ЗВІТ З ПЕРЕВІРКИ НА ПЛАГІАТ

ЦЕЙ ЗВІТ ЗАСВІДЧУЄ, ЩО ПРИКРПЛЕНА РОБОТА
Кононенко звіт
БУЛА ПЕРЕВІРЕНА СЕРВІСОМ ДЛЯ ЗАПОБІГАННЯ ПЛАГІАТУ
MY.PLAG.COM.UA І МАЄ:
СХОЖІСТЬ

1%

РИЗИК ПЛАГІАТУ

16%

ПЕРЕФРАЗУВАННЯ

0%

НЕПРАВИЛЬНІ ЦИТУВАННЯ

0%

Назва файлу: Кононенко А.В. (ІПЗ-111М).docx
Файл перевірено: 2023-01-12
Звіт створено: 2023-01-12

МІНІСТЕРСТВО ОСВІТИ ТА НАУКИ УКРАЇНИ
ПрАТ «ПРИВАТНИЙ ВИЩИЙ НАВЧАЛЬНИЙ ЗАКЛАД
«ЗАПОРІЗЬКИЙ ІНСТИТУТ ЕКОНОМІКИ
ТА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ» (mon.gov.ua)

Кафедра економічної кібернетики та інженерії програмного забезпечення

ДО ЗАХИСТУ ДОПУЩЕНИЙ
Зав.кафедри _____
д.е.н., доц. Левицький С.І.

ДИПЛОМНА РОБОТА

Моделювання дата-майнингу в розробці додатків для онлайн-торгівлі

Виконав ст. гр. ІПЗ-111М _____ Кононенко А.В.
Керівник д.е.н., проф. _____ Левицький С.І.

Запоріжжя
2022

РЕФЕРАТ

Магістерська дипломна робота: 85 сторінок, 16 рисунків, 26 джерел.

Мета магістерської дипломної роботи – дослідження методів та моделей дата-майнингу в сфері онлайн-торгівлі.

Об’єкт дослідження: застосування системи створення аналітичної звітності у онлайн-магазині з використанням інструментів дата-майнингу.

Методи дослідження: аналіз сучасних методів та класифікації моделей в сучасній теорії дата-майнингу.

Наукова новизна отриманих результатів полягає в пошуці найбільш точних та менш ресурсовитратних методів аналізу і прогнозування для випадку аналізу даних в сфері онлайн-торгівлі, реалізація прогнозування доходу з використанням регрессії.

Практична цінність результатів полягає у розробці клієнт-серверного web-додатоку, що буде надавати послуги аналітичної обробки даних, перетворюючи та агрегуючи надані дані продажів, інформації про магазин, користувачів та товари.

Область застосування: аналітика в сфері онлайн-торгівлі.

Значення роботи та висновки: використовуючи інструменти дата-майнингу, розроблено алгоритми для формування аналітичного звіту за наданий період роботи онлайн-магазину. Реалізований алгоритм для прогнозування величини доходу в майбутні періоди. Робота алгоритмів продемонстрована на масиві згенерованих даних.

DATA-MINING, МОДЕЛЬ DATA-MINING, KDD, ІНТЕЛЕКТУАЛЬНИЙ АНАЛІЗ, ДАНІ, БАЗА ДАНИХ, МІКРОСЕРВІСНА АРХІТЕКТУРА

ЗМІСТ

ВСТУП

Аналітична діяльність (аналітика) є напрямом інтелектуальної діяльності людей, яке спрямоване на вирішення завдань, що виникають у різних сферах життя. Аналітична діяльність стає найважливішою характеристикою сучасного суспільства.

З процесом інформаційного розвитку людства, з’явилась можливість перенести велику частину аналітичного апарату у сферу інформаційних технологій – починаючи з дата-майнингу, та закінчуючи аналізом існуючих даних, створенням аналітичних звітів та навіть прогнозування майбутнього розвитку процесів на основі існуючих даних.

Аналітична діяльність (аналітика) є напрямом інтелектуальної діяльності людей, яке спрямоване на вирішення завдань, що виникають у різних сферах життя. Аналітична діяльність стає найважливішою характеристикою сучасного суспільства.

З процесом інформаційного розвитку людства, з’явилась можливість перенести велику частину аналітичного апарату у сферу інформаційних технологій – починаючи з дата-майнингу, та закінчуючи аналізом існуючих даних, створенням аналітичних звітів та навіть прогнозування майбутнього розвитку процесів на основі існуючих даних.

Актуальність теми. Інформаційна аналітика в сучасному світі - це міждисциплінарна галузь, що використовує сукупність прийомів та методів фундаментальної та прикладної науки для аналізу та вирішення завдань у різних сферах людської діяльності.

Метою дослідження є розробка програми для формування аналітичного звіту торгівельної діяльності онлайн-магазину, використовуючи наданий масив даних про покупців, покупки, перегляди товарів за певний період часу, а також формування передбачень про розмір доходу за певний період у майбутньому.

Об'єктом дослідження у межах цієї роботи є застосування системи створення аналітичної звітності у онлайн-магазині з використанням інструментів дата-майнингу.

Об’єктом розробки є клієнт-серверний web-додаток, що буде надавати послуги аналітичної обробки даних, перетворюючи та агрегуючи надані дані

продаж, інформації про магазин, користувачів, інформацію відвідувань та перегляду товарів.

Наукова новизна. Надання послуг з аналізу даних - це напрямок, що бурхливо і всебічно розвивається у сьогоденні. Пошук найбільш точних чи менш ресурсовитратних методів аналізу і прогнозування, та розуміння того, у яких випадках використовувати той чи інший алгоритм – це актуальна задача пошуку оптимального аналітичного методу для певної ситуації. В роботі описана реалізація моделей кластеризації даних та прогнозування доходу з використанням регрессії.

РОЗДІЛ 1 ПОСТАНОВКА ЗАДАЧ ДОСЛІДЖЕННЯ І ОГЛЯД ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Аналітика даних у сучасному цифровому світі

В даний час практично всі компанії визнають високу цінність відданих клієнтів. Багато з них намагаються впроваджувати програми підвищення лояльності, адже загальновизнано, що утримання старих клієнтів обходиться компанії дешевшим, ніж залучення нових. Економляться витрати на рекламу, різні акції з просування товарів та послуг, на винагороду торгових агентів, заохочуваних за залучення нових клієнтів, тощо. А сама лояльність перетворюється на один із головних критеріїв успішності бізнесу.

Значення лояльності як конкурентоспроможності підтверджується конкретними статистичними даними. За словами Ф. Ф. Райхельда (куратора програми «Лояльність на практиці» консалтингової компанії Bain & Company та автора «Ефекту лояльності»), низький рівень лояльності в бізнес-середовищі може знизити ефективність економічної діяльності на 25-50% , іноді навіть більше. Залежно від галузі збільшення утримання клієнтів на 5% збільшує середню вартість покупки клієнта на 25-100%. У більшості галузей прибуток на клієнта зростає в міру того, як компанія працює з ним. Часто потрібно кілька нових клієнтів, щоб компенсувати втрату одного старого клієнта.

Однак мало де існує повноцінна працююча програма з утримання клієнтів та залучення нових. У більшості випадків всі дії компаній зводяться до так званих прямих методів заохочення, адресованих всім клієнтам: до нарахування та списання бонусів, до роздачі знижок, проведення разових акцій. Вочевидь, що у всіх випадках бажана мета остаточно не досягається. Поодинокі операції не дозволяють працювати з лояльністю на регулярній основі, а роздача знижок усім без урахування потреб та зацікавленості призводить до втрат. Знижки потрібно надавати клієнтам, враховуючи їхню цінність для компанії, інакше витрати на їх утримання можуть перевищити доходи від роботи з ними. Тим більше прихильність клієнтів конкретної компанії за фактом визначається не тільки активними діями торгових компаній щодо їх утримання та корисними персональними пропозиціями, а й

зовсім іншими факторами, серед яких можна відзначити: розташування магазину, ціна товарів, їх якість, рівень сервісу та ін. Як показав досвід зарубіжних компаній, методи прямого заохочення клієнтів часто призводять до здійснення разових покупок і не мотивують споживача і надалі залишатися клієнтом однієї компанії [1-6].

Коли керівництво будь-якої компанії приходить до висновку про те, що потрібно щось змінювати у своїй роботі з клієнтами та вирішує запустити програму лояльності, то для її старту та подальшого повноцінного функціонування необхідно відповісти на безліч важких питань:

1. Хто є клієнтами магазину, на які сегменти вони розбиті та що відрізняє один сегмент від іншого?
2. Які чинники впливають поведінку клієнтів, яка структура споживання?
3. Через які канали на них можна вплинути, яка віддача від цього?
4. Як виміряти лояльність, які фактори говорять про зміну тенденцій?
5. і багато іншого

Вирішення задачі без дотримання певних методик неможливе.

Тож розглянемо різноманітні методи аналізу.

1.2 Методи аналізу та моделювання у сфері онлайн-торгівлі

Вага до ціни товару має приділятися у компанії постійно. Регулярний моніторинг та уточнення цінового позиціонування практично завжди призводять до зростання продажів продукту. Правильна ціна допомагає підкреслити якість товару, допомагає покупцю продемонструвати свій соціальний статус оточуючим, може стимулювати пробні покупки та надавати товару виняткову привабливість.

Саме через таку силу впливу на купівлю цінова сегментація ринку стає важливим пунктом будь-якого маркетингового плану. Але через свою важливість цей процес не став складним, навпаки, існує світова відпрацьована практика цінового сегментування ринку, яка допомагає визначити, описати та оцінити розмір кожного сегмента.

У загальносвітовій практиці виділяють 4 типи цінових сегментів:

- низько-ціновий сегмент або сегмент економ пропозицій (low-priced);
- середньо-ціновий сегмент (middle-priced);
- високо-ціновий сегмент (high-priced);
- преміум сегмент (luxury).

Низько-ціновий сегмент

Ціни на товари низько-цінового сегмента знаходяться на рівні нижче середньої роздрібної ціни по ринку. У сегменті представлені товари з базовими характеристиками та мінімальним функціоналом. Споживач не очікує від продуктів низько-цінового сегмента надвластивостей та високого результату, лише фактичне виконання заявлених обіцянок. Супер-властивості, що пропонуються товарами сегмента, часто сприймаються покупцем як обман.

Середньо-ціновий сегмент

Рівень цін товарів середньоцінового сегмента знаходиться на рівні середньої ціни по ринку. У такому сегменті вже можна побачити товари, що мають додатковий функціонал і властивості, в цікавій упаковці, з цікавими смаками, видами і т.д. Цей сегмент зазвичай має найвищий рівень диференціації продуктів. Споживач очікує від продукту середньоцінового сегмента точного виконання заявлених обіцянок.

Високо-ціновий сегмент

Рівень цін товарів високо-цінового сегмента вищий за середню ціну по ринку. У такому сегменті зазвичай представлені товари, які прагнуть зарекомендувати себе в образі "фахівців та експертів" ринку. Часто у високо-ціновому сегменті є спеціалізовані торгові марки з обмеженим асортиментом. Споживач очікує від продукту високо-цінового сегмента понад виконання заявлених обіцянок, високу ефективність, обслуговування та якість.

Цікаво те, що чутливість до ціни у цьому сегменті незначна. Основна рекламна підтримка сегмента спрямовано формування лояльності до товару, на рекомендації і побудові правильного іміджу продукту.

Преміум сегмент

Рівень цін товарів преміум або luxury сегмента значно вищий за середню ціну по ринку. У такому сегменті зазвичай представлені товари, здатні підкреслити високий соціальний статус та особливий імідж покупця. Споживач очікує від продукту преміум сегмента не лише високої якості та виконання всіх бажань, а й визнання індивідуального підходу. У преміум сегменті є бренди, які не можна зустріти в інших цінових сегментах ринку.

Чутливість до ціни у цьому сегменті відсутня. Основна рекламна підтримка сегмента спрямовано формування лояльності до товару, залучення покупця у продукт, створення «культу товару».

Відповідно да сегменту товару, необхідна різна рекламна політика.

ABC-аналіз ранжує категорії за різними параметрами. Таким чином можна ранжувати постачальників, запаси на складі, покупців і тривалі періоди продажів — усе це з достатньою статистикою. Результатом ABC-аналізу є групування об'єктів за їх впливом на загальний результат.

ABC-аналіз базується на принципі дисбалансу, під час якого будується діаграма залежності кумулятивного ефекту від кількості елементів. Такі графіки називають кривими Парето, кривими Лоренца або кривими ABC. На основі результатів аналізу категорійні пункти ранжуються та групуються відповідно до того, наскільки вони сприяють загальному ефекту. У логістиці ABC-аналіз часто використовують для відстеження поставок певних товарів і частоти звернень до тієї чи іншої позиції в асортименті, а також для ранжування клієнтів на основі того, скільки або скільки замовлень вони розміщують.

FMR (абревіатура від англ. Fastest Medium Rare – швидко, середньо, повільно) – аналіз товарного асортименту за частотою звернень/взяття.

В управлінні продажами попит або частота запитів на певні групи продуктів є важливим критерієм позиціонування груп продуктів у стратегіях маркетингового менеджменту. В управлінні запасами FMR застосовує визначення місць зберігання запасів. Так, найбільш часто запитувана («швидка») локація ближче до місця проведення.

За частотою звернень асортимент зазвичай розбивається на три групи:

- категорія F — товари, що найчастіше запитуються (80 % від загальної кількості);

- категорія M — категорія продуктів, що менш часто запитується (15 % від звернень);
- категорія R.

$$\sum_{j=1} P_j \quad (1.1)$$

где Pi - кількість відпуску і товару, - відпуск зі складу.

Групи також зазвичай визначають, використовуючи Закон Парето (поділ на групи: 80 %, 15 % та 5 %). $\sum_{j=1} P_j$

Аналіз XYZ – аналіз, де використовуються такі поняття, як коефіцієнт варіації - це відношення середньоквадратичного відхилення до середнього арифметичного вимірювань ресурсів.

Реальне значення коефіцієнта варіації для різних груп може відрізнятися з таких причин:

- сезонність продажів,
- тренд,
- акції,
- дефіцит
- тощо.

Є кілька різновидів XYZ-аналізу, наприклад, аналіз планових даних з фактичними, що дає більш точний % відхилення від прогнозу. Дуже часто XYZ-аналіз проводять спільно з ABC-аналізом дозволяючи виділяти більш точні групи щодо їх властивостей.

Коефіцієнт варіації визначається за формулою (1.2).

$$\bar{x} \quad \sqrt{} \quad n \quad n \quad (1.2)$$

де:

V - коефіцієнт варіації,

σ - середньоквадратичне відхилення,

$\bar{x}$ - Середньоарифметичне,

x_i - і-те значення статистичного ряду,

n - кількість значень у статистичному ряді.

RFM аналіз - спеціальна методика, яка допомагає сегментувати клієнтів за трьома ключовими параметрами:

Recency. Давність покупки з моменту останнього замовлення.

Frequency. Частота замовлень.

Monetary. Розмір чека (загальна вартість).

За допомогою зазначених параметрів можна розподілити клієнтів за конкретними групами. В результаті з'являються сегменти, з якими можна працювати різними способами. Насамперед сегментація допомагає персоналізувати комунікацію. Наприклад, клієнтам, які зробили першу покупку, можна запропонувати акцію на повторне замовлення, тоді як постійним користувачам можна надавати VIP-бонуси.

Крім того, можна визначити категорії для кожного атрибуту. Наприклад, Recency (давність) може бути розбита на три категорії: клієнти, які здійснювали покупки протягом останніх 90 днів; між 91 і 365 днями; і довше 365 днів. Такі категорії можуть бути похідні від правил ведення бізнесу або використання методів обміну даними, щоб знайти значущі перерви.

Після того, як для кожного з атрибутів визначені відповідні категорії, сегменти створюються з перетину значень. Якби для кожного атрибута було три категорії, то отримана матриця мала б двадцять сім можливих комбінацій (один з підходів використовує п'ять блоків для кожного атрибуту, що дає 125 сегментів). Компанії також можуть вирішити згортати певні підсегменти, якщо градації здаються занадто малими, щоб бути корисними. Отримані сегменти можуть бути впорядковані від найцінніших (найвища відвідуваність, частота та значення) до найменш цінних (найнижча кількість відвідувань, частота та значення) [1-6].

1.2 Сучасні рішення у сфері аналітичних програм

Для актуалізації ситуації в сфері веб-аналітики розглянемо сучасні популярні платформи та сервіси.

1. Convead

Convead – комплексний маркетинговий сервіс для інтернет-магазинів. Це одне з найкращих рішень на ринку, за допомогою якого можна збирати аналітику, заточену під e-commerce, збільшувати продаж та підвищувати лояльність клієнтів.

Основні можливості:

- збирання та угруповання даних про користувальницьку поведінку на всіх етапах вирви продажів – враховуються переходи, перегляди, замовлення, відмови та інші показники;
- сегментування аудиторії на підставі поведінкових патернів;
- аналіз ключових метрик за вибраними сегментами відвідувачів, джерелами трафіку, звітними періодами;
- створення персоналізованих кампаній на основі накопичених даних;
- A/B тестування гіпотез та оптимізація витрат за рахунок відмови від неефективних маркетингових інструментів;
- формування ємних наочних звітів із метриками, відстежуваними в реальному часі.

Плюси Convead:

- швидке встановлення без допомоги програміста;
- наочна та проста для розуміння аналітика ключових показників електронної комерції;
- оцінка ефективності рекламних кампаній із фінансової точки зору;
- відстеження статистики за заздалегідь встановленими критеріями;
- великий набір інструментів (віджети, тригерні та масові розсилки, push, онлайн-чат).

Мінуси:

-дуже висока ціна.

2. Roistat

Roistat – система наскрізної бізнес-аналітики, яка збирає інформацію з різноманітних рекламних майданчиків, сайту та CRM. У сервісі відстежується 66 важливих показників для e-commerce, включаючи конверсію, середній чек, виручку, прибуток, ROI, LTV, CPA/CPO та інше. За потреби можна настроювати власні метрики.

Основні можливості:

- оцінка динаміки ключових показників у багаторівневих зведених звітах з будь-яких зрізів;

- розумне керування рекламою: планування, аудит, контроль витрат, автоуправління ставками;

- розрахунок прибутковості різноманітних рекламних каналів;

- когортний аналіз довжини циклу та повторних продажів;

- аналіз трафіку сайту;

- трекінг заявок з онлайн та офлайн-джерел.

Плюси Roistat:

- просте підключення системи аналітики;

- групування даних із різних джерел у зручну єдину базу;

- зрозумілі звіти з візуалізацією, гнучким налаштуванням та можливістю збереження у форматі Excel;

- аналіз безлічі важливих бізнес-показників у динаміці.

Мінуси:

- досить складні налаштування та інтеграції;

- висока вартість користування.

3. Calltouch

Calltouch – система колтрекінгу та наскрізної аналітики з набором функціональних інструментів для автоматичного управління рекламою. Виймаючи показники з рекламних майданчиків, CRM, систем аналітики та сайту, сервіс об'єднує дані та виводить їх у зручних деталізованих звітах.

Основні можливості:

- відстеження даних на всіх каналах: бюджет, сесії, покази, ліди, кліки, оплати, угоди, виручка, ROI;

- формування настроюваних інтерактивних звітів, зрізів, порівняльних таблиць за різними періодами (враховується понад 30 метрик);

- аналіз ефективності реклами та коригування кампаній;

- візуалізація шляху клієнта з першого торкання;

- рекомендації щодо підвищення конверсії;

- відстеження дзвінків, контроль та планування роботи call-центру;

- встановлення віджету зворотного дзвінка на сайт;

- автоматичне керування контекстною рекламою.

- об'єднання всіх рекламних витрат в єдиному інтерфейсі;

Плюси:

- величезна кількість звітів – дашборди, статистика з трафіку, звіти по клієнтам, подіям, менеджерам, періодам, сайтам-донорам та інші;
- безліч повноцінних інтеграцій на один клік;
- безкоштовні інструменти для збирання лідів.

Мінуси:

- не підходить для невеликих рекламних бюджетів та стартапів, єфективно працює в розвинених системах бізнесу.

4. Google Analytics

Google Analytics – це сервіс безкоштовної аналітики відвідуваності ресурсу, розроблений пошуковою системою Google. Він надає докладну інформацію щодо відвідувачів сайту, у т. ч. операційну систему, провайдера, джерело трафіку та інші важливі параметри.

Основні можливості:

- доступ до багатьох аналітичних інструментів. Користувальницькі змінні, API-інструменти, зведення відвідуваності, візуалізація трафіку, підтримка доступу для співробітників тощо;
- аналіз змісту. Статистика сторінок ресурсу, відстеження подій, аудит швидкості завантаження. Різноманітність інструментів дозволяє отримати дані про найбільш популярні сторінки сайту;
- аналіз мобільних даних. Аналіз за програмами, статистика переходів з гаджетів, оцінка ефективності оголошень на мобільних пристроях;
- аналіз соціальної активності Детальна інформація про репоста матеріали, вплив соцмереж та переходи, комерційна вигода;
- аналіз конверсії. Детальна статистика для конверсій та цілей, оптимізація продажів, контроль ефективності РК, візуалізація переходів на ресурсі;
- аналіз реклами. Інструмент, який дозволяє аналізувати та порівнювати різні джерела реклами, підвищувати ефективність рекламних кампаній та інтеграції з AdWords.

Плюси:

- інструмент підтримує велику кількість мовних установок.
- широкий функціонал, що включає десятки фільтрів, приблизно 100 видів звітів, безліч різноманітних налаштувань;

- детальна статистика для всіх проектів за багатьма параметрами;
- безкоштовна версія.

Мінуси:

- безкоштовна версія обмежена;
- довгий час обробки актуалізації статистики (від 4 годин);
- потрібно повну інтеграцію магазина з сервісами гугл.

1.3 Постановка завдання дослідження

Відповідно, як бачимо, сучасні аналітичні системи здебільшого розраховані на великий бізнес, маючи широкий функціонал, можливість обробки великого масиву даних.

Проте такі системи можуть працювати зі значною затримкою в часі через свою масштабність.

Широкий функціонал і багата система параметрів, користувацькі звіти вимагають досить значного часу на налаштування та конфігурацію та можуть потребувати послуг окремого спеціаліста.

Зазвичай широкий функціонал сервісів компенсується високою ціною за користування та малий бізнес не може дозволити собі їх використання.

Відповідно, виникає необхідність формування веб-аналітики даних малого онлайн-бізнесу без великих затрат та витрачання часу на складну конфігурацію. Очікуваним результатом буде аналітичний звіт за бажаний період роботи магазину в певному стандартизованому форматі.

Відповідно до всього опрацьованого вище матеріалу, поставимо таке завдання:

Має бути створений веб-серверний додаток, який:

1. Має змогу прийняти масив даних онлайн-магазина про користувачів, продукти та покупки в одному з реалізованих підтримуємих додатком форматі.
2. Перетворити дані до стандартної моделі даних, до якої будуть застосовані аналітичні та статистичні алгоритми.

3. Описанти алгоритм розрахування необхідних метрик та моделей дата-майнингу. Проаналізувати дані в зазначеному періоді часу, розрахувавши важливі статистичні дані, такі як: кількість покупок, топ товарів за виручкою, найбільш активні користувачі, середній дохід за день, тенденція розвитку прибутку магазину.

4. Сформувати модель даних результатів аналізу та зберегти до бази даних.

5. Створити візуальний звіт, представивши проаналізовану інформацію у вигляді графіків, таблиць та числових значень.

РОЗДІЛ 2 ДАТА-МАЙНИНГ

2.1 Дата-майнинг: сутність і затосування

Навіщо потрібен Data Mining? В сучасному світі інформації, суспільство генерує її терабайтами кожної секунди. Це і дані про покупки в магазині, аренду домівки чи автомобіля, історія пошуку в інтернеті тощо. Компанії чи держава, що володіють інформацією, мають змогу забезпечити потреби людей якнайбільш повно, якщо віднайдуть певні закономірності в поведінці людей за допомогою даних. Результатами такого аналізу можуть бути рекомендації компліментарних продуктів користувачам магазину при оформленні замовлення, або зміна напрямів смуг на певних дорогах у певні часи дня для забезпечення стабільної логістичної ситуації в місті, або оптимізація поставок певних одиниць товару до складів магазину.

Data Mining має зв’язки з багатьма іншими дисциплінами та областями досліджень, як це показано на рис.2.1.

Рисунок 2.1 - Взаємозв’язок Data Mining з іншими галузями знань

Саме задачею аналізу та “видобування” корисної інформації з великого масиву наявних даних і займається Data Mining.

В загальноому випадку постановка завдання така:

1. Існує великий масив інформації;
2. Ця інформація має деякі приховані, корисні знання.
3. Постає завдання розробити методи виявлення прихованих знань у великому масиві вихідної «сирої»інформації.

2.2 Завдання методології Data Mining

Виділяють також декілька спеціалізованих завдань Data Mining. Наведемо такж приклади різних сфер інформаційного світу, що ставлять такі завдання.

1. Завдання класифікації – визначення категорії для кожного об'єкта дослідження. У сфері фінтеху таким завданням буде оцінка кредитоспроможності потенційних позичальників. Це допоможе знизити ризики втрати коштів під час роботи з некредитоспроможними клієнтами.

2. Завдання прогнозування, тобто виявлення нових можливих значень у певній числовій послідовності. У e-commerce таке завдання вирішується для попередньої установки цін залежно від сезонів та трендів. Завдяки цьому можна прогнозувати рівень продажів.

3. Завдання кластеризації (сегментації) - розбивка велкої множини об'єктів за якими характеристиками на групи. Так, наприклад, сегментація даних про покупців інтернет-магазину за віком, статтю чи перевагами допомагає формувати для кожної групи спеціальні пропозиції.

4. Завдання визначення взаємозв'язків - знаходження частоти наборів об'єктів, що зустрічаються, серед множини наборів. Цей спосіб допомагає, зокрема, визначити склад споживчого кошика та оптимізувати розміщення інформації про супутні товари в інтернет-магазині.

5. Завдання аналізу послідовностей - виявлення закономірностей у послідовностях подій. Цей аналіз можна застосовувати для відстеження сторінок, на яких найчастіше відвідувачі переривають перегляд сайту. Такий спосіб роботи з даними дозволяє усунути недоліки сайтів та підвищити його відвідуваність;

6. Завдання аналізу відхилень - визначення даних, що значно відрізняються від норми. Даний аналіз використовується у фінтеху для виявлення шахрайських операцій із банківськими картками. Він дає змогу забезпечити надійний захист клієнтів.

2.3 Знання Data Mining

Знання, виявлені у процесі Data Mining, кінцевий результат процесу аналізу, мають задовольняти певним вимогам. Розглянемо їх більш розгорнуто.

1. Знання повинні бути новими, раніше невідомими. Зусилля, витрачені на відкриття того, що користувачі вже знають, не винагороджуються. Тому цінні нові знання, які раніше були невідомі.
2. Знання повинні мати значення. Дослідження та аналіз мають знайти неочевидні, несподівані моделі даних, які становлять так звані приховані знання. Результати, які можна отримати більш простими засобами (наприклад, візуальним оглядом), не виправдовують використання потужних методів аналізу даних.
3. Знання повинні мати практичну цінність. Виявлені знання повинні бути застосовні, в тому числі до нових даних, з досить високою надійністю. Корисно те, що застосування цих знань може дати певну користь.
4. Знання мають бути зрозумілими для людини. Знайдені закономірності повинні бути логічно зрозумілими, інакше є ймовірність випадковості. Крім того, отримані знання повинні бути представлені у зрозумілій для людини формі.

Знання можуть описувати зв'язок між властивостями бізнес-об'єктів, розраховувати значення певних значень з урахуванням інших тощо. Нові знання повинні бути використані до нових об'єктів.

Практична корисність знань обумовлена можливістю їх використання у процесі підтримки прийняття управлінських рішень та вдосконаленні діяльності підприємства.

Доступність знань можна описати як представлення їх у формі, зрозумілій користувачам без спеціальної математичної підготовки. Наприклад, найлегшою для сприйняття є логічна структура «якщо, то». Якщо отримані знання непрозорі для користувача, повинні існувати методи постобробки, щоб перетворити їх у форму, яку можна інтерпретувати.

2.3 Методи дата-майнингу

Методи дата-майнингу - доволі вагома та велика тема.

Далі розглянемо деякі важливі.

Кластерний аналіз

Метою кластеризації є пошук існуючих структур.

Кластеризація є описовим процесом, він не робить жодних статистичних висновків, але пропонує можливість провести пошуковий аналіз і вивчити «структуру даних».

Саме поняття «кластер» неоднозначне: у кожного дослідження є свій «кластер». Поняття cluster перекладається як «скупчення», «скупчення». Кластер можна охарактеризувати як набір об’єктів зі спільними властивостями (рис. 2.2).

Рисунок 2.2 - Зображення кластерів

Існує дві функції, які можна назвати функціями кластера:

- внутрішня однорідність;
- зовнішня ізоляція.

Одна з проблем, яку задають аналітики при вирішенні багатьох завдань, полягає в тому, як організувати дані у візуальні структури, тобто розширену таксономію.

Кластеризація вперше була широко використана в таких наукових галузях, як біологія, антропологія та психологія. Протягом тривалого часу, через специфіку економічних даних і явищ, кластеризація рідко використовується для вирішення економічних завдань.

Кластери можуть бути непересічними, або ексклюзивними (non-overlapping, exclusive), і тими, що перетинаються (overlapping).

Байєсовські мережі

Байєсовські мережі — це структури графів, які представляють імовірнісні зв’язки між великою кількістю змінних і виконують імовірнісні висновки щодо цих змінних. «Наївна» (байєсовська) класифікація є досить прозорим і легким для розуміння методом класифікації. Його називають «наївним», тому що він базується на припущенні про незалежність ознак одна від одної (рис. 2.3).

Класифікаційні властивості:

1. Використання всіх змінних і визначення всіх залежностей між ними.

2. Існування двох припущень щодо змінної: Усі змінні однаково важливі; Усі змінні є статистично незалежними, тобто значення однієї змінної не пов’язане зі значенням іншої змінної.

Рисунок 2.3 - Байєсовська мережа

Штучні нейронні мережі

Розглянемо також нейронні мережі.

Штучні нейронні мережі можуть бути синхронними і асинхронними. У синхронній нейронній мережі тільки один нейрон змінює свій стан в кожен момент. В асинхронних нейронах стан всієї групи нейронів часто змінюється відразу по всьому шару. Можна виділити дві основні архітектури - багаторівневі мережі та повністю підключені мережі. Концепція рівнів є ключовою для багатошарових мереж (рис. 2.4). Шар — це один або кілька нейронів, вхідні дані яких забезпечують однаковий загальний сигнал. Ієрархічні нейронні мережі — це нейронні мережі, які поділяють нейрони на окремі групи (шари), завдяки чому інформація обробляється шар за шаром. У багатошаровій мережі нейрони рівня і отримують вхідний сигнал, перетворюють його та передають нейронам рівня (i+1) через точки розгалуження. Таким чином, аж до рівня k, він випромінює вихідні сигнали для інтерпретатора та користувача. Кількість нейронів у кожному шарі не має нічого спільного з кількістю нейронів в інших шарах і може бути довільною. Усередині рівня дані обробляються паралельно, тоді як у масштабі всієї мережі обробка відбувається послідовно — від одного рівня до іншого. Ієрархічні нейронні мережі включають, наприклад, багатошарові

персептрони, мережі радіальних базисних функцій, когнітивні, некогнітивні, мережі асоціативної пам’яті. Однак сигнали не завжди надходять до всіх нейронів шару. Наприклад, у CogniMachine кожен нейрон поточного шару отримує сигнали лише від найближчих нейронів попереднього шару.

У свою чергу ієрархічні мережі можуть бути одношаровими і багатошаровими.

Одношарова мережа – це мережа, що складається з одного шару.

Багаторівнева мережа — це мережа з кількома рівнями.

У багаторівневій мережі перший рівень називається вхідним, наступні — внутрішніми або прихованими, а останній — вихідним. Вхідний рівень організований вхідними нейронами (вхідними нейронами), які отримують дані та розподіляють їх на вхід нейронів у прихованих шарах мережі. Прихований нейрон — це нейрон, розташований у прихованому шарі нейронної мережі. Вихідний нейрон (outputneuron), включаючи вихідний рівень організованої мережі, дає результат нейронної мережі.

Рисунок 2.4 - структура штучної нейронної мережі

Навчання нейронних мереж

Перш ніж використовувати нейронну мережу, її необхідно навчити. Процес навчання нейронної мережі передбачає налаштування її внутрішніх параметрів під конкретне завдання. Алгоритми нейронних мереж ітераційні, з кроками, які називаються епохами або циклами. Епоха — це ітерація в процесі навчання, яка складається з показу всіх прикладів у навчальному наборі та, можливо, перевірки якості навчання на контрольному наборі. Процес навчання є зразком навчання. Навчальні зразки включають вхідні значення набору даних і відповідні їм вихідні значення. В результаті навчання нейронна мережа виявляє певні залежності вихідних полів від вхідних. Отже, перед нами постає питання - які поля введення (символи) нам потрібно використовувати. По-перше, вибір є евристичним, а потім кількість вхідних даних можна варіювати.

Аналітики повинні визначити кількість шарів у мережі та кількість нейронів у кожному шарі. Далі необхідно призначити такі ваги і значення зміщення, щоб мінімізувати помилки в прийнятті рішень. Ваги та зміщення автоматично регулюються, щоб мінімізувати різницю між бажаним сигналом і вихідним сигналом, відому як помилка навчання. Обчислює похибку

навчання побудованої нейронної мережі шляхом порівняння початкового та цільового (очікуваного) значень. З отриманих різниць формується функція похибок.

Функція помилок — це функція, яку нейронна мережа повинна мінімізувати під час контрольованого процесу навчання. Функція помилок дозволяє оцінити якість роботи нейронної мережі під час навчання. Наприклад, за допомогою суми квадратів помилок. Його здатність вирішувати завдання залежить від якості навчання нейронної мережі.

Інтелектуальний аналіз даних є мультидисциплінарним, оскільки він включає елементи чисельних методів, математичної статистики та теорії ймовірностей, теорії інформації та математичної логіки, штучного інтелекту та машинного навчання.

Завдання бізнес-аналізу формулюються по-різному, але рішення більшості завдань зводиться до того чи іншого завдання інтелектуального аналізу даних або їх комбінації. Наприклад, оцінка ризику є вирішенням проблеми регресії або класифікації, сегментація ринку є кластеризацією, а стимулювання попиту є правилами асоціації. Насправді завдання Data Mining — це елементи, які можна «збирати» для вирішення більшості реальних бізнес-завдань.

Останнім часом в інтелектуальному аналізі даних набули популярності такі методи: нейронні мережі, дерева рішень, алгоритми кластеризації (в тому числі масштабовані), алгоритми виявлення асоціативних зв'язків між подіями тощо.

2.4 Моделі у Data Mining

Нарешті, визначимо найбільш важливе та змістовне поняття в цьому розділі - модель.

Інтелектуальний аналіз даних – це процес виявлення корисної інформації у великих наборах даних. У інтелектуальному аналізі даних математичний аналіз використовується для визначення закономірностей і тенденцій у даних. Часто такі закономірності неможливо виявити традиційною візуалізацією даних через складність зв’язків або обсяг даних.

Ці закономірності та тенденції можна зібрати разом і визначити як моделі аналізу даних. Моделі аналізу даних можна застосовувати в конкретних сценаріях, а саме:

1. Прогнозування.
2. Розрахунок ризику
3. Підготовка пропозицій.
4. Аналіз послідовностей.
5. Синтез груп.

Розглянемо класифікацію моделей у дата-майнингу (рис. 2.5).

Рисунок 2.5 - Класифікація моделей у дата-майнингу

Існують два види моделей: передбачувальні та описові.

Передбачувальні (predictive) моделі.

Послідовний пошук: аналізуйте вибір клієнтів у процесі покупки та прогнозуйте наступну можливу подію.

Групування: розділіть клієнтів або події на кластери пов’язаних елементів, аналізуйте та прогнозуйте спільні характеристики.

Прогнозні моделі будуються на основі набору даних із відомими результатами. Вони використовуються для прогнозування результатів на основі інших наборів даних. При цьому, природно, модель має бути максимально точною, статистично значущою, правдоподібною тощо.

До них відносяться такі моделі:

- моделі класифікації;
- моделі послідовностей.

Описові моделі.

Описова модель фокусується на природі залежностей набору даних і їх взаємодії. Ключовим моментом цієї моделі є сприйняття людиною легкості та прозорості. Виявлена закономірність може бути специфічною особливістю даних конкретного дослідження, якої більше ніде немає, але вона, тим не менш, корисна, і тому її слід знати.

Існують такі види описових моделей:

- регресійні моделі – описують функціональні залежності між безперервними числовими параметрами та між категоріальними параметрами;
- моделі кластеризації – описують групи (кластери), на які можна розділити об’єкти аналізованих даних. Об’єкти (спостереження, події) групуються з урахуванням даних (атрибутів), які описують їх природу. Об'єкти в кластері повинні бути «схожі» один на одного і відрізнятися від об'єктів, що містяться в інших кластерах. Чим більше схожих об’єктів у кластері та чим більше відмінностей між кластерами, тим точнішою буде кластеризація;
- моделі винятків – описують аномалії в записах (наприклад, окремих пацієнтів), які певним чином відрізняються від основного набору записів (групи пацієнтів). Знання про винятки можна використовувати двома способами. Ці записи можуть відображати випадкові збої, наприклад помилки операторів, які вводять дані в комп’ютер. Типовий випадок: якщо оператор помиляється і ставить десяткову крапку не в тому місці, то така помилка відразу може призвести до різкого «сплеску» замовлення. Цей «шумний» випадковий компонент виправдано відхилено та виключено з подальшого вивчення, оскільки більшість методів інтелектуального аналізу даних, які обговорюватимуться в цій лекції, дуже чутливі до наявності «викидів», тобто різних точок зору, рідкісних, нетипових справа. З іншого боку, індивідуальні ідіосинкратичні записи можуть представляти незалежний дослідницький інтерес (оскільки вони можуть вказувати, наприклад, на якесь рідкісне, але важливе аномальне захворювання).
- підсумкові моделі — призначені для визначення обмежень на аналіз даних масиву. Наприклад, при вивченні вибірки даних пацієнтів з інфарктом міокарда віком до 30 років виявилося, що всі описані у вибірці пацієнти або викурюють більше 5 пачок сигарет на день, або важать не менше 95 кг. Такі обмеження важливі для розуміння даних масиву; справді, це нове знання, отримане за допомогою аналізу. Таким чином, це виявлення будь-якого факту, що всі або майже всі записи у досліджуваній вибірці даних є правильними, але досить рідкісними у всіх мислимих різновидах записів, які мають однаковий формат, наприклад, розподілені. Якщо порівняти дані всіх пацієнтів, то частка завзятих курців і пацієнтів з ожирінням дуже мала. Ми можемо сміливо сказати, що завдання неявної класифікації вирішено, хоча фактично задано лише один клас, представлений наявними даними;

- моделі асоціації – встановлюють закономірності між зв'язаними подіями. Наприклад, закономірность, щовизначає, що з події Y випливає подія Z. Такі правила - асоціативні.

2.5 Data Mining у сфері онлайн-торгівлі

В сфері онлайн-торгівлі аналітика даних займає дуже важливе місце.

Веб-аналітика – безперервний процес зі збирання та вимірювання даних, необхідних для оптимізації та покращення роботи. Сам собою інструмент не вирішує проблеми, а лише надає інформацію. Її потрібно правильно інтерпретувати: визначити недоліки та слабкі місця ресурсу, вивчити поведінку, потреби цільової аудиторії. На підставі результатів вносяться правки до стратегії та приймаються серйозні бізнес-рішення.

Ключові завдання веб-аналітики:

- оцінка якості трафіку;
- виявлення технічних помилок та інших недоліків на сайті;
- формування моделі поведінки та портрета цільового клієнта;
- пошук ефективних методів підвищення конверсії;
- визначення найвигідніших маркетингових каналів;
- зниження вартості залучення покупця;
- відстеження актуальних тенденцій ринку;
- збільшення прибутку з клієнта;
- порівняння показників ресурсу з лідерами у ніші.

Моніторинг дає зрозуміти, що зараз відбувається на сайті, чи окупаються вкладення у розробку, обслуговування, SEO-просування, підтримка репутації, різні види реклами. Без цього інструменту робота здійснюється наосліп. Підприємець не може побачити, від чого йде максимальна віддача, що потребує покращень, у якому напрямку краще рухатися. Тому налаштовувати аналітику потрібно ще з моменту запуску сайту.

Для оцінки розвитку бізнесу використовуються Key Performance Indicators.

Веб-аналітика дозволяє виявити та проаналізувати такі важливі показники як:

- відвідуваність сайту;
- перегляди сторінок з товарами;
- середній час присутності на сайті та середня кількість переглянутих сторінок;
- сторінки виходу з сайту;
- шляхи залучення користувачів сайту;
- конверсія;
- показник повернення покупців;
- дохід відкористувачів;
- кількість покинутих кошиків;
- тощо. [7-16]

РОЗДІЛ 3 ІНСТРУМЕНТИ РОЗРОБКИ

Опишемо основні інструменти розробки додатку.

3.1 Мова програмування Java

Java – об'єктно-орієнтована мову програмування із дотриманням типізації.

Рисунок 3.1 – Логотип Java

Творці Java реалізували принцип WORA: «пиши один раз, запускай скрізь». Це завдання вирішується завдяки компіляції написаного Java коду в байт-код.

Java після майже 30 років свого існування користується великим попитом та є досить популярною мовою програмування.

На Java розроблено та розробляють:

- Мобільні додатки на платформу Android;
- Додатки для інших носимих гаджетів, які мають відповідну підтримку Java;
- Великих серверних додатки, у тому числі банківські системи, системи обробки та аналізу даних тощо;
- Веб-додатки (із розвитком інших сучасних мов програмування та світових трендів на Java зараз пишеться виключно Backend-частина) [17, 18].

3.2 Програма для автоматичного складання проектів Maven

Рисунок 3.2 – Логотип Apache Maven

Maven забезпечує виконання важливих етапів створення додатку на мові програмування Java: компіляцію, створення jar, створення дистрибутива програми, генерації документації.

Для успішного збирання проекту Maven необхідно створити файл конфігурації у форматі xml, де містяться спеціальні інструкції.

Основними елементами конфігураційного файлу є:

- project (обов’язковий кореневий тег) - задає версію конфігураційного файла;
- groupId (обов’язковий) - задає ім’я групи проекта;
- artifactId (обов’язковий) - задає ім’я (або інший унікальний ідентифікатор проекта) проекта;
- version (обов’язковий) - задає версію проекта;
- packaging - визначає тип результату збірки (за замовчанням jar);
- dependencies - визначає всі залежності (бібліотеки) проекту;
- build - зберігає інформацію про етап збирання проекту;
- repositories - зберігає інформацію про всі зовнішні підключені ресурси бібліотек.

Основні переваги Maven:

- Незалежність від OS. Збірка проекту відбувається в будь-якій операційній системі. Файл проекту один і той же.
- Yправління залежностями. Мавен дозволяє управляти такими складними залежностями. Що дозволяє вирішувати конфлікти версій і в разі потреби легко переходити на нові версії бібліотек.
- Можлива збірка з командного рядка. Проста інтеграція з Continuous Integration.
- Гарна інтеграція з середовищами розробки. Приклад - Intellij Idea.
- Зручний спосіб конфігурації.
- Декларативний опис проекту [19].

3.3 Рушій обробки даних Apache Spark

Apache Spark — це високопродуктивний механізм для обробки даних, що зберігаються в кластерах Hadoop (рис. 3.3). Порівняно з механізмом MapReduce, доступним у Hadoop, Spark забезпечує в 100 разів більшу продуктивність під час обробки даних у пам’яті та в 10 разів більшу продуктивність під час розміщення даних на диску.

Рисунок 3.3 – Логотип Apache Spark

Spark можна використовувати як для типових сценаріїв обробки даних, подібних до MapReduce, так і для реалізації конкретних методів, таких як потокова передача, SQL, інтерактивні та аналітичні запити, вирішення проблем машинного навчання та робота з графіками. Програми обробки даних можуть бути написані на Scala, Java, Python і R.

Драйвер Spark виконує функції:

- Зберігати та обробляти інформацію про стан програми
- Відповідати на запити програм користувача
- Проаналізувати та розподілити завдання.

Робота кластера лише імітується. . Це корисно для розробки та тестування. Цей режим зазвичай використовується для розробки, коли розподілене сховище не потрібне та використовується локальна файлова система.

Основною структурою даних, яку використовує Spark, є DataFrame. Він містить таблицю, що складається з рядків і стовпців. Список стовпців і їх типів називається схемою (можна переглянути, викликавши метод df.printSchema()).

DataFrames у Spark є незмінними, але ви можете застосувати до них перетворення для створення нових DataFrames.

Наприклад:

evenRows = myData.where("number % 2 = 0") // утворюємо DataFrame evenRows з датафрейму myData. [20]

Перетворення ліниві та не виконуватимуться, вони лише додаватимуться до плану обчислень, поки користувач не попросить щось зробити. Операція може виводити дані на консоль, файл або базу даних або

збирати їх в об’єктах мовою, якою був написаний запит. Прикладом дії є evenRows.count() — підраховує кількість рядків у DataFrame.

3.4 Додаткові Java-бібліотеки

Для ефективного та швидкого написання коду будемо використовувати також декілька додаткових бібліотек.

Lombok - це бібліотека, за допомогою якої можна скоротити кількість шаблонного коду, який потрібно писати на Java. Дозволяє автоматично створювати сеттери, геттери, конструктори для об’єктів за допомогою декількох анотацій.

Apache Commons - бібліотека з величезною кількістю допоміжних методів для маніпулювання колекціями, операціями зі строками та об’єктами тощо.

РОЗДІЛ 4 РОЗРОБКА АНАЛІТИЧНОГО ДОДАТКУ

Безпосередньо перед розробкою аналітичного додатку розглянемо важливі сучасну концепцію, що будуть необхідні для його побудови правильної архітектури.

4.1 Сучасні концепція розробки: мікросервісний підхід до побудови додатку

В сучасній сфері розробки програмного забезпечення існують основні принципи побудови архітектури додатку - це моноліт та мікросервіси.

Монолітна архітектура — це традиційна модель програмного забезпечення, яка є єдиним модулем, що працює автономно і незалежно від інших додатків (рис. 4.1).

Монолітна архітектура — це окрема велика обчислювальна мережа з єдиною базою коду, де об'єднані всі бізнес-завдання. Щоб внести зміни в таку програму, необхідно оновити весь стек через базу коду, а також створити та розгорнути оновлену версію інтерфейсу, що знаходиться на стороні служби. Це обмежує роботу з оновленнями та потребує багато часу.

Моноліти зручно використовувати на початкових етапах проектів, щоб полегшити розгортання та не витрачати надто багато розумових зусиль при керуванні кодом. Це дозволяє одразу випускати все, що є в монолітному додатку.

Рисунок 4.1 - Монолітна архітектура додатку

Проте постійно зростаючі вимоги до сучасних веб-додатків, такі як здатність надавати програмні інтерфейси, інтеграція з іншими веб-додатками, обробляти великі обсяги запитів, масштабованість і забезпечувати необхідну швидкість доступу до інформації, призвели до корпоративного монолітного

веб-програми. додатки Незручно розробляти, важко тестувати та виходити в онлайн, а час затримки великий.

Мікросервісна архітектура – це метод створення розподілених програм у вигляді набору незалежно розроблюваних і розгортаються невеликих сервісів, що запускаються як один або кілька ізольованих процесів (рис. 4.2).

Рисунок 4.2 - Мікросервісна архітектура додатку

Основна мета такого поділу – можливість зміни окремо взятого мікросервісу, не змінюючи при цьому пов'язаних із ним компонентів. Бізнес-логіка додатку розбивається на окремі частини, кожна з яких є невеликою програма, що виконує одну бізнес-завдання. Число таких додатків нічим не обмежене і між собою вони спілкуються, використовуючи API, побудоване, наприклад, на основі HTTP протоколу або за допогою асинхроного потоку подій.

Такий підхід має цілу низку переваг:

- Гнучке масштабування. Коли мікросервіс досягає граничного навантаження, можна швидко виконати розгортання нових екземплярів цієї служби у супутньому кластері та знизити навантаження. Тепер ми працюємо з кількома копіями та без збереження стану, а клієнти рівномірно розподілені. З таким підходом можна підтримувати екземпляри значно більшого розміру.
- Безперервне розгортання. Реалізація регулярних та прискорених циклів релізу. Оновлення можна робити двічі-тричі на день.
- Легкість обслуговування та тестування. Команди можуть експериментувати з новими функціями та повертатися до попередньої версії, якщо щось не працює. Це спрощує оновлення коду та прискорює випуск нових функцій на ринок. Крім того, в окремих службах легко знаходити та виправляти помилки та баги.
- Незалежне розгортання. Мікросервіси є окремими модулями, тому з ними можна легко і швидко виконувати незалежне розгортання окремих функцій.
- Гнучкість технологій. Y разі використання архітектури мікросервісів команди можуть вибирати інструменти з урахуванням своїх переваг.

- Висока надійність. Розгортаючи зміни для конкретної служби, можна не боятися, що програма вийде повністю з ладу. [21-26]

4.2 Структура додатку

На початковому етапі розробки додатку дуже важливо розробити додаток на структурні елементи та визначити зв’язки між ними (рис. 4.3).

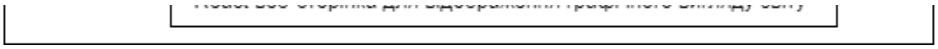


Рисунок 4.3 - Структурна схема додатку

Опис структури.

Для проведення аналізу даних онлайн-магазину необхідний доступ до них. Доступ може реалізовуватися декількома способами:

1. Надання доступу до API магазину.
2. Безпосередній доступ до даних в базі даних.
3. Надання файлів бекапу бази даних чи у вигляді файлів-таблиць.

Зазначимо, що перший спосіб є найбільш надійним та безпечним для власника магазина, проте може бути недоступний у разі, якщо магазин не був зконструйований на базі платформи онлайн-магазинів, що підтримує функції управління даними (наприклад, Shopify), та (якщо магазин є унікальним проектом) не має відповідних кінцвих точок.

Доступ до бази даних зазвичай виконується так:

Створюється окрема схема (або навіть база даних) з materialized views (також можуть використовуватися звичайні таблиці), куди копіюються дані з основних таблиць з необхідними даними. Створюється користувач бази даних з правами тільки на читання та його дані надаються групі аналітиків. Зручність такого способу для останніх очевидна - можна запросити створення таблиць в певному, зручному для аналізу форматі.

Останній спосіб є найбільш небезпечним, довгим та небажаним.

На фінальній стадії розробки аналітичний додаток буде мати декілька інтеграцій з різними API: завантаження даних з Shopify, Excel таблиць, та зовнішньої бази даних Postgres.

Відповідно, модуль синхронізації даних буде мати певну абстрактну процедуру завантаження даних для аналізу та перетворення їх до зручного для аналізу вигляду (рис. 4.4).

Рисунок 4.4 - лістинг коду абстракції для завантаження даних зі стороннього API

Основне місце в наведеній абстракції будуть займати реалізації Abstract Connection - саме цей класс буде займатись створенням з’єднання зі стороннім API та скачуванням даних.

Місцем для зберігання завантажених даних була обрана база даних Postgres - швидка реляційна база.

Після завантаження даних до бази даних проекту починається етап аналізу даних.

Для цього запускається Apache Spark master та Apache Spark worker.

В додатку, використовуючи Java-бібліотеку для Apache Spark, підключаємося до запущеного Spark та створюємо робочу сесію.

Наступним кроком є запуск завдання в межах робочої Spark сесії. Використовуючи Spark Postgres Driver, під’єднуємо активне Spark завдання до бази даних проекту.

Виконуємо завантаження та необхідні агрегації над даними.

Приводимо отримані результати до моделі аналітичного звіту та зберігаємо її до бази даних Postgres.

Останнім шагом є візуальна презентація результатів аналізу - отриманий звіт можна скачати з бази даних та переглянути, наприклад, в Excel.

4.3 Опис моделей даних для аналізу

Для проведення аналізу даних роботи онлайн-магазину, необхідно мати дані про користувачів магазину, продукти та покупки в магазині.

Проте на етапі синхронізації даних з зовнішнього API необхідно привести дані до єдиного формату, з яким можна просто та ефективно працювати.

Дані будемо зберігати до бази даних Postgres, тому створимо декілька пов’язаних таблиць для всіх типів даних.

Зобразимо схему бази даних (рис. 4.5).

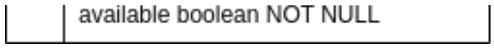


Рисунок 4.5 - Схема бази даних

Опишемо моделі даних для аналізу.

Модель користувача.

Серед даних користувача для аналітики нам потрібний унікальний ідентифікатор, усі інші дані будуть використані лише для візуального звіту.

Властивості моделі:

id - унікальний ідентифікатор користувача магазину. Строковий тип.

firstName - ім’я користувача. Строковий тип.

lastName - прізвище користувача. Строковий тип.

email - електронна пошта користувача. Строковий тип.

Лістинг коду моделі на Java.

Анотація @Data генерує сетери, гетери, перевизначає методи toString() та equals().

```
@Data
class Customer {
   private String id;
   private String firstName;
   private String lastName;
   private String email;
}
```

Модель продукта.

Серед даних продукта важливими властивостями для аналітики будуть унікальний ідентифікатор та поле наявності продукта, усі інші дані будуть використані лише для візуального звіту.

Властивості моделі:
id - унікальний ідентифікатор продукта. Строковий тип.
title - назва продукта. Строковий тип.
shopUrl - посилання на продукт. Строковий тип.

Лістинг коду моделі на Java.

```
@Data
class Product {
   private String id;
   private String title;
   private String shopUrl;
}
```

Модель покупки в магазині.
Найважливіша модель даних для аналізу.

Цікавим моментом є те, як в базі даних можна представити покупку декількох продуктів. Для кожного купленого продукту створюється окремий запис в базі даних та заносяться дані про нього: хто купив, який саме продукт, кількість одиниць, вартість одиниці продукції. Для того, щоб все ж таки зберегти зв’язок з продуктів, що входили до однієї покупки, до властивостей моделі додається ще одна, яка є унікільним ідентификітором покупки. В той же час запис містить також і власний унікальний ідентифікатор.

Властивості моделі:

purchaseItemId - унікальний ідентифікатор позиції покупки в магазині. Строковий тип.

purchaseId - унікальний ідентифікатор покупки в магазині. Строковий тип.

customerId - унікальний ідентифікатор користувача магазина, який здійснив покупку. Строковий тип.

productId - унікальний ідентифікатор продукта магазина. Строковий тип.

quantity - кількість куплених одиниць продукта. Цілочисельний тип.

itemPrice - вартість однієї одиниці продукта. Числовий тип з плаваючою комою.

purchasedTimestamp - дата покупки продукта. Часовий тип.

Лістинг коду моделі на Java.

```
@Data
class Purchase {
    private String purchaseItemId;
    private String purchaseId;
    private String customerId;
    private String productId;
    private Long quantity;
    private Double itemPrice;
    private LocalDate purchasedTimestamp;
}
```

4.4 Опис розранку метрик та побудови моделей дата-майнингу

В аналітичному звіті програми буде наведена інформація, отримана за допомогою моделей дата-майнингу, а також багато важливих корисних метрик про успішність онлайн-магазину.

4.4.1 Опис розрахунку метрик для онлайн-магазину

Аналіз даних зазвичай передбачає, що буде проаналізовано частину даних за деякий чітко визначений період.

В розробляємому проекті часові межі можна буде регулювати дуже точно - шляхом їх безпосереднього визначення та передачі до аналітичного модулю.

Тепер перерахуємо всі метрики, що будуть розраховані та опишемо алгоритм їх визначення.

1. Кількість унікальних покупців за весь час. Для розрахунку треба порахувати кількість записів у таблиці customer.
2. Кількість унікальних активних користувачів за період, що аналізується. Для розрахунку треба порахувати кількість користувачів, що зробили покупку у магазині за даний період.
3. Кількість унікальних найменувань товарів у магазині. Для розрахунку треба порахувати кількість записів у таблиці product.
4. Кількість унікальних найменувань товарів, що були придбані за період, що аналізується. Для розрахунку треба порахувати кількість товарів, що були куплені у магазині за даний період.
5. Загальна кількість одиниць купленого товару. Розраховується як сума кількості одиниць куплених товарів усіх найменувань.
6. Кількість успішних продажів за увесь час. Для розрахунку треба порахувати кількість order_id в таблиці purchase_item (як було зазначено вище, це значення є зв’язуючим для продуктів в одному продажі).

7. Кількість успішних продажів за даний період. Для розрахунку треба порахувати кількість order_id в таблиці purchase_item за визначений період.

8. Загальний дохід за даний період від продажів. Для розрахунку необхідно скласти ціну всіх куплених товарів за визначений період.

9. Середнє значення доходу від кожного продажу. Розраховується як загальний дохід за період поділений на кількість продажів.

10. Середня кількість унікальних найменувань товарів у кожному продажу. Розраховується як середнє арифметичне від кількості унікальних найменувань товарів у прожажах за даний період. В майбутньому може використовуватися для розширеного аналізу корзин покупців.

4.4.2 Опис моделей дата-майнингу

Тепер опишемо моделі, які допоможуть віднайти приховані знання в масивах даних про клієнтів, продукти та продажі магазину.

В попередньому розділі вже розглядався ABC-аналіз, що дозволяє класифікувати товари фірми за ступенем їх важливості.

Створимо модель кластеризації для такого аналізу. Для цього опишемо асоціативні правила, за якими будемо формувати кластери (сегменти) товарної продукції. Застосуємо правило Парето з роподіленням 80/15/5.

Правила:

1. Продукт належить до сегменту A, якщо входить у множину продуктів, що приносять найбільше доходу та формують 80% доходу від загального.

2. Продукт належить до сегменту B, якщо не ввійшов до сегменту A та входить у множину продуктів, що формують 15% доходу від загального.

3. Продукт належить до сегменту C, якщо не ввійшов до сегменту A та B та входить у множину продуктів, що формують залишок доходу - 5% від загального.

Для аналізу даних покупців проведемо RFM-аналіз. Він також вже згадувався у попередньому розділі та має цілиї три ключових параметри:

- Recency. Давність останньої покупки.
- Frequency. Частота замовлень.
- Monetary. Розмір чеку.

Кожний атрибут розіб’ємо на три категорії:

Recency: поділимо термін, що аналізується, на 3 рівні частини, створивши три категорії покупців - ті, що купували товари недавно, певний час тому, та давно.

Frequency: для того, щоб поділити на категорії цей параметр, візьмемо максимальне значення частоти замовлень користувача та поділимо його на 3 частини.

Monetary: використаємо той самий алгоритм, що і для frequency.

Відповідно, отримаємо цілих 27 сегментів, за якими можна досліджувати дані про покупців та приймати відповідні до кожного сегменту бізнес-рішення.

Створимо також регресійну модель для прогнозування величини продаж на кожен день наступного місяця, базуючись на даних визначеного періода.

Для цього:

1. Визначимо величину продаж за кожен день у даному періоді.
2. Застосуємо алгоритм лінійної регресії, аби сформувати найбільш пасуючу до даних лінію та отримати очікувані значення на майбутній період.

4.4.3 Опис моделі аналітичного звіту

Базуючись на описаних метриках та моделях дата-майнингу, опишемо модель аналітичного звіту в коді.

```
public class AnalyticsReport {
   private Long customerCount;
   private Long activeCustomerCount;
   private Long uniqueProductCount;
   private Long uniqueBoughtProductCount;
```

```
    private Long boughtProductItemCount;
    private Long purchasesCount;
    private Long uniquePurchasesInSpecifiedPeriod;
    private Double totalValue;
    private Double averagePurchaseValue;
    private Double averagePurchaseUniqueItemsCount;
    private List<Customer> customersWithStats;
    private List<Product> productsWithStats;
    private List<ChartPoint> revenueValueChartByDay;
    public static class Customer {
        private String id;
        private String firstName;
        private String lastName;
        private String email;
        private LocalDateTime firstPurchaseTimestamp;
        private LocalDateTime lastPurchaseTimestamp;
        private Long purchasesCount;
        private Double totalValue;
        private String recencySegment;
        private String monetarySegment;
        private String frequencySegment;
    }
    public static class Product {
        private String id;
        private String title;
        private String shopUrl;
        private String segment;
        private BigDecimal totalRevenue;
    }
    public static class ChartPoint {
        private LocalDate date;
        private Double value;
        private Boolean forecast;
    }
}
```

4.5 Опис агрегацій, відтворених у Spark

Опишемо агрегації в Spark.

Дані зберігаються в базі даних Postgres, до якої ми можемо під’єднатися за допомогою Apache Spark та провести необхідні нам розрахунки.

Оскільки необхідно враховувати надані часові межі для аналітичного звіту, метод для формування звіту, в метод будемо передавати інформацію про магазин, для якого необхідно створити аналітичний звіт та часові межі, в яких необхідно аналізувати наявні дані.

Лістинг класу, що буде передаватися як аргумент до методу створення аналітичного звіту наведено нижче.

```
public class AnalyticsTaskRequest {
    private final Long shopId;
    private final LocalDateTime periodStart;
    private final LocalDateTime periodEnd;
}
```

Для зручного написання коду, винесемо назви полей в базі даних в константи та будемо оперувати ними. Це виключить можливість помилитися при написанні назви поля та зробить процес написання коду більш гнучким.

```
@UtilityClass
public class DataModelFields {
    public static final String CUSTOMER_ID_FIELD = "customer_id";
    public static final String CUSTOMER_FIRST_NAME_FIELD = "customer_first_name";
    public static final String CUSTOMER_LAST_NAME_FIELD = "customer_last_name";
    public static final String CUSTOMER_EMAIL_FIELD = "customer_email";

    public static final String PRODUCT_ID_FIELD = "product_id";
    public static final String PRODUCT_TITLE_FIELD = "product_title";
    public static final String PRODUCT_SHOP_URL_FIELD = "product_shop_url";
```

```
    public static final String PURCHASE_ITEM_ID_FIELD = "purchase_item_id";
    public static final String PURCHASE_ID_FIELD = "purchase_id";
    public static final String PURCHASE_CUSTOMER_ID_FIELD = "purchase_customer_id";
    public static final String PURCHASE_PRODUCT_ID_FIELD = "purchase_product_id";
    public static final String PURCHASE_ITEM_QUANTITY_FIELD = "purchase_item_quantity";
    public static final String PURCHASE_ITEM_PRICE_FIELD = "purchase_item_price";
    public static final String PURCHASE_TIMESTAMP_FIELD = "purchase_purchased_timestamp";
}
```

Для початку, створимо dataset для кожного типу даних, які ми маємо в наявності, завантаживши дані з бази даних Postgres.

```
Dataset<Row> customerData = sparkService.loadCustomerData();
Dataset<Row> productData = sparkService.loadProductData();
Dataset<Row> purchaseData = sparkService.loadPurchaseData();
```

Розрахуємо кількість унікальних записів кожного типу. Врахуємо, що кількість покупок важливо рахувати за полем PURCHASE_ID_FIELD, оскільки в ньому міститься спільний id для всіх позицій в покупці і є унікальним для кожної окремої покупки.

```
long uniqueCustomerCount = customerData.count();
long uniqueProductCount = productData.count();
long uniquePurchasesCount = purchaseData
        .select(countDistinct(PURCHASE_ID_FIELD))
        .collectAsList().get(0).getLong(0);
```

Для подальших розрахунків створимо спільний dataset для всіх типів даних в рамках наданого часового інтервалу. filter використовується для фільтрації покупок, що виходять за межі визначеного інтервалу. Наступним кроком виконується операція приєднання даних користувачів та продуктів до відфільтрованих даних про покупки. Останною операцією withColumn додається нова колонка PURCHASE_LINE_VALUE_FIELD, в яку записується загальна вартість однієї позиції покупки за формулою (3.1).

item_total = item_price * item_quantity (3.1)

```
Dataset<Row> allJoinedData = purchaseData
```

```
        .filter(col(PURCHASE_TIMESTAMP_FIELD).$less$eq(upperTimeBound)
                .and(col(PURCHASE_TIMESTAMP_FIELD)
                        .$greater$eq(lowerTimeBound)))
        .join(customerData, customerData.col(CUSTOMER_ID_FIELD)
                .equalTo(purchaseData.col(PURCHASE_CUSTOMER_ID_FIELD)))
        .join(productData, productData.col(PRODUCT_ID_FIELD)
                .equalTo(purchaseData.col(PURCHASE_PRODUCT_ID_FIELD)))
        .withColumn(PURCHASE_LINE_VALUE_FIELD,
                col(PURCHASE_ITEM_PRICE_FIELD)
                .multiply(col(PURCHASE_ITEM_QUANTITY_FIELD)));
```

Отримавши dataset в рамках часових меж, можемо розрахувати одразу декілька важливих метрик для нашого звіту.

Кількість користувачів, що зробили покупку за визначений період.

```
long activeCustomersCount = allJoinedData
    .select(countDistinct(PURCHASE_CUSTOMER_ID_FIELD))
            .collectAsList().get(0).getLong(0);
```

Кількість унікальних продуктів, що були куплені за визначений період.

```
long uniqueBoughtProductCount = allJoinedData
    .select(countDistinct(PURCHASE_PRODUCT_ID_FIELD))
            .collectAsList().get(0).getLong(0);
```

Кількість покупок, що були здійснені за визначений період.

```
long uniquePurchasesInSpecifiedPeriod = allJoinedData
    .select(countDistinct(PURCHASE_ID_FIELD))
            .collectAsList().get(0).getLong(0);
```

Кількість одиниць товару, що була куплена за визначений період.

```
long boughtProductItemCount = allJoinedData
    .select(sum(PURCHASE_ITEM_QUANTITY_FIELD))
            .collectAsList().get(0).getLong(0);
```

Величина доходу від продаж за визначений період.

```
double totalValue = allJoinedData
    .select(sum(PURCHASE_LINE_VALUE_FIELD))
            .collectAsList().get(0).getDouble(0);
```

Середня величина вартості однієї покупки за визначений період.

```
double averagePurchaseValue = allJoinedData
   .select(avg(PURCHASE_LINE_VALUE_FIELD))
            .collectAsList().get(0).getDouble(0);
```

Середня величина кількості різних найменувань товарів в одній покупці за визначений період.

```
double averagePurchaseItemsCount = allJoinedData
  .select(avg(PURCHASE_ITEM_QUANTITY_FIELD))
            .collectAsList().get(0).getDouble(0);
```

Перейдемо до реалізації моделей дата-майнингу.

Для початку, реалізуємо модель кластеризації для ABC-аналізу товарної продукції магазину.

Для цього:

1. Розрахуємо сумарний дохід, який приніс кожний продукт окремо.
2. Отсортуємо продукти за величиною цього доходу, починаючи з найбільшого.
3. Для кожного продукту в списку просумуємо величини доходів від усіх попередніх та поточного продуктів у списку.
4. Для усіх продуктів, для яких значення менше чи дорівнює 80% від загального доходу визначаємо сегмент A. Для усіх, де менше 95% - сегмент B, для останніх - C.

Лістинг.

```
Dataset<Row> productWithStatsDataset =
        allJoinedData.groupBy(col(PURCHASE_ITEM_PRODUCT_ID))
           .agg(
              first(col(PRODUCT_TITLE_FIELD)).as(PRODUCT_TITLE_FIELD),
             first(col(PRODUCT_URL_FIELD)).as(PRODUCT_URL_FIELD),
                // new fields for analytics
```

```
            sum(col(PURCHASE_ITEM_QUANTITY))
                .as(PRODUCT_TOTAL_ITEMS_BOUGHT),
            sum(col(PURCHASE_LINE_VALUE))
                .as(PRODUCT_TOTAL_VALUE)
        )
        .withColumn(PREVIOUS_TOTALS_SUM,
            sum(PRODUCT_TOTAL_VALUE)
                .over(Window.orderBy(col(PRODUCT_TOTAL_VALUE).desc())
                    .rowsBetween(Window.unboundedPreceding(), Window.currentRow())))
        .withColumn(PRODUCT_VALUE_RANK,
            col(PREVIOUS_TOTALS_SUM).divide(totalValue))
        .withColumn(PRODUCT_SEGMENT,
            when(col(PRODUCT_VALUE_RANK).$less$eq(0.8), "A")
                .otherwise(when(col(PRODUCT_VALUE_RANK).$less$eq(0.95), "B")
                    .otherwise("C"))
        )
        .sort(col(PRODUCT_TOTAL_VALUE).desc());
```

Заповнюємо об’єкти у Java даними зі Spark.

```
List<AnalyticsReport.Product> productsWithStats = productWithStatsDataset
    .collectAsList()
    .stream()
    .map(this::mapRowToProduct)
    .collect(Collectors.toList());
```

Для перетворення запису зі Spark використовуємо метод mapRowToProduct().

```
private AnalyticsReport.Product mapRowToProduct(Row row) {
    return AnalyticsReport.Product.builder()
        .id(row.getString(0))
        .title(row.getString(1))
        .shopUrl(row.getString(2))
        .totalRevenue(row.getDecimal(4))
        .segment(row.getString(7))
```

```
            .build();
    }
```

Реалізуємо модель для RFM-аналізу.

Для початку підготуємо усі необхідні дані для кластерного аналізу:

1. Усі дані про користувачів.
2. Дохід від кожного користувача (для Monetary сегментів)
3. Дату останньої покупки для кожного користувача (для Recency сегментів)
4. Частоту покупок за аналізуємий період за формулою = кількість покупок / (дата останньої покупки - дата першої) (для Frequency сегментів)

```
Dataset<Row> customersWithPurchasesInfo =
allJoinedData.groupBy(col(CUSTOMER_ID_FIELD))
            .agg(
                first(col(CUSTOMER_FIRST_NAME_FIELD))
                    .as(CUSTOMER_FIRST_NAME_FIELD),
                first(col(CUSTOMER_LAST_NAME_FIELD))
                    .as(CUSTOMER_LAST_NAME_FIELD),
                first(col(CUSTOMER_EMAIL_FIELD))
                    .as(CUSTOMER_EMAIL_FIELD),

                // new field for analytics
                countDistinct(PURCHASE_ITEM_ORDER_ID)
                    .as(CUSTOMER_PURCHASES_COUNT),
                sum(col(PURCHASE_LINE_VALUE))
                    .as(CUSTOMER_TOTAL_VALUE_FIELD),
                max(col(PURCHASE_ITEM_PURCHASE_TIMESTAMP))
                    .as(LAST_CUSTOMER_PURCHASE_TIMESTAMP),
                min(col(PURCHASE_ITEM_PURCHASE_TIMESTAMP))
                    .as(FIRST_CUSTOMER_PURCHASE_TIMESTAMP))
.withColumn(PURCHASE_FREQUENCY,
                col(CUSTOMER_PURCHASES_COUNT)
    .divide(unix_timestamp(lit(Timestamp.valueOf(request.getPeriodEnd())))
      .minus(unix_timestamp(col(FIRST_CUSTOMER_PURCHASE_TIMESTAMP)))))
.withColumn(LAST_PURCHASE_RECENCY, percent_rank()
```

```
        .over(Window.orderBy(LAST_CUSTOMER_PURCHASE_TIMESTAMP)))
.withColumn(PURCHASE_FREQUENCY_RANK, percent_rank()
        .over(Window.orderBy(PURCHASE_FREQUENCY)))
.withColumn(PURCHASE_MONETARY_RANK, percent_rank()
                .over(Window.orderBy(CUSTOMER_TOTAL_VALUE_FIELD)))
        .orderBy(col(CUSTOMER_TOTAL_VALUE_FIELD).desc());
```

Відповідно до отриманих показників визначимо сегмент кожного користувача за кожним параметром.

```
    customersWithPurchasesInfo = customersWithPurchasesInfo
        .withColumn(PURCHASE_RECENCY_SEGMENT,
            when(col(LAST_PURCHASE_RECENCY).$greater$eq(0.67), "A")
              .otherwise(when(col(LAST_PURCHASE_RECENCY).$greater$eq(0.33), "B")
                    .otherwise("C")))
        .withColumn(PURCHASE_FREQUENCY_SEGMENT,
            when(col(PURCHASE_FREQUENCY_RANK).$greater$eq(0.67), "A")
                .otherwise(when(col(PURCHASE_FREQUENCY_RANK)
                        .$greater$eq(0.33), "B")
                    .otherwise("C")))
        .withColumn(PURCHASE_MONETARY_SEGMENT,
            when(col(PURCHASE_MONETARY_RANK).$greater$eq(0.67), "A")
                .otherwise(when(col(PURCHASE_MONETARY_RANK)
                        .$greater$eq(0.33), "B")
                    .otherwise("C")));
```

Заповнюємо об’єкти у Java даними зі Spark.

```
    List<AnalyticsReport.Customer> customersWithStats = customersWithPurchasesInfo
            .collectAsList()
            .stream()
            .map(this::mapRowToCustomer)
            .collect(Collectors.toList());
```

Для перетворення запису зі Spark використовуємо метод mapRowToCustomer().

```
private AnalyticsReport.Customer mapRowToCustomer(Row row) {
    return AnalyticsReport.Customer.builder()
            .id(row.getString(0))
            .firstName(row.getString(1))
            .lastName(row.getString(2))
            .email(row.getString(3))
            .purchasesCount(row.getLong(4))
            .totalValue(getDouble(row, 5))
            .lastPurchaseTimestamp(row.getTimestamp(6).toLocalDateTime())
            .firstPurchaseTimestamp(row.getTimestamp(7).toLocalDateTime())
            .recencySegment(row.getString(12))
            .frequencySegment(row.getString(13))
            .monetarySegment(row.getString(14))
            .build();
}
```

Нарешті, створимо регрессивну модель: виокримемо дані для графіку доходу з грануляцією даних в один день. Для цього додамо до нашого dataset ще одну колонку - DAY_TIMESTAMP_FIELD, в яку запишему дату дня покупки (відсікаючи час), використовуючи функцію date_trunc з аргументом "DAY". Далі згрупуємо дані, використовуючи отримане поле, та, в отриманих групах за кожен день, визначимо сумарну величину доходу.

Останній етап - сортировка отриманих даних в хронологічному порядку за полем DAY_TIMESTAMP_FIELD та приведення даних зі Spark до вигляду моделі у Java за допомогою метода mapRowToChartPoint.

```
private ChartPoint mapRowToChartPoint(Row row) {
    return new ChartPoint(row.getTimestamp(0) .toLocalDateTime().toLocalDate(),
                          row.getDouble(1));
}

List<ChartPoint> revenueValueChartByDay = productsWithPurchasesInfo
```

```
        .withColumn(DAY_TIMESTAMP_FIELD,
                date_trunc("DAY", col(PURCHASE_TIMESTAMP_FIELD)))
        .groupBy(col(DAY_TIMESTAMP_FIELD))
        .agg(sum(col(PURCHASE_LINE_VALUE_FIELD))
                .as(PRODUCT_TOTAL_VALUE_FIELD))
        .orderBy(col(DAY_TIMESTAMP_FIELD))
        .collectAsList()
        .stream()
        .map(this::mapRowToChartPoint)
        .collect(Collectors.toList());
```

Тепер, використовуючи аналітичний модуль org.apache.commons.math, за допомогою лінійної регресії розрахуємо очікувані значення доходу в найближчі 2 місяці.

```
// calculate regression
SimpleRegression simpleRegression = new SimpleRegression();

double[] x = new double[existingPoints.size()];
double[] y = new double[existingPoints.size()];

for (int i = 0; i < existingPoints.size(); i++) {
   simpleRegression.addData(i, existingPoints.get(i).getValue());
}

LocalDate lastExistingDate = existingPoints.get(existingPoints.size() - 1).getDate();

for (int i = 0; i < DAYS_TO_PREDICT; i++) {
   existingPoints.add(new AnalyticsReport.ChartPoint(
         lastExistingDate.plusDays(i + 1),
         simpleRegression.predict(existingPoints.size() + (double) i),
         true));
}
```

Нарешті, отримавши усі необхідні дані для створення аналітичного звіту, утворюємо його модель, заповнюючи отриманими значеннями. Для цього використаємо паттерн builder.

```
AnalyticsReport report = AnalyticsReport.builder()
    .customerCount(uniqueCustomerCount)
    .activeCustomerCount(activeCustomersCount)
    .uniqueProductCount(uniqueProductCount)
    .uniqueBoughtProductCount(uniqueBoughtProductCount)
    .boughtProductItemCount(boughtProductItemCount)
    .purchasesCount(uniquePurchasesCount)
    .uniquePurchasesInSpecifiedPeriod(uniquePurchasesInSpecifiedPeriod)
    .totalValue(totalValue)
    .averagePurchaseValue(averagePurchaseValue)
    .averagePurchaseUniqueItemsCount(averagePurchaseItemsCount)
    .customersWithStats(customersWithStats)
    .productsWithStats(productsWithStats)
    .revenueValueChartByDay(revenueValueChartByDay)
    .build();
```

Та зберігаємо до бази даних Postgres.

```
analyticsReportService.save(
      new AnalyticsReportEntity(System.currentTimeMillis(),
            request.getShopId(),
            report,
            request.getPeriodStart(),
            request.getPeriodEnd()
      )
);
```

РОЗДІЛ 5 ДЕМОНСТРАЦІЯ РЕАЛІЗОВАНИХ АЛГОРИТМІВ НА ЗГЕНЕРОВАНИХ ДАНИХ

5.1 Опис даних для тестування аналітичного додатку

Для тестування алгоритму було сгенеровано дані за 2 роки роботи магазину іграшок. Було сгенеровано 150 унікальних покупців, 60 різноманітних товарів та історію покупок за 2 роки роботи магазину.

Саме ці дані й були використані для перевірки роботи алгоритмів.

5.2 Презентація та аналіз результатів виконання

Для аналізу був вибран період останнього року роботи магазину.

В результаті проведеного аналізу, отримано такі результати.

1. Загальна кількість покупців за увесь час: 150 чол.
2. Кількість покупців за останній рік: 137 чол.
3. Кількість унікальних товарів у магазині: 60 позицій.
4. Кількість унікальних товарів, придбаних за останній рік: 60 поз.
5. Загальна кількість одниць товарів, що була придбана за останній рік: 7484 шт.
6. Кількість продаж за весь період: 1375 шт.
7. Кількість продаж за останній рік: 1246 шт.
8. Загальний дохід за останній рік: 24 743 739 грн.
9. Середня вартість однієї покупки: 6558,11 грн.
10. Середня кількість унікальних позицій (товарів) в покупці: 2 шт.

Результати кластеризації продуктів.

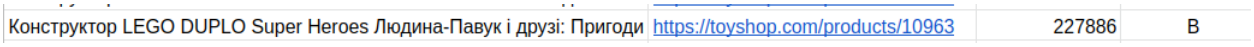
Відповідно до виконаного ABC-аналізу, виявилось, що товари розділені на сегменти таким чином:

Сегмент A - 24 позицій.

Сегмент B - 20 позицій.

Сегмент C - 16 позицій.

Нижче наведено топ 30 продуктів за сумарним доходом (рис. 5.1).



Конструктор LEGO DUPLO Super Heroes Людина-Павук і друзі: Пригоди	https://toyshop.com/products/10963	227886	B	

Рисунок 5.1 - топ 30 товарів за сумарним доходом з визначеним сегментом

Відповідно до сегментації можно зробити висновок, що товари з сегменту А необхідно закупати більше і скорочувати поставки сегменту С. Альтернативою може бути реклама продукції сегменту С.

Результати кластеризації покупців.

Відповідно до виконаного RFM-аналізу, виявилось, що покупці розділені на сегменти таким чином:

Recency:

Сегмент А - 43 позицій.

Сегмент В - 49 позицій.

Сегмент С - 48 позицій.

Frequency:

Сегмент А - 45 позицій.

Сегмент В - 49 позицій.

Сегмент С - 46 позицій.

Monetary:

Сегмент А - 46 позицій.

Сегмент В - 48 позицій.

Сегмент С - 46 позицій.

Нижче наведено топ 30 покупців за сумарними витратами на покупки (рис. 5.2).

Відповідно до сегментації можно приймати бізнес рішення. 13 користувачів, що зробили покупку в 2 роки тому, не зробили її у минулому. Таким користувачам можна відправити email зі скидкою на продукцію, аби спробувати повернути користувача у список активних. Для користувачів з сегментів А і В можна розробити систему бонусів за регулярні покупки, а користувачів з сегменту С - залучати через e-mail листування.

Ксенія	Лесько	kseniialesko@gmail.com	262525	10	B	B	A

Рисунок 5.2 - топ 30 покупців за сумарними витратами з визначеними сегментами

Також, приведемо результати регресивної моделі по доходу за день (рис. 5.3).

В графіку представлена залежність доходу від часу.

Пряма лінія в кінці представляє собою очікуваниу динаміку доходу в наступні 2 місяці.

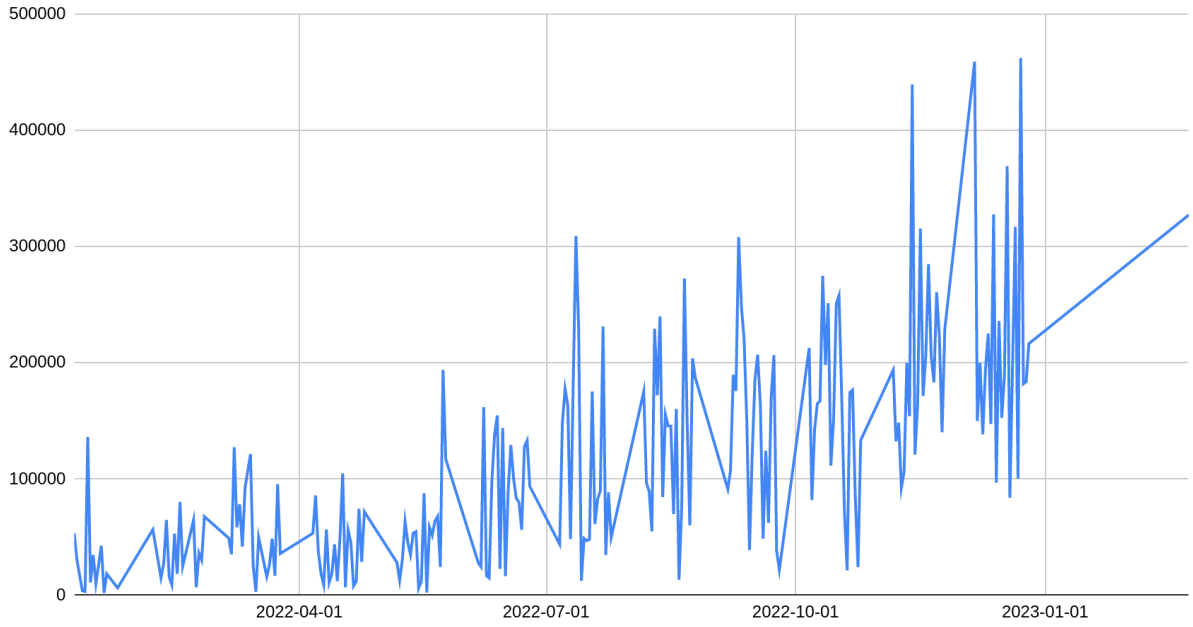


Рисунок 5.3 - Графік доходу онлайн-магазину іграшок за останій рік по днях. Останні два місяці - передбачення.

ВИСНОВКИ

Аналітичні інформаційні системи дозволяють обробляти велику кількість даних, загострюючи увагу лише на ключових факторах ефективності, моделюючи результат різних варіантів дій, відстежуючи результати прийняття рішень. (ir.kneu.edu.ua) Такі системи підтримують багато бізнес-рішень — від операційних до стратегічних.

Тема аналітичних інформаційних систем буде залишатися актуальною, як і теми способів аналітики даних, наприклад, дата-майнинг.

В ході роботи були вирішені наступні завдання:

1. Виконано аналіз предменої області.
2. Проведено дослідження теоретичних основ, сучасних методик та підходів в сфері дата-майнингу.
3. Проаналізовано та дана оцінка сучасному стану сфери послуг онлайн-аналітики.
4. Створено додаток на мікросервісній основі, що генерує аналітичні звіти за визначений період, оперуючи масивами даних про покупців, товари та покупки у онлайн-магазині.
5. На основі існуючих методів та моделей дата-майнингу було розроблено алгоритми для розрахунку важливих метрик діяльності магазину.
6. Реалізована модель кластеризації продуктів магазину відповідно до ABC-аналізу та модель кластеризації покупців відповідно до RFM-аналізу.
7. Реалізовано регресивну модель для прогнозування доходу магазину у майбутні періоди.
8. Було представлено роботу алгоритму на масиві згенерованих даних.
9. Виконано аналіз отриманих результатів.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Брантон С.Л. Аналіз даних в науці та техніці / С.Л. Брантон, Дж.Н.Куц - 2021 - 542 с.

2. Барсегян А. А., Технологии анализа данных. Data Mining, Visual Mining, Text Mining, OLAP (tekhnosfera.com) / А. А. Барсегян, М. С. Куприянов, В. В. Степаненко, И. И. Холод - 2008. - 384 с.

3. Тоби Сегаран. Программируем коллективный разум / Тоби Сегаран - Вид.: Символ-Плюс, 2012 - 368 с.

4. Том Фоусет. Data Science для бізнесу. Як збирати, аналізувати і використовувати дані / Том Фоусет, Фостер Провост - Видавництво: Наш формат, 2019.

5. В. А. Дюк. Применение технологий интеллектуального анализа данных в естественнонаучных, технических и гуманитарных областях (ela.kpi.ua) / В. А. Дюк, А. В. Флегонтов, И. К. Фомина - 2011.

6. О. С. Коваленко, Обзор проблем и перспектив анализа данных.

7. А. А. Ежов, С. А. Шумский, Лекция: Извлечение знаний с помощью нейронных сетей.

8. Стаття Microsoft SQL Server 2008 R2: новый подход к управлению информацией - 2014.

9. Стаття Data Mining от Oracle: настоящее и будущее - 2012.

10. Стаття Степанов Р. Г. Технология Data Mining: Интеллектуальный Анализ Данных - 2017.

11. Н. Паклин Бизнес-аналитика. От данных к знаниям / Вид: Питер - 2013, 706 с.

12. Ian (doi.org) H. Witten. Data Mining: Practical Machine Learning Tools and Techniques (The Morgan Kaufmann Series in Data Management Systems) (doi.org) / Ian H. Witten, Eibe Frank, Mark A. Hall (www.cse.hcmut.edu.vn) - 2011, 664 с.

13. Методы и модели анализа данных: OLAP и Data Mining (piglos.ru) Арменак Барсегян, М. С. Куприянов, В. В. Степаненко, И. И. Холод - Вид: БХВ-Петербург - 2004, 336 с.

14. Григорий Пятецкий-Шапиро, Data Mining и перегрузка информацией // Вступительная статья к книге: Анализ данных и процессов /

А. А. Барсегян, М. С. Куприянов, И. И. Холод, М. Д. Тесс, С. И. Елизаров. 3-е изд. перераб. и доп. СПб.: БХВ-Петербург, 2009. 512 с. С. 13.

15. Обсуждаем термин: data mining / Школа технического перевода - 2014.

16. Миллнер, Хан. Переход к Big Data - 2022.

17. Сайт Oracle, [Електронний ресурс] / Режим доступу: https://docs.oracle.com/en/.

18. Эккель Б. Философия Java. 4-е полное изд. — СПб.: Питер, 2018. — 1168 с.: ил. — (Серия «Классика computer science»).

19. Sonatype Company – Maven: The Definitive Guide / O'Reilly Media, Inc., 2008.

20. Сайт Wikipedia (Опис регресії найменших квадратів) [Електронний ресурс] / Режим доступу: https://en.wikipedia.org/wiki/Least_squares.

21. IEEE Std. 1471-2000. Recommended Practice for Architectural Description of Software-Intensive Systems. (cdn.scipeople.com) [Електронний ресурс]. – Режим доступу: http://cabibbo.dia.uniroma3.it/ids/altrui/ieee1471.pdf.

22. Bass, Len, Kazman, Rick і Clements, Paul. Software Architecture in Practice. Third Edition. Boston: Addison-Wesley Professional - 2012, 624 с..

23. Fowler Chad. From Homogeneous Monolith to Heterogeneous Microservices Architecture / Fowler Chad - 2015.

24. Microservices a definition of this new architectural term. Fowler, Martin. [Електронний ресурс]. – Режим доступу: https://bitly.su/9LVBolxX, вільний. – Загл. з екрану.

25. Going to Market Faster: Most Companies Are Deploying Code Weekly, Daily, or Hourly. (de.wikipedia.org) [Електронний ресурс]. – Режим доступу: https://blog.newrelic.com/technology/data-culture-survey-results-faster-deployment, вільний. – Загл. з екрану.

26. Fairbanks, George. Just Enough Software Architecture. б.м.: Marshall & Brainerd 2010.