = sns.heatmap(data=pivot_support, annot=True, fmt='.2f', cmap='YlGnBu', cbar=True)
plt.title("Items' Support Matrix", fontsize=16, y=1.02)
```

```
ax.set_xlabel("Consequents",fontsize=16)
ax.set_ylabel("Antecedents",fontsize=16)
plt.xticks(rotation=45)
plt.yticks(rotation=0)
plt.show()
ur = set(rules['antecedents']).union(rules['consequents'])
print(len(ur))
plt.figure(figsize=(10, 10))
bk_rules = rules
bk_network = bk_rules[['antecedents', 'consequents']]
bk_network_G = nx.from_pandas_edgelist(
   bk_network,
   source = 'antecedents',
   target = 'consequents',
   create_using = nx.DiGraph())
network_bt = nx.in_degree_centrality(bk_network_G)
bk_network_bt = pd.DataFrame(list(network_bt.items()), columns = ['item', 'centrality'])
bk_network_bt = bk_network_bt.dropna(axis=0)
pos = nx.kamada_kawai_layout(bk_network_G)
sizes = [x[1]*100 for x in bk_network_G.degree()]
nx.draw_networkx(bk_network_G, pos,
   with_labels = True,
   node_size = sizes,
   width = 0.1, alpha = 0.8,
   arrowsize = 3, linewidths = 0)
plt.axis('off')
plt.show()
```

В.2 Агоритм Табу-Пошук

```
import pandas as pd
import numpy as np
from mlxtend.frequent_patterns import fpgrowth
from mlxtend.frequent_patterns import association_rules
# Load data
df = pd.read_csv('C:/Users/User/Desktop/GoodsStore307.csv')
uid_sales =
df.pivot_table(values='product_id',columns=['product_name'],index='check_id')#,aggfunc=np.sum)
uid_sales[np.isnan(uid_sales)] = 0
uid_sales[uid_sales>0] = 1
#print(uid_sales.head())
# Convert data to one-hot encoded format
df_onehot = uid_sales#pd.get_dummies(df)
# Perform market basket analysis using FP-Growth algorithm
frequent_itemsets = fpgrowth(df_onehot, min_support=0.05, use_colnames=True)
# Define tabu search function
def tabu_search(frequent_itemsets, num_iterations, tabu_length):
    tabu_list = []
    best_itemset = frequent_itemsets.copy()
    best_support = best_itemset.iloc[0]['support']
    for i in range(num_iterations):
        neighbors = []
        for itemset in best_itemset['itemsets']:
            for item in itemset:
                neighbor = itemset - frozenset([item])
                if neighbor not in tabu_list:
                    neighbors.append(neighbor)
        best_neighbor = None
        best_neighbor_support = 0
        for neighbor in neighbors:
            if frequent_itemsets[frequent_itemsets['itemsets'] == neighbor].empty==False:
                neighbor_support = frequent_itemsets[frequent_itemsets['itemsets'] ==
neighbor]['support'].values[0]
                if neighbor_support > best_neighbor_support:
                    best_neighbor = neighbor
                    best_neighbor_support = neighbor_support
        if best_neighbor_support > best_support:
            best_itemset = frequent_itemsets[frequent_itemsets['itemsets'] == best_neighbor]
            best_support = best_neighbor_support
        tabu_list.append(best_itemset.iloc[0]['itemsets'])
```

```
        if len(tabu_list) > tabu_length:
            tabu_list.pop(0)
    return best_itemset
# Run tabu search algorithm on frequent itemsets
tabu_itemset = tabu_search(frequent_itemsets, num_iterations=100, tabu_length=10)
# Generate association rules
rules = association_rules(tabu_itemset, metric="lift", min_threshold=1)
# Print top 10 rules sorted by lift
print(rules.sort_values(by = ["support","confidence"], ascending=False))
import matplotlib.pyplot as plt
import networkx as nx
import seaborn as sns
rules['antecedents'] = rules['antecedents'].apply(lambda a: ', '.join(list(a)))
rules['consequents'] = rules['consequents'].apply(lambda a: ', '.join(list(a)))
pivot_support = rules.pivot(index='antecedents', columns='consequents', values='support')
sns.set_context("talk")
plt.style.use('ggplot')
plt.subplots(figsize=(12, 16))
sns.set()
ax = sns.heatmap(data=pivot_support, annot=True, fmt='.2f', cmap='YlGnBu', cbar=True)
plt.title("Items' Support Matrix", fontsize=16, y=1.02)
ax.set_xlabel("Consequents",fontsize=16)
ax.set_ylabel("Antecedents",fontsize=16)
plt.xticks(rotation=45)
plt.yticks(rotation=0)
plt.show()
```