**Міністерство освіти і науки України**

**Чернівецький національний університет імені Юрія Федьковича**

**Інститут фізико-технічних та комп’ютерних наук**

(повна назва інституту/факультету)

**Кафедра комп’ютерних наук**

(повна назва кафедри)

КУРСОВА РОБОТА

з дисципліни

«Інтелектуальний аналіз даних»

на тему: **«Система аналізу ринкового кошика покупців мережі продуктових магазинів**»

Виконав: студент 4 курсу, групи 444СК

Напряму підготовки 122 «Комп’ютерні науки»

(шифр і назва спеціальності)

Бараболін Євгеній Володимирович \_

(прізвище та ініціали)

Керівник к.б.н., асистент Талах М.В. \_

(прізвище та ініціали)

Національна шкала \_

Кількість балів \_Оцінка: ECTS \_

Чернівці – 2021

**Чернівецький національний університет імені Юрія Федьковича**

Факультет Кафедра Спеціальність Освітній ступінь Форма навчання курс група

ЗАТВЕРДЖУЮ

Завідувач кафедри



( *підпис*) (*ініціали, прізвище*)

«\_ » \_ 20 року

**ЗАВДАННЯ**

**НА КУРСОВУ РОБОТУ СТУДЕНТА**

(прізвище , ім’я, по батькові)

1. Тема роботи



затверджена наказом від « » 20 року №

1. Термін подання студентом закінченої роботи
2. Вхідні дані до роботи



1. Зміст розрахунково-пояснювальної записки (перелік питань, які треба

розробити)



1. Перелік графічного, наочного матеріалу



1. Консультант(и) курсової роботи

**Календарний план підготовки курсової роботи**

|  |  |  |  |
| --- | --- | --- | --- |
| № | Назва етапів курсової роботи | Термін виконання етапів роботи | Примітки |
| 1 | Отримання завдання на курсову роботу |  |  |
| 2 | Аналіз предметної області, дослідження літератури та матеріалів на задану тему |  |  |
| 3 | Аналіз існуючих аналогів програмного забезпечення |  |  |
| 4 | Постановка задачі за темою курсової роботи |  |  |
| 5 | Вибір інструментальних засобів розробки системи |  |  |
| 6 | Проєктування структури та алгоритму роботи розроблюваної системи |  |  |
| 7 | Розробка компонентів системи |  |  |
| 8 | Розгортання та тестування розробленої системи |  |  |
| 9 | Написання розділів пояснювальної записки |  |  |
| 10 | Представлення закінченої роботи на перевірку |  |  |
| 11 | Захист курсової роботи |  |  |

**Студент**

(*підпис*)

**Керівник**

(*підпис*) (*прізвище , ім’я, по батькові*)

«\_ » 20 року

**АНОТАЦІЯ**

У курсовій роботі розроблено інформаційну систему, що забезпечує аналіз країн світу за рівнем життя з використанням методів інтелектуального аналізу даних. Досліджено різні фактори які впливають на рівень життя в різних країнах. Проаналізовано вимоги до інформаційних систем аналізу, здійснено характеристику їх функціональних можливостей, обґрунтовано вибір алгоритму \*\*\* для здійснення аналізу. Обґрунтовано вибір \*\*\*\* у якості джерела вхідних даних продемонстровано підходи до їх отримання. Показано ефективність застосування алгоритму(ів) \*\*\* для підготовки масиву вхідних даних. Налаштування параметрів моделі проводилось з використанням \*\*\*. Проведена оцінка якості моделі показала \*\*\*.З врахуванням поставлених завдань та форматів отриманих даних були обрані наступні види візуалізацій\*\*\*.

Пояснювальна записка складається з трьох розділів загальним обсягом \* сторінки, містить \* таблиці, \* додатки, \* використаних джерел.

**Ключові слова**: Data Mining, Python, алгоритм k-means, задача кластеризації, заклад вищої освіти

**ЗМІСТ**

[ПЕРЕЛІК СКОРОЧЕНЬ, УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ ТА ТЕРМІНІВ](#_4h6666m2kshu) 7

[ВСТУП](#_ygynjw1llhye) 8

1. [ОГЛЯД ТА ПОРІВНЯЛЬНИЙ АНАЛІЗ АНАЛОГІВ ОБ’ЄКТА ДОСЛІДЖЕННЯ](#_mr248aoxwrhc) 9
   1. [Аналіз предметної області. Обґрунтування необхідності створення програмного продукту](#_k5qnhvn932ya) 9
   2. [Обґрунтування вибору алгоритмів та технічних засобів вирішення поставленої задачі](#_f9dl3rdoxevm) 10
2. [ПРОЄКТУВАННЯ ТА РОЗРОБКА МОДУЛЮ АНАЛІЗУ ДАНИХ](#_f7b9yypfv2y5) 11
   1. [Опис вхідних та вихідних даних](#_vexvk0srkc83) 11
   2. [Проєктування та розробка повного конвеєра аналізу даних](#_r2ooe5g8sne6) 12
3. [ДЕМОНСТРАЦІЯ РОБОТИ МОДУЛЮ НА ОСНОВІ КОНКРЕТНИХ ДАНИХ](#_wvhcctmooy72) 13

[ВИСНОВКИ](#_o67a3he4p92z) 14

[СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ](#_urfyiv66kapk) 15

[ДОДАТКИ](#_7daq091so3mz) 16

[ДОДАТОК А](#_it2rshlrcu08) 16

[ДОДАТОК Б](#_yb2gf3dyy90o) 17

# 

# **ПЕРЕЛІК СКОРОЧЕНЬ, УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ ТА ТЕРМІНІВ**

|  |  |
| --- | --- |
| Скорочення | Пояснення |
| КР | курсова робота |
| POS | point-of-sale |
| FPGrowth | Frequent Pattern Growth |
| САРК | система аналізу ринкового кошика |
| VS Code | Visual Studio Code |

# **ВСТУП**

Аналіз ринкового кошика є одним із ключових методів, які використовують великі роздрібні торговці, щоб виявити приховані асоціації між товарами. Аналіз ринкового кошика використовує дані про транзакції, тобто список усіх товарів, куплених клієнтом за одну покупку, щоб визначити, які продукти замовлені або куплені разом, і визначити закономірності спільного появи. Цей потужний аналіз допомагає виявити вподобання споживачів, які буде дуже складно визначити за допомогою онлайн-опитування. Роздрібні продавці використовують результати аналізу ринкового кошика, щоб керувати розміщенням продуктів у магазинах, міжкатегорійними та спільно-маркетинговими просуваннями, серед іншого.

# **ОГЛЯД ТА ПОРІВНЯЛЬНИЙ АНАЛІЗ АНАЛОГІВ ОБ’ЄКТА ДОСЛІДЖЕННЯ**

## **Аналіз предметної області. Обґрунтування необхідності створення програмного продукту**

Об’єктом дослідження в даній КР є система аналізу ринкового кошика покупців, а предметом КР є розробка системи аналізу ринкового кошика покупців.

Аналіз ринкового кошика – це метод аналізу даних, який використовується роздрібними продавцями для збільшення продажів шляхом кращого розуміння моделей купівлі клієнтів. Це передбачає аналіз великих наборів даних, таких як історія покупок, щоб виявити групи продуктів, а також продукти, які, ймовірно, будуть придбані разом.

Впровадженню аналізу ринкового кошика сприяла поява систем електронних точок продажу (POS). У порівнянні з рукописними записами, які зберігаються власниками магазинів, цифрові записи, створені POS-системами, полегшують додаткам обробку та аналіз великих обсягів даних про покупки.

Реалізація аналізу ринкового кошика вимагає досвіду в галузі статистики та науки про дані, а також деяких навичок алгоритмічного комп’ютерного програмування. Для тих, хто не має необхідних технічних навичок, існують комерційні, стандартні інструменти такі як Microsoft Excel, Alteryx і Tableau, але головним їхнім недоліком являється те, що вони платні.

Одним із прикладів є інструмент «Аналіз кошика покупок» у Microsoft Excel, який аналізує дані транзакцій, що містяться в електронній таблиці, і виконує аналіз ринкового кошика. Елементи, що підлягають аналізу, мають бути пов’язані за ідентифікатором транзакції. Потім інструмент «Аналіз кошика» для покупок створює два робочі аркуша: робочу таблицю «Групи товарів» кошика для покупок, у якій перелічено товари, які часто купуються разом, і робочу таблицю «Правила кошика», яка показує, як товари пов’язані (наприклад, покупці продукту «А», ймовірно, куплять товар «В»).

**Види аналізу ринкового кошика**

Існує два типи аналізу ринкового кошика:

* прогнозний аналіз ринкового кошика: цей тип розглядає предмети, куплені в послідовності, щоб визначити перехресний продаж;
* аналіз диференційного ринкового кошика: цей тип враховує дані в різних магазинах, а також покупки від різних груп клієнтів у різний час доби, місяця чи року. Якщо правило діє в одному вимірі (наприклад, магазин, проміжок часу або група клієнтів), але не діє в інших, аналітики можуть визначити фактори, які є винятками. Ці знання можуть привести до нових пропозицій продуктів, які сприятимуть зростанню продажів;

**Алгоритми, пов'язані з аналізом ринкового кошика**

При аналізі ринкового кошика, асоціативні правила використовуються для прогнозування ймовірності того, що продукти будуть куплені разом. асоціативні правила підраховують частоту елементів, які зустрічаються разом, намагаючись знайти асоціації, які зустрічаються набагато частіше, ніж очікувалося.

Для розуміння того що таке асоціативні правила розглянемо приклад. Припустимо, що є 100 клієнтів, 10 з них купили молоко, 8 купили масло і 6 купили обидва продукти. З відси випливає що якщо людина купила молоко, то вона купить масло, або If {молоко} ⇒ Then {масло}.

Асоціативні правила мають наступні поняття:

* Підтримка (support) – це показник того, як часто набір елементів з’являється в наборі даних. Підтримка має формулу наведену нижче (1.1.1).

|  |  |
| --- | --- |
|  | (1.1.1) |

де *A* і *B* – це окремі транзакції, які були здійснені в межах загальної сукупності записаних операцій.

*P* – це частота транзакції *A* разом з *B*.

Тобто, з нашого приклада випливає що підтримка молока з маслом дорівнює наступному:

, тобто 6%.

* Впевненість (confidence) – це відсоток усіх транзакцій, які задовольняють *A*, які також задовольняють *B*. Впевненість визначається формулою зазначеною нижче (1.1.2).

|  |  |
| --- | --- |
|  | (1.1.2) |

Тобто, з нашого приклада випливає що впевненість молока з маслом дорівнює наступному:

, тобто 60%

* Підвищення (lift) правила - це відношення спостережуваної підтримки до очікуваної, якби *A* і *B* були незалежними, і визначається як наступна формула (1.1.3).

|  |  |
| --- | --- |
|  | (1.1.3) |

Тобто, з нашого приклада випливає що впевненість молока з маслом дорівнює наступному:

.

Якби правило підвищення дорівнювало 1, це означало б, що ймовірність виникнення антецеденту та ймовірності наступного незалежні одна від одної. Якщо дві події не залежать одна від одної, не можна скласти правило, що стосується цих двох подій.

Якщо підвищення більше ніж 1, це дає нам знати, наскільки ці два випадки залежать один від одного, і робить ці правила потенційно корисними для прогнозування наслідків у майбутніх наборах даних.

Якщо підвищення менше ніж 1, це дає нам знати, що елементи замінюють один одного. Це означає, що наявність одного предмета негативно впливає на наявність іншого, і навпаки.

Цінність підвищення полягає в тому, що воно враховує як підтримку правила, так і загальний набір даних.

Деякі добре відомі алгоритми – це Apriori, Eclat і FPGrowth, але вони виконують лише половину роботи, оскільки є алгоритмами для пошуку частих наборів елементів. Ще один крок потрібно зробити після того, щоб створити правила з частих наборів елементів, що знаходяться в базі даних.

Алгоритми, які використовують правила асоціації, включають AIS, SETM, FPGrowth, ECLAT та Apriori. Алгоритм Apriori зазвичай цитується науковцями з даних у дослідницьких статтях про аналіз ринкового кошика, і він використовується для визначення частих елементів у базі даних, а потім оцінки їх частоти, коли набори даних розширюються до більших розмірів.

## **Обґрунтування вибору алгоритмів та технічних засобів вирішення поставленої задачі**

**Приклади аналізу ринкового кошика**

На вебсайті Amazon використовується добре відомий приклад аналізу ринкового кошика. На сторінці продукту Amazon представляє користувачам пов’язані товари під заголовками «Часто купують разом» і «Клієнти, які купили цей товар, також купили».

Аналіз ринкового кошика також поширюється на звичайні магазини. Якщо аналіз показав, що покупки в журналах часто включають покупку закладки (що можна вважати несподіваною комбінацією, оскільки споживач не купував книгу), то книжковий магазин міг би розмістити вибір закладок біля журнальної стійки.

**Переваги аналізу ринкового кошика**

Аналіз ринкового кошика може збільшити продажі та задоволеність клієнтів. Використовуючи дані, щоб визначити, що продукти часто купуються разом, роздрібні продавці можуть оптимізувати розміщення продуктів, пропонувати спеціальні пропозиції та створювати нові пакети продуктів, щоб стимулювати подальший продаж цих комбінацій.

Ці вдосконалення можуть створити додаткові продажі для роздрібного продавця, водночас роблячи покупки більш продуктивними та цінними для клієнтів. Використовуючи аналіз ринкового кошика, клієнти можуть відчувати сильніші настрої або лояльність до бренду до компанії.

**Вибір алгоритму**

Основним алгоритмом пошуку асоціативних правил є алгоритм Apriori. Оскільки алгоритм Apriori є першим алгоритмом, який був запропонований у цій галузі, він був покращений з точки зору обчислювальної ефективності (тобто були зроблені швидші альтернативи).

Є дві швидші альтернативи алгоритму Apriori, які є сучасними: одна з них – FPGrowth, а інша – ECLAT. Між FPGrowth та ECLAT немає очевидного переможця з погляду часу виконання: воно залежатиме від різних даних та різних налаштувань в алгоритмі.

Для аналізу ринкового кошика був вибраний алгоритм FPGrowth, оскільки, як було зазначено вище, він швидший й використовує менше пам'яті, що є критичним для систем з обмеженим обсягом пам'яті. Також, немало важливим фактором є те, що цей алгоритм присутній в модулі Mlxtend для мови програмування Python, яка і використовувалася для створення цієї системи.

**Асоціативні правила**

Асоціативні правила широко використовуються для аналізу даних про ринкових кошиків або даних про транзакції, та призначені для виявлення суворих правил у даних транзакцій, з використанням показників ступеня інтересу, заснованих на концепції суворих правил.

Для створення САРК було вибрано мову програмування Python, а для створення вебінтерфейсу — JavaScript.

Мова програмування Python — це одна із найпопулярніших мов програмування, її часто використовують для машинного навчання й аналізу даних. В цій КР використовувалися наступні модулі для Python:

* Pandas. Pandas — бібліотека з відкритим вихідним кодом на Python. Він забезпечує готові до використання високопродуктивні структури даних та інструменти аналізу даних. Модуль Pandas працює поверх NumPy, і він широко використовується для науки та аналітики даних;
* Matplotlib. Matplotlib — це повна бібліотека для створення статичних, анімованих та інтерактивних візуалізацій на Python. Matplotlib робить легкі речі легкими, а важкі можливими;
* Mlxtend. Mlxtend (розширення для машинного навчання) — це бібліотека Python корисних інструментів для повсякденних завдань з обробки та аналізу даних;
* Flask. Flask — фреймворк для створення вебзастосунків мовою програмування Python, що використовує набір інструментів Werkzeug, а також шаблонізатор Jinja2. Належить до категорії так званих мікрофреймворків - мінімалістичних каркасів вебдодатків, які свідомо надають лише самі базові можливості.

Для створення вебінтерфейсу використовувався фреймворк Vue.js. Vue.js - JavaScript-фреймворк з відкритим вихідним кодом для створення інтерфейсів користувача. Легко інтегрується у проєкти з використанням інших JavaScript-бібліотек. Може функціонувати як веб-фреймворк для розробки односторінкових програм у реактивному стилі.

На даний момент підтримується автором Еваном Ю та іншими активними членами основної команди з різних компаній, таких як Netlify, Netguru, Baidu, Livestorm.

Опитування, проведене у 2016 році для JavaScript, показало, що Vue має 89% задоволеності розробників. На GitHub проєкт має 188 тисяч зірок є третім за величиною проєкт в історії Github.

# **ПРОЄКТУВАННЯ ТА РОЗРОБКА МОДУЛЮ АНАЛІЗУ ДАНИХ**

## **Опис вхідних та вихідних даних**

Спочатку САРК зчитує файл формату «csv», вона повинна мати 8 стовпців під назвами, опис стовпців можна побачити в наступній таблиці (табл. 2.1.1).

Таблиця 2.1.1 - Опис стовпців

|  |  |  |  |
| --- | --- | --- | --- |
| № | Назва стовпця | Тип даних | Опис |
| 1 | UserId | int64 | Унікальний ідентифікатор користувача. |
| 2 | TransactionId | int64 | Унікальний ідентифікатор транзакції. Якщо той самий TransactionId міститься в кількох рядках, тоді всі ці продукти купуються разом в одній транзакції. |
| 3 | TransactionTime | object | Час, у який виконується транзакція. |
| 4 | ItemCode | int64 | Унікальний ідентифікатор придбаного товару. |
| 5 | ItemDescription | object | Простий опис придбаного продукту. |
| 6 | NumberOfItemsPurchased | int64 | Кількість продукту, придбаного в транзакції. |
| 7 | CostPerItem | float64 | Ціна за кожну одиницю товару. |
| 8 | Country | object | Країна, з якої здійснено покупку. |

В цій таблиці знаходять різні данні про транзакції. Приклад даних які знаходяться у таблиці можна побачити нижче (рисунок 2.1.1).

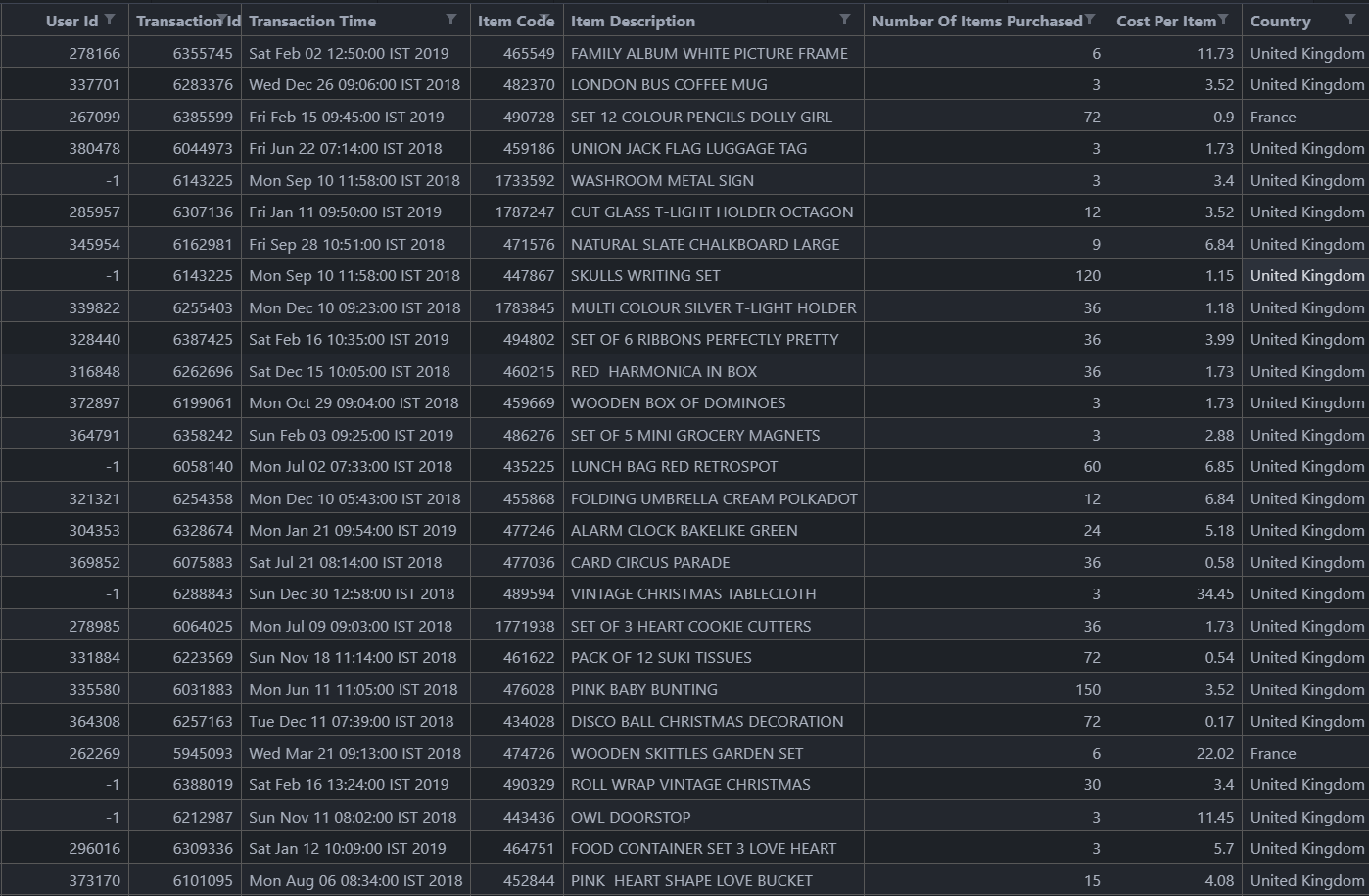


Рисунок 2.1.1 – Структура вхідних даних

Вхідними даними є параметри, які можна задати через вебінтерфейс, цими параметрами є: *support*, *confidens* і *lift*.

Вихідними даними є асоціативні правила, які можна переглянути через вебінтерфейс у вигляді таблиці, або зберегти файл з розширенням «csv». Приклад вихідних даних зображено нижче (рисунок 2.1.2).



Рисунок 2.1.2 – Структура вихідних даних

## **Проєктування та розробка повного конвеєра аналізу даних**

**Середовище розробки**

Проєктування та розробка САРК проводилася в програмному забезпеченні Visula Studio Code.

Visual Studio Code поєднує в собі простоту редактора вихідного коду з потужними інструментами розробника, такими як завершення та налагодження коду IntelliSense.

Перш за все, це редактор, який виходить за рамки простого редактора коду. Дивовижний цикл редагування, збирання і налагодження програмного застосунку означає менше часу на те, щоб возитися з вашим середовищем, і більше часу на реалізацію ваших ідей.

В основі Visual Studio Code - блискавичний редактор вихідного коду, що ідеально підходить для повсякденного використання. Завдяки підтримці сотень мов VS Code допоможе вам миттєво підвищити продуктивність завдяки виділенню синтаксису, зіставленню дужок, автоматичному відступу, виділенню поля, фрагментам та багато іншого. Інтуїтивно зрозумілі сполучення клавіш, просте налаштування та зіставлення сполучень клавіш від спільноти дозволяють з легкістю переміщатися за кодом.

Для серйозного кодування вам часто будуть корисні інструменти з більшим розумінням коду, ніж блоки тексту. Visual Studio Code включає вбудовану підтримку автозавершення коду IntelliSense, глибокого розуміння семантичного коду та навігації, а також рефакторинг коду.

А коли код стає складним, найскладніше – налагодження. Налагодження часто є тією функцією, яку розробники найбільше не беруть до уваги при більш компактному кодуванні, тому VS Code зробили це. Visual Studio Code включає інтерактивний налагоджувач, тому можна виконувати покрокове виконання вихідного коду, перевіряти змінні, переглядати стеки викликів і виконувати команди в консолі.

VS Code також інтегрується із засобами складання та створення сценаріїв для виконання спільних завдань, прискорюючи повсякденні робочі процеси. VS Code підтримує Git, тому ви можете працювати із системою керування версіями, не виходячи з редактора, включаючи перегляд змін в очікуваних змінах.

VS Code дозволяє налаштовувати кожну функцію на свій смак і встановлювати будь-яку кількість сторонніх розширень. Хоча більшість сценаріїв працюють "з коробки" без будь-якої конфігурації, VS Code також росте разом з вами, і вони рекомендують оптимізувати свій досвід відповідно до наших унікальних потреб. VS Code - це проєкт з відкритим вихідним кодом, тому будь-хто може зробити свій внесок у зростаючу та активну спільноту на GitHub.

**Аналіз даних**

Спочатку система зчитує файл з транзакціями, який називається «transactions\_data.csv», у функції.

Ми будемо користуватися з Python і пакетом pandas. Ми також будемо використовувати бібліотеку Mlxtend, оскільки вона містить вбудовані функції для алгоритму FPGrowth та асоціативні правила.

Спочатку завантажуються необхідні бібліотеки. Далі дані зчитуються у фрейм даних pandas і дізнаємося про ознаки даних зображено на наступному рисунку (рисунок 2.2.1).

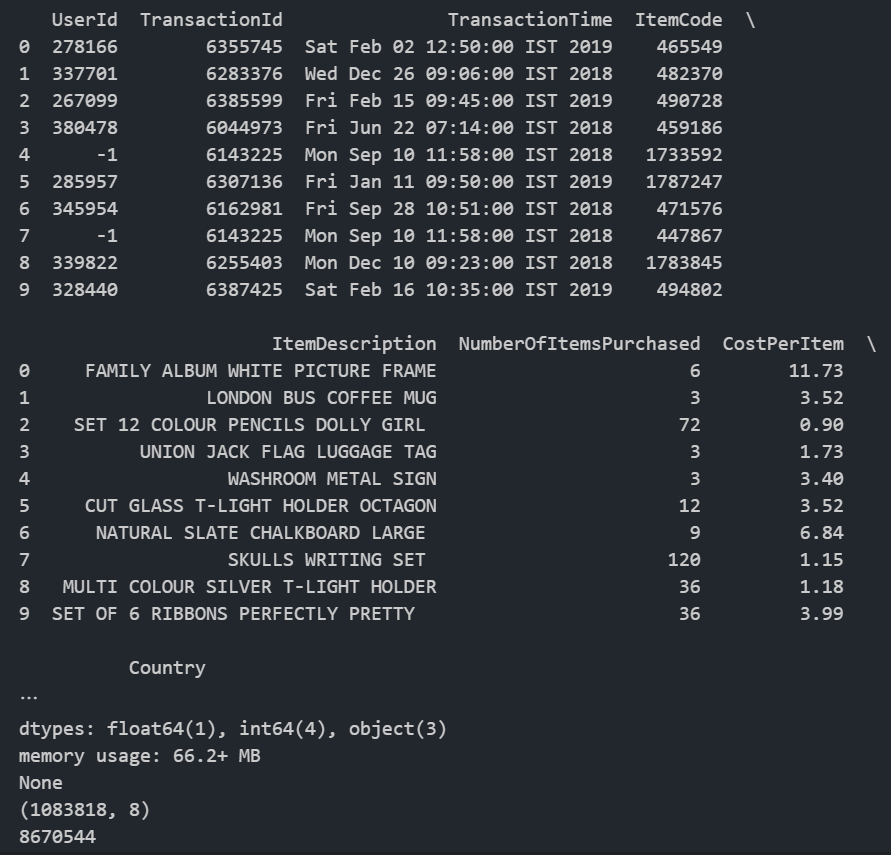


Рисунок 2.2.1 – Інформація фрейм даних pandas

Потім виконується дослідний аналіз та очищення даних шляхом видалення всіх рядків зі значеннями менше ніж 0, пустими значенням та роком, рівним «2028». Після виконаних операцій ми отримаємо наступний фрейм даних (рисунок 2.2.2).



Рисунок 2.2.2 – Перші 10 рядків фрейму даних після обробки

Подивімося, скільки транзакцій відбувається кожного місяця року. По суті ми створюємо стовпець з ім'ям «month\_year», щоб розділити точки даних на місяць, в якому вони відбулися. Тоді ми приймаємо число унікальних транзакцій, що відбуваються щомісяця, та будуємо графік за допомогою бібліотеки Matplotlib (рисунок 2.2.3).

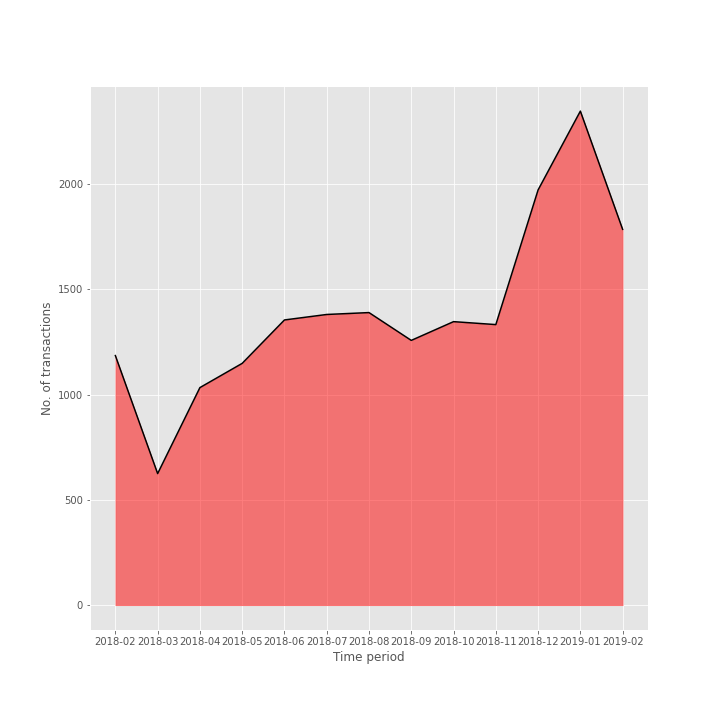


Рисунок 2.2.3 – Кількість унікальних транзакцій протягом кожного місяця

Як ми бачимо, з часом інтернет-магазин отримує дедалі більше покупок, максимум яких припав на 19 січня.

Давайте досліджуємо кількість товарів, куплених у кожній транзакції. Як ми бачимо, мінімальна кількість елементів дорівнює 1, а максимальна – 540. Отже, нам потрібно побудувати нашу гістограму, як показано нижче (рисунок 2.2.4).

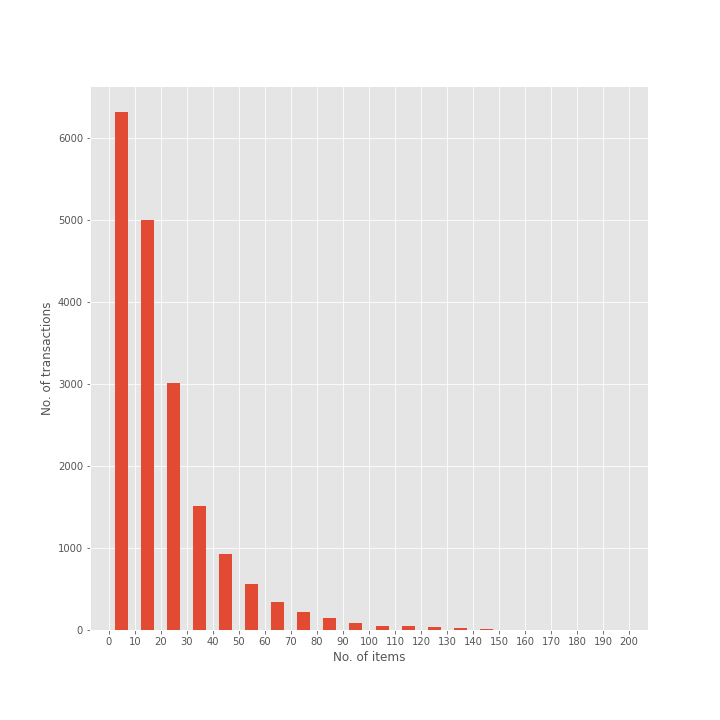


Рисунок 2.2.4 – Кількість товарів, куплених у кожній транзакції

Дізнаймося товари які продаються найкраще. Це можна інтерпретувати по-різному, наприклад, предмет, який приносить максимальний дохід, предмети, які були виявлені внаслідок максимальної кількості транзакцій тощо. Ми розглядатимемо предмети, які приносять максимальний дохід (рисунок 2.2.5).

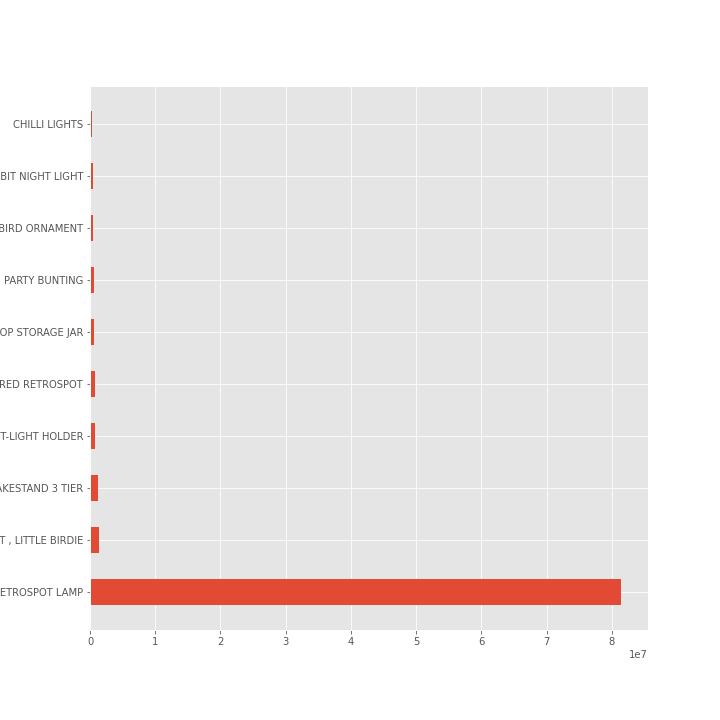


Рисунок 2.2.5 – Товарів які приносять максимальний дохід

З наведеного вище графіка, що показує 10 найпопулярніших товарів, ми можемо бачити, що «ретро точна лампа» має максимальну вартість продажу.

Кількість унікальних *TransactionId* становить 18334, а кількість унікальних *ItemDescription* - 3871.

Тепер, коли у нас є дані після всієї попередньої обробки, впорядкуймо їх за допомогою *TransactionId* як індекс і *ItemDescriptions* як стовпці із загальною кількістю елементів, куплених у кожній транзакції кожного елемента, як точка даних.

Отримано такий фрейм даних (рисунок 2.2.6).

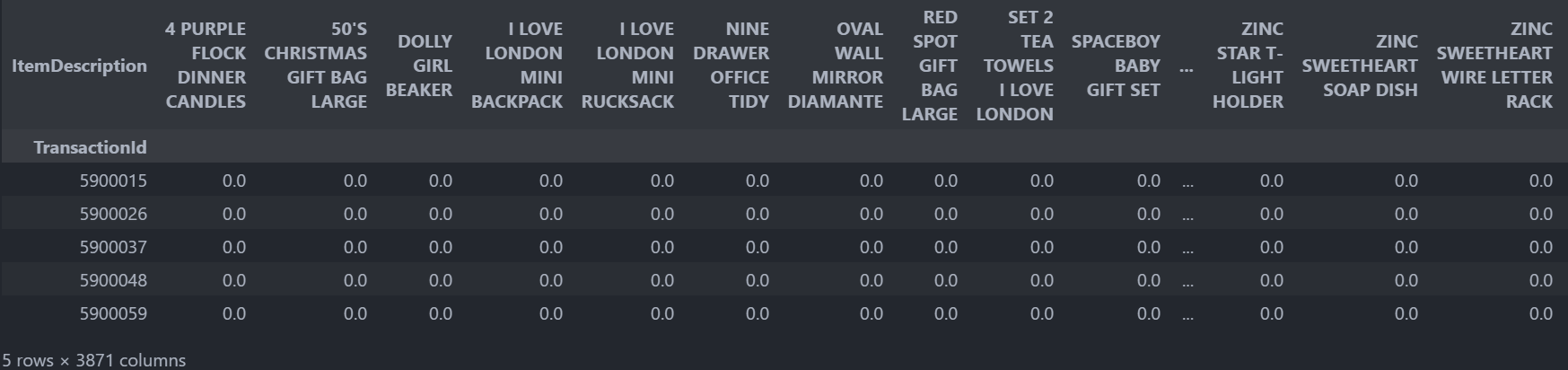


Рисунок 2.2.6 – Оброблений фрейм даних

Нам потрібно переконатися, що будь-які позитивні значення закодовані в одиницю, а всі від’ємні значення (якщо такі є) в нуль (рисунок 2.2.7).

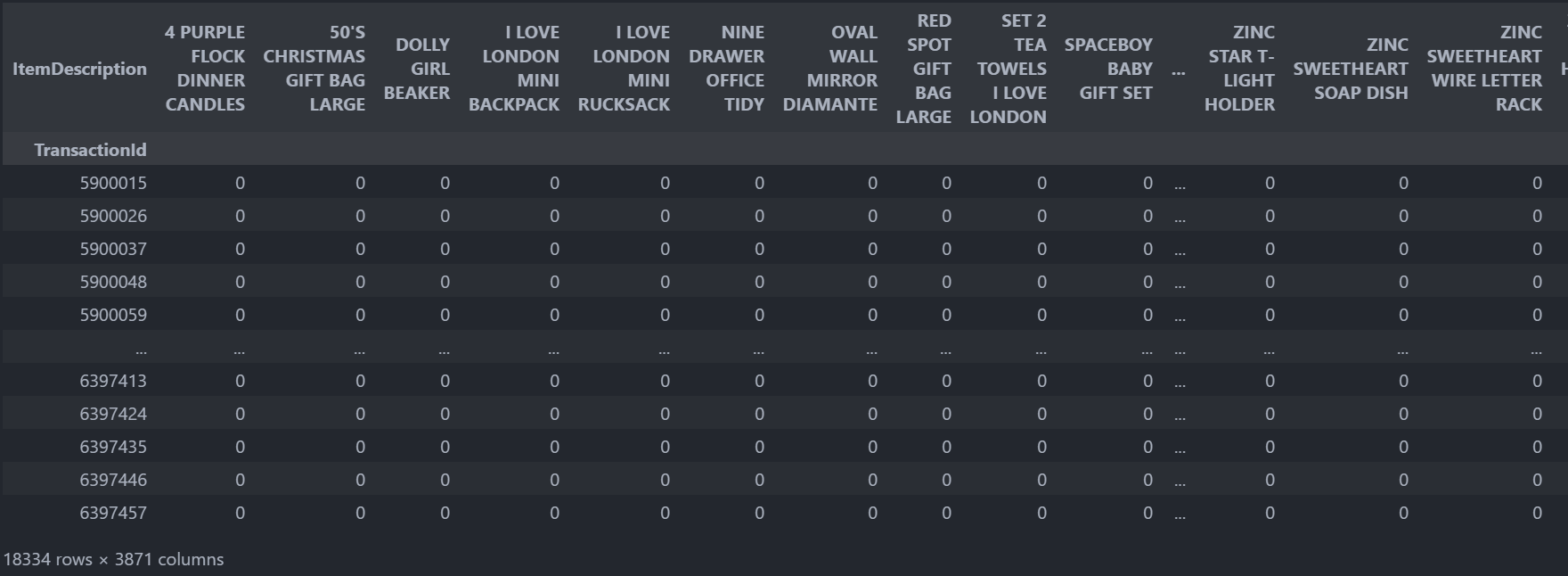


Рисунок 2.2.7 – Закодований фрейм даних

Коли фрейм даних готовий, ми можемо застосувати алгоритм FPGrowth, щоб отримати наборів товарів, що часто купуються разом. Тут мінімальне порогове значення підтримки встановлено на рівні 1,5%. Отримуємо наступні набори предметів (Рисунок 2.2.8).



Рисунок 2.2.8 – Наборів товарів, що часто купуються разом зі значення підтримки на рівні 1,5%

Потім ми розташовуємо набори елементів у порядку спадання їх підтримки, що дає нам наступний рисунок (рисунок 2.2.9).

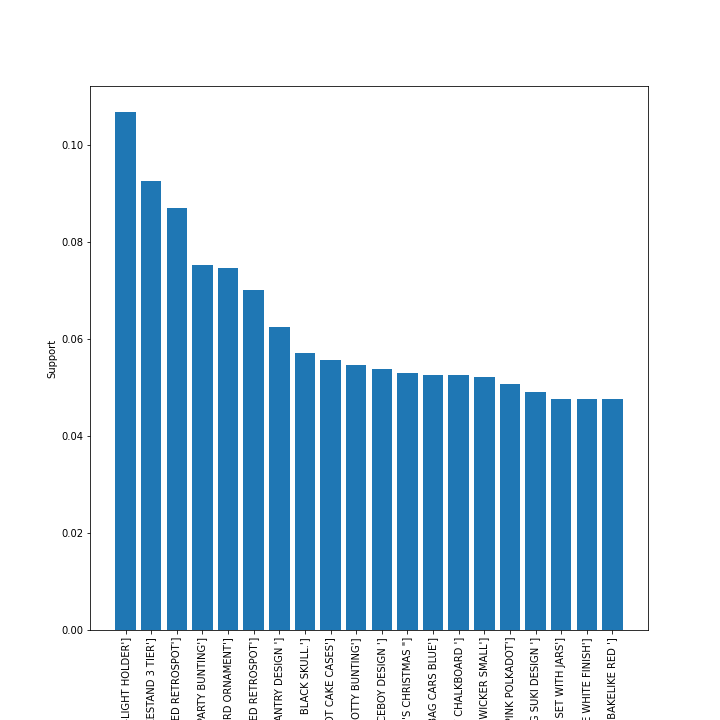


Рисунок 2.2.9 – Наборів товарів, що часто купуються разом відсортовані за підтримкою

Потім ми застосовуємо асоціативні правила до цих наборів елементів, сформованих за допомогою алгоритму FPGrowth. Тут використовується показник впевненості, а його мінімальне порогове значення встановлено на 0,2. Отримано наступний фрейм даних (рисунок 2.2.10).



Рисунок 2.2.10 – Асоціативні правила, які сформовані за допомогою алгоритму FPGrowth

Всі антецеденти з відповідними консеквентами перераховані за їх індивідуальною підтримкою, загальною підтримкою набору елементів та іншими показниками (рисунок 2.2.11).



Рисунок 2.2.10 – Асоціативні правила, які сформовані за допомогою алгоритму FPGrowth

Зведення кадрів даних правил дає нам таку інформацію:

* усього існує 230 правил;
* зведення різних показників.

Візьмімо головні правила із максимальною впевненістю.

asdf

a

# 

# **ДЕМОНСТРАЦІЯ РОБОТИ МОДУЛЮ НА ОСНОВІ КОНКРЕТНИХ ДАНИХ**

Демонстрація

# **ВИСНОВКИ**

Висновки

# **СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ**

Список

# **ДОДАТКИ**

## **ДОДАТОК А**

Додаток А

## **ДОДАТОК Б**

Додаток Б